



**EDB**

# **Postgres Enterprise Manager - Upgrade Guide**

***Release 8.4.0***

**EDB**

**Apr 10, 2022**



# CONTENTS

- 1 Upgrading a PEM installation 3**
  - 1.1 Upgrading a PEM native package installation on a Linux host . . . . . 3
    - 1.1.1 Prerequisites to upgrade a PEM installation on Linux host . . . . . 4
    - 1.1.2 Upgrading a PEM agent native package installation . . . . . 5
    - 1.1.3 Upgrading a PEM server native package installation . . . . . 6
    - 1.1.4 Configuring the PEM server . . . . . 7
  - 1.2 Upgrading a PEM Installation on a Windows host . . . . . 7
    - 1.2.1 Upgrading a PEM agent on a Windows host . . . . . 7
    - 1.2.2 Upgrading the PEM server on a Windows host . . . . . 14
- 2 Upgrading the PEM backend Postgres database 27**
- 3 Upgrading SQL Profiler 37**
  - 3.1 Upgrading a SQL profiler native package installation on a Linux host . . . . . 37
  - 3.2 Upgrading a SQL Profiler installation on a Windows host . . . . . 39
- 4 Moving the PEM server 43**



This guide provides detailed information about upgrading the Postgres Enterprise Manager (PEM) Components:

- **Upgrading a PEM Installation** - Provides information about upgrading your PEM Server, PEM Agent and SQL Profiler from one major version to another.
- **Upgrading the PEM Backend Database** - Provides detailed information about upgrading the backend database, while maintaining the same version of the PEM Server.
- **Upgrading SQL Profiler** - Provides detailed information about upgrading the SQL Profiler.
- **Moving the PEM Server** – Provides detailed information about moving the PEM Server from one host to another host.
- **Troubleshooting** – Provides detailed information about troubleshooting the errors that you may encounter during PEM upgrade.

This document uses the term **Postgres** to mean either the PostgreSQL or the Advanced Server database or EDB Postgres Extended (formerly known as 2ndQPostgres).

---

**Note:** If you are upgrading PEM 7.x to the latest version and the PEM backend database is PostgreSQL or EDB Postgres Advanced Server version 10 or earlier then follow these steps: 1. Upgrade the PEM backend database 2. Upgrade the PEM Installation

---

If the PEM backend database is PostgreSQL or EDB Postgres Advanced Server version 11 or later then upgrading the PEM backend database is optional.



## UPGRADING A PEM INSTALLATION

The process of upgrading a PEM installation is platform-specific. You can update a PEM Agent or Server on a Linux platform by using the native packages and on a Windows platform by using the PEM graphical installer available for Windows.

This section provides detailed information about upgrading the PEM components:

- **Upgrading a PEM Installation on Linux** - Provides detailed information about upgrading your PEM Server, PEM Agent, and SQL Profiler from one major version to another on Linux platforms.
- **Upgrading a PEM Installation on Windows** - Provides information about upgrading your PEM Server, PEM Agent, and SQL Profiler from one major version to another on the Windows platform.

---

**Note:**

- Prior to PEM 7.8 release, PEM Agent or Server could be installed on Linux either by using the graphical installer or by using the RPMs. - From PEM version 7.8 onwards, PEM graphical installers for Linux are discontinued. To upgrade a PEM Agent or Server on a Linux host from any lower version to PEM 7.9 or higher versions, you must use native packages.
- 

Links to PEM installers and RPMs are available at the EDB website .

### 1.1 Upgrading a PEM native package installation on a Linux host

To upgrade PEM component software on Linux hosts, simply install a newer version of the PEM component native packages in the following order:

1. Invoke the PEM agent native package installer on each monitored node **except** the PEM server host.
2. Invoke the PEM server native package installer; it will upgrade **both** the PEM server and the PEM agent that resides on the PEM server host.

During an installation, the component installation will automatically detect an existing installation, and perform an upgrade. After upgrading the PEM agent and server, you can upgrade SQL Profiler if required; this step is platform-specific.

---

**Note:** If you have already configured or are planning to configure any shell/batch script run by a Linux agent that is upgraded from any lower version to 7.11 or later version, the user for the *batch\_script\_user* parameter must be specified in the *agent.cfg* file. It is strongly recommended that a non-root user is used to run the scripts. Using the root user may result in compromising the data security and operating system security. However, if you want to restore the *pemagent* to its original settings using a root user to run the scripts, then the *batch\_script\_user* parameter value must be set to *root*.

---

The following sections will walk you through the upgrade process on a Linux host, step-by-step.

### 1.1.1 Prerequisites to upgrade a PEM installation on Linux host

PEM is dependent on third-party components from the vendor repository, including the python3, libboost, openssl, snmp++, libcurl, etc. To ensure these components are up to date, you should update your operating system using following platform-specific commands. Minimum version require for openssl is 1.0.2k. If you are using a version of PostgreSQL or Advanced Server older than version 10, before the upgrade you must install the `libs` package for version 10 or above on the system where the PEM server is installed. Use the following platform-specific commands to install the `libs` version 10 or above on your host:

#### Prerequisites to upgrade a PEM installation on a CentOS or RHEL host

To upgrade packages on a CentOS or RHEL 7.x host

```
yum update
yum upgrade
```

To upgrade packages on a Rocky Linux or AlmaLinux or RHEL 8.x host

```
dnf update
dnf upgrade
```

To upgrade Advanced Server libs:

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

To upgrade PostgreSQL libs:

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

Where `<X>` is the PostgreSQL or Advanced Server version whose `libs` package you want to install.

#### Prerequisites to upgrade a PEM installation on a Debian or Ubuntu host

To upgrade packages on a Debian or Ubuntu host

```
apt-get update
apt-get upgrade
```

To upgrade Advanced Server libs:

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

To upgrade PostgreSQL libs:

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

Where `<X>` is the PostgreSQL or Advanced Server version whose `libs` package you want to install.

## **Prerequisites to upgrade a PEM installation on a SLES host**

To upgrade packages on a SLES host

```
zypper update  
zypper upgrade
```

To upgrade Advanced Server libs:

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

To upgrade PostgreSQL libs:

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

Where <X> is the PostgreSQL or Advanced Server version whose `libs` package you want to install.

## **1.1.2 Upgrading a PEM agent native package installation**

You can use native packages to upgrade existing PEM Agents initially installed using native packages. The upgrade process does not update the PEM agent configuration file. After installing the new agent, you must manually copy the configuration file of the existing agent to the new installation location.

### **Upgrading the PEM agent on a CentOS or RHEL host**

For CentOS or RHEL 7.x or RHEL 8.x:

To upgrade a PEM agent, use the following command:

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

For Rocky Linux or AlmaLinux or RHEL 8.x, you can also use the following command:

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

### **Upgrading a PEM agent on a Debian or Ubuntu host**

To upgrade a PEM Agent, use the following command:

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

### **Upgrading a PEM agent on a SLES host**

To upgrade a PEM Agent, use the following command:

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

### 1.1.3 Upgrading a PEM server native package installation

If you initially used native packages to install your PEM server, you can use native packages to upgrade your PEM server. The commands to upgrade are platform-specific; please refer to your platform's section below.

If you wish to upgrade a PEM server that is installed on a machine in an isolated network, you need to create a PEM repository on that machine before you upgrade the PEM server. For more information about creating a PEM repository on an isolated network, see [Creating a PEM Repository on an Isolated Network](#).

#### Upgrading a PEM server on a CentOS, Rocky Linux, AlmaLinux, or RHEL host

To use an RPM package to upgrade an existing RPM installation you must:

You can use the `yum` package manager to upgrade the installed version of the PEM server on CentOS/RHEL 7.x or Rocky Linux/AlmaLinux/RHEL 8.x:

```
yum upgrade edb-pem
```

You can also use the `dnf` command on Rocky Linux/AlmaLinux/RHEL 8.x:

```
dnf upgrade edb-pem
```

---

**Note:** If you are doing a fresh installation of the PEM Server on CentOS or RHEL 7.x host, the installer will install the `edb-python3-mod_wsgi` package along with the installation as the package is a requirement of the operating system. If you are *upgrading* the PEM Server on CentOS or RHEL 7.x host, the `mod_wsgi` package will be replaced by the `edb-python3-mod_wsgi` package to meet the requirements of the operating system.

---

After upgrading the PEM Server using `yum` or `dnf`, you must configure the PEM Server. For more detailed information see

*[Configuring the PEM server](#).*

#### Upgrading the PEM server on a Debian or Ubuntu host

You can use the `apt-get` package manager to upgrade the installed version of the PEM Server on supported versions of Debian or Ubuntu:

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

After upgrading the PEM Server with `apt-get`, you need to configure the PEM Server. For more detailed information see *[Configuring the PEM server](#)*.

#### Upgrading the PEM server on a SLES host

You can use the `zypper` package manager to upgrade the installed version of the PEM Server on supported versions of SLES Host:

```
zypper update edb-pem
```

After upgrading the PEM Server using `zypper`, you need to configure the PEM Server. For more detailed information see *[Configuring the PEM server](#)*.

### 1.1.4 Configuring the PEM server

After upgrading the PEM Server you can use the following command to configure the PEM Server:

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

The configure script uses the values from the old PEM Server configuration file while running the script.

After executing the PEM server configuration file, use your version-specific service control command to restart the `httpd` service.

For detailed information, please see the [Configuring the PEM Server on Linux Platforms](#).

---

**Note:** From PEM 7.11 and later, the configure script only requires a superuser password post the upgrade process.

---

## 1.2 Upgrading a PEM Installation on a Windows host

To upgrade PEM component software on Windows hosts, simply invoke a newer version of the PEM component installers in the following order:

1. Invoke the PEM agent installer on each monitored node **except** the PEM Server host.
2. Invoke the PEM Server installer; this installer will upgrade **both** the PEM Server and the PEM Agent that resides on the PEM Server host.

During an installation, the component installer will automatically detect an existing installation, and perform an upgrade. After upgrading the PEM Agent and Server, you can upgrade SQL Profiler if required; this step is platform-specific.

The following sections will walk you through the upgrade process on a Windows host, step-by-step.

### 1.2.1 Upgrading a PEM agent on a Windows host

To upgrade a system that is currently monitored by a PEM agent to a more-recent PEM agent, simply download and invoke a newer version of the PEM Agent installer on the system that the agent is monitoring.

You can invoke the installer by right-clicking on the downloaded installer's icon, and selecting **Run as Administrator**. The **PEM Agent Setup Wizard** opens, welcoming you.

Read and accept the **License Agreement** before clicking **Next** to continue.

The setup wizard will automatically detect an existing agent, and upgrade the installed version. Click **Next** to continue.

The **pemAgent** service account dialog may prompt you for the password of the account under which the PEM Agent service runs.

If prompted, provide the password, and click **Next** to continue.

When the **Ready to Install** dialog informs you that the installation is about to begin, click **Next** to continue. It will upgrade your PEM Agent to the latest version.

The setup wizard displays progress bars to inform you of each component that is being installed.

The **PEM Agent Setup Wizard** will inform you when the installation completes. Click **Finish** to exit the wizard and close the window.

After the installation completes, a window pops-up to restart the machine. Click **Yes** to restart the machine and the PEM Agent.

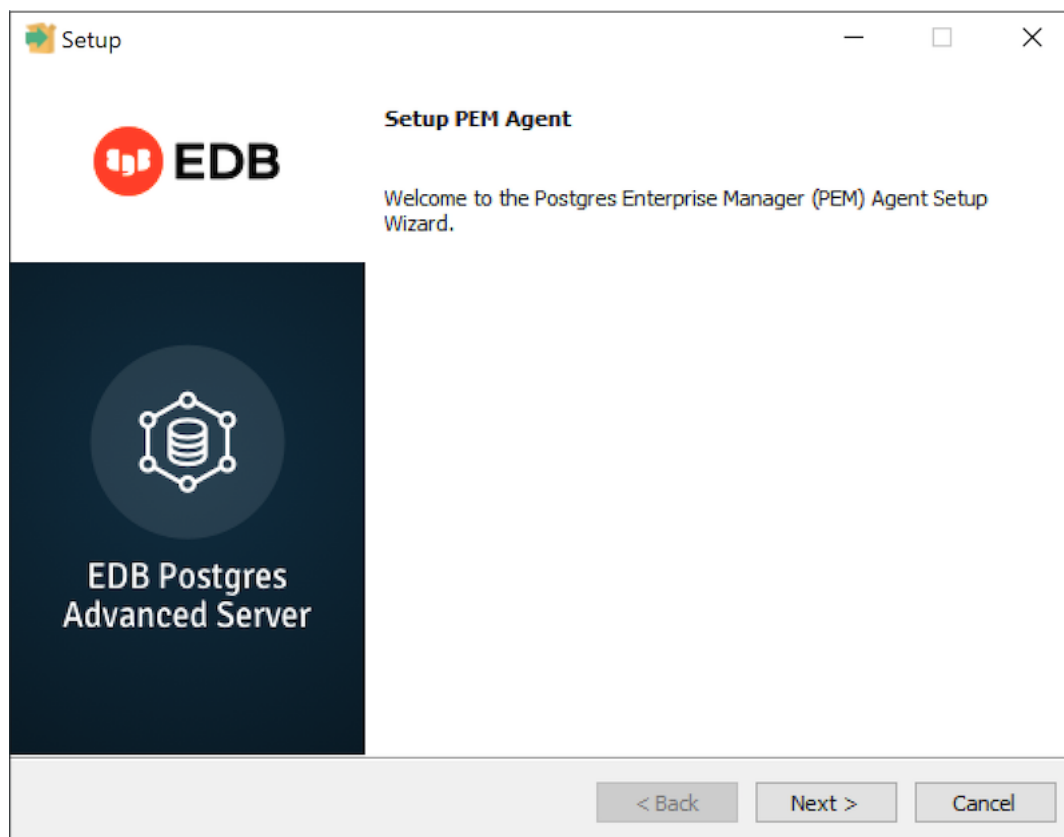


Fig. 1: The PEM Agent Installer - Welcome Window

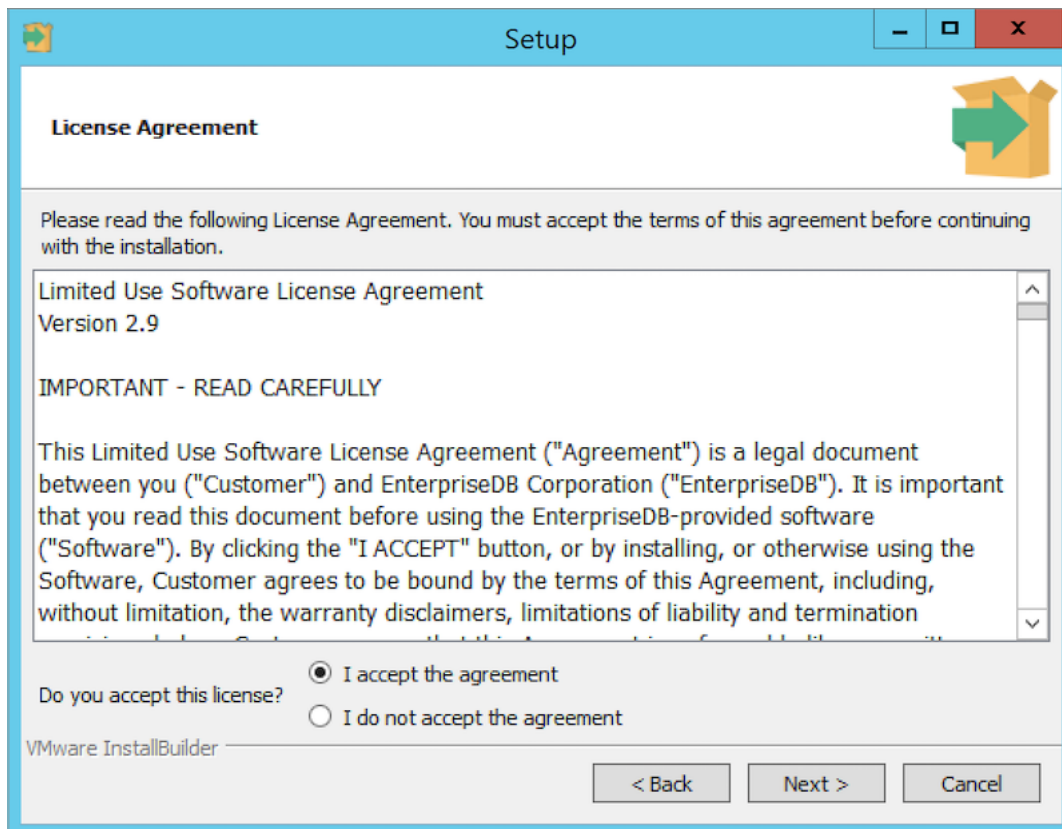


Fig. 2: The PEM Agent Installer - License Agreement

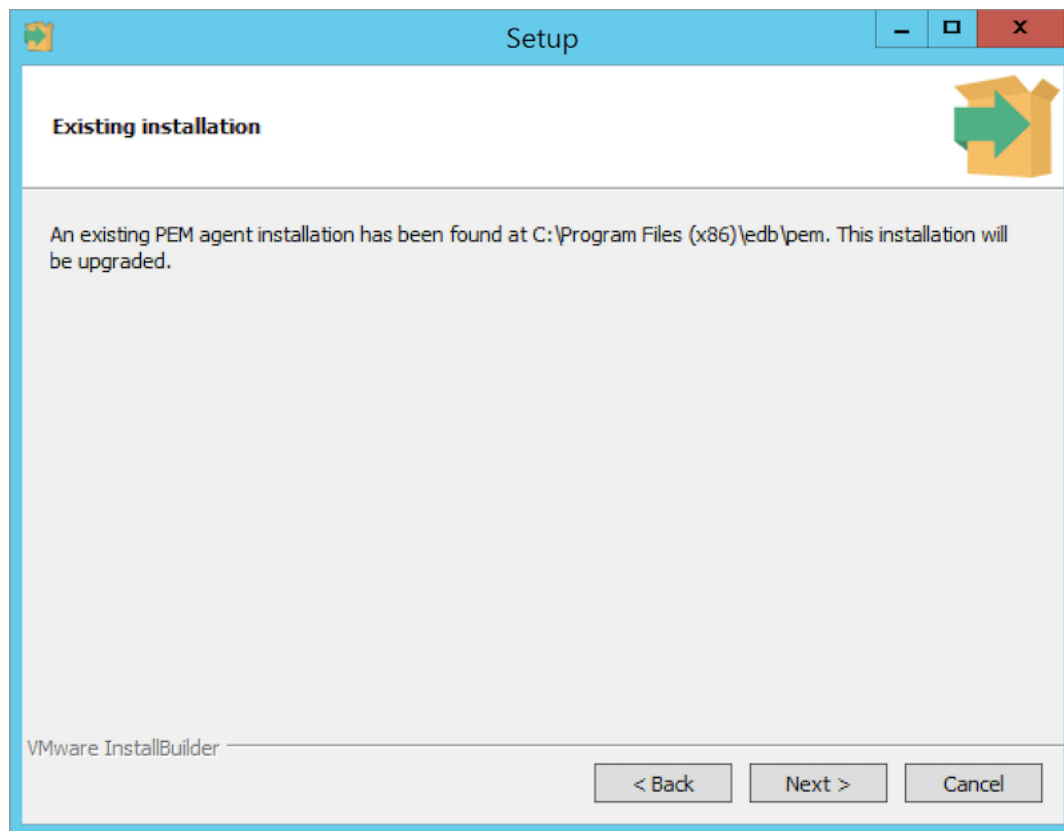


Fig. 3: The PEM Agent Installer - Existing Installation Location

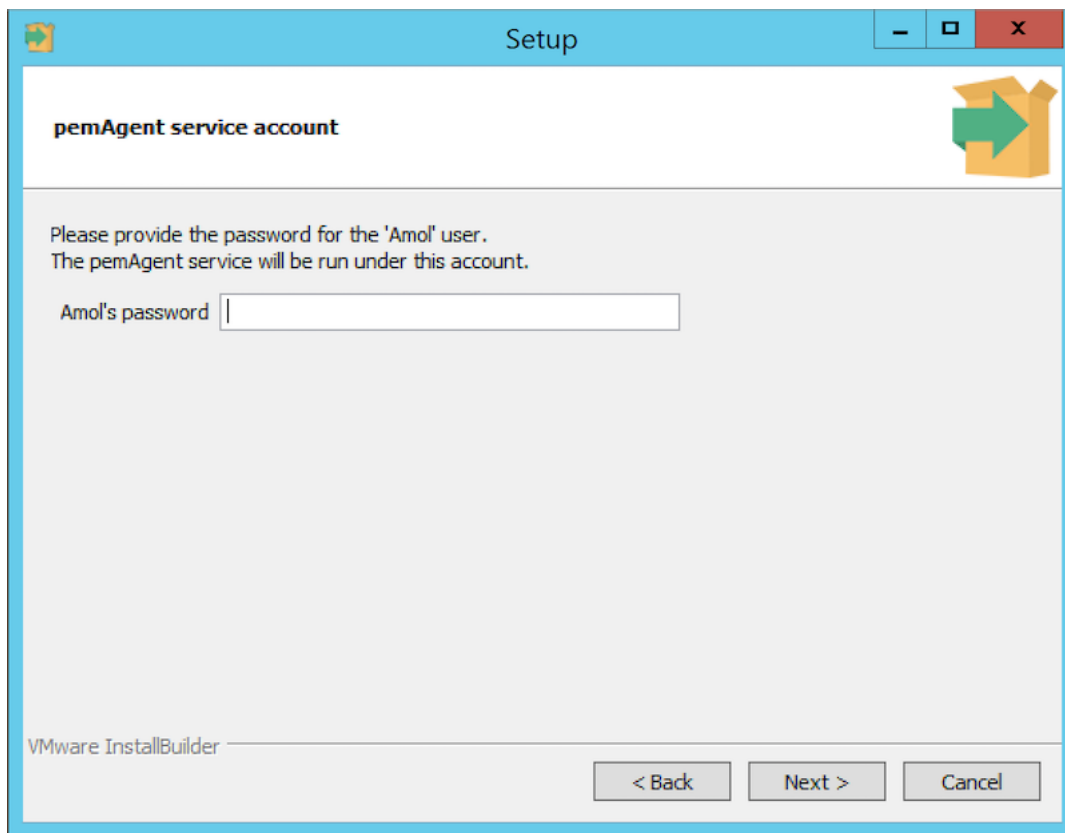


Fig. 4: The PEM Agent Installer - pemAgent Service Account password

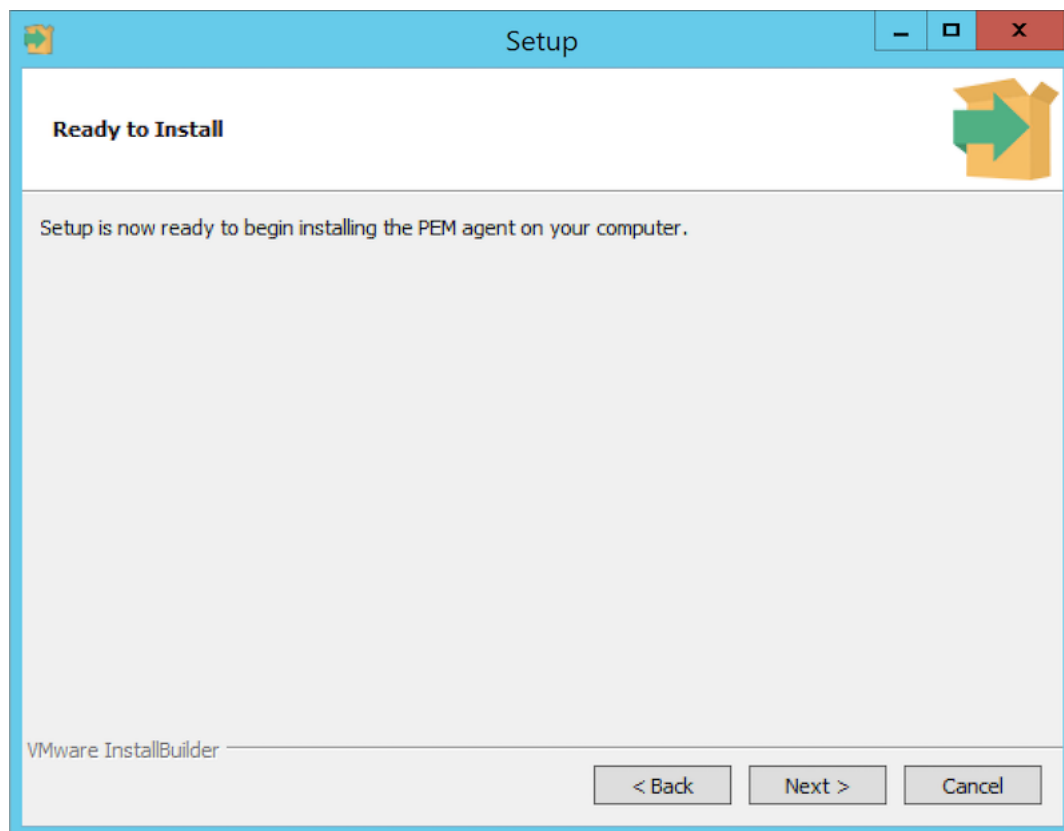


Fig. 5: The PEM Agent Installer - Ready to Install

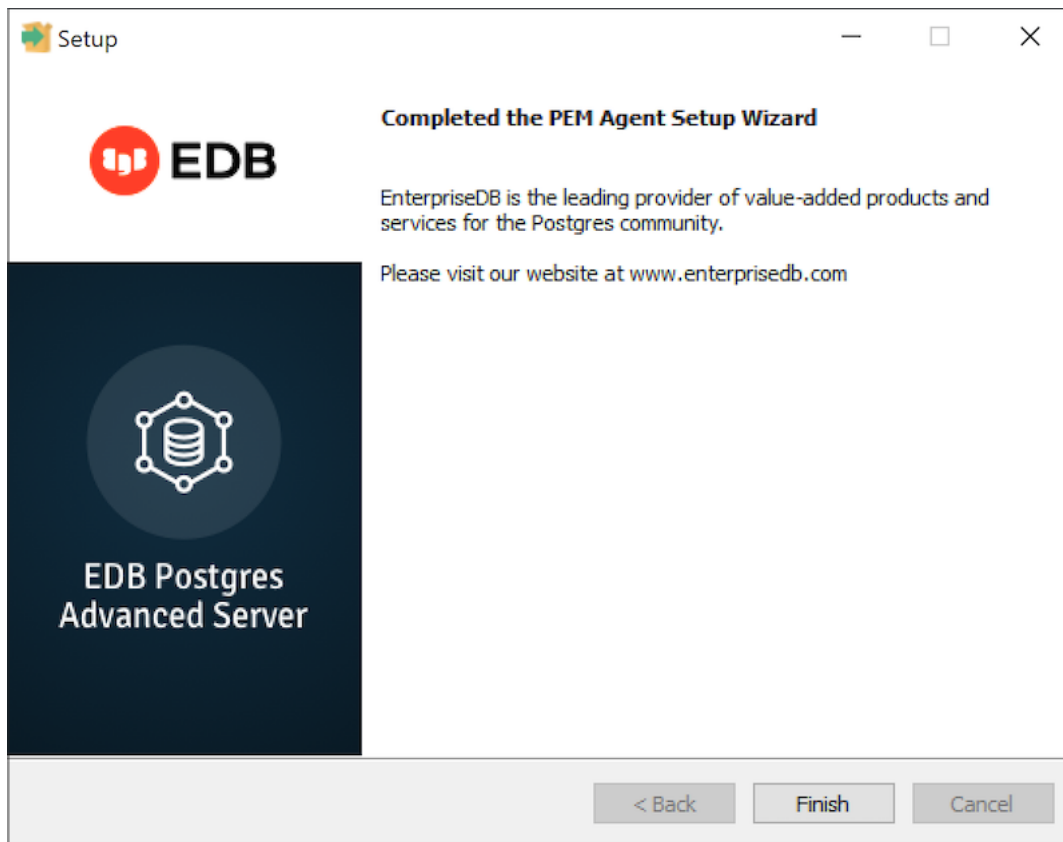


Fig. 6: The PEM Agent Installer - Finish

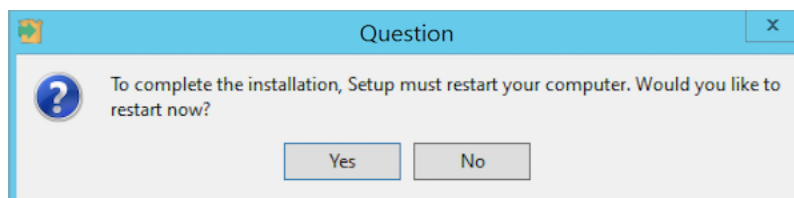


Fig. 7: The PEM Agent Installer - Restart pop-up

## 1.2.2 Upgrading the PEM server on a Windows host

The PEM Server installer facilitates upgrading directly between major versions of the PEM Server; you can upgrade directly from version 5.0 to version 7.16 without first upgrading to version 6.0.

You can invoke the installer by right-clicking on the downloaded installer's icon, and selecting **Run as Administrator**.

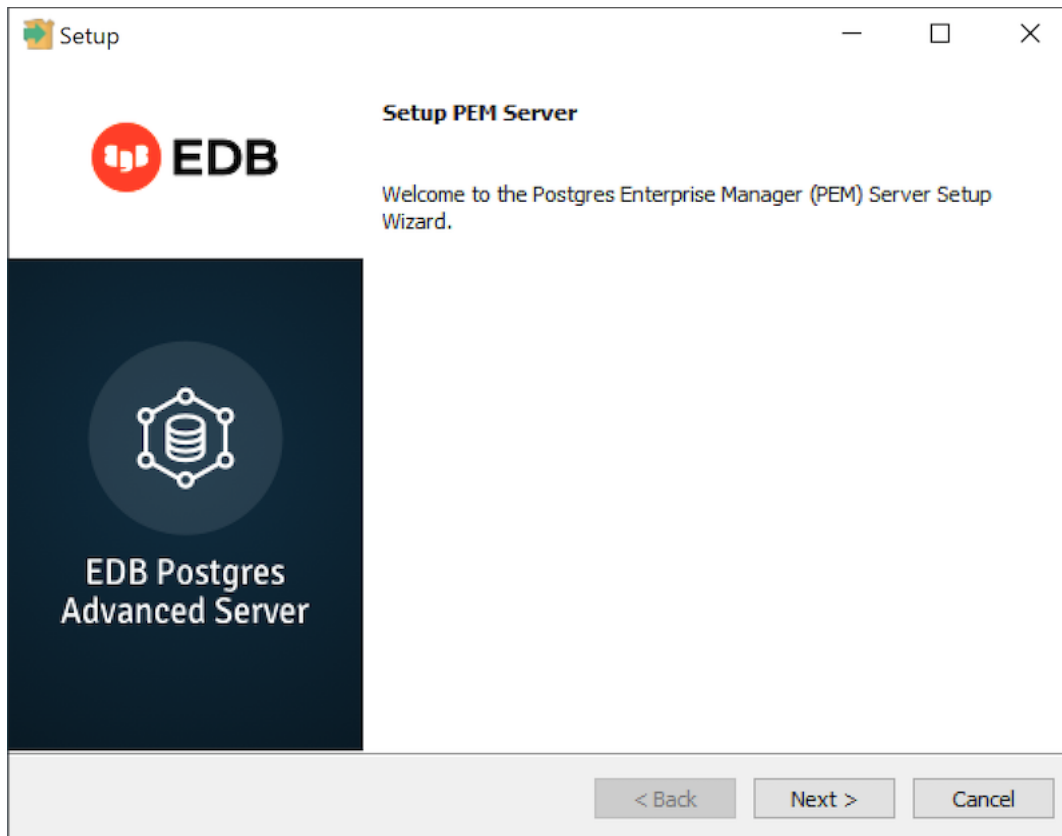


Fig. 8: The PEM Server Installer - Welcome Window

The PEM Server Setup Wizard welcomes you, as shown in the image. Click **Next** to continue to the License Agreement.

The PEM Server setup wizard will prompt you to accept the License Agreement. After reviewing the license agreement, check the radio button next to **I accept the agreement**, and click **Next** to continue to the Existing installation dialog.

The wizard will check the PEM Server host for an existing PEM Server installation; if the wizard locates an installation, it will perform an upgrade. Click **Next** to continue.

Before upgrading the PEM Server, the wizard will confirm that the requirements of the new PEM Server are present. If any supporting components are missing, or are a version that will not support the new PEM installation, the PEM installation wizard will inform you that it must upgrade the dependencies, and will invoke the required installers.

When the installation wizards complete the dependency upgrades, then a window pops-up asking whether you want to restart the machine or not.

Click on **No** to continue the upgrade process.

The wizard then opens the **Database Server Installation Details** dialog, prompting you for connection credentials for the database superuser of the PEM backend database. Provide:

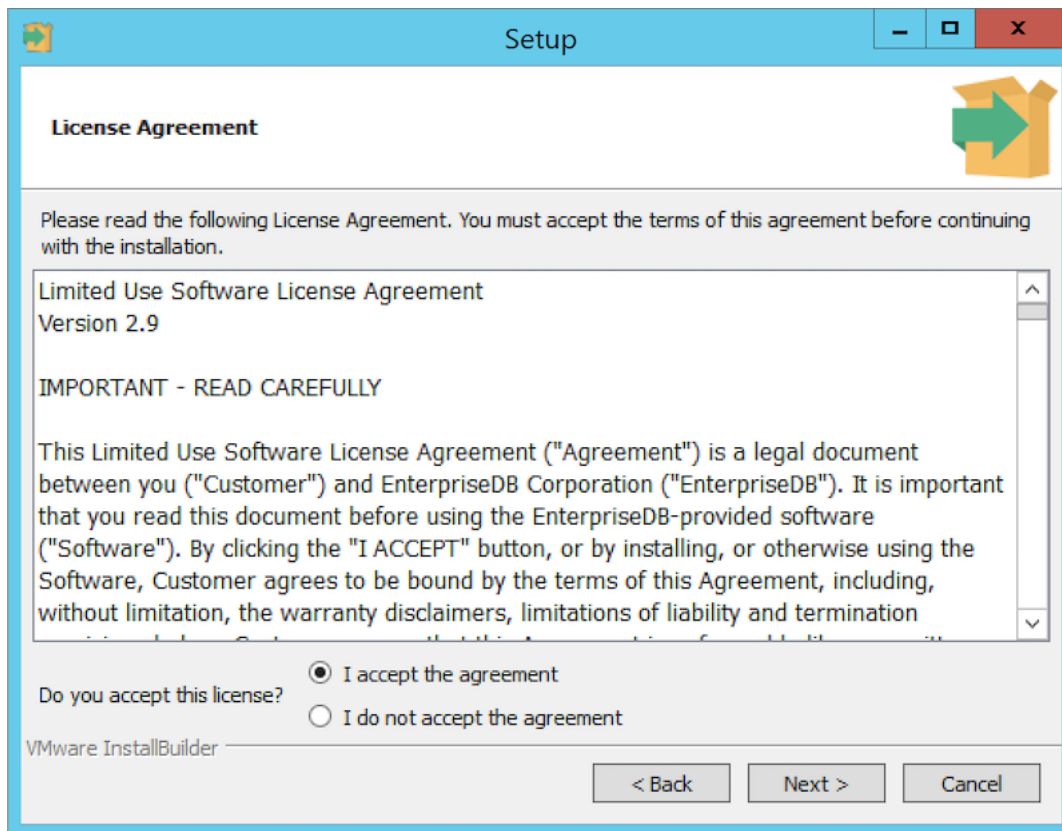


Fig. 9: The PEM Server Installer - License Agreement

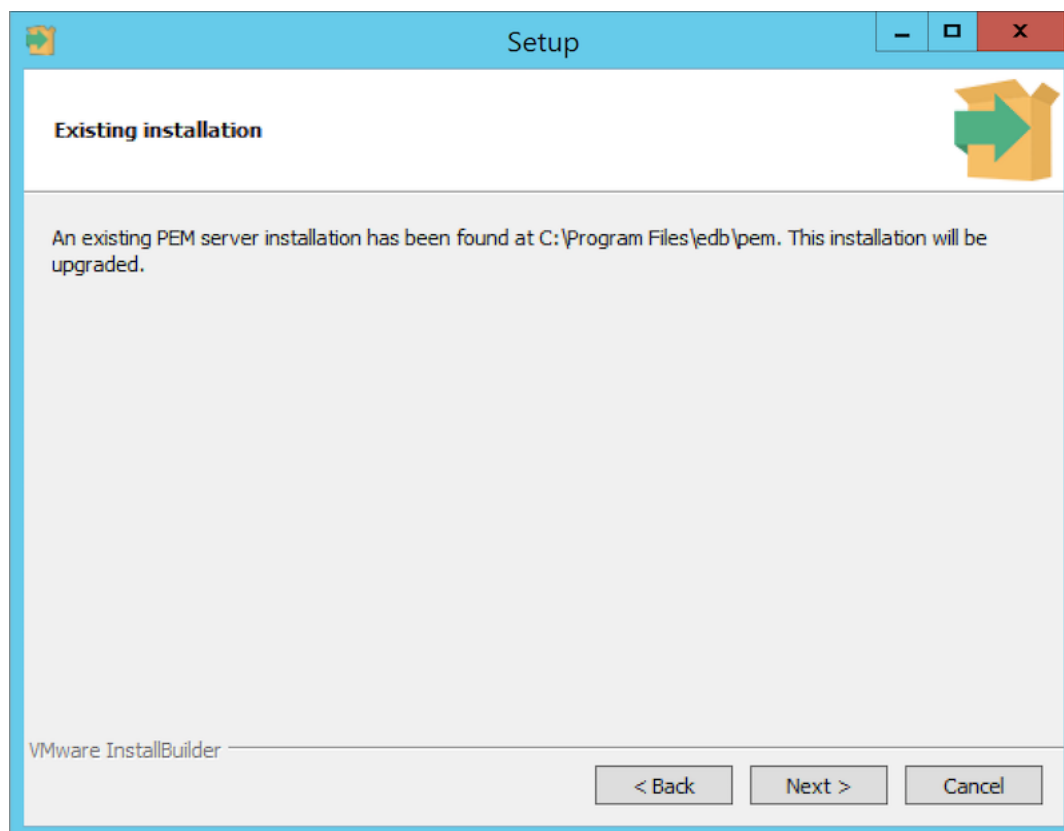


Fig. 10: The PEM Server Installer - Existing PEM Server Location

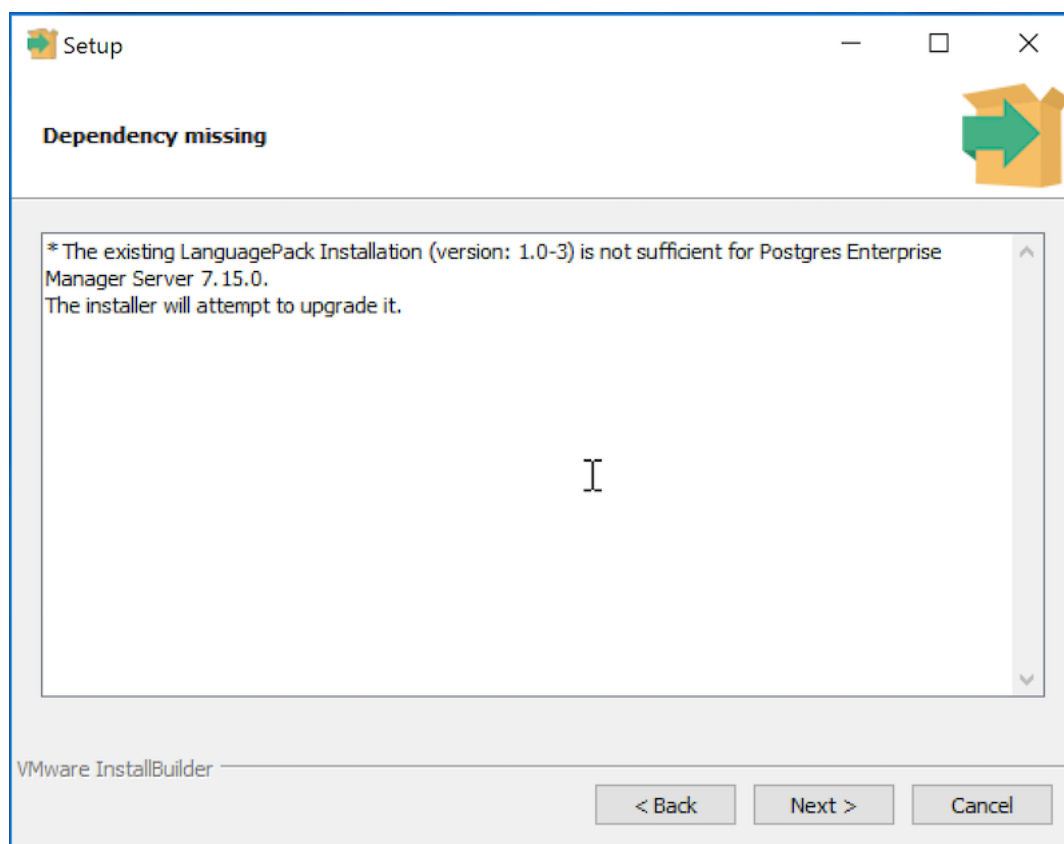


Fig. 11: The PEM Server Installer - Missing Dependencies

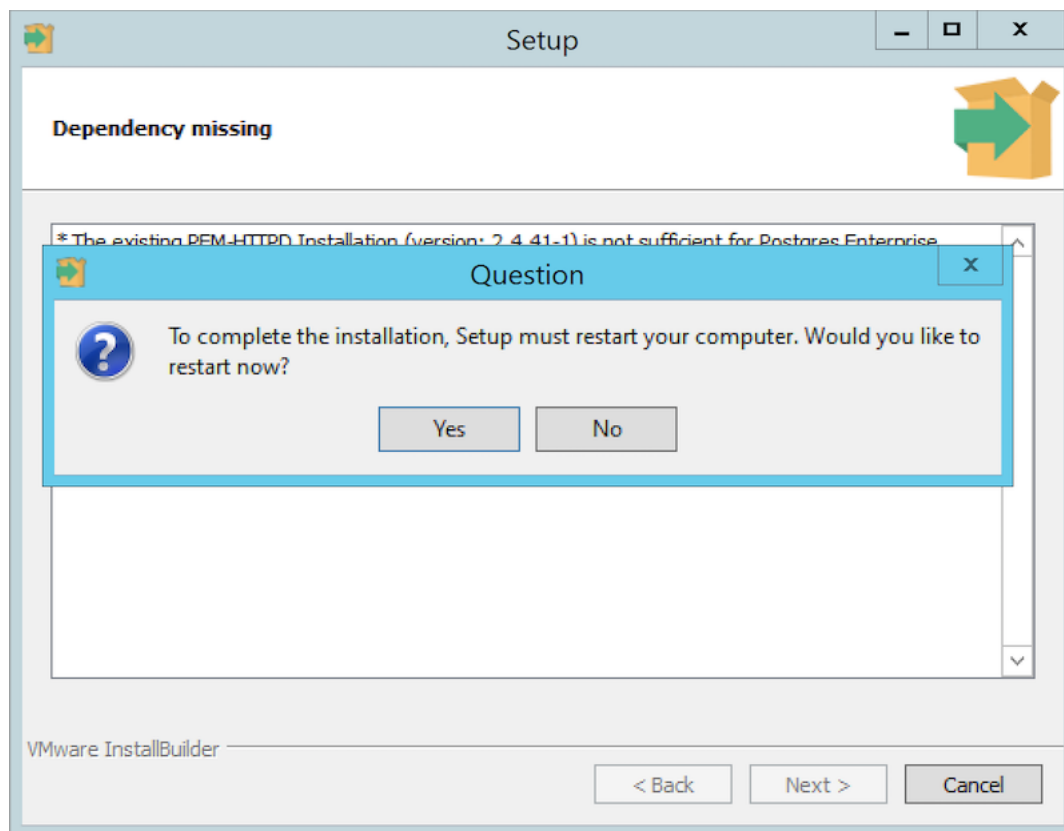


Fig. 12: The PEM Server Installer - Restart pop-up

- The name of the database superuser in the *User* field.
- The password associated with the database superuser in the *Password* field.

Click **Next** to continue.

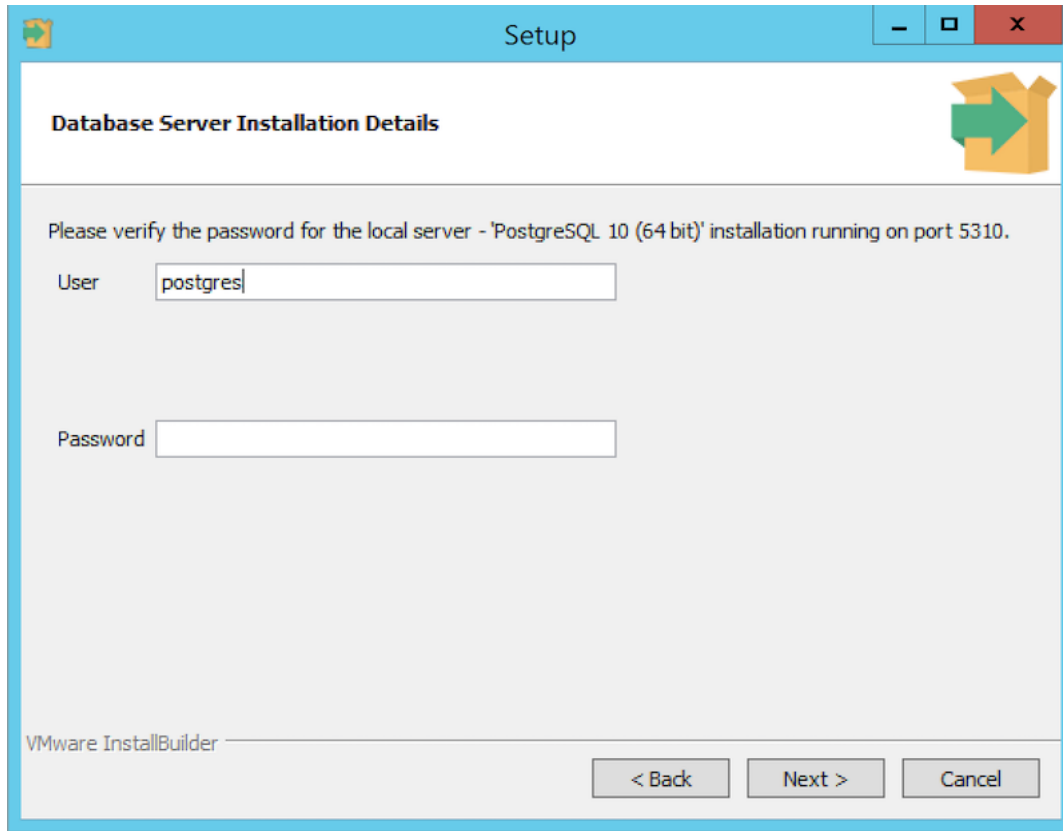


Fig. 13: The PEM Server Installer - Database Server Installation Details

The `pemAgent` service account dialog may prompt you for the password of the account under which the PEM agent service runs.

If prompted, provide the password, and click **Next** to continue.

The **Ready to Install** dialog will inform you that the setup wizard is ready to perform the installation. Click **Next** to start the installation.

During the installation, progress bars will keep you informed of the progress of the update.

After upgrading the PEM Server (and the agent that resides on the same host as the PEM server) and configuring the web service, the PEM setup wizard notifies you of the port on which the service is listening. Use this port number when connecting to the PEM Server with the PEM client.

Click **OK** to close the **Info** popup. The PEM server setup wizard informs you that the installation is complete.

If the window pops-up asking to restart the machine, then click on **Yes** to restart the machine and hence the `httpd` service.

If you have installed the PEM backend database server and PEM-HTTPD on different hosts, then you must run the PEM Server installer twice – once on each host. Extract the language pack installer, and install it on the host of PEM-HTTPD before invoking the PEM installer. Include the following keywords when invoking the installer to extract the language pack:

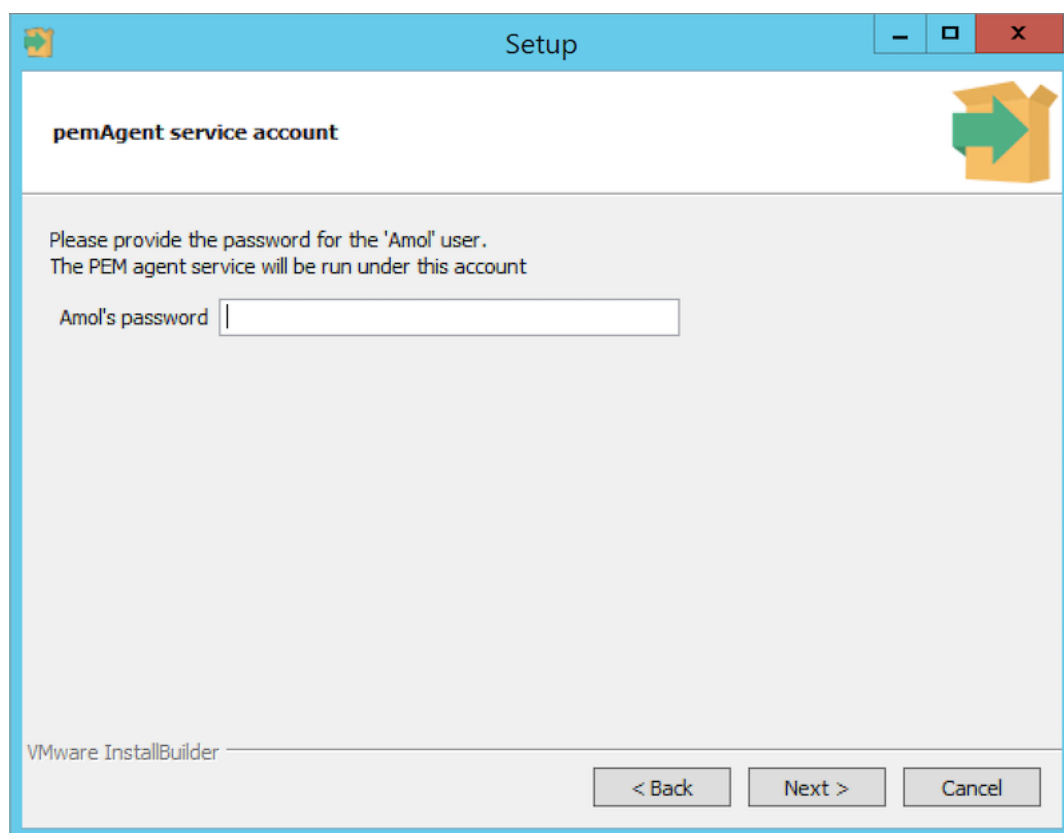


Fig. 14: The PEM Server Installer - pemAgent Service Account password

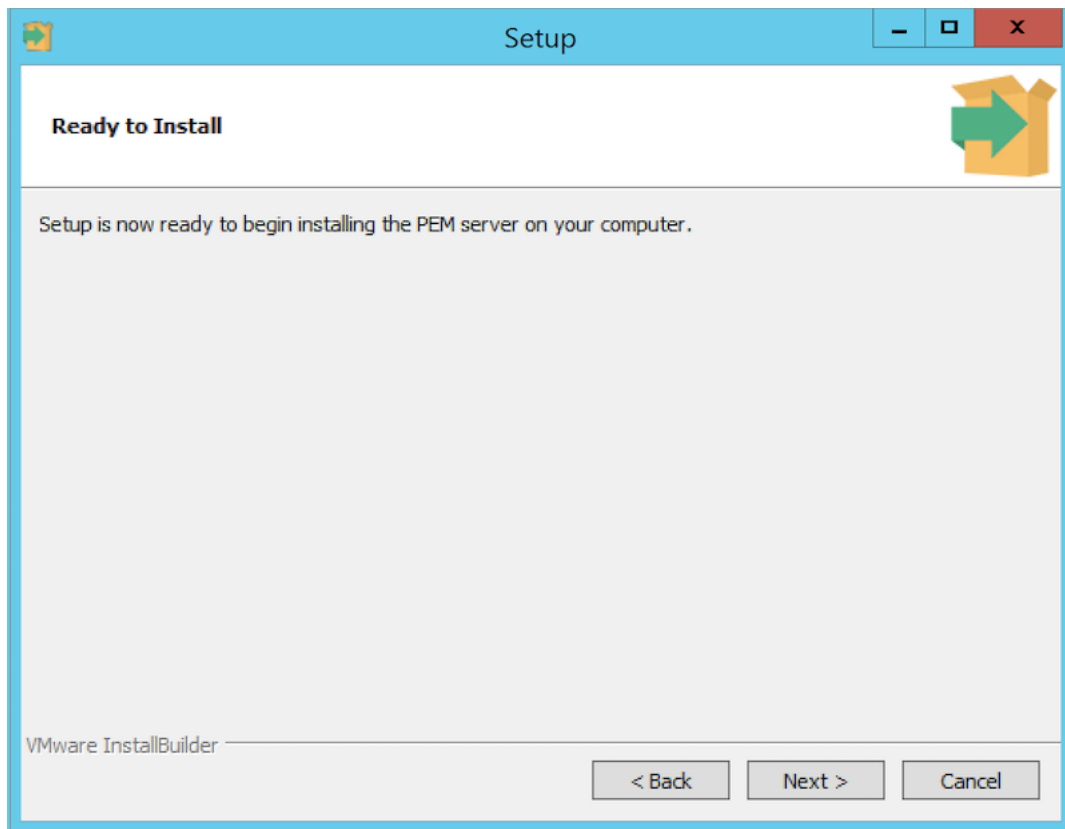


Fig. 15: The PEM Server Installer - Ready to Install

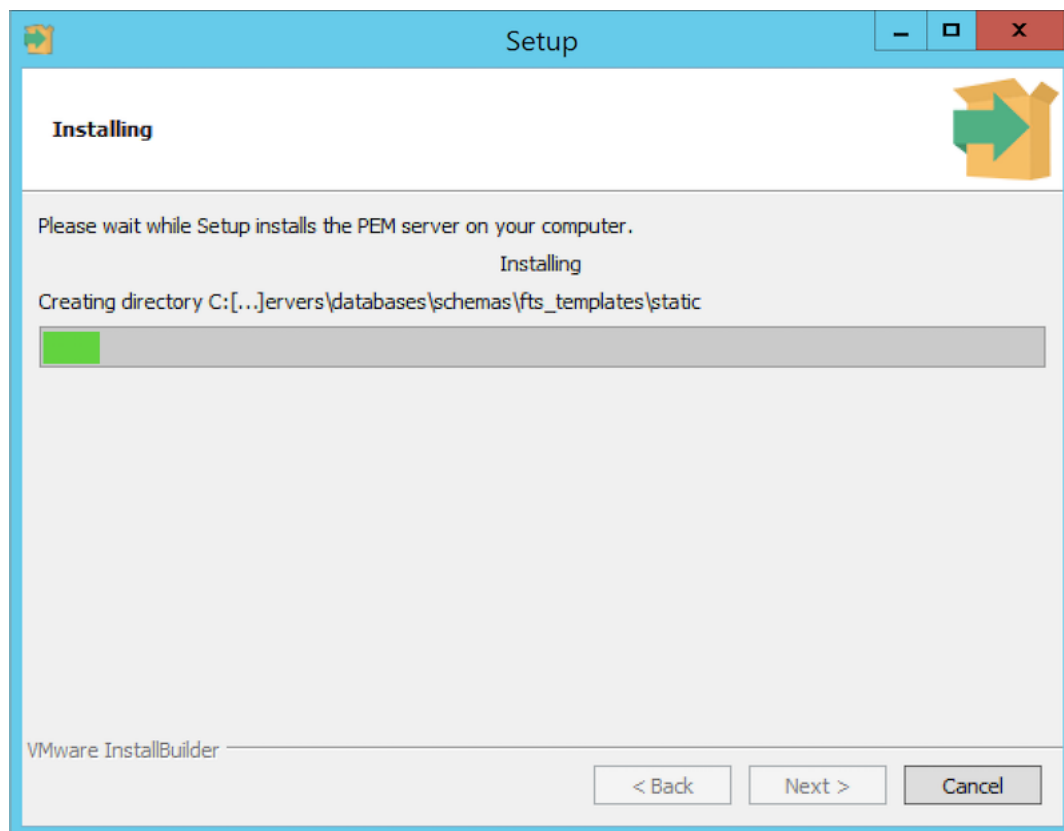


Fig. 16: The PEM Server installation in progress

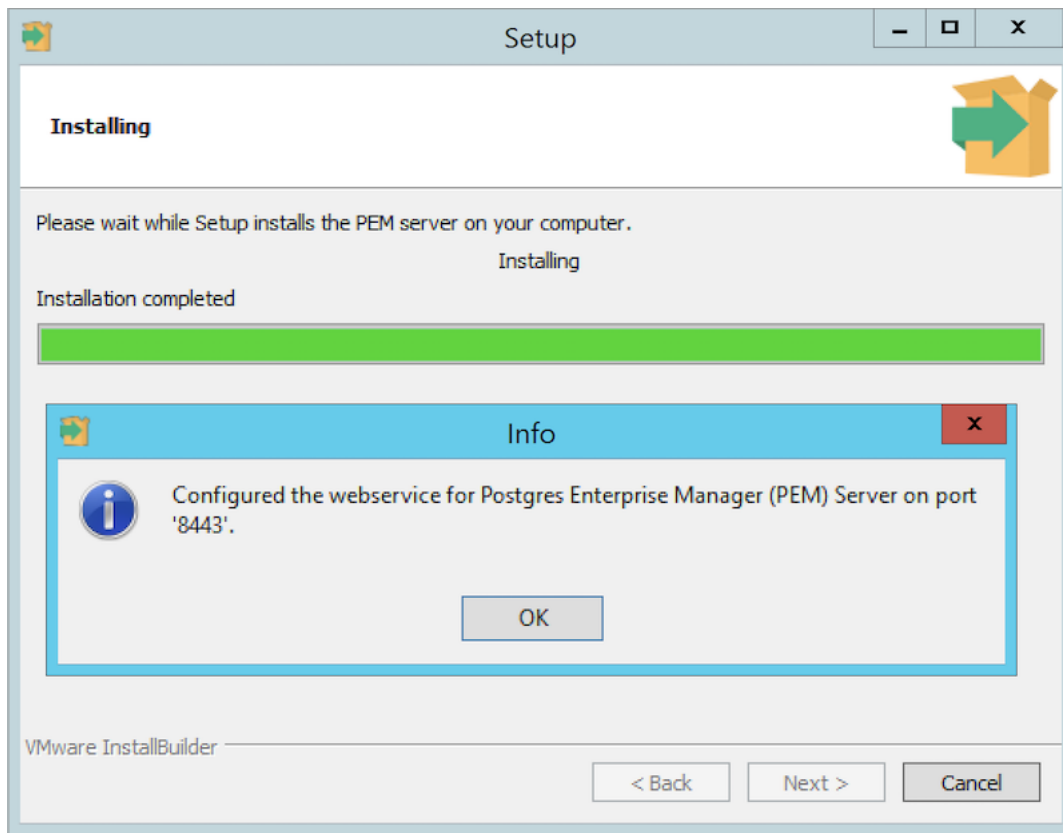


Fig. 17: The PEM Server Installer - the PEM Web Service configured

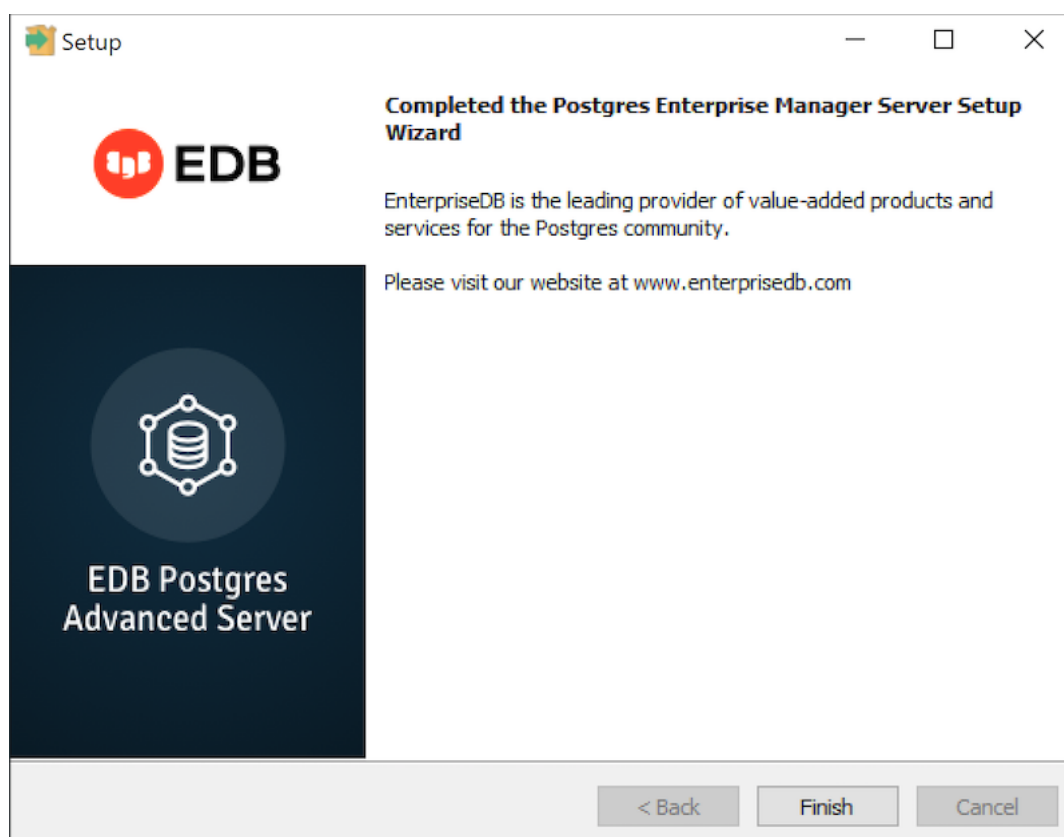


Fig. 18: The PEM Server Installer - Upgrade Completion

`--extract-languagepack <path>`

Where `<path>` specifies an existing path for extracting the language pack installer.

**Note:** By default EDB Language Pack is installed in `C:\edb\languagepack\v1`.

If you are upgrading the PEM Server via StackBuilder Plus then you might face the error shown below; after displaying the error, PEM will say that installation is completed. Please note that the installation is not done and you will need to do the installation by invoking the installer file from the location where it is downloaded.

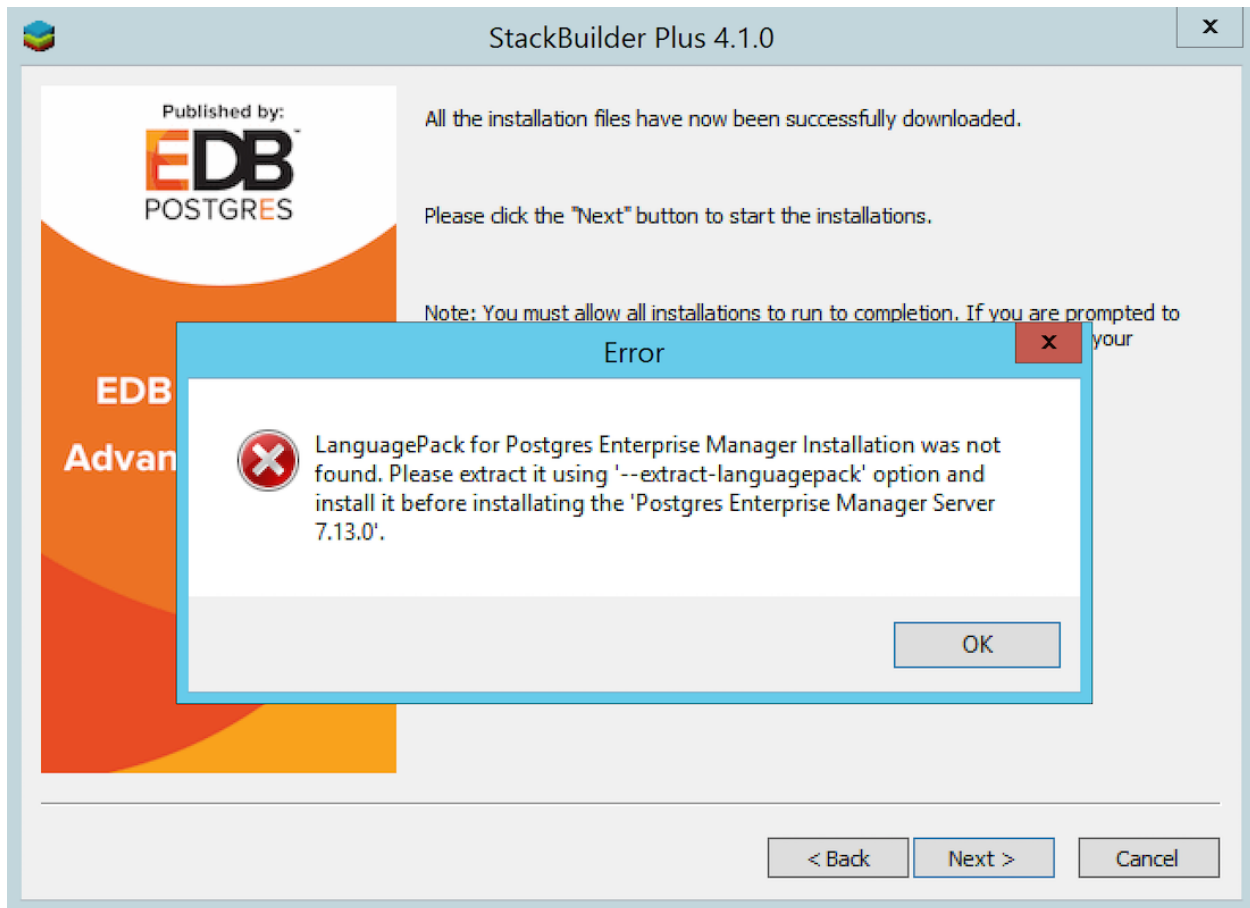


Fig. 19: The PEM Server upgrade error

After upgrading the PEM Server, you may wish to upgrade the backend database to a more recent version. For information about upgrading the backend database, see [Upgrading the Backend Postgres Database](#).



## UPGRADING THE PEM BACKEND POSTGRES DATABASE

If you are updating both PEM components and the PEM backend database, you should perform PEM component updates (the server and Agent) before updating the backend database. For more information about updating PEM component software, see [Upgrading a PEM installation](#).

---

**Note:** From PEM 8.0 onwards, the PostgreSQL or EPAS version 11 or higher are only supported as backend database server. Hence if your backend database server is lower than version 11 then first you need to upgrade your backend database server and then upgrade the PEM components.

---

The update process described in this section uses the `pg_upgrade` utility to migrate from one version of the backend server to a more recent version. `pg_upgrade` facilitates migration between any supported version of Postgres, and any subsequent release of Postgres that is supported on the same platform.

---

**Note:** If the source PEM Server is lower than the 7.16 version, then you need to replace the following functions before you run `pg_upgrade`:

---

- The `abstime`, `reltime`, and `tinterval` datatypes are deprecated from Postgres version 12 or later, hence to replace those datatypes with `timestampz` data type use below command:

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

(continues on next page)

(continued from previous page)

```

        ALTER COLUMN valuntil SET DATA TYPE timestamptz;
    -- Now update the pem.probe_column itself
    UPDATE pem.probe_column
    SET sql_data_type = 'timestamptz'
    WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
END;
$$ LANGUAGE 'plpgsql';

```

- Replace the below function to avoid any alert errors:

```

CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
template_id pem.alert_template.id%type, params text[]
)
RETURNS bool AS $FUNC$
DECLARE
    res bool := TRUE;
BEGIN
    /*
    * During restoring the pem database, it does not maintain the order while
    * inserting data in the table, and uses the sort table based on the
    * names.
    * Hence - we need to check the foreign key constraint is present before
    * validating these values.
    */
    IF EXISTS(
        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;

```

(continues on next page)

(continued from previous page)

```

RETURN res;
END
$FUNC$ LANGUAGE plpgsql;

```

`pg_upgrade` supports a transfer of data between servers of the same type. For example, you can use `pg_upgrade` to move data from a PostgreSQL 9.6 backend database to a PostgreSQL 11 backend database, but not to an Advanced Server 11 backend database. If you wish to migrate to a different type of backend database (i.e from a PostgreSQL server to Advanced Server), see *Moving the Postgres Enterprise Manager Server*.

You can find more information about using `pg_upgrade` at:

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

1. Download and invoke the updated installer; installers for PostgreSQL and Advanced Server are available through the EDB website:

<https://www.enterprisedb.com/software-downloads-postgres>

After downloading the installer for the server version to which you will be upgrading, invoke the installer on the host of the PEM server. Follow the onscreen instructions of the installation wizard to configure and install the Postgres server.

You can optionally use a custom-built PostgreSQL server as a host of the PEM backend database. Note that if you are upgrading from a PostgreSQL backend database listening on port 5432, the new server must be configured to listen on a different port.

2. Configure SSL utilities on the new server. The new backend database must be running the same version of `sslutils` that the current backend database is running; you can download the SSL Utils package that is used in EDB installers at:

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

You are *not* required to manually add the `sslutils` extension when using the Advanced Server as the new backend database. The process of configuring `sslutils` is platform-specific.

#### OnLinux

- On an Advanced Server backend database, the `sslutils` extension is installed by default.
- If you are using a PostgreSQL as PEM backend database, ensure you have access to the PostgreSQL community repository, and use the command:

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

Where `<X>` is the server version.

- If you are using a EDB one-click installer of PostgreSQL as PEM backend database

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

Set the value of `PATH` so it can locate the `pg_config` program

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

Move into the sslutils folder, and enter

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

Use psql to create the sslutils extension

```
CREATE EXTENSION sslutils
```

Please note that Debian 10 and Ubuntu 20 has increased the requirements to accept the certificates due to security reason. If a user wants to install the PEM Agent on any of the machines, they must upgrade *sslutils* to 1.3 where 4096 bit RSA key and sha256 signature algorithm support has added. If the user does not upgrade *sslutils* to 1.3, then PEM Agent may fail to connect to the PEM backend database server, and it might log the error *ca md too weak* .

### OnWindows

- *sslutils* must be compiled on the new backend database with the same compiler that was used to compile *sslutils* on the original backend database. If you are moving to a PostgreSQL database that was installed using a PostgreSQL one-click installer (from EDB) or an Advanced Server installer, use Visual Studio to build *sslutils* . If you are upgrading to PostgreSQL 9.6 or later, use Visual Studio 2010.

For detailed information about building a specific version of Postgres on Windows, please consult the core documentation for that version. Core documentation is available at the PostgreSQL project website at:

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

- While specific details of the process will vary by platform and compiler, the basic steps on each platform are the same. The example that follows demonstrates compiling OpenSSL support for PostgreSQL on a 32-bit Windows system.

Before compiling the OpenSSL extension, you must locate and install OpenSSL for your version of Windows. Before invoking the OpenSSL installer you may be required to download and install a pre-requisite redistributable (such as *vcredist\_x86.exe* ).

After installing OpenSSL, download and unpack the *sslutils* utility package available at:

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

- Copy the unpacked *sslutils* folder to the Postgres installation directory (i.e. *C:\ProgramFilesPostgreSQL<x.x>* )

- Open the Visual Studio command line, and navigate into the *sslutils* directory. Use the following commands to build *sslutils* :

```
SET USE_PGXS=1

SET 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
```

Where:

- *path\_to\_gettext* specifies the location of the *GETTEXT* library and header files.
- *path\_to\_openssl* specifies the location of the openssl library and header files.
- *path\_to\_pg\_installation\_dir* specifies the location of the Postgres installation.
- For example, the following set of commands builds OpenSSL support into the PostgreSQL 11 server:

```
SET USE_PGXS=1

SET 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
```

- When the build completes, the *sslutils* directory will contain the following files:

```
sslutils-1.3.sql
sslutils-unpackaged-1.3.sql
sslutils-pemagent.sql.in
sslutils.dll
```

- Copy the compiled *sslutils* files to the appropriate directory for your installation; for example:

```
COPY sslutils*.sql "%PGPATH%\share\extension\  
COPY sslutils.dll "%PGPATH%\lib\  
COPY sslutils.control "%PGPATH%\share\extension\  

```

3. Stop the services of both the old backend database and the new backend database.

On RHEL or CentOS 7.x or 8.x, open a command line and assume the identity of a superuser. Enter the command:

```
systemctl <service_name> stop
```

Where *<service\_name>* specifies the name of the Postgres service.

On Windows, you can use the *Services* dialog to control the service. To open the *Services* dialog, navigate through the *Control Panel* to the *System and Security* menu. Select *Administrative Tools* , and then double-click the *Services* icon. When the *Services* dialog opens, highlight the service name in the list, and use the option provided on the dialog to stop the service.

4. Use the `pg_upgrade` utility to perform an in-place transfer of existing data between the old backend database and the new backend database. If your server is configured to enforce md5 authentication, you may need to add an entry to the `.pgpass` file that specifies the connection properties (and password) for the database superuser, or modify the `pg_hba.conf` file to allow trust connections before invoking `pg_upgrade` . For more information about creating an entry in the `.pgpass` file, please see the PostgreSQL core documentation, available at:

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

During the upgrade process, `pg_upgrade` will write a series of log files. The cluster owner must invoke `pg_upgrade` from a directory in which they have write privileges. If the upgrade completes successfully, `pg_upgrade` will remove the log files when the upgrade completes. To instruct `pg_upgrade` to not delete the upgrade log files, include the `--retain` keyword when invoking `pg_upgrade` .

To invoke `pg_upgrade` , assume the identity of the cluster owner, navigate into a directory in which the cluster owner has write privileges, and execute the command:

```
<path_to_pg_upgrade> pg_upgrade  
-d <old_data_dir_path>  
-D <new_data_dir_path>  
-b <old_bin_dir_path> -B <new_bin_dir_path>  
-p <old_port> -P <new_port>  
-u <user_name>
```

Where:

- *path\_to\_pg\_upgrade* specifies the location of the *pg\_upgrade* utility. By default, *pg\_upgrade* is installed in the *bin* directory under your Postgres directory. - *old\_data\_dir\_path* specifies the complete path to the data directory of the old backend database. - *new\_data\_dir\_path* specifies the complete path to the data directory of the new backend database. - *old\_bin\_dir\_path* specifies the complete path to the bin directory of the old backend database. - *new\_bin\_dir\_path* specifies the complete path to the bin directory of the old backend database. - *old\_port* specifies the port on which the old server is listening. - *new\_port* specifies the port on which the new server is listening. - *user\_name* specifies the name of the cluster owner.

For example, the following command:

```
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
```

Instructs *pg\_upgrade* to migrate the PEM database from PostgreSQL 9.6 to PostgreSQL 11 on a Windows system (if the backend databases are installed in their default locations).

Once invoked, *pg\_upgrade* will perform consistency checks before moving the data to the new backend database. When the upgrade is finished, *pg\_upgrade* will notify you that the upgrade is complete.

For detailed information about using *pg\_upgrade* options, or troubleshooting the upgrade process, please see:

[<http://www.postgresql.org/docs/current/static/pgupgrade.html>](http://www.postgresql.org/docs/current/static/pgupgrade.html)

5. Copy the following certificate files from the data directory of the old backend database to the data directory of the new backend database:

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

Once in place on the target server, the files should have the (platform-specific) permissions described below:

#### Permissions and Ownership on Linux

| File Name          | Owner    | Permissions |
|--------------------|----------|-------------|
| ca_certificate.crt | postgres | -rw-----    |
| ca_key.key         | postgres | -rw-----    |
| root.crt           | postgres | -rw-----    |
| root.crl           | postgres | -rw-----    |
| server.key         | postgres | -rw-----    |
| server.crt         | postgres | -rw-r--r--  |

On Linux, the certificate files must be owned by *postgres* . You can use the following command at the command line to modify the ownership of the files:

```
chown postgres <file_name>
```

Where *file\_name* specifies the name of the certificate file.

The *server.crt* file may only be modified by the owner of the file, but may be read by any user. You can use the following command to set the file permissions for the *server.crt* file:

```
chmod 644 server.crt
```

The other certificate files may only be modified or read by the owner of the file. You can use the following command to set the file permissions:

```
chmod 600 <file_name>
```

Where *file\_name* specifies the name of the file.

#### **Permissions and Ownership on Windows**

On Windows, the certificate files moved from the source host must be owned by the service account that performed the PEM server and backend database installation on the target host. If you invoked the PEM server and Postgres installer using the *Run as Administrator* option (selected from the context menu of the installer), the owner of the certificate files will be *Administrators* .

To review and modify file permissions on Windows, right-click on the file name, and select *Properties* .

![The Security tab](../images/upgrade\_backendDB\_permission\_Windows\_security.png) \*\* The Security tab \*\*

Navigate to the *Security* tab and highlight a *Group or user name* to view the assigned permissions. Select *Edit* or *Advanced* to access dialogs that allow you to modify the permissions associated with the selected user.

6. The `postgresql.conf` file contains parameter settings that specify server behavior. You will need to modify the `postgresql.conf` file on the new server to match the configuration specified in the `postgresql.conf` file of the old server.

By default, the `postgresql.conf` file is located:

- For Postgres version lower than 10 on Linux, in `/opt/PostgreSQL/<X>/data` - For Postgres version 10 or higher when installed with graphical installers on Linux, in `/opt/PostgreSQL/<X>/data` - For Postgres version 10 or higher when installed with an RPM on Linux, in `/usr/pgsql/<X>/data` - For any Postgres version on Windows, in `C:\Program Files\PostgreSQL\<X>\data`

Where, `<X>` is the version of Postgres on your system.

Use your choice of editor to update the `postgresql.conf` file of the new server. Modify the following parameters:

- The `port` parameter to listen on the port monitored by your original backend database (typically, 5432). - The `ssl` parameter should be set to on.

You must also ensure that the following parameters are enabled. If the parameters are commented out, remove the pound sign from in front of each `postgresql.conf` file entry:

```

• `ssl_cert_file = 'server.crt' # (change requires restart)` - `ssl_key_
  file = 'server.key' # (change requires restart)` - `ssl_ca_file =
  'root.crt' # (change requires restart)` - `ssl_crl_file = 'root.crl
  '`

```

Your installation may have other parameter settings that require modification to ensure that the new backend database behaves in a manner comparable to the old backend database. Review the `postgresql.conf` files carefully to ensure that the configuration of the new server matches the configuration of the old server.

7. The `pg_hba.conf` file contains parameter settings that specify how the server will enforce host-based authentication. When you install the PEM server, the installer modifies the `pg_hba.conf` file, adding entries to the top of the file:

```

# 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

```

By default, the `pg_hba.conf` file is located at the following location:

- For Postgres version lower than 10 on Linux, in `/opt/PostgreSQL/<X>/data` - For Postgres version 10 or higher when installed with graphical installers on Linux, in `/opt/PostgreSQL/<X>/data` - For Postgres version 10 or higher when installed with RPMs on Linux, in `/var/lib/pgsql/<X>/data` - For Advanced Server version 10 or higher when installed with RPMs on Linux, in `/var/lib/edb/as<X>/data` - For any Postgres version on Windows, in `C:\Program Files\PostgreSQL\<X>\data`

Where, `<X>` is the version of Postgres on your system.

Using your editor of choice, copy the entries from the `pg_hba.conf` file of the old server to the `pg_hba.conf` file for the new server.

8. Restart the service of the new backend database.

On RHEL or CentOS 7.x or 8.x, open a command line and assume the identity of a superuser. Enter the command:

```
systemctl stop <service_name>
```

Where *service\_name* is the name of the backend database server.

If you are using Windows, you can use the *Services* dialog to control the service. To open the *Services* dialog, navigate through the *Control Panel* to the *System and Security* menu. Select *Administrative Tools*, and then double-click the *Services* icon. When the *Services* dialog opens, highlight the service name in the list, and use the option provided on the dialog to start the service.

## UPGRADING SQL PROFILER

The process of upgrading SQL Profiler is platform-specific. You can update SQL Profiler on a Linux platform by using the native packages and on a Windows platform by using the graphical installer available for Windows.

This section provides detailed information about upgrading the SQL Profiler:

- **Upgrading SQL Profiler Installation on Linux** – Provides detailed information about upgrading your SQL Profiler on Linux platforms.
- **Upgrading SQL Profiler Installation on Windows** - Provides information about upgrading your SQL Profiler from one major version to another on the Windows platform.

---

**Note:**

- Prior to PEM 7.8 release, SQL Profiler could be installed on Linux either by using the graphical installer or by using the RPMs. - From PEM version 7.8 onwards, SQL Profiler graphical installers for Linux are discontinued. To upgrade a SQL Profiler on a Linux host from any lower version to PEM 7.9 or higher versions, you must use native packages.
- 

Links to SQL Profiler installers and RPMs are available at the [EDB website](#) .

### 3.1 Upgrading a SQL profiler native package installation on a Linux host

To upgrade a SQL Profiler installation that resides on a Linux host:

1. Delete the existing SQL Profiler query set on each node by invoking the `uninstall-sql-profiler.sql` script. By default, on a Linux host the script resides in the `share/contrib` directory under your Advanced Server or PostgreSQL installation.

You can use the following server-specific command:

For PostgreSQL:

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

Where, *x* is the version of PostgreSQL and *-d* specifies the name of the maintenance database.

For Advanced Server:

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

Where,  $x$  is the version of Advanced Server and  $-d$  specifies the name of the maintenance database.

2. Invoke the new SQL Profiler installer on each node you wish to profile.

For PostgreSQL:

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

Where,  $x$  is the version of the PostgreSQL.

For Advanced Server:

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

Where  $x$  is the version of Advanced Server.

The installer will detect the existing SQL Profiler installation and upgrade with the latest version of SQL Profiler.

Please see the following example of upgrading SQL Profiler for PostgreSQL:

3. Then, run the `sql-profiler.sql` script file in the maintenance database.

For PostgreSQL:

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

Where,  $x$  is the version of PostgreSQL and  $-d$  specifies the name of the maintenance database.

For Advanced Server:

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

```
[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!
```

Fig. 1: Upgrading SQL Profiler

Where,  $x$  is the version of Advanced Server and  $-d$  specifies the name of the maintenance database.

- Restart the PostgreSQL/Advanced Server to resume profiling the node from the PEM Client.

## 3.2 Upgrading a SQL Profiler installation on a Windows host

If you are using SQL Profiler on a Windows host, Windows will lock any files that have been executed or loaded into memory. To release any locked files, you must stop the Postgres server before performing an upgrade.

On Windows, you can use the Services dialog to control the service. To open the Services dialog, navigate through the Control Panel to the System and Security menu. Select Administrative Tools, and then double-click the Services icon. When the Services dialog opens, highlight the service name in the list, and use the option provided on the dialog to Stop the service.

After stopping the Postgres Server:

- Delete the existing SQL Profiler query set on each node by invoking the `uninstall-sql-profiler.sql` script. By default, on a Windows host the script resides in the `share\contrib` directory under your Advanced Server or PostgreSQL installation.

You can use the following server-specific commands.

For PostgreSQL:

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

Where, *x* is the version of PostgreSQL and *-d* specifies the name of the maintenance database.

For Advanced Server:

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

Where *x* is the version of Advanced Server and *-d* specifies the name of the maintenance database.

2. Invoke the new SQL Profiler installer on each node you wish to profile; run the installer as **Administrator**.

For PostgreSQL:

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

Where, *x* is the version of the PostgreSQL and *y* is the version of SQL Profiler. For example: *sqlprofiler-pg-12-7.14.0-1-windows-x64.exe*.

For Advanced Server:

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

Where *x* is the version of Advanced Server and *y* is the version of SQL Profiler. For example: *sqlprofiler-edb-as12-7.14.0-1-windows-x64.exe*.

The SQL Profiler installer will detect the existing *SQL Profiler* installation and upgrade it with the latest version of SQL Profiler.

3. Run the `sql-profiler.sql` script file in the maintenance database.

For PostgreSQL:

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

Where *x* is the version of PostgreSQL and *-d* specifies the name of the maintenance database.

For Advanced Server:

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

Where,  $x$  is the version of Advanced Server and  $-d$  specifies the name of the maintenance database.

4. Then, restart the Postgres Server to resume profiling the node from a PEM client.



## MOVING THE PEM SERVER

The steps in this section describe how to move a PEM server from one host machine to a new host machine. The PEM server on the new host (the target) must be installed with the same version of the PEM server installer as the original host (the source). Please note that if you do not use the same installer version, you may encounter a schema-mismatch error.

The backend database of the target server (either PostgreSQL or Advanced Server) may be of the same type and version, or a different type and version than the backend database of the source PEM server. A PEM server that resides on a PostgreSQL host can be migrated to an Advanced Server host, or vice versa.

Before starting the server migration, you should ensure that the firewalls between the source host, the target host, and the host of any PEM Agent will allow connections between the services.

### 1. Prepare the Target Host

Invoke the installer for the PEM server on the target host. Please note that you must use the same version of the PEM server installer that you used when installing the source PEM server.

The backend database of the target server may be a different version or type than the backend database of the source. If the new PEM server does *not* reside on the same type of backend database as the original server, you must ensure that the same version of the `sslutils` extension is installed on the new server host. The version of `sslutils` that is distributed with the PEM installers is freely available for download from the EDB website at:

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

For information about installing the PEM server or the `sslutils` extension, please refer to the PEM Installation Guides, available at:

<https://www.enterprisedb.com/docs/p/edb-postgres-enterprise-manager>

### 2. Drop Existing Schemas from the New PEM Server

The migration process re-creates the `pem`, `pemdata`, and `pemhistory` schemas from the source PEM server on the target PEM server. In preparation for the move, use the `psql` client to delete these schemas from the `pem` database on the target host. You can open the `psql` client at the command line, or by selecting `SQL Shell (psql)` from the `Postgres Enterprise Manager` menu.

When the `psql` client opens, connect to the `pem` backend database as the database superuser. After connecting to the `pem` database on the target host, use the following commands to drop the schemas:

```
DROP SCHEMA pem CASCADE;  
  
DROP SCHEMA pemdata CASCADE;  
  
DROP SCHEMA pemhistory CASCADE;
```

When dropping the schemas, you must include the *CASCADE* keyword, instructing the server to delete all dependent objects. When executing the command, the *psql* client displays a list of the dependent objects; the client confirms each the schema is removed by displaying *DROP SCHEMA* .

### 3. Prepare the PEM Agents on the New PEM Server

Before moving the PEM server, you must identify the number of Agents that are monitored by the source PEM server, and create identities for that number of Agents (less one) on the target server. To discover the total number of PEM Agents monitored by the PEM server, connect to the pem database on the source host with the *psql* client, and query the *pem.agent* table.

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

You must manually create the number of Agents that reside on the original PEM server, less one; the PEM server installer has already created one Agent on the target host. For example, if the source server contains three Agents, you must manually create two additional Agents. Open a *psql* session with the *pem* database on the target server, and create the required Agents. Use the command:

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

Where <X> specifies an Agent number. Remember, *agent1* is created on the target host by the PEM Server installer.

Then, use the *GRANT* command to assign each Agent that resides on the target PEM Server *pem\_agent* permissions:

```
GRANT pem_agent TO agent<X>;
```

Where <X> specifies an agent number.

### 4. Generate a Backup Script of the Source PEM Server

You can use the *pg\_dump* utility to generate a script that contains the commands required to recreate the *pem* database on the target host. By default, *pg\_dump* is installed in the *bin* directory under your Postgres installation. To invoke *pg\_dump* , open a command line, navigate to the *bin* directory, and enter:

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

Where:

- *<user\_name>* specifies the name of the database superuser for the PEM backend database. - *<db\_name>* specifies the name of the PEM backend database. - *<file\_name>* specifies the name of the script generated by *pg\_dump*.

When prompted, provide the password associated with the user specified.

The command shown instructs *pg\_dump* to generate a script that (when executed) will re-create the *pem* database. The script will be named *backup.sql*, and will be created in the *tmp* directory. *pg\_dump* is connecting to the server using the credentials of the user, *postgres*.

Note that invoking the *pg\_dump* utility will not interrupt current database users.

---

**Note:** If the source PEM Server is lower than the 7.16 version, then you need to replace the following functions before you run *pg\_dump* to take backup:

---

- The *abstime*, *reltime*, and *interval* datatypes are deprecated from Postgres version 12 or later, hence to replace those datatypes with *timestampz* data type use below command:

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

- Replace the below function to avoid any alert errors:

```
CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
template_id pem.alert_template.id%type, params text[]
```

(continues on next page)

(continued from previous page)

```

)
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. Move the Backup to the Target Host

Move the script generated by the `pg_dump` utility to the target host of the PEM server.

#### 6. Restore the Backup on the Target Host

Open a command line on the target host and navigate into the `bin` directory (under the Postgres backend database installation directory). Start `psql`, executing the script generated by the `pg_dump` utility:

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

Where:

- `<user_name>` specifies the name of the database superuser. The user specified must have connection privileges for the backend database. - `<file_name>` specifies the complete path to the backup script generated by `pg_dump`.

When prompted, provide the password associated with the database superuser.

The example shown uses the `psql` client to invoke a script named `backup.sql` to recreate the `pem` database. The script is invoked using the privileges associated with the database superuser, `postgres`.

#### 7. Stop the Database Server on the Target Host

To stop the PEM Server on CentOS or RHEL 7.x or 8.x, use the command:

```
systemctl stop <service_name>
```

Where, `<service_name>` specifies the name of the backend database server. For a PostgreSQL backend database, the service name is `postgresql-<x>`, and for an Advanced Server backend database, the service name is `edb-as-<X>`, where `<X>` specifies the version number.

If you are using Windows, you can use the *Services* dialog to control the service. To open the *Services* dialog, navigate through the *Control Panel* to the *System and Security* menu. Select *Administrative Tools*, and then double-click the *Services* icon. When the *Services* dialog opens, highlight the service name in the list, and use the option provided on the dialog to Stop the service.

#### 8. Copy the Certificate Files to the Target Host

You must replace the certificate files that are created when the target host is installed with the certificate files of the source host. Copy the following files from the source PEM server to the target PEM server:

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

Copy the files to the data directory under the Postgres installation that provides the backend database for the target cluster.

On Linux, the files reside in:

```
/var/lib/pgsql/<X>/data/
```

On Windows, the files reside in:

```
C:\Program Files\PostgreSQL\<X>\data
```

Where:

`<X>` specifies the version of PostgreSQL on your system.

The files will already exist on the target cluster; delete the existing files before performing the copy, or overwrite the existing files with the files from the source server. Once in place on the target server, the files should have the (platform-specific) permissions described in the sections that follow.

#### **Permissions and Ownership on Linux**

| <b>File Name</b>   | <b>Owner</b> | <b>Permissions</b> |
|--------------------|--------------|--------------------|
| ca_certificate.crt | postgres     | -rw-----           |
| ca_key.key         | postgres     | -rw-----           |
| root.crt           | postgres     | -rw-----           |
| root.crl           | postgres     | -rw-----           |
| server.key         | postgres     | -rw-----           |
| server.crt         | postgres     | -rw-r--r--         |

On Linux, the certificate files must be owned by postgres. You can use the following command at the command line to modify the ownership of the files:

```
chown postgres <file_name>
```

Where *file\_name* specifies the name of the certificate file.

The server.crt file may only be modified by the owner of the file, but may be read by any user. You can use the following command to set the file permissions for the server.crt file:

```
chmod 644 server.crt
```

The other certificate files may only be modified or read by the owner of the file. You can use the following command to set the file permissions:

```
chmod 600 <file_name>
```

Where *file\_name* specifies the name of the file.

#### **Permissions and Ownership on Windows**

On Windows, the certificate files moved from the source host must be owned by the service account that performed the PEM server and backend database installation on the target host. If you invoked the PEM server and Postgres installer using the *Run as Administrator* option (selected from the context menu of the installer), the owner of the certificate files will be *Administrators*.

To review and modify file permissions on Windows, right-click on the file name, and select *Properties*.

! [The Permissions tab] (./images/move\_server\_windows\_permission.png) \*\* The Permissions tab \*\*

Navigate to the *Security* tab and highlight a *Group or user name* to view the assigned permissions. Select *Edit* or *Advanced* to access dialogs that allow you to modify the permissions associated with the selected user.

#### 9. Move the PEM Agent Certificate Files to the PEM Server Host

You must move the certificate files used by the PEM Agent of the source PEM server to the target host. This step is platform-specific.

##### OnLinux

Copy the `agent1.key` and `agent1.crt` files from the source host to the target host. By default, on Linux, the files are installed in `/root/.pem`; copy the files to the same directory on the target host.

File ownership and permissions of the files must be set to:

| File Name               | Owner             | Permissions             |
|-------------------------|-------------------|-------------------------|
| <code>agent1.key</code> | <code>root</code> | <code>-rw-----</code>   |
| <code>agent1.crt</code> | <code>root</code> | <code>-rw-r--r--</code> |

If necessary, navigate to `/root/.pem`, and use the following commands to modify the permissions and ownership of the `agent1.key` file:

```
chmod 600 agent1.key
chown root agent1.key
```

Use the following commands to modify the permissions and ownership of the `agent1.crt` file:

```
chmod 644 agent1.crt
chown root agent1.crt
```

##### OnWindows

Copy the `agent1.key` and `agent1.crt` files from the source host to the target host. On Windows, the files are located in:

```
C:\Users\<user_name>\AppData\Roaming\pem
```

Where *user\_name* is the name of the user that invoked the PEM installer.

The ownership and permissions associated with the certificate files on the target machine should match the ownership and permissions of the certificate files on the source machine. If you invoked the PEM server and Postgres installer using the *Run as Administrator* option (selected from the context menu of the installer), the owner of the Agent certificate files will be *Administrators*.

To review and modify file permissions on Windows, right-click on the file name, and select *Properties*. Navigate to the *Security* tab and highlight a *Group or user name* to view the assigned permissions. Select *Edit* or *Advanced* to access dialogs that allow you to modify the permissions associated with the selected user.

### 10. Update the pg\_hba.conf Files on the Target Host

Modify the `pg_hba.conf` file on the target host to allow connections from each PEM Agent. By default, the `pg_hba.conf` file is located in the data directory under your Postgres installation.

### 11. Start the Server on the Target Host

After modifying the `pg_hba.conf` file, you must restart the server for the changes to take effect.

To restart the database server on Linux, use the command:

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

Where *service\_name* is the name of the backend database server.

If you are using Windows, you can use the *Services* dialog to control the service. To open the *Services* dialog, navigate through the *Control Panel* to the *System and Security* menu. Select *Administrative Tools*, and then double-click the *Services* icon. When the *Services* dialog opens, highlight the service name in the list, and use the option provided on the dialog to Start the service.

### 12. Connecting Monitored Agents to the New PEM Server Host

To instruct existing PEM Agents to connect to the new PEM server host, you must:

- Ensure that the PEM Agent host can connect to the new PEM server host. - Modify the registry (on each Windows host with a PEM Agent) or the Agent configuration files (on each Linux host with a PEM Agent), specifying the IP address and port of the new PEM server. - Restart the PEM Agent's service. These steps are platform-specific: - [Prerequisites to upgrade a PEM installation on Linux host](#) - [Prerequisites to upgrade a PEM installation on Linux host](#)

#### **If the PEM Agent Resides on Linux**

Use your choice of editor to modify the `agent.cfg` file, specifying the new IP address and port number of the PEM server in the `pem_host` and `pem_port` parameters.

By default, the `agent.cfg` file is located in:

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

After modifying the `agent.cfg` file, you must restart the PEM Agent service; you can use the `pemagent` service script on the Linux command line to restart the service:

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

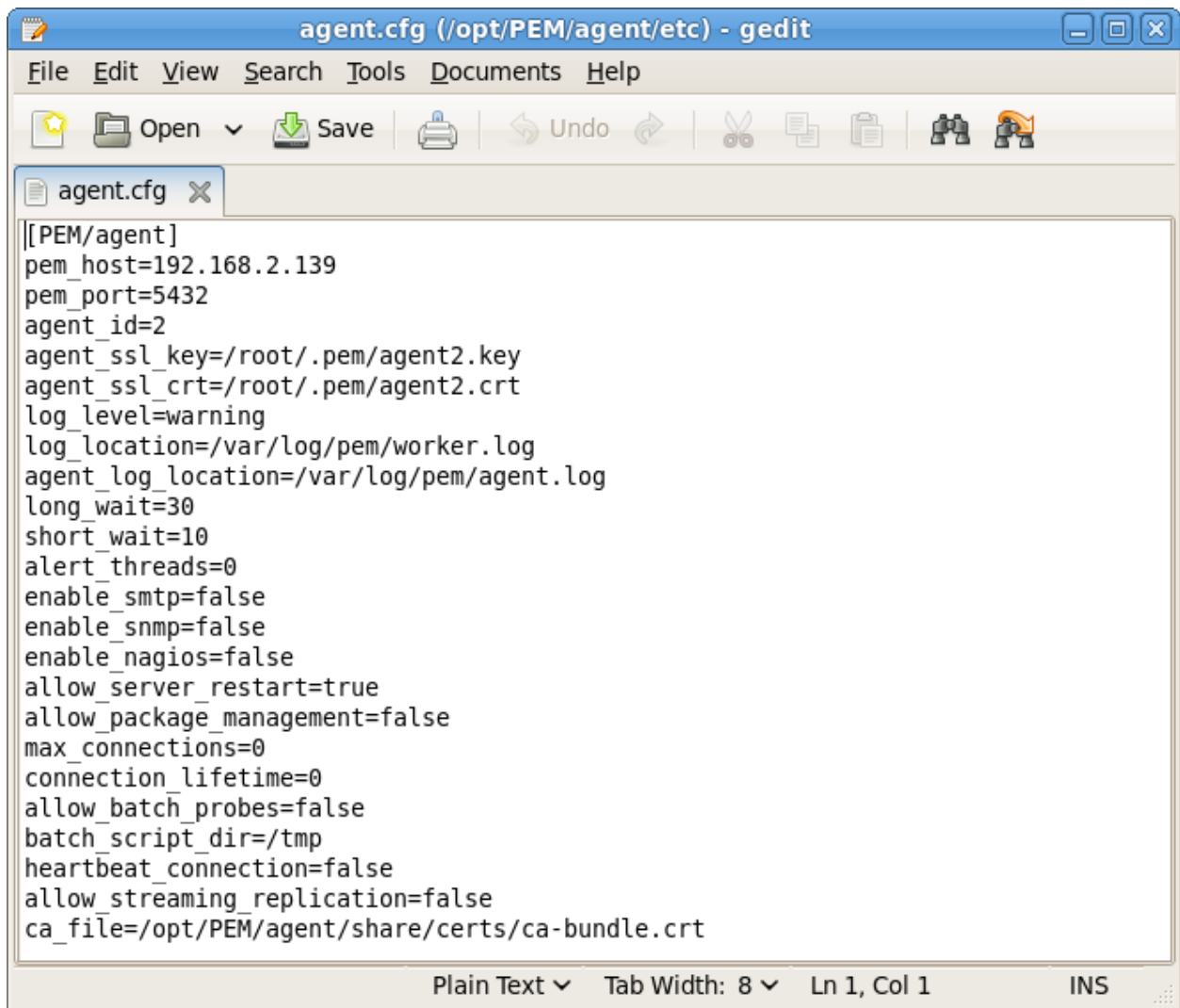


Fig. 1: The agent.cfg file

### If the PEM Agent Resides on Windows

Before modifying the Windows registry on the monitored node, confirm that the firewall on the host of the PEM Agent will allow connections to the PEM server. After confirming that the PEM Agent host can connect to the PEM server host, you can use the Windows *Registry Editor* to review and edit the *PEM\_HOST* and *PEM\_PORT* entries to ensure that they correctly identify the host and port used by the PEM server. To open the *Registry Editor*, enter *regedit* in the Windows *Run* dialog or in the Windows start menu search box. Navigate through the registry tree control to view or modify registry entries.

On 64-bit Windows, the PEM Agent registry entries are located:

*HKEY\_LOCAL\_MACHINE SOFTWARE wow6432Mode EnterpriseDB PEM Agent*

On 32-bit Windows, the PEM Agent registry entries are located:

*HKEY\_LOCAL\_MACHINE SOFTWARE EnterpriseDB PEM Agent*

![The Windows Registry Editor](../images/move\_pem\_server\_windows\_registry.png)

The *PEM\_HOST* and *PEM\_PORT* entries must specify the address and port number of the new PEM server on the target host. To modify a registry entry, right click on the entry *Name*, and select *Modify* from the context menu to open the *Edit String* dialog.

![The Windows Registry Editor](../images/move\_pem\_server\_windows\_registry\_edit.png)

Use the *Edit String* dialog to make any changes to the value of the entry. When you're finished, click *OK* to save your changes, or *Cancel* to exit without saving.

After modifying the registry, you must restart the PEM Agent's service; you can use the *Services* dialog (accessed through the Windows *Control Panel*) to restart the *Postgres Enterprise Manager - pemagent* service.

![Restarting the PEM Agent service](../images/move\_pem\_server\_restart\_agent.png) \*\* Restarting the PEM Agent service \*\*

After moving the server, change the connection properties in any installed PEM clients to connect to the new host of the PEM server, Agents, and monitored servers.