

THE UNIVERSITY OF GEORGIA

Department of Computer Science



**UNIVERSITY OF
GEORGIA**

MAJOR PROJECT REPORT

Semester: Spring 2025

Real-Time Device Intelligence and Monitoring through Grafana Dashboards

Prepared by:

Name: Vishal Gandla

UGA ID: 811405063

Under the guidance of:

Dr. Roberto Perdisci

School of Computing, University of Georgia

Table of Contents

1. Abstract
2. Introduction
3. System Architecture
4. Implementation and Development
5. Results and Discussion
6. Conclusion and Future Work

1. Abstract

In the contemporary digital landscape, effective monitoring of network traffic and device statuses is crucial for ensuring the reliability and security of IT infrastructures. The need for real-time insights into DNS query activities and device health has led to the developing of an integrated data visualisation system using Grafana, a powerful open-source analytics platform, coupled with Mysql as the backend data storage solution. This report discusses the design, architecture, and implementation of a network monitoring system that collects and visualises DNS query data and device status information.

The system allows users to track DNS query patterns, monitor device availability, and detect potential issues, such as DNS misconfigurations, network congestion, or security threats. By leveraging Grafana's advanced querying capabilities, the system dynamically pulls data from the Mysql database, providing users with interactive dashboards to view DNS query counts, device statuses, and trends over specified periods. The Mysql database stores device status logs and DNS query information, optimised for efficient querying and analysis.

SQL queries play a central role in retrieving and aggregating relevant data from the database, which is then displayed in Grafana through various types of visualisations, such as bar charts, tables. This integration facilitates actionable insights for network administrators and IT professionals, empowering them to quickly identify anomalies and respond proactively.

The report further delves into application scenarios where the system can be applied, such as DNS query monitoring for cybersecurity purposes, device uptime tracking for network administration, and troubleshooting network issues. The future development roadmap includes incorporating additional data sources, implementing more sophisticated data aggregation methods, and enhancing the alerting system to improve real-time monitoring capabilities.

By offering a comprehensive and scalable solution for DNS and device monitoring, this system demonstrates the potential of Python, Grafana and Mysql as a powerful combination for real-time network analysis. As networks grow in complexity and scale, the need for effective visualisation and monitoring tools becomes more critical, and this system aims to address these challenges while providing a solid foundation for future improvements and expansions.

2. Introduction

2.1 Background

In the modern digital era, the explosion of smart devices and Internet of Things (IoT) technology has transformed the way homes, workplaces, and industries operate. Devices like smart speakers, security cameras, smart plugs, and thermostats are continuously exchanging data across networks, often using protocols such as Multicast DNS (mDNS) to advertise and discover services within local networks without manual configuration. While mDNS simplifies network management, it also opens new avenues for observing device behaviour through network traffic analysis.

Fingerprinting — the process of identifying a device based on its communication behaviour — has become an essential technique in network diagnostics and security. Passive fingerprinting involves silently listening to mDNS packets broadcast on the network, while active fingerprinting sends specific queries to trigger identifiable responses. Both methods offer insights into the types of devices connected to a network and how they behave over time.

To capitalise on this opportunity, the project integrates a combination of packet capture analysis, fingerprinting techniques, and visual dashboards to monitor and interpret device behaviour. A user-friendly Python GUI and Grafana dashboards built on MySQL further enhance data interaction, making technical insights accessible to non-experts.

2.2 The Problem

As networks grow in complexity and device diversity, administrators and security analysts face growing challenges:

- **Lack of visibility** into what types of devices are communicating on the local network.
- **Difficulty identifying suspicious behaviour**, especially when dealing with encrypted traffic or diverse vendor implementations.
- **No intuitive visualisation tools** that can combine low-level packet analysis with higher-level insights for decision-making.
- **Fragmented data monitoring**, requiring manual correlation between raw packet captures and analytical tools.

Without robust fingerprinting and monitoring solutions, unauthorised or misconfigured devices may go unnoticed, and troubleshooting becomes reactive rather than proactive.

2.3 Objectives of the Report

The primary objectives of this report are:

- **To analyse and implement passive and active fingerprinting techniques** using the MDNS protocol to identify and categorise devices on a local network.
- **To develop a user-friendly Python-based GUI** for real-time PCAP file analysis that visualises protocol behaviour, device identity, and packet timelines.
- **To design and integrate Grafana dashboards with Mysql** to provide real-time, historical, and aggregated views of DNS queries and device health status.
- **To illustrate the system architecture and components**, including data pipelines, query systems, and visualisation strategies.
- **To demonstrate use-case scenarios** such as device behaviour monitoring, anomaly detection, and network performance analysis.
- **To provide a roadmap** for future system enhancements, including more advanced analytics, real-time streaming capabilities, and broader protocol support.

Through this project, we aim to bridge the gap between low-level network traffic and high-level insights that are meaningful for system administrators, researchers, and security professionals.

3. System Architecture

The system is designed to analyse PCAP files, extract Multicast DNS (mdns) traffic, fingerprint devices based on their behaviour and present the results through both a custom-built Python GUI and a real-time Grafana dashboard backed by a Mysql database. This layered architecture ensures both deep packet-level analysis and high-level visualisation, catering to both developers and security professionals.

3.1 Overview

The architecture is modular and consists of the following main components:

- **Packet Capture Module (PCAP Reader)**
- **Passive & Active Fingerprinting Engine**
- **Feature Extraction and Enrichment**
- **Storage Layer (Mysql Database)**
- **Visualisation Layer (Grafana Dashboards)**
- **GUI Layer (Python Tkinter or PyQt Interface)**

Each component communicates via structured data pipelines and operates on a batch or real-time basis depending on the use-case scenario.

3.2 Detailed Component Description

3.2.1 Packet Capture and Reader

This module is responsible for ingesting PCAP files using Scapy or pyshark. It parses each packet, identifies relevant MDNS traffic, and forwards it to the fingerprinting engine.

Python

```
temp.py > ...
1  import pyshark
2
3  cap = pyshark.FileCapture('network_traffic.pcap', display_filter='mdns')
4  for packet in cap:
5      # Extract packet info: src IP, domain queried, service type, etc.
6      print(packet.mdns.dns_name)
```

3.2.2 Fingerprinting Engine

- **Passive Fingerprinting:** Observes existing mdns broadcast packets to determine device characteristics based on the advertised service types (_airplay._tcp.local, _googlecast._tcp.local, etc.), Time-To-Live (TTL), MAC address vendor prefixes, etc.
- **Active Fingerprinting:** Optionally sends custom MDNS queries using Scapy to stimulate responses from devices to gather additional behavioural attributes.

python

```
from scapy.all import *

mdns_query = IP(dst="224.0.0.251")/UDP(dport=5353)/DNS(rd=1, qd=DNSQR(qname="_services._dns-sd._udp.local", qtype="PTR"))
send(mdns_query)
```

3.2.3 Feature Extraction

This module transforms raw packet fields into structured features such as:

- Device hostname
- Advertised services
- Query response delay
- TTL values
- Frequency of response
- Unique identifiers like MAC or name

The output is prepared for both insertion into the database and direct GUI visualisation.

3.3 Database Architecture (Mysql)

The Mysql database stores pre-processed results from the fingerprinting engine.

Key tables include:

- devices: stores device metadata (MAC, IP, hostname, vendor)
- queries: logs each DNS query observed
- responses: tracks DNS responses, including TTL and response time
- fingerprints: stores derived device behaviour profiles

Sample SQL Schema Snippet

```
CREATE TABLE devices (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  mac_address VARCHAR(50),  
  hostname VARCHAR(100),  
  vendor VARCHAR(100),  
  last_seen TIMESTAMP  
);  
CREATE TABLE queries (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  device_id INT,  
  query_name VARCHAR(255),  
  timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  FOREIGN KEY (device_id) REFERENCES devices(id)  
);
```

3.4 Grafana Dashboards (Visualisation Layer)

Grafana is connected to the Mysql backend using a data source connector. It visualises:

- Top 10 Most Active Devices
- Most Queried Services
- Live DNS Query Timeline
- Device Vendor Distribution

Sample SQL Query for Grafana Panel

```
SELECT hostname AS metric, COUNT(*) AS query_count FROM queries  
JOIN devices ON queries.device_id = devices.id GROUP BY hostname  
ORDER BY query_count DESC LIMIT 10;
```

These panels enable real-time observability and historical analysis of how devices behave over time.

3.5 GUI Layer (Python Desktop Application)

The GUI is developed in Python using **Tkinter**. It supports:

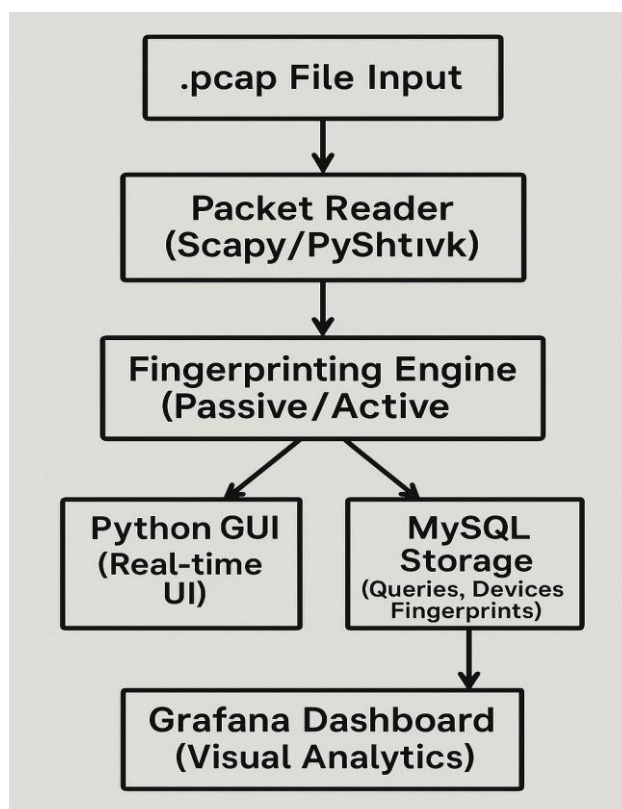
- Uploading PCAP files
- Initiating passive or active fingerprinting
- Displaying the fingerprinted device list
- Showing packet timelines and query maps

Python

```
import tkinter as tk
from tkinter import filedialog
def open_pcap():
    file_path = filedialog.askopenfilename()
    process_pcap(file_path)

root = tk.Tk()
tk.Button(root, text="Load PCAP", command=open_pcap).pack()
root.mainloop()
```

3.6 Component Interaction Flow



3.7 Design Rationale

- **Modularity:** Each component can be updated independently (e.g., swap Scapy with tshark).
- **Extensibility:** Can later include SSDP, Netbios, or DHCP fingerprinting.
- **Security-Aware:** Supports identification of rogue devices by behaviour patterns.
- **User-Friendly:** GUI + Grafana allow both novice and expert users to gain insights.

4. Implementation and Development

This chapter delves into the practical aspects of building the MDNS-based device fingerprinting and monitoring system. It outlines the development process, technologies employed, and integration strategies that culminate in a cohesive solution for network analysis and visualisation.

4.1 Development Environment










The system was developed using a combination of tools and technologies to facilitate efficient packet analysis, data storage, and visualisation:

- Programming Language: Python 3
- Libraries: scapy, pyshark, tkinter, matplotlib
- Database: Mysql
- Visualisation Tool: Grafana
- Operating System: Ubuntu

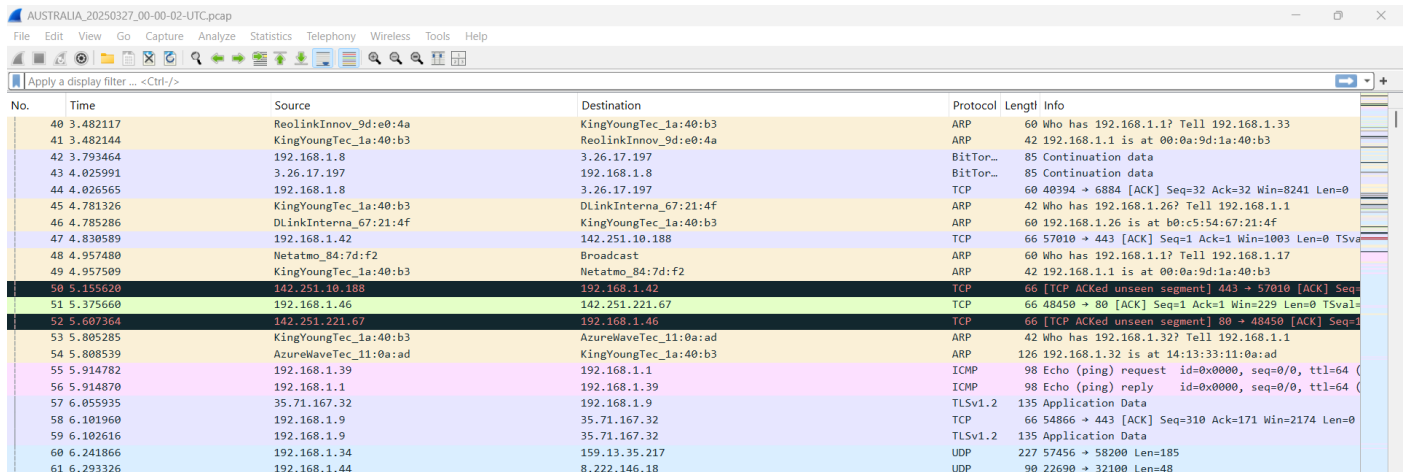
4.2 Packet Capture and Parsing

The initial step involves capturing network traffic, specifically focusing on mDNS packets. Utilising Pyshark, the system reads PCAP files and filters for MDNS traffic:

Captured PCAP Files from the Australian Network

 AUSTRALIA_20250327_00-00-02-UTC	27/03/2025 00:21	Wireshark capture ...	3,610 KB
 AUSTRALIA_20250327_00-05-02-UTC	27/03/2025 00:25	Wireshark capture ...	2,920 KB
 AUSTRALIA_20250327_00-10-02-UTC	27/03/2025 00:28	Wireshark capture ...	4,473 KB
 AUSTRALIA_20250327_00-15-02-UTC	27/03/2025 00:36	Wireshark capture ...	3,201 KB
 AUSTRALIA_20250327_00-20-02-UTC	27/03/2025 00:41	Wireshark capture ...	3,556 KB
 AUSTRALIA_20250327_00-25-02-UTC	27/03/2025 00:44	Wireshark capture ...	3,530 KB
 AUSTRALIA_20250327_00-30-02-UTC	27/03/2025 00:49	Wireshark capture ...	3,374 KB
 AUSTRALIA_20250327_00-35-02-UTC	27/03/2025 00:53	Wireshark capture ...	3,016 KB
 AUSTRALIA_20250327_00-40-02-UTC	27/03/2025 01:00	Wireshark capture ...	2,685 KB

Packets present in PCAP Files



No.	Time	Source	Destination	Protocol	Length	Info
40	3.482117	ReolinkInnov_9d:e0:4a	KingYoungTec_1a:40:b3	ARP	60	Who has 192.168.1.1? Tell 192.168.1.33
41	3.482144	KingYoungTec_1a:40:b3	ReolinkInnov_9d:e0:4a	ARP	42	192.168.1.1 is at 00:0a:9d:1a:40:b3
42	3.793464	192.168.1.8	3.26.17.197	BitTorrent	85	Continuation data
43	4.025991	3.26.17.197	192.168.1.8	BitTorrent	85	Continuation data
44	4.026565	192.168.1.8	3.26.17.197	TCP	60	40394 → 6884 [ACK] Seq=32 Ack=32 Win=8241 Len=0
45	4.781326	KingYoungTec_1a:40:b3	DLinkInterna_67:21:4f	ARP	42	Who has 192.168.1.26? Tell 192.168.1.1
46	4.785286	DLinkInterna_67:21:4f	KingYoungTec_1a:40:b3	ARP	60	192.168.1.26 is at b0:c5:54:67:21:4f
47	4.830589	192.168.1.42	142.251.10.188	TCP	66	57010 → 443 [ACK] Seq=1 Ack=1 Win=1003 Len=0 TSval=
48	4.957480	Netatmo_84:7d:f2	Broadcast	ARP	60	Who has 192.168.1.1? Tell 192.168.1.17
49	4.957599	KingYoungTec_1a:40:b3	Netatmo_84:7d:f2	ARP	42	192.168.1.1 is at 00:0a:9d:1a:40:b3
50	5.155620	142.251.10.188	192.168.1.42	TCP	66	[TCP ACKed unseen segment] 443 → 57010 [ACK] Seq=
51	5.375660	192.168.1.46	142.251.221.67	TCP	66	48450 → 80 [ACK] Seq=1 Ack=1 Win=229 Len=0 TSval=
52	5.607364	142.251.221.67	192.168.1.46	TCP	66	[TCP ACKed unseen segment] 80 → 48450 [ACK] Seq=
53	5.805285	KingYoungTec_1a:40:b3	AzureWaveTec_11:0a:ad	ARP	42	Who has 192.168.1.32? Tell 192.168.1.1
54	5.808539	AzureWaveTec_11:0a:ad	KingYoungTec_1a:40:b3	ARP	126	192.168.1.32 is at 14:13:33:11:0a:ad
55	5.914782	192.168.1.39	192.168.1.1	ICMP	98	Echo (ping) request id=0x0000, seq=0/0, ttl=64 (
56	5.914870	192.168.1.1	192.168.1.39	ICMP	98	Echo (ping) reply id=0x0000, seq=0/0, ttl=64 (
57	6.055935	35.71.167.32	192.168.1.9	TLSv1.2	135	Application Data
58	6.101960	192.168.1.9	35.71.167.32	TCP	66	54866 → 443 [ACK] Seq=310 Ack=171 Win=2174 Len=0
59	6.102616	192.168.1.9	35.71.167.32	TLSv1.2	135	Application Data
60	6.241866	192.168.1.34	159.13.35.217	UDP	227	57456 → 58200 Len=185
61	6.293326	192.168.1.44	8.222.146.18	UDP	90	22690 → 32100 Len=48

4.3 Data Storage Schema

Extracted information is structured and stored in a Mysql database.

All Database Tables to store the data:

```
mysql> show tables;
+-----+
| Tables_in_main |
+-----+
| destination_traffic |
| device_info |
| device_status |
| dns_queries_info |
| processed_pcaps |
+-----+
5 rows in set (0.00 sec)
```

```
mysql> select * from destination_traffic limit 15;
+-----+-----+-----+-----+-----+-----+-----+-----+
| id | src_ip | dest_ip | date | tcp_bytes_sent | tcp_bytes_received | udp_bytes_sent | udp_bytes_received |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | 192.168.2.37 | 54.82.96.225 | 2023-03-18 | 37731 | 0 | 0 | 0 |
| 2 | 192.168.2.17 | 51.105.242.204 | 2023-03-18 | 13185 | 0 | 0 | 0 |
| 3 | 192.168.2.15 | 203.201.60.12 | 2023-03-18 | 0 | 0 | 165787 | 0 |
| 4 | 203.201.60.12 | 192.168.2.15 | 2023-03-18 | 0 | 0 | 0 | 726897 |
| 5 | 192.168.2.33 | 157.175.147.136 | 2023-03-18 | 0 | 0 | 333622 | 0 |
| 6 | 192.168.2.25 | 34.117.13.189 | 2023-03-18 | 40580 | 0 | 0 | 0 |
| 7 | 192.168.2.34 | 157.175.147.136 | 2023-03-18 | 0 | 0 | 334144 | 0 |
| 8 | 192.168.2.15 | 47.91.10.48 | 2023-03-18 | 0 | 0 | 120662 | 0 |
| 9 | 192.168.2.15 | 172.104.127.160 | 2023-03-18 | 0 | 0 | 120954 | 0 |
| 10 | 192.168.2.15 | 139.162.76.21 | 2023-03-18 | 0 | 0 | 120954 | 0 |
| 11 | 34.117.13.189 | 192.168.2.25 | 2023-03-18 | 0 | 36354 | 0 | 0 |
| 12 | 172.104.127.160 | 192.168.2.15 | 2023-03-18 | 0 | 0 | 0 | 52662 |
| 13 | 139.162.76.21 | 192.168.2.15 | 2023-03-18 | 0 | 0 | 0 | 53412 |
| 16 | 104.198.46.246 | 192.168.2.23 | 2023-03-18 | 0 | 72832 | 0 | 0 |
| 17 | 192.168.2.23 | 104.198.46.246 | 2023-03-18 | 76408 | 0 | 0 | 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+
15 rows in set (0.01 sec)
```

```
mysql> select * from device_info limit 10;
```

id	ip_address	device_name	date	udp_bytes_sent	udp_bytes_received
1	192.168.2.37	Wemo_WiFiSmartLightSwitch	2023-03-18	3099619	2991749
2	192.168.2.17	Netatmo_SmartHomeWeatherStation	2023-03-18	465312	468664
3	192.168.2.15	Wansview_WirelessCloudcamera	2023-03-18	17182589	21807105
4	192.168.2.33	Reolink_RLC520Camera1	2023-03-18	9112491	1914300
5	192.168.2.25	Philips_HueBridge	2023-03-18	16513035	1644406
6	192.168.2.34	Reolink_RLC520Camera2	2023-03-18	9109728	1912520
7	192.168.2.23	Lifx_SmarterLights	2023-03-18	471632	445636
8	192.168.2.38	Ecobee_Switch+	2023-03-18	725842	1447546
9	192.168.2.41	Blink Mini indoor Plug-In HD smart security Camera	2023-03-18	1095176	2063643
10	192.168.2.8	Sengled_SmartBulbStarterKit	2023-03-18	756890	438830

```
10 rows in set (0.00 sec)
```

```
mysql> select * from device_status limit 10;
```

id	ip_address	device_name	last_seen_time	previous_seen_time
1	192.168.2.37	Wemo_WiFiSmartLightSwitch	2023-03-18 08:09:17	753 days ago
2	192.168.2.17	Netatmo_SmartHomeWeatherStation	2025-04-03 21:16:42	16 days ago
3	192.168.2.15	Wansview_WirelessCloudcamera	2025-04-03 21:18:37	16 days ago
4	192.168.2.33	Reolink_RLC520Camera1	2025-04-03 21:18:35	16 days ago
5	192.168.2.25	Philips_HueBridge	2025-04-03 21:18:39	16 days ago
6	192.168.2.34	Reolink_RLC520Camera2	2025-04-03 21:18:38	16 days ago
7	192.168.2.23	Lifx_SmarterLights	2025-04-03 21:18:27	16 days ago
8	192.168.2.38	Ecobee_Switch+	2025-04-03 21:18:37	16 days ago
9	192.168.2.41	Blink Mini indoor Plug-In HD smart security Camera	2023-03-18 08:09:31	753 days ago
10	192.168.2.8	Sengled_SmartBulbStarterKit	2025-04-03 21:18:30	16 days ago

```
10 rows in set (0.00 sec)
```

```
mysql> select * from dns_queries_info limit 10;
```

id	ip_address	domain_name	query_count	query_date
1	192.168.2.10	_eerogw._tcp.local.	383	2023-03-18
2	192.168.2.42	_googlezone._tcp.local.	1201	2023-03-18
3	192.168.2.11	_eerogw._tcp.local.	378	2023-03-18
4	192.168.2.42	0aca6283-e84c-383b-cdf1-96f8a60ffec6._googlezone._tcp.local.	592	2023-03-18
6	192.168.2.42	google-nest-mini-0aca6283e84c383bcd196f8a60ffec6._googlecast._tcp.local.	591	2023-03-18
7	192.168.2.42	_googlecast._tcp.local.	1189	2023-03-18
10	192.168.2.38	Office (2)._hap._tcp.local.	2692	2023-03-18
19	192.168.2.9	_sengled._udp.local.	108	2023-03-18
22	192.168.2.26	_dcp._tcp.local.	37	2023-03-18
47	192.168.2.49	Office._hap._tcp.local.	641	2023-03-18

```
10 rows in set (0.00 sec)
```

```
mysql> select * from processed_pcaps limit 10;
```

id	file_path	processed_at
1	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-00-02-UTC.pcap	2025-04-10 00:40:01
2	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-05-02-UTC.pcap	2025-04-10 00:40:42
3	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-10-02-UTC.pcap	2025-04-10 00:41:34
4	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-15-02-UTC.pcap	2025-04-10 00:42:21
5	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-20-02-UTC.pcap	2025-04-10 00:43:03
6	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-25-02-UTC.pcap	2025-04-10 00:43:50
7	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-30-02-UTC.pcap	2025-04-10 00:44:35
8	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-35-02-UTC.pcap	2025-04-10 00:45:17
9	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-40-02-UTC.pcap	2025-04-10 00:46:05
10	/home/vishall/pcaps/20250327/AUSTRALIA_20250327_00-45-02-UTC.pcap	2025-04-10 00:46:55

```
10 rows in set (0.00 sec)
```

4.4 Grafana Dashboard

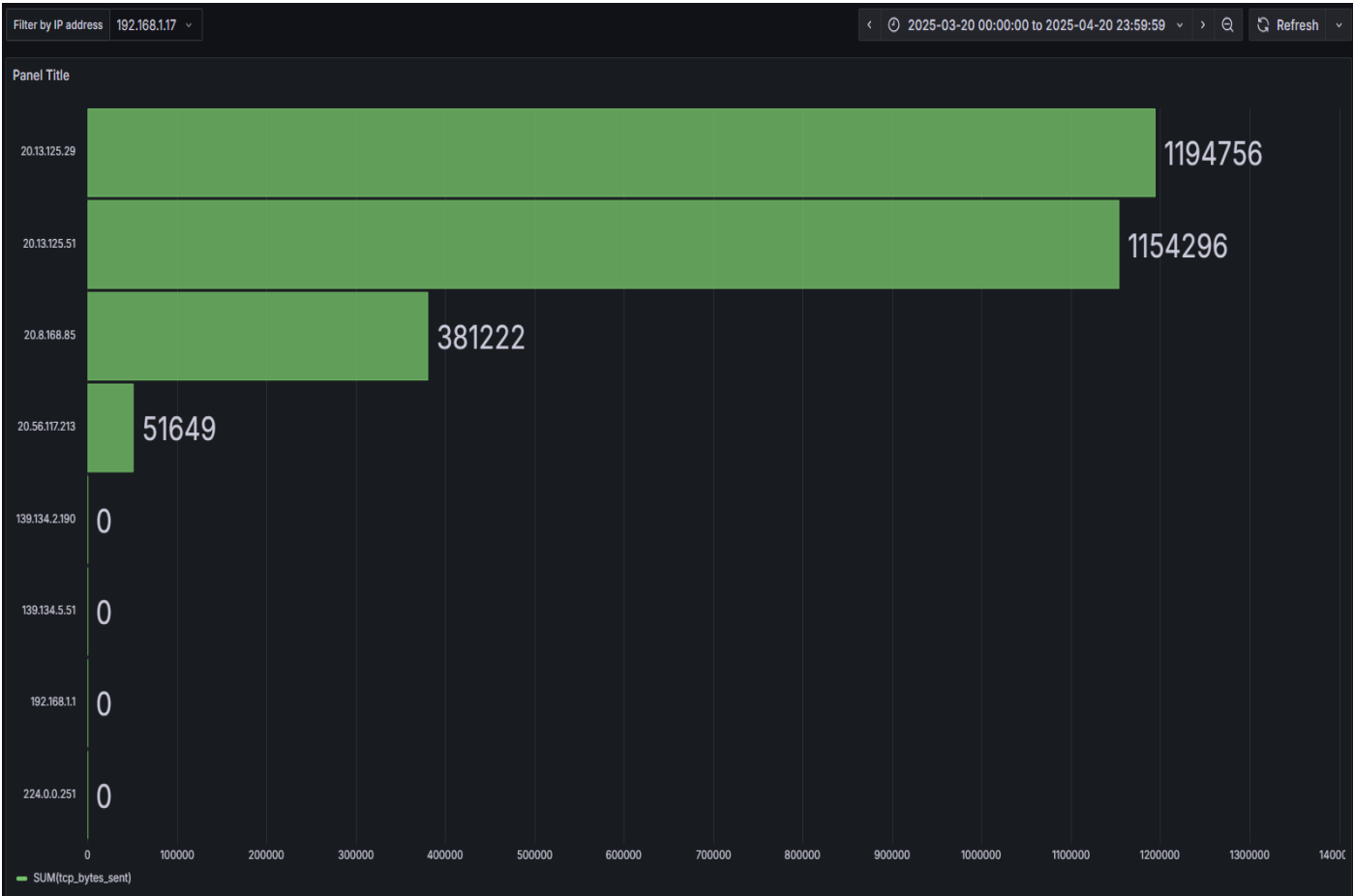
Grafana is configured to connect to the Mysql database, enabling dynamic visualisation of the collected data. Dashboards are created to display metrics such as:

- Total TCP, UDP, ICMP bytes and DNS queries of all devices
- TCP bytes sent to a destination IP address by all devices
- UDP bytes sent to a destination IP address by all devices
- All the Devices online
- DNS Queries and Distinct DNS Queries sent by all devices
- Query count to different domains by all devices
- ICMP bytes sent and received by all devices
- TCP bytes sent and received by all devices
- UDP bytes sent and received by all devices

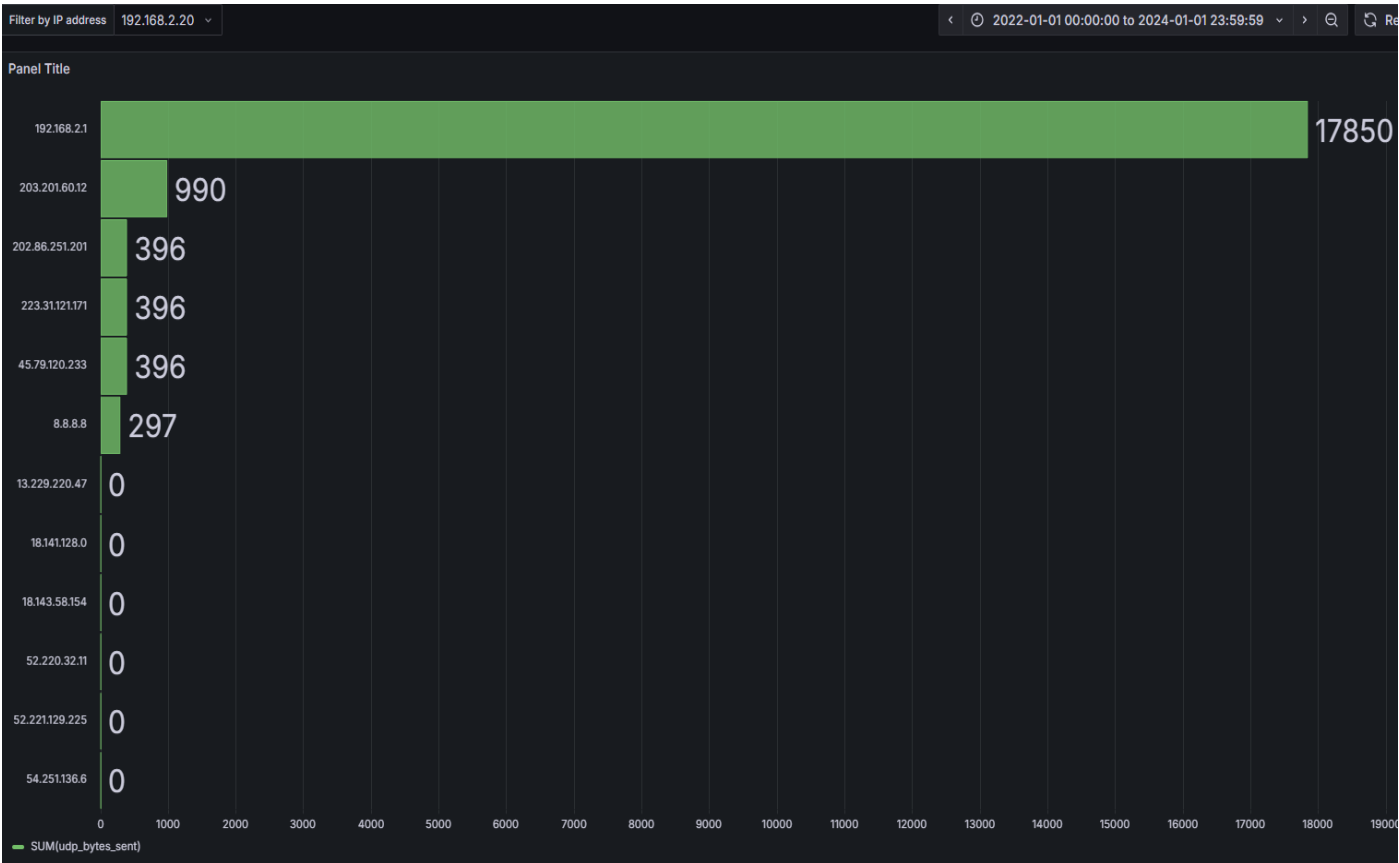
Total TCP, UDP, ICMP bytes and DNS queries of all devices:



TCP bytes sent to the destination IP address



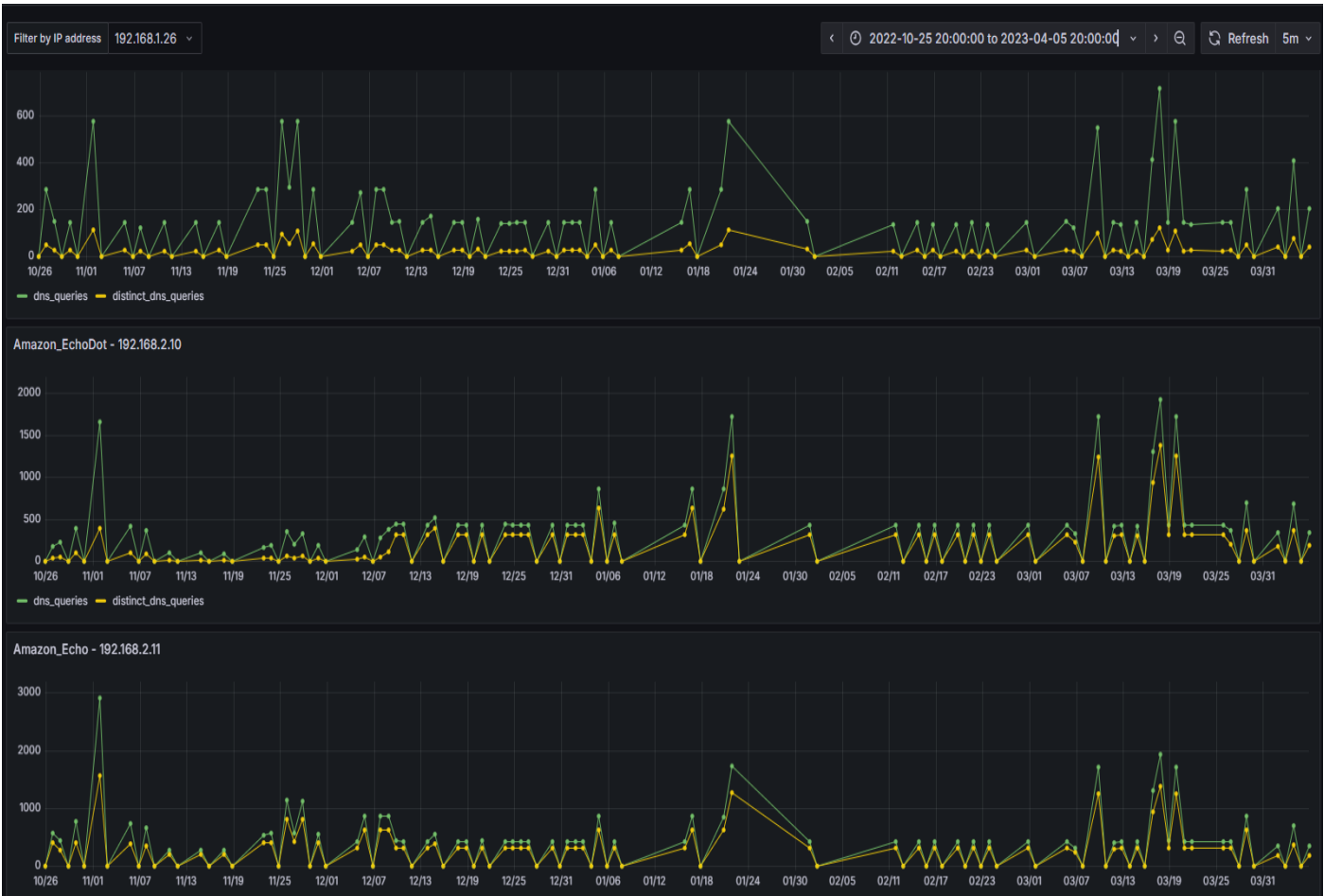
UDP bytes sent to a destination IP address by all devices



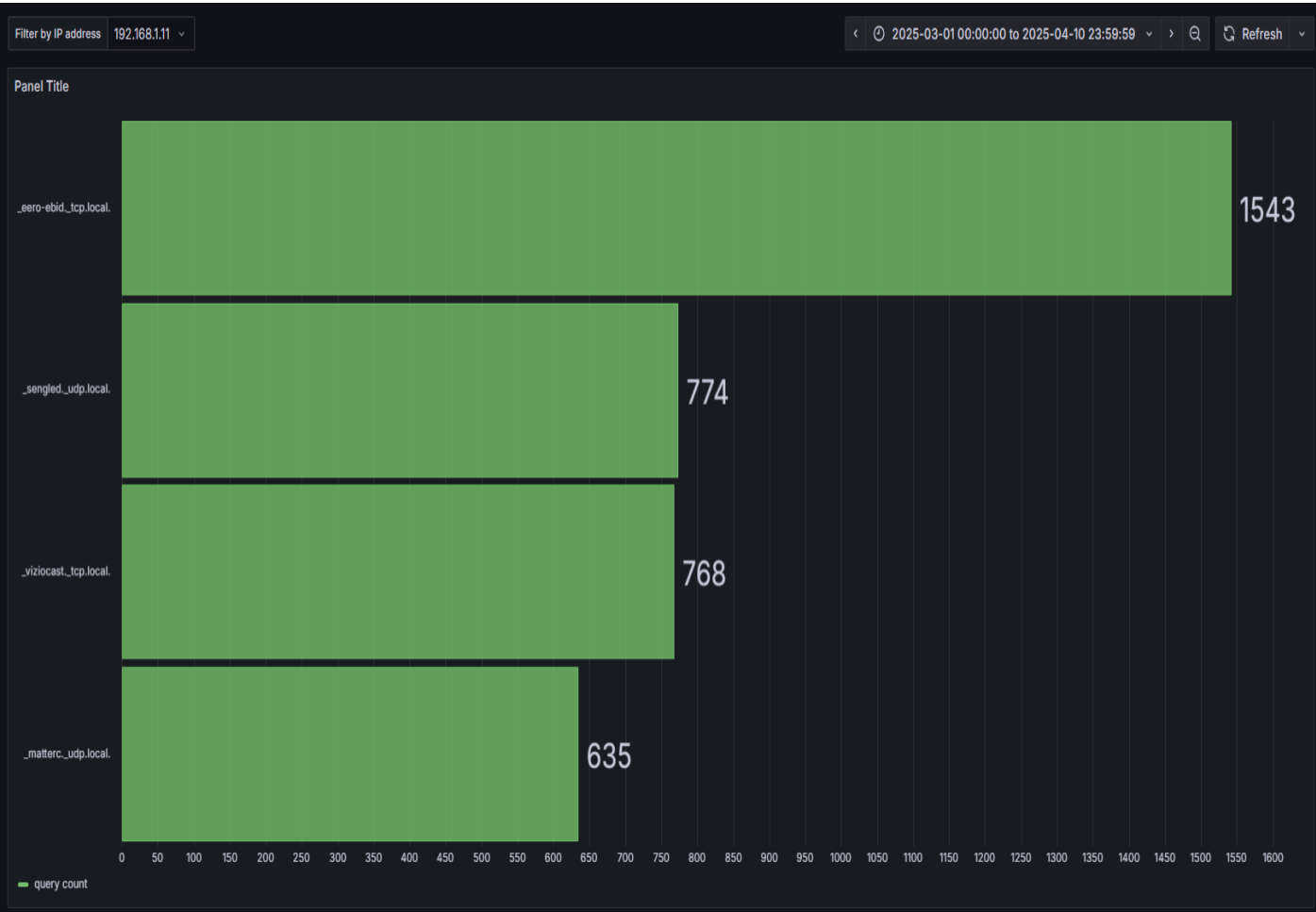
All the Devices online

Panel Title			
device_name	last_seen_time	previous_seen_time	ip_address
Amazon_Echo	2025-04-03 17:18:38	16 days ago	192.168.2.11
YI_1080pHomeCameraAIPlus	2025-04-03 17:18:30	16 days ago	192.168.2.31
Blink_Sync Module 2	2025-04-03 17:18:37	16 days ago	192.168.2.40
Amazon_EchoDot	2023-03-18 04:09:42	753 days ago	192.168.2.10
Samsung_WisenetSmartCam_A1	2025-04-03 17:18:38	16 days ago	192.168.2.4
iRobot_RoombaRobotVaccum	2025-04-03 17:14:43	16 days ago	192.168.2.32
D-Link_FullHDPan&TiltProHDWifiCamera	2025-04-03 17:18:37	16 days ago	192.168.2.26
Amazon_EchoDot3rdGeneration	2025-04-03 17:18:37	16 days ago	192.168.2.9
Meross_SmartWiFiGarageDoorOpener	2023-03-18 04:09:27	753 days ago	192.168.2.30
TP-Link_TapoHomeSecurityCamera	2025-04-03 17:18:38	16 days ago	192.168.2.5
Google nest Mini	2025-04-03 17:18:39	16 days ago	192.168.2.42
Google_NestThermostat	2025-04-03 17:18:36	16 days ago	192.168.2.50
PixStar_FotoConnect	2025-04-03 17:18:28	16 days ago	192.168.2.2
TP-Link_TapoMiniSmartWifiSocket1	2025-04-03 17:18:39	16 days ago	192.168.2.19
TP-Link_KasaSmartWiFiLightBulbMulticolor	2025-04-03 17:14:35	16 days ago	192.168.2.24
TP-Link_KasaSmartLightStrip	2025-04-03 17:18:36	16 days ago	192.168.2.47
TP-Link_KasaSmartWifiPlugMini1	2025-04-03 17:17:11	16 days ago	192.168.2.21
Ecobee_3liteSmartThermostat	2025-04-03 17:18:36	16 days ago	192.168.2.49
Ring_Doorbell4	2023-03-18 04:09:07	753 days ago	192.168.2.48
TP-Link_TapoMiniSmartWifiSocket2	2023-03-18 04:08:34	753 days ago	192.168.2.20

DNS Queries and Distinct DNS Queries sent by all devices



Query count to different domains by all devices



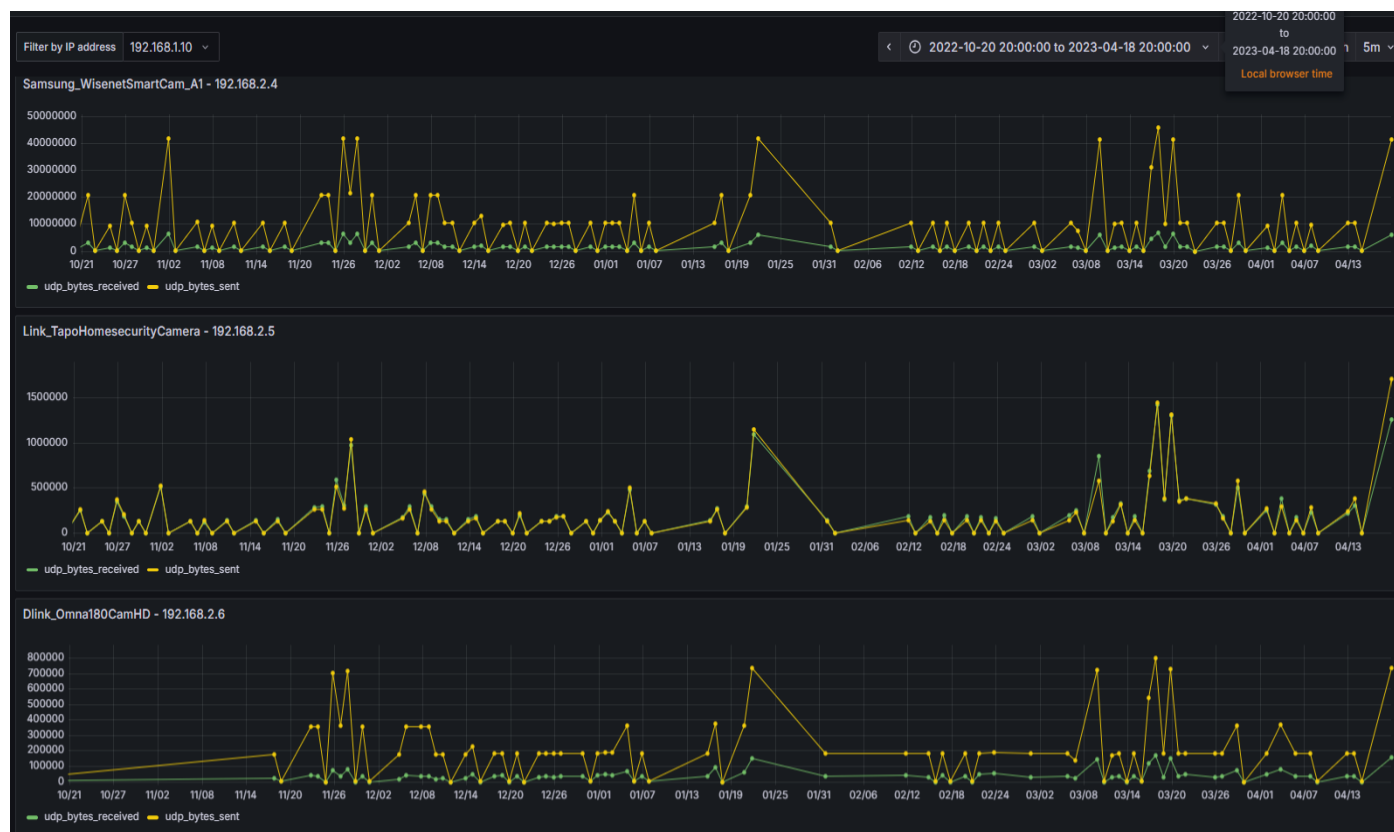
ICMP bytes sent and received by all devices



TCP bytes sent and received by all devices



UDP bytes sent and received by all devices

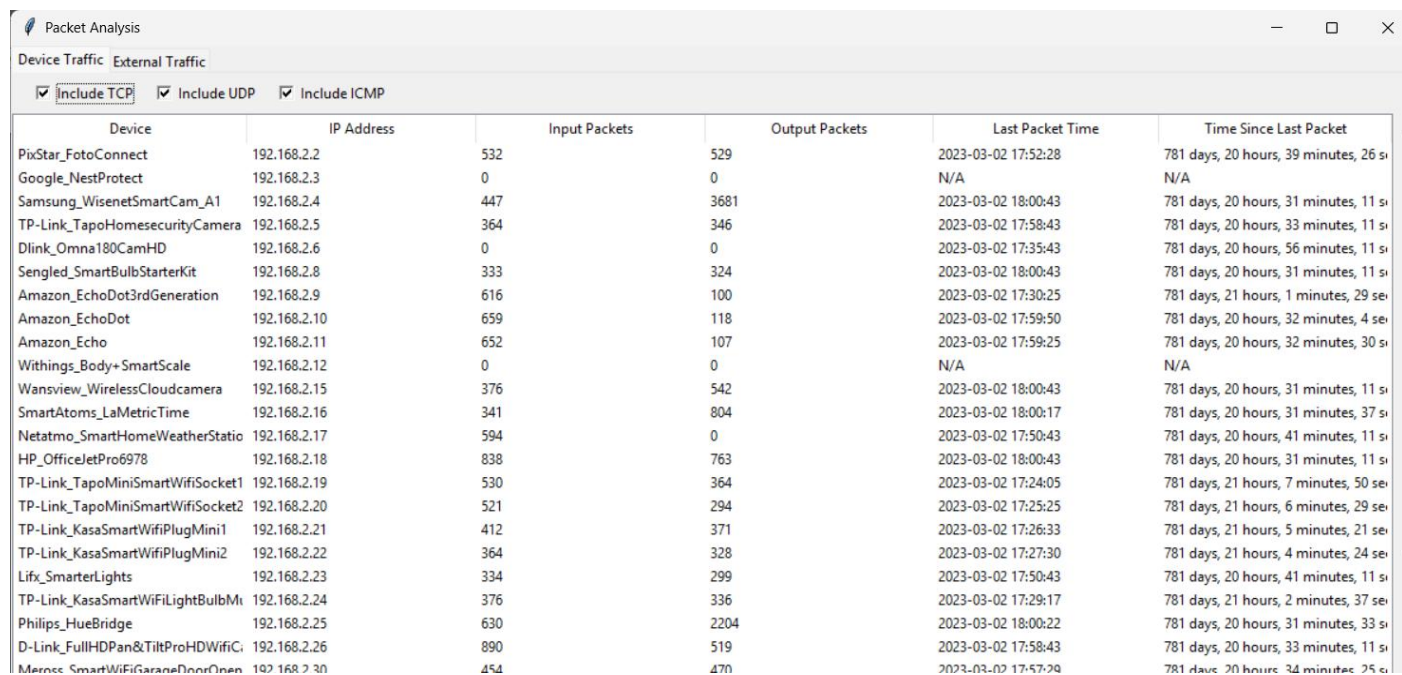


4.5 Graphical User Interface (GUI)

A user-friendly GUI is developed using tkinter to facilitate interaction with the system. Features include:

- Loading and analysing PCAP files
- Displaying identified devices and their attributes
- Visualising query timelines

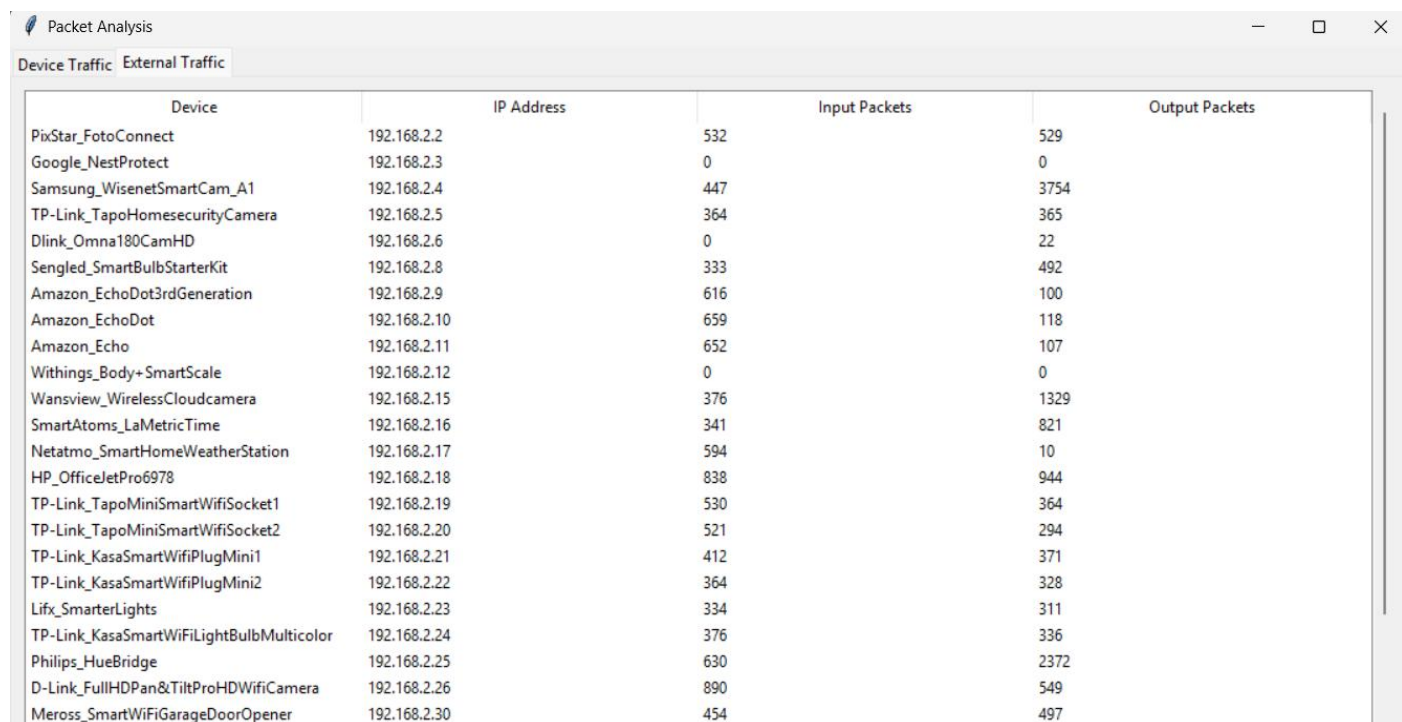
GUI showing TCP, UDP and ICMP packets and the last time the device was online:



The screenshot shows a window titled "Packet Analysis" with a tab labeled "Device Traffic". Below the tab are three checked checkboxes: "Include TCP", "Include UDP", and "Include ICMP". The main area contains a table with the following columns: Device, IP Address, Input Packets, Output Packets, Last Packet Time, and Time Since Last Packet. The table lists 26 devices with their respective IP addresses, packet counts, and last activity times.

Device	IP Address	Input Packets	Output Packets	Last Packet Time	Time Since Last Packet
PixStar_FotoConnect	192.168.2.2	532	529	2023-03-02 17:52:28	781 days, 20 hours, 39 minutes, 26 s
Google_NestProtect	192.168.2.3	0	0	N/A	N/A
Samsung_WisenetSmartCam_A1	192.168.2.4	447	3681	2023-03-02 18:00:43	781 days, 20 hours, 31 minutes, 11 s
TP-Link_TapoHomeSecurityCamera	192.168.2.5	364	346	2023-03-02 17:58:43	781 days, 20 hours, 33 minutes, 11 s
Dlink_Omna180CamHD	192.168.2.6	0	0	2023-03-02 17:35:43	781 days, 20 hours, 56 minutes, 11 s
Sengled_SmartBulbStarterKit	192.168.2.8	333	324	2023-03-02 18:00:43	781 days, 20 hours, 31 minutes, 11 s
Amazon_EchoDot3rdGeneration	192.168.2.9	616	100	2023-03-02 17:30:25	781 days, 21 hours, 1 minutes, 29 s
Amazon_EchoDot	192.168.2.10	659	118	2023-03-02 17:59:50	781 days, 20 hours, 32 minutes, 4 s
Amazon_Echo	192.168.2.11	652	107	2023-03-02 17:59:25	781 days, 20 hours, 32 minutes, 30 s
Withings_Body+ SmartScale	192.168.2.12	0	0	N/A	N/A
Wansview_WirelessCloudcamera	192.168.2.15	376	542	2023-03-02 18:00:43	781 days, 20 hours, 31 minutes, 11 s
SmartAtoms_LaMetricTime	192.168.2.16	341	804	2023-03-02 18:00:17	781 days, 20 hours, 31 minutes, 37 s
Netatmo_SmartHomeWeatherStation	192.168.2.17	594	0	2023-03-02 17:50:43	781 days, 20 hours, 41 minutes, 11 s
HP_OfficeJetPro6978	192.168.2.18	838	763	2023-03-02 18:00:43	781 days, 20 hours, 31 minutes, 11 s
TP-Link_TapoMiniSmartWiFiSocket1	192.168.2.19	530	364	2023-03-02 17:24:05	781 days, 21 hours, 7 minutes, 50 s
TP-Link_TapoMiniSmartWiFiSocket2	192.168.2.20	521	294	2023-03-02 17:25:25	781 days, 21 hours, 6 minutes, 29 s
TP-Link_KasaSmartWiFiPlugMini1	192.168.2.21	412	371	2023-03-02 17:26:33	781 days, 21 hours, 5 minutes, 21 s
TP-Link_KasaSmartWiFiPlugMini2	192.168.2.22	364	328	2023-03-02 17:27:30	781 days, 21 hours, 4 minutes, 24 s
Lifx_SmarterLights	192.168.2.23	334	299	2023-03-02 17:50:43	781 days, 20 hours, 41 minutes, 11 s
TP-Link_KasaSmartWiFiLightBulbMulticolor	192.168.2.24	376	336	2023-03-02 17:29:17	781 days, 21 hours, 2 minutes, 37 s
Philips_HueBridge	192.168.2.25	630	2204	2023-03-02 18:00:22	781 days, 20 hours, 31 minutes, 33 s
D-Link_FullHDPan&TiltProHDWiFiCamera	192.168.2.26	890	519	2023-03-02 17:58:43	781 days, 20 hours, 33 minutes, 11 s
Meross_SmartWiFiGarageDoorOpener	192.168.2.30	454	470	2023-03-02 17:57:20	781 days, 20 hours, 34 minutes, 25 s

GUI showing outgoing Traffic:



The screenshot shows the same "Packet Analysis" window, but with the "External Traffic" tab selected. The table displays outgoing traffic for the same 26 devices, showing only the "Output Packets" column.

Device	IP Address	Output Packets
PixStar_FotoConnect	192.168.2.2	529
Google_NestProtect	192.168.2.3	0
Samsung_WisenetSmartCam_A1	192.168.2.4	3754
TP-Link_TapoHomeSecurityCamera	192.168.2.5	365
Dlink_Omna180CamHD	192.168.2.6	22
Sengled_SmartBulbStarterKit	192.168.2.8	492
Amazon_EchoDot3rdGeneration	192.168.2.9	100
Amazon_EchoDot	192.168.2.10	118
Amazon_Echo	192.168.2.11	107
Withings_Body+ SmartScale	192.168.2.12	0
Wansview_WirelessCloudcamera	192.168.2.15	1329
SmartAtoms_LaMetricTime	192.168.2.16	821
Netatmo_SmartHomeWeatherStation	192.168.2.17	10
HP_OfficeJetPro6978	192.168.2.18	944
TP-Link_TapoMiniSmartWiFiSocket1	192.168.2.19	364
TP-Link_TapoMiniSmartWiFiSocket2	192.168.2.20	294
TP-Link_KasaSmartWiFiPlugMini1	192.168.2.21	371
TP-Link_KasaSmartWiFiPlugMini2	192.168.2.22	328
Lifx_SmarterLights	192.168.2.23	311
TP-Link_KasaSmartWiFiLightBulbMulticolor	192.168.2.24	336
Philips_HueBridge	192.168.2.25	2372
D-Link_FullHDPan&TiltProHDWiFiCamera	192.168.2.26	549
Meross_SmartWiFiGarageDoorOpener	192.168.2.30	497

4.6 Challenges and Solutions

During development, several challenges were encountered:

- **High Volume of Data:** Processing large PCAP files requires optimisation of parsing algorithms to maintain performance.
- **Device Identification:** Differentiating devices with similar service advertisements necessitated the incorporation of additional fingerprinting parameters.
- **Real-time Visualisation:** Ensuring that Grafana dashboards reflected up-to-date information involved configuring appropriate data refresh intervals and optimising SQL queries.

These challenges were addressed through code optimisation, enhancing the fingerprinting criteria, and fine-tuning the database and Grafana configurations.

5. Results and Discussion

This chapter presents a detailed overview of the data collected and visualised through the Grafana dashboards, focusing on protocol-specific traffic metrics and DNS behaviour across the network. The insights gained highlight how this passive monitoring system aids in real-time observability and anomaly detection.

5.1 Overview of Network Activity

The system captures traffic data on a per-device basis, including total bytes sent and received across TCP, UDP, and ICMP protocols. DNS activity is also logged, covering total and unique queries. These values are aggregated and visualised using Grafana, backed by a Mysql database.

5.2 Total TCP, UDP, ICMP Bytes and DNS Queries

Monitoring total bytes transmitted through each protocol reveals overall network usage. TCP remains the dominant protocol, particularly for web traffic and device communication. UDP traffic is consistent, mostly associated with DNS and local services like mdns. ICMP usage is minimal and mainly attributed to periodic pings or diagnostic tools.

5.3 Destination-Wise TCP and UDP Bytes

Analysing traffic by destination IP addresses surfaced key usage patterns:

- Certain IPs received a disproportionately high number of TCP bytes, indicating media consumption or cloud storage synchronization.
 - UDP traffic was primarily directed toward public DNS servers and internal multicast addresses.
-

5.4 DNS Query Volume and Distinct Domains

Per-device DNS queries show:

- Most devices, especially assistants, consistently queried the same set of domains periodically.

Distinct domain analysis proved especially useful:

- Devices with minimal unique domains (0–3) were found to be light bulbs and passive IoT devices like plugs.
-

5.5 Discussion

The deployment and visualisation framework effectively highlighted both normal and anomalous behaviour across a small-scale monitored network. It proved especially useful in:

- **Identifying DNS-related anomalies** through query count and diversity.
- **Detecting traffic-heavy devices**, which can be optimized or further investigated.
- **Providing live visibility into online device presence**, crucial for asset tracking.

The system's strength lies in its **passive, low-overhead monitoring approach** combined with **real-time, visually rich Grafana dashboards**. Future enhancements could include:

- Integration of alerting for high-risk patterns.
- Trend prediction models using historical data.
- User tagging for device identification and behaviour modelling.

6. Conclusion and Future Work

6.1 Conclusion

The project successfully demonstrates a lightweight, scalable, and effective network observability system that leverages Grafana for real-time dashboard visualisation of network metrics. By capturing and analysing protocol-specific traffic—namely TCP, UDP, ICMP, and DNS—on a per-device basis, the system provides deep insights into user behaviour, device roles, and potential security anomalies.

Through structured MySQL-backed storage and dynamic dashboards, the platform:

- Enabled **real-time tracking of online devices**.
- Highlighted **distinct traffic patterns** for individual devices.
- Revealed **anomalies in DNS activity**, such as spikes in distinct domain queries.
- Visualized **traffic distribution across destination IPs**, aiding in identifying abnormal communication behaviours.
- Supported **cross-protocol analysis**, giving a holistic view of the network.

The modular nature of the pipeline—from traffic collection to Grafana rendering—makes it adaptable to different network sizes and conditions. Even in environments without deep-packet inspection, this passive analysis delivers meaningful security and operational insights.

6.2 Future Work

Although the system offers powerful baseline capabilities, there is significant scope for enhancement. Future improvements can make it more intelligent, secure, and user-friendly:

6.2.1 Enhanced Alerting and Notifications

Integrating Grafana's alert manager or third-party systems (like Prometheus or custom Python alert logic) can notify administrators in real time when traffic anomalies or suspicious DNS patterns are detected (e.g., too many distinct queries in a short span).

6.2.2 Traffic Enrichment with WHOIS and Geoip

Incorporating WHOIS and Geoip lookups for destination IPS will allow dashboards to show **geographic and ownership context**, useful for identifying unauthorised foreign connections.

6.2.3 Machine Learning Integration

Using historical traffic logs, machine learning models can:

- Detect **behavioural anomalies**.
- Predict **device activity trends**.
- Perform **clustering of devices** based on their traffic patterns.

6.2.4 User/Device Tagging System

Associating MAC addresses or IPs with actual usernames, device types, or departments can make the dashboards even more human-readable and help with policy enforcement and audit trails.

6.2.5 Extended Protocol Monitoring

Expanding support to analyse application-layer protocols like HTTP, HTTPS (via SNI), or TLS handshakes can increase visibility without needing deep inspection.

6.2.6 Historical Aggregation and Archival

To better handle long-term performance, data archiving strategies such as **time-based aggregation** or **migration to data lakes** (e.g., Amazon S3 or Influx DB) could be introduced, preserving analytics while reducing storage load.

6.2.7 Security Hardening

As a network monitoring tool, the collector and dashboard components should be hardened:

- Encrypted communication between the collector, database, and Grafana.
- Authentication and role-based access control for viewing sensitive dashboards.
- Logging and audit trail for dashboard access and changes.

6.3 Final Thoughts

With rising complexities in modern networks—ranging from cloud-native apps to IoT—visualizing traffic and understanding device behaviour passively is not just a luxury, but a necessity. This project lays the groundwork for a **non-intrusive, cost-effective network intelligence solution** powered by open-source tools. It bridges the gap between low-level packet statistics and high-level operational insights, empowering both administrators and analysts to stay ahead of emerging issues.

As the system evolves, its potential use cases span beyond just monitoring it can aid in **compliance auditing, policy enforcement, digital forensics, and even predictive maintenance** of IT infrastructure.