



Postgres Enterprise Manager

リリース **8.0**

PEM Security Guide

2020 年 12 月 19 日

目次

第 1 章	Apache HTTPD Security Configurations	3
第 2 章	PEM application Security Configurations	11
第 3 章	Conclusion	17
索引		19

このドキュメントでは、PEM を構成する際に考慮する必要があるセキュリティ対策に関する情報を提供します。PEM 機能では、これらのプラクティスを実施する必要はありません。ただし、EDB は、システム全体のセキュリティを強化するためにこれらのプラクティスを推奨しています。

PEM は、ApacheWeb サーバー、OpenSSL、snmp++、libcurl などを含むベンダーリポジトリのサードパーティコンポーネントに依存しています。これらのコンポーネントが最新であることを確認するには、オペレーティングシステムを更新し、定期的にセキュリティ更新プログラムを適用して、セキュリティの脆弱性。最新のセキュリティパッチがなければ、システムは潜在的にサイバー攻撃に対して脆弱です。セキュリティパッチは、最新の更新プログラムを適用して最新の脅威から保護することにより、デバイスとそのデータを保護します。

セキュリティパッチを定期的に適用することには、次のような利点があります。

- サイバー攻撃への露出を削減
- 生産性の低下を回避する
- マルウェア（ランサムウェアなど）からのデータ保護
- セキュリティの抜け穴を使用してネットワーク上に広がるワーム感染を回避する

第 1 章

Apache HTTPD Security Configurations

Windows システムでは、ApacheHTTPD は PEMHTTPD という名前で、Apachehttpd 構成ファイル (pme.conf) は <Apache_Installation_Path>/conf/addons ディレクトリにあります。ssl 構成ファイル (httpd-ssl-pem.conf) は <Apache_Installation_Path>/conf/addons ディレクトリにあります。

Linux システムでは、apachehttpd 構成ファイル (edb-pem.conf) は <Apache_Installation_Path>/conf.d ディレクトリにあり、SSL 構成ファイル (edb-ssl-pem.conf) は <Apache_Installation_Path>/conf.d ディレクトリにあります。

1.1 SSLv2 および SSLv3 を無効にする

SSL バージョン SSLv2、SSLv3、TLS1、および TLS1.1 を無効にする必要があります。これらのバージョンは最も脆弱であり、暗号化の問題の影響を受けるためです。

バージョンを無効にするには、次のコマンドを Apachehttpd 設定ファイルに追加してください。

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

設定ファイルに変更を適用するには、Web サーバーを再起動する必要があります。

デフォルトでは、PEM は SSL 構成ファイルに次の行を追加して、セキュリティのために TLS1.2 を使用できるようにします。

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

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

1.2 SSL 証明書で安全な httpd

EDB は、Web アプリケーション用に SSL セキュリティの追加レイヤーを持つことを推奨しています。

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

Linux システムでは、ファイルの書き込み権限を持つユーザーとして、テキストエディターで `Apachehttpd` 構成ファイル (`edb-ssl-pem.conf`) を開く必要があります。また、構成ファイル内のサーバー名とファイル名を変更して、証明書ファイルと一致させる必要があります。

PEMVirtualHost セクション内には、更新する必要がある 2 つの SSL ディレクティブがあります。

- `SSLCertificateFile` は DigiCert 証明書ファイルです (例: `your_domain_name.crt`)。
- `SSLCertificateKeyFile` は、CSR の作成時に生成された `key` ファイルです (例: `your_private.key`)。

たとえば、更新:

```
SSLEngine on

SSLCertificateFile /path/to/your_domain_name.crt

SSLCertificateKeyFile /path/to/your_private.key
```

PostgresEnterpriseManager で、`httpd` 自己署名 SSL 証明書を信頼できる CA 署名付き証明書に置き換えることもできます。手順については、以下に示すリンクに従ってください。

<https://www.enterprisedb.com/postgres-tutorials/how-replacing-httpd-self-signed-ssl-certificates-trusted-ca-signed-certificates>

1.3 Web サーバーの情報漏えいを無効にする

EDB は、Web サーバーのセキュリティの一部として、すべての Web サーバーの署名を無効にすることをお勧めします。Web サーバーはソフトウェア署名を公開します。署名を無効にするには、次のパラメーターを `Apachehttpd` 構成ファイルに追加します。デフォルトでは、PEM は以下のパラメーターを `Apachehttpd` 構成ファイルに追加することにより、情報の公開を無効にします。

```
ServerTokens Prod

ServerSignature Off
```

`ServerTokens` ディレクティブは、クライアントに返されるサーバー応答ヘッダーフィールドを制御します。`Apachehttpd` 構成ファイルにこのパラメーターを追加して、Apache サーバーのバージョンを非表示にすることをお勧めします。

`ServerSignature` ディレクティブには、サーバーで作成されたドキュメントのフッターが含まれます。フッターには、Apache やオペレーティングシステムのバージョンなど、Apache の構成に関する情報が含まれています。そのような情報の表示を制限するには、`Apachehttpd` 構成ファイルでこのディレクティブを無効にすることをお勧めします。

`Apachehttpd` 構成ファイルに変更を適用するには、Web サーバーを再起動する必要があります。

1.4 ディレクトリリストを無効にする

ディレクトリのリストにより、攻撃者は完全なディレクトリの内容を表示できます。デフォルトでは、Web サーバーはこのオプションを有効にしており、攻撃者は任意のファイルを発見および表示できます。このリストは、攻撃者がアプリケーションをリバースエンジニアリングしてソースコードを取得し、考えられるセキュリティ上の欠陥について分析し、アプリケーションに関する詳細情報を発見する可能性があります。

これを回避するには、`Apachehttpd` 構成ファイルで `Options` ディレクティブを設定して、ディレクトリリストを無効にする必要があります。デフォルトでは、`PEM` は Web サーバー設定ファイルで以下のオプションを設定することにより、ディレクトリリストを無効にします。

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

構成ファイルに加えた変更を適用するには、Web サーバーを再起動する必要があります。

1.5 ネットワークまたは IP アドレスへのアクセスを制限する

Apache は、クライアントのホスト名または IP アドレスに基づいてアクセス制御を提供します。特定の IP アドレスまたはネットワークでアプリケーションを表示するには、ユーザーは以下に示すように Apache 構成ファイルを変更して、`Allow` ディレクティブ内でネットワークアドレスを提供できます。

```
<Directory /application/hostname>

Options None

AllowOverride None

Order deny,allow

Deny from all

Allow from 192.168.0.0/24

</Directory>
```

`PEM` は、アプリケーション構成ファイルの `ALLOWED_HOSTS` 構成パラメーターを使用して、許可されたホストの IP アドレスのリストを提供します。アプリケーション構成の `config_local.py` ファイルは `<PEM_INSTALLATION_PATH>/web` にあります。

デフォルトでは、`PEM` はすべてのホストがアプリケーションに接続できるようにします。

例:

以下のように、構成ファイルで IP アドレスの範囲を設定できます。

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

以下のように、構成ファイルでサブネットレベルのホストを許可するように IP アドレスを設定できます。

```
ALLOWED_HOSTS = ['192.0.2.0/28', ':::192.0.2.0/124']
```

以下のように、構成ファイルで特定の個々のホストアドレス（IP アドレスに基づいて）を設定できます。

```
ALLOWED_HOSTS = ['127.0.0.1', '192.168.0.1']
```

アプリケーション構成ファイルに変更を適用するには、Web サーバーを再起動する必要があります。

1.6 クロスサイトトレース

Web サーバー接続をデバッグするには、TRACE と TRACK の 2 つの HTTP メソッドがあります。HTTPTRACE 要求がそれをサポートする Web サーバーに送信されると、そのサーバーは応答し、HTTP ヘッダーを含む、渡されたデータをエコーします。Apache 構成内でこれらのメソッドを無効にすることをお勧めします。

すべての仮想ホストで TRACE メソッドを無効にするには、次の行を `Apachehttpd` 構成ファイルに追加します。

```
TraceEnable off
```

特定の仮想ホストでこれらの方法を無効にするには、Apache 構成ファイルで各仮想ホストに次の行を追加します。PEM は、`Apachehttpd` 構成ファイルに次の行を追加します。

```
RewriteEngine on
```

```
RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK|OPTIONS)
```

```
RewriteRule .* - [F]
```

1.7 非特権ユーザーアカウントから Web サーバーを実行する

root ユーザーとして ApacheWeb サーバーを実行すると、セキュリティの問題が発生します。Web サーバーを一意的な非特権ユーザーとして実行することを常にお勧めします。これにより、セキュリティ違反が発生した場合に実行されている他のサービスを保護できます。

PEM は WSGI アプリケーションとして実行されます。実行中の WSGI アプリケーションを委任するには、`WSGIDaemonProcess` ディレクティブを使用して個別のデーモンプロセスを作成できます。

Linux では、ApacheWeb サーバーはルートユーザーとして起動しますが、デーモンはどの PEM アプリケーションが pem ユーザーとして実行するかを処理します。Windows では、`WSGIDaemonProcess` ディレクティブとその機能は使用できません。PEMHTTPD はインストール中にサービスとしてインストールされ、ユーザーはサービスのインストールを成功させるために Administrators グループのメンバーである必要があります。

デフォルトでは、Apache サービスはシステムユーザー（`LocalSystem` アカウント）として実行するように登録されています。

1.8 PEMWebServer でセキュリティ HTTP ヘッダーをカスタマイズする

PEM には、次のセキュリティ問題を修正するための独自の構成ファイルが含まれています。 `config.py` ではなく `config_local.py` のみの構成をオーバーライドすることをお勧めします。ほとんどの場合、`config_local.py` はシステムに存在しません。したがって、ユーザーはアプリケーションレベルの構成を上書きするために作成する必要があります。PEM のアップグレード中、`config_local.py` は上書きされませんが、`config.py` と `config_distro.py` の変更は上書きされます。ユーザーは、PEM をアンインストールした後に `config_local.py` を削除する必要があります。

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

1.9 ホストヘッダーインジェクション攻撃

HTTP ホストヘッダー攻撃は、ホストヘッダー値を安全でない方法で処理する脆弱な Web サイトを悪用します。サーバーが暗黙的にホストヘッダーを信頼し、適切に検証またはエスケープできない場合、攻撃者はこの入力を使用して、サーバー側の動作を操作する有害なペイロードを挿入できる可能性があります。通常、Web アプリケーションは、セットアップ中に構成ファイルで指定されない限り、デプロイされているドメインを認識しません。たとえば、現在のドメインを知る必要がある場合、ホストヘッダーからドメインを取得して絶対 URL を生成することに頼ることがあります。ホストヘッダーは、他のさまざまな脆弱性、特に Web キャッシュポイズニングと SQL インジェクションを悪用する可能性のあるベクトルです。

1.9.1 X-Frame-Options

X-Frame-Options を使用して、ブラウザーが<iframe>タグでページをレンダリングできるようにするかどうかに示すことができます。クリックジャックから保護するために特別に設計されました。PEM には、このような種類の攻撃を防ぐためのホスト検証 X_FRAME_OPTIONS オプションがあり、`config_local.py` ファイルで設定できます。デフォルトは次のとおりです。

```
X_FRAME_OPTIONS = /"SAMEORIGIN/"
```

1.9.2 コンテンツセキュリティポリシー

Content-Security-Policy は HTML5 標準の一部であり、X-Frame-Options ヘッダー（置き換えられる）よりも幅広い保護を提供します。Web サイトの作成者は、リソース（スクリプト、スタイルシート、フォントなど）を読み込むことができる個々のドメイン、およびページの埋め込みを許可するドメインをホワイトリストに登録できるように設計されています。

PEM には、このような種類の攻撃を防ぐためのホスト検証 CONTENT_SECURITY_POLICY オプションがあり、`config_local.py` ファイルで設定できます。デフォルトは次のとおりです。

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

1.9.3 厳格な輸送セキュリティ

Strict-Transport-Security 応答ヘッダー（多くの場合 HSTS と略されます）を使用すると、Web サイトまたは Web アプリケーションは、HTTP ではなく HTTPS を使用してのみアクセスする必要があることをブラウザーに通知できます。このオプションを使用すると、中間者攻撃を防ぐことができます。デフォルトは次のとおりです。

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

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

1.9.4 X-Content-Type-Options

X-Content-Type-Options 応答 HTTP ヘッダーは、Content-Type ヘッダーでアダプタイズされた MIME タイプを変更して従わないことを示すためにサーバーが使用するマーカーです。これは、MIME タイプのスニффイングをオプトアウトする方法、つまり、MIME タイプが意図的に設定されていると言う方法です。デフォルトは次のとおりです。

```
X_CONTENT_TYPE_OPTIONS = /"nosniff/"
```

1.9.5 X-XSS-Protection

クロスサイトスクリプティング（XSS）は、Web サーバーで最も一般的なアプリケーション層の脆弱性の 1 つです。XSS を使用すると、攻撃者はクライアント側のスクリプトを他のユーザーが表示する Web ページに挿入できます。ヘッダーに対する HTTPX-XSS-Protection 応答は、Internet Explorer、Chrome、および Safari の機能であり、反射されたクロスサイトスクリプティング（XSS）攻撃を検出するとページの読み込みを停止します。インライン JavaScript（「unsafe-inline」）の使用を無効にする強力な Content-Security-Policy をサイトが実装している場合、これらの保護は現代のブラウザーではほとんど不要ですが、CSP をまだサポートしていない古い Web ブラウザーのユーザーに保護を提供できます。デフォルトは次のとおりです。

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

これを回避するには、Apache 構成ファイルに次のオプションを追加する必要があります。

```
<IfModule mod_headers.c>
```

```
Header set X-XSS-Protection /"1; mode=block/"
```

```
</IfModule>
```

構成ファイルの変更を適用するには、Web サーバーの再起動が必要です。

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

この変更を有効にするには、Apache サービスの再起動が必要です

`config.py` ファイルの詳細については、「PEM オンラインヘルプ<https://www.enterprisedb.com/edb-docs/d/edb-postgres-enterprise-manager/reference/online-documentation/latest/pem_managing_configuration_settings>」を参照してください。<code>.html>'</code>_</code>。

1.9.6 クッキーのセキュリティ

Cookie は、サーバーがブラウザに送信して構成データを保存できるデータの小さなパケットです。ブラウザはすべてのリクエストとともにそれらを同じサーバーに自動的に送信するため、Cookie を保護する方法を知ることが重要です。<code>config.py</code> で参照できる Cookie を安全にするために PEM によって提供される複数の構成オプションがありますが、以下の 3 つが最も重要です。

- (a) **SESSION_COOKIE_SECURE** -セキュアフラグを設定すると、Cookie が暗号化されていない接続で送信されることを防ぎます。基本的に、暗号化されたチャネルを使用しないサーバーへのリクエストに Cookie を追加しないようにブラウザに指示します。ブラウザは、HTTPS などの接続にのみ Cookie を追加します。デフォルトは次のとおりです。

```
SESSION_COOKIE_SECURE = True
```

- (b) **SESSION_COOKIE_HTTPONLY** -デフォルトでは、Cookie のコンテンツは JavaScript を介して読み取ることができます。HTTPOnly フラグは、スクリプトが Cookie を読み取れないようにします。HTTPOnly という名前が示すように、ブラウザは HTTP (S) リクエストでのみ Cookie を使用します。これにより、ハッカーが XSS 脆弱性を使用して Cookie のコンテンツを学習することを防ぎます。たとえば、<code>sessionId</code> Cookie の場合、クライアント側のスクリプトで Cookie を読み取る必要はないため、<code>sessionId</code> Cookie の場合は常に HTTPOnly フラグを設定できます。デフォルトは次のとおりです。

```
SESSION_COOKIE_HTTPONLY = True
```

- (c) **ENHANCED_COOKIE_PROTECTION** - このオプションを True に設定すると、IP アドレスとユーザーエージェントに従ってトークンが生成されます。後続のすべてのリクエストでは、トークンが再計算され、最初のリクエストに対して計算されたトークンと照合されます。セッション Cookie が盗まれ、攻撃者が別の場所からそれを使用しようとする、生成されるトークンは異なります。その場合、拡張機能はセッションをクリアし、リクエストをブロックします。デフォルトは次のとおりです。

```
ENHANCED_COOKIE_PROTECTION = True
```

注釈: このオプションは、サーバーが Kubernetes やロードバランサーの背後などの動的 IP アドレスホスティング環境に展開されている場合に問題を引き起こす可能性があります。そのような場合、このオプションは `False` に設定する必要があります。

この変更を有効にするには、Apache サービスの再起動が必要です。

`config.py` ファイルの詳細については、「PEM オンラインヘルプ<https://www.enterprisedb.com/edb-docs/d/edb-postgres-enterprise-manager/reference/online-documentation/latest/pem_managing_configuration_settings>」を参照してください。</p></div>
<div data-bbox="110 935 138 952" data-label="Page-Footer">10</div>
<div data-bbox="517 935 891 953" data-label="Page-Footer">第 1 章 Apache HTTPD Security Configurations</div>

第 2 章

PEM application Security Configurations

2.1 セッションタイムアウト

Web アプリケーションによるセッションの有効期限が不十分だと、攻撃者が有効なセッション ID を再利用して関連するセッションをハイジャックする時間を確保できるため、他のセッションベースの攻撃の危険性が高まります。セッション間隔が短いほど、攻撃者が有効なセッション ID を使用する必要がある時間が短くなります。このセキュリティ問題を回避するために、Web アプリケーションの非アクティブタイムアウトを低い値に設定することをお勧めします。

PostgresEnterpriseManager は、ユーザーセッションのタイムアウト値を設定する方法を提供します。Web コンソールで指定された期間ユーザーのアクティビティがない場合、PEM は Web コンソールからユーザーをログアウトします。PEM 管理者は、非アクティブな時間の長さを設定できます。この値は、個々のユーザーではなく、アプリケーションごとです。タイムアウト時間を設定するには、<PEM_INSTALLATION_PATH>/web ディレクトリにある config_local.py ファイルの USER_INACTIVITY_TIMEOUT パラメーターを変更します。デフォルトでは、この機能は無効になっています。

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

```
USER_INACTIVITY_TIMEOUT = 900
```

注釈: タイムアウト値は秒単位で指定されます。

この変更を有効にするには、Apache サービスの再起動が必要です。

config.py ファイルの詳細については、「PEM オンラインヘルプ」<https://www.enterprisedb.com/edb-docs/d/edb-postgres-enterprise-manager/reference/online-documentation/latest/pem_managing_configuration_settings>を参照してください。<.html>‘_。

2.2 RestAPI ヘッダーのカスタマイズ

要件に応じて RestAPI トークンヘッダーをカスタマイズできます。デフォルト値は config.py ファイルによって公開されません。config_local.py ファイルの次のヘッダーをカスタマイズします。

2.2.1 PEM_HEADER_SUBJECT_TOKEN_KEY

この構成オプションを使用すると、HTTP ヘッダー名を変更して、生成されたトークンを取得できます。デフォルトでは、ユーザーがトークンを作成するリクエストを送信すると、サーバーレスポンスには「X-Subject-Token」ヘッダーが含まれ、これには新しく生成されたトークンの値が含まれます。ヘッダー名をカスタマイズする場合は、次のように config_local.py ファイルを更新できます。

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

次の出力が得られます。

```
curl -ik -X POST -d '{"username":"/enterprisedb/", "password":"/edb/"}' -H "Content-Type:application/json/" https://localhost:8443/pem/api/token/
```

```
HTTP/1.1 201 CREATED
```

```
Date:Thu, 29 Oct 2020 11:03:48 GMT
```

```
Server:Apache
```

```
Content-Length:326
```

```
Pem-RestAPI-Generate-Token:997aef95-d46d-4d84-932a-a80146eaf84f
```

2.2.2 PEM_HEADER_TOKEN_KEY

この構成オプションを使用すると、トークンを PEM サーバーに送信する際に使用する HTTP 要求ヘッダー名を変更できます。デフォルトでは、ユーザーがトークンを生成するリクエストを送信すると、トークンヘッダーは X-Auth-Token になります。RestAPI リクエストヘッダー名をカスタマイズする場合は、次のように config_local.py ファイルを更新できます。

```
PEM_HEADER_TOKEN_KEY = 'Pem-Token'
```

これにより、トークンを送信できます。

```
$ curl -Lk -X GET -H "Pem-Token:gw5rzaloxyp91tttd1c97w24b5sv60cllc24sxy9/" https://localhost:8443/pem/api/v4/agent
```

2.2.3 PEM_TOKEN_EXPIRY

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

```
PEM_TOKEN_EXPIRY = 600
```

この変更を有効にするには、Apache サービスの再起動が必要です。

2.3 PEM の役割ベースのアクセス制御

役割ベースのアクセス制御（RBAC）は、組織内でのユーザーの役割に基づいてアプリケーションアクセスを制限し、アクセス制御の主要な方法の 1 つです。RBAC のロールは、ユーザーがアプリケーションに対して持つアクセスのレベルを指します。ユーザーは、職務を効果的に遂行するために必要な情報にのみアクセスできます。PEM のロールは、添字ではなく継承可能で付加的です。簡単に言えば、PEM 管理者として、ユーザーに最低レベルの役割を付与し、ユーザーがそれぞれのタスクを実行するために必要な役割を付与する必要があります。たとえば、SQL プロファイラーのみにアクセスを許可するには:

```
CREATE ROLE user_sql_profiler WITH LOGIN NOSUPERUSER NOCREATEDB
NOCREATEROLE INHERIT NOREPLICATION CONNECTION LIMIT -1 PASSWORD 'xxxxxxx';

GRANT pem_user, pem_comp_sqlprofiler TO user_sql_profiler;
```

ロールの詳細については、「PEMRoles<https://www.enterprisedb.com/edb-docs/d/edb-postgres-enterprise-manager/reference/online-documentation/latest/pem_roles.html>」を参照してください‘_。

2.4 SQL/Protect プラグイン

SQL インジェクション攻撃の防止は、通常、アプリケーション開発者の責任です。データベース管理者は通常、潜在的な脅威をほとんどまたはまったく制御できません。データベース管理者にとっての困難は、アプリケーションが適切に機能するためにデータにアクセスできる必要があることです。

SQL/Protect は、データベース管理者が SQL インジェクション攻撃からデータベースを保護できるようにするモジュールです。SQL/Protect は、標準的なデータベースセキュリティポリシーに加えて、一般的な SQL インジェクションプロファイルの受信クエリを調べることにより、セキュリティのレイヤーを提供します。

SQL インジェクション攻撃を実行するために使用されるいくつかの異なる手法があります。特定の署名が各手法を特徴づけます。SQL/Protect は、不正な関係、ユーティリティコマンド、SQL トートロジー、無制限の DML ステートメントのクエリを調べます。SQL/Protect は、潜在的に危険なクエリを管理者に警告してブロックすることにより、データベース管理者に制御を戻します。

注釈: このプラグインは EPAS サーバーでのみ機能するため、PEM データベースが EPAS サーバーでホストされている場合にのみ有効です。

SQL プロファイラープラグインの詳細については、「PEM オンラインヘルプ-SQL プロファイラー<<https://www.enterprisedb.com/edb-docs/d/edb-postgres-enterprise-manager/reference/online-documentation/>>」を参照してください。latest/toc_pem_sql_profiler.html>’。

2.5 パスワード管理

PEM 管理ユーザーのセキュリティ上のヒントの 1 つは、PEM ログインパスワードを定期的に新しいものに変更することです。パスワードを変更すると、次のような多くの危険を回避できます。

- 複数のアカウントの侵害
- 絶え間ないアクセスを防ぐ
- 物理的に保護されていないシステムで保存されたパスワードを使用できないようにします
- キーストロークロガーによって得られるアクセスを制限する

2.6 非 root ユーザーで pemAgent ジョブを実行する

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

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

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

```
HKEY_LOCAL_MACHINE/Software/Wow6432Node/EnterpriseDB/PEM/agent
```

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

デフォルトでは、PEM をインストールすると、インストーラーは pemAgent および PEM データベースサーバーの自己署名証明書を生成して使用します。PemAgent は、PEM データベースサーバーに接続するときにこれらの証明書を使用します。pemAgent および PEM データベースサーバーに独自の SSL 証明書を使用するには、PEM Administrators's Guide の Managing Certificates セクションに記載されている手順に従います。

https://www.enterprisedb.com/edb-docs/d/edb-postgres-enterprise-manager/user-guides/administrators-guide/7.16/managing_certificates.html<ここで、managing_certificates.html ファイルへのリンクを提供する必要があります>。

注釈: PEM は、SSLCA 証明書をカスタムの場所に配置することをサポートしていません。ca_certificate.crt と ca_key.key の場所を変更しないでください。

第 3 章

Conclusion

PostgresEnterpriseManager セキュリティガイド

著作権 ©2020EnterpriseDBCorporation。無断複写・転載を禁じます。

EnterpriseDB®Corporation34CrosbyDrive、Suite201、Bedford、MA01730、USA

T+17813573390F+19784671307Einfo@enterprisedb.comwww.enterprisedb.com

- EDB および PostgresEnterpriseManager は、EnterpriseDBCorporation の登録商標です。EDB および EDB-Postgres は、EnterpriseDBCorporation の商標です。Oracle は、Oracle、Inc の登録商標です。その他の商標は、それぞれの所有者の商標です。
- EDB は、PostgresEnterpriseManager が存在する場合、ログオン UI の設計、コーディングのベストプラクティスの確立、レビュー、入力検証を検証します。EDB は追加の入力コンポーネントに対して同じアプローチに従いますが、製品の性質上、限定された検証が可能な信頼できるユーザーによって入力および送信されるフリーフォーム SQL、WMI またはその他の文字列を受け入れる必要がある場合があります。このような場合、ユーザーが誤った入力や危険な入力を禁止することはできません。
- EDB は、将来、認証された信頼できるユーザーからのフリーフォーム SQL、WMI またはその他の潜在的に危険な入力を受け入れる製品に機能を追加する権利を留保しますが、そのような機能はすべて、可能な限り最小のリスクを提供するように設計およびテストされます可能であれば、スーパーユーザーまたは同等の特権が必要です。
- EDB は、すべての潜在的な脅威を予測できることを保証しません。したがって、すべての潜在的な脆弱性が対処または検討されたことをプロセスが完全に保証することはできません。

索引

Apache HTTPD Security Configurations, 3
Conclusion, 17
PEM application Security Configurations, 11