Practical 4 – Incident Response Simulation

Objective

The primary objective of this practical is to simulate a phishing attack in a controlled environment and collect relevant system artifacts to support an **incident response investigation**. By executing a controlled malicious payload on the Windows VM, we can understand how phishing attacks manifest in the system, how the operating system logs process and network activity, and how tools like Velociraptor can assist in artifact collection. This exercise helps reinforce practical skills in **digital forensics, threat hunting, and incident response**.

Concept

Phishing attacks remain one of the most common entry points for attackers. They often involve social engineering to trick a user into executing a malicious file or script. In many enterprise environments, PowerShell is used by attackers because it is a **legitimate, built-in administrative tool**, and its execution can be manipulated with flags like `-ExecutionPolicy Bypass` to avoid standard security controls.

Incident response is the practice of detecting, analyzing, and responding to security incidents. In this practical, we focus on two main types of artifacts:

- **Process artifacts:** Information about running processes, such as `powershell.exe`, and associated command lines, which can reveal malicious execution patterns.

- **Network artifacts:** Active network connections and ports monitored using `netstat` to detect any suspicious external communication initiated by the payload.

By combining both process and network information, incident responders can **trace the attack path, identify compromised endpoints, and gather evidence** for remediation or further analysis. Tools like **Velociraptor** automate artifact collection across endpoints and enable live queries, significantly accelerating incident response efforts.

VM Setup

Both virtual machines were started successfully:

- **Windows VM:** Acts as the victim endpoint, simulating a user machine in an enterprise network.

- **Kali Linux VM:** Simulates the attacker environment, capable of generating phishing payloads and capturing network traffic.

This setup allows students to understand both the attacker's and the defender's perspectives in a controlled lab environment, without exposing real systems to risk.

Step 1 – Simulate Phishing Payload

On the Windows VM:

1. Open **Notepad**.

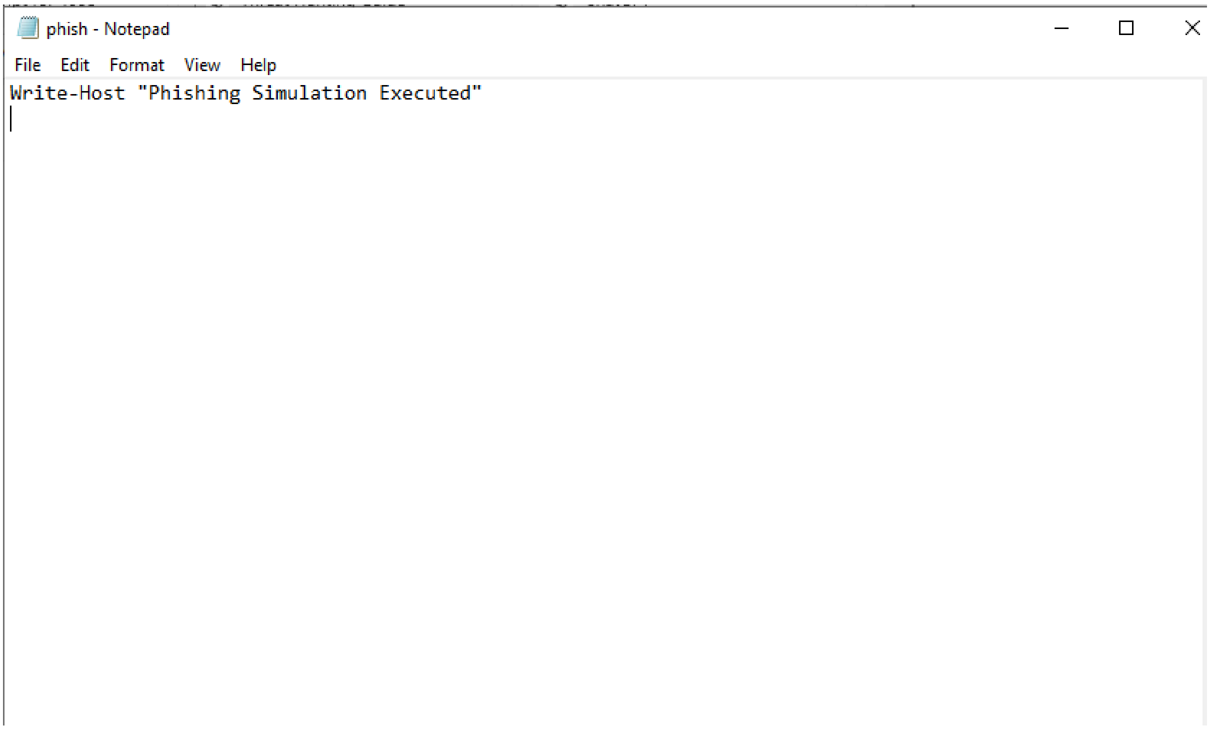
2. Create a file named **phish.ps1**.

3. Add the following line of code:

```
Write-Host "Phishing Simulation Executed"
```

4. Save the file to `C:\Users\vboxuser\Desktop\phish.ps1`.

This simple PowerShell script simulates a phishing payload. While it does not perform malicious actions, it generates **observable system artifacts**, which allow us to study the behavior of PowerShell execution in the system.



Step 2 – Execute Payload

1. Open **PowerShell as Administrator**.
2. Execute the script using:

```
powershell -ExecutionPolicy Bypass -File C:\Users\vboxuser\Desktop\phish.ps1
```

Expected Output:

```
Phishing Simulation Executed
```

This command simulates the behavior of a real phishing attack, bypassing normal execution restrictions. The creation of the `powershell.exe` process demonstrates how malicious scripts are executed on Windows systems.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> powershell -ExecutionPolicy Bypass -File C:\Users\vboxuser\Desktop\phish.ps1
Phishing Simulation Executed
PS C:\Windows\system32>
PS C:\Windows\system32>
```

Result: Payload executed successfully, producing process artifacts for collection.

Step 3 – Artifact Collection

Process Artifacts

Command used:

```
tasklist
```

Key Output :

```
powershell.exe          5536 Console     1    73,840 K

notepad.exe             3920 Console     1    33,392 K
```

By observing the process list, we can confirm that the PowerShell process is running. This provides evidence of script execution, which is a crucial step in incident response investigations. Collecting the **Process ID (PID)**, memory usage, and command-line arguments allows investigators to correlate process behavior with suspicious activity.

```
PS C:\Windows\system32> tasklist

Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
System Idle Process              0 Services                   0          8 K
System                           4 Services                   0        140 K
Registry                        92 Services                   0     22,548 K
smss.exe                       316 Services                   0        916 K
csrss.exe                      412 Services                   0      4,528 K
csrss.exe                      488 Console                    1      5,180 K
wininit.exe                    508 Services                   0      6,212 K
winlogon.exe                   552 Console                    1      9,132 K
services.exe                   616 Services                   0      7,304 K
lsass.exe                      636 Services                   0     15,956 K
fontdrvhost.exe                728 Services                   0      2,160 K
fontdrvhost.exe                736 Console                    1      3,848 K
svchost.exe                    744 Services                   0     24,440 K
svchost.exe                    860 Services                   0     11,796 K
dwm.exe                        952 Console                    1     66,992 K
svchost.exe                    368 Services                   0     71,240 K
svchost.exe                    416 Services                   0     17,804 K
svchost.exe                    484 Services                   0     29,028 K
svchost.exe                    940 Services                   0     59,116 K
svchost.exe                   1104 Services                   0     28,876 K
svchost.exe                   1236 Services                   0     17,052 K
Memory Compression            1432 Services                   0     44,484 K
svchost.exe                   1532 Services                   0     10,188 K
svchost.exe                   1616 Services                   0     11,664 K
svchost.exe                   1648 Services                   0      9,772 K
svchost.exe                   1660 Services                   0      8,024 K
svchost.exe                   1876 Services                   0     16,516 K
spoolsv.exe                   1956 Services                   0     13,004 K
svchost.exe                    876 Services                   0     14,176 K
svchost.exe                   2084 Services                   0     35,096 K
SearchIndexer.exe             2468 Services                   0     25,568 K
svchost.exe                   2852 Services                   0      7,444 K
sihost.exe                    2956 Console                    1     26,568 K
```

```
svchost.exe                   4888 Services                   0      9,900 K
msedge.exe                    5880 Console                    1     22,916 K
WinStore.App.exe              3964 Console                    1      1,432 K
ApplicationFrameHost.exe      3164 Console                    1     30,032 K
RuntimeBroker.exe             5836 Console                    1      7,648 K
svchost.exe                   1132 Services                   0      7,112 K
MsMpEng.exe                   3848 Services                   0    172,944 K
MpDefenderCoreService.exe     6000 Services                   0     21,208 K
NisSrv.exe                    5620 Services                   0     10,992 K
dllhost.exe                   2764 Services                   0     12,216 K
dllhost.exe                   5264 Console                    1     12,984 K
dllhost.exe                   3284 Console                    1      7,432 K
msedge.exe                     372 Console                    1     49,084 K
msedge.exe                    4820 Console                    1    270,004 K
msedge.exe                    2348 Console                    1     35,004 K
taskhostw.exe                 5704 Console                    1     20,940 K
smartscreen.exe               1456 Console                    1     23,284 K
notepad.exe                   3920 Console                    1     33,392 K
audiodg.exe                   2512 Services                   0     12,024 K
powershell.exe                5536 Console                    1     73,840 K
conhost.exe                   3924 Console                    1     18,808 K
tasklist.exe                  5140 Console                    1      9,276 K
WmiPrvSE.exe                  3424 Services                   0      9,404 K
```

Network Artifacts

Command used:

```
netstat -ano
```

Key Output :

```
TCP   10.0.2.15:49784   4.213.25.240:443   ESTABLISHED   368

TCP   10.0.2.15:49810   35.190.80.1:443    ESTABLISHED   4548
```

Observation: No network connections were associated with the `powershell.exe` process, which is expected because the payload is local-only. In a real-world scenario, phishing scripts often communicate with **command-and-control servers** to exfiltrate data or download further payloads.

```
PS C:\Windows\system32> netstat -ano

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       860
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       1104
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING       2108
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       636
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING       508
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING       484
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING       368
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING       1956
  TCP    0.0.0.0:49669          0.0.0.0:0              LISTENING       616
  TCP    10.0.2.15:139          0.0.0.0:0              LISTENING       4
  TCP    10.0.2.15:49784        4.213.25.240:443       ESTABLISHED     368
  TCP    10.0.2.15:49810        35.190.80.1:443        ESTABLISHED     4548
  TCP    10.0.2.15:49926        4.213.25.242:443       ESTABLISHED     368
  TCP    10.0.2.15:49986        185.199.110.153:443    ESTABLISHED     4548
  TCP    10.0.2.15:49996        4.213.133.109:443      CLOSE_WAIT      4548
  TCP    10.0.2.15:49998        23.48.245.227:443      CLOSE_WAIT      4032
  TCP    [::]:135               [::]:0                 LISTENING       860
  TCP    [::]:445               [::]:0                 LISTENING       4
  TCP    [::]:7680              [::]:0                 LISTENING       2108
  TCP    [::]:49664             [::]:0                 LISTENING       636
  TCP    [::]:49665             [::]:0                 LISTENING       508
  TCP    [::]:49666             [::]:0                 LISTENING       484
  TCP    [::]:49667             [::]:0                 LISTENING       368
  TCP    [::]:49668             [::]:0                 LISTENING       1956
  TCP    [::]:49669             [::]:0                 LISTENING       616
  UDP    0.0.0.0:5050           *:*                                    1104
  UDP    0.0.0.0:5353           *:*                                    4380
  UDP    0.0.0.0:5353           *:*                                    1236
  UDP    0.0.0.0:5353           *:*                                    4380
  UDP    0.0.0.0:5355           *:*                                    1236
  UDP    0.0.0.0:61728          *:*                                    4548
  UDP    0.0.0.0:62225          *:*                                    4548
  UDP    10.0.2.15:137          *:*                                    4
  UDP    10.0.2.15:138          *:*                                    4
  UDP    10.0.2.15:1900         *:*                                    1132
  UDP    10.0.2.15:55339        *:*                                    1132
  UDP    127.0.0.1:1900         *:*                                    1132
  UDP    127.0.0.1:53594        *:*                                    368
  UDP    127.0.0.1:55340        *:*                                    1132
  UDP    [::]:5353              *:*                                    1236
  UDP    [::]:5353              *:*                                    4380
```

Step 4 – Velociraptor (Simulation)

Velociraptor allows automated collection of endpoint artifacts and logs. Typical live queries include:

```
SELECT * FROM processes
```

```
SELECT * FROM netstat
```

Using Velociraptor, analysts can gather process and network artifacts **across multiple endpoints simultaneously**, which reduces manual effort and enables faster response to security incidents.

Step 5 – Attack Path Summary

A phishing simulation was conducted on the Windows endpoint by executing a PowerShell script. Upon execution, a new process `powershell.exe` was spawned, which was logged by the system. Process artifacts, including the PID and memory usage, were collected using a tasklist. Network artifacts were also collected using `netstat` to monitor any external connections, but none were observed because the payload did not include network activity. This demonstrates how even simple scripts generate measurable artifacts that can be used to trace an attack's execution flow. Collecting and analyzing such data allows incident responders to **identify compromised systems, reconstruct attack paths, and plan containment actions**. By simulating both the attacker (Kali VM) and defender (Windows VM + artifact collection), students gain a realistic understanding of the **complete incident response workflow**, bridging theory and practice.

Step 6 – Indicators of Compromise (IOCs)

Indicator	Value
Process Name	powershell.exe
PID	5536
Script File	phish.ps1
Execution Method	`-ExecutionPolicy Bypass`
Network Activity	None for this process

These IOCs provide a starting point for further investigation and correlate observed artifacts with potential threats.

Kali VM Usage

The Kali Linux VM was used to simulate an attacker's environment. It provides tools to generate phishing payloads, run automated simulations with **CALDERA or Metasploit**, and capture network traffic. This dual perspective (attacker and victim) allows students to understand both attack and defense techniques in a **controlled lab environment**.

Conclusion

- PowerShell phishing payload executed successfully.
- Process and network artifacts were collected and analyzed.
- Indicators of Compromise were identified, including script location, execution method, and process details.
- This lab demonstrates how **incident response workflows** operate in real-world scenarios, emphasizing artifact collection, attack path reconstruction, and IOC documentation.
- Students gain practical experience bridging **theory, DFIR tools, and manual investigation techniques**.