Sécurisation des applications *Web*

Activité 4 : Brèche sur des informations confidentielles

Propriétés	Description
Intitulé long	Exploitation d'une plateforme d'apprentissage des vulnérabilités des applications *Web*
Intitulé court	Sécurisation des applications *Web*
Formation concernée	BTS Services Informatiques aux Organisations
Matière	Bloc 3 : Cybersécurité des services informatiques en deuxième année SLAM
Présentation	Ce Côté labo a pour objectif d'exploiter la plateforme d'apprentissage Mutillidae du groupe *OWASP* (*OpenWeb Application Security Project*) afin de se familiariser avec les principales vulnérabilités des applications *Web*. Chaque activité couvre une problématique spécifique (*SQLi*, *XSS*, *CSRF*…) en référence au top 10 des vulnérabilités décrites par l'*OWASP*. Dans un premier temps, l'étudiant doit réaliser les attaques associées à chaque vulnérabilité. Dans un deuxième temps, l'objectif est d'analyser et de comprendre les codes sources des scripts présentés dans leur forme non sécurisée puis sécurisée en tant que contre-mesure. **Cette quatrième activité** traite des vulnérabilités associées aux brèches sur des informations confidentielles. Cette faille arrive en 3ième position dans le classement *OWASP* 2017.

Compétences	• Protéger les données à caractère personnel ;
	  ○ Identifier les risques liés à la collecte, au traitement, au stockage et à la diffusion de données à caractère personnel.
	• Garantir la disponibilité, l'intégrité et la confidentialité des services informatiques et des données de l'organisation face à des cyberattaques.
	  ○ Caractériser les risques liés à l'utilisation malveillante d'un service informatique ;
	  ○ Recenser les conséquences d'une perte de disponibilité, d'intégrité ou de confidentialité.

Savoirs	• Chiffrement, authentification et preuve ; principes et techniques ; • Sécurité des applications web : risques, menaces et protocoles.
Prérequis	Commandes de base d'administration d'un système *Linux*, langages *PHP* et *JavaScript*. Dans l'activité 1, avoir lu la présentation (*owasp-presentation-v1.1*) et réalisé les installations décrites dans le fichier *owasp-mise_en_place-v1.1*.
Outils	Deux machines éventuellement virtualisées sont nécessaires avec *Linux* comme système d'exploitation. Sites officiels : https://www.owasp.org et https://portswigger.net/burp/communitydownload
Mots-clés	OWASP, Mutillidae 2.6.60, BurpSuite 1.7.29, vulnérabilités, SQLi, XSS, IDOR.
Durée	Une heure minimum pour cette activité.
Auteur(es)	Patrice DIGNAN avec la relecture de Valéry TSCHAEN
Version	v 1.0
Date de publication	Novembre 2020

I Problématique des informations confidentielles

Les informations confidentielles sont des données ne devant en aucun cas être dévoilées à des tiers autres que ceux qui disposent des autorisations pour les consulter ou les traiter. Parmi les données confidentielles, on trouve :

Les données à caractère personnel

D'après la CNIL, une **donnée à caractère personnel (DCP)** est toute information se rapportant à une personne physique identifiée ou identifiable. Une personne physique peut être identifiée directement ou indirectement à partir du croisement d'un ensemble de données. Parmi les informations personnelles, certaines **données sont qualifiées de sensibles** car révélant une appartenance politique ou syndicale, des problèmes de santé ou toute autre information qui pourrait entraîner une discrimination.

Quelques exemples :

- numéro de sécurité social ;
- traitement pris par un patient ;
- identifiants de connexion ;
- informations sur des moyens de paiement...

Les autres données confidentielles

Il s'agit notamment des données techniques relatives à des configurations systèmes et réseaux qui ne doivent pas être divulguées au risque d'être exploitées de manière malveillante.

Quelques exemples :

- fichier de configuration décrivant l'architecture d'un serveur *Web* ;
- plan du réseau d'une entreprise avec l'adressage IP ;
- détails des configurations ;
- code source de certains programmes ;
- tout type d'information sensible pouvant affecter la sécurité du système informatique...

II Classification technique des données confidentielles

La sécurisation des données confidentielles nécessite de bien appréhender la classification technique suivante :

- **Les données au repos** : il s'agit des données archivées sur un support et ne faisant pas l'objet d'un traitement.
- **Les données en cours de traitement** : il s'agit des données qui sont manipulées par un programme effectuant un traitement permettant d'obtenir un résultat.
- **Les données en transit** : il s'agit des données qui circulent à travers un réseau de communication (filaire ou sans fil).

Une donnée peut passer d'une catégorie à une autre. Cette classification n'est pas spécifique aux données confidentielles mais est nécessaire pour appliquer les traitements de sécurisation adéquats.

III Les enjeux de la sécurité des données confidentielles

1 Panorama des risques

Les principaux **évènements redoutés** liées à une carence de sécurité sur les données confidentielles sont les suivants :

- accès illégitime aux DCP et aux autres informations confidentielles ;
- modification non désirée des DCP et des autres informations confidentielles ;
- disparition des DCP et des autres informations confidentielles ;
- exploitation malveillante des informations confidentielles obtenues.

Ces évènements redoutés peuvent se manifester via les **menaces** suivantes :

- espionnage d'un matériel permettant d'observer des données interprétables tels que des identifiants de connexion ;
- détournement d'usage d'une application permettant d'obtenir de manière illégale des données, d'élever ses privilèges et de croiser des données ;
- analyse d'un logiciel via l'étude d'un code source permettant de déterminer les défauts exploitables, de tester des réponses d'une base de données à des requêtes malveillantes ;
- attaque de type *MITM* (*Man In The Middle*) via un canal informatique permettant de modifier ou d'ajouter des données à un flux réseau et d'effectuer des rejeux (réémission d'un flux intercepté) ;
- dépassement des limites d'un logiciel permettant de substituer un dispositif légitime par un dispositif illégitime de captation des données ;
- limites d'une personne : manipulation et ingénierie sociale permettant de révéler des informations et de les croiser afin d'obtenir des informations confidentielles.

2 Conséquences des évènements redoutés

Les conséquences d'une brèche sur la confidentialité des informations sont les suivantes :

- usurpation d'identité permettant d'effectuer des opérations frauduleuses : envoi de mail mensongers, fraudes financières… ;
- impact sur l'image et la vie privée des salariés ;
- préparation et mise en place d'une attaque sur le réseau informatique suite aux informations recueillies, mise en place d'une porte dérobée (*backdoor)* permettant à l'attaquant de revenir sur le système cible ;
- perte d'argent pour l'entreprise suite aux opérations frauduleuses commises ou suite aux condamnations judiciaires dues à des négligences sur le stockage et le traitement des données confidentielles ;
- atteinte à la réputation de l'entreprise, les clients préférant s'adresser à un concurrent qui sécurise mieux les données confidentielles ;
- destruction des données pouvant compromettre la survie de l'entreprise ;
- impacts organisationnels liées aux conséquences sur les services des fuites constatées ;
- impact catastrophique si l'organisation victime est classée comme Opérateur d'Importance Vitale (OVI). Un **opérateur d'importance vitale** (**OIV**) est, en France, une organisation identifiée par l'État comme ayant des activités indispensables à la survie de la nation ou dangereuses pour la population (https://fr.wikipedia.org/wiki/Op%C3%A9rateur_d%27importance_vitale) .

IV La réglementation concernant les données confidentielles

La réglementation sur l'archivage et la protection des données prévoit des limitations de durée pour la conservation de ces données et prévoit aussi que des mécanismes de protections adaptés soient mis en place.

Concernant la durée de conservation des données, les principaux chiffres associés au RGPD (Règlement Général sur la Protection des Données) sont :

- 3 ans : les données personnelles des personnes inactives depuis 3 ans doivent être supprimées ;
- 13 mois : il faut redemander tous les 13 mois le consentement des visiteurs pour le traitement des cookies ;
- 1 mois : délai pour traiter une demande d'accès de rectification ou de suppression de donnée personnelle.

Concernant la protection de ces données, on peut citer les articles suivants :

- article 32 du RGPD sur l'obligation d'appliquer un traitement sécurisé aux données à caractère personnel ;
- article 33 du RGPD sur la notification à l'autorité de contrôle d'une violation des données à caractère personnel ;
- article 35 du RGPD sur la mise en place d'une analyse d'impact relatif à la protection des données (AIPD) ;

V Bonnes pratiques

Pour éviter une brèche sur les données confidentielles, les bonnes pratiques de réflexion suivantes peuvent être mises en place :

- Recenser les données confidentielles : fichiers, DCP…
- Classer le trafic des données selon les protocoles utilisés : *HTTP*, *HTTPS*, *SMTP*…
- S'assurer que le trafic interne et externe au réseau est sécurisé eu égard aux critères *DIC* (confidentialité, intégrité et disponibilité) notamment en ce qui concerne les données en transit qui font l'objet d'une sauvegarde.
- Le réseau de l'entreprise fait-il appel à des algorithmes de chiffrement ou des protocoles non faiblement sécurisé et obsolètes ?
- S'assurer d'une gestion sécurisée des clés et les certificats utilisés par l'entreprise, par exemple via des mécanismes de révocation quand cela est nécessaire.
- Appliquer des contrôles de sécurité sur les données confidentielles en fonction de leur état (repos, transit, traitement).
- Ne pas stocker de données ou des fichiers confidentiels inutiles aux traitements visés. Seul ce qui est vraiment utile doit être stocké.
- Désactiver les caches pour les réponses des serveurs qui contiennent des informations confidentielles.
- Stocker les mots de passe de manière hachée avec une fonction de salage.
- Utiliser les technologies de confidentialité persistante *PFS (Perfect Forward Secrecy*) et *HSTS* (*HTTP Strict Transport Security*) sur les sites *Web*.

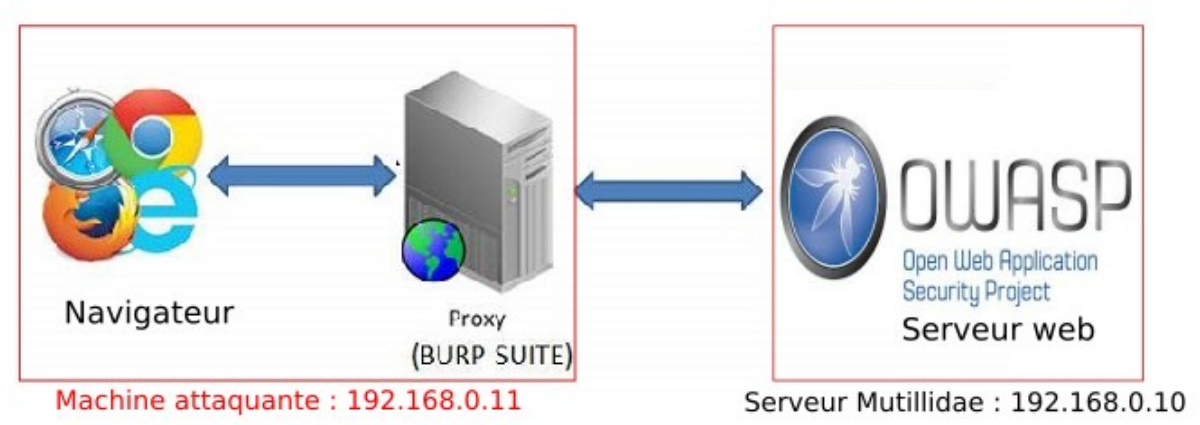
VI Objectifs et architecture générale des labos

Deux défis sont proposés pour illustrer la problématique des brèches sur les informations confidentielles :

1. découverte d'une page cachée contenant des informations de configurations confidentielles (page *phpinfo*) ;
2. attaque de type *SSLSTRIP* visant à profiter d'une brèche dans la redirection *HTTPS* afin d'obtenir des identifiants de connexion.

Ces deux défis peuvent être réalisés de manière indépendante. Chaque défi est associé à un dossier documentaire.

Pour rappel, l'environnement de travail est le suivant :



Le serveur *Mutillidae* propose un site *Web* conçu pour identifier et tester les failles de sécurité identifiées par l'*OWASP*. Il est possible pour chacune d'entre elles, de définir le niveau de sécurité appliqué.

Notre démarche consistera, pour les défis présentés :

- à partir de la version non sécurisée de la page concernée et à mettre en évidence la faille de sécurité ;
- nous constaterons ensuite que dans la version sécurisée de cette page fournie par *Mutillidae*, l'attaque n'est plus possible ;
- l'étude des mécanismes de sécurisation utilisés, donc du code de la page associée, permettra de dégager des bonnes pratiques de programmation.

Le premier défi nécessite d'utiliser l'outil *BurpSuite*. Quant à la machine attaquante, elle comprend un navigateur ainsi que le *proxy BurpSuite* qui permet d'intercepter les requêtes avant de les envoyer au serveur. L'objectif étant de modifier les paramètres de certaines requêtes afin de tester des injections de code. Par exemple, la valeur saisie pour le *login* sera remplacée par du code *JavaScript*.

Le deuxième défi cible toujours le serveur *Mutillidae*. Les outils d'attaques utilisés sont les logiciels ***arpspoof*** et ***sslstrip***.

VII Premier défi : Affichage d'une page de configuration confidentielle

1 Objectif

Le premier défi a pour objectif d'afficher le contenu de la page ***phpinfo*** qui contient des informations de configurations confidentielles sur le serveur *Web*. Ces informations pourraient être exploitées de manière malveillante.

2 À vous de jouer

Travail à faire 1 Mise en place de l'attaque par *fuzzing*

Le but de cette première série de questions est de mettre en œuvre l'attaque permettant d'afficher une page confidentielle contenant des informations de configuration.

Q1. Commencer par préparer votre environnement de travail en démarrant le serveur *Mutillidae* ainsi que la machine cliente contenant le *proxy BurpSuite*. Vérifier que vos deux machines communiquent à l'aide de la commande *ping*. Puis, positionner le niveau de sécurité de *Mutillidae* à 0.

Q2. À l'aide du dossier documentaire n°1, réaliser le premier défi permettant d'afficher la page confidentielle *phpinfo* en étant non identifié. Vous prendrez soin de réaliser vos propres captures d'écrans dans votre compte rendu de TP.

Travail à faire 2 Nouvelle tentative en mode sécurisé et analyse du code source

Le but de cette deuxième partie est de tester à nouveau l'attaque après activation du codage sécurisé et de comprendre l'encodage mis en place.

Test du niveau 1 de sécurité :

Q1. Est-ce que le niveau de sécurité 1 permet d'éviter cette brèche d'information ?

Q1. Se rendre sur le code source de la page *phpinfo.php* sur le serveur pour expliquer le comportement du serveur face à cette attaque avec ce niveau de sécurité.

Test du niveau 5 de sécurité :

Q2. Est-ce que le niveau de sécurité 5 permet d'éviter cette brèche d'information ?

Q3. Expliquer le mécanisme de sécurité mis en œuvre dans le code source. Tous les utilisateurs ont-ils interdiction d'accéder à cette page ?

Q4. Proposer une solution de sécurité basée sur la configuration du fichier *php.ini* du serveur *Web* qui empêcherait tout utilisateur, y compris l'administrateur, d'accéder à cette page par le *Web*. Le fichier *php.ini* est situé dans */etc/php/7.0/apache2*.

Prolongement possible :
Reprendre une application existante (TP/PPE) et mettre en place le niveau de sécurité 5.

VIII Deuxième défi : Brèche dans la configuration *SSL*

1 Objectif

Ce second défi a pour objectif de transformer une redirection *HTTPS* en connexion non chiffrée *HTTP* afin de capturer les identifiants de la page de connexion.

2 À vous de jouer

Les questions suivantes se traitent en suivant les étapes décrites dans le dossier documentaire

Travail à faire 3 Mise en place de l'attaque

Dans un premier temps, l'objectif est de réaliser une attaque visant à rediriger une connexion HTTPS en HTTP afin de capturer des identifiants de connexion.

Q1. Commencer par préparer votre environnement de travail en démarrant les trois machines suivantes :
- le serveur *mutillidae* ;
- la machine attaquante (celle qui contient le logiciel BurpSuite) ;
- la machine victime.

Vérifier que ces trois machines communiquent puis positionner le niveau de sécurité de *Mutillidae* à 0.

Q2. À l'aide du dossier documentaire n°2, réaliser le deuxième défi permettant de compromettre les identifiants de connexions de la victime via une brèche dans la configuration *SSL* (redirection *HTTP*). Vous prendrez soin de réaliser vos propres captures d'écrans dans votre compte rendu de TP. Pour vous aider à comprendre ce défi, vous pouvez consulter la page suivante sur *Mutillidae : A3=> Sensitive Data Exposure > SSL Misconfiguration.*

Travail à faire 4 Codage sécurisé et analyse du code source

Le but de cette deuxième partie est de tester à nouveau l'attaque après activation du codage sécurisé et de comprendre le codage mis en place pour sécuriser la page.

Test du niveau 1 de sécurité :

Q1. Est-ce que le niveau de sécurité 1 permet d'empêcher cette attaque ?

Q2. Est-ce que le niveau de sécurité 1 permet d'empêcher les conséquences de cette attaque ? Où se situe la faille ?

Q3. Expliquer le code source mis en œuvre avec ce niveau de sécurité en analysant la page *index.php*.

Test du niveau 5 de sécurité :

Q4. Est-ce que le niveau de sécurité 5 permet d'empêcher l'attaque ?

Q5. Est-ce que le niveau de sécurité 5 permet d'empêcher les conséquences de l'attaque ?

Q6. Expliquer le code mis en œuvre avec ce niveau de sécurité en analysant la page *index.php*. Vers quelles page est-on redirigé en cas d'attaque ? En déduire la bonne pratique à mettre en œuvre en cas de besoin de confidentialité sur les applications *Web* ?

Prolongement possible :
Reprendre une application existante et mettre en place le niveau de sécurité 5.

IX Conclusion

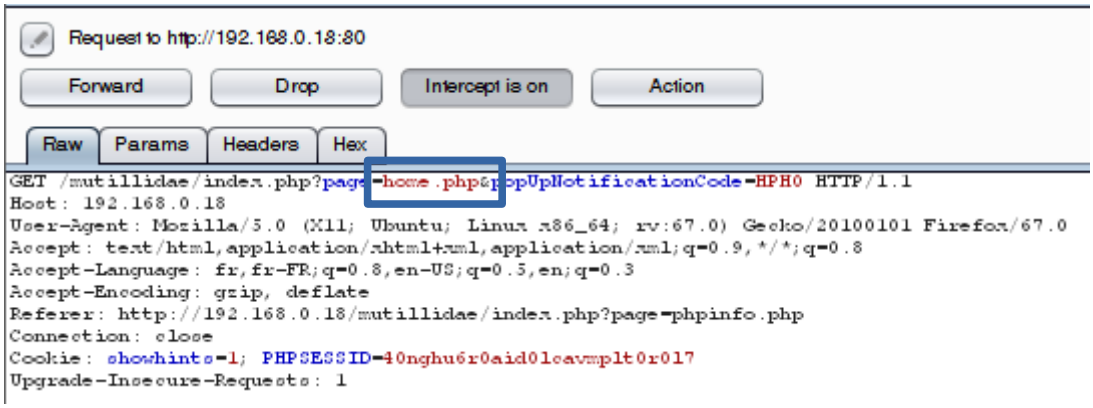
La gestion des informations confidentielles est essentielle pour toute organisation. Un travail en amont doit être effectué afin d'identifier et de classer ces informations et d'y appliquer des mesures de confidentialité adéquates. Si le chiffrement reste une contre-mesure essentielle, le développeur doit correctement intégrer cette dernière pour éviter que des identifiants de connexion soient compromis. La loi rend obligatoire la mise en place de traitements permettant de gérer la sûreté et la sécurité de ces informations.

Dossier documentaire

Dossier 1 : Récupération de la page secrète de configuration *phpinfo*

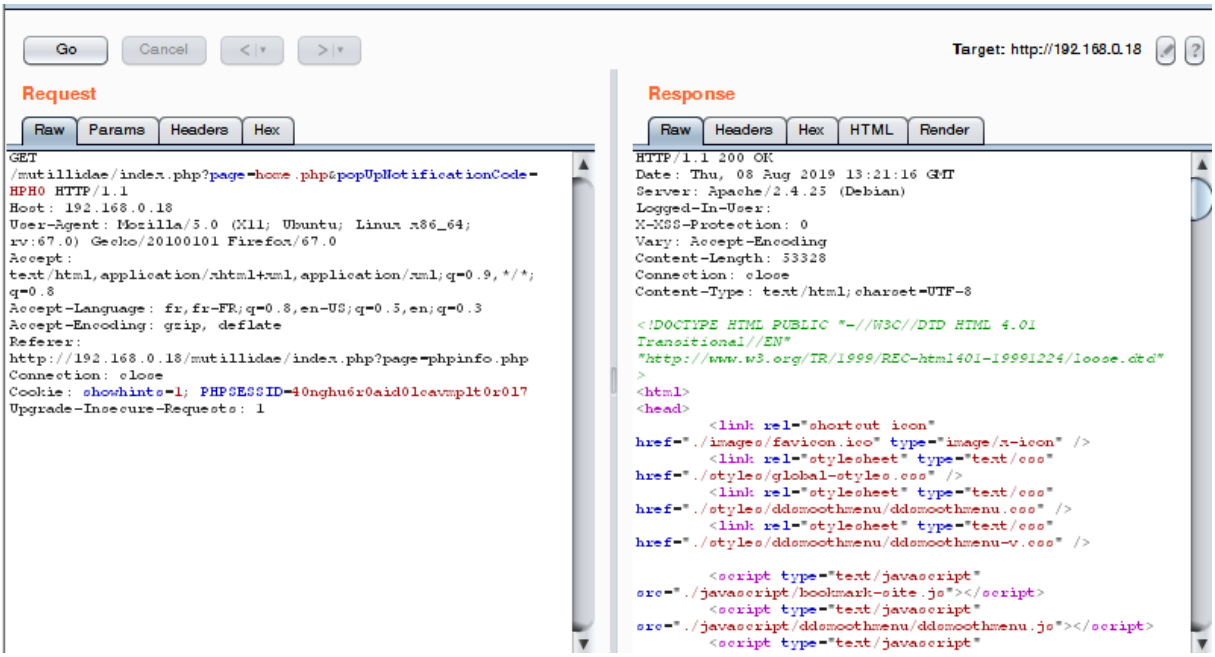
Étape n°1 : Préparation de l'attaque

S'assurer que le *proxy BurpSuite* est à **intercept is on**. Ensuite, se rendre sur la page d'accueil de *Mutillidae*. La connexion est interceptée par le *proxy*.



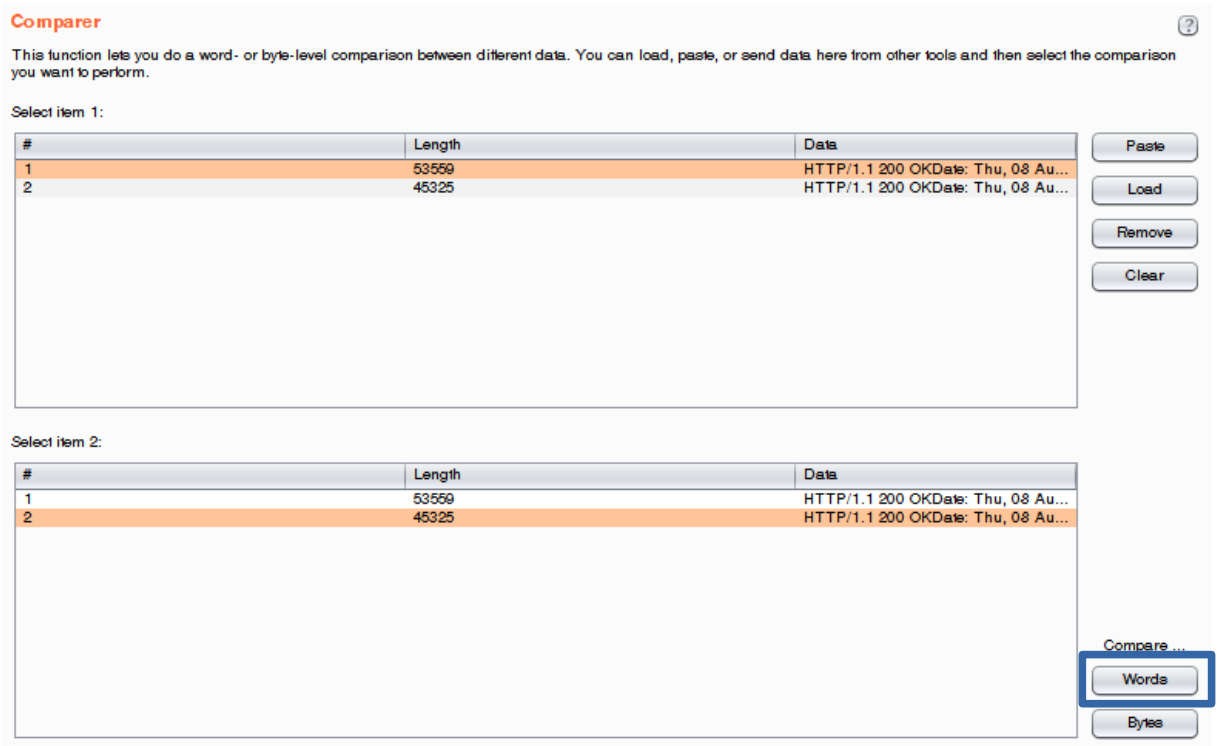
Il s'agit ici d'une page légitime qui existe vraiment sur le serveur. L'idée est de comparer la réponse retournée par le serveur entre une page légitime et une page inexistante.

Faire un clic droit au milieu de l'intercepteur et cliquer sur **Send to Repeater**. Se rendre sur l'onglet **Repeater** et cliquer sur le bouton **Go** pour voir la réponse obtenue sur la page existante *home.php*.

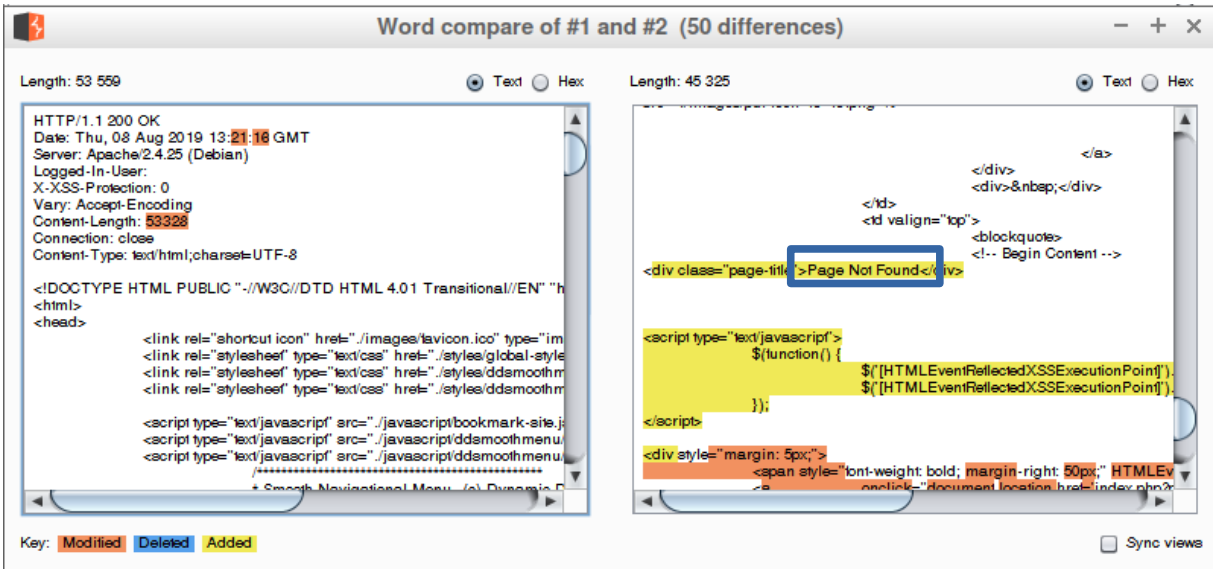


Faire ensuite un clic droit au milieu de la réponse obtenue (partie droite) et cliquer sur **Send to Comparer**.

Reproduire ensuite toutes les étapes précédentes avec une page inexistante sur le serveur comme ***home777.php***. Envoyer la réponse obtenue par le serveur au comparateur (*Send to Comparer*). Vous devriez obtenir ceci au niveau de l'onglet Comparer de *BurpSuite*.



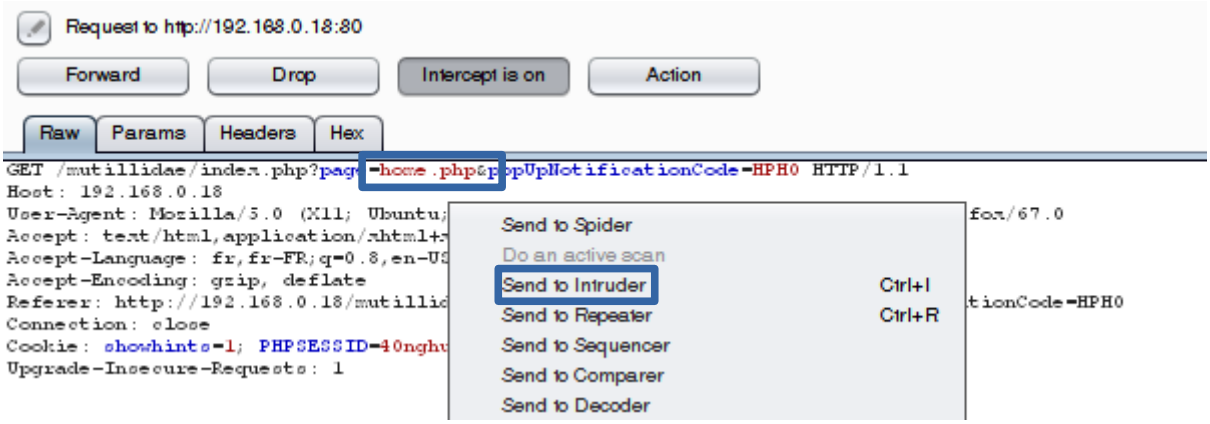
Cliquer ensuite sur le bouton **Words** en bas du comparateur et relever l'affichage de la phrase faisant référence à l'erreur 404 (*Page Not Found*).



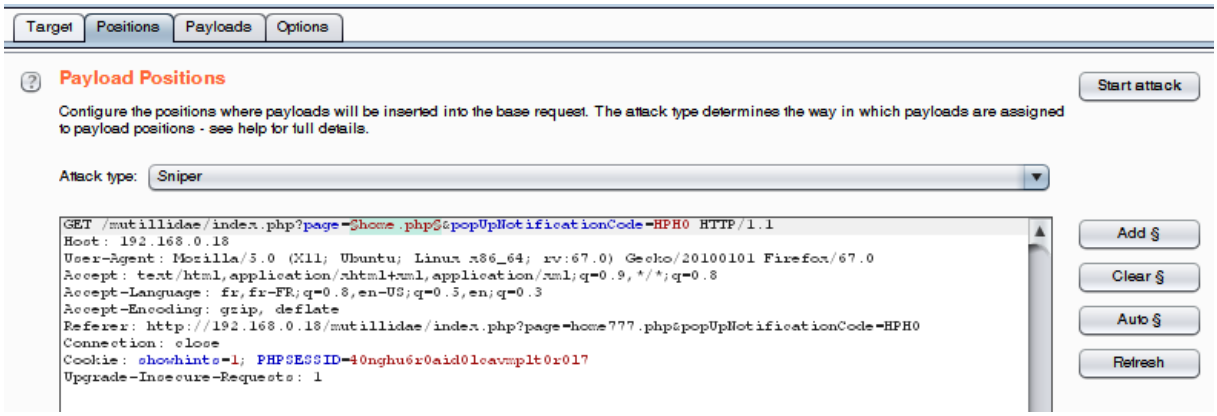
C'est sur cette chaîne de caractère que nous allons nous appuyer pour scanner les pages du site afin de détecter, via **un dictionnaire de noms de pages**, si ces pages existent. Le dictionnaire comprendra des noms de pages typiques associées à des configurations (*.htaccess*, *.htpasswd*…). Il s'agit donc d'une attaque par dictionnaire. Cette technique s'appelle le ***fuzzing*** de pages *Web*.

Étape n°2 : Réalisation de l'attaque

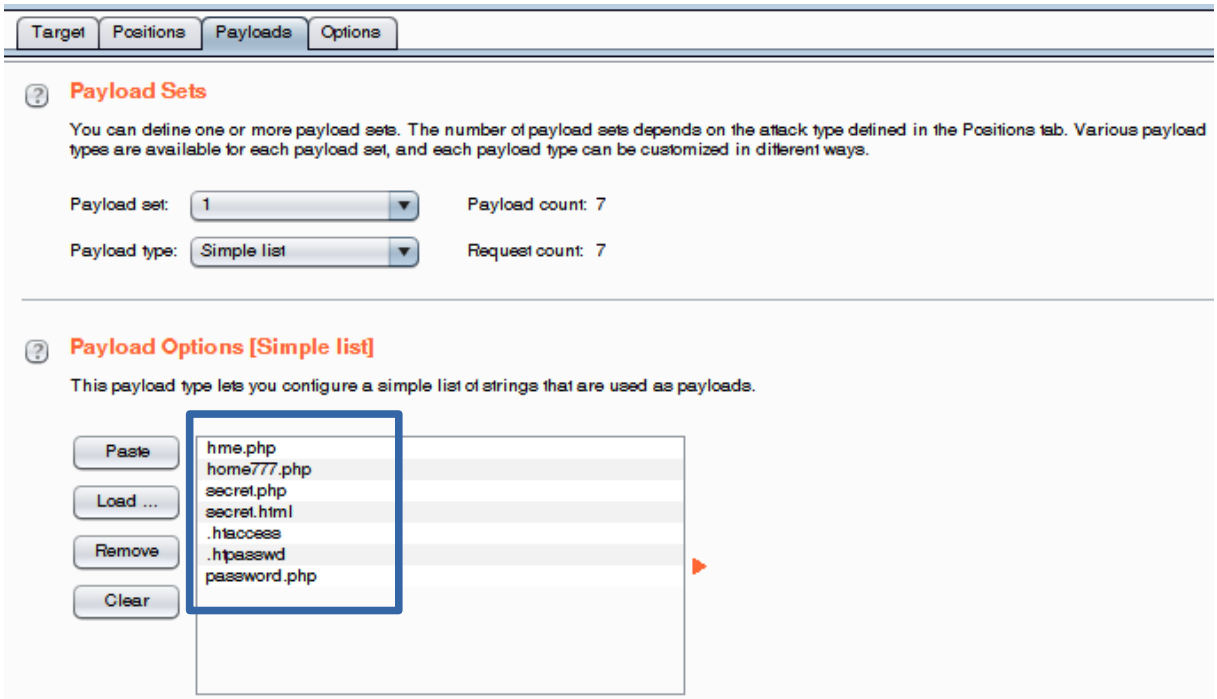
Revenir à la page d'accueil ***home.php*** avec *Burpsuite* en mode interception (*Intercept is on*) et faire un clic droit au milieu de la réponse obtenue et cliquer sur ***Send to Intruder***.



Se rendre dans l'onglet ***Intruder***. Dans l'onglet ***Positions***, cliquer sur le bouton ***Clear*** puis double cliquer sur le nom de la page ***home.php*** puis cliquer ensuite sur le bouton ***Add*** afin que le nom de la page serve de paramètre au *fuzzing*. Le type d'attaque reste ***Sniper*** par défaut.

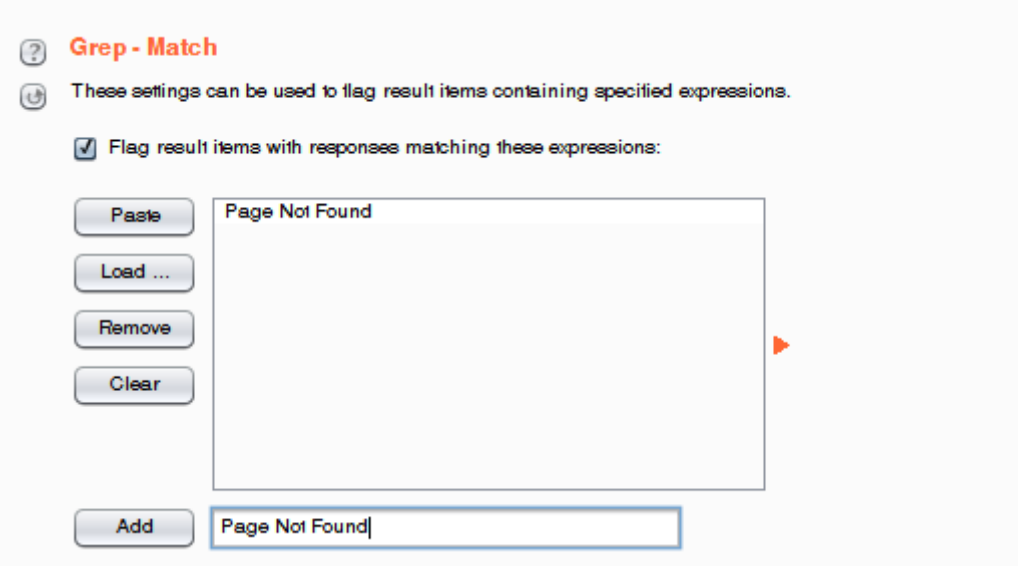


Dans l'onglet ***Payloads***, dans la rubrique ***Payload Options***, ajouter la référence à un dictionnaire contenant des noms de fichiers. Il faut au préalable créer ce dictionnaire à l'aide d'un éditeur de texte. Le test peut se réaliser avec le dictionnaire suivant :

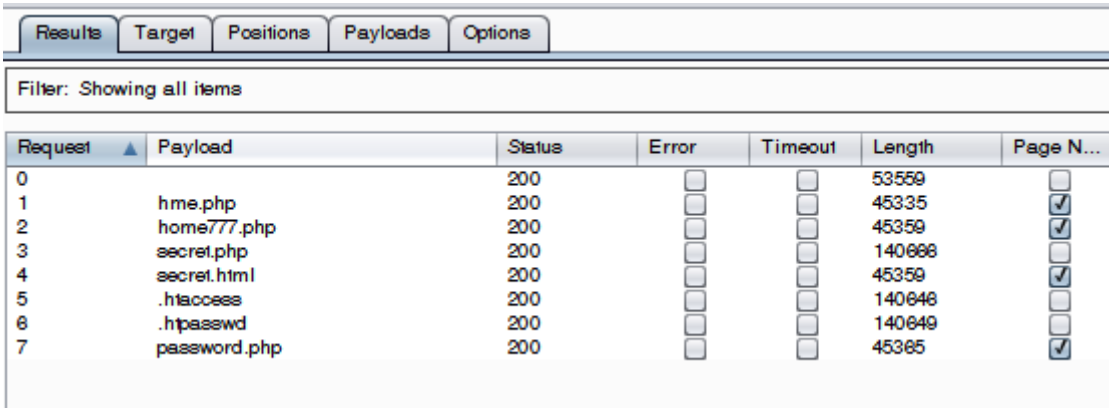


Ensuite, dans l'onglet ***Options***, dans la rubrique *Grep Match*, il faut ajouter un filtre associé à la chaîne de caractère précédemment découverte qui est ***Page Not Found***.

Pour cela, commencer par vider le contenu de cette rubrique en cliquant sur le bouton ***Clear*** puis ajouter la chaîne ***Page Not Found*** dans la zone de texte prévue à cet effet.

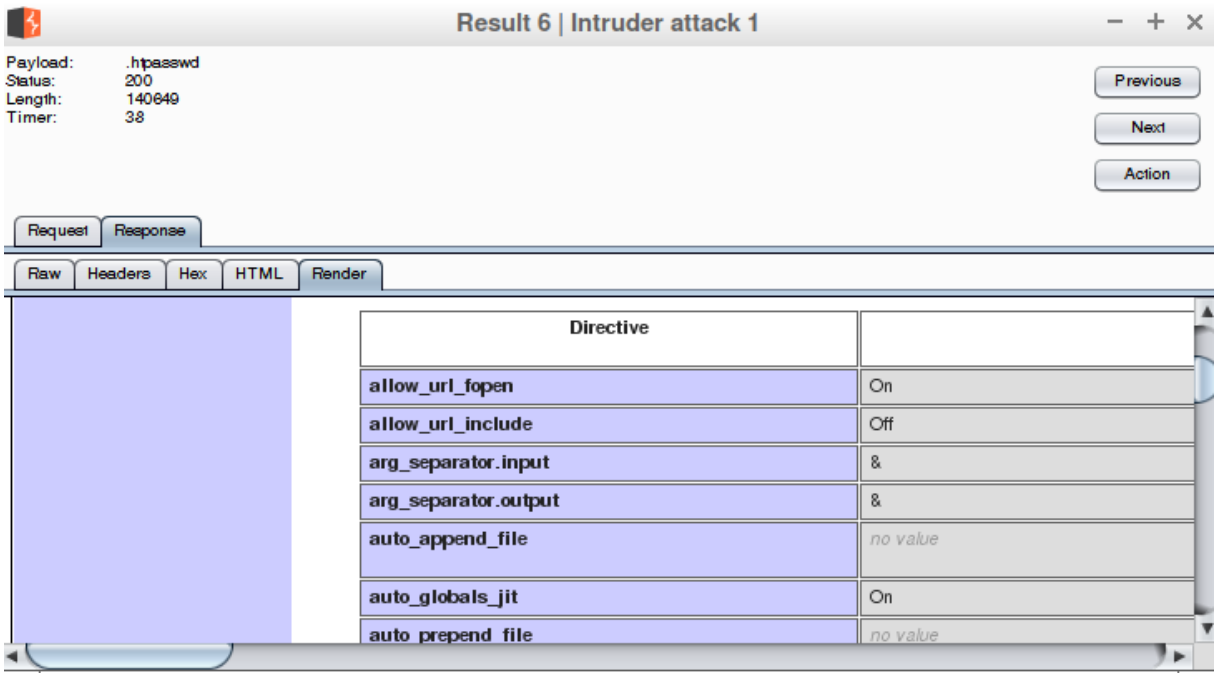


Il ne manque plus qu'à lancer l'attaque en cliquant sur le bouton ***Start Attack***. Toutes les pages non cochées correspondent à des pages existantes sur le serveur.



Request	Payload	Status	Error	Timeout	Length	Page N...
0		200	☐	☐	53559	☐
1	hme.php	200	☐	☐	45335	☑
2	home777.php	200	☐	☐	45359	☑
3	secret.php	200	☐	☐	140668	☐
4	secret.html	200	☐	☐	45359	☑
5	.htaccess	200	☐	☐	140646	☐
6	.htpasswd	200	☐	☐	140649	☐
7	password.php	200	☐	☐	45365	☑

Double cliquer ensuite sur la ligne associée à la page *.htpasswd*. Se rendre ensuite dans l'onglet *Response* puis dans le sous-onglet *Render*.



On peut alors visualiser la page ***phpinfo*** qui donne de précieuses informations sur la configuration du serveur. Si cette page est accessible via le mode sécurité 0 (***OWASP 2017 => A3 : Sensitive Data Exposure = > Information Disclosure => PHP Info Page***), il n'en sera pas de même avec le niveau de sécurité 5.

Dossier 2 : Attaque *SSLSTRIP* via un positionnement *MITM*

Étape n°1 : Préparation de l'environnement de travail

Commencer par activer le **module *SSL*** ainsi que le ***virtualhost HTTPS*** associé à l'application *Mutillidae*. Pour cela, notre maquette de travail comprendre trois machines au sein d'un même réseau, le serveur *Mutillidae*, la victime et l'attaquant.

Travail sur la machine hébergeant l'application *Mutillidae* :

```
#a2enmod ssl
```

```
#a2ensite default-ssl.conf
```

Redémarrer ensuite le serveur *Apache*.

```
#service apache2 restart
```

Travail sur la machine victime :

Il faut ensuite démarrer une nouvelle machine du contexte qui fera office de machine victime. Cette machine doit avoir accès à l'application *Web Mutillidae* via son navigateur. Rien n'est à installer sur cette machine, elle doit simplement disposer d'un navigateur.

Travail sur la machine attaquante :

Il s'agit de la machine précédemment utilisée sur laquelle est installé *BurpSuite* (non utilisé dans ce défi). Nous allons enrichir cette machine attaquante de deux nouveaux outils :

- *arpspoof* pour réaliser le positionnement *MITM* (*Man In The Middle*) via un empoisonnement de cache *arp* ;
- *sslstrip* pour tromper la redirection *HTTPS* afin de capturer les identifiants de connexion.

Ces outils s'installent avec la commande suivante.

```
#apt install dsniff sslstrip
```

Il faut ensuite activer le routage (flux traversants) sur notre machine attaquante. C'est en effet depuis cette machine que les flux de la victime transiteront. Pour cela, ouvrir le fichier */etc/sysctl.conf* et supprimer le commentaire sur la ligne suivante :

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Enfin, il faut programmer une redirection vers le serveur ***sslstrip*** sur les flux capturés :

```
#iptables -t nat -A PREROUTING -p tcp –destination-port 80 -j REDIRECT –to-port 7777
```

```
iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port
 8080
```

Attention, pour que l'effet de redirection de cette commande soit persistant, il serait préférable de l'intégrer dans un script lancé automatiquement au démarrage.

Étape n°2 : Réalisation de l'attaque

Travail sur la machine attaquante :
Après avoir relevé l'adresse IP de la victime, ouvrir un terminal et lancer la commande d'empoisonnement suivante :

#arpspoof -i enps03 -t <ip-victime> <ip-serveur-mutillidae>

Il faut remplacer la valeur *enp0s3* par le label de l'interface utilisée par la machine attaquante (*eth0*, *eth1*, *enp0s8*…). Dans la capture d'écran ci-dessous la victime a pour adresse IP 192.168.0.24 et le serveur a pour adresse IP 192.168.0.18.

Le label de l'interface peut se trouver à l'aide des commandes suivantes : **ifconfig**, **ip a** ou **lshw -c network**.

```
root@prof:~/Téléchargements# arpspoof -i enp0s3 -t 192.168.0.24 192.168.0.18
8:0:27:13:2a:69 8:0:27:cb:70:1b 0806 42: arp reply 192.168.0.18 is-at 8:0:27:13:2a:69
8:0:27:13:2a:69 8:0:27:cb:70:1b 0806 42: arp reply 192.168.0.18 is-at 8:0:27:13:2a:69
8:0:27:13:2a:69 8:0:27:cb:70:1b 0806 42: arp reply 192.168.0.18 is-at 8:0:27:13:2a:69
8:0:27:13:2a:69 8:0:27:cb:70:1b 0806 42: arp reply 192.168.0.18 is-at 8:0:27:13:2a:69
8:0:27:13:2a:69 8:0:27:cb:70:1b 0806 42: arp reply 192.168.0.18 is-at 8:0:27:13:2a:69
8:0:27:13:2a:69 8:0:27:cb:70:1b 0806 42: arp reply 192.168.0.18 is-at 8:0:27:13:2a:69
```

Ouvrir ensuite un deuxième terminal afin de préparer le serveur *sslstrip* en écoute sur le port choisi 7777.

#sslstrip -l 7777

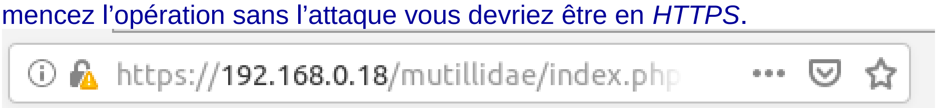
```
root@prof:~/Téléchargements# sslstrip -l 7777

sslstrip 0.9 by Moxie Marlinspike running...
```

Travail sur la machine de la victime:
Ouvrir le navigateur et se connecter à l'application *Mutillidae* en *HTTP*.

http://<ip-serveur>/mutillidae

Ensuite, cliquer sur le bouton *Enforce SSL* situé en haut de la page.

Ensuite, se rendre sur la page d'authentification en cliquant sur ***Login/register***. Normalement, une redirection *HTTPS* est mise en place mais celle-ci est inopérante en raison de l'attaque *SSLSTRIP*. Si vous recommencez l'opération sans l'attaque vous devriez être en *HTTPS*.



Tenter une authentification quelconque en saisissant un login et un mot de passe. Comme la connexion est en *HTTP*, l'attaquant peut capturer les identifiants.

Travail sur la machine de l'attaquant:
Depuis le répertoire où est exécuté le serveur *SSLSTRIP*, ouvrir le fichier *sslstrip.log*. Celui doit contenir tous les identifiants capturés lors de l'attaque.

```
prof@prof:~/Téléchargements$ cat sslstrip.log
2019-08-08 19:02:14,451 SECURE POST Data (192.168.0.18):
username=test&password=ok&login-php-submit-button=Login
```