EDB
Postgres® for the AI Generation

Postgres Enterprise Manager

Release 9.8.0

EnterpriseDB Corporation

Dec 03, 2024

Contents

1 Introduction **3**
- 1.1 Postgres compatibility . . . 3
- 1.2 Release notes . . . 3
  - 1.2.1 Postgres Enterprise Manager 9.8.0 release notes . . . 4
  - 1.2.2 Postgres Enterprise Manager 9.7.0 release notes . . . 5
  - 1.2.3 Postgres Enterprise Manager 9.6.0 release notes . . . 6
  - 1.2.4 Postgres Enterprise Manager 9.5.1 release notes . . . 6
  - 1.2.5 Postgres Enterprise Manager 9.5.0 release notes . . . 7
  - 1.2.6 Postgres Enterprise Manager 9.4.1 release notes . . . 7
  - 1.2.7 Postgres Enterprise Manager 9.4.0 release notes . . . 8
  - 1.2.8 Postgres Enterprise Manager 9.3.1 release notes . . . 8
  - 1.2.9 Postgres Enterprise Manager 9.3.0 release notes . . . 8
  - 1.2.10 Postgres Enterprise Manager 9.2.2 release notes . . . 9
  - 1.2.11 Postgres Enterprise Manager 9.2.1 release notes . . . 9
  - 1.2.12 Postgres Enterprise Manager 9.2.0 release notes . . . 10
  - 1.2.13 Postgres Enterprise Manager 9.1.1 release notes . . . 11
  - 1.2.14 Postgres Enterprise Manager 9.1.0 release notes . . . 11
  - 1.2.15 Postgres Enterprise Manager 9.0.0 release notes . . . 11
- 1.3 Platform compatibility . . . 12
- 1.4 Hardware requirements for installing the PEM server on Linux . . . 12

2 Planning **13**
- 2.1 Understanding Postgres Enterprise Manager components and architecture . . . 13
  - 2.1.1 PEM architecture . . . 14
- 2.2 Deployment considerations . . . 18
  - 2.2.1 Using Failover Manager for high availability . . . 18
  - 2.2.2 Connection pooling using PgBouncer . . . 27
  - 2.2.3 Authentication options . . . 37
  - 2.2.4 Securing your deployment . . . 49
  - 2.2.5 Web server installation options . . . 58
  - 2.2.6 How PEM licensing works . . . 59

3 Installing and Upgrading **61**
- 3.1 Installing Postgres Enterprise Manager server . . . 61
  - 3.1.1 Linux x86-64 (amd64) . . . 61
  - 3.1.2 Linux Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le) . . . 62

3.1.3 Linux Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64) 62
3.1.4 Windows 62
3.2 Installing Postgres Enterprise Manager agent 106
3.2.1 Linux x86-64 (amd64) 106
3.2.2 Linux Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le) 107
3.2.3 Linux Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64) 107
3.2.4 Windows 107
3.3 Upgrade and migration 120
3.3.1 Upgrading a PEM installation 121
3.3.2 Upgrading the PEM backend Postgres database 125
3.3.3 Adding entries for PEM Agens and admins to connect to PEM server 133
3.3.4 Adding entries (localhost) for PEM Agens and admins to connect to PEM server 133
3.3.5 Upgrading SQL Profiler 133
3.3.6 Upgrading Apache HTTPD server 134
3.3.7 Moving the PEM server 135
3.4 Uninstalling 143
3.4.1 Uninstalling Postgres Enterprise Manager components on Linux 143
3.4.2 Uninstalling Postgres Enterprise Manager components on Windows 144
3.5 Troubleshooting 144
3.5.1 Server installation errors 144
3.5.2 Reconfiguring the PEM server 146
3.5.3 PEM web client not loading 146
3.5.4 Error connecting to PostgreSQL server 147
3.5.5 PEM web server status check 147

4 Administering **149**
4.1 Changing the default port 149
4.1.1 For RHEL 149
4.1.2 For Debian and Ubuntu 150
4.2 Registering a Postgres server 150
4.2.1 What does registering a Postgres server do? 150
4.2.2 Using the pemworker utility to register a server 151
4.2.3 Registering a server with automatic server discovery 153
4.2.4 Manually registering a database server 154
4.2.5 Verifying the connection and binding 160
4.3 Registering a PEM agent 160
4.3.1 How to register PEM agents 160
4.3.2 Registering a PEM agent using the pemworker utility 160
4.3.3 Unregistering a PEM agent 162
4.3.4 Advanced usage 162
4.4 Managing a PEM server 165
4.4.1 Starting and stopping the PEM server and agents 166
4.4.2 Controlling the PEM server or PEM agent on Linux 166
4.4.3 Controlling the PEM server or PEM agent on Windows 166
4.4.4 Controlling the HTTPD server 167
4.4.5 Managing PEM authentication 167
4.4.6 Editing the PEM server configuration 168
4.4.7 Managing security 168
4.4.8 Server configuration 171
4.4.9 Server configuration parameters - reference 171
4.5 Managing Postgres servers 179
4.5.1 Starting and stopping monitored database servers 179
4.5.2 Modifying the pg_hba.conf file 180
4.5.3 Creating and maintaining databases and objects 181

4.5.4 Template Linux service script . . . 181
4.6 Managing a PEM agent . . . 182
4.6.1 Managing job notifications . . . 182
4.6.2 Managing scheduled jobs . . . 183
4.6.3 Modifying agent configuration . . . 186
4.6.4 Reviewing or modifying agent properties . . . 190
4.6.5 Setting agent privileges . . . 191
4.7 Managing SSL certificates . . . 194
4.7.1 Web-server certificates . . . 195
4.7.2 PEM backend database server and agent connection certificates . . . 195
4.7.3 Generate a common agent certificate and key pair . . . 196
4.7.4 Use certificates and keys signed by trusted CA . . . 198
4.7.5 Testing certificates . . . 200
4.8 Managing configuration settings . . . 204
4.9 Troubleshooting agent issues . . . 204
4.9.1 Restoring a deleted PEM agent . . . 205
4.9.2 Updating configuration file path (agent status displaying as unknown) . . . 205
4.9.3 Using the command line to delete a PEM agent with down or unknown status . . . 205

5 Using 207
5.1 Accessing the web interface . . . 207
5.1.1 Accessing in-app help for UI details . . . 209
5.1.2 PEM client object browser . . . 209
5.1.3 PEM tabbed browser window . . . 210
5.1.4 Using chart, graph, and table controls . . . 212
5.1.5 Browser toolbar . . . 213
5.1.6 PEM menu bar . . . 213
5.1.7 Controlling and customizing charts, graphs, and tables . . . 216
5.2 Connecting and disconnecting a database server . . . 217
5.2.1 Disconnecting from a server . . . 217
5.3 Monitoring performance . . . 217
5.3.1 Probes . . . 220
5.3.2 Alerts . . . 237
5.3.3 Notifications . . . 257
5.3.4 Capacity Manager . . . 266
5.3.5 Audit Manager . . . 271
5.3.6 Log Manager . . . 275
5.3.7 Charts . . . 287
5.3.8 Dashboards . . . 290
5.3.9 Remote monitoring . . . 296
5.4 Monitoring Barman . . . 297
5.4.1 Prerequisites for monitoring Barman . . . 297
5.4.2 Configuring a Barman server . . . 297
5.4.3 Viewing the Barman server details on a PEM dashboard . . . 299
5.5 Monitoring EDB Postgres Distributed . . . 300
5.6 Monitoring Failover Manager . . . 300
5.6.1 Replacing a primary node . . . 301
5.6.2 Switchover EFM cluster . . . 302
5.7 Monitoring Replication Server . . . 302
5.8 Monitoring event history . . . 302
5.9 Tuning performance . . . 303
5.9.1 Tuning Wizard . . . 303
5.9.2 Postgres Expert . . . 305
5.9.3 Performance Diagnostic . . . 307

5.10 Profiling workloads . . . 311
5.10.1 Installing the SQL Profiler extension . . . 311
5.10.2 Using SQL Profiler . . . 313
5.10.3 Using the Index Advisor . . . 315
5.11 Viewing system reports . . . 316
5.11.1 Alert History report . . . 316
5.11.2 System Configuration report . . . 316
5.11.3 Core Usage report . . . 317
5.12 PEM command line interface . . . 317
5.12.1 Example . . . 320
5.13 PEM Rest API . . . 320

6 Developing **321**
6.1 Using the Query tool . . . 321
6.1.1 The Query tool toolbar . . . 322
6.1.2 The SQL editor panel . . . 324
6.1.3 The Data Output panel . . . 324
6.1.4 Query History panel . . . 328
6.1.5 Connection status . . . 328
6.1.6 Macros . . . 328
6.1.7 Accessing in-app help for UI details . . . 329
6.2 Using the Schema Diff tool . . . 329
6.2.1 Schema Diff Object Comparison panel . . . 329
6.2.2 Schema Diff DDL Comparison panel . . . 330
6.3 Using the ERD tool . . . 330
6.3.1 Toolbar . . . 331
6.3.2 File options . . . 331
6.3.3 Export options . . . 332
6.3.4 Editing options . . . 332
6.3.5 Table relationship options . . . 332
6.3.6 Utility Options . . . 332
6.3.7 Zoom options . . . 333
6.3.8 Table dialog . . . 333
6.3.9 Table node . . . 333
6.3.10 One-to-many link dialog box . . . 333
6.3.11 Many-to-many link dialog box . . . 334
6.3.12 Table link . . . 334
6.3.13 Table notes . . . 335

Welcome to Postgres Enterprise Manager (PEM). PEM consists of components that provide the management and analytical functionality for your EDB Postgres Advanced Server or PostgreSQL database. PEM is based on the Open Source pgAdmin 4 project.

PEM is a comprehensive database design and management system. PEM is designed to meet the needs of both novice and experienced Postgres users alike, providing a powerful graphical interface that simplifies the creation, maintenance, and use of database objects.

CHAPTER 1

Introduction

1.1 Postgres compatibility

Supported versions of Postgres for PEM 9.x:

	Monitored Instance	Backend Instance
EDB Postgres Advanced Server (EPAS)	12, 13, 14, 15, 16, 17	12, 13, 14, 15, 16, 17
PostgreSQL (PG)	12, 13, 14, 15, 16, 17	12, 13, 14, 15, 16, 17
EDB Postgres Extended Server (PGE)	12, 13, 14, 15, 16, 17	13, 14, 15, 16, 17[^1]

1.2 Release notes

The Postgres Enterprise Manager (PEM) documentation describes the latest version of PEM 9 including minor releases and patches. The release notes in this section provide information on what was new in each release. For new functionality introduced after 8.0.0 in a minor or patch release, there are also indicators within the content about what release introduced the feature.

Version	Release Date	Upstream Merges	Accessibility Conformance
Postgres Enterprise Manager 9.8.0 release notes	26 Nov 2024	NA	Conformance Report
Postgres Enterprise Manager 9.7.0 release notes	13 Aug 2024	NA	Conformance Report
Postgres Enterprise Manager 9.6.0 release notes	15 May 2024	NA	Conformance Report
Postgres Enterprise Manager 9.5.1 release notes	04 Mar 2024	NA	Conformance Report

continues on next page

Table 2 – continued from previous page

Version	Release Date	Upstream Merges	Accessibility Conformance
Postgres Enterprise Manager 9.5.0 release notes	15 Feb 2024	NA	Conformance Report
Postgres Enterprise Manager 9.4.1 release notes	28 Nov 2023	NA	Conformance Report
Postgres Enterprise Manager 9.4.0 release notes	16 Nov 2023	NA	Conformance Report
Postgres Enterprise Manager 9.3.1 release notes	12 Oct 2023	NA	Conformance Report
Postgres Enterprise Manager 9.3.0 release notes	31 Aug 2023	NA	Conformance Report
Postgres Enterprise Manager 9.2.2 release notes	14 Jul 2023	NA	Conformance Report
Postgres Enterprise Manager 9.2.1 release notes	03 Jul 2023	NA	Conformance Report
Postgres Enterprise Manager 9.2.0 release notes	24 May 2023	NA	Conformance Report
Postgres Enterprise Manager 9.1.1 release notes	31 Mar 2023	NA	Conformance Report
Postgres Enterprise Manager 9.1.0 release notes	14 Feb 2023	NA	Conformance Report
Postgres Enterprise Manager 9.0.0 release notes	9 Dec 2022	NA	Conformance Report

Often only select issues are included in the upstream merges. The specific issues included in the merges are listed in the release note topics.

1.2.1 Postgres Enterprise Manager 9.8.0 release notes

Released: 26th Nov 2024

Note: This release includes a fix for a bug that previously prevented the correct purging of historical data when the configured retention time expired. As a result, the first purge job to run after this update is expected to delete more data than usual and may take longer to complete.

The purge job should begin within three hours of the upgrade. Additionally, after this job and the subsequent vacuuming process are finished, you may notice a significant reduction in the total size of your PEM database.

Type	Description
Enhancement	Added support for EDB Advanced Server, EDB Postgres Distributed, and PostgreSQL 17.
Enhancement	Added support for RHEL9 ARM64 platform.
Enhancement	Added support to allow sslmode connections from PEM agent to PEM server.
Bug fix	Fixed an issue for PEM port while registering the server. Now *pemworker* picks up the PEM port from the agent config file while registering the server.
Bug fix	Fixed an issue where PEM agent installation on Windows was failing if a % character was used to set the service for the agent.

continues on next page

Table 3 – continued from previous page

Type	Description
Bug fix	Fixed an issue where some historical probe data was not purged because the system job that tracks data for purging would be erroneously deleted by another system job.
Bug fix	Fixed an issue for displaying errors on EINTR signal. Now spoolers *select()* error will be displayed on EINTR signal only when log level is set to INFO.

1.2.2 Postgres Enterprise Manager 9.7.0 release notes

Released: 13th Aug 2024

New features, enhancements, bug fixes, and other changes in PEM 9.7.0 include:

Type	Description
Enhancement	Added PGD extension details in the Core usage report.
Enhancement	If max_connections is set to 1 in the *agent.cfg* file, PEM agent startup process makes it to 2 in the backend code to avoid PEM's probe data collection failure. [Support ticket: #101924]
Enhancement	Probes are now re-run on restart of agent and monitored server.
Enhancement	Introduced v12 REST API to fix the API schema for alerts (GET) and alert templates (GET/PUT/POST).
Bug fix	Fixed an issue where the version information was missing in the *pemworker* utility usage.
Bug fix	Fixed an issue with *reminder_notification_interval* setting. Now you can configure this parameter to a value less than 1 hour.
Bug fix	Fixed an issue for webhook notification settings while copying the alerts. Now the webhook notification settings are copied while copying the alerts.
Bug fix	Fixed an issue with the mismatched version label in REST API version 11.
Bug fix	Fixed an issue with the PGD alerts. Now disabling the PGD probes disables the PGD alerts as well.
Bug fix	Fixed an issue while changing the Job type from Batch to SQL and vice-versa. Now it allows to change the job type from Batch to SQL and vice-versa without any errors.
Bug fix	Updated the Servers Down and Agents Down alert template by adding detailed information in the SQL.
Bug fix	Updated the help text of the alert script execution. The help text now says that the scrip execution code must be in batch/shell format and not in SQL.
Bug fix	Fixed an issue with the *current_pid*. Now the *current_pid* is not null even if the *agent_id* is null in the *probe_schedule* table.
Bug fix	Fixed an issue with messages for SQL profiler permission errors.
Bug fix	Fixed an issue with the *user_info* probe to get the exact password expiry date.
Bug fix	Fixed an issue with the ServerHeartbeat. The ServerHeartbeat is released once DB server is deleted from PEM UI.
Bug fix	Fixed an issue with capacity manager. The capacity manager's option *report type* now automatically selects HTML when *New Tab* and *Previous Tab* are selected.

1.2.3 Postgres Enterprise Manager 9.6.0 release notes

Released: 15 May 2024

New features, enhancements, bug fixes, and other changes in PEM 9.6.0 include:

Type	Description
Enhancement	Added an option to enable probes using the pemworker utility.
Enhancement	PEM now includes the error message in email notifications of a failed job. The default character limit for error output is 1000, which can be configured now using the *job_notification_error_limit* parameter in the server configuration.
Enhancement	Added two alert templates (*Inactive replication slots* and *Conflicting replication slots*) and two probes (*Replication slots* and *Multixact ID Exhaustion (Wraparound)*).
Enhancement	The default display name for a newly registered agent is now the host's fully qualified domain name (FQDN), falling back to the previous default of the hostname if the FQDN isn't set.
Enhancement	Updated the alert template *Table size in server/database/schema* with more detailed SQL query.
Enhancement	Added an option to download the Capacity Manager report in JSON format.
Enhancement	You can now specify a CA Certificate path when registering a Barman server using pemworker without having to specify a client certificate and key.
Bug fix	Fixed an issue where audit manager was setting *edb_audit_statement* parameter to '' instead of 'none' when log statement parameter was left empty in the GUI resulting in failure of server restart.
Bug fix	Fixed an issue where *os_info* probe was throwing error whenever timezone/timestamp were changed to some specific values.
Bug fix	Fixed an issue whereby an error "*NoneType* has no *len()*" occurred when alerts were listed on the alert dashboard.
Bug fix	Fixed an issue where team role wasn't working with PEM agents as expected.
Bug fix	Fixed an issue with the SQL query for the PEM agent registration. Added the public schema name in the table prefix to execute the SQL query successfully even if the schema name isn't in the search path.

1.2.4 Postgres Enterprise Manager 9.5.1 release notes

Released: 04 Mar 2024

New features, enhancements, bug fixes, and other changes in PEM 9.5.1 include:

Type	Description
Security fix	This is a security fix for CVE-2023-50782 . Upgraded bundled Python cryptography package to 42.x.

1.2.5 Postgres Enterprise Manager 9.5.0 release notes

Released: 15 Feb 2024

New features, enhancements, bug fixes, and other changes in PEM 9.5.0 include:

Type	Description
Enhancement	Added *EmailGroup* and *EmailGroupID* in webhook payload.
Enhancement	Added an alert template to validate password expiry for users in EDB Postgres Advanced Server. Also modified the alert template *A user expired in N days* to *Number of users expiring in N days* that checks the number of users expiring.[^*]
Enhancement	Added a button on the Manage Probes window that resets the probe configurations to the defaults.
Enhancement	Added a **Cleared Alert** button in notifications. This gives you the flexibility to disable notifications for cleared alerts while still receiving notifications when alerts occur.
Enhancement	You can now create your own web server certificates, as EDB details are no longer added to the web server certificates generated by PEM.
Enhancement	Added a capability to identify the user who changed the blackout status, date, and time.
Bug fix	Fixed an issue where a deleted custom probe was still visible in the *Probe Dependency* tab of Alert Templates.
Bug fix	Fixed a *missing permission* issue by granting the *pem_agent* user permission to the *custom_email_template* table.
Bug fix	Excluded zero-sized tables from PEM alert template *largest index by table-size percentage*. Also added a detailed SQL statement for this template.
Bug fix	Fixed an issue whereby the CLI attempted to call an obsolete version of the PEM API, resulting in commands failing.
Bug fix	Now, the members of the *pem_user* and *pem_manage_efm* roles can access the EFM features. The member of the *pem_user* and `pem_server_service_manager`role can start/stop the database service from PEM GUI.
Bug fix	Added *application_name* column in the *session_info* probe.
Bug fix	Fixed the *too few arguments* exception for the webhooks.
Bug fix	Resolved the issue where the User column displayed the current username logged into the PEM user interface in the chart display of the *pg_hba.conf* file. Now it displays the respective usernames in the *pg_hba.conf* file.
Bug fix	Fixed a *Dashboard info not found* error while opening the dashboards in the user interface.

1.2.6 Postgres Enterprise Manager 9.4.1 release notes

Released: 28 Nov 2023

New features, enhancements, bug fixes, and other changes in PEM 9.4.1 include:

Type	Description
Security fix	This is a security fix for CVE-2023-38545 and CVE-2023-38546 . This patch fixes the libcurl-pem issues.

1.2.7 Postgres Enterprise Manager 9.4.0 release notes

Released: 16 Nov 2023

New features, enhancements, bug fixes, and other changes in PEM 9.4.0 include:

Type	Description
Enhancement	Added support for EDB Advanced Server, EDB Postgres Distributed, and PostgreSQL 16.
Enhancement	PEM jobs can now be scheduled by way of the REST API.
Enhancement	Improved the performance diagnostics to provide aggregated statistics over the time interval.
Bug fix	Added schema qualifications for the *process_one_alert* and *get_detail_alert_info* functions.
Bug fix	Added the optional ICU parameters required for creating databases in PostgreSQL versions 15 and 16.
Bug fix	Added the frontend validations for empty agent details while adding the Barman server.
Bug fix	Fixed an issue when downloading the system config report. The system config report didn't download if any agent had some missing parameter or was in an unknown state.
Bug fix	Added a server configuration parameter *deleted_objects_data_retention_time* to improve purging of the obsolete data for server, agent, or tools.

1.2.8 Postgres Enterprise Manager 9.3.1 release notes

Released: 12 Oct 2023

New features, enhancements, bug fixes, and other changes in PEM 9.3.1 include:

Type	Description
Security fix	This is a security fix for CVE-2023-5002 . This security fix includes updates for the pgAdmin vulnerability issues. This patch affects the PEM server only, no need to update PEM agents.
Security fix	This is a security fix for CVE-2023-4863 . This security fix includes updates for the libwebp issues. This patch affects the PEM server only, no need to update PEM agents.

1.2.9 Postgres Enterprise Manager 9.3.0 release notes

Released: 31 Aug 2023

New features, enhancements, bug fixes, and other changes in PEM 9.3.0 include:

Type	Description
Enhancement	Added the functionality to delete all the existing alerts while using the copy alert feature.

continues on next page

Table 11 – continued from previous page

Type	Description
Enhancement	Added PostgreSQL and EDB Postgres Advanced Server 16 beta support as a backend database server and monitoring server.
Enhancement	Added support to customize the default email templates.
Enhancement	Added support to download the alert history report for agents and servers.
Enhancement	Added options to reuse the existing agent SSL certificate and key file while configuring the PEM server.
Bug Fix	Added the functionality to add/view ignore mount points in the agent REST API.
Bug Fix	Added the *hostaddr* body parameter to the server REST API.
Bug Fix	Fixed an issue by removing the newline character in the alert details from the payload of Webhook.
Bug Fix	Added support to allow the Webhook endpoint registration without providing the client SSL certificates and key files or CA certificates using *allow_insecure_webhooks* parameter in the *agent.cfg* file.
Bug Fix	Fixed an issue related to the backup sizes displaying on the PEM Barman dashboard.
Bug Fix	Fixed an issue with unformatted data while importing the audit logs.

1.2.10 Postgres Enterprise Manager 9.2.2 release notes

Released: 14 Jul 2023

New features, enhancements, bug fixes, and other changes in PEM 9.2.2 include:

Type	Description
Security fix	This is a security fix for CVE-2023-2650 and is recommended for all the Windows users. This security fix includes updates for the Apache HTTPD bundled with the Windows installer of PEM. This patch affects the PEM server only, no need to update PEM agents.

1.2.11 Postgres Enterprise Manager 9.2.1 release notes

Released: 03 Jul 2023

New features, enhancements, bug fixes, and other changes in PEM 9.2.1 include:

Type	Description
Security fix	Added support for Flask 2.x. This is a security fix for CVE-2023-30861 and is recommended for all users. This patch affects the PEM server only, no need to update PEM agents.

1.2.12 Postgres Enterprise Manager 9.2.0 release notes

Released: 24 May 2023

New features, enhancements, bug fixes, and other changes in PEM 9.2.0 include:

Type	Description
Enhancement	The default sample end time for performance diagnostics is now **Now** rather than the time the window was first opened. The data now updates when you select **Run** .
Bug fix	Fixed the Replication Time Lag graph status issue on the Streaming Analysis dashboard. This fix displays the status as stopped if the server is down/offline. [Support ticket: #91220]
Bug fix	Removed the agent status from Failover Manager dashboard under *efm_cluster_node_status*. [Support ticket: #1228607]
Bug fix	Fixed an *unknown error* issue while updating/creating the server from the PEM web client.
Bug fix	Fixed an issue to return a valid message for invalid parameters and removed unwanted parameters in PUT server REST API.
Bug fix	Updated cryptography package to the secured version.
Bug fix	Updated webpack(NPM) package to the secured version.
Bug fix	Fixed an error caused by an increase/decrease in the number of parameters when changing the alert templates.
Bug fix	Fixed the *Static object initialization fiasco* issue caused while initializing the BARMAN singleton object at the startup.
Bug fix	Fixed the *Login Group/Roles 500 error: UNION types char and text cannot be matched* error. This error was appearing while trying to open the dependents tab for any role with PostgreSQL or EDB Postgres Advanced Server 15 as the backend server.
Bug fix	Provided helpful message instead of default Python error if a token is invalid. [Support ticket: #89276]
Bug fix	Fixed extension name warning while creating a probe. This warning was appearing even after providing a valid extension name.
Bug fix	Fixed the error in the *--config-file* switch while trying to register the Barman server in PEM using the command line interface.
Bug fix	Fixed the missing favicon icon for the help docs and API link.
Bug fix	Fixed the warning message while deleting a folder in the Manage Template section of Capacity Manager.
Bug fix	Fixed the tuning wizard report whereby the number of servers selected didn't match the servers displayed in the **GoTo** list.
Bug fix	Added client-side validation for alert creation for parameters in alert templates.
Bug fix	Added validations for date/time empty entries in *Log Analysis Expert*.
Bug fix	Fixed an issue whereby the chart didn't update after the period was changed from **Historical days and threshold** to **Start time and end time** .
Bug fix	Added validations for empty log rotation entries in Audit Manager.
Bug fix	Added the missing extension-level metrics while creating chart.
Bug fix	Removed the parameter validation for the system alerts.
Bug fix	Fixed an issue whereby PEM was overwriting the user-supplied certificate and key during agent registration. [Support ticket: #90841]

1.2.13 Postgres Enterprise Manager 9.1.1 release notes

Released: 31 Mar 2023

New features, enhancements, bug fixes, and other changes in PEM 9.1.0 include:

Type	Description
Bug fix	Fixed an issue whereby an error would occur when registering a PEM agent on Ubuntu 22.04.

1.2.14 Postgres Enterprise Manager 9.1.0 release notes

Released: 14 Feb 2023

New features, enhancements, bug fixes, and other changes in PEM 9.1.0 include:

Type	Description
Enhancement	Added support for EDB Advanced Server 15 and EDB Postgres Distributed 5.
Enhancement	Added the ability to copy probes and alerts to all servers in a group without having to select them individually.
Bug Fix	Fixed an issue whereby alert notifications were sent for Low or Medium alerts when the user disabled the option to send them. [Support Ticket #87664]
Bug Fix	Fixed an issue whereby webhooks weren't called if there was an inactive endpoint in the spool. [Support Ticket #87365]
Other	The PGD Worker Errors probe and associated alert template and dashboard chart were removed as they produced incorrect data. They will be replaced with a new probe in a future release.

1.2.15 Postgres Enterprise Manager 9.0.0 release notes

Released: 09 Dec 2022

New features, enhancements, bug fixes, and other changes in PEM 9.0.0 include:

Type	Description	ID
Enhancement	Improved layout of the user interface and updated with new branding.	PEM-4472
Enhancement	Allow users to specify agent-id and existing SSL certificate and key files from the command line during agent registration to improve the user experience and use of the trusted certificates.[Support Ticket #1022060]	PEM-3546
Enhancement	Added a REST API to create a server group. [Support Ticket #78430]	PEM-4403

continues on next page

Table 17 – continued from previous page

Type	Description	ID
Enhancement	Revalidated, corrected and clarified the OpenAPI documentation for the REST API. [Support Ticket #84746]	PEM-4453
Enhancement	Allow use of Performance Diagnostics without a superuser account. [Support Ticket #82569]	PEM-4515
Enhancement	Added support for the PostgreSQL 15 as the backend database server for Postgres Enterprise Manager.	PEM-4558
Enhancement	Generated the SQLProfiler plugin for PostgreSQL 15.	PEM-4599
Bug Fix	Fixed an issue whereby *number of lagging replicas* metrics would return incorrect results in certain topologies.	PEM-4631
Bug Fix	Fixed an issue whereby only 50 sessions could be viewed in Dashboard > Server Activity. [Support Ticket #82563]	PEM-4492
Bug Fix	Fixed the Log Manager issue where it was creating a server reload job instead of server restart when logging_collector is ON.	PEM-4593

1.3 Platform compatibility

For information about the platforms and versions supported by PEM, see Platform Compatibility .

Note: Postgres Enterprise Manager 8.3 and later is supported on SLES.

1.4 Hardware requirements for installing the PEM server on Linux

For optimum performance when monitoring servers and rendering dashboards, we recommend installing PEM on a system with at least:

- 4 CPU cores
- 8 GB of RAM
- 100 GB of storage

Additional disk space is required for data storage. Resource use varies based on the probes that are defined and enabled and the activity level on the monitored databases. Monitoring server resources as you use PEM lets you know when you need to expand your initial system configuration.

CHAPTER 2

Planning

2.1 Understanding Postgres Enterprise Manager components and architecture

Postgres Enterprise Manager (PEM) monitors and manages multiple Postgres servers through a single graphical interface. PEM can monitor the following areas of the infrastructure:

- **Hosts** — One or more servers (physical or virtual) and their operating systems.
- **Database servers** — One or more instances of PostgreSQL, EDB Postgres Advanced Server, or EDB Postgres Extended Server (formerly known as 2ndQPostgres) running on a host.
- **Databases** — One or more databases and their schema objects, such as tables and indexes.

Note: The term Postgres refers to PostgreSQL, EDB Postgres Advanced Server, or EDB Postgres Extended Server.

PEM consists of individual software components:

- **PEM server** — The PEM server is the data repository for monitoring data and a server to which agents and clients connect. The PEM server consists of an instance of PostgreSQL, an associated database for storing monitoring data, and a server that provides web services.
- **PEM agent** — The PEM agent is responsible for executing tasks and reporting statistics from the agent host and the monitored Postgres instances to the PEM server. A single PEM agent can monitor multiple installed instances of Postgres that reside on the same host where the agent is installed or on multiple remote hosts. You can install an agent as part of the PEM server installation or you can install a stand-alone agent.
- **PEM web client** — The PEM web interface allows you to manage and monitor Postgres servers and use PEM extended functionality. The web interface software is installed with the PEM server and is accessed using any supported web browser.
- **SQL Profiler** — SQL Profiler is a Postgres server extension to record the monitoring data and query plans for the SQL Profiler tool to analyze in PEM. This is an optional component of PEM, but the extension must be enabled in each instance of Postgres for which you want to use it. You can use the SQL Profiler with any supported

version of an EDB distribution of Postgres, not just those managed through the PEM server. See *Installing the SQL Profiler extension* for details and supported versions.

2.1.1 PEM architecture

The following architectural diagram shows the relationships between the PEM server, clients, and managed and unmanaged Postgres servers.

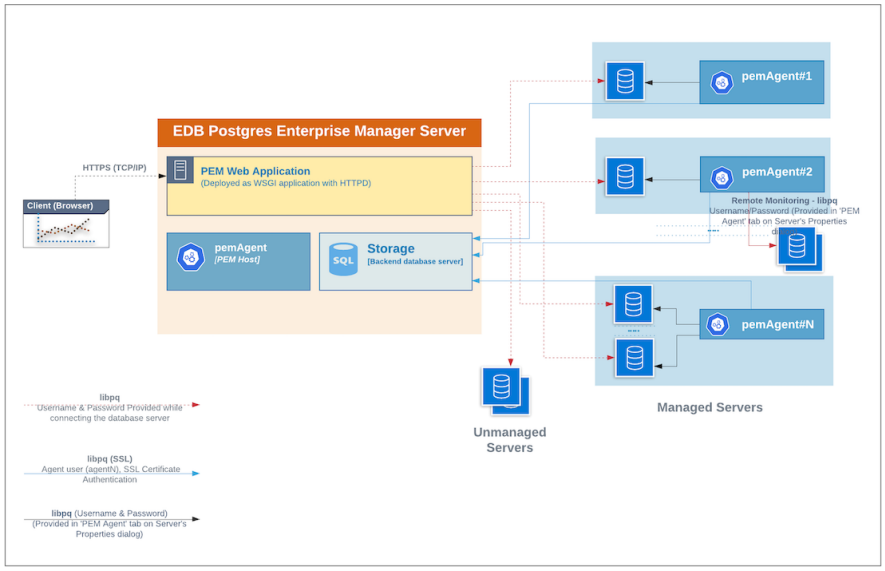


Fig. 1: PEM Architecture

PEM server

The PEM server consists of an instance of Postgres, an instance of the Apache web server providing web services to the client, and a PEM Agent. PEM uses a server-side cryptographic plugin to generate authentication certificates.

The instance of Postgres (a database server) and an instance of the Apache web-server HTTPD) can be on the same host or on separate hosts.

Note: All the PEM features are available on either backend database server you select: PostgreSQL or EDB Postgres Advanced Server.

- **Postgres instance (database server)** — This is the backend database server. It hosts a database named `pem` , which acts as the repository for PEM server. The `pem` database contains several schemas that store metric data collected from each monitored host, server, and database.
- `pem` — This schema is the core of the PEM application. It contains the definitions of configuration functions, tables, or views required by the application.
- `pemdata` — This schema stores the current snapshot of the monitored data.
- `pemhistory` — This schema stores the historical monitored data.
- **Apache web server (HTTPD)** — The PEM web application is deployed as a WSGI application with HTTPD to provide web services to the client. It is made up of the following:

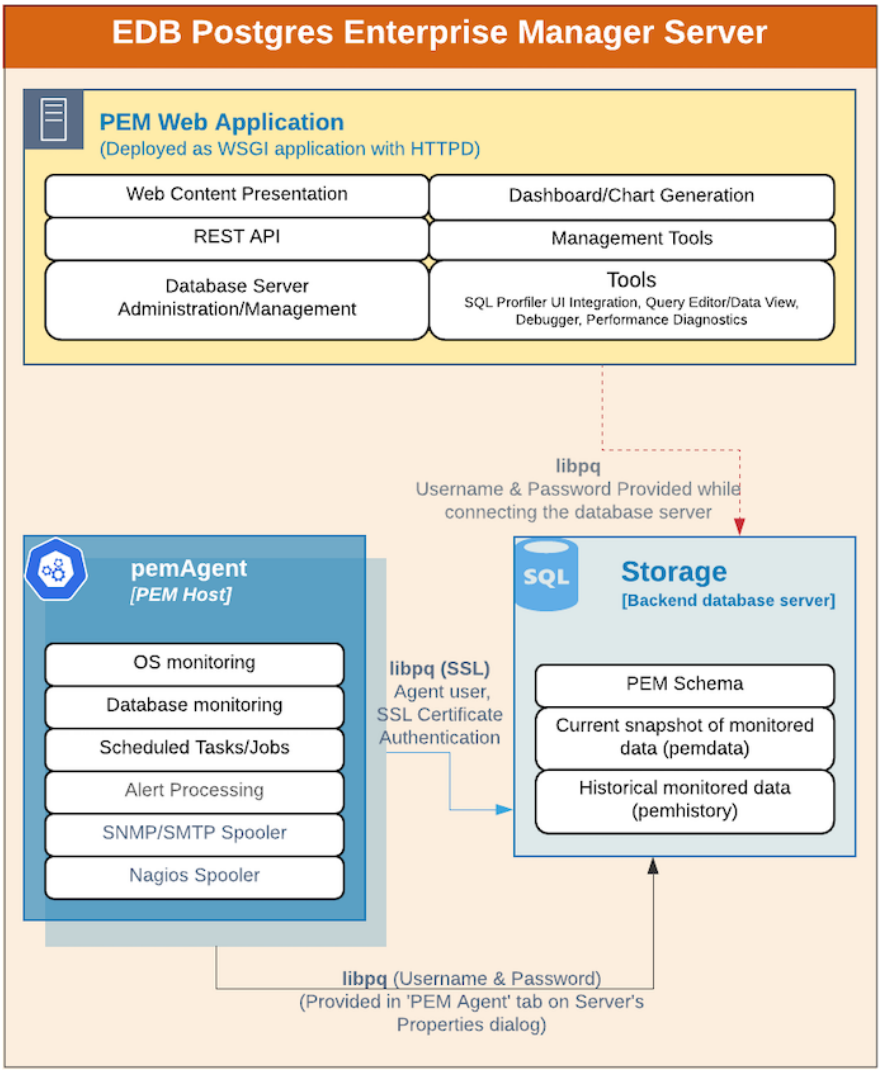


Fig. 2: PEM Server

- **Web content presentation** — The presentation layer is created by the web application (such as browser and login page).
- **Rest API** — The REST API allows integration with other apps and services.
- **Database server administration/management** — You can perform database server administration and management activities like CREATE, ALTER, and DROP for managed and unmanaged servers.
- **Dashboard/chart generation** — Internally, the web application includes functionality that generates dashboards and charts.
- **Management tools** — The Audit Manager, Capacity Manager, Log Manager, Postgres Expert, Postgres Log Analysis Expert, and the Tuning Wizard are available in the web ppplication.
- Other tools provide functionality on managed or unmanaged servers: - **SQL Profiler UI integration** — SQL Profiler generates easily analyzed traces of session content. - **Query editor/data view** —The Query editor allows you to query, edit, and view data. - **Debugger** — The debugger helps you debug queries. - **Performance diagnostics** — Performance diagnostics help you analyze the performance of Postgres instances.

We recommend that you use a dedicated machine to host production instances of the PEM backend database. The host might be subject to high levels of data throughput, depending on the number of database servers that are being monitored and the workloads the servers are processing.

PEM agent

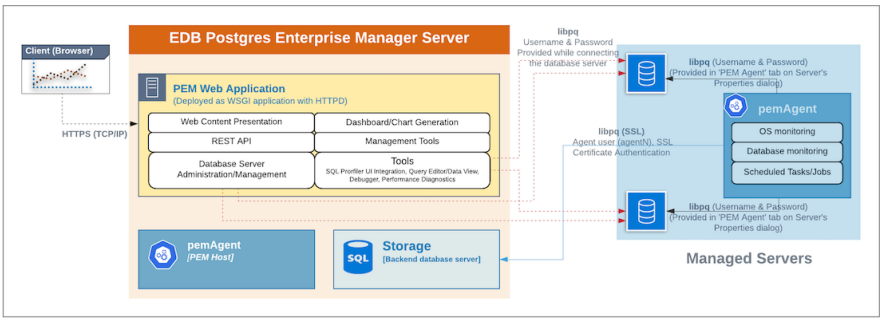


Fig. 3: PEM Agent

The PEM agent is responsible for collecting monitoring data from the machine and operating system and from each of the Postgres instances to which they are bound. Each PEM agent can monitor one physical or virtual machine and is capable of monitoring multiple database servers locally that are installed on other systems. These servers can be installed on the same system or remotely. It' s also responsible for executing other tasks that the user might schedule such as server shutdowns, SQL Profiler traces, and custom jobs.

A PEM agent is installed by default on the PEM server when you install the PEM server. It is generally referred to as a PEM agent on the PEM host. Separately, you can also install the PEM agent on the other servers hosting the Postgres instances you want to monitor using PEM.

Whether monitoring locally or remotely, the PEM agent connects to the PEM server using PostgreSQL' s libpq, using SSL certificate-based authentication. The PEM agent installer in Windows and pemworker CLI in Linux is responsible for registering each agent with the PEM server and generating and installing the required certificates.

There is only one-way traffic between the PEM agent and PEM server. The PEM agent always connects to the PEM server.

The PEM agent must be able to connect to each database server that it monitors. This connection is made over a TCP/IP connection (or, optionally, a Unix Domain Socket on Unix hosts), and can optionally use SSL. You must configure the connection and authentication to the monitored server.

Once configured, each agent collects statistics and other information on the host and each database server and database that it monitors. Each piece of information is known as a *metric* and is collected by a *probe*. Most probes collect multiple metrics at once for efficiency. Examples of the metrics collected include:

- Disk I/O statistics
- Network statistics
- Database server version string
- Database server configuration option (GUC) values
- Table access statistics
- Table and index sizes

For a list of PEM probes, see *Probes* .

By default, the PEM agent bound to the database server collects the OS/database monitoring statistics and also runs any scheduled tasks/jobs for that particular database server, storing data in the `pem` database on the PEM server.

The alert processing, SNMP/SMTP spoolers, and Nagios spooler data is stored in the `pem` database on the PEM server and is then processed by the PEM agent on the PEM host by default. However, you can enable processing by other PEM Agents by adjusting the SNMP/SMTP and Nagios parameters of the PEM agents.

For more information about these parameters, see *Server configuration parameters - reference* .

PEM web client

The PEM client is a web-based application that runs in supported browsers. The client's web interface connects to the PEM server and allows direct management of managed or unmanaged servers and the databases and schemas that reside on them.

The client allows you to use PEM functionality that makes use of the data logged on the server through features such as dashboards, the Postgres Log Analysis Expert, and Capacity Manager.

SQL Profiler extension

You don't have to install the SQL Profiler extension on every server, but you must install and enable the extension on each server on which you want to use SQL Profiler. You might also want to install and enable the SQL Profiler extension on unmonitored development servers. You can also temporarily install the SQL Profiler extension for ad hoc use.

You can use SQL Profiler on servers that aren't managed through PEM. However, to perform scheduled traces, a server must have the extension installed and enabled and must be managed by an installed and configured PEM agent.

For more information, see *Installing the SQL Profiler extension* and *Using SQL Profiler* .

2.2 Deployment considerations

There are a number of things to consider before deploying Postgres Enterprise Manager.

Considerations	Implementation instructions
Does your EDB subscription cover your planned use of PEM?	*Query information*
Is a standalone server sufficient or do you need a high availability architecture?	Installing the server or *Query information*
Do you need to implement connection pooling?	Deploying connection pooling
What type of authentication to use?	*Authentication options*
What actions should you take to avoid security vulnerabilities?	*Securing your deployment*
Where to host the Apache web server?	*Query information*

2.2.1 Using Failover Manager for high availability

Note: This procedure is for setting up Failover Manager for a PEM server with a new installation, not with an existing one. The provided commands apply to the configuration on RHEL-based systems where HTTPD is used for the web server services.

Postgres Enterprise Manager (PEM) helps database administrators, system architects, and performance analysts to administer, monitor, and tune Postgres database servers.

Failover Manager is a high-availability tool from EDB that enables a Postgres primary node to failover to a standby node during a software or hardware failure on the primary. The examples that follow use these IP addresses:

- 172.16.161.200 - PEM Primary
- 172.16.161.201 - PEM Standby 1
- 172.16.161.202 - PEM Standby 2
- 172.16.161.203 - EFM Witness Node
- 172.16.161.245 - PEM VIP (used by agents and users to connect)

The following must use the VIP address:

- The PEM agent binding of the monitored database servers
- Accessing the PEM web client
- Accessing the webserver services

Initial product installation and configuration

1. Install the following on the primary and one or more standbys:

- `EDB Postgres Advanced Server <https://www.enterprisedb.com/docs/epas/latest/installing/>`_ (backend database for PEM Server)
- `PEM server <https://www.enterprisedb.com/docs/pem/latest/installing/>`_

- `EDB Failover Manager 4.1 <https://www.enterprisedb.com/docs/efm/latest/installing/>`_

Refer to the installation instructions in the product documentation using these links or see the instructions on the EDB repos website . Replace `USERNAME:PASSWORD` with your username and password in the instructions to access the EDB repositories.

Make sure that the database server is configured to use the scram-sha-256 authentication method, as the PEM server configuration script doesn' t work with trust authentication.

You must install the `java-1.8.0-openjdk` package to install EFM.

2. Configure the PEM server on the primary server as well as on all the standby servers with an initial configuration of type 1 (web services and database):

```
/usr/edb/pem/bin/configure-pem-server.sh -t 1
```

For more detail on configuration types see, Configuring the PEM server on Linux .

3. Add the following ports in the firewall on the primary and all the standby servers to allow the access:

- `8443` for PEM Server (https)
- `5444` for EPAS 13
- `7800` for EFM
- `7908` for EFM Admin

For example:

```
$ sudo firewall-cmd --zone=public --add-port=5444/tcp --permanent
success
$ sudo firewall-cmd --zone=public --add-port=8443/tcp --permanent
success
$ sudo firewall-cmd --zone=public --add-port=7800/tcp --permanent
success
$ sudo firewall-cmd --zone=public --add-port=7809/tcp --permanent
success
$ sudo firewall-cmd --reload
success
```

Set up the primary node for streaming replication

1. Create the replication role:

```
$ /usr/edb/as13/bin/psql -h 172.16.161.200 -p 5444 -U enterprisedb edb -c "CREATE ROLE␣
↪repl REPLICATION LOGIN PASSWORD password";
CREATE ROLE
```

Give the password of your choice.

2. Configure the following in the `postgresql.conf` file:

```
wal_level = replica
max_wal_senders = 10
```

(continues on next page)

(continued from previous page)

```
wal_keep_size = 500
max_replication_slots = 10
```

For more information on configuring parameters for streaming replication, see the PostgreSQL documentation .

Note: The configuration parameters might differ for different versions of the database server. You can email EDB Support at techsupport@enterprisedb.com for help with setting up these parameters.

3. Add the following entry in the host-based authentication (`/var/lib/edb/as13/data/pg_hba.conf`) file to allow the replication user to connect from all the standbys:

```
hostssl  replication  repl  172.16.161.201/24  scram-sha-256
```

Note: You can change the cidr range of the IP address, if needed.

4. Modify the host-based authentication (`/var/lib/edb/as13/data/pg_hba.conf`) file for the pem_user role to connect to all databases using the scram-sha-256 authentication method:

```
# Allow local PEM agents and admins to connect to PEM server
hostssl all    +pem_user   172.16.161.201/24  scram-sha-256
hostssl pem    +pem_user   127.0.0.1/32       scram-sha-256
hostssl pem    +pem_agent  127.0.0.1/32         cert
# Allow remote PEM agents and users to connect to PEM server
hostssl pem   +pem_user    0.0.0.0/0          scram-sha-256
hostssl pem   +pem_agent   0.0.0.0/0          cert
```

5. Restart the EPAS 13 server.

```
systemctl restart edb-as-13.service
```

Set up the standby nodes for streaming replication

1. Stop the service for EPAS 13 on all the standby nodes:

```
$ systemctl stop edb-as-13.service
```

Note: This example uses the pg_basebackup utility to create the replicas of the PEM backend database server on the standby servers. When using pg_basebackup, you need to stop the existing database server and remove the existing data directories.

2. Remove the data directory of the database server on all the standby nodes:

```
$ sudo su - enterprisedb

$ rm -rf /var/lib/edb/as13/data/*
```

3. Create the `.pgpass` file in the home directory of the enterprisedb user on all the standby nodes:

```
$ sudo su - enterprisedb

$ cat > ~/.pgpass << _EOF_
172.16.161.200:5444:replication:repl:CHANGE_ME
172.16.161.201:5444:replication:repl:CHANGE_ME
172.16.161.202:5444:replication:repl:CHANGE_ME
_EOF_

$ chmod 600 ~/.pgpass
```

4. Take the backup of the primary node on each of the standby nodes using pg_basebackup:

```
$ sudo su - enterprisedb /usr/edb/as13/bin/pg_basebackup -h 172.16.161.200 \
-D /var/lib/edb/as13/data -U repl -v -P -Fp -R -p 5444
```

The `backup` command creates the `postgresql.auto.conf` and `standby.signal` files on the standby nodes. The `postgresql.auto.conf` file has the following content:

```
sudo su - enterprisedb cat /var/lib/edb/as13/data/postgresql.auto.conf
# Do not edit this file manually
# It will be overwritten by the ALTER SYSTEM command.
primary_conninfo = 'user=repl passfile='' /var/lib/edb/.pgpass'' channel_binding=prefer␣
↪host=172.16.161.200 port=5444 sslmode=prefer sslcompression=0 ssl_min_protocol_
↪version=TLSv1.2 gssencmode=prefer krbsvrname=postgres target_session_attrs=any'
```

5. In the `postgresql.conf` file on each of the standby nodes, edit the following parameter:

```
hot_standby = on
```

6. Start the EPAS 13 database server on each of the standby nodes:

```
$ systemctl enable edb-as-13

$ systemctl start edb-as-13
```

7. Copy the following files from the primary node to the standby nodes at the same location, overwriting any existing files. Set the permissions on the files:

- `/etc/httpd/conf.d/edb-pem.conf`
- `/etc/httpd/conf.d/edb-ssl-pem.conf`
- `/root/.pem/agent1.crt`
- `/root/.pem/agent1.key`
- `/usr/edb/pem/agent/etc/agent.cfg`
- `/usr/edb/pem/share/.install-config`
- `/usr/edb/pem/web/pem.wsgi`
- `/usr/edb/pem/web/config_setup.py`

For example:

```
$ mkdir -p /root/.pem
$ chown root:root /root/.pem
$ chmod 0755      /root/.pem
$ mkdir -p        /var/lib/pemhome/.pem
$ chown pem:pem   /var/lib/pemhome/.pem
$ chmod 0700      /var/lib/pemhome/.pem
$ mkdir -p        /usr/edb/pem/logs
$ chown root:root /usr/edb/pem/logs
$ chmod 0755      /usr/edb/pem/logs
$ for file in /etc/httpd/conf.d/edb-pem.conf     \
              /etc/httpd/conf.d/edb-ssl-pem.conf \
              /root/.pem/agent1.crt \
              /usr/edb/pem/agent/etc/agent.cfg \
              /usr/edb/pem/share/.install-config \
              /usr/edb/pem/web/pem.wsgi \
              /usr/edb/pem/web/config_setup.py; do \
       chown root:root ${file}; \
       chmod 0644      ${file}; \
    done;
$ chmod 0600 /root/.pem/agent1.key
$ chown root:root /root/.pem/agent1.key
```

This code ensures that the webserver is configured on the standby and is disabled by default. Switchover by EFM enables the webserver.

Note: Manually keep the certificates in sync on master and standbys whenever the certificates are updated.

8. Run the `configure-selinux.sh` script to configure the SELinux policy for PEM:

```
$ /usr/edb/pem/bin/configure-selinux.sh
getenforce found, now executing getenforce command
Configure the httpd to work with the SELinux
Allow the httpd to connect the database (httpd_can_network_connect_db = on)
Allow the httpd to connect the network (httpd_can_network_connect = on)
Allow the httpd to work with cgi (httpd_enable_cgi = on)
Allow to read & write permission on the pem user home directory
SELinux policy is configured for PEM

$ sudo chmod 640 /root/.pem/agent1.crt
```

9. Disable and stop HTTPD and PEM agent services if they' re running on all replica nodes:

```
systemctl stop pemagent
systemctl stop httpd
systemctl disable pemagent
systemctl disable httpd
```

Note: At this point, a PEM primary server and two standbys are ready to take over from the primary whenever needed.

Set up EFM to manage failover on all hosts

1. Prepare the primary node to support EFM:

- Create a database user efm to connect to the database servers.
- Grant the execute privileges on the functions related to WAL logs and the monitoring privileges to the user.
- Add entries in `pg_hba.conf` to allow the efm database user to connect to the database server from all nodes on all the hosts.
- Reload the configurations on all the database servers. For example:

```
$ cat > /tmp/efm-role.sql << _EOF_
-- Create a role for EFM
CREATE ROLE efm LOGIN PASSWORD password;

-- Give privilege to efm user to connect to a database
GRANT CONNECT ON DATABASE edb TO efm;

-- Give privilege to efm user to do backup operations
GRANT EXECUTE ON FUNCTION pg_current_wal_lsn() TO efm;
GRANT EXECUTE ON FUNCTION pg_last_wal_replay_lsn() TO efm;
GRANT EXECUTE ON FUNCTION pg_wal_replay_resume() TO efm;
GRANT EXECUTE ON FUNCTION pg_wal_replay_pause() TO efm;
GRANT EXECUTE ON FUNCTION pg_reload_conf() TO efm;

-- Grant monitoring privilege to the efm user
GRANT pg_monitor TO efm;
_EOF_

$ /usr/edb/as13/bin/psql -h 172.16.161.200 -p 5444 -U enterprisedb edb -f /tmp/efm-role.
↪sql
CREATE ROLE
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT
GRANT ROLE

$ rm -f /tmp/efm-role.sql

$ cat > /var/lib/edb/as13/data/pg_hba.conf <<< _EOF_
hostssl      edb     efm     172.16.161.200/32     scram-sha-256
hostssl      edb     efm     172.16.161.201/32     scram-sha-256
hostssl      edb     efm     172.16.161.202/32     scram-sha-256
hostssl      edb     efm     172.16.161.203/32     scram-sha-256
_EOF_

$ /usr/edb/as13/bin/psql -h 172.16.161.200 -p 5444 -U enterprisedb edb -c "SELECT pg_
↪reload_conf();"
```

2. Create the scripts on each node to start/stop the PEM agent:

```
$ sudo cat > /usr/local/bin/start-httpd-pemagent.sh << _EOF_
#!/bin/sh
/bin/sudo /bin/systemctl enable httpd
/bin/sudo /bin/systemctl start httpd
/bin/sudo /bin/systemctl enable pemagent
/bin/sudo /bin/systemctl start pemagent
_EOF_
$ sudo cat > /usr/local/bin/stop-httpd-pemagent.sh << _EOF_
#!/bin/sh

/bin/sudo /bin/systemctl stop pemagent
/bin/sudo /bin/systemctl disable pemagent
/bin/sudo /bin/systemctl stop httpd
/bin/sudo /bin/systemctl disable httpd
_EOF_
$ sudo chmod 770 /usr/local/bin/start-pemagent.sh
$ sudo chmod 770 /usr/local/bin/stop-pemagent.sh
```

3. Create a `sudoers` file (`/etc/sudoers.d/efm-pem`) on each node to allow the efm user to start/stop the pemagent:

```
$ sudo cat > /etc/sudoers.d/efm-pem << _EOF_
efm    ALL=(ALL)           NOPASSWD:   /bin/systemctl enable pemagent
efm    ALL=(ALL)           NOPASSWD:   /bin/systemctl disable pemagent
efm    ALL=(ALL)           NOPASSWD:   /bin/systemctl stop pemagent
efm    ALL=(ALL)           NOPASSWD:   /bin/systemctl start pemagent
efm    ALL=(ALL)           NOPASSWD:   /bin/systemctl status pemagent
_EOF_
```

4. Create an `efm.nodes` file on all nodes using the sample file (`/etc/edb/efm-4.1/efm.nodes.in`), and give read-write access to the efm OS user:

```
$ sudo cp /etc/edb/efm-4.1/efm.nodes.in /etc/edb/efm-4.1/efm.nodes
$ sudo chown efm:efm /etc/edb/efm-4.1/efm.nodes
$ sudo chmod 600 /etc/edb/efm-4.1/efm.nodes
```

5. Add the IP address and efm port of the primary node in the `/etc/edb/efm-4.1/efm.nodes` file on the standby nodes:

```
$ sudo cat > /etc/edb/efm-4.1/efm.nodes <<< _EOF_
172.16.161.200:7800
_EOF_
```

6. Create the `efm.properties` file on all the nodes using the sample file (`/etc/edb/efm-4.1/efm.properties.in`). Grant read access to all the users:

```
$ sudo cp /etc/edb/efm-4.1/efm.properties.in /etc/edb/efm-4.1/efm.properties
$ sudo chown efm:efm /etc/edb/efm-4.1/efm.properties
$ sudo chmod a+r /etc/edb/efm-4.1/efm.properties
```

7. Encrypt the efm user' s password using the efm utility:

```
$ export EFMPASS=password
$ /usr/edb/efm-4.1/bin/efm encrypt efm --from-env
096666746b05b081d1a98e43d94c9dad
```

8. Edit the following parameters in the properties file:

```
db.user=efm
db.password.encrypted=096666746b05b081d1a98e43d94c9dad
db.port=5444
db.database=edb
db.service.owner=enterprisedb
db.service.name=edb-as-13
db.bin=/usr/edb/as13/bin
db.data.dir=/var/lib/edb/as13/data
jdbc.sslmode=require
user.email=username@example.com
from.email=node1@efm-pem
notification.level=INFO
notification.text.prefix=[PEM/EFM]
bind.address=172.16.161.200:7800
admin.port=7809
is.witness=false
local.period=10
local.timeout=60
local.timeout.final=10
remote.timeout=10
node.timeout=50
encrypt.agent.messages=true
stop.isolated.primary=true
stop.failed.primary=true
primary.shutdown.as.failure=false
update.physical.slots.period=0
ping.server.ip=8.8.8.8
ping.server.command=/bin/ping -q -c3 -w5
auto.allow.hosts=false
stable.nodes.file=false
db.reuse.connection.count=0
auto.failover=true
auto.reconfigure=true
promotable=true
use.replay.tiebreaker=true
standby.restart.delay=0
reconfigure.num.sync=false
reconfigure.sync.primary=false
minimum.standbys=0
recovery.check.period=1
restart.connection.timeout=60
auto.resume.period=0
virtual.ip=172.16.161.245
virtual.ip.interface=ens33
virtual.ip.prefix=24
virtual.ip.single=true
```

(continues on next page)

(continued from previous page)

```
check.vip.before.promotion=true
pgpool.enable=false
sudo.command=sudo
sudo.user.command=sudo -u %u
syslog.host=localhost
syslog.port=514
syslog.protocol=UDP
syslog.facility=LOCAL1
file.log.enabled=true
syslog.enabled=false
jgroups.loglevel=INFO
efm.loglevel=INFO
jvm.options=-Xmx128m
script.remote.post.promotion=/usr/local/bin/stop-pemagent.sh
script.post.promotion=/usr/local/bin/start-pemagent.sh
```

9. Set the value of the `is.witness` configuration parameter on the witness node to `true` :

```
is.witness=true
```

10. Enable and start the EFM service on the primary node:

```
$ systemctl enable edb-efm-4.1
$ systemctl start edb-efm-4.1
```

11. Allow the standbys to join the cluster started on the primary node:

```
/usr/edb/efm-4.1/bin/efm allow-node  efm  172.16.161.201
/usr/edb/efm-4.1/bin/efm allow-node  efm  172.16.161.202
/usr/edb/efm-4.1/bin/efm allow-node  efm  172.16.161.203
```

12. Enable and start the EFM service on the standby nodes and the EFM witness node:

```
$ systemctl enable edb-efm-4.1
$ systemctl start edb-efm-4.1
```

13. Check the EFM cluster status from any node:

```
$ sudo /usr/edb/efm-4.1/bin/efm cluster-status efm
Cluster Status: efm
    Agent Type  Address              DB       VIP
    --------------------------------------------------------------
    Primary     172.16.161.200      UP       172.16.161.245*
    Standby     172.16.161.201      UP       172.16.161.245
    Standby     172.16.161.202      UP       172.16.161.245
    Witness     172.16.161.203      N/A      172.16.161.245

Allowed node host list:
    172.16.161.200 172.16.161.201 172.16.161.202 172.16.161.203

Membership coordinator: 172.16.161.200

Standby priority host list:
```

(continues on next page)

(continued from previous page)

```
    172.16.161.201 172.16.161.202

Promote Status:

    DB Type     Address              WAL Received LSN   WAL Replayed LSN   Info
    ---------------------------------------------------------------------------
    Primary     172.16.161.200                          0/F7A3808
    Standby     172.16.161.201      0/F7A3808           0/F7A3808
    Standby     172.16.161.202      0/F7A3808           0/F7A3808

    Standby database(s) in sync with primary. It is safe to promote.
```

This status confirms that EFM is set up successfully and managing the failover for the PEM server.

In case of failover, any of the standbys are promoted as the primary node, and PEM agents connect to the new primary node. You can replace the failed primary node with a new standby using this procedure.

Current limitations

The current limitations include:

- Web console sessions for the users are lost during the switchover.
- Per-user settings set from the Preferences dialog box are lost, as they’re stored in local configuration files on the file system.
- Background processes, started by the Backup, Restore, and Maintenance dialogs boxes, and their logs aren’t shared between the systems. They are lost during switchover.

2.2.2 Connection pooling using PgBouncer

You can use PgBouncer as a connection pooler for limiting the number of connections from the PEM agent to the Postgres Enterprise Manager (PEM) server on non-Windows machines:

- ```
  :ref:`PEM server and agent connection management mechanism <PEM server and agent␣
  ↪connection management mechanism>`  provides an introduction of the PgBouncer-PEM␣
  ↪infrastructure.
  ```
- ```
  :ref:`Preparing the PEM database server <Preparing the PEM database server>` ␣
  ↪provides information about preparing the PEM database server to be used with␣
  ↪PgBouncer.
  ```
- ```
  :ref:`Configuring PgBouncer <Configuring PgBouncer>`  provides detailed information␣
  ↪about configuring PgBouncer to allow it to work with the PEM database server.
  ```
- ```
  :ref:`Configuring the PEM agent <Configuring the PEM agent>`  provides detailed␣
  ↪information about configuring a PEM agent to connect to PgBouncer.
  ```

For detailed information about using the PEM web interface, see the *Accessing the web interface* .

PEM server and agent connection management mechanism

Without PgBouncer

In the default configuration, each PEM agent connects to the PEM database server directly using SSL for encryption. Each PEM agent uses its own dedicated user for the connection.

For example, a user agent with ID 1 connects to the PEM database server as agent1.

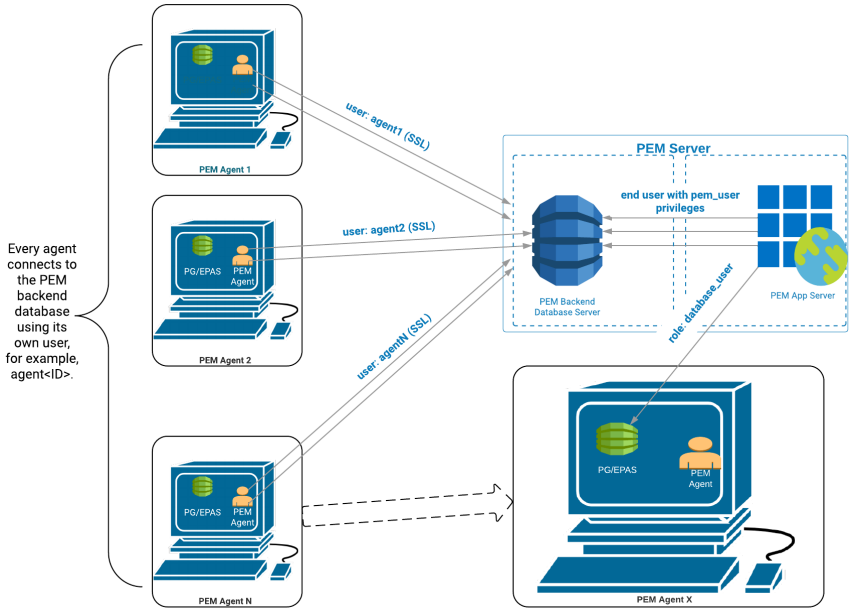


Fig. 4: Connecting to the PEM database without PgBouncer

With PgBouncer

In a PgBouncer-enabled environment, PEM agents can't connect to the PEM database server directly. PEM agents must use a proxy user that you configure specifically for the connection to PgBouncer. In the example, the proxy user handling all PEM agent connections is called `pem_agent_user1` .

Once the PEM agents connect to PgBouncer using SSL, PgBouncer is responsible for managing connection requests to the PEM database server. PgBouncer uses the rules you configured for connection pooling to manage the incoming connection requests, for example, by respecting the established maximum number of active connections.

!!!note Use PgBouncer version 1.9.0 or later as the connection pooler. Versions 1.9.0 or later support cert authentication.

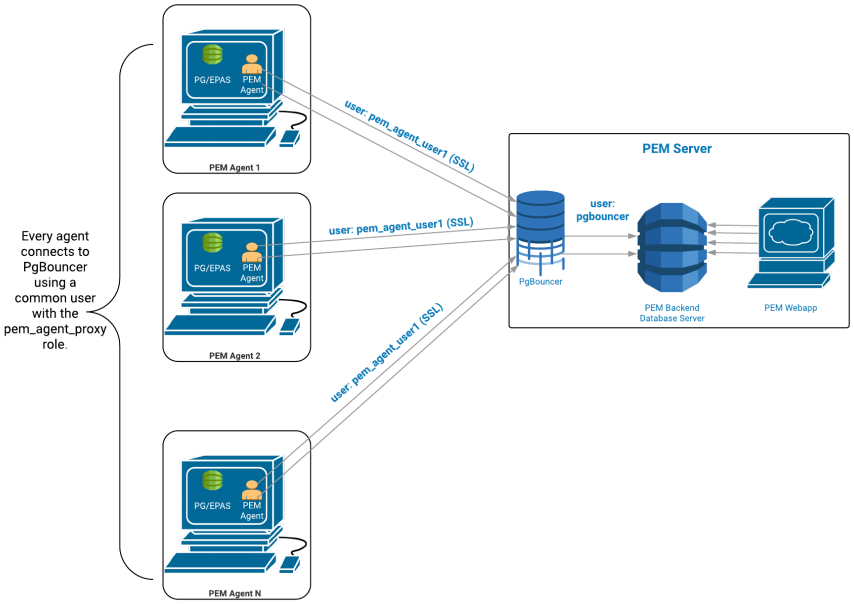


Fig. 5: Connecting to pgBouncer.

Preparing the PEM database server

To enable connection pooling for PEM with PgBouncer, you must configure dedicated users and create an SSL key and certificate on the PEM database server.

This example shows how to prepare the PEM database server with the enterprisedb user on a RHEL-based operating system with EDB Postgres Advanced Server version 16. The location of your data, the configuration and key files, and the user you use to perform the configuration may differ depending on your OS and Postgres distribution.

Prerequisites

- Connect to the `pem` database of the PEM database server.
- Connect as the enterprisedb or postgres user based on your Postgres distribution.

Postgres distribution	User
EDB Postgres Advanced Server	enterprisedb
EDB Postgres Extended Server	postgres
PostgreSQL	postgres

Creating users and roles for PgBouncer-PEM connections

1. Create a dedicated user named pgbouncer with `pem_agent_pool` membership. This user will serve connections from PgBouncer to the PEM database by forwarding all agent database queries.

```
CREATE ROLE pgbouncer PASSWORD ANY_PASSWORD LOGIN;
__OUTPUT__
CREATE ROLE
```

```
GRANT pem_agent_pool TO pgbouncer;
__OUTPUT__
GRANT ROLE
```

1. Create a user named pem_admin1 (not a superuser) with pem_admin and pem_agent_pool role membership. This user registers the agent to the PEM server and manages access to the PEM database.

```
CREATE ROLE pem_admin1 PASSWORD ANY_PASSWORD LOGIN CREATEROLE;
__OUTPUT__
CREATE ROLE
```

```
GRANT pem_agent_pool TO pem_admin1;
__OUTPUT__
GRANT ROLE
```

```
GRANT pem_agent TO pem_admin1 WITH ADMIN OPTION;
__OUTPUT__
GRANT ROLE
```

1. Grant CONNECT privileges to the pgbouncer user:

```
GRANT CONNECT ON DATABASE pem TO pgbouncer;
__OUTPUT__
GRANT
```

1. Grant USAGE privileges to the pgbouncer user for the `pem` schema:

```
GRANT USAGE ON SCHEMA pem TO pgbouncer;
__OUTPUT__
GRANT
```

1. Grant EXECUTE privileges to the pgbouncer user on the `pem.get_agent_pool_auth(text)` function. For example:

```
GRANT EXECUTE ON FUNCTION pem.get_agent_pool_auth(text) TO pgbouncer;
__OUTPUT__
GRANT
```

1. Use the `pem.create_proxy_agent_user(varchar)` function to create a user named pem_agent_user1. This proxy user will serve connections between all Agents and PgBouncer.

```
SELECT pem.create_proxy_agent_user(pem_agent_user1);
__OUTPUT__
create_proxy_agent_user
```

(continues on next page)

(continued from previous page)

```
-------------------------
(1 row)
```

The function creates a user with the same name and a random password and grants pem_agent and pem_agent_pool roles to the user. This approach allows PgBouncer to use a proxy user on behalf of the agent.

Updating the configuration files to allow PgBouncer-PEM connections

1. Allow the pgbouncer user to connect to the `pem` database using the SSL authentication method by adding the `hostssl pem` entry in the `pg_hba.conf` file of the PEM database server.

In the list of rules, ensure you place the `hostssl pem` entry before any other rules assigned to the `+pem_agent` user.

```
# Allow the PEM agent proxy user (used by pgbouncer)
# to connect the to PEM server using SSL

hostssl pem     +pem_agent_pool 127.0.0.1/32  cert map=pem_agent_pool
```

1. Allow the PEM server to map all users involved in PgBouncer-PEM connections by adding these lines to the `$PGDATA/pg_ident.conf` user mapping file:

```
pem_agent_pool  pem_agent_pool  pem_agent_user1
pem_agent_pool  pem_agent_pool  pem_admin1
pem_agent_pool  pem_agent_pool  pgbouncer
```

1. Restart the Postgres service. Replace `<postgres_service>` with the name of the Postgres instance systemd service name:

```
systemctl restart <postgres_service>
```

Creating the SSL key and certificate for PgBouncer-PEM authentication

Create a key and certificate for the pem_agent_pool group role. Then, move the files to the PgBouncer instance to allow authentication between the PEM database server and PgBouncer.

This example runs EDB Postgres Advanced Server on RHEL. When setting your environment variables, choose the correct directories according to your operating system and Postgres distribution.

1. Set the `$DATA_DIR` environment variable to your data directory:

```
export DATA_DIR=/var/lib/edb/as16/data
```

Data directories per OS and Postgres version

Here are some examples of other default data directories per operating system and Postgres version.

Postgres version	RHEL/Rocky Linux/AlmaLinux/SLES	Debian/Ubuntu
EDB Postgres Advanced Server 16	/var/lib/edb/as16/data	/var/lib/edb-as/16/main
EDB Postgres Extended Server 16	/var/lib/edb/edb-pge/16/data	/var/lib/edb-pge/16/main

continues on next page

Table 3 – continued from previous page

Postgres version	RHEL/Rocky Linux/AlmaLinux/SLES	Debian/Ubuntu
PostgreSQL 16	/var/lib/edb/pgsql/16/data	/etc/postgresql/16/main

1. Set the `$USER_HOME` environment variable to the home directory accesible to the user:

```
export USER_HOME=/var/lib/edb
```

User home directories per OS and Postgres version

Here are some examples of other default home directories per operating system and Postgres version.

Postgres version	RHEL/Rocky Linux/AlmaLinux/SLES	Debian/Ubuntu
EDB Postgres Advanced Server 16	/var/lib/edb	/var/lib/edb-as
EDB Postgres Extended Server 16	/var/lib/pgsql	/var/lib/postgresql
PostgreSQL 16	/var/lib/pgsql	/var/lib/postgresql

1. Create the signing key with openssl:

```
openssl genrsa -out pem_agent_pool.key 4096
```

1. Create a certificate-signing request (CSR). Replace the `-subj` attributes in `<...>` as required. Ensure the common name (CN) is set to the pem_agent_pool group role name:

```
openssl req -new -key pem_agent_pool.key -out pem_agent_pool.csr -subj /C=<COUNTRY>/ST=
↪<STATE>/L=<LOCATION>/O=<ORGANISATION>/CN=pem_agent_pool
```

1. Use the PEM CA and key to sign the CSR:

```
openssl x509 -req -days 365 -in pem_agent_pool.csr -CA $DATA_DIR/ca_certificate.crt -
↪CAkey $DATA_DIR/ca_key.key -CAcreateserial -out pem_agent_pool.crt
```

1. Move the created key and certificate to a path the `enterprisedb` user can access.

In this example, create a folder called `~/.postgresql` in the home directory of the `enterprisedb` user and ensure it has permissions:

```
mkdir -p $USER_HOME/.postgresql
mv pem_agent_pool.key pem_agent_pool.crt $USER_HOME/.postgresql
chmod 0600 $USER_HOME/.postgresql/pem_agent_pool.key
chmod 0644 $USER_HOME/.postgresql/pem_agent_pool.crt
chown enterprisedb:enterprisedb $USER_HOME/.postgresql/pem_agent_pool.*
```

Configuring PgBouncer

You must configure PgBouncer to work with the PEM database server.

Prerequisites

- If you're running EDB Postgres Advanced Server, install EDB PgBouncer .
- If you're running EDB Postgres Extended Server or PostgreSQL, install community PgBouncer .

EDB PgBouncer and PgBouncer installation considerations

The name and location of the directories and files in the configuration steps, as well as the user, depend on whether you installed the community version of PgBouncer or EDB PgBouncer. If you installed community PgBouncer (whether you install it from the community repo or the EDB repo), replace the names of the files and directories in the worked example with the values for PgBouncer.

Name	PgBouncer	EDB PgBouncer
PgBouncer directory	*/etc/pgbouncer*	*/etc/edb/pgbouncer<1.x>*
ini file	*pgbouncer.ini*	*edb-pgbouncer.ini*
HBA file	*/etc/pgbouncer/hba_file*	*/etc/edb/pgbouncer<1.x>/hba_file*
Service file	*pgbouncer*	*edb-pgbouncer-<1.x>*
User	*postgres*	*enterprisedb*

Configuring PgBouncer

This example configures EDB PgBouncer with the enterprisedb system user.

If you're running community PgBouncer, replace the names of the directories, files, and user as explained in *EDB PgBouncer and PgBouncer installation considerations* .

1. Open a terminal window and navigate to the PgBouncer directory.
2. Change the owner of the `etc` directory for PgBouncer (where `edb-pgbouncer.ini` resides) to `enterprisedb` , and change the directory permissions to `0700` :

```
chown -R enterprisedb:enterprisedb /etc/edb/pgbouncer<1.x>
chmod 0700 /etc/edb/pgbouncer<1.x>
```

1. Change the contents of the `edb-pgbouncer.ini` file:

```
  [databases]
  ;; Change the pool_size according to maximum connections allowed
  ;; to the PEM database server as required.
  ;; auth_user will be used for authenticate the db user (proxy
  ;; agent user in our case)
  pem = port=5444 host=127.0.0.1 dbname=pem auth_user=pgbouncer pool_size=80 pool_
↪mode=transaction
- = port=5444 host=127.0.0.1 dbname=pem auth_user=pgbouncer pool_size=10
```

(continues on next page)

(continued from previous page)

```
[pgbouncer]
logfile = /var/log/edb/pgbouncer<1.x>/edb-pgbouncer-<1.x>.log
pidfile = /var/run/edb/pgbouncer<1.x>/edb-pgbouncer-<1.x>.pid
listen_addr = *
;; Agent needs to use this port to connect the pem database now
listen_port = 6432
;; Set to require to ensure SSL certificates are used for connections
;; for PEM Agents
client_tls_sslmode = require
;; These are the root.crt, server.key, server.crt files present
;; in the present under the data directory of the PEM database
;; server, used by the PEM Agents for connections.
client_tls_ca_file = /var/lib/edb/as16/data/root.crt
client_tls_key_file = /var/lib/edb/as16/data/server.key
client_tls_cert_file = /var/lib/edb/as16/data/server.crt
;; Allow pgBouncer to use pem_agent_pool certificate
;; and key for connections to the server.
server_tls_key_file = /var/lib/edb/.postgresql/pem_agent_pool.key
server_tls_cert_file = /var/lib/edb/.postgresql/pem_agent_pool.crt
;; Use hba file for client connections
auth_type = hba
;; HBA file
auth_hba_file = /etc/edb/pgbouncer<1.x>/hba_file
;; Use pem.get_agent_pool_auth(TEXT) function to authenticate
;; the db user (used as a proxy agent user).
auth_query = SELECT * FROM pem.get_agent_pool_auth($1)
;; DB User for administration of the pgbouncer
admin_users = pem_admin1
;; auth_dbname and auth_user allow
;; admin console login by admin_users and stats_users
auth_dbname = pem
auth_user = pgbouncer
;; DB User for collecting the statistics of pgbouncer
stats_users = pem_admin1
server_reset_query = DISCARD ALL
;; Change based on the number of agents installed/required
max_client_conn = 500
;; Close server connection if its not been used in this time.
;; Allows to clean unnecessary connections from pool after peak.
server_idle_timeout = 60
```

!!!note For more information on *auth_user* see Authentication settings . .. ::

1. Create an HBA file *(/etc/edb/pgbouncer<1.x>/hba_file)* for PgBouncer that contains the following content:

```
# Use the authentication method scram-sha-256 for local connections
# between the pem database & the pgbouncer (virtual) database.
local pgbouncer all scram-sha-256
# Use the authentication method scram-sha-256 for remote connections
# to pgbouncer (virtual database) using the enterprisedb user.
host pgbouncer,pem pem_admin1 0.0.0.0/0 scram-sha-256
# Use the authentication method cert for TCP/IP connections
```

(continues on next page)

(continued from previous page)

```
# to the pem database using pem_agent_user1
hostssl pem pem_agent_user1 0.0.0.0/0 cert
```

1. Change the owner of the HBA file (`/etc/edb/pgbouncer<1.x>/hba_file`) to `enterprisedb` , and change the directory permissions to `0600` :

```
chown enterprisedb:enterprisedb /etc/edb/pgbouncer<1.x>/hba_file
chmod 0600 /etc/edb/pgbouncer<1.x>/hba_file
```

1. Enable the PgBouncer service, and start the service:

```
systemctl enable edb-pgbouncer-<1.x>
__OUTPUT__
Created symlink from
/etc/systemd/system/multi-user.target.wants/edb-pgbouncer-<1.x>.service
to /usr/lib/systemd/system/edb-pgbouncer-<1.x>.service.
```

```
systemctl start edb-pgbouncer-<1.x>
```

Configuring the PEM agent

Prerequisites

Install the PEM agent .

!!!note Don’ t configure PEM agents with `enable_smtp` , `enable_snmp` , or `enable_webhook` set to `true` in the `agent.cfg` file to connect through PgBouncer. SNMP, SMTP, and Webhook spoolers use the LISTEN/NOTIFY mechanism provided by Postgres to send notifications asynchronously. Since PgBouncer doesn’ t support the LISTEN/NOTIFY mechanism in transaction mode, connecting the agent to PgBouncer can cause notifications to be delayed or not delivered at all. Instead, connect the PEM agent directly to the PEM backend database.

Configuring a new PEM agent

After installing the PEM agent, configure it to work with a particular PEM database server:

```
PGSSLMODE=require PEM_SERVER_PASSWORD=pem_admin1_password \
      /usr/edb/pem/agent/bin/pemworker \
      --register-agent \
      --pem-server 172.16.254.22 \
      --pem-port 6432 \
      --pem-user pem_admin1 \
      --pem-agent-user pem_agent_user1 \
      --display-name *Agent_Name* \
__OUTPUT__
Postgres Enterprise Manager Agent registered successfully!
```

In this command, the `--pem-agent-user` argument instructs the agent to create an SSL certificate and key pair for the pem_agent_user1 database user in the `/root/.pem` directory.

For example:

```
/root/.pem/pem_agent_user1.crt

/root/.pem/pem_agent_user1.key
```

The PEM agent uses the keys to connect to the PEM database server as pem_agent_user1. It also creates an agent configuration file named `/usr/edb/pem/agent/etc/agent.cfg` .

A line mentioning the agent-user to use appears in the `agent.cfg` configuration file. For example:

```
cat /usr/edb/pem/agent/etc/agent.cfg
[PEM/agent]
pem_host=172.16.254.22
pem_port=6432
agent_id=12
agent_user=pem_agent_user1
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
log_level=warning
log_location=/var/log/pem/worker.log
agent_log_location=/var/log/pem/agent.log
long_wait=30
short_wait=10
alert_threads=0
enable_smtp=false
enable_snmp=false
enable_webhook=false
max_webhook_retries=3
allow_server_restart=true
allow_package_management=false
allow_streaming_replication=false
max_connections=0
connect_timeout=-1
connection_lifetime=0
allow_batch_probes=false
heartbeat_connection=false
```

Configuring an existing PEM agent

If you’re using an existing PEM agent, you can copy the SSL certificate and key files to the target machine and reuse the files. You must modify the files, adding a new parameter and replacing some parameters in the existing `agent.cfg` file.

1. Add a line to use agent_user as the agent:

```
agent_user=pem_agent_user1
```

1. Update the port to specify the PgBouncer port:

```
pem_port=6432
```

1. Update the certificate and key path locations:

```
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
```

As an alternative, you can run the agent self-registration script. However, that process creates a new agent id. If you run the agent self-registration script, you must replace the new agent id with the existing id and disable the entry for the new agent id in the `pem.agent` table. For example:

```
pem=# UPDATE pem.agent SET active = false WHERE id = <new_agent_id>;
__OUTPUT__

UPDATE 1
```

Note: Keep a backup of the existing SSL certificate, key file, and agent configuration file.

2.2.3 Authentication options

PEM also supports Kerberos and 2FA authentication. For implementation instructions, see:

On Linux:

- `:ref:`Configuring the PEM server to use Kerberos authentication <Configuring the PEM server to use Kerberos authentication>`
- `:ref:`Configuring the PEM server to use Windows Active Directory domain services for Kerberos authentication (SSPI) <Configuring the PEM server to use Windows Active Directory domain services for Kerberos authentication (SSPI)>`

On Linux and Windows:

- `:ref:`Configuring the PEM server to use two-factor authentication <Configuring the PEM server to use two-factor authentication>`

Configuring the PEM server to use Kerberos authentication

You can configure Kerberos authentication for the PEM server. The Kerberos server works with hostnames and not with IP addresses. To use single sign-on in PEM server using Kerberos authentication, configure the following machines with hostnames using the DNS (realm).

```
-   Kerberos server -   PEM server (PEM web server and PEM backend database server) -  Client machine
```

For example, if the realm on Kerberos server is `edbpem.org` , then you can set the Kerberos server hostname to `Krb5server.edbpem.org` , the PEM server hostname to `pem.edbpem.org` , and the client' s hostname to `pg12.edbpem.org` .The convention is to use the DNS domain name as the name of the realm.

1. Install Kerberos, the PEM server, and the PEM backend database

Install Kerberos on the machine that functions as the authentication server. Install the PEM server on a separate machine. For more information, see *Installing the PEM server and PEM-HTTPD on separate hosts* .

Install the PEM backend database (Postgres/EDB Postgres Advanced Server) on the same machine as the PEM server or on a different one. For more information, see the Installation steps on EDB Docs website .

2. Add principals on Kerberos server

Add the principals for the PEM web application deployed under an Apache web server (HTTPD/Apache2) and the PEM Backend Database Server (PostgreSQL/EDB Postgres Advanced Server).

```
$ sudo kadmin.local -q "addprinc -randkey HTTP/<HOSTNAME_OF_PEM_SERVER>"
$ sudo kadmin.local -q "addprinc -randkey postgres/<HOSTNAME_OF_PEM_SERVER>"
```

`HOSTNAME_OF_PEM_SERVER` must contain the realm of the Kerberos server. For example, you can specify `pemdb.edbpem.org` as the hostname of PEM server, with `edbpem.org` as the realm.

Note: If the PEM web application and the PEM backend database server are on different machines, then hostname is different for each one.

3. Extract key tables from Kerberos server

Extract the key tables from Kerberos for the PEM web application and the PEM backend database server:

```
sudo kadmin.local "ktadd -k <NAME_OF_PEM_WEB_FILE>.keytab HTTP/<HOSTNAME_OF_PEM_SERVER>"
sudo kadmin.local "ktadd -k <NAME_OF_PEM_DB_FILE>.keytab postgres/<HOSTNAME_OF_PEM_
↪SERVER>"
```

Copy the key tables from the Kerberos server to the PEM server:

```
scp <NAME_OF_PEM_WEB_FILE>.keytab <OS_USERNAME_ON_PEM_SERVER>@<HOSTNAME_OF_PEM_SERVER>:/
↪tmp
scp <NAME_OF_PEM_DB_FILE>.keytab <OS_USERNAME_ON_PEM_SERVER>@<HOSTNAME_OF_PEM_SERVER>:/
↪tmp
```

On the PEM server, move the key tables to the required location and change ownership:

```
mv /tmp/<NAME_OF_PEM_WEB_FILE>.keytab <PEM_INSTALLATION_DIRECTORY>/share
chown pem <PEM_INSTALLATION_DIRECTORY>/share/<NAME_OF_PEM_WEB_FILE>.keytab
```

```
mv /tmp/<NAME_OF_PEM_DB_FILE>.keytab <DATA_DIRECTORY_OF_POSTGRES>/
chown enterprisedb <DATA_DIRECTORY_OF_POSTGRES>/<NAME_OF_PEM_DB_FILE>.keytab
```

Where:

- `NAME_OF_PEM_WEB_FILE` is the name specified for the key table for the PEM web application.
- `NAME_OF_PEM_DB_FILE` is the name specified for the key table for the PEM backend database server.
- `OS_USERNAME_ON_PEM_SERVER` is the name of the operating system user on the PEM server.

- DATA_DIRECTORY_OF_POSTGRES is the path of the data directory of the installed Postgres database (PostgreSQL/EDB Postgres Advanced Server).

4. Configure the PEM backend database server

Add the key table location in the postgresql.conf file:

```
krb_server_keyfile=FILE:/<DATA_DIRECTORY_OF_POSTGRES>/<NAME_OF_PEM_DB_FILE>.keytab
```

Where:

- NAME_OF_PEM_DB_FILE is the name specified for the key table for the PEM backend database server.
- DATA_DIRECTORY_OF_POSTGRES is the path of the data directory of the installed Postgres database (PostgreSQL/EDB Postgres Advanced Server).

Edit the krb5.conf file:

```
$ sudo vim /etc/krb5.conf
[libdefaults]
 default_realm = EDBPEM.ORG
Forwardable = True

[domain_realm]
.edbpem.org = EDBPEM.ORG
edbpem.org = EDBPEM.ORG

[realms]
EDBPEM.ORG = {
   kdc = krb5server.edbpem.org
   admin_server = krb5server.edbpem.org
}
```

Restart the database server to reflect the changes:

```
systemctl restart <POSTGRES_SERVICE_NAME>
```

POSTGRES_SERVICE_NAME is the service name of the Postgres (PostgreSQL/EDB Postgres Advanced Server) database, for example, postgresql-13 for PostgreSQL 13 database on a RHEL or Rocky Linux platforms.

5. Obtain and view the initial ticket

The kinit utility obtains and caches Kerberos tickets. You typically use this utility to obtain the ticket-granting ticket, entering a password to decrypt the credential from the key distribution center (KDC). The ticket-granting ticket is then stored in your credential cache.

You can view the details of the ticket using the klist utility.

Note: Install the Kerberos client on the PEM server and the client machine before using kinit and klist.

```
$ kinit <USERNAME@REALM>
$ klist
```

It displays the principal along with the Kerberos ticket.

Note: The *USERNAME@REALM* specified here must be a database user having the pem_admin role and CONNECT privilege on *pem* database.

6. Configure the PEM server

Run the PEM configure script on the PEM server to use Kerberos authentication:

```
$ sudo PEM_APP_HOST=<HOSTNAME_OF_PEM_SERVER> PEM_KRB_KTNAME=<PEM_INSTALLATION_DIRECTORY/
↪share/<NAME_OF_PEM_WEB_FILE>.keytab <PEM_INSTALLATION_DIRECTORY>/bin/configure-pem-
↪server.sh
```

Configure `PEM_DB_HOST` in the `config_setup.py` file. Check that the value of `PEM_AUTH_METHOD` is set to `'kerberos'` .

```
$ sudo vim <PEM_INSTALLATION_DIRECTORY>/share/web/config_setup.py
PEM_DB_HOST=`<HOSTNAME_OF_PEM_SERVER>`
```

Configure the host in the `.install-config` file:

```
$ sudo vim <PEM_INSTALLATION_DIRECTORY>/share/.install-config
HOST=`<HOSTNAME_OF_PEM_SERVER>`
```

If the PEM server uses Kerberos authentication:

- All the monitored servers default to use the same authentication. To override the default, in the `config_local.py` file, add the parameter `ALLOW_DATABASE_CONNECTION_WITHOUT_KERBEROS` and set it to `True` .
- All the authenticated user principals are appended with the realm (USERNAME@REALM) and passed as the database user name by default. To override the default, in the `config_local.py` file, add the parameter `PEM_USER_KRB_INCLUDE_REALM` and set it to `False` .
- Restart the Apache server

```
sudo systemctl restart <SERVICE_NAME>
```

- Edit the entries at the top of `pg_hba.conf` to use the gss authentication method, and reload the database server.

```
host  pem       +pem_user    <ip_of_pem_server>/32   gss
host  postgres  +pem_user    <ip_of_pem_server>/32   gss
```

```
systemctl reload <POSTGRES_SERVICE_NAME>
```

`POSTGRES_SERVICE_NAME` is the service name of the Postgres (PostgreSQL/EDB Postgres Advanced Server) database, for example, `postgresql-13` for PostgreSQL 13 database on a `RHEL` or Rocky Linux platforms.

Note: If you're using PostgreSQL or EDB Postgres Advanced Server 12 or later, then you can specify connection type as *hostgssenc* to allow only gss-encrypted connection.

7. Browser settings

Configure the browser on the client machine to access the PEM web client to use the Spnego/Kerberos.

For Mozilla Firefox:

1. Open the low-level Firefox configuration page by loading the `about:config` page.
2. In the search box, enter `network.negotiate-auth.trusted-uris` .
3. Double-click the `network.negotiate-auth.trusted-uris` preference and enter the hostname or the domain of the web server that' s protected by Kerberos HTTP SPNEGO. Separate multiple domains and hostnames with a comma.
4. In the search box, enter `network.negotiate-auth.delegation-uris` .
5. Double-click the `network.negotiate-auth.delegation-uris` preference and enter the hostname or the domain of the web server that' s protected by Kerberos HTTP SPNEGO. Separate multiple domains and hostnames with a comma.
6. Select **OK**.

For Google Chrome on Linux or MacOS:

- Add the `--auth-server-whitelist` parameter to the `google-chrome` command. For example, to run Chrome from a Linux prompt, run the `google-chrome` command as follows:

```
google-chrome --auth-server-whitelist = "hostname/domain"
```

- After configuring the PEM server, you can access the PEM web interface in your browser. Navigate to:

```
https://<ip_address_of_PEM_server>:8443/pem
```

Note: You might see the following error while connecting to your Postgres cluster:

```
psql -h hostname template1     psql:  GSSAPI continuation error:  Unspecified GSS failure.
Minor code may provide more information     GSSAPI continuation error:  Key version is not
available
```

Add encryption types to the keytab using ktutil or by recreating the Postgres keytab with all crypto systems from AD.

Configuring the PEM server to use Windows Active Directory domain services for Kerberos authentication (SSPI)

The Windows Active Directory domain service works with hostnames and not with IP addresses. To use single sign-on in PEM Server using Active Directory domain services, configure the following machines with hostnames using the DNS:

- Windows server (domain controller)
- PEM server (PEM web server and PEM backend database server)
- Client machine

For example, if the realm on Windows Active Directory is `edbpem.internal` , then you can set the Windows server hostname to `Krb5server.edbpem.internal` , the PEM server hostname to `pem.edbpem.internal` , and the client' s hostname to `pg12.edbpem.internal` .

1. Install Active Directory, the PEM server, and the PEM backend database server

Perform the following installations:

- Install Active Directory on the Windows server (domain controller) that functions as the authentication server. Also, configure the Active Directory domain services to use Kerberos authentication, and then start it.
- Install the PEM server on a separate Linux machine. For more information, see *Installing the PEM server and PEM-HTTPD on separate hosts* .
- Install the PEM backend database (Postgres/EDB Postgres Advanced Server) on the same Linux machine as the PEM server or a different one. For more information, see the installation steps on the EDB Docs website .

2. Create users in Active Directory to map with service principals

Create users in Active Directory of the Windows server to map with the HTTP service principal for the PEM web application.

1. Open **Active Directory Users and Computers > <DOMAIN_NAME> > Users**. Right-click and select **New > User**.

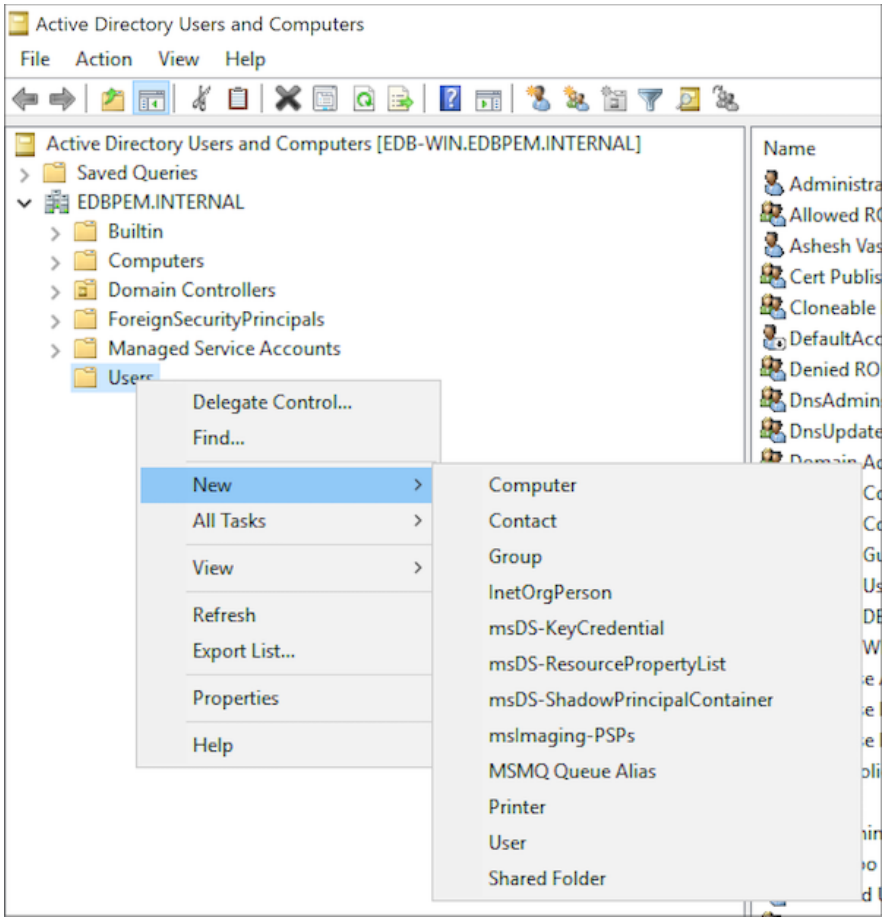


Fig. 6: Active Directory Users and Computers

1. Enter the user details.

2. Give the password and make sure to clear **User must change password at next logon**. Also select **User cannot change password** and **Password never expires**.
3. Review the user details.
4. On the PEMServer Web Properties dialog box, add the users as members of the Domain Users group:

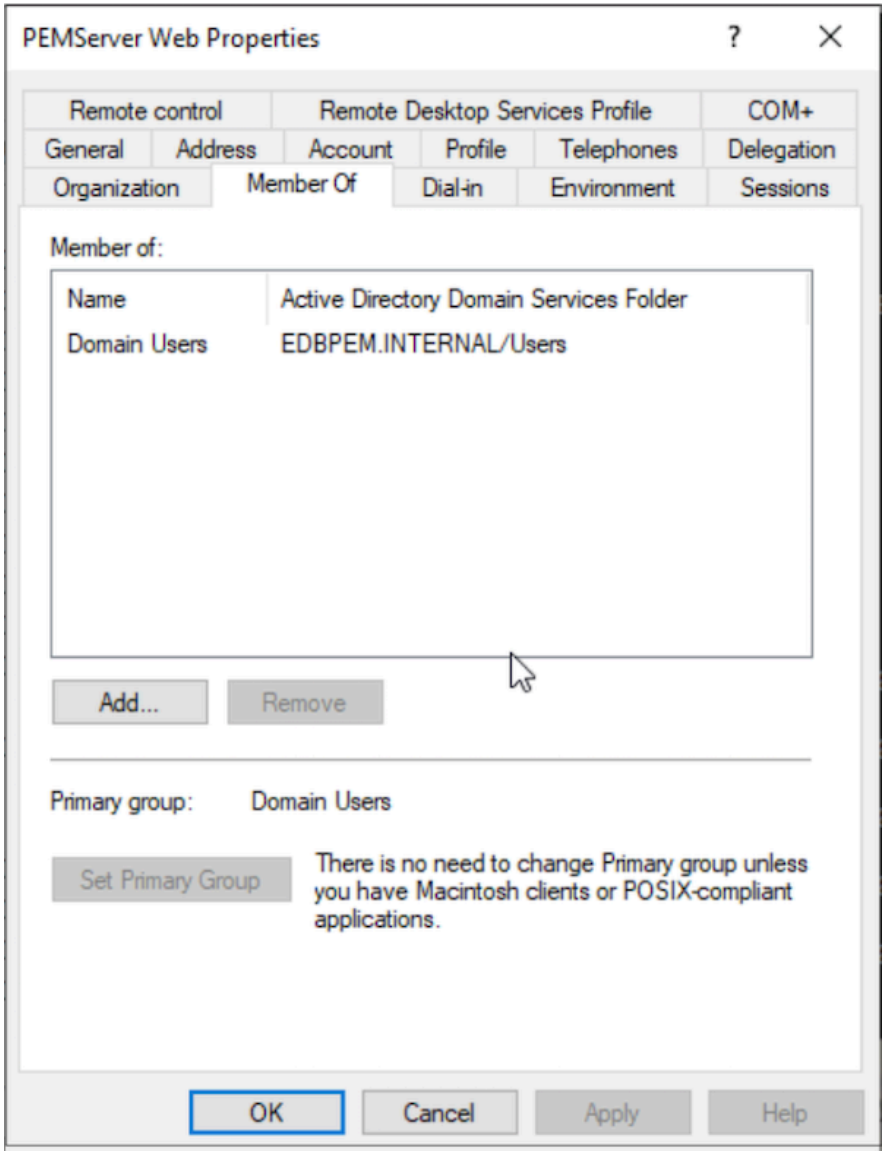


Fig. 7: PEM Server Web Properties

1. Create the user (for example, pemserverdb) in Active Cirectory of the Windows server to map with the Postgres service principal for the PEM backend database.

3. Extract key tables from Active Directory

Extract the key tables for the service principals and map them with the respective domain users you created.

1. Open Windows PowerShell as an administrator. Create a key table for HTTP service principal mapping with the user pemserverweb and a key table for Postgres service principal mapping with the user pemserverdb:

```
ktpass /out pemserver.keytab /princ HTTP/pem.edbpem.internal@EDBPEM.INTERNAL /mapuser␣
↪pemserverweb /crypto AES256-SHA1 +rndpass /target EDBPEM.INTERNAL -ptype KRB5_NT_
↪PRINCIPAL -kvno 0
ktpass /out pemdb.keytab /princ postgres/pem.edbpem.internal@EDBPEM.INTERNAL /mapuser␣
↪pemserverdb /crypto AES256-SHA1 +rndpass /target EDBPEM.INTERNAL -ptype KRB5_NT_
↪PRINCIPAL -kvno 0
```

Where:

```
-   `pemserver.keytab`  is the name of the key table for the PEM web application. -  ␣
↪`pemdb.keytab`  is the name of the key table for the PEM backend database server. -
 `pem.edbpem.internal@EDBPEM.INTERNAL`  is the hostname of the PEM server. Here, @EDBPEM.
↪INTERNAL means @REALM. -   `pemserverweb`  is the user for the PEM web application. -
 `pemserverdb`  is the user for the PEM backend database server. -   `EDBPEM.INTERNAL` ␣
↪is the domain of the target.
```

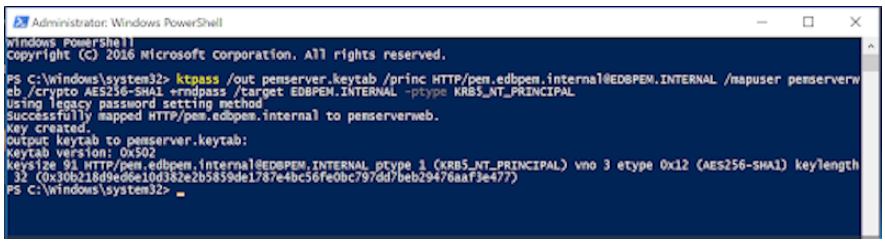


Fig. 8: Windows PowerShell ktpass

Note: The command line argument *+rndpass* resets the password for the domain user pemserverweb to a random password. The */target* option is optional.

1. On the **Accounts** tab, add Kerberos support for the user accounts.

Note: On the **Accounts** tab, the user logon name shows **HTTP/pem.edbpem.internal@EDBPEM.INTERNAL**. The **Delegation** tab is enabled for the pemserverweb user.

1. On the **Delegation** tab, select **Trust this user for delegation to any service (Kerberos only)** for the users you created.
2. Copy both the key tables to the PEM server host or to the PEM web server and PEM backend database server hosts if installed on different hosts.
3. On the PEM server, move the key tables to the required location and change the ownership:

```
mv /tmp/pemserver.keytab <PEM_INSTALLATION_DIRECTORY>/share
chown pem <PEM_INSTALLATION_DIRECTORY>/share/pemserver.keytab
```

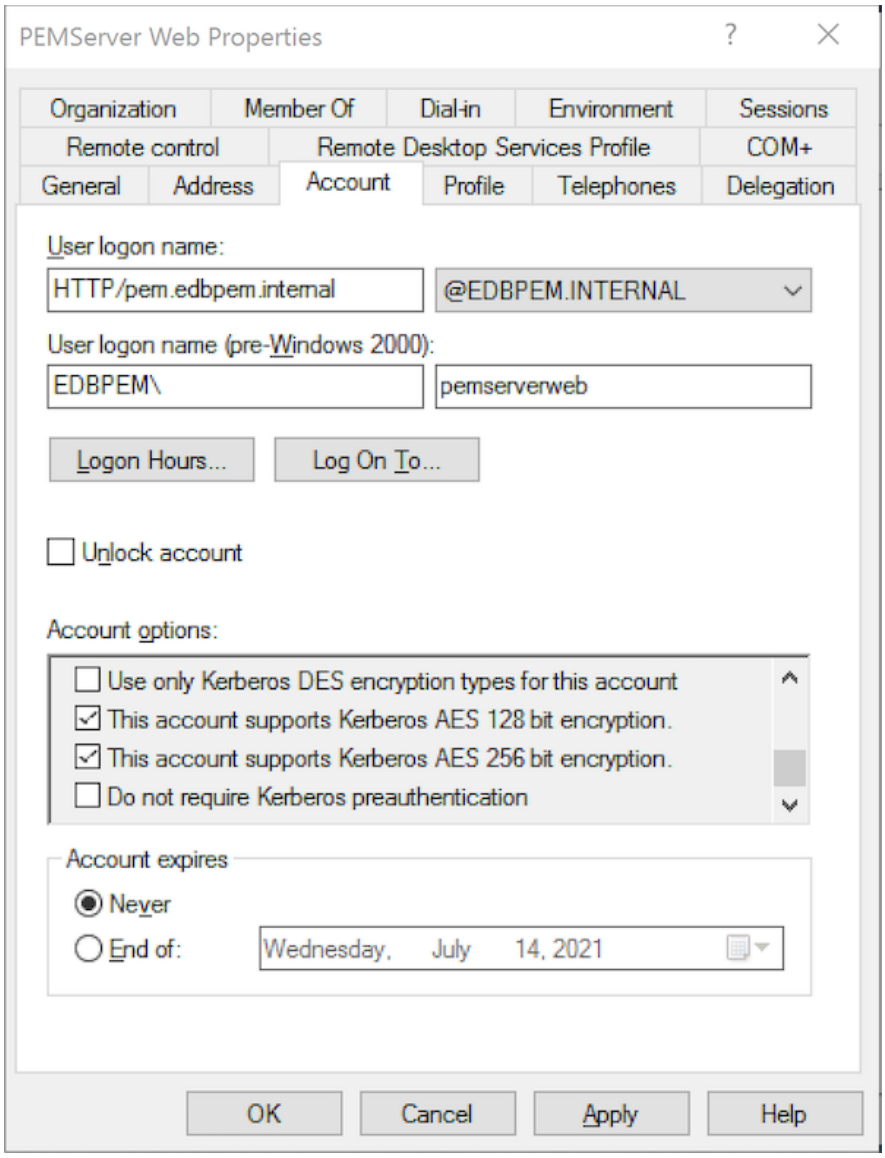


Fig. 9: PEM Server Web Properties

```
mv /tmp/pemdb.keytab <DATA_DIRECTORY_OF_POSTGRES>/
chown enterprisedb <DATA_DIRECTORY_OF_POSTGRES>/pemdb.keytab
```

Where:

```
-   `OS_USERNAME_ON_PEM_SERVER`  is the name of the operating system user on the PEM␣
↪server. -   `DATA_DIRECTORY_OF_POSTGRES`  is the path of the data directory of the␣
↪installed Postgres database (PostgreSQL/EDB Postgres Advanced Server).
```

4. Configure the PEM backend database server

Add the key table location in the `postgresql.conf` file.

```
krb_server_keyfile=FILE:/<DATA_DIRECTORY_OF_POSTGRES>/pemdb.keytab
```

Where `DATA_DIRECTORY_OF_POSTGRES` is the path of the data directory of the installed Postgres database (PostgreSQL/EDB Postgres Advanced Server).

Edit the `krb5.conf` file:

```
$ sudo vim /etc/krb5.conf
[libdefaults]
default_realm = EDBPEM.INTERNAL
Forwardable = True

[domain_realm]
.edbpem.org = EDBPEM.INTERNAL
edbpem.org = EDBPEM.INTERNAL

[realms]
EDBPEM.INTERNAL = {
   kdc = krb5server.edbpem.internal
   admin_server = krb5server.edbpem.internal
}
```

Restart the database server to reflect the changes:

```
systemctl restart <POSTGRES_SERVICE_NAME>
```

`POSTGRES_SERVICE_NAME` is the service name of the Postgres (PostgreSQL/EDB Postgres Advanced Server) database, for example, `postgresql-13` for PostgreSQL 13 database on RHEL or Rocky Linux platforms.

5. Obtain and view the initial ticket

The kinit utility obtains and caches Kerberos tickets. You typically use this utility to obtain the ticket-granting ticket, using a password you entered to decrypt the credential from the key distribution center (KDC). The ticket-granting ticket is then stored in your credential cache.

You can view the details of the ticket using the klist utility.

Note: You must install the Kerberos client on the PEM server and the client machine before using kinit and klist.

```
$ kinit <USERNAME@REALM>
$ klist
```

It displays the principal along with the Kerberos ticket.

Note: The *USERNAME@REALM* specified here must be a database user having the pem_admin role and CONNECT privileges on the *pem* database.

6. Configure the PEM server

Run the PEM configure script on the PEM server to use Kerberos authentication:

```
$ sudo PEM_APP_HOST=pem.edbpem.internal PEM_KRB_KTNAME=<PEM_INSTALLATION_DIRECTORY/share/
↪pemserver.keytab <PEM_INSTALLATION_DIRECTORY>/bin/configure-pem-server.sh
```

In the `config_setup.py` file, configure `PEM_DB_HOST` and check that the value of `PEM_AUTH_METHOD` is set to `'kerberos'` .

```
$ sudo vim <PEM_INSTALLATION_DIRECTORY>/share/web/config_setup.py
PEM_DB_HOST=`pem.edbpem.internal`
```

Configure `HOST` in the `.install-config` file.

```
$ sudo vim <PEM_INSTALLATION_DIRECTORY>/share/.install-config
HOST=`pem.edbpem.internal`
```

If the PEM server uses Kerberos authentication:

- All the monitored servers default to use the same authentication. To override the default, in the `config_local.py` file, add the parameter `ALLOW_DATABASE_CONNECTION_WITHOUT_KERBEROS` and set it to `True` .
- All the authenticated user principals are appended with the realm (USERNAME@REALM) and passed as the database user name by default. To override the default, in the `config_local.py` file, add the parameter `PEM_USER_KRB_INCLUDE_REALM` and set it to `False` .

Restart the Apache server:

```
sudo systemctl restart <SERVICE_NAME>
```

Edit the entries at the top in `pg_hba.conf` to use the gss authentication method. Then reload the database server.

```
host  pem       +pem_user    <ip_of_pem_server>/32   gss
host  postgres  +pem_user    <ip_of_pem_server>/32   gss
```

```
systemctl reload <POSTGRES_SERVICE_NAME>
```

`POSTGRES_SERVICE_NAME` is the service name of the Postgres (PostgreSQL/EDB Postgres Advanced Server) database, for example, `postgresql-13` for PostgreSQL 13 database on RHEL or Rocky Linux platforms.

Note: You can't specify the connection type as *hostgssenc* . Windows doesn't support gss encrypted connection.

7. Browser settings

Configure the browser on the client machine to access the PEM web client to use the Spnego/Kerberos.

For Mozilla Firefox:

- Open the low-level Firefox configuration page by loading the `about:config` page.
- In the search box, enter `network.negotiate-auth.trusted-uris` .
- Double-click the `network.negotiate-auth.trusted-uris` preference and enter the hostname or the domain of the web server that' s protected by Kerberos HTTP SPNEGO. Separate multiple domains and hostnames with commas.
- In the search box, enter `network.negotiate-auth.delegation-uris` .
- Double-click the `network.negotiate-auth.delegation-uris` preference and enter the hostname or the domain of the web server that' s protected by Kerberos HTTP SPNEGO. Separate multiple domains and hostnames with commas.
- Select **OK**.

For Google Chrome on Linux or MacOS:

- Add the `--auth-server-whitelist` parameter to the `google-chrome` command. For example, to run Chrome from a Linux prompt, run the `google-chrome` command as follows:

```
google-chrome --auth-server-whitelist = "hostname/domain"
```

- After configuring the PEM server, you can access the PEM web interface in your browser. Navigate to:

```
https://<ip_address_of_PEM_server>:8443/pem
```

Note: You might see the following error while connecting to your Postgres cluster:

```
psql -h hostname template1     psql:  GSSAPI continuation error:  Unspecified GSS failure.
Minor code may provide more information     GSSAPI continuation error:  Key version is not
available
```

Add encryption types to the keytab using ktutil or by recreating the Postgres keytab with all crypto systems from AD.

Configuring the PEM server to use two-factor authentication

PEM supports two methods for 2FA:

- Email authentication
- Authenticator app (such as Google Authenticator)

To enable 2FA, you can copy these settings from the `config.py` file to the `config_local.py` file and modify the following parameters.

Parameter	Description
MFA_ENABLED	Set to *true* to enable the two-factor authentication. Default value is *false*.
MFA_FORCE_REGISTRA'	Set to *true* to ask the users to register forcefully for the two-factor authentication methods at login. Default value is *false*.

continues on next page

Table 6 – continued from previous page

Parameter	Description
MFA_SUPPORTED_METH	Set to *email* to use the email authentication method (send a one-time code by email) or *authenticator* to use the TOTP-based application authentication method.
MFA_EMAIL_SUBJECT	Set to the subject of the email for email authentication. Default value is *<APP_NAME> - Verification Code*.

Mail server configuration

To use the email authentication method, you need to configure mail server settings.

PEM server can send an email using either the SMTP configurations saved in the PEM configuration or using Flask-Mail.

To send the email verification code using the internal SMTP configuration from the PEM configuration, set the parameter `MAIL_USE_PEM_INTERNAL` to `True` . If set to `False` , the following mail configuration is used to send the code on the user-specified email address:

- MAIL_SERVER = 'localhost'
- MAIL_PORT = 25
- MAIL_USE_TLS = False
- MAIL_USE_SSL = False
- MAIL_USERNAME = None
- MAIL_PASSWORD = None
- MAIL_DEFAULT_SENDER = None

For more details about these configurations, see the Flask-Mail documentation .

Note: PEM SMTP alerts don't use this configuration.

2.2.4 Securing your deployment

When you configure PEM, there are security practices to consider. These security practices are optional. However, EDB recommends these practices to enhance the overall system security.

PEM depends on third-party components from the vendor repository, including the Apache web server, OpenSSL, snmp++, and libcurl. To ensure these components are up to date, update your operating system and regularly apply security updates to avoid any security vulnerability. Without the most recent security patches, your system is vulnerable to cyberattacks. Security patches protect your devices and their data by applying the latest updates to guard against the latest threats.

Some of the benefits of regularly applying security patches include:

- Reducing exposure to cyberattacks
- Avoiding lost productivity
- Protecting data from malware like ransomware
- Avoiding worm infections that use security loopholes to spread over the network

Apache HTTPD security configurations

On Windows, Apache HTTPD is named PEM HTTPD. The Apache HTTPD configuration file is `pem.conf` and the SSL configuration file is `httpd-ssl-pem.conf` . Both configuration files are in the `<Apache_Installation_Path>/conf/addons` directory.

On Linux, the Apache HTTPD configuration file is `edb-pem.conf` and the SSL configuration file is `edb-ssl-pem.conf` . Both configurations files are in the `<Apache_Installation_Path>/conf.d` directory.

Disable SSLv2 and SSLv3

Disable SSL versions SSLv2, SSLv3, TLS 1, and TLS 1.1. These versions are the most vulnerable and have cryptographic concerns.

To disable the versions, add the following command to the Apache HTTPD configuration file:

```
SSLProtocol -ALL +TLSv1.2
```

Restart the web server to apply the changes to the configuration file.

PEM adds the following lines to the SSL configuration file to allow for the use of TLS 1.2 as security:

```
SSLProtocol -All TLSv1.2
```

```
SSLProxyProtocol -All TLSv1.2
```

Secure HTTPD with SSL certificates

We recommend an additional layer of SSL security for the web application.

During PEM installation, PEM generates and uses self-signed certificates for the Apache/HTTPD server. If you want to use your own SSL certificate for PEM, you must update the Apache configuration file.

On Linux, you need to open the Apache HTTPD configuration file (`edb-ssl-pem.conf`) in a text editor and be a user with write permission on the file. You must also change the server name and file names in the configuration file to match your certificate files.

Update these two SSL directives in the PEM VirtualHost section:

- `SSLCertificateFile` is your DigiCert certificate file (for example, your_domain_name.crt).
- `SSLCertificateKeyFile` is the .key file generated when you created the CSR (for example, your_private.key).

For example, make the following updates:

```
SSLEngine on
SSLCertificateFile /path/to/your_domain_name.crt
SSLCertificateKeyFile /path/to/your_private.key
```

You can also replace the HTTPD self-signed SSL certificates with trusted CA-signed certificates in PEM. For instructions, see this Postgres tutorial .

Disable web server information exposure

We recommend that you disable all web server signatures as part of web server security. To disable the web server signature, add the following parameters to the Apache HTTPD configuration file. PEM disables exposure of the information by adding the parameters to the Apache HTTPD configuration file.

```
ServerTokens Prod
ServerSignature Off
```

The `ServerTokens` directive controls the server response header field, which returns to the client. We recommend that you hide the Apache server version by adding this parameter in the Apache HTTPD configuration file.

The `ServerSignature` directive includes a footer for server-produced documents. The footer contains information regarding the Apache configuration, like the Apache and operating system version. To limit the exposure of this information, we recommend that you disable the directive in the Apache HTTPD configuration file.

Restart the web server to apply any changes to the Apache HTTPD configuration file.

Disable directory listing

The directory listing allows an attacker to view the complete contents of the directory. By default, the web server enables this listing, allowing an attacker to discover and view any file. This listing might lead to the attacker reverse engineering an application to obtain the source code, analyze it for possible security flaws, and discover more information about an application.

To avoid this risk, disable the directory listing by setting the `Options` directive in the Apache HTTPD configuration file. PEM disables the directory listing by setting the following option in the web server configuration file:

```
<Directory /application/directory> Options -Indexes </Directory>
```

Restart the web server to apply the changes made to the configuration file.

Restrict the access to a network or IP address

Apache allows you to provide the client hostnames or IP addresses that can access the application. To give access to an IP address or network, enter the network address in the `Allow` directive.

```
<Directory /application/hostname>
Options None
AllowOverride None
Order deny,allow
Deny from all
Allow from 192.168.0.0/24
</Directory>
```

The `ALLOWED_HOSTS` configuration parameter in the application configuration file sets the allowed hosts by IP address. The application configuration file (`config_local.py`) is located in `<PEM_INSTALLATION_PATH>/web` .

PEM allows all the hosts to connect with the application.

For example:

```
#  You can set the range of IP addresses in the configuration file:

ALLOWED_HOSTS = [225.0.0.0/8, 226.0.0.0/7, 228.0.0.0/6]

# You can set the IP adresses to allow a host on a subnet level in the configuration␣
↪file:
ALLOWED_HOSTS = [192.0.2.0/28, ::192.0.2.0/124]

# You can set a specific individual host address (based on the IP address) in the␣
↪configuration file:
ALLOWED_HOSTS = [127.0.0.1, 192.168.0.1]
```

To apply the application configuration file changes, restart the web server.

Cross-site tracing

You can use two HTTP methods to debug the web server connections: TRACE and TRACK. When an HTTP TRACE request is sent to a supported web server, the server responds and echoes the data passed to it, including any HTTP headers. We recommend that you disable these methods in the Apache configuration.

To disable the TRACE method for all virtual hosts, add the following line to the Apache HTTPD configuration file:

```
TraceEnable off
```

To disable these methods for a specific virtual host, add the following lines for each virtual host in the Apache configuration file. PEM also adds the following lines to the Apache HTTPD configuration file:

```
RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK|OPTIONS)
RewriteRule .\* - [F]
```

Run web server from a nonprivileged user account

Running the Apache web service as a root user can create a security issue. We recommend that you run the web server as a unique nonprivileged user. Doing so helps to secure any other services running during a security breach.

PEM runs as a WSGI application. To delegate the WSGI applications that are running, create distinct daemon processes using the `WSGIDaemonProcess` directive.

On Linux, the Apache web server starts as the root user. Meanwhile, PEM runs the daemon processes as the pem user. On Windows, the `WSGIDaemonProcess` directive and features aren't available. During the installation, PEM HTTPD installs as a service. For this service installation to succeed, you must be a member of the Administrators group.

By default, the Apache services run as the system user (the `LocalSystem` account).

Customize security HTTP headers in PEM web server

PEM contains a separate configuration file to fix the following security issues. We recommend that you override the configuration only of `config_local.py` and not of `config.py` . The `config_local.py` file is present only on some systems. You must create it to override the application-level configurations. During a PEM upgrade, you can't overwrite `config_local.py` , but you can overwrite changes in `config.py` and `config_distro.py` . Remove `config_local.py` after uninstalling PEM.

By default, `config_local.py` is in `/usr/edb/pem/web` on Linux and at `C:\ProgramFiles\edb\pem\server\share\web` on Windows.

Host header injection attacks

HTTP host header attacks exploit vulnerable websites that handle the host header value in an unsafe way. If the server implicitly trusts the host header and fails to validate or escape it properly, an attacker can use this input to inject harmful payloads that manipulate server-side behavior. The web applications typically don't know the domain they are deployed on unless specified in a configuration file during setup. When they need to know the current domain, for example, they can resort to retrieving the domain from the host header to generate an absolute URL. The host header is a potential vector for exploiting a range of other vulnerabilities, most notably web cache poisoning and SQL injections.

X-Frame-Options

X-Frame-Options indicate whether a browser is allowed to render a page in an <iframe> tag. It specifically protects against clickjacking. PEM has a host validation `X_FRAME_OPTIONS` option to prevent these attacks, which you can configure in the `config_local.py` file. The default is:

```
X_FRAME_OPTIONS = "SAMEORIGIN"
```

Content-Security-Policy

Content-Security-Policy is part of the HTML5 standard. It provides a broader range of protection than the X-Frame-Options header, which it replaces. It is designed so that website authors can whitelist domains. The authors can load resources (like scripts, stylesheets, and fonts) from the whitelisted domains and also from domains that can embed a page.

PEM has a host validation `CONTENT_SECURITY_POLICY` option to prevent attacks, which you can configure in the `config_local.py` file. The default is:

```
CONTENT_SECURITY_POLICY = "default-src https: data: blob: unsafe-inline 'unsafe-eval;"
```

Strict-Transport-Security

The Strict-Transport-Security (HSTS) response header can prevent a man-in-the-middle attack. When you enable the option, websites or web applications tell browsers that they accept only HTTPS and not HTTP. The default is:

```
STRICT_TRANSPORT_SECURITY = "max-age=31536000;includeSubDomains"
```

Note: Adding this parameter can cause problems if config is changed. Therefore, we recommend that you add it only after PEM installation is complete and tested.

X-Content-Type-Options

The X-Content-Type-Options response HTTP header is a marker. The server uses the marker to indicate that the MIME types advertised in Content-Type headers can't be changed and followed. The following is a way to opt out of MIME type sniffing, that is, to say that the MIME types are deliberately configured. The default is:

```
X_CONTENT_TYPE_OPTIONS = "nosniff"
```

X-XSS-Protection

Cross-site scripting (XSS) is one of the most common application layer vulnerabilities in the web servers. XSS enables attackers to inject client-side scripts into web pages that other users view. The HTTP X-XSS-Protection response to the header is a feature of Internet Explorer, Chrome, and Safari. It stops pages from loading when they detect reflected cross-site scripting (XSS) attacks. These protections are unnecessary in modern browsers when sites implement a strong Content-Security-Policy that disables the use of inline JavaScript ('unsafe-inline'). However, these protections can still provide protections for users of older web browsers that don't yet support CSP. The default is:

```
X_XSS_PROTECTION = "1; mode=block"
```

To avoid this, add the following options to the Apache configuration file:

```
<IfModule mod_headers.c>
Header set X-XSS-Protection "1; mode=block"
</IfModule>
```

To apply the configuration file changes, restart the web server.

By default, PEM sets `X-XSS-Protection to "1; mode=block"` in the application configuration file, which is located at `/usr/edb/pem/web/config.py` .

To apply the changes, restart the Apache service.

For detailed information on the `config.py` file, see *Managing configuration settings* .

Cookie security

Cookies are small packets of data that a server sends to your browser to store configuration data. The browser sends them and all other requests to the same server, so it's important to know how to secure cookies. Multiple configuration options in `config.py` can make cookies secure. These are the three most important options:

- SESSION_COOKIE_SECURE — The flag prevents cookies from sending over an unencrypted connection. The browser can't add the cookie to any request to a server without an encrypted channel. The browser can add cookies only to connections such as HTTPS. The default is:

```
SESSION_COOKIE_SECURE = True
```

- SESSION_COOKIE_HTTPONLY — By default, JavaScript can read the content of cookies. The `HTTPOnly` flag prevents scripts from reading the cookie. Instead, the browser uses the cookie only with HTTP or HTTPS requests. Hackers can't exploit XSS vulnerabilities to learn the contents of the cookie. For example, the `sessionId` cookie never requires that it be read with a client-side script. So, you can set the `HTTPOnly` flag for `sessionId` cookies. The default is:

```
SESSION_COOKIE_HTTPONLY = True
```

- ENHANCED_COOKIE_PROTECTION — When you set this option to `True` , then a token is generated according to the IP address and user agent. In all subsequent requests, the token recalculates and compares to the one computed for the first request. If the session cookie is stolen and the attacker uses it from another location, the generated token is different. In that case, the extension clears the session and blocks the request. The default is:

```
ENHANCED_COOKIE_PROTECTION = True
```

Note: This option can cause problems when the server deploys in dynamic IP address hosting environments, such as Kubernetes or behind load balancers. In such cases, set this option to *False* .

To apply the changes, restart the Apache service.

For detailed information on `config.py` file, see *Managing configuration settings* .

PEM application security configurations

Session timeout

Insufficient session expiration by the web application increases the exposure of other session-based attacks. The attacker has more time to reuse a valid session ID and hijack the associated session. The shorter the session interval is, the less time an attacker has to use the valid session ID. We recommend that you set the inactivity timeout for the web application to a low value to avoid this security issue.

In PEM, you can set the timeout value for a user session. When there's no user activity for a specified duration on the web console, PEM logs out the user from the web console. A PEM administrator can set the length of time for inactivity. This value is for the whole application and not for each user. To configure the timeout duration, modify the `USER_INACTIVITY_TIMEOUT` parameter in the `config_local.py` file, located in the `<PEM_INSTALLATION_PATH>/web` directory. By default, this functionality is disabled.

For example, to specify for an application to log out a user after 15 minutes of inactivity, set:

```
USER_INACTIVITY_TIMEOUT = 900
```

Note: The timeout value is specified in seconds.

To apply the changes, restart the Apache service.

For detailed information on the `config.py` file, see *Managing configuration settings* .

RestAPI header customization

You can customize the RestAPI token headers to meet your requirements. The default values aren't exposed by the `config.py` file. Customize the following headers in the `config_local.py` file:

PEM_HEADER_SUBJECT_TOKEN_KEY

This configuration option allows you to change the HTTP header name to get the generated token. By default, when you send a request to create a token, the server response has an `X-Subject-Token` header. This header contains the value of a newly generated token. If you want to customize the header name, then you can update the `config_local.py` file:

```
PEM_HEADER_SUBJECT_TOKEN_KEY = Pem-RestAPI-Generate-Token
```

This command produces the following output:

```
curl -ik -X POST -d {"username":"enterprisedb","password":"edb"} -H "Content-Type:␣
↪application/json" https://localhost:8443/pem/api/token/
HTTP/1.1 201 CREATED
Date: Thu, 29 Oct 2020 11:03:48 GMT
Server: Apache
Content-Length: 326
Pem-RestAPI-Generate-Token: 997aef95-d46d-4d84-932a-a80146eaf84f
```

PEM_HEADER_TOKEN_KEY

This configuration option allows you to change the HTTP request header name. With this header name, you can send the token to the PEM server. By default, when you send a request to generate a token, the token header name is `X-Auth-Token` . If you want to customize the RestAPI request header name, then you can update the `config_local.py` file:

```
PEM_HEADER_TOKEN_KEY = Pem-Token
```

This setting allows you to send the token:

```
$ curl -Lk -X GET -H "Pem-Token: gw5rzaloxydp91ttd1c97w24b5sv60clic24sxy9" https://
↪localhost:8443/pem/api/v4/agent
```

PEM_TOKEN_EXPIRY

This configuration option allows you to change the PEM RestAPI token expiry time after it's generated. By default, the token expiry time is set to 20 minutes (1200 seconds). If you want to change the token expiry time to 10 minutes, then you can update the `config_local.py` file:

```
PEM_TOKEN_EXPIRY = 600
```

To apply the changes, restart the Apache service.

Role-based access control in PEM

Role-based access control (RBAC) restricts application access based on a user' s role in an organization and is one of the primary methods for access control. The roles in RBAC refer to the levels of access that users have to the application. Users are allowed to access only the information needed to do their jobs. Roles in PEM are inheritable and additive, rather than subscriptive. In other words, as a PEM admin you need to grant the lowest level role to the user and then grant the roles the user needs to perform their job. For example, to give access only to SQL profiler:

```
CREATE ROLE user_sql_profiler WITH LOGIN NOSUPERUSER NOCREATEDB NOCREATEROLE INHERIT␣
↪NOREPLICATION CONNECTION LIMIT -1 PASSWORD xxxxxx;
GRANT pem_user, pem_comp_sqlprofiler TO user_sql_profiler;
```

For detailed information on roles, see *PEM Roles* .

SQL/Protect plugin

Often, preventing an SQL injection attack is the responsibility of the application developer, while the database administrator has little or no control over the potential threat. The difficulty for database administrators is that the application must have access to the data to function properly.

SQL/Protect is a module that allows a database administrator to protect a database from SQL injection attacks. SQL/Protect examines incoming queries for typical SQL injection profiles in addition to the standard database security policies.

Attackers can perpetrate SQL injection attacks with several different techniques. A specific signature characterizes each technique. SQL/Protect examines queries for unauthorized relations, utility commands, SQL tautology, and unbounded DML statements. SQL/Protect gives the control back to the database administrator by alerting the administrator to potentially dangerous queries and then blocking those queries.

Note: This plugin works only on the EDB Postgres Advanced Server server, so this is useful only when your PEM database is hosted on the EDB Postgres Advanced Server server.

For detailed information about the SQL Profiler plugin, see *Upgrading SQL Profiler* .

Password management

One security tip for PEM administrative users is to change your PEM login passwords to something new regularly. Changing your password:

- Prevents breaches of multiple accounts
- Prevents constant access
- Prevents the use of saved passwords on a physically unsecured system
- Limits access gained by keystroke loggers

Run pemAgent jobs with a non-root user

In most cases, pemAgent is installed as a root user and runs as a daemon process with root privileges. By default, PEM disables running the scheduled jobs/task. PEM provides support for running scheduled jobs as a non-root user by changing the pemAgent configuration file.

To run scheduled jobs as a non-root user, modify the entry for the `batch_script_user` parameter in the `agent.cfg` file and specify the user to run the script. You can either specify a non-root user or root user identity. If you don' t specify a user, or the specified user doesn' t exist, then the script doesn' t execute. Restart the agent after modifying the file. If a non-root user is running `pemagent` , then the value of `batch_script_user` is ignored, and the same non-root user used for running the `pemagent` executes the script.

To invoke a script on a Windows system, set the registry entry for `AllowBatchJobSteps` to `true` and restart the PEM agent. PEM registry entries are located in:

```
HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent
```

Changing the pemAgent and PEM backend database server certificates

By default, when you install PEM, the installer generates and uses self-signed certificates for the pemAgent and PEM database server. PemAgent uses these certificates when connecting to the PEM database server. To use your own SSL certificate for the pemAgent and PEM database server, see Managing certificates .

Note: PEM doesn't support placing the SSL CA certificates at a custom location. Don't change the location of *ca_certificate.crt* and *ca_key.key* .

2.2.5 Web server installation options

During the PEM server installation, you can specify your hosting preferences for the Apache web server.

PEM server and Apache web server on separate hosts

1. Install the PEM server on both the hosts. See *Installing the PEM server and PEM-HTTPD on separate hosts* .
2. Configure the PEM server host by selecting the **Database** option on the first host.
3. Configure an Apache web server by selecting the **Web Services** option on the second host.

For more information about configuring a PEM server, see *Configuring the PEM server on Linux* .

Note: For production environments, best practice is to have the PEM server and Apache web server on separate hosts.

PEM server and Apache web server on the same host

1. Install the PEM server. See *Installing the PEM server and PEM-HTTPD on separate hosts* .
2. Run the configuration script. Select the **Web Services and Database** option to install the PEM server and Apache web server on the same host. See *Configuring the PEM server on Linux* .

Note: For nonproduction environments, best practice is to have the PEM server and Apache web server on same hosts.

2.2.6 How PEM licensing works

PEM is intended to be used for monitoring and managing Postgres instances supported by EDB. As such, if you have a subscription that entitles you to use PEM, adding it to your stack to monitor such instances generally incurs no additional license costs.

Licensing rules

To use PEM, you must have an EDB Standard or Enterprise subscription.

Any Postgres servers you monitor using PEM must be covered by an EDB Standard or Enterprise subscription. This is true whether you're installing the PEM agent locally, through remote monitoring, or by making a client connection from the PEM web application.

Non-Postgres servers that don't otherwise consume cores (for example PGD Proxy nodes or machines running Barman) may be monitored by PEM. No additional cores are consumed by monitoring such servers with PEM.

The Postgres instance used as the PEM backend doesn't consume cores from your subscription. Likewise, no cores are consumed by any other PEM components. All these components are covered by your support agreement.

Examples

Adding PEM to a fully supported environment

Suppose a customer with an Enterprise subscription for 32 cores has 8 x 4-core servers running EDB Postgres Advanced Server, thereby fully consuming their 32 cores. This customer can install PEM on a fifth server and use it to monitor their 8 EDB Postgres Advanced Server servers. This configuration requires no change to their subscription, as the PEM server doesn't consume cores, and the monitored EDB Postgres Advanced Server instances are already fully covered.

Likewise, if this customer added a Barman server and connected it to PEM for monitoring, they would consume no additional cores.

Adding unsupported servers to PEM

Suppose a customer with a Standard subscription for 36 cores has PEM and 6 x 6-core servers running PostgreSQL covered by EDB support, thereby consuming their 36 cores, as in the previous example.

The customer also has 10 x 2-core PostgreSQL servers that aren't covered by any EDB subscription. These servers must not be monitored by PEM, as they aren't covered by a Standard or Enterprise subscription. If the customer wants to monitor these servers, they must add 20 more cores to their subscription.

Note: An EDB Community 360 subscription for the additional 20 cores isn't sufficient. Servers monitored by PEM must be covered by Standard or Enterprise.

CHAPTER 3

Installing and Upgrading

3.1 Installing Postgres Enterprise Manager server

Select a link to access the applicable installation instructions:

3.1.1 Linux x86-64 (amd64)

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64* , *Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64* , *Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky* , *Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky*
- *Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky* , *Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*

Debian and derivatives

- *Installing Postgres Enterprise Manager server on Ubuntu 22.04 x86_64* , *Installing Postgres Enterprise Manager server on Ubuntu 20.04 x86_64*
- *Installing Postgres Enterprise Manager server on Debian 12 arm64* , *Installing Postgres Enterprise Manager server on Debian 11 x86_64*

3.1.2 Linux Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le)

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64* , *Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*

3.1.3 Linux Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64)

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*

Debian and derivatives

- *Installing Postgres Enterprise Manager server on Debian 12 arm64*

3.1.4 Windows

- *Windows Server 2019*

Dependencies of the PEM Server and Agent on Linux

The PEM Server and Agent packages for Linux have dependencies on various system libraries. These dependencies are detailed below for reference.

Note: A PEM Agent is always installed alongside PEM Server, so all dependencies must be present on hosts where PEM Server (either the database or the web application) is installed.

Typically, PEM is built against the latest version of each dependency available from the vendor repository for a given platform and architecture. In some cases, PEM requires a newer version of a library than is available in the vendor repository. In these cases a newer version of the package, prefixed with `edb-` is sourced from EDB' s repositories.

Note: This information is provided for reference. Packages from vendor repositories are not supported or patched by EDB. Refer to your operating system documentation or support provider for details of these packages.

Because these dependencies are updated frequently, the tables below are valid only for the latest patch release of PEM.

Python 3 and mod_wsgi (a Python module for Apache HTTPD) are required for PEM Server.

Platform	Architecture	Python/mod_wsgi package	Python version	Python path
RHEL 7	x86_64	*edb-python39/edb-python39-mod-wsgi*	3.9	*/usr/libexec/edb-python39/bin/python3*
	ppc64le	*edb-python39/edb-python39-mod-wsgi*	3.9	*/usr/libexec/edb-python39/bin/python3*
RHEL 8	x86_64	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
	ppc64le	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
	s390x	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
RHEL 9	x86_64	*python39/python39-mod_wsgi*	3.9	*/usr/bin/python3*
	ppc64le	*python39/python39-mod_wsgi*	3.9	*/usr/bin/python3*
	s390x	*python39/python39-mod_wsgi*	3.9	*/usr/bin/python3*
	arm64	*python39/python39-mod_wsgi*	3.9	*/usr/bin/python3*
SLES 15	x86_64	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
	ppc64le	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
	s390x	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
Ubuntu 20	amd64	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
Ubuntu 22	amd64	*python310//libapache2-mod-wsgi-py3*	3.10	*/usr/bin/python3*
Debian 10	amd64	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
Debian 11	amd64	*edb-python310/edb-python310-mod-wsgi*	3.10	*/usr/libexec/edb-python310/bin/python3*
Debian 12	amd64/arm	*python311/libapache2-mod-wsgi-py3*	3.11	*/usr/bin/python3*

OpenSSL

The PEM Server and Agent require OpenSSL.

Platform	Architecture	package-version
RHEL 7	x86_64	*openssl-1.0.2k*
	ppc64le	*openssl-1.0.2k*
RHEL 8	x86_64	*openssl-1.1.1k*

continues on next page

Table 2 – continued from previous page

Platform	Architecture	package-version
	ppc64le	*openssl-1.1.1k*
	s390x	*openssl-1.1.1k*
RHEL 9	x86_64	*openssl-3.0.7*
	ppc64le	*openssl-3.0.7*
	s390x	*openssl-3.0.7*
	arm64	*openssl-3.0.7*
SLES 12	x86_64	*openssl-1.0.2p*
	ppc64le	*openssl-1.0.2p*
	s390x	*openssl-1.0.2p*
SLES 15	x86_64	*openssl-1.1.1l*
	ppc64le	*openssl-1.1.1l*
	s390x	*openssl-1.1.1l*
Ubuntu 20	amd64	*openssl-1.1.1f*
Ubuntu 22	amd64	*openssl-3.0.2*
Debian 10	amd64	*openssl-1.1.1n*
Debian 11	amd64	*openssl-1.1.1w*
Debian 12	amd64/arm64	*openssl-3.0.11*

Libcurl

The PEM Agent requires libcurl.

Platform	Architecture	package-version
RHEL 7	x86_64	*libcurl-pem-8.4.0*
	ppc64le	*libcurl-pem-8.4.0*
RHEL 8	x86_64	*libcurl-pem-8.4.0*
	ppc64le	*curl-7.61.1*
	s390x	*curl-7.61.1*
RHEL 9	x86_64	*curl-7.76.1*
	ppc64le	*curl-7.76.1*
	s390x	*curl-7.76.1*
SLES 15	x86_64	*curl-8.0.1*
	ppc64le	*curl-8.0.1*
	s390x	*curl-8.0.1*
Ubuntu 20	amd64	*libcurl4-7.68.0*
Ubuntu 22	amd64	*libcurl4-7.81.0*
Debian 10	amd64	*libcurl4-7.64.0*
Debian 11	amd64	*libcurl4-7.74.0*
Debian 12	amd64/arm64	*libcurl4-7.88.1*

SNMP++

The PEM Agent requires SNMP++.

Platform	Architecture	package-version
RHEL 7	x86_64	*snmp++-3.4.2*
	ppc64le	*snmp++-3.4.2*
RHEL 8	x86_64	*snmp++-3.4.2*
	ppc64le	*edb-snmp++-3.4.10*
	s390x	*edb-snmp++-3.4.7*
RHEL 9	x86_64	*edb-snmp++-3.4.10*
	ppc64le	*edb-snmp++-3.4.10*
	s390x	*edb-snmp++-3.4.10*
SLES 15	x86_64	*edb-snmp++-3.4.10*
	ppc64le	*edb-snmp++-3.4.10*
	s390x	*edb-snmp++-3.4.7*
Ubuntu 20	amd64	*edb-snmp++-3.4.10*
Ubuntu 22	amd64	*edb-snmp++-3.4.10*
Debian 10	amd64	*edb-snmp++-3.4.10*
Debian 11	amd64	*edb-snmp++-3.4.10*
Debian 12	amd64/arm64	*edb-snmp++-3.4.10*

Boost libraries

The PEM Agent requires the Boost libraries.

Platform	Architecture	package-version
RHEL 7	x86_64	*boost169-system-1.69.0*
	ppc64le	*None boost package*
RHEL 8	x86_64	*boost169-system-1.69.0*
	ppc64le	*boost-system-1.66.0*
	s390x	*boost-system-1.66.0*
RHEL 9	x86_64	*boost-system-1.75.0*
	ppc64le	*boost-system-1.75.0*
	s390x	*boost-system-1.75.0*
SLES 15	x86_64	*libboost_regex1_66_1-1.66.0*
	ppc64le	*libboost_regex1_66_1-1.66.0*
	s390x	*libboost_regex1_66_1-1.66.0*
Ubuntu 20	amd64	*libboost-system1.71.0-1.71.0*
Ubuntu 22	amd64	*libboost-system1.74.0-1.74.0*
Debian 10	amd64	*libboost-system1.67.0-1.67.0*
Debian 11	amd64	*libboost-system1.74.0-1.74.0*
Debian 12	amd64/arm64	*libboost-system1.74.0-1.74.0*

Installing Postgres Enterprise Manager server on Linux x86 (amd64)

Operating system-specific install instructions are described in the corresponding documentation:

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky*
- *Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*

Debian and derivatives

- *Installing Postgres Enterprise Manager server on Ubuntu 22.04 x86_64*
- *Installing Postgres Enterprise Manager server on Ubuntu 20.04 x86_64*
- *Installing Postgres Enterprise Manager server on Debian 12 arm64*
- *Installing Postgres Enterprise Manager server on Debian 11 x86_64*

Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 x86_64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on RHEL 8 or OL 8 x86_64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser

to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.

- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you' re using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they' re different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you’re using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you’re using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you’re using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you’re using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you’re using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where `<x>` is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on SLES 15 x86_64

Note: Postgres Enterprise Manager 8.3 and later is supported on SLES.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they' re different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you' re using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you' re using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo zypper install edb-as<x>-server-sslutils
```

- If you' re using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo zypper install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo zypper install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo zypper update
```

Install the package

```
sudo zypper -n install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Ubuntu 22.04 x86_64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo apt-get install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo apt-get install postgresql-<x>-sslutils
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo apt-get install edb-postgresextended-sslutils-<x>
```

Note: Ubuntu 20 changed the requirements for accepting certificates.

```
- If you want to install the PEM agent on a machine with an old version of sslutils,␣
↪then you must upgrade sslutils to 1.3. Version 1.3 has a 4096-bit RSA key and sha256␣
↪signature algorithm support added to it. - If you don't upgrade sslutils to 1.3, then␣
↪PEM agent might fail to connect to the PEM backend database server, and it might log␣
↪the error "ca md too weak."
```

```
iptables -t filter -A INPUT -p TCP --dport 8443 -j ACCEPT
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo apt-get update
```

Install the package

```
sudo apt-get -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Ubuntu 20.04 x86_64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo apt-get install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo apt-get install postgresql-<x>-sslutils
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo apt-get install edb-postgresextended-sslutils-<x>
```

Note: Ubuntu 20 changed the requirements for accepting certificates.

```
- If you want to install the PEM agent on a machine with an old version of sslutils,␣
↪then you must upgrade sslutils to 1.3. Version 1.3 has a 4096-bit RSA key and sha256␣
↪signature algorithm support added to it. - If you don't upgrade sslutils to 1.3, then␣
↪PEM agent might fail to connect to the PEM backend database server, and it might log␣
↪the error "ca md too weak."
```

```
iptables -t filter -A INPUT -p TCP --dport 8443 -j ACCEPT
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo apt-get update
```

Install the package

```
sudo apt-get -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Debian 12 x86_64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo apt-get install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo apt-get install postgresql-<x>-sslutils
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo apt-get install edb-postgresextended-sslutils-<x>
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo apt-get upgrade
```

Install the package

```
sudo apt-get -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Debian 11 x86_64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they' re different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you' re using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you' re using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo apt-get install edb-as<x>-server-sslutils
```

- If you' re using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo apt-get install postgresql-<x>-sslutils
```

- If you' re using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo apt-get install edb-postgresextended-sslutils-<x>
```

Note: Debian 10 changed the requirements for accepting certificates.

```
- If you want to install the PEM agent on a machine with an old version of sslutils,␣
↪then you must upgrade sslutils to 1.3. Version 1.3 has a 4096-bit RSA key and sha256␣
↪signature algorithm support added to it. - If you don't upgrade sslutils to 1.3, then␣
↪PEM agent might fail to connect to the PEM backend database server, and it might log␣
↪the error "ca md too weak"
```

```
iptables -t filter -A INPUT -p TCP --dport 8443 -j ACCEPT
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo apt-get update
```

Install the package

```
sudo apt-get -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le)

Operating system-specific install instructions are described in the corresponding documentation:

Red Hat Enterprise Linux (RHEL)

- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*

Installing Postgres Enterprise Manager server on RHEL 9 ppc64le

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on RHEL 8 ppc64le

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on SLES 15 ppc64le

Note: Postgres Enterprise Manager 8.3 and later is supported on SLES.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they' re different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you' re using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you' re using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo zypper install edb-as<x>-server-sslutils
```

- If you' re using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo zypper install sslutils_<x> postgresql<x>-contrib
```

- If you' re using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo zypper install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you' re using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo zypper update
```

Install the package

```
sudo zypper -n install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64)

Operating system-specific install instructions are described in the corresponding documentation:

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64*

Debian and derivatives

- *Installing Postgres Enterprise Manager server on Debian 12 arm64*

Installing Postgres Enterprise Manager server on RHEL 9 or OL 9 arm64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo dnf install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo dnf install sslutils_<x> postgresql<x>-contrib
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo dnf install edb-postgresextended<x>-sslutils edb-postgresextended<x>-contrib
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp

firewall-cmd --reload
```

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo dnf upgrade
```

Install the package

```
sudo dnf -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Debian 12 arm64

You can install PEM on a single server, or you can install the web application server and the backend database on two separate servers. You must prepare your servers for PEM installation.

After fulfilling the prerequisites and completing the installation procedure described in the following steps, you must *Configure the email notifications* PEM. If you're using two servers, install and configure PEM on both servers.

Prerequisites

Before you begin the installation process:

1. Install a supported Postgres instance for PEM to use as a backend database.

You can install this instance on the same server to be used for the PEM web application or on a separate server. You can also use an existing Postgres instance if it is configured as detailed in the next steps.

2. Configure authentication on the Postgres backend database by updating the `pg_hba.conf` file.

Make the following changes manually, prior to configuration. (Additional changes are necessary during *Managing configuration settings* .)

- To create the relations required for PEM, the PEM configuration script connects to the Postgres backend database as a superuser of your choice using password authentication. This requires you to permit your chosen superuser to authenticate using a password. This user must be able to connect from any location where you run the configuration script. In practice, this means the server where the backend database is located and the server where the PEM web application is to be installed, if they're different.
- To allow the chosen superuser to connect using password authentication, add a line to `pg_hba.conf` that allows `host` connections using `md5` or `scram-sha-256` authentication, such as `host all superusername 127.0.0.1/32 scram-sha-256` .

Note: If you're using EDB Postgres Advanced Server, see Modifying the pg_hba.conf file .

If you're using PostgreSQL, see Client Authentication . .. ::
2_blank_1_3. Verify that the *sslutils* extension is installed on your Postgres server.

If you're using PostgreSQL or EDB Postgres Extended Server on RHEL/AlmaLinux/Rocky Linux or SLES, you also need to install the `hstore contrib` module.

- If you're using EDB Postgres Advanced Server, you can install the `sslutils` extension as follows, where <x> is the EDB Postgres Advanced server version.

```
sudo apt-get install edb-as<x>-server-sslutils
```

- If you're using PostgreSQL, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the PostgreSQL version.

```
sudo apt-get install postgresql-<x>-sslutils
```

- If you're using EDB Postgres Extended Server, you can install the `sslutils` and, if required, `hstore` modules as follows, where <x> is the EDB Postgres Extended Server version.

```
sudo apt-get install edb-postgresextended-sslutils-<x>
```

4. If you're using a firewall, allow access to port 8443 on the server where the PEM web application will be located:

5. Make sure the components Postgres Enterprise Manager depends on are up to date on all servers. You can do this by updating the whole system using your package manager as shown below. If you prefer to update individual packages, a full list of dependencies is provided in *Dependencies of the PEM Server and Agent on Linux* .

```
sudo apt-get upgrade
```

Install the package

```
sudo apt-get -y install edb-pem
```

Initial configuration

```
#  You can configure the PEM server using the following command:

sudo /usr/edb/pem/bin/configure-pem-server.sh
```

For more details, see *Configuring the PEM server on Linux* .

Note:

- The operating system user pem is created while installing the PEM server. The PEM server web application is a WSGI application, which runs under Apache HTTPD. The pem application data and the session is saved to this user's home directory.

Supported locales

Currently, the Postgres Enterprise Manager server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale or a separator character other than a period might cause errors.

Installing Postgres Enterprise Manager server on Windows

On Windows, the PEM server graphical installer installs and configures the PEM server, a PEM agent, and the software required to connect to the PEM web interface with your choice of browser.

At the heart of each PEM installation is the PEM server. In a production environment, the server typically is a dedicated machine, monitoring a large number of Postgres servers or a smaller number of busy servers.

The PEM server backend database can be an EDB distribution of the PostgreSQL or EDB Postgres Advanced Server database server or an existing Postgres server installed from another source. The Postgres backend database server must be version 11 or later and contains a database named `pem` , which is used by the PEM server as a repository.

- If you want to use an existing Postgres server to host the PEM server, the PEM server installer can create the `pem` database on the Postgres host. You must manually satisfy the software prerequisites if you choose to use an existing server.

For more information about using an existing Postgres server to host the PEM server backend database, see *Installing the PEM server on an existing Postgres servers* .

- If you don't want to use an existing installation of Postgres as the PEM server host, the PEM server installer can:
- Install PostgreSQL
- Satisfy the server host's software prerequisites
- Create an instance (a PostgreSQL database cluster) that contains the `pem` database, which is the simplest PEM server installation option
- PEM-HTTPD is made available for Postgres installations through the PEM server installer or the StackBuilder utility. If PEM-HTTPD is already installed on the host, the PEM server installer reviews and updates the existing installation if needed. If the PEM server host doesn't contain an existing PEM-HTTPD installation, the PEM server installer adds it.
- Before installing the PEM server, you must decide if you want to run PostgreSQL and PEM-HTTPD on the same host or on separate hosts. If you intend to run the PostgreSQL database server and PEM-HTTPD on different hosts, then you must run the PEM server installer twice, once on each host. See Installing the PEM server and PEM-HTTPD on separate hosts .
- For detailed information about installing the PEM server and PEM-HTTPD on the same host, see Installing the PEM server and PEM-HTTPD on the same host .
- For detailed information about installing and configuring a standalone PEM agent, see *Installing Postgres Enterprise Manager agent on Windows* .
- Language pack installers contain supported languages that you can use with EDB Postgres Advanced Server and EDB PostgreSQL database installers. The language pack installer allows you to install Perl, TCL/TK, and Python without installing supporting software from third-party vendors. For more information about installing and using the language pack, see EDB Postgres language pack .
- For troubleshooting the installation or configuration of the PEM agent, see *Troubleshooting agent issues* .

The PEM server installer also installs the software required to access the server using the PEM web interface. You can access the web interface with a supported version of your browser. You can use the web interface to:

- Review information about objects that reside on monitored servers
- Manage databases and database objects that reside on monitored servers
- Review statistical information gathered by the PEM server

Note: If you're using SSL certificates, make sure that all the SSL certificates are in the data directory in the backend database server. If the certificates aren't in the data directory, then the PEM server's configuration might fail because it looks in the data directory while configuring the PEM server.

Installing the PEM server and PEM-HTTPD on separate hosts

To use separate hosts for the PEM server backend database and PEM-HTTPD:

1. Invoke the PEM server installer on the host of the Postgres server that will contain the `pem` database. During the installation, select the **Database** option on the Advanced Options dialog box, and provide connection information for the Postgres server.
2. Modify the `pg_hba.conf` file of the Postgres installation on which the PEM server (and `pem` database) resides, allowing connections from the host of the PEM-HTTPD server.
3. Invoke the PEM server installer on the host of the PEM-HTTPD server, selecting the **Web Services** option on the Installation Type dialog box.

To start the installation:

1. On a Windows system, invoke the PEM server installer. Right-click the installer and select **Run as Administrator**. The installer displays a Welcome screen. Select **Next**.
2. Review the license agreement before selecting the appropriate radio button and accepting the agreement. Select **Next**.
3. Use fields on the Installation Directory dialog box to specify the directory for the PEM server to reside and to open the Advanced Options dialog box during installation:

- By default, the PEM server is installed in `C:\Program Files\edb\pem` on Windows. Accept the default location, or use the **Installation Directory** field to open a browser dialog box and select the directory to install the PEM server in.
- To install the PEM server and PEM-HTTPD on separate hosts, use the Advanced Options dialog box to specify the installation type (**Web Services** or **Database**). Select **Show advanced options** to include the Advanced Options dialog box in the installation process.
- Select **Next**.

1. Use the Advanced Options dialog box to specify the components that you want to install:

- Select **Web Services and Database** to indicate that the Postgres server and PEM-HTTPD will both reside on the current host. If you select **Web Services and Database**, the PEM server installer lets you specify the Postgres server to use for the PEM server before checking for a PEM-HTTPD installation.
- Select **Web Services** to install PEM-HTTPD on the current host, while using a Postgres database server that resides on another host to host the PEM server and `pem` database.

Note: Before using the **Web Services** option to install PEM-HTTPD, you must complete the PEM server installation process on the host of the PEM server and *pem* backend database. Select **Database** on the Advanced Options dialog box and modify the connection properties of the *pg_hba.conf* file on the PEM server.

This option invokes the installation steps described in *Installing web services* .

- Select **Database** to use an existing Postgres server (version 11 or later) or to install only the database server that's distributed with the PEM server installer. This option invokes the installation steps described in *Specifying a database host* .

1. After selecting an installation option, select **Next**.

Specifying a database host

1. Select **Database** on the Advanced Options dialog box to specify connection information for the host where the PEM server backend database (named `pem`) will reside. Select **Next**.
2. Use the list on the Database Server Selection dialog box to select a host for the PEM server backend database. You can:

- Select a host from existing Postgres installations that reside on the current host.

Note: You might need to add the *sslutils* package to your installation.`2_blank_4_-` Select **PostgreSQL <x>** to install the Postgres server that's distributed with the PEM server installer, where *<x>* is the PostgreSQL database server version. If you decide to use the version of PostgreSQL that's bundled with the PEM server installer, the EnterpriseDB one-click PostgreSQL installer opens and walks you through the installation.`1_blank_4_-` Select **Other Database Server** to specify connection information for a Postgres server that wasn't installed using a one-click graphical installer

from EnterpriseDB. For information about the software prerequisites for the PEM server database host, see *Preparing the Postgres server* .`2_blank_4_-` Select **Next**.

If the PEM server will reside on an existing Postgres server, the Database Server Installation Details dialog box opens.

1. The information required on the Database Server Installation Details dialog box can vary. The PEM server installer asks you to provide only the information about the selected installation that it can' t locate:

- Specify the name of a Postgres database superuser in the **User** field.
- Specify the password associated with that user in the **Password** field.
- Select **Next**.

1. If prompted, provide the system password for the service account under which the PEM agent will run. Select **Next**.
2. Use the Network Details dialog box to specify the CIDR-style network address from which PEM agents will connect to the server (the client-side address). The specified address is added to the server' s `pg_hba.conf` file.

Note: You can specify more network addresses by manually adding entries to the *pg_hba.conf* file on the PostgreSQL server.

Accept the default (specifying the localhost), or specify a **Network address** range. Select **Next**.

The PEM server installer installs a PEM agent on the host on which the server resides to monitor the server and provide alert processing and garbage collection services. A certificate is also installed in the location specified in the **Agent certificate path** field.

1. You can enter an alternate description or an alternate agent certificate path for the PEM agent or accept the defaults. Select **Next**.
2. The wizard is now ready to install the PEM server. Select **Next** to proceed with the installation.

During the installation process, the installer copies files to the system and sets up the PEM server' s backend database. A confirmation indicates that the `pem` database was created and configured.

1. Select **OK**

When the database portion of the PEM server installation is complete, you can invoke the PEM server on another host to install or upgrade PEM-HTTPD.

Installing web services

Select **Web Services** on the Advanced Options dialog box to either install PEM-HTTPD on the current host or update an existing PEM-HTTPD installation.

Note: The current host might not be the host of the PEM backing database.

Before selecting this option, you must:

- Install the PEM server installer on a host system, during which you specify a backing database for the PEM server.
- Modify the `pg_hba.conf` file on the PEM server database host to allow connections from the PEM-HTTPD host.
- Restart the database server.

1. Select **Web Services** and select **Next**. The PEM server installer checks the current host for existing PEM-HTTPD and LanguagePack installations.
2. If the installer doesn't locate the components, the installer informs you that one or more dependencies are missing. Select **Next** to install the language pack. After installing language pack, the installer invokes the PEM-HTTPD setup wizard.
3. Follow the onscreen instructions of the PEM-HTTPD Setup wizard. When the wizard completes the installation, select **Finish** to open the Database Server Installation Details dialog box.
4. Use the Database Server Installation Details dialog box to provide connection information for the Postgres installation that's hosting the PEM server installation:

- Enter the name or IP address of the PEM server host in the **Host** field.
- Enter the port number on which the Postgres server is listening in the **Port** field.
- Enter the name of a Postgres database superuser in the **User** field.
- Enter the password associated with the Postgres superuser in the **Password** field.

1. Select **Next**.

Before completing the PEM server installation, the installer contacts the database host. For the installation to continue, the `pg_hba.conf` file on the PEM database host must be configured to accept connections from the host of the httpd server, and the firewall must allow a connection. The PEM server installer completes the PEM server installation, adding only those items that must reside on the host of the PEM-HTTPD server.

Installing the PEM server and PEM-HTTPD on the same host

The easiest PEM server installation configuration consists of a PEM backend database server (hosted on a PostgreSQL database installed with the PEM server installer) and a PEM-HTTPD service that reside on the same host. In this configuration, the PEM server installer provides the prerequisite software for the backend host to register the service (on Windows).

1. Invoke the PEM server installer. On a Windows system, right-click the installer icon and select **Run as Administrator**. The installer displays a Welcome screen. Select **Next**.
2. Review the license agreement before selecting the appropriate radio button and accepting the agreement. Select **Next**.
3. Use the Installation Directory dialog box to specify the location of the PEM server:

- By default, the PEM server is installed in `C:\Program Files\edb\pem` on Windows. Accept the default location, or use the **Installation Directory** button to open a dialog box and select the directory to install the PEM server in.
- Use the **Show advanced options** check box to open the Advanced Options dialog box during installation. Use the Advanced Options dialog box when installing the Postgres database server and the PEM-HTTPD on different hosts or if you want the PEM server to reside on an existing Postgres server installation.
- To install the PostgreSQL server package with the installer and PEM-HTTPD on the same host, clear **Show advanced options** and select **Next**.

The PEM server installer performs a preinstallation check for PEM-HTTPD, Language Pack, and PostgreSQL 13. If the installer doesn't locate these packages, it informs you in the Dependency Missing dialog box.

Note: By default EDB Language Pack is installed in *C:edblanguagepackv1* .

1. If the dependencies are missing, the PEM server installer launches the respective installation wizards. Follow the wizard' s onscreen directions each package.

After installing any missing dependencies, the installation process continues by displaying the Database Server Installation Details dialog box. The information provided on the Database Server Installation Details dialog box enables the installer to connect to the PostgreSQL server.

1. Provide the user name and password of a database superuser. After supplying the requested information, select **Next**.
2. After providing the name and password of the Postgres database superuser, you might be prompted for the password to the user account under which the PEM agent will run. If prompted, provide the password, and select **Next**.
3. Use the Network Details dialog box to specify the CIDR-style network address from which the PEM agents will connect to the server (the *client-side* address).

You can specify the address of a network host or a network address range. For example, if you want to monitor database servers with the addresses `192.168.10.23` , `192.168.10.76` , and `192.168.10.184` , enter `192.168.10.0/24` to allow connections with hosts in that network.

The specified address is added to the server' s `pg_hba.conf` file. You can specify more network addresses by manually adding entries to the `pg_hba.conf` file on the PostgreSQL server. Use the first entry as a template.

After you add the network address, select **Next**.

The PEM server installer installs a PEM agent on the host where the server resides to monitor the server and provide alert processing and garbage collection services. A certificate is also installed in the location specified in the **Agent certificate path** field.

1. Enter an alternate description or select an alternate agent certificate path for the PEM agent, or accept the defaults. Select **Next**.
2. The wizard is now ready to install the PEM server. Select **Next** to continue with the installation.

During the installation process, the installer copies files to the system and sets up the database and web services required to run PEM. When the installation completes, a confirmation indicates that: - The web service was configured. - The web service is listening on port 8443. - The `pem` database was created and configured.

1. Select **OK**.

Installing the PEM server on an existing Postgres servers

You can use an existing Postgres server (version 11 or later) to host the PEM server and the `pem` database. Postgres installers and prerequisite software extensions are freely available on the EnterpriseDB website .

You can configure an existing Postgres server for a PEM server installation.

Note: These instructions are guidelines. the actual steps needed to configure your Postgres installation depend on the configuration of your Postgres server.

The following versions of Postgres are preconfigured to contain the `sslutils` extension and a service script. No additional preparation is required to use the following Postgres versions as a PEM backend database server:

- PostgreSQL 11 or later (as bundled with the PEM Server installer)
- EDB Postgres Advanced Server 11 or later

Preparing the Postgres server

Before installing the PEM server on an existing Postgres server, you must:

- Ensure that the Postgres server contains an installation of the `sslutils` extension. For more information, see *Installing the sslutils extension* .
- Register the server with the Windows service manager. For more information, see *Registering the service* .

After preparing the server, you can use the PEM server installer to install PEM on the existing Postgres server.

Installing the sslutils extension

The Postgres server where the PEM server will reside must contain the `sslutils` extension. The `sslutils` package is freely available for download from the EDB website .

When the web page opens, select the link for the `SRC- SSL Utils 1.3` package. When the download completes, extract the file, and copy it into the Postgres installation directory.

Note: You don't need to manually add the *sslutils* extension when using the following Postgres installations:

- PostgreSQL 10 or later (as distributed with the PEM server installer)
- Advanced Server 10 or later

Build `sslutils` with the same compiler that you used to compile the backend Postgres installation. If you' re using a backend Postgres database that was installed on a Windows platform using a PostgreSQL one-click installer (from EnterpriseDB) or an EDB Postgres Advanced Server installer, you must use Visual Studio to build `sslutils` .

While specific details of the installation process vary by platform and compiler, the basic steps are the same. You must:

1. Copy the `sslutils` package to the Postgres installation directory.
2. Open the command line of the appropriate compiler, and navigate to the `sslutils` directory.
3. Use the following commands to build `sslutils` :

```
SET USE_PGXS=1

SET GETTEXTPATH=<path_to_gettext>

SET OPENSSLPATH=<path_to_openssl>

SET PGPATH=<path_to_pg_installation_dir>

SET ARCH=x86

REM Set ARCH x64 for 64 bit

msbuild sslutils.proj /p:Configuration=Release
```

Where:

`path_to_gettext` specifies the location of the `GETTEXT` library and header files.

`path_to_openssl` specifies the location of the `openssl` library and header files.

`path_to_pg_installation_dir` specifies the location of the Postgres installation.

4. Copy the compiled `sslutils` files to the appropriate directory for your installation. The `sslutils` directory contains the following files:

```
sslutils--1.3.sql

sslutils--unpackaged--1.3.sql

sslutils--pemagent.sql.in

sslutils.dll
```

Copy the `.dll` libraries and `.sql` files into place:

```
COPY sslutils*.sql* "%PGPATH%\share\extension\"

COPY sslutils.dll "%PGPATH%\lib\"
```

Registering the service

When you install a PostgreSQL or EDB Postgres Advanced Server database using an installer from EnterpriseDB, the installer registers the service for you.

If you' re using Windows to host the PEM backend database, you must register the name of the Postgres server with the Windows service manager. If you' re using a Postgres server that was created using an EnterpriseDB installer, the service is registered automatically. If you are manually building the installation, you can use the `register` clause of the Postgres `pg_ctl` command to register the service:

```
pg_ctl register [-N <service_name>] [-U <user_name>]
 | [-P <password>] [-D <data_directory>]
```

Where:

- `service name` specifies the name of the Postgres cluster.
- `user_name` specifies the name of an operating system user with sufficient privileges to access the Postgres installation directory and start the Postgres service.
- `password` specifies the operating system password associated with the user.
- `data_directory` specifies the location of the Postgres data directory.

For more information about using the `pg_ctl` command and the available command options, see the Postgres core documentation .

Invoking the PEM server installer

After preparing the existing Postgres server, invoke the PEM server installer. Assume administrator privileges and navigate to the directory that contains the installer. Then, invoke the installer:

```
pem_server-7.<x>.<x>-<x>-<platform>
```

Where <x> is the major and minor versions of PEM, and <platform> is the platform.

Then, begin the installation:

1. The installer displays a Welcome screen. Select **Next**.

2. Review the license agreement before selecting the appropriate radio button and accepting the agreement. Select **Next**.
3. Use the Installation Directory dialog box to specify the location of the PEM server and access the Advanced Options dialog box:

- Use the **Installation Directory** field to open a browser dialog and select the directory to install the PEM server in.
- If you're installing the PEM server on an existing server, select **Show advanced options** to include the Advanced Options dialog box in the installation process.
- Select **Next**.

1. Use the Advanced Options dialog box to specify an installation type. Select:

- **Web Services and Database** if both the Postgres server and the PEM-HTTPD server will reside on the current host. This option is valid if you're using an existing Postgres server to host the PEM server or using the PEM server installer to install the Postgres server where the PEM server will reside.

If you select **Web Services and Database**, the PEM server installer checks the current host for a PEM-HTTPD installation and upgrades or installs PEM-HTTPD if necessary.

- **Web Services** if only the PEM-HTTPD server will reside on the current host. See Installing web services for more information about invoking this option.
- **Database** if you're installing only the PEM server (and creating the `pem` backend database) on the current host. This option is valid if you're using an existing Postgres server to host the PEM server or using the PEM server installer to install the PostgreSQL server where PEM will reside.

After selecting an installation option, select **Next**.

1. Use the list on the Database Server Selection dialog box to select a backend database for the PEM server:

- Select the name of a Postgres server on the current host that was installed using a Postgres one-click installer or EDB Postgres Advanced Server installer.
- Select **PostgreSQL x (Packaged)** to install and use the PostgreSQL server that's packaged with the PEM server installer, where `x` is the version of the PostgreSQL database server.
- Select **Other Database Server** to to use a Postgres database that was installed from a source other than an EnterpriseDB installer (such as from an rpm or built from source).

Note: The selected database server must include an installation of the *sslutils* contrib module and a registered service (on Windows).

For more information, see *Preparing the Postgres server* .

If you selected **Web Services and Database** on the Advanced Options dialog box, the installation wizard checks the current host for an existing PEM-HTTPD installation and upgrades or installs the service as needed.

If you selected **Database** on the Advanced Options dialog box, the Database Server Installation Details dialog box opens.

1. Use the fields on the Database Server Installation Details dialog box to describe the connection to the Postgres server that will host the PEM server:

- Enter the name of a database superuser in the **User** field.
- Enter the password associated with the superuser in the **Password** field.

Select **Next**.

1. Provide the administrator password for the PEM agent service to run under. Select **Next**.
2. Use the Network Details dialog box to specify the CIDR-style network address for the PEM agents to connect to the server (the `client-side` address). The specified address is added to the server's `pg_hba.conf file.`

You can specify more network addresses by manually adding entries to the `pg_hba.conf` file on the PostgreSQL server. Use the first entry as a template.

After you add the network address, select **Next**.

The PEM server installer installs a PEM agent to the host where the server resides to monitor the server and provide alert processing and garbage collection services. A certificate is also installed in the location specified in the **Agent certificate path** field.

1. You can enter an alternate description or an alternate agent certificate path for the PEM agent or accept the defaults. Select **Next**.
2. The wizard is now ready to install the PEM server. Select **Next** to continue with the installation.

During the installation process, the installer copies files to the system and sets up the PEM server's backend database. A confirmation indicates that the `pem` database was created and configured.

1. Select **OK**.

If you're using a PEM-HTTPD service that resides on a separate host, you must:

- Modify the `pg_hba.conf` file on the Postgres server host to allow connections between the hosts.
- Invoke the PEM server installer on the host of the PEM-HTTPD server. See Installing Web Services for more information about installing PEM-HTTPD.

Invoking the server installer from the command line

The command line options of the PEM server and PEM agent installers offer functionality in situations where a graphical installation might not work because of limited resources or system configuration. You can:

- Include the `--mode unattended` option when invoking the installer to perform an installation without user input.

Not all command line options are suitable for all platforms. For a complete reference guide to the command line options, use the `--help` option when you invoke the installer.

Invoking the PEM server installer in unattended mode

You can perform an unattended PEM server installation by providing installation preferences on the command line when invoking the installer. The system on which you're installing the PEM server must have internet access.

You must have administrator privileges to install the PEM server. Before invoking the PEM server installer, you must install the following dependencies:

- PostgreSQL
- pem-httpd
- Language Pack

You can use the PEM server installer to satisfy the dependencies of the PEM server. Navigate to the location of the installer, and use the following command to extract the dependencies:

```
pem-server-7.<x>.<x>-windows-x64.exe --extract-dependents C:\
```

In this example, the files are extracted to the `C:\` directory. After extracting the files, you must install each program. Navigate to the directory that contains the files (in this example, `C:\)` , and enter:

```
edb-languagepack-<version>-windows-x64.exe --mode unattended
pem-httpd-<version>-windows-x64.exe --mode unattended
postgresql-<version>-windows-x64.exe --mode unattended
```

Then, you can invoke the PEM server installer:

```
pem-server-7.<x>.<x>-windows-x64.exe --mode unattended
- -existing-user <registered_edb_user> --existing-password
<edb_user_password> --pgport <port> --pguser postgres
- -agent_description pem-agent --systempassword <windows_password>
- -agent-crt-path C:\edb``
```

Where:

- `registered_edb_user` specifies the name of a registered EnterpriseDB user. To register, visit the EDB website .
- `edb_user_password` specifies the password associated with the EDB user account.
- `port` specifies the port used by the backing PostgreSQL database. By default, the PostgreSQL database uses port 5432.
- `cidr_address_range` specifies the address range to add to the `pg_hba.conf` file of the PEM server' s backing database to allow connections from the agents for the server to monitor. You can specify a network range (for example, 192.168.2.0/24) to provide server access to agents that reside on the same network.
- `windows_password` specifies the password associated with the Windows administrator' s account.

Note: When invoked in unattended mode, the PostgreSQL installer creates a user named *postgres* with a password of *postgres* .

Creating an EDB repository on an isolated network

You can create a local repository to act as a host for the PEM RPM packages if the server where you want to upgrade PEM can' t directly access the EDB repository. This is a high-level overview of the steps required. You might need to modify the process for your own network.

To create and use a local repository:

1. Use the following commands on a system with Internet access to download the dependencies for PEM:

```
yum install yum-plugin-downloadonly
mkdir /<pem_dir>
yum install --downloadonly --downloaddir=/<pem_dir>/ edb-pem
mkdir /<epel_dir>
yum install --downloadonly --downloaddir=/<epel_dir>/ epel-release*
```

Where `<pem_dir>` and `<epel_dir>` are the local directories that you create for downloading the RPMs.

2. Copy the directories `/<pem_dir>` and `/<epel_dir>` to the machine in the isolated network.
3. Create the repositories:

```
createrepo /<pem_dir>
createrepo /<epel_dir>
```

4. Create a repository configuration file called `/etc/yum.repos.d/pem.repo` with connection information that specifies:

```
[pemrepo]
name=PEM Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

5. Create a repository configuration file called `/etc/yum.repos.d/epel.repo` with connection information that specifies:

```
[epelrepo]
name=epel Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

6. After specifying the location and connection information for your local repository, you can use dnf commands to install or upgrade PEM server:

To install PEM server:

```
dnf install edb-pem
```

To upgrade PEM server:

```
dnf upgrade edb-pem
```

Configuring the PEM server on Linux

The PEM server package includes a script (`configure-pem-server.sh`) to help automate the configuration process for Linux platform installations. The script is installed in the `/usr/edb/pem/bin` directory. To invoke the script, use the command:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

Note: If you're using the SSL certificates, then make sure that all the SSL certificates are in the data directory in the backend database server. If the certificates aren't in the data directory, then the PEM server's configure script might fail because it looks into the data directory while configuring the PEM server.

When invoking the script, you can include command line options to specify configuration properties. The script prompts you for values that you omit on the command line. The accepted options are:

Option	Description
-acp or *--pemagent-certificate-path*	Defines PEM agent certificate path. The default is */root/.pem*.
-ci or *--cidr-address*	CIDR-formatted network address range that agents connect to the server from, to be added to the server's *pg_hba.conf* file, for example, *192.168.1.0/24*. The default is *0.0.0.0/0*.
-dbi or *--db-install-path*	Directory for the database server installation, for example, */usr/edb/as12* for EDB Postgres Advanced Server or */usr/pgsql-12* for PostgreSQL.
-ds or *--db-unitfile*	Unit file name of the PEM database server. For EDB Postgres Advanced Server, the default file name is *edb-as-12*. For PostgreSQL, it's *postgresql-12*.
-ho or *--host*	Host address of the PEM database server.
-p or *--port*	Port number of the PEM database server.
-ps or *--pemagent-servicename*	Service name of the pemagent. The default value is *pemagent*.
-sp or *--superpassword*	Superuser password of the PEM database server. This value is required.
-su or *--superuser*	Superuser name of the PEM database server.
-au or *--use-agent-user*	PEM agent user name.
-t or *--type*	Installation type: Specify *1* if the configuration is for web services and backend database, *2* if you're configuring web services, or *3* if you're configuring the backend database. If you specify *3*, the database must reside on the local host.
-un or *--uninstall-pem-server*	Uninstalls the PEM server.
-nhc or *--no-hba-change*	Skips the changes done to *pg_hba.conf* and *pg_config* files.
-uac or *--use-agent-sslcert*	Reuses the existing agent SSL certificate while configuring the PEM server.
-uak or *--use-agent-sslkey*	Reuses the existing agent SSL key while configuring the PEM server.
-scs or *--server-certificate-subject*	Provides the custom web server certificate subject. The format is */C=CountryCode/ST=StateName/L=CityName/O=CompanySelectionName/CN=CommonNameorHostna* Provide the *C=CountryCode* as *Alpha-2* code.
-h or *--help*	Lists all the available options while configuring the PEM server.

If you don't provide configuration properties on the command line, the script prompts you for values. When you invoke the script, choose from:

1. `Web Services and Database` — Select this option if the web server and database both reside on the same host as the PEM server.
2. `Web Services` — Select this option if the web server resides on a different host from the PEM server.
3. `Database` — Select this option to configure the PEM backend database for use by the PEM server. The specified database must reside on the local host.

Note: If the web server and the backend database (PEM server) reside on separate hosts, configure the database server first (option 3) and then web services (option 2). The script proceeds only if the backend database is configured before web services.

After selecting a configuration option, the script prompts you for configuration properties. When the script finishes, it creates the objects required by the PEM server or performs the configuration steps required. To view help for the script, use the command:

```
/usr/edb/pem/bin/configure-pem-server.sh --help
```

Post-configuration steps when web server and PEM backend database are installed separately

If you choose to run the web application server on a separate host to the backend database, you need to perform some additional manual steps before PEM is fully operational.

Make sure that the backend Postgres database accepts the connections from any user permitted to log in to PEM from the web application server. To achieve this, add this entry to `pg_hba.conf` :

```
host pem +pem_user <web_app_ip>/32 md5
```

Where `<web_app_ip>` is the IP address of the web application server.

Additionally, if the IP address of the web application server isn't within the network address range specified when the script is executed, you must add two entries to allow the PEM agent on this server to connect:

```
host pem +pem_agent <web_app_ip>/32 md5
host pem +pem_agent <web_app_ip>/32 cert
```

Where `<web_app_ip>` is the IP address of the web application server.

Accessing the PEM application

After configuring the PEM server, you can access the PEM web interface in your browser. Navigate to:

```
https://<ip_address_of_PEM_server>:8443/pem
```

By default, the web services listen on port 8443. To change the port, see *Changing the default port* .

3.2 Installing Postgres Enterprise Manager agent

Select a link to access the applicable installation instructions:

3.2.1 Linux x86-64 (amd64)

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64* , *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64* , *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky* , *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky* , *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*

Debian and derivatives

- *Installing Postgres Enterprise Manager agent on Ubuntu 22.04 x86_64* , *Installing Postgres Enterprise Manager agent on Ubuntu 20.04 x86_64*
- *Installing Postgres Enterprise Manager agent on Debian 12 arm64* , *Installing Postgres Enterprise Manager agent on Debian 11 x86_64*

3.2.2 Linux Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le)

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64* , *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*

3.2.3 Linux Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64)

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*

Debian and derivatives

- *Installing Postgres Enterprise Manager agent on Debian 12 arm64*

3.2.4 Windows

- *Installing Postgres Enterprise Manager agent on Windows*

Installing Postgres Enterprise Manager agent on Linux x86 (amd64)

Operating system-specific install instructions are described in the corresponding documentation:

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*

Debian and derivatives

- *Installing Postgres Enterprise Manager agent on Ubuntu 22.04 x86_64*
- *Installing Postgres Enterprise Manager agent on Ubuntu 20.04 x86_64*
- *Installing Postgres Enterprise Manager agent on Debian 12 arm64*
- *Installing Postgres Enterprise Manager agent on Debian 11 x86_64*

Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 x86_64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
dnf repolist | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.

3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on RHEL 8 or OL 8 x86_64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

`dnf repolist | grep enterprisedb`

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
dnf repolist | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
dnf repolist | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on SLES 15 x86_64

Note: Postgres Enterprise Manager 8.3 and later is supported on SLES.

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
zypper lr -E | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

- Activate the required SUSE module:

```
# You can use SLES 15 SP3 for PEM 8.3 and later:
sudo SUSEConnect -p PackageHub/15.3/x86_64

# You can use SLES 15 SP4 for PEM 8.6 and later:
sudo SUSEConnect -p PackageHub/15.4/x86_64
```

- Refresh the metadata:

```
sudo zypper refresh
```

Install the package

```
sudo zypper -n install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Ubuntu 22.04 x86_64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

`apt-cache search enterprisedb`

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo apt-get -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Ubuntu 20.04 x86_64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

`apt-cache search enterprisedb`

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo apt-get -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Debian 12 x86_64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
apt-cache search enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo apt-get -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Debian 11 x86_64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

`apt-cache search enterprisedb`

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo apt-get -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Linux IBM Power (ppc64le)

Operating system-specific install instructions are described in the corresponding documentation:

Red Hat Enterprise Linux (RHEL)

- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*

SUSE Linux Enterprise (SLES)

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*

Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
dnf repolist | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
dnf repolist | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on SLES 15 ppc64le

Note: Postgres Enterprise Manager 8.3 and later is supported on SLES.

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
zypper lr -E | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

- Activate the required SUSE module:

```
# You can use SLES 15 SP3 for PEM 8.3 and later:
sudo SUSEConnect -p PackageHub/15.3/ppc64le

# You can use SLES 15 SP4 for PEM 8.6 and later:
sudo SUSEConnect -p PackageHub/15.4/ppc64le
```

- Refresh the metadata:

```
sudo zypper refresh
```

Install the package

```
sudo zypper -n install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Linux AArch64 (ARM64)

Operating system-specific install instructions are described in the corresponding documentation:

Red Hat Enterprise Linux (RHEL) and derivatives

- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*
- *Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64*

Debian and derivatives

- *Installing Postgres Enterprise Manager agent on Debian 12 arm64*

Installing Postgres Enterprise Manager agent on RHEL 9 or OL 9 arm64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

```
dnf repolist | grep enterprisedb
```

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo dnf -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Debian 12 arm64

Prerequisites

Before you begin the installation process:

- Set up the EDB repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To determine if your repository exists, enter this command:

`apt-cache search enterprisedb`

If no output is generated, the repository isn't installed.

To set up the EDB repository:

1. Go to EDB repositories .
2. Select the button that provides access to the EDB repository.
3. Select the platform and software that you want to download.
4. Follow the instructions for setting up the EDB repository.

Install the package

```
sudo apt-get -y install edb-pem-agent
```

After installing PEM agent, you need to register the PEM agent. For detailed information see *Registering a PEM agent* .

Installing Postgres Enterprise Manager agent on Windows

The PEM agent graphical installer for Windows installs and registers the PEM agent.

To invoke the PEM agent installer, assume administrator privileges and navigate to the directory that contains the installer. Then, invoke the installer:

```
pem_agent-9.<x>.<x>-<x>-platform.exe
```

1. The Setup screen opens, welcoming you to the PEM agent installer. Select **Next**.
2. Review the license agreement before selecting the appropriate radio button and accepting the agreement. Select **Next**.

3. By default, the PEM agent is installed in the `C:\Program Files (x86)\edb\pem` directory. You can accept the default installation directory. Or you can modify the contents of the **Installation Directory** field, specifying an alternate installation directory for the PEM agent.

- By default, the PEM agent installer places a certificate in the Administrator's `%APPDATA%\pem` directory. Select **Show advanced options** to include a dialog box in the installation process that allows you to specify an alternate path for the certificate file.
- Select **Register now?** to register the newly installed PEM agent with the PEM server.
- Select **Next**.

1. Enter the connection details for the PEM server in the PEM Server Installation Details dialog box.

- Specify the name or IP address of the system where the PEM database server resides in the **Host** field. If the PEM-HTTPD web server and PEM database are hosted on different systems, you must specify the host of the PEM database.
- Specify the name of the database superuser in the **User Name** field.
- Specify the password associated with the database superuser in the **Password** field.
- Specify the port that PostgreSQL is monitoring in the **Port** field.

1. Select **Next** to continue to pemAgent Service Account. The installer attempts to connect to the server to verify that the details are correct.

Note: The PEM server must allow connections from the PEM agent installer. If you encounter a connection error, confirm the connection properties specified in the PEM Server Installation Details dialog box are correct. Confirm that the *pg_hba.conf* file on the PEM server allows a connection to the server described in the error message.

---

1. Provide the password for the edb account for the pemAgent service to run under. The agent certificate and key files are created in the `C:\Users\edb\AppData\Roaming\pem` directory. Select **Next**.
2. The tree displayed in the Browser panel of the PEM web interface displays the value entered in the **Description** field to identify the PEM agent. Specify a descriptive name for the agent, such as the hostname of the machine the agent is installed on or a name that reflects the host's functionality.

Provide a descriptive name, or accept the default provided by the PEM Agent host. Select **Next**.

1. If you selected **Show advanced options**, the Advanced Options dialog box opens. By default, the PEM agent installer places the certificate in the `C:\Program Files (x86)\edb\pem` directory. Specify an alternate path for the certificate, or accept the default and select **Next**.
2. The wizard is ready to install the PEM agent. Select **Next**.
3. On the Ready to Install dialog box, select **Next**. The installer copies files to the system and registers the agent on the PEM server.

When the installation completes, the PEM agent is running and reporting operating system and host data to the PEM server. To start monitoring Postgres instances on the host of the PEM agent, add them to PEM's enterprise directory and bound them to the agent.

Invoking the agent installer from the command ine

The command line options of the PEM Agent installers offer functionality in situations where a graphical installation might not work because of limited resources or system configuration. You can:

- Include the `--mode unattended` option when invoking the installer to perform an installation without user input.

Not all command line options are suitable for all platforms. For a complete reference guide to the command line options, include the `--help` option when you invoke the installer.

Invoking the PEM agent installer in unattended mode

You can perform an unattended PEM server installation by providing installation preferences on the command line when invoking the installer. The system where you' re installing the PEM server must have internet access.

Before invoking the PEM agent installer in unattended mode, you must:

- Install the PEM server. The `pg_hba.conf` file of the PEM server must allow connections from the host of the PEM agent.
- Ensure that the monitored Postgres database has SSL enabled and is accepting connections.

You need administrator privileges to install the PEM Agent. Use the following command to invoke the PEM agent installer in unattended mode:

```
pem-agent-9.<x>.<x>-windows-x64.exe --mode unattended
- -pghost <pem_server_host_address> --pgport <pem_server_port>
- -pguser postgres --pgpassword <pguser_password>
- -agent_description <agent_name>
```

Where:

- `pem_server_host_address` specifies the IP address of the host of the PEM server.
- `pem_server_port` specifies the port used by the backing PEM database. By default, the database uses port 5432.
- `pguser_password` specifies the password associated with the PEM database superuser.
- `agent_name` specifies a descriptive name for the PEM agent.

Note: When configuring a shell/batch script run by a Windows agent, set the *AllowBatchJobSteps* parameter to *True* in the *agent.cfg* file. The PEM agent doesn't execute any batch/shell script by default.

3.3 Upgrade and migration

You can upgrade the Postgres Enterprise Manager (PEM) components including the PEM installation, the backend database, and the SQL Profiler. You can also move the PEM server.

Note: If you're upgrading PEM 7.x to the latest version and the PEM backend database is PostgreSQL or EDB Postgres Advanced Server version 10 or earlier: 1. Upgrade the PEM backend database. 2. Upgrade the PEM installation.

If the PEM backend database is PostgreSQL or EDB Postgres Advanced Sever version 11 or later, then upgrading the PEM backend database is optional.

3.3.1 Upgrading a PEM installation

The process of upgrading a PEM installation is platform specific. You can update a PEM agent or server on a Linux platform by using the native packages and on a Windows platform by using the PEM graphical installer available for Windows.

Note:

- Prior to PEM 7.8 release, you installed PEM agent or server on Linux either by using the graphical installer or by using the RPMs. - For PEM version 7.8 and later, PEM graphical installers for Linux are discontinued. To upgrade a PEM agent or server on a Linux host from any earlier version to PEM version 7.9 or later, you must use native packages.

Links to PEM installers and RPMs are available at the EDB website .

Upgrading a PEM installation on a Linux host

To upgrade PEM component software on Linux hosts, install a newer version of the PEM component packages in the following order:

1. Invoke the PEM agent package installer on each monitored node except the PEM server host.
2. Invoke the PEM server package installer. It upgrades both the PEM server and the PEM agent that resides on the PEM server host.

During an installation, the component installation detects an existing installation and performs an upgrade. After upgrading the PEM agent and server, you can upgrade SQL Profiler if required. That step is platform specific.

Note: If you already configured or are planning to configure any shell/batch script run by a Linux agent that's upgraded from any earlier version to version 7.11 or later, you must specify the user for the *batch_script_user* parameter in the *agent.cfg* file. We strongly recommended that you use a non-root user to run the scripts. Using the root user can result in compromising the data security and operating system security. However, if you want to restore the pemagent to its original settings using a root user to run the scripts, then you must set the *batch_script_user* parameter to *root* .

Upgrading a PEM server installation

If you want to upgrade a PEM server that' s installed on a machine in an isolated network, you need to create a PEM repository on that machine before you upgrade the PEM server. For more information, see *Creating an EDB repository on an isolated network* .

To upgrade a PEM server installation:

```
sudo <package-manager> upgrade edb-pem
```

Where `<package-manager>` is the package manager used with your operating system:

Package manager	Operating system
dnf	RHEL 8/9 and derivatives
zypper	SLES
apt-get	Debian and Ubuntu

After upgrading the PEM server, you must configure the PEM server. For detailed information, see *Configuring the PEM server on Linux* .

Note: If you upgrade the PEM backend database server and the PEM server, update the *PG_INSTALL_PATH* and *DB_UNIT_FILE* parameters pointing to the new version in the */usr/edb/pem/share/.install-config* file before you run the configure script.

Configuring the PEM server

After upgrading the PEM server, you can use the following command to configure the PEM server:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

The configure script uses the values from the old PEM server configuration file while running the script. For details, see *Configuring the PEM server on Linux platforms* .

Note:

- The configure script requires a superuser password only after the upgrade process.

```
- If your configure script gets stuck, then stop the PEM agent with `alert_threads>0` .␣
↪To get the details of such agents, execute the query:
```

```
SELECT agent_id FROM pem.agent_config WHERE param=alert_threads AND value > 0;
```

Stop the running agents and rerun the configure script.

If the problem persists, then run the query to terminate the stuck alert processes:

```
SELECT pg_terminate_backend(pid) FROM pg_stat_activity WHERE query=SELECT pem.process_
↪one_alert();
```

Then rerun the configure script.

Upgrading a PEM agent installation

To upgrade a PEM agent:

```
sudo <package-manager> upgrade edb-pem-agent
```

Where `<package-manager>` is the package manager used with your operating system:

Package manager	Operating system
dnf	RHEL 8/9 and derivatives
zypper	SLES
apt-get	Debian and Ubuntu

Upgrading a PEM installation on a Windows host

To upgrade PEM component software on Windows hosts, invoke a newer version of the PEM component installers in the following order:

1. Invoke the PEM agent installer on each monitored node except the PEM server host.
2. Invoke the PEM server installer. This installer upgrades both the PEM server and the PEM agent that resides on the PEM server host.

During an installation, the component installer automatically detects an existing installation and performs an upgrade. After upgrading the PEM gent and server, you can upgrade SQL profiler if required. This step is platform specific.

Upgrading a PEM agent on a Windows host

To upgrade a system that is currently monitored by a PEM agent to a more recent PEM agent, download and invoke a newer version of the PEM Agent installer on the system that the agent is monitoring.

1. To invoke the installer, right-click the downloaded installer icon and select **Run as Administrator**. The PEM Agent Setup wizard opens, welcoming you.
2. Read and accept the license agreement and then select **Next**.
3. The setup wizard automatically detects an existing agent and upgrades the installed version. Select **Next**.
4. The pemAgent Service Account dialog box might prompt you for the password of the account under which the PEM agent service runs. If prompted, provide the password, and select **Next**.
5. When the Ready to Install dialog box informs you that the installation is about to begin, select **Next**. The wizard upgrades your PEM agent to the latest version.
6. The PEM Agent Setup wizard informs you when the installation completes. Select **Finish**.
7. After the installation completes, you' re prompted to restart the machine. Select **Yes** to restart the machine and the PEM agent.

Upgrading the PEM server on a Windows host

The PEM server installer enables you to upgrade between major versions of the PEM server. You can upgrade from version 5.0 to version 7.16 without first upgrading to version 6.0.

1. To invoke the installer, right-click the downloaded installer, and select **Run as Administrator**.
2. The PEM Server Setup wizard welcomes you. Select **Next**.
3. The PEM Server Setup wizard prompts you to accept the license agreement. After reviewing the license agreement, select **I accept the agreement** and select **Next**.
4. The wizard checks the PEM server host for an existing PEM server installation. If the wizard locates an installation, it performs an upgrade. Select **Next**.

Before upgrading the PEM server, the wizard confirms that the requirements of the new PEM server are present. If any supporting components are missing or are at a version that doesn't support the new PEM installation, the wizard informs you that it must upgrade the dependencies. It then invokes the required installers.

1. When the installation wizards completes the dependency upgrades, you're prompted to restart the machine. Select **No** to continue the upgrade process.
2. The wizard then opens the Database Server Installation Details dialog box, prompting you for connection credentials for the database superuser of the PEM backend database. Provide:

- The name of the database superuser in the **User** field.
- The password associated with the database superuser in the **Password** field.

Select **Next**.

1. The pemAgent Service Account dialog box might prompt you for the password of the account under which the PEM agent service runs. If prompted, provide the password, and select **Next**.
2. The Ready to Install dialog box informs you that the setup wizard is ready to perform the installation. Select **Next** to start the installation.

After upgrading the PEM server (and the agent that resides on the same host as the PEM server) and configuring the web service, the PEM setup wizard notifies you of the port on which the service is listening. Use this port number when connecting to the PEM server with the PEM client.

1. Select **OK**. The PEM server setup wizard informs you that the installation is complete.
2. If you're prompted to restart the machine, select **Yes** to restart the machine and the httpd service.

If you installed the PEM backend database server and PEM-HTTPD on different hosts, then you must run the PEM server installer twice: once on each host. Extract the language pack installer, and install it on the host of PEM-HTTPD before invoking the PEM installer. Include the following keywords when invoking the installer to extract the language pack:

```
--extract-languagepack <path>
```

Where `<path>` specifies an existing path for extracting the language pack installer.

Note: By default EDB Language Pack is installed in *C:edblanguagepackv1* .

If you're upgrading the PEM Server using StackBuilder Plus, then you might see an error. After displaying the error, PEM reports that installation is completed. However, the installation isn't complete. You need to do the installation by invoking the installer file from the location where it is downloaded.

After upgrading the PEM server, you might want to upgrade the backend database to a more recent version. For information, see *Upgrading the PEM backend Postgres database* .

Upgrading a PEM installation on high-availability mode using HTTPD

If you're using *Using Failover Manager for high availability* on your PEM server, you can perform a major or minor upgrade of the PEM installation by updating the PEM packages.

This example uses these IP addresses:

- Primary server: 172.17.0.2
- Secondary server (standby): 172.17.0.3

The example upgrade is performed on a RHEL-based operating system where HTTPD is used for the web server services.

Upgrading the packages

1. On the primary server (172.17.0.2), update the PEM packages:

```
dnf update edb-pem
```

Updating ensures that the latest version of the script required for the upgrade is available.

1. Execute the upgrade script. It upgrades PEM to the latest available version and restarts the backend database, the web server, and the PEM agent:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

1. On the secondary standby server (172.17.0.3), update the PEM packages:

```
dnf update edb-pem
```

1. Copy the configuration files, certificate, and key from the primary server to the secondary server:

```
/etc/httpd/conf.d/edb-pem.conf
/etc/httpd/conf.d/edb-ssl-pem.conf
/usr/edb/pem/share/.install-config
/usr/edb/pem/resources/server-pem.crt
/usr/edb/pem/resources/server-pem.key
/usr/edb/pem/web/pem.wsgi
/usr/edb/pem/web/config_setup.py
```

!!!note If you have more than one standby server, update the packages and copy the configuration files, certificate, and key from the primary server to each standby instance. .. ::
The upgrade is complete.

3.3.2 Upgrading the PEM backend Postgres database

If you’re updating PEM components and the PEM backend database, perform PEM component updates on the server and agent before updating the backend database. For more information about updating PEM component software, see *Upgrading a PEM installation on high-availability mode using HTTPD* .

Note:

- From PEM 8.0 onwards, PostgreSQL or EPAS versions 11 or later are only supported as backend database servers. As a result, if your backend database server is earlier than version 11, you need to first upgrade your backend database server and then upgrade the PEM components.

```
-   After upgrading the backend database server, if you encounter this error while␣
↪creating the server in the PEM web interface:
```

```
Error - User does not have enough permission to add new server.
        Please contact the administrator to grant pem_database_server_registration role
        to the enterprisedb user.
```

Resolve the error by updating the roles and granting appropriate permissions:

```
UPDATE pem.roles SET rolid = pr.oid FROM pg_roles pr WHERE pr.rolname = pem_ ||␣
↪component;
```

The update process uses the pg_upgrade utility to migrate from one version of the backend server to a more recent version. pg_upgrade enables migration between any supported version of Postgres and any subsequent release of Postgres that's supported on the same platform.

If the source PEM server is earlier than the 7.16 version, then you need to replace the following functions before you run pg_upgrade:

- The `abstime` , `reltime` , and `tinterval` datatypes are deprecated from Postgres version 12 or later, hence to replace those dataypes with `timestamptz` data type use this command:

```
DO
$$
DECLARE
    rec record;
    cnt integer;
BEGIN
    -- Check for the deprecated type in our user info probe
    SELECT count(*) INTO cnt
    FROM pem.probe_column
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
        IF cnt = 0 THEN
            RETURN;
        END IF;
    ALTER TABLE pemdata.user_info
        ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    ALTER TABLE pemhistory.user_info
            ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    -- Now update the pem.probe_column itself
    UPDATE pem.probe_column
    SET sql_data_type = 'timestamptz'
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
END;
$$ LANGUAGE 'plpgsql';
```

- Replace this function to avoid any alert errors:

```
  CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
  template_id pem.alert_template.id%type, params text[]
  )
  RETURNS bool AS $FUNC$
  DECLARE
      res bool := TRUE;
  BEGIN
      /*
- During restoring the pem database, it does not maintain the order while
- inserting data in the table, and uses the sort table based on the
- names.
- Hence - we need to check the foreign key constraint is present before
- validating these values.
      */
      IF EXISTS(
```

(continues on next page)

(continued from previous page)

```
        SELECT 1 FROM information_schema.table_constraints
        WHERE constraint_name=alert_template_id_fkey AND
        table_name=alert AND table_schema=pem
    ) THEN
   /*
- Need to use the IS TRUE construct outside the main query, because
- otherwise if theres no template by that ID then the query would return
- 0 rows and the result of the function would be undefined and CHECK
- constraint would succeed.
- Probably this is being over-cautious, because pem.alert.template_id
- references pem.alert_template.id. But the SQL standard (probably) does
- not define the order in which the CHECK or the FOREIGN KEY constraints
- should be validated; in case CHECK is validated first, we want it to
- fail.
    */
  EXECUTE $SQL$
      SELECT (
        SELECT pem.check_array_size_equal(t.param_names, $2)
        FROM pem.alert_template AS t
        WHERE id = $1
      ) IS TRUE
    $SQL$ INTO res USING template_id, params;
  END IF;
   RETURN res;
  END
  $FUNC$ LANGUAGE plpgsql;
```

pg_upgrade supports a transfer of data between servers of the same type. For example, you can use pg_upgrade to move data from a PostgreSQL 10 backend database to a PostgreSQL 11 backend database but not to an EDB Postgres Advanced Server 11 backend database. If you want to migrate to a different type of backend database (such as from a PostgreSQL server to EDB Postgres Advanced Server), see *Moving the PEM server* .

You can find more information about using pg_upgrade at pg_upgrade .

1. Download and invoke the updated installer. Installers for PostgreSQL and EDB Postgres Advanced Server are available through the EDB website .

After downloading the installer for the server version you are upgrading to, invoke the installer on the host of the PEM server. Follow the onscreen instructions of the installation wizard to configure and install the Postgres server.

You can optionally use a custom-built PostgreSQL server as a host of the PEM backend database. If you're upgrading from a PostgreSQL backend database listening on port 5432, the new server must be configured to listen on a different port.

2. Configure SSL utilities on the new server. The new backend database must be running the same version of sslutils that the current backend database is running. You can download the SSL Utils package from the EDB website .

You don't need to manually add the sslutils extension when using the EDB Postgres Advanced Server as the new backend database. The process of configuring sslutils is platform specific.

On Linux

- On an EDB Postgres Advanced Server backend database, the sslutils extension is installed by default.
- If you're using PostgreSQL as a PEM backend database, verify that you have access to the PostgreSQL community repository, and use the command:

```
yum install sslutils_<X>
```

Where <X> is the server version.

```
-   If you're using a EDB one-click installer of PostgreSQL as a PEM backend database,␣
↪use the command:
```

```
yum install gcc openssl-devel
```

Set the value of PATH so it can locate the pg_config program

```
export PATH=$PATH:/opt/postgres_inst_dir/<X>/bin/
```

Move into the `sslutils` folder, and enter:

```
make USE_PGXS=1
make USE_PGXS=1 install
```

Use psql to create the sslutils extension

```
CREATE EXTENSION sslutils
```

Debian 10 and Ubuntu 20 have increased the requirements to accept the certificates for security reasons. If you want to install the PEM agent on any of the machines, you must upgrade ssltuils to 1.3 where 4096-bit RSA key and sha256 signature algorithm support was added. If you don' t upgrade sslutils to 1.3, then PEM agent might fail to connect to the PEM backend database server, and it might log the error "ca md too weak."

On Windows

```
-   You must compile sslutils on the new backend database with the same compiler that␣
↪was used to compile sslutils on the original backend database. If you're moving to a␣
↪Postgres database that was installed using a PostgreSQL one-click installer (from EDB)␣
↪or an EDB Postgres Advanced Server installer, use Visual Studio to build sslutils. If␣
↪you're upgrading to PostgreSQL 10 or later, use Visual Studio 2010.
```

For detailed information about building a specific version of Postgres on Windows, consult the core documentation for that version. Core documentation is available at the PostgreSQL project website .

```
-   While specific details of the process vary by platform and compiler, the basic steps␣
↪on each platform are the same. The example that follows shows compiling OpenSSL␣
↪support for PostgreSQL on a 32-bit Windows system.
```

Before compiling the OpenSSL extension, you must locate and install OpenSSL for your version of Windows. Before invoking the OpenSSL installer you might need to download and install a prerequisite redistributable (such as `vcredist_x86.exe`).

After installing OpenSSL, download and unpack the sslutils utility package .

```
-   Copy the unpacked `sslutils`  folder to the Postgres installation directory (`C:\
↪ProgramFiles\PostgreSQL\<x.x>` ).

-   Open the Visual Studio command line, and navigate into the `sslutils`  directory.␣
↪Use the following commands to build sslutils:
```

```
SET USE_PGXS=1

SET GETTEXTPATH=\ <path_to_gettext>

SET OPENSSLPATH=\ <path_to_openssl>

SET PGPATH=\ <path_to_pg_installation_dir>

SET ARCH=x86

msbuild sslutils.proj /p:Configuration=Release
```

Where:

```
-   `path_to_gettext`  specifies the location of the `GETTEXT`  library and header files.

-   `path_to_openssl`  specifies the location of the openssl library and header files.

-   `path_to_pg_installation_dir`  specifies the location of the Postgres installation.
```

For example, the following set of commands builds OpenSSL support into the PostgreSQL 11 server:

```
SET USE_PGXS=1

SET OPENSSLPATH=C:\OpenSSL-Win32

SET GETTEXTPATH="C:\Program Files\PostgreSQL1"

SET PGPATH="C:\Program Files\PostgreSQL1"

SET ARCH=x86

msbuild sslutils.proj /p:Configuration=Release
```

```
-   When the build completes, the `sslutils`  directory contains the following files:
```

```
sslutils--1.3.sql
sslutils--unpackaged--1.3.sql
sslutils--pemagent.sql.in
sslutils.dll
```

```
-   Copy the compiled sslutils files to the appropriate directory for your installation;␣
↪for example:
```

```
COPY sslutils*.sql "%PGPATH%\share\extension\"

COPY sslutils.dll "%PGPATH%\lib\"

COPY sslutils.control "%PGPATH%\share\extension\"
```

3. Stop the services of both the old backend database and the new backend database.

On RHEL 8.x, open a command line and assume the identity of a superuser. Enter the command:

```
systemctl <service_name> stop
```

Where `<service_name>` specifies the name of the Postgres service.

On Windows, you can use the Services dialog box to control the service. To stop the service: 1. On the Control Panel select **System and Security > Administrative Tools**. 1. Double-click **Services**. 1. In the Services dialog box, select the service name and selet **Stop**.

4. Use the pg_upgrade utility to perform an in-place transfer of existing data between the old backend database and the new backend database. If your server is configured to enforce md5 authentication, you might need to add an entry to the `.pgpass` file that specifies the connection properties (and password) for the database superuser. Or you might need to modify the `pg_hba.conf` file to allow trust connections before invoking `pg_upgrade` . For more information about creating an entry in the `.pgpass` file, see the PostgreSQL core documentation .

During the upgrade process, pg_upgrade writes a series of log files. The cluster owner must invoke pg_upgrade from a directory in which they have write privileges. If the upgrade completes successfully, pg_upgrade removes the log files when the upgrade completes. If you don' t want pg_upgrade to delete the upgrade log files, include the `--retain` keyword when invoking pg_upgrade.

To invoke pg_upgrade, assume the identity of the cluster owner, navigate to a directory in which the cluster owner has write privileges, and execute the command:

```
<path_to_pg_upgrade> pg_upgrade

-d <old_data_dir_path>

-D <new_data_dir_path>

-b <old_bin_dir_path> -B <new_bin_dir_path>

-p <old_port> -P <new_port>

-u <user_name>
```

Where:

```
-   `path_to_pg_upgrade`  specifies the location of the pg_upgrade utility. By default,
 pg_upgrade is installed in the `bin`  directory under your Postgres directory. -   `old_
↪data_dir_path`  specifies the complete path to the data directory of the old backend␣
↪database. -   `new_data_dir_path`  specifies the complete path to the data directory␣
↪of the new backend database. -   `old_bin_dir_path`  specifies the complete path to␣
↪the bin directory of the old backend database. -   `new_bin_dir_path`  specifies the␣
↪complete path to the bin directory of the old backend database. -   `old_port` ␣
↪specifies the port on which the old server is listening. -   `new_port`  specifies the␣
↪port on which the new server is listening. -   `user_name`  specifies the name of the␣
↪cluster owner.
```

For example, the following command instructs pg_upgrade to migrate the PEM database from PostgreSQL 9.6 to PostgreSQL 11 on a Windows system (if the backend databases are installed in their default locations):

```
C:\>"C:\Program Files\PostgreSQL1\bin\pg_upgrade.exe"

-d "C:\Program Files\PostgreSQL0\data"
```

(continues on next page)

(continued from previous page)

```
-D "C:\Program Files\PostgreSQL1\data"

-b "C:\Program Files\PostgreSQL0\bin"

-B "C:\Program Files\PostgreSQL1\bin"

-p 5432 -P 5433

-U postgres
```

Once invoked, pg_upgrade performs consistency checks before moving the data to the new backend database. When the upgrade is finished, pg_upgrade notifies you that the upgrade is complete.

For detailed information about using pg_upgrade options or troubleshooting the upgrade process, see pg_upgrade .

5. Copy the following certificate files from the `data` directory of the old backend database to the `data` directory of the new backend database:

`ca_certificate.crt`

`ca_key.key`

`root.crt`

`root.crl`

`server.key`

`server.crt`

Once in place on the target server, make sure the files have these platform-specific permissions:

Permissions and ownership on Linux

File name	Owner	Permissions
ca_certificate.crt	postgres	-rw-------
ca_key.key	postgres	-rw-------
root.crt	postgres	-rw-------
root.crl	postgres	-rw-------
server.key	postgres	-rw-------
server.crt	postgres	-rw-r--r--

On Linux, the certificate files must be owned by postgres. You can use the following command at the command line to modify the ownership of the files:

```
chown postgres <file_name>
```

Where `file_name` specifies the name of the certificate file.

Only the owner of the `server.crt` file can modify it, but any user can read it. You can use the following command to set the file permissions for the `server.crt` file:

```
chmod 644 server.crt
```

Only the owner of the other certificate files can modify or read the file. You can use the following command to set the file permissions:

```
chmod 600 <file_name>
```

Where `file_name` specifies the name of the file.

Permissions and ownership on Windows

On Windows, the service account that performed the PEM server and backend database installation on the target host must own the certificate files moved from the source host. If you invoked the PEM server and Postgres installer using **Run as Administrator** from the context menu of the installer, the owner of the certificate files is Administrators.

To review and modify file permissions on Windows, right-click the file name and select **Properties**.

On the **Security** tab select a group or user name to view the assigned permissions. Select **Edit** or **Advanced** to open dialog boxes that allow you to modify the permissions associated with the selected user.

6. The `postgresql.conf` file contains parameter settings that specify server behavior. Modify the `postgresql.conf` file on the new server to match the configuration specified in the `postgresql.conf` file of the old server.

By default, the `postgresql.conf` file is located:

- For Postgres version earlier than 10 on Linux, in `/opt/PostgreSQL/<X>/data`
- For Postgres version 10 or later when installed with graphical installers on Linux, in `/opt/PostgreSQL/<X>/data`
- For Postgres version 10 or later when installed with an RPM on Linux, in `/usr/pgsql/<X>/data`
- For any Postgres version on Windows, in `C:\Program Files\PostgreSQL\<X>\data`

Where `<X>` is the version of Postgres on your system.

Use your choice of editor to update the `postgresql.conf` file of the new server. Modify the following parameters:

- The `port` parameter to listen on the port monitored by your original backend database (typically set to `5432`).
- The `ssl` parameter to be set to `on`

You must also ensure that the following parameters are enabled. If the parameters are commented out, remove the pound sign from in front of each `postgresql.conf` file entry:

- `ssl_cert_file = 'server.crt' # (change requires restart)`
- `ssl_key_file = 'server.key' # (change requires restart)`
- `ssl_ca_file = 'root.crt' # (change requires restart)`
- `ssl_crl_file = 'root.crl'`

Your installation might have other parameter settings that require modification to ensure that the new backend database behaves like the old backend database. Review the `postgresql.conf` files carefully to ensure that the configuration of the new server matches the configuration of the old server.

7. The `pg_hba.conf` file contains parameter settings that specify how the server enforces host-based authentication. When you install the PEM server, the installer modifies the `pg_hba.conf` file, adding entries to the top of the file:

3.3.3 Adding entries for PEM Agens and admins to connect to PEM server

```
hostssl pem +pem_user 192.168.2.0/24 md5
hostssl pem +pem_agent 192.168.2.0/24 cert
```

3.3.4 Adding entries (localhost) for PEM Agens and admins to connect to PEM server

```
hostssl pem +pem_user 127.0.0.1/32 md5
hostssl postgres +pem_user 127.0.0.1/32 md5
hostssl pem +pem_user 127.0.0.1/32 md5
hostssl pem +pem_agent 127.0.0.1/32 cert
```

By default, the `pg_hba.conf` file is located at the following location:

- For Postgres version earlier than 10 on Linux, in `/opt/PostgreSQL/<X>/data`
- For Postgres version 10 or later when installed with graphical installers on Linux, in `/Opt/PostgreSQL/<X>/data`
- For Postgres version 10 or later when installed with RPMs on Linux, in `/var/lib/pgsql/<X>/data`
- For Advanced Server version 10 or later when installed with RPMs on Linux, in `/var/lib/edb/as<X>/data`
- For any Postgres version on Windows, in `C:\Program Files\PostgreSQL\<X>\data`

Where `<X>` is the version of Postgres on your system.

Using your editor of choice, copy the entries from the `pg_hba.conf` file of the old server to the `pg_hba.conf` file for the new server.

8. Restart the service of the new backend database.

On RHEL 8.x, at the command line as superuser enter:

```
systemctl stop <service_name>
```

Where `service_name` is the name of the backend database server.

If you're using Windows, you can use the **Services** dialog box to control the service: 1. In the Control Panel, select **System and Security > Administrative Tools**. 1. Double-click the **Services** icon. 1. In the Services dialog box, select the service name and start the service.

3.3.5 Upgrading SQL Profiler

If you are upgrading from v4.0 (or an earlier version) of SQL Profiler, where the functions where created by loading the SQL file included in a plugin, first migrate to the extension and later update to 4.1:

```
#  running as superuser

CREATE EXTENSION sql_profiler VERSION 4.0;

ALTER EXTENSION sql_profiler UPDATE TO 4.1;
```

3.3.6 Upgrading Apache HTTPD server

The Apache HTTPD web server is used to serve the PEM web application. On Linux, PEM doesn't bundle HTTPD but depends on the version provided by the system package manager. On Windows, the PEM installer bundles a version of HTTPD.

Upgrading HTTPD on Linux

Important: If you're using Red Hat Enterprise Linux (RHEL), Rocky Linux, AlmaLinux, or Oracle Linux, and you're updating HTTPD to address a vulnerability, read *Note on HTTPD versioning in RHEL and RHEL derivatives* .

If you're running Linux, manage HTTPD along with other system software on your servers. We recommend always using the latest version available from your package manager.

You don't need to stop PEM or back up any files prior to upgrade. Initiate the upgrade from your package manager.

RHEL / Rocky Linux / AlmaLinux / Oracle Linux

```
sudo yum update httpd
```

Debian / Ubuntu

```
sudo apt update
sudo apt upgrade apache2
```

SLES

```
sudo zypper update apache2
```

HTTPD versioning in RHEL and RHEL derivatives

You might notice that the latest version of HTTPD available from RHEL or other RHEL derivatives has a significantly older version number than the latest community release. Specifically, RHEL 7 provides HTTPD 2.4.6, and RHEL 8 provides 2.4.37. It's very important to understand that Red Hat builds custom HTTPD packages for RHEL and backports security fixes.

For this reason, don't assume that a vulnerability present in the community 2.4.37 release is present in the RHEL package of the same version. If you're trying to install a newer version of HTTPD for this reason, it's almost certainly not necessary. See the Apache HTTPD versions supported by Red Hat article in the Red Hat knowledgebase for more information.

Warning: Red Hat doesn't support community HTTPD or HTTPD built from source. If you're considering using any other source of HTTPD on a RHEL system, read the Apache HTTPD versions supported by Red Hat article and make sure you understand the implications.

Upgrading HTTPD on Windows

To upgrade to the latest version of HTTPD packaged for PEM by EDB, download the latest PostgreSQL or EDB Advanced Server installer for Windows and use Stack Builder

or StackBuilder Plus respectively to update PEM-HTTPD. PEM-HTTPD is located under the Web Development category.

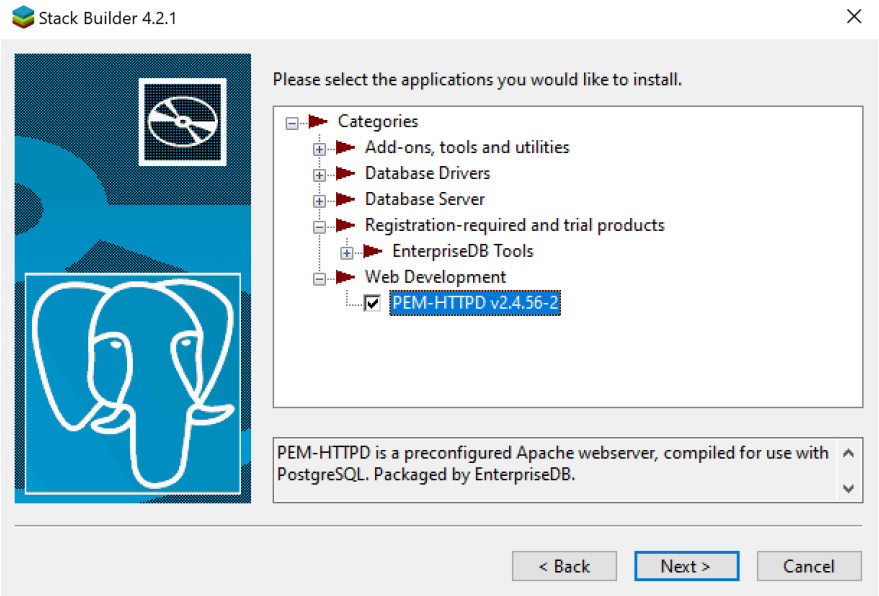


Fig. 1: PEM-HTTPD in Stack Builder

3.3.7 Moving the PEM server

You can move a PEM server from one host machine to a new host machine. The PEM server on the new host (the target) must be installed with the same version of the PEM server installer as the original host (the source). If you don't use the same installer version, you might encounter a schema-mismatch error.

The backend database of the target server (either PostgreSQL or EDB Postgres Advanced Server) can have the same type and version or a different type and version from the backend database of the source PEM server. You can migrate a PEM server that resides on a PostgreSQL host to an EDB Postgres Advanced Server host and vice versa.

Before starting the server migration, make sure that the firewalls between the source host, the target host, and the host of any PEM agent allows connections between the services.

1. Prepare the target host.

Invoke the installer for the PEM server on the target host. You must use the same version of the PEM server installer that you used when installing the source PEM server.

The backend database of the target server can have a different version or type from the backend database of the source. If the new PEM server doesn't reside on the same type of backend database as the original server, you must ensure that the same version of the sslutils extension is installed on the new server host. The version of sslutils that's distributed with the PEM installers is freely available for download from the EDB website .

For information about installing the PEM server or the sslutils extension, see the PEM installation steps .

2. Drop existing schemas from the new PEM server.

The migration process re-creates the `pem` , `pemdata` , and `pemhistory` schemas from the source PEM server on the target PEM server. To prepare for the move, use the psql client to delete these schemas from the `pem` database on the target host. You can open the psql client at the command line or by selecting **Postgres Enterprise Manager > SQL Shell (psql)**.

When the psql client opens, connect to the `pem` backend database as the database superuser. After connecting to the `pem` database on the target host, drop the schemas:

```
DROP SCHEMA pem CASCADE;

DROP SCHEMA pemdata CASCADE;

DROP SCHEMA pemhistory CASCADE;
```

When dropping the schemas, you must include the `CASCADE` keyword, instructing the server to delete all dependent objects. When executing the command, the psql client displays a list of the dependent objects. The client confirms each the schema is removed by displaying `DROP SCHEMA` .

3. Prepare the PEM agents on the new PEM server.

Before moving the PEM server, you must identify the number of agents that are monitored by the source PEM server and create identities for that number of agents, less one, on the target server. To discover the total number of PEM agents monitored by the PEM server, connect to the `pem` database on the source host with the psql client, and query the `pem.agent` table.

```
SELECT id FROM pem.agent WHERE active = true;
```

You must manually create the number of agents that reside on the original PEM server, less one. (The PEM server installer creates one agent on the target host.) For example, if the source server contains three agents, you must manually create two more agents. Open a psql session with the `pem` database on the target server, and create the required agents:

```
CREATE USER agent<X>;
```

Where `<X>` specifies an agent number. `agent1` is created on the target host by the PEM server installer.

Then, use the `GRANT` command to assign each agent that resides on the target PEM server `pem_agent` permissions:

```
GRANT pem_agent TO agent<X>;
```

Where `<X>` specifies an agent number.

4. Generate a backup script of the source PEM server.

You can use the pg_dump utility to generate a script that contains the commands required to re-create the `pem` database on the target host. By default, pg_dump is installed in the `bin` directory under your Postgres installation. To invoke pg_dump, in the `bin` directory, enter:

```
pg_dump -U <user_name> <db_name> > <file_name>
```

Where:

```
-   `<user_name>`  specifies the name of the database superuser for the PEM backend␣
↪database. -   `<db_name>`  specifies the name of the PEM backend database. -   `<file_
↪name>`  specifies the name of the script generated by pg_dump.
```

When prompted, provide the password associated with the user specified.

The command shown instructs pg_dump to generate a script that, when executed, re-creates the `pem` database. The script is named `backup.sql` and is created in the `tmp` directory. pg_dump is connecting to the server using the credentials of the user postgres.

Invoking the pg_dump utility doesn't interrupt current database users.

Note: If the source PEM server is earlier than the 7.16 version, then you need to replace the following functions before you run pg_dump to take backup:

```
-   The `abstime` , `reltime` , and `tinterval`  datatypes are deprecated from Postgres␣
↪version 12 or later. To replace those dataypes  with `timestamptz`  data type, use␣
↪this command:
```

```
DO
$$
DECLARE
    rec record;
    cnt integer;
BEGIN
    -- Check for the deprecated type in our user info probe
    SELECT count(*) INTO cnt
    FROM pem.probe_column
    WHERE sql_data_type =  'abstime'  AND internal_name =  'valuntil' ;
      IF cnt = 0 THEN
        RETURN;
      END IF;
    ALTER TABLE pemdata.user_info
      ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    ALTER TABLE pemhistory.user_info
      ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    -- Now update the pem.probe_column itself
    UPDATE pem.probe_column
    SET sql_data_type =  'timestamptz'
    WHERE sql_data_type =  'abstime'  AND internal_name =  'valuntil' ;
END;
$$ LANGUAGE  'plpgsql' ;
```

```
-   Replace the this function to avoid any alert errors:
```

```
      CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
      template_id pem.alert_template.id%type, params text[]
      )
      RETURNS bool AS $FUNC$
      DECLARE
        res bool := TRUE;
      BEGIN
        /*
- During restoring the pem database, it does not maintain the order while
- inserting data in the table, and uses the sort table based on the
- names.
- Hence - we need to check the foreign key constraint is present before
```

(continues on next page)

(continued from previous page)

```
- validating these values.
         */
       IF EXISTS(
        SELECT 1 FROM information_schema.table_constraints
        WHERE constraint_name=alert_template_id_fkey AND
        table_name=alert AND table_schema=pem
       ) THEN
        /*
- Need to use the IS TRUE construct outside the main query, because
- otherwise if theres no template by that ID then the query would return
- 0 rows and the result of the function would be undefined and CHECK
- constraint would succeed.
- Probably this is being over-cautious, because pem.alert.template_id
- references pem.alert_template.id. But the SQL standard (probably) does
- not define the order in which the CHECK or the FOREIGN KEY constraints
- should be validated; in case CHECK is validated first, we want it to
- fail.
         */
      EXECUTE $SQL$
          SELECT (
              SELECT pem.check_array_size_equal(t.param_names, $2)
              FROM pem.alert_template AS t
              WHERE id = $1
          ) IS TRUE
        $SQL$ INTO res USING template_id, params;
      END IF;
       RETURN res;
      END
      $FUNC$ LANGUAGE plpgsql;
```

5. Move the backup to the target host.

Move the script generated by the pg_dump utility to the target host of the PEM server.

6. Restore the backup on the target host.

On the target host, in the `bin` directory under the Postgres backend database installation directory, start psql, executing the script generated by the pg_dump utility:

```
psql -U <user_name> -d pem -f <file_name>
```

Where:

```
-   `<user_name>`  specifies the name of the database superuser. The user specified must␣
↪have connection privileges for the backend database. -   `<file_name>`  specifies the␣
↪complete path to the backup script generated by pg_dump.
```

When prompted, provide the password associated with the database superuser.

The example shown uses the psql client to invoke a script named `backup.sql` to recreate the `pem` database. The script is invoked using the privileges associated with the database superuser postgres.

7. Stop the database server on the target host.

To stop the PEM server on RHEL 8.x, use the command:

```
systemctl stop <service_name>
```

Where <service_name> specifies the name of the backend database server. For a PostgreSQL backend database, the service name is postgresql-<x> . For an EDB Postgres Advanced Server backend database, the service name is edb-as-<X> , where <X> specifies the version number.

If you're using Windows, you can use the Services dialog box to control the service. To open the Services dialog box, from the Control Panel, select **System and Security > Administrative Tools**. Double-click the **Services** icon. In the Services dialog box, select the service name in the list, and select **Stop**.

8. Copy the certificate files to the target host.

You must replace the certificate files that are created when the target host is installed with the certificate files of the source host. Copy the following files from the source PEM server to the target PEM server:

- ca_certificate.crt
- ca_key.key
- root.crt
- root.crl
- server.key
- server.crt

Copy the files to the data directory under the Postgres installation that provides the backend database for the target cluster.

On Linux, the files reside in:

```
/var/lib/pgsql/<X>/data/
```

On Windows, the files reside in:

```
C:\Program Files\PostgreSQL\<X>\data
```

Where:

<X> specifies the version of PostgresSQL on your system.

The files already exist on the target cluster. Delete the existing files before performing the copy, or overwrite the existing files with the files from the source server. Once in place on the target server, the files must have the platform-specific permissions shown.

On Linux

File name	Owner	Permissions
ca_certificate.crt	postgres	-rw-------
ca_key.key	postgres	-rw-------
root.crt	postgres	-rw-------
root.crl	postgres	-rw-------
server.key	postgres	-rw-------
server.crt	postgres	-rw-r--r--

On Linux, the certificate files must be owned by postgres. Use the following command to modify the ownership of the files:

```
chown postgres <file_name>
```

Where `file_name` specifies the name of the certificate file.

Only the owner of the `server.crt` file can modify the file, but any user can read it. Use the following command to set the file permissions for the `server.crt` file:

```
chmod 644 server.crt
```

Only the owner of the other certificate files can modify or read the files. Use the following command to set the file permissions:

```
chmod 600 <file_name>
```

Where `file_name` specifies the name of the file.

On Windows

On Windows, the service account that performed the PEM server and backend database installation on the target host must own the certificate files moved from the source host. If you invoked the PEM server and Postgres installer using **Run as Administrator** from the installer context menu, the owner of the certificate files is Administrators.

To review and modify file permissions on Windows, right-click the file name and select **Properties**. On the **Security** tab, select a group or user name to view the assigned permissions. Select **Edit** or **Advanced** to open dialog boxes that allow you to modify the permissions associated with the selected user.

9. Move the PEM agent certificate files to the PEM server host.

You must move the certificate files used by the PEM agent of the source PEM server to the target host. This step is platform specific.

On Linux

Copy the `agent1.key` and `agent1.crt` files from the source host to the target host. By default, on Linux, the files are installed in `/root/.pem` . Copy the files to the same directory on the target host.

File ownership and permissions of the files must be set to:

File name	Owner	Permissions
agent1.key	root	-rw-------
agent1.crt	root	-rw-r--r--

If necessary, navigate to `/root/.pem` , and use the following commands to modify the permissions and ownership of the `agent1.key` file:

```
chmod 600 agent1.key

chown root agent1.key
```

Use the following commands to modify the permissions and ownership of the `agent1.crt` file:

```
chmod 644 agent1.crt

chown root agent1.crt
```

On Windows

Copy the `agent1.key` and `agent1.crt` files from the source host to the target host. On Windows, the files are located in:

```
C:\Users\<user_name>\AppData\Roaming\pem
```

Where `user_name` is the name of the user that invoked the PEM installer.

The ownership and permissions associated with the certificate files on the target machine must match the ownership and permissions of the certificate files on the source machine. If you invoked the PEM server and Postgres installer using **Run as Administrator** on the installer context menu, the owner of the agent certificate files is Administrators.

To review and modify file permissions on Windows, right-click the file name and select **Properties**. On the **Security** tab, select a group or user name to view the assigned permissions. Select **Edit** or **Advanced** to open dialog boxes that allow you to modify the permissions associated with the selected user.

10. Update the `pg_hba.conf` files on the target host.

Modify the `pg_hba.conf` file on the target host to allow connections from each PEM agent. By default, the `pg_hba.conf` file is located in the data directory under your Postgres installation.

11. Start the server on the target host.

After modifying the `pg_hba.conf` file, you must restart the server for the changes to take effect.

To restart the database server on Linux, use the command:

```
/etc/init.d/<service_name> start
```

Where `service_name` is the name of the backend database server.

On Windows, you can use the Services dialog box to control the service. To open the Services dialog box, on the Control Panel, select **System and Security > Administrative Tools**. Double-click the **Services** icon. When the Services dialog box opens, select the service name in the list, and start the service.

12. Connecting monitored agents to the new PEM server host.

To instruct existing PEM agents to connect to the new PEM server host, you must:

- Ensure that the PEM agent host can connect to the new PEM server host.
- Modify the registry on each Windows host with a PEM agent or the agent configuration files on each Linux host with a PEM agent, specifying the IP address and port of the new PEM server.
- Restart the PEM agent' s service. These steps are platform specific: - *If the PEM agent resides on Linux* - *If the PEM agent resides on Windows*

If the PEM agent resides on Linux

Use your choice of editor to modify the `agent.cfg` file, specifying the new IP address and port number of the PEM server in the `pem_host` and `pem_port` parameters.

By default, the `agent.cfg` file is located in:

`/usr/edb/pem/agent/etc/agent.cfg`

After modifying the `agent.cfg` file, you must restart the PEM agent service. You can use the `pemagent` service script on the Linux command line to restart the service:

```
/etc/init.d/pemagent restart
```

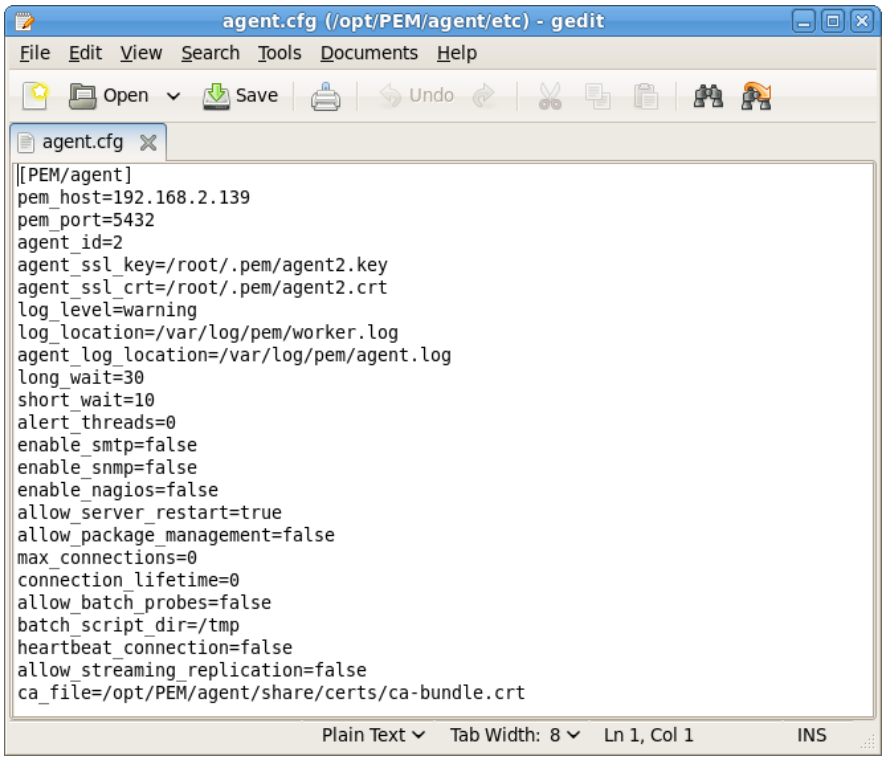


Fig. 2: The agent.cfg file

If the PEM agent resides on Windows

Before modifying the Windows registry on the monitored node, confirm that the firewall on the host of the PEM agent allows connections to the PEM server. After confirming that the PEM agent host can connect to the PEM server host, you can use the Windows Registry Editor to review and edit the `PEM_HOST` and `PEM_PORT` entries to ensure that they correctly identify the host and port used by the PEM server. To open the Registry Editor, enter `regedit` in the Windows Run dialog box or in the Windows start menu search box. Navigate through the registry tree control to view or modify registry entries.

The PEM agent registry entries are located at: `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent`

The `PEM_HOST` and `PEM_PORT` entries must specify the address and port number of the new PEM server on the target host. To modify a registry entry, right-click the entry name and select **Modify** from the context menu. Then use the Edit String dialog box to make any changes to the value of the entry. After you finish, select **OK**.

After modifying the registry, you must restart the PEM agent' s service. You can use the Services dialog box, accessed through the Windows Control Panel, to restart the `Postgres Enterprise Manager - pemagent` service.

After moving the server, change the connection properties in any installed PEM clients to connect to the new host of the PEM server, agents, and monitored servers.

Note: After moving the server, if you encounter this error while creating the server in the PEM web interface:

```
Error - User does not have enough permission to add new server.
        Please contact the administrator to grant pem_database_server_registration role
        to the enterprisedb user.
```

Resolve the error by updating the roles and granting appropriate permissions:

```
UPDATE pem.roles SET rolid = pr.oid FROM pg_roles pr WHERE pr.rolname = pem_ ||␣
↪component;
```

3.4 Uninstalling

To uninstall PEM components see:

- ```
  :ref:`Uninstalling Postgres Enterprise Manager components on Linux <Uninstalling␣
  ↪Postgres Enterprise Manager components on Linux>`
  ```
- ```
  :ref:`Uninstalling Postgres Enterprise Manager components on Windows <Uninstalling␣
  ↪Postgres Enterprise Manager components on Windows>`
  ```

3.4.1 Uninstalling Postgres Enterprise Manager components on Linux

The process of uninstalling the PEM server or Agent is platform-specific. The name of the package for PEM server is `edb-pem-server` and for PEM Agent is `edb-pem-agent` .

If you uninstall the PEM server package from a host, the PEM Agent package installed on the same host doesn' t get uninstalled. But if you uninstall the PEM Agent package, then the PEM server package installed on the same host also gets uninstalled.

Note: Before uninstalling the PEM Agent, you need to de-register the agent first. You can de-register the agent using the *pemworker* command-line utility. After that, you can proceed with the uninstallation steps.

Uninstalling PEM components from RHEL or Rocky Linux or AlmaLinux hosts

You can use variations of the `rpm` command to remove the installed packages from Rocky Linux/AlmaLinux/RHEL 8.x hosts. Also, you can use the `dnf remove` command to remove the installed package from Rocky Linux/AlmaLinux/RHEL 8.x hosts. Note that removing a package does not damage the PEM data directory.

Include the `-e` option when invoking the rpm command to remove an installed package; the command syntax is:

```
rpm -e package_name
```

Where `package_name` is the name of the package that you would like to remove.

- You can use `dnf remove` command to remove the pem server or Agent along with the `edb-pem` and `edb-pem-docs` dependencies on Rocky Linux or AlmaLinux or RHEL 8.x hosts. To remove a package, open a terminal window, assume superuser privileges, and enter the command:

```
dnf remove package_name
```

Uninstalling PEM components from Debian or Ubuntu hosts

You can use `apt-get remove` or `apt-get purge` command to uninstall the PEM server or Agent package from a Debian or Ubuntu host:

- To uninstall PEM server or Agent from a Debian or Ubuntu host without impacting the configuration files and data directories, invoke the following command:

```
apt-get remove package_name
```

- To uninstall PEM server or Agent along with the configuration files and data directory, invoke the following command:

```
apt-get purge package_name
```

Where `package_name` is the name of the package that you would like to remove.

Uninstalling PEM components from SLES hosts

To uninstall PEM server or Agent from a SLES host, invoke the following command:

```
zypper remove package_name
```

Where `package_name` is the name of the package that you would like to remove.

3.4.2 Uninstalling Postgres Enterprise Manager components on Windows

If you uninstall the PEM server from a host, the PEM agent installed on the same host is uninstalled. But if you uninstall the PEM agent, then the PEM server installed on the same host will not be uninstalled.

You can use the Windows `Add/Remove Programs` application to remove PEM components from a Windows host. Select the `Add/Remove Programs` option from the Windows `Control Panel` . When the `control panel` opens, locate the name of the PEM component in the program list. Click the `Remove` button to remove the component.

You can also invoke the uninstaller that resides at the following location:

For the PEM Server, `C:\Program Files\edb\pem\server\uninstall-pemserver`

For the PEM Agent, `C:\Program Files\edb\pem\agent\uninstall-pemagent`

3.5 Troubleshooting

3.5.1 Server installation errors

RHEL 8

1. While installing the PEM server on RHEL 8, if you see this error:

```
[root@etpgxlt firstuser]# dnf install edb-pem
Updating Subscription Management repositories.
Last metadata expiration check: 0:01:33 ago on Wed 30 Mar 2022 01:28:16 AM EDT.
Error:
```

(continues on next page)

(continued from previous page)

```
 Problem: problem with installed package python3-mod_wsgi-4.6.4-4.el8.s390x
  - package python39-mod_wsgi-4.7.1-4.module+el8.4.0+9822+20bf1249.s390x conflicts with␣
↪python3-mod_wsgi provided by python3-mod_wsgi-4.6.4-4.el8.s390x
  - package python39-mod_wsgi-4.7.1-4.module+el8.4.0+9822+20bf1249.s390x conflicts with␣
↪python3-mod_wsgi provided by python3-mod_wsgi-4.6.4-3.el8.s390x
  - package edb-pem-server-8.4.0-7.rhel8.s390x requires python39-mod_wsgi >= 4.7, but␣
↪none of the providers can be installed
  - package edb-pem-8.4.0-7.rhel8.s390x requires edb-pem-server = 8.4.0-7.rhel8, but␣
↪none of the providers can be installed
  - cannot install the best candidate for the job
(try to add --allowerasing to command line to replace conflicting packages or --skip-
↪broken to skip uninstallable packages or --nobest to use not only best candidate␣
↪packages)
[root@etpgxlt firstuser]#
```

Remove the `python3-mod_wsgi` package first:

```
dnf remove python3-mod_wsgi
```

Try installing the PEM server again.

2. On RHEL 8, if you see this error in the worker.log after configuring the PEM server:

```
Tue Nov 28 03:02:19 2023 WARNING: Error clearing zombies: ERROR:  failed to JIT module:␣
↪Added modules have incompatible data layouts: E-m:e-i1:8:16-i8:8:16-i64:64-f128:64-
↪a:8:16-n32:64 (module) vs E-m:e-i1:8:16-i8:8:16-i64:64-f128:64-v128:64-a:8:16-n32:64␣
↪(jit)
CONTEXT:  SQL statement "WITH running_agent_job AS (
        SELECT j.jobid, j.agent_id
        FROM pem.job j LEFT JOIN pem.joblog jl ON (j.jobid = jl.jlgjobid)
        WHERE jl.jlgstatus = r AND agent_id = $1 AND j.jobarid != agent_runtime_id
        ORDER BY j.jobid, jl.jlgjobid
        FOR UPDATE SKIP LOCKED
    ), joblog_status_update AS (
        UPDATE pem.joblog jl SET jlgstatus=d
        FROM running_agent_job r
        WHERE r.jobid = jl.jlgjobid AND jl.jlgstatus=r
        RETURNING r.agent_id, jl.jlgjobid, jl.jlgid
    ), jobsteplog_status_update AS (
        UPDATE pem.jobsteplog js SET jslstatus=d
        FROM joblog_status_update jl
        WHERE js.jsljlgid = jl.jlgid AND js.jslstatus=r
        RETURNING jl.agent_id, jl.jlgjobid AS job_id
    )
    UPDATE pem.job j SET jobprocessid=NULL, jobnextrun=NULL, jobarid=NULL
    FROM (SELECT DISTINCT agent_id, job_id FROM jobsteplog_status_update) js
    WHERE js.job_id = j.jobid"
PL/pgSQL function pem.clear_job_zombies(integer) line 8 at SQL statement
```

To resolve the error, set the `jit` parameter to `off` in `postgresql.conf` file of the backend database server:

```
jit=off
```

Restart the backend database server.

3.5.2 Reconfiguring the PEM server

In some situations, you might need to uninstall the PEM server, reinstall it, and then configure the server again. To do so:

1. Remove the PEM server configuration and uninstall:

```
/usr/edb/pem/bin/configure-pem-server.sh -un
```

2. Remove the PEM packages:

```
yum erase edb-pem-server
```

3. Drop the `pem` database:

```
DROP DATABASE pem
```

4. Move the certificates from `/root/.pem/` to another location:

```
mv /root/.pem/* <new_location>
```

5. Move the `agent.cfg` file from `/usr/edb/pem/agent/etc/agent.cfg` to another location:

```
mv /usr/edb/pem/agent/etc/agent.cfg <new_location>
```

6. Then, configure the PEM server again:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

3.5.3 PEM web client not loading

If the PEM web client isn't loading, check the HTTPD log in the `/var/log/httpd/error_log` file.

If you see the following message in the log:

```
Truncated or oversized response headers received from daemon process edbpem: /usr/edb/
↪pem/web/pem.wsgi
```

Add the following statement at the bottom of the Apache `httpd.conf` file located in the `/etc/httpd/conf` folder:

```
WSGIApplicationGroup %{GLOBAL}
```

Restart the HTTPD server after adding the statement:

```
sudo systemctl restart httpd.service
```

3.5.4 Error connecting to PostgreSQL server

When connecting to a PostgreSQL server, you might get one of these error messages. Review the message carefully. Each error message attempts to incorporate the information you need to resolve the problem.

- **Connection to the server has been lost**: This error message indicates that the connection attempt took longer than the specified threshold. There might be a problem with the connection properties provided on the Server dialog box, network connectivity issues, or the server might not be running.
- **Could not connect to Server: Connection refused**: There are two possible reasons for this error:
  - The database server isn't running. Start the server.
  - The server isn't configured to accept TCP/IP requests on the address shown.

For security reasons, a PostgreSQL server "out of the box" doesn't listen on TCP/IP ports. Instead, you must enable it to listen for TCP/IP requests. Add `tcpip = true` to the `postgresql.conf` file for Versions 7.3.x and 7.4.x. Add `listen_addresses='*'` for Version 8.0.x and above. These additions make the server accept connections on any IP interface.

For more information, refer to the PostgreSQL documentation about runtime configuration .

- **FATAL: no pg_hba.conf entry**: If PEM displays this message when connecting, your server can be contacted correctly over the network, but it isn't configured to accept your connection. Your client wasn't detected as a legal user for the database.

To connect to a server, configure the `pg_hba.conf` file on the database server to accept connections from the host of the PEM client. Modify the `pg_hba.conf` file on the database server host, and add an entry in the form:

- `host template1 postgres 192.168.0.0/24 md5` for an IPV4 network
- `host template1 postgres ::ffff:192.168.0.0/120 md5` for an IPV6 network

For more information, see the PostgreSQL documentation about client authentication .

- **FATAL: password authentication failed**: The `password authentication failed for user` error message indicates there might be a problem with the password you entered. Retry the password to confirm you entered it correctly. If the error message returns, make sure that you have the correct password, that you are authorized to access the server, and that the access was correctly configured in the server's `postgresql.conf` configuration file.

3.5.5 PEM web server status check

Run this command to check whether the PEM webserver is up and running:

```
curl https://<SERVER_ADDR>:8443/pem/misc/ping -k -i
__OUTPUT__
PING
```

Where `SERVER_ADDR` is the IP address of your PEM server. The output `PING` confirms the PEM web server is up and running.

CHAPTER 4

Administering

4.1 Changing the default port

By default, the 8443 port is assigned for the web services at the time of configuration of the PEM server. You can change the port after configuration by changing a few parameters in the Apache configuration files. The Apache configuration files names and locations are platform specific.

4.1.1 For RHEL

1. Edit the file `/etc/httpd/conf.d/edb-ssl-pem.conf`, replacing 8443 with your port number in the following parameters:

```
Listen 8443
<VirtualHost _default_:8443>
ServerName localhost:8443
RewriteRule ^(.*)$ https://%{HTTP_HOST}:8443%{REQUEST_URI} [L,R=301]
```

1. If `semanage` isn't installed, install it:

```
sudo yum -y install policycoreutils-python
```

1. Check whether your port is listed:

```
semanage port -l | grep http
```

1. If your port doesn't appear in the list, run this command:

```
sudo semanage port -a -t http_port_t -p tcp <your_port_number>
```

1. Restart the httpd service:

```
sudo systemctl restart httpd
```

You can now access the PEM web interface using your port. For more details, see *Accessing the web interface* .

4.1.2 For Debian and Ubuntu

1. Edit the file `/etc/apache2/sites-available/edb-ssl-pem.conf` , replacing 8443 with your port number in the following parameters:

```
Listen 8443
<VirtualHost _default_:8443>
ServerName HTTPD-EDBPEM-SERVER-v8:8443
```

1. Restart the httpd service:

```
sudo systemctl restart apache2
```

You can now access the PEM web interface using your port. For more details, see *Accessing the web interface* .

4.2 Registering a Postgres server

Before you can manage or monitor a database server with PEM, you must register the database server with PEM and bind the database server to a previously registered PEM agent. You can bind a database server to a remote agent (an agent that resides on a different host). However, if the agent resides on a different host, it doesn' t have access to all of the statistical information about the instance.

4.2.1 What does registering a Postgres server do?

Registering a server does two things:

- It adds the server to the server tree in the PEM web application, allowing users to open connections from the web application to the server. - It binds the server to a particular agent, meaning that agent will open connections to the server to execute probes and scheduled jobs.

After registration, the agent opens a connection to the specified database. It uses this connection to determine the other databases it can connect to with the same user and then spawns one connection to each.

There are three ways to register a server.

- *Using the pemworker utility to register a server* from the command line on the host where the agent you want to bind is located. - *Registering a server with automatic server discovery* in the PEM web application to automatically register Postgres servers located on the same host as an agent. - *Manually registering a database server* by adding all the details in the PEM web application.

4.2.2 Using the pemworker utility to register a server

You can use the pemworker utility to register a Postgres server. During registration, the pemworker utility binds the new server to the agent that resides on the system from which you invoked the registration command.

To register a server on a Linux host, use the command:

```
pemworker --register-server
```

To register a server on a Windows host, use the command:

```
pemworker.exe REGISTER-SERVER
```

Append command line options to the command string when invoking the pemworker utility. Follow each option with a corresponding value.

Option	Description
--pem-user	Specifies the name of the PEM administrative user (must have the pem_admin role) on the PEM server to use to write the server details to the PEM database. Required.
--pem-port	Specifies the port on which to connect to the PEM server when writing the server details to the PEM database. Optional, defaults to the value of *--pem-port* set during agent registration (PEM 9.8 or newer only). Otherwise defaults to 5432.
Server parameters	These parameters are used to populate the connection properties in the PEM web application. They're used when a user connects from the PEM web application to the monitored server. They're also used for connections from the agent to the monitored server unless overridden by Agent Server Binding parameters (see the **Agent-server binding parameters** in this table).
--server-addr	Specifies the IP address or fully qualified domain name of the monitored server. On Linux systems, you can leave the address field blank to use the default PostgreSQL Unix Domain Socket on the local machine. Or you can set it to an alternative path containing a PostgreSQL socket. If you enter a path, the path must begin with a forward slash (/). Required.
--server-port	Specifies the port number of the monitored server. Required.
--server-database	Specifies the name of the database on the monitored server to which to connect. Required.
--server-user	Specifies the name of the user used to connect to the monitored server. Required.
--server-service-name	Specifies the name of the operating system service that manages the monitored Postgres server, for example, a systemd service unit on Linux. Optional.
Agent-server binding parameters	Use these parameters to override the server parameters. Use these if you want the agent to connect to the monitored server using different credentials from those used by the PEM web application.
--asb-host-name	Specifies the IP address or fully qualified domain name of the monitored server. Optional, defaults to *--server-addr* if not supplied.
--asb-host-port	Specifies the port number of the monitored server. Optional, defaults to *--server-port* if not supplied.
--asb-host-db	Specifies the name of the database on the monitored server to which the agent connects. Optional, defaults to *--server-database* if not supplied.
--asb-host-user	Specifies the name of the user used by the agent to connect to the monitored server. Optional, defaults to *--server-user* if not supplied.
--asb-ssl-mode	Specifies the type of SSL authentication to use for connections. Supported values include: *prefer*, *require*, *disable*, *verify-CA*, and *verify-full*. Optional, defaults to *prefer*.
Server metadata	These parameters determine how the server is shown in the PEM web application.
--group	Specifies the name of the group in which the server is displayed. Optional.

continues on next page

Table 1 – continued from previous page

Option	Description
--team	Specifies a Postgres role on the PEM server to be assigned as the team to which the monitored server belongs. Only users with this role can access the server. Optional, defaults to none, meaning all users can access.
--owner	Specifies a Postgres user on the PEM server to assign as the owner of the monitored server. Optional, defaults to *--pem-user*.
--display-name <name>	Specifies the display name of the monitored database server. Optional, defaults to the system hostname.
Other parameters	
--remote-monitoring	Set to *yes* if the server isn't located on the same host as the agent. When remote monitoring is enabled (*yes*), agent-level statistics for the monitored server aren't available for custom charts and dashboards, and the remote server isn't accessible by some PEM utilities (such as Audit Manager, Capacity Manager, Log Manager, Postgres Expert, and Tuning Wizard). Optional, defaults to *no*.
--efm-cluster-name	Specifies the name of the EDB Failover Manager cluster that monitors the server (if applicable). Optional.
--efm-install-path	Specifies the complete path to the installation directory of EDB Failover Manager (if applicable). Optional.
--config-dir	Specifies the directory path of the agent configuration file. Optional, defaults to *<pem-worker path>/../etc*.

Set the environment variable `PEM_SERVER_PASSWORD` to provide the password for the PEM server to allow the pemworker to connect as a PEM admin user.

Set the environment variable `PEM_MONITORED_SERVER_PASSWORD` to provide the password of the database server being registered and monitored by the PEM agent.

If you don't provide the password, a password authentication error occurs. The PEM server acknowledges that the server was registered properly.

Examples

This example registers a server using only the required parameters. The user `admin01` will be used to connect to the PEM server. The credentials supplied for the `--server-*` parameters will be used to create a connection from the agent to the monitored server. The same details will also be used to populate the server connection details in the PEM web application.

```
pemworker --register-server \
          --pem-user admin01 \
          --server-addr pg123.prod.infra.business \
          --server-port 5432 \
          --server-database postgres \
          --server-user prod_admin
```

The following example specifies different parameters to used for the connection from the agent to the monitored server. They override the fully qualified domain name provided in `--server-addr` with `localhost` instead. They also override the user to use a `local_monitor` user. This might be a user who's permitted to connect only from the same host, for example.

This example also specifies a service name, `postgresql` , which means the PEM agent can restart this server to apply configuration changes.

```
pemworker --register-server \
          --pem-user admin01 \
          --server-addr pg123.prod.infra.business \
          --server-port 5432 \
          --server-database postgres \
          --server-user prod_admin \
          --server-service-name postgresql \
          --asb-host-name localhost \
          --asb-host-user local_monitor
```

Using the pemworker utility to unregister a server

You can use the pemworker utility to unregister a database server. To unregister a server, invoke the pemworker utility.

On a Linux host, use the command:

`pemworker --unregister-server`

On a Windows host, use the command:

`pemworker.exe UNREGISTER-SERVER`

Append command line options to the command string when invoking the pemworker utility. Follow each option with by a corresponding value:

Option	Description
--pem-user	Specifies the name of the PEM administrative user. Required.
--server-addr	Specifies the IP address of the server host or the fully qualified domain name. On Unix-based systems, you can leave the address field blank to use the default PostgreSQL Unix Domain Socket on the local machine. Or, you can set it to an alternatibe path containing a PostgreSQL socket. If you enter a path, the path must begin with a forward slash (/). Required.
--server-port	Specifies the port number of the host. Required.
--config-dir	Specifies the directory path of the agent configuration file. Optional, defaults to *<pem-worker path>/../etc*.

Use the `PEM_SERVER_PASSWORD` environment variable to provide the password for the PEM server to allow the pemworker utility to connect to the PEM server as `--pem-user` .

If you don' t provide the password, a password authentication error occurs. The PEM server acknowledges that the server is unregistered.

4.2.3 Registering a server with automatic server discovery

If the server you want to monitor resides on the same host as the monitoring agent, you can use the Auto Discovery dialog box to simplify the registration and binding process.

To enable auto discovery for a specific agent, you must enable the Server Auto Discovery probe. To do so, select the PEM agent in the PEM client tree, and select **Management > Manage Probes**. When the **Manage Probes** tab opens, confirm that the slider in the **Enabled?** column is set to **Yes**.

To open the Auto Discovery dialog box, select a PEM agent in the PEM client tree and select **Management > Auto Discovery**.

When the Auto Discovery dialog box opens, the **Discovered Database Servers** box displays a list of servers that currently aren't being monitored by a PEM agent. Select the box next to a server name to display information about the server in the **Server Connection Details** box and connection properties for the agent in the **Agent Connection Details** box.

Use **Check All** to select the box next to all of the displayed servers or **Uncheck All** to clear all of the boxes to the left of the server names.

The fields in the **Server Connection Details** box provide information about the server that PEM monitors:

- Accept or modify the name of the monitored server in the **Name** field. The specified name is displayed in the tree of the PEM client.
- Use the **Server group** list to select the server group under which the server is displayed in the PEM client tree.
- Use the **Host name/address** field to specify the IP address of the monitored server.
- The **Port** field displays the port that's monitored by the server. You can't modify this field.
- Provide the name of the service in the **Service ID** field. You must provide the service name to enable some PEM functionality.
- The **Maintenance database** field specifies the database to use for the initial connection and any global maintenance operations. Customize the content of the **Maintenance database** field for your installation.

The fields in the **Agent Connection Details** box specify the properties for the PEM agent to use when connecting to the server:

- The **Host** field displays the IP address used for the PEM agent binding.
- The **User name** field displays the name used by the PEM agent when connecting to the selected server.
- The **Password** field displays the password associated with the specified user name.
- Use the **SSL mode** field to specify your SSL connection preferences.

After you finish specifying the connection properties for the servers that you're binding for monitoring, select **OK** to register the servers.

After selecting **OK**, the newly registered server is displayed in the PEM tree and is monitored by the PEM server.

4.2.4 Manually registering a database server

To manage or monitor a database server with PEM, you must:

- Register your EDB Postgres Advanced Server or PostgreSQL server with the PEM server.
- Bind the database server to a PEM agent.

You can use the Create Server dialog box to provide registration information for a server, bind a PEM agent, and display the server in the PEM client tree. To open the Create Server dialog box, select **Object > Create > Server**.

Note:

- You must ensure the *pg_hba.conf* file of the Postgres server that you're registering allows connections from the host of the PEM client before attempting to connect. - Only database superusers or users with the *pem_admin* role can bind a database server to a PEM agent.

Use the **General** tab to describe the general properties of the server:

- Use the **Name** field to specify a name for the server. The name identifies the server in the PEM browser tree.

- You can use groups to organize your servers and agents in the tree. Using groups can help you manage large numbers of servers more easily. For example, you can have a production group, a test group, or LAN-specific groups. Use the **Group** list to select the server group in which to display the new server.
- Use the **Team** field to specify a PostgreSQL role name. Only PEM users who are members of this role, who created the server initially, or have superuser privileges on the PEM server see this server when they log in to PEM. If this field is left blank, by default all PEM users see the server. You can use the `show_objects_with_no_team` parameter in the Server Configuration dialog box to change the behavior. If `show_objects_with_no_team` is set to `false` , the server with no team isn't visible to all other users.
- Use the **Background** color selector to select the color to display in the PEM tree behind database objects that are stored on the server.
- Use the **Foreground** color selector to select the font color of labels in the PEM tree for objects stored on the server.
- Select **Connect now?** to attempt a server connection when you select **Save**. Clear **Connect now?** if you don't want the PEM client to validate the specified connection parameters until a later connection attempt.
- Provide notes about the server in the **Comments** field.

Use the **Connection tab** to specify connection details for the server:

- Specify the IP address of the server host or the fully qualified domain name in the **Host name/address** field. On Unix-based systems, leave the address field blank to use the default PostgreSQL Unix Domain Socket on the local machine. Or you can set an alternative path containing a PostgreSQL socket. If you enter a path, the path must begin with a forward slash (/).
- Specify the port number of the host in the **Port** field.
- Use the **Maintenance database** field to specify the name of the initial database for PEM to connect to that's expected to contain pgAgent schema and adminpack objects installed (both optional). On PostgreSQL version 8.1 and later, the maintenance DB is normally called `postgres` . On earlier versions `template1` is often used, although it's better to create a `postgres` database to avoid cluttering the template database.
- Set **Kerberos Authentication** to **Yes** to use the Kerberos authentication for a monitored server. By default, the monitored server uses the same authentication method as the PEM server. If the monitored server doesn't want to use Kerberos authentication, then set `ALLOW_DATABASE_CONNECTION_WITHOUT_KERBEROS` to `TRUE` in the `config_local.py` file.
- Specify the name to use when authenticating with the server in the **Username** field.
- Provide the password associated with the specified user in the **Password** field.
- Select **Save password?** to store passwords in encrypted format in a PEM backend database for later reuse. Each password is stored on a per-user, per-server basis and isn't shared with other team members. PEM uses the saved password to connect the database server next time. To remove a saved password, disconnect the database server first, and then select **Object > Clear Saved Password**.
- Use the **Role** field to specify the name of the role that's assigned the privileges for the client to use after connecting to the server. This value allows you to connect as one role and then assume the permissions of another role (the one you specified in this field) when the connection is established. The connecting role must be a member of the role specified.

Use the **SSL** tab to configure SSL:

- In the **SSL mode** field, select the type of SSL connection for the server to use. For more information about using SSL encryption, see the PostgreSQL documentation .

You can use the platform-specific file manager dialog box to upload files that support SSL encryption to the server. To open the file manager, select the icon located to the right of each of the following fields:

- Use the **Client certificate** field to specify the file containing the client SSL certificate. This file replaces the default `~/.postgresql/postgresql.crt` file if PEM is installed in Desktop mode and `<STORAGE_DIR>/<USERNAME>/.postgresql/postgresql.crt` if PEM is installed in Web mode. This parameter is ignored if an SSL connection isn't made.
- Use the **Client certificate key** field to specify the file containing the secret key used for the client certificate. This file replaces the default `~/.postgresql/postgresql.key` if PEM is installed in Desktop mode and `<STORAGE_DIR>/<USERNAME>/.postgresql/postgresql.key` if PEM is installed in Web mode. This parameter is ignored if an SSL connection isn't made.
- Use the **Root certificate** field to specify the file containing the SSL certificate authority. This file replaces the default `~/.postgresql/root.crt` file. This parameter is ignored if an SSL connection isn't made.
- Use the **Certificate revocation list** field to specify the file containing the SSL certificate revocation list. This list replaces the default list, found in `~/.postgresql/root.crl` . This parameter is ignored if an SSL connection isn't made.
- When **SSL compression?** is set to **True**, data sent over SSL connections is compressed. The default value is **False** (compression is disabled). This parameter is ignored if an SSL connection isn't made.

Warning: Certificates, private keys, and the revocation list are stored in the per-user file storage area on the server, which is owned by the user account under which the PEM server process is run. This means that administrators of the server might be able to access those files. Use caution before using this feature.

Use the **SSH Tunnel** tab to configure SSH tunneling. You can use a tunnel to connect a database server through an intermediary proxy host to a server that resides on a network to which the client might not be able to connect directly.

- Set **Use SSH tunneling** to **Yes** to use an SSH tunnel when connecting to the specified server.
- Specify the name or IP address of the SSH host (through which client connections are forwarded) in the **Tunnel host** field.
- Specify the port of the SSH host through which client connections are forwarded in the **Tunnel port** field.
- Specify the name of a user with login privileges for the SSH host in the **Username** field.
- Specify the type of authentication to use when connecting to the SSH host in the **Authentication** field.
- Select **Password** to use a password for authentication to the SSH host. This is the default.
- Select **Identity file** to use a private key file when connecting.
- If the SSH host is expecting a private key file for authentication, use the **Identity file** field to specify the location of the key file.
- If the SSH host is expecting a password, use the **Password** field to specify the password. If an identity file is being used, specify the passphrase.

Use the **Advanced** tab to specify details that are used to manage the server:

- Specify the IP address of the server host in the **Host Address** field.
- Use the **DB restriction** field to specify a SQL restriction to use against the pg_database table to limit the databases displayed in the tree. For example, you might enter: `'live_db'` , `'test_db'` to display only the `live_db` and `test_db` databases. You can also limit the schemas shown in the database from the database Properties dialog box by entering a restriction against pg_namespace .
- Use the **Password file** field to specify the location of a password file (`.pgpass`). The `.pgpass` file allows a user to log in without providing a password when they connect. The file must be present on the PEM server. For more information, see the Postgres documentation .

- Use the **Service ID** field to specify parameters to the database service process. For servers that are stored in the Enterprise Manager directory, enter the service ID. On Windows machines, this is the identifier for the Windows service. On Linux machines, the name of the init script used to start the server is `/etc/init.d` and the name of the systemd script to start the server is `systemctl` . For example, the name of the EDB Postgres Advanced Server 11 service is `edb-as-11` . For local servers, the setting is operating system dependent:
- If the PEM client is running on a Windows machine, it can control the postmaster service if you have enough access rights. Enter the name of the service. In case of a remote server, prepend it with the machine name (such as `PSE1\pgsql-8.0`). PEM automatically discovers services running on your local machine.
- If the PEM client is running on a Linux machine, it can control processes running on the local machine if you have enough access rights. Provide a full path and needed options to access the `pg_ctl` program. When executing service control functions, PEM appends status/start/stop keywords to this. For example:

```
sudo /usr/pgsql-x/bin/pg_ctl -D /var/lib/pgsql/x/data
```

where `x` is the version of the PostgreSQL database server.

- If the server is a member of a Failover Manager cluster, you can use PEM to monitor the health of the cluster and to replace the primary node if necessary. To enable PEM to monitor Failover Manager, use the **EFM cluster name** field to specify the cluster name. The cluster name is the prefix of the name of the Failover Manager cluster properties file. For example, if the cluster properties file is named `efm.properties` , the cluster name is efm.
- If you're using PEM to monitor the status of a Failover Manager cluster, use the **EFM installation path** field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in `/usr/edb/efm-x.x/bin` , where `x.x` specifies the Failover Manager version.

Use the **PEM Agent** tab to specify connection details for the PEM agent.

On the **Connection Parameters** tab:

- Set **Remote monitoring?** to **Yes** to indicate that the PEM agent doesn't reside on the same host as the monitored server. When remote monitoring is enabled, agent level statistics for the monitored server aren't available for custom charts and dashboards, and the remote server can't be accessed by some PEM utilities (such as Audit Manager, Capacity Manager, Log Manager, Postgres Expert, and Tuning Wizard).
- Use the **Bound agent** list to select a PEM agent. One agent can monitor multiple Postgres servers.
- In the **Host** field, enter the IP address or socket path for the agent to use when connecting to the database server. By default, the agent uses the host address shown on the **General** tab. On a Unix server, you might want to specify a socket path, such as `/tmp` .
- Enter the port number for the agent to use when connecting to the server. By default, the agent uses the port defined on the **Properties** tab.
- Use the **SSL** field to specify an SSL operational mode. Specify **require**, **prefer**, **allow**, **disable**, **verify-ca**, or **verify-full**.

Mode	Description
require	Require SSL encryption for transactions between the server and the agent.
prefer	Use SSL encryption between the server and the agent if SSL encryption is available.
allow	Allow the connection to use SSL if required by the server.
disable	Disable SSL encryption between the agent and the server.
verify-ca	Require SSL encryption and require the server to authenticate using a certificate registered by a certificate authority.
verify-full	Require SSL encryption and require the server to authenticate using a certificate registered by a trusted certificate authority.

For more information about using SSL encryption, see the PostgreSQL documentation .

- Use the **Database** field to specify the name of the Postgres database to which the agent initially connects. If you're registering a EDB Postgres Distributed database node then specify the EDB Postgres Distributed-enabled database name in this field.
- In the **User name** field, specify the name of the role for the agent to use when connecting to the server. The specified role must be a database superuser for all of the features to work as expected. For the list of features that don't work if the specified role isn't a database superuser, see *Setting agent privileges* . If you're using the Kerberos authentication method, then specify the user having the pgd_monitor or pgd_superuser role in this field.

If you're using Postgres version 10 or later, you can use the pg_monitor role to grant the required privileges to a non-superuser. For information about the pg_monitor role, see Default Roles .

- In the **User name** field, specify the name of the user for the agent to use when connecting to the server. The specified role must be a database superuser for all of the features to work as expected. If you're using Postgres version 10 or later, you can use the pg_monitor role to grant the required privileges to a nonsuperuser. For information about the pg_monitor role, see Default Roles .
- Specify the password for the agent to use when connecting to the server in the **Password** field and **Confirm password** fields. If you don't specify a password, you must configure the authentication for the agent manually. For example, you can use a `.pgpass` file, which must be present and accessible on the system where agent is installed.

On the **Advanced** tab:

- Set **Allow takeover?** to **Yes** to specify that another agent can take over the server. This feature allows an agent to take responsibility for the monitoring of the database server if, for example, the server moved to another host as part of a high-availability failover process.
- Use the plus sign (**+**) to add the database you want to exclude from the PEM monitoring. You can't exclude the database mentioned on the **Connection Parameters** tab of the **PEM Agent** tab.

Note: The database-level probes don't execute for excluded databases, but the server-level probes can collect the database statistics.

Use the **General** tab under the **BART** tab to describe the general properties of the BART server that map to the PEM server:

- Use the **BART server** field to select the BART server name. All the BART servers configured in the PEM console are listed.
- Use the **Server name** field to specify a name for the database server that you want to back up using the BART server. This name gets stored in the BART configuration file.
- Use the **Backup name** field to specify a template for user-defined names to assign to the backups of the database server. If you don't specify a backup name template, then you can reference the backup in BART sub-commands only by using the BART-assigned integer backup identifier.
- Use the **Host address** field to specify the IP address of the database server that you want to configure for backup.
- Use the **Port** field to specify the port to use for the database that you want to back up.
- Use the **User** field to specify the user of the database that you want to back up using BART through the PEM console. If you want to enable incremental backups for this database server, then the user must be a superuser.
- Use the **Password** field to specify the password for the user of the database that you want to back up.

- Use the **Cluster Owner** field to specify the Linux operating system user account that owns the database cluster. This is typically enterprisedb for EDB Postgres Advanced Server database clusters installed in the Oracle databases compatible mode. It is typically postgres for PostgreSQL database clusters and for EDB Postgres Advanced Server database clusters installed in the PostgreSQL databases compatible mode.
- Use the **Archive command** field to specify the desired format of the archive command string to use in the `bart.cfg` file. Inputs provided for the Archive command overwrite the database server's `Postgresql.conf` file. After the server gets added, the database server restarts or database configurations are reloaded.
- Use **Allow incremental backup?** to specify whether to enable incremental backup for this database server.
- Use **Setup passwordless SSH?** to specify if you want to create SSH certificates to allow passwordless logins between the database server and the BART server. Be sure to bind a PEM agent before setting up the passwordless SSH authentication. Passwordless SSH doesn't work for a database server being remotely monitored by a PEM agent.

Use the **Misc** tab under the **BART** tab to describe the miscellaneous properties of the BART Server:

- Use **Override default configuration?** to override the BART server configurations with the specific database server configurations.
- Use the **Xlog** method to specify how to collect the transaction log while pg_basebackup is executing.
- Use the **Retention policy** field to specify the retention policy for the backup. This determines when to mark an active backup as obsolete and hence a candidate for deletion. You can specify the retention policy in terms of number of backups or duration (days, weeks, or months).
- Use **WAL compression** to specify if you want to compress the archived Xlog/WAL files in gzip format. To enable WAL compression, the gzip compression program must be present in the BART user account's PATH. Don't enable the **WAL compression** setting for database servers for which you need to take incremental backups.
- Use the **Copy WALs during restore** field to specify how the archived WAL files are collected when invoking the RESTORE operation. Set to **Enabled** to copy the archived WAL files from the BART backup catalog to the `<restore_path>/archived_wals` directory prior to the database server archive recovery. Set to **Disabled** to retrieve the archived WAL files directly from the BART backup catalog during the database server archive recovery.
- Use the **Thread count** field to specify the number of threads to copy the blocks. You must set **Thread count** to **1** if you want to take a backup using the pg_basebackup utility.
- Use the **Batch size** field to specify the number of blocks of memory used for copying modified blocks. This option applies only to incremental backups.
- Use the **Scan interval** field to specify the number of seconds for the WAL scanner to wait befor scanning the new WAL files.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for MBM files before timing out. This option applies only to incremental backups.

To view the properties of a server, right-click the server name in the PEM client tree, and select **Properties** from the context menu. To modify a server's properties, disconnect from the server before opening the Properties dialog box.

4.2.5 Verifying the connection and binding

Once registered, the new server is added to the PEM browser tree and is displayed on the Global Overview dashboard.

When first connecting to a newly bound server, the Global Overview dashboard might display the new server with a status of "unknown" in the server list. Before recognizing the server, the bound agent must execute a number of probes to examine the server. These probes might take a few minutes to complete, depending on network availability.

In a few minutes, bar graphs on the Global Overview dashboard show that the agent connected successfully. The new server is included in the **Postgres Server Status** list.

If after five minutes the Global Overview dashboard still doesn't list the new server, check the logfiles for the monitoring agent for errors. Right-click the agent's name in the tree, and select the **Dashboards > Probe Log Analysis** from the context menu.

4.3 Registering a PEM agent

Before a PEM agent can be used, you must register it with a PEM server. **PEM agents installed by the PEM server package are registered automatically during server configuration.** For all other agents, you must follow these instructions.

Note: After upgrading the PEM agent, you need to restart it. You don't need to register it again.

4.3.1 How to register PEM agents

On Linux and Windows hosts, the PEM agent package includes a command line utility called pemworker,. You can use it to perform management tasks, including *Registering a PEM agent using the pemworker utility* .

On Windows, the PEM agent graphical installer allows you to register the agent when installing it. This convenience option doesn't support all the possibilities provided by the pemworker utility. If you don't want the installer to register the agent, clear the **Register now** checkbox. For more details, see the installation instructions .

4.3.2 Registering a PEM agent using the pemworker utility

The pemworker utility is installed automatically with the PEM agent. It's located in the `/usr/edb/pem/agent/bin` directory on Linux and `C:\Program Files\edb\pem\agent-x64\bin` on Windows. To register an agent, set the PEM server password, invoke the utility as shown in the examples, and add the relevant options from the table as needed. Follow each option with a corresponding value.

Linux

```
export PEM_SERVER_PASSWORD=edb

#  Running as root

pemworker --register-agent
```

Windows

```
set PEM_SERVER_PASSWORD=edb

#  Running as admin

./pemworker.exe REGISTER
```

Option	Description
--pem-server	The IP address of the PEM backend database server. This parameter is required.
--pem-port	The port of the PEM backend database server. The default value is *5432*.
--pem-user	The name of the database user having superuser privileges of the PEM backend database server. This parameter is required.
--pem-agent-user	The agent user to connect the PEM server backend database server.
--pem-ssl-mode	Type of SSL authentication used for connecting PEM agent to the PEM server. The possible values are *prefer*, *require*,`disable`,`verify-CA`, and *verify-full*.
--cert-path	The complete path to the directory where certificates are created. If you don't provide a path, certificates are created in *~/.pem* on Linux and *%APPDATA%/pem* on Windows.
--config-dir	The directory path for the configuration file. The default is *<pemworker path>/../etc*.
--display-name	A user-friendly name for the agent to display in the PEM browser tree. In PEM 9.6 and later, the default is the host's fully qualified domain name (FQDN), falling back to the hostname if this option isn't set. For releases earlier than PEM 9.6, the default is the hostname.
--force-registration	Include the *force_registration* clause to register the agent with the arguments provided. This clause is useful if you're overriding an existing agent configuration. The default value is *Yes*.
--group	The name of the group in which the agent is displayed.
--team	The name of the database role on the PEM backend database server with access to the monitored database server.
--owner	The name of the database user on the PEM backend database server who owns the agent.
--allow_server_restart	Enable the *allow_server_restart* parameter to allow PEM to restart the monitored server. The default value is *True*.
--allow-batch-probes	Enable the *allow-batch-probes* parameter to allow PEM to run batch probes on this agent. The default value is *False*.
--batch-script-user	The operating system user to use for executing the batch/shell scripts. The default value is none. The scripts don't execute if you leave this parameter blank or the specified user doesn't exist.
--enable-heartbeat-connection	Enable the *enable-heartbeat-connection* parameter to create a dedicated heartbeat connection between the PEM agent and server to update the active status. The default value is *False*.
--enable-smtp	Enable the *enable-smtp parameter* to allow the PEM agent to send the email on behalf of the PEM server. The default value is *False*.
--enable-snmp	Enable the *enable-snmp parameter* to allow the PEM agent to send the SNMP traps on behalf of the PEM server. The default value is *False*.
-o	Specifies whether you want to override the configuration file options.

Authenticating the pemworker utility

Before any changes are made on the PEM database, the connection is authenticated with the PEM database server. When invoking the pemworker utility, you must provide the password associated with the PEM server administrative user role (postgres). You can specify the administrative password in three ways:

- Set the `PEM_SERVER_PASSWORD` environment variable.
- Provide the password on the command line with the `PGPASSWORD` keyword.
- Create an entry in the `.pgpass` file.

If you don't provide the password, a password authentication error occurs. After authentication succeeds, you're prompted for any other missing required information. When the registration is complete, the server confirms that the agent was successfully registered.

4.3.3 Unregistering a PEM agent

You can use the pemworker utility to unregister a PEM agent. To unregister an agent, invoke the pemworker utility as shown in the examples that follow.

Linux

```
#  Running as root

pemworker --unregister-agent
```

Windows

```
./pemworker.exe UNREGISTER-AGENT
```

When invoking the pemworker utility, append command line options to the command string. Follow each option with a corresponding value.

Option	Description
--pem-user <username>	Specifies the name of the database user (member of pem_admin role) of the PEM backend database server. This parameter is required.
--config-dir	Specifies the directory path for the configuration file. The default is *"<pemworker path>/../etc"*.

4.3.4 Advanced usage

The following are some advanced options for PEM agent registration.

Setting the agent ID

Each registered PEM agent must have a unique agent ID. The value `max(id)+1` is assigned to each agent ID unless a value is provided using the `-o` options as shown *Overriding default configurations - examples* .

Overriding default configurations - examples

This example shows how to register the PEM agent overriding the default configurations.

Register the PEM agent using the command line. Assign an `agent_id` value of 8 using the `-o` option.

```
#  Running as root

/usr/edb/pem/agent/bin/pemworker --register-agent \
- -pem-server pemserver \
- -pem-user postgres \
- -pem-port 5432 \
- -display-name agent8 \
- o agent_id=8
__OUTPUT__
Postgres Enterprise Manager Agent registered successfully!
```

Because the `agent_id` of 8 is available, the PEM agent registers successfully. If the given ID is already in use by the existing agent, it throws an error.

Register the PEM agent using the command line. Assign the existing SSL certificates and key files to avoid generating new ones for a particular agent ID. The SSL certificates and key files must be valid for the database user `agent<ID>` , where `<ID>` must be the same as provided using the command line. Use the `-o` option.

```
#  Running as root

#  List the location of valid SSL certificates and key files.

ls -l /root/.pem/agent5.*
- rw------- 1 root root 2192 Nov  7 11:27 /root/.pem/agent5.crt
- rw------- 1 root root 3244 Nov  7 11:27 /root/.pem/agent5.key


#  Register the PEM agent using command line. Assign the

#  SSL certificates and key files using the -o option.

/usr/edb/pem/agent/bin/pemworker --register-agent \
- -pem-server pemserver \
- -pem-user postgres \
- -pem-port 5432 \
- -config-dir /tmp/pem-config \
- -display-name agent5 \
- o agent_id=5 \
- o agent_ssl_crt=/root/.pem/agent5.crt \
- o agent_ssl_key=/root/.pem/agent5.key
__OUTPUT__
Postgres Enterprise Manager Agent registered successfully!
```

Because the valid SSL certificates and key files are available at the given location with proper permissions, the PEM agent registers successfully. If the certificate or key files are not valid or do not have proper permissions it throws an error.

Using a non-root user account to register a PEM agent on Linux

To use a non-root user account to register a PEM agent, you must first install the PEM agent as a root user. After installation, assume the identity of a non-root user, such as edb. Then:

1. Log in as edb. Create `pem` and `logs` directories and assign read, write, and execute permissions:

```
# Running as nonroot user edb
mkdir /home/edb/pem
mkdir /home/edb/pem/logs
chmod 700 /home/edb/pem
chmod 700 /home/edb/pem/logs
```

2. Register the agent with PEM server:

```
export PEM_SERVER_PASSWORD=edb

# Use the following command to create agent certificates and an agent
# configuration file (`agent.cfg`) in the `/home/edb/pem` directory.
/usr/edb/pem/agent/bin/pemworker --register-agent --pem-server <172.19.11.230> --pem-
↪user postgres --pem-port 5432 --display-name non_root_pem_agent --cert-path /home/edb/
↪pem --config-dir /home/edb/pem

# Use the following command to assign read and write permissions to
# these files:
chmod -R 600 /home/edb/pem/agent*
```

3. Change the parameters of the `agent.cfg` file:

```
vi /home/edb/pem/agent.cfg
agent_ssl_key=/home/edb/pem/agent<id>.key
agent_ssl_crt=/home/edb/pem/agent<id>.crt
log_location=/home/edb/pem/worker.log
agent_log_location=/home/edb/pem/agent.log
```

Where `<id>` is the assigned PEM agent ID.

4. Create a `tmp` directory, set the environment variable, and start the agent:

```
mkdir /home/edb/pem/tmp

# Create a script file, add the environment variable, give permissions, and execute:
vi /home/edb/pem/run_pemagent.sh
#!/bin/bash
export TEMP=/home/edb/agent/tmp
/usr/edb/pem/agent/bin/pemagent -c /home/edb/agent/agent.cfg
chmod a+x /home/edb/pem/run_pemagent.sh
cd /home/edb/pem
./run_pemagent.sh
```

Your PEM agent is now registered and started with the edb user. If your machine restarts, then this agent doesn't restart automatically. You need to start it manually using the previous command.

5. Optionally, you can create the service for this PEM agent as the root user to start this agent automatically at machine restart as follows:

a. Update the values for the configuration file path and the user in the `pemagent` service file as superuser:

```
# Running as superuser
sudo vi  /usr/lib/systemd/system/pemagent.service
[Service]
Type=forking
WorkingDirectory=/home/edb/pem
Environment=LD_LIBRARY_PATH=/usr/edb/pem/agent/lib:/usr/libexec/edb-snmp++/lib
Environment=TEMP=/home/edb/pem/tmp
ExecStart=/usr/edb/pem/agent/bin/pemagent -c /home/edb/pem/agent.cfg
```

b. Stop the running agent process, and then restart the agent service:

```
# Find the process id of the running pem agent and pem worker process and kill that␣
↪process
ps -ax | grep pemagent
kill -9 <process_id_of_pemagent>
ps -ax | grep pemworker
kill -9 <process_id_of_pemworker>
# Enable and start pemagent service
sudo systemctl enable pemagent
sudo systemctl start pemagent
sudo systemctl status pemagent
```

6. Check the agent status on the PEM dashboard.

Note:

- Any probes and jobs that require root permission or access to a file owned by another user (for example, enterprisedb) fail. - If you move the *agent.cfg* file from its default location to another, the PEM dashboard might display the agent status as *unknown* . See *Troubleshooting agent issues* , for more information.

4.4 Managing a PEM server

Some of the tasks related to managing the PEM server include:

- Restarting the PEM server and agent
- Controlling the PEM server or PEM agent
- Controlling the HTTPD service on Linux and Windows
- Controlling the HTTPD server
- Managing PEM authentication and security
- Modifying the `pg_hba.conf` file
- Modifying PEM to use a proxy server

4.4.1 Starting and stopping the PEM server and agents

The PEM server starts, stops, and restarts when the Postgres server instance where it resides starts, stops, or restarts. Use the same commands to control the PEM server that you use to control the Postgres server. On Linux platforms, the command that stops and starts the service script varies by platform and OS version.

The PEM agent is controlled by a service named pemagent.

The Windows operating system includes a graphical service controller that displays the server status and offers interactive server control. You can access the Services utility through the Windows Control Panel. When the utility opens, navigate through the listed services and select the service name.

Select **Stop**, **Pause**, **Start**, or **Restart** to control the state of the service.

Any user or client application connected to the Postgres server is abruptly disconnected if you stop the service. For more information about controlling a service, see EDB Postgres Advanced Server .

4.4.2 Controlling the PEM server or PEM agent on Linux

On Linux platforms, the name of the service script that controls:

- A PEM server on EDB Postgres Advanced Server is `edb-as-<x>` or `ppas-<x>` .
- A PEM server on PostgreSQL is `postgresql-<x>` .
- A PEM agent is `pemagent` .

Where `x` indicates the server version number.

You can use the service script to control the service.

To control a service on RHEL 8.x, at the command prompt, assume superuser privileges and issue the command:

```
systemctl <service_name> <action>
```

Where:

`service_name` is the name of the service.

`action` specifies the action taken by the service. Specify:

- `start` to start the service.
- `stop` to stop the service.
- `restart` to stop and then start the service.
- `status` to check the status of the service.

4.4.3 Controlling the PEM server or PEM agent on Windows

The Windows operating system includes a graphical service controller that displays the server status and offers interactive server control. The registered name of the service that controls:

- A PEM server host on PostgreSQL is `postgresql-<x>` .
- A PEM server host on Advanced Server is `edb-as-<x>` or `ppas-<x>` .
- A PEM agent is `Postgres Enterprise Manager - pemAgent` .

Where `x` indicates the server version number.

In the Windows Control Panel, open the Services utility to see the list of services.

Select **Stop the service** to stop a service. Any user or client application connected to the server is abruptly disconnected if you stop the service.

Select **Pause the service** to reload a service's configuration parameters. **Pause the service** is an effective way to reset parameters without disrupting user sessions for many of the configuration parameters.

Select **Start the service** to start a service.

4.4.4 Controlling the HTTPD server

On Linux, you can confirm the status of the PEM-HTTPD service by entering the following command at the command prompt:

```
ps -ef | grep httpd
```

If Linux responds with an answer like the following, httpd isn't running:

```
user 13321 13267 0 07:37 pts/1 00:00:00 grep httpd
```

To start the service on a RHEL 8.x system, use the command:

```
systemctl start httpd
```

On Windows, you can use the Services applet to check the status of the PEM HTTPD service. After opening the Services applet, select the PEM HTTPD service from the list.

The Status column displays the current state of the server. Select **Start** to start PEM HTTPD if the service isn't running.

4.4.5 Managing PEM authentication

Postgres supports a number of authentication methods:

- Secure password (md5)
- GSSAPI
- SSPI
- Kerberos
- Ident
- LDAP
- RADIUS
- Certificate (SSL)
- PAM

Postgres and PEM authentication is controlled by the `pg_hba.conf` configuration file. Entries in the configuration file specify:

- Who can connect to a specific database
- The type of authentication required before that user is allowed to connect

A typical entry in the `pg_hba.conf` file that allows a user named postgres to connect to all databases from the local host (127.0.0.1/32) using secure password (md5) authentication connections takes the form:

```
host all postgres 127.0.0.1/32 md5
```

Depending on your system configuration, you might also need to create a password file for the user account that the PEM agent uses to connect to the server. This file allows the agent to properly respond to the server' s authentication request. An entry in the password file for a user named postgres, with a password of `1safepwd` takes the form:

```
localhost:5432:*:postgres:1safepwd
```

The password file is usually named `~root/.pgpass` on Linux systems or `%APPDATA%\postgresql\pgpass.conf` on Windows. For more information about configuring a password file, see the PostgreSQL website .

For more information about the authentication methods supported by Postgres, see the PostgreSQL core documentation .

4.4.6 Editing the PEM server configuration

You can use the PEM client to manage the configuration parameters of the PEM server to enable features or modify default settings. To open the Server Configuration dialog box, select **Management > Server Configuration**. Then edit the parameter values as needed.

4.4.7 Managing security

PEM provides an interface for managing your Postgres roles and servers.

Login roles

When you connect to the PEM server, you must provide role credentials that allow access to the database on which the PEM server stores data. By default, the postgres superuser account is used to initially connect to the server, but we strongly recommend (for both security and auditing purposes) creating individual roles for each connecting user. You can use the PEM Query tool, the PEM web interface Create – Login/Group Role dialog box, or a command line client (such as psql) to create a role.

To use the Create – Login/Group Role dialog box to create a role:

1. In the PEM tree, expand the node for the server where you want to create the role.
2. Right-click the **Login/Group Roles** node.
3. From the context menu, select **Create > Login/Group Role**.
4. Use the tabs of the Create – Login/Group Role dialog box to define the role.
5. After you finish defining the new role, to create the role, select **Save**.

To modify the properties of an existing login role, right-click the name of a login role in the tree and select **Properties** from the context menu. To delete a login role, right-click the name of the role and select **Delete/Drop** from the context menu.

For more complete information about creating and managing a role, see the PostgreSQL online documentation .

Group roles

Group roles can serve as containers and are used to dispense system privileges, such as creating databases, and object privileges, such as inserting data into a table. The primary purpose of a group role is to make the mass management of system and object permissions easier for a DBA. Rather than assigning or modifying privileges individually across many different login accounts, you can assign or change privileges for a single role and then grant that role to many login roles at once.

Use the **Group Roles** node (located beneath the name of each registered server in the PEM tree) to create and manage group roles. Options on the context menu provide access to a dialog box that allows you to create a new role or modify the properties of an existing role. You can find more information about creating roles in the PostgreSQL documentation .

Using PEM predefined roles to manage access to PEM functionality

You can use the Login/Group Role dialog box to allow a role with limited privileges to access PEM features such as the Audit Manager, Capacity Manager, or SQL Profiler. PEM predefined roles allow access to PEM functionality. Roles that are assigned membership in these roles can access the associated feature.

When defining a user, use the **Membership** tab to specify the roles in which the new user is a member. The new user shares the privileges associated with each role in which it' s a member. For a user to have access to PEM extended functionality, the role must be a member of the pem_user role and the predefined role that grants access to the feature. Use the **Roles** field to select predefined role names from a list.

The **SQL** tab displays the SQL command that the server executes when you select **Save**.

This example creates a login role named acctg_clerk that has access to Audit Manager. The role can make unlimited connections to the server at any given time.

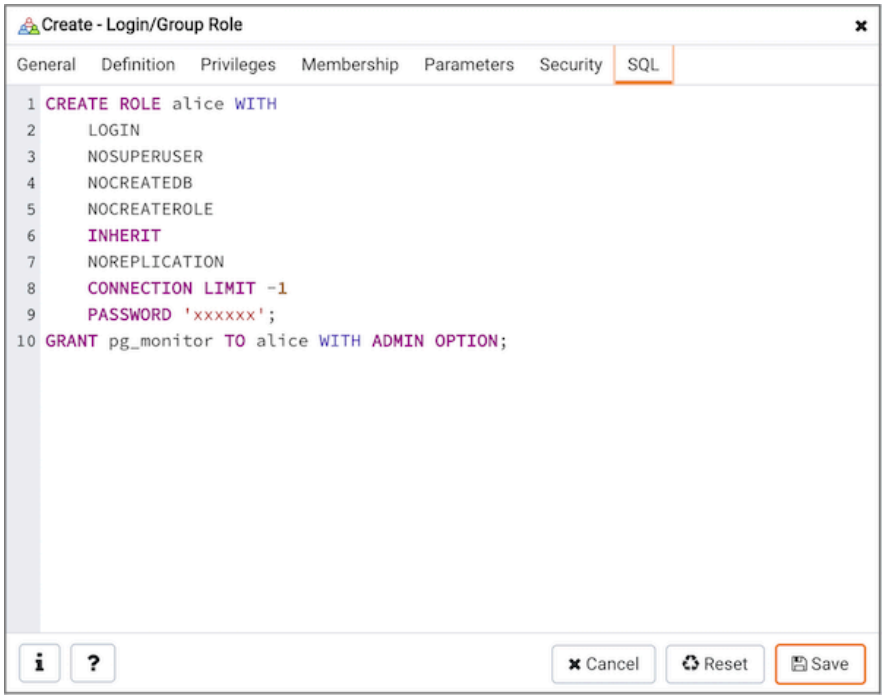


Fig. 1: Create Login Role - SQL tab

You can use PEM predefined roles to allow access to the capabilities listed in the table.

Value	Parent role	Description
pem_super_admin		Role to manage/configure everything on PEM
pem_admin	pem_super_admin	Role for administration/management/configuration of all visible agents/servers and monitored objects
pem_user		Role for having read-only access to all the agents, servers, or monitored objects that are visible to a user having pem_user role. A user with pem_user role can view only those objects where this role has been mentioned in the Team field under the server's properties.
pem_config	pem_admin	Role for configuration management of PEM
pem_component	pem_admin	Role to run/execute all wizard/dialog box-based components
pem_rest_api	pem_admin	Role to access the REST API
pem_server_service_man	pem_admin	Role for allowing to restart/reload the monitored database server (if server-id provided)
pem_manage_schedule_ta	pem_admin	Role to configure the schedule tasks
pem_manage_chart	pem_admin	Role for managing/configuring custom charts.
pem_manage_alert	pem_admin	Role for managing/configuring alerts and its templates
pem_config_alert	pem_config, pem_manage_alert	Role for configuring the alerts on any monitored objects
pem_manage_probe	pem_admin	Role to create, update, and delete the custom probes and change custom probe configuration
pem_config_probe	pem_config, pem_manage_probe	Role for probe configuration (history retention, execution frequency, enable/disable the probe) on all visible monitored objects
pem_database_server_reg	pem_admin	Role to register a database server
pem_comp_postgres_exp	pem_component	Role to run the Postgres Expert
pem_comp_auto_discove	pem_component	Role to run the auto discovery of a database server dialog box
pem_comp_log_analysis_	pem_component	Role to run the Log Analysis Expert
pem_comp_sqlprofiler	pem_component	Role to run the SQL Profiler
pem_manage_efm	pem_admin	Role to manage Failover Manager
pem_comp_capacity_mar	pem_component	Role to run the Capacity Manager
pem_comp_log_manager	pem_component	Role to run the Log Manager
pem_comp_audit_manag	pem_component	Role to run the Audit Manager
pem_comp_tuning_wizar	pem_component	Role to run the Tuning Wizard
pem_comp_bart	pem_component	Role to configure and manage BART server.
pem_comp_performance_	pem_component	Role to run the Performance Diagnostics.

Using a team role

When you register a server for monitoring by PEM, you can specify a team to associate with the server. A team is a group role that you can use to allow or restrict access to one or more monitored servers to a limited group of role members. The PEM client displays a server with a specified team only to those users who are:

- A member of the team role
- The role that created the server
- A role with superuser privileges on the PEM server

To create a team role:

1. In the PEM tree, expand the node for the server where you want to create the role.
2. Right-click the **Login/Group Roles** node.
3. From the context menu, select **Create > Login/Group Role**.
4. In the Create - Login/Group Role dialog box, use the fields to specify the properties of the team role.

Object permissions

A role must be granted sufficient privileges before accessing, executing, or creating any database object. PEM allows you to assign (`GRANT`) and remove (`REVOKE`) object permissions to group roles or login accounts using the PEM client interface.

Object permissions are managed with the graphical object editor for each particular object. For example, to assign privileges to access a database table, right-click the table name in the tree and select **Properties** from the context menu. Use the **Privileges** tab to assign privileges for the table.

The PEM client also contains a Grant wizard (accessed through the **Tools** menu) that allows you to manage many object permissions at once.

4.4.8 Server configuration

You can use the Server Configuration dialog box to modify values of user-configurable parameters that control PEM behavior. To access the Server Configuration dialog box, connect to the PEM server, and select **File > Server Configuration**.

Enter a parameter name in the search box in the upper-right corner of the dialog to locate a specific parameter in the list.

To modify a parameter value, edit the content displayed in the **Value** field to the right of a parameter name. To save your changes, select the **Save** icon in the upper-right corner of the dialog box.

4.4.9 Server configuration parameters - reference

You can use global configuration options to modify aspects of the PEM Server' s behavior. The list of configuration parameters is subject to change.

Parameter name	Value and unit	Description
audit_log_retention_time	30 days	Specifies the number of days for an audit log to be retained on the PEM server.
auto_create_agent_alerts	true	Specifies whether to create default agent level alerts automatically when an agent is registered.
auto_create_server_alerts	true	Specifies whether to create default server level alerts automatically when a server is bound to an agent.
chart_disable_bullets	false	Enables/disables bullets on line charts on dashboards and Capacity Manager reports.
cm_data_points_per_repo	50	Specifies the number of data points to plot on charts on Capacity Manager reports.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
cm_max_end_date_in_ye	5 years	Specifies the maximum amount of time for the Capacity Manager to extrapolate data for. Ensures that threshold-based end dates of reports aren't extrapolated indefinitely.
dash_alerts_timeout	60 seconds	Specifies the number of seconds after which the components of the Alerts dashboard are refreshed.
dash_db_comrol_span	7 days	Specifies the number of days worth of data to plot on the Commit/Rollback Analysis chart on the Database Analysis and Server Analysis dashboards.
dash_db_comrol_timeout	1800 seconds	Specifies the number of seconds after which the Commits/Rollbacks line chart is refreshed on the Database Analysis and Server Analysis dashboards.
dash_db_connovervw_tin	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is refreshed on the Database Analysis dashboard.
dash_db_eventlag_span	7 days	Specifies the number of days worth of data to plot on the Number of Events Lag chart for slony replication on the Database Analysis dashboard.
dash_db_eventlag_timeou	1800 seconds	Specifies the number of seconds after which the Number of Events Lag line chart for slony replication is refreshed on the Database Analysis dashboard.
dash_db_hottable_rows	25 rows	Specifies the number of rows to show on the HOT Table Analysis table on the Database Analysis dashboard.
dash_db_hottable_timeou	300 seconds	Specifies the number of seconds after which the Hot Tables table is refreshed on the Database Analysis dashboard.
dash_db_io_span	7 days	Specifies the number of days worth of data to plot on the Database I/O Analysis chart on the Database Analysis and I/O Analysis dashboards.
dash_db_io_timeout	1800 seconds	Specifies the number of seconds after which the Database I/O line chart is refreshed on the Database Analysis and I/O Analysis dashboards.
dash_db_rowact_span	7 days	Specifies the number of days worth of data to plot on the Row Activity Analysis chart on the Database Analysis, I/O Analysis, and Server Analysis dashboards.
dash_db_rowact_timeout	1800 seconds	Specifies the number of seconds after which the Row Activity line chart is refreshed on the Database Analysis, I/O Analysis, and Server Analysis dashboards.
dash_db_storage_timeout	300 seconds	Specifies the number of seconds after which the Storage bar chart is refreshed on the Database Analysis dashboard.
dash_db_timelag_span	7 days	Specifies the number of days worth of data to plot on the Time Lag chart for Slony replication on the Database Analysis dashboard.
dash_db_timelag_timeout	1800 seconds	Specifies the number of seconds after which the Time Lag line chart for Slony replication is refreshed on the Database Analysis dashboard.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
dash_db_useract_span	7 days	Specifies the number of days worth of data to plot on the User Activity Analysis chart on the Database Analysis dashboard.
dash_db_useract_timeout	1800 seconds	Specifies the number of seconds after which the User Activity line chart is refreshed on the Database Analysis dashboard.
dash_efm_timeout	300 seconds	Specifies the number of seconds after which the Failover Manager Node Status and Failover Manager Cluster Info line chart is refreshed on the Streaming Replication dashboard.
dash_global_overview_ti	30 seconds	Specifies the number of seconds after which the components of the Global Overview dashboard are refreshed.
dash_header_timeout	60 seconds	Specifies the number of seconds after which the information on the header of all the dashboards are refreshed.
dash_io_chkpt_span	7 days	Specifies the number of days worth of data to plot on the Checkpoints chart on the I/O Analysis dashboard.
dash_io_chkpt_timeout	1800 seconds	Specifies the number of seconds after which the Checkpoints line chart is refreshed on the I/O Analysis dashboard.
dash_io_hotindx_timeout	300 seconds	Specifies the number of seconds after which the Hot Indexes bar chart is refreshed on the I/O Analysis dashboard.
dash_io_hottbl_timeout	300 seconds	Specifies the number of seconds after which the Hot Tables bar chart is refreshed on the I/O Analysis dashboard.
dash_io_index_objectio_r	25 rows	Specifies the number of rows displayed on the Index Activity table on the I/O Analysis and Object Activity Analysis dashboards.
dash_io_index_objectio_t	60 seconds	Specifies the number of seconds after which the Index Activity table is refreshed on the I/O Analysis and Object Activity Analysis dashboards.
dash_io_objectio_rows	25 rows	Specifies the number of rows displayed on the Object I/O Details table on the I/O Analysis and Object Activity Analysis dashboards.
dash_io_objectio_timeout	300 seconds	Specifies the number of seconds after which the Object I/O Details table is refreshed on the I/O Analysis and Object Activity Analysis Dashboards.
dash_memory_hostmema	7 days	Specifies the number of days worth of data to plot on the Host Memory Activity Analysis chart on the Memory Analysis dashboard.
dash_memory_hostmema	1800 seconds	Specifies the number of seconds after which the Host Memory Activity line chart is refreshed on the Memory Analysis dashboard.
dash_memory_hostmemc	300 seconds	Specifies the number of seconds after which the Host Memory Configuration pie chart is refreshed on the Memory Analysis and Server Analysis dashboards.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
dash_memory_servmema	7 days	Specifies the number of days worth of data to plot on the server Memory Activity Analysis chart on the Memory Analysis dashboard.
dash_memory_servmema	1800 seconds	Specifies the number of seconds after which the Server Memory Activity line chart is refreshed on the Memory Analysis dashboard.
dash_memory_servmemc	300 seconds	Specifies the number of seconds after which the Server Memory Configuration pie chart is refreshed on the Memory Analysis dashboard.
dash_objectact_objstorag	15 rows	Specifies the number of rows to show on the Object Storage table on the Object Activity Analysis dashboard.
dash_objectact_objstorag	300 seconds	Specifies the number of seconds after which the Object Storage table is refreshed on the Object Activity Analysis dashboard.
dash_objectact_objtopind	300 seconds	Specifies the number of seconds after which the Top 5 Largest Indexes bar chart is refreshed on the Object Activity Analysis dashboard.
dash_objectact_objtoptab	300 seconds	Specifies the number of seconds after which the Top 5 Largest Tables bar chart is refreshed on the Object Activity Analysis dashboard.
dash_os_cpu_span	7 days	Specifies the number of days worth of data to plot on the CPU chart on the Operating System Analysis dashboard.
dash_os_cpu_timeout	1800 seconds	Specifies the number of seconds after which the CPU line chart is refreshed on the Operating System Analysis dashboard.
dash_os_data_span	7 days	Specifies the number of days worth of data to plot on the I/O line chart on the Operating System Analysis dashboard.
dash_os_disk_span	7 days	Specifies the number of days worth of data to plot on the Utilisation chart on the Operating System Analysis dashboard.
dash_os_hostfs_timeout	1800 seconds	Specifies the number of seconds after which the Host File System Details table is refreshed on the Operating System Analysis dashboard.
dash_os_io_timeout	1800 seconds	Specifies the number of seconds after which the I/O line chart is refreshed on the Operating System Analysis dashboard.
dash_os_memory_span	7 days	Specifies the number of days worth of data to plot on the Memory chart on the Operating System Analysis dashboard.
dash_os_memory_timeou	1800 seconds	Specifies the number of seconds after which the Memory line chart is refreshed on the Operating System Analysis dashboard.
dash_os_packet_span	7 days	Specifies the number of days worth of data to plot on the Packet chart on the Operating System Analysis dashboard.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
dash_os_packet_timeout	1800 seconds	Specifies the number of seconds after which the Network Packets line chart is refreshed on the Operating System Analysis dashboard.
dash_os_process_span	7 days	Specifies the number of days worth of data to plot on the Process chart on the Operating System Analysis dashboard.
dash_os_process_timeout	1800 seconds	Specifies the number of seconds after which the Process line chart is refreshed on the Operating System Analysis dashboard.
dash_os_storage_timeout	1800 seconds	Specifies the number of seconds after which the Storage pie chart is refreshed on the Operating System Analysis dashboard.
dash_os_traffic_span	7 days	Specifies the number of days worth of data to plot on the Traffic chart on the Operating System Analysis dashboard.
dash_os_traffic_timeout	1800 seconds	Specifies the number of seconds after which the Traffic line chart is refreshed on the Operating System Analysis dashboard.
dash_os_util_timeout	1800 seconds	Specifies the number of seconds after which the Utilization line chart is refreshed on the Operating System Analysis dashboard.
dash_probe_log_timeout	300 seconds	Specifies the number of seconds after which the Probe Log table refreshed.
dash_replication_archives	7 days	Specifies the number of days worth of data to plot on the WAL Archive Status chart on the Streaming Replication Analysis dashboard.
dash_replication_archives	1800 seconds	Specifies the number of seconds after which the WAL Archive Status line chart is refreshed on the Streaming Replication dashboard.
dash_replication_pagelag	7 days	Specifies the number of days worth of data to plot on the WAL Lag Pages chart on the Streaming Replication dashboard.
dash_replication_pagelag	1800 seconds	Specifies the number of seconds after which the WAL Lag Pages line chart is refreshed on the Streaming Replication dashboard.
dash_replication_segment	7 days	Specifies the number of days worth of data to plot on the WAL Lag Segments chart on the Streaming Replication dashboard.
dash_replication_segment	1800 seconds	Specifies the number of seconds after which the WAL Lag Segments line chart is refreshed on the Streaming Replication dashboard.
dash_replication_timelag	7 days	Specifies the number of days worth of data to plot on the Replication Lag Time chart on the Streaming Replication dashboard.
dash_replication_timelag	1800 seconds	Specifies the number of seconds after which the Replication Lag Time line chart is refreshed on the Streaming Replication dashboard.
dash_server_buffers_writ	168 hours	Specifies the number of days worth of data to plot on the Background Writer Statistics chart on the Server Analysis dashboard.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
dash_server_buffers_writ	300 seconds	Specifies the number of seconds after which the Background Writer Statistics line chart is refreshed on the Server Analysis dashboard.
dash_server_connovervw_	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is refreshed on the Server Analysis dashboard.
dash_server_database_tin	300 seconds	Specifies the number of seconds after which the Databases table is refreshed on the Server Analysis dashboard.
dash_server_dbsize_span	7 days	Specifies the number of days worth of data to plot on the Database Size Analysis on the Server Analysis dashboard.
dash_server_dbsize_time	1800 seconds	Specifies the number of seconds after which the Database Size line chart is refreshed on the Server Analysis dashboard.
dash_server_disk_timeou	1800 seconds	Specifies the number of seconds after which the Disk line chart is refreshed on the Server Analysis dashboard.
dash_server_global_span	7 days	Specifies the number of days worth of data to plot on the Disk line chart on the Server Analysis dashboard.
dash_server_sharedbuff_s	7 days	Specifies the number of days worth of data to plot on the Shared Buffer chart on the Server Analysis dashboard.
dash_server_sharedbuff_t	1800 seconds	Specifies the number of seconds after which the Shared Buffers line chart is refreshed on the Server Analysis dashboard.
dash_server_tabspacesize	7 days	Specifies the number of days worth of data to plot on the Tablespace Size chart on the Server Analysis dashboard.
dash_server_tabspacesize	1800 seconds	Specifies the number of seconds after which the Tablespace Size line chart is refreshed on the Server Analysis dashboard.
dash_server_useract_spar	7 days	Specifies the number of days worth of data to plot on the User Activity chart on the Server Analysis dashboard.
dash_server_useract_time	1800 seconds	Specifies the number of seconds after which the User Activity line chart is refreshed on the Server Analysis dashboard.
dash_sess_waits_timewai	300 seconds	Specifies the number of seconds after which the Session Waits By Time Waited pie chart is refreshed on the Session Waits Analysis dashboard.
dash_sess_waits_waitdtl_	300 seconds	Specifies the number of seconds after which the Session Waits Details table is refreshed on the Session Waits Analysis dashboard.
dash_sessact_lockact_tim	300 seconds	Specifies the number of seconds after which the Session Lock Activity table is refreshed on the Session Activity Analysis dashboard.
dash_sessact_workload_ti	300 seconds	Specifies the number of seconds after which the Session Workload table is refreshed on the Session Activity Analysis dashboard.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
dash_storage_dbdtls_time	300 seconds	Specifies the number of seconds after which the Database Details table is refreshed on the Storage Analysis dashboard.
dash_storage_dboverviw_t	300 seconds	Specifies the number of seconds after which the Database Overview pie chart is refreshed on the Storage Analysis dashboard.
dash_storage_hostdtls_tin	300 seconds	Specifies the number of seconds after which the Host Details table is refreshed.
dash_storage_hostovervw	300 seconds	Specifies the number of seconds after which the Host Overview pie chart is refreshed on the Storage Analysis dashboard.
dash_storage_tblspcdtls_t	300 seconds	Specifies the number of seconds after which the Tablespace Details table is refreshed on the Storage Analysis dashboard.
dash_storage_tblspcoverv	300 seconds	Specifies the number of seconds after which the Tablespace Overview pie chart is refreshed on the Storage Analysis dashboard.
dash_sys_waits_nowaits_	300 seconds	Specifies the number of seconds after which the System Waits By Number Of Waits pie chart is refreshed on the System Waits Analysis dashboard.
dash_sys_waits_timewait_	300 seconds	Specifies the number of seconds after which the System Waits By Time Waited pie chart is refreshed on the System Waits Analysis dashboard.
dash_sys_waits_waitdtl_t	300 seconds	Specifies the number of seconds after which the System Waits Details table is refreshed on the System Waits Analysis dashboard.
deleted_charts_retention_	7 days	Specifies the number of days that a custom chart (displayed on a user-defined dashboard) is stored.
deleted_objects_data_rete	3 days	Specifies the number of days after which the probe data of any deleted object, for example, server or agents or Barman server, is deleted from the pemhistory and pemdata schema. It deletes the job in case you don't want to delete the data of those obsolete objects.
deleted_probes_retention_	7 days	Specifies the number of days that a custom probe (displayed on a user-defined dashboard) is stored.
download_chart_format	jpeg	Specifies the format in which a downloaded chart is stored. Can be jpeg or png.
flapping_detection_state_cha	3	Specifies the number of state changes detected within a specified interval to define a given alert as flapping. Flapping starts when more than *N* state changes have occurred over [*N* + 1 *(min(probe_interval) * 2)] minutes and the fine state is not None. The default value of *N* is 2 or 3, and min(probe_interval) is the smallest interval for all the probes used by the alert. Flapping ends when ZERO state changes have occurred over [2 *N* * min(probe_interval)] minutes.
job_retention_time	30 days	Specifies the number of days that nonrecurring scheduled tasks and their associated jobs are retained.
long_running_transaction	5 minutes	Specifies the number of minutes a query executes before being considered long running.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
nagios_cmd_file_name	<file_name>	Specifies nagios command file to which passive service check results are sent.
nagios_enabled	t	Specifies whether alert notification are submitted to nagios.
nagios_medium_alert_as_cr	f	Specifies whether a medium level PEM alert is considered critical in nagios.
nagios_spool_retention_tim	7 days	Specifies the number of days to retain nagios messages in the spool table before they are discarded.
probe_log_retention_time	30 days	Specifies the number of days that probe log records are retained.
reminder_notification_inter	24 hours	Specifies the number of hours after which a reminder email is sent in case an alert wasn't cleared.
server_log_retention_time	30 days	Specifies the number of days that the server log is retained on the PEM server.
show_data_tab_on_graph	false	If *true*, a **Data** tab is added to each graph. Select the **Data** tab to review the data that's plotted on the graph.
smtp_authentication	false	Specifies whether to enable/disable authentication over SMTP.
smtp_enabled	true	Specifies whether to enable/disable sending of emails.
smtp_encryption	false	Specifies whether to send SMTP email using an encrypted connection.
smtp_password		Specifies the password to use to connect to the SMTP server.
smtp_port	25	Specifies the SMTP server port to use for sending email.
smtp_server	127.0.0.1	Specifies the SMTP server host address to use for sending email.
smtp_spool_retention_tin	7 days	Specifies the number of days to retain sent email messages in the spool table before they are discarded.
smtp_username		Specifies the username to used to connect to an SMTP server.
snmp_authentication_pas		Specifies the authentication password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
snmp_authentication_pro	NONE	Specifies the authentication type for SNMP traps. Its possible values are NONE, HMACMD5, and HMACSHA. Used only with SNMPv3.
snmp_community	public	Specifies the SNMP community used when sending traps. Used only with SNMPv1 and SNMPv2.
snmp_context_engine_id		Specifies the context engine id, the identifier for MIB objects when sending SNMP traps. If not specified, snmp_security_engine_id is used. Used only with SNMPv3.
snmp_context_name		Specifies the context name, the identifier for MIB objects when sending SNMP traps. Used only with SNMPv3.
snmp_enabled	true	Specifies whether to enable/disable sending SNMP traps.
snmp_port	162	Specifies the SNMP server port to use for sending SNMP traps.

continues on next page

Table 7 – continued from previous page

Parameter name	Value and unit	Description
snmp_privacy_password		Specifies the privacy password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
snmp_privacy_protocol	NONE	Specifies the privacy protocol for SNMP traps. Its possible values are NONE, DES, AES128, IDEA, AES192, or AES256. Used only with SNMPv3.
snmp_security_engine_id		Specifies the engine id of the SNMP agent on the SNMP server. Used only with SNMPv3.
snmp_security_level	NOAUTH_NOPRIV	Specifies security level. Its possible values are: - AUTH_NOPRIV - Authentication, No Privacy. Then NONE option is possible for privacy protocol only. AUTH_PRIV - Authentication, Privacy. Then NONE option selection shouldn't be allowed for authentication and privacy protocol. NOAUTH_NOPRIV - no Authentication, no Privacy. Then only NONE option is possible for authentication and privacy protocol. Used only with SNMPv3.
snmp_security_name		Specifies the user name or security name for sending SNMP traps. Used only with SNMPv3.
snmp_server	127.0.0.1	Specifies the SNMP server host address to use for sending SNMP traps.
snmp_spool_retention_tir	7 days	Specifies the number of days to retain sent traps in the spool table before they're discarded.
webclient_help_pg	EDB hosted documentation	Specifies the location of the online PostgreSQL core documentation.

4.5 Managing Postgres servers

4.5.1 Starting and stopping monitored database servers

PEM lets you start up and shut down managed server instances with the PEM client. To configure a server to allow PEM to manage the service, complete the Server Registration dialog box. Use the dialog box to register the database server with a PEM agent, which is the local agent installed on the same host as the database server. You can also:

- Specify the **Store on PEM Server** option on the Properties dialog box.
- Specify the name of a service script in the **Service ID** field on the **Advanced** tab:
- For EDB Postgres Advanced Server, the service name is `edb-as-<x>` or `ppas-<x>` .
- For PostgreSQL, the service name is `postgresql-<x>` .

Where `<x>` indicates the server version number.

After connecting to the server, you can start or stop the server by selecting the server name in the tree and selecting **Tools > Queue Server Startup** or **Tools > Queue Server Shutdown**.

4.5.2 Modifying the pg_hba.conf file

Entries in the `pg_hba.conf` file control network authentication and authorization. The `pg_hba.conf` file on the PEM server host must allow connections between the PEM server and PEM-HTTPD, the PEM agent, and the monitored servers.

The PEM server installation process prompts you for the IP address and connection information for hosts for PEM to monitored. This information is added to the top of the `pg_hba.conf` file of the PEM backing database.

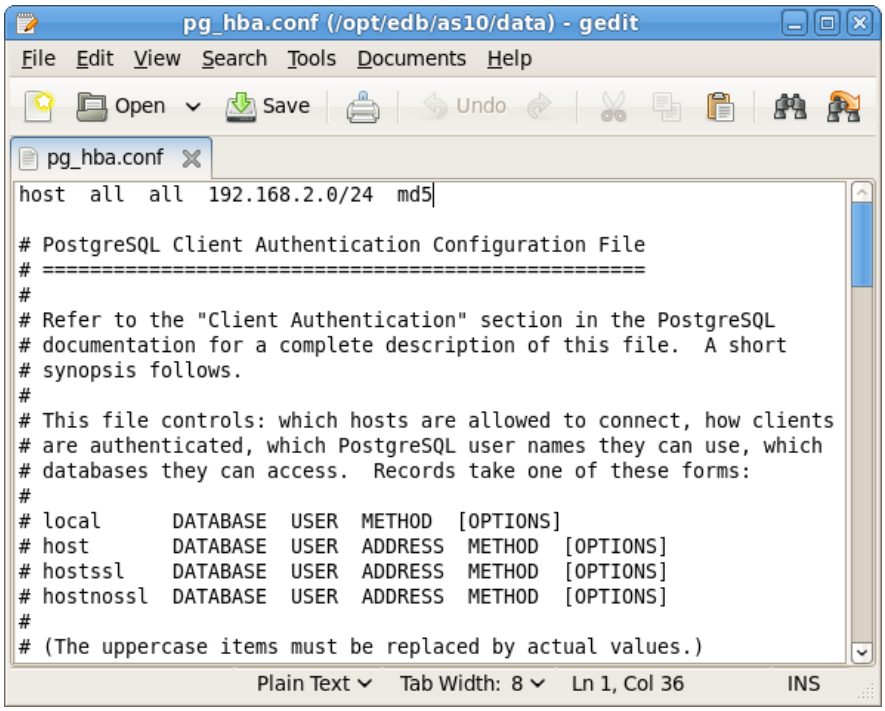


Fig. 2: PEM entries in the pg_hba.conf file

You might also need to manually modify the `pg_hba.conf` file to allow connections between the PEM server and other components. For example, if your PEM-HTTPD installation isn't on the same host as the PEM server, you must modify the `pg_hba.conf` file on the PEM server host to allow PEM-HTTPD to connect to the server.

By default, the `pg_hba.conf` file resides in the data directory, under your Postgres installation. For example, on an EDB Postgres Advanced Server 10 host, the default location of `pg_hba.conf` is:

`/var/lib/edb/as10/data/pg_hba.conf`

You can modify the `pg_hba.conf` file with your editor of choice. After modifying the file, restart the server for changes to take effect.

The following example shows a `pg_hba.conf` entry that allows an md5 password authenticated connection from a user named postgres to the `postgres` database on the host where `pg_hba.conf` resides. The connection is coming from an IP address of 192.168.10.102:

```
#  TYPE     DATABASE        USER        CIDR-ADDRESS          METHOD

#  IPv4 local connections:

 host     postgres       postgres    192.168.10.102/32       md5
```

You can specify the address of a network host or a network address range. For example, if you want to allow connections from servers with the addresses 192.168.10.23, 192.168.10.76, and 192.168.10.184, enter a CIDR-ADDRESS of `192.168.10.0/24` to allow connections from all of the hosts in that network:

```
#  TYPE     DATABASE        USER        CIDR-ADDRESS       METHOD

#  IPv4 local connections:

  host      postgres         all       192.168.10.0/24       md5
```

For more information about formatting a `pg_hba.conf` file entry, see the PostgreSQL core documentation .

Before you can connect to a Postgres server with PEM, you must ensure that the `pg_hba.conf` file on both servers allows the connection. If you receive an error when connecting to the database server, modify the `pg_hba.conf` file, adding an entry that allows the connection.

4.5.3 Creating and maintaining databases and objects

Each instance of a Postgres server manages one or more databases. Each user must provide authentication information to connect to the database before accessing the information it contains. The PEM client lets you create and manage databases and the objects that comprise a database, such as tables, indexes, and stored procedures.

To create a database in PEM, right-click any managed server' s **Databases** node and select **Create > Database**. After defining a database, you can create objects in the new database.

For example, to create a table, right-click a **Tables** node and select **Create > Table**. Specify the attributes of the table in the New Table dialog box.

PEM provides similar dialog boxes for creating and managing other database objects such as:

- Tables
- Indexes
- Stored procedures
- Functions
- Triggers
- Views
- Constraints

Each object type is listed in the tree. Right-click the node that corresponds to an object type to access the **Create** menu and create an object. To perform administrative tasks for the selected object, select **Properties** from the context menu of a named node.

4.5.4 Template Linux service script

A service script for the database server allows the PEM server to start, stop, or restart the database server. Doing so might be necessary when performing configuration management, certificate management, and other administrative tasks. Service scripts are platform specific.

The Postgres server on which the PEM server resides must contain a service script. Postgres installers in Windows generated by EDB create a service script for you. If you' re using a Postgres server from another source like native packages, you must provide a service script.

Note: On Rocky Linux or RHEL 8.x, the service script resides in */usr/lib/systemd/system* .

For information about customizing a Postgres service, see the PostgreSQL documentation .

4.6 Managing a PEM agent

Use the Postgres Enterprise Manager browser-based console to manage PEM agents.

The PEM agent is responsible for executing tasks and reporting statistics from the agent host and the monitored Postgres instances to the PEM server. A single PEM agent can monitor multiple installed instances of Postgres that reside on the same host where the agent is installed or on multiple remote hosts. It' s also responsible for executing other tasks that the user might schedule, such as server shutdowns, SQL Profiler traces, and custom jobs.

You can perform the following tasks to manage the PEM agents:

- *Managing job notifications*
- *Managing scheduled jobs*
- *Modifying agent configuration*
- *Reviewing or modifying agent properties*
- *Setting agent privileges*

4.6.1 Managing job notifications

In the PEM console, you can configure the settings for sending the SMTP trap on success or failure of a system-generated job listed under scheduled tasks or a custom agent job. You can configure these email notification settings at the following three levels to send email notifications to the specified user group. These levels are shown in order of precedence.

- Job level
- Agent level
- PEM server level (default)

Configuring job notifications at job level

You can configure email notification settings at the job level only for a custom agent job in one of the following ways:

- For a new agent job, you can configure the email notification settings in the **Notification** tab of the Create Agent Job wizard while creating the job.
- For an existing custom job, you can edit the properties of the job and configure the notification settings.

Use the **Notifications** tab to configure the email notification settings at the job level:

- Use the **Send the notifications** field to specify when you want to send the email notifications.
- Use the **Email group** field to specify the email group to send the email notification to.

Configuring job notifications at agent level

Select the agent in the tree view, right-click, and select **Properties**. In the Properties dialog box, select the **Job Notifications** tab.

Use the Job notifications tab to configure the email notification settings at the agent level:

- Use the **Override default configuration?** switch to specify if you want the agent level job notification settings to override the default job notification settings. Select **Yes** to enable the rest of the settings on this dialog box to define when and to whom to send the job notifications.
- Use the **Email on job completion?** switch to specify whether to send the job notification when the job completes successfully.
- Use the **Email on a job failure?** switch to specify whether to send the job notification when the job fails.
- Use the **Email group** field to specify the email group to send the job notification to.

Configuring job notifications at server level

You can use the Server Configuration dialog box to provide information about your email notification configuration at the PEM server level. To open the Server Configuration dialog box, select **Management > Server Configuration**.

Four server configuration parameters specify information about your job notification preferences at PEM server level:

- Use the **job_failure_notification** switch to specify if you want to send an email notification after each job failure.
- Use the **job_notification_email_group** parameter to specify the email group to send the email notification to.
- Use the **job_retention_time parameter** to specify the number of days to retain nonrecurring scheduled tasks in the system.
- Use the **job_status_change_notification** switch to specify if you want to send an email notification after each job status change, that is, failure, success, or interrupted.

4.6.2 Managing scheduled jobs

You can create a PEM scheduled job to perform a set of steps you define in a specified sequence. These steps can contain SQL code or a batch/shell script that you can run on a server that' s bound with the agent. You can schedule these jobs to suit your business requirements. For example, you can create a job for taking a backup of a particular database server and schedule it to run on a specific date and time of every month.

To create or manage a PEM scheduled job, use the PEM tree to browse to the PEM agent for which you want to create the job. The tree displays a **Jobs** node, under which currently defined jobs are listed. To add a job, right-click the **Jobs** node and select **Create Job** from the context menu.

Use the tabs on the Create - Agent Job dialog box to define the steps and schedule that make up a PEM scheduled job, as described in the following sections.

Provide general information about the job

Use the **General** tab to provide general information about a job:

- Provide a name for the job in the **Name** field.
- Set the **Enabled** switch to **Yes** to enable a job. Set it to **No** to disable a job.
- Use the **Comment** field to store notes about the job.

Define the job steps

Use the **Steps** tab to define and manage the steps that the job performs. Select **Add** (+) to add a step. Then, select the compose icon, located at the left side of the header, to open the Step Definition dialog box.

Use the Step Definition dialog box to define the step:

- Provide a name for the step in the **Name** field. Steps are performed in alphanumeric order by name.
- Use the **Enabled** switch to include the step when executing the job (**True**) or to disable the step (**False**).
- Use the **Kind** switch to indicate if the job step invokes SQL code (**SQL**) or a batch script (**Batch**).
- If you select **SQL**, use the **Code** tab to provide SQL code for the step.
- If you select **Batch**, use the **Code** tab to provide the batch script to execute during the step.
- Use the **On error** list to specify the behavior of pgAgent if it encounters an error while executing the step. Select from:
- **Fail** — Stop the job if you encounter an error while processing this step.
- **Success** — Mark the step as completing successfully and continue.
- **Ignore** — Ignore the error and continue.
- If you selected SQL as your input for the **Kind** switch:
- Use the **Server** field to specify the server that' s bound with the agent for which you are creating the PEM scheduled job.
- Use the **Database** field to specify the database that' s associated with the server that you selected.
- Use the **Comment** field to provide a comment about the step.
- Use the context-sensitive field on the Step Definition dialog box **Code** tab to provide the SQL code or batch script to execute during the step:
- If the step invokes SQL code, provide one or more SQL statements in the **SQL query** field.
- If the step invokes a batch script, provide the script in the **Code** field. If you' re running on a Windows server, use standard batch file syntax. On a Linux server, you can use any shell script, provided that you specify a suitable interpreter on the first line (such as `#!/bin/sh`). Along with the defined inline code, you can also provide the path of any batch script, shell script, or SQL file on the filesystem.

```
-    To invoke a script on a Linux system, you must modify the entry for the `batch_
↪script_user`  parameter in the `agent.cfg`  file and specify the user who runs the␣
↪script. You can specify either a nonroot user or root for this parameter. If you don't␣
↪specify a user or the specified user doesn't exist, then the script doesn't execute.␣
↪Restart the agent after modifying the file.

-    To invoke a script on a Windows system, set the registry entry for␣
```

(continues on next page)

(continued from previous page)

```
↪`AllowBatchJobSteps`  to `true`  and restart the PEM agent. PEM registry entries are␣
↪located in `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent` .
```

After providing all the information required by the step, select **Save**.

Define the job schedule

Select **Add** (+) to add each step, or select the **Schedules** tab to define the job schedule.

On the **Schedules** tab, select **Add** (+) to add a schedule for the job. Then select the compose icon located at the left side of the header to open the Schedule Definition dialog box.

Use the **Schedules definition** tab to specify the days and times for the job to execute.

- Provide a name for the schedule in the **Name** field.
- Use the **Enabled** switch to indicate for pgAgent to use the schedule (**Yes**) or to disable the schedule (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.
- Use the **Comment** field to provide a comment about the schedule.

Select the **Repeat** tab to define the days when the schedule executes in a cron-style format. The job executes on each date or time element selected on the **Repeat** tab.

In each field, select a value to add it to the list of selected values for the field. To clear the values from a field, select the X located at the right-side of the field.

- Use the fields in the **Days** box to specify the days when the job executes:
- Use the **Week Days** field to select the days when the job executes.
- Use the **Month Days** field to select the numeric days when the job executes. Select **Last Day** to perform the job on the last day of the month, regardless of the date.
- Use the **Months** field to select the months when the job executes.
- Use the fields in the **Times** box to specify the times when the job executes in hours and minutes.

Select the **Exceptions** tab to specify any days when you don' t want the schedule to execute. For example, you might not want to run jobs on holidays.

Select **Add** (+) to add a row to the exception table. Then:

- In the **Date** column, open a calendar selector and select a date when you don' t want the job to execute. Specify ****<Any>**** to indicate that you don' t want the job to execute on any day at the time selected.
- In the **Time** column, open a time selector and specify a time when you don' t want the job to execute. Specify ****<Any>**** to indicate that you don' t want the job to execute at any time on the day selected.

Configure the email notifications

Select the **Notifications** tab to configure the email notification settings on job level:

- Use the **Send the notifications** field to specify when you want to send the email notifications.
- Use the **Email group** field to specify the email group to send the email notification to.

When you finish defining the schedule, you can use the **SQL** tab to review the code that creates or modifies your job.

Select **Save** to save the job definition.

After you save a job, the job is listed under the **Jobs** node of the PEM tree of the server on which it was defined. The **Properties** tab in the PEM console displays a high-level overview of the selected job, and the **Statistics** tab shows the details of each run of the job. To modify an existing job or to review detailed information about a job, right-click a job name, and select **Properties** from the context menu.

4.6.3 Modifying agent configuration

A number of configurable parameters control the behavior of the PEM agent. On Linux, these are stored in a configuration file. On Windows they' re stored as registry keys.

Most agent configuration is managed automatically. We recommend against manually modifying any of these configuration parameters unless specifically directed to do so.

Note: If you add, remove, or modify any of the parameters, you must restart the agent to apply the changes.

Contents of the configuration file

On Linux systems, PEM configuration options are stored in the `agent.cfg` file, located in `/usr/edb/pem/agent/etc` . The `agent.cfg` file contains the entries shown in the following table.

Parameter name	Description	Default value
pem_host	The IP address or hostname of the PEM server.	127.0.0.1.
pem_port	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.
pem_agent	A unique identifier assigned to the PEM agent.	The first agent is '1', the second agent is '2', and so on.
agent_ssl_key	The complete path to the PEM agent's key file.	/root/.pem/agent.key
agent_ssl_crt	The complete path to the PEM agent's certificate file.	/root/.pem/agent.crt
agent_flag_dir	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default.
log_level	Specifies the type of event to write to the PEM log files, one of *debug2*, *debug*, *info*, *warning*, *error*. These are in descending order of logging verbosity; *debug2* logs everything possible, and *error* only logs errors.	warning
log_location	Specifies the location of the PEM worker log file.	127.0.0.1.
agent_log_location	Specifies the location of the PEM agent log file.	/var/log/pem/agent.log

continues on next page

Table 8 – continued from previous page

Parameter name	Description	Default value
long_wait	The maximum length of time (in seconds) for the PEM agent to wait before attempting to connect to the PEM server if an initial connection attempt fails.	30 seconds
short_wait	The minimum length of time (in seconds) for the PEM agent to wait before checking which probes are next in the queue waiting to run.	10 seconds
alert_threads	The number of alert threads to be spawned by the agent. For more information, see *About alert threads* .	Set to 1 for the agent that resides on the host of the PEM server, 0 for all other agents.
enable_smtp	When set to true, this agent will attempt to send email notifications as configured in the PEM web application.	true for PEM server host, false for all others.
enable_snmp	When set to true, this agent will attempt to send SNMP notifications as configured in the PEM web application.	true for PEM server host, false for all others.
enable_nagios	When set to true, Nagios alerting is enabled.	true for PEM server host, false for all others.
enable_webhook	When set to true, Webhook alerting is enabled.	true for PEM server host, false for all others.
max_webhook_retries	Used to set the maximum number of times pemAgent retries to call webhooks on failure.	Default 3.
connect_timeout	The maximum time in seconds (a decimal integer string) for the agent to wait for a connection.	Not set by default. Set to 0 to indicate for the agent to wait indefinitely.
allow_server_restart	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features might be enabled/disabled, depending on the value of this parameter.	False
max_connections	The maximum number of probe connections used by the connection throttler.	0 (an unlimited number)
connection_lifetime	Used ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection to a different database is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
allow_batch_probes	If set to TRUE, the user can create batch probes using the custom probes feature.	false
heartbeat_connection	When set to TRUE, a dedicated connection is used for sending the heartbeats. If set to TRUE, the *max_connections* parameter must be set to greater than 1.	false
batch_script_dir	Provides the path where script file (for alerting) is stored.	/tmp
connection_custom_setup	Used to provide SQL code to invoke when a new connection with a monitored server is made.	Not set by default.
ca_file	The path to a CA certificate to use instead of the platform default for verifying webhook server certificates. You can override this value with the --*webhook_ssl_ca_crt* option when defining webhooks.	Not set by default.

continues on next page

Table 8 – continued from previous page

Parameter name	Description	Default value
batch_script_user	The name of the user to use for executing the batch/shell scripts.	None
Webhook parameters	You can specify the following options multiple times. Each time, precede the option with a header of the form *[WEBHOOK/<name>]*, where *<name>* is the name of a previously created webhook. These settings are automatically added when webhooks are created. We don't recommend adding them manually.	N/A
webhook_ssl_key	The complete path to the webhook's SSL client key file.	N/A
webhook_ssl_crt	The complete path to the webhook's SSL client certificate file.	N/A
webhook_ssl_crl	The complete path of the CRL file to validate webhook server certificate.	N/A
webhook_ssl_ca_crt	The complete path to the webhook's SSL ca certificate file.	N/A
allow_insecure_webhooks	When set to true, allow webhooks to call with insecure flag.	false

Contents of the registry

On Windows systems, PEM registry entries are located in:

`HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent`

The registry contains the entries shown in the following table.

Parameter name	Description	Default value
PEM_HOST	The IP address or hostname of the PEM server.	127.0.0.1.
PEM_PORT	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.
AgentID	A unique identifier assigned to the PEM agent.	The first agent is '1', the second agent is '2', and so on.
AgentKeyPath	The complete path to the PEM agent's key file.	%APPDATA%\Roaming\pem\agent.key.
AgentCrtPath	The complete path to the PEM agent's certificate file.	%APPDATA%\Roaming\pem\agent.crt
AgentFlagDir	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default.
LogLevel	Specifies the type of event to write to the PEM log files. These are in descending order of logging verbosity; *debug2* logs everything possible, and *error* only logs errors.	warning
LongWait	The maximum length of time (in seconds) that the PEM agent waits before attempting to connect to the PEM server if an initial connection attempt fails.	30 seconds

continues on next page

Table 9 – continued from previous page

Parameter name	Description	Default value
shortWait	The minimum length of time in seconds that the PEM agent waits before checking which probes are next in the queue (waiting to run).	10 seconds
AlertThreads	The number of alert threads for the agent to spawn. For more information, see *About alert threads* .	Set to 1 for the agent that resides on the host of the PEM server, 0 for all other agents.
EnableSMTP	When set to true, the SMTP email feature is enabled.	true for PEM server host, false for all others.
EnableSNMP	When set to true, the SNMP trap feature is enabled.	true for PEM server host, false for all others.
EnableWebhook	When set to true, Webhook alerting is enabled.	true for PEM server host, false for all others.
MaxWebhookRetries	Sets the maximum number of times for pemAgent to retry to call webhooks on failure.	Default 3.
ConnectTimeout	The maximum time in seconds (a decimal integer string) that the agent waits for a connection.	Not set by default. If set to 0, the agent waits indefinitely.
AllowServerRestart	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features might be enabled/disabled, depending on the value of this parameter.	true
MaxConnections	The maximum number of probe connections used by the connection throttler.	0 (an unlimited number)
ConnectionLifetime	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection to a different database is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
AllowBatchProbes	If set to TRUE, the user can create batch probes using the custom probes feature.	false
HeartbeatConnection	When set to TRUE, a dedicated connection is used for sending the heartbeats.	false
BatchScriptDir	The path to store the script file for alerting.	/tmp
ConnectionCustomSetup	Used to provide SQL code to invoke when a new connection with a monitored server is made.	Not set by default.
AllowBatchJobSteps	If set to true, the batch/shell scripts are executed. Scripts are executed by the user account under which the PEM agent is running.	None
Webhook parameters	You can specify the following options multiple times. They must be placed in a subkey *SoftwareEnterpriseDBPEMagentWEBHOOK<name>*, where *<name>* is the name of a previously created webhook. These settings are automatically added when webhooks are created. We don't recommended adding them manually.	
WebhookSSLKey	The complete path to the webhook's SSL client key file.	
WebhookSSLCrt	The complete path to the webhook's SSL client certificate file.	

continues on next page

Table 9 – continued from previous page

Parameter name	Description	Default value
WebhookSSLCrl	The complete path of the CRL file to validate webhook server certificate.	
WebhookSSLCaCrt	The complete path to the webhook's SSL ca certificate file.	
AllowInsecureWebhooks	When set to true, allow webhooks to call with insecure flag.	false

About alert threads

The number of alert threads spawned by an agent is determined by the `alert_threads` or `AlertThreads` parameter. In general, we recommend setting this parameter to 1 on the agent that resides on the PEM server and 0 for all other agents. However, on PEM server instances with very large numbers of alerts (caused by many monitored servers, many enabled alerts, or high alert frequency), it may be necessary to increase this parameter if alerts are not being evaluated at the configured frequency. In this situation, we recommend setting this parameter to around 8 on the agent that resides on the PEM server and 0 for all other agents and tuning up or down accordingly.

When tuning this parameter, note that any agent can process any alert, so in general it is unnecessary to have a non-zero number of alert threads on more than one agent. The capacity of the PEM instance to process alerts is determined by the total number of alert thread across all agents. Increasing the number of threads on a specific agent does not give any additional performance for alerts pertaining to servers monitored by that agent.

Each alert thread opens a connection to the PEM server backend, so allocating more threads than necessary does result in additional memory and CPU usage on the PEM server.

4.6.4 Reviewing or modifying agent properties

The PEM Agent Properties dialog box provides information about the PEM agent from which the dialog box was opened. To open the dialog box, right-click an agent name in the PEM client tree and select **Properties** from the context menu.

Use the PEM Agent Properties dialog box to review or modify information about the PEM agent:

- The **Description** field displays a modifiable description of the PEM agent. This description is displayed in the tree of the PEM client.
- You can use groups to organize your servers and agents in the PEM client tree. Use the **Group** list to select the group in which the agent is displayed.
- Use the **Team** field to specify the name of the group role that can access servers monitored by the agent. The servers monitored by this agent are visible in the PEM client tree to connected team members. This is a convenience feature. The **Team** field doesn't provide true isolation. Don't use it for security purposes.
- The **Heartbeat interval** fields display the length of time that elapses between reports from the PEM agent to the PEM server. Use the selectors next to the **Minutes** or **Seconds** fields to modify the interval.

Use the **Job Notifications** tab to configure the email notification settings on agent level:

```
-   Use the **Override default configuration?** switch to specify if you want the agent␣
↪level job notification settings to override the default job notification settings.
 Select **Yes** to enable the rest of the settings on this dialog box to define when and␣
↪to whom to send the job notifications.  -   Use the **Email on job completion?**␣
↪switch to specify whether to send the job notification when the job completes␣
↪successfully. -   Use the **Email on a job failure?** switch to specify whether to␣
```

(continues on next page)

(continued from previous page)

```
↪send the job notification when the job fails. -   Use the **Email group** field to␣
↪specify the email group to send the job notification to.
```

The **Agent Configurations** tab displays all the current configurations and capabilities of an agent.

```
-   The **Parameter** column displays a list of parameters. -   The **Value** column␣
↪displays the current value of the corresponding parameter. -   The **Category** column␣
↪displays the category of the corresponding parameter. It can be either␣
↪**configuration** or **capability**.
```

4.6.5 Setting agent privileges

By default, the PEM agent is installed with root privileges for the operating system host and superuser privileges for the database server. These privileges allow the PEM agent to invoke unrestricted probes on the monitored host and database server about system usage, retrieving and returning the information to the PEM server.

Root user versus non-root user

PEM functionality lessens as the privileges of the PEM agent decrease. For complete functionality, run the PEM agent as root. If the PEM agent runs under the database server' s service account, PEM probes don' t have complete access to the statistical information used to generate reports, and functionality is limited to the capabilities of that account. If the PEM agent runs under another lesser-privileged account, functionality is limited even further.

Feature name	Works with root user	Works with non-root user	Works with remote PEM agent
Audit Manager	yes	The Audit Log Manager might not be able to apply requested modifications if the service can't be restarted. The user running the PEM agent might be different from the user who owns the data directory of the database server. Thus the user running the PEM agent might not be able to change the configuration and also might not be able to restart the services of the database server.	no
Capacity Manager	yes	yes	yes Note: There's no correlation between the database server and operating system metrics

continues on next page

Table 10 – continued from previous page

Feature name	Works with root user	Works with non-root user	Works with remote PEM agent
Log Manager	yes	The Log Manager might not be able to apply requested modifications if the service can't restart. The user running the PEM agent might be different from the user who owns the data directory of the database server. Thus the user running the PEM agent might not be able to change the configuration and also might not be able to restart the services of the database server.	no
Manage Alerts	yes	yes	yes Note: When Run Alert Script on the database server is selected, it runs on the machine where the bound PEM agent is running and not on the actual database server machine.
Manage Charts	yes	yes	yes
Manage Dashboards	yes	Some dashboards might not be able to show complete data. For example, columns such as swap usage, CPU usage, IO read, and IO write appear as 0 in the session activity dashboard.	Some dashboards might not be able to show complete data. For example, the operating system information of the database server doesn't appear as not available.
Manage Probes	yes	Some of the PEM probes don't return information, and some of functionality might be affected.	Some of the PEM probes don't return information, and some of the functionality might be affected.
Postgres Expert	yes	The Postgres Expert can access the configuration expert and schema expert but not the security expert.	The Expert provides partial information, as operating system information isn't available.
Postgres Log Analysis Expert	yes	The Postgres Log Analysis Expert might not be able to do the analysis, as it depends on the logs imported by Log Manager, which doesn't work as required.	The Postgres Log Analysis Expert can't do the analysis, as it depends on the logs imported by Log Manager, which doesn't work as required.
Scheduled Tasks	yes	For Linux, if the user is the same as batch_script_user in *agent.cfg*, then the shell script runs.	Scheduled tasks work only for the database server. Scripts run on a remote agent.

continues on next page

Table 10 – continued from previous page

Feature name	Works with root user	Works with non-root user	Works with remote PEM agent
Tuning Wizard	yes	The Tuning Wizard can't run if the service can't restart. The user running the PEM agent might be different from the user who owns the data directory of the database server. Thus the user running the PEM agent might not be able to change the configuration and also might not be able to restart the services of the database server.	no
System Reports	yes	yes	yes
Core Usage Reports	yes	yes	The Core usage report doesn't show complete information. For example, the platform, number of cores, and total RAM aren't displayed.
Managing BART	yes	BART and the BART scanner might not be able to start or reload.	no Note: BART requires passwordless authentication between two machines where database server and BART are installed.

Functionality affected by limiting operating system privileges

If you limit the operating system privileges of the PEM agent, some of the PEM probes don't return information, and the following functionality might be affected.

Note: The list isn't comprehensive but provides an overview of the type of functionality that's limited.

Probe or action	Operating system	PEM functionality affected
Data And Logfile Analysis	Linux/ Windows	The Postgres Expert can't access complete information.
Session Information	Linux	The per-process statistics are incomplete.
PG HBA	Linux/ Windows	The Postgres Expert can't access complete information.
Service restart functionality	Linux/ Windows	The Audit Log Manager, Server Log Manager Log Analysis Expert, and PEM might not be able to apply requested modifications.
Package Deployment	Linux/ Windows	PEM can't run downloaded installation modules.
Batch Task	Windows	PEM can't run scheduled batch jobs in Windows.
Collect data from server (root access required)	Linux/ Windows	Columns such as swap usage, CPU usage, IO read, IO write appear as 0 in the session activity dashboard.

Functionality affected by limiting database privileges

If you restrict the database privileges of the PEM agent, the following PEM functionality might be affected:

Probe	Operating system	PEM functionality affected
Audit Log Collection	Linux/Windows	PEM receives empty data from the PEM database.
Server Log Collection	Linux/Windows	PEM can't collect server log information.
Database Statistics	Linux/Windows	The Database/Server Analysis dashboards contain incomplete information.
Session Waits/System Waits	Linux/Windows	The Session/System Waits dashboards contain incomplete information.
Locks Information	Linux/Windows	The Database/Server Analysis dashboards contain incomplete information.
Streaming Replication	Linux/Windows	The Streaming Replication dashboard doesn't display information.
Slony Replication	Linux/Windows	Slony-related charts on the Database Analysis dashboard don't display information.
Tablespace Size	Linux/Windows	The Server Analysis dashboard doesn't display complete information.
xDB Replication	Linux/Windows	PEM can't send xDB alerts and traps.

Error handling

If the probe is querying the operating system without enough privileges, the probe might return a `permission denied` error. If the probe is querying the database without enough privileges, the probe might return a `permission denied` error or display the returned data in a PEM chart or graph as an empty value.

When a probe fails, an entry is written to the log file that contains the name of the probe, the reason the probe failed, and a hint that helps you resolve the problem.

You can view probe-related errors that occurred on the server in the Probe Log dashboard or review error messages in the PEM worker log files. On Linux, the default location of the log file is:

```
/var/log/pem/worker.log
```

On Windows, log information is available on the Event Viewer.

4.7 Managing SSL certificates

PEM uses self-signed SSL certificates:

- To secure requests to the *Post-configuration steps when web server and PEM backend database are installed separately* (Apache httpd), which provides the user interface and REST API.
- To secure and authenticate the *PEM backend database server and agent connection certificates* .

4.7.1 Web-server certificates

PEM uses self-signed SSL certificates for the Apache httpd server. The self-signed SSL certificate and key files for the Apache httpd server are generated during PEM installation.

To use your own SSL certificate for PEM, update the Apache HTTP configuration file `edb-ssl-pem.conf` .

Update these two SSL directives in the PEM VirtualHost section:

- `SSLCertificateFile` is your certificate file, for example, `your_domain_name.crt` .
- `SSLCertificateKeyFile` is the `.key` file generated when you created the certificate signing request (CSR), for example, `your_private.key` .

For example, make the following updates:

```
#  Change the server name and file names in the configuration file to

#  match your certificate files.

SSLEngine on
SSLCertificateFile /path/to/your_domain_name.crt
SSLCertificateKeyFile /path/to/your_private.key
```

To increase security, you can replace the httpd self-signed SSL certificates with trusted CA signed certificates in PEM. For more information, see Replacing httpd self-signed SSL certificates .

4.7.2 PEM backend database server and agent connection certificates

By default, PEM implements secured SSL/TLS connections between PEM agents and the backend database. It also acts as its own certificate authority (CA) to generate certificates and keys for the PEM server and agent. The self-signed SSL certificates and keys for the PEM server and agent are generated during PEM installation. These certificates and keys encrypt the connection from agent to server. In addition, PEM agents are authenticated using their certificate rather than a password.

You can replace the PEM self-signed SSL certificates and keys with the trusted CA certificates and keys. For more information, see *Use certificates and keys signed by trusted CA* .

How PEM self-signed SSL certificates work

The PEM server configuration script generates self-signed SSL certificate and key files for the PEM backend database server. The backend database server uses these certificates and keys to authenticate and encrypt the agent connections. Each certificate has an expiry date. Regenerate the certificates when they near expiration. For more information, see *Regenerating the server self-signed SSL certificates* .

The PEM agent connects to the PEM backend database server using the libpq interface, acting as a client of the backend database server. The agent self-signed SSL certificates and keys get generated during *Registering a PEM agent* . PEM agent establishes the connection with the PEM backend database server using the self-signed SSL certificate and key files.

Each agent has a unique identifier, and the agent certificates and keys have the corresponding identifier. Each certificate has an expiry date. Regenerate the certificates when they near expiration. For more information, see *Regenerating the agent self-signed SSL certificates* .

If required, you can use the same certificate for all agents rather than one certificate per agent. For more information, see *Generate a common agent certificate and key pair* .

For more information on using the SSL certificates to connect in Postgres, see Securing TCP/IP connections with SSL .

Certificate and key files generation

The PEM server generates the certificates and key files in the data directory of the backend database server:

- `ca_certificate.crt`
- `ca_key.key`
- `root.crt`
- `root.crl`
- `server.crt`
- `server.key`

The `ca_certificate.crt` and `ca_key.key` files are used during the agent registration process to generate the agent's SSL certificates and key files.

The `root.crt` file is a copy of the `ca_certificate.crt` file. You use the root certificate for the backend database server by setting the `ssl_ca_file` parameter as `root.crt` in the `postgresql.conf` file.

The `root.crl` has the certificate revocation list (CRL) of digital certificates revoked by the issuing CA before their actual or assigned expiration date.

The `server.crt` file is the signed certificate for the PEM server, and the `server.key` file is the private key to the certificate. The PEM agent certificates are generated using these server certificate and key files.

PEM self-signed SSL certificate renewal

The PEM agent installed with the PEM server monitors the expiration date of the `ca_certificate.crt` file. When the certificate is about to expire, PEM:

- Makes a backup of the existing certificate files
- Creates new certificate files and appends the new CA certificate file to the `root.crt` file on the PEM server
- Creates a job to renew the certificate file of any active agents
- Restarts the PEM server

4.7.3 Generate a common agent certificate and key pair

By creating and using a single Postgres user for all PEM agents rather than one user per agent (the default), you can use the same certificate for all agents.

Create a user, generate an agent certificate and key pair, and use them for all PEM agents.

1. Create one common agent user in the PEM backend database. Grant the `pem_agent` role to the user.

```
# Running as enterprisedb
psql -p 5444 -U enterprisedb -d pem
CREATE USER pem_agent_common_user;
GRANT pem_agent TO pem_agent_common_user;
```

1. Generate an agent key:

```
# Running as root
openssl genrsa -out agent.key 4096
```

1. Generate a CSR for the agent:

```
openssl req -new -key agent.key -out agent.csr -subj /C=IN/ST=MH/L=Pune/O=PEM/CN=<agent_
↪user>
```

Where `-subj` is provided as per your requirements.

1. Use the `openssl x509` command to sign the CSR and generate an agent certificate:

```
openssl x509 -req -days 365 -in agent.csr -CA ca_certificate.crt -CAkey ca_key.key -
↪CAcreateserial -out agent.crt
```

1. Change the permissions on the `agent.crt` and `agent.key` file:

```
chmod 600 agent.crt agent.key
```

1. Use this agent certificate and key pair:
   - For registering the new PEM agent from the remote host to the PEM server.

a. Copy the agent certificate and key pair to the remote agent host and register the agent:

```
export PEM_SERVER_PASSWORD=edb

/usr/edb/pem/agent/bin/pemworker --register-agent \
--pem-server 192.168.99.130 \
--pem-user enterprisedb \
--pem-port 5444 \
--pem-agent-user pem_agent_common_user \
-o agent_ssl_crt= agent.crt \
-o agent_ssl_key= agent.key
```

b. Enable and start the pemagent services:

```
systemctl enable pemagent
systemctl start pemagent
```

- To replace the agent certificate and key pair with the registered agent. a. Edit the `agent_user` , `agent_ssl_key` , and `agent_ssl_crt` parameters in `agent.cfg` file of the agent host:

```
vi /usr/edb/pem/agent/etc/agent.cfg
# Edit the agent username
agent_user=pem_agent_common_user
# Edit the ssl parameters with new certificate and key file location
agent_ssl_key=<new_location>/agent.key
agent_ssl_crt=<new_location>/agent.crt
```

b. Restart the pemagent service:

```
systemctl restart pemagent
```

4.7.4 Use certificates and keys signed by trusted CA

Replace the PEM self-signed SSL certificates and keys with certificates and keys signed by a trusted CA.

After obtaining the trusted CA certificates and keys, replace the *Replace the self-signed server SSL certificates with the certificates signed by the trusted CA* and *Replace the self-signed agent SSL certificates with the certificates signed by the trusted CA* certificates and keys.

Replace the self-signed server SSL certificates with the certificates signed by the trusted CA

1. Back up the old server certificate and key files:

```
# Running as root
mkdir /var/lib/edb/as<x>/data/certs
cd /var/lib/edb/as<x>/data/
mv server.* root.* ca_* /var/lib/edb/as<x>/data/certs
```

1. Generate a private key for the server:

```
openssl genrsa -out server.key 4096
```

1. Generate a CSR for the server:

```
openssl req -new -key server.key -out server.csr -subj /C=IN/ST=MH/L=Pune/O=EDB/CN=PEM
```

Where `-subj` is provided as per your requirements. We recommend using the hostname or domain qualified full name of the PEM server host for `CN` .

1. Obtain the CA certificate (`trusted_ca.crt`) from a trusted CA.
2. Ask your CA to sign the CSR and generate the server certificate for you.
3. Verify the details of the new server certificate aren't tampered with and match your provided details:

```
openssl x509 -noout -text -in server.crt
```

1. Use the new certificate obtained from the CA as the `root.crt` file:

```
cp trusted_ca.crt root.crt
```

1. If the trusted CA doesn't provide CRL, disable CRL usage by the server. To disable the CRL usage, comment the `ssl_crl_file` parameter in the `postgresql.conf` file.

Note: If you accidentally leave a CRL from a previous CA in place and do not comment out *ssl_crl_file* , the server will start but authentication will fail with an SSL error message *tlsv1 alert unknown ca* . The error doesn't specify that the CRL is the cause, so this can be difficult to debug if encountered out of context.

1. Copy the new `root.crt` , `server.key` , and `server.crt` files to the data directory of the backend database server:

```
cp root.crt server.key server.crt /var/lib/edb/as<x>/data
```

1. Change the owner and permissions of the new certificates and key files to be the same as the data directory:

```
cd /var/lib/edb/as<x>/data/
chown enterprisedb server.* root.crt ca_certificate.crt
chmod 600 server.* root.crt ca_certificate.crt
```

Note: Don't restart the PEM server now. If you restart the PEM server, all the registered agents will stop working.

1. Replace each PEM agent SSL certificates with the trusted CA certificates. For more information, see these *Replace the self-signed agent SSL certificates with the certificates signed by the trusted CA* .
2. Restart the PEM server.
   - On Linux:

```
# Running as root
systemctl start edb-as-<x>
```

Restarting the backend database server restarts the PEM server.

Replace the self-signed agent SSL certificates with the certificates signed by the trusted CA

Replace the self-signed agent SSL certificates only after replacing the self-signed server certificates `server.crt` and `server.key` and CA certificate `root.crt` .

1. Use psql to find all the agent identifiers (IDs) needed to replace the SSL certificates:

```
psql -U enterprisedb -d pem --no-psqlrc -t -A -c "SELECT id FROM pem.agent WHERE␣
↪active=true"
```

1. After identifying the agents that need key files, generate an `agent<ID>.key` for each agent:

```
openssl genrsa -out agent<ID>.key 4096
```

Where `<ID>` is the agent identifier.

1. Generate a CSR for each agent:

```
openssl req -new -key agent<ID>.key -out agent<ID>.csr -subj /C=IN/ST=MH/L=Pune/O=PEM/
↪CN=agent<ID>
```

Where `-subj` is provided as per your requirements. Replace `<ID>` in `CN` with an appropriate agent identifier.

Note: If you prefer to use a single certificate for all PEM agents rather than one per agent, create a common Postgres user and supply this username in place of *ID* . See *Generate a common agent certificate and key pair* .

1. Ask your CA to sign the CSR and generate the agent certificate for you.
2. Copy the certificate and key files on the respective hosts, where `<ID>` matches the `agent_id` in the `/usr/edb/pem/agent/etc/agent.cfg` file.
3. Change the ownership and permission on the new `agent<ID>.crt` and `agent<ID>.key` file:

```
chown root agent<ID>.crt agent<ID>.key
chmod 600 agent<ID>.crt agent<ID>.key
```

1. Back up the old agent certificate and key file:

```
# Running as root
mkdir root/.pem/certs
mv root/.pem/agent<ID>.* root/.pem/certs
```

1. Replace each agent's certificate and key file with the newly generated files:

```
cp agent<ID>.key agent<ID>.crt root/.pem
```

7. Restart the PEM agent service.
   - On Linux:

```
# Running as root
systemctl restart pemagent
```

- On Windows: Use the Services applet to restart the PEM agent. The PEM agent service is named Postgres Enterprise Manager Agent. Select the service name in the Services dialog box, and select **Restart the service**.

4.7.5 Testing certificates

If you experience authentication problems, you can use these tests to validate certificates.

Validate a certificate against the root certificate

To check whether a PEM agent certificate is trusted according to the server's `root.crt`, copy both certificates to the same machine. Then execute the following command:

```
openssl verify -verbose -CAfile root.crt  agent1.crt
```

This command returns `agent1.crt: OK` on success or an explanatory message on failure.

Make a test connection to the PEM backend database

To verify whether the agent user can connect using a certificate, on the server where the agent is located, execute the following commands as root:

```
PGHOST=<pem_host>
PGPORT=<pem_db_port>
PGUSER=agent<ID>
PGSSLCERT=/root/.pem/agent<ID>.crt
PGSSLKEY=/root/.pem/agent<ID>.key
PGSSLMODE=require

export PGHOST PGPORT PGUSER PGSSLCERT PGSSLKEY PGSSLMODE

<psql_path> -A -t -c "SELECT version()"
```

Where:

- `<psql_path>` is the full path to the psql executable, for example `/usr/edb/as15/bin/psql`.
- `<pem_host>` is the hostname or IP address of PEM server.

- <pem_db_port> is the PEM backend database server port.
- <ID> is the ID of the agent you' re testing, as defined in the file /usr/edb/pem/agent/etc/agent.cfg .

Note: If you used the instructions in *Generate a common agent certificate and key pair*

you must set PGUSER to the common agent username.

If the connection succeeds, it returns the Postgres version of the database server. Success means that your certificate is valid and the Postgres user is correctly configured.

Regenerating the server self-signed SSL certificates

If the PEM backend database server certificates are near expiring, plan to regenerate the certificates and key files.

To replace the self-signed SSL certificates:

1. Stop all running PEM agents, first on the server host and then on any monitored host. - On Linux:

```
# Running as root
systemctl stop pemagent
```

- On Windows: Use the Services applet to stop the PEM agent. The PEM agent service is named Postgres Enterprise Manager Agent. In the Services dialog box, select the service name, and select **Stop the service**.

1. Back up the existing SSL certificates and keys:

```
cd /var/lib/edb/as<x>/data
mkdir certs
mv server.* root.* ca_* certs/
```

1. Use the openssl command to generate the ca_key.key file:

```
openssl genrsa -out ca_key.key 4096
```

1. Move the ca_key.key file to the data directory of the backend server, and change the permissions:

```
mv ca_key.key /var/lib/edb/as<x>/data
chmod 600 /var/lib/edb/as<x>/data/ca_key.key
```

1. Use ca_key.key to generate the ca_certificate.crt file:

```
openssl req -x509 -nodes -days 3650 -newkey rsa:4096 -keyout ca_key.key -out ca_
↪certificate.crt
```

1. Change the permissions of the ca_certificate.crt file:

```
chmod 600 /var/lib/edb/as<x>/data/ca_certificate.crt
```

1. Reuse the ca_certificate.crt file as the root.crt file:

```
cp /var/lib/edb/as<x>/data/ca_certificate.crt /var/lib/edb/as<x>/data/root.crt
```

1. Change the owner and permissions on the root.crt file:

```
chown enterprisedb /var/lib/edb/as<x>/data/root.crt
chmod 600 /var/lib/edb/as<x>/data/root.crt
```

1. Use the `openssl_rsa_generate_crl()` function to create the certificate revocation list `root.crl` :

```
psql -U enterprisedb -d pem --no-psqlrc -t -A -c
"SELECT openssl_rsa_generate_crl(/var/lib/edb/as<x>/data/ca_certificate.crt, /var/lib/
↪edb/as<x>/data/ca_key.key)" > /var/lib/edb/as<x>/data/root.crl
```

1. Change the ownership and permissions of the `root.crl` file:

```
chown enterprisedb /var/lib/edb/as<x>/data/root.crl
chmod 600 /var/lib/edb/as<x>/data/root.crl
```

1. Use the `openssl` command to generate the `server.key` file:

```
openssl genrsa -out server.key 4096
```

1. Move the `server.key` to the data directory of the backend server, and change the ownership and permissions:

```
mv server.key /var/lib/edb/as<x>/data
chown enterprisedb /var/lib/edb/as<x>/data/server.key
chmod 600 /var/lib/edb/as<x>/data/server.key
```

1. Use the `openssl req` command to create the CSR:

```
openssl req -new -key server.key -out server.csr -subj /C=IN/ST=MH/L=Pune/O=EDB/CN=PEM
```

Where `-subj` is provided as per your requirements. You define `CN` asthe hostname/domain name of the PEM server host.

1. Use the `openssl x509` command to sign the CSR and generate a server certificate. Move the `server.crt` to the data directory of the backend database server:

```
openssl x509 -req -days 365 -in server.csr -CA ca_certificate.crt -CAkey ca_key.key -
↪CAcreateserial -out server.crt
mv server.crt /var/lib/edb/as<x>/data
```

Where `-req` indicates the input is a CSR. The `-CA` and `-CAkey` options specify the root certificate and private key to use for signing the CSR.

1. Change the owner and the permissions on the `server.crt` file:

```
chown enterprisedb /var/lib/edb/as<x>/data/server.crt
chmod 600 /var/lib/edb/as<x>/data/server.crt
```

1. Restart the PEM server:

```
systemctl restart edb-as-<x>
```

Restarting the backend database server restarts the PEM server.

1. Regenerate each PEM agent' s self-signed SSL certificates. For more information, see *Regenerating the agent self-signed SSL certificates* .

Regenerating the agent self-signed SSL certificates

You need to regenerate the agent certificates and key files:

- If the PEM server certificates are regenerated
- If the PEM agent certificates are near expiring

You must regenerate a certificate and a key for each agent interacting with the PEM server and copy it to the agent.

Each agent has a unique identifier that's stored in the pem.agent table of the pem database. You must replace the certificate and key files with the certificate or key files that corresponds to the agent's identifier.

Prerequisites:

- PEM server has self-signed certificates.
- `ca_certificate.crt` and `ca_key.key` are in the data directory of the PEM backend database server.
- `ca_certificate.crt` is the same as `root.crt` .
- `ca_certificate.crt` and `ca_key.key` are valid SSL certificates and keys.

To generate a PEM agent certificate and key file pair:

1. Use psql to find the number of agents and their corresponding identifiers:

```
# Running as enterprisedb
psql -p 5444 -U enterprisedb -d pem --no-psqlrc -t -A -c "SELECT id FROM pem.agent WHERE␣
↪active=true"
```

1. Stop all the running PEM agents:

```
# Running as root
systemctl stop pemagent
```

On Windows, use the Services applet to stop the PEM agent. The PEM agent service is named Postgres Enterprise Manager Agent. In the Services dialog box, select the service name, and select **Stop the service**.

1. After identifying the agents that need key files, generate an `agent.key` for each agent:

```
openssl genrsa -out agent<ID>.key 4096
```

Where `ID` is the agent identifier.

1. Generate a certificate signing request (CSR) for each agent:

```
openssl req -new -key agent<ID>.key -out agent<ID>.csr -subj /C=IN/ST=MH/L=Pune/O=PEM/
↪CN=agent<ID>
```

Where `CN` is the `agent<ID>` .

1. Use the `openssl x509` command to sign the CSR and generate an agent certificate:

```
openssl x509 -req -days 365 -in agent<ID>.csr -CA ca_certificate.crt -CAkey ca_key.key -
↪CAcreateserial -out agent<ID>.crt
```

Where `-req` indicates the input is a CSR. The `-CA` and `-CAkey` options specify the root certificate and private key to use for signing the CSR.

Before generating the next certificate and key file pair, move the `agent.key` and `agent.crt` files generated in the steps 2 and 4 on their respective PEM agent host.

1. Change the permission on the new `agent<ID>.crt` and `agent<ID>.key` file:

```
chmod 600 agent<ID>.crt agent<ID>.key
```

1. Back up the old agent certificate and key files:

```
mkdir root/.pem/certs
mv root/.pem/agent<ID>.* root/.pem/certs
```

1. Replace each agent's certificate and key file with the newly generated files:

```
cp agent<ID>.key agent<ID>.crt root/.pem
```

1. Start the PEM agent service.
   - On Linux:

```
# Running as root
systemctl start pemagent
```

```
- On Windows: Use the Services applet to start the PEM agent. The PEM agent service is␣
↪named Postgres Enterprise Manager Agent. In the Services dialog box, select the␣
↪service name, and select **Start the service**.
```

4.8 Managing configuration settings

Postgres Enterprise Manager reads multiple configuration files at startup:

- `config.py` — This is the main configuration file. Don't modify it. You can use it as a reference for configuration settings that one of the following files might be override.
- `config_distro.py` — This file is read after `config.py` and is intended for packagers to change any settings that are required for their Postgres Enterprise Manager distribution. This typically includes certain paths and file locations. This file is optional. Packagers can create it in the same directory as `config.py` if it's needed.
- `config_local.py` — This file is read after `config_distro.py` and is intended for end users to change any default or packaging specific settings that they want to adjust to meet local preferences or standards. This file is optional. Create it in the same directory as `config.py` if it's needed.

A copy of the default `config.py` file is included in the PEM online help for reference.

4.9 Troubleshooting agent issues

You can troubleshoot these issues that can occur with the PEM agent.

4.9.1 Restoring a deleted PEM agent

If an agent was deleted from the `pem.agent` table then you can't restore it. You need to use the pemworker utility to reregister the agent.

If an agent was deleted from the PEM web client but still has an entry in the `pem.agent` table with value of `active = f`, then you can restore the agent:

1. Check the values of the `id` and `active` fields:

```
pem=# SELECT * FROM pem.agent;
```

2. Update the status for the agent to `true` in the `pem.agent` table:

```
pem=# UPDATE pem.agent SET active=true WHERE id=<x>;
```

Where `x` is the identifier that was displayed in the output of the query used in step 1.

3. Refresh the PEM web client.

The deleted agent is restored. However, the servers that were bound to that agent might appear to be down. To resolve this issue, modify the PEM agent properties of the server to add the bound agent again. After the successful modification, the servers appear as running properly.

4.9.2 Updating configuration file path (agent status displaying as unknown)

If you move the agent configuration file (`agent.cfg`) from its default location `/usr/edb/pem/agent/etc` to another location, the PEM dashboard might display the agent status as "unknown". In that case, you need to update the value of the agent configuration file path in the `pemagent` service file.

1. Use the following command to modify the `pemagent` service file as a superuser:

```
# Running as superuser
sudo vi  /usr/lib/systemd/system/pemagent.service
```

2. Update the `agent.cfg` file path. For example, on a Redhat host, update the file path:

```
ExecStart=/usr/edb/pem/agent/bin/pemagent -c /usr/edb/pem/agent/etc/agent.cfg
```

3. Reload `systemd`, to update the modified service script:

```
sudo systemctl daemon-reload
```

4. Restart the PEM agent:

```
sudo systemctl restart pemagent
```

4.9.3 Using the command line to delete a PEM agent with down or unknown status

Using the PEM web interface to delete PEM agents with Down or Unknown status can be difficult if the number of such agents is large. In this situation, you can use the command line interface to delete Down or Unknown agents.

Use the following query to delete the agents that are Down for more than *N* number of hours:

```
UPDATE pem.agent SET active=false WHERE id IN
(SELECT a.id FROM pem.agent
a JOIN pem.agent_heartbeat b ON (b.agent_id=a.id)
WHERE a.id IN
(SELECT agent_id FROM pem.agent_heartbeat WHERE (EXTRACT (HOUR FROM now())-
EXTRACT (HOUR FROM last_heartbeat)) > <N> ));
```

Use the following query to delete the agents with an Unknown status:

```
UPDATE pem.agent SET active=false WHERE id IN
(SELECT id FROM pem.agent WHERE id NOT IN
(SELECT agent_id FROM pem.agent_heartbeat));
```

CHAPTER 5

Using

5.1 Accessing the web interface

After installing a PEM server and agent, you can configure PEM to start monitoring and managing PostgreSQL or EDB Postgres Advanced Server instances. The PEM server installer installs the PEM web interface. You can use the interface to review information about objects that reside on monitored servers or to review statistical information gathered by the PEM server.

After the server installation completes, you can open the PEM interface in your choice of browser by navigating to:

`https://<ip_address_of_PEM_host>:8443/pem`

`ip_address_of_PEM_host` specifies the IP address of the host of the PEM server.

Use the fields on the Postgres Enterprise Manager Login window to authenticate yourself with the PEM server:

- Provide the name of a pem database user in the **Username** field. For the first user connecting, this is the name provided when installing the PEM server.
- Provide the password associated with the user in the **Password** field.

After providing your credentials, select **Login** to connect to PEM.

The PEM web interface opens, displaying the Global Overview dashboard.

The Browser pane displays a tree that provides access to information about the database objects that reside on each server. The tree expands to display a hierarchical view of the servers and objects that the PEM server monitors.

Before you can use the PEM web interface to manage or monitor a database server, you must register that server with the PEM server. When you register a server, you describe the connection to the server, provide authentication information for the connection, and specify any management preferences (optionally binding an agent).

A server can be managed or unmanaged:

- A *managed* server is bound to a PEM agent. The PEM agent monitors the server to which it's bound and performs tasks or reports statistics for display on the PEM dashboards. A managed server has access to extended PEM functionality such as custom alerting. When registering a server, you can also allow PEM to restart a managed server as required.

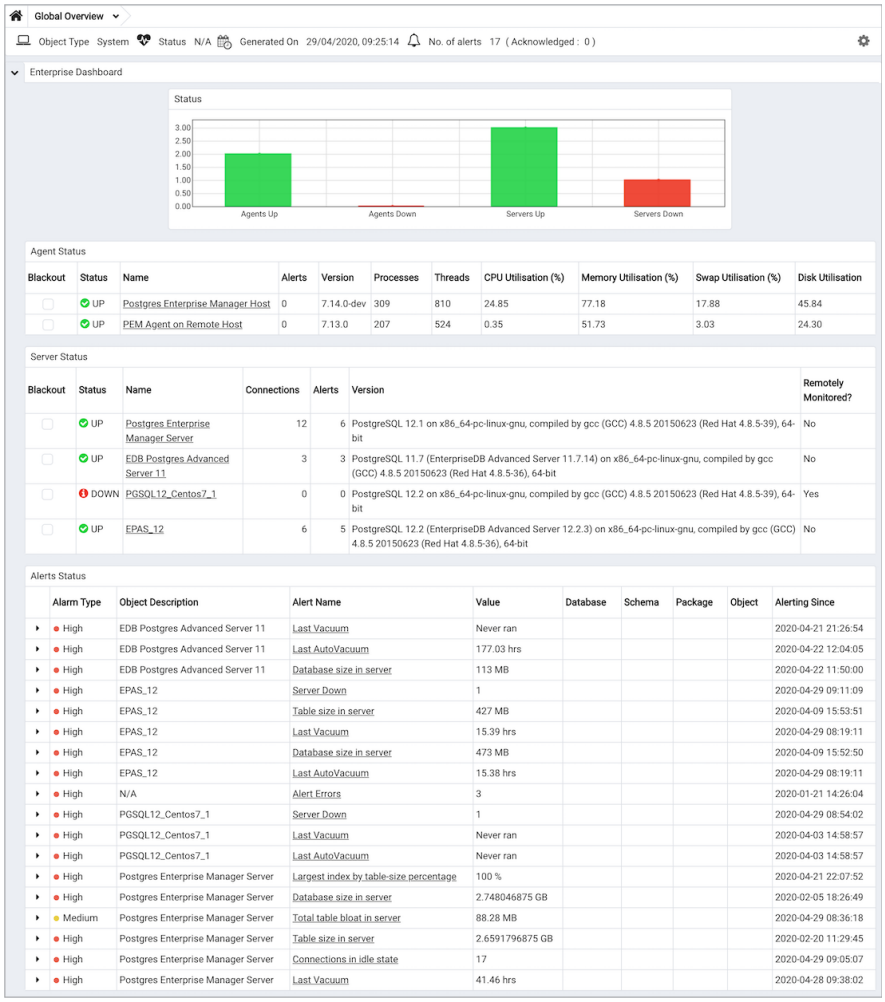


Fig. 1: The Global Overview Dashboard

- An *unmanaged* server isn't bound to a PEM agent. You can create database objects on an unmanaged server, but extended PEM functionality (such as custom alerting) isn't supported on an unmanaged server.

You must also ensure the `pg_hba.conf` file of the server that you're registering allows connections from the host of the PEM web interface.

5.1.1 Accessing in-app help for UI details

The in-app help provides comprehensive details about the PEM web interface, including the menu items, charts and graphs on the dashboards, and the options on the dialog boxes. The in-app help also includes instructions for creating database objects.

- To access the landing page for the help, select **Help > Online help**.
- To access context-sensitive help for a dialog box, select **?**.

5.1.2 PEM client object browser

The Browser tree control provides access to information and management options for the database objects that reside on each server. The tree control expands to display a hierarchical view of the servers and objects that are monitored by the PEM server. You can use context menu options, accessed by right-clicking nodes of the tree control, to create new objects and modify and delete existing objects if your role holds the required privileges.

Expand nodes in the tree control to display a hierarchical view of the database objects that reside on a selected server:

- Use the plus sign (+) to the left of a node to expand a segment of the tree control.
- Use the minus sign (-) to the left of a node to close that node.

Context menu options can include one or more of the following selections:

Option	Action
Add named restore point	Create and enter the name of a restore point.
Backup	Open the *Tools menu* dialog box to back up database objects.
Backup Globals	Open the *Tools menu* dialog box to back up cluster objects.
Backup Server	Open the *Tools menu* dialog box to back up a server.
Connect Server	Establish a connection with the selected server.
Create	Access a context menu that provides context-sensitive selections. Your selection opens a Create dialog box for creating a new object.
CREATE Script	Open the *Query information* to edit or view the CREATE script.
Dashboards	Select for quick access to PEM dashboards.
Delete/Drop	Delete the currently selected object from the server.
Disconnect Database	Terminate a database connection.
Disconnect Server	Refresh the currently selected object.
Drop Cascade	Delete the currently selected object and all dependent objects from the server.
Debugging	Access the Debugger tool.
Grant Wizard	Access the *Tools menu* tool.
Maintenance	Open the *Management menu* dialog box to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Management	Access management tasks that are relevant to the node.
Properties	Review or modify the currently selected object's properties.
Refresh	Refresh the currently selected object.
Reload Configuration	Update configuration files without restarting the server.

continues on next page

Table 1 – continued from previous page

Option	Action
Restore	Access the *Tools menu* dialog box to restore database files from a backup.
View Data	Use the **View Data** option to access the data stored in a selected table with the **Data Output** tab of the Query tool.

The context-sensitive menus associated with Tables and nested Table nodes provides additional display options.

Option	Action
Import/Export	Open the Import/Export dialog box to import data to or export data from the selected table.
Reset Statistics	Reset statistics for the selected table.
Scripts	Open the *Query information* to edit or view the selected script from the flyout menu.
Truncate	Remove all rows from a table.
Truncate Cascade	Remove all rows from a table and its child tables.
View First 100 Rows	Access the data grid that displays the first 100 rows of the selected table.
View Last 100 Rows	Access the data grid that displays the last 100 rows of the selected table.
View All Rows	Access the data grid that displays all rows of the selected table.
View Filtered Rows	Access the **Data Filter** popup to apply a filter to a set of data.

5.1.3 PEM tabbed browser window

The main panel of the PEM client contains a collection of tabs that display information about the object currently selected in the tree control.

The **Dashboard** tab is context sensitive. When you navigate to the **Dashboard** tab from a server group or the PEM Agents node, the EDB Postgres Welcome window opens, where you can:

- Select the **Add New Server** icon to open the *Manually registering a database server* to define a connection to a server.
- Select the **Configure PEM** icon to open the *Editing the PEM server configuration* and modify server parameters.
- Select the **Getting Started** icon to open a new tab, displaying the PEM Getting Started section at the EnterpriseDB website.
- Select the **EDB Website** icon to navigate to the home page of the EnterpriseDB website. The EnterpriseDB website features news about upcoming events and other projects.
- Select the **PostgreSQL Website** icon to navigate to the PostgreSQL project website. The PostgreSQL site features news about recent releases and other project information.
- Select the **EDB Blogs** icon to navigate to the EDB Blog page, where you can review the most-recent employee posts to Postgres related blogs.

Select the name of an agent or server and navigate to the **Dashboard** tab to review session or server activity for the currently selected object.

When opened from the name of an agent or server, the **Dashboard** tab provides a graphical analysis of usage statistics:

- The Server sessions or Database sessions graph displays the interactions with the server or database.
- The Transactions per second graph displays the commits, rollbacks, and total transactions per second that are taking place on the server or database.
- The Tuples In graph displays the number of tuples inserted, updated, and deleted on the server or database.

- The Tuples Out graph displays the number of tuples fetched and returned from the server or database.
- The Block I/O graph displays the number of blocks read from the file system or fetched from the buffer cache (but not the operating system's file system cache) for the server or database.
- The Server activity tabbed panel displays tables that contain session information, session locks, prepared transactions, and configuration.

Navigate to the **Properties** tab to review the properties of the item currently selected in the tree control.

The **SQL** tab displays the SQL code used to generate the object currently selected in the Browser tree control.

The **Statistics** tab displays the statistics gathered for each object on the tree control. The statistics displayed in the table vary by the type of object that's highlighted. Select a column heading to sort the table by the data displayed in the column. Select it again to reverse the sort order. The following table lists some of the statistics that might be displayed.

Panel	Description
PID	The process ID associated with the row.
User	The name of the user that owns the object.
Database	The database name.
Backends	The number of current connections to the database.
Backend start	The start time of the backend process.
Xact Committed	The number of transactions committed to the database in the last week.
Xact Rolled Back	The number of transactions rolled back in the last week.
Blocks Read	The number of blocks read from memory in the last week, in MB.
Blocks Hit	The number of blocks hit in the cache in the last week, in MB.
Tuples Returned	The number of tuples returned in the last week.
Tuples Fetched	The number of tuples fetched in the last week.
Tuples Inserted	The number of tuples inserted into the database in the last week.
Tuples Updated	The number of tuples updated in the database in the last week.
Tuples Deleted	The number of tuples deleted from the database in the last week.
Last statistics reset	The time of the last statistics reset for the database.
Tablespace conflicts	The number of queries canceled because of recovery conflict with dropped tablespaces in database.
Lock conflicts	The number of queries canceled because of recovery conflict with locks in database.
Snapshot conflicts	The number of queries canceled because of recovery conflict with old snapshots in database.
Bufferpin conflicts	The number of queries canceled because of recovery conflict with pinned buffers in database.
Temporary files	The total number of temporary files, including those used by the statistics collector.
Size of temporary files	The size of the temporary files.
Deadlocks	The number of queries canceled because of a recovery conflict with deadlocks in database.
Block read time	The number of milliseconds required to read the blocks read.
Block write time	The number of milliseconds required to write the blocks read.
Size	The size of the selected database, in MB.

The **Dependencies** tab displays the objects on which the currently selected object depends. To ensure the integrity of the database structure, the server makes sure that you don't accidentally drop objects that other objects depend on. You must use `DROP CASCADE` to remove an object on which another object depends.

The **Dependencies** table displays:

- The **Type** field, which specifies the parent object type.
- The **Name** field, which specifies the identifying name of the parent object.

- The **Restriction** field, which describes the dependency relationship between the currently selected object and the parent.

The **Dependents** tab displays a table of objects that depend on the object currently selected in the Browser tree. A dependent object can be dropped without affecting the object currently selected in the Browser tree control.

- The **Type** field specifies the dependent object type.
- The **Name** field specifies the identifying name for the dependent object.
- The **Restriction** field describes the dependency relationship between the currently selected object and the parent.

Navigate to the **Monitoring** tab to access information presented on *Dashboards* . Dashboards display statistical information about the objects monitored by the PEM server.

PEM opens additional tabs when you access PEM functionality through the Management or Tools dialog boxes. Right-click the current tab and select from a context menu that allows you to customize the display for your working style:

- Select **Remove Panel** to remove the currently selected panel.
- Select **Rename Panel** to rename the currently selected panel.
- Select **Detach Panel** to detach the currently selected panel, repositioning it for convenience.
- Select **Add Panel** and select any of the available options to add to the panels.

The PEM client preserves any adjustments when you exit the program. To reset the PEM client to its original format, select **File > Reset Layout**.

5.1.4 Using chart, graph, and table controls

Use the icons in the upper-right corner of each graphic on a PEM client dashboard to control, download, and customize the charts, graphs, and tables displayed in the PEM client.

Use the **Refresh** icon to display the most recent content available from the PEM probes.

Select the **Download** icon to download a .jpeg or .png image of the chart or graph. By default, the file is in .jpeg format. To save the file as a .png, use the **Personalize** icon to modify the download format.

Select the **Fullscreen** icon to expand the chart or graph to fill the main pane of the PEM client.

Select the **Personalize** icon to modify the display properties of the chart or graph for your session only.

Use the **Information** icon to access information about the chart or graph.

Personalizing a graphic

When you select the **Personalize** icon, the Personalize chart configuration dialog box opens.

Use controls on the Personalize chart configuration dialog box to modify the properties of the graphic:

- Use the **Auto Refresh** control to increase or decrease the number of seconds between refreshes.
- Use the **Auto Refresh** field to specify the number of seconds between updates of the data displayed in the table or chart.
- If applicable, use the **Download as** field to indicate if you want to download a chart as a .jpeg image or a .png image.
- If applicable, use the **Colors** selectors to specify the display colors to use on a chart.

- If applicable, set the **Show Acknowledged Alerts** switch to **Yes** to indicate that you want the table to display alerts that you have acknowledged with a check box in the **Ack' ed** column. Set the field to **No** to indicate for the table to hide any acknowledged alerts. Acknowledged alerts aren' t purged from the table content until the time specified in the alert definition passes.

After personalizing the display properties, use the controls in the upper-right hand corner to apply your changes:

- Use the **Delete** icon to reset the properties of the graphic to their default settings. Use the menu to specify whether to apply the change to only this instance of the graphic or to the same graphic when displayed on other dashboards.
- Use the **Save** icon to save your changes to the properties for the graphic. Use the menu to specify to apply the change to only this instance of the graphic or to the same graphic when displayed on other dashboards.

5.1.5 Browser toolbar

The browser toolbar provides shortcuts for frequently used features, like View Data and the Query tool. This toolbar is visible on the Browser panel. Buttons get enabled/disabled based on the selected browser node.

- Use the **Query tool** button to open the Query tool in the current database context.
- Use the **View Data** button to view/edit the data stored in a selected table.
- Use the **Filtered Rows** button to access the Data Filter popup to apply a filter to a set of data for viewing/editing.

5.1.6 PEM menu bar

The PEM menu bar provides access to commands and features that you can use to manage your database servers and the objects that reside on those servers. If an option is disabled:

- The database server to which you' re currently connected might not support the selected feature.
- The selected menu option might not apply to the current object.
- The role that you used to connect to the server might not have enough privileges to change the selected object.

Context-sensitive menus across the top of the PEM web interface allow you to customize your environment and provide access to the enterprise management features of PEM.

File menu

Use the **File** menu to access the following options.

Option	Action
Preferences	Open the Preferences dialog box to customize your PEM client settings.
Lock Layout	Open a submenu to select the level for locking the UI layout.
Server Configuration	Open the Server Configuration dialog box and update your PEM server configuration settings.
Reset Layout	If a workspace panel is popped out, you can reset it to the default using **Reset Layout** .

Object menu

The **Object** menu is context sensitive. Use the **Object** menu to access the following options.

Option	Action
Create	Access a menu that provides context-sensitive selections.
Refresh	Refresh the currently selected object.
Delete/Drop	Delete the currently selected object from the server.
Connect Server	Open the Connect to Server dialog box to establish a connection with a server.
CREATE Script	Open the Query tool to edit or view the selected script.
Disconnect Server	Refresh the currently selected object.
BART	Access a menu that provides options for removing BART configuration, taking a BART backup, or revalidating the BART configuration.
Clear Saved Password	If you saved the database server password, clear the saved password. Enabled only after password is saved.
Clear SSH Tunnel Password	If you saved the ssh tunnel password, clear the saved password. Enabled only after password is saved.
Drop Cascade	Delete the currently selected object and all dependent objects from the server.
Hide	Hide the currently selected group. To view hidden groups, enable the **Show hidden groups** option in Preferences.
Properties	Review or modify the currently selected object's properties.
Trigger(s)	Disable or enable triggers for the currently selected table.
Truncate	Remove all rows from a table (**Truncate**) or remove all rows from a table and its child tables (**Truncate Cascade**).
View Data	Access a menu that provides several options for viewing data.
Remove Server	Remove the selected server from the browser tree.
Delete/Drop	Delete the currently selected object from the server.
Connect Database	Connect to selected database.
Count Rows	Count the number of rows of the selected table.
Reset Statistics	Reset the statistics of the selected table.
Scripts	CREATE, DELETE, INSERT, SELECT, and UPDATE script for the selected table.

Management menu

Use the **Management** menu to access the following PEM features.

Option	Action
Audit Manager	Open the Audit Manager and configure auditing on your monitored servers.
Auto Discovery	Open the Auto Discovery dialog box to configure a PEM agent to locate and bind monitored database servers.
Capacity Manager	Open the Capacity Manager dialog box and analyze historical or project future resource usage.
Log Manager	Open the Log Manager dialog box and configure log collection for a server.
Manage Alerts	Open the **Manage Alerts** tab and create or modify behavior for alerts.
Manage Charts	Open the **Manage Charts** tab to create or modify PEM charts.
Manage Dashboards	Open the Manage Dashboards dialog box to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Manage Probes	Open the Manage Probes dialog box to VACUUM, ANALYZE, REINDEX, or CLUSTER.

continues on next page

Table 6 – continued from previous page

Option	Action
Postgres Expert	Open the Postgres Expert wizard and perform a static analysis of your servers and databases.
Postgres Log Analysis Expert	Open the Postgres Log Analysis Expert dialog box to analyze log file contents for usage trends.
Scheduled Tasks	Open the **Scheduled Tasks** tab and review tasks that are pending or recently completed.
Tuning Wizard	Open the Tuning Wizard dialog box to generate a set of tuning recommendations for your server.
Reports	Open the Reports dialog box to generate the alert history report, system configuration report, or core usage report for your server.
Schedule Alert Blackout	Open the Schedule Alert Blackout dialog box and schedule the alerts blackout for your servers and agents.

Dashboards menu

Use the context-sensitive **Dashboards** menu to access dashboards.

Option	Action
Alerts	Open the Alerts dashboard for the selected node.
Audit Log	Open the Audit Log Analysis dashboard for the selected node.
PGD Admin	Open the PGD Admin dashboard for the selected node.
PGD Group Monitoring	Open the PGD Group Monitoring dashboard for the selected node.
PGD Node Monitoring	Open the PGD Node Monitoring dashboard for the selected node.
Database Server	Open the Database Analysis dashboard for the selected node.
I/O Analysis	Open the I/O Analysis dashboard for the selected node.
Memory	Open the Memory Analysis dashboard for the selected node.
Object Activity	Open the Object Activity Analysis dashboard for the selected node.
Operating System	Open the Operating System Analysis dashboard for the selected node.
Probe Log	Open the Probe Log Analysis dashboard for the selected node.
Server Log	Open the Server Log Analysis dashboard for the selected node.
Session Activity	Open the Session Activity Analysis dashboard for the selected node.
Storage	Open the Storage Analysis dashboard for the selected node.
Streaming Replication	Open the Streaming Replication Analysis dashboard for the selected node.
System Wait	Open the System Wait Analysis dashboard for the selected node.
Session Waits	Open the Session Waits Analysis Dasbhoard for the selected node.
Custom Dashboards	Open the Custom Dashboards that list the custom dashboards configured by the user.

Tools menu

Use the options on the **Tools** menu to access the following features.

Option	Action
Schema Diff	Open the Schema Diff dialog box to compare the schema objects between two database schemas.
Search objects	Open the Search Objects dialog box to search the database objects in a database.

continues on next page

Table 8 – continued from previous page

Option	Action
Server	Access the various server-related tools such as Add Named Restore Point, Performance Diagnostics, Queue Server Startup, Queue Server Shutdown, Replace Cluster Primary, Switchover EFM Cluster, and SQL Profiler.
Query tool	Open the Query tool for the currently selected object.
Reload Configuration	Update configuration files without restarting the server.
Pause replay of WAL	Pause replay of the WAL log.
Resume replay of WAL	Resume replay of the WAL log.
Import/Export	Open the Import/Export Data dialog box to import or export data from a table.
Maintenance	Open the Maintenance dialog box to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Backup	Open the Backup dialog box to back up database objects.
Backup Globals	Open the Backup Globals dialog box to back up cluster objects.
Backup Server	Open the Backup Server dialog box to back up a server.
Restore	Open the Restore dialog box to restore database files from a backup.
Grant Wizard	Open the Grant Wizard tool.
Schedule Backup	Open the Schedule Backup dialog box for BART backups.
New ERD Project(Beta)	Open the ERD Tool and start designing your database.
Storage Manager	Open the Storage manager to upload, delete or download the backup files.

Help menu

Use the options on the **Help** menu to access the online help documents or to review information about the PEM installation.

Option	Action
Quick Search	Type your keywords in the Quick Search field. Typing at least three characters displays all the matching possibilities under Menu items and the relevant documents under Help articles. Select the options under Menu items to perform action of particular functionality or object. Select any of the Help articles to open the help of that topic with highlighted text in a separate window. **Note** : If any of the options under Menu items is disabled, then it provides information by way of an info icon.
Online Help	Open in-app documentation for Postgres Enterprise Manager.
REST API Reference	Open the REST API reference.
EDB Website	Open the EDB website in a browser window.
About Postgres Enterprise Manager	Locate versioning and user information for Postgres Enterprise Manager.

5.1.7 Controlling and customizing charts, graphs, and tables

Use the icons in the upper-right corner of each graphic on a dashboard to control and customize the charts, graphs, and tables displayed in the PEM client for your current user session.

The following table gives the information about icons.

Icons	Information
! icon6	View information about the chart, graph, or table.

continues on next page

Table 10 – continued from previous page

Icons	Information
! icon1	Refresh the content of a chart, graph, or table.
! icon3	Personalize the chart, graph, or table settings for the current user.
! icon4	Download an image of the chart or graph.
! icon5	View the legends that are used in the chart, graph, or table.
! icon2	Expand the chart or graph to full-screen.

For more information about customizing the graphics displayed on the PEM dashboards, see the PEM client online help.

5.2 Connecting and disconnecting a database server

After defining a server connection, use the `Connect to Server` dialog to authenticate with a server and access the objects stored on the server. To access the dialog, right click on the server name in the PEM client tree control, and select `Connect Server` from the context menu.

If prompted, provide authentication information for the selected server:

- Use the `Password` field to provide the user' s password associated with the defined server.
- Select the box next to `Save Password` to instruct the server to save the password for future connections. If you save the password, you are not prompted while reconnecting to the database server with this server definition.

The browser displays a message in a green status bar in the lower right corner when the server connects successfully.

If you receive an error message while attempting a connection, verify that your network is allowing PEM and the host of the database server to communicate. For detailed information about a specific error message, see the Troubleshooting help page.

To review or modify connection details, right-click on the name of the server, and select `Properties...` from the context menu.

5.2.1 Disconnecting from a server

To disconnect from a server, right-click on the server name in the `Browser` tree control and select Disconnect Server from the context menu. A popup will ask you to confirm that you wish to disconnect the selected server.

5.3 Monitoring performance

PEM enables you to implement enterprise-wide performance monitoring of all managed servers.

The top-level dashboard is Global Overview. The Global Overview dashboard presents a status summary of all the servers and agents that are being monitored by the PEM server, a list of the monitored servers, and the state of any currently triggered alerts.

PEM provides a number of benefits not found in any other PostgreSQL management tool:

- Management en masse design — PEM is designed for enterprise database management and is built to tackle the management of large numbers of servers across geographical boundaries. Global dashboards visually keep you up to date on the up/down/performance status of all your servers.

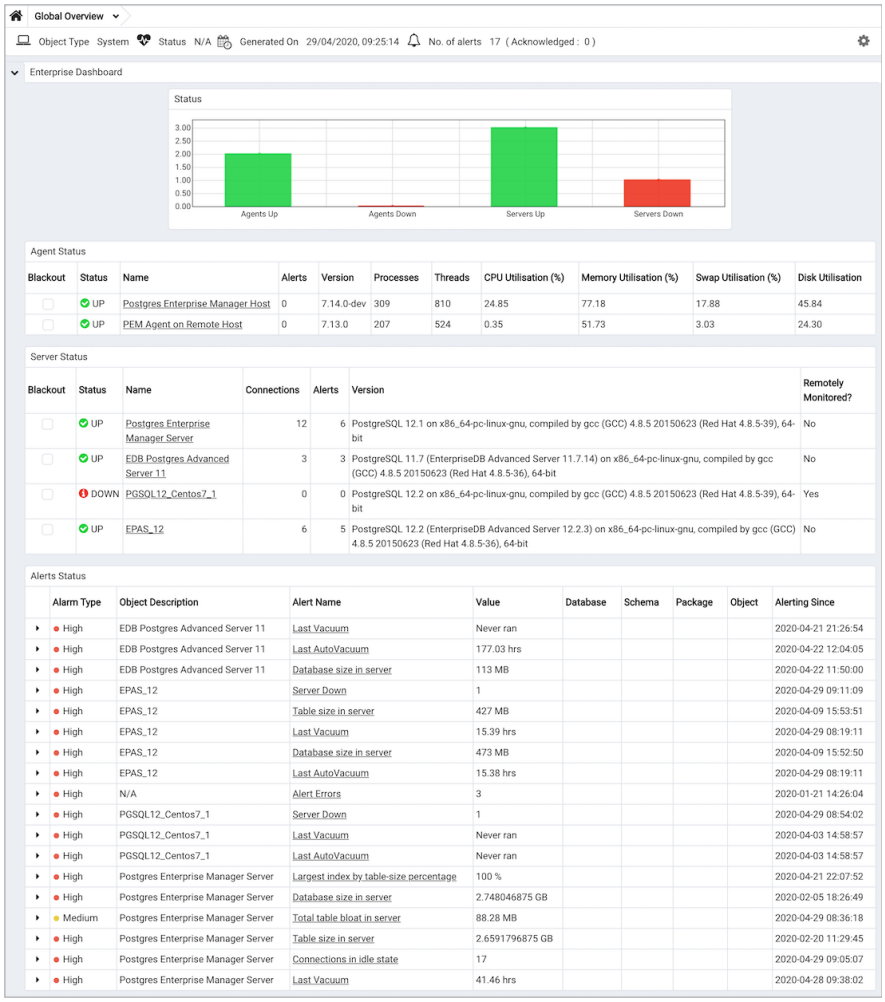


Fig. 2: The Global Overview dashboard

- Distributed architecture — The PEM architecture maximizes the ability to gather statistical information and to perform operations remotely on machines regardless of operating system platform.
- Graphical administration — All aspects of database administration can be carried out in the PEM client using a graphical interface. You can handle server startup and shutdown, configuration management, storage and security control, object creation, performance management, and more from a single console.
- Full SQL IDE — PEM contains a robust SQL integrated development environment (IDE) that provides ad hoc SQL querying, stored procedure/function development, and a graphical debugger.
- Enterprise performance monitoring — PEM provides enterprise-class performance monitoring for all managed database servers. Lightweight and efficient agents monitor all aspects of each database server's operations as well as each machine's underlying operating system. They provide detailed statistics to easily navigate performance pages in the interface.
- Proactive alert management — PEM lets you create performance thresholds for each key metric, such as memory and storage, that are monitored around the clock. Any threshold violation results in an alert sent to a centralized dashboard that communicates the nature of the problem and the actions needed to prevent the situation from jeopardizing the overall performance of the server.
- Simplified capacity planning — All key performance-related statistics are collected and retained for a specified period in the PEM repository. The Capacity Manager utility allows you to select various statistics and perform trend analysis to understand things such as peak load periods and storage consumption trends. A forecasting mechanism in the tool allows you to also forecast resource usage so you can plan and budget accordingly.
- Audit Manager — Audit Manager configures audit logging on EDB Postgres Advanced Server instances. You can log activities such as connections to a database, disconnections from a database, and the SQL statements run against a database. You can then use the Audit Log dashboard to filter and view the log.
- Log Manager — The Log Manager wizard configures server logging parameters with (optional) log collection into a central table. Use the wizard to specify your preference for logging behaviors such as log file rotation, log destination, and the error message severity. Use the Server Log dashboard to filter and review the collected server log entries.
- SQL workload profiling — PEM contains a SQL profiling utility that allows you to trace the SQL statements that are executed against one or more servers. SQL profiling can be done either in an ad hoc or scheduled manner. You can then filter captured SQL statements so you can easily identify and tune SQL statements that are running poorly. You can also feed SQL statements into an index advisor on EDB Postgres Advanced Server that analyzes each statement and makes recommendations on new indexes to create to help performance.
- Expert database analysis — PEM includes the Postgres Expert utility. Postgres Expert analyzes selected databases for best practice enforcement purposes. Areas such as general configuration and security setup are examined. Any deviations from recommended best practices are reported back to you, along with an explanation of each issue and expert help on what to do to make things right.
- Streaming replication monitor — You can monitor the Streaming Replication dashboard or use options on the PEM client to promote a replica node to the primary node.
- Secure client connectivity — PEM supports secure client connections through an encrypted SSH tunnel. The full-featured PEM client includes an SSH Tunnel definition dialog box that allows you to provide connection information for a secure connection.
- Wide platform support — PEM supports most major Linux and Windows platforms.

5.3.1 Probes

A *probe* is a scheduled task that retrieves information about the database objects that are being monitored by the PEM agent. PEM uses the collected information to build the graphs displayed on each homepage. The **Management > Manage Probes** tab allows you to modify the data collection schedule and the length of time that PEM retains information returned by a specific probe.

System probes

Unless otherwise noted, Postgres Enterprise Manager enables the following probes at the server, database, schema, extension (starting with version 8.4), or agent levels:

Name	Information monitored by probe	Level
Background Writer Statistics	Monitors information about the background writer. The information includes the number of: Timed checkpoints Requested checkpoints Buffers written (by checkpoint) Buffers written (by background writer) Background writer cycles Background buffers written Buffers allocated	Server
Barman Configuration	Information about the Barman tool global configuration.	Agent
Barman Information	Information about the Barman tool.	Agent
Barman Server	Information about the respective database server configuration monitored by Barman.	Agent
Barman Server Status	Information about the respective database server status monitored by Barman.	Agent
Barman Server Backup	Information about the backups of the respective database servers.	Agent
Barman Server WAL Status	Information about the Barman server WAL files.	Agent
Blocked Session Information	Information about blocked sessions.	Server
CPU Usage	CPU usage information.	Agent
Data and Log File Analysis	Information about log files. The information includes: The name of the log file The directory in which the log file resides	Server
Database Frozen XID	The frozen XID of each database.	Server
Database Size	Information about the size of the monitored databases. The information includes: The time the information was gathered The database name The database size (in MB).	Server
Database Statistics	Database statistics. The information includes: The number of backends The number of transactions committed The number of transactions rolled back The number of blocks read The number of blocks hit The number of rows returned The number of rows fetched The number of rows inserted The number of rows updated The number of rows deleted	Server
Disk Busy Info	Information about disk activity. Note: This probe isn't supported on Mac OS X, Solaris, or HP-UX.	Agent

continues on next page

Table 11 – continued from previous page

Name	Information monitored by probe	Level
Disk Space	Information about disk space usage. The information includes: The amount of disk space used The amount of disk space available	Agent
EDB Audit Configuration	The audit logging configuration of EDB Postgres Advanced Server.	Server
Failover Manager Cluster Info	Monitors a Failover Manager cluster, returning information about the cluster. This probe is enabled when a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog box.	Server
Failover Manager Node Status	Monitors a Failover Manager cluster, returning details about each node within the cluster. This probe is enabled when a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog box.	Server
Function Statistics	Monitors a database, retrieving information about functions. The information includes: Function names Argument types Return values	Database
Index Size	Monitors a database, retrieving information about indexes. The information includes: The name of the index The time the data was gathered The size of the index (in MB)	Database
Index Statistics	Index statistics. The information includes: The number of index scans The number of rows read The number of rows fetched The number of blocks read The number of blocks hit	Database
Installed Packages	The packages that are currently installed. The information gathered includes: The name of the installed package The version of the installed package The date and time that the probe executed	Agent
IO Analysis	Disk I/O information. The information includes: The number of blocks read The number of blocks written The date and time that the probe executed Note: This probe isn't supported on Mac OS X.	Agent
Load Average	CPU load averages. The information includes: The 1-minute load average The 5-minute load average The 15-minute load average Note: This probe isn't supported on Windows.	Agent
Lock Information	Lock information. The information includes: The database name The lock type The lock mode The process holding the lock	Server
Memory Usage	Information about system memory usage. The information includes: Total RAM in MB Free RAM in MB Total swap memory in MB Free swap memory in MB Shared system memory in MB (used by tuning wizard to tune the memory parameters for the database server) - On non-Windows systems, it is the *shmmax* value and read from */proc/sys/kernel/shmmax*. - On Windows, it is same as total memory.	Agent

continues on next page

Table 11 – continued from previous page

Name	Information monitored by probe	Level
Network Statistics	Network statistics. The information includes: The interface IP address The number of packets sent The number of packets received The number of bytes sent The number of bytes received The link speed (in MB/second)	Agent
Number of Prepared Transactions	Stores the number of prepared transactions.	Server
Number of WAL Files	The number of WAL files.	Server
Object Catalog: Database	Monitors a list of databases and their properties. The information includes: The database name The database encoding type If the database allows user connections or system connections	Server
Object Catalog: Foreign Key	Monitors a list of foreign keys and their properties. The information includes: The name of the table that contains the foreign key The name of the table that the foreign key references The name of the database in which the table resides The name of the schema in which the table resides	Schema
Object Catalog: Function	Monitors a list of functions and their properties. The information includes: The name of the function The name of the schema in which the function resides The name of the database in which the function resides	Schema
Object Catalog: Index	Monitors a list of indexes and their properties. The information includes: The name of the index The name of the table that the index is associated with The name of the database in which the indexed table resides	Schema
Object Catalog: Schema	Monitors a list of schemas and their associated databases and servers.	Database
Object Catalog: Sequence	Monitors a list of sequences and their properties.	Schema
Object Catalog: Table	Monitors a list of table information. The information includes: The table name The name of the schema in which the table resides The name of the database in which the schema resides A Boolean indicator that shows whether the table has a primary key	Schema
Object Catalog: Tablespace	Monitors a list of tablespaces.	Server
Operating System Information	Operating system details and boot time.	Agent
Package Catalog	The packages that are currently available for installation. The information gathered includes: The package name The package version	Agent
PG HBA Conf	Authentication configuration information from the *pg_hba.conf* file.	Server
Server Information	Server information.	Server
Session Information	Session information. The information includes: The name of the session user The date and time that the session connected to the server The status of the session at the time that the information was gathered (idle, waiting, and so on) The client address and port number	Server

continues on next page

Table 11 – continued from previous page

Name	Information monitored by probe	Level
Settings	Monitors the values currently assigned to GUC variables.	Server
SQL Protect	Monitors a server, retrieving information about SQL injection attacks.	Server
Slony Replication	Lag data for clusters replicated using Slony.	Database
Streaming Replication	Monitors a cluster that is using streaming replication, retrieving information about: The sent Xlog location (in bytes) The write Xlog location (in bytes) The flush Xlog location (in bytes) The replay Xlog location (in bytes) The Xlog lag (in segments) The Xlog lag (in pages)	Server
Streaming Replication Lag Time	Monitors a cluster that's using streaming replication, retrieving lag information about: Replication lag time (in seconds) Current status of replication (running/paused)	Server
Streaming Replication Database Conflicts	Monitors a database that's using streaming replication, retrieving information about any conflicts that arise. This includes information about queries that were canceled due to the number of: Drop tablespace conflicts Lock timeout conflicts Old snapshot conflicts Pinned buffer conflicts Deadlock conflicts	Server
Table Bloat	Information about the current table bloat. The information includes: The name of the table The name of the schema in which the table resides The estimated number of pages The estimated number of wasted pages The estimated number of bytes per row	Database
Table Frozen XID	Monitors the frozen XID of each table.	Schema
Table Size	Information about table size. The information includes: Table size (in MB) Total index size (in MB) Total table size, with indexes and TOAST (in MB)	Database
Table Statistics	Table statistics. The information includes: The number of sequential scans The number of sequential scan rows The number of index scans The number of index scan rows The number of rows inserted The number of rows updated The number of rows deleted The number of live rows The number of dead rows The last VACUUM The last auto-vacuum The last ANALYZE The last auto-analyze The number of pages estimated by ANALYZE The number of rows estimated by ANALYZE	Database
Tablespace Size	A list of tablespaces and their sizes.	Server
User Information	A list of the current users. The stored information includes: The user name The user type (superuser or non-superuser) The server to which the user is connected	Server
WAL Archive Status	Status of the WAL archive. The stored information includes: The number of WAL archives done The number of WAL archives pending The last archive time The number of WAL archives failed The time of the last failure	Server

continues on next page

Table 11 – continued from previous page

Name	Information monitored by probe	Level
xDB Replication	Lag data for clusters replicated using xDB replication.	Database
Multixact ID Exhaustion (Wraparound)	Information about the multixact ID. The stored information includes: The name of the database The oldest current multixact ID in the database Percentage toward wraparound Percentage toward emergency autovacuum	Server
Replication Slots	Information about the replication slots. The stored information includes: A unique, cluster-wide identifier for the replication slot. The base name of the shared object containing the output plugin this logical slot is using, or null for physical slot. The slot type: physical or logical. The OID of the database this slot is associated with, or null. Only logical slots have an associated database. The name of the database this slot is associated with, or null. Only logical slots have an associated database. Active field is True if this slot is currently actively being used. The oldest transaction that this slot needs the database to retain. VACUUM can't remove tuples deleted by any later transaction The oldest transaction affecting the system catalogs that this slot needs the database to retain. VACUUM can't remove catalog tuples deleted by any later transaction. The address (LSN) of oldest WAL, which still might be required by the consumer of this slot and thus won't be automatically removed during checkpoints.	Server

EDB Postgres Distributed (PGD) probes

To monitor the EDB Postgres Distributed group using dashboards, you must enable the following probes. Configure these probes at extension level.

Note: Prior to version 8.4, all these probes are available at the server level.

The user with pgd_superuser can view information from all the following probes. If you are using a version of EDB Postgres Distributed earlier than 4.0, all these probes work with EDB Postgres Distributed Enterprise Edition.

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Conflict History Summary	Information about row conflicts per conflict type. The stored information includes: The local time of the conflict The type of the conflict	Yes	Yes

continues on next page

Table 12 – continued from previous page

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Global Locks	Information about global locks in a EDB Postgres Distributed group. The stored information includes: The name of the node where the global lock originated The PID of the process holding the lock The type of lock (DDL or DML) The name of the locked relation (for DML Locks) or keys (for advisory locks) The internal state of the lock acquistion process The list of backends waiting for the same global lock The time when the global lock acquire was initiated by origin node The time when the local node started trying to acquire the local lock The time acquire_stage last changed	Yes	Yes
PGD Group Camo Details	Information about camos in a EDB Postgres Distributed group. The stored information includes: The name of the node The node name for whom this node is partner The node name for whom this node is origin The connection status The readiness status The number of pending or unresolved camo transactions The LSN of last applied WAL log The LSN of last received WAL log	No	No
PGD Group Replication Slot Details	Information about replication slots in a EDB Postgres Distributed group. The stored information includes: The name of the EDB Postgres Distributed group The name of the origin node The name of the target node The slot name on the origin node used by this subscription The active status The state of the replication (catchup,streaming, disconnected,..) The approximate lag time for reported write The approximate lag time for reported flush The approximate lag time for reported replay The bytes difference between sent_lsn and current WAL write position The bytes difference between write_lsn and current WAL write position The bytes difference between flush_lsn and current WAL write position	No	Yes
PGD Group Subscription Summary	Information about the summary of subscriptions in the EDB Postgres Distributed group. The stored information includes: The name of the origin of the subscription The name of the target of the subscription The timestamp of the last replayed transaction The lag between now and time of last replayed transaction	No	Yes
PGD Monitor Group Raft	The status and message of a cluster-wide raft check.	Yes	Yes

continues on next page

Table 12 – continued from previous page

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Group Raft Details	Information about raft consensus status from all the nodes in a EDB Postgres Distributed group. The stored information includes: The name of the node The raft worker state on the node The node id of the RAFT_LEADER The raft election internal id The raft snapshot internal id	Yes	Yes
PGD Monitor Group Versions	The status and message of cluster-wide version check.	Yes	Yes
PGD Group Versions Details	Information about version details of the installed Postgres, pglogical, EDB Postgres Distributed for each node in the EDB Postgres Distributed group. The stored information includes: The name of the node The installed Postgres version on the node The installed pglogical version on the node The version of the EDB Postgres Distributed on the node The EDB Postgres Distributed edition (Standard/Enterprise) for versions earlier than 4.0.	No	Yes
PGD Node Replication Rates	Information about outgoing replication activity from a given node. The stored information includes: The name of the target peer node The latest sent position The latest position reported as replayed The approximate lag time for reported replay The bytes difference between replay_lsn and current WAL write position on origin The human-readable bytes difference between replay_lsn and current WAL write position Approximate time required for the peer node to catch up to all the changes that are yet to be applied	Yes	No
PGD Node Slots	Information about the mapping of local EDB Postgres Distributed database nodes to replication slots, their status, and replication progress. The stored information includes: The name of the slot The name of the target node The name of the EDB Postgres Distributed group The database name on the target node The PID of the process attached to the slot The catalog XID needed by the slot The IP address of the client connection The latest sent position The latest position reported as replayed The approximate lag time for reported replay The bytes difference between replay_lsn and current WAL write position The human-readable bytes difference between replay_lsn and current WAL write position	Yes	Yes

continues on next page

Table 12 – continued from previous page

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Node Summary	Information about all the nodes in the EDB Postgres Distributed group. The stored information includes: The name of the node The name of the EDB Postgres Distributed group the node is part of The consistent state of the node in human-readable form The state which the node is trying to reach (during join or promotion) The name of subscribed repsets	Yes	Yes
PGD Workers	Information about workers in a EDB Postgres Distributed node. The stored information includes: The PID of the worker process The worker query start timestamp The worker state change timestamp The worker wait event type The worker wait event The worker state The worker role name The worker commit timestamp The worker local timestamp The name of the origin node The receive LSN The receive commit LSN The last exact replay LSN The last exact flush LSN The last exact replay timestamp The worker query	Yes	Yes
PGD Work Errors	Information about the work errors in EDB Postgres Distributed database node. The stored information includes: The process id of the worker causing the error The name of the EDB Postgres Distributed group the node is part of The name of the origin node The name of the source node The name of the target node The name of the subscription The internal identifier of the role of this worker The name of the role of this worker The date and time of the error The age of the error The description of the error The context in which the error happened The remote relation id The subscription writer id The subscription writer name	Yes	Yes

Customizing probes

A probe is a scheduled task that returns a set of performance metrics about a specific monitored object. A probe retrieves statistics from a monitored server, database, operating system, or agent. You can use the **Manage Probes** tab to override the default configuration and customize the behavior of each probe.

To open the **Manage Probes** tab, select **Management > Manage Probes**. The **Manage Probes** tab provides a set of icons that you can use to create and manage probes:

- Select **Manage Custom Probes** to open the **Custom Probes** tab and create or modify a custom probe.
- Select **Copy Probes** to open the Copy Probe dialog box and copy the probe configurations from the currently selected object to one or more monitored objects.

Note: Copy Probe isn't supported for the extension-level probes.

A probe monitors a unique set of metrics for each object type (server, database, database object, or agent). Select the name of an object in the tree to review the probes for that object.

To modify the properties associated with a probe, select the name of a probe and customize the settings that are displayed in the Probes table:

- Move the **Default** switch in the Execution Frequency columns to **N** to enable the **Minutes** and **Seconds** selectors, and specify a custom value for the length of time between executions of the probe.
- Move the **Default** switch in the Enabled? column to **No** to change the state of the probe and indicate if the probe is active or not.

Note: If data from a disabled probe is used in a chart, the chart displays an information icon in the upper-left corner that allows you to enable the probe by clicking the provided link.

- Move the **Default** switch in the Data Retention column to **No** to enable the **Day(s)** field and specify the number of days that information gathered by the probe is stored on the PEM server.

The **Manage Probes** tab might display information about probes that you can't modify from the current node. If you can't modify a probe from the current dialog box, the switches are disabled. Generally, you can modify a disabled probe from a node that's higher in the hierarchy of the PEM client tree control. Select another object in the tree control to modify the probes that are displayed or enabled in the **Manage Probes** tab.

Creating a custom probe

You can use the **PEM Custom Probes** tab to create a new probe or modify an existing custom probe. To open the **Custom Probes** tab, from the **Manage Probes** tab, select **Manage Custom Probes**.

Use the **Show System Probes?** switch to show or hide the system probes on the **Custom Probes** tab.

You can use the **Custom Probes** tab to create a probe or modify an existing one. To create a probe:

1. Select **Add** in the upper-right corner of the tab.
2. Provide a name for the new probe in the Probe Name column.
3. Select **Edit** (located to the left of the probe name) to review or add the probe definition.

Use the **General** tab to modify the definition of an existing probe or to specify the properties of a new probe:

- Use the **Probe Name** field to provide a name for a new probe.
- Use the **Collection method** field to specify the probe type. From the list, select:
- SQL — The probe gathers information by way of a SQL statement.
- WMI — The probe gathers information by way of a Windows Management Instrumentation extension.
- Batch — The probe uses a command-script or shell-script to gather information.

Before creating a batch probe on a Linux system, you must modify the `agent.cfg` file, setting the `allow_batch_probes` parameter to `true` . Then restart the PEM agent. The `agent.cfg` file is located in one of the following directories:

- If you installed PEM using graphical installer: `/opt/edb/pem/agent/etc/agent.cfg`
- If you installed PEM using RPM: `/usr/edb/pem/agent/etc/agent.cfg`

On Windows systems, agent settings are stored in the registry. Before creating a batch probe, modify the registry entry for the `AllowBatchProbes` registry entry and restart the PEM agent. PEM registry entries are located in `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent` .

Batch probes are platform specific. If you specify a collection method of **Batch**, you must specify a platform type in the **Platform** field.

To invoke a script on a Linux system, you must modify the entry for the `batch_script_user` parameter of the `agent.cfg` file and specify the user to use to run the script. You can either specify a non-root user or root for this parameter. If you don't specify a user or the specified user doesn't exist, then the script doesn't run. Restart the agent after modifying the file.

To invoke a script on a Windows system, set the registry entry for `AllowBatchJobSteps` to `true` and restart the PEM agent.

- Use the **Target Type** list to select the object type for the probe to monitor. **Target Type** is disabled if **Collection method** is WMI. Configure **Target Type** in probes to ensure that the collected metrics are relevant and accurate for the intended level of the system hierarchy. By setting the target type, you can effectively monitor various components, ranging from individual database objects like tables and indexes to server-wide monitoring.

Target type configuration matrix

The following table provides an overview of the target type configurations, their monitoring scope, and the mandatory fields for configuring them.

In this table:

```
- The **Target type** column determines the object you want to monitor with the probe.
```

- The **Execution level** column determines if the monitoring probes execute once per database, per server, per agent, or per extension level.
- The **Mandatory columns** column indicates the coloumns you must configure in the probe query to ensure the required data is collected.
- The **Probe examples** column provides some existing probes you can explore to better understand how probes are used in practice.

Target type	Execution level	Mandatory columns	Probe examples
Agent	Agent	None	cpu_usage
Server	Server	None	server_info
Database	Database	None	database_size
Schema	Database	schema_name	oc_extension
Table	Database	schema_name, table_name	table_size
Index	Database	schema_name, index_name	index_size
Sequence	Database	schema_name, sequence_name	oc_sequence
View	Database	schema_name, view_name	mview_size
Function	Database	schema_name, arg_types, function_type, function_name	function_statistics
Extension	Extension	None	bdr_node_summary

!!!note`1_blank_4_-` The custom probes set to a database or larger target type (including schema, table, index, view, sequence, and functions) collect the information at the database level. *1_blank_4_-* The system probes set to a schema or larger target type collect the information up to the schema level. .. ::
</details>

Proper configuration of the **Target Type** in probes ensures that the collected metrics are relevant and accurate for the intended level of the system hierarchy. By setting the appropriate target type, users can effectively monitor various components, from the server level down to individual database objects like tables and indexes.

- Use the **Minutes** and **Seconds** selectors to specify how often the probe collects data.

- Use the **Probe Enable?** switch to specify if the probe in enabled by default. Specify **Yes** to enable the probe by default or **No** to disable the probe by default.

Note: If data from a disabled probe is used in a chart, the chart displays an information icon in the upper-left corner that allows you to enable the probe by clicking the provided link.

- Use the **Data Retention** field to specify the number of days that gathered information is retained in the probe' s history table.
- Use the switch next to **Discard from history** to specify if the server creates a history table for the probe. Select **Yes** to discard probe history or **No** to retain the probe history in a table.
- Use the **Platform** list to specify the type of platform that the probe monitors. This field is enabled only when the **Collection** method is **Batch**.

Use the **Columns** tab to define the columns in which to store the probe data. To define a new column:

1. Navigate to the **Columns** tab and select **Add** in the upper-right corner.
2. In the **Name** field, provide a column name.
3. Select **Edit** (to the left of the new column name) to provide information about the column:

- Provide a descriptive name for the column in the **Name** field.
- The **Internal Name** field isn' t enabled for custom probes.
- Use the **Column Type** list to specify if the column is a key column (a primary key) or a non-key column. Non-key columns are generally metric items (values that can be graphed).
- Use the **Data Type** list to specify the type of data to store in the column.
- Use the **Unit** field to specify the unit of measure that applies to the metric stored in the column. This unit is displayed on the y-axis of a custom chart or a Capacity Manager chart. This is an optional field.
- Use the **Graphable** switch to specify if the defined metric can be graphed and to make the probe accessible from the Capacity Manager or Manage Charts dialog boxes.
- Use the **Is PIT** switch to specify whether to store the metric by point-in-time.

```
*Point-in-time* metrics change (increase or decrease) at any given point of time. For␣
↪example, database size is a point-in-time metric. At any given point-in-time, the size␣
↪of the database is fluctuating. Metrics that aren't point-in-time (also referred to as␣
↪cumulative metrics) are those whose size always increases over time. For example,␣
↪Blocks Read and Tuples Read are cumulative metrics. The value stays the same or␣
↪increases.
```

- Use the **Calculate PIT** switch to specify for the server to calculate a point-in-time value for the metric data. **Calculate PIT** is disabled if **Is PIT** is **Yes**.

PEM allows you to store point-in time-values of cumulative metrics as well. PEM subtracts the last collected value of a cumulative metric from the current value and stores the difference as a point-in-time value.

Use the **Code** tab to specify the default code for the propbe to execute:

- If the probe is a SQL probe, you must specify the **SQL SELECT** statement invoked by the probe on the **Code** tab. The column names returned by the query must match the **Internal Name** specified on the **Columns** tab. The number of columns returned by the query, as well as the column name, data type, and so on, must match the information specified on the **Columns** tab.

- If the probe is a batch probe, you must specify the shell or .bat script to invoke when the probe runs. The output of the script is as follows:
- The first line must contain the names of the columns provided on the **Columns** tab. Each column name must be separated by a tab (t) character. From the second line onwards, each line must contain the data for each column, separated by a tab character.
- If a specified column is defined as a key column, make sure that the script doesn't produce duplicate data for that column across lines of output. The number of columns specified in the **Columns** tab and their names, data type, and so on must match with the output of the script output.
- If the probe is a WMI probe, you must specify the WMI query as a **SELECT WMI** query. The column name referenced in the **SELECT** statement must be same as the name of the corresponding column specified on the **Column** tab. The column names returned by the query must match the **Internal Name** specified on the **Columns** tab. The number of columns returned by the query, as well as the column name, data type, and so on, must match the information specified on the **Columns** tab.

Use the **Alternate Code** tab to provide code to invoke if the probe fires on a specific version of the server. To provide version-specific code, move the **Applies to any server version?** switch to **No** and select **Add**. Then, use the **Database Version(s)** list to select a version, and select **Edit** (to the left of the version name) to provide the code to execute when the probe fires.

If you select a database version and leave the Probe Code column blank, PEM invokes the code specified on the **Code** tab when the probe executes on a server that matches that version.

When you've finished defining the probe, select **Save** (in the corner of the **Custom Probes** tab) to save the definition and make the probe data available for use on custom charts and graphs.

Exporting or importing a probe

To export the probe, select any probes from the **Manage Custom Probes** tab, and then select **Export** in the upper-right corner of the table. To generate the JSON file, select **Save File** and then select **OK**.

To import the probe, on the **Manage Custom Probes** tab, select **Import** in the upper-right corner of the table. Select **Browse** to select the JSON file with the probe code to import, and then select **Import**.

After selecting the file to import, you can select the **Skip existing** check box. This option skips the probe if it already exists and displays a message letting you know that the import was skipped.

- If you don't select the check box and the probe already exists, then it doesn't import the probe and an error message is reported.

Note: **Import** can't overwrite the existing probe as it might be configured to retain historical data as per the configured retention policy.

- If you don't select the check box and the probe doesn't exist but the corresponding table in the `pem` schema does, then it imports the probe successfully using the same table.

Note: It's possible for the probe to be deleted and not listed on **Manage Custom Probe** tab and for the table holding the data of that probe to exist in the *pem* schema.

Deleting a probe

You can delete only custom probes. To delete a probe, select the probe name in the probes table, and select **Delete** (located to the upper-right corner of the table). The probe history persists for the length of time specified in the **History Retention** field in the probe definition. During the deletion, the probe definition is deleted and any corresponding tables are dropped from the `pemdata` and `pemhistory` schemas.

System probes are the built-in probes provided by PEM and are part of the PEM schema. You can only modify system probes. If you attempt to delete a system probe, PEM reports an error.

Copying a probe

You can use the Copy Probe Configuration dialog box to copy probe definitions from one monitored object to one or more monitored objects of the same type. To open the Copy Probe Configuration dialog box:

1. Un the PEM client tree, select the object from which you are copying probes.
2. Select **Management > Manage Probes**.
3. From the **Manage Probes** tab, select **Copy Probe**.

The dialog box copies the probe definitions from the object from which you opened the Copy Probe Configuration dialog box to the locations selected on the tree control.

If you specify a parent node in the **Copy Probe Configuration** tree, PEM copies the probe configurations to each object of the same type that resides under that node in the tree. For example, to copy the probe definitions from one schema to all schemas that reside in a database, select only the parent database of the target schemas. A red warning symbol is displayed to the left of the name of a listed target object if that object is the source of the probe that' s being copied.

After you select the target object or objects, select **Configure Probes** to copy the probe definitions to the location selected in the dialog box.

Managing probes using pemworker

From version 9.6, you can enable and disable a probe using the pemworker utility using `--probe-display-name` , `--probe-internal-name` , or `--probe-id` option.

Examples

On Linux

This example shows how to disable a probe using the probe display name:

```
export PEM_SERVER_PASSWORD=edb
./pemworker --pem-user enterprisedb --disable-probe --probe-display-name "cpu usage"
__OUTPUT__
Wed Mar 27 01:43:40 2024 INFO: Probe status updated for agent (id: 1)
```

This example shows how to enable a probe using the probe display name:

```
export PEM_SERVER_PASSWORD=edb
./pemworker  --pem-user enterprisedb --enable-probe --probe-display-name "database size"
__OUTPUT__
Wed Mar 27 02:02:20 2024 INFO: Probe status updated for server (id: 1)
```

On Windows

This example shows how to disable a probe using the probe display name:

```
set PEM_SERVER_PASSWORD=edb
./pemworker.exe DISABLE-PROBE --pem-user postgres --probe-display-name "CPU usage"
__OUTPUT__
INFO: Probe status updated for agent (id: 1)
```

This example shows how to enable a probe using the probe display name:

```
set PEM_SERVER_PASSWORD=edb
./pemworker.exe ENABLE-PROBE --pem-user postgres --probe-display-name "CPU usage"
__OUTPUT__
INFO: Probe status updated for agent (id: 1)
```

PEM probes - reference

A probe is a scheduled task that retrieves information about the database objects that are being monitored by the PEM agent. PEM uses the collected information to build the graphs displayed on each dashboard. The **Manage Probes** tab (accessed from the **Management** menu) allows you to modify the data collection schedule and the length of time that PEM retains information returned by a specific probe.

Name	Information monitored	Level
Background Writer Statistics	Information about the background writer. The information includes: The number of timed checkpoints The number of requested checkpoints The number of buffers written (by checkpoint) The number of buffers written (by background writer) The number of background writer cycles The number of background buffers written The number of buffers allocated	Server
Blocked Session Information	Information about blocked sessions.	Server
CPU Usage	CPU Usage information.	Agent
Data and Log File Analysis	Information about log files. The information includes: The name of the log file The directory in which the log file resides	Server
Database Frozen XID	The frozen XID of each database.	Server
Database Size	Information about the size of the monitored databases. The information includes: The time the information was gathered The database name The database size (in MB).	Server
Database Statistics	Database statistics. The information includes the number of: Backends Transactions committed Transactions rolled back Blocks read Blocks hit Rows returned Rows fetched Rows inserted Rows updated Rows deleted	Server
Disk Busy Info	Information about disk activity. Note: This probe isn't supported on Mac OS X, Solaris, or HP-UX.	Agent

continues on next page

Table 14 – continued from previous page

Name	Information monitored	Level
Disk Space	Information about disk space usage. The information includes: The amount of disk space used The amount of disk space available	Agent
EDB Audit Configuration	The audit logging configuration of EDB Postgres Advanced Server.	Server
Failover Manager Cluster Info	Monitors a Failover Manager cluster, returning information about the cluster. This probe is enabled when you provide a cluster name and path of the Failover Manager binary in the Server Properties dialog box.	Server
Failover Manager Node Status	Monitors a Failover Manager cluster, returning detailed about each node in the cluster. This probe is enabled when you provide a cluster name and path of the Failover Manager binary in the Server Properties dialog box.	Server
Function Statistics	Monitors a database, retrieving information about functions. The information includes: Function names Argument types Return values	Database
Index Size	Monitors a database, retrieving information about indexes. The information includes: The name of the index The time the data was gathered The size of the index (in MB)	Database
Index Statistics	Index statistics. The information includes the number of: Index scans Rows read Rows fetched Blocks read Blocks hit	Database
Installed Packages	The packages that are currently installed. The information gathered includes: The name of the installed package The version of the installed package The date and time that the probe executed	Agent
IO Analysis	Monitors disk I/O information. The information includes: The number of blocks read The number of blocks written The date and time that the probe executed Note: This probe isn't supported on Mac OS X.	Agent
Load Average	CPU load averages. The information includes: The 1-minute load average The 5-minute load average The 15-minute load average Note: This probe isn't supported on Windows.	Agent
Lock Information	Monitors lock information. The information includes: The database name The lock type The lock mode The process holding the lock	Server
Memory Usage	Information about system memory usage. The information includes: Total RAM in MB Free RAM in MB Total swap memory in MB Free swap memory in MB Shared system memory in MB (used by tuning wizard to tune the memory parameters for the database server) - On non-Windows system, it's the *shmmax* value and read from */proc/sys/kernel/shmmax*. - On Windows, it's same as total memory.	Agent

continues on next page

Table 14 – continued from previous page

Name	Information monitored	Level
Network Statistics	Network statistics. The information includes: The interface IP address The number of packets sent The number of packets received The number of bytes sent The number of bytes received The link speed (in MB/second)	Agent
Number of Prepared Transactions	Stores the number of prepared transactions.	Server
Number of WAL Files	The number of WAL files.	Server
Object Catalog: Database	Monitors a list of databases and their properties. The information includes: The database name The database encoding type If the database allows user connections or system connections	Server
Object Catalog: Foreign Key	Monitors a list of foreign keys and their properties. The information includes: The name of the table that contains the foreign key The name of the table that the foreign key references The name of the database in which the table resides The name of the schema in which the table resides	Schema
Object Catalog: Function	Monitors a list of functions and their properties. The information includes: The name of the function The name of the schema in which the function resides The name of the database in which the function resides	Schema
Object Catalog: Index	Monitors a list of indexes and their properties. The information includes: The name of the index The name of the table that the index is associated with The name of the database in which the indexed table resides	Schema
Object Catalog: Schema	Monitors a list of schemas and their associated databases and servers.	Database
Object Catalog: Sequence	Monitors a list of sequences and their properties.	Schema
Object Catalog: Table	Monitors a list of table information. The information includes: The table name The name of the schema in which the table resides The name of the database in which the schema resides A Boolean indicator that shows whether the table has a primary key	Schema
Object Catalog: Tablespace	Monitors a list of tablespaces.	Server
Operating System Information	The operating system details and boot time.	Agent
Package Catalog	The packages that are currently available for installation. The information gathered includes: The package name The package version	Agent
PG HBA Conf	Authentication configuration information from the *pg_hba.conf* file.	Server
Server Information	Server information.	Server
Session Information	Session information. The information includes: The name of the session user The date and time that the session connected to the server The status of the session at the time that the information was gathered (idle, waiting, and so on) The client address and port number	Server

continues on next page

Table 14 – continued from previous page

Name	Information monitored	Level
Settings	The values currently assigned to GUC variables.	Server
SQL Protect	Monitors a server, retrieving information about SQL injection attacks.	Server
Slony Replication	Lag data for clusters replicated using Slony.	Database
Streaming Replication	Monitors a cluster that's using streaming replication, retrieving information about: The sent Xlog location (in bytes) The write Xlog location (in bytes) The flush Xlog location (in bytes) The replay Xlog location (in bytes) The Xlog lag (in segments) The Xlog lag (in pages)	Server
Streaming Replication Lag Time	Monitors a cluster that's using streaming replication, retrieving lag information about: Replication lag time (in seconds) Current status of replication (running/paused)	Server
Streaming Replication Database Conflicts	Monitors a database that's using streaming replication, retrieving information about any conflicts that arise. This includes information about queries that were canceled due to the number of: Drop tablespace conflicts Lock timeout conflicts Old snapshot conflicts Pinned buffer conflicts Deadlock conflicts	Server
Table Bloat	Information about the current table bloat. The information includes: The name of the table The name of the schema in which the table resides The estimated number of pages The estimated number of wasted pages The estimated number of bytes per row	Database
Table Frozen XID	The frozen XID of each table.	Schema
Table Size	Information about table size. The information includes: Table size (in MB) Total index size (in MB) Total table size, with indexes and TOAST (in MB)	Database
Table Statistics	Table statistics. The information includes: The number of sequential scans The number of sequential scan rows The number of index scans The number of index scan rows The number of rows inserted The number of rows updated The number of rows deleted The number of live rows The number of dead rows The last VACUUM The last auto-vacuum The last ANALYZE The last auto-analyze The number of pages estimated by ANALYZE The number of rows estimated by ANALYZE	Database
Tablespace Size	A list of tablespaces and their sizes.	Server
User Information	A list of the current users. The stored information includes: The user name The user type (superuser or non-superuser) The server to which the user is connected	Server
WAL Archive Status	The status of the WAL archive. The stored information includes: The number of WAL archives done The number of WAL archives pending The last archive time The number of WAL archives failed The time of the last failure	Server

continues on next page

Table 14 – continued from previous page

Name	Information monitored	Level
xDB Replication	Monitors lag data for clusters replicated using xDB replication.	Database

5.3.2 Alerts

PEM continually monitors registered servers. It compares performance metrics against predefined and user-specified thresholds that specify good or acceptable performance for each statistic. Any deviation from an acceptable threshold value triggers an alert. An alert is a system-defined or user-defined set of conditions that PEM compares to the system statistics. Alerts tell you about conditions on registered servers that require your attention.

Viewing the alerts via Global dashboard

When your system statistics deviate from the boundaries specified for that statistic, the alert triggers. The alert displays a high (red), low (yellow), or medium (orange) severity warning in the left-most column of the Alert Status table on the Global Overview dashboard.

Alerts Status

Alarm Type	Object Description	Alert Name	Value	Database	Schema	Package	Object	Alerting Since
High	EDB Postgres Advanced Server 11	Database size in server	113 MB					2020-04-22 11:50:00
High	EDB Postgres Advanced Server 11	Last Vacuum	Never ran					2020-04-21 21:26:54
High	EDB Postgres Advanced Server 11	Last AutoVacuum	140.21 hrs					2020-04-22 12:04:05
High	EPAS_12	Table size in server	410 MB					2020-04-09 15:53:51
Medium	EPAS_12	Last Vacuum	5.18 hrs					2020-04-27 20:47:50
High	EPAS_12	Database size in server	455 MB					2020-04-09 15:52:50
Medium	EPAS_12	Last AutoVacuum	5.16 hrs					2020-04-27 20:47:50
High	N/A	Alert Errors	3					2020-01-21 14:26:04
High	PGSQL12_Centos7_1	Server Down	1					2020-04-27 20:48:50
High	PGSQL12_Centos7_1	Last Vacuum	Never ran					2020-04-03 14:58:57
High	PGSQL12_Centos7_1	Last AutoVacuum	Never ran					2020-04-03 14:58:57
High	Postgres Enterprise Manager Server	Largest index by table-size percentage	100 %					2020-04-21 22:07:52
High	Postgres Enterprise Manager Server	Database size in server	2.072265625 GB					2020-02-05 18:26:49
High	Postgres Enterprise Manager Server	Table size in server	1.9814453125 GB					2020-02-20 11:29:45
Medium	Postgres Enterprise Manager Server	Connections in idle state	12					2020-04-27 16:20:32
Medium	Postgres Enterprise Manager Server	Last Vacuum	4.99 hrs					2020-04-27 20:47:50

Fig. 3: The Alert Status table

The PEM server includes a number of predefined alerts that are actively monitoring your servers. The alert definition might make details available about the cause of the alert. Select the down arrow to the right of the severity warning to open a dialog box that has details about the condition that triggered the alert.

Alert Details (Auto-refresh paused whilst rows are expanded.)

Ack'ed	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since
	High	Table size in server	1.9814453125 GB		Postgres Enterprise Manager Server					2020-02-20 11:29:45

General Parameters

Table name	Schema name	Database name	Total table size(MB)
table_statistics	pemhistory	pem	1087
server_logs	pemdata	pem	263
index_statistics	pemhistory	pem	237
session_info	pemhistory	pem	137
lock_info	pemhistory	pem	88

Fig. 4: Alert details

PEM also provides an interface that lets you create customized alerts. Each alert uses metrics defined on an alert template. An alert template defines how the server evaluates the statistics for a resource or metric. The PEM server includes predefined alert templates, and you can create custom alert templates.

Viewing the alerts via Alerts dashboard

Use the **Dashboards** menu (on the **Monitoring** tab) to open the Alerts dashboard. The Alerts dashboard shows a summary of the active alerts and the status of each alert.

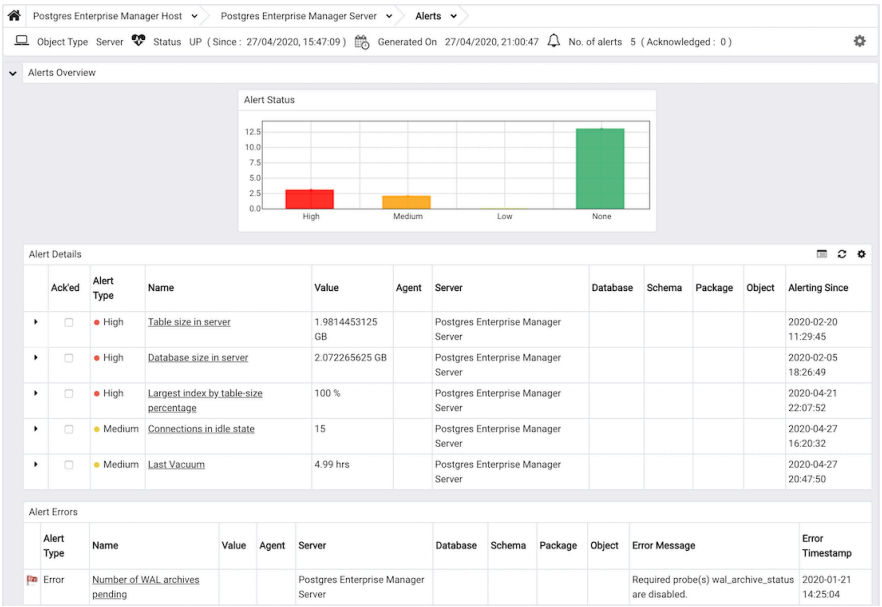


Fig. 5: The Alerts Dashboard

The Alerts dashboard header shows the date and time that the dashboard was last updated and the number of current alerts.

The Alerts Overview section shows a visual representation of the active alerts and a count of the current high, low, and medium alerts. The vertical bar on the left of the graph provides the count of the alerts displayed in each column. Hover over a bar to display the alert count for the selected alert severity in the upper-right corner of the graph.

The Alert Details table provides a list of the alerts that are currently triggered. The entries appear in order from high severity to low severity. Each entry includes information that lets you identify the alert and recognize the condition that triggered the alert. Select an alert to review detailed information about the alert definition.

The Alert Errors table shows configuration-related errors, such as accidentally disabling a required probe or improperly configuring an alert parameter. You can use the information provided in the Error Message column to identify and resolve the conflict that' s causing the error.

Customizing the Alerts dashboard

You can customize tables and charts that appear on the Alerts dashboard. To customize a table or chart, select **Settings** in the upper-right corner.

Use fields on the Personalize Chart Configuration dialog box to provide your display preferences:

- Use the **Auto Refresh** field to specify the number of seconds between updates of the data displayed in the table or chart.
- Use the **Download as** field to indicate whether to download a chart as a JPEG image or as a PNG image.
- Use **Colours selectors** to specify the colors to use on a chart.
- Set the **Show Acknowledged Alerts** switch to **Yes** if you want the table to display alerts that you acknowledged with a check box in the Ack' ed column. Set it to **No** to hide any acknowledged alerts. Acknowledged alerts are purged from the table content only when the time specified in the alert definition passes.

To save your customizations, select **Save** (a checkmark) in the upper-right corner. To delete any previous changes and revert to the default values, select **Delete**. Use the **Save** and **Delete** menus to specify whether to apply your preferences to all dashboards or to a selected server or database.

Managing alerts

Use the PEM client' s **Manage Alerts** tab to define, copy, or manage alerts. To open the **Manage Alerts** tab, select **Management > Manage Alerts**.

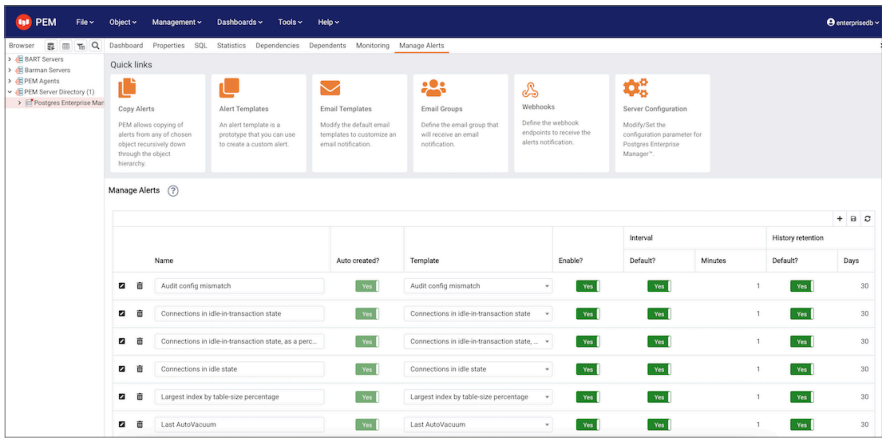


Fig. 6: The Manage Alerts tab

Use the Quick Links toolbar to open dialog boxes and tabs for managing alerts:

- Select **Copy Alerts** to open the Copy Alert Configuration dialog box and copy an alert definition.
- Select **Alert Templates** to open the **Alert Template** tab and modify or create an alert template.
- Select **Email Templates** to open the Email Template dialog box and modify the default email template to customize an email notification.
- Select **Email Groups** to open the **Email Groups** tab and modify or create an email group.
- Select **Webhooks** to open the **Webhooks** tab and create or manage the webhooks endpoints.
- Select **Server Configurations** to open the Server Configuration dialog box and review or modify server configuration settings.

- Select **Help** to open the PEM online help in a new tab.

Use the table in the Alerts section of the **Manage Alerts** tab to create new alerts or manage existing alerts.

Alert templates

An alert template is a prototype that defines the properties of an alert. An alert instructs the server to compare the current state of the monitored object to a threshold specified in the alert template to determine if a situation requires administrative attention.

You can use the **Alert Templates** tab to define a custom alert template or view the definitions of existing alert templates. To open the **Alert Templates** tab, select **Management > Manage Alerts**. From the **Manage Alerts** tab, on the Quick Links toolbar, select **Alert Templates**.

Use the **Show System Template** list to filter the alert templates that are displayed in the Alert Templates table. From the list, select a level of the PEM hierarchy to view all of the templates for that level.

Defining a new alert template

To define a new alert template, from the **Show System Template** list, select **None**. Then click the plus sign (+) in the upper-right corner of the alert template table. The alert template editor opens.

Use fields on the **General** tab to specify general information about the template:

- Use the **Template name** field to specify a name for the new alert template.
- Use the **Description** field to provide a description of the alert template.
- Use the **Target type** list to select the type of object that is the focus of the alert.
- Use the **Applies to server** list to specify the server type (EDB Postgres Advanced Server or PostgreSQL) to which to apply the alert. You can specify a single server type or **ALL**.
- Use the **History retention** field to specify the number of days to store the result of the alert execution on the PEM server.
- Use the **Threshold unit** field to specify the unit type of the threshold value.
- Use fields in the **Auto create** box to specify for PEM to use the template to generate an automatic alert. If you enable this option, PEM creates an alert when a new server or agent, as specified by the **Target type** list, is added and deletes that alert when the target object is dropped.
- Move the **Auto create?** slider to **Yes** to specify for PEM to create alerts based on the template. If you modify an existing alert template by changing the **Auto create?** slider to **Yes**, PEM creates alerts on the existing agents and servers. If you change the slider from **Yes** to **No**, the default threshold values in existing alerts are erased, and you can't recover them.
- Use the **Operator** list to select the operator for PEM to use when evaluating the current system values.

Select a greater-than sign (>) to trigger the alert when the system values are greater than the values entered in the **Threshold values** fields.

Select a less-than sign (<) to indicate to trigger the alert when the system values are less than the values entered in the **Threshold values** fields.

- Use the threshold fields to specify the values for PEM to compare to the system values to determine whether to raise an alert. You must specify values for all three thresholds (Low, Medium, and High).

- Use the **Check frequency** field to specify the default number of minutes between alert executions. This value specifies how often the server invokes the SQL code specified in the definition and compares the result to the threshold value specified in the template.

Use the fields on the **Probe Dependency** tab to specify the names of probes referred to in the SQL query specified on the **SQL** tab:

- Use the **Probes** list to select from a list of the available probes.
- To add the probe to the list of probes used by the alert template, select a probe name and select **Add**.
- To remove a probe from the selected probes list, select the probe name and select **Delete**.
- Use the **Parameters** tab to define the parameters to use in the SQL code specified on the **SQL** tab. Select the plus sign (+). Then:
- Use the **Name** field to specify the parameter name.
- Use the **Data type** list to specify the type of parameter.
- Use the **Unit** field to specify the type of unit specified by the parameter.
- Use the **Code** field on the **SQL** tab to provide the text of the SQL query for the server to invoke when executing the alert. The SQL query provides the result against which to compare the threshold value. If the alert result deviates from the specified threshold value, an alert is raised.

In the query, reference parameters defined on the **Parameters** tab sequentially by using the variable param_`x` . The `x` indicates the position of the parameter definition in the parameter list. For example, param_1 refers to the first parameter in the parameter list, param_2 refers to the second parameter in the parameter list, and so on.

The query can also include the following variables:

Variable description	Variable name
agent identifier	'${agent_id}'
server identifier	'${server_id}'
database name	'${database_name}'
schema name	'${schema_name}'
Table	'${object_name}'
index	'${object_name}'
sequence	'${object_name}'
function name	'${object_name}'

- Use the **Detailed Information SQL** field to provide a SQL query to invoke if the alert is triggered. The result set of the query might be displayed as part of the detailed alert information on the Alerts dashboard or Global Overview dashboard.

Note: If the specified query depends on one or more probes from different levels in the PEM hierarchy (server, database, schema, and so on), and a probe becomes disabled, any resulting alerts are displayed as follows:

```
-   If the alert definition and the probe referenced by the query are from the same␣
↪level in the PEM hierarchy, the server displays any alerts that reference the alert␣
↪template on the Alert Error table of the Global Alert dashboard. -   If the alert␣
↪definition and the probe referenced by the query are from different levels of the PEM␣
↪hierarchy, the server displays any triggered alerts that reference the alert template␣
↪on the Alert Details table of the hierarchy on which the alert was defined.
```

To save the alert template definition and add the template name to the Alert Templates list, select **Save**. After saving a custom alert template, you can use the Alerting dialog box to define an alert based on the template.

Exporting or importing alert templates

To export the alert template:

1. Select any alert template from the **Alert Templates** tab.
2. Select **Export** in the upper-right corner of the table.
3. Select **Save File**.
4. To generate the JSON file, select **OK**.

To import the Alert Template:

1. On the **Alert Templates** tab, select **Import** in the upper-right corner.
2. To select the JSON file with the code import, select **Browse**, and then select **Import**.
3. After selecting the file to import, you can select the following check boxes:

- **Skip existing** — Skip the alert template if it already exists.
- **Skip existing dependent probe** — The alert templates depend on probes. Select this check box to skip the dependent probe if it already exists.

If both the check boxes are selected and the alert template already exists, then it skips importing the alert template.

If you don't select the **Skip existing** check box, select **Skip dependent probe**, and the alert template already exists, then the alert template imports successfully.

If both the check boxes are cleared and the alert template doesn't exist, then it successfully imports the alert template.

Modifying or deleting an alert template

To view the definition of an existing template (including PEM predefined alert templates), use the **Show System Template** list to select the type of object monitored. When you select the object type, the Alert Templates table displays the alert templates that correspond with that object type.

Select a template name in the list, and select **Edit** at the left end of the row to review the template definition.

Use the Alert Templates dialog box to view detailed information about the alert template:

- The **General** tab displays general information.
- The **Probe Dependency** tab lists the names of probes that provide data for the template.
- The **Parameters** tab lists the names of any parameters referred to in the SQL code.
- The **SQL** tab displays the SQL code that defines the behavior of the alert.

To delete an alert template, select the template name in the alert templates table and select **Delete**, located in the upper-right corner of the table. The alert history persists for the time specified in the **History Retention** field in the template definition.

Predefined alert templates – reference

An alert definition contains a system-defined or user-defined set of conditions that PEM compares to the system statistics. If the statistics deviate from the boundaries specified for that statistic, the alert triggers, and the PEM client displays a warning on the Alerts Overview page and optionally sends a notification to a monitoring user.

The tables that follow list the system-defined alert templates that you can use to create an alert. This list is subject to change and can vary by system.

Templates applicable on agent

Template name	Description	Probe dependency
Load Average (1 minute)	1-minute system load average	load_average
Load Average (5 minutes)	5-minute system load average	load_average
Load Average (15 minutes)	15-minute system load average	load_average
Load Average per CPU Core (1 minutes)	1-minute system load average per CPU core	load_average
Load Average per CPU Core (5 minutes)	5-minute system load average per CPU core	load_average
Load Average per CPU Core (15 minutes)	15-minute system load average per CPU core	load_average
CPU utilization	Average CPU consumption	cpu_usage
Number of CPUs running higher than a	Number of CPUs running at greater than K% utilization threshold	cpu_usage
Free memory percentage	Free memory as a percent of total system memory	memory_usage
Memory used percentage	Percentage of memory used	memory_usage
Swap consumption	Swap space consumed (in megabytes)	memory_usage
Swap consumption percentage	Percentage of swap area consumed	memory_usage
Disk Consumption	Disk space consumed (in megabytes)	disk_space
Disk consumption percentage	Percentage of disk consumed	disk_space
Disk Available	Disk space available (in megabytes)	disk_space
Disk busy percentage	Percentage of disk busy	disk_busy_info
Most used disk percentage	Percentage used of the most utilized disk on the system	disk_space
Total table bloat on host	The total space wasted by tables on a host, in MB	table_bloat, settings
Highest table bloat on host	The most space wasted by a table on a host, in MB	table_bloat, settings
Average table bloat on host	The average space wasted by tables on host, in MB	table_bloat, settings
Table size on host	The size of tables on host, in MB	table_size
Database size on host	The size of databases on host, in MB	database_size
Number of ERRORS in the logfile on agent N in last X hours.	The number of ERRORS in the logfile on agent N in last X hours	N/A
Number of ERRORS in the audit logfile on agent N in last X hours	The number of ERRORS in the audit logfile on agent N in last X hours	N/A
Number of WARNINGS in the logfile on agent N in last X hours	The number of WARNINGS in the logfile on agent N in last X hours	N/A
Number of WARNINGS in the audit logfile on agent N in last X hours	The number of WARNINGS in the audit logfile on agent N in last X hours	N/A

continues on next page

Table 16 – continued from previous page

Template name	Description	Probe dependency
Number of WARNINGS or ERRORS in the logfile on agent N in last X hours	The number of WARNINGS or ERRORS in the logfile on agent N in last X hours	N/A
Number of WARNINGS or ERRORS in audit the logfile on agent N in last X hours	The number of WARNINGS or ERRORS in the logfile on agent N in last X hours	N/A
Package version mismatch	Check for package version mismatch as per catalog	N/A
Total materialized view bloat on host	The total space wasted by materialized views on a host, in MB	mview_bloat, settings
Highest materialized view bloat on host	The most space wasted by a materialized view on a host, in MB	mview_bloat, settings
Average materialized view bloat on host	The average space wasted by materialized views on host, in MB	mview_bloat, settings
Materialized view size on host	The size of materialized views on host, in MB	mview_size
Agent Down	Specified agent is currently down	N/A

Templates applicable on server

Template name	Description	Probe dependency
Total table bloat in server	The total space wasted by tables in server, in MB	table_bloat, settings
Largest table (by multiple of unbloated size)	Largest table in server, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB	table_bloat, settings
Highest table bloat in server	The most space wasted by a table in server, in MB	table_bloat, settings
Average table bloat in server	The average space wasted by tables in server, in MB	table_bloat, settings
Table size in server	The size of tables in server, in MB	table_size
Database size in server	The size of databases in server, in MB	database_size
Number of WAL files	Total number of Write Ahead Log files	number_of_wal_files
Number of prepared transactions	Number of transactions in prepared state	number_of_prepared_transactions
Total connections	Total number of connections in the server	session_info
Total connections as percentage of max_connections	Total number of connections in the server as a percentage of maximum connections allowed on server, settings	session_info, settings
Unused, non-superuser connections	Number of unused, non-superuser connections on the server, user_info, settings	session_info, user_info, settings
Unused, non-superuser connections as percentage of max_connections	Number of unused, non-superuser connections on the server as a percentage of max_connections of max_connections, user_info, settings	session_info, user_info, settings
Ungranted locks	Number of ungranted locks in server	blocked_session_info

continues on next page

Table 17 – continued from previous page

Template name	Description	Probe dependency
Percentage of buffers written by backends	The percentage of buffers written by backends vs. the total buffers written	background_writer_statistics
Percentage of buffers written by checkpoint	The percentage of buffers written by the checkpoints vs. the total buffers written	background_writer_statistics
Buffers written per second	Number of buffers written per second, over the last two probe cycles	background_writer_statistics
Buffers allocated per second	Number of buffers allocated per second, over the last two probe cycles	background_writer_statistics
Connections in idle state	Number of connections in server that are in idle state	session_info
Connections in idle-in-transaction state	Number of connections in server that are in idle-in-transaction state	session_info
Connections in idle-in-transaction state, as percentage of max_connections	Number of connections in server that are in idle-in-transaction state, as a percentage of maximum connections allowed on server, settings	session_info, settings
Long-running idle connections	Number of connections in the server that have been idle for more than N seconds	session_info
Long-running idle connections and idle transactions	Number of connections in the server that have been idle or transactions idle-in-transaction for more than N seconds	session_info
Long-running idle transactions	Number of connections in the server that have been idle in transaction for more than N seconds	session_info
Long-running transactions	Number of transactions in server that have been running for more than N seconds	session_info
Long-running queries	Number of queries in server that have been running for more than N seconds	session_info
Long-running vacuums	Number of vacuum operations in server that have been running for more than N seconds	session_info
Long-running autovacuums	Number of autovacuum operations in server that have been running for more than N seconds	session_info
Committed transactions percentage	Percentage of transactions in the server that committed vs. that rolled-back over last N minutes	database_statistics
Shared buffers hit percentage	Percentage of block read requests in the server that were satisfied by shared buffers, over last N minutes	database_statistics
Tuples inserted	Tuples inserted into server over last N minutes	database_statistics
InfiniteCache buffers hit percentage	Percentage of block read requests in the server that were satisfied by InfiniteCache, over last N minutes	database_statistics
Tuples fetched	Tuples fetched from server over last N minutes	database_statistics
Tuples returned	Tuples returned from server over last N minutes	database_statistics
Dead Tuples	Number of estimated dead tuples in server	table_statistics
Tuples updated	Tuples updated in server over last N minutes	database_statistics
Tuples deleted	Tuples deleted from server over last N minutes	database_statistics
Tuples hot updated	Tuples hot updated in server, over last N minutes	table_statistics
Sequential Scans	Number of full table scans in server, over last N minutes	table_statistics
Index Scans	Number of index scans in server, over last N minutes	table_statistics
Hot update percentage	Percentage of hot updates in the server over last N minutes	table_statistics
Live Tuples	Number of estimated live tuples in server	table_statistics
Dead tuples percentage	Percentage of estimated dead tuples in server	table_statistics

continues on next page

Table 17 – continued from previous page

Template name	Description	Probe dependency
Last Vacuum	Hours since last vacuum on the server	table_statistics
Last AutoVacuum	Hours since last autovacuum on the server	table_statistics
Last Analyze	Hours since last analyze on the server	table_statistics
Last AutoAnalyze	Hours since last autoanalyze on the server	table_statistics
Percentage of buffers written by backends over the last N minutes	The percentage of buffers written by backends vs. the total buffers backends over last N	background_writer_statistics
Table Count	Total number of tables in server	oc_table
Function Count	Total number of functions in server	oc_function
Sequence Count	Total number of sequences in server	oc_sequence
Number of users expiring in N days	Number of users whose accounts are expiring in N days	user_info
Number of users whose password expiring in N days	Number of users whose password have expired or are expiring in N days	user_info
Index size as a percentage of table size	Size of the indexes in server, as a percentage of their tables' size	index_size, oc_index, table_size
Largest index by table-size percentage	Largest index in server, calculated as percentage of its table's size, oc_index, table_size	index_size, oc_index, table_size
Number of ERRORS in the logfile on server M in the last X hours	The number of ERRORS in the logfile on server M in last X hours	N/A
Number of WARNINGS in the logfile on server M in the last X hours	The number of WARNINGS in logfile on server M in the last X hours	N/A
Number of WARNINGS or ERRORS in the logfile on server M in the last X hours	The number of WARNINGS or ERRORS in the logfile on server M in the last X hours	N/A
Number of attacks detected in the last N minutes	The number of SQL injection attacks occurred in the last N minutes	sql_protect
Number of attacks detected in the last N minutes by username	The number of SQL injection attacks occurred in the last N minutes by username	sql_protect
Number of replica servers lag behind the primary by write location	Streaming Replication: number of replica servers lag behind the primary by write location	streaming_replication
Number of replica servers lag behind the primary by flush location	Streaming Replication: number of replica servers lag behind the primary by flush location	streaming_replication
Number of replica servers lag behind the primary by replay location	Streaming Replication: number of replica servers lag behind the primary by replay location	streaming_replication
Replica server lag behind the primary by write location	Streaming Replication: replica server lag behind the primary by write location in MB	streaming_replication
Replica server lag behind the primary by flush location	Streaming Replication: replica server lag behind the primary by flush location in MB	streaming_replication
Replica server lag behind the primary by replay location	Streaming Replication: replica server lag behind the primary by replay location in MB	streaming_replication
Replica server lag behind the primary by size (MB)	Streaming Replication: replica server lag behind the primary by size in MB	streaming_replication
Replica server lag behind the primary by WAL segments	Streaming Replication: replica server lag behind the primary by WAL segments	streaming_replication
Replica server lag behind the primary by WAL pages	Streaming Replication: replica server lag behind the primary by WAL pages	streaming_replication

continues on next page

Table 17 – continued from previous page

Template name	Description	Probe dependency
Total materialized view bloat in server	The total space wasted by materialized views in server, in MB	mview_bloat, settings
Largest materialized view (by multiple of unbloated size)	Largest materialized view in server, calculated as a multiple of its own estimated unbloated size; exclude materialized views smaller than N MB	mview_bloat, settings
Highest materialized view bloat in server	The most space wasted by a materialized view in server, in MB	mview_bloat, settings
Average materialized view bloat in server	The average space wasted by materialized views in server, in MB	mview_bloat, settings
Materialized view size in server	The size of materialized view in server, in MB	mview_size
View Count	Total number of views in server	oc_views
Materialized View Count	Total number of materialized views in server	oc_views
Audit config mismatch	Check for audit config parameter mismatch	audit_configuration
Server Down	Specified server is currently inaccessible	N/A
Number of WAL archives pending	Streaming Replication: number of WAL files pending to be replayed at replica	wal_archive_status
Number of minutes lag of replica server from primary server	Streaming Replication: number of minutes replica node is lagging behind the primary node	streaming_replication_lag_time
Log config mismatch	Check for log config parameter mismatch	log_configuration
PGD Group Raft Consensus	PGD group Raft consensus not working	bdr_monitor_group_raft
PGD Group Raft Leader ID not matching	PGD group Raft leader ID not matching	bdr_group_raft_details
PGD Group versions check	PGD/pglogical version mismatched in PGD group	bdr_monitor_group_raft
PGD worker error detected	PGD worker error detected reported for PGD node	
Transaction ID exhaustion (wraparound)	Check for transaction ID exhaustion (wraparound)	
Inactive replication slots	Check for slots that are inactive for a particular server	
Conflicting replication slots	Check for slots that are conflicting for a particular server	
Multixact ID exhaustion (wraparound)	Check for multixact ID exhaustion (wraparound)	

Templates applicable on database

Template name	Description	Probe dependency
Total table bloat in database	The total space wasted by tables in database, in MB	table_bloat, settings
Largest table (by multiple of unbloated size)	Largest table in database, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB	table_bloat, settings
Highest table bloat in database	The most space wasted by a table in database, in MB	table_bloat, settings
Average table bloat in database	The average space wasted by tables in database, in MB	table_bloat, settings

continues on next page

Table 18 – continued from previous page

Template name	Description	Probe dependency
Table size in database	The size of tables in database, in MB	table_size
Database size	The size of the database, in MB	database_size
Total connections	Total number of connections in the database	session_info
Total connections as percentage of max_connections	Total number of connections in the database as a percentage of maximum connections allowed on server, settings	session_info, settings
Ungranted locks	Number of ungranted locks in database	blocked_session_info
Connections in idle state	Number of connections in database that are in idle state	session_info
Connections in idle-in-transaction state	Number of connections in database that are in idle-in-transaction state	session_info
Connections in idle-in-transaction state,as percentage of max_connections	Number of connections in database that are in idle-in-transaction state, as a percentage of maximum connections allowed on server, settings	session_info, settings
Long-running idle connections	Number of connections in the database that have been idle for more than N seconds	session_info
Long-running idle connections and idle transactions	Number of connections in the database that have been idle or idle-in-transaction for more than N seconds	session_info
Long-running idle transactions	Number of connections in the database that have been idle in transaction for more than N seconds	session_info
Long-running transactions	Number of transactions in database that have been running for more than N seconds	session_info
Long-running queries	Number of queries in database that have been running for more than N seconds	session_info
Long-running vacuums	Number of vacuum operations in database that have been running for more than N seconds	session_info
Long-running autovacuums	Number of autovacuum operations in database that have been running for more than N seconds	session_info
Committed transactions percentage	Percentage of transactions in the database that committed vs. that rolled-back over last N minutes	database_statistics
Shared buffers hit percentage	Percentage of block read requests in the database that were satisfied by shared buffers, over last N minutes	database_statistics
InfiniteCache buffers hit percentage	Percentage of block read requests in the database that were satisfied by InfiniteCache, over last N minutes	database_statistics
Tuples fetched	Tuples fetched from database over last N minutes	database_statistics
Tuples returned	Tuples returned from database over last N minutes	database_statistics
Tuples inserted	Tuples inserted into database over last N minutes	database_statistics
Tuples updated	Tuples updated in database over last N minutes	database_statistics
Tuples deleted	Tuples deleted from database over last N minutes	database_statistics
Tuples hot updated	Tuples hot updated in database, over last N minutes	table_statistics
Sequential Scans	Number of full table scans in database, over last N minutes	table_statistics
Index Scans	Number of index scans in database, over last N minutes	table_statistics
Hot update percentage	Percentage of hot updates in the database over last N minutes	table_statistics
Live Tuples	Number of estimated live tuples in database	table_statistics
Dead Tuples	Number of estimated dead tuples in database	table_statistics
Dead tuples percentage	Percentage of estimated dead tuples in database	table_statistics

continues on next page

Table 18 – continued from previous page

Template name	Description	Probe dependency
Last Vacuum	Hours since last vacuum on the database	table_statistics
Last AutoVacuum	Hours since last autovacuum on the database	table_statistics
Last Analyze	Hours since last analyze on the database	table_statistics
Last AutoAnalyze	Hours since last autoanalyze on the database	table_statistics
Table Count	Total number of tables in database	oc_table
Function Count	Total number of functions in database	oc_function
Sequence Count	Total number of sequences in database	oc_sequence
Index size as a percentage of table size	Size of the indexes in database, as a percentage of their tables' size	table_size
Largest index by table-size percentage	Largest index in database, calculated as percentage of its table's size, oc_index, table_size	index_size, oc_index, table_size
Database Frozen XID	The age (in transactions before the current transaction) of the database's frozen transaction ID	database_frozenxid
Number of attacks detected in the last N minutes	The number of SQL injection attacks occurred in the last N minutes	sql_protect
Number of attacks detected in the last N minutes by username	The number of SQL injection attacks occurred in the last N minutes by last N minutes by username	sql_protect
Queries that have been cancelled due to dropped tablespaces	Streaming Replication: number of queries that have been cancelled due to dropped tablespaces	streaming_replication_db_conflicts
Queries that have been cancelled due to lock timeouts	Streaming Replication: number of queries that have been cancelled due to lock timeouts	streaming_replication_db_conflicts
Queries that have been cancelled due to old snapshots	Streaming Replication: number of queries that have been cancelled due to old snapshots	streaming_replication_db_conflicts
Queries that have been cancelled due to pinned buffers	Streaming Replication: number of queries that have been cancelled due to pinned buffers	streaming_replication_db_conflicts
Queries that have been cancelled due to deadlocks	Streaming Replication: number of queries that have been cancelled due to deadlocks	streaming_replication_db_conflicts
Total events lagging in all slony clusters	Slony Replication: total events lagging in all slony clusters	slony_cluster
Events lagging in one slony cluster	Slony Replication: events lagging in one slony cluster	slony_cluster
Lag time (minutes) in one slony cluster	Slony Replication: lag time (minutes) in one slony cluster	slony_cluster
Total rows lagging in xdb single primary replication	xDB Replication: Total rows lagging in xdb single primary replication	xdb_smr_mmr_replication
Total rows lagging in xdb multi primary replication	xDB Replication: Total rows lagging in xdb multi primary replication	xdb_smr_mmr_replication
Total materialized view bloat in database	The total space wasted by materialized views in database, in MB	mview_bloat, settings
Largest materialized view (by multiple of unbloated size)	Largest materialized view in database, calculated as a multiple of its estimated unbloated size; exclude materialized views smaller than N MB	mview_bloat, settings
Highest materialized view bloat in database	The most space wasted by a materialized view in database, in MB	mview_bloat, settings
Average materialized view bloat in database	The average space wasted by materialized views in database, in MB	mview_bloat, settings
Materialized view size in database	The size of materialized view in database, in MB	mview_size
View Count	Total number of views in database	oc_views
Materialized View Count	Total number of materialized views in database	oc_views

continues on next page

Table 18 – continued from previous page

Template name	Description	Probe dependency

Templates applicable on schema

Template name	Description	Probe dependency
Total table bloat in schema	The total space wasted by tables in schema, in MB	table_bloat, settings
Largest table (by multiple of unbloated size)	Largest table in schema, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB	table_bloat, settings
Highest table bloat in schema	The most space wasted by a table in schema, in MB	table_bloat, settings
Average table bloat in schema	The average space wasted by tables in schema, in MB	table_bloat, settings
Table size in schema	The size of tables in schema, in MB	table_size
Tuples inserted	Tuples inserted in schema over last N minutes	table_statistics
Tuples updated	Tuples updated in schema over last N minutes	table_statistics
Tuples deleted	Tuples deleted from schema over last N minutes	table_statistics
Tuples hot updated	Tuples hot updated in schema, over last N minutes	table_statistics
Sequential Scans	Number of full table scans in schema, over last N minutes	table_statistics
Index Scans	Number of index scans in schema, over last N minutes	table_statistics
Hot update percentage	Percentage of hot updates in the schema over last N minutes	table_statistics
Live Tuples	Number of estimated live tuples in schema	table_statistics
Dead Tuples	Number of estimated dead tuples in schema	table_statistics
Dead tuples percentage	Percentage of estimated dead tuples in schema	table_statistics
Last Vacuum	Hours since last vacuum on the schema	table_statistics
Last AutoVacuum	Hours since last autovacuum on the schema	table_statistics
Last Analyze	Hours since last analyze on the schema	table_statistics
Last AutoAnalyze	Hours since last autoanalyze on the schema	table_statistics
Table Count	Total number of tables in schema	oc_table
Function Count	Total number of functions in schema	oc_function
Sequence Count	Total number of sequences in schema	oc_sequence
Index size as a percentage of table size	Size of the indexes in schema, as a percentage of their table's size	table_size
Largest index by table-size percentage	Largest index in schema, calculated as percentage of its table's size, oc_index, table_size	index_size, oc_index, table_size
Materialized view bloat	Space wasted by the materialized view, in MB	mview_bloat, settings
Total materialized view bloat in schema	The total space wasted by materialized views in schema, in MB	mview_bloat, settings
Materialized view size as a multiple of unbloated size	Size of the materialized view as a multiple of estimated unbloated size	mview_bloat

continues on next page

Table 19 – continued from previous page

Template name	Description	Probe dependency
Largest materialized view (by multiple of unbloated size)	Largest materialized view in schema, calculated as a multiple of its own estimated unbloated size; exclude materialized view smaller than N MB	mview_bloat, settings
Highest materialized view bloat in schema	The most space wasted by a materialized view in schema, in MB	mview_bloat, settings
Average materialized view bloat in schema	The average space wasted by materialized views in schema, in MB	mview_bloat, settings
Materialized view size	The size of materialized view, in MB	mview_size
Materialized view size in schema	The size of materialized views in schema, in MB	mview_size
View Count	Total number of views in schema	oc_views
Materialized View Count	Total number of materialized views in schema	ov_views
Materialized View Frozen XID	The age (in transactions before the current transaction) of the materialized view's frozen transaction ID	mview_frozenxid

Templates applicable on table

Template name	Description	Probe dependency
Table bloat	Space wasted by the table, in MB	table_bloat, settings
Table size	The size of table, in MB	table_size
Table size as a multiple of unbloated size	Size of the table as a multiple of estimated unbloated size	table_bloat
Tuples inserted	Tuples inserted in table over last N minutes	table_statistics
Tuples updated	Tuples updated in table over last N minutes	table_statistics
Tuples deleted	Tuples deleted from table over last N minutes	table_statistics
Tuples hot updated	Tuples hot updated in table, over last N minutes	table_statistics
Sequential Scans	Number of full table scans on table, over last N minutes	table_statistics
Index Scans	Number of index scans on table, over last N minutes	table_statistics
Hot update percentage	Percentage of hot updates in the table over last N minutes	table_statistics
Live Tuples	Number of estimated live tuples in table	table_statistics
Dead Tuples	Number of estimated dead tuples in table	table_statistics
Dead tuples percentage	Percentage of estimated dead tuples in table	table_statistics
Last Vacuum	Hours since last vacuum on the table	table_statistics
Last AutoVacuum	Hours since last autovacuum on the table	table_statistics
Last Analyze	Hours since last analyze on the table	table_statistics
Last AutoAnalyze	Hours since last autoanalyze on the table	table_statistics
Row Count	Estimated number of rows in a table	table_statistics
Index size as a percentage of table size	Size of the indexes on table, as a percentage of table's size	table_size
Table Frozen XID	The age (in transactions before the current transaction) of the table's frozen transaction ID	table_frozenxid

Global templates

Template name	Description	Probe dependency
Agents Down	Number of agents that haven't reported in recently	N/A
Servers Down	Number of servers that are currently inaccessible	N/A
Alert Errors	Number of alerts in an error state	N/A

Audit log alerting

PEM provides alert templates that let you use the Alerting dialog to create an alert that triggers when an `ERROR` or `WARNING` statement is written to a log file for a specific server or agent. To open the Alerting dialog, select the server or agent in the PEM client Object browser tree control, and select **Management > Alerting**.

To create an alert to notify you of error or warning messages in the log file for a specific server, create an alert that uses one of the following alert templates:

- Number of ERRORS in the logfile on server M in last X hours
- Number of WARNINGS in the logfile on server M in last X hours
- Number of ERRORS or WARNINGS in the logfile on server M in last X hours

To create an alert to notify you of error or warning messages for a specific agent, create an alert that uses one of the following alert templates. This functionality is supported only on EDB Postgres Advanced Server.

- Number of ERRORS in the logfile on agent M in last X hours
- Number of WARNINGS in the logfile on agent M in last X hours
- Number of ERRORS or WARNINGS in the logfile on agent M in last X hours

Defining a new alert

Use the PEM client **Manage Alerts** tab to define, copy, or manage alerts. To open the **Manage Alerts** tab, select **Management > Manage Alerts**.

The **Manage Alerts** tab displays a table of alerts that are defined on the object currently selected in the PEM client tree. You can use the Alerts table to modify an existing alert or to create a new alert.

To open the alert editor and create an alert, select the plus sign (+) in the upper-right of the table. The editor opens.

Use the fields on the **General** tab to provide information about the alert:

- Enter the name of the alert in the **Name** field.
- Use the **Template** list to select a template for the alert. An alert template is a function that uses one or more metrics or parameters to generate a value to which PEM compares user-specified alert boundaries. If the value returned by the template function evaluates to a value that' s within the boundary of a user-defined alert as specified by the **Operator** and **Threshold values** fields, PEM:
- Raises an alert
- Adds a notice to the Alerts overview display

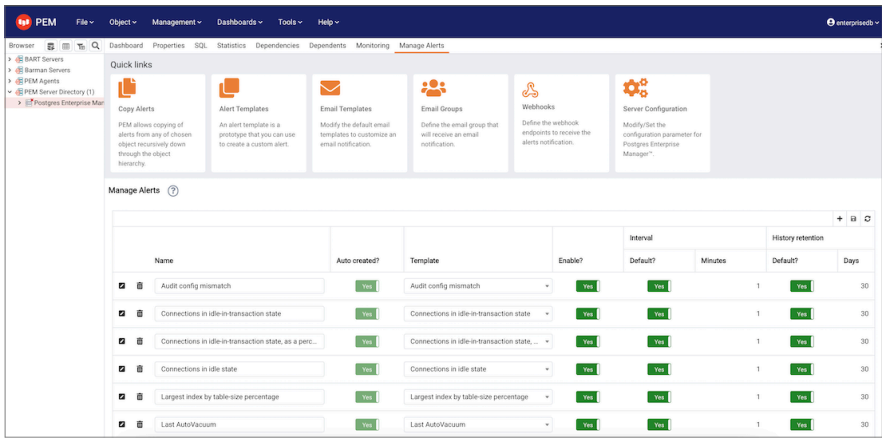


Fig. 7: The Manage Alerts tab

- Performs any actions specified on the template
- Use the **Enable?** switch to specify if the alert is enabled (**Yes**) or disabled (**No**).
- Use the **Interval** box to specify how often the alert confirms if the alert conditions are satisfied. Use the **Minutes** selector to specify an interval value. Use the **Default** switch to set or reset the **Minutes** value to the default (recommended) value for the selected template.
- Use the **History retention** box to specify the number of days that PEM stores data collected by the alert. Use the **Days** selector to specify the number of days to store the data. Use the **Default** switch to set or reset the **Days** value to the default value (30 days).
- Use controls in the **Threshold values** box to define the triggering criteria for the alert. When the value specified in the **Threshold values** fields evaluates to greater than or less than the system value (as specified with the Operator), PEM raises a Low, Medium or High alert level.
- Use the **Operator** list to select the operator for PEM to use when evaluating the current system values:
- Select a greater-than sign (>) to trigger the alert when the system values are greater than the values entered in the **Threshold values** fields.
- Select a less-than sign (<) to trigger the alert when the system values are less than the values entered in the **Threshold values** fields.
- Use the **Threshold** fields to specify the values for PEM to compare to the system values to determine whether to raise an alert. You must specify values for all three thresholds (Low, Medium, and High).

The Parameter Options table contains a list of parameters that are required by the selected template. The table displays both predefined parameters and parameters for which you must specify a value. You must specify a value for any parameter that displays a prompt in the **Value** column.

PEM can send a notification or execute a script if an alert is triggered or if an alert is cleared. Use the **Notification** tab to specify how PEM behaves if an alert is raised.

Use the **Email notification** box to specify the email group to receive an email notification if the alert is triggered at the specified level. Use the *Creating an email group* tab to create an email group that contains the address of the users to notify when an alert is triggered. To access the **Email Groups** tab, select **Email Groups** located in the **Quick Links** menu of the **Manage Alerts** tab.

- To instruct PEM to send an email when a specific alert level is reached, set the slider next to an alert level to **Yes**. Use the list to select the predefined user or group to notify.

You must configure the PEM server to use an SMTP server to deliver email before PEM can send email notifications.

Use the **Webhook notification** box to specify one or multiple endpoints if the alert is triggered at the specified level. Use the *Webhook* to create a webhook endpoint to receive the notifications when an alert is triggered. To access the **Webhooks** tab, select **Webhooks** located in the **Quick Links** menu of the **Manage Alerts** tab.

- Set **Enable?** to **Yes** to send the alert notifictions to the webhook endpoint.
- Set **Override default configuration?** to **Yes** to set the customized alert levels as per the requirement. Once it's set to **Yes**, all the alert levels are enabled to configure.
- Use the list to select a predefined endpoint to send a notification to for **Low alerts?**, **Medium alerts?**, **High alerts?**, and **Cleared alerts?**.

Use the **Trap notification** options to configure trap notifications for this alert:

- Set **Send trap** to **Yes** to send SNMP trap notifications when the state of this alert changes.
- Set **SNMP Ver** to **v1**, **v2**, or **v3** to identify the SNMP version.
- Use the **Low alert**, **Med alert**, and **High alert** sliders to select the levels of alert to trigger the trap. For example, if you set the slider next to **High alert** to **Yes**, PEM sends a notification when an alert with a high-severity level is triggered.

You must configure the PEM server to send notifications to an SNMP trap/notification receiver before notifications can be sent. For sending SNMP v3 traps, pemAgent uses 'User Security Model(USM)', which is in charge of authenticating, encrypting, and decrypting SNMP packets.

While sending SNMP v3 traps, the agent creates the `snmp_boot_counter` file. This file is created in the location mentioned by the `batch_script_dir` parameter in `agent.cfg`. If this parameter isn't configured or if the directory isn't accessible due to authentication restrictions, then the file is created in the operating system temporary directory. If that's also not possible, then the file is created in your home directory.

Use the **Nagios notification** box to instruct the PEM server to notify Nagios network-alerting software when the alert is triggered or cleared. For more details, see *Using PEM with Nagios*

- Set the **Submit passive service check result to Nagios** switch to **Yes** to notify Nagios when the alert is triggered or cleared.
- Use the **Script execution** box to optionally define a script that executes if an alert is triggered and to specify details about the script execution.
- Set the **Execute script** slider to **Yes** to instruct PEM to execute the provided script if an alert is triggered.
- Set the **Execute on alert cleared** slider to **Yes** to instruct PEM to execute the provided script when the situation that triggered the alert is resolved.
- Use the **Execute script on** options to indicate for the script to execute on the PEM server or the monitored server.
- In the **Code** field, provide the script for PEM to execute. You can provide a batch/shell script. In the script, you can use placeholders for the following:

`%AlertName%` — The name of the triggered alert.

`%ObjectName%` — The name of the server or agent on which the alert was triggered.

`%ThresholdValue%` — The threshold value reached by the metric when the alert triggered.

`%CurrentValue%` — The current value of the metric that triggered the alert.

`%CurrentState%` — The current state of the alert.

`%OldState%` — The previous state of the alert.

`%AlertRaisedTime%` — The time that the alert was raised or the most recent time that the alert state was changed.

To invoke a script on a Linux system, you must modify the entry for the `batch_script_user` parameter of the `agent.cfg` file and specify the user to use to run the script. You can either specify a non-root user or root for this parameter.

If you don't specify a user or the specified user doesn't exist, then the script doesn't execute. Restart the agent after modifying the file.

To invoke a script on a Windows system, set the registry entry for `AllowBatchJobSteps` to true and restart the PEM agent. PEM registry entries are located in `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent` .

After you define the alert attributes, select **Edit** to close the alert definition editor and then **Save** in the upper-right corner of the Alerts table.

To discard your changes, select **Refresh**. A message prompts you to confirm that you want to discard the changes.

Note: Suppose you need to use the alert configuration placeholder values in an external script. You can do so either by passing them as the command-line arguments or exporting them as environment variables. The external script must have proper execution permissions.

- You can run the script with any of the placeholders as command-line arguments.

For example:

```
#!/bin/bash

bash <path_to_script>/script.sh "%AlertName%  %AlertLevel% %AlertDetails%"
```

- You can define the environment variables for any of the placeholders and then use those environment variables in the script.

For example:

```
#!/bin/bash

export AlertName=%AlertName%
export AlertState=%AlertState%

bash <path_to_script>/script.sh
```

Modifying an alert

Use the Alerts table to manage an existing alert or create a new alert. Select an object in the PEM client tree to view the alerts that monitor that object.

You can modify some properties of an alert in the Alerts table:

- The Alert name column displays the name of the alert. To change the alert name, replace the name in the table and select **Save**.
- The Alert template column displays the name of the alert template that specifies properties used by the alert. You can use the list to change the alert template associated with an alert.
- Use the **Alert enable?** switch to specify if an alert is enabled (**Yes**) or disabled (**No**).
- Use the Interval column to specify how often PEM checks whether the alert conditions are satisfied. Set the **Default** switch to **No** and specify an alternate value, in minutes. Or set the **Default** switch to **Yes** to reset the value to its default setting. By default, PEM checks the status of each alert once every minute.
- Use the **History retention** field to specify the number of days that PEM stores data collected by the alert. Set the **Default** switch to **No** and specify an alternative value in days. Or set the **Default** switch to **Yes** to reset the value to its default setting. By default, PEM stores historical data for 30 days.

After modifying an alert, select **Save** (located in the upper-right corner of the table) to preserve your changes.

To modify other alert attributes, select **Edit** to the left of an alert name to open an editor. The editor provides access to the complete alert definition.

Use the Alert Details dialog box to modify the definition of the selected alert. After you modify the alert definition, select **Save**.

Deleting an alert

To mark an alert for deletion, select the alert name in the Alerts table. Then select **Delete** to the left of the name. The alert remains in the list in red strike-through font.

Delete is a toggle. You can undo the deletion by selecting it a second time. To permanenetly dete the alert defintion, select **Save**.

Copying an alert

To speed up the deployment of alerts in the PEM system, you can copy alert definitions from one object to one or more target objects.

To copy alerts from an object, select the object in the PEM client tree on the main PEM window. Then, select **Management > Copy Alerts**. On the **Manage Alerts** tab, from the Quick Links toolbar, select **Copy Alerts**.

The Copy Alert Configuration dialog box copies all alerts from the object selected in the PEM client tree to the objects selected on the dialog box. Expand the tree to select nodes to specify as the target objects. The tree displays a red warning indicator next to the source object.

To copy alerts to multiple objects at once, select a parent node of the targets. For example, to copy the alerts from one table to all tables in a schema, select the check box next to the schema. PEM copies alerts only to targets that are the same type as the source object.

Select **Ignore duplicates** to prevent PEM from updating any existing alerts on the target objects with the same name as those being copied.

Select **Replace duplicates** to replace existing alerts with alerts of the same name from the source object.

Select **Delete Existing Alerts** to delete all the alerts from the target object and copy all the alerts from the source object to the target object.

Select **Configure Alerts** to copy the alerts from the source object to all objects of the same type in or under those objects selected on the Copy Alert Configuration dialog box.

Schedule an alert blackout

You can use the **Management > Schedule Alert Blackout** to schedule an alert blackout for your Postgres servers and PEM agents during maintenance. Alerts aren' t raised during a defined blackout period.

To schedule an alert blackout, select **Management > Schedule Alert Blackout**.

In the Schedule Alert Blackout dialog box, use the tabs to define the blackout period for servers and agents. On the **Server** tab, to add a row, select the plus sign (+) at the top-right corner.

Use the **Server** tab to provide information about an alert blackout period. After you save the blackout period, you can' t edit it.

- Use the **Start time** field to provide the date and time to start the alert blackout.

- Use the **Duration** field to provide the interval for which you want to black out the alerts.
- Use the **Servers** field to provide the server name for which you want to black out the alerts. You can also select multiple servers to black out the alerts for all of those servers.

After providing details, select **Save**. The alerts don't appear on the Alerts dashboard for the scheduled interval of that server.

You can also schedule a blackout period for PEM agents using the **Agent** tab on the dialog box. To add a row, on the **Agent** tab, select the plus sign (+) at the top-right corner.

Use the **Agent** tab to provide the information about an alert blackout period. After you save the blackout period, you can't edit it.

- Use the **Start time** field to provide the date and time to start the alert blackout.
- Use the **Duration** field to provide the interval for which you want to black out the alerts.
- Use the **Agents** field to provide the agent name for which you want to black out the alerts. All server-level alerts for the servers bound to that agent black out.

After providing details, save the details by selecting **Save**. The alerts aren't displayed on the Alert dashboard for the scheduled interval for that PEM agent.

You can select **Clone** from the top-right corner of the dialog box to clone the scheduling of an alert blackout. To create the cloned copy of all the selected servers or agents, select the servers or agents you want to clone, and then select **Clone**. You can edit newly created schedules as needed, and then select **Save**.

Select **Delete** from the top-right corner of the dialog box to remove a scheduled alert blackout. Select the servers or agents and then select **Delete**.

Select a server for which you want to delete the scheduled alert blackout, and then select **Delete**. The server prompts for confirmation before deleting that row.

You can select **Reset** to reset the details on the Alert Blackout dialog box to the default settings. Saved blackouts aren't affected.

You can view the scheduled alert blackout details from the `event_history` table in the `pem` schema once the schedule is executed. For more information, see *Monitoring event history* .

5.3.3 Notifications

PEM can send a notification or execute a script if an alert is triggered or cleared. You can send notifications using the following options:

- SMTP
- Webhooks
- SNMP
- Nagios

Use the **Notification** tab to specify PEM behavior when an alert is raised.

SMTP

You must configure the PEM server to use an SMTP server to deliver email before PEM can send email notifications.

Creating an email group

PEM monitors your system for conditions that require attention. You can use an email group to specify the email addresses of users for the server to notify if current values deviate from threshold values specified in an alert definition. An email group can notify multiple users or target specific users during the time periods you specify.

Use the **Email Groups** tab to configure groups of SMTP email recipients. To open the **Email Groups** tab, in the PEM client, select **Management > Manage Alerts**. When the **Manage Alerts** tab opens, select **Email Groups** from the Quick Links toolbar.

The **Email Groups** tab displays a list of the currently defined email groups. To modify an existing group, select a group name and select **Edit** at the far left end of the row.

To define a new email group, select the plus sign (+) in the upper-right corner of the **Email Groups** table. Use the Email Groups dialog box to define an email group and its members.

Each row in the email group definition associates a set of email addresses with a specific time period. When an alert is triggered, the server evaluates the times specified in each row and sends the message to those group members whose definitions are associated with the time that the alert triggered.

1. Provide a name for the email group in the **Group Name** field.
2. To open the **Options** tab, select the plus sign (+) in the group members table.
3. Add the member addresses to receive notifications for the time period specified:

- Enter a comma-delimited list of recipient addresses in the **Reply to Addresses** field.
- Enter a comma-delimited list of addresses to receive a copy of the email in the **CC Addresses** field.
- Enter a comma-delimited list of addresses to receive a copy of the email without the knowledge of other recipients in the **Bcc Addresses** field.
- Enter the email address to send the messages from in the **From Address** field.
- Use the **Subject prefix** field to provide a message to add to the start of each subject line when a notification is sent.
- Use the **From Time** and **To Time** selectors to specify the time range for notifications to the group members that are identified on this row. Provide these values in the locale of the PEM client host. The PEM server translates the time into other time zones as required.

1. Select **Add** to add a row to the table, and specify another time period and the email addresses to notify during those hours.
2. When you' ve finished defining the email group, select **Save**.

After creating the email group, you can use the **Manage Alerts** tab to set up the notification details for an alert to direct notifications to the group.

Deleting an email group

To delete an email group, in the Email Group table, select the name of the group and select **Delete**, located to the left of the group name.

The group name appears in the Email Group table in red. Select **Save** to permanently remove the group from the table.

Webhook

You must configure the PEM Server to use webhooks to receive notification of alert events on threshold value violations in your configured applications.

Creating a webhook

PEM monitors your system for conditions that require user attention. You can use a webhook to create the endpoints to receive a notification if current values deviate from threshold values specified in an alert definition. Based on the events triggered, PEM sends a notification to multiple webhook endpoints or to specific target webhook endpoints.

Use the **Webhooks** tab to configure endpoint recipients. To open the **Webhooks** tab, select **Management > Manage Alerts**. From the **Manage Alerts** tab, on the Quick Links toolbar, select **Webhooks**.

The **Webhooks** tab displays a list of the currently defined recipient applications as endpoints. Select an endpoint and select **Edit** at the far left end of the row to modify an existing endpoint.

To define a new webhook, select the plus sign (+) in the upper-right corner of the table. You can then use the **General** tab to define the basic details of the webhook:

- Provide a name for the webhook in the **Name** field.
- Specify a webhook URL to deliver all the notifications to in the **URL** field.
- Set the request method type used to make the call in the **Request Method** field: **POST** or **PUT**.
- By default, webhooks are enabled. To disable a webhook, set **Enable?** to **No**.

Note: The **Enable?** setting works only if the *enable_webhook* parameter is set to *true* in the *agent.cfg* file. By default, the *enable_webhook* parameter is set to *true* only for the agent running on the PEM server host. For all other agents running on other hosts, you need to set it to *true* manually.

Defining webhook SSL configurations

You can define the webhook SSL parameters in the respective agent configuration file or registry in Windows. You can find the list of webhook SSL parameters in *Modifying agent configuration* . If you add or remove any of the agent configuration parameters, you must restart the agent to apply them.

- On Windows systems, PEM registry entries for webhooks are located in `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent\WEBHOOK` .
- On Linux systems, PEM configuration options for webhooks are stored in the `agent.cfg file` , located by default in `/usr/edb/pem/agent/etc` .

```
[WEBHOOK/Django]
webhook_ssl_key=<webhook_client_ssl_key_path>
webhook_ssl_crt=<webhook_client_ssl_certificate_path>
webhook_ssl_ca_crt=<webhook_server_ca_certificate_path>
webhook_ssl_crl=<crl_file_path_to_validate_webhook_server>
allow_insecure_webhooks=<true|false>
```

Fig. 8: Example - Webhook SSL Parameters in agent.cfg file

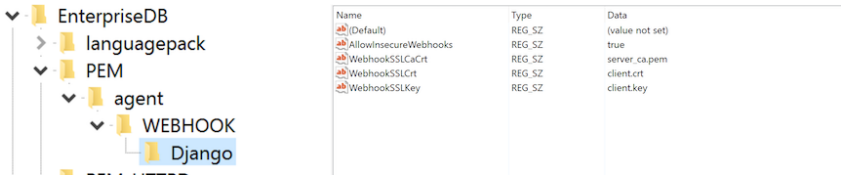


Fig. 9: Example - Webhook SSL Parameters in windows registry

Defining webhook headers

Use the **HTTP Headers** tab to define the header parameters to pass while calling the webhook endpoints:

- Specify all the values as a key and value pair.
- Specify a key parameter in the **Key** field and a value in the **Value** field.
- To add HTTP headers, select the plus sign (+) in the upper-right corner of the HTTP Headers table.
- To delete HTTP headers, select **Delete** to the left of **Key**. The header remains in the list but in strike-through font. Select **Save** to permanently delete the headers.
- To edit the HTTP headers, select **Edit** to the left of **Key**.

Defining webhook payloads

Use the **Payload** tab to define the JSON data to send to the endpoint when an alert is triggered:

- **Type** specifies data to send in the format type, that is, JSON.
- Use **Template** to configure JSON data sent to endpoints. In the template, you can use placeholders for the following:
- `%AlertID%` — The id of the triggered alert.
- `%AlertName%` — The name of the triggered alert.
- `%ObjectName%` — The name of the server or agent on which the alert was triggered.
- `%ObjectType%` — The type on which the alert was generated.
- `%ThresholdValue%` — The threshold value reached by the metric when the alert triggered.
- `%CurrentValue%` — The current value of the metric that triggered the alert.
- `%CurrentState%` — The current state of the alert.
- `%OldState%` — The previous state of the alert.
- `%AlertRaisedTime%` — The time that the alert was raised or the most recent time that the alert state changed.
- `%AgentID%` — The id of the agent by which the alert was generated.

- `%AgentName%` — The name of the agent by which the alert was generated.
- `%ServerID%` — The id of the server on which the alert was generated.
- `%ServerName%` — The name of the server on which the alert was generated.
- `%ServerIP%` — The ip or address of the server on which the alert was generated.
- `%ServerPort%` — The the port of the server on which the alert was generated.
- `%DatabaseName%` — The name of the database on which the alert was generated.
- `%SchemaName%` — The name of the schema on which the alert was generated.
- `%PackageName%` — The name of the package on which the alert was generated.
- `%DatabaseObjectName%` — The name of the database object, like table name or function name, on which the alert was generated.
- `%Parameters%` — The list of custom parameters used to generate the alert.
- `%AlertInfo%` — The detailed database object-level information of the alert.
- Select **Test Connection** to test notification delivery to the mentioned endpoint.

Defining webhook alert levels

Use the **Notifications** tab to specify an alert level for webhook endpoints:

- Set **All alerts** to **Yes** to enable all alert levels to send notifications.
- To send a notification when a specific alert level is reached, set the slider next to an alert level to **Yes**. You must set **All alerts** to **No** to configure an individual alert level.

Example: Sending notifications to Slack

In Slack, follow the instructions in Getting started with incoming webhooks to:

- Create a Slack app.
- Activate incoming webhooks for that app.
- Add a webhook that posts to a channel or user of your choice.

The new webhook has a unique URL similar to `https://hooks.slack.com/services/x/y/z` . You can now configure PEM to send notifications to this URL.

In PEM, *Creating a webhook* , give it a descriptive name, and copy the URL you obtained earlier to the **URL** field. Ensure that **Request method** is set to **POST** and **Enable?** is set to **Yes**. Set all the sliders under **Alert Notifications** to **Yes**.

Add a header under HTTP headers with the key `Content-Type` and the value `application/json` .

Under **Payload**, delete the default template and specify a template with `text` as the top-level key as in the following example:

```
{"text": "%AlertName% on %ObjectType% %ObjectName% is now %CurrentState%"}
```

You can now test the connection. If it succeeds, PEM issues a notification, and the template you specified appears in your Slack channel as a message.

Save the webhook and continue using PEM as usual. PEM now sends all the alerts to your Slack channel.

Deleting a webhook

To mark a webhook for deletion, in the Webhooks table, select the webhook name and select **Delete** to the left of the name. The alert remains in the list but in strikethrough font.

Delete is a toggle. You can undo the deletion by selecting **Delete** a second time. Select **Save** to permanently delete the webhook definition.

SNMP

You must configure the PEM server to send the notifications to an SNMP trap/notification receiver before notifications can be sent. Set the SNMP ver to v1, v2, or v3 to identify the SNMP version.

Example - Configure `SNMP V3` traps with `net-snmp` trap receiver

1. Set `snmp_security_engine_id` to `PEM_SNMP_AGENT` in plain text format in the Server Configuration dialog box.
2. Convert the plain text value to hexadecimal format to use it in `snmptrapd.conf` file. (You can have hexadecimal values of `snmp_security_engine_id` up to 32 octets length).

```
echo PEM_SNMP_AGENT | hexdump -v -e /1 "%02X"
50454D5F534E4D505F4147454E540A
```

1. Set the following parameters in the Server Configuration dialog box:

- **snmp_security_nam** to **pem_snmp_user**
- **snmp_authentication_protocol** to **MD5**
- **snmp_authentication_password** to **pem_auth_pass**
- **snmp_privacy_protocol** to **DES**
- **snmp_privacy_password** to **pem_priv_pass**

1. The `snmptrapd.conf` file has the following values:

```
createUser -e 0x50454D5F534E4D505F4147454E540A pem_snmp_user MD5 pem_auth_pass DES pem_
↪priv_pass
authUser log pem_snmp_user
```

!!!note The agent must run as root to allow sending notifications with SNMP V3.

Using PEM with Nagios

The PEM server can send a passive alert result to Nagios network-alerting software when a user-defined alert is triggered. To instruct the PEM server to notify Nagios of a triggered alert, you must:

- Enable Nagios notification for each alert that triggers a notification from the PEM server to Nagios. You must configure PEM alerting before you create the `host.cfg` and `services.cfg` files.
- Configure Nagios-related behaviors of the PEM server.
- Create the `host.cfg` and `services.cfg` configuration files.
- If necessary, modify the Nagios configuration file and restart the Nagios server.

- Install the PEM agent on the system where the Nagios server is installed and register it with the PEM Server. Set `enable_nagios` configuration to `true` in the `agent.cfg` file for that agent, and restart the agent service.

After configuring the server to enable Nagios alerting, any triggered alerts send a passive check result to the Nagios service. The syntax of a passive alert is:

```
<timestamp> PROCESS_SERVICE_CHECK_RESULT; <host_name> ; <service_name> ; <service_status>
;
```

Where:

`timestamp` is the date and time that the alert was triggered.

`host_name` is the name of the server or agent.

`service_name` is the name of the alert.

`service_status` is the numeric service status value:

- 0 if the service status is OK
- 1 if the service status is WARNING
- 2 if the service status is CRITICAL
- 3 if the service status is UNKNOWN

The PEM server uses the following rules to evaluate the service status:

- If the PEM alert level is CLEARED, the warning message reads OK.
- If the PEM alert level is LOW, the warning message reads WARNING.
- If the `is_nagios_medium_alert_as_critical` flag (specified in the PEM server configuration dialog box) is set to FALSE and the alert level MEDIUM, the warning message reads WARNING.
- If the `is_nagios_medium_alert_as_critical` flag (specified in the PEM server configuration dialog box) is set to TRUE and the alert level is MEDIUM, the warning message reads CRITICAL.
- If the PEM alert level is HIGH, the warning message reads CRITICAL.

Enabling Nagios notification for an alert

The PEM server maintains a unique set of notification properties for each enabled alert. Use the **Notification** tab of the **Manage Alerts** tab to specify that, when triggered, a given alert sends an alert notice to Nagios.

To modify the notification properties of an alert, right-click the name of the object monitored by the alert, and select **Management > Manage Alerts**. On the **Manage Alerts** tab, select **Edit** to the left of the alert name in the Alerts list. When the edit pane opens, select the **Notification** tab.

To enable Nagios notification, move the slider next to **Submit passive service check result to Nagios** to **Yes**. Then select **Save**.

Configuring Nagios-related behavior of the PEM Server

You can use the Server Configuration dialog box to provide information about your Nagios configuration to the PEM server. To open dialog box, select **Management > Server Configuration**.

Four server configuration parameters specify information about your Nagios installation and PEM server behavior related to Nagios:

- Use the `nagios_cmd_file_name` parameter to specify the location of the Nagios pipeline file that receives passive check alerts from PEM. The default value of this parameter is `/usr/local/nagios/var/rw/nagios.cmd` . If your `nagios.cmd` file resides elsewhere, specify the file location in the **Value** field.
- Move the slider in the `nagios_enabled` parameter to **Yes** to instruct the PEM server to send passive check alerts to Nagios.
- Use the `nagios_medium_alert_as_critical` slider to specify the warning severity that the PEM server passes to Nagios if a medium alert is triggered:
- If the `is_nagios_medium_alert_as_critical` flag is set to **FALSE** and the alert level is **MEDIUM**, the warning message reads WARNING.
- If the `is_nagios_medium_alert_as_critical` flag is set to **TRUE** and the alert level is **MEDIUM**, the warning message reads CRITICAL.
- Use the `nagios_spool_retention_time` parameter to specify the number of days of notification history to store on the PEM server. The default value is 7 days.

After modifying parameter values, select **Save** in the upper-right corner of the dialog box.

Creating the hosts.cfg and services.cfg file

The `templates.cfg` file (by default, located in `/usr/local/nagios/etc/objects`) specifies the properties of a generic host and generic service. The properties specify the parameters used in the `hosts.cfg` and `services.cfg` files.

In most cases (when PEM is installed in a default configuration), you don't need to modify the `templates.cfg` file before creating the `hosts.cfg` and `services.cfg` files. If necessary, you can modify the `templates.cfg` file to specify alternative values for parameters or to create new templates.

Before modifying the Nagios configuration file, use the following command to create a `hosts.cfg` file that contains information about the PEM hosts that reside on the local system:

```
psql -U postgres -p 5433 -d pem -A -t -c "select pem.create_nagios_host_config('generic-host')"
> /usr/local/nagios/etc/objects/hosts.cfg
```

Then, use the following command to create a `services.cfg` file that contains information about the PEM services that reside on the local system:

```
psql -U postgres -p 5433 -d pem -A -t -c "select pem.create_nagios_service_config('generic-service')"
> /usr/local/nagios/etc/objects/services.cfg
```

If you want to use a `custom template.cfg` file entry, specify the entry name in place of `generic-host` or `generic-service` in these commands.

Modifying the Nagios configuration file

After creating the `host.cfg` and `services.cfg` files, you must specify their location in the Nagios configuration file (by default, `/usr/local/nagios/etc/nagios.cfg`). Modify the configuration file, adding entries that specify the location of the files:

```
cfg_file=/usr/local/etc/objects/hosts.cfg
```

```
cfg_file=/usr/local/etc/objects/services.cfg
```

You can use the following command to confirm that Nagios is properly configured:

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

After confirming that Nagios is configured correctly, restart the Nagios service:

```
/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

Customizing email templates

PEM monitors your system for conditions that require user attention and sends email notifications. Use **Email Templates** to customize the email subject and the payload that the server sends if the current values deviate from the threshold values specified by an alert.

These are the default email templates that are customizable:

- Alert detected
- Alert level increased
- Alert level decreased
- Alert cleared
- Alert reminder
- Job success
- Job failure
- Job cancellation
- Job step
- Job step (Database server)

To customize the email template, select the edit button next to the email template name. As needed, edit the **Subject** and **Payload** fields from the available list of placeholders. After modifying the field values, select **Save**.

If any of the default email templates are customized, then a green tick mark is displayed for that template in the `Custom Template?` column on the **Email Templates** tab.

To restore the customized email template as the default template, select the custom template and select the undo button at the top of the **Email Templates** tab.

5.3.4 Capacity Manager

PEM's Capacity Manager analyzes collected statistics (metrics) to generate a graph or table that displays the historical usage statistics of an object. It can also project the anticipated usage statistics for an object. You can configure Capacity Manager to collect and analyze metrics for a specific host, server, database, or database object.

You can tailor the content of the Capacity Manager report by choosing:

- Specific metrics to include in the report
- The time range over which the metrics were gathered
- A high or low threshold for the metrics analyzed

You can also specify a start and end date for the Capacity Manager report. If the end date of the report specifies a time in the future, Capacity Manager analyzes the historical usage of the selected object to extrapolate the projected object usage in the future.

To open Capacity Manager, in the PEM client, select **Management > Capacity Manager**. The Capacity Manager wizard opens, displaying a tree on the **Metrics** tab. On the **Metrics** tab, expand the tree to review the metrics for the node that you want to analyze. Select the metric to include it in your report.

Capacity Manager uses the aggregation method specified with the **Aggregation** list. Capacity Manager uses the aggregation method to evaluate and plot the metric values. Select from:

- **Average** — Use the average of the values recorded during the time period.
- **Maximum** — Use the maximum value recorded during the time period.
- **Minimum** — Use the minimum value recorded during the time period.
- **First** — Use the first value recorded during the time period.

To remove a metric from the Capacity Manager report, clear the box to the left of the name of a metric.

Move the slider next to **Graph/chart metrics individually?** to **Yes** to produce a separate report for each metric selected on the **Metrics** tab. If the option is set to **No**, all selected metrics are merged into a single graph or table.

Select **Generate** to display the report onscreen (accepting the default configuration options), or use the **Options** tab to customize sampling boundaries, report type, and report destination. The times displayed on the **Options** tab are from the time zone where the PEM client resides.

Use the fields in the **Time Period** box to define the boundaries of the Capacity Manager report:

- Use the **Period** list to select the type of time period to use for the report. You can select:

Start time and end time	Specify a start date and an end date/time for the report
Start time and threshold	Specify a start date and time and a threshold to determine the end time and date for the report.
Historical days and extrapolated days	Specify a start date for the report that is a number of days in the past and an end date that is a number of days in the future. This option is useful for report templates that don't specify fixed dates.
Historical days and threshold	Specify a start date that is a number of days in the past, and end it when a threshold value is reached.

After specifying the type of time period for the report, select from other options in the **Time Period** box to define the time period for the report:

- Use the date and time selectors next to the **Start time** field to specify the starting date and time of the sampling period. Or select the number of historical days of data to include in the report. The date and time specified in the **Start time** field can't be later than the current date/time.

By default, Capacity Manager selects a start time one week prior to the current date and time.

- The end boundary for the report can be a time, a number of days in the future, or the point at which a selected metric reaches a specified threshold value. Use the date and time selectors next to **End time** to specify an end boundary for the report. Or select the number of extrapolated days of data to include in the report. The time specified in the **End time** field must be later than the time specified in the **Start time** field.

If you select an end date and time in the future, Capacity Manager uses historical usage information to extrapolate anticipated future usage. Since the projected usage is based on the sampling of historical data, the accuracy of the future usage trend improves with a longer sampling period.

To specify a threshold value, use the list in the **Threshold** field to select a metric, an operator (**Exceeds** or **Falls below**), and a target value for the metric. If you choose to define the end of the report using a threshold, the Capacity Manager report terminates when the value for the selected metric exceeds or falls below the specified value.

The `cm_max_end_date_in_years` configuration parameter defines a default time value for the end boundary of a Capacity Manager report. If you specify a threshold value as the end boundary of a report and the anticipated usage of the boundary isn't met before the maximum time passes, the report terminates at the time specified by the `cm_max_date_in_years` parameter. By default, `cm_max_end_date_in_years` is 5. You can use the Server Configuration dialog box to modify the value of `cm_max_end_date_in_years` .

The fields in the **Report** box specify the report type and destination. Use the **Include on report** options to specify the type of report produced by Capacity Manager:

- Select **Graph** to display the report in the form of a line graph in the PEM client window.
- Select **Table of data** to display a table containing the report data in the PEM client window.
- Select **Graph and table of data** to display both a line graph and a data table in the PEM client window.

Use the **Report destination** options to set where to display or save the report:

- Select **New tab** to display the report on a new tab in the PEM client. You must select **New tab** to display the first generation of a Capacity Manager report. For subsequent reports, you can select **Previous tab**.
- Select **Previous tab** to reuse a previously opened tab when displaying the report.
- Select **Download the report as a file** and specify a file name to write the report to the specified file.
- For **Report Type?**, select **HTML** or **JSON** type (version 9.6 and later).

After you specify the report boundaries and select the type and destination of the Capacity Manager report, select **Generate** to create the report.

You can review a Capacity Manager HTML report with any web browser that supports scalable vector graphics (SVG). Browsers that don't support SVG can't display a Capacity Manager graph and might include unwanted characters.

Capacity Manager templates

After defining a report, you can save the definition as a template for future reports. All PEM users can access Capacity Manager report templates. To save a report definition as a template:

1. Use the **Metrics** and **Options** tabs to define your report.
2. Select **Save**.
3. In the **Save Template** dialog box, Provide a report name in the **Title** field and select a location to store the template in the tree.
4. Select **OK**.

When creating a report, you can use **Load Template** to browse and open an existing template. Once opened, you can modify the report definition and save it again, either as a new template or by overwriting the original template.

Select **Manage Templates** to open a dialog box that lets rename or remove unwanted templates.

Capacity Manager metrics - reference

The Capacity Manager metrics available vary by platform and are subject to change. The available metrics can include the metrics described in the table.

Metric name	Description
# Dead Tuples	The number of dead tuples in the selected table.
# Dead Tuples+	The cumulative number of dead tuples in the selected table.
# Heap Tuples Fetched by Index Scans	The number of heap tuples fetched by index scans.
# Heap Tuples Fetched by Index Scans	The cumulative number of heap tuples fetched by index scans.
# Idle Backends+	The cumulative number of currently idle backend clients.
# Index Scans	The number of index scans performed on the specified object.
# Index Scans+	The cumulative number of index scans performed on the specified object.
# Index Tuples Read	The number of index tuples read.
# Index Tuples Read+	The cumulative number of index tuples read.
# Live Tuples	The number of tuples visible to transactions.
# Live Tuples+	The cumulative number of tuples visible to transactions.
# Pages Estimated by ANALYZE	The number of pages estimated by ANALYZE.
# Pages Estimated by ANALYZE+	The cumulative number of pages estimated by ANALYZE.
# Sequential Scans	The number of sequential scans performed on the specific table.
# Sequential Scans+	The cumulative number of sequential scans performed on the specific table.
# Sequential Scan Tuples	The number of tuples sequentially scanned in the specific table.
# Sequential Scan Tuples+	The cumulative number of tuples sequentially scanned in the specific table.
# Tuples Deleted	The number of tuples deleted.
# Tuples Deleted+	The cumulative number of tuples deleted.
# Tuples Estimated by ANALYZE	The number of live (visible) tuples estimated by ANALYZE.
# Tuples Estimated by ANALYZE+	The cumulative number of live tuples estimated by ANALYZE.
# Tuples HOT Updated	The number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.

continues on next page

Table 23 – continued from previous page

Metric name	Description
# Tuples HOT Updated+	The cumulative number of tuples HOT updated.
# Tuples Inserted	The number of tuples inserted into the specified table.
# Tuples Inserted+	The cumulative number of tuples inserted into the specified table.
# Tuples Updated	The number of tuples updated in the selected table.
# Tuples Updated+	The cumulative number of tuples updated in the selected table.
Blocks Hit	The number of blocks found in the cache.
Blocks Hit+	The cumulative number of blocks found in the cache.
Blocks Read	The number of blocks read.
Blocks Read+	The cumulative number of blocks read.
Blocks Read from InfiniteCache	The number of blocks read from InfiniteCache.
Blocks Read from InfiniteCache+	The cumulative number of blocks read from InfiniteCache.
Blocks Written	The number of blocks written.
Blocks Written+	The cumulative number of blocks written.
Buffers Allocated	The number of buffers allocated.
Buffers Allocated+	The cumulative number of buffers allocated.
Buffers Written - Backends	The number of buffer blocks written to disk by server processe (processes connected to a client application).
Buffers Written - Backends+	The cumulative number of buffer blocks written to disk by server processes.
Buffers Written - Checkpoint	The number of blocks written to disk by the checkpoint process.
Buffers Written - Checkpoint+	The cumulative number of blocks written to disk by the checkpoint process.
Buffers Written - Cleaning Scan	The number of blocks written to disk by the autovacuum process.
Buffers Written - Cleaning Scan+	The cumulative number of blocks written to disk by the autovacuum process.
Bytes Received (KB)	The number of bytes received from the client (in kilobytes).
Bytes Received (KB)+	The cumulative number of bytes received (in kilobytes).
Bytes Sent (KB)	The number of bytes sent to the client (in kilobytes).
Bytes Sent (KB)+	The cumulative number of bytes sent (in kilobytes).
Checkpoints - Timed	The number of checkpoint operations triggered by the checkpoint interval.
Checkpoints - Timed+	The cumulative number of checkpoint operations triggered by the checkpoint interval.
Checkpoints - Untimed	The number of checkpoint operations triggered by checkpoint size.
Checkpoints - Untimed+	The cumulative number of checkpoint operations triggered by checkpoint size.
Database Size (MB)	The size of the specified database (in megabytes).
Free RAM Memory	The amount of free RAM memory (in megabytes).
Free Swap Memory	The amount of free swap space on disk (in megabytes).
Heap Blocks Hit	The number of heap blocks found in the cache.
Heap Blocks Hit+	The cumulative number of heap blocks found in the cache.
Heap Blocks Read	The number of heap blocks read.
Heap Blocks Read+	The cumulative number of heap blocks read.
Index Blocks Hit	The number of index blocks found in the cache.
Index Blocks Hit+	The cumulative number of index blocks found in the cache.
Index Blocks Read	The number of index blocks read.
Index Blocks Read+	The cumulative number of index blocks read.
Index Size (MB)	The size of the specified index (in megabytes).
In Packets Discards	The number of inbound packets discarded.

continues on next page

Table 23 – continued from previous page

Metric name	Description
In Packets Discards+	The cumulative number of inbound packets discarded.
In Packets Errors	The number of inbound packets that contain errors.
In Packets Errors+	The cumulative number of inbound packets that contain errors.
Link Bandwidth (Mbit/s)	The speed of the network adapter (in megabits per second).
Load Average - 15 Minute	CPU saturation (in percent) - 15 minute sampling average.
Load Average - 1 Minute	CPU saturation (in percent) - 1 minute sampling average.
Load Average - 5 Minute	CPU saturation (in percent) - 5 minute sampling average.
Load Percentage	CPU saturation in percent.
Number of Prepared Transactions+	The cumulative number of prepared transactions.
Number of WAL Files+	The cumulative number of write-ahead log files.
Out Packets Discards	The number of outbound packets discarded.
Out Packets Discards+	The cumulative number of outbound packets discarded.
Out Packets Errors	The number of outbound packets that contain errors.
Out Packets Errors+	The cumulative number of outbound packets that contain errors.
Packets Received	The number of packets received.
Packets Received+	The cumulative number of packets received.
Packets Sent	The number of packets sent.
Packets Sent+	The cumulative number of packets sent.
Size (MB)	The total size of the disk (in megabytes).
Size of Indexes (MB)	The size of indexes on the specified table (in megabytes).
Space Available (MB)	The current disk space available (in megabytes).
Space Used (MB)	The current disk space used (in megabytes).
Table Size (MB)	The size of the specified table (in megabytes).
Tablespace Size (MB)	The size of the specified tablespace (in megabytes).
Temp Buffers (MB)	The size of temporary buffers (in megabytes).
Toast Blocks Hit	The number of TOAST blocks found in the cache.
Toast Blocks Hit+	The cumulative number of TOAST blocks found in the cache.
Toast Blocks Read	The number of TOAST blocks read.
Toast Blocks Read+	The cumulative number of TOAST blocks read.
Total RAM Memory	The total amount of RAM memory on the system (in megabytes).
Total Swap Memory	The total amount of swap space on the system (in megabytes).
Total Table Size w/Indexes and Toast	The total size of the specified table (including indexes and associated oversized attributes).
Transactions Aborted	The number of aborted transactions.
Transactions Aborted+	The cumulative number of aborted transactions.
Transactions Committed	The number of committed transactions.
Transactions Committed+	The cumulative number of committed transactions.
Tuples Deleted	The number of tuples deleted from the specified table.
Tuples Deleted+	The cumulative number of tuples deleted from the specified table.
Tuples Estimated by ANALYZE	The number of visible tuples in the specified table.
Tuples Estimated by ANALYZE+	The cumulative number of visible tuples in the specified table.
Tuples Fetched	The number of tuples fetched from the specified table.
Tuples Fetched+	The cumulative number of tuples fetched from the specified table.
Tuples HOT Updated	The number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.
Tuples HOT Updated+	The cumulative number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.

continues on next page

Table 23 – continued from previous page

Metric name	Description
Tuples Inserted	The number of tuples inserted into the specified table.
Tuples Inserted+	The cumulative number of tuples inserted into the specified table.
Tuples Returned	The number of tuples returned in result sets.
Tuples Returned+	The cumulative number of tuples returned in result sets.
Tuples Updated	The number of tuples updated in the specified table.
Tuples Updated+	The cumulative number of tuples updated in the specified table.
WAL Segment Size (MB)	The segment size of the write-ahead log (in megabytes).

Note: The + following the name of a metric means that the data for the metric is gathered cumulatively. Metrics that aren't followed by the '+' sign are collected as a point-in-time value.

5.3.5 Audit Manager

You can use the PEM Audit Manager to simplify audit log configuration for EDB Postgres Advanced Server instances. With Audit Manager, you can configure logging attributes such as:

- How often PEM collects log files
- The type of database activities that are included in the log files
- How often to rotate log files and when

Audit logs can include the following activities:

- All connections made to the database instance
- Failed connection attempts
- Disconnections from the database instance
- All queries (SELECT statements)
- All DML statements (INSERT, UPDATE, DELETE)
- All DDL statements (for example, CREATE, DROP, ALTER)

Once the audit logs are stored on the PEM server, you can use the Audit Log dashboard to review the information in an easy-to-read form. The Audit Log dashboard allows you to filter the log file by:

- Timestamp range (when an activity occurred)
- The database on which the activity occurred
- The user performing the activity
- The type of command being invoked

Setting the EDB Postgres Advanced Server instance service ID

To configure logging for an EDB Postgres Advanced Server instance, the server must be a PEM-managed server with a bound agent, and the server registration must include the name of a service script. When registering a new server, include the service name in the **Service ID** field on the **Advanced** tab of the New Server dialog box.

Before adding a service name to an existing, registered and connected server, you must disconnect the server:

1. Right-click the server name, and select **Disconnect server** from the context menu.
2. Right-click the server name and select **Properties** from the context menu.
3. Select the **Advanced** tab, and add a service name to the **Service ID** field.

The Service ID field allows the PEM server to stop and start the service.

- The name of the Advanced Server 12 service script is `edb-as-12` .
- The name of the Advanced Server 11 service script is `edb-as-11` .
- The name of the Advanced Server 10 service script is `edb-as-10` .

Setting the EDB Audit Configuration probe

Before configuring audit logging of EDB Postgres Advanced Server servers, you must ensure that the EDB Audit Configuration probe is enabled. To open the **Manage Probes** tab and check the status of the probe, right-click the name of a registered EDB Postgres Advanced Server server in the tree, and select **Management > Manage Probes**.

Ensure that the Enabled column in the Probe Configuration dialog box is set to **Yes** for the EDB Audit Configuration probe. If EDB Audit Configuration isn't enabled, use the **Enabled?** switch on the **Manage Probes** tab to enable it.

Configuring audit logging with the Audit Manager

1. To open the Audit Manager wizard, select **Management > Audit Manager**. The Audit Manager - Welcome dialog box opens. Select **Next** to continue.
2. Use the Select Servers tree to specify the servers to which to apply the auditing configuration. To make a server available in the tree, you must provide the Service ID on the **Advanced** tab of the Create Server dialog box when registering a server for monitoring by PEM. Only EDB Postgres Advanced Server supports auditing. PostgreSQL servers don't appear in the tree.

Select **Next** to continue.

1. The Auditing Parameters Configuration dialog box lets you enable or disable auditing and choose how often log records are collected into PEM. Use the Auditing Parameters Configuration dialog box to specify auditing preferences:

- Use the **Auditing** switch to enable or disable auditing on the specified servers.
- Use the **Audit destination** list to select a destination for the audit logs. Select **File** or **Syslog**. This feature is supported only on EDB Postgres Advanced Server 10 and later releases.
- Use the **Import logs to PEM** switch to periodically import log records from each server to the PEM Server. Set the switch to **Yes** to import log files. The default is **No**.
- Use the **Collection frequency** list to specify how often PEM collects log records from monitored servers when log collection is enabled.
- Use the **Log format** list to select the raw log format to write on each server. If log collection is enabled, the PEM server uses CSV format.

- Use the **File name** field to specify the format used when generating log file names. By default, the format is set to `audit-%Y-%m-%d_%H%M%S` where:

`audit` is the file name specified in the Audit Directory Name field `Y` is the year that the log was stored `m` is the month that the log was stored `d` is the day that the log was stored `H` is the hour that the log was stored `M` is the minute that the log was stored `S` is the second that the log was stored

- Select **Change Log Directory for selected servers?** and use the **Audit Directory Name** field to specify a directory name to which to write the audit logs. The directory resides beneath the data directory on the PEM server.
- Use the **Log directory** box to specify information about the directory in which the log files are saved:

```
-   Move the **Change log directory for selected servers?** switch to **Yes** to enable␣
↪the **Directory name** field.

-   Use the **Directory name** field to specify the name of the directory on each server␣
↪to write audit logs to. The directory specified is created as a subdirectory of the␣
↪data directory on the server.
```

Select **Next** to continue.

1. The Audit log configuration dialog box is available only if you enabled auditing on the Auditing Parameters Configuration dialog box. Use the Audit Log Configuration dialog box to specify log configuration details to apply to each server:

- Use the **Connection attempts** switch to specify whether to log connection attempts. Specify:

None to disable connection logging

All to indicate that all connection attempts are logged

Failed to log any connection attempts that fail

- Use the **Disconnection attempts** switch to specify whether to log disconnections. Specify:

None if you don' t want to log disconnections

All to enable disconnection logging

- Use the **Log statements** field to specify the statement types that are logged. Select from:

Select — Log all statements that include the `SELECT` keyword.

Error — Log all statements that result in an error.

DML — Log all DML (data modification anguage) statements.

DDL — Log all DDL (data definition language) statements, that is, those that add, delete or alter data. Select **Select All** to select all statement types.

Select **Unselect All** to clear all statement types.

- Use the **Audit tag** field to specify a tracking tag for the collected logs. Audit tagging functionality is available only for supported versions of EDB Postgres Advanced Server.
- Use the **Log rotation** box to specify how to manage the log files on each server:

Use the **Enable?** switch to rotate log files. Use a new log file periodically to prevent a single file from becoming too large.

Use the **Day** list to select days on which to rotate the log file.

Use the **Size (MB)** field to specify a size in megabytes at which to rotate the log file.

Use the **Time (seconds)** field to specify the number of seconds between log file rotations.

Select **Next** to continue:

1. Use the Schedule Auditing Changes dialog box to determine when auditing configuration changes take effect.

- Select **Configure logging now?** if you want the auditing configuration changes to take place immediately. The affected database servers restart so the auditing changes can take effect.
- Use the **Time?** selector to schedule the auditing configuration changes to take place at some point in the future. Select the desired date and time from the lists. The affected database servers restart at the specified date and time to put the auditing changes into effect.

Select **Finish** to complete the auditing configuration process.

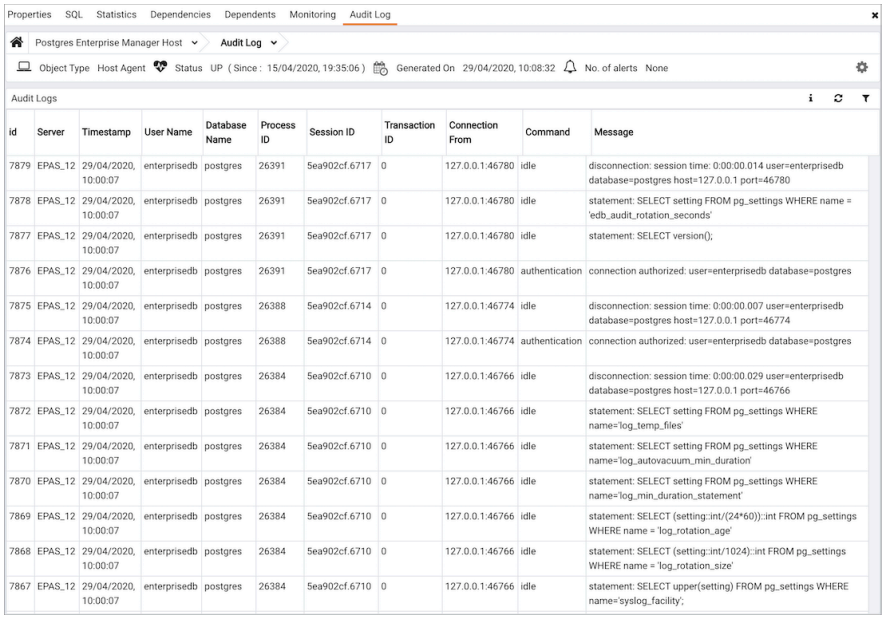
Audit Manager schedules a job to apply the configuration to each server. The job consists of two tasks: one to update the audit logging configuration on the server and one to restart the server with the new configuration.

You can use the **Scheduled Tasks** tab to review a list of scheduled jobs. To open the **Scheduled Tasks** tab, select the name of a server or agent and select **Management > Scheduled Tasks**.

Viewing the log with the Audit Log dashboard

Use the Audit Log dashboard to view the audit log from EDB Postgres Advanced Server database instances.

To open the Audit Log dashboard, right-click a server or agent node and select **Audit Log Analysis** from the **Dashboards** menu. The Audit Log dashboard displays the audit records in reverse chronological order.



id	Server	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
7879	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	idle	disconnection: session time: 0:00:00.014 user=enterprisedb database=postgres host=127.0.0.1 port=46780
7878	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT setting FROM pg_settings WHERE name = 'edb_audit_rotation_seconds'
7877	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT version();
7876	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	authentication	connection authorized: user=enterprisedb database=postgres
7875	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	5ea902cf.6714	0	127.0.0.1:46774	idle	disconnection: session time: 0:00:00.007 user=enterprisedb database=postgres host=127.0.0.1 port=46774
7874	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	5ea902cf.6714	0	127.0.0.1:46774	authentication	connection authorized: user=enterprisedb database=postgres
7873	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	disconnection: session time: 0:00:00.029 user=enterprisedb database=postgres host=127.0.0.1 port=46766
7872	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_temp_files'
7871	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_autovacuum_min_duration'
7870	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_min_duration_statement'
7869	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/(24*60))::int FROM pg_settings WHERE name = 'log_rotation_age'
7868	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/1024)::int FROM pg_settings WHERE name = 'log_rotation_size'
7867	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT upper(setting) FROM pg_settings WHERE name='syslog_facility';

Fig. 10: The Audit Log dashboard

You can use filtering to limit the number of audit records that are displayed. Select **Show Filters** to expose the filters panel. Use the filters panel to provide selection criteria for the audit records you want to display.

- Use the **Start** field to specify a start date for the report.
- Use the **End** field to specify an end date for the report.
- Use the **User** field to display only those entries where the activity was started by the given Postgres user.

- Use the **Database** field to display only those entries where the activity was issued on the given database.
- Use the **Command type** field to display only those entries where the activity was of the given type. Command types you can specify are **idle**, **authentication**, and **SELECT**. (For viewing SQL statements from user applications, specify the idle command type.)

Select **Filter** to apply the filtering criteria to the log entries.

5.3.6 Log Manager

You can use the PEM Log Manager to simplify server log configuration for Postgres instances. With Log Manager, you can modify all of your server log parameters including:

- Where log files are written
- How often log files are written
- The type of information written to log files
- The format of log file entries
- Log rotation properties

Before using Log Manager to define logging properties for a server, you must specify the name of the associated Advanced Server or PostgreSQL database server in the **Service ID** field on the **Advanced** tab of the New Server Registration or Properties dialog box. The server is available for configuration on the Server Selection dialog box only if you specify the name of the service in the **Service ID** field.

For example, suppose you're setting logging preferences for an Advanced Server 9.4 instance that resides on a Linux host. Set the **Service ID** field on the **Advanced** tab of the **Properties** dialog box for the monitored server to `ppas-9.4`.

Note:

- Log Manager depends on Settings and Server Log Configuration probes to populate all the fields in the wizard. Therefore, ensure that those probes for selected servers are enabled. In addition, set the execution frequency for those probes to a minimum to ensure that Log Manager reflects the latest log configurations. - Rerun Log Manager if you make any manual changes related to logging in the configuration files, such as *postgresql.conf* . PEM doesn't reflect those changes automatically.

To configure logging for a Postgres instance, you must register the server as a PEM-managed server, and the registration information must include the name of a service script.

1. To open Log Manager, in the PEM client, select **Management > Log Manager**. The wizard opens and welcome message appears. Select **Next**.
2. The Server Selection dialog box displays a list of the server connections monitored by PEM. Select the check boxes next to the names of servers to which you want the Log Manager wizard to apply the specified configuration. Log Manager is disabled for any server displaying a red exclamation mark to the left of its name in the Server selection tree. A server might not be enabled for several reasons:

- Log Manager can configure only a server that specifies a service ID on the **Advanced** tab of the Properties dialog box. To provide a service ID: 1. In the tree, right-click the server name and select **Disconnect Server** from the context menu. 1. If prompted, provide a password. 1. From the context menu for the server, select **Properties**. 1. On the **Advanced** tab, enter the name of the service in the **Service ID** field. 1. Select **Save**.
- If the PEM agent bound to the server doesn't have enough privileges to restart the server, the server is disabled.
- If the PEM agent bound to the server is an older version than the associated PEM server, the server is disabled.

Select **Next**.

3. Use the Log configuration dialog box to specify how often to import log files to PEM and to specify log rotation details.

Options in the **Import Logs** box specify how often to import the log files to PEM:

- Use the switch next to **Import logs to PEM** to specify whether to import the log files to PEM and to display them on the Server Log Analysis dashboard.
- Use the **Import Frequency** list to specify how often to import log files to PEM.

Use the **Log rotation configuration** box to specify the maximum length (lifespan or size) of a log file:

- Use the **Rotation Size** field to specify the maximum size of a log file in megabytes. The default value is 10 MB. Set to 0 if you don't want to place a limit on the maximum size of a log file.
- Use the **Rotation Time** field to specify the number of whole days to store in each log file. The default value is 1 day.

Use the **Truncation on Rotation** switch to specify server behavior for time-based log file rotation:

- Select **On** to overwrite any existing log file that has the same name as a new file.
- Select **Off** to append any new log file entries to an existing log file with the same name as a new log file. This is the default behavior.

Select **Next**.

4. Use the Where to Log dialog box to specify where to write log files:

- Select an option from the **Log Destination** box to specify a destination for the server log output: - Set the **stderr** switch to **Yes** to write log files to stderr. - Set the **csvlog** switch to **Yes** to write log files in a comma-separated value format. This option is enabled and not editable if you select **Import logs to PEM** on the Schedule dialog box. If you aren't importing server log files to PEM, you can edit this option. - Set the **syslog** switch to **Yes** to write log files to the system log files. - On Windows, set the **eventlog** switch to **Yes** to write log files to the event log.
- Use the **Log collection** box to specify your collection preferences: - Set the **Log Collector** switch to **Enable** to redirect captured log messages (directed to stderr) into log files. - Set the **Log Silent Mode** switch to **Enable** to run the server silently in the background, disassociated from the controlling terminal.
- Use the **Log Directory** box to specify log file location preferences: - Set the **Change log directory for selected servers?** switch to **Yes** to maintain each set of log files in a separate directory. - Use the **Directory name** field to specify the directory to which to write log files. The directory resides beneath the `pg_log` directory under the installation directory of the monitored server.
- Use the **Log File Name** field to specify a format for the log file name. If set to **Default**, the format is `enterprisedb-%Y-%m-%d_%H%M%S` , where: - `enterprisedb` is the file name prefix - `Y` is the year that the log was stored - `m` is the month that the log was store - `d` is the day that the log was stored - `H` is the hour that the log was stored - `M` is the minute that the log was store - `S` is the second that the log was stored
- When logging to syslog is enabled: - Use the **Syslog Facility** list to specify the syslog facility to use. - Use the **Syslog Ident** field to specify the program name that identifies EDB Postgres Advanced Server entries in system logs.

Select **Next**.

5. Use the When to Log dialog box to specify the events that initiate a log file entry.

- The severity levels in order of severity, from most severe to least severe, are:
  - **panic** — Errors that cause all database sessions to abort.
  - **fatal** — Errors that cause a session to abort.

  - **log** — Information messages of interest to administrators.
  - **error** — Errors that cause a command to abort.
  - **warning** — Error conditions in which a command completes but might not perform as expected.
  - **notice** — Items of interest to users. This is the default.
  - **info** — Information implicitly requested by the user.
  - **debug5** through **debug1** — Detailed debugging information useful to developers.
- Use the **Client min messages** list to specify the lowest severity level of message sent to the client application.
- Use the **Log min messages** list to specify the lowest severity level to write to the server log.
- By default, when an error message is written to the server log, the text of the SQL statement that initiated the log entry isn't included. Use the **Log min error statement** list to specify a severity level that triggers SQL statement logging. If a message is of the specified severity or higher, the SQL statement that produced the message is written to the server log.
- Use the **Log min duration statement** list to specify a statement duration, in milliseconds. Any statements that exceed the specified number of milliseconds are written to the server log. A value of **-1** disables all duration-based logging. A value of **0** logs all statements and their duration.
- Use the **Log temp files** field to specify a file size, in kilobytes. When a temporary file reaches the specified size, it's logged. Specify a value of **-1** (the default) to disable this functionality.
- Use the **Log autoVacuum min duration** field to specify a time length, in milliseconds. If auto-vacuuming exceeds the length of time specified, the activity is logged. Specify a value of **-1** (the default) to disable this functionality.

Select **Next**.

6. Use the What to Log dialog box to specify log entry options that are useful for debugging and auditing.

Use the switches in the **Debug options** box to include information in the log files related to query execution that might be of interest to a developer:

- Set the **Parse tree** switch to **Yes** to include the parse tree in the log file.
- Set the **Rewriter output** switch to **Yes** to include query rewriter output in the log file.
- Set the **Execution plan** switch to **Yes** to include the execution plan for each executed query in the log file.

When the **Indent Debug Options Output in Log** switch is set to **Yes**, the server indents each line that contains a parse tree entry, a query rewriter entry, or query execution plan entry. While indentation makes the resulting log file more readable, it results in a longer log file.

Use the switches in the **General Options** box to include auditing information in the log file:

- Set the **Checkpoints** switch to **Yes** to include checkpoints and restartpoints in the server log.
- Set the **Connections** switch to **Yes** to include each attempted connection to the server as well as successfully authenticated connections in the server log.
- Set the **Disconnections** switch to **Yes** to include a server log entry for each terminated session that provides the session information and session duration.
- Set the **Duration** switch to **Yes** to include the amount of time required to execute each logged statement in the server log.
- Set the **Hostname** switch to **Yes** to include both the IP address and host name in each server log entry. By default, only the IP address is logged. This might cause a performance penalty.

- Set the **Lock Waits** switch to **Yes** to write a log entry for any session that waits longer than the time specified in the `deadlock_timeout` parameter to acquire a lock. This is useful when trying to determine if lock waits are the cause of poor performance.

Use the **Error verbosity** list to specify the detail written to each entry in the server log:

- Select **default** to include the error message, DETAIL, HINT, QUERY, and CONTEXT in each server log entry.
- Select **terse** to log only the error message.
- Select **verbose** to include the error message, the DETAIL, HINT, QUERY, and CONTEXT error information, SQLSTATE error code and source code file name, the function name, and the line number that generated the error.

Use the **Prefix string** field to specify a printf-style string that is written at the beginning of each log file entry. For information about the options supported, see the log_line_prefix documentation in the Postgres core documentation .

Use the **Statements** list to specify the SQL statements to include in the server log:

- Specify **none** (the default) to disable logging of SQL statements.
- Specify **ddl** to instruct to log ddl (data definition language) statements, such as CREATE, ALTER, and DROP.
- Specify **mod** to log all ddl statements as well as all dml (data modification language) statements, such as INSERT, UPDATE, DELETE, TRUNCATE, and COPY FROM.
- Specify **all** to log all SQL statements.

Select **Next**.

7. Use the Schedule Logging Changes dialog box to specify when logging applies configuration changes:

- Set the **Configure logging now** switch to **Yes** to enable your configuration preferences. The server restarts when you complete the Log Manager wizard.
- Set **Configure logging now** to **No** to use the **Schedule it for some other time** calendar selector to specify a convenient time to apply logging configuration preferences and for the server to restart.

When you apply the configuration changes specified by the Log Manager wizard, the server restarts, temporarily interrupting use of the database server for users.

8. Select **Finish** to exit the wizard. Either restart the server or schedule the server restart for the time specified on the scheduling dialog box.

Reviewing the Server Log Analysis dashboard

After invoking the Log Manager wizard and importing your log files to PEM, you can use the Server Log Analysis dashboard to review the log files for a selected server. To open the Server Log Analysis dashboard, right-click the name of a monitored server in the PEM client tree and select **Dashboards > Server Log Analysis**.

The header information on the Server Log Analysis dashboard displays the date and time that the server was started, the date and time that the page was last updated, and the current number of triggered alerts.

Entries in the Server Log table appear in chronological order, with the most recent log entries first. Use the scroll bars to navigate through the log entries or to view columns that are off of the display.

Headings at the top of the server log table identify the information stored in each column. Hover over a column heading to view a tooltip that contains a description of the content of each column.

You can use filtering to limit the number of server log records that are displayed. Select **Show Filters** to open the filters panel and define a filter. Use the **filter definition** box to describe the selection criteria for selecting a subset of a report to display:

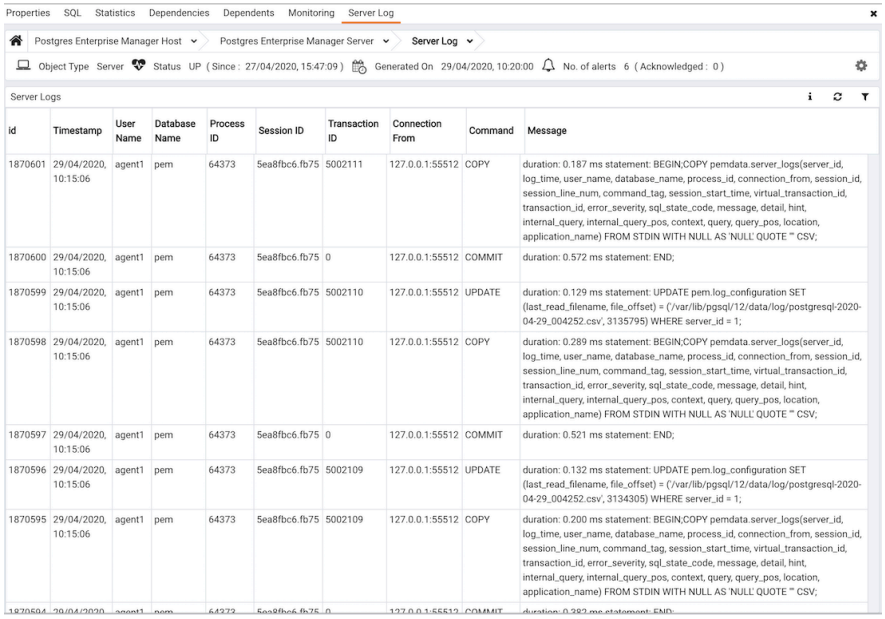


Fig. 11: The Server Log Analysis dashboard

- Use the **From** field to specify a starting date for the displayed server log.
- Use the **To** field to specify an ending date for the displayed server log.
- Enter a role name in the **Username** field to display transactions performed only by that user.
- Enter a database name in the **Database** field to limit the displayed records to transactions that were performed against the specified database.
- Use the **Command Type** field to specify selection criteria for the commands that appear in the filtered report.

After you specify the criteria for filtering the server logs, select **Filter** to display the filtered server log in the Server Log table.

Postgres Log Analysis Expert

The PEM Log Analysis Expert analyzes the log files of servers that are registered with Postgres Enterprise Manager and produces an analysis of your Postgres cluster's usage based on log file entries. You can use information on the Log Analysis Expert reports to make decisions about optimizing your cluster usage and configuration to improve performance.

Before using the PEM Log Analysis Expert, you must specify a **Service ID** value on the **Advanced** tab of the Server Properties dialog box. You must also use the Log Manager wizard to enable log collection by the PEM server.

1. To open the Postgres Log Analysis Expert wizard, in the PEM client, select **Management > Postgres Log Analysis Expert**. In the wizard's Welcome screen, select **Next**.

The wizard's Analyzer selection dialog box displays a list of analyzers from which you can select. Each analyzer generates a corresponding table, chart, or graph that contains information collected from the log files.

2. Select the check box next to an analyzer to specify for the Log Analysis Expert to prepare the corresponding table, chart, or graph. After making your selections, select **Next**.
3. Use the Server selection tree to specify the servers you want the Postgres Log Analysis Expert to analyze. If you select multiple servers, the resulting report contains the corresponding result set for each server in a separate but

continuous list. Select **Next** to continue to the Report options dialog box.

4. Use the **Options** section to specify the analysis method and the maximum length of any resulting tables:

- Use the **Aggregate method** list to select the method used by the Log Analysis Expert to consolidate data for the selected time span: - **SUM** calculates a value that is the sum of the collected values for the specified time span. - **AVG** calculates a value that is the average of the collected values for the specified time span. - **MAX** uses the maximum value that occurs in a specified time span. - **MIN** uses the minimum value that occurs in a specified time span.
- Use the **Time span** field to specify the number of minutes that the analyzer incorporates into each calculation for a point on a graph. For example, if the time span is five minutes, and the aggregate method is **AVG**, each point on the given graph contains the average value of the activity that occurred in a five-minute time span.
- Use the **Rows limit** field to specify the maximum number of rows to include in a table.

5. Use the **Time Intervals** section to specify the time range for the Log Analysis Expert to analyze:

    - Set **Relative days** to **Yes** to enable the **(+/-) From date** field and specify the number of days before or after the date and time selected in the **From** field.
    - Use the **From** field to specify the starting date and time for the analysis.
    - Use the **To** field to specify the ending date and time for the analysis.
    - Use the **(+/-) From date** selector to specify the number of days before or after the **From** date to include in the analysis.

After you specify the report options, select **Next** to continue to the Report destination dialog box.

6. You can select the default option and select **Finish** to view the Log Analysis Expert report in the PEM client's tabbed browser. Or select **Download the report** to save a copy of the report to an HTML file for later use.

Reviewing the Postgres Log Analysis Expert report

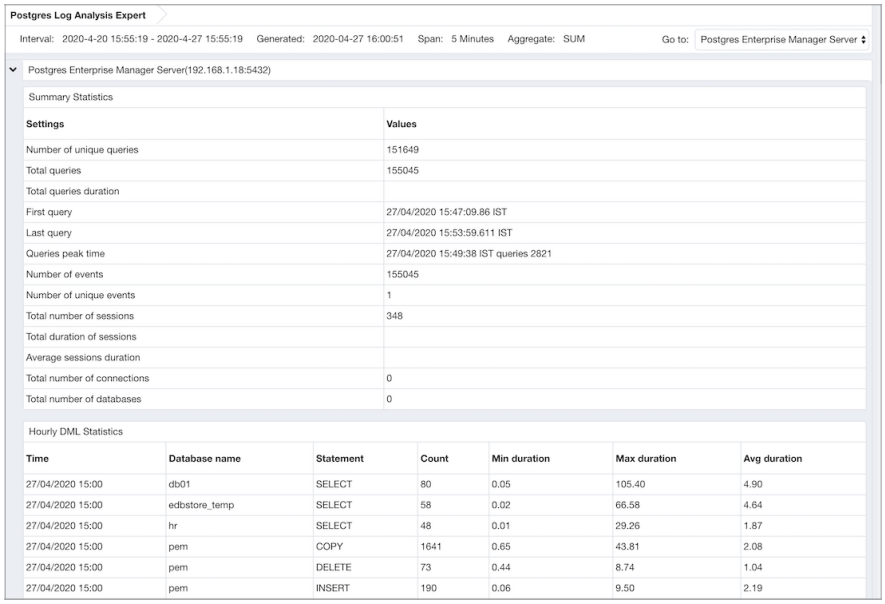
If you choose to review the report immediately, the Postgres Log Analysis Expert report is displayed in the PEM Client window. The report header displays:

- The date and time that the report was generated
- The time period that the report spans
- The aggregation method specified when defining the report

The name of the server for which information is displayed appears at the start of each section of the report.

The report displays the tables, graphs, and charts selected in the Log Analysis Expert wizard. Select **Jump To**, located in the lower-right corner of the screen, to navigate to a specific graphic.

If the report contains an analysis of more than one monitored server, charts and tables are displayed in sets. First the graphs, tables, and charts that display statistics for one server appear. Then the graphics for the next server in the report appear.



Postgres Log Analysis Expert

Interval: 2020-4-20 15:55:19 - 2020-4-27 15:55:19 Generated: 2020-04-27 16:00:51 Span: 5 Minutes Aggregate: SUM Go to: Postgres Enterprise Manager Server

Postgres Enterprise Manager Server(192.168.1.18:5432)

Summary Statistics

Settings	Values
Number of unique queries	151649
Total queries	155045
Total queries duration	
First query	27/04/2020 15:47:09.86 IST
Last query	27/04/2020 15:53:59.611 IST
Queries peak time	27/04/2020 15:49:38 IST queries 2821
Number of events	155045
Number of unique events	1
Total number of sessions	348
Total duration of sessions	
Average sessions duration	
Total number of connections	0
Total number of databases	0

Hourly DML Statistics

Time	Database name	Statement	Count	Min duration	Max duration	Avg duration
27/04/2020 15:00	db01	SELECT	80	0.05	105.40	4.90
27/04/2020 15:00	edbstore_temp	SELECT	58	0.02	66.58	4.64
27/04/2020 15:00	hr	SELECT	48	0.01	29.26	1.87
27/04/2020 15:00	pem	COPY	1641	0.65	43.81	2.08
27/04/2020 15:00	pem	DELETE	73	0.44	8.74	1.04
27/04/2020 15:00	pem	INSERT	190	0.06	9.50	2.19

Fig. 12: The Postgres Log Analysis Expert Report

Summary Statistics table

The Summary Statistics table displays a summary of server activity for the selected server.

Row name	Description
Number of unique queries	Count of unique queries made against the selected server in the specified time period.
Total queries	Count of queries made against the selected server in the specified time period.
Total queries duration	Amount of time used to execute queries against the server.
First query	Time, within the specified time period, that the first query executed against the server.
Last query	Time, within the specified time period, that the last query executed against the server.
Queries peak time	Point in time, within the specified time period, that query activity reached its highest level.
Number of events	Count of log events within the specified time period.
Number of unique events	Count of unique server events.
Total number of sessions	Count of the number of sessions recorded in the time period.
Total duration of sessions	Amount of time that sessions were connected during the specified time period.
Average sessions duration	Average length of each session.
Total number of connections	Number of user connections made to the server.
Total number of databases	Number of databases on the selected server.

Hourly Statistics table

The Hourly DML Statistics table displays the statistics related to the use of various DML commands (`SELECT` , `INSERT` , `UPDATE` , `DELETE` , `COPY` , and `FETCH`) within a one-hour period. To generate values in the Min Duration(sec), Max Duration(sec), and Avg Duration(sec) columns of this table, you must specify a value greater than or equal to 0 in the `log_min_duration_statement` configuration parameter. You can set the parameter either by modifying the `postgresql.conf` file with your editor of choice or by specifying a value of `0` or greater in the **Log Min Duration Statement** field of the Log Manager wizard.

Column name	Description
Time	Start of the one-hour period for which data was analyzed.
Database	Name of the database in which the specified DML command executed.
Command Type	DML command type.
Total Count	Number of times that a command of the specified command type executed during the one-hour period analyzed by the report.
Min Duration(sec)	Shortest amount of time used by the server to respond to the specified command type, in seconds.
Max Duration(sec)	Longest amount of time used by the server to respond to the specified command type, in seconds.
Avg Duration(sec)	Average length of time used by the server when responding to the specified command type, in seconds.

DML Statistics Timeline section

The DML Statistics Timeline section of the Log Analysis Expert report displays information about DML statement usage.

Graphic	Description
Line graph	Analysis of statement usage during the selected time period. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Percent of statement usage of each respective DML statement type during the selected time period.

DDL Statistics Timeline section

The DDL Statistics Timeline section of the Log Analysis Expert report displays information about DDL statement usage.

Graphic	Description
Line graph	Analysis of statement usage during the selected time period. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Percent of statement use of each respective DDL statement type during the selected time period.

Commit and Rollback Statistics Timeline section

The Commit and Rollback Statistics Timeline section of the Log Analysis Expert report displays information about the `COMMIT` , `ROLLBACK` , and `SAVEPOINT` statements logged during the specified time period.

Graphic	Description
Line graph	Analysis of the commit and rollback activity during the specified time period. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Comparative percent of *COMMIT*, *SAVEPOINT*, or *ROLLBACK* statements executed during the specified time period.

Checkpoint Statistics Timeline section

The Checkpoint Statistics Timeline section of the Log Analysis Expert report displays information about the checkpoint operations logged during the specified time period.

Graphic	Description
Line graph	Analysis of the checkpoint operation activity during the specified time period. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Comparative percent of different types of checkpoint activity logged during the specified time period.

Log Event Statistics table

The Log Event Statistics table lists log entries with a severity level of WARNING, ERROR, FATAL, PANIC, HINT, or CONTEXT. The level of logging detail for error messages is controlled by the `log_min_error_statement` parameter. You can set the parameter either by modifying the `postgresql.conf` file with your editor of choice or by specifying a value in the **Log Min Error Statement** field of the Log Manager wizard.

Column name	Description
Error Severity	Severity level of the log entry.
Message	The log message.
Total Count	Number of times that the log entry occurred.

Log Statistics table

The Log Statistics table lists log entries that indicate an operational severity level of LOG, DETAIL, DEBUG, NOTICE, INFO, or STATEMENT. The level of logging detail for informational messages is controlled by the `log_min_messages` parameter. You can set the parameter either by modifying the `postgresql.conf` file with your editor of choice or by specifying a value in the **Log Min Messages** field of the Log Manager wizard.

Column name	Description
Error Severity	Severity level of the log entry.
Total Count	Number of times that the log entry occurred.

Temp Generated Queries table

The Temp Generated Queries table displays a list of queries that created temporary files.

Column name	Description
Log Time	The time that the log entry was generated.
TempFile Size(Bytes)	The size of the temporary file in bytes.
Query	The text of the query that created the temporary file.

Temp File Statistics Timeline graph

The Temp File Statistics Timeline graph displays the size of temporary files over the specified time period. Hover over a specific point to view detailed information about that point on the graph.

Lock Statistics Timeline section

The Lock Statistics Timeline section of the Log Analysis Expert report displays information about the locks held during the specified time period:

Graphic	Description
Graph	Number of locks held at any given point during the time period. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Displays the relative percentage of each type of lock used during the selected time period.

Waiting Statistics Timeline section

The Waiting Statistics Timeline section of the Log Analysis Expert report displays information about DML statements that are waiting for a lock during the specified time period.

Graphic	Description
Graph	Number of DML statements that are waiting at any given point during the time period. Each colored line represents a statement type. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Relative percentage of each type of DML statement that waited for a lock during the selected time period.

Idle Statistics Timeline section

The Idle Statistics Timeline section of the Log Analysis Expert report displays information about the amount of time that a connection to the server is idle. An `IDLE` server is waiting for a connection from a client. A connection that is `IDLE in transaction` has started a transaction but hasn't yet committed or rolled back the transaction and is waiting for a command from the client. A session that is `IDLE in transaction (aborted)` started a transaction but hasn't yet committed or rolled back the transaction and is waiting for a command from the client. An error has occurred in the transaction, and the transaction can only be rolled back.

Graphic	Description
Graph	Times at which the server is *IDLE*, *IDLE in transaction*, and *IDLE in transaction (aborted)*. Hover over a specific point to view detailed information about that point on the graph.
Pie chart	Relative percentage of each type of lock used during the selected time period.

Autovacuum Statistics table

The Autovacuum Statistics table displays statistics about autovacuum activity on monitored servers.

Column name	Description
Log Time	Time that the autovacuum activity was written to the log.
Relation	Name of the table on which the autovacuum was performed.
Index Details	Number of index scans that were performed.
Page Details	Number of pages that were removed and number of pages that remain.
Tuple Details	Number of tuples that were removed and number of tuples that remain.
Buffer Usage	Number of buffers hit, missed, or dirty.
Read Rate	Average read rate in MBs per second.
System Usage	Percent of CPU time used performing autovacuum activities.

Autoanalyze Statistics table

The Autoanalyze Statistics table displays logged autoanalyze activity.

Column name	Description
Log Time	Time that the autoanalyze activity was written to the log.
Relation	Name of the table on which the autoanalyze was performed.
System Usage	Percent of CPU time used performing autoanalyze activities.

Slow Query Statistics table

The Slow Query Statistics table displays the slowest queries executed on monitored servers. The table includes the number of entries specified in the Rows Limit field of the Log Analysis Expert.

Column name	Description
Log Time	Time that the query activity was written to the log.
Tag	Command type.
Query	Text of the performed query.
Parameters	Parameters if the query is a parameterized query.
Duration	Length of time that it took the server to execute the query.
Host	Name of the host on which the query executed.
Database	Name of the database on which the query executed.

Frequently Executed Query Statistics table

The Frequently Executed Query Statistics table displays the most frequently executed query statements. The table includes the number of entries specified in the **Rows Limit** field of the Log Analysis Expert.

Column name	Description
Query	Text of the performed query.
Parameters	Parameters if the query is a parameterized query.
No. of Times Executed	Number of times that the query executed.
Total Duration	Length of time that it took the server to execute the query.

Most Time Executed Query Statistics table

The Most Time Executed Query Statistics table displays the queries that took the most execution time on the server. The table includes the number of entries specified in the **Rows Limit** field of the Log Analysis Expert.

Column name	Description
Query	Text of the performed query.
Parameters	Parameters if the query is a parameterized query.
No. of Times Executed	Number of times that the query executed.

continues on next page

Table 40 – continued from previous page

Column name	Description
Total Duration	Length of time that it took the server to execute the query.

Connections Overview Timeline section

The Connections Overview Timeline section of the Log Analysis Expert report displays information about successful and unsuccessful connection attempts during the specified time period.

Graphic	Description
Timestamp graph	Number of server connections attempted and connections authenticated at any given point during the specified time period. Hover over a specific point to view detailed information about that point on the graph.
Summary pie chart	The relative percentage of connections attempted and connections authenticated during the specified time period.

5.3.7 Charts

You can use the **Manage Charts** tab to create or modify a custom line chart or table or to import a Capacity Manager template to use in a custom chart. After defining a chart, you can display the chart on a custom dashboard.

To open the **Manage Charts** tab, in the PEM client, select **Management > Manage Charts**. The **Manage Charts** tab provides a Quick Links menu for accessing dialog boxes to:

- Create a new chart for use on a custom dashboard.
- Import a Capacity Manager template to use as a template for creating a custom chart.
- Access online help.

The Custom Charts table displays a list of charts you defined. When a chart is new, the font is green. When you add a chart or refresh the screen, the font is black. Use the search box in the upper-right of the Custom Charts table to search through your custom charts. Specify:

- Chart name
- Type
- Level
- Metrics category

Use icons to the left of a chart name in the Custom Charts table to manage a chart:

- Select **Edit** to open the Chart Configuration wizard and modify aspects of the chart or table.
- Select **Delete** to delete the selected chart.

Creating a custom chart

Select **Create New Chart** in the Quick Links section of the **Manage Charts** tab to open the Create Chart wizard. The wizard takes you through the steps required to define a new chart.

1. Use the Configure Chart dialog box to specify general information about the chart:

- Specify the name of the chart in the **Name** field.
- Use the list in the **Category** field to specify the category in which to display this chart. When adding a custom chart to a custom dashboard, you can select the chart from the category specified.
- Use the **Type** field to specify if the chart is a line chart or a table.
- Provide a description of the chart in the **Description** field. The description is displayed on a custom dashboard when the user selects the information icon next to the chart.

After you complete the fields on the Configure Chart dialog box, select **Next**.

1. Use the Select Metrics dialog box to select the metrics to display on the chart.

- Use the **Metric level** list to specify the level of the PEM hierarchy from which you want to select metrics. You can specify **Agent**, **Database**, or **Server**. Each level offers access to a unique set of probes and metrics.
- Use the tree in the **Available metrics** box to select the metrics to display on the chart.

If you're creating a table, you can select metrics from only one probe. Each node of the tree lists the metrics returned by a single probe. Expand a node of the tree and select the check box next to a metric name to include that metric data in the table.

If you're creating a line chart, expand the nodes of the tree and double-click each metric that you want to include in the chart.

- Use the Selected Metrics panel to specify how the metric data appears in your chart. The selection panel displays the name of the metric in the nonmodifiable Metric [Probe] column. You can:

```
-    Select the trash can to delete a metric from the list of selected metrics.
-    Use the lists in the Selection Criteria column to specify the order of the data␣
↪displayed.
-    Use the **Limit** field to specify the number of rows in a table or lines in a␣
↪chart: up to 32 lines and 100 rows.
```

- If you're creating a line chart, PEM supports comparisons of cross-hierarchy metrics.

```
-    Select **Compare** to select one or more probe-specific attributes (such as CPUs,␣
↪interfaces, and databases) to compare in the chart.
-    Select **Copy** to apply your selections to all of the metrics for the same probe.
```

After you complete the fields on the Select Metrics dialog box, select **Next**.

1. Use the Set Options dialog box to specify display options for your chart:

- Use the **Auto Refresh** field to specify the number of minutes between chart updates. Choose a value from 1 to 120. The default auto refresh rate is 2 minutes.
- Use fields under **Line chart options** to specify display preferences for a line chart:

```
-    Use the **Points to plot** field to specify the maximum number of points to plot on␣
↪the chart.
-    Use the fields to the right of **Historical span** to specify how much historical␣
↪data to display on the chart in days, hours, and minutes.
```

- Use the fields in the **Data extrapolation** box to specify whether PEM generates extrapolated data based on historical data:

```
-    Select **No Extrapolation** to omit extrapolated data from the chart.
-    Select **Span** to use the **Days** and **Hours** selectors to specify the period␣
↪of time spanned by the metrics on the chart.
-    Select **Threshold** to use threshold selectors to specify a maximum or minimum␣
↪value for the chart.
```

After you complete the fields in the Set Options dialog box, select **Next**.

1. Use the Set Permissions dialog box to specify display options for your chart.

- Set the **Share with all** slider to **Yes** to make the chart available to all authorized users. Select **No** to restrict access to the users or groups specified in the **Access permissions** field.
- Use the **Access permissions** field to select the groups you want to give access to the chart to.

After you finish defining the chart, select **Finish** to save your edits and add your chart to the list on the **Manage Charts** tab.

Importing a Capacity Manager template

Select **Import Capacity Manager Template** in the Quick Links section of the **Manage Charts** tab to open the Create Chart dialog box and use a Capacity Manager template as a starting point for a chart or table.

1. When the Create Chart dialog box opens, provide information about the custom chart:

- Use the **Import capacity template** list to select the name of the template on which the chart is based.
- Specify the name of the chart in the **Name** field.
- Use the **Category** list to specify the category in which to display this chart. When adding a custom chart to a custom dashboard, you can select the chart from this category.
- Use the **Type** field to specify if the chart is a line chart or a table.
- Provide a description of the chart in the **Description** field. The description is displayed on the custom dashboard when user selects the information icon next to the chart.

Select **Next**.

1. The Select Metrics window allows you to review the metrics specified by the selected template. The bottom panel of the chart editor displays the metrics to include in the chart. You can't modify the metrics included in the chart using the chart editor. To modify the metrics, use the Capacity Manager utility to update the template.

After you review the metrics, select **Next**.

1. Use the Set Options window to specify display options for your chart:

- Use the **Auto Refresh** field to specify the number of minutes between chart updates. Choose a value from 1 to 120. The default auto refresh rate is 2 minutes.
- Use the **Data extrapolation** box to specify the time period covered by the chart. You can either:
- Select **Historical days and extrapolated days** and provide: - The number of days of historical data to chart in the **Historical** field. - The number of projected days to chart in the **Extrapolated** field.
- Select **Historical days and threshold** and provide: - The number of days of historical data to chart in the **Historical** field. - The threshold value at which the chart ends.

After you complete the Set Options window, select **Next**.

1. Use the Set Permissions window to specify display options for your chart:
    - Set the **Share with all slider** to **Yes** to make the chart available to all authorized users. Set it to **No** to restrict access to the users or groups specified in the **Access permissions** field. - Use the **Access permissions** field to select the groups to give access to the chart to.

After you finish defining the chart, select **Finish** to save your edits and add your chart to the list on the **Manage Charts** tab.

5.3.8 Dashboards

PEM displays performance statistics through a number of dashboards. Each dashboard contains a series of summary views that contain charts, graphs, and tables that display the statistics related to the selected object.

Dashboards overview

The PEM client displays the Global Overview dashboard when it connects to the PEM server. Additional dashboards provide statistical information about monitored objects.

Accessing in-app help for UI details

The in-app help provides comprehensive details about the PEM web interface, including the menu items, charts and graphs on the dashboards, and the options on the dialog boxes. The in-app help also includes instructions for creating database objects.

- To access the landing page for the help, select **Help > Online help**.
- To access context-sensitive help for a dialog box, select **?**.

Opening dashboards

Dashboards are presented in a hierarchy comparable to the PEM client tree control. The dashboard for each object in the tree control displays the information for that object as well as for any monitored object that resides below that level in the tree control, if appropriate.

Each dashboard header displays the date and time that the server was started, if relevant, the date and time that the dashboard was last updated, and the current number of triggered alerts. Navigation menus displayed in the dashboard header provide easy access to other dashboards. Menus are organized hierarchically. Only those menus appropriate for the object currently highlighted in the tree control are available.

- Select **Global Overview** from any dashboard to return to the Global Overview dashboard.
- Select the name of an agent from the **Agents** menu to navigate to the Operating System Analysis dashboard for that agent.
- Select a server name from the **Servers** menu to navigate to the Server Analysis dashboard for that server.
- Select a database name from the **Databases** menu to navigate to the Database Analysis dashboard for that database.
- Use the **Dashboards** menu to navigate to informational dashboards at the global level or for the selected agent, server, or database.

Dashboards display statistical information in the form of:

- Tables that provide statistical information collected by a PEM probe.

- Pie charts that display information collected by the most recent execution of a probe.
- Bar graphs that display comparative statistics collected by the most recent execution of a probe.
- Line graphs that display statistical data collected by PEM probes.

You can open a dashboard using either of these techniques:

- from the **Management > Dashboards** menu, select an active dashboard name.
- Right-click the name of a monitored object in the tree. From the **Dashboards** menu, select the name of the dashboard to review.

Each dashboard is displayed on the **Monitoring** tab in the main panel of the client window. After opening a dashboard, you can navigate to other dashboards in the same tab.

Each dashboard header includes navigation menus that allow you to navigate to other dashboards. Use your browser's forward and back buttons to scroll through previously viewed dashboards. Use **Refresh** to update the current dashboard.

To sort statistics that are provided in table form, select a column heading. Select it again to reverse the sort order. Each table offers a stable sort feature. For example, to sort a table by ascending Session ID in each user name group, sort first by the Session ID column, then sort by the User Name column.

Hover over the upper-right corner of each graph, chart, or table to reveal the PEM client toolbar icons. Hover over an icon to display a tooltip that briefly explains the icon's functionality:

- Use the **Refresh** icon to update the information displayed on a dashboard.
- Use the **Save Chart as Image** icon to save the selected chart as a `.jpeg` image.
- Use the **Full Screen** icon to enlarge the chart to reveal granular details about the charted data.
- Use the **Personalize the chart configuration** icon to access a control panel that allows you to select chart-specific display details.
- Hover over the **Explain** icon to review a description of the information shared in the graph or chart.

In the lower-right corner of each graph or chart is a legend that identifies each item plotted in the graph or chart.

If it's displayed, select the information icon in the upper-left corner of a chart to display a note about the chart content and, if applicable, a link that allows you to enable one or more probes that retrieve content for the chart.

Available dashboards

PEM offers the following dashboards. For more information on each of the available dashboards, see the PEM client's online help.

Alerts dashboard

The Alerts dashboard displays the currently triggered alerts. If opened from the Global Overview, the dashboard displays the current alerts for all monitored nodes on the system. If opened from a node in a server, the report shows alerts related to that node and all monitored objects that reside below that object in the tree.

Audit Log Analysis dashboard

For EDB Postgres Advanced Server users, the Audit Log Analysis dashboard allows you to browse the audit logs that were collected from instances with audit logging and collection enabled.

PGD Admin dashboard

The PGD Admin dashboard displays overview information about the EDB Postgres Distributed node, group, and worker.

PGD Group Monitoring dashboard

The PGD Group Monitoring dashboard displays information about EDB Postgres Distributed group subscription and group replication slots.

PGD Node Monitoring dashboard

The PGD Node Monitoring dashboard displays information about EDB Postgres Distributed node slots, node replication rates, and conflict history summary for the selected node.

Database Analysis dashboard

The Database Analysis dashboard displays performance statistics for the selected database.

I/O Analysis dashboard

The I/O Analysis dashboard displays I/O activity across various areas such as object DML activity and log operations.

Memory Analysis dashboard

The Memory Analysis dashboard supplies statistics concerning various memory-related metrics for the Postgres server.

Object Activity Analysis dashboard

The Object Activity Analysis dashboard provides performance details on tables/indexes of a selected database.

Operating System Analysis dashboard

The Operating System Analysis dashboard supplies information regarding the performance of the underlying machine's operating system.

Probe Log Analysis dashboard

The Probe Log Analysis dashboard displays any error messages returned by a PEM agent.

Server Analysis dashboard

The Server Analysis dashboard provides general performance information about the overall operations of a selected Postgres server.

Server Log Analysis dashboard

The Server Log Analysis dashboard allows you to filter and review the contents of server logs that are stored on the PEM server.

Session Activity Analysis dashboard

The Session Activity Analysis dashboard provides information about the session workload and lock activity for the selected server

Session Waits Analysis dashboard

The Session Waits Analysis dashboard provides an overview of the current DRITA wait events for an Advanced Server session.

Storage Analysis dashboard

The Storage Analysis dashboard displays space-related metrics for tablespaces and objects.

System Waits Analysis dashboard

The System Waits Analysis dashboard displays a graphical analysis of system wait information for an EDB Postgres Advanced Server session.

Streaming Replication Analysis dashboard

The Streaming Replication Analysis dashboard displays statistical information about WAL activity for a monitored server and allows you to monitor the status of Failover Manager clusters.

Dashboard configuration

Options on the Dashboard Configuration dialog box allow you to link the timelines of all of the line graphs on the dashboard. To open the Dashboard Configuration dialog box, select **Settings** in the dashboard header.

Use the Dashboard Configuration dialog box to control attributes of the charts displayed on the dashboard:

- Set **Link timelines of all the line charts** to **Enable** to apply the specified timeline to line graphs displayed on the dashboard. If set to **Disable**, your preferences are preserved for later use but don't modify the amount of data displayed.
- Use the **Days** selector to specify the number of days of gathered data to display on line graphs.
- Use the **Hour(s)** selector to specify the number of hours of gathered data to display on line graphs.
- Select **Remember configuration for this dashboard** to apply the customized time span to only the current dashboard only. Leave it cleared to apply the time span globally to line graphs on all dashboards.

Settings specified on the Dashboard Configuration dialog box are applied only to the current user's session.

After you specify your preferences, select **Save**.

Managing custom dashboards

PEM displays performance statistics on a number of system-defined dashboards. Each dashboard contains a series of summary views that contain charts, graphs, and tables that display statistics related to the selected object. You can use the **Manage Dashboards** tab to create and manage custom dashboards that display the information that's most relevant to your system.

To create a custom dashboard, select **Create New Dashboard** located in the Quick Links section of the **Manage Dashboards** tab.

To modify an existing dashboard, select **Edit** next to the dashboard name. The dashboard editor opens, displaying the definition of the dashboard. After you finish modifying the dashboard's definition, select **Save**.

To delete a dashboard, select **Delete** next to the dashboard name.

Creating a custom dashboard

You can use the PEM dashboard editor to create or modify a custom dashboard. The custom dashboard can include predefined charts, custom charts, or a mix of predefined and custom charts.

Use the **Configure** section to specify general information about the dashboard:

- Specify a name for the dashboard in the **Name** field.
- Use the **Level** list to specify the level of the PEM hierarchy in the PEM client on which to display the dashboard. You can access a dashboard using the **Dashboards** menu on a global level, an agent level, the server level, or the database level. Each selected level in the list exposes a different set of metrics on which to base the custom dashboard's charts.
- Provide a description of the dashboard in the **Description** field.

Provide information in the **Ops dashboard options** box to use the dashboard as an Ops dashboard:

- Set **Ops Dashboard?** to **Yes** to create a dashboard that's formatted to display on an Ops monitor.
- Set **Show Title?** to **Yes** to display the dashboard name at the top of the Ops dashboard.
- Use the **Font** list to select a custom font style for the title.

- Use the **Font size** list to select a custom font size for the title.

Use the **Permissions** box to specify the users who can view the new dashboard:

- Set **Share with all** to **Yes** to allow all teams to access the dashboard. Set it to **No** to enable the **Access permissions** field.
- Use the **Access permissions** field to specify the roles that can view the new dashboard. In the field, select from the list of users to add a role to the list of users with dashboard access.

After you complete the **Configure Dashboard** section, select the arrow in the upper-right corner to close the section and access the **Dashboard Layout Design** section.

Select **Edit** in a section header to specify a section name. Then, select the plus sign (+) to add a chart to the section.

Use the arrows to the right of each chart category to display the charts available and select a chart.

Use the chart detail selectors to specify placement details for the chart:

- Use the **Chart width** selector to indicate the width of the chart. Select **50%** to display the chart in half of the dashboard or **100%** to use the whole dashboard width.
- Use the **Chart alignment** selector to indicate the position of the chart in the section, left, center, or right justified.

Tables are always displayed centered.

When creating or editing a custom dashboard, you can drag to rearrange the charts in a section or to move a chart to a different section.

To add another chart to your dashboard, select the plus sign (+) in the section header. After you finish editing the dashboard, select **Save**.

Creating an Ops dashboard

You can use the PEM dashboard editor to create a custom dashboard formatted for display on an Ops monitor. An Ops dashboard displays the specified charts and graphs while omitting header information and minimizing extra banners, titles, and borders.

To create an Ops dashboard, provide detailed information about the Ops display in the **Ops dashboard options** section of the Create Dashboard dialog box.

- Set **Ops Dashboard?** to **Yes** to create a dashboard that' s formatted for display on an Ops monitor.
- Set **Show Title?** to **Yes** to display the dashboard name at the top of the Ops dashboard.
- Use the **Font** list to select a custom font style for the title.
- Use the **Font size** list to select a custom font size for the title.

After adding charts and tables to the Ops dashboard, select **Save**. You can then access the dashboard by navigating through the **Dashboards** menu of the hierarchy level specified in the **Level** field on the New Dashboard dialog box.

5.3.9 Remote monitoring

Remote monitoring is monitoring your Postgres cluster using a PEM Agent residing on a different host.

To remotely monitor a Postgres cluster with PEM, you must register the cluster with PEM and bind a PEM agent. See *Registering a Postgres server* for more information.

The following scenarios require remote monitoring using PEM:

- Postgres cluster running on AWS RDS
- `:ref:`Postgres cluster running on BigAnimal </edb-postgres-ai/cloud-service/using_cluster/monitoring_and_logging/>``

PEM remote monitoring supports:

Feature Name	Remote monitoring supported?	Comments
Query information	Yes	
Viewing system reports	Yes	
Query information	Limited	There's no correlation between the Postgres cluster and operating system metrics.
Query information	Limited	When you run an alert script on the Postges cluster, it runs on the machine where the bound PEM agent is running and not on the actual Postgres cluster machine.
Query information	Limited	Some dashboards might not be able to show complete data. For example, the operating system information where the Postgres cluster is running isn't displayed as it isn't available.
Query information	Limited	Some of the PEM probes don't return information, and some of the functionality might be affected. For details about probe functionality, see *Query information* .
Query information	Limited	The Postgres Expert provides partial information as operating system information isn't available.
Query information	Limited	Scheduled tasks work only for Postgres clusters, and scripts run on a remote agent.
Query information	Limited	The Core Usage reports don't show complete information. For example, the platform, number of cores, and total RAM aren't displayed.
Query information	No	
Query information	No	
Postgres Log Analysis Expert	No	
Query information	No	

5.4 Monitoring Barman

Barman (Backup and Recovery Manager) is an open-source administration tool for remote backups and disaster recovery of PostgreSQL servers in business-critical environments. It relies on PostgreSQL' s point-in-time recovery technology, allowing DBAs to remotely manage a complete catalog of backups and the recovery phase of multiple remote servers from one location. For more information, see *Monitoring Barman* .

Starting with version 8.4, you can monitor a Barman server through the PEM console.

5.4.1 Prerequisites for monitoring Barman

Before adding a Barman server to the PEM console:

- You must manually install and configure Barman on the Barman host. For more information about installing and configuring Barman, see *Monitoring Barman* .
- Install the pg-backup-api tool on Barman host. For more information about installing, see pg-backup-api .

5.4.2 Configuring a Barman server

You can configure and edit your Barman server using:

- PEM web client
- `pemworker` command line

Using PEM web client

Configure

You can use the Create–BARMAN Server dialog box to register an existing Barman server with the PEM server. To open the dialog box, right-click the BARMAN Servers node and select **Create-BARMAN Server**.

Use the **General** tab to describe the general properties of the Barman server:

- Use the **Name** field to specify a name for the server. The name identifies the server in the browser tree.
- Use the **URL** field to specify the URL of the host where Barman is installed.
- Use the **Team** field to specify a PostgreSQL role name. Only PEM users who are members of this role, who created the server initially, or have superuser privileges on the PEM server can see this server when they log on to PEM. If this field is left blank, all PEM users see the server.

Use the **PEM Agent** tab to specify connection details for the PEM Agent:

- Use the **Bound Agent** field to select the agent that you want to configure as a Barman server. Only those PEM agents that are supported for Barman are listed.
- Use the **Probe Frequency** field to specify the number of seconds to execute the probes with the specified interval.
- Use the **Hearbeat** field to specify the interval to check the availability of PEM agent in seconds.

Note: After registering the Barman server, you need to restart the PEM agent.

Editing

To edit your Barman server, right-click the server from the browser tree and select **Properties**.

- Use the **PEM Agent** tab to modify the bound agent, probe frequency, and heartbeat. Only the owner of the Barman server can modify the fields on the **PEM Agent** tab.
- Use the **Information** tab to view the detailed information about your Barman server. This tab gets populated whenever the Barman related probes are executed.
- Use the **Configuration** tab to view the configuration settings of your Barman server. This tab gets populated whenever the Barman related probes are executed.

Using pemworker command line

You can configure Barman server using `pemworker` command line options.

```
asheshvashi@pem:~/PEM/agent$ ./pemworker --update-barman --help
./pemworker --update-barman [barman-update-options]

barman-update-options:
  --id <barman-id> (ID for the existing BARMAN API pg-backup-api)
  --api-url <url> (URL of the BARMAN API pg-backup-api)
  --probe-execution-frequency <interval> (Default: 30, Probe the BARMAN API pg-backup-
↪api at regular interval in seconds and fetch the metrics.)
  --heartbeat-interval <interval> (Default: 10, Ping the BARMAN API pg-backup-api status␣
↪API at a regular interval in seconds for checking its availability.)
  --ssl-crt <certificate_file> (SSL certificate file for the BARMAN API.)
  --ssl-key <key_file> (Private SSL key for the BARMAN API.)
  --ssl-ca-cert <ca_file> (CA certificate to verify peer against the BARMAN API.)
  --config-file/-c <config_file> (Path to the agent configuration file.)

asheshvashi@pem:~/PEM/agent$ ./pemworker --unregister-barman --help
./pemworker --unregister-barman [barman-unregistration-options]

barman-unregistration-options:
  --id <barman-id> (ID for the existing BARMAN API, registered with the PEM Server.pg-
↪backup-api)
  --config-file/-c <config_file> (Path to the agent configuration file.)

asheshvashi@pem:~/PEM/agent$ ./pemworker --register-barman --help
./pemworker --register-barman [barman-registration-options]

barman-registration-options:
  --api-url <url> (URL of the BARMAN API pg-backup-api)
  --description <name> (Description to show on the UI User interface for the BARMAN API.)
  --probe-execution-frequency <interval> (Default: 30, Probe the BARMAN API pg-backup-
↪api at regular interval in seconds and fetch the metrics.)
  --heartbeat-interval <interval> (Default: 10, Ping the BARMAN API pg-backup-api status␣
↪API at a regular interval in seconds for checking its availability.)
  --ssl-crt <certificate_file> (SSL certificate file for the BARMAN API.)
  --ssl-key <key_file> (Private SSL key for the BARMAN API.)
  --ssl-ca-cert <ca_file> (CA certificate to verify peer against the BARMAN API.)
  --team <database-role> (Specify the name of the database group role, on the PEM␣
```

(continues on next page)

(continued from previous page)

```
↪backend database server, that should have access to this BARMAN API Server.)
 --owner <database-user> (Specify the name of the database user, on the PEM backend␣
↪database server, who will own the BARMAN API Server.)
 --config-file/-c <config_file> (Path to the agent configuration file.)
```

Note: After registering the Barman server, you need to restart the PEM agent.

5.4.3 Viewing the Barman server details on a PEM dashboard

Once the Barman server is configured, you can see the entire backup- and server-related details for that Barman server on the PEM dashboard.

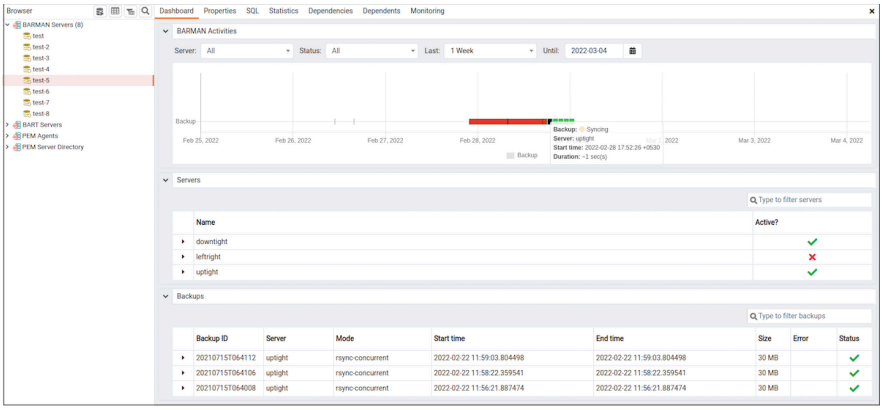


Fig. 13: BARMAN dashboard

When you select a monitored Barman server, details of all the associated database servers along with their activities are displayed as a chart on the dashboard in the Barman Activities panel. You can select the activities on any criteria that you specify in the filter boxes (the database server, status, duration, or date).

The Servers panel displays a list of all the database servers managed by that Barman server along with the active status.

The Backups panel displays a list of all the database server backups managed by that Barman server. You can filter the list to display the details of any database server. You can also filter the list on any criteria that you specify in the filter box. Typically, this filter works with any kind of string value (excluding date, time, and size) listed under the columns. For example, you can enter `tar` to filter the list and display only those backups that are in tar format.

Backup details include the Backup ID, Server, Mode, Start time, End time, Size, Error, and Status column.

5.5 Monitoring EDB Postgres Distributed

EDB Postgres Distributed provides multi-master replication and data distribution with advanced conflict management, data-loss protection, and throughput up to 5X faster than native logical replication , and enables distributed PostgreSQL clusters with high availability up to five 9s. Before you monitor nodes in a EDB Postgres Distributed cluster through the PEM console, you must first deploy a EDB Postgres Distributed cluster and ensure that your database nodes are up and running. For more information on installing EDB Postgres Distributed see *Monitoring EDB Postgres Distributed* .

You can configure PEM to display status information about one or more EDB Postgres Distributed database nodes using dashboards in PEM version 8.1.0 and EDB Postgres Distributed version 3.7.9 and later.

To configure PEM to monitor EDB Postgres Distributed database nodes, use the PEM web client to create a server definition. Use the tabs on the *Manually registering a database server* dialog box to specify general connection properties for the EDB Postgres Distributed database node with the following exceptions:

- Specify the EDB Postgres Distributed-enabled database name in the **Database** field of the **PEM Agent** tab.
- Specify the user having the pgd_monitor or pgd_superuser role in the **username** field of the **PEM Agent** tab.

After saving the server definition, the EDB Postgres Distributed database node is included in the list of servers under the PEM server directory in the PEM client object browser tree. You can monitor the nodes from EDB Postgres Distributed (PGD) dashboards.

To include monitoring information on the EDB Postgres Distributed (PGD) dashboards, you must enable the relative probes for each EDB Postgres Distributed group. See the complete list of EDB Postgres Distributed *Probes* .

To enable a probe, right-click the node name, and select **Management > Manage Probes**.

To monitor the EDB Postgres Distributed database node, right-click the name of the node in the object browser tree. From the **Dashboards** menu, select the PGD Admin, PGD Group Monitoring, or PGD Node Monitoring dashboard.

5.6 Monitoring Failover Manager

If you’re using EDB Failover Manager to monitor your replication scenario, you must manually install and configure Failover Manager. For detailed information about installing Failover Manager, see the EDB website .

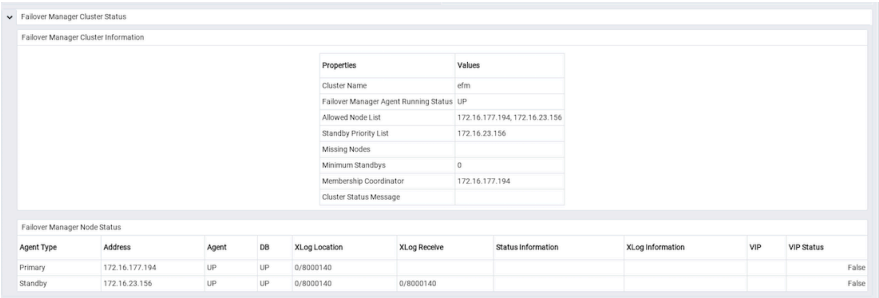
To monitor the status of a Failover Manager cluster on the Streaming Replication dashboard, you must provide the following information on the **Advanced** tab of the **Server Properties** dialog box for each node of the cluster:

- Use the **EFM Cluster Name** field to specify the name of the Failover Manager cluster. The cluster name is the prefix of the name of the cluster properties file. For example, if your cluster properties file is named `efm.properties` , your cluster name is efm.
- Use the **EFM Installation Path** field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in `/usr/efm-<X>/bin` , where `<X>` is the EFM version.

Note: To monitor Failover Manager, the PEM agent executes the *efm cluster-status-json <cluster_name>* command as the user efm. This command fails if the cluster properties file isn't in the efm user's home directory. In this case, create a symlink to the cluster properties files in the home directory of efm, and ensure that efm has permission to read and execute the cluster properties file.

After registering your servers, the Streaming Replication Analysis dashboard displays status information about your EFM cluster near the bottom of the dashboard.

The **Failover Manager Cluster Status** section of the Streaming Replication Analysis dashboard displays information about the monitored cluster.



Failover Manager Cluster Status

Failover Manager Cluster Information

Properties	Values
Cluster Name	efm
Failover Manager Agent Running Status	UP
Allowed Node List	172.16.177.194, 172.16.23.156
Standby Priority List	172.16.23.156
Missing Nodes	
Minimum Standbys	0
Membership Coordinator	172.16.177.194
Cluster Status Message	

Failover Manager Node Status

Agent Type	Address	Agent	DB	XLog Location	XLog Receive	Status Information	XLog Information	VIP	VIP Status
Primary	172.16.177.194	UP	UP	0/8000140					False
Standby	172.16.23.156	UP	UP	0/8000140	0/8000140				False

Fig. 14: The Failover Manager cluster status report

The Failover Manager Cluster Information table provides information about the Failover Manager cluster:

- The Properties column displays the name of the cluster property.
- The Values column displays the current value of the property.

The Failover Manager Node Status table displays information about each node of the Failover Manager cluster:

- The Agent Type column displays the type of agent that resides on the node. The possible values are Primary, Replica, Witness, Idle, and Promoting.
- The Address column displays the IP address of the node.
- The Agent column displays the status of the agent that resides on the node.
- The DB column displays the status of the database that resides on the node.
- The XLog Location column displays the transaction log location of the database.
- The Status Information column displays any error-related information about the node.
- The XLog Information column displays any error-related information about the transaction log.
- The VIP column displays the VIP address that’s associated with the node.
- The VIP Status column displays True if the VIP is active for the node, False if not.

5.6.1 Replacing a primary node

You can use the PEM client to replace the primary node of a Failover Manager cluster with a replica node. To start the failover process, select **Tools > Server > Replace Cluster Primary**. You are prompted to confirm that you want to replace the current primary node.

Select **Yes** to remove the current primary node from the Failover Manager cluster and promote a replica node to the role of read/write primary node in a Failover Manager cluster. The node with the highest promotion priority (defined in Failover Manager) becomes the new primary node. PEM reports the job status.

When the job completes and the Streaming Replication Analysis dashboard refreshes, you can review the Failover Manager Node Status table to confirm that a replica node was promoted to the role of primary in the Failover Manager cluster.

5.6.2 Switchover EFM cluster

You can use the PEM client to replace the primary node of a Failover Manager cluster with a replica node. To start the switchover process, select **Tools > Switchover EFM Cluster**. You are prompted to confirm that you want to switch over EFM cluster. Select **Yes** to:

- Start the Failover Manager switchover.
- Promote a replica node to the role of read/write primary node.
- Reconfigure the primary database as a new replica in a Failover Manager cluster.

The node with the highest promotion priority (defined in Failover Manager) becomes the new primary node. PEM reports the job status. When the job completes and the Streaming Replication Analysis dashboard refreshes, you can review the Failover Manager Node Status table to confirm that a switchover occurred.

5.7 Monitoring Replication Server

Before configuring PEM to retrieve statistics from an EDB Postgres Advanced Server or PostgreSQL database that' s part of an xDB replication scenario, you must manually install and configure xDB replication. For more information about xDB replication solutions and documentation, see Multi-master replication operation .

The PEM xDB Replication probe monitors lag data for clusters that use xDB multi-primary or single-primary replication that have an EDB Postgres Advanced Server or PostgreSQL publication database. If you configured replication between other proprietary database hosts (that is, Oracle or SQL Server) and EDB Postgres Advanced Server or PostgreSQL, the probe can' t return lag information.

By default, the xDB Replication probe is disabled. To enable the xDB Replication probe:

1. Right-click the name of the server and select **Connect** from the context menu.
2. If prompted, provide authentication information.
3. After connecting, expand the server node of the tree and select the name of the replicated database.
4. Select **Management > Manage Probes**.

Use the **Manage Probes** tab to configure the xDB Replication probe:

- Set **Default** to **No** to modify the minutes and seconds between probe executions.
- Use the **Enabled?** slider to enable the xDB Replication probe.
- Set **Default** slider in the **Data Retention** field to **No** to modify the number of days for PEM to store the information retrieved by the probe.

After enabling the probe, you can use the metrics returned to create custom charts and dashboards in the PEM client.

5.8 Monitoring event history

You can monitor the executed event details in the `pem.event_history` table. It provides the username, execution time of the event, type of the event occurred, and the details regarding the event. The fields in the table are:

!!!Note Currently, `pem.event_history` table records only the alert blackout history.

Field name	Description
recorded_time	Records the execution time of the event. Displays the date and time along with the fractional seconds.
user_name	The username of the user who executed or scheduled the event.
component	The event name.
operation	The action taken for the event. For example, if the user has enabled an alert blackout, it displays *enable_alert_blackout*.
message	The description of the action taken for the event. For example, if the user has enabled an alert blackout for the server, it displays *Enabled the alert blackout for the server*.
details	The JSON output of the event that occurred.

5.9 Tuning performance

The following features help in tuning the performance:

- `:ref:`Tuning Wizard <Tuning Wizard>`
- `:ref:`Postgres Expert <Postgres Expert>`
- `:ref:`Performance Diagnostic <Performance Diagnostic>`

5.9.1 Tuning Wizard

The Tuning Wizard reviews your PostgreSQL or EDB Postgres Advanced Server installation, recommending a set of configuration options tuned to the installation' s anticipated workload. Benchmarking systems or systems with a high work load might require also require manual tuning to reach optimum performance.

Before using the Tuning Wizard, you must specify the name of the service in the server' s Properties dialog box. In the **Advanced** tab, enter the service name in the **Server ID** field.

The Tuning Wizard can make recommendations only for those servers that reside on the same server as their bound PEM agent. If you specify a value of **Yes** in the **Remote monitoring** field when defining your server, the server doesn' t display in the Tuning Wizard tree.

1. To open the Tuning Wizard, in the PEM client select **Management > Tuning Wizard**.
2. Select **Next**.
3. When you expand the **Servers** node of the tree, a list of servers appears. All of these servers are currently monitored by PEM and available for tuning. Select a server to tune it.

Note: If you don't provide the server's service name, then the Tuning Wizard displays a warning next to the server name on the tree.

4. Select **Next**.
5. Select an option in the **Machine utilization** field to specify the type of work performed by each server. The type of work performed by the server determines how the Tuning Wizard allocates system resources:
   - Select **Dedicated** to dedicate the majority of the system resources to the database server.
   - Select **Mixed use** to dedicate a moderate amount of system resources to the database server.

- Select **Developer workstation** to dedicate a relatively small amount of system resources to the database server.
- Select an option in the **Workload Selection** field to specify the type of workload typically performed on the selected server:
  - Select **OLTP** if the selected server is used primarily to process online transaction workloads.
  - Select **Mixed** if the selected server provides a mix of transaction processing and data reporting.
  - Select **Data warehouse** if the server is used for heavy data reporting.

6. Select **Next**.
7. The tree on the Tuning Changes Summary dialog box displays the parameter setting modifications recommended for each server analyzed by the Tuning Wizard. Select the recommendations that you want the Tuning Wizard to apply or to include in a preview report:
   - Select a parameter name, and the Tuning Wizard includes the parameter setting.
   - Select the server name, and the Tuning Wizard includes all parameter setting recommendations for the specified server.
8. Select **Next**.
9. In the Schedule or Run? dialog box, either select a time for PEM to apply the recommended changes or generate a report that details the recommended changes.

PEM makes the recommended changes that you selected in the Tuning Changes Summary dialog box. If you choose to generate a report, then PEM creates a report. It contains a list of the current values and recommended changes to the configuration parameters as selected in the Tuning Changes Summary dialog box. To implement changes, you must open the Tuning Wizard a second time, selecting the parameters you want to modify in the Tuning Changes Summary dialog box.

Select **Schedule changes** to view and specify your scheduling options.

You can set the **Configuration now?** slider to:

```
-  **Yes** &mdash; Apply the Tuning Wizard's recommendations and restart the server.

-  **No** &mdash; Enable the **Time?** field where you can specify a date and time with␣
↪the calendar selector. PEM applies the recommended changes and restarts the server at␣
↪this time.
```

Select **Generate report** to view your report options.

You can set the **View report now?** slider to:

```
-  **Yes** &mdash; Display the Tuning Wizard report onscreen.

-  **No** &mdash; Enable the **Save the report to file** field where you can specify a␣
↪file name and location.
```

10. Select **Finish**.

To confirm that the Tuning Wizard implemented the recommended changes, review the `postgresql.conf` file for the modified server. When the change is applied, the Tuning Wizard adds a comment above each modified parameter in the `postgresql.conf` file.

You can also confirm a parameter value by querying the server. For example, to confirm the value of the shared_buffers parameter, open a SQL command line using either the Query tool (accessed through the Tools menu) or the psql client, and issue the command:

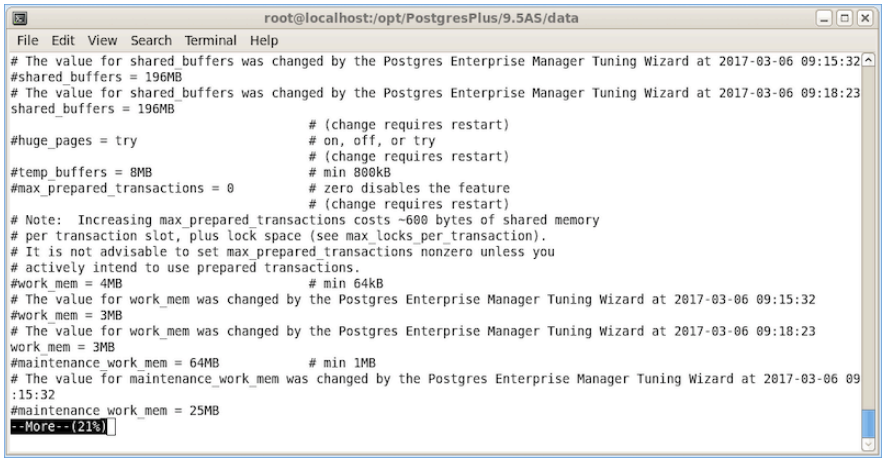


Fig. 15: Confirming a change in the postgresql.conf file

```
SHOW shared_buffers;
```

The value returned by the server confirms whether the parameter was modified.

5.9.2 Postgres Expert

The Postgres Expert utility provides expert advice on how to best configure your Postgres servers for optimal performance, security, and more. Postgres Expert serves as a PostgreSQL DBA-in-a-box by evaluating your servers for deviations in best practices. Postgres Expert contains three specialized experts:

- Configuration Expert
- Schema Expert
- Security Expert

You can select specific rules for each expert to evaluate or accept all rules. Then, review Postgres Expert reports detailing any best practice issues that require your attention.

Using the Postgres Expert wizard

To use the Postgres Expert wizard, in the PEM client select **Management > Postgres Expert**.

1. In the Welcome message, select **Next**.
2. The **Experts/Rules** tree lists the available experts and rules for identifying best practice deviations. Select the ones you want to use to evaluate the selected servers or databases.

The tree categorizes the rules under three experts:

- **Configuration Expert** evaluates the parameter settings of the server or operating system to find any adjustments that might improve system performance.
- **Schema Expert** evaluates schema objects, such as locating missing primary keys or foreign keys without indexes.
- **Security Expert** evaluates the system to find security vulnerabilities.

3. Use the check box to the left of an **Experts/Rules** to indicate for Postgres Expert to analyze the configuration of the selected servers for any best practice deviations related to the selected item.

- Use the check box next to **Experts/Rules** to select or deselect all of the items listed in the tree control.
- Use the check box next to the name of an expert to select or deselect all of the configuration items listed under that node of the tree control.
- Use the check box next to a rule to select or deselect the rule for inclusion in the Postgres Expert report. After making your selections, select **Next**.

4. Select or clear the servers and databases that you want to evaluate. If you select multiple servers or databases, then the resulting report contains a separate evaluation of each target. After you finish, select **Next**.
5. To view the report in the client, select **View the report now**. To save a copy to an HTML file, select **Download the report**. If you download the report, then the file saves to your default download directory. Select **Finish**.

Reviewing Postgres Expert recommendations

Postgres Expert produces a report that contains an evaluation of the selected rules. Each rule is categorized as high, medium, or low severity for the selected servers.

The report header contains a summary of the report that includes:

- The date and time of the report
- The number of rules evaluated
- The number of deviations from best practices

To jump to the section for a given server, select the server from the **Go to** list.

The body of the report contains the detailed findings for each of the selected servers. The findings are sorted by expert. Under each heading, any rule violations are ranked by severity.

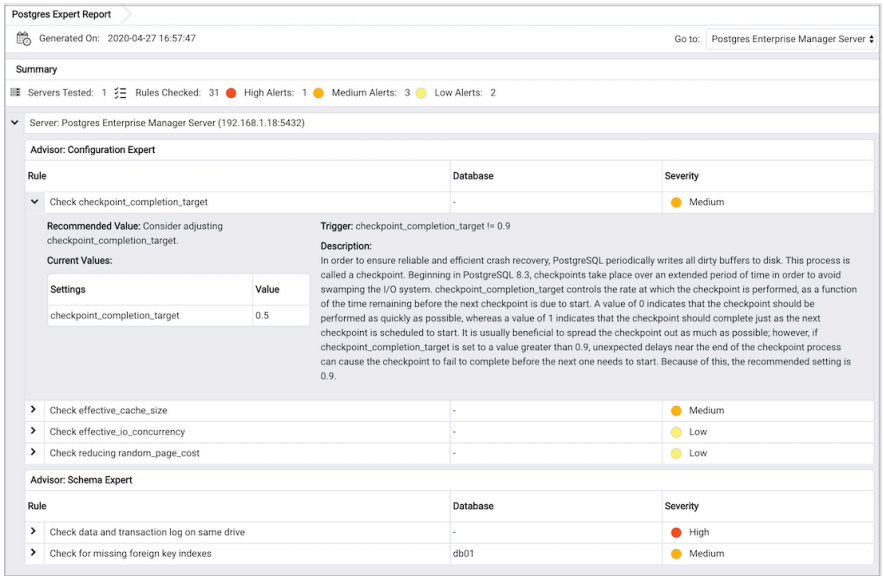


Fig. 16: The detailed recommendation for a rule

Select each rule in the report to display details and recommendations for that rule. Selecting the rule displays these section headings:

- Advisor lists the name of the Postgres Expert advisor that prompted the recommendation.
- Trigger displays a description of the rule that raised the alert.

- Recommended Value displays the value to which Postgres Expert recommends setting the selected parameter.
- Description displays information and advice about the parameter that caused the alert.
- Current Values displays the current values of any parameters that influence the Postgres Expert evaluation.

Accessing in-app help for UI details

The in-app help provides comprehensive details about the PEM web interface, including the menu items, charts and graphs on the dashboards, and the options on the dialog boxes. The in-app help also includes instructions for creating database objects.

- To access the landing page for the help, select **Help > Online help**.
- To access context-sensitive help for a dialog box, select **?**.

5.9.3 Performance Diagnostic

The Performance Diagnostic dashboard analyzes the database performance for Postgres instances by monitoring the wait events. To display the diagnostic graphs, PEM uses the data collected by the EDB wait states module. For more information on EDB wait states, see EDB wait states .

To analyze the wait states data on multiple levels, narrow down the data you select. The data you select at the higher level of the graph populates the lower level.

Note: The Performance Diagnostic dashboard is available for all the Postgres variants and versions for which the EDB wait states module is available.

Prerequisites

- Install the EDB wait states package:

```
-  For PostgreSQL, see  `EDB repositories <https://repos.enterprisedb.com/>`_  .

-  For EDB Postgres Advanced Server, see  :ref:`EDB wait states </pg_extensions/wait_
↪states/installing/>`  .
```

- After you install the EDB wait states module of EDB Postgres Advanced Server: 1. Configure the list of libraries in the `postgresql.conf` file:

```
shared_preload_libraries = $libdir/edb_wait_states
```

2. Restart the database server. 3. Create the following extension in the maintenance database:

```
CREATE EXTENSION edb_wait_states;
```

- If you want to access the Performance Diagnostic dashboard as a user without superuser privileges, specific *Login roles* must be granted to the user based on the server to which the user is connected or logged into:
- Grant `pem_comp_performance_diagnostic` role to the user (`test1` in this example), who is logged into the PEM server.

```
GRANT pem_comp_performance_diagnostic TO test1;
```

```
- Grant `pg_monitor`  role and `EXECUTE`  privilege on all functions of `edb_wait_
↪states`  to the database user (`test2`  in this example). The `test2`  user is␣
↪connected to the database server with EDB wait states.
```

```
# Grant `pg_monitor` role
GRANT pg_monitor TO test2;

# Grant `Execute` privilege on all functions
GRANT EXECUTE ON FUNCTION edb_wait_states_sessions TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_data TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_queries TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_samples TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_purge TO test2;
```

If the prerequisites aren’t met, an error appears when you access the Performance Diagnostic dashboard.

Using the Performance Diagnostic dashboard

To open the Performance Diagnostic dashboard, on the PEM client select **Tools > Server > Performance Diagnostic**.

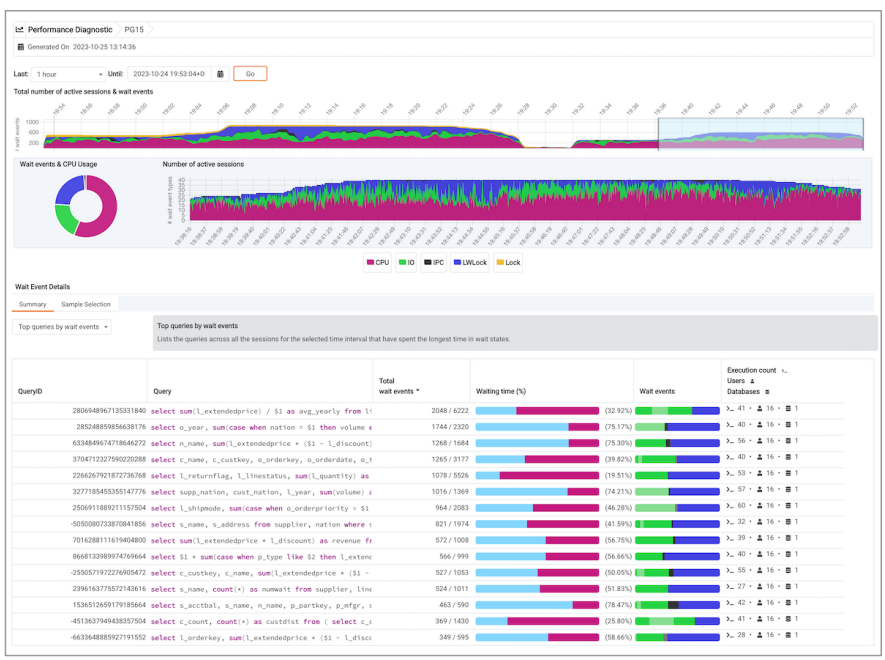


Fig. 17: Performance Diagnostic dashboard

By default, the top Performance Diagnostic graph pulls the data of the last hour, starting from the current date and time. This graph shows the time series containing the number of active sessions. Each point of this time series represents the sum of wait events and CPU usage over the previous 15 seconds. These sessions might be waiting for a wait event or using the CPU at a particular point in time. This time series is generated based on the wait event samples collected by the EDB Wait States extension.

To open the Performance Diagnostic dashboard in a new browser tab, select **Preferences > Open in New Browser Tab?**.

Performance Diagnostic dashboard

- The first graph displays the number of active sessions and wait event types for the selected duration. You can change the duration in the first graph to analyze the data for a specific time period.

By default, the duration selection in the first graph is 15 minutes. The duration can be extended up to one hour. To see the duration on a graph, select a duration from the **Last** list. To display the data for a specified date and time, select a date and time from the **Until** list.

- The next section plots the following graphs based on the selected duration in the first graph:
- Donut graph — Shows total wait event types and CPU usage according to the duration selection in the first graph. This graph can provide a better understanding of how much time those sessions spent waiting for an event.
  - Line graph — Plots a time series with each point representing the active sessions for each sample time.

To differentiate each wait event type and the CPU usage more clearly, the graph for each wait event type displays in a different color.

Select a time on the line graph to analyze the wait events. To show or hide a wait event type in all the graphs, select the graph legends. The analysis is simpler when you can see only the wait event types you want to analyze.

- The third section displays the wait event details in the Performance Diagnostic dashboard based on your selected duration in the second graph. It displays the wait event details on two tabs:
- The **Summary** tab displays the list of top 15 SQL queries with query IDs, total CPU events, percentage of CPU usage, wait events, and execution count for the selected time interval. These SQL queries are grouped into four categories:
  - **Top queries by wait events** — Lists the queries across all the sessions for the selected sample time that have spent the longest time in wait states. It displays the query ID, query, total wait events, waiting time in percentage, wait events, and the execution count along with number of users that executed that query in the number of databases.
  - **Top queries by CPU usage** — Lists the queries across all the sessions for the selected sample time that used the most CPU time. It displays the query ID, query, total CPU events, CPU usage percentage, wait events, and execution count along with the number of users that executed that query in the number of databases. The data in the table is in decreasing order of the number of CPU events.
  - **Top users by wait events** — Lists the users across all the sessions for the selected sample time whose queries have spent the longest time in wait states. It displays the usernames, database name, total wait events, waiting time percentage, wait events, and the number of queries that have spent the longest time in wait states.
  - **Top users by CPU usage** — Lists the users across all the sessions for the selected sample time whose queries used the most CPU time. It displays the username, database name, total CPU events, CPU usage percentage, wait events, and the number of queries that used the most CPU time.
- The **Sample Selection** tab displays the details of the wait events for the selected sample time in the second graph. The wait events are grouped by:
- **Queries** — Lists the queries along with query IDs and number of sessions that executed the query in the selected sample time. Select the eye in any row to display a window with details about that query. For more information, see *Query information* .
  - **Users** — Lists the details of the wait events grouped by users that executed the queries along with the number of events and execution count of the query.

- **Waits** — Lists the number of wait events belonging to each wait event type for the selected sample time. You can filter the data displayed in the rows under both the tabs. You can also sort the data alphabetically by selecting the column headers.

Query information

To open the query information window, select the eye in any row of the **Queries** grouping under the **Sample Selection** tab. This window displays in the **Query information** section a query ID and its corresponding session IDs in a list at that selected sample time. You can select the **Session ID** list for the selected query whose data you want to analyze. The details corresponding to the selected session ID and query ID appear. The **Query information** table also displays the SQL query.

The **Wait event types** section displays the total number of wait event types for the selected session ID and query ID. It shows two types of graphs:

- Donut graph — Shows the proportions of categorical data.
- Timeline bar graph — Visualizes trends in counts of wait event types over time.

To differentiate, each wait event type is represented by a different color in the bar graph.

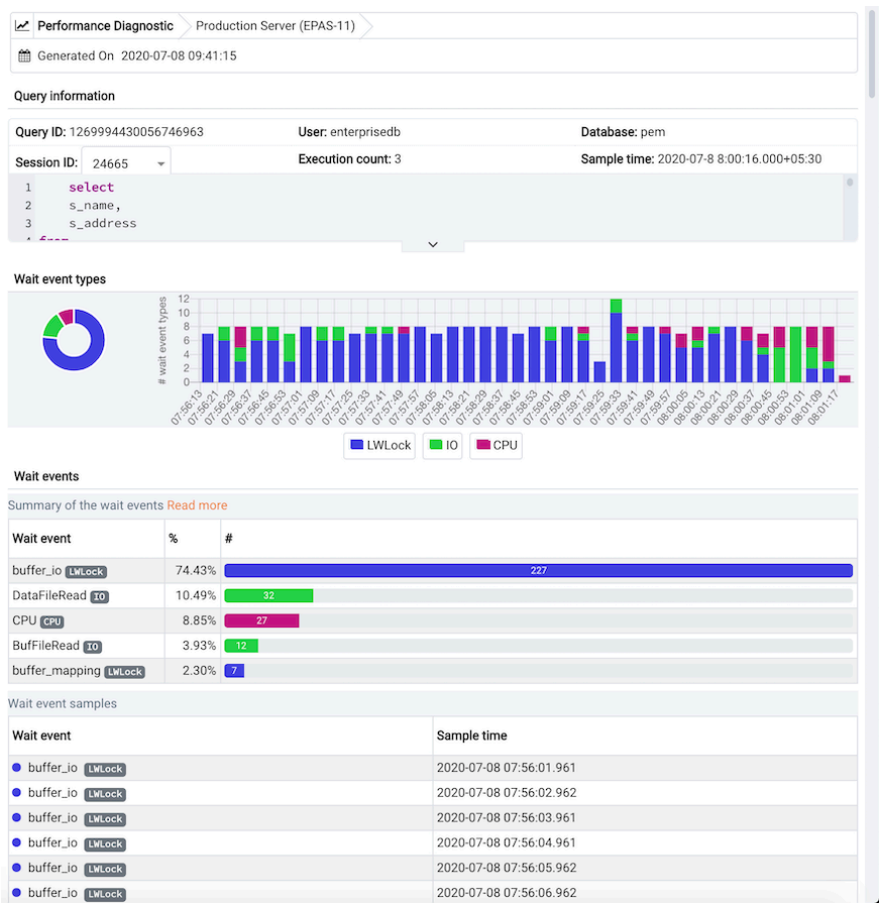


Fig. 18: Performance Diagnostic - Query dashboard

The **Wait events** section has a table displaying all the wait events that occurred during the query execution. It displays data in decreasing order by number of wait events. The second table displays the wait event with sample times that

occurred over the period of the whole query execution. It allows you to analyze the wait events during the query execution over the period of time. Also, it shows the actual samples collected by the EDB Wait States extension for that query ID and session ID.

5.10 Profiling workloads

Most RDBMS experts agree that inefficient SQL code is the leading cause of most database performance problems. The challenge for DBAs and developers is to locate that SQL code in large and complex systems and then optimize that code for better performance.

The following extensions works with PEM to allow you to profile a server's workload:

- SQL Profiler — allows a database superuser to locate and optimize inefficient SQL code
- See *Installing the SQL Profiler extension*
- See *Upgrading SQL Profiler*
- See *Using SQL Profiler*
- *Using the Index Advisor* — works with SQL Profiler by examining collected SQL statements and making indexing recommendations for any underlying tables to improve SQL response time

5.10.1 Installing the SQL Profiler extension

You must install and enable the SQL Profiler extension on each server on which you want to use it. For example, if you have a host running PostgreSQL 14 and PostgreSQL 15, you must enable the extension on each server.

SQL Profiler is supported on the same platforms as the Postgres distribution you're using. Support for the SQL Profiler extension starts with version 11 for PostgreSQL and EDB Postgres Advanced Server and version 12 for EDB Postgres Extended. See:

- EDB Postgres Advanced Server Product Compatibility
- PostgreSQL Product Compatibility
- EDB Postgres Extended Server Product Compatibility

Installing the package

Prerequisites

Before you begin the installation process:

- Install Postgres. See: - Installing EDB Postgres Advanced Server
- Installing PostgreSQL
- Installing EDB Postgres Extended Server
- Set up the repository.

Setting up the repository is a one-time task. If you have already set up your repository, you don't need to perform this step.

To set up the repository, go to EDB repositories and follow the instructions provided there.

Install the package

The syntax for the install command is:

```
sudo <package-manager> -y install edb-<postgres><postgres_version>-server-sqlprofiler
```

Where:

- <package-manager> is the package manager used with your operating system:

Package manager	Operating system
dnf	RHEL 8 and derivatives
zypper	SLES
apt-get	Debian and derivatives

- <postgres> is the distribution of Postgres you're using:

Postgres distribution	Value
PostgreSQL	pg
EDB Postgres Advanced Server	as
EDB Postgres Extended	postgresextended

- <postgres_version> is the version of Postgres you're using.

For example, to install the latest version of SQL Profiler for EDB Postgres Advanced Server 15 on a RHEL 8 platform:

```
sudo dnf -y install edb-as15-server-sqlprofiler
```

Enabling the extension

To enable the extension:

1. Edit the `postgresql.conf` file on the server you want to profile, modifying the `shared_preload_libraries` parameter:

```
shared_preload_libraries = $libdir/sql-profiler
```

1. Restart the Postgres server.
2. Create the SQL Profiler extension in your maintenance database with the following command:

```
# Connect to your maintenance database using psql
psql -d postgres -U enterprisedb
# Create the extension
CREATE EXTENSION sql_profiler;
```

Note: If you are connected to the PEM server with the PEM client before configuring SQL Profiler, you must disconnect and reconnect with the server to enable SQL Profiler functionality.

After enabling the extension, SQL Profiler is ready to use with all databases that reside on the server.

5.10.2 Using SQL Profiler

The SQL Profiler extension allows a user to locate and optimize inefficient SQL code. Microsoft's SQL Server Profiler is very similar to PEM's SQL Profiler in operation and capabilities.

SQL Profiler works with PEM to allow you to profile a server's workload. You can install and enable the SQL Profiler extension on servers with or without a PEM agent. However, you can run traces only in ad hoc mode on unmanaged servers and you can schedule them only on managed servers.

SQL Profiler captures and displays a specific SQL workload for analysis in a SQL trace. You can start and review captured SQL traces immediately or save captured traces for review later. You can use SQL Profiler to create and store up to 15 named traces.

Permissions for SQL profiler

To access the SQL profiler tool on PEM, there are two prerequisites:

1. The user logged in to the PEM GUI must either be a superuser or a member of group `pem_comp_sqlprofiler` within the PEM server database. For more information, see *pem groups* .
2. The user configured for the database server in the server tree (`Username`), must be a superuser on the database server that the trace is being run on (monitored server).

Creating a trace

You can use the Create Trace dialog box to define a SQL trace for any database on which SQL Profiler was installed and configured. To open the dialog box, select the database in the PEM client tree and select **Tools > Server > SQL Profiler > Create trace**.

Use the **Trace options** tab to specify details about the new trace:

- Provide a name for the trace in the **Name** field.
- Use the **User filter** field to specify the roles whose queries to include in the trace. Select **Select All** to include queries from all roles.
- Use the **Database filter** field to specify the databases to trace. Select **Select All** to include queries against all databases.
- Specify a trace size in the **Maximum Trace File Size** field. SQL Profiler terminates the trace when it reaches approximately the size specified.
- Select **Yes** in the **Run Now** field to start the trace when you select **Create**. Select **No** to enable fields on the **Schedule** tab.

Use the **Schedule** tab to specify scheduling details for the new trace:

- Use the **Start time** field to specify the starting time for the trace.
- Use the **End time** field to specify the ending time for the trace.
- Select **Yes** in the **Repeat?** field to repeat the trace every day at the times specified. Select **No** to enable fields on the **Periodic job options** tab.

Use the **Periodic job options** tab to specify scheduing details about a recurring trace. Use the **Days** section to specify the days when the job executes:

- Use the **Week days** field to select the days of the week for the trace to execute.
- Use the **Month days** field to select the days of the month for the trace to execute.

- Use the **Months** field to select the months when the trace executes.

Use the **Times** section to specify a schedule for the trace execution. Use the **Hours** and **Minutes** fields to specify the time when the trace executes.

After you complete the Create Trace dialog box, select **Create** to start the trace or to schedule the trace.

If you execute the trace immediately, the trace results appear in the PEM client.

Opening an existing trace

To view a previous trace, select the profiled database in the PEM client tree and select **Management > SQL Profiler > Open trace**. The Open Trace dialog box opens.

Select an entry in the trace list and select **Open** to open the selected trace in the **SQL Profiler** tab.

Filtering a trace

A filter is a named set of one or more rules, each of which can hide events from the trace view. When you apply a filter to a trace, the hidden events aren't removed from the trace but are excluded from the display.

Select **Filter** to open the Trace Filter dialog box and create a rule or set of rules that define a filter. Each rule screens the events in the current trace based on the identity of the role that invoked the event or the query type invoked during the event.

To open an existing filter, select **Open**. To define a new filter, select **Add (+)** to add a row to the table displayed on the **General** tab and provide rule details:

- Use the **Type** list to specify the trace field that the filter rule applies to.
- Use the **Condition** list to specify the type of operator for SQL Profiler to apply to the value when it filters the trace:
- Select **Matches** to filter events that contain the specified value.
- Select **Does not match** to filter events that don't contain the specified value.
- Select **Is equal to** to filter events that contain an exact match to the string specified in the **Value** field.
- Select **Is not equal to** to filter events that don't contain an exact match to the string specified in the **Value** field.
- Select **Starts with** to filter events that begin with the string specified in the **Value** field.
- Select **Does not start with** to filter events that don't begin with the string specified in the **Value** field.
- Select **Less than** to filter events that have a numeric value less than the number specified in the **Value** field.
- Select **Greater than** to filter events that have a numeric value greater than the number specified in the **Value** field.
- Select **Less than or equal to** to filter events that have a numeric value less than or equal to the number specified in the **Value** field.
- Select **Greater than or equal to** to filter events that have a numeric value greater than or equal to the number specified in the **Value** field.
- Use the **Value** field to specify the string, number, or regular expression to search for.

After you finish defining a rule, select **Add (+)** to add another rule to the filter. To delete a rule from a filter, select the rule and select **Delete**.

Select **Save** to save the filter definition to a file without applying the filter. To apply the filter, select **OK**.

Deleting a trace

To delete a trace:

1. Select the profiled database in the PEM client tree.
2. Select **Management > SQL Profiler > Delete trace(s)**.
3. In the Delete Traces dialog box, select the icon to the left of a trace name to mark one or more traces for deletion.
4. Select **Delete**. The PEM client acknowledges that the selected traces were deleted.

Viewing scheduled traces

To view a list of scheduled traces, select the profiled database in the PEM client tree. Select **Management > SQL Profiler > Scheduled traces**.

The Scheduled Traces dialog box displays a list of the traces that are awaiting execution. Select **Edit** to the left of a trace name to see detailed information about the trace:

- The **Status** field lists the status of the current trace.
- The **Enabled?** switch displays Yes if the trace is enabled, No if it is disabled.
- The **Name** field displays the name of the trace.
- The **Agent** field displays the name of the agent responsible for executing the trace.
- The **Last run** field displays the date and time of the last execution of the trace.
- The **Next run** field displays the date and time of the next scheduled trace.
- The **Created** field displays the date and time that the trace was defined.

5.10.3 Using the Index Advisor

Index Advisor is distributed with EDB Postgres Advanced Server. Index Advisor works with SQL Profiler by examining collected SQL statements and making indexing recommendations for any underlying tables to improve SQL response time. Index Advisor works on all DML (INSERT, UPDATE, DELETE) and SELECT statements that are invoked by a superuser.

Diagnostic output from the Index Advisor includes:

- Forecasted performance benefits from any recommended indexes
- The predicted size of any recommended indexes
- DDL statements you can use to create the recommended indexes

Index Advisor can make indexing recommendations based on trace data captured by SQL Profiler. To open Index Advisor, select one or more queries in the SQL Profiler Trace Data pane and select **Index Advisor** from the toolbar. For more information about configuring and using Index Advisor, see *Setting the EDB Postgres Advanced Server instance service ID* .

Note: Index Advisor can't analyze statements invoked by a non-superuser. If you attempt to analyze statements invoked by a non-superuser, the server log includes the following error:

```
ERROR: access to library "index_advisor" is not allowed
```

Note: We recommend that you disable Index Advisor while using the pg_dump functionality.

5.11 Viewing system reports

You can generate the System Configuration report and Core Usage report for all locally and remotely managed servers. To generate this report, select **Management > Reports**.

Reports has the following options:

- Alert History Report (JSON)
- System Configuration Report (JSON)
- System Configuration Report (HTML)
- Core Usage Report (JSON)
- Core Usage Report (HTML)

Only superusers or the users with the pem_admin role can download the System Configuration or Core Usage reports.

Information in these reports shows the latest probe run time.

5.11.1 Alert History report

The Alert History report provides detailed information about the alerts history at the agent or server level in JSON format.

Select **Management > Reports > Alert History Report**. From the dialog box, select your options:

- **Agent/Server Name** — Select the agents and servers from the list. The **Overall System Report** option is also available to generate a report for all the registered and active agents and servers.
- **Timeframe** — Select the required timeframe from the list.
- **Alert Types** — Select the alert types you want to generate the alert history report for.

5.11.2 System Configuration report

The System Configuration report provides detailed information about the PEM Agents group, PEM Server Directory group, and custom groups listed under the browser tree. These groups can contain PEM server, PEM agent, and database servers. You can download this report in HTML and JSON formats.

The Postgres Enterprise Manager Summary provides details about:

- The Postgres Enterprise Manager backend database server version
- Application version
- User name accessing the application
- Python version
- Flask version
- Platform specific information

The summary provides information about the number of agents and servers. The Group: PEM Agents panel provides details about the PEM agent, CPU cores, disk utilization, and memory information.

The Group: PEM Server Directory panel provides details about:

- Database server version
- Host
- Port
- Database name
- Database size
- Tablespace size

The group server name depends on the group name to which the server is added.

5.11.3 Core Usage report

The Core Usage report provides detailed information about the number of cores specific to:

- Server type
- Database version
- Platform and group name

The report also gives detailed information about locally managed servers:

- Type
- Host
- Port
- Platform
- Cores
- RAM

5.12 PEM command line interface

The PEM CLI allows you to export and import the alert templates or probes from one PEM server to another.

You can install the PEM CLI using the package `edb-pem-cli-8.5.0-1-<platform_name>` separately from the `edb.repo` repository. It creates the binary file `/usr/edb/pem/cli/bin/pem` and the license file `/usr/edb/pem/edb-pem-cli-license.txt` .

After installing the PEM CLI package, use the `./pem -h` command to see all the available options.

```
$ ./pem -h
__OUTPUT__
PEM CLI (Postgres Enterprise Manager Command Line Interface)
Usage: ./pem [OPTIONS] [SUBCOMMAND]

Options:
  -h,--help                   Print this help message and exit
```

(continues on next page)

(continued from previous page)

```
  -v,--version                Show the app version and exit

Logging:
  --log-file [/home/asheshvashi/.pem-cli.log]
                              Set the log file

                              NOTE: Directory must exist

  -l,--log-level :value in {FATAL->1,ERROR->2,WARNING->3,DEBUG->5,INFO->4,VERBOSE->6} OR
↪{1,2,3,5,4,6} [INFO]
                              Set the log level

PEM Information:
  -p,--pem-url REQUIRED (Env:PEM_SERVER_URL)
                              Set the URL for accessing Postgres Enterprise Manager
                              e.g.
                              https://127.0.0.1:8443/pem
                              https://pem_host:8443/pem

  -u,--pem-user REQUIRED (Env:PEM_API_USER)
                              User to access the PEM REST API

  -f,--password-file (Env:PEM_PASSWORD_FILE)
                              Location of the file, which contains the password.
                              The permission on a password file must disallow any access␣
↪to world or group.

                              NOTE:
  - The first non-empty line in the file will be treated as a
                                password.
  - Environment variable PEM_API_PASSWORD can be used for
                                providing the password instead of the password file

  -i,--insecure               Set the insecure connection


Subcommands:
  import-alert-templates      Import the alert template(s) into the PEM server.
  list-alert-templates        List the alert templates in the PEM server.
  export-alert-templates      Export the custom (user-defined) alert templates from the␣
↪PEM server.
  import-probes               Import the probe(s) into the PEM server.
  export-probes               Export the custom (user-defined) probes from the PEM␣
↪server.
  list-probes                 List the probes in the PEM server.
```

You can see all the available options with the `import-alert-templates` subcommand:

```
$ ./pem import-alert-templates -h
__OUTPUT__
Import the alert template(s) into the PEM server.
Usage: ./pem import-alert-templates [OPTIONS]
```

(continues on next page)

(continued from previous page)

```
Options:
  -h,--help                   Print this help message and exit
  -i,--in-file                Alert template file
  --ignore-existing           Ignore the existing alert template in the PEM server
  --ignore-existing-probes    Ignore the existing probes in the PEM server
```

You can see all the available options with the `list-alert-templates` subcommand:

```
$ ./pem list-alert-templates -h
__OUTPUT__
List the alert templates in the PEM server.
Usage: ./pem list-alert-templates [OPTIONS]

Options:
  -h,--help                   Print this help message and exit
  -c,--alert-template-category :value in {ALL->2,CUSTOM->1,SYSTEM->0} OR {2,1,0} [CUSTOM]
                              List down the alert templates of this category
```

You can see all the available options with the `export-alert-templates` subcommand:

```
$ ./pem export-alert-templates -h
__OUTPUT__
Export the custom (user-defined) alert templates from the PEM server
Usage: ./pem export-alert-templates [OPTIONS]

Options:
  -h,--help                   Print this help message and exit
  -i,--id INT ...             List of id for the custom alert templates to be exported
  -o,--out-file [Standard output (console)]
                              Output file name
  -w,--overwrite              Overwrite the existing output file
```

You can see all the available options with the `import-probes` subcommand:

```
$ ./pem import-probes -h
__OUTPUT__
Import the probe(s) into the PEM server.
Usage: ./pem import-probes [OPTIONS]

Options:
  -h,--help                   Print this help message and exit
  -i,--in-file                Probes file
  --ignore-existing           Ignore the existing custom probes in the PEM server
```

You can see all the available options with the `export-probes` subcommand:

```
$ ./pem export-probes -h
__OUTPUT__
Export the custom (user-defined) probes from the PEM server
Usage: ./pem export-probes [OPTIONS]
```

(continues on next page)

(continued from previous page)

```
Options:
  -h,--help                   Print this help message and exit
  -i,--probe-id INT ...       List of the custom probe-ids to be exported
  -o,--out-file [Standard output (console)]
                              Output file name
  -w,--overwrite              Overwrite the existing output file
```

You can see all the available options with the `list-probes` subcommand:

```
$ ./pem list-probes -h
__OUTPUT__
List the probes in the PEM server.
Usage: ./pem list-probes [OPTIONS]

Options:
  -h,--help                   Print this help message and exit
  -c,--probe-category :value in {ALL->2,CUSTOM->1,SYSTEM->0} OR {2,1,0} [CUSTOM]
                              List down the probes of this category
```

5.12.1 Example

To list all the probes (system as well as custom probes) use the `-c all` option:

```
$ PEM_PASSWORD_FILE=/tmp/pass PEM_API_USER=postgres PEM_SERVER_URL=https://172.17.0.4/
↪pem ./pem -i list-probes -c all
```

Where:

- `PEM_PASSWORD_FILE=/tmp/pass` is the password file with the password of the postgres user.
- `PEM_API_USER=postgres` is the username through which you' re running the PEM CLI.
- `PEM_SERVER_URL=https://172.17.0.4/pem` is the URL of the PEM server containing the probes.
- `-i list-probes` is the `insecure` option with the `list-probes` subcommand to allow insecure connection and to list the probes.
- `-c all` is the `-c` option with the `all` value to list all the system as well as custom probes.

5.13 PEM Rest API

EDB Postgres Enterprise Manager supports the REST API feature. The REST API feature offers effective data read and write capabilities and access to various functionalities without logging in to the PEM UI. For more information, see REST API usage .

CHAPTER 6

Developing

6.1 Using the Query tool

PEM contains a feature-rich interactive development environment (IDE) that allows you to issue ad hoc SQL queries against Postgres servers.

You can access the Query tool by selecting **Tools > Query tool** or through the context menu of select nodes of the Browser tree. The Query tool allows you to:

- Issue ad hoc SQL queries.
- Execute arbitrary SQL commands.
- Edit the result set of a SELECT query if it' s *Data Output tab* .
- Configure and display current connection and transaction status.
- Save the data displayed in the output panel to a CSV file.
- Review the execution plan of a SQL statement in either text, graphical, or table format (similar to https://explain.depesz.com).
- View analytical information about a SQL statement.

The Query tool features two panels:

- The upper panel displays the SQL editor. You can use the panel to enter, edit, or execute a query. It also shows the **History** tab, which you can use to view the queries that ran in the session. You can use the scratch pad to hold text snippets during editing. To reopen a closed scratch pad or open a new one, right-click the SQL editor and select **Add Panel**. You can also open new scratch pads from other panels using the same technique.
- The lower panel displays the Data Output panel. The tabbed panel displays:
- The result set returned by a query
- Information about a query' s execution plan
- server messages related to the query' s execution

- Any asynchronous notifications received from the server.

6.1.1 The Query tool toolbar

The toolbar uses context-sensitive icons that provide shortcuts to frequently performed tasks.

Icon	Behavior	Shortcut
Open File	Display a previously saved query in the SQL editor.	Accesskey + O
Save	Save a query or access the **Save** menu: - Select **Save** to save the selected content of the SQL editor panel in a file. - Select **Save As** to open a new browser dialog box and specify a new location to save the selected content of the SQL editor panel.	Accesskey + S
Save Data Changes	Save the data changes (insert, update, or delete) in the Data Output Panel to the server.	F6
Find	Search, replace, or navigate the code displayed in the SQL editor: - Select **Find** to provide a search target and search the SQL editor contents. - Select **Find next** to locate the next occurrence of the search target. - Select **Find previous** to move to the last occurrence of the search target. - Select **Pesistent find** to identify all occurrences of the search target in the editor. - Select **Replace** to locate and replace (with prompting) individual occurrences of the target. - Select **Replace all** to locate and replace all occurrences of the target in the editor. - Select **Jump** to navigate to the next occurrence of the search target.	Cmd+F Cmd+G Cmd+Shift+G Cmd+Shift+F Alt+G
Copy	Copy the content that's currently highlighted in the Data Output panel when in View/Edit data mode.	Accesskey + C
Paste	Paste a copied row into a new row when in View/Edit data mode.	Accesskey + P
Delete	Mark selected rows for deletion. Select **Save Data Changes** to update the content with your deletions.	Accesskey + D
Edit	Use options on the **Edit** menu to access text editing tools. The options operate on the text displayed in the SQL editor panel when in Query tool mode: - Select **Indent Selection** to indent the currently selected text. - Select **Unindent Selection** to remove indentation from the currently selected text. - Select **Inline Comment Selection** to enclose any lines that contain the selection in SQL-style comment notation. - Select **Inline Uncomment Selection** to remove SQL-style comment notation from the selected line. - Select **Block Comment** to enclose all lines that contain the selection in C-style comment notation. This option acts as a toggle.	Tab Shift+Tab Cmd+/ Cmd+. Shift+Cmd+/

continues on next page

Table 1 – continued from previous page

Icon	Behavior	Shortcut
Filter	Set filtering and sorting criteria for the data when in View/Edit data mode. Select the down arrow to access other filtering and sorting options: -Select **Sort/Filter** to open the sorting and filtering dialog box. - Select **Filter by Selection** to show only the rows containing the values in the selected cells. - Select **Exclude by Selection** to show only the rows that don't contain the values in the selected cells. - Select **Remove Sort/Filter** to remove any previously selected sorting or filtering options.	Accesskey + F
Limit Selector	Set a value for the maximum number of rows in a dataset.	Accesskey + R
Stop	Cancel the currently running query.	Accesskey + Q
Execute/Refresh	Execute or refresh the query selected in the SQL editor panel. Select the down arrow to access other execution options: - Select **Auto-Rollback** to roll back a transaction in case an error occurs during the transaction. - Select **Auto-Commit** to commit each transaction. Any changes made by the transaction are visible to others and durable in the event of a crash.	F5
Explain	View an explanation plan for the current query. The result of *EXPLAIN* is displayed graphically on the **Explain** tab of the output panel and in text form on the **Data Output** tab.	F7
Explain analyze	Invoke an *EXPLAIN ANALYZE* command on the current query. From the **Explain Options** menu: - Select **Verbose** to display more information about the query plan. - Select **Costs** to include information on the estimated startup and total cost of each plan node as well as the estimated number of rows and the estimated width of each row. - Select **Buffers** to include information on buffer usage. - Select **Timing** to include information about the startup time and the amount of time spent in each node of the query. - Select **Summary** to include the summary information about the query plan.	
Commit	Commit the transaction.	Shift+CTRL+M
Rollback	Roll back the transaction.	Shift+CTRL+R
Clear	Use options on the **Clear** menu to erase display contents: - Select **Clear Query Window** to erase the content of the SQL editor panel. - Select **Clear History** to erase the content of the **History** tab.	Accesskey + L
Download as CSV	Download the result set of the current query to a comma-separated list. You can specify the CSV settings through the **Preferences -> SQL editor -> CSV output** dialog box.	F8
Macros	Create, edit, or clear the macros by selecting **Manage Macros** .	

6.1.2 The SQL editor panel

The SQL editor panel is a workspace where you can manually provide a query, copy a query from another source, or read a query from a file. The SQL editor features syntax coloring and auto-completion.

To use auto-complete, begin typing your query. When you want the editor to suggest object names or commands that might be next in your query, press **Control+Space**. For example, type `\*SELECT \* FROM\*` (with a trailing space), and then press **Control+Space** to select from a menu of auto-complete options.

After entering a query, select **Execute/Refresh** from the toolbar. The database server receives the complete contents of the SQL editor panel to execute. To execute only a section of the code that's displayed in the SQL editor, select the text that you want the server to execute, and select **Execute/Refresh**.

The message returned by the server when a command executes is displayed on the **Messages** tab. If the command is successful, the **Messages** tab displays execution details.

The **Edit** menu helps with code formatting and commenting:

- Use auto-indent to indent text to the same depth as the previous line by pressing **Return**.
- Block indent text by selecting two or more lines and pressing **Tab**.
- Implement or remove SQL-style or toggle C-style comment notation within your code.

You can also drag certain objects from the tree to save time spent typing long object names. Text containing the object name is fully qualified with the schema name. Double quotes are added if required. For functions and procedures, the function name along with parameter names are pasted into the Query tool.

6.1.3 The Data Output panel

The Data Output panel displays data and statistics generated by the most recently executed query.

Data Output tab

The **Data Output** tab displays the result set of the query in a table format. You can:

- Select and copy from the result set.
- Use the **Execute/Refresh** options to retrieve query execution information and set query execution options.
- Select **Download as CSV** to download the content of the **Data Output** tab as a comma-delimited file.
- Edit the data in the result set of a `SELECT` query if it's updatable.

A result set is updatable if:

- All columns are either selected directly from a single table or they aren't actually a table column (for example, the concatenation of two columns). You can edit only columns that are selected directly from the table. Other columns are read-only.
- All the primary key columns or OIDs of the table are selected in the result set.

Any columns that are renamed or selected more than once are also read-only.

Note: To work with an updatable query result set, you must have psycopg2 driver version 2.8 or later installed.

Editable and read-only columns are identified using pencil and lock icons in the column headers.

An updatable result set is similar to the data grid in View/Edit Data mode, and you can modify it in the same way.

If auto-commit is off, data changes are made as part of the ongoing transaction. If no transaction is ongoing, a new one is started. The data changes aren't committed to the database unless the transaction is committed.

If any errors occur while saving (for example, trying to save a NULL into a column with a NOT NULL constraint), the data changes are rolled back to a savepoint to ensure any previously executed queries in the ongoing transaction aren't rolled back.

All rowsets from previous queries or commands that are displayed in the Data Output panel get discarded when you invoke another query. Open another Query tool browser tab to keep your previous results available.

Explain tab

To generate the Explain or Explain Analyze plan of a query, select **Explain** or **Explain Analyze** in the toolbar.

You can select more options related to **Explain** and **Explain Analyze** from the menu.

Note: PEM generates the Explain Analyze plan in JSON format.

On successful generation of an Explain plan, three tabs/panels appear under the **Explain** panel.

Graphical tab

To download the plan as an SVG file, select **Download** in the top-left corner of the Explain canvas. **Download as SVG** isn't supported on Internet Explorer.

The query plan that accompanies **Explain Analyze** is available on the **Data Output** tab.

Analysis tab

The **Analysis** tab shows the plan details in table format, generating a format similar to the one available at `explain.depsez.com`. Each row of the table represents the data for an Explain Plan node. The output can contain the node information, exclusive timing, inclusive timing, actual versus planned rows, actual rows, planned rows, or loops. Child rows of the selected row are marked with an orange dot.

If the percentage of the exclusive/inclusive timings of the total query time is:

- Greater than 90, red appears
- Greater than 50, orange appears
- Greater than 10, yellow appears

If the planner has misestimated the number of rows (actual verse planned) by:

- 10 times, yellow appears
- 100 times, orange appears
- 1000 times, red appears

Statistics tab

The **Statistics** tab displays information in two tables:

- Statistics per Node Type tells you how many times each node type was referenced.
- Statistics per Table tells you how many times each table was referenced by the query.

Messages tab

Use the **Messages** tab to view information about the most recently executed query. If the server returns an error, the error message appears on the **Messages** tab, and the syntax that caused the error is underlined in the SQL editor. If a query succeeds, the **Messages** tab shows how long the query took to complete and how many rows were retrieved.

Notifications tab

Use the **Notifications** tab to view the notifications using the PostgreSQL Listen/Notify feature. For more details, see the PostgreSQL documentation .

For example:

1. Execute `LISTEN "foo"` in first Query tool session.

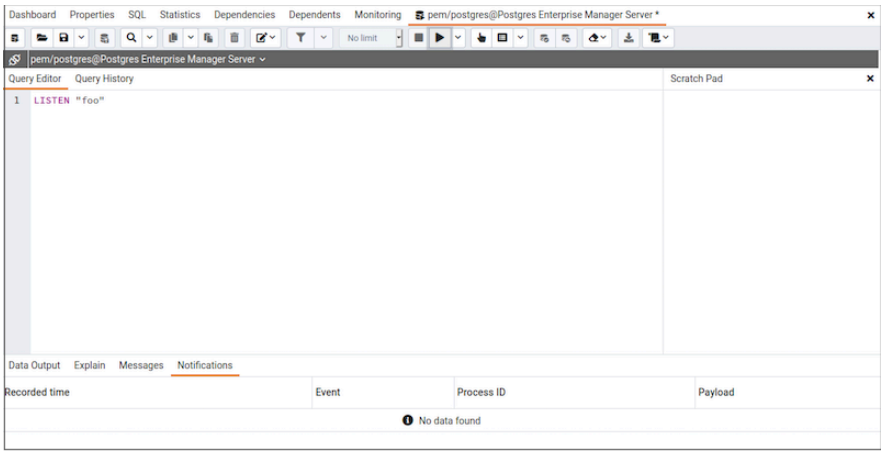


Fig. 1: Query tool - notifications listen

2. In the another Query tool session, execute the `Notify` command or `pg_notify` function to send the notification of the event together with the payload.
3. You can observe the **Notification** tab in the first Query tool session where it shows the recorded time, event, process ID, and the payload of the channel.

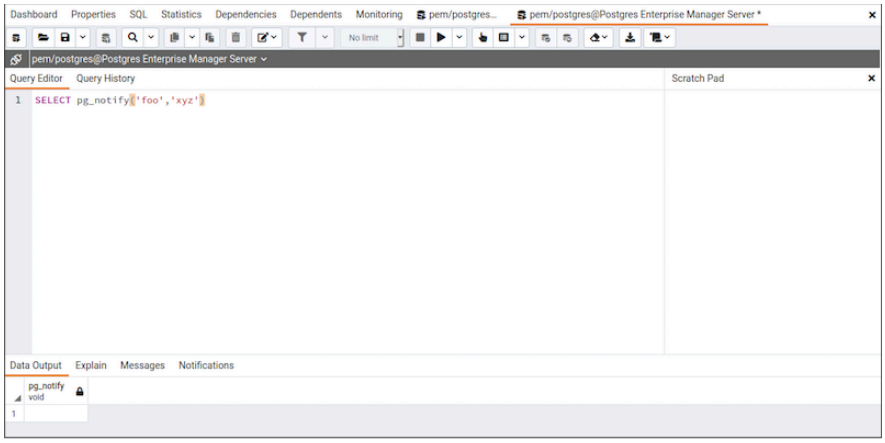


Fig. 2: Query tool - notifications notify

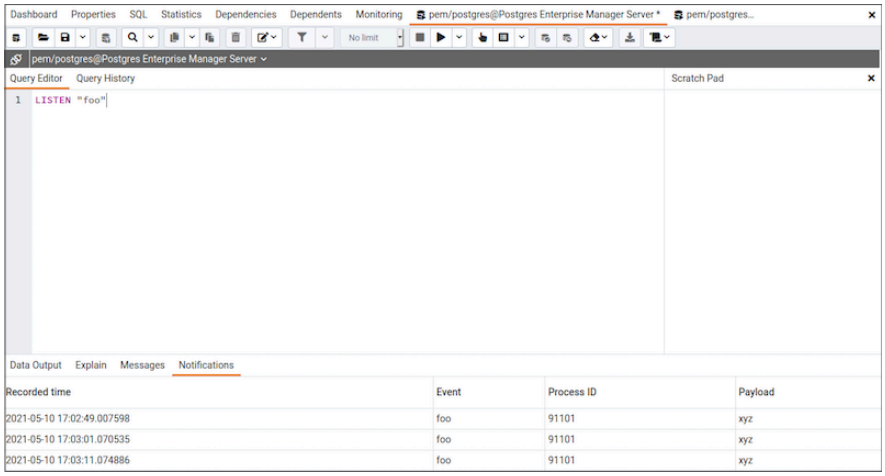


Fig. 3: Query tool - Output Notifications panel

6.1.4 Query History panel

Use the **Query History** tab to review activity for the current session. The **Query History** tab displays information about recent commands including:

- The date and time that a query was invoked.
- The text of the query.
- The number of rows returned by the query.
- The amount of time it took the server to process the query and return a result set.
- Messages returned by the server (not noted on the **Messages** tab).
- The source of the query (indicated by icons corresponding to the toolbar).

You can show or hide the queries generated internally by pgAdmin (during View/Edit Data or Save Data operations).

To erase the content of the **Query History** tab, select **Clear > Clear history**.

Query history is maintained across sessions for each database on a per-user basis when running in Query tool mode. In View/Edit Data mode, history isn't retained. By default, the last 20 queries are stored for each database. You can adjust this in `config_local.py` by overriding the `MAX_QUERY_HIST_STORED` value.

6.1.5 Connection status

Use **Connection Status** to view the current connection and transaction status by selecting the status in the Query tool.

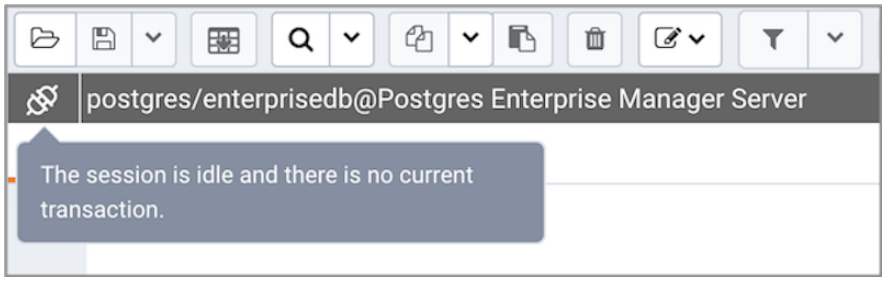


Fig. 4: Query tool - Connection Status button

6.1.6 Macros

Query tool macros enable you to execute predefined SQL queries by pressing a single key. Predefined queries can contain the placeholder `$SELECTION$` . When the macro executes, the placeholder is replaced with the currently selected text in the Query Editor pane of the Query tool.

To create a macro:

1. In the Query tool, select **Macros > Manage Macros**.
2. Select the key you want to use. Enter the name of the macro and the query.
3. Optionally, include the selection placeholder.
4. Select **Save**.

To clear a macro, in the Manage Macros dialog box, select the macro and select **Clear**. Respond **Yes** to the prompt.

To clear all macros, select **Clear** next to **Key**. Respond **Yes** to the prompt.

To execute a macro, select the shortcut keys, or select it from the **Macros** menu.

6.1.7 Accessing in-app help for UI details

The in-app help provides comprehensive details about the PEM web interface, including the menu items, charts and graphs on the dashboards, and the options on the dialog boxes. The in-app help also includes instructions for creating database objects.

- To access the landing page for the help, select **Help > Online help**.
- To access context-sensitive help for a dialog box, select **?**.

6.2 Using the Schema Diff tool

Schema Diff compares schema objects between two database schemas. Use the **Tools** menu to access Schema Diff.

Schema Diff allows you to:

- Compare and synchronize the database schemas (from source to target).
- Visualize the differences between database schemas.
- List the differences in SQL statement for target schema objects.
- Generate synchronization scripts.

To open the selection panel, select **Tools > Schema Diff**. Select the source and target servers, databases, and schemas to compare. After selecting the objects, select **Compare**.

Note: The source and target databases must use the same major version.

You can open multiple copies of Schema Diff in individual tabs. To specify whether to open Schema Diff in a new browser tab, select the Preferences dialog box. Set **Open in new browser tab** to **true**.

The Schema Diff panel is divided into two panels: an Object Comparison panel and a DDL Comparison panel.

6.2.1 Schema Diff Object Comparison panel

In the Object Comparison panel, you can select the source and target servers of the same major version, databases, and schemas to compare. You can select any server listed under the browser tree whether it's connected or disconnected. If you select a server that isn't connected, then you must enter the password before using that server.

Select the databases to compare. The databases can be the same or different, and from the same or different servers.

Select the source and target schemas to compare.

After you select servers, databases, and schemas, select **Compare** to get the comparison results.

Use the lists of **Functions**, **Materialized Views**, **Tables**, **Trigger Functions**, **Procedures**, and **Views** to view the DDL statements of all the schema objects.

To filter the schema objects, select **Filter** in the upper-right corner of the Object Comparison panel. Filter the schema objects according to these criteria:

- **Identical** — If the object is found in both schemas with the same SQL statement, then the comparison result is identical.
- **Different** — If the object is found in both schemas with different SQL statements, then the comparison result is different.
- **Source Only** — If the object is found only in source schema and not in target schema, then the comparison result is source only.
- **Target Only** — If the object is found only in target schema and not in source schema, then the comparison result is target only.

Select any of the schema objects in the Object Comparison panel to display the DDL statements for that object in the DDL Comparison panel.

6.2.2 Schema Diff DDL Comparison panel

The DDL Comparison panel displays three columns:

- The first column displays the DDL statement of the object from the source schema.
- The second column displays the DDL statement of the object from the target schema.
- The third column displays the difference in the SQL statement of the target schema object.

To check for differences in the SQL statements, review the DDL statements of all the schema objects.

The Schema Diff tool can generate a SQL script with the differences found in the target schema object. The SQL script compares the target schema object to the SQL statement of the source schema object. To generate the script:

1. In the Object Comparison panel, select the check boxes of the schema objects.
2. Select **Generate Script**.

To open the Query tool in a new tab and display the differences in the SQL statement in the Query Editor:

1. Select the schema objects.
2. Select **Generate Script**.

If you select the schema object to check the difference generated in the DDL Comparison panel but don' t select the check box of the schema object, then PEM opens the Query tool in a new tab. The Query tool displays the differences in the SQL statements in the Query Editor.

To copy the difference generated in the DDL Comparison panel, select **Copy**.

Apply the SQL Statement in the target schema to synchronize the schemas.

6.3 Using the ERD tool

The Entity-Relationship Diagram (ERD) tool is a database design tool that provides a graphical representation of database tables, columns, and inter-relationships. ERD can give sufficient information for the database administrator to follow when developing and maintaining the database. To access the ERD tool, select **Tools > ERD Tool**. The ERD tool allows you to:

- Design and visualize the database tables and their relationships.
- Add notes to the diagram.
- Align the tables and links for a cleaner look.

- Save the diagram and open it later to continue working on it.
- Generate ready-to-run SQL from the database design.
- Generate the database diagram for an existing database.

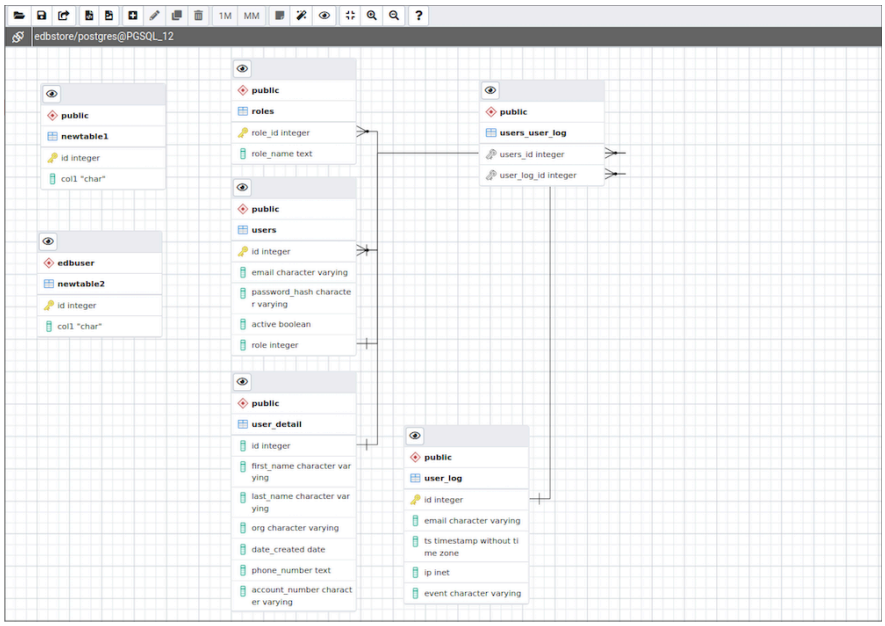


Fig. 5: ERD Tool

You can open multiple copies of the ERD tool in individual tabs simultaneously. To close a copy of the ERD tool, select the **X** in the upper-right corner of the tab bar.

6.3.1 Toolbar

The ERD Tool toolbar uses context-sensitive icons that provide shortcuts to frequently performed tasks.

6.3.2 File options

Icon	Behavior	Shortcut
Open File	Load a saved diagram.	Ctrl + O
Save	Save changes to a saved diagram or save the diagram to a file.	Ctrl + S
Save as	Open a new dialog box and specify a new location to save the diagram.	Ctrl + Shift + S

6.3.3 Export options

Icon	Behavior	Shortcut
Generate SQL	Generate the DDL SQL for the diagram and open a Query tool with the generated SQL ready for execution.	Option + Ctrl + S
Download image	Save the ERD diagram in an image format.	Option + Ctrl + I

6.3.4 Editing options

Icon	Behavior	Shortcut
Add table	Add a new table to the diagram. In the table dialog box, enter the table details.	Option/Alt + Ctrl + A
Edit table	Edit a selected table on the diagram. In the table dialog box, enter the table details.	Option/Alt + Ctrl + E
Clone table	Clone the complete table structure, name it with a auto-generated name, and put it in the diagram.	Option/Alt + Ctrl + C
Drop table/link	Drop a selected table or link.	Option/Alt + Ctrl + D

6.3.5 Table relationship options

Icon	Behavior	Shortcut
1M	Open a one-to-many relationship dialog box to add a relationship between two tables. The selected table is the referencing table and has the *many* endpoint of the link.	Option/Alt + Ctrl + O
MM	Open a many-to-many relationship dialog box to add a relationship between two tables. This option creates a linked table from the selected columns of the two relating tables.	Option/Alt + Ctrl + M

6.3.6 Utility Options

Icon	Behavior	Shortcut
Add/Edit note	Make notes on table nodes while designing the database.	Option/Alt + Ctrl + N
Auto align	Align all tables and links for a cleaner look.	Option/Alt + Ctrl + L
Show details	Toggle the column details. You can also choose to show few or more column details.	Option/Alt + Shift + D

6.3.7 Zoom options

Icon	Behavior	Shortcut
Zoom to fit	Zoom in/out automatically and fit all the tables to the view.	Option/Alt + Shift + F
Zoom in	Zoom in on the diagram.	Option/Alt + Shift + "+"
Zoom out	Zoom out on the diagram.	Option/Alt + Shift + "-"

6.3.8 Table dialog

The table dialog box allows you to:

- Change the table structure details.
- Edit an existing table or add a new one.
- See Table dialog in the PEM client online help for information on different fields.

6.3.9 Table node

The table node shows table details in a graphical representation:

- On the top bar:
- To show column details, select **Details toggle**.
- When you add a note, **Note** becomes enabled. Select **Note** to changes notes.
- The first row displays the table' s schema name, like `public` in the example.
- The second row displays the table' s name, like `users` in the example.
- The remaining rows dispalys the table' s column names and their data types.
- A column that is a primary key displays as a lock key, like `id integer` in the example.
- Other columns diplay as a green column, like `email character varying` in the example.
- Select the node to drag it on the canvas.
- To open the table dialog box and change the table details, either double-click the table node or select **Edit** on the toolbar. See Table dialog in the PEM client online help for information on different fields.

6.3.10 One-to-many link dialog box

The one-to-many link dialog box allows you to:

- Add a foregin key relationship between two tables.
- Select a Local Table that references another table. The Local Table has the `many` end point.
- Select a Local Column that references another column.
- Select the Referenced Table. The Referenced Table has the `one` end point.
- Select the Referenced Column.

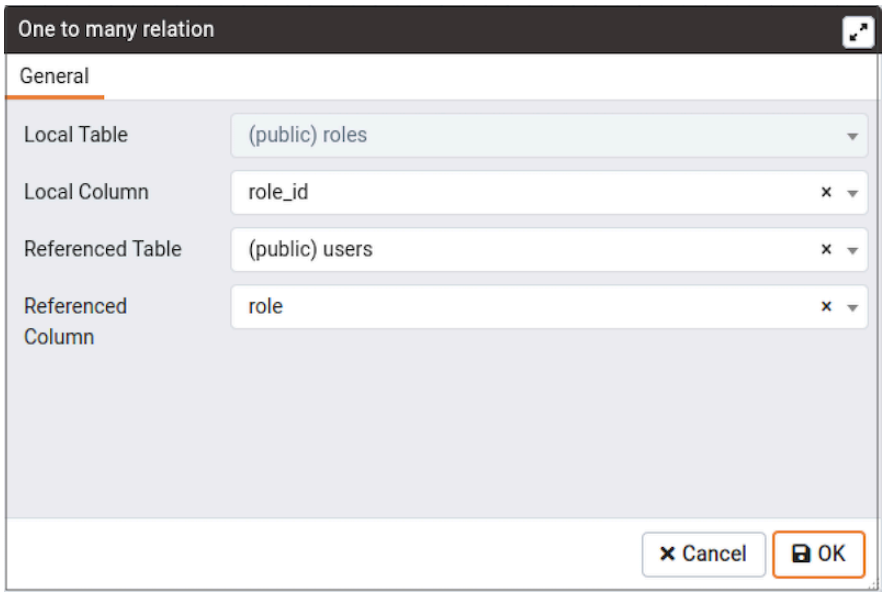


Fig. 6: ERD Tool - One-to-Many Dialog Box

6.3.11 Many-to-many link dialog box

The many-to-many link dialog box allows you to:

- Add a many-to-many relationship between two tables.
- Create a relation table with linked columns that are derived from the two tables.
- Select the Left Table, which is the first table that is linked. The Left Table has the one endpoint of the link with the new relation table.
- Select the Left table Column, which is the first table' s column. The Left table Column is a primary key.
- Select the Right Table, which is the second table that is linked. The Right Table has the one endpoint of the link with the new relation table.
- Select the Right table Column, which is the second table' s column. The Right table Column is a primary key.

6.3.12 Table link

The table link shows the relationship between tables:

- The single-line endpoint of the link shows the column that is the reference.
- The three-line endpoint of the link shows the column that references.
- If one of the columns that is a reference or that references is removed from the table, then the link drops.
- Select the link to drag it on the canvas.

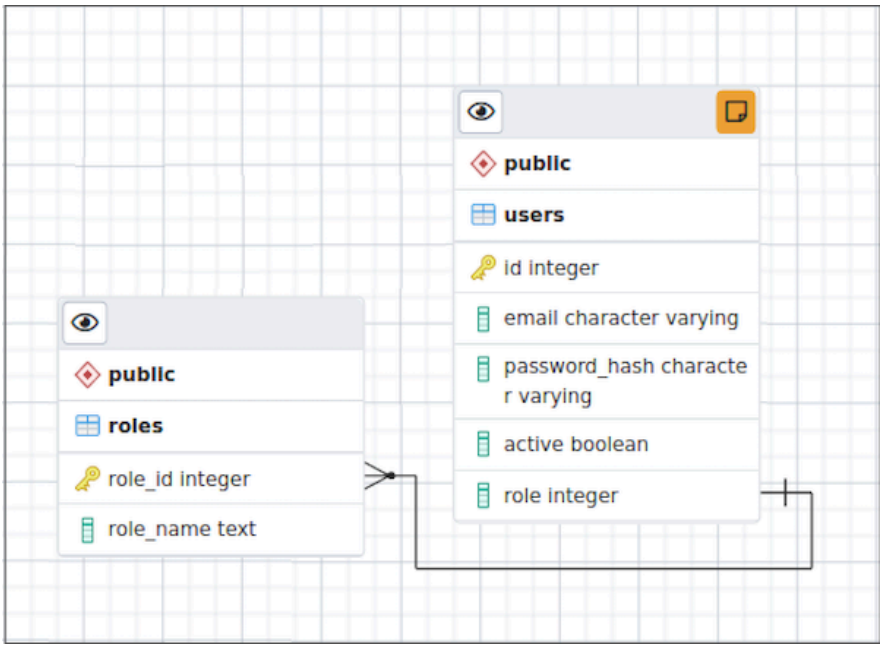


Fig. 7: ERD Tool - Table Link

6.3.13 Table notes

To add a note to a table:

1. On the toolbar, select **Add/Edit note**.
2. In the window, enter your note.

When a note is added to a table, **Note** becomes enabled on the Table node. To check or update notes, select **Note**.