

SPLUNK

NETWORKS AND CYBER SECURITY 2 PROJECT



December 3, 2024

cy-d

Abdul Sami Qasim (22i-1725)

Ahmad Abdullah (22i-1609)

Talha Bin Obaid (22i-1577)

Contents

[Introduction 3](#_Toc184152241)

[Tools used 3](#_Toc184152242)

[1. Splunk 3](#_Toc184152243)

[2. Suricata 3](#_Toc184152244)

[3. Snort 3](#_Toc184152245)

[4. UFW 3](#_Toc184152246)

[5. NMAP 4](#_Toc184152247)

[6. Rsyslog 4](#_Toc184152248)

[Setups 4](#_Toc184152249)

[Installation Steps for Splunk Enterprise on Windows 4](#_Toc184152250)

[**1.** Download the Splunk Enterprise Installer 4](#_Toc184152251)

[**2.** Run the Installer 4](#_Toc184152252)

[**3.** Accept the License Agreement 4](#_Toc184152253)

[**4.** Choose Installation Type 5](#_Toc184152254)

[**5.** Select Installation Directory 5](#_Toc184152255)

[**6.** Set the Administrator Password 5](#_Toc184152256)

[**7.** Choose Splunk Web Port 5](#_Toc184152257)

[**8.** Start the Installation 5](#_Toc184152258)

[**9.** Access Splunk Enterprise 5](#_Toc184152259)

[**10.** Configuring Indexes 5](#_Toc184152260)

[**11.** Setting Up Ports 6](#_Toc184152261)

[Installation Steps for Splunk Universal Forwarder on Kali Linux 6](#_Toc184152262)

[1. Download the Splunk Universal Forwarder Installer 6](#_Toc184152263)

[2. Install the Splunk Universal Forwarder 6](#_Toc184152264)

[3. Start the Splunk Universal Forwarder 6](#_Toc184152265)

[4. Set the Splunk Forwarder Admin Password 6](#_Toc184152266)

[5. Enable the Splunk Universal Forwarder to Start on Boot 7](#_Toc184152267)

[6. Configure the Forwarder to Send Data 7](#_Toc184152268)

[7. Access Splunk Enterprise 8](#_Toc184152269)

[Connection to Mobile App 8](#_Toc184152270)

[Implementations 8](#_Toc184152271)

[1. Network Traffic Analysis 8](#_Toc184152272)

[UFW 8](#_Toc184152273)

[SPL (search processing language) 10](#_Toc184152274)

[Alert 11](#_Toc184152275)

[Difficulties during implementation 12](#_Toc184152276)

[2. Suspicious file download 12](#_Toc184152277)

[Suricata 12](#_Toc184152278)

[SPL 12](#_Toc184152279)

[Alert 13](#_Toc184152280)

[Difficulties during implementation 14](#_Toc184152281)

[3. Data Exfiltration 14](#_Toc184152282)

[Suricata 14](#_Toc184152283)

[SPL 14](#_Toc184152284)

[Alert 15](#_Toc184152285)

[Difficulties during implementation 16](#_Toc184152286)

[Shortcomings 16](#_Toc184152287)

[1. Suspicious data destinations 16](#_Toc184152288)

[2. Monitoring file creation and modification 16](#_Toc184152289)

[3. Creating dashboards to visualize data trends 16](#_Toc184152290)

# Introduction

This project involves using splunk to detect some common malicious events across more than one device. We do this by forwarding logs generated by each device into a singular splunk receiver where we get alerts based on the rules we set in splunk.

For our project, we looks at three events, network traffic analysis, suspicious file download and data exfiltration.

# Tools used

## Splunk

This was the main platform that we had to use in this project and what it does for us is that it puts logs from all the devices it is connected to, in one place which lets us easily analyze them. We can also configure it to run a script in response to a generated alert.

## Suricata

Suricata is an open-source, high-performance Network IDS, IPS, and Network Security Monitoring (NSM) system. It is capable of real-time traffic analysis and logging, supporting multiple protocols including HTTP, DNS, and FTP. Suricata analyzes network traffic for suspicious activity, detecting anomalies, intrusions, and attacks. It integrates with various security tools and platforms to provide comprehensive protection and monitoring capabilities.

## Snort

Snort is an open-source network intrusion detection system (NIDS) and network intrusion prevention system (NIPS) developed by Cisco. It is used to monitor network traffic and detect a wide range of malicious activities, including denial-of-service (DoS) attacks, buffer overflow attempts, and port scanning. Snort works by inspecting packets in real-time, using signature-based detection to identify known threats, along with protocol analysis and anomaly detection for identifying suspicious behaviors.

## UFW

UFW (Uncomplicated Firewall) is a frontend for managing iptables firewall rules in Linux. It simplifies the process of configuring a firewall by providing an easy-to-use command-line interface. UFW is often used to configure rules for allowing or blocking incoming and outgoing traffic based on IP addresses, ports, and protocols.

## NMAP

Nmap (Network Mapper) is an open-source tool used for network discovery and security auditing. It helps identify hosts and services on a computer network by sending packets and analyzing the responses. Nmap is widely used for port scanning, network inventory, and vulnerability scanning.

## Rsyslog

rsyslog is a system logging utility for Linux and Unix systems. It collects, filters, and stores log data generated by system services, applications, and network devices. rsyslog is highly configurable and supports forwarding logs to remote systems, making it essential for centralized log management and troubleshooting.

# Setups

## Installation Steps for Splunk Enterprise on Windows

### Download the Splunk Enterprise Installer

* + I began by visiting the official Splunk Downloads page to download the Splunk Enterprise installer for Windows. I selected the appropriate .msi file for my version of Windows.

### Run the Installer

* + Once the installer was downloaded, I located the .msi file and double-clicked it to begin the installation process. When prompted by Windows, I clicked Run to allow the installer to make necessary changes to my system.

### Accept the License Agreement

* + In the Splunk Setup Wizard, I reviewed the License Agreement and selected the option to accept the terms by clicking I accept the terms in the License Agreement.

### Choose Installation Type

* + I was then prompted to choose between a Typical installation (recommended for most users) or a Custom installation. I chose the Typical installation, as it suited my needs and provided the default settings.

### Select Installation Directory

* + The default installation directory was C:\Program Files\Splunk, which I decided to keep. I clicked Next to continue with the installation.

### Set the Administrator Password

* + I set up the admin username (the default) and created a strong password. This password is crucial, as it would be required to log into the Splunk web interface.

### Choose Splunk Web Port

* + By default, the Splunk web interface is accessible on port 8000, which I left unchanged. However, I could have modified this if needed. I clicked Next to proceed.

### Start the Installation

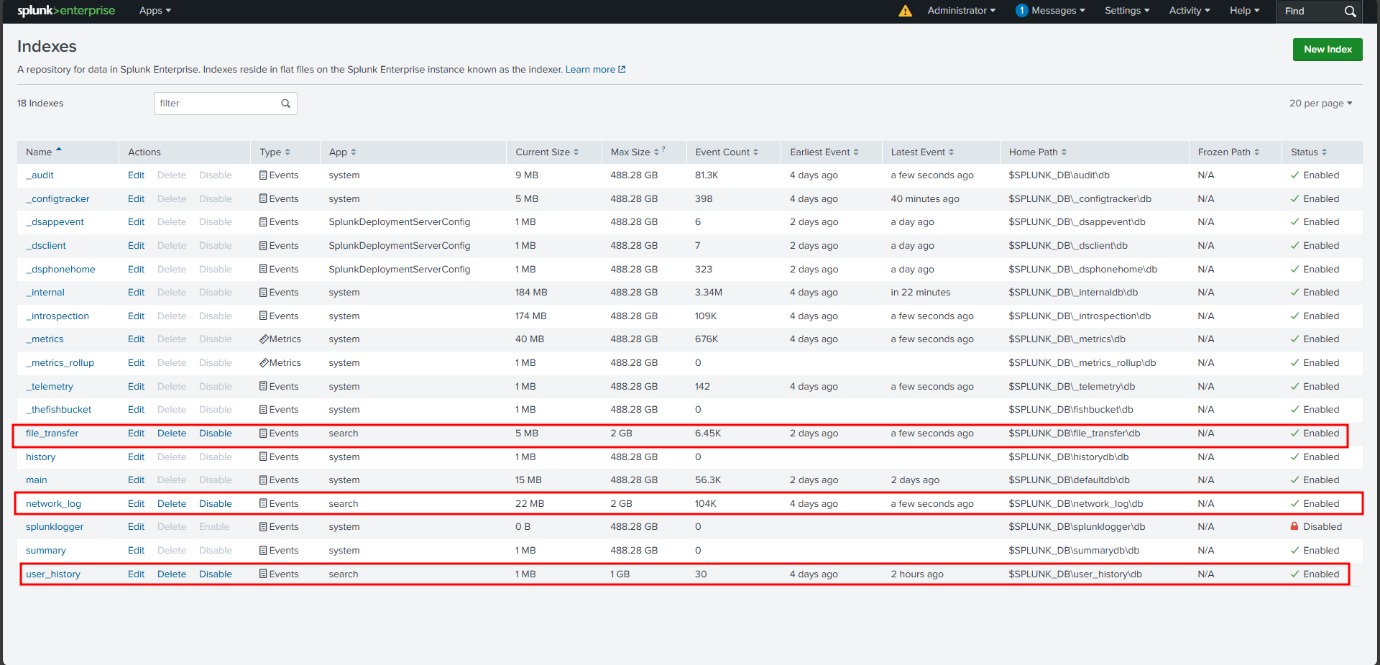
* + After reviewing my selections, I clicked Install to begin the installation. It took several minutes to complete, and I waited for the process to finish.

### Access Splunk Enterprise

* + Once the installation was completed, I was able to access Splunk by navigating to the web interface at http://127.0.0.1:8000 and logging in using the admin username and the password I set up earlier.

### Configuring Indexes

* Created proper indexes to facilitate better Searches using SPL. Different indexes were created for different types of logs so it became easy for us to analyse events shown in the splunk.

****

### Setting Up Ports

* Set up proper ports to receive logs and relevant files from forwarders.

A screenshot of a computer

Description automatically generated

After this, Splunk Enterprise was ready to receive logs from the other machines. So, the next setup was to configure the forwarder on my VM machine.

## Installation Steps for Splunk Universal Forwarder on Kali Linux

### Download the Splunk Universal Forwarder Installer

* + I started by visiting the official Splunk Downloads page and downloading the appropriate .deb package for Kali Linux**.**

### Install the Splunk Universal Forwarder

* + After the .deb package was downloaded, I opened a terminal in Kali Linux and navigated to the directory where the file was saved.
  + I used the following command to install the package:

**sudo dpkg -i splunkforwarder-<version>-<build>.deb**

### Start the Splunk Universal Forwarder

* + Once the installation was complete, I started the Splunk Universal Forwarder service by running:

**sudo /opt/splunkforwarder/bin/splunk start --accept-license**

* + This command starts the forwarder and accepts the Splunk license agreement.

### Set the Splunk Forwarder Admin Password

* + During the startup process, I was prompted to set the admin password for the forwarder. I created a strong password, which would be required to configure and manage the forwarder.

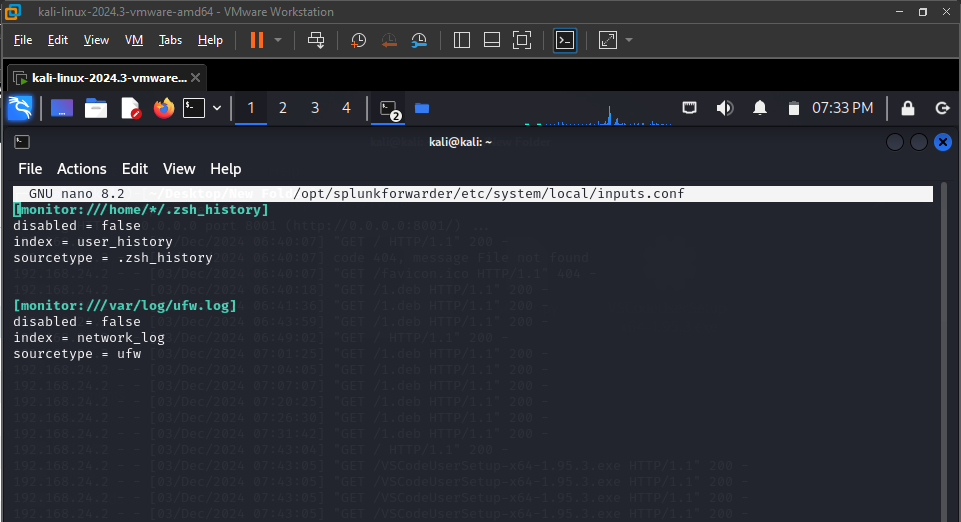
### Enable the Splunk Universal Forwarder to Start on Boot

* + To ensure the forwarder starts automatically when the system boots, I ran the following command:

**sudo /opt/splunkforwarder/bin/splunk enable boot-start**

### Configure the Forwarder to Send Data

* + I then configured the Splunk Universal Forwarder to send data to my Splunk Enterprise instance. This was done by setting the indexer IP address in the configuration file:



**sudo /opt/splunkforwarder/bin/splunk add forward-server 192.168:9997**

**A screenshot of a computer

Description automatically generated**

### Access Splunk Enterprise

* + Finally, I logged into the Splunk Enterprise web interface and verified that data from the forwarder was being received.

## Connection to Mobile App

After these setups, we connected the splunk indexer to our mobile app via splunk secure gateway so that we get our alerts there.

# Implementations

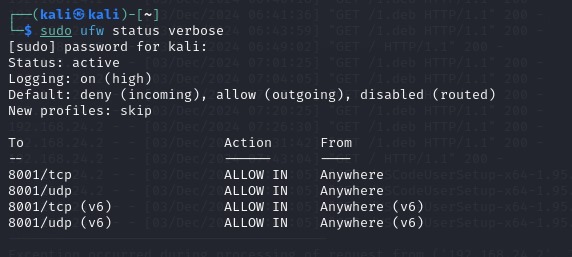
The following points were to be implemented in this project so that they generate alerts in splunk.

## Network Traffic Analysis

This point refers to analyzing network traffic in order to detect if port scanning is being carried out.

### UFW

To detect port scanning on our linux VM, we used UFW and set its logging on high to get as much information as we could. We forwarded the logs it made in /var/log/ufw.log to splunk.



ufw status

A screen shot of a computer

Description automatically generated

Sample log entry in ufw

A screenshot of a computer

Description automatically generated

Inputs.conf for ufw

We also used rsyslog to manage the logs that ufw generated, without this, ufw logs are not generated.

We used ufw because we have used it before so we knew how it works which made it easier for us to analyze network traffic with this. We wanted to get as much logs from here as possible and for this, we didn’t use rules in this, and to get alerts, we forwarded all of these logs to splunk and made alerts there.

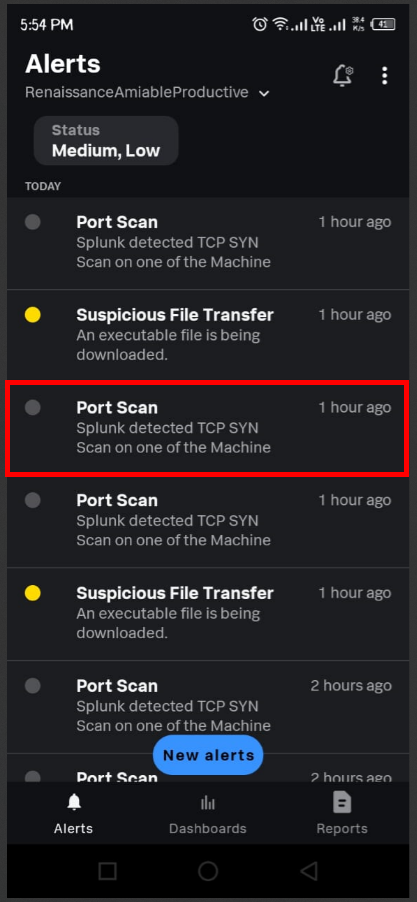
### SPL (search processing language)

To filter out the logs, we use an SPL and for this network analysis, we used the following:

*index=network\_log "PROTO=TCP" "SYN" | stats count by SRC | where count > 50*

in this SPL, we are looking for TCP packets with SYN flag from one source IP which exceeds a count of 50.

### Alert



Generated Alert for port scanning via network analysis

### Difficulties during implementation

For implementing the network analysis, we had to look at different firewalls for our linux environment, which took us a great deal of time and effort.

## Suspicious file download

This refers to detecting if malicious files (.bat, .exe or .docm) are being downloaded via curl, wget or http.

### Suricata

For this module, we were using a windows environment because we have seen that different type of exploits start from a simple curl or wget command from a C2 server to retrieve a malicious file over the internet to give an intruder a better foothold in the system.

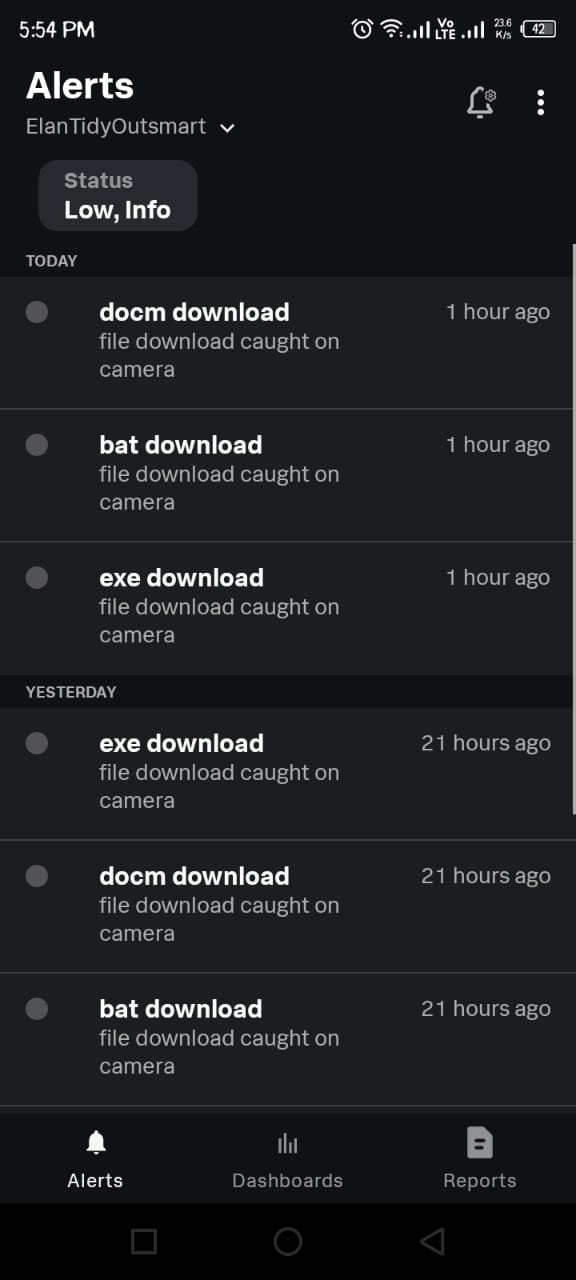
As we’re in a windows environment, we used Suricata to detect malicious file downloads.

### SPL

*index="file\_transfer" "http" | spath app\_proto | search app\_proto=http| spath "http.url" | search ".exe" OR ".pdf" OR ".sh" OR ".docm" OR ".xlsm" OR ".bat" OR ".bin" OR ".dll"*

This SPL query searches the "file\_transfer" index for HTTP traffic and looks for specific file types that may indicate suspicious or potentially malicious activity. It first filters logs to ensure they are related to HTTP traffic and then extracts the app\_proto and http.url fields. The query specifically searches for URLs containing extensions such as .exe, .pdf, .sh, .docm, .xlsm, .bat, .bin, and .dll, which are commonly associated with executable or potentially harmful files, allowing for detection of suspicious file transfers.

### Alert



### Difficulties during implementation

This became one of the most difficult task in this whole project because we couldn’t get suricata to generate alerts at first and to fix this we had to make a lot of changes to it’s configuration file and this whole event took up 2 days and 2 people.

## Data Exfiltration

For this task, we are detecting if a large amount of data is being sent out of the device (for the sake of simulating this event, we set the “large amount” to 100MB).

### Suricata

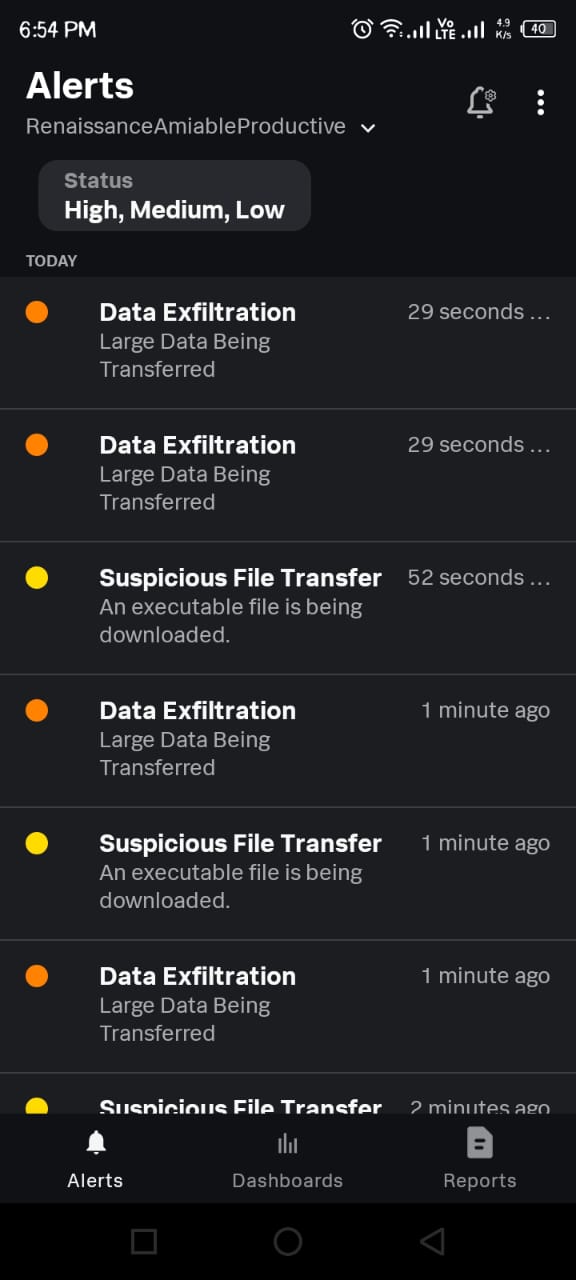
For this module, we again use suricata and this time, we also set it up on linux.

### SPL

*index="file\_transfer" | spath "http.length" | search "http.length">=10000000*

This SPL query searches the "file\_transfer" index for HTTP traffic where the http.length field, representing the size of the HTTP payload, is greater than or equal to 100,000,000 bytes (100 MB). It extracts the http.length field using spath and filters the results to identify very large HTTP transfers. Such large file transfers could potentially indicate significant data movements or exfiltration attempts, making this query useful for detecting abnormal or suspiciously large outbound file transfers that may need further investigation.

### Alert



### Difficulties during implementation

For this specific task, we had to do some time consuming stuff like setup a web server on a VM to upload/download a file in order to simulate a data exfiltration event and then we had to make an SPL while took quite a bit of time as we tried to figure out how to filter packets that made up this event.

# Shortcomings

In our proposal, we had some things in mind that we wanted to implement that we couldn’t because of time constraints and the complexity of said task in comparison to our skill level today.

## Suspicious data destinations

We had planned to identify and classify specific destination IP addresses or domains as suspicious, aiming to flag any traffic directed towards these destinations as potential data exfiltration attempts. However, due to time constraints and the complexity of properly classifying and validating such destinations, we were unable to implement this feature in the project.

## Monitoring file creation and modification

Another feature we wanted to implement was the continuous monitoring of file creation and modification activities on the endpoints. This would allow us to detect any unauthorized file changes or new file introductions, a common tactic in data exfiltration and malware activities. Unfortunately, we faced technical challenges with integrating this functionality within the given time frame.

## Creating dashboards to visualize data trends

We intended to create dynamic dashboards within Splunk to visualize trends and patterns in network traffic, file transfers, and other monitored activities. These dashboards would have helped in identifying anomalies more easily, but the complexity of integrating the data streams with the visualization tools led to this task being deferred.