

EDB
Postgres® for the AI Generation

Postgres Enterprise Manager

リリース *9.7.0*

EnterpriseDB Corporation.

2024 年 09 月 17 日

目 次

第 1 章 Introduction **3**
1.1 Postgres の互換性 . . . 3
1.2 Release notes . . . 3
1.2.1 Postgres Enterprise Manager 9.7.0 release notes . . . 4
1.2.2 Postgres Enterprise Manager 9.6.0 release notes . . . 5
1.2.3 Postgres Enterprise Manager 9.5.1 release notes . . . 6
1.2.4 Postgres Enterprise Manager 9.5.0 release notes . . . 7
1.2.5 Postgres Enterprise Manager 9.4.1 release notes . . . 8
1.2.6 Postgres Enterprise Manager 9.4.0 release notes . . . 8
1.2.7 Postgres Enterprise Manager 9.3.1 release notes . . . 8
1.2.8 Postgres Enterprise Manager 9.3.0 release notes . . . 9
1.2.9 Postgres Enterprise Manager 9.2.2 release notes . . . 10
1.2.10 Postgres Enterprise Manager 9.2.1 release notes . . . 10
1.2.11 Postgres Enterprise Manager 9.2.0 release notes . . . 10
1.2.12 Postgres Enterprise Manager 9.1.1 release notes . . . 12
1.2.13 Postgres Enterprise Manager 9.1.0 release notes . . . 12
1.2.14 Postgres Enterprise Manager 9.0.0 release notes . . . 12
1.3 Platform compatibility . . . 13
1.4 Hardware requirements for installing the PEM server on Linux . . . 14

第 2 章 Planning **15**
2.1 Understanding Postgres Enterprise Manager components and architecture . . . 15
2.1.1 PEM アーキテクチャ . . . 16
2.2 Deployment considerations . . . 20
2.2.1 Using Failover Manager for high availability . . . 20
2.2.2 Connection pooling using PgBouncer . . . 31
2.2.3 Authentication options . . . 43
2.2.4 Securing your deployment . . . 57
2.2.5 Web server installation options . . . 68

第 3 章 Installing and Upgrading **69**
3.1 Installing Postgres Enterprise Manager server . . . 69
3.1.1 Linux x86-64 (amd64) . . . 69
3.1.2 Linux Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le) . . . 70
3.1.3 Linux Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64) . . . 70
3.1.4 Windows . . . 70
3.2 Installing Postgres Enterprise Manager agent . . . 125
3.2.1 Linux x86-64 (amd64) . . . 125
3.2.2 Linux Installing Postgres Enterprise Manager server on Linux IBM Power (ppc64le) . . . 126
3.2.3 Linux Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64) . . . 126

3.2.4 Windows . . . 126
3.3 Upgrade and migration . . . 142
3.3.1 Upgrading a PEM installation . . . 143
3.3.2 Upgrading the PEM backend Postgres database . . . 148
3.3.3 Adding entries for PEM Agens and admins to connect to PEM server . . . 157
3.3.4 Adding entries (localhost) for PEM Agens and admins to connect to PEM server . . . 157
3.3.5 Upgrading SQL Profiler . . . 158
3.3.6 Upgrading Apache HTTPD server . . . 158
3.3.7 Moving the PEM server . . . 160
3.4 Uninstalling . . . 169
3.4.1 Uninstalling Postgres Enterprise Manager components on Linux . . . 169
3.4.2 Uninstalling Postgres Enterprise Manager components on Windows . . . 171
3.5 Troubleshooting . . . 171
3.5.1 サーバーインストールエラー . . . 171
3.5.2 PEM サーバーの再構成 . . . 173
3.5.3 PEM Web クライアントがロードされません . . . 173

第 4 章 Administering **177**
4.1 Changing the default port . . . 177
4.1.1 RHEL の場合 . . . 177
4.1.2 Debian および Ubuntu の場合 . . . 178
4.2 Registering a Postgres server . . . 178
4.2.1 Postgres サーバーを登録するとどうなりますか? . . . 178
4.2.2 pemworker ユーティリティを使用してサーバーを登録する . . . 179
4.2.3 自動サーバー検出へのサーバーの登録 . . . 182
4.2.4 データベースサーバーの手動登録 . . . 183
4.2.5 接続とバインディングの検証 . . . 190
4.3 Registering a PEM agent . . . 190
4.3.1 PEM エージェントを登録する方法 . . . 190
4.3.2 pemworker ユーティリティを使用した PEM エージェントの登録 . . . 191
4.3.3 PEM エージェントの登録解除 . . . 193
4.3.4 高度な使用法 . . . 193
4.4 Managing a PEM server . . . 197
4.4.1 PEM サーバーとエージェントの起動と停止 . . . 197
4.4.2 Linux での PEM サーバーまたは PEM エージェントの制御 . . . 198
4.4.3 Windows での PEM サーバーまたは PEM エージェントの制御 . . . 198
4.4.4 HTTPD サーバーの制御 . . . 199
4.4.5 PEM 認証の管理 . . . 199
4.4.6 PEM サーバー構成の編集 . . . 200
4.4.7 セキュリティの管理 . . . 200
4.4.8 サーバー構成 . . . 204
4.4.9 サーバー構成パラメーター - リファレンス . . . 204
4.5 Managing Postgres servers . . . 214
4.5.1 監視対象データベースサーバーの起動と停止 . . . 214
4.5.2 pg_hba.conf ファイルの変更 . . . 214
4.5.3 データベースとオブジェクトの作成と保守 . . . 216
4.5.4 テンプレート Linux サービススクリプト . . . 216

4.6 Managing a PEM agent . . . 217
4.6.1 Managing job notifications . . . 217
4.6.2 Managing scheduled jobs . . . 218
4.6.3 Modifying agent configuration . . . 221
4.6.4 Reviewing or modifying agent properties . . . 227
4.6.5 Setting agent privileges . . . 228
4.7 Managing SSL certificates . . . 232
4.7.1 Web サーバー証明書 . . . 232
4.7.2 PEM バックエンドデータベースサーバーとエージェントの接続証明書 . . . 233
4.7.3 共通エージェント証明書とキーペアを生成する . . . 234
4.7.4 信頼できる CA によって署名された証明書とキーを使用する . . . 236
4.7.5 証明書のテスト . . . 238
4.8 Managing configuration settings . . . 243
4.9 Troubleshooting agent issues . . . 243
4.9.1 削除された PEM エージェントの復元 . . . 244
4.9.2 構成ファイルパスの更新エージェントのステータスが不明と表示されます . . . 244
4.9.3 コマンドラインを使用してダウンまたは不明なステータスの PEM エージェントを削除する . . . 245

第 5 章 Using **247**
5.1 Accessing the web interface . . . 247
5.1.1 UI の詳細についてのアプリ内ヘルプへのアクセス . . . 249
5.1.2 PEM クライアントオブジェクトブラウザ . . . 249
5.1.3 PEM タブ付きブラウザウィンドウ . . . 251
5.1.4 チャート、グラフ、テーブルコントロールを使用する . . . 253
5.1.5 ブラウザツールバー . . . 254
5.1.6 PEM メニューバー . . . 255
5.1.7 チャート、グラフ、テーブルの制御とカスタマイズ . . . 259
5.2 Connecting and disconnecting a database server . . . 260
5.2.1 サーバーからの切断 . . . 260
5.3 Monitoring performance . . . 260
5.3.1 Probes . . . 263
5.3.2 Alerts . . . 284
5.3.3 Notifications . . . 310
5.3.4 Capacity Manager . . . 319
5.3.5 Audit Manager . . . 325
5.3.6 Log Manager . . . 330
5.3.7 Charts . . . 344
5.3.8 Dashboards . . . 348
5.3.9 Remote monitoring . . . 355
5.4 Monitoring Barman . . . 356
5.4.1 Barman をモニタリングするための前提条件 . . . 356
5.4.2 Barman サーバーの構成 . . . 357
5.4.3 PEM ダッシュボードでの Barman サーバーの詳細の表示 . . . 359
5.5 Monitoring EDB Postgres Distributed . . . 360
5.6 Monitoring Failover Manager . . . 360
5.6.1 プライマリノードの置き換え . . . 362

5.6.2 スイッチオーバー EFM クラスター . . . 362
5.7 Monitoring Replication Server . . . 362
5.8 Monitoring event history . . . 363
5.9 Tuning performance . . . 364
5.9.1 Tuning Wizard . . . 364
5.9.2 Postgres Expert . . . 366
5.9.3 Performance Diagnostic . . . 369
5.10 Profiling workloads . . . 372
5.10.1 Installing the SQL Profiler extension . . . 374
5.10.2 Using SQL Profiler . . . 376
5.10.3 Using the Index Advisor . . . 379
5.11 Viewing system reports . . . 380
5.11.1 アラート履歴レポート . . . 380
5.11.2 システム構成レポート . . . 381
5.11.3 コア使用状況レポート . . . 381
5.12 PEM command line interface . . . 382
5.12.1 例 . . . 385
5.13 PEM Rest API . . . 385

第 6 章 Developing **387**
6.1 Using the Query tool . . . 387
6.1.1 クエリツールツールバー . . . 388
6.1.2 SQL エディターパネル . . . 390
6.1.3 [データ出力] パネル . . . 391
6.1.4 Query History パネル . . . 395
6.1.5 接続ステータス . . . 395
6.1.6 マクロ . . . 396
6.1.7 UI の詳細についてのアプリ内ヘルプへのアクセス . . . 396
6.2 Using the Schema Diff tool . . . 396
6.2.1 Schema Diff Object Comparison パネル . . . 397
6.2.2 Schema Diff DDL Comparison パネル . . . 398
6.3 Using the ERD tool . . . 398
6.3.1 ツールバー . . . 399
6.3.2 ファイルオプション . . . 399
6.3.3 エクスポートオプション . . . 400
6.3.4 編集オプション . . . 400
6.3.5 テーブルリレーションシップオプション . . . 400
6.3.6 ユーティリティオプション . . . 401
6.3.7 ズームオプション . . . 401
6.3.8 テーブルダイアログ . . . 401
6.3.9 テーブルノード . . . 401
6.3.10 1 対多のリンクダイアログボックス . . . 402
6.3.11 多対多のリンクダイアログボックス . . . 403
6.3.12 テーブルリンク . . . 403
6.3.13 テーブルノート . . . 404

Postgres Enterprise ManagerPEM へようこそ。 PEM は、 EDB Postgres Advanced Server または PostgreSQL データベースの管理および分析機能を提供するコンポーネントで構成されています。PEM は、オープンソースの pgAdmin 4 プロジェクトに基づいています。

第1章　Introduction

PEM は、包括的なデータベース設計および管理システムです。PEM は、初心者と経験豊富な Postgres ユーザーのニーズを満たすように設計されており、データベースオブジェクトの作成、メンテナンス、および使用を簡素化する強力なグラフィカルインターフェイスを提供します。

1.1 Postgres の互換性

PEM 9.x の Postgres のサポートされているバージョン

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

[^1] sslutils は IBM Power の RHEL 7 では利用できないため、このディストリビューションはバックエンドとして PGE を使用できません。

1.2 Release notes

Postgres Enterprise ManagerPEM ドキュメントでは、マイナーリリースとパッチを含む PEM 9 の最新バージョンについて説明しています。このセクションのリリースノートは、各リリースの新機能に関する情報を提供します。8.0.0 以降のマイナーリリースまたはパッチリリースで導入された新機能の場合、コンテンツ内にどのリリースで機能が導入されたかを示すインジケーターもあります。

Version	Release Date	Upstream Merges	Accessibility Conformance
Postgres Enterprise Manager 9.7.0 release notes	13 Aug 2024	NA	Conformance Report
Postgres Enterprise Manager 9.6.0 release notes	15 May 2024	NA	Conformance Report
Postgres Enterprise Manager 9.5.1 release notes	04 Mar 2024	NA	Conformance Report
Postgres Enterprise Manager 9.5.0 release notes	15 Feb 2024	NA	Conformance Report

次のページに続く

表 2 – 前のページからの続き

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

多くの場合、選択された銘柄のみが上流のマージに含まれます。マージに含まれる特定の問題は、リリースノートのトピックにリストされています。

1.2.1 Postgres Enterprise Manager 9.7.0 release notes

リリース日 2024 年 8 月 13 日

PEM 9.7.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	コア使用レポートに PGD 拡張機能の詳細を追加しました。
Enhancement	`agent.cfg`ファイルで max_connections が 1 に設定されている場合、PEM エージェント起動プロセスはバックエンドコードで 2 に設定して、PEM のプローブデータ収集の失敗を回避します。 [サポートチケット #101924]
Enhancement	エージェントと監視対象サーバーの再起動時にプローブが再実行されるようになりました。
Enhancement	v12 REST API を導入して、アラート GET およびアラートテンプレート GET/PUT/POST の API スキーマを修正しました。
Bug fix	`pemworker`ユーティリティの使用法でバージョン情報が欠落している問題を修正しました。

次のページに続く

表 3 – 前のページからの続き

Type	Description
Bug fix	`reminder_notification_interval`設定に関する問題を修正しました。このパラメーターを 1 時間未満の値に構成できるようになりました。
Bug fix	アラートをコピーする際の Webhook 通知設定の問題を修正しました。アラートのコピー中に Webhook 通知設定がコピーされるようになりました。
Bug fix	REST API バージョン 11 の不一致のバージョンラベルに関する問題を修正しました。
Bug fix	PGD アラートの問題を修正しました。 PGD プローブを無効にすると、PGD アラートも無効になります。
Bug fix	ジョブタイプをバッチから SQL に変更する際の問題を修正しました。エラーなしでジョブタイプをバッチから SQL に、およびその逆に変更できるようになりました。
Bug fix	SQL の詳細情報を追加して、サーバーダウンおよびエージェントダウンのアラートテンプレートを更新しました。
Bug fix	アラートスクリプト実行のヘルプテキストを更新しました。ヘルプテキストには、スクリプト実行コードは SQL なくバッチ/シェル形式である必要があると記載されています。
Bug fix	`current_pid`の問題を修正しました。 `probe_schedule`テーブルで`agent_id`が null であっても、`current_pid`は null になりません。
Bug fix	SQL プロファイラーの権限エラーのメッセージの問題を修正しました。
Bug fix	`user_info`プローブの問題を修正して、正確なパスワードの有効期限を取得します。
Bug fix	ServerHeartbeat の問題を修正しました。 DB サーバーが PEM UI から削除されると、ServerHeartbeat が解放されます。
Bug fix	キャパシティマネージャーに関する問題を修正しました。 `New Tab`および`Previous Tab`が選択されたときに、キャパシティマネージャーのオプション`report type`が HTML を自動的に選択するようになりました。

1.2.2 Postgres Enterprise Manager 9.6.0 release notes

リリース日 2024 年 5 月 15 日

PEM 9.6.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	pemworker ユーティリティを使用してプローブを有効にするオプションを追加しました。
Enhancement	PEM は、失敗したジョブの電子メール通知にエラーメッセージを含めるようになりました。エラー出力のデフォルトの文字制限は 1000 で、サーバー構成の `job_notification_error_limit`パラメーターを使用して設定できます。

次のページに続く

表 4 – 前のページからの続き

Type	Description
Enhancement	2 つのアラートテンプレート *Inactive replication slots* `および `*Conflicting replication slots* と 2 つのプローブ *Replication slots* `および `*Multixact ID Exhaustion (Wraparound)* を追加しました。
Enhancement	新しく登録されたエージェントのデフォルトの表示名はホストの完全修飾ドメイン名 FQDN になり、FQDN が設定されていない場合は、以前のデフォルトのホスト名に戻ります。
Enhancement	より詳細な SQL クエリーでアラートテンプレート `Table size in server/database/schema`を更新しました。
Enhancement	Capacity Manager レポートを JSON 形式でダウンロードするオプションを追加しました。
Enhancement	pemworker を使用して Barman サーバーを登録するときに、クライアント証明書とキーを指定せずに CA 証明書パスを指定できるようになりました。
Bug fix	GUI で log ステートメントパラメーターが空のままである場合、監査マネージャーが `edb_audit_statement`パラメーターを'none' ではなく" に設定していたため、サーバーの再起動が失敗した問題を修正しました。
Bug fix	タイムゾーン/タイムスタンプが特定の値に変更されるたびに、`os_info`プローブがエラーをスローしていた問題を修正しました。
Bug fix	アラートがアラートダッシュボードにリストされているときにエラー「*NoneType* has no *len()*」が発生する問題を修正しました。
Bug fix	チームロールが予想通りに PEM エージェントで機能しなかった問題を修正しました。
Bug fix	PEM エージェント登録の SQL クエリーの問題を修正しました。テーブルの接頭辞にパブリックスキーマ名を追加して、スキーマ名が検索パスにない場合でも SQL クエリーを正常に実行します。

1.2.3 Postgres Enterprise Manager 9.5.1 release notes

リリース日 2024 年 3 月 4 日

PEM 9.5.1 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Security fix	これは、 CVE-2023-50782 のセキュリティ修正です。バンドルされた Python 暗号化パッケージを 42.x にアップグレードしました。

1.2.4 Postgres Enterprise Manager 9.5.0 release notes

リリース日 2024 年 2 月 15 日

PEM 9.5.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	Webhook ペイロードに `EmailGroup`および `EmailGroupID`を追加しました。
Enhancement	EDB Postgres Advanced Server のユーザーのパスワードの有効期限を検証するためのアラートテンプレートを追加しました。また、有効期限が迫っているユーザーの数を確認するアラートテンプレート `A user expired in N days` `Number of users expiring in N days`に変更しました。[^*]
Enhancement	Manage Probes の Manage Probes ウィンドウに、プローブ構成をデフォルトにリセットするボタンを追加しました。
Enhancement	通知に **Cleared Alert** ボタンを追加しました。これにより、アラートが発生したときに通知を受信しながら、クリアされたアラートの通知を無効にすることができます。
Enhancement	EDB の詳細は PEM によって生成された Web サーバー証明書に追加されなくなるため、独自の Web サーバー証明書を作成できるようになりました。
Enhancement	ブラックアウトステータス、日付、時刻を変更したユーザーを識別する機能を追加しました。
Bug fix	削除されたカスタムプローブがアラートテンプレートの `Probe Dependency`タブに表示されたままである問題を修正しました。
Bug fix	`pem_agent`ユーザーに `custom_email_template`テーブルに対する権限を付与することにより、`missing permission`の問題を修正しました。
Bug fix	PEM アラートテンプレート `largest index by table-size percentage`からサイズ 0 のテーブルを除外しました。このテンプレートの詳細な SQL ステートメントも追加しました。
Bug fix	CLI が PEM API の古いバージョンを呼び出そうとしたため、コマンドが失敗する問題を修正しました。
Bug fix	これで、 `pem_user`および `pem_manage_efm`ロールのメンバーは EFM 機能にアクセスできるようになりました。 `pem_user`および `pem_server_service_manager`ロールのメンバーは、PEM GUI からデータベースサービスを開始/停止できます。
Bug fix	`session_info`プローブに `application_name`列を追加しました。
Bug fix	Webhook の `too few arguments`例外を修正しました。
Bug fix	`pg_hba.conf`ファイルのグラフ表示で、PEM ユーザーインターフェイスにログインしている現在のユーザー名が User 列に表示される問題を解決しました。 `pg_hba.conf`ファイルに各ユーザー名が表示されるようになりました。
Bug fix	ユーザーインターフェイスでダッシュボードを開く際の `Dashboard info not found`エラーを修正しました。

[^*] 送信アラートテンプレート `A user expired in N days` を使用して設定されたアラートは、新しいテンプレートを使用して再作成する必要があります。

1.2.5 Postgres Enterprise Manager 9.4.1 release notes

リリース日 2023 年 11 月 28 日

PEM 9.4.1 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Security fix	これは、CVE-2023-38545 および CVE-2023-38546 のセキュリティ修正です。このパッチは、libcurl-pem の問題を修正します。

1.2.6 Postgres Enterprise Manager 9.4.0 release notes

リリース日 2023 年 11 月 16 日

PEM 9.4.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	EDB Advanced Server、 EDB Postgres Distributed、および PostgreSQL 16 のサポートを追加しました。
Enhancement	REST API を介して PEM ジョブをスケジュールできるようになりました。
Enhancement	パフォーマンス診断を改善して、時間間隔で集計された統計を提供します。
Bug fix	`process_one_alert`および`get_detail_alert_info`ファンクションのスキーマ修飾を追加しました。
Bug fix	PostgreSQL バージョン 15 および 16 でデータベースを作成するために必要なオプションの ICU パラメーターを追加しました。
Bug fix	Barman サーバーを追加するときに、空のエージェントの詳細のフロントエンド検証を追加しました。
Bug fix	システム構成レポートをダウンロードするときの問題を修正しました。エージェントに不足しているパラメーターがある場合、または不明な状態にある場合、システム構成レポートはダウンロードされません。
Bug fix	サーバー構成パラメーター`deleted_objects_data_retention_time`を追加して、サーバー、エージェント、またはツールの古いデータのパージを改善します。

1.2.7 Postgres Enterprise Manager 9.3.1 release notes

リリース日 2023 年 10 月 12 日

PEM 9.3.1 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Security fix	これは、CVE-2023-5002 のセキュリティ修正です。このセキュリティ修正には、pgAdmin 脆弱性問題の更新が含まれています。このパッチは PEM サーバーにのみ影響します。PEM エージェントを更新する必要はありません。
Security fix	これは、CVE-2023-4863 のセキュリティ修正です。このセキュリティ修正には、libwebp の問題の更新が含まれています。このパッチは PEM サーバーにのみ影響します。PEM エージェントを更新する必要はありません。

1.2.8 Postgres Enterprise Manager 9.3.0 release notes

リリース日 2023 年 8 月 31 日

PEM 9.3.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	アラートのコピー機能の使用中に、既存のアラートをすべて削除する機能を追加しました。
Enhancement	バックエンドデータベースサーバーおよび監視サーバーとして、PostgreSQL および EDB Postgres Advanced Server 16 ベータサポートを追加しました。
Enhancement	デフォルトのメールテンプレートをカスタマイズするサポートを追加しました。
Enhancement	エージェントとサーバーのアラート履歴レポートのダウンロードのサポートを追加しました。
Enhancement	PEM サーバーの構成中に、既存のエージェント SSL 証明書とキーファイルを再利用するオプションを追加しました。
Bug Fix	エージェント REST API で無視するマウントポイントを追加/表示する機能を追加しました。
Bug Fix	*hostaddr* body パラメーターをサーバー REST API に追加しました。
Bug Fix	Webhook のペイロードからアラートの詳細の改行文字を削除することにより、問題を修正しました。
Bug Fix	`agent.cfg`ファイルの`allow_insecure_webhooks`パラメーターを使用して、クライアント SSL 証明書とキーファイルまたは CA 証明書を提供せずに Webhook エンドポイント登録を許可するサポートを追加しました。
Bug Fix	PEM Barman ダッシュボードに表示されるバックアップサイズに関連する問題を修正しました。
Bug Fix	監査ログのインポート中のフォーマットされていないデータの問題を修正しました。

1.2.9 Postgres Enterprise Manager 9.2.2 release notes

リリース日 2023 年 7 月 14 日

PEM 9.2.2 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Security fix	これは CVE-2023-2650 のセキュリティ修正であり、すべての Windows ユーザーに推奨されます。このセキュリティ修正には、PEM の Windows インストーラーにバンドルされている Apache HTTPD の更新が含まれています。このパッチは PEM サーバーにのみ影響します。PEM エージェントを更新する必要はありません。

1.2.10 Postgres Enterprise Manager 9.2.1 release notes

リリース日:2023 年 7 月 3 日

PEM 9.2.1 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Security fix	Flask 2.x のサポートを追加しました。これは CVE-2023-30861 のセキュリティ修正であり、すべてのユーザーにお勧めします。このパッチは PEM サーバーにのみ影響します。PEM エージェントを更新する必要はありません。

1.2.11 Postgres Enterprise Manager 9.2.0 release notes

リリース日 2023 年 5 月 24 日

PEM 9.2.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	パフォーマンス診断のデフォルトのサンプル終了時間は、ウィンドウが最初に開かれた時間ではなく **Now** になりました。 **Run** を選択するとデータが更新されるようになりました。
Bug fix	Streaming Analysis ダッシュボードの Replication Time Lag グラフステータスの問題を修正しました。この修正では、サーバーがダウン/オフラインの場合、ステータスが停止として表示されます。 [サポートチケット #91220]
Bug fix	`efm_cluster_node_status`の下のフェールオーバーマネージャーダッシュボードからエージェントステータスを削除しました。 [サポートチケット:#1228607]
Bug fix	PEM Web クライアントからサーバーを更新/作成中の `unknown error`問題を修正しました。

次のページに続く

表 13 – 前のページからの続き

Type	Description
Bug fix	無効なパラメーターに対して有効なメッセージを返す問題を修正し、PUT サーバー REST API の不要なパラメーターを削除しました。
Bug fix	暗号化パッケージを安全なバージョンに更新しました。
Bug fix	webpack(NPM) パッケージを安全なバージョンに更新しました。
Bug fix	アラートテンプレートを変更する際のパラメーターの数の増加/減少によって発生するエラーを修正しました。
Bug fix	起動時に BARMAN シングルトンオブジェクトを初期化するときに発生した`Static object initialization fiasco`問題を修正しました。
Bug fix	`Login Group/Roles 500 error: UNION types char and text cannot be matched`エラーを修正しました。このエラーは、バックエンドサーバーとして PostgreSQL または EDB Postgres Advanced Server 15 を使用してロールの依存関係タブを開こうとしたときに表示されていました。
Bug fix	トークンが無効な場合、デフォルトの Python エラーの代わりに役立つメッセージを提供します。 [サポートチケット #89276]
Bug fix	プローブの作成中の拡張機能名警告を修正しました。この警告は、有効な拡張機能名を指定した後でも表示されていました。
Bug fix	コマンドラインインターフェイスを使用して PEM に Barman サーバーを登録しようとした際の`--config-file`スイッチのエラーを修正しました。
Bug fix	ヘルプドキュメントと API リンクの欠落しているファビコンアイコンを修正しました。
Bug fix	Capacity Manager の Manage Template の Manage Template セクションでフォルダーを削除する際の警告メッセージを修正しました。
Bug fix	選択されたサーバーの数が **GoTo** リストに表示されるサーバーと一致しないチューニングウィザードレポートを修正しました。
Bug fix	アラートテンプレートのパラメーターのアラート作成のためのクライアントサイドの検証を追加しました。
Bug fix	`Log Analysis Expert`の日付/時刻の空のエントリの検証を追加しました。
Bug fix	期間が **履歴日数としきい値** から **開始時間と終了時間** に変更された後、チャートが更新されない問題を修正しました。
Bug fix	Audit Manager に空のログローテーションエントリの検証を追加しました。
Bug fix	グラフの作成中に、欠落していた拡張レベルのメトリックを追加しました。
Bug fix	システムアラートのパラメーター検証を削除しました。
Bug fix	PEM がエージェント登録中にユーザー指定の証明書とキーを上書きしていた問題を修正しました。 [サポートチケット #90841]

1.2.12 Postgres Enterprise Manager 9.1.1 release notes

リリース日 2023 年 3 月 31 日

PEM 9.1.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Bug fix	Ubuntu 22.04 で PEM エージェントを登録するときにエラーが発生する問題を修正しました。

1.2.13 Postgres Enterprise Manager 9.1.0 release notes

リリース日 2023 年 2 月 14 日

PEM 9.1.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description
Enhancement	EDB Advanced Server 15 および EDB Postgres Distributed 5 のサポートを追加しました。
Enhancement	プローブとアラートを個別に選択せずに、グループ内のすべてのサーバーにコピーする機能を追加しました。
Bug Fix	ユーザーが低または中アラートを送信するオプションを無効にしているときに、アラート通知が送信される問題を修正しました。 [サポートチケット #87664]
Bug Fix	スプール内に非アクティブなエンドポイントがある場合、Webhook が呼び出されない問題を修正しました。 [サポートチケット #87365]
Other	PGD Worker Errors プローブと、関連するアラートテンプレート、およびダッシュボードチャートは、誤ったデータを生成したため削除されました。これらは、将来のリリースでは新しいプローブに置き換えられます。

1.2.14 Postgres Enterprise Manager 9.0.0 release notes

リリース日 2022 年 12 月 09 日

PEM 9.0.0 の新機能、機能強化、バグ修正、およびその他の変更には次のものが含まれます。

Type	Description	ID
Enhancement	ユーザーインターフェイスのレイアウトが改善され、新しいブランドで更新されました。	PEM-4472

次のページに続く

表 16 – 前のページからの続き

Type	Description	ID
Enhancement	ユーザーがエージェント登録中にコマンドラインからエージェント ID と既存の SSL 証明書とキーファイルを指定できるようにして、信頼できる証明書のユーザーエクスペリエンスと使用を向上させます。[サポートチケット #1022060]	PEM-3546
Enhancement	サーバーグループを作成するための REST API を追加しました。 [サポートチケット #78430]	PEM-4403
Enhancement	REST API の OpenAPI ドキュメントを再検証、修正、および明確化しました。 [サポートチケット #84746]	PEM-4453
Enhancement	スーパーユーザーアカウントなしでパフォーマンス診断の使用を許可します。 [サポートチケット #82569]	PEM-4515
Enhancement	Postgres Enterprise Manager のバックエンドデータベースサーバーとしての PostgreSQL 15 のサポートを追加しました。	PEM-4558
Enhancement	PostgreSQL 15 の SQLProfiler プラグインを生成しました。	PEM-4599
Bug Fix	特定のトポロジで `number of lagging replicas`メトリックが誤った結果を返す問題を修正しました。	PEM-4631
Bug Fix	[ダッシュボード] > [サーバーアクティビティ] で 50 セッションのみを表示できる問題を修正しました。 [サポートチケット #82563]	PEM-4492
Bug Fix	logging_collector が ON のときにサーバーを再起動するのではなく、サーバーリロードジョブを作成していた Log Manager の問題を修正しました。	PEM-4593

1.3 Platform compatibility

PEM がサポートするプラットフォームとバージョンの詳細については、
Platform Compatibility を参照してください。

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

1.4 Hardware requirements for installing the PEM server on Linux

サーバーを監視し、ダッシュボードをレンダリングする際の最適なパフォーマンスを得るには、少なくとも次のシステムに PEM をインストールすることをお勧めします。

- 4 つの CPU コア
- 8 GB の RAM
- 100 GB のストレージ

データストレージには追加のディスク領域が必要です。リソースの使用量は、定義および有効化されたプローブと、監視対象データベースのアクティビティレベルによって異なります。PEM を使用しながらサーバーリソースを監視すると、初期システム構成を拡張する必要がある場合に通知されます。

第2章　Planning

2.1 Understanding Postgres Enterprise Manager components and architecture

Postgres Enterprise ManagerPEM は、単一のグラフィカルインターフェイスを介して複数の Postgres サーバーを監視および管理します。PEM は、インフラストラクチャの次の領域を監視できます。

- **ホスト** —1 つ以上のサーバー物理または仮想とそのオペレーティングシステム。
- **データベースサーバー** —ホストで実行されている PostgreSQL、EDB Postgres Advanced Server、または EDB Postgres Extended Server 旧 2ndQPostgres の 1 つ以上のインスタンス。
- **データベース** —1 つ以上のデータベースと、テーブルやインデックスなどのスキーマオブジェクト。

注釈: Postgres という用語は、PostgreSQL、EDB Postgres Advanced Server、または EDB Postgres Extended Server を指します。

PEM は個々のソフトウェアコンポーネントで構成されています。

- **PEM サーバー** —PEM サーバーは、データを監視するためのデータリポジトリであり、エージェントとクライアントが接続するサーバーです。PEM サーバーは、PostgreSQL のインスタンス、監視データを保存するための関連データベース、および Web サービスを提供するサーバーで構成されています。
- **PEM エージェント** —PEM エージェントは、タスクを実行し、エージェントホストおよび監視対象の Postgres インスタンスから PEM サーバーに統計を報告します。単一の PEM エージェントは、エージェントがインストールされている同じホストまたは複数のリモートホストに存在する Postgres の複数のインストール済みインスタンスを監視できます。PEM サーバーインストールの一部としてエージェントをインストールするか、スタンドアロンエージェントをインストールできます。
- **PEM Web クライアント** — PEM Web インターフェイスを使用すると、Postgres サーバーを管理および監視し、PEM 拡張機能を使用できます。Web インターフェイスソフトウェアは、PEM サーバーと共にインストールされ、サポートされている Web ブラウザーを使用してアクセスされます。
- **SQL プロファイラー** —SQL プロファイラーは、PEM で分析する SQL プロファイラーツールの監視データとクエリプランを記録する Postgres サーバー拡張機能です。これは PEM のオプショナルコンポーネントですが、拡張機能は、使用する Postgres の各インスタンスで有効にする必要があります。SQL プロファイラーは、PEM サーバーで管理されているものだけでなく、Postgres の EDB ディストリビューションのサポートされているバージョンで使用できます。詳細およびサポートされているバージョンについては、*Installing the SQL Profiler extension* を参照してください。

2.1.1 PEM アーキテクチャ

次のアーキテクチャ図は、PEM サーバー、クライアント、管理対象および非管理対象の Postgres サーバー間の関係を示しています。

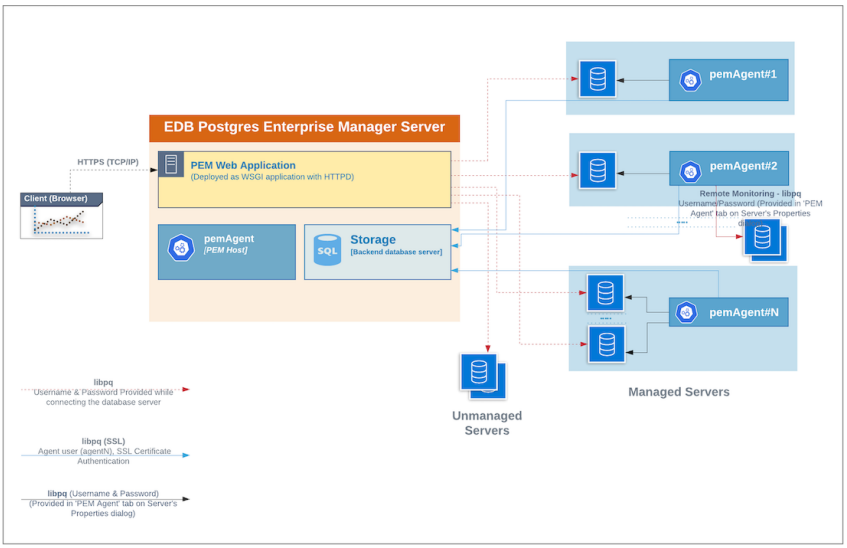


図 1: PEM Architecture

PEM サーバー

PEM サーバは Postgres のインスタンス、クライアントに Web サービスを提供する Apache Web サーバのインスタンス、および PEM エージェントで構成されています。PEM は、サーバーサイドの暗号化プラグインを使用して認証証明書を生成します。

Postgres のインスタンスデータベースサーバと Apache ウェブサーバ HTTPD のインスタンスは、同じホスト上または別のホスト上にあってもかまいません。

注釈: すべての PEM 機能は、選択したバックエンドデータベースサーバー PostgreSQL または EDB Postgres Advanced Server で利用できます。

- **Postgres インスタンスデータベースサーバー**—これはバックエンドデータベースサーバーです。 `pem` という名前のデータベースをホストし、PEM サーバーのリポジトリとして機能します。`pem` データベースには、各監視対象ホスト、サーバー、およびデータベースから収集されたメトリックデータを保存する複数のスキーマが含まれています。
  - `pem` —このスキーマは、PEM アプリケーションのコアです。これには、アプリケーションに必要な構成ファンクション、テーブル、またはビューの定義が含まれています。
  - `pemdata` —このスキーマには、監視対象データの現在のスナップショットが保存されます。
  - `pemhistory` —このスキーマには、履歴監視データが保存されます。

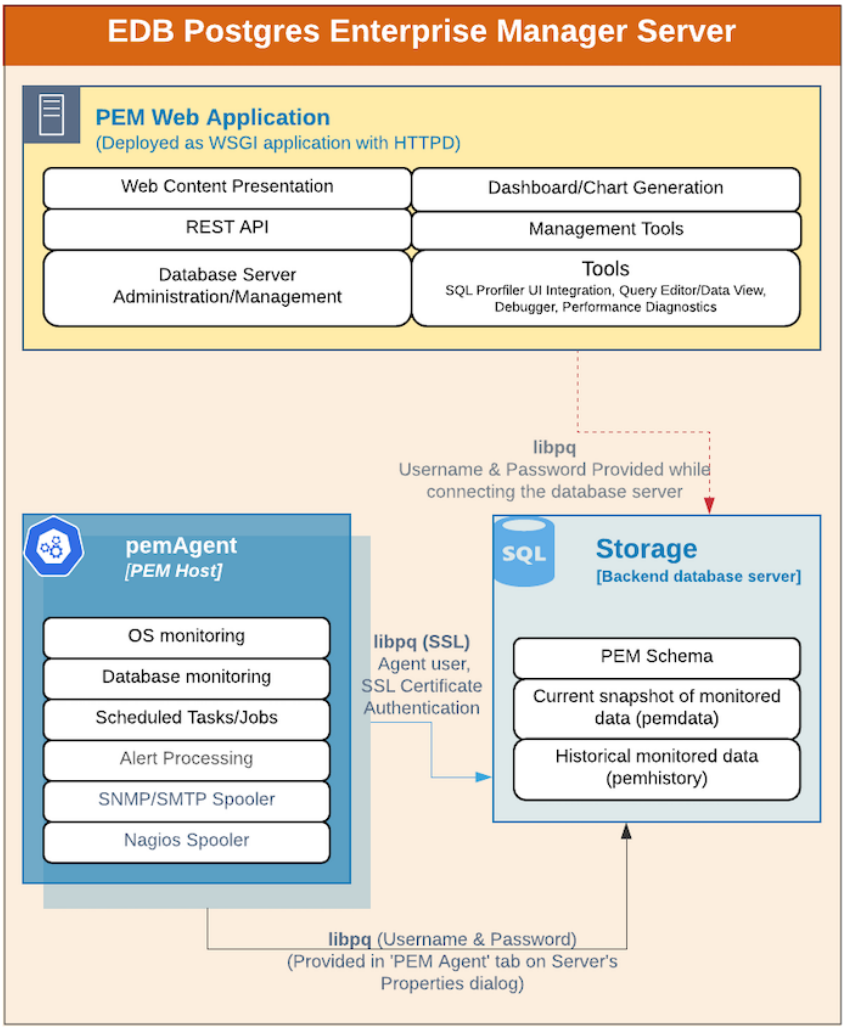


図 2: PEM Server

- **Apache Web サーバー HTTPD** — PEM Web アプリケーションは、HTTPD を使用した WSGI アプリケーションとして展開され、クライアントに Web サービスを提供します。これは次で構成されています。
  - **Web コンテンツプレゼンテーション**—プレゼンテーションレイヤーは、Web アプリケーションブラウザやログインページなどによって作成されます。
  - **Rest API** — REST API により、他のアプリやサービスとの統合が可能になります。
  - **データベースサーバーの管理/management** —データベースサーバーの管理と、管理対象サーバーと非管理対象サーバーの CREATE、ALTER、DROP などの管理アクティビティを実行できます。
  - **ダッシュボード/チャートの生成**—内部的に、Web アプリケーションには、ダッシュボードとチャートを生成する機能が含まれています。
  - **管理ツール** — Audit Manager、Capacity Manager、Log Manager、Postgres Expert、Postgres Log Analysis Expert、およびチューニングウィザードは、Web ppplication.
  - で利用できます。他のツールは、管理対象または非管理対象サーバーで機能を提供します。
    - **SQL プロファイラー UI の統合** &mdash ; SQL プロファイラーは、セッションコンテンツの簡単に分析されたトレースを生成します。
    - **クエリエディタ/データビュー** — Query Editor を使用すると、データの照会、編集、および表示ができます。
    - **Debugger** —デバッガーは、クエリーのデバッグに役立ちます。
    - **パフォーマンス診断**—パフォーマンス診断は、Postgres インスタンスのパフォーマンスを分析するのに役立ちます。

専用マシンを使用して、PEM バックエンドデータベースの運用インスタンスをホストすることをお勧めします。監視対象のデータベースサーバーの数、およびサーバーが処理しているワークロードによっては、ホストは高レベルのデータスループットの影響を受ける場合があります。

PEM エージェント

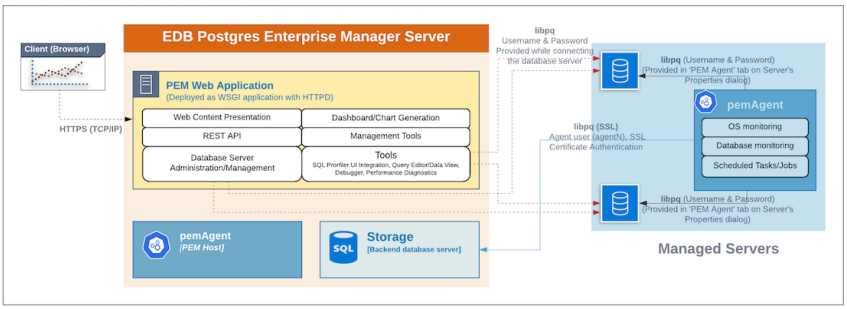


図 3: PEM Agent

PEM エージェントは、マシンとオペレーティングシステムと、それらがバインドされている各 Postgres インスタンスから監視データを収集します。各 PEM エージェントは 1 つの物理マシンまたは仮想マシンを監視でき、他のシステムにインストールされている複数のデータベースサーバーをローカルに監視できます。これらのサーバーは、同じシステムまたはリモートにインストールできます。また、サーバーのシャット

ダウン、SQL プロファイラーのトレース、カスタムジョブなど、ユーザーがスケジュールした他のタスクを実行する責任もあります。

PEM エージェントは、PEM サーバーのインストール時にデフォルトで PEM サーバーにインストールされます。通常、PEM ホスト上の PEM エージェントと呼ばれます。別に、PEM を使用して監視する Postgres インスタンスをホスティングする他のサーバーに PEM エージェントをインストールすることもできます。

ローカルで監視するかリモートでモニタリングかにかかわらず、PEM エージェントは、SSL 証明書ベースの認証を使用して、PostgreSQL の libpq を使用して PEM サーバーに接続します。Windows の PEM エージェントインストーラーと Linux の pemworker CLI は、各エージェントを PEM サーバーに登録し、必要な証明書を生成およびインストールします。

PEM エージェントと PEM サーバーの間は一方通行だけです。PEM エージェントは常に PEM サーバーに接続します。

PEM エージェントは、監視対象の各データベースサーバーに接続できる必要があります。この接続は、TCP/IP 接続またはオプションで Unix ホスト上の Unix ドメインソケットを介して確立され、オプションで SSL を使用できます。監視対象サーバーへの接続と認証を構成する必要があります。

構成が完了すると、各エージェントは、ホストと、監視対象の各データベースサーバーとデータベースに関する統計情報やその他の情報を収集します。各情報は**メトリック** と呼ばれ、 **プローブ**によって収集されます。ほとんどのプローブは、効率のために一度にマルチプルのメトリックを収集します。収集されるメトリックの例は次のとおりです。

- ディスク I/O 統計
- ネットワーク統計
- データベースサーバーバージョン文字列
- データベースサーバー構成オプション GUC 値
- テーブルアクセス統計
- テーブルとインデックスのサイズ

PEM プローブのリストについては、 *Probes* を参照してください。

デフォルトでは、データベースサーバにバインドされた PEM エージェントは OS /データベースモニタリング統計を収集し、その特定のデータベースサーバのスケジュールされたタスク/ジョブを実行し、PEM サーバーの pem データベースにデータを保存します。

アラート処理、SNMP / SMTP スプーラー、および Nagios スプーラーデータは、PEM サーバー上の pem データベースに保存され、デフォルトで PEM ホスト上の PEM エージェントによって処理されます。ただし、PEM エージェントの SNMP / SMTP および Nagios パラメーターを調整することにより、他の PEM エージェントによる処理を有効にできます。

これらのパラメーターの詳細については、 **サーバー構成パラメーター - リファレンス** を参照してください。

PEM Web クライアント

PEM クライアントは、サポートされているブラウザーで実行される Web ベースのアプリケーションです。クライアントの Web インターフェイスは PEM サーバーに接続し、管理対象または非管理対象サーバーと、それらに存在するデータベースとスキーマを直接管理できます。

クライアントでは、ダッシュボード、 Postgres Log Analysis Expert、Capacity Manager などの機能を介してサーバーに記録されたデータを利用する PEM 機能を使用できます。

SQL プロファイラー拡張機能

すべてのサーバーに SQL プロファイラー拡張機能をインストールする必要はありませんが、SQL プロファイラーを使用する各サーバーに拡張機能をインストールして有効にする必要があります。監視されていない開発サーバーに SQL プロファイラー拡張機能をインストールして有効にすることもできます。アドホックに使用するために SQL プロファイラー拡張機能を一時的にインストールすることもできます。

PEM で管理されていないサーバーで SQL プロファイラーを使用できます。ただし、スケジュールされたトレースを実行するには、サーバーに拡張機能をインストールして有効にし、インストールおよび構成済みの PEM エージェントで管理する必要があります。

詳細については、 *Installing the SQL Profiler extension* および *Using SQL Profiler* を参照してください。

2.2 Deployment considerations

Postgres Enterprise Manager を展開する前に、考慮する必要があることが数多くあります。

Considerations	Implementation instructions
スタンドアロンサーバーで十分ですか、それとも高可用性アーキテクチャが必要ですか。	Installing the server or *Query information*
接続プーリングを実装する必要がありますか?	Deploying connection pooling
どのような認証の種類を使用しますか?	*Authentication options*
セキュリティの脆弱性を回避するにはどのようなアクションを実行する必要がありますか?	*Securing your deployment*
Apache Web サーバーをホストする場所はどこですか?	*Query information*

2.2.1 Using Failover Manager for high availability

注釈: この手順は、既存のインストールではなく、新規インストールで PEM サーバーのフェールオーバーマネージャーを設定するための手順です。提供されたコマンドは、Web サーバーサービスに HTTPD が使用される RHEL ベースのシステムの構成に適用されます。

Postgres Enterprise ManagerPEM は、データベース管理者、システムアーキテクト、およびパフォーマンスアナリストが Postgres データベースサーバーを管理、監視、および調整するのに役立ちます。

Failover Manager は、EDB の高可用性ツールで、プライマリでソフトウェアまたはハードウェアに障害が発生しているときに Postgres プライマリノードをスタンバイノードにフェールオーバーできます。以下の例では、次の IP アドレスを使用します。

- 172.16.161.200-PEM プライマリ
- 172.16.161.201-PEM スタンバイ 1
- 172.16.161.202-PEM スタンバイ 2
- 172.16.161.203 - EFM Witness ノード
- 172.16.161.245-PEM VIP エージェントとユーザーが接続するために使用します

以下は VIP アドレスを使用する必要があります。

- 監視対象データベースサーバーの PEM エージェントバインディング
- PEM Web クライアントへのアクセス
- ウェブサーバーサービスへのアクセス

製品の初期インストールと構成

1. プライマリと 1 つ以上のスタンバイに以下をインストールします。
    - PEM サーバーのバックエンドデータベース
    -
    -

これらのリンクを使用して製品ドキュメントのインストール手順を参照するか、
EDB repos website の手順を参照してください。

EDB リポジトリにアクセスする手順の `USERNAME:PASSWORD` を自分のユーザー名とパスワードに置き換えます。

PEM サーバー構成スクリプトは信頼できる認証では機能しないため、データベースサーバーが scram-sha-256 認証方法を使用するように構成されていることを確認します。

EFM をインストールするには、`java-1.8.0-openjdk` パッケージをインストールする必要があります。

2. プライマリサーバーとすべてのスタンバイサーバーで、タイプ 1 Web サービスとデータベースの初期構成を使用して PEM サーバーを構成します。

```
    /usr/edb/pem/bin/configure-pem-server.sh -t 1
``` 構成タイプの詳細については、を参照してください。

3. プライマリサーバーとすべてのスタンバイサーバーのファイアウォールに次のポートを追加して、アクセスを許可します。

(次のページに続く)

(前のページからの続き)

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

例

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

ストリーミングレプリケーションのプライマリノードを設定する

1. レプリケーションロールを作成します。

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

選択したパスワードを入力します。

2. postgresql.conf ファイルで次を構成します。

```
wal_level = replica
max_wal_senders = 10
wal_keep_size = 500
max_replication_slots = 10
```

ストリーミングレプリケーションのパラメーターの構成の詳細については、
PostgreSQL documentation を参照してください。

注釈: 構成パラメーターは、データベースサーバーのバージョンが異なると異なる場合があります。これらのパラメーターの設定についてのヘルプが必要な場合は、EDB サポート techsupport@enterprisedb.com に電子メールを送信できます。

```

3. ホストベースの認証/var/lib/edb/as13/data/pg_hba.conf ファイルに次のエントリを追加して、レプリケーションユーザーがすべてのスタンバイから接続できるようにします。

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

注釈: 必要に応じて、IP アドレスの cidr レンジを変更できます。

4. pem_user ロールのホストベースの認証/var/lib/edb/as13/data/pg_hba.conf ファイルを変更して、scram-sha-256 認証方法を使用してすべてのデータベースに接続します。

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

5. EPAS 13 サーバーを再起動します。

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

ストリーミングレプリケーションのスタンバイノードを設定する

1. すべてのスタンバイノードで EPAS 13 のサービスを停止します。

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

注釈: この例では、 pg_basebackup ユーティリティを使用して、スタンバイサーバーに PEM バックエンドデータベースサーバーのレプリカを作成します。 pg_basebackup を使用する場合、既存のデータベースサーバーを停止し、既存のデータディレクトリを削除する必要があります。

2. すべてのスタンバイノードでデータベースサーバーのデータディレクトリを削除します。

```
$ sudo su - enterprisedb

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

3. すべてのスタンバイノードの enterprisedb ユーザーのホームディレクトリに.pgpass ファイルを作成します。

```
$ sudo su - enterprisedb
```

(次のページに続く)

(前のページからの続き)

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

$ chmod 600 ~/.pgpass
```

4. pg_basebackup を使用して、各スタンバイノードでプライマリノードのバックアップを取得します。

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

backup コマンドは、スタンバイノードに postgresql.auto.conf および standby.signal ファイルを作成します。postgresql.auto.conf ファイルには次のコンテンツが含まれています。

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

5. 各スタンバイノードの postgresql.conf ファイルで、次のパラメーターを編集します。

```
hot_standby = on
```

6. 各スタンバイノードで EPAS 13 データベースサーバーを起動します。

```
$ systemctl enable edb-as-13

$ systemctl start edb-as-13
```

7. 次のファイルをプライマリノードからスタンバイノードの同じ場所にコピーし、既存のファイルを上書きします。ファイルのアクセス許可を設定します。

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

例

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

このコードは、ウェブサーバーがスタンバイで構成され、デフォルトで無効になっていることを確認します。EFM によるスイッチオーバーにより、Web サーバーが有効になります。

注釈: 証明書が更新されるたびに、マスターとスタンバイで証明書の同期を手動で維持します。

---

8. configure-selinux.sh スクリプトを実行して、PEM の SELinux ポリシーを構成します。

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

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

9. HTTPD および PEM エージェントサービスがすべてのレプリカノードで実行されている場合、無効にして停止します。

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

注釈: この時点で、PEM プライマリサーバーと 2 つのスタンバイは、必要に応じてプライマリから引き継ぐ準備が整います。

すべてのホストでフェールオーバーを管理するように EFM を設定する

1. EFM をサポートするようにプライマリノードを準備する
   - データベースサーバーに接続するデータベースユーザー efm を作成します。
   - WAL ログに関連する機能の実行特権とモニタリング特権をユーザーに付与します。
   - pg_hba.conf にエントリを追加して許可しますすべてのホストのすべてのノードからデータベースサーバーに接続する efm データベースユーザー。
   - すべてのデータベースサーバーの構成をリロードします。例

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

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

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

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

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

(次のページに続く)

(前のページからの続き)

```
GRANT
GRANT
GRANT
GRANT ROLE

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

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

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

2. 各ノードでスクリプトを作成して、PEM エージェントを起動/停止します。

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

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

3. 各ノードで sudoers ファイル /etc/sudoers.d/efm-pem を作成して、efm ユーザーが pemagent を起動/停止できるようにします。

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

(次のページに続く)

(前のページからの続き)

```
efm    ALL=(ALL)           NOPASSWD:   /bin/systemctl status pemagent
_EOF_
```

4. サンプルファイル/etc/edb/efm-4.1/efm.nodes.in を使用してすべてのノードで efm.nodes ファイルを作成し、efm OS ユーザーに読み取り/書き込みアクセスを付与します。

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

5. スタンバイノードの/etc/edb/efm-4.1/efm.nodes ファイルに、プライマリノードの IP アドレスと efm ポートを追加します。

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

6. サンプルファイル/etc/edb/efm-4.1/efm.properties.in を使用して、すべてのノードで efm.properties ファイルを作成します。すべてのユーザーに読み取りアクセスを付与します。

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

7. efm ユーティリティを使用して efm ユーザーのパスワードを暗号化します。

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

8. プロパティファイルの次のパラメーターを編集します。

```
db.user=efm
db.password.encrypted=096666746b05b081d1a98e43d94c9dad
db.port=5444
db.database=edb
db.service.owner=enterprisedb
db.service.name=edb-as-13
db.bin=/usr/edb/as13/bin
db.data.dir=/var/lib/edb/as13/data
jdbc.sslmode=require
user.email=username@example.com
from.email=node1@efm-pem
notification.level=INFO
notification.text.prefix=[PEM/EFM]
bind.address=172.16.161.200:7800
```

(次のページに続く)

(前のページからの続き)

```
admin.port=7809
is.witness=false
local.period=10
local.timeout=60
local.timeout.final=10
remote.timeout=10
node.timeout=50
encrypt.agent.messages=true
stop.isolated.primary=true
stop.failed.primary=true
primary.shutdown.as.failure=false
update.physical.slots.period=0
ping.server.ip=8.8.8.8
ping.server.command=/bin/ping -q -c3 -w5
auto.allow.hosts=false
stable.nodes.file=false
db.reuse.connection.count=0
auto.failover=true
auto.reconfigure=true
promotable=true
use.replay.tiebreaker=true
standby.restart.delay=0
reconfigure.num.sync=false
reconfigure.sync.primary=false
minimum.standbys=0
recovery.check.period=1
restart.connection.timeout=60
auto.resume.period=0
virtual.ip=172.16.161.245
virtual.ip.interface=ens33
virtual.ip.prefix=24
virtual.ip.single=true
check.vip.before.promotion=true
pgpool.enable=false
sudo.command=sudo
sudo.user.command=sudo -u %u
syslog.host=localhost
syslog.port=514
syslog.protocol=UDP
syslog.facility=LOCAL1
file.log.enabled=true
syslog.enabled=false
jgroups.loglevel=INFO
efm.loglevel=INFO
jvm.options=-Xmx128m
```

(次のページに続く)

(前のページからの続き)

```
script.remote.post.promotion=/usr/local/bin/stop-pemagent.sh
script.post.promotion=/usr/local/bin/start-pemagent.sh
```

9. 監視ノードの is.witness 構成パラメーターの値を true に設定します。

```
is.witness=true
```

10. プライマリノードで EFM サービスを有効にして開始します。

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

11. スタンバイがプライマリノードで起動されたクラスターに参加できるようにします。

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

12. スタンバイノードと EFM 監視ノードで EFM サービスを有効にして開始します。

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

13. 任意のノードから EFM クラスターのステータスを確認します。

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

Allowed node host list:
    172.16.161.200 172.16.161.201 172.16.161.202 172.16.161.203

Membership coordinator: 172.16.161.200

Standby priority host list:
    172.16.161.201 172.16.161.202

Promote Status:

    DB Type     Address              WAL Received LSN   WAL Replayed LSN   Info
    ---------------------------------------------------------------------------
```

(次のページに続く)

(前のページからの続き)

```
Primary     172.16.161.200                          0/F7A3808
Standby     172.16.161.201      0/F7A3808           0/F7A3808
Standby     172.16.161.202      0/F7A3808           0/F7A3808

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

このステータスは、EFM が正常にセットアップされ、PEM サーバーのフェールオーバーを管理していることを確認します。

フェールオーバーが発生した場合、スタンバイのいずれかがプライマリノードとして昇格し、PEM エージェントが新しいプライマリノードに接続します。この手順を使用して、障害が発生したプライマリノードを新しいスタンバイに置き換えることができます。

現在の制限

現在の制限には次のものが含まれます。

- ユーザーの Web コンソールセッションは、スイッチオーバー中に失われます。
- [設定] ダイアログボックスから設定したユーザーごとの設定は、ファイルシステム上のローカル構成ファイルに保存されているため、失われます。
- バックグラウンドプロセス、バックアップ、リストア、およびメンテナンスダイアログボックスによって開始され、そのログはシステム間で共有されません。これらは、スイッチオーバー中に失われます。

2.2.2 Connection pooling using PgBouncer

PEM エージェントから非 Windows マシン上の Postgres Enterprise ManagerPEM サーバーへの接続数を制限するための接続プーラーとして PgBouncer を使用できます。

- *PEM server and agent connection management mechanism* は、PgBouncer-PEM インフラストラクチャの紹介を提供します。
- *Preparing the PEM database server* は、PgBouncer で使用する PEM データベースサーバーの準備に関する情報を提供します。
- *Configuring PgBouncer* は、PEM データベースサーバーと連携できるように PgBouncer の構成に関する詳細情報を提供します。
- *Configuring the PEM agent* は、PgBouncer に接続するための PEM エージェントの構成に関する詳細情報を提供します。

PEM Web インタフェースの使用の詳細については、*Accessing the web interface* を参照してください。

PEM server and agent connection management mechanism

PgBouncer なし

デフォルト構成では、各 PEM エージェントは、暗号化に SSL を使用して PEM データベースサーバーに直接接続します。各 PEM エージェントは、接続に独自の専用ユーザーを使用します。

たとえば、ID 1 のユーザエージェントは、agent1 として PEM データベースサーバーに接続します。

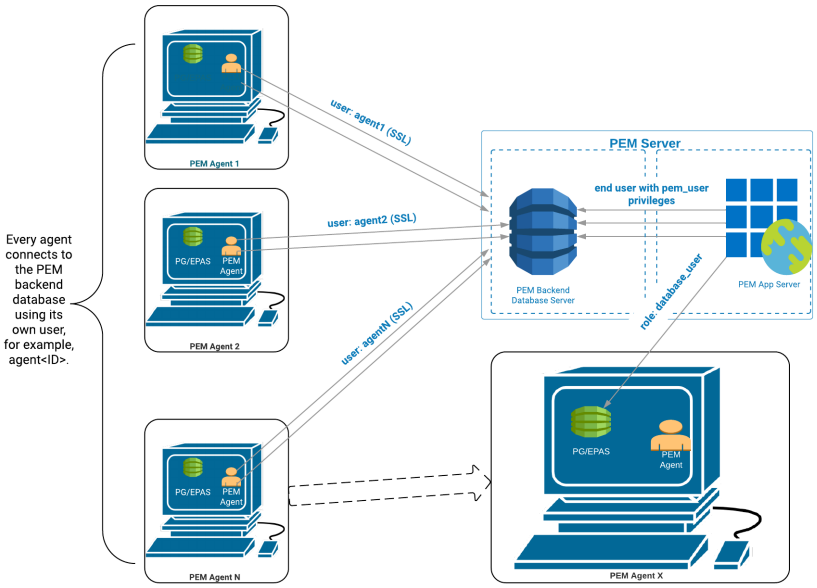


図 4: Connecting to the PEM database without PgBouncer

PgBouncer を使用する場合

PgBouncer 対応環境では、PEM エージェントは PEM データベースサーバーに直接接続できません。PEM エージェントは、PgBouncer への接続専用に構成したプロキシユーザーを使用する必要があります。この例では、すべての PEM エージェント接続を処理するプロキシユーザーの名前は `pem_agent_user1` です。

PEM エージェントが SSL を使用して PgBouncer に接続すると、PgBouncer は PEM データベースサーバーへの接続要求を管理します。PgBouncer は、接続プーリング用に構成されたルールを使用して、たとえば、アクティブな接続の確立された最大数を尊重することにより、着信接続要求を管理します。

!!!note 接続プーラーとしては PgBouncer バージョン 1.9.0 以降を使用します。バージョン 1.9.0 以降は、証明書認証をサポートしています。

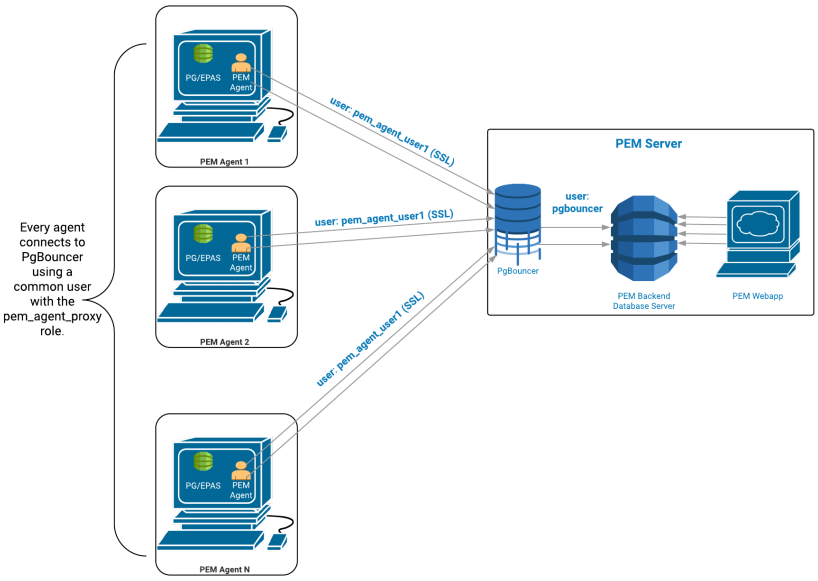


図 5: Connecting to pgBouncer.

Preparing the PEM database server

PEM データベースサーバーで専用ユーザーを構成し、SSL キーと証明書を作成して、PgBouncer で PEM の接続プーリングを有効にする必要があります。

この例では、 EDB Postgres Advanced Server バージョン 16 を使用する RHEL ベースのオペレーティングシステムで `enterprisedb` ユーザーを使用して PEM データベースサーバーを準備する方法を示しています。データの場所、構成ファイルとキーファイル、および構成を実行するために使用するユーザー OS および Postgres ディストリビューションによって異なる場合があります。

前提条件

- PEM データベースサーバーの `pem` データベースに接続しています。
- `enterprisedb` または `postgres` ユーザーとして接続しています。ユーザーは Postgres ディストリビューションに依存しています。

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

PgBouncer-PEM 接続のユーザーとロールの作成

1. pem_agent_pool メンバーシップで pgbouncer という名前の専用ユーザーを作成します。このユーザーは、すべてのエージェントデータベースクエリを転送することにより、PgBouncer から PEM データベースへの接続を提供します。

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

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

1. pem_admin および pem_agent_pool ロールメンバーシップを持つ pem_admin1 スーパーユーザーではないという名前のユーザーを作成します。このユーザーは、エージェントを PEM サーバーに登録し、PEM データベースへのアクセスを管理するために使用されます。

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

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

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

1. pgbouncer ユーザーに CONNECT 特権を付与します。

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

1. pem スキーマの pgbouncer ユーザーに USAGE 特権を付与します。

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

1. pem.get_agent_pool_auth(text) ファンクションで pgbouncer ユーザーに EXECUTE 特権を付与します。例

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

1. pem.create_proxy_agent_user(varchar) ファンクションを使用して、pem_agent_user1 という名前のユーザーを作成します。このプロキシユーザーは、すべてのエージェントと PgBouncer 間の接続を提供します。

```
SELECT pem.create_proxy_agent_user(pem_agent_user1);
__OUTPUT__
create_proxy_agent_user
-------------------------
(1 row)
```

このファンクションは、同じ名前とランダムなパスワードを持つユーザーを作成し、pem_agent および pem_agent_pool ロールをユーザーに付与します。このアプローチにより、PgBouncer は、エージェントに代わってプロキシユーザーを使用できます。

構成ファイルを更新して PgBouncer-PEM 接続を許可する

1. PEM データベースサーバーの pg_hba.conf ファイルに hostssl pem エントリーを追加することにより、pgbouncer ユーザーが SSL 認証方法を使用して pem データベースに接続できるようにします。

ルールのリストで、+pem_agent ユーザーに割り当てられた他のルールの前に hostssl pem エントリーを配置してください。

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

(次のページに続く)

(前のページからの続き)

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

1. 次の行を$PGDATA/pg_ident.conf ユーザーマッピングファイルに追加することにより、PEM サーバーが PgBouncer-PEM 接続に関係するすべてのユーザーをマップできるようにします。

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

1. Postgres サービスを再起動します。 <postgres_service> プレースホルダーを Postgres インスタンスの systemd サービス名の名前に置き換えます。

```
systemctl restart <postgres_service>
```

PgBouncer-PEM 認証の SSL キーと証明書の作成

pem_agent_pool グループロールのキーと証明書を作成します。次に、ファイルを PgBouncer インスタンスに移動して、PEM データベースサーバーと PgBouncer 間の認証を許可します。

この例では、RHEL で EDB Postgres Advanced Server を実行します。環境変数を設定するときは、オペレーティングシステムと Postgres ディストリビューションに応じて正しいディレクトリを選択します。

1. $DATA_DIR 環境変数をデータディレクトリに設定します。

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

OS および Postgres バージョンごとのデータディレクトリ

ここで、オペレーティングシステムおよび Postgres バージョンごとの他のデフォルトデータディレクトリの例をいくつか示します。

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

1. $USER_HOME 環境変数を、ユーザーがアクセスできるホームディレクトリに設定します。

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

OS および Postgres バージョンごとのユーザーホームディレクトリ

ここで、オペレーティングシステムおよび Postgres バージョンごとの他のデフォルトのホームディレクトリの例をいくつか示します。

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

1. openssl で署名キーを作成します。

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

1. 証明書署名要求 CSR を作成します。必要に応じて、<...> の-subj 属性を置き換えます。 Common NameCN が pem_agent_pool グループロール名に設定されていることを確認します。

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

1. PEM CA とキーを使用して CSR に署名します。

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

1. 作成した鍵と証明書を enterprisedb ユーザーがアクセスできるパスに移動します。

この例では、 enterprisedb ユーザーのホームディレクトリに~/.postgresql というフォルダーを作成し、権限があることを確認します。

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

Configuring PgBouncer

PEM データベースサーバーと連携するように PgBouncer を構成する必要があります。

前提条件

- EDB Postgres Advanced Server を実行している場合、インストールは完了しています。

または

- EDB Postgres Extended Server または PostgreSQL を実行している場合、コミュニティ PgBouncer がインストールされています。

EDB PgBouncer および PgBouncer インストールの考慮事項

構成手順でのディレクトリとファイルの名前と場所、およびユーザーは、PgBouncer のコミュニティバージョンと EDB PgBouncer のどちらをインストールしたかによって異なります。コミュニティ PgBouncer をインストールしている場合は、コミュニティリポジトリと EDB リポジトリからインストールするかにかかわらず、作業例のファイルとディレクトリの名前を PgBouncer の値に置き換えます。

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

PgBouncer の構成

この例では、 `enterprisedb` システムユーザーを使用して EDB PgBouncer を構成します。

コミュニティ PgBouncer を実行している場合は、*EDB PgBouncer および PgBouncer インストールの考慮事項* の説明に従って、ディレクトリ、ファイル、およびユーザーの名前を置き換えます。

1. ターミナルウィンドウを開き、PgBouncer ディレクトリに移動します。
2. PgBouncer の `etc` ディレクトリ `edb-pgbouncer.ini` が存在する場所の所有者を `enterprisedb` に変更し、ディレクトリのアクセス許可を `0700` に変更します。

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

1. `edb-pgbouncer.ini` ファイルの内容を変更します。

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

  [pgbouncer]
  logfile = /var/log/edb/pgbouncer<1.x>/edb-pgbouncer-<1.x>.log
  pidfile = /var/run/edb/pgbouncer<1.x>/edb-pgbouncer-<1.x>.pid
  listen_addr = *
  ;; Agent needs to use this port to connect the pem database now
  listen_port = 6432
  ;; Set to require to ensure SSL certificates are used for connections
  ;; for PEM Agents
  client_tls_sslmode = require
  ;; These are the root.crt, server.key, server.crt files present
  ;; in the present under the data directory of the PEM database
  ;; server, used by the PEM Agents for connections.
  client_tls_ca_file = /var/lib/edb/as16/data/root.crt
  client_tls_key_file = /var/lib/edb/as16/data/server.key
  client_tls_cert_file = /var/lib/edb/as16/data/server.crt
  ;; Allow pgBouncer to use pem_agent_pool certificate
  ;; and key for connections to the server.
  server_tls_key_file = /var/lib/edb/.postgresql/pem_agent_pool.key
  server_tls_cert_file = /var/lib/edb/.postgresql/pem_agent_pool.crt
  ;; Use hba file for client connections
  auth_type = hba
  ;; HBA file
  auth_hba_file = /etc/edb/pgbouncer<1.x>/hba_file
  ;; Use pem.get_agent_pool_auth(TEXT) function to authenticate
  ;; the db user (used as a proxy agent user).
  auth_query = SELECT * FROM pem.get_agent_pool_auth($1)
  ;; DB User for administration of the pgbouncer
  admin_users = pem_admin1
  ;; auth_dbname and auth_user allow
  ;; admin console login by admin_users and stats_users
  auth_dbname = pem
  auth_user = pgbouncer
  ;; DB User for collecting the statistics of pgbouncer
  stats_users = pem_admin1
  server_reset_query = DISCARD ALL
  ;; Change based on the number of agents installed/required
```

(次のページに続く)

(前のページからの続き)

```
max_client_conn = 500
;; Close server connection if its not been used in this time.
;; Allows to clean unnecessary connections from pool after peak.
server_idle_timeout = 60
```

!!!note *auth_user* の詳細については、 Authentication settings を参照してください。 .. ::

1. 次のコンテンツを含む PgBouncer の HBA ファイル *(/etc/edb/pgbouncer<1.x>/hba_file)* を作成します。

```
# Use the authentication method scram-sha-256 for local connections
# between the pem database & the pgbouncer (virtual) database.
local pgbouncer all scram-sha-256
# Use the authentication method scram-sha-256 for remote connections
# to pgbouncer (virtual database) using the enterprisedb user.
host pgbouncer,pem pem_admin1 0.0.0.0/0 scram-sha-256
# Use the authentication method cert for TCP/IP connections
# to the pem database using pem_agent_user1
hostssl pem pem_agent_user1 0.0.0.0/0 cert
```

1. HBA ファイル (/etc/edb/pgbouncer<1.x>/hba_file) の所有者を enterprisedb に変更し、ディレクトリ権限を 0600 に変更します。

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

1. PgBouncer サービスを有効にし、サービスを開始します。

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

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

Configuring the PEM agent

前提条件

installed the PEM agent があります。

!!!note PgBouncer を介して接続するように、agent.cfg ファイルで true に設定された enable_smtp 、enable_snmp 、または enable_webhook を使用して PEM エージェントを構成しないでください。SNMP、SMTP、および Webhook スプーラーは、Postgres が提供する LISTEN/NOTIFY メカニズムを使用して、通知を非同期に送信します。PgBouncer はトランザクションモードでの LISTEN/NOTIFY メカニズムをサポー

トしていないため、エージェントを PgBouncer に接続すると、通知が遅延されるか、まったく配信されない場合があります。代わりに、PEM エージェントを PEM バックエンドデータベースに直接接続します。

PgBouncer の **新しい *PEM* エージェントの構成** または **既存の *PEM* エージェントの構成** を選択できるようになりました。

新しい PEM エージェントの構成

PEM エージェントをインストールした後、特定の PEM データベースサーバーと連携するように構成します。次のコマンドを使用します。

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

このコマンドでは、 `--pem-agent-user` 引数は、pem_agent_user1 データベースユーザーの SSL 証明書とキーペアを`/root/.pem` ディレクトリに作成するようにエージェントに指示します。

例

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

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

PEM エージェントは、キーを使用して PEM データベースサーバーに pem_agent_user1 として接続します。また、`/usr/edb/pem/agent/etc/agent.cfg` という名前のエージェント構成ファイルも作成します。

使用する agent-user を記載した行が `agent.cfg` 構成ファイルにあります。例

```
cat /usr/edb/pem/agent/etc/agent.cfg
[PEM/agent]
pem_host=172.16.254.22
pem_port=6432
agent_id=12
agent_user=pem_agent_user1
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
log_level=warning
log_location=/var/log/pem/worker.log
agent_log_location=/var/log/pem/agent.log
long_wait=30
```

(次のページに続く)

(前のページからの続き)

```
short_wait=10
alert_threads=0
enable_smtp=false
enable_snmp=false
enable_webhook=false
max_webhook_retries=3
allow_server_restart=true
allow_package_management=false
allow_streaming_replication=false
max_connections=0
connect_timeout=-1
connection_lifetime=0
allow_batch_probes=false
heartbeat_connection=false
```

既存の PEM エージェントの構成

既存の PEM エージェントを使用している場合、SSL 証明書とキーファイルをターゲットマシンにコピーし、ファイルを再利用できます。ファイルを変更し、新しいパラメーターを追加して、既存の `agent.cfg` ファイル内のいくつかのパラメーターを置き換える必要があります。

1. agent_user をエージェントとして使用する行を追加します。

```
agent_user=pem_agent_user1
```

1. ポートを更新して PgBouncer ポートを指定します。

```
pem_port=6432
```

1. 証明書とキーパスの場所を更新します。

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

別の方法として、エージェント自己登録スクリプトを実行できます。ただし、そのプロセスは新しいエージェント ID を作成します。エージェント自己登録スクリプトを実行する場合、新しいエージェント ID を既存の ID に置き換え、`pem.agent` テーブルの新しいエージェント ID のエントリを無効にする必要があります。例

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

UPDATE 1
```

注釈: 既存の SSL 証明書、キーファイル、およびエージェント構成ファイルのバックアップを保存します。

2.2.3 Authentication options

PEM は、Kerberos と 2FA 認証もサポートしています。実装手順については、以下を参照してください。

Linux の場合

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

Linux および Windows の場合

- *Configuring the PEM server to use two-factor authentication*

Configuring the PEM server to use Kerberos authentication

PEM サーバーの Kerberos 認証を構成できます。Kerberos サーバーは、IP アドレスではなくホスト名で動作します。Kerberos 認証を使用して PEM サーバーでシングルサインオンを使用するには、DNS レルムを使用してホスト名で次のマシンを構成します。

・Kerberos サーバー ・PEM サーバー PEM Web サーバーおよび PEM バックエンドデータベースサーバー・クライアントマシン

たとえば、Kerberos サーバー上のレルムが `edbpem.org` の場合、Kerberos サーバーホスト名を `Krb5server.edbpem.org` に、PEM サーバーホスト名を `pem.edbpem.org` に、クライアントのホスト名を `pg12.edbpem.org` に設定できます。規約は、DNS ドメイン名を次の名前として使用することです。レルム。

1.Kerberos、PEM サーバー、および PEM バックエンドデータベースをインストールします

認証サーバーとして機能するマシンに Kerberos をインストールします。PEM サーバーを別のマシンにインストールします。詳細については、*Installing the PEM server and PEM-HTTPD on separate hosts* を参照してください。

PEM バックエンドデータベース Postgres/ EDB Postgres Advanced Server を PEM サーバーと同じマシンまたは別のマシンにインストールします。詳細については、

EDB Docs website のインストール手順を参照してください。

2.Kerberos サーバーにプリンシパルを追加する

Apache Web サーバー HTTPD / Apache2 および PEM バックエンドデータベースサーバー PostgreSQL / EDB Postgres Advanced Server の下に展開された PEM Web アプリケーションのプリンシパルを追加します。

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

HOSTNAME_OF_PEM_SERVER には、Kerberos サーバーのレルムが含まれている必要があります。たとえば、レルムとして edbpem.org を使用して、PEM サーバーのホスト名として pemdb.edbpem.org を指定できます。

注釈: PEM Web アプリケーションと PEM バックエンドデータベースサーバーが別のマシンにある場合、ホスト名はそれぞれに異なります。

3.Kerberos サーバーからキーテーブルを抽出します

PEM Web アプリケーションと PEM バックエンドデータベースサーバーの Kerberos からキーテーブルを抽出します。

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

キーテーブルを Kerberos サーバーから PEM サーバーにコピーします。

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

PEM サーバーで、キーテーブルを必要な場所に移動し、所有権を変更します。

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

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

そこで

- NAME_OF_PEM_WEB_FILE は、PEM Web アプリケーションのキーテーブルに指定された名前です。
- NAME_OF_PEM_DB_FILE は、PEM バックエンドデータベースサーバーのキーテーブルに指定された名前です。

- OS_USERNAME_ON_PEM_SERVER は、PEM サーバー上のオペレーティングシステムユーザーの名前です。
- DATA_DIRECTORY_OF_POSTGRES は、インストールされた Postgres データベース PostgreSQL/ EDB Postgres Advanced Server のデータディレクトリのパスです。

4.PEM バックエンドデータベースサーバーの構成

postgresql.conf ファイルにキーテーブルの場所を追加します。

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

そこで

- NAME_OF_PEM_DB_FILE は、PEM バックエンドデータベースサーバーのキーテーブルに指定された名前です。
- DATA_DIRECTORY_OF_POSTGRES は、インストールされた Postgres データベース PostgreSQL/ EDB Postgres Advanced Server のデータディレクトリのパスです。

krb5.conf ファイルを編集します。

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

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

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

データベースサーバーを再起動して、変更を反映します。

```
systemctl restart <POSTGRES_SERVICE_NAME>
```

POSTGRES_SERVICE_NAME は、 Postgres PostgreSQL/ EDB Postgres Advanced Server データベースのサービス名です。たとえば、RHEL または Rocky Linux プラットフォーム上の PostgreSQL 13 データベースの postgresql-13 です。

5. 最初のチケットを取得して表示します

kinit ユーティリティは、Kerberos チケットを取得およびキャッシュします。通常、このユーティリティを使用して、チケット交付チケットを取得し、パスワードを入力してキー配布センター KDC から資格情報を復号します。チケット交付チケットは、資格情報キャッシュに保存されます。

klist ユーティリティを使用して、チケットの詳細を表示できます。

注釈: kinit および klist を使用する前に、PEM サーバーとクライアントマシンに Kerberos クライアントをインストールします。

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

プリンシパルと Kerberos チケットが表示されます。

注釈: ここで指定される `USERNAME@REALM` は、 *pem* データベースに対する pem_admin ロールと CONNECT 特権を持つデータベースユーザーである必要があります。

6.PEM サーバーの構成

PEM サーバーで PEMconfigure スクリプトを実行して、Kerberos 認証を使用します。

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

config_setup.py ファイルで PEM_DB_HOST を構成します。PEM_AUTH_METHOD の値が'kerberos' に設定されていることを確認します。

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

.install-config ファイルでホストを構成します。

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

PEM サーバーが Kerberos 認証を使用する場合

- すべての監視対象サーバーはデフォルトで同じ認証を使用します。デフォルトをオーバーライドするには、config_local.py ファイルに、パラメーター ALLOW_DATABASE_CONNECTION_WITHOUT_KERBEROS を追加し、True に設定します。

- 認証されたユーザープリンシパルはすべてレルム USERNAME@REALM が追加され、デフォルトでデータベースユーザー名として渡されます。デフォルトをオーバーライドするには、config_local.py ファイルに、パラメーター PEM_USER_KRB_INCLUDE_REALM を追加し、False に設定します。
- Apache サーバーを再起動します

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

- gss 認証方法を使用するように pg_hba.conf の上部にあるエントリを編集し、データベースサーバーをリロードします。

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

```
systemctl reload <POSTGRES_SERVICE_NAME>
```

POSTGRES_SERVICE_NAME は、 Postgres PostgreSQL/ EDB Postgres Advanced Server データベースのサービス名です。たとえば、RHEL または Rocky Linux プラットフォーム上の PostgreSQL 13 データベースの postgresql-13 です。

注釈: PostgreSQL または EDB Postgres Advanced Server 12 以降を使用している場合、接続タイプを `hostgssenc` として指定して、gss 暗号化された接続のみを許可できます。

---

7. ブラウザ設定

クライアントマシンのブラウザーを構成して PEM Web クライアントにアクセスし、Spnego / Kerberos を使用します。

Mozilla Firefox の場合

1. about:config ページをロードして、低レベルの Firefox 構成ページを開きます。
2. 検索ボックスに「network.negotiate-auth.trusted-uris 」と入力します。
3. network.negotiate-auth.trusted-uris 設定をダブルクリックし、Kerberos HTTP SPNEGO によって保護される Web サーバーのホスト名またはドメインを入力します。複数のドメインとホスト名はコンマで区切ります。
4. 検索ボックスに「network.negotiate-auth.delegation-uris 」と入力します。
5. network.negotiate-auth.delegation-uris 設定をダブルクリックし、Kerberos HTTP SPNEGO によって保護される Web サーバーのホスト名またはドメインを入力します。複数のドメインとホスト名はコンマで区切ります。
6. **OK** を選択します。

Linux または MacOS での Google Chrome の場合

- google-chrome コマンドに--auth-server-whitelist パラメーターを追加します。たとえば、Linux プロンプトから Chrome を実行するには、次のように google-chrome コマンドを実行します。

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

- PEM サーバーを構成すると、ブラウザーで PEM Web インターフェイスにアクセスできます。次の場所に移動します

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

注釈: Postgres クラスターへの接続中に、次のエラーが表示される場合があります。

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

ktutil を使用するか、AD のすべての暗号システムで Postgres キータブを再作成して、暗号化タイプをキータブに追加します。

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

Windows Active Directory ドメインサービスは、IP アドレスではなくホスト名で動作します。Active Directory ドメインサービスを使用して PEM サーバーでシングルサインオンを使用するには、DNS を使用してホスト名で次のマシンを構成します。

・Windows サーバードメインコントローラ

- PEM サーバー PEM Web サーバーおよび PEM バックエンドデータベースサーバー
- クライアントマシン

たとえば、 Windows Active Directory のレルムが `edbpem.internal` の場合、 Windows サーバーホスト名を `Krb5server.edbpem.internal` に、PEM サーバーホスト名を `pem.edbpem.internal` に、クライアントのホスト名を `pg12.edbpem.internal` に設定できます。

1. Active Directory、PEM サーバー、および PEM バックエンドデータベースサーバーのインストール

次のインストールを実行します。

・ 認証サーバーとして機能する Windows サーバードメインコントローラに Active Directory をインストールします。また、Kerberos 認証を使用するように Active Directory ドメインサービスを構成し、起動します。

- PEM サーバーを別の Linux マシンにインストールします。詳細については、*Installing the PEM server and PEM-HTTPD on separate hosts* を参照してください。
- PEM バックエンドデータベース Postgres/ EDB Postgres Advanced Server を PEM サーバーと同じ Linux マシンまたは別の Linux マシンにインストールします。詳細については、EDB Docs website のインストール手順を参照してください。

2. サービスプリンシパルとマップする Active Directory にユーザーを作成する

Windows サーバーの Active Directory にユーザーを作成して、PEM Web アプリケーションの HTTP サービスプリンシパルとマップします。

1. **Active Directory ユーザーとコンピューター > <DOMAIN_NAME> > Users** を開きます。右クリックして、 **New > User** を選択します。

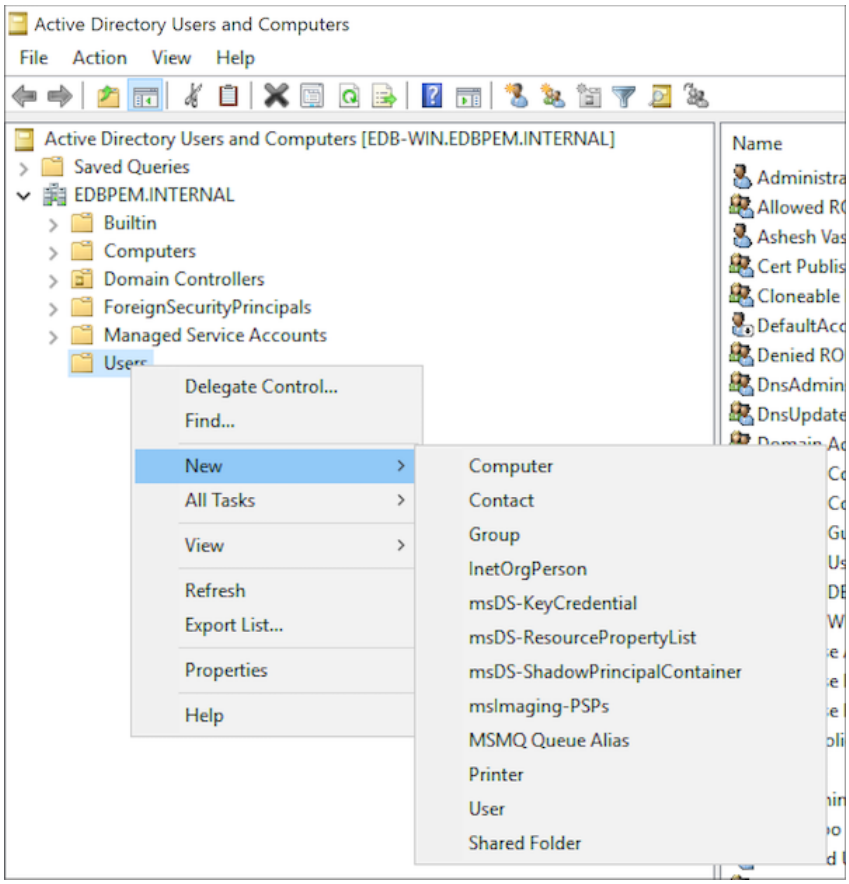


図 6: Active Directory Users and Computers

1. ユーザーの詳細を入力します。
2. パスワードを入力し、**ユーザーは次回ログオンにパスワードを変更する必要があります**をクリアします。また、 **ユーザーはパスワードを変更できません** と**パスワードの有効期限はありません** を選択します。
3. ユーザーの詳細を確認します。
4. [PEMServer Web Properties] ダイアログボックスで、ユーザーを Domain Users グループのメンバーとして追加します。

1. Windows サーバーのアクティブディレクトリにユーザー pemserverdb などを作成して、PEM バックエンドデータベースの Postgres サービスプリンシパルとマップします。

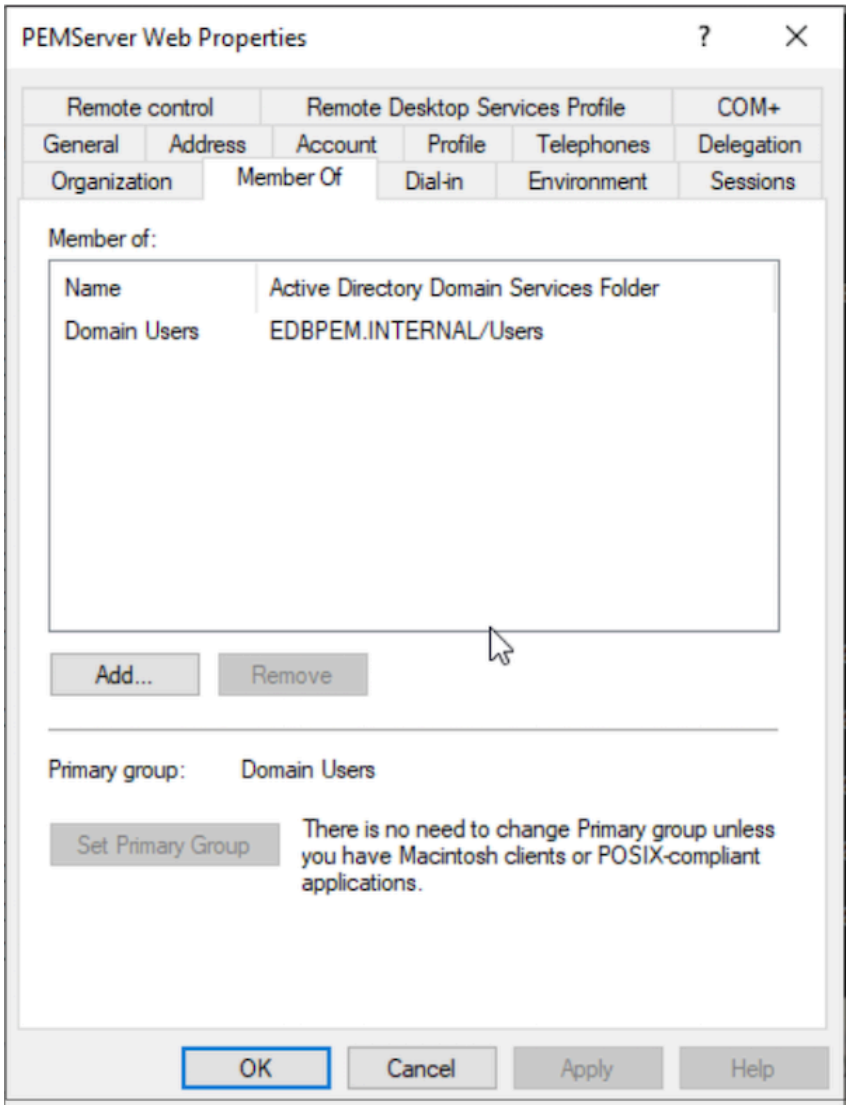


図 7: PEM Server Web Properties

3. Active Directory からキーテーブルを抽出する

サービスプリンシパルのキーテーブルを抽出し、作成した各ドメインユーザーにマップします。

1. 管理者として Windows PowerShell を開きます。ユーザー pemserverweb を使用して HTTP サービスプリンシパルマッピングのキーテーブルを作成し、ユーザー pemserverdb を使用して Postgres サービスプリンシパルマッピングのキーテーブルを作成します。

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

そこで

- pemserver.keytab は、PEM Web アプリケーションのキーテーブルの名前です。 - pemdb.keytab は、PEM バックエンドデータベースサーバーのキーテーブルの名前です。 - pem.edbpem.internal@EDBPEM.INTERNAL は、PEM サーバーのホスト名です。ここで、@EDBPEM.INTERNAL は@REALM を意味します。- pemserverweb は、PEM Web アプリケーションのユーザーです。 - pemserverdb は、PEM バックエンドデータベースサーバーのユーザーです。 - EDBPEM.INTERNAL は、ターゲットのドメインです。

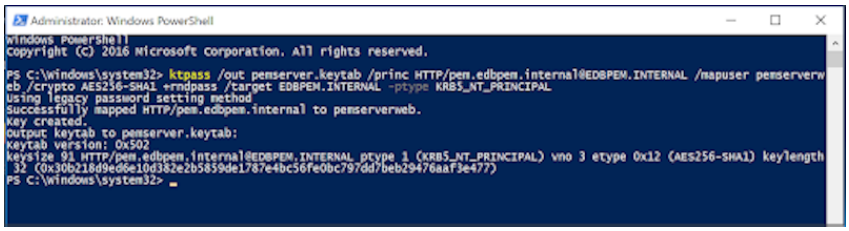


図 8: Windows PowerShell ktpass

注釈: コマンドライン引数`+rndpass` は、ドメインユーザー pemserverweb のパスワードをランダムなパスワードにリセットします。 */target* オプションはオプショナル。

---

1. [**Accounts**] タブで、ユーザーアカウントの Kerberos サポートを追加します。

注釈: [**Accounts**] タブでは、ユーザーログオン名に**HTTP/pem.edbpem.internal@EDBPEM.INTERNAL** が表示されます。 [**Delegation**] タブは pemserverweb ユーザーに対して有効になっています。

---

1. [**Delegation**] タブで、作成したユーザーに **Trust this user for delegation to any serviceKerberos Only)** を選択します。

2. 両方のキーテーブルを PEM サーバーホスト、または PEM Web サーバーと PEM バックエンドデータベースサーバーホストにコピーします。別のホストにインストールされている場合。

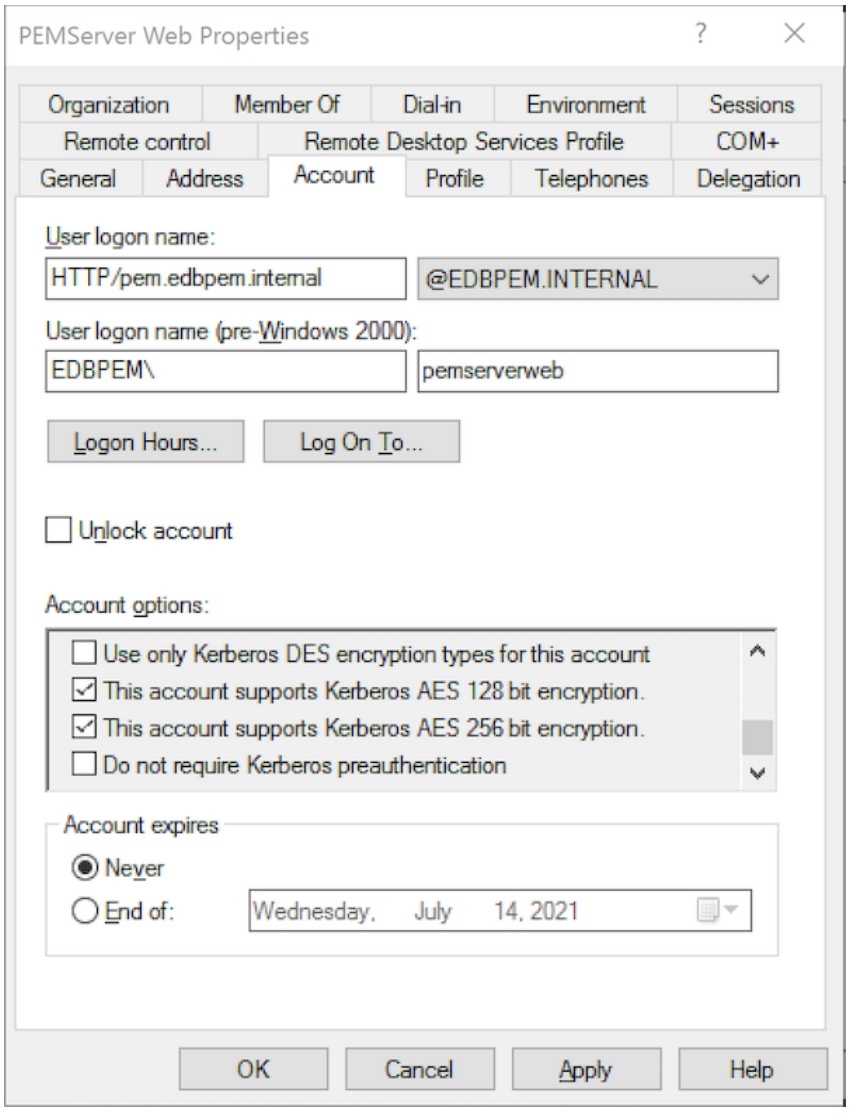


図 9: PEM Server Web Properties

3. PEM サーバーで、キーテーブルを必要な場所に移動し、所有権を変更します。

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

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

そこで

- OS_USERNAME_ON_PEM_SERVER は、PEM サーバー上のオペレーティングシステムユーザーの名前です。 - DATA_DIRECTORY_OF_POSTGRES は、インストールされた Postgres データベース PostgreSQL/ EDB Postgres Advanced Server のデータディレクトリのパスです。

4.PEM バックエンドデータベースサーバーの構成

postgresql.conf ファイルにキーテーブルの場所を追加します。

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

DATA_DIRECTORY_OF_POSTGRES は、インストールされた Postgres データベース PostgreSQL/ EDB Postgres Advanced Server のデータディレクトリのパスです。

krb5.conf ファイルを編集します。

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

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

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

データベースサーバーを再起動して、変更を反映します。

```
systemctl restart <POSTGRES_SERVICE_NAME>
```

POSTGRES_SERVICE_NAME は、 Postgres PostgreSQL/ EDB Postgres Advanced Server データベースのサービス名です。たとえば、RHEL または Rocky Linux プラットフォーム上の PostgreSQL 13 データベースの postgresql-13 です。

5. 最初のチケットを取得して表示します

kinit ユーティリティは、Kerberos チケットを取得およびキャッシュします。通常、このユーティリティを使用して、キー配布センター KDC から資格情報を復号化するために入力したパスワードを使用して、チケット交付チケットを取得します。チケット交付チケットは、資格情報キャッシュに保存されます。

klist ユーティリティを使用して、チケットの詳細を表示できます。

注釈: kinit および klist を使用する前に、PEM サーバーとクライアントマシンに Kerberos クライアントをインストールする必要があります。

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

プリンシパルと Kerberos チケットが表示されます。

注釈: ここで指定される `USERNAME@REALM` は、 *pem* データベースに対する pem_admin ロールと CONNECT 特権を持つデータベースユーザーである必要があります。

6.PEM サーバーの構成

PEM サーバーで PEMconfigure スクリプトを実行して、Kerberos 認証を使用します。

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

config_setup.py ファイルで、PEM_DB_HOST を構成し、PEM_AUTH_METHOD の値が'kerberos' に設定されていることを確認します。

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

.install-config ファイルで HOST を構成します。

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

PEM サーバーが Kerberos 認証を使用する場合

- すべての監視対象サーバーはデフォルトで同じ認証を使用します。デフォルトをオーバーライドするには、config_local.py ファイルに、パラメーター ALLOW_DATABASE_CONNECTION_WITHOUT_KERBEROS を追加し、True に設定します。
- 認証されたユーザープリンシパルはすべてレルム USERNAME@REALM が追加され、デフォルトでデータベースユーザー名として渡されます。デフォルトをオーバーライドするには、config_local.py ファイルに、パラメーター PEM_USER_KRB_INCLUDE_REALM を追加し、False に設定します。

Apache サーバーを再起動します。

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

gss 認証方法を使用するように pg_hba.conf の上部にあるエントリを編集します。次に、データベースサーバーをリロードします。

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

```
systemctl reload <POSTGRES_SERVICE_NAME>
```

POSTGRES_SERVICE_NAME は、 Postgres PostgreSQL/ EDB Postgres Advanced Server データベースのサービス名です。たとえば、RHEL または Rocky Linux プラットフォーム上の PostgreSQL 13 データベースの postgresql-13 です。

注釈: 接続タイプを`hostgssenc` として指定することはできません。 Windows は gss 暗号化接続をサポートしていません。

---

7. ブラウザ設定

クライアントマシンのブラウザーを構成して PEM Web クライアントにアクセスし、Spnego / Kerberos を使用します。

Mozilla Firefox の場合

- about:config ページをロードして、低レベルの Firefox 構成ページを開きます。
- 検索ボックスに「network.negotiate-auth.trusted-uris 」と入力します。
- network.negotiate-auth.trusted-uris 設定をダブルクリックし、Kerberos HTTP SPNEGO によって保護された Web サーバーのホスト名またはドメインを入力します。複数のドメインとホスト名はコンマで区切ります。
- 検索ボックスに「network.negotiate-auth.delegation-uris 」と入力します。
- network.negotiate-auth.delegation-uris 設定をダブルクリックし、Kerberos HTTP SPNEGO によって保護された Web サーバーのホスト名またはドメインを入力します。複数のドメインとホスト名はコンマで区切ります。
- **OK** を選択します。

Linux または MacOS での Google Chrome の場合

- google-chrome コマンドに--auth-server-whitelist パラメーターを追加します。たとえば、Linux プロンプトから Chrome を実行するには、次のように google-chrome コマンドを実行します。

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

- PEM サーバーを構成すると、ブラウザーで PEM Web インターフェイスにアクセスできます。次の場所に移動します

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

注釈: Postgres クラスターへの接続中に、次のエラーが表示される場合があります。

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

ktutil を使用するか、AD のすべての暗号システムで Postgres キータブを再作成して、暗号化タイプをキータブに追加します。

Configuring the PEM server to use two-factor authentication

PEM は、2FA の 2 つの方法をサポートしています。

- メール認証
- 認証アプリ Google Authenticator など

2FA を有効にするには、これらの設定を `config.py` ファイルから `config_local.py` ファイルにコピーし、次のパラメーターを変更できます。

Parameter	Description
MFA_ENABLED	二要素認証を有効にするには、`true`に設定します。デフォルト値は`false`です。
MFA_FORCE_REGISTR/	ユーザーにログイン時に二要素認証方法を強制的に登録するように求めるには`true`に設定します。デフォルト値は`false`です。
MFA_SUPPORTED_METI	メール認証方法を使用する場合は`email`に設定し、TOTP ベースのアプリケーション認証方法を使用する場合は`authenticator`に設定します。
MFA_EMAIL_SUBJECT	メール認証のメールの件名に設定します。デフォルト値は`<APP_NAME> - Verification Code`です。

メールサーバーの構成

メール認証方法を使用するには、メールサーバーの設定を構成する必要があります。

PEM サーバーは、PEM 構成に保存された SMTP 構成、または Flask-Mail を使用して電子メールを送信できます。

PEM 構成から内部 SMTP 構成を使用してメール確認コードを送信するには、パラメーター `MAIL_USE_PEM_INTERNAL` `True` に設定します。 `False` に設定されている場合、次のメール構成は使用され、ユーザー指定のメールアドレスにコードを送信します。

- MAIL_SERVER = 'localhost'
- MAIL_PORT = 25
- MAIL_USE_TLS = False
- MAIL_USE_SSL = False
- MAIL_USERNAME = なし
- MAIL_PASSWORD = なし
- MAIL_DEFAULT_SENDER = なし

これらの構成の詳細については、 Flask-Mail documentation を参照してください。

注釈: PEM SMTP アラートはこの構成を使用しません。

2.2.4 Securing your deployment

PEM を構成するときに、考慮する必要があるセキュリティ慣行があります。これらのセキュリティ対策はオプショナル。ただし、EDB は、システム全体のセキュリティを強化するためにこれらのプラクティスをお勧めします。

PEM は、Apache Web サーバー、OpenSSL、snmp++、libcurl など、ベンダーリポジトリのサードパーティコンポーネントに依存しています。これらのコンポーネントを最新の状態にするには、オペレーティングシステムを更新し、セキュリティ更新を定期的に適用して、セキュリティの脆弱性を回避します。最新のセキュリティパッチが適用されていない場合、システムはサイバー攻撃に対して脆弱です。セキュリティパッチは、最新の更新を適用して最新の脅威から守ることにより、デバイスとそのデータを保護します。

セキュリティパッチを定期的に適用することの利点には、次のようなものがあります。

- サイバー攻撃への露出の削減
- 生産性の損失の回避
- ランサムウェアのようなマルウェアからデータを保護する
- セキュリティの抜け穴を使用してネットワーク上に拡散するワーム感染の回避

Apache HTTPD security configurations

Windows、Apache HTTPD の名前は PEM HTTPD です。 Apache HTTPD 構成ファイルは `pem.conf` であり、SSL 構成ファイルは `httpd-ssl-pem.conf` です。どちらの構成ファイルも `<Apache_Installation_Path>/conf/addons` ディレクトリにあります。

Linux では、Apache HTTPD 構成ファイルは `edb-pem.conf` であり、SSL 構成ファイルは `edb-ssl-pem.conf` です。どちらの構成ファイルも `<Apache_Installation_Path>/conf.d` ディレクトリにあります。

SSLv2 および SSLv3 を無効にする

SSL バージョン SSLv2、SSLv3、TLS 1、および TLS 1.1 を無効にします。これらのバージョンは最も脆弱であり、暗号化に関する懸念があります。

バージョンを無効にするには、次のコマンドを Apache HTTPD 構成ファイルに追加します。

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

Web サーバーを再起動して、構成ファイルに変更を適用します。

PEM は、SSL 構成ファイルに次の行を追加して、セキュリティとして TLS 1.2 を使用できるようにします。

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

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

SSL 証明書を使用したセキュア HTTPD

Web アプリケーションの SSL セキュリティの追加レイヤーをお勧めします。

PEM のインストール中に、PEM は Apache / HTTPD サーバーの自己署名証明書を生成および使用します。PEM に独自の SSL 証明書を使用する場合は、Apache 構成ファイルを更新する必要があります。

Linux では、Apache HTTPD 構成ファイル `edb-ssl-pem.conf` をテキストエディタで開き、ファイルへの書き込み権限を持つユーザーである必要があります。証明書ファイルと一致するように、構成ファイルのサーバー名とファイル名を変更する必要もあります。

PEM VirtualHost セクションで、これらの 2 つの SSL 指示子を更新します。

- `SSLCertificateFile` は、DigiCert 証明書ファイルたとえば、your_domain_name.crt です。
- `SSLCertificateKeyFile` は、CSR たとえば、your_private.key などの作成時に生成された.key ファイルです。

例、次の更新を行います。

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

HTTPD 自己署名 SSL 証明書を、PEM の信頼できる CA 署名証明書に置き換えることもできます。手順については、この
Postgres tutorial を参照してください。

Web サーバー情報の公開を無効にする

Web サーバーセキュリティの一環として、すべての Web サーバー署名を無効にすることをお勧めします。Web サーバーの署名を無効にするには、次のパラメーターを Apache HTTPD 構成ファイルに追加します。PEM は、Apache HTTPD 構成ファイルにパラメーターを追加することにより、情報の公開を無効にします。

```
ServerTokens Prod
ServerSignature Off
```

ServerTokens 指示子は、クライアントに返されるサーバー応答ヘッダーフィールドを制御します。このパラメーターを Apache HTTPD 構成ファイルに追加して、Apache サーバーバージョンを非表示にすることをお勧めします。

ServerSignature 指示子には、サーバーで生成されたドキュメントのフッターが含まれます。フッターには、Apache やオペレーティングシステムのバージョンなどの Apache 構成に関する情報が含まれています。この情報の公開を制限するには、Apache HTTPD 構成ファイルで指示子を無効にすることをお勧めします。

Web サーバーを再起動して、Apache HTTPD 構成ファイルに変更を適用します。

ディレクトリリストを無効にする

ディレクトリリストにより、攻撃者はディレクトリの完全なコンテンツを表示できます。デフォルトでは、Web サーバーはこのリストを有効にし、攻撃者が任意のファイルを検出および表示できるようにします。このリストにより、攻撃者はアプリケーションをリバースエンジニアリングして、ソースコードを取得し、考えられるセキュリティ上の欠陥が分析され、アプリケーションに関する詳細情報を発見する可能性があります。

このリスクを回避するには、Apache HTTPD 構成ファイルで Options 指示子を設定することにより、ディレクトリリストを無効にします。PEM は、Web サーバー構成ファイルで次のオプションを設定することにより、ディレクトリリストを無効にします。

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

Web サーバーを再起動して、構成ファイルに加えた変更を適用します。

ネットワークまたは IP アドレスへのアクセスを制限する

Apache では、アプリケーションにアクセスできるクライアントホスト名または IP アドレスを提供できます。IP アドレスまたはネットワークへのアクセスを付与するには、 Allow 指示子にネットワークアドレスを入力します。

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

(次のページに続く)

(前のページからの続き)

```
Allow from 192.168.0.0/24
</Directory>
```

アプリケーション構成ファイルの ALLOWED_HOSTS 構成パラメーターは、IP アドレスごとに許可ホストを設定します。アプリケーション構成ファイル config_local.py は<PEM_INSTALLATION_PATH>/web にあります。

PEM により、すべてのホストがアプリケーションに接続できます。

例

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

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

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

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

アプリケーション構成ファイルの変更を適用するには、Web サーバーを再起動します。

クロスサイトトレース

TRACE と TRACK の 2 つの HTTP メソッドを使用して、Web サーバー接続をデバッグできます。HTTP TRACE 要求がサポートされている Web サーバーに送信されると、サーバーは応答して、HTTP ヘッダーを含む自分に渡されたデータをエコーします。Apache 構成でこれらのメソッドを無効にすることをお勧めします。

すべての仮想ホストの TRACE 方法を無効にするには、次の行を Apache HTTPD 構成ファイルに追加します。

```
TraceEnable off
```

特定の仮想ホストでこれらの方法を無効にするには、Apache 構成ファイルの仮想ホストごとに次の行を追加します。また、PEM は、次の行を Apache HTTPD 構成ファイルに追加します。

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

非特権ユーザアカウントから Web サーバーを実行する

root ユーザーとして Apache Web サービスを実行すると、セキュリティ問題が発生する可能性があります。一意の非特権ユーザーとして Web サーバーを実行することをお勧めします。これにより、セキュリティ侵害中に実行されている他のサービスの保護に役立ちます。

PEM は WSGI アプリケーションとして実行されます。実行中の WSGI アプリケーションを委任するには、`WSGIDaemonProcess` 指示子を使用して個別のデーモンプロセスを作成します。

Linux では、Apache Web サーバーは root ユーザーとして起動します。一方、PEM は pem ユーザーとしてデーモンプロセスを実行します。Windows では、 `WSGIDaemonProcess` 指示子と機能は使用できません。インストール中に、PEM HTTPD はサービスとしてインストールされます。このサービスのインストールを成功させるには、Administrators グループのメンバーである必要があります。

デフォルトでは、Apache サービスはシステムユーザー `LocalSystem` アカウントとして実行されます。

PEM Web サーバーのセキュリティ HTTP ヘッダーをカスタマイズする

PEM には、次のセキュリティ問題を修正するための別の構成ファイルが含まれています。`config.py` の構成ではなく、`config_local.py` の構成のみをオーバーライドすることをお勧めします。 `config_local.py` ファイルは、一部のシステムにのみ存在します。これを作成して、アプリケーションレベルの構成をオーバーライドする必要があります。PEM のアップグレード中に、`config_local.py` を上書きすることはできませんが、 `config.py` および `config_distro.py` の変更を上書きすることはできます。PEM をアンインストールした後、`config_local.py` を削除します。

デフォルトでは、 `config_local.py` は Linux では`/usr/edb/pem/web` に、 Windows`C:\ProgramFiles\edb\pem\server\share\web` にあります。

ホストヘッダーインジェクション攻撃

HTTP ホストヘッダー攻撃は、ホストヘッダー値を安全でない方法で処理する脆弱な Web サイトを悪用します。サーバーがホストヘッダーを暗黙的に信頼し、適切に検証またはエスケープできない場合、攻撃者はこの入力を使用して、サーバーサイドの動作を操作する有害なペイロードを注入できます。通常、Web アプリケーションは、セットアップ中に構成ファイルで指定しない限り、展開されているドメインを知りません。たとえば、現在のドメインを知る必要がある場合、ホストヘッダーからドメインを取得して、絶対 URL を生成できます。ホストヘッダーは、他のさまざまな脆弱性、最もよく知られている Web キャッシュポイズニングと SQL インジェクションを悪用するための潜在的なベクトルです。

X-Frame-Options

X-Frame-Options は、ブラウザーが<iframe>タグでページをレンダリングできるかどうかを示します。特にクリックジャッキングから保護します。PEM には、これらの攻撃を防ぐためのホスト検証 `X_FRAME_OPTIONS` オプションがあり、 `config_local.py` ファイルで構成できます。デフォルトは次のとおりです。

```
X_FRAME_OPTIONS = "SAMEORIGIN"
```

コンテンツセキュリティポリシー

Content-Security-Policy は、HTML5 標準の一部です。置き換える X-Frame-Options ヘッダーよりも幅広い保護を提供します。ウェブサイトの作成者がドメインをホワイトリストに登録できるように設計されています。作成者は、ホワイトリスト登録されたドメイン、およびページを埋め込むことができるドメインからスクリプト、スタイルシート、フォントなどのリソースをロードできます。

PEM には、攻撃を防止するためのホスト検証 `CONTENT_SECURITY_POLICY` オプションがあり、`config_local.py` ファイルで構成できます。デフォルトは次のとおりです。

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

厳密なトランスポートセキュリティ

Strict-Transport-Security HSTS 応答ヘッダーは、中間者攻撃を防止できます。このオプションを有効にすると、Web サイトまたは Web アプリケーションは、HTTPS のみを受け入れ、HTTP は受け入れないことをブラウザーに指示します。デフォルトは次のとおりです。

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

注釈: このパラメーターを追加すると、構成が変更された場合に問題が発生する可能性があります。したがって、PEM のインストールが完了してテストされた後にのみ追加することをお勧めします。

X-Content-Type-Options

X-Content-Type-Options 応答 HTTP ヘッダーはマーカーです。サーバーはマーカーを使用して、Content-Type ヘッダーでアドバタイズされた MIME タイプを変更および追跡できないことを示します。次は、MIME タイプスニッフィングをオプトアウトする方法、つまり、MIME タイプが意図的に構成されていることを示します。デフォルトは次のとおりです。

```
X_CONTENT_TYPE_OPTIONS = "nosniff"
```

X-XSS-プロテクション

クロスサイトスクリプティング XSS は、Web サーバーで最も一般的なアプリケーションレイヤーの脆弱性の 1 つです。XSS を使用すると、攻撃者は他のユーザーが表示する Web ページにクライアントサイドスクリプトを挿入できます。ヘッダーへの HTTP X-XSS-Protection 応答は、Internet Explorer、Chrome、および Safari の機能です。反射されたクロスサイトスクリプティング XSS 攻撃を検出すると、ページのロードを停止します。これらの保護は、サイトがインライン JavaScript 'unsafe-inline' の使用を無効にする強力なコンテンツセキュリティポリシーを実装している場合、最新のブラウザーでは必要ありません。ただし、これらの保護は、CSP をまだサポートしていない古い Web ブラウザーのユーザーに保護を提供できます。デフォルトは次のとおりです。

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

これを回避するには、次のオプションを Apache 構成ファイルに追加します。

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

構成ファイルの変更を適用するには、Web サーバーを再起動します。

デフォルトでは、PEM は/usr/edb/pem/web/config.py にあるアプリケーション構成ファイルで X-XSS-Protection to "1; mode=block" を設定します。

変更を適用するには、Apache サービスを再起動します。

config.py ファイルの詳細については、*Managing configuration settings* を参照してください。

Cookie のセキュリティ

Cookie は、構成データを保存するためにサーバーがブラウザーに送信するデータの小さなパケットです。ブラウザーはそれらと他のすべての要求を同じサーバーに送信するため、Cookie を保護する方法を知ることが重要です。config.py の複数の構成オプションにより、Cookie を安全にできます。これらは 3 つの最も重要なオプションです。

- SESSION_COOKIE_SECURE —このフラグは、暗号化されていない接続を介して Cookie が送信されるのを防ぎます。ブラウザーは、暗号化されたチャネルがないとサーバーへの要求に Cookie を追加できません。ブラウザーは、HTTPS などの接続にのみ Cookie を追加できます。デフォルトは次のとおりです。

```
SESSION_COOKIE_SECURE = True
```

- SESSION_COOKIE_HTTPONLY —デフォルトでは、JavaScript は Cookie のコンテンツを読み取ることができます。HTTPOnly フラグは、スクリプトが Cookie を読み取ることを防ぎます。代わりに、ブラウザーは HTTP または HTTPS 要求でのみ Cookie を使用します。ハッカーは XSS の脆弱性を悪用して Cookie の内容を学習することはできません。たとえば、sessionId Cookie は、クライアントサイドスクリプトで読み取る必要はありません。したがって、sessionId Cookie の HTTPOnly フラグを設定できます。デフォルトは次のとおりです。

```
SESSION_COOKIE_HTTPONLY = True
```

- ENHANCED_COOKIE_PROTECTION —このオプションを True に設定すると、IP アドレスとユーザーエージェントに基づいてトークンが生成されます。後続のすべての要求で、トークンは再計算され、最初の要求で計算されたものと比較されます。セッション Cookie が盗まれ、攻撃者が別の場所から使用した場合、生成されるトークンは異なります。その場合、拡張機能はセッションをクリアし、要求をブロックします。デフォルトは次のとおりです。

```
ENHANCED_COOKIE_PROTECTION = True
```

注釈: このオプションは、Kubernetes またはロードバランサーの背後などの動的 IP アドレスホスティング環境にサーバーが展開される場合、問題を発生させる可能性があります。このような場合、このオプションを `False` に設定します。

変更を適用するには、Apache サービスを再起動します。

`config.py` ファイルの詳細については、*Managing configuration settings* を参照してください。

PEM application security configurations

セッションタイムアウト

Web アプリケーションによるセッションの有効期限が不十分であると、他のセッションベースの攻撃の危険が増加します。攻撃者は、有効なセッション ID を再利用し、関連するセッションをハイジャックする時間が増えます。セッション間隔が短いほど、攻撃者が有効なセッション ID を使用する時間が短くなります。このセキュリティ問題を回避するには、Web アプリケーションの非アクティビティタイムアウトを低い値に設定することをお勧めします。

PEM では、ユーザーセッションのタイムアウト値を設定できます。Web コンソールで指定された期間にユーザーアクティビティがない場合、PEM は Web コンソールからユーザーをログアウトします。PEM 管理者は、非アクティブな時間を設定できます。この値は、アプリケーション全体に対するものであり、各ユーザーに対するものではありません。タイムアウト時間を構成するには、`<PEM_INSTALLATION_PATH>/web` ディレクトリにある `config_local.py` ファイルの `USER_INACTIVITY_TIMEOUT` パラメーターを変更します。デフォルトでは、この機能は無効になっています。

たとえば、アプリケーションが 15 分間非アクティブになった後にユーザーをログアウトするように指定するには、次のように設定します。

```
USER_INACTIVITY_TIMEOUT = 900
```

注釈: タイムアウト値は秒単位で指定します。

変更を適用するには、Apache サービスを再起動します。

`config.py` ファイルの詳細については、*Managing configuration settings* を参照してください。

RestAPI ヘッダーのカスタマイズ

要件を満たすように RestAPI トークンヘッダーをカスタマイズできます。デフォルト値は config.py ファイルでは公開されません。 config_local.py ファイルの次のヘッダーをカスタマイズします。

PEM_HEADER_SUBJECT_TOKEN_KEY

この構成オプションを使用すると、HTTP ヘッダー名を変更して、生成されたトークンを取得できます。デフォルトでは、トークンを作成する要求を送信すると、サーバー応答に X-Subject-Token ヘッダーが含まれます。このヘッダーには、新しく生成されたトークンの値が含まれています。ヘッダー名をカスタマイズする場合は、config_local.py ファイルを更新できます。

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

このコマンドは、次の出力を生成します。

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

PEM_HEADER_TOKEN_KEY

この構成オプションでは、HTTP 要求ヘッダー名を変更できます。このヘッダー名を使用して、トークンを PEM サーバーに送信できます。デフォルトでは、トークンを生成する要求を送信するときのトークンヘッダー名は X-Auth-Token です。RestAPI 要求ヘッダー名をカスタマイズする場合は、config_local.py ファイルを更新できます。

```
PEM_HEADER_TOKEN_KEY = Pem-Token
```

この設定では、トークンを送信できます。

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

PEM_TOKEN_EXPIRY

この構成オプションを使用すると、PEM RestAPI トークンの有効期限を生成後に変更できます。デフォルトでは、トークンの有効期限は 20 分 1200 秒に設定されています。トークンの有効期限を 10 分に変更する場合は、config_local.py ファイルを更新できます。

```
PEM_TOKEN_EXPIRY = 600
```

変更を適用するには、Apache サービスを再起動します。

PEM のロールベースのアクセス制御

ロールベースのアクセス制御 RBAC は、組織内のユーザーのロールに基づいてアプリケーションアクセスを制限し、アクセス制御の主要な方法の 1 つです。RBAC のロールは、ユーザーがアプリケーションに対して持っているアクセスレベルを参照します。ユーザーは、ジョブを実行するために必要な情報のみにアクセスできます。PEM のロールは継承可能であり、追加的なものです。つまり、PEM 管理者は、最下位レベルのロールをユーザーに付与し、ユーザーがジョブを実行するために必要なロールを付与する必要があります。例、SQL プロファイラーへのアクセスのみを付与するには

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

ロールの詳細については、 *PEM Roles* を参照してください。

SQL/Protect プラグイン

多くの場合、SQL インジェクション攻撃の防止はアプリケーション開発者の責任ですが、データベース管理者は潜在的な脅威をほとんどまたは全く制御できません。データベース管理者にとっての問題は、アプリケーションが適切に機能するにはデータにアクセスする必要があることです。

SQL/Protect は、データベース管理者が SQL インジェクション攻撃からデータベースを保護できるモジュールです。SQL/Protect は、標準のデータベースセキュリティポリシーに加えて、一般的な SQL インジェクションプロファイルの受信クエリを調べます。

攻撃者は、いくつかの異なる手法を使用して SQL インジェクション攻撃を実行する可能性があります。特定のシグネチャが各テクニックを特徴付けます。SQL/Protect は、未承認の関係、ユーティリティコマンド、SQL トートロジー、およびバインドされていない DML ステートメントのクエリを検査します。SQL/Protect は、潜在的に危険なクエリについて管理者に警告し、これらのクエリをブロックすることにより、データベース管理者に制御を戻します。

注釈: このプラグインは EDB Postgres Advanced Server サーバーでのみ動作するため、これは PEM データベースが EDB Postgres Advanced Server サーバーでホストされている場合にのみ役立ちます。

SQL Profiler プラグインの詳細については、*Upgrading SQL Profiler* を参照してください。

パスワード管理

PEM 管理ユーザーのためのセキュリティのヒントの 1 つは、PEM ログインパスワードを新しいものに定期的に変更することです。パスワードの変更

- 複数のアカウントの侵害を防止します
- 常時アクセスを防止
- 物理的に保護されていないシステムでの保存されたパスワードの使用を防止します
- キーストロークロガーによって取得されるアクセスを制限します

非 root ユーザーで pemAgent ジョブを実行する

ほとんどの場合、pemAgent は root ユーザーとしてインストールされ、root 権限を持つデーモンプロセスとして実行されます。デフォルトでは、PEM はスケジュールされたジョブ/タスクの実行を無効にします。PEM は、pemAgent 構成ファイルを変更することにより、スケジュールされたジョブを非 root ユーザーとして実行するためのサポートを提供します。

スケジュールされたジョブを非 root ユーザーとして実行するには、`agent.cfg` ファイルの `batch_script_user` パラメーターのエントリーを変更し、スクリプトを実行するユーザーを指定します。非 root ユーザーまたは root ユーザー ID を指定できます。ユーザーを指定しない場合、または指定されたユーザーが存在しない場合、スクリプトは実行されません。ファイルを変更した後、エージェントを再起動します。非 root ユーザーが `pemagent` を実行している場合、`batch_script_user` の値は無視され、`pemagent` の実行に使用されたものと同じ非 root ユーザーがスクリプトを実行します。

Windows システムでスクリプトを呼び出すには、`AllowBatchJobSteps` のレジストリエントリを `true` に設定し、PEM エージェントを再起動します。PEM レジストリエントリは次の場所にあります。

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

pemAgent および PEM バックエンドデータベースサーバー証明書の変更

デフォルトでは、PEM をインストールすると、インストーラーは pemAgent と PEM データベースサーバーの自己署名証明書を生成および使用します。PEMAgent は、PEM データベースサーバーに接続するときにこれらの証明書を使用します。pemAgent および PEM データベースサーバーに独自の SSL 証明書を使用するには、Managing certificates を参照してください。

注釈: PEM は、カスタムの場所への SSL CA 証明書の配置をサポートしていません。 *ca_certificate.crt* および `ca_key.key` の場所は変更しないでください。

2.2.5 Web server installation options

PEM サーバーのインストール中に、Apache Web サーバーのホスティング設定を指定できます。

別のホスト上の PEM サーバーと Apache Web サーバー

1. 両方のホストに PEM サーバーをインストールします。 *Installing the PEM server and PEM-HTTPD on separate hosts* を参照してください。
2. 最初のホストで **Database** オプションを選択して、PEM サーバーホストを構成します。
3. 2 番目のホストで **Web Services** オプションを選択して、Apache Web サーバーを構成します。

PEM サーバーの構成の詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈: 実稼働環境の場合、ベストプラクティスは PEM サーバーと Apache Web サーバーを別のホストに配置することです。 ## 同じホスト上の PEM サーバーと Apache Web サーバー

---

1. PEM サーバーをインストールします。 *Installing the PEM server and PEM-HTTPD on separate hosts* を参照してください。
2. 構成スクリプトを実行します。 **Web Services and Database** オプションを選択して、PEM サーバーと Apache Web サーバーを同じホストにインストールします。*Configuring the PEM server on Linux* を参照してください。

注釈: 非実稼働環境の場合、ベストプラクティスは PEM サーバーと Apache Web サーバーを同じホストに配置することです。

第3章　Installing and Upgrading

3.1 Installing Postgres Enterprise Manager server

リンクを選択して、該当するインストール手順にアクセスします。

3.1.1 Linux x86-64 (amd64)

Red Hat Enterprise Linux RHEL とその派生

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

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*、*Installing Postgres Enterprise Manager server on SLES 12 ppc64le*

Debian とデリバティブ

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

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

Red Hat Enterprise Linux RHEL とその派生

- *Installing Postgres Enterprise Manager server on RHEL 9 ppc64le* 、*Installing Postgres Enterprise Manager server on RHEL 8 ppc64le*

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le* 、*Installing Postgres Enterprise Manager server on SLES 12 ppc64le*

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

Debian とデリバティブ

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

3.1.4 Windows

- *Windows Server 2019*

Dependencies of the PEM Server and Agent on Linux

Linux の PEM サーバーおよびエージェントパッケージは、さまざまなシステムライブラリに依存しています。参考として、これらの依存関係について以下に詳しく説明します。

注釈: PEM エージェントは常に PEM サーバーとともにインストールされるため、すべての依存関係は PEM サーバーデータベースまたは Web アプリケーションがインストールされるホストに存在する必要があります。通常、PEM は、特定のプラットフォームとアーキテクチャのベンダーリポジトリから入手可能な各依存関係の最新バージョンに対してビルドされます。場合によっては、PEM には、ベンダーリポジトリで利用可能なものよりも新しいバージョンのライブラリが必要です。これらの場合、パッケージの新しいバージョンの接頭辞 `edb-` は EDB のリポジトリから取得されます。

注釈: この情報は参考のために提供されています。ベンダーリポジトリからのパッケージは、EDB によってサポートまたはパッチされていません。 これらのパッケージの詳細については、オペレーティングシステムのドキュメントまたはサポートプロバイダーを参照してください。

これらの依存関係は頻繁に更新されるため、以下の表は PEM の最新パッチリリースでのみ有効です。

PEM サーバーには、Python 3 と mod_wsgi Apache HTTPD の Python モジュールが必要です。

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

OpenSSL

PEM サーバーとエージェントには OpenSSL が必要です。

Platform	Architecture	package-version
RHEL 7	x86_64	*openssl-1.0.2k*
	ppc64le	*openssl-1.0.2k*
RHEL 8	x86_64	*openssl-1.1.1k*
	ppc64le	*openssl-1.1.1k*
	s390x	*openssl-1.1.1k*
RHEL 9	x86_64	*openssl-3.0.7*
	ppc64le	*openssl-3.0.7*
	s390x	*openssl-3.0.7*
SLES 12	x86_64	*openssl-1.0.2p*
	ppc64le	*openssl-1.0.2p*
	s390x	*openssl-1.0.2p*
SLES 15	x86_64	*openssl-1.1.1l*
	ppc64le	*openssl-1.1.1l*
	s390x	*openssl-1.1.1l*
Ubuntu 20	amd64	*openssl-1.1.1f*
Ubuntu 22	amd64	*openssl-3.0.2*
Debian 10	amd64	*openssl-1.1.1n*
Debian 11	amd64	*openssl-1.1.1w*
Debian 12	amd64/arm64	*openssl-3.0.11*

Libcurl

PEM エージェントには libcurl が必要です。

Platform	Architecture	package-version
RHEL 7	x86_64	*libcurl-pem-8.4.0*
	ppc64le	*libcurl-pem-8.4.0*
RHEL 8	x86_64	*libcurl-pem-8.4.0*
	ppc64le	*curl-7.61.1*
	s390x	*curl-7.61.1*
RHEL 9	x86_64	*curl-7.76.1*
	ppc64le	*curl-7.76.1*
	s390x	*curl-7.76.1*
SLES 12	x86_64	*curl-8.0.1*
	ppc64le	*curl-8.0.1*
	s390x	*curl-8.0.1*
SLES 15	x86_64	*curl-8.0.1*

次のページに続く

表 3 – 前のページからの続き

Platform	Architecture	package-version
	ppc64le	*curl-8.0.1*
	s390x	*curl-8.0.1*
Ubuntu 20	amd64	*libcurl4-7.68.0*
Ubuntu 22	amd64	*libcurl4-7.81.0*
Debian 10	amd64	*libcurl4-7.64.0*
Debian 11	amd64	*libcurl4-7.74.0*
Debian 12	amd64/arm64	*libcurl4-7.88.1*

SNMP++

PEM エージェントには SNMP++が必要です。

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

Boost ライブラリ

PEM エージェントには、Boost ライブラリが必要です。

Platform	Architecture	package-version
RHEL 7	x86_64	*boost169-system-1.69.0*
	ppc64le	*None boost package*
RHEL 8	x86_64	*boost169-system-1.69.0*
	ppc64le	*boost-system-1.66.0*
	s390x	*boost-system-1.66.0*
RHEL 9	x86_64	*boost-system-1.75.0*
	ppc64le	*boost-system-1.75.0*
	s390x	*boost-system-1.75.0*
SLES 12	x86_64	*libboost_system1_54_0-1.54.0*
	ppc64le	*libboost_system1_54_0-1.54.0*
	s390x	*libboost_system1_54_0-1.54.0*
SLES 15	x86_64	*libboost_regex1_66_1-1.66.0*
	ppc64le	*libboost_regex1_66_1-1.66.0*
	s390x	*libboost_regex1_66_1-1.66.0*
Ubuntu 20	amd64	*libboost-system1.71.0-1.71.0*
Ubuntu 22	amd64	*libboost-system1.74.0-1.74.0*
Debian 10	amd64	*libboost-system1.67.0-1.67.0*
Debian 11	amd64	*libboost-system1.74.0-1.74.0*
Debian 12	amd64/arm64	*libboost-system1.74.0-1.74.0*

Installing Postgres Enterprise Manager server on Linux x86 (amd64)

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Red Hat Enterprise Linux RHEL とその派生

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

- *Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*
- *Installing Postgres Enterprise Manager server on SLES 12 ppc64le*

Debian とデリバティブ

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

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

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、**メール通知を設定する** PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. `pg_hba.conf` ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベー

スが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。

- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo dnf upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

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

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo dnf upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on AlmaLinux 9 or Rocky

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. `pg_hba.conf` ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo dnf upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on AlmaLinux 8 or Rocky

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. `pg_hba.conf` ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo dnf upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on SLES 15 x86_64

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します host all superusername 127.0.0.1/32 scram-sha-256 などの md5 または scram-sha-256 認証を使用した host 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 hstore contrib モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように sslutils 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo zypper update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on SLES 12 x86_64

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します host all superusername 127.0.0.1/32 scram-sha-256 などの md5 または scram-sha-256 認証を使用した host 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 hstore contrib モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように sslutils 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo zypper update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on Ubuntu 22.04 x86_64

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します host all superusername 127.0.0.1/32 scram-sha-256 などの md5 または scram-sha-256 認証を使用した host 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

注釈: Ubuntu 20 では、証明書を受け入れるための要件が変更されました。

- sslutils の古いバージョンを使用するマシンに PEM エージェントをインストールする場合は、sslutils を 1.3 にアップグレードする必要があります。バージョン 1.3 には、4096 ビット RSA キーと sha256 署名アルゴリズムのサポートが追加されています。- sslutils を 1.3 にアップグレードしない場合、PEM エージェントは PEM バックエンドデータベースサーバーへの接続に失敗し、エラー「ca md too promote.」ログに記録される場合があります。

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

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo apt-get update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on Ubuntu 20.04 x86_64

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

注釈: Ubuntu 20 では、証明書を受け入れるための要件が変更されました。

- sslutils の古いバージョンを使用するマシンに PEM エージェントをインストールする場合は、sslutils を 1.3 にアップグレードする必要があります。バージョン 1.3 には、4096 ビット RSA キーと sha256 署名アルゴリズムのサポートが追加されています。- sslutils を 1.3 にアップグレードしない場合、PEM エージェントは PEM バックエンドデータベースサーバーへの接続に失敗し、エラー「ca md too promote.」ログに記録される場合があります。

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

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo apt-get update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on Debian 12 x86_64

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. `pg_hba.conf` ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo apt-get upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on Debian 11 x86_64

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. `pg_hba.conf` ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

注釈: Debian 10 は、証明書を受け入れるための要件を変更しました。

- sslutils の古いバージョンを使用するマシンに PEM エージェントをインストールする場合は、sslutils を 1.3 にアップグレードする必要があります。バージョン 1.3 には、4096 ビット RSA キーと sha256 署名アルゴリズムのサポートが追加されています。- sslutils を 1.3 にアップグレードしない場合、PEM エージェントは PEM バックエンドデータベースサーバーへの接続に失敗し、エラー「ca md too promote」がログに記録される場合があります。

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

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo apt-get update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

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

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Red Hat Enterprise LinuxRHEL

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

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager server on SLES 15 ppc64le*
- *Installing Postgres Enterprise Manager server on SLES 12 ppc64le*

Installing Postgres Enterprise Manager server on RHEL 9 ppc64le

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::

2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo dnf upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on RHEL 8 ppc64le

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo dnf upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on SLES 15 ppc64le

---

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo zypper update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on SLES 12 ppc64le

---

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. pg_hba.conf ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を pg_hba.conf に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように `sslutils` 、および必要に応じて `hstore` モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

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

firewall-cmd --reload
```

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo zypper update
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on Linux AArch64 (ARM64)

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Debian とデリバティブ

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

Installing Postgres Enterprise Manager server on Debian 12 arm64

1 つのサーバーに PEM をインストールするか、Web アプリケーションサーバーとバックエンドデータベースを 2 つの別のサーバーにインストールできます。PEM インストールのためにサーバーを準備する必要があります。

前提条件を満たし、次の手順で説明するインストール手順を完了した後、メール通知を設定する PEM を行う必要があります。2 つのサーバーを使用している場合は、両方のサーバーに PEM をインストールして構成します。

前提条件

インストールプロセスを開始する前に

1. バックエンドデータベースとして使用する PEM の をインストールします。

このインスタンスは、PEM Web アプリケーションに使用するのと同じサーバーにインストールすることも、別のサーバーにインストールすることもできます。次の手順で詳細に説明するように構成されている場合、既存の Postgres インスタンスを使用することもできます。

2. `pg_hba.conf` ファイルを更新することにより、Postgres バックエンドデータベースで認証を構成します。

構成する前に、次の変更を手動で加えます。*Managing configuration settings* の実行中に追加の変更が必要です。

- PEM に必要な関係を作成するために、PEM 構成スクリプトは、パスワード認証を使用して選択したスーパーユーザーとして Postgres バックエンドデータベースに接続します。これには、選択したスーパーユーザーにパスワードを使用した認証を許可する必要があります。このユーザーは、構成スクリプトを実行する任意の場所から接続できる必要があります。実際、これは、バックエンドデータベースが存在するサーバーと PEM Web アプリケーションがインストールされるサーバーが異なる場合を意味します。
- 選択したスーパーユーザーがパスワード認証を使用して接続できるようにするには、次の行を `pg_hba.conf` に追加します `host all superusername 127.0.0.1/32 scram-sha-256` などの `md5` または `scram-sha-256` 認証を使用した `host` 接続を許可します。

注釈: EDB Postgres Advanced Server を使用している場合、 を参照してください。

PostgreSQL を使用している場合、 Client Authentication を参照してください。 .. ::
2_blank_1_3. sslutils 拡張機能が Postgres サーバーにインストールされていることを確認します。

RHEL/AlmaLinux/Rocky Linux または SLES で PostgreSQL または EDB Postgres Extended Server を使用している場合は、 `hstore contrib` モジュールをインストールする必要があります。

- EDB Postgres Advanced Server を使用している場合、次のように `sslutils` 拡張機能をインストールできます。 <x> は、 EDB Postgres Advanced サーバーバージョンです。

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

- PostgreSQL を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は PostgreSQL バージョンです。

edb_notranlate_1 - EDB Postgres Extended Server を使用している場合、次のように sslutils 、および必要に応じて hstore モジュールをインストールできます。<x> は EDB Postgres Extended Server バージョンです。

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

4. ファイアウォールを使用している場合は、PEM Web アプリケーションが配置されるサーバーでポート 8443 へのアクセスを許可します。

5. Postgres Enterprise Manager が依存するコンポーネントがすべてのサーバーで最新の状態であることを確認します。これは、以下に示すようにパッケージマネージャーを使用してシステム全体を更新することにより実行できます。個々のパッケージを更新する場合、依存関係の完全なリストが *Dependencies of the PEM Server and Agent on Linux* で提供されています。

```
sudo apt-get upgrade
```

パッケージをインストールする

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

初期構成

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

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

詳細については、 *Configuring the PEM server on Linux* を参照してください。

注釈:

- オペレーティングシステムユーザー pem は、PEM サーバーのインストール中に作成されます。 PEM サーバー Web アプリケーションは、Apache HTTPD で実行される WSGI アプリケーションです。 pem アプリケーションデータとセッションは、このユーザーのホームディレクトリに保存されます。

サポートされているロケール

現在、Postgres Enterprise Manager サーバーと Web インターフェイスは、English(US) en_US ロケールと言語区切り文字としてのピリオド. の使用をサポートしています。代替ロケールまたはピリオド以外の区切り文字を使用すると、エラーが発生する場合があります。

Installing Postgres Enterprise Manager server on Windows

Windows では、PEM サーバーグラフィカルインストーラーは、PEM サーバー、PEM エージェント、および選択したブラウザーで PEM Web インターフェイスに接続するために必要なソフトウェアをインストールおよび構成します。

各 PEM インストールの中心となるのは PEM サーバーです。実稼働環境では、サーバーは通常専用マシンであり、多数の Postgres サーバーまたは少数のビジーサーバーを監視します。

PEM サーバーバックエンドデータベースは、PostgreSQL または EDB Postgres Advanced Server データベースサーバーの EDB ディストリビューション、または別のソースからインストールされた既存の Postgres サーバーです。Postgres バックエンドデータベースサーバーはバージョン 11 以降である必要があり、PEM サーバーがリポジトリとして使用する pem という名前のデータベースが含まれています。

- 既存の Postgres サーバーを使用して PEM サーバーをホストする場合、PEM サーバーインストーラーは Postgres ホストに pem データベースを作成できます。既存のサーバーを使用する場合、ソフトウェア前提条件を手動で満たす必要があります。

既存の Postgres サーバーを使用して PEM サーバーバックエンドデータベースをホストする方法の詳細については、*Installing the PEM server on an existing Postgres servers* を参照してください。

- Postgres の既存のインストールを PEM サーバーホストとして使用したくない場合、PEM サーバーインストーラーは次のことを行うことができます。
  - PostgreSQL のインストール
  - サーバーホストのソフトウェア前提条件を満たす s
  - pem データベースを含むインスタンス PostgreSQL データベースクラスターを作成します。これは、最も単純な PEM サーバーのインストールオプションです
- PEM-HTTPD は、PEM サーバーインストーラーまたは StackBuilder ユーティリティを介して Postgres インストールで利用できるようになります。PEM-HTTPD がホストに既にインストールされている場合、PEM サーバーインストーラーは、必要に応じて既存のインストールを確認および更新します。PEM サーバーホストに既存の PEM-HTTPD インストールが含まれていない場合、PEM サーバーインストーラーはそれを追加します。
- PEM サーバーをインストールする前に、PostgreSQL と PEM-HTTPD を同じホストで実行するか、別のホストで実行するかを決定する必要があります。PostgreSQL データベースサーバーと PEM-HTTPD を別のホストで実行する場合は、PEM サーバーインストーラーを各ホストで 1 回ずつ、2 回実行する必要があります。Installing the PEM server and PEM-HTTPD on separate hosts を参照してください。
- 同じホストへの PEM サーバーと PEM-HTTPD のインストールの詳細については、Installing the PEM server and PEM-HTTPD on the same host を参照してください。

- スタンドアロン PEM エージェントのインストールと構成の詳細については、*Installing Postgres Enterprise Manager agent on Windows* を参照してください。
- 言語パックインストーラーには、 EDB Postgres Advanced Server および EDB PostgreSQL データベースインストーラーで使用できるサポートされている言語が含まれています。言語パックインストーラーを使用すると、サードパーティベンダーからサポートソフトウェアをインストールせずに、Perl、TCL / TK、および Python をインストールできます。言語パックのインストールと使用の詳細については、「を参照してください。
- PEM エージェントのインストールまたは構成のトラブルシューティングについては、*Troubleshooting agent issues* を参照してください。

PEM サーバーインストーラーは、PEM Web インターフェイスを使用してサーバーにアクセスするために必要なソフトウェアもインストールします。Web インターフェイスには、サポートされているバージョンのブラウザーでアクセスできます。Web インターフェイスを使用して、次のことができます。

- 監視対象サーバーにあるオブジェクトに関する情報を確認する
- 監視対象サーバーにあるデータベースとデータベースオブジェクトの管理
- PEM サーバーによって収集された統計情報を確認します

注釈: SSL 証明書を使用している場合、すべての SSL 証明書がバックエンドデータベースサーバーのデータディレクトリにあることを確認します。証明書がデータディレクトリにない場合、PEM サーバーの構成中にデータディレクトリを検索するため、PEM サーバーの構成が失敗する可能性があります。

Installing the PEM server and PEM-HTTPD on separate hosts

PEM サーバーバックエンドデータベースと PEM-HTTPD に別のホストを使用するには

1. `pem` データベースを含む Postgres サーバーのホストで PEM サーバーインストーラーを起動します。インストール中に、[Advanced Options] ダイアログボックスで [**Server**] オプションを選択し、Postgres サーバーの接続情報を提供します。
2. PEM サーバーおよび `pem` データベースが存在する Postgres インストールの `pg_hba.conf` ファイルを変更し、PEM-HTTPD サーバーのホストからの接続を許可します。
3. PEM-HTTPD サーバーのホストで PEM サーバーインストーラーを起動し、[Installation Type] ダイアログボックスで [**Web Services**] オプションを選択します。

インストールを開始するには

1. Windows システムでは、PEM サーバーインストーラーを起動します。インストーラーを右クリックし、[**Run as Administrator**] を選択します。インストーラーには、 Welcome 画面が表示されます。**Next** を選択します。
2. 適切なラジオボタンを選択して契約に同意する前に、ライセンス契約を確認してください。**Next** を選択します。
3. Installation Directory ダイアログボックスのフィールドを使用して、PEM サーバーのディレクトリを指定し、インストール中に Advanced Options ダイアログボックスを開きます。

- デフォルトでは、PEM サーバーは Windows の `C:\Program Files\edb\pem` にインストールされます。デフォルトの場所をそのまま使用するか、**インストールディレクトリ**フィールドを使用してブラウザダイアログボックスを開き、PEM サーバーをインストールするディレクトリを選択します。
- 別のホストに PEM サーバーと PEM-HTTPD をインストールするには、詳細オプションを使用しますダイアログボックスで、インストールタイプ **Web Services** または **Database** を指定します。インストールプロセスに Advanced Options ダイアログボックスを含めるには、 **Show advanced options** を選択します。
- **Next** を選択します。

4. Advanced Options ダイアログボックスを使用して、インストールするコンポーネントを指定します。
   - **Web Services and Database** を選択して、Postgres サーバーと PEM-HTTPD の両方が現在のホストに存在することを示します。**Web Services と Database** を選択した場合、PEM サーバーインストーラーでは、PEM-HTTPD インストールをチェックする前に PEM サーバーに使用する Postgres サーバーを指定できます。
   - PEM-HTTPD をインストールするには **Web Services** を選択します別のホストにある Postgres データベースサーバーを使用して PEM サーバーと `pem` データベースをホストします。

注釈: **Web Services**オプションを使用して PEM-HTTPD をインストールする前に、PEM サーバーと `pem` バックエンドデータベースのホストで PEM サーバーのインストールプロセスを完了する必要があります。 Advanced Options ダイアログボックスで**Server**を選択し、PEM サーバー上の `pg_hba.conf` ファイルの接続プロパティを変更します。

このオプションは、*Web* **サービスのインストール** で説明されているインストール手順を呼び出します。

- **Database** を選択して、既存の Postgres サーバーバージョン 11 以降を使用するか、PEM サーバーインストーラーで配布されるデータベースサーバーのみをインストールします。このオプションは、**データベースホストの指定** で説明されているインストール手順を呼び出します。

1. インストールオプションを選択したら、[**Next**] を選択します。

データベースホストの指定

1. Advanced Options ダイアログボックスで **Server** を選択して、PEM サーバーのバックエンドデータベース `pem` という名前が存在するホストの接続情報を指定します。 **Next** を選択します。
2. [Database Server Selection] ダイアログボックスのリストを使用して、PEM サーバーバックエンドデータベースのホストを選択します。
   - 現在のホストに存在する既存の Postgres インストールからホストを選択します。

注釈: *sslutils* パッケージをインストールに追加する必要がある場合があります。*2_blank_4_*- **PostgreSQL <x>**を選択して、PEM サーバーインストーラーで配布される Postgres サーバーをインストールします。*<x>* は、PostgreSQL データベースサーバーのバージョンです。 PEM サーバーインストーラーにバンドルされた PostgreSQL のバージョンを使用することにした場合は、EnterpriseDB ワンクリックの PostgreSQL

インストーラーが開き、インストールをガイドします。*1_blank_4_-* **Other Database Server**を選択して、次の Postgres サーバーの接続情報を指定します。 EnterpriseDB からワンクリックのグラフィカルインストーラーを使用してインストールされていません。 PEM サーバーデータベースホストのソフトウェア前提条件については、 *Postgres* **サーバーの準備** .`2_blank_4_-` を参照してください。 ** Next**を選択します。

PEM サーバーが既存の Postgres サーバーにある場合、[Database Server Installation Details] ダイアログボックスが開きます。

1. [Database Server Installation Details] ダイアログボックスで必要な情報は異なる場合があります。PEM サーバーインストーラーは、選択したインストールに関する見つからない情報のみを提供するように求めます。
   - **User** フィールドに Postgres データベースのスーパーユーザーの名前を指定します。
   - そのユーザーに関連付けられたパスワードを指定します **Password** フィールド.
   - **Next** を選択します。
2. プロンプトが表示されたら、PEM エージェントを実行するサービスアカウントのシステムパスワードを入力します。**Next** を選択します。
3. Network Details ダイアログボックスを使用して、PEM エージェントがサーバークライアントサイドアドレスに接続する CIDR スタイルのネットワークアドレスを指定します。指定されたアドレスがサーバーの `pg_hba.conf` ファイルに追加されます。

注釈: PostgreSQL サーバー上の `pg_hba.conf` ファイルにエントリを手動で追加することにより、より多くのネットワークアドレスを指定できます。

デフォルトローカルホスト指定を受け入れるか、 **ネットワークアドレス**範囲を指定します。 **Next** を選択します。

PEM サーバーインストーラーは、サーバーが存在するホストに PEM エージェントをインストールして、サーバーを監視し、アラート処理とガベージコレクションサービスを提供します。証明書は、**エージェント証明書パス**フィールドで指定された場所にもインストールされます。

1. PEM エージェントの代替説明または代替エージェント証明書パスを入力するか、デフォルトを受け入れます。**Next** を選択します。
2. ウィザードで PEM サーバーをインストールする準備が整いました。 **Next** を選択して、インストールを続行します。

インストールプロセス中に、インストーラーはシステムにファイルをコピーし、PEM サーバーのバックエンドデータベースをセットアップします。確認は、`pem` データベースが作成および構成されたことを示します。

1. **OK** を選択します

PEM サーバーのインストールのデータベース部分が完了したら、別のホストで PEM サーバーを呼び出して、PEM-HTTPD をインストールまたはアップグレードできます。

Web サービスのインストール

Advanced Options ダイアログボックスで **Web Services** を選択して、現在のホストに PEM-HTTPD をインストールするか、既存の PEM-HTTPD インストールを更新します。

注釈: 現在のホストは、PEM バッキングデータベースのホストではない可能性があります。

このオプションを選択する前に、以下を行う必要があります。

- ホストシステムに PEM サーバーインストーラーをインストールします。この際、PEM サーバーのバッキングデータベースを指定します。
- PEM サーバーデータベースホスト上の `pg_hba.conf` ファイルを変更して、PEM-HTTPD ホストからの接続を許可します。
- データベースサーバーを再起動します。

1. **Web Services** を選択し、 **Next** を選択します。PEM サーバーインストーラーは、現在のホストで既存の PEM-HTTPD および言語パックのインストールを確認します。
2. インストーラーがコンポーネントを見つけられない場合、インストーラーは、1 つ以上の依存関係が欠落していることを通知します。**Next** を選択して言語パックをインストールします。言語パックをインストールすると、インストーラーは PEM-HTTPD セットアップウィザードを起動します。
3. PEM-HTTPD Setup ウィザードの画面上の指示に従います。ウィザードがインストールを完了したら、[**Finish**] を選択して [Database Server Installation Details] ダイアログボックスを開きます。
4. Database Server Installation Details ダイアログボックスを使用して、PEM サーバーインストールをホストしている Postgres インストールの接続情報を提供します。
   - **Host** フィールドに PEM サーバーホストの名前または IP アドレスを入力します。
   - ポートを入力します Postgres サーバーが **ポート**フィールドにリッスンしている番号。
   - **User** フィールドに Postgres データベーススーパーユーザーの名前を入力します。
   - **Password* に Postgres スーパーユーザーに関連付けられたパスワードを入力します。*フィールド。
5. **Next** を選択します。

PEM サーバーのインストールが完了する前に、インストーラーはデータベースホストに接続します。インストールを続行するには、PEM データベースホスト上の `pg_hba.conf` ファイルを httpd サーバーのホストからの接続を受け入れるように構成し、ファイアウォールが接続を許可する必要があります。PEM サーバーインストーラーは、PEM サーバーのインストールを完了し、PEM-HTTPD サーバーのホストに存在する必要がある項目のみを追加します。

Installing the PEM server and PEM-HTTPD on the same host

最も簡単な PEM サーバーインストール構成は、PEM バックエンドデータベースサーバー PEM サーバーインストーラーでインストールされた PostgreSQL データベースでホストされ、同じホストに存在する PEM-HTTPD サービスで構成されます。この構成では、PEM サーバーインストーラーは、バックエンドホストがサービスを登録するための前提条件ソフトウェアを提供し Windows。

1. PEM サーバーインストーラーを起動します。Windows システムでは、インストーラーアイコンを右クリックし、 **Run as Administrator** を選択します。インストーラーには、 Welcome 画面が表示されます。 **Next** を選択します。

2. 適切なラジオボタンを選択して契約に同意する前に、ライセンス契約を確認してください。**Next** を選択します。

3. Installation Directory ダイアログボックスを使用して、PEM サーバーの場所を指定します。

- デフォルトでは、PEM サーバーは Windows の `C:\Program Files\edb\pem` にインストールされます。デフォルトの場所をそのまま使用するか、**インストールディレクトリ**ボタンを使用してダイアログボックスを開き、PEM サーバーをインストールするディレクトリを選択します。

- **Show advanced options** チェックボックスを使用して、 Advanced Options ダイアログボックスを開きますインストール中に。Postgres データベースサーバと PEM-HTTPD を別のホストにインストールする場合、または既存の Postgres サーバーインストールに PEM サーバーを存在させたい場合に、Advanced Options ダイアログボックスを使用します。

- インストーラーと PEM-HTTPD を使用して PostgreSQL サーバーパッケージをインストールするには同じホストで、**Show advanced options** をクリアし、 **Next** を選択します。

PEM サーバーインストーラーは、PEM-HTTPD、言語パック、および PostgreSQL 13 のインストール前チェックを実行します。インストーラーがこれらのパッケージを見つけられない場合、[dependency Missing] ダイアログボックスで通知します。

注釈: デフォルトでは、EDB Language Pack は `C:edblanguagepackv1` にインストールされます。

---

1. 依存関係が欠落している場合、PEM サーバーインストーラーは、各インストールウィザードを起動します。各パッケージは、ウィザードの画面上の指示に従ってください。

不足している依存関係をインストールすると、[Database Server Installation Details] ダイアログボックスが表示されてインストールプロセスが続行されます。 [Database Server Installation Details] ダイアログボックスで提供される情報により、インストーラーは PostgreSQL サーバーに接続できます。

1. データベースのスーパーユーザーのユーザー名とパスワードを入力します。要求された情報を入力したら、[**Next**] を選択します。

2. Postgres データベーススーパーユーザーの名前とパスワードを入力すると、PEM エージェントを実行するユーザーアカウントのパスワードの入力を求められる場合があります。プロンプトが表示されたら、パスワードを入力し、**Next** を選択します。

3. Network Details ダイアログボックスを使用して、PEM エージェントがサーバー**クライアントサイド**アドレスに接続する CIDR スタイルのネットワークアドレスを指定します。

ネットワークホストのアドレスまたはネットワークアドレスレンジを指定できます。たとえば、アドレス 192.168.10.23 、192.168.10.76 、および 192.168.10.184 を持つデータベースサーバーを監視する場合は、そのネットワーク内のホストとの接続を許可するように 192.168.10.0/24 と入力します。

指定されたアドレスがサーバーの pg_hba.conf ファイルに追加されます。PostgreSQL サーバー上の pg_hba.conf ファイルにエントリを手動で追加することにより、より多くのネットワークアドレスを指定できます。最初のエントリをテンプレートとして使用します。

ネットワークアドレスを追加したら、[**Next**] を選択します。

PEM サーバーインストーラーは、サーバーが存在するホストに PEM エージェントをインストールして、サーバーを監視し、アラート処理とガベージコレクションサービスを提供します。証明書は、**エージェント証明書パス**フィールドで指定された場所にもインストールされます。

1. 代替説明を入力するか、PEM エージェントの代替エージェント証明書パスを選択するか、デフォルトを受け入れます。**Next** を選択します。

2. ウィザードで PEM サーバーをインストールする準備が整いました。 **Next** を選択して、インストールを続行します。

インストールプロセス中に、インストーラーはファイルをシステムにコピーし、PEM の実行に必要なデータベースと Web サービスをセットアップします。インストールが完了すると、確認が次のことを示します。- Web サービスが構成されました。 - Web サービスはポート 8443 でリッスンしています。 - pem データベースが作成および構成されました。

1. **OK** を選択します。

Installing the PEM server on an existing Postgres servers

既存の Postgres サーバーバージョン 11 以降を使用して、PEM サーバーと pem データベースをホストできます。Postgres インストーラーと前提条件ソフトウェア拡張機能は、 EnterpriseDB website で無料で入手できます。

PEM サーバーインストール用に既存の Postgres サーバーを構成できます。

注釈: これらの指示はガイドラインです。 Postgres インストールの構成に必要な実際の手順は、Postgres サーバーの構成によって異なります。

Postgres の次のバージョンは、sslutils 拡張機能とサービススクリプトを含むように事前構成されています。次の Postgres バージョンを PEM バックエンドデータベースサーバーとして使用するために、追加の準備は必要ありません。

- PostgreSQL 11 以降 PEM Server インストーラーにバンドルされている
- EDB Postgres Advanced Server 11 以降

Postgres サーバーの準備

既存の Postgres サーバーに PEM サーバーをインストールする前に、次のことを行う必要があります。

- Postgres サーバーに sslutils 拡張機能のインストールが含まれていることを確認します。詳細については、*sslutils* 拡張機能のインストール を参照してください。
- サーバーを Windows サービスマネージャーに登録します。詳細については、サービスの登録 を参照してください。

サーバーの準備ができたら、PEM サーバーインストーラーを使用して、既存の Postgres サーバーに PEM をインストールできます。

sslutils 拡張機能のインストール

PEM サーバーが存在する Postgres サーバーには、sslutils 拡張機能が含まれている必要があります。sslutils パッケージは、 EDB website から無料でダウンロードできます。

Web ページが開いたら、SRC- SSL Utils 1.3 パッケージのリンクを選択します。ダウンロードが完了したら、ファイルを抽出し、Postgres インストールディレクトリにコピーします。

注釈: 次の Postgres インストールを使用する場合、*sslutils* 拡張機能を手動で追加する必要はありません。

- PostgreSQL 10 以降 PEM サーバーインストーラーと共に配布
- Advanced Server 10 以降

バックエンド Postgres インストールのコンパイルに使用したものと同じコンパイラーで sslutils をビルドします。PostgreSQL ワンクリックインストーラー EnterpriseDB からまたは EDB Postgres Advanced Server インストーラーを使用して Windows プラットフォームにインストールされたバックエンド Postgres データベースを使用している場合、Visual Studio を使用して sslutils をビルドする必要があります。

インストールプロセスの具体的な詳細はプラットフォームとコンパイラーによって異なりますが、基本的な手順は同じです。あなたは以下を行う必要があります。

1. sslutils パッケージを Postgres インストールディレクトリにコピーします。
2. 適切なコンパイラーのコマンドラインを開き、 sslutils ディレクトリに移動します。
3. 次のコマンドを使用して sslutils をビルドします。

```
SET USE_PGXS=1

SET GETTEXTPATH=<path_to_gettext>

SET OPENSSLPATH=<path_to_openssl>

SET PGPATH=<path_to_pg_installation_dir>
```

(次のページに続く)

(前のページからの続き)

```
SET ARCH=x86

REM Set ARCH x64 for 64 bit

msbuild sslutils.proj /p:Configuration=Release
```

そこで

path_to_gettext は、GETTEXT ライブラリとヘッダーファイルの場所を指定します。

path_to_openssl は、openssl ライブラリとヘッダーファイルの場所を指定します。

path_to_pg_installation_dir は、Postgres をインストールする場所を指定します。

4. コンパイルした sslutils ファイルをインストールの適切なディレクトリにコピーします。sslutils ディレクトリには、次のファイルが含まれています。

```
sslutils--1.3.sql

sslutils--unpackaged--1.3.sql

sslutils--pemagent.sql.in

sslutils.dll
```

.dll ライブラリと.sql ファイルを適切な場所にコピーします。

```
COPY sslutils *.sql*  "%PGPATH%\share\extension\"

COPY sslutils.dll "%PGPATH%\lib\"
```

サービスの登録

EnterpriseDB のインストーラーを使用して PostgreSQL または EDB Postgres Advanced Server データベースをインストールすると、インストーラーはサービスを登録します。

Windows を使用して PEM バックエンドデータベースをホストしている場合、Postgres サーバーの名前を Windows サービスマネージャーに登録する必要があります。EnterpriseDB インストーラーを使用して作成された Postgres サーバーを使用している場合、サービスは自動的に登録されます。インストールを手動でビルドする場合は、Postgres pg_ctl コマンドの register 句を使用してサービスを登録できます。

```
pg_ctl register [-N <service_name>] [-U <user_name>]
 | [-P <password>] [-D <data_directory>]
```

そこで

- service name は、Postgres クラスターの名前を指定します。

- user_name は、Postgres インストールディレクトリにアクセスし、Postgres サービスを開始するための十分な特権を持つオペレーティングシステムユーザーの名前を指定します。
- password は、ユーザーに関連付けられたオペレーティングシステムパスワードを指定します。
- data_directory は、Postgres データディレクトリの場所を指定します。

pg_ctl コマンドと使用可能なコマンドオプションの使用の詳細については、 Postgres core documentation を参照してください。

PEM サーバーインストーラーの呼び出し

既存の Postgres サーバーを準備した後、PEM サーバーインストーラーを起動します。管理者権限を想定し、インストーラーが含まれるディレクトリに移動します。次に、インストーラーを呼び出します。

```
pem_server-7.<x>.<x>-<x>-<platform>
```

<x> は PEM のメジャーバージョンとマイナーバージョンであり、<platform> はプラットフォームです。

次に、インストールを開始します。

1. インストーラーには、 Welcome 画面が表示されます。 **Next** を選択します。
2. 適切なラジオボタンを選択して契約に同意する前に、ライセンス契約を確認してください。**Next** を選択します。
3. Installation Directory ダイアログボックスを使用して PEM サーバーの場所を指定し、Advanced Options ダイアログボックスにアクセスします。
   - **Installation Directory** フィールドを使用してブラウザダイアログを開き、PEM サーバーをインストールするディレクトリを選択します。
   - 既存のサーバーに PEM サーバーをインストールする場合は、 **Show advanced options** を選択して、インストールプロセスに Advanced Options ダイアログボックスを含めます。
   - **Next** を選択します。
4. Advanced Options ダイアログボックスを使用して、インストールタイプを指定します。Postgres サーバーと PEM-HTTPD サーバーの両方が現在のホストに存在する場合、
   - **Web Services and Database** を選択します。このオプションは、既存の Postgres サーバーを使用して PEM サーバーをホストする場合、または PEM サーバーインストーラーを使用して PEM サーバーが存在する場所に Postgres サーバーをインストールする場合に有効です。

Web サービスとデータベースを選択した場合、PEM サーバーインストーラーは、PEM-HTTPD インストールの現在のホストを確認し、必要に応じて PEM-HTTPD をアップグレードまたはインストールします。

- **Web Services** PEM-HTTPD サーバーのみの場合現在のホストに存在します。このオプションを呼び出す方法の詳細については、Installing web services を参照してください。現在のホストに PEM サーバーのみをインストールし、pem バックエンドデータベースを作成する場合は、
- **データベース**。このオプションは、既存の Postgres サーバーを使用して PEM サーバーをホストしている場合、または PEM サーバーインストーラーを使用して PEM が存在する PostgreSQL サーバーをインストールしている場合に有効です。

インストールオプションを選択したら、[**Next**] を選択します。

1. [Database Server Selection] ダイアログボックスのリストを使用して、PEM サーバーのバックエンドデータベースを選択します。

    - Postgres ワンクリックインストーラーまたは EDB Postgres Advanced Server インストーラーを使用してインストールされた、現在のホスト上の Postgres サーバーの名前を選択します.
    - **PostgreSQL x (Packaged)** を選択して、PEM サーバーインストーラーでパッケージ化された PostgreSQL サーバーをインストールして使用します。**x** は、PostgreSQL データベースサーバーのバージョンです。
    - 使用する **Other Database Server** を選択します EnterpriseDB インストーラー以外のソースからインストールされた Postgres データベース rpm やソースからビルドなど。

注釈: 選択したデータベースサーバーには、 *sslutils* contrib モジュールのインストールと登録済みサービス Windows が含まれている必要があります。

詳細については、 *Postgres* **サーバーの準備** を参照してください。

Advanced Options ダイアログボックスで **Web Services and Database** を選択した場合、インストールウィザードは、現在のホストで既存の PEM-HTTPD インストールを確認し、必要に応じてサービスをアップグレードまたはインストールします。

Advanced Options ダイアログボックスで **Server** を選択した場合、Database Server Installation Details ダイアログボックスが開きます。

1. [Database Server Installation Details] ダイアログボックスのフィールドを使用して、PEM サーバーをホストする Postgres サーバーへの接続を説明します。

    - **User** フィールドにデータベースのスーパーユーザーの名前を入力します。
    - 関連付けられたパスワードを入力します **Password** フィールドでスーパーユーザーと。

Next を選択します。

1. 実行する PEM エージェントサービスの管理者パスワードを入力します。**Next** を選択します。
2. Network Details ダイアログボックスを使用して、サーバーに接続する PEM エージェントの CIDR スタイルのネットワークアドレス `client-side` アドレスを指定します。指定されたアドレスがサーバーの `pg_hba.conf file.` に追加されます

PostgreSQL サーバー上の `pg_hba.conf` ファイルにエントリを手動で追加することにより、より多くのネットワークアドレスを指定できます。最初のエントリをテンプレートとして使用します。

ネットワークアドレスを追加したら、[**Next**] を選択します。

PEM サーバーインストーラーは、サーバーが存在するホストに PEM エージェントをインストールして、サーバーを監視し、アラート処理とガベージコレクションサービスを提供します。証明書は、**エージェント証明書パス**フィールドで指定された場所にもインストールされます。

1. PEM エージェントの代替説明または代替エージェント証明書パスを入力するか、デフォルトを受け入れます。**Next** を選択します。

2. ウィザードで PEM サーバーをインストールする準備が整いました。 **Next** を選択して、インストールを続行します。

インストールプロセス中に、インストーラーはシステムにファイルをコピーし、PEM サーバーのバックエンドデータベースをセットアップします。確認は、pem データベースが作成および構成されたことを示します。

1. **OK** を選択します。

別のホストにある PEM-HTTPD サービスを使用している場合、以下を行う必要があります。

- Postgres サーバーホスト上の pg_hba.conf ファイルを変更して、ホスト間の接続を許可します。
- PEM-HTTPD サーバーのホストで PEM サーバーインストーラーを起動します。PEM-HTTPD のインストールの詳細については、Installing Web Services を参照してください。

コマンドラインからのサーバーインストーラーの呼び出し

PEM サーバーと PEM エージェントインストーラーのコマンドラインオプションは、リソースまたはシステム構成が制限されているためにグラフィカルインストールが機能しない場合に機能を提供します。次のことができます。

- インストーラーを呼び出すときに--mode unattended オプションを含めて、ユーザー入力なしでインストールを実行します。

すべてのコマンドラインオプションがすべてのプラットフォームに適しているわけではありません。コマンドラインオプションの完全なリファレンスガイドについては、インストーラーを起動するときに--help オプションを使用します。

無人モードでの PEM サーバーインストーラーの呼び出し

インストーラーを呼び出すときにコマンドラインでインストール設定を指定することにより、PEM サーバーの無人インストールを実行できます。PEM サーバーをインストールするシステムは、インターネットにアクセスできること必要です。

PEM サーバーをインストールするには管理者権限が必要です。PEM サーバーインストーラーを呼び出す前に、次の依存関係をインストールする必要があります。

- PostgreSQL
- pem-httpd
- 言語パック

PEM サーバーインストーラーを使用して、PEM サーバーの依存関係を満たすことができます。インストーラーの場所に移動し、次のコマンドを使用して依存関係を抽出します。

```
pem-server-7.<x>.<x>-windows-x64.exe --extract-dependents C:\
```

この例では、ファイルは C:\ ディレクトリに抽出されます。ファイルを抽出した後、各プログラムをインストールする必要があります。ファイルこの例では C:\) が含まれるディレクトリに移動し、次のように入力します。

```
edb-languagepack-<version>-windows-x64.exe --mode unattended
pem-httpd-<version>-windows-x64.exe --mode unattended
postgresql-<version>-windows-x64.exe --mode unattended
```

次に、PEM サーバーインストーラーを呼び出すことができます。

```
pem-server-7.<x>.<x>-windows-x64.exe --mode unattended
- -existing-user <registered_edb_user> --existing-password
<edb_user_password> --pgport <port> --pguser postgres
- -agent_description pem-agent --systempassword <windows_password>
- -agent-crt-path C:\edb``
```

そこで

- registered_edb_user は、登録された EnterpriseDB ユーザーの名前を指定します。登録するには、EDB website にアクセスします。
- edb_user_password は、EDB ユーザーアカウントに関連付けられたパスワードを指定します。
- port は、バッキング PostgreSQL データベースが使用するポートを指定します。デフォルトでは、PostgreSQL データベースはポート 5432 を使用します。
- cidr_address_range は、PEM サーバーのバッキングデータベースの pg_hba.conf ファイルに追加して、サーバーが監視するエージェントからの接続を許可するアドレスレンジを指定します。ネットワーク範囲たとえば 192.168.2.0/24 を指定して、同じネットワーク上にあるエージェントにサーバーアクセスを提供できます。
- windows_password は、Windows 管理者のアカウントに関連付けられたパスワードを指定します。

注釈: PostgreSQL インストーラーは、無人モードで呼び出されると、パスワードが`postgres` である`postgres` という名前のユーザーを作成します。

Creating an EDB repository on an isolated network

PEM をアップグレードするサーバーが EDB リポジトリに直接アクセスできない場合、PEM RPM パッケージのホストとして機能するローカルリポジトリを作成できます。これは、必要な手順の高レベルの概要です。独自のネットワークに合わせてプロセスを変更する必要がある場合があります。

ローカルリポジトリを作成して使用するには

1. インターネットにアクセスできるシステムで次のコマンドを使用して、PEM の依存関係をダウンロードします。

```
yum install yum-plugin-downloadonly
mkdir /<pem_dir>
yum install --downloadonly --downloaddir=/<pem_dir>/ edb-pem
```

(次のページに続く)

(前のページからの続き)

```
mkdir /<epel_dir>
yum install --downloadonly --downloaddir=/<epel_dir>/ epel-release*
```

<pem_dir> および<epel_dir> は、RPM をダウンロードするために作成するローカルディレクトリです。

2. ディレクトリ/<pem_dir> および/<epel_dir> を、隔離されたネットワーク内のマシンにコピーします。

3. リポジトリを作成します。

```
createrepo /<pem_dir>
createrepo /<epel_dir>
```

4. 次のことを指定する接続情報を使用して、/etc/yum.repos.d/pem.repo という名前のリポジトリー構成ファイルを作成します。

```
[pemrepo]
name=PEM Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

5. 次のことを指定する接続情報を使用して、 /etc/yum.repos.d/epel.repo という名前のリポジトリー構成ファイルを作成します。

```
[epelrepo]
name=epel Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

6. ローカルリポジトリの場所と接続情報を指定した後、dnf コマンドを使用して PEM サーバーをインストールまたはアップグレードできます。

PEM サーバーをインストールするには

```
dnf install edb-pem
```

PEM サーバーをアップグレードするには

```
dnf upgrade edb-pem
```

Configuring the PEM server on Linux

PEM サーバーパッケージには、Linux プラットフォームインストールの構成プロセスの自動化に役立つスクリプト `configure-pem-server.sh` が含まれています。スクリプトは`/usr/edb/pem/bin` ディレクトリにインストールされます。スクリプトを呼び出すには、次のコマンドを使用します。

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

注釈: SSL 証明書を使用している場合、すべての SSL 証明書がバックエンドデータベースサーバーのデータディレクトリにあることを確認します。証明書がデータディレクトリにない場合、PEM サーバーの構成中にデータディレクトリを調べるため、PEM サーバーの configure スクリプトは失敗する場合があります。

スクリプトを呼び出すときに、構成プロパティを指定するコマンドラインオプションを含めることができます。スクリプトは、コマンドラインで省略した値の入力を求めます。受け入れられるオプションは次のとおりです。

Option	Description
-acp or *--pemagent-certificate-path*	PEM エージェント証明書パスを定義します。デフォルトは`/root/.pem`です。
-ci or *--cidr-address*	エージェントがサーバーに接続する CIDR 形式のネットワークアドレスレンジ。サーバーの `pg_hba.conf` ファイル `192.168.1.0/24`などに追加されます。デフォルトは`0.0.0.0/0`です。
-dbi or *--db-install-path*	データベースサーバーインストールのディレクトリたとえば、EDB Postgres Advanced Server の場合は`/usr/edb/as12`、PostgreSQL の場合は`/usr/pgsql-12`です。
-ds or *--db-unitfile*	PEM データベースサーバーのユニットファイル名。 EDB Postgres Advanced Server の場合、デフォルトのファイル名は`edb-as-12`です。 PostgreSQL の場合、それは`postgresql-12`です。
-ho or *--host*	PEM データベースサーバーのホストアドレス。
-p or *--port*	PEM データベースサーバーのポート番号。
-ps or *--pemagent-servicename*	pemagent のサービス名。デフォルト値は`pemagent`です。
-sp or *--superpassword*	PEM データベースサーバーのスーパーユーザーパスワード。この値は必須です。
-su or *--superuser*	PEM データベースサーバーのスーパーユーザー名。
-au or *--use-agent-user*	PEM エージェントのユーザー名。
-t or *--type*	インストールタイプ 構成が Web サービスとバックエンドデータベースの場合は`1`、Web サービスを構成する場合は`2`、またはバックエンドデータベースを構成する場合は`3`を指定します。 `3`を指定する場合、データベースはローカルホストに存在する必要があります。
-un or *--uninstall-pem-server*	PEM サーバーをアンインストールします。
-nhc or *--no-hba-change*	`pg_hba.conf`および`pg_config`ファイルに行われた変更をスキップします。

次のページに続く

表 6 – 前のページからの続き

Option	Description
-uac or *--use-agent-sslcert*	PEM サーバーの構成中に、既存のエージェント SSL 証明書を再利用します。
-uak or *--use-agent-sslkey*	PEM サーバーの構成中に、既存のエージェント SSL キーを再利用します。
-scs or *--server-certificate-subject*	カスタム Web サーバー証明書のサブジェクトを提供します。フォーマットは `/C=CountryCode/ST=StateName/L=CityName/O=CompanySelectionName/CN=CommonNameorHos す。 `C=CountryCode`を `Alpha-2`コードとして提供します。
-h or *--help*	PEM サーバーの構成中に使用可能なすべてのオプションをリストします。

コマンドラインで構成プロパティを指定しない場合、スクリプトは値の入力を求めます。スクリプトを呼び出すときに、次から選択します

1. `Web Services and Database` —Web サーバーとデータベースの両方が PEM サーバーと同じホストにある場合に、このオプションを選択します。
2. `Web Services` —Web サーバーが PEM サーバーとは別のホストにある場合、このオプションを選択します。
3. `Database` —PEM サーバーが使用する PEM バックエンドデータベースを構成するには、このオプションを選択します。指定されたデータベースはローカルホストに存在する必要があります。

注釈: Web サーバーとバックエンドデータベース PEM サーバーが別のホストにある場合、最初にデータベースサーバーオプション 3 を構成し、次に Web サービスオプション 2 を構成します。スクリプトは、Web サービスの前にバックエンドデータベースが構成されている場合にのみ続行されます。

構成オプションを選択すると、スクリプトは構成プロパティの入力を求めます。スクリプトが完了すると、PEM サーバーに必要なオブジェクトを作成するか、必要な構成手順を実行します。スクリプトのヘルプを表示するには、次のコマンドを使用します。

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

Web サーバーと PEM バックエンドデータベースが個別にインストールされている場合の構成後の手順

バックエンドデータベースとは別のホストで Web アプリケーションサーバーを実行する場合、PEM が完全に動作する前に、いくつかの追加の手動手順を実行する必要があります。

バックエンドの Postgres データベースが、Web アプリケーションサーバーから PEM へのログインが許可されたユーザーからの接続を受け入れることを確認します。これを行うには、次のエントリを `pg_hba.conf` に追加します。

edb_notranlate_2 <`web_app_ip`> は、Web アプリケーションサーバーの IP アドレスです。

さらに、Web アプリケーションサーバーの IP アドレスが、スクリプトの実行時に指定されたネットワークアドレス範囲内にない場合、2 つのエントリを追加して、このサーバーの PEM エージェントが接続できるようにする必要があります。

```
host pem +pem_agent <web_app_ip>/32 md5
host pem +pem_agent <web_app_ip>/32 cert
```

<web_app_ip> は、Web アプリケーションサーバーの IP アドレスです。

PEM アプリケーションへのアクセス

PEM サーバーを構成すると、ブラウザーで PEM Web インターフェイスにアクセスできます。次の場所に移動します

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

デフォルトでは、Web サービスはポート 8443 でリッスンします。ポートを変更するには、*Changing the default port* を参照してください。

3.2 Installing Postgres Enterprise Manager agent

リンクを選択して、該当するインストール手順にアクセスします。

3.2.1 Linux x86-64 (amd64)

Red Hat Enterprise Linux RHEL とその派生

- *Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le* 、*Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le* 、*Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky* 、 *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky* 、 *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le* 、*Installing Postgres Enterprise Manager agent on SLES 12 ppc64le*

Debian とデリバティブ

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

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

Red Hat Enterprise Linux RHEL とその派生

- *Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le*、*Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*、*Installing Postgres Enterprise Manager agent on SLES 12 ppc64le*

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

Debian とデリバティブ

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

3.2.4 Windows

- *Installing Postgres Enterprise Manager agent on Windows*

Installing Postgres Enterprise Manager agent on Linux x86 (amd64)

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Red Hat Enterprise Linux RHEL とその派生

- *Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky*
- *Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky*

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*
- *Installing Postgres Enterprise Manager agent on SLES 12 ppc64le*

Debian とデリバティブ

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

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

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

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

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on AlmaLinux 9 or Rocky

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on AlmaLinux 8 or Rocky

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on SLES 15 x86_64

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

- 必要な SUSE モジュールをアクティブにします。

```
# You can use SLES 15 SP3 for PEM 8.3 and later:
sudo SUSEConnect -p PackageHub/15.3/x86_64
```

(次のページに続く)

(前のページからの続き)

```
# You can use SLES 15 SP4 for PEM 8.6 and later:
sudo SUSEConnect -p PackageHub/15.4/x86_64
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on SLES 12 x86_64

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

- 必要な SUSE モジュールをアクティブにします。

```
sudo SUSEConnect -p PackageHub/12.5/x86_64
sudo SUSEConnect -p sle-sdk/12.5/x86_64
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on Ubuntu 22.04 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on Ubuntu 20.04 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on Debian 12 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on Debian 11 x86_64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

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

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Red Hat Enterprise LinuxRHEL

- *Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le*
- *Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le*

SUSE Linux Enterprise SLES

- *Installing Postgres Enterprise Manager agent on SLES 15 ppc64le*
- *Installing Postgres Enterprise Manager agent on SLES 12 ppc64le*

Installing Postgres Enterprise Manager agent on RHEL 9 ppc64le

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on RHEL 8 ppc64le

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
dnf repolist | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on SLES 15 ppc64le

---

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

- 必要な SUSE モジュールをアクティブにします。

```
# You can use SLES 15 SP3 for PEM 8.3 and later:
sudo SUSEConnect -p PackageHub/15.3/ppc64le

# You can use SLES 15 SP4 for PEM 8.6 and later:
sudo SUSEConnect -p PackageHub/15.4/ppc64le
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on SLES 12 ppc64le

---

注釈: Postgres Enterprise Manager 8.3 以降は SLES でサポートされています。

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
zypper lr -E | grep enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

- 必要な SUSE モジュールをアクティブにします。

```
sudo SUSEConnect -p PackageHub/12.5/ppc64le
sudo SUSEConnect -p sle-sdk/12.5/ppc64le
```

- メタデータを更新します。

```
sudo zypper refresh
```

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on Linux AArch64 (ARM64)

オペレーティングシステム固有のインストール手順は、対応するドキュメントに記載されています。

Debian とデリバティブ

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

Installing Postgres Enterprise Manager agent on Debian 12 arm64

前提条件

インストールプロセスを開始する前に

- EDB リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリが存在するかどうかを判断するには、次のコマンドを入力します。

```
apt-cache search enterprisedb
```

出力が生成されない場合、リポジトリはインストールされていません。

EDB リポジトリを設定するには

1. EDB repositories に移動します .
2. EDB リポジトリへのアクセスを提供するボタンを選択します。
3. ダウンロードするプラットフォームとソフトウェアを選択します。
4. EDB リポジトリの設定手順に従ってください。

パッケージをインストールする

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

PEM エージェントをインストールした後、PEM エージェントを登録する必要があります。詳細については、*Registering a PEM agent* を参照してください。

Installing Postgres Enterprise Manager agent on Windows

Windows の PEM エージェントグラフィカルインストーラーは、PEM エージェントをインストールおよび登録します。

PEM エージェントインストーラーを起動するには、管理者権限を引き受け、インストーラーが含まれるディレクトリに移動します。次に、インストーラーを呼び出します。

```
pem_agent-9.<x>.<x>-<x>-platform.exe
```

1. [Setup] 画面が開き、PEM エージェントインストーラーが開きます。**Next** を選択します。
2. 適切なラジオボタンを選択して契約に同意する前に、ライセンス契約を確認してください。**Next** を選択します。
3. デフォルトでは、PEM エージェントは `C:\Program Files (x86)\edb\pem` ディレクトリにインストールされます。デフォルトのインストールディレクトリをそのまま使用できます。または、**インストールディレクトリ**フィールドの内容を変更し、PEM エージェントの代替インストールディレクトリを指定できます。
   - デフォルトでは、PEM エージェントインストーラーは、管理者の`%APPDATA%\pem` ディレクトリに証明書を配置します。インストールプロセスに証明書ファイルの代替パスを指定できるダイアログボックスを含めるには、**Show advanced options** を選択します。
   - **Register now?** を選択して、新しくインストールされた PEM エージェントを PEM サーバーに登録します.
   - **Next** を選択します。
4. [PEM Server Installation Details] ダイアログボックスに PEM サーバーの接続の詳細を入力します。
   - **Host** フィールドに PEM データベースサーバーが存在するシステムの名前または IP アドレスを指定します。PEM-HTTPD Web サーバーと PEM データベースが別のシステムでホストされている場合、PEM データベースのホストを指定する必要があります。
   - **User Name** フィールドにデータベーススーパーユーザーの名前を指定します。
   - に関連付けられたパスワードを指定します **Password** フィールドにデータベースのスーパーユーザー。
   - PostgreSQL が **Port** フィールドで監視しているポートを指定します。
5. **Next** を選択して、pemAgent サービスアカウントに進みます。インストーラーはサーバーに接続して、詳細が正しいことを確認しようとします。

注釈: PEM サーバーは、PEM エージェントインストーラーからの接続を許可する必要があります。接続エラーが発生した場合は、[PEM Server Installation Details] ダイアログボックスで指定した接続プロパティが正しいことを確認します。 PEM サーバー上の `pg_hba.conf` ファイルで、エラーメッセージに記載されているサーバーへの接続が許可されていることを確認します。

1. 実行する pemAgent サービスの edb アカウントのパスワードを入力します。エージェント証明書とキーファイルは `C:\Users\edb\AppData\Roaming\pem` ディレクトリに作成されます。 **Next** を選択します。

2. PEM Web インターフェースの [ブラウザ] パネルに表示されるツリーには、**Description** フィールドに入力された値が表示され、PEM エージェントを識別します。エージェントがインストールされているマシンのホスト名、またはホストの機能を反映した名前など、エージェントの説明的な名前を指定します。

わかりやすい名前を指定するか、PEM エージェントホストが提供するデフォルトを受け入れます。**Next** を選択します。

1. **Show advanced options** を選択した場合、 Advanced Options ダイアログボックスが開きます。デフォルトでは、PEM エージェントインストーラーは証明書を `C:\Program Files (x86)\edb\pem` ディレクトリに配置します。証明書の代替パスを指定するか、デフォルトを受け入れて **Next** を選択します。

2. ウィザードで PEM エージェントをインストールする準備が整います。**Next** を選択します。

3. [Ready to Install] ダイアログボックスで、[**Next**] を選択します。インストーラーはファイルをシステムにコピーし、PEM サーバーにエージェントを登録します。

インストールが完了すると、PEM エージェントが実行され、オペレーティングシステムとホストデータが PEM サーバーに報告されます。PEM エージェントのホストで Postgres インスタンスの監視を開始するには、PEM のエンタープライズディレクトリに追加し、エージェントにバインドします。

コマンドインからエージェントインストーラーの呼び出し

PEM エージェントインストーラーのコマンドラインオプションは、リソースまたはシステム構成が制限されているためにグラフィカルインストールが機能しない場合に機能を提供します。次のことができます。

- インストーラーを呼び出すときに`--mode unattended` オプションを含めて、ユーザー入力なしでインストールを実行します。

すべてのコマンドラインオプションがすべてのプラットフォームに適しているわけではありません。コマンドラインオプションの完全なリファレンスガイドについては、インストーラーを起動するときに`--help` オプションを含めます。

無人モードでの PEM エージェントインストーラーの呼び出し

インストーラーを呼び出すときにコマンドラインでインストール設定を指定することにより、PEM サーバーの無人インストールを実行できます。PEM サーバーをインストールするシステムはインターネットにアクセスできること必要です。

無人モードで PEM エージェントインストーラーを呼び出す前に、以下を行う必要があります。

- PEM サーバーをインストールします。 PEM サーバーの pg_hba.conf ファイルは、PEM エージェントのホストからの接続を許可する必要があります。
- 監視対象の Postgres データベースで SSL が有効になっていて、接続を受け入れていることを確認します。

PEM エージェントをインストールするには管理者権限が必要です。次のコマンドを使用して、PEM エージェントインストーラーを無人モードで呼び出します。

```
pem-agent-9.<x>.<x>-windows-x64.exe --mode unattended
- -pghost <pem_server_host_address> --pgport <pem_server_port>
- -pguser postgres --pgpassword <pguser_password>
- -agent_description <agent_name>
```

そこで

- pem_server_host_address は、PEM サーバーのホストの IP アドレスを指定します。
- pem_server_port は、バッキング PEM データベースが使用するポートを指定します。デフォルトでは、データベースはポート 5432 を使用します。
- pguser_password は、PEM データベースのスーパーユーザーに関連付けられたパスワードを指定します。
- agent_name は、PEM エージェントの説明的な名前を指定します。

注釈: Windows エージェントによって実行されるシェル/バッチスクリプトを構成する場合、*agent.cfg* ファイルで `AllowBatchJobSteps` パラメーターを `True` に設定します。 PEM エージェントは、デフォルトではバッチ/シェルスクリプトを実行しません。

3.3 Upgrade and migration

PEM インストール、バックエンドデータベース、SQL プロファイラーを含む Postgres Enterprise ManagerPEM コンポーネントをアップグレードできます。PEM サーバーを移動することもできます。

注釈: PEM 7.x を最新バージョンにアップグレードしており、PEM バックエンドデータベースが PostgreSQL または EDB Postgres Advanced Server バージョン 10 以前の場合 1. PEM バックエンドデータベースをアップグレードします。 2. PEM インストールをアップグレードします。

PEM バックエンドデータベースが PostgreSQL または EDB Postgres Advanced Sever バージョン 11 以降の場合、PEM バックエンドデータベースのアップグレードはオプショナル。

3.3.1 Upgrading a PEM installation

PEM インストールをアップグレードするプロセスはプラットフォーム固有です。Linux プラットフォームではネイティブパッケージを使用し、Windows プラットフォームでは Windows で使用可能な PEM グラフィカルインストーラーを使用して PEM エージェントまたはサーバーを更新できます。

注釈:

- PEM 7.8 リリースより前は、グラフィカルインストーラーまたは RPM を使用して、Linux に PEM エージェントまたはサーバーをインストールしました。 - PEM バージョン 7.8 以降では、Linux 用の PEM グラフィカルインストーラーが廃止されました。 Linux ホスト上の PEM エージェントまたはサーバーを以前のバージョンから PEM バージョン 7.9 以降にアップグレードするには、ネイティブパッケージを使用する必要があります。

PEM インストーラーと RPM へのリンクは、 EDB website で入手できます。

Upgrading a PEM installation on a Linux host

Linux ホストで PEM コンポーネントソフトウェアをアップグレードするには、次の順序で PEM コンポーネントパッケージの新しいバージョンをインストールします。

1. PEM サーバーホストを除く各監視対象ノードで PEM エージェントパッケージインストーラーを起動します。
2. PEM サーバーパッケージインストーラーを起動します。PEM サーバーと、PEM サーバーホストにある PEM エージェントの両方をアップグレードします。

インストール中に、コンポーネントのインストールは既存のインストールを検出し、アップグレードを実行します。PEM エージェントとサーバーをアップグレードした後、必要に応じて SQL プロファイラーをアップグレードできます。この手順はプラットフォーム固有です。

注釈: 以前のバージョンからバージョン 7.11 以降にアップグレードされた Linux エージェントによって実行されるシェル/バッチスクリプトを既に構成している場合、または構成する予定がある場合には、*agent.cfg* ファイルの `batch_script_user` パラメーターのユーザーを指定する必要があります。非 root ユーザーを使用してスクリプトを実行することを強くお勧めします。 root ユーザーを使用すると、データセキュリティとオペレーティングシステムのセキュリティが危険にさらされる可能性があります。ただし、 root ユーザーを使用してスクリプトを実行する場合、pemagent を元の設定に復元する場合は、 *batch_script_user* パラメーターを `root` に設定する必要があります。

PEM サーバーインストールのアップグレード

隔離されたネットワーク内のマシンにインストールされている PEM サーバーをアップグレードする場合は、PEM サーバーをアップグレードする前にそのマシンに PEM リポジトリを作成する必要があります。詳細については、*Creating an EDB repository on an isolated network* を参照してください。

PEM サーバーインストールをアップグレードするには

```
sudo <package-manager> upgrade edb-pem
```

<package-manager> は、オペレーティングシステムで使用されるパッケージマネージャーです。

Package manager	Operating system
dnf	RHEL 8/9 and derivatives
zypper	SLES
apt-get	Debian and Ubuntu

PEM サーバーをアップグレードした後、PEM サーバーを構成する必要があります。詳細については、*Configuring the PEM server on Linux* を参照してください。

注釈: PEM バックエンドデータベースサーバーと PEM サーバーをアップグレードする場合は、configure スクリプトを実行する前に、*/usr/edb/pem/share/.install-config* ファイルの新しいバージョンを指す `PG_INSTALL_PATH` および `DB_UNIT_FILE` パラメーターを更新します。

PEM サーバーの構成

PEM サーバーをアップグレードした後、次のコマンドを使用して PEM サーバーを構成できます。

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

configure スクリプトは、スクリプトの実行中に古い PEM サーバー構成ファイルの値を使用します。詳細については、を参照してください。

注釈: -configure スクリプトは、アップグレードプロセスの後にのみスーパーユーザーパスワードを必要とします。

- configure スクリプトがスタックした場合は、`alert_threads>0` で PEM エージェントを停止します。このようなエージェントの詳細を取得するには、次のクエリを実行します。

```
SELECT agent_id FROM pem.agent_config WHERE param=alert_threads AND value > 0;
```

実行中のエージェントを停止し、configure スクリプトを再実行します。

問題が解決しない場合は、クエリを実行して、スタックしたアラートプロセスを終了します。

```
SELECT pg_terminate_backend(pid) FROM pg_stat_activity WHERE query=SELECT pem.process_
↪one_alert();
```

次に、configure スクリプトを再実行します。

PEM エージェントインストールのアップグレード

PEM エージェントをアップグレードするには

```
sudo <package-manager> upgrade edb-pem-agent
```

<package-manager> は、オペレーティングシステムで使用されるパッケージマネージャーです。

Package manager	Operating system
dnf	RHEL 8/9 and derivatives
zypper	SLES
apt-get	Debian and Ubuntu

Upgrading a PEM installation on a Windows host

Windows ホストで PEM コンポーネントソフトウェアをアップグレードするには、次の順序で PEM コンポーネントインストーラーの新しいバージョンを呼び出します。

1. PEM サーバーホストを除く各監視対象ノードで PEM エージェントインストーラーを起動します。
2. PEM サーバーインストーラーを起動します。このインストーラーは、PEM サーバーと、PEM サーバーホストにある PEM エージェントの両方をアップグレードします。

インストール中に、コンポーネントインストーラーは既存のインストールを自動的に検出し、アップグレードを実行します。PEM エージェントとサーバーをアップグレードした後、必要に応じて SQL プロファイラーをアップグレードできます。この手順はプラットフォーム固有です。

Windows ホストでの PEM エージェントのアップグレード

現在 PEM エージェントによって監視されているシステムをより新しい PEM エージェントにアップグレードするには、エージェントが監視しているシステムで PEM エージェントインストーラーの新しいバージョンをダウンロードして起動します。

1. インストーラーを起動するには、ダウンロードしたインストーラーアイコンを右クリックし、**Run as Administrator** を選択します。 PEM Agent Setup ウィザードが開きます。
2. ライセンス契約を読んで同意し、 **Next** を選択します。
3. セットアップウィザードは、既存のエージェントを自動的に検出し、インストールされたバージョンをアップグレードします。**Next** を選択します。

4. pemAgent Service Account ダイアログボックスは、PEM エージェントサービスを実行するアカウントのパスワードの入力を求められる場合があります。プロンプトが表示されたら、パスワードを入力し、**Next** を選択します。

5. インストールが間もなく開始されることを通知する [Ready to install] ダイアログボックスは、[**Next**] を選択します。ウィザードは PEM エージェントを最新バージョンにアップグレードします。

6. PEM Agent Setup ウィザードは、インストールが完了すると通知します。**Finish** を選択します。

7. インストールが完了すると、マシンを再起動するように求められます。**Yes** を選択して、マシンと PEM エージェントを再起動します。

Windows ホストでの PEM サーバーのアップグレード

PEM サーバーインストーラーを使用すると、PEM サーバーのメジャーバージョン間でアップグレードできます。最初にバージョン 6.0 にアップグレードせずに、バージョン 5.0 からバージョン 7.16 にアップグレードできます。

1. インストーラーを起動するには、ダウンロードしたインストーラーを右クリックし、**Run as Administrator** を選択します。

2. PEM Server Setup ウィザードがあなたを迎えます。 **Next** を選択します。

3. PEM Server Setup ウィザードは、ライセンス契約に同意するように求めます。ライセンス契約を確認した後、**I accept the Agreement** を選択し、 **Next** を選択します。

4. ウィザードは、PEM サーバーホストで既存の PEM サーバーインストールを確認します。ウィザードはインストールを見つけると、アップグレードを実行します。**Next** を選択します。

PEM サーバーをアップグレードする前に、ウィザードは、新しい PEM サーバーの要件が存在することを確認します。サポートコンポーネントが欠落しているか、新しい PEM インストールをサポートしていないバージョンである場合、ウィザードは依存関係をアップグレードする必要があることを通知します。次に、必要なインストーラーを呼び出します。

1. インストールウィザードが依存関係のアップグレードを完了すると、マシンを再起動するように求められます。**No** を選択して、アップグレードプロセスを続行します。

2. ウィザードは [Database Server Installation Details] ダイアログボックスを開き、PEM バックエンドデータベースのデータベーススーパーユーザーの接続資格情報の入力を求めます。提供する

- **User** フィールドのデータベーススーパーユーザーの名前。
- **Password** フィールドのデータベーススーパーユーザーに関連付けられたパスワード。

Next を選択します。

1. pemAgent Service Account ダイアログボックスは、PEM エージェントサービスを実行するアカウントのパスワードの入力を求められる場合があります。プロンプトが表示されたら、パスワードを入力し、**Next** を選択します。

2. Ready to Install ダイアログボックスは、セットアップウィザードがインストールを実行する準備ができたことを示します。**Next** を選択してインストールを開始します。

PEM サーバーおよび PEM サーバーと同じホストにあるエージェントをアップグレードし、Web サービスを構成すると、PEM セットアップウィザードは、サービスがリッスンしているポートを通知します。PEM クライアントを使用して PEM サーバーに接続するときにこのポート番号を使用します。

1. **OK** を選択します。PEM サーバーセットアップウィザードは、インストールが完了したことを通知します。
2. マシンを再起動するようにプロンプトが表示されたら、 **Yes** を選択してマシンと httpd サービスを再起動します。

PEM バックエンドデータベースサーバーと PEM-HTTPD を別のホストにインストールした場合、PEM サーバーインストーラーを各ホストで 1 回ずつ、2 回実行する必要があります。言語パックインストーラーを抽出し、PEM インストーラーを呼び出す前に PEM-HTTPD のホストにインストールします。インストーラーを呼び出して言語パックを抽出するときに、次のキーワードを含めます。

```
--extract-languagepack <path>
```

<path> は、言語パックインストーラーを抽出するための既存のパスを指定します。

注釈: デフォルトでは、EDB Language Pack は `C:edblanguagepackv1` にインストールされます。

StackBuilder Plus を使用して PEM サーバーをアップグレードしている場合、エラーが表示される場合があります。エラーが表示された後、PEM はインストールが完了したことを報告します。ただし、インストールは完了していません。ダウンロードした場所からインストーラーファイルを呼び出して、インストールを行う必要があります。

PEM サーバーをアップグレードした後、バックエンドデータベースをより新しいバージョンにアップグレードすることができます。詳細については、*Upgrading the PEM backend Postgres database* を参照してください。

Upgrading a PEM installation on high-availability mode using HTTPD

PEM サーバーで *Using Failover Manager for high availability* を使用している場合、PEM パッケージを更新することにより、PEM インストールのメジャーまたはマイナーアップグレードを実行できます。

この例では、次の IP アドレスを使用します。

- プライマリサーバー 172.17.0.2
- セカンダリサーバースタンバイ 172.17.0.3

このアップグレード例は、Web サーバーサービスに HTTPD が使用される RHEL ベースのオペレーティングシステムで実行されます。

パッケージのアップグレード

1. プライマリサーバー 172.17.0.2 で、PEM パッケージを更新します。

```
dnf update edb-pem
```

更新すると、アップグレードに必要なスクリプトの最新バージョンを利用できるようになります。

1. アップグレードスクリプトを実行します。PEM を利用可能な最新バージョンにアップグレードし、バックエンドデータベース、Web サーバー、および PEM エージェントを再起動します。

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

1. セカンダリスタンバイサーバー 172.17.0.3 で、PEM パッケージを更新します。

```
dnf update edb-pem
```

1. 構成ファイル、証明書、およびキーをプライマリサーバーからセカンダリサーバーにコピーします。

```
/etc/httpd/conf.d/edb-pem.conf
/etc/httpd/conf.d/edb-ssl-pem.conf
/usr/edb/pem/share/.install-config
/usr/edb/pem/resources/server-pem.crt
/usr/edb/pem/resources/server-pem.key
/usr/edb/pem/web/pem.wsgi
/usr/edb/pem/web/config_setup.py
```

!!!note 複数のスタンバイサーバーがある場合は、パッケージを更新し、構成ファイル、証明書、およびキーをプライマリサーバーから各スタンバイインスタンスにコピーします。 .. ::
アップグレードが完了しました。

3.3.2 Upgrading the PEM backend Postgres database

PEM コンポーネントと PEM バックエンドデータベースを更新する場合は、バックエンドデータベースを更新する前にサーバーとエージェントで PEM コンポーネントの更新を実行します。PEM コンポーネントソフトウェアの更新の詳細については、*Upgrading a PEM installation on high-availability mode using HTTPD* を参照してください。

注釈:

- PEM 8.0 以降、PostgreSQL または EPAS バージョン 11 以降は、バックエンドデータベースサーバーとしてのみサポートされています。その結果、バックエンドデータベースサーバーがバージョン 11 より前の場合、最初にバックエンドデータベースサーバーをアップグレードし、次に PEM コンポーネントをアップグレードする必要があります。

- バックエンドデータベースサーバーをアップグレードした後、PEM Web インターフェイスでサーバーを作成中にこのエラーが発生した場合

```
Error - User does not have enough permission to add new server.
        Please contact the administrator to grant pem_database_server_registration␣
↪role
        to the enterprisedb user.
```

ロールを更新し、適切な権限を付与することにより、エラーを解決します。

```
UPDATE pem.roles SET rolid = pr.oid FROM pg_roles pr WHERE pr.rolname = pem_ ||␣
↪component;
```

更新プロセスでは、pg_upgrade ユーティリティを使用して、バックエンドサーバーのあるバージョンからより新しいバージョンに移行します。pg_upgrade は、サポートされている Postgres のバージョンと、同じプラットフォームでサポートされている Postgres の後続のリリース間の移行を有効にします。

ソース PEM サーバーが 7.16 バージョンより前の場合、pg_upgrade を実行する前に次の機能を置き換える必要があります。

- abstime 、reltime 、および tinterval データ型は Postgres バージョン 12 以降では非推奨であるため、これらのデータタイプを timestamptz データ型に置き換えるには、次のコマンドを使用します。

```
DO
$$
DECLARE
    rec record;
    cnt integer;
BEGIN
    -- Check for the deprecated type in our user info probe
    SELECT count(*) INTO cnt
    FROM pem.probe_column
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
        IF cnt = 0 THEN
            RETURN;
        END IF;
    ALTER TABLE pemdata.user_info
        ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    ALTER TABLE pemhistory.user_info
            ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    -- Now update the pem.probe_column itself
    UPDATE pem.probe_column
    SET sql_data_type = 'timestamptz'
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
END;
$$ LANGUAGE 'plpgsql';
```

- アラートエラーを回避するには、このファンクションを置き換えます。

```
  CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
```

(次のページに続く)

(前のページからの続き)

```
  template_id pem.alert_template.id%type, params text[]
  )
  RETURNS bool AS $FUNC$
  DECLARE
      res bool := TRUE;
  BEGIN
      /*
- During restoring the pem database, it does not maintain the order while
- inserting data in the table, and uses the sort table based on the
- names.
- Hence - we need to check the foreign key constraint is present before
- validating these values.
      */
      IF EXISTS(
          SELECT 1 FROM information_schema.table_constraints
          WHERE constraint_name=alert_template_id_fkey AND
          table_name=alert AND table_schema=pem
      ) THEN
    /*
- Need to use the IS TRUE construct outside the main query, because
- otherwise if theres no template by that ID then the query would return
- 0 rows and the result of the function would be undefined and CHECK
- constraint would succeed.
- Probably this is being over-cautious, because pem.alert.template_id
- references pem.alert_template.id. But the SQL standard (probably) does
- not define the order in which the CHECK or the FOREIGN KEY constraints
- should be validated; in case CHECK is validated first, we want it to
- fail.
     */
  EXECUTE $SQL$
      SELECT (
          SELECT pem.check_array_size_equal(t.param_names, $2)
          FROM pem.alert_template AS t
          WHERE id = $1
      ) IS TRUE
    $SQL$ INTO res USING template_id, params;
  END IF;
   RETURN res;
  END
  $FUNC$ LANGUAGE plpgsql;
```

pg_upgrade は、同じタイプのサーバー間のデータの転送をサポートしています。たとえば、pg_upgrade を使用して、PostgreSQL 10 バックエンドデータベースから PostgreSQL 11 バックエンドデータベースにデータを移動できますが、EDB Postgres Advanced Server 11 バックエンドデータベースには移動できません。別の種類のバックエンドデータベースに移行する場合 PostgreSQL サーバーから EDB Postgres Advanced Server

などに移行する場合、*Moving the PEM server* を参照してください。

pg_upgrade の使用の詳細については、 pg_upgrade を参照してください。

1. 更新されたインストーラーをダウンロードして起動します。PostgreSQL および EDB Postgres Advanced Server のインストーラーは、EDB website から入手できます。

アップグレードするサーバーバージョンのインストーラーをダウンロードした後、PEM サーバーのホストでインストーラーを起動します。インストールウィザードの画面上の指示に従って、Postgres サーバーを構成およびインストールします。

オプションで、カスタム構築の PostgreSQL サーバーを PEM バックエンドデータベースのホストとして使用できます。ポート 5432 でリッスンする PostgreSQL バックエンドデータベースからアップグレードする場合、新しいサーバーは別のポートでリッスンするように構成する必要があります。

2. 新しいサーバーで SSL ユーティリティを構成します。新しいバックエンドデータベースは、現在のバックエンドデータベースが実行しているものと同じバージョンの sslutils を実行している必要があります。SSL Utils パッケージは、 EDB website からダウンロードできます。

EDB Postgres Advanced Server を新しいバックエンドデータベースとして使用する場合、sslutils 拡張機能を手動で追加する必要はありません。sslutils の構成 sslutils はプラットフォーム固有です。

Linux の場合

- EDB Postgres Advanced Server バックエンドデータベースでは、sslutils 拡張機能がデフォルトでインストールされます。
- PostgreSQL を PEM バックエンドデータベースとして使用している場合、PostgreSQL コミュニティリポジトリにアクセスできることを確認し、使用しますコマンド

```
yum install sslutils_<X>
```

<X> はサーバーバージョンです。

- PostgreSQL の EDB ワンクリックインストーラーを PEM バックエンドデータベースとして使用している場合、次のコマンドを使用します。

```
yum install gcc openssl-devel
```

pg_config プログラムを見つけられるように PATH の値を設定します

```
export PATH=$PATH:/opt/postgres_inst_dir/<X>/bin/
```

sslutils フォルダーに移動し、次のように入力します。

```
make USE_PGXS=1
make USE_PGXS=1 install
```

psql を使用して sslutils 拡張機能を作成する

```
CREATE EXTENSION sslutils
```

Debian 10 および Ubuntu 20 では、セキュリティ上の理由から証明書を受け入れるための要件が強化されました。いずれかのマシンに PEM エージェントをインストールする場合は、sslutils を 1.3 にアップグレードする必要があります。4096 ビット RSA キーと sha256 署名アルゴリズムのサポートが追加されました。sslutils を 1.3 にアップグレードしない場合、PEM エージェントは PEM バックエンドデータベースサーバーへの接続に失敗し、エラー「ca md too promote」がログに記録される場合があります。

**Windows の場合*

- 元のバックエンドデータベースで sslutils をコンパイルするために使用されたものと同じコンパイラーを使用して、新しいバックエンドデータベースで sslutils をコンパイルする必要があります。PostgreSQL ワンクリックインストーラー EDB からまたは EDB Postgres Advanced Server インストーラーを使用してインストールされた Postgres データベースに移動する場合は、Visual Studio を使用して sslutils をビルドしてください 。 PostgreSQL 10 以降にアップグレードする場合は、Visual Studio 2010 を使用してください。

Windows で Postgres の特定のバージョンをビルドする方法の詳細については、そのバージョンのコアドキュメントを参照してください。コアドキュメントは
PostgreSQL project website で入手できます。

- プロセスの具体的な詳細はプラットフォームとコンパイラーによって異なりますが、各プラットフォームの基本的な手順は同じです。次の例は、32 ビット Windows システムでの PostgreSQL の OpenSSL サポートのコンパイルを示しています。

OpenSSL 拡張機能をコンパイルする前に、ご使用の Windows バージョンの OpenSSL を見つけてインストールする必要があります。OpenSSL インストーラーを呼び出す前に、前提条件の再配布可能 `vcredist_x86.exe` などをダウンロードしてインストールする必要がある場合があります。

OpenSSL のインストール後、 sslutils utility package をダウンロードして解凍します。

- 解凍した `sslutils` フォルダーを Postgres インストールディレクトリ `C:\ProgramFiles\PostgreSQL\<x.x>` にコピーします。
- Visual Studio コマンドラインを開き、 `sslutils` ディレクトリに移動します。次のコマンドを使用して sslutils をビルドします。

```
SET USE_PGXS=1

SET GETTEXTPATH=\ <path_to_gettext>

SET OPENSSLPATH=\ <path_to_openssl>

SET PGPATH=\ <path_to_pg_installation_dir>

SET ARCH=x86

msbuild sslutils.proj /p:Configuration=Release
```

そこで

- `path_to_gettext` は、 `GETTEXT` ライブラリとヘッダーファイルの場所を指定します。
- `path_to_openssl` は、openssl ライブラリとヘッダーファイルの場所を指定します。

- path_to_pg_installation_dir は、Postgres インストールの場所を指定します。

たとえば、次のコマンドセットは、PostgreSQL 11 サーバーに OpenSSL サポートをビルドします。

```
SET USE_PGXS=1

SET OPENSSLPATH=C:\OpenSSL-Win32

SET GETTEXTPATH="C:\Program Files\PostgreSQL1"

SET PGPATH="C:\Program Files\PostgreSQL1"

SET ARCH=x86

msbuild sslutils.proj /p:Configuration=Release
```

- ビルドが完了すると、sslutils ディレクトリには次のファイルが含まれます。

```
sslutils--1.3.sql

sslutils--unpackaged--1.3.sql

sslutils--pemagent.sql.in

sslutils.dll
```

- コンパイルした sslutils ファイルをインストールの適切なディレクトリにコピーします。例

```
COPY sslutils*.sql "%PGPATH%\share\extension\"

COPY sslutils.dll "%PGPATH%\lib\"

COPY sslutils.control "%PGPATH%\share\extension\"
```

3. 古いバックエンドデータベースと新しいバックエンドデータベースの両方のサービスを停止します。

RHEL 8.x で、コマンドラインを開き、スーパーユーザーの ID を引き継ぎます。次のコマンドを入力します。

```
systemctl <service_name> stop
```

<service_name> は、Postgres サービスの名前を指定します。

Windows、[Services] ダイアログボックスを使用して Services を制御できます。サービスを停止するには 1. Control Panel で、 **System and Security > Administrative Tools** を選択します。 1. **Services** をダブルクリックします。 1. [Services] ダイアログボックスで、サービス名を選択し、[**Stop**] を選択します。

4. pg_upgrade ユーティリティを使用して、古いバックエンドデータベースと新しいバックエンドデータベース間で既存のデータのインプレース転送を実行します。サーバーが md5 認証を強制するように構成されている場合、データベースのスーパーユーザーの接続プロパティおよびパスワードを指定するエントリーを.pgpass ファイルに追加する必要があります。または、 pg_upgrade を呼び出す前に、pg_hba.conf ファイルを変更して信頼接続を許可する必要がある場合があります。.pgpass ファイルへのエントリーの作成の詳細については、PostgreSQL core documentation を参照してください。

アップグレードプロセス中に、pg_upgrade は一連のログファイルを書き込みます。クラスタ所有者は、書き込み権限があるディレクトリから pg_upgrade を呼び出す必要があります。アップグレードが正常に完了した場合、pg_upgrade はアップグレードの完了時にログファイルを削除します。pg_upgrade でアップグレードログファイルを削除したくない場合は、pg_upgrade を呼び出すときに--retain キーワードを含みます。

pg_upgrade を呼び出すには、クラスター所有者の ID を引き継ぎ、クラスター所有者が書き込み権限があるディレクトリに移動し、コマンドを実行します。

```
<path_to_pg_upgrade> pg_upgrade

-d <old_data_dir_path>

-D <new_data_dir_path>

-b <old_bin_dir_path> -B <new_bin_dir_path>

-p <old_port> -P <new_port>

-u <user_name>
```

そこで

- path_to_pg_upgrade は、pg_upgrade ユーティリティの場所を指定します。デフォルトでは、pg_upgrade は Postgres ディレクトリの下の bin ディレクトリにインストールされます。 - old_data_dir_path は、古いバックエンドデータベースのデータディレクトリへの完全なパスを指定します。- new_data_dir_path は、新しいバックエンドデータベースのデータディレクトリへの完全なパスを指定します。- old_bin_dir_path は、古いバックエンドデータベースの bin ディレクトリへの完全なパスを指定します。- new_bin_dir_path は、古いバックエンドデータベースの bin ディレクトリへの完全なパスを指定します。- old_port は、古いサーバーがリッスンしているポートを指定します。- new_port は、新しいサーバーがリッスンするポートを指定します。 - user_name は、クラスター所有者の名前を指定します。

たとえば、次のコマンドは、 Windows システムで PostgreSQL 9.6 から PostgreSQL 11 に PEM データベースを移行するように pg_upgrade に指示しますバックエンドデータベースがデフォルトの場所にインストールされている場合。

```
C:\>"C:\Program Files\PostgreSQL1\bin\pg_upgrade.exe"

-d "C:\Program Files\PostgreSQL0\data"

-D "C:\Program Files\PostgreSQL1\data"

-b "C:\Program Files\PostgreSQL0\bin"

-B "C:\Program Files\PostgreSQL1\bin"

-p 5432 -P 5433

```

(次のページに続く)

(前のページからの続き)

```
-U postgres
```

pg_upgrade は、呼び出されると、データを新しいバックエンドデータベースに移動する前に一貫性チェックを実行します。アップグレードが完了すると、pg_upgrade はアップグレードが完了したことを通知します。

pg_upgrade オプションの使用またはアップグレードプロセスのトラブルシューティングの詳細については、
pg_upgrade を参照してください。

5. 次の証明書ファイルを、古いバックエンドデータベースの data ディレクトリから新しいバックエンドデータベースの data ディレクトリにコピーします。

`ca_certificate.crt`

`ca_key.key`

`root.crt`

`root.crl`

`server.key`

`server.crt`

ターゲットサーバーに配置されたら、ファイルに次のプラットフォーム固有の権限があることを確認します。

Linux での権限と所有権

File name	Owner	Permissions
ca_certificate.crt	postgres	-rw-------
ca_key.key	postgres	-rw-------
root.crt	postgres	-rw-------
root.crl	postgres	-rw-------
server.key	postgres	-rw-------
server.crt	postgres	-rw-r--r--

Linux では、証明書ファイルは postgres が所有する必要があります。コマンドラインで次のコマンドを使用して、ファイルの所有権を変更できます。

```
chown postgres <file_name>
```

`file_name` は、証明書ファイルの名前を指定します。

`server.crt` ファイルの所有者のみが変更できますが、すべてのユーザーが読み取ることができます。次のコマンドを使用して、`server.crt` ファイルのファイル許可を設定できます。

```
chmod 644 server.crt
```

他の証明書ファイルの所有者のみがファイルを変更または読み取ることができます。次のコマンドを使用して、ファイルのアクセス許可を設定できます。

```
chmod 600 <file_name>
```

file_name は、ファイルの名前を指定します。

Windows での権限と所有権

Windows では、ターゲットホストで PEM サーバーとバックエンドデータベースのインストールを実行したサービスアカウントは、ソースホストから移動した証明書ファイルを所有している必要があります。インストーラーのコンテキストメニューから **Run as Administrator** を使用して PEM サーバーと Postgres インストーラーを呼び出した場合、証明書ファイルの所有者は Administrators です。

Windows でファイルのアクセス許可を確認および変更するには、ファイル名を右クリックし、**プロパティ** を選択します。

[**セキュリティ**] タブで、グループまたはユーザー名を選択して、割り当てられたアクセス許可を表示します。**Edit** または **Advanced** を選択して、選択したユーザーに関連付けられている権限を変更できるダイアログボックスを開きます。

6. postgresql.conf ファイルには、サーバーの動作を指定するパラメーター設定が含まれています。古いサーバーの postgresql.conf ファイルで指定された構成と一致するように、新しいサーバーの postgresql.conf ファイルを変更します。

デフォルトでは、 postgresql.conf ファイルは次の場所にあります。

- Linux で Postgres バージョン 10 より前の場合、/opt/PostgreSQL/<X>/data
- Linux でグラフィカルインストーラーでインストールした Postgres バージョン 10 以降の場合、/opt/PostgreSQL/<X>/data
- Linux で RPM でインストールした場合の Postgres バージョン 10 以降の場合、/usr/pgsql/<X>/data
- Windows 上の Postgres バージョンの場合、C:\Program Files\PostgreSQL\<X>\data

<X> は、システム上の Postgres のバージョンです。

選択したエディターを使用して、新しいサーバーの postgresql.conf ファイルを更新します。次のパラメーターを変更します。

- 元のバックエンドデータベースによって監視されるポートでリッスンする port パラメーター通常は 5432 に設定します。
- on に設定する ssl パラメーター

また、次のパラメーターが有効になっていることを確認する必要があります。パラメーターがコメントアウトされている場合は、各 postgresql.conf ファイルエントリーの前にあるポンド記号を削除します。

- ssl_cert_file = 'server.crt' # (change requires restart)
- ssl_key_file = 'server.key' # (change requires restart)
- ssl_ca_file = 'root.crt' # (change requires restart)
- ssl_crl_file = 'root.crl'

インストールには、新しいバックエンドデータベースが古いバックエンドデータベースのように動作するように変更が必要な他のパラメーター設定がある場合があります。postgresql.conf ファイルを注意深く確認して、新しいサーバーの構成が古いサーバーの構成と一致することを確認します。

7. pg_hba.conf ファイルには、サーバーがホストベースの認証を強制する方法を指定するパラメーター設定が含まれています。PEM サーバーをインストールすると、インストーラーは pg_hba.conf ファイルを変更し、ファイルの先頭にエントリを追加します。

3.3.3 Adding entries for PEM Agens and admins to connect to PEM server

```
hostssl pem +pem_user 192.168.2.0/24 md5

hostssl pem +pem_agent 192.168.2.0/24 cert
```

3.3.4 Adding entries (localhost) for PEM Agens and admins to connect to PEM server

```
hostssl pem +pem_user 127.0.0.1/32 md5

hostssl postgres +pem_user 127.0.0.1/32 md5

hostssl pem +pem_user 127.0.0.1/32 md5

hostssl pem +pem_agent 127.0.0.1/32 cert
```

デフォルトでは、 pg_hba.conf ファイルは次の場所にあります。

- Linux で 10 より前の Postgres バージョンの場合、/opt/PostgreSQL/<X>/data
- Linux でグラフィカルインストーラーでインストールされた Postgres バージョン 10 以降の場合、/Opt/PostgreSQL/<X>/data
- Postgres バージョン 10 以降の場合 Linux での RPM、/var/lib/pgsql/<X>/data
- Linux で RPM でインストールされた Advanced Server バージョン 10 以降の場合、/var/lib/edb/as<X>/data
- で Windows の Postgres バージョン、C:\Program Files\PostgreSQL\<X>\data

<X> は、システム上の Postgres のバージョンです。

選択したエディターを使用して、古いサーバーの pg_hba.conf ファイルから新しいサーバーの pg_hba.conf ファイルにエントリをコピーします。

8. 新しいバックエンドデータベースのサービスを再起動します。

RHEL 8.x では、スーパーユーザーとしてコマンドラインで次のように入力します。

```
systemctl stop <service_name>
```

service_name は、バックエンドデータベースサーバーの名前です。

Windows を使用している場合は、 **Services** ダイアログボックスを使用してサービスを制御できます。1. Control Panel で、**System and Security > Administrative Tools** を選択します。 1. **Services** アイコンをダブルクリックします。 1. [Services] ダイアログボックスで、サービス名を選択し、サービスを開始します。

3.3.5 Upgrading SQL Profiler

プラグインに含まれる SQL ファイルをロードしてファンクションが作成される SQL プロファイル v4.0 または以前のバージョンからアップグレードする場合、最初に拡張機能に移行し、後で 4.1 に更新します。

```
#  running as superuser

CREATE EXTENSION sql_profiler VERSION 4.0;

ALTER EXTENSION sql_profiler UPDATE TO 4.1;
```

3.3.6 Upgrading Apache HTTPD server

Apache HTTPD Web サーバーは、PEM Web アプリケーションを提供するために使用されます。Linux では、PEM は HTTPD をバンドルしていませんが、システムパッケージマネージャーが提供するバージョンに依存しています。Windows では、PEM インストーラーは HTTPD のバージョンをバンドルします。

Linux での HTTPD のアップグレード

重要: Red Hat Enterprise LinuxRHEL、Rocky Linux、AlmaLinux、または Oracle Linux を使用しており、HTTPD を更新して脆弱性に対処する場合、 *Note on HTTPD versioning in RHEL and RHEL derivatives* を読んでください。

Linux を実行している場合は、サーバー上の他のシステムソフトウェアとともに HTTPD を管理します。パッケージマネージャーから入手可能な最新バージョンを常に使用することをお勧めします。

アップグレードする前に PEM を停止したり、ファイルをバックアップしたりする必要はありません。パッケージマネージャーからアップグレードを開始します。

RHEL / Rocky Linux / AlmaLinux / Oracle Linux

```
sudo yum update httpd
```

Debian / Ubuntu

```
sudo apt update
sudo apt upgrade apache2
```

SLES

```
sudo zypper update apache2
```

RHEL および RHEL 派生における HTTPD バージョン管理

RHEL または他の RHEL 派生から入手可能な HTTPD の最新バージョンのバージョン番号が、最新のコミュニティリリースよりもかなり古いことに気づくかもしれません。具体的には、RHEL 7 は HTTPD 2.4.6 を提供し、RHEL 8 は 2.4.37 を提供します。 Red Hat は RHEL 用のカスタム HTTPD パッケージをビルドし、セキュリティ修正をバックポートしていることを理解することが非常に重要です。

このため、コミュニティ 2.4.37 リリースに存在する脆弱性が同じバージョンの RHEL パッケージに存在するとは考えないでください。この理由で HTTPD の新しいバージョンをインストールしようとしている場合、それはほぼ確実に必要ありません。詳細については、Red Hat ナレッジベースの Apache HTTPD versions supported by Red Hat 記事を参照してください。

> **警告:** Red Hat は、コミュニティ HTTPD またはソースからビルドされた HTTPD をサポートしていません。 RHEL システムで HTTPD の他のソースの使用を検討している場合は、 Apache HTTPD versions supported by Red Hat 記事を読んで、その影響を理解してください。

Windows での HTTPD のアップグレード

EDB による PEM 用にパッケージされた HTTPD の最新バージョンにアップグレードするには、Windows の最新の PostgreSQL または EDB Advanced Server インストーラーをダウンロードし、 Stack Builder を使用します

または、それぞれ PEM-HTTPD を更新します。 PEM-HTTPD は、Web Development カテゴリの下にあります。

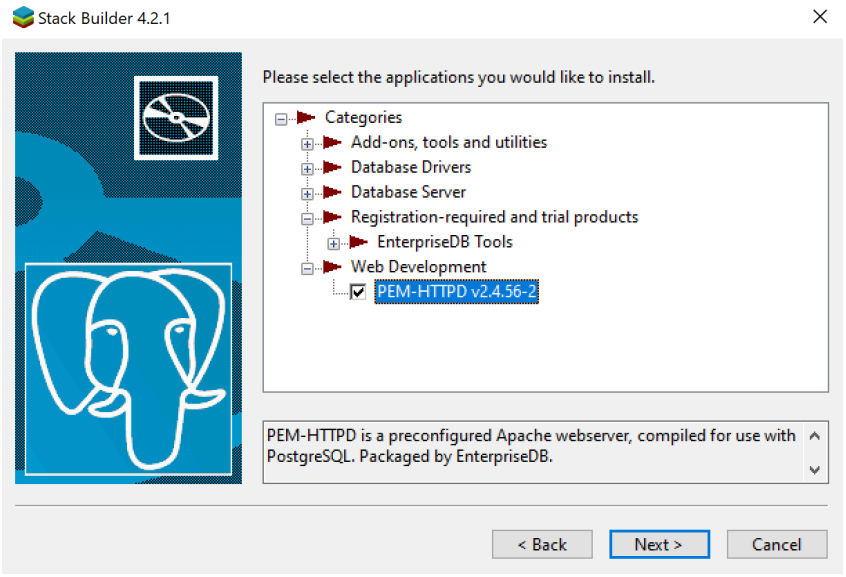


図 1: PEM-HTTPD in Stack Builder

3.3.7 Moving the PEM server

PEM サーバーをあるホストマシンから新しいホストマシンに移動できます。新しいホストターゲット上の PEM サーバーは、元のホストソースと同じバージョンの PEM サーバーインストーラーでインストールする必要があります。同じインストーラーバージョンを使用しない場合、スキーマ不一致エラーが発生する場合があります。

ターゲットサーバーのバックエンドデータベース PostgreSQL または EDB Postgres Advanced Server は、ソース PEM サーバーのバックエンドデータベースと同じタイプとバージョン、または異なるタイプとバージョンを持つことができます。PostgreSQL ホスト上にある PEM サーバーは、 EDB Postgres Advanced Server ホストに移行できます。その逆も可能です。

サーバーの移行を開始する前に、ソースホスト、ターゲットホスト、および PEM エージェントのホスト間のファイアウォールがサービス間の接続を許可していることを確認してください。

1. ターゲットホストを準備します。

ターゲットホストで PEM サーバーのインストーラーを起動します。ソース PEM サーバーのインストール時に使用したものと同じバージョンの PEM サーバーインストーラーを使用する必要があります。

ターゲットサーバーのバックエンドデータベースは、ソースのバックエンドデータベースとは異なるバージョンまたはタイプを持つことができます。新しい PEM サーバーが元のサーバーと同じタイプのバックエンドデータベースにない場合、新しいバージョンの sslutils 拡張機能が新しいサーバーホストにインストールされていることを確認する必要があります。PEM インストーラーと共に配布される sslutils のバージョンは、 EDB website から無料でダウンロードできます。

PEM サーバーまたは sslutils 拡張機能のインストールについては、PEM installation steps を参照してください。

2. 新しい PEM サーバーから既存のスキーマを削除します。

移行プロセスは、ターゲット PEM サーバー上のソース PEM サーバーから `pem`、`pemdata`、および `pemhistory` スキーマを再作成します。移動の準備をするには、psql クライアントを使用して、ターゲットホスト上の `pem`

データベースからこれらのスキーマを削除します。コマンドラインで、または **Postgres Enterprise Manager > SQL Shell (psql)** を選択して、psql クライアントを開くことができます。

psql クライアントが開いたら、データベーススーパーユーザーとして pem バックエンドデータベースに接続します。ターゲットホスト上の pem データベースに接続した後、スキーマを削除します。

```
DROP SCHEMA pem CASCADE;

DROP SCHEMA pemdata CASCADE;

DROP SCHEMA pemhistory CASCADE;
```

スキーマを削除するときは、CASCADE キーワードを含め、すべての依存オブジェクトを削除するようにサーバーに指示する必要があります。コマンドを実行すると、psql クライアントは依存オブジェクトのリストを表示します。クライアントは、DROP SCHEMA を表示することにより、各スキーマが削除されることを確認します。

3. 新しい PEM サーバーで PEM エージェントを準備します。

PEM サーバーを移動する前に、ソース PEM サーバーによって監視されているエージェントの数を特定し、ターゲットサーバーでその数のエージェント 1 つ未満の ID を作成する必要があります。PEM サーバーによって監視される PEM エージェントの合計数を検出するには、psql クライアントを使用してソースホスト上の pem データベースに接続し、pem.agent テーブルを照会します。

```
SELECT id FROM pem.agent WHERE active = true;
```

元の PEM サーバーにあるエージェントの数から 1 つ未満を手動で作成する必要があります。PEM サーバーインストーラーは、ターゲットホストに 1 つのエージェントを作成します。たとえば、ソースサーバーに 3 つのエージェントが含まれている場合、さらに 2 つのエージェントを手動で作成する必要があります。ターゲットサーバー上の pem データベースとの psql セッションを開き、必要なエージェントを作成します。

```
CREATE USER agent<X>;
```

<X> はエージェント番号を指定します。 agent1 は、PEM サーバーインストーラーによってターゲットホストに作成されます。

次に、 GRANT コマンドを使用して、ターゲット PEM サーバーにある各エージェントに pem_agent 権限を割り当てます。

```
GRANT pem_agent TO agent<X>;
```

<X> はエージェント番号を指定します。

4. ソース PEM サーバーのバックアップスクリプトを生成します。

pg_dump ユーティリティを使用して、ターゲットホストで pem データベースを再作成するために必要なコマンドを含むスクリプトを生成できます。デフォルトでは、pg_dump は Postgres インストールの下の bin ディレクトリにインストールされます。 pg_dump を呼び出すには、bin ディレクトリで次のように入力します。

```
pg_dump -U <user_name> <db_name> > <file_name>
```

そこで

- <user_name> は、PEM バックエンドデータベースのデータベーススーパーユーザーの名前を指定します。- <db_name> は、PEM バックエンドデータベースの名前を指定します。 - <file_name> は、pg_dump によって生成されたスクリプトの名前を指定します。

プロンプトが表示されたら、指定したユーザーに関連付けられたパスワードを入力します。

表示されているコマンドは、 pg_dump に実行時に pem データベースを再作成するスクリプトを生成するように指示します。スクリプトの名前は backup.sql で、 tmp ディレクトリに作成されます。pg_dump は、ユーザー postgres の資格情報を使用してサーバーに接続しています。

pg_dump ユーティリティを呼び出しても、現在のデータベースユーザーは中断されません。

注釈: ソース PEM サーバーが 7.16 バージョンより前の場合、 pg_dump を実行してバックアップを取る前に、次のファンクションを置き換える必要があります。

- abstime 、reltime 、および tinterval データ型は、Postgres バージョン 12 以降から非推奨です。これらのデータタイプを timestamptz データ型に置き換えるには、次のコマンドを使用します。

```
DO
$$
DECLARE
    rec record;
    cnt integer;
BEGIN
    -- Check for the deprecated type in our user info probe
    SELECT count(*) INTO cnt
    FROM pem.probe_column
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
      IF cnt = 0 THEN
        RETURN;
      END IF;
    ALTER TABLE pemdata.user_info
      ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    ALTER TABLE pemhistory.user_info
      ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    -- Now update the pem.probe_column itself
    UPDATE pem.probe_column
    SET sql_data_type = 'timestamptz'
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
END;
$$ LANGUAGE 'plpgsql';
```

- アラートエラーを回避するようにこのファンクションを置き換えます。

```
      CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
      template_id pem.alert_template.id%type, params text[]
      )
      RETURNS bool AS $FUNC$
      DECLARE
        res bool := TRUE;
      BEGIN
        /*
- During restoring the pem database, it does not maintain the order while
- inserting data in the table, and uses the sort table based on the
- names.
- Hence - we need to check the foreign key constraint is present before
- validating these values.
         */
       IF EXISTS(
        SELECT 1 FROM information_schema.table_constraints
        WHERE constraint_name=alert_template_id_fkey AND
        table_name=alert AND table_schema=pem
       ) THEN
        /*
- Need to use the IS TRUE construct outside the main query, because
- otherwise if theres no template by that ID then the query would return
- 0 rows and the result of the function would be undefined and CHECK
- constraint would succeed.
- Probably this is being over-cautious, because pem.alert.template_id
- references pem.alert_template.id. But the SQL standard (probably) does
- not define the order in which the CHECK or the FOREIGN KEY constraints
- should be validated; in case CHECK is validated first, we want it to
- fail.
         */
      EXECUTE $SQL$
          SELECT (
              SELECT pem.check_array_size_equal(t.param_names, $2)
              FROM pem.alert_template AS t
              WHERE id = $1
          ) IS TRUE
        $SQL$ INTO res USING template_id, params;
      END IF;
       RETURN res;
      END
      $FUNC$ LANGUAGE plpgsql;
```

5. バックアップをターゲットホストに移動します。

pg_dump ユーティリティによって生成されたスクリプトを PEM サーバーのターゲットホストに移動します。

6. ターゲットホストでバックアップを復元します。

ターゲットホストで、Postgres バックエンドデータベースインストールディレクトリの下の bin ディレクトリで、psql 起動し、pg_dump ユーティリティによって生成されたスクリプトを実行します。

```
psql -U <user_name> -d pem -f <file_name>
```

そこで

- <user_name> は、データベースのスーパーユーザーの名前を指定します。指定されたユーザーは、バックエンドデータベースに対する接続特権を持っている必要があります。- <file_name> は、pg_dump によって生成されたバックアップスクリプトへの完全なパスを指定します。

プロンプトが表示されたら、データベースのスーパーユーザーに関連付けられたパスワードを入力します。

この例では、 psql クライアントを使用して backup.sql という名前のスクリプトを呼び出し、 pem データベースを再作成します。スクリプトは、データベーススーパーユーザー postgres に関連付けられた特権を使用して呼び出されます。

7. ターゲットホストでデータベースサーバーを停止します。

RHEL 8.x で PEM サーバーを停止するには、次のコマンドを使用します。

```
systemctl stop <service_name>
```

<service_name> は、バックエンドデータベースサーバーの名前を指定します。PostgreSQL バックエンドデータベースの場合、サービス名は postgresql-<x> です。 EDB Postgres Advanced Server バックエンドデータベースの場合、サービス名は edb-as-<X> で、<X> はバージョン番号を指定します。

Windows を使用している場合は、[Services] ダイアログボックスを使用して Services を制御できます。 [Services] ダイアログボックスを開くには、Control Panel から **System and Security > Administrative Tools** を選択します。 **Services** アイコンをダブルクリックします。 [Services] ダイアログボックスで、リストからサービス名を選択し、[**Stop**] を選択します。

8. 証明書ファイルをターゲットホストにコピーします。

ターゲットホストのインストール時に作成される証明書ファイルを、ソースホストの証明書ファイルに置き換える必要があります。次のファイルをソース PEM サーバーからターゲット PEM サーバーにコピーします。

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

ターゲットクラスタのバックエンドデータベースを提供する Postgres インストールの下の data ディレクトリにファイルをコピーします。

Linux では、ファイルは次の場所にあります。

```
/var/lib/pgsql/<X>/data/
```

Windows では、ファイルは次の場所にあります。

```
C:\Program Files\PostgreSQL\<X>\data
```

そこで

<X> は、システム上の PostgresSQL のバージョンを指定します。

ファイルは既にターゲットクラスタに存在します。コピーを実行する前に既存のファイルを削除するか、既存のファイルをソースサーバーのファイルで上書きします。ターゲットサーバーに配置されたら、ファイルには、表示されているプラットフォーム固有の権限が必要です。

Linux の場合

File name	Owner	Permissions
ca_certificate.crt	postgres	-rw-------
ca_key.key	postgres	-rw-------
root.crt	postgres	-rw-------
root.crl	postgres	-rw-------
server.key	postgres	-rw-------
server.crt	postgres	-rw-r--r--

Linux では、証明書ファイルは postgres が所有する必要があります。次のコマンドを使用して、ファイルの所有権を変更します。

```
chown postgres <file_name>
```

`file_name` は、証明書ファイルの名前を指定します。

`server.crt` ファイルの所有者のみがファイルを変更できますが、すべてのユーザーが読み取ることができます。次のコマンドを使用して、`server.crt` ファイルのファイル許可を設定します。

```
chmod 644 server.crt
```

他の証明書ファイルの所有者のみがファイルを変更または読み取ることができます。次のコマンドを使用して、ファイルのアクセス許可を設定します。

```
chmod 600 <file_name>
```

`file_name` は、ファイルの名前を指定します。

**Windows の場合*

Windows では、ターゲットホストで PEM サーバーとバックエンドデータベースのインストールを実行したサービスアカウントは、ソースホストから移動した証明書ファイルを所有している必要があります。インストーラーコンテキストメニューから **Run as Administrator** を使用して PEM サーバーと Postgres インストーラーを呼び出した場合、証明書ファイルの所有者は Administrators です。

Windows でファイルのアクセス許可を確認および変更するには、ファイル名を右クリックし、**プロパティ**を選択します。[**Security**] タブで、グループまたはユーザー名を選択して、割り当てられたアクセス許可を表示します。**Edit** または **Advanced** を選択して、選択したユーザーに関連付けられている権限を変更できるダイアログボックスを開きます。

9. PEM エージェント証明書ファイルを PEM サーバーホストに移動します。

ソース PEM サーバーの PEM エージェントが使用する証明書ファイルをターゲットホストに移動する必要があります。この手順はプラットフォーム固有です。

Linux の場合

agent1.key および agent1.crt ファイルをソースホストからターゲットホストにコピーします。デフォルトでは、Linux では、ファイルは/root/.pem にインストールされます。ファイルをターゲットホスト上の同じディレクトリにコピーします。

ファイルの所有権とファイルのアクセス許可を次のように設定する必要があります。

File name	Owner	Permissions
agent1.key	root	-rw-------
agent1.crt	root	-rw-r--r--

必要に応じて、/root/.pem に移動し、次のコマンドを使用して agent1.key ファイルの権限と所有権を変更します。

```
chmod 600 agent1.key

chown root agent1.key
```

次のコマンドを使用して、agent1.crt ファイルの権限と所有権を変更します。

```
chmod 644 agent1.crt

chown root agent1.crt
```

**Windows の場合*

agent1.key および agent1.crt ファイルをソースホストからターゲットホストにコピーします。Windows では、ファイルは次の場所にあります。

```
C:\Users\<user_name>\AppData\Roaming\pem
```

user_name は、PEM インストーラーを呼び出したユーザーの名前です。

ターゲットマシン上の証明書ファイルに関連付けられている所有権と権限は、ソースマシン上の証明書ファイルの所有権と権限と一致する必要があります。インストーラーコンテキストメニューの **Run as Administrator** を使用して PEM サーバーと Postgres インストーラーを呼び出した場合、エージェント証明書ファイルの所有者は Administrators です。

Windows でファイルのアクセス許可を確認および変更するには、ファイル名を右クリックし、**プロパティ**を選択します。 [**Security**] タブで、グループまたはユーザー名を選択して、割り当てられたアクセス許可を表示します。**Edit** または **Advanced** を選択して、選択したユーザーに関連付けられている権限を変更できるダイアログボックスを開きます。

10. ターゲットホスト上の pg_hba.conf ファイルを更新します。

ターゲットホスト上の pg_hba.conf ファイルを変更して、各 PEM エージェントからの接続を許可します。デフォルトでは、pg_hba.conf ファイルは Postgres インストールの下のデータディレクトリにあります。

11. ターゲットホストでサーバーを起動します。

pg_hba.conf ファイルを変更した後、サーバーを再起動して変更を有効にする必要があります。

Linux でデータベースサーバーを再起動するには、次のコマンドを使用します。

```
/etc/init.d/<service_name> start
```

service_name は、バックエンドデータベースサーバーの名前です。

Windows、[Services] ダイアログボックスを使用して Services を制御できます。 [Services] ダイアログボックスを開くには、Control Panel で **System and Security > Administrative Tools** を選択します。 **Services** アイコンをダブルクリックします。 [Services] ダイアログボックスが開いたら、リストからサービス名を選択し、サービスを開始します。

12. 監視対象エージェントを新しい PEM サーバーホストに接続します。

既存の PEM エージェントに新しい PEM サーバーホストに接続するよう指示するには、以下を行う必要があります。

- PEM エージェントホストが新しい PEM サーバーホストに接続できることを確認します。
- PEM エージェントまたはエージェント構成ファイルを使用して各 Windows ホストのレジストリを変更する PEM エージェントを使用する各 Linux ホストで、新しい PEM サーバーの IP アドレスとポートを指定します。
- PEM エージェントのサービスを再起動します。これらの手順はプラットフォーム固有です。
  - *PEM* **エージェントが** *Linux* **にある場合**
  - *PEM* **エージェントが** *Windows* **にある場合**

PEM エージェントが Linux にある場合

エディターの選択を使用して、 agent.cfg ファイルを変更し、pem_host および pem_port パラメーターで PEM サーバーの新しい IP アドレスとポート番号を指定します。

デフォルトでは、 agent.cfg ファイルは次の場所にあります。

/usr/edb/pem/agent/etc/agent.cfg

agent.cfg ファイルを変更した後、PEM エージェントサービスを再起動する必要があります。Linux コマンドラインで pemagent サービススクリプトを使用して、サービスをリスタートできます。

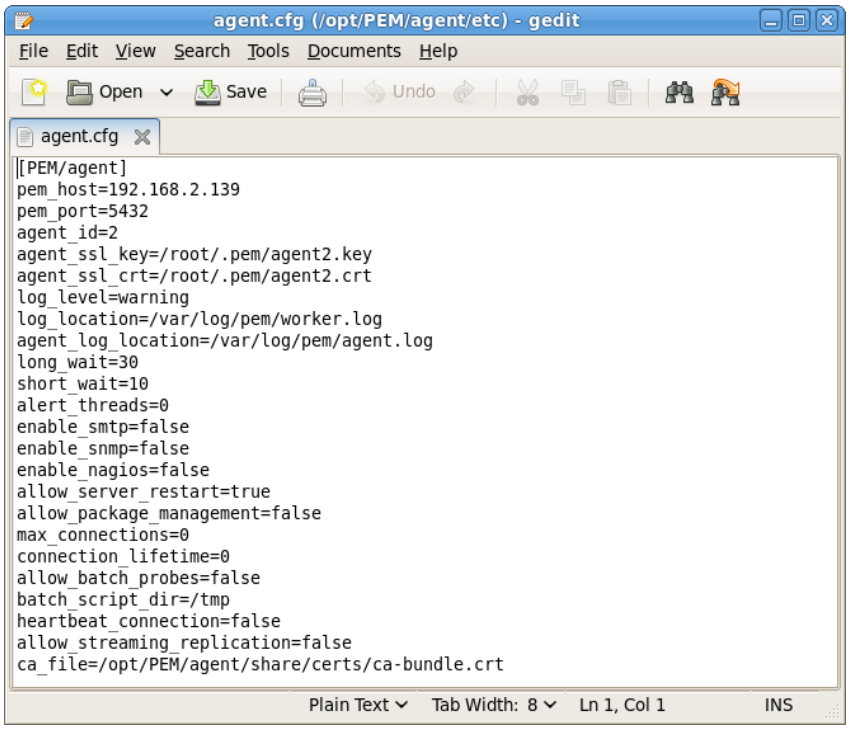


図 2: The agent.cfg file

```
/etc/init.d/pemagent restart
```

PEM エージェントが Windows にある場合

監視対象ノードの Windows レジストリを変更する前に、PEM エージェントのホスト上のファイアウォールが PEM サーバーへの接続を許可していることを確認してください。PEM エージェントホストが PEM サーバーホストに接続できることを確認した後、Windows Registry Editor を使用して PEM_HOST および PEM_PORT エントリを確認および編集し、PEM サーバーが使用するホストおよびポートを正しく識別できることを確認できます。Registry Editor を開くには、 Windows の [Run] ダイアログボックスまたは Windows の [スタート] メニューの検索ボックスに regedit と入力します。レジストリツリーコントロールをナビゲートして、レジストリエントリを表示または変更します。

PEM エージェントのレジストリエントリは次の場所にあります。HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent

PEM_HOST および PEM_PORT エントリは、ターゲットホスト上の新しい PEM サーバーのアドレスとポート番号を指定する必要があります。レジストリエントリを変更するには、エントリ名を右クリックし、コンテキストメニューから **Modify** を選択します。次に、[Edit String の Edit String] ダイアログボックスを使用して、エントリの値を変更します。完了したら、[**OK**] を選択します。

レジストリを変更した後、PEM エージェントのサービスを再起動する必要があります。Windows コントロールパネルからアクセスする [Services] ダイアログボックスを使用して、Postgres Enterprise Manager - pemagent サービスを再起動できます。

サーバーを稼働した後、インストールされている PEM クライアントの接続プロパティを変更して、PEM

サーバー、エージェント、監視対象サーバーの新しいホストに接続します。

注釈: サーバーを稼働した後、PEM Web インタフェースでサーバーを作成中にこのエラーが発生した場合

```
Error - User does not have enough permission to add new server.
        Please contact the administrator to grant pem_database_server_registration␣
↪role
        to the enterprisedb user.
```

ロールを更新し、適切な権限を付与することにより、エラーを解決します。

```
UPDATE pem.roles SET rolid = pr.oid FROM pg_roles pr WHERE pr.rolname = pem_ ||␣
↪component;
```

3.4 Uninstalling

PEM コンポーネントをアンインストールするには、以下を参照してください。

- *Uninstalling Postgres Enterprise Manager components on Linux*
- *Uninstalling Postgres Enterprise Manager components on Windows*

3.4.1 Uninstalling Postgres Enterprise Manager components on Linux

PEM サーバーまたはエージェントのアンインストールのプロセスはプラットフォーム固有です。PEM サーバーのパッケージの名前は edb-pem-server で、PEM エージェントのパッケージの名前は edb-pem-agent です。

ホストから PEM サーバーパッケージをアンインストールしても、同じホストにインストールされている PEM エージェントパッケージはアンインストールされません。ただし、PEM エージェントパッケージをアンインストールすると、同じホストにインストールされている PEM サーバーパッケージもアンインストールされます。

注釈: PEM エージェントをアンインストールする前に、最初にエージェントの登録を解除する必要があります。 *pemworker* コマンドラインユーティリティを使用して、エージェントの登録を解除できます。その後、アンインストール手順を続行できます。

RHEL、Rocky Linux、または AlmaLinux ホストからの PEM コンポーネントのアンインストール

rpm コマンドのバリエーションを使用して、Rocky Linux/AlmaLinux/RHEL 8.x ホストからインストールされたパッケージを削除できます。また、dnf remove コマンドを使用して、インストールされたパッケージを Rocky Linux/AlmaLinux/RHEL 8.x ホストから削除することもできます。パッケージを削除しても PEM データディレクトリは損傷しないことに注意してください。

rpm コマンドを呼び出して、インストールされたパッケージを削除するときに-e オプションを含めます。コマンド構文は次のとおりです。

```
rpm -e package_name
```

package_name は、削除するパッケージの名前です。

- dnf remove コマンドを使用して、pem サーバーまたはエージェントと、Rocky Linux または AlmaLinux または RHEL 8.x ホスト上の edb-pem および edb-pem-docs 依存関係を削除できます。パッケージを削除するには、ターミナルウィンドウを開き、スーパーユーザー権限を引き受けて、次のコマンドを入力します。

```
dnf remove package_name
```

Debian または Ubuntu ホストからの PEM コンポーネントのアンインストール

apt-get remove または apt-get purge コマンドを使用して、Debian または Ubuntu ホストから PEM サーバーまたはエージェントパッケージをアンインストールできます。

- 構成ファイルやデータディレクトリに影響を与えることなく、Debian または Ubuntu ホストから PEM サーバーまたはエージェントをアンインストールするには、次のコマンドを呼び出します。

```
apt-get remove package_name
```

- PEM サーバーまたはエージェントを構成ファイルおよびデータディレクトリとともにアンインストールするには、次のコマンドを呼び出します。

```
apt-get purge package_name
```

package_name は、削除するパッケージの名前です。

SLES ホストからの PEM コンポーネントのアンインストール

SLES ホストから PEM サーバーまたはエージェントをアンインストールするには、次のコマンドを呼び出します。

```
zypper remove package_name
```

package_name は、削除するパッケージの名前です。

3.4.2 Uninstalling Postgres Enterprise Manager components on Windows

ホストから PEM サーバーをアンインストールすると、同じホストにインストールされている PEM エージェントがアンインストールされます。ただし、PEM エージェントをアンインストールしても、同じホストにインストールされている PEM サーバーはアンインストールされません。

Windows Add/Remove Programs アプリケーションを使用して、Windows ホストから PEM コンポーネントを削除できます。 Windows Control Panel から Add/Remove Programs オプションを選択します。control panel が開いたら、プログラムリストで PEM コンポーネントの名前を見つけます。Remove ボタンをクリックして、コンポーネントを削除します。

次の場所にあるアンインストーラーを呼び出すこともできます。

PEM サーバーの場合、C:\Program Files\edb\pem\server\uninstall-pemserver

PEM エージェントの場合 C:\Program Files\edb\pem\agent\uninstall-pemagent

3.5 Troubleshooting

3.5.1 サーバーインストールエラー

RHEL 8

1. RHEL 8 に PEM サーバーのインストール中に、次のエラーが表示された場合

```
[root@etpgxlt firstuser]# dnf install edb-pem
Updating Subscription Management repositories.
Last metadata expiration check: 0:01:33 ago on Wed 30 Mar 2022 01:28:16 AM EDT.
Error:
 Problem: problem with installed package python3-mod_wsgi-4.6.4-4.el8.s390x
  - package python39-mod_wsgi-4.7.1-4.module+el8.4.0+9822+20bf1249.s390x conflicts␣
↪with python3-mod_wsgi provided by python3-mod_wsgi-4.6.4-4.el8.s390x
  - package python39-mod_wsgi-4.7.1-4.module+el8.4.0+9822+20bf1249.s390x conflicts␣
↪with python3-mod_wsgi provided by python3-mod_wsgi-4.6.4-3.el8.s390x
  - package edb-pem-server-8.4.0-7.rhel8.s390x requires python39-mod_wsgi >= 4.7, but␣
↪none of the providers can be installed
  - package edb-pem-8.4.0-7.rhel8.s390x requires edb-pem-server = 8.4.0-7.rhel8, but␣
↪none of the providers can be installed
  - cannot install the best candidate for the job
(try to add --allowerasing to command line to replace conflicting packages or --skip-
↪broken to skip uninstallable packages or --nobest to use not only best candidate␣
↪packages)
[root@etpgxlt firstuser]#
```

最初に python3-mod_wsgi パッケージを削除します。

```
dnf remove python3-mod_wsgi
```

PEM サーバーを再度インストールしてみてください。

2. RHEL 8 で、PEM サーバーの構成後にワーカー.log にこのエラーが表示された場合

```
Tue Nov 28 03:02:19 2023 WARNING: Error clearing zombies: ERROR:  failed to JIT␣
↪module: Added modules have incompatible data layouts: E-m:e-i1:8:16-i8:8:16-i64:64-
↪f128:64-a:8:16-n32:64 (module) vs E-m:e-i1:8:16-i8:8:16-i64:64-f128:64-v128:64-
↪a:8:16-n32:64 (jit)
CONTEXT:  SQL statement "WITH running_agent_job AS (
        SELECT j.jobid, j.agent_id
        FROM pem.job j LEFT JOIN pem.joblog jl ON (j.jobid = jl.jlgjobid)
        WHERE jl.jlgstatus = r AND agent_id = $1 AND j.jobarid != agent_runtime_id
        ORDER BY j.jobid, jl.jlgjobid
        FOR UPDATE SKIP LOCKED
    ), joblog_status_update AS (
        UPDATE pem.joblog jl SET jlgstatus=d
        FROM running_agent_job r
        WHERE r.jobid = jl.jlgjobid AND jl.jlgstatus=r
        RETURNING r.agent_id, jl.jlgjobid, jl.jlgid
    ), jobsteplog_status_update AS (
        UPDATE pem.jobsteplog js SET jslstatus=d
        FROM joblog_status_update jl
        WHERE js.jsljlgid = jl.jlgid AND js.jslstatus=r
        RETURNING jl.agent_id, jl.jlgjobid AS job_id
    )
    UPDATE pem.job j SET jobprocessid=NULL, jobnextrun=NULL, jobarid=NULL
    FROM (SELECT DISTINCT agent_id, job_id FROM jobsteplog_status_update) js
    WHERE js.job_id = j.jobid"
PL/pgSQL function pem.clear_job_zombies(integer) line 8 at SQL statement
```

エラーを解決するには、バックエンドデータベースサーバーの `postgresql.conf` ファイルで `jit` パラメーターを `off` に設定します。

```
jit=off
```

バックエンドデータベースサーバーを再起動します。

3.5.2 PEM サーバーの再構成

状況によっては、PEM サーバーのアンインストール、再インストール、およびサーバーの再構成が必要になる場合があります。そのためには

1. PEM サーバー構成を削除し、アンインストールします。

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

2. PEM パッケージを削除します。

```
yum erase edb-pem-server
```

3. pem データベースを削除します。

```
DROP DATABASE pem
```

4. 証明書を/root/.pem/ から別の場所に移動します。

```
mv /root/.pem/* <new_location>
```

5. agent.cfg ファイルを/usr/edb/pem/agent/etc/agent.cfg から別の場所に移動します。

```
mv /usr/edb/pem/agent/etc/agent.cfg <new_location>
```

6. 次に、PEM サーバーを再度構成します。

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

3.5.3 PEM Web クライアントがロードされません

PEM Web クライアントがロードされていない場合は、/var/log/httpd/error_log ファイルの HTTPD ログを確認します。

ログに次のメッセージが表示された場合

```
Truncated or oversized response headers received from daemon process edbpem: /usr/edb/
↪pem/web/pem.wsgi
```

/etc/httpd/conf フォルダーにある Apache httpd.conf ファイルの下部に次のステートメントを追加します。

```
WSGIApplicationGroup %{GLOBAL}
```

次のステートメントを追加した後、HTTPD サーバーを再起動します。

```
sudo systemctl restart httpd.service
``` ## PostgreSQL サーバーへの接続エラー

PostgreSQL サーバーに接続すると、次のいずれかのエラーメッセージが表示される場合があります。メッセージを注意深く確認します。各エラーメッセージは、問題を解決するために必要な情報を取り入れようとします。

- **サーバーへの接続が失われました** このエラーメッセージは、接続の試行が指定されたしきい値よりも時間がかかったことを示します。［サーバー］ダイアログボックスに表示される接続プロパティに問題があるか、ネットワーク接続の問題、またはサーバーが実行されていない可能性があります。

- **サーバーに接続できませんでした 接続は拒否されました** このエラーには 2 つの理由が考えられます。

  - データベースサーバーは実行されていません。サーバーを起動します。
  - サーバーは、表示されているアドレスで TCP/IP 要求を受け入れるように構成されていません。

セキュリティ上の理由から、PostgreSQL サーバーは「そのままでは」TCP/IP ポートでリッスンしません。代わりに、TCP/IP 要求をリッスンするように有効にする必要があります。バージョン 7.3.x および 7.4.x の `postgresql.conf` ファイルに `tcpip = true` を追加します。バージョン 8.0.x 以降に `listen_addresses='*'` を追加します。これらの追加により、サーバーは任意の IP インターフェイスでの接続を受け入れます。

詳細については、 `runtime configuration <http://www.postgresql.org/docs/current/
↪interactive/runtime-config.html>`_ に関する PostgreSQL ドキュメントを参照してください。

- **FATAL pg_hba.conf エントリー** 接続時に PEM がこのメッセージを表示する場合、ネットワーク経由でサーバーに正しく接続できますが、接続を受け入れるように構成されていません。クライアントは、データベースの合法的なユーザーとして検出されませんでした。

サーバーに接続するには、PEM クライアントのホストからの接続を受け入れるようにデータベースサーバー上の `pg_hba.conf` ファイルを構成します。データベースサーバホスト上の `pg_hba.conf` ファイルを変更し、次のフォームでエントリを追加します。

  - `host template1 postgres 192.168.0.0/24 md5` IPV4 ネットワークの場合
  - `host template1 postgres ::ffff:192.168.0.0/120 md5` IPV6 ネットワークの場合

詳細については、 `client authentication <http://www.postgresql.org/docs/current/
↪interactive/client-authentication.html>`_ に関する PostgreSQL ドキュメントを参照してください。

- **FATAL パスワード認証に失敗しました** `password authentication failed for user` エラーメッセージは、入力したパスワードに問題がある可能性を示しています。パスワードを再試行して、正しく入力したことを確認します。エラーメッセージが返された場合は、正しいパスワードを持っていること、サーバーへのアクセスが許可されていること、およびサーバーの `postgresql.conf` 構成ファイ

(次のページに続く)

(前のページからの続き)

ルでアクセスが正しく構成されていることを確認します。

PEM Web サーバーのステータスチェック

次のコマンドを実行して、PEM ウェブサーバーが稼働しているかどうかを確認します。

```shell
curl https://<SERVER_ADDR>:8443/pem/misc/ping -k -i
__OUTPUT__
PING
```

SERVER_ADDR は、PEM サーバーの IP アドレスです。出力 PING は、PEM Web サーバーが稼働していることを確認します。

第4章　Administering

4.1 Changing the default port

デフォルトでは、8443 ポートは、PEM サーバーの構成時に Web サービスに割り当てられます。Apache 構成ファイルのいくつかのパラメーターを変更することにより、構成後にポートを変更できます。Apache 構成ファイルの名前と場所はプラットフォーム固有です。

4.1.1 RHEL の場合

1. ファイル/etc/httpd/conf.d/edb-ssl-pem.conf を編集し、次のパラメーターの 8443 をポート番号に置き換えます。

```
Listen 8443
<VirtualHost _default_:8443>
ServerName localhost:8443
RewriteRule ^(.*)$ https://%{HTTP_HOST}:8443%{REQUEST_URI} [L,R=301]
```

1. semanage がインストールされていない場合は、インストールします。

```
sudo yum -y install policycoreutils-python
```

1. ポートがリストされているかどうかを確認します。

```
semanage port -l | grep http
```

1. ポートがリストに表示されない場合は、次のコマンドを実行します。

```
sudo semanage port -a -t http_port_t -p tcp <your_port_number>
```

1. httpd サービスを再起動します。

```
sudo systemctl restart httpd
```

ポートを使用して PEM Web インターフェイスにアクセスできるようになりました。詳細については、*Accessing the web interface* を参照してください。

4.1.2 Debian および Ubuntu の場合

1. ファイル/etc/apache2/sites-available/edb-ssl-pem.conf を編集し、次のパラメーターの 8443 をポート番号に置き換えます。

```
Listen 8443
<VirtualHost _default_:8443>
ServerName HTTPD-EDBPEM-SERVER-v8:8443
```

1. httpd サービスを再起動します。

```
sudo systemctl restart apache2
```

ポートを使用して PEM Web インターフェイスにアクセスできるようになりました。詳細については、*Accessing the web interface* を参照してください。

4.2 Registering a Postgres server

PEM でデータベースサーバーを管理またはモニタする前に、データベースサーバーを PEM に登録し、以前に登録した PEM エージェントにデータベースサーバーをバインドする必要があります。データベースサーバーをリモートエージェント別のホストにあるエージェントにバインドできます。ただし、エージェントが別のホストにある場合、インスタンスに関するすべての統計情報にアクセスできません。

4.2.1 Postgres サーバーを登録するとどうなりますか?

サーバーを登録すると、次の 2 つのことが行われます。

- PEM Web アプリケーションのサーバーツリーにサーバーを追加し、ユーザーが Web アプリケーションからサーバーへの接続を開くことができます。- サーバーを特定のエージェントにバインドします。つまり、エージェントがサーバーへの接続を開き、プローブとスケジュールされたジョブを実行します。

登録後、エージェントは指定されたデータベースへの接続を開きます。この接続を使用して、同じユーザーで接続できる他のデータベースを判断し、それぞれに 1 つの接続を生成します。

サーバーを登録するには 3 つの方法があります。

- バインドするエージェントが存在するホストのコマンドラインから *pemworker* **ユーティリティを使用してサーバーを登録する**- PEM Web アプリケーションの **自動サーバー検出へのサーバーの登録** を使用して、エージェントと同じホストにある Postgres サーバーを自動的に登録します。- **データベースサーバーの手動登録** PEM Web アプリケーションのすべての詳細を追加します。

4.2.2 pemworker ユーティリティを使用してサーバーを登録する

pemworker ユーティリティを使用して、Postgres サーバーを登録できます。登録時に、pemworker ユーティリティは、登録コマンドを呼び出したシステムにあるエージェントに新しいサーバーをバインドします。

Linux ホストでサーバーを登録するには、次のコマンドを使用します。

```
pemworker --register-server
```

Windows ホストにサーバーを登録するには、次のコマンドを使用します。

```
pemworker.exe REGISTER-SERVER
```

pemworker ユーティリティを呼び出すときに、コマンド文字列にコマンドラインオプションを追加します。各オプションの後には、対応する値を続けます。

Option	Description
--pem-user	サーバーの詳細を PEM データベースに書き込むために使用する PEM サーバー上の PEM 管理ユーザー pem_admin ロールが必要な名前を指定します。必須。
Server parameters	これらのパラメーターは、PEM Web アプリケーションに接続プロパティを設定するために使用されます。これらは、ユーザーが PEM Web アプリケーションから監視対象サーバーに接続するときに使用されます。これらは、エージェントサーバーバインディングパラメーターでオーバーライドされない限り、エージェントから監視対象サーバーへの接続にも使用されますこの表の **エージェントサーバーバインディングパラメーター** を参照してください。
--server-addr	監視対象サーバーの IP アドレスまたは完全修飾ドメイン名を指定します。Linux システムでは、アドレスフィールドを空白のままにして、ローカルマシン上のデフォルトの PostgreSQL Unix ドメインソケットを使用できます。または、PostgreSQL ソケットを含む代替パスに設定できます。パスを入力する場合、パスはスラッシュ/で始まる必要があります。必須。
--server-port	監視対象サーバーのポート番号を指定します。必須。
--server-database	接続する監視対象サーバー上のデータベースの名前を指定します。必須。
--server-user	監視対象サーバーへの接続に使用するユーザーの名前を指定します。必須。
--server-service-name	監視対象の Postgres サーバー、たとえば、Linux の systemd サービスユニットを管理するオペレーティングシステムサービスの名前を指定します。オプショナル。
Agent-server binding parameters	これらのパラメーターを使用して、サーバーパラメーターをオーバーライドします。 PEM Web アプリケーションで使用される資格情報とは異なる資格情報を使用してエージェントを監視対象サーバーに接続する場合、これらを使用します。
--asb-host-name	監視対象サーバーの IP アドレスまたは完全修飾ドメイン名を指定します。オプショナル。指定しない場合、デフォルトの `--server-addr`。
--asb-host-port	監視対象サーバーのポート番号を指定します。オプショナル。指定しない場合、デフォルトの `--server-port`。
--asb-host-db	エージェントが接続する監視対象サーバー上のデータベースの名前を指定します。オプショナル。指定しない場合、デフォルトの `--server-database`。

次のページに続く

表 1 – 前のページからの続き

Option	Description
--asb-host-user	エージェントが監視対象サーバーに接続するために使用するユーザーの名前を指定します。オプショナル。指定しない場合、デフォルトの`--server-user`。
--asb-ssl-mode	接続に使用する SSL 認証のタイプを指定します。サポートされている値には、*prefer*、*require*、*disable*、*verify-CA*、および`verify-full`が含まれます。オプション、デフォルトは`prefer`です。
Server metadata	これらのパラメーターは、PEM Web アプリケーションでサーバーがどのように表示されるかを決定します。
--group	サーバーが表示されるグループの名前を指定します。オプショナル。
--team	監視対象サーバーが属するチームとして割り当てられる PEM サーバー上の Postgres ロールを指定します。このロールを持つユーザーのみがサーバーにアクセスできます。オプショナル。デフォルトは none で、すべてのユーザーがアクセスできます。
--owner	監視対象サーバーの所有者として割り当てる PEM サーバー上の Postgres ユーザーを指定します。オプション、デフォルトは`--pem-user`です。
--display-name <name>	監視対象データベースサーバーの表示名を指定します。オプショナル、デフォルトはシステムホスト名です。
Other parameters	NA
--remote-monitoring	サーバーがエージェントと同じホストにない場合、*yes* **に設定します。リモート監視が有効になっている** *`yes*、監視対象サーバーのエージェントレベルの統計はカスタムグラフとダッシュボードで利用できず、リモートサーバーは一部の PEM ユーティリティ Audit Manager、Capacity Manager、Log Manager などからアクセスできません。 、Postgres Expert、および Tuning Wizard)。オプション、デフォルトは`no`です。
--efm-cluster-name	サーバーを監視する EDB Failover Manager クラスターの名前を指定します該当する場合。オプショナル。
--efm-install-path	EDB Failover Manager のインストールディレクトリへの完全なパスを指定します該当する場合。オプショナル。
--config-dir	エージェント構成ファイルのディレクトリパスを指定します。オプション、デフォルトは`<pemworker path>/../etc`です。

環境変数 `PEM_SERVER_PASSWORD` を設定して PEM サーバーのパスワードを提供し、pemworker が PEM 管理ユーザーとして接続できるようにします。

環境変数 `PEM_MONITORED_SERVER_PASSWORD` を設定して、PEM エージェントによって登録および監視されるデータベースサーバーのパスワードを提供します。

パスワードを入力しない場合、パスワード認証エラーが発生します。PEM サーバーは、サーバーが適切に登録されたことを確認します。

例

この例では、必要なパラメーターのみを使用してサーバーを登録します。ユーザー admin01 は、PEM サーバーに接続するために使用されます。--server-* パラメーターに指定された資格情報は、エージェントから監視対象サーバーへの接続を作成するために使用されます。同じ詳細は、PEM Web アプリケーションにサーバー接続の詳細を入力するために使用されます。

```
pemworker --register-server \
          --pem-user admin01 \
          --server-addr pg123.prod.infra.business \
          --server-port 5432 \
          --server-database postgres \
          --server-user prod_admin
```

次の例では、エージェントから監視対象サーバーへの接続に使用するさまざまなパラメーターを指定します。これらは、--server-addr で提供された完全修飾ドメイン名を代わりに localhost でオーバーライドします。また、ユーザーをオーバーライドして local_monitor ユーザーを使用します。これは、たとえば、同じホストからのみ接続を許可されたユーザーである場合があります。

この例では、サービス名 postgresql も指定します。これは、PEM エージェントがこのサーバーを再起動して構成変更を適用できることを意味します。

shell pemworker --register-server \ --pem-user admin01 \ --server-addr pg123.prod.infra.business \ --server-port 5432 \ --server-database postgres \ --server-user prod_admin \ --server-service-name postgresql \ --asb-host-name localhost \ --asb-host-user local_monitor ### pemworker ユーティリティを使用してサーバーの登録を解除する

pemworker ユーティリティを使用して、データベースサーバーの登録を解除できます。サーバーの登録を解除するには、pemworker ユーティリティを呼び出します。

Linux ホストでは、次のコマンドを使用します。

```
pemworker --unregister-server
```

Windows ホストでは、次のコマンドを使用します。

```
pemworker.exe UNREGISTER-SERVER
```

pemworker ユーティリティを呼び出すときに、コマンド文字列にコマンドラインオプションを追加します。各オプションの後には、対応する値を入力します。

Option	Description
--pem-user	PEM 管理ユーザーの名前を指定します。必須。
--server-addr	サーバーホストの IP アドレスまたは完全修飾ドメイン名を指定します。 Unix ベースのシステムでは、アドレスフィールドを空白のままにして、ローカルマシンでデフォルトの PostgreSQL Unix ドメインソケットを使用できます。または、PostgreSQL ソケットを含む代替パスに設定できます。パスを入力する場合、パスはスラッシュ/で始まる必要があります。必須。

次のページに続く

表 2 – 前のページからの続き

Option	Description
--server-port	ホストのポート番号を指定します。必須。
--config-dir	エージェント構成ファイルのディレクトリパスを指定します。オプション、デフォルトは`<pemworker path>/../etc`です。

PEM_SERVER_PASSWORD 環境変数を使用して PEM サーバーのパスワードを提供し、pemworker ユーティリティが--pem-user として PEM サーバーに接続できるようにします。

パスワードを入力しない場合、パスワード認証エラーが発生します。PEM サーバーは、サーバーが登録解除されたことを確認します。

4.2.3 自動サーバー検出へのサーバーの登録

モニタするサーバーがモニタリングエージェントと同じホストにある場合は、[Auto Discovery] ダイアログボックスを使用して登録とバインドのプロセスを簡素化できます。

Auto Discovery ダイアログボックスを開くには、PEM クライアントツリーで PEM エージェントを選択し、**Management > Auto Discovery** を選択します。

[Auto Discovery] ダイアログボックスが開くと、[**Discovered Database Servers**] ボックスに、現在 PEM エージェントによって監視されていないサーバーのリストが表示されます。サーバー名の横にあるボックスを選択して、**Server Connection Details** ボックスにサーバーに関する情報を表示し、**Agent Connection Details** ボックスにエージェントの接続プロパティを表示します。

Check All を使用して、表示されているすべてのサーバーの横にあるボックスを選択するか、**Uncheck All** を使用してサーバー名の左側にあるすべてのボックスをクリアします。

Server Connection Details ボックスのフィールドは、PEM がモニタするサーバーに関する情報を提供します。

- **Name** フィールドの監視対象サーバーの名前を受け入れるか、変更します。指定した名前は PEM クライアントのツリーに表示されます。
- **Server group** リストを使用して、PEM クライアントツリーでサーバーが表示されるサーバーグループを選択します。
- **Host name/address** フィールドを使用して、監視対象サーバーの IP アドレスを指定します。
- **Port** フィールドには、サーバーによって監視されているポートが表示されます。このフィールドは変更できません。
- **Service ID** フィールドにサービスの名前を入力します。一部の PEM 機能を有効にするには、サービス名を提供する必要があります。
- **Maintenance Database** フィールドは、初期接続とグローバルメンテナンス操作に使用するデータベースを指定します。インストールの**メンテナンスデータベース** フィールドの内容をカスタマイズします。

Agent Connection Details ボックスのフィールドは、サーバーへの接続時に使用する PEM エージェントのプロパティを指定します。

- **Host** フィールドには、PEM エージェントのバインディングに使用される IP アドレスが表示されます。
- **User name** フィールドには、選択したサーバーに接続するときに PEM エージェントが使用する名前が表示されます。
- **Password** フィールドには、指定されたユーザー名に関連付けられたパスワードが表示されます。
- **SSLmode** フィールドを使用して、SSL 接続設定を指定します。

モニタリングにバインドしているサーバーの接続プロパティの指定が終了したら、**OK** を選択してサーバーを登録します。

OK を選択すると、新しく登録されたサーバーが PEM ツリーに表示され、PEM サーバーによって監視されます。

4.2.4 データベースサーバーの手動登録

PEM でデータベースサーバーを管理またはモニタするには、次のことを行う必要があります。

- EDB Postgres Advanced Server または PostgreSQL サーバーを PEM サーバーに登録します。
- データベースサーバーを PEM エージェントにバインドします。

Create Server ダイアログボックスを使用して、サーバーの登録情報を提供し、PEM エージェントをバインドし、PEM クライアントツリーにサーバーを表示できます。[Create Server] ダイアログボックスを開くには、 **Object > Create > Server** を選択します。

注釈:

- 接続する前に、登録している Postgres サーバーの `pg_hba.conf` ファイルで PEM クライアントのホストからの接続が許可されていることを確認する必要があります。 - データベーススーパーユーザーまたは`pem_admin` ロールを持つユーザーのみがデータベースサーバーを PEM エージェントにバインドできます。 **General**タブを使用して、サーバーの一般的なプロパティを説明します。

- **Name** フィールドを使用して、サーバーの名前を指定します。名前は、PEM ブラウザツリー内のサーバーを識別します。
- グループを使用して、ツリー内のサーバーとエージェントを整理できます。グループを使用すると、多数のサーバーをより簡単に管理できます。たとえば、実稼働グループ、テストグループ、または LAN 固有のグループを持つことができます。**Group** リストを使用して、新しいサーバーを表示するサーバーグループを選択します。
- **Team** フィールドを使用して、PostgreSQL ロール名を指定します。このロールのメンバーである PEM ユーザー、最初にサーバーを作成した人、または PEM サーバーのスーパーユーザー権限を持つ PEM ユーザーのみが、PEM にログインするときにこのサーバーを表示します。このフィールドを空白のままにした場合、デフォルトでは、すべての PEM ユーザーにサーバーが表示されます。[Server Configuration] ダイアログボックスの `show_objects_with_no_team` パラメーターを使用して、動作を変更できます。`show_objects_with_no_team` が `false` に設定されている場合、チームのないサーバーは他のすべてのユーザーに表示されません。

- **Background** カラーセレクターを使用して、サーバーに保存されているデータベースオブジェクトの背後にある PEM ツリーに表示する色を選択します。
- **Foreground** カラーセレクターを使用して、サーバーに保存されているオブジェクトの PEM ツリーのラベルのフォント色を選択します。
- **Save** を選択するときに **Connect now?** を選択してサーバー接続を試行します。PEM クライアントが、後の接続試行まで指定された接続パラメーターを検証したくない場合は、**Connect now?** をクリアします。
- **Comments** フィールドにサーバーに関するメモを入力します。

Connection タブ を使用して、サーバーの接続の詳細を指定します。

- **Host name/address** フィールドに、サーバホストの IP アドレスまたは完全修飾ドメイン名を指定します。Unix ベースのシステムでは、アドレスフィールドを空白のままにして、ローカルマシンでデフォルトの PostgreSQL Unix ドメインソケットを使用します。または、PostgreSQL ソケットを含む代替パスを設定できます。パスを入力する場合、パスはスラッシュ/で始まる必要があります。
- **Port** フィールドにホストのポート番号を指定します。
- **Maintenance Database** フィールドを使用して、インストールされた pgAgent スキーマと adminpack オブジェクトを含むことが予想される接続する PEM の初期データベースの名前を指定します両方オプショナル。PostgreSQL バージョン 8.1 以降では、メンテナンス DB は通常 `postgres` と呼ばれます。以前のバージョンでは `template1` がよく使用されますが、テンプレートデータベースの混乱を避けるために `postgres` データベースを作成することをお勧めします。
- 監視対象サーバーで Kerberos 認証を使用するには、 **Kerberos Authentication** を **Yes** に設定します。デフォルトでは、監視対象サーバーは PEM サーバーと同じ認証方法を使用します。監視対象サーバーが Kerberos 認証を使用したくない場合は、`config_local.py` ファイルで `ALLOW_DATABASE_CONNECTION_WITHOUT_KERBEROS` から `TRUE` を設定します。
- **Username** フィールドにサーバーで認証するときに使用する名前を指定します。
- 指定されたユーザーに関連付けられたパスワードを **Password** フィールドに入力します。
- 後で再利用できるように PEM バックエンドデータベースにパスワードを暗号化された形式で保存するには、**Savepassword?** を選択します。各パスワードはユーザーごと、サーバーごとに保存され、他のチームメンバーと共有されません。PEM は、保存されたパスワードを使用して、次回データベースサーバーに接続します。保存されたパスワードを削除するには、最初にデータベースサーバーを切断し、次に [**Object > Clear Saved Password] を選択します。
- **Role** フィールドを使用して、サーバーに接続した後にクライアントが使用する特権を割り当てられたロールの名前を指定します。この値により、1 つのロールとして接続し、接続が確立されたときに別のロールこのフィールドで指定したロールの権限を引き継ぐことができます。接続するロールは、指定されたロールのメンバーである必要があります。

SSL タブを使用して、SSL を構成します。

- **SSLmode** フィールドで、サーバーが使用する SSL 接続のタイプを選択します。SSL 暗号化の使用の詳細については、 the PostgreSQL documentation を参照してください。

プラットフォーム固有のファイルマネージャーダイアログボックスを使用して、SSL 暗号化をサポートするファイルをサーバーにアップロードできます。ファイルマネージャーを開くには、次の各フィールドの

右側にあるアイコンを選択します。

- **Clientcertificate** フィールドを使用して、クライアント SSL 証明書を含むファイルを指定します。このファイルは、PEM がデスクトップモードでインストールされている場合はデフォルトの~/.postgresql/postgresql.crt ファイルを置き換え、PEM が Web モードでインストールされている場合は<STORAGE_DIR>/<USERNAME>/.postgresql/postgresql.crt ファイルを置き換えます。SSL 接続が作成されない場合、このパラメーターは無視されます。
- **Clientcertificate key** フィールドを使用して、クライアント証明書に使用される秘密キーを含むファイルを指定します。このファイルは、PEM がデスクトップモードでインストールされている場合はデフォルトの~/.postgresql/postgresql.key を、PEM が Web モードでインストールされている場合は<STORAGE_DIR>/<USERNAME>/.postgresql/postgresql.key を置き換えます。SSL 接続が作成されない場合、このパラメーターは無視されます。
- **ルート証明書**フィールドを使用して、SSL 認証局を含むファイルを指定します。このファイルは、デフォルトの~/.postgresql/root.crt ファイルを置き換えます。SSL 接続が作成されない場合、このパラメーターは無視されます。
- **証明書失効リスト**フィールドを使用して、SSL 証明書失効リストを含むファイルを指定します。このリストは、~/.postgresql/root.crl にあるデフォルトリストを置き換えます。SSL 接続が作成されない場合、このパラメーターは無視されます。
- **SSLcompression?** が **True** に設定されている場合、SSL 接続を介して送信されるデータは圧縮されます。デフォルト値は **False** 圧縮は無効です。SSL 接続が作成されない場合、このパラメーターは無視されます。

> **警告:** 証明書、プライベート鍵、および失効リストは、PEM サーバープロセスが実行されるユーザアカウントが所有するサーバー上のユーザーごとのファイル格納領域に格納されます。これは、サーバーの管理者がこれらのファイルにアクセスできることを意味します。この機能を使用する前に注意してください。

SSH Tunnel タブを使用して、SSH トンネリングを構成します。トンネルを使用して、中間プロキシホストを介してデータベースサーバーを、クライアントが直接接続できないネットワーク上にあるサーバーに接続できます。

- 指定されたサーバーへの接続時に SSH トンネルを使用するには、 **Use SSH 確かにトンネリング** を **Yes** に設定します。
- **Tunnel host** フィールドに、SSH ホストクライアント接続が転送される名前または IP アドレスを指定します。
- クライアント接続が転送される SSH ホストのポートを **トンネルポート**フィールドに指定します。
- **Username** フィールドに、SSH ホストのログイン特権を持つユーザーの名前を指定します。
- **Authentication** フィールドに SSH ホストに接続するときに使用する認証の種類を指定します。
- SSH ホストへの認証にパスワードを使用する場合は、 **Password** を選択します。これがデフォルトです。
- 接続時にプライベートキーファイルを使用するには、 **Identity file** を選択します。

- SSH ホストが認証用にプライベートキーファイルを想定している場合、**Identity file** フィールドを使用して、キーファイルの場所を指定します。
- SSH ホストがパスワードを想定している場合は、**Password** フィールドを使用してパスワードを指定します。ID ファイルを使用している場合、パスフレーズを指定します。

Advanced タブを使用して、サーバーの管理に使用される詳細を指定します。

- **Host address** フィールドにサーバーホストの IP アドレスを指定します。
- **DB rotation** フィールドを使用して、 pg_database テーブルに対して使用する SQL 制限を指定して、ツリーに表示されるデータベースを制限します。たとえば、`live_db` および `test_db` データベースのみを表示するには、`'live_db'` , `'test_db'` と入力できます。データベースのプロパティダイアログボックスから pg_namespace に対する制限を入力して、データベースに表示されるスキーマを制限することもできます。
- **Password file** フィールドを使用して、パスワードファイル`.pgpass` の場所を指定します。 `.pgpass` ファイルを使用すると、ユーザーは接続時にパスワードを入力せずにログインできます。ファイルは PEM サーバーに存在する必要があります。詳細については、Postgres documentation を参照してください。
- **Service ID** フィールドを使用して、データベースサービスプロセスのパラメーターを指定します。Enterprise Manager ディレクトリに保存されているサーバーの場合、サービス ID を入力します。Windows マシンでは、これは Windows サービスの識別子です。Linux マシンでは、サーバーの起動に使用される init スクリプトの名前は`/etc/init.d` で、サーバーを起動する systemd スクリプトの名前は `systemctl` です。たとえば、 EDB Postgres Advanced Server 11 サービスの名前は `edb-as-11` です。ローカルサーバーの場合、設定はオペレーティングシステムに依存します。
  - PEM クライアントが Windows マシンで実行されている場合、十分なアクセス権限がある場合、postmaster サービスを制御できます。サービスの名前を入力します。リモートサーバーの場合、マシン名 `PSE1\pgsql-8.0` などを先頭に付けます。PEM は、ローカルマシンで実行されているサービスを自動的に検出します。
  - PEM クライアントが Linux マシンで実行されている場合、十分なアクセス権限があれば、ローカルマシンで実行されているプロセスを制御できます。`pg_ctl` プログラムにアクセスするためにフルパスと必要なオプションを提供します。サービス制御機能を実行すると、PEM は status/start/stop キーワードをこれに追加します。例

```
sudo /usr/pgsql-x/bin/pg_ctl -D /var/lib/pgsql/x/data
```

`x` は、PostgreSQL データベースサーバーのバージョンです。

- サーバーがフェールオーバーマネージャークラスターのメンバーである場合は、PEM を使用してクラスターの状態を監視し、必要に応じてプライマリノードを置き換えることができます。PEM がフェールオーバーマネージャーを監視できるようにするには、 **EFM cluster name** フィールドを使用してクラスター名を指定します。クラスター名は、フェールオーバーマネージャークラスタープロパティファイルの名前の接頭辞です。たとえば、クラスタープロパティファイルの名前が `efm.properties` の場合、クラスター名は efm です。
- PEM を使用してフェールオーバーマネージャークラスターのステータスをモニタいる場合は、**EFM installation path** フィールドを使用して、Failover Manager バイナリファイルの場所を指定します。デフォルトでは、Failover Manager バイナリファイルは`/usr/edb/efm-x.x/bin` にインストールされます。`x.x` は Failover Manager のバージョンを指定します。

[**PEM Agent**] タブを使用して、PEM エージェントの接続の詳細を指定します。

[**Connection Parameters**] タブで

- **Remote Monitoring?** を **Yes** に設定して、PEM エージェントが監視対象サーバーと同じホストにないことを示します。リモート監視が有効になっている場合、監視対象サーバーのエージェントレベルの統計はカスタムグラフとダッシュボードで利用できず、リモートサーバーは一部の PEM ユーティリティ Audit Manager、Capacity Manager、Log Manager、Postgres Expert などからアクセスできません。 、およびチューニングウィザード)。
- **Bound agent** リストを使用して、PEM エージェントを選択します。1 つのエージェントでマルチプルの Postgres サーバーをモニタできます。
- **Host** フィールドに、データベースサーバーに接続するときに使用するエージェントの IP アドレスまたはソケットパスを入力します。デフォルトでは、エージェントは [**General**] タブに表示されるホストアドレスを使用します。Unix サーバーでは、`/tmp` などのソケットパスを指定することができます。
- サーバーへの接続時に使用するエージェントのポート番号を入力します。デフォルトでは、エージェントは [**プロパティ**] タブで定義されたポートを使用します。
- **SSL** フィールドを使用して、SSL 操作モードを指定します。 **require** 、 **prefer** 、 **allow** 、 **disable** 、 **verify-ca** 、または **verify-full** を指定します。

Mode	Description
require	サーバーとエージェント間のトランザクションには SSL 暗号化が必要です。
prefer	SSL 暗号化が利用できる場合、サーバーとエージェント間で SSL 暗号化を使用します。
allow	サーバーで必要な場合、接続が SSL を使用できるようにします。
disable	エージェントとサーバー間の SSL 暗号化を無効にします。
verify-ca	SSL 暗号化を必要とし、認証局によって登録された証明書を使用してサーバーに認証を要求します。
verify-full	SSL 暗号化を必要とし、信頼できる認証局によって登録された証明書を使用してサーバーに認証を要求します。

SSL 暗号化の使用の詳細については、 PostgreSQL documentation を参照してください。

- **Server** フィールドを使用して、エージェントが最初に接続する Postgres データベースの名前を指定します。EDB Postgres Distributed データベースノードを登録する場合、このフィールドに EDB Postgres Distributed 対応データベース名を指定します。
- **User name** フィールドに、サーバーへの接続時に使用するエージェントのロールの名前を指定します。すべての機能が期待どおりに動作するには、指定されたロールがデータベーススーパーユーザーである必要があります。指定されたロールがデータベーススーパーユーザーでない場合、機能しない機能のリストについては、*Setting agent privileges* を参照してください。Kerberos 認証方法を使用している場合、このフィールドに pgd_monitor または pgd_superuser ロールを持つユーザーを指定します。

Postgres バージョン 10 以降を使用している場合は、pg_monitor ロールを使用して、必要な特権を非スーパーユーザーに付与できます。pg_monitor ロールの詳細については、 Default Roles を参照してください。

- **User name** フィールドに、サーバーへの接続時に使用するエージェントのユーザーの名前を指定し

ます。すべての機能が期待どおりに動作するには、指定されたロールがデータベーススーパーユーザーである必要があります。Postgres バージョン 10 以降を使用している場合は、pg_monitor ロールを使用して、必要な特権を非スーパーユーザーに付与できます。pg_monitor ロールの詳細については、 Default Roles を参照してください。

- サーバーに接続するときに使用するエージェントのパスワードを **Password** フィールドと **確認パスワード**フィールドに指定します。パスワードを指定しない場合、エージェントの認証を手動で構成する必要があります。たとえば、`.pgpass` ファイルは、エージェントがインストールされているシステムに存在し、アクセスできる必要があります。

[**Advanced**] タブで

- **Allow takeover?** を **Yes** に設定して、別のエージェントがサーバーを引き継ぎできることを指定します。この機能により、エージェントは、高可用性フェールオーバープロセスの一部としてサーバーが別のホストに移動した場合など、データベースサーバーの監視を引き継ぐことができます。
- プラス記号 **+** を使用して、PEM 監視から除外するデータベースを追加します。 [**PEM エージェント**] タブの [**Connection Parameters**] タブに記載されているデータベースを除外することはできません。

注釈: データベースレベルのプローブは、除外されたデータベースに対して実行されませんが、サーバーレベルのプローブはデータベース統計を収集できます。

BART タブの下の **General** タブを使用して、PEM サーバーにマップする BART サーバーの一般的なプロパティを説明します。

- **BART server** フィールドを使用して、BART サーバー名を選択します。PEM コンソールで構成されたすべての BART サーバーがリストされます。
- **Server name** フィールドを使用して、BART サーバーを使用してバックアップするデータベースサーバーの名前を指定します。この名前は、BART 構成ファイルに保存されます。
- **Backup name** フィールドを使用して、データベースサーバーのバックアップに割り当てるユーザー定義名のテンプレートを指定します。バックアップ名テンプレートを指定しない場合、BART が割り当てた整数バックアップ識別子を使用してのみ、BART サブコマンドでバックアップを参照できます。
- **Host address** フィールドを使用して、バックアップ用に構成するデータベースサーバーの IP アドレスを指定します。
- **Port** フィールドを使用して、バックアップするデータベースに使用するポートを指定します。
- **User** フィールドを使用して、PEM コンソールから BART を使用してバックアップするデータベースのユーザーを指定します。このデータベースサーバーのインクリメンタルバックアップを有効にする場合、ユーザーはスーパーユーザーである必要があります。
- **Password** フィールドを使用して、バックアップするデータベースのユーザーのパスワードを指定します。
- **Cluster Owner** フィールドを使用して、データベースクラスターを所有する Linux オペレーティングシステムのユーザーアカウントを指定します。これは通常、Oracle データベース互換モードでインストールされた EDB Postgres Advanced Server データベースクラスターの enterprisedb です。これ

は、通常、PostgreSQL データベースクラスター、および PostgreSQL データベース互換モードでインストールされた EDB Postgres Advanced Server データベースクラスターの postgres です。

- **Archive command** フィールドを使用して、`bart.cfg` ファイルで使用するアーカイブコマンド文字列の目的の形式を指定します。Archive コマンドに提供された入力は、データベースサーバーの `Postgresql.conf` ファイルを上書きします。サーバーが追加されると、データベースサーバーが再起動するか、データベース構成がリロードされます。
- **Allow attempts attempts?** を使用して、このデータベースサーバーのインクリメンタルバックアップを有効にするかどうかを指定します。
- **SetuppasswordlessSSH?** を使用して、SSH 証明書を作成して、データベースサーバーと BART サーバー間でパスワードなしのログインを許可するかどうかを指定します。パスワードなしの SSH 認証を設定する前に、必ず PEM エージェントをバインドしてください。パスワードレス SSH は、PEM エージェントによってリモートで監視されているデータベースサーバーでは機能しません。

BART タブの下の **Misc** タブを使用して、BART Server のその他のプロパティを説明します。

- **Overridedefault configuration?** を使用して、BART サーバー構成を特定のデータベースサーバー構成でオーバーライドします。
- **Xlog** メソッドを使用して、pg_basebackup の実行中にトランザクションログを収集する方法を指定します。
- **Retentionpolicy** フィールドを使用して、バックアップの保持ポリシーを指定します。これにより、アクティブなバックアップを古いものとして、したがって削除の候補としてマークする時期が決定されます。バックアップの数または期間日、週、または月の観点から保持ポリシーを指定できます。
- **WAL 圧縮** を使用して、アーカイブされた Xlog / WAL ファイルを gzip 形式で圧縮するかどうかを指定します。WAL 圧縮を有効にするには、gzip 圧縮プログラムが BART ユーザーアカウントの PATH に存在する必要があります。インクリメンタルバックアップする必要があるデータベースサーバーの **WAL 圧縮**設定を有効にしないでください。
- **Copy WALs while recovery** フィールドを使用して、RESTORE 操作を呼び出したときにアーカイブされた WAL ファイルを収集する方法を指定します。データベースサーバーアーカイブリカバリーの前に、アーカイブされた WAL ファイルを BART バックアップカタログから`<restore_path>/archived_wals` ディレクトリにコピーするには、**Enabled** に設定します。データベースサーバーアーカイブリカバリ中に BART バックアップカタログからアーカイブされた WAL ファイルを直接取得するには、**Disabled** に設定します。
- **Thread count** フィールドを使用して、ブロックをコピーするスレッドの数を指定します。pg_basebackup ユーティリティを使用してバックアップを取る場合は、**Thread count** を **1** に設定する必要があります。
- **Batch size** フィールドを使用して、変更したブロックのコピーに使用されるメモリのブロック数を指定します。このオプションは、インクリメンタルバックアップにのみ適用されます。
- **Scan interval** フィールドを使用して、新しい WAL ファイルをスキャンする前に WAL スキャナーが待機する秒数を指定します。
- **MBM scan timeout** フィールドを使用して、タイムアウトになるまでに MBM ファイルを待機する秒数を指定します。このオプションは、インクリメンタルバックアップにのみ適用されます。

サーバーのプロパティをビューするには、PEM クライアントツリーでサーバー名前を右クリックし、コンテキストメニューから **Properties** を選択します。サーバーのプロパティを変更するには、[Properties] ダ

イアログボックスを開く前にサーバーから切断します。

4.2.5 接続とバインディングの検証

登録されると、新しいサーバーは PEM ブラウザツリーに追加され、グローバル概要ダッシュボードに表示されます。

新しくバインドされたサーバーに最初に接続すると、グローバル概要ダッシュボードには、サーバーリストに「不明」のステータスで新しいサーバーが表示される場合があります。サーバーを認識する前に、バインドされたエージェントは多数のプローブを実行してサーバーを調べる必要があります。ネットワークの可用性によっては、これらのプローブが完了するまでに数分かかる場合があります。

数分で、Global Overview ダッシュボードの棒グラフに、エージェントが正常に接続されたことが示されます。新しいサーバーは **Postgres Server Status** リストに含まれます。

5 分後、Global Overview ダッシュボードはまだ新しいサーバーをリストしていない場合は、モニタリングエージェントのログファイルでエラーを確認します。ツリーでエージェントの名前を右クリックし、コンテキストメニューから **Dashboards > Probe Log Analysis** を選択します。

4.3 Registering a PEM agent

PEM エージェントを使用する前に、PEM サーバーに登録する必要があります。**PEM サーバーパッケージによってインストールされた PEM エージェントは、サーバーの構成中に自動的に登録されます。**他のすべてのエージェントの場合、これらの手順に従う必要があります。

注釈: PEM エージェントをアップグレードした後、再起動する必要があります。再度登録する必要はありません。

4.3.1 PEM エージェントを登録する方法

Linux および Windows ホストでは、PEM エージェントパッケージには、pemworker と呼ばれるコマンドラインユーティリティが含まれています。これを使用して、*pemworker* ユーティリティを使用した *PEM* エージェントの登録 を含む管理タスクを実行できます。

Windows では、PEM エージェントグラフィカルインストーラーを使用して、インストール時にエージェントを登録できます。この便利なオプションは、pemworker ユーティリティが提供するすべての可能性をサポートしているわけではありません。インストーラーにエージェントを登録したくない場合は、**Register now** チェックボックスをオフにします。詳細については、を参照してください。

4.3.2 pemworker ユーティリティを使用した PEM エージェントの登録

pemworker ユーティリティは、PEM エージェントとともに自動的にインストールされます。これは、Linux では/usr/edb/pem/agent/bin ディレクトリ、WindowsC:\Program Files\edb\pem\agent-x64\bin にあります。エージェントを登録するには、PEM サーバーのパスワードを設定し、例に示すようにユーティリティを呼び出し、必要に応じてテーブルから関連オプションを追加します。各オプションの後には、対応する値を続けます。

Linux

```
export PEM_SERVER_PASSWORD=edb
#  Running as root

pemworker --register-agent
```

Windows

```
set PEM_SERVER_PASSWORD=edb
#  Running as admin

./pemworker.exe REGISTER
```

Option	Description
--pem-server	PEM バックエンドデータベースサーバーの IP アドレス。このパラメーターは必須です。
--pem-port	PEM バックエンドデータベースサーバーのポート。デフォルト値は`5432`です。
--pem-user	PEM バックエンドデータベースサーバーのスーパーユーザー権限を持つデータベースユーザーの名前。このパラメーターは必須です。
--pem-agent-user	PEM サーバーのバックエンドデータベースサーバーに接続するエージェントユーザー。
--cert-path	証明書が作成されるディレクトリへの完全なパス。パスを提供しない場合、証明書は Linux では`~/.pem`、 Windows`%APPDATA%/pem`で作成されます。
--config-dir	構成ファイルのディレクトリパス。デフォルトは`<pemworker path>/../etc`です。
--display-name	PEM ブラウザツリーに表示されるエージェントのわかりやすい名前。 PEM 9.6 以降では、デフォルトはホストの完全修飾ドメイン名 FQDN であり、このオプションが設定されていない場合はホスト名にフォールバックします。PEM 9.6 より前のリリースの場合、デフォルトはホスト名です。

次のページに続く

表 4 – 前のページからの続き

Option	Description
--force-registration	`force_registration`句を含めて、指定された引数でエージェントを登録します。この句は、既存のエージェント構成をオーバーライドする場合に役立ちます。デフォルト値は`Yes`です。
--group	エージェントが表示されるグループの名前。
--team	監視対象データベースサーバーへのアクセスを持つ PEM バックエンドデータベースサーバー上のデータベースロールの名前。
--owner	エージェントを所有する PEM バックエンドデータベースサーバー上のデータベースユーザーの名前。
--allow_server_restart	`allow_server_restart`パラメーターを有効にして、PEM が監視対象サーバーを再起動できるようにします。デフォルト値は`True`です。
--allow-batch-probes	`allow-batch-probes`パラメーターを有効にして、PEM がこのエージェントでバッチプローブを実行できるようにします。デフォルト値は`False`です。
--batch-script-user	バッチ/シェルスクリプトの実行に使用するオペレーティングシステムユーザー。デフォルト値は none です。このパラメーターを空白のままにするか、指定されたユーザーが存在しない場合、スクリプトは実行されません。
--enable-heartbeat-connection	`enable-heartbeat-connection`パラメーターを有効にして、PEM エージェントとサーバーの間に専用のハートビート接続を作成し、アクティブステータスを更新します。デフォルト値は`False`です。
--enable-smtp	`enable-smtp parameter`を有効にして、PEM エージェントが PEM サーバーに代わってメールを送信できるようにします。デフォルト値は`False`です。
--enable-snmp	`enable-snmp parameter`を有効にして、PEM エージェントが PEM サーバーに代わって SNMP トラップを送信できるようにします。デフォルト値は`False`です。
-o	構成ファイルオプションをオーバーライドするかどうかを指定します。

pemworker ユーティリティの認証

PEM データベースに変更を加える前に、接続は PEM データベースサーバーで認証されます。pemworker ユーティリティを呼び出すときに、PEM サーバーの管理ユーザーロール postgres に関連付けられたパスワードを提供する必要があります。管理パスワードは 3 つの方法で指定できます。

- `PEM_SERVER_PASSWORD` 環境変数を設定します。
- `PGPASSWORD` キーワードを使用して、コマンドラインにパスワードを入力します。
- `.pgpass` ファイルにエントリーを作成します。

パスワードを入力しない場合、パスワード認証エラーが発生します。認証が成功すると、その他の不足している必須情報の入力を求められます。登録が完了すると、サーバーはエージェントが正常に登録されたことを確認します。

4.3.3 PEM エージェントの登録解除

pemworker ユーティリティを使用して、PEM エージェントの登録を解除できます。エージェントの登録を解除するには、以下の例に示すように pemworker ユーティリティを呼び出します。

Linux

```
#  Running as root

pemworker --unregister-agent
```

Windows

```
./pemworker.exe UNREGISTER-AGENT
```

pemworker ユーティリティを呼び出すときに、コマンド文字列にコマンドラインオプションを追加します。各オプションの後には、対応する値を続けます。

Option	Description
--pem-user <username>	PEM バックエンドデータベースサーバーのデータベースユーザー pem_admin ロールのメンバーの名前を指定します。このパラメーターは必須です。
--config-dir	構成ファイルのディレクトリパスを指定します。デフォルトは `"<pemworker path>/../etc"` です。

4.3.4 高度な使用法

以下は、PEM エージェント登録の詳細なオプションの一部です。

エージェント ID の設定

登録された各 PEM エージェントには、一意のエージェント ID が必要です。デフォルト構成のオーバーライド-例 に示すように、 `-o` オプションを使用して値が提供されない限り、値 `max(id)+1` は各エージェント ID に割り当てられます。

デフォルト構成のオーバーライド-例

この例は、デフォルト構成をオーバーライドして PEM エージェントを登録する方法を示しています。

コマンドラインを使用して PEM エージェントを登録します。 -o オプションを使用して、agent_id 値 8 を割り当てます。

```
#  Running as root

/usr/edb/pem/agent/bin/pemworker --register-agent \
- -pem-server pemserver \
- -pem-user postgres \
- -pem-port 5432 \
- -display-name agent8 \
- o agent_id=8
__OUTPUT__
Postgres Enterprise Manager Agent registered successfully!
```

8 の agent_id が利用可能なため、PEM エージェントは正常に登録されました。指定された ID が既存のエージェントによって既に使用されている場合、エラーがスローされます。

コマンドラインを使用して PEM エージェントを登録します。既存の SSL 証明書とキーファイルを割り当てて、特定のエージェント ID に対して新しい証明書が生成されないようにします。SSL 証明書とキーファイルは、データベースユーザー agent<ID> に対して有効である必要があります。<ID> は、コマンドラインを使用して指定されたものと同じである必要があります。-o オプションを使用します。

```
#  Running as root

#  List the location of valid SSL certificates and key files.

ls -l /root/.pem/agent5.*
- rw------- 1 root root 2192 Nov  7 11:27 /root/.pem/agent5.crt
- rw------- 1 root root 3244 Nov  7 11:27 /root/.pem/agent5.key

#  Register the PEM agent using command line. Assign the

#  SSL certificates and key files using the -o option.

/usr/edb/pem/agent/bin/pemworker --register-agent \
- -pem-server pemserver \
- -pem-user postgres \
- -pem-port 5432 \
- -config-dir /tmp/pem-config \
- -display-name agent5 \
- o agent_id=5 \
- o agent_ssl_crt=/root/.pem/agent5.crt \
- o agent_ssl_key=/root/.pem/agent5.key
```

(次のページに続く)

(前のページからの続き)

```
__OUTPUT__
Postgres Enterprise Manager Agent registered successfully!
```

``` 有効な SSL 証明書とキーファイルは、適切な権限で指定された場所で利用できるため、PEM エージェントは正常に登録されます。証明書またはキーファイルが有効でない場合、または適切な権限がない場合、エラーがスローされます。

### 非 root ユーザーアカウントを使用して Linux で PEM エージェントを登録する

非 root ユーザーアカウントを使用して PEM エージェントを登録するには、最初に root ユーザーとして PEM エージェントをインストールする必要があります。インストール後、edb などの非 root ユーザーの ID を引き継ぎます。次に

1. edb としてログインします。 `pem` および `logs` ディレクトリを作成し、読み取り、書き込み、および実行権限を割り当てます。

```shell
    # Running as nonroot user edb
    mkdir /home/edb/pem
    mkdir /home/edb/pem/logs
    chmod 700 /home/edb/pem
    chmod 700 /home/edb/pem/logs
```

2. エージェントを PEM サーバーに登録します。

```
export PEM_SERVER_PASSWORD=edb

# Use the following command to create agent certificates and an agent
# configuration file (`agent.cfg`) in the `/home/edb/pem` directory.
/usr/edb/pem/agent/bin/pemworker --register-agent --pem-server <172.19.11.230> --pem-
↪user postgres --pem-port 5432 --display-name non_root_pem_agent --cert-path /home/
↪edb/pem --config-dir /home/edb/pem

# Use the following command to assign read and write permissions to
# these files:
chmod -R 600 /home/edb/pem/agent*
```

3. agent.cfg ファイルのパラメーターを変更します。

```
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
```

<id> は、割り当てられた PEM エージェント ID です。
```

4. tmp ディレクトリを作成し、環境変数を設定して、エージェントを起動します。

```
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
```

これで、PEM エージェントが登録され、edb ユーザーで起動されました。マシンが再起動しても、このエージェントは自動的に再起動しません。前述のコマンドを使用して手動で起動する必要があります。

5. オプションで、ルートユーザーとしてこの PEM エージェントのサービスを作成して、次のようにマシンの再起動時にこのエージェントを自動的に起動できます。

a. 構成ファイルパスの値と、pemagent サービスファイルのユーザーをスーパーユーザーとして更新します。

```
# Running as superuser
sudo vi  /usr/lib/systemd/system/pemagent.service
[Service]
Type=forking
WorkingDirectory=/home/edb/pem
Environment=LD_LIBRARY_PATH=/usr/edb/pem/agent/lib:/usr/libexec/edb-snmp++/lib
Environment=TEMP=/home/edb/pem/tmp
ExecStart=/usr/edb/pem/agent/bin/pemagent -c /home/edb/pem/agent.cfg
```

b. 実行中のエージェントプロセスを停止し、エージェントサービスを再起動します。

```
# 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. PEM ダッシュボードでエージェントのステータスを確認します。

注釈:

- root 権限または別のユーザーたとえば、enterprisedb が所有するファイルへのアクセスを必要とするプローブとジョブは失敗します。 - *agent.cfg* ファイルをデフォルトの場所から別の場所に移動すると、PEM ダッシュボードにエージェントステータスが `unknown` として表示される場合があります。詳細については、 *Troubleshooting agent issues* を参照してください。

4.4 Managing a PEM server

PEM サーバーの管理に関連するタスクの一部は次のとおりです。

- PEM サーバーとエージェントの再起動

・ PEM サーバーまたは PEM エージェントの制御

- Linux および Windows での HTTPD サービスの制御
- HTTPD サーバーの制御
- PEM 認証とセキュリティの管理
- `pg_hba.conf` ファイルの変更
- プロキシサーバーを使用するように PEM を変更する

4.4.1 PEM サーバーとエージェントの起動と停止

PEM サーバーは、それが常駐する Postgres サーバインスタンスが起動、停止、または再起動したときに起動、停止、および再起動します。Postgres サーバーの制御に使用するのと同じコマンドを使用して PEM サーバーを制御します。Linux プラットフォームでは、サービススクリプトを停止および開始するコマンドは、プラットフォームおよび OS バージョンによって異なります。

PEM エージェントは、pemagent という名前のサービスによって制御されます。

Windows オペレーティングシステムには、サーバーの状態を表示し、対話型のサーバー制御を提供するグラフィカルサービスコントローラが含まれています。Windows Control Panel から Services ユーティリティにアクセスできます。ユーティリティが開いたら、リストされたサービスをナビゲートし、サービス名を選択します。

サービスの状態を制御するには、 **Stop** 、 **Pause** 、 **Start** 、または **Restart** を選択します。

Postgres サーバーに接続しているユーザまたはクライアントアプリケーションは、サービスを停止すると突然切断されます。サービスの制御の詳細については、を参照してください。

4.4.2 Linux での PEM サーバーまたは PEM エージェントの制御

Linux プラットフォームで、以下を制御するサービススクリプトの名前。

- EDB Postgres Advanced Server の PEM サーバーは edb-as-<x> または ppas-<x> です。
- PostgreSQL の PEM サーバーは postgresql-<x> です。
- PEM エージェントは pemagent です。

x は、サーバーバージョン番号を示します。

サービススクリプトを使用して、サービスを制御できます。

RHEL 8.x でサービスを制御するには、コマンドプロンプトで、スーパーユーザー権限を引き受けて、コマンドを発行します。

```
systemctl <service_name> <action>
```

そこで

service_name はサービスの名前です。

action は、サービスが実行するアクションを指定します。以下を指定します

- start サービスを開始します。
- stop サービスを停止します。
- restart サービスを停止してから開始します。
- status サービスのステータスを確認します。

4.4.3 Windows での PEM サーバーまたは PEM エージェントの制御

Windows オペレーティングシステムには、サーバーの状態を表示し、対話型のサーバー制御を提供するグラフィカルサービスコントローラが含まれています。以下を制御するサービスの登録名前

- PostgreSQL の PEM サーバーホストは postgresql-<x> です。
- Advanced Server 上の PEM サーバーホストは edb-as-<x> または ppas-<x> です。
- PEM エージェントは Postgres Enterprise Manager - pemAgent です。

x は、サーバーバージョン番号を示します。

Windows Control Panel で、 Services ユーティリティを開き、Services のリストを表示します。

サービスを停止するには、 **Stop the service** を選択します。サーバーに接続しているユーザまたはクライアントアプリケーションは、サービスを停止すると突然切断されます。

Pause the service を選択して、サービスの構成パラメーターをリロードします。**サービスの一時停止**は、多くの構成パラメーターのユーザセッションを中断せずにパラメーターをリセットする効果的な方法です。

サービスを開始するには、 **Start the service** を選択します。

4.4.4 HTTPD サーバーの制御

Linux では、コマンドプロンプトで次のコマンドを入力して、PEM-HTTPD サービスのステータスを確認できます。

```
ps -ef | grep httpd
```

Linux が次のような回答で応答した場合、httpd は実行されていません。

```
user 13321 13267 0 07:37 pts/1 00:00:00 grep httpd
```

RHEL 8.x システムでサービスをスタートするには、次のコマンドを使用します。

```
systemctl start httpd
```

Windows、サービスアプレットを使用して、PEM HTTPD サービスのステータスを確認できます。サービスアプレットを開いた後、リストから PEM HTTPD サービスを選択します。

Status 列には、サーバーの現在の状態が表示されます。サービスが実行されていない場合は、**Start** を選択して PEM HTTPD を起動します。

4.4.5 PEM 認証の管理

Postgres は多くの認証方法をサポートしています

- 安全なパスワード MD5
- GSSAPI
- SSPI
- ケルベロス
- アイデント
- LDAP
- RADIUS
- 証明書 SSL
- PAM

Postgres および PEM 認証は、`pg_hba.conf` 構成ファイルによって制御されます。構成ファイルのエントリでは、次のことを指定します。

- 特定のデータベースに接続できる人
- そのユーザーが接続を許可される前に必要な認証の種類

postgres という名前のユーザーが認証接続がフォームを取る安全なパスワード MD5 を使用して、ローカルホスト 127.0.0.1/32 からすべてのデータベースに接続できます `pg_hba.conf` ファイル内の一般的なエントリー

```
host all postgres 127.0.0.1/32 md5
```

システム構成によっては、PEM エージェントがサーバーへの接続に使用するユーザーアカウントのパスワードファイルを作成する必要がある場合があります。このファイルにより、エージェントはサーバーの認証要求に適切に応答できます。パスワードが `1safepwd` である postgres 名前付けのユーザーのパスワードファイルのエントリーは、次のフォームます。

```
localhost:5432:*:postgres:1safepwd
```

パスワードファイルの名前は、通常、Linux システムでは`~root/.pgpass` 、または Windows`%APPDATA%\postgresql\pgpass.conf` です。パスワードファイルの構成の詳細については、 PostgreSQL website を参照してください。

Postgres でサポートされている認証方法の詳細については、
PostgreSQL core documentation を参照してください。

4.4.6 PEM サーバー構成の編集

PEM クライアントを使用して PEM サーバーの構成パラメーターを管理し、機能を有効にしたり、デフォルト設定を変更したりできます。[Server Configuration] ダイアログボックスを開くには、[**Management > Server Configuration**] を選択します。次に、必要に応じてパラメーター値を編集します。

4.4.7 セキュリティの管理

PEM は、Postgres のロールとサーバーを管理するためのインターフェイスを提供します。

ログインロール

PEM サーバーに接続するときは、PEM サーバーがデータを保存するデータベースへのアクセスを許可するロール資格情報を提供する必要があります。デフォルトでは、postgres スーパーユーザーアカウントはサーバーへの最初の接続に使用されますが、セキュリティと監査の両方の目的で、接続するユーザーごとに個別のロールを作成することを強くお勧めします。PEM クエリツール、PEM Web インターフェイスの Create – Login/Group Role ダイアログボックス、またはコマンドラインクライアント psql などを使用して、ロールを作成できます。

[Create – Login/Group Role] ダイアログボックスを使用してロールを作成するには

1. PEM ツリーで、ロールを作成するサーバーのノードを展開します。
2. **Login/Group Roles** ノードを右クリックします。
3. コンテキストメニューから、 **Create > Login/Group Role** を選択します。
4. [Create – Login/Group Role] ダイアログボックスのタブを使用して、ロールを定義します。
5. 新しいロールの定義が終了したら、[**Save**] を選択してロールを作成します。

既存のログインロールのプロパティを変更するには、ツリーでログインロールの名前を右クリックし、コンテキストメニューから**プロパティ**を選択します。ログインロールを削除するには、ロールの名前を右クリックし、コンテキストメニューから **Delete/Drop** を選択します。

ロールの作成と管理の詳細については、 PostgreSQL online documentation を参照してください。

グループロール

グループロールはコンテナとして機能し、データベースの作成などのシステム特権やテーブルへのデータの挿入などのオブジェクト特権を分配するために使用されます。グループロールの主な目的は、DBA によるシステムおよびオブジェクト権限の大量管理を簡単にすることです。多くの異なるログインアカウントに個別に特権を割り当てまたは変更するのではなく、単一のロールの特権を割り当てまたは変更し、そのロールを一度に多くのログインロールに付与できます。

Group Roles ノード PEM ツリーの各登録サーバーの名前の下にありますを使用して、グループロールを作成および管理します。コンテキストメニューのオプションは、新しいロールを作成したり、既存のロールのプロパティを変更したりできるダイアログボックスへのアクセスを提供します。ロールの作成の詳細については、 PostgreSQL documentation を参照してください。

PEM の事前定義されたロールを使用して PEM 機能へのアクセスを管理する

Login/Group Role ダイアログボックスを使用して、制限された特権を持つロールに Audit Manager、Capacity Manager、SQL プロファイラーなどの PEM 機能にアクセスを許可できます。PEM の事前定義されたロールにより、PEM 機能へのアクセスが許可されます。これらのロールのメンバーシップが割り当てられたロールは、関連付けられた機能にアクセスできます。

ユーザーを定義するときは、 **Membership** タブを使用して、新しいユーザーがメンバーであるロールを指定します。新しいユーザーは、メンバーである各ロールに関連付けられた特権を共有します。ユーザーが PEM 拡張機能にアクセスするには、ロールが pem_user ロールと、機能へのアクセスを許可する事前定義されたロールのメンバーである必要があります。**役割**フィールドを使用して、リストから事前定義されたロール名を選択します。

[**SQL**] タブには、 **Save** を選択したときにサーバーが実行 SQL コマンドが表示されます。

この例では、監査マネージャーにアクセスする acctg_clerk という名前のログインロールを作成します。ロールは、任意の時点で、サーバーへの無制限の接続を make ことができます。

PEM の事前定義されたロールを使用して、表にリストされている機能へのアクセスを許可できます。

Value	Parent role	Description
pem_super_admin		PEM 上のすべてを管理/構成するロール
pem_admin	pem_super_admin	すべての可視エージェント/サーバーおよび監視対象オブジェクトの管理/管理/構成のためのロール

次のページに続く

表 6 – 前のページからの続き

Value	Parent role	Description
pem_user		pem_user ロールを持つユーザに表示されるすべてのエージェント、サーバー、または監視対象オブジェクトへの読み取り専用アクセスを持つロール。 pem_user ロールを持つユーザーは、サーバーのプロパティの下の Team フィールドでこのロールが記載されているオブジェクトのみを表示できます。
pem_config	pem_admin	PEM の構成管理のためのロール
pem_component	pem_admin	すべてのウィザード/ダイアログボックスベースのコンポーネントを実行/実行するロール
pem_rest_api	pem_admin	REST API にアクセスするロール
pem_server_service_ma	pem_admin	監視対象データベースサーバーを再起動/リロードできるロールサーバー ID が提供されている場合
pem_manage_schedule_	pem_admin	スケジュールタスクを構成するロール
pem_manage_chart	pem_admin	カスタムチャートを管理/構成するためのロール。
pem_manage_alert	pem_admin	アラートとそのテンプレートを管理/構成するためのロール
pem_config_alert	pem_config, pem_manage_alert	監視対象オブジェクトのアラートを構成するロール
pem_manage_probe	pem_admin	カスタムプローブを作成、更新、および削除し、カスタムプローブ構成を変更するロール
pem_config_probe	pem_config, pem_manage_probe	すべての目に見える監視対象オブジェクトのプローブ構成のロール履歴保持、実行頻度、プローブの有効化/無効化
pem_database_server_re	pem_admin	データベースサーバーを登録するロール
pem_comp_postgres_ex	pem_component	Postgres Expert を実行するロール
pem_comp_auto_discov	pem_component	データベースサーバーの自動検出ダイアログボックスを実行するロール
pem_comp_log_analysis	pem_component	Log Analysis Expert を実行するロール
pem_comp_sqlprofiler	pem_component	SQL プロファイラーを実行するロール
pem_manage_efm	pem_admin	フェールオーバーマネージャーを管理するロール
pem_comp_capacity_ma	pem_component	Capacity Manager を実行するロール
pem_comp_log_manage	pem_component	Log Manager を実行するロール
pem_comp_audit_manag	pem_component	Audit Manager を実行するロール
pem_comp_tuning_wiza	pem_component	チューニングウィザードを実行するロール
pem_comp_bart	pem_component	BART サーバーを構成および管理するロール。
pem_comp_performance	pem_component	パフォーマンス診断を実行するロール。

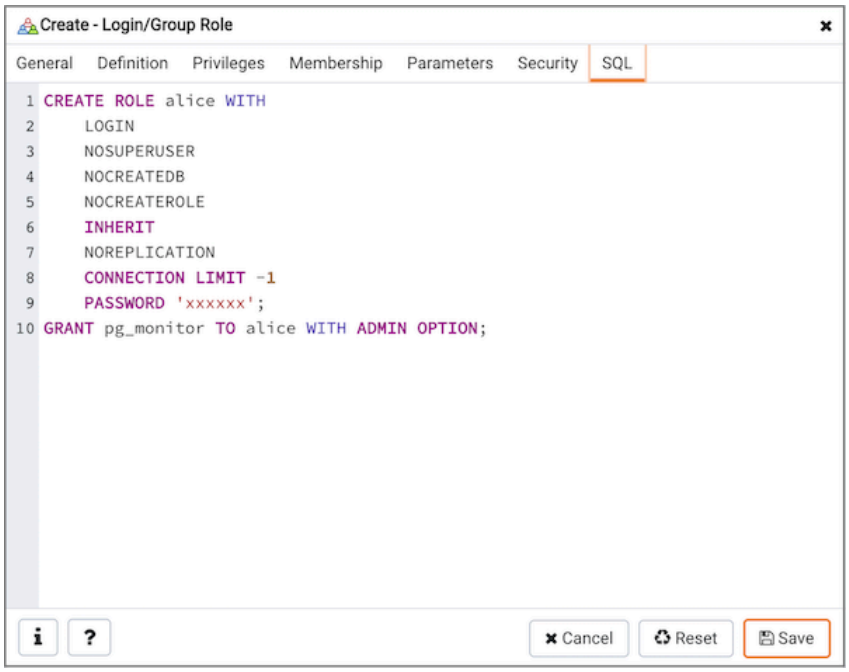


図 1: Create Login Role - SQL tab

チームロールを使用する

PEM による監視のためのサーバーを登録するときに、サーバーに関連付けるチームを指定できます。チームは、1 つ以上の監視対象サーバーへのアクセスを限定されたロールメンバーのグループに許可または制限するために使用できるグループロールです。PEM クライアントは、指定されたチームのサーバーを次のユーザーにのみ表示します。

- チームロールのメンバー
- サーバーを作成したロール
- PEM サーバーでスーパーユーザー権限を持つロール

チームロールを作成するには

1. PEM ツリーで、ロールを作成するサーバーのノードを展開します。
2. **Login/Group Roles** ノードを右クリックします。
3. コンテキストメニューから、 **Create > Login/Group Role** を選択します。
4. [Create - Login/Group Role] ダイアログボックスで、フィールドを使用してチームロールのプロパティを指定します。

オブジェクト権限

データベースオブジェクトにアクセス、実行、または作成する前に、ロールに十分な特権を付与する必要があります。PEM では、PEM クライアントインターフェイスを使用して、グループロールまたはログインアカウントにオブジェクト権限を割り当て GRANT および削除 REVOKE できます。

オブジェクトの権限は、特定のオブジェクトごとにグラフィカルオブジェクトエディタで管理されます。たとえば、データベーステーブルにアクセスする特権を割り当てるには、ツリーでテーブル名前を右クリックし、コンテキストメニューから**プロパティ** を選択します。 **Privileges** タブを使用して、テーブルの特権を割り当てます。

また、PEM クライアントには、Grant ウィザード **Tools** メニューからアクセスが含まれており、一度に多くのオブジェクト権限を管理できます。

4.4.8 サーバー構成

[Server Configuration] ダイアログボックスを使用して、PEM 動作を制御するユーザー設定可能なパラメーターの値を変更できます。[Server Configuration] ダイアログボックスにアクセスするには、PEM サーバーに接続し、 **File > Server Configuration** を選択します。

ダイアログの右上隅の検索ボックスにパラメーター名を入力して、リスト内の特定のパラメーターを見つけます。

パラメーター値を変更するには、パラメーター名の右側にある **Value** フィールドに表示される内容を編集します。変更を保存するには、ダイアログボックスの右上隅にある **Save** アイコンを選択します。

4.4.9 サーバー構成パラメーター - リファレンス

グローバル構成オプションを使用して、PEM サーバーの動作の側面を変更できます。構成パラメーターのリストは変更される場合があります。

Parameter name	Value and unit	Description
audit_log_retention_time	30 days	監査ログが PEM サーバーに保持される日数を指定します。
auto_create_agent_alerts	true	エージェントの登録時に、デフォルトのエージェントレベルのアラートを自動的に作成するかどうかを指定します。
auto_create_server_alert	true	サーバーがエージェントにバインドされたときに、デフォルトのサーバーレベルのアラートを自動的に作成するかどうかを指定します。
chart_disable_bullets	false	ダッシュボードの折れ線グラフと Capacity Manager レポートの箇条書きを有効/無効にします。
cm_data_points_per_rep	50	Capacity Manager レポートのチャートにプロットするデータポイントの数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
cm_max_end_date_in_y	5 years	Capacity Manager がデータを推定する最大時間を指定します。レポートのしきい値ベースの終了日が無期限に推定されないようにします。
dash_alerts_timeout	60 seconds	アラートダッシュボードのコンポーネントが更新される秒数を指定します。
dash_db_comrol_span	7 days	データベース分析およびサーバー分析ダッシュボードのコミット/ロールバック分析チャートにプロットするデータの日数を指定します。
dash_db_comrol_timeou	1800 seconds	データベース分析およびサーバー分析ダッシュボードでコミット/ロールバック折れ線グラフが更新されるまでの秒数を指定します。
dash_db_connovervw_ti	300 seconds	データベース分析ダッシュボードで接続の概要円グラフが更新されるまでの秒数を指定します。
dash_db_eventlag_span	7 days	データベース分析ダッシュボードのスロニーレプリケーションのイベント数ラグチャートにプロットするデータの日数を指定します。
dash_db_eventlag_timec	1800 seconds	slony レプリケーションの Number of Events Lag 折れ線グラフがデータベース分析ダッシュボードで更新されるまでの秒数を指定します。
dash_db_hottable_rows	25 rows	データベース分析ダッシュボードの HOT Table Analysis テーブルに表示する行数を指定します。
dash_db_hottable_timeo	300 seconds	データベース分析ダッシュボードでホットテーブルテーブルが更新されるまでの秒数を指定します。
dash_db_io_span	7 days	データベース分析および I/O 分析ダッシュボードのデータベース I/O 分析チャートにプロットするデータの日数を指定します。
dash_db_io_timeout	1800 seconds	データベース分析および I/O 分析ダッシュボードでデータベース I/O 折れ線グラフが更新されるまでの秒数を指定します。
dash_db_rowact_span	7 days	データベース分析、I/O 分析、およびサーバー分析ダッシュボードの行アクティビティ分析チャートにプロットするデータの日数を指定します。
dash_db_rowact_timeou	1800 seconds	データベース分析、I/O 分析、およびサーバー分析のダッシュボードで行アクティビティ折れ線グラフが更新されるまでの秒数を指定します。
dash_db_storage_timeou	300 seconds	データベース分析ダッシュボードでストレージ棒グラフが更新されるまでの秒数を指定します。
dash_db_timelag_span	7 days	データベース分析ダッシュボードの Slony レプリケーションのタイムラグチャートにプロットするデータの日数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
dash_db_timelag_timeou	1800 seconds	データベース分析ダッシュボードで Slony レプリケーションのタイムラグ折れ線グラフが更新されるまでの秒数を指定します。
dash_db_useract_span	7 days	データベース分析ダッシュボードのユーザーアクティビティ分析チャートにプロットするデータの日数を指定します。
dash_db_useract_timeou	1800 seconds	データベース分析ダッシュボードでユーザーアクティビティ折れ線グラフが更新されるまでの秒数を指定します。
dash_efm_timeout	300 seconds	ストリーミングレプリケーションダッシュボードでフェールオーバーマネージャーノードのステータスとフェールオーバーマネージャークラスター情報の折れ線グラフが更新されるまでの秒数を指定します。
dash_global_overview_t	30 seconds	グローバル概要ダッシュボードのコンポーネントが更新されるまでの秒数を指定します。
dash_header_timeout	60 seconds	すべてのダッシュボードのヘッダーの情報が更新されるまでの秒数を指定します。
dash_io_chkpt_span	7 days	I/O Analysis ダッシュボードのチェックポイントチャートにプロットするデータの日数を指定します。
dash_io_chkpt_timeout	1800 seconds	I/O 分析ダッシュボードでチェックポイント折れ線グラフが更新されるまでの秒数を指定します。
dash_io_hotindx_timeou	300 seconds	I/O 分析ダッシュボードでホットインデックス棒グラフが更新されるまでの秒数を指定します。
dash_io_hottbl_timeout	300 seconds	I/O 分析ダッシュボードでホットテーブル棒グラフが更新されるまでの秒数を指定します。
dash_io_index_objectio_	25 rows	I/O 分析およびオブジェクトアクティビティ分析ダッシュボードのインデックスアクティビティテーブルに表示される行数を指定します。
dash_io_index_objectio_	60 seconds	I/O 分析およびオブジェクトアクティビティ分析ダッシュボードでインデックスアクティビティテーブルが更新されるまでの秒数を指定します。
dash_io_objectio_rows	25 rows	I/O 分析およびオブジェクトアクティビティ分析ダッシュボードの Object I/O Details テーブルに表示される行数を指定します。
dash_io_objectio_timeou	300 seconds	I/O 分析およびオブジェクトアクティビティ分析ダッシュボードで Object I/O Details テーブルが更新されるまでの秒数を指定します。
dash_memory_hostmem	7 days	メモリ分析ダッシュボードのホストメモリアクティビティ分析チャートにプロットするデータの日数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
dash_memory_hostmem	1800 seconds	メモリ分析ダッシュボードでホストメモリアクティビティ折れ線グラフが更新されるまでの秒数を指定します。
dash_memory_hostmem	300 seconds	メモリ分析およびサーバー分析ダッシュボードでホストメモリ構成円グラフが更新されるまでの秒数を指定します。
dash_memory_servmem	7 days	サーバーのメモリ分析ダッシュボードのメモリアクティビティ分析チャートにプロットするデータの日数を指定します。
dash_memory_servmem	1800 seconds	メモリ分析ダッシュボードでサーバーメモリアクティビティ折れ線グラフが更新されるまでの秒数を指定します。
dash_memory_servmem	300 seconds	メモリ分析ダッシュボードでサーバーメモリ構成円グラフが更新されるまでの秒数を指定します。
dash_objectact_objstora	15 rows	オブジェクトアクティビティ分析ダッシュボードのオブジェクトストレージテーブルに表示する行数を指定します。
dash_objectact_objstora	300 seconds	オブジェクトアクティビティ分析ダッシュボードでオブジェクトストレージテーブルが更新されるまでの秒数を指定します。
dash_objectact_objtopin	300 seconds	オブジェクトアクティビティ分析ダッシュボードでトップ 5 最大インデックス棒グラフが更新されるまでの秒数を指定します。
dash_objectact_objtoptal	300 seconds	オブジェクトアクティビティ分析ダッシュボードで上位 5 最大テーブル棒グラフが更新されるまでの秒数を指定します。
dash_os_cpu_span	7 days	オペレーティングシステム分析ダッシュボードの CPU チャートにプロットするデータの日数を指定します。
dash_os_cpu_timeout	1800 seconds	オペレーティングシステム分析ダッシュボードで CPU 折れ線グラフが更新されるまでの秒数を指定します。
dash_os_data_span	7 days	オペレーティングシステム分析ダッシュボードの I/O 折れ線グラフにプロットするデータの日数を指定します。
dash_os_disk_span	7 days	オペレーティングシステム分析ダッシュボードの使用率グラフにプロットするデータの日数を指定します。
dash_os_hostfs_timeout	1800 seconds	オペレーティングシステム分析ダッシュボードでホストファイルシステムの詳細テーブルが更新されるまでの秒数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
dash_os_io_timeout	1800 seconds	オペレーティングシステム分析ダッシュボードで I/O 折れ線グラフが更新されるまでの秒数を指定します。
dash_os_memory_span	7 days	オペレーティングシステム分析ダッシュボードのメモリチャートにプロットするデータの日数を指定します。
dash_os_memory_timeo	1800 seconds	オペレーティングシステム分析ダッシュボードでメモリ折れ線グラフが更新されるまでの秒数を指定します。
dash_os_packet_span	7 days	オペレーティングシステム分析ダッシュボードのパケットチャートにプロットするデータの日数を指定します。
dash_os_packet_timeout	1800 seconds	オペレーティングシステム分析ダッシュボードでネットワークパケット折れ線グラフが更新されるまでの秒数を指定します。
dash_os_process_span	7 days	オペレーティングシステム分析ダッシュボードのプロセスチャートにプロットするデータの日数を指定します。
dash_os_process_timeou	1800 seconds	オペレーティングシステム分析ダッシュボードでプロセス折れ線グラフが更新されるまでの秒数を指定します。
dash_os_storage_timeou	1800 seconds	オペレーティングシステム分析ダッシュボードでストレージ円グラフが更新されるまでの秒数を指定します。
dash_os_traffic_span	7 days	オペレーティングシステム分析ダッシュボードのトラフィックチャートにプロットするデータの日数を指定します。
dash_os_traffic_timeout	1800 seconds	オペレーティングシステム分析ダッシュボードでトラフィック折れ線グラフが更新されるまでの秒数を指定します。
dash_os_util_timeout	1800 seconds	オペレーティングシステム分析ダッシュボードで使用率折れ線グラフが更新されるまでの秒数を指定します。
dash_probe_log_timeout	300 seconds	プローブログテーブルが更新されるまでの秒数を指定します。
dash_replication_archive	7 days	Streaming Replication Analysis ダッシュボードの WAL Archive Status チャートにプロットするデータの日数を指定します。
dash_replication_archive	1800 seconds	ストリーミングレプリケーションダッシュボードで WAL Archive Status 折れ線グラフが更新されるまでの秒数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
dash_replication_pagela	7 days	ストリーミングレプリケーションダッシュボードの WAL Lag Pages チャートにプロットするデータの日数を指定します。
dash_replication_pagela	1800 seconds	ストリーミングレプリケーションダッシュボードで WAL Lag Pages 折れ線グラフが更新されるまでの秒数を指定します。
dash_replication_segmer	7 days	ストリーミングレプリケーションダッシュボードの WAL ラグセグメントチャートにプロットするデータの日数を指定します。
dash_replication_segmer	1800 seconds	ストリーミングレプリケーションダッシュボードで WAL ラグセグメント折れ線グラフが更新されるまでの秒数を指定します。
dash_replication_timelag	7 days	ストリーミングレプリケーションダッシュボードのレプリケーションラグタイムチャートにプロットするデータの日数を指定します。
dash_replication_timelag	1800 seconds	ストリーミングレプリケーションダッシュボードでレプリケーションラグタイム折れ線グラフが更新されるまでの秒数を指定します。
dash_server_buffers_wri	168 hours	Server Analysis ダッシュボードの Background Writer Statistics チャートにプロットするデータの日数を指定します。
dash_server_buffers_wri	300 seconds	Server Analysis ダッシュボードで Background Writer Statistics の折れ線グラフが更新されるまでの秒数を指定します。
dash_server_connovervv	300 seconds	Server Analysis ダッシュボードで Connection Overview 円グラフが更新されるまでの秒数を指定します。
dash_server_database_ti	300 seconds	Server Analysis ダッシュボードで Databases テーブルが更新されるまでの秒数を指定します。
dash_server_dbsize_spar	7 days	Server Analysis ダッシュボードのデータベースサイズ分析にプロットするデータの日数を指定します。
dash_server_dbsize_time	1800 seconds	Server Analysis ダッシュボードでデータベースサイズの折れ線グラフが更新されるまでの秒数を指定します。
dash_server_disk_timeou	1800 seconds	Server Analysis ダッシュボードでディスク折れ線グラフが更新されるまでの秒数を指定します。
dash_server_global_spar	7 days	Server Analysis ダッシュボードのディスク折れ線グラフにプロットするデータの日数を指定します。
dash_server_sharedbuff_	7 days	Server Analysis ダッシュボードの共有バッファチャートにプロットするデータの日数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
dash_server_sharedbuff_	1800 seconds	Server Analysis ダッシュボードで共有バッファーの折れ線グラフが更新されるまでの秒数を指定します。
dash_server_tabspacesiz	7 days	Server Analysis ダッシュボードのテーブルスペースサイズチャートにプロットするデータの日数を指定します。
dash_server_tabspacesiz	1800 seconds	Server Analysis ダッシュボードでテーブルスペースサイズの折れ線グラフが更新されるまでの秒数を指定します。
dash_server_useract_spa	7 days	Server Analysis ダッシュボードのユーザーアクティビティチャートにプロットするデータの日数を指定します。
dash_server_useract_tim	1800 seconds	Server Analysis ダッシュボードでユーザーアクティビティ折れ線グラフが更新されるまでの秒数を指定します。
dash_sess_waits_timewa	300 seconds	セッション待機時間分析ダッシュボードでセッション待機時間ごとの円グラフが更新されるまでの秒数を指定します。
dash_sess_waits_waitdtl	300 seconds	Session Waits Analysis ダッシュボードの Session Waits Details テーブルが更新されるまでの秒数を指定します。
dash_sessact_lockact_tir	300 seconds	セッションアクティビティ分析ダッシュボードのセッションロックアクティビティテーブルが更新されるまでの秒数を指定します。
dash_sessact_workload_	300 seconds	セッションアクティビティ分析ダッシュボードのセッションワークロードテーブルが更新されるまでの秒数を指定します。
dash_storage_dbdtls_tim	300 seconds	ストレージ分析ダッシュボードでデータベースの詳細テーブルが更新されるまでの秒数を指定します。
dash_storage_dbovervw_	300 seconds	ストレージ分析ダッシュボードでデータベースの概要円グラフが更新されるまでの秒数を指定します。
dash_storage_hostdtls_ti	300 seconds	Host Details テーブルが更新されるまでの秒数を指定します。
dash_storage_hostoverv	300 seconds	ストレージ分析ダッシュボードでホスト概要円グラフが更新されるまでの秒数を指定します。
dash_storage_tblspcdtls_	300 seconds	Storage Analysis ダッシュボードで Tablespace Details テーブルが更新されるまでの秒数を指定します。
dash_storage_tblspcover	300 seconds	ストレージ分析ダッシュボードでテーブルスペースの概要円グラフが更新されるまでの秒数を指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
dash_sys_waits_nowaits_	300 seconds	システム待機数分析ダッシュボードで待機数別のシステム待機円グラフが更新されるまでの秒数を指定します。
dash_sys_waits_timewai	300 seconds	システム待機時間分析ダッシュボードでシステム待機時間ごとの円グラフが更新されるまでの秒数を指定します。
dash_sys_waits_waitdtl_	300 seconds	システム待機分析ダッシュボードのシステム待機の詳細テーブルが更新されるまでの秒数を指定します。
deleted_charts_retention	7 days	カスタムグラフユーザー定義のダッシュボードに表示されるを保存する日数を指定します。
deleted_objects_data_ret	3 days	削除されたオブジェクトたとえば、サーバーまたはエージェントまたは Barman サーバーのプローブデータが pemhistory および pemdata スキーマから削除されるまでの日数を指定します。これらの古いオブジェクトのデータを削除したくない場合、ジョブを削除します。
deleted_probes_retentio	7 days	カスタムプローブユーザー定義のダッシュボードに表示されるを保存する日数を指定します。
download_chart_format	jpeg	ダウンロードしたチャートを保存する形式を指定します。 jpeg または png にすることができます。
flapping_detection_state_cha	3	指定された間隔内で検出された状態変更の数を指定して、特定のアラートをフラッピングとして定義します。フラッピングは、 *N* **を超える状態変更が** *[N* + 1 *(min(probe_interval) * 2)] 分間発生し、詳細な状態が None 以外の場合に開始されます。 *N* **のデフォルト値は** *2* **または** *3* **で、** *min(probe_interval)* **は、アラートで使用されるすべてのプローブの最小間隔です。フラッピングは、状態の変更が** *[2 N* * min(probe_interval)] 分間発生したときに終了します。
job_retention_time	30 days	非定期的なスケジュールされたタスクとそれに関連するジョブが保持される日数を指定します。
long_running_transactio	5 minutes	長時間実行されていると考えられるまでに、クエリが実行される分数を指定します。
nagios_cmd_file_name	<file_name>	パッシブサービスチェック結果を送信する nagios コマンドファイルを指定します。
nagios_enabled	t	アラート通知を nagios に送信するかどうかを指定します。
nagios_medium_alert_as_	f	中レベルの PEM アラートが nagios で重要であると考えられるかどうかを指定します。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
na-gios_spool_retention_tin	7 days	nagios メッセージが破棄されるまでにスプールテーブルに保持する日数を指定します。
probe_log_retention_tim	30 days	プローブログレコードが保持される日数を指定します。
re-minder_notification_inte	24 hours	アラートがクリアされなかった場合に、リマインダー電子メールを送信する時間を指定します。
server_log_retention_tin	30 days	サーバーログが PEM サーバーに保持される日数を指定します。
show_data_tab_on_grap	false	`true`の場合、 **Data** タブが各グラフに追加されます。 **Data** タブを選択して、グラフにプロットされたデータを確認します。
smtp_authentication	false	SMTP での認証を有効/無効にするかどうかを指定します。
smtp_enabled	true	メールの送信を有効にするか無効にするかを指定します。
smtp_encryption	false	暗号化された接続を使用して SMTP 電子メールを送信するかどうかを指定します。
smtp_password		SMTP サーバーへの接続に使用するパスワードを指定します。
smtp_port	25	電子メールの送信に使用する SMTP サーバーポートを指定します。
smtp_server	127.0.0.1	電子メールの送信に使用する SMTP サーバーホストアドレスを指定します。
smtp_spool_retention_ti	7 days	送信された電子メールメッセージが破棄されるまでにスプールテーブルに保持する日数を指定します。
smtp_username		SMTP サーバーへの接続に使用するユーザー名を指定します。
snmp_authentication_pa		snmp_security_name で指定されたセキュリティ名に関連付けられた認証パスワードを指定します。 SNMPv3 にのみ使用されます。
snmp_authentication_pr	NONE	SNMP トラップの認証タイプを指定します。可能な値は、NONE、HMACMD5、および HMACSHA です。 SNMPv3 でのみ使用されます。
snmp_community	public	トラップ送信時に使用される SNMP コミュニティを指定します。 SNMPv1 および SNMPv2 でのみ使用されます。
snmp_context_engine_id		コンテキストエンジン ID、SNMP トラップ送信時の MIB オブジェクトの識別子を指定します。指定しない場合、snmp_security_engine_id が使用されます。 SNMPv3 でのみ使用されます。

次のページに続く

表 7 – 前のページからの続き

Parameter name	Value and unit	Description
snmp_context_name		SNMP トラップを送信するときに MIB オブジェクトの識別子であるコンテキスト名を指定します。 SNMPv3 でのみ使用されます。
snmp_enabled	true	SNMP トラップの送信を有効にするか無効にするかを指定します。
snmp_port	162	SNMP トラップの送信に使用する SNMP サーバーポートを指定します。
snmp_privacy_password		snmp_security_name で指定されたセキュリティ名に関連付けられたプライバシーパスワードを指定します。 SNMPv3 にのみ使用されます。
snmp_privacy_protocol	NONE	SNMP トラップのプライバシープロトコルを指定します。可能な値は、NONE、DES、AES128、IDEA、AES192、または AES256 です。 SNMPv3 でのみ使用されます。
snmp_security_engine_i		SNMP サーバー上の SNMP エージェントのエンジン ID を指定します。 SNMPv3 でのみ使用されます。
snmp_security_level	NOAUTH_NOPRIV	セキュリティレベルを指定します。その可能な値は次のとおりです。 AUTH_NOPRIV - 認証、プライバシーなし。その場合、 NONE オプションはプライバシープロトコルでのみ可能です。 AUTH_PRIV - 認証、プライバシー。次に、認証とプライバシープロトコルで NONE オプション選択を許可しないでください。 NOAUTH_NOPRIV - 認証なし、プライバシーなし。その場合、認証とプライバシープロトコルでは NONE オプションのみが可能です。 SNMPv3 でのみ使用されます。
snmp_security_name		SNMP トラップを送信するためのユーザー名またはセキュリティ名を指定します。 SNMPv3 でのみ使用されます。
snmp_server	127.0.0.1	SNMP トラップの送信に使用する SNMP サーバーホストアドレスを指定します。
snmp_spool_retention_ti	7 days	送信されたトラップが破棄されるまでにスプールテーブルに保持する日数を指定します。
webclient_help_pg	EDB hosted documentation	オンラインの PostgreSQL コアドキュメントの場所を指定します。

4.5 Managing Postgres servers

4.5.1 監視対象データベースサーバーの起動と停止

PEM では、PEM クライアントを使用して管理対象サーバーインスタンスを起動およびシャットダウンできます。PEM がサービスを管理できるようにサーバーを構成するには、[Server Registration] ダイアログボックスに値を入力します。ダイアログボックスを使用して、データベースサーバーを PEM エージェントに登録します。これは、データベースサーバーと同じホストにインストールされるローカルエージェントです。次のこともできます。

- [Properties ダイアログボックスで **Store on PEM Server**] オプションを指定します。
- [**Advanced**] タブの **Service ID** フィールドにサービススクリプトの名前を指定します。
  - EDB Postgres Advanced Server の場合、サービス名は `edb-as-<x>` または `ppas-<x>` .
  - PostgreSQL の場合、サービス名は `postgresql-<x>` です。 。

<x> は、サーバーバージョン番号を示します。

サーバーに接続した後、ツリーでサーバー名前を選択し、 **Tools > Queue Server Startup** または **Tools > Queue Server Shutdown** を選択することにより、サーバーをスタートまたは停止できます。

4.5.2 pg_hba.conf ファイルの変更

`pg_hba.conf` ファイルのエントリは、ネットワークの認証と承認を制御します。PEM サーバーホスト上の `pg_hba.conf` ファイルは、PEM サーバーと PEM-HTTPD、PEM エージェント、および監視対象サーバー間の接続を許可する必要があります。

PEM サーバーのインストールプロセスでは、監視対象への PEM のホストの IP アドレスと接続情報の入力を求められます。この情報は、PEM バッキングデータベースの `pg_hba.conf` ファイルの先頭に追加されます。

`pg_hba.conf` ファイルを手動で変更して、PEM サーバーと他のコンポーネント間の接続を許可する必要がある場合があります。たとえば、PEM-HTTPD インストールが PEM サーバーと同じホストにない場合、PEM サーバーホスト上の `pg_hba.conf` ファイルを変更して、PEM-HTTPD がサーバーに接続できるようにする必要があります。

デフォルトでは、`pg_hba.conf` ファイルは Postgres インストールの下の data ディレクトリにあります。たとえば、EDB Postgres Advanced Server 10 ホストでは、`pg_hba.conf` のデフォルトの場所は次のとおりです。

```
/var/lib/edb/as10/data/pg_hba.conf
```

`pg_hba.conf` ファイルは、選択したエディターで変更できます。ファイルを変更した後、サーバーを再起動して変更を有効にします。

次の例は、postgres という名前のユーザーから `pg_hba.conf` が存在するホスト上の `postgres` データベースへの md5 パスワード認証接続を許可する `pg_hba.conf` エントリーを示しています。接続は、192.168.10.102 の IP アドレスから来ています。

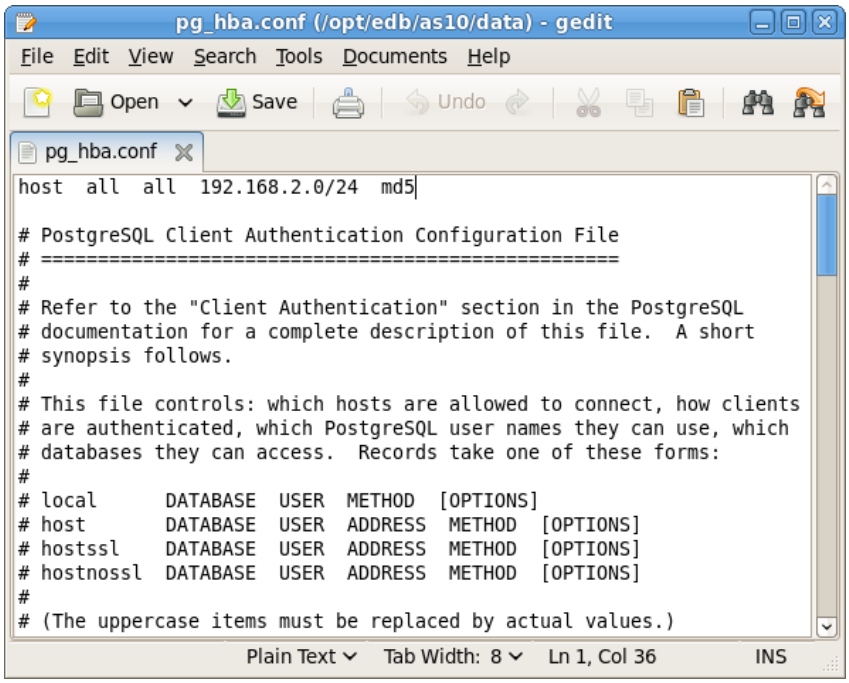


図 2: PEM entries in the pg_hba.conf file

```
#  TYPE     DATABASE        USER        CIDR-ADDRESS          METHOD

#  IPv4 local connections:

 host     postgres        postgres    192.168.10.102/32       md5
```

ネットワークホストのアドレスまたはネットワークアドレスレンジを指定できます。たとえば、アドレス 192.168.10.23、192.168.10.76、および 192.168.10.184 とサーバーからの接続を許可したい場合は、そのネットワーク内のすべてのホストからの接続を許可するように `192.168.10.0/24` の CIDR-ADDRESS を入力します。

```
#  TYPE     DATABASE        USER        CIDR-ADDRESS       METHOD

#  IPv4 local connections:

  host     postgres         all       192.168.10.0/24       md5
```

`pg_hba.conf` ファイルエントリーのフォーマットの詳細については、

PostgreSQL core documentation を参照してください。

PEM を使用して Postgres サーバーに接続する前に、両方のサーバーの `pg_hba.conf` ファイルで接続が許可されていることを保証必要があります。データベースサーバーへの接続時にエラーを受け取った場合は、`pg_hba.conf` ファイルを変更し、接続を許可するエントリーを追加します。

4.5.3 データベースとオブジェクトの作成と保守

Postgres サーバーの各インスタンスは、1 つ以上のデータベースを管理します。各ユーザーは、データベースに含まれる情報にアクセスする前に、データベースに接続するための認証情報を提供する必要があります。PEM クライアントを使用すると、データベースと、テーブル、インデックス、ストアドプロシージャなどのデータベースを構成するオブジェクトを作成および管理できます。

PEM でデータベースを作成するには、管理対象サーバーの **Databases** ノードを右クリックし、 **Create > Database** を選択します。データベースを定義したら、新しいデータベースにオブジェクトを作成できます。

たとえば、テーブルを作成するには、 **Tables** ノードを右クリックし、**Create > Table** を選択します。 [New Table] ダイアログボックスでテーブルの属性を指定します。

PEM は、次のような他のデータベースオブジェクトを作成および管理するための同様のダイアログボックスを提供します。

- テーブル
- インデックス
- ストアドプロシージャ
- ファンクション
- トリガー
- ビュー
- コンストレイン

各オブジェクトタイプはツリーにリストされます。オブジェクトタイプに対応するノードを右クリックして [**Create**] メニューにアクセスし、オブジェクトを作成します。選択したオブジェクトの管理タスクを実行するには、名前付けノードのコンテキストメニューから **Properties** を選択します。

4.5.4 テンプレート Linux サービススクリプト

データベースサーバーのサービススクリプトを使用すると、PEM サーバーはデータベースサーバーを起動、停止、または再起動できます。これは、構成管理、証明書管理、およびその他の管理タスクを実行するときに必要になる場合があります。サービススクリプトはプラットフォーム固有です。

PEM サーバーが存在する Postgres サーバーには、サービススクリプトが含まれている必要があります。EDB によって生成された Windows の Postgres インストーラーは、サービススクリプトを作成します。ネイティブパッケージのような別のソースからの Postgres サーバーを使用している場合、サービススクリプトを提供する必要があります。

注釈: Rocky Linux または RHEL 8.x では、サービススクリプトは `/usr/lib/systemd/system` にあります。

Postgres サービスのカスタマイズについては、

PostgreSQL documentation を参照してください。

4.6 Managing a PEM agent

Postgres Enterprise Manager ブラウザベースのコンソールを使用して PEM エージェントを管理します。

PEM エージェントは、タスクを実行し、エージェントホストおよび監視対象の Postgres インスタンスから PEM サーバーに統計を報告します。単一の PEM エージェントは、エージェントがインストールされている同じホストまたは複数のリモートホストに存在する Postgres の複数のインストール済みインスタンスを監視できます。また、サーバーのシャットダウン、SQL プロファイラーのトレース、カスタムジョブなど、ユーザーがスケジュールした他のタスクを実行する責任もあります。

次のタスクを実行して PEM エージェントを管理できます。

- *Managing job notifications*
- *Managing scheduled jobs*
- *Modifying agent configuration*
- *Reviewing or modifying agent properties*
- *Setting agent privileges*

4.6.1 Managing job notifications

PEM コンソールでは、スケジュールされたタスクまたはカスタムエージェントジョブの下にリストされているシステム生成ジョブの成功または失敗に応じて SMTP トラップを送信する設定を構成できます。これらのメール通知設定を次の 3 つのレベルで構成して、指定したユーザーグループにメール通知を送信できます。これらのレベルは優先度の順に表示されています。

- ジョブレベル
- エージェントレベル
- PEM サーバーレベルデフォルト

ジョブレベルでジョブ通知を設定する

カスタムエージェントジョブにのみ、次のいずれかの方法でジョブレベルでメール通知設定を構成できます。

- 新しいエージェントジョブの場合、ジョブの作成中に、Create Agent Job ウィザードの **Notification** タブでメール通知設定を構成できます。
- 新しいエージェントジョブの場合、ジョブ自分自身の作成中に、`CreateAgent Job` ウィザードの [`Notification`] タブでメール通知設定を構成できます。

Notifications タブを使用して、ジョブレベルでメール通知設定を構成します。

- **Send the notification** フィールドを使用して、メール通知を送信する時期を指定します。
- **Send the notifications** フィールドを使用して、メール通知の送信先のメールグループを指定します。

エージェントレベルでのジョブ通知の設定

ツリービューでエージェントを選択し、右クリックして、**プロパティ**を選択します。[Properties ダイアログボックスで、**Job Notifications** タブを選択します。

[ジョブ通知] タブを使用して、エージェントレベルでメール通知設定を構成します。

- **Overridedefault configuration?** スイッチを使用して、エージェントレベルのジョブ通知設定でデフォルトのジョブ通知設定をオーバーライドするかどうかを指定します。**Yes** を選択して、このダイアログボックスの残りの設定を有効にして、ジョブ通知をいつ、誰に送信するかを定義します。
- **Send the notifications?** スイッチを使用して、ジョブが正常に完了したときにジョブ通知を送信するかどうかを指定します。
- **Send the notifications?** スイッチを使用して、ジョブが失敗したときにジョブ通知を送信するかどうかを指定します。
- **Send the notifications** フィールドを使用して、ジョブ通知の送信先の電子メールグループを指定します。

サーバーレベルでジョブ通知を設定する

[Server Configuration] ダイアログボックスを使用して、PEM サーバーレベルでのメール通知構成に関する情報を提供できます。[Server Configuration] ダイアログボックスを開くには、[**Management > Server Configuration**] を選択します。

4 つのサーバー構成パラメーターは、PEM サーバーレベルでのジョブ通知設定に関する情報を指定します。

- **job_failure_notification** スイッチを使用して、各ジョブが失敗した後にメール通知を送信するかどうかを指定します。
- **job_notification_email_group** パラメーターを使用して、メール通知の送信先のメールグループを指定します。
- **job_retention_time パラメーター**を使用して、システムに非定期的なスケジュールされたタスクを保持する日数を指定します。
- **job_status_change_notification** スイッチを使用して、各ジョブステータスの変更、つまり失敗、成功、または中断の後に電子メール通知を送信するかどうかを指定します。

4.6.2 Managing scheduled jobs

指定した順序で定義した一連のステップを実行する PEM スケジュールされたジョブを作成できます。これらの手順には、エージェントにバインドされたサーバーで実行できる SQL コードまたはバッチ/シェルスクリプトが含まれる場合があります。ビジネス要件に合わせてこれらのジョブをスケジュールできます。たとえば、特定のデータベースサーバーのバックアップを作成するジョブを作成し、毎月の特定の日時に実行するようにスケジュールできます。

PEM スケジュールされたジョブを作成または管理するには、PEM ツリーを使用して、ジョブを作成する PEM エージェントを参照します。ツリーには **Jobs** ノードが表示され、その下に、現在定義されているジョ

ブがリストされます。ジョブを追加するには、**Jobs** ノードを右クリックし、コンテキストメニューから **Create Job** を選択します。

以下のセクションで説明するように、[Create - Agent Job] ダイアログボックスのタブを使用して、PEM スケジュールされたジョブを構成するステップとスケジュールを定義します。

ジョブに関する一般的な情報を提供します

General タブを使用して、ジョブに関する一般的な情報を提供します。

- **Name** フィールドにジョブの名前を入力します。
- ジョブを有効にするには、 **Enabled** スイッチを **Yes** に設定します。ジョブを無効にするには、 **No** に設定します。
- **Comment** フィールドを使用して、ジョブに関するメモを保存します。

ジョブステップを定義する

Steps タブを使用して、ジョブが実行するステップを定義および管理します。 **Add** +を選択してステップを追加します。次に、ヘッダーの左側にある compose アイコンを選択して、Step Definition ダイアログボックスを開きます。

[Step Definition] ダイアログボックスを使用して、ステップを定義します。

- **Name** フィールドにステップの名前を入力します。手順は、名前の英数字オーダー実行されます。
- **Enabled** スイッチを使用して、ジョブの実行時にステップを含める **True** か、ステップを無効にします **False** 。
- **Kind** スイッチを使用して、ジョブステップが SQL コード **SQL** を呼び出すか、バッチスクリプト **Batch** を呼び出すかを示します。
  - **SQL** を選択した場合、**を使用しますステップの** *SQL* **コードを提供する** Code** タブ。
  - **Batch** を選択した場合、 **Code** タブを使用して、ステップ中に実行するバッチスクリプトを提供します。
- **On error** リストを使用して、ステップの実行中にエラーが発生した場合の pgAgent の動作を指定します。以下から選択してください
  - **Fail** —このステップの処理中にエラーが発生した場合は、ジョブを停止します。
  - **Success** —ステップを正常に完了したとしてマークを付けて、続行します。
  - **Ignore** —エラーを無視して続行します。
- **Kind** スイッチの入力として SQL を選択した場合
  - **Server** フィールドを使用して、PEM スケジュールジョブを作成しているエージェントにバインドされたサーバーを指定します。

  - を使用します **Database** フィールドを使用して、選択したサーバーに関連付けられるデータベースを指定します。

- **Comment** フィールドを使用して、ステップに関するコメントを入力します。
- [ステップ定義] ダイアログボックスの [**コード**] タブの状況依存フィールドを使用して、ステップ中に実行する SQL コードまたはバッチスクリプトを提供します。
  - ステップが SQL コードを呼び出す場合、**に *1* つ以上の *SQL* ステートメントを提供します** SQL クエリー** フィールド.
  - ステップがバッチスクリプトを呼び出す場合、 **Code** フィールドにスクリプトを提供します。Windows サーバーで実行している場合は、標準のバッチファイル構文を使用します。Linux サーバーでは、最初の行で適切なインタープリター`#!/bin/sh` などを指定する限り、任意のシェルスクリプトを使用できます。定義されたインラインコードに加えて、ファイルシステム上のバッチスクリプト、シェルスクリプト、または SQL ファイルのパスを提供することもできます。
    * Linux システムでスクリプトを呼び出すには、`agent.cfg` の `batch_script_user` パラメーターのエントリーを変更する必要がありますファイルを開き、スクリプトを実行するユーザーを指定します。このパラメーターには、非 root ユーザーまたは root を指定できます。ユーザーを指定しない場合、または指定されたユーザーが存在しない場合、スクリプトは実行されません。ファイルを変更した後、エージェントを再起動します。
    * Windows システムでスクリプトを呼び出すには、`AllowBatchJobSteps` のレジストリエントリを `true` に設定し、PEM エージェントを再起動します。PEM レジストリエントリは `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent` にあります。

ステップで必要な情報をすべて入力したら、[**Save**] を選択します。

ジョブスケジュールを定義する

Add (+) を選択して各ステップを追加するか、**Schedules** タブを選択してジョブスケジュールを定義します。

[**Schedules**] タブで、[**Add** +] を選択してジョブのスケジュールを追加します。次に、ヘッダーの左側にある作成アイコンを選択して、Schedule Definition ダイアログボックスを開きます。

[**Schedules configuration**] タブを使用して、ジョブを実行する日と時刻を指定します。

- **Name** フィールドにスケジュールの名前を入力します。
- **Enabled** スイッチを使用して、pgAgent がスケジュールを使用するか **Yes** か、スケジュールを無効にするか **No** を示します。
- **Start** フィールドのカレンダーセレクターを使用して、スケジュールの開始日時を指定します。
- **End** フィールドのカレンダーセレクターを使用して、スケジュールの終了日時を指定します。
- **Comment** フィールドを使用して、スケジュールに関するコメントを入力します。

Repeat タブを選択して、スケジュールが cron スタイルの形式で実行日を定義します。ジョブは、[**Repeat**] タブで選択された各日付または時刻要素に実行されます。

各フィールドで、値を選択して、フィールドで選択した値のリストに追加します。フィールドから値をクリアするには、フィールドの右側にある X を選択します。

- **Days** ボックスのフィールドを使用して、ジョブを実行日を指定します。
  - **Week Days** フィールドを使用して、ジョブを実行日を選択します。
  - **Month Days** フィールドを使用を押して、ジョブを実行する数値を選択します。日付に関係なく、月の最終日にジョブを実行するには、**Last Day** を選択します。
  - **Months** フィールドを使用して、ジョブを実行する月を選択します。
- **Times** ボックスのフィールドを使用して、ジョブを時と分単位で実行する時間を指定します。

Exceptions タブを選択して、スケジュールを実行したくない日を指定します。たとえば、休日にはジョブを実行したくない場合があります。

Add (+) を選択して、例外テーブルに行を追加します。次に

- **Date** 列で、カレンダーセレクターを開き、ジョブを実行したくない日付を選択します。**** <Any> **** を指定して、選択した日の時刻にジョブを実行しないことを示します。
- **Time** 列で、時間セレクターを開き、ジョブを実行したくない時間を指定します。**** <Any> **** を指定して、選択した日の任意の時点でジョブを実行しないことを示します。

メール通知を設定する

Notifications タブを選択して、ジョブレベルでメール通知設定を構成します。

- **Send the notification** フィールドを使用して、メール通知を送信する時期を指定します。
- **Send the notifications** フィールドを使用して、メール通知の送信先のメールグループを指定します。

スケジュールの定義が終了したら、[**SQL**] タブを使用して、ジョブを作成または変更するコードを確認できます。

Save を選択して、ジョブ定義を保存します。

ジョブを保存すると、ジョブは、定義されたサーバーの PEM ツリーの **Jobs** ノードの下にリストされます。PEM コンソールの **プロパティ**タブには、選択したジョブの高レベルの概要が表示され、 **統計**タブには、ジョブの各実行の詳細が表示されます。既存のジョブを変更するか、ジョブに関する詳細情報を確認するには、ジョブ名を右クリックし、コンテキストメニューから**プロパティ** を選択します。

4.6.3 Modifying agent configuration

多数の構成可能なパラメーターが PEM エージェントの動作を制御します。Linux では、これらは構成ファイルに保存されます。Windows、これらはレジストリキーとして保存されます。

エージェント構成のほとんどは自動的に管理されます。特に指示されない限り、これらの構成パラメーターを手動で変更しないことをお勧めします。

注釈: パラメーターを追加、削除、または変更した場合、エージェントを再起動して変更を適用する必要があります。

構成ファイルの内容

Linux システムでは、PEM 構成オプションは/usr/edb/pem/agent/etc にある agent.cfg ファイルに保存されます。 agent.cfg ファイルには、次の表に示すエントリが含まれています。

Parameter name	Description	Default value
pem_host	PEM サーバーの IP アドレスまたはホスト名。	127.0.0.1.
pem_port	エージェントが PEM サーバーと通信するために接続するデータベースサーバポート。	Port 5432.
pem_agent	PEM エージェントに割り当てられる一意の識別子。	The first agent is '1', the second agent is '2', and so on.
agent_ssl_key	PEM エージェントのキーファイルへの完全なパス。	/root/.pem/agent.key
agent_ssl_crt	PEM エージェントの証明書ファイルへの完全なパス。	/root/.pem/agent.crt
agent_flag_dir	HA サポートに使用されます。別のサーバーの監視を引き継ぐ要求についてチェックされるディレクトリパスを指定します。要求は、指定されたフラグディレクトリにファイルの形式で作成されます。	Not set by default.
log_level	PEM ログファイルに書き込むイベントのタイプを`debug2`、*debug*、*info*、*warning*、`error`のいずれか指定します。これらはロギングの詳細度の降順です。 `debug2`は可能なすべてをログに記録し、`error`はエラーのみをログに記録します。	warning
log_location	PEM ワーカーログファイルの場所を指定します。	127.0.0.1.
agent_log_location	PEM エージェントログファイルの場所を指定します。	/var/log/pem/agent.log
long_wait	最初の接続試行が失敗した場合に、PEM サーバーに接続しようとする前に PEM エージェントが待機する最大時間秒単位。	30 seconds
short_wait	キューで次に実行を待機しているプローブを確認するまでに PEM エージェントが待機する最小時間秒単位。	10 seconds
alert_threads	エージェントによって生成されるアラートスレッドの数。詳細については、 アラートスレッドについて を参照してください。	Set to 1 for the agent that resides on the host of the PEM server, 0 for all other agents.
enable_smtp	true に設定されている場合、このエージェントは PEM Web アプリケーションで構成されたとおりにメール通知を送信しようとします。	true for PEM server host, false for all others.
enable_snmp	true に設定されている場合、このエージェントは PEM Web アプリケーションで構成されたとおりに SNMP 通知を送信しようとします。	true for PEM server host, false for all others.
enable_nagios	true に設定すると、Nagios アラートが有効になります。	true for PEM server host, false for all others.

次のページに続く

表 8 – 前のページからの続き

Parameter name	Description	Default value
enable_webhook	true に設定すると、Webhook アラートが有効になります。	true for PEM server host, false for all others.
max_webhook_retries	失敗した場合に pemAgent が Webhook の呼び出しを再試行する最大回数を設定するために使用されます。	Default 3.
connect_timeout	エージェントが接続を待機する秒単位の最大時間 10 進整数文字列。	Not set by default. Set to 0 to indicate for the agent to wait indefinitely.
allow_server_restart	TRUE に設定されている場合、エージェントは監視対象のデータベースサーバーを再起動できます。このパラメーターの値に応じて、一部の PEM 機能が有効/無効になる場合があります。	False
max_connections	接続スロットラーが使用するプローブ接続の最大数。	0 (an unlimited number)
connection_lifetime	ConnectionLifetime または connection_lifetime を使用して、開いているがアイドル状態の接続が保持される最小秒数を指定します。 MaxConnections で指定された値に到達し、待機中の要求を満たすために別のデータベースへの新しい接続が必要な場合、このパラメーターは無視されます。	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
allow_batch_probes	TRUE に設定されている場合、ユーザーはカスタムプローブ機能を使用してバッチプローブを作成できます。	false
heartbeat_connection	TRUE に設定すると、ハートビートの送信に専用接続が使用されます。 TRUE に設定している場合、`max_connections`パラメーターを 1 より大きい値に設定する必要があります。	false
batch_script_dir	スクリプトファイルアラート用が保存されるパスを提供します。	/tmp
connection_custom_setup	監視対象サーバーとの新しい接続が作成されたときに呼び出される SQL コードを提供するために使用されます。	Not set by default.
ca_file	Webhook サーバー証明書を検証するためにプラットフォームのデフォルトの代わりに使用する CA 証明書へのパス。 Webhook を定義するときに`--webhook_ssl_ca_crt`オプションでこの値をオーバーライドできます。	Not set by default.
batch_script_user	バッチ/シェルスクリプトの実行に使用するユーザーの名前。	None

次のページに続く

表 8 – 前のページからの続き

Parameter name	Description	Default value
Webhook parameters	次のオプションは複数回指定できます。毎回、オプションの前に`[WEBHOOK/<name>]`フォームのヘッダーを配置します。`<name>`は、以前に作成した Webhook の名前です。これらの設定は、Webhook の作成時に自動的に追加されます。手動で追加することはお勧めしません。	
webhook_ssl_key	Webhook の SSL クライアントキーファイルへの完全なパス。	
webhook_ssl_crt	Webhook の SSL クライアント証明書ファイルへの完全なパス。	
webhook_ssl_crl	Webhook サーバー証明書を検証するための CRL ファイルの完全なパス。	
webhook_ssl_ca_crt	Webhook の SSL ca 証明書ファイルへの完全なパス。	
allow_insecure_webhooks	true に設定されている場合、Webhook が安全でないフラグを使用して呼び出すことを許可します。	false

レジストリの内容

Windows システムでは、PEM レジストリエントリは次の場所にあります。

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

レジストリには、次の表に示すエントリが含まれています。

Parameter name	Description	Default value
PEM_HOST	PEM サーバーの IP アドレスまたはホスト名。	127.0.0.1.
PEM_PORT	エージェントが PEM サーバーと通信するために接続するデータベースサーバポート。	Port 5432.
AgentID	PEM エージェントに割り当てられる一意の識別子。	The first agent is '1', the second agent is '2', and so on.
AgentKeyPath	PEM エージェントのキーファイルへの完全なパス。	%APPDATA%\Roaming\pem\agent.key.
AgentCrtPath	PEM エージェントの証明書ファイルへの完全なパス。	%APPDATA%\Roaming\pem\agent.crt

次のページに続く

表 9 – 前のページからの続き

Parameter name	Description	Default value
AgentFlagDir	HA サポートに使用されます。別のサーバーの監視を引き継ぐ要求についてチェックされるディレクトリパスを指定します。要求は、指定されたフラグディレクトリにファイルの形式で作成されます。	Not set by default.
LogLevel	PEM ログファイルに書き込むイベントの種類を指定します。これらはロギングの詳細度の降順です。 `debug2`は考えられるすべてのログを記録し、 `error`はエラーのみをログに記録します。	warning
LongWait	最初の接続試行が失敗した場合に、PEM サーバーに接続を試行する前に PEM エージェントが待機する最大時間秒単位。	30 seconds
shortWait	どのプローブが次にキューにある実行待機中かを確認するまでに PEM エージェントが待機する秒単位の最小時間。	10 seconds
AlertThreads	エージェントが生成するアラートスレッドの数。詳細については、 アラートスレッドについて を参照してください。	Set to 1 for the agent that resides on the host of the PEM server, 0 for all other agents.
EnableSMTP	true に設定すると、SMTP メール機能が有効になります。	true for PEM server host, false for all others.
EnableSNMP	true に設定すると、SNMP トラップ機能が有効になります。	true for PEM server host, false for all others.
EnableWebhook	true に設定すると、Webhook アラートが有効になります。	true for PEM server host, false for all others.
MaxWebhookRetries	失敗した場合に pemAgent が Webhook の呼び出しを再試行する最大回数を設定します。	Default 3.
ConnectTimeout	エージェントが接続を待機する秒単位の最大時間 10 進整数文字列。	Not set by default. If set to 0, the agent waits indefinitely.
AllowServerRestart	TRUE に設定されている場合、エージェントは監視対象のデータベースサーバーを再起動できます。このパラメーターの値に応じて、一部の PEM 機能が有効/無効になる場合があります。	true
MaxConnections	接続スロットラーが使用するプローブ接続の最大数。	0 (an unlimited number)
ConnectionLifetime	ConnectionLifetime または connection_lifetime を使用して、開いているがアイドル状態の接続が保持される最小秒数を指定します。 MaxConnections で指定された値に到達し、待機中の要求を満たすために別のデータベースへの新しい接続が必要な場合、このパラメーターは無視されます。	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).

次のページに続く

表 9 – 前のページからの続き

Parameter name	Description	Default value
AllowBatchProbes	TRUE に設定されている場合、ユーザーはカスタムプローブ機能を使用してバッチプローブを作成できます。	false
HeartbeatConnection	TRUE に設定すると、ハートビートの送信に専用接続が使用されます。	false
BatchScriptDir	アラート用のスクリプトファイルを保存するパス。	/tmp
ConnectionCustomSetup	監視対象サーバーとの新しい接続が作成されたときに呼び出される SQL コードを提供するために使用されます。	Not set by default.
AllowBatchJobSteps	true に設定されている場合、バッチ/シェルスクリプトが実行されます。スクリプトは、PEM エージェントが実行されているユーザーアカウントによって実行されます。	None
Webhook parameters	次のオプションは複数回指定できます。これらはサブキー `SoftwareEnterpriseDBPEMagentWEBHOOK<name>`に配置する必要があります。`<name>`は、以前に作成した Webhook の名前です。これらの設定は、Webhook の作成時に自動的に追加されます。手動で追加することはお勧めしません。	
WebhookSSLKey	Webhook の SSL クライアントキーファイルへの完全なパス。	
WebhookSSLCrt	Webhook の SSL クライアント証明書ファイルへの完全なパス。	
WebhookSSLCrl	Webhook サーバー証明書を検証するための CRL ファイルの完全なパス。	
WebhookSSLCaCrt	Webhook の SSL ca 証明書ファイルへの完全なパス。	
AllowInsecureWebhooks	true に設定されている場合、Webhook が安全でないフラグを使用して呼び出すことを許可します。	false

アラートスレッドについて

エージェントによって生成されたアラートスレッドの数は、 `alert_threads` または `AlertThreads` パラメーターによって決まります。通常、PEM サーバーにあるエージェントではこのパラメーターを 1 に設定し、他のすべてのエージェントは 0 に設定することをお勧めします。ただし、非常に多数のアラートを持つ PEM サーバーインスタンスでは、多くの監視対象サーバー、多くの有効なアラート、または高いアラート頻度が原因で、構成された頻度でアラートが評価されていない場合、このパラメーターを増やす必要がある場合があります。この状況では、PEM サーバーにあるエージェントでこのパラメーターを約 8 に設定し、他のすべてのエージェントで 0 を設定し、それに応じてチューニングアップまたはダウンすることをお勧めします。

このパラメーターを調整するときは、すべてのエージェントが任意のアラートを処理できることに注意してください。一般に、複数のエージェントでゼロ以外の数のアラートスレッドを使用する必要はありません。アラートを処理する PEM インスタンスの容量は、すべてのエージェントのアラートスレッドの合計数によって決まります。特定のエージェントのスレッド数を増やしても、そのエージェントによって監視されているサーバーに関連するアラートのパフォーマンスは追加されません。

各アラートスレッドは PEM サーバーバックエンドへの接続を開くため、必要以上に多くのスレッドを割り当てると、PEM サーバーのメモリと CPU の使用量が追加されます。

4.6.4 Reviewing or modifying agent properties

[PEM Agent Properties] ダイアログボックスは、ダイアログボックスが開かれた PEM エージェントに関する情報を提供します。ダイアログボックスを開くには、PEM クライアントツリーでエージェント名を右クリックし、コンテキストメニューから**プロパティ** を選択します。

[PEM Agent Properties] ダイアログボックスを使用して、PEM エージェントに関する情報を確認または変更します。

- **Description** フィールドには、PEM エージェントの変更可能な説明が表示されます。この説明は、PEM クライアントのツリーに表示されます。
- グループを使用して、PEM クライアントツリーでサーバーとエージェントを編成できます。**Group** リストを使用して、エージェントが表示されるグループを選択します。
- **Team** フィールドを使用して、エージェントによって監視されるサーバーにアクセスできるグループロールの名前を指定します。このエージェントによって監視されるサーバーは、接続しているチームメンバーに PEM クライアントツリーで表示されます。これは便利な機能です。**チーム**フィールドは、真の分離を提供しません。セキュリティ目的には使用しないでください。
- **Heartbeat interval** フィールドには、PEM エージェントから PEM サーバーまでで経過時間が表示されレポートされます。間隔を変更するには、**Minutes** または **Seconds** フィールドの横にあるセレクターを使用し間隔を調整します。

ジョブ通知タブを使用して、エージェントレベルでメール通知設定を構成します。

- **Overridedefault configuration?** スイッチを使用して、エージェントレベルのジョブ通知設定でデフォルトのジョブ通知設定をオーバーライドするかどうかを指定します。**Yes** を選択して、このダイアログボックスの残りの設定を有効にして、ジョブ通知をいつ、誰に送信するかを定義します。- **Send the notifications?** スイッチを使用して、ジョブが正常に完了したときにジョブ通知を送信するかどうかを指定します。- **Send the notifications?** スイッチを使用して、ジョブが失敗したときにジョブ通知を送信するかどうかを指定します。- **Send the notifications** フィールドを使用して、ジョブ通知の送信先の電子メールグループを指定します。

Agent Configurations タブには、エージェントの現在の構成と機能がすべて表示されます。

- **Parameter** 列には、パラメーターのリストが表示されます。 - **Value** 列には、対応するパラメーターの現在値が表示されます。 - **Category** 列には、対応するパラメーターのカテゴリが表示されます。これは、**構成** または **機能** のいずれかです。

4.6.5 Setting agent privileges

デフォルトでは、PEM エージェントは、オペレーティングシステムホストの root 権限とデータベースサーバーのスーパーユーザー権限でインストールされます。これらの特権により、PEM エージェントは、システムの使用状況について監視対象ホストおよびデータベースサーバーで無制限のプローブを呼び出し、情報を取得して PEM サーバーに返すことができます。

root ユーザーと非 root ユーザー

PEM エージェントの権限が低下すると、PEM 機能が低下します。完全な機能を実現するには、root として PEM エージェントを実行します。PEM エージェントがデータベースサーバーのサービスアカウントで実行されている場合、PEM プローブはレポートの生成に使用される統計情報に完全にアクセスできず、機能はそのアカウントの機能に制限されます。PEM エージェントが別の権限の低いアカウントで実行されている場合、機能はさらに制限されます。

Feature name	Works with root user	Works with non-root user	Works with remote PEM agent
Audit Manager	yes	The Audit Log Manager might not be able to apply requested modifications if the service can't be restarted. The user running the PEM agent might be different from the user who owns the data directory of the database server. Thus the user running the PEM agent might not be able to change the configuration and also might not be able to restart the services of the database server.	no
Capacity Manager	yes	yes	yes Note: There's no correlation between the database server and operating system metrics

次のページに続く

表 10 – 前のページからの続き

Feature name	Works with root user	Works with non-root user	Works with remote PEM agent
Log Manager	yes	The Log Manager might not be able to apply requested modifications if the service can't restart. The user running the PEM agent might be different from the user who owns the data directory of the database server. Thus the user running the PEM agent might not be able to change the configuration and also might not be able to restart the services of the database server.	no
Manage Alerts	yes	yes	yes Note: When Run Alert Script on the database server is selected, it runs on the machine where the bound PEM agent is running and not on the actual database server machine.
Manage Charts	yes	yes	yes
Manage Dashboards	yes	Some dashboards might not be able to show complete data. For example, columns such as swap usage, CPU usage, IO read, and IO write appear as 0 in the session activity dashboard.	Some dashboards might not be able to show complete data. For example, the operating system information of the database server doesn't appear as not available.
Manage Probes	yes	Some of the PEM probes don't return information, and some of functionality might be affected.	Some of the PEM probes don't return information, and some of the functionality might be affected.
Postgres Expert	yes	The Postgres Expert can access the configuration expert and schema expert but not the security expert.	The Expert provides partial information, as operating system information isn't available.
Postgres Log Analysis Expert	yes	The Postgres Log Analysis Expert might not be able to do the analysis, as it depends on the logs imported by Log Manager, which doesn't work as required.	The Postgres Log Analysis Expert can't do the analysis, as it depends on the logs imported by Log Manager, which doesn't work as required.

次のページに続く

表 10 – 前のページからの続き

Feature name	Works with root user	Works with non-root user	Works with remote PEM agent
Scheduled Tasks	yes	For Linux, if the user is the same as batch_script_user in *agent.cfg*, then the shell script runs.	Scheduled tasks work only for the database server. Scripts run on a remote agent.
Tuning Wizard	yes	The Tuning Wizard can't run if the service can't restart. The user running the PEM agent might be different from the user who owns the data directory of the database server. Thus the user running the PEM agent might not be able to change the configuration and also might not be able to restart the services of the database server.	no
System Reports	yes	yes	yes
Core Usage Reports	yes	yes	The Core usage report doesn't show complete information. For example, the platform, number of cores, and total RAM aren't displayed.
Managing BART	yes	BART and the BART scanner might not be able to start or reload.	no Note: BART requires passwordless authentication between two machines where database server and BART are installed.

オペレーティングシステムの権限の制限によって影響を受ける機能

PEM エージェントのオペレーティングシステム権限を制限すると、一部の PEM プローブは情報を結果ず、次の機能が影響を受ける可能性があります。

注釈: このリストは包括的なものではありませんが、制限されている機能の種類の概要を提供します。

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.

次のページに続く

表 11 – 前のページからの続き

Probe or action	Operating system	PEM functionality affected
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.

データベース特権の制限によって影響を受ける機能

PEM エージェントのデータベース特権を制限すると、次の PEM 機能が影響を受ける可能性があります。

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.

エラー処理

プローブは、十分な権限なしでオペレーティングシステムを検索している場合、プローブは `permission denied` エラーを結果ことができます。プローブが十分な権限なしでデータベースを照会している場合、プローブは `permission denied` エラーを返すか、返されたデータを空の値として PEM チャートまたはグラフに表示する場合があります。

プローブが失敗すると、プローブの名前、プローブが失敗した理由、問題の解決に役立つヒントを含むエントリがログファイルに書き込まれます。

サーバーで発生したプローブ関連のエラーをプローブログダッシュボードで表示したり、PEM ワーカーログファイルのエラーメッセージを確認したりできます。Linux では、ログファイルのデフォルトの場所は次のとおりです。

```
/var/log/pem/worker.log
```

Windows、 Event Viewer でログ情報を利用できます。

4.7 Managing SSL certificates

PEM は自己署名 SSL 証明書を使用します。

- ユーザーインターフェイスと REST API を提供する *Web* サーバーと *PEM* バックエンドデータベースが個別にインストールされている場合の構成後の手順 Apache httpd への要求を保護するため。
- *PEM* バックエンドデータベースサーバーとエージェントの接続証明書 を保護および認証するため。

4.7.1 Web サーバー証明書

PEM は、Apache httpd サーバーの自己署名 SSL 証明書を使用します。 Apache httpd サーバーの自己署名 SSL 証明書とキーファイルは、PEM のインストール中に生成されます。

PEM に独自の SSL 証明書を使用するには、Apache HTTP 構成ファイル `edb-ssl-pem.conf` を更新します。

PEM VirtualHost セクションで、これらの 2 つの SSL 指示子を更新します。

- `SSLCertificateFile` は、証明書ファイル、例、`your_domain_name.crt` です。
- `SSLCertificateKeyFile` は、証明書署名要求 CSR、`your_private.key` などを作成したときに生成された `.key` ファイルです。

例、次の更新を行います。

```
#  Change the server name and file names in the configuration file to

#  match your certificate files.

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

セキュリティを強化するために、PEM で httpd 自己署名 SSL 証明書を信頼できる CA 署名証明書に置き換えることができます。詳細については、
Replacing httpd self-signed SSL certificates を参照してください。

4.7.2 PEM バックエンドデータベースサーバーとエージェントの接続証明書

デフォルトでは、PEM は PEM エージェントとバックエンドデータベース間で安全な SSL/TLS 接続を実装します。また、独自の認証局 CA として機能して、PEM サーバーとエージェントの証明書とキーを生成します。PEM サーバーとエージェントの自己署名 SSL 証明書とキーは、PEM のインストール中に生成されます。これらの証明書とキーは、エージェントからサーバーへの接続を暗号化します。さらに、PEM エージェントは、パスワードではなく証明書を使用して認証されます。

PEM 自己署名 SSL 証明書とキーを、信頼できる CA 証明書とキーに置き換えることができます。詳細については、**信頼できる *CA* によって署名された証明書とキーを使用する** を参照してください。

PEM 自己署名 SSL 証明書の仕組み

PEM サーバー構成スクリプトは、PEM バックエンドデータベースサーバーの自己署名 SSL 証明書とキーファイルを生成します。バックエンドデータベースサーバーは、これらの証明書とキーを使用して、エージェント接続を認証および暗号化します。各証明書には有効期限があります。証明書の有効期限が近づくと、証明書を再生成します。詳細については、*Regenerating the server self-signed SSL certificates* を参照してください。

PEM エージェントは、libpq インターフェイスを使用して PEM バックエンドデータベースサーバーに接続し、バックエンドデータベースサーバーのクライアントとして機能します。エージェントの自己署名 SSL 証明書とキーは、*Registering a PEM agent* 中に生成されます。PEM エージェントは、自己署名 SSL 証明書とキーファイルを使用して、PEM バックエンドデータベースサーバーとの接続を確立します。

各エージェントには一意の識別子があり、エージェントの証明書とキーには対応する識別子があります。各証明書には有効期限があります。証明書の有効期限が近づくと、証明書を再生成します。詳細については、*Regenerating the agent self-signed SSL certificates* を参照してください。

必要に応じて、エージェントごとに 1 つの証明書を使用するのではなく、すべてのエージェントに同じ証明書を使用できます。詳細については、**共通エージェント証明書とキーペアを生成する** を参照してください。

SSL 証明書を使用して Postgres で接続する方法の詳細については、
Securing TCP/IP connections with SSL を参照してください。

証明書とキーファイルの生成

PEM サーバーは、バックエンドデータベースサーバーのデータディレクトリに証明書とキーファイルを生成します。

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

ca_certificate.crt および ca_key.key ファイルは、エージェント登録プロセス中に使用され、エージェントの SSL 証明書とキーファイルを生成します。

root.crt ファイルは、ca_certificate.crt ファイルのコピーです。postgresql.conf ファイルで ssl_ca_file パラメーターを root.crt として設定することにより、バックエンドデータベースサーバーのルート証明書を使用します。

root.crl には、実際のまたは割り当てられた有効期限の前に発行 CA によって失効されたデジタル証明書の証明書失効リスト CRL があります。

server.crt ファイルは PEM サーバーの署名された証明書であり、server.key ファイルは証明書のプライベートキーです。PEM エージェント証明書は、これらのサーバー証明書とキーファイルを使用して生成されます。

PEM 自己署名 SSL 証明書の更新

PEM サーバーと共にインストールされる PEM エージェントは、ca_certificate.crt ファイルの有効期限を監視します。証明書の有効期限が間もなくエキスパイアと、PEM は次のことを行います。

- 既存の証明書ファイルのバックアップを作成します
- 新しい証明書ファイルを作成し、新しい CA 証明書ファイルを PEM サーバー上の root.crt ファイルに追加します
- アクティブなエージェントの証明書ファイルを更新するジョブを作成します
- PEM サーバーを再起動します

4.7.3 共通エージェント証明書とキーペアを生成する

エージェントごとに 1 ユーザーデフォルトではなく、すべての PEM エージェントに単一の Postgres ユーザーを作成して使用することにより、すべてのエージェントに同じ証明書を使用できます。

ユーザーを作成し、エージェント証明書とキーペアを生成し、すべての PEM エージェントにそれらを使用します。

1. PEM バックエンドデータベースに 1 つの共通エージェントユーザーを作成します。pem_agent ロールをユーザーに付与します。

```
# Running as enterprisedb
psql -p 5444 -U enterprisedb -d pem
CREATE USER pem_agent_common_user;
GRANT pem_agent TO pem_agent_common_user;
```

1. エージェントキーを生成します。

```
# Running as root
openssl genrsa -out agent.key 4096
```

1. エージェントの CSR を生成します。

```
openssl req -new -key agent.key -out agent.csr -subj /C=IN/ST=MH/L=Pune/O=PEM/CN=
↪<agent_user>
```

-subj は、要件に従って提供されます。

1. openssl x509 コマンドを使用して CSR に署名し、エージェント証明書を生成します。

```
openssl x509 -req -days 365 -in agent.csr -CA ca_certificate.crt -CAkey ca_key.key -
↪CAcreateserial -out agent.crt
```

1. agent.crt および agent.key ファイルのアクセス許可を変更します。

```
chmod 600 agent.crt agent.key
```

1. 次のエージェント証明書とキーペアを使用します。

- 新しい PEM エージェントをリモートホストから PEM サーバーに登録するため。

a. エージェント証明書とキーペアをリモートエージェントホストにコピーし、エージェントを登録します。

```
export PEM_SERVER_PASSWORD=edb

/usr/edb/pem/agent/bin/pemworker --register-agent \
--pem-server 192.168.99.130 \
--pem-user enterprisedb \
--pem-port 5444 \
--pem-agent-user pem_agent_common_user \
-o agent_ssl_crt= agent.crt \
-o agent_ssl_key= agent.key
```

b. pemagent サービスを有効にして開始します。

```
systemctl enable pemagent
systemctl start pemagent
```

- エージェント証明書とキーペアを登録済みのエージェントに置き換えます。a. エージェントホストの agent.cfg ファイルの agent_user 、agent_ssl_key 、および agent_ssl_crt パラメーターを編集します。

```
        vi /usr/edb/pem/agent/etc/agent.cfg
        # Edit the agent username
        agent_user=pem_agent_common_user
        # Edit the ssl parameters with new certificate and key file location
        agent_ssl_key=<new_location>/agent.key
        agent_ssl_crt=<new_location>/agent.crt
``` b. pemagent サービスを再起動します。
```

(次のページに続く)
```

(前のページからの続き)

```shell
      systemctl restart pemagent
```

4.7.4 信頼できる CA によって署名された証明書とキーを使用する

PEM 自己署名 SSL 証明書とキーを、信頼できる CA によって署名された証明書とキーに置き換えます。

信頼できる CA 証明書とキーを取得した後、**自己署名サーバー *SSL* 証明書を、信頼できる *CA* によって署名された証明書に置き換えます。** および **自己署名エージェント *SSL* 証明書を、信頼できる *CA* によって署名された証明書に置き換えます。** 証明書とキーを置き換えます。

自己署名サーバー SSL 証明書を、信頼できる CA によって署名された証明書に置き換えます。

1. 古いサーバー証明書とキーファイルをバックアップします。

```
# Running as root
mkdir /var/lib/edb/as<x>/data/certs
cd /var/lib/edb/as<x>/data/
mv server. *root.*  ca_* /var/lib/edb/as<x>/data/certs
```

1. サーバーのプライベートキーを生成します。

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

1. サーバーの CSR を生成します。

edb_notranlate_12 -subj は、要件に従って提供されます。 CN には、PEM サーバーホストのホスト名またはドメイン修飾フルネームを使用することをお勧めします。

1. 信頼できる CA から CA 証明書 trusted_ca.crt を取得します。
2. CA に CSR に署名し、サーバー証明書を生成するように依頼します。
3. 新しいサーバー証明書の詳細が改ざんされていないことを確認し、指定した詳細と一致することを確認します。

```
openssl x509 -noout -text -in server.crt
```

1. CA から取得した新しい証明書を root.crt ファイルとして使用します。

```
cp trusted_ca.crt root.crt
```

1. 信頼できる CA が CRL を提供しない場合、サーバーによる CRL の使用を無効にします。CRL の使用を無効にするには、postgresql.conf ファイルの ssl_crl_file パラメーターをコメント化します。

注釈: 誤って以前の CA からの CRL を適切な場所に残し、*ssl_crl_file* をコメントアウトしない場合、サーバーは起動しますが、認証は SSL エラーメッセージ `tlsv1 alert unknown ca` で失敗します。エラーは、CRL が原因であることを指定していないため、これはコンテキスト外で発生した場合、デバッグが困難になる場合があります。

1. 新しい root.crt 、server.key 、および server.crt ファイルをバックエンドデータベースサーバーのデータディレクトリにコピーします。

```
cp root.crt server.key server.crt /var/lib/edb/as<x>/data
```

1. 新しい証明書とキーファイルの所有者と権限をデータディレクトリと同じになるように変更します。

edb_notranlate_16 .. Note::
今は PEM サーバーを再起動しないでください。 PEM サーバーを再起動すると、登録されているすべてのエージェントが動作を停止します。

1. 各 PEM エージェントの SSL 証明書を、信頼できる CA 証明書に置き換えます。詳細については、これらの**自己署名エージェント *SSL* 証明書を、信頼できる *CA* によって署名された証明書に置き換えます。** を参照してください。

2. PEM サーバーを再起動します。

- Linux の場合

```
# Running as root
systemctl start edb-as-<x>
```

バックエンドデータベースサーバーを再起動すると、PEM サーバーが再起動されます。

自己署名エージェント SSL 証明書を、信頼できる CA によって署名された証明書に置き換えます。

自己署名サーバー証明書 server.crt および server.key および CA 証明書 root.crt を置き換えた後にのみ、自己署名エージェント SSL 証明書を置き換えます。

1. psql を使用して、SSL 証明書の置き換えに必要なすべてのエージェント識別子 ID を見つけます。

```
psql -U enterprisedb -d pem --no-psqlrc -t -A -c "SELECT id FROM pem.agent WHERE␣
↪active=true"
```

1. 2. キーファイルを必要とするエージェントを特定した後、各エージェントに対して agent<ID>.key を生成します。

edb_notranlate_19 <ID> はエージェント識別子です。

1. 各エージェントの CSR を生成します。

edb_notranlate_20 -subj は、要件に応じて提供されます。 CN の<ID> を適切なエージェント識別子に置き換えます。

注釈: エージェントごとに 1 つではなく、すべての PEM エージェントに単一の証明書を使用する場合、共通の Postgres ユーザーを作成し、 *ID* の代わりにこのユーザー名を指定します。 共通エージェント証明書とキーペアを生成する を参照してください。

1. CA に CSR に署名し、エージェント証明書を生成するように依頼します。
2. 各ホストに証明書とキーファイルをコピーします。ここで、 <ID> は、/usr/edb/pem/agent/etc/agent.cfg ファイルの agent_id と一致します。
3. 新しい agent<ID>.crt および agent<ID>.key ファイルの所有権と権限を変更します。

```
chown root agent<ID>.crt agent<ID>.key
chmod 600 agent<ID>.crt agent<ID>.key
```

1. 古いエージェント証明書とキーファイルをバックアップします。

```
# Running as root
mkdir root/.pem/certs
mv root/.pem/agent<ID>.* root/.pem/certs
```

1. 各エージェントの証明書とキーファイルを、新しく生成されたファイルに置き換えます。

```
cp agent<ID>.key agent<ID>.crt root/.pem
```

7. PEM エージェントサービスを再起動します。

- Linux の場合

```
# Running as root
systemctl restart pemagent
```

- Windows の場合サービスアプレットを使用して PEM エージェントを再起動します。PEM エージェントサービスの名前は Postgres Enterprise Manager Agent です。 [Services] ダイアログボックスでサービス名を選択し、[**Restart the service**] を選択します。

4.7.5 証明書のテスト

認証の問題が発生した場合は、これらのテストを使用して証明書を検証できます。

ルート証明書と照合して証明書を検証する

サーバーの root.crt に従って PEM エージェント証明書が信頼されるかどうかを確認するには、両方の証明書を同じマシンにコピーします。次に、次のコマンドを実行します。

```
openssl verify -verbose -CAfile root.crt  agent1.crt
```

このコマンドは、成功した場合は agent1.crt: OK 、失敗した場合は説明メッセージを返します。

PEM バックエンドデータベースへのテスト接続を作成する

エージェントユーザーが証明書を使用して接続できるかどうかを確認するには、エージェントが存在するサーバーで、root として次のコマンドを実行します。

```shell PGHOST= PGPORT= PGUSER=agent PGSSLCERT=/root/.pem/agent.crt PGSSLKEY=/root/.pem/agent.key PGSSLMODE=require

export PGHOST PGPORT PGUSER PGSSLCERT PGSSLKEY PGSSLMODE

-A -t -c "SELECT version()" ``` ここで
```

- <psql_path> は、 psql 実行可能ファイル、たとえば/usr/edb/as15/bin/psql へのフルパスです。
- <pem_host> は、PEM サーバーのホスト名または IP アドレスです。
- <pem_db_port> は、PEM バックエンドデータベースサーバーポートです。
- <ID> は、テストするエージェントの ID です。/usr/edb/pem/agent/etc/agent.cfg ファイルで定義されています。

注釈: 共通エージェント証明書とキーペアを生成する の指示を使用した場合

PGUSER を共通エージェントのユーザー名に設定する必要があります。接続が成功すると、データベースサーバーの Postgres バージョンを結果ます。成功は、証明書が有効で、Postgres ユーザーが正しく構成されていることを意味します。

Regenerating the server self-signed SSL certificates

PEM バックエンドデータベースサーバー証明書の有効期限が間もなくエキスパイアと、証明書とキーファイルの再生成を計画します。

自己署名 SSL 証明書を置き換えるには

1. 実行中のすべての PEM エージェントを、最初にサーバホストで、次に監視対象ホストで停止します。- Linux の場合

```
# Running as root
systemctl stop pemagent
```

- Windows の場合サービスアプレットを使用して PEM エージェントを停止します。PEM エージェントサービスの名前は Postgres Enterprise Manager Agent です。 [Services] ダイアログボックスで、サービス名を選択し、[**Stop the service**] を選択します。

1. 既存の SSL 証明書とキーをバックアップします。

```
cd /var/lib/edb/as<x>/data
mkdir certs
mv server. *root.*  ca_* certs/
```

1. openssl コマンドを使用して、ca_key.key ファイルを生成します。

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

1. ca_key.key ファイルをバックエンドサーバーのデータディレクトリに移動し、権限を変更します。

```
mv ca_key.key /var/lib/edb/as<x>/data
chmod 600 /var/lib/edb/as<x>/data/ca_key.key
```

1. ca_key.key を使用して ca_certificate.crt ファイルを生成します。

```
openssl req -x509 -nodes -days 3650 -newkey rsa:4096 -keyout ca_key.key -out ca_
↪certificate.crt
```

1. ca_certificate.crt ファイルのアクセス許可を変更します。

```
chmod 600 /var/lib/edb/as<x>/data/ca_certificate.crt
```

1. ca_certificate.crt ファイルを root.crt ファイルとして再利用しファイル。

```
cp /var/lib/edb/as<x>/data/ca_certificate.crt /var/lib/edb/as<x>/data/root.crt
```

1. root.crt ファイルの所有者と権限を変更します。

```
chown enterprisedb /var/lib/edb/as<x>/data/root.crt
chmod 600 /var/lib/edb/as<x>/data/root.crt
```

1. openssl_rsa_generate_crl() ファンクションを使用して、証明書失効リスト root.crl を作成します。

```
psql -U enterprisedb -d pem --no-psqlrc -t -A -c
"SELECT openssl_rsa_generate_crl(/var/lib/edb/as<x>/data/ca_certificate.crt, /var/lib/
↪edb/as<x>/data/ca_key.key)" > /var/lib/edb/as<x>/data/root.crl
```

1. root.crl ファイルの所有権と権限を変更します。

```
chown enterprisedb /var/lib/edb/as<x>/data/root.crl
chmod 600 /var/lib/edb/as<x>/data/root.crl
```

1. openssl コマンドを使用して、server.key ファイルを生成します。

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

1. server.key をバックエンドサーバーのデータディレクトリに移動し、所有権と権限を変更します。

```
mv server.key /var/lib/edb/as<x>/data
chown enterprisedb /var/lib/edb/as<x>/data/server.key
chmod 600 /var/lib/edb/as<x>/data/server.key
```

1. openssl req コマンドを使用して CSR を作成します。

```
openssl req -new -key server.key -out server.csr -subj /C=IN/ST=MH/L=Pune/O=EDB/CN=PEM
```

-subj は、要件に従って提供されます。 CN を PEM サーバーホストのホスト名/ドメイン名として定義します。

1. openssl x509 コマンドを使用して CSR に署名し、サーバー証明書を生成します。server.crt をバックエンドデータベースサーバーのデータディレクトリに移動します。

```
openssl x509 -req -days 365 -in server.csr -CA ca_certificate.crt -CAkey ca_key.key -
↪CAcreateserial -out server.crt
mv server.crt /var/lib/edb/as<x>/data
```

-req は、入力が CSR であることを示します。 -CA および-CAkey オプションは、CSR の署名に使用するルート証明書とプライベートキーを指定します。

1. server.crt ファイルの所有者と権限を変更します。

```
chown enterprisedb /var/lib/edb/as<x>/data/server.crt
chmod 600 /var/lib/edb/as<x>/data/server.crt
```

1. PEM サーバーを再起動します。

shell systemctl restart edb-as-<x> バックエンドデータベースサーバーを再起動すると、PEM サーバーが再起動されます。

1. 各 PEM エージェントの自己署名 SSL 証明書を再生成します。詳細については、*Regenerating the agent self-signed SSL certificates* を参照してください。

Regenerating the agent self-signed SSL certificates

エージェント証明書とキーファイルを再生成する必要があります。

- PEM サーバー証明書が再生成された場合
- PEM エージェント証明書の有効期限が間もなくエキスパイアと

PEM サーバーと対話するエージェントごとに証明書とキーを再生成し、エージェントにコピーする必要があります。

各エージェントには、pem データベースの pem.agent テーブルに保存されている一意の識別子があります。証明書とキーファイルを、エージェントの識別子に対応する証明書またはキーファイルに置き換える必要があります。

前提条件

- PEM サーバーには自己署名証明書があります。
- ca_certificate.crt および ca_key.key は、PEM バックエンドデータベースサーバーのデータディレクトリにあります。
- ca_certificate.crt は root.crt と同じです。
- ca_certificate.crt および ca_key.key は、有効な SSL 証明書とキーです。

PEM エージェント証明書とキーファイルペアを生成するには

1. psql を使用して、エージェントの数とそれに対応する識別子を見つけます。

```
# Running as enterprisedb
psql -p 5444 -U enterprisedb -d pem --no-psqlrc -t -A -c "SELECT id FROM pem.agent␣
↪WHERE active=true"
```

1. 実行中のすべての PEM エージェントを停止します。

```
# Running as root
systemctl stop pemagent
```

Windows では、サービスアプレットを使用して PEM エージェントを停止します。PEM エージェントサービスの名前は Postgres Enterprise Manager Agent です。 [Services] ダイアログボックスで、サービス名を選択し、[**Stop the service**] を選択します。

1. 2. キーファイルを必要とするエージェントを特定した後、各エージェントに対して agent.key を生成します。

edb_notranlate_2 ID はエージェント識別子です。

1. 各エージェントの証明書署名要求 CSR を生成します。

edb_notranlate_3 CN は agent<ID> です。

1. openssl x509 コマンドを使用して CSR に署名し、エージェント証明書を生成します。

edb_notranlate_4 -req は、入力が CSR であることを示します。 -CA および-CAkey オプションは、CSR の署名に使用するルート証明書とプライベートキーを指定します。

次の証明書とキーファイルのペアを生成する前に、各 PEM エージェントホストで手順 2 および 4 で生成された agent.key および agent.crt ファイルを移動します。

1. 新しい agent<ID>.crt および agent<ID>.key ファイルの権限を変更します。

```
chmod 600 agent<ID>.crt agent<ID>.key
```

1. 古いエージェント証明書とキーファイルをバックアップします。

```
mkdir root/.pem/certs
mv root/.pem/agent<ID>.* root/.pem/certs
```

1. 各エージェントの証明書とキーファイルを、新しく生成されたファイルに置き換えます。

```
cp agent<ID>.key agent<ID>.crt root/.pem
```

1. PEM エージェントサービスを開始します。

- Linux の場合

```
# Running as root
systemctl start pemagent
```

- Windows の場合サービスアプレットを使用して PEM エージェントを起動します。PEM エージェントサービスの名前は Postgres Enterprise Manager Agent です。 [Services] ダイアログボックスで、サービス名を選択し、[**Start the service**] を選択します。

4.8 Managing configuration settings

Postgres Enterprise Manager は、起動時に複数の構成ファイルを読み取ります。

- `config.py` —これはメイン構成ファイルです。変更しないでください。次のファイルのいずれかがオーバーライドされる可能性がある構成設定のリファレンスとして使用できます。
- `config_distro.py` —このファイルは `config.py` の後に読み込まれ、パッケージャーが Postgres Enterprise Manager ディストリビューションに必要な設定を変更することを目的としています。これには通常、特定のパスとファイルの場所が含まれます。このファイルはオプショナル。パッケージソフトは、必要に応じて `config.py` と同じディレクトリに作成できます。
- `config_local.py` —このファイルは `config_distro.py` の後に読み込まれ、エンドユーザーがローカルの設定または標準を満たすように調整するデフォルトまたはパッケージ固有の設定を変更することを目的としています。このファイルはオプショナル。必要に応じて、`config.py` と同じディレクトリに作成します。

デフォルトの `config.py` ファイルのコピーは、参照用に PEM オンラインヘルプに含まれています。

4.9 Troubleshooting agent issues

PEM エージェントで発生する可能性のあるこれらの問題をトラブルシューティングできます。

4.9.1 削除された PEM エージェントの復元

エージェントが pem.agent テーブルから削除されている場合、復元することはできません。pemworker ユーティリティを使用してエージェントを再登録する必要があります。

エージェントが PEM Web クライアントから削除されたが、値が active = f の pem.agent テーブルにエントリがまだある場合、エージェントを復元できます。

1. id および active フィールドの値を確認します。

```
pem=# SELECT * FROM pem.agent;
```

2. エージェントのステータスを pem.agent テーブルの true に更新します。

```
pem=# UPDATE pem.agent SET active=true WHERE id=<x>;
```

x は、ステップ 1 で使用したクエリーの出力に表示された識別子です。

3. PEM Web クライアントを更新します。

削除されたエージェントが復元されます。ただし、そのエージェントにバインドされたサーバーはダウンしているように見える場合があります。この問題を解決するには、サーバーの PEM エージェントプロパティを変更して、バインドされたエージェントを再度追加します。変更が成功すると、サーバーは正常に実行されているように表示されます。

4.9.2 構成ファイルパスの更新エージェントのステータスが不明と表示されます

エージェント構成ファイル agent.cfg をデフォルトの場所/usr/edb/pem/agent/etc から別の場所に移動すると、PEM ダッシュボードにエージェントステータスが「不明」と表示される場合があります。その場合、pemagent サービスファイルのエージェント構成ファイルパスの値を更新する必要があります。

1. 次のコマンドを使用して、スーパーユーザーとして pemagent サービスファイルを変更します。

```
# Running as superuser
sudo vi  /usr/lib/systemd/system/pemagent.service
```

2. agent.cfg ファイルパスを更新します。たとえば、Redhat ホストでは、ファイルパスを更新します。

```
ExecStart=/usr/edb/pem/agent/bin/pemagent -c /usr/edb/pem/agent/etc/agent.cfg
```

3. systemd をリロードして、変更したサービススクリプトを更新します。

```
sudo systemctl daemon-reload
```

4. PEM エージェントを再起動します。

```
sudo systemctl restart pemagent
```

4.9.3 コマンドラインを使用してダウンまたは不明なステータスの PEM エージェントを削除する

PEM Web インターフェイスを使用して、ダウンまたは不明なステータスの PEM エージェントを削除することは、このようなエージェントの数が多い場合、困難になる場合があります。この状況では、コマンドラインインターフェイスを使用して、ダウンまたは不明なエージェントを削除できます。

次のクエリーを使用して、 *N* 時間以上ダウンしているエージェントを削除します。

```
UPDATE pem.agent SET active=false WHERE id IN
(SELECT a.id FROM pem.agent
a JOIN pem.agent_heartbeat b ON (b.agent_id=a.id)
WHERE a.id IN
(SELECT agent_id FROM pem.agent_heartbeat WHERE (EXTRACT (HOUR FROM now())-
EXTRACT (HOUR FROM last_heartbeat)) > <N> ));
```

次のクエリーを使用して、不明なステータスのエージェントを削除します。

```
UPDATE pem.agent SET active=false WHERE id IN
(SELECT id FROM pem.agent WHERE id NOT IN
(SELECT agent_id FROM pem.agent_heartbeat));
```

第5章　Using

5.1 Accessing the web interface

PEM サーバーとエージェントをインストールした後、PostgreSQL または EDB Postgres Advanced Server インスタンスの監視と管理を開始するように PEM を構成できます。PEM サーバーインストーラーは、PEM Web インターフェイスをインストールします。インターフェイスを使用して、監視対象サーバーにあるオブジェクトに関する情報を確認したり、PEM サーバーによって収集された統計情報を確認したりできます。

サーバーのインストールが完了したら、次の場所に移動して、選択したブラウザーで PEM インターフェイスを開くことができます。

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

`ip_address_of_PEM_host` は、PEM サーバーのホストの IP アドレスを指定します。

Postgres Enterprise Manager Login ウィンドウのフィールドを使用して、PEM サーバーで自分自身を認証します。

- **Username** フィールドに pem データベースユーザーの名前を入力します。最初に接続するユーザー、これは PEM サーバーのインストール時に指定された名前です。
- ユーザーに関連付けられたパスワードを **Password** フィールドに入力します。

資格情報を入力したら、 **Login** を選択して PEM に接続します。

PEM Web インタフェースが開き、 Global Overview ダッシュボードが表示されます。

[ブラウザ] ペインには、各サーバーにあるデータベースオブジェクトに関する情報へのアクセスを提供するツリーが表示されます。ツリーが展開され、PEM サーバーが監視するサーバーとオブジェクトの階層ビューが表示されます。

PEM Web インターフェイスを使用してデータベースサーバーを管理またはモニタする前に、そのサーバーを PEM サーバーに登録する必要があります。サーバーを登録するときは、サーバーへの接続を説明し、接続の認証情報を提供し、管理設定オプションでエージェントをバインドします。

サーバーは管理対象または非管理対象にすることができます。

- **管理対象** サーバーは PEM エージェントにバインドされます。PEM エージェントは、バインドされたサーバーを監視し、タスクを実行するか、PEM ダッシュボードに表示する統計を報告します。管理対象サーバーは、カスタムアラートなどの拡張 PEM 機能にアクセスできます。サーバーを登録するときに、必要に応じて PEM が管理対象サーバーを再起動できるようにすることもできます。
- **非管理対象**サーバーは PEM エージェントにバインドされません。非管理対象サーバーでデータベースオブジェクトを作成できますが、拡張 PEM 機能カスタムアラートなどは、非管理対象サーバーではサポートされていません。

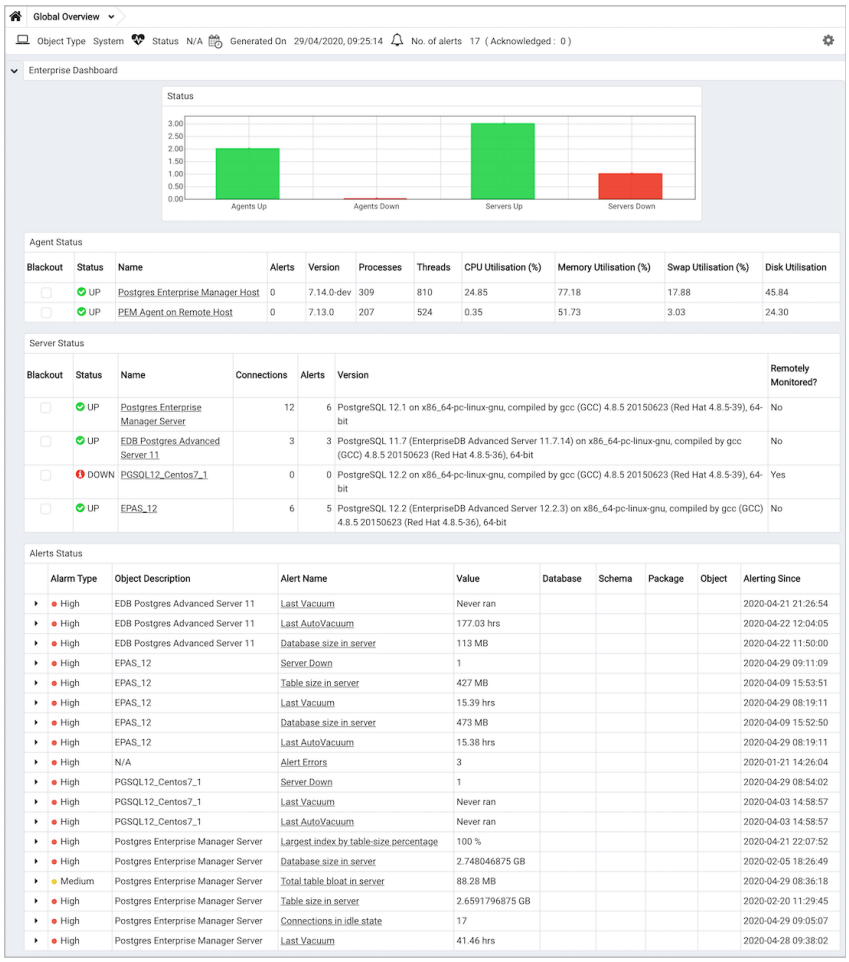


図 1: The Global Overview Dashboard

また、登録するサーバーの pg_hba.conf ファイルで、PEM Web インターフェイスのホストからの接続が許可されていることを確認する必要があります。

5.1.1 UI の詳細についてのアプリ内ヘルプへのアクセス

アプリ内ヘルプは、メニュー項目、ダッシュボードのチャートとグラフ、ダイアログボックスのオプションを含む PEM Web インターフェイスに関する包括的な詳細を提供します。アプリ内のヘルプには、データベースオブジェクトを作成する手順も含まれています。

- ヘルプのランディングページにアクセスするには、 **Help > Online help** を選択します。
- ダイアログボックスの状況依存ヘルプにアクセスするには、 **?** を選択します。

5.1.2 PEM クライアントオブジェクトブラウザ

Browser ツリーコントロールは、各サーバーに存在するデータベースオブジェクトの情報と管理オプションへのアクセスを提供します。ツリーコントロールが展開され、PEM サーバーによって監視されているサーバーとオブジェクトの階層ビューが表示されます。ロールが必要な特権を保持している場合、ツリーコントロールのノードを右クリックしてアクセスするコンテキストメニューオプションを使用して、新しいオブジェクトを作成し、既存のオブジェクトを変更および削除できます。

ツリーコントロールのノードを展開して、選択したサーバーにあるデータベースオブジェクトの階層ビューを表示します。

- ノードの左側のプラス記号+を使用して、ツリーコントロールのセグメントを展開します。
- ノードの左側にマイナス記号-を使用して、そのノードを閉じます。

コンテキストメニューオプションには、次の 1 つ以上の選択を含めることができます。

Option	Action
Add named restore point	復元ポイントの名前を作成して入力します。
Backup	*[ツール] メニュー* ダイアログボックスを開き、データベースオブジェクトをバックアップします。
Backup Globals	*[ツール] メニュー* ダイアログボックスを開き、クラスターオブジェクトをバックアップします。
Backup Server	*[ツール] メニュー* ダイアログボックスを開き、サーバーをバックアップします。
Connect Server	選択したサーバーとの接続を確立します。
Create	状況依存の選択を提供するコンテキストメニューにアクセスします。選択すると、新しいオブジェクトを作成するための Create ダイアログボックスが開きます。
CREATE Script	*Query information* を開き、 CREATE スクリプトを編集または表示します。
Dashboards	PEM ダッシュボードにすばやくアクセスする場合に選択します。
Delete/Drop	現在選択されているオブジェクトをサーバーから削除します。

次のページに続く

表 1 – 前のページからの続き

Option	Action
Disconnect Database	データベース接続を終了します。
Disconnect Server	現在選択されているオブジェクトを更新します。
Drop Cascade	現在選択されているオブジェクトとすべての依存オブジェクトをサーバーから削除します。
Debugging	Debugger ツールにアクセスします。
Grant Wizard	*[ツール] メニュー* ツールにアクセスします。
Maintenance	**管理メニュー** ダイアログボックスを開き、VACUUM、ANALYZE、REINDEX、または CLUSTER します。
Management	ノードに関連するアクセス管理タスク。
Properties	現在選択されているオブジェクトのプロパティを確認または変更します。
Refresh	現在選択されているオブジェクトを更新します。
Reload Configuration	サーバーを再起動せずに構成ファイルを更新します。
Restore	*[ツール] メニュー* ダイアログボックスにアクセスして、バックアップからデータベースファイルを復元します。
View Data	**View Data** オプションを使用して、クエリツールの **データ出力** タブで選択したテーブルに保存されているデータにアクセスします。

テーブルおよびネストされたテーブルノードに関連付けられている状況依存メニューは、追加の表示オプションを提供します。

Option	Action
Import/Export	Import/Export ダイアログボックスを開き、選択したテーブルにデータをインポートまたはエクスポートします。
Reset Statistics	選択したテーブルの統計をリセットします。
Scripts	*Query information* を開き、フライアウトメニューから選択したスクリプトを編集または表示します。
Truncate	テーブルからすべての行を削除します。
Truncate Cascade	テーブルとその子テーブルからすべての行を削除します。
View First 100 Rows	選択したテーブルの最初の 100 行を表示するデータグリッドにアクセスします。
View Last 100 Rows	選択したテーブルの最後の 100 行を表示するデータグリッドにアクセスします。
View All Rows	選択したテーブルのすべての行を表示するデータグリッドにアクセスします。
View Filtered Rows	**Data Filter** ポップアップにアクセスして、データセットにフィルターを適用します。

5.1.3 PEM タブ付きブラウザウィンドウ

PEM クライアントのメインパネルには、ツリーコントロールで現在選択されているオブジェクトに関する情報を表示するタブのコレクションが含まれています。

Dashboard タブは状況依存です。サーバーグループまたは PEM エージェントノードから**ダッシュボード**タブに移動すると、 EDB Postgres の Welcome ウィンドウが開きます。ここで次のことができます。

- **Add New Server** アイコンを選択して **データベースサーバーの手動登録** を開き、サーバーへの接続を定義します。
- **Configure PEM** アイコンを選択して *PEM* **サーバー構成の編集** を開き、サーバーパラメーターを変更します。
- **Getting Started** アイコンを選択して新しいタブを開き、EnterpriseDB Web サイトの PEM 入門セクションを表示します。
- **EDB Website** アイコンを選択して、EnterpriseDB Web サイトのホームページに移動します。 EnterpriseDB Web サイトには、今後のイベントやその他のプロジェクトに関するニュースが掲載されています。
- **PostgreSQL Website** アイコンを選択して、PostgreSQL プロジェクトの Web サイトに移動します。PostgreSQL サイトには、最近のリリースおよびその他のプロジェクト情報に関するニュースが掲載されています。
- **EDB Blogs** アイコンを選択して EDB Blog ページに移動します。ここで、Postgres 関連ブログへの最新の従業員の投稿を確認できます。

エージェントまたはサーバーの名前を選択し、**ダッシュボード**タブに移動して、現在選択しているオブジェクトのセッションまたはサーバーアクティビティを確認します。

エージェントまたはサーバーの名前から開くと、 **ダッシュボード**タブは使用統計のグラフィカル分析を提供します。

- サーバーセッションまたはデータベースセッショングラフには、サーバーまたはデータベースとの相互作用が表示されます。
- 1 秒ごとのトランザクショングラフには、サーバーまたはデータベースで発生している 1 秒ごとのコミット、ロールバック、および合計トランザクションが表示されます。
- Tuples In グラフには、サーバーまたはデータベースで挿入、更新、および削除されたタプルの数が表示されます。
- Tuples Out グラフには、サーバーまたはデータベースから取得および返されたタプルの数が表示されます。
- Block I/O グラフには、サーバーまたはデータベースのファイルシステムから読み取られたブロック、またはバッファキャッシュただし、オペレーティングシステムのファイルシステムキャッシュから取得されたブロック数が表示されます。
- Server アクティビティのタブ付きパネルには、セッション情報、セッションロック、準備されたトランザクション、構成を含むテーブルが表示されます。

[**プロパティ**] タブに移動して、ツリーコントロールで現在選択されている項目のプロパティを確認します。

[**SQL**] タブには、ブラウザツリーコントロールで現在選択されているオブジェクトを生成するために使用される SQL コードが表示されます。

[**Statistics**] タブには、ツリーコントロールのオブジェクトごとに収集された統計が表示されます。テーブルに表示される統計は、強調表示されたオブジェクトの種類によって異なります。列見出しを選択して、列に表示されているデータでテーブルを並べ替えます。並べ替え順序を逆にするには、それを再度選択します。次の表は、表示される統計の一部をリストしています。

Panel	Description
PID	行に関連付けられたプロセス ID。
User	オブジェクトを所有するユーザーの名前。
Database	データベース名。
Backends	データベースへの現在の接続数。
Backend start	バックエンドプロセスの開始時刻。
Xact Committed	過去 1 週間にデータベースにコミットされたトランザクションの数。
Xact Rolled Back	過去 1 週間にロールバックされたトランザクションの数。
Blocks Read	過去 1 週間にメモリから読み取られたブロック数 MB 単位。
Blocks Hit	先週にキャッシュでヒットしたブロック数 MB 単位。
Tuples Returned	先週に返されたタプルの数。
Tuples Fetched	先週に取得されたタプルの数。
Tuples Inserted	過去 1 週間にデータベースに挿入されたタプルの数。
Tuples Updated	過去 1 週間にデータベースで更新されたタプルの数。
Tuples Deleted	過去 1 週間にデータベースから削除されたタプルの数。
Last statistics reset	データベースの最後に統計がリセットされた時間。
Tablespace conflicts	データベース内の削除されたテーブルスペースとのリカバリの競合のためにキャンセルされたクエリの数。
Lock conflicts	データベースのロックとのリカバリの競合のためにキャンセルされたクエリの数。
Snapshot conflicts	データベース内の古いスナップショットとのリカバリの競合のためにキャンセルされたクエリの数。
Bufferpin conflicts	データベース内の固定バッファとのリカバリの競合のためにキャンセルされたクエリの数。
Temporary files	一時ファイルの合計数。統計コレクターが使用するファイルを含む。
Size of temporary files	一時ファイルのサイズ。
Deadlocks	データベースのデッドロックとのリカバリの競合のためにキャンセルされたクエリの数。
Block read time	読み取られたブロックの読み取りに必要なミリ秒数。
Block write time	読み取られたブロックの書き込みに必要なミリ秒数。
Size	選択したデータベースのサイズ MB 単位。

[**Dependencies**] タブには、現在選択されているオブジェクトが依存するオブジェクトが表示されます。データベース構造の整合性を保証するために、サーバーは、他のオブジェクトが依存するオブジェクトを誤ってドロップしないように確認します。`DROP CASCADE` を使用して、別のオブジェクトが依存するオブジェクトを削除する必要があります。

Dependencies テーブルには以下が表示されます。

- **Type** フィールド。親オブジェクトタイプを指定します。
- **Name** フィールド。親オブジェクトの識別名を指定します。
- **Restriction** フィールド、現在選択されているオブジェクトと親間の依存関係を説明します。

[**Dependents**] タブには、ブラウザツリーで現在選択されているオブジェクトに依存するオブジェクトのテーブルが表示されます。依存オブジェクトは、ブラウザツリーコントロールで現在選択されているオブジェクトに影響を与えることなくドロップできます。

- **Type** フィールドは、依存オブジェクトタイプを指定します。
- **Name** フィールドは、依存オブジェクトの識別名を指定します。
- **Restriction** フィールドは、現在選択されているオブジェクトと親間の依存関係を説明します。

Monitoring タブに移動して、*Dashboards* で表示される情報にアクセスします。ダッシュボードには、PEM サーバーによって監視されているオブジェクトに関する統計情報が表示されます。

[Management] または [Tools] ダイアログボックスから PEM 機能にアクセスすると、PEM は追加のタブを開きます。現在のタブを右クリックし、作業スタイルに合わせて表示をカスタマイズできるコンテキストメニューから選択します。

- 現在選択しているパネルを削除するには、 **Remove Panel** を選択します。
- **Rename Panel** を選択して、現在選択しているパネルの名前を変更します。
- **Detach Panel** を選択して、現在選択しているパネルをデタッチし、使いやすいように再配置します。
- **Add Panel** を選択し、パネルに追加する使用可能なオプションを選択します。

PEM クライアントは、プログラムを終了するときに調整を保存します。PEM クライアントを元の形式にリセットするには、 **File > Reset Layout** を選択します。

5.1.4 チャート、グラフ、テーブルコントロールを使用する

PEM クライアントダッシュボードの各グラフィックの右上隅にあるアイコンを使用して、PEM クライアントに表示されるチャート、グラフ、テーブルを制御、ダウンロード、およびカスタマイズします。

Refresh アイコン  を使用して、PEM プローブから利用可能な最新のコンテンツを表示します。

Download アイコン  を選択して、チャートまたはグラフの.jpeg または.png 画像をダウンロードします。デフォルトでは、ファイルは.jpeg 形式です。ファイルを.png として保存するには、**Personalize** アイコンを使用してダウンロード形式を変更します。

フルスクリーンアイコン  を選択して、チャートまたはグラフを展開して PEM クライアントのメインペイン全体に表示されます。

Personalize  アイコンを選択して、セッションのみにチャートまたはグラフの表示プロパティを変更します。

Information  アイコンを使用して、チャートまたはグラフに関する情報にアクセスします。

グラフィックのパーソナライズ

Personalize アイコンを選択すると、[Personalize chart configuration] ダイアログボックスが開きます。

[Personalize chart configuration] ダイアログボックスのコントロールを使用して、グラフィックのプロパティを変更します。

- **Auto Refresh** コントロールを使用して、更新間の秒数を増減します。
- **Auto Refresh** フィールドを使用して、テーブルまたはチャートに表示されるデータの更新の間の秒数を指定します。
- 必要に応じて、 **Download as** フィールドを使用して、チャートを.jpeg 画像または.png 画像としてダウンロードするかどうかを示します。
- 必要に応じて、 **Colors** セレクターを使用して、チャートで使用する表示色を指定します。
- 必要に応じて、 **Show Acknowledged Alerts** スイッチを **Yes** に設定して、 **Ack'ed** 列のチェックボックスで確認したアラートをテーブルに表示することを示しますフィールドを **No** に設定して、テーブルが確認されたアラートを非表示にすることを示します。確認されたアラートは、アラート定義で指定された時間が経過するまで、テーブルコンテンツから削除されません。

表示プロパティをパーソナライズした後、右上隅のコントロールを使用して変更を適用します。

- **Delete** アイコンを使用して、グラフィックのプロパティをデフォルト設定にリセットします。メニューを使用して、グラフィックのこのインスタンスにのみに変更を適用するか、他のダッシュボードに表示されるときに同じグラフィックに変更を適用するかを指定します。
- **Save** アイコンを使用して、グラフィックのプロパティへの変更を保存します。メニューを使用して、グラフィックのこのインスタンスのみに変更を適用するか、他のダッシュボードに表示されるときは同じグラフィックに変更を適用することを指定します。

5.1.5 ブラウザツールバー

ブラウザツールバーには、データの表示やクエリツールなど、頻繁に使用される機能のショートカットが提供されます。このツールバーは [ブラウザ] パネルに表示されます。ボタンは、選択したブラウザノードに基づいて有効/無効になります。

- **Query tool** ボタンを使用して、現在のデータベースコンテキストでクエリツールを開きます。
- **View Data** ボタンを使用して、選択したテーブルに保存されているデータを表示/編集します。
- **Filtered Rows** ボタンを使用して Data Filter ポップアップにアクセスし、表示/編集のためにデータセットにフィルターを適用します。

5.1.6 PEM メニューバー

PEM メニューバーは、データベースサーバーとこれらのサーバーに存在するオブジェクトの管理に使用できるコマンドと機能へのアクセスを提供します。オプションが無効になっている場合

- 現在接続しているデータベースサーバは、選択した機能をサポートしていない可能性があります。
- 選択されたメニューオプションは、現在のオブジェクトに適用されない場合があります。
- サーバーへの接続に使用したロールには、選択したオブジェクトを変更するための十分な権限がない可能性があります。

PEM Web インタフェースの上部にあるコンテキスト依存メニューにより、環境をカスタマイズ、PEM のエンタープライズ管理機能へのアクセスを提供できます。

ファイルメニュー

File メニューを使用して、次のオプションにアクセスします。

Option	Action
Preferences	[Preferences] ダイアログボックスを開き、PEM クライアント設定をカスタマイズします。
Lock Layout	サブメニューを開き、UI レイアウトをロックするレベルを選択します。
Server Configuration	[Server Configuration] ダイアログボックスを開き、PEM サーバーの構成設定を更新します。
Reset Layout	ワークスペースパネルがポップアウトされた場合は、 **レイアウトのリセット** を使用してデフォルトにリセットできます。

オブジェクトメニュー

Object メニューは状況依存です。 **Object** メニューを使用して、次のオプションにアクセスします。

Option	Action
Create	状況依存の選択を提供するメニューにアクセスします。
Refresh	現在選択されているオブジェクトを更新します。
Delete/Drop	現在選択されているオブジェクトをサーバーから削除します。
Connect Server	[Connect to Server] ダイアログボックスを開き、サーバーとの接続を確立します。
CREATE Script	クエリツールを開き、選択したスクリプトを編集または表示します。
Disconnect Server	現在選択されているオブジェクトを更新します。
BART	BART 構成の削除、BART バックアップの取得、または BART 構成の再検証のオプションを提供するメニューにアクセスします。

次のページに続く

表 5 – 前のページからの続き

Option	Action
Clear Saved Password	データベースサーバーパスワードを保存した場合、保存したパスワードをクリアします。パスワードが保存された後にのみ有効になります。
Clear SSH Tunnel Password	ssh トンネルパスワードを保存した場合、保存したパスワードをクリアします。パスワードが保存された後にのみ有効になります。
Drop Cascade	現在選択されているオブジェクトとすべての依存オブジェクトをサーバーから削除します。
Hide	現在選択されているグループを非表示にします。非表示のグループを表示するには、設定で **Show hidden groups** オプションを有効にします。
Properties	現在選択されているオブジェクトのプロパティを確認または変更します。
Trigger(s)	現在選択されているテーブルのトリガーを無効または有効にします。
Truncate	テーブルからすべての行を削除する **Truncate** か、テーブルとその子テーブルからすべての行を削除します **Truncate Cascade** 。
View Data	データを表示するための複数のオプションを提供するメニューにアクセスします。
Remove Server	選択したサーバーをブラウザツリーから削除します。
Delete/Drop	現在選択されているオブジェクトをサーバーから削除します。
Connect Database	選択したデータベースに接続します。
Count Rows	選択したテーブルの行数をカウントします。
Reset Statistics	選択したテーブルの統計をリセットします。
Scripts	選択したテーブルの CREATE、DELETE、INSERT、SELECT、および UPDATE スクリプト。

管理メニュー

Management メニューを使用して、次の PEM 機能にアクセスします。

Option	Action
Audit Manager	Audit Manager を開き、監視対象サーバーで監査を構成します。
Auto Discovery	[Auto Discovery] ダイアログボックスを開き、監視対象データベースサーバーを見つけてバインドするように PEM エージェントを構成します。
Capacity Manager	Capacity Manager ダイアログボックスを開き、履歴またはプロジェクトの将来のリソース使用量を分析します。
Log Manager	Log Manager ダイアログボックスを開き、サーバーのログ収集を構成します。
Manage Alerts	**Manage Alerts** タブを開き、アラートの動作を作成または変更します。
Manage Charts	**Manage Charts** タブを開き、PEM チャートを作成または変更します。
Manage Dashboards	VACUUM、ANALYZE、REINDEX、または CLUSTER への Manage Dashboards ダイアログボックスを開きます。
Manage Probes	VACUUM、 ANALYZE、REINDEX、または CLUSTER への Manage Probes ダイアログボックスを開きます。
Postgres Expert	Postgres Expert ウィザードを開き、サーバーとデータベースの静的分析を実行します。

次のページに続く

表 6 – 前のページからの続き

Option	Action
Postgres Log Analysis Expert	Postgres Log Analysis Expert ダイアログボックスを開き、ログファイルの内容を分析して使用傾向を調べます。
Scheduled Tasks	[**Scheduled Tasks**] タブを開き、保留中または最近完了したタスクを確認します。
Tuning Wizard	[Tuning Wizard] ダイアログボックスを開き、サーバーの調整に関する推奨事項のセットを生成します。
Reports	[Reports] ダイアログボックスを開き、サーバーのアラート履歴レポート、システム構成レポート、またはコア使用状況レポートを生成します。
Schedule Alert Blackout	Schedule Alert Blackout ダイアログボックスを開き、サーバーとエージェントのアラートブラックアウトをスケジュールします。

ダッシュボードメニュー

状況依存の **Dashboards** メニューを使用して、ダッシュボードにアクセスします。

Option	Action
Alerts	選択したノードの Alerts ダッシュボードを開きます。
Audit Log	選択したノードの Audit Log Analysis ダッシュボードを開きます。
PGD Admin	選択したノードの PGD Admin ダッシュボードを開きます。
PGD Group Monitoring	選択したノードの PGD Group Monitoring ダッシュボードを開きます。
PGD Node Monitoring	選択したノードの PGD Node Monitoring ダッシュボードを開きます。
Database Server	選択したノードのデータベース分析ダッシュボードを開きます。
I/O Analysis	選択したノードの I/O Analysis ダッシュボードを開きます。
Memory	選択したノードの Memory Analysis ダッシュボードを開きます。
Object Activity	選択したノードの Object Activity Analysis ダッシュボードを開きます。
Operating System	選択したノードの Operating System Analysis ダッシュボードを開きます。
Probe Log	選択したノードの Probe Log Analysis ダッシュボードを開きます。
Server Log	選択したノードの Server Log Analysis ダッシュボードを開きます。
Session Activity	選択したノードの Session activity Analysis ダッシュボードを開きます。
Storage	選択したノードの Storage Analysis ダッシュボードを開きます。
Streaming Replication	選択したノードの Streaming Replication Analysis ダッシュボードを開きます。
System Wait	選択したノードの System Wait Analysis ダッシュボードを開きます。
Session Waits	選択したノードの Session Waits Analysis ダスボードを開きます。
Custom Dashboards	ユーザーが構成したカスタムダッシュボードをリストする Custom Dashboards を開きます。

[ツール] メニュー

Tools メニューのオプションを使用して、次の機能にアクセスします。

Option	Action
Schema Diff	[Schema Diff] ダイアログボックスを開き、2 つのデータベーススキーマ間のスキーマオブジェクトを比較します。
Search objects	[Search Objects] ダイアログボックスを開き、データベース内のデータベースオブジェクトを検索します。
Server	名前付け復元ポイントの追加、パフォーマンス診断、キューサーバーのスタートアップ、キューサーバーのシャットダウン、クラスタープライマリの置換、EFM クラスターのスイッチオーバー、 SQL プロファイラーなどのさまざまなサーバー関連ツールにアクセスします。
Query tool	現在選択しているオブジェクトのクエリツールを開きます。
Reload Configuration	サーバーを再起動せずに構成ファイルを更新します。
Pause replay of WAL	WAL ログの再生を一時停止します。
Resume replay of WAL	WAL ログの再生を再開します。
Import/Export	[Import/Export Data] ダイアログボックスを開き、テーブルからデータをインポートまたはエクスポートします。
Maintenance	VACUUM、 ANALYZE、REINDEX、または CLUSTER への Maintenance ダイアログボックスを開きます。
Backup	[Backup] ダイアログボックスを開き、データベースオブジェクトをバックアップします。
Backup Globals	クラスターオブジェクトをバックアップするために Backup Globals ダイアログボックスを開きます。
Backup Server	サーバーをバックアップするには、 Backup Server ダイアログボックスを開きます。
Restore	[Restore] ダイアログボックスを開き、バックアップからデータベースファイルを復元します。
Grant Wizard	Grant Wizard ツールを開きます。
Schedule Backup	BART バックアップの [Schedule Backup] ダイアログボックスを開きます。
New ERD Project(Beta)	ERD ツールを開き、データベースの設計を開始します。
Storage Manager	ストレージマネージャーを開き、バックアップファイルをアップロード、削除、またはダウンロードします。

ヘルプメニュー

Help メニューのオプションを使用して、オンラインヘルプドキュメントにアクセスするか、PEM インストールに関する情報を確認します。

Option	Action
Quick Search	Quick Search フィールドにキーワードを入力します。少なくとも 3 文字を入力すると、メニュー項目の下に一致するすべての可能性が表示され、ヘルプ記事の下に関連ドキュメントが表示されます。メニュー項目の下のオプションを選択して、特定の機能またはオブジェクトのアクションを実行します。ヘルプ記事のいずれかを選択すると、そのトピックのヘルプが開き、別のウィンドウで強調表示されたテキストが表示されます。 **注** メニュー項目の下のオプションが無効になっている場合、情報アイコンを介して情報を提供します。
Online Help	Postgres Enterprise Manager のアプリ内ドキュメントを開きます。
REST API Reference	REST API リファレンスを開きます。
EDB Website	ブラウザウィンドウで EDB ウェブサイトを開きます。
About Postgres Enterprise Manager	Postgres Enterprise Manager のバージョンとユーザー情報を見つけます。

5.1.7 チャート、グラフ、テーブルの制御とカスタマイズ

ダッシュボードの各グラフィックの右上隅にあるアイコンを使用して、現在のユーザーセッションの PEM クライアントに表示されるチャート、グラフ、テーブルを制御およびカスタマイズします。

次の表に、アイコンに関する情報を示します。

Icons	Information
icon6 	View information about the chart, graph, or table.
icon1 	Refresh the content of a chart, graph, or table.
icon3 	Personalize the chart, graph, or table settings for the current user.
icon4 	Download an image of the chart or graph.
icon5 	View the legends that are used in the chart, graph, or table.
icon2 	Expand the chart or graph to full-screen.

PEM ダッシュボードに表示されるグラフィックのカスタマイズの詳細については、PEM クライアントのオンラインヘルプを参照してください。

5.2 Connecting and disconnecting a database server

サーバー接続を定義した後、Connect to Server ダイアログを使用してサーバーで認証し、サーバーに保存されているオブジェクトにアクセスします。ダイアログにアクセスするには、PEM クライアントツリーコントロールでサーバー名を右クリックし、コンテキストメニューから Connect Server を選択します。

プロンプトが表示されたら、選択したサーバーの認証情報を入力します。

- Password フィールドを使用して、定義されたサーバーに関連付けられたユーザーのパスワードを提供します。
- Save Password の横にあるボックスを選択して、今後の接続のためにパスワードを保存するようにサーバーに指示します。パスワードを保存すると、このサーバー定義でデータベースサーバーに再接続するときにプロンプトは表示されません。

サーバーが正常に接続すると、ブラウザーは右下隅の緑色のステータスバーにメッセージを表示します。

接続しようとしたときにエラーメッセージを受け取った場合は、ネットワークが PEM とデータベースサーバーのホストの通信を許可していることを確認します。特定のエラーメッセージの詳細については、ヘルプページを参照してください。

接続の詳細を確認または変更するには、サーバーの名前を右クリックし、コンテキストメニューから Properties... を選択します。

5.2.1 サーバーからの切断

サーバーから切断するには、Browser ツリーコントロールでサーバー名前を右クリックし、コンテキストメニューから Disconnect Server を選択します。選択したサーバーを切断することを確認するポップアップが表示されます。

5.3 Monitoring performance

PEM を使用すると、すべての管理対象サーバーの企業全体のパフォーマンス監視を実装できます。

トップレベルのダッシュボードは Global Overview です。 Global Overview ダッシュボードには、PEM サーバーによって監視されているすべてのサーバーとエージェントのステータスの概要、監視対象サーバーのリスト、および現在トリガーされているアラートの状態が表示されます。

PEM は、他の PostgreSQL 管理ツールにはない多くの利点を提供します。

- 一括デザインの管理 —PEM は、エンタープライズデータベース管理用に設計され、地理的境界を越えた多数のサーバーの管理に取り組むように構築されています。グローバルダッシュボードは、すべてのサーバーの稼働/ダウン/パフォーマンスステータスを視覚的に最新の状態に保ちます。
- 分散アーキテクチャ —PEM アーキテクチャは、統計情報を収集し、オペレーティングシステムプラットフォームに関係なくマシンでリモートで操作を実行する機能を最大化します。
- グラフィカル管理—データベース管理のすべての側面は、グラフィカルインターフェイスを使用して PEM クライアントで実行できます。サーバーの起動とシャットダウン、構成管理、ストレージとセキュリティ制御、オブジェクト作成、パフォーマンス管理などを単一のコンソールから処理できます。

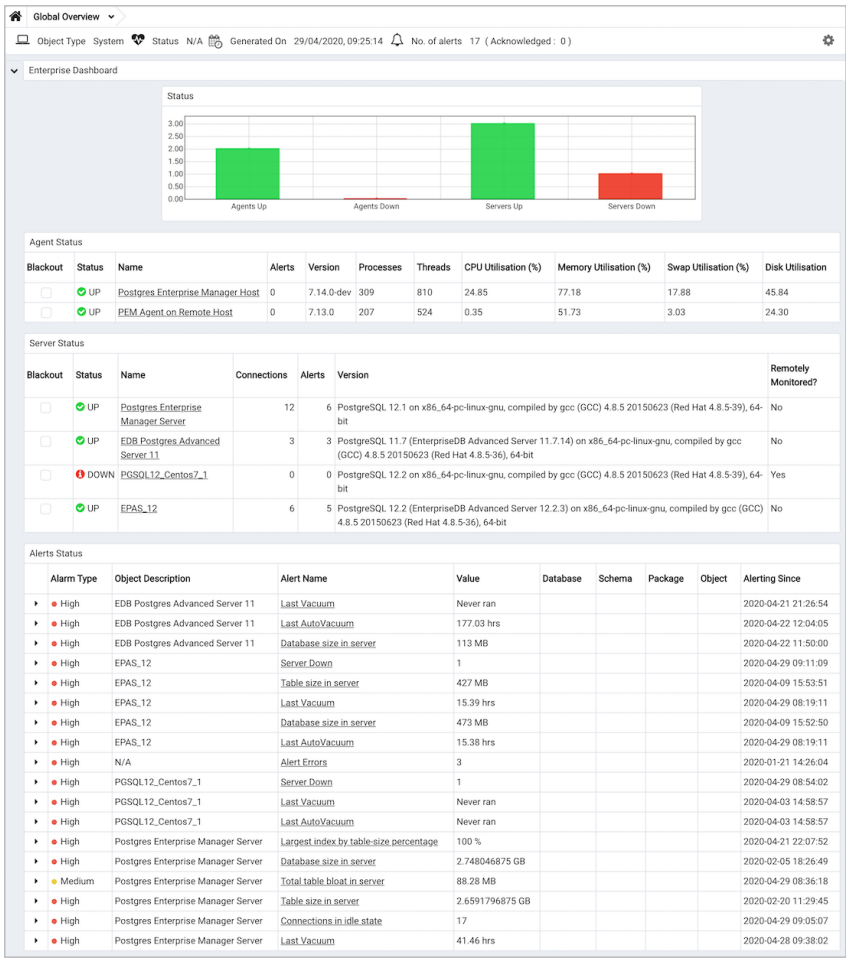


図 2: The Global Overview dashboard

- 完全な SQL IDE —PEM には、アドホック SQL クエリー、ストアドプロシージャ/ファンクション開発、およびグラフィカルデバッガーを提供する堅牢な SQL 統合開発環境 IDE が含まれています。
- エンタープライズパフォーマンスモニタリング —PEM は、すべての管理対象データベースサーバーにエンタープライズクラスのパフォーマンス監視を提供します。軽量で効率的なエージェントは、各データベースサーバーの操作と各マシンの基になるオペレーティングシステムのあらゆる側面を監視します。これらは、インターフェイスでパフォーマンスページを簡単に移動するための詳細な統計を提供します。
- プロアクティブなアラート管理 —PEM では、24 時間監視されるメモリやストレージなどのキーメトリックごとにパフォーマンスしきい値を作成できます。しきい値違反は、一元化されたダッシュボードにアラートが送信され、問題の性質と、サーバーの全体的なパフォーマンスを危険にさらす状況を防ぐために必要なアクションを通知します。
- 簡素化されたキャパシティプランニング—すべての主要なパフォーマンス関連の統計が収集され、指定された期間 PEM リポジトリに保持されます。Capacity Manager ユーティリティを使用すると、さまざまな統計を選択し、傾向分析を実行して、ピーク負荷期間やストレージ消費傾向などを理解できます。ツールの予測メカニズムを使用すると、リソースの使用量を予測できるため、それに応じて計画と予算を立てることができます。
- 監査マネージャー — Audit Manager は、 EDB Postgres Advanced Server インスタンスに監査ログを構成します。データベースへの接続、データベースからの切断、データベースに対して実行された SQL ステートメントなどのアクティビティをログに記録できます。Audit Log ダッシュボードを使用して、ログをフィルタリングおよび表示できます。
- ログマネージャー—ログマネージャーウィザードは、中央のテーブルへのオプショナルのログ収集を使用してサーバーロギングパラメータを構成します。ウィザードを使用して、ログファイルのローテーション、ログの宛先、エラーメッセージの重大度などのロギング動作の設定を指定します。Server Log ダッシュボードを使用して、収集されたサーバーログエントリをフィルタリングおよび確認します。
- SQL ワークロードプロファイリング —PEM には、1 つ以上のサーバーに対して実行される SQL ステートメントをトレースできる SQL プロファイリングユーティリティが含まれています。SQL プロファイリングは、アドホックまたはスケジュールされた方法で実行できます。キャプチャした SQL ステートメントをフィルタリングできるため、実行不良の SQL ステートメントを簡単に特定および調整できます。EDB Postgres Advanced Server のインデックスアドバイザーに SQL ステートメントをフィードすることもでき、各ステートメントを分析し、パフォーマンスを向上させるために作成する新しいインデックスに関する推奨事項を作成します。
- 専門家によるデータベース分析 — PEM には、Postgres Expert ユーティリティが含まれています。Postgres Expert は、ベストプラクティスの適用を目的として、選択されたデータベースを分析します。一般的な構成やセキュリティ設定などの領域が検査されます。推奨されるベストプラクティスからの逸脱は、各問題の説明と、物事を正しくするために何をすべきかについての専門家のヘルプとともに報告されます。
- ストリーミングレプリケーションモニタ—ストリーミングレプリケーションダッシュボードをモニタするか、PEM クライアントのオプションを使用して、レプリカノードをプライマリノードにプロモートせることができます。
- セキュアクライアント接続 - mdash; PEM は、暗号化された SSH トンネルを介したセキュアクライアント接続をサポートします。フル機能の PEM クライアントには、安全な接続のための接続情報を提

供できる SSH トンネル定義ダイアログボックスが含まれています。

- 幅広いプラットフォームのサポート —PEM は、ほとんどのメジャー Linux および Windows プラットフォームをサポートしています。

5.3.1 Probes

プローブは、PEM エージェントによって監視されているデータベースオブジェクトに関する情報を取得するスケジュールされたタスクです。PEM は、収集された情報を使用して、各ホームページに表示されるグラフを構築します。** Management > Manage Probes** タブでは、データ収集スケジュールと、特定のプローブから返された情報を PEM が保持する期間を変更できます。

システムプローブ

特に断りのない限り、Postgres Enterprise Manager は、サーバー、データベース、スキーマ、拡張機能バージョン 8.4 以降、またはエージェントレベルで次のプローブを有効にします。

Name	Information monitored by probe	Level
Background Writer Statistics	バックグラウンドライタに関する情報を監視します。この情報には、次の数が含まれます。時限チェックポイント 要求されたチェックポイント 書き込まれたバッファチェックポイントによって 書き込まれたバッファバックグラウンドライタによって バックグラウンドライタサイクル 書き込まれたバックグラウンドバッファ 割り当てられたバッファ	Server
Barman Configuration	Barman ツールのグローバル構成に関する情報。	Agent
Barman Information	Barman ツールに関する情報。	Agent
Barman Server	Barman によって監視される各データベースサーバー構成に関する情報。	Agent
Barman Server Status	Barman によって監視される各データベースサーバーのステータスに関する情報。	Agent
Barman Server Backup	各データベースサーバーのバックアップに関する情報。	Agent
Barman Server WAL Status	Barman サーバーの WAL ファイルに関する情報。	Agent
Blocked Session Information	ブロックされたセッションに関する情報。	Server
CPU Usage	CPU 使用率情報。	Agent
Data and Log File Analysis	ログファイルに関する情報。情報には以下が含まれます。 ログファイルの名前 ログファイルが存在するディレクトリ	Server
Database Frozen XID	各データベースの凍結された XID。	Server

次のページに続く

表 11 – 前のページからの続き

Name	Information monitored by probe	Level
Database Size	監視対象データベースのサイズに関する情報。情報には次のものが含まれます。情報が収集された時間 データベース名 データベースサイズ MB 単位。	Server
Database Statistics	データベースの統計。情報には以下が含まれます。 バックエンドの数 コミットされたトランザクションの数 ロールバックしたトランザクションの数 読み取られたブロックの数 ブロックヒットの数 返された行数 フェッチされた行数 挿入された行数 更新された行数削除された行数	Server
Disk Busy Info	ディスクアクティビティに関する情報。 注このプローブは、Mac OS X、Solaris、または HP-UX ではサポートされていません。	Agent
Disk Space	ディスク領域の使用状況に関する情報。情報には次のものが含まれます 使用されたディスク領域の量 利用可能なディスク領域の量	Agent
EDB Audit Configuration	EDB Postgres Advanced Server の監査ログ構成 。	Server
Failover Manager Cluster Info	フェールオーバーマネージャークラスターを監視し、クラスターに関する情報を結果ます。このプローブは、フェールオーバーマネージャーバイナリのクラスター名前とパスがサーバーのプロパティダイアログボックスで提供されている場合に有効になります。	Server
Failover Manager Node Status	フェールオーバーマネージャークラスターを監視し、クラスター内の各ノードに関する詳細を結果ます。このプローブは、フェールオーバーマネージャーバイナリのクラスター名前とパスがサーバーのプロパティダイアログボックスで提供されている場合に有効になります。	Server
Function Statistics	データベースを監視し、機能に関する情報を取得します。情報には次のものが含まれます ファンクション名 引数型 戻り値	Database
Index Size	データベースを監視し、インデックスに関する情報を取得します。情報には次のものが含まれます。インデックスの名前 データが収集された時刻 インデックスのサイズ MB 単位	Database
Index Statistics	インデックス統計。情報には次のものが含まれます。 インデックススキャン数 読み取られた行数 取得された行数 読み取られたブロック数 ヒットしたブロック数	Database

次のページに続く

表 11 – 前のページからの続き

Name	Information monitored by probe	Level
Installed Packages	現在インストールされているパッケージ。収集される情報には次のものが含まれます。インストールされたパッケージの名前 インストールされたパッケージのバージョン プローブが実行された日時	Agent
IO Analysis	ディスク I/O 情報。情報には次のものが含まれます。 読み取られたブロック数 書き込まれたブロック数 プローブが実行された日時 注このプローブは Mac OS X ではサポートされていません。	Agent
Load Average	CPU 負荷平均。情報には次のものが含まれます。1 分間のロード平均 5 分間のロード平均 15 分間のロード平均 注 このプローブは Windows でサポートされていません。	Agent
Lock Information	ロック情報。情報には次のものが含まれます。データベース名 ロックタイプ ロックモード ロックを保持しているプロセス	Server
Memory Usage	システムメモリの使用量に関する情報。この情報には以下が含まれます。合計 RAM MB 単位の空き RAM MB 単位の合計スワップメモリ MB 単位の空きスワップメモリ MB 単位の共有システムメモリ MB 単位の共有システムメモリデータベースサーバーのメモリパラメーターを調整するためにチューニングウィザードによって使用されます。非 Windows システムでは、 *shmmax* **値であり、**/proc/sys/kernel/shmmax`から読み取られます。 - Windows、合計メモリと同じです。	Agent
Network Statistics	ネットワーク統計。この情報には以下が含まれます。インターフェイス IP アドレス 送信されたパケット数 受信されたパケット数 送信されたバイト数 受信されたバイト数 リンク速度 MB/秒	Agent
Number of Prepared Transactions	準備されたトランザクションの数を保存します。	Server
Number of WAL Files	WAL ファイルの数。	Server
Object Catalog: Database	データベースとそのプロパティのリストを監視します。情報には以下が含まれます。 データベース名 データベースエンコードタイプ データベースがユーザー接続またはシステム接続を許可するかどうか	Server

次のページに続く

表 11 – 前のページからの続き

Name	Information monitored by probe	Level
Object Catalog: Foreign Key	外部キーとそのプロパティのリストを監視します。情報には以下が含まれます。外部キーを含むテーブルの名前 外部キーが参照するテーブルの名前 テーブルが存在するデータベースの名前 テーブルが存在するスキーマの名前	Schema
Object Catalog: Function	ファンクションとそのプロパティのリストを監視します。情報には次のものが含まれます。ファンクションの名前 ファンクションが存在するスキーマの名前 ファンクションが存在するデータベースの名前	Schema
Object Catalog: Index	インデックスとそのプロパティのリストを監視します。情報には次のものが含まれます。インデックスの名前 インデックスが関連付けられているテーブルの名前 インデックス付きテーブルが存在するデータベースの名前	Schema
Object Catalog: Schema	スキーマとそれに関連するデータベースとサーバーのリストを監視します。	Database
Object Catalog: Sequence	シーケンスとそのプロパティのリストを監視します。	Schema
Object Catalog: Table	テーブル情報のリストを監視します。情報には以下が含まれます。テーブル名 テーブルが存在するスキーマの名前 スキーマが存在するデータベースの名前 テーブルに主キーがあるかどうかを示すブールインジケーター	Schema
Object Catalog: Tablespace	テーブルスペースのリストを監視します。	Server
Operating System Information	オペレーティングシステムの詳細と起動時間。	Agent
Package Catalog	現在インストール可能なパッケージ。収集される情報には次のものが含まれます。 パッケージ名 パッケージバージョン	Agent
PG HBA Conf	`pg_hba.conf` ファイルからの認証構成情報。	Server
Server Information	サーバー情報。	Server
Session Information	セッション情報。情報には次のものが含まれます。 セッションユーザーの名前 セッションがサーバーに接続した日時 情報が収集された時点のセッションのステータスアイドル、待機中などクライアントのアドレスとポート番号	Server
Settings	GUC 変数に現在割り当てられている値を監視します。	Server
SQL Protect	サーバーを監視し、 SQL インジェクション攻撃に関する情報を取得します。	Server

次のページに続く

表 11 – 前のページからの続き

Name	Information monitored by probe	Level
Slony Replication	Slony を使用して複製されたクラスターのラグデータ。	Database
Streaming Replication	ストリーミングレプリケーションを使用しているクラスターを監視し、以下に関する情報を取得します。送信された Xlog ロケーションバイト単位書き込み Xlog ロケーションバイト単位フラッシュ Xlog ロケーションバイト単位再生 Xlog ロケーションバイト単位 Xlog ラグ単位 Xlog ラグページ単位	Server
Streaming Replication Lag Time	ストリーミングレプリケーションを使用しているクラスターを監視し、以下に関する遅延情報を取得します。レプリケーション遅延時間秒単位レプリケーションの現在のステータス実行中/一時停止	Server
Streaming Replication Database Conflicts	ストリーミングレプリケーションを使用しているデータベースを監視し、発生した競合に関する情報を取得します。これには、次の数のためにキャンセルされたクエリに関する情報が含まれます ドロップテーブルスペースの競合 ロックタイムアウトの競合 古いスナップショットの競合 固定されたバッファの競合 デッドロックの競合	Server
Table Bloat	現在のテーブルの肥大化に関する情報。情報には以下が含まれます。テーブルの名前 テーブルが存在するスキーマの名前 推定ページ数 無駄なページの推定数 行ごとの推定バイト数	Database
Table Frozen XID	各テーブルのフリーズした XID を監視します。	Schema
Table Size	テーブルサイズに関する情報。テーブルサイズ MB 単位 合計インデックスサイズ MB 単位 合計テーブルサイズ、インデックスと TOAST を含む MB 単位	Database
Table Statistics	テーブル統計。情報には以下が含まれます。シークエンシャル スキャンの数 シークエンシャル スキャンの行数 インデックススキャンの数 インデックススキャン行の数 挿入された行数 更新された行数 削除された行数 ライブ行数 デッド数 rows 最後の VACUUM 最後の auto-vacuum 最後の ANALYZE 最後の auto-analyze ANALYZE によって推定されたページ数 ANALYZE によって推定された行数	Database
Tablespace Size	テーブルスペースとそのサイズのリスト。	Server

次のページに続く

表 11 – 前のページからの続き

Name	Information monitored by probe	Level
User Information	現在のユーザーのリスト。保存される情報には次のものが含まれます。 ユーザー名 ユーザータイプスーパーユーザーまたは非スーパーユーザーユーザーが接続しているサーバー	Server
WAL Archive Status	WAL アーカイブのステータス。保存される情報には以下が含まれます。完了した WAL アーカイブの数 保留中の WAL アーカイブの数 最後のアーカイブ時間 失敗した WAL アーカイブの数 最後の失敗の時間	Server
xDB Replication	xDB レプリケーションを使用してレプリケートされたクラスターの遅延データ。	Database
Multixact ID Exhaustion (Wraparound)	multixact ID に関する情報。保存される情報には次のものが含まれます。 データベースの名前 データベース内の最も古いマルチ xact ID ラップアラウンドに対するパーセンテージ 緊急自動バキュームに対するパーセンテージ	Server
Replication Slots	レプリケーションスロットに関する情報。保存される情報には以下が含まれます。レプリケーションスロットの一意のクラスター全体の識別子。 この論理スロットが使用している出力プラグインを含む共有オブジェクトのベース名、または物理スロットの場合は null。 スロットタイプ 物理的または論理。 このスロットが関連付けられているデータベースの OID、または null。論理スロットのみに、関連するデータベースがあります。 このスロットが関連付けられているデータベースの名前、または null。論理スロットのみに、関連するデータベースがあります。 このスロットが現在アクティブに使用されている場合、 Active フィールドは True です。 このスロットがデータベースで保持する必要がある最も古いトランザクション。 VACUUM は、後のトランザクションによって削除されたタプルを削除できません。このスロットがデータベースが保持する必要があるシステムカタログに影響を与える最も古いトランザクション。 VACUUM は、後のトランザクションによって削除されたカタログタプルを削除できません。 最も古い WAL のアドレス LSN。このスロットのコンシューマにまだ必要とされる可能性があるため、チェックポイント中に自動的に削除されません。	Server

EDB Postgres Distributed PGD プローブ

ダッシュボードを使用して EDB Postgres Distributed グループをモニタするには、次のプローブを有効にする必要があります。これらのプローブは拡張レベルで設定します。

注釈: バージョン 8.4 より前のバージョンでは、これらのプローブはすべてサーバーレベルで利用できました。

pgd_superuser を持つユーザーは、次のすべてのプローブからの情報を表示できます。4.0 より前のバージョンの EDB Postgres Distributed を使用している場合、これらのプローブはすべて EDB Postgres Distributed Enterprise Edition で動作します。

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Conflict History Summary	競合タイプごとの行の競合に関する情報。保存される情報には次のものが含まれます。 競合の現地時間 競合のタイプ	Yes	Yes
PGD Global Locks	EDB Postgres Distributed グループのグローバルロックに関する情報。保存される情報には以下が含まれます。グローバルロックが発生したノードの名前 ロックを保持しているプロセスの PID ロックの種類 DDL または DML ロックされたリレーションの名前 DML ロックの場合またはキーアドバイザリロックの場合ロック取得プロセスの内部状態 同じグローバルロックを待機するバックエンドのリスト グローバルロックの取得がオリジンノードによって開始された時間 ローカルノードがローカルロックの取得を試行し始めた時間 acquire_stage が最後に変更された時間	Yes	Yes
PGD Group Camo Details	EDB Postgres Distributed グループの迷彩に関する情報。保存される情報には以下が含まれます。ノードの名前 このノードがパートナーであるノード名 このノードがオリジンであるノード名 接続ステータス 準備状況ステータス 保留中または未解決の迷彩トランザクションの数 最後に適用された WAL ログの LSN 最後に受信した WAL ログの LSN	No	No

次のページに続く

表 12 – 前のページからの続き

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Group Replication Slot Details	EDB Postgres Distributed グループのレプリケーションスロットに関する情報。保存される情報には次のものが含まれます EDB Postgres Distributed グループの名前オリジンノードの名前ターゲットノードの名前このサブスクリプションが使用するオリジンノードのスロット名アクティブステータスレプリケーションの状態キャッチアップ、ストリーミング、切断,..) 報告された書き込みのおおよそのラグタイム 報告されたフラッシュのおおよそのラグタイム 報告されたリプレイのおおよそのラグタイム send_lsn と現在の WAL 書き込み位置のバイト差 write_lsn と現在の WAL 書き込み位置のバイト差 flash_lsn 間のバイト差と現在の WAL 書き込み位置	No	Yes
PGD Group Subscription Summary	EDB Postgres Distributed グループのサブスクリプションの概要に関する情報。保存される情報には以下が含まれます。サブスクリプションのオリジンの名前 サブスクリプションのターゲットの名前 最後に再生されたトランザクションのタイムスタンプ 現在と最後に再生されたトランザクションの時間のラグ	No	Yes
PGD Monitor Group Raft	クラスター全体の raft チェックのステータスとメッセージ。	Yes	Yes
PGD Group Raft Details	EDB Postgres Distributed グループ内のすべてのノードからの raft コンセンサスステータスに関する情報。保存される情報には以下が含まれます。ノードの名前 ノードの raft ワーカー状態 RAFT_LEADER のノード ID raft 選挙の内部 id raft スナップショットの内部 id	Yes	Yes
PGD Monitor Group Versions	クラスタ全体のバージョンチェックのステータスとメッセージ。	Yes	Yes

次のページに続く

表 12 – 前のページからの続き

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Group Versions Details	EDB Postgres Distributed グループ内の各ノードにインストールされている Postgres、pglogical、 EDB Postgres Distributed のバージョンの詳細に関する情報。保存される情報には以下が含まれます。ノードの名前 ノードにインストールされている Postgres バージョン ノードにインストールされている pg 論理バージョン ノードに分散されている EDB Postgres のバージョン 4.0 より前のバージョンの EDB Postgres Distributed エディション Standard/Enterprise 。	No	Yes
PGD Node Replication Rates	特定のノードからの発信レプリケーションアクティビティに関する情報。保存される情報には以下が含まれます。ターゲットピアノードの名前 最後に送信された位置 再生されたとして報告された最新の位置 報告されたリプレイのおおよそのラグタイム replay_lsn とオリジンの現在の WAL 書き込み位置のバイト差 replay_lsn と人間の可読バイト差現在の WAL 書き込み位置 ピアノードがまだ適用されていないすべての変更に追いつくまでに必要なおおよその時間	Yes	No
PGD Node Slots	ローカル EDB Postgres Distributed データベースノードのレプリケーションスロット、そのステータス、およびレプリケーションの進行状況へのマッピングに関する情報。保存される情報には以下が含まれます。スロットの名前 ターゲットノードの名前 EDB Postgres Distributed グループの名前 ターゲットノード上のデータベース名 スロットに接続されたプロセスの PID スロットに必要なカタログ XID IP アドレスクライアント接続の 最後に送信された位置 再生されたとして報告された最新の位置 報告された再生のおおよそのラグタイム attempts_lsn と現在の WAL 書き込み位置のバイト差 attempts_lsn と現在の WAL 書き込み位置の人間が判読可能なバイトの差	Yes	Yes

次のページに続く

表 12 – 前のページからの続き

Probe name	Information monitored by probe	*pgd_monitor* role required?	Works with PGD SE?
PGD Node Summary	EDB Postgres Distributed グループのすべてのノードに関する情報。保存される情報には以下が含まれます。ノードの名前 EDB Postgres の名前 ノードが属している分散グループ 人間が判読可能な形式でのノードの一貫した状態 ノードが到達しようとしている状態参加または昇格中にサブスクライブされた repset の名前	Yes	Yes
PGD Workers	EDB Postgres Distributed ノードのワーカーに関する情報。保存される情報には以下が含まれます。ワーカープロセスの PID ワーカークエリ開始タイムスタンプワーカー状態変更タイムスタンプワーカー待機イベントタイプワーカー待機イベントワーカー状態ワーカーロール名ワーカーコミットタイムスタンプワーカーローカルタイムスタンプ オリジンの名前ノード 受信 LSN 受信コミット LSN 最後の正確なリプレイ LSN 最後の正確なフラッシュ LSN 最後の正確なリプレイタイムスタンプワーカークエリー	Yes	Yes
PGD Work Errors	EDB Postgres Distributed データベースノードの作業エラーに関する情報。保存される情報には以下が含まれます。エラーを発生させたワーカーのプロセス ID ノードが一部である EDB Postgres Distributed グループの名前 オリジンノードの名前 ソースノードの名前 ターゲットノードの名前 サブスクリプションの名前このワーカーのロールの内部識別子 このワーカーのロールの名前 エラーの日時 エラーの経過時間 エラーの説明 エラーが発生したコンテキスト リモートリレーション ID サブスクリプションライタ ID サブスクリプションライタの名前	Yes	Yes

プローブのカスタマイズ

プローブは、特定の監視対象オブジェクトに関する一連のパフォーマンスメトリックを返すスケジュールされたタスクです。プローブは、監視対象サーバー、データベース、オペレーティングシステム、またはエージェントから統計を取得します。**Manage Probes** タブを使用して、デフォルト構成をオーバーライドし、各プローブの動作をカスタマイズできます。

[**Manage Probes**] タブを開くには、[**Management > Manage Probes**] を選択します。 **Manage Probes** タブは、プローブの作成および管理に使用できるアイコンセットを提供します。

- **Manage Custom Probes** を選択して [**Custom Probes**] タブを開き、カスタムプローブを作成または変更します。
- **Copy Probes** を選択して Copy Probe ダイアログボックスを開き、現在選択しているオブジェクトから 1 つ以上の監視対象オブジェクトにプローブ構成をコピーします。

注釈: コピープローブは、拡張レベルのプローブではサポートされていません。

プローブは、オブジェクトタイプサーバー、データベース、データベースオブジェクト、またはエージェントごとに一意のメトリックセットを監視します。ツリー内のオブジェクトの名前を選択して、そのオブジェクトのプローブを確認します。

プローブに関連付けられているプロパティを変更するには、プローブの名前を選択し、Probes テーブルに表示される設定をカスタマイズします。

- Execution Frequency 列の **Default** スイッチを **N** に移動して、**Minutes** および **Seconds** セレクターを有効にし、実行間の時間の長さのカスタム値を指定しますプローブ。
- Enabled?で **Default** スイッチを移動します。列を **No** して、プローブの状態を変更し、プローブがアクティブかどうかを示します。

注釈: 無効になったプローブからのデータがチャートで使用されている場合、チャートの左上隅に情報アイコンが表示され、提供されたリンクをクリックしてプローブを有効にできます。

- Data Retention 列の **Default** スイッチを **No** に移動して、**Day(s)** フィールドを有効にし、プローブによって収集された情報が PEM サーバーに保存される日数を指定します。

[**Manage Probes**] タブには、現在のノードから変更できないプローブに関する情報が表示される場合があります。現在のダイアログボックスからプローブを変更できない場合、スイッチは無効になっています。通常、無効になったプローブは、PEM クライアントツリーコントロールの階層の上位にあるノードから変更できます。ツリーコントロールで別のオブジェクトを選択して、[**Manage Probes**] タブに表示または有効化されているプローブを変更します。

カスタムプローブの作成

PEM Custom Probes タブを使用して、新しいプローブを作成するか、既存のカスタムプローブを変更できます。**Custom Probes** タブを開くには、 **Manage Probes** タブから **Manage Custom Probes** を選択します。

Show System Probes? スイッチを使用して、 **Custom Probes** タブでシステムプローブを表示または非表示にします。

Custom Probes タブを使用して、プローブを作成したり、既存のプローブを変更したりできます。プローブを作成するには

1. タブの右上隅にある **Add** を選択します。
2. 新しいプローブの名前を Probe Name 列に入力します。
3. プローブ名の左側にある **Edit** を選択して、プローブ定義を確認または追加します。

General タブを使用して、既存のプローブの定義を変更するか、新しいプローブのプロパティを指定します。

- **Probe Name** フィールドを使用して、新しいプローブの名前を提供します。
- **Collection Method** フィールドを使用して、プローブタイプを指定します。リストから次を選択します
  - SQL —プローブは、 SQL ステートメントを介して情報を収集します。
  - WMI —プローブは、 Windows Management Instrumentation 拡張機能を介して情報を収集します。
  - Batch —プローブは、コマンドスクリプトまたはシェルスクリプトを使用して情報を収集します。

Linux システムでバッチプローブを作成する前に、`agent.cfg` ファイルを変更し、 `allow_batch_probes` パラメーターを `true` に設定する必要があります。次に、PEM エージェントを再起動します。`agent.cfg` ファイルは、次のディレクトリのいずれかにあります。

- グラフィカルインストーラーを使用して PEM をインストールした場合`/opt/edb/pem/agent/etc/agent.cfg`
- RPM を使用して PEM をインストールした場合`/usr/edb/pem/agent/etc/agent.cfg`

Windows システムでは、エージェント設定はレジストリに保存されます。バッチプローブを作成する前に、`AllowBatchProbes` レジストリエントリのレジストリエントリを変更し、PEM エージェントを再起動します。PEM レジストリエントリは `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent` にあります。

バッチプローブはプラットフォーム固有です。 **Batch** の収集方法を指定する場合、 **Platform** フィールドでプラットフォームタイプを指定する必要があります。

Linux システムでスクリプトを呼び出すには、`agent.cfg` ファイルの `batch_script_user` パラメーターのエントリーを変更し、スクリプトの実行に使用するユーザーを指定する必要があります。このパラメーターには、非 root ユーザーまたは root を指定できます。ユーザーを指定しない場合、または指定されたユーザーが存在しない場合、スクリプトは実行されません。ファイルを変更した後、エージェントを再起動します。

Windows システムでスクリプトを呼び出すには、`AllowBatchJobSteps` のレジストリエントリを `true` に設定し、PEM エージェントを再起動します。

- **Target Type** リストを使用して、モニタするプローブのオブジェクトタイプを選択します。**収集方法** が WMI の場合、 **ターゲットタイプ**は無効になっています。プローブで **ターゲットタイプ**を構成して、収集されたメトリックがシステム階層の目的のレベルに関連し正確であることを確認します。ターゲットタイプを設定することにより、テーブルやインデックスなどの個々のデータベースオブジェクトからサーバー全体の監視に至るまで、さまざまなコンポーネントを効果的に監視できます。

ターゲットタイプ構成マトリックス

次の表は、ターゲットタイプの構成、その監視スコープ、およびそれらの構成のための必須フィールドの概要を提供します。

このテーブルでは

```
-   **Target type** 列は、プローブでモニタするオブジェクトを決定します。
```

- **実行レベル**列は、モニタリングプローブがデータベースごと、サーバーごと、エージェントごと、または拡張機能レベルごとに 1 回実行するかどうかを決定します。
- **必須列**列は、必要なデータが収集されるようにプローブクエリーで構成する必要がある列を示します。
- **プローブの例**列は、プローブがどのように使用されるかをより深く理解するために探索できる既存のプローブをいくつか提供します。実際に。

Target type	Execution level	Mandatory columns	Probe examples
Agent	Agent	None	cpu_usage
Server	Server	None	server_info
Database	Database	None	database_size
Schema	Database	schema_name	oc_extension
Table	Database	schema_name, table_name	table_size
Index	Database	schema_name, index_name	index_size
Sequence	Database	schema_name, sequence_name	oc_sequence
View	Database	schema_name, view_name	mview_size
Function	Database	schema_name, arg_types, function_type, function_name	function_statistics
Extension	Extension	None	bdr_node_summary

!!! note

1_blank_4_- データベース以上のターゲットタイプスキーマ、テーブル、インデックス、ビュー、シーケンス、ファンクションを含むに設定されたカスタムプローブは、データベースレベルで情報を収集します。 *1_blank_4_-* スキーマ以上のターゲットタイプに設定されたシステムプローブは、スキーマレベルまでの情報を収集します。 ..

プローブの **ターゲットタイプ**を適切に構成すると、収集されたメトリックがシステム階層の目的のレベルに関連しており、正確であることが保証されます。適切なターゲットタイプを設定することにより、ユーザーはサーバーレベルからテーブルやインデックスのような個々のデータベースオブジェクトに至るまで、さまざまなコンポーネントを効果的に監視できます。

- **Minutes** および **Seconds** セレクターを使用して、プローブがデータを収集する頻度を指定します。

- **Probe Enable?** スイッチを使用して、プローブをデフォルトで有効にするかどうかを指定します。デフォルトでプローブを有効にするには **Yes** を指定し、デフォルトでプローブを無効にするには **No** を指定します。

注釈: 無効になったプローブからのデータがチャートで使用されている場合、チャートの左上隅に情報アイコンが表示され、提供されたリンクをクリックしてプローブを有効にできます。

- **Data Retention** フィールドを使用して、収集された情報がプローブの履歴テーブルに保持される日数を指定します。
- **Discard fromhistory** の横にあるスイッチを使用して、サーバーがプローブの履歴テーブルを作成するかどうかを指定します。プローブ履歴を破棄する場合は **Yes** を選択し、テーブルにプローブ履歴を保持する場合は **No** を選択します。
- **Platform** リストを使用して、プローブがモニタするプラットフォームのタイプを指定します。このフィールドは、**Collection** 方法が **Batch** の場合にのみ有効になります。

Columns タブを使用して、プローブデータを保存する列を定義します。新しい列を定義するには

1. [**Columns**] タブに移動し、右上隅にある **Add** を選択します。
2. **Name** フィールドに、列の名前を入力します。
3. 新しい列名の左側にある **Edit** を選択して、列に関する情報を提供します。

- **Name** フィールドに列の説明的な名前を入力します。
- **Internal Name** フィールドは、カスタムプローブでは有効になっていません。
- **Column Type** リストを使用して、列がキー列主キー列であるか、非キー列であるかを指定します。非キー列は通常メトリック項目グラフ化できる値です。
- **Data Type** リストを使用して、列に保存するデータの種類を指定します。
- **Unit** フィールドを使用して、列に保存されているメトリックに適用される測定単位を指定します。この単位は、カスタムチャートまたは Capacity Manager チャートの y 軸に表示されます。これはオプショナルフィールドです。
- **Graphable** スイッチを使用して、定義したメトリックをグラフ化できるかどうかを指定し、Capacity Manager または Manage Charts ダイアログボックスからプローブにアクセスできるようにします。
- **Is PIT** スイッチを使用して、ポイントインタイムごとにメトリックを保存するかどうかを指定します。

ポイントインタイムメトリックは、任意の時点で変化します増加または減少します。たとえば、データベースサイズはポイントインタイムメトリックです。任意の時点で、データベースのサイズは変動します。ポイントインタイムではないメトリック累積メトリックとも呼ばれ、そのサイズは時間の経過とともに常に増加します。たとえば、Blocks Read と Tuples Read は累積メトリックです。値は同じか増加します。

- **Calculate PIT** スイッチを使用して、サーバーがメトリックデータのポイントインタイム値を計算するように指定します。**Is PIT** が **Yes** の場合、 **PIT の計算** は無効になっています。

PEM では、累積メトリックのポイントインタイム値を保存することもできます。PEM は、現在の値から累積メトリックの最後に収集された値を減算し、差をポイントインタイム値として保存します。

Code タブを使用して、実行するプロパティのデフォルトコードを指定します。

- プローブが SQL プローブの場合、**コード**タブでプローブによって呼び出される **SQL SELECT** ステートメントを指定する必要があります。クエリによって返される列名は、[**列**] タブで指定された **内部名**と一致する必要があります。クエリによって返される列の数、列名、データ型などは、**Columns** タブで指定された情報と一致する必要があります。
- プローブがバッチプローブの場合、プローブの実行時に呼び出すシェルまたは.bat スクリプトを指定する必要があります。スクリプトの出力は次のとおりです。
  - 最初の行には、 **Columns** タブに提供される列の名前が含まれている必要があります。各列名はタブ t 文字で区切る必要があります。2 行目以降、各行には、各列のデータをタブ文字で区切って含める必要があります。
  - 指定された列がキー列として定義されている場合、スクリプトが行をまたがってその列の重複データを生成しないことを確認します出力の。[**Columns**] タブで指定された列の数とその名前、データ型などは、スクリプト出力の出力と一致する必要があります。
- プローブが WMI プローブの場合、WMI クエリーを **SELECT WMI** クエリーとして指定する必要があります。 **SELECT** ステートメントで参照される列名は、 **Column** タブで指定された対応する列の名前と同じである必要があります。クエリによって返される列名は、[**列**] タブで指定された **内部名**と一致する必要があります。クエリによって返される列の数、列名、データ型などは、**Columns** タブで指定された情報と一致する必要があります。

Alternate Code タブを使用して、サーバーの特定のバージョンでプローブが起動した場合に呼び出すコードを提供します。バージョン固有のコードを提供するには、**Applies to any server version?** スイッチを **No** に移動し、**Add** を選択します。次に、**Server Version(s)** リストを使用してバージョンを選択し、**Edit** バージョン名の左側を選択して、プローブが起動したときに実行するコードを提供します。

データベースバージョンを選択し、[プローブコード] 列を空白のままにすると、PEM は、そのバージョンと一致するサーバーでプローブが実行されるときに、[**コード**] タブで指定されたコードを呼び出します。

プローブの定義が終了したら、[**カスタムプローブ**] タブの隅にある**保存**を選択して定義を保存し、プローブデータをカスタムチャートやグラフで使用できるようにします。

プローブのエクスポートまたはインポート

プローブをエクスポートするには、 **Manage Custom Probes** タブからプローブを選択し、テーブルの右上隅にある **Export** を選択します。 JSON ファイルを生成するには、 **Save File** を選択し、**OK** を選択します。

プローブをインポートするには、 **Manage Custom Probes** タブで、テーブルの右上隅にある **Import** を選択します。 **Browse** を選択して、インポートするプローブコードを含む JSON ファイルを選択し、**Import** を選択します。

インポートするファイルを選択した後、**Skip stable** チェックボックスを選択できます。このオプションは、プローブが既に存在する場合はスキップし、インポートがスキップされたことを通知するメッセージを表示します。

- チェックボックスを選択せず、プローブが既に存在する場合、プローブはインポートされず、エラーメッセージが報告されます。

注釈: **Import**は、構成された保持ポリシーに従って履歴データを保持するように構成されている場合があるため、既存のプローブを上書きできません。

- チェックボックスを選択せず、プローブは存在しないが、pem スキーマの対応するテーブルが存在する場合、同じテーブルを使用してプローブは正常にインポートされます。

注釈: プローブが削除され、**カスタムプローブの管理**タブにリストされず、そのプローブのデータを保持するテーブルが `pem` スキーマに存在する可能性があります。

プローブの削除

削除できるのはカスタムプローブのみです。プローブを削除するには、プローブテーブルでプローブ名前を選択し、テーブルの右上隅にある **Delete** を選択します。プローブ履歴は、プローブ定義の **履歴保持**フィールドで指定された時間持続します。削除中に、プローブ定義が削除され、対応するテーブルが pemdata および pemhistory スキーマから削除されます。

システムプローブは、PEM が提供するビルトインプローブであり、PEM スキーマの一部です。システムプローブのみを変更できます。システムプローブを削除しようとすると、PEM はエラーを報告します。

プローブのコピー

Copy Probe Configuration ダイアログボックスを使用して、1 つの監視対象オブジェクトから同じタイプの 1 つ以上の監視対象オブジェクトにプローブ定義をコピーできます。[Copy Probe Configuration] ダイアログボックスを開くには

1. PEM クライアントツリーで、プローブをコピーするオブジェクトを選択します。
2. **Management > Manage Probes** を選択します。
3. **Manage Probes** タブから、 **Copy Probe** を選択します。

ダイアログボックスは、Copy Probe Configuration ダイアログボックスを開いたオブジェクトからツリーコントロールで選択した場所にプローブ定義をコピーします。

Copy Probe Configuration ツリーで親ノードを指定すると、PEM は、ツリー内のそのノードの下にある同じタイプの各オブジェクトにプローブ構成をコピーします。たとえば、1 つのスキーマからデータベース内にあるすべてのスキーマにプローブ定義をコピーするには、ターゲットスキーマの親データベースのみを選択します。そのオブジェクトがコピーされるプローブのソースである場合、リストされたターゲットオブジェクトの名前の左側に赤い警告記号が表示されます。

ターゲットオブジェクトを選択したら、[**Configure Probes**] を選択して、ダイアログボックスで選択した場所にプローブ定義をコピーします。

pemworker を使用したプローブの管理

バージョン 9.6 から、 --probe-display-name 、--probe-internal-name 、または--probe-id オプションを使用して pemworker ユーティリティを使用してプローブを有効または無効にできます。

例

Linux の場合

次の例は、プローブ表示名を使用してプローブを無効にする方法を示しています。

```
export PEM_SERVER_PASSWORD=edb
./pemworker --pem-user enterprisedb --disable-probe --probe-display-name "cpu usage"
__OUTPUT__
Wed Mar 27 01:43:40 2024 INFO: Probe status updated for agent (id: 1)
```

この例は、プローブ表示名を使用してプローブを有効にする方法を示しています。

```
export PEM_SERVER_PASSWORD=edb
./pemworker  --pem-user enterprisedb --enable-probe --probe-display-name "database␣
↪size"
__OUTPUT__
Wed Mar 27 02:02:20 2024 INFO: Probe status updated for server (id: 1)
```

**Windows の場合*

次の例は、プローブ表示名を使用してプローブを無効にする方法を示しています。

```
set PEM_SERVER_PASSWORD=edb
./pemworker.exe DISABLE-PROBE --pem-user postgres --probe-display-name "CPU usage"
__OUTPUT__
INFO: Probe status updated for agent (id: 1)
```

この例は、プローブ表示名を使用してプローブを有効にする方法を示しています。

```
set PEM_SERVER_PASSWORD=edb
./pemworker.exe ENABLE-PROBE --pem-user postgres --probe-display-name "CPU usage"
__OUTPUT__
INFO: Probe status updated for agent (id: 1)
```

PEM プローブ - リファレンス

プローブは、PEM エージェントによって監視されているデータベースオブジェクトに関する情報を取得するスケジュールされたタスクです。PEM は、収集された情報を使用して、各ダッシュボードに表示されるグラフを構築します。**Manage Probes** タブ **Management** メニューからアクセスでは、データ収集スケジュールと、特定のプローブから返された情報を PEM が保持する期間を変更できます。

Name	Information monitored	Level
Background Writer Statistics	バックグラウンドライタに関する情報。この情報には以下が含まれます。時限チェックポイントの数 要求されたチェックポイントの数 書き込まれたバッファの数チェックポイントによって書き込まれたバッファの数バックグラウンドライタによって バックグラウンドライタのサイクル数 書き込まれたバックグラウンドバッファの数 割り当てられたバッファの数	Server
Blocked Session Information	ブロックされたセッションに関する情報。	Server
CPU Usage	CPU 使用率情報。	Agent
Data and Log File Analysis	ログファイルに関する情報。情報には以下が含まれます。 ログファイルの名前 ログファイルが存在するディレクトリ	Server
Database Frozen XID	各データベースの凍結された XID。	Server
Database Size	監視対象データベースのサイズに関する情報。情報には次のものが含まれます。情報が収集された時間 データベース名 データベースサイズ MB 単位。	Server
Database Statistics	データベースの統計。情報には、次の数が含まれます バックエンド コミットされたトランザクション トランザクションロールバック 読み取られたブロック ブロックヒット 返された行 取得された行 行挿入 更新された行 削除された行	Server
Disk Busy Info	ディスクアクティビティに関する情報。 注このプローブは、Mac OS X、Solaris、または HP-UX ではサポートされていません。	Agent
Disk Space	ディスク領域の使用状況に関する情報。情報には次のものが含まれます 使用されたディスク領域の量 利用可能なディスク領域の量	Agent
EDB Audit Configuration	EDB Postgres Advanced Server の監査ログ構成 。	Server

次のページに続く

表 14 – 前のページからの続き

Name	Information monitored	Level
Failover Manager Cluster Info	フェールオーバーマネージャークラスターを監視し、クラスターに関する情報を結果ます。このプローブは、[サーバーのプロパティ] ダイアログボックスでフェールオーバーマネージャーバイナリのクラスター名前とパスを指定すると、有効になります。	Server
Failover Manager Node Status	フェールオーバーマネージャークラスターを監視し、クラスター内の各ノードに関する詳細を結果ます。このプローブは、[サーバーのプロパティ] ダイアログボックスでフェールオーバーマネージャーバイナリのクラスター名前とパスを指定すると、有効になります。	Server
Function Statistics	データベースを監視し、機能に関する情報を取得します。情報には次のものが含まれます ファンクション名 引数型 戻り値	Database
Index Size	データベースを監視し、インデックスに関する情報を取得します。情報には次のものが含まれます。インデックスの名前 データが収集された時刻 インデックスのサイズ MB 単位	Database
Index Statistics	インデックス統計。情報には、次の数が含まれます インデックススキャン 読み取られた行 フェッチされた行 読み取られたブロック ブロックヒット	Database
Installed Packages	現在インストールされているパッケージ。収集される情報には次のものが含まれます。インストールされたパッケージの名前 インストールされたパッケージのバージョン プローブが実行された日時	Agent
IO Analysis	ディスク I/O 情報を監視します。情報には次のものが含まれます。 読み取られたブロック数 書き込まれたブロック数 プローブが実行された日時 注このプローブは Mac OS X ではサポートされていません。	Agent
Load Average	CPU 負荷平均。情報には次のものが含まれます。1 分間のロード平均 5 分間のロード平均 15 分間のロード平均 注 このプローブは Windows でサポートされていません。	Agent
Lock Information	ロック情報を監視します。情報には次のものが含まれます。 データベース名 ロックタイプ ロックモード ロックを保持しているプロセス	Server

次のページに続く

表 14 – 前のページからの続き

Name	Information monitored	Level
Memory Usage	システムメモリの使用量に関する情報。この情報には以下が含まれます。合計 RAM MB 単位の空き RAM MB 単位の合計スワップメモリ MB 単位の空きスワップメモリ MB 共有システムメモリ MB 単位の共有システムメモリデータベースサーバーのメモリパラメーターを調整するためにチューニングウィザードによって使用されます。非 Windows システムでは、 *shmmax*`値と/proc/sys/kernel/shmmax`から読み取られます。- Windows、合計メモリと同じです。	Agent
Network Statistics	ネットワーク統計。この情報には以下が含まれます。インターフェイス IP アドレス 送信されたパケット数 受信されたパケット数 送信されたバイト数 受信されたバイト数 リンク速度 MB/秒	Agent
Number of Prepared Transactions	準備されたトランザクションの数を保存します。	Server
Number of WAL Files	WAL ファイルの数。	Server
Object Catalog: Database	データベースとそのプロパティのリストを監視します。情報には以下が含まれます。 データベース名 データベースエンコードタイプ データベースがユーザー接続またはシステム接続を許可するかどうか	Server
Object Catalog: Foreign Key	外部キーとそのプロパティのリストを監視します。情報には以下が含まれます。外部キーを含むテーブルの名前 外部キーが参照するテーブルの名前 テーブルが存在するデータベースの名前 テーブルが存在するスキーマの名前	Schema
Object Catalog: Function	ファンクションとそのプロパティのリストを監視します。情報には次のものが含まれます。ファンクションの名前 ファンクションが存在するスキーマの名前 ファンクションが存在するデータベースの名前	Schema
Object Catalog: Index	インデックスとそのプロパティのリストを監視します。情報には次のものが含まれます。インデックスの名前 インデックスが関連付けられているテーブルの名前 インデックス付きテーブルが存在するデータベースの名前	Schema
Object Catalog: Schema	スキーマとそれに関連するデータベースとサーバーのリストを監視します。	Database
Object Catalog: Sequence	シーケンスとそのプロパティのリストを監視します。	Schema

次のページに続く

表 14 – 前のページからの続き

Name	Information monitored	Level
Object Catalog: Table	テーブル情報のリストを監視します。情報には以下が含まれます。テーブル名 テーブルが存在するスキーマの名前 スキーマが存在するデータベースの名前 テーブルに主キーがあるかどうかを示すブールインジケーター	Schema
Object Catalog: Tablespace	テーブルスペースのリストを監視します。	Server
Operating System Information	オペレーティングシステムの詳細と起動時間。	Agent
Package Catalog	現在インストール可能なパッケージ。収集される情報には次のものが含まれます。 パッケージ名 パッケージバージョン	Agent
PG HBA Conf	`pg_hba.conf` ファイルからの認証構成情報。	Server
Server Information	サーバー情報。	Server
Session Information	セッション情報。情報には次のものが含まれます。 セッションユーザーの名前 セッションがサーバーに接続した日時 情報が収集された時点のセッションのステータスアイドル、待機中などクライアントのアドレスとポート番号	Server
Settings	現在 GUC 変数に割り当てられている値。	Server
SQL Protect	サーバーを監視し、 SQL インジェクション攻撃に関する情報を取得します。	Server
Slony Replication	Slony を使用して複製されたクラスターのラグデータ。	Database
Streaming Replication	ストリーミングレプリケーションを使用しているクラスターを監視し、以下に関する情報を取得します。送信された Xlog の場所バイト単位書き込み Xlog ロケーションバイト単位フラッシュ Xlog ロケーションバイト単位再生 Xlog ロケーションバイト単位 Xlog ラグセグメント単位) Xlog ラグページ単位	Server
Streaming Replication Lag Time	ストリーミングレプリケーションを使用しているクラスターを監視し、以下に関する遅延情報を取得します。レプリケーション遅延時間秒単位 レプリケーションの現在のステータス実行中/一時停止	Server
Streaming Replication Database Conflicts	ストリーミングレプリケーションを使用しているデータベースを監視し、発生した競合に関する情報を取得します。これには、次の数のためにキャンセルされたクエリに関する情報が含まれます ドロップテーブルスペースの競合 ロックタイムアウトの競合 古いスナップショットの競合 固定されたバッファの競合 デッドロックの競合	Server

次のページに続く

表 14 – 前のページからの続き

Name	Information monitored	Level
Table Bloat	現在のテーブルの肥大化に関する情報。情報には以下が含まれます。テーブルの名前 テーブルが存在するスキーマの名前 推定ページ数 無駄なページの推定数 行ごとの推定バイト数	Database
Table Frozen XID	各テーブルの固定された XID。	Schema
Table Size	テーブルサイズに関する情報。テーブルサイズ MB 単位 合計インデックスサイズ MB 単位 合計テーブルサイズ、インデックスと TOAST を含む MB 単位	Database
Table Statistics	テーブル統計。情報には以下が含まれます。シークエンシャル スキャンの数 シークエンシャル スキャンの行数 インデックススキャンの数 インデックススキャン行の数 挿入された行数 更新された行数 削除された行数 ライブ行数 デッド数 rows 最後の VACUUM 最後の auto-vacuum 最後の ANALYZE 最後の auto-analyze ANALYZE によって推定されたページ数 ANALYZE によって推定された行数	Database
Tablespace Size	テーブルスペースとそのサイズのリスト。	Server
User Information	現在のユーザーのリスト。保存される情報には次のものが含まれます。 ユーザー名 ユーザータイプスーパーユーザーまたは非スーパーユーザー ユーザーが接続しているサーバー	Server
WAL Archive Status	WAL アーカイブのステータス。保存される情報には以下が含まれます。完了した WAL アーカイブの数 保留中の WAL アーカイブの数 最後のアーカイブ時間 失敗した WAL アーカイブの数 最後の失敗の時間	Server
xDB Replication	xDB レプリケーションを使用してレプリケートされたクラスターの遅延データを監視します。	Database

5.3.2 Alerts

PEM は、登録されたサーバーを継続的に監視します。パフォーマンスメトリックを、各統計について良好または許容可能なパフォーマンスを指定する事前定義されたユーザー指定のしきい値と比較します。許容可能なしきい値からの逸脱は、アラートをトリガーします。アラートは、PEM がシステム統計と比較するシステム定義またはユーザー定義の条件セットです。アラートは、注意が必要な登録サーバーの状態を通知します。

グローバルダッシュボードを介したアラートの表示

システム統計がその統計に指定された境界から逸脱すると、アラートがトリガーされます。アラートは、グローバル概要ダッシュボードのアラートステータステーブルの左端の列に、重大度の高赤、低黄、または中オレンジの警告を表示します。

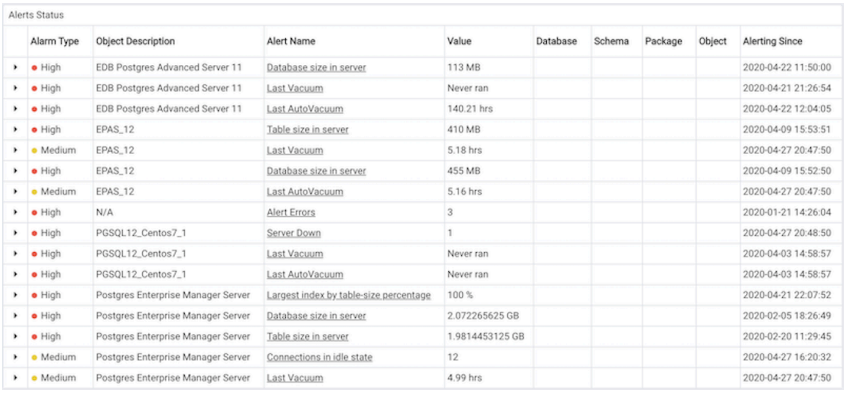


図 3: The Alert Status table

PEM サーバーには、サーバーをアクティブに監視する多数の事前定義されたアラートが含まれています。アラート定義により、アラートの原因に関する詳細を利用できる場合があります。重大度警告の右側にある下矢印を選択して、アラートをトリガーした条件の詳細を含むダイアログボックスを開きます。

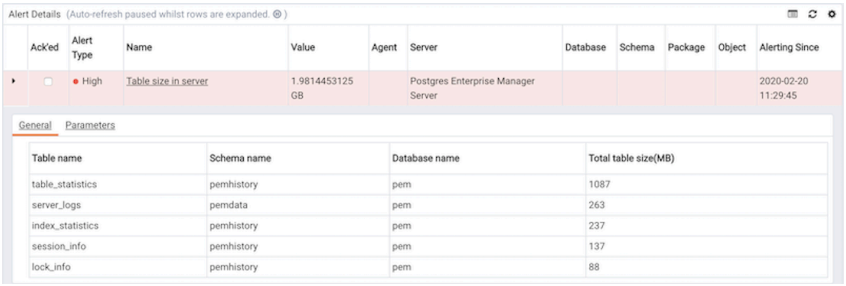


図 4: Alert details

PEM は、カスタマイズされたアラートを作成できるインターフェイスも提供します。各アラートは、アラートテンプレートで定義されたメトリックを使用します。アラートテンプレートは、サーバーがリソースまたはメトリックの統計を評価する方法を定義します。PEM サーバーには、事前定義されたアラートテンプレートが含まれており、カスタムアラートテンプレートを作成できます。

アラートダッシュボードを介したアラートの表示

Dashboards メニュー **Monitoring** タブを使用して、Alerts ダッシュボードを開きます。アラートダッシュボードには、アクティブなアラートの概要と各アラートのステータスが表示されます。

Alerts ダッシュボードヘッダーには、ダッシュボードが最後に更新された日時と、現在のアラートの数が表示されます。

[Alerts Overview] セクションには、アクティブなアラートの視覚的表現と、現在の高、低、および中アラートの数が表示されます。グラフの左側の垂直バーは、各列に表示されるアラートの数を提供します。バーにカーソルを合わせると、グラフの右上隅に選択したアラートの重大度のアラート数が表示されます。

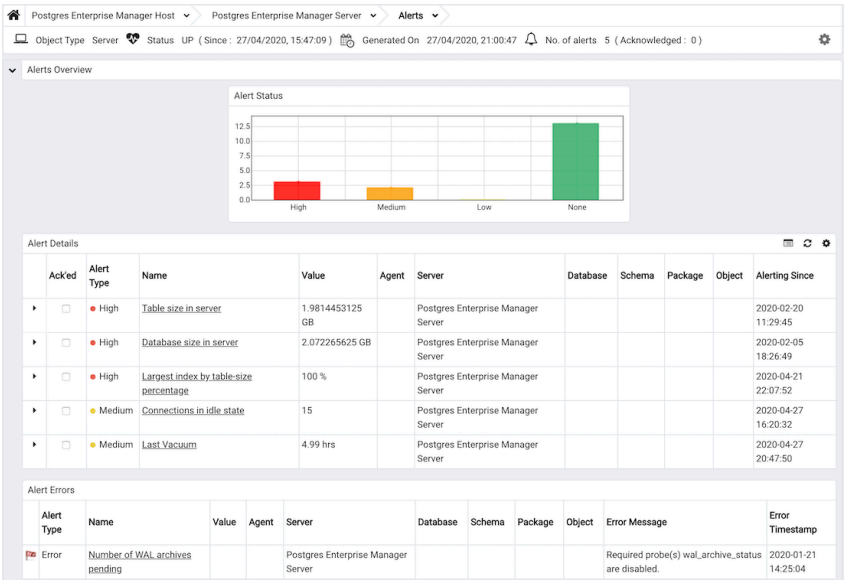


図 5: The Alerts Dashboard

Alert Details テーブルは、現在トリガーされているアラートのリストを提供します。エントリは、重大度の高いものから低いものの順に表示されます。各エントリには、アラートを識別し、アラートをトリガーした条件を認識できる情報が含まれています。アラートを選択して、アラート定義に関する詳細情報を確認します。

Alert Errors テーブルには、必要なプローブを誤って無効にしたり、アラートパラメーターの不適切な構成などの構成関連のエラーが表示されます。[Error Message] 列で提供される情報を使用して、エラーの原因となっている競合を特定して解決できます。

アラートダッシュボードのカスタマイズ

アラートダッシュボードに表示されるテーブルとグラフをカスタマイズできます。テーブルまたはグラフをカスタマイズするには、右上隅にある**設定** を選択します。

[Personalize Chart Configuration] ダイアログボックスのフィールドを使用して、表示設定を提供します。

- **Auto Refresh** フィールドを使用して、テーブルまたはチャートに表示されるデータの更新の間の秒数を指定します。
- **Download as** フィールドを使用して、チャートを JPEG 画像としてダウンロードするか、PNG 画像としてダウンロードするかを示します。
- **カラーセレクター** を使用して、チャートで使用する色を指定します。
- Ack'ed 列のチェックボックスで確認したアラートをテーブルに表示する場合は、**Show Acknowledged Alerts** スイッチを **Yes** に設定します。確認されたアラートを非表示にするには、 **No** に設定します。確認されたアラートは、アラート定義で指定された時間が経過した場合にのみテーブルコンテンツから削除されます。

カスタマイズを保存するには、右上隅の **Save** チェックマークを選択します。以前の変更を削除し、デフォルト値に戻すには、**Delete** を選択します。 **Save** および **Delete** メニューを使用して、設定をすべてのダッシュボードに適用するか、選択したサーバーまたはデータベースに適用するかを指定します。

アラートの管理

PEM クライアントの **Manage Alerts** タブを使用して、アラートを定義、コピー、または管理します。 [**Manage Alerts**] タブを開くには、[**Management > Manage Alerts**] を選択します。

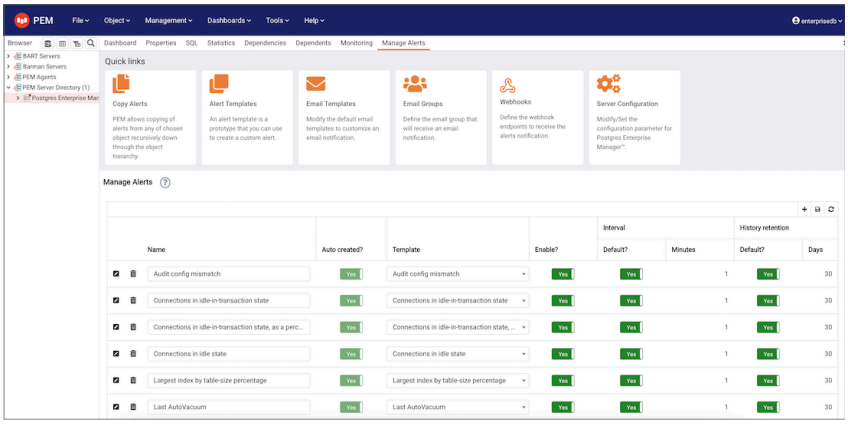


図 6: The Manage Alerts tab

クイックリンクツールバーを使用して、アラートを管理するためのダイアログボックスとタブを開きます。

- **Copy Alerts** を選択して Copy Alert Configuration ダイアログボックスを開き、アラート定義をコピーします。
- **Alert Templates** を選択して [**Alert Template**] タブを開き、アラートテンプレートを変更または作成します。
- **Send the notifications** を選択して [Email Template] ダイアログボックスを開き、デフォルトのメールテンプレートを変更してメール通知をカスタマイズします。
- **Send the notifications** を選択して [**Send the notifications**] タブを開き、メールグループを変更または作成します。
- **Webhooks** を選択して [**Webhooks**] タブを開き、Webhook エンドポイントを作成または管理します。
- **Server Configurations** を選択して Server Configuration ダイアログボックスを開き、サーバー構成設定を確認または変更します。
- **Help** を選択して、新しいタブで PEM オンラインヘルプを開きます。

Manage Alerts タブの Alerts セクションのテーブルを使用して、新しいアラートを作成するか、既存のアラートを管理します。

アラートテンプレート

アラートテンプレートは、アラートのプロパティを定義するプロトタイプです。アラートは、監視対象オブジェクトの現在の状態をアラートテンプレートで指定されたしきい値と比較して、管理上の注意が必要な状況かどうかを判断するようにサーバーに指示します。

Alert Templates タブを使用して、カスタムアラートテンプレートを定義したり、既存のアラートテンプレートの定義を表示したりできます。[**Alert Templates**] タブを開くには、[**Management > Manage Alerts**] を選択します。 [**Manage Alerts**] タブの [Quick Links] ツールバーで、[**Alert Templates**] を選択します。

Show System Template リストを使用して、Alert Templates テーブルに表示されるアラートテンプレートをフィルタリングします。リストから PEM 階層のレベルを選択して、そのレベルのすべてのテンプレートを表示します。

新しいアラートテンプレートの定義

新しいアラートテンプレートを定義するには、 **Show System Template** リストから **None** を選択します。次に、アラートテンプレートテーブルの右上隅にあるプラス記号+をクリックします。アラートテンプレートエディターが開きます。

General タブのフィールドを使用して、テンプレートに関する一般的な情報を指定します。

- **Template name** フィールドを使用して、新しいアラートテンプレートの名前を指定します。
- **Description** フィールドを使用して、アラートテンプレートの説明を提供します。
- **Target type** リストを使用して、アラートの焦点であるオブジェクトのタイプを選択します。
- **Applies to server** リストを使用して、アラートを適用するサーバータイプ EDB Postgres Advanced Server または PostgreSQL を指定します。単一のサーバータイプまたは **ALL** を指定できます。
- **History recovery** フィールドを使用して、PEM サーバーでアラートの実行の結果を保存する日数を指定します。
- **Thresholdunit** フィールドを使用して、しきい値の単位タイプを指定します。
- **Auto create** ボックスのフィールドを使用して、テンプレートを使用して自動アラートを生成するように PEM が指定します。このオプションを有効にすると、PEM は、**ターゲットタイプ**リストで指定された新しいサーバーまたはエージェントが追加されたときにアラートを作成し、ターゲットオブジェクトがドロップされたときにそのアラートを削除します。
  - **Auto create?** スライダーを **Yes** に設定して、テンプレートに基づいてアラートを作成するように PEM に指定します。**Auto create?** スライダーを **Yes** に変更して、既存のアラートテンプレートを変更すると、PEM は既存のエージェントとサーバーにアラートを作成します。スライダーを **Yes** から **No** に変更すると、既存のアラートのデフォルトのしきい値が消去され、復元することはできません。
  - **Operator** リストを使用して、演算子を選択します現在のシステム値を評価するときに使用する PEM。

システム値が **しきい値**フィールドに入力された値より大きい場合にアラートをトリガーするには、大なり記号>を選択します。

小なり記号<を選択して、システム値が **しきい値**フィールドに入力された値より小さい場合にアラートをトリガーすることを示します。

- しきい値フィールドを使用して、システム値と比較する PEM の値を指定して、アラートを生成するかどうかを決定します。3 つのしきい値すべての値を指定する必要があります低、中、および高。
- **Check frequency** フィールドを使用して、アラートを実行する間のデフォルトの分数を指定します。この値は、サーバーが定義で指定された SQL コードを呼び出し、結果をテンプレートで指定されたしきい値と比較する頻度を指定します。

Probe dependency タブのフィールドを使用して、 **SQL** タブで指定された SQL クエリーで参照されるプローブの名前を指定します。

- **Probes** リストを使用して、使用可能なプローブのリストから選択します。
  - アラートテンプレートで使用されるプローブのリストにプローブを追加するには、プローブ名を選択し、**Add** を選択します。
  - 選択したプローブリストからプローブを削除するには、プローブ名を選択し、**Delete** を選択します。
- **Parameters** タブを使用して、 **SQL** タブで指定された SQL コードで使用するパラメーターを定義します。プラス記号+を選択します。次に
  - **Name** フィールドを使用して、パラメーター名を指定します。
  - **Data type** リストを使用して、パラメーターのタイプを指定します。
  - **Unit** フィールドを使用して、指定された単位のタイプを指定しますパラメーターによる。
- **SQL** タブの **Code** フィールドを使用して、アラートの実行時にサーバーが呼び出す SQL クエリのテキストを提供します。SQL クエリーは、しきい値と比較する結果を提供します。アラート結果が指定されたしきい値から逸脱すると、アラートが発生します。

クエリでは、変数 param_**x** を使用して、 **Parameters** タブで定義されたパラメーターを順番に参照します。**x** は、パラメーターリスト内のパラメーター定義の位置を示します。たとえば、param_1 はパラメーターリストの最初のパラメーターを参照し、param_2 はパラメーターリストの 2 番目のパラメーターを参照します。

クエリーには、次の変数を含めることもできます。

Variable description	Variable name
agent identifier	'${agent_id}'
server identifier	'${server_id}'
database name	'${database_name}'
schema name	'${schema_name}'
Table	'${object_name}'
index	'${object_name}'
sequence	'${object_name}'
function name	'${object_name}'

- **Detailed Information SQL** フィールドを使用して、アラートがトリガーされた場合に呼び出す SQL クエリーを提供します。クエリの結果セットは、詳細なアラート情報の一部として、アラートダッシュボードまたはグローバル概要ダッシュボードに表示される場合があります。

注釈: 指定された照会が PEM 階層のさまざまなレベルサーバー、データベース、スキーマなどの 1 つ以上のプローブに依存し、プローブが無効になる場合、結果のアラートは次のように表示されます。

- アラート定義とクエリによって参照されるプローブが PEM 階層内の同じレベルからのものである場合、サーバーはグローバルアラートダッシュボードの Alert Error テーブルにアラートテンプレートを参照するアラートを表示します。 - アラート定義とクエリによって参照されるプローブが PEM 階層の異なるレベルからのものである場合、サーバーは、アラートが定義された階層の [アラートの詳細] テーブルにアラートテンプレートを参照するトリガーされたアラートを表示します。

アラートテンプレート定義を保存し、テンプレート名を Alert Templates リストに追加するには、 **Save** を選択します。カスタムアラートテンプレートを保存した後、[Alerting] ダイアログボックスを使用して、テンプレートに基づいてアラートを定義できます。

アラートテンプレートのエクスポートまたはインポート

アラートテンプレートをエクスポートするには

1. **Alert Templates** タブからアラートテンプレートを選択します。
2. テーブルの右上隅にある **Export** を選択します。
3. **Save File** を選択します。
4. JSON ファイルを生成するには、 **OK** を選択します。

アラートテンプレートをインポートするには

1. [**Alert Templates**] タブで、右上隅にある **Import** を選択します。
2. コードをインポートする JSON ファイルを選択するには、 **Browse** を選択し、 **Import** を選択します。
3. インポートするファイルを選択した後、次のチェックボックスを選択できます。
   - **既存のファイルをスキップ**—アラートテンプレートが既に存在する場合はスキップします。
   - **既存の依存プローブをスキップ**—アラートテンプレートはプローブに依存しています。依存プローブが既に存在する場合、スキップするにはこのチェックボックスを選択します。

両方のチェックボックスが選択され、アラートテンプレートが既に存在する場合、アラートテンプレートのインポートはスキップされます。

Skip stable チェックボックスを選択せず、 **Skip dependencies** を選択し、アラートテンプレートが既に存在する場合、アラートテンプレートは正常にインポートされます。

両方のチェックボックスがオフで、アラートテンプレートが存在しない場合、アラートテンプレートは正常にインポートされます。

アラートテンプレートの変更または削除

既存のテンプレート PEM の事前定義されたアラートテンプレートを含むの定義を表示するには、**Show System Template** リストを使用して監視対象オブジェクトのタイプを選択します。オブジェクトタイプを選択すると、そのオブジェクトタイプに対応するアラートテンプレートが Alert Templates テーブルに表示されます。

リストからテンプレート名を選択し、行の左端にある **Edit** を選択してテンプレート定義を確認します。

[Alert Templates] ダイアログボックスを使用して、アラートテンプレートに関する詳細情報を表示します。

- **General** タブには、一般的な情報が表示されます。
- **Probe dependency** タブには、テンプレートのデータを提供するプローブの名前がリストされます。
- **Parameters** タブには、SQL コードで参照されるパラメーターの名前がリストされます。
- [**SQL**] タブには、アラートの動作を定義する SQL コードが表示されます。

アラートテンプレートを削除するには、アラートテンプレートテーブルでテンプレート名を選択し、テーブルの右上隅にある **Delete** を選択します。アラート履歴は、テンプレート定義の **履歴保持**フィールドで指定された時間持続します。

事前定義されたアラートテンプレート – リファレンス

アラート定義には、PEM がシステム統計と比較するシステム定義またはユーザー定義の条件セットが含まれています。統計がその統計に指定された境界から逸脱すると、アラートがトリガーされ、PEM クライアントは [アラートの概要] ページに警告を表示し、オプションで監視ユーザーに通知を送信します。

以下の表は、アラートの作成に使用できるシステム定義のアラートテンプレートをリストしています。このリストは変更される場合があり、システムによって異なる場合があります。

エージェントに適用できるテンプレート

Template name	Description	Probe dependency
Load Average (1 minute)	1 分間のシステム負荷平均	load_average
Load Average (5 minutes)	5 分間のシステム負荷平均	load_average
Load Average (15 minutes)	15 分間のシステム負荷平均	load_average
Load Average per CPU Core (1 minutes)	CPU コアごとの 1 分間のシステム負荷平均	load_average
Load Average per CPU Core (5 minutes)	CPU コアごとの 5 分間のシステム負荷平均	load_average
Load Average per CPU Core (15 minutes)	CPU コアごとの 15 分間のシステム負荷平均	load_average
CPU utilization	平均 CPU 使用量	cpu_usage

次のページに続く

表 16 – 前のページからの続き

Template name	Description	Probe dependency
Number of CPUs running higher than a	K%の使用率しきい値を超えて実行されている CPU の数	cpu_usage
Free memory percentage	合計システムメモリの割合としての空きメモリ	memory_usage
Memory used percentage	使用されたメモリの割合	memory_usage
Swap consumption	消費されたスワップ領域メガバイト単位	memory_usage
Swap consumption percentage	消費されたスワップ領域の割合	memory_usage
Disk Consumption	消費されたディスク領域メガバイト単位	disk_space
Disk consumption percentage	消費されたディスクの割合	disk_space
Disk Available	使用可能なディスク領域メガバイト単位	disk_space
Disk busy percentage	ビジーなディスクの割合	disk_busy_info
Most used disk percentage	システムで最も使用されているディスクの使用率	disk_space
Total table bloat on host	ホスト上のテーブルによって浪費された合計領域 MB 単位	table_bloat, settings
Highest table bloat on host	ホスト上のテーブルによって浪費された最も多くの領域 MB 単位	table_bloat, settings
Average table bloat on host	ホスト上のテーブルによって浪費された平均領域 MB 単位	table_bloat, settings
Table size on host	ホスト上のテーブルのサイズ MB 単位	table_size
Database size on host	ホスト上のデータベースのサイズ MB 単位	database_size
Number of ERRORS in the logfile on agent N in last X hours.	過去 X 時間のエージェント N のログファイルの ERRORS の数	N/A
Number of ERRORS in the audit logfile on agent N in last X hours	過去 X 時間のエージェント N の監査ログファイルの ERRORS の数	N/A
Number of WARNINGS in the logfile on agent N in last X hours	過去 X 時間のエージェント N のログファイルの WARNING の数	N/A
Number of WARNINGS in the audit logfile on agent N in last X hours	過去 X 時間のエージェント N の監査ログファイルの WARNING の数	N/A
Number of WARNINGS or ERRORS in the logfile on agent N in last X hours	過去 X 時間のエージェント N のログファイルの WARNING または ERRORS の数	N/A
Number of WARNINGS or ERRORS in audit the logfile on agent N in last X hours	過去 X 時間のエージェント N のログファイルの WARNING または ERRORS の数	N/A
Package version mismatch	カタログごとにパッケージバージョンの不一致を確認します	N/A
Total materialized view bloat on host	ホスト上のマテリアライズドビューによって浪費された合計領域 MB 単位	mview_bloat, settings
Highest materialized view bloat on host	ホスト上のマテリアライズドビューによって無駄にされる最大の領域 MB 単位	mview_bloat, settings
Average materialized view bloat on host	ホスト上のマテリアライズドビューによって浪費される平均領域 MB 単位	mview_bloat, settings

次のページに続く

表 16 – 前のページからの続き

Template name	Description	Probe dependency
Materialized view size on host	ホスト上のマテリアライズドビューのサイズ MB 単位	mview_size
Agent Down	指定されたエージェントは現在ダウンしています	N/A

サーバーに適用できるテンプレート

Template name	Description	Probe dependency
Total table bloat in server	サーバー内のテーブルによって浪費された合計領域 MB 単位	table_bloat, settings
Largest table (by multiple of unbloated size)	サーバー内で最大のテーブル。肥大化されていない独自の推定サイズの倍数として計算されます。 N MB より小さいテーブルを除外する	table_bloat, settings
Highest table bloat in server	サーバー内のテーブルによって無駄にされた最も多くの領域 MB 単位	table_bloat, settings
Average table bloat in server	サーバー内のテーブルによって浪費された平均領域 MB 単位	table_bloat, settings
Table size in server	サーバー内のテーブルのサイズ MB 単位	table_size
Database size in server	サーバー内のデータベースのサイズ MB 単位	database_size
Number of WAL files	先行書き込みログファイルの合計数	number_of_wal_files
Number of prepared transactions	準備済み状態のトランザクションの数	number_of_prepared_transactions
Total connections	サーバー内の合計接続数	session_info
Total connections as percentage of max_connections	サーバーで許可される最大接続の割合としてのサーバー内の合計接続数、設定	session_info, settings
Unused, non-superuser connections	サーバー上の未使用の非スーパーユーザー接続の数、user_info、設定	session_info, user_info, settings
Unused, non-superuser connections as percentage of max_connections	max_connections、user_info、settings の max_connections の割合としてのサーバー上の未使用の非スーパーユーザー接続の数	session_info, user_info, settings
Ungranted locks	サーバー内の未付与のロックの数	blocked_session_info
Percentage of buffers written by backends	バックエンドによって書き込まれたバッファーの割合と書き込まれた合計バッファーの割合	background_writer_statistics
Percentage of buffers written by checkpoint	チェックポイントによって書き込まれたバッファーの割合と書き込まれた合計バッファーの割合	background_writer_statistics

次のページに続く

表 17 – 前のページからの続き

Template name	Description	Probe dependency
Buffers written per second	最後の 2 つのプローブサイクルで 1 秒ごとに書き込まれるバッファーの数	background_writer_statistics
Buffers allocated per second	最後の 2 つのプローブサイクルで、1 秒ごとに割り当てられたバッファーの数	background_writer_statistics
Connections in idle state	サーバー内のアイドル状態にある接続の数	session_info
Connections in idle-in-transaction state	アイドル中トランザクション状態にあるサーバー内の接続の数	session_info
Connections in idle-in-transaction state, as percentage of max_connections	サーバーで許可される最大接続の割合としての、トランザクション中アイドル状態にあるサーバー内の接続の数、設定	session_info, settings
Long-running idle connections	N 秒以上アイドル状態になったサーバー内の接続の数	session_info
Long-running idle connections and idle transactions	N 秒を超えてアイドル状態になったサーバー内の接続、またはトランザクション内アイドル状態のトランザクションの数	session_info
Long-running idle transactions	トランザクションで N 秒を超えてアイドル状態になったサーバー内の接続の数	session_info
Long-running transactions	N 秒以上実行されているサーバー内のトランザクションの数	session_info
Long-running queries	N 秒以上実行されているサーバー内のクエリーの数	session_info
Long-running vacuums	N 秒以上実行されているサーバー内のバキューム操作の数	session_info
Long-running autovacuums	N 秒以上実行されているサーバー内の自動バキューム操作の数	session_info
Committed transactions percentage	過去 N 分間にコミットしたサーバー内のトランザクションとロールバックしたトランザクションの割合	database_statistics
Shared buffers hit percentage	過去 N 分間、共有バッファによって満たされたサーバー内のブロック読み取り要求の割合	database_statistics
Tuples inserted	過去 N 分間にサーバーに挿入されたタプル	database_statistics
InfiniteCache buffers hit percentage	過去 N 分間、InfiniteCache によって満たされたサーバー内のブロック読み取り要求の割合	database_statistics
Tuples fetched	過去 N 分間にサーバーから取得されたタプル	database_statistics
Tuples returned	過去 N 分間にサーバーから返されたタプル	database_statistics
Dead Tuples	サーバー内の推定デッドタプルの数	table_statistics
Tuples updated	過去 N 分間にサーバーで更新されたタプル	database_statistics
Tuples deleted	過去 N 分間にサーバーから削除されたタプル	database_statistics
Tuples hot updated	過去 N 分間にサーバーでタプルがホット更新されました	table_statistics
Sequential Scans	過去 N 分間のサーバー内のフルテーブルスキャンの数	table_statistics

次のページに続く

表 17 – 前のページからの続き

Template name	Description	Probe dependency
Index Scans	過去 N 分間のサーバー内のインデックススキャンの数	table_statistics
Hot update percentage	過去 N 分間のサーバー内のホットアップデートの割合	table_statistics
Live Tuples	サーバー内の推定ライブタプルの数	table_statistics
Dead tuples percentage	サーバー内の推定デッドタプルの割合	table_statistics
Last Vacuum	サーバーでの最後のバキュームからの時間	table_statistics
Last AutoVacuum	サーバーでの最後の自動バキュームからの時間	table_statistics
Last Analyze	サーバーでの最後の分析からの時間	table_statistics
Last AutoAnalyze	サーバーでの最後の自動分析からの時間	table_statistics
Percentage of buffers written by backends over the last N minutes	バックエンドによって書き込まれたバッファの割合と最後の N のバッファバックエンドの合計	background_writer_statistics
Table Count	サーバー内のテーブルの合計数	oc_table
Function Count	サーバー内のファンクションの合計数	oc_function
Sequence Count	サーバー内のシーケンスの合計数	oc_sequence
Number of users expiring in N days	アカウントの有効期限が N 日になるユーザーの数	user_info
Number of users whose password expiring in N days	パスワードの有効期限が切れた、または N 日で有効期限が切れるユーザーの数	user_info
Index size as a percentage of table size	サーバー内のインデックスのサイズ、テーブルのサイズのパーセンテージ	index_size, oc_index, table_size
Largest index by table-size percentage	サーバー内の最大のインデックス、テーブルのサイズ、oc_index、table_size のパーセンテージとして計算されます。	index_size, oc_index, table_size
Number of ERRORS in the logfile on server M in the last X hours	過去 X 時間のサーバー M のログファイルの ERRORS の数	N/A
Number of WARNINGS in the logfile on server M in the last X hours	過去 X 時間のサーバー M のログファイルの WARNING の数	N/A
Number of WARNINGS or ERRORS in the logfile on server M in the last X hours	過去 X 時間のサーバー M のログファイルの WARNING または ERRORS の数	N/A
Number of attacks detected in the last N minutes	過去 N 分間に発生した SQL インジェクション攻撃の数	sql_protect
Number of attacks detected in the last N minutes by username	過去 N 分間に発生したユーザー名別の SQL インジェクション攻撃の数	sql_protect
Number of replica servers lag behind the primary by write location	ストリーミングレプリケーション レプリカサーバーの数は、書き込み場所ごとにプライマリよりも遅れています	streaming_replication
Number of replica servers lag behind the primary by flush location	ストリーミングレプリケーション レプリカサーバーの数は、フラッシュロケーションごとにプライマリよりも遅れています	streaming_replication

次のページに続く

表 17 – 前のページからの続き

Template name	Description	Probe dependency
Number of replica servers lag behind the primary by replay location	ストリーミングレプリケーション レプリカサーバーの数は、再生場所ごとにプライマリよりも遅れています	streaming_replication
Replica server lag behind the primary by write location	ストリーミングレプリケーション レプリカサーバーは、MB 単位の書き込み場所によりプライマリよりも遅延します。	streaming_replication
Replica server lag behind the primary by flush location	ストリーミングレプリケーション レプリカサーバーは、MB 単位のフラッシュロケーションにより、プライマリよりも遅延します。	streaming_replication
Replica server lag behind the primary by replay location	ストリーミングレプリケーション レプリカサーバーは、MB 単位の再生場所によりプライマリよりも遅延します。	streaming_replication
Replica server lag behind the primary by size (MB)	ストリーミングレプリケーション レプリカサーバーは MB 単位のサイズでプライマリよりも遅延しています	streaming_replication
Replica server lag behind the primary by WAL segments	ストリーミングレプリケーション レプリカサーバーは WAL セグメントごとにプライマリよりも遅延しています	streaming_replication
Replica server lag behind the primary by WAL pages	ストリーミングレプリケーション レプリカサーバーは WAL ページごとにプライマリよりも遅延します	streaming_replication
Total materialized view bloat in server	サーバー内のマテリアライズドビューによって浪費された合計領域 MB 単位	mview_bloat, settings
Largest materialized view (by multiple of unbloated size)	サーバー内で最大のマテリアライズドビュー。肥大化されていない独自の推定サイズの倍数として計算されます。 N MB より小さいマテリアライズドビューを除外する	mview_bloat, settings
Highest materialized view bloat in server	サーバー内のマテリアライズドビューによって浪費された最も多くの領域 MB 単位	mview_bloat, settings
Average materialized view bloat in server	サーバー内のマテリアライズドビューによって浪費される平均領域 MB 単位	mview_bloat, settings
Materialized view size in server	サーバー内のマテリアライズドビューのサイズ MB 単位	mview_size
View Count	サーバー内のビューの合計数	oc_views
Materialized View Count	サーバー内のマテリアライズドビューの合計数	oc_views
Audit config mismatch	監査構成パラメーターの不一致のチェック	audit_configuration
Server Down	指定されたサーバーは現在アクセスできません	N/A
Number of WAL archives pending	ストリーミングレプリケーション レプリカで再生保留中の WAL ファイルの数	wal_archive_status
Number of minutes lag of replica server from primary server	ストリーミングレプリケーション レプリカノードがプライマリノードから遅れている分数	streaming_replication_lag_time

次のページに続く

表 17 – 前のページからの続き

Template name	Description	Probe dependency
Log config mismatch	ログ構成パラメーターの不一致のチェック	log_configuration
PGD Group Raft Consensus	PGD グループ Raft コンセンサスが機能しない	bdr_monitor_group_raft
PGD Group Raft Leader ID not matching	PGD グループの Raft リーダー ID が一致しません	bdr_group_raft_details
PGD Group versions check	PGD グループ内の PGD/pg 論理バージョンが不一致です	bdr_monitor_group_raft
PGD worker error detected	PGD ノードで検出された PGD ワーカーエラーが報告されました	
Transaction ID exhaustion (wraparound)	トランザクション ID の枯渇の確認ラップアラウンド	
Inactive replication slots	特定のサーバーで非アクティブなスロットを確認する	
Conflicting replication slots	特定のサーバーで競合しているスロットを確認する	
Multixact ID exhaustion (wraparound)	multixact ID の枯渇の確認ラップアラウンド	

データベースに適用できるテンプレート

Template name	Description	Probe dependency
Total table bloat in database	データベース内のテーブルによって浪費された合計領域 MB 単位	table_bloat, settings
Largest table (by multiple of unbloated size)	データベース内で最大のテーブル。肥大化していない独自の推定サイズの倍数として計算されます。 N MB より小さいテーブルを除外する	table_bloat, settings
Highest table bloat in database	データベース内のテーブルによって浪費された最も多くの領域 MB 単位	table_bloat, settings
Average table bloat in database	データベース内のテーブルによって浪費された平均領域 MB 単位	table_bloat, settings
Table size in database	データベース内のテーブルのサイズ MB 単位	table_size
Database size	データベースのサイズ MB 単位	database_size
Total connections	データベース内の合計接続数	session_info
Total connections as percentage of max_connections	サーバーで許可される最大接続の割合としてのデータベース内の接続の合計数、設定	session_info, settings
Ungranted locks	データベース内の未付与のロックの数	blocked_session_info
Connections in idle state	データベース内のアイドル状態にある接続の数	session_info
Connections in idle-in-transaction state	データベース内のアイドルイントランザクション状態にある接続の数	session_info

次のページに続く

表 18 – 前のページからの続き

Template name	Description	Probe dependency
Connections in idle-in-transaction state,as percentage of max_connections	サーバーで許可される最大接続の割合としての、トランザクション中アイドル状態にあるデータベース内の接続の数、設定	session_info, settings
Long-running idle connections	N 秒以上アイドル状態であるデータベース内の接続の数	session_info
Long-running idle connections and idle transactions	N 秒を超えてアイドルまたはトランザクション中アイドル状態であったデータベース内の接続の数	session_info
Long-running idle transactions	トランザクションで N 秒を超えてアイドル状態になったデータベース内の接続の数	session_info
Long-running transactions	N 秒以上実行されているデータベース内のトランザクションの数	session_info
Long-running queries	N 秒以上実行されているデータベース内のクエリーの数	session_info
Long-running vacuums	N 秒を超えて実行されたデータベースのバキューム操作の数	session_info
Long-running autovacuums	N 秒以上実行されているデータベースの自動バキューム操作の数	session_info
Committed transactions percentage	過去 N 分間にコミットされたデータベース内のトランザクションとロールバックされたトランザクションの割合	database_statistics
Shared buffers hit percentage	過去 N 分間、共有バッファによって満たされたデータベース内のブロック読み取り要求の割合	database_statistics
InfiniteCache buffers hit percentage	過去 N 分間、InfiniteCache によって満たされたデータベース内のブロック読み取り要求の割合	database_statistics
Tuples fetched	過去 N 分間にデータベースから取得されたタプル	database_statistics
Tuples returned	過去 N 分間にデータベースから返されたタプル	database_statistics
Tuples inserted	過去 N 分間にデータベースに挿入されたタプル	database_statistics
Tuples updated	過去 N 分間にデータベースで更新されたタプル	database_statistics
Tuples deleted	過去 N 分間にデータベースから削除されたタプル	database_statistics
Tuples hot updated	過去 N 分間にデータベースのタプルがホット更新されました	table_statistics
Sequential Scans	過去 N 分間のデータベースのフルテーブルスキャンの数	table_statistics
Index Scans	過去 N 分間のデータベースのインデックススキャンの数	table_statistics
Hot update percentage	過去 N 分間のデータベースのホット更新の割合	table_statistics
Live Tuples	データベース内の推定ライブタプルの数	table_statistics
Dead Tuples	データベース内の推定デッドタプルの数	table_statistics
Dead tuples percentage	データベース内の推定デッドタプルの割合	table_statistics

次のページに続く

表 18 – 前のページからの続き

Template name	Description	Probe dependency
Last Vacuum	データベースの最後のバキュームからの時間	table_statistics
Last AutoVacuum	データベースの最後の自動バキュームからの時間	table_statistics
Last Analyze	データベースの最後の分析からの時間	table_statistics
Last AutoAnalyze	データベースの最後の自動分析からの時間	table_statistics
Table Count	データベース内のテーブルの合計数	oc_table
Function Count	データベース内のファンクションの合計数	oc_function
Sequence Count	データベース内の配列の合計数	oc_sequence
Index size as a percentage of table size	データベース内のインデックスのサイズ、テーブルのサイズのパーセンテージ	table_size
Largest index by table-size percentage	テーブルのサイズ、oc_index、table_size のパーセンテージとして計算されたデータベース内の最大のインデックス	index_size, oc_index, table_size
Database Frozen XID	データベースの凍結されたトランザクション ID の経過時間現在のトランザクションの前のトランザクション	database_frozenxid
Number of attacks detected in the last N minutes	過去 N 分間に発生した SQL インジェクション攻撃の数	sql_protect
Number of attacks detected in the last N minutes by username	過去 N 分ごとにユーザー名ごとに発生した SQL インジェクション攻撃の数	sql_protect
Queries that have been cancelled due to dropped tablespaces	ストリーミングレプリケーション テーブルスペースが削除されたためにキャンセルされたクエリの数	streaming_replication_db_conflicts
Queries that have been cancelled due to lock timeouts	ストリーミングレプリケーション ロックタイムアウトのためにキャンセルされたクエリの数	streaming_replication_db_conflicts
Queries that have been cancelled due to old snapshots	ストリーミングレプリケーション 古いスナップショットが原因でキャンセルされたクエリの数	streaming_replication_db_conflicts
Queries that have been cancelled due to pinned buffers	ストリーミングレプリケーション 固定されたバッファのためキャンセルされたクエリの数	streaming_replication_db_conflicts
Queries that have been cancelled due to deadlocks	ストリーミングレプリケーション デッドロックのためにキャンセルされたクエリの数	streaming_replication_db_conflicts
Total events lagging in all slony clusters	Slony レプリケーション すべての Slony クラスターで遅延しているイベントの合計	slony_cluster
Events lagging in one slony cluster	Slony レプリケーション 1 つのスロニークラスターで遅延しているイベント	slony_cluster
Lag time (minutes) in one slony cluster	Slony レプリケーション 1 つの slony クラスターでの遅延時間分	slony_cluster
Total rows lagging in xdb single primary replication	xDB レプリケーション xdb シングルプライマリレプリケーションの合計行遅延	xdb_smr_mmr_replication
Total rows lagging in xdb multi primary replication	xDB レプリケーション xdb マルチプライマリレプリケーションの合計行遅延	xdb_smr_mmr_replication

次のページに続く

表 18 – 前のページからの続き

Template name	Description	Probe dependency
Total materialized view bloat in database	データベース内のマテリアライズドビューによって浪費された合計領域 MB 単位	mview_bloat, settings
Largest materialized view (by multiple of unbloated size)	データベース内の最大のマテリアライズドビュー。肥大化されていない推定サイズの倍数として計算されます。 N MB より小さいマテリアライズドビューを除外する	mview_bloat, settings
Highest materialized view bloat in database	データベース内のマテリアライズドビューによって浪費された最も多くの領域 MB 単位	mview_bloat, settings
Average materialized view bloat in database	データベース内のマテリアライズドビューによって浪費された平均領域 MB 単位	mview_bloat, settings
Materialized view size in database	データベース内のマテリアライズドビューのサイズ MB 単位	mview_size
View Count	データベース内のビューの合計数	oc_views
Materialized View Count	データベース内のマテリアライズドビューの合計数	oc_views

スキーマに適用できるテンプレート

Template name	Description	Probe dependency
Total table bloat in schema	スキーマ内のテーブルによって浪費された合計領域 MB 単位	table_bloat, settings
Largest table (by multiple of unbloated size)	スキーマ内で最大のテーブル。肥大化されていない独自の推定サイズの倍数として計算されます。 N MB より小さいテーブルを除外する	table_bloat, settings
Highest table bloat in schema	スキーマ内のテーブルによって浪費された最大領域 MB 単位	table_bloat, settings
Average table bloat in schema	スキーマ内のテーブルによって浪費された平均領域 MB 単位	table_bloat, settings
Table size in schema	スキーマ内のテーブルのサイズ MB 単位	table_size
Tuples inserted	過去 N 分間にスキーマに挿入されたタプル	table_statistics
Tuples updated	過去 N 分間にスキーマで更新されたタプル	table_statistics
Tuples deleted	過去 N 分間にスキーマから削除されたタプル	table_statistics
Tuples hot updated	過去 N 分間にスキーマのタプルがホット更新されました	table_statistics
Sequential Scans	過去 N 分間のスキーマのフルテーブルスキャンの数	table_statistics
Index Scans	過去 N 分間のスキーマのインデックススキャンの数	table_statistics

次のページに続く

表 19 – 前のページからの続き

Template name	Description	Probe dependency
Hot update percentage	過去 N 分間のスキーマのホット更新の割合	table_statistics
Live Tuples	スキーマ内の推定ライブタプルの数	table_statistics
Dead Tuples	スキーマ内の推定デッドタプルの数	table_statistics
Dead tuples percentage	スキーマ内の推定デッドタプルの割合	table_statistics
Last Vacuum	スキーマの最後のバキュームからの時間	table_statistics
Last AutoVacuum	スキーマの最後の自動バキュームからの時間	table_statistics
Last Analyze	スキーマの最後の分析からの時間	table_statistics
Last AutoAnalyze	スキーマの最後の自動分析からの時間	table_statistics
Table Count	スキーマ内のテーブルの合計数	oc_table
Function Count	スキーマ内のファンクションの合計数	oc_function
Sequence Count	スキーマ内のシーケンスの合計数	oc_sequence
Index size as a percentage of table size	スキーマ内のインデックスのサイズ、テーブルのサイズのパーセンテージ	table_size
Largest index by table-size percentage	テーブルのサイズ、oc_index、table_size のパーセンテージとして計算されるスキーマ内の最大のインデックス	index_size, oc_index, table_size
Materialized view bloat	マテリアライズドビューによって無駄にされた領域 MB 単位	mview_bloat, settings
Total materialized view bloat in schema	スキーマのマテリアライズドビューによって浪費された合計領域 MB 単位	mview_bloat, settings
Materialized view size as a multiple of unbloated size	推定される肥大化されていないサイズの倍数としてのマテリアライズドビューのサイズ	mview_bloat
Largest materialized view (by multiple of unbloated size)	スキーマ内の最大のマテリアライズドビュー。肥大化されていない独自の推定サイズの倍数として計算されます。 N MB より小さいマテリアライズドビューを除外する	mview_bloat, settings
Highest materialized view bloat in schema	スキーマのマテリアライズドビューによって浪費される最大領域 MB 単位	mview_bloat, settings
Average materialized view bloat in schema	スキーマのマテリアライズドビューによって浪費される平均領域 MB 単位	mview_bloat, settings
Materialized view size	マテリアライズドビューのサイズ MB 単位	mview_size
Materialized view size in schema	スキーマのマテリアライズドビューのサイズ MB 単位	mview_size
View Count	スキーマ内のビューの合計数	oc_views
Materialized View Count	スキーマ内のマテリアライズドビューの合計数	ov_views
Materialized View Frozen XID	マテリアライズドビューの凍結されたトランザクション ID の経過時間現在のトランザクションの前のトランザクション	mview_frozenxid

テーブルに適用できるテンプレート

Template name	Description	Probe dependency
Table bloat	テーブルによって無駄にされた領域 MB 単位	table_bloat, settings
Table size	テーブルのサイズ MB 単位	table_size
Table size as a multiple of unbloated size	推定される肥大化されていないサイズの倍数としてのテーブルのサイズ	table_bloat
Tuples inserted	過去 N 分間にテーブルに挿入されたタプル	table_statistics
Tuples updated	過去 N 分間にテーブルで更新されたタプル	table_statistics
Tuples deleted	過去 N 分間にテーブルから削除されたタプル	table_statistics
Tuples hot updated	過去 N 分間にテーブル内のタプルがホット更新されました	table_statistics
Sequential Scans	過去 N 分間のテーブルのフルテーブルスキャンの数	table_statistics
Index Scans	過去 N 分間のテーブルのインデックススキャンの数	table_statistics
Hot update percentage	過去 N 分間のテーブル内のホット更新の割合	table_statistics
Live Tuples	テーブル内の推定ライブタプルの数	table_statistics
Dead Tuples	テーブル内の推定デッドタプルの数	table_statistics
Dead tuples percentage	テーブル内の推定デッドタプルの割合	table_statistics
Last Vacuum	テーブル上の最後のバキュームからの時間	table_statistics
Last AutoVacuum	テーブルの最後の自動バキュームからの時間	table_statistics
Last Analyze	テーブルでの最後の分析からの時間	table_statistics
Last AutoAnalyze	テーブルでの最後の自動分析からの時間	table_statistics
Row Count	テーブルの推定行数	table_statistics
Index size as a percentage of table size	テーブルのインデックスのサイズ、テーブルのサイズのパーセンテージ	table_size
Table Frozen XID	テーブルの凍結されたトランザクション ID の経過時間現在のトランザクションの前のトランザクション	table_frozenxid

グローバルテンプレート

Template name	Description	Probe dependency
Agents Down	最近報告されていないエージェントの数	N/A
Servers Down	現在アクセスできないサーバーの数	N/A
Alert Errors	エラー状態のアラートの数	N/A

監査ログアラート

PEM は、 Alerting ダイアログを使用して、ERROR または WARNING ステートメントが特定のサーバーまたはエージェントのログファイルに書き込まれたときにトリガーされるアラートを作成できるアラートテンプレートを提供します。Alerting ダイアログを開くには、PEM クライアントオブジェクトブラウザツリーコントロールでサーバーまたはエージェントを選択し、**Management > Alerting** を選択します。

特定のサーバーのログファイルにエラーまたは警告メッセージを通知するアラートを作成するには、次のいずれかのアラートテンプレートを使用するアラートを作成します。

- 過去 X 時間のサーバー M のログファイルの ERRORS の数
- 過去 X 時間のサーバー M のログファイルの WARNING の数
- 過去 X 時間のサーバー M のログファイルのエラーまたは警告の数

特定のエージェントのエラーまたは警告メッセージを通知するアラートを作成するには、次のいずれかのアラートテンプレートを使用するアラートを作成します。この機能は、EDB Postgres Advanced Server でのみサポートされています。

- 過去 X 時間のエージェント M のログファイルの ERRORS の数
- 過去 X 時間のエージェント M のログファイルの WARNING の数
- 過去 X 時間のエージェント M のログファイルのエラーまたは警告の数

新しいアラートの定義

PEM クライアントの **Manage Alerts** タブを使用して、アラートを定義、コピー、または管理します。 [**Manage Alerts**] タブを開くには、[**Management > Manage Alerts**] を選択します。

[**Manage Alerts**] タブには、PEM クライアントツリーで現在選択されているオブジェクトに定義されているアラートのテーブルが表示されます。Alerts テーブルを使用して、既存のアラートを変更したり、新しいアラートを作成したりできます。

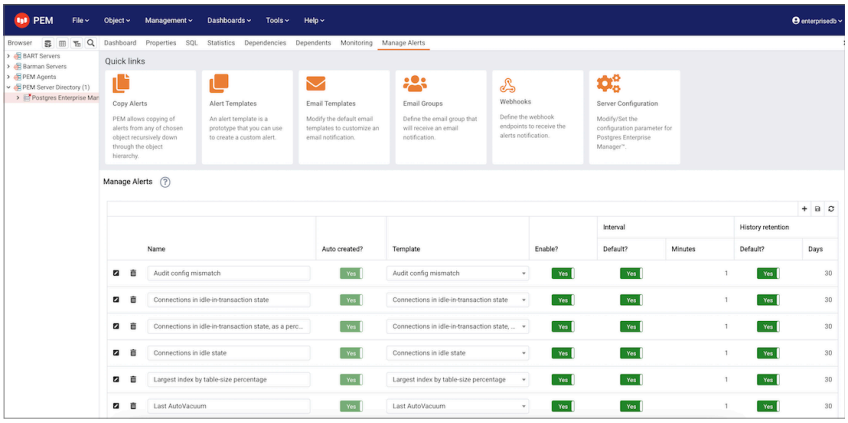


図 7: The Manage Alerts tab

アラートエディターを開き、アラートを作成するには、テーブルの右上にあるプラス記号+を選択します。エディターが開きます。

General タブのフィールドを使用して、アラートに関する情報を提供します。

- **Name** フィールドにアラートの名前を入力します。
- **Template** リストを使用して、アラートのテンプレートを選択します。アラートテンプレートは、1 つ以上のメトリックまたはパラメーターを使用して、PEM がユーザー指定のアラート境界を比較する値を生成するファンクションです。テンプレートファンクションによって返された値が、**演算子** および **しきい値**フィールドで指定されたユーザー定義のアラートの境界内の値と評価された場合、PEM:
  - アラートを発生させます b_tran_4 通知を追加します表示されるアラートの概要 b_tran_4 テンプレートで指定されたアクションを実行します
- **Enable?** スイッチを使用して、アラートを有効にする **Yes** か、無効にする **No** を指定します。
- **Interval** ボックスを使用して、アラート条件が満たされたかどうかをアラートが確認する頻度を指定します。**Minutes** セレクターを使用して、間隔値を指定します。 **Default** スイッチを使用して、**Minutes** 値を選択したテンプレートのデフォルト推奨値に設定またはリセットします。
- **History recovery** ボックスを使用して、アラートによって収集されたデータを PEM が保存する日数を指定します。**Days** セレクターを使用して、データを保存する日数を指定します。**Default** スイッチを使用して、 **Days** 値をデフォルト値 30 日に設定またはリセットします。
- **しきい値**ボックスのコントロールを使用して、アラートのトリガー基準を定義します。**しきい値**フィールドで指定された値がシステム値演算子で指定より大きいまたは小さいと評価されると、PEM は低、中、または高のアラートレベルを引き上げます。
- **Operator** リストを使用して、現在のシステム値を評価するときに使用する PEM の演算子を選択します。
  - システム値が入力された値より大きい場合にアラートをトリガーするには、大なり記号>を選択します**しきい値** フィールド.
  - システム値が **しきい値**フィールドに入力された値より小さい場合にアラートをトリガーするには、小なり記号<を選択します。
- **Threshold** フィールドを使用して、システム値と比較する PEM の値を指定して、アラートを生成するかどうかを決定します。3 つのしきい値すべての値を指定する必要があります低、中、および高。

Parameter Options テーブルには、選択したテンプレートに必要なパラメーターのリストが含まれています。テーブルには、事前定義されたパラメーターと値を指定する必要があるパラメーターの両方が表示されます。**Value** 列にプロンプトを表示するパラメーターの値を指定する必要があります。

PEM は、アラートがトリガーされた場合、またはアラートがクリアされた場合に、通知を送信したり、スクリプトを実行したりできます。**Notification** タブを使用して、アラートが生成された場合に PEM がどのように動作するかを指定します。

電子メール通知ボックスを使用して、指定したレベルでアラートがトリガーされた場合に電子メール通知を受信する電子メールグループを指定します。メールグループの作成 タブを使用して、アラートがトリガーされたときに通知するユーザーのアドレスを含むメールグループを作成します。[**Send the notifications**] タブにアクセスするには、[**Manage Alerts**] タブの [**Quick Links**] メニューにある **Email Groups** を選択します。

- 特定のアラートレベルに到達したときに電子メールを送信するように PEM に指示するには、アラートレベルの横にあるスライダーを **Yes** に設定します。リストを使用して、通知する事前定義されたユーザーまたはグループを選択します。

PEM が電子メール通知を送信する前に、SMTP サーバーを使用して電子メールを配信するように PEM サーバーを構成する必要があります。

指定したレベルでアラートがトリガーされる場合、 **Webhook notification** ボックスを使用して 1 つ以上のエンドポイントを指定します。*Webhook* を使用して、アラートがトリガーされたときに通知を受信する Webhook エンドポイントを作成します。[**Webhooks**] タブにアクセスするには、[**Manage Alerts**] タブの [**Quick Links**] メニューにある **Webhooks** を選択します。

- アラート通知を Webhook エンドポイントに送信するには、 **Enable?** を **Yes** に設定します。
- 要件に従ってカスタマイズされたアラートレベルを設定するには、**Overridedefault configuration?** を **Yes** に設定します。 **Yes** に設定すると、すべてのアラートレベルの構成が有効になります。
- リストを使用して、 **低アラート?** 、 **中程度のアラート?** 、**高アラート?** 、および **クリアされたアラート?** の通知を送信する事前定義されたエンドポイントを選択します。

Trap notification オプションを使用して、このアラートのトラップ通知を構成します。

- このアラートの状態が変更されたときに SNMP トラップ通知を送信するには、**Send attempts** を **Yes** に設定します。
- SNMP バージョンを識別するには、 **SNMP Ver** を **v1** 、 **v2** 、または **v3** に設定します。
- **低アラート** 、 **中アラート** 、および **高アラート**スライダーを使用して、トラップをトリガーするアラートレベルを選択します。たとえば、**高アラート** の横にあるスライダーを **Yes** に設定すると、高重大度レベルのアラートがトリガーされたときに PEM は通知を送信します。

通知を送信する前に、SNMP トラップ/通知レシーバーに通知を送信するように PEM サーバーを構成する必要があります。SNMP v3 トラップを送信するために、pemAgent は SNMP パケットの認証、暗号化、および復号化を担当する’User Security Model(USM)’を使用します。

SNMP v3 トラップの送信中に、エージェントは `snmp_boot_counter` ファイルを作成します。このファイルは、 `agent.cfg` の `batch_script_dir` パラメーターで指定された場所に作成されます。このパラメーターが構成されていない場合、または認証制限のためディレクトリにアクセスできない場合、ファイルはオペレーティングシステムの一時ディレクトリに作成されます。それも不可能な場合、ファイルはホームディレクトリに作成されます。

Nagios notification ボックスを使用して、アラートがトリガーまたはクリアされたときに Nagios ネットワークアラートソフトウェアに通知するように PEM サーバーに指示します。詳細はこちら *Using PEM with Nagios*

- アラートがトリガーまたはクリアされたときに Nagios に通知するには、**Submit Passive service check result to Nagios** スイッチを **Yes** に設定します。
- **Script attempts** ボックスを使用して、アラートがトリガーされた場合に実行するスクリプトをオプションで定義し、スクリプトの実行に関する詳細を指定します。
- **Execute script** スライダーを **Yes** に設定して、アラートがトリガーされた場合に提供されたスクリプトを実行するように PEM に指示します。

- **Execute on agent individually** スライダーを **Yes** に設定して、アラートをトリガーした状況が解決されたときに提供されたスクリプトを実行するように PEM に指示します。
- **Execute script on** オプションを使用して、PEM サーバーまたは監視対象サーバーで実行するスクリプトを示します。
- **Code** フィールドに、実行する PEM のスクリプトを入力します。バッチ/シェルスクリプトを提供できます。スクリプトでは、次のプレースホルダーを使用できます。

`%AlertName%` —トリガーされたアラートの名前。

`%ObjectName%` —アラートがトリガーされたサーバーまたはエージェントの名前。

`%ThresholdValue%` —アラートがトリガーされたときにメトリックが到達したしきい値。

`%CurrentValue%` —アラートをトリガーしたメトリックの現在値。

`%CurrentState%` —アラートの現在の状態。

`%OldState%` —アラートの以前の状態。

`%AlertRaisedTime%` —アラートが発生した時間、またはアラート状態が変更された最新の時間。

Linux システムでスクリプトを呼び出すには、`agent.cfg` ファイルの `batch_script_user` パラメーターのエントリーを変更し、スクリプトの実行に使用するユーザーを指定する必要があります。このパラメーターには、非 root ユーザーまたは root を指定できます。ユーザーを指定しない場合、または指定されたユーザーが存在しない場合、スクリプトは実行されません。ファイルを変更した後、エージェントを再起動します。

Windows システムでスクリプトを呼び出すには、`AllowBatchJobSteps` のレジストリエントリを true に設定し、PEM エージェントを再起動します。PEM レジストリエントリは `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent` にあります。

アラート属性を定義したら、 **Edit** を選択してアラート定義エディターを閉じ、Alerts テーブルの右上隅にある **Save** を選択します。

変更を破棄するには、 **Refresh** を選択します。変更を破棄することを確認するメッセージが表示されます。

注釈: 外部スクリプトでアラート構成プレースホルダー値を使用する必要があるとします。これは、コマンドライン引数として渡すか、環境変数としてエクスポートすることにより実行できます。外部スクリプトには、適切な実行権限が必要です。

- プレースホルダーをコマンドライン引数として使用してスクリプトを実行できます。

例

```
#!/bin/bash

bash <path_to_script>/script.sh "%AlertName%  %AlertLevel% %AlertDetails%"
```

- プレースホルダーの環境変数を定義し、スクリプトでそれらの環境変数を使用できます。

例

```
#!/bin/bash

export AlertName=%AlertName%
export AlertState=%AlertState%

bash <path_to_script>/script.sh
```

アラートの変更

Alerts テーブルを使用して、既存のアラートを管理するか、新しいアラートを作成します。PEM クライアントツリーでオブジェクトを選択して、そのオブジェクトを監視するアラートを表示します。

Alerts テーブルでアラートの一部のプロパティを変更できます。

- [Alert name] 列には、アラートの名前が表示されます。アラート名を変更するには、テーブル内の名前を置き換えて、**Save** を選択します。
- **アラートテンプレート**列には、アラートで使用されるプロパティを指定するアラートテンプレートの名前が表示されます。リストを使用して、アラートに関連付けられているアラートテンプレートを変更できます。
- **Alert allow?** スイッチを使用して、アラートを有効にする **Yes** か、無効にする **No** かを指定します。
- Interval 列を使用して、PEM がアラート条件が満たされるかどうかをチェックする頻度を指定します。**Default** スイッチを **No** に設定し、代替値を分単位で指定します。または、 **Default** スイッチを **Yes** に設定して、値をデフォルト設定にリセットします。デフォルトでは、PEM は 1 分ごとに各アラートのステータスを確認します。
- **History recovery** フィールドを使用して、アラートによって収集されたデータを PEM が保存する日数を指定します。**Default** スイッチを **No** に設定し、日単位の代替値を指定します。または、 **Default** スイッチを **Yes** に設定して、値をデフォルト設定にリセットします。デフォルトでは、PEM は 30 日間の履歴データを保存します。

アラートを変更した後、テーブルの右上隅にある **Save** を選択して変更を保存します。

他のアラート属性を変更するには、アラート名の左側にある **Edit** を選択してエディターを開きます。エディターは、完全なアラート定義へのアクセスを提供します。

[Alert Details] ダイアログボックスを使用して、選択したアラートの定義を変更します。アラート定義を変更した後、[**Save**] を選択します。

アラートの削除

削除するアラートをマークするには、Alerts テーブルでアラート名前を選択します。次に、名前の左側にある **Delete** を選択します。アラートはリストに赤い取り消し線フォントで残ります。

Delete はトグルです。2 回目に選択すると、削除を元に戻すことができます。アラート定義を永続的に検出するには、**Save** を選択します。

アラートのコピー

PEM システムでのアラートの展開を高速化するために、1 つのオブジェクトから 1 つ以上のターゲットオブジェクトにアラート定義をコピーできます。

オブジェクトからアラートをコピーするには、メイン PEM ウィンドウの PEM クライアントツリーでオブジェクトを選択します。次に、**Management > Copy Alerts** を選択します。 [**Manage Alerts**] タブで、[Quick Links] ツールバーから [**Copy Alerts**] を選択します。

Copy Alert Configuration ダイアログボックスは、PEM クライアントツリーで選択したオブジェクトからダイアログボックスで選択したオブジェクトにすべてのアラートをコピーします。ツリーを展開して、ターゲットオブジェクトとして指定するノードを選択します。ツリーには、ソースオブジェクトの横に赤い警告インジケータが表示されます。

アラートを一度に複数のオブジェクトにコピーするには、ターゲットの親ノードを選択します。たとえば、1 つのテーブルからスキーマ内のすべてのテーブルにアラートをコピーするには、スキーマの横にあるチェックボックスをオンにします。PEM は、ソースオブジェクトと同じタイプのターゲットにのみアラートをコピーします。

PEM がコピーされるものと同じ名前のターゲットオブジェクトの既存のアラートを更新しないようにするには、**Ignore Duplications** を選択します。

既存のアラートをソースオブジェクトからの同じ名前のアラートに置き換えるには、**Replace promotes** を選択します。

Delete Existing Alerts を選択して、ターゲットオブジェクトからすべてのアラートを削除し、すべてのアラートをソースオブジェクトからターゲットオブジェクトにコピーします。

Configure Alerts を選択して、ソースオブジェクトから Copy Alert Configuration ダイアログボックスで選択したオブジェクトまたはその下にある同じタイプのすべてのオブジェクトにアラートをコピーします。

アラートブラックアウトをスケジュールする

Management > Schedule Alert Blackout を使用して、メンテナンス中に Postgres サーバーと PEM エージェントのアラートブラックアウトをスケジュールできます。定義されたブラックアウト期間中にアラートは生成されません。

アラートブラックアウトをスケジュールするには、 **Management > Schedule Alert Blackout** を選択します。

Schedule Alert Blackout ダイアログボックスで、タブを使用してサーバーとエージェントのブラックアウト期間を定義します。[**Server**] タブで、行を追加するには、右上隅にあるプラス記号+を選択します。

[**Server**] タブを使用して、アラートのブラックアウト期間に関する情報を提供します。ブラックアウト期間を保存した後、編集することはできません。

- **開始時刻**フィールドを使用して、アラートブラックアウトを開始する日時を提供します。
- **Duration** フィールドを使用して、アラートをブラックアウトする間隔を提供します。
- **Servers** フィールドを使用して、アラートをブラックアウトするサーバー名を提供します。複数のサーバーを選択して、それらのすべてのサーバーのアラートをブラックアウトすることもできます。

詳細を入力したら、[**Save**] を選択します。アラートは、そのサーバーのスケジュールされた間隔でアラートダッシュボードに表示されません。

ダイアログボックスの **エージェント**タブを使用して、PEM エージェントのブラックアウト期間をスケジュールすることもできます。行を追加するには、**Agent** タブで、右上隅にあるプラス記号+を選択します。

[**Agent**] タブを使用して、アラートのブラックアウト期間に関する情報を提供します。ブラックアウト期間を保存した後、編集することはできません。

- **開始時刻**フィールドを使用して、アラートブラックアウトを開始する日時を提供します。
- **Duration** フィールドを使用して、アラートをブラックアウトする間隔を提供します。
- **Agents** フィールドを使用して、アラートを黒く塗りつぶすエージェントの名前を提供します。そのエージェントにバインドされたサーバーのすべてのサーバーレベルのアラートがブラックアウトします。

詳細を入力したら、**Save** を選択して詳細を保存します。アラートは、その PEM エージェントのスケジュールされた間隔でアラートダッシュボードに表示されません。

ダイアログボックスの右上隅から **Clone** を選択して、アラートブラックアウトのスケジュールを複製できます。選択したすべてのサーバーまたはエージェントのクローンコピーを作成するには、クローンを作成するサーバーまたはエージェントを選択し、**Clone** を選択します。必要に応じて、新しく作成したスケジュールを編集し、**Save** を選択できます。

ダイアログボックスの右上隅から **Delete** を選択して、スケジュールされたアラートブラックアウトを削除します。サーバーまたはエージェントを選択し、**Delete** を選択します。

スケジュールされたアラートブラックアウトを削除するサーバーを選択し、**Delete** を選択します。サーバーは、その行を削除する前に確認を求めます。

リセットを選択して、[アラートブラックアウト] ダイアログボックスの詳細をデフォルト設定にリセットできます。保存されたブラックアウトは影響を受けません。

スケジュールが実行されると、`pem` スキーマの `event_history` テーブルからスケジュールされたアラートのブラックアウトの詳細を表示できます。詳細については、*Monitoring event history* を参照してください。

5.3.3 Notifications

アラートがトリガーまたはクリアされた場合、PEM は通知を送信するか、スクリプトを実行できます。次のオプションを使用して通知を送信できます。

- SMTP
- Webhook
- SNMP
- Nagios

Notification タブを使用して、アラートが生成されたときの PEM 動作を指定します。

SMTP

PEM が電子メール通知を送信する前に、SMTP サーバーを使用して電子メールを配信するように PEM サーバーを構成する必要があります。

メールグループの作成

PEM は、注意が必要な条件がないかシステムを監視します。電子メールグループを使用して、サーバーが現在の値がアラート定義で指定されたしきい値から逸脱した場合に通知するユーザーの電子メールアドレスを指定できます。電子メールグループは、指定した期間に複数のユーザーに通知するか、特定のユーザーをターゲットにできます。

Send the notifications タブを使用して、SMTP メール受信者のグループを構成します。 [**Send the notifications**] タブを開くには、PEM クライアントで、[**Management > Manage Alerts**] を選択します。 [**Manage Alerts**] タブが開いたら、[Quick Links] ツールバーから [**Email Groups**] を選択します。

[**Send the notifications**] タブには、現在定義されているメールグループのリストが表示されます。既存のグループを変更するには、グループ名を選択し、行の左端にある **Edit** を選択します。

新しいメールグループを定義するには、 **メールグループ**テーブルの右上隅にあるプラス記号+を選択します。 [Email Groups] ダイアログボックスを使用して、電子メールグループとそのメンバーを定義します。

電子メールグループ定義の各行は、一連の電子メールアドレスを特定の期間に関連付けます。アラートがトリガーされると、サーバーは各行で指定された時間を評価し、アラートがトリガーされた時間に定義が関連付けられているグループメンバーにメッセージを送信します。

1. **Group Name** フィールドにメールグループの名前を入力します。
2. [**Options**] タブを開くには、グループメンバーテーブルでプラス記号+を選択します。
3. 指定した期間に通知を受信するようにメンバーアドレスを追加します。
    - **Reply to addresss** フィールドに受信者アドレスのコンマ区切りリストを入力します。
    - コピーを受信するアドレスのコンマ区切りリストを入力します。 **CC Addresses** フィールドのメール。

- **Bcc Addresses** フィールドに他の受信者に知られることなくメールのコピーを受信するアドレスのコンマ区切りリストを入力します。
- **From Address** フィールドからメッセージを送信します。
- **Subject prefix** フィールドを使用して、通知が送信されるときに各件名行の先頭に追加するメッセージを提供します。
- **From Time を使用する** および **To Time** セレクターを使用して、この行で指定されたグループメンバーへの通知の時間範囲を指定します。これらの値を PEM クライアントホストのロケールで提供します。PEM サーバーは、必要に応じて時間を他のタイムゾーンに変換します。

4. **Add** を選択してテーブルに行を追加し、別の期間とそれらの時間帯に通知する電子メールアドレスを指定します。
5. メールグループの定義が終了したら、[**Save**] を選択します。

メールグループを作成した後、 **Manage Alerts** タブを使用して、アラートの通知の詳細を設定し、グループに通知を転送できます。

メールグループの削除

電子メールグループを削除するには、電子メールグループテーブルでグループの名前を選択し、グループ名の左側にある **Delete** を選択します。

グループ名は [電子メールグループ] テーブルに赤で表示されます。 **Save** を選択して、テーブルからグループを完全に削除します。

Webhook

Webhook を使用して、構成済みのアプリケーションのしきい値違反に関するアラートイベントの通知を受信するように PEM サーバーを構成する必要があります。

Webhook の作成

PEM は、ユーザーの注意を必要とする条件がシステムを監視します。Webhook を使用して、現在の値がアラート定義で指定されたしきい値から逸脱した場合に通知を受信するエンドポイントを作成できます。トリガーされたイベントに基づいて、PEM は複数の Webhook エンドポイントまたは特定のターゲット Webhook エンドポイントに通知を送信します。

Webhooks タブを使用して、エンドポイントの受信者を構成します。 [**Webhooks**] タブを開くには、[**Management > Manage Alerts**] を選択します。 [**Manage Alerts**] タブの [Quick Links] ツールバーで、[**Webhooks**] を選択します。

[**Webhooks**] タブには、エンドポイントとして現在定義されている受信者アプリケーションのリストが表示されます。エンドポイントを選択し、行の左端にある **Edit** を選択して、既存のエンドポイントを変更します。

新しい Webhook を定義するには、テーブルの右上隅にあるプラス記号+を選択します。次に、**General** タブを使用して、Webhook の基本的な詳細を定義できます。

- Webhook の名前を **Name** フィールドに入力します。
- **URL** フィールドに、すべての通知を配信する Webhook URL を指定します。
- **Request Method** フィールドで、呼び出しに使用される要求方法タイプを設定します **POST** または **PUT** 。
- デフォルトでは、Webhook は有効になっています。Webhook を無効にするには、 **Enable?** を **No** に設定します。

注釈: **Enable?**設定は、*agent.cfg* ファイルで `enable_webhook` パラメーターが `true` に設定されている場合にのみ機能します。デフォルトでは、*enable_webhook* パラメーターは、PEM サーバーホストで実行されているエージェントに対してのみ `true` に設定されます。他のホストで実行されている他のすべてのエージェントの場合、手動で `true` に設定する必要があります。

Webhook SSL 構成の定義

Windows の各エージェント構成ファイルまたはレジストリで Webhook SSL パラメーターを定義できます。Webhook SSL パラメーターのリストは *Modifying agent configuration* にあります。エージェント構成パラメーターを追加または削除した場合、エージェントを再起動してそれらを適用する必要があります。

- Windows システムでは、Webhook の PEM レジストリエントリは `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent\WEBHOOK` にあります。
- Linux システムでは、Webhook の PEM 構成オプションは `agent.cfg file` に保存されます。デフォルトで`/usr/edb/pem/agent/etc` にあります。

```
[WEBHOOK/Django]
webhook_ssl_key=<webhook_client_ssl_key_path>
webhook_ssl_crt=<webhook_client_ssl_certificate_path>
webhook_ssl_ca_crt=<webhook_server_ca_certificate_path>
webhook_ssl_crl=<crl_file_path_to_validate_webhook_server>
allow_insecure_webhooks=<true|false>
```

図 8: Example - Webhook SSL Parameters in agent.cfg file

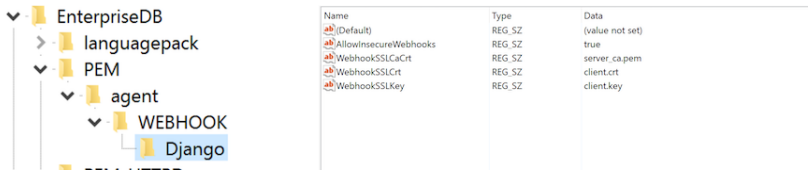


図 9: Example - Webhook SSL Parameters in windows registry

Webhook ヘッダーの定義

[**HTTP Headers**] タブを使用して、Webhook エンドポイントの呼び出し中に渡すヘッダーパラメーターを定義します。

- すべての値をキーと値のペアとして指定します。
- **Key** フィールドにキーパラメーターを指定し、 **Value** フィールドに値を指定します。
- HTTP ヘッダーを追加するには、HTTP Headers テーブルの右上隅にあるプラス記号+を選択します。
- HTTP ヘッダーを削除するには、 **Key** の左側にある **Delete** を選択します。ヘッダーはリストに残りますが、取り消し線フォントになります。**Save** を選択して、ヘッダーを完全に削除します。
- HTTP ヘッダーを編集するには、 **Key** の左側にある **Edit** を選択します。

Webhook ペイロードの定義

Payload タブを使用して、アラートがトリガーされたときにエンドポイントに送信する JSON データを定義します。

- **Type** は、形式タイプ JSON で送信するデータを指定します。
- **テンプレート**を使用して、エンドポイントに送信される JSON データを構成します。テンプレートでは、次のプレースホルダーを使用できます。
  - `%AlertID%` —トリガーされたアラートの ID。
  - `%AlertName%` —トリガーされたアラートの名前。
  - `%ObjectName%` —アラートがトリガーされたサーバーまたはエージェントの名前。
  - `%ObjectType%` —アラートが生成されたタイプ。
  - `%ThresholdValue%` —アラートがトリガーされたときにメトリックが到達したしきい値。
  - `%CurrentValue%` —アラートをトリガーしたメトリックの現在値。
  - `%CurrentState%` — agent.
  - `%OldState%` —の現在の状態 alert.
  - `%AlertRaisedTime%` —の以前の状態アラートが発生した時間、またはアラートの状態が変更された最新の時間。
  - `%AgentID%` —アラートが生成されたエージェントの ID。
  - `%AgentName%` —アラートが生成されたエージェントの名前。
  - `%ServerID%` —アラートが生成されたサーバーの ID。
  - `%ServerName%` —アラートが生成されたサーバーの名前。
  - `%ServerIP%` —アラートが生成されたサーバーの IP またはアドレス。
  - `%ServerPort%` —アラートが生成されたサーバーのポート。

  - %DatabaseName% —アラートが生成されたデータベースの名前。
  - %SchemaName% —アラートが生成されたスキーマの名前。
  - %PackageName% —アラートが生成されたパッケージの名前。
  - %DatabaseObjectName% —テーブル名やファンクション名などのデータベースオブジェクトの名前。アラートが生成された。
  - %Parameters% —アラートの生成に使用されるカスタムパラメーターのリスト。
  - %AlertInfo% —アラートの詳細なデータベースオブジェクトレベルの情報。
- **Test Connection** を選択して、記載されたエンドポイントへの通知配信をテストします。

Webhook アラートレベルの定義

Notifications タブを使用して、Webhook エンドポイントのアラートレベルを指定します。

- すべてのアラートレベルで通知を送信できるようにするには、**すべてのアラート** を **Yes** に設定します。
- 特定のアラートレベルに到達したときに通知を送信するには、アラートレベルの横にあるスライダーを **Yes** に設定します。個々のアラートレベルを構成するには、**すべてのアラート** を **No** に設定する必要があります。

例 Slack に通知を送信する

Slack では、 Getting started with incoming webhooks の指示に従って、次のことを行います。

- Slack アプリを作成します。
- そのアプリの着信 Webhook をアクティブにします。
- 選択したチャンネルまたはユーザーに投稿する Webhook を追加します。

新しい Webhook には、`https://hooks.slack.com/services/x/y/z` のような一意の URL があります。この URL に通知を送信するように PEM を構成できるようになりました。

PEM *Webhook の作成* で、わかりやすい名前を付け、前に取得した URL を **URL** フィールドにコピーします。 **Request Method** が **POST** に設定され、**Enable?** が **Yes** に設定されていることを確認します。 **Alert Notifications** の下のすべてのスライダーを **Yes** に設定します。

HTTP ヘッダーの下に、キー `Content-Type` と値 `application/json` を使用してヘッダーを追加します。

Payload の下で、デフォルトのテンプレートを削除し、次の例のように `text` をトップレベルキーとしてテンプレートを指定します。

```
{"text": "%AlertName% on %ObjectType% %ObjectName% is now %CurrentState%"}
```

これで、接続をテストできます。成功した場合、PEM は通知を発行し、指定したテンプレートがメッセージとして Slack チャネルに表示されます。

Webhook を保存し、通常のように PEM の使用を続行します。PEM はすべてのアラートを Slack チャネルに送信するようになりました。

Webhook の削除

削除の Webhook をマークするには、Webhooks テーブルで Webhook 名前を選択し、名前の左側にある **Delete** を選択します。アラートはリストに残りますが、取り消し線フォントになります。

Delete はトグルです。 2 度目に **Delete** を選択することにより、削除を元に戻すことができます。 **Save** を選択して、Webhook 定義を完全に削除します。

SNMP

通知を送信する前に、SNMP トラップ/通知レシーバーに通知を送信するように PEM サーバーを構成する必要があります。SNMP バージョンを識別するには、SNMP バージョンを v1、v2、または v3 に設定します。

例-net-snmp トラップレシーバーを使用して SNMP V3 トラップを構成する

1. [サーバーの構成] ダイアログボックスでプレーンテキスト形式で `snmp_security_engine_id ～PEM_SNMP_AGENT` を設定します。
2. プレーンテキスト値を 16 進形式に変換して、`snmptrapd.conf` ファイルで使用します。`snmp_security_engine_id` の 16 進値は、最大 32 オクテット長まで指定できます。

```
echo PEM_SNMP_AGENT | hexdump -v -e /1 "%02X"
50454D5F534E4D505F4147454E540A
```

1. Server Configuration ダイアログボックスで次のパラメーターを設定します。
   - **snmp_security_nam** から **pem_snmp_user**
   - **snmp_authentication_protocol** から **MD5**
   - **snmp_authentication_password** から **pem_auth_pass**
   - **snmp_privacy_protocol** から **DES**
   - **snmp_privacy_password** から **pem_priv_pass**
2. `snmptrapd.conf` ファイルには次の値があります。

edb_notranlate_2 !!!note エージェントを root として実行して、SNMP V3 で通知を送信できるようにする必要があります。 ## Nagios で PEM を使用する

PEM サーバーは、ユーザー定義のアラートがトリガーされると、パッシブアラート結果を Nagios ネットワークアラートソフトウェアに送信できます。トリガーされたアラートを Nagios に通知するように PEM サーバーに指示するには、次のことを行う必要があります。

- PEM サーバーから Nagios への通知をトリガーする各アラートの Nagios 通知を有効にします。`host.cfg` および `services.cfg` ファイルを作成する前に、PEM アラートを構成する必要があります。
- PEM サーバーの Nagios 関連の動作を構成します。
- `host.cfg` および `services.cfg` 構成ファイルを作成します。

- 必要に応じて、Nagios 構成ファイルを変更し、Nagios サーバーを再起動します。
- Nagios サーバーがインストールされているシステムに PEM エージェントをインストールし、PEM サーバーに登録します。そのエージェントの agent.cfg ファイルで enable_nagios 構成を true に設定し、エージェントサービスをリスタートします。

Nagios アラートを有効にするようにサーバーを構成すると、トリガーされたアラートはパッシブチェック結果を Nagios サービスに送信します。パッシブアラートの構文は次のとおりです。

```
<timestamp> PROCESS_SERVICE_CHECK_RESULT; <host_name> ; <service_name> ;
<service_status> ;
```

そこで

timestamp は、アラートがトリガーされた日時です。

host_name は、サーバーまたはエージェントの名前です。

service_name はアラートの名前です。

service_status は、サービスステータスの数値です。

- サービスステータスが OK の場合は 0
- サービスステータスが WARNING の場合 1
- サービスステータスが CRITICAL の場合 2
- 3 サービスステータスが UNKNOWN の場合

PEM サーバーは、次のルールを使用してサービスステータスを評価します。

- PEM 警告レベルが CLEARED の場合、警告メッセージは OK となります。
- PEM 警告レベルが低い場合、警告メッセージは WARNING となります。
- is_nagios_medium_alert_as_critical フラグ PEM サーバー構成ダイアログボックスで指定が FALSE に設定され、アラートレベル MEDIUM の場合、警告メッセージは WARNING となります。
- is_nagios_medium_alert_as_critical フラグ PEM サーバー構成ダイアログボックスで指定が TRUE に設定され、アラートレベルが MEDIUM の場合、警告メッセージは CRITICAL となります。
- PEM 警告レベルが HIGH の場合、警告メッセージは CRITICAL となります。

アラートの Nagios 通知を有効にする

PEM サーバーは、有効なアラートごとに一意の通知プロパティセットを維持します。**Manage Alerts** タブの **Notification** タブを使用して、トリガーされたときに、特定のアラートが Nagios にアラート通知を送信するように指定します。

アラートの通知プロパティを変更するには、アラートによって監視されるオブジェクトの名前を右クリックし、**管理 > アラートの管理** を選択します。 [**Manage Alerts**] タブで、Alerts リストでアラート名の左側にある **Edit** を選択します。編集ペインが開いたら、[**Notification**] タブを選択します。

Nagios 通知を有効にするには、**Nagios へのパッシブサービスチェック結果の送信**の横にスライダーを移動し、 **Yes** をクリックします。次に、 **Save** を選択します。

PEM サーバーの Nagios 関連動作の構成

[Server Configuration] ダイアログボックスを使用して、Nagios 構成に関する情報を PEM サーバーに提供できます。ダイアログボックスを開くには、**Management > Server Configuration** を選択します。

4 つのサーバー構成パラメーターは、Nagios インストールと Nagios に関連する PEM サーバーの動作に関する情報を指定します。

- `nagios_cmd_file_name` パラメーターを使用して、PEM からパッシブチェックアラートを受信する Nagios パイプラインファイルの場所を指定します。このパラメーターのデフォルト値は`/usr/local/nagios/var/rw/nagios.cmd` です。 `nagios.cmd` ファイルが他の場所にある場合、 **Value** フィールドにファイルの場所を指定します。
- `nagios_enabled` パラメーターのスライダーを **Yes** に移動して、パッシブチェックアラートを Nagios に送信するように PEM サーバーに指示します。
- `nagios_medium_alert_as_critical` スライダーを使用して、中アラートがトリガーされた場合に PEM サーバーが Nagios に渡す警告の重大度を指定します。
  - `is_nagios_medium_alert_as_critical` フラグが **FALSE** に設定され、アラートレベルが **MEDIUM** の場合、警告メッセージは WARNING.
  - `is_nagios_medium_alert_as_critical` フラグが **TRUE** に設定され、アラートレベルが **MEDIUM** の場合、警告メッセージは CRITICAL となります。
- `nagios_spool_retention_time` パラメーターを使用して、PEM サーバーに保存する通知履歴の日数を指定します。デフォルト値は 7 日です。

パラメーター値を変更した後、ダイアログボックスの右上隅にある **Save** を選択します。

hosts.cfg および services.cfg ファイルの作成

`templates.cfg` ファイルデフォルトでは`/usr/local/nagios/etc/objects` にありますは、汎用ホストと汎用サービスのプロパティを指定します。プロパティは、`hosts.cfg` および `services.cfg` ファイルで使用されるパラメーターを指定します。

ほとんどの場合、PEM がデフォルト構成でインストールされている場合、`hosts.cfg` および `services.cfg` ファイルを作成する前に `templates.cfg` ファイルを変更する必要はありません。必要に応じて、`templates.cfg` ファイルを変更して、パラメーターの代替値を指定したり、新しいテンプレートを作成したりできます。

Nagios 構成ファイルを変更する前に、次のコマンドを使用して、ローカルシステムにある PEM ホストに関する情報を含む `hosts.cfg` ファイルを作成します。

```
psql -U postgres -p 5433 -d pem -A -t -c "select pem.create_nagios_host_config('generic-host')"
> /usr/local/nagios/etc/objects/hosts.cfg
```

次に、次のコマンドを使用して、ローカルシステムにある PEM サービスに関する情報を含む `services.cfg` ファイルを作成します。

```
psql -U postgres -p 5433 -d pem -A -t -c "select pem.create_nagios_service_config('generic-service')"
> /usr/local/nagios/etc/objects/services.cfg
```

custom template.cfg ファイルエントリーを使用する場合は、これらのコマンドで generic-host または generic-service の代わりにエントリー名を指定します。

Nagios 構成ファイルの変更

host.cfg および services.cfg ファイルを作成した後、Nagios 構成ファイルデフォルトでは/usr/local/nagios/etc/nagios.cfg でそれらの場所を指定する必要があります。構成ファイルを変更し、ファイルの場所を指定するエントリを追加します。

```
cfg_file=/usr/local/etc/objects/hosts.cfg

cfg_file=/usr/local/etc/objects/services.cfg
```

次のコマンドを使用して、Nagios が適切に構成されていることを確認できます。

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

Nagios が正しく構成されていることを確認した後、Nagios サービスをリスタートします。

```
/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

メールテンプレートのカスタマイズ

PEM は、ユーザーの注意を必要とする条件がシステムを監視し、電子メール通知を送信します。**電子メールテンプレート**を使用して、電子メールの件名と、現在の値がアラートで指定されたしきい値から逸脱している場合にサーバーが送信するペイロードをカスタマイズします。

これらは、カスタマイズ可能なデフォルトのメールテンプレートです。

- アラートが検出されました
- 警戒レベルが上昇しました
- 警戒レベルの低下
- アラートがクリアされました
- アラートリマインダー
- 仕事の成功
- ジョブの失敗
- ジョブのキャンセル

・ジョブステップ

・ジョブステップデータベースサーバ

電子メールテンプレートをカスタマイズするには、電子メールテンプレート名の横にある edit ボタンを選択します。必要に応じて、使用可能なプレースホルダーのリストから **Subject** および **Payload** フィールドを編集します。フィールド値を変更した後、[**Save**] を選択します。

デフォルトのメールテンプレートがカスタマイズされている場合、**メールテンプレート** タブの Custom Template? 列にそのテンプレートの緑色のチェックマークが表示されます。

カスタマイズされたメールテンプレートをデフォルトのテンプレートとして復元するには、カスタムテンプレートを選択し、**メールテンプレート** タブの上部にある元に戻すボタンを選択します。

5.3.4 Capacity Manager

PEM の Capacity Manager は、収集された統計メトリックを分析して、オブジェクトの履歴使用統計を表示するグラフまたは表を生成します。オブジェクトの予想使用統計を投影することもできます。特定のホスト、サーバー、データベース、またはデータベースオブジェクトのメトリックを収集および分析するように Capacity Manager を構成できます。

以下を選択して、Capacity Manager レポートの内容を調整できます。

- レポートに含める特定のメトリック
- メトリックが収集された時間範囲
- 分析されるメトリックの高しきい値または低しきい値

Capacity Manager レポートの開始日と終了日を指定することもできます。レポートの終了日が将来の時刻を指定している場合、Capacity Manager は選択されたオブジェクトの使用履歴を分析し、将来の予測オブジェクト使用量を推定します。

Capacity Manager を開くには、PEM クライアントで、 **Management > Capacity Manager** を選択します。Capacity Manager ウィザードが開き、[**メトリック**] タブにツリーが表示されます。 [**Metrics**] タブで、ツリーを展開して、分析するノードのメトリックを確認します。レポートに含めるメトリックを選択します。

Capacity Manager は、 **Aggregation** リストで指定された集計方法を使用します。 Capacity Manager は、集約方法を使用してメトリック値を評価およびプロットします。以下から選択してください

- **平均** —期間中に記録された値の平均を使用します。
- **最大** —期間中に記録された最大値を使用します。
- **最小** —期間中に記録された最小値を使用します。
- **最初** —期間中に記録された最初の値を使用します。

Capacity Manager レポートからメトリックを削除するには、メトリックの名前の左側にあるボックスをクリアします。

Graph/chart metrics individually? の横にスライダーを移動し**をクリックして、** メトリック**タブで選択した各メトリックの個別のレポートを生成します。オプションが** No**に設定されている場合、選択されたすべてのメトリックが単一のグラフまたは表にマージされます。

Generate を選択してレポートを画面に表示するデフォルトの構成オプションを受け入れますか、**オプション**タブを使用してサンプリング境界、レポートタイプ、レポートの宛先をカスタマイズします。[**オプション**] タブに表示される時間は、PEM クライアントが存在するタイムゾーンからのものです。

Time Period ボックスのフィールドを使用して、Capacity Manager レポートの境界を定義します。

- **Period** リストを使用して、レポートに使用する期間の種類を選択します。以下を選択できます。

Start time and end time	Specify a start date and an end date/time for the report
Start time and threshold	Specify a start date and time and a threshold to determine the end time and date for the report.
Historical days and extrapolated days	Specify a start date for the report that is a number of days in the past and an end date that is a number of days in the future. This option is useful for report templates that don't specify fixed dates.
Historical days and threshold	Specify a start date that is a number of days in the past, and end it when a threshold value is reached.

レポートの期間の種類を指定した後、 **期間**ボックスの他のオプションから選択してレポートの期間を定義します。

- **開始時刻**フィールドの横にある日付と時刻セレクターを使用して、サンプリング期間の開始日時を指定します。または、レポートに含めるデータの履歴日数を選択します。**開始時刻**フィールドで指定された日付と時刻は、現在の日付/時刻より後であってはなりません。

デフォルトでは、Capacity Manager は現在の日時の 1 週間前の開始時刻を選択します。

- レポートの終了境界は、時間、将来の日数、または選択したメトリックが指定されたしきい値に到達したポイントです。**終了時間**の横にある日付と時刻セレクターを使用して、レポートの終了境界を指定します。または、レポートに含めるデータの推定日数を選択します。**終了時間** フィールドで指定された時間は、 **開始時間**フィールドで指定された時間よりも後でなければなりません。

将来の終了日時を選択した場合、Capacity Manager は使用履歴情報を使用して、予想される将来の使用量を推定します。予想される使用量は履歴データのサンプリングにもとづいていますので、サンプリング期間が長くなると、将来の使用傾向の精度が向上します。

しきい値を指定するには、 **Threshold** フィールドのリストを使用して、メトリック、演算子 **Exceeds** または **Falls Below** 、およびメトリックのターゲット値を選択します。しきい値を使用してレポートの終了を定義する場合、Capacity Manager レポートは、選択したメトリックの値が指定された値を超えるまたは下回ると終了します。

`cm_max_end_date_in_years` 構成パラメーターは、Capacity Manager レポートの終了境界のデフォルト時間値を定義します。レポートの終了境界としてしきい値を指定し、最大時間が経過する前に境界の予想使用量が満たされない場合、レポートは `cm_max_date_in_years` パラメーターで指定された時間に終了します。デフォルトでは、`cm_max_end_date_in_years` は 5 です。[サーバーの構成] ダイアログボックスを使用して、`cm_max_end_date_in_years` の値を変更できます。

Report ボックスのフィールドは、レポート種類と宛先を指定します。**含めるレポート** オプションを使用して、Capacity Manager によって生成されるレポートの種類を指定します。

- PEM クライアントウィンドウに折れ線グラフの形式でレポートを表示するには、**Graph** を選択します。
- **Table of data** を選択して、レポートデータを含む表を PEM クライアントウィンドウに表示します。
- PEM クライアントウィンドウに折れ線グラフとデータテーブルの両方を表示するには、**Graph and table of data** を選択します。

レポートの宛先オプションを使用して、レポートを表示または保存する場所を設定します。

- PEM クライアントの新しいタブにレポートを表示するには、**New tab** を選択します。 Capacity Manager レポートの第 1 世代を表示するには、**新しいタブ** を選択する必要があります。後続のレポートでは、**前のタブ** を選択できます。
- レポートを表示するときに以前に開いたタブを再利用するには、**前のタブ** を選択します。
- **Download the Report as a file** を選択し、ファイル名を指定して、指定したファイルにレポートを書き込みます。
- **Report Type?** で、 **HTML** または **JSON** タイプバージョン 9.6 以降を選択します。

レポート境界を指定し、Capacity Manager レポートの種類と宛先を選択した後、**Generate** を選択してレポートを作成します。

Capacity Vector HTML レポートは、スケーラブルベクターグラフィック SVG をサポートする Web ブラウザーで確認できます。SVG をサポートしていないブラウザーは Capacity Manager グラフを表示できません。そして不要な文字が含まれる場合があります。

Capacity Manager テンプレート

レポートを定義したら、今後のレポートのテンプレートとして定義を保存できます。すべての PEM ユーザーは、Capacity Manager レポートテンプレートにアクセスできます。レポート定義をテンプレートとして保存するには

1. **Metrics** および **Options** タブを使用して、レポートを定義します。
2. **Save** を選択します。
3. **Save Template** ダイアログボックスの **Title** フィールドにレポート名を入力し、ツリー内のテンプレートを保存する場所を選択します。
4. **OK** を選択します。

レポートを作成するときに、 **Load Template の Load Template** を使用して、既存のテンプレートを参照して開くことができます。開いたら、レポート定義を変更し、新しいテンプレートとして、または元のテンプレートを上書きして、再度保存できます。

Manage Templates を選択して、不要なテンプレートの名前を変更したり削除したりできるダイアログボックスを開きます。

Capacity Manager メトリック - リファレンス

使用可能な Capacity Manager メトリックはプラットフォームによって異なり、変更される場合があります。使用可能なメトリックには、表で説明されているメトリックを含めることができます。

Metric name	Description
# Dead Tuples	選択したテーブルのデッドタプルの数。
# Dead Tuples+	選択したテーブルのデッドタプルの累積数。

次のページに続く

表 23 – 前のページからの続き

Metric name	Description
# Heap Tuples Fetched by Index Scans	インデックススキャンによって取得されたヒープタプルの数。
# Heap Tuples Fetched by Index Scans	インデックススキャンによって取得されたヒープタプルの累積数。
# Idle Backends+	現在アイドル状態のバックエンドクライアントの累積数。
# Index Scans	指定されたオブジェクトで実行されたインデックススキャンの数。
# Index Scans+	指定されたオブジェクトで実行されたインデックススキャンの累積数。
# Index Tuples Read	読み取られたインデックスタプルの数。
# Index Tuples Read+	読み取られたインデックスタプルの累積数。
# Live Tuples	トランザクションに可視されるタプルの数。
# Live Tuples+	トランザクションに表示されるタプルの累積数。
# Pages Estimated by ANALYZE	ANALYZE によって推定されたページ数。
# Pages Estimated by ANALYZE+	ANALYZE によって推定された累積ページ数。
# Sequential Scans	特定のテーブルで実行されたシークエンシャル スキャンの数。
# Sequential Scans+	特定のテーブルで実行されたシークエンシャル スキャンの累積数。
# Sequential Scan Tuples	特定のテーブルで順次スキャンされたタプルの数。
# Sequential Scan Tuples+	特定のテーブルで連続してスキャンされたタプルの累積数。
# Tuples Deleted	削除されたタプルの数。
# Tuples Deleted+	削除されたタプルの累積数。
# Tuples Estimated by ANALYZE	ANALYZE によって推定されたライブ可視タプルの数。
# Tuples Estimated by ANALYZE+	ANALYZE によって推定されたライブタプルの累積数。
# Tuples HOT Updated	タプル HOT の数が更新されました。 HOT 更新では、新しいタプルは元のタプルと同じブロックにあり、タプルはインデックスエントリを共有します。
# Tuples HOT Updated+	更新されたタプル HOT の累積数。
# Tuples Inserted	指定されたテーブルに挿入されたタプルの数。
# Tuples Inserted+	指定されたテーブルに挿入されたタプルの累積数。
# Tuples Updated	選択したテーブルで更新されたタプルの数。
# Tuples Updated+	選択したテーブルで更新されたタプルの累積数。
Blocks Hit	キャッシュで見つかったブロックの数。
Blocks Hit+	キャッシュで見つかったブロックの累積数。
Blocks Read	読み取られたブロックの数。
Blocks Read+	読み取られたブロックの累積数。
Blocks Read from InfiniteCache	InfiniteCache から読み取られたブロックの数。
Blocks Read from InfiniteCache+	InfiniteCache から読み取られたブロックの累積数。
Blocks Written	書き込まれたブロックの数。
Blocks Written+	書き込まれたブロックの累積数。

次のページに続く

表 23 – 前のページからの続き

Metric name	Description
Buffers Allocated	割り当てられたバッファの数。
Buffers Allocated+	割り当てられたバッファの累積数。
Buffers Written - Backends	サーバープロセスクライアントアプリケーションに接続されているプロセスごとにディスクに書き込まれたバッファブロックの数。
Buffers Written - Backends+	サーバープロセスによってディスクに書き込まれたバッファブロックの累積数。
Buffers Written - Checkpoint	チェックポイントプロセスによってディスクに書き込まれたブロック数。
Buffers Written - Checkpoint+	チェックポイントプロセスによってディスクに書き込まれた累積ブロック数。
Buffers Written - Cleaning Scan	自動バキュームプロセスによってディスクに書き込まれたブロック数。
Buffers Written - Cleaning Scan+	自動バキュームプロセスによってディスクに書き込まれた累積ブロック数。
Bytes Received (KB)	クライアントから受信したバイト数キロバイト単位。
Bytes Received (KB)+	累積受信バイト数キロバイト単位。
Bytes Sent (KB)	クライアントに送信されたバイト数キロバイト単位。
Bytes Sent (KB)+	累積送信バイト数キロバイト単位。
Checkpoints - Timed	チェックポイント間隔によってトリガーされたチェックポイント操作の数。
Checkpoints - Timed+	チェックポイント間隔によってトリガーされたチェックポイント操作の累積数。
Checkpoints - Untimed	チェックポイントサイズによってトリガーされたチェックポイント操作の数。
Checkpoints - Untimed+	チェックポイントサイズによってトリガーされたチェックポイント操作の累積数。
Database Size (MB)	指定されたデータベースのサイズメガバイト単位。
Free RAM Memory	空き RAM メモリの量メガバイト単位。
Free Swap Memory	ディスク上の空きスワップ領域の量メガバイト単位。
Heap Blocks Hit	キャッシュで見つかったヒープブロックの数。
Heap Blocks Hit+	キャッシュで見つかったヒープブロックの累積数。
Heap Blocks Read	読み取られたヒープブロックの数。
Heap Blocks Read+	読み取られたヒープブロックの累積数。
Index Blocks Hit	キャッシュで見つかったインデックスブロックの数。
Index Blocks Hit+	キャッシュで見つかったインデックスブロックの累積数。
Index Blocks Read	読み取られたインデックスブロックの数。
Index Blocks Read+	読み取られたインデックスブロックの累積数。
Index Size (MB)	指定されたインデックスのサイズメガバイト単位。
In Packets Discards	破棄されたインバウンドパケットの数。
In Packets Discards+	破棄されたインバウンドパケットの累積数。
In Packets Errors	エラーを含む受信パケットの数。
In Packets Errors+	エラーを含む受信パケットの累積数。
Link Bandwidth (Mbit/s)	ネットワークアダプターの速度メガビット/秒。
Load Average - 15 Minute	CPU 飽和パーセント単位-15 分のサンプリング平均。

次のページに続く

表 23 – 前のページからの続き

Metric name	Description
Load Average - 1 Minute	CPU 飽和パーセント単位-1 分間のサンプリング平均。
Load Average - 5 Minute	CPU 飽和パーセント単位-5 分間のサンプリング平均。
Load Percentage	パーセント単位の CPU 飽和度。
Number of Prepared Transactions+	準備されたトランザクションの累積数。
Number of WAL Files+	ログ先行書き込みファイルの累積数。
Out Packets Discards	破棄された送信パケットの数。
Out Packets Discards+	破棄された送信パケットの累積数。
Out Packets Errors	エラーを含む送信パケットの数。
Out Packets Errors+	エラーを含む送信パケットの累積数。
Packets Received	受信したパケット数。
Packets Received+	受信したパケットの累積数。
Packets Sent	送信されたパケット数。
Packets Sent+	送信されたパケットの累積数。
Size (MB)	ディスクの合計サイズメガバイト単位。
Size of Indexes (MB)	指定されたテーブルのインデックスのサイズメガバイト単位。
Space Available (MB)	現在使用可能なディスク領域メガバイト単位。
Space Used (MB)	現在使用されているディスク領域メガバイト単位。
Table Size (MB)	指定されたテーブルのサイズメガバイト単位。
Tablespace Size (MB)	指定されたテーブルスペースのサイズメガバイト単位。
Temp Buffers (MB)	一時バッファのサイズメガバイト単位。
Toast Blocks Hit	キャッシュで見つかった TOAST ブロックの数。
Toast Blocks Hit+	キャッシュで見つかった TOAST ブロックの累積数。
Toast Blocks Read	読み取られた TOAST ブロックの数。
Toast Blocks Read+	読み取られた TOAST ブロックの累積数。
Total RAM Memory	システム上の RAM メモリの合計量メガバイト単位。
Total Swap Memory	システム上のスワップ領域の合計量メガバイト単位。
Total Table Size w/Indexes and Toast	指定されたテーブルの合計サイズインデックスおよび関連する特大属性を含む
Transactions Aborted	中止されたトランザクションの数。
Transactions Aborted+	中止されたトランザクションの累積数。
Transactions Committed	コミットされたトランザクションの数。
Transactions Committed+	コミットされたトランザクションの累積数。
Tuples Deleted	指定されたテーブルから削除されたタプルの数。
Tuples Deleted+	指定されたテーブルから削除されたタプルの累積数。
Tuples Estimated by ANALYZE	指定されたテーブルにある可視のタプルの数。
Tuples Estimated by ANALYZE+	指定されたテーブル内の可視タプルの累積数。
Tuples Fetched	指定されたテーブルから取得されたタプルの数。
Tuples Fetched+	指定されたテーブルから取得されたタプルの累積数。

次のページに続く

表 23 – 前のページからの続き

Metric name	Description
Tuples HOT Updated	タプル HOT の数が更新されました。 HOT 更新では、新しいタプルは元のタプルと同じブロックにあり、タプルはインデックスエントリを共有します。
Tuples HOT Updated+	更新されたタプル HOT の累積数。 HOT 更新では、新しいタプルは元のタプルと同じブロックにあり、タプルはインデックスエントリを共有します。
Tuples Inserted	指定されたテーブルに挿入されたタプルの数。
Tuples Inserted+	指定されたテーブルに挿入されたタプルの累積数。
Tuples Returned	結果セットで返されたタプルの数。
Tuples Returned+	結果セットで返されたタプルの累積数。
Tuples Updated	指定されたテーブルで更新されたタプルの数。
Tuples Updated+	指定されたテーブルで更新されたタプルの累積数。
WAL Segment Size (MB)	ログ先行書き込みのセグメントサイズメガバイト単位。

注釈: メトリックの名前の後にある+は、メトリックのデータが累積的に収集されることを意味します。 '+' 記号が続いていないメトリックは、ポイントインタイム値として収集されます。

5.3.5 Audit Manager

PEM Audit Manager を使用して、 EDB Postgres Advanced Server インスタンスの監査ログ構成を簡素化できます。 Audit Manager を使用すると、次のようなロギング属性を構成できます。

- PEM がログファイルを収集する頻度
- ログファイルに含まれるデータベースアクティビティの種類
- ログファイルをローテーションする頻度とタイミング

監査ログには、次のアクティビティが含まれる場合があります。

- データベースインスタンスに対して行われたすべての接続
- 失敗した接続試行
- データベースインスタンスからの切断
- すべてのクエリー SELECT ステートメント
- すべての DML ステートメント INSERT、UPDATE、DELETE
- すべての DDL ステートメントたとえば、 CREATE、DROP、ALTER

監査ログが PEM サーバーに保存されると、監査ログダッシュボードを使用して、読みやすい形式で情報を確認できます。Audit Log ダッシュボードでは、次の基準でログファイルをフィルタリングできます。

- タイムスタンプ範囲アクティビティが発生したとき
- アクティビティが発生したデータベース

- アクティビティを実行するユーザー
- 呼び出されるコマンドの種類

EDB Postgres Advanced Server インスタンスサービス ID の設定

EDB Postgres Advanced Server インスタンスのロギングを構成するには、サーバーがバインドされたエージェントを備えた PEM 管理サーバーであり、サーバー登録にサービススクリプトの名前が含まれている必要があります。新しいサーバーを登録するときは、新しいサーバーダイアログボックスの **Advanced** タブの **サービス ID** フィールドにサービス名を含めます。

既存の登録および接続されたサーバーにサービス名を追加する前に、サーバーを切断する必要があります。

1. サーバー名を右クリックし、コンテキストメニューから **Disconnect server** を選択します。
2. サーバー名を右クリックし、コンテキストメニューから **Properties** を選択します。
3. **Advanced** タブを選択し、サービス名を **Service ID** フィールドに追加します。

サービス ID フィールドにより、PEM サーバーはサービスを停止および開始できます。

- Advanced Server 12 サービススクリプトの名前は `edb-as-12` です。
- Advanced Server 11 サービススクリプトの名前は `edb-as-11` です。
- Advanced Server 10 サービススクリプトの名前は `edb-as-10` です。

EDB Audit Configuration プローブの設定

EDB Postgres Advanced Server サーバーの監査ログを設定する前に、EDB Audit Configuration プローブが有効になっていることを確認する必要があります。**Manage Probes** タブを開き、 Manage Probes のステータスを確認するには、ツリーで登録済みの EDB Postgres Advanced Server サーバーの名前を右クリックし、**Management > Manage Probes** を選択します。

EDB Audit Configuration probe Probe Configuration ダイアログボックスの Enabled 列が **Yes** に設定されていることを確認します。 EDB Audit Configuration が有効になっていない場合は、 **Manage Probes** タブの **Enabled?** スイッチを使用して有効にします。

Audit Manager を使用した監査ログの構成

1. Audit Manager ウィザードを開くには、**Management > Audit Manager** を選択します。 Audit Manager - Welcome ダイアログボックスが開きます。**Next** を選択して続行します。
2. Select Servers ツリーを使用して、監査構成を適用するサーバーを指定します。ツリー内のサーバーを利用できるようにするには、PEM による監視のためのサーバーを登録するときに使う、Create Server ダイアログボックスの **Advanced** タブにサービス ID を提供しなければならない。 EDB Postgres Advanced Server のみが監査をサポートしています。PostgreSQL サーバーはツリーに表示されません。

Next を選択して続行します。

1. Auditing Parameters Configuration ダイアログボックスでは、監査を有効または無効にし、ログレコードが PEM に収集される頻度を選択できます。Auditing Parameters Configuration ダイアログボックスを使用して、監査設定を指定します。

   - **Auditing** スイッチを使用して、指定したサーバーで監査を有効または無効にします。
   - **Audit destination** リストを使用して、監査ログの宛先を選択します。 **ファイル** または **Syslog** を選択します。この機能は EDB Postgres Advanced Server 10 以降のリリースでのみサポートされています。
   - **Import logs to PEM** スイッチを使用して、各サーバーから PEM サーバーにログレコードを定期的にインポートします。ログファイルをインポートするには、スイッチを **Yes** に設定します。デフォルトは **No** です。
   - **Collection frequency** リストを使用して、ログ収集が有効になっているときに PEM が監視対象サーバーからログレコードを収集する Collection frequency **を指定します。

   -** Log format **リストを使用して、生のファイルを選択します各サーバーに書き込むログ形式。ログ収集が有効になっている場合、PEM サーバーは CSV 形式を使用します。

   -** File name**フィールドを使用して、ログファイル名を生成するときに使用される形式を指定します。デフォルトでは、形式は `audit-%Y-%m-%d_%H%M%S` に設定されます。ここで

`audit` は監査ディレクトリ名フィールドで指定されたファイル名です `Y` はログが保存された年 `m` はログが保存された月 `d` はログが保存された日 `H` はログが保存された時間 `M` はログが保存された分 `S` はログが保存された秒です b_tran_9 **Change Log Directory for selected servers?** を選択し、 **Audit Directory Name** フィールドを使用して、監査を書き込むディレクトリ名を指定しますログ。ディレクトリは PEM サーバーのデータディレクトリの下にあります。

- **Log directory** ボックスを使用して、ログファイルが保存されるディレクトリに関する情報を指定します。
  - **Change log directory for selected servers?** を移動します **Directory name** フィールドを有効にするには、 **Yes** に切り替えます。
  - **Directory name** フィールドを使用して、監査ログを書き込む各サーバーのディレクトリの名前を指定します。指定されたディレクトリは、サーバー上のデータディレクトリのサブディレクトリとして作成されます。

Next を選択して続行します。

1. Audit log configuration ダイアログボックスは、Auditing Parameters Configuration ダイアログボックスで監査を有効にしている場合にのみ使用できます。Audit Log Configuration ダイアログボックスを使用して、各サーバーに適用するログ構成の詳細を指定します。

   - **Connection attempts** スイッチを使用して、Connection attempts ログに記録するかどうかを指定します。以下を指定します

接続ログを無効にする **None**

All は、すべての接続試行がログに記録されることを示します

Failed は、ログ記録に失敗した接続試行のログ b_tran_1 **Disconnection attempts** スイッチを使用して、Disconnection attempts ログに記録するかどうかを指定します。以下を指定します

切断ログに記録したくない場合は **None**

切断ロギングを有効にするための **All** bb_tran_1 **Log attempts** フィールドを使用して、ログに記録されるステートメントタイプを指定します。以下から選択してください

選択 — SELECT キーワードを含むすべてのステートメントをログに記録します。

Error —エラーになったすべてのステートメントをログに記録します。

DML —すべての DML データ変更言語ステートメントのログを記録します。

DDL —すべての DDL データ定義言語ステートメント、つまりデータを追加、削除、または変更するステートメントをログに記録します。**Select All** を選択して、すべてのステートメントタイプを選択します。

すべてのステートメントタイプをクリアするには、 **Unselect All** を選択します。

- **Audit tag** フィールドを使用して、収集されたログの追跡タグを指定します。監査タグ付け機能は、EDB Postgres Advanced Server のサポートされているバージョンでのみ利用できます。
- **Log rotation** ボックスを使用して、各サーバーでログファイルを管理する方法を指定します。

Enable? スイッチを使用して、ログファイルをローテーションします。新しいログファイルを定期的に使用して、単一のファイルが大きくなりすぎないようにします。

Day リストを使用して、ログファイルをローテーションする日を選択します。

Size (MB) フィールドを使用して、ログファイルをローテーションするサイズをメガバイト単位で指定します。

Time (seconds) フィールドを使用して、ログファイルのローテーション間の秒数を指定します。

Next を選択して続行します。

1. Schedule Auditing Changes ダイアログボックスを使用して、監査構成の変更を有効にする時期を決定します。
    - 監査構成の変更をすぐに実行する場合は、 **Configure locations now?** を選択します。影響を受けるデータベースサーバーが再起動するため、監査変更を有効にできます。
    - **Time?** セレクターを使用して、将来のある時点で監査構成の変更をスケジュールします。リストから目的の日付と時刻を選択します。影響を受けるデータベースサーバーは、指定された日時に再起動して、監査の変更を有効にします。

Finish を選択して、監査構成プロセスを完了します。

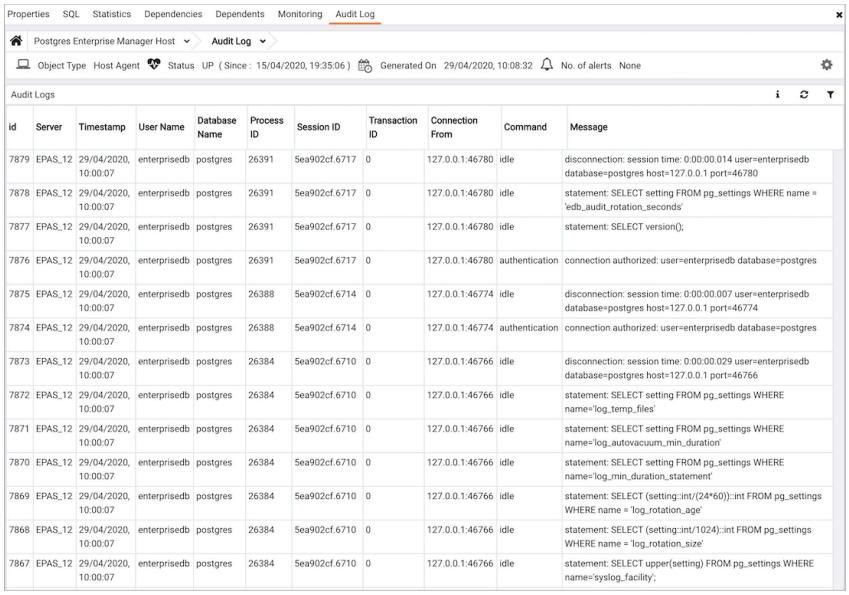
Audit Manager は、各サーバーに構成を適用するジョブをスケジュールします。ジョブは 2 つのタスクで構成されています。1 つはサーバー上の監査ログ構成を更新し、もう 1 つは新しい構成でサーバーを再起動します。

Scheduled Tasks タブを使用して、スケジュールされたジョブのリストを確認できます。**Scheduled Tasks** タブを開くには、サーバーまたはエージェントの名前を選択し、 **Management > Scheduled Tasks** を選択します。

Audit Log ダッシュボードでのログの表示

Audit Log ダッシュボードを使用して、 EDB Postgres Advanced Server データベースインスタンスから audit log を表示します。

Audit Log ダッシュボードを開くには、サーバーまたはエージェントノードを右クリックし、**Dashboards** メニューから **Audit Log Analysis** を選択します。 Audit Log ダッシュボードには、監査レコードが新しい順に表示されます。



id	Server	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
7879	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	idle	disconnection: session time: 0:00:00.014 user=enterprisedb database=postgres host=127.0.0.1 port=46780
7878	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT setting FROM pg_settings WHERE name = 'edb_audit_rotation_seconds'
7877	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT version();
7876	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	5ea902cf.6717	0	127.0.0.1:46780	authentication	connection authorized: user=enterprisedb database=postgres
7875	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	5ea902cf.6714	0	127.0.0.1:46774	idle	disconnection: session time: 0:00:00.007 user=enterprisedb database=postgres host=127.0.0.1 port=46774
7874	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	5ea902cf.6714	0	127.0.0.1:46774	authentication	connection authorized: user=enterprisedb database=postgres
7873	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	disconnection: session time: 0:00:00.029 user=enterprisedb database=postgres host=127.0.0.1 port=46766
7872	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_temp_files'
7871	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_autovacuum_min_duration'
7870	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_min_duration_statement'
7869	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/(24*60))::int FROM pg_settings WHERE name = 'log_rotation_age'
7868	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/1024)::int FROM pg_settings WHERE name = 'log_rotation_size'
7867	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	5ea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT upper(setting) FROM pg_settings WHERE name='syslog_facility';

図 10: The Audit Log dashboard

フィルタリングを使用して、表示される監査レコードの数を制限できます。**Show Filters** を選択してフィルタパネルを公開します。Filters パネルを使用して、表示する監査レコードの選択基準を提供します。

- **Start** フィールドを使用して、レポートの開始日を指定します。
- **End** フィールドを使用して、レポートの終了日を指定します。
- **User** フィールドを使用して、特定の Postgres ユーザーによってアクティビティが開始されたエントリのみを表示します。
- **Server** フィールドを使用して、特定のデータベースでアクティビティが発行されたエントリのみを表示します。
- **Command type** フィールドを使用して、アクティビティが指定されたタイプのエントリのみを表示します。指定できるコマンドタイプは **idle** 、 **authentication** 、および **SELECT** です。ユーザアプリケーションから SQL ステートメントを表示する場合、idle コマンドタイプを指定します。

Filter を選択して、フィルター項目をログ項目に適用します。

5.3.6 Log Manager

PEM Log Manager を使用して、Postgres インスタンスのサーバーログ構成を簡素化できます。ログマネージャーを使用すると、次を含むすべてのサーバーログパラメーターを変更できます。

- ログファイルが書き込まれる場所
- ログファイルが書き込まれる頻度
- ログファイルに書き込まれる情報の種類
- ログファイルエントリのフォーマット
- ログローテーションのプロパティ

Log Manager を使用してサーバーのロギングプロパティを定義する前に、新しいサーバー登録またはプロパティの **Advanced** タブの **Service ID** フィールドに、関連する Advanced Server または PostgreSQL データベースサーバーの名前を指定する必要がありますダイアログボックス。サーバーは、**Service ID** フィールドでサービスの名前を指定した場合にのみ、Server Selection ダイアログボックスの構成に使用できます。

たとえば、Linux ホストにある Advanced Server 9.4 インスタンスのロギング設定を設定しているとします。監視対象サーバーの **Properties** ダイアログボックスの **Advanced** タブの **Service ID** フィールドを `ppas-9.4` に設定します。

注釈:

- Log Manager は、設定とサーバーログ構成プローブに依存して、ウィザードのすべてのフィールドにデータを入力します。したがって、選択したサーバーのこれらのプローブが有効になっていることを確認します。さらに、これらのプローブの実行頻度を最小限に設定して、Log Manager が最新のログ構成を反映するようにします。 - *postgresql.conf* などの構成ファイルのロギングに関連する手動変更を加えた場合、Log Manager を再実行します。 PEM は、これらの変更を自動的に反映しません。

Postgres インスタンスのロギングを構成するには、サーバーを PEM 管理サーバーとして登録し、登録情報にサービススクリプトの名前が含まれている必要があります。

1. Log Manager を開くには、PEM クライアントで、 **Management > Log Manager** を選択します。ウィザードが開き、Welcome メッセージが表示されます。**Next** を選択します。
2. Server Selection ダイアログボックスには、PEM によって監視されているサーバー接続のリストが表示されます。Log Manager ウィザードで指定した構成を適用するサーバーの名前の横にあるチェックボックスをオンにします。Log Manager は、サーバー選択ツリーで名前の左側に赤い感嘆符が表示されているサーバーでは無効になっています。サーバーはいくつかの理由で有効にされない場合があります。
   - Log Manager は、[Properties ダイアログボックスの **Advanced**] タブでサービス ID を指定するサーバーのみを構成できます。サービス ID を提供するには
     1. ツリーでサーバー名を右クリックし、コンテキストメニューから **Disconnect Server** を選択します。
     2. プロンプトが表示されたら、パスワードを入力します。

3. サーバーのコンテキストメニューから、 **Properties** を選択します。

4. **Advanced** タブで、 **Service ID** フィールドにサービスの名前を入力します。

5. **Save** を選択します。

- サーバーにバインドされた PEM エージェントがサーバーを再起動するための十分な権限がない場合、サーバーは無効になります。
- サーバーにバインドされた PEM エージェントが、関連付けられた PEM よりも古いバージョンの場合サーバー、サーバーは無効になっています。

Next を選択します。

3. Log configuration ダイアログボックスを使用して、ログファイルを PEM にインポートする頻度を指定し、ログローテーションの詳細を指定します。

Import Logs ボックスのオプションは、ログファイルを PEM にインポートする頻度を指定します。

- **Import logs to PEM** の横にあるスイッチを使用して、ログファイルを PEM にインポートし、それらを表示するかどうかを指定します the Server Log Analysisashboard.
- **Import Frequency** リストを使用して、ログファイルを PEM にインポートする頻度を指定します。

Log rotation configuration ボックスを使用して、ログファイルの最大長ライフスパンまたはサイズを指定します。

- **Rotation Size** フィールドを使用して、ログファイルの最大サイズをメガバイト単位で指定します。デフォルト値は 10MB です。ログファイルの最大サイズに制限を設けたくない場合は、0 に設定します。
- **Rotation Time** フィールドを使用して、各ログファイルに保存する全体の日数を指定します。デフォルト値は 1 日です。

Truncation on Rotation スイッチを使用して、時間ベースのログファイルローテーションのサーバー動作を指定します。

- **On** を選択して、新しいファイルと同じ名前の既存のログファイルを上書きします。
- **Off を選択します**新しいログファイルと同じ名前の既存のログファイルに、新しいログファイルエントリを追加します。これはデフォルトの動作です。

Next を選択します。

4. Where to Log ダイアログボックスを使用して、ログファイルを書き込む場所を指定します。

   - **Log Destination** ボックスからオプションを選択して、サーバーログ出力の宛先を指定します。

     – **stderr** スイッチを次のように設定します。コンマ区切り値形式でログファイルを書き込むには、**csvlog** スイッチを **Yes** に設定します。このオプションは、[Schedule] ダイアログボックスで **Import logs to PEM** を選択した場合、有効になり編集できません。サーバーログファイルを PEM にインポートしない場合は、このオプションを編集できます。

     – ログファイルをシステムログファイルに書き込むには、 **syslog** スイッチを **Yes** に設定します。

     – Windows では、**を設定しますログファイルをイベントログに書き込むには、**eventlog** スイッチを **Yes** にします。

- **Log collection** ボックスを使用して、収集設定を指定します。
  - **Log Collector** スイッチを **Enable に設定します**キャプチャされたログメッセージ stderr にリダイレクトされます。
  - **Log Silent Mode** スイッチを **Enable** に設定して、制御端末からの関連付けを解除して、バックグラウンドでサーバーをサイレント実行します。
- **Log Directory** ボックスを使用して、ログファイルの場所設定を指定します。
  - ログファイルの各セットを別のディレクトリに維持するには、**Change log directory for selected servers?** スイッチを **Yes** に設定します。
  - 使用 **ディレクトリ名**フィールドを使用して、ログファイルを書き込むディレクトリを指定します。ディレクトリは、監視対象サーバーのインストールディレクトリの下の `pg_log` ディレクトリの下にあります。
- **Log File Name** フィールドを使用して、ログファイル名の形式を指定します。**デフォルト**に設定されている場合、フォーマットは `enterprisedb-%Y-%m-%d_%H%M%S` であり、ここで
  - `enterprisedb` はファイル名接頭辞です b_tran_31 `Y` はログが保存された年です b_tran_31 `m` はログが保存された月です b_tran_31 `d` はログが保存された日です b_31 `H` はログが保存された時間です
  - `M` はログが保存された分です
  - `S` はログが保存された秒です b_tran_29 syslog へのロギングが有効な場合
  - **Syslog Facility** リストを使用して、syslog 機能を指定します use.
  - **Syslog Ident** フィールドを使用して、システムログの EDB Postgres Advanced Server エントリを識別するプログラム名を指定します。

Next を選択します。

5. When to Log ダイアログボックスを使用して、ログファイルエントリを開始するイベントを指定します。
   - 重大度の順に、重大度レベルは最も重大度の高いものから最も低いものは次のとおりです。
     - **panic** —すべてのデータベースセッションを中止するエラー。
     - **fatal** —セッションを中断する原因になったエラー。
     - **log** —管理者が関心を持つ情報メッセージ.
     - **error** —コマンドを中止するエラー。
     - **警告**—コマンドは完了しましたが、予想通りに実行されないエラー状態。
     - **notice** —ユーザーの関心のある項目。これはデフォルトです。
     - **info** — user.
     - **debug5** 〜 **debug1** によって暗黙的に要求された情報—開発者に役立つ詳細なデバッグ情報。
   - **Client min attempts** リストを使用して、クライアントアプリケーションに送信するメッセージの最も低い重大度レベルを指定します。

- **Log min attempts** リストを使用して、サーバーログに書き込む最も低い重大度レベルを指定します。
- デフォルトでは、エラーメッセージがサーバーログに書き込まれるときに、ログエントリを開始した SQL ステートメントのテキストは含まれません。**Log min error Statement** リストを使用して、SQL ステートメントロギングをトリガーする重大度レベルを指定します。メッセージが指定された重大度以上の場合、メッセージを生成した SQL ステートメントはサーバー log.
- に書き込まれます。指定されたミリ秒数を超えるステートメントはサーバーログに書き込まれます。値**-1** は、すべての期間ベースのロギングを無効にします。値 **0** は、すべてのステートメントとその期間をログに記録します。
- **Log temp files** フィールドを使用して、ファイルサイズをキロバイト単位で指定します。一時ファイルが指定されたサイズに達すると、ログに記録されます。この機能を無効にするには、**-1** デフォルトの値を指定します。
- **Log autoVacuum minduration** フィールドを使用して、ミリ秒単位の時間長を指定します。自動バキュームが指定された時間を超えると、アクティビティがログに記録されます。この機能を無効にするには、値**-1** デフォルトを指定します。

Next を選択します。

6. What to Log ダイアログボックスを使用して、デバッグと監査に役立つログエントリオプションを指定します。

Debug options ボックスのスイッチを使用して、クエリの実行に関連するログファイルに、開発者が関心を持つ可能性のある情報を含めます。

- ログファイルの構文解析ツリー。
- ログファイルにクエリリライター出力を含めるには、**Rewriter Output** スイッチを **Yes** に設定します。
- **Execution plan** スイッチを **Yes** に設定します実行された各クエリの実行プランをログファイルに含めます。

Indent Debug Options Output in Log スイッチが **Yes** に設定されている場合、サーバーはパースツリーエントリ、クエリリライターエントリ、またはクエリ実行プランエントリを含む各行をインデントします。インデントにより、生成されるログファイルが読みやすくなりますが、ログファイルが長くなります。

General Options ボックスのスイッチを使用して、ログファイルに監査情報を含めます。

- サーバーログにチェックポイントとリスタートポイントを含めるには、**Checkpoints** スイッチを **Yes** に設定します。
- **を設定します** Connections** スイッチを **Yes** に設定して、サーバーへの接続が試行された各接続と、サーバーログに正常に認証された接続を含めます。
- サーバーログを含めるには、**Disconnections** スイッチを **Yes** に設定しますセッション情報とセッション期間を提供する、終了したセッションのエントリー。
- **Duration** スイッチを **Yes** に設定して、サーバーにログに記録された各ステートメントの実行に必要な時間を含めます。log.
- **を設定します** Hostname** は **Yes** に切り替えて、各サーバーログエントリに IP アドレスとホスト名の両方を含めます。デフォルトでは、IP アドレスのみがログに記録されます。これにより、パフォー

マンスのペナルティが発生する可能性があります。

- ロックを取得するために deadlock_timeout パラメーターで指定された時間より長く待機するセッションのログエントリを書き込むには、**Lock Waits** スイッチを **Yes** に設定します。これは、ロック待機がパフォーマンスの低下の原因であるかどうかを判断する場合に役立ちます。

Error verbosity リストを使用して、サーバーログの各エントリに書き込まれる詳細を指定します。

- **default** を選択して、各サーバーログエントリにエラーメッセージ、詳細、ヒント、クエリ、およびコンテキストを含めます。
- エラーメッセージのみをログに記録するには **terse** を選択します。
- エラーメッセージ、DETAIL、HINT、QUERY、および CONTEXT エラー情報、SQLSTATE エラーコードとソースコードファイル名、ファンクション名を含めるには **verbose** を選択します、およびエラーを生成した行番号。

Prefix string フィールドを使用して、各ログファイルエントリの先頭に書き込まれる printf スタイルの文字列を指定します。サポートされるオプションについては、

Postgres core documentation の log_line_prefix ドキュメントを参照してください。

Statements リストを使用して、サーバーログに含める SQL ステートメントを指定します。

- SQL ステートメントのロギングを無効にするには **none** デフォルトを指定します。
- ログ ddl に指示するには **ddl** を指定します CREATE、ALTER、DROP などのデータ定義言語ステートメント。
- すべての ddl ステートメントと、INSERT、UPDATE、DELETE、TRUNCATE などのすべての dml データ変更言語ステートメントをログに記録するには **mod** を指定します。と COPY FROM.
- すべての SQL ステートメントをログに記録するには **all** を指定します。

Next を選択します。

7. Schedule Logging Changes ダイアログボックスを使用して、ロギングが構成変更を適用する時期を指定します。
    - **Configure locations now** スイッチを **Yes** に設定して、構成設定を有効にします。ログマネージャーウィザード.
    - が完了すると、サーバーが再起動します。設定とサーバーを再起動します。

ログマネージャーウィザードで指定された構成の変更を適用すると、サーバーが再起動し、ユーザーのデータベースサーバーの使用を一時的に中断します。

8. **Finish** を選択してウィザードを終了します。サーバーを再起動するか、スケジュールダイアログボックスで指定された時間にサーバーの再起動をスケジュールします。

Server Log Analysis ダッシュボードの確認

ログマネージャーウィザードを起動し、ログファイルを PEM にインポートした後、Server Log Analysis ダッシュボードを使用して、選択したサーバーのログファイルを確認できます。Server Log Analysis ダッシュボードを開くには、PEM クライアントツリーで監視対象サーバーの名前を右クリックし、**Dashboards > Server Log Analysis** を選択します。

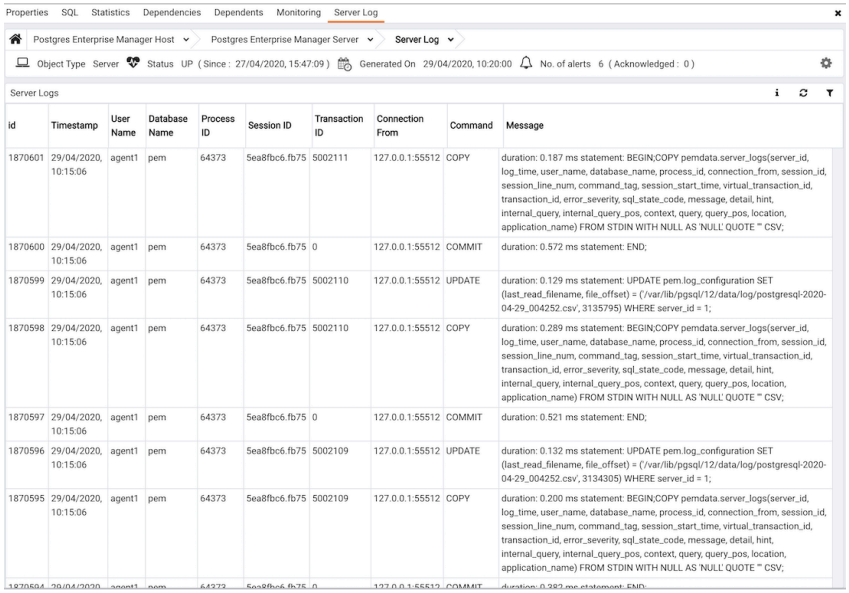


図 11: The Server Log Analysis dashboard

Server Log Analysis ダッシュボードのヘッダー情報には、サーバーが起動された日時、ページが最後に更新された日時、およびトリガーされたアラートの現在の数が表示されます。

Server Log テーブルのエントリは時系列で表示され、最新のログエントリが最初に表示されます。スクロールバーを使用して、ログエントリを移動したり、表示されていない列を表示したりします。

サーバーログテーブルの上部の見出しは、各列に格納されている情報を示します。列見出しにカーソルを合わせると、各列のコンテンツの説明を含むツールチップが表示されます。

フィルタリングを使用して、表示されるサーバーログレコードの数を制限できます。**Show Filters** を選択してフィルタパネルを開き、フィルタを定義します。**フィルタ定義**ボックスを使用して、表示するレポートのサブセットを選択するための選択基準を説明します。

- **From** フィールドを使用して、表示されるサーバーログの開始日を指定します。
- **To** フィールドを使用して、表示されるサーバーログの終了日を指定します。
- **Username** フィールドにロール名を入力して、そのユーザーのみが実行したトランザクションを表示します。
- **Server** フィールドにデータベース名を入力して、表示されるレコードを指定したデータベースに対して実行されたトランザクションに制限します。
- **Command Type** フィールドを使用して、フィルタリングされたレポートに表示されるコマンドの選択基準を指定します。

サーバーログをフィルタリングするための基準を指定した後、 **Filter** を選択して、フィルタリングされたサーバーログを Server Log テーブルに表示します。

Postgres Log Analysis Expert

PEM Log Analysis Expert は、Postgres Enterprise Manager に登録されているサーバーのログファイルを分析し、ログファイルエントリに基づいて Postgres クラスターの使用状況の分析を生成します。Log Analysis Expert レポートの情報を使用して、クラスターの使用率と構成を最適化してパフォーマンスを向上させることに関する決定を下すことができます。

PEM Log Analysis Expert を使用する前に、Server Properties ダイアログボックスの **Advanced** タブで **Service ID** 値を指定する必要があります。また、ログマネージャーウィザードを使用して、PEM サーバーによるログ収集を有効にする必要があります。

1. Postgres Log Analysis Expert ウィザードを開くには、PEM クライアントで、**Management > Postgres Log Analysis Expert** を選択します。ウィザードの [Welcome] 画面で、[**Next**] を選択します。

ウィザードの Analyzer selection ダイアログボックスには、選択可能なアナライザーのリストが表示されます。各アナライザーは、ログファイルから収集された情報を含む対応するテーブル、チャート、またはグラフを生成します。

2. Log Analysis Expert で指定するアナライザーの横にあるチェックボックスを選択して、対応するテーブル、チャート、またはグラフを準備します。選択したら、[**Next**] を選択します。
3. サーバー選択ツリーを使用して、Postgres Log Analysis Expert で分析するサーバーを指定します。複数のサーバーを選択した場合、結果のレポートには、サーバーごとに対応する結果セットが個別のただし連続したリストが含まれます。**Next** を選択して、[レポートオプション] ダイアログボックスに進みます。
4. **Options** セクションを使用して、分析方法と結果のテーブルの最大長を指定します。

- **Aggregate Method** リストを使用して、Log Analysis Expert が選択したデータを統合するために使用する方法を選択します time interval:
  - **SUM** は、指定された時間 span.
  - **AVG** は、指定された時間 span.
  - に収集された値の合計である値を計算します。
  - **MAX** は、指定された時間スパンで発生する最大値を使用します。
  - **MIN** は、指定された時間スパンで発生する最小値を使用します。
- **Timespan** フィールドを使用して、スパン数を指定しますアナライザーがグラフ上のポイントの各計算に組み込む分。たとえば、時間スパンが 5 分で、集計方法が **AVG** の場合、指定されたグラフの各ポイントには、5 分のタイムスパンで発生したアクティビティの平均値が含まれます。
- **を使用します Rowslimit** フィールドを使用して、テーブルに含める最大行数を指定します。

5. **Time Intervals** セクションを使用して、Log Analysis Expert が分析する時間範囲を指定します。

- **Relative days** を **Yes** に設定して、 **(+/-) From date を有効にします。** フィールドを使用し、 **From** フィールドで選択した日時の前後の日数を指定します。

- **From** フィールドを使用して、分析の開始日時を指定します。
- **を使用します分析の終了日時を指定する** To** フィールド。
- **(+/-) From date** セレクターを使用して、 **From** 日付の前後の日数を指定します分析。

レポートオプションを指定したら、[**Next**] を選択して [レポートの宛先] ダイアログボックスに進みます。

6. デフォルトオプションを選択して [**Finish**] を選択すると、PEM クライアントのタブ付きブラウザーで Log Analysis Expert レポートを表示できます。または、 **レポートのダウンロード**を選択して、レポートのコピーを HTML ファイルに保存して後で使用できます。

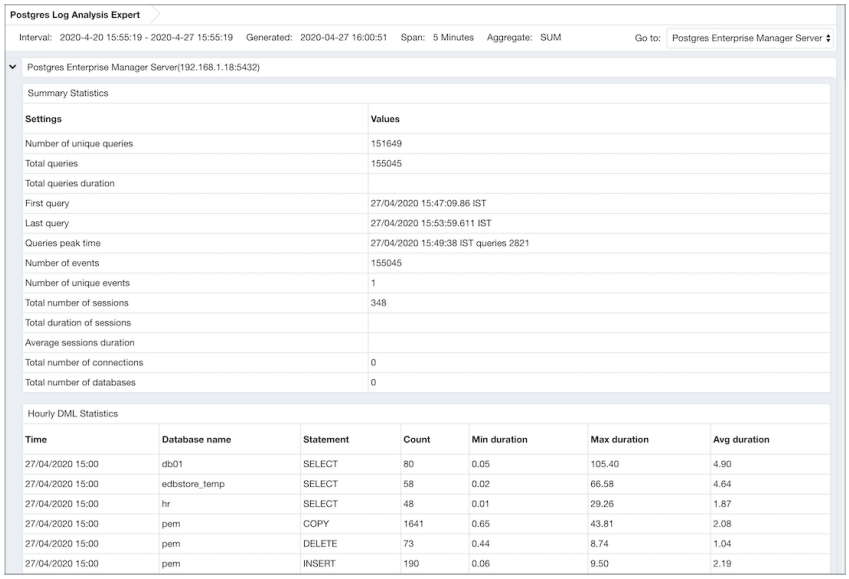
Postgres Log Analysis Expert レポートの確認

レポートをすぐに確認することを選択した場合、Postgres Log Analysis Expert レポートが PEM Client ウィンドウに表示されます。レポートヘッダーには次のものが表示されます。

- レポートが生成された日時
- レポートの期間
- レポートを定義するときに指定された集計方法

情報が表示されるサーバーの名前は、レポートの各セクションの先頭に表示されます。

レポートには、Log Analysis Expert ウィザードで選択したテーブル、グラフ、およびチャートが表示されます。画面の右下隅にある **Jump To** を選択して、特定のグラフィックに移動します。



Postgres Log Analysis Expert

Interval: 2020-4-20 15:55:19 - 2020-4-27 15:55:19 Generated: 2020-04-27 16:00:51 Span: 5 Minutes Aggregate: SUM Go to: Postgres Enterprise Manager Server

Postgres Enterprise Manager Server(192.168.1.18:5432)

Summary Statistics

Settings	Values
Number of unique queries	151649
Total queries	155045
Total queries duration	
First query	27/04/2020 15:47:09.86 IST
Last query	27/04/2020 15:53:59.611 IST
Queries peak time	27/04/2020 15:49:38 IST queries 2821
Number of events	155045
Number of unique events	1
Total number of sessions	348
Total duration of sessions	
Average sessions duration	
Total number of connections	0
Total number of databases	0

Hourly DML Statistics

Time	Database name	Statement	Count	Min duration	Max duration	Avg duration
27/04/2020 15:00	db01	SELECT	80	0.05	105.40	4.90
27/04/2020 15:00	edbstore_temp	SELECT	58	0.02	66.58	4.64
27/04/2020 15:00	hr	SELECT	48	0.01	29.26	1.87
27/04/2020 15:00	pem	COPY	1641	0.65	43.81	2.08
27/04/2020 15:00	pem	DELETE	73	0.44	8.74	1.04
27/04/2020 15:00	pem	INSERT	190	0.06	9.50	2.19

図 12: The Postgres Log Analysis Expert Report

レポートに複数の監視対象サーバーの分析が含まれる場合、グラフと表がセットで表示されます。最初に、1 つのサーバーの統計を表示するグラフ、テーブル、およびチャートが表示されます。次に、レポート内の次のサーバーのグラフィックが表示されます。

概要統計テーブル

Summary Statistics テーブルには、選択したサーバーのサーバーアクティビティの概要が表示されます。

Row name	Description
Number of unique queries	指定された期間に選択されたサーバーに対して行われた一意のクエリーの数。
Total queries	指定された期間に選択されたサーバーに対して行われたクエリーの数。
Total queries duration	サーバーに対するクエリーの実行に使用された時間。
First query	指定された期間内の、最初にクエリーがサーバーに対して実行された時間。
Last query	指定された期間内の、最後のクエリがサーバーに対して実行された時間。
Queries peak time	指定された期間内のポイント、そのクエリアクティビティは最高レベルに到達しました。
Number of events	指定された期間内のログイベントの数。
Number of unique events	一意のサーバーイベントの数。
Total number of sessions	期間内に記録されたセッションの数。
Total duration of sessions	指定された期間にセッションが接続された時間。
Average sessions duration	各セッションの平均長さ。
Total number of connections	サーバーに対して行われたユーザー接続の数。
Total number of databases	選択したサーバー上のデータベースの数。

時間ごとの統計テーブル

Hourly DML Statistics テーブルには、1 時間以内のさまざまな DML コマンド `SELECT` 、`INSERT` 、`UPDATE` 、`DELETE` 、`COPY` 、および `FETCH` の使用に関連する統計が表示されます。このテーブルの Min Duration(sec)、Max Duration(sec)、および Avg Duration(sec) 列の値を生成するには、`log_min_duration_statement` 構成パラメーターに 0 以上の値を指定する必要があります。選択したエディタで `postgresql.conf` ファイルを変更するか、Log Manager ウィザードの **Log Minduration Statement** フィールドで `0` 以上の値を指定することにより、パラメーターを設定できます。

Column name	Description
Time	データが分析された 1 時間期間の開始。
Database	指定された DML コマンドが実行されたデータベースの名前。
Command Type	DML コマンドタイプ。
Total Count	レポートによって分析された 1 時間の間に、指定されたコマンドタイプのコマンドが実行された回数。
Min Duration(sec)	サーバーが指定されたコマンドタイプに応答するために使用された最短時間秒単位。

次のページに続く

表 25 – 前のページからの続き

Column name	Description
Max Duration(sec)	サーバーが指定されたコマンドタイプに応答するために費やした最長時間秒単位。
Avg Duration(sec)	指定されたコマンドタイプに応答するときにサーバーが使用した平均時間秒単位。

DML 統計タイムラインセクション

Log Analysis Expert レポートの DML Statistics Timeline セクションには、DML ステートメントの使用状況に関する情報が表示されます。

Graphic	Description
Line graph	選択した期間中のステートメントの使用状況の分析。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。
Pie chart	選択した期間中の各 DML ステートメントタイプのステートメント使用率。

DDL 統計タイムラインセクション

Log Analysis Expert レポートの DDL Statistics Timeline セクションには、DDL ステートメントの使用状況に関する情報が表示されます。

Graphic	Description
Line graph	選択した期間中のステートメントの使用状況の分析。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。
Pie chart	選択した期間中の各 DDL ステートメントタイプのステートメント使用の割合。

コミットとロールバックの統計タイムラインセクション

Log Analysis Expert レポートの Commit and Rollback Statistics Timeline セクションには、指定された期間にログに記録された `COMMIT` 、`ROLLBACK` 、および `SAVEPOINT` ステートメントに関する情報が表示されます。

Graphic	Description
Line graph	指定された期間中のコミットおよびロールバックアクティビティの分析。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。

次のページに続く

表 28 – 前のページからの続き

Graphic	Description
Pie chart	指定された期間に実行された `COMMIT`、*SAVEPOINT*、または `ROLLBACK`ステートメントの比較パーセント。

チェックポイント統計タイムラインセクション

Log Analysis Expert レポートの Checkpoint Statistics Timeline セクションには、指定された期間にログに記録されたチェックポイント操作に関する情報が表示されます。

Graphic	Description
Line graph	指定された期間中のチェックポイント操作アクティビティの分析。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。
Pie chart	指定された期間にログに記録されたさまざまな種類のチェックポイントアクティビティの比較パーセント。

ログイベント統計テーブル

Log Event Statistics テーブルには、重大度レベル WARNING、ERROR、FATAL、PANIC、HINT、または CONTEXT のログエントリがリストされています。エラーメッセージのロギングの詳細レベルは、`log_min_error_statement` パラメーターによって制御されます。選択したエディタで `postgresql.conf` ファイルを変更するか、Log Manager ウィザードの **Log Min Error Statement** フィールドに値を指定することにより、パラメーターを設定できます。

Column name	Description
Error Severity	ログエントリの重大度レベル。
Message	ログメッセージ。
Total Count	ログエントリが発生した回数。

ログ統計テーブル

Log Statistics テーブルには、LOG、DETAIL、DEBUG、NOTICE、INFO、または STATEMENT の操作重大度レベルを示すログエントリがリストされています。情報メッセージのロギングの詳細レベルは、`log_min_messages` パラメーターによって制御されます。選択したエディタで `postgresql.conf` ファイルを変更するか、Log Manager ウィザードの **Log Min Messages** フィールドに値を指定することにより、パラメーターを設定できます。

Column name	Description
Error Severity	ログエントリの重大度レベル。
Total Count	ログエントリが発生した回数。

Temp Generated Query テーブル

Temp Generated Queries テーブルには、一時ファイルを作成したクエリのリストが表示されます。

Column name	Description
Log Time	ログエントリが生成された時刻。
TempFile Size(Bytes)	一時ファイルのバイト単位のサイズ。
Query	一時ファイルを作成したクエリのテキスト。

一時ファイル統計タイムライングラフ

Temp File Statistics Timeline グラフには、指定された期間の一時ファイルのサイズが表示されます。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。

ロック統計タイムラインセクション

Log Analysis Expert レポートの Lock Statistics Timeline セクションには、指定された期間に保持されたロックに関する情報が表示されます。

Graphic	Description
Graph	期間中の任意の時点で保持されたロックの数。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。
Pie chart	選択した期間に使用された各タイプのロックの相対的な割合が表示されます。

待機統計タイムラインセクション

Log Analysis Expert レポートの Waiting Statistics Timeline セクションには、指定された期間にロックを待機している DML ステートメントに関する情報が表示されます。

Graphic	Description
Graph	期間中の任意の時点で待機している DML ステートメントの数。色付きの各線は、ステートメントタイプを表します。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。

次のページに続く

表 34 – 前のページからの続き

Graphic	Description
Pie chart	選択した期間にロックを待機した各タイプの DML ステートメントの相対パーセンテージ。

Idle Statistics Timeline セクション

Log Analysis Expert レポートの Idle Statistics Timeline セクションには、サーバーへの接続がアイドル状態である時間に関する情報が表示されます。`IDLE` サーバーは、クライアントからの接続を待機しています。`IDLE in transaction` である接続は、トランザクションを開始しましたが、トランザクションをまだコミットまたはロールバックしておらず、クライアントからのコマンドを待機しています。`IDLE in transaction (aborted)` であるセッションは、トランザクションを開始しましたが、トランザクションをまだコミットまたはロールバックしておらず、クライアントからのコマンドを待機しています。トランザクションでエラーが発生しました。トランザクションはロールバックのみです。

Graphic	Description
Graph	サーバーが`IDLE`、*IDLE in transaction*、および`IDLE in transaction (aborted)`である時間。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。
Pie chart	選択した期間に使用された各種類のロックの相対的な割合。

自動バキューム統計テーブル

Autovacuum Statistics テーブルには、監視対象サーバーでの自動バキュームアクティビティに関する統計が表示されます。

Column name	Description
Log Time	自動バキュームアクティビティがログに書き込まれた時間。
Relation	自動バキュームが実行されたテーブルの名前。
Index Details	実行されたインデックススキャンの数。
Page Details	削除されたページ数と残ったページ数。
Tuple Details	削除されたタプルの数と残っているタプルの数。
Buffer Usage	バッファヒット、ミス、またはダーティなバッファの数。
Read Rate	1 秒あたりの MB 単位の平均読み取り速度。
System Usage	自動バキュームアクティビティの実行に使用された CPU 時間の割合。

自動分析統計テーブル

Autoanalyze Statistics テーブルには、ログに記録された自動分析アクティビティが表示されます。

Column name	Description
Log Time	自動分析アクティビティがログに書き込まれた時間。
Relation	自動分析が実行されたテーブルの名前。
System Usage	自動分析アクティビティの実行に使用された CPU 時間の割合。

スロークエリー統計テーブル

Slow Query Statistics テーブルには、監視対象サーバーで実行された最も遅いクエリーが表示されます。テーブルには、Log Analysis Expert の Rows Limit フィールドで指定されたエントリの数が含まれています。

Column name	Description
Log Time	クエリアクティビティがログに書き込まれた時間。
Tag	コマンドタイプ。
Query	実行されたクエリーのテキスト。
Parameters	クエリがパラメーター化されたクエリの場合のパラメーター。
Duration	サーバーがクエリーの実行に要した時間。
Host	クエリーが実行されたホストの名前。
Database	クエリが実行されたデータベースの名前。

頻繁に実行されるクエリ統計テーブル

Frequently Executed Query Statistics テーブルには、最も頻繁に実行されたクエリステートメントが表示されます。テーブルには、Log Analysis Expert の **Rows Limit** フィールドで指定されたエントリの数が含まれています。

Column name	Description
Query	実行されたクエリーのテキスト。
Parameters	クエリがパラメーター化されたクエリの場合のパラメーター。
No. of Times Executed	クエリーが実行された回数。
Total Duration	サーバーがクエリーの実行に要した時間。

ほとんどの時間実行されたクエリ統計テーブル

Most Time Executed Query Statistics テーブルには、サーバーで最も実行時間を要したクエリが表示されます。テーブルには、Log Analysis Expert の **Rows Limit** フィールドで指定されたエントリの数が含まれています。

Column name	Description
Query	実行されたクエリーのテキスト。
Parameters	クエリがパラメーター化されたクエリの場合のパラメーター。
No. of Times Executed	クエリーが実行された回数。
Total Duration	サーバーがクエリーの実行に要した時間。

接続の概要 タイムラインセクション

Log Analysis Expert レポートの Connections Overview Timeline セクションには、指定された期間中の成功および失敗した接続試行に関する情報が表示されます。

Graphic	Description
Timestamp graph	指定された期間中の任意の時点で試行されたサーバー接続と認証された接続の数。特定のポイントにカーソルを合わせると、グラフ上のそのポイントに関する詳細情報が表示されます。
Summary pie chart	指定された期間に試行された接続と認証された接続の相対的な割合。

5.3.7 Charts

Manage Charts タブを使用して、カスタム折れ線グラフまたは表を作成または変更したり、カスタムチャートで使用する Capacity Manager テンプレートをインポートしたりできます。チャートを定義したら、カスタムダッシュボードにチャートを表示できます。

Manage Charts タブを開くには、PEM クライアントで、 **Management > Manage Charts** を選択します。[**Manage Charts**] タブは、次のことを行うダイアログボックスにアクセスするためのクイックリンクメニューを提供します。

- カスタムダッシュボードで使用する新しいチャートを作成します。
- Capacity Manager テンプレートをインポートして、カスタムチャートを作成するためのテンプレートとして使用します。
- オンラインヘルプにアクセスします。

Custom Charts テーブルには、定義したチャートのリストが表示されます。チャートが新しい場合、フォントは緑色になります。グラフを追加するか、画面を更新すると、フォントが黒になります。Custom Charts テーブルの右上にある検索ボックスを使用して、カスタムチャートを検索します。以下を指定します

- チャート名
- 種類
- レベル
- メトリックカテゴリ

Custom Charts テーブルのチャート名の左側にあるアイコンを使用して、チャートを管理します。

- **Edit** を選択して Chart Configuration ウィザードを開き、チャートまたはテーブルの側面を変更します。
- 選択したチャートを削除するには、 **Delete** を選択します。

カスタムチャートの作成

[**Manage Charts**] タブの [Quick Links] セクションで [**Create New Chart**] を選択して、Create Chart ウィザードを開きます。ウィザードは、新しいチャートを定義するために必要な手順をガイドします。

1. Configure Chart ダイアログボックスを使用して、チャートに関する一般情報を指定します。
    - **Name** フィールドにチャートの名前を指定します。
    - **Category** フィールドのリストを使用して、次のカテゴリを指定します。このチャートを表示するにはカスタムグラフをカスタムダッシュボードに追加する場合、指定したカテゴリからグラフを選択できます。
    - **Type** フィールドを使用して、グラフが折れ線グラフかテーブルかどうかを指定します。
    - でチャートの説明を提供します。 **Description** フィールド。説明は、ユーザーがチャートの横にある情報アイコンを選択すると、カスタムダッシュボードに表示されます。

[Configure Chart] ダイアログボックスのフィールドに入力したら、[**Next**] を選択します。

1. Select Metrics ダイアログボックスを使用して、チャートに表示するメトリックを選択します。
    - **メトリックレベル**リストを使用して、メトリックを選択する PEM 階層のレベルを指定します。**エージェント** 、 **データベース** 、または **サーバー**を指定できます。各レベルは、プローブとメトリックの一意のセットへのアクセスを提供します。
    - **Available metrics** ボックスのツリーを使用して、チャートに表示するメトリックを選択します。

テーブルを作成する場合、1 つのプローブからのみメトリックを選択できます。ツリーの各ノードは、単一のプローブによって返されたメトリックをリストします。ツリーのノードを展開し、メトリック名の横にあるチェックボックスを選択して、そのメトリックデータをテーブルに含めます。

折れ線グラフを作成する場合は、ツリーのノードを展開し、チャートに含める各メトリックをダブルクリックします。

- Selected Metrics パネルを使用して、チャートにメトリックデータを表示する方法を指定します。選択パネルの変更不可の Metric [Probe] 列にメトリックの名前が表示されます。次のことができます。

  - ゴミ箱を選択して、選択したメトリックのリストからメトリックを削除します。
  - Selection Criteria 列のリストを使用して、表示されるデータの順序を指定します。
  - **Limit** フィールドを使用して数を指定するテーブルの行またはチャート内の行数最大 32 行 100 行。
- 折れ線グラフを作成している場合、PEM は階層間のメトリックの比較をサポートします。
  - **Compare** を選択して 1 つ以上を選択しますチャートで比較するプローブ固有の属性 CPU、インターフェイス、データベースなど。
  - **Copy** を選択して、同じプローブのすべてのメトリックに選択を適用します。

[Select Metrics] ダイアログボックスのフィールドに入力したら、[**Next**] を選択します。

1. Set Options ダイアログボックスを使用して、チャートの表示オプションを指定します。
   - **Auto Refresh** フィールドを使用して、チャートの更新の間分数を指定します。1〜120 の値を選択してください。デフォルトの自動リフレッシュレートは 2 分です。
   - **Line chart options** の下のフィールドを使用して、折れ線グラフの表示設定を指定します。
     - **プロットするポイント**フィールドを使用して指定しますチャートにプロットするポイントの最大数。
     - **履歴スパン**の右側のフィールドを使用して、チャートに日、時間、および分単位で表示する履歴データの量を指定します。
   - のフィールドを使用する **Data extrapolation** ボックスは、履歴データに基づいて PEM が推定データを生成するかどうかを指定します。
     - チャートから推定データを省略するには、 **No Extrapolation** を選択します。
     - **Days** を使用するには **Span** を選択します
     - しきい値セレクターを使用してチャートの最大値または最小値を指定するには、**Threshold** を選択します。

[Set Options] ダイアログボックスのフィールドに入力したら、[**Next**] を選択します。

1. Set Permissions ダイアログボックスを使用して、チャートの表示オプションを指定します。
   - 許可されたすべてのユーザーがチャートを利用できるようにするには、**Share with all** スライダーを **Yes** に設定します。**Accessmissions** フィールドで指定されたユーザーまたはグループへのアクセスを制限するには、**No** を選択します。
   - **Access attempts** フィールドを使用して、チャートへのアクセスを付与するグループを選択します。

チャートの定義が終了したら、 **Finish** を選択して編集を保存し、**Manage Charts** タブのリストにチャートを追加します。

Capacity Manager テンプレートのインポート

Manage Charts タブの Quick Links セクションで **Import Capacity Manager Template** を選択して Create Chart ダイアログボックスを開き、Capacity Manager テンプレートをチャートまたはテーブルの開始点として使用します。

1. [Create Chart] ダイアログボックスが開いたら、カスタムチャートに関する情報を提供します。

    - **Import Capacity Template** リストを使用して、チャートのベースとなるテンプレートの名前を選択します。
    - チャートの名前を指定します **Name** フィールド.
    - **Category** リストを使用して、このチャートを表示するカテゴリを指定します。カスタムダッシュボードにカスタムチャートを追加する場合、このカテゴリーからチャートを選択できます。
    - **Type** フィールドを使用して、チャートが折れ線グラフまたはテーブルであることを指定します。
    - でチャートの説明を提供します。 **Description** フィールド。説明は、ユーザーがチャートの横にある情報アイコンを選択すると、カスタムダッシュボードに表示されます。

Next を選択します。

1. Select Metrics ウィンドウでは、選択したテンプレートで指定されたメトリックを確認できます。チャートエディターの下部パネルには、チャートに含めるメトリックが表示されます。チャートエディターを使用して、チャートに含まれるメトリックを変更することはできません。メトリックを変更するには、Capacity Manager ユーティリティを使用してテンプレートを更新します。

メトリックを確認したら、[**Next**] を選択します。

1. Set Options ウィンドウを使用して、チャートの表示オプションを指定します。

    - **Auto Refresh** フィールドを使用して、チャートの更新の間分数を指定します。1〜120 の値を選択してください。デフォルトの自動リフレッシュレートは 2 分です。
    - **Data extrapolation** ボックスを使用して、チャートの対象とする期間を指定します。次のいずれかができます。
    - **履歴日と外挿日** を選択し、以下を提供します。
        - **Historical** フィールドでグラフ化する履歴データの日数。
        - **Extrapolated でグラフ化する予測日数** field.
    - **履歴の日数としきい値** を選択して、以下を提供します。
        - **Historical** フィールドでチャートする履歴データの日数。
        - チャートが終了するしきい値。

[Set Options] ウィンドウの完了後、[**Next**] を選択します。

1. Set Permissions ウィンドウを使用して、チャートの表示オプションを指定します。

- 許可されたすべてのユーザーがチャートを利用できるようにするには、**Share with all slider** を **Yes** に設定します。**Accessmissions** フィールドで指定されたユーザーまたはグループへのアクセスを制限

するには、**No** に設定します。 - **Access attempts** フィールドを使用して、チャートへのアクセスを付与するグループを選択します。

チャートの定義が終了したら、 **Finish** を選択して編集を保存し、**Manage Charts** タブのリストにチャートを追加します。

5.3.8 Dashboards

PEM は、多数のダッシュボードを介してパフォーマンス統計を表示します。各ダッシュボードには、選択したオブジェクトに関連する統計を表示するチャート、グラフ、表を含む一連の概要ビューが含まれています。

ダッシュボードの概要

PEM クライアントは、PEM サーバーに接続すると、Global Overview ダッシュボードを表示します。追加のダッシュボードは、監視対象オブジェクトに関する統計情報を提供します。

UI の詳細についてのアプリ内ヘルプへのアクセス

アプリ内ヘルプは、メニュー項目、ダッシュボードのチャートとグラフ、ダイアログボックスのオプションを含む PEM Web インターフェイスに関する包括的な詳細を提供します。アプリ内のヘルプには、データベースオブジェクトを作成する手順も含まれています。

- ヘルプのランディングページにアクセスするには、 **Help > Online help** を選択します。
- ダイアログボックスの状況依存ヘルプにアクセスするには、 **?** を選択します。

ダッシュボードを開く

ダッシュボードは、PEM クライアントツリーコントロールに相当する階層で表示されます。ツリーコントロール内の各オブジェクトのダッシュボードには、そのオブジェクトと、ツリーコントロールのそのレベルの下にある監視対象オブジェクトの情報が必要に応じて表示されます。

各ダッシュボードヘッダーには、サーバーが起動された日時、関連する場合、ダッシュボードが最後に更新された日時、およびトリガーされたアラートの現在の数が表示されます。ダッシュボードヘッダーに表示されるナビゲーションメニューから、他のダッシュボードに簡単にアクセスできます。メニューは階層的に編成されています。ツリーコントロールで現在強調表示されているオブジェクトに適切なメニューのみが使用できます。

- ダッシュボードから **Global Overview** を選択して、Global Overview ダッシュボードに戻ります。
- **Agents** メニューからエージェントの名前を選択して、そのエージェントのオペレーティングシステム分析ダッシュボードに移動します。
- **Servers** メニューからサーバー名を選択して、そのサーバーの Server Analysis ダッシュボードに移動します。

- **Databases** メニューからデータベース名を選択して、そのデータベースのデータベース分析ダッシュボードに移動します。
- **Dashboards** メニューを使用して、グローバルレベルまたは選択したエージェント、サーバー、またはデータベースの情報ダッシュボードに移動します。

ダッシュボードには、次の形式で統計情報が表示されます。

- PEM プローブによって収集された統計情報を提供するテーブル。
- プローブの最新の実行によって収集された情報を表示する円グラフ。
- プローブの最新の実行によって収集された比較統計を表示する棒グラフ。
- PEM プローブによって収集された統計データを表示する折れ線グラフ。

次のいずれかの手法を使用して、ダッシュボードを開くことができます。

- **Management > Dashboards** メニューから、アクティブなダッシュボード名を選択します。
- ツリー内の監視対象オブジェクトの名前を右クリックします。**Dashboards** メニューから、確認するダッシュボードの名前を選択します。

各ダッシュボードは、クライアントウィンドウのメインパネルの **Monitoring** タブに表示されます。ダッシュボードを開いた後、同じタブ内の他のダッシュボードに移動できます。

各ダッシュボードヘッダーには、他のダッシュボードに移動できるナビゲーションメニューが含まれています。ブラウザーの [進む] および [戻る] ボタンを使用して、以前に表示したダッシュボードをスクロールします。**Refresh** を使用して、現在のダッシュボードを更新します。

表形式で提供される統計を並べ替えるには、列見出しを選択します。並べ替え順序を逆にするには、それを再度選択します。各テーブルは、安定したソート機能を提供します。たとえば、各ユーザー名グループのセッション ID の昇順でテーブルをソートするには、最初に Session ID 列でソートし、次に User Name 列でソートします。

各グラフ、チャート、またはテーブルの右上隅にカーソルを合わせると、PEM クライアントツールバーアイコンが表示されます。アイコンにカーソルを合わせると、アイコンの機能を簡単に説明するツールチップが表示されます。

- **Refresh** アイコンを使用して、ダッシュボードに表示される情報を更新します。
- **Save Chart as Image** アイコンを使用して、選択したチャートを `.jpeg` 画像として保存します。
- **フルスクリーン**アイコンを使用してグラフを拡大し、グラフデータの詳細を表示します。
- **Personalize the chart configuration** アイコンを使用して、チャート固有の表示の詳細を選択できるコントロールパネルにアクセスします。
- **Explain** アイコンにカーソルを合わせて、グラフまたはチャートで共有されている情報の説明を確認します。

各グラフまたはチャートの右下隅には、グラフまたはチャートにプロットされた各項目を識別する凡例があります。

表示されている場合は、チャートの左上隅にある情報アイコンを選択して、チャートのコンテンツに関するメモと、必要に応じて、チャートのコンテンツを取得する 1 つ以上のプローブを有効にできるリンクを表示します。

使用可能なダッシュボード

PEM は次のダッシュボードを提供しています。使用可能な各ダッシュボードの詳細については、PEM クライアントのオンラインヘルプを参照してください。

アラートダッシュボード

アラートダッシュボードには、現在トリガーされているアラートが表示されます。グローバル概要から開いた場合、ダッシュボードには、システム上のすべての監視対象ノードの現在のアラートが表示されます。サーバー内のノードから開いた場合、レポートには、そのノードと、ツリー内のそのオブジェクトの下にあるすべての監視対象オブジェクトに関連するアラートが表示されます。

監査ログ分析ダッシュボード

EDB Postgres Advanced Server ユーザーの場合、 Audit Log Analysis ダッシュボードを使用して、監査ログと収集が有効になっているインスタンスから収集された監査ログを参照できます。

PGD 管理ダッシュボード

PGD Admin ダッシュボードには、 EDB Postgres Distributed ノード、グループ、およびワーカーに関する概要情報が表示されます。

PGD グループモニタリングダッシュボード

PGD Group Monitoring ダッシュボードには、 EDB Postgres Distributed グループサブスクリプションとグループレプリケーションスロットに関する情報が表示されます。

PGD ノードモニタリングダッシュボード

PGD ノードモニタリングダッシュボードには、 EDB Postgres Distributed ノードスロット、ノードレプリケーションレート、および選択したノードの競合履歴の概要に関する情報が表示されます。

データベース分析ダッシュボード

Database Analysis ダッシュボードには、選択したデータベースのパフォーマンス統計が表示されます。

I/O 分析ダッシュボード

I/O 分析ダッシュボードには、オブジェクト DML アクティビティやログ操作など、さまざまな領域の I/O アクティビティが表示されます。

メモリ分析ダッシュボード

Memory Analysis ダッシュボードは、Postgres サーバーのさまざまなメモリ関連のメトリックに関する統計を提供します。

オブジェクトアクティビティ分析ダッシュボード

オブジェクトアクティビティ分析ダッシュボードは、選択したデータベースのテーブル/インデックスのパフォーマンスの詳細を提供します。

オペレーティングシステム分析ダッシュボード

オペレーティングシステム分析ダッシュボードは、基になるマシンのオペレーティングシステムのパフォーマンスに関する情報を提供します。

プローブログ分析ダッシュボード

Probe Log Analysis ダッシュボードには、PEM エージェントから返されたエラーメッセージが表示されます。

サーバー分析ダッシュボード

Server Analysis ダッシュボードは、選択した Postgres サーバーの全体的な操作に関する一般的なパフォーマンス情報を提供します。

サーバーログ分析ダッシュボード

Server Log Analysis ダッシュボードを使用すると、PEM サーバーに保存されているサーバーログの内容をフィルタリングおよび確認できます。

セッションアクティビティ分析ダッシュボード

セッションアクティビティ分析ダッシュボードは、選択したサーバーのセッションワークロードとロックアクティビティに関する情報を提供します

セッション待機分析ダッシュボード

Session Waits Analysis ダッシュボードは、 Advanced Server セッションの現在の DRITA 待機イベントの概要を提供します。

ストレージ分析ダッシュボード

Storage Analysis ダッシュボードには、テーブルスペースとオブジェクトのスペース関連のメトリックが表示されます。

システム待機分析ダッシュボード

System Waits Analysis ダッシュボードには、 EDB Postgres Advanced Server セッションのシステム待機情報のグラフィカル分析が表示されます。

ストリーミングレプリケーション分析ダッシュボード

Streaming Replication Analysis ダッシュボードには、監視対象サーバーの WAL アクティビティに関する統計情報が表示され、フェールオーバーマネージャークラスターのステータスを監視できます。

ダッシュボード構成

[Dashboard Configuration] ダイアログボックスのオプションを使用すると、ダッシュボード上のすべての折れ線グラフのタイムラインをリンクできます。[Dashboard Configuration] ダイアログボックスを開くには、ダッシュボードヘッダーの **Settings** を選択します。

[Dashboard Configuration] ダイアログボックスを使用して、ダッシュボードに表示されるチャートの属性を制御します。

- 指定されたタイムラインをダッシュボードに表示される折れ線グラフに適用するには、**すべての折れ線グラフのタイムラインを** 有効にする**を設定します。**Disable**に設定されている場合、設定は後で使用できるように保存されますが、表示されるデータの量は変更されません。
- **Days** セレクターを使用して、折れ線グラフに表示する収集データの日数を指定します。
- **Hour(s)** セレクターを使用して、折れ線グラフに表示する収集データの時間数を指定します。
- **このダッシュボードの構成を記憶する**を選択して、カスタマイズされた時間スパンを現在のダッシュボードにのみ適用します。すべてのダッシュボードの折れ線グラフにタイムスパンをグローバルに適用することはオフのままにします。

[Dashboard Configuration] ダイアログボックスで指定した設定は、現在のユーザーのセッションにのみ適用されます。

設定を指定したら、[**Save**] を選択します。

カスタムダッシュボードの管理

PEM は、多数のシステム定義のダッシュボードにパフォーマンス統計を表示します。各ダッシュボードには、選択したオブジェクトに関連する統計を表示するチャート、グラフ、表を含む一連の概要ビューが含まれています。**Manage Dashboards** タブを使用して、システムに最も関連する情報を表示するカスタムダッシュボードを作成および管理できます。

カスタムダッシュボードを作成するには、**Manage Dashboards** タブの Quick Links セクションにある **Create New Dashboard** を選択します。

既存のダッシュボードを変更するには、ダッシュボード名の横にある **Edit** を選択します。ダッシュボードエディターが開き、ダッシュボードの定義が表示されます。ダッシュボードの定義の変更が終了したら、[**Save**] を選択します。

ダッシュボードを削除するには、ダッシュボード名の横にある **Delete** を選択します。

カスタムダッシュボードの作成

PEM ダッシュボードエディターを使用して、カスタムダッシュボードを作成または変更できます。カスタムダッシュボードには、事前定義されたチャート、カスタムチャート、または事前定義とカスタムチャートの組み合わせを含めることができます。

Configure セクションを使用して、ダッシュボードに関する一般情報を指定します。

- **Name** フィールドにダッシュボードの名前を指定します。
- **Level** リストを使用して、ダッシュボードを表示する PEM クライアントの PEM 階層のレベルを指定します。グローバルレベル、エージェントレベル、サーバーレベル、またはデータベースレベルで **Dashboards** メニューを使用して、ダッシュボードにアクセスできます。リストで選択された各レベルは、カスタムダッシュボードのグラフの基になるさまざまなメトリックセットを公開します。
- **Description** フィールドにダッシュボードの説明を入力します。

ダッシュボードを Ops ダッシュボードとして使用するには、**Ops ダッシュボードオプション** ボックスに情報を入力します。

- Ops モニタに表示するようにフォーマットされたダッシュボードを作成するには、**Ops Dashboard?** を **Yes** に設定します。
- Ops ダッシュボードの上部にダッシュボード名を表示するには、 **Show Title?** を **Yes** に設定します。
- **Font** リストを使用して、タイトルのカスタムフォントスタイルを選択します。
- **Font size** リストを使用して、タイトルのカスタムフォントサイズを選択します。

Permissions ボックスを使用して、新しいダッシュボードを表示できるユーザーを指定します。

- すべてのチームがダッシュボードにアクセスできるように、 **Share with all** を **Yes** に設定します。**Accessmissions** フィールドを有効にするには、 **No** に設定します。
- **Access attempts** フィールドを使用して、新しいダッシュボードを表示できるロールを指定します。フィールドで、ユーザーのリストから選択して、ダッシュボードへのアクセスを持つユーザーのリストにロールを追加します。

Configure Dashboard セクションが完了したら、右上隅の矢印を選択してセクションを閉じ、**Dashboard Layout Design** セクションにアクセスします。

セクションヘッダーで **Edit** を選択して、セクション名を指定します。次に、プラス記号+を選択してグラフをセクションに追加します。

各チャートカテゴリの右側にある矢印を使用して、使用可能なチャートを表示し、チャートを選択します。

チャートの詳細セレクターを使用して、チャートの配置の詳細を指定します。

- **Chart width** セレクターを使用して、チャートの幅を示します。チャートをダッシュボードの半分に表示するには **50%** を選択し、ダッシュボード全体の幅を使用するには **100%** を選択します。
- **チャートアラインメント**セレクターを使用して、セクション内のチャートの位置を左詰め、中央揃え、または右詰めで示します。

テーブルは常に中央に表示されます。

カスタムダッシュボードを作成または編集するときに、ドラッグしてセクション内のグラフを並べ替えたり、グラフを別のセクションに移動したりできます。

ダッシュボードに別のグラフを追加するには、セクションヘッダーのプラス記号+を選択します。ダッシュボードの編集が終了したら、[**Save**] を選択します。

Ops ダッシュボードの作成

PEM ダッシュボードエディターを使用して、Ops モニターに表示するようにフォーマットされたカスタムダッシュボードを作成できます。Ops ダッシュボードには、指定されたチャートとグラフが表示されますが、ヘッダー情報を省略し、余分なバナー、タイトル、境界線を最小限に抑えます。

Ops ダッシュボードを作成するには、Create Dashboard ダイアログボックスの **Ops ダッシュボードオプション**セクションに Ops 表示に関する詳細情報を提供します。

- Ops モニタに表示するようにフォーマットされたダッシュボードを作成するには、**Ops Dashboard?** を **Yes** に設定します。
- Ops ダッシュボードの上部にダッシュボード名を表示するには、 **Show Title?** を **Yes** に設定します。
- **Font** リストを使用して、タイトルのカスタムフォントスタイルを選択します。
- **Font size** リストを使用して、タイトルのカスタムフォントサイズを選択します。

Ops ダッシュボードにチャートとテーブルを追加した後、 **Save** を選択します。新しいダッシュボードダイアログボックスの **レベル**フィールドで指定された階層レベルの **ダッシュボード**メニューをナビゲートすることにより、ダッシュボードにアクセスできます。

5.3.9 Remote monitoring

リモートモニタリングは、別のホストにある PEM エージェントを使用して Postgres クラスターを監視することです。

PEM で Postgres クラスターをリモートでモニタするには、クラスターを PEM に登録し、PEM エージェントをバインドする必要があります。詳細については、*Registering a Postgres server* を参照してください。

次のシナリオでは、PEM を使用したリモート監視が必要です。

- AWS RDS で実行されている Postgres クラスター
- Postgres cluster running on BigAnimal

PEM リモート監視は以下をサポートします。

Feature Name	Remote monitoring supported?	Comments
Query information	Yes	NA
Viewing system reports	Yes	NA
Query information	Limited	Postgres クラスターとオペレーティングシステムのメトリックの間には、相関関係がありません。
Query information	Limited	Postges クラスターでアラートスクリプトを実行すると、実際の Postgres クラスターマシンではなく、バインドされた PEM エージェントが実行されているマシンで実行されます。
Query information	Limited	一部のダッシュボードは、完全なデータを表示できない場合があります。たとえば、Postgres クラスターが実行されているオペレーティングシステム情報は利用できないため、表示されません。
Query information	Limited	一部の PEM プローブは情報を結果ず、一部の機能が影響を受ける可能性があります。プローブ機能の詳細については、 *Query information* を参照してください。
Query information	Limited	オペレーティングシステム情報が利用できないため、Postgres Expert は部分的な情報を提供します。

次のページに続く

表 42 – 前のページからの続き

Feature Name	Remote monitoring supported?	Comments
Query information	Limited	スケジュールされたタスクは Postgres クラスターでのみ動作し、スクリプトはリモートエージェントで実行されます。
Query information	Limited	コア使用状況レポートには、完全な情報が表示されません。たとえば、プラットフォーム、コア数、および合計 RAM は表示されません。
Query information	No	NA
Query information	No	NA
Postgres Log Analysis Expert	No	NA
Query information	No	NA

5.4 Monitoring Barman

Barman Backup and Recovery Manager は、ビジネスクリティカルな環境での PostgreSQL サーバーのリモートバックアップとディザスターリカバリーのためのオープンソース管理ツールです。PostgreSQL のポイント インタイムリカバリーテクノロジーに依存しているため、DBA はバックアップの完全なカタログと、複数のリモートサーバーのリカバリフェーズを 1 つの場所からリモートで管理できます。詳細については、*Monitoring Barman* を参照してください。

バージョン 8.4 以降、PEM コンソールを介して Barman サーバーをモニタできるようになりました。

5.4.1 Barman をモニタリングするための前提条件

Barman サーバーを PEM コンソールに追加する前に

- Barman ホストに Barman を手動でインストールして構成する必要があります。Barman のインストールと構成の詳細については、*Monitoring Barman* を参照してください。
- Barman ホストに pg-backup-api ツールをインストールします。インストールの詳細については、pg-backup-api を参照してください。

5.4.2 Barman サーバーの構成

次を使用して Barman サーバーを構成および編集できます。

- PEM Web クライアント
- `pemworker` コマンドライン

PEM Web クライアントを使用する

構成

Create –BARMAN Server ダイアログボックスを使用して、既存の Barman サーバーを PEM サーバーに登録できます。ダイアログボックスを開くには、BARMAN Servers ノードを右クリックし、 **Create-BARMAN Server** を選択します。

General タブを使用して、Barman サーバーの一般的なプロパティを説明します。

- **Name** フィールドを使用して、サーバーの名前を指定します。名前は、ブラウザツリー内のサーバーを識別します。
- **URL** フィールドを使用して、Barman がインストールされているホストの URL を指定します。
- **Team** フィールドを使用して、PostgreSQL ロール名を指定します。このロールのメンバーである PEM ユーザー、最初にサーバーを作成した人、または PEM サーバーのスーパーユーザー権限を持つ PEM ユーザーのみが、PEM にログオンするとこのサーバーを表示できます。このフィールドを空白のままにした場合、すべての PEM ユーザーにサーバーが表示されます。

[**PEM Agent**] タブを使用して、PEM エージェントの接続の詳細を指定します。

- **Bound Agent** フィールドを使用して、Barman サーバーとして構成するエージェントを選択します。Barman でサポートされている PEM エージェントのみがリストされています。
- **Probe Frequency** フィールドを使用して、指定した間隔でプローブを実行する秒数を指定します。
- **Hearbeat** フィールドを使用して、PEM エージェントの可用性を秒単位で確認する間隔を指定します。

注釈: Barman サーバーを登録した後、PEM エージェントを再起動する必要があります。

編集

Barman サーバーを編集するには、ブラウザツリーからサーバーを右クリックし、**プロパティ** を選択します。

- **PEM Agent** タブを使用して、バインドされたエージェント、プローブ周波数、およびハートビートを変更します。Barman サーバーの所有者のみが [**PEM エージェント**] タブのフィールドを変更できます。
- **Information** タブを使用して、Barman サーバーに関する詳細情報を表示します。このタブには、Barman 関連のプローブが実行されるたびに値が設定されます。

- **Configuration** タブを使用して、Barman サーバーの構成設定を表示します。このタブには、Barman 関連のプローブが実行されるたびに値が設定されます。

pemworker コマンドラインを使用する

pemworker コマンドラインオプションを使用して Barman サーバーを構成できます。

```
asheshvashi@pem:~/PEM/agent$ ./pemworker --update-barman --help
./pemworker --update-barman [barman-update-options]

barman-update-options:
  --id <barman-id> (ID for the existing BARMAN API pg-backup-api)
  --api-url <url> (URL of the BARMAN API pg-backup-api)
  --probe-execution-frequency <interval> (Default: 30, Probe the BARMAN API pg-backup-
↪api at regular interval in seconds and fetch the metrics.)
  --heartbeat-interval <interval> (Default: 10, Ping the BARMAN API pg-backup-api␣
↪status API at a regular interval in seconds for checking its availability.)
  --ssl-crt <certificate_file> (SSL certificate file for the BARMAN API.)
  --ssl-key <key_file> (Private SSL key for the BARMAN API.)
  --ssl-ca-cert <ca_file> (CA certificate to verify peer against the BARMAN API.)
  --config-file/-c <config_file> (Path to the agent configuration file.)

asheshvashi@pem:~/PEM/agent$ ./pemworker --unregister-barman --help
./pemworker --unregister-barman [barman-unregistration-options]

barman-unregistration-options:
  --id <barman-id> (ID for the existing BARMAN API, registered with the PEM Server.pg-
↪backup-api)
  --config-file/-c <config_file> (Path to the agent configuration file.)

asheshvashi@pem:~/PEM/agent$ ./pemworker --register-barman --help
./pemworker --register-barman [barman-registration-options]

barman-registration-options:
  --api-url <url> (URL of the BARMAN API pg-backup-api)
  --description <name> (Description to show on the UI User interface for the BARMAN␣
↪API.)
  --probe-execution-frequency <interval> (Default: 30, Probe the BARMAN API pg-backup-
↪api at regular interval in seconds and fetch the metrics.)
  --heartbeat-interval <interval> (Default: 10, Ping the BARMAN API pg-backup-api␣
↪status API at a regular interval in seconds for checking its availability.)
  --ssl-crt <certificate_file> (SSL certificate file for the BARMAN API.)
  --ssl-key <key_file> (Private SSL key for the BARMAN API.)
  --ssl-ca-cert <ca_file> (CA certificate to verify peer against the BARMAN API.)
  --team <database-role> (Specify the name of the database group role, on the PEM␣
```

(次のページに続く)

(前のページからの続き)

```
↪backend database server, that should have access to this BARMAN API Server.)
  --owner <database-user> (Specify the name of the database user, on the PEM backend␣
↪database server, who will own the BARMAN API Server.)
  --config-file/-c <config_file> (Path to the agent configuration file.)
```

注釈: Barman サーバーを登録した後、PEM エージェントを再起動する必要があります。

5.4.3 PEM ダッシュボードでの Barman サーバーの詳細の表示

Barman サーバーが構成されると、その Barman サーバーのバックアップ全体とサーバー関連の詳細を PEM ダッシュボードで表示できます。

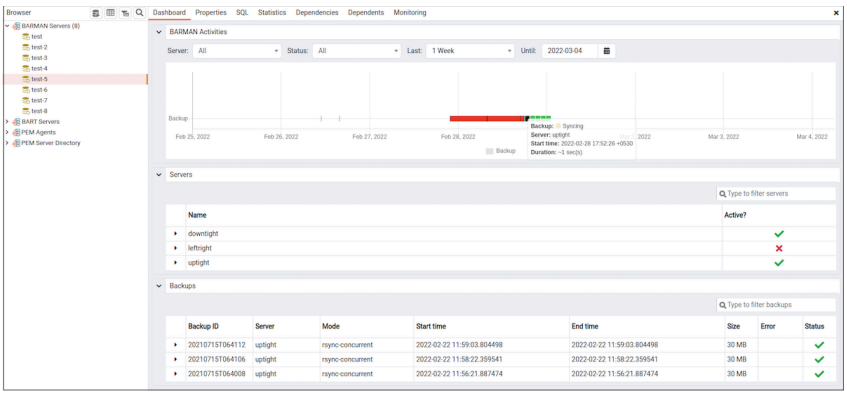


図 13: BARMAN dashboard

監視対象の Barman サーバーを選択すると、関連するすべてのデータベースサーバーの詳細とそのアクティビティが Barman アクティビティパネルのダッシュボードにグラフとして表示されます。フィルタボックスで指定した任意の基準データベースサーバー、ステータス、期間、または日付に基づいてアクティビティを選択できます。

[Servers] パネルには、その Barman サーバーによって管理されているすべてのデータベースサーバーのリストがアクティブなステータスとともに表示されます。

[Backups] パネルには、その Barman サーバーによって管理されているすべてのデータベースサーバーのバックアップのリストが表示されます。リストをフィルタリングして、データベースサーバーの詳細を表示できます。フィルタボックスで指定した基準でリストをフィルタリングすることもできます。通常、このフィルタは、列の下にリストされている任意の種類の文字列値日付、時刻、サイズを除きます。たとえば、tar と入力してリストをフィルタリングし、tar 形式のバックアップのみを表示できます。

バックアップの詳細には、バックアップ ID、サーバー、モード、開始時刻、終了時刻、サイズ、エラー、ステータス列が含まれます。

5.5 Monitoring EDB Postgres Distributed

EDB Postgres Distributed は、高度な競合管理、データ損失保護、および.

PEM バージョン 8.1.0 および EDB Postgres Distributed バージョン 3.7.9 以降のダッシュボードを使用して、1 つ以上の EDB Postgres Distributed データベースノードに関するステータス情報を表示するように PEM を構成できます。

EDB Postgres Distributed データベースノードを監視するように PEM を構成するには、PEM Web クライアントを使用してサーバー定義を作成します。**データベースサーバーの手動登録** ダイアログボックスのタブを使用して、 EDB Postgres Distributed データベースノードの一般的な接続プロパティを指定します。ただし、次の例外。

- [**PEM Agent**] タブの **Database** フィールドに EDB Postgres Distributed 対応データベース名を指定します。
- [**PEM Agent**] タブの **username** フィールドに pgd_monitor または pgd_superuser ロールを持つユーザーを指定します。

サーバー定義を保存すると、 EDB Postgres Distributed データベースノードは、PEM クライアントオブジェクトブラウザツリーの PEM サーバーディレクトリの下のサーバーのリストに含まれます。EDB Postgres DistributedPGD ダッシュボードからノードをモニタできます。

EDB Postgres DistributedPGD ダッシュボードにモニタリング情報を含めるには、各 EDB Postgres Distributed グループの相対プローブを有効にする必要があります。EDB Postgres Distributed *Probes* の完全なリストを参照してください。

プローブを有効にするには、ノード名を右クリックし、 **Management > Manage Probes** を選択します。

EDB Postgres Distributed データベースノードをモニタするには、オブジェクトブラウザツリーでノードの名前を右クリックします。**Dashboards** メニューから、PGD Admin、PGD Group Monitoring、または PGD Node Monitoring ダッシュボードを選択します。

5.6 Monitoring Failover Manager

EDB Failover Manager を使用してレプリケーションシナリオをモニタしている場合、Failover Manager を手動でインストールして構成する必要があります。 Failover Manager のインストールの詳細については、EDB website を参照してください。

ストリーミングレプリケーションダッシュボードでフェールオーバーマネージャークラスターのステータスを監視するには、クラスターの各ノードの **Server Properties** ダイアログボックスの **Advanced** タブで次の情報を提供する必要があります。

- **EFM Cluster Name** フィールドを使用して、Failover Manager クラスターの名前を指定します。クラスター名は、クラスタープロパティファイルの名前の接頭辞です。たとえば、クラスタープロパティファイルの名前が `efm.properties` の場合、クラスター名は efm です。
- **EFM Installation Path** フィールドを使用して、Failover Manager バイナリファイルの場所を指定します。デフォルトでは、Failover Manager バイナリファイルは`/usr/efm-<X>/bin` にインストールされます。`<X>` は EFM バージョンです。

注釈: フェールオーバーマネージャーを監視するために、PEM エージェントはユーザー efm として `efm cluster-status-json <cluster_name>` コマンドを実行します。クラスタープロパティファイルが efm ユーザーのホームディレクトリにない場合、このコマンドは失敗します。この場合、efm のホームディレクトリにクラスタープロパティファイルへのシンボリックリンクを作成し、efm がクラスタープロパティファイルを読み取りおよび実行する権限があることを確認します。サーバーを登録すると、Streaming Replication Analysis ダッシュボードのダッシュボードの下部近くに EFM クラスターに関するステータス情報が表示されます。

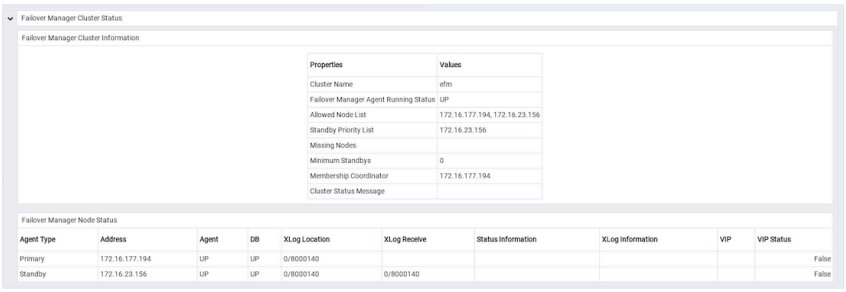


図 14: The Failover Manager cluster status report

Streaming Replication Analysis ダッシュボードの **Failover Manager Cluster Status** セクションには、監視対象クラスターに関する情報が表示されます。

Failover Manager Cluster Information テーブルは、Failover Manager クラスターに関する情報を提供します。

- Properties 列には、クラスタープロパティの名前が表示されます。
- Values 列には、プロパティの現在の値が表示されます。

[Failover Manager Node Status] テーブルには、Failover Manager クラスターの各ノードに関する情報が表示されます。

- Agent Type 列には、ノードにあるエージェントの種類が表示されます。使用可能な値は、プライマリ、レプリカ、Witness、Idle、および Promoting です。
- Address 列には、ノードの IP アドレスが表示されます。
- Agent 列には、ノードにあるエージェントのステータスが表示されます。
- DB 列には、ノードにあるデータベースの状態が表示されます。
- XLog Location 列には、データベースのトランザクションログの場所が表示されます。
- Status Information 列には、ノードに関するエラー関連の情報が表示されます。
- XLog Information 列には、トランザクションログに関するエラー関連の情報が表示されます。
- VIP 列には、ノードに関連付けられた VIP アドレスが表示されます。
- VIP Status 列には、VIP がノードでアクティブな場合は True が表示され、そうでない場合は False が表示されます。

5.6.1 プライマリノードの置き換え

PEM クライアントを使用して、フェールオーバーマネージャークラスターのプライマリノードをレプリカノードに置き換えることができます。フェールオーバープロセスを開始するには、**Tools > Server > Replace Cluster Primary** を選択します。現在のプライマリノードを置き換えることを確認するプロンプトが表示されます。

Yes を選択して、現在のプライマリノードをフェールオーバーマネージャークラスターから削除し、レプリカノードをフェールオーバーマネージャークラスターの読み取り/書き込みプライマリノードの役割に昇格させます。最も高いプロモーション優先度フェールオーバーマネージャーで定義されたノードが、新しいプライマリノードになります。PEM はジョブステータスを報告します。

ジョブが完了し、ストリーミングレプリケーション分析ダッシュボードが更新されたら、フェールオーバーマネージャーノードのステータステーブルを確認して、レプリカノードがフェールオーバーマネージャークラスター内のプライマリの役割に昇格したことを確認できます。

5.6.2 スイッチオーバー EFM クラスター

PEM クライアントを使用して、フェールオーバーマネージャークラスターのプライマリノードをレプリカノードに置き換えることができます。スイッチオーバープロセスを開始するには、**Tools > Switchover EFM Cluster** を選択します。EFM クラスターをスイッチオーバーすることを確認するプロンプトが表示されます。以下を行うには **Yes** を選択します

- フェールオーバーマネージャーのスイッチオーバーを開始します。
- レプリカノードを読み取り/書き込みプライマリノードの役割に昇格させます。
- プライマリデータベースをフェールオーバーマネージャークラスター内の新しいレプリカとして再構成します。

最も高いプロモーション優先度フェールオーバーマネージャーで定義されたノードが、新しいプライマリノードになります。PEM はジョブステータスを報告します。ジョブが完了し、Streaming Replication Analysis ダッシュボードが更新されたら、Failover Manager ノードのステータステーブルを確認して、スイッチオーバーが発生したことを確認できます。

5.7 Monitoring Replication Server

xDB レプリケーションシナリオの一部である EDB Postgres Advanced Server または PostgreSQL データベースから統計を取得するように PEM を構成する前に、xDB レプリケーションを手動でインストールして構成する必要があります。xDB レプリケーションソリューションとドキュメントの詳細については、を参照してください。

PEM xDB Replication プローブは、 EDB Postgres Advanced Server または PostgreSQL パブリケーションデータベースがある xDB マルチプライマリまたはシングルプライマリレプリケーションを使用するクラスターの遅延データを監視します。他の独自のデータベースホスト Oracle または SQL Server と EDB Postgres Advanced Server または PostgreSQL 間でレプリケーションを構成している場合、プローブは遅延情報を返すことができません。

デフォルトでは、xDB Replication プローブは無効になっています。 xDB Replication プローブを有効にするには

1. サーバーの名前を右クリックし、コンテキストメニューから **Connect** を選択します。
2. プロンプトが表示されたら、認証情報を入力します。
3. 接続した後、ツリーのサーバーノードを展開し、レプリケートデータベースの名前を選択します。
4. **Management > Manage Probes** を選択します。

Manage Probes タブを使用して、xDB Replication プローブを構成します。

- プローブの実行間の分と秒を変更するには、 **Default** を **No** に設定します。
- **Enabled?** スライダーを使用して、xDB Replication プローブを有効にします。
- **Data Retention** フィールドの **Default** スライダーを **No** に設定して、PEM がプローブによって取得された情報を保存する日数を変更します。

プローブを有効にした後、返されたメトリックを使用して、PEM クライアントでカスタムグラフとダッシュボードを作成できます。

5.8 Monitoring event history

`pem.event_history` テーブルで実行されたイベントの詳細をモニタできます。ユーザー名、イベントの実行時間、発生したイベントのタイプ、イベントに関する詳細を提供します。テーブルのフィールドは次のとおりです。

!!!注 現在、`pem.event_history` テーブルは、アラートブラックアウト履歴のみを記録します。

Field name	Description
recorded_time	イベントの実行時間を記録します。日付と時刻を小数部秒とともに表示します。
user_name	イベントを実行またはスケジュールしたユーザーのユーザー名。
component	イベント名。
operation	イベントに対して実行されたアクション。たとえば、ユーザーがアラートブラックアウトを有効にしている場合、`enable_alert_blackout`が表示されます。
message	イベントに対して実行されたアクションの説明。たとえば、ユーザーがサーバーのアラートブラックアウトを有効にしている場合、`Enabled the alert blackout for the server`が表示されます。
details	発生したイベントの JSON 出力。

5.9 Tuning performance

次の機能は、パフォーマンスの調整に役立ちます。

- *Tuning Wizard*
- *Postgres Expert*
- *Performance Diagnostic*

5.9.1 Tuning Wizard

Tuning Wizard は、 PostgreSQL または EDB Postgres Advanced Server インストールを確認し、インストールの予想されるワークロードに調整された構成オプションセットを推奨します。ベンチマークシステムまたは作業負荷の高いシステムでは、最適なパフォーマンスを達成するために手動調整が必要になる場合があります。

Tuning Wizard を使用する前に、サーバーの Properties ダイアログボックスでサービスの名前を指定する必要があります。[**Advanced**] タブの [**Server ID**] フィールドにサービス名を入力します。

チューニングウィザードは、バインドされた PEM エージェントと同じサーバーにあるサーバーに対してのみ推奨事項を作成できます。サーバーを定義するときに**リモートモニタリング** フィールドに **Yes** の値を指定すると、サーバーはチューニングウィザードツリーに表示されません。

1. Tuning Wizard を開くには、PEM クライアントで **Management > Tuning Wizard** を選択します。
2. **Next** を選択します。
3. ツリーの **Servers** ノードを展開すると、サーバーのリストが表示されます。これらのサーバーはすべて PEM によって監視されており、チューニングに利用できます。調整するサーバーを選択します。

注釈: サーバーのサービス名を指定しない場合、チューニングウィザードはツリー上のサーバー名の横に警告を表示します。

---

4. **Next** を選択します。
5. **Machine attempts** フィールドのオプションを選択して、各サーバーによって実行される作業の種類を指定します。サーバーによって実行される作業のタイプにより、チューニングウィザードがシステムリソースを割り当てる方法が決まります。
   - システムリソースの大部分をデータベースサーバーに専用にするには、**Dbuilt** を選択します。
   - 適度な量を専用にするには、 **Mixed use** を選択します
   - 比較的少量のシステムリソースをデータベースサーバー専用にするには、**Developer workstation** を選択します。

- **Workload Selection** フィールドのオプションを選択して、通常のワークロードのタイプを指定します選択したサーバーで実行
  - 選択したサーバーが主にオンライントランザクションワークロードの処理に使用される場合は、**OLTP** を選択します。
  - 選択したサーバーがトランザクション処理とデータレポートを組み合わせて提供する場合は、**Mixed** を選択します。
  - **を選択します**データウェアハウス** サーバーが大量のデータのレポートに使用される場合。

6. **Next** を選択します。
7. Tuning Changes Summary ダイアログボックスのツリーには、チューニングウィザードによって分析された各サーバーに推奨されるパラメーター設定の変更が表示されます。チューニングウィザードを適用する、またはプレビューレポートに含める推奨事項を選択します。
   - パラメーター名を選択すると、チューニングウィザードにパラメーター設定が含まれます。
   - サーバー名を選択すると、チューニングウィザードにすべてのパラメーター設定の推奨事項が含まれます指定されたサーバー。
8. **Next** を選択します。
9. スケジュールまたは実行ですか?ダイアログボックスで、PEM が推奨される変更を適用する時間を選択するか、推奨される変更の詳細を含むレポートを生成します。

PEM は、[Tuning Changes Summary] ダイアログボックスで選択した推奨変更を加えます。レポートを生成することを選択した場合、PEM はレポートを作成します。これには、現在値のリストと、[Tuning Changes Summary] ダイアログボックスで選択した構成パラメーターの推奨変更が含まれています。変更を実装するには、Tuning Wizard をもう一度開き、[Tuning Changes Summary] ダイアログボックスで変更するパラメーターを選択する必要があります。

Schedule Changes を選択して、スケジュールオプションを表示および指定します。

Configuration now? スライダーを次のように設定できます。

```
-   **Yes**  &mdash; チューニングウィザードの推奨事項を適用し、サーバーを再起動します。

-   **No**  &mdash; カレンダーセレクターで日付と時刻を指定できる **Time?** フィールドを有効にします。この時点で、PEM は推奨される変更を適用し、サーバーを再起動します。
```

Generate Report を選択して、レポートオプションを表示します。

View Report now? スライダーを次のように設定できます。

```
-   **Yes**  &mdash; チューニングウィザードレポートを画面に表示する。

-   **No**  &mdash; ファイル名と場所を指定できる **レポートをファイルに保存します** フィールドを有効にします。
```

10. **Finish** を選択します。

チューニングウィザードが推奨される変更を実装したことを確認するには、変更したサーバーの postgresql.conf ファイルを確認します。変更が適用されると、チューニングウィザードは、postgresql.conf ファイルの変更された各パラメーターの上にコメントを追加します。

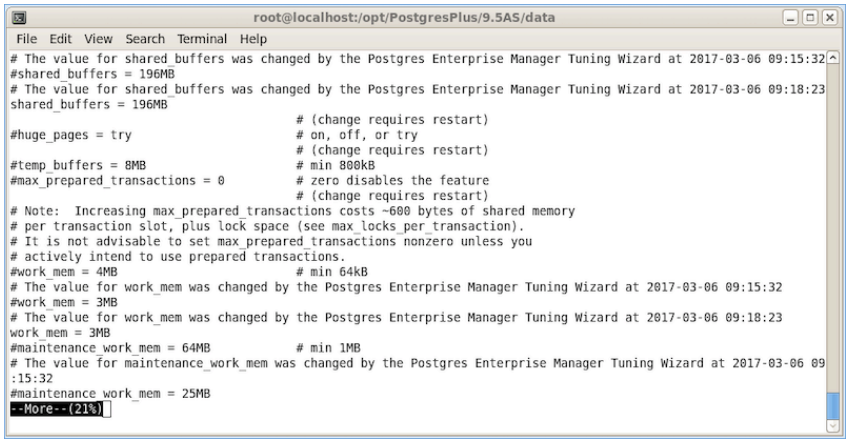


図 15: Confirming a change in the postgresql.conf file

サーバーを照会することにより、パラメーター値を確認することもできます。たとえば、shared_buffers パラメーターの値を確認するには、 クエリツール [Tools] メニューからアクセスまたは psql クライアントを使用して SQL コマンドラインを開き、コマンドを発行します。

```
SHOW shared_buffers;
```

サーバーから返された値は、パラメーターが変更されたかどうかを確認します。

5.9.2 Postgres Expert

Postgres Expert ユーティリティは、最適なパフォーマンス、セキュリティなどを実現する Postgres サーバーの最適な構成方法に関する専門家のアドバイスを提供します。Postgres Expert は、ベストプラクティスの逸脱についてサーバーを評価することにより、PostgreSQL DBA-in-a-box として機能します。 Postgres Expert には、3 人の専門家が含まれています。

- Configuration Expert
- スキーマエキスパート
- セキュリティエキスパート

各エキスパートに特定のルールを選択して、すべてのルールを評価または受け入れることができます。次に、注意が必要なベストプラクティスの問題を詳細に記載した Postgres Expert レポートを確認します。

Postgres Expert ウィザードを使用する

Postgres Expert ウィザードを使用するには、PEM クライアントで **Management > Postgres Expert** を選択します。

1. Welcome メッセージで、 **Next** を選択します。
2. **Experts/Rules** ツリーには、ベストプラクティスの逸脱を特定するための利用可能なエキスパートとルールがリストされています。選択したサーバーまたはデータベースの評価に使用するものを選択します。

ツリーは、3 人のエキスパートの下でルールを分類しています。

- **Configuration Expert** は、サーバーまたはオペレーティングシステムのパラメーター設定を評価して、システムパフォーマンスを向上させる可能性のある調整を見つけます。
- **Schema Expert** は、スキーマオブジェクトの配置などを評価します欠落している主キーまたはインデックスのない外部キー。
- **Security Expert** は、システムを評価してセキュリティの脆弱性を見つけます。

3. **Experts/Rules** の左側のチェックボックスを使用して、Postgres Expert が、選択したアイテムに関連するベストプラクティスの逸脱について、選択したサーバーの構成を分析することを示します。
- **の横にあるチェックボックスを使用しますツリーコントロールにリストされているすべての項目を選択または選択解除する**エキスパート/ルール**
- エキスパートの名前の横にあるチェックボックスを使用して、ツリーコントロールのそのノードの下にリストされているすべての構成項目を選択または選択解除します。
- ルールの横にあるチェックボックスを使用して、Postgres Expert レポートに含めるルールを選択または選択解除します。選択したら、[**Next**] を選択します。

4. 評価するサーバーとデータベースを選択またはクリアします。複数のサーバーまたはデータベースを選択した場合、結果のレポートには、各ターゲットの個別の評価が含まれます。完了したら、[**Next**] を選択します。
5. クライアントでレポートを表示するには、 **View the Report now** を選択します。コピーを HTML ファイルに保存するには、[**レポートのダウンロード**] を選択します。レポートをダウンロードすると、ファイルはデフォルトのダウンロードディレクトリに保存されます。**Finish** を選択します。

Postgres Expert の推奨事項の確認

Postgres Expert は、選択したルールの評価を含むレポートを生成します。各ルールは、選択したサーバーの重大度、高、中、または低として分類されます。

レポートヘッダーには、以下を含むレポートの概要が含まれています。

- レポートの日時
- 評価されたルールの数
- ベストプラクティスからの逸脱の数

特定のサーバーのセクションにジャンプするには、 **Go to** リストからサーバーを選択します。

レポートの本文には、選択した各サーバーの詳細な結果が含まれています。結果は専門家ごとに並べ替えられます。各見出しの下では、ルール違反が重大度によってランク付けされます。

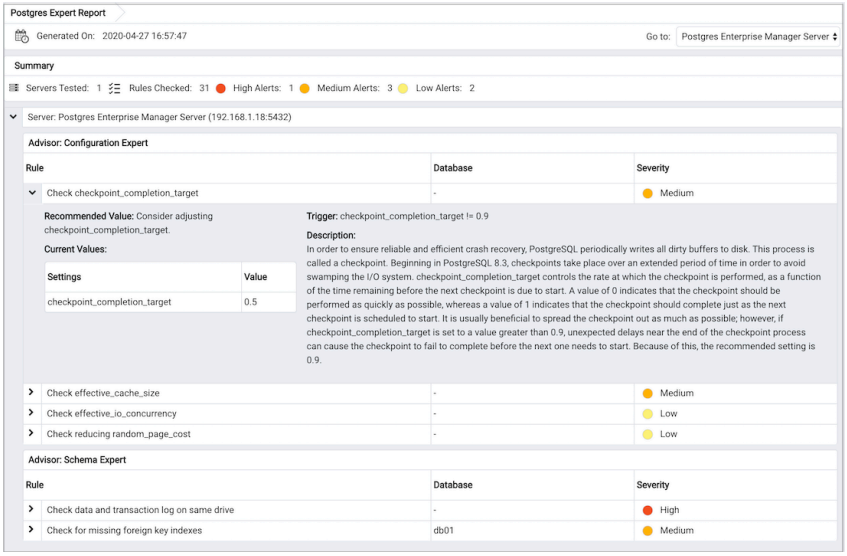


図 16: The detailed recommendation for a rule

レポート内の各ルールを選択して、そのルールの詳細と推奨事項を表示します。ルールを選択すると、次のセクション見出しが表示されます。

- Advisor は、推奨を促した Postgres Expert アドバイザーの名前をリストします。
- Trigger には、アラートを発生させたルールの説明が表示されます。
- [推奨値] には、Postgres Expert が選択したパラメーターを設定することをお勧めする値が表示されます。
- Description には、アラートの原因となったパラメーターに関する情報とアドバイスが表示されます。
- Current Values には、Postgres Expert の評価に影響を与えるパラメーターの現在値が表示されます。

UI の詳細についてのアプリ内ヘルプへのアクセス

アプリ内ヘルプは、メニュー項目、ダッシュボードのチャートとグラフ、ダイアログボックスのオプションを含む PEM Web インターフェイスに関する包括的な詳細を提供します。アプリ内のヘルプには、データベースオブジェクトを作成する手順も含まれています。

- ヘルプのランディングページにアクセスするには、 **Help > Online help** を選択します。
- ダイアログボックスの状況依存ヘルプにアクセスするには、 **?** を選択します。

5.9.3 Performance Diagnostic

パフォーマンス診断ダッシュボードは、待機イベントを監視することにより、Postgres インスタンスのデータベースパフォーマンスを分析します。診断グラフを表示するために、PEM は EDB 待機状態モジュールによって収集されたデータを使用します。EDB 待機状態の詳細については、 EDB wait states を参照してください。

複数のレベルで待機状態データを分析するには、選択したデータを絞り込みます。グラフの上位レベルで選択したデータは、下位レベルに設定されます。

注釈: パフォーマンス診断ダッシュボードは、EDB 待機状態モジュールが利用可能なすべての Postgres バリアントとバージョンで利用できます。

前提条件

- EDB 待機状態パッケージのインストール

```
-  PostgreSQL の場合、 `EDB repositories <https://repos.enterprisedb.com/>`_ を参照してください。

-  EDB Postgres Advanced Server の場合、 :ref:`EDB wait states </pg_extensions/
↪wait_states/installing/>` を参照してください。
```

- EDB Postgres Advanced Server の EDB 待機状態モジュールをインストールした後 1. `postgresql.conf` ファイルにライブラリのリストを構成します。

```
shared_preload_libraries = $libdir/edb_wait_states
```

2. データベースサーバーを再起動します。 3. メンテナンスデータベースに次の拡張機能を作成します。

```
CREATE EXTENSION edb_wait_states;
```

- スーパーユーザー権限のないユーザーとしてパフォーマンス診断ダッシュボードにアクセスする場合は、ユーザーが接続またはログインしているサーバーに基づいて、特定の ログインロール をユーザーに付与する必要があります。

- PEM サーバーにログインしているユーザーこの例では `test1` に `pem_comp_performance_diagnostic` ロールを付与します。

edb_notranlate_2 - `edb_wait_states` のすべての機能に対する `pg_monitor` ロールと `EXECUTE` 特権をデータベースユーザーこの例では `test2` に付与します。 `test2` ユーザーは、EDB 待機状態でデータベースサーバーに接続しています。

```
# Grant `pg_monitor` role
GRANT pg_monitor TO test2;
```

(次のページに続く)

(前のページからの続き)

```
# Grant `Execute` privilege on all functions
GRANT EXECUTE ON FUNCTION edb_wait_states_sessions TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_data TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_queries TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_samples TO test2;
GRANT EXECUTE ON FUNCTION edb_wait_states_purge TO test2;
```

前提条件が満たされていない場合、パフォーマンス診断ダッシュボードにアクセスするとエラーが表示されます。

パフォーマンス診断ダッシュボードを使用する

Performance Diagnostic ダッシュボードを開くには、PEM クライアントで **Tools > Server > Performance Diagnostic** を選択します。

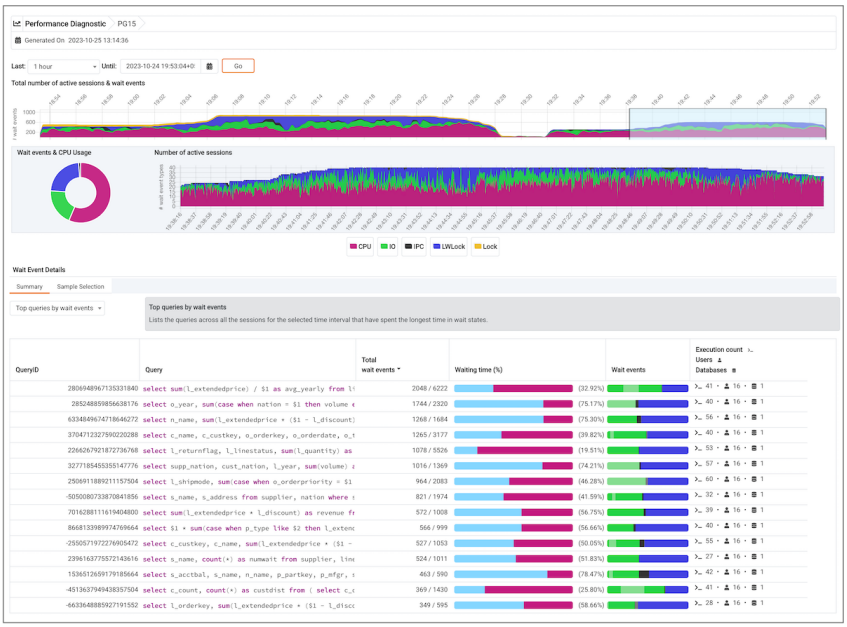


図 17: Performance Diagnostic dashboard

デフォルトでは、上部のパフォーマンス診断グラフは、現在の日時から開始して、過去 1 時間のデータを取得します。このグラフは、アクティブなセッションの数を含む時系列を示しています。この時系列の各ポイントは、過去 15 秒間の待機イベントと CPU 使用量の合計を表します。これらのセッションは、wait イベントを待機しているか、特定の時点で CPU を使用している可能性があります。この時系列は、EDB Wait States 拡張機能によって収集された待機イベントサンプルに基づいて生成されます。

新しいブラウザータブでパフォーマンス診断ダッシュボードを開くには、**Preferences > Open in New Browser Tab?** を選択します。

パフォーマンス診断ダッシュボード

- 最初のグラフには、選択した期間でアクティブなセッションの数と待機イベントタイプが表示されます。最初のグラフの期間を変更して、特定の期間のデータを分析できます。

デフォルトでは、最初のグラフの期間選択は 15 分です。持続時間は 1 時間まで延長できます。グラフで期間を表示するには、**Last** リストから期間を選択します。指定した日付と時刻のデータを表示するには、**Until** リストから日付と時刻を選択します。

- 次のセクションでは、最初のグラフで選択した期間に基づいて、次のグラフをプロットします。
- ドーナツグラフ—最初のグラフでの期間選択に従って、合計待機イベントタイプと CPU 使用率を表示します。このグラフは、これらのセッションがイベントの待機に費やした時間をより深く理解できます。
  - 折れ線グラフ—各ポイントが各サンプル時間のアクティブなセッションを表す時系列をプロットします。

各待機イベントタイプと CPU 使用率をより明確に区別するために、各待機イベントタイプのグラフは別の色で表示されます。

折れ線グラフの時間を選択して、待機イベントを分析します。すべてのグラフで待機イベントタイプを表示または非表示にするには、グラフの凡例を選択します。分析する待機イベントタイプのみを表示できると、分析が簡単になります。

- 3 番目のセクションには、2 番目のグラフで選択した期間に基づいて、パフォーマンス診断ダッシュボードに待機イベントの詳細が表示されます。2 つのタブに待機イベントの詳細が表示されます。
- **Overview** タブには、クエリ ID、合計 CPU イベント、CPU 使用率、待機イベント、および選択した時間間隔の実行数を含む上位 15SQLSQL のリストが表示されます。これらの SQL クエリーは 4 つのカテゴリにグループ化されています。
  - **待機イベント別の上位クエリー**—選択したサンプル時間のすべてのセッションで、待機状態で最も長い時間を費やしたクエリーをリストします。クエリ ID、クエリ、合計待機イベント、パーセンテージでの待機時間、待機イベント、および実行数と、そのクエリを実行したユーザーの数がデータベースの数で表示されます。
  - **CPU 使用率別の上位クエリ** &mdash ; 選択したサンプル時間のすべてのセッションで、最も CPU 時間を使用したクエリーをリストします。クエリ ID、クエリ、合計 CPU イベント、CPU 使用率、待機イベント、および実行数と、データベースの数でそのクエリを実行したユーザーの数が表示されます。テーブルのデータは、CPU イベントの数の降順です。
  - **待機イベント別のトップユーザー**—選択したサンプル時間のすべてのセッションで、クエリが待機状態で最も長い時間を費やしたユーザーをリストします。ユーザー名、データベース名、合計待機イベント、待機時間の割合、待機イベント、および待機状態で最も長い時間を費やしたクエリーの数が表示されます。
  - **CPU 使用率によるトップユーザー**—選択したサンプル時間のすべてのセッションで、クエリが最も多くの CPU 時間を使用したユーザーをリストします。ユーザー名、データベース名、合計 CPU イベント、CPU 使用率、待機イベント、および最も多くの CPU 時間を使用したクエリーの数が表示されます。

- **Sample Selection** タブには、選択したものの待機イベントの詳細が表示されます。2 番目のグラフのサンプル時間。待機イベントは次の基準でグループ化されています。
- **クエリ**—クエリ ID と選択したサンプル時間でクエリを実行したセッション数とともにクエリをリストします。任意の行の目を選択して、そのクエリに関する詳細を示すウィンドウを表示します。詳細については、クエリー情報 .
  - **ユーザー**—を参照してください。クエリを実行したユーザーごとにグループ化された待機イベントの詳細と、クエリのイベント数と実行回数をリストします。
  - **Waits** —選択したサンプル時間の各待機イベントタイプに属する待機イベントの数をリストします。両方のタブの下の行に表示されるデータをフィルタリングできます。列ヘッダーを選択して、データをアルファベット順に並べ替えることもできます。

クエリー情報

クエリ情報ウィンドウを開くには、 **Sample Selection** タブの下の **Queries** グループの任意の行で目を選択します。このウィンドウには、**Query information** セクションに、選択したサンプル時間でのクエリ ID とそれに対応するセッション ID がリストに表示されます。分析するデータを含む選択したクエリの**セッション ID** リストを選択できます。選択したセッション ID とクエリ ID に対応する詳細が表示されます。**クエリ情報** テーブルには、 SQL クエリーも表示されます。

Wait Event locations セクションには、選択したセッション ID とクエリ ID の待機イベントタイプの合計数が表示されます。2 種類のグラフが表示されます。

- ドーナツグラフ —カテゴリカルデータの割合を示します。
- タイムライン棒グラフ—時間の経過に伴う待機イベントタイプの数の傾向を視覚化します。

区別するために、各待機イベントタイプは棒グラフの異なる色で表されます。

Wait events セクションには、クエリの実行中に発生したすべての待機イベントを表示するテーブルがあります。待機イベントの数の降順にデータを表示します。2 番目のテーブルには、クエリ実行全体の期間に発生したサンプル時間と待機イベントが表示されます。これにより、一定の期間にわたってクエリ実行中の待機イベントを分析できます。また、そのクエリ ID とセッション ID の EDB Wait States 拡張機能によって収集された実際のサンプルも表示します。

5.10 Profiling workloads

RDBMS の専門家のほとんどは、非効率な SQL コードがほとんどのデータベースパフォーマンス問題の主な原因であることに同意しています。DBA と開発者にとっての課題は、大規模で複雑なシステムで SQL コードを見つけ、そのコードを最適化してパフォーマンスを向上させることです。

次の拡張機能は PEM と連携して、サーバーのワークロードをプロファイリングできます。

- SQL プロファイラー—データベーススーパーユーザーは、非効率な SQL コードを見つけて最適化できます
- *Installing the SQL Profiler extension* を参照してください
- *Upgrading SQL Profiler* を参照してください

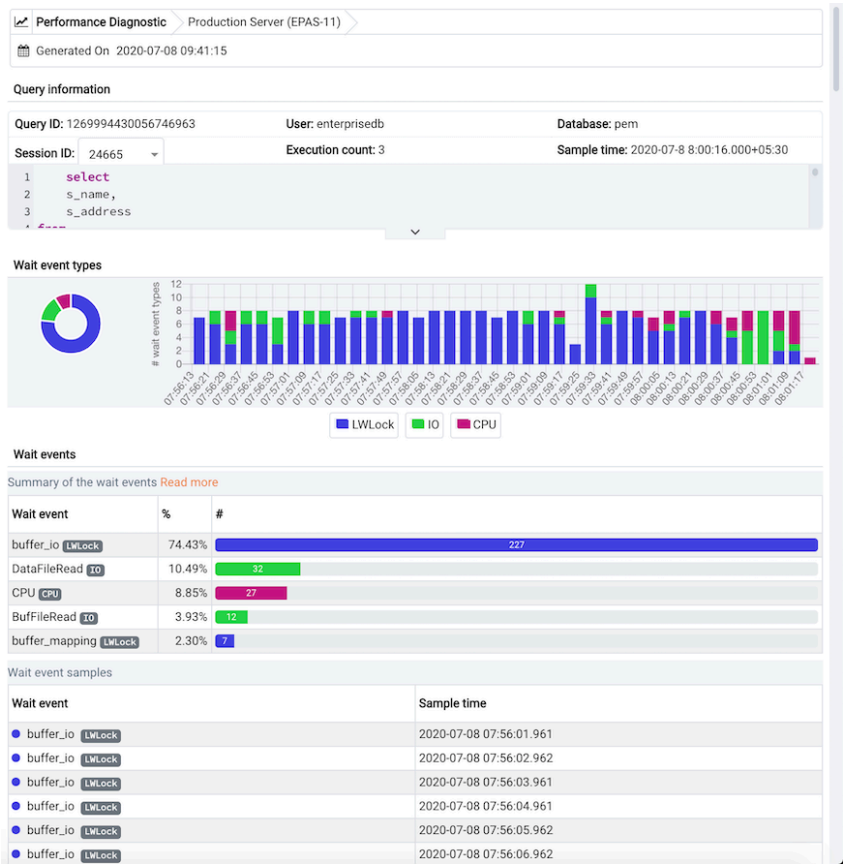


図 18: Performance Diagnostic - Query dashboard

- *Using SQL Profiler* を参照してください
- *Using the Index Advisor* —収集された SQL ステートメントを検査し、基になるテーブルのインデックス作成の推奨事項を作成することにより、SQL プロファイラーと連携して SQL の応答時間を向上させます

5.10.1 Installing the SQL Profiler extension

使用する各サーバーに SQL プロファイラー拡張機能をインストールして有効にする必要があります。たとえば、PostgreSQL 14 および PostgreSQL 15 を実行しているホストがある場合は、各サーバーで拡張機能を有効にする必要があります。

SQL プロファイラーは、使用している Postgres ディストリビューションと同じプラットフォームでサポートされています。SQL プロファイラー拡張機能のサポートは、PostgreSQL および EDB Postgres Advanced Server のバージョン 11 と EDB Postgres Extended のバージョン 12 から始まります。参照

- EDB Postgres Advanced Server Product Compatibility
- PostgreSQL Product Compatibility
- EDB Postgres Extended Server Product Compatibility

パッケージのインストール

前提条件

インストールプロセスを開始する前に

- Postgres をインストールします。参照
-
- Installing PostgreSQL
-
- リポジトリの設定をします。

リポジトリのセットアップは 1 回限りのタスクです。すでにリポジトリを設定している場合、この手順を実行する必要はありません。

リポジトリを設定するには、

EDB repositories に移動し、そこに提供されている指示に従ってください。

パッケージをインストールする

install コマンドの構文は次のとおりです。

```
sudo <package-manager> -y install edb-<postgres><postgres_version>-server-sqlprofiler
```

そこで

- <package-manager> は、オペレーティングシステムで使用されるパッケージマネージャーです。

Package manager	Operating system
dnf	RHEL 8 and derivatives
zypper	SLES
apt-get	Debian and derivatives

- <postgres> は、使用している Postgres のディストリビューションです。

Postgres distribution	Value
PostgreSQL	pg
EDB Postgres Advanced Server	as
EDB Postgres Extended	postgresextended

- <postgres_version> は、使用している Postgres のバージョンです。

たとえば、RHEL 8 プラットフォームに EDB Postgres Advanced Server 15 の SQL プロファイラーの最新バージョンをインストールするには

```
sudo dnf -y install edb-as15-server-sqlprofiler
```

拡張機能の有効化

拡張機能を有効にするには

1. プロファイルするサーバー上の postgresql.conf ファイルを編集し、shared_preload_libraries パラメーターを変更します。

```
shared_preload_libraries = $libdir/sql-profiler
```

1. Postgres サーバーを再起動します。
2. 次のコマンドを使用して、メンテナンスデータベースに SQL プロファイラー拡張機能を作成します。

```
# Connect to your maintenance database using psql
psql -d postgres -U enterprisedb
```

(次のページに続く)

(前のページからの続き)

```
# Create the extension
CREATE EXTENSION sql_profiler;
```

注釈: SQL プロファイラーを構成する前に PEM クライアントを使用して PEM サーバーに接続している場合、サーバーを切断して再接続して SQL プロファイラー機能を有効にする必要があります。

拡張機能を有効にすると、サーバーに存在するすべてのデータベースで SQL プロファイラーを使用できるようになります。

5.10.2 Using SQL Profiler

SQL プロファイラー拡張機能を使用すると、ユーザーは非効率な SQL コードを見つけて最適化できます。Microsoft の SQL Server プロファイラーは、操作と機能が PEM の SQL プロファイラーに非常に似ています。

SQL プロファイラーは PEM と連携して、サーバーのワークロードをプロファイリングできます。PEM エージェントの有無にかかわらずサーバーに SQL プロファイラー拡張機能をインストールして有効にできます。ただし、トレースは非管理対象サーバーでアドホックモードでのみ実行でき、管理対象サーバーでのみスケジュールできます。

SQL プロファイラーは、SQL トレースでの分析のために特定の SQL ワークロードをキャプチャおよび表示します。キャプチャされた SQL のトレースをすぐに開始して確認するか、キャプチャされたトレースを保存して後で確認できます。SQL プロファイラーを使用して、最大 15 の名前付けトレースを作成および保存できます。

SQL プロファイラーの権限

PEM で SQL プロファイラーツールにアクセスするには、2 つの前提条件があります。

1. PEM GUI にログインしているユーザーは、スーパーユーザーまたは PEM サーバーデータベース内のグループ `pem_comp_sqlprofiler` のメンバーである必要があります。詳細については、 を参照してください。
2. サーバーツリー `Username` でデータベースサーバー用に構成されたユーザーは、トレースが実行されているデータベースサーバー監視対象サーバーのスーパーユーザーである必要があります。

トレースの作成

[Create Trace] ダイアログボックスを使用して、SQL プロファイラーがインストールおよび構成されているデータベースの SQL トレースを定義できます。ダイアログボックスを開くには、PEM クライアントツリーでデータベースを選択し、**Tools > Server > SQL Profiler > Create Trace** を選択します。

Trace options タブを使用して、新しいトレースに関する詳細を指定します。

- **Name** フィールドにトレースの名前を入力します。
- **User filter** フィールドを使用して、トレースにクエリを含めるロールを指定します。すべてのロールからのクエリを含めるには、**Select All** を選択します。
- **Server Filter** フィールドを使用して、トレースするデータベースを指定します。すべてのデータベースに対するクエリーを含めるには、**Select All** を選択します。
- **最大トレースファイルサイズ**フィールドにトレースサイズを指定します。SQL プロファイラーは、指定されたサイズにほぼ達すると、トレースを終了します。
- **Create** を選択したときにトレースを開始するには、 **Run Now** フィールドで **Yes** を選択します。 [**Schedule**] タブのフィールドを有効にするには、 **No** を選択します。

Schedule タブを使用して、新しいトレースのスケジュールの詳細を指定します。

- **Start time** フィールドを使用して、トレースの開始時間を指定します。
- **End time** フィールドを使用して、トレースの終了時間を指定します。
- 毎日指定された時間にトレースを繰り返すには、**Repeat?** フィールドで **Yes** を選択します。 [**Periodic job options**] タブのフィールドを有効にするには、 **No** を選択します。

Periodic job options タブを使用して、定期的なトレースに関するスケジュールの詳細を指定します。**Days** セクションを使用して、ジョブを実行する日を指定します。

- **Week days** フィールドを使用して、トレースを実行する曜日を選択します。
- **Month days** フィールドを使用して、トレースを実行する日を選択します。
- **Months** フィールドを使用して、トレースを実行する月を選択します。

Times セクションを使用して、トレース実行のスケジュールを指定します。**Hours** および **Minutes** フィールドを使用して、トレースを実行する時間を指定します。

[Create Trace] ダイアログボックスの完了後、[**Create**] を選択してトレースを開始またはトレースをスケジュールします。

トレースをすぐに実行すると、トレース結果が PEM クライアントに表示されます。

既存のトレースを開く

以前のトレースを表示するには、PEM クライアントツリーでプロファイルされたデータベースを選択し、**Management > SQL Profiler > Open Trace** を選択します。「 Open Trace 」ダイアログボックスが開きます。

トレースリストからエントリを選択し、 **Open** を選択して、選択したトレースを **SQL Profiler** タブで開きます。

トレースのフィルタリング

フィルタは、1 つ以上のルールの名前付けセットであり、それぞれがトレースビューからイベントを非表示にできます。トレースにフィルタを適用すると、非表示のイベントはトレースから削除されませんが、表示から除外されます。

Filter を選択して [Trace Filter] ダイアログボックスを開き、フィルタを定義するルールまたはルールセットを作成します。各ルールは、イベントを呼び出したロールの ID またはイベント中に呼び出されたクエリタイプに基づいて、現在のトレース内のイベントをスクリーニングします。

既存のフィルタを開くには、 **Open** を選択します。新しいフィルタを定義するには、 **Add (+)** を選択して [**General**] タブに表示されるテーブルに行を追加し、ルールの詳細を提供します。

- **Type** リストを使用して、フィルタルールを適用するトレースフィールドを指定します。
- **Condition** リストを使用して、トレースをフィルタリングするときに値に適用する SQL プロファイラーの演算子の種類を指定します。
  - 指定された値を含むイベントをフィルタリングするには、 **Matches** を選択します。
  - **Does を選択します not match** 指定された値を含まないイベントをフィルタリングします。
  - **Value** フィールドで指定された文字列との完全な一致を含むイベントをフィルタリングするには、**Is 等しい** を選択します。
  - **を選択します** Value **フィールドで指定された文字列との完全な一致を含まないイベントをフィルタリングするには、**と等しくない**-で指定された文字列で始まるイベントをフィルタリングするには、Starts with を選択します Value** フィールド.
  - **Value** フィールドで指定された文字列で始まらないイベントをフィルタリングするには、**Does not start with** を選択します。
  - **Less than** を選択して、次のイベントをフィルタリングします **Value** フィールドで指定された数より小さい数値。
  - nu を持つイベントをフィルタリングするには **より大きい**を選択します **Value** フィールドで指定された数より大きいメトリック値。
  - **Value** で指定された数以下の数値を持つイベントをフィルタリングするには、**Less than or 等しい** を選択します field.
  - **Value** フィールドで指定された数以上の数値を持つイベントをフィルタリングするには、**Greater than or equal to** を選択します。
- **Value** フィールドを使用して、検索する文字列、数値、または正規表現を指定します。

ルールの定義が終了したら、 **Add (+)** を選択して別のルールをフィルタに追加します。フィルタからルールを削除するには、ルールを選択し、**Delete** を選択します。

フィルタを適用せずにフィルタ定義をファイルに保存するには、 **Save** を選択します。フィルタを適用するには、 **OK** を選択します。

トレースの削除

トレースを削除するには

1. PEM クライアントツリーでプロファイルされたデータベースを選択します。
2. **Management > SQL Profiler > Delete Trace(s)** を選択します。
3. Delete Traces ダイアログボックスで、トレース名の左側にあるアイコンを選択して、削除する 1 つ以上のトレースをマークします。
4. **削除** を選択します。PEM クライアントは、選択されたトレースが削除されたことを確認します。

スケジュールされたトレースの表示

スケジュールされたトレースのリストを表示するには、PEM クライアントツリーでプロファイルされたデータベースを選択します。**Management > SQL Profiler > Scheduled Traces** を選択します。

[Scheduled Traces] ダイアログボックスには、実行待機中のトレースのリストが表示されます。トレース名の左側にある **Edit** を選択して、トレースに関する詳細情報を表示します。

- **Status** フィールドは、現在のトレースのステータスをリストします。
- **Enabled?** スイッチは、トレースが有効になっている場合は Yes、無効になっている場合は No と表示されます。
- **Name** フィールドには、トレースの名前が表示されます。
- **Agent** フィールドには、トレースを実行するエージェントの名前が表示されます。
- **Last run** フィールドには、トレースが最後に実行された日時が表示されます。
- **Next run** フィールドには、次にスケジュールされたトレースの日時が表示されます。
- **Created** フィールドには、トレースが定義された日時が表示されます。

5.10.3 Using the Index Advisor

インデックスアドバイザーからの診断出力には次のものが含まれます。

- 推奨されるインデックスからの予測パフォーマンス利点
- 推奨されるインデックスの予測サイズ
- 推奨されるインデックスの作成に使用できる DDL ステートメント

インデックスアドバイザーは、SQL プロファイラーによってキャプチャされたトレースデータに基づいて、インデックス作成の推奨事項を作成できます。Index Advisor を開くには、 SQL Profiler Trace Data ペインで 1 つ以上のクエリーを選択し、ツールバーから **Index Advisor** を選択します。インデックスアドバイザーの構成と使用の詳細については、*EDB Postgres Advanced Server* **インスタンスサービス** *ID* **の設定** を参照してください。

注釈: インデックスアドバイザーは、非スーパーユーザーによって呼び出されたステートメントを分析できません。非スーパーユーザーによって呼び出されたステートメントを分析しようとすると、サーバーログに次のエラーが含まれます。

```
ERROR: access to library "index_advisor" is not allowed
```

注釈: pg_dump 機能の使用中はインデックスアドバイザーを無効にすることをお勧めします。

5.11 Viewing system reports

すべてのローカルおよびリモートで管理されているサーバーのシステム構成レポートとコア使用状況レポートを生成できます。このレポートを生成するには、**Management > Reports** を選択します。

Reports には次のオプションがあります。

- アラート履歴レポート JSON
- システム構成レポート JSON
- システム構成レポート HTML
- コア使用状況レポート JSON
- コア使用状況レポート HTML

スーパーユーザーまたは pem_admin ロールを持つユーザーのみが、システム構成またはコア使用状況レポートをダウンロードできます。

これらのレポートの情報は、最新のプローブランタイムを示しています。

5.11.1 アラート履歴レポート

Alert History レポートは、エージェントまたはサーバーレベルでのアラート履歴に関する詳細情報を JSON 形式で提供します。

管理>レポート>アラート履歴レポートを選択します。ダイアログボックスから、オプションを選択します。

- **エージェント/サーバー名**—リストからエージェントとサーバーを選択します。 **Overall System Report** オプションは、すべての登録済みおよびアクティブなエージェントとサーバーのレポートを生成するために使用できます。

- **タイムフレーム** —リストから必要な時間枠を選択します。
- **アラートタイプ**—アラート履歴レポートを生成するアラートタイプを選択します。

5.11.2 システム構成レポート

システム構成レポートは、ブラウザツリーの下にリストされている PEM エージェントグループ、PEM サーバーディレクトリグループ、およびカスタムグループに関する詳細情報を提供します。これらのグループには、PEM サーバー、PEM エージェント、およびデータベースサーバーを含めることができます。このレポートは、HTML および JSON 形式でダウンロードできます。

Postgres Enterprise Manager の概要では、次の詳細を提供します。

- Postgres Enterprise Manager バックエンドデータベースサーバーバージョン
- アプリケーションバージョン
- アプリケーションにアクセスするユーザー名
- Python バージョン
- Flask バージョン
- プラットフォーム固有の情報

この概要は、エージェントとサーバーの数に関する情報を提供します。 Group: PEM Agents パネルは、PEM エージェント、CPU コア、ディスク使用率、およびメモリ情報の詳細を提供します。

Group: PEM Server Directory パネルには、以下に関する詳細が提供されます。

- データベースサーバーバージョン
- ホスト
- ポート
- データベース名
- データベースのサイズ
- テーブル領域のサイズ

グループサーバー名は、サーバーが追加されるグループ名によって異なります。

5.11.3 コア使用状況レポート

Core Capacity レポートは、以下に固有のコア数に関する詳細情報を提供します。

- サーバーの種類
- データベースバージョン
- プラットフォームとグループ名

レポートでは、ローカル管理サーバーに関する詳細情報も提供します。

- 種類
- ホスト
- ポート
- プラットフォーム
- コア
- RAM

5.12 PEM command line interface

PEM CLI を使用すると、アラートテンプレートまたはプローブを PEM サーバーから別の PEM サーバーにエクスポートおよびインポートできます。

edb.repo リポジトリとは別にパッケージ edb-pem-cli-8.5.0-1-<platform_name> を使用して PEM CLI をインストールできます。バイナリファイル/usr/edb/pem/cli/bin/pem とライセンスファイル/usr/edb/pem/edb-pem-cli-license.txt を作成します。

PEM CLI パッケージをインストールした後、./pem -h コマンドを使用して、使用可能なすべてのオプションを表示します。

```
$ ./pem -h
__OUTPUT__
PEM CLI (Postgres Enterprise Manager Command Line Interface)
Usage: ./pem [OPTIONS] [SUBCOMMAND]

Options:
  -h,--help                   Print this help message and exit
  -v,--version                Show the app version and exit

Logging:
  --log-file [/home/asheshvashi/.pem-cli.log]
                              Set the log file

                              NOTE: Directory must exist

  -l,--log-level :value in {FATAL->1,ERROR->2,WARNING->3,DEBUG->5,INFO->4,VERBOSE->6}␣
↪OR {1,2,3,5,4,6} [INFO]
                              Set the log level

PEM Information:
  -p,--pem-url REQUIRED (Env:PEM_SERVER_URL)
                              Set the URL for accessing Postgres Enterprise Manager
                              e.g.
                              https://127.0.0.1:8443/pem
```

(次のページに続く)

(前のページからの続き)

```
                              https://pem_host:8443/pem

  -u,--pem-user REQUIRED (Env:PEM_API_USER)
                              User to access the PEM REST API

  -f,--password-file (Env:PEM_PASSWORD_FILE)
                              Location of the file, which contains the password.
                              The permission on a password file must disallow any␣
↪access to world or group.

                              NOTE:
  - The first non-empty line in the file will be treated as a
                                password.
  - Environment variable PEM_API_PASSWORD can be used for
                                providing the password instead of the password file

  -i,--insecure               Set the insecure connection


Subcommands:
  import-alert-templates      Import the alert template(s) into the PEM server.
  list-alert-templates        List the alert templates in the PEM server.
  export-alert-templates      Export the custom (user-defined) alert templates from␣
↪the PEM server.
  import-probes               Import the probe(s) into the PEM server.
  export-probes               Export the custom (user-defined) probes from the PEM␣
↪server.
  list-probes                 List the probes in the PEM server.
```

import-alert-templates サブコマンドで使用可能なすべてのオプションを表示できます。

```
$ ./pem import-alert-templates -h
__OUTPUT__
Import the alert template(s) into the PEM server.
Usage: ./pem import-alert-templates [OPTIONS]

Options:
  -h,--help                   Print this help message and exit
  -i,--in-file                Alert template file
  --ignore-existing           Ignore the existing alert template in the PEM server
  --ignore-existing-probes    Ignore the existing probes in the PEM server
```

list-alert-templates サブコマンドで使用可能なすべてのオプションを表示できます。

```
$ ./pem list-alert-templates -h
```

(次のページに続く)

(前のページからの続き)

```
__OUTPUT__
List the alert templates in the PEM server.
Usage: ./pem list-alert-templates [OPTIONS]

Options:
  -h,--help                    Print this help message and exit
  -c,--alert-template-category :value in {ALL->2,CUSTOM->1,SYSTEM->0} OR {2,1,0}␣
↪[CUSTOM]
                               List down the alert templates of this category
```

export-alert-templates サブコマンドで使用可能なすべてのオプションを表示できます。

```
$ ./pem export-alert-templates -h
__OUTPUT__
Export the custom (user-defined) alert templates from the PEM server
Usage: ./pem export-alert-templates [OPTIONS]

Options:
  -h,--help                    Print this help message and exit
  -i,--id INT ...              List of id for the custom alert templates to be exported
  -o,--out-file [Standard output (console)]
                               Output file name
  -w,--overwrite               Overwrite the existing output file
```

import-probes サブコマンドで使用可能なすべてのオプションを表示できます。

```
$ ./pem import-probes -h
__OUTPUT__
Import the probe(s) into the PEM server.
Usage: ./pem import-probes [OPTIONS]

Options:
  -h,--help                    Print this help message and exit
  -i,--in-file                 Probes file
  --ignore-existing            Ignore the existing custom probes in the PEM server
```

export-probes サブコマンドで使用可能なすべてのオプションを表示できます。

```
$ ./pem export-probes -h
__OUTPUT__
Export the custom (user-defined) probes from the PEM server
Usage: ./pem export-probes [OPTIONS]

Options:
  -h,--help                    Print this help message and exit
  -i,--probe-id INT ...        List of the custom probe-ids to be exported
```

(次のページに続く)

(前のページからの続き)

```
  -o,--out-file [Standard output (console)]
                              Output file name
  -w,--overwrite              Overwrite the existing output file
```

list-probes サブコマンドで使用可能なすべてのオプションを表示できます。

```
$ ./pem list-probes -h
__OUTPUT__
List the probes in the PEM server.
Usage: ./pem list-probes [OPTIONS]

Options:
  -h,--help                   Print this help message and exit
  -c,--probe-category :value in {ALL->2,CUSTOM->1,SYSTEM->0} OR {2,1,0} [CUSTOM]
                              List down the probes of this category
```

5.12.1 例

すべてのプローブシステムおよびカスタムプローブをリストするには、-c all オプションを使用します。

```
$ PEM_PASSWORD_FILE=/tmp/pass PEM_API_USER=postgres PEM_SERVER_URL=https://172.17.0.4/
↪pem ./pem -i list-probes -c all
```

そこで

- PEM_PASSWORD_FILE=/tmp/pass は、postgres ユーザーのパスワードを含むパスワードファイルです。
- PEM_API_USER=postgres は、PEM CLI を実行しているユーザー名です。
- PEM_SERVER_URL=https://172.17.0.4/pem は、プローブを含む PEM サーバーの URL です。
- -i list-probes は、 list-probes サブコマンドを使用した insecure オプションであり、安全でない接続を許可し、プローブをリストします。
- -c all は、すべてのシステムとカスタムプローブをリストするための all 値を備えた-c オプションです。

5.13 PEM Rest API

EDB Postgres Enterprise Manager は、REST API 機能をサポートしています。REST API 機能は、効果的なデータの読み取りおよび書き込み機能、および PEM UI にログインせずにさまざまな機能へのアクセスを提供します。詳細については、

REST API usage を参照してください。

第6章　Developing

6.1 Using the Query tool

PEM には、Postgres サーバーに対してアドホック SQL クエリを発行できる機能が豊富なインタラクティブ開発環境 IDE が含まれています。

クエリツールには、 **ツール> クエリツール**を選択するか、ブラウザツリーの選択したノードのコンテキストメニューからアクセスできます。クエリツールを使用すると、次のことができます。

- アドホック SQL クエリーを発行します。
- 任意の SQL コマンドを実行します。
- SELECT クエリーの結果セットが *[データ出力] タブ* の場合、編集します。
- 現在の接続とトランザクションステータスを構成および表示します。
- 出力パネルに表示されているデータを CSV ファイルに保存します。
- テキスト、グラフィカル、またはテーブル形式< https://explain.depesz.com >と同様に、SQL ステートメントの実行プランを確認します。
- SQL ステートメントに関する分析情報を表示します。

クエリツールには、2 つのパネルがあります。

- 上部パネルには SQL エディターが表示されます。パネルを使用して、クエリーを入力、編集、または実行できます。また、**History** タブも表示され、セッションで実行されたクエリを表示するために使用できます。編集中にスクラッチパッドを使用して、テキストのスニペットを保持できます。閉じたスクラッチパッドを再度開くか、新しいスクラッチパッドを開くには、SQL エディターを右クリックし、 **Add Panel** を選択します。同じテクニックを使用して、他のパネルから新しいスクラッチパッドを開くこともできます。
- 下部のパネルには、[Data Output] パネルが表示されます。タブ付きパネルには以下が表示されます。
  - クエリーによって返された結果セット
  - クエリーの実行に関する情報 plandb_tran_5 クエリーの実行に関連するサーバーメッセージ
  - サーバーから受信した非同期通知。

6.1.1 クエリツールツールバー

ツールバーには、状況依存のアイコンが使用されており、頻繁に実行されるタスクへのショートカットを提供します。

Icon	Behavior	Shortcut
Open File	以前に保存したクエリーを SQL エディターに表示します。	Accesskey + O
Save	クエリーを保存するか、 **Save** メニューにアクセスします。- **Save** を選択して、 SQL エディターパネルで選択したコンテンツをファイルに保存します。 - **Save As** を選択して、新しいブラウザーダイアログボックスを開き、 SQL エディターパネルで選択したコンテンツを保存する新しい場所を指定します。	Accesskey + S
Save Data Changes	データ出力パネルのデータの変更挿入、更新、または削除をサーバーに保存します。	F6
Find	SQL エディターに表示されるコードを検索、置換、または移動します。- **Find** を選択して検索ターゲットを提供し、 SQL エディターのコンテンツを検索します。 - **Find next** を選択して、次に出現する検索ターゲットを見つけます。 - **以前の検索** を選択して、検索ターゲットの最後に出現した場所に移動します。 - **Pesistent find** を選択して、エディターで検索ターゲットのすべてのオカレンスを識別します。 - **Replace** を選択して、ターゲットの個々のオカレンスを見つけて置換しますプロンプトを表示します。 - **Replace all** を選択して、エディターでターゲットのすべてのオカレンスを見つけて置換します。 - **Jump** を選択して、次の検索ターゲットに移動します。	Cmd+F Cmd+G Cmd+Shift+G Cmd+Shift+F Alt+G
Copy	表示/編集データモードのときに、[データ出力] パネルで現在強調表示されているコンテンツをコピーします。	Accesskey + C
Paste	表示/編集データモードで、コピーした行を新しい行に貼り付けます。	Accesskey + P
Delete	選択した行を削除対象としてマークします。 **Save Data Changes** を選択して、削除したコンテンツを更新します。	Accesskey + D

次のページに続く

表 1 – 前のページからの続き

Icon	Behavior	Shortcut
Edit	**Edit** メニューのオプションを使用して、テキスト編集ツールにアクセスします。オプションは、クエリツールモードのときに SQL エディタパネルに表示されるテキストに動作します。-現在選択しているテキストをインデントするには、**Indent Selection** を選択します。 - 現在選択しているテキストからインデントを削除するには、 **Unindent Selection** を選択します。 - **Inline Comment Selection** を選択して、SQL スタイルのコメント表記での選択を含む行を囲みます。 - **Inline Uncomment Selection** を選択して、選択した行から SQL スタイルのコメント表記を削除します。 - **Block Comment** を選択して、C スタイルのコメント表記で選択を含むすべての行を囲みます。このオプションはトグルとして機能します。	Tab Shift+Tab Cmd+/ Cmd+. Shift+Cmd+/
Filter	データの表示/編集モードでデータのフィルタリングと並べ替え基準を設定します。下矢印を選択して、他のフィルタリングと並べ替えオプションにアクセスします。 - **Sort/Filter** を選択して、並べ替えとフィルタリングダイアログボックスを開きます。 - 選択したセルの値を含む行のみを表示するには、 **Filter by Selection** を選択します。 - 選択したセルの値を含まない行のみを表示するには、 **Exclude by Selection** を選択します。 - **Remove Sort/Filter** を選択して、以前に選択した並べ替えまたはフィルタリングオプションを削除します。	Accesskey + F
Limit Selector	データセット内の最大行数の値を設定します。	Accesskey + R
Stop	現在実行中のクエリをキャンセルします。	Accesskey + Q
Execute/Refresh	SQL エディターパネルで選択したクエリーを実行または更新します。下矢印を選択して、他の実行オプションにアクセスします。- トランザクション中にエラーが発生した場合、トランザクションをロールバックするには **Auto-Rollback** を選択します。 - 各トランザクションをコミットするには、 **Auto-Commit** を選択します。トランザクションによって行われた変更は他の人に可視され、クラッシュが発生した場合に持続します。	F5
Explain	現在のクエリの説明プランを表示します。`EXPLAIN`の結果は、出力パネルの **Explain** タブにグラフィカルに表示され、 **Data Output** タブにテキスト形式で表示されます。	F7

次のページに続く

表 1 – 前のページからの続き

Icon	Behavior	Shortcut
Explain analyze	現在のクエリーで`EXPLAIN ANALYZE`コマンドを呼び出します。 **Explain Options** メニューから - **Verbose** を選択して、クエリプランに関する詳細情報を表示します。 - **Costs** を選択して、各プランノードの推定起動と合計コスト、および各行の推定行数と推定幅に関する情報を含めます。 - バッファーの使用状況に関する情報を含めるには、 **Buffers** を選択します。 - **Timing** を選択して、起動時間とクエリの各ノードで費やされた時間に関する情報を含めます。 - クエリプランに関する概要情報を含めるには、** Overview** を選択します。	
Commit	トランザクションをコミットします。	Shift+CTRL+M
Rollback	トランザクションをロールバックします。	Shift+CTRL+R
Clear	**Clear** メニューのオプションを使用して、表示内容を消去します。- **Clear Query Window** を選択して、 SQL エディターパネルのコンテンツを消去します。 - ** Clear History** を選択して、**History** タブのコンテンツを消去します。	Accesskey + L
Download as CSV	現在のクエリの結果セットをコンマ区切りリストにダウンロードします。 **設定-> SQL エディター-> CSV 出力** ダイアログボックスを介して CSV 設定を指定できます。	F8
Macros	**Manage Macros** を選択して、マクロを作成、編集、またはクリアします。	

6.1.2 SQL エディターパネル

SQL エディターパネルは、クエリーを手動で提供したり、別のソースからクエリーをコピーしたり、ファイルからクエリーを読み取ることができるワークスペースです。SQL エディターには、構文のカラーリングとオートコンプリート機能があります。

オートコンプリートを使用するには、クエリの入力を開始します。エディターにクエリーの次のオブジェクト名またはコマンドを提案したい場合は、**Control+Space** を押します。たとえば、`\ *SELECT \*FROM\*` 末尾にスペースを入力し、** Control+Space**を押してオートコンプリートオプションのメニューから選択します。

クエリーを入力したら、ツールバーから **Execute/Refresh** を選択します。データベースサーバーは、実行する SQL エディタパネルの完全な内容を受信します。SQL エディターに表示されるコードのセクションのみを実行するには、サーバーに実行するテキストを選択し、**Execute/Refresh** を選択します。

コマンドの実行時にサーバーから返されたメッセージは、 **メッセージ**タブに表示されます。コマンドが成功した場合、[**Messages**] タブに実行の詳細が表示されます。

Edit メニューは、コードのフォーマットとコメント付けに役立ちます。

- オートインデントを使用して、 **Return** を押して、前の行と同じ深さまでテキストをインデントします。
- 2 行以上を選択して **Tab** を押して、インデントテキストをブロックします。
- コード内の SQL スタイルの実装または削除するか、C スタイルのコメント表記を切り替えます。

ツリーから特定のオブジェクトをドラッグして、長いオブジェクト名の入力に費やす時間を節約することもできます。オブジェクト名を含むテキストは、スキーマ名で完全修飾されます。必要に応じて二重引用符が追加されます。ファンクションとプロシージャーの場合、ファンクション名とパラメーター名がクエリツールに貼り付けられます。

6.1.3 [データ出力] パネル

[Data Output] パネルには、最近実行されたクエリによって生成されたデータと統計が表示されます。

[データ出力] タブ

[**Data Output**] タブには、クエリの結果セットがテーブル形式で表示されます。次のことができます。

- 結果セットから選択してコピーします。
- **Execute/Refresh** オプションを使用して、クエリ実行情報を取得し、クエリ実行オプションを設定します。
- **Download as CSV** を選択して、 **Data Output** タブのコンテンツをコンマ区切りファイルとしてダウンロードします。
- SELECT クエリーの結果セットのデータが更新可能な場合は、編集します。

結果セットは次の場合に更新可能です。

- すべての列は、単一のテーブルから直接選択されるか、実際にはテーブルの列ではありませんたとえば、2 つの列の連結。テーブルから直接選択された列のみを編集できます。他の列は読み取り専用です。
- テーブルのすべての主キー列または OID が結果セットで選択されます。

複数回名前が変更または選択された列も読み取り専用です。

注釈: 更新可能なクエリー結果セットを使用するには、 psycopg2 ドライバーバージョン 2.8 以降をインストールする必要があります。

編集可能および読み取り専用の列は、列ヘッダーの鉛筆と鍵のアイコンを使用して識別されます。

更新可能な結果セットは、データの表示/編集モードのデータグリッドに似ており、同じ方法で変更できます。

自動コミットがオフの場合、データは進行中のトランザクションの一部として作成されます。進行中のトランザクションがない場合、新しいトランザクションが開始されます。トランザクションがコミットされない限り、データの変更はデータベースにコミットされません。

保存中にエラーが発生した場合たとえば、 NOT NULL 制約を使用して列に NULL を保存しようとすると、データの変更がセーブポイントにロールバックされ、進行中のトランザクションで以前に実行されたクエリがロールバックされないことを確認します。

[データ出力] パネルに表示される以前のクエリまたはコマンドからのすべての行セットは、別のクエリを呼び出すと破棄されます。別のクエリツールのブラウザタブを開き、以前の結果を利用可能なままにします。

「Explain」タブ

クエリの Explain または Explain Analyze プランを生成するには、ツールバーで **Explain** または **Explain Analyze** を選択します。

メニューから **Explain** および **Explain Analyze** に関連するオプションを選択できます。

注釈: PEM は、JSON 形式で Explain Analyze プランを生成します。

Explain プランが正常に生成されると、 **Explain** パネルの下に 3 つのタブ/パネルが表示されます。

グラフィカルタブ

プランを SVG ファイルとしてダウンロードするには、Explain キャンバスの左上隅にある **Download** を選択します。 **SVG としてのダウンロード** は、Internet Explorer ではサポートされていません。

Explain Analyze に付随するクエリプランは、 **Data Output** タブで利用できます。

[分析] タブ

[**Anarise**] タブには、プランの詳細が表形式で表示され、`explain.depsez.com` で使用可能なものと同様の形式が生成されます。テーブルの各行は、Explain Plan ノードのデータを表します。出力には、ノード情報、排他的タイミング、包含タイミング、実際の行対計画的な行、実際の行、計画的な行、またはループを含めることができます。選択した行の子行は、オレンジ色の点でマークされます。

合計クエリ時間の排他的/包括的タイミングのパーセンテージが次の場合

- 90 を超えると、赤が表示されます
- 50 を超えると、オレンジ色が表示されます
- 10 を超えると黄色が表示されます

プランナーが次の理由で行数実際のバース計画を誤って見積もった場合。

- 10 回、黄色が表示されます

- 100 回、オレンジ色が表示されます
- 1000 回、赤が表示されます

[統計] タブ

[**Statistics**] タブには、2 つのテーブルの情報が表示されます。

- ノードタイプごとの統計は、各ノードタイプが参照された回数を示します。
- テーブルごとの統計は、各テーブルがクエリによって参照された回数を示します。

[メッセージ] タブ

Messages タブを使用して、最近実行されたクエリに関する情報を表示します。サーバーがエラーを返した場合、エラーメッセージは [**メッセージ**] タブに表示され、SQL エディターではエラーの原因となった構文に下線が引かれます。クエリが成功した場合、[**Messages**] タブには、クエリが完了するまでに要した時間と取得された行数が表示されます。

[通知] タブ

[**Notifications**] タブを使用して、PostgreSQL の Listen/Notify 機能を使用して通知を表示します。詳細については、

> PostgreSQL documentation を参照してください。

例

1. 最初のクエリツールセッションで `LISTEN "foo"` を実行します。

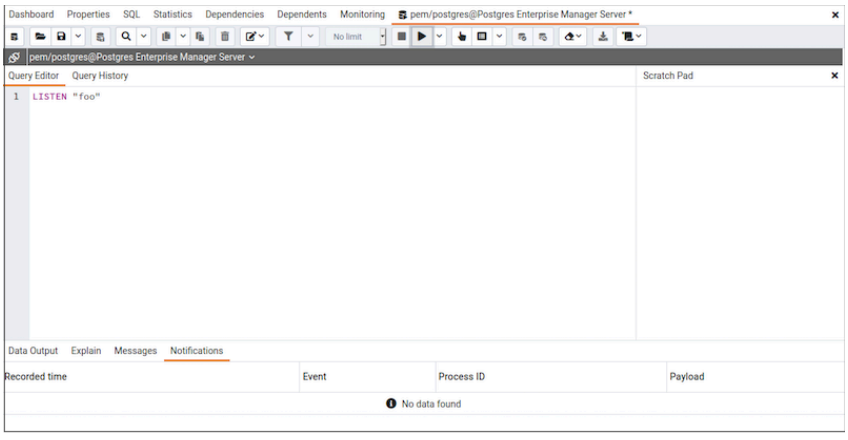


図 1: Query tool - notifications listen

2. 別のクエリツールセッションで、`Notify` コマンドまたは `pg_notify` ファンクションを実行して、ペイロードとともにイベントの通知を送信します。
3. 最初のクエリツールセッションで [**Notification**] タブを観察できます。ここには、チャネルの記録時間、イベント、プロセス ID、およびペイロードが表示されます。

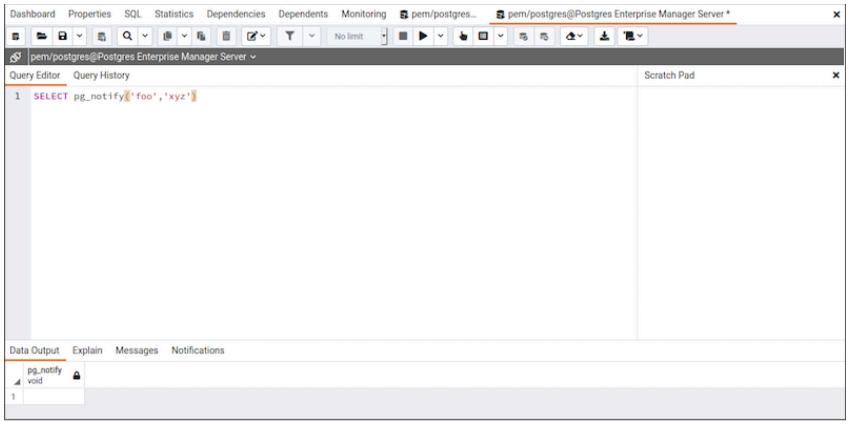


図 2: Query tool - notifications notify

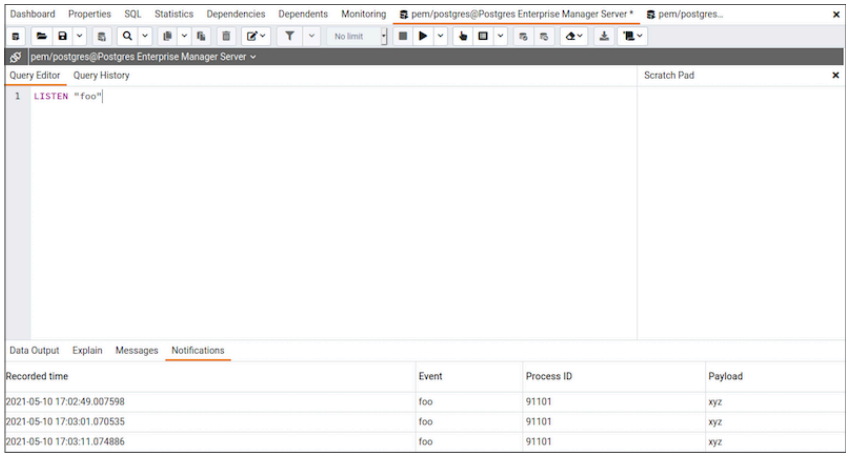


図 3: Query tool - Output Notifications panel

6.1.4 Query History パネル

Query History タブを使用して、現在のセッションのアクティビティを確認します。 [**Query History**] タブには、次のような最近のコマンドに関する情報が表示されます。

- クエリーが呼び出された日時。
- クエリのテキスト。
- クエリーによって返された行数。
- サーバーがクエリーを処理し、結果セットを返すまでに要した時間。
- サーバーから返されたメッセージ [**メッセージ**] タブには記載されていません。
- クエリーのソースツールバーに対応するアイコンで示されます。

pgAdmin によって内部的に生成されたクエリーは、データの表示/編集またはデータの保存操作中に表示または非表示にできます。

Query History タブの内容を消去するには、 **Clear > Clearhistory** を選択します。

クエリ履歴は、クエリツールモードで実行している場合、ユーザーごとに各データベースのセッションを超えて維持されます。View/Edit Data モードでは、履歴は保持されません。デフォルトでは、最後の 20 クエリーがデータベースごとに保存されます。`MAX_QUERY_HIST_STORED` 値をオーバーライドすることにより、`config_local.py` でこれを調整できます。

6.1.5 接続ステータス

Connection Status を使用して、クエリツールでステータスを選択することにより、現在の接続とトランザクションのステータスを表示します。

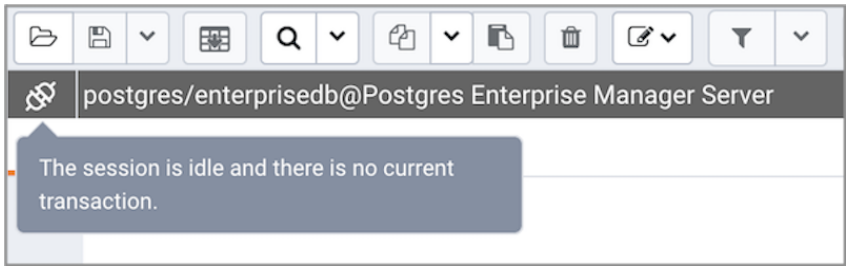


図 4: Query tool - Connection Status button

6.1.6 マクロ

クエリツールマクロを使用すると、単一のキーを押すだけで事前定義された SQL クエリーを実行できます。事前定義されたクエリーには、プレースホルダー$SELECTION$ を含めることができます。マクロが実行されると、プレースホルダーは、クエリツールの [クエリエディター] ペインで現在選択されているテキストに置き換えられます。

マクロを作成するには

1. クエリツールで、 **Macros > Manage Macros** を選択します。
2. 使用するキーを選択します。マクロとクエリの名前を入力します。
3. オプションで、選択プレースホルダーを含めます。
4. **Save** を選択します。

マクロをクリアするには、[Manage Macros] ダイアログボックスでマクロを選択し、 **Clear** を選択します。プロンプトに **Yes** と応答します。

すべてのマクロをクリアするには、 **Key** の横にある **Clear** を選択します。プロンプトに **Yes** と応答します。

マクロを実行するには、ショートカットキーを選択するか、 **Macros** メニューから選択します。

6.1.7 UI の詳細についてのアプリ内ヘルプへのアクセス

アプリ内ヘルプは、メニュー項目、ダッシュボードのチャートとグラフ、ダイアログボックスのオプションを含む PEM Web インターフェイスに関する包括的な詳細を提供します。アプリ内のヘルプには、データベースオブジェクトを作成する手順も含まれています。

- ヘルプのランディングページにアクセスするには、 **Help > Online help** を選択します。
- ダイアログボックスの状況依存ヘルプにアクセスするには、 **?** を選択します。

6.2 Using the Schema Diff tool

Schema Diff は、2 つのデータベーススキーマ間のスキーマオブジェクトを比較します。**Tools** メニューを使用して、Schema Diff にアクセスします。

スキーマ差分を使用すると、次のことができます。

- データベーススキーマソースからターゲットへを比較および同期します。
- データベーススキーマ間の違いを視覚化します。
- ターゲットスキーマオブジェクトの SQL ステートメントの差異をリストします。
- 同期スクリプトを生成します。

選択パネルを開くには、 **Tools > Schema Diff** を選択します。比較するソースおよびターゲットサーバー、データベース、およびスキーマを選択します。オブジェクトを選択したら、[**Compare**] を選択します。

注釈: ソースデータベースとターゲットデータベースは、同じメジャーバージョンを使用する必要があります。

スキーマ差分の複数のコピーを個々のタブで開くことができます。新しいブラウザタブでスキーマ差分を開くかどうかを指定するには、[設定] ダイアログボックスを選択します。**新しいブラウザタブで開く** を **true** に設定します。

[Schema Diff] パネルは、 Object Comparison パネルと DDL Comparison パネルの 2 つのパネルに分割されています。

6.2.1 Schema Diff Object Comparison パネル

[Object Comparison] パネルでは、比較する同じメジャーバージョン、データベース、およびスキーマのソースサーバーとターゲットサーバーを選択できます。接続されているか切断されているかにかかわらず、ブラウザツリーの下にリストされているサーバーを選択できます。接続されていないサーバーを選択した場合、そのサーバーを使用する前にパスワードを入力する必要があります。

比較するデータベースを選択します。データベースは同じでも異なっていても、同じまたは別のサーバーからのものでもかまいません。

比較するソーススキーマとターゲットスキーマを選択します。

サーバー、データベース、およびスキーマを選択したら、 **Compare** を選択して比較結果を取得します。

ファンクション 、 **マテリアライズドビュー** 、 **テーブル** 、**トリガーファンクション** 、 **プロシージャ** 、および **ビュー s** のリストを使用して、すべての DDL ステートメントを表示しますスキーマオブジェクト。

スキーマオブジェクトをフィルタリングするには、[Object Comparison] パネルの右上隅にある **Filter** を選択します。次の基準に従ってスキーマオブジェクトをフィルタリングします。

- **同一**—同じ SQL ステートメントを使用して両方のスキーマでオブジェクトが見つかった場合、比較結果は同じです。
- **異なる**—オブジェクトが異なる SQL ステートメントを使用して両方のスキーマで見つかった場合、比較結果は異なります。
- **ソースのみ**—オブジェクトがソーススキーマでのみ見つかり、ターゲットスキーマで見つからない場合、比較結果はソースのみです。
- **ターゲットのみ**—オブジェクトがターゲットスキーマでのみ見つかり、ソーススキーマで見つからない場合、比較結果はターゲットのみです。

Object Comparison パネルでスキーマオブジェクトのいずれかを選択して、そのオブジェクトの DDL ステートメントを DDL Comparison パネルに表示します。

6.2.2 Schema Diff DDL Comparison パネル

[DDL Comparison] パネルには、3 つの列が表示されます。

- 最初の列には、ソーススキーマからのオブジェクトの DDL ステートメントが表示されます。
- 2 番目の列には、ターゲットスキーマからのオブジェクトの DDL ステートメントが表示されます。
- 3 番目の列には、ターゲットスキーマオブジェクトの SQL ステートメントの差異が表示されます。

SQL ステートメントの違いを確認するには、すべてのスキーマオブジェクトの DDL ステートメントを確認します。

Schema Diff ツールは、ターゲットスキーマオブジェクトで見つかった差分を使用して SQL スクリプトを生成できます。SQL スクリプトは、ターゲットスキーマオブジェクトをソーススキーマオブジェクトの SQL ステートメントと比較します。スクリプトを生成するには

1. Object Comparison パネルで、スキーマオブジェクトのチェックボックスをオンにします。
2. **Generate Script** を選択します。

新しいタブでクエリツールを開き、クエリエディタで SQL ステートメントの違いを表示するには

1. スキーマオブジェクトを選択します。
2. **Generate Script** を選択します。

スキーマオブジェクトを選択して、[DDL 比較] パネルで生成された差分を確認したが、スキーマオブジェクトのチェックボックスを選択しない場合、PEM は新しいタブでクエリツールを開きます。Query ツールは、 SQL ステートメントの差異をクエリエディターに表示します。

[DDL Comparison] パネルで生成された差分をコピーするには、 **Copy** を選択します。

ターゲットスキーマに SQL ステートメントを適用して、スキーマを同期します。

6.3 Using the ERD tool

Entity-Relationship DiagramERD ツールは、データベーステーブル、列、および相互関係のグラフィカル表現を提供するデータベースデザインツールです。ERD は、データベース管理者がデータベースを開発および保守する際に従うための十分な情報を提供できます。ERD ツールにアクセスするには、 **Tools > ERD Tool** を選択します。ERD ツールを使用すると、次のことができます。

- データベーステーブルとその関係をデザインおよび視覚化します。
- 図にノートを追加します。
- テーブルとリンクを揃えて、すっきりした外観にします。
- 図を保存し、後で開いて作業を続行します。
- データベースデザインからすぐに実行できる SQL を生成します。
- 既存のデータベースのデータベース図を生成します。

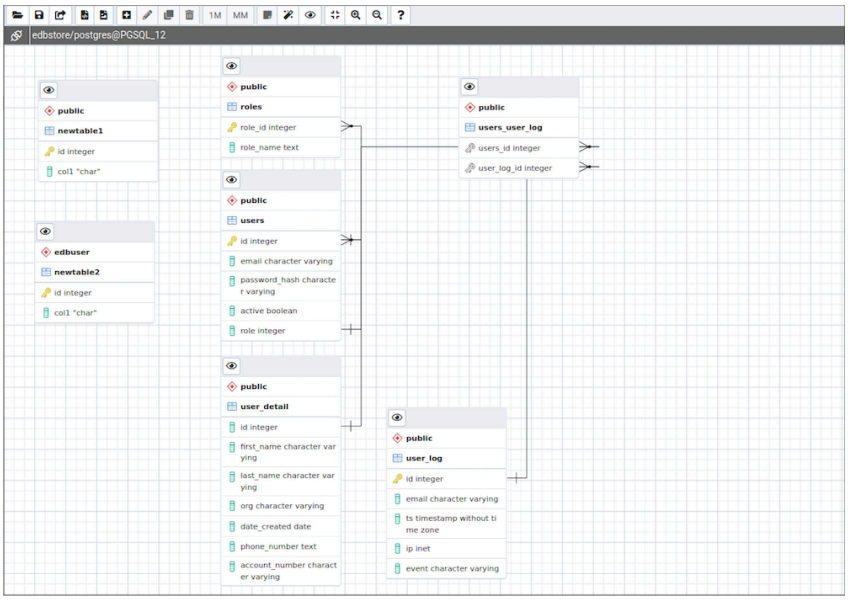


図 5: ERD Tool

ERD ツールの複数のコピーを個々のタブで同時に開くことができます。ERD ツールのコピーを閉じるには、タブバーの右上隅にある **X** を選択します。

6.3.1 ツールバー

ERD ツールツールバーは、頻繁に実行されるタスクへのショートカットを提供する状況依存のアイコンを使用します。

6.3.2 ファイルオプション

Icon	Behavior	Shortcut
Open File	保存した図をロードします。	Ctrl + O
Save	保存した図に変更を保存するか、図をファイルに保存します。	Ctrl + S
Save as	新しいダイアログボックスを開き、図を保存する新しい場所を指定します。	Ctrl + Shift + S

6.3.3 エクスポートオプション

Icon	Behavior	Shortcut
Generate SQL	図の DDL SQL を生成し、生成された SQL を実行可能な状態でクエリツールを開きます。	Option + Ctrl + S
Download image	ERD 図を画像形式で保存します。	Option + Ctrl + I

6.3.4 編集オプション

Icon	Behavior	Shortcut
Add table	新しいテーブルを図に追加します。テーブルダイアログボックスに、テーブルの詳細を入力します。	Option/Alt + Ctrl + A
Edit table	図で選択したテーブルを編集します。テーブルダイアログボックスに、テーブルの詳細を入力します。	Option/Alt + Ctrl + E
Clone table	完全なテーブル構造のクローンを作成し、自動生成された名前で名前を付けて、図に配置します。	Option/Alt + Ctrl + C
Drop table/link	選択したテーブルまたはリンクをドロップします。	Option/Alt + Ctrl + D

6.3.5 テーブルリレーションシップオプション

Icon	Behavior	Shortcut
1M	1 対多のリレーションシップダイアログボックスを開き、2 つのテーブル間のリレーションシップを追加します。選択したテーブルは参照テーブルであり、リンクの `many`エンドポイントがあります。	Option/Alt + Ctrl + O
MM	多対多のリレーションシップダイアログボックスを開き、2 つのテーブル間のリレーションシップを追加します。このオプションは、2 つの関連するテーブルの選択した列からリンクテーブルを作成します。	Option/Alt + Ctrl + M

6.3.6 ユーティリティオプション

Icon	Behavior	Shortcut
Add/Edit note	データベースの設計中に、テーブルノードにメモを作成します。	Option/Alt + Ctrl + N
Auto align	すべてのテーブルとリンクを揃えて、見た目をすっきりさせます。	Option/Alt + Ctrl + L
Show details	列の詳細を切り替えます。いくつかの列の詳細を表示することを選択することもできます。	Option/Alt + Shift + D

6.3.7 ズームオプション

Icon	Behavior	Shortcut
Zoom to fit	自動的にズームイン/アウトし、すべてのテーブルをビューに合わせます。	Option/Alt + Shift + F
Zoom in	図を拡大します。	Option/Alt + Shift + "+"
Zoom out	図をズームアウトします。	Option/Alt + Shift + "-"

6.3.8 テーブルダイアログ

テーブルダイアログボックスを使用すると、次のことができます。

- テーブル構造の詳細を変更します。
- 既存のテーブルを編集するか、新しいテーブルを追加します。
- さまざまなフィールドについては、PEM クライアントのオンラインヘルプのテーブルダイアログを参照してください。

6.3.9 テーブルノード

テーブルノードには、テーブルの詳細がグラフィカル表現で表示されます。

- トップバー
  - 列の詳細を表示するには、 **詳細トグル** を選択します。
  - ノートを追加すると、 **Note** が有効になります。ノートを変更するには **Note** を選択します。
- 最初の行には、例の `public` のように、テーブルのスキーマ名が表示されます。
- 2 行目には、例の `users` のように、テーブルの名前が表示されます。
- 残りの行には、テーブルの列名とそのデータ型が表示されます。

  - 例の `id integer` のように、主キーである列がロックキーとして表示されます。
  - 他の列は、例の `email character varying` のように、緑色の列として表示されます。

- ノードを選択して、キャンバスにドラッグします。
- テーブルダイアログボックスを開き、テーブルの詳細を変更するには、テーブルノードをダブルクリックするか、ツールバーの **Edit** を選択します。さまざまなフィールドについては、PEM クライアントのオンラインヘルプのテーブルダイアログを参照してください。

6.3.10 1 対多のリンクダイアログボックス

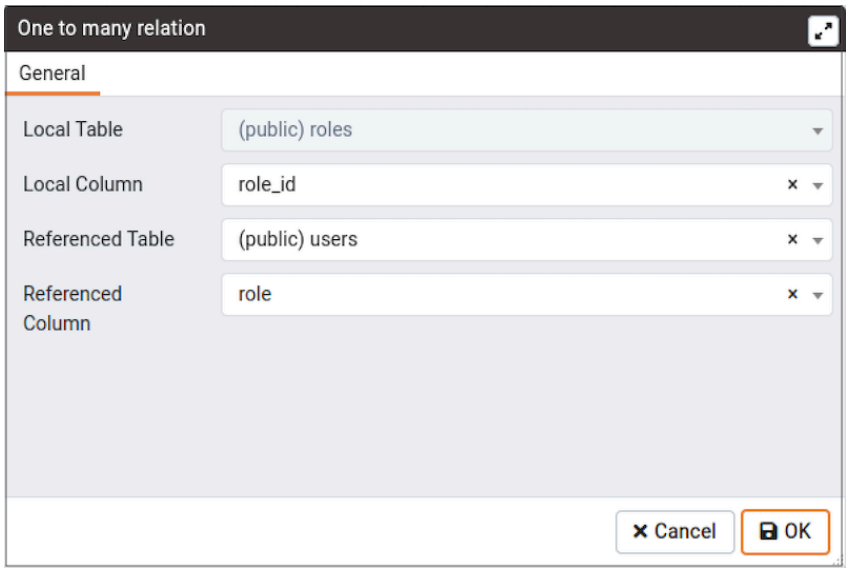


図 6: ERD Tool - One-to-Many Dialog Box

1 対多のリンクダイアログボックスを使用すると、次のことができます。

- 2 つのテーブル間に外部キー関係を追加します。
- 別のテーブルを参照するローカルテーブルを選択します。ローカルテーブルには many エンドポイントがあります。
- 別の列を参照するローカル列を選択します。
- 参照されるテーブルを選択します。参照テーブルには one エンドポイントがあります。
- 参照される列を選択します。

6.3.11 多対多のリンクダイアログボックス

多対多のリンクダイアログボックスを使用すると、次のことができます。

- 2 つのテーブル間に多対多のリレーションシップを追加します。
- 2 つのテーブルから派生するリンクされた列を含むリレーションテーブルを作成します。
- リンクされる最初のテーブルである左側のテーブルを選択します。左側のテーブルには、新しいリレーションテーブルとのリンクの one エンドポイントがあります。
- 最初のテーブルの列である左側のテーブル列を選択します。左側のテーブルの列は主キーです。
- リンクされている 2 番目のテーブルである右テーブルを選択します。右テーブルには、新しいリレーションテーブルとのリンクの one エンドポイントがあります。
- 2 番目のテーブルの列である右テーブルの列を選択します。右テーブルの列は主キーです。

6.3.12 テーブルリンク

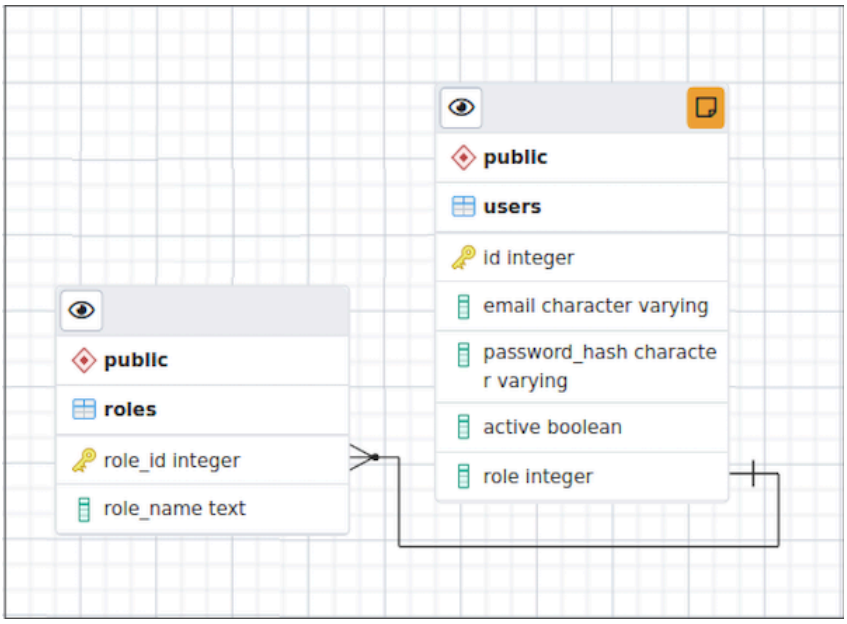


図 7: ERD Tool - Table Link

テーブルリンクは、テーブル間の関係を示します。

- リンクの単線のエンドポイントは、参照である列を示します。
- リンクの 3 本線のエンドポイントは、参照する列を示します。
- 参照または参照している列のいずれかがテーブルから削除されると、リンクがドロップします。
- リンクを選択して、キャンバスにドラッグします。

6.3.13 テーブルノート

テーブルにノートを追加するには

1. ツールバーで、 **Add/Edit note** を選択します。
2. ウィンドウで、メモを入力します。

ノートがテーブルに追加されると、テーブルノードで **Note** が有効になります。ノートを確認または更新するには、 **Note** を選択します。