



EDB

PEM Security Guide

リリース 8.3.0

EnterpriseDB

2021 年 12 月 26 日

目次

第 1 章	Apache HTTPD Security Configurations	3
1.1	SSLv2 および SSLv3 を無効にする	3
1.2	SSL 証明書で安全な httpd	3
1.3	Web サーバーの情報漏えいを無効にする	4
1.4	ディレクトリ一覧の無効化	4
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	5
1.6	クロスサイトトレース	5
1.7	特権のないユーザーアカウントから Web サーバーを実行する	6
1.8	PEM WebServer でセキュリティ HTTP ヘッダーをカスタマイズする	6
1.9	ホストヘッダーインジェクション攻撃	7
1.9.1	X-Frame-Options	7
1.9.2	コンテンツセキュリティポリシー	7
1.9.3	厳格な輸送セキュリティ	7
1.9.4	X-Content-Type-Options	8
1.9.5	X-XSS-Protection	8
1.9.6	Cookie セキュリティ	8
第 2 章	PEM application Security Configurations	11
2.1	セッションタイムアウト	11
2.2	RestAPI ヘッダーのカスタマイズ	11
2.2.1	PEM_HEADER_SUBJECT_TOKEN_KEY	12
2.2.2	PEM_HEADER_TOKEN_KEY	12
2.2.3	PEM_TOKEN_EXPIRY	12
2.3	PEM での役割ベースのアクセス制御	13
2.4	SQL/ Protect プラグイン	13
2.5	パスワード管理	13
2.6	非 root ユーザーで pemAgent ジョブを実行する	14
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	14

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 はこれらのプラクティスを推奨して、システム全体のセキュリティを強化しています。

PEM は、Apache ウェブサーバ、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、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い場合、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。

- サイバー攻撃の危険性を軽減
- 生産性の低下の回避
- 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、Apache HTTPD の名前付けは PEM HTTPD で、Apache httpd 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、apache httpd 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、TLS 1、および TLS 1.1 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 apache httpd 設定ファイルに追加してください：

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

構成ファイルに変更を適用するには、Web サーバーをリスタート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 TLS 1.2 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、ファイルに対する書き込みパーミッションを持つユーザとして、テキストエディタで Apache httpd 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前とファイル名を変更して、証明書ファイルとマッチさせる必要があります。

PEM VirtualHost 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 (たとえば、your_domain_name.crt) です。

- SSLCertificateKeyFile は、CSR の作成時に生成されたキーファイル (例: your_private.key)。

例、更新:

SSLEngine on

SSLCertificateFile /path/to/your_domain_name.crt

SSLCertificateKeyFile /path/to/your_private.key

Postgres Enterprise Manager 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、ウェブサーバのセキュリティのパートとして、すべてのウェブサーバの署名を無効にすることをお勧めします。ウェブサーバはソフトウェアシグネチャを公開します。シグネチャを無効にするには、Apache httpd 構成ファイルに次のパラメーターを追加します。デフォルトでは、PEM は以下のパラメーターを Apache httpd 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フィールドを制御します。Apache httpd 構成ファイルにこのパラメータを追加して、Apache サーバ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。このような情報の表示を制限するには、Apache httpd 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。

Apache httpd 構成ファイルに変更を適用するには、ウェブサーバをリスタートする必要があります。

1.4 ディレクトリー覧の無効化

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ビューできます。デフォルトでは、ウェブサーバはこのオプションを有効にし、攻撃者は任意のファイルを発見およびビューできます。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を分析して、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、Apache httpd 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 はウェブサーバ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、ウェブサーバをリスタート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別にアプリケーションをビューするには、ユーザは Apache 構成ファイルを次のように変更して、**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** 設定パラメータを使用して、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 アドレスのレンジを設定できます。

```
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、ウェブサーバをリスタートする必要があります。

1.6 クロスサイトトレース

ウェブサーバ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。HTTP **TRACE** 要求がそれをサポートするウェブサーバに送信されると、そのサーバは応答し、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 Apache **httpd** 構成ファイルに追加します。

```
TraceEnable off
```

特定の仮想ホストでこれらの方法を無効にするには、Apache 構成ファイルで各仮想ホストに次の行を追加します。PEM は、Apache httpd 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 特権のないユーザーアカウントから Web サーバーを実行する

ルートユーザとして Apache ウェブ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。ウェブサーバーを一意的な非特権ユーザ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イベント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別個のデーモンプロセスを作成できます。

Linux では、Apache ウェブサーバーはルートユーザ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。PEM HTTPD 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 グループのメンバであるニーズがあります。

デフォルトでは、Apache サービスはシステムユーザ (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 PEM WebServer 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 の構成のみ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ることにノートしてください。PEM をアンインストールした後、ユーザーは 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、WindowsC:\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ハンドルする脆弱な 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、サーバサイド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。例、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、他のレンジな脆弱性、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、ブラウザがページを <iframe> タグでレンダリングできるようにするかどうかを示すことができます。クリックジャッキング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バリデーション `X_FRAME_OPTIONS` オプションがあり、`config_local.py` ファイルで構成できます。デフォルトは次のとおりです。

```
X_FRAME_OPTIONS = "SAMEORIGIN"
```

1.9.2 コンテンツセキュリティポリシー

Content-Security-Policy は HTML5 標準のパートであり、X-Frame-Options ヘッダー（置き換えられる）よりもレンジ保護を提供します。ウェブサイトの作成者は、リソース（スクリプト、スタイルシート、フォントなど）をロードできる個々のドメイン、およびページの埋め込みを許可されてい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バリデーション `CONTENT_SECURITY_POLICY` オプションがあり、`config_local.py` ファイルで構成できます。デフォルトは次のとおりです。

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

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"
```

!!! Note * このパラメータを追加すると、構成が変更された場合に問題が発生する可能性があります。したがって、PEM のインストールが完了してテストされた後にのみ、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 ヘッダーでアドバタイズされた 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 タイプのスニッフィングをオプトアウトする方法、つまり、MIME タイプが意図的に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、攻撃者は他のユーザーが表示するウェブページにクライアント側のスクリプトを挿入できます。ヘッダーへの HTTP X-XSS-Protection 応答は、Internet Explorer、Chrome、および Safari の機能であり、クロスサイトスクリプティング (XSS) 反射攻撃を検出するとページのローディングを停止します。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ではほとんど不要ですが、まだ CSP を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、ウェブサーバーのリスタートが必要です。

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。

この変更を有効にオーダーには、Apache サービスのリスタートが必要です

`config.py` ファイルの詳細については、[PEM Online Help](#) を参照してください。

1.9.6 Cookie 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。ブラウザはそれらをすべてのリクエストとともに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。`config.py` で参照できる Cookie を安全に make に PEM によって提供されるマルチプル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。

1. **** 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`
2. **** `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) 要求でのみ Cookie を使用します。これにより、ハッカーが XSS の脆弱性を使用して Cookie のコンテンツを学習することを防ぎます。例、`sessionId` Cookie の場合、クライアント側スクリプトで Cookie を読み取る必要はないため、`sessionId` Cookie の場合は、HTTPOnly フラグを常に設定できます。デフォルトは次のとおりです。
- `SESSION_COOKIE_HTTPONLY = True`
3. **** `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と、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`
 - **!!! Note**
 - このオプションは、サーバーが Kubernetes やロードの背後などの動的 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。
 - この変更を有効にオーダーには、Apache サービスのリスタートが必要です。
 - `config.py` ファイルの詳細については、[PEM Online Help](#) を参照してください。

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であると、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乗っ取ることができるため、他のセッションベースの攻撃の危険性が高まります。セッションインターバルが短いほど、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。

Postgres Enterprise Manager は、ユーザ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 コンソールに指定された期間ユーザ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をログします。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、個々のユーザではなく、アプリケーションごとです。タイムアウト期間を構成するには、<PEM_INSTALLATION_PATH>/web ディレクトリにある `config_local.py` ファイルの `USER_INACTIVITY_TIMEOUT` パラメータを変更します。デフォルトでは、この機能は無効になっています。

例、アプリケーションが 15 分間非アクティブになった後にユーザをログ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
```

!!! Note * タイムアウト値は秒単位で指定されます。

この変更を有効にオーダーには、Apache サービスのリスタートが必要です。

`config.py` ファイルの詳細については、[PEM Online Help](#) を参照してください。

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します。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 ヘッダー名前を変更して、生成されたトークンを取得できます。デフォルトでは、ユーザがトークンを作成する要求を送信すると、サーバー応答には「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。ヘッダー名前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 HTTP 要求ヘッダー名前を変更できます。デフォルトでは、ユーザ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前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により、PEM RestAPI 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
```

この変更を有効にオーダーには、Apache サービスのリスタートが必要です。

2.3 PEMで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、組織内のユーザーのロールに基づいてアプリケーションアクセスを制限し、アクセス制御のプライマリ方法の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レベルを指します。ユーザーは、ジョブ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。シンプルに言えば、PEM管理者として、ユーザに最低レベルのロールを付与し、ユーザ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。例、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 'xxxxxx';
```

```
GRANT pem_user, pem_comp_sqlprofiler TO user_sql_profiler;
```

ロールの詳細については、[PEM Roles](#) をご覧ください。

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。通常、データベース管理者は、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ファンクション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着信クエリを調べることで、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。特定のシグネチャ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。

!!! Note * 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 Profiler プラグインの詳細については、[PEM Online Help - SQL Profiler](#) を参照してください。

2.5 パスワード管理

PEM管理ユーザーのセキュリティのヒントの1つは、PEMログインパスワードを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。

- マルチプルアカウントの侵害
- 定数アクセスを防止
- 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の使用を防止します
- 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 はルートユーザとしてインストールされ、ルート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、非ルートユーザ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として実行するには、agent.cfg ファイルの batch_script_user パラメータ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を指定します。非ルートユーザまたはルートユーザ ID を指定できます。ユーザが存在しない場合、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。非ルートユーザが pemagent を実行している場合、batch_script_user の値は無視され、pemagent の実行に使用される同じ非ルートユーザ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 エージェントをリスタートします。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、インストーラは pemAgent および PEM データベースサーバ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への接続時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。

!!! Note * PEM は、SSL CA 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。