TP4: Sécurité des applications WEB

Objectifs

- Méthodologies de conduites d'un pentest
- HTTP
- Web Fuzzing
- Injections SQL
- File Upload
- Post-Exploitation/Maintien de l'accès/Exfiltration
- XSS

Au cours de ce TP vous allez vous familiariser avec les risques de sécurité des applications Web. Lors de ce TP vous allez vous trouvez face à un cas réaliste. Pour le mener à bien vous allez suivre les différentes phases de réalisation d'un test d'intrusion. Après avoir mener une phase de reconnaissance, vous allez exploiter certaines des vulnérabilités WEB les plus connues et les critiques. Enfin, vous serez sensibilisés aux problématiques de POST Exploitation.

Préparation du TP

Mise en place

Vous disposez d'une VM Kali qui sera branchée sur le même réseau que le serveur WEB vulnérable.

Dans ce TP vous utiliserez différents outils (embarqués dans Kali) durant chaque phase de l'attaque. Parmis eux vous utiliserez des outils avec lesquels vous avez déjà travaillés (nmap, John the Ripper, etc.)

Introduction au proxy Burp

En cours, vous avez étudier le rôle du proxy (HTTP/HTTPs) en défense dans le cadre des architectures sécurisées, notamment pour ses capacités de filtrage, cloisonnement et traçage.

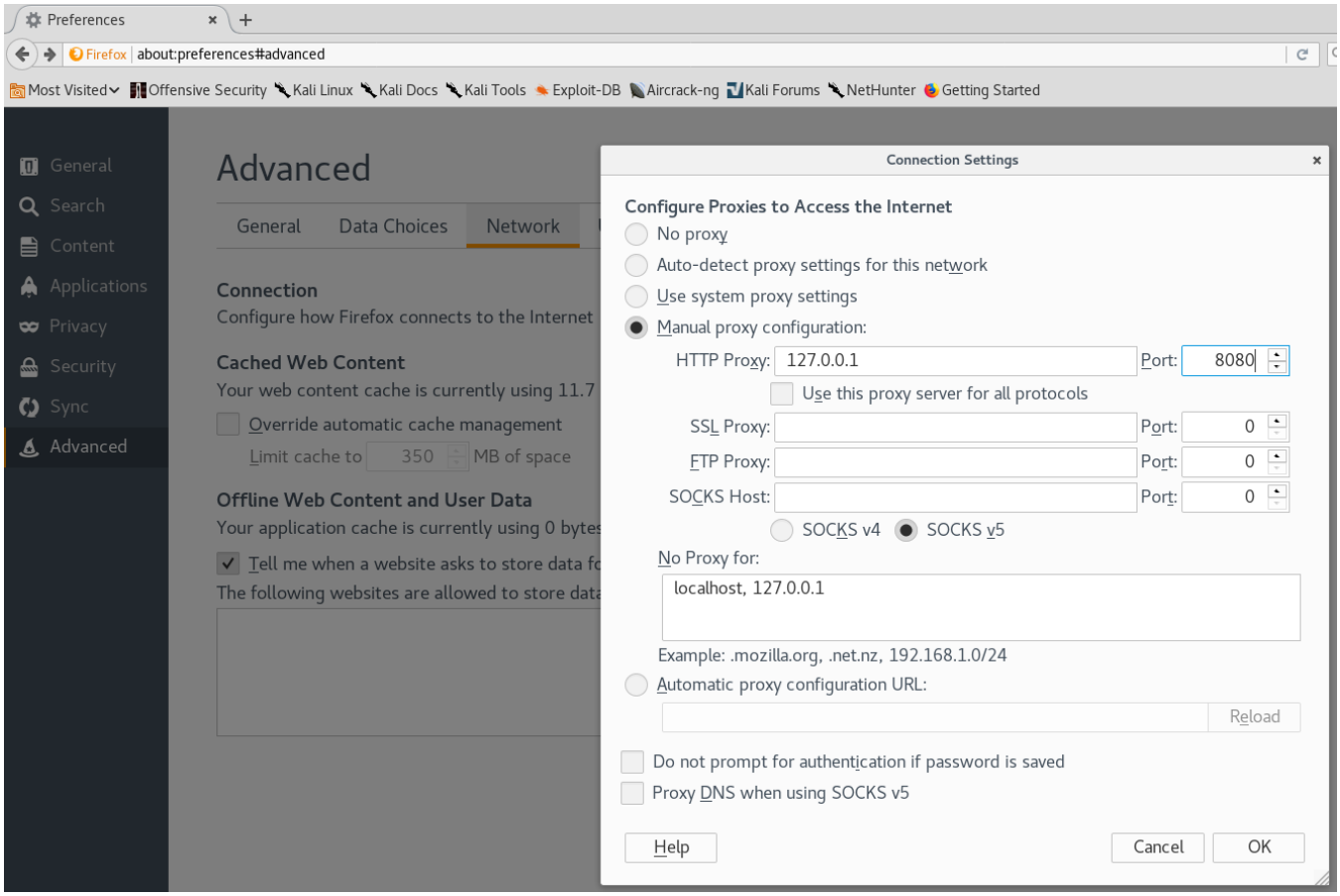
En réalité le proxy est aussi un outil utilisé en attaque. Le proxy est un support précieux pour un attaquant. Il permet de garder un historique de l'ensemble des requêtes qui ont été générées entre l'attaquant et le serveur web victime. Un attaquant peut se servir de ces requêtes pour comprendre le fonctionnement de l'application WEB, retrouver les détails qui sont masquées par le navigateur ou les outils d'attaques (headers, requêtes AJAX, etc.). De point de vue d'un attaquant, vous pouvez l'imaginez comme une sorte de Wireshark spécialisé pour le WEB

C'est un outil indispensable pour mener des attaques WEB. Evidemment, des proxys orientés attaques ont été développés, et mettent en avant les points intéressant pour un attaquant, ainsi que de nouvelles fonctionnalités (rejeu/interception/modification/forge de nouvelles requêtes, attaques automatiques, etc).

Dans ce TP vous allez utiliser BURP, un de ces proxys, qui est présent dans la distribution Kali et implémente ces fonctionnalités.

Reconnaissance / Fingerprinting

Lancez le proxy BURP et laissez le tourner pour le reste de la séance, n'hésitez pas à le consulter durant chaque attaque/navigation. Configurez votre navigateur Firefox de manière à ce qu'il utilise votre proxy (uniquement pour le trafic HTTP, pas HTTPS).



Attention: n'oubliez pas de spécifier le proxy lors de vos attaques. Observez les options qui correspondent au proxy. Les outils d'attaques WEB supportent quasiment tous l'usage d'un proxy.

Naviguez sur le WEB et familiarisez vous avec BURP, regardez notamment le format des requêtes et des réponse HTTP, les headers (cookies,etc.).

Utilisez nmap avec les bonnes options afin de retrouver l'adresse IP du serveur WEB sur le réseau. (Il est uniquement accessible en HTTP)

Ajoutez l'@ IP du serveur dans votre fichier /etc/hosts, nommez le unsafe

Une simple requête GET permet de retrouver un grand nombre d'informations sur le serveur à attaquer.

Utilisez l'outil curl avec l'option -I (pour afficher uniquement les headers HTTP) pour faire un GET sur unsafe. Identifiez le système d'exploitation, le serveur WEB et sa version, ainsi que la technologie utilisée pour développer l'application Web.

Souvent, certaines pages d'un site sont cachés aux utilisateurs (c'est notamment le cas pour les urls d'administration). Cela ne veut pas dire qu'elles ne sont pas accessibles.

Utilisez le fuzzer HTTP wfuzz afin de retrouver les urls sensibles du serveur unsafe. Voici la syntaxe à utiliser:

```
wfuzz -c -w /usr/share/wordlists/wfuzz/general/big.txt -p {@IP_PROXY:PORT} --hc 404 http://unsafe/FUZZ
```

En analysant les requêtes générées par cet outil sur Burp, expliquez le fonctionnement du fuzzer Web wfuzz. Expliquez le rôle de l'option --hc 404.

Entrainement SQL

Vous allez avoir accès à un serveur MySQL pendant le TP pour vous entraîner à la syntaxe de ce SGBDR afin de mieux exploiter les injections SQL.

Utilisez la commande suivante (en adaptant l'@IP) pour vous connecter à la Base de donnée "pokedex" :

mysql -h {@IP} pokedex --user test --password=test

Entrainez vous tant que vous voulez sur cette base, notamment les requêtes suivantes:

SELECT 1,2; **//affiche des données**
SHOW TABLES; **//liste toutes les tables de la base**
DESC monsters; **//affiche tous les champs de la table "monsters"**
SELECT name, type1, type2 FROM monsters; **//affiche le contenu de champ de toutes les entrées de la table "monsters"**
SELECT name, type1, type2 FROM monsters WHERE name="Pikachu";
SELECT name, type1, type2 FROM monsters WHERE name="Pikachu" UNION SELECT 1,2,3;
Etc...

Revenez vous entraîner sur ce serveur à chaque fois que vous aurez un doute sur vos requêtes pendant l'exploitation de l'injection SQL.

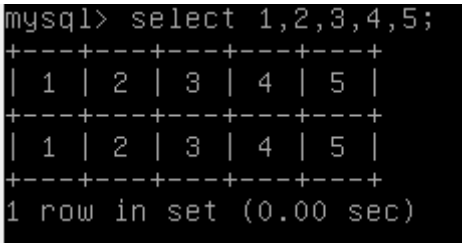
Détecter une injection SQL

Naviguez sur l'URL http://unsafe/picture.php?id=1

Changez la valeur du paramètre id, essayez différentes combinaisons. Que constatez-vous ?

Rappel: format d'un requête SQL permettant de retrouver des informations stockées en base de donnée:

```
SELECT colonne1, colonne2, FROM table1 WHERE colonne4='string1' AND colonne5=entier1;
```



Voici le code PHP utilisé par le serveur web vulnérable:

```
$id = $_GET["id"];
$result= mysql_query("SELECT * FROM articles WHERE id=".$id);
```

Le paramètre id est directement ajouté à la requête SQL.

En fonction du SGBDR, de la requête SQL, de la techno Web utilisé, du paramètre vulnérable, etc. il peut être difficile de détecter une injection SQL.

En "jouant" avec le paramètre id et le langage SQL identifiez plusieurs méthodes différentes squi permettent de détecter l'injection dans notre cas.

Exploiter une injection SQL

Félicitations ! Vous avez réussi à détecter une injection SQL sur le serveur WEB. Désormais, il vous reste à l'exploiter pour retirer des informations stockées en base qui pourrait vous intéresser.

Rappel: Le mot clé **UNION** permet de récupérer ensemble les informations de 2 requêtes

```
SELECT id, nom, prix FROM articles WHERE id=2
UNION SELECT id, login, password  FROM users
```

Attention: Il faut que les 2 requêtes retournent le même nombres de colonnes pour que la requêtes SQL soit valide !

Essayez une injection avec le morceau de requête: UNION SELECT 1
Que ce passe-t'il? Expliquez pourquoi la requête n'est pas valide.

Le mot clé SQL **ORDER BY** peut être utilisé pour retrouver le nombre de colonnes de la requête principale.

```
SELECT prenom, nom, age, groupe FROM users ORDER BY 3
```

Le résultat de cette requête sera trié selon la 3e colonne. Si cela dépasse le nombre de colonne, la requête SQL est invalide et on aurait une erreur:

> **Unknown column '10' in 'order clause'**

En utilisant ce comportement, retrouvez le nombre de colonnes de la requête SQL principale.

Maintenant que vous connaissez le nombre de colonnes. Il est possible de retirer des informations de la base. Il nous faudrait par exemple le nom des tables et des colonnes.

Grâce aux messages de debug, nous savons que le serveur de base de donnée utilisé est MySQL. Or MySQL met à disposition des tables génériques permettant de retrouver les informations que l'on recherche:

- La liste des tables: **SELECT table_name FROM information_schema.columns**
- La liste des colonnes: **SELECT column_name FROM information_schema.columns**

Utilisez le mot clé SQL UNION et ces requêtes pour retrouver le noms des tables et colonnes de la base de données du serveur unsafe.

Utilisez la fonction MySQL concat pour avoir un résultat lisible: concat(table_name,':', column_name)

En utilisant ces informations, forgez une requêtes SQL pour retrouver les logins/mots de passes des utilisateurs de l'application Web.

A votre avis, en tant que développeur, comment peut on se protéger contre les injections SQL ?

Cracking

Les mots de passes n'étaient pas stockés en clair.

Utilisez internet et John the Ripper (attaque par dictionnaire en utilisant un dictionnaire présent sur kali ==> /usr/share/wordlists/fasttrack.txt) pour casser le mot de passe des utilisateurs.

File Upload

Connectez vous sur l'application avec les identifiants de l'administrateur puis rendez vous sur l'url d'administration que vous aviez découverte en début de TP avec wfuzz.

Vous avez maintenant accès à de nouvelles fonctionnalités. Notamment à l'upload d'images.

L'idée de l'attaque va être d'uploader du code PHP à faire exécuter par le serveur.

Créez un ficher .php contenant le code suivant:

```
<?php
        echo "Hello Master";
```

Essayez d'uploader ce fichier sur le serveur, que ce passe-t-il ?

L'extension .php semble filtrée, néanmoins par défaut apache est configuré pour interprété le code PHP sur un grand nombre d'extensions (.php2, .php3, etc.)

Essayez de modifier l'extension du fichier précédent de telle sorte a ce qu'il soit quand même interprété par le serveur WEB, puis uploadez le de nouveau. que ce passe-t-il ?

En vous aidant de la fonction php => "system", uploadez un fichier php qui exécutera les commandes que vous lui envoyez en paramètre (le code php pour récupérer le contenu d'un paramètre monParam en GET est $_GET["monParam"]).

A votre avis, en tant que développeur, comment peut on se protéger contre les vulnérabilité de type File Upload ?

Post exploitation

Bravo ! Vous avez maintenant pris la main sur le serveur via votre script PHP. On appelle cet accès un **webshell**. Il possède l'avantage d'être relativement discret et de passer les protections réseaux (firewall).

Néanmoins c'est un accès assez rudimentaire, et il ne fonctionnera pas si l'on veut utiliser des outils interactifs, notamment pour continuer à se propager. Pour cela nous aimerions un véritable shell interactif. Il s'agit d'une problématique de POST Intrusion, c'est à dire de manière générale, que faire une fois qu'on a pris la main sur une machine.

En vous aidant de l'outil ncat (netcat) présent sur le serveur WEB compromis. Améliorez votre accès de manière à avoir un shell interactif. Pour cela, lancez un serveur TCP en écoute sur un port et qui exécutera vos commandes. Pour le client sur kali vous utiliserez nc (une autre variante de netcat).