Minor Project

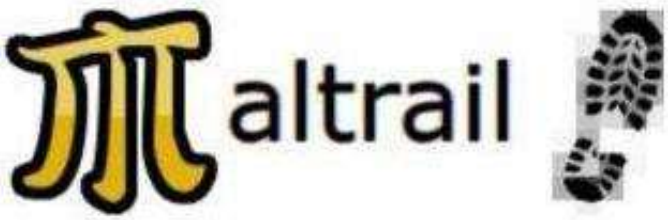
About ***IDS & IPS , NSM***

(Network Intrusion Detection Network Intrusion Prevention Network Security Monitoring)

Semester - IV

Dept : Cyber Forensic & Information Security

Mentor : Dr. Sudipta Ghosal [HOD OF CFS]



I

I I

I I I

I I

I

Introduction about IDS

Intrusion detection systems (IDSs) are software or hardware systems that automate the process of monitoring the events occurring in a computer system or network, analyzing them for signs of security problems. As network attacks have increased in number and severity over the past few years, intrusion detection systems have become a necessary addition to the security infrastructure of most organizations. This guidance document is intended as a primer in intrusion detection, developed for those who need to understand what security goals intrusion detection mechanisms serve, how to select and configure intrusion detection systems for their specific system and network environments, how to manage the output of intrusion detection systems, and how to integrate intrusion detection functions with the rest of the organizational security infrastructure. References to other information sources are also provided for the reader who requires specialized or more detailed advice on specific intrusion detection issues.

Overview of Intrusion Detection Systems 2.1. What is intrusion detection?

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions, defined as attempts to compromise the confidentiality, integrity, availability, or to bypass the security mechanisms of a computer or network. Intrusions are caused by attackers accessing the systems from the Internet, authorized users of the systems who attempt to gain additional privileges for which they are not authorized, and authorized users who misuse the privileges given them. Intrusion Detection Systems (IDSs) are software or hardware products that automate this monitoring and analysis process.

2.2. Why should I use Intrusion Detection Systems?

Intrusion detection allows organizations to protect their systems from the threats that come with increasing network connectivity and reliance on information systems. Given the level and nature of modern network security threats, the question for security professionals should not be whether to use intrusion detection, but which intrusion detection features and capabilities to use. IDSs have gained acceptance as a necessary addition to every

organization's security infrastructure. Despite the documented contributions intrusion detection technologies make to system security, in many organizations one must still justify the acquisition of IDSs. There are several compelling reasons to acquire and use IDSs: 1. To prevent problem behaviors by increasing the perceived risk of discovery and punishment for those who would attack or otherwise abuse the system,

2. To detect attacks and other security violations that are not prevented by other security measures, 3. To detect and deal with the preambles to attacks (commonly experienced as network probes and other "doorknob rattling" activities), 4. To document the existing threat to an organization 5. To act as quality control for security design and administration, especially of large and complex enterprises 6. To provide useful information about intrusions that do take place, allowing improved diagnosis, recovery, and correction of causative factors.

• 2.2.1. Preventing problems by increasing the perceived risk of discovery and punishment of attackers

A fundamental goal of computer security management is to affect the behavior of individual users in a way that protects information systems from security problems. Intrusion detection systems help organizations accomplish this goal by increasing the perceived risk of discovery and punishment of attackers. This serves as a significant deterrent to those who would violate security policy.

• 2.2.2. Detecting problems that are not prevented by other security measures

Attackers, using widely publicized techniques, can gain unauthorized access to many, if not most systems, especially those connected to public networks. This often happens when known vulnerabilities in the systems are not corrected. Although vendors and administrators are encouraged to address vulnerabilities (e.g. through public services such as ICAT, http://icat.nist.gov) lest they enable attacks, there are many situations in which this is not possible:

• In many legacy systems, the operating systems cannot be patched or updated.

• Even in systems in which patches can be applied, administrators sometimes have neither sufficient time nor resource to track and install all the necessary patches. This is a common problem, especially in environments that include a large number of hosts or a wide range of different hardware or software environments.

• Users can have compelling operational requirements for network services and protocols that are known to be vulnerable to attack.

• Both users and administrators make errors in configuring and using systems.

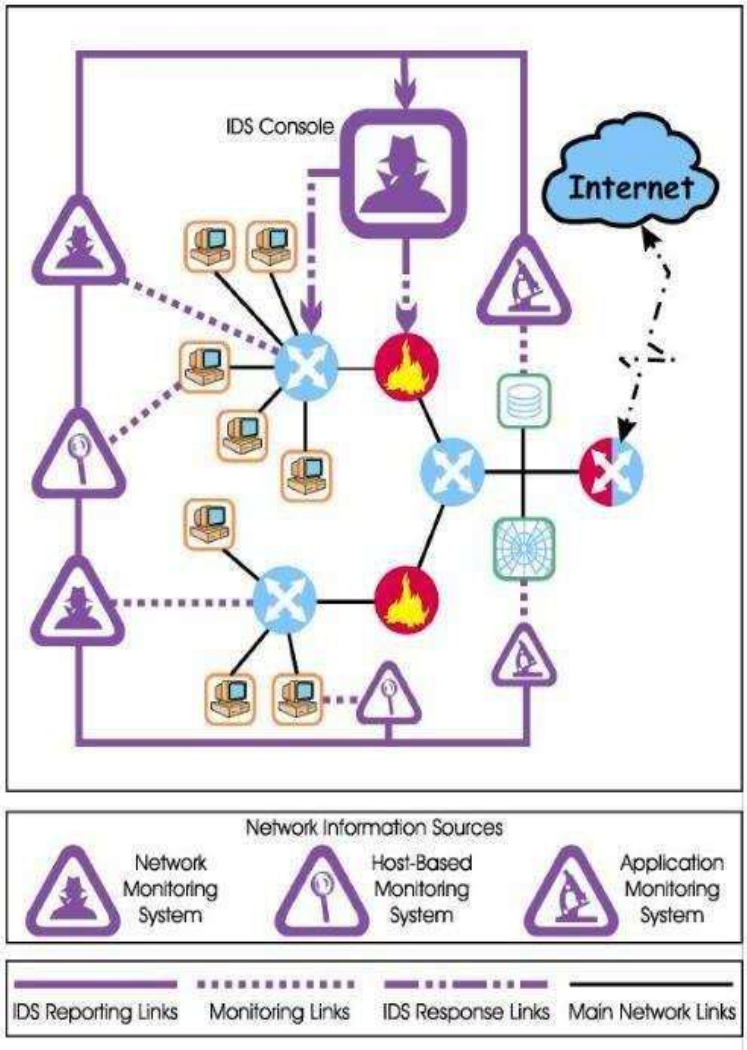


Figure 1: Centralized Control

• In configuring system access control mechanisms to reflect an organization's procedural computer use policy, discrepancies almost always occur. These disparities allow legitimate users to perform actions that are ill advised or that overstep their authorization. In an ideal world, commercial software vendors would minimize vulnerabilities in their products, and user organizations would correct all reported vulnerabilities quickly and reliably. However, in the real world, this seldom happens thanks to our

reliance on commercial software where new flaws and vulnerabilities are discovered on a daily basis. Given this state of affairs, intrusion detection can represent an excellent approach to protecting a system. An IDS can detect when an attacker has penetrated a system by exploiting an uncorrected or uncorrectable flaw. Furthermore, it can serve an important function in system protection, by bringing the fact that the system has been attacked to the attention of the administrators who can contain and recover any damage that results. This is far preferable to simply ignoring network security threats where one allows the attackers continued access to systems and the information on them.

Advantages and Disadvantages of IDS & IPS Mechanism

Intrusion Detection Systems (IDS)

Function: -Detection: IDS monitors network traffic for suspicious activity and potential threats but does not take action to prevent them. It generates alerts for the network administrator to investigate.

Advantages:

1. Visibility: - Provides comprehensive visibility into network traffic and potential threats without altering or blocking the traffic.

2. Threat Identification: - Can identify a wide range of threats, including known attacks, zero-day vulnerabilities, and policy violations through anomaly-based detection.

3. Passive Monitoring: - Operates passively, which means it doesn't interfere with network operations or introduce latency

. ***Disadvantages:***

1. No Active Response: - Does not block or mitigate threats, leading to potential delays in response.

2. Alert Overload: - Can generate a large number of alerts, including false positives, requiring significant resources to analyze and manage.

3. Requires Skilled Personnel: - Effective operation depends on having skilled personnel to analyze and respond to alerts.

Intrusion Detection System (IPS)

Advantages:

1. Active Defense: - Provides real-time protection by automatically blocking or mitigating threats.

2. Immediate Response: - Reduces the time to respond to threats by taking automated actions like dropping malicious packets, blocking IP addresses, or terminating connections.

3. Reduction in Alert Fatigue: - Typically generates fewer alerts compared to IDS, as it acts directly on the threats.

Disadvantages:

1. Network Latency: - Since it operates inline with traffic, it can introduce latency and potentially impact network performance.

2. Risk of False Positives: - Incorrectly identifying legitimate traffic as malicious can disrupt normal network operations or block critical services.

3. Complex Configuration: - Requires careful configuration and tuning to avoid unintended disruptions and ensure accurate threat detection. **Summary** - IDS is ideal for environments where visibility and alerting are critical, without the risk of impacting network performance. - IPS is suited for scenarios where proactive blocking and prevention of threats are necessary, with a focus on real-time response.

Coordinator List

Co-leader and Tester , Gaurab Paul
REG NO. **D222303467**
Date:

_____
Signature

Leader and Author , Sayuj Sur
REG NO. **D222303474**
Date:

_____
Signature

Gathering Information , Chayan Das
REG NO. **D222301669**
Date:

_____
Signature

Gathering Information , Shuvadeep Basu
REG NO. **D222303495**
Date:

_____
Signature

ACKNOWLEDGEMENT

We are deeply grateful for the opportunity to undertake this minor project, "Intrusion Detection System (IDS) & Intrusion Prevention System (IPS), Network Security Monitoring (NSM)," as part of the 4th Semester of the CFIS program.

First and foremost, We would like to express the sincere gratitude to our mentor, Dr. Sudipta Kr. Ghosal, for his invaluable guidance, support, and encouragement throughout the duration of this project. His profound expertise in network security and continuous feedback were instrumental in the successful completion of this work.

I also extend my thanks to my professors and peers for their insightful discussions and constructive criticism, which significantly contributed to the enhancement of this project.

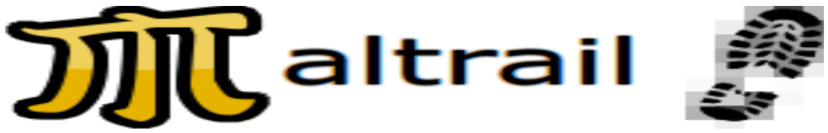
Lastly, I would like to acknowledge the support and encouragement of my total supporting team, whose belief in my abilities motivated me to strive for excellence.

Thank you all for your unwavering support.

Dr. Sudipta Kr Ghosal
HOD of CFS department
BEHALA GOVERNMENT POLYTECHNIC

Traffic Detection System using Matrail

Process pdf is help to do and for knowledge. That had been verified By Me (**Sayuj Sur**) Author and Leader of Minor Project of IDS & IPS.



Maltrail is a malicious traffic detection system, utilizing publicly available (black)lists containing malicious and/or generally suspicious trails, along with static trails compiled from various AV reports and custom user defined lists, where trail can be anything from domain name (e.g. zvpprsensinaix.[com] for Banjori malware), URL (e.g. http://109.162.38.120/harsh02.[exe] for known malicious executable), IP address (e.g. 185.130.5.231 for known attacker) or HTTP User-Agent header value (e.g. sqlmap for automatic SQL injection and database takeover tool).Also, it uses (optional) advanced heuristic mechanisms that can help in discovery of unknown threats (e.g. new malware).

Features

- Uses multiple public blacklists (alientvault, autoshun, badips, sblam etc)
- Has extensive static trails for identification (domain names, URLs, IP addresses or User-Agent values)
- Optional heuristic mechanisms for detection of unknown threats
- Based on Traffic -> Sensor <-> Server <-> Client Architecture
- Web reporting interface.

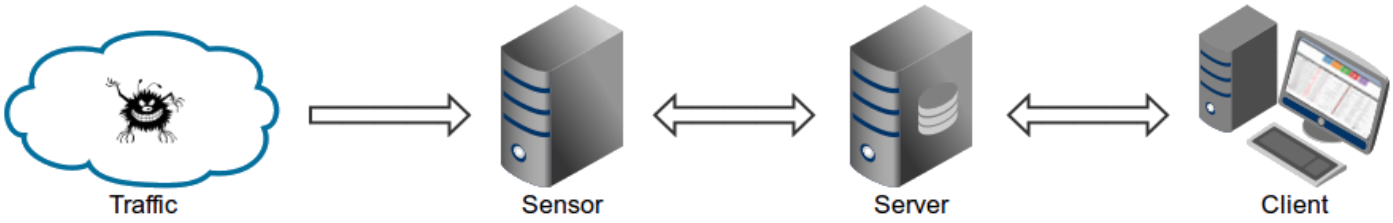
Architecture

Maltrail is based on the Traffic -> Sensor <-> Server <-> Client architecture. Sensor(s) is a standalone component running on the monitoring node (e.g. Linux platform connected passively to the SPAN/mirroring port or transparently inline on a Linux bridge) or at the standalone machine (e.g. Honeypot) where it "monitors" the passing Traffic for blacklisted items/trails (i.e. domain names, URLs and/or IPs).

In case of a positive match, it sends the event details to the (central) Server where they are being stored inside the appropriate logging directory (i.e. LOG_DIR described in the Configuration section). If Sensor is being run on the same machine as Server (default configuration), logs are stored directly into the local logging directory. Otherwise, they are being sent via UDP messages to the remote server (i.e. LOG_SERVER described in the Configuration section).

Traffic Detection System using Matrail

Server's primary role is to store the event details and provide back-end support for the reporting web application. In default configuration, server and sensor will run on the same machine. So, to prevent potential disruptions in sensor activities, the front-end reporting part is based on the "Fat client" architecture (i.e. all data post-processing is being done inside the client's web browser instance). Events (i.e. log entries) for the chosen (24h) period are transferred to the Client, where the reporting web application is solely responsible for the presentation part. Data is sent toward the client in compressed chunks, where they are processed sequentially. The final report is created in a highly condensed form, practically allowing presentation of virtually unlimited number of events.



Installation Guide for MALTRAIL

The Installation Guide will be in four segment:

1. First Segment compromises of all necessary cloning of repository
2. Second Segment is on making directories and moving the appropriate file paths.
3. Third Segment setting cron jobs, to auto start Mlatrail each time your system is rebooted.
4. Fourth Segment would be copying the sensor, and the server to the right paths, and then starting the services.

Maintain the Process Follow As Line Mentioned **(Any mistake may issue with connection)***

OS Type : Kali Linux

https://github.com/stamparm/maltrail firstly visit the website of MALTRAIL to understand the working and network architecture also some of the part is discussed in the above parts.

1st Segment

In the first segment it's the installation and basic configuration part so open the terminal and install some essential tools for development and network analysis for that enter the command:

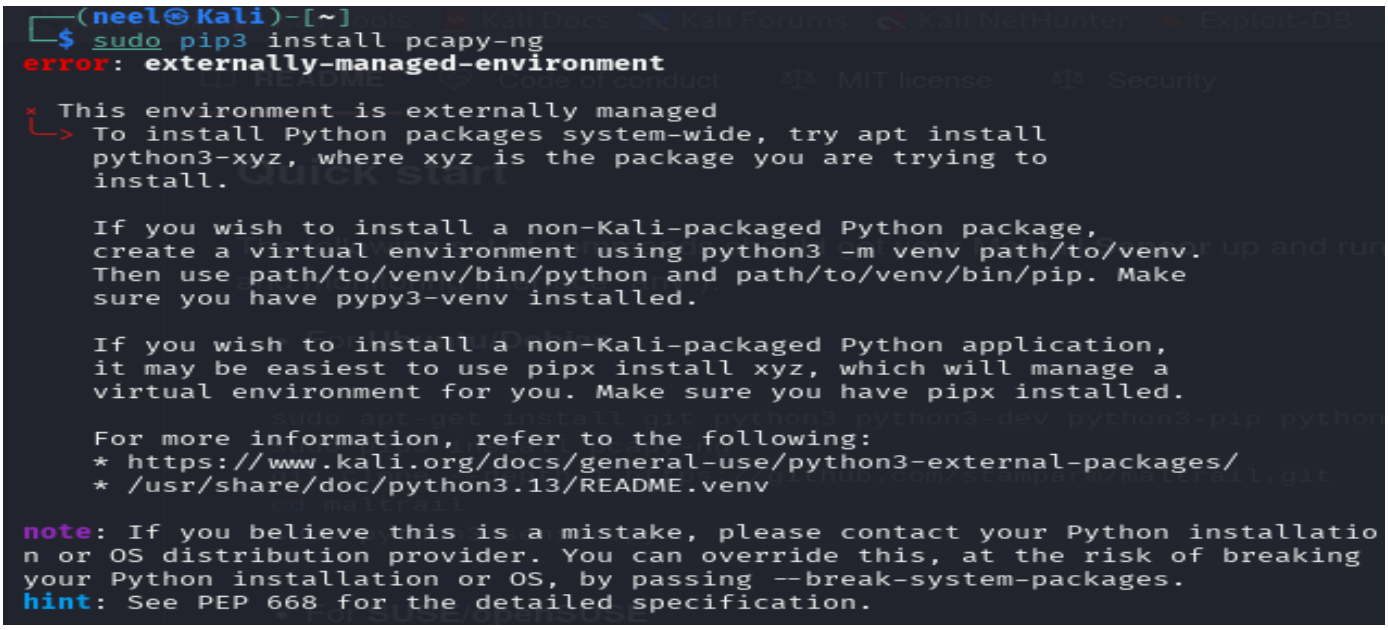
sudo apt-get install git python3 python3-dev python3-pip python-is-python3 libpcap-dev build-essential procps schedtool

```
┌──(neel㉿Kali)-[~]
└─$ sudo apt-get install git python3 python3-dev python3-pip python-is-python3
 libpcap-dev build-essential procps schedtool
[sudo] password for neel:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
git is already the newest version (1:2.47.2-0.1).
git set to manually installed.
python3 is already the newest version (3.13.2-2).
python3 set to manually installed.
python3-dev is already the newest version (3.13.2-2).
python3-dev set to manually installed.
python-is-python3 is already the newest version (3.11.4-1).
python-is-python3 set to manually installed.
build-essential is already the newest version (12.12).
procps is already the newest version (2:4.0.4-7).
The following additional packages will be installed:
  dbus dbus-bin dbus-daemon dbus-session-bus-common dbus-system-bus-common
  dbus-user-session dbus-x11 libcap-dev libcap2 libcap2-bin libdbus-1-3
  libdbus-1-dev libnss-systemd libpam-systemd libpcap0.8-dev libpkgconf3
  libsystemd-dev libsystemd-shared libsystemd0 libudev1 pkgconf pkgconf-bin
```

Traffic Detection System using Matrail

It will take some time depends upon your internet speed also it has huge installation so stay patient while downloading. After installation done in the next step install the pcapy libraries of python to capture network packets.

Install pcapy type **sudo pip3 install pcapy-ng** this command uses the Python 3 package installer to download and install the pcapy-ng library, which will enable your Python 3 scripts to interact with network traffic at a low level. You'll be able to capture, filter, and analyze network packets directly from your Python code.



Note: If you face this kind of error, **Externally Managed Environment:** Your Python installation is being managed by your operating system's package manager (in this case, likely apt since you're on Kali Linux). This means that the system expects you to install Python packages using apt rather than pip directly for system-wide installations.

Stick with pip in a virtual environment for pcapy-ng:

Despite apt not finding it, pcapy-ng might still be available on PyPI (the Python Package Index). Let's try the virtual environment approach again, as it often works for packages not directly in the system repositories:

python3 -m venv venv

source venv/bin/activate

pip install pcapy-ng

This will instruct pip to download and install pcapy-ng and its dependencies within the isolated virtual environment.

Traffic Detection System using Matrail

```
┌──(neel㉿Kali)-[~]
└─$ python3 -m venv venv

┌──(neel㉿Kali)-[~]
└─$ source venv/bin/activate

┌──(venv)─(neel㉿Kali)-[~]
└─$ pip install pcapy-ng
Collecting pcapy-ng
  Downloading pcapy-ng-1.0.9.tar.gz (38 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Building wheels for collected packages: pcapy-ng
  Building wheel for pcapy-ng (pyproject.toml) ... done
  Created wheel for pcapy-ng: filename=pcapy_ng-1.0.9-cp313-cp313-linux_x86_64
.whl size=86658 sha256=4d34b2e77f92abc7b3232bf72c6f2a1a92539e38b1e7282431f089d
a82d68ca2
  Stored in directory: /home/neel/.cache/pip/wheels/28/60/5a/27f8e8710cb6af16e
b1375d65af08053978b7481f1fbf091c7
Successfully built pcapy-ng
Installing collected packages: pcapy-ng
Successfully installed pcapy-ng-1.0.9
```

Type command **cd /tmp** navigates the current terminal session into the system's temporary directory. You'll be working within the /tmp folder after running this command. It's a common place for short-lived files or temporary work areas.

Now clone the MALTRAIL repo using the command **git clone --depth 1 https://github.com/stamparm/maltrail.git** , this command downloads the most recent version of the Maltrail project from GitHub to a new folder named maltrail in your current directory, without downloading its entire history. It's a quick way to get the latest code if you don't need to see how the project evolved over time.

```
┌──(neel㉿Kali)-[~]
└─$ cd /tmp

┌──(neel㉿Kali)-[/tmp]
└─$ git clone --depth 1 https://github.com/stamparm/maltrail.git
Cloning into 'maltrail' ...
remote: Enumerating objects: 2790, done.
remote: Counting objects: 100% (2790/2790), done.
remote: Compressing objects: 100% (2080/2080), done.
Receiving objects:  52% (1451/2790), 6.14 MiB | 425.00 KiB/s
```

Transfers the maltrail directory from the temporary path to a linux directory./tmp the **source** path, indicating the maltrail directory (and its contents) located within the /tmp directory. **/opt**, the **destination** path, a standard directory in Linux used for installing optional application software packages. It's a way to install the Maltrail software in a conventional location for optional applications. Also use the **sudo chown -R $USER:$USER /opt/maltrail** command to **use administrator rights to recursively change the owner and group of the /opt/maltrail directory and all its contents to your current user.** This gives you full control over the Maltrail installation files, allowing you to modify or execute them without needing sudo every time.

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo mv /tmp/maltrail /opt

┌──(neel㉿Kali)-[/tmp]
└─$ sudo chown -R $USER:$USER /opt/maltrail
```

Finally, the first segment was done perfectly going to the next segment.

Traffic Detection System using Matrail

Second Segment

Make a parent directory for storing log files also a maltrail subdirectory to store logs specific to the Maltrail application using the command **sudo mkdir -p /var/log/maltrail**

In short, this command uses administrator rights to create the /var/log/maltrail directory, and it will also create any necessary parent directories (/var, /var/log) along the way if they don't already exist. This ensures that the log directory for Maltrail is properly set up.

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo mkdir -p /var/log/maltrail
[sudo] password for neel:

┌──(neel㉿Kali)-[/tmp]
└─$
```

The directory has been created successfully, after that firstly in the **/etc** directory is the standard location in Linux for storing system-wide configuration files. Create a subdirectory named maltrail within it, likely to hold configuration files specific to the Maltrail application like the previous ones.

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo mkdir -p /etc/maltrail
```

Make a copy of the maltrail.conf configuration file from the Maltrail installation directory in **/opt** to the dedicated configuration directory **/etc/maltrail**. This is a standard practice to keep configuration files separate from the application's main files type

sudo cp /opt/maltrail/maltrail.conf /etc/maltrail

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo cp /opt/maltrail/maltrail.conf /etc/maltrail
```

Once it’s done move onto next step open the Maltrail configuration file **(maltrail.conf)** located in the /etc/maltrail directory using the nano text editor, allowing to view and make any changes to its settings.

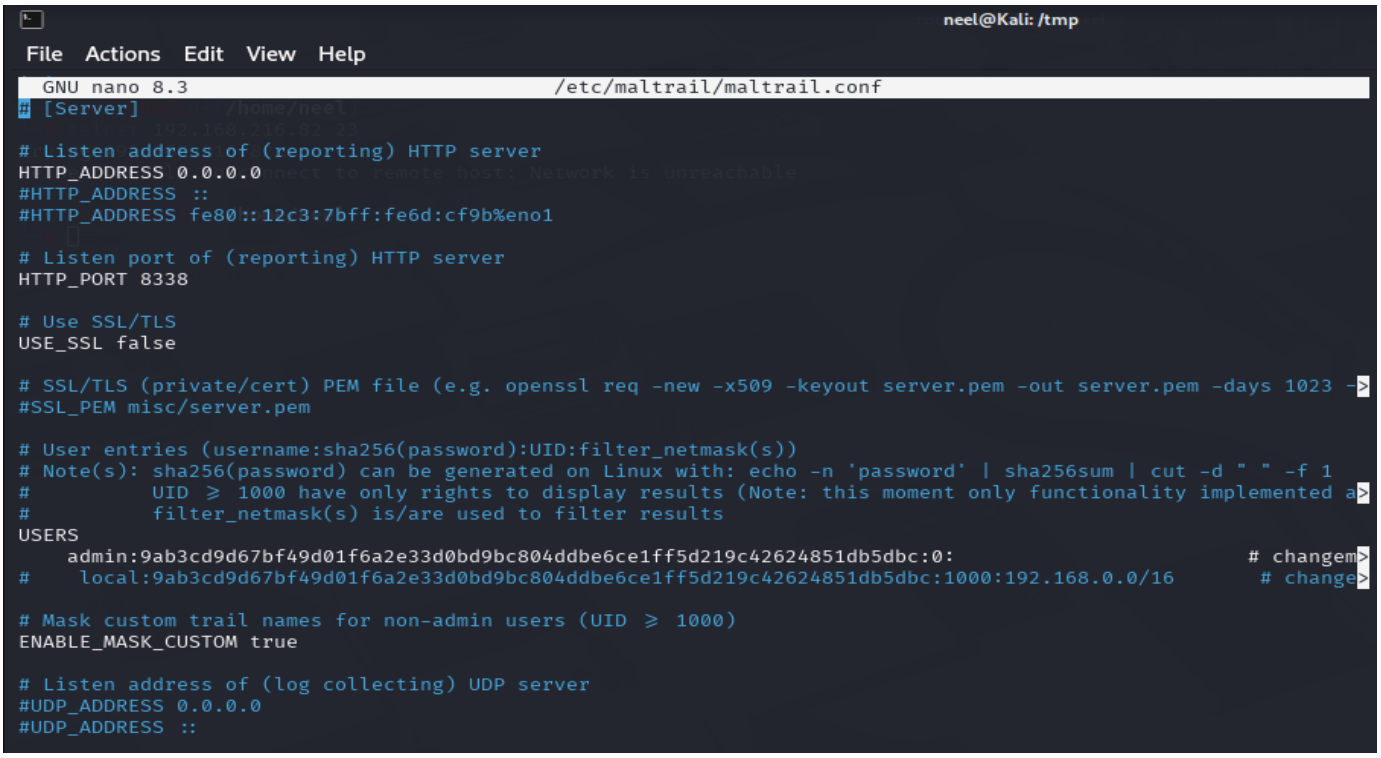
Command : **sudo nano /etc/maltrail/maltrail.conf**

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo nano /etc/maltrail/maltrail.conf
```

After hitting enter the configuration file is opened in text editer and you will then change the settings which continues in the third segment.

Traffic Detection System using Matrail

3rd Segment



After entering the maltrail.conf file as mention this line (sudo nano /etc/maltrail/ maltrail.conf) . Scroll down and go to the end need to setup of some rules as mention details below

crontab -e */5 * * * * if [-n "$(ps -ef | grep -v grep | grep 'server.py')"]; then : ; else python3 /opt/maltrail/server.py -c /etc/maltrail/maltrail.conf; fi0 1 * * * cd /opt/maltrail && git pull

Brekdown of the above code:

Line 1: */5 * * * * if [-n "$(ps -ef | grep -v grep | grep 'server.py')"]; then : ; else python3 /opt/maltrail/server.py -c /etc/maltrail/maltrail.conf; fi

- ***/5 * * * ***: This is the cron schedule. It means:
  - */5: Run every 5 minutes.
  - *: Every hour.
  - *: Every day of the month.
  - *: Every month.
  - *: Every day of the week.
- **if [-n "$(ps -ef | grep -v grep | grep 'server.py')"]; then : ; else ... fi**: This is a conditional statement in Bash:
  - ps -ef: Lists all running processes on the system.
  - grep -v grep: Excludes the grep process itself from the results.
  - grep 'server.py': Filters the process list to find any process containing 'server.py'.

- then :: If a 'server.py' process is running, do nothing (: is a no-op command).
- else python3 /opt/ maltrail/server.py -c /etc/maltrail/maltrail.conf: If no 'server.py' process is running, execute the Maltrail server script using Python 3, with the specified configuration file.

In short, this line checks every 5 minutes if the Maltrail server (server.py) is running. If it's not running, it starts it using the provided configuration file.

Line 2: 0 1 * * * cd /opt/maltrail && git pull

- **0 1 * * ***: This is the cron schedule. It means:
  - 0: At minute 0.
  - 1: At hour 1 (1 AM).
  - *: Every day of the month.
  - *: Every month.
  - *: Every day of the week.
- **cd /opt/maltrail**: Changes the current directory to /opt/maltrail.
- **git pull**: Fetches changes from the remote Git repository and tries to merge them into your current branch. This is used to update the Maltrail codebase.

In short, this line runs at 1:00 AM every day, navigates to the /opt/maltrail directory, and then attempts to update the Maltrail codebase by pulling changes from the remote Git repository.

crontab -e */1 * * * * if [-n "$(ps -ef | grep -v grep | grep 'sensor.py')"]; then : ; else python3 /opt/maltrail/sensor.py -c /etc/maltrail/maltrail.conf; fi2 1 * * * /usr/bin/pkill -f maltrail

Brekdown of the above code:

Line 3: crontab -e

- **crontab -e:** This command is used to edit your crontab file. It will open the crontab file in a text editor (usually nano or vi) so you can add, modify, or delete scheduled tasks. This line itself doesn't schedule a task; it's the command to manage your cron jobs.

In short, this line is the command you use to open and edit the list of scheduled tasks (cron jobs) for your user.

Line 4: */1 * * * * if [-n "$(ps -ef | grep -v grep | grep 'sensor.py')"]; then : ; else python3 /opt/maltrail/sensor.py -c /etc/maltrail/maltrail.conf; fi

- ***/1 * * * *: This is the cron schedule. It means:**
  - ***/1: Run every 1 minute.**
  - ***: Every hour.**
  - ***: Every day of the month.**
  - ***: Every month.**

- ***: Every day of the week.**

- **if [-n "$(ps -ef | grep -v grep | grep 'sensor.py')"]; then : ; else ... fi: This is a conditional statement in Bash, similar to the first line, but it checks for a process containing 'sensor.py'.**
- **else python3 /opt/ maltrail/sensor.py -c /etc/maltrail/maltrail.conf: If no 'sensor.py' process is running, execute the Maltrail sensor script using Python 3 with the specified configuration file.**

In short, this line checks every minute if the Maltrail sensor (sensor.py) is running. If it's not running, it starts it using the provided configuration file.

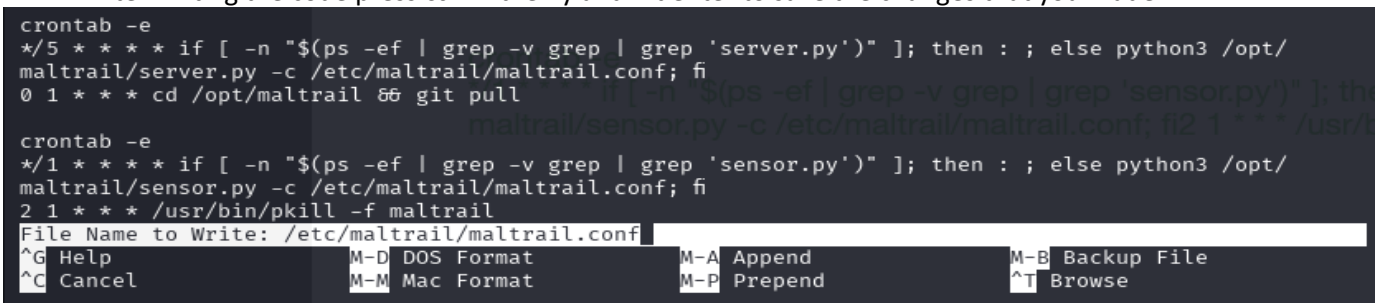
Line 5: 2 1 * * * /usr/bin/pkill -f maltrail

- **2 1 * * *: This is the cron schedule. It means:**
  - **2:** At minute 2.
  - **1:** At hour 1 (1 AM).
  - ***:** Every day of the month.
  - ***:** Every month.
  - ***:** Every day of the week.
- **/usr/bin/pkill -f maltrail:** This command is used to kill processes whose full command line matches the pattern 'maltrail'.
  - **/usr/bin/pkill:** The path to the pkill command**.**
  - **-f:** Matches against the full command line, not just the process name**.**
  - **maltrail:** The pattern to search for in the command line.

In short, this line runs at 1:02 AM every day and forcefully terminates any running processes that have 'maltrail' in their full command line. This is likely a scheduled restart of the Maltrail system.

Traffic Detection System using Matrail

After writing the code press **ctrl+x** then y and hit enter to save the changes that you made.

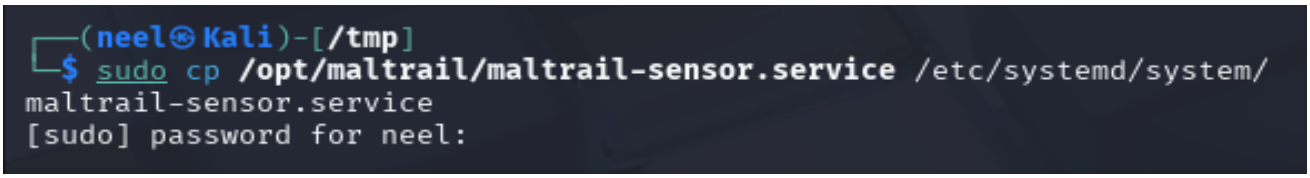


3rd segment is also completed now in the last segment copying the sensor, and the server to the right paths, and then starting the services.

4th segment

In this segment copying the opt directory it's likely a systemd service unit file for the Maltrail sensor, containing instructions on how to manage the sensor as a service (start, stop, restart, etc.) to the d**estination directory** and the name for the copied file. /etc/systemd/system/ is the standard location for user-defined or locally installed systemd service unit files.

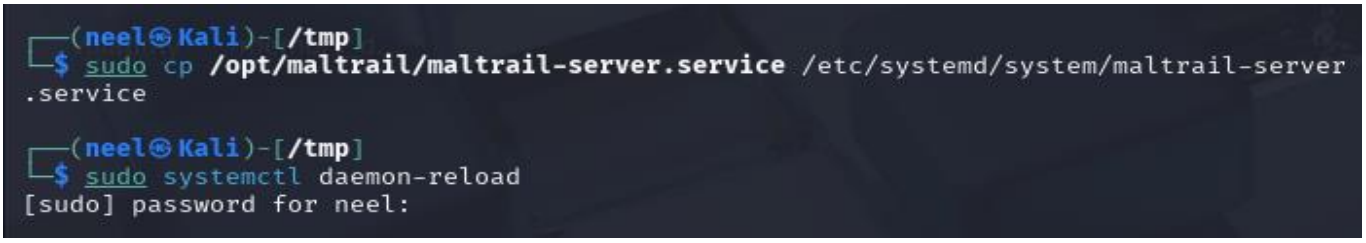
Command : **sudo cp /opt/maltrail/maltrail-sensor.service /etc/systemd/system/maltrail-sensor.service**



In short, this command uses administrator rights to copy the Maltrail sensor's **systemd** service definition file from the Maltrail installation directory in /opt to the **systemd** service configuration directory **in /etc/systemd/system/.** This makes **systemd** aware of the Maltrail sensor service, allowing you to manage it using commands like **systemctl.**

Moving to next command it uses administrator rights to copy the Maltrail server's in the previous step we done this for Maltrail sensor systemd service definition file from the Maltrail installation directory in /opt to the systemd service configuration directory in /etc/systemd/system/. This allows systemd to manage the Maltrail server as a service using commands like systemctl.

Command : **sudo cp /opt/maltrail/maltrail-server.service /etc/systemd/system/maltrail-server.service**



After copying the Maltrail server's directory to the **systemd/system** directory now run the maltrail's server as a service using systemctl command also reload it's unit files. Unit files are the configuration files that describe how services should be managed.

Traffic Detection System using Matrail

Command: **sudo systemctl daemon-reload**

Now start the service unit file of maltrail server which will initiate the Maltrail server application using the command: **sudo systemctl start maltrail-server.service** in the server like an database where the modules are stored

Also start the service unit file that defines how the Maltrail sensor should be started and managed by typing the command: **sudo systemctl start maltrail-sensor.service** and the sensor will pick the suspicious traffic coming publicly.

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo systemctl start maltrail-server.service
[sudo] password for neel:

┌──(neel㉿Kali)-[/tmp]
└─$ sudo systemctl start maltrail-sensor.service
```

This will notify the maltrail to scan it's database to check the IP address or the incoming communication has a signature which it has stored or reported by various online resources.

After enabling the necessary services that are required for the server and the sensor configure the Maltrail server and sensor services to automatically start whenever your system boots up. This ensures that the Maltrail server and sensor will be running without you having to manually start it after each reboot. Type the following commands to run the services as shown in the below picture.

```
┌──(neel㉿Kali)-[/tmp]
└─$ sudo systemctl enable maltrail-server.service
Created symlink '/etc/systemd/system/multi-user.target.wants/maltrail-server.service'
 → '/etc/systemd/system/maltrail-server.service'.

┌──(neel㉿Kali)-[/tmp]
└─$ sudo systemctl enable maltrail-sensor.service
Created symlink '/etc/systemd/system/multi-user.target.wants/maltrail-sensor.service'
 → '/etc/systemd/system/maltrail-sensor.service'.
```

Check the status if the services are running properly or not by simply type the command:

systemctl status maltrail-server.service && systemctl status maltrail-sensor.service

```
┌──(neel㉿Kali)-[/tmp]
└─$ systemctl status maltrail-server.service && systemctl status maltrail-sensor.service
● maltrail-server.service - Maltrail. Server of malicious traffic detection system
     Loaded: loaded (/etc/systemd/system/maltrail-server.service; enabled; preset: disabled)
     Active: active (running) since Wed 2025-05-14 11:05:31 IST; 17min ago
 Invocation: dd061b62a2334c8ca8057096e5d7c2a8
       Docs: https://github.com/stamparm/maltrail#readme
             https://github.com/stamparm/maltrail/wiki
   Main PID: 262638 (python3)
      Tasks: 1 (limit: 4548)
     Memory: 18M (peak: 20M)
        CPU: 425ms
     CGroup: /system.slice/maltrail-server.service
             └─262638 /usr/bin/python3 server.py

May 14 11:05:31 Kali systemd[1]: Started maltrail-server.service - Maltrail. Server of malicio>
May 14 11:05:32 Kali python3[262638]: /opt/maltrail/core/httpd.py:189: SyntaxWarning: invalid >
May 14 11:05:32 Kali python3[262638]:   content = re.sub(b'\s*<script[^>]+src="js/demo.js"></s>
lines 1-16/16 (END)
```

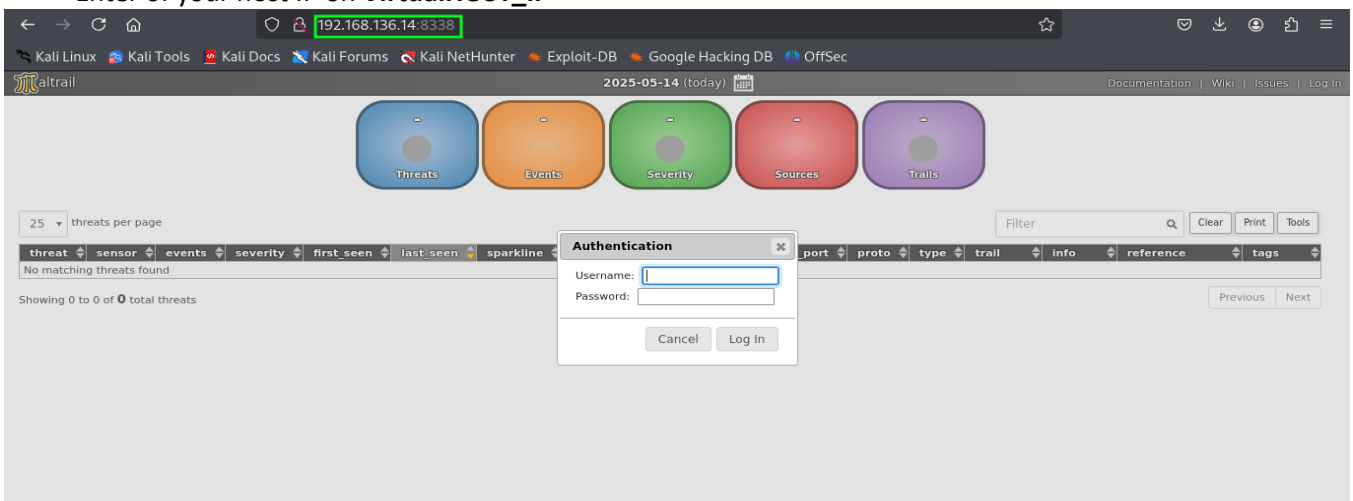
You'll get information as shown in the above image it shows the services are enabled and active properly. Fourth segment done successfully now it's time to assess or test the maltail.

Traffic Detection System using Matrail

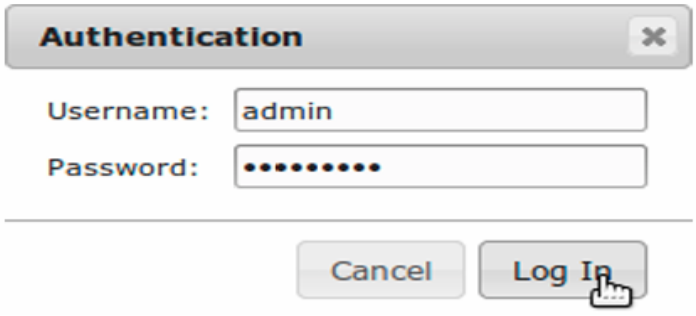
Check IP address of your host machine with this command **Ifconfig** open with another terminal

```
┌──(neel㉿Kali)-[~]
└─$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.2
55
        ether 02:42:35:a8:bb:8a  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 138 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.136.14  netmask 255.255.255.0  broadcast 192.16
8.136.255
        inet6 [redacted]  prefixlen 64  s
copeid 0x0<global>
        inet6 [redacted]  prefixlen 64  scopeid 0x20<l
ink>
        inet6 [redacted]  prefixlen 64
scopeid 0x0<global>
        ether 08:00:27:7a:9d:c8  txqueuelen 1000  (Ethernet)
        RX packets 125094  bytes 140546364 (134.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 77764  bytes 10294441 (9.8 MiB)
        TX errors 0  dropped 27 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 2629  bytes 249652 (243.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2629  bytes 249652 (243.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

After all four segment : DONE , it means that the dashboard has been generated against with the host machine HTTP Server Example : **http://0.0.0.0:8339/ or 0.0.0.0:8338** or **(VirtualHOST_IP:8338)**
Enter of your host IP on **VirtualHOST_IP**



You see the dialogue box that notify to proceed with authentication



Username : **admin**
Password : **changeme!**

Traffic Detection System using Matrail

Testing Phase - I

Open the browser in your host machine and type these two commands

nslookup morphed.ru

the nslookup tool to find the IP address and other DNS information for the morphed.ru domain. It will show you the IP address where the domain is hosted, and potentially other information like mail servers or name servers.

cat /var/log/maltrail/$(date +"%Y-%m-%d").log

this command displays the contents of the Maltrail log file for the current day. It dynamically constructs the filename using the current date in YYYY-MM-DD format.Now these are the default commands provided by the maltrail to test the application for developers.

```
┌──(neel㉿Kali)-[~]
└─$ nslookup morphed.ru
Server:         2001:4490:888:41cf::1c
Address:        2001:4490:888:41cf::1c#53

Non-authoritative answer:
Name:   morphed.ru
Address: 52.11.240.239
```

```
┌──(neel㉿Kali)-[~]
└─$ cat /var/log/maltrail/2025-05-14.log
"2025-05-14 12:15:23.984456" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 39254 2001:4490:888:41cf::1c 53 UDP DNS m
orphed.ru "andromeda (malware)" (static)
"2025-05-14 12:16:24.620470" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 49015 2001:4490:888:41cf::1c 53 UDP DNS m
orphed.ru "andromeda (malware)" (static)
```

These log lines indicate that on May 14, 2025, at the specified times, Maltrail detected DNS queries for the domain morphed.ru. Based on its threat intelligence, Maltrail classified this activity as being associated with the "andromeda" malware (identified through static signatures). The communication involved UDP protocol on ports 39254 and 53 between the given IPv6 addresses.

After doing 1st trial run successfully and check on dashboard and make refresh the website server.

As you can see in the dashboard it shows

#This code Severity level is : HIGH

#Trail : morphed.ru

#Reference : Static

#Type : DNS

It Means that it run : Well!

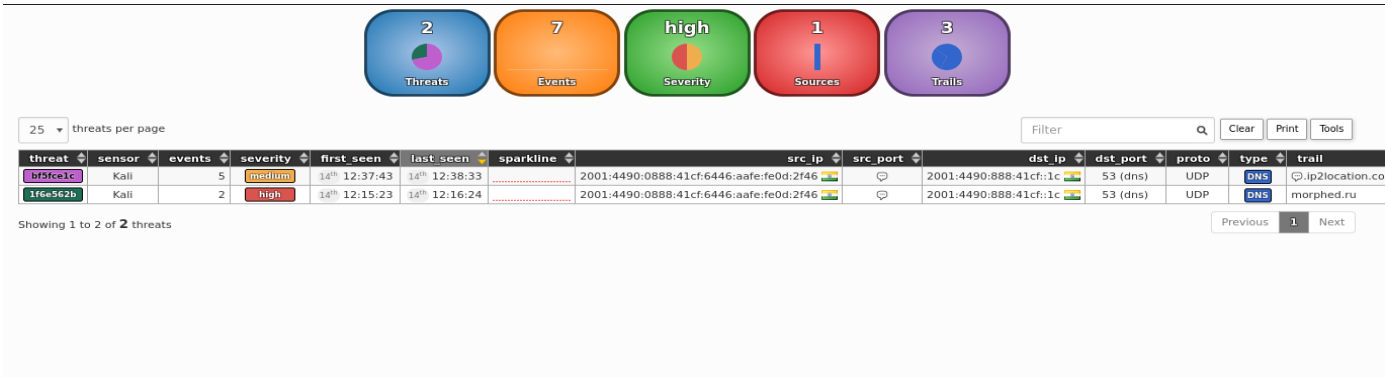
CFIS| Behala Govt Polytechnic Date : 27th Aug 2024

Traffic Detection System using Matrail

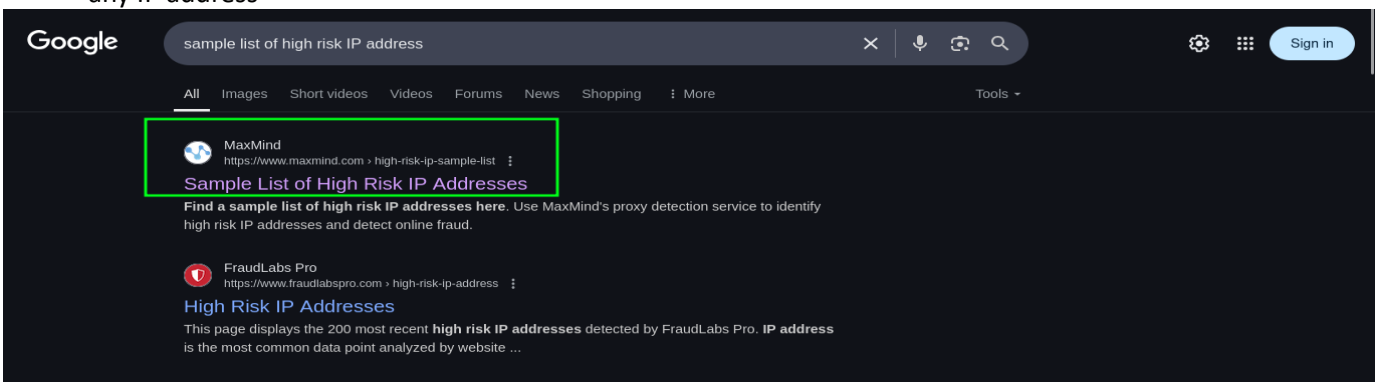
Test Phase 2

Search : Any Iplocation service as **ip2location.com** in your browser

Go back to the maltrail dashboard refresh it and you will see the Maltrail dashboard shows that your "Kali" sensor has detected two threats. One is of medium severity related to DNS activity with lj2location.co, and the other is of high severity related to DNS activity with morphed.ru, which is previously Maltrail has identified as being associated with the "andromeda (malware)" family. The timestamps indicate when these threats were first and last observed. The network details (source and destination IPs and ports, protocol) provide context for the detected activity.



Search on the browser **sample list of high risk IP addresses** and click on the first site link and select any IP address

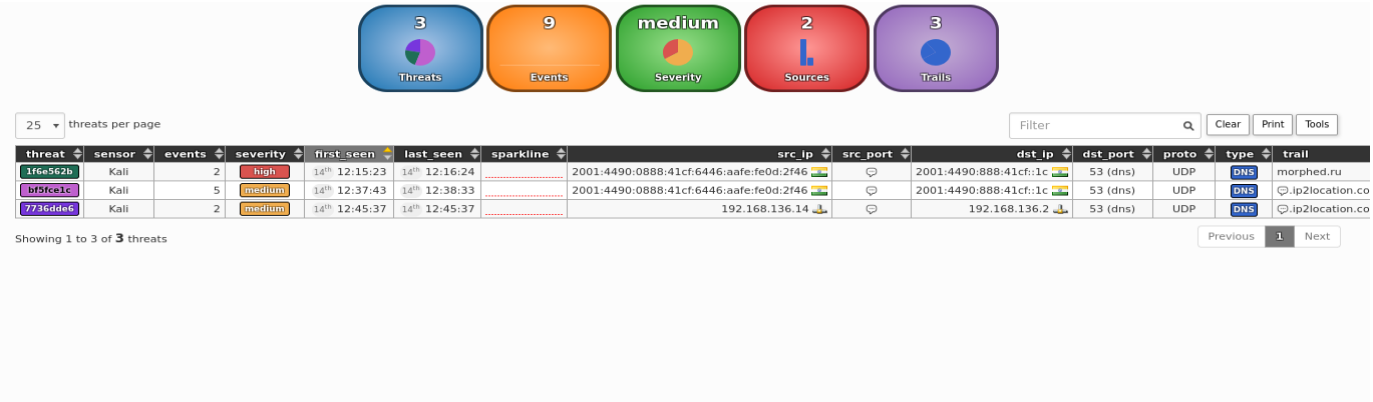


In this case using this IP address **173.165. 212.162** for this test go to terminal type the command

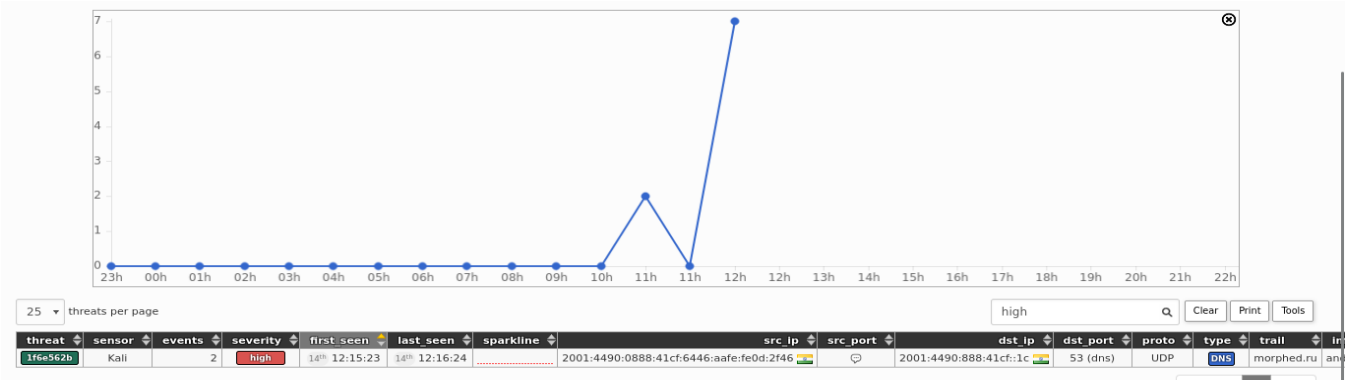
```
┌──(neel㉿Kali)-[~]
└─$ ping -c 1 68.107.77.145
cat /var/log/maltrail/2025-05-14.log
PING 68.107.77.145 (68.107.77.145) 56(84) bytes of data.
64 bytes from 68.107.77.145: icmp_seq=1 ttl=47 time=405 ms

--- 68.107.77.145 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 405.208/405.208/405.208/0.000 ms
"2025-05-14 12:15:23.984456" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 39254 2001:4490:888:41cf::1c 53 UDP DNS morphed.ru "and
romeda (malware)" (static)
"2025-05-14 12:16:24.620470" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 49015 2001:4490:888:41cf::1c 53 UDP DNS morphed.ru "and
romeda (malware)" (static)
"2025-05-14 12:37:43.760864" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 39514 2001:4490:888:41cf::1c 53 UDP DNS ip2location.com
 "ipinfo (suspicious)" (static)
"2025-05-14 12:37:44.488301" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 44330 2001:4490:888:41cf::1c 53 UDP DNS (www).ip2locati
on.com "ipinfo (suspicious)" (static)
"2025-05-14 12:37:45.531990" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 43403 2001:4490:888:41cf::1c 53 UDP DNS (cdn).ip2locati
on.com "ipinfo (suspicious)" (static)
"2025-05-14 12:38:32.788848" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 49491 2001:4490:888:41cf::1c 53 UDP DNS (www).ip2locati
on.com "ipinfo (suspicious)" (static)
"2025-05-14 12:38:33.832113" Kali 2001:4490:0888:41cf:6446:aafe:fe0d:2f46 46060 2001:4490:888:41cf::1c 53 UDP DNS (cdn).ip2locati
on.com "ipinfo (suspicious)" (static)
"2025-05-14 12:45:37.379956" Kali 192.168.136.14 46374 192.168.136.2 53 UDP DNS (www).ip2location.com "ipinfo (suspicious)" (stat
ic)
"2025-05-14 12:45:37.484485" Kali 192.168.136.14 49746 192.168.136.2 53 UDP DNS (cdn).ip2location.com "ipinfo (suspicious)" (stat
ic)
```

Traffic Detection System using Matrail

You successfully pinged the IP address 68.107.77.145 with a round-trip time of 405 milliseconds. Below that, the Maltrail logs for today continue to show detections of potentially malicious DNS activity involving morphed.ru and suspicious activity related to i.p2location.com.

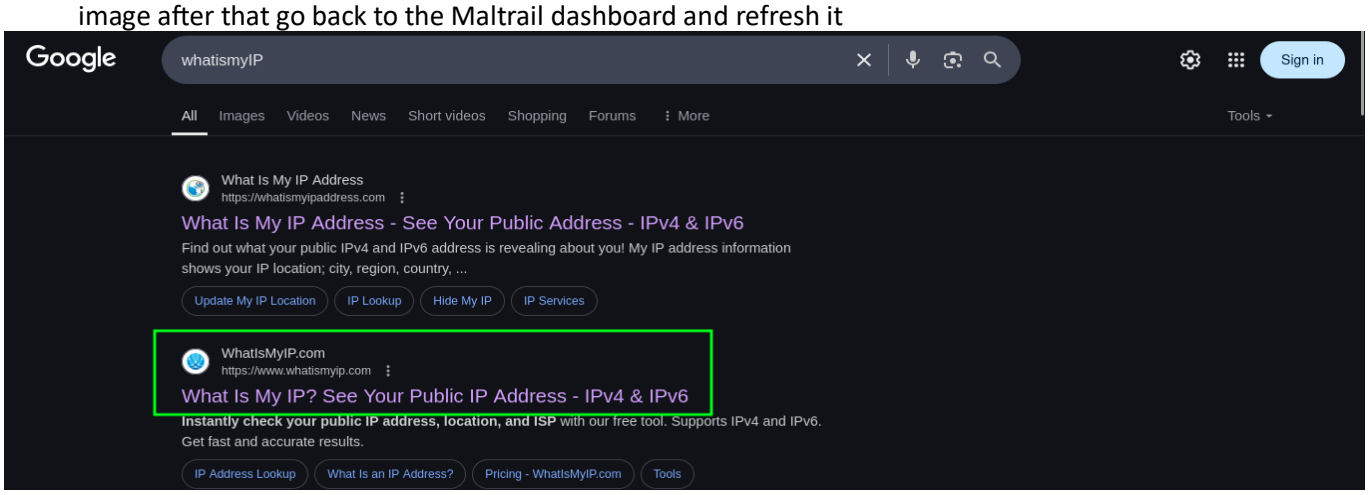


But you can't see any kind of updation in the dashboard because **Maltrail is behaving as expected by not flagging your ICMP ping scan as a threat.** It's designed to identify more specific types of potentially malicious communication based on its threat intelligence. Standard network diagnostic tools like ping generally don't trigger these alerts unless they are part of a larger, more suspicious pattern (like a flood attack, which involves sending many pings).

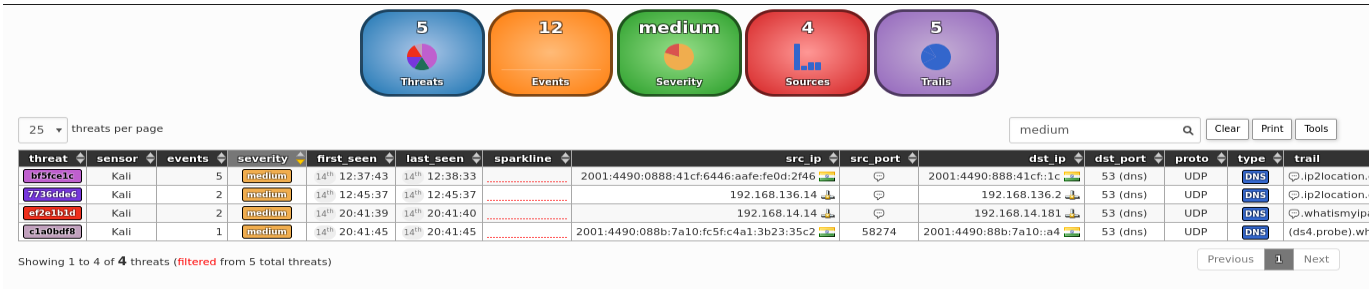


The graph shows the timeline of the "high" severity threat related to morphed.ru. There was no activity until around 11:00, a small burst at 11:00, a peak of 7 events at 12:00, and then no further detections for the rest of the displayed day. This visualizes when the malicious activity associated with this threat was most active.

In the browser open a new tab and type **whatismyIP** and click on the 2nd link as shown in below image after that go back to the Maltrail dashboard and refresh it



Traffic Detection System using Matrail



Got it you can see the Maltrail dashboard now shows 5 total threats, with 12 associated events. The highest displayed severity is "medium," and 4 different sensors have reported activity. The table is filtered to show the 4 "medium" severity threats, detailing the timestamps, network information (source and destination IPs and ports, protocol), and the specific threat intelligence trails that were matched for each.

Test Phase 3

It define another ipmachine that define from outsource test Generate Ping test Some Testing (MENTION_IP) , replace on (MENTION_IP) below main code

- 192.42.116.175
- 192.42.116.219
- 185.220.100.243

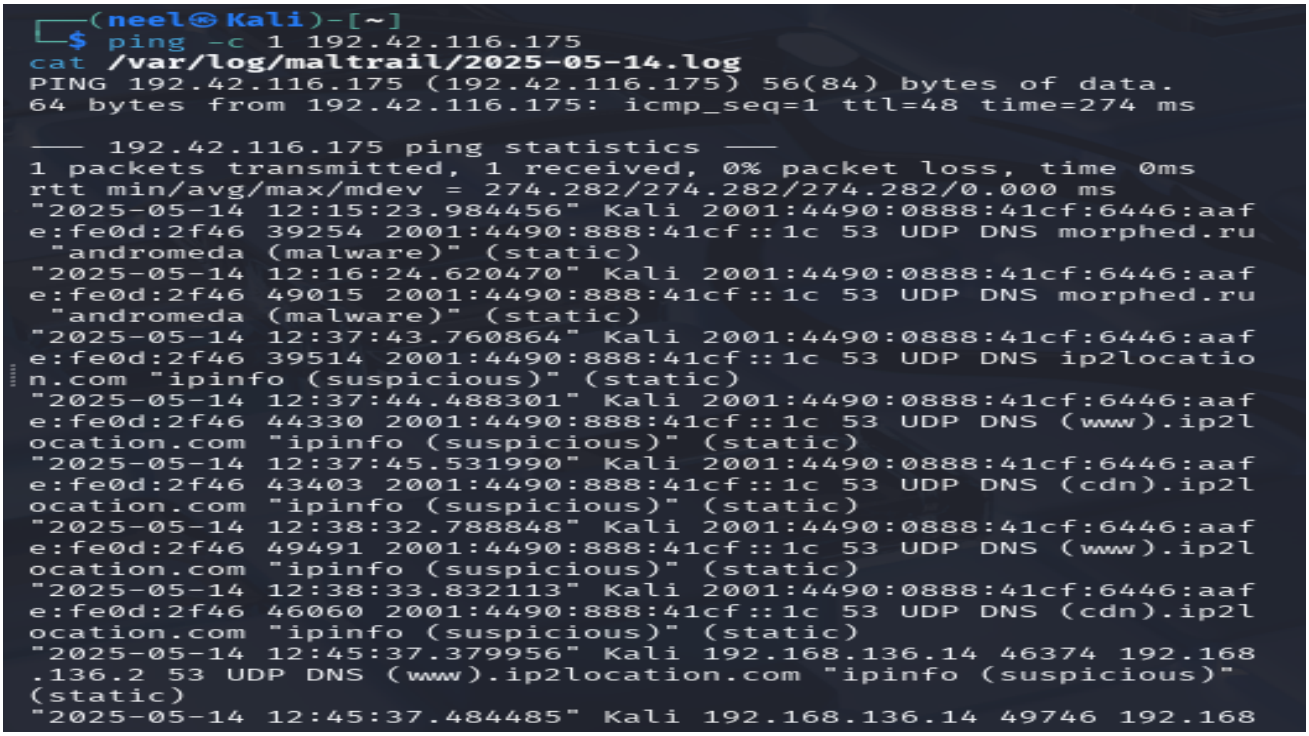
Caution dont try this ip's on HOST_MACHINE , It may damaged or Trojan or Malicious activity possibility.

Test some malicious IP Looping Reverse Thread (IPLRT)

ping -c 1 (MENTION_IP)

cat /var/log/maltrail/$(date +"%Y-%m-%d").log

Type this code on terminal as it is just change the IP address as shown in the below image.



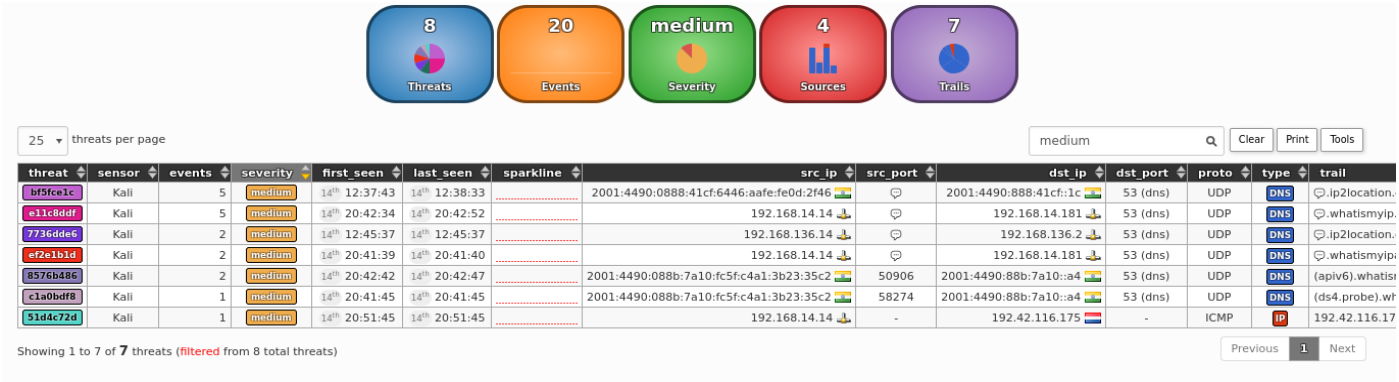
Traffic Detection System using Matrail

Ping to 192.42.116.175: This indicates a successful ping attempt to the IP address 192.42.116.175. It shows that the host is reachable and the round-trip time (rtt) was 274 ms. The ping statistics confirm 1 packet was transmitted and received with 0% packet loss.

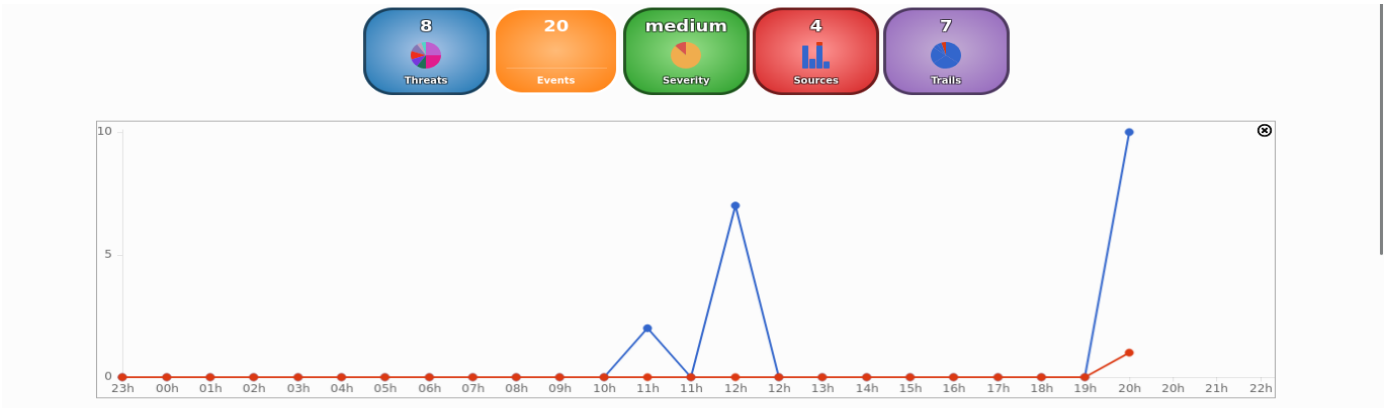
Source and Destination Information: Each line includes IPv6 addresses (starting with "fe80" or "2001") and sometimes IPv4 addresses (like "192.168.136.14" and "192.168.136.2"). It also includes port numbers (e.g., "39254", "49015", "53").

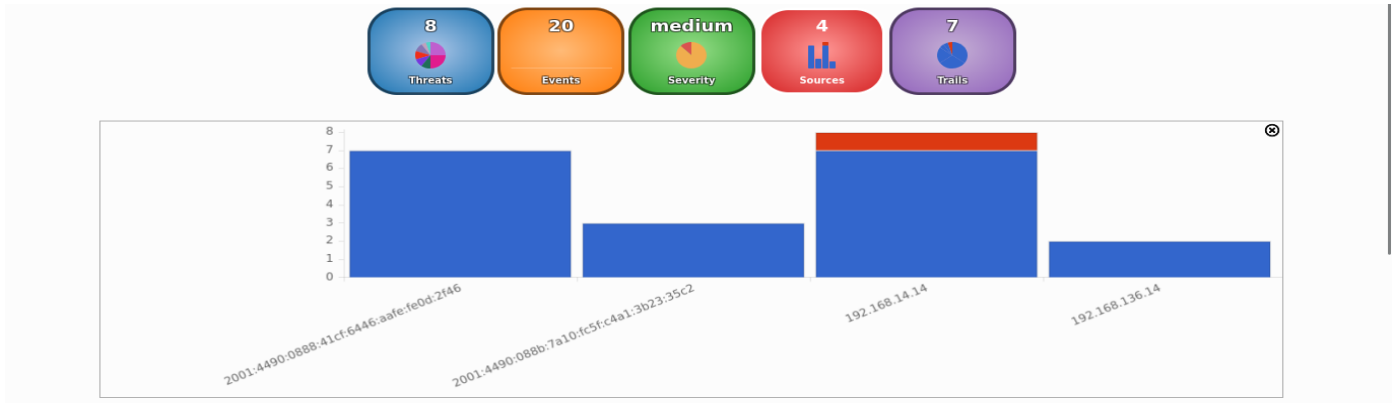
Protocol: The protocol used is "UDP DNS".

Flags: Keywords like "(malware)" and "(suspicious)" indicate how the traffic was classified.



Now go back to the maltrail dashboard and refresh the page It displays summary statistics: 8 Threats, 20 Events, medium Severity (likely an aggregated severity level), 4 Sources of these events, and 7 Trails (possibly related sequences of events).

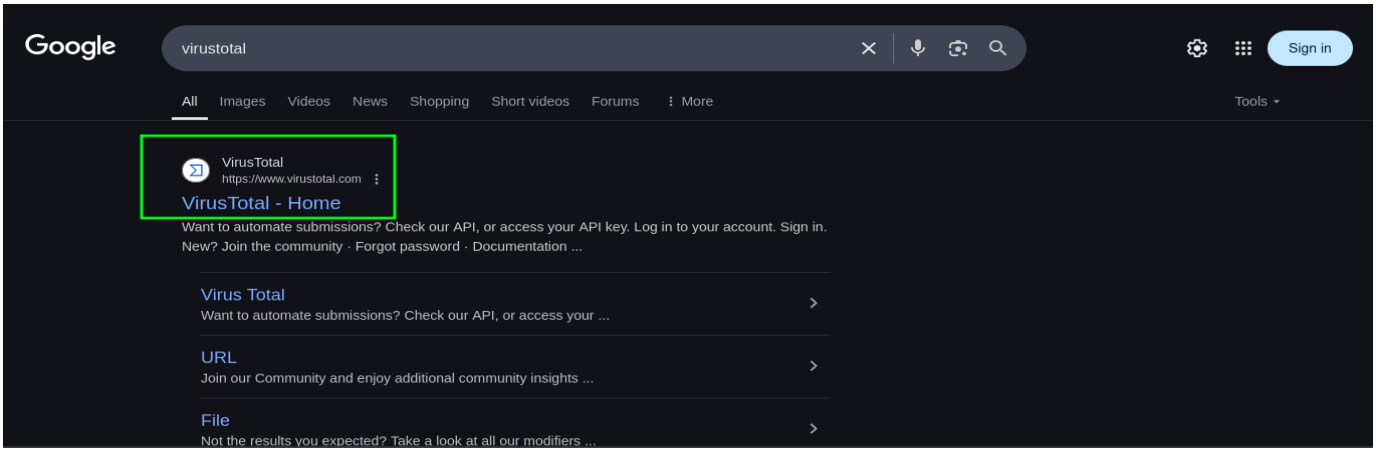




Traffic Detection System using Matrail

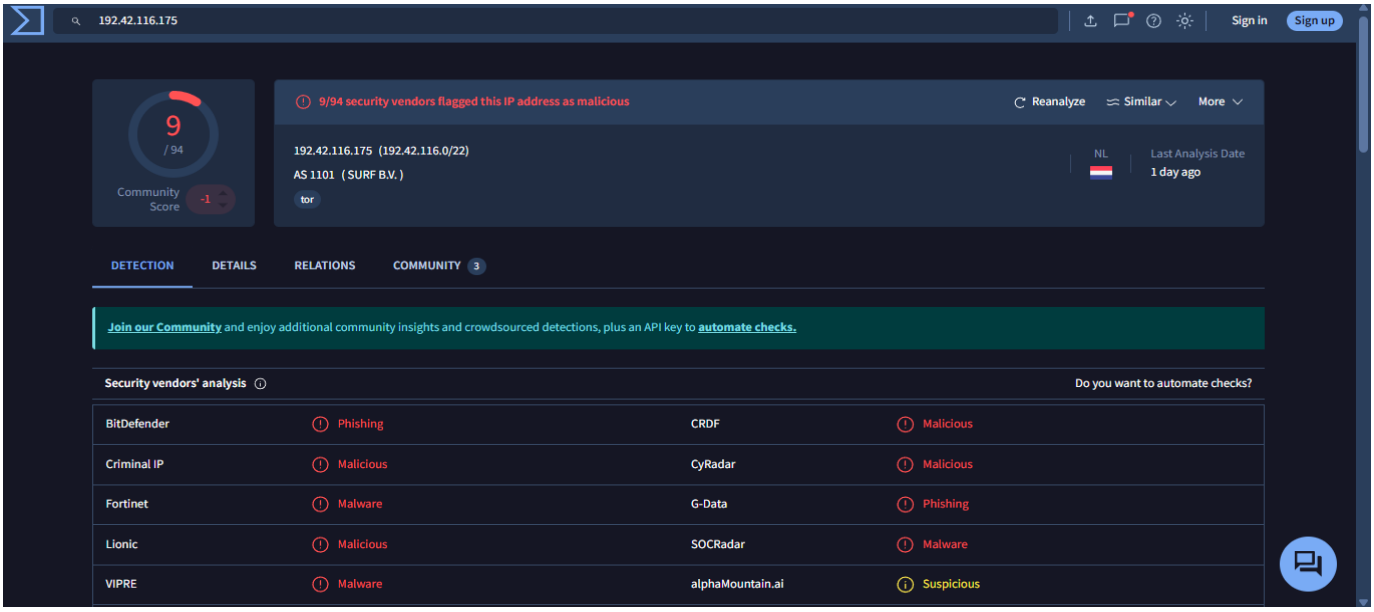
Test Phase 4

Check above IP's on VIRUSTOTAL website



Type the any of the above IP address hit search an you will find the risk score and other related information

Traffic Detection System using Matrail



As you can see which websites are reported this IP address as malicious.