



PEM Upgrade and Migration

リリース 8.3.0

EnterpriseDB

2021 年 12 月 25 日

目次

第 1 章	Upgrading a PEM Installation	3
1.1	Upgrading a PEM Installation on Windows Host	3
1.1.1	Windows ホスト上の PEM エージェントのアップグレード	3
1.1.2	Windows ホスト上の PEM サーバーのアップグレード	8
1.1.3	Windows ホストでの SQL プロファイラーのアップグレード	16
1.2	Upgrading a PEM Native Package Installation on a Linux Host	17
1.2.1	Linux ホストで PEM インストールをアップグレードするための前提条件	18
1.2.2	PEM エージェントネイティブパッケージインストールのアップグレード	19
1.2.3	PEM サーバーネイティブパッケージインストールのアップグレード	20
1.2.4	SQL プロファイラーネイティブパッケージインストールのアップグレード	21
1.3	Creating a PEM Repository on an Isolated Network	23
1.4	Upgrading a Graphical PEM Installation on a Linux Host	24
1.4.1	Linux ホスト上の PEM インストールをアップグレードするための前提条件	24
1.4.2	CentOS または RHEL ホストでの PEM エージェント（グラフィカルインストール） のアップグレード	25
1.4.3	グラフィカルインストーラーでインストールされた PEM サーバーのアップグレード	28
1.4.4	グラフィカルインストーラーを使用した SQL プロファイラーインストールのアップ グレード	31
1.5	Configuring a PEM Server on a Linux Host	33
第 2 章	Upgrading the PEM Backend Postgres Database	35
第 3 章	Moving the Postgres Enterprise Manager Server	45
第 4 章	Troubleshooting	55
4.1	PEM サーバーの再構成	55

このガイドは、Postgres Enterprise Manager (PEM) コンポーネントのアップグレード処理に関する詳細情報を提供します。

- **** PEM インストールのアップグレード****-このセクションでは、PEM サーバー、PEM エージェント、および SQL プロファイラーをメジャーバージョンから別のバージョン（6.0 から 7.14）にアップグレード処理する方法について説明します。
- **バックエンドデータベースのアップグレード**-このセクションでは、同じバージョンの PEM サーバーを維持しながら、バックエンドデータベースのアップグレード処理に関する詳細情報を提供します。
- **** PEM サーバーの移動**** -このセクションでは、あるホストから別のホストへの PEM サーバーの移動に関する詳細情報を提供します。
- **トラブルシューティング**-このセクションでは、PEM のアップグレード中に発生する可能性のあるエラーのトラブルシューティングに関する詳細情報を提供します。

このドキュメントでは、Postgres という用語を使用して、PostgreSQL、Advanced Server データベース、または EDB Postgres Extended（旧称 2ndQPostgres）を意味します。

第1章 Upgrading a PEM Installation

PEM インストールのアップグレード処理プロセスはプラットフォーム固有です。Windows 用の PEM グラフィカルインストーラを使用して、Windows ホスト上の PEM エージェントまたはサーバーを更新でき Windows。PEM 7.8 リリースより前は、グラフィカルインストーラまたは RPM を使用して、PEM エージェントまたはサーバーを Linux にインストールできました。PEM バージョン 7.8 以降、Linux 用の PEM グラフィカルインストーラは廃止されました。Linux ホスト上の PEM エージェントまたはサーバーを下位バージョンから PEM 7.9 以降のバージョンに更新するには、ネイティブパッケージソフトを使用する必要があります。

PEM および SQL Profiler インストーラーと RPM へのリンクは、[EDB website](#) で入手できます。

1.1 Upgrading a PEM Installation on 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、次のオーダーで新しいバージョンの PEM コンポーネントインストーラーを呼び出すだけです。

1. PEM サーバーホストの各監視対象ノード **except** で PEM エージェントインストーラを起動します。
PEM サーバーインストーラを起動します。このインストーラは、**both** を PEM サーバーおよび PEM サーバーホスト上にある PEM エージェントにアップグレードします。

インストール中、コンポーネントインストーラ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処理した後、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。このステップはプラットフォーム固有です。

以下のセクションでは、Windows ホストでのアップグレードプロセスについて順を追って説明します。

1.1.1 Windows ホスト上の 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最新の PEM エージェントにアップグレードするには、エージェントがモニタリングしているシステムに新しいバージョンの PEM エージェントインストーラをダウンロードして呼び出します。

ダウンロードしたインストーラーのアイコンを右クリックして、**Run as Administrator** を選択すると、インストーラを起動できます。PEM Agent Setup Wizard が開き、お迎えします。

Next をクリックして続行する前に、**License Agreement** を読んで受け入れます。

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、インストールされているバージョンをアップグレードします。**Next** をクリックして続行します。

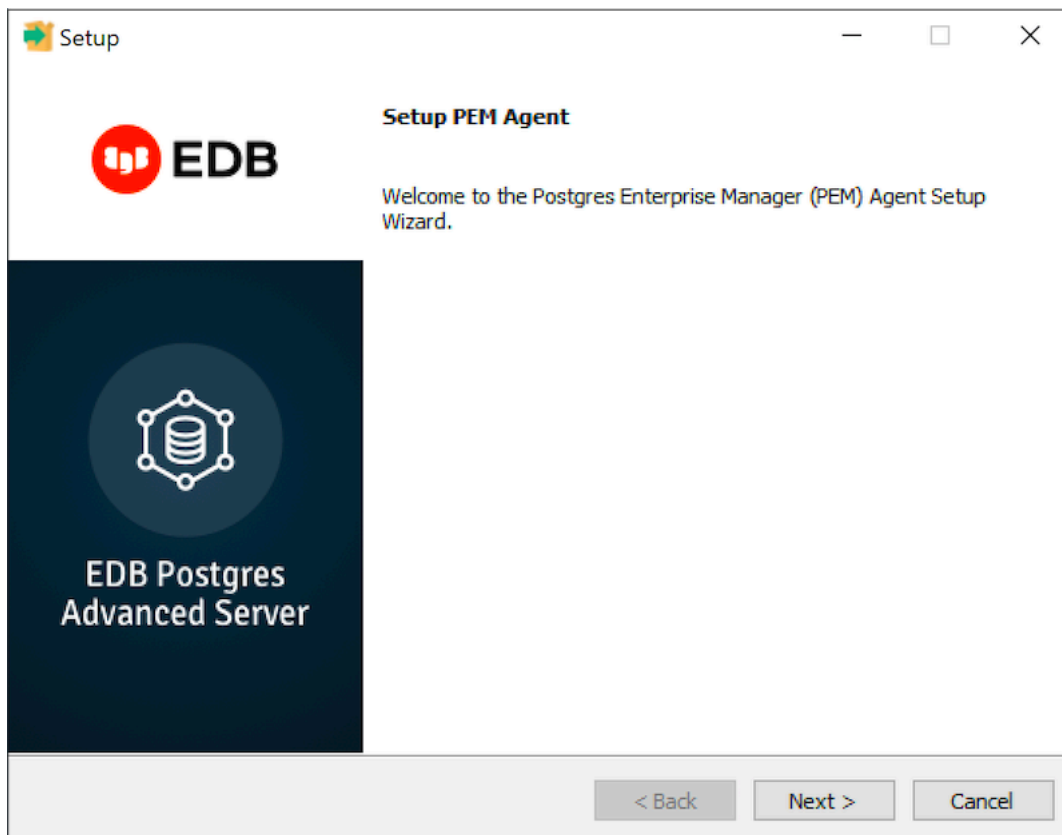


図 1: The PEM Agent Installer - Welcome Window

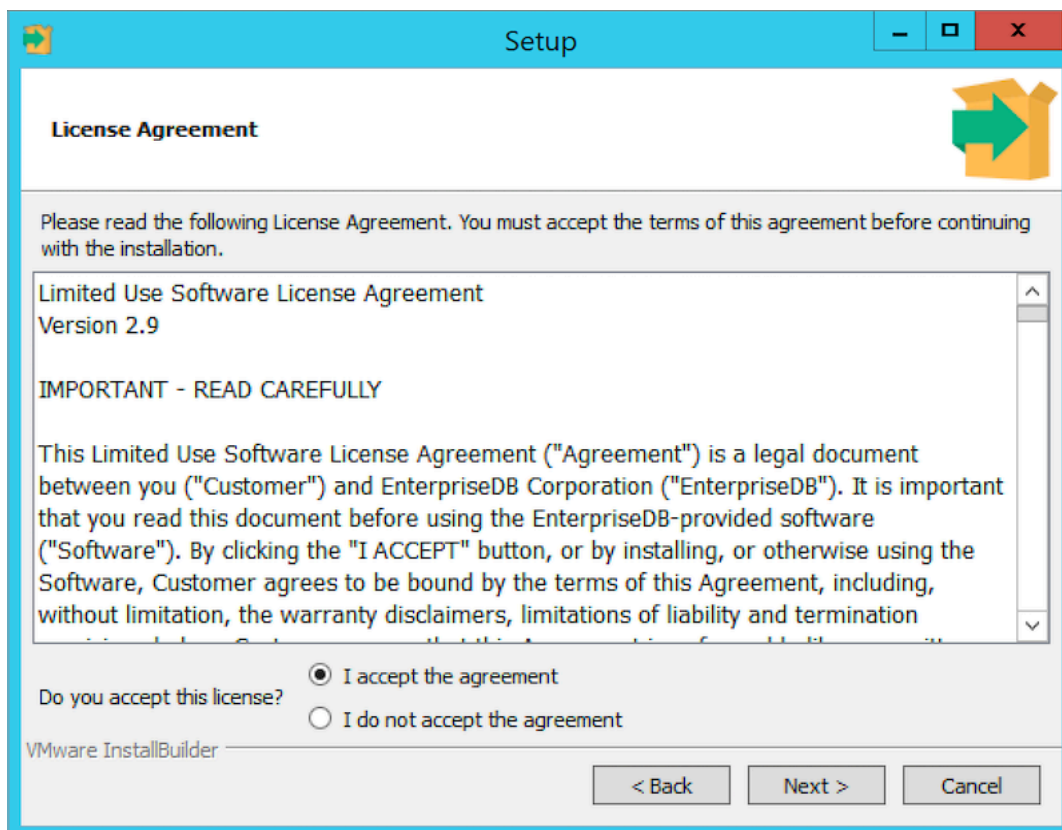


図 2: The PEM Agent Installer - License Agreement

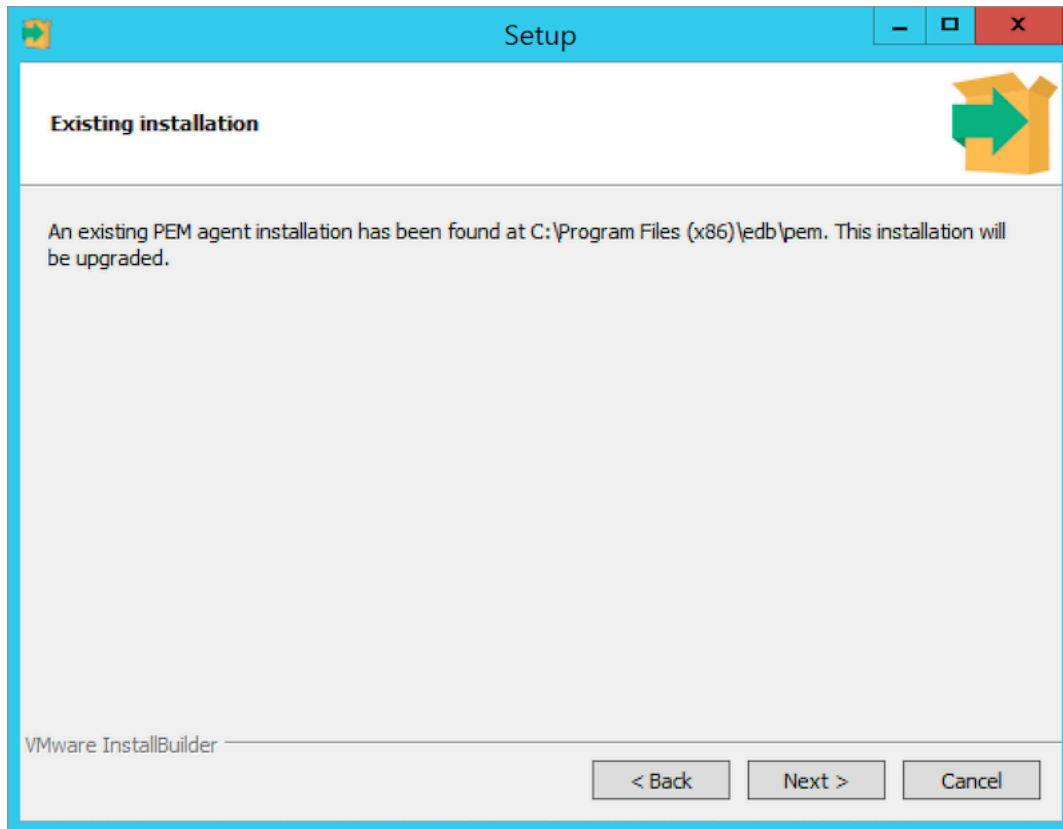


図 3: The PEM Agent Installer - Existing Installation Location

pemAgent service account ダイアログでは、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を求められる場合があります。プロンプト。

プロンプトが表示されたら、パスワードを入力し、**Next** をクリックして続行します。

Ready to Install ダイアログにインストールの開始が通知されたら、**Next** をクリックして続行します。PEM エージェントが最新バージョンにアップグレードされます。

セットアップウィザードに進行状況バーが表示され、インストールされている各コンポーネントが通知されます。

PEM Agent Setup Wizard は、インストールが完了すると通知します。**Finish** をクリックしてウィザードを終了し、ウィンドウを閉じます。

インストールが完了すると、ウィンドウがポップアップしてマシンをリスタートします。**Yes** をクリックして、マシンと PEM エージェントをリスタートします。

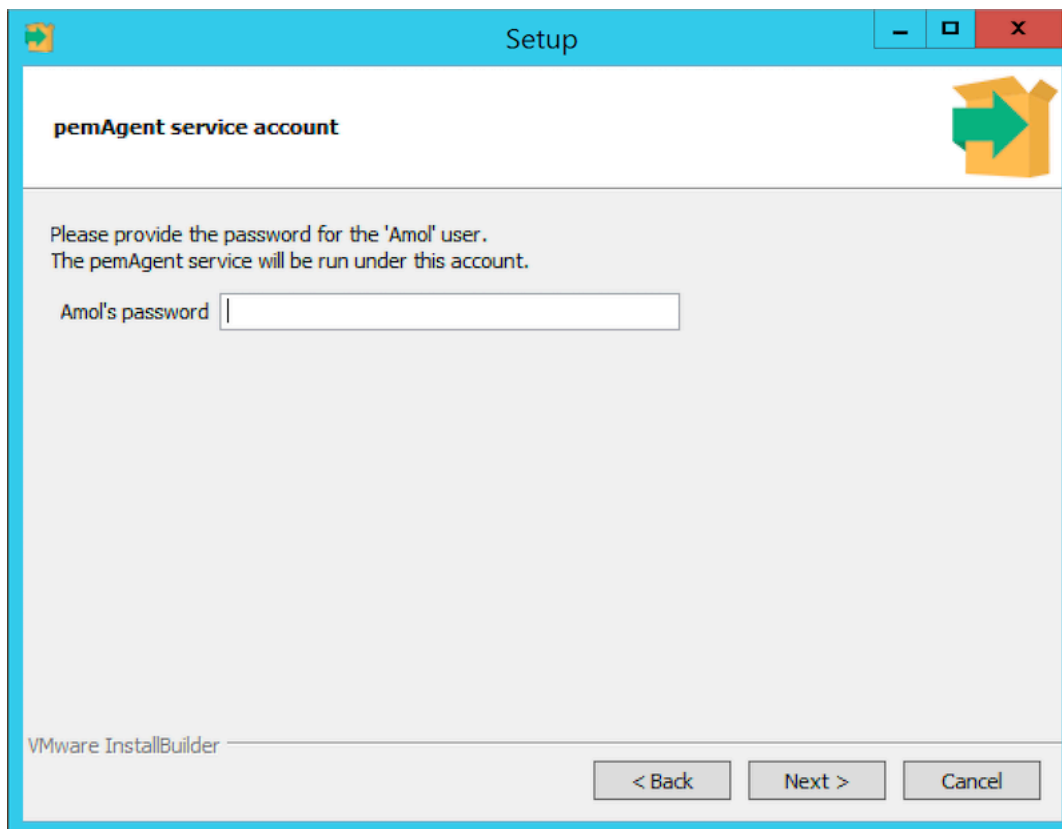


図 4: The PEM Agent Installer - pemAgent Service Account password

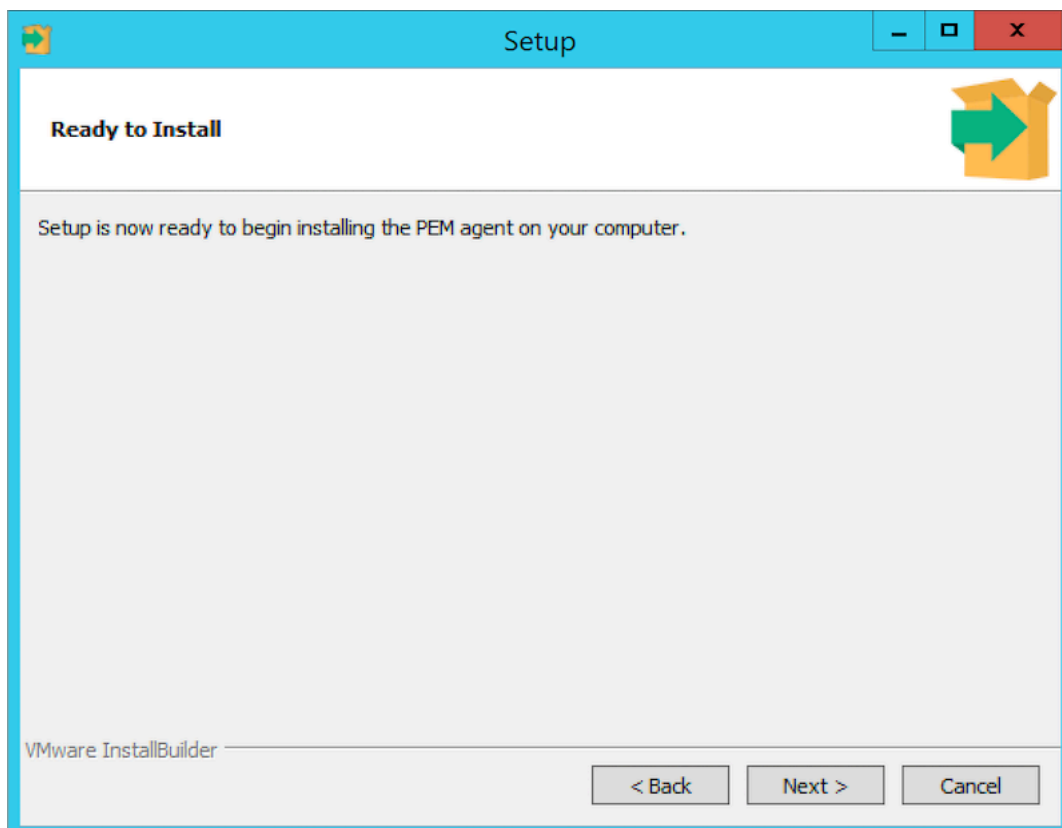


図 5: The PEM Agent Installer - Ready to Install

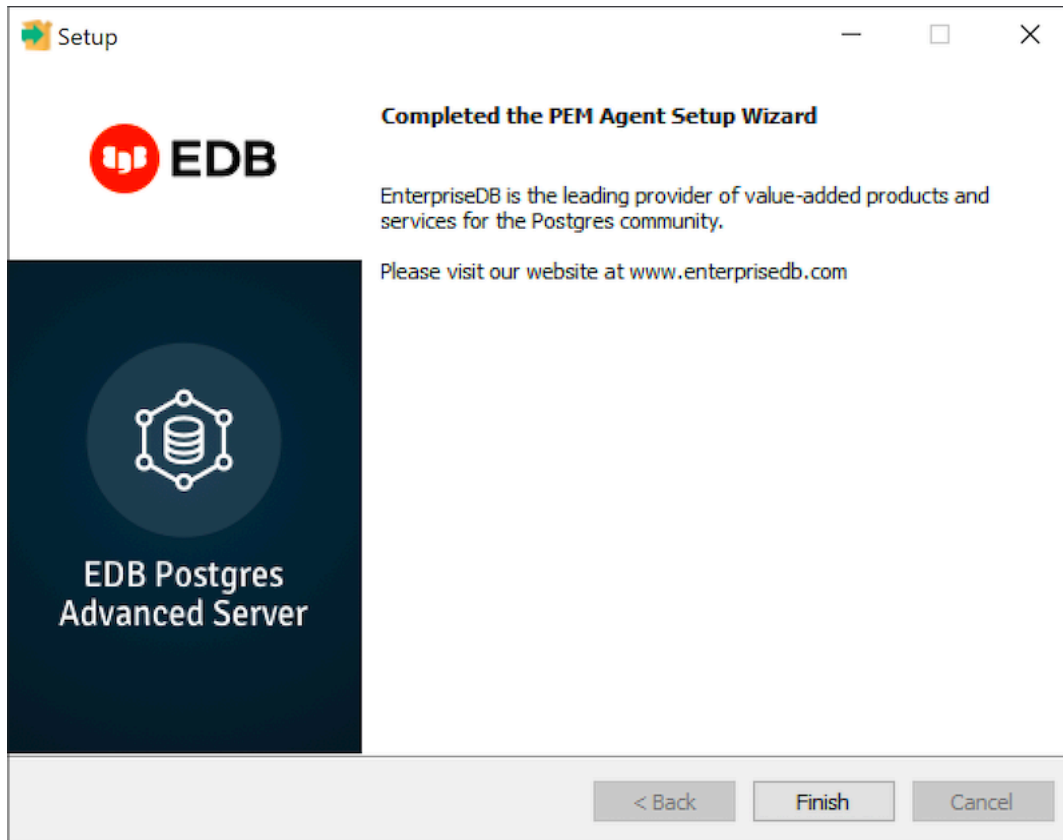


図 6: The PEM Agent Installer - Finish

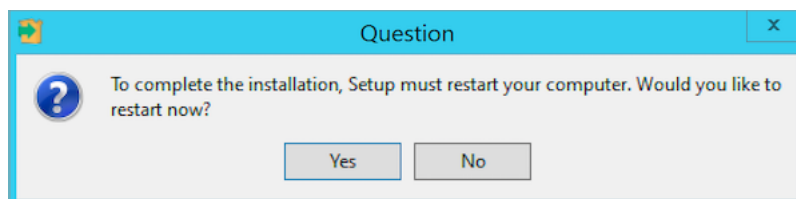


図 7: The PEM Agent Installer - Restart pop-up

1.1.2 Windows ホスト上の PEM サーバーのアップグレード

PEM サーバーインストーラは、PEM サーバーのメジャーバージョン間の直接アップグレード処理を容易にします。最初にバージョン 6.0 にアップグレード処理しなくても、バージョン 5.0 からバージョン 7.16 に直接アップグレードできます。

ダウンロードしたインストーラのアイコンを右クリックして、**Run as Administrator** を選択すると、インストーラを起動できます。

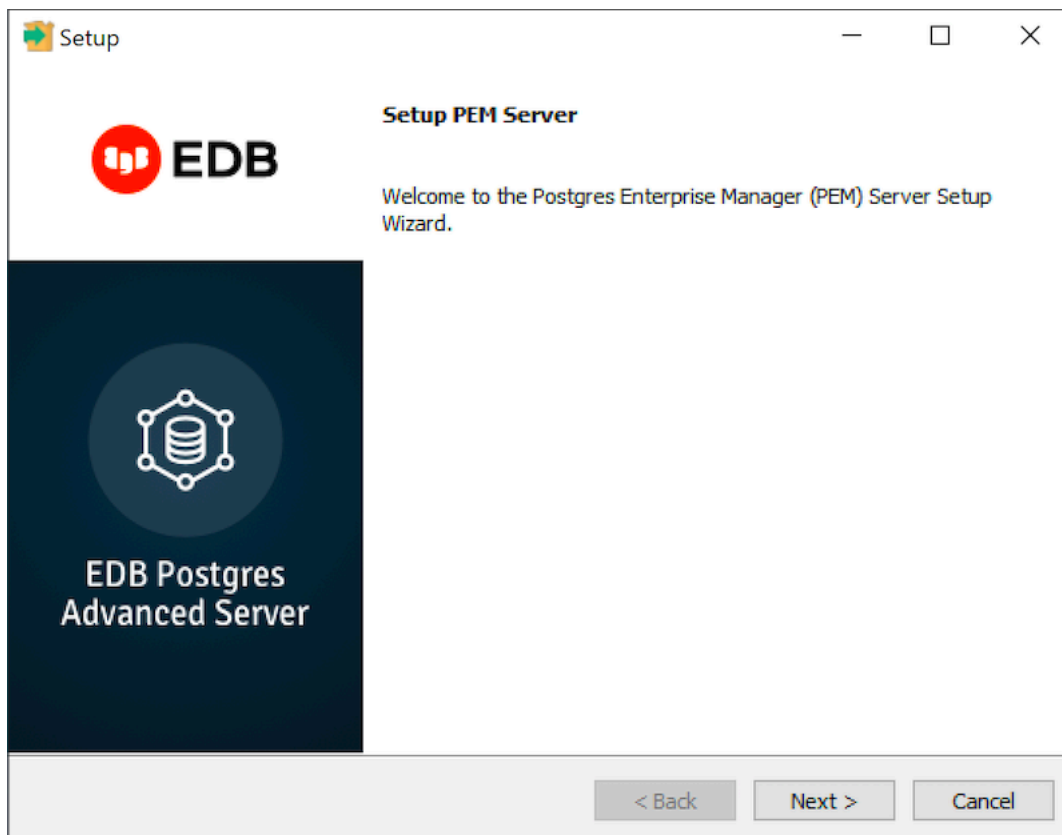


図 8: The PEM Server Installer - Welcome Window

イメージに示すように、PEM Server Setup Wizard はあなたを歓迎します。**Next** をクリックして、**License Agreement** に進みます。

PEM Server setup wizard は、プロンプトを受け入れるように求めます。使用許諾契約を確認した後、**I accept the agreement** の横にあるラジオボタンをオンにし、**Next** をクリックして **Existing installation dialog** に進みます。

ウィザードは、既存の PEM サーバーのインストールについて PEM サーバーホストをチェックします。ウィザードが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** をクリックして続行します。

PEM サーバーをアップグレード処理する前に、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。サポートするコンポーネントが欠落している場合、または新しい PEM インストールをサポートしないバージョンの場合、PEM インストール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、必要なインストーラを起動します。

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、マシンをリスタートするかどうかを尋ねるウィンドウがポップアップ表示されます。

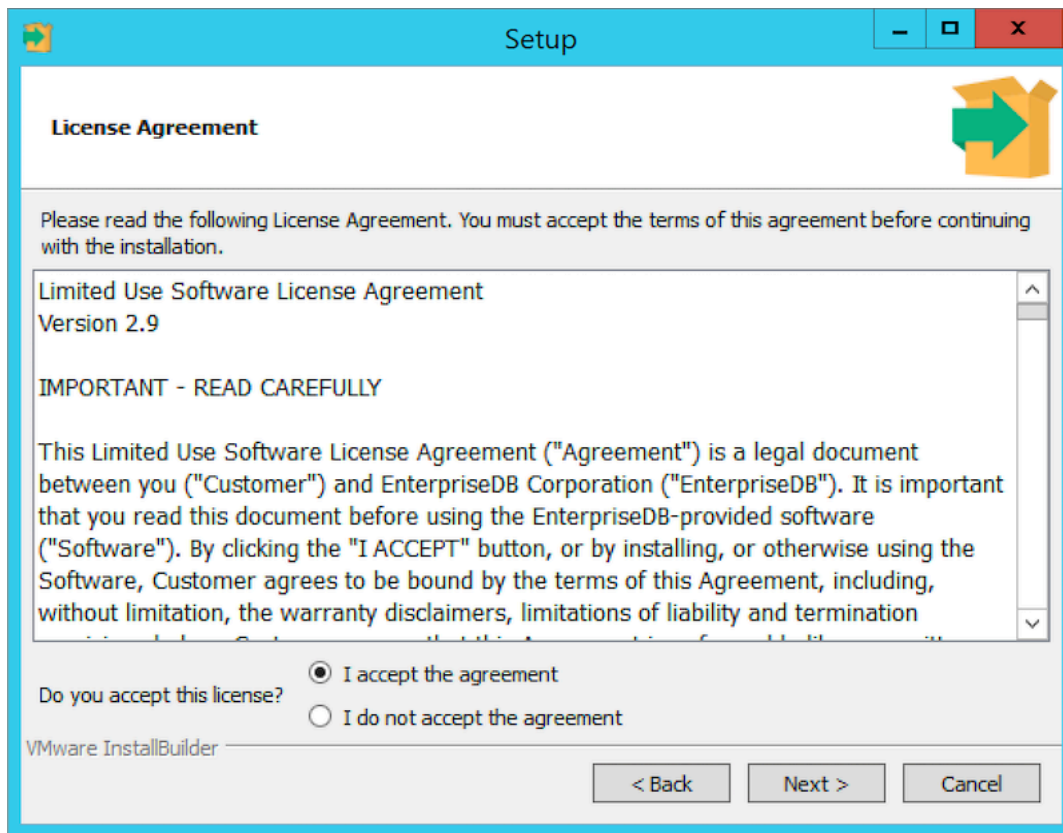


図 9: The PEM Server Installer - License Agreement

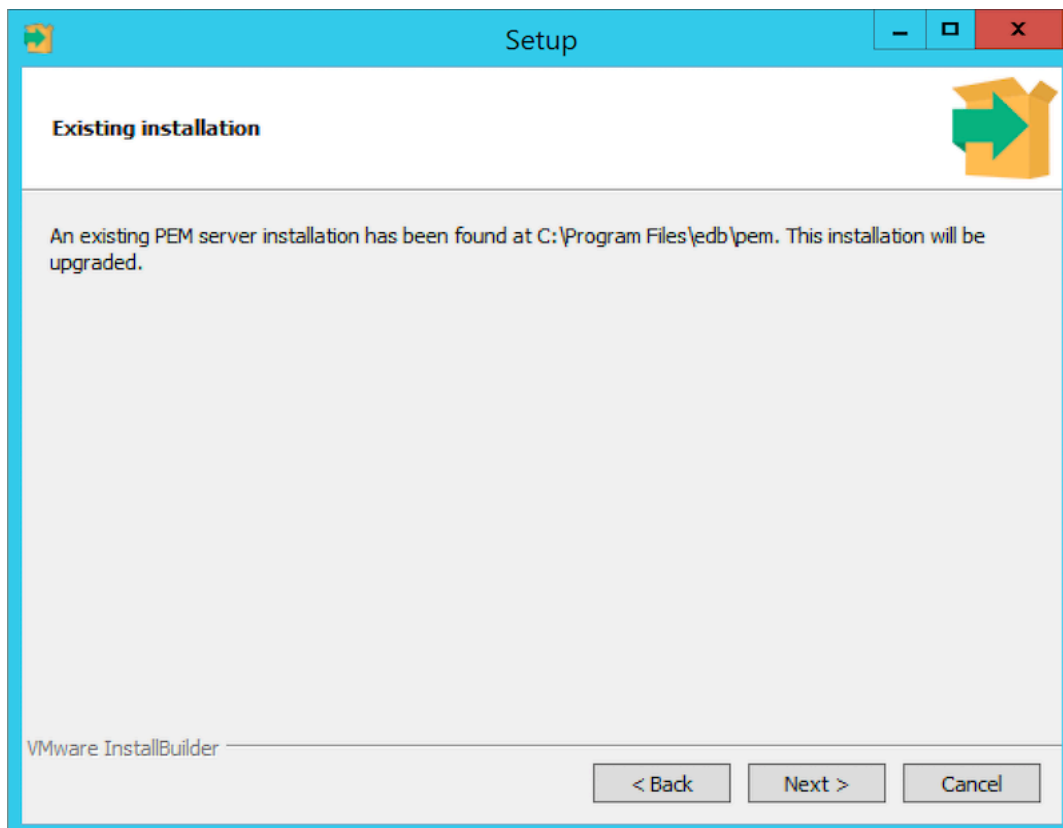


図 10: The PEM Server Installer - Existing PEM Server Location

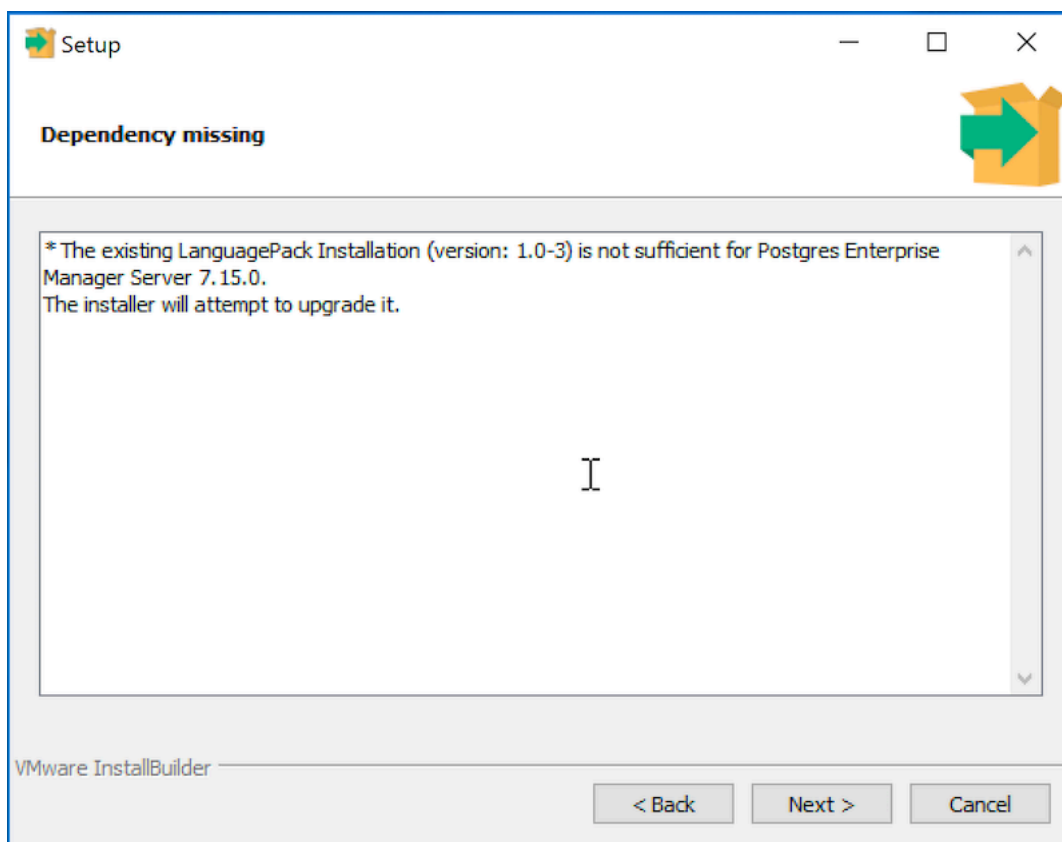


図 11: The PEM Server Installer - Missing Dependencies

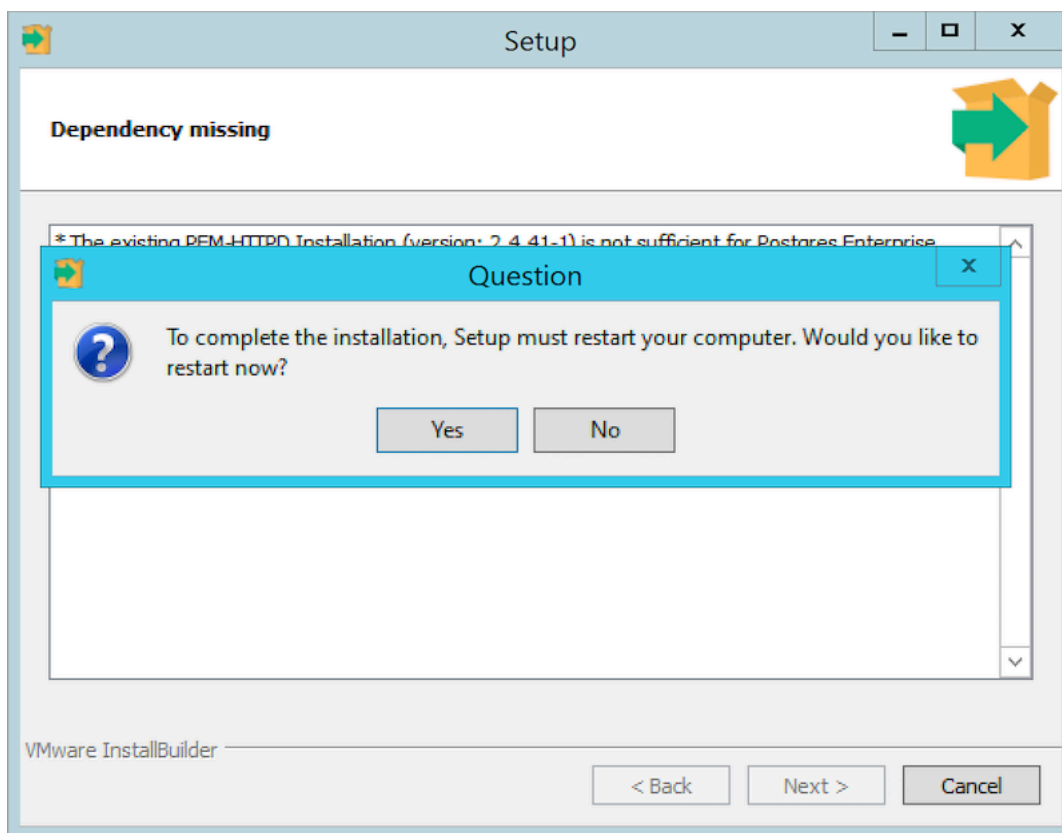


図 12: The PEM Server Installer - Restart pop-up

No をクリックして、アップグレードプロセスを続行します。

次に、ウィザードは Database Server Installation Details ダイアログを開き、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の接続資格情報の入力をお願いします。提供：

- User フィールドのデータベーススーパーユーザの名前。
- Password フィールドのデータベーススーパーユーザに関連付けられたパスワード。

Next をクリックして続行します。

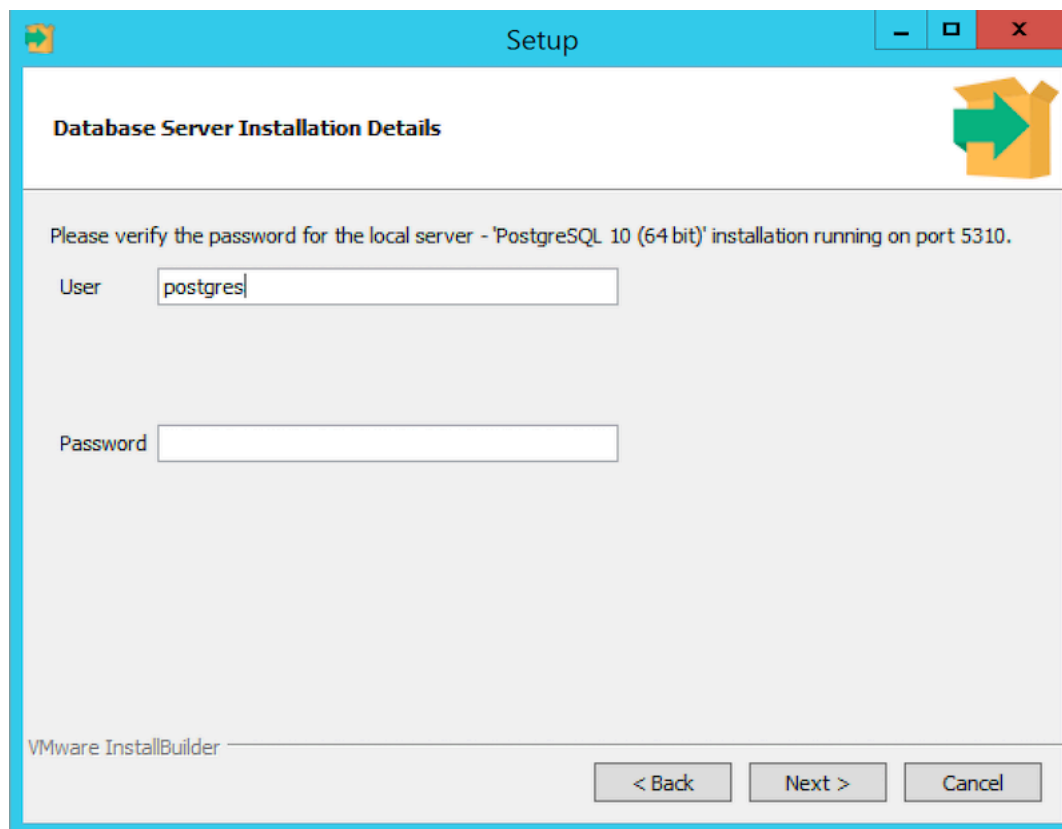


図 13: The PEM Server Installer - Database Server Installation Details

pemAgent service account ダイアログでは、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を求められる場合がありますプロンプト。

プロンプトが表示されたら、パスワードを入力し、Next をクリックして続行します。

Ready to Install ダイアログは、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通知します。Next をクリックして、インストールをスタートします。

インストール中、進行状況バーに更新の進行状況が表示されます。

PEM サーバー（および PEM サーバーと同じホスト上にあるエージェント）をアップグレード処理し、Web サービスを構成した後、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 クライアントで 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。

OK をクリックして、Info ポップアップを閉じます。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。

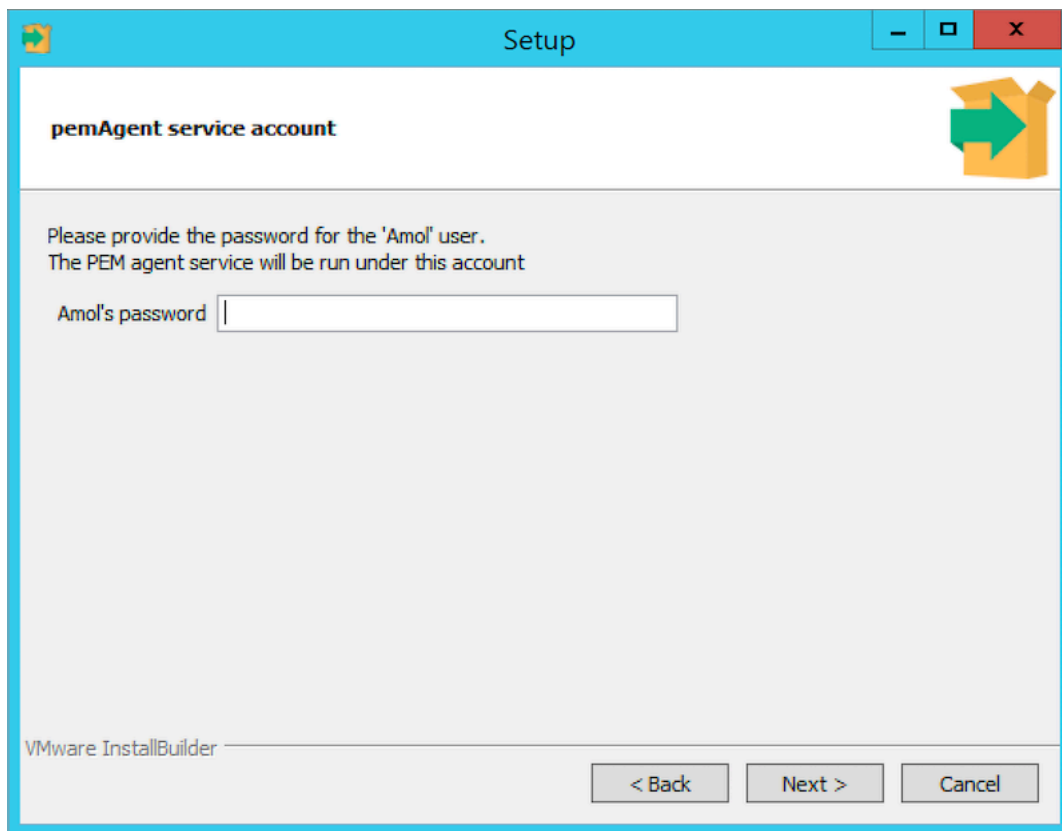


図 14: The PEM Server Installer - pemAgent Service Account password

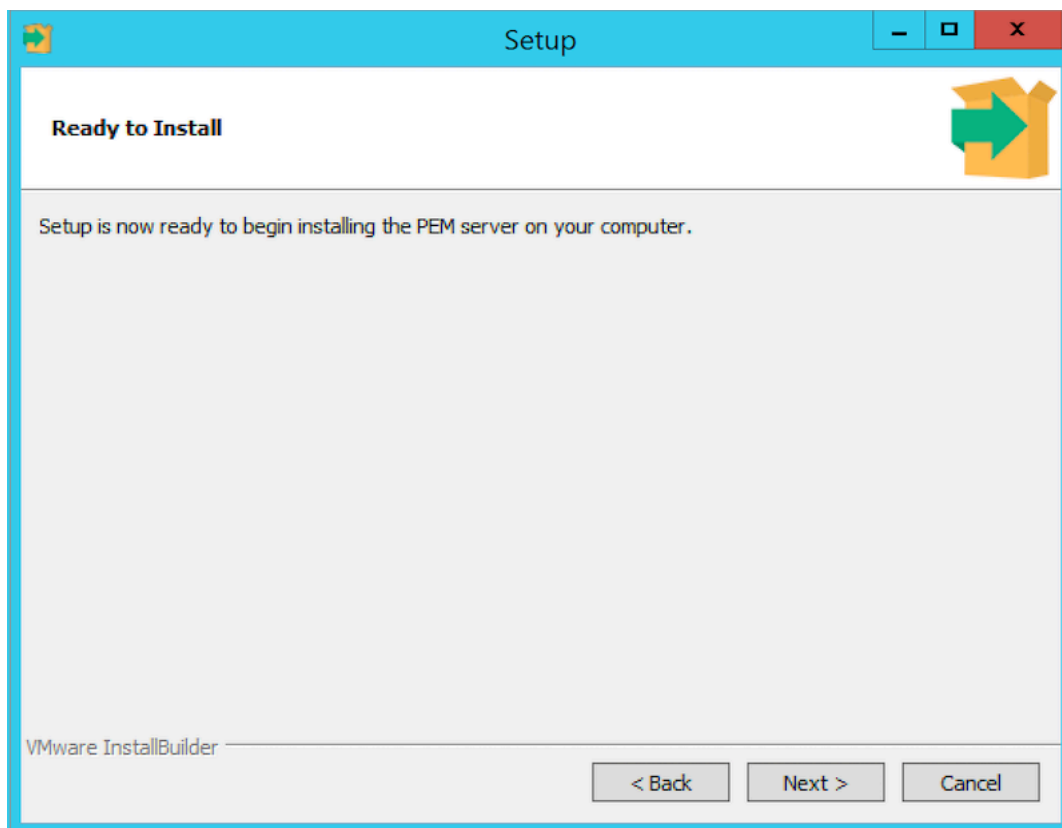


図 15: The PEM Server Installer - Ready to Install

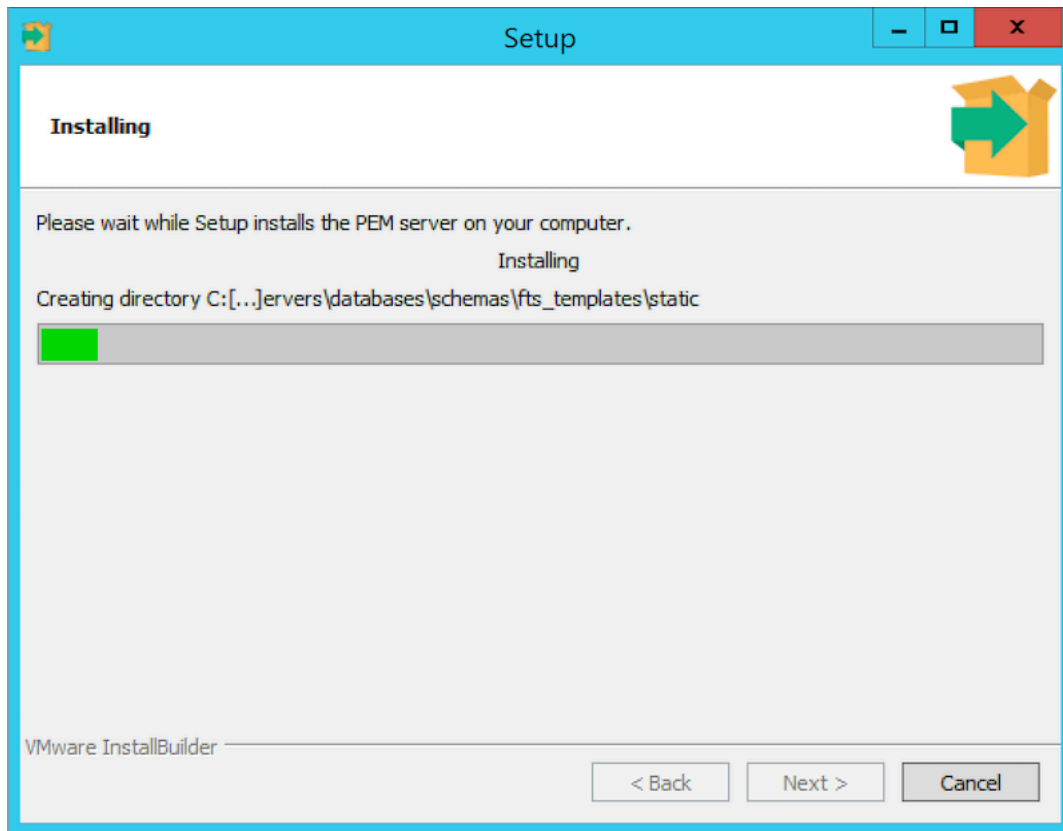


図 16: The PEM Server installation in progress

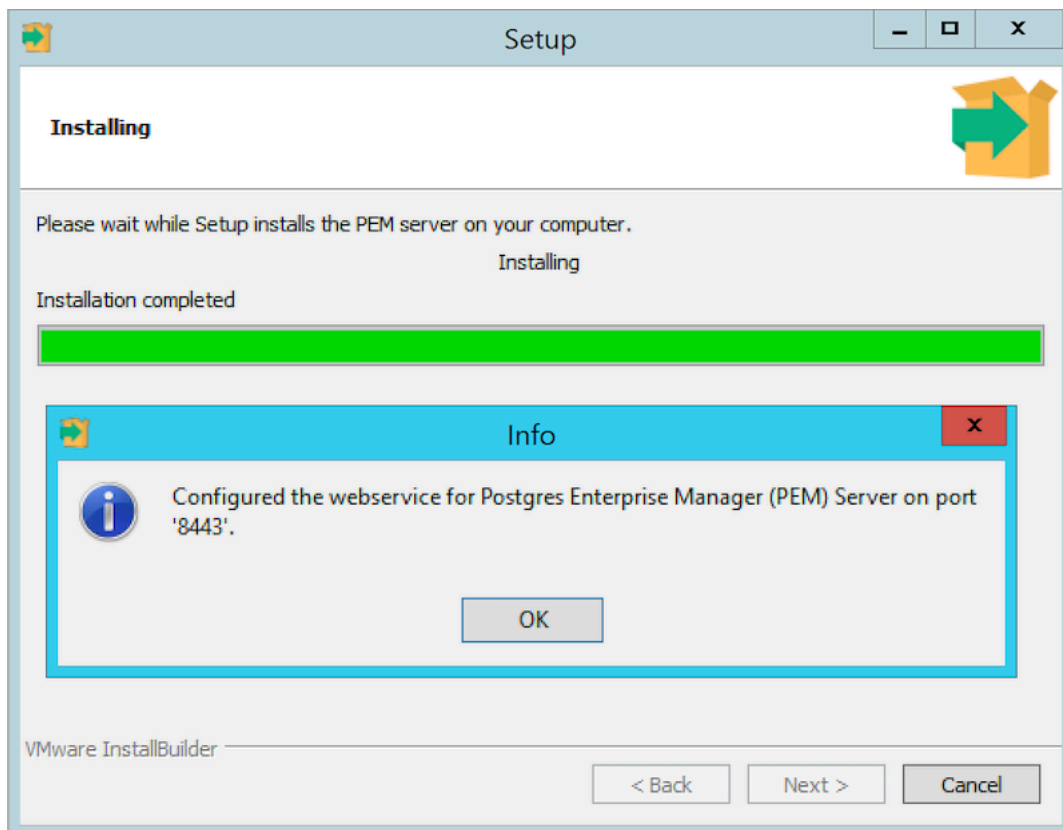


図 17: The PEM Server Installer - the PEM Web Service configured

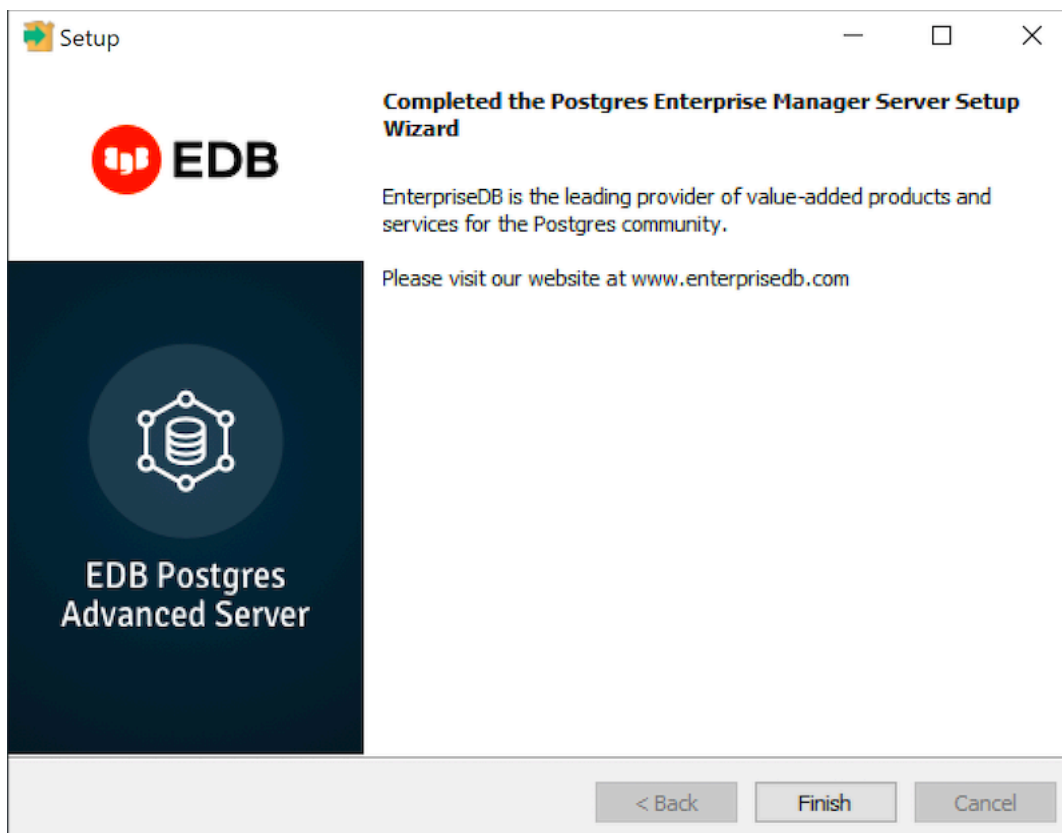


図 18: The PEM Server Installer - Upgrade Completion

ウィンドウがポップアップしてマシンのリスタートを求められたら、**Yes** をクリックしてマシンをリスタートします。したがって、**httpd** サービスが再起動します。

PEM バックエンドデータベースサーバと PEM-HTTPD を異なる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を 2 回（各ホストに 1 回）実行する必要があります。言語パックインストーラを抽出し、PEM インストーラを起動する前に PEM-HTTPD のホストにインストールします。インストーラを呼び出して言語パックを抽出する場合、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を抽出するための既存のパスを指定します。

!!! Note * デフォルトでは、EDB Language Pack は C:\edb\languagepack\v1 にインストールされます。

StackBuilder Plus を介して PEM サーバーをアップグレード処理する場合、以下に示すエラーが発生する可能性があります。エラーを表示した後、PEM はインストールが完了したことを通知します。インストールは完了していないため、ダウンロードした場所からインストーラファイルを読み出してインストールする必要があることにノートしてください。

PEM サーバーをアップグレード処理した後、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。バックエンドデータベースのアップグレード処理については、[Upgrading the Backend Postgres Database](#) を参照してください。

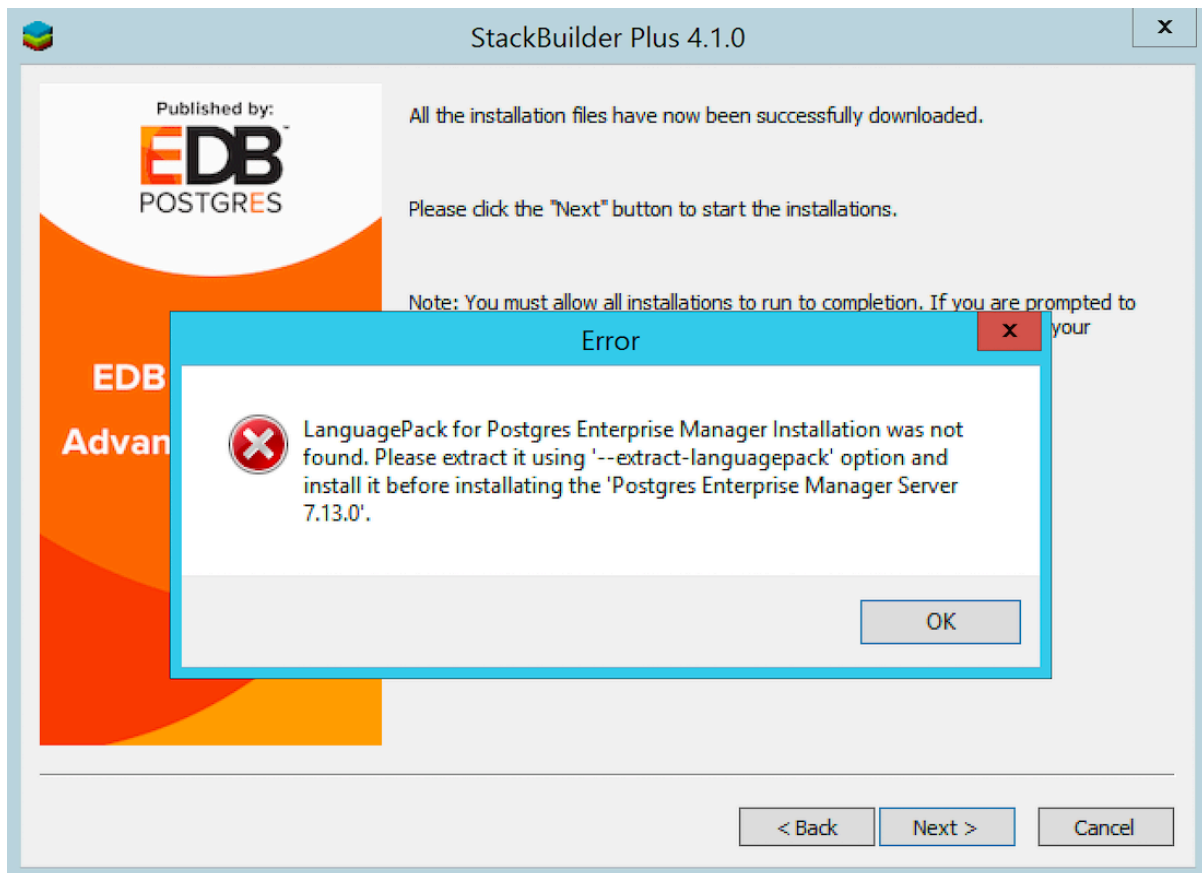


図 19: The PEM Server upgrade error

1.1.3 Windows ホストでの SQL プロファイラーのアップグレード

Windows ホストで SQL プロファイラを使用している場合、Windows は実行またはメモリにロードされたファイルをロックします。ロックされたファイルをリリースするには、アップグレードを実行する前に Postgres サーバーを停止する必要があります。

Windows、Services ダイアログを使用してサービスを制御できます。Services ダイアログを開くには、Control Panel から System and Security menu までナビゲートします。Administrative Tools を選択し、Services アイコンをダブルクリックします。Services ダイアログが開いたら、リストでサービス名前を強調表示し、ダイアログで提供されているオプションを使用して Stop サービスを実行します。

Postgres サーバーを停止した後：

1. `uninstall-sql-profiler.sql` スクリプトを呼び出して、各ノードの既存の SQL プロファイラークエリセットを削除します。デフォルトでは、Windows ホストでは、スクリプトは Advanced Server または PostgreSQL インストールの `share\contrib` ディレクトリにあります。
 - 次のサーバー固有のコマンドを使用できます。

PostgreSQL の場合：

```
* psql -f C:\Program Files\PostgreSQL\<x>\share\contrib\uninstall-sql-profiler.sql -
→d postgres -U postgres
```

- ここで、`x` は PostgreSQL のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

Advanced Server の場合：

```
* psql -f C:\Program Files\edb\as<x>\share\contrib\uninstall-sql-profiler.sql -d
→edb -U enterprisedb
```

- `x` は Advanced Server のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

2. プロファイリングする各ノードで新しい SQL Profiler インストーラを呼び出します。インストーラを Administrator として実行します。

PostgreSQL の場合：

```
* sqlprofiler-pg-<x>-<y>-windows-x64.exe
```

- ここで、`x` は PostgreSQL のバージョンであり、`y` は SQL プロファイラーのバージョンです。例：`sqlprofiler-pg-12-7.14.0-1-windows-x64.exe`。

Advanced Server の場合：

```
* sqlprofiler-edb-as<x>-<y>-windows-x64.exe
```

- `x` は Advanced Server のバージョン、`y` は SQL プロファイラーのバージョンです。例：`sqlprofiler-edb-as12-7.14.0-1-windows-x64.exe`。

SQL Profiler インストーラは、既存の SQL Profiler インストールを検出し、最新バージョンの SQL Profiler でアップグレードします。

3. メンテナンスデータベースで `sql-profiler.sql` スクリプトファイルを実行します。

PostgreSQL の場合：

```
* psql -f C:\Program Files\PostgreSQL\<x>\share\contrib\sql-profiler.sql -d
→postgres -U postgres
```

- ここで、`x` は PostgreSQL のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

Advanced Server の場合：

```
* psql -f C:\Program Files\edb\as<x>\share\contrib\sql-profiler.sql -d edb -U
→enterprisedb
```

- ここで、`x` は Advanced Server のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

4. 次に、Postgres サーバーをリスタートして、PEM クライアントからのノードのプロファイリングを再開します。

PEM コンポーネントを更新したら、バックエンドデータベースを更新する準備ができました。

1.2 Upgrading a PEM Native Package 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、PEM コンポーネントネイティブパッケージソフトの新しいバージョンを次のオーダーでインストールするだけです。

1. PEM サーバホストの各監視対象ノード `except` で PEM エージェントネイティブパッケージインストーラを起動します。PEM サーバーネイティブパッケージインストーラを呼び出します。`both` を PEM サーバーと PEM サーバホストにある PEM エージェントにアップグレードします。

インストール中、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処理した後、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。このステップはプラットフォーム固有です。

!!! Note * 以前のバージョンから 7.11 以降のバージョンにアップグレードされた Linux エージェントで実行されるシェル/バッチスクリプトを既に構成している場合、または構成する場合は、`agent.cfg` ファイルで `batch_script_user` パラメータのユーザを指定する必要があります。ルートユーザ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。ルートユーザを使用すると、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低下する可能性があります。ただし、ルートユーザを使用して `pemagent` をオリジナルの設定に復元してスクリプトを実行する場合は、`batch_script_user` パラメータ値を `root` に設定する必要があります。

以下のセクションでは、Linux ホストでのアップグレードプロセスについて順を追って説明します。

1.2.1 Linux ホストで PEM インストールをアップグレードするための前提条件

PEM は、python3、libboost、openssl、snmp ++、libcurl などを含むベンダ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が最新であることを保証には、次のプラットフォーム固有のコマンドを使用してオペレーティングシステムを更新する必要があります。openssl に必要な最小バージョンは 1.0.2k です。バージョン 10 より古いバージョンの PostgreSQL または Advanced Server を使用している場合、アップグレードの前に、PEM サーバーがインストールされているシステムにバージョン 10 以上の libs パッケージをインストールする必要があります。次のプラットフォーム固有のコマンドを使用して、ホストに libs バージョン 10 以降をインストールします。

CentOS または RHEL ホストで PEM インストールをアップグレードするための前提条件

CentOS または RHEL 7.x ホストでパッケージソフトをアップグレードするには

```
yum upgrade
```

CentOS または RHEL 8.x ホストでパッケージソフトをアップグレードするには

```
dnf upgrade
```

Advanced Server ライブラリをアップグレードするには：

```
yum install edb-as<X>-server-libs
```

PostgreSQL ライブラリをアップグレードするには：

```
yum install postgresql<X>-libs
```

<X>は、libs パッケージをインストールする PostgreSQL または Advanced Server バージョンです。

Debian または Ubuntu ホストで PEM インストールをアップグレードするための前提条件

Debian または Ubuntu ホスト上のパッケージソフトをアップグレードするには

```
apt-get update
```

Advanced Server ライブラリをアップグレードするには：

```
apt-get install edb-as<X>-server-libs
```

PostgreSQL ライブラリをアップグレードするには：

```
apt-get install postgresql<X>-libs
```

<X>は、libs パッケージをインストールする PostgreSQL または Advanced Server バージョンです。

SLES ホスト上の PEM インストールをアップグレードするための前提条件

SLES ホストでパッケージソフトをアップグレードするには

```
zypper update
```

Advanced Server ライブラリをアップグレードするには：

```
zypper install edb-as<x>-server-libs
```

PostgreSQL ライブラリをアップグレードするには：

```
zypper install postgresql<x>-libs
```

<X>は、libs パッケージをインストールする PostgreSQL または Advanced Server バージョンです。

1.2.2 PEM エージェントネイティブパッケージインストールのアップグレード

ネイティブパッケージソフトをパッケージソフトとして最初にインストールされた既存の PEM エージェントをアップグレードします。アップグレードプロセスでは、PEM エージェントの構成ファイルは更新されません。新しいエージェントをインストールした後、既存のエージェントの構成ファイルを新しいインストール場所に手動でコピーする必要があります。

CentOS または RHEL ホストでの PEM エージェントのアップグレード

CentOS または RHEL 7.x または 8.x の場合：

PEM エージェントをアップグレードするには、次のコマンドを使用します。

```
yum upgrade edb-pem-agent
```

CentOS または RHEL 8.x の場合、次のコマンドも使用できます。

```
dnf upgrade edb-pem-agent
```

Debian または Ubuntu ホストでの PEM エージェントのアップグレード

PEM エージェントをアップグレードするには、次のコマンドを使用します。

```
apt-get upgrade edb-pem-agent
```

SLES ホスト上の PEM エージェントのアップグレード

PEM エージェントをアップグレードするには、次のコマンドを使用します。

```
zypper update edb-pem-agent
```

1.2.3 PEM サーバーネイティブパッケージインストールのアップグレード

最初にネイティブパッケージソフトを使用して PEM サーバーをインストールした場合、ネイティブパッケージソフトを使用して PEM サーバーをアップグレードします。アップグレードするコマンドはプラットフォーム固有です。以下のプラットフォームのセクションを参照してください。

分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、PEM サーバーをアップグレードする前に、そのマシン上に PEM リポジトリを作成する必要があります。分離されたネットワークでの PEM リポジトリの作成の詳細については、[Creating a PEM Repository on an Isolated Network](#) を参照してください。

CentOS または RHEL ホスト上の PEM サーバーのアップグレード

RPM パッケージを使用して既存の RPM インストールをアップグレードするには、以下を行う必要があります。

yum パッケージマネージャを使用して、CentOS または RHEL 7.x または 8.x にインストールされている PEM サーバーのバージョンをアップグレード。

```
yum upgrade edb-pem
```

CentOS または RHEL 8.x で dnf コマンドを使用することもできます。

```
dnf upgrade edb-pem
```

!!! Note * CentOS または RHEL 7.x ホストに PEM サーバーを新規インストールする場合、パッケージはオペレーティングシステムの要件であるため、インストーラは edb-python3-mod_wsgi パッケージをインストールとともにインストールします。CentOS または RHEL 7.x ホスト上の PEM サーバーをアップグレード処理している場合、mod_wsgi パッケージは、オペレーティングシステムの要件を満たすために edb-python3-mod_wsgi パッケージに置き換えられます。

yum または dnf を使用して PEM サーバーをアップグレード処理した後、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 Platforms](#) を参照してください。

Debian または Ubuntu ホスト上の PEM サーバーのアップグレード

apt-get パッケージマネージャを使用して、サポートされている Debian または Ubuntu のバージョンで PEM サーバーのインストール済みバージョンをアップグレード。

```
apt-get upgrade edb-pem
```

apt-get を使用して PEM サーバーをアップグレード処理した後、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 Platforms](#) を参照してください。

SLES ホスト上の PEM サーバーのアップグレード

zypper パッケージマネージャを使用して、サポートされているバージョンの SLES ホストにインストールされている PEM サーバーのバージョンをアップグレード。

```
zypper update edb-pem
```

zypper を使用して PEM サーバーをアップグレード処理した後、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 platforms](#) を参照してください。

1.2.4 SQL プロファイラーネイティブパッケージインストールのアップグレード

Linux ホストにある SQL Profiler インストールをアップグレードするには：

1. `uninstall-sql-profiler.sql` スクリプトを呼び出して、各ノードの既存の SQL プロファイラークエリーセットを削除します。デフォルトでは、Linux ホストでは、スクリプトは Advanced Server または PostgreSQL インストールの `share/contrib` ディレクトリにあります。

- 次のサーバー固有のコマンドを使用できます。

PostgreSQL の場合：

```
* /usr/pgsql-<x>/bin/psql -f /usr/pgsql-<x>/share/contrib/uninstall-sql-profiler.  
→sql -d postgres -U postgres
```

- ここで、`x` は PostgreSQL のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

Advanced Server の場合：

```
* /usr/edb/as<x>/bin/psql -f /usr/edb/as<x>/share/contrib/uninstall-sql-profiler.  
→sql -d edb -U enterprisedb
```

- ここで、`x` は Advanced Server のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

2. プロファイリングする各ノードで新しい SQL Profiler インストーラを呼び出します。

PostgreSQL の場合：

```
* yum upgrade postgresql<x>-sqlprofiler
```

- ここで、**x** は PostgreSQL のバージョンです。

Advanced Server の場合：

```
* yum upgrade edb-as<x>-server-sqlprofiler
```

- x** は Advanced Server のバージョンです。

インストーラは既存の SQL Profiler インストールを検出し、SQL プロファイラーの最新バージョンでアップグレードします。

PostgreSQL 用の SQL Profiler のアップグレード処理の次の例を参照してください。

```
[root@localhost Downloads]# yum install postgresql12-sqlprofiler
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirrors.piconets.webwerks.in
* epel: hkg.mirror.rackspace.com
* extras: mirrors.piconets.webwerks.in
* updates: mirrors.piconets.webwerks.in
Resolving Dependencies
--> Running transaction check
--> Package postgresql12-sqlprofiler.x86_64 0:7.12.0-1.rhel7 will be updated
--> Package postgresql12-sqlprofiler.x86_64 0:7.14.0-1.rhel7 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package                               Arch              Version           Repository        Size
=====
Updating:
postgresql12-sqlprofiler              x86_64            7.14.0-1.rhel7    edb               88 k
=====
Transaction Summary
=====
Upgrade 1 Package

Total download size: 88 k
Is this ok [y/d/N]: y
Downloading packages:
No Presto metadata available for edb
postgresql12-sqlprofiler-7.14.0-1.rhel7.x86_64.rpm | 88 kB 00:00:04
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating : postgresql12-sqlprofiler-7.14.0-1.rhel7.x86_64 1/2
  Cleanup : postgresql12-sqlprofiler-7.12.0-1.rhel7.x86_64 2/2
  Verifying : postgresql12-sqlprofiler-7.14.0-1.rhel7.x86_64 1/2
  Verifying : postgresql12-sqlprofiler-7.12.0-1.rhel7.x86_64 2/2
Updated:
  postgresql12-sqlprofiler.x86_64 0:7.14.0-1.rhel7
Complete!
```

図 20: Upgrading SQL Profiler

3. 次に、メンテナンスデータベースで `sql-profiler.sql` スクリプトファイルを実行します。

PostgreSQL の場合：

```
* /usr/pgsql-<x>/bin/psql -f /usr/pgsql-<x>/share/contrib/sql-profiler.sql -d postgres -U postgres
```

- ここで、**x** は PostgreSQL のバージョンであり、**-d** はメンテナンスデータベースの名前を指定します。

Advanced Server の場合：

```
* /usr/edb/as<x>/bin/psql -f /usr/edb/as<x>/share/contrib/sql-profiler.sql -d edb -U enterprisedb
```

- ここで、`x` は Advanced Server のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。
4. PostgreSQL/ Advanced Server を再起動して、PEM クライアントからノードのプロファイリングを再開します。

PEM コンポーネントを更新したら、バックエンドデータベースを更新する準備ができました。

1.3 Creating a PEM 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 パッケージソフト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、必要な手順のノートです。個々のネットワークの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、以下を行う必要があります。

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 --downloadaddir=/<pem_dir>/ edb-pem
* mkdir /<epel_dir>
* yum install --downloadonly --downloadaddir=/<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>` を分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. 次を指定する接続情報を使用して、`/etc/yum.repos.d/pem.repo` というリポジトリ構成ファイルを作成します。

```
* [pemrepo]
* name=PEM Repository
* baseurl=file:///pem7/
* enabled=1
* gpgcheck=0
```

5. 次を指定する接続情報を使用して、`/etc/yum.repos.d/epel.repo` というリポジトリ構成ファイルを作成します。

```
* [epelrepo]
* name=epel Repository
```

(次のページに続く)

(前のページからの続き)

```
* 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ら、yum 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します。

- PEM サーバーをインストールするには：

```
* yum install edb-pem
```

- PEM サーバーをアップグレードするには：

```
* yum upgrade edb-pem
```

ローカル yum リポジトリの作成の詳細については、次の Web サイトをご覧ください：< <https://wiki.centos.org/HowTos/CreateLocalRepos> >

1.4 Upgrading a Graphical 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、PEM コンポーネントネイティブパッケージソフトの新しいバージョンを次のオーダーでインストールするだけです。

1. PEM サーバホスト **except** の各監視対象ノードで PEM エージェントネイティブパッケージのインストールを呼び出します。PEM サーバーネイティブパッケージインストールを呼び出します。**both** を PEM サーバーと PEM サーバホストにある PEM エージェントをアップグレードします。

インストール中、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処理した後、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。このステップはプラットフォーム固有です。

次のセクションでは、Linux ホストのグラフィカルインストーラを介して実行された PEM インストールのアップグレード処理プロセスについて説明します。

1.4.1 Linux ホスト上の PEM インストールをアップグレードするための前提条件

バージョン 10 より古いバージョンの Postgres または Advanced Server を使用している場合、アップグレードの前に、PEM サーバーがインストールされているシステムにバージョン 10 以上の libs パッケージをインストールする必要があります。

Advanced Server の場合：

```
yum install edb-as<X>-server-libs
```

PostgreSQL の場合：

```
yum install postgresql<X>-libs
```

<X>は、libs パッケージをインストールする Postgres または Advanced Server のバージョンです。

1.4.2 CentOS または RHEL ホストでの PEM エージェント（グラフィカルインストーラ）のアップグレード

グラフィカルインストーラによってインストールされたときの PEM エージェントのデフォルトのインストール場所は `/opt/edb/pem` です。次の例では、サーバーのインストール場所を `<PEM_installation_path>` に置き換えます。

1. バージョン固有のコマンドを使用して、`pemagent` サービスを停止します。

- CentOS または RHEL 7.x Or 8.x ホストの場合：

```
* systemctl stop pemagent
```

2. 次のいずれかのコマンドを実行して、サポートする `epel-release` パッケージをホストにインストールします。

```
* * yum install epel-release
```

```
* * yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-<X>.  
→noarch.rpm
```

- ここで、X は OS バージョンです。
- !!! Note
- `CentOS-Base.repo` ファイル (`/etc/yum.repos.d` にある) で `[extras]` リポジトリ定義を有効にする必要がある場合があります。

```
* If you are a Red Hat Network user,  
* - You must also enable the `rhel-<x>-server-optional-rpms` repository to use  
→EPEL packages, where *x* specifies the version of RHEL on the host. You can make  
→the repository accessible by enabling the `RHEL optional subchannel` for `RHN-  
→Classic`. If you have a certificate-based subscription, then you must also enable  
→`rhel-<x>-server-eus-optional-rpms` repository to use EPEL packages or please see  
→the `Red Hat Subscription Management Guide` for the required repository.  
* - You must also enable the `rhel-<x>-server-extras-rpms` repository, where `x`  
→specifies the version of the RHEL on the host.
```

3. `edb.repo` ファイルをインストールして構成します。

- a. EDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報を要求するには、次をご覧ください。
- [EDB Repository Access Steps](#)。

- b. リポジトリ構成ファイルを作成します。スーパーユーザ特権を想定し、次のコマンドを呼び出します。

```
* * yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.  
→noarch.rpm
```

リポジトリ構成ファイルの名前付けは `edb.repo` です。ファイルは `/etc/yum.repos.d` にあります。

- c. `edb.repo` ファイルを作成した後、有効なパラメータの値が保証になり、`baseurl` 仕様の `USERNAME` および `PASSWORD` プレースホルダーが登録済み EDB ユーザの名前とパスワードに置き換えられるようにするには、次のコマンドを実行します。

```
* * sed -i "s@<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

- 分離されたネットワークにあるマシンに PEM エージェントをインストールする場合、最初にそのマシンに PEM リポジトリを作成する必要があります。分離されたネットワークでの PEM リポジトリの作成の詳細については、[Creating a PEM repository in an Isolated Network](#) を参照してください。
4. `yum makecache` コマンドは、現在有効なリポジトリのメタデータをダウンロードします。コマンドが完了したら、利用可能なパッケージソフトをチェックして、リストに最新の PEM エージェントが含まれていることを確認します。

- CentOS または RHEL 7.x または 8.x の場合：

```
* yum makecache  
* yum list edb-pem-agent
```

- CentOS または RHEL 8.x の場合：

```
* dnf makecache  
* dnf list edb-pem-agent
```

5. PEM エージェント RPM をインストールします。インストールが完了したら、`yum info` コマンドを使用して、PEM エージェントのインストール情報を確認できます。

- CentOS または RHEL 7.x または 8.x の場合：

```
* yum install edb-pem-agent  
* yum info edb-pem-agent
```

- CentOS または RHEL 8.x の場合：

```
* dnf install edb-pem-agent
```

6. インストール後、PEM エージェント構成ファイル (`agent.cfg`) を以前の場所から RPM インストーラが必要とする場所にコピーします。

```
* cp /PEM_installation_path/agent/etc/agent.cfg /usr/edb/pem/agent/etc/agent.cfg
```

- !!! Note

- パッケージ管理機能と複製レプリケーション機能は、PEM 7.16 バージョンから非推奨になりました。PEM エージェントサービスのリスタート後、ワーカログファイルにワーニングメッセージが表示されないように、ファイルをコピーする際にこれらのパラメーターを agent.cfg ファイルから削除してください-allow_package_management および allow_streaming_replication。
- a. 次の場所にあるエージェント構成ファイルを開きます。

```
* * /usr/edb/pem/agent/etc/agent.cfg
```

- 次に、ca_file パラメータの値を設定します。

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

- b. サービスファイルとエージェント証明書のバックアップを作成します。
- CentOS または RHEL 7.x または 8.x では、次のコマンドを使用してサービスファイルをバックアップします。

```
* cp /usr/lib/systemd/system/pemagent.service /usr/lib/systemd/system/pemagent.  
→service_bkp
```

- 次に、エージェント証明書をコピーします。次のコマンドでは、<agent_id>はエージェント識別子（例または agent3 など）を指定する必要があります。

```
* mv /root/.pem/<agent_id>.key /root/.pem/<agent_id>.key.bkp  
* mv /root/.pem/<agent_id>.crt /root/.pem/<agent_id>.crt.bkp
```

- c. bitrock uninstaller を使用して PEM エージェントをアンインストールします。

```
* /PEM_installation_path/agent/uninstall-pemagent
```

- d. バージョン固有のコマンドを使用して、サービスファイルのバックアップとエージェント証明書をオリジナルの場所に復元します。例：
- RHEL または CentOS 7.x または 8.x ホストの場合：

```
* cp /usr/lib/systemd/system/pemagent.service_bkp /usr/lib/systemd/system/pemagent.  
→service
```

- 次に、エージェント証明書ファイルを移動します。次のコマンドでは、<agent_id>はエージェント識別子（例または agent3 など）を指定する必要があります。

```
* mv /root/.pem/<agent_id>.key.bkp /root/.pem/<agent_id>.key  
* mv /root/.pem/<agent_id>.crt.bkp /root/.pem/<agent_id>.crt
```

7. pemagent サービスを有効にし、pemagent と httpd をスタートします。

- RHEL または CentOS 7.x または 8.x ホストでは、次のコマンドを使用します。

```
* systemctl enable pemagent
* systemctl start pemagent
```

この時点で、PEM エージェントが稼働しているはずです。PEM Web インタフェースを使用して、エージェントのバージョンとステータスを確認できます。

1.4.3 グラフィカルインストーラーでインストールされた PEM サーバーのアップグレード

グラフィカルインストーラによってインストールされたときの PEM サーバーのデフォルトのインストール場所は `/opt/edb/pem` です。次の例では、サーバーのインストール場所を `<PEM_installation_path>` に置き換えます。

CentOS または RHEL ホストにグラフィカルインストーラーでインストールされた PEM サーバーのアップグレード

1. PEM からログアウトします。
2. PEM サーバーで PEMHTTPD サービスを停止します。PEM サーバーとウェブサーバが 2 つの異なるシステムにある場合、ウェブサーバでコマンドを実行します。
 - CentOS または RHEL 7.x Or 8.x ホストの場合：

```
* systemctl stop PEMHTTPD
* systemctl stop pemagent
```

3. 次のコマンドのいずれかを実行して、サポートする `epel-release` パッケージをホストにインストールします。

```
* * yum install epel-release
```

```
* * yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-<X>.  
→noarch.rpm
```

- ここで、`<X>` は OS バージョンです。
- !!! Note
- CentOS-Base.repo ファイル (`/etc/yum.repos.d` にある) で `[extras]` リポジトリ定義を有効にする必要がある場合があります。
- Red Hat Network ユーザの場合、
- `-rhel-<x>-server-optional-rpms` リポジトリが EPEL パッケージソフトを使用できるようにする必要があります。*x* はホスト上の RHEL のバージョンを指定します。RHN-Classic の RHEL optional subchannel を有効にすると、リポジトリにアクセスできる make になります。証明書ベースのサブスクリプションを持っている場合は、`rhel-<x>-server-eus-optional-rpms` リポジトリを有効に

して EPEL パッケージソフトを使用するか、必要なリポジトリについては Red Hat Subscription Management Guide を参照してください。-rhel-<x>-server-extras-rpms リポジトリも有効にする必要があります。ここで、x はホスト上の RHEL のバージョンを指定します。

4. edb.repo ファイルをインストールして構成します。

- a. EDB リポジトリへのアクセスを許可する資格情報も必要です。資格情報を要求するには、次をご覧ください。
- [EDB Repository Access Steps](#).
- b. リポジトリ構成ファイルを作成します。スーパーユーザ特権を想定し、次のコマンドを呼び出します。

```
* * yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.  
↪noarch.rpm
```

リポジトリ構成ファイルの名前付けは edb.repo です。ファイルは/etc/yum.repos.d にあります。

- c. edb.repo ファイルを作成した後、enabled パラメータの値が保証になり、baseurl 仕様の username および password プレースホルダーが登録済み EDB ユーザの名前とパスワードに置き換えられるようにするには、次のコマンドを実行します。

```
* * sed -i "s@<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

- 分離されたネットワークにあるマシンに PEM エージェントをインストールする場合、最初にそのマシンに PEM リポジトリを作成する必要があります。分離されたネットワークでの PEM リポジトリの作成の詳細については、[Creating a PEM repository in an Isolated Network](#) を参照してください。

5. yum makecache コマンドは、現在有効なリポジトリのメタデータをダウンロードします。コマンドが完了したら、利用可能なパッケージソフトをチェックして、リストに最新の PEM サーバーが含まれていることを確認します。

- CentOS または RHEL 7.x または 8.x の場合：

```
* yum makecache  
* yum list edb-pem
```

- CentOS または RHEL 8.x の場合：

```
* dnf makecache  
* dnf list edb-pem
```

6. PEM サーバー RPM をインストールします。インストールが完了したら、yum info コマンドを使用して PEM サーバーのインストール情報を確認できます。

- CentOS または RHEL 7.x または 8.x の場合：

```
* yum install edb-pem  
* yum info edb-pem
```

- CentOS または RHEL 8.x の場合：

```
* dnf install edb-pem
```

7. インストール後、agent.cfg ファイルを現在の場所（グラフィカルインストーラに必要な場所）から RPM パッケージに必要な場所にコピーします。

```
* cp /<PEM_installation_path>/agent/etc/agent.cfg /usr/edb/pem/agent/etc/agent.cfg
```

8. 次の場所にあるエージェント構成ファイルを開きます。

```
* * /usr/edb/pem/agent/etc/agent.cfg
```

- 次に、ca_file パラメータの値を設定します。

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

9. pem.db ファイル（およびその他の必要なファイル）を RPM インストール場所にコピーし、ファイルの所有権を変更します。PEM サーバーとウェブサーバが2つの異なるシステム上にある場合、ウェブサーバで以下のコマンドを実行します。

```
* cp -r /<PEM_installation_path>/server/share/pemhome/.pem/* /var/lib/pemhome/.pem/  
* chown -R pem:pem /var/lib/pemhome/.pem/
```

10. passwd ファイルのホームディレクトリを、グラフィカルインストーラによって識別された場所から RPM の場所に変更します。PEM サーバーとウェブサーバが2つの異なるシステム上にある場合、PEM サーバーと Web サーバーでコマンドを実行しウェブサーバ。

```
* usermod -m -d /var/lib/pemhome pem  
* cat /etc/passwd | grep pem
```

11. PEM サービスファイルとエージェント証明書のバックアップを作成します。

```
* cp /usr/lib/systemd/system/pemagent.service /usr/lib/systemd/system/pemagent.  
→service_bkp  
* mv /root/.pem/agent1.key /root/.pem/agent1.key.bkp  
* mv /root/.pem/agent1.crt /root/.pem/agent1.crt.bkp
```

12. PEM サーバーおよびウェブサーバマシンからグラフィカルアンインストーラーを使用して PEM サーバーをアンインストールします。

```
* /<PEM_installation_path> /server/uninstall-pemserver
```

13. サービスファイルのバックアップとエージェント証明書を PEM サーバーのオリジナルの場所に復元します。

```
* cp /usr/lib/systemd/system/pemagent.service_bkp /usr/lib/systemd/system/pemagent.  
→service  
* mv /root/.pem/agent1.crt.bkp /root/.pem/agent1.crt  
* mv /root/.pem/agent1.key.bkp /root/.pem/agent1.key
```

14. RPM またはインストーラーを使用して sslutils をインストールします（Windows の場合、インストーラーはファイルを上書きできないため、インストール前にデータベースサーバを停止する必要があります。そうしないと、システムのリスタートが必要になります）

データベースサーバを再起動します（拡張機能の共有オブジェクトがアップグレードされたため、データベースサーバを再起動する必要があります）

- セクション [Install sslutils](#) を参照してください
- 「ALTER EXTENSION」ステートメントを使用してバージョンをアップグレードします

```
* ALTER EXTENSION sslutils UPDATE [ TO new_version ]
```

15. PEM サーバーおよびウェブサーバで PEM RPM 構成スクリプトを実行します。プロンプトが表示されたら、バックエンドデータベースの詳細を指定します。エラーを生成せずにスクリプトを実行する必要があります。

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

16. PEM サーバーとウェブサーバで pemagent サービスを有効にします。ウェブサーバで pemagent および httpd サービスを開始します。

- CentOS または RHEL 7.x Or 8.x ホストの場合：

```
* systemctl enable pemagent
* systemctl start pemagent
* systemctl start httpd
```

17. PEM Web インタフェースを起動します。サーバーとエージェントをチェックして、PEM バージョン、サーバーステータス、およびスキーマバージョンを確認します。この時点で、すべてが稼働しているはずです。

18. 使用されなくなった PEMHTTD サービスをウェブサーバからアンインストールできるようになりました。

```
* /opt/edb/pem/httpd/uninstall-pemhttpd
```

1.4.4 グラフィカルインストーラーを使用した SQL プロファイラーインストールのアップグレード

SQL プロファイラーのグラフィカルインストーラーは、11.x より前の PostgreSQL および Advanced Server バージョン（10.x、9.6）で使用できます。

Linux ホスト上の SQL Profiler インストールをアップグレードするには：

1. `uninstall-sql-profiler.sql` スクリプトを呼び出して、各ノードの既存の SQL プロファイラーのエリセットを削除します。デフォルトでは、グラフィカルインストーラーでインストールされた Linux ホストで PostgreSQL を使用している場合、スクリプトは PostgreSQL インストールの下での `share/postgresql/contrib` ディレクトリにあります。Advanced Server の場合、スクリプトは Advanced

Server インストールの `share/contrib` ディレクトリにあります。次のサーバー固有のコマンドを使用できます。

PostgreSQL の場合：

```
* /opt/PostgreSQL/<x>/bin/psql -f /opt/PostgreSQL/<x>/share/postgresql/contrib/  
→uninstall-sql-profiler.sql -d postgres -U postgres
```

- ここで、`x` は PostgreSQL のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

Advanced Server の場合：

```
* /opt/edb/as<x>/bin/psql -f /opt/edb/as<x>/share/contrib/uninstall-sql-profiler.  
→sql -d edb -U enterprisedb
```

- ここで、`x` は Advanced Server のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

2. 次に、プロファイリングする各ノードで新しい SQL Profiler インストーラを呼び出します。管理者としてインストーラを実行します。

PostgreSQL ホストの場合：

```
* sqlprofiler-pg-<x>-<y>-linux.exe
```

- `x` は PostgreSQL のバージョン、`y` は SQL プロファイラーのバージョンです。例：
`sqlprofiler-pg-12-7.14.0-1-linux.exe`。

Advanced Server の場合：

```
* sqlprofiler-edb-as<x>-<y>-linux.exe
```

- `x` は Advanced Server のバージョン、`y` は SQL プロファイラーのバージョンです。例：
`sqlprofiler-edb-as12-7.14.0-1-linux.exe`。

SQL Profiler インストーラは、既存の SQL Profiler インストールを検出し、最新バージョンの SQL Profiler でアップグレードします。

3. メンテナンスデータベースで `sql-profiler.sql` スクリプトファイルを実行します。

PostgreSQL の場合：

```
* /opt/PostgreSQL/<x>/bin/psql -f /opt/PostgreSQL/<x>/share/postgresql/contrib/sql-  
→profiler.sql -d postgres -U postgres
```

- ここで、`x` は PostgreSQL のバージョンであり、`-d` はメンテナンスデータベースの名前を指定します。

Advanced Server の場合：

```
* /opt/edb/as<x>/bin/psql -f /opt/edb/as<x>/share/contrib/sql-profiler.sql -d edb -  
→U enterprisedb
```

- ここで、**x** は Advanced Server のバージョンであり、**-d** はメンテナンスデータベースの名前を指定します。
4. PostgreSQL/ Advanced Server を再起動して、PEM クライアントからノードのプロファイリングを再開します。

PEM コンポーネントを更新したら、バックエンドデータベースを更新する準備ができました。

1.5 Configuring a PEM Server on a Linux Host

PEM サーバーをアップグレード処理した後、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
```

構成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プロンプトを求めます。受け入れられるオプションは次のとおりです。

Option	Description
-acp	PEM エージェント証明書パスを定義します。デフォルトはルートです。
-ci	エージェントがサーバーに接続し、サーバーの <code>pg_hba.conf</code> ファイルに追加される CIDR 形式のネットワークアドレスレンジ。例：192.168.1.0/24。デフォルトは 0.0.0.0/0 です。
-dbi	データベースサーバをインストールするためのディレクトリ。例、Advanced Server の場合は <code>/usr/edb/as11</code> 、PostgreSQL の場合は <code>/usr/pgsql-11</code> です。
-ds	PEM データベースサーバのユニットファイル名前。Advanced Server の場合、デフォルトのファイル名前は <code>edb-as-11</code> です。PostgreSQL の場合、 <code>postgresql-11</code> です。
-ho	PEM データベースサーバのホストアドレス。
-p	PEM データベースサーバのポート番号。
-ps	pemagent のサービス名前。デフォルト値は <code>pemagent</code> です。
-sp	PEM データベースサーバのスーパーユーザパスワード。この値は必須です。
-su	PEM データベースサーバのスーパーユーザ名前。
-t	インストールの種類：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、バック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ることにノートしてください。
-un	PEM サーバーの登録を解除します。
-h	ヘルプを表示します。
	NA

コマンドラインで構成プロパティを指定しない場合、スクリプトによって値の入力が求められます。スクリプト関連のヘルプをビューするには、次のコマンドを使用します。

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

PEM サーバー構成ファイルを実行した後、バージョン固有のサービス制御コマンドを使用して `httpd` サービスをリスタートします。

RPM パッケージを使用して PEM サーバーまたは PEM エージェントをインストールまたは構成する方法の詳細については、[PEM Linux Installation Guide](#) を参照してください。

第2章 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の両方を更新場合は、バックエンドデータベースを更新する前に PEM コンポーネントの更新（サーバーとエージェント）を実行する必要があります。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](#) を参照してください。

!!! Note * PEM 8.0 以降では、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としてのみサポートされています。したがって、バックエンドデータベースサーバがバージョン 11 よりも低い場合、まずバックエンドデータベースサーバをアップグレードしてから、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。

このセクションで説明する更新プロセスでは、`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容易にします。

!!! Note * ソース PEM サーバーが 7.16 バージョンよりも低い場合、`pg_upgrade` を実行する前に次の関数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 以降では非推奨です。したがって、これらのデータ型を `timestampz` データタイプに置き換えるには、以下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 timestampz;
*     ALTER TABLE pemhistory.user_info
*         ALTER COLUMN valuntil SET DATA TYPE timestampz;
```

(次のページに続く)

```
*      -- Now update the pem.probe_column itself
*      UPDATE pem.probe_column
*      SET sql_data_type = 'timestampz'
*      WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil' ;
*  END;
*  $$ LANGUAGE 'plpgsql' ;
```

- 以下のファンクションを置き換えて、アラートエラーを回避します。

```
*  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 there's 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
```

(前のページからの続き)

```
*      $SQL$ 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転送をサポートします。例、pg_upgrade を使用して、PostgreSQL 9.6 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、Advanced Server 11 バックエンドデータベースには移動できません。異なるタイプのバックエンドデータベース（つまり、PostgreSQL サーバーから Advanced Server へ）に移行する場合は、[Moving the Postgres Enterprise Manager Server](#) を参照してください。

pg_upgrade の使用に関する詳細については、次を参照してください。

< <http://www.postgresql.org/docs/current/pgupgrade.html> >

1. 更新されたインストーラをダウンロードして起動します。PostgreSQL および Advanced Server のインストーラーは、EDB ウェブサイトから入手できます。

- <https://www.enterprisedb.com/software-downloads-postgres>
- アップグレード処理するサーババージョンのインストーラをダウンロードしたら、PEM サーバーのホストでインストーラを起動します。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処理場合、新しいサーバーは別のポートで監視するように構成する必要があることに注意してください。

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されているのと同じバージョンの sslutils を実行する必要があります。EDB インストーラーで使用される SSL Utils パッケージは、次の場所からダウンロードできます。

- <https://www.enterprisedb.com/downloads/modified-gpl-source-code>

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 の構成プロセスはプラットフォーム固有です。

- **** Linux で****
- -Advanced 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。
- -PEM バックエンドデータベースとして PostgreSQL を使用している場合、PostgreSQL コミュニティリポジトリにアクセスできることを保証し、次のコマンドを使用します。

```
*      yum install sslutils_<X>
```

- `<X>` はサーババージョンです。

- -PEM バックエンドデータベースとして PostgreSQL の EDB ワンクリックインストーラを使用している場合

```
*      yum install gcc openssl-devel
```

- PATH の値を設定して、pg_config プログラムを見つけられるようにします

```
*      export PATH=$PATH:/opt/postgres_inst_dir/<X>/bin/
```

- sslutils フォルダーに移動して、入力します

```
*      make USE_PGXS=1
*      make USE_PGXS=1 install
```

psql を使用して sslutils 拡張機能を作成します

```
*      CREATE EXTENSION sslutils
```

- Debian 10 および Ubuntu 20 では、セキュリティの理由により、証明書を受け入れるための要件が増加していることにノートしてください。ユーザがマシンに PEM エージェントをインストールする場合、4096 ビット RSA キーと sha256 シグネチャアルゴリズムのサポートが追加されている `sslutils` を 1.3 にアップグレードする必要があります。ユーザが `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に接続すると、エラー `ca md too weak` がログに記録される場合があります。

- **** Windows ****

- -sslutils は、オリジナルのバックエンドデータベースで sslutils をコンパイルに使用したのと同じコンパイラを使用して、新しいバックエンドデータベースで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（EDB から）または Advanced Server インストーラ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、Visual Studio を使用して sslutils をビルドします。PostgreSQL 9.6 以降にアップグレード処理する場合は、Visual Studio 2010 を使用してください。

- Windows 上で特定のバージョンの Postgres 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文書を参照してください。コア文書は、PostgreSQL プロジェクトのウェブサイトで入手できます。

- <<https://www.postgresql.org/docs/current/install-windows.html>>

- -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、ご使用のバージョンの Windows 用の OpenSSL を見つけてインストールする必要があります。OpenSSL インストーラを呼び出す前に、前提条件の再配布可能ファイル（`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。
(次のページに続く)

(前のページからの続き)

- OpenSSL をインストールした後、次から入手可能な `sslutils` ユーティリティパッケージをダウンロードして解凍します。

• `<https://www.enterprisedb.com/downloads/modified-gpl-source-code>`

- -解凍した sslutils フォルダを Postgres インストールディレクトリ（つまり 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 GETTEXT_PATH=\ <path_to_gettext>
*      SET OPENSSL_PATH=\ <path_to_openssl>
*      SET PG_PATH=\ <path_to_pg_installation_dir>
*      SET ARCH=x86
*      msbuild sslutils.proj /p:Configuration=Release
```

- どこで：
- -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。
- -例、次のコマンドセットは、PostgreSQL 11 サーバーに OpenSSL サポートを組み込みます。

```
*      SET USE_PGXS=1
*      SET OPENSSL_PATH=C:\OpenSSL-Win32
*      SET GETTEXT_PATH="C:\Program Files\PostgreSQL\11"
*      SET PG_PATH="C:\Program Files\PostgreSQL\11"
*      SET ARCH=x86
*      msbuild sslutils.proj /p:Configuration=Release
```

- -ビルドが完了すると、sslutils ディレクトリに次のファイルが含まれます。

• `sslutils--1.3.sql`

• `sslutils--unpacked--1.3.sql`

• `sslutils--pemagent.sql.in`

• `sslutils.dll`

- -コンパイルされた sslutils ファイルをインストールに適した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 または CentOS 7.x または 8.x で、コマンドラインを開き、スーパーユーザの 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 ダイアログを使用してサービスを制御できます。Services ダイアログを開くには、Control Panel から System and Security メニューに移動します。Administrative Tools を選択し、Services アイコンをダブルクリックします。Services ダイアログが開いたら、リストでサービス名前を強調表示し、ダイアログで提供されているオプションを使用してサービスを停止します。

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 認証を実施するように構成されている場合、データベーススーパーユーザの接続プロパティ（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か、`pg_upgrade` を呼び出す前に信頼接続を許可するように `pg_hba.conf` ファイルを変更する必要があります。`.pgpass` ファイルにエントリーを作成する方法の詳細については、次の URL にある PostgreSQL コア文書を参照してください。

- < <http://www.postgresql.org/docs/current/libpq-pgpass.html> >

アップグレードプロセス中に、`pg_upgrade` はシリーズのログファイルを書き込みます。クラスターの所有者は、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。アップグレードログファイルを削除しないように `pg_upgrade` に指示するには、`pg_upgrade` を呼び出すときに `--retain` キーワードを含めます。

- `pg_upgrade` を呼び出すには、クラスター所有者の ID を想定し、クラスター所有者が書き込み権限を持つディレクトリに移動して、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>
```

- どこで：
- `-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 ディレクトリの下に `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してい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。

例、次のコマンド：

```
* C:\>"C:\Program Files\PostgreSQL\11\bin\pg_upgrade.exe"
* -d "C:\Program Files\PostgreSQL\9.6\data"
* -D "C:\Program Files\PostgreSQL\11\data"
* -b "C:\Program Files\PostgreSQL\9.6\bin"
* -B "C:\Program Files\PostgreSQL\11\bin"
* -p 5432 -P 5433
* -U postgres
```

- Windows システムで PEM データベースを PostgreSQL 9.6 から PostgreSQL 11 に移行するように 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）。
- 呼び出されると、pg_upgrade は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、以下を参照してください。
- < <http://www.postgresql.org/docs/current/ 静的/pgupgrade.html> >

5. 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 ディレクトリに次の証明書ファイルをコピーします。

- 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したら、ファイルには以下で説明する（プラットフォーム固有の）権限が必要です。
- ** Linux での許可と所有権**
 - ファイル名 | 所有者 | 許可 || ----- | ---- | ----- || ca_certificate.crt | ポストGRES | -rw --- - || ca_key. キー | ポストGRES | -rw --- - || ルート.crt | ポストGRES | -rw --- - || ルート.crl | ポストGRES | -rw --- - || サーバー. キー | ポストGRES | -rw --- - || server.crt | ポストGRES | -rw-r-r- |
- Linux では、証明書ファイルは postgres が所有している必要があります。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>
```

- ここで、`file_name` は証明書ファイルの名前を指定します。
- `server.crt` ファイルは、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は、ファイルの所有者のみが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>
```

- ここで、`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、ソースホストから移動した証明書ファイルは、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インストールを実行したサービスアカウントが所有する必要があります。Run as Administrator オプション（インストーラのコンテキストメニューから選択）を使用して PEM サーバーと Postgres インストーラを呼び出した場合、証明書ファイルの所有者は Administrators になります。

- 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、ファイル名前を右クリックして、Properties を選択します。

```
![The Security tab](../images/upgrade_backendDB_permission_Windows_security.png)
```

```
** The Security tab **
```

- Security タブに移動し、Group or user name を強調ビューして、割り当てられた権限を表示します。Edit または Advanced を選択して、選択したユーザに関連付けられた権限を変更できるダイアログにアクセスします。

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マッチするように、新しいサーバーの `postgresql.conf` ファイルを変更する必要があります。

デフォルトでは、`postgresql.conf` ファイルは次の場所にあります。

- -Linux の Postgres バージョン 10 未満、`/opt/PostgreSQL/<X>/data-Linux` の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 以上、`/opt/PostgreSQL/<X>/data-Linux` の RPM でインストールされた Postgres バージョン 10 以上、`/usr/pgsql/<X>/data` 任意の Postgres Windows のバージョン、`C:\Program Files\PostgreSQL\<X>\data`
- ここで、`<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。以下のパラメーターを変更します。
- -オリジナルのバックエンドデータベース（通常は 5432）によって監視されているポートで監視する `port` パラメータ。-`ssl` パラメータを `on` に設定する必要があります。

- また、次のパラメーターが有効になっていることを保証する必要があります。パラメーターがコメント化されている場合は、各 `postgresql.conf` ファイルエントリーの前から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と同等の方法で動作することを保証に変更が必要な他のパラメータ設定がある場合があります。`postgresql.conf` ファイルを注意深く保証して、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認証を実施する方法を指定するパラメータ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。

- `# 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`
- `# 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 の Postgres バージョン 10 未満、`/opt/PostgreSQL/<X>/data`-Linux の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 以上、`/Opt/PostgreSQL/<X>/data`-Linux の RPM でインストールされた Postgres バージョン 10 以上、`/var/lib/pgsql/<X>/data`-Advanced Server バージョン Linux の RPM とともに `/var/lib/edb/as<X>/data` にインストールする場合は 10 以上 Windows の Postgres バージョンの場合、`C:\Program Files\PostgreSQL\<X>\data`
- ここで、`<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 または CentOS 7.x または 8.x で、コマンドラインを開き、スーパーユーザの ID を引き継ぎます。次のコマンドを入力します。

```
* systemctl stop <service_name>
```

- ここで、`service_name` はバックエンドデータベースサーバの名前です。

Windows を使用している場合は、Services ダイアログを使用してサービスを制御できます。Services ダイアログを開くには、Control Panel から System and Security メニューに移動します。Administrative

Tools を選択し、Services アイコンをダブルクリックします。Services ダイアログが開いたら、リストでサービス名前を強調表示し、ダイアログで提供されているオプションを使用してサービスをスタートします。

第3章 Moving the Postgres Enterprise Manager Server

このセクションの手順では、あるホストマシンから新しいホストマシンに PEM サーバーを移動する方法について説明します。新しいホスト（ターゲット）上の PEM サーバーは、オリジナルのホスト（ソース）と同じバージョンの PEM サーバーインストーラでインストールする必要があります。同じインストーラ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可能性があることにノートしてください。

ターゲットサーバー（PostgreSQL または Advanced Server）のバックアップエンドデータベースは、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である場合があります。PostgreSQL ホストにある PEM サーバーは、Advanced Server ホストに移行できます。

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することを保証必要があります。

1. ターゲットホストの準備

- ターゲットホストで PEM サーバーのインストーラを起動します。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一バージョンの PEM サーバーインストーラを使用する必要があります。ことにノートしてください。
- ターゲットサーバーのバックアップ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である場合があります。新しい PEM サーバーがオリジナル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存在しない場合、`sslutils` 拡張機能の同じバージョンが新しいサーバホストにインストールされていることを保証必要があります。PEM インストーラとともに配布される `sslutils` のバージョンは、EDB ウェブサイトから無料でダウンロード。
- <https://www.enterprisedb.com/downloads/modified-gpl-source-code>
- 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、次の場所にある PEM Installation Guides を参照してください。
- <https://www.enterprisedb.com/docs/p/edb-postgres-enterprise-manager>

2. 新しい PEM サーバーから既存のスキーマを削除する

移行プロセスにより、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` スキーマが再作成されます。移動の準備として、`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 `psql` クライアントを開くか、Postgres Enterprise Manager メニューから SQL Shell (`psql`) を選択します。

- `psql` クライアントが開いたら、`pem` バックエンドデータベースにデータベーススーパーユーザとして接続します。ターゲットホストの `pem` データベースに接続した後、次のコマンドを使用して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** キーワードを含めて、すべての依存オブジェクトを削除するよう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** を表示して、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する

- 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ている PEM エージェントの総数を検出するには、**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** テーブルをクエリーします。

```
* SELECT id FROM pem.agent WHERE active = true;
```

オリジナルの PEM サーバーに存在するエージェントの数を 1 つ少なくして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は、ターゲットホスト上に 1 つのエージェントをすでに作成しています。例、ソースサーバーに 3 つのエージェントが含まれる場合、2 つの追加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** データベースで **psql** セッションを開き、必要なエージェントを作成します。次のコマンドを使用します。

```
* CREATE USER agent<X>;
```

- ここで、<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によってターゲットホスト上に作成されます。
- 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>;
```

- ここで、<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する

- 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 **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、コマンドラインを開き、**bin** ディレクトリに移動して、次のように入力します。

```
* pg_dump -U <user_name> <db_name> > <file_name>
```

- どこで：
- <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、指定したユーザに関連付けられたパスワードを入力します。
- 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。スクリプトの名前付けは`backup.sql`で、`tmp`ディレクトリに作成されます。`pg\_dump`は、ユーザ`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ないことに注意してください。
- !!! Note
- ソース PEM サーバーが 7.16 バージョンよりも低い場合、pg_dump を実行してバックアップを取る前に、次の関数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 以降では非推奨です。したがって、これらのデータ型を`timestampz`データタイプに置き換えるには、以下のコマンドを使用します。

```
* DO
* $$
* DECLARE
*     rec record;
*     cnt integer;
* BEGIN
*     -- Check for the deprecated type in our user info probe
*     SELECT count(*) INTO cnt
*     FROM pem.probe_column
*     WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil' ;
*     IF cnt = 0 THEN
*         RETURN;
*     END IF;
*     ALTER TABLE pemdata.user_info
*         ALTER COLUMN valuntil SET DATA TYPE timestampz;
*     ALTER TABLE pemhistory.user_info
*         ALTER COLUMN valuntil SET DATA TYPE timestampz;
*     -- Now update the pem.probe_column itself
*     UPDATE pem.probe_column
*     SET sql_data_type = 'timestampz'
*     WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil' ;
* END;
* $$ LANGUAGE 'plpgsql' ;
```

- -以下のファンクションを置き換えて、アラートエラーを回避します。

```

*      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 there's 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する

- `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する

- ターゲットホストでコマンドラインを開き、bin ディレクトリ（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）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実行して、psql を開始します。

```
* psql -U <user_name> -d pem -f <file_name>
```

- どこで：
- <user_name>は、データベーススーパーユーザの名前を指定します。指定されたユーザは、バックエンドデータベースの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、データベーススーパーユーザ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 名前付け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 postgres に関連付けられた特権を使用して呼び出されます。

7. ターゲットホスト上のデータベースサーバーを停止する

- CentOS または RHEL 7.x または 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>
```

- ここで、<service_name>はバックエンドデータベースサーバ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、サービス名前は postgresql-<x>であり、Advanced Server バックエンドデータベースの場合、サービス名前は edb-as-<X>です。ここで、<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 ダイアログを使用してサービスを制御できます。Services ダイアログを開くには、Control Panel から System and Security メニューに移動します。Administrative Tools を選択し、Services アイコンをダブルクリックします。Services ダイアログが開いたら、リスト内のサービス名前を強調表示し、ダイアログに表示されるオプションを使用してサービスを停止します。

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
```

- どこで：
- <X>は、システム上の PostgreSQL のバージョンを指定します。
- ファイルはターゲットクラスタに既に存在します。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したら、ファイルには、以降のセクションで説明する（プラットフォーム固有の）権限が必要です。
- ** Linux での許可と所有権**
- **ファイル名 | 所有者 | 許可** | ———— | ———— | ———— || ca_certificate.crt | ポストGRES | -rw ———— || ca_key. キー | ポストGRES | -rw ———— || ルート.crt | ポストGRES | -rw ———— || ルート.crl | ポストGRES | -rw ———— || サーバー. キー | ポストGRES | -rw ———— || server.crt | ポストGRES | -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>
```

- ここで、file_name は証明書ファイルの名前を指定します。
- server.crt ファイルは、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は、ファイルの所有者のみが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>
```

- ここで、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、ソースホストから移動した証明書ファイルは、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インストールを実行したサービスアカウントが所有している必要があります。Run as Administrator オプション（インストーラのコンテキストメニューから選択）を使用して PEM サーバーと Postgres インストーラを呼び出した場合、証明書ファイルの所有者は Administrators になります。

- 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、ファイル名前を右クリックして、Properties を選択します。

```
![The Permissions tab](../images/move_server_windows_permission.png)
```

```
** The Permissions tab **
```

- Security タブに移動し、Group or user name を強調ビューして、割り当てられた権限を表示します。Edit または Advanced を選択して、選択したユーザに関連付けられた権限を変更できるダイアログにアクセスします。

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する

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。このステップはプラットフォーム固有です。

- **** Linux で****
- **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。
- ファイルのファイル所有権とアクセス許可は、次のように設定する必要があります。
- **ファイル名 | 所有者 | 許可** || ——— -| ——— | ——— || エージェント 1。キー | ルート | -rw — - || agent1.crt | ルート | -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****
- **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 インストーラを呼び出したユーザの名前です。
- 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許可は、ソースマシン上の証明書ファイルの所有権と許可とマッチする必要があります。Run as Administrator オプション（インストーラのコンテキストメニューから選択）を使用して PEM サーバーと Postgres インストーラを呼び出した場合、エージェント証明書ファイルの所有者は Administrators になります。
- 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、ファイル名前を右クリックして、Properties を選択します。Security タブに移動し、Group or user name を強調ビューして、割り当てられた権限を表示します。Edit または Advanced を選択して、選択したユーザに関連付けられた権限を変更できるダイアログにアクセスします。

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、サーバーをリスタート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をリスタートするには、次のコマンドを使用します。

```
* /etc/init.d/<service_name> start
```

- ここで、service_name はバックエンドデータベースサーバの名前です。

Windows を使用している場合は、Services ダイアログを使用してサービスを制御できます。Services ダイアログを開くには、Control Panel から System and Security メニューに移動します。Administrative Tools を選択し、Services アイコンをダブルクリックします。Services ダイアログが開いたら、リストでサービス名前を強調表示し、ダイアログに表示されるオプションを使用して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 サーバホストに接続する

- 既存の PEM エージェントに新しい PEM サーバホストに接続するよう指示するには、次の手順を実行する必要があります。
- -PEM エージェントホストが新しい PEM サーバホストに接続できることを確認します。-新しい PEM サーバの IP アドレスとポートを指定して、レジストリ（PEM エージェントのある各 Windows ホスト）またはエージェント構成ファイル（PEM エージェントのある各 Linux ホスト）を変更します。-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 上にある場合**

エディタの選択を使用して agent.cfg ファイルを変更し、pem_host パラメーターと pem_port パラメーターで PEM サーバの新しい 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

```
![The agent.cfg file](../images/move_pem_server_agent_cfg.png)
```

- agent.cfg ファイルを変更した後、PEM エージェントサービスをリスタート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。

```
* /etc/init.d/pemagent restart
```

- ** PEM エージェントが Windows に存在する場合**
- 監視対象ノードの Windows レジストリを変更前に、PEM エージェントのホスト上のファイアウォールが PEM サーバへの接続を許可することを確認してください。PEM エージェントホストが PEM サーバホストに接続できることを確認したら、Windows Registry Editor を使用して PEM_HOST および PEM_PORT エントリを保証および編集し、PEM サーバが使用するホストとポートを正しく識別できるようにします。Registry Editor を開くには、Windows の Run ダイアログまたは 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ビューまたは変更します。
- 64 ビット Windows では、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 wow6432Mode EnterpriseDB PEM Agent
- 32 ビット Windows では、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

![The Windows Registry Editor](../images/move_pem_server_windows_registry.png)

- 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 **Name** を右クリックし、コンテキストメニューから **Modify** を選択して **Edit String** ダイアログを開きます。

![The Windows Registry Editor](../images/move_pem_server_windows_registry_edit.png)

- **Edit String** ダイアログを使用して、エントリの値を変更 make ます。完了したら、OK をクリックして変更を保存するか、Cancel をクリックして保存せずに終了します。
- レジストリを変更した後、PEM エージェントのサービスをリスタートする必要があります。Services ダイアログ (Windows Control Panel からアクセス) を使用して、Postgres Enterprise Manager - pemagent サービスをリスタートできます。

![Restarting the PEM Agent service](../images/move_pem_server_restart_agent.png)

** Restarting the PEM Agent service **

- サーバーを移動した後、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、および監視対象サーバーの新しいホストに接続します。

第4章 Troubleshooting

</ div>

4.1 PEM サーバーの再構成

状況によっては、PEM サーバーをアンインストールし、再度インストールしてから、サーバーを再構成する必要があります。次のコマンドを指定されたシーケンスで 사용합니다。

1. 次のコマンドを使用して、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. 次のコマンドを使用して、PEM パッケージソフトを削除します。

```
* yum erase edb-pem-server
```

3. 次のコマンドを使用して、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、次のコマンドを使用して PEM サーバーを再構成します。

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