



Postgres Enterprise Manager Administration Guide

Release 8.6.0

EnterpriseDB Corporation

Oct 27, 2022

CONTENTS

1	Changing the default port	3
1.1	For RHEL and CentOS	3
1.2	For Debian and Ubuntu	4
2	Registering a Postgres server	5
2.1	Manually registering a database server	5
2.2	Automatic server discovery	10
2.3	Using the pemworker utility to register a server	11
2.3.1	Using the pemworker utility to unregister a server	12
2.4	Verifying the connection and binding	13
2.5	Registering Postgres clusters on AWS	13
2.5.1	Monitoring a Postgres cluster running on AWS EC2	13
2.5.2	Monitoring a Postgres cluster running on AWS RDS	14
3	Registering a PEM agent	15
3.1	Unregistering a PEM agent	16
3.2	Setting PEM agent configuration parameters	17
3.3	Using a nonroot user account to register a PEM agent	17
4	Managing a PEM server	21
4.1	Starting and stopping the PEM server and agents	21
4.2	Controlling the PEM server or PEM agent on Linux	21
4.3	Controlling the PEM server or PEM agent on Windows	22
4.4	Controlling the HTTPD server	22
4.5	Managing PEM authentication	23
4.6	Editing the PEM server configuration	23
4.7	Managing security	24
4.7.1	Login roles	24
4.7.2	Group roles	24
4.7.3	Using PEM predefined roles to manage access to PEM functionality	24
4.7.4	Using a team role	26
4.7.5	Object permissions	26
4.8	Server configuration parameters - reference	26
5	Managing Postgres servers	35
5.1	Starting and stopping monitored database servers	35
5.2	Modifying the pg_hba.conf file	35
5.3	Creating and maintaining databases and objects	37
5.4	Template Linux service script	38
6	Managing a PEM agent	39

6.1	Managing job notifications	39
6.1.1	Configuring job notifications at job level	39
6.1.2	Configuring job notifications at agent level	39
6.1.3	Configuring job notifications at server level	40
6.2	Managing PEM scheduled jobs	40
6.3	Agent privileges	42
6.4	Agent configuration	43
6.5	Agent properties	47
7	Managing certificates	49
7.1	Replacing SSL certificates	49
7.2	Updating agent SSL certificates	52
8	Managing configuration settings	55

Welcome to Postgres Enterprise Manager (PEM) Administration Guide. 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.

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.

1.1 For RHEL and CentOS

1. Edit the file `/etc/httpd/conf.d` file, 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](#) .

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](#) .

REGISTERING A POSTGRES SERVER

Before you can manage or monitor a server with PEM, you must register the server with PEM and bind an agent. You can bind a server to a remote agent (an agent that resides on a different host). However, if the agent doesn't reside on the same host, it doesn't have access to all of the statistical information about the instance.

2.1 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.

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 on 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 **Connectnow?** to attempt a server connection when you select **Save**. Clear **Connectnow?** 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 **Connectiontab** to specify connection details for the server:

- Specify the IP address of the server host or the fully qualified domain name in the **Hostname/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 **Maintenancedatabase** 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 **KerberosAuthentication** 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 **Savepassword?** 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>ClearSavedPassword** .
- 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 **SSLmode** 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 **Clientcertificate** 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 **Clientcertificatekey** 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 **Rootcertificate** 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 **Certificaterevocationlist** 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 **SSLcompression?** 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 **SSHTunnel** 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 **UseSSHTunneling** 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 **Tunnelport** 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 **Identityfile** to use a private key file when connecting.
- If the SSH host is expecting a private key file for authentication, use the **Identityfile** 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 **HostAddress** field.
- Use the **DBrestriction** 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 **Passwordfile** 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 **ServiceID** 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 **EFMclustername** 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 **EFMinstallationpath** 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 **PEMagent** tab to specify connection details for the PEM agent.

On the **ConnectionParameters** tab:

- Set **Remotemonitoring?** 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 **Boundagent** 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 **Username** 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 [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 **Username** 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 **Allowtakeover?** 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 **ConnectionParameters** tab of the **PEMAgent** 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 **BARTserver** field to select the BART server name. All the BART servers configured in the PEM console are listed.
- Use the **Servername** 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 **Backupname** 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 **Hostaddress** 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 **ClusterOwner** 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 **Archivecommand** 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 **Allowincrementalbackup?** to specify whether to enable incremental backup for this database server.
- Use **SetuppasswordlessSSH?** 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 **Overridedefaultconfiguration?** 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 **Retentionpolicy** 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 **WALcompression** 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 **WALcompression** setting for database servers for which you need to take incremental backups.
- Use the **CopyWALsduringrestore** 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 **Threadcount** field to specify the number of threads to copy the blocks. You must set **Threadcount** to **1** if you want to take a backup using the pg_basebackup utility.
- Use the **Batchsize** field to specify the number of blocks of memory used for copying modified blocks. This option applies only to incremental backups.
- Use the **Scaninterval** field to specify the number of seconds for the WAL scanner to wait before scanning the new WAL files.
- Use the **MBMscantimeout** 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.

2.2 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>ManageProbes**. When the **ManageProbes** 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>AutoDiscovery**.

When the Auto Discovery dialog box opens, the **DiscoveredDatabaseServers** 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 **ServerConnectionDetails** box and connection properties for the agent in the **AgentConnectionDetails** box.

Use **CheckAll** to select the box next to all of the displayed servers or **UncheckAll** to clear all of the boxes to the left of the server names.

The fields in the **ServerConnectionDetails** 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 **Servergroup** list to select the server group under which the server is displayed in the PEM client tree.
- Use the **Hostname/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 **ServiceID** field. You must provide the service name to enable some PEM functionality.
- By default, the **Maintenancedatabase** field indicates that the selected server uses a Postgres maintenance database. Customize the content of the **Maintenancedatabase** field for your installation.

The fields in the **AgentConnectionDetails** 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 **Username** 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 **SSLmode** 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.

2.3 Using the pemworker utility to register a server

You can use the pemworker utility to register a server for monitoring by the PEM server or to unregister a database 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
```

On a Windows host, use the command:

```
pemworker.exe REGISTER-SERVICE
```

Append command line options to the command string when invoking the pemworker utility. Follow each option with a corresponding value.

Option	Description
<i>-pem-user</i>	Specifies the name of the PEM administrative user. Required.
<i>-server-addr</i>	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 alternate path containing a PostgreSQL socket. If you enter a path, the path must begin with a forward slash (/). Required.
<i>-server-port</i>	Specifies the port number of the host. Required.
<i>-server-database</i>	Specifies the name of the database to which the server connects. Required.
<i>-server-user</i>	Specifies the name of the user used by the agent when monitoring the server. Required.
<i>-server-service-name</i>	Specifies the name of the database service that controls operations on the server that's being registered (STOP, START, RESTART, and so on).Optional.
<i>-remote-monitoring</i>	Include the <i>-remote-monitoring</i> clause and a value of <i>no</i> (the default) to indicate that the server is installed on the same machine as the PEM agent. When remote monitoring is enabled (<i>yes</i>), 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). Required.

continues on next page

Table 2 – continued from previous page

Option	Description
<code>-efm-cluster-name</code>	Specifies the name of the Failover Manager cluster that monitors the server (if applicable). Optional.
<code>-efm-install-path</code>	Specifies the complete path to the installation directory of Failover Manager, if applicable. Optional.
<code>-asb-host-name</code>	Specifies the name of the host to which the agent is connecting.
<code>-asb-host-port</code>	Specifies the port number for the agent to use when connecting to the database.
<code>-asb-host-db</code>	Specifies the name of the database to which the agent connects.
<code>-asb-host-user</code>	Specifies the database user name that the agent supplies when authenticating with the database.
<code>-asb-ssl-mode</code>	Specifies the type of SSL authentication to use for connections. Supported values include: prefer , require , disable , verify-CA , verify-full .
<code>-group</code>	Specifies the name of the group in which the server is displayed.
<code>-team</code>	Specifies the name of the group role that's allowed to access the server.
<code>-owner</code>	Specifies the name of the role that owns the monitored server.
<code>-display-name <name></code>	Specifies the display name of the monitored database server. The default is the system hostname.
<code>-config-dir</code>	Specifies the directory path of the agent configuration file. The default is " <code><pem-worker path>/../etc</code> ".

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 pemagent.

If you don't provide the password, a password authentication error occurs. The PEM server acknowledges that the server was registered properly.

2.3.1 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-SERVICE
```

Append command line options to the command string when invoking the pemworker utility. Follow each option with by a corresponding value:

Option	Description
<code>-pem-user</code>	Specifies the name of the PEM administrative user. Required.
<code>-server-addr</code>	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 alternate path containing a PostgreSQL socket. If you enter a path, the path must begin with a forward slash (/). Required.
<code>-server-port</code>	Specifies the port number of the host. Required.

continues on next page

Table 3 – continued from previous page

Option	Description
<code>-config-dir</code>	Specifies the directory path of the agent configuration file. The default is “<pem-worker path>/../etc”.

Use the `PEM_SERVER_PASSWORD` environment variable to provide the password for the PEM server to allow the pem-worker to connect as a PEM admin user.

If you don't provide the password, a password authentication error occurs. The PEM server acknowledges that the server is unregistered.

2.4 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 **PostgresServerStatus** 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>ProbeLogAnalysis** from the context menu.

2.5 Registering Postgres clusters on AWS

There are two scenarios in which you can monitor a Postgres cluster on an AWS host using PEM. You can monitor a:

- Postgres cluster running on AWS EC2
- Postgres cluster running on AWS RDS

2.5.1 Monitoring a Postgres cluster running on AWS EC2

After creating a Postgres cluster on AWS EC2, you can use the PEM server to register and monitor your cluster. The following scenarios are currently supported:

- Postgres cluster and PEM agent running on the same AWS EC2 and a PEM server running on your local machine
- Postgres cluster and PEM agent running on the same local machine and a PEM Server running on AWS EC2
- Postgres cluster and PEM agent running on the same AWS EC2 and a PEM Server running in different AWS EC2

Note: In the first two scenarios, you must configure the VPN on AWS EC2 so the AWS EC2 instance can access the *pem* database. Contact your network administrator to set up the VPN if needed.

Since the PEM agent is on a different host from the PEM server, register the PEM agent to the PEM server first. Also, make sure to use the AWS EC2 instance hostname while registering the PEM agent to the PEM server. For more details on registering the PEM agent see, [Registering an agent](#).

You can monitor your Postgres cluster using PEM after you:

- Register the PEM agent with the PEM server
- Bind the PEM agent to the Postgres cluster while adding to the PEM server

2.5.2 Monitoring a Postgres cluster running on AWS RDS

While creating an AWS RDS database, select **PostgreSQL** when prompted for engine options. See Remote monitoring for next steps.

REGISTERING A PEM AGENT

You must register each PEM agent installed on a separate host with the PEM server. (The PEM agent is different from the PEM server host.) The registration process provides the PEM server with the information it needs to communicate with the agent. The PEM agent graphical installer for Windows supports self-registration for the agent. On a Linux host, you must use the `pemworker` utility to register the agent.

The PEM agent package installer places the PEM agent in the `/usr/edb/pem/agent/bin` directory. To register an agent, include the `--register-agent` keywords along with registration details when invoking the `pemworker` utility:

```
pemworker --register-agent
```

Note:

- The PEM agent installed on the PEM server host is registered with the PEM server by default. - After upgrading the PEM agent, you need to restart it. It doesn't require registration.

Append command line options to the command string when invoking the `pemworker` utility. Follow each option with a corresponding value.

Option	Description
<code>-pem-server</code>	The IP address of the PEM backend database server. This parameter is required.
<code>-pem-port</code>	The port of the PEM backend database server. The default value is 5432.
<code>-pem-user</code>	The name of the database user having superuser privileges of the PEM backend database server. This parameter is required.
<code>-pem-agent-user</code>	The agent user to connect the PEM server backend database server.
<code>-cert-path</code>	The complete path to the directory where certificates are created. If you don't provide a path, certificates are created in <code>~/.pem</code> on Linux and <code>%APPDATA%/pem</code> on Windows
<code>-config-dir</code>	The directory path for the configuration file. The default is <code><pemworker path>/../etc</code> .
<code>-display-name</code>	A user-friendly name for the agent to display in the PEM browser tree. The default is the system hostname.
<code>-force-registration</code>	Include the <code>force_registration</code> 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 <i>Yes</i> .
<code>-group</code>	The name of the group in which the agent is displayed.
<code>-team</code>	The name of the database role on the PEM backend database server with access to the monitored database server.
<code>-owner</code>	The name of the database user on the PEM backend database server who owns the agent.
<code>-allow_server_restart</code>	Enable the <code>allow-server_restart</code> parameter to allow PEM to restart the monitored server. The default value is <i>True</i> .

continues on next page

Table 1 – continued from previous page

Option	Description
<i>--allow-batch-probes</i>	Enable the <i>allow-batch-probes</i> parameter to allow PEM to run batch probes on this agent. The default value is <i>False</i> .
<i>--batch-script-user</i>	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.
<i>--enable-heartbeat-connection</i>	Enable the <i>enable-heartbeat-connection</i> parameter to create a dedicated heartbeat connection between the PEM agent and server to update the active status. The default value is <i>False</i> .
<i>--enable-smtp</i>	Enable the <i>enable-smtp</i> parameter to allow the PEM agent to send the email on behalf of the PEM server. The default value is <i>False</i> .
<i>--enable-snmp</i>	Enable the <i>enable-snmp</i> parameter to allow the PEM agent to send the SNMP traps on behalf of the PEM server. The default value is <i>False</i> .
<i>-o</i>	Specifies whether you want to override the configuration file options.

If you want to use any PEM feature for which a database server restart is required by the pemagent (such as Audit Manager, Log Manager, or the Tuning Wizard), then you must set the value for `allow_server_restart` to `true` in the `agent.cfg` file.

Note: When configuring a shell/batch script run by a PEM agent that has PEM version 7.11 or later installed, you must specify the user for the *batch_script_user* parameter. We strongly recommend that you use a nonroot user to run the scripts. Using the root user might result in compromising the data security and operating system security. However, if you want to restore the pemagent to its original settings using root user to run the scripts, then you must set the *batch_script_user* parameter to *root*.

Before any changes are made on the PEM database, the connecting agent 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_MONITORED_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. You are prompted for any other missing required information. When the registration is complete, the server confirms that the agent was successfully registered.

3.1 Unregistering a PEM agent

You can use the pemworker utility to unregister a PEM agent. To unregister an agent, include the `--unregister-agent` keywords along with the details when invoking the pemworker utility:

```
pemworker --unregister-agent
```

Append command line options to the command string when invoking the pemworker utility. Follow each option with a corresponding value:

Option	Description
<code>-pem-user <username></code>	Specifies the name of the database user (member of <code>pem_admin</code> role) of the PEM backend database server. This parameter is required.
<code>-config-dir</code>	Specifies the directory path for the configuration file. The default is “ <code><pemworker path>/../etc</code> ”.

3.2 Setting PEM agent configuration parameters

The PEM agent RPM installer creates a sample configuration file named `agent.cfg.sample` in the `/usr/edb/pem/agent/etc` directory. When you register the PEM agent, the `pemworker` program creates the actual agent configuration file, named `agent.cfg`. You must modify the `agent.cfg` file, adding the following configuration parameter:

```
heartbeat_connection = true
```

You must also add the location of the `ca-bundle.crt` certificate authority file. By default, the installer creates a `ca-bundle.crt` file in the location specified in your `agent.cfg.sample` file. You can copy the default parameter value from the sample file or, if you use a `ca-bundle.crt` file that's stored in a different location, specify that value in the `ca_file` parameter:

```
ca_file=/usr/libexec/libcurl-pem7/share/certs/ca-bundle.crt
```

Then, use a platform-specific command to start the PEM agent service. The service is named `pemagent`.

On a RHEL or CentOS 7.x or 8.x host, use `systemctl` to start the service:

```
systemctl start pemagent
```

The service confirms that it's starting the agent. When the agent is registered and started, it appears on the Global Overview dashboard and in the Object browser tree of the PEM web interface.

For information about using the `pemworker` utility to register a server, see [Manually registering a database server](#).

3.3 Using a nonroot user account to register a PEM agent

To use a nonroot 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 nonroot user, such as `edb`. Then:

1. Log in as `edb`. Create `pem` and `logs` directories and assign read, write, and execute permissions:

```
$ 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
```

(continues on next page)

(continued from previous page)

```
# 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:

```
$ 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.

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

4.1 Starting and stopping the PEM server and agents

The PEM server starts, stops, and restarts when the Postgres server instance on 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* .

4.2 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 or CentOS version 7.x or 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.

4.3 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 **Stoptheservice** to stop a service. Any user or client application connected to the server is abruptly disconnected if you stop the service.

Select **Pausetheservice** to reload a service's configuration parameters. **Pausetheservice** is an effective way to reset parameters without disrupting user sessions for many of the configuration parameters.

Select **Starttheservice** to start a service.

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 CentOS or RHEL 7.x or 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.

4.5 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](#).

4.6 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>ServerConfiguration**. Then edit the parameter values as needed.

4.7 Managing security

PEM provides an interface for managing your Postgres roles and servers.

4.7.1 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 it is strongly recommended (for both security and auditing purposes) that individual roles are created for each connecting user. You can use the PEM Query tool, the PEM web interface **Create – Login/Group Role** dialog, 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/GroupRoles** node.
3. From the context menu, select **Create>Login/GroupRole**.
4. Use the tabs of the **Create – Login/Group Role** dialog box to define the role.
5. After you finish defining the new role, select **Save** to create the role.

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](#).

4.7.2 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 **GroupRoles** 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](#).

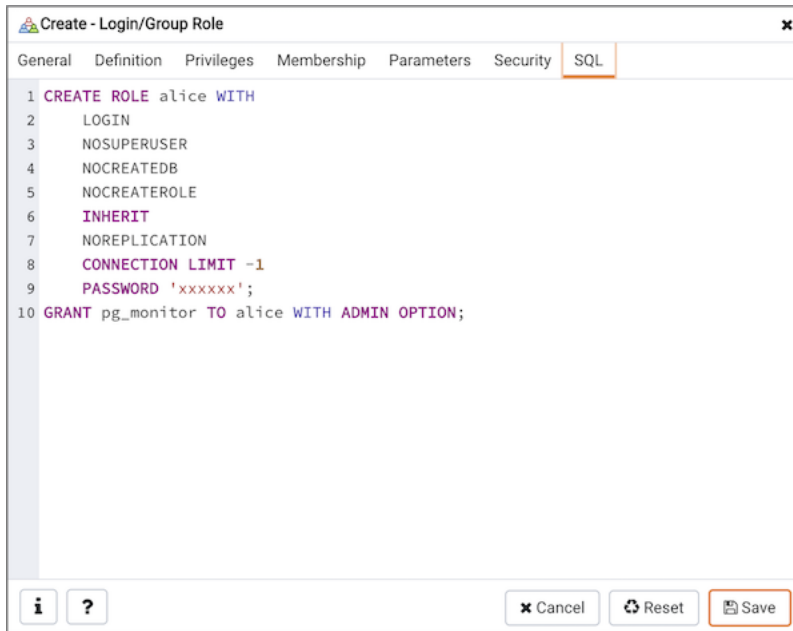
4.7.3 Using PEM predefined roles to manage access to PEM functionality

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 is 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 the Audit Manager. The role can make unlimited connections to the server at any given time.



The screenshot shows a window titled "Create - Login/Group Role" with a close button (X) in the top right corner. The window has several tabs: General, Definition, Privileges, Membership, Parameters, Security, and SQL. The "SQL" tab is currently selected and highlighted with an orange border. Inside the SQL tab, there is a text area containing the following SQL script:

```
1 CREATE ROLE alice WITH
2     LOGIN
3     NOSUPERUSER
4     NOCREATEDB
5     NOCREATEROLE
6     INHERIT
7     NOREPLICATION
8     CONNECTION LIMIT -1
9     PASSWORD 'xxxxxx';
10 GRANT pg_monitor TO alice WITH ADMIN OPTION;
```

At the bottom of the window, there are three buttons: "Cancel", "Reset", and "Save". The "Save" button is highlighted with an orange border.

You can use PEM predefined roles to allow access to the functionality listed in the table.

Value	Parent role	Description
<code>pem_super_admin</code>		Role to manage/configure everything on PEM
<code>pem_admin</code>	<code>pem_super_admin</code>	Role for administration/management/configuration of all visible agents/servers and monitored objects
<code>pem_config</code>	<code>pem_admin</code>	Role for configuration management of PEM
<code>pem_component</code>	<code>pem_admin</code>	Role to run/execute all wizard/dialog box-based components
<code>pem_rest_api</code>	<code>pem_admin</code>	Role to access the REST API
<code>pem_server_service_management</code>	<code>pem_admin</code>	Role for allowing to restart/reload the monitored database server (if server-id provided)
<code>pem_manage_schedule_tasks</code>	<code>pem_admin</code>	Role to configure the schedule tasks
<code>pem_manage_alert</code>	<code>pem_admin</code>	Role for managing/configuring alerts and its templates
<code>pem_config_alert</code>	<code>pem_config</code> , <code>pem_manage_alert</code>	Role for configuring the alerts on any monitored objects
<code>pem_manage_probe</code>	<code>pem_admin</code>	Role to create, update, and delete the custom probes and change custom probe configuration
<code>pem_config_probe</code>	<code>pem_config</code> , <code>pem_manage_probe</code>	Role for probe configuration (history retention, execution frequency, enable/disable the probe) on all visible monitored objects
<code>pem_database_server_registration</code>	<code>pem_admin</code>	Role to register a database server
<code>pem_comp_postgres_expert</code>	<code>pem_component</code>	Role to run the Postgres Expert
<code>pem_comp_auto_discovery</code>	<code>pem_component</code>	Role to run the auto discovery of a database server dialog box
<code>pem_comp_log_analysis_expert</code>	<code>pem_component</code>	Role to run the Log Analysis Expert
<code>pem_comp_sqlprofiler</code>	<code>pem_component</code>	Role to run the SQL Profiler
<code>pem_manage_efm</code>	<code>pem_admin</code>	Role to manage Failover Manager
<code>pem_comp_capacity_management</code>	<code>pem_component</code>	Role to run the Capacity Manager

continues on next page

Table 1 – continued from previous page

Value	Parent role	Description
pem_comp_log_manager	pem_component	Role to run the Log Manager
pem_comp_audit_manager	pem_component	Role to run the Audit Manager
pem_comp_tuning_wizard	pem_component	Role to run the Tuning Wizard

4.7.4 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/GroupRoles** node.
3. From the context menu, select **Create>Login/GroupRole**.
4. In the Create - Login/Group Role dialog box, use the fields to specify the properties of the team role.

4.7.5 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.

4.8 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.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
chart_disable_bullets	false	Enables/disables bullets on line charts on dashboards and Capacity Manager reports.
cm_data_points_per_report	50	Specifies the number of data points to plot on charts on Capacity Manager reports.
cm_max_end_date_in_years	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_timeout	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_timeout	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_timeout	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.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
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.
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_timeout	300 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_rows	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_timeout	300 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_hostmemact_span	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_hostmemact_timeout	300 seconds	Specifies the number of seconds after which the Host Memory Activity line chart is refreshed on the Memory Analysis dashboard.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
dash_memory_hostmemconf	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.
dash_memory_servmemact	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_servmemact	1800 seconds	Specifies the number of seconds after which the Server Memory Activity line chart is refreshed on the Memory Analysis dashboard.
dash_memory_servmemconf	300 seconds	Specifies the number of seconds after which the Server Memory Configuration pie chart is refreshed on the Memory Analysis dashboard.
dash_objectact_objstorage	15 rows	Specifies the number of rows to show on the Object Storage table on the Object Activity Analysis dashboard.
dash_objectact_objstorage	300 seconds	Specifies the number of seconds after which the Object Storage table is refreshed on the Object Activity Analysis dashboard.
dash_objectact_objtopindex	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_objtoptable	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_timeout	1800 seconds	Specifies the number of seconds after which the Memory line chart is refreshed on the Operating System Analysis dashboard.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
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.
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_archive_status_span	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_archive_status_timeout	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_span	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_timeout	1800 seconds	Specifies the number of seconds after which the WAL Lag Pages line chart is refreshed on the Streaming Replication dashboard.
dash_replication_segmentlag_span	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_segmentlag_timeout	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_span	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_timeout	1800 seconds	Specifies the number of seconds after which the Replication Lag Time line chart is refreshed on the Streaming Replication dashboard.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
dash_server_buffers_written	168 hours	Specifies the number of days worth of data to plot on the Background Writer Statistics chart on the Server Analysis dashboard.
dash_server_buffers_written	300 seconds	Specifies the number of seconds after which the Background Writer Statistics line chart is refreshed on the Server Analysis dashboard.
dash_server_connoverview	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is refreshed on the Server Analysis dashboard.
dash_server_database_time	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_timeout	1800 seconds	Specifies the number of seconds after which the Database Size line chart is refreshed on the Server Analysis dashboard.
dash_server_disk_timeout	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_span	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_time	1800 seconds	Specifies the number of seconds after which the Shared Buffers line chart is refreshed on the Server Analysis dashboard.
dash_server_tabspacesize_span	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_time	1800 seconds	Specifies the number of seconds after which the Tablespace Size line chart is refreshed on the Server Analysis dashboard.
dash_server_useract_span	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_timeout	1800 seconds	Specifies the number of seconds after which the User Activity line chart is refreshed on the Server Analysis dashboard.
dash_sessact_lockact_time	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_time	300 seconds	Specifies the number of seconds after which the Session Workload table is refreshed on the Session Activity Analysis dashboard.
dash_sess_waits_timewait	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.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
dash_sess_waits_waitdtl_time	300 seconds	Specifies the number of seconds after which the Session Waits Details table is refreshed on the Session Waits Analysis dashboard.
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_dbovervw_time	300 seconds	Specifies the number of seconds after which the Database Overview pie chart is refreshed on the Storage Analysis dashboard.
dash_storage_hostdtls_time	300 seconds	Specifies the number of seconds after which the Host Details table is refreshed.
dash_storage_hostovervw_time	300 seconds	Specifies the number of seconds after which the Host Overview pie chart is refreshed on the Storage Analysis dashboard.
dash_storage_tblspcdtls_time	300 seconds	Specifies the number of seconds after which the Tablespace Details table is refreshed on the Storage Analysis dashboard.
dash_storage_tblspcovervw_time	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_time	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_time	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_time	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_time	7 days	Specifies the number of days that a custom chart (displayed on a user-defined dashboard) is stored.
deleted_probes_retention_time	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_change	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 $[N + 1 * (\min(\text{probe_interval}) * 2)]$ minutes and the fine state is not None. The default value of N is 2 or 3, and $\min(\text{probe_interval})$ is the smallest interval for all the probes used by the alert. Flapping ends when ZERO state changes have occurred over $[2 N * \min(\text{probe_interval})]$ minutes.
job_retention_time	30 days	Specifies the number of days that nonrecurring scheduled tasks and their associated jobs are retained.
long_running_transaction_time	5 minutes	Specifies the number of minutes a query executes before being considered long running.
nagios_cmd_file_name	<file_name>	Specifies nagios command file to which passive service check results are sent.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
nagios_enabled	t	Specifies whether alert notification are submitted to nagios.
na-gios_medium_alert_as_critical	f	Specifies whether a medium level PEM alert is considered critical in nagios.
na-gios_spool_retention_time	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.
re-minder_notification_interval	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 <i>true</i> , 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_time	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_community	public	Specifies the SNMP community used when sending traps. Used only with SNMPv1 and SNMPv2.
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.
snmp_server	127.0.0.1	Specifies the SNMP server host address to use for sending SNMP traps.
snmp_spool_retention_time	7 days	Specifies the number of days to retain sent traps in the spool table before they are discarded.
snmp_security_name		Specifies the user name or security name for sending SNMP traps. 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 or AUTH_PRIV - Authentication, Privacy or NOAUTH_NOPRIV - no Authentication, no Privacy. Used only with SNMPv3.

continues on next page

Table 2 – continued from previous page

Parameter name	Value and unit	Description
snmp_context_name		Specifies the context name, the identifier for MIB objects when sending SNMP traps. Used only with SNMPv3.
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_authentication_protocol	NONE	Specifies the authentication type for SNMP traps. Its possible values are NONE, HMACMD5, and HMAC-SHA. Used only with 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_authentication_password		Specifies the authentication password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
snmp_privacy_password		Specifies the privacy password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
webclient_help_pg	EDB hosted documentation	Specifies the location of the online PostgreSQL core documentation.

MANAGING POSTGRES SERVERS

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 **StoreonPEMServer** option on the Properties dialog box.
- Specify the name of a service script in the **ServiceID** 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>QueueServerStartup** or **Tools>QueueServerShutdown** .

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.

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:

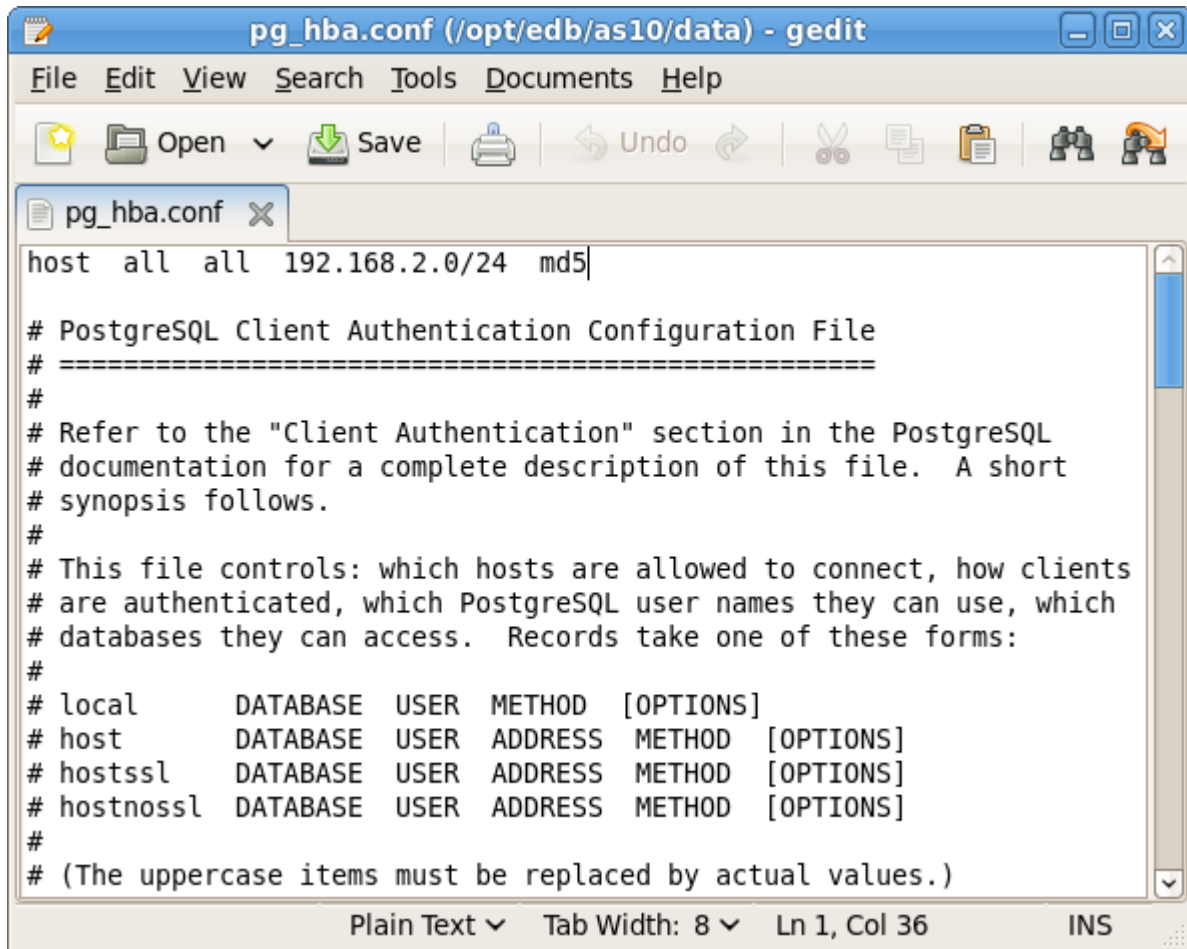


Fig. 1: PEM entries in the pg_hba.conf file

#	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.

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.

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 CentOS or RHEL 7.x | 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](#) .

MANAGING A PEM AGENT

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)

6.1.1 Configuring job notifications at job level

You can configure email notification settings at 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 on job level:

- Use the **Sendthenotifications** field to specify when you want to send the email notifications.
- Use the **Emailgroup** field to specify the email group to send the email notification to.

6.1.2 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 **JobNotifications** tab.

Use the Job notifications tab to configure the email notification settings at agent level:

- Use the **Overridedefaultconfiguration?** 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 **Emailonjobcompletion?** switch to specify whether to send the job notification when the job completes successfully.
- Use the **Emailonajobfailure?** switch to specify whether to send the job notification when the job fails.
- Use the **Emailgroup** field to specify the email group to send the job notification to.

6.1.3 Configuring job notifications at server level

You can use the Server Configuration dialog box to provide information about your email notification configuration at PEM server level. To open the Server Configuration dialog box, select **Management>ServerConfiguration**.

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.

6.2 Managing PEM scheduled jobs

You can create a PEM scheduled job to perform a set of steps you define in the 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 **CreateJob** 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.

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.

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 **Onerror** 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 ****SQLquery**** 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 `'AllowBatchJobSteps'` to `'true'` and restart the PEM agent. PEM registry entries are located in `'HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent'`.

After providing all the information required by the step, select **Save**.

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 **Schedulesdefinition** 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 **WeekDays** field to select the days when the job executes. - Use the **MonthDays** field to select the numeric days when the job executes. Select **LastDay** 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.

Select the **Notifications** tab to configure the email notification settings on job level:

- Use the **Sendthenotifications** field to specify when you want to send the email notifications.
- Use the **Emailgroup** 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.

6.3 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.

PEM functionality lessens as the privileges of the PEM agent decrease. For complete functionality, run the PEM agent as root. If the PEM agent is run 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 is run under another lesser-privileged account, functionality is limited even further.

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.

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.

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.

6.4 Agent configuration

A number of configurable parameters and registry entries control the behavior of the PEM agent. You might need to modify the PEM agent's parameter settings to enable some PEM functionality. After modifying values in the PEM agent configuration file, restart the PEM agent to apply any changes.

With the exception of the `PEM_MAXCONN` parameter, we strongly recommend against modifying any of these configuration parameters or registry entries without first consulting EDB support experts unless you need the modifications to enable PEM functionality.

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 following entries.

Parameter name	Description	Default value
<code>pem_host</code>	The IP address or hostname of the PEM server.	127.0.0.1.
<code>pem_port</code>	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.

continues on next page

Table 3 – continued from previous page

Parameter name	Description	Default value
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_cert	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	Log level specifies the type of event to write to the PEM log files.	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
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.	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 for multiple PEM Agents (7.13 or earlier) it might send more duplicate emails. Whereas for PEM Agents (7.14 or higher) it might send fewer duplicate emails.	true for PEM server host, false for all others.
enable_snmp	When set to true for multiple PEM Agents (7.13 or earlier) it might send more duplicate traps. Whereas for PEM Agents (7.14 or higher) it might send fewer duplicate traps.	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	Set maximum number of times pemAgent retries to call webhooks on failure.	Default 3.
connect_timeout	The max 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	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).

continues on next page

Table 3 – continued from previous page

Parameter name	Description	Default value
allow_batch_probes	If set to TRUE, the user can't create batch probes using the custom probes feature.	false
heartbeat_connection	When set to TRUE, a dedicated connection is used for sending the heartbeats.	false
batch_script_dir	Provide the path where script file (for alerting) is stored.	/tmp
connection_custom_setup	Use to provide SQL code to invoke when a new connection with a monitored server is made.	Not set by default.
ca_file	Provide the path where the CA certificate resides.	Not set by default.
batch_script_user	Provide the name of the user to use for executing the batch/shell scripts.	None
webhook_ssl_key	The complete path to the webhook's SSL client key file.	
webhook_ssl_cert	The complete path to the webhook's SSL client certificate file.	
webhook_ssl_crl	The complete path of the CRL file to validate webhook server certificate.	
webhook_ssl_ca_cert	The complete path to the webhook's SSL ca certificate file.	
allow_insecure_webhooks	When set to true, allow webhooks to call with insecure flag.	false

On 64-bit Windows systems, PEM registry entries are located in:

HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent

The registry contains the entries shown in the 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.
AgentCertPath	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.	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
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

continues on next page

Table 4 – continued from previous page

Parameter name	Description	Default value
AlertThreads	The number of alert threads for the agent to spawn.	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	Set maximum number of times for pemAgent to retry to call webhooks on failure.	Default 3.
ConnectTimeout	The max 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't 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	Provide the path to store the script file for alerting.	/tmp
ConnectionCustom-Setup	Use to provide SQL code to invoke when a new connection with a monitored server is made.	Not set by default.
ca_file	Provide the path where the CA certificate resides.	Not set by default.
AllowBatchJobSteps	If set to true, the batch/shell scripts are executed using Administrator user account.	None
WebhookSSLKey	The complete path to the webhook's SSL client key file.	
WebhookSSLCrt	The complete path to the webhook's SSL client certificate file.	
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

6.5 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 **Heartbeatinterval** 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 **JobNotifications** tab to configure the email notification settings on agent level:

```
- Use the **Overridedefaultconfiguration?** 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 **Emailonjobcompletion?**
↪switch to specify whether to send the job notification when the job completes
↪successfully. - Use the **Emailonajobfailure?** switch to specify whether to send
↪the job notification when the job fails. - Use the **Emailgroup** field to specify
↪the email group to send the job notification to.
```

The **AgentConfigurations** 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**.
```


MANAGING CERTIFICATES

Files stored in the data directory of the PEM server backing database contain information that helps the PEM server use secure connections:

- `ca_certificate.crt`
- `ca_key.key`
- `server.crt`
- `server.key`
- `root.crl`
- `root.crt`

The PEM agent that's 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, appending the new CA certificate file to the `root.crt` file on the PEM server.
- Creates a job that renews the certificate file of any active agents.
- Restarts the PEM server.

When you uninstall an agent, the certificate associated with that agent is added to the certificate revocation list maintained in the `root.crl` file to ensure that you can't use the certificate to connect to the PEM server.

You can manually replace certificate files.

7.1 Replacing SSL certificates

You can replace the SSL certificates on an existing PEM installation. If you plan to upgrade your server to a new version at the same time, invoke all of the PEM installers (first the server installer, then agent installers) before replacing the SSL certificates. Then:

1. Stop all running PEM agents, first on the server host, and then on any monitored node.

To stop a PEM agent on a Linux host, open a terminal window, assume superuser privileges, and enter the command:

On Linux with systemd, for example, Centos 7 or 8

```
systemctl stop pemagent
```

On a Windows host, you can use the Services applet to stop the PEM agent. The PEM agent service is named Postgres Enterprise Manager Agent. Select the service name in the Services dialog box and select **Stoptheservice**.

2. Take a backup of the existing SSL keys and certificates. The SSL keys and certificates are stored in the data directory under your PEM installation. For example, the default location on a Linux system is:

/var/lib/pgsql/x/data , where x is the PostgreSQL database version

Copy the following files, adding an extension to each file to make the name unique:

```
- `ca_certificate.crt` - `ca_key.key` - `root.crt` - `root.crl` - `server.  
↪key` - `server.crt`
```

For example, to create a backup of the ca_certificate file with the word old appended to the entry, use this command:

```
# cp ca_certificate.crt ca_certificate_old.crt
```

3. Use the openssl_rsa_generate_key() function to generate the ca_key.key file:

```
/usr/pgsql-x.x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT public.openssl_  
↪rsa_generate_key(1024)" > /var/lib/pgsql/x/data/ca_key.key
```

After creating the ca_key.key file, cat the contents to the variable CA_KEY for use when generating the ca_certificate.crt file. Modify the privileges on the ca_key.key file:

```
CA_KEY=$(cat /var/lib/pgsql/x/data/ca_key.key)  
  
chmod 600 /var/lib/pgsql/x/data/ca_key.key
```

4. Use the key to generate the ca_certificate.crt file. For simplicity, place the SQL query in a temporary file with a unique name:

```
echo "SELECT openssl_csr_to_cert(openssl_rsa_key_to_csr({CA_KEY},  
PEM,US, MA, Bedford, Postgres Enterprise Manager,  
support@enterprisedb.com), NULL,  
/var/lib/pgsql/x/data/ca_key.key)" > /tmp/_random.$$
```

Then use the variable to execute the query, placing the content in the ca_certificate.crt file.

```
/usr/pgsql-x.x/bin/psql -U postgres -d pem --no-psqlrc -t -A -f /tmp/_random.$$ > /var/lib/  
↪pgsql/x/data/ca_certificate.crt
```

Modify the permissions of the ca_certificate.crt file, and remove the temporary file that contained the SQL command:

```
chmod 600 /var/lib/pgsql/x/data/ca_certificate.crt  
  
rm -f /tmp/_random.$$
```

5. Reuse the ca_certificate.crt file as the root.crt file:

```
cp /var/lib/pgsql/x/data/ca_certificate.crt /var/lib/pgsql/x/data/root.crt
```

Modify the permissions of the root.crt file:

```
chmod 600 /var/lib/pgsql/x/data/root.crt
```

6. Use the openssl_rsa_generate_crl() function to create the certificate revocation list (root.crl):

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c
"SELECT openssl_rsa_generate_crl(/var/lib/pgsql/x/data/ca_certificate.crt, /var/lib/
↳pgsql/x/data/ca_key.key)" > /var/lib/pgsql/x/data/root.crl
```

Modify the permissions of the `root.crl` file:

```
chmod 600 /var/lib/pgsql/x/data/root.crl
```

7. Use the `openssl_rsa_generate_key()` function to generate the `server.key` file:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT public.openssl_rsa_
↳generate_key(1024)" >> /var/lib/pgsql/x/data/server.key
```

After creating the `server.key` file, cat the contents to the variable `SSL_KEY` for use when generating the `server.crt` file and modify the privileges on the `server.key` file:

```
SSL_KEY=$(cat /var/lib/pgsql/x/data/server.key)

chmod 600 /var/lib/pgsql/x/data/server.key
```

8. Use the `SSL_KEY` to generate the server certificate. Save the certificate in the `server.crt` file. For simplicity, first place the SQL query into a temporary file with a unique name:

```
echo "SELECT openssl_csr_to_crt(openssl_rsa_key_to_csr(${SSL_KEY},
PEM,US, MA, Bedford, Postgres Enterprise Manager,
support@enterprisedb.com),
/var/lib/pgsql/x/data/ca_certificate.crt,
/var/lib/pgsql/x/data/ca_key.key)" > /tmp/_random.$$

/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -f /tmp/_random.$$ >> /var/
↳lib/pgsql/x/data/server.crt
```

9. Modify the privileges on the `server.crt` file, and delete the temporary file:

```
chmod 600 /var/lib/pgsql/x/data/server.crt

rm -f /tmp/_random.$$
```

10. Restart the Postgres server:

On Linux with `init.d`, for example, on a Centos6 host:

```
/etc/init.d/postgresql-x restart
```

On Linux with `systemd`, for example, on a Centos7 host:

```
systemctl restart postgresql-x
```

7.2 Updating agent SSL certificates

For each agent that interacts with the PEM server, you must:

- Generate an rsa key and a certificate.
- Copy the key and certificate to the agent.
- Restart the agent.

Each agent has a unique identifier that's stored in the `pem.agent` table in the `pem` database. You must replace the key and certificate files with the key or certificate that corresponds to the agent's identifier. You must move the `agent.key` and `agent.crt` files generated in Steps 2 and 3 into place on their respective PEM agent host before generating the next key file pair. Subsequent commands overwrite the previously generated file.

To generate a PEM agent key file pair:

1. Use `psql` to find the number of agents and their corresponding identifiers:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT ID FROM pem.agent"
```

- On Linux, you can also find the agent identifier and location of the keys and certificates in the `PEMAgent` section of the `/etc/postgres-reg.ini` file.
- On Windows, the information is stored in the registry:

– On a 64-bit Windows installation, check:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\EnterpriseDB\PEM\agent
```

– On a 32-bit Windows installation, check:

```
HKEY_LOCAL_MACHINE\SOFTWARE\EnterpriseDB\PEM\agent
```

2. After identifying the agents that need key files, generate an `agent.key` for each agent. To generate the key, execute the following command, capturing the output in a file:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT openssl_rsa_generate_key(1024)" > agent.key
```

Modify the privileges of the `agent.key` file:

```
chmod 600 agent.key
```

3. Generate a certificate for each agent. To generate a certificate, execute the following command, capturing the output in a certificate file:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c
"SELECT openssl_csr_to_cert(openssl_rsa_key_to_csr($(cat agent.key),
agent<$ID>, US, MA, Bedford, Postgres Enterprise Manager,
support@enterprisedb.com),
/var/lib/pgsql/x/data/ca_certificate.crt,
/var/lib/pgsql/x/data/ca_key.key)" > agent.crt
```

Where `$ID` is the agent number of the agent (retrieved by the `psql` command line).

4. Modify the privileges of the `agent.crt` file:

```
chmod 600 agent.crt
```

5. Replace each agent's key and certificate file with the newly generated files before restarting the PEM agent service:
On Linux with `init.d`, restart the service with the command:

```
/etc/init.d/pemagent start
```

On Linux with `systemd`, restart the service with the command:

```
systemctl start pemagent
```

On a Windows host, you can use the Services applet to start the PEM agent. The PEM agent service is named Postgres Enterprise Manager Agent. Select the service name in the Services dialog box and select **Start the service**.

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.