POLITECNICO DI TORINO

Master Degree in Computer Engineering

Master Degree Thesis

Mobility detection through electromagnetic fingerprints in 5G networks

Supervisors
Prof. Claudio Ettore Casetti
Prof. Paolo Giaccone

Candidate
Riccardo Rusca
matricola: 262737

Academic year 2019-2020

Abstract

In recent years, our cities have become increasingly smart, thanks to the massive use of IoT (*Internet-of-Things*) sensors and smart devices owned by people, life in the city has improved considerably.

One of the most popular and important studies concerns the monitoring of people flows with the aim to guarantee both better services to citizens, from the point of view of dimensioning transportation networks, bike lanes, and public routes. However, it is also significant for identifying and quantifying people in sensitive areas (e.g., for safety and security purposes, such as during large crowd gatherings) or in areas of transit. It is essential to introduce new technologies for tracking people in case of health emergencies, such as the one we have been experiencing for many months.

In this thesis work, the analysis of people's mobility is studied in a restricted area between the Politecnico di Torino and Porta Susa train station. The study uses six commercial probe-detection sensors (APs scanners) installed in strategic positions to collect data from smart devices carried by passers-by.

The scanners can collect WiFi and Bluetooth probe requests messages sent periodically by all the active devices searching for known APs or devices to connect with. The source MAC address of such messages (even if randomized and anonymized) allow inferring the mobility of each mobile device. By means of two NFV (*Network functions virtualization*) on the Edge Cloud, all the data are retrieved, stored, and visualized with some graphs and charts, relative to the detected MAC addresses, in a real-time dashboard.

Thanks to the analysis of the data collected, it is possible to identify the most frequent patterns and types of mobility around the area of interest. Furthermore, it is relevant to see the effect that the lockdown, caused by COVID-19, has had on campus life but also in people's everyday travels. The platform's implementation took into account a future expansion on the number of sensors throughout the city; the platform adopts the MEC (*Multi-Access Edge Computing*) paradigm by which the computation is distributed across the mobile infrastructure, allowing to scale the approach and its accuracy.

Acknowledgements

I would like to thank my supervisors, Claudio Casetti and Paolo Giaccone, for their professionalism, availability and enthusiasm they put into each project.

Also, a special thanks go to my parents, my girlfriend Martina and all my friends, especially Davide, for their support in all these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, Internet-of-Things (IoT) systems have a substantial impact on people's everyday lives. With the birth of 5G networks, lots of uses cases have been developed to improve our cities' quality of life.

A smart city is an urban area that uses different types of IoT electronic sensors to collect data and efficiently provide, through the analysis of the data captured, resources and services to finally improve life in the city. The smart city concept integrates information and communication technologies (ICT) and various physical devices connected to the city's IoT network; this permits to calibrate and optimize the efficiency of the operations and services that the city offers to the citizens, and also it is easier to monitor what is happening in the city in real-time and how the city is evolving. Citizens interact with the smart city through the use of their smart devices such as smartphones, tablets and smartwatches, but also through cars and motorcycles connected with the entire ecosystem.

Torino is the fourth largest Italian city in terms of population with almost 900 thousand people that move on more than 3000 km of routes every day. People need to move fast from one point to another, so it is crucial to control people's flows to provide better services from the point of view of dimensioning transportation networks, bike lanes, and public routes. On the other hand, it is also important to identify and quantify people in sensitive areas (e.g., for safety and security purposes, such as during large crowd gatherings) or in areas of transit.

The purpose of this thesis work aims to find a solution to that problem by concentrate the study in a small area of interest between the area of Politecnico and Porta Susa train station. The main goal is to detect the main patterns of mobility in the considered area, thanks to six APs scanners installed in strategic positions that can collect data from smart devices carried by passers-by.

These sensors can scan WiFi and Bluetooth bands to capture probes messages transmitted passively by smartphones or other smart devices that are trying to identify known nearby WiFi Access Points (APs) or other devices to connect to.

During this study, various problematic factors will also be analyzed to achieve the goal, among which the most interesting of all is the use of randomization. From a couple of years, mobile device vendors have introduced the MAC address randomization technique to protect user privacy. Also, the sensors installed have some limitations:

- they can only detect the presence of people, not the path they are following;
- they can only detect people who carry smart devices;
- the information transmitted is non-customizable, and it is affected by some vendor implementations.

The second point of the bulleted list above is one of the limitations on which we cannot intervene because we cannot know if every person on foot, by bike or by car (or by motorcycle) detected by the scanners brings with them just one smart device, or if they carry more than one. Also, we cannot know which interfaces are enabled on the devices themselves, if only the WiFi one, only the Bluetooth one, both or neither.

Finally, thanks to the use of the 5G backbone and the MEC paradigm (*Multi-access Edge Computing*), the data detected by the scanners were stored for further analysis and displayed in real-time through interactive dashboards.

Alternative solutions exist, for example, counting people using advanced techniques (as video image processing) is technically feasible, but very expensive. For example, state of the art YOLO_V3 library [29] can identify and count the heads in a 30 FPS video in real-time, but it requires a high-level GPU and a highly equipped server, thus the overall video-based detection platform is too expensive for large-scale use.

Over the years, several other approaches have been used to address the problem of presence detection; for example, infrared sensors, cameras, pressure sensors, visible light sensors, RFID, UWB, audio-processing and PC activity. However, the techniques mentioned above do not provide satisfactory results in relation to the cost of implementation and performance [19].

Chapter 2

Technological issues

This chapter will present the state of the art of WiFi and Bluetooth probe requests, how MAC addresses randomization is implemented in our smart devices, and finally, some notes about 5G networks and the edge cloud architecture.

2.1 WiFi probe request

Mobile devices can discover networks in two different ways, either passive scanning or active scanning.

Devices who use passive scanning techniques are all the time listening to beacons sent by APs to discover known networks. These techniques are not very efficient, so nowadays, proactive scanning is preferred to discover APs by sending probe request management frames on 802.11b/g/n channels.

Probe requests include a unique device identifier (e.g., the MAC address) and some device's capabilities (e.g., supported 802.11 standards). They can be directed to one specific AP by indicating its SSID (*Service Set IDentifier*) or broadcasted to all APs within range. In Figure 2.1, there is an example of the structure of the probe request management frame.

After receiving a probe request, an AP can inform the client of its presence using a probe response. If there are no APs in proximity or the APs are in passive mode, the client will not receive any response.

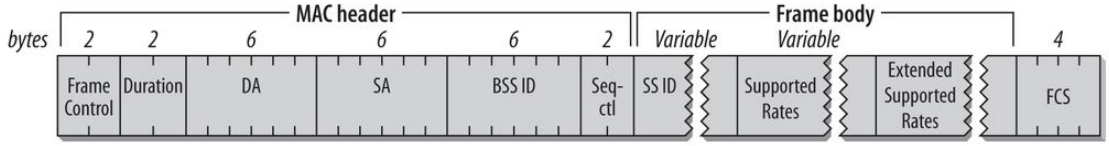


Figure 2.1: Structure of probe request frame (reproduced from [23])

Probes can help maintain WiFi connections, even when mobile devices are asleep. When the device is already connected to a network, the scanning process continues, searching for networks with higher signal strength, to always provide the best possible connection quality to the user. Probe requests are also the only solution to connect to hidden networks because the APs do not broadcast beacons.

The 802.11 standard does not force a broadcasting frequency for probe requests and beacons, so all the vendors can implement a different solution.

Factors that influence the increase in the frequency of sending probes [5]:

- Device WiFi interface enabled;
- Device screen turned on;
- Device unlocked.

Factors that NOT influence the frequency of sending probes:

- Device in flight mode;
- Mobile device in charge or not.

iOS devices send probe requests also when they are in sleep mode and no network connection. From iOS 9, the device's awakening or answering a call could increase the frequency of sending probes [14]. Also, in Android, the answering of a call could influence the frequency of probes requests.

From Android 9, each foreground app can scan four times in a 2-minute period. This allows a burst of scans in a short time; also, all background apps combined can scan one time in a 30-minute period [22].

Thanks to a recent study [5], we can approximate the number of probes n as follows: $n = k \cdot l \cdot 13$, where 13 is the number of 802.11b/g/n channels, k is the number of SSIDs stored in memory, and $l = 45$ is the estimated hourly rate of probing. The conclusion was that some mobile devices send almost 55 probe requests per hour on average, but even if a mobile device is not charging and in sleep mode, it might broadcast about 2000 probes per hour.

In another study [7], it was found experimentally that probe request frames can be sent with intervals ranging from 5 seconds to more than 10 minutes, with approximately 88% of the frames sent in the first 5 minutes.

2.2 Bluetooth probe request

Bluetooth devices operate in the 2.4 GHz short-range radio frequency spectrum, which is a globally unlicensed frequency.

Each device that has the Bluetooth interface ON can be in two different states:

- discoverable/visible;
- NOT discoverable/NOT visible.

In case the target device is not in visible mode, the only way to start the pairing process with it is to know its MAC address.

Bluetooth protocol uses a similar way, like WiFi probe requests, to discover other devices. This way is called inquiry mode, where a device sends out inquiry packets and other devices within range and in inquiry scan, can answer with inquiry reply. So they can exchange the necessary information to set up a connection.

The inquiry procedure can take up to 10.24 seconds, according to the specification [3], after which the inquiring device should know all the devices within range.

2.3 MAC address

A MAC address (*Media Access Control address*) is a unique identifier assigned to a network interface controller (NIC, an IEEE 802 capable device) for use as a network address in communications within a network segment. Every network interface device has a 48-bit MAC address layer-2 hardware identifier (like in Figure 2.2), designed to be persistent and globally unique. In order to guarantee the uniqueness of MAC addresses across devices, the Institute of Electrical and Electronics Engineers (IEEE) assigns blocks of addresses to organizations. Each particular three-byte prefix of the MAC address, the OUI [9] (also known as an *Organizationally Unique Identifier*), is under the control and responsibility of a vendor, manufacturer, or other organizations. The manufacturers are then free to assign to the remaining low-order three bytes (2^{24} distinct addresses) any value they wish when initializing devices, subject to the condition that they do not use the same MAC address twice. Also, devices with more than one network interface, such as routers and multilayer switches, must have a unique MAC address for each NIC.

IEEE also provides the possibility to purchase a “private” OUI, which does not include the company’s name in the register.

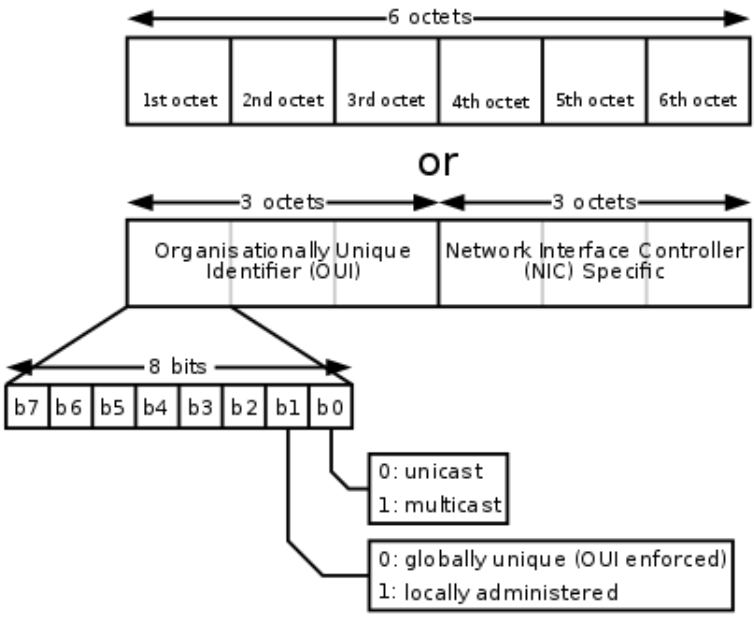


Figure 2.2: 48-bit MAC Address Structure (reproduced from [2])

Addresses can either be locally administered addresses or universally administered addresses. They are distinguished by setting the second-least-significant bit of the address's first octet (as highlighted in Figure 2.2). A locally administered address is assigned to a device by a network administrator, overriding the default one (burned-in address for physical devices). Instead, the least significant bit of an address's first octet identifies the type of transmission, unicast or multicast.

2.3.1 MAC address randomization

The randomization of MAC addresses is a strategy that is intended to prevent potential observers from identifying which mobile devices are within reach of a sensor.

As discussed in Section 2.1 and 2.2, probe request frames require a source MAC address. However, if a device uses its globally unique MAC address, then it is effectively broadcasting its identity all times to any AP receiver that is nearby. Smart devices of users can then be easily tracked across temporal and spatial boundaries as their devices transmit their unique identity. To ensure the users' privacy, both Android and Apple iOS operating systems uses locally assigned random MAC addresses to perform active scanning while the devices are in a disassociated state. Since the MAC address is now random, users gain a measure of anonymity up until they associate with an AP.

With the advent of randomized MAC addresses tracking a wireless device is no longer trivial. Only when a mobile device attempts to connect to an AP, it reverts to using its globally unique MAC address. For this reason, in the last years, lots of closed spaces such as malls, airports, and station offers to users free WiFi connection so that they can easily track flows of people through shops or other facilities and provide a better user experience. Recent studies have overcome this limit by implementing a sort of de-randomizer [18], an algorithm that is based on the analysis of common characteristics present in different probe requests, that remain constant even with randomized MAC addresses, and other parameters that allow calculating the probability that two random MAC addresses belong to the same device.

Android implementation - WiFi

From Android 6.0 the randomization is implemented in the software, but each vendor can decide whether to use it and how to use it [14].

Starting from Android 8.0, Android devices use randomized MAC addresses when probing for new networks while not currently associated with a network. In Android 9, there is a developer option (disabled by default) that causes the device to use a randomized MAC address when connecting to a WiFi network [12].

Also, the official Android documentation says that randomized MAC addresses are generated per SSID and are persistent.

However, in a recent study [7], it is stated that Android's MAC randomization is largely absent even if the version of the operating system does support this feature.

iOS implementation - WiFi

From iOS 8.0, the randomization is implemented and used by Apple [13]. According to [19], iPhone devices with iOS 10.1.1 use a new random MAC every time that:

- the device is locked or unlocked;
- a WiFi interface is activated or deactivated;
- a connection to a WiFi access point is made or attempted.

It can be stated that in Apple devices, it is not possible to estimate the time interval in which the same random MAC address is used, as this period depends on the way users interact with their devices.

2.3.2 MAC randomization adoption by the main vendors

All subsequent data have been drawn from the following previous studies [13, 14], and provide a general line on the use (or not) of the MAC address randomization techniques by the main smart devices vendors.

An important aspect to keep in mind is that randomization techniques and their implementation on smart devices are rapidly evolving. With the release of new software versions, various manufacturers often change some implementations to improve user privacy.

Samsung

Despite Samsung being one of the leaders in the mobile phone market, in the studies carried out they did not detect any use of randomization techniques by the vendor.

Huawei

Huawei from EMUI 8.0 uses MAC randomization techniques. Huawei mobile phones change the MAC address with a new randomized address every time they connect to a different AP [8].

Motorola

The studies found that a subset of Motorola devices performs randomization using neither a CID nor an OUI with the local bit set. These devices use one of Motorola's proprietary OUIs, using the global MAC address occasionally, and a new randomized MAC address when transmitting probe requests.

This behaviour is strange because it means that Motorola uses random global addresses, thus not respecting the rule that prohibits having the same MAC address on more than one device; indeed, it could happen that a Motorola device temporarily has the same MAC address as the real one of another device.

Apple

As for Motorola, it also emerged for Apple that MAC addresses are randomly generated by iOS devices without share any common prefix. All bits, including the 24-bit of the OUI, are completely random except the local bit, which is always set to 1, and the multicast bit, which is set to 0.

Windows

Microsoft supports randomization starting from Windows 10, as long as the hardware supports it. Furthermore, the studies show that Windows systems use random addresses both for sending probe requests and as MAC address when connecting to a network. This behaviour can be easily changed in the device settings.

Linux

Linux instead, added the support for the use of randomized MAC address during network scans from kernel version 3.18. Also, the MAC address is randomized at each scan iteration.

2.4 Evolution of mobile networks up to 5G

Since 1G was introduced in the early 1980s, every ten years a new wireless mobile telecommunications technology has been released; all of them refer to a new technology used by telco carriers and devices. Every technology has different speeds and features that improve on the generation before it.

In 1980s, the first generation of mobile networks was delivered; in particular, 1G delivered analog voice.

Ten years after, in the early 1990s, the second generation (2G) introduced digital voice (e.g., CDMA - *Code Division Multiple Access*) and encryption of voice and text (max speed 50 Kbps).

With the born of 3G, in the early 2000s, mobile networks allow transferring both "voice" data (digital phone calls) and "non-voice" data, such as downloading from the Internet, sending and receiving emails and instant messaging. The max speed of 3G is estimated to be around 2 Mbps for non-moving devices and 384 Kbps for moving vehicles. The theoretical max speed for HSPA+ is 21.6 Mbps [1].

The fourth-generation widely used today is the 4G LTE that ushered in the era of mobile broadband. 4G introduced new supports such as gaming services, HD mobile TV, video conferencing, and other things that demand higher speeds. The max speed of a 4G LTE network when the device is moving is 100 Mbps or up to 1 Gbps for low mobility communication [1].

Mobile services are so engaging that the market demands higher bit rates, lower latency, greater reliability and robust coverage.

The explosive growth in data traffic for mobile networks fuelled by applications such as ultra-high-definition video, augmented reality, and tactile Internet drives the industry toward the next generation (5G) of mobile wireless networks.

In 2019 began the deployment of the 5G networks; like its predecessors, the service area is divided into small geographical areas called cells. All cells have a local antenna that emits radio waves, which allow all wireless devices to connect to the Internet and telephone network.

The need for higher bit rates and capacity requires the use of new radio frequencies, the Extremely High Frequency (EHF), from 30 to 300 gigahertz (GHz). Radio waves in this band have wavelengths from ten to one millimetre, so radiation in this band is called millimetre waves (mmWave, Figure 2.3), which will be deployed in small cells with a limited coverage area. To ensure wide service, 5G networks operate on up to three frequency bands, low, medium, and high. Each frequency bands require different types of cells and different types of antennas, each giving a different trade-off of download speed vs. distance and service area. Mobile devices with 5G technology and wireless devices will connect to the network through the highest speed antenna within range at their location.

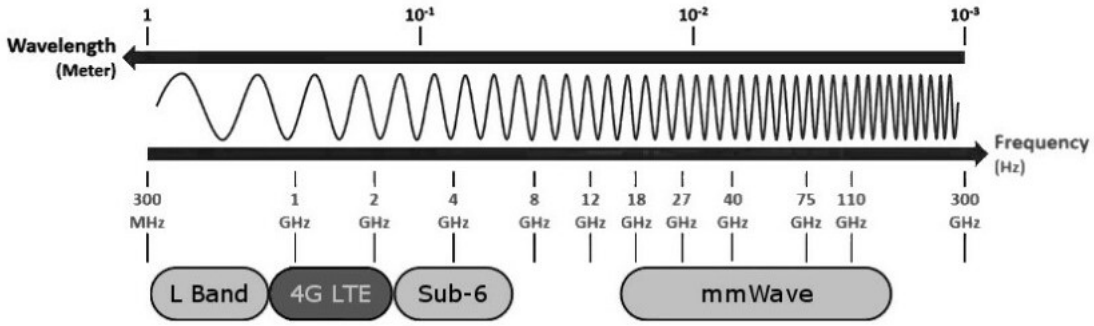


Figure 2.3: Networking Spectrum Bands (reproduced from [17])

2.5 5G networks

5G networks are designed to target traditional key performance indicators (KPIs), such as capacity and coverage, and new KPIs like connection density, latency, reliability, and power efficiency (as we can see in Figure 2.4).

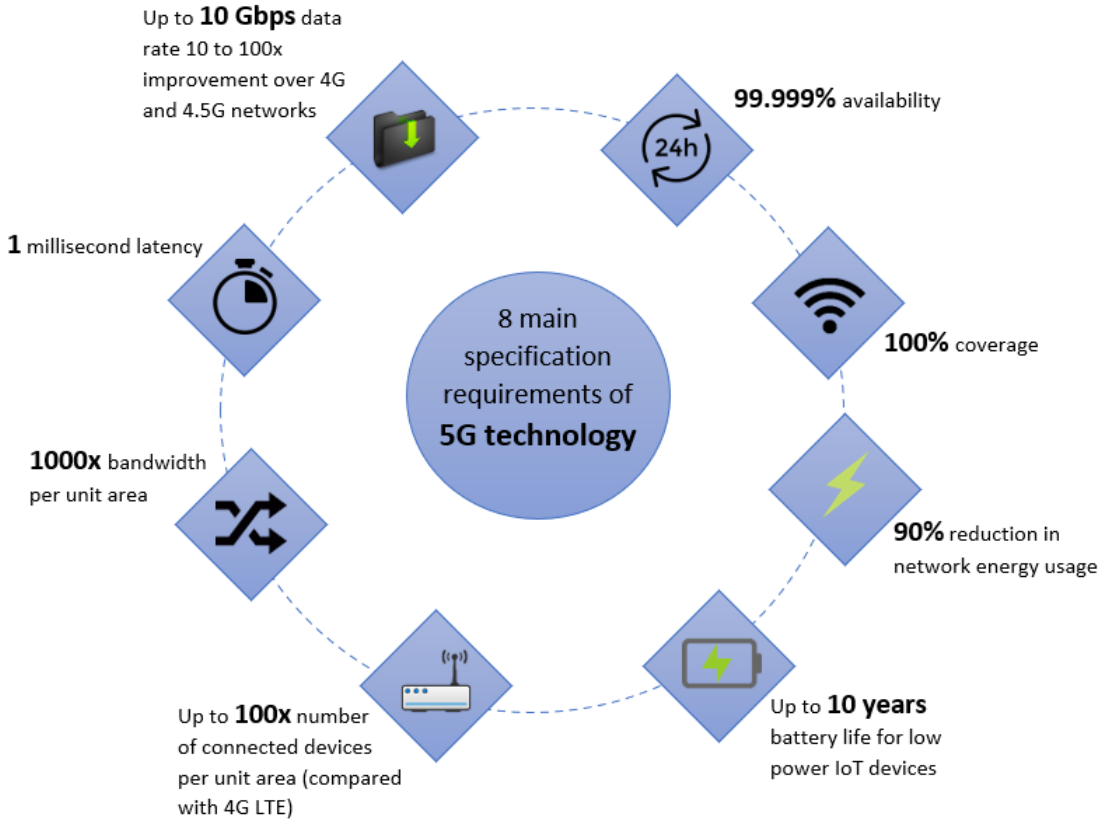


Figure 2.4: 5G specification requirements

Smart cities could use 5G in several ways to transform the lives of the people who live there. Through greater network efficiency, people would enjoy better connectivity between people and things, higher data rates, and lower latency than ever before, encouraging in areas such as automotive safety, infrastructure, virtual reality, and entertainment a substantial improvement.

There are three main application areas for the enhanced capabilities of 5G.

1. Enhanced mobile broadband (eMBB).
   Uses 5G as a progression from 4G LTE mobile broadband services, with faster connections, higher throughput and more capacity;

2. Massive machine-type communication (mMTC).
   Includes all kinds of IoT devices connected to the Internet. This requires support for a huge number of attached devices, deep coverage and long device battery life;

3. Ultra-Reliable Low Latency Communications (URLLC).
   Includes machine–machine communication that requires uninterrupted and robust data exchange, ultra-low latency and extreme reliability.

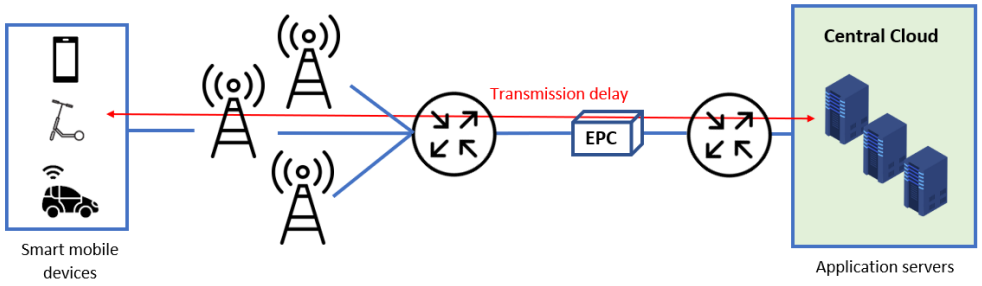
To achieve the necessary flexibility and requirements, 5G networks take advantage of two new technologies: *Network Functions Virtualization* (NFV) and *Multi-access Edge Computing* (MEC).

2.5.1 MEC - Multi-access Edge Computing

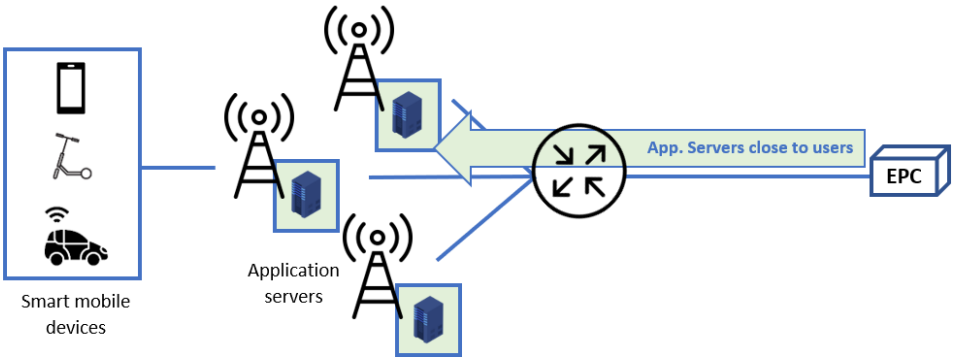
The Multi-access Edge Computing (MEC) paradigm was born to address the challenge of reducing the latency up to 1 millisecond in the core network. MEC's main concept is to reduce the physical distance between users, with their smart mobile devices, and the application servers. Thanks to the MEC paradigm, the processing resources are placed at the edge of the network, closer to the users [15, 26]. In Figure 2.5, the current 4G architecture and the new 5G architecture are compared.

The 4G architecture represented is the Centralized Cloud Computing (CCC) one. In the CCC architecture, the application servers are located at the central cloud, and all the tasks are processed inside the network functions that are virtualized on the servers. If the physical distance between the central cloud and the users is long, this architecture causes a delay in the core network.

Instead, the 5G architecture using the MEC paradigm can reduce transmission delay by placing all the application servers with the computing resources (edge cloud) near the users.



(a) *Centralized Cloud Computing (4G)*



(b) *Multi-access Edge Computing (5G)*

Figure 2.5: CCC vs. MEC architectures

Multi-Access Edge Computing allows analyzing, processing, and storing the data in the network edge instead of sending all data to a cloud for processing. Collecting and processing data closer to the users guarantee ultra-low latency and bring real-time performance to high-bandwidth applications and support latency-critical services. This helps to improve both the quality experienced by the end-user and the interactivity of the services.

Another advantage of this solution is given by the fact that by keeping the traffic related to the application locally, the MEC permits to reduce the bandwidth requirements in the transport network and increase the security of communications.

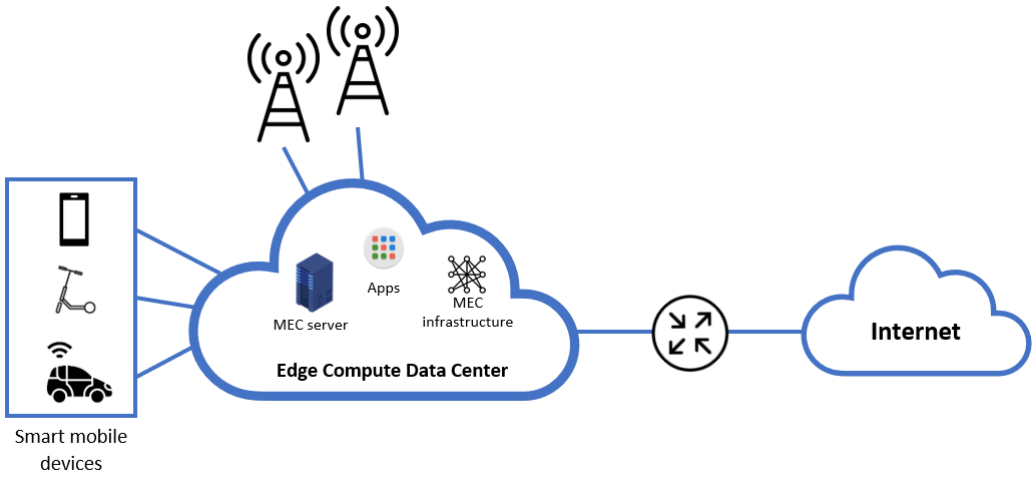


Figure 2.6: Multi-access Edge Computing (MEC)

MEC provides a new ecosystem and value chain. Telco operators can open their RAN (*Radio Access Network*) edge to authorized third-parties, allowing them to deploy innovative applications and services towards mobile users and enterprises.

2.5.2 NFV - Network Functions Virtualization

Network Functions Virtualization (NFV) is a way to virtualize all the network services such as routers, firewalls, and load balancers, that conventionally run on proprietary hardware. These services are grouped as virtual machines (VMs) on consumer hardware, usually, a Commercial Off The Shelf (COTS) server based on a standard architecture; this allows service providers to run their NFV on standard servers rather than proprietary ones.

NFV does not require the availability of dedicated hardware for every network function, but in a single server can run a lot of VMs. NFV also increases horizontal scalability by allowing providers to deploy new network services and applications on-demand without adding additional hardware resources.

Thanks to the MEC paradigm, network functions will be more distributed and close to the network edge in order to address the ultra-low latency requirements of the new applications [27] (like in 5G EVE pilot architecture, explained in Section 3.4).

Chapter 3

Proposal architecture

This chapter will present the implemented architecture and the IoT scenario, providing a complete detail on all the probe request parameters captured by the scanners. Finally, the core of the master's thesis work will be highlighted and described, i.e., the implementation in different network functions of a mobility tracking application.

3.1 5G EVE architecture

The 5G EVE architecture [4] is an instance of the 5G architecture, it implements the MEC (*Multi-access Edge Computing*) paradigm, and each application is realized through the use of NFV (*Network Functions Virtualization*).

In Figure 3.1, we can see in detail the implemented architecture. Following the flow from left to right, we can find firstly the APs scanners, which can scan and detect WiFi and Bluetooth probe request messages coming from smart devices carried by passers-by. All the scanners are connected through the RAN (*Radio Access Network*) to the OneM2M server [20], managed by TIM.

All the data stored in the OneM2M platform are retrieved by the MOB NFV and saved in a local database for subsequent uses. Finally, the experimenter can connect to the VIS NFV to see in real-time the data collected by the scanner through a web-based visualization tool.

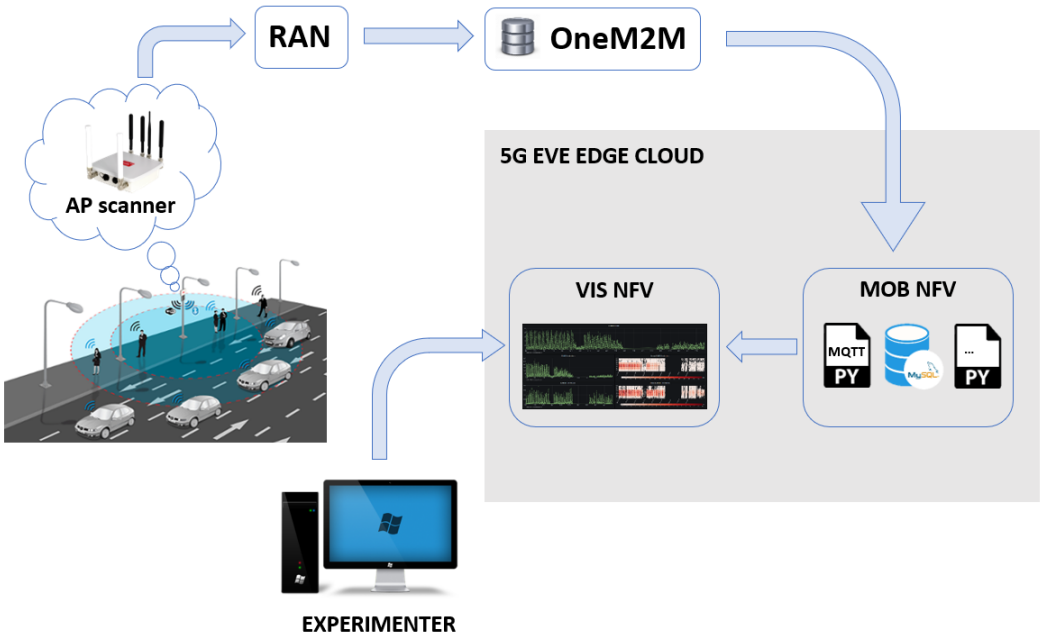


Figure 3.1: 5G EVE architecture

The platform's edge cloud (grey part in Figure 3.1) is located within a group of servers inside the Politecnico di Torino; therefore, they are easily accessible and manageable in case of breakdowns or other technical problems.

Thanks to the use of the MEC paradigm, the proposed architecture is able to scale horizontally in the number of scanners installed. While the use of NFV to realize the entire mobility tracking application provides great freedom from the point of view of the required hardware resources and their scalability.

3.2 IoT Scenario

The area considered in this thesis is the area from the campus of Politecnico di Torino to Porta Susa train station. It is characterized by a large inflow and outflow of people that every day moves into and out of the campus. In this area, a bicycle lane is also present, so it is possible to capture lots of signals from devices carried by people moving on foot, by bike (or micro-electric mobility), or by car.
Thanks to the collaboration of TIM, an Italian mobile operator, six WiFi and Bluetooth probe-detection scanners, Meshlium by Libelium [11], were installed (blue placeholder with name in coverage area map in Figure 3.2). Two of them were installed inside the campus while the other four, with the support of IREN and the City of Turin, on the top of as many traffic lights (like the one in Figure 3.3b).

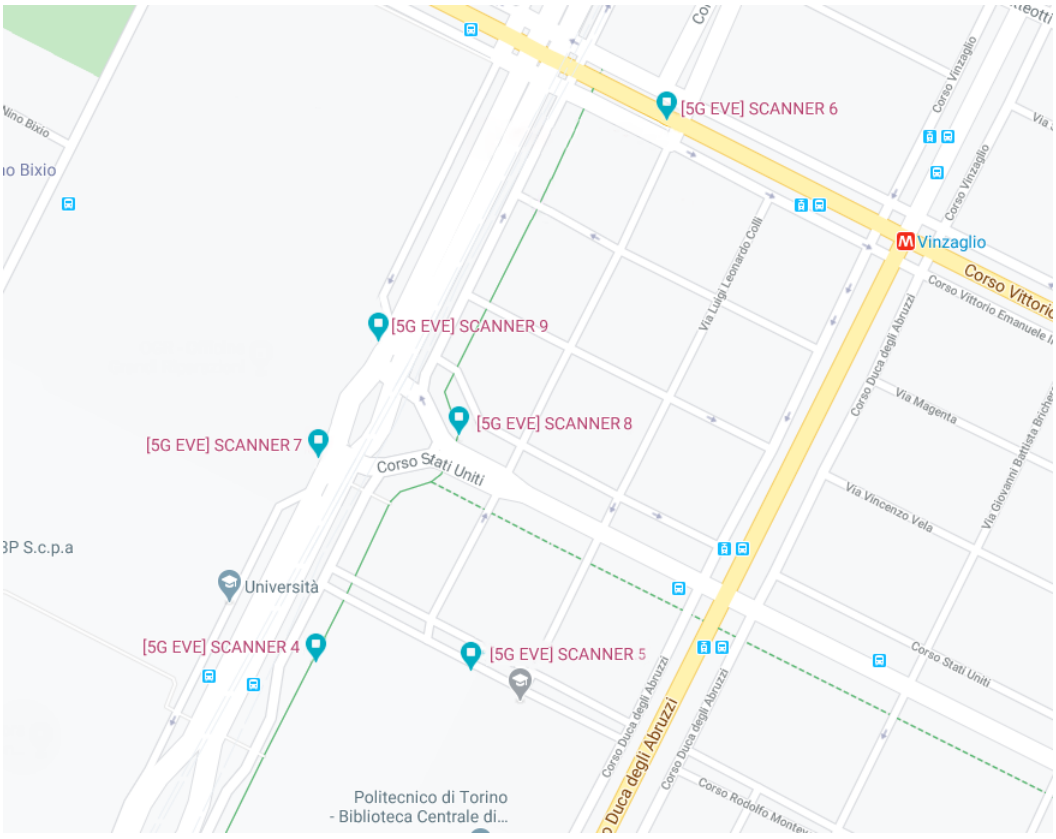


Figure 3.2: Coverage area map with scanners placeholder

(a) *Meshlium scanner*

(b) *Meshlium scanner installed on a traffic light pole*

Figure 3.3: WiFi and Bluetooth scanner

3.3 APs scanner

Each scanner can receive WiFi and Bluetooth probes by scanning WiFi bands at 2.4 and 5.0 GHz and Bluetooth ones. From such probes, the scanner extracts and store some information as:

1. The RSSI at the receiver, it has not been considered since the Libelium documentation does not explain how the value of RSSI is evaluated (e.g., it is unclear if the value detected is the average or the maximum) and how the RSSI of multiple probe requests received during the same sampling period is computed. Furthermore, as well known in the literature, the RSSI cannot be considered a reliable metric for mobility tracking;

2. The interface vendor, it does not permit to identify the interface uniquely and in many cases is equal to "Unknown";

3. The timestamp, given with the precision of one second. The same sampling time of 51-second is reported for all probe requests observed during the same sampling period. So it is not possible to have a detailed timing sequence of the probe requests, making the tracking extremely challenging. Furthermore, multiple probe requests coming from the same device during the same sampling period are collapsed into a single sample;

4. MAC address, this has been obtained by digesting the device MAC address through an SHA-224 function. Since the anonymized MAC address is not considered to be personal data, the adopted process is compatible with the EU General Data Protection Regulation (GDPR). In particular, the default hash function available in the Libelium scanner digests the MAC address together with the current time, not allowing the identification of the same MAC address at different times. To avoid this problem, the hashing mechanism was modified to digest just the MAC address;

5. Some other fields for Bluetooth probes, not considered for privacy reasons.

Figure 3.4 and 3.5 show some examples of the JSON file produced in output by the scanners.


```
{
  "count": 5,
  "from": "2020-08-10 00:55:00",
  "to": "2020-08-10 00:57:00",
  "data[0].RSSI": "-85",
  "data[0].Vendor": "Ubiquiti Networks, Inc.",
  "data[0].TimeStamp": "2020-08-10 00:55:14",
  "data[0].MAC": "464F368FA0A7A0F7992FBE19B9F596AE138CD5DF0B0A1CAE57B6F958",
  "data[1].RSSI": "-85",
  "data[1].Vendor": "ZyXEL Communications Corporation",
  "data[1].TimeStamp": "2020-08-10 00:55:14",
  "data[1].MAC": "8C0E3AC26FD518FC1CCEC6D0E1CD9ACE0CDAAC66AE758C1D7BF2C846",
  "data[2].RSSI": "-81",
  "data[2].Vendor": "XEROX CORPORATION",
  "data[2].TimeStamp": "2020-08-10 00:56:06",
  "data[2].MAC": "5F21A510F47C7D5EDD5D71FB720F5C7BD8B335DA58D0963CC6353582",
  "data[3].RSSI": "-67",
  "data[3].Vendor": "Intel Corporate",
  "data[3].TimeStamp": "2020-08-10 00:56:06",
  "data[3].MAC": "001784194A13225370A14CF8AF779FCBA41300670DDE22461D5F174A",
  "data[4].RSSI": "-1",
  "data[4].Vendor": "Unknown",
  "data[4].TimeStamp": "2020-08-10 00:56:06",
  "data[4].MAC": "BD67F19682B1535A3D3F9AFA1A85BAE1BBD9132690D834CD5C1C40C8",
  "chunk": 1,
  "ct": "20200809T230009",
  "rn": "4-20200809230009200KNoH",
  "lbl": "libeliumscanner,wifi",
  "ts": 1597014009000
},
```


Figure 3.4: WiFi JSON example

```
{
  "count": 1,
  "from": "2020-08-10 00:51:00",
  "to": "2020-08-10 00:53:00",
  "data[0].RSSI": "-76",
  "data[0].CoD": "SmartPhone",
  "data[0].ID": "",
  "data[0].Vendor": "Unknown",
  "data[0].TimeStamp": "2020-08-10 00:51:41",
  "data[0].MAC": "6BCD30B1AAB8D208C53078CCFC27475E73C54AB660BAB0C64127C537",
  "chunk": 1,
  "ct": "20200809T225609",
  "rn": "4-202008092256092280pqb",
  "lbl": "libeliumscanner,bluetooth",
  "ts": 1597013769000
},
```


(a) *Smartphone detection example*


```
{
  "count": 1,
  "from": "2020-08-10 08:55:00",
  "to": "2020-08-10 08:57:00",
  "data[0].RSSI": "-82",
  "data[0].CoD": "A/V Handsfree",
  "data[0].ID": "skoda bt 2583",
  "data[0].Vendor": "ALPS ERECTRIC CO.,LTD",
  "data[0].TimeStamp": "2020-08-10 08:55:12",
  "data[0].MAC": "199F5A58B9649292FA92C4B8A04D1AA098DBC1FF787FAEEDBE2D3A83",
  "chunk": 1,
  "ct": "20200810T070010",
  "rn": "4-20200810070010066dGLG",
  "lbl": "libeliumscanner,bluetooth",
  "ts": 1597042810000
},
```


(b) *Car detection example*

Figure 3.5: Bluetooth JSON example

Each scanner is connected to the 5G EVE platform [4], and every two minutes, the scanners upload the log of the WiFi and Bluetooth probe requests transmitted from nearby devices to the OneM2M platform [20], using their cellular link.

The OneM2M platform is available through the 5G EVE infrastructure and managed by TIM.

3.4 Pilot architecture

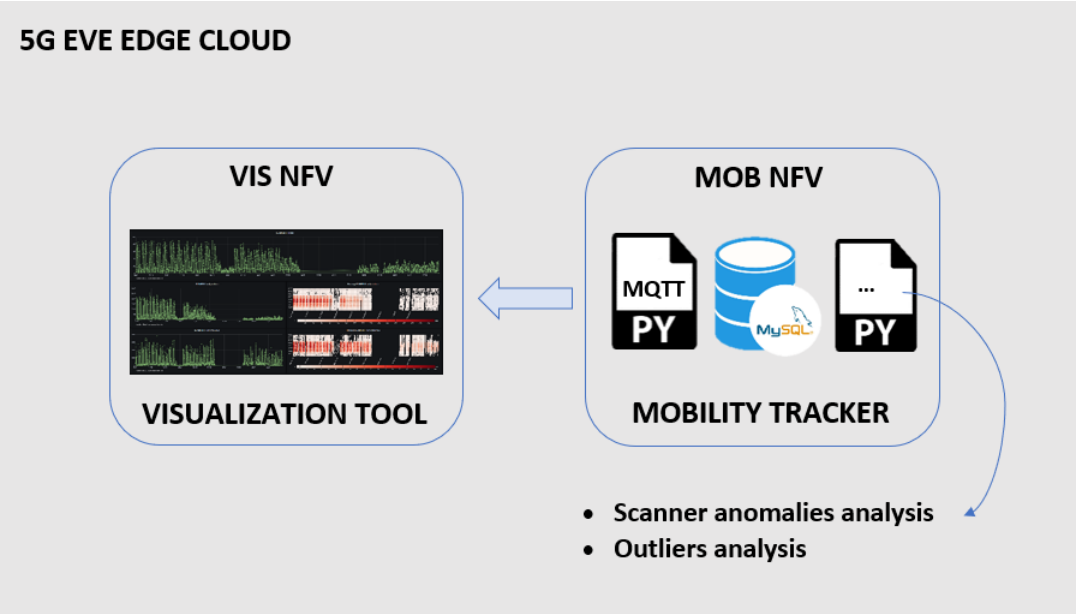


Figure 3.6: Pilot architecture

The adopted pilot architecture is shown in Figure 3.6, where the two main components of the use case are the MOB and the VIS NFV running in the 5G EVE edge cloud.

The MOB NFV is a Mobility Flow Tracker, whose aim is to correlate the time each MAC address has been observed by the different scanners and compute the corresponding path. The MOB NFV interacts with the OneM2M platform and retrieves the data from the scanners in real-time, adopting MQTT as publish-subscribe protocol (see Section 4.1). Also, in the MOB NFV runs some other script whose purposes are those of searching for anomalies in scanners operation and outlier MAC addresses in the database.

The VIS NFV is instead responsible for visualizing the data, stored in the MOB NFV database, through an interactive dashboard (see Chapter 6 for more details) that is available to the experimenter.

The VSB (*Vertical Service Blueprint*) graphic visualization for the use case is available in Figure 3.7, and it describes in a structured format the VIS and MOB components and their interactions. The VSB describes only the components under the control of the 5G EVE infrastructure, indeed excluding the experimenter client and the OneM2M infrastructure. The elements in Figure 3.7 represent respectively atomic components (cubes in dark blue), connectivity services (circles in pale blue with arrows), endpoints (edges), and Service Access Points (SAP) (dark grey circles with cloud). The VIS and MOB are encoded as atomic components in the Blueprint. They communicate with each other via two connectivity services, one for data plane traffic and the other for management traffic. The latter is used by the 5G EVE infrastructure to configure the atomic components at deployment time or during their life-cycle. The MOB has access to a SAP connected to the OneM2M platform to receive data from sensors. The VIS component, instead, is connected to a SAP to provide access to the visualization dashboard.

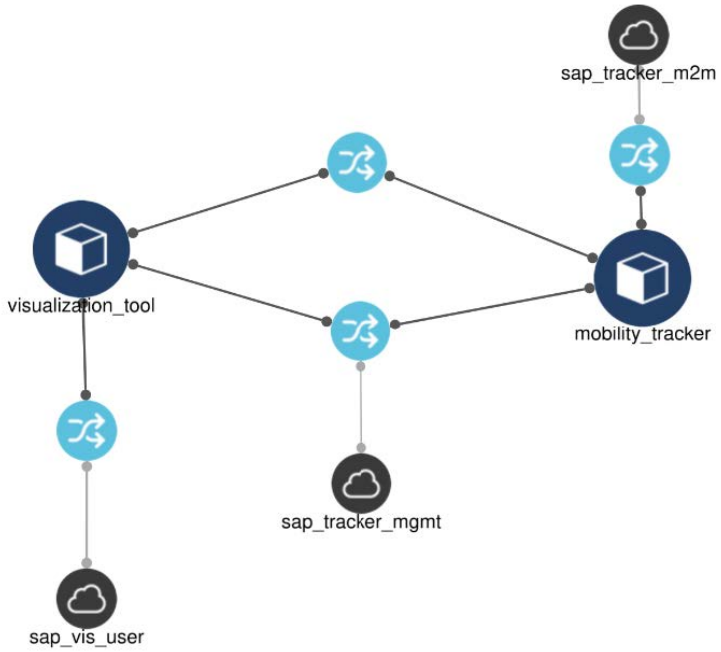


Figure 3.7: VSB graphical visualization

Chapter 4

Flow mobility tracking

This chapter will present how the data are retrieved from the sensors, which filters are applied, how they are stored, and how they are used to create charts, graphs, and real-time dashboards. Also, all the implementation choices made will be explained.

4.1 MQTT client

All the data collected from APs scanners are uploaded in real-time on the OneM2M platform through a mobile connection. The OneM2M platform contains six main containers, one for each scanner, and each of them is divided into two sub-containers, one for WiFi data and one for Bluetooth data.

To retrieve the data from the remote platform, an MQTT client has been implemented. MQTT (*Message Queuing Telemetry Transport*) is an open transport protocol, very lightweight, based on the publish-subscribe paradigm. The protocol runs over TCP/IP, so it is reliable and it prevents the out-of-order delivery of data and the packets loss. MQTT protocol was designed for constrained environments such as M2M (*Machine to Machine*) and IoT (*Internet-of-Things*), where a small code footprint is required, or the network bandwidth is limited.

Returning to our use case, in particular, for the connection with the OneM2M platform has been used an MQTTS (*MQTT over SSL*) protocol with the type: exactly once (2) for the value of the Quality of Service (QoS). This type of QoS guarantees that each message is received only once by the intended recipients. QoS 2 is the safest and slowest quality of service level [16]. If a packet gets lost along the way, the sender is responsible for retransmitting the message within a reasonable amount of time.

Thanks to the persistent session of the client and the QoS 2, all the messages sent are queued for offline clients until the client is available again, so in case of malfunctions of the network or the VM, we do not lose any messages containing the captured data.

As we can see in the figure below (Figure 4.1), the MQTT client inside the VM makes a Subscription for each container for which it wants to be notified when new data are uploaded. The mechanism involves creating a Subscription resource within that container, indicating MQTT broker where you will be notified with a message whenever a resource (containing data) is created in the inbox.

When a new message arrives at the broker from the APs scanner, it will be saved in the inbox, and then it will also be sent to the MQTT client where the message will be parsed, analyzed and then stored into the database.

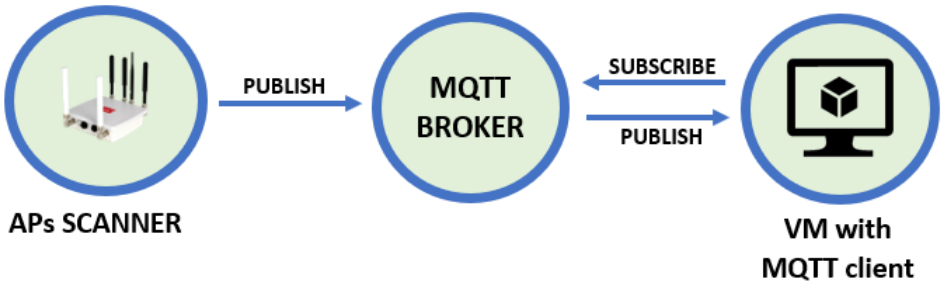


Figure 4.1: MQTT flow

The MQTT client was written in Python using one of the few available MQTT library, the paho.mqtt.client project of the Eclipse Foundation [21]. The MQTT client was developed into an Ubuntu Distribution [28], where it loops forever, waiting for incoming messages from the APs scanners.

The implemented MQTT client also has a mechanism of logging using directly syslog of the kernel and a mechanism of reconnection in case of an unexpected disconnection occurs.

4.2 Data filtering

As explained in the previous section (Section 4.1), the MQTT client receives from the OneM2M platform the data collected by the APs scanners in the form of JSON-encoded messages.

The first operation after the decode of the message is the update of the timestamp of the last message sent by the scanner, so we can be able to check anytime for any anomalies on the scanners (see Section 4.5.1 for further details). After that, before saving the data on the database, two more operations are done:

1. the extraction of the first 16 characters of each MAC addresses;
2. the outliers analysis of the MAC addresses and its classification.

Extraction of the first 16 characters of each MAC addresses

For the implementation choices, explained in the next section (Section 4.3), regarding the selection of the database, it was decided to save only the first 16 characters of each MAC address. This choice, supported by the graphs in Figure 4.2, was made primarily to save space in the database to be scalable in the future, and then because the first 16 characters of the MAC address are enough to identify a device uniquely, even if the data stored grows.

Consider m elements chosen uniformly at random, with repetition, from a set of cardinality n, with $m < n$. The probability $p(n)$ that at least one pair of equal elements has been chosen (i.e., a "collision" has occurred) is:

$$p(n) \approx 1 - e^{-\frac{m^2}{2n}} \tag{4.1}$$

As we can see in Figure 4.2, with almost 15 million MACs, the probability of collided MACs is 0.0006% considering only the first 16 hexadecimal digits of the address. With 30 million MACs, the probability drop to 0.0024%.

Considering the values currently recorded, almost a year after the entire system's deployment, we can consider right the choice made on the number of hexadecimal digits to keep to identify a MAC address.

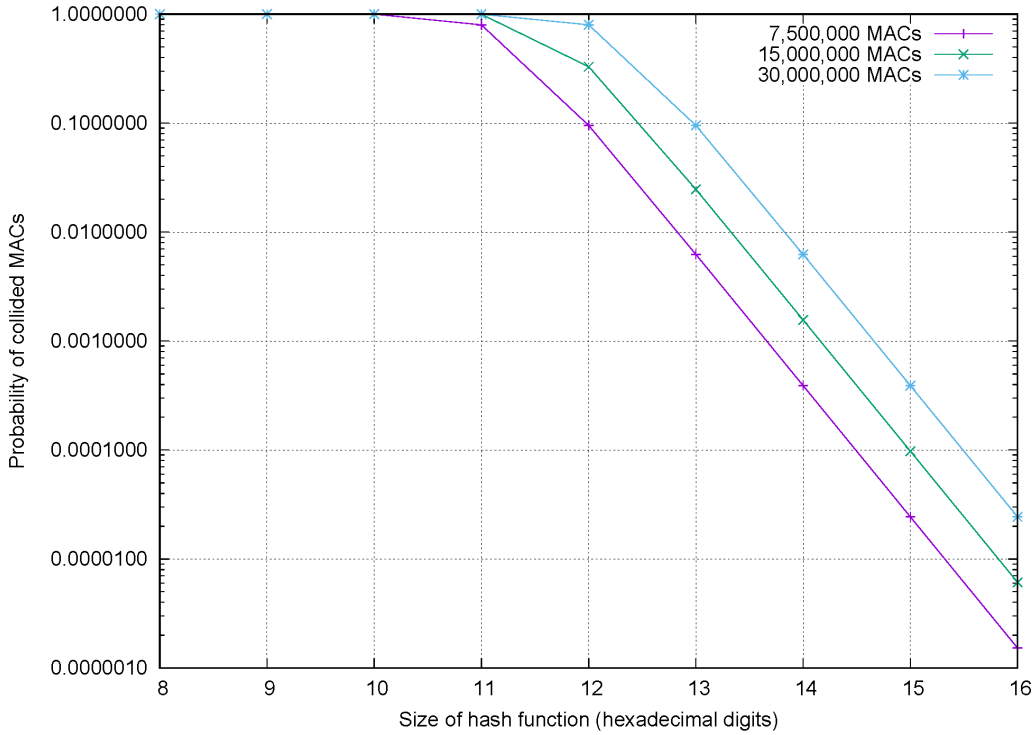


Figure 4.2: Minimum fingerprint

Classification of MAC addresses

In order to understand the effect of MAC randomization, thanks to the collaboration of the IT Area of Politecnico di Torino, a list of 34,927 MAC addresses of devices used by community members to connect to the campus WiFi network was collected. They comprise students, professors and administrative employees. For privacy reasons, these MAC addresses were anonymized through the same SHA-224 hash function used by the APs scanners. These MAC addresses are not randomized since they are collected after the device has been associated with one of the Politecnico APs.

So before saving each MAC address extrapolated from the messages, it is classified following these rules:

- professors and administrative employees ⇒ 2
- students ⇒ 1
- non-Politecnico community members ⇒ 0

This classification is very useful both for the subsequent visualization of the data in the dashboards and for the analysis of patterns with non-randomized MAC addresses; so every time an APs scanner detects a MAC address that is part of the Politecnico's list, we are sure that the address is not randomized.

Outliers analysis of the MAC addresses

In order to clean the data from the outliers, MAC addresses of fixed devices around the APs, like Smart TVs, IoT devices, and others that send probe requests continuously and with high frequency at each time (also during the night), three categories have been implemented:

1. whitelist ⇒ 0
2. graylist ⇒ 1
3. blacklist ⇒ 2

In the whitelist category, there are all the MAC addresses that belong to the Politecnico's list, so for sure, all these addresses are not randomized.

In the blacklist category, there are all the MAC considered outliers, so they will not be displayed in the real-time dashboards; in Section 4.5.2, there is a detailed description of how a MAC address is regarded as an outlier and so added to the blacklist.

Last, in the graylist category, there are all the other addresses that are not in the previous categories, they are displayed in the real-time dashboard, and each day at a specific time, there is an automatic script that takes all the graylist addresses of the day before and analyzes them searching for new outliers. After the analysis of the graylist, all the addresses that are not outliers are "promoted" to the whitelist.

Figure 4.3 shows the full flow of operations done by the MQTT client when a new message arrives.

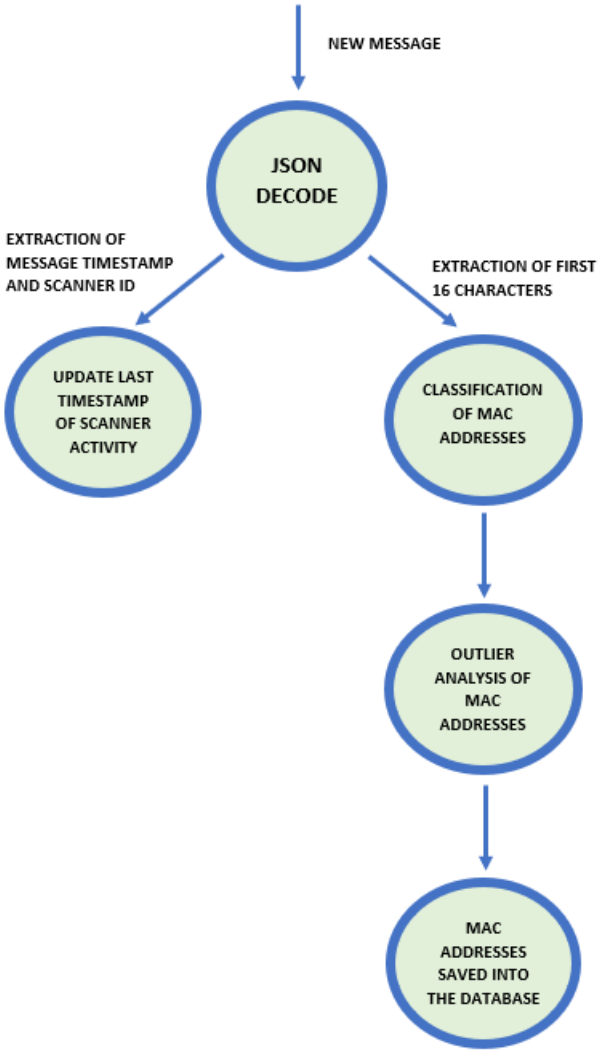


Figure 4.3: Data filtering flow

4.3 DB storage

The last operation done by the MQTT client after the data filtering is saving all the data, received and calculated, into the database. This is a fundamental operation to be able to do future analyses on data and in-depth studies more quickly without having to connect to the platform every time and download the necessary data.

The first choice made was to choose between relational and non-relational databases.

4.3.1 Relational database

A relational database is a digital database based on the relational model of data. The data is stored in tables containing rows (which represents an entry) and columns (which stores and sorts a specific type of information).

Advantages

- Can handle lots of complex queries, database transactions and routine analysis of data [25];
- ACID (Atomicity, Consistency, Isolation, Durability) properties that ensure reliable database transactions;
- Easy configuration, simple import and export of data;
- Lot of online support and compatibility with other software.

Disadvantages

- Cannot store complex or very large images, numbers, designs and multimedia products [25];
- Can become very costly with maintenance and new servers;
- Do not scale horizontally very well.

4.3.2 NON-Relational database

A non-relational, or NoSQL database, works differently. It has to deal with semi-structured data where each entry fits in a single JSON record. As a result, it can process ANY type of data without needing to modify the architecture.

There are four different types of NoSQL databases:

1. Document-oriented databases – This database is designed for storing, retrieving and managing document-oriented information. Document databases usually pair each key with a complex data structure;
2. Key-Value Stores – This database uses different keys where each key is associated with only one value in a collection, like a dictionary;
3. Wide-Column Stores – This database uses tables, rows and columns, but unlike a relational database, the names and format of the columns can vary from row to row in the same table;
4. Graph Stores – A graph database uses graph structures for semantic queries with nodes, edges and properties to represent and store data.

Advantages

- Large volumes of structured, semi-structured and unstructured data [25];
- Object-oriented programming that is easy to use and very flexible;
- Efficient, scale-out architecture instead of expensive, monolithic architecture.

Disadvantages

- Less support since NoSQL databases are usually open-source [25];
- NoSQL databases require technical skill to install and maintain;
- Less mature - NoSQL databases are still growing, and many features are still being implemented;
- Less compatibility with other software.

4.3.3 Implemented database

The final choice was highly influenced by the visualization tool (more details in Section 4.4) because all the open-source tools considered can only deal with time-series databases or relational ones. Any visualization tool allows using non-relational database as data source.

The implemented database is a MySQL database; it has six tables, one for each APs scanner. Each table has the same structure:

- id - int auto-generated
- timestamp - DateTime
- mac - char(16)
- type - tinyint(1) with 0, 1, or 2 as possible values (see Section 4.2)
- firewall - tinyint(1) with 0, 1, or 2 as possible values (see Section 4.2)

As explained before, in the database, only the first 16 hexadecimal digits are saved with the classification type (professors and administrative employees, students or non-Politecnico community members) and the outliers identification value called "firewall" on the database.

4.4 Data visualization

For the visualization tool, the two options were Grafana [6] and Kibana [10].

Grafana is a multi-platform open-source analytics and interactive visualization web application. It permits to create complex monitoring dashboards using interactive query builders on databases and provide charts, graphs and alerts.

Kibana is an open-source frontend application that sits on top of the Elastic Stack, ELK (Elasticsearch + Logstash + Kibana), providing search and data visualization capabilities for data indexed in Elasticsearch.

4.4.1 Grafana vs. Kibana

Grafana	**Kibana**
Grafana is designed for the analysis and visualization of metrics	Kibana is mainly used for analyzing log messages
Grafana was designed to work as a UI for analyzing metrics, and it can work with different time-series databases	Kibana was designed to work only with Elasticsearch and does not support any other type of data source
Grafana has a built-in access control mechanism that allows restricted access to dashboards only to authorized users	Kibana's dashboards are open and accessible to the public
Grafana has a built-in alerting mechanism that allows users to insert conditional rules to dashboard panels	Kibana does not offer an alerting mechanism in the free version
Both Grafana and Kibana have a greater variety of customization options and allows an easier and more immediate modification of the various panel options	

Table 4.1: Grafana vs. Kibana

4.4.2 Dashboard and database comparison

Grafana MySQL	PRO: - Allow historical series (add data in the past); - DB easily readable, editable and exportable; - MAC address directly saved in the DB; - Very light graphic tool $\sim$ 300 MB. CONS: - Very large DB; - Loading time depends on operation time on the DB; - Alerting problem: new data with zero value needed.
Grafana Prometheus	PRO: - Very light and optimized DB; - Initial loading time $\sim$ 1s; - Loading time for each refresh $\sim$ 1s; - Very light graphic tool $\sim$ 300 MB. CONS: - NOT allow historical series (add data in the past); - Data in the DB NOT accessible and NOT editable; - DB stores only the count of MACs detected in a given time; - Log file required to save data for future analysis; - Alerting problem: new data with zero value needed.
Kibana Elasticsearch	PRO: - Allow historical series (add data in the past); - DB viewable only through the graphic tool; - MAC address directly saved in the DB. CONS: - Very large DB; - Data in the DB NOT editable; - Very slow time to insert historical data; - Alerting problem not available in the free version; - Graphics not customizable like Grafana; - Heavy graphics tool $\sim$ 1.5 GB.

Table 4.2: Grafana MySQL vs. Grafana Prometheus vs. Kibana Elasticsearch

The following table (Table 4.3) compares the size on disk of a log file, a MySQL database and an Elasticsesarch one, how the size is affected by increasing data an estimate of the annual size of the data.

	Log file	**MySQL**	**Elasticsearch**
True size on disk	50 MB 100 MB	55 MB (+10%) 110 MB (+10%)	78 MB (+56%) 175 MB (+75%)
Estimated annual size for 6 scanners always working (MACs only 16 Byte)	2 GB	2.2 GB	Not estimable (huge)
New column add (cost in size)	+10%	+10%	Not estimable

Table 4.3: Size comparison between Log file vs. MySQL vs. Elasticsearch

4.4.3 Implemented Dashboards

Putting together all the previous considerations, the choice was to use Grafana as a visualization tool because it offers much freedom to customize dashboards, graphs and above all, it has a wider selection of data sources, which Kibana is not able to offer. With Kibana, the database would have become unsustainable in size in a very short time.

Grafana runs in another VM than the one where the MQTT client and MySQL database are deployed. The two VMs are on the same local network, so they communicate with each other by using integrated Grafana API and SQL queries. The MySQL database, containing all the data collected by the APs, is the data source for the dashboards.

There were implemented seven different dashboards, one for each scanner with five different graphs (one for all users, one for only Politecnico students with a relative heatmap and one for only Politecnico employees with a relative heatmap) and one for all the scanners together for an immediate comparison. More details, examples and figures in Chapter 6.

All the dashboards are interactive, so it is possible to change the time horizon and zoom in a specific time interval to see more in detail all the data. Also, it is possible to set or modify the refresh rate of the data to see real-time values coming from the scanners.

4.5 Automatic analysis scripts

Together with the MQTT client, there are two other Python scripts that run every day during the night when data traffic is less heavy.

1. The first script analyzes the log files saved in the log directory of the VM, searching for anomalies in the operation of scanners. This has been implemented because even if the scanners are high-end commercial products, in the almost one year from the date of installation, they have had numerous problems and we often noticed malfunctions even a few days later. Instead, thanks to this script, every day is verified the correct functioning of all the APs scanners;

2. The second script instead works on database data. For each scanner, it downloads the data of the day before with value of firewall = 1, so the graylist one. Then it searches for newer outliers and finally updates the firewall value from 1 to 0 if the MAC is not in the blacklist.

4.5.1 Scanner anomalies analysis

As said in Section 4.2, the first action that the MQTT client does when arrive a new message is to decode it and after that update the timestamp, relative to the scanner that has sent the message, in the log file saved on the VM disk. So at each message corresponds a write on disk to update the timestamp.

Every day at 2:00 AM, a Python script automatically runs to check if there are any anomalies. In particular, for each scanner, it opens the relative log file, takes the timestamp written on it and if the current time minus the timestamp is more than 3600 seconds (1 hour), it saves in another file the name of the scanner. Then if the scanner name was not present in the list of non-working scanners, it sends an email to notify the new non-working scanner, also specifying the last timestamp detected. Instead, if it was already present in the list of non-working scanners, it does not send the email to not be wordy. Also, if a scanner was on the non-working list, but now it works, it sends an email to notify that now the scanner is working properly.

Thanks to this monitoring system, we do not need to check every day that everything is working right. Only when we receive an email with the name of the scanner/s that does not work we can investigate the cause of the malfunction or report the problem to TIM for further investigation.

4.5.2 Outliers analysis

As explained in Section 4.2, before saving the data on the database, a firewall value is assigned to the MAC address. If the MAC belongs to the Politecnico's list, it is whitelisted with firewall = 0, if it belongs to the blacklist the value of firewall is set to 2, all the other values instead have as firewall value 1, so they are in the graylist.

Every day at 2:45 AM, a Python script automatically runs with the aim to find new possible MAC addresses to add to the blacklist and to update the firewall value for all the graylist if they are not outliers. More in particular, the first operation done by the script is to open a connection with the database and retrieve all the data from NOW() to 1 day before that have firewall = 1. The MAC addresses are saved into a dictionary, a Python structure, where the MAC is the key and the value is an array of timestamps associated with the MAC. After that for each candidate is evaluated the number of timestamp per hour. If there are more than 15 timestamps per hour for at least 20 hours over 24 (empirically chosen), this MAC is considered an outlier so it is added to the blacklist and no more considered by the visualization tool.

After the update of the blacklist with new MAC address, if any, all the database tables are updated. All blacklisted MACs will have firewall 2 value in the database, while all other values with firewall 1 will be updated with value 0. See Appendix A.1 for the complete Python code.

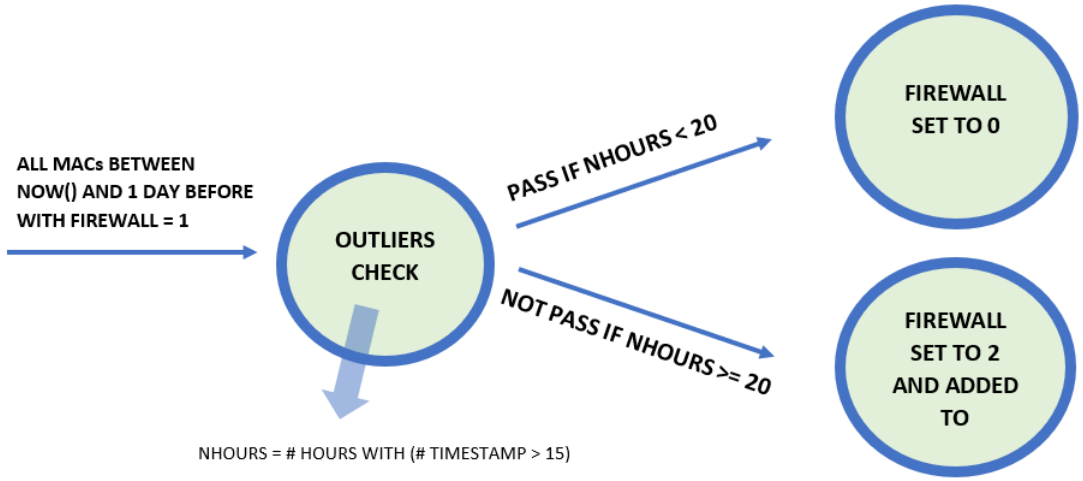


Figure 4.4: Flow of the outliers analysis

Chapter 5

Experimental evaluation

This chapter will show and analyze the most important and significant results obtained.

As said before, even though these scanner APs are high-end commercial products, they have had numerous problems. For the Bluetooth data, the scanners worked intermittently, for some periods, they captured the data correctly while in others, inexplicably, they did not capture anything. For the WiFi interface, we were luckier because it has had fewer problems. The scanner 7 had and still has issues, from the first days of January 2020, it stopped working completely (both WiFi interface and Bluetooth interface). Currently, it has been removed from the traffic light pole, where it was installed, and taken to a TIM laboratory where some experts are investigating the causes of the malfunction.

Bluetooth data are, compared to WiFi data, lower in number by a factor of 10 as it is much more common to find the WiFi interface enabled than the Bluetooth one in people's mobile devices. Also, the Politecnico's list of MAC addresses is referred only to the WiFi interface; for Bluetooth, we do not have any data to classify Politecnico community members.

For all this reason, for studying the various patterns of people's mobility, flows and data analysis, we focused almost exclusively on the data coming from the captures on the WiFi interface.

All the data presented are collected from the 1st October 2019, and the captures will continue for other years for a complete analysis.

In Table 5.1, it is possible to see the size of the table for each scanner almost after a year of detection. Also, it is possible to see, for each scanner, the total number of distinct MACs and the total ones.

Table	**Size of table**	**Total MACs**	**Distinct MACs**
Scanner 4	240 MB	5.5 M	2.2 M
Scanner 5	300 MB	6.7 M	2.2 M
Scanner 6	510 MB	10.7 M	5.6 M
Scanner 7	140 MB	2.7 M	1.5 M
Scanner 8	150 MB	3.0 M	1.4 M
Scanner 9	301 MB	6.3 M	3.3 M

Table 5.1: Detailed information about the implemented database

All the numbered scanners in the following charts and graphs are referred to the numeration in Figure 3.2.

5.1 Graphs relating to the effect of the lockdown due to COVID-19

The following graphs show the trend of the number of MACs detected in 4 consecutive weeks between the second half of February and the first half of March. Very important and relevant weeks because in the last two weeks of February, the Winter session exams were present at the university, so the Politecnico area was crossed by numerous students, professors, and employees of the Politecnico. While from 9th March the lockdown officially began throughout Italy, bringing people's movements to the bare minimum.

5.1.1 Bluetooth interface

As we can see in Figure 5.1 and 5.2, many more MAC addresses were detected in the three weeks before the lockdown than in the first week of total closure (red segment in the graph); about 50-60% fewer detections than in previous weeks.

It is also very evident that there are far fewer people passing by on Saturdays and Sundays than on working days. It can also be seen how the profiles of the curves of the various weeks are similar.

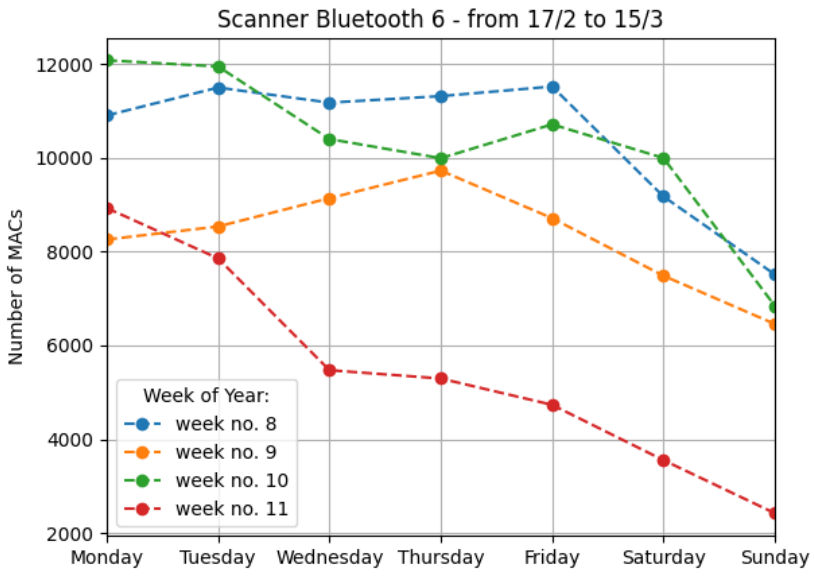
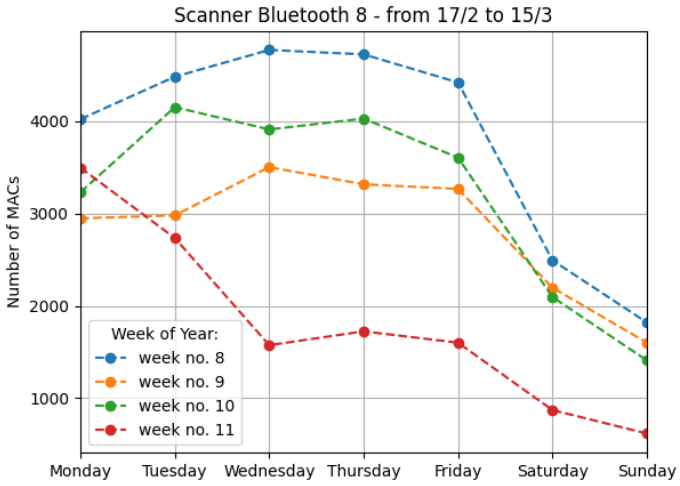
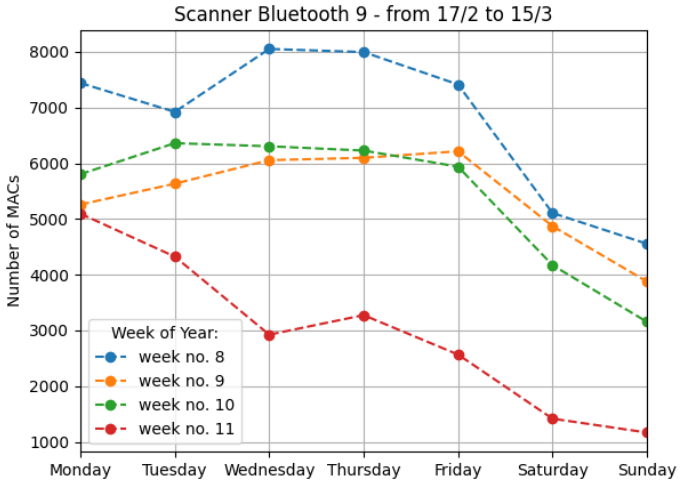


Figure 5.1: Lockdown effect Scanner 6 - Bluetooth

Scanner 6 is near Porta Susa train station, so it is the one with the higher number of detections. Scanner 9 is also in a very strategic point because it is in front of OGR (*Officine Grandi Riparazioni*) that is home to offices, exhibitions, co-working spaces and much more, including, in the lockdown period, a temporary hospital. Scanner 8 and 9 are separated by a very busy road crossed by both cars and bicycles, thanks to the cycle path, and from people moving towards the station and the campus.



(a) *Scanner 8*

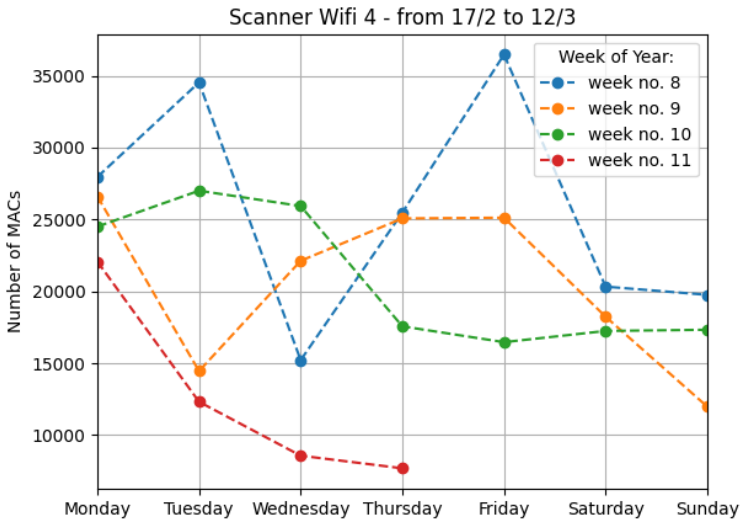


(b) *Scanner 9*

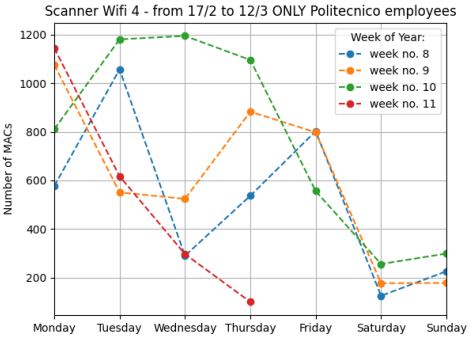
Figure 5.2: Lockdown effect Scanner 8 and 9 - Bluetooth

5.1.2 WiFi interface

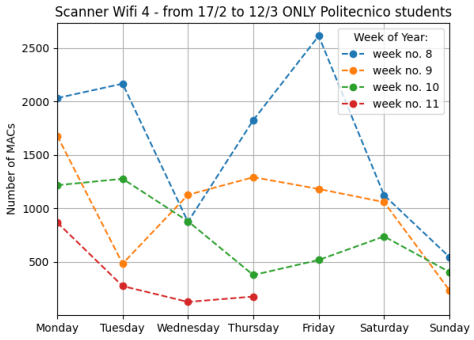
In the next series of graphs (Figures 5.3, 5.4, 5.5, 5.6), there are three graphs each; in particular in graph (a) are considered all the users, so both members of the Politecnico and not. In graph (b) only Politecnico employees and in graph (c) only Politecnico students.



(a) *All users*



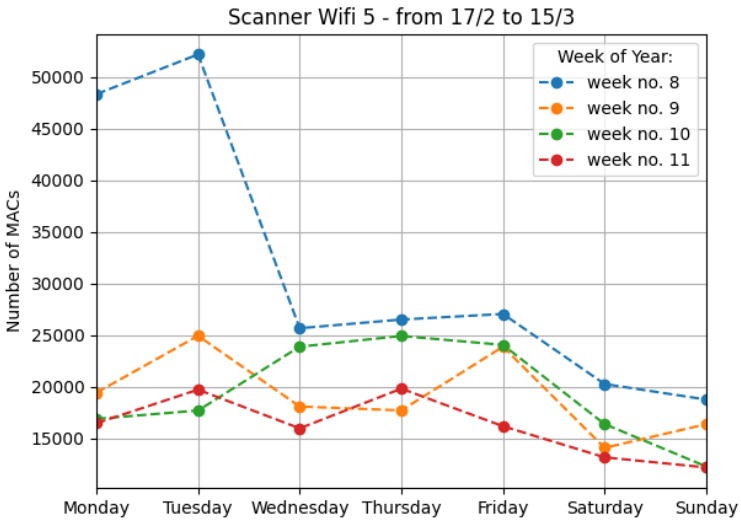
(b) *Only Politecnico employees*



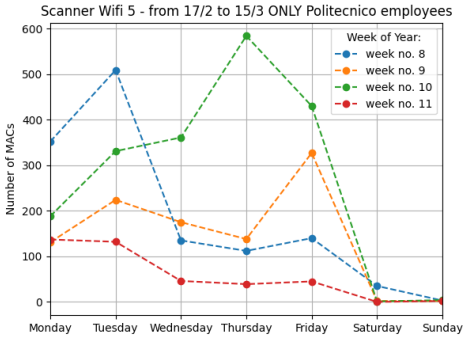
(c) *Only Politecnico students*

Figure 5.3: Lockdown effect Scanner 4 - WiFi

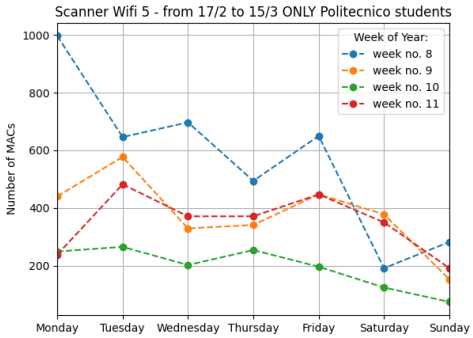
Unfortunately, as we can see from the graph, the scanner 4 stopped working on 12th March because the pole on which it is mounted is a pole of a gate inside the Politecnico and the 12/03 the electricity was cut following the closure of the whole campus caused by the lockdown.



(a) *All users*



(b) *Only Politecnico employees*

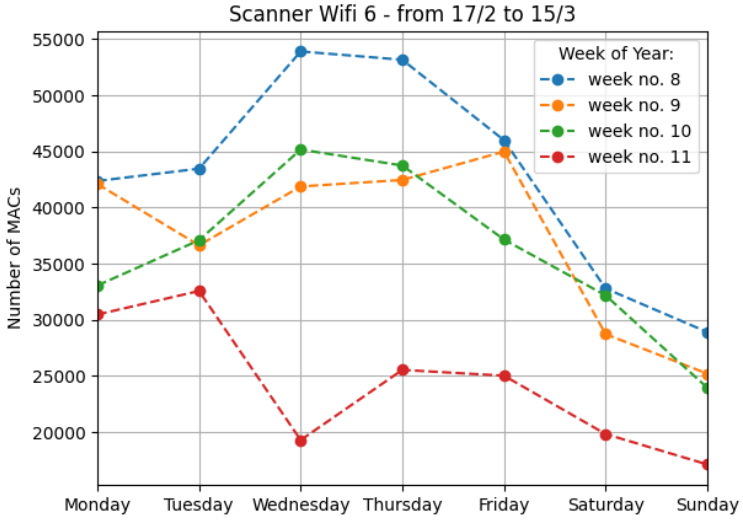


(c) *Only Politecnico students*

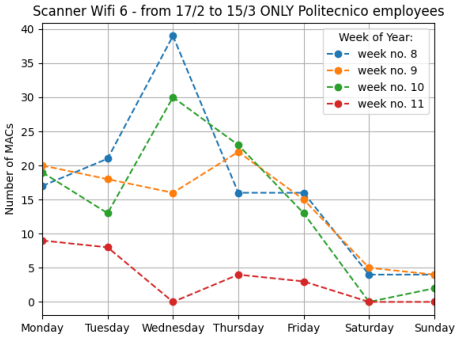
Figure 5.4: Lockdown effect Scanner 5 - WiFi

As can be seen from the graphs, the number of Politecnico employees detected by scanners 4 and 5 is much higher than the number detected by the other scanners in the area; this is because scanners 4 and 5 are located within the boundaries of the Politecnico itself and close to the main entrances.

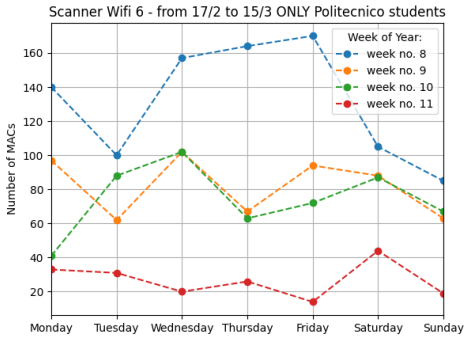
It is interesting to see that certainly 10% of Politecnico employees walk (or ride) along Corso Castelfidardo in the direction, most likely, of the Porta Susa train station.



(a) *All users*



(b) *Only Politecnico employees*

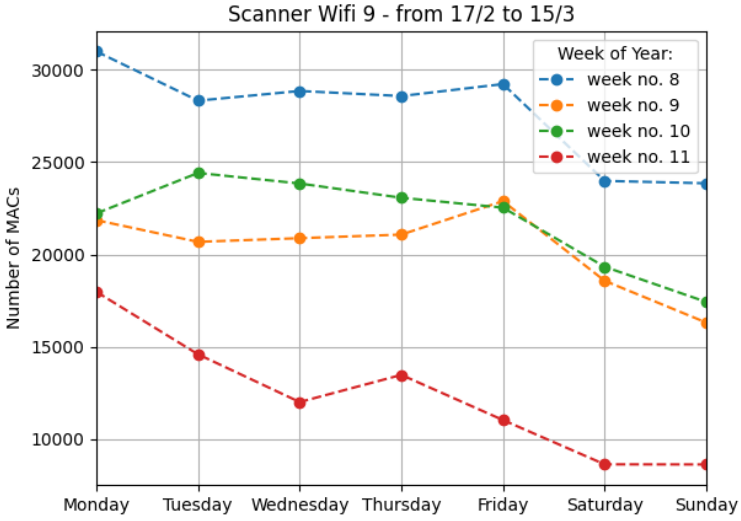


(c) *Only Politecnico students*

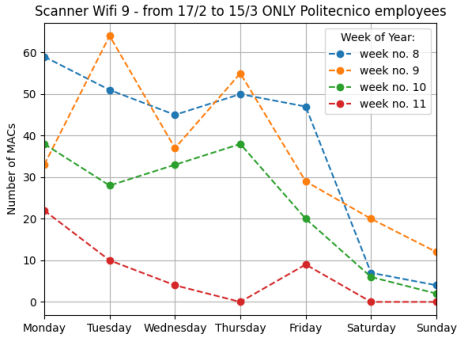
Figure 5.5: Lockdown effect Scanner 6 - WiFi

Another interesting point is that before the lockdown on Saturday and Sunday, the number of people detected is almost the same in each graph, except for the week no. 8, which had more detections.

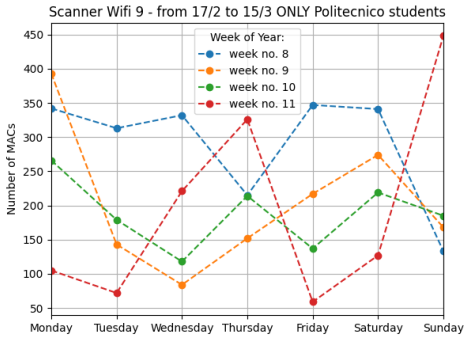
Regarding the graphs (b) of the Politecnico employees, it can be observed that on Saturdays and Sundays, the number of detections is close to zero; this is the expected result as there are (usually) no lessons or exams on weekends. While on Wednesday, there is a peak in attendance.



(a) *All users*



(b) *Only Politecnico employees*



(c) *Only Politecnico students*

Figure 5.6: Lockdown effect Scanner 9 - WiFi

From the graphs (c) of the Politecnico students, it can be highlighted as starting from week no. 9, when the exams of Winter session were suspended, the number of detections dropped significantly.

In graph (c) of Figure 5.6, it is also possible to see that the trend of the curve of week no. 11 is anomalous as it has peaks higher than all the previous weeks despite the lockdown.

The effect of non-school days, Saturdays and Sundays, is not as evident as it is for Politecnico employees; this could mean that many students live near the Politecnico and pass through the campus area even on weekends.

5.2 Heatmap charts for mobility patterns frequency

The following charts show, through a heatmap, the frequency of detections for each pattern as the hours' change. All data used for the charts are data captured between the 1st October 2019 and the 25th April 2020, except for scanners 4 and 7, which stopped working on 13th March and 2nd January, respectively.

The term *pattern* identifies a sequence composed of 1 or more scanners in which the same MAC address has been detected between one scanner and the other within 5 minutes (e.g., pattern 489 means that the same MAC address has been detected first by scanner 4 then by scanner 8 and last by scanner 9, and the delta between the timestamps of two consecutive detections is less than 5 minutes or 300 seconds).

The darker the color, the higher the number of detections (higher frequency) and vice versa for the lighter color that means lower frequency.

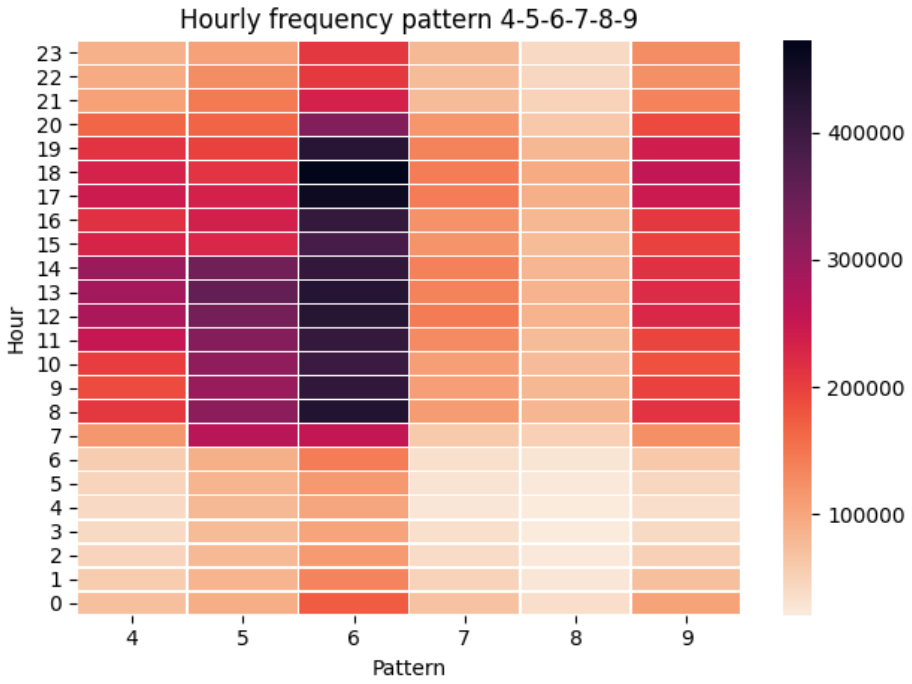


Figure 5.7: Heatmap chart - All users - Scanners 4-5-6-7-8-9

In Figure 5.7, all the single patterns are represented, so all the devices that have been detected by a scanner and then either "disappear", no longer pass under a scanner, or take longer than the average travel time, and therefore this is not considered as the same flow or, finally, the MAC address is changed due to randomization effect.

As we expected, the street covered by scanner 6 is the busiest as it is close to the train station. Also, it is normal that from 8 to 20, we have the 70/80% of all the detections of the day except for scanner 6, where the time interval extends up to 1 am.

Frequencies in Figure 5.7 are very high compared with the other charts because the data suffer a lot from the MEC address's randomization.

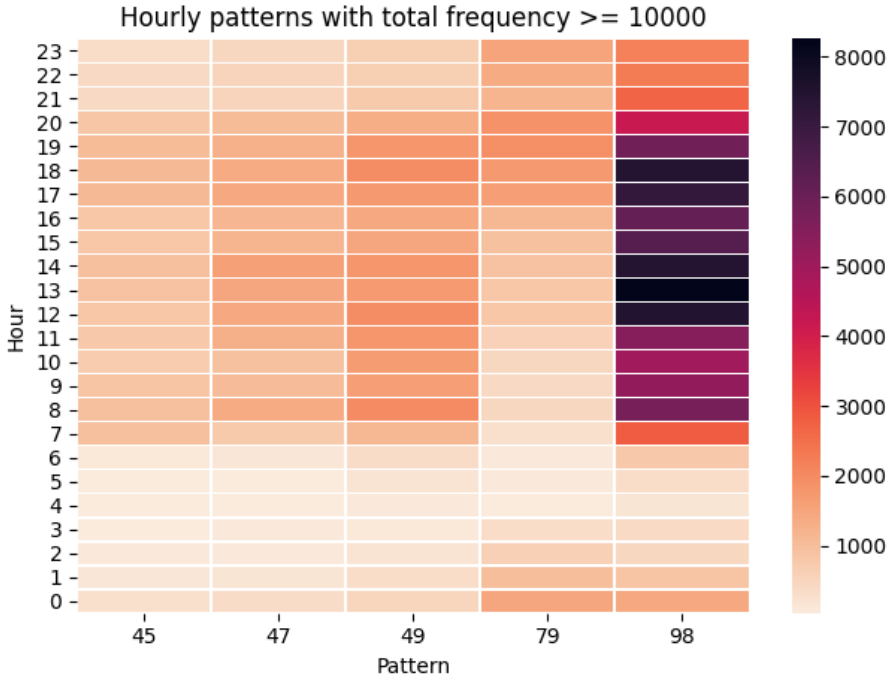


Figure 5.8: Heatmap chart - All users - Total freq. $\geq$ 10000

The *total frequency* is calculated by summing all the hourly frequencies.
In Figure 5.8, we can see the eight most popular patterns. It is interesting that pattern 98 is the most popular; it represents the pedestrian crossing of Corso Castelfidardo to go in the direction of Corso Duca Degli Abruzzi.

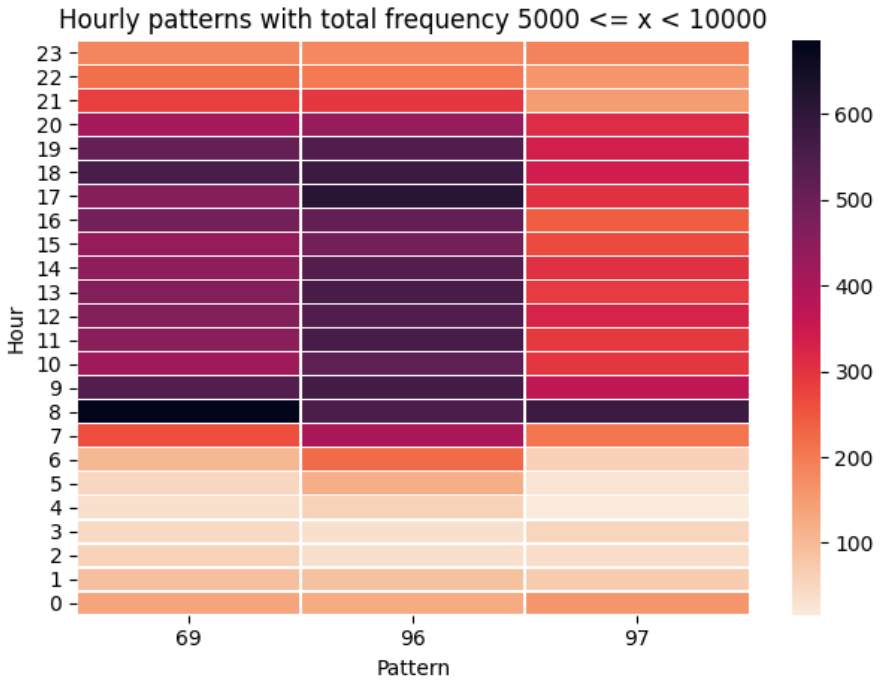
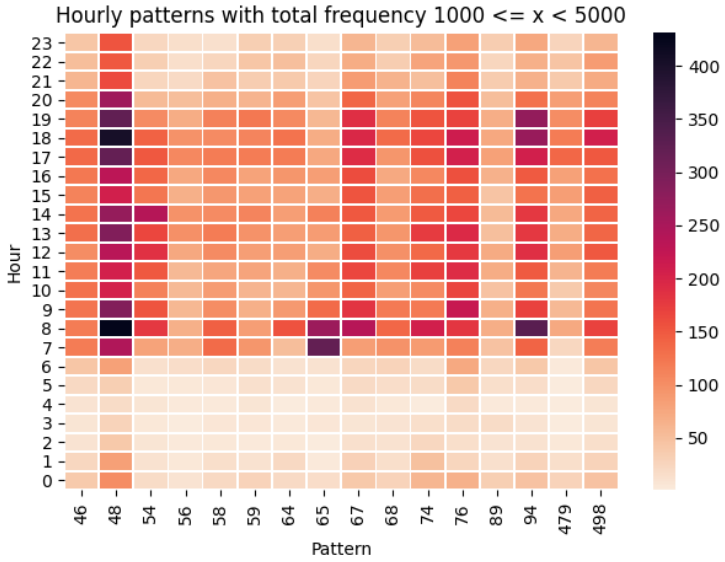


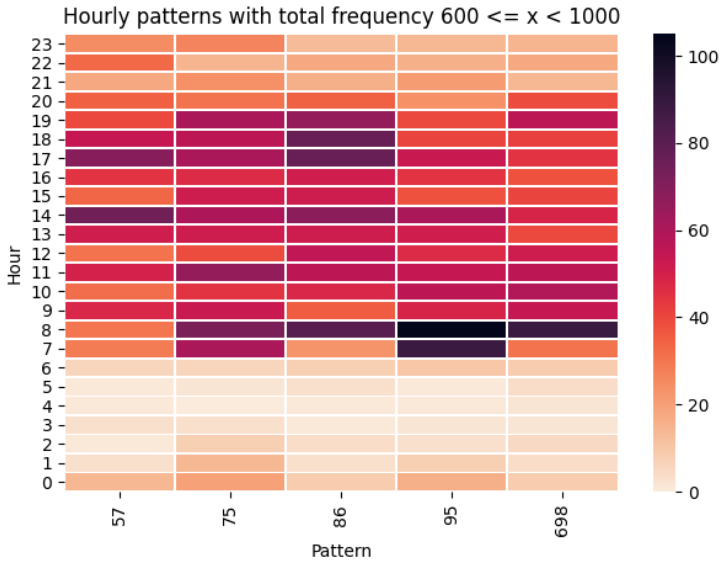
Figure 5.9: Heatmap chart - All users - Total freq. $5000 \leq x < 10000$

It is also interesting that the opposite of patterns 47 and 49 are not frequent, this could be explained by the fact that people going to the OGR cross the road near the Politecnico while on their return, they continue on the sidewalk along the General Motors.

As we can see in Figure 5.9, the patterns 69 and 96 are very similar, meaning that the two patterns are exploited equally in both directions. Also, despite the few data relating to scanner 7, pattern 97 is one of the most frequent.



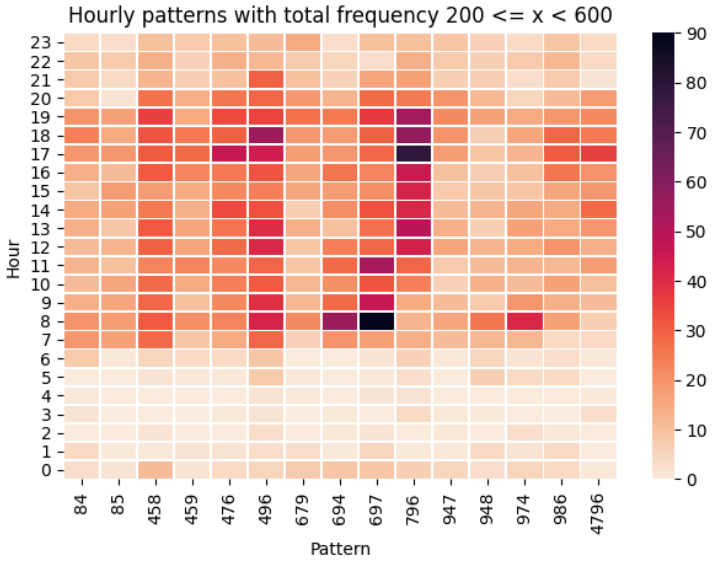
(a) *Patterns with total frequency* $1000 \leq x < 5000$



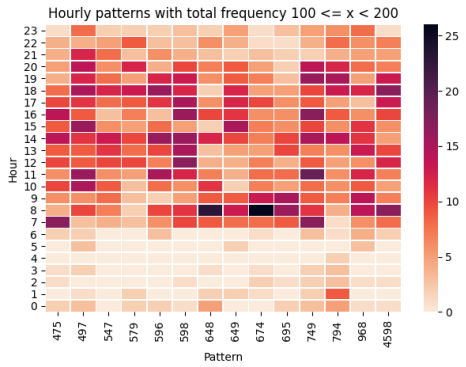
(b) *Patterns with total frequency* $600 \leq x < 1000$

Figure 5.10: Heatmap charts - All users - Total freq. $600 \leq$ x < 5000

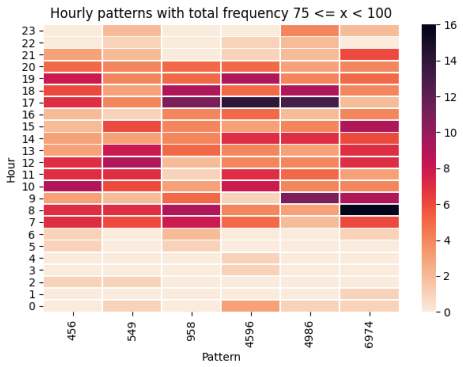
Pattern 48 is very popular thanks to the presence of the cycle path travelled by many people both by bike and with electric micro-mobility (e.g., electric scooters or electric bikes).



(a) *Patterns with total frequency* $200 \leq x < 600$



(b) *Patterns with total frequency* $100 \leq x < 200$



(c) *Patterns with total frequency* $75 \leq x < 100$

Figure 5.11: Heatmap charts - All users - Total freq. $75 \leq x < 600$

In figure 5.11, there are patterns with lower frequency. We must take into account some factors:

- for scanner 7 we have only data about three months over seven;
- for scanner 4 we do not have almost 45 days of data;
- patter 48 and 84 are certainly affected by the effect of the Winter season and therefore the less use of the cycle path;
- the randomization effect on long paths.

Charts in log10 scale and sorted by decreasing total frequency from sx to dx

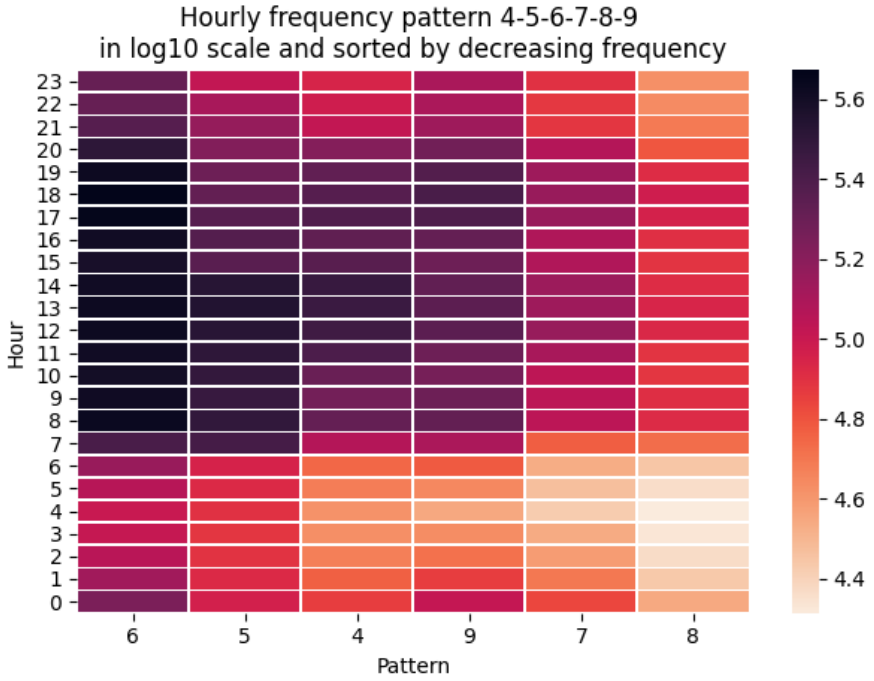


Figure 5.12: Heatmap chart - All users in log10 scale - Single patterns

In Figure 5.12 and 5.13, it is very easy to see which are the most popular patterns and compare the hourly frequencies.

The two charts are represented in a log10 scale and sorted graphically by decreasing frequency from left to right.

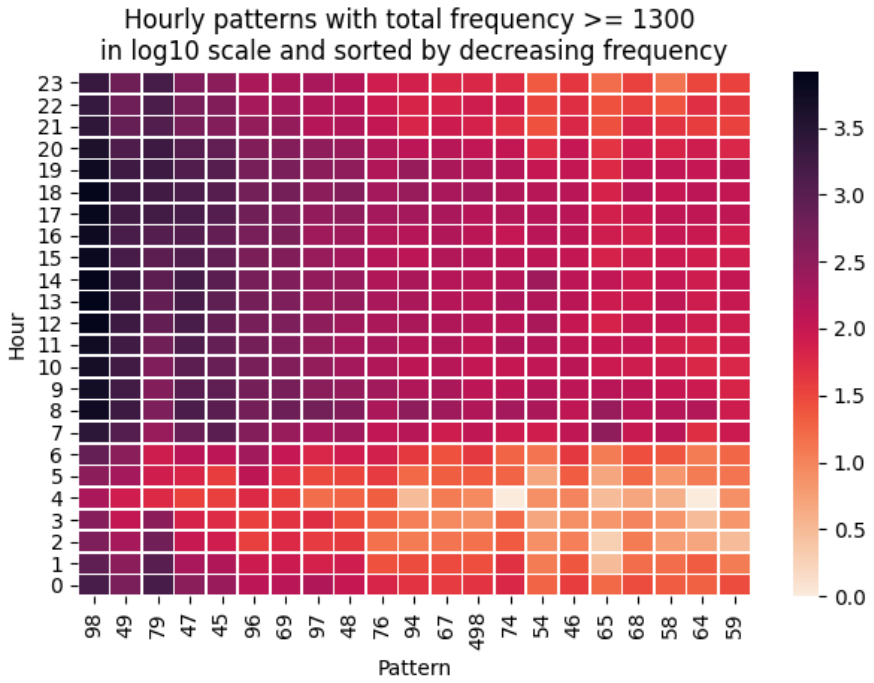


Figure 5.13: Heatmap chart - All users in log10 scale - Multiple patterns

It is very evident that between 7 and 8 in the morning, there is the first peak of people detected and that in any time slot the frequency is zero, also during the night.

It is also interesting to see that in the top 10 most frequent patterns, four have scanner 7 as one of the scanners, despite the limited data available for that scanner; undoubtedly due to the fact that the sidewalk on the OGR side is very busy.

5.2.1 Heatmap charts with only Politecnico members

As we could imagine, the scanners with the highest number of detections, for only Politecnico community members, are scanners 4 and 5, as they are located within the perimeter of the Politecnico itself.

Also, the hourly frequencies reflect the employees' working hours and the hours of lessons/exams for the students.

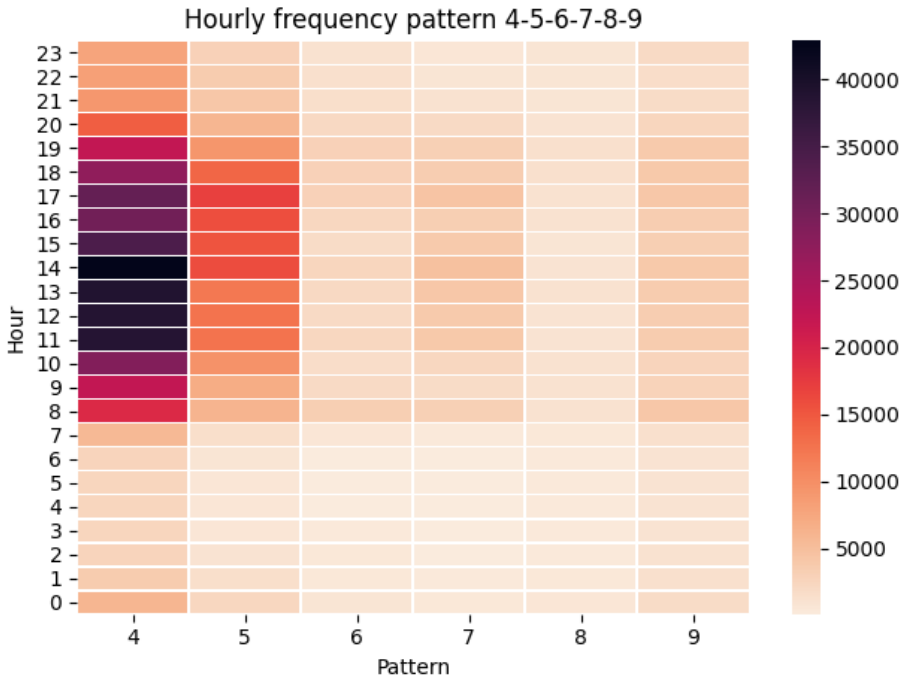


Figure 5.14: Heatmap chart - Single patterns of Politecnico members

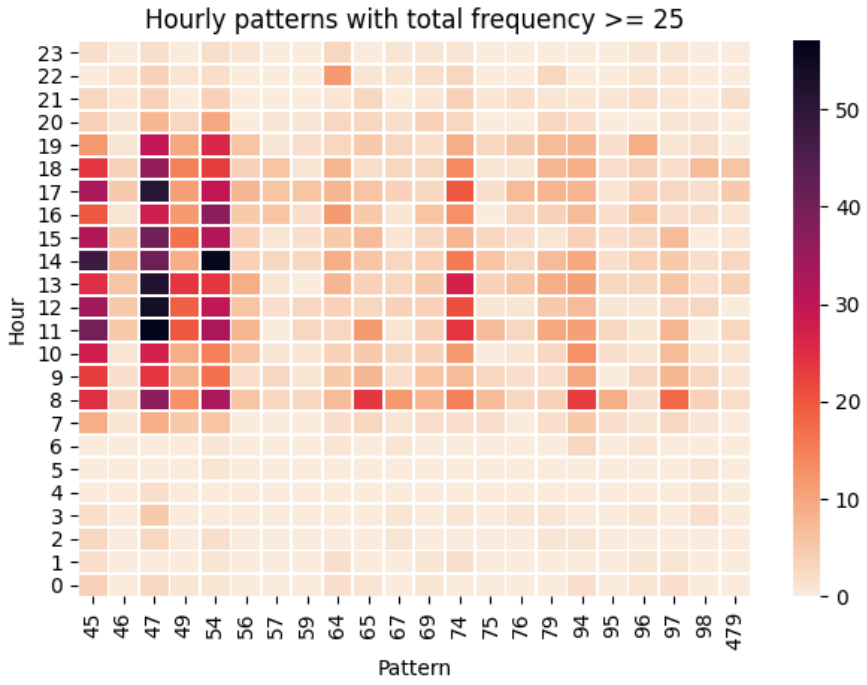


Figure 5.15: Heatmap chart - Multiple patterns of Politecnico members

Here in Figure 5.14 and 5.15, we can see the most frequent mobility patterns for the Politecnico community members.

The most popular patterns (47, 45 and 54) are those relating to the movements of students and teachers from one part of the campus to another or relating to movements between the headquarters and the Politecnico's *Cittadella*, across the street.

Charts in log10 scale and sorted by decreasing total frequency from sx to dx

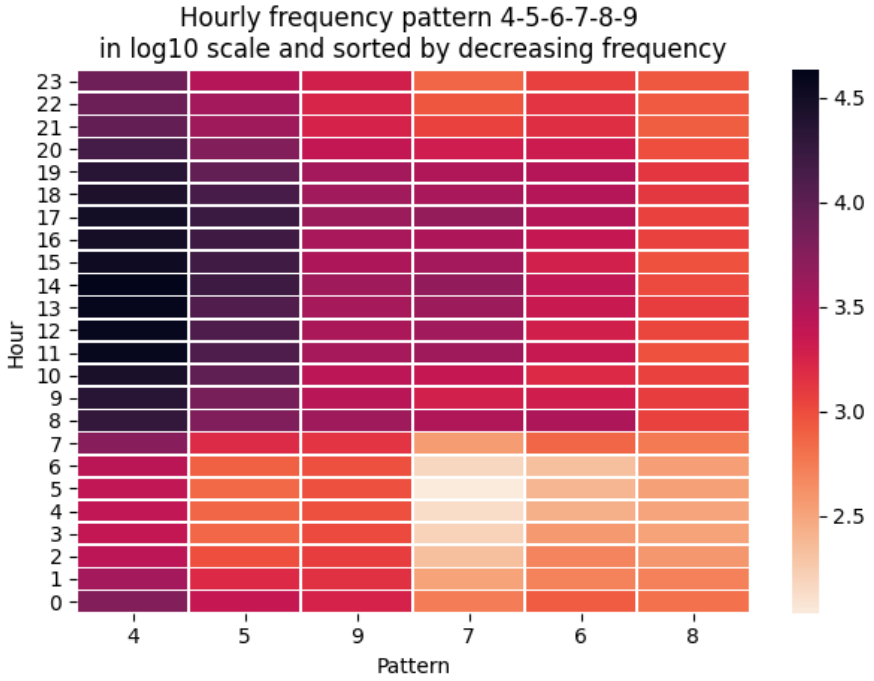


Figure 5.16: Heatmap chart - Single patterns of Politecnico members in log10 scale

Here in Figure 5.16 and 5.17, like before, there are two charts where the data are represented in a log10 scale and sorted graphically by decreasing frequency from left to right. Also, the data are filtered for only the Politecnico community members.

As expected, the two single patterns with the highest number of detections are those relating to scanners 4 and 5, which are located within the perimeter of the Politecnico.

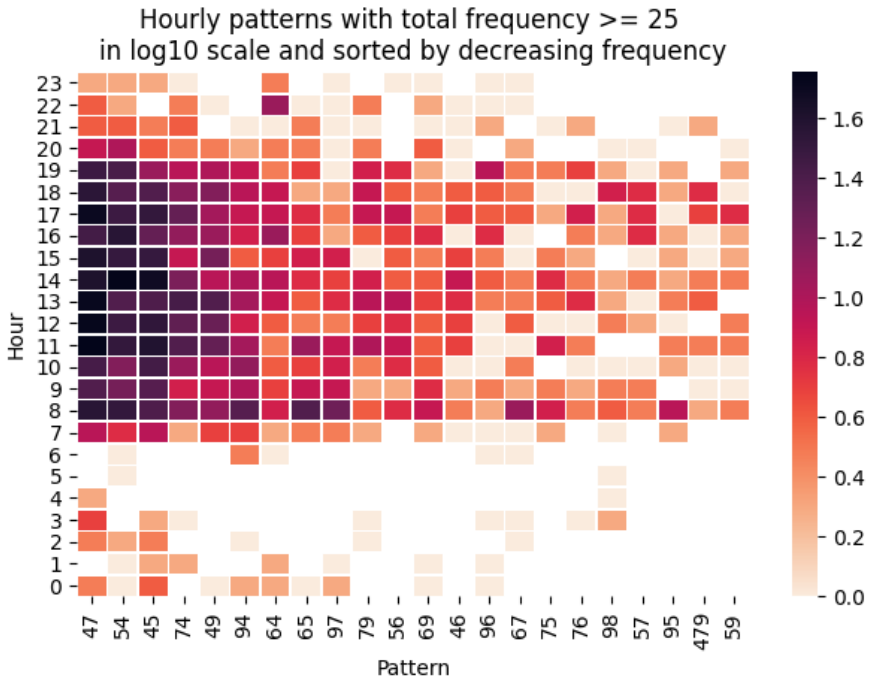


Figure 5.17: Heatmap chart - Multiple patterns of Politecnico members in log10 scale

In Figure 5.17, it is possible to see and compare the hourly frequencies of the different patterns.

It is interesting how many patterns have no detections in some time slots (white squares in the chart of Figure 5.17), especially early in the morning.

Also, before the Politecnico opening hour, at 7:30 am, there are very few detections than during the other working hours.

5.3 Mobility flows direction

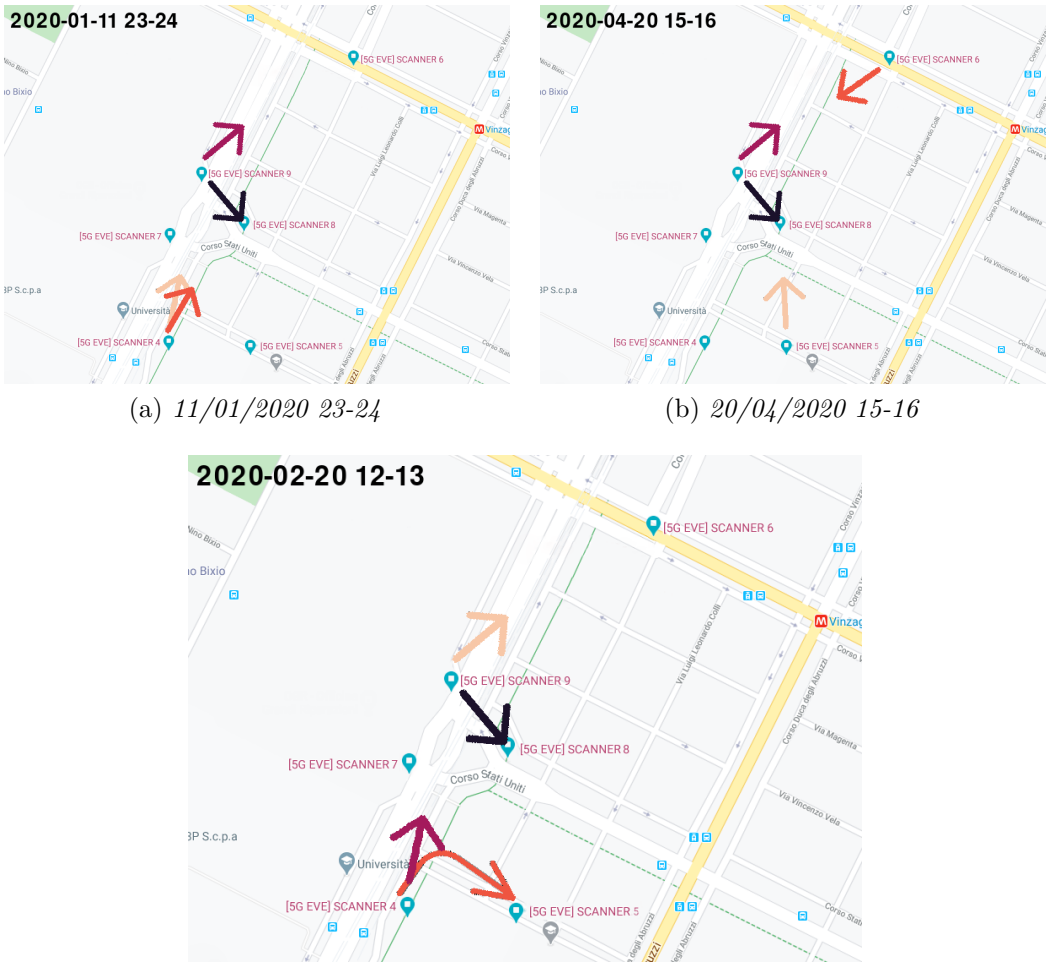
This section shows a series of images where, thanks to some colored arrows superimposed on the map of the Politecnico area, the 3/4 most frequent patterns on a given day and in a date-time slot are highlighted.

As in Section 5.2, the color tone differentiates the detection frequency, where it is darker means higher frequency, and vice versa for lighter colors.

Figure 5.18: Mobility flow on 08/12/2019 12-13

In Figure 5.18, we can see the mobility flow directions of the 8th December 2019 between the 12 and the 13, where the most frequent pattern is 79, followed by 47, then 96 and finally 69.

So we can say that the main flow was from the Politecnico to Porta Susa train station.



(a) *11/01/2020 23-24*

(b) *20/04/2020 15-16*

(c) *20/02/2020 12-13*

Figure 5.19: Mobility flows

In Figure 5.19 above, there are three other examples that show the mobility flows in different days and times.

With this type of image, it is very easy to understand the mobility flows thanks to the arrows on the real map of the area covered by the 6 APs scanners.

The images are created with the Python library pygame [24], first are calculated the top 4 patterns in the date and hour selected, then each arrow is rotated and filled up with the right color.

5.4 Mobility type

In this last section of the results chapter, we want to analyze, by using some graphs, the type of mobility of people in the most popular patterns.

In particular, the patterns between scanners that are not close to each other will be analyzed, as it is easier to see multiple types of mobility not overlapping each other. If we analyzed patterns such as 45, 98 or 79, it would be challenging to distinguish pedestrians from bicycles (or electric scooters) and cars (or motorbikes) as the scanners are very close and travel times would be about the same.

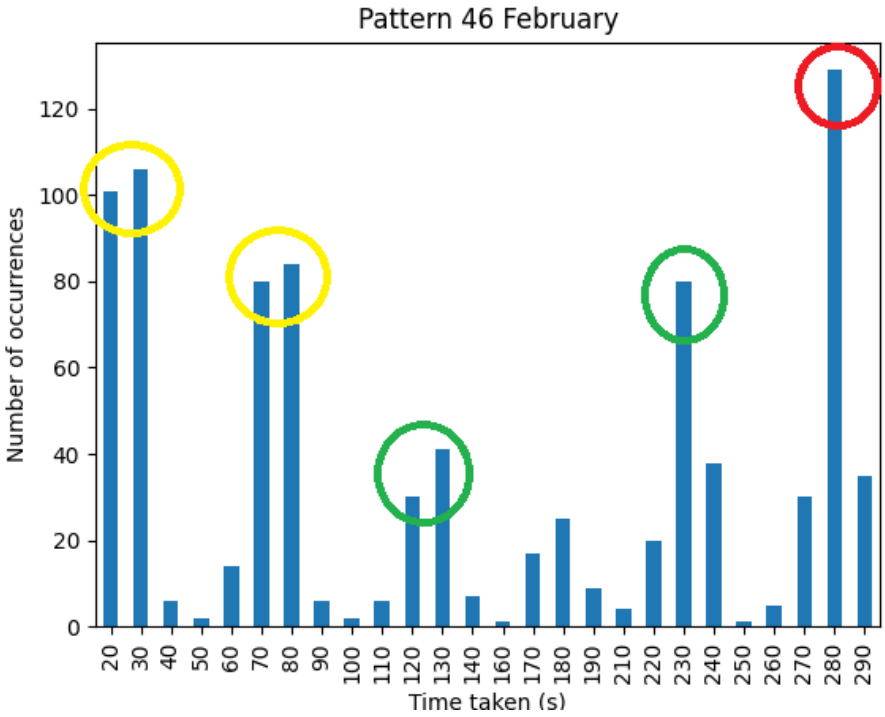
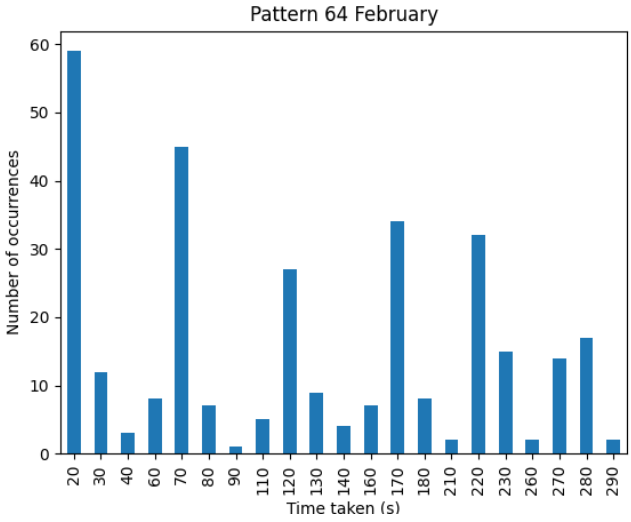


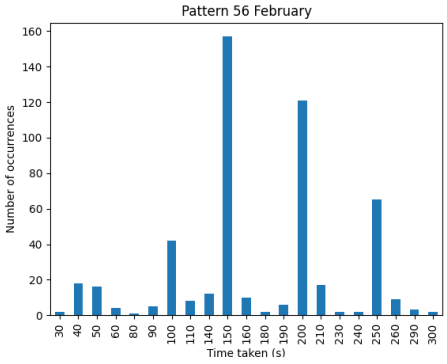
Figure 5.20: Mobility type - Pattern 46

In Figure 5.20 for simplicity, the five main peaks have been highlighted using colored circles. Between scanners 4 and 6, there are approximately 450 meters. The first yellow circle on the left corresponds to an average travel time of 25 seconds (an average speed of 65 km/h), while the second about 75 seconds (an average speed of 22 km/h). Followed by the two green circles with average travel times of 125 and 230 seconds (corresponding to 13 km/h and 7 km/h respectively). Finally, the red circle with an average travel time of 280 seconds (about 5 km/h).

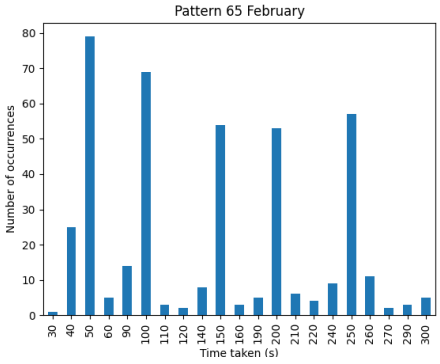
From the travel speeds expressed above, it is evident that the two peaks circled in yellow most likely correspond to cars (or motorbikes) or, in the second peak, some bikes or electric scooters could be included. The two peaks circled in green correspond to people on bicycles and in the first peak, probably, to some cars in traffic. Finally, the peak circled in red corresponds to all the people who have moved on foot.



(a) *Pattern 64*



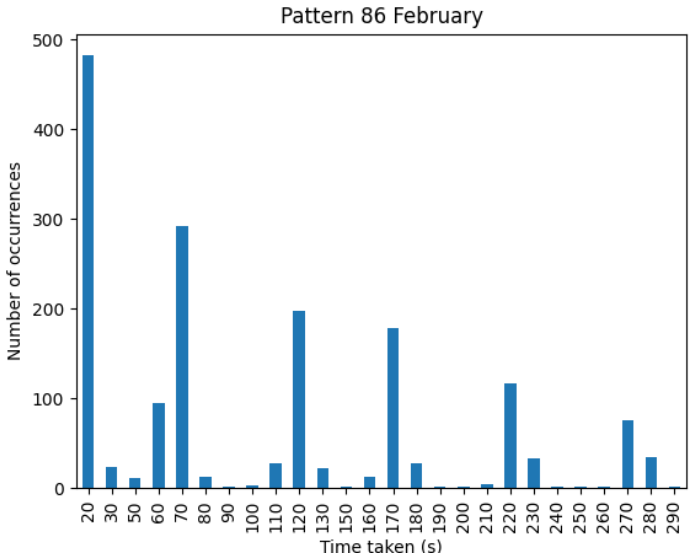
(b) *Pattern 56*



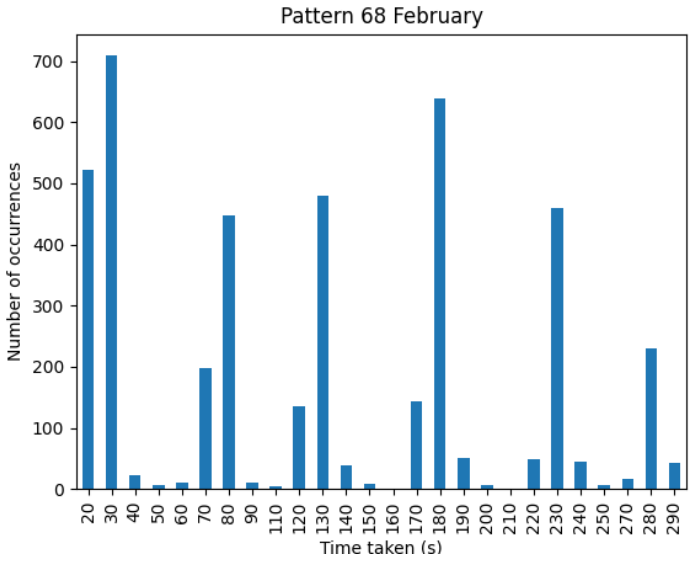
(c) *Pattern 65*

Figure 5.21: Mobility type - Patterns 64, 56 and 65

Figure 5.21 shows that pattern 56 has a slightly different behaviour from all the other patterns analyzed, indeed it does not have five evident peaks but only three starting from the 150th second onwards. The behaviour is strange when compared with the opposite pattern, graph (c) in Figure 5.21.



(a) *Pattern 86*

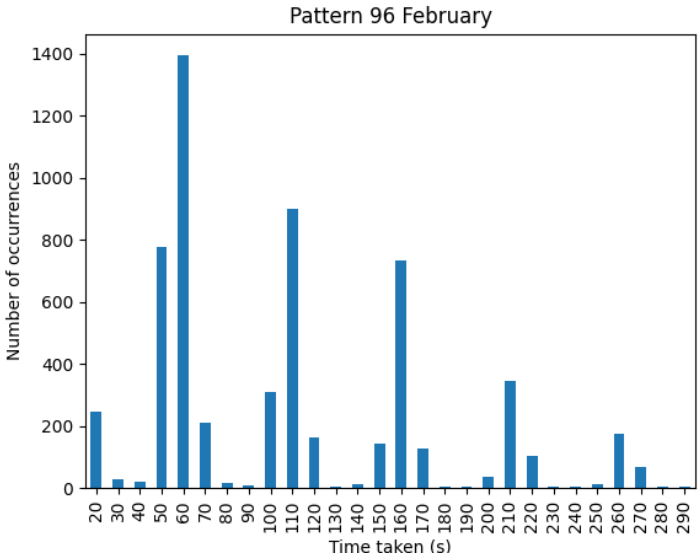


(b) *Pattern 68*

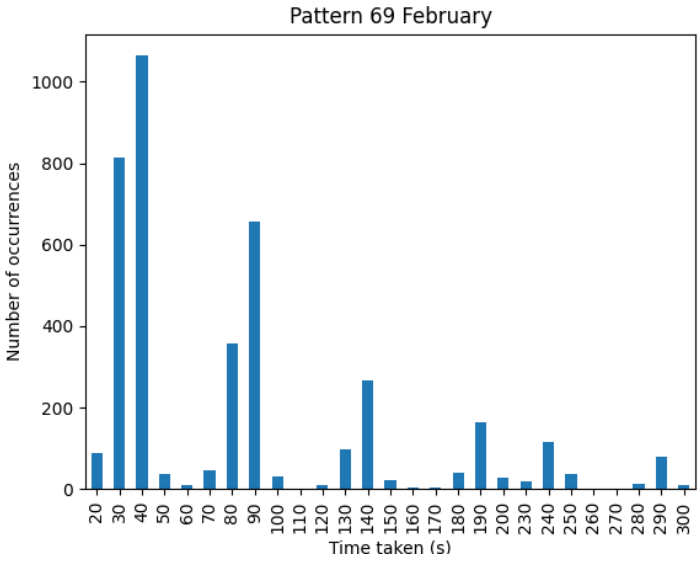
Figure 5.22: Mobility type - Patterns 86 and 68

As we can see from Figure 5.22, the highest peak is about at 20/30 seconds, which corresponds to the mobility of the cars (or motorbikes) as the corresponding average speed is about 63 km/h for about 350 meters.

The other mobility types are also quite evident, as for the previous graphs, 5/6 time peaks are always highlighted.



(a) *Pattern 96*



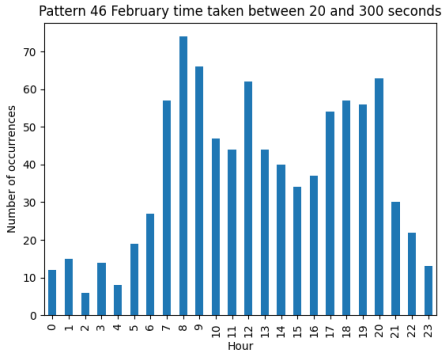
(b) *Pattern 69*

Figure 5.23: Mobility type - Patterns 96 and 69

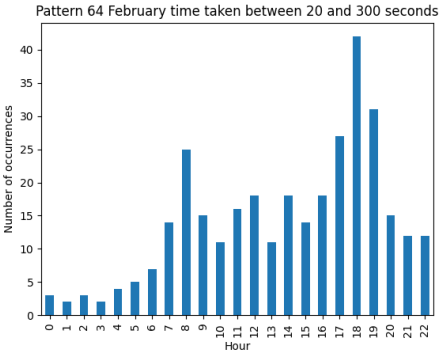
Like previous graphs, in Figure 5.23, there are five peaks, here it can be seen that as the time taken increases, the peaks decrease more and more; this is because the distance between the two scanners is the minimum compared to all the other cases analyzed previously. Also, people's average time on foot is not as long as in the other patterns.

In Figure 5.24, there is another type of visualization of the data; in particular, for four patterns the number of occurrences is shown as the hours of the day change.

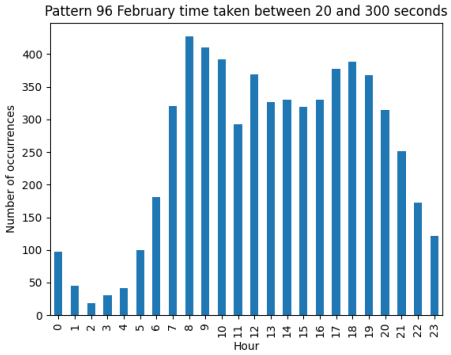
In the creation of the graphs, only the data with a travel time between 20 and 300 seconds were taken into consideration.



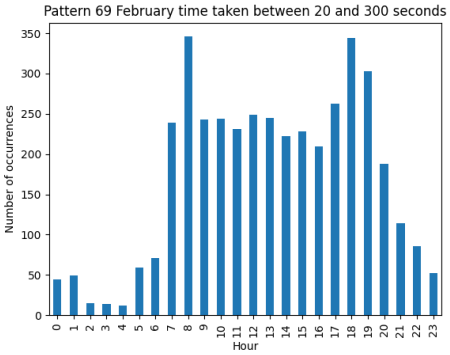
(a) *Pattern 46*



(b) *Pattern 64*



(c) *Pattern 96*



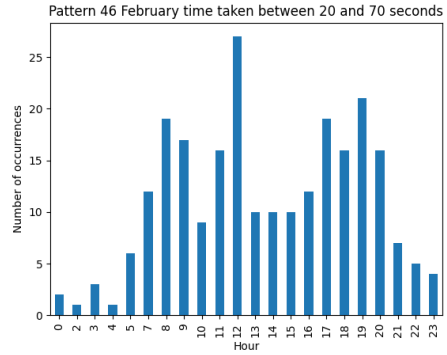
(d) *Pattern 69*

Figure 5.24: Number of occurrences for each hour - All type of mobility

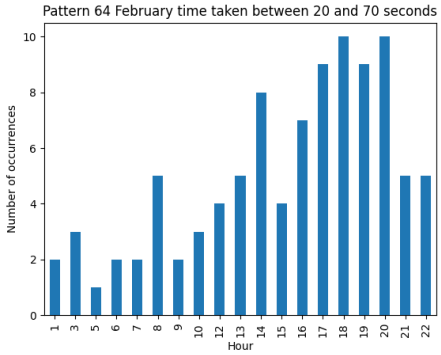
As we can see from the graphs above, the two main peaks are at 8 in the morning and at 6 in the evening, which is when people move to the city to go to work and when they return home in the evening.

It can also be noted that pattern 64 has about half of the detections of the opposite sense, that is pattern 46; while for patterns 96 and 69, the values are approximately the same.

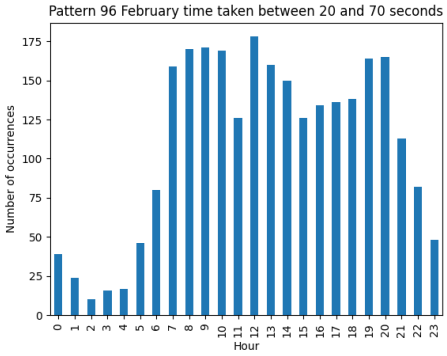
In Figure 5.25, the number of occurrences is analyzed as the hours' change but only for data that have a travel time of the space between the two scanners from 20 to 70 seconds, so focusing the attention on the mobility of cars and motorbikes.



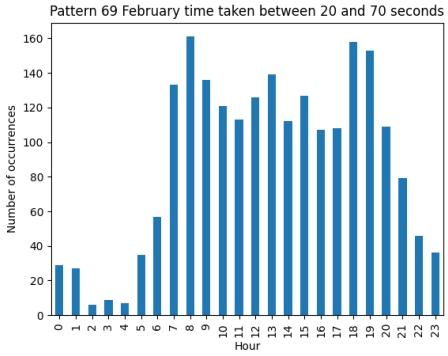
(a) *Pattern 46*



(b) *Pattern 64*



(c) *Pattern 96*



(d) *Pattern 69*

Figure 5.25: Number of occurrences for each hour - Only car and motorbike mobility

It is evident that for patterns 96 and 69, the number of occurrences between 7 and 20 is approximately always the same, while this behaviour is not seen for patterns 46 and 64.

In graph (b), we can see how the number of detections increases during the day until reaching a peak between 18 and 20.

On the other hand, in graph (a), the number of detections does not have a defined trend but peaks scattered throughout the day.

Chapter 6

Real-time metrics

This chapter will list in detail all Grafana features and show all the implemented dashboards.

As said in Section 4.4.3, the real-time dashboards are implemented using Grafana [6] as visualization tool. Grafana main features:

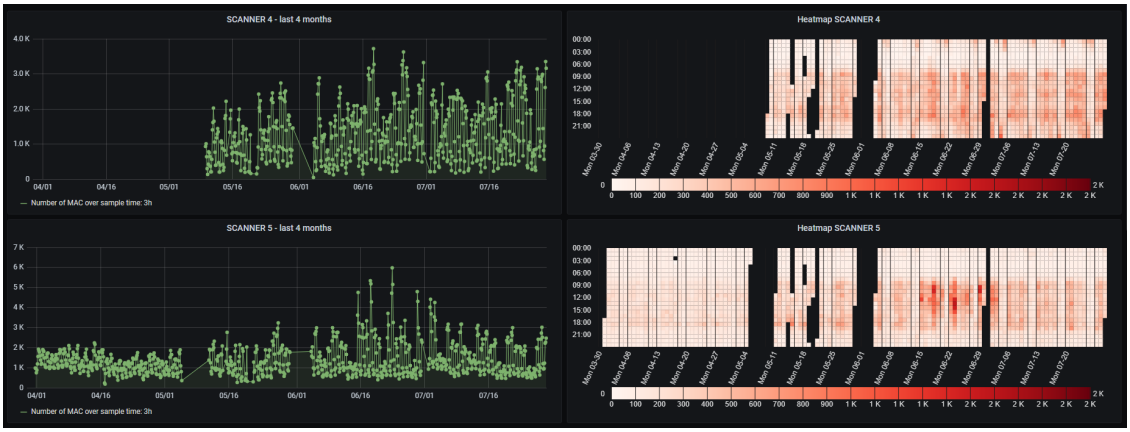
- The dashboards are private and visible only after authentication;
- Easy configuration of data sources, MySQL database in this case;
- Allows adding a new type of graphs by adding plug-ins (self-made or from the official ones);
- The retrieval of data for graphs is done through simple guided SQL queries;
- The graphs are highly customizable (e.g., color scale for heatmaps, range values, labels, size of graphs, display options and others);
- The dashboards are interactive, so it is possible to zoom in a specific time range in the graph or is possible to set a time horizon to see the detailed data in the selected time (e.g., last 5, 15, 30 minutes or last 1, 3, 6, 12, 24 hours or custom date-time interval), see Figure 6.8;
- It is possible to set or modify the refresh rate of each dashboard, from some seconds to minutes or hours; this permits to receive and see real-time data coming from the APs scanners;
- Possibility of setting threshold values above or below witch to generate alerts;
- Allows seeing in a JSON format the retrieved data from the data source.

6.1 Main dashboard

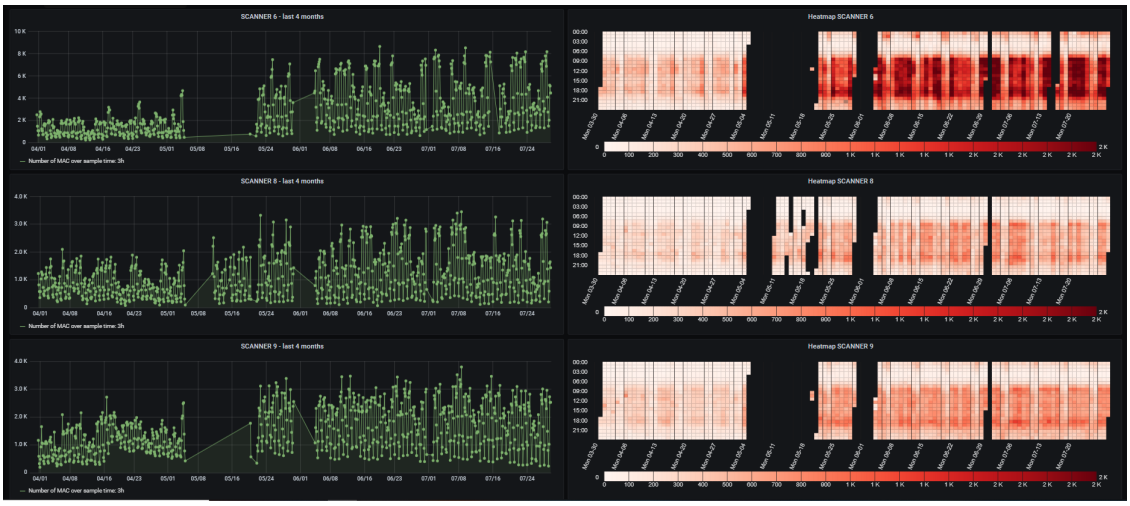
In Figure 6.1, we can see the main dashboard. There are two graphs for each scanner, one line chart and one heatmap.

All the data are obtained through an SQL query, the data with firewall = 2 are discarded (so those in the blacklist), and only the data of the last four months are filtered for a quick view of the last period.

All the heatmaps have the same color scale and the same range of values, between 0 and 2k, so it is very easy to compare the different scanners' graphs.

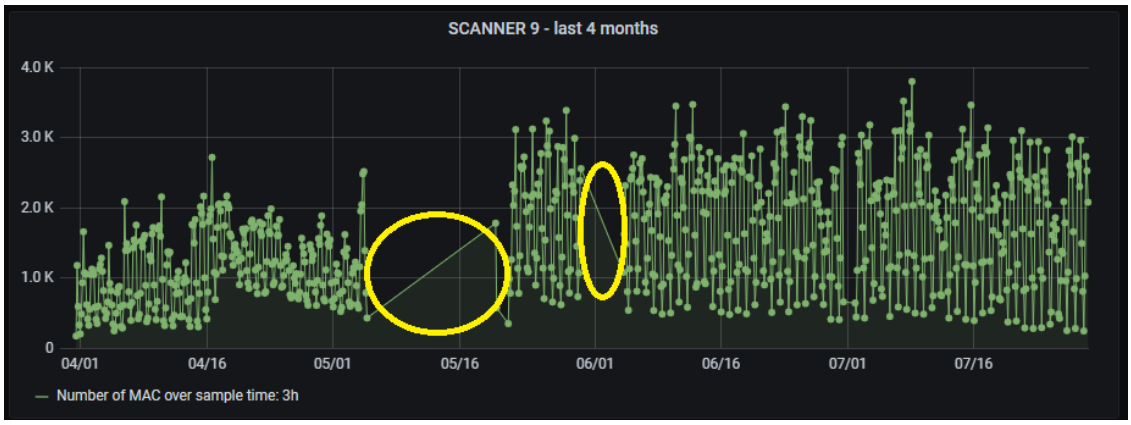


(a) *Scanners 4 and 5*

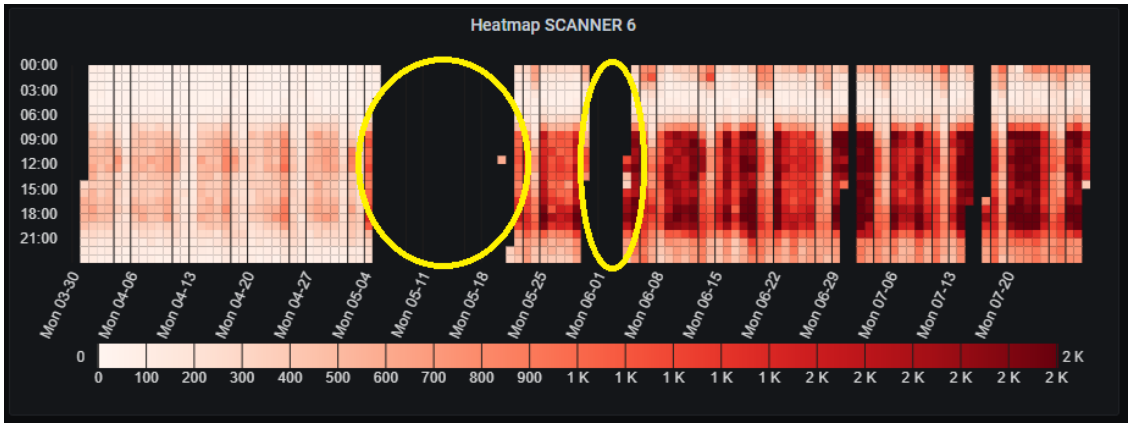


(b) *Scanners 6, 8 and 9*

Figure 6.1: Dashboard ALL - Line charts and heatmaps



(a) *Line chart scanners 9*



(b) *Heatmap scanner 6*

Figure 6.2: Dashboard ALL - Scanners malfunctions detail

In the graphs above (Figure 6.2), some time intervals are highlighted through yellow circles, so it is possible to see the periods where the scanners did not work; therefore no data was detected.

In the heatmap, when there is no data in a period of one hour, it is represented with an uncolored square, i.e., black. On the other hand, in the line chart, we can see a continuous line that connects the last measured value with the next one.

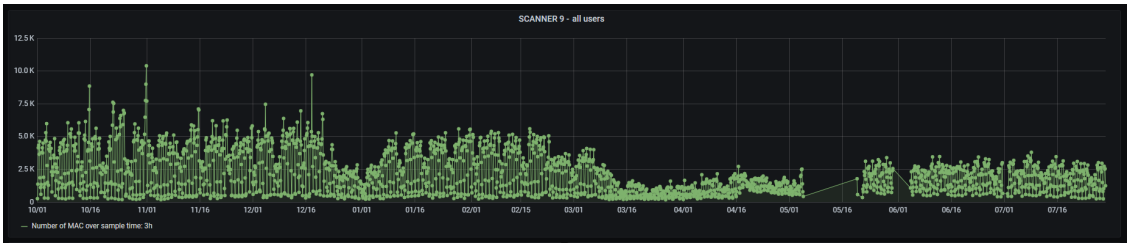
6.2 Detailed scanners dashboards

This section will present the other dashboards created, one for each scanner.

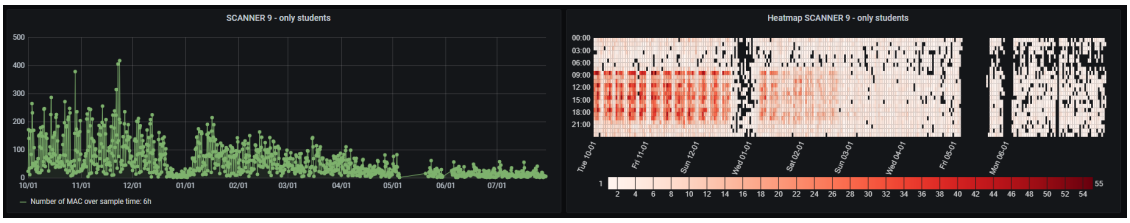
The dashboard of scanner 7 will not be shown as the data present are very few and very dirty because it has been in a TIM laboratory for many months, where some experts are trying to analyze the anomalies of the scanner.

For each dashboard, there are five graphs with all the data captured from the 01/10/2019:

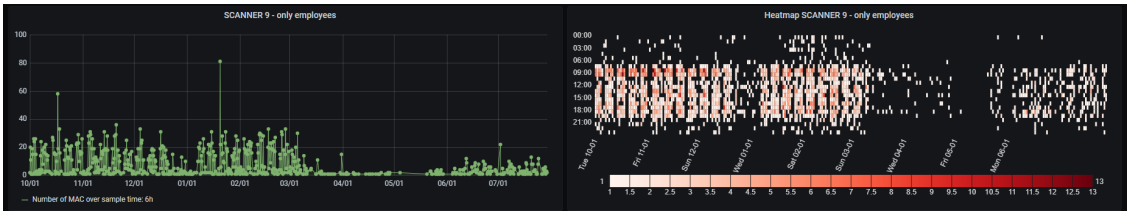
1. line chart with all users data;
2. line chart with only Politecnico employees;
3. line chart with only Politecnico students;
4. heatmap with only Politecnico employees;
5. heatmap with only Politecnico students.



(a) *Line chart scanners 9 - all users*

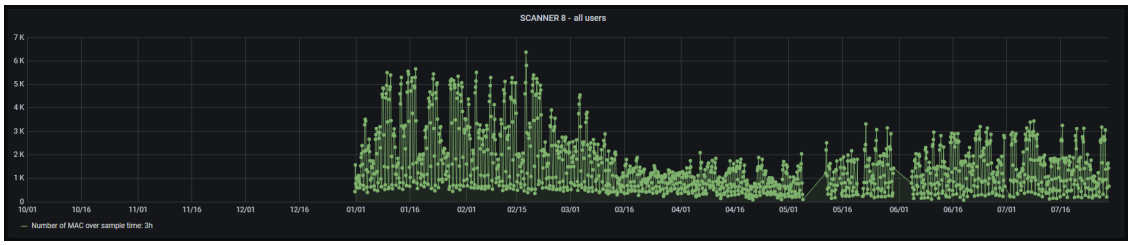


(b) *Line chart and heatmap scanners 9 - only students*

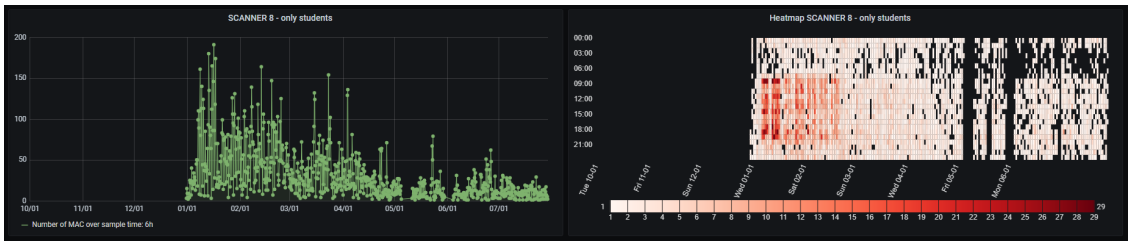


(c) *Line chart and heatmap scanners 9 - only employees*

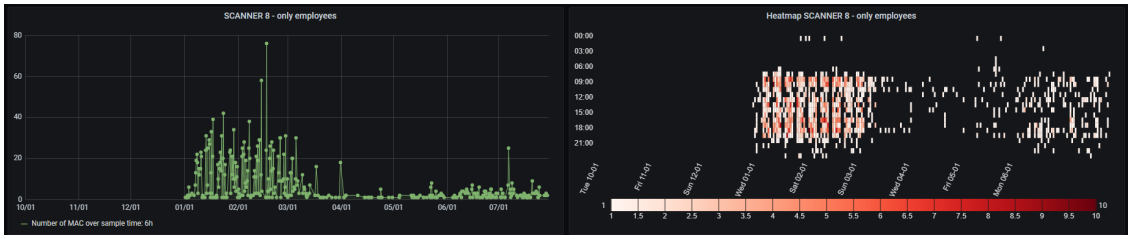
Figure 6.3: Dashboards scanner 9



(a) *Line chart scanners 8 - all users*



(b) *Line chart and heatmap scanners 8 - only students*

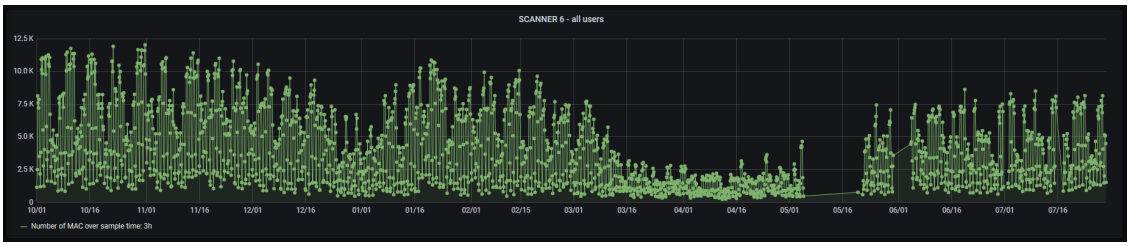


(c) *Line chart and heatmap scanners 8 - only employees*

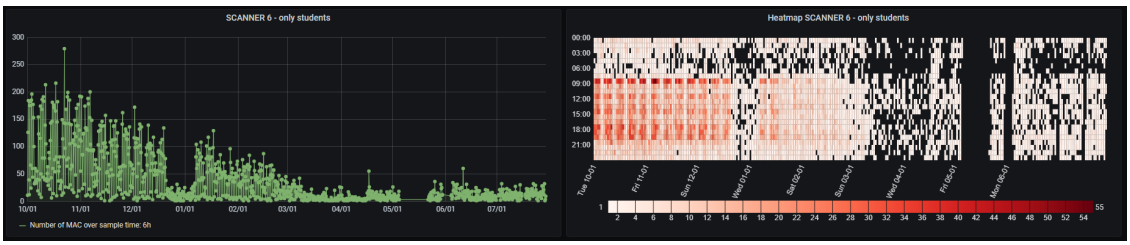
Figure 6.4: Dashboards scanner 8

As we can see, scanner 8 had problems in the first months after installation; indeed, the first detection data dates back to the early days of 2020.

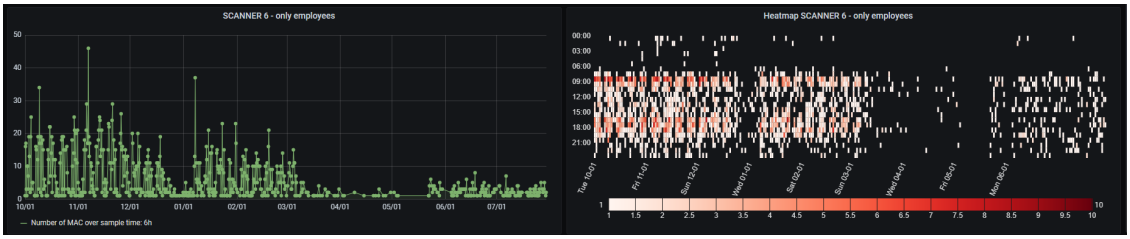
The effect of the lockdown and the subsequent increase of remote work, smart working, is very evident as well as on all the other scanners, except for scanner 6, as seen in Figure 6.5.



(a) *Line chart scanners 6 - all users*



(b) *Line chart and heatmap scanners 6 - Only students*

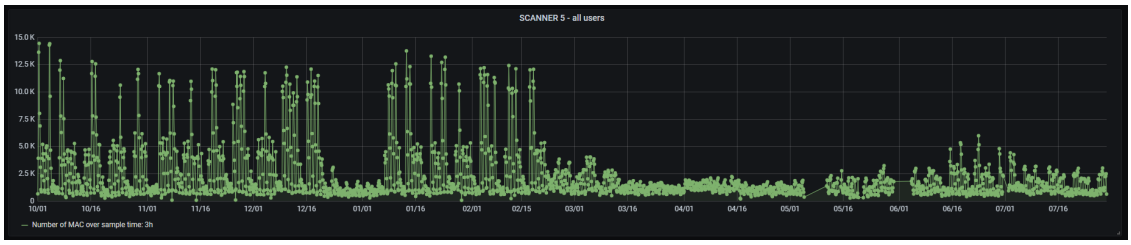


(c) *Line chart and heatmap scanners 6 - Only employees*

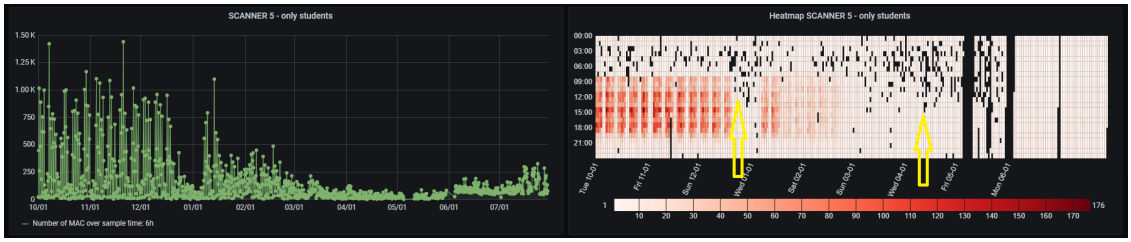
Figure 6.5: Dashboards scanner 6

Comparing the heatmaps of scanners 4, 5, 6 and 9 relating to only the data of Politecnico employees, it can be observed that as we move away from the campus, the data decreases because only a percentage of Politecnico employees are commuter and so goes to the station of Porta Susa to take a train.

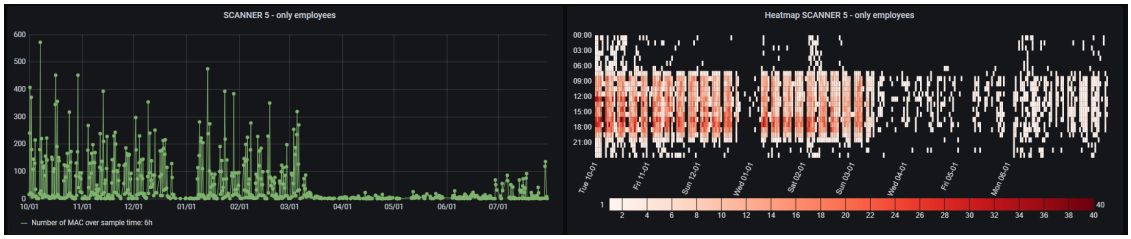
Another thing that can be seen in graph (a) of Figure 6.5 is how the number of detections after the lockdown, from May/June, has risen to near the standards of the months before COVID-19.



(a) *Line chart scanners 5 - all users*



(b) *Line chart and heatmap scanners 5 - only students*

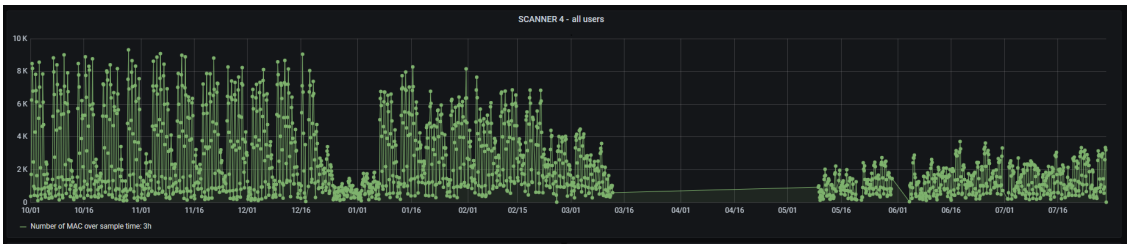


(c) *Line chart and heatmap scanners 5 - only employees*

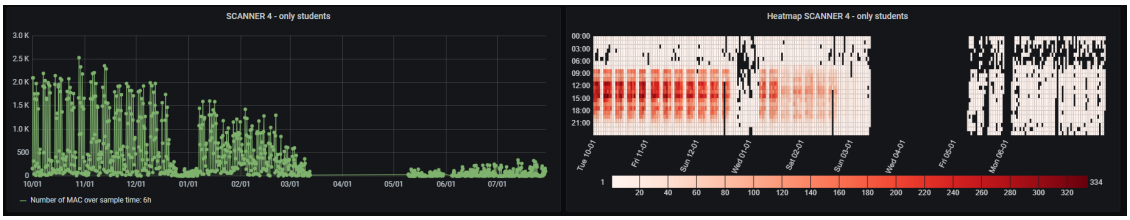
Figure 6.6: Dashboards scanner 5

In graph (b) of Figure 6.6, black squares have been highlighted using yellow arrows; in this case, these black squares do not mean that the scanner worked intermittently but that in those hours, no data was detected as no one was passed near the scanner. On the other hand, when a whole column is without any colored squares, it means that the scanner has had malfunctions that day and did not detect any data.

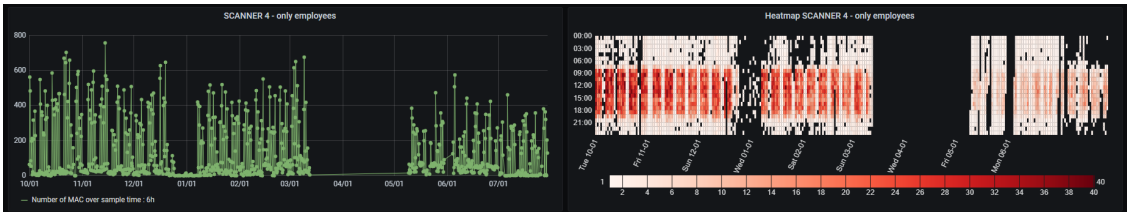
From the line graph (a) of the figure above, the effect that the lockdown caused is very evident; indeed, the trend has gone from peaks from 12.5k devices detected in the months before COVID-19 to about 2.5k detections in the last few months.



(a) *Line chart scanners 4 - all users*



(b) *Line chart and heatmap scanners 4 - Only students*

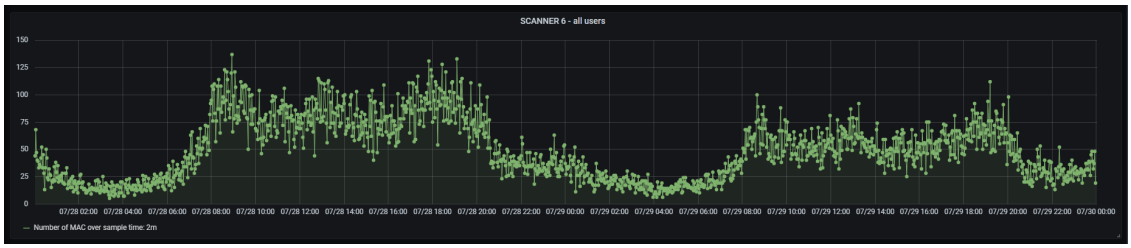


(c) *Line chart and heatmap scanners 4 - Only employees*

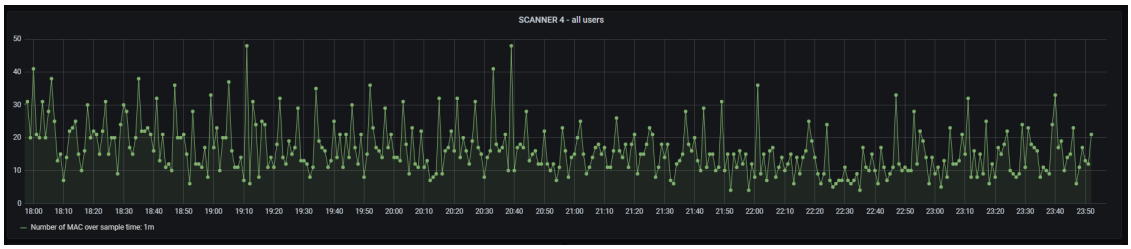
Figure 6.7: Dashboards scanner 4

The dashboard of scanner 4 is undoubtedly the best, together with the one of scanner 5, to analyze data relating to the Politecnico members. The heatmaps are complete and clear, so they allow a quick first analysis.

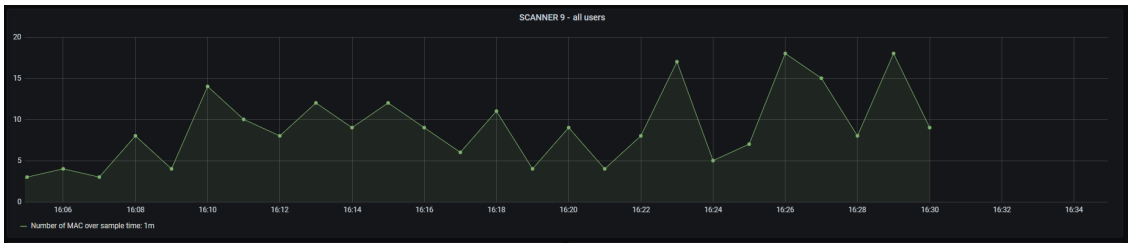
It is very interesting to see in the line graph (c) of Figure 6.7 the trend of the detections of the employees of the Politecnico in the months after the reopening, from May onwards. Compared to all the other trends of the other scanners, here the number of employees who have returned to work in presence in the university is very evident. This trend cannot be seen from the data of scanner 5 because the Politecnico from May has reopened only the entrances on Corso Castelfidardo (near scanner 4) and the one on Corso Duca Degli Abruzzi (main entrance).



(a) *Line chart scanners 6 - 2 days detail*



(b) *Line chart scanners 4 - 6 hours detail*



(c) *Line chart scanners 9 - last 30 minutes detail*

Figure 6.8: Dashboards scanners 4, 6 and 9 - Time interval details

In Figure 6.8, it is possible to see three examples of different data granularity. In particular, in graph (a), the selected time interval is two days, in graph (b) are represented data of 6 hours, and finally, in graph (c) there is a very detailed graph, therefore with very fine granularity (last 30 minutes).

For the first graph, the sample time is 2 minutes, while in the other two it is 1 minute, so each point in the graphs correspond to a single scan.

Chapter 7

Conclusion

Technology today is making great strides forward. With the birth of 5G networks and the emergence of Internet-of-Things devices (IoT devices), the lives of all of us will undergo significant improvements. One of the big problems we are facing, a hot spot for many months (since the beginning of the pandemic), is the monitoring of people flows. First of all, to monitor and avoid gatherings of people and guarantee everyone the right safety, but also, improve life in big cities, improving the movement of people and the public services offered to them.

In Chapter 1, a brief overview was made on the thesis's main objectives for the analysis of people flows and the possible alternative implementations. In this use case, the main role is played by the probe request signals emitted by smart devices owned by passers-by. For this reason, in Chapter 2, these signals and everything concerning them, like the structure of MAC addresses, the use of MAC randomization to increase people's privacy, the use of 5G networks (last mobile network infrastructure born), the use of NFV (Network functions virtualization) and MEC (*Multi-Access Edge Computing*) paradigm, have been analyzed in detail.

The software and hardware architecture described in Chapters 3 and 4 made it possible to create a system capable of detecting WiFi and Bluetooth signals emitted by smart devices and providing quick visual feedback through the dashboards created. Furthermore, thanks to the analysis and classification of each MAC address detected, we can provide detailed graphs and measurements on the flows of the Politecnico members. In this way, we can improve both student life and working life for employees, professors and students, based on the detected real needs.

Despite the continuous progress, by the major smart devices vendors, in the field of user privacy, thanks to the results detected and analyzed, we can say that a good percentage of these devices do not carry out any randomization technique to limit the tracking of people or if used, it is not completely effective.

It is undeniable that the percentage of devices that use the randomized MAC address is not marginal; nevertheless, the goal set at the beginning of this long work has been achieved. As we can see in Chapters 5 and 6, we were able to detect people's main mobility patterns in the area covered by the scanners and also the type of mobility, distinguishing cars or motorcycles from bicycles (and electric scooters) and people moving on foot.

Thanks to the data provided by the Politecnico, we were also able to carry out more in-depth studies on the mobility of Politecnico community members.

From the studies done and shown in the graphs of Chapter 5, and even more from the real-time dashboards, it is possible to see the effect that the lockdown due to COVID-19 has caused in people's everyday travels.

A final consideration should be made on the scanners used for the study. Although they are high-level commercial sensors, they have had numerous reliability problems. Thanks to a script executed every day we could automate the control of their functionality, notifying their operating status via email.

7.1 Future work

This pandemic period has taught us that the study of the flows of people, as well as the counting of people in a given place at a certain moment, will be increasingly fundamental to avoid large overcrowding and gatherings.

This master thesis is intended as a starting point for a much wider project. For sure, the first step would be the expansion of the area under analysis to permit to refine the studies on the detection of the flows of people. An expansion of the area of interest to a much larger portion of the city is already architecturally supported, thanks to the use of the MEC (see Section 2.5.1) paradigm.

Certainly, the work done can also be the starting point for other projects, such as the possibility of counting the number of real-time people in a given place, both outdoors and indoors. If we think at a university like the Politecnico di Torino, it would be very useful to analyze the number of people present in a classroom or in a commonplace (e.g., the canteen, bars, refreshment points or study rooms) to improve students life within the campus. Provide real-time indications to students in which study room go to because it is less crowded or which path to follow to go from one part of the campus to the other, to avoid the numerous traffic jams that take place in the corridors outside the classrooms.

Appendix A

Appendix

A.1 Outliers check on database

```
db = pymysql.connect(host="localhost", user="5Geve", password="
    XXXXXXXX", db="5g_eve_scanner")
with db.cursor() as cursor:
  blacklist = set()
  with open('blacklist.txt', 'r') as fileRead:
    data = fileRead.readlines()
    for line in data:
      blacklist.add(line.rstrip('\n'))

  candidates = dict()
  for i in {4, 5, 6, 7, 8, 9}:
    cursor.execute("SELECT mac, timestamp FROM scanner" + str(i) +
     " WHERE firewall = 1 and timestamp > (NOW() - INTERVAL 1 DAY)")
    myresult = cursor.fetchall()
    for x in myresult:
      mac = x[0]
      if mac not in blacklist:
        if candidates.get(mac):
          candidates[mac].append(x[1])
        else:
          candidates[mac] = [x[1]]

  toadd = set()
  for x in candidates.items():
    hour = dict()
    if len(x[1]) > 15 * 20:
      for entry in x[1]:
        if hour.get(entry.hour):
          hour[entry.hour] += 1
        else:
          hour[entry.hour] = 1
      if sum(value > 15 for value in hour.values()) >= 20:
```

```
31          toadd.add(x[0])
32          blacklist.add(x[0])
33
34    with open('blacklist.txt', 'a') as fileWrite:
35      for entry in toadd:
36        fileWrite.write("{}\n".format(entry))
37
38    for i in {4, 5, 6, 7, 8, 9}:
39      format_strings = ','.join(['%s'] * len(blacklist))
40      cursor.execute("UPDATE scanner" + str(i) + " SET firewall = 2
       WHERE mac IN (%s)" % format_strings, tuple(blacklist))
41
42    for i in {4, 5, 6, 7, 8, 9}:
43      cursor.execute("UPDATE scanner" + str(i) + " SET firewall = 0
       WHERE firewall = 1 and timestamp > (NOW() - INTERVAL 1 DAY)")
44
45    db.commit()
```

In line 1, there is the connection with the database ("5g_eve_scanner") that is on the same NFV where this script runs.

The first operation done is reading the blacklist file where all the outliers MACs are saved (lines 3-7); after that, from line 9 to 19, all MACs captured between NOW() and 1 day before are retrieved from the tables inside the DB. If the MAC retrieved is not in the blacklist, it is added to the list of candidates; this list is a dictionary where there is an array of timestamps as value for each MAC address.

When the dictionary is filled up, each entry is analyzed searching for all MACs with more than 15 timestamps each hour for at least 20 hours in a day; this is the empirical condition for which a MAC is considered as outlier. If a MAC address satisfies the previous condition, it is added to the blacklist.

From lines 34 to 36, the blacklist file is updated with the new entries, if any. Finally, all the database tables are updated, setting the firewall value to 2 if the MAC address is inside the blacklist, 0 vice versa.

Bibliography

[1] *1G, 2G, 3G and 4G*. [Online]. URL: https://www.awaaznation.com/amazing-facts/what-is-1g-2g-3g-4g/.

[2] *48-bit MAC Address Structure*. [Online]. URL: https://www.wikiwand.com/en/MAC_address.

[3] *Bluetooth Special Interest Group. Bluetooth specification version 1.1 and 1.2*. [2001]. URL: http://www.bluetooth.com.

[4] *European 5G validation platform for extensive trials*. [Online]. URL: https://www.5g-eve.eu/.

[5] Julien Freudiger. "How talkative is your mobile device? An experimental study of Wi-Fi probe requests". In: *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. 2015, pp. 1–6.

[6] *Grafana by Grafana Labs*. [Online]. URL: https://grafana.com/.

[7] Hande Hong, Girisha Durrel De Silva, and Mun Choon Chan. "CrowdProbe: Non-Invasive Crowd Monitoring with Wi-Fi Probe". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2.3 (Sept. 2018). DOI: 10.1145/3264925. URL: https://doi.org/10.1145/3264925.

[8] *Huawei MAC address randomization*. [Online]. URL: https://consumer.huawei.com/sa-en/support/content/en-us00754915/#:~:text=For%20your%20privacy%20and%20security,your%20real%20device%20MAC%20address..

[9] *IEEE. OUI Public Listing*. [Online]. URL: http://standards.ieee.org/develop/regauth/oui/oui.txt.

[10] *Kibana by Elastic*. [Online]. URL: https://www.elastic.co/kibana.

[11] *Libelium Meshlium*. [Online]. URL: http://www.libelium.com/products/meshlium/.

[12] *MAC randomization - Developer Android*. [Online]. URL: https://source.android.com/devices/tech/connect/wifi-mac-randomization.

[13] Jeremy Martin, Travis Mayberry, Collin Donahue, Lucas Foppe, Lamont Brown, Chadwick Riggins, Erik C Rye, and Dane Brown. "A study of MAC address randomization in mobile devices and when it fails". In: *Proceedings on Privacy Enhancing Technologies* 2017.4 (2017). DOI: 10.1515/popets-2017-0054, pp. 365–383.

[14] Célestin Matte. "Wi-Fi tracking: Fingerprinting attacks and countermeasures". PhD thesis. 2017.

[15] R. S. Montero, E. Rojas, A. A. Carrillo, and I. M. Llorente. "Extending the Cloud to the Network Edge". In: *Computer* 50.04 (Apr. 2017), pp. 91–95. ISSN: 1558-0814. DOI: 10.1109/MC.2017.118.

[16] *MQTT QoS*. [Online]. URL: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/.

[17] *Networking Spectrum Bands*. [Online]. URL: https://pradeeptechnology5.blogspot.com/2019/01/what-is-5g-and-what-can-we-expect-from.html.

[18] Michele Nitti, Francesca Pinna, Lucia Pintor, Virginia Pilloni, and Benedetto Barabino. "iABACUS: A Wi-Fi-Based Automatic Bus Passenger Counting System". In: *Energies* 13.6 (2020), p. 1446.

[19] Luiz Oliveira, Daniel Schneider, Jano De Souza, and Weiming Shen. "Mobile Device Detection Through WiFi Probe Request Analysis". In: *IEEE Access* 7 (2019). DOI: 10.1109/ACCESS.2019.2925406, pp. 98579–98588.

[20] *OneM2M*. [Online]. URL: https://www.onem2m.org/.

[21] *Paho MQTT Python client*. [Online]. URL: https://pypi.org/project/paho-mqtt/.

[22] *Probe request - Developer Android*. [Online]. URL: https://developer.android.com/guide/topics/connectivity/wifi-scan.

[23] *Probe Request frame*. [Online]. URL: https://www.oreilly.com/library/view/80211-wireless-networks/0596100523/ch04.html.

[24] *Pygame*. [Online]. URL: https://www.pygame.org/.

[25] *Relational VS Non-Relational Databases*. [Online]. URL: https://medium.com/@zhenwu93/relational-vs-non-relational-databases-8336870da8bc.

[26] J. Ren, G. Yu, Y. He, and G. Y. Li. "Collaborative Cloud and Edge Computing for Latency Minimization". In: *IEEE Transactions on Vehicular Technology* 68.5 (2019), pp. 5031–5044.

[27] N. K. Shankaranarayanan and A. Ghosh. "5G". In: *IEEE Internet Computing* 21.5 (2017), pp. 8–10.

[28] *Ubuntu distribution*. [Online]. URL: https://www.ubuntu-it.org/.

[29] *YOLOv3*. [Online]. URL: https://pjreddie.com/darknet/yolo/.