

**TP – Sécurité système**

**ENSEIGNANTE :** Margot PRIEM

Table des matières

[1 Installation de la machine virtuelle 4](#_Toc506298643)

[2 Installation d’un serveur web 5](#_Toc506298644)

[2.1 Installation de Apache 5](#_Toc506298645)

[2.2 Déploiement de l’application 5](#_Toc506298646)

[2.2.1 Déploiement des sources 5](#_Toc506298647)

[2.2.2 Installation de PHP 5](#_Toc506298648)

[2.2.3 Installation de MySQL 5](#_Toc506298649)

[2.3 Analyse de flux 8](#_Toc506298650)

[2.3.1 Analyse flux non sécurisé 8](#_Toc506298651)

[2.3.2 Installation d’un certificat 10](#_Toc506298652)

[2.3.3 Analyse flux sécurisé 12](#_Toc506298653)

[3 Installation serveur ssh 13](#_Toc506298654)

[3.1 Installation de openssh 13](#_Toc506298655)

[3.2 Configuration 13](#_Toc506298656)

[3.2.1 Les fichiers de configuration 13](#_Toc506298657)

[4 Installation d’un WAF 16](#_Toc506298658)

[4.1 Installation 16](#_Toc506298659)

[4.2 Configuration 16](#_Toc506298660)

[4.2.1 Fichier de configuration 16](#_Toc506298661)

[4.2.2 Règles 17](#_Toc506298662)

[4.2.3 Création de nouvelles règles 18](#_Toc506298663)

[4.2.4 Brute-Force 19](#_Toc506298664)

[4.2.5 Upload de fichier 20](#_Toc506298665)

[4.2.6 Scan Nmap 20](#_Toc506298666)

[4.2.7 Signature serveur 21](#_Toc506298667)

[5 Outils d’audit 22](#_Toc506298668)

[5.1 Installation 22](#_Toc506298669)

[5.2 Exécution 22](#_Toc506298670)

1. Introduction

L’objectif de ce TP est de vous introduire les fondamentaux de quelques bons procédés système utilisés en sécurité.

Dans un premier temps, vous attacherez à déployer un serveur web avec les sources d’une application vulnérable, vous en profiterez pour capturer et analyser le flux réseau entre un client et le serveur. Ensuite, vous vous intéresserez de plus près au serveur SSH afin de configurer à distance votre serveur et à la mise en place d’un WAF pour protéger votre applicatif. Enfin, vous découvrirez un script qui vous permet d’auditer le niveau de sécurité de votre serveur web.

Si jamais vous venez à bloquer sur une des parties alors passez à la suivante puisqu’elles sont indépendantes.

**Rendu :** Vous répondrez à chacune des questions directement dans le document présent. A noter que pour chacune des questions présentes dans ce TP, **vous vous attacherez à donner une explication et à fournir une capture d’écran de vos résultats.**

Le TP est **à renvoyer en fin de séance, ou au plus tard en fin de journée.** Si tel n’est pas le cas, je considèrerai le document comme non rendu. Vous l’enverrez par mail à [margot.priem@fr.clara.net](mailto:margot.priem@fr.clara.net) avec le sujet suivant : [DA2I][TP-SEC-SYS][NOM PRENOM]

Merci et bon courage à vous !

1. Installation de la machine virtuelle

Dans un premier temps, veiller à installer la machine virtuelle suivante : ubuntu-14.04.5-desktop-amd64.iso disponible à cette adresse : <http://releases.ubuntu.com/14.04/>

**NB :** Veillez à ne pas installer les mises à jour lors de l’installation. Déroulez la procédure d’installation par défaut.

1. Installation d’un serveur web

Installation de Apache

Installer le serveur

sudo apt-get install apache2

Lancer le serveur apache

sudo service apache2 start

Vérifier que le serveur web est bien lancé

[http://127.0.0.1](http://127.0.0.1/)

Déploiement de l’application

Déploiement des sources

Nous allons nous appuyer sur une application vulnérable fournie par l’OWASP

Télécharger les sources ici :<https://sourceforge.net/projects/mutillidae/files/mutillidae-project/LATEST-mutillidae-2.6.55.zip/download>

Copier les sources dans le dossier web

sudo cp -R ~/Downloads/mutillidae/ /var/www/html/

Installation de PHP

sudo apt-get install python-software-properties

sudo add-apt-repository ppa:ondrej/php

sudo apt-get update

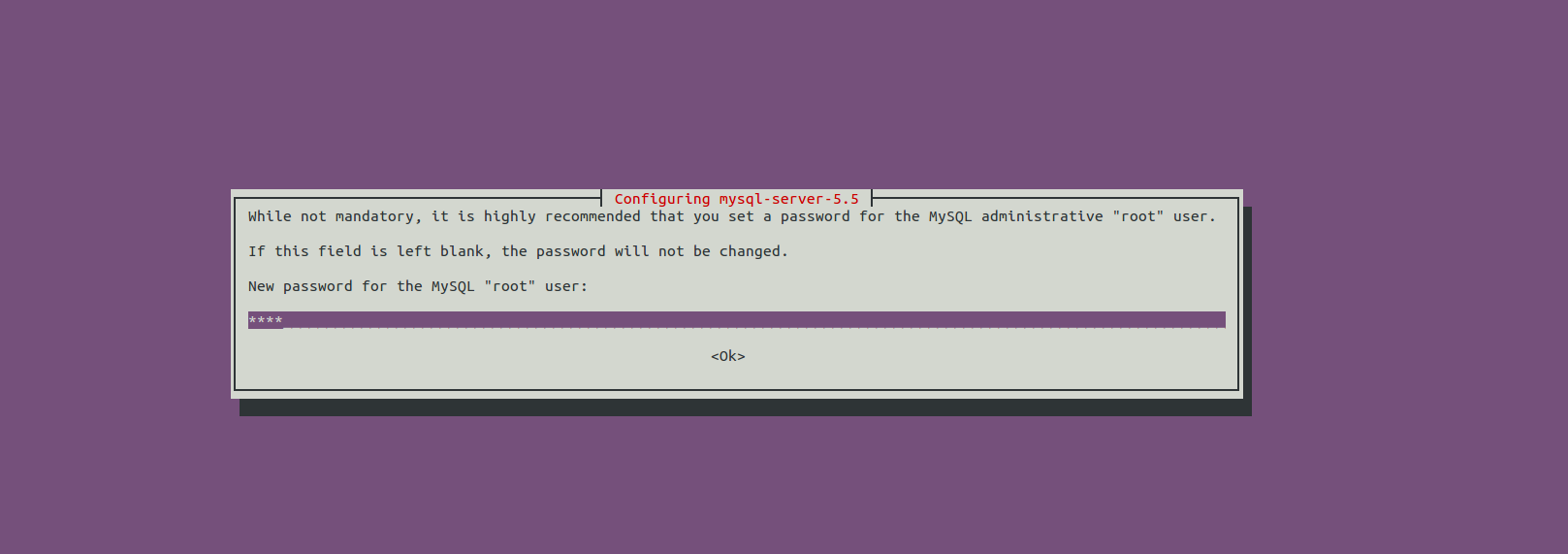
sudo apt-get install php5.6 php5.6-mbstring php5.6-xml php5.6-mysql

php5.6 -v

Installation de MySQL

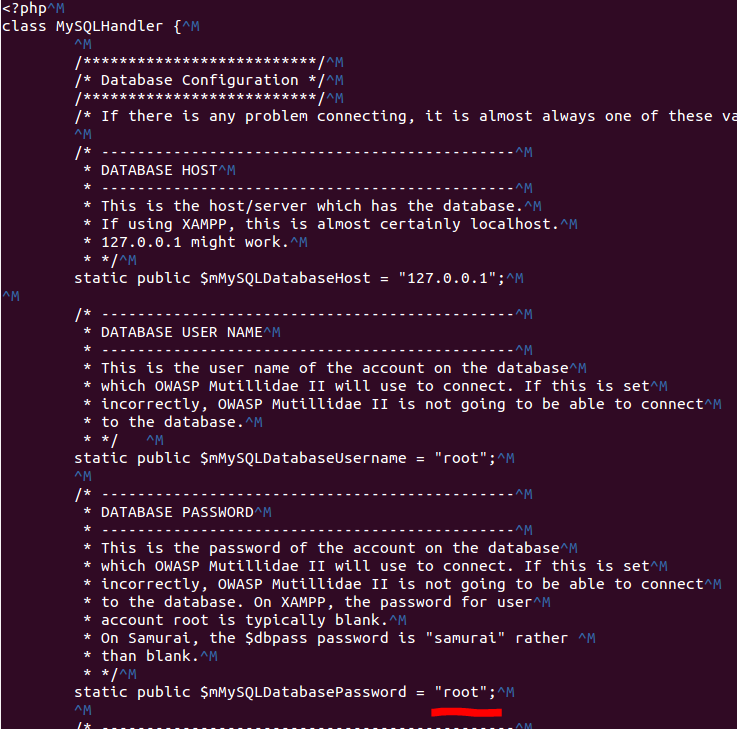
sudo apt-get install mysql-server libapache2-mod-auth-mysql  
  
sudo service mysql start

Choisissez un mot de passe pour l’utilisateur root du serveur mysql

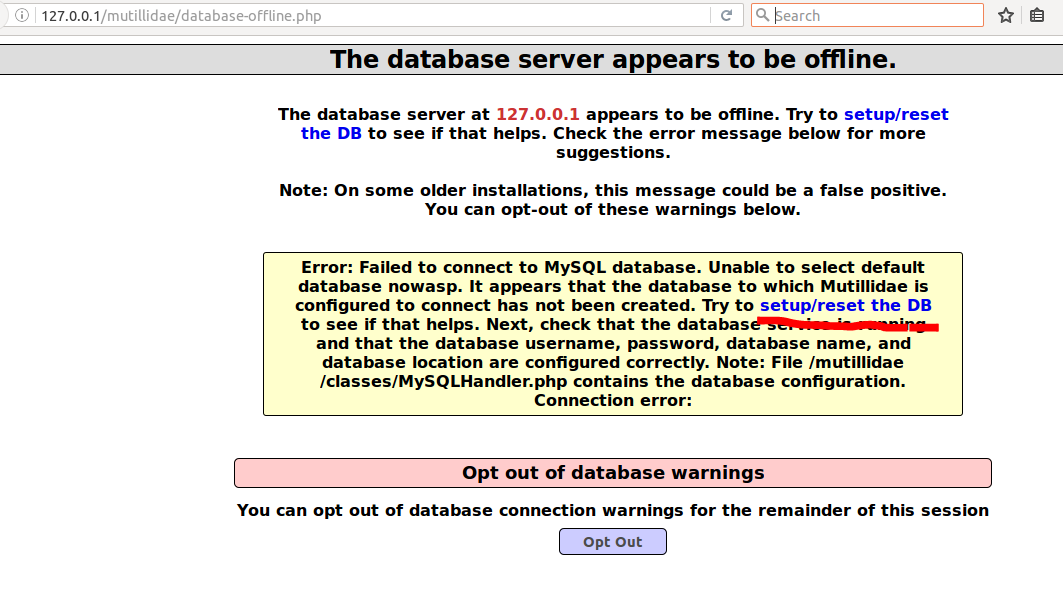


Pensez à ajouter le mot de passe à votre fichier de configuration de base de données

sudo vi /var/www/html/mutillidae/classes/MySQLHandler.php

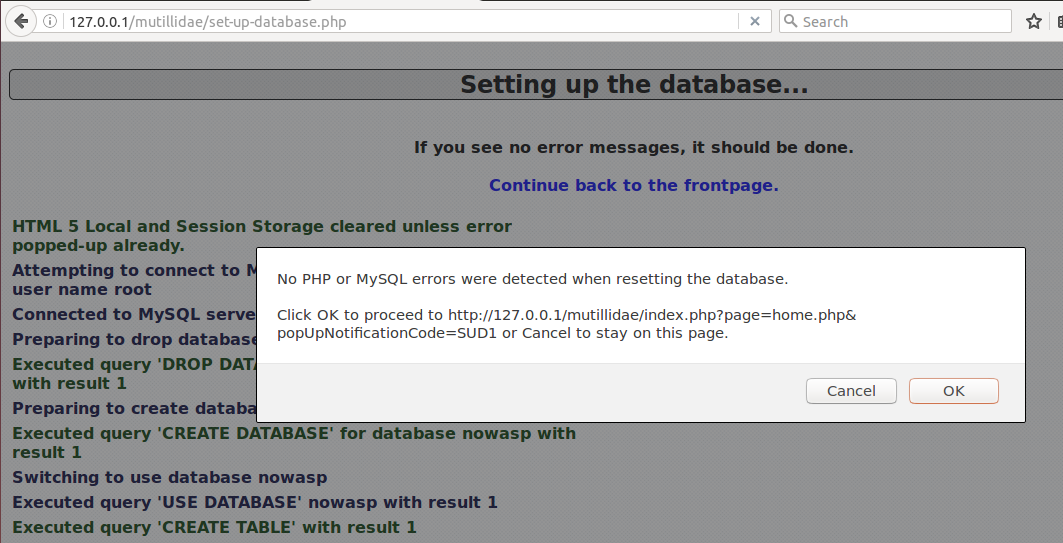


Normalement, si vous accédez de nouveau à l’interface de l’application via votre navigateur vous devriez avoir le message suivant :



Cliquer sur « setup/reset the DB” et validez.

Vous avez maintenant accès à l’application



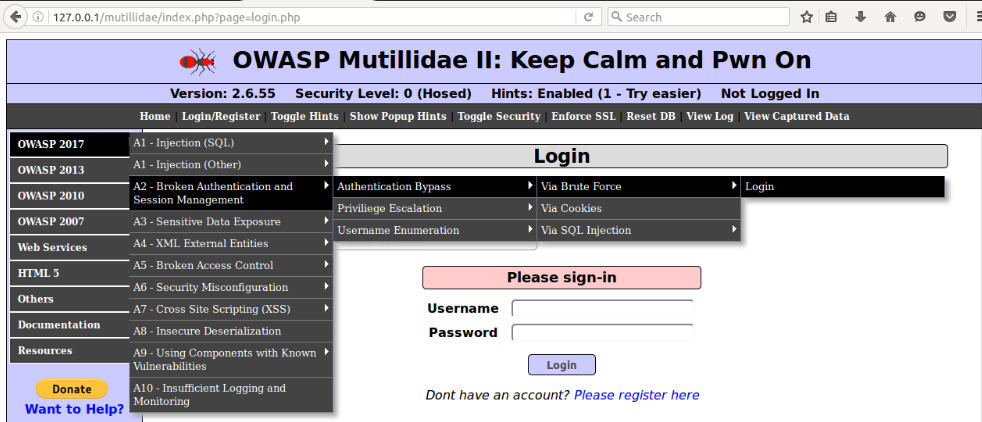


**NB:** Si vous rencontrez des erreurs lors de la procédure d’installation, pensez à regarder dans les logs d’erreurs du serveur apache.

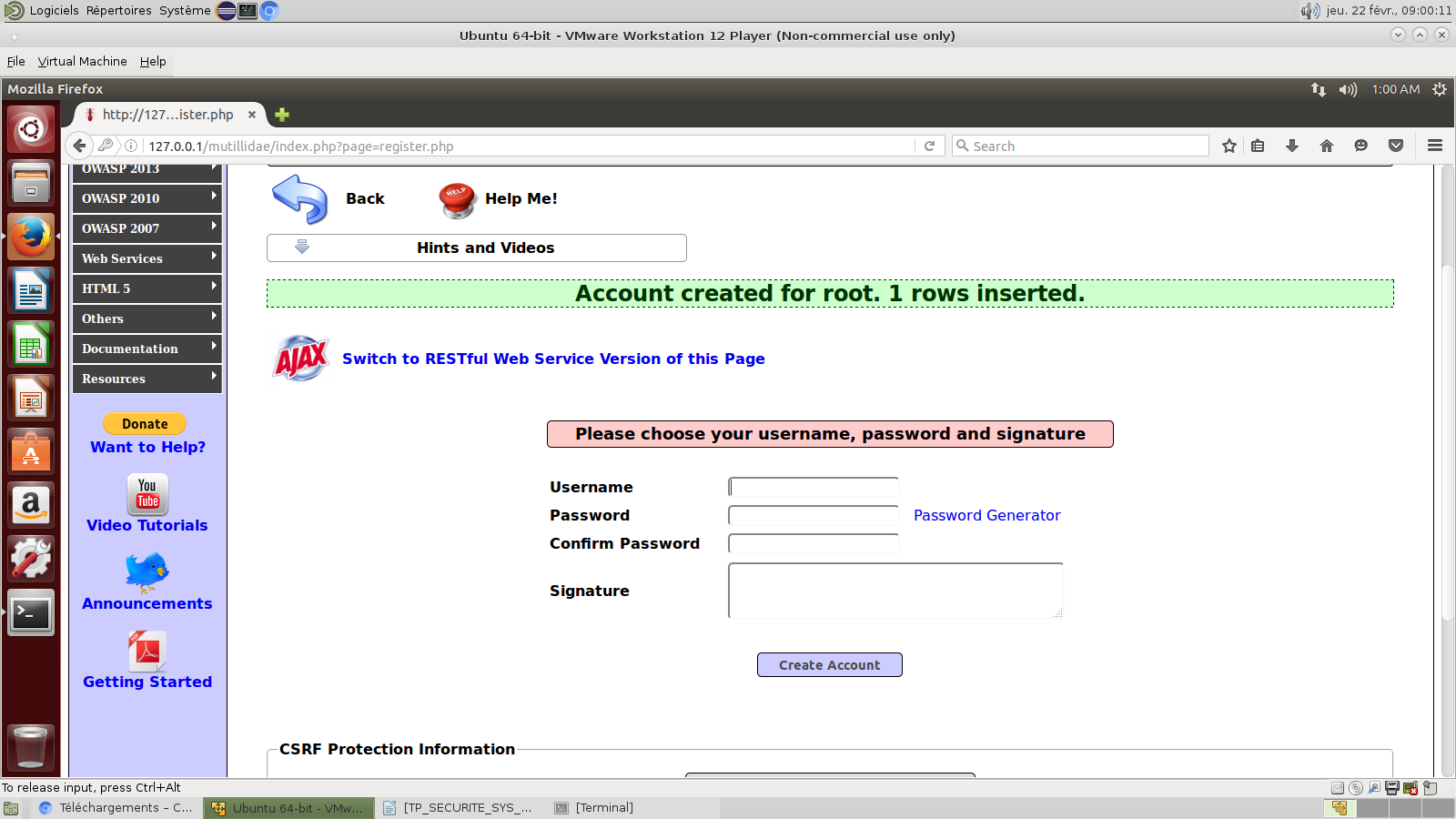
tail -f /var/log/apache2/error.log

Analyse de flux

Au sein de l’application précédemment installée, rendez-vous à l’exercice « Authentification Bypass »



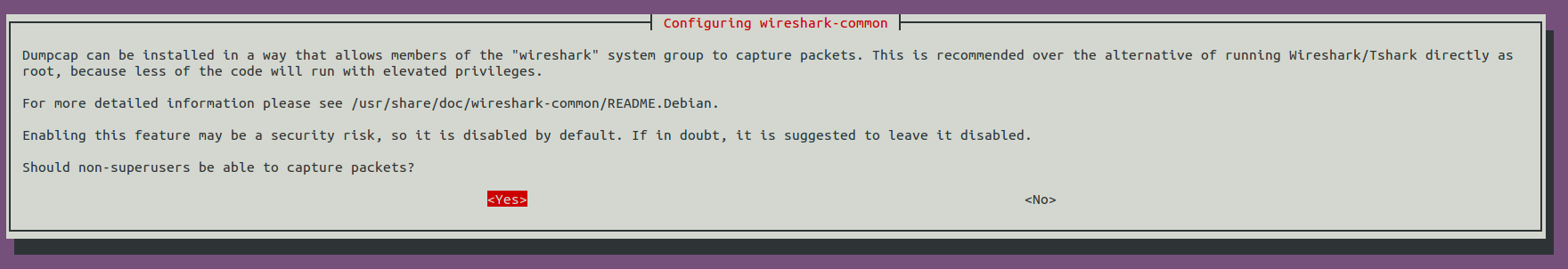
Créez vous un utilisateur qui vous servira par la suite (voir formulaire Register).



Analyse flux non sécurisé

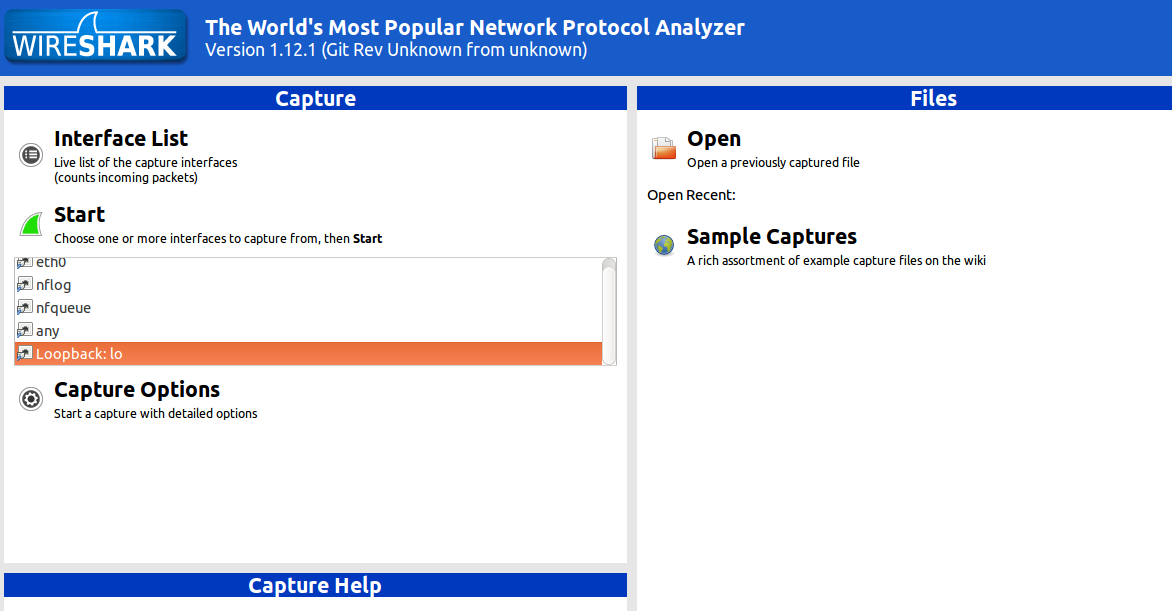
Installez wireshark

sudo apt-get install wireshark



Une fois l’installation finie, lancer wireshark. Attention à bien lancer l’outil avec les droits « root » sinon vous n’aurez pas accès à vos interfaces réseaux.

sudo wireshark &



Interface Wireshark

Choisissez votre interface, dans votre cas : loopback. Puis cliquez sur Start.   
En parallèle, retournez sur votre application et authentifiez-vous avec votre utilisateur fraichement créé. Une fois cela fait, stoppez la capture sur wireshark.

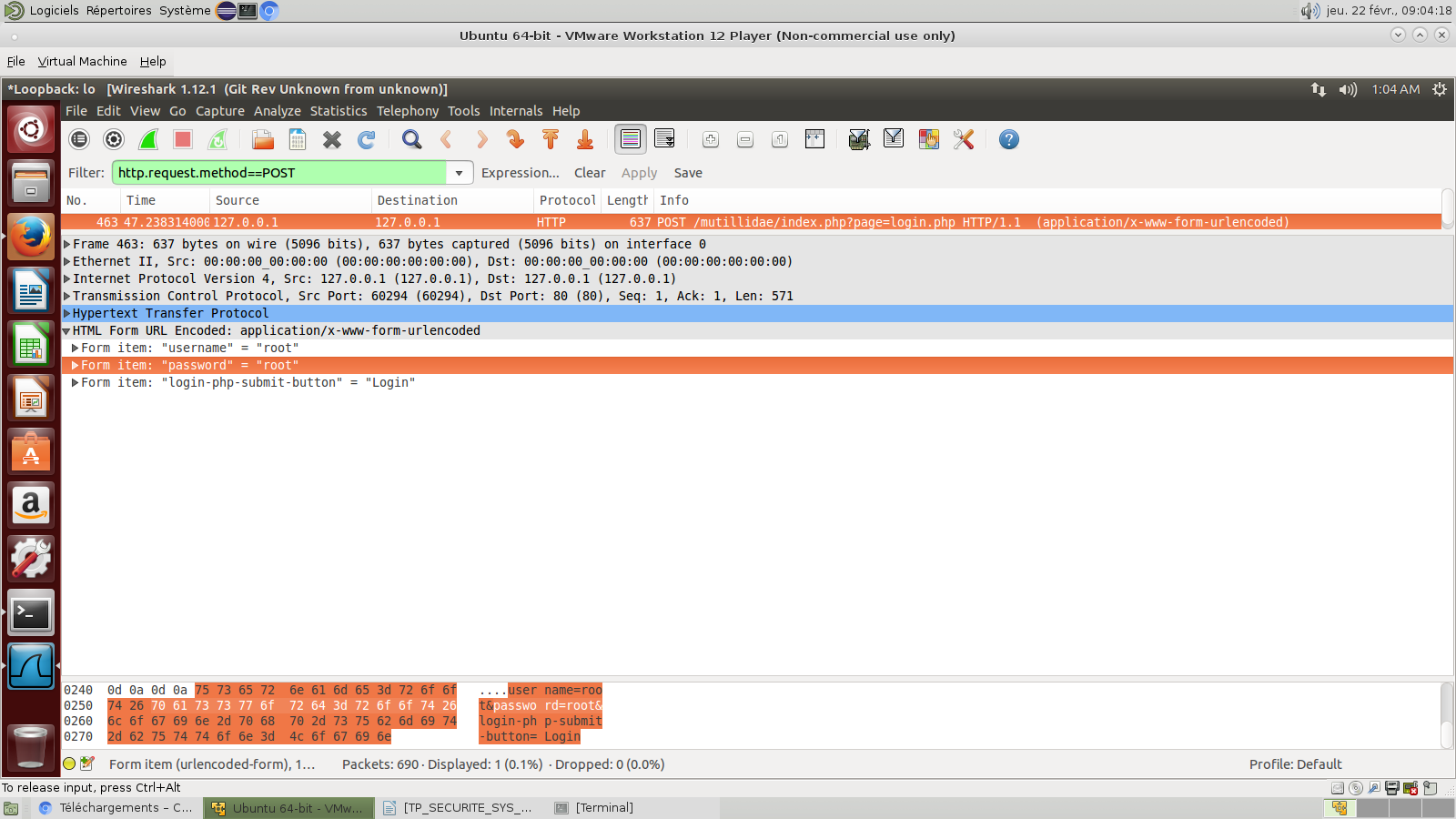
Analysez votre capture et filtrer sur les requêtes POST

Filtre wireshark : http.request.method==POST



Filtrage sur les requêtes POST

**1. Analyser votre ou vos requêtes. Qu’observez-vous ?**

On peux voir les mots de passes et nom utilisateur

**2. Quels sont les risques ?**

Les mots de passes sont disponible à n'importe qui

**3. Quel moyen pouvons-nous mettre en œuvre pour se prémunir ?**

Cryptage Ssl avec Protocole Https

Installation d’un certificat

Génération du certificat

On crée la clé privée avec l'algorithme RSA 2048 bits.

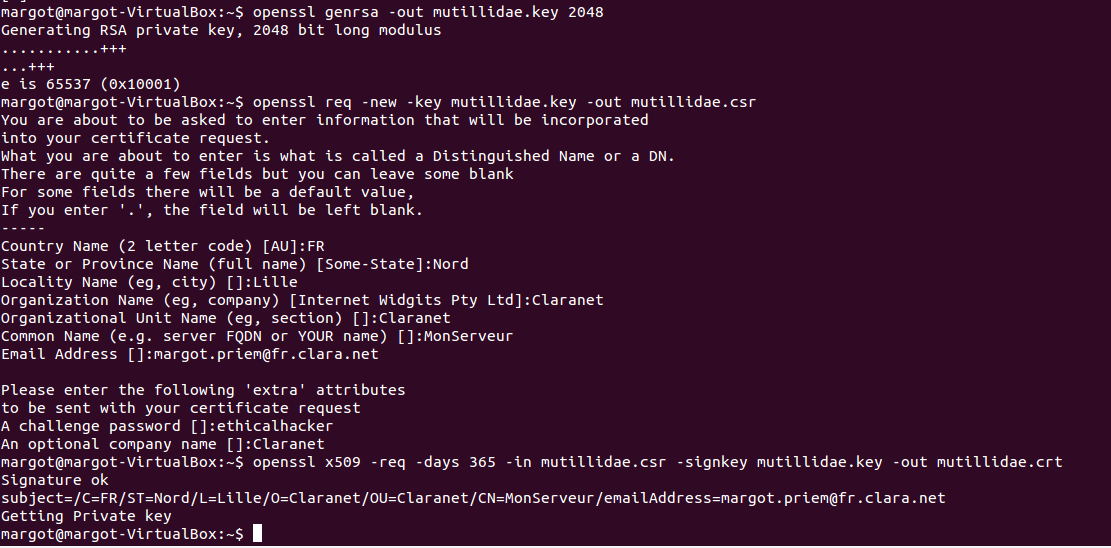
openssl genrsa -out mutillidae.key 2048

Ensuite il faut générer un fichier de « demande de signature de certificat », en anglais CSR.

openssl req -new -key mutillidae.key -out mutillidae.csr

Répondre aux différentes questions. Ensuite, on génère le certificat signé au format x509 (ici pour 365jours auto-signé)

openssl x509 -req -days 365 -in mutillidae.csr -signkey mutillidae.key -out mutillidae.crt

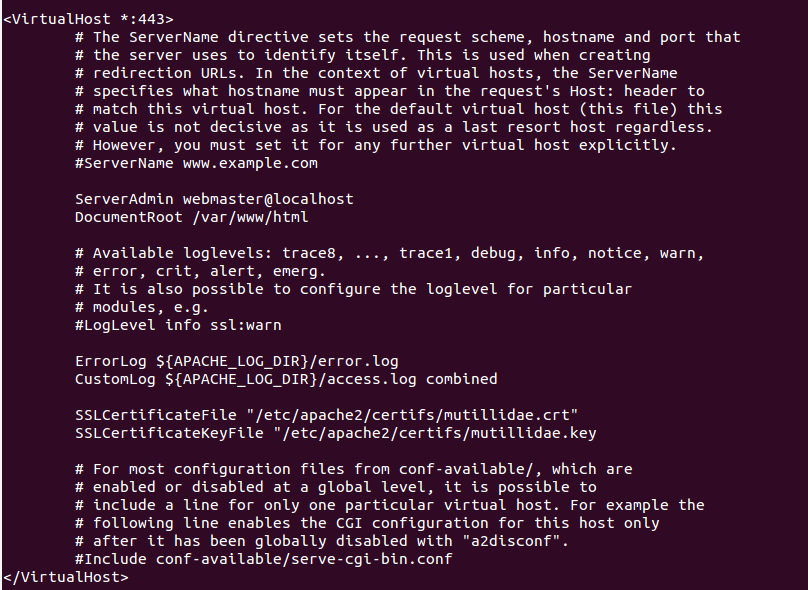


**ATTENTION :** Ce certificat n'est authentifié par aucune autorité, vous aurez donc un message d'avertissement quand vous vous connectez au serveur. On peut ajouter le fichier mutillidae dans la liste des certificats navigateurs afin de ne pas devoir accepter le certificat à chaque fois.

Installation du certificat

sudo vi /etc/apache2/sites-available/000-default.conf

Modifier votre configuration en ajoutant un nouveau vhost sur le port 443

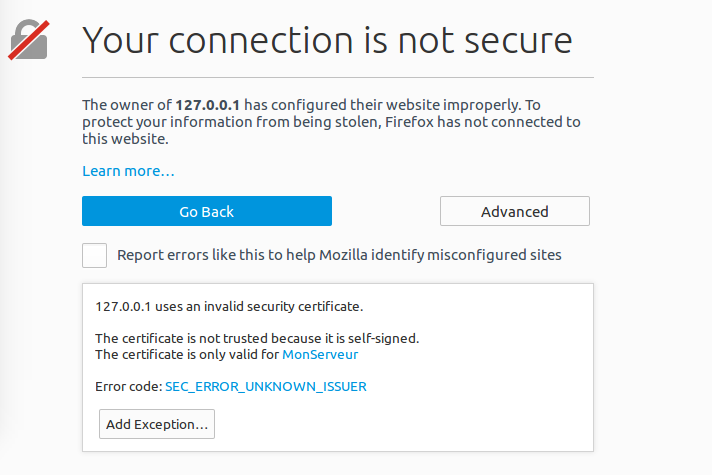


Il faut maintenant activer le module ssl d’Apache pour que cela fonctionne

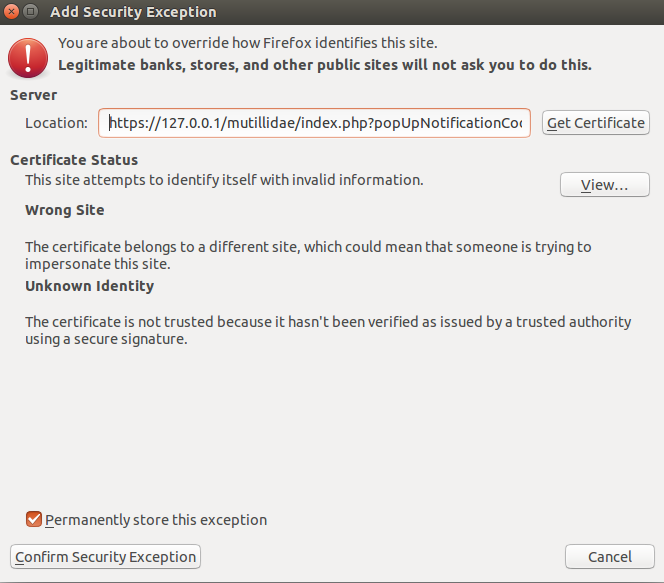
sudo a2enmod ssl

sudo service apache restart

Vous devez maintenant pouvoir accéder à votre site en https.



Alerte certificat autosigné



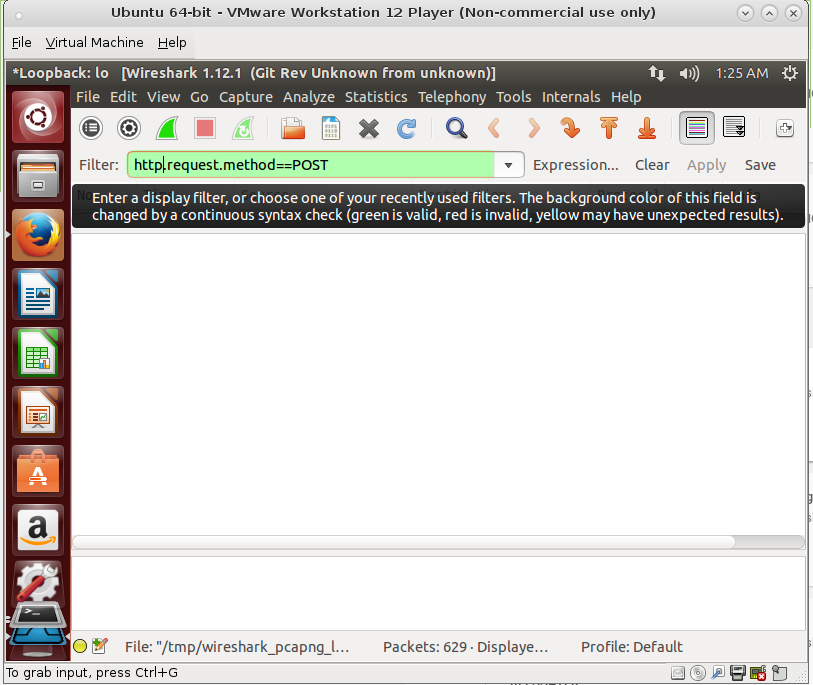
Ajouter une exception

Analyse flux sécurisé

**1. Maintenant réitérer les actions précédemment menées avec wireshark**

**2. Qu’observez-vous ?**

On ne retrouve plus la requêtes véhiculant les informations



**3. Cela vous semble-t-il suffisant ?**

Sans doute pas

1. Installation serveur ssh

Installation de openssh

Installez le paquet openssh-server sur votre poste.

sudo apt-get install openssh-server

Configuration

Il existe une multitude d’options SSH aussi bien pour le serveur que pour les clients. Ces options sont soit stockées dans des fichiers de configuration ou fournies sur la ligne de commande.

Les fichiers de configuration

* /etc/ssh/sshd\_config : Fichier de configuration du serveur OpenSSH.
* /etc/ssh/ssh\_config : Fichier de configuration du client OpenSSH.
* ~/.ssh/ : Dossier de configuration SSH de l’utilisateur
* ~/.ssh/authorized\_keys ou ~/.ssh/authorized\_keys : Liste les clés publiques (RSA / DSA) du l’utilisateur.

Le port par défaut SSH : TCP 22. A partir de cet accès, vous pouvez saisir n’importe quelle commande. Si l’accès est root, vous avez un accès total au système.

Cas pratique 1

Allez voir le contenu de sshd config et ssh config et plus particulièrement des paramètres suivants.

– Port 22

– PermitRootLogin yes

– X11Forwarding yes

**1. A quoi sert chacun de ces paramètres ?**

Le port 22 est le port par défaut de ssh

Le permitroologin en yes autorise à être root à distance

X11Forwarding permet de capturer l'interface graphique

**NB :** pour plus d’infos, faites un man sshd config ou un man ssh config

Cas pratique 2

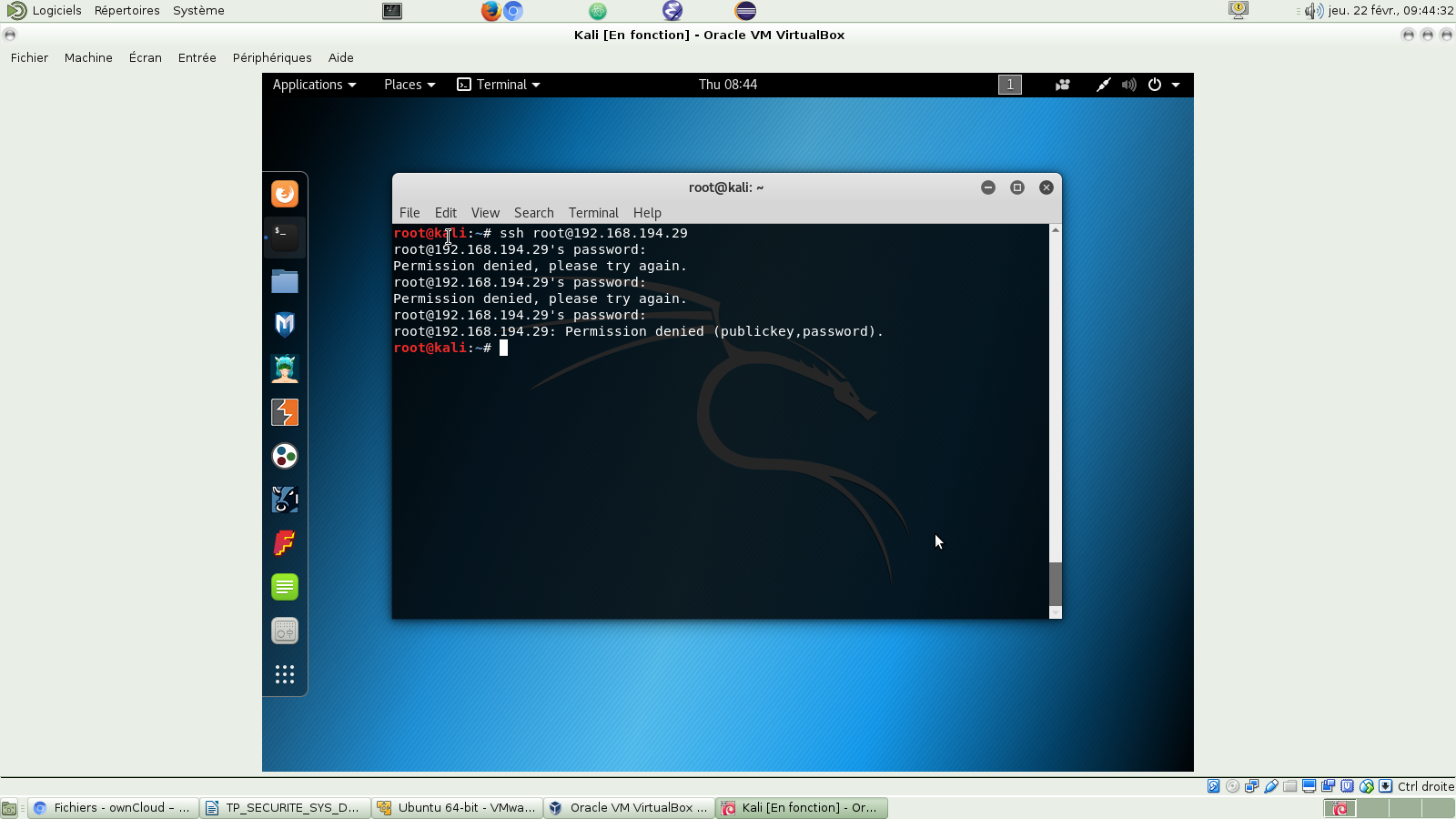
Affectez un mot de passe au root du serveur avec la commande :

sudo passwd root

Connectez-vous en tant que root sur le serveur depuis votre kali. Modifiez le fichier de configuration de votre serveur de telle sorte que la connexion SSH via votre compte root soit interdite. Réessayez de vous loguer en root.

**1. Quel paramètre avez-vous modifiez ? Vous copierez et collerez une capture de la réponse serveur.**

**Le PermitRootLogin**

****

Cas pratique 3

**1. En une seule commande (c’est-à-dire sans connexion SSH préalable), récupérez la liste des processus de l’ordinateur qui tourne sur la machine serveur. Indiquez votre commande.**

ssh [demoraen@192.168.194.29](mailto:demoraen@192.168.194.29) 'ps -elf'

**2. Une machine distante peut lancer une application graphique. Connectez-vous au serveur et lancer firefox. Qu’observez-vous ?**

ssh -X margot@10.0.2.6

**Firefox ne s'ouvre pas**

**3. Quelle option de configuration du serveur SSH vous permet d’autoriser ou d’interdire l’envoi de données de votre serveur graphique (un indice : sous Linux, le serveur graphique est souvent appelé serveur X).**

**X11Forwarding**

**4. Modifiez cette option pour autoriser l’utilisation de votre serveur graphique à distance et lancez firefox depuis une machine cliente. Quelle option avez-vous modifié ?**

**X11Forwarding yes**

**5. Que se passe-t-il ?**

Firefox s'ouvre bien

Cas pratique 4

Il est aussi possible de n’autoriser que certains utilisateurs.

**1. Quel paramètre permet cela ? Appliquez la configuration à votre utilisateur.**

AllowUsers

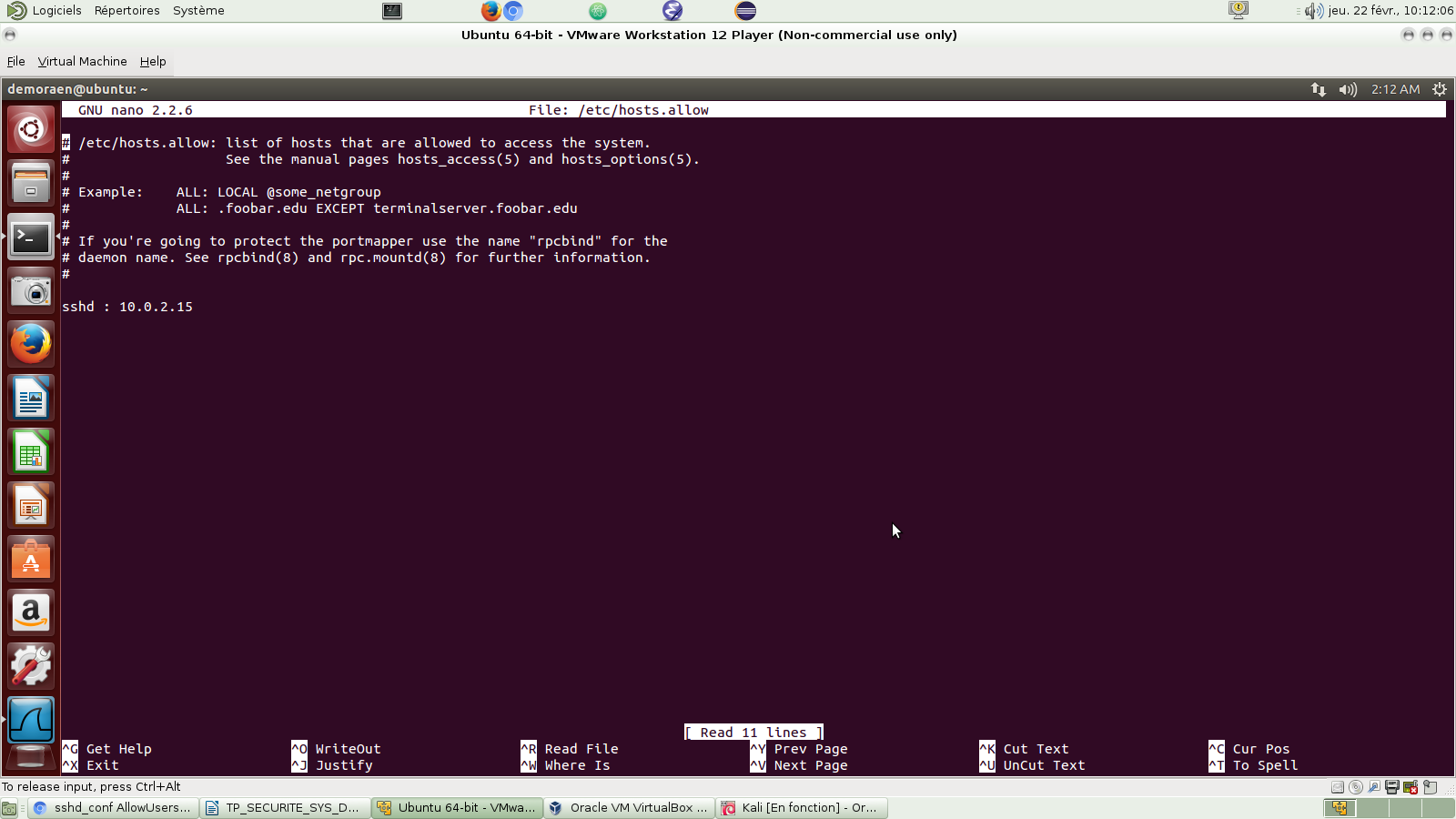
De même vous pouvez interdire certains utilisateurs, par exemple :

DenyUsers www-data margot

Vous pouvez limiter l’accès SSH à certaines machines. Au lieu de bloquer le port par iptables, vous pouvez déclarer les machines autorisées partir des IPs depuis dans le fichier : /etc/hosts.allow

sshd : 192.168.1.2 172.16.23.12

**2. Autorisez uniquement votre kali à pouvoir se connecter en ssh. Vous fournirez un Screenshot.**

****

Cas pratique 5

Le port Knocking est la dissimulation d’un service à travers une séquence de ports prédéfinies. Le service est inaccessible via des règles Iptables. En utilisant, une séquence de port, une règle iptables va être créées qui va rendre le service accessible sur l’IP qui émet cette séquence.

Pour se faire, vous devez installer un daemon de port knocking comme knockd

sudo apt-get install knockd

Sur le serveur, knockd doit tourner en daemon, pour cela, on modifie /etc/default/knockd :

START\_KNOCKD=1

Ensuite on modifie /etc/knockd.conf en créant la séquence de ports qui va permettre de créer la règle.

Par exemple, ci-dessous, il faut envoyer une connexion TCP sur les ports 7000 puis 8000 et enfin 9000 afin que la règle Iptables permettant l’accès soit générée :

(attention, bien mettre -I INPUT pour l’open)

[options]

UseSyslog

[openrsync] sequence = 7000,8000,9000

seq\_timeout = 5

command = /sbin/iptables -I INPUT -s %IP% -p tcp --dport 22 -j ACCEPT

tcpflags = syn

[closerync]

sequence = 9000,8000,7000

seq\_timeout = 5

command = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT

tcpflags = syn

Enfin, on crée la règle pour fermer le port 22

/sbin/iptables -A INPUT -p tcp --dport 22 -j DROP

Côté client, vous pouvez générer la séquence comme ceci :

knock -v host port1 port2 port3

Puis :

ssh user@host

1. Installation d’un WAF

Installation

Modsecurity est disponible dans les repo officiels de Debian et de Ubuntu.

apt-get install libapache2-modsecurity

Pour vérifier que le module a bien été chargé par Apache, exécuter la commande qui suit :

apachectl -M | grep security

Si c’est bien le cas, vous devriez voir le message suivant s’afficher :

security2\_module (shared)

Erreur :

AH00526: Syntax error on line 40 of /usr/share/modsecurity-crs/rules/REQUEST-941-APPLICATION-ATTACK-XSS.conf:

Error parsing actions: Unknown action: \\

J'ai essayé d'upgrade la version d'apache mais ça ne fonctionne toujours pas

Configuration

Fichier de configuration

Lors de l’installation, le module mod-security embarque un fichier de configuration par défaut. Nous devons créer le notre pour appliquer notre propre configuration.

mv /etc/modsecurity/modsecurity.conf-recommended modsecurity.conf

Une fois le nouveau fichier créé, il est nécessaire de faire un reload d’Apache pour que la nouvelle configuration soit effective. Lors de tout changement, il faudra également effectuer un reload du service.

service apache2 reload

Vous devriez trouver un nouveau fichier de log pour mod\_security dans le dossier des logs du serveur apache

margot@kali:/etc/modsecurity# ls /var/log/apache2/  
access.log error.log modsec\_audit.log other\_vhosts\_access.log

Dans la configuration par défaut, mod\_security est configuré pour effectuer uniquement de l’alerting: il ne bloque rien. Afin de le passer en mode bloquant, vous devez éditer votre configuration et modifier la directive **SecRuleEngine.**

vi /etc/modsecurity/modsecurity.conf

Trouvez la ligne

SecRuleEngine DetectionOnly

Et modifiez-la en

SecRuleEngine On

**Attention**: En production, on passe uniquement en mode bloquant quand nous sommes sûrs que les règles ont été correctement appliquées et que la maîtrise de l’applicatif est suffisant.

Pour la suite du TP, il préférable que vous restiez en mode transparent.

Règles

Une fois que mod\_security est installé, il ne fait à vrai dire pas grand-chose. Toute la force de mod\_security réside dans les règles qu'on lui demande d'appliquer aux requêtes et réponses qu'il filtre. Ces règles se nomment CRS (**Core Rule Set**), il existe des exemples qui sont stockés sur le système, mais non utilisés à l'installation de mod\_security. On peut également trouver les dernières versions en ligne sur le site de l'OWASP qui diffuse et met à jour gratuitement ces règles.

L’ensemble des règles disponibles de base se trouve ici :

ls -l /usr/share/modsecurity/

On trouve dans ce dossier plusieurs types de CRS (règles) : des règles de base, des règles expérimentales, des règles optionnelles, etc.

**NB :** Il peut arriver que les règles ne soient pas installées par défaut, dans ce cas procédé à l’installation manuellement : <https://modsecurity.org/crs/>.

Dans ce cas, veiller à bien utiliser le bon chemin lors de la mise en place de votre configuration. Notamment pour les questions qui vont suivre.

Pour savoir quelles sont les règles en cours d’application, il faut vérifier quelles sont les règles du module qui sont actives. Pour cela :

vi /etc/apache2/mods-enabled/security2.conf

Quand on affiche le fichier de configuration, on constate que tous les fichiers terminant par .conf dans le dossier /etc/modsecurity sont pris en compte, les autres non. De base, le bon répertoire n’est pas renseigné, modifiez le pour utiliser ceux dans: /usr/share/modsecurity-crs/.

**NB :** N’oubliez pas de recharger votre configuration Apache.

Afin de connaitre la version des règles utilisées, vous pouvez utiliser la commande suivante :

apt-cache policy modsecurity-crs

**1. Quelle est la version présente sur votre serveur ?**

Rendez-vous sur votre application vulnérable et tentez quelques injections pour voir si des choses sont relevées. Si c’est bien le cas nous devrions obtenir des informations dans nos logs.

tail -f /var/log/apache2/modsec\_audit.log

Faites différents tests avec les exercices présents, entre autres :

* Injection SQL
* XSS Réfléchi
* Injection de commandes

**2. Qu’observez-vous ?**

**3. Qu’en déduisez-vous ?**

**4. Cela vous semble-il suffisant ? Justifiez.**

Création de nouvelles règles

Il est possible de créer ses propres règles, en les ajoutant dans un fichier.

vi /etc/modsecurity/modsecurity\_custom\_rules.conf

En vous appuyant sur la documentation officielle et vos recherches, créez une nouvelle règle pour la partie « Command Injection »

**1. Proposez votre règle pour empêcher l’injection de commande et n’autoriser que les IP**

Brute-Force

Avec mod-security

**1. Utiliser l’exercice de brute force et tenter d’en effectuer un sur l’application. Vous êtes libre de choisir l’outil que vous souhaitez pour l’effectuer. Entrez la commande saisie :**

**2. Observez les logs. Qu’en déduisez-vous ?**

**3. Mettez en place votre propre règle sur le WAF pour vous prémunir contre ce type d’attaque. Faite un copier-coller de la règle ci-dessous.**

Avec mod-evasive

Il s’agit à nouveau d’un module apache qui bloque les requêtes trop fréquentes en provenance d'une même IP (brute-force/DOS).

En utilisant maintenant le module mod-evasive, réitérer l’opération afin de prémunir votre application contre les attaques de type brute force.

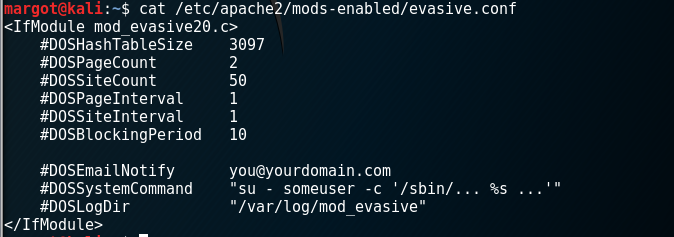
sudo apt-get install libapache2-mod-evasive

On vérifie que le module a correctement été chargé

sudo apachectl -M | grep evasive

Toute la configuration de ce module se trouve dans le fichier

vi /etc/apache2/mods-enabled/evasive.conf



**1. Appliquer une configuration qui permette de se prémunir contre le brute force et qui respecte les critères suivants :**

* Pas plus de 3 pages par seconde
* Pas plus de 200 requêtes par seconde
* Client bloqué durant une heure

Quand votre configuration est effectuée faites un reload d’Apache et vérifiez que cela fonctionne.

Vous veillerez à Copier/Coller votre configuration ci-dessous :

**2. Faites une capture d’écran de votre liste d’IP blacklistées**

Upload de fichier

Passons maintenant à la partie upload de fichier.

Dans un premier temps Uploader un webshell au format .php

**1. Qu’observer vous ?**

**2. Qu’en déduisez-vous ?**

**3. Dans un second temps uploader un fichier webshell en tentant de bypasser le WAF qui est en place. Proposer votre méthode et apporter les preuves de vos tests.**

Scan Nmap

Lancer un scan nmap classique sur l’application

Nmap -sT -sV $[ip](http://www.securityfail.com/)

**1. Qu’observez-vous ?**

**2. Pourquoi ce comportement ?**

Signature serveur

Effectuez une requête à la racine de [votre](http://www.securityfail.com/) site et observez la réponse renvoyée par le serveur.

curl – I $ip

**1. Quelle est la version de votre serveur ?**

Modifiez à l’aide de mod security le header Server, pour ce faire, il faut placer dans la configuration (/etc/modsecurity/modsecurity.conf) la directive "**SecServerSignature**" suivie du message désiré.

SecServerSignature "Stupid Server"

Vérifiez que la réponse serveur ne contient plus la version utilisée

A la différence d’Apache, ce changement permet uniquement de modifier le nom du serveur renvoyé. Il est donc préférable d’apporter directement la modification dans la configuration de apache.

sudo vi /etc/apache2/apache2.conf

Apporter le changement et observer la différence.

1. Outils d’audit

Il existe quelques outils qui peuvent vous aidez à jauger le niveau de sécurité de votre serveur, nous allons nous intéresser ici à lynis : <https://cisofy.com/lynis/>

Installation

sudo apt-get install lynis

Exécution

Pour lancer le scan de son système, il faut taper :

sudo lynis -c

Si vous voulez lancer un scan automatique, il faut utiliser cette commande :

sudo lynis --check-all -Q

Les logs sont stockés ici : /var/log/lynis.log

Pour visualiser rapidement dans vos logs uniquement les choses urgentes (warning), vous pouvez utiliser la commande grep :

sudo grep Warning /var/log/lynis.log

**1. Analysez les remontées faîtes par Lynis ? Qu’en pensez-vous ? Avez-vous des recommandations ?**

Les données remontées sont difficiles à comprendre.

Expliquer plus en détails les problèmes.