Grau en Enginyeria Informàtica

WI-FI TRACKING SYSTEM AND ANALYSIS

Memòria

Autor: Jordi Aballó

Ponent: Léonard Janer

Director: Òscar Jofre

Primavera 2017

Acknowledgements

I wish to express my sincere thanks to Oscar Jofre, the director of this project, to give me the opportunity of develop this system and being always available.

I would like to thank my tutor Léonard Janer for the guidance since the very first day and all the people involved directly or indirectly to this project.

I take this opportunity to express also gratitude to my family for believing in me and for the support they always showed.

Finally, I would like to thank my friends and university colleagues.

Abstract

The aim purpose of this project is to create a system through Wi-Fi antennas to detect devices within a perimeter. This allows us to know very valuable information such as which devices are located both inside and outside the perimeter, the distance of each of them with the antenna, stay times, among other data that, after being processed and analyzed, is shown as charts in real-time.

Resum

L'objectiu d'aquest projecte es crear un sistema que a través d'antenes Wi-Fi detectin dispositius dins d'un perímetre. Això, ens permetrà saber informació molt valuosa tal com conèixer quins dispositius es troben tant l'interior com a l'exterior del perímetre, a quina distància es troba cadascun d'ells de l'antena, temps d'estada, entre d'altres dades, que després de ser processades i analitzades, es mostren en forma de gràfics i ens temps real.

Resumen

El objetivo de este proyecto es crear un sistema que a través de antenas Wi-Fi, detecten dispositivos dentro de un perímetro. Esto nos permitirá conocer información muy valiosa como, por ejemplo, qué dispositivos se encuentran tanto al interior como al exterior del perímetro, la distancia de cada uno de ellos con la antena, tiempos de estada, entre otros datos, que después de ser procesados y analizados, se muestran en forma de gráficos y en tiempo real.

Table of contents

1. Introduction 1

1.1. Scope 1

1.2. Objectives 2

1.3. Risks 2

1.4. Workflow 3

2. Technologies research 5

2.1. Data-processing pipeline 5

2.1.1. Logstash 5

2.1.2. Filebeat 7

2.1.3. Logagent 8

2.2 Non-relational database 9

2.2.1. MongoDB 10

2.2.2. Elasticsearch 11

2.3. Data visualization software 11

2.3.1. SumoLogic 12

2.3.2. Loggly 12

2.3.3. Kibana 13

2.4. Access Points 13

2.4.1. Mikrotik and Open-Mesh differences 15

2.4.2. Winbox vs Cloudtrax 15

2.4.3. Models 15

2.5. Server 20

2.6. Conclusion *20*

3. System process **23**

3.1. Capturing data *24*

3.1.1. Mikrotik 24

3.1.2. Open-Mesh 26

3.2. Process data *28*

3.2.1. Input 29

3.2.2. Filters 30

3.2.3. Output 31

3.3. Visualization *31*

4. Data processing **33**

4.1. Initial data *33*

4.1.1. Open-Mesh 33

4.1.2. Mikrotik 35

4.2. Building a readable structure *36*

4.2.1. Sample structure 38

4.3. Calculating distance between device and AP *39*

4.4. Additional data *40*

4.4.1. OUI (Organizationally Unique Identifier) 40

4.4.2. Client 41

4.4.3. Range 42

4.5. Final data structure *42*

5. Analysis **45**

5.1. Devices recognized *45*

5.1.1. OUI Organization distribution 50

5.1.2. Insiders *vs* Outsiders 51

5.2. Distance *52*

6. Real scenario **55**

6.1. Metrics *56*

6.2. Kibana dashboard *60*

7. Conclusion **63**

Annex A. Installation & Configuration **65**

A.1 Prerequisites *65*

A.2 ElasticSearch *65*

A.3 Kibana *67*

A.4 NGINX *68*

Bibliography **73**

List of figures

Figure 1. System's workflow 3

Figure 2. Logstash typical use-case 6

Figure 3. Logstash typical use-case with centralized server 7

Figure 4. Filebeat's workflow 7

Figure 5. Mikrotik RB951Ui-2HnD 15

Figure 6. Mikrotik wAP ac 17

Figure 7. Open-Mesh OM2P-V2 19

Figure 8. System process overview 23

Figure 9. Graphical representation of 2.4 GHz band channels overlapping 40

Figure 10. Office map and distribution 45

Figure 11. AP01 insiders' area 48

Figure 12. OM and AP02 insiders' area 48

Figure 13. Triangulation of a device 49

Figure 14. Counting devices comparative 49

Figure 15. OUI Organization distribution 50

Figure 16. Graphical representation of outsiders and insiders 51

Figure 17. APs and tracked device distribution 52

Figure 18. Distance analysis 52

Figure 19. Distance analysis after SNR filter 53

Figure 20. Customer store distribution 55

Figure 21. Visitors distribution by hours 57

Figure 22. Storefront visitors distribution 57

Figure 23. Device organizations distribution 58

Figure 24. Visitors distribution for the last 7 days 58

Figure 25. Number of devices in a given radius distance 59

Figure 26. Snapshot 1 of customer dashboard 60

Figure 27. Snapshot 2 of customer dashboard 61

Figure 28. Snapshot 3 of customer dashboard 61

List of tables

Table 1. Mikrotik RB951Ui-2HnD wireless specifications at 2.4 GHz 16

Table 2. Mikrotik RB951Ui-2HnD device specifications 16

Table 3. Mikrotik wAP ac wireless specifications at 2.4 GHz 17

Table 4. Mikrotik wAP ac wireless specifications at 5 GHz 18

Table 5. Mikrotik wAP ac device specifications 18

Table 6. Open-Mesh OM2P-V2 device specifications 19

Table 7. Server details 20

Table 8. Magnitude order estimation of Mikrotik captured sequences 25

Table 9. Magnitude order estimation of Open-Mesh captured sequences 27

Table 10. Initial structure of Open-Mesh data stored into Elasticsearch 34

Table 11. Initial structure of Mikrotik data stored into ElasticSearch 35

Table 12. Open-Mesh structure of data after applying initial filters 38

Table 13. Mikrotik structure of data after applying initial filters 39

Table 14. Mac-OUI example 41

Table 15. Final structure of data 43

Table 16. Detected devices comparative between different APs 46

Table 17. Nearby devices comparative filtered by frequency source and network source 47

Table 18. Total count of devices. 50

Table 19. Top-5 OUI organization AP01 vs OM 50

Table 20. Insider vs outsider comparison 51

Table 21. Real distance between tracked device and APs 52

Table 22. Top-5 Device organization 58

1. Introduction

Today, more and more companies need to create new strategies to meet consumer behavior. This project provides a solution in real time to know the movements and behavior of a company's customers.

This solution is based on detecting devices inside the establishment and therefore knows how many people are inside, how many people pass by the entrance without falling them inside, what movements do customers inside the establishment and how long they are until they leave among other useful information that will be detailed later.

This project is developed jointly with SocialWifi S.L. *[1]*, a company that provides social Wi-Fi to different establishments.

This project along with the company's current solution, make a very powerful tool. It will detect nearby devices and transform presence data to KPIs[1], but also, people who access through social Wi-Fi are identified and we can get these KPIs more segmented as we can get to know more specific details, such as age, gender, interests, nationality, among others.

1.1. Scope

The scope of this system is mainly aimed at stores that need to know customer behavior and create different marketing strategies Retail sector, shopping centers and places where events are held would be the major target. Despite of it could be implemented in any crowded space.

It can be either used temporarily for smaller shops in order to know customer's behavior and then, create different marketing strategies, or permanently for bigger spaces such as local public areas, malls, or others.

1

[1]KPI (Key Performance Indicator): is a measurable value that demonstrates how effectively a company is achieving key business objectives.

1.2. Objectives

The motivation that leads me to do this project is to create an innovative system with these features with almost non-existent competitors.

The main objective is to ensure that the system is able to provide the most accurate and reliable data. Furthermore, it aims to be a *plug & play* solution and the minimum installation process if proceed. It is a difficult challenge because it involves several factors plus I do not know all the tools that I will work with both hardware and software.

The main KPIs we want to achieve at the end of this project are:

- Number of establishment visitors.
- Number of visitors who has stopped at storefront.
- Rate between insiders and outsiders.
- Most frequented time inside the establishment.
- Most frequented time at storefront.
- Most used smartphone by visitors.
- Most affluence day of potential buyers.
- Top 10 visitors.
- Store's hot zones.
- Visitors return rate.

1.3. Risks

This system comprises all devices in a range of an Access Point (AP). One drawback is that the device must have Wi-Fi turned on in order to be detected. However, it is not necessary for device to be connected in any network.

Another important factor is the distance we get between device and AP might not be 100% accurate as there will probably be interferences caused by other devices. We accomplished to obtain a margin error to a minimum of half a meter and a maximum of three meters using a mathematic formula.

1.4. Workflow

The process of this system starts by installing a number of Wi-Fi APs capable to detect devices within different frequencies and retrieve information. This utility is called *snooper [2]* and not all APs have this feature.

Once this information is captured, antenna sends it automatically to the server and data start to be processed. These data are filtered to discard devices we do not want to count, such as printers, routers, or other APs. What we mainly want to count are smart devices.

After this first filtering, a data-cleaning process begins because there are a lot of useless and unnecessary information. At this point, data is now more appropriate and we can treat and transform them to obtain new relevant information such as signal strength, distance, time that has been captured, among others. Subsequently it is stored to a non-relational database.

Finally, data is ready to be displayed in a proper manner, and so, it's shown like graphs, which generate statistics, KPIs and very useful information.

Below you will see at **Figure 1** the workflow of this system:

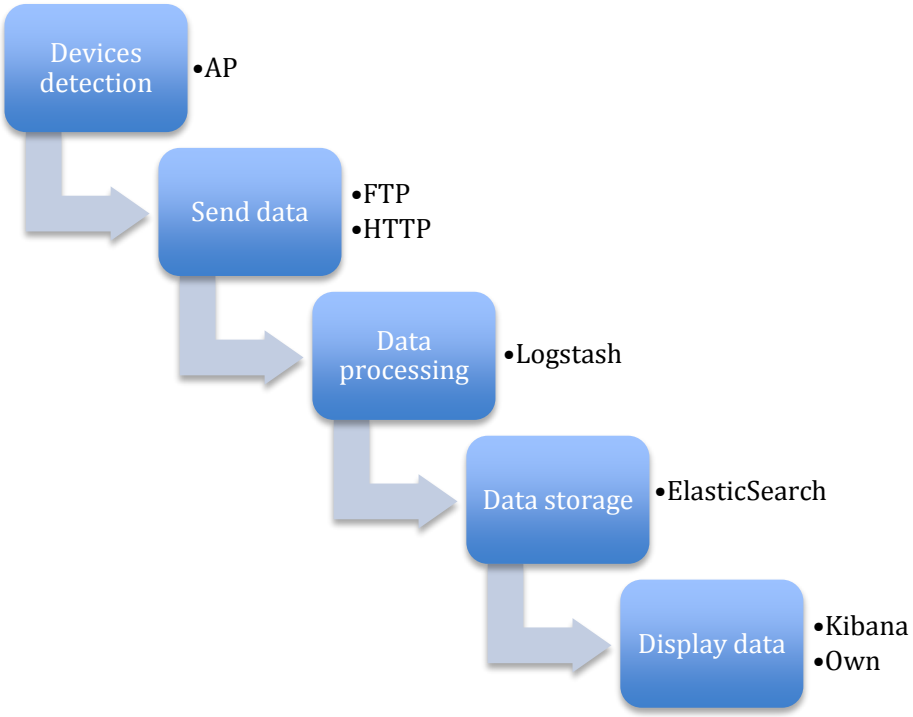


Figure 1. System's workflow

2. Technologies research

One of the project's principles, as mentioned previously, is to achieve this solution being low-cost and plug & play. In order to accomplish that, we will choose between the most competitive open-source technologies.

System main technologies:

- Data-processing pipeline
- Non-relational database
- Data visualization software
- APs
- Server

These technologies must provide the most reliable and accurate information. Below, there is a comparative of different software that solves these technologies.

2.1. Data-processing pipeline

Data-processing pipeline is a server-side technology responsible to ingest captured data from multiple sources to a configured stash. During the process, this technology will clean and filter all data. Then, it will be formatted in a specific manner for later on store into the database.

I found out that Logstash *[3]*, Filebeat *[4]* and Logagent *[5]* are the most used and reliable software to process and manage large amounts of data.

2.1.1. Logstash

Logstash is an open source tool for collecting and managing log files. Basically, you can take pretty much any kind of data, enrich it as you wish, and then push it to lots of destinations. It's a part of an open-source stack that includes Elasticsearch *[6]* for indexing and searching through data and, Kibana *[7]*, for charting and visualizing data.

Strengths

- Flexibility, due to the number of plugins.
- Clear documentation
- Straightforward configuration
- A very good tool if you do combine it with Elastic Stack

Weaknesses

- Extensibility
- Not buffering
- Performance and resource consumption (default heap size 1GB)

Typical use-cases

Logstash is a great tool for prototyping, especially for more complex parsing. If you have big servers, you might as well install Logstash on each. You won't need buffering if you're tailing files, because the file itself can act as a buffer (i.e. Logstash remembers where it left off, see *Figure 2*):

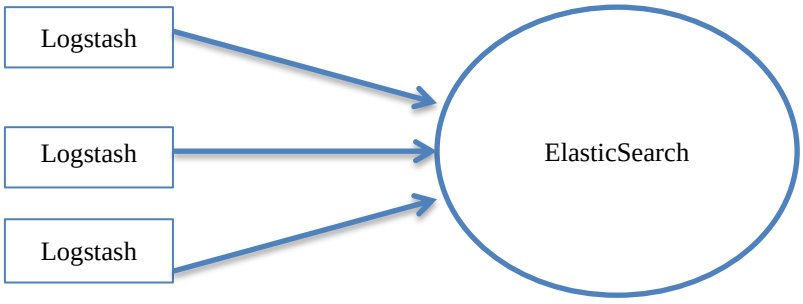


Figure 2. Logstash typical use-case

If you have small servers, installing Logstash on each is a no go, so you'll need a lightweight log shipper on them that could push data to Elasticsearch through one (or more) central Logstash servers, see

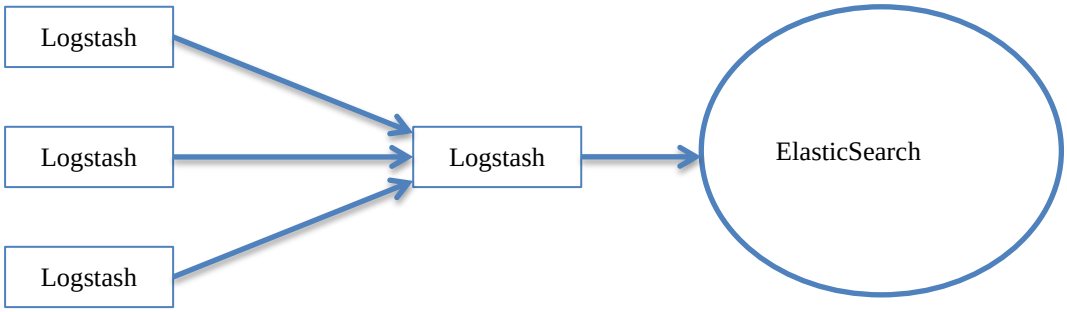


Figure **3**:

Figure 3. Logstash typical use-case with centralized server

2.1.2. Filebeat

Filebeat is a lightweight log shipper that came to life precisely to address the weakness of Logstash: Filebeat was made to be that lightweight log shipper that pushes to Logstash.

With version 5.x, Elasticsearch has some parsing capabilities (like Logstash filters) called ingest. This means you can push directly from Filebeat to Elasticsearch, and have Elasticsearch do both parsing and storing. You should not need a buffer when tailing files because, just as Logstash, Filebeat remembers where it left off, see *Figure 4*:

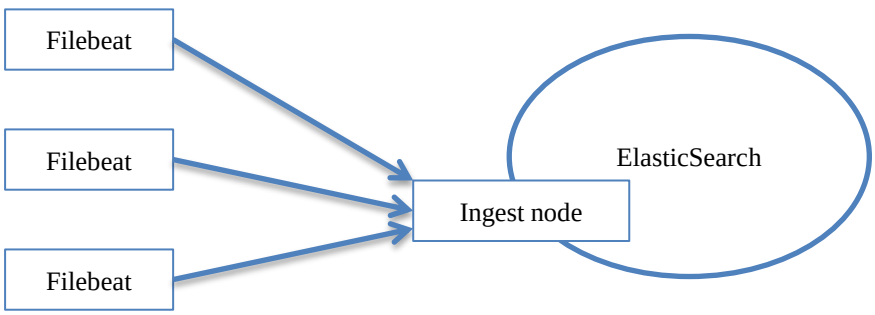


Figure 4. Filebeat's workflow

Strengths

- Low resources and consumption
- Reliable
- Fast

Weaknesses

- Very limited scope
- Poor documentation and online community

Typical use-cases

Filebeat is great for solving a specific problem: you log to files, and you want to either:

- Ship directly to Elasticsearch. This works if you want to just *grep*[1] them or if you log in JSON
- Put them in Kafka/Redis[2] *[8][9]*, so another shipper (e.g. Logstash, or a custom Kafka consumer) can do the enriching and shipping.

2.1.3. Logagent

Logagent is a modern, open-source, lightweight log shipper with out of the box and extensible log parsing, on-disk buffering, secure transport and bulk indexing to Elasticsearch and Logsene[3] *[10]*. It can be just as easily used to push data to Elasticsearch.

Strengths

- Extra-easy to work

[1] Grep: Linux command-line utility for searching plain-text data sets for lines matching a regular expression.

[2] Kafka/Redis buffering workaround

[3] Logsene is a cloud solution based on ElasticSearch Stack

- GeoIP enriching based on IPs
- Light and fast
- Compatible with node.js
- Local buffering

Weaknesses

- Young
- Not very flexible

Typical use-cases

Logagent is a good choice of a shipper that can do everything (tail, parse, buffer and ship) that you can install on each logging server. Logagent is embedded in *Sematext Docker Agent [11]* to parse and ship *Docker* containers logs.

2.2 Non-relational database

A non-relational database (NoSQL) is fundamental to build this system. Regarding to this project, NoSQL integrates everything much better than a traditional relational database (SQL), mainly because of the mass amount of data that has to be stored. NoSQL provides all the features needed to build a content-rich app. It can incorporate any type of data and it is very friendly to scalability. Besides it is faster and more productive, according to MongoDB, the cost in terms of hardware and productivity is just 10% of a relational database *[12]*.

After researching and comparing among non-relational databases, MongoDB *[13]*, Cassandra *[14]* and Elasticsearch are the most known and used and its documentation is way the most complete. This is an important factor because it will be needed to configure the database and just in case you become stalled at some point. Also, the fact that you do not need to purchase for a license is a plus.

I discarded Cassandra because all data-processing pipeline tools mentioned before works better either with MongoDB or Elasticsearch and their documentation refer all the time to Elasticsearch.

2.2.1. MongoDB

MongoDB is the most popular NoSQL database management system (DBMS), - according to *db-engines [15]* – open-source cross-platform document-oriented database program coded in C++. It has a very flexible approach in how it stores data and doesn't have any schema.

MongoDB has the ability to deal with:

- High volumes of data
- Horizontal Scalability[1] with sharding[2]
- MapReduce[3] system
- Stemming[4]
- BSON (Binary + JSON) documents

Can't deal with:

- Own API REST interface
- Find related documents
- ACID[5] transactions
- Deep search

[1] Horizontal scalability: unlike the vertical scalability (which can be expensive) this solution offers to split the load among several servers.

[2] Sharding: to split among several servers, pieces of index to divide the load of the disk. This also allows splitting the load of a service because there is at least one service per server so they are less in demand as they run in parallel.

[3] MapReduce: MapReduce is an algorithm allowing to process a large amount of data shared on several servers by aggregating data.

[4] Stemming: i.e.: retrieve a document with the word "vaccination" when you are looking for "vaccine"

[5] ACID: Atomicity, Consistency, Isolation, Durability

2.2.2. Elasticsearch

Elasticsearch is a document-oriented search engine programmed in Java using Lucene *[16]*. Created in 2012, it is becoming more and more popular with a growing community. It is free, open source but there are plugins and tools that you have to pay for.

Elasticsearch should not be used as a main database system. It's a search engine and not a database.

Elasticsearch has the ability to deal with:

- Complex queries
- Horizontal scalability
- Intelligent management of shards
- No SPOF[1]
- Domain-specific language[2]
- Custom indexing rules

Elasticsearch can't deal with:

- Reindexing takes some time
- ACID transactions

2.3. Data visualization software

This software is necessary to display all stored data. Without it, it would be impossible to interpret all the information because there are millions of entries to analyze and display. It is responsible to convert all data to useful information and will show you all

[1] SPOF (No Single Point Of Failure) part of a system that if fails, will stop the entire system from working. SPOFs are undesirable in any system with a goal of high availability or reliability.

[2] It has his own DSL (Domain-specific language) based on a JSON format enabling to make some queries through REST API more easily than a SQL query.

the analyzed information in a simple way so you can draw conclusions quickly. Mainly, one this software's feature is that most of the results are expressed in charts.

Some tools that will let us interpret the information showing up graphs and other views from analyzed data are, between others, SumoLogic *[17]*, Kibana and Loggly *[18]*.

All these tools have the common issue that there might be a lag between the time data is logged and the time it's visible to the service (data visualization software).

2.3.1. SumoLogic

SumoLogic's unique new cloud-based dashboards streaming query engine, allows to troubleshoot, monitor, and extract operational and business insights from large amounts of data from the entirety of an IT infrastructure.

Pros

- Search and chart mass amounts of data
- SaaS
- Easy setup
- Ability to establish baselines and to actively notify when key metrics change after an event

Cons

- Additional overhead on machines depending on logging throughput
- Price

2.3.2. Loggly

Loggly is also a robust log analyzer, focusing on simplicity and ease of use for DevOps[1] *[19]* audience.

[1] DevOps: (a clipped compound of "software Development" and "information technology operations") is a term used to refer a set of practices that emphasize the collaboration and communication both software developers and information technology professionals while automating the process of software delivery infrastructure changes

This is a tool mainly for DevOps to parse data coming from your app servers. Anything beyond that, you'll have to build yourself.

Pros

- Developer-friendly (finds and fixes operational problems)
- Custom performance
- Transparent pricing

Cons

- Can't scale into a full-blown infrastructure, security or analytics solution.

2.3.3. Kibana

Kibana is an open-source data visualization plugin for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster.

Pros

- Aggregation capabilities
- Allows to manage and monitoring Elasticsearch Database
- ELK pack
- Multiple charts (histograms, line graphs, pie charts, sunbursts, and more.

Cons

- Specifically designed to use Elasticsearch API
- They do not return compatible data structures rather than JSON responses.

2.4. Access Points

Access Points are fundamental for this project. They will detect all nearby devices with its Wi-Fi activated. These APs will capture information from detected devices and then will push it to data-process pipeline.

There are some terms that should be understood in order to comprehend the behavior of antennas:

dBm or *decibel-milliwatt*

Is an abbreviation for the power ratio in decibels (dB) of the measured power referenced to one milliwatt (mW).

The power in dBm ($P_{(dBm)}$) is equal to 10 times base 10 logarithm of the power in milliwatts ($P_{(mW)}$).

$$P_{(dBm)} = 10 \cdot log10(P_{(mW)} / 1mW)$$

1 milliwatt is equal to 0 dBm

$$1mW = 0dBm$$

1 watt is equal to 30dBm

$$1W = 1000mW = 30dBm$$

Transmit power and receive sensitivity

These indicators allow us to know the signal strength. The transmit power is the lowest measure and indicates that the device is very close to the AP, while the receive sensitivity indicates the highest measure that the node can capture from the device, i.e.: -20 dBm (very close to the AP) and -96 dBm (very away from the AP)

PoE

Power Over Ethernet. This is a solution for powering devices using Ethernet cables. It passes the electric power along with data.

dBi

The forward gain of an antenna compared with the hypothetical isotropic antenna, which uniformly distributes energy in all directions.

The APs we will work with will be either model *RB951Ui-2HnD [20]* or *wAP ac [21]* from Mikrotik, though we will test Open-Mesh OM2P-V2 *[22]* AP to compare results.

2.4.1. Mikrotik and Open-Mesh differences

An important difference between Mikrotik and Open-Mesh APs is that the second ones have an embedded script that discards network devices, such as routers and switches whereas Mikrotik AP does not have this script and we will have to analyze the information retrieved to discard manually these specific devices.

Another difference between them is that Mikrotik can only use chains whether listening devices or providing Internet access while Open-Mesh can do both simultaneously.

2.4.2. Winbox vs Cloudtrax

Winbox *[23]* is a small utility that allows administration of all features for Mikrotik devices and Cloudtrax *[24]* works for Open-Mesh devices but it is much more limited in terms of configuration. Mikrotik have way more features than Open-Mesh.

You will specify in these utilities where will data be sent and enable the snooper to capture data.

2.4.3. Models

Mikrotik RB951Ui-2HnD

Figure 5. Mikrotik RB951Ui-2HnD

The RB951Ui-2HnD *(See Figure 5)* is a wireless AP with a new generation Atheros CPU and more processing power. It has five Ethernet ports, one USB 2.0 port and a high power 2.4GHz 1000mW 802.11b/g/n wireless AP with two antennas built in. It has a 600MHz CPU, 128MB of RAM and PoE output function for port #5.

Next tables (*Table 1*, *Table 2*) show detailed technical information about wireless and AP specifications.

Wireless Specifications

	Transmit power (dBm)	Receive sensibility (dBm)
6 Mbit/s	30	-96
54 Mbit/s	25	-80
MCS0	30	-96
MCS7	23	-78

Table 1. Mikrotik RB951Ui-2HnD wireless specifications at 2.4 GHz

Device specifications

Details	
CPU nominal frequency	600 MHz
CPU core count	1
RAM	128MB
10/100 Ethernet ports	5
USB ports	1
Wireless Standards	802.11 b/g/n
PoE in / out	Yes / Yes
Input voltage	7V – 31V
Dimensions	113x138x29 mm
Antenna gain DBI	2.5
Max power consumption	7W
Number of chains	2
Storage	128 MB NAND

Table 2. Mikrotik RB951Ui-2HnD device specifications

Price

Currently price 56,00 €.

Mikrotik wAP ac

Figure 6. Mikrotik wAP ac

Mikrotik wAP ac *(Figure 6)* has one Gigabit Ethernet port; it supports 802.11ac technology and can work at both the 2.4GHz and 5GHz frequencies simultaneously.

The wAP ac is weatherproof and can be fixed to any external wall from the inside of the case - so that it is securely attached to its mounting location.

This AP has three chains for 5GHz frequencies and two chains for 2.4 GHz frequencies.

Below, there are two different tables (*Table 3*, *Table 4*) with wireless technical information for 2.4 GHz and 5 GHz.

Wireless Specifications at 2.4 GHz

	Transmit power (dBm)	**Receive sensibility (dBm)**
1 Mbit/s	25	-95
11 Mbit/s	25	-90
6 Mbit /s	25	-95
54 Mbit/s	25	-80
MCS0	25	-95
MCS7	22	-74

Table 3. Mikrotik wAP ac wireless specifications at 2.4 GHz

Wireless Specifications at 5 GHz

	Transmit power (dBm)	Receive sensibility (dBm)
6 Mbit/s	25	-96
54 Mbit/s	25	-81
MCS0	25	-96
MCS7	24	-77
MCS9	23	-72

Table 4. Mikrotik wAP ac wireless specifications at 5 GHz

Device specifications

The next table (**Table 5. Mikrotik wAP ac device specificationsTable 5**) shows device's technical information.

Details	
CPU nominal frequency	720 MHz
CPU core count	1
RAM	64MB
10/100/1000 Ethernet ports	1
USB ports	0
Wireless Standards	802.11 a/n/ac
Secondary wireless standards	802.11 b/g/n
PoE in / out	Yes / No
Input voltage	11V – 57V
Dimensions	185x85x30 mm
Antenna gain DBI	2
Max power consumption	12W
Number of chains	3
Storage	16 MB FLASH

Table 5. Mikrotik wAP ac device specifications

Price

The current price for this device is 83,00 €.

Open-Mesh OM2P-V2

The OM2P *(Figure 7)* features 23dBm power (200 mW) at even the highest speeds (5-8 dBi higher than most business class access points). With an external AP with standard RP-SMA (coaxial) connector, it provides the flexibility to work with 3rd-party external antennas.

Figure 7. Open-Mesh OM2P-V2

Device specifications

Next table (**Table 6**) shows technical information about OM2P-V2.

Details	
RAM	64MB
10/100 Ethernet ports	2
USB ports	0
Wireless Standards	802.11 g/n
PoE in / out	Yes / No
Dimensions	3.75’’x2.75’’x1’’
Number of chains	1

Table 6. Open-Mesh OM2P-V2 device specifications

Price

The current price for this device is 72,00 €.

2.5. Server

A CentOS cloud-based server will host all these technologies and a reverse-proxy will be needed to allow external access to Kibana as well to redirect HTTP traffic generated by OpenMesh APs towards Logstash.

The details of the server we will do all testing and development are listed on *Table 7*.

Server details	
CPU	Intel Xeon E3-1275 v5 Quad-Core Skylake
RAM	64 GB DDR4 ECC RAM
Hard drive	2x4TB SATA 6Gb/s 7200 RPM
Connection	1 Gbit/s-Port
Guaranteed bandwidth	1 Gbit/s
Operating System	CentOS 7

Table 7. Server details

2.6. Conclusion

Regarding to databases, while MongoDB and Elasticsearch are both document-based databases they both have very different functions. Mongo is great for storing documents, aggregation and retrieval. It can be very fast, but trying to shoe horn in relational data is a bad idea. Relational data belongs in a relational database.

Elasticsearch, great for applications that rely on advanced search features. Say you want to query text based data (like a description or subject) in many different locations and want to do it based on users geographic location etc. Elasticsearch works great for this, you can do this with SQL server, but in my opinion it's easier with Elasticsearch. You can also build more complex searches by using Like-Searches.

Personally I consider Elastic pack (Logstash, Elasticsearch and Kibana) is a very integrated solution with wide documentation, which is very important because I haven’t worked with any of these tools. Elastic pack fits very well with the purpose of this project.

As a reminder:

- Elasticsearch is a distributed, real-time search and analytics engine.
- Logstash is a technology for parsing log data and streaming it into Elasticsearch.
- Kibana is a UI layer that gives the ability to search and graph data.

The most reliable AP, after testing both Mikrotik models, is RB951Ui-2HnD. At Data Analysis section I show differences between them to see which one provides better information.

3. System process

The process of this system starts at the moment where AP captures information of nearby devices. After an interval of time, AP sends the information to the server via HTTP or FTP.

The second step, Logstash is waiting all the time to read incoming data, apply filters and send the information to Elasticsearch, where it will be stored. Once there, we can start to analyze data through Kibana creating charts and different kind of views.

You can see an overview of this process at **Figure 8**. On the left side of the figure, there are represented three APs with its range colored. Then, each AP sends the information via HTTP (for Open-Mesh) or FTP (for Mikrotik) to the server. The cloud represents Internet and the server, on the right side of the figure, hosts Logstash, Elasticsearch and Kibana. Finally, the user who will read all analyzed data, is connected via HTTP to the server and requests the information to Kibana.

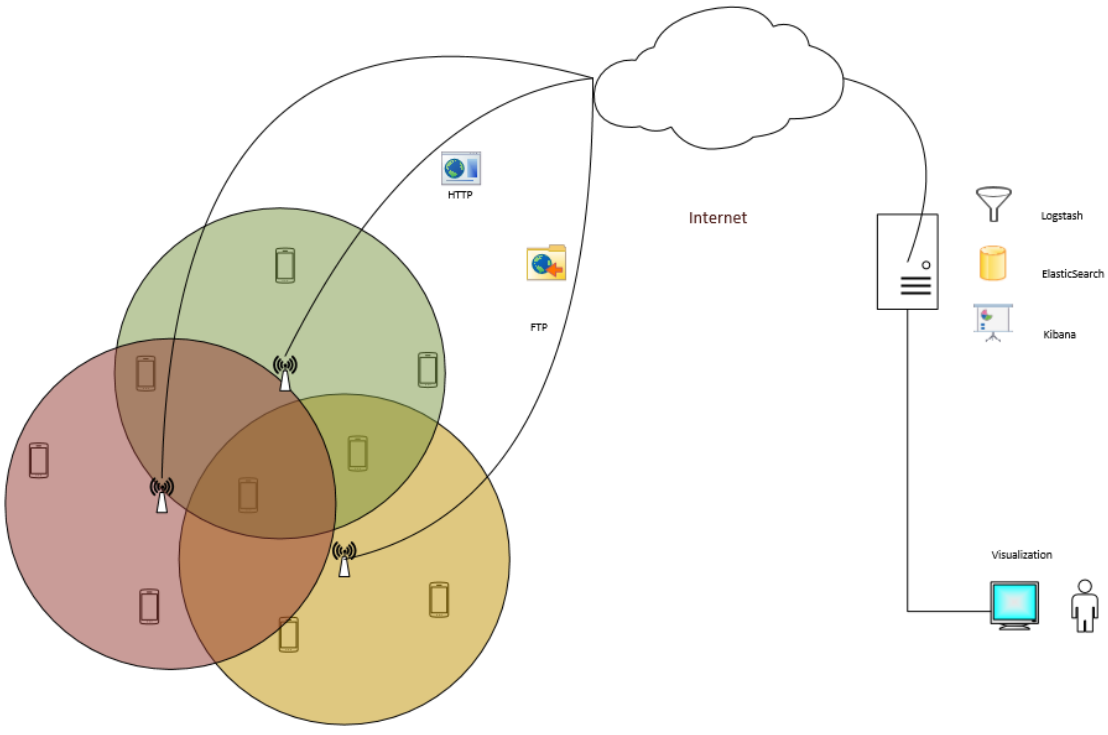


Figure 8. System process overview

3.1. Capturing data

We call presence data the information we capture from nearby devices. Presence data is gathered from the probe request frames sent out by clients that APs could see.

3.1.1. Mikrotik

For Mikrotik APs, to capture presence data, we have to create a script and execute it continuously from AP. This script activates snooper's AP during 55 seconds and saves all captured information into a txt file. Then, this file is sent via FTP to the server.

During these 55 seconds, snooper is capturing presence data in different channel frequencies. At the end of this process, there will be multiple sequences of information for each device captured. The amount of sequences captured for device will depend on the proximity between the device and the AP. If device is very near to the AP there will be around 40 sequences and the furthest devices a lower amount of sequences are captured that it might be one or two sequences. Passed these 55 seconds, file is sent automatically by FTP to the server, where all data will be processed.

To execute this process continuously, we create a job on the AP to run the script every 60 seconds. There is only a five-second timeout, which is when the AP sends the file, that snooper is off.

Magnitude order

Below, **Table 8** showing an estimation of sequences captured during different time intervals for different number of devices. We estimate an average of 20 sequences captured per device for an interval of one minute.

Interval	**# Sequences**		
	1 device	*50 devices*	*100 devices*
1 minute	20	1.000	2.000
60 minutes	1.200	60.000	120.000
8 hour	9.600	480.000	960.000
12 hour	14.400	720.000	1.440.000
1 day	28.800	1.440.000	2.880.000

Table 8. Magnitude order estimation of Mikrotik captured sequences

Sequences

According to Mikrotik documentation *[25]*, there might be different fields between sequences. This is because they capture different type of devices and the information may vary.

The fields we are going to deal with are explained below:

- **Type:** indicates the category of captured device. It can be *station, network* or *frequency.*
    - *Station*: station devices can either be smartphones, laptops, gadgets, printers and others.
    - *Network:* they are network-based devices. They might be routers, access points, switches or other network devices.
    - *Frequency:* sequences captured in antenna's range that could not be identified.
- **Channel:** Defines set of used data rates, channel frequencies and widths.
- **Bitrate:** it measures how much data is transmitted by device at the moment of capture.
- **Active:** if a node is discovered and then it becomes disabled or leaves the area, while snooper is still running then, the value will be false.

- **Address:** provides the mac address of the end client device.
- **Network-ssid:** provides the SSID of the network connected to device.
- **Signal-to-noise:** The noise level indicates the amount of background noise in device environment. If the noise level is too high, it can result in degraded strength and performance for device wireless signal strength. Signal-to-noise ratio (SNR) is the power ratio between the signal strength and the noise level. This value is represented as a +dBm value. In general, we should have a minimum of +25 dBm signal-to-noise ratio. Lower values than +25 dBm result in poor performance and speeds.
- **ssid-source:** from where did snooper learn the SSID, usually beacon, or none if SSID is unknown.
- **use-of-freq:** percentage usage of frequency.
- **use-of-traffic:** percentage usage of traffic
- **freq-source:** where the frequency of the node is learned from.
- **signal-strength:** RSSI[1] measurement of the power present in a received radio signal.
- **network-source:** from where snooper learned about the possible network.

3.1.2. Open-Mesh

Open-Mesh have this script embedded in their APs. There is an option in Cloudtrax that lets you introduce a URL to send the information captured as JSON. You can also specify there the time-interval to send the information. A difference from Mikrotik is that Open-Mesh use protocol HTTP rather than FTP to send the information.

[1] RSSI: Received Signal Strength Indicator

Magnitude order

Open-Mesh arranges the sequences by mac address. So, instead of having 20 sequences for each device as it happens with Mikrotik, we will get one sequence per device and it will provide a field with the number of times it has been counted. (See **Table 9**)

Interval	**# Sequences**		
	1 device	*50 devices*	*100 devices*
1 minute	1	50	100
60 minutes	60	3.000	6.000
8 hour	480	24.000	48.000
12 hour	720	36.000	72.000
1 day	1.440	72.000	144.000

Table 9. Magnitude order estimation of Open-Mesh captured sequences

Sequences

According to CloudTrax documentation *[26]*, the data sent consists of the following elements:

- **node_mac**: Mac address of the Access Point reporting the presence data
- **mac**: mac address of the end client device for which presence data is being reported.
- **Count**: Number of times the specific end client device was seen by the AP, within the time period specified by the "First seen" and "Last seen" timestamps.
- **Min_signal**: Lowest RSSI reading on the AP for the specific client within the time period.
- **Max_signal**: Highest RSSI reading on the AP for the specific client within the time period.
- **Avg_signal**: Average RSSI reading on the AP for the specific client within the time period.
- **Last_seen_signal**: reported RSSI reading on the AP for when this client was last seen
- **First_seen**: Timestamp of the first time this client was seen, during the reporting period.

- **Last_seen**: timestamp of the last time this client was seen.
- **Associated**: indication of whether the client is associated to the AP or not.

Data is sent via a HTTP POST message to the server. No data is stored locally on the APs or on CloudTrax, so if the server specified in the configuration is down or otherwise unreachable, HTTP POST request will fail and cause the data to be lost.

3.2. Process data

This is the most complex and important part of system process. Here we will filter and parse presence data with Logstash. In Logstash configuration folder we set the input channels of data as well as the filters we do want to apply, and the destination of final data.

Once processed data, we will have all the information parsed and ready to be analyzed.

The configuration file has to be located at *conf.d* directory of Logstash and it has the following structure:

```
input  {   ... }

filter {   ... }

output {   ... }
```

3.2.1. Input

Logstash needs to know what input channels of data are to do the filtering process. In our case, input channel will be the directory we send data through FTP and HTTP POST requests.

```
input {
  http {
    port => 8090
    additional_codecs => {
      "application/json" => "json"
      "application/x-www-form-urlencoded" => "json"
    }
    tags => ["openmesh"]
  }
  file {
     path => "/home/rtlsftp/*.txt"
    start_position => "beginning"
    tags => ["mikrotik"]
    max_open_files => 4095
    codec => multiline {
      pattern => "\s*type="
      negate => "true"
      what => "previous"
    }
  }
}
```

HTTP

Logstash is configured to listen HTTP requests from port 8090 and its content will be a JSON structure. A new tag "*openmesh*" is added and it will be helpful to differentiate from Open-Mesh and Mikrotik APs.

FILE

The second input is specific for reading files from a directory. In that directory will be the txt files that Mikrotik sends after snooping process. A tag "*Mikrotik*"

is added to recognize easily Mikrotik sequences. Logstash will read a maximum of 4095 files at the same time as more than that Logstash will considerably slow down.

The *multiline* codec means that Logstash has to start reading the sequences by some *pattern*. Sequences are stored in the file with the first line indented followed by "*type=".* The *what* parameter must be *previous* or *next* and indicates the relation to the multi-line event. The *negate* parameter can be *true* or *false*. If *true*, a sequence not matching the *pattern* parameter will constitute a match of the multiline filter and the *what* parameter will be applied.

3.2.2. Filters

In filter section there will be all filters we need to apply to parse data and leave it as our necessities. We apply different filters, such renaming fields, create new data from incoming information, retrieve Organization information of captured devices, etc.

A filter plugin performs intermediary processing on an event. Filters are often applied conditionally depending on the characteristics of the event.

We will basically work with the following filters:

- **Grok**: Parse arbitrary text and structure it. Grok is currently the best way in Logstash to parse crappy unstructured log data into something structured and queryable.
- **Mutate**: The mutate filter allows you to perform general mutations on fields. You can rename, remove, replace, and modify fields in your events.
- **KV**: This filter helps automatically parse messages (or specific event fields) which are of the foo=bar variety
- **Ruby**: Executes ruby code. If you need to create additional events, it cannot be done as in other filters where you would use yield, you must use a specific syntax.
- **OUI**: It parses OUI data from mac address.
- **Split**: Split filter clones an event by splitting one of its fields and placing each value resulting from the split into a clone of the original event. The field being split can either be a string or an array.

3.2.3. Output

An output plugin sends event data to a particular destination. Outputs are the final stage in the event pipeline. We need to store all data to Elasticsearch so we will indicate this into the output section.

```
Output {
   Elasticsearch { hosts => [“localhost:9200”] }
}
```

3.3. Visualization

Kibana will be the tool for which data is displayed. It is connected permanently to Elasticsearch and it can be used to display Elastic data and to manage and make some configurations of Elasticsearch. It is accessed using any web browser.

Kibana accepts http requests to retrieve data and so it has been considered to develop a customized dashboard instead of using Kibana for displaying data. It’s not yet discarded to do it in a near future.

Kibana allows you to save configured graphs and create multiple dashboards with different visualizations on it.

Visualizing data is the last part of the system process.

4. Data processing

This section explains all steps for data processing in order to get desired information. It will be explained the most important filters applied and how the information is treated.

4.1. Initial data

It is necessary to understand how Logstash sends data structures to Elasticsearch and how is it saved without applying any filters. Then, after a first overview, we will be able to format desired data, remove unnecessary data, and obtain new information.

4.1.1. Open-Mesh

Below, on **Table 10** there is a sample of data structure from Open-Mesh APs with a short description of its fields.

Field	Value	Description
@timestamp	May 18th 2017, 23:42:35.724	Timestamp of the insert into ElasticSearch (ES)
@version	1	Logstash variable
_id	AVwdgy_cRUUPbOeCD886	Document identifier in ES
_index	logstash-2017.05.18	Index name which is stored this document in ES
_score	-	Document relevance
_type	logs	Document type by ES
headers.content_length	975	HTTP headers
headers.content_type	Application/json	HTTP headers
headers.http_accept	*/*	HTTP headers
headers.http_connection	close	HTTP headers
headers.http_host	88.99.160.236	HTTP headers
headers.http_signature	a4106a3b2c4b3889772f18efdd026a88b8245d54306238423fb21086e2d479e4	HTTP headers
headers.http_version	HTTP/1.0	HTTP headers
headers.http_x_forwarded_for	52.24.41.247	HTTP headers
headers.http_x_real_ip	52.24.41.247	HTTP headers

Field	Value	Description
headers.request_method	POST	HTTP headers
headers.request_path	/openmesh/	HTTP headers
headers.request_uri	/openmesh/	HTTP headers
host	127.0.0.1	ES host
network_id	235932	Cloudtrax network id
node_mac	AC:86:74:0D:DC:D0	Mac address of Open-mesh antenna
probe_requests	{ "first_seen": 1495143737, "last_seen": 1495143737, "last_seen_signal": -42, "max_signal": -39, "associated": false, "count": 2, "min_signal": -42, "avg_signal": -40, "mac": "04:15:52:15:7d:32" }, { "first_seen": 1495143695, "last_seen": 1495143695, "last_seen_signal": -89, "max_signal": -87, "associated": false, "count": 2, "min_signal": -89, "avg_signal": -88, "mac": "24:09:95:eb:d4:32" }	Captured sequences from snooper
tags	openmesh	Open-Mesh tag added in the input configuration
version	1	Logstash variable

Table 10. Initial structure of Open-Mesh data stored into Elasticsearch

4.1.2. Mikrotik

Mikrotik structures of data are a bit different of Open-Mesh AP's basically because it is imported from files rather than HTTP and also the information provided by Mikrotik AP is different and more detailed. See **Table 11**, to see a sample of Mikrotik data structure.

Field	Value	Description
@timestamp	May 18th 2017, 23:42:21.631	Timestamp of the insert into ES
@version	1	Logstash variable
_id	AVwdgvjIRUUPbOeCD883	Identifier of the document in ES
_index	logstash-2017.05.18	Index name which is stored this document in ES
_score	-	Relevance of the document
_type	logs	Document type by ES
host	CentOS-73-64-minimal	ES host
message	type=station channel="2412/20/gn" bitrate=19.9kbps active=yes address=E4:8D:8C:53:AF:9B network-ssid="MikroTik" signal-to-noise=69dB ssid-source=beacon use-of-freq=2.1% use-of-traffic=10.8% freq-source=network-on-freq signal-strength=-37dB network-source=forms-network	Sequence captured from Mikrotik snooper
path	/home/rtlsftp/2017may18_23_39_39_6281058C55E3.txt	Path of the origin file
tags	multiline, mikrotik	Tags field. We added Mikrotik tag in the input section of configuration file, so we recognize this is a sequence of Mikrotik. We also specified this is a multiline as the document provides multiple sequences.

Table 11. Initial structure of Mikrotik data stored into ElasticSearch

4.2. Building a readable structure

Now that initial structures of data stored into Elasticsearch, both for Mikrotik and Open-Mesh, are explained, you can proceed to apply filters and do a structure of relevant data.

As seen in both samples structure for Open-Mesh and Mikrotik APs, there is a lot of unnecessary and useless information that should be avoided storing into Elasticsearch. Moreover, it is necessary a readable structure of data in both cases and read the information by same name fields.

Open-Mesh

For Open-Mesh structures of data, we need to apply the following filters to Logstash:

Split probe requests

We do split *probe_requests* field in order to have all the information provided by CloudTrax in different fields instead of having them in different arrays.

Renaming fields

We do rename the following fields so we can read the same field name for Mikrotik and Open-Mesh information.

1. probe_requests.mac → mac_address
2. probe_requests.last_seen_signal → signal_strength_int
3. node_mac → device_id

Additional fields

Open-Mesh do always capture information at 2426 MHz frequency channel and so there is not a field with the channel frequency on the information they provide. For that reason, we add a field named *channel_frequency_int* with *2426* value.

Removing useless data

We remove the fields *http, host, network_id, probe_requests.associated, probe_requests.first_seen, probe_requests.last_seen, probe_requests.min-signal* and *probe_requests.max_signal* because we don't need this information.

Mikrotik

For Mikrotik structures of data, we need to apply the following filters to Logstash:

Drop undesired sequences

We will drop undesired sequences such as *frequencies*, *network* and *logs* type. We also drop the entire file if it comes empty.

Split message

We split *message* field into different fields, so instead of having all snooper sequence in a straight line, we have a field for each value. This will help to find results easily when querying.

Channel, signal_strength and signal_to_noise

Next filter we are going to apply is for the fields: *channel*, *signal_strength* and *signal_to_noise*. We need to cut channel number off because we will need frequency channel as an integer value. We also need signal strength and signal-to-noise as an integer value and not with unit measurement *dB*. We will need these values to calculate the distance.

Renaming fields

We will rename some fields for the same reason we did it for Open-Mesh data.

1. Channel → channel_frequency_int
2. Signal_strength → signal_strength_int
3. Signal_to_noise → signal_to_noise_int
4. Address → mac_address

Removing fields

We remove field *message* because we already have the information in different fields when we split it.

4.2.1. Sample structure

After applying these initial filters, **Table 12** and **Table 13** show the result of these filters, for Open-Mesh and Mikrotik respectively.

Open-Mesh

Field	Value
@timestamp	May 21st 2017, 23:42:21.631
@version	1
_id	AVwdgvjIRUUPbOeCD883
_index	logstash-2017.05.21
_score	-
_type	logs
Channel_frequency_int	2426
Device_id	AC:86:74:0D:DC:D0
Mac_address	34:23:BA:CB:77:4B
Probe_requests.avg_signal	-85
Probe_requests.count	2
Signal_strength_int	-86

Table 12. Open-Mesh structure of data after applying initial filters

Mikrotik

Field	Value
@timestamp	May 21st 2017, 23:32:21.631
@version	1
_id	AVwdgvjIRUUPbOeCD883
_index	logstash-2017.05.21
_score	-
_type	station
Channel_frequency_int	2412
Freq_source	Network_on_freq
Mac_address	E4:8D:8C:53:AF:9B

Field	Value
Network_source	Forms_network
Signal_strength_int	-39
Signal_to_noise_int	68
Tags	Multiline, mikrotik
Type	station

Table 13. Mikrotik structure of data after applying initial filters

4.3. Calculating distance between device and AP

Channel frequency band and signal strength will be helpful to obtain distance parameter.

The formula *[27]* applied to obtain distance between device and AP is the following:

$$Distance = 10 \wedge ((FSPL - (20 * \log_{10}(FC)) + |SS|) / 20)$$

FSPL (Free-Space Path Loss)

FSPL *[28]* is the loss in signal strength of an electromagnetic wave through free space with no obstacles to cause reflection. It is expressed in dB and there are different standard constants defined depending on distance and frequency channel measures.

- 87.55 dB when distance is expressed in meters and frequency in KHz.
- 27.55 dB when distance is expressed in meters and frequency in MHz.
- 32.45 dB when distance is expressed in kilometers and frequency in MHz.

FC (Frequency Channel)

Frequency channel is the frequency channel band on which AP has detected the device. There are different channels and frequencies on which devices can communicate with other devices. For Wi-Fi communications, it is usually 2.4 GHz (2400Mhz) and 5GHz (5000MHz). Each of these bands, have multiple channels.

The 802.11 WLAN standards specify a bandwidth of 22 MHz and channels are on a 5 MHz incremental step. Often nominal figures for the channel bandwidth of 20 MHz are often given. The 20 / 22 MHz bandwidth and channel separation of 5 MHz means that adjacent channels overlap and signals on adjacent channels will interfere with each other. **Figure 9**, shows a graphical representation of 2.4 GHz band channels. Mikrotik and Open-Mesh APs will detect presence data within these bandwidths.

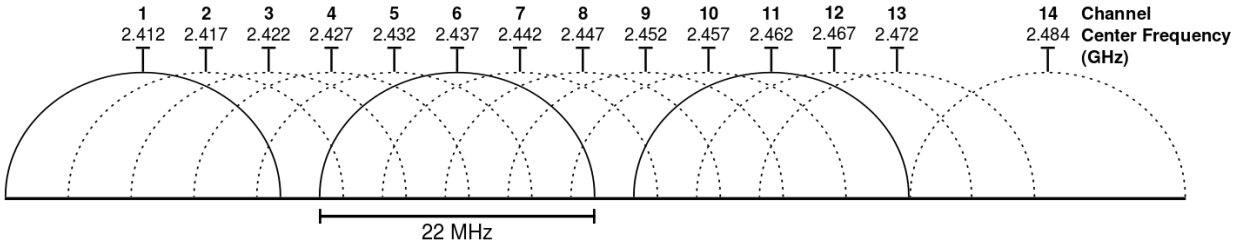


Figure 9. Graphical representation of 2.4 GHz band channels overlapping

SS (Signal Strength)

Signal Strength refers to the transmitter power output as received by a reference AP at a distance from the transmitting area. It is usually expressed in dBm, decibels above a reference level of one milliwatt. Its range becomes from 0 dBm (very close to the AP) and -100 dBm (very far from the AP).

4.4. Additional data

We will add more data to enrich the information that we are going to provide and get more value to the information we already have.

4.4.1. OUI (Organizationally Unique Identifier)

To know the device manufacturer we are going to use OUI filter. It will return the following structure of information by just providing the mac address:

- Organization: Organization name (Manufacturer)
- Address: Information relative to the address of the Organization headquarters.

- ID: Organization identifier
- Country: Country of the Organization headquarters.

We will only need organization name so we will just take *organization* field of the structure.

You can see few examples on **Table 14**

Mac Address (first 6 digits)	OUI			
	Organization	Address	Id	Country
18:AF:61:00:00:00	Apple, Inc.	1 Infinite Loop Cupertino CA 95014	267602	US
84:0B:2D:00:00:00	Samsung Electronics Co.,Ltd	94-1, Imsoo-Dong Gumi Gyeongbuk 730-350	15205274	KR
5C:51:88:3A:DE:37	Motorola Mobility LLC, a Lenovo Company	222 West Merchandise Mart Plaza	8654861	US

Table 14. Mac-OUI example

4.4.2. Client

We want to add a field that identifies the client, which establishment of the client (if it has more than one) and also a descriptive name of the AP responsible of the captured sequence.

As shown on Mikrotik's sample data, it doesn't provide an identifying field as Open-Mesh does with *device_id field*. For that reason, Mikrotik AP will send the file named with the current date-time and the AP's serial number. Then, when data is processed, we will add a field with the same name as Open-Mesh sequences *(device-id).*

We will add the following fields based on device-id:

- Client: Name of the client where we implemented the solution.

- AP: distinctive name for the AP. It will be useful to be identified if there is more than one AP.
- Location: refers to the establishment of the client. It is useful if a client has multiple locations or establishments with this solution. This way it will be very helpful to do statistics of different locations for the same client.

4.4.3. Range

Range field indicates if the node captured belongs insider or outsider. It will be relative to the distance between AP and device. For each AP, a maximum distance is defined, so over that distance, range will be outsider.

We set a maximum distance for each AP because one might have a radius of five meters for just insiders and another one might be ten meters radius.

We could obviate this field as we can calculate it from Kibana or our own visualization software.

4.5. Final data structure

After processing all incoming information, Mikrotik and Open-Mesh structures of data will have the same structure and fields. See **Table 15**.

Field	Value
@timestamp	May 21st 2017, 23:32:21.631
@version	1
_id	AVwdgvjIRUUPbOeCD883
_index	logstash-2017.05.21
_score	-
_type	station
Antenna	01
Channel_frequency_int	2412
Client	VIGNA
Device_id	4AC704B6606B
Distance	30.95
Freq_source	Network_on_freq
Location	ST_JOSEP
Mac_address	00:21:E9:53:AF:9B

Field	Value
Network_source	Seen_data_frame
Oui.organization	Apple Inc.
Range	Outsider
Signal_strength_int	-39
Signal_to_noise_int	68
Tags	Multiline, mikrotik
Type	station

Table 15. Final structure of data

5. Analysis

On this section we are going to analyze the information through Kibana and adjust filters in order to get the information accurate. Graph views will be helpful to analyze the information for Open-Mesh and Mikrotik APs. We do this analysis because as we mentioned at the beginning, Open-Mesh has this script integrated with Cloudtrax and we develop the script to implement this with Mikrotik APs.

Below, different analyses are done in a small office of approximately 10 square meters. This office is located in an office complex. You can see a map of the office at **Figure 10**. It is located in a second floor level. At the end of the office there is a big window with the views of a very concurrent street.

APs for analyses:

- Antenna 01: Mikrotik wAP ac
- Antenna 02: Mikrotik RB951Ui-2HnD
- Openmesh: OM2P-V2

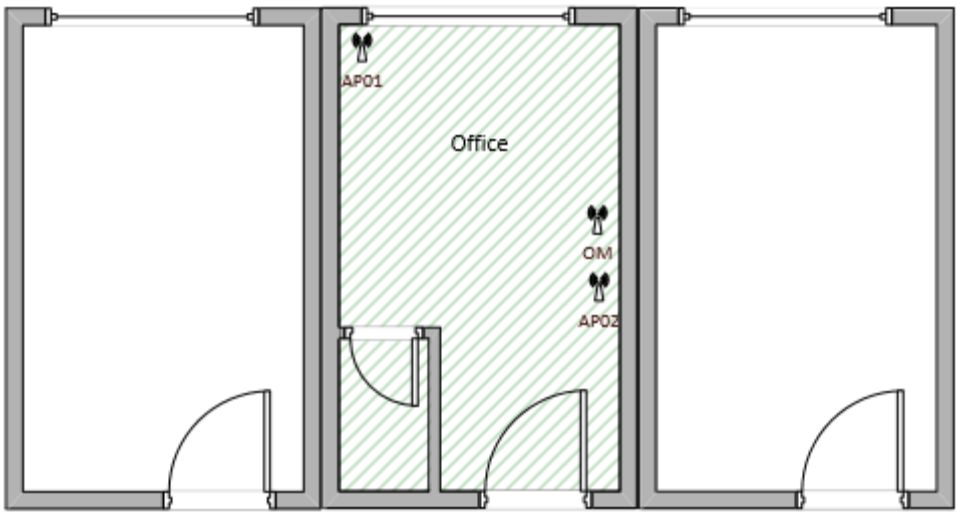


Figure 10. Office map and distribution

5.1. Devices recognized

First analysis we will run will be based on the number of devices that are detected for each AP in different circumstances.

First, we turn on only one device inside the office and we are going to check on Kibana what is detected for each AP within a time interval of five minutes.

As we can see on *Table 16*, Mikrotik APs finds more devices than Open-Mesh, in order to know why does it happen, we will analyze in detail Mikrotik's captured devices.

	Antenna 01	**Antenna 02**	**Open-Mesh AP**
Filter	*Client: SWB AND antenna: 01 AND device_type: insider*	*Client: SWB AND antenna: 02 AND device_type: insider*	*Openmesh AND insider*
Total devices	6	3	3
Devices	***00:08:22:22:EC:FB*** 00:03:AB:DD:84:C1 ***04:15:52:15:7D:32*** ***0C:D6:BD:F3:89:61*** 00:03:AB:20:E1:86 00:08:22:3B:7B:4A	***00:08:22:22:EC:FB*** 00:03:AB:DD:84:C1 ***04:15:52:15:7D:32***	***0C:D6:BD:F3:89:61*** ***04:15:52:15:7D:32*** ***00:08:22:22:EC:FB***

Table 16. Detected devices comparative between different APs

Logstash shows that fields *freq_source* and *network_source* of the sequences have different values:

Freq_source

- Seen_frame_on_freq: Package from device was seen on this frequency.
- Network_on_freq: Device works in network, which is on this frequency.

Network_source:

- None: unknown.
- Forms_network: device wants to form network (is an access point).

- Seen_data_frame: data package from this device on this network.
- Seen_successful_assoc: device association with this network was seen.
- Seen_successful_auth: device authentication with this network was seen.

Now, next table (See *Table 17*) shows a more accurate result of nearby devices for Mikrotik APs.

	Antenna 01	**Antenna 02**	**Open-Mesh AP**
Filter	*Client: SWB AND antenna: 01 AND device_type: insider AND freq_source: seen_frame_on_network AND nework_source: none.*	*Client: SWB AND antenna: 02 AND device_type: insider AND freq_source: seen_frame_on_network AND nework_source: none.*	*Openmesh AND insider*
Total devices	4	2	3
Devices	***00:08:22:22:EC:FB*** ***04:15:52:15:7D:32*** 0C:D6:BD:F3:89:61 ***00:08:22:3B:7B:4A***	***00:08:22:22:EC:FB*** ***04:15:52:15:7D:32***	***0C:D6:BD:F3:89:61*** ***04:15:52:15:7D:32*** ***00:08:22:22:EC:FB***

Table 17. Nearby devices comparative filtered by frequency source and network source

With this new filter applied, we can see that two devices match in three different APs and one more devices is detected on Mikrotik AP01 and Open-Mesh AP. There is another one that could only be detected by Mikrotik AP01.

The main reason, for which Mikrotik AP01 is detecting more devices than others, is that the antenna is more sensible and powerful than the others. Even so, other devices might

be detected because the APs are not in the middle of the office but in a side of it. So the other devices might be in the office next to ours as shows **Figure 11** and **Figure 12**. This problem can be fixed using more than one AP and triangulating the position of the device (see **Figure 13** on the next page).

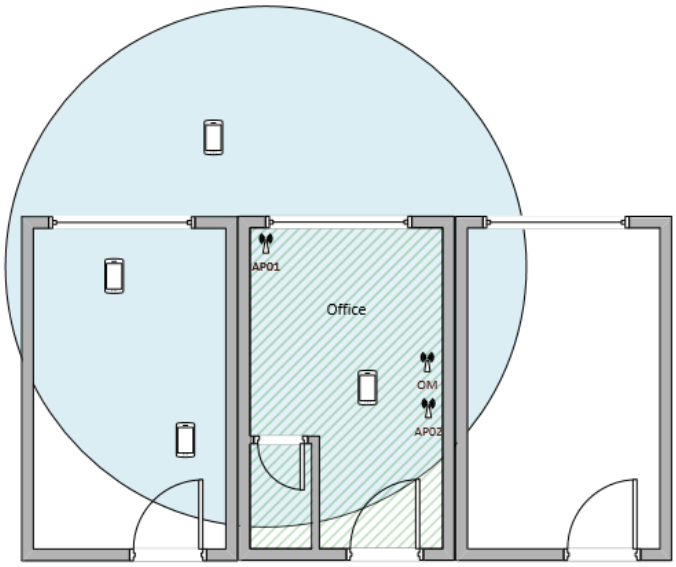


Figure 11. AP01 insiders' area

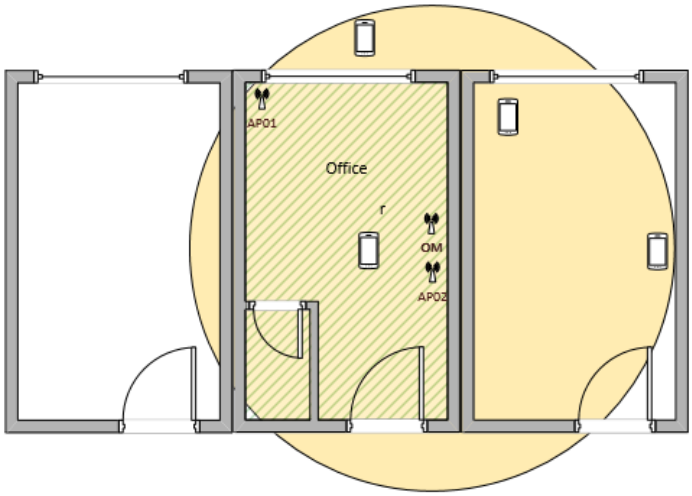


Figure 12. OM and AP02 insiders' area

Figure 13 shows a visual representation of tracking devices with different APs and triangulating their position. This helps to discard devices when a device is not seen at the same time by the three APs.

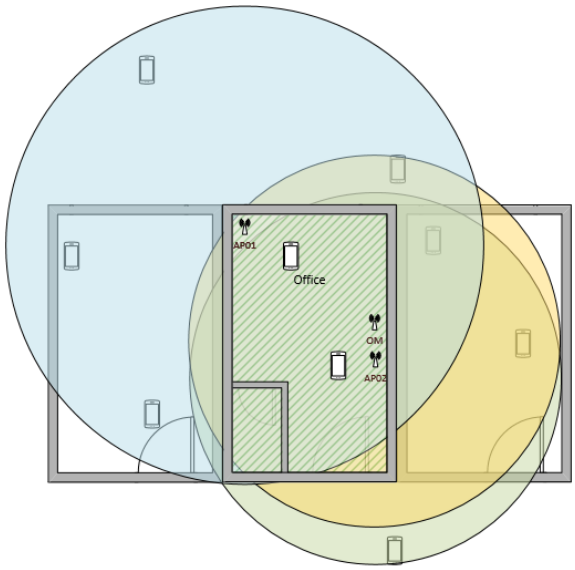


Figure 13. Triangulation of a device

Graphical representation of counted devices

Next graph (see **Figure 14**) shows, in an interval of 24 hours, the total amount of devices captured for each AP for insiders and outsiders. Green line refers to Mikrotik AP01, red line to Open-Mesh, and blue line, Mikrotik AP02. Total count of devices is commented on **Table 18.**

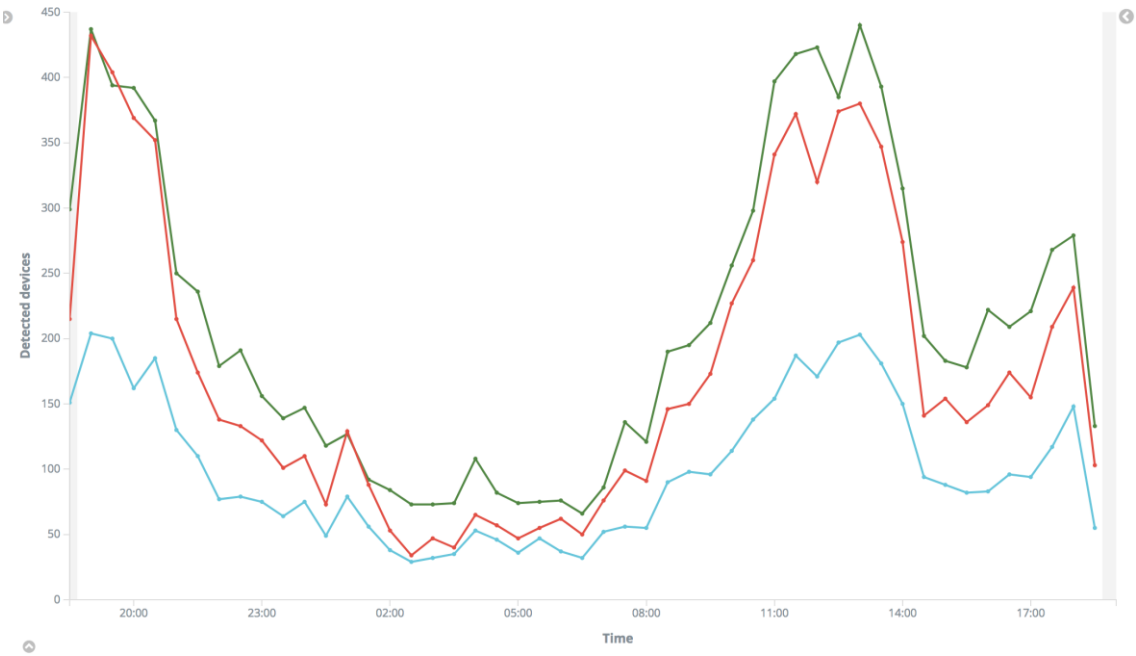


Figure 14. Counting devices comparative

AP01	AP02	OM
6.789	3.175	6.808

Table 18. Total count of devices.

5.1.1. OUI Organization distribution

If we look closer to the devices Mikrotik AP01 and Open-Mesh have detected, we can see that they are seeing almost the same devices. On **Figure 15** you can see the OUI organization distribution in the same 24-hour interval time.

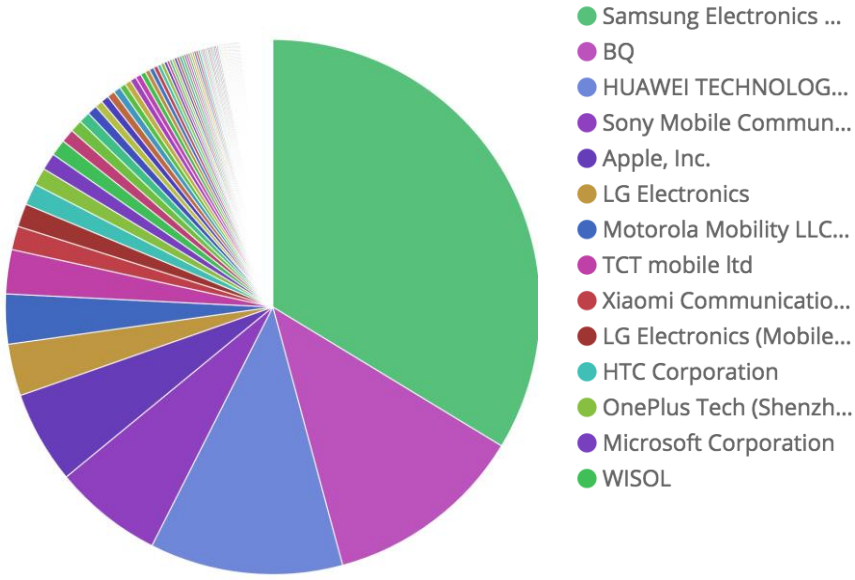


Figure 15. OUI Organization distribution

Table 19 shows the TOP 5 organization of captured devices in the last 24 hours grouped by AP01 and OM AP.

Organization	AP01	OM
Samsung Electronics	1.608 (47,70 %)	1.222 (56,76 %)
Huawei	554 (16,43 %)	275 (12,77 %)
BQ	493 (14,62 %)	293 (13,61 %)
Motorola	443 (13,14 %)	140 (6,50 %)
Sony	273 (8,10 %)	223 (10.36)
Total	***3.371 (100 %)***	***2.153 (100 %)***

Table 19. Top-5 OUI organization AP01 vs OM

5.1.2. Insiders *vs* Outsiders

This analysis consists to see the differences between ranges from testing APs. Insiders' range should be almost the same but there will be differences on outsiders because of the sensibility of the antenna.

As you can see on **Table 20**, AP01 and OM are detecting the same amount of devices whereas AP02 is quite far to get the same results.

	AP01	AP02	OM
Insiders	2	3	2
Outsiders	2.362	1.085	2.182

Table 20. Insider vs outsider comparison

Graphical representation

You can see a distribution of these detected devices by insiders and outsiders for each AP on **Figure 16**.

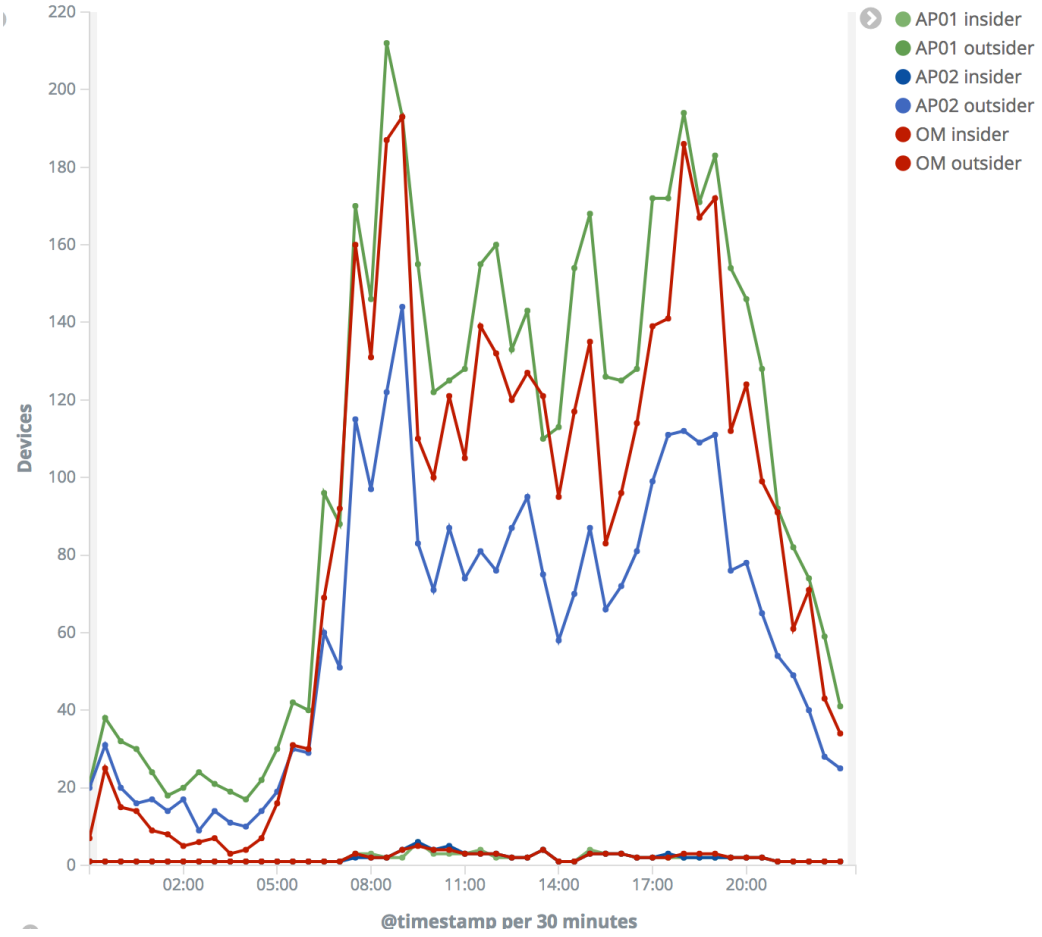


Figure 16. Graphical representation of outsiders and insiders

5.2. Distance

The same device we used in the previous analysis will be now tracked to check how distance is accomplished. The next table (see **Table 21**) shows the real distance between the device and different APs and **Figure 17** shows office distribution.

Figure 17. APs and tracked device distribution

	AP01	AP02	OM
Ipad	2.5 meters	1.5 meter	1 meter

Table 21. Real distance between tracked device and APs

As we see on **Figure 18**, there are a lot of peaks for Mikrotik APs. This is caused by the noise there is in the signal. Green line refers to AP01, blue line to AP02 and red line to OM. As we can see, Open-mesh is the most stable one. Device has been in the same place for all measurement.

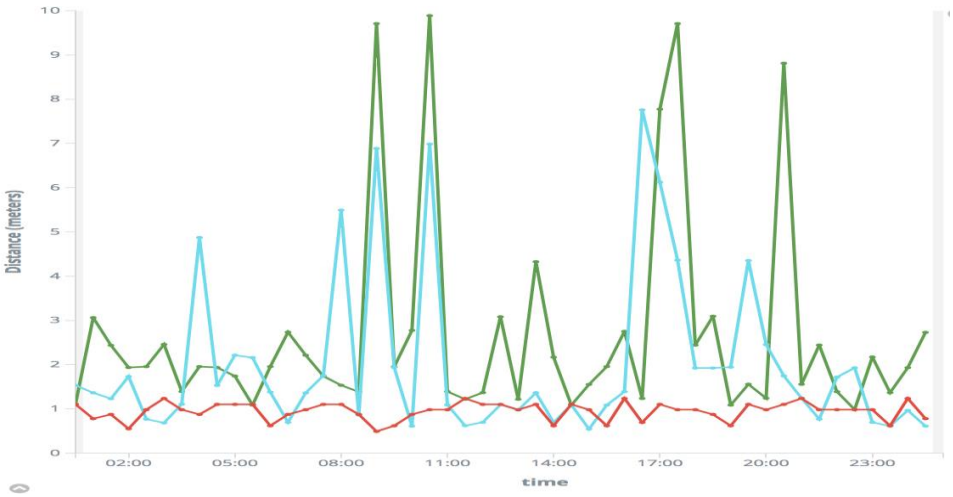


Figure 18. Distance analysis

Now, if we filter the sequences of Mikrotik APs by adjusting SNR (signal to noise ratio), we will accomplish more accurate distance. The next line graph, see **Figure 19**, shows the distance with filtered noise level.

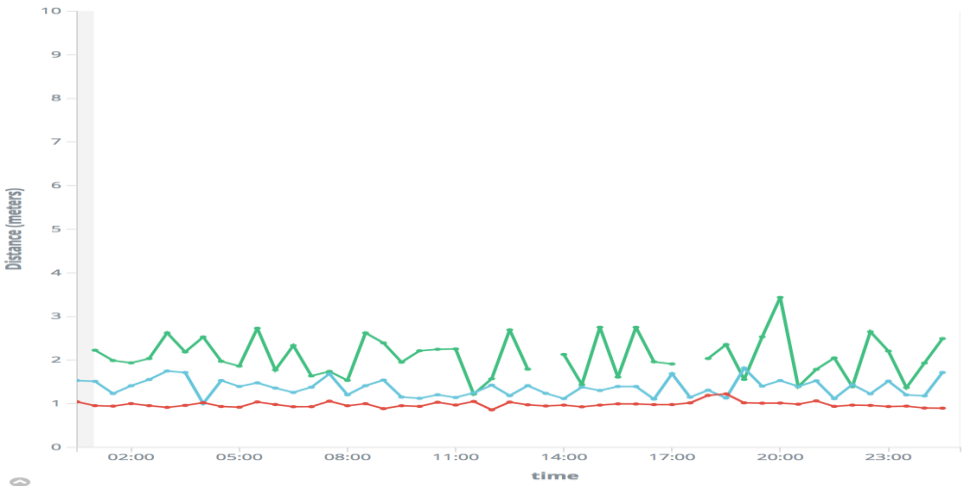


Figure 19. Distance analysis after SNR filter

6. Real scenario

This section explains a real scenario in a small store. This shop is located in a centric street of Mataró and they sell woman clothes.

At this store, there is only one AP installed. Next figure (**Figure 20**) shows a representation of the store.

Figure 20. Customer store distribution

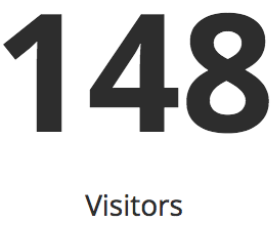
Customer asks us if we could answer these questions:

1. Number of customers inside the store.
2. Number of customers stopped by the storefront.
3. Ratio between customers stopped by the storefront and customers entered in.
4. Time with more customers.
5. Time with more customers at storefront.
6. The smartphone most used by customers.
7. Most concurrent day of the week.
8. Top 10 customers.
9. Warmest zones of the store.
10. Ratio of customer's return.

6.1. Metrics

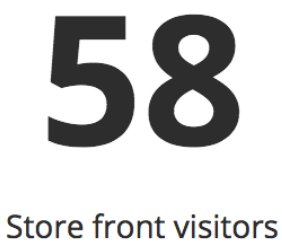
1. Number of customers inside the store.

Counter of figure X shows the visitors that entered in the store on Saturday, May 27th.



2. Number of customers stopped by the storefront.

Counter of figure X shows the visitors that stopped at the storefront on Saturday, May 27th. In order to be counted, visitor has to be at the storefront at least 1 minute.



3. Ratio between customers stopped by the storefront and customers entered in.

This metric will be implemented on the next version of the system. Although we have the values to accomplish with the results of this metric, the system is not ready to display it. In order to display it, we will export grouped data from Elasticsearch to a MySQL database.

4. Time with most customers.

Customer asks to provide him what is the time with most visitors. A good perspective of this metric is showing a distribution of these visitors.

Next graph (see **Figure 21**) represents with bars the total amount of customers entered in the store. It is grouped by hours so it can be easily interpreted. We can see that the most crowded time is at 6pm. By hovering cursor over the bar, it will say the number of customer and the time.

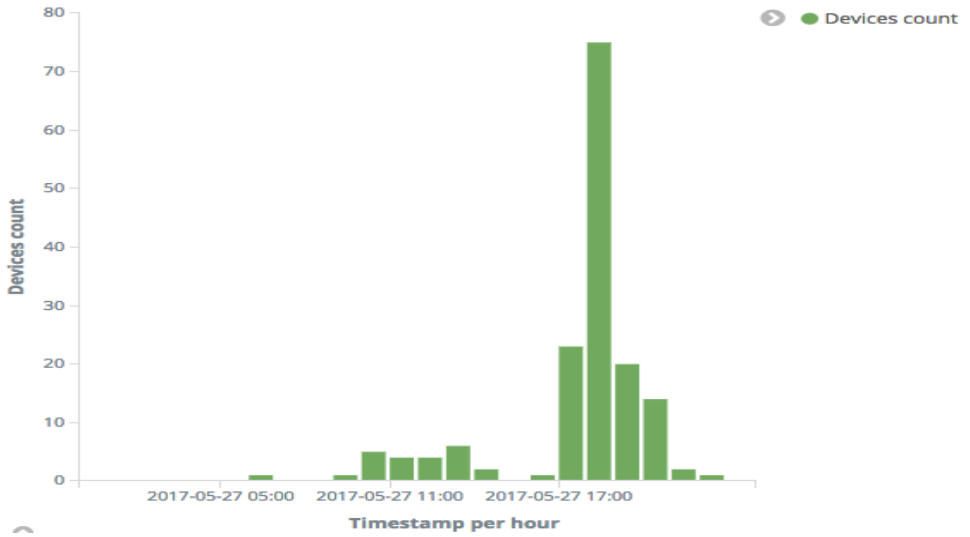


Figure 21. Visitors distribution by hours

5. Time with most customers at storefront.

It would be very nice if Kibana would let us overlap the previous bar graph with this line graph (see **Figure 22**). Next line graph shows a distribution of customers stopped at the storefront grouped by an interval of 30 minutes.

This is only estimation and does not fit the reality with exactitude. This is because it is only measured by one AP, and we base the customer is there by the distance from the AP. This will never be 100% accurate with just one AP because AP captures devices in area and those devices detected can be x meters in the opposite direction.

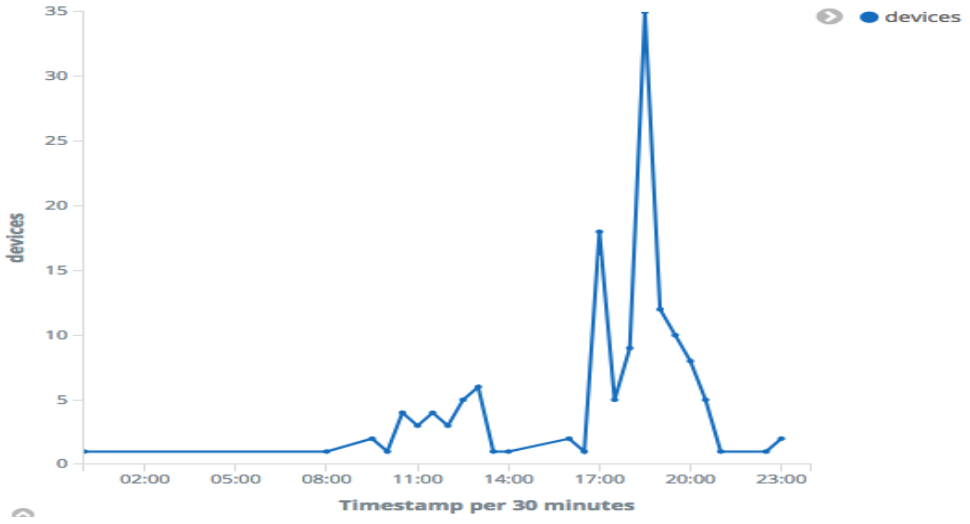


Figure 22. Storefront visitors distribution

6. Most used smartphone by customers.

This metric is represented in a pie chart and shows the top 5 devices organization of all customers entered in the store. Shows a real graph during a day in the store.

Top 5	
Motorola	77
Samsung	29
Apple	10
BQ	9
Huawei	8

Table 22. Top-5 Device organization

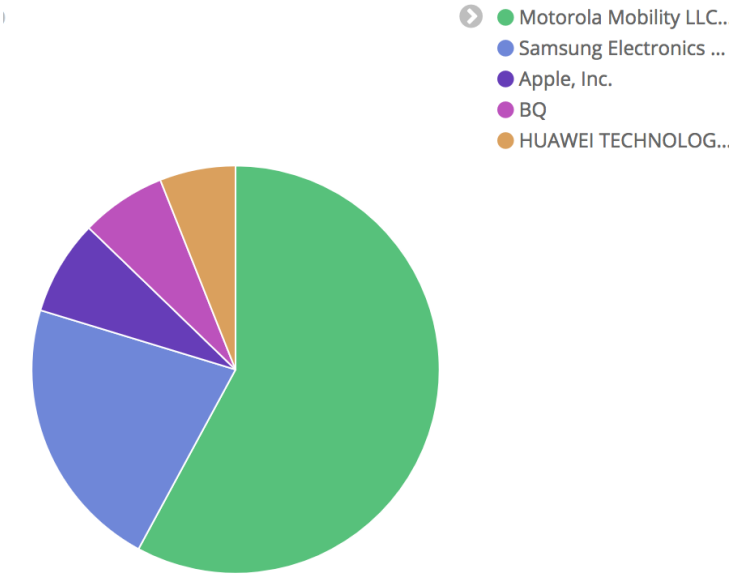


Figure 23. Device organizations distribution

8. Most concurrent day of the week.

Next graph (see figure x) shows a quick overview of which day for the last seven days was the most concurrent. Based on the graph below, we can see that the day with most visitors was on May 27th, which is a Saturday.

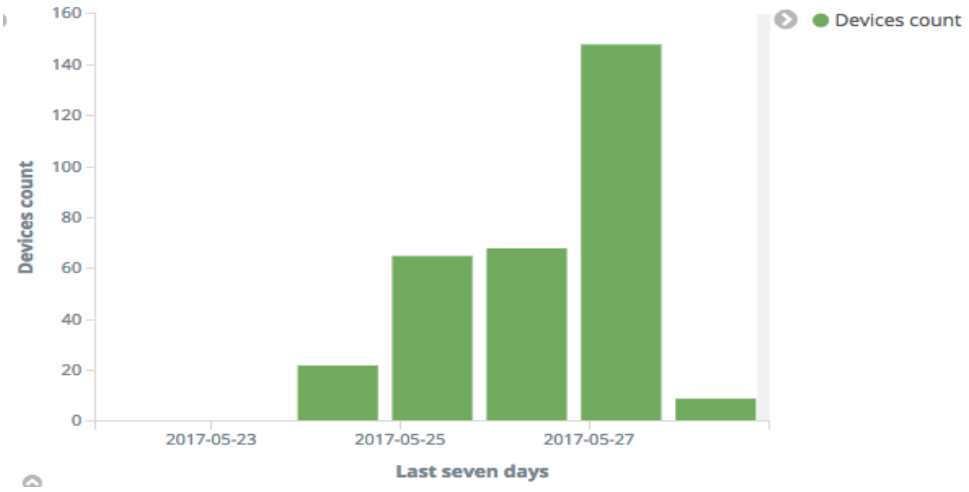


Figure 24. Visitors distribution for the last 7 days

9. Top 10 customers

This metric provides the 10 customers with more return to the store. Same as for metric #3, it will be implemented on the next version. We will export data from Elasticsearch to a MySQL database to later on, show all these metric in our custom dashboard.

10. Warmest zones of the store

At the moment we can only calculate this metric by a graph rather than using a warm map. This graph can be interpreted by Y-Axis de number of counted devices and X-Axis for the distance at any point of AP's radius.

Next graph (see **Figure 25**) shows a total count of devices every half a meter for the last 7 days. The best solution in that cause would use three APs instead of one and determine with exactitude the location of the device.

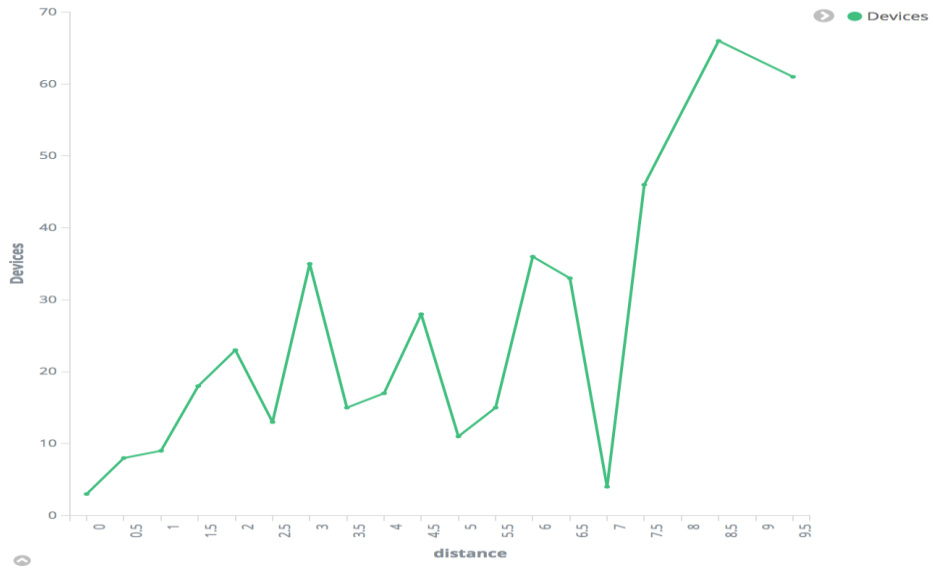


Figure 25. Number of devices in a given radius distance

11. Customer return ratio.

This metric will be implemented on the next version just for the same reasons as metrics #3 and #9.

6.2. Kibana dashboard

A Kibana dashboard displays a collection of saved visualizations. For this customer, it has been created a customized dashboard *[29]* with the previous explained metrics.

From the dashboard, you can choose the time interval that will be displayed all graph data. First metrics shown on the dashboard are count number of visitors and storefront visitors, a distribution graph of these visitors, and another graph with the hourly visitors within the last seven days. You can see a preview of this information on **Figure 26**.

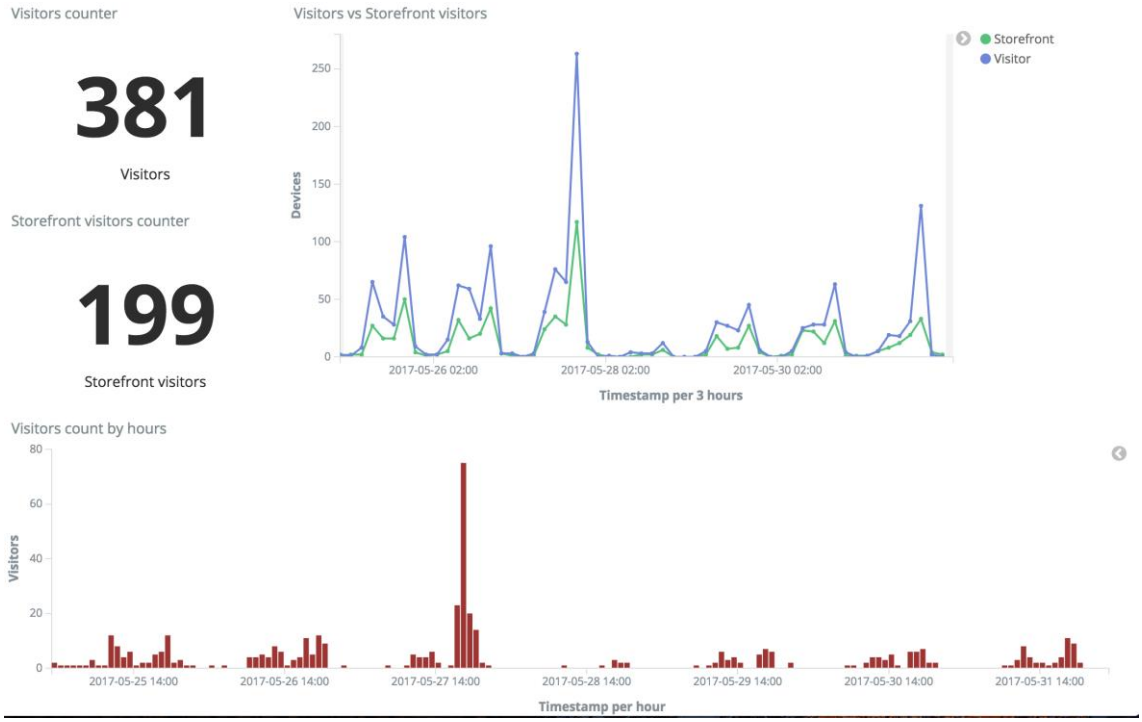


Figure 26. Snapshot 1 of customer dashboard

Above these graphs, there will be another line graph with a distribution of visitors who are out of the store and visitors who enter the store. Followed, there is a graph with a distribution of the storefront visitors and another graph of daily visitors. **Figure 27** shows a snapshot of these graphs.

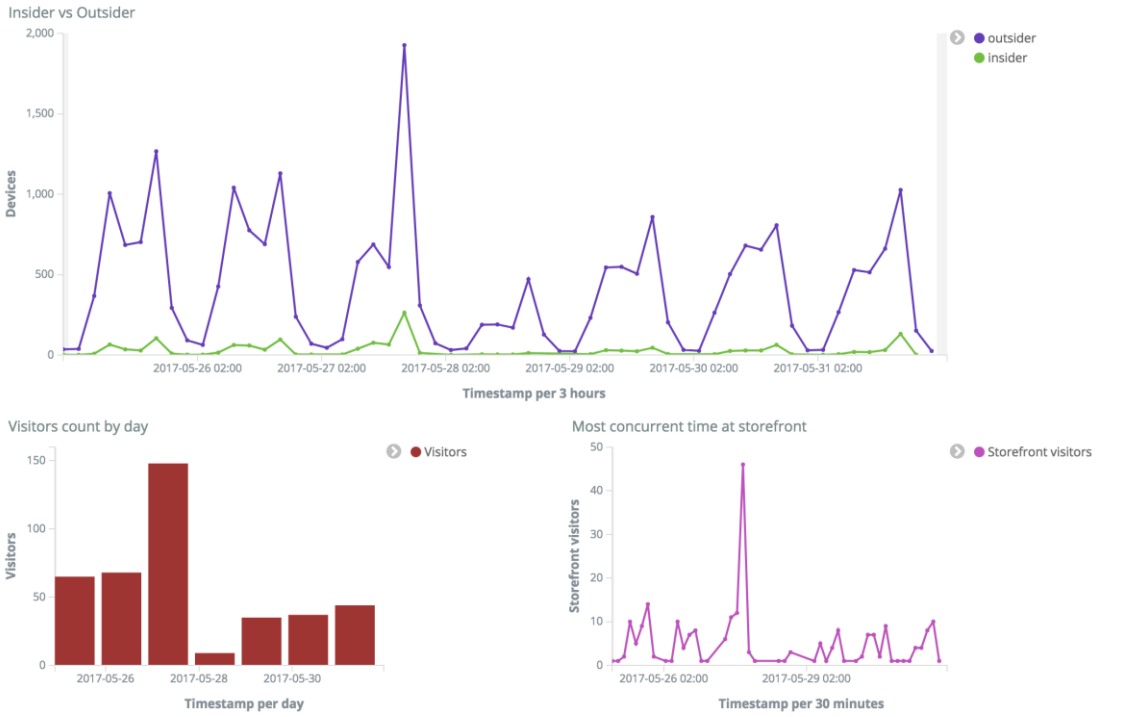


Figure 27. Snapshot 2 of customer dashboard

Finally, last graphs shown in this dashboard provide information of the warmest zones of the store and the top five device organizations. See **Figure 28.**

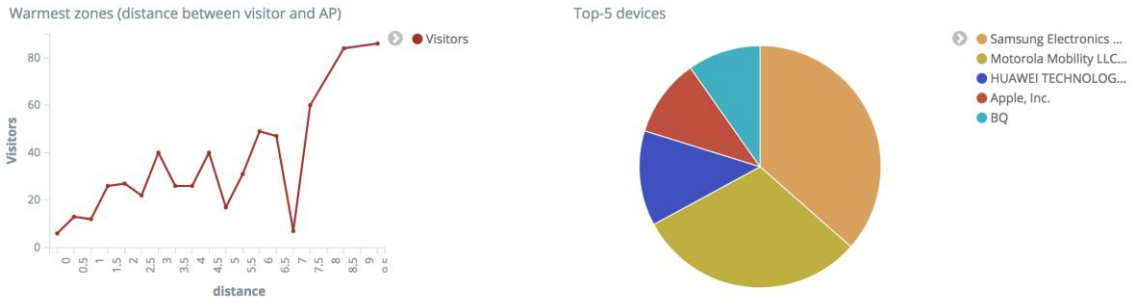


Figure 28. Snapshot 3 of customer dashboard

7. Conclusion

This project has met all expectations. It has been a complex process since certain technologies were unknown and what have led the most headaches have been Logstash filters. One of the most complex parts, besides from the Logstash filters, has been the entire analysis of information in order to get these accurate results. However, it is not a 100% reliable counting system, just because if a person does not have a device with its Wi-Fi activated it won't be counted.

It has to be said that we would like to get the export of accumulated data from Elasticsearch to a relational database such as MySQL. That way, we could have developed our own data visualization tool, since giving access to Kibana to a client might be dangerous as it can modify Elasticsearch parameters. Moreover we could have shown all metrics that customer has asked.

The data visualization platform is already being developed. MySQL database will be shared with the current database of SocialWifi S.L., and, as discussed at the beginning, it will be possible to match these devices with the users who have accessed Wi-Fi through Social Wifi, and provide more valuable information to the client.

The work environment to develop has been very pleasant since I had all the facilities to work with. In addition, I have been talking with Mikrotik's technical support that has helped me to understand the operations and data provided by the snooper of these APs.

Annex A. Installation & Configuration

This section explains the installation process in detail. It starts from the base on which CentOS server is already installed and configured for an Internet access.

A.1 Prerequisites

In order to install and configure everything without any trouble, you will require root access on CentOS.

Elasticsearch and Logstash require Java, so it has to be installed before. Elasticsearch recommends to have installed the latest version.

A.2 ElasticSearch

Elasticsearch can easily be installed by adding Elastic's package repository. First, you have to create a new file in directory */etc/yum.repos.d/elasticsearch.repo* using nano command so you can add the following code, and then proceed to install ElasticSearch:

```
[elasticsearch-5.x]

name=Elasticsearch repository for 5.x packages

baseurl=https://artifacts.elastic.co/packages/5.x/yum

gpgcheck=1

gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch

enabled=1

autorefresh=1

type=rpm-md

yum install elasticsearch
```

When installation is complete, create a new service named *elasticsearch.service*, so it can be started and stopped anytime by referencing service's name.

First, you have to reload *systemd* by scanning for new or changed units, and then, enable Elasticsearch's service to start automatically on boot up:

```
systemctl daemon-reload

systemctl enable elasticsearch.service
```

Once installed, edit the configuration by restricting outside access to ElasticSearch instance (Port 9200), so outsiders can't read data or shutdown ElasticSearch cluster through the HTTP API.

```
nano /etc/elasticsearch/elasticsearch.yml
```

To do so, just uncomment the line that specifies *network.host* and replace its value with *"localhost".*

```
network.host: localhost
```

It is recommended to uncomment the line that says *bootstrat.memory_lock: true* because when JVM does a major garbage collection it touches every page of the heap. If any of those pages are swapped out to disk they will have to be swapped back in to memory. That causes lots of disk thrashing that ElasticSearch would much rather use to service requests.

Now, we are ready to start the service

```
systemctl start elasticsearch.service
```

To test that it works, you can do a curl call to ElasticSearch:

```
Curl localhost:9200
```

A.3 Kibana

Kibana package shares the same GPG Key as ElasticSearch, and we already installed that public key.

The previous step before installing Kibana is the same as for ElasticSearch, create and edit a new yum repository file for Kibana:

```
Nano /etc/yum.repos.d/elasticsearch.repo
```

```
[kibana-5.x]

name=Kibana repository for 5.x packages

baseurl=https://artifacts.elastic.co/packages/5.x/yum

gpgcheck=1

gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch

enabled=1

autorefresh=1

type=rpm-md
```

```
yum install kibana
```

When the installation process is done, edit the configuration file; in this case uncomment the lines:

```
nano /etc/kibana/kibana.yml

server.port: 5601

server.host: "localhost"
```

```
elasticsearch.url: http://localhost:9200

kibana.index: ".kibana"
```

Once edited the configuration file, enable kibana.service and start it.

```
systemctl daemon-reload

systemctl enable kibana.service

systemctl start kibana.service
```

A.4 NGINX

Before you can use the Kibana web interface, you have to set up a reverse proxy because Kibana is configured to listen on localhost. Nginx will be configured as a reverse proxy to allow external access to Kibana.

To proceed, add the EPEL repository to yum

```
yum install epel-release
```

Now, you are able to install nginx and httpd-tools

```
yum install nginx httpd-tools
```

Once installed, create a new user in order to access the Kibana web interface

```
htpasswd -c /etc/nginx/htpasswd.users jordi
```

Now, you have to edit the configuration file of *nginx* and find the default server block, which starts with *"**server {"*** and is the last configuration block of the file, and delete it. When you are done, the last two lines in the file should look like this:

```
nano /etc/nginx/nginx.conf

    include /etc/nginx/conf.d/*.conf;

}
```

Now you can create an nginx server block in a new file so it is better structured and easy to find if you want to change it anytime.

```
Nano /etc/nginx/conf.d/kibana.conf



server {
   listen 80;
   server_name localhost;
   auth_basic "Restricted Access";
   auth_basic_user_file /etc/nginx/htpasswd.users;


   location / {
      proxy_pass http://localhost:5601;
      proxy_http_version 1.1;
      proxy_set_header Upgrade $http_upgrade;
      proxy_set_header Connection 'upgrade';
      proxy_set_header Host $host;
      proxy_cache_bypass $http_upgrade;
   }
}
```

In case SELinux is enabled you have to run the command:

```
sudo setsebool -P httpd_can_network_connect 1
```

Selinux is a security module for Linux kernel that provides a mechanism to support security policies for control access.

At this point, reverse proxy is properly installed and configured and service can be enabled and started.

```
systemctl daemon-reload

Systemctl enable nginx.service

Systemctl start nginx.service
```

A.5 Logstash

The first thing you have to do, as previously, is to configure a new repository file for Logstash.

```
Nano /etc/yum.repos.d/logstash.repo
```

```
[logstash-5.x]

name=Elastic repository for 5.x packages

baseurl=https://artifacts.elastic.co/packages/5.x/yum

gpgcheck=1

gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch

enabled=1

autorefresh=1

type=rpm-md
```

Now you can start with installation process. When finished, reload daemons, enable and start the service.

```
Yum install logstash

systemctl daemon-reload

systemctl enable logstash.service

systemctl start logstash.service
```

Now, we have to set up logstash service to be executed by root user:

```
Nano /etc/systemd/system/logstash.service

        User=root

        Group=root

        ExecReload=/bin/kill -SIGHUP $MAINPID
```

```
systemctl restart logstash
```

At this moment, Elastic pack is installed at the server and ready to process data.

Bibliography

1. **Social Marketing Intelligence**. (2017). *Socialwibox - Social Marketing Intelligence.* [online] Available at: http://www.socialwibox.com [Accessed 5 Mar. 2017].
2. **Wiki.mikrotik.com**. (2017). *Manual:Interface/Wireless - MikroTik Wiki.* [online] Available at: https://wiki.mikrotik.com/wiki/Manual:Interface/Wireless#Snooper [Accessed 21 Apr 2017].
3. **Elastic.co**. (2017). *Logstash: Collect, Parse, Transform Logs | Elastic*. [online] Available at: https://www.elastic.co/products/logstash [Accessed 21 Apr 2017].
4. **Elastic.co**. (2017). *Filebeat: Lightweight Log Analysis & Elasticsearch | Elastic*. [online] Available at: https://www.elastic.co/products/beats/filebeat [Accessed 21 Apr. 2017].
5. **Sematext.com.** (2017). *Logagent.* [online] Available at: https://sematext.com/logagent/ [Accessed 21 Apr. 2017].
6. **Elastic.co.** (2017). *Elasticsearch: RESTful, Distributed Search & Analytics | Elastic*. [online] Available at: https://www.elastic.co/products/elasticsearch [Accessed 21 Apr. 2017].
7. **Elastic.co.** (2017). *Kibana: Explore, Visualize, Discover Data | Elastic*. [online] Available at: https://www.elastic.co/products/kibana [Accessed 21 Apr. 2017].
8. **Kafka-apache.** (2017). *Apache Kafka – A distributed streaming platform*. [online] Available at: https://kafka.apache.org/ [Accessed 21 Apr. 2017].
9. **Redis.io**. (2017). *Redis*. [online] Available at: https://redis.io/ [Accessed 24 Apr. 2017].
10. **Sematext.com**. (2017). *Logsene*. [online] Available at: https://sematext.com/logsene/ [Accessed 23 May 2017].
11. **Sematext.com**. (2017). *Docker Monitoring & Logging*. [online] Available at: https://sematext.com/docker/ [Accessed 24 Apr. 2017].
12. **MongoDB**. (2017). *Relational Vs Non Relational Database*. [online] Available at: https://www.mongodb.com/scale/relational-vs-non-relational-database [Accessed 25 Apr. 2017].

13. **MongoDB**. (2017). What Is MongoDB?. [online] Available at: https://www.mongodb.com/what-is-mongodb [Accessed 28 Apr. 2017].

14. **Cassandra.apache.org**. (2017). *Apache Cassandra*. [online] Available at: http://cassandra.apache.org/ [Accessed 30 Apr. 2017].

15. **Db-engines.com**. (2017). *DB-Engines Ranking - popularity ranking of database management systems*. [online] Available at: https://db-engines.com/en/ranking [Accessed 15 Mar. 2017].

16. **Lucene.apache.org**. (2017). *Apache Lucene - Apache Lucene Core*. [online] Available at: https://lucene.apache.org/core/ [Accessed 18 Mar. 2017].

17. **Sumo Logic**. (2017). How the Sumo Logic Platform Works - Sumo Logic. [online] Available at: https://www.sumologic.com/how-it-works/ [Accessed 5 Apr. 2017].

18. **Loggly**. (2017). *Log Management | Loggly*. [online] Available at: https://www.loggly.com/ [Accessed 9 Apr. 2017].

19. **O'reilly**. (2012*). What is DevOps?*. [online] Available at: http://radar.oreilly.com/2012/06/what-is-devops.html [Accessed 14 Apr. 2017].

20. **Routerboard.com**. (2017). *RouterBoard.com : RB951Ui-2HnD*. [online] Available at: https://routerboard.com/RB951Ui-2HnD [Accessed 12 Mar. 2017].

21. **Routerboard.com.** (2017). *RouterBoard.com : wAP ac*. [online] Available at: https://routerboard.com/RBwAPG-5HacT2HnD [Accessed 10 Mar. 2017].

22. **Open-mesh.com**. (2017). *OM2P 150 Mbps Access Point with External Antenna - Access Points - Store*. [online] Available at: http://www.open-mesh.com/products/access-points/grp-om2p.html [Accessed 12 Mar. 2017].

23. **Mikrotik.com**. (2017). *Winbox*. [online] Available at: https://mikrotik.com/download [Accessed 5 Feb. 2017].

24. **CloudTrax**. (2017). *Part 1: CloudTrax Guide Overview*. [online] Available at: https://help.cloudtrax.com/hc/en-us/articles/202465650-Part-1-CloudTrax-Guide-Overview [Accessed 21 Apr. 2017].

25. **Wiki.mikrotik.com.** (2017). *Manual:Interface/Wireless - MikroTik Wiki*. [online] Available at: https://wiki.mikrotik.com/wiki/Manual:Interface/Wireless#Nstreme [Accessed 8 Mar. 2017].

26. **CloudTrax**. (2017). *CloudTrax Presence Reporting API*. [online] Available at: https://help.cloudtrax.com/hc/en-us/articles/207985916-CloudTrax-Presence-Reporting-API [Accessed 11 Apr. 2017].

27. **Ece.uvic.ca**. (2017). *Derivation the dB version of the Path Loss Equation for Free Space.*. [online] Available at: http://www.ece.uvic.ca/~peterd/35001/ass1a/node1.html [Accessed 13 May 2017].

28. **Radio-electronics.com**. (2017). *Radio Signal Path Loss :: Radio-Electronics.Com*. [online] Available at: http://www.radio-electronics.com/info/propagation/path-loss/rf-signal-loss-tutorial.php [Accessed 14 May 2017].

29. ***Wi-fi tracking system dashboard***. *(2017).* http://88.99.160.236/app/kibana#/dashboard/53c26720-43f4-11e7-bf37-07b61b6393ca?_g=(refreshInterval%3A(display%3AOff%2Cpause%3A!f%2Cvalue%3A0)%2Ctime%3A(from%3Anow-7d%2Cmode%3Aquick%2Cto%3Anow))