Splunk SIEM: Setup and Incident Investigation

Date: 02.06.2025

Project: Splunk SIEM: Lab setup and incident investigation

By Thomas Lium

Contents

Executive Summary

Introduction

This project involves setting up a **Splunk SIEM environment** within a controlled virtual lab to simulate and analyze logs of cyberattacks against vulnerable machines. The primary objective is to demonstrate how Splunk can be used to **collect logs**, **detect suspicious activity**, and **investigate security incidents** through attack simulations.

The lab environment consists of multiple virtual machines, some intentionally left vulnerable to simulate high-risk assets, and one machine acting as an attacker used for reconnaissance and exploitation techniques. Splunk will be configured as the central SIEM platform to monitor, collect, and analyze logs from these endpoints as attack occur.

This project demonstrates practical skills in SIEM deployment, security monitoring, and threat detection. It simulates a real-world security operations center (SOC) workflow—**from initial compromise to detection and incident investigation using Splunk.**

Project Objectives

The objective is to demonstrate how a properly configured **Splunk SIEM** can be used to detect and investigate these threats in real time by leveraging logs and telemetry collected from various endpoints and network systems.

This lab is designed as a **hands-on simulation** of a Security Operations Center (SOC) workflow, covering the entire security monitoring lifecycle — from attack execution to threat detection, log analysis, alerting, and post-incident investigation.

Objectives to be explored:

- Splunk deployment and installation of logging agents (Splunk Forwarders)
- Basic search queries of logs
- Creation of alert rules to identify key Indicators of Compromise (IOCs)
- Investigation of simulated attacks (port scans, exploits, initial access, privilege escalation)

Scenario attack overview

This project simulates a realistic cybersecurity scenario where multiple machines within an internal network are exposed to targeted attacks from an unauthorized actor. The attacker operates from a separate machine connected to the same network sequence, emulating the tactics, techniques, and procedures (TTPs) of a real-world threat actor.

In this scenario, a threat actor has already gained access to our local network. The attacker's Kali Linux machine is attempting to identify other devices on local network and find a vulnerable machine or two. The attacker initiates a series of reconnaissance activities aimed at discovering other devices within the same subnet to later compromise, gain initial access and attempt to escalate privileges with the goal of exfiltrating various files using netcat. All actions to generate Splunk logs for monitoring and investigation.

These actions target a mixture of Windows and Linux hosts within the network, some of which are intentionally configured to be vulnerable for logging purposes.

Lab machines overview

Machine OS	**Machine name**	**Machine IP**	**Splunk ingested**
Windows 10	WINDOWS-KEVIN	10.0.0.21	✅ Yes
Ubuntu Linux	User-VMware-Virtual-Platform	10.0.0.10	✅ Yes
Linux server	Metasploitable	10.0.0.9	✅ Yes
Linux server	Darkhole	10.0.0.22	✅ Yes
Kali Linux (attacker's)	Kali	10.0.0.13	❌ No
Windows 11	DESKTOP-C1SC5GT (Splunk server)	10.0.0.2	❌ No

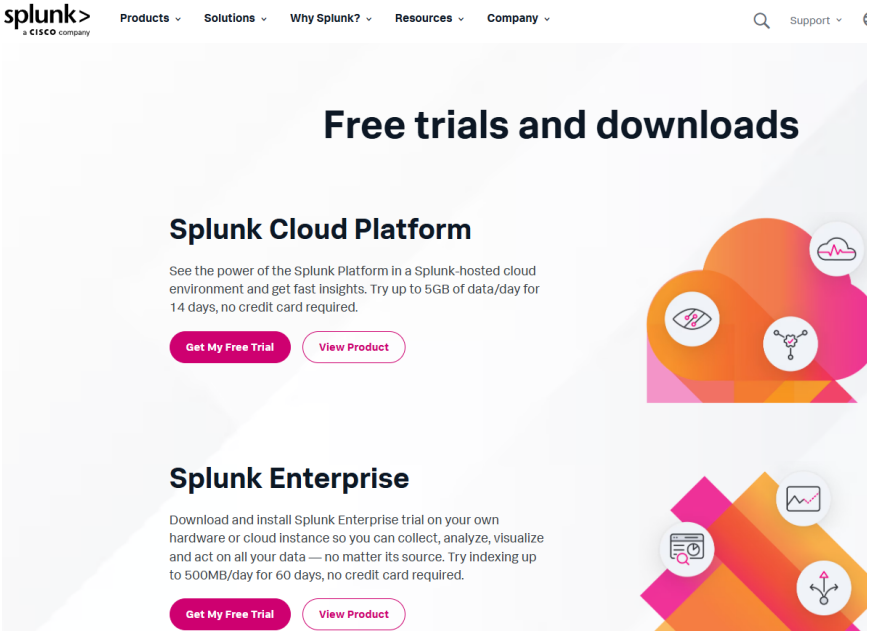
The linux servers, "Metasploitable" and "Darkhole 2" are intentionally vulnerable in order to generate relevant logs for the SIEM for later investigation.

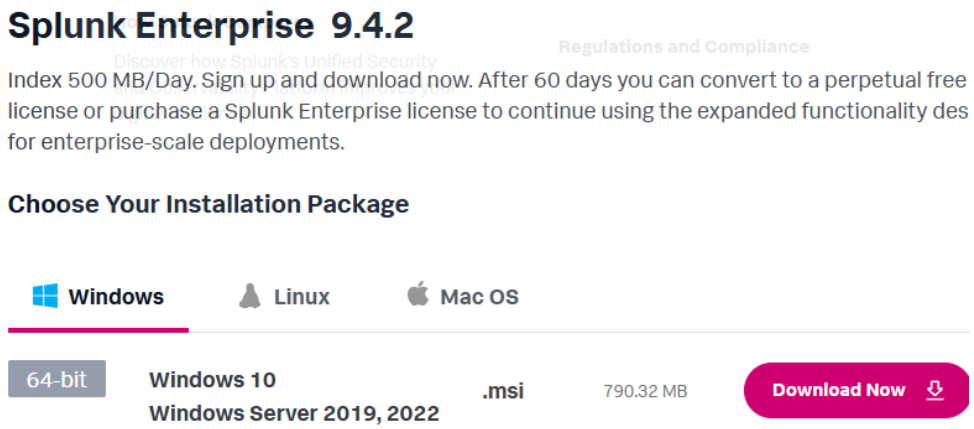
Disclaimer

This project is for educational and training purposes only. All scenarios and activities were conducted within the controlled virtual environment hosted on my local network. No real systems, networks, or users were involved or harmed.

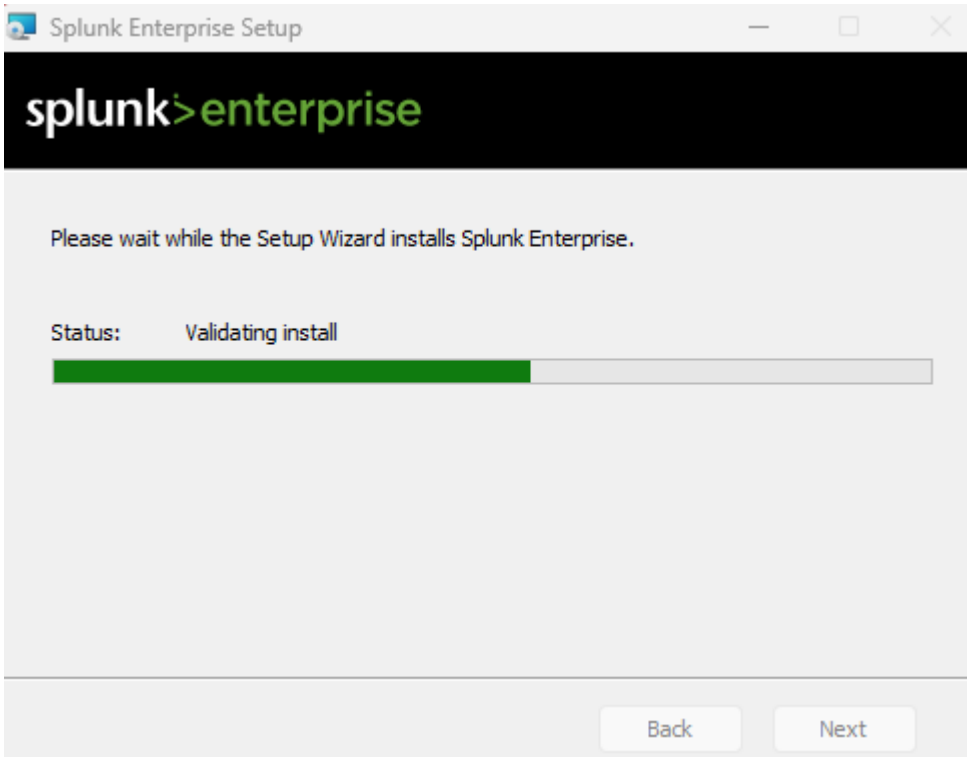
Splunk Deployment and installation of logging agents

This section will cover deployment of a Splunk SIEM server on a Windows 11 desktop machine.

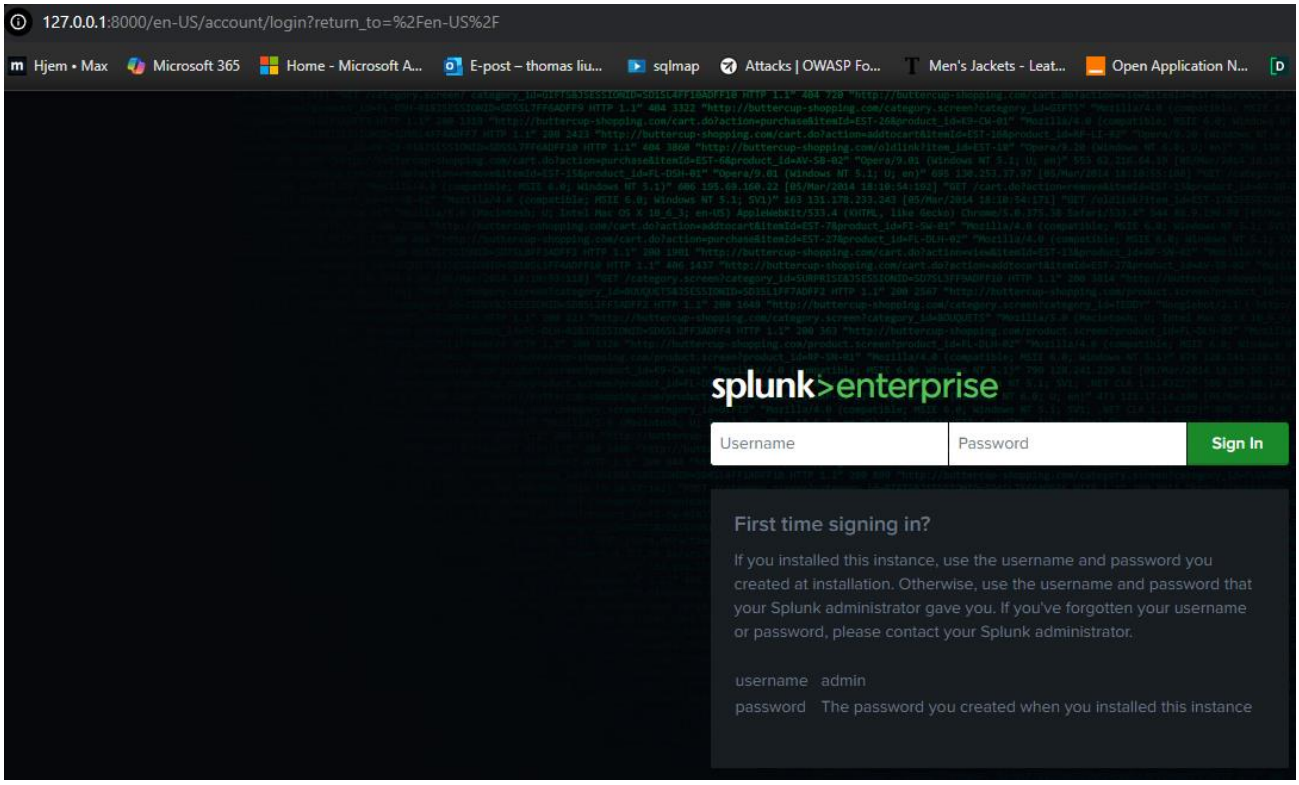




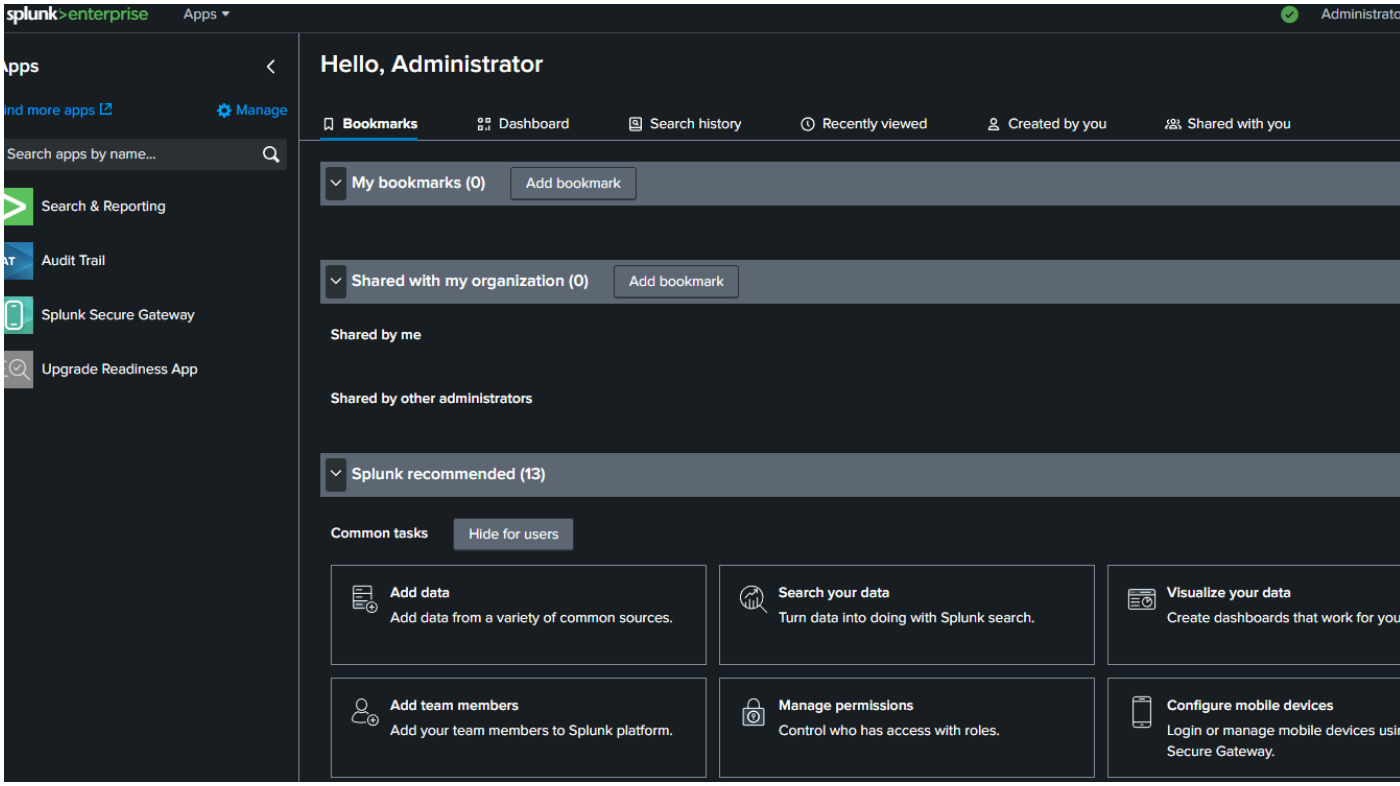
Going to the Splunk website download section, install the Windows compatible installer file and complete the download process. Set a username and password for login and proceed to the login page. A Splunk service will be set up on port 8000 under the local machine. Hosted on 127.0.0.1:8000 / 10.0.0.2:8000



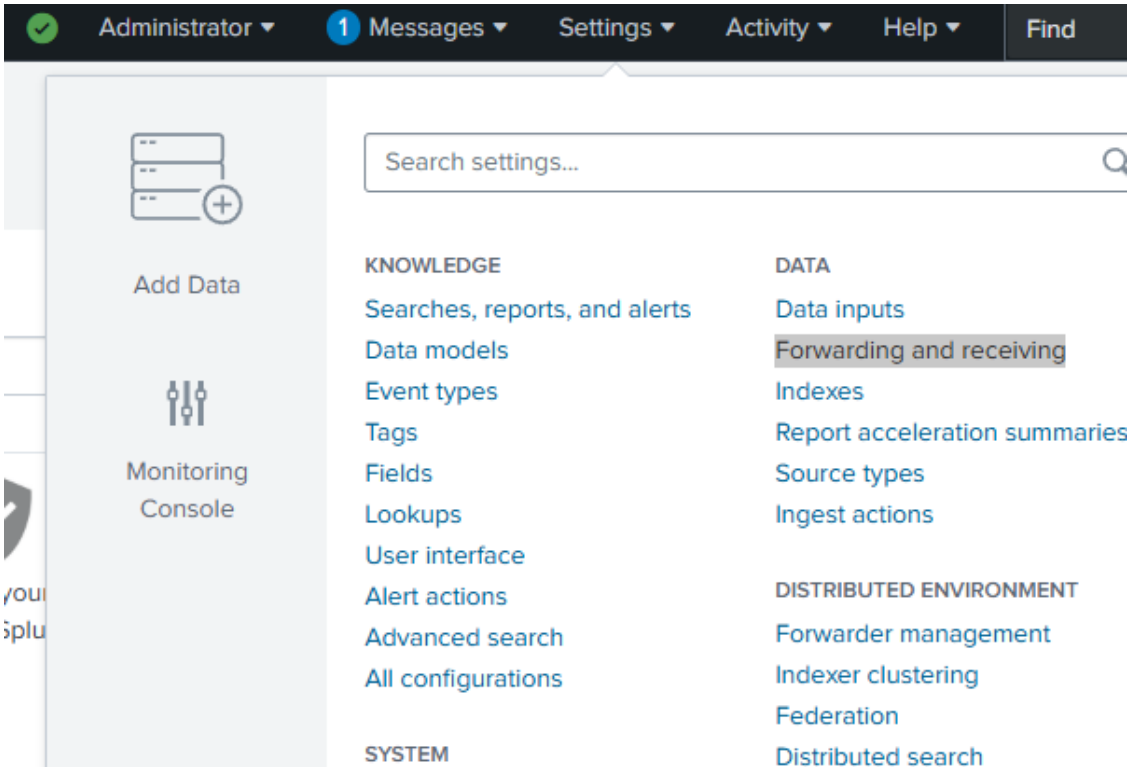
After installation, visit the service at http://127.0.0.1:8000 on the Windows 11 machine.



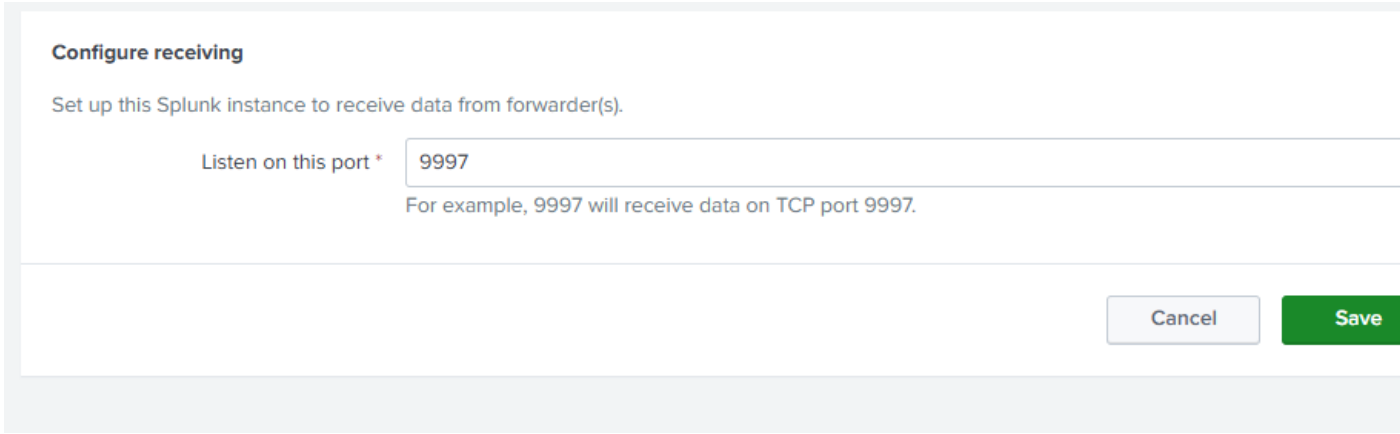
Once logged in, we are granted with the Splunk dashboard. Many options are available, but we will mostly go through data forwarding and index options, as well as Search and Reporting in Splunk.

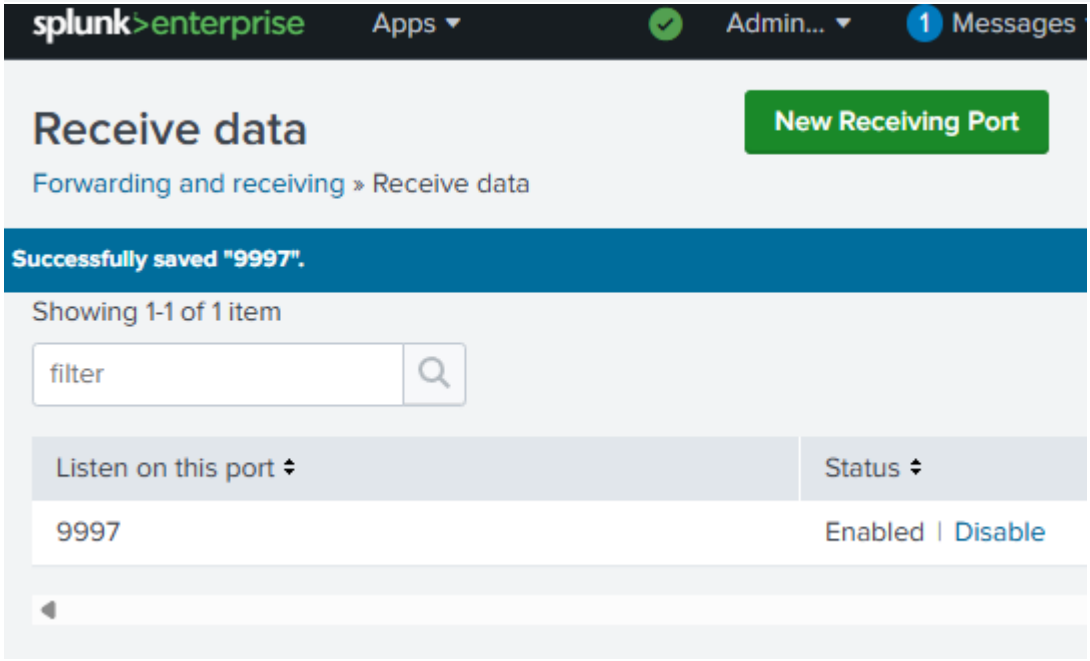


In the “Settings” section at top right corner, select “Forwarding and Receiving”



Set up receiving data port at port 9997 for incoming future logs being forwarded into Splunk:



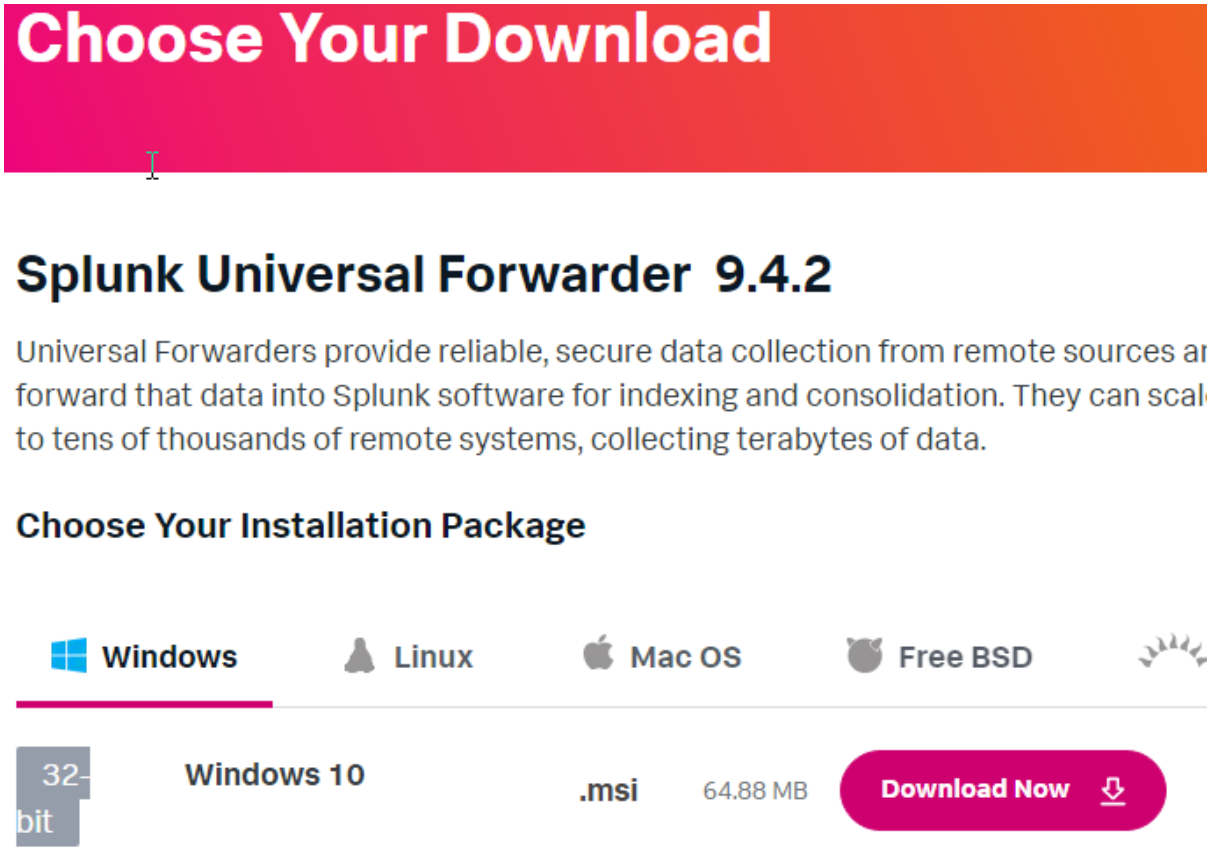


The receiving data port is now listening and waiting for data to be forwarded into Splunk.

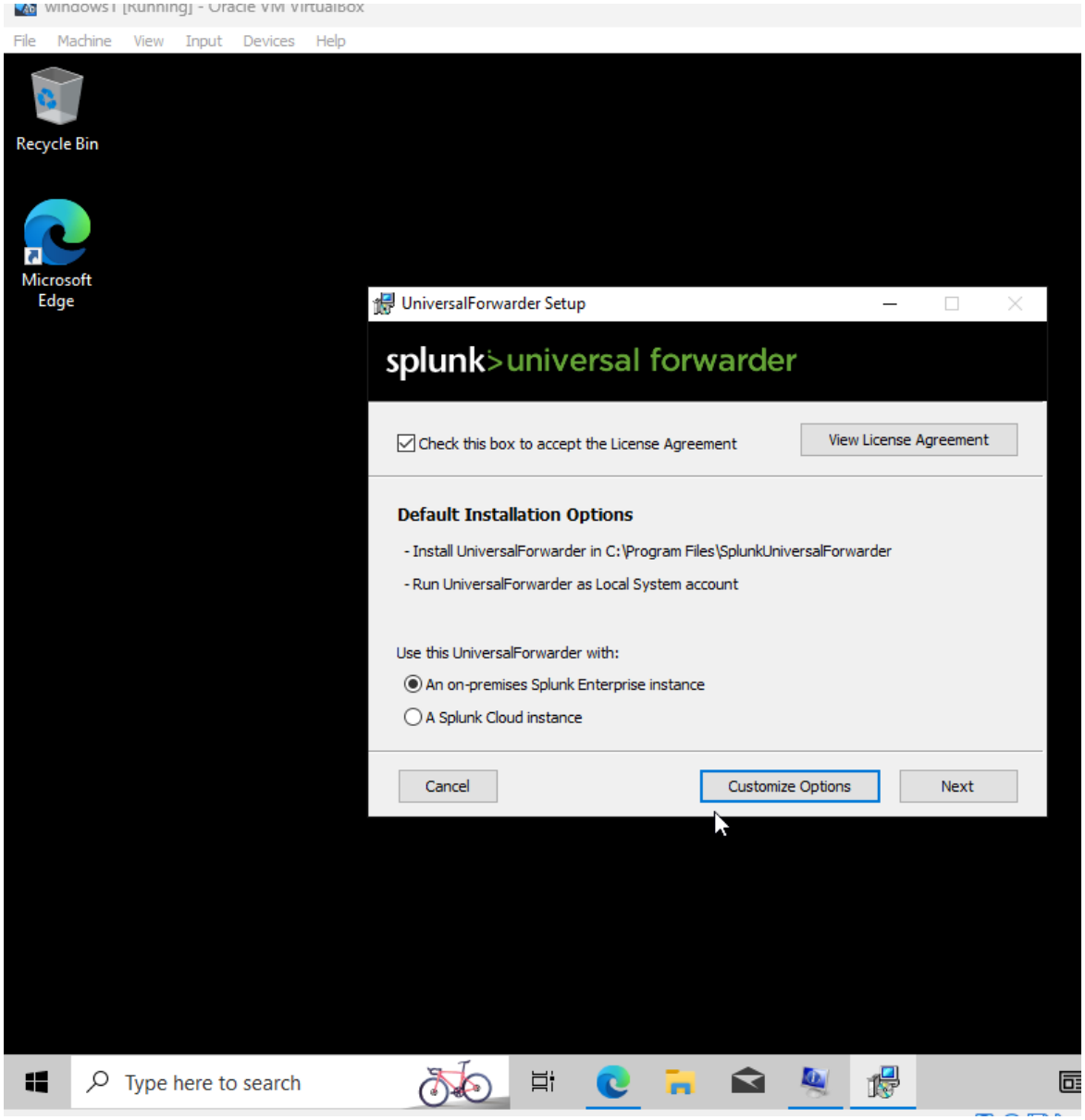
Log ingestion

1. **Windows 10 machine**

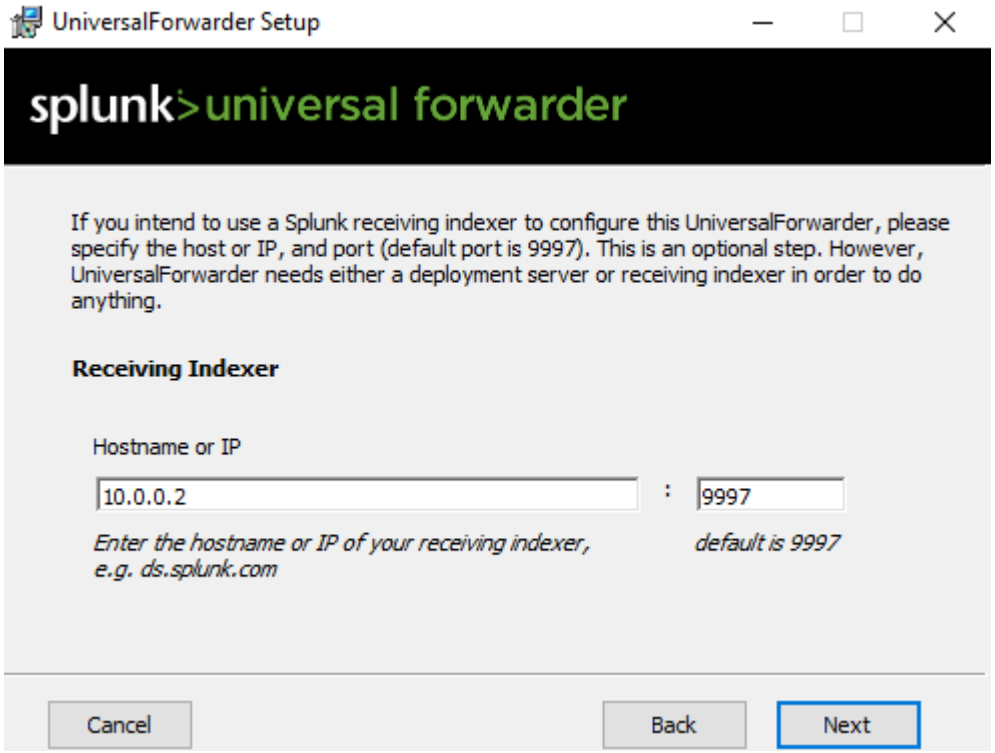
From here I will install the Splunk forwarder application on a Windows 10 machine configured to send relevant logs to the SIEM. The following screenshots are taken from the Win 10 machine.



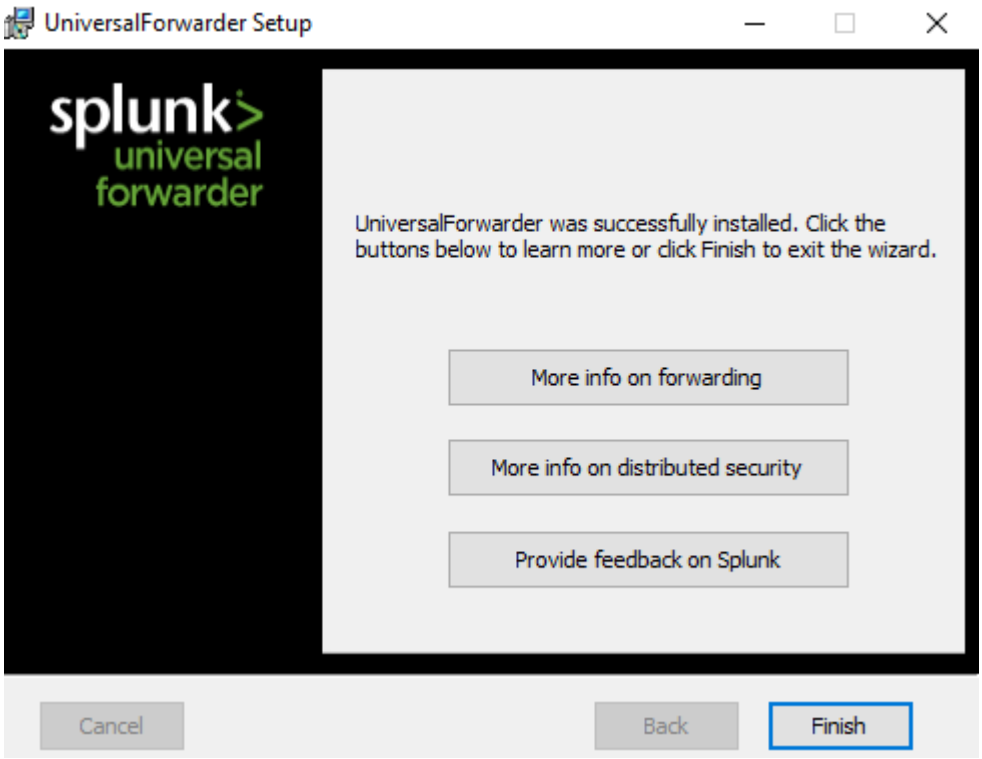
Install Splunk Universal Forwarder from the official website.



Continue the installation steps from the Universal Forwarder Setup application.



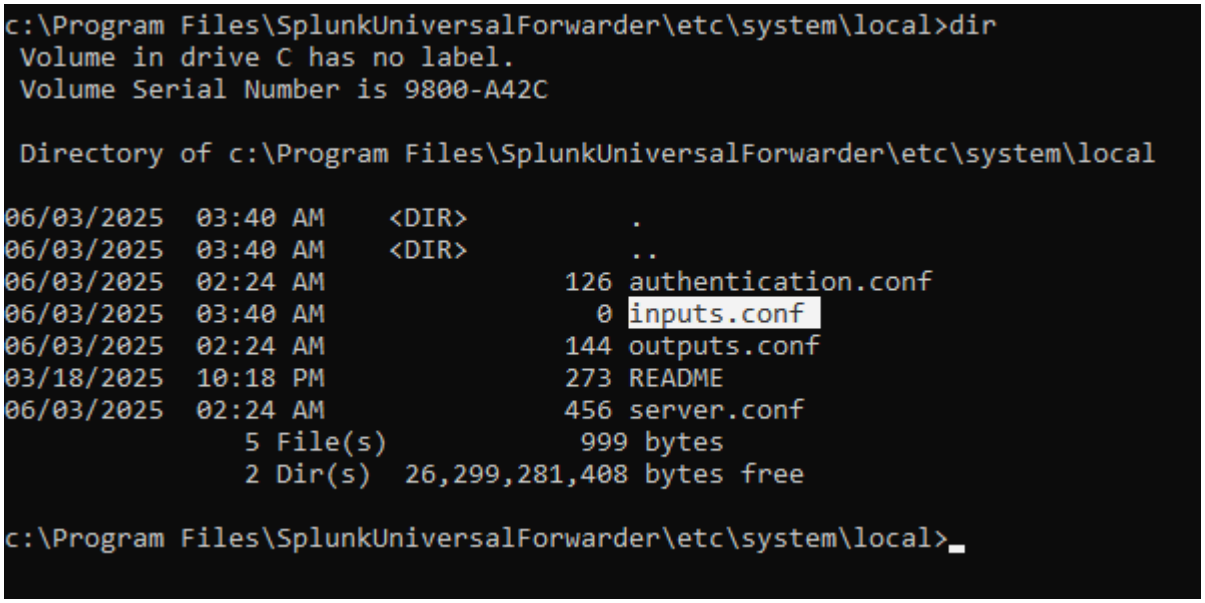
Setup the IP address and port of the Splunk server receiving data port at 10.0.0.2:9997



After installation, \SplunkUniversalForwarder\bin folder has been created. Visit the directory in cmd. Use command “splunk list forward-server" to see if Forwarder is active to the Splunk server:

```
c:\Program Files\SplunkUniversalForwarder\bin>splunk list forward-server
Your session is invalid.  Please login.
Splunk username: thomas
Password:
Active forwards:
        10.0.0.2:9997
Configured but inactive forwards:
        None

c:\Program Files\SplunkUniversalForwarder\bin>
```

The service is infact active and is ready to forward logs. Now we must configure which specific logs to be ingested into Splunk. This can be done by editing the “inputs.conf” file in directory \SplunkUniversalForwarder\etc\system\local



Select the inputs.conf file and edit it using NotePad:

*inputs.conf - Notepad

File Edit Format View Help

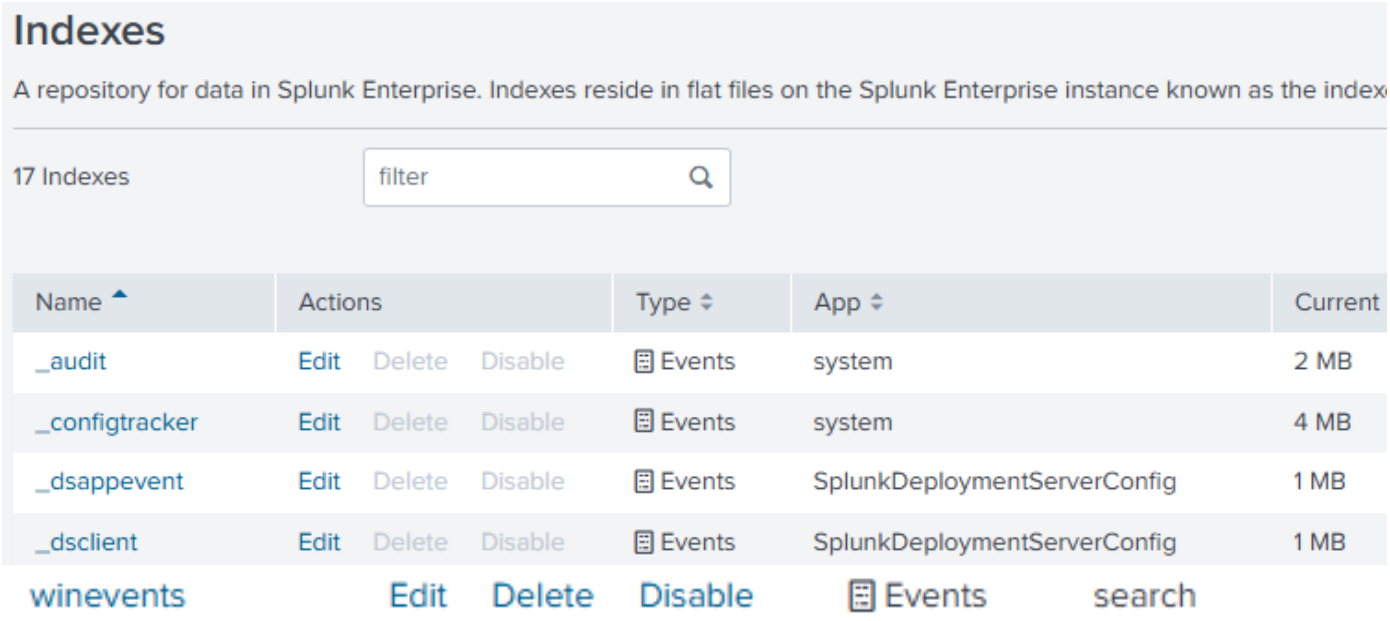
```
[WinEventLog://Security]
disabled = 0
sourcetype = security
index = winevents

[WinEventLog://System]
disabled = 0
sourcetype = system
index = winevents


[WinEventLog://Setup]
disabled = 0
sourcetype = setup
index = winevents

[WinEventLog://ForwardedEvents]
disabled = 0
index = winevents

[WinEventLog://Microsoft-Windows-Sysmon/Operational]
disabled = 0
sourcetype = sysmon
index = winevents
renderXml = true

[monitor://C:\Windows\System32\LogFiles\Firewall\pfirewall.log]
disabled = 0
sourcetype = firewall
index = winevents
```

Add in the following logs to monitor:

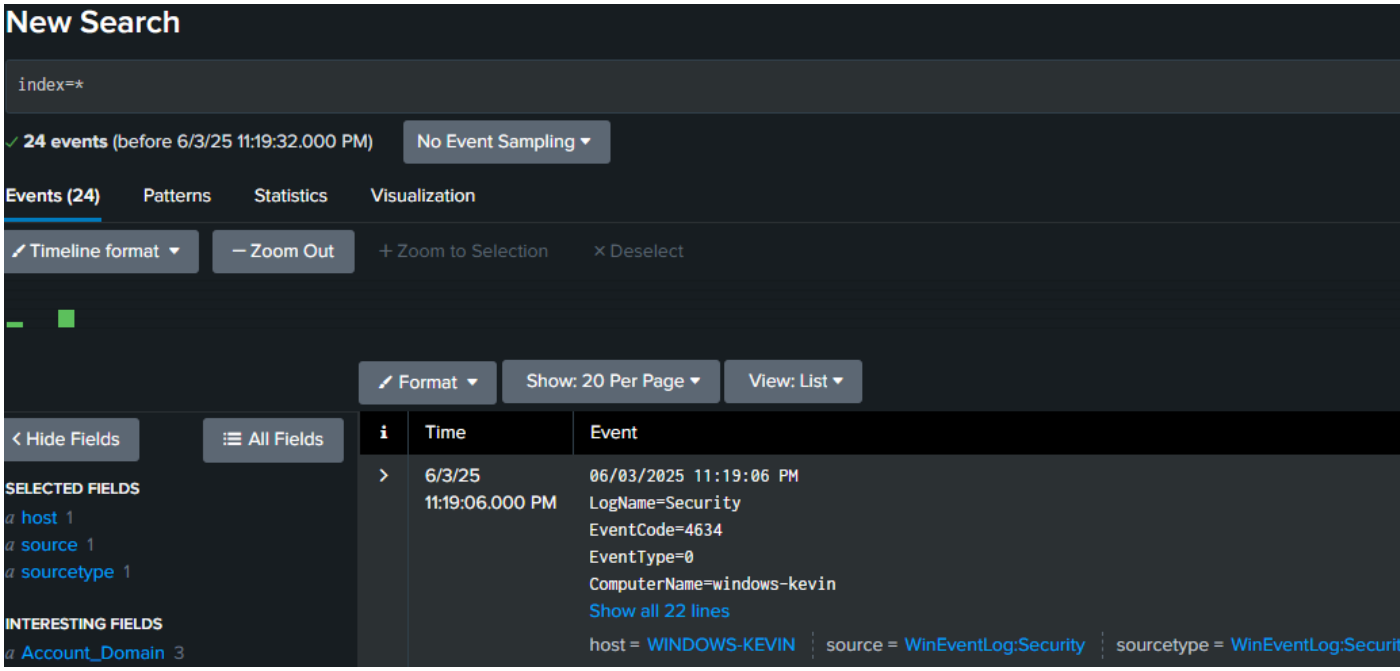
- Security event Logs
- System event Logs
- Setup event Logs
- Forwarded Events
- Sysmon events Logs
- Firewall Log (pfirewall.log)

All the added logs are under a common index listed as “winevents”.

Restart Splunk forwarder service once the inputs.conf file has been edited and saved.



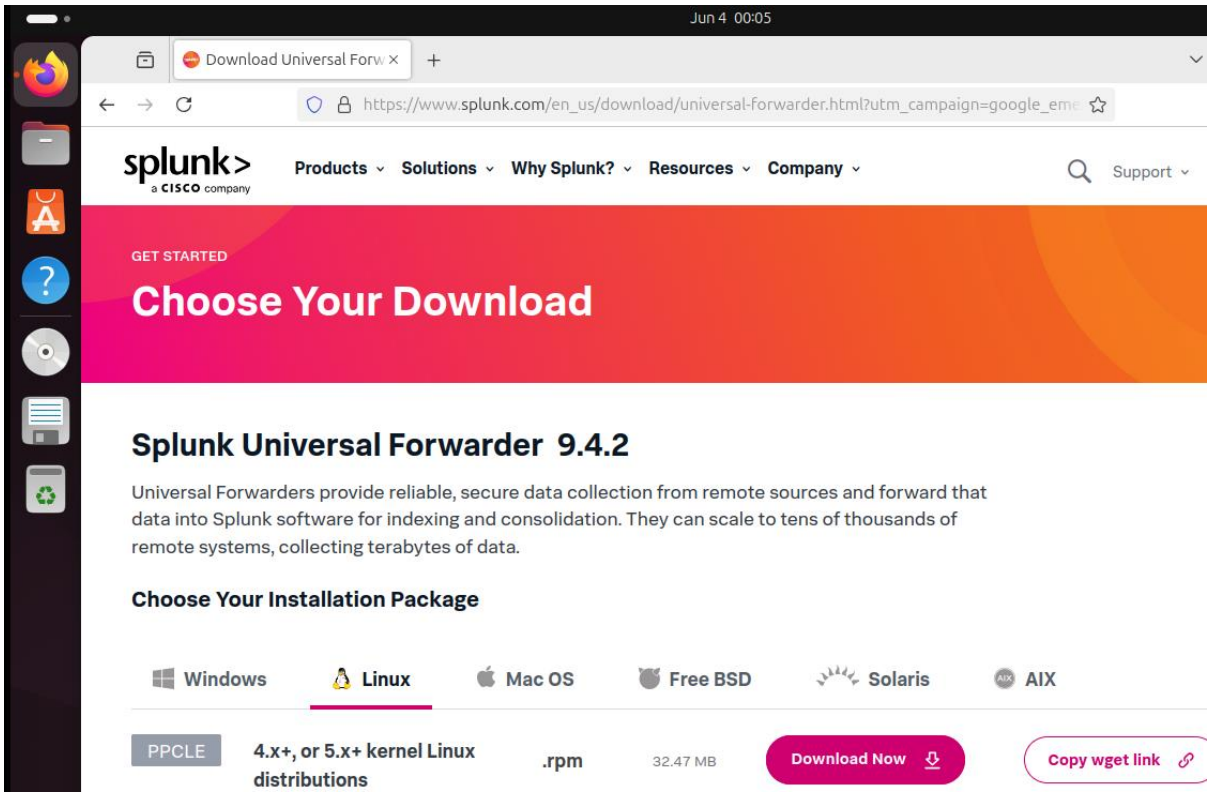
In Splunk settings, click on “Indexes” and add in a new index named as “winevents”. By doing this, ingested logs from the Windows 10 machine will be forwarded into the index “winevents”.



Ingestion was successful for the windows 10 machine, as logs have started to show up within the Splunk search.

2. Ubuntu Linux machine

Screenshots below have been taken from the Ubuntu desktop machine.



Download the compatible forwarder Linux based .tgz file from the official Splunk website

```
user@user-VMware-Virtual-Platform:~/Desktop$ sudo tar -xvzf splunkforwarder-9.4.
2-ec45dd264916-linux-ppc64le.tgz -C /opt
splunkforwarder/
```

```
user@user-VMware-Virtual-Platform:~$ sudo /opt/splunkforwarder/bin/splunk start --accept-license

This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup. Otherwise, you cannot log in.
Create credentials for the administrator account.
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username:
```

Starting the splunk forwarder service using the command line in bash.

```
rtual-Platform:/opt/splunkforwarder/bin$ sudo ./splunk add forward-server 10.0.0.2:999
7
Splunk username: thomas
Password:
Added forwarding to: 10.0.0.2:9997.
user@user-VMware-Virtual-Platform:/opt/splunkforwarder/bin$
```

Add the forward server of the Splunk server at 10.0.0.2:9997

The forward configuration is active to the Splunk server. Now we must configure which specific logs to be ingested into Splunk. This can be done by editing the “inputs.conf” file in directory

/SplunkUniversalForwarder/etc/system/local

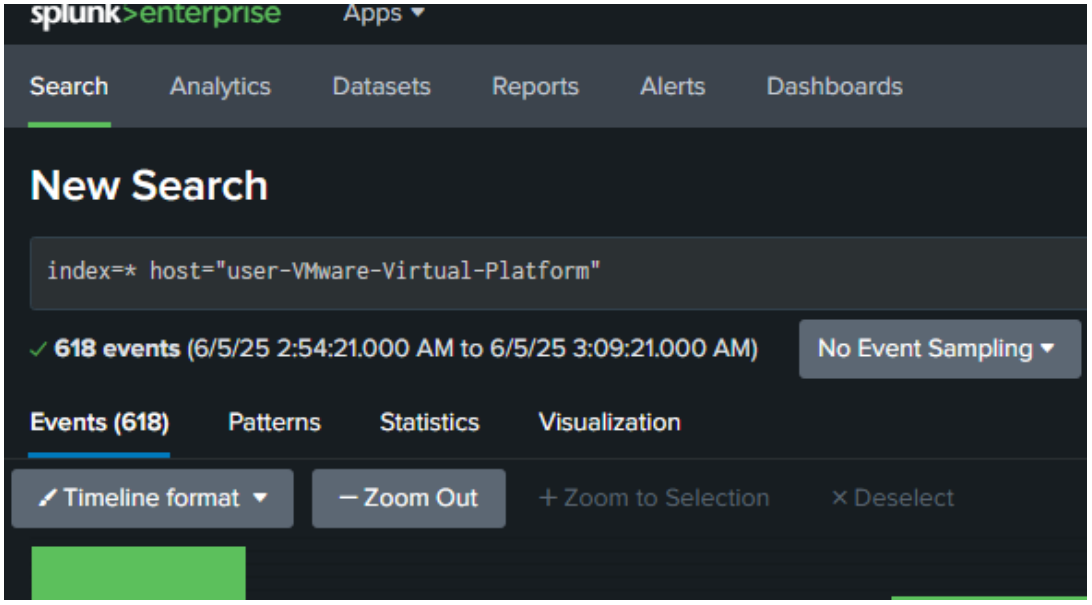
Edit the file of input.conf and add in the following relevant logs:

- /var/log/auth.log
- /var/log/syslog
- /home/*/.bash_history
- /var/log/bash_command.log
- /var/log/daemon.log
- /var/log/kern.log
- /var/log/samba/*

```
 GNU nano 7.2                          inputs.conf *
[monitor:///var/log/auth.log]
disabled = false
sourcetype = linux_secure
INDEX = ubuntu_logs

[monitor:///var/log/syslog]
disabled = false
sourcetype = syslog
INDEX = ubuntu_logs

[monitor:///home/*/.bash_history]
disabled = false
sourcetype = bash_history
INDEX = ubuntu_logs

[monitor:///var/log/bash_command.log]
disabled = false
sourcetype = realtime_bash
INDEX = ubuntu_logs

[monitor:///var/log/daemon.log]
sourcetype = daemon
INDEX = ubuntu_logs

[monitor:///var/log/kern.log]
sourcetype = kernel
INDEX = ubuntu_logs

[monitor:///var/log/samba/*]
sourcetype = samba
INDEX = ubuntu_logs
```

```
user@user-VMware-Virtual-Platform:/opt/splunkforwarder/bin$ sudo ./splunk restart
[sudo] password for user:
splunkd 6178 was not running.
Stopping splunk helpers...

Done.
```

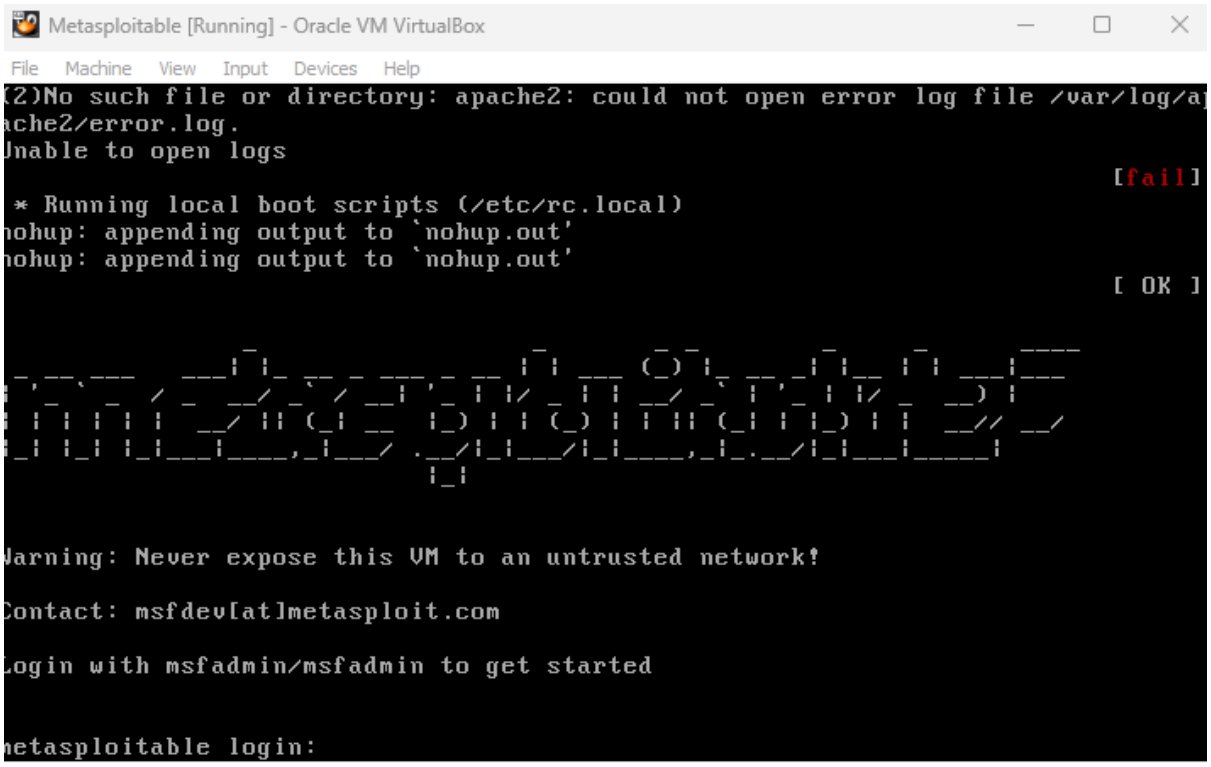
After editing the input.conf configuration file, restart the Splunk Forwarder service.

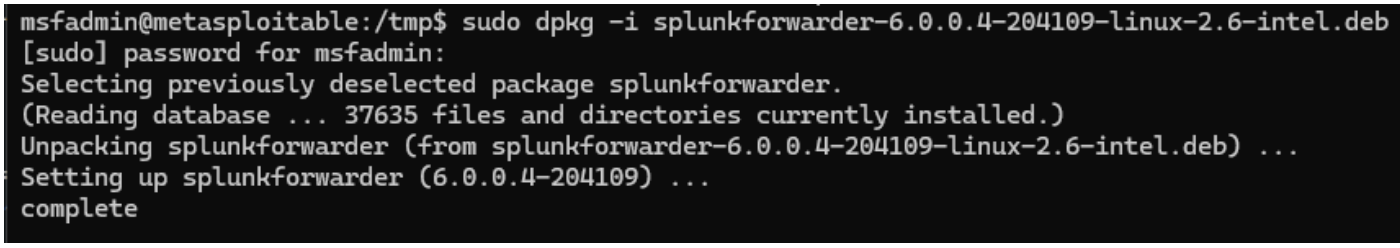


Log Ingestion was successful for the Ubuntu Desktop machine, as logs have started to show up within the Splunk search.

3. Metasploitable Linux server

Screenshots below have been taken from the Ubuntu desktop machine.





Install the Splunk Forwarder application on the Metasploitable Linux machine through command line in bash.

After installation, change directory to /opt/splunkforwarder/bin/ and start the service

```
msfadmin@metasploitable:/opt/splunkforwarder/bin$ sudo ./splunk start --accept-license

This appears to be your first time running this version of Splunk.

Splunk> Take the sh out of IT.

Checking prerequisites...
        Checking mgmt port [8089]: open
                Creating: /opt/splunkforwarder/var/lib/splunk
```

Start the Splunk Forwarder service and type in the following commands:

1. sudo ./splunk enable boot-start
2. sudo ./splunk add forward-server 10.0.0.2:9997 -auth admin:changeme
3. sudo ./splunk list forward-server

```
msfadmin@metasploitable:/opt/splunkforwarder/bin$ sudo ./splunk list forward-server
[sudo] password for msfadmin:
Your session is invalid.  Please login.
Splunk username: admin
Password:
Active forwards:
        10.0.0.2:9997
Configured but inactive forwards:
        None
```

Splunk forwarder is active through 10.0.0.2:9997

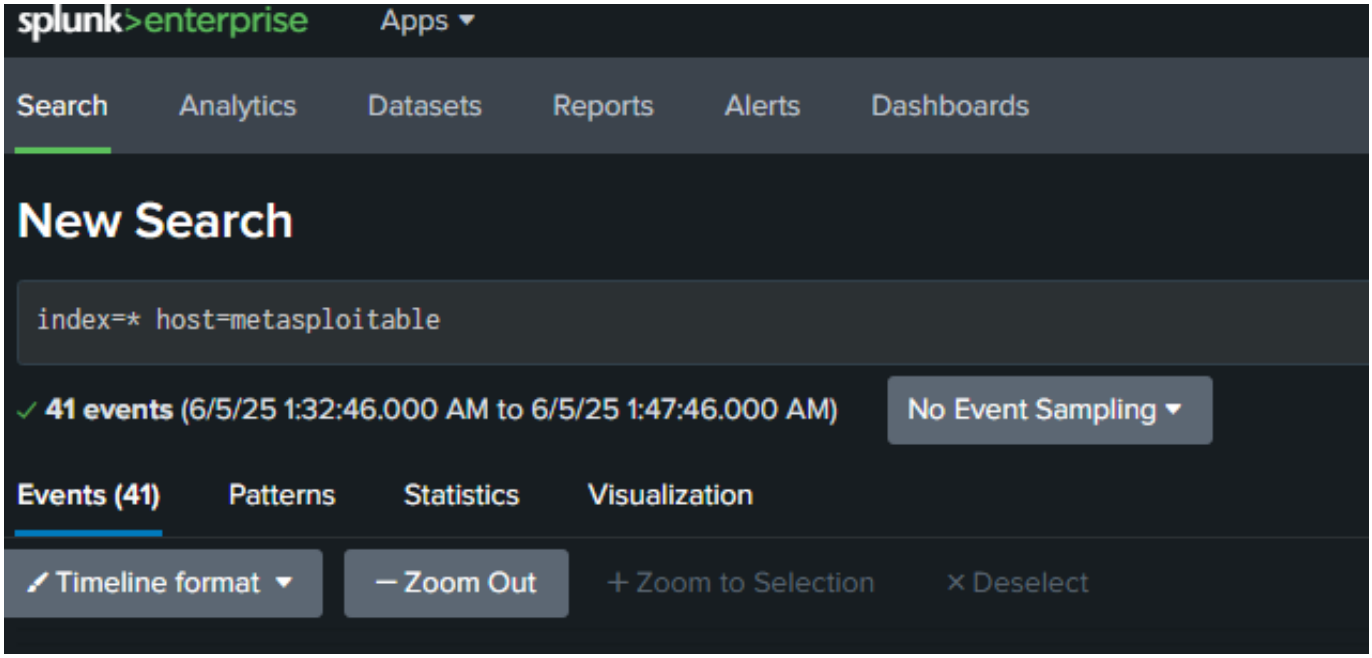
Now is the step to tell Forwarder what logs to monitor and forward to Splunk, edit “inputs.conf” file:

sudo nano /opt/splunkforwarder/etc/system/local/inputs.conf

```
 GNU nano 2.0.7                File: /opt/splunkforwarder/etc/system/local/inputs.conf

[default
host = metasploitable

[monitor:///var/log/auth.log]
disabled = false
sourcetype = linux_secure
index = metasploitable_logs

[monitor:///var/log/syslog]
disabled = false
sourcetype = syslog
index = metasploitable_logs

[monitor:///home/*/.bash_history]
disabled = false
sourcetype = bash_history
index = metasploitable_logs

[monitor:///var/log/bash_command.log]
disabled = false
sourcetype = realtime_bash
index = metasploitable_logs

[monitor///var/log/daemon.log]
sourcetype = daemon
index = metasploitable_logs

[monitor:///var/log/kern.log]
sourcetype = kernel
index = metasploitable_logs

[monitor:///var/log/samba/]
sourcetype = samba
index = metasploitable_logs
```

Screenshot above lists the relevant logs we want to monitor and forward to Splunk.

```
msfadmin@metasploitable:/opt/splunkforwarder/bin$ sudo /opt/splunkforwarder/bin/splunk restart
Stopping splunkd...
Shutting down.  Please wait, as this may take a few minutes.
.
Stopping splunk helpers...

Done.
```



The log ingestion process was successful for the Metasploitable Linux machine, as can be viewed in Splunk.

4. **Darkhole 2 Linux server**

Screenshots below have been taken from the "Darkhole 2" Linux machine.

```
root@darkhole:/tmp# sudo dpkg -i /tmp/splunkforwarder-9.4.3-237ebbd22314-linux-amd64.deb
Selecting previously unselected package splunkforwarder.
(Reading database ... 109392 files and directories currently installed.)
Preparing to unpack .../splunkforwarder-9.4.3-237ebbd22314-linux-amd64.deb ...
```

Install the Splunk Forwarder application on the Metasploitable Linux machine through command line in bash. Then start the Splunk Forwarder service:

```
complete
root@darkhole:/tmp# sudo /opt/splunkforwarder/bin/splunk start --accept-license
```

```
root@darkhole:/opt/splunkforwarder/bin# sudo ./splunk add forward-server 10.0.0.2:9997 -auth thomas:Hardflip123
Warning: Attempting to revert the SPLUNK_HOME ownership
root@darkhole:/opt/splunkforwarder/bin#
```

Add the forwarder server at 10.0.0.2:9997

```
root@darkhole:/opt/splunkforwarder/bin# ./splunk list forward-server
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R splunkfwd:splunkfwd /opt/splunkforwarder"
Active forwards:
        10.0.0.2:9997
```

The forwarder service is active and forwards to the Splunk server.

Add in the following relevant logs for monitoring into Splunk using listed index “darkhole2”:

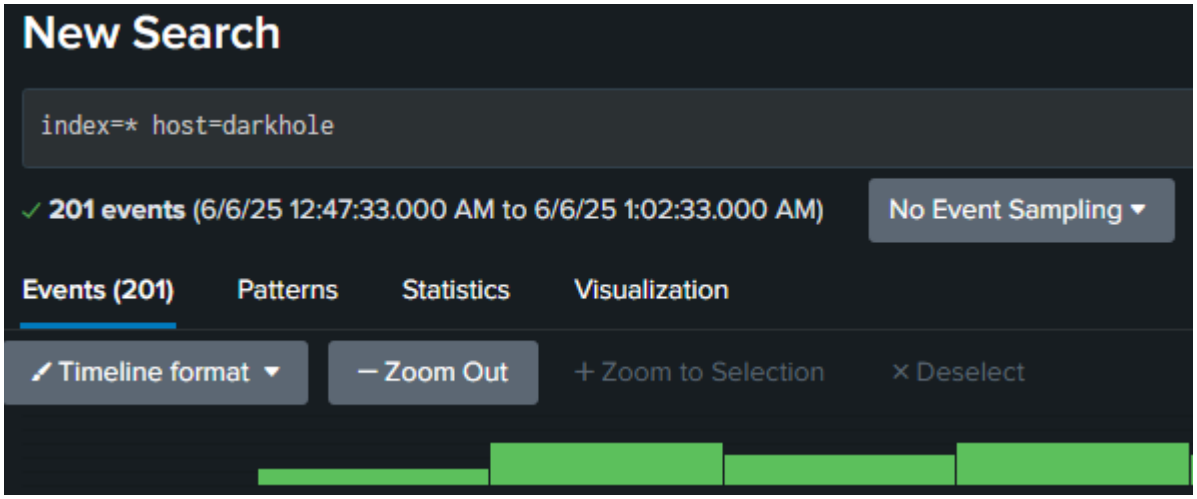
Sudo nano /opt/splunkforwarder/etc/system/local/inputs.conf

```
  GNU nano 4.8   /opt/splunkforwarder/etc/system/local/inputs.conf
[monitor:///var/log/auth.log]
disabled = false
sourcetype = linux_secure
index = darkhole2

[monitor:///var/log/syslog]
disabled = false
sourcetype = syslog
index = darkhole2

[monitor:///home/*/.bash_history]
disabled = false
sourcetype = bash_history
index = darkhole2

[monitor:///root/.bash_history]
disabled = false
sourcetype = bash_history
index = darkhole2
```

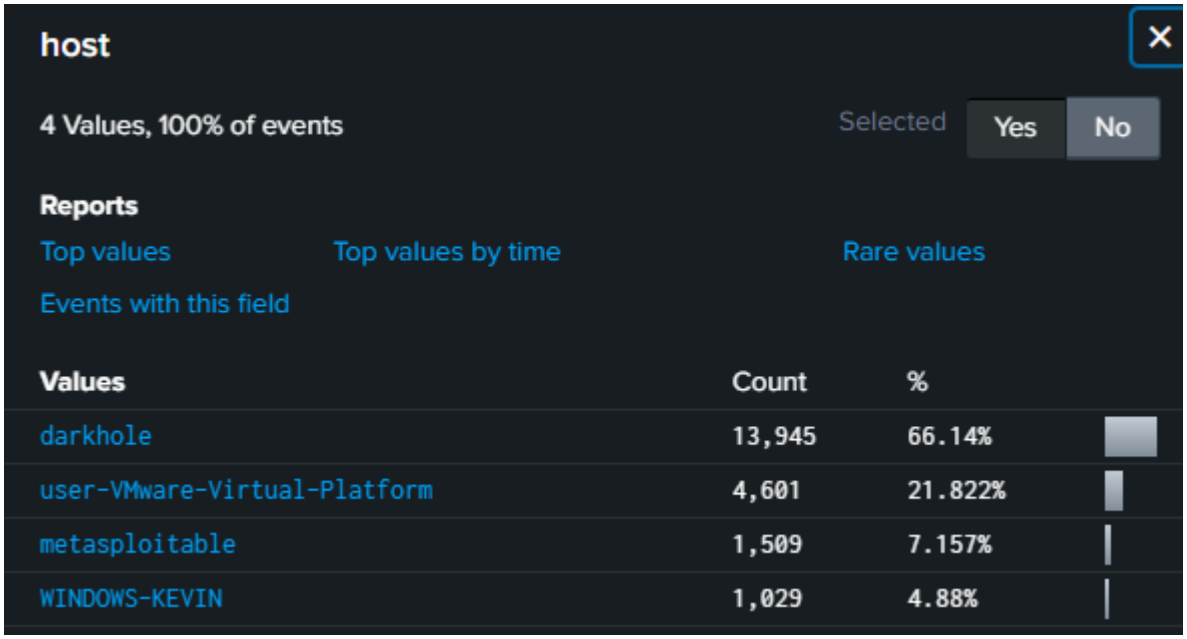
```
[monitor:///var/log/bash_command.log]
disabled = false
sourcetype = realtime_bash
index = darkhole2

[monitor:///var/log/daemon.log]
sourcetype = daemon
index = darkhole2

[monitor:///var/log/kern.log]
sourcetype = kernel
index = darkhole2

[monitor:///var/log/apache2/access.log]
disabled = false
sourcetype = apache_web
index = darkhole2
```

After configuring the inputs.conf file, restart the Splunk Forwarder service.

```
root@darkhole:/tmp# sudo /opt/splunkforwarder/bin/splunk restart
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R splunkfwd:splunkfwd /opt/splunkforwarder"
Stopping splunkd...
Shutting down.  Please wait, as this may take a few minutes.
```



The log ingestion process was successful for the Darkhole2 Linux machine, as can be viewed in Splunk.

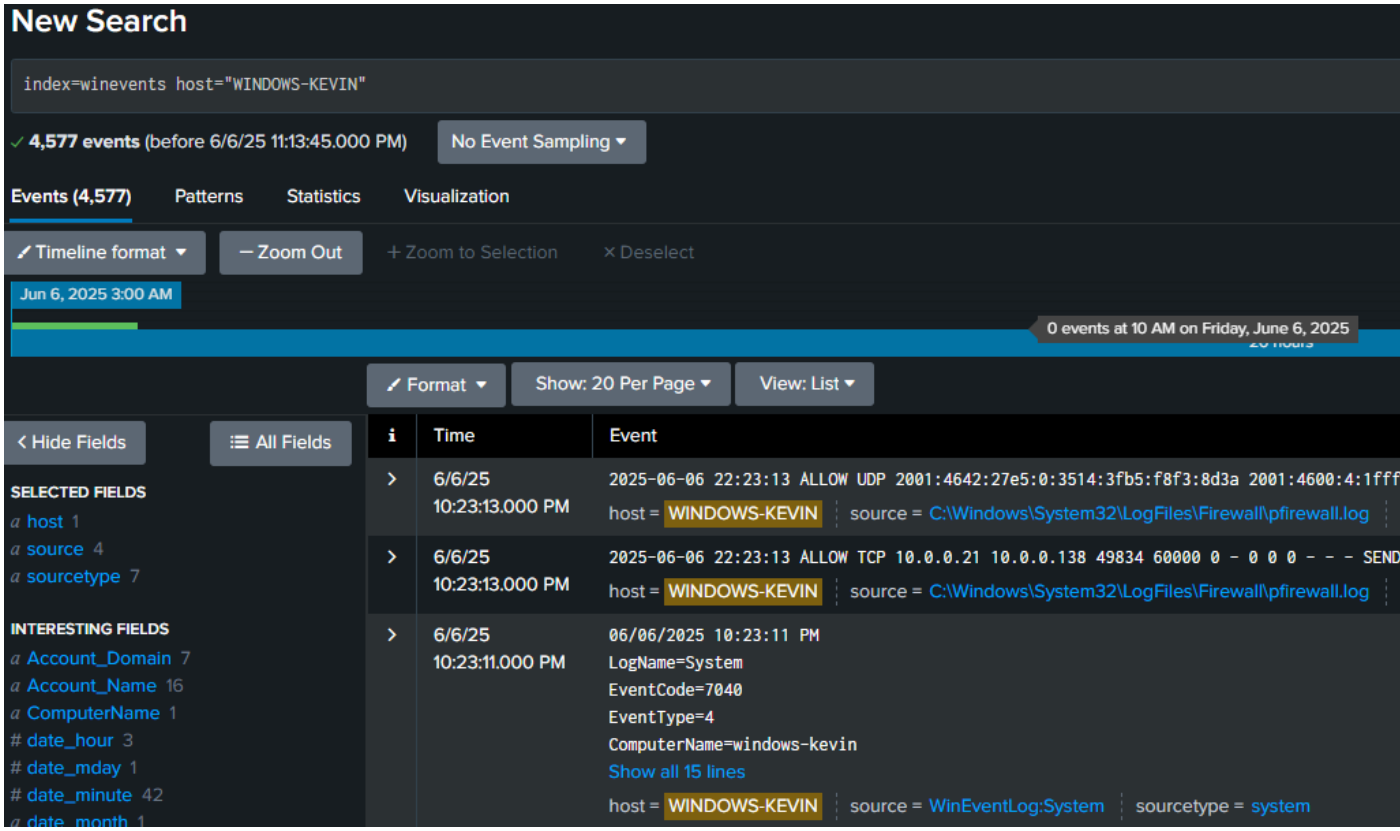


All 4 hosts of this lab have added log ingestion for monitoring to the Splunk SIEM platform.

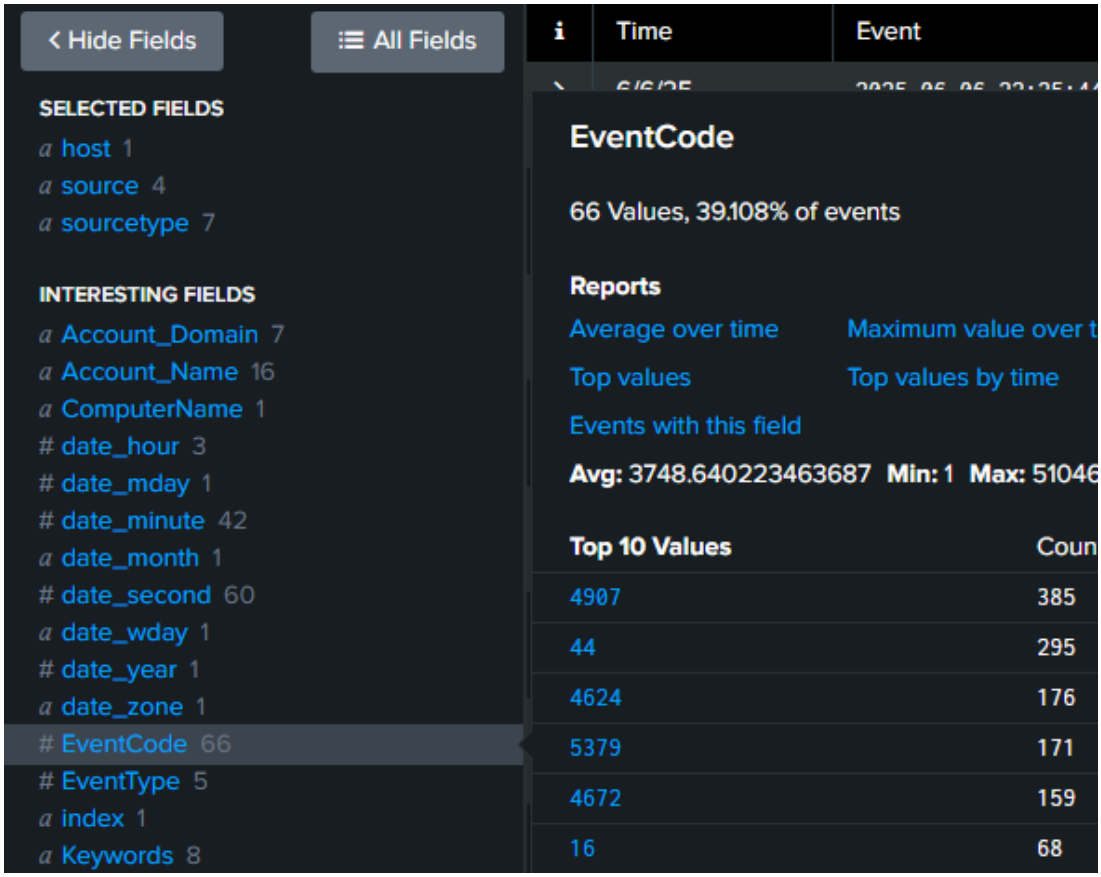
Basic Splunk search queries of logs

As all 4 hosts have been ingested into Splunk SIEM, let's try to show off some basic search queries for looking up potential logs in future objectives.

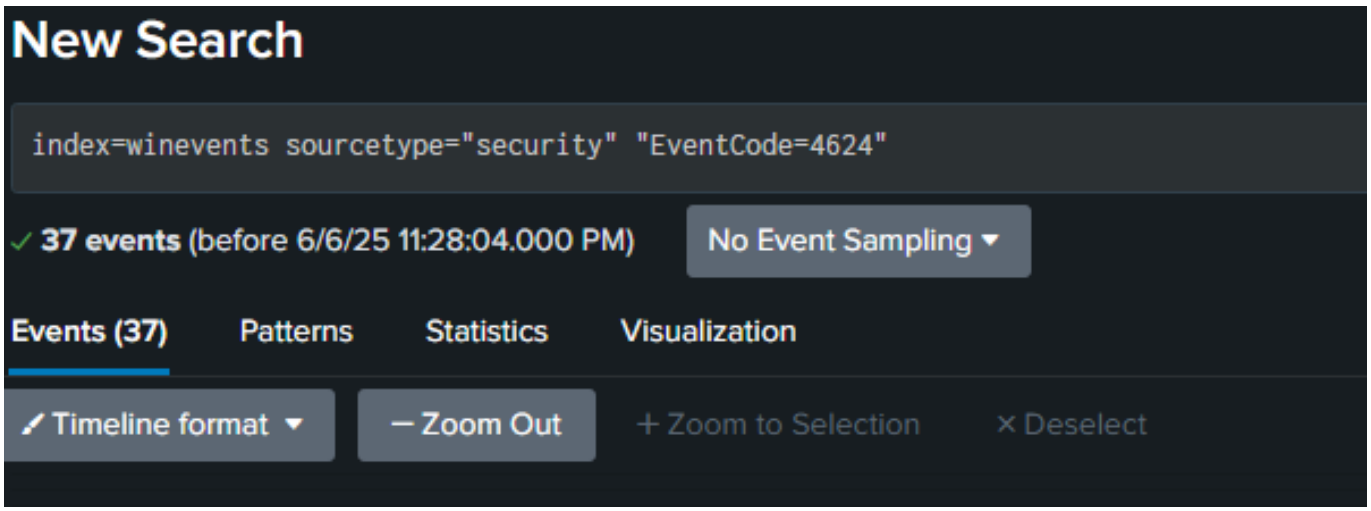
Select all logs from the last 24 hours.



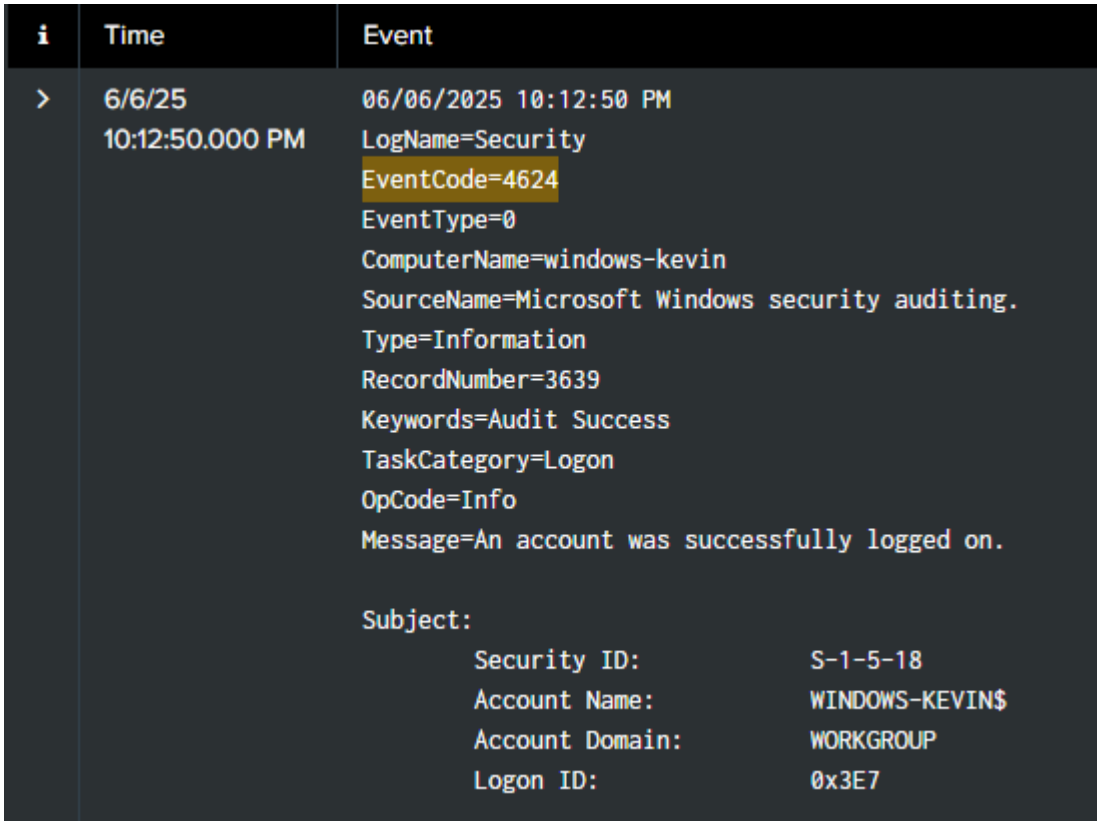
"Index=winevents host='WINDOWS-KEVIN'" will list events from the WIN 10 machine only.



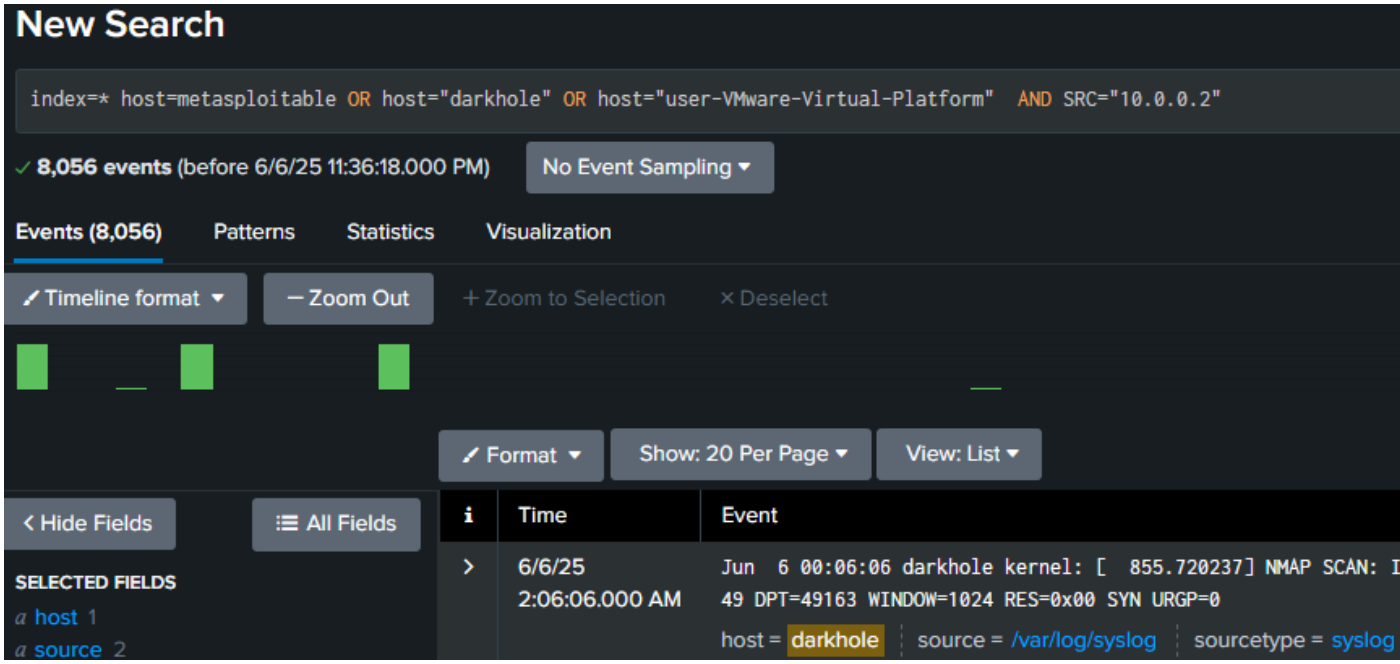
The “interesting fields” section lists potential search query recommendations to be used.



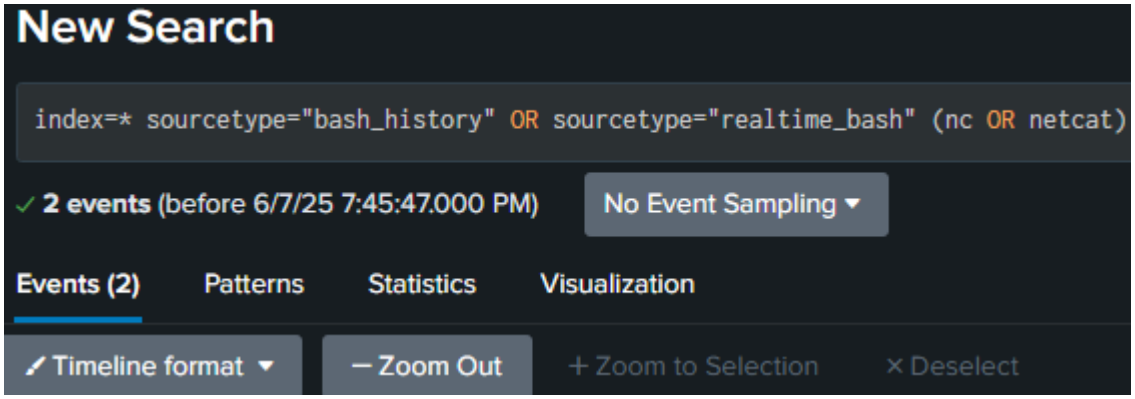
The search query “Index=winevents sourcetype=security EventCode=4624” will list events specifically from ”system” sourcetype logs.



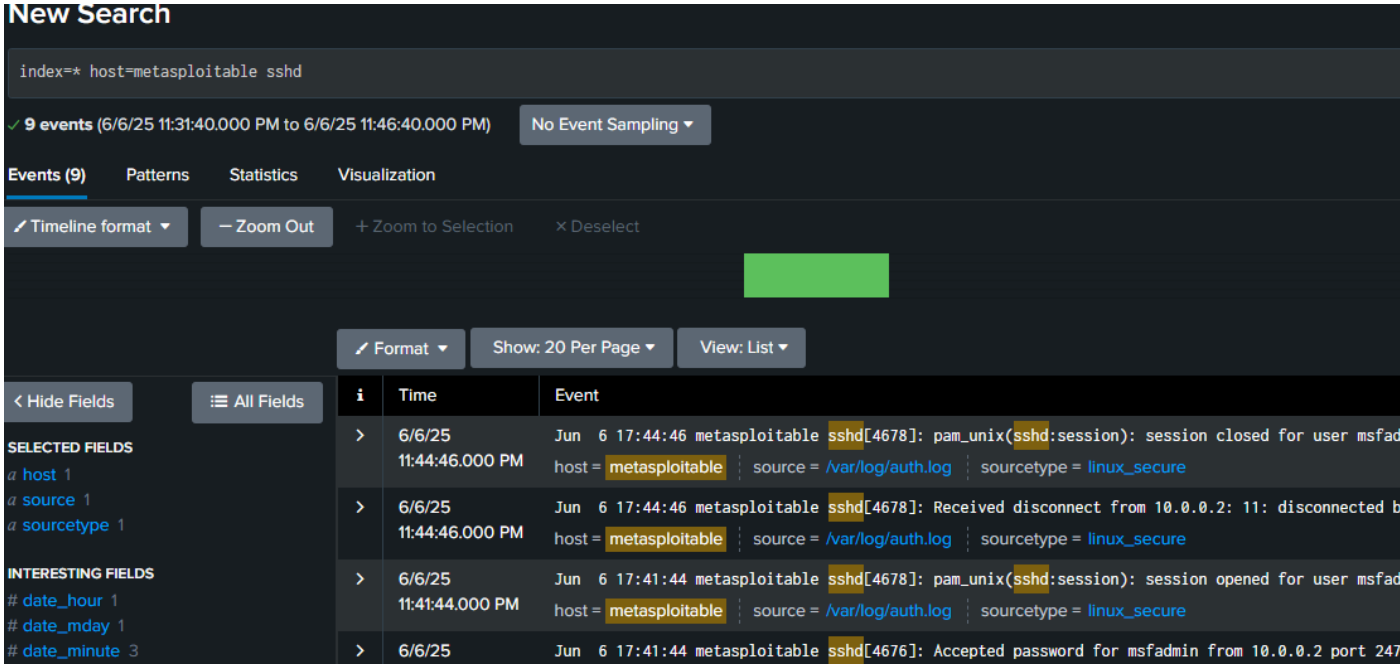
A close look at a random event log for event 4624 (Successful account log in)



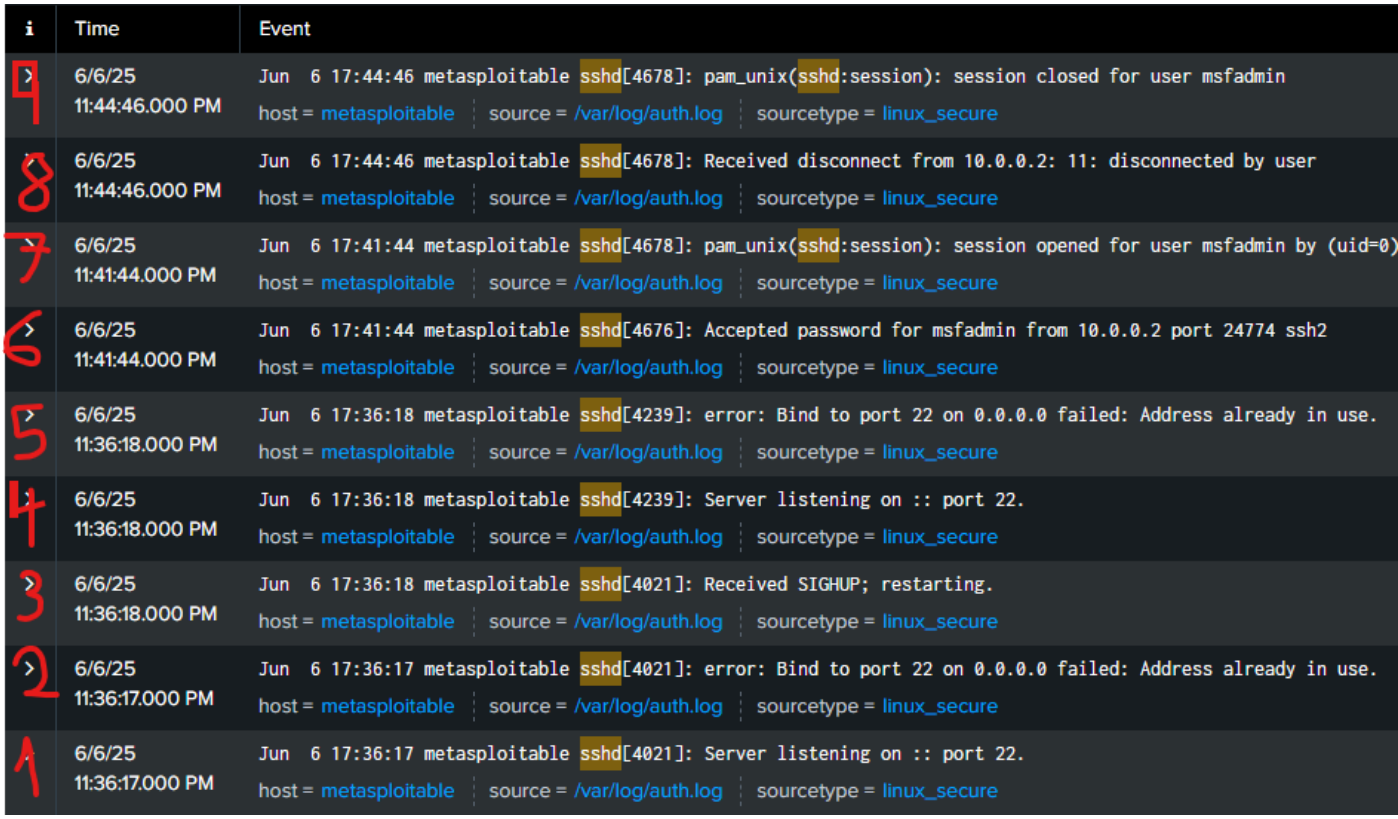
Another example looks at a search that includes all the Linux based machines and logs related to source ip ”10.0.0.2”



Index search to include logs from bash_history of Linux based machines, specifically of commands related to netcat commands. (signs of malicious intent)



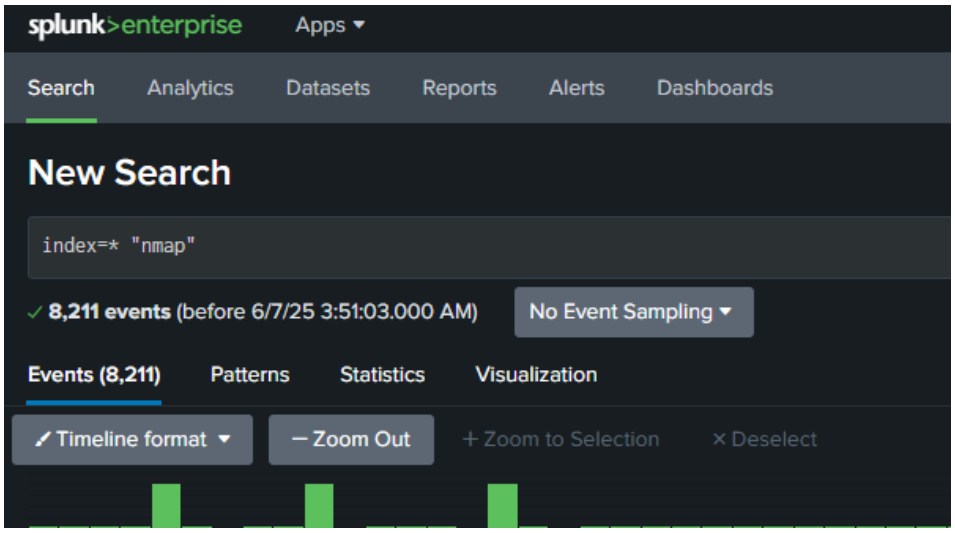
Index search to look into SSH usage based logs for sessions openings and closing of sessions for logged in users. This search shows a total of 9 events related to the Linux host ”metasploitable”.



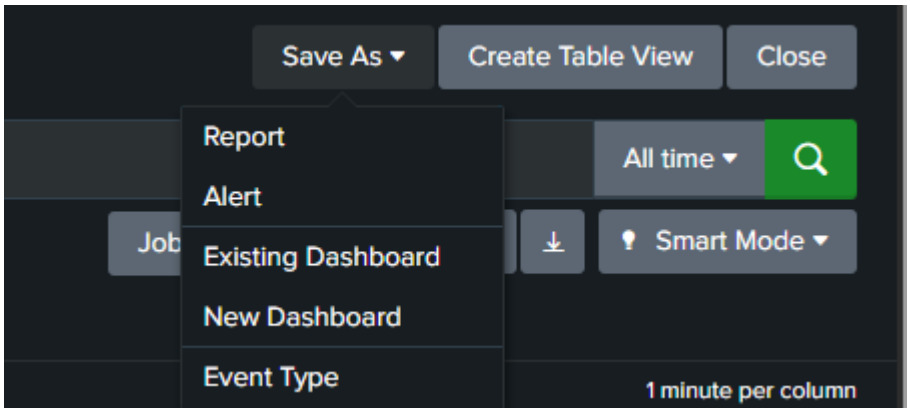
The log events provided from the previous search paints a picture of log event correlation is chronical order from first event to the ninth event. This event sequence shows how 2 login events for SSH failed due to the address already being in use. In the third attempt, user ”msfadmin” successfully logged in from IP ”10.0.0.2” and closed session shortly after.

Brief Creation of alert rules to identify Indicators of Compromise

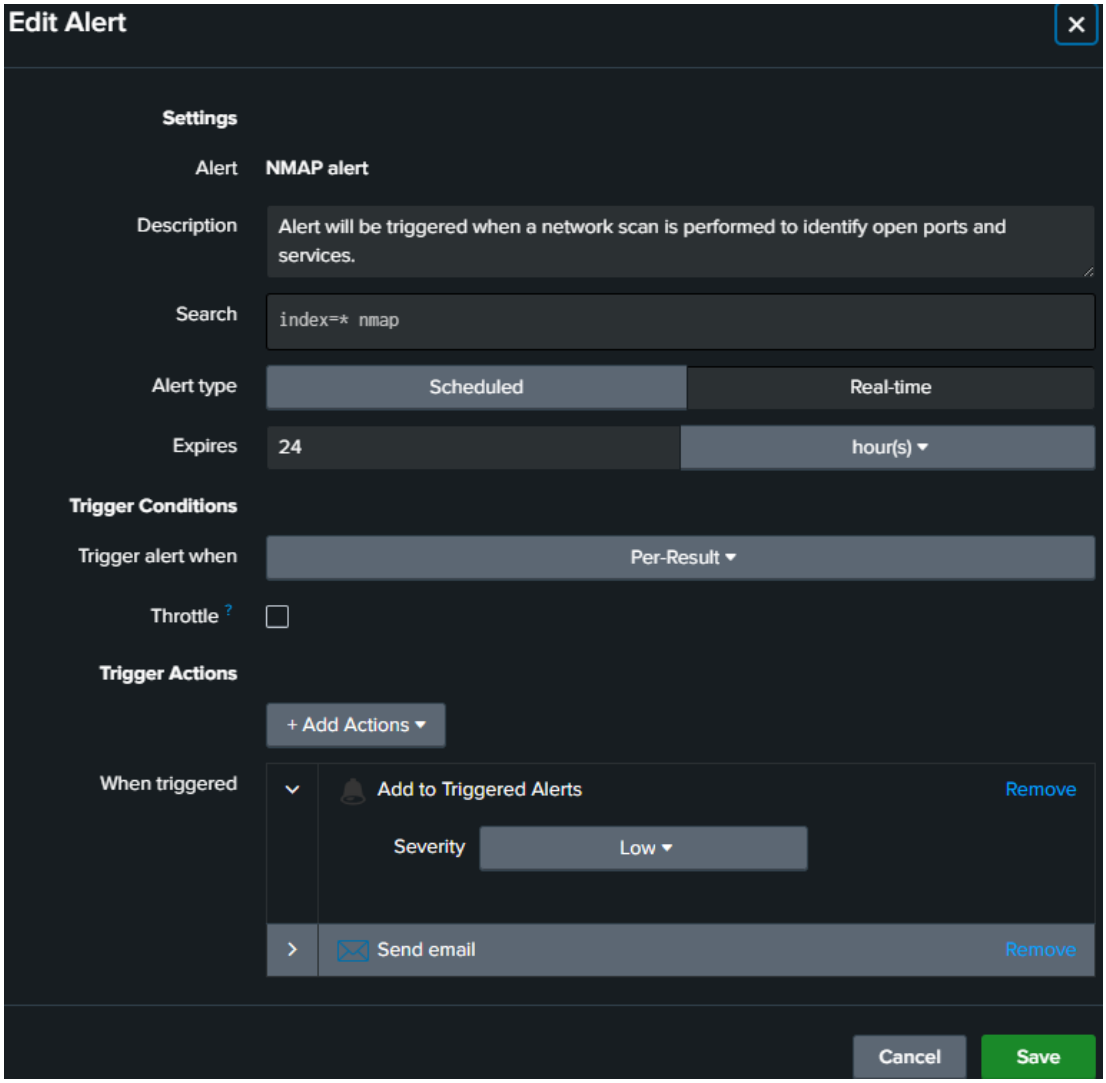
Creating alert rules in Spkunk is pretty straight forward. Simply type in the search query of a specific event or events and save the search query as a Alert. Name the alert type, give a description and select conditions of when and how the alert should be triggered.

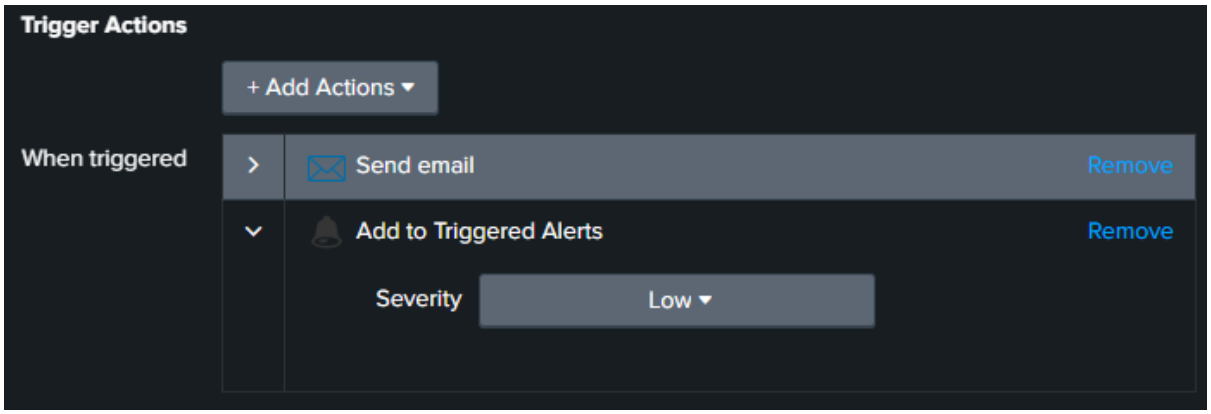


Type in a search query we wish to be a triggered alert, in this case searching for event related to the keyword “nmap” to indicate a network scan. In top right corner, select “save as”



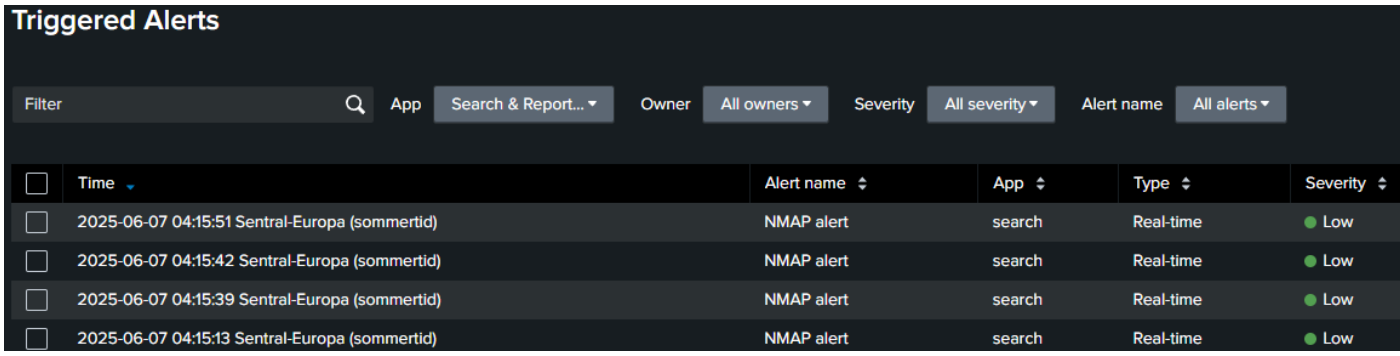
Save as “Alert”:



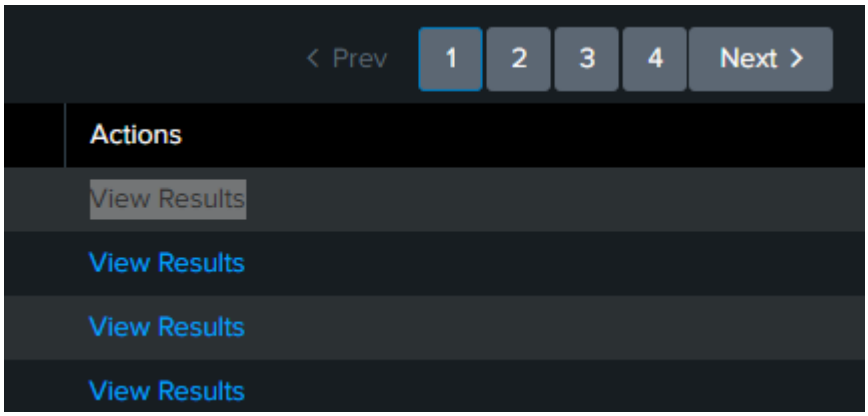


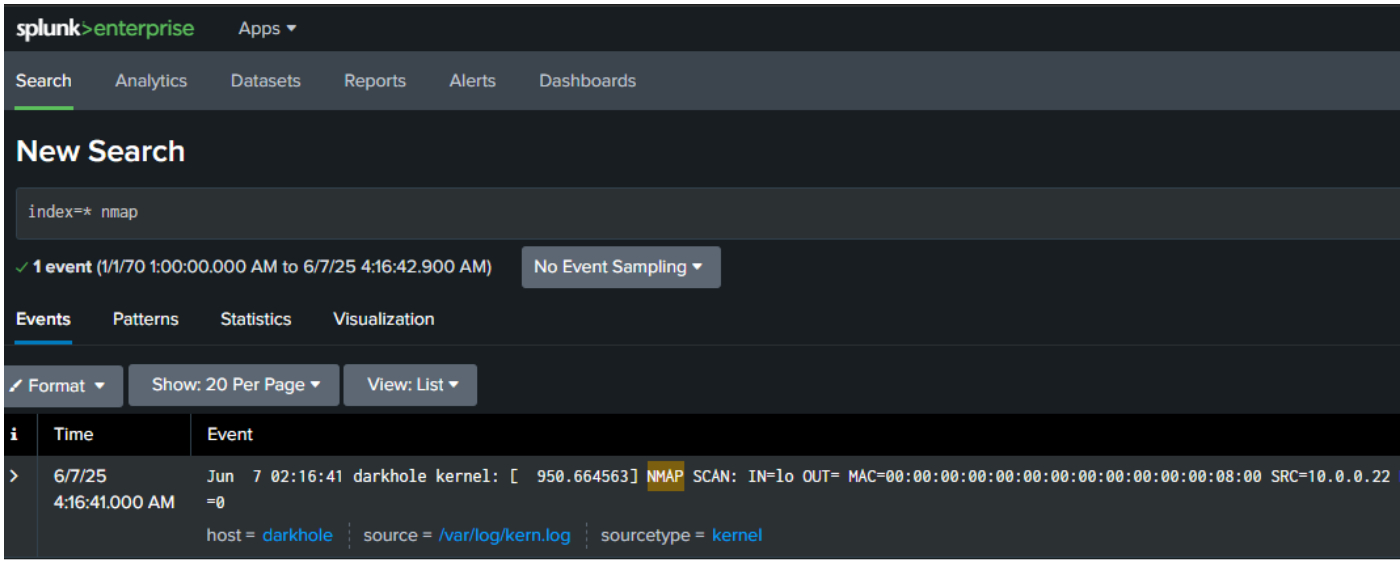
Save the alert with the description given and select event to be added to ”Triggered alerts” and to send an email to the Splunk users email account when event gets triggered.

The event has now been saved as a triggered alert as they come in. Simply simulate a nmap scan from any host and observe if event gets triggered and alerted.



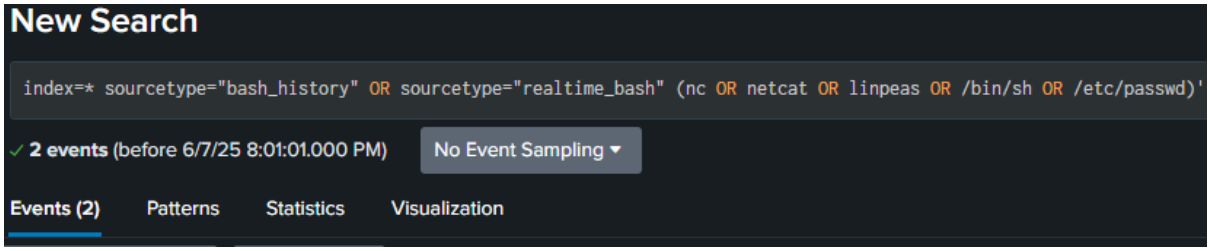
Tested successfully, a nmap scan was alerted in the ”Triggered Alerts” section of Splunk. Clicking on “view results” will specify the event:



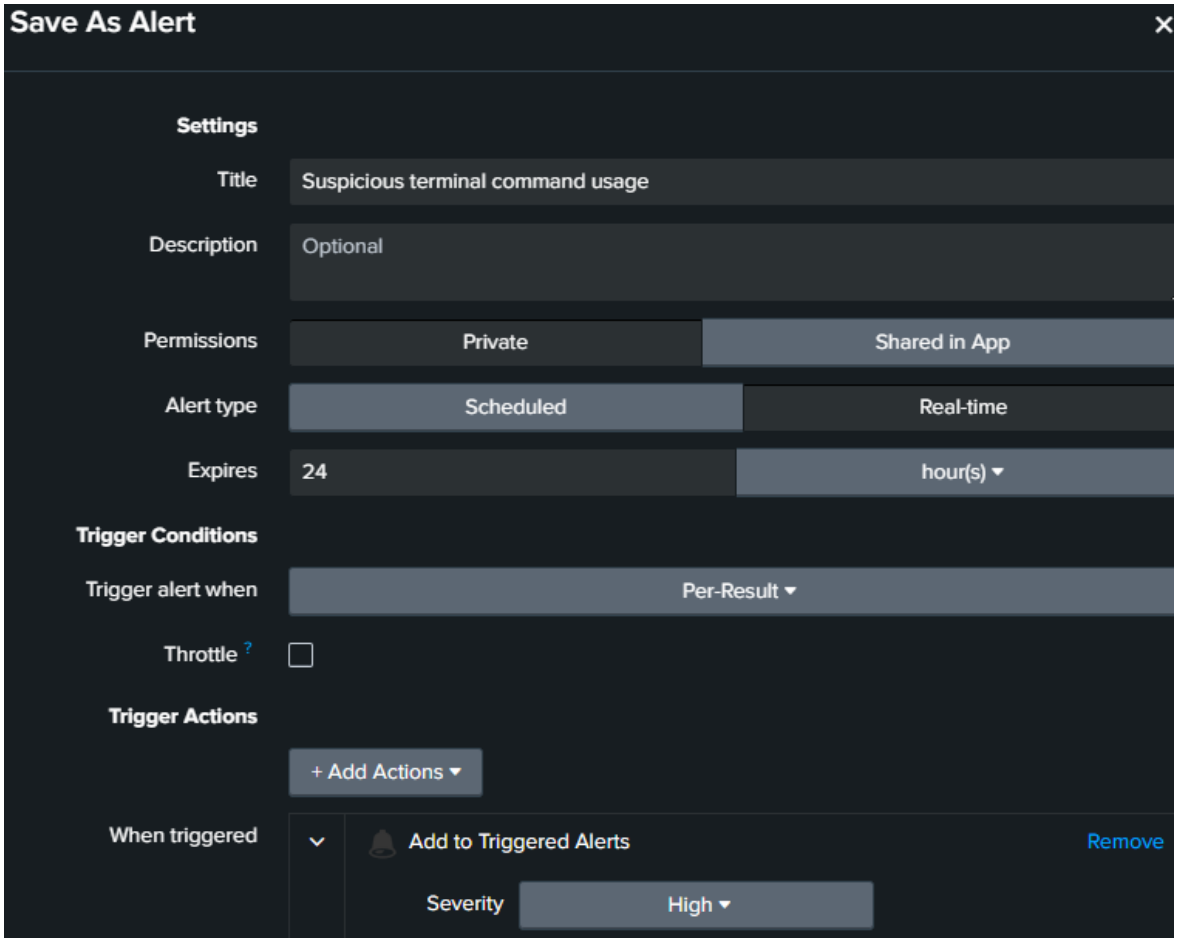


The specific event result showed an event detected as ”NMAP SCAN” on host ”darkhole” Linux machine coming from the IP 10.0.0.22.

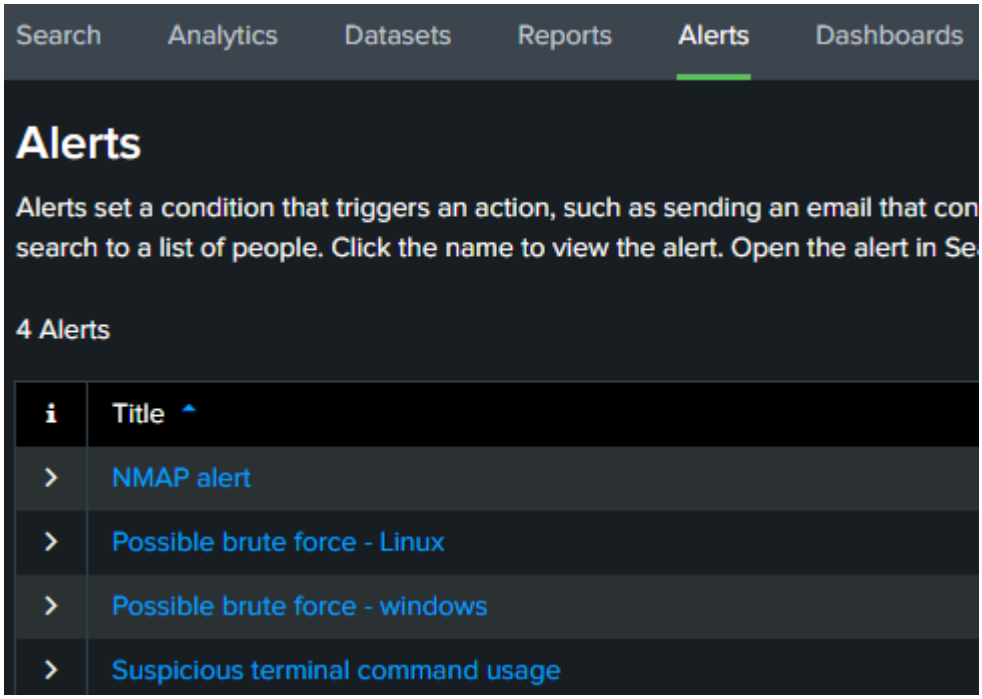
As another example, briefly type in a search query that detects suspiscious terminal command usage from any bash_history file of any host that includes usage of netcat, linPEAS or viewing sensitive files:



index= sourcetype="bash_history" OR sourcetype="realtime_bash" (nc OR netcat OR linpeas OR /bin/sh OR /etc/passwd)*



Save the search query as an Alert and give a title, set trigger conditions and add to triggered alerts.

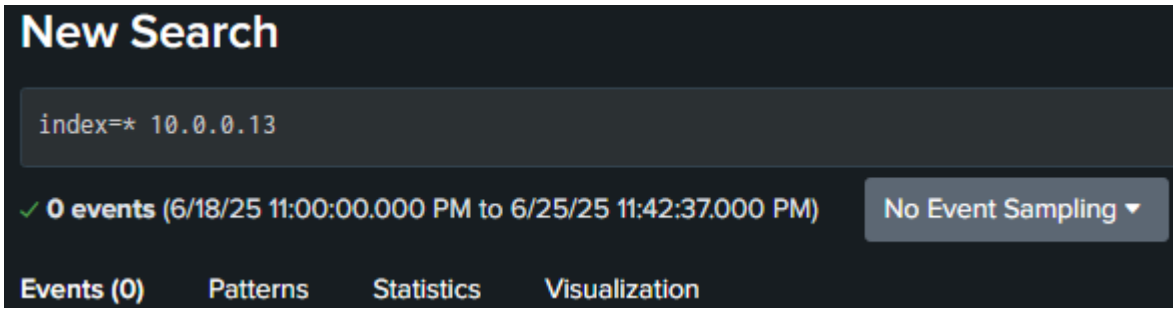


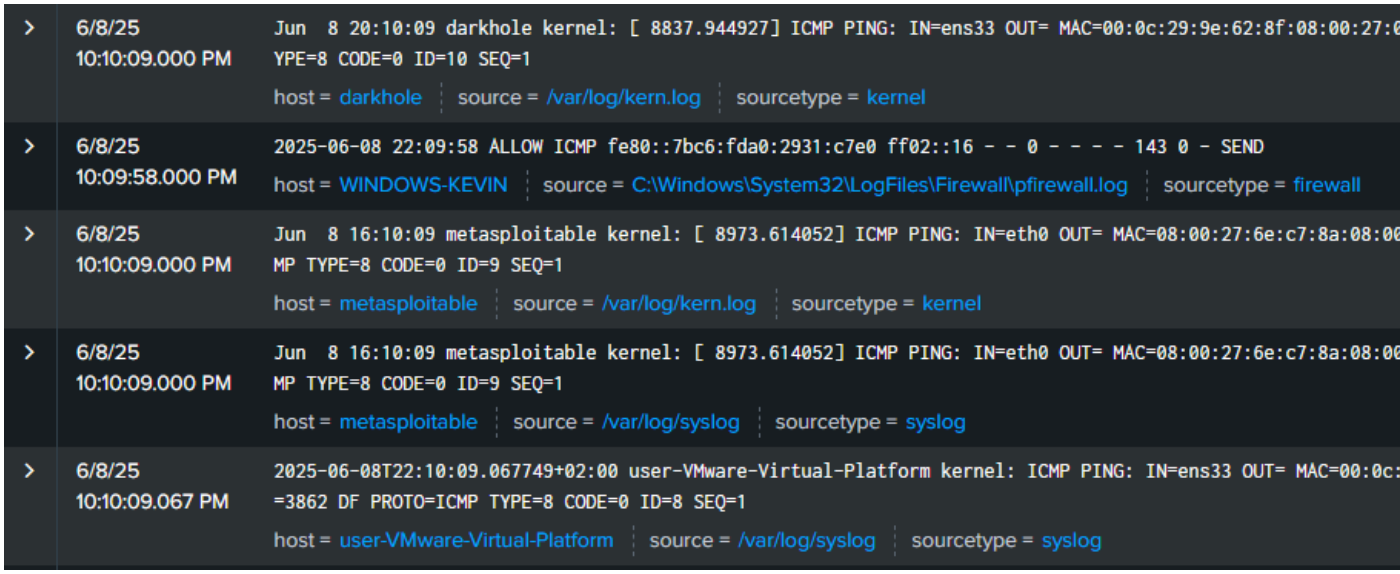
Saved 4 different alert types for examples of creating triggered alerts in Splunk.

Post-Incident Investigation of simulated attacks

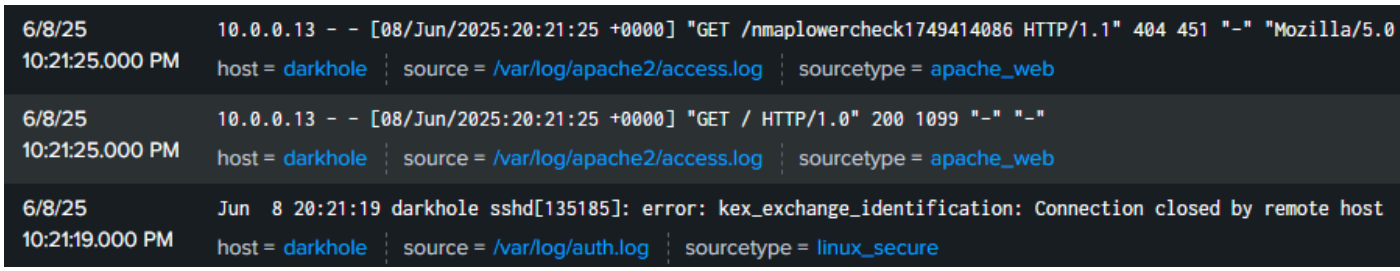
An attacker has gained access to our local network. The attacker's Kali Linux machine is attempting to identify other devices on local network and find vulnerable machines. The attacker initiates a series of reconnaissance activities aimed at discovering other devices within the same subnet to later compromise, gain initial access and attempt to escalate privileges with the goal of exfiltrating various files using netcat. Your job is to investigate the incident through the Splunk logs.

Attacker at IP address 10.0.0.13 (Kali Linux machine) did ping sweep of local network, ICMP packets was received on all 4 hosts monitored on Splunk within the same timestamp.

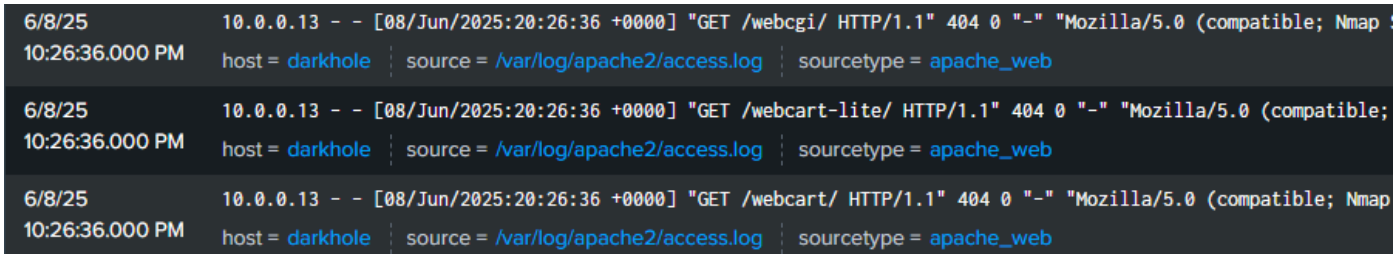




ICMP packets were received around the same timestamp, indicating it was done as a part of a ping sweep to detect live hosts within the same subnet sequence.



Next sequence of logs from the same IP address (10.0.0.13) include an attempted SSH connection, followed by some GET request to the Darkhole web service.

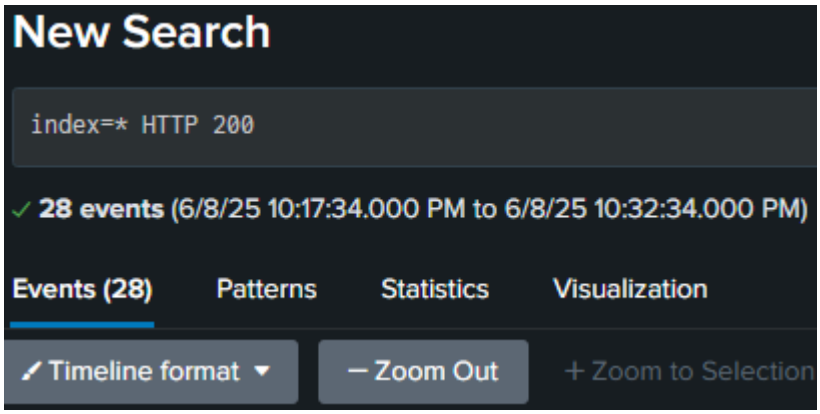


"GET /webcart/ HTTP/1.1" 404 0 "-" "Mozilla/5.0 (compatible; Nmap Scripting Engine; https://nmap.org/book/nse.html)

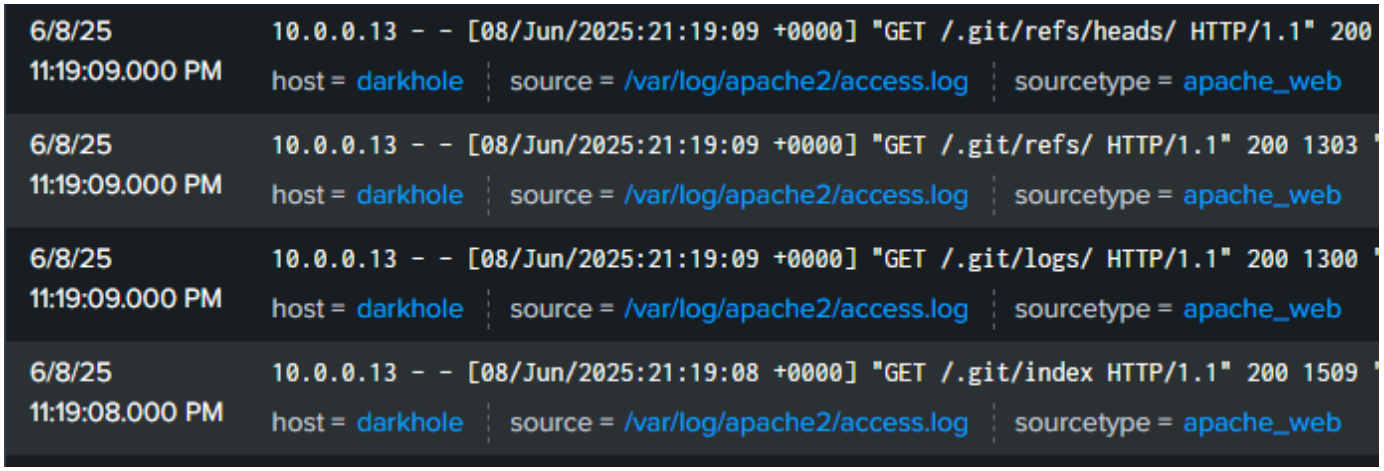
Several GET request was made to identify possible web service directories through enumeration. The requests are labelled with “compatible: Nmap”, indicating that the enumeration is part of a NMAP scan.

The labelled requests are noted as “404” web codes, meaning the directory do not exist on the application.

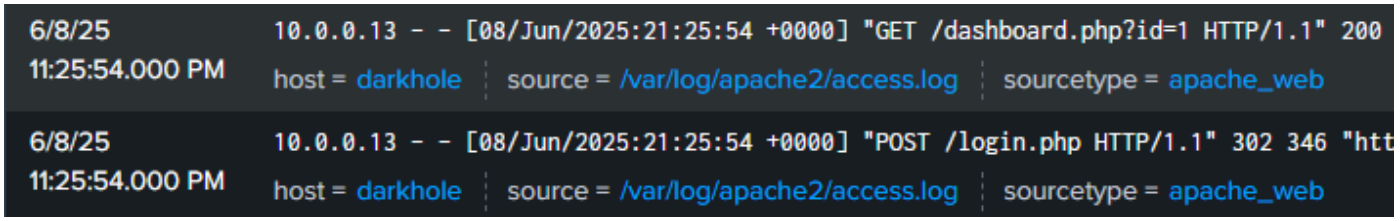
Identify HTTP GET requests by narrowing search down to HTTP 200 codes, meaning the directory does exist on the application and successfully responded to the request. By viewing the logs related to the HTTP service, we can take a closer look at their activities.



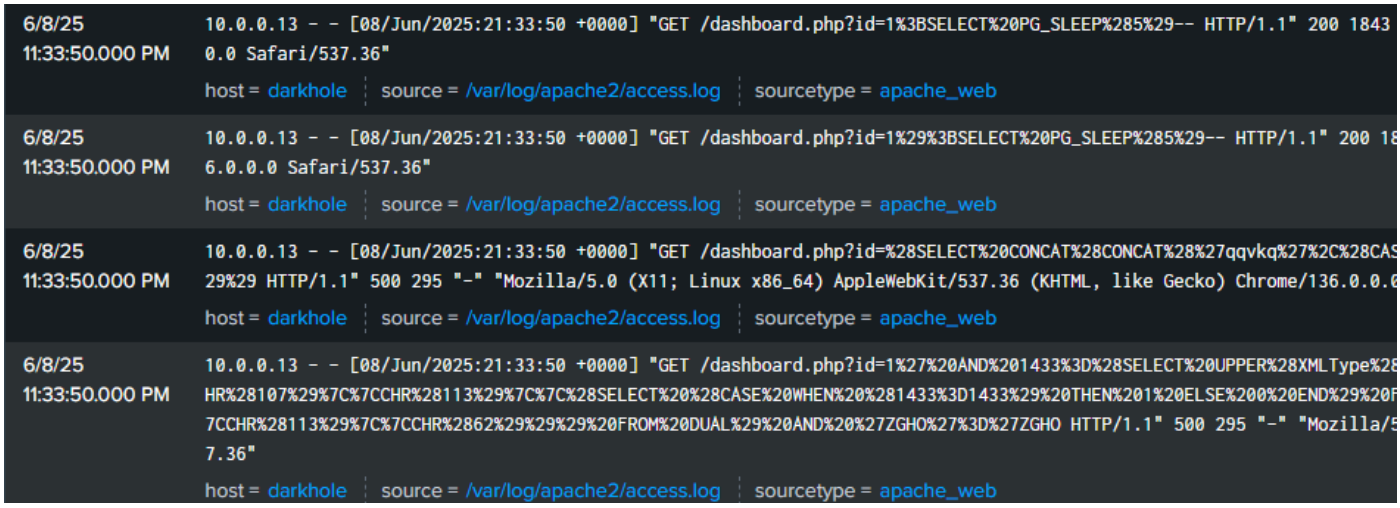
28 events related to “HTTP 200” code directories were found on the application, meaning that the nmap scan was successful in enumerating directories.



The attacker has identified the common /.git/ directory at web code 200, meaning the /.git directory is publicly accessible on the application, possibly disclosing sensitive login credentials

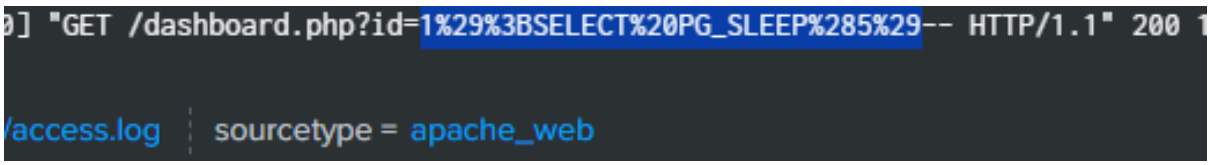


Minutues after accessing the /.git directory, the attacker reached the /login directory and somehow managed to log in to a user and reach /dashboard. This means that the /.git folder likely leaked login credentials.

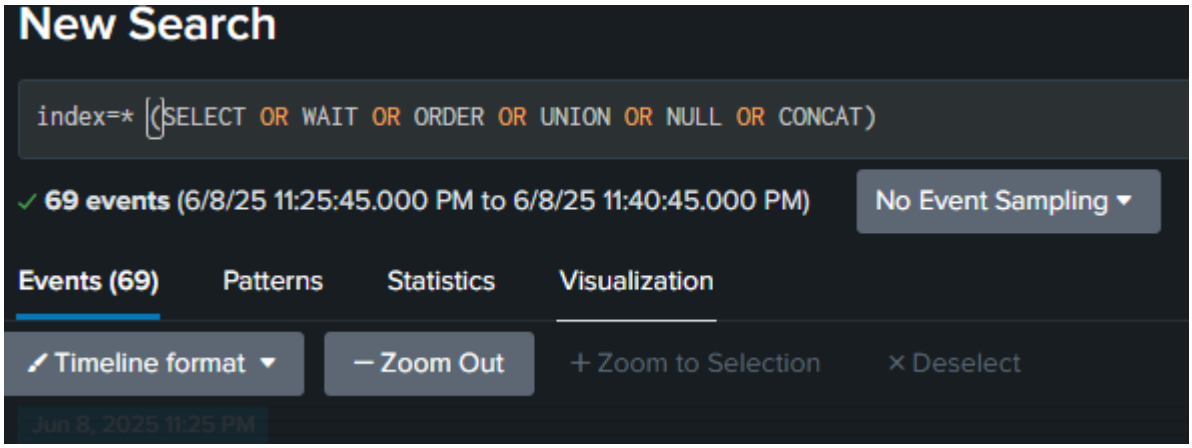


Once dashboard was reached at '/dashboard.php?id=1', it looks like the attacker performed an

automated SQL injection payloads attack on the ?id=1 parameter. This is a common potential endpoint for SQL injections. Note that the requests are made at the same timestamp...

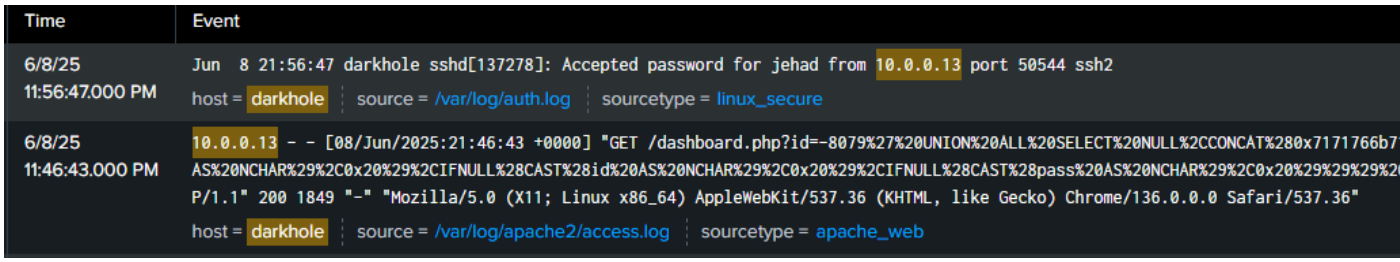


The highlighted payload is a basic SQL injection payload to test how the application responds.



Narrow down the search by selecting keywords that are common in SQL injection payloads

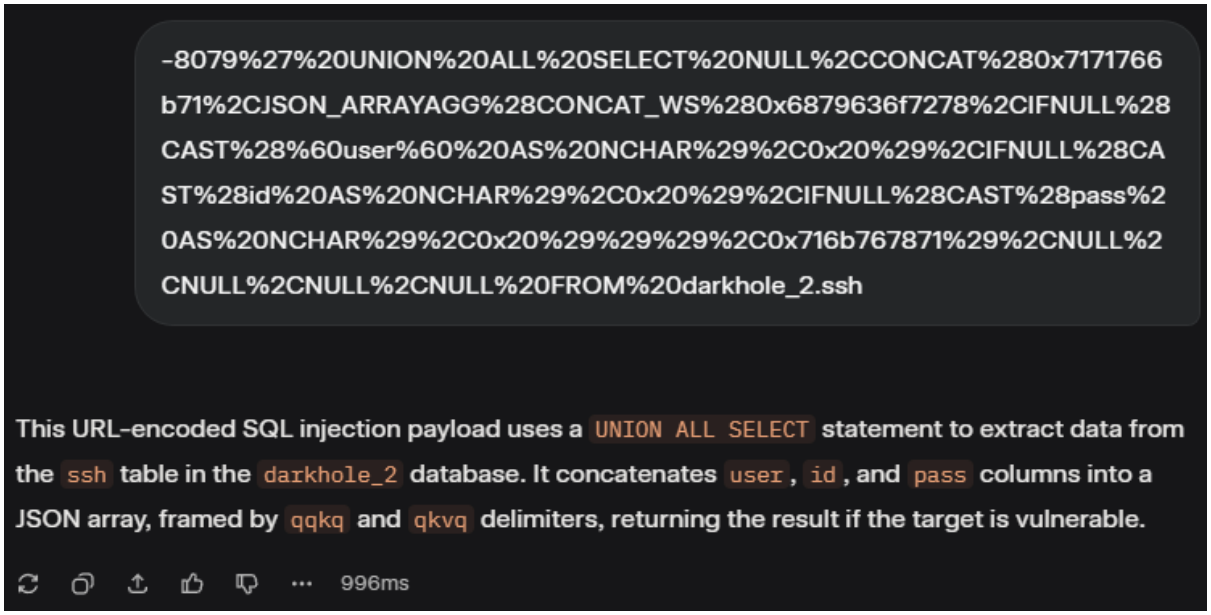
69 total events relating to injection payload keywords were found around the same or closely related timestamps, confirming that this was an automated attack.



Shortly after the last SQL injection payload request was received, a SSH connection was accepted and opened from the IP ”10.0.0.13” (Attacker’s Kali Linux machine)

Let's take a closer look at what the last payload does and requests from the database. We will use Grok AI chatbot to recognize encoded payload and give a description of what the payload requests.

```
GET /dashboard.php?id-
8079%27%20UNION%20ALL%20SELECT%20NULL%2CCONCAT%280x7171766b71%2CJSON_ARRAYAGG%28CONCA
T_WS%280x6879636f7278%2CIFNULL%28CAST%28%60user%60%20AS%20NCHAR%29%2C0x20%29%2CIFNULL
%28CAST%28id%20AS%20NCHAR%29%2C0x20%29%2CIFNULL%28CAST%28pass%20AS%20NCHAR%29%2C0x20%
29%29%29%2C0x716b767871%29%2CNULL%2CNULL%2CNULL%2CNULL%20FROM%20darkhole_2.ssh--%20-
HTTP/1.1" 200
```

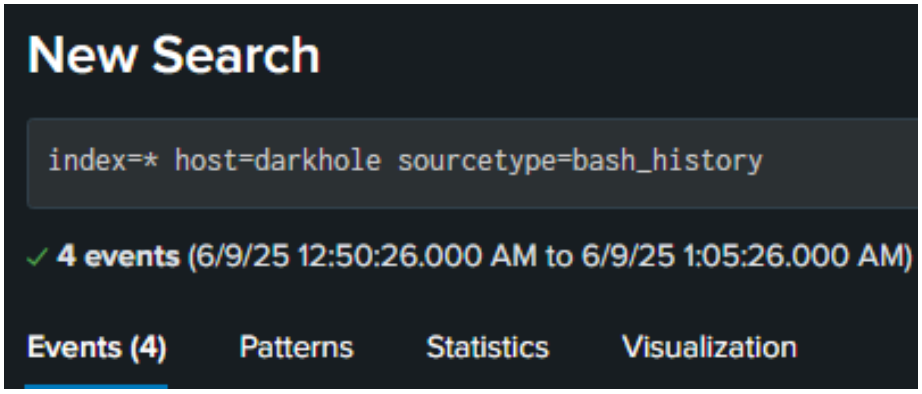


Grok AI

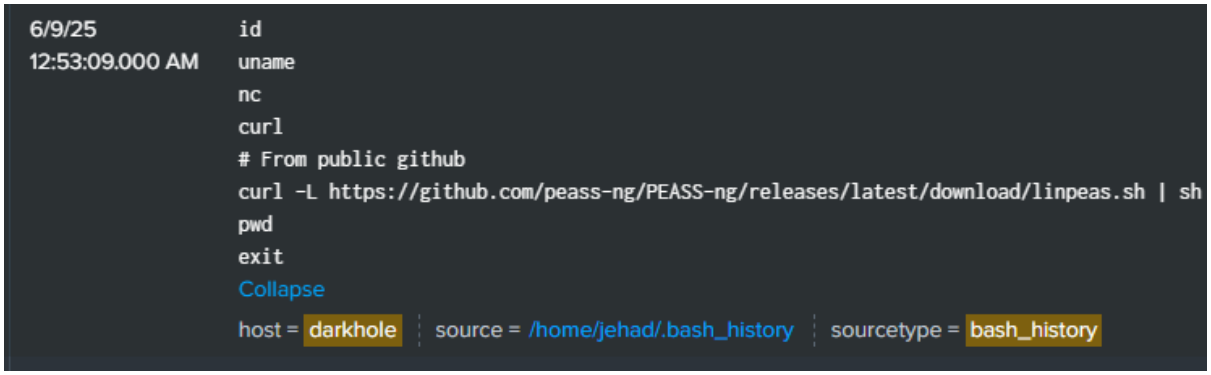
When asking Grok AI to decode and explain function of payload, it says it requests database data from the “ssh” table inside the “darkhole_2” database. The table included columns for username, id and password of SSH credentials. SSH credentials were leaked and used to open a SSH connection.

An SSH connection was successfully logged in using the correct credentials for user ”jehad”.

Check the bash history of jehad to see what was done.



'index=* host=darkhole sourcetype=bash_history' will list bash history of all connections opened and closed. Look for the history of the accepted SSH connection for user "jehad"



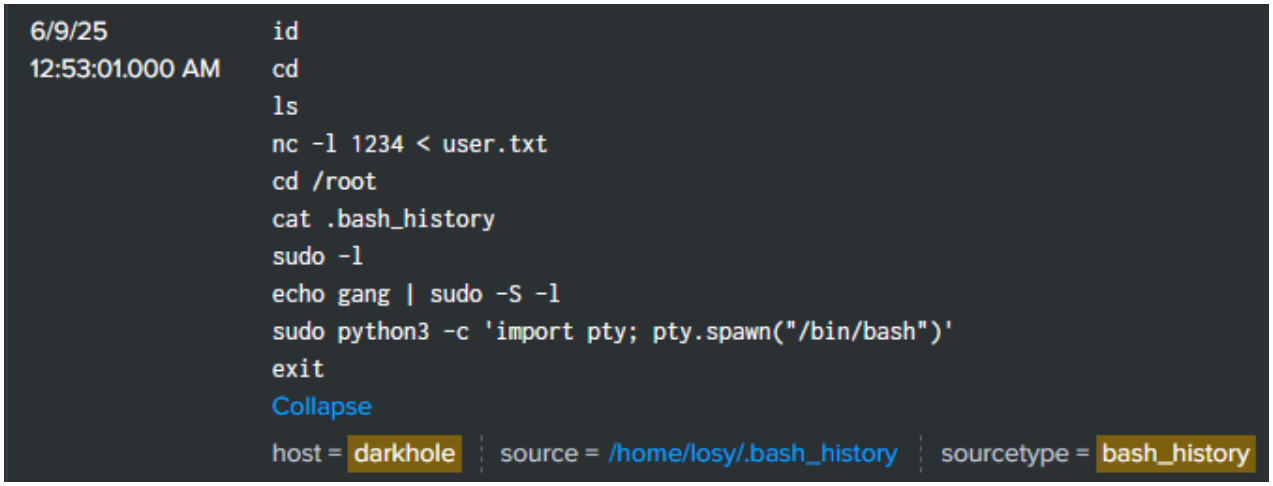
Once the attacker logged into "jehad" through SSH, they immediately attempt to find information to do privilege escalation further up the latter.

The attacker used the curl command to install a bash script from github to the machine.

Did privilege escalation using linPEAS, a common script used in Linux environments to do internal enumeration on possible ways to exploit and escalate privileges and permission to higher level.

Next upcoming event suggest they succeeded to escalate to another user:

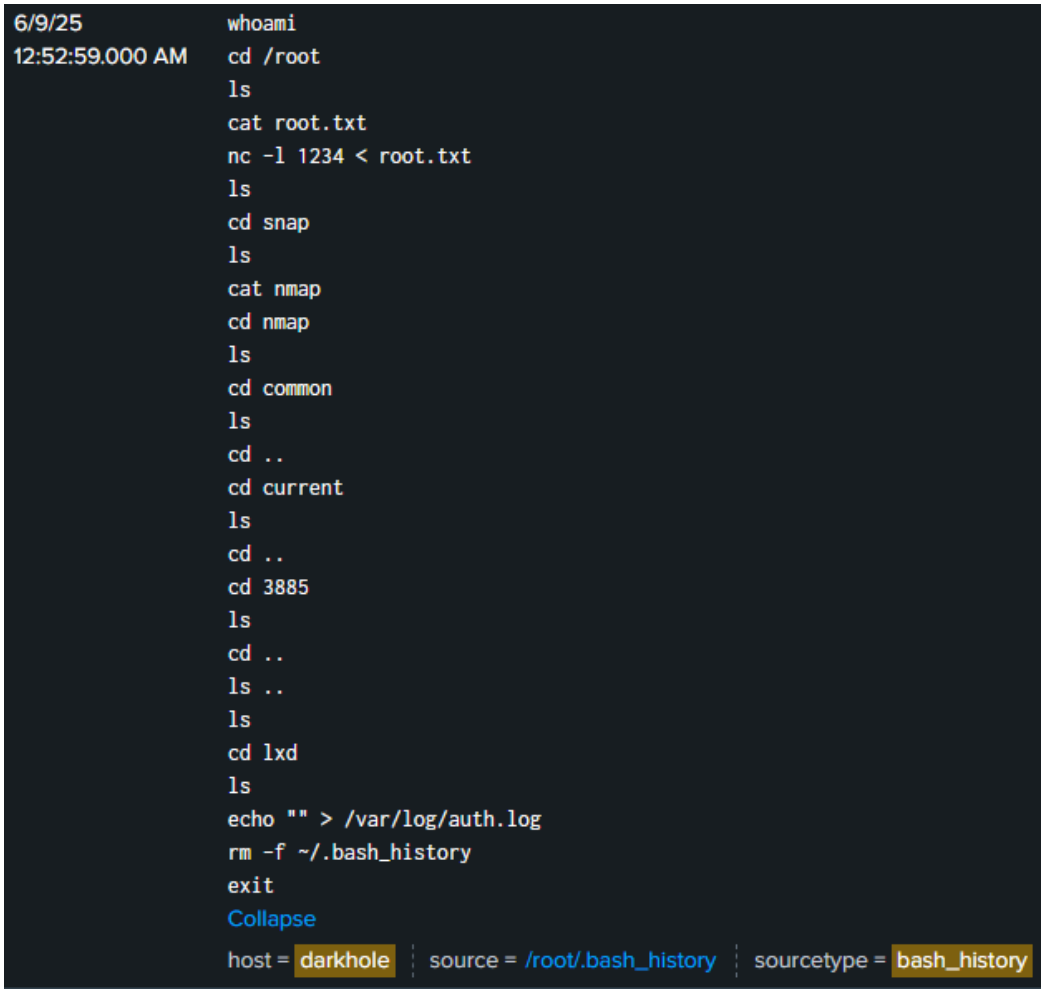
(Timestamp is set after the attack, because bash history saves when connection closes)



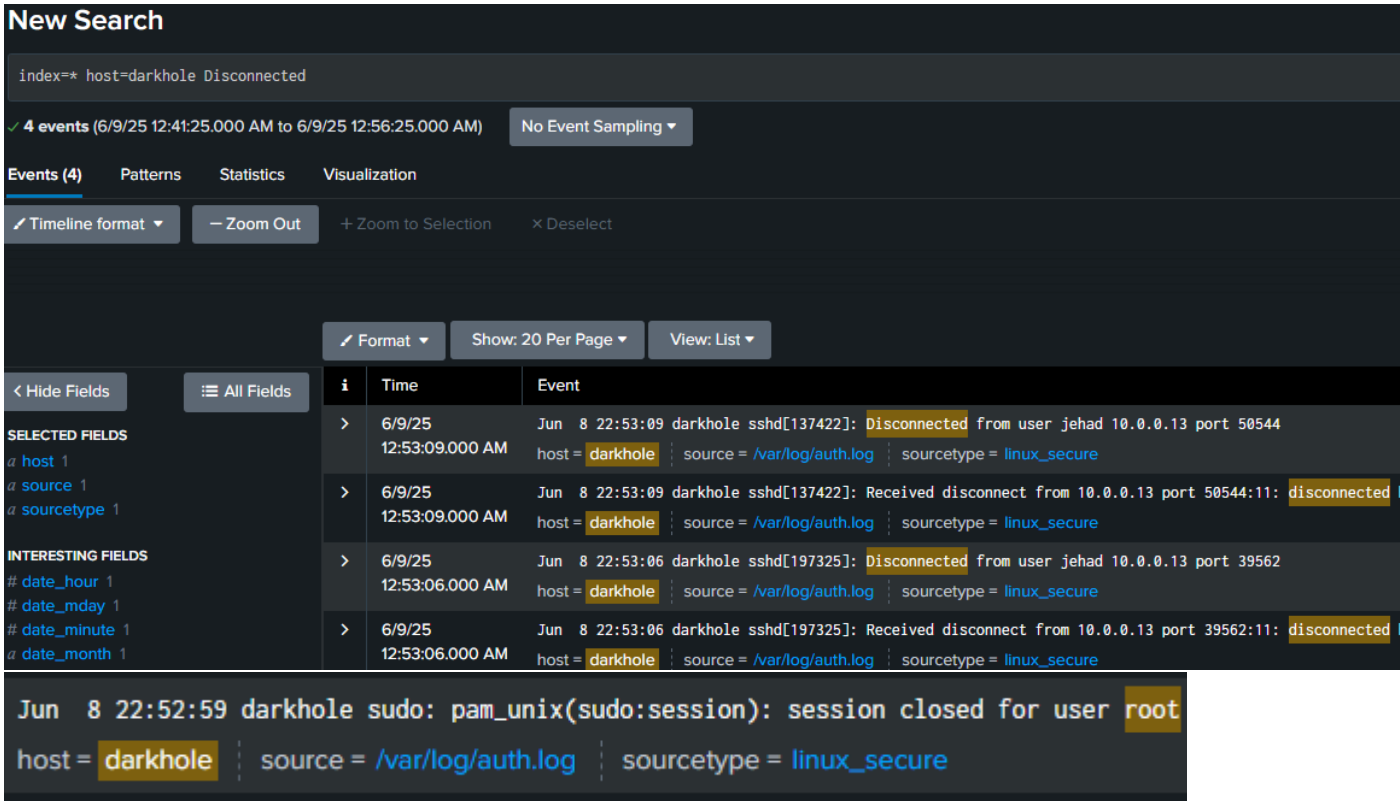
Another SSH connection was opened for user ”losy”. The user’s bash history file reveals that the attacker used netcat to exfiltrate a file ”user.txt” on port 1234. Then viewed the ”losy” user bash history commands, possibly revealing a credentials within the history file. Then used the leaked credentials to perform ”sudo -l” command, which lists allowed commands that can be executed using root permissions. By the looks of it, the command ”python3” could possibly be executed with root privileges.

The last command ” sudo python3 -c 'import pty; pty.spawn("/bin/bash")' “ uses python3 to spawn a reverse shell of the root user since python3 is executed with root privileges.

Root user login confirmed. A connection was opened for root in the upcoming event.



Viewing the root bash history reveals final activies of the attacker once reached root. The attacker used netcat on port 4444 to exfiltrate a file “root.txt” back to attacker machine, navigated through the system possibly looking for more files to exfiltrate and at last deleted the log files on the system. Logs of “/var/log/auth.log” were erased and changed with no value and contents of /.bash_history were deleted in an attempt to cover tracks of the attack.

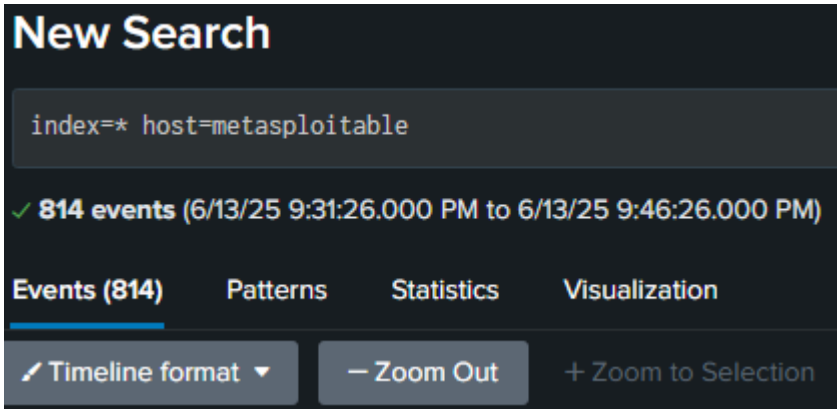


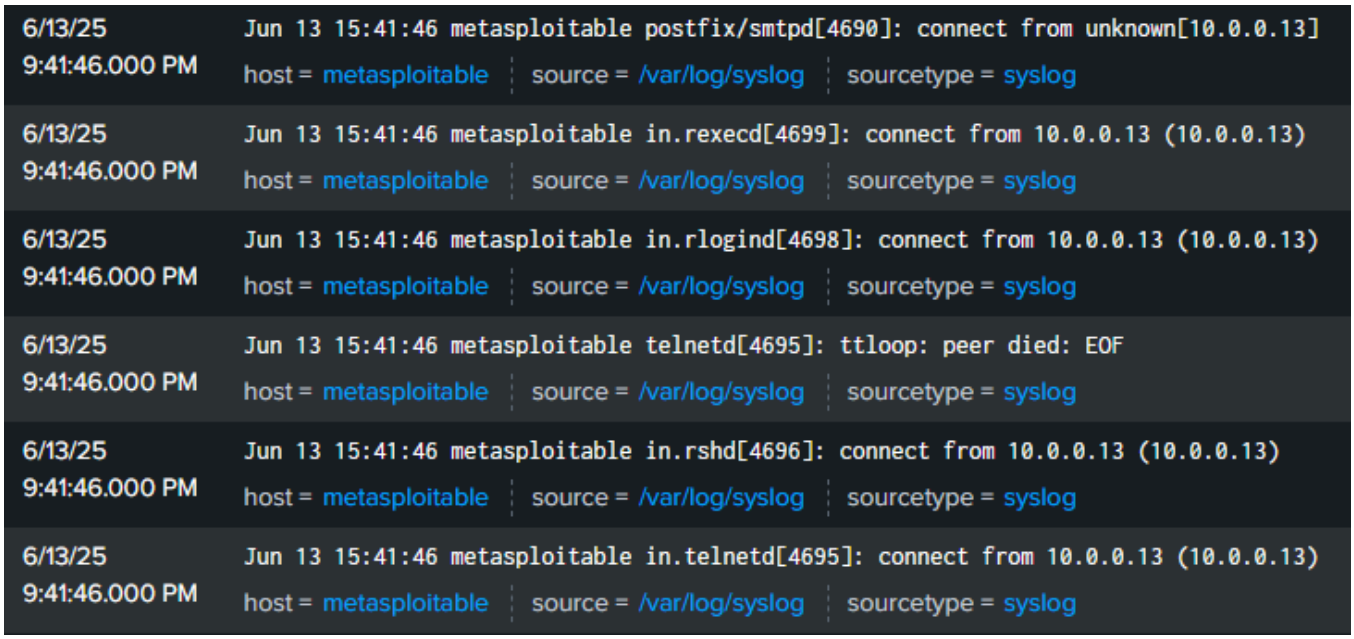
Final logs of the Darkhole 2 Linux machine attack shows various connections closed, including the previous connections of user “jehad”, “losy” and “root” user.

Attack on Darkhole Linux machine is finished at this point.

Metasploitable Linux machine attack

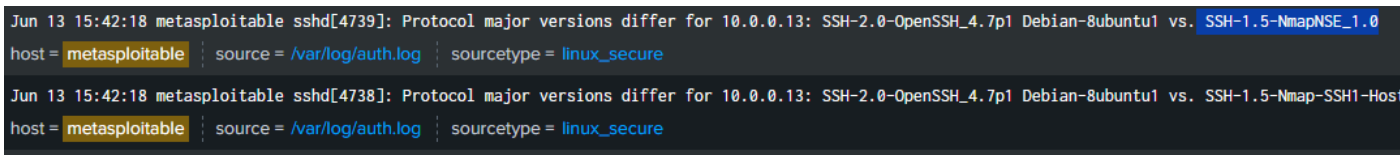
A few days later, the attacker came back and targeted another Linux machine “Metasploitable”.



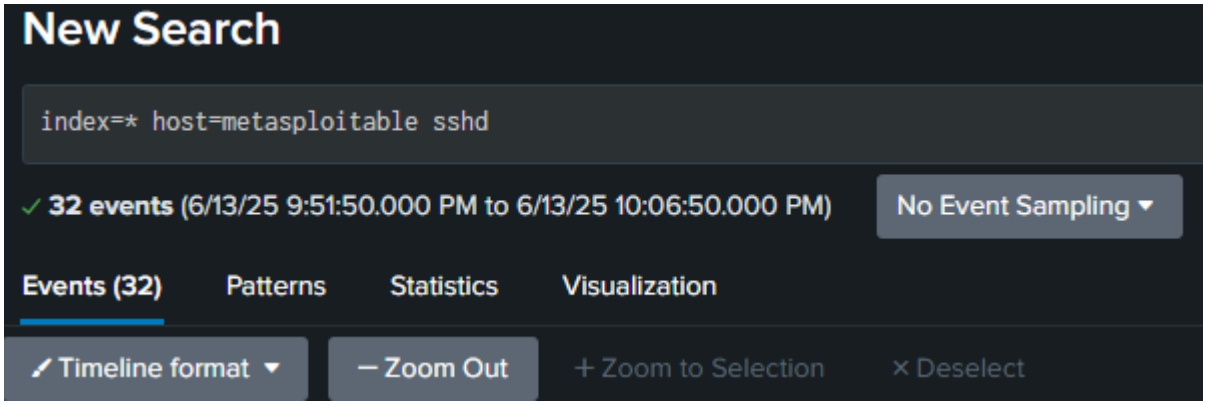


Various services on the Metasploitable machine received connections from 10.0.0.13 (Attacker's machine. Services such as SSH, telnet, smtp and others received connection at the same timestamp.

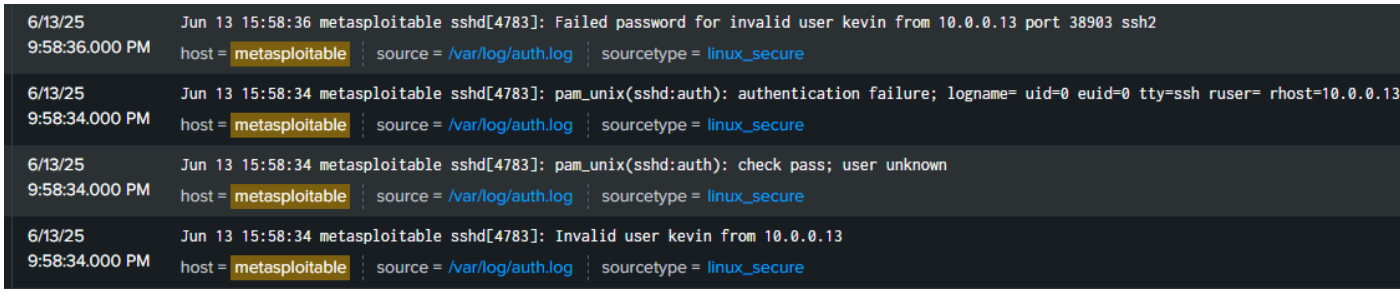
This suggests a nmap scan was possibly formed.



Service version were detected in the following logs, as well as "NmapNSE_1.0" being mentioned in the log. At this point, it is clear that a network scan was performed to identify services and its versions.

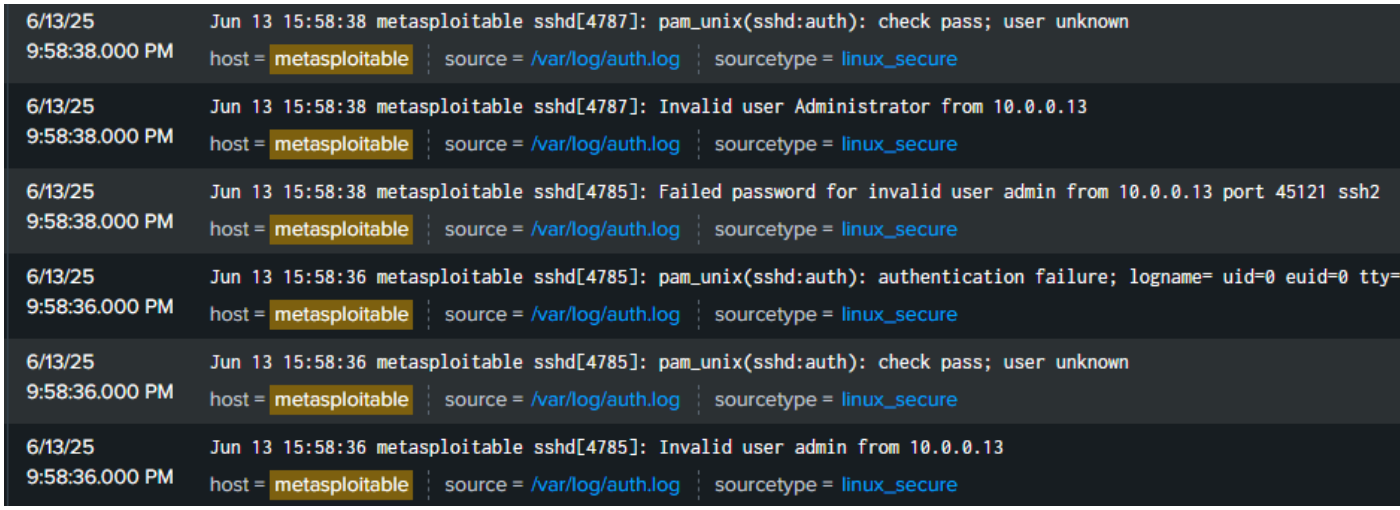


Multiple logs related to SSH activity were detected. The search query ” index=* host=metasploitable sshd ” lists 32 ssh events.



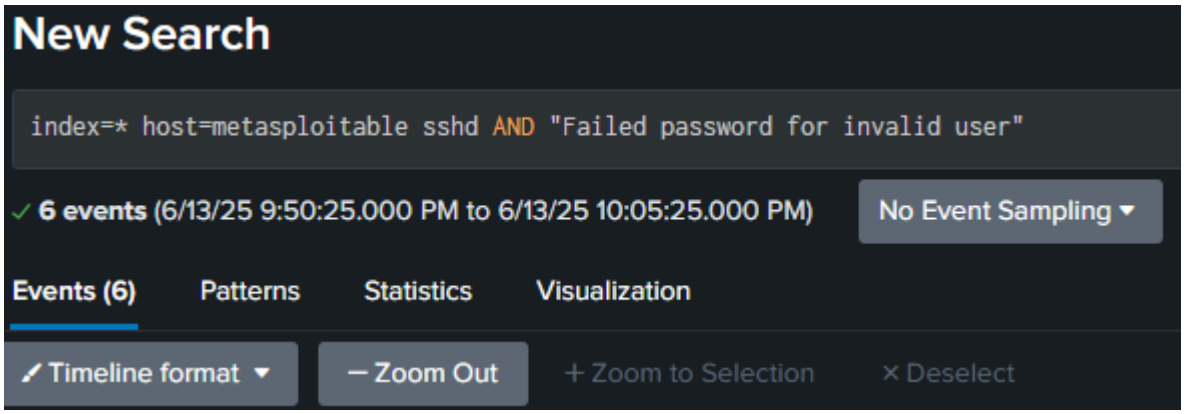
The SSH logs shows shows a series of connection attempts from IP 10.0.0.13 (Attacker machine).

The first connection was an attempted logon by user “kevin”, followed by a response as “user unknown” and then failed password before closing the connection.

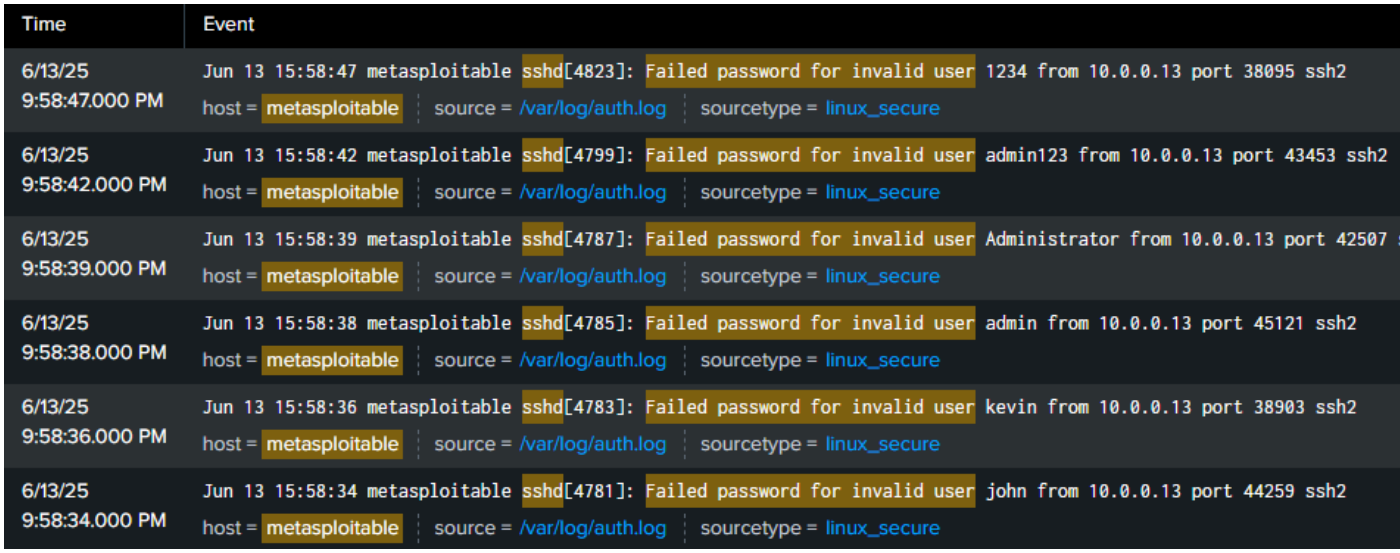


Another event of an attempted login by user ”admin”, followed by ”user unknown” and failed password for invalid user logon. Both of these user logon attempts were made on the exact same timestamp.

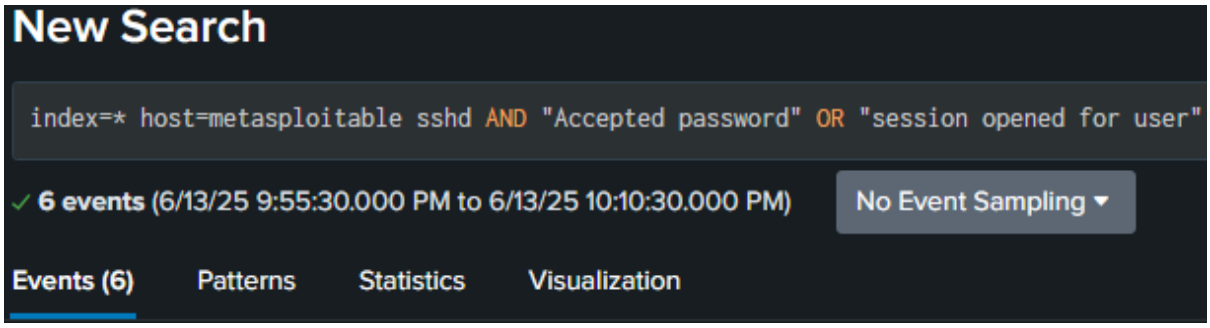
Possible brute force attack, look into how many logins failed and what succeded:



6 failed SSH login events were found.



Invalid logins for 6 different users were attempted, none of these users exist in the host machine.



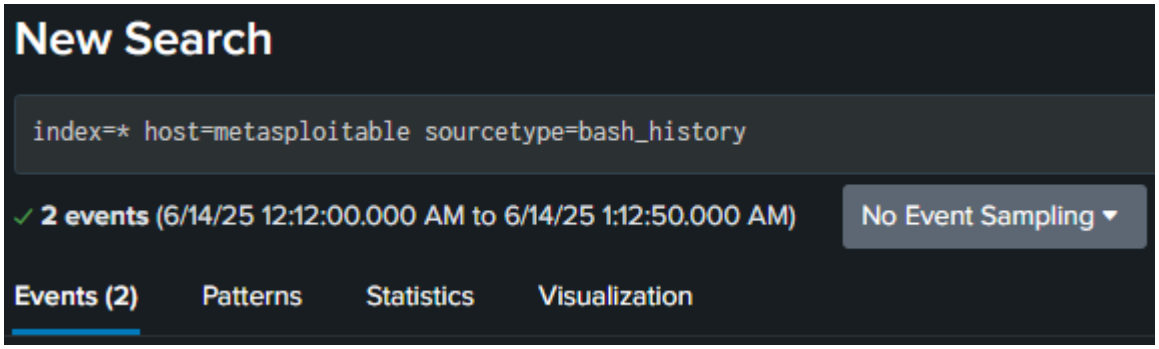
List if any brute force attempts were successful for any users on the metasploitable machine.

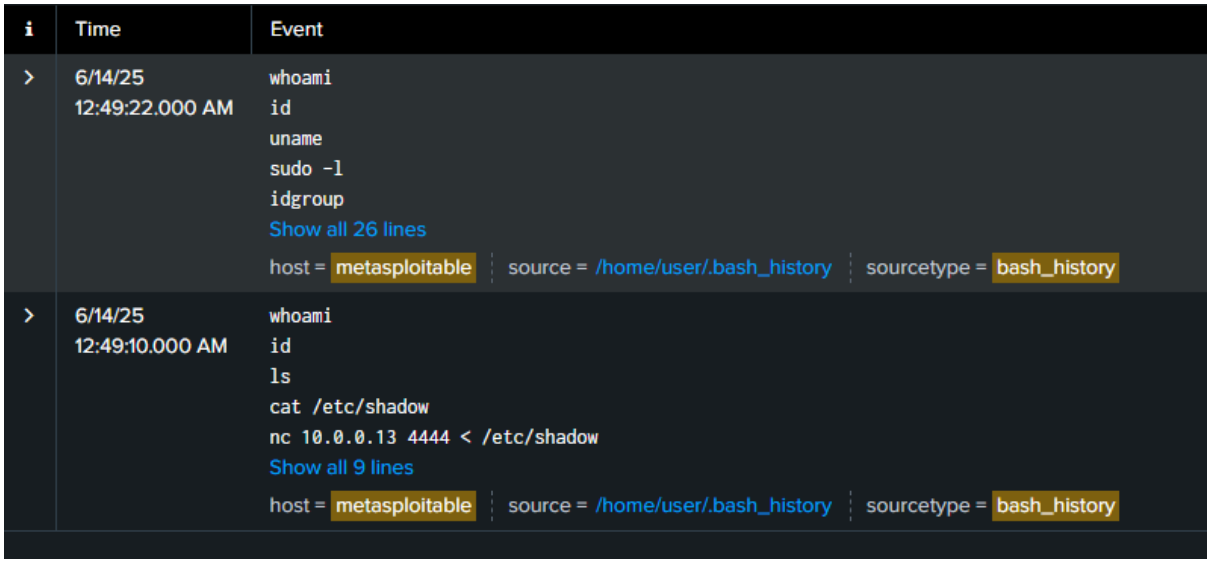
Time	Event
6/13/25 9:58:43.000 PM	Jun 13 15:58:43 metasploitable sshd[4813]: pam_unix(sshd:session): session opened for user user by (uid=0) host = metasploitable source = /var/log/auth.log sourcetype = linux_secure
6/13/25 9:58:43.000 PM	Jun 13 15:58:43 metasploitable sshd[4811]: Accepted password for user from 10.0.0.13 port 46803 ssh2 host = metasploitable source = /var/log/auth.log sourcetype = linux_secure
6/13/25 9:58:42.000 PM	Jun 13 15:58:42 metasploitable sshd[4803]: pam_unix(sshd:session): session opened for user service by (uid=0) host = metasploitable source = /var/log/auth.log sourcetype = linux_secure
6/13/25 9:58:42.000 PM	Jun 13 15:58:42 metasploitable sshd[4801]: Accepted password for service from 10.0.0.13 port 46789 ssh2 host = metasploitable source = /var/log/auth.log sourcetype = linux_secure
6/13/25 9:58:39.000 PM	Jun 13 15:58:39 metasploitable sshd[4791]: pam_unix(sshd:session): session opened for user msfadmin by (uid=0) host = metasploitable source = /var/log/auth.log sourcetype = linux_secure
6/13/25 9:58:39.000 PM	Jun 13 15:58:39 metasploitable sshd[4789]: Accepted password for msfadmin from 10.0.0.13 port 39051 ssh2 host = metasploitable source = /var/log/auth.log sourcetype = linux_secure

6 events of 3 different users were found of accepted password and opened session.

Users msfadmin, user and service were successfully brute forced, probably weak credentials.

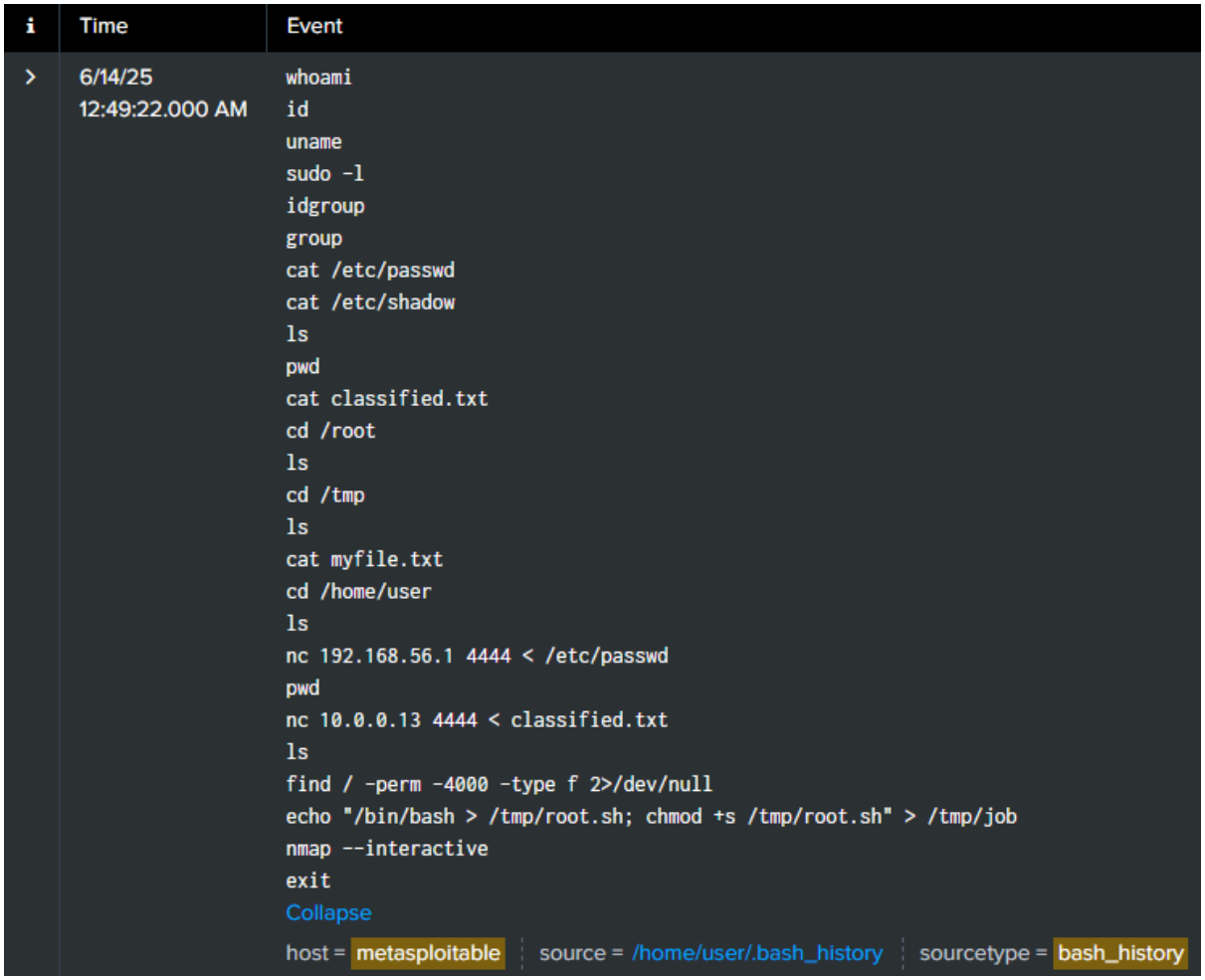
3 users were compromised and opened sessions, the attacker now has access to the 3 accounts, investigate which user they logged in and what commands were executed in bash history.





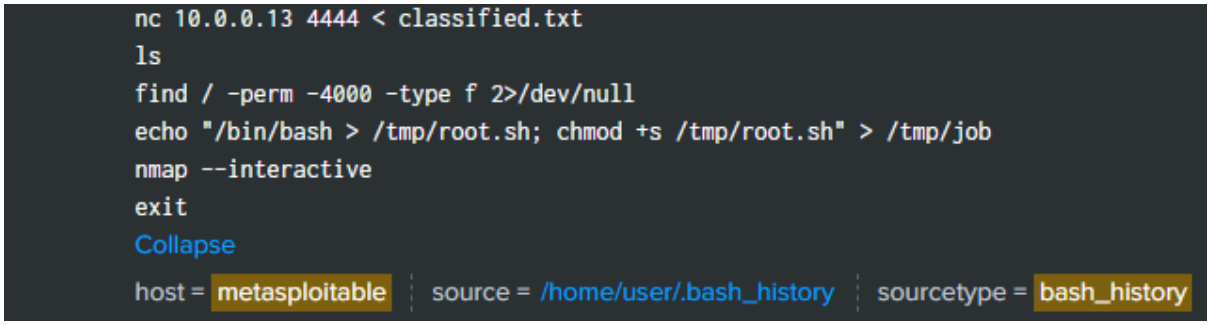
2 event logs for bash history were found. Both are listed to be from the user named ”user”.

Investigate the full bash histroy files:



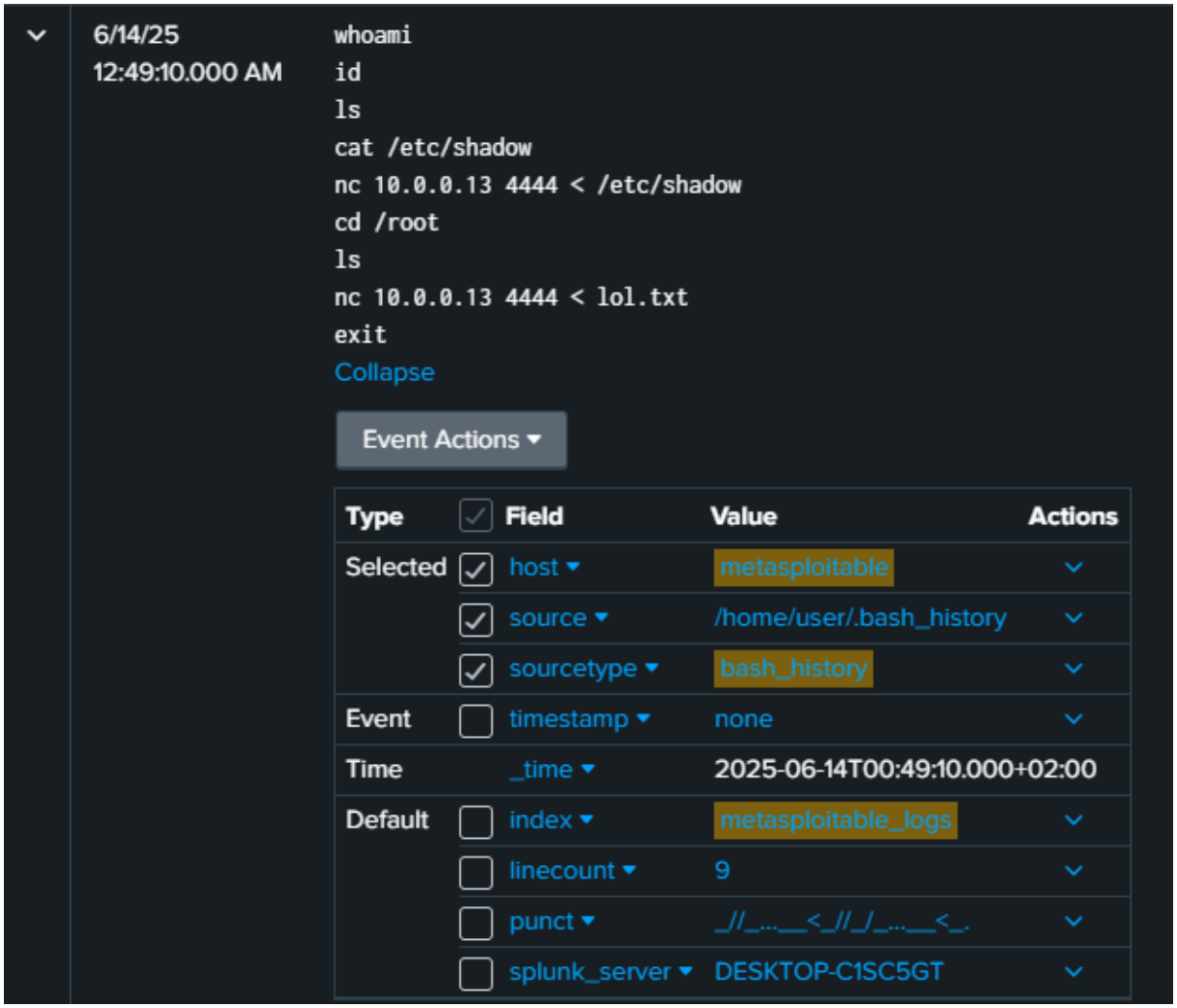
As ”user”, the attacker attempted to view the /etc/passwd and shadow file, but likely denied access.

Proceeded to view a file named ”classified.txt”, then attempted to exfiltrate /etc/passwd file on port 4444 using netcat. This was likely denied, but did likely succeed in exfiltrating ”classified.txt” file.



Attacker transferred classified file, did privilege escalation by listing bin commands that executes with root privileges. Once a exploitable bin command was found, commands were used to likely exploited vulnerable bin for ”nmap” command.

New session was then created, investigate the last bash history event log for the next connection:



Attacker likely escalated to root, although the bash history file is under the user named ”user”.

Root user viewed /etc/shadow password file, exfiltrated it back to attacker machine on port 4444. Then navigated to /root directory and exfiltrated a file named “lol.txt” located in the root directory back to attacker machine. Exit and closed the connection.

Tools used

Splunk SIEM

ChatGPT & Grok AI