**Attaque d'une base de données MySQL**

## Objectifs

Au cours de ces travaux pratiques, vous aurez accès à un fichier PCAP provenant d'une précédente attaque contre une base de données SQL.

**Partie 1: Ouvrir Wireshark et charger le fichier PCAP.**

**Partie 2: Examiner l'attaque par Injection SQL.**

**Partie 3: L'attaque par injection SQL continue...**

**Partie 4: L'attaque par injection SQL fournit des informations système.**

**Partie 5: L'attaque par injection SQL et les informations des tables.**

**Partie 6: L'attaque par injection SQL se termine.**

## Contexte/scénario

Les attaques par injection SQL permettent aux hackers de taper des instructions SQL dans un site web et de recevoir une réponse de la base de données. Ils peuvent ainsi altérer les données se trouvant dans la base de données, usurper des identités et commettre divers méfaits.

Un fichier PPCE a été créé pour que vous puissiez voir une précédente attaque contre une base de données SQL. Vous allez également voir des attaques de base de données SQL et répondre aux questions.

## Ressources requises

- Machine virtuelle Security Workstation

## Instructions

Utilisez Wireshark, un analyseur de paquets réseau courant, pour analyser le trafic réseau. Après avoir lancé Wireshark, ouvrez une capture réseau précédemment enregistrée pour étudier les différentes étapes d'une attaque par injection SQL contre une base de données SQL.

### Partie 1: Ouvrir Wireshark et charger le fichier PCAP.

Vous pouvez ouvrir l'application Wireshark de différentes manières sur un poste de travail Linux.

a. Démarrez la VM Security Workstation.

b. Cliquez **sur Applications > CyberOPS > Wireshark** sur le bureau et accédez à l'application Wireshark.

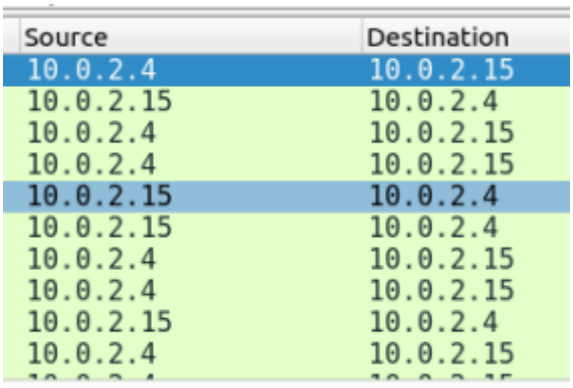
c. Dans l'application Wireshark, cliquez sur **Ouvrir** au milieu de l'application sous Fichiers.

d. Parcourez le répertoire **/home/analyst/** et recherchez **lab .support.files**. Dans le répertoire **lab.support.files** et ouvrez le fichier **SQL_Lab.pcap** .

e. Le fichier PCAP s'ouvre dans Wireshark et affiche le trafic réseau capturé. Ce fichier de capture couvre une période de 8 minutes (441 secondes), soit la durée de cette attaque par injection SQL.

Quelles sont les deux adresses IP impliquées dans cette attaque par injection SQL selon les informations affichées?

10.0.2.4 et 10.0.2.15 sont les 2 adresses IP impliquées.



| Source | Destination |
|---|---|
| 10.0.2.4 | 10.0.2.15 |
| 10.0.2.15 | 10.0.2.4 |
| 10.0.2.4 | 10.0.2.15 |
| 10.0.2.4 | 10.0.2.15 |
| 10.0.2.15 | 10.0.2.4 |
| 10.0.2.15 | 10.0.2.4 |
| 10.0.2.4 | 10.0.2.15 |
| 10.0.2.4 | 10.0.2.15 |
| 10.0.2.15 | 10.0.2.4 |
| 10.0.2.4 | 10.0.2.15 |

## Partie 2: Examiner l'attaque par Injection SQL.

Au cours de cette étape, vous allez visualiser le début d'une attaque.

a. Dans la capture Wireshark, cliquez avec le bouton droit de la souris sur la ligne 13 et sélectionnez **Follow** > **HTTP Stream**. La ligne 13 a été choisie, car il s'agit d'une requête GET HTTP. Cette fonction permet de suivre le flux de données tel que la couche application le voit et mène au test des requêtes à la recherche d'injection de code SQL.

Le trafic de la source s'affiche en rouge. La source a envoyé une requête GET à l'hôte 10.0.2.15. En bleu, l'appareil de destination répond à la source.

b. Dans le champ **Find** Rechercher , saisissiez **1=1**. Cliquez sur **Find Next** (Trouver le suivant)..

c. L'attaquant a entré une requête (1=1) dans une boîte de recherche UserID sur la cible 10.0.2.15 pour voir si l'application est vulnérable à l'injection SQL. Au lieu de répondre avec un message d'échec de connexion, l'application répond avec un enregistrement d'une base de données. Le hacker a donc vérifié que la base de données répond lorsqu'il tape une commande SQL. La chaîne de recherche 1=1 crée une instruction SQL qui sera toujours vraie. Dans l'exemple, quelle que soit la requête saisie dans le champ, elle sera toujours vraie.

d. Fermez la fenêtre Follow HTTP Stream (Suivre le flux HTTP).

e. Cliquez sur **Clear display filter** (Effacer le filtre) d'affichage pour afficher la totalité de la conversation Wireshark.

## Partie 3: L'attaque par injection SQL continue...

Au cours de cette étape, vous allez voir la suite d'une attaque.

a. Dans la capture Wireshark, cliquez avec le bouton droit de la souris sur la ligne 19, puis cliquez sur **Follow** (Suivre) > **HTTP Stream** (Flux http).

b. Dans le champ **Find** (Rechercher), entrez **1=1**. Cliquez sur **Find Next** (Trouver le suivant).

c. L'attaquant a saisi une requête (1' ou 1=1 union select database(), user()#) dans un champ de recherche UserID sur la cible 10.0.2.15. Au lieu de répondre avec un message d'échec de connexion, l'application répond avec les informations suivantes:

Le nom de la base de données est **dvwa** et l'utilisateur de la base de données est **root@localhost**. Plusieurs comptes d'utilisateurs sont également affichés.

d. Fermez la fenêtre Follow HTTP Stream (Suivre le flux HTTP).

e. Cliquez sur **Clear display filter** (Effacer le filtre) d'affichage pour afficher la totalité de la conversation Wireshark.

## Partie 4: L'attaque par injection SQL fournit des informations système.

Le hacker continue et commence à cibler des informations plus précises.

a. Dans la capture Wireshark, cliquez avec le bouton droit de la souris sur la ligne 22 et sélectionnez **Follow** > **HTTP Stream**. Le trafic source indiqué en rouge envoie la requête GET à l'hôte 10.0.2.15. En bleu, l'appareil de destination répond à la source.

b. Dans le champ **Find** (Rechercher), entrez **1=1**. Cliquez sur **Find Next** (Trouver les suivant).

c. L'attaquant a entré une requête (1' or 1=1 union select null, version ()#) dans une boîte de recherche UserID (d'ID d'utilisateur) sur la cible 10.0.2.15 pour trouver l'identifiant de version. Remarquez comment l'identifiant de version à la fin de la sortie, juste avant le </pre>. code HTML de fermeture.

Quelle est la version?

5.7.12-0ubuntu1.1 est la version du système.

```
1' or 1=1 union select null, version ()#<br />First name: <br />Surname:
5.7.12-0ubuntu1.1</pre>
```

d. Fermez la fenêtre Follow HTTP Stream (Suivre le flux HTTP).

e. Cliquez sur **Clear display filter** (Effacer le filtre) d'affichage pour afficher la totalité de la conversation Wireshark.

## Partie 5: L'attaque par injection SQL et les informations sur les tables.

Le hacker sait qu'il y a de nombreuses tables SQL contenant beaucoup d'informations. Le hacker essaie de les trouver.

a. Dans la capture Wireshark, cliquez avec le bouton droit sur la ligne 25 et sélectionnez **Follow** > **HTTP Stream**. La source est indiquée en rouge. L'appareil source a envoyé une requête GET à l'hôte 10.0.2.15. En bleu, l'appareil de destination répond à la source.

b. Dans le champ **Find** (Rechercher), saisissez **Users** (les utilisateurs). Cliquez sur **Find Next** (Trouver le suivant).

c. L'attaquant a saisi une requête (1'or 1=1 union select null, table_name from information_schema.tables#) dans un champ de recherche UserID (d'ID d'utilisateur) sur la cible 10.0.2.15 pour afficher toutes les tables de la base de données. Une sortie indiquant de nombreuses tables est générée, car le hacker a spécifié « null » sans plus de précision.

Quelle serait la commande modifiée de (**1' OR 1=1 UNION SELECT null, column_name FROM INFORMATION_SCHEMA.columns WHERE table_name='users'**) ?


Le hacker a changer la partie schema.columns en schema.tables

d. Fermez la fenêtre Follow HTTP Stream (fenêtre Suivre le flux HTTP).

e. Cliquez sur **Clear display filter** (Effacer le filtre) d'affichage pour afficher la totalité de la conversation Wireshark.

## Partie 6: L'attaque par injection SQL se termine.

L'attaque se termine ayant permis au hacker de récupérer le meilleur butin possible: les valeurs de hash des mots de passe.

a. Dans la capture Wireshark, cliquez avec le bouton droit de la souris sur la ligne 28 et sélectionnez **Follow** > **HTTP Stream**. La source est indiquée en rouge. L'appareil source a envoyé une requête GET à l'hôte 10.0.2.15. En bleu, l'appareil de destination répond à la source.

b. Cliquez sur **Find** (Rechercher) et tapez **1=1**. Recherchez cette entrée. Lorsque le texte est localisé, cliquez sur **Cancel** (Annuler) dans la zone de recherche du texte à trouver.

Le hacker a saisi une requête (1'or 1=1 union select user, password from users#) dans une boîte de recherche d'ID d'utilisateur sur la cible 10.0.2.15 pour extraire les noms d'utilisateurs et les valeurs de hash des mots de passe !

À quel utilisateur correspond la valeur de hash de mot de passe 8d3533d75ae2c3966d7e0d4fcc69216b?


1337 est le User

c. En utilisant un site Web tel que https://crackstation.net/, copiez le hachage du mot de passe dans le craqueur de hachage de mot de passe et commencez à craquer.

Quel est le mot de passe en texte clair?

charley est le MDP


d. Fermez la fenêtre Follow HTTP Stream (Suivre le flux HTTP). Fermez les fenêtres ouvertes.

## Questions de réflexion

1. Quel est le risque de voir les plateformes utiliser le langage SQL ?

Si une application ne valide pas correctement les données entrées par l'utilisateur, un pirate peut insérer du code SQL malveillant dans un champ de saisie (comme un formulaire de connexion).

Ce code peut alors être exécuté par le serveur de base de données, ce qui permet d'accerder à des informations confidentielles (mots de passe, données personnelles) ou de modifier ou supprimer des données mais aussi de prendre le contrôle du serveur de base de données.

**Exemple :**

```
Entrée utilisateur : ' OR '1'='1
```

Si le code de vérification est mal protégé, cette entrée peut forcer une requête à toujours être vraie et permettre à un pirate de se connecter sans mot de passe.

2. Naviguez sur Internet et effectuez une recherche sur "prévenir les attaques par injection SQL". Quelles sont les 2 méthodes ou mesures à appliquer pour empêcher les attaques par injection SQL?

### Mesures de sécurité pour protéger votre base de données contre l'injection SQL

**Utilisation d'instructions préparées et de requêtes paramétrées**. Lorsqu'un développeur utilise des requêtes paramétrées, il doit définir l'intégralité du code SQL, puis passer chaque paramètre. Il est donc impossible pour un attaquant de modifier l'objet de la requête ultérieurement.

**Utilisation de la validation des saisies dans la liste d'autorisation**. Dans certains cas, les développeurs peuvent définir une valeur attendue, comme le nom autorisé d'un tableau ou d'une colonne. Cette validation permet d'empêcher l'ajout d'une saisie utilisateur non approuvée à une requête.

Source : https://www.crowdstrike.com/fr-fr/cybersecurity-101/cyberattacks/sql-injection-attack/