

1. Lab Environment Setup

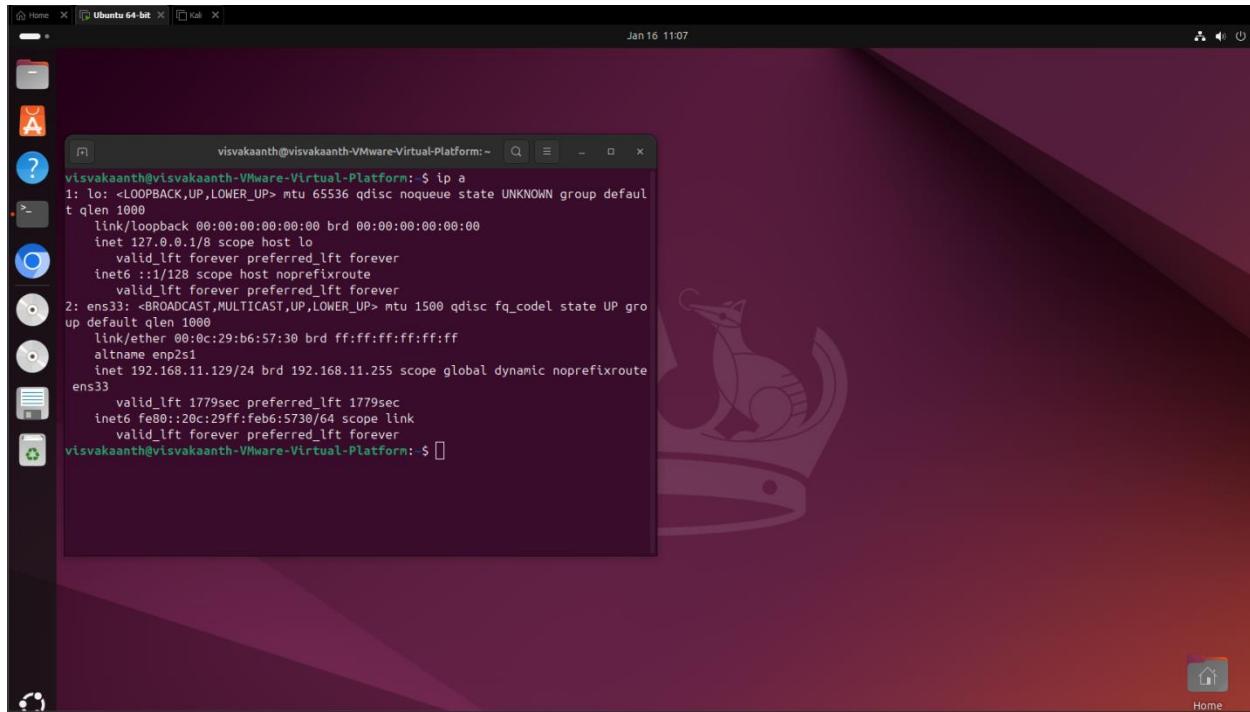
As part of this SOC lab simulation, an Ubuntu virtual machine was deployed using VMware Workstation to represent a typical employee workstation found in an enterprise environment. This machine acts as the target system for the simulated attacker activity.

The Ubuntu VM was configured with the following:

- Operating System: Ubuntu Linux
- Virtualization Platform: VMware Workstation
- Network Mode: (NAT)
- Assigned IP Address: 192.168.11.129

The purpose of this machine is to generate realistic system and authentication logs that can be forwarded to Splunk for monitoring and analysis. These logs will be used to detect and investigate malicious activity performed by the attacker during the simulation.

A screenshot of the terminal output showing the assigned IP address is provided below for reference



```
visvakaanth@visvakaanth-VMware-Virtual-Platform: ~ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:b6:57:30 brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet 192.168.11.129/24 brd 192.168.11.255 scope global dynamic noprefixroute
        valid_lft 1779sec preferred_lft 1779sec
        inet6 fe80::20c:29ff:feb6:5730/64 scope link
            valid_lft forever preferred_lft forever
```

1.1 Firewall Configuration on Ubuntu Endpoint

The Ubuntu endpoint was secured using UFW (Uncomplicated Firewall) to simulate a baseline security posture commonly implemented on enterprise workstations.

Inbound connections were restricted by default, and only essential services were permitted. Specifically, all inbound traffic was denied except for SSH access, which was required for administrative purposes.

2. Splunk Deployment & Log Receiver Configuration

Splunk Enterprise was deployed on the Windows host system to act as the central log collection and analysis platform for this SOC lab environment.

To enable log ingestion from the Ubuntu endpoint, Splunk was configured to listen for incoming data on TCP port 9997, which is the default port used by Splunk Universal Forwarders for log forwarding.

The following steps were performed:

- Splunk Enterprise was installed and verified to be running successfully.
- A receiving port (9997) was enabled.
- The listener was activated to allow inbound log data from the Ubuntu VM.

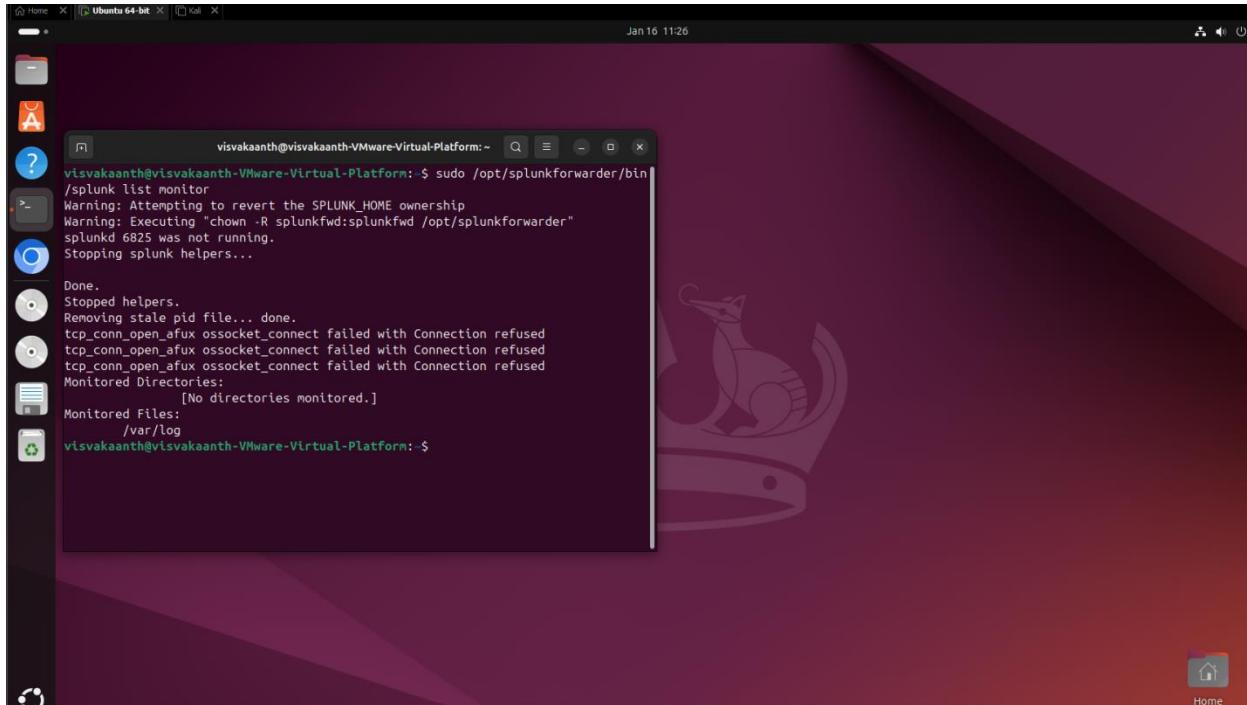
A screenshot showing Splunk listening on port 9997 is provided below for reference.

The screenshot shows the Splunk Enterprise web interface. The top navigation bar includes links for 'splunk>enterprise', 'Apps', 'Administrator', 'Messages', 'Settings', 'Activity', 'Help', and a search bar. Below the navigation is a breadcrumb trail: 'Forwarding and receiving > Receive data'. The main content area has a title 'Receive data' and a sub-section 'Forwarding and receiving > Receive data'. It displays a table with one item: 'Showing 1-1 of 1 item'. The table has columns for 'Listen on this port', 'Status', and 'Actions'. The first row shows '9997' in the 'Listen on this port' column, 'Enabled | Disable' in the 'Status' column, and a 'Delete' link in the 'Actions' column. There are also 'filter' and 'New Receiving Port' buttons at the top of the table. The bottom of the page features a footer with links for 'Documentation', 'Community', 'Support', 'Blog', 'Events', and 'About'.

This configuration allows the Ubuntu system to forward system and authentication logs to Splunk in real time, enabling centralized monitoring and detection of suspicious activity.

3. Ubuntu Log Forwarder Configuration

The Splunk Universal Forwarder on the Ubuntu endpoint was configured to monitor the “/var/log” directory, including authentication and system logs. These logs were forwarded in real-time to the Splunk Enterprise instance over TCP port 9997.

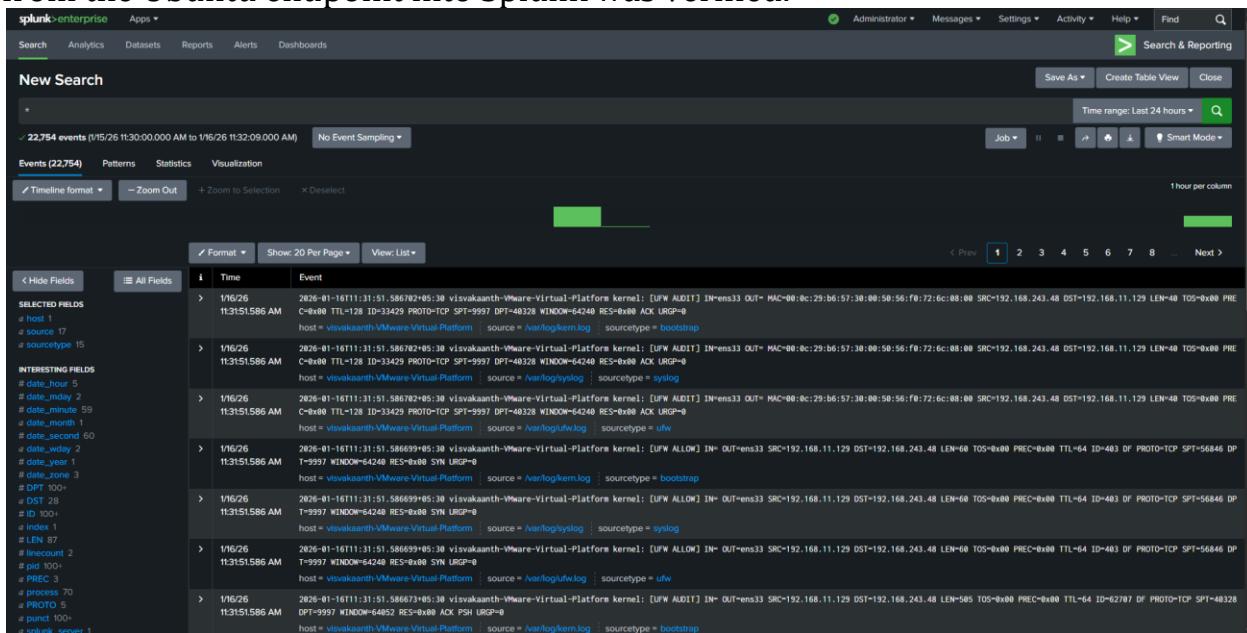


A screenshot of an Ubuntu terminal window titled "Ubuntu 64-bit". The terminal shows the command \$ sudo /opt/splunkforwarder/bin/splunk list monitor being run. The output indicates that the Splunk Home ownership is being reverted, and the splunkd process is not running. It then stops the splunk helpers and removes a stale pid file. The monitored directories and files are listed as empty. Finally, the user exits the terminal.

```
visvakaanth@visvakaanth-VMware-Virtual-Platform: ~$ sudo /opt/splunkforwarder/bin/splunk list monitor
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R splunkfwd:splunkfwd /opt/splunkforwarder"
splunkd 6825 was not running.
Stopping splunk helpers...

Done.
Stopped helpers.
Removing stale pid file... done.
tcp_conn_open_afux ossocket_connect failed with Connection refused
tcp_conn_open_afux ossocket_connect failed with Connection refused
tcp_conn_open_afux ossocket_connect failed with Connection refused
Monitored Directories:
[No directories monitored.]
Monitored Files:
/var/log
visvakaanth@visvakaanth-VMware-Virtual-Platform: ~$
```

To validate that log forwarding was functioning correctly, real-time log ingestion from the Ubuntu endpoint into Splunk was verified.



A screenshot of the Splunk Enterprise search interface. The search bar shows a time range of "Last 24 hours". The results table displays 22,754 events from the "Events" tab. The table includes columns for Time and Event. The first few events are listed as follows:

Time	Event
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW AUDIT] IN=en33 OUT= MAC=08:0c:29:b6:57:30:00:50:56:08:72:6c:08:00 SRC=192.168.243.48 DST=192.168.11.129 LEN=40 TOS=0x00 PRE	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW AUDIT] IN=en33 OUT= MAC=08:0c:29:b6:57:30:00:50:56:08:72:6c:08:00 SRC=192.168.243.48 DST=192.168.11.129 LEN=40 TOS=0x00 PRE	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW AUDIT] IN=en33 OUT= MAC=08:0c:29:b6:57:30:00:50:56:08:72:6c:08:00 SRC=192.168.243.48 DST=192.168.11.129 LEN=40 TOS=0x00 PRE	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW AUDIT] IN=en33 OUT= MAC=08:0c:29:b6:57:30:00:50:56:08:72:6c:08:00 SRC=192.168.243.48 DST=192.168.11.129 LEN=40 TOS=0x00 PRE	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW ALLOW] IN= OUT=en33 SRC=192.168.11.129 DST=192.168.243.48 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=483 DF PROTO=TCP SPT=56846 DP	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW ALLOW] IN= OUT=en33 SRC=192.168.11.129 DST=192.168.243.48 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=483 DF PROTO=TCP SPT=56846 DP	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW ALLOW] IN= OUT=en33 SRC=192.168.11.129 DST=192.168.243.48 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=483 DF PROTO=TCP SPT=56846 DP	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW ALLOW] IN= OUT=en33 SRC=192.168.11.129 DST=192.168.243.48 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=483 DF PROTO=TCP SPT=56846 DP	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap
2026-01-16T11:31:51.586782+05:30 visvakaanth-VMware-Virtual-Platform kernel: [UFW ALLOW] IN= OUT=en33 SRC=192.168.11.129 DST=192.168.243.48 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=483 DF PROTO=TCP SPT=56846 DP	host = visvakaanth-VMware-Virtual-Platform source = /var/log kern.log sourcetype = bootstrap

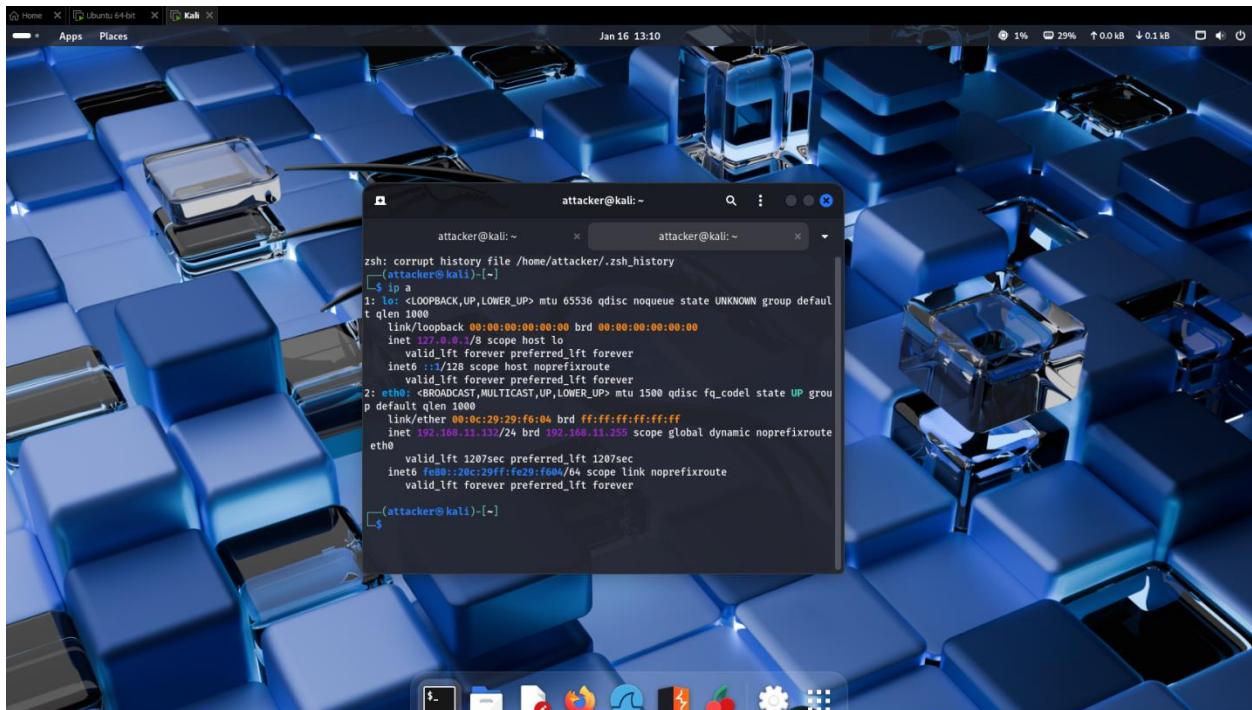
4. Attack Simulation – Reconnaissance & Port Scanning

To simulate real-world attacker behavior, a Kali Linux virtual machine was deployed to act as the attacker system within the lab environment. Kali Linux is a penetration testing distribution commonly used by threat actors and security professionals to perform reconnaissance and exploitation activities.

The objective of this phase was to perform reconnaissance on the Ubuntu endpoint in order to identify open ports and running services that could potentially be targeted for further exploitation.

The Kali VM was configured with the following:

- Operating System: Kali Linux
- Virtualization Platform: VMware Workstation
- Network Mode: (NAT)
- Assigned IP Address: 192.168.11.132



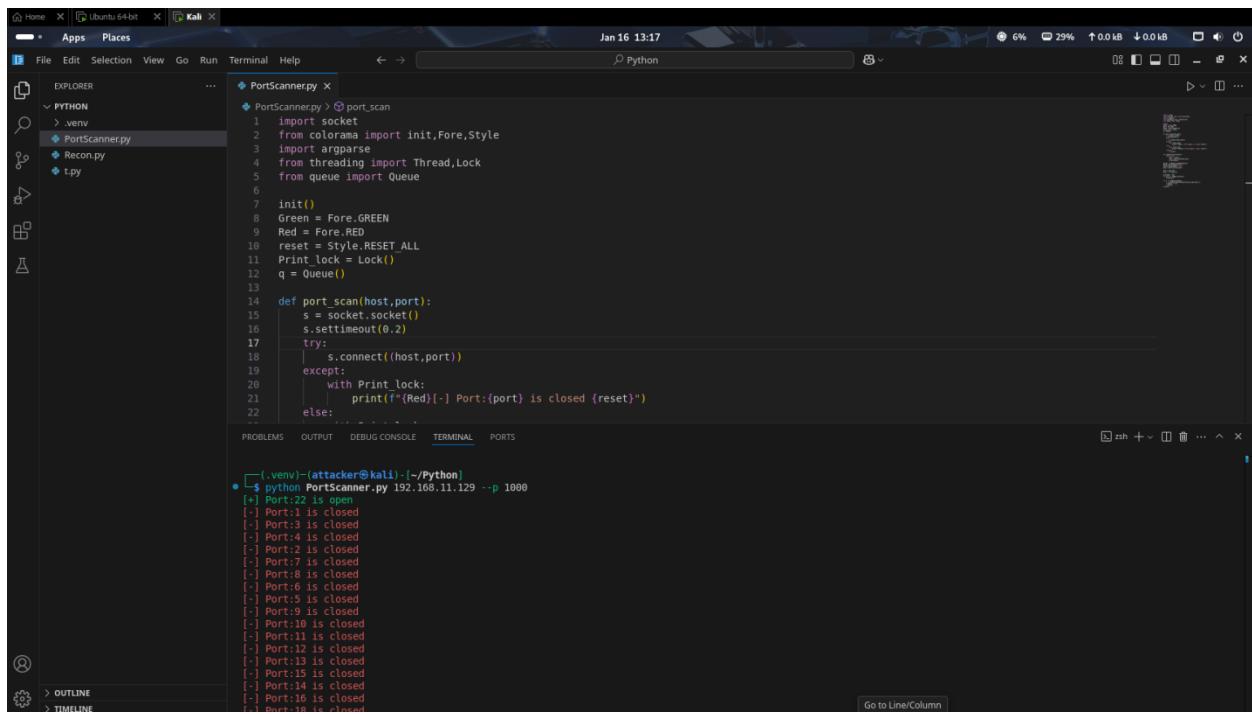
5. Custom Port Scanning Attack from Kali Linux

To simulate attacker reconnaissance activity, a custom Python-based port scanning script was executed from the Kali Linux attacker machine against the Ubuntu target endpoint. Instead of using standard tools such as Nmap, a custom script utilizing Python's socket library was developed to mimic how threat actors may use their own tooling to evade basic detection mechanisms.

The objective of this scan was to identify open ports running on the Ubuntu system that could be targeted for further exploitation.

The script attempted TCP connections against a range of ports on the Ubuntu host to determine which services were accessible.

A screenshot of the Kali terminal executing the custom port scanning script is provided below



```
PortScanner.py
1 import socket
2 from colorama import init,Fore,Style
3 import argparse
4 from threading import Thread,Lock
5 from queue import Queue
6
7 init()
8 Green = Fore.GREEN
9 Red = Fore.RED
10 reset = Style.RESET_ALL
11 Print_lock = Lock()
12 q = Queue()
13
14 def port_scan(host,port):
15     s = socket.socket()
16     s.settimeout(0.2)
17     try:
18         s.connect((host,port))
19     except:
20         with Print_lock:
21             print(f"{Red}[-] Port:{port} is closed {reset}")
22     else:
23         with Print_lock:
24             print(f"{Green}[+] Port:{port} is open{reset}")

$ python PortScanner.py 192.168.11.129 --p 1000
[+] Port:22 is open
[-] Port:3 is closed
[-] Port:4 is closed
[-] Port:5 is closed
[-] Port:6 is closed
[-] Port:7 is closed
[-] Port:8 is closed
[-] Port:9 is closed
[-] Port:10 is closed
[-] Port:11 is closed
[-] Port:12 is closed
[-] Port:13 is closed
[-] Port:14 is closed
[-] Port:15 is closed
[-] Port:16 is closed
[-] Port:18 is closed
```

During the reconnaissance phase, the custom Python port scanning script did not identify any open ports on the Ubuntu endpoint, with the exception of the SSH service.

This outcome was expected due to the restrictive UFW firewall configuration in place.

6. Detection & Alerting - Port Scanning Activity

To detect reconnaissance activity targeting the Ubuntu endpoint, Splunk Enterprise was configured with a custom alert designed to identify port scanning behavior.

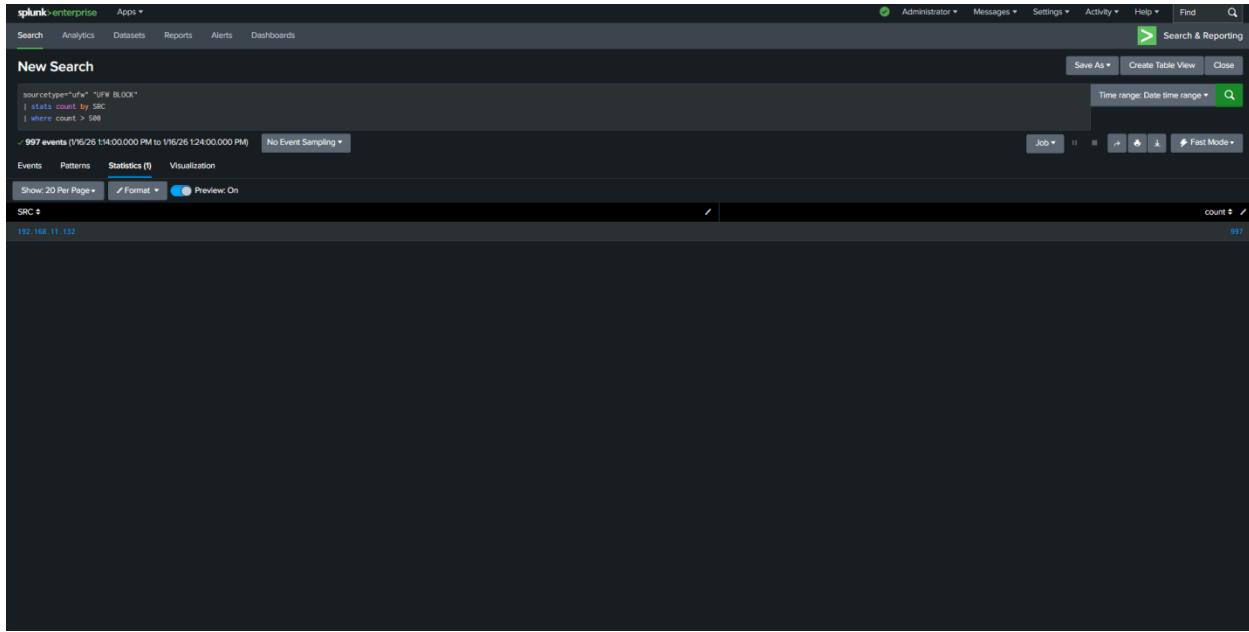
The alert is based on a search query that monitors repeated connection attempts from a single source IP address to multiple ports within a defined time window. This pattern is commonly associated with automated scanning activity performed during the reconnaissance phase of an attack.

During the execution of the custom Python port scanning script from the Kali Linux attacker machine, the alert was successfully triggered. This confirms that the detection logic is effective in identifying suspicious scanning behavior in real time.

A screenshot of the triggered Splunk alert is provided below for reference.

The screenshot shows the Splunk Enterprise web interface. At the top, there's a navigation bar with links for Search, Analytics, Datasets, Reports, Alerts, and Dashboards. On the right side of the header, there are links for Administrator, Messages, Settings, Activity, Help, and a search bar labeled "Search & Reporting". Below the header, the main content area has a title "Port Scan" and a sub-section "Trigger History". The "Trigger History" table lists 11 entries, each with a timestamp and a "View Results" link. The columns are "TriggerTime" (sorted by descending time) and "Actions". The first few rows show timestamps like "2026-01-16 13:27:00 India Standard Time" and "2026-01-16 13:26:00 India Standard Time". The last row shows "2026-01-16 12:19:01 India Standard Time".

TriggerTime	Actions
2026-01-16 13:27:00 India Standard Time	View Results
2026-01-16 13:26:00 India Standard Time	View Results
2026-01-16 13:25:00 India Standard Time	View Results
2026-01-16 13:24:01 India Standard Time	View Results
2026-01-16 13:23:01 India Standard Time	View Results
2026-01-16 13:22:01 India Standard Time	View Results
2026-01-16 13:21:01 India Standard Time	View Results
2026-01-16 13:20:01 India Standard Time	View Results
2026-01-16 13:19:01 India Standard Time	View Results
2026-01-16 13:18:01 India Standard Time	View Results
2026-01-16 12:19:01 India Standard Time	View Results



7. Attack Simulation – SSH Brute Force Attack

Following the reconnaissance phase, the attacker targeted the exposed SSH service on the Ubuntu endpoint in an attempt to gain unauthorized access. Since SSH was the only service permitted through the firewall, it became the primary attack vector.

A credential-based brute force attack was executed from the Kali Linux attacker machine against the Ubuntu system using repeated authentication attempts with different username and password combinations. This technique is commonly used by attackers to gain initial access when weak or reused credentials are present.

After multiple failed login attempts, the attacker successfully authenticated to the system, resulting in unauthorized access to the Ubuntu endpoint and the establishment of an interactive SSH session.

The sequence of events observed was:

1. Multiple failed SSH authentication attempts
2. Successful SSH login from the attacker system
3. Establishment of an interactive session on the target host

A screenshot showing the brute force activity and successful login is provided below.

```

stop on success => true
msf auxiliary(scanner/ssh/ssh_login) > run
[*] 192.168.11.129:22 - Starting brute force
[*] 192.168.11.129:22 - Failed: 'visvakaanth:V7!nQe$P2mZ@9A'
[!] No active DB -- Credential data will not be saved!
[-] 192.168.11.129:22 - Failed: 'visvakaanth:x84Kp18wSQT'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:9D1cZQm2PBL'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:A57xIWQRZP'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:mT8qZ1Pq7D'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:K19P9xxZ7QF'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:R21ZqX#R0K'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:7Qg1ZxP#R0K'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:DfZ71Pq9K'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:JfZ71Pq9K'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:x071ZQx7KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:Z9P9tQh7KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:K7ZgPw09KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:1QZ7gP9xxKR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:R2Z9P9t0xx7K'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:x7a2P10#KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:9g1x07P#KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:PwZ9Qx7KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:Q1Z7gP9xxKR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:@9107#KR'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:NeuralShadowRiver'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:Quantum@reams42'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:DarkNeuronC1p00Lf'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:Gh0stP4ctc88'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:VoidKernel_27'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:DarkNeuronSk8'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:Cyber@ronin_91'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:DreamLogic77'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:PhantomITrace6'
[-] 192.168.11.129:22 - Failed: 'visvakaanth:Secon'
[+] 192.168.11.129:22 - Success: 'visvakaanth:secon' 'uid=1000(visvakaanth) gid=1000(visvakaanth) groups=1000(visvakaanth),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),100(users),114(lpadmin) Linux visvakaanth-V
[*] Msf::Virtual-Platform 6.14.0-37-generic #37--04.1-Ubuntu SMP PREEMPT_DYNAMIC Thu Nov 20 10:25:38 UTC 2026-01-16 15:56:40 +0530
[*] SSh session 1 opened (192.168.11.132:41091 -> 192.168.11.129:22 (192.168.11.129)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
[*] msf auxiliary(scanner/ssh/ssh_login) > sessions
```

Active sessions

Id	Name	Type	Information	Connection
1		shell	linux	SSH attacker @ 192.168.11.132:41091 -> 192.168.11.129:22 (192.168.11.129)

8. Detection & Analysis – SSH Brute Force Activity

To detect credential-based attacks targeting the Ubuntu endpoint, a custom alert was configured in Splunk to identify excessive failed SSH authentication attempts from a single source IP address within a defined time window.

This detection logic is designed to identify brute force behavior, which is commonly characterized by repeated login failures followed by a potential successful authentication.

During the execution of the SSH brute force attack from the Kali Linux attacker machine, the alert was successfully triggered. The alert captured multiple failed login attempts originating from the same source, followed by a successful authentication event, confirming unauthorized access.

This sequence of events is a strong indicator of malicious activity and would require immediate investigation and response in a production environment.

A screenshot of the triggered Splunk alert is provided below for reference.

splunk-enterprise Apps ▾

Administrator ▾ Messages ▾ Settings ▾ Activity ▾ Help ▾ Find ▾

Search Analytics Datasets Reports Alerts Dashboards

SSH Brute Force Attempt

Enabled: Yes [Disable](#)

App: search

Permissions: Private by admin. [Edit](#)

Modified: Jan 16, 2026 3:45:45 PM

Alert Type: Scheduled, Cron Schedule. [Edit](#)

Trigger Condition: Number of Results is > 0. [Edit](#)

Actions: 1 Action [Edit](#)

Add to Triggered Alerts

Trigger History

20 per page ▾

TriggerTime	Actions
1 2026-01-16 16:03:01 India Standard Time	View Results
2 2026-01-16 16:02:00 India Standard Time	View Results
3 2026-01-16 16:01:00 India Standard Time	View Results
4 2026-01-16 16:00:00 India Standard Time	View Results
5 2026-01-16 15:59:00 India Standard Time	View Results
6 2026-01-16 15:58:00 India Standard Time	View Results
7 2026-01-16 15:57:00 India Standard Time	View Results
8 2026-01-16 15:56:00 India Standard Time	View Results

Search & Reporting [Edit](#)

splunk-enterprise Apps ▾

Administrator ▾ Messages ▾ Settings ▾ Activity ▾ Help ▾ Find ▾

Search Analytics Datasets Reports Alerts Dashboards

New Search

Time range: Date time range ▾

Save As ▾ Create Table View Close

src = "ssh" "failed"
| rex "from (\[ssrc>\d+\.\d+\.\d+\.\d+])"
| stats count by src
| where count > 10

30 events (1/16/26 3:53:00.000 PM to 1/16/26 4:03:00.000 PM) No Event Sampling

Events Patterns Statistics (1) Visualization

Show: 20 Per Page ▾ Format ▾ Preview: On

src 5 count 5

192.168.11.132

38

Search & Reporting

