

# Trabajo Fin de Grado

## Ingeniería de Telecomunicación

### Estudio de las tramas Probe Request y su uso como identificador único de dispositivos WiFi

Autora: Celia Gómez Megías

Tutores: Sergio Luis Toral Marín y Pablo Aguilera Bonet

**Dpto. de Electrónica**  
**Escuela Técnica Superior de Ingeniería**  
**Universidad de Sevilla**

Sevilla, 2020





Trabajo Fin de Grado  
Ingeniería de Telecomunicación

# **Estudio de las tramas Probe Request y su uso como identificador único de dispositivos WiFi**

Autora:

Celia Gómez Megías

Tutores:

Sergio Luis Toral Marín

Catedrático de Universidad

Pablo Aguilera Bonet

Profesor Colaborador

Dpto. de Electrónica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2020



Trabajo Fin de Grado: Estudio de las tramas Probe Request y su uso como identificador único de dispositivos WiFi

Autora: Celia Gómez Megías

Tutores: Sergio Luis Toral Marín y Pablo Aguilera Bonet

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2020

El Secretario del Tribunal



*A mi familia*

*A mí misma*





# Agradecimientos

---

Tras muchos meses de trabajo, esfuerzo y adversidades, finalmente he terminado este proyecto. Este trabajo supone el final de esta etapa de Universidad y da comienzo a una nueva. A lo largo de estos años de estudio siento que he evolucionado mucho a todos los niveles, pero esto no habría sido posible sin todas esas personas que han estado a mi lado.

Las primeras personas a las que les debo esto son mis padres, quienes me han brindado la oportunidad de estar hoy aquí, escribiendo estas palabras y sintiéndome un poco más cerca de mis metas. Tengo una familia maravillosa y no podría estar más agradecida de tenerles. Se lo debo todo y el trabajo de todos estos años va por ellos.

Justo después está mi hermana, esa persona con la que he compartido toda mi vida y todos estos años de convivencia. Ella, que me ha aguantado en todo momento, que me ha animado y respaldado siempre. No siempre hemos sabido encontrarnos, pero sin duda ha sido y es uno de mis mejores apoyos en todo este proceso.

Agradezco a todas esas personas que han estado estos años en mi camino, las que han estado de paso y las que aún siguen aquí. Han sido años de crecimiento y tanto las buenas experiencias como las malas han sido cruciales para ser la persona que soy hoy. Sin duda, aprecio mucho más a todas esas personas que están a mi lado y que, de una forma o de otra, me han demostrado su apoyo. Las tardes de estudio, las prácticas, los cafés, los aprobados, los suspensos, los ratos de cantar, de hacer música, de bailar, de comer, de juegos de mesa, los 4 de marzo en el parque del Alamillo... Todo esto me ha dado la fuerza de estar aquí hoy.

También quiero agradecer a mis profesores su paciencia conmigo, su ayuda en los momentos indicados y su guía, especialmente a lo largo de estos meses. He aprendido mucho trabajando con el Doctor Pablo Aguilera y me llevo una gran experiencia de este gran profesional.

Por último, quiero agradecer a Jesús, mi psicólogo, por su paciencia y calma conmigo en estos meses que tan difíciles han sido para mí. Me ha conducido a lo largo de rincones muy oscuros ayudándome a conocerme mejor y tener menos miedo de mí misma. Estar hoy en este momento probablemente no habría sido posible o, sin duda, habría sido muchísimo más difícil, es por eso que no puede faltar esta mención.

Siento que debo estar agradecida también a mí misma. A veces no siempre confiamos en nosotros mismos tanto como nos gustaría. Nos surgen dudas, miedos, inseguridades y es muy fácil tirar la toalla y dejar las cosas estar. A lo largo de estos años he aprendido a tomar esos caminos llenos de incertidumbre y darles luz y fuerza al trabajo que hago. He aprendido la valía de mi trabajo y que, para saber si podemos con algo, siempre hay que intentarlo.

Me llevo una experiencia inolvidable y este trabajo es fruto de ello.

*Celia Gómez Megías*

*Sevilla, 2020*



# Resumen

---

Este proyecto aborda el estudio de las tramas WiFi del subtipo “Probe Request” y su utilidad a la hora de contar dispositivos en las inmediaciones de una red WiFi que esté a la escucha. Estas tramas las emiten los dispositivos móviles (smartphones, tablets, ordenadores portátiles) de forma periódica, incluso cuando no están conectados ni asociados con ninguna red WiFi. El sistema propuesto monitoriza estas tramas a su alrededor, por uno o varios puntos de acceso o dongles WiFi, y luego procesa los campos internos de dichas tramas a fin de evitar contar más dispositivos de los que realmente hay. Este sobre-recuento se suele producir en sistemas que no post-procesan las tramas “Probe Request”, ya que interpretan cada trama con una dirección MAC desconocida que le llega como un nuevo dispositivo.

La dirección MAC (específicamente la dirección MAC origen) es para un dispositivo WiFi como la matrícula para un coche: lo identifica unívocamente respecto del resto. No obstante, en los últimos años hemos observado cómo los fabricantes de dispositivos WiFi modernos (Apple, Android, Windows, etc) falsifican (o aleatorizan) estas direcciones MAC en las tramas “Probe Request”, con el objetivo de engañar o perturbar a los sistemas de recuento de dispositivos en un entorno. Esto es legítimo por parte de los fabricantes, pero también es legítimo por parte de los investigadores del campo de las comunicaciones inalámbricas el indagar en este comportamiento a fin de hacer sistemas de recuento fiables y robustos (siempre dentro de las normativas europeas u nacionales sobre protección de datos).

Para ello, el sistema consiste en analizar los campos de la trama “Probe Request”, olvidándose de la dirección MAC como identificador fiable (pues sabemos que se falsea o aleatoriza) y atendiendo a otros campos que sí permanecen constantes a lo largo de distintas tramas del mismo dispositivo. En la analogía de los coches, sería como obviar la matrícula (pues puede haber sido manipulada) para fijarnos en el color, la marca, modelo, kilometraje, averías o imperfecciones de la carrocería del coche para identificarlo de forma unívoca. De esta forma se genera una fingerprint o huella dactilar de cada dispositivo, independientemente de la dirección MAC que haya exhibido en cada trama recibida, y que otorga recuentos más fiables que simplemente mirando la dirección MAC que el sistema ha escuchado.

Se ha probado el sistema de recuento en distintos escenarios urbanos, escuchando tramas “Probe Request” de dispositivos móviles reales que pasaban cerca del sensor. Se permite el análisis del tráfico en tiempo real con los datos capturados, y también en diferido con datos capturados anteriormente y almacenados en un archivo. Aun así, el sistema tiene sus limitaciones, particularmente cuando hay varios dispositivos iguales en las inmediaciones de la red: parecerán el mismo dispositivo a ojos de nuestro sistema. También se propondrán posibles mejoras futuras para mejorar la precisión, especialmente en el caso comentado de que haya varios dispositivos iguales de forma concurrente.



# Abstract

---

This project addresses the study of WiFi frames of the "Probe Request" subtype and their usefulness in counting devices in the vicinity of a listening WiFi network. These frames are issued by mobile devices (smartphones, tablets, laptops) on a regular basis, even when they are not connected or associated with any WiFi network. The proposed system monitors these frames around it, by one or more access points or WiFi dongles, and then processes the internal fields of these frames in order to avoid counting more devices than there really are. This over-count usually occurs on systems that do not post-process the "Probe Request" frames, since they interpret each frame with an unknown MAC address that arrives as a new device.

The MAC address (specifically the source MAC address) is for a WiFi device like the license plate for a car: it uniquely identifies it regarding to the rest. However, in recent years we have observed how the manufacturers of modern WiFi devices (Apple, Android, Windows, etc.) falsify (or randomize) these MAC addresses in the "Probe Request" frames, with the aim of deceiving or disturbing the device counting systems in an environment. This is legitimate on the part of manufacturers, but it is also legitimate on the part of researchers in the field of wireless communications to inquire into this behavior in order to make counting systems reliable and robust.

For this, the system consists of analyzing the fields of the "Probe Request" frame, forgetting the MAC address as a reliable identifier (since we know that it is falsified or randomized) and attending to other fields that do remain constant throughout different frames from the same device. In the analogy of cars, it would be like ignoring the license plate (as it may have been manipulated) focus on at the color, brand, model, mileage, breakdowns or imperfections of the car body to uniquely identify it. In this way, a fingerprint of each device is generated, regardless of the MAC address that has been displayed in each frame received, and which provides more reliable counts than simply looking at the MAC address that the system has listened to.

The counting system has been tested in different urban settings, listening to "Probe Request" frames from real mobile devices that passed near the sensor. Traffic analysis is allowed in real time with the captured data, and also delayed with previously captured data stored in a file. Even so, the system has its limitations, particularly when there are several similar devices in the immediate vicinity of the network: they will appear to be the same device in the eyes of our system. Possible future enhancements to improve accuracy will also be proposed, especially in the commented case that multiple devices are concurrently the same.



<b>Agradecimientos</b>	<b>9</b>
<b>Resumen</b>	<b>11</b>
<b>Abstract</b>	<b>13</b>
<b>Índice</b>	<b>15</b>
<b>Índice de Figuras</b>	<b>17</b>
<b>Índice de Tablas</b>	<b>19</b>
<b>1. Revisión del estado del arte</b>	<b>21</b>
1.1. Mecanismos de descubrimiento de redes WiFi	21
1.2. Propuestas similares ya existentes	22
1.3. Objetivo	25
1.4. Privacidad y protección de datos	27
<b>2. Descripción genérica del sistema</b>	<b>29</b>
2.1. Puesta en marcha del entorno	29
2.1.1. Sistema operativo	29
2.1.2. PyCharm	29
2.1.3. Python	34
2.1.4. Scapy	35
2.1.4.1. Netaddr	36
2.1.5. Interfaces	36
2.1.6. Wireshark	36
2.2. Introducción a Scapy	37
2.3. Visión general	38
<b>3. Implementación</b>	<b>39</b>
3.1. Diagramas de flujo del sistema	39
3.1.1. Puesta en marcha	39
3.1.2. Fichero de comprobación	40
3.1.3. Fichero principal	41
3.1.4. Función almacen()	42
3.1.5. Función es_verdadera()	43
3.2. Desarrollo y enfoque del código	44

<b>4. Experimentos</b>	<b>45</b>
4.1. Fingerprint	45
4.2. Casos	47
4.2.1.Lecturas 2013	47
4.2.2.Restaurante Salón Romero, Zafra	50
4.2.3.Centro comercial Torre Sevilla	51
4.2.4.Edificio de viviendas Sevilla	52
4.2.5.Centro Sevilla	53
4.2.6.Centro comercial Nervión Plaza, Sevilla	54
4.3. Análisis y comparativa de casos	55
4.3.1.Proporción de la cantidad de probes procedentes de dispositivos que ofrecen MAC verdadera y MAC falsa de cada caso	55
4.3.2.Porcentaje de usuarios detectados frente a la ubicación donde se realizó la lectura	56
4.3.3.Porcentaje de usuarios detectados frente al tiempo que duró la lectura	57
4.3.4.Cantidad de usuarios calculados con ratio frente a los calculados con el script	58
<b>5. Conclusiones y líneas futuras</b>	<b>59</b>
5.1. Conclusiones	59
5.2. Líneas futuras	59
<b>Referencias</b>	<b>61</b>
<b>Glosario</b>	<b>63</b>
<b>ANEXO A: Campos de Probe Request</b>	<b>65</b>



# Índice de Figuras

---

Figura 1. Probe Request Frame.	22
Figura 2. Association Frame.	23
Figura 3. Conexión WiFi. Intercambio de mensajes.	23
Figura 4. OUI.	24
Figura 5. Cisco OUI.	24
Figura 6. Galgus OUI.	24
Figura 7. Captura Wireshark destacando el tipo de trama y los elementos de las direcciones MAC falsas.	25
Figura 8. Estructura de Probe Request Frame obtenida mediante observación de tráfico.	26
Figura 9. Download PyCharm.	30
Figura 10. Extracción PyCharm.	30
Figura 11. Apertura terminal.	31
Figura 12. User Agreement.	31
Figura 13. PyCharm License Activation.	32
Figura 14. Create New Project.	32
Figura 15. Hello World.	33
Figura 16. Hello World 2.	33
Figura 17. Hello World 3.	34
Figura 18. Hello World 4.	34
Figura 19. Scapy GitHub.	35
Figura 20. Scapy en nuestro proyecto.	35
Figura 21. TP-LINK Archer T4U.	36
Figura 22. Código activador de interfaz.	36
Figura 23. Diagrama puesta en marcha.	39
Figura 24. Diagrama comprobación.	40
Figura 25. Diagrama fichero principal.	41
Figura 26. Diagrama almacén.	42
Figura 27. Diagrama es_verdadera.	43
Figura 28. Comparación entre las 5 fingerprints iniciales con la cantidad de dispositivos que detectan respecto al fichero de comprobación.	45
Figura 29. Comparación entre las 4 fingerprints con la mejor cifra de dispositivos detectados respecto al fichero de comprobación.	46

Figura 30. Comparación entre las 3 fingerprints con la mejor cifra de dispositivos detectos respecto al fichero de comprobación.	46
Figura 31. Antigua 1.	47
Figura 32. Antigua 2.	48
Figura 33. Antigua 3	48
Figuta 34. Antigua 4.	49
Figura 35. Restaurante Salón Romero, Zafra.	50
Figura 36. Captura Restaurante Salón Romero, Zafra.	50
Figura 37. Centro comercial Torre Sevilla.	51
Figura 38. Captura centro comercial Torre Sevilla.	51
Figura 39. Viviendas en calle Arjona, Sevilla.	52
Figura 40. Captura viviendas en calle Arjona, Sevilla.	52
Figura 41. Avenida Reyes Católicos, Centro Sevilla.	53
Figura 42. Captura avenida Reyes Católicos, Centro Sevilla.	53
Figura 43. Centro comercial Nervión Plaza, Sevilla.	54
Figura 44. Captura centro comercial Nervión Plaza, Sevilla.	54
Figura 45. Balance de MACs verdaderas y falsas recibidas.	55
Figura 46. Comparación la cantidad de dispositivos detectados respecto al fichero de comprobación frente a la ubicación donde se tomaron las muestras.	56
Figura 47. Comparación la cantidad de dispositivos detectados respecto al fichero de comprobación frente al tiempo que duró la toma de muestras.	57
Figura 48. Comparación la cantidad de dispositivos detectados por nuestro scrip y por el fichero de comprobación en cada ubicación donde se tomaron muestras.	58

# Índice de Tablas

---

Tabla 1. Datos obtenidos de las lecturas.	55
Tabla 2. Campos Probe Request.	65



# 1 REVISIÓN DEL ESTADO DEL ARTE

---

“El camino del progreso no es ni rápido ni fácil.”

- Marie Curie -

## 1.1 Mecanismos de descubrimiento de redes WiFi

Una red WiFi es aquella formada por puntos de acceso (APs – Access Points) y estaciones cliente (STAs – stations), pudiendo ser estas últimas smartphones, tablets, ordenadores portátiles, etc. Cuando las estaciones o dispositivos clientes están conectadas a los APs, éstas tienen acceso a Internet de forma inalámbrica. Para que la comunicación se produzca con éxito, el estándar 802.11 (en el que se basa la tecnología WiFi) propone tres tipos de tramas:

- Tramas de datos: que portan la información de usuario de capas superiores. Por ejemplo un vídeo, un mensaje de texto, etc.
- Tramas de control: que ayudan a que el mensaje llegue correctamente. Por ejemplo, asentimientos, petición de reserva del canal, etc.
- Tramas de gestión: que protocolizan la conexión de nuevos dispositivos. Por ejemplo, balizas que ayudan a descubrir las redes cercanas, control de asociación, etc.

La trama “Probe Request” es una trama del tercer tipo (gestión), y básicamente se utiliza para que un dispositivo móvil pida a las redes WiFi cercanas que está buscando conexión. Al recibirlas, un punto de acceso habitualmente responde con sus capacidades por si el dispositivo de usuario está de acuerdo en establecer una conexión con él y así tener acceso a Internet.

Esta trama, en mayor o menor medida, es emitida por todos los dispositivos móviles, aunque su comportamiento depende de muchas circunstancias. Hay dispositivos que solo las emiten cuando se están quedando sin cobertura, o que dejan de emitirlas si tienen poca batería. También se observan comportamientos distintos dependiendo del fabricante o la versión del sistema operativo, e incluso dependiendo de si tienen activado el botón WiFi o GPS de su panel de opciones (incluso se han visto en modo avión). Esta es una parte de la red WiFi que no podemos controlar, pero sí podemos diseñar un sistema que acepte toda esta casuística y explote la información que le llega a fin de implementar un método de recuento más robusto que simplemente “contar direcciones MAC”, pues como hemos visto son aleatorizadas por el fabricante.

## 1.2 Propuestas parecidas ya existentes

Al inicio de este proyecto se han realizado diferentes búsquedas en busca de estudios relacionados con la detección de dispositivos WiFi no conectados a una red mediante otros métodos que no sea identificarlos con su dirección MAC. Algunos de ellos hablan sobre el uso de cierto tipo de tramas emitidas por los dispositivos, otros se centran en los elementos que más pueden ayudarnos a en la identificación. Aquí se recojen los elementos más importantes:

### ➤ Probe Requests [1][2].

Primero definiremos qué son las Probe Requests. Estas son tramas enviadas por los dispositivos de forma automática cada cierto intervalo de tiempo y en rachas de varias tramas a la vez, siendo configurables el intervalo de tiempo y el número de tramas por intervalo. Estas tramas se utilizan para buscar puntos de acceso a los que conectarse, ya sea uno en específico o cualquiera disponible. Con estas tramas también se solicita información del punto de acceso. Al fin y al cabo, todo orientado para acelerar el proceso de conexión WiFi.

Las Probe Request son especialmente interesantes porque:

- Son las únicas tramas que emite el dispositivo **sin estar conectado** a un AP.
- Son las únicas tramas que emite un dispositivo **en todos los canales**.

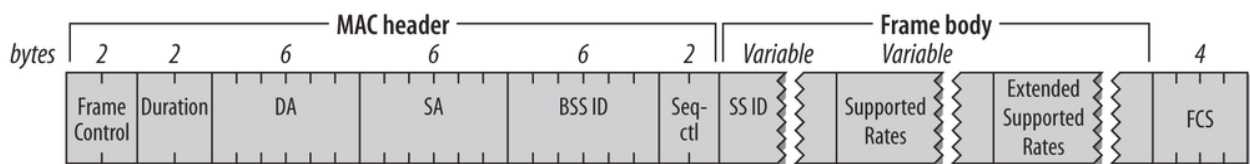


Figura 1. Probe Request Frame

Los campos de la trama están detallados en el Anexo A al final de este documento.

Algunos de estos campos son **estáticos**, es decir, no cambian entre tramas del mismo dispositivo. Estos son especialmente interesantes pues nos permiten identificar a un dispositivo incluso aunque falsee su dirección MAC. En la analogía del coche, sería la marca y modelo del coche o el número de puertas.

Otros campos son **dinámicos**, es decir, que cambian entre distintas tramas del mismo dispositivo. No debemos fiarnos de estos campos, pues podrían pertenecer al mismo dispositivo o a otro parecido (ya hemos visto qué sucede con las direcciones MAC aleatorias), con una excepción: aquellos campos dinámicos pero que exhiben un patrón de comportamiento que podemos explotar. En la analogía del coche, estos campos dinámicos que siguen un patrón serían el kilometraje o la cantidad de combustible que le queda en el depósito.

En este proyecto nos centraremos en los campos estáticos, pues son más sencillos de tratar, y dejaremos los dinámicos como línea de investigación abierta para trabajos futuros.

En este caso, ha sido llamativo el uso de las Probe Request porque en los diversos estudios encontrados es el tipo de trama con mayor información para crear una firma del dispositivo. Estas tramas contienen campos como la dirección MAC del dispositivo, sus SSIDs preferidos o anteriormente registrados, la tasa de envío y diversas características del dispositivo entre muchos otros.

Es por ello que son las tramas más utilizadas para este tipo de investigaciones.

➤ **Association frames** [1].

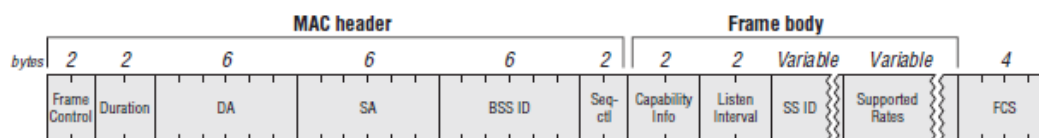


Figura 2. Association Frame

Este tipo de tramas también son interesantes porque nos aportan también información del dispositivo, aunque en una **fase más avanzada de la conexión**, puesto que primero hay que pasar la parte de la autenticación en caso de que la hubiera. Existen trabajos que utilizan esta trama para aportar información adicional a la fingerprint o huella dactilar, pero en este proyecto **no las utilizaremos** ya que queremos contar dispositivos que no tienen intención de conectarse a la red WiFi: simplemente pasan cerca de nuestros nodos de escucha.

Lo más interesante encontrado en estas tramas es que ofrece información sobre diversas capacidades del cliente cuando envía la solicitud al AP.

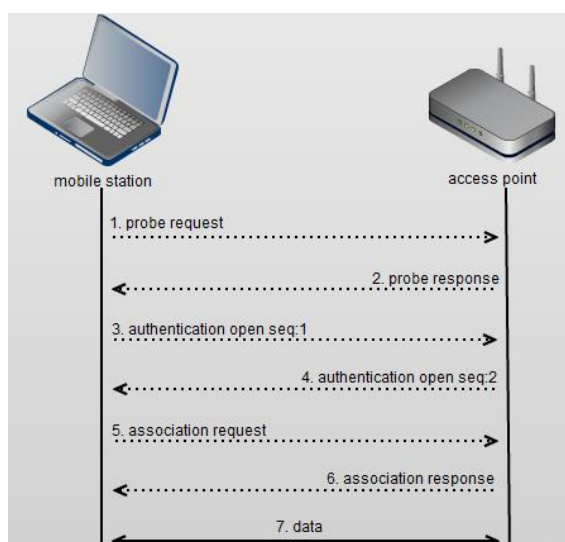


Figura 3. Conexión WiFi. Intercambio de mensajes.

- Los dispositivos que usan el **mismo chipset WiFi pero diferente software** tendrán similitudes en sus firmas, pero a menudo tienen suficientes diferencias que les permiten distinguirse [2]. Esto es debido a que podemos encontrar dispositivos con Sistemas Operativos diversos como Android o iOS y puede suponer una diferencia o factor extra a la hora de identificar un dispositivo.
- El **OUI** (Organizationally Unique Identifier) [3].

Cada interfaz de red en un dispositivo con capacidad 802.11 tiene un identificador de hardware de capa 2 de dirección MAC de 48 bits. Las direcciones MAC están diseñadas para ser persistentes y globales. Para garantizar la unicidad de direcciones MAC en todos los dispositivos del Institute of Electrical and Electronics Engineers (IEEE) asigna bloques de direcciones a organizaciones a cambio de una tarifa. Un MAC Address Block Large (MA-L), comúnmente conocido como el Identificador Único Organizacional (OUI), puede ser comprado y registrado con el IEEE, que da el control de la organización y la responsabilidad de todos los anuncios se viste con un prefijo particular de tres bytes. El fabricante es libre de asignar el orden inferior restante, tres bytes, de cualquier valor que deseen al inicializar dispositivos, sujeto a la condición de que no usen la misma dirección MAC dos veces.

Una implicación del sistema de registro IEEE es que es trivial buscar el fabricante de un dispositivo dada su dirección MAC. Esto significa que cualquiera que escucha el tráfico al 802.11 puede determinar el fabricante de dispositivos cercanos.

Para combatir esto, el IEEE también proporciona la capacidad de comprar un OUI "privado" que no incluya el nombre de la empresa en el registro. Sin embargo, esta característica de privacidad adicional no es utilizada actualmente por los principales fabricantes que conocemos.

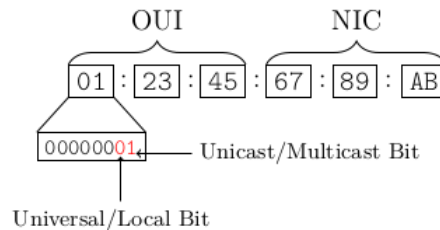


Figura 4. OUI

A día de hoy existen herramientas para detectar los fabricantes de los dispositivos a partir de la dirección MAC. Más concretamente a partir de su OUI. Un ejemplo de estas herramientas es <https://www.wireshark.org/tools/oui-lookup.html>.

Podemos ver ejemplos como los siguientes:

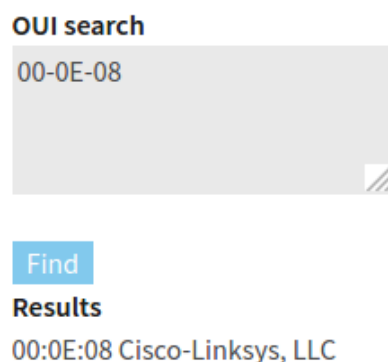


Figura 5. Cisco OUI

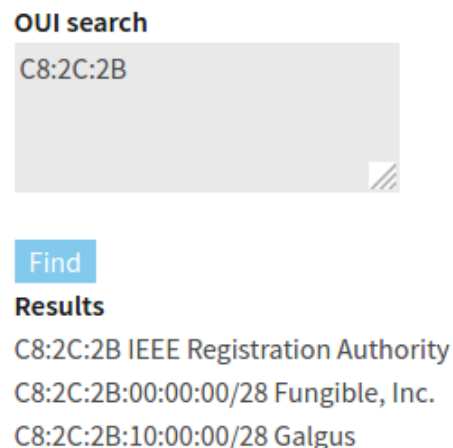


Figura 6. Galgus OUI

### ➤ Locally Assigned Addresses [3].

Además de la dirección MAC pública, globalmente única y asignada por el fabricante, los dispositivos modernos con frecuencia usan direcciones asignadas localmente que se distinguen por un **bit Universal / Local** en el byte más significativo. Las direcciones asignadas localmente no están garantizadas para ser únicas, y generalmente no se utilizan de forma persistente. Las direcciones asignadas localmente se utilizan en una variedad de contextos, incluido el SSID, la configuración de APS, puntos de conexión conectados a dispositivos móviles y servicios P2P. Una representación visual de la estructura de bytes de la dirección MAC se ilustra en la Figura 4.

Lo más importante de esta investigación es que, asignadas localmente las direcciones también se pueden usar para crear direcciones MAC aleatorias como una **medida adicional de privacidad**.

Este tipo de direcciones generadas de forma aleatoria a las que llamaremos “direcciones MAC falsas” dada su modificación, se caracterizan por tener como segundo carácter uno de éstos: {2, 6, A, E}. De esta forma podremos identificarlas. Ejemplos de direcciones MAC falsas son:



12:A3:56:25:0C, 2E:35:F6:00:E7.

La aleatorización de direcciones MAC es una técnica de privacidad mediante la cual los dispositivos móviles rotan a través de direcciones de hardware aleatorias para evitar que los observadores lean el tráfico o identifiquen la ubicación física de otros dispositivos cercanos. Sin embargo, la adopción de esta tecnología ha sido esporádica y variada entre los fabricantes de dispositivos.

Este es uno de los puntos más interesantes puesto que los dispositivos solo mostrarán su dirección física real cuando estén conectados a una red. Si no lo están habrá que tratar con estas direcciones y ver cómo podremos crear una firma de dichos dispositivos.

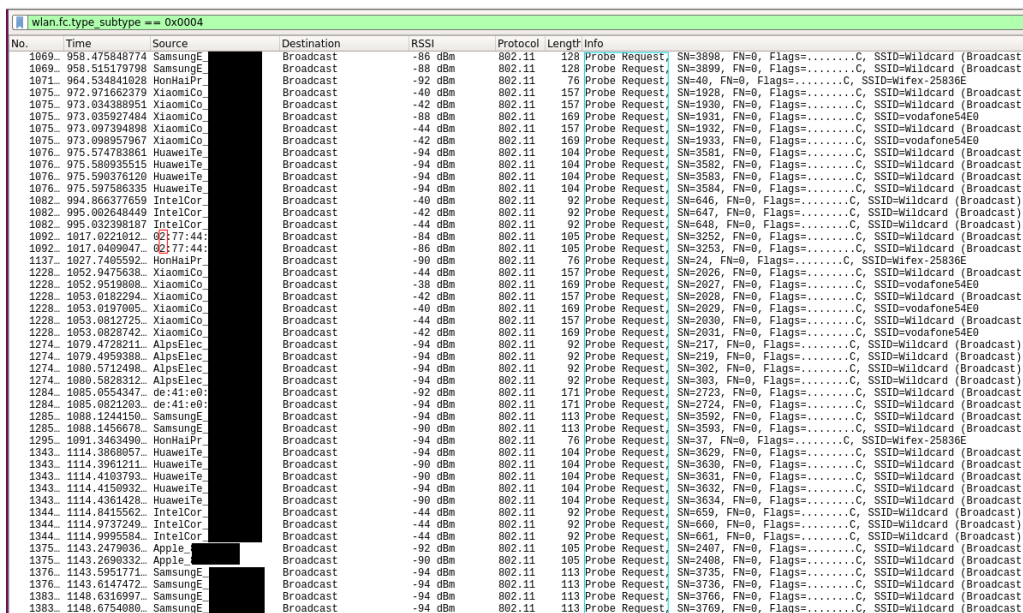
### 1.3 Objetivo

Consciente de que cada vez más dispositivos móviles falsean (o aleatorizan) su dirección MAC, el objetivo del proyecto es **crear un algoritmo que detecte unívoca los dispositivos WiFi no conectados a la red**, proporcionando con él estadísticas en función de los datos recogidos. Esto permitiría que en lugar de identificar 1 dispositivo por cada dirección MAC que recibimos (lo cual sólo sería válido si las direcciones MAC fuesen verdaderas), identificásemos 1 dispositivo sólo cuando estimamos que se trata de un dispositivo único (aunque exhiba muchas MACs diferentes en sus tramas). Para ello hemos tomado información de los estudios previos encontrados añadiéndoles algunas mejoras que se detallan a continuación.

Primero, tras de estudiar la documentación encontrada, para llevar a cabo este trabajo, junto con el Doctor Pablo Aguilera Bonet, se ha decidido utilizar la información proporcionada por:

- **Tramas Probe Request:** Nos proporcionan la información necesaria para obtener la información del dispositivo de forma pasiva, sin tener que estimularlo, y en la fase más temprana de la comunicación que es la búsqueda del AP.
- **OUI:** Dato que será usado como elemento que participa en la firma del dispositivo correspondiente.
- **Locally Assigned Addresses:** Dado que podemos encontrarlos con direcciones MAC falsas cuyo origen hemos descrito en el apartado anterior, dividiremos el estudio de nuestras firmas entre las direcciones MAC verdaderas y las falsas.

Una vez vista la información de partida, se llevará a cabo la monitorización, en diferentes escenarios y con diferentes dispositivos, capturando el tráfico inalámbrico con Wireshark y filtrando por tramas Probe Request:



No.	Time	Source	Destination	RSSI	Protocol	Length	Info
1069	958.475848774	SamsungE	Broadcast	-86 dBm	802.11	128	Probe Request, SN=3898, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1069	958.515179798	SamsungE	Broadcast	-88 dBm	802.11	128	Probe Request, SN=3899, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1071	964.534841928	HonHaiPr	Broadcast	-92 dBm	802.11	76	Probe Request, SN=40, FN=0, Flags=.....C, SSID=Wifex-25836E
1075	972.971662378	XiaomiCo	Broadcast	-40 dBm	802.11	157	Probe Request, SN=1928, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1075	973.934388954	XiaomiCo	Broadcast	-42 dBm	802.11	157	Probe Request, SN=1930, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1075	973.935927484	XiaomiCo	Broadcast	-88 dBm	802.11	169	Probe Request, SN=1931, FN=0, Flags=.....C, SSID=vodafone54E9
1075	973.997394898	XiaomiCo	Broadcast	-44 dBm	802.11	157	Probe Request, SN=1932, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1075	973.998957967	XiaomiCo	Broadcast	-42 dBm	802.11	169	Probe Request, SN=1933, FN=0, Flags=.....C, SSID=vodafone54E9
1076	975.574783861	HuaweiTe	Broadcast	-84 dBm	802.11	104	Probe Request, SN=3581, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1076	975.580838515	HuaweiTe	Broadcast	-94 dBm	802.11	104	Probe Request, SN=3582, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1076	975.590378128	HuaweiTe	Broadcast	-94 dBm	802.11	104	Probe Request, SN=3583, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1076	975.597586335	HuaweiTe	Broadcast	-94 dBm	802.11	104	Probe Request, SN=3584, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1082	994.886377659	IntelCor	Broadcast	-40 dBm	802.11	92	Probe Request, SN=646, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1082	995.802484440	IntelCor	Broadcast	-42 dBm	802.11	92	Probe Request, SN=647, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1082	995.832398187	IntelCor	Broadcast	-44 dBm	802.11	92	Probe Request, SN=648, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1092	1017.9221812	02:77:44:	Broadcast	-84 dBm	802.11	105	Probe Request, SN=3252, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1092	1017.9489947	02:77:44:	Broadcast	-86 dBm	802.11	105	Probe Request, SN=3253, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1137	1027.7486932	HonHaiPr	Broadcast	-90 dBm	802.11	76	Probe Request, SN=24, FN=0, Flags=.....C, SSID=Wifex-25836E
1228	1052.9475638	XiaomiCo	Broadcast	-44 dBm	802.11	157	Probe Request, SN=2026, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1228	1052.95198808	XiaomiCo	Broadcast	-38 dBm	802.11	169	Probe Request, SN=2027, FN=0, Flags=.....C, SSID=vodafone54E9
1228	1053.91922394	XiaomiCo	Broadcast	-42 dBm	802.11	157	Probe Request, SN=2028, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1228	1053.9197895	XiaomiCo	Broadcast	-40 dBm	802.11	169	Probe Request, SN=2029, FN=0, Flags=.....C, SSID=vodafone54E9
1228	1053.9812725	XiaomiCo	Broadcast	-44 dBm	802.11	157	Probe Request, SN=2030, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1228	1053.9828742	XiaomiCo	Broadcast	-42 dBm	802.11	169	Probe Request, SN=2031, FN=0, Flags=.....C, SSID=vodafone54E9
1274	1079.4728211	AlpsElec	Broadcast	-94 dBm	802.11	92	Probe Request, SN=217, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1274	1079.4958988	AlpsElec	Broadcast	-94 dBm	802.11	92	Probe Request, SN=219, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1274	1080.5712498	AlpsElec	Broadcast	-94 dBm	802.11	92	Probe Request, SN=302, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1274	1080.5828312	AlpsElec	Broadcast	-94 dBm	802.11	92	Probe Request, SN=303, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1284	1085.9554947	de:41:e0:	Broadcast	-92 dBm	802.11	171	Probe Request, SN=2723, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1284	1085.9821293	de:41:e0:	Broadcast	-94 dBm	802.11	171	Probe Request, SN=2724, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1285	1088.1244150	SamsungE	Broadcast	-94 dBm	802.11	113	Probe Request, SN=3592, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1285	1088.1456678	SamsungE	Broadcast	-90 dBm	802.11	113	Probe Request, SN=3593, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1295	1091.3483990	HonHaiPr	Broadcast	-94 dBm	802.11	76	Probe Request, SN=27, FN=0, Flags=.....C, SSID=Wifex-25836E
1343	1114.3888057	HuaweiTe	Broadcast	-94 dBm	802.11	104	Probe Request, SN=3629, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1343	1114.3961211	HuaweiTe	Broadcast	-90 dBm	802.11	104	Probe Request, SN=3630, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1343	1114.4103793	HuaweiTe	Broadcast	-90 dBm	802.11	104	Probe Request, SN=3631, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1343	1114.4158932	HuaweiTe	Broadcast	-94 dBm	802.11	104	Probe Request, SN=3632, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1343	1114.4361428	HuaweiTe	Broadcast	-90 dBm	802.11	104	Probe Request, SN=3634, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1344	1114.8415562	IntelCor	Broadcast	-44 dBm	802.11	92	Probe Request, SN=659, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1344	1114.9737249	IntelCor	Broadcast	-44 dBm	802.11	92	Probe Request, SN=660, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1344	1114.9905584	IntelCor	Broadcast	-94 dBm	802.11	92	Probe Request, SN=661, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1375	1143.2479836	Apple	Broadcast	-92 dBm	802.11	105	Probe Request, SN=2407, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1375	1143.2690332	Apple	Broadcast	-90 dBm	802.11	105	Probe Request, SN=2408, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1376	1143.5951771	SamsungE	Broadcast	-94 dBm	802.11	113	Probe Request, SN=3735, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1376	1143.6147472	SamsungE	Broadcast	-94 dBm	802.11	113	Probe Request, SN=3736, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1383	1148.6316997	SamsungE	Broadcast	-94 dBm	802.11	113	Probe Request, SN=3766, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)
1383	1148.6754880	SamsungE	Broadcast	-94 dBm	802.11	113	Probe Request, SN=3769, FN=0, Flags=.....C, SSID=Wildcard (Broadcast)

Figura 7. Captura Wireshark destacando el tipo de trama y los elementos de las direcciones MAC falsas.

Como vemos en la Figura 7 podemos apreciar que tenemos una dirección MAC falsa (recuadro rojo) y que estamos filtrando por trama Probe Request (recuadro azul). Además, de estas capturas obtenemos la siguiente estructura de trama:

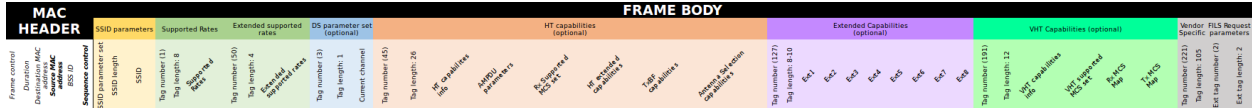


Figura 8. Estructura de Probe Request Frame obtenida mediante observación de tráfico.

Se tomarán algunas estadísticas por separado y otras en conjunto.

- Caso dirección MAC verdadera.
- Caso dirección MAC falsa: para comprobar si la dirección MAC es falsa se compararán los cuatro bits menos significativos del octeto más significativo con los valores {2, 6, A, C}. Si hay coincidencia, estaremos ante una dirección MAC falsa.

La firma estará compuesta por 3 elementos que son los siguientes:

- **Rates:** campo extraído de la trama Probe Request que contiene las tasas a las que transmite el dispositivo en cuestión.
- **HTCapabilities:** campo extraído de la trama Probe Request que contiene información de diversas características del dispositivo en cuestión.
- **Fabricante:** elemento extraído de la dirección MAC correspondiente a sus 3 octetos más significativos, cumpliendo así la normativa respecto a la privacidad y protección de datos del usuario (desarrollado en el siguiente apartado).

Además, se generará un identificador único utilizando una función hash() con los elementos de la firma como parámetro.

Al terminar de ejecutarse el script nos devuelve un reporte sobre la información recabada por los dispositivos y una serie de estadísticas.

- La información recogida corresponde a los elementos que conforman la firma del dispositivo.
- Las estadísticas generadas serán la siguiente lista:
  - Número total de probes
  - Número total de MACs
  - Media de probes por MAC
  - Número total de usuarios

## 1.4 Privacidad y protección de datos

La dirección MAC es una parte crucial de la comunicación WiFi, ya que se incluye en cada trama de la capa de enlace que se envía desde o hacia el dispositivo. Desafortunadamente, esto plantea un **problema de privacidad** evidente porque cualquier tercero que escuche el tráfico WiFi cercano puede identificar de manera única los dispositivos cercanos y su tráfico a través de sus direcciones MAC.

Ahondando un poco más en la **legislación europea** podemos comprobar que, efectivamente, la dirección MAC de un dispositivo es considerado un dato personal. El Artículo 4 de esa normativa define los "datos personales" de la siguiente manera:

*"Datos personales significa cualquier información relacionada con una persona física identificada o identificable ("sujeto de datos"); Una persona física identificable es aquella que puede ser identificada, directa o indirectamente, en particular por referencia a un identificador, como un nombre, un número de identificación, datos de ubicación o uno o más factores específicos a lo físico, fisiológico, identidad genética, mental, económica, cultural o social de esa persona física;*

Esta definición incluye una de estas dos opciones:

1. Cualquier información relacionada con una persona identificada (es decir, que hace que dicha información sea personal para esa persona).
2. Cualquier información relacionada con alguien que pueda identificarse en función de una variedad de identificadores.

La definición es relativamente sencilla para las personas "identificadas" pero se vuelve un poco confusa para las personas "identificables". Una persona puede ser identificable por medios directos o indirectos.

Cuanto más indirectos sean los identificadores, más dependerá de las circunstancias del entorno para determinar si la información califica como información personal protegida.

Los datos personales recogidos no tienen por qué identificar explícitamente a un individuo, pero en combinación podrían hacerlo.

Por lo tanto, es probable que las direcciones MAC del dispositivo califiquen de manera similar a las direcciones IP y por ello es considerado dato personal." [5]



# 2 DESCRIPCIÓN GENÉRICA DEL SISTEMA

---

“Lo importante es no tener arrugas en el cerebro.”

- Margarita Salas -

## 2.1 Puesta en marcha del entorno

Para llevar a cabo este proyecto, se utilizaron diversas herramientas y recursos: Python, el entorno de desarrollo PyCharm y la herramienta de manipulación de paquetes para redes Scapy. A continuación se explica paso a paso como configurar el entorno de desarrollo para poder ejecutar el proyecto dedicándose un apartado posterior a introducir Scapy.

### 2.1.1 Sistema operativo

Lo primero de todo es que partimos de un desarrollo en entorno Linux por la facilidad que proporciona a la hora de instalar paquetes, aplicaciones, dependencias, etc. y por su facilidad de uso aprendida a lo largo de estos años. Mas concretamente utilizaremos la distribución Ubuntu 16.04.1 .



No se asegura que en otros sistemas operativos o en entornos de virtualización pueda funcionar. En dichos casos es probable que haya que hacer ajustes o directamente no se pueda.

### 2.1.2 PyCharm

Para desarrollar el código del script usamos PyCharm que es uno de los entornos de desarrollo más populares para Python y que es bastante intuitivo y fácil de usar.

1. Accedemos a <https://www.jetbrains.com/pycharm/download/#section=linux> y descargamos la versión "Free Trial".

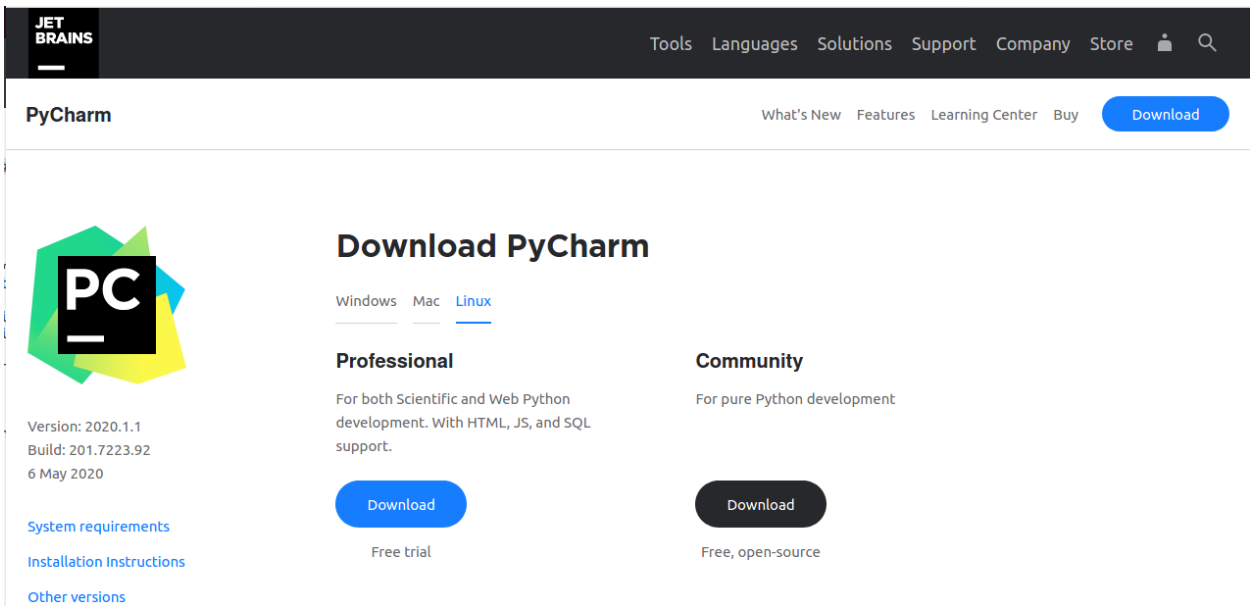


Figura 9. Download PyCharm.

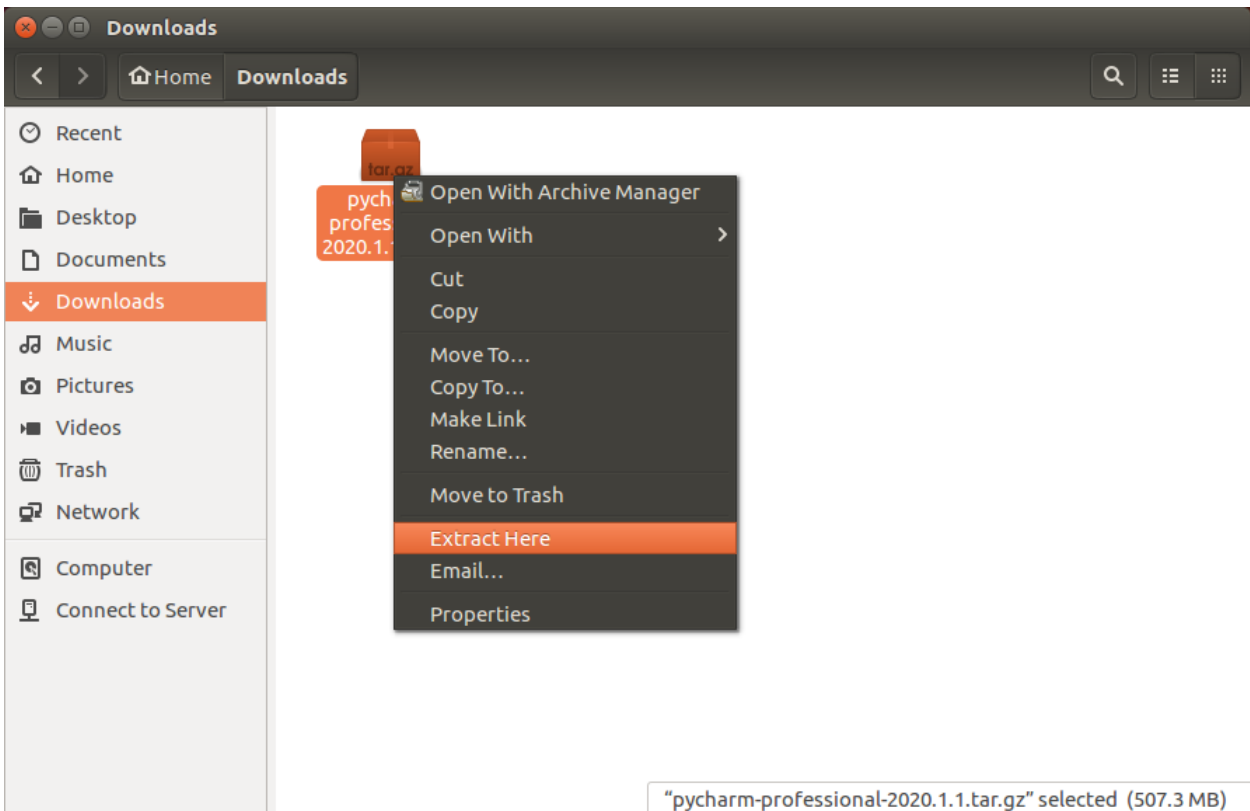


Figura 10. Extracción PyCharm.

2. Abrimos el terminal y nos vamos al directorio *bin* con el comando: `cd Descargas/pycharm-*/bin` o abrimos el terminal directamente en el directorio.

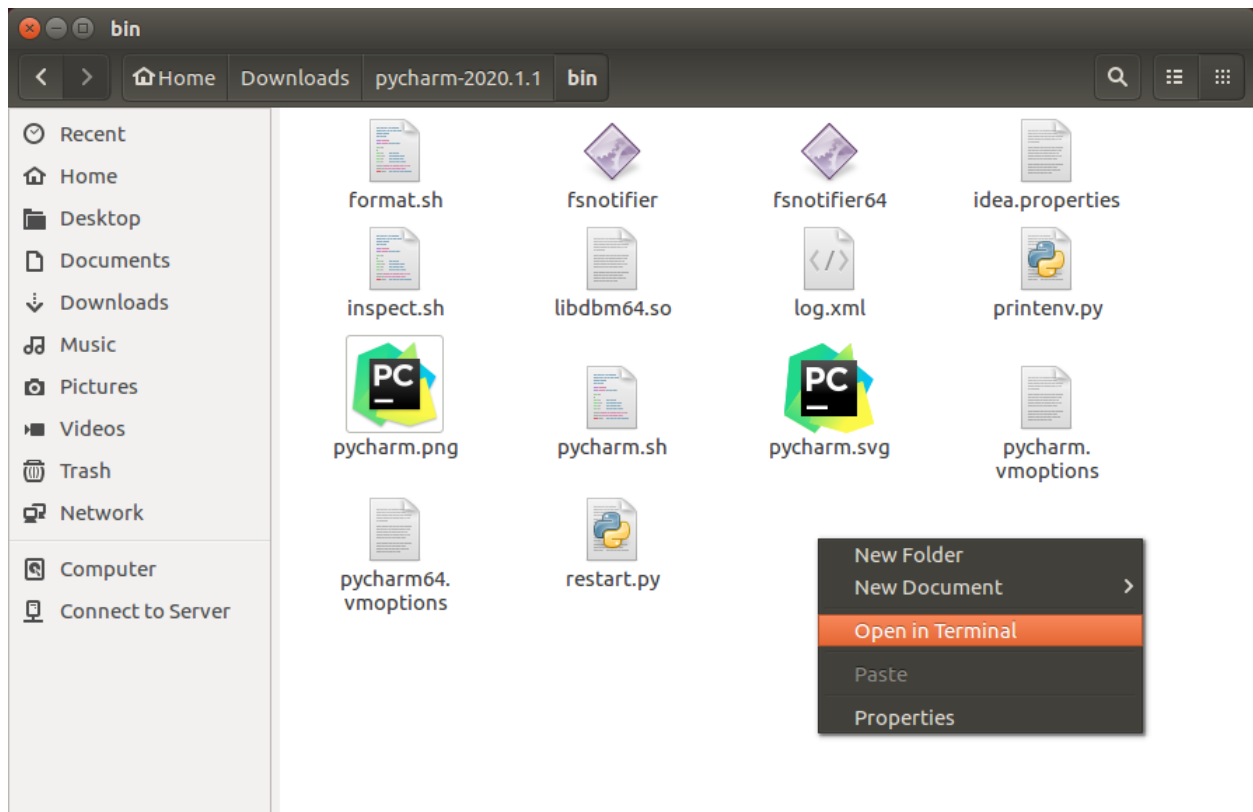


Figura 11. Apertura terminal.

3. Ejecutamos el instalador: `./pycharm.sh` y aceptamos las condiciones del “User Agreement”.

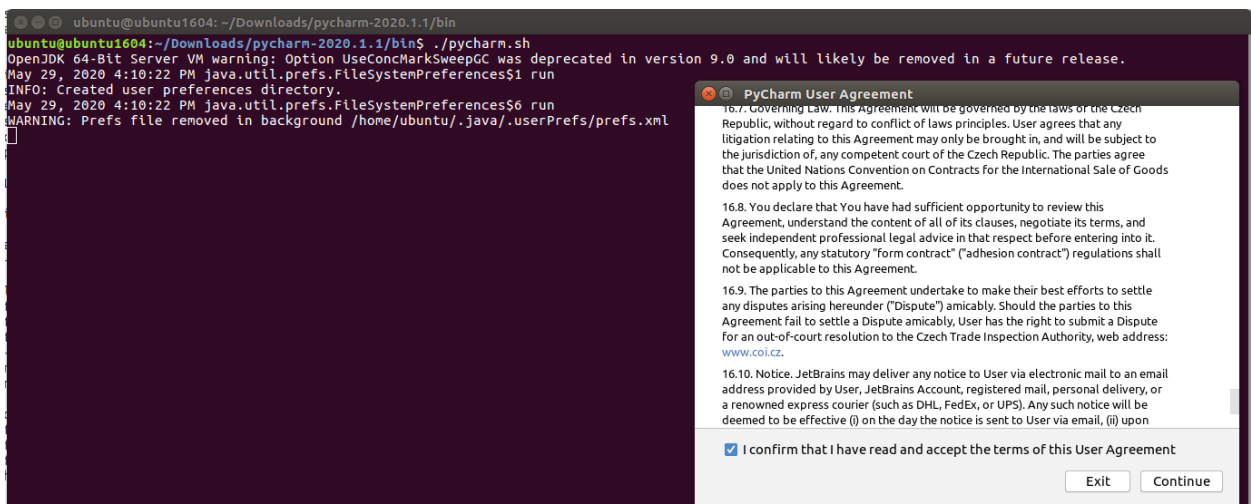


Figura 12. User Agreement.

4. Seleccionamos “Evaluate for free” e introducimos los datos que consideremos. Clickamos en “Evaluate” y, una vez todo en orden, “Continue”.

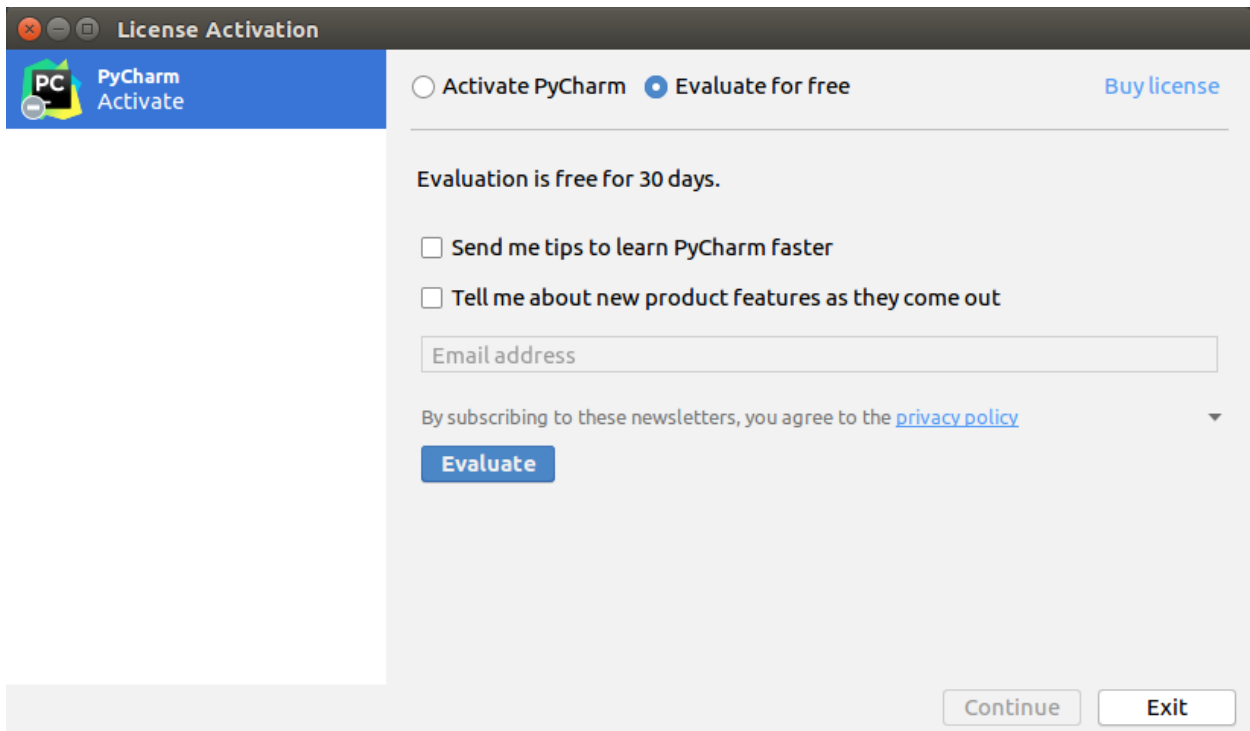


Figura 13. PyCharm License Activation.

5. Ya podemos elegir si queremos abrir un proyecto o crearlo. En un principio crearemos uno nuevo para seguir explicando el montaje del entorno que llamaremos “HelloWorld”.

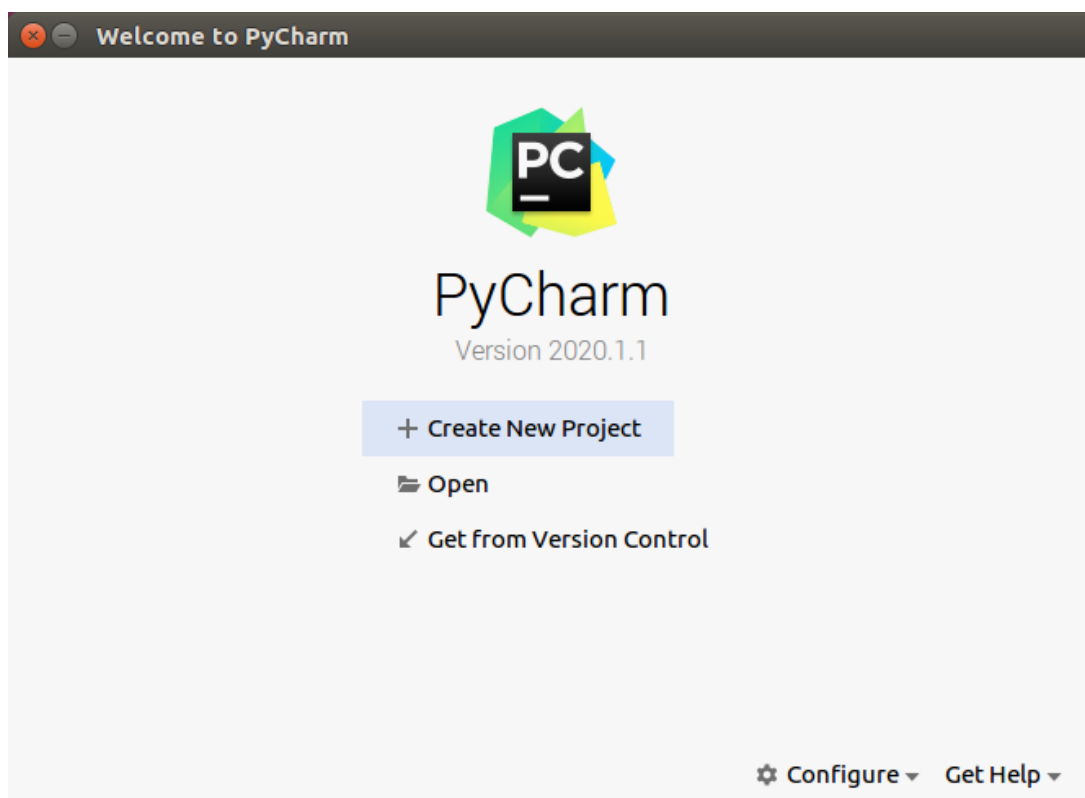


Figura 14. Create New Project.



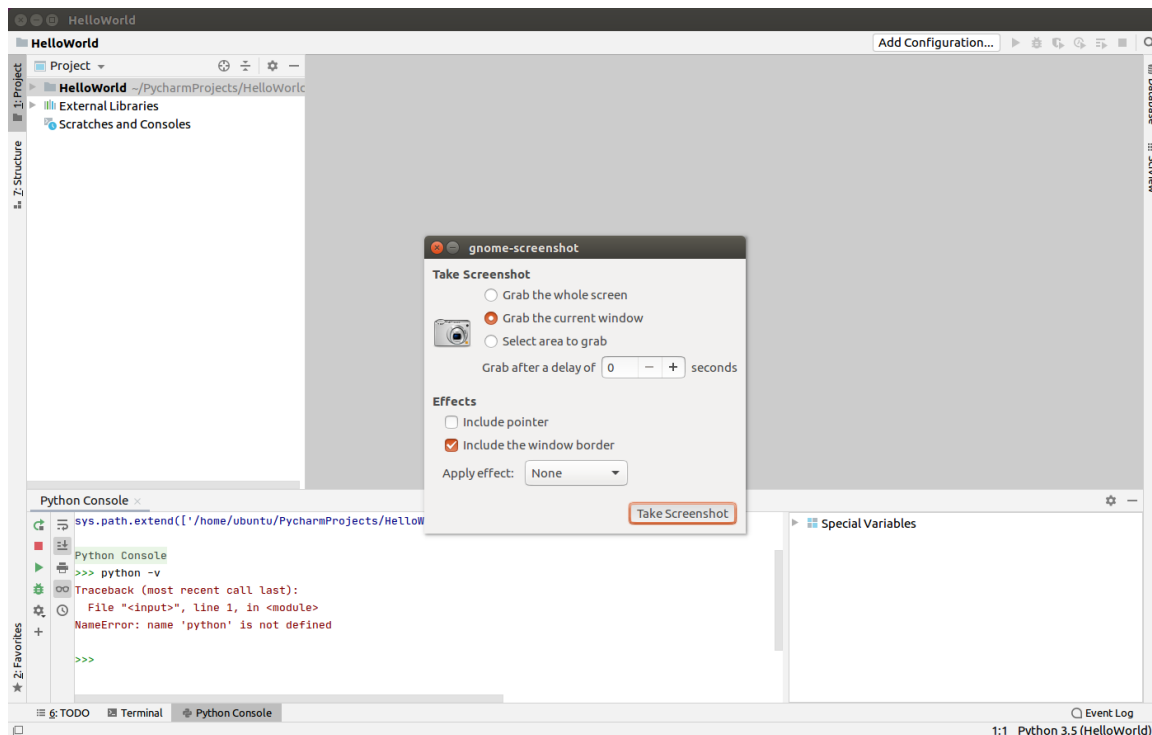


Figura 15. Hello World.

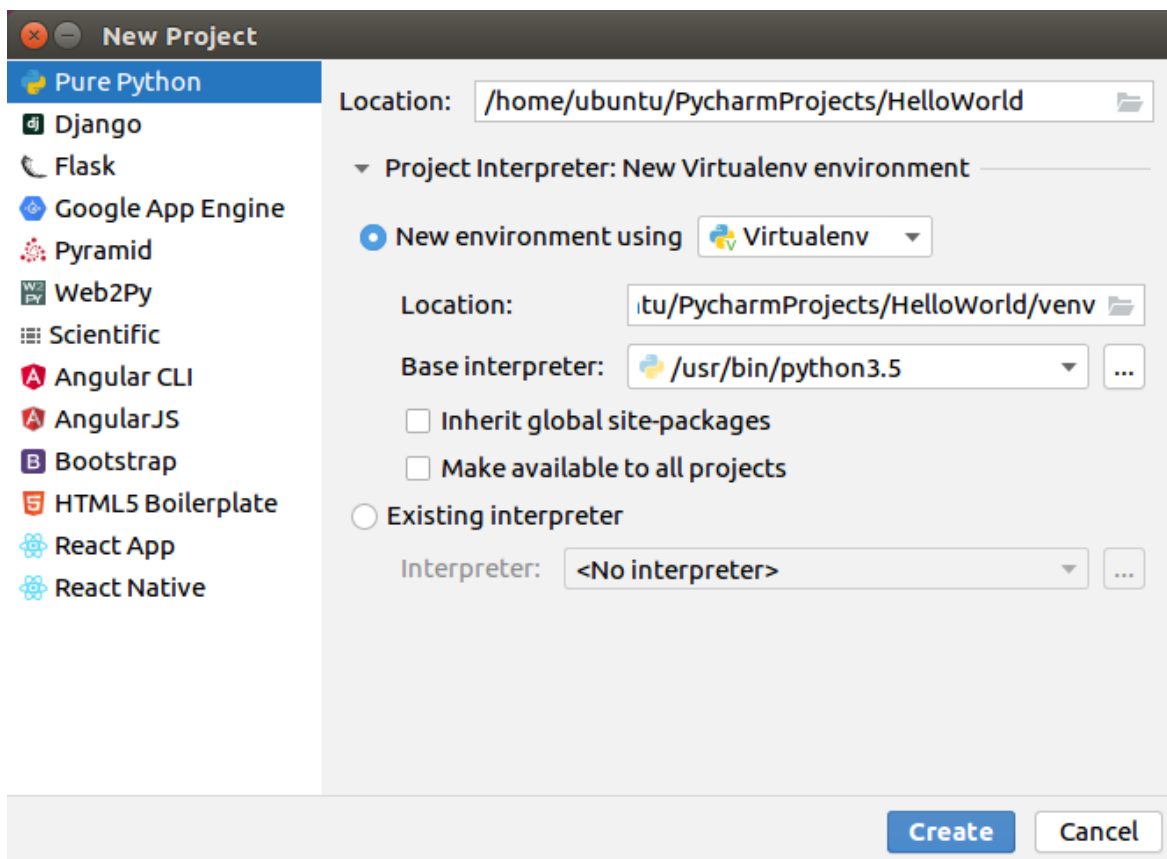


Figura 16. Hello World 2.

### 2.1.3 Python

Debemos tener instalado Python en nuestro sistema. Para ello comprobamos desde la consola Python de Pycharm si tenemos instalado python y, en este caso, lo tenemos. Procedemos a instalarlo.

En caso de no tenerlo utilizar este enlace <https://www.hostinger.es/tutoriales/instalar-python-pip-ubuntu/> de una guía para instalar python y la herramienta pip para instalar módulos.

Para comprobar que está todo en orden creamos un documento llamado “helloworld.py” y lo ejecutamos:

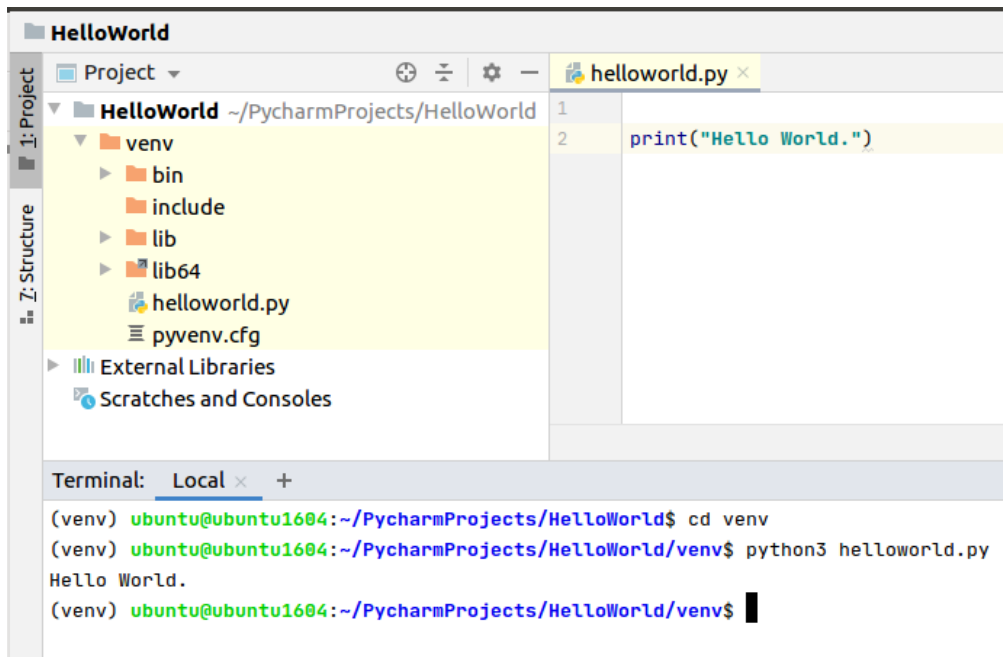


Figura 17. Hello World 3.

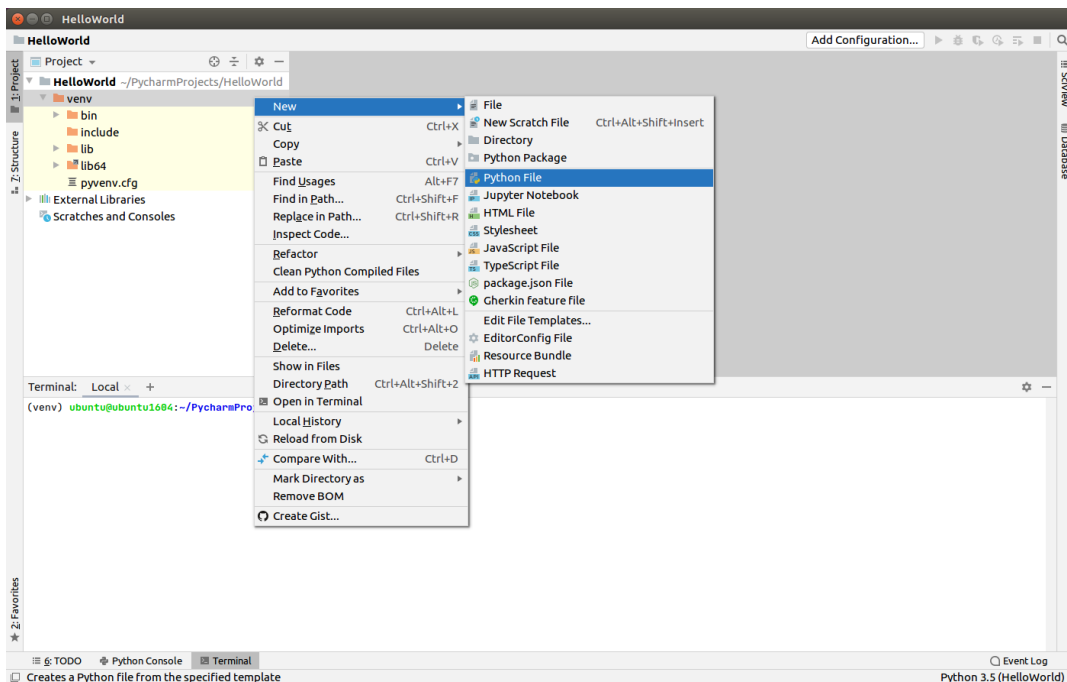


Figura 18. Hello World 4.

Todo funciona correctamente así que vamos al siguiente paso.

## 2.1.4 Scapy

Para poder extraer la información que nos interesa de las tramas Probe Request necesitamos una herramienta que tenga este tipo de funcionalidades. Para Python tenemos Scapy que es una herramienta de manipulación de paquetes para redes. Su adición al proyecto es muy sencilla, lo muestro a continuación:

### OPCIÓN 1

1. Descargar el repositorio de <https://github.com/secdev/scapy>.

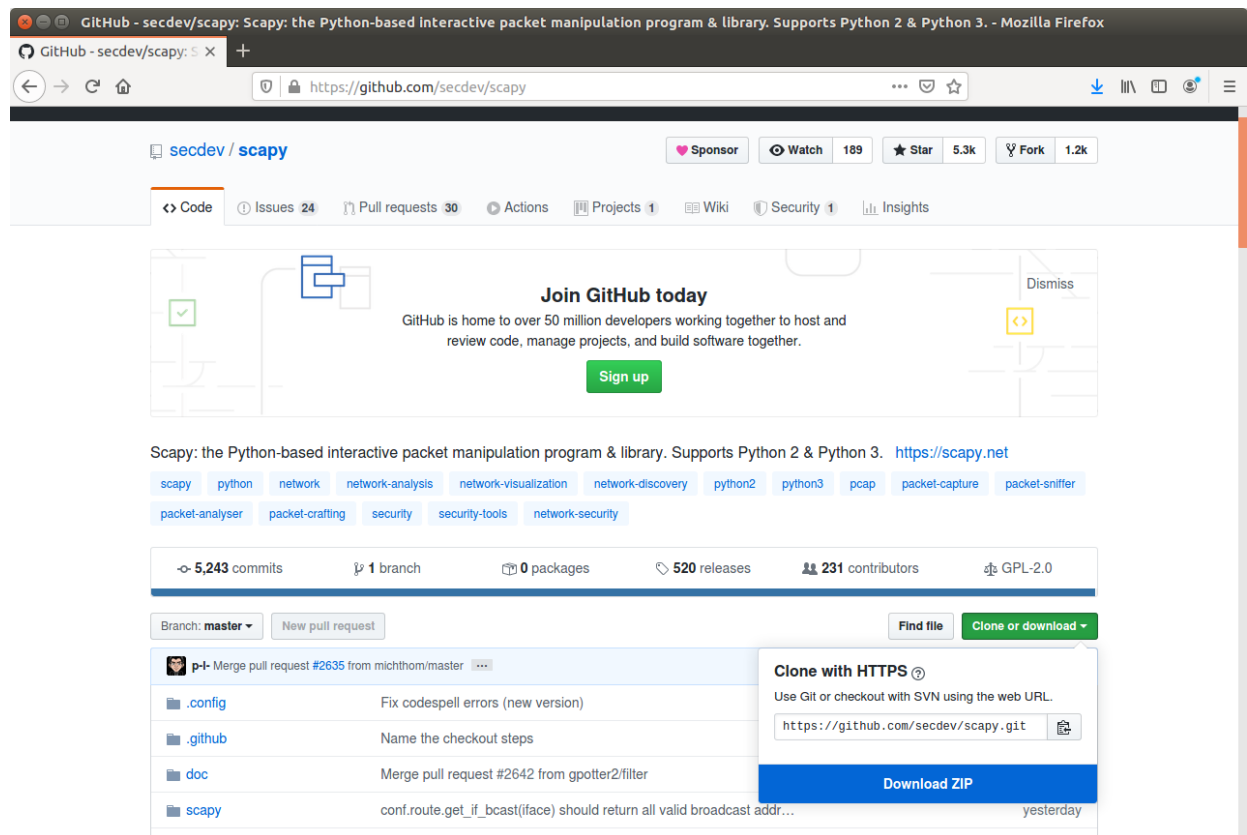


Figura 19. Scapy GitHub.

2. Se descargará un comprimido zip con nombre scapy-master. Al extraer el contenido tendremos un directorio scapy-master y dentro, entre otros, el directorio scapy. Incluimos el directorio /scapy dentro de nuestro proyecto que es el que contiene todas las funciones que necesitamos.



Figura 20. Scapy en nuestro proyecto.

## OPCIÓN 2

Podemos optar por instalar scapy desde terminal: ubicarnos en el directorio del proyecto y ejecutar comandos del siguiente enlace <https://scapy.readthedocs.io/en/stable/installation.html> .

### 2.1.4.1 NETADDR

Una vez integrado Scapy en el proyecto solo nos falta la librería netaddr para poder importarla, acceder a los campos de las tramas y obtener la información que necesaria para crear la firma.

La forma más sencilla es usando el siguiente comando: `pip install netaddr` .

### 2.1.5 Interfaces



Figura 21. TP-LINK Archer T4U

Para llevar a cabo este proyecto es imprescindible capturar tráfico WiFi. Para poder hacerlo utilizaremos un adaptador de la marca TP-LINK modelo Archer T4U. Ver Figura 19.

Además tendremos que configurar la interfaz para dicho adaptador. Para ello he creado un script para ejecutar al principio antes de ejecutar cualquier código. Ver Figura 20. El contenido no hace más que levantar la interfaz correspondiente a nuestro adaptador y ponerla en modo monitor para que pueda captar el tráfico.

```
#!/bin/bash
sudo ifconfig wlan1 down
sudo iwconfig wlan1 mode monitor
sudo ifconfig wlan1 up
```

Figura 22. Código activador de interfaz.

### 2.1.6 Wireshark

Una vez configurada la interfaz WiFi podemos realizar capturas de tráfico con algún programa de monitorización de tráfico. En este proyecto se utilizó Wireshark. El uso de este tipo de herramientas es muy importante porque nos aportan más claridad de información y es un gran apoyo para saber lo que hay realmente en la red.

Además, para este proyecto usaremos capturas pcap que proporcionaremos al script para analizar el tráfico, aunque también podrían realizarse lecturas directas del tráfico en vivo.

Para poder ver todas las interfaces con Wireshark lo ideal es lanzarlo desde línea de comandos con `sudo wireshark` y podremos ver nuestra interfaz WiFi `wlan1`.

Como ayuda para realizar los filtrados y otro tipo de funciones aquí tenemos un manual sobre Wireshark <http://www.tcpdump.org/manpages/pcap-filter.7.html> .

## 2.2 Introducción a Scapy

Scapy es una herramienta de parseo y manipulación de paquetes para redes, originalmente escrita en Python por Philippe Biondi. Puede falsificar o decodificar paquetes, enviarlos por cable, capturarlos y hacer coincidir solicitudes y respuestas. También puede manejar tareas como escaneo, rastreo, sondeo, pruebas unitarias, ataques y descubrimiento de redes.

Scapy proporciona una interfaz de Python en libpcap o sockets nativos sin procesar, de manera similar a la que Wireshark proporciona una GUI de visualización y captura. Difiere al admitir inyección de paquetes, formatos de paquetes personalizados y secuencias de comandos. Si bien es una herramienta de línea de comandos, aún puede interactuar con otros programas para proporcionar visualización. A partir de la versión 2.4.0, Scapy admite Python 2.7 y 3.4+.

Por esto es ampliamente utilizado para implementar gran cantidad de operaciones sobre protocolos de red, entre los que figuran: testing, debugging, auditoría de seguridad, fuzzing, etc.

En cierto modo se puede comparar con Wireshark, como sistema de parseo y disección de protocolos de red. Pero además Scapy permite realizar simples programas para tratar, modificar, grabar, leer, e inyectar estos datos a la red, lo cual concede infinitas posibilidades de manipulación de tramas a todos aquellos que no deseen enfrentarse a otros lenguajes más complejos.

Debido a esto, Scapy no es una herramienta que ofrezca un gran rendimiento para el envío masivo de paquetes o la lectura de los mismos, ya que tiene que procesar uno a uno.

Para el desarrollo de este trabajo se ha utilizado el libro “Python Scapy Dot11 PROGRAMACIÓN EN PYTHON PARA PENTESTERS WI-FI” de Yago Hansen que muestra mediante una progresión de ejemplos cada vez más complejos el uso que se puede hacer de Scapy con el protocolo 802.11. Podemos encontrar en internet los ejemplos utilizados en este libro [4].

Funciones más útiles para el desarrollo de este trabajo:

➤ **ls()**

Si ejecutamos esta función sin parámetros nos aparecerá una lista de las clases de paquete. Al ejecutar esta función pasando como argumento cualquiera de dichas clases se podrá obtener sus campos. Así pues, si ejecutamos `ls(ProbeReq)` obtendremos los campos de la trama Probe Request.

➤ **sniff(iface="wlx18d6c7119941", count=int(cuenta\_paquetes), lfilter=lambda pqt: Dot11ProbeReq in pqt, prn=handle\_packet)**  
**sniff(offline=cuenta\_paquetes, lfilter=lambda pqt: Dot11ProbeReq in pqt, prn=handle\_packet)**

Con la función `sniff()` sniffamos paquetes especificando la interfaz a utilizar y el número de paquetes a contar con “iface” y “count” respectivamente, o indicando un pcap del que leer con “offline”. El manejador de paquetes se indica con “prn” y da como parámetro el paquete recibido (pkt). Con “lfilter” tomaremos solo las tramas Probe Request.

Esta es la función más importante del código. Con ella se inicia el procesamiento de los paquetes.

➤ **show()**

Si ejecutamos esta función con el paquete como parámetro (pkt) se muestra por pantalla toda la trama y de ahí ha sido posible obtener datos, la estructura de la trama, el identificador de los campos.

## 2.3 Visión general

Una vez conocidos todos los elementos que conforman el sistema y posibilitan el desarrollo del mismo, se puede comenzar a desarrollar el código del proyecto. Para crear nuestro script necesitaremos monitorizar tráfico, que lo haremos o bien con nuestra interfaz inalámbrica o con los pcaps y con la función `sniff()` para iniciar el procesamiento.

Clasificaremos dicho tráfico en función de si nos proporciona dirección MAC verdadera o no y extraeremos los datos que nos interesan para crear la firma del dispositivo. La información de la firma se almacenará en listas de datos donde el índice será el identificador creado para ese dispositivo.

Por último, se mostrará por el terminal un resumen de los datos obtenidos y estadísticas que resultan de interés para este proyecto.

Paralelamente, en un código secundario, sobre la misma lectura, se generarán aproximaciones de la cantidad de dispositivos que ofrecen MAC verdadera y MAC falsa para compararlas con las lecturas del código principal.

# 3 IMPLEMENTACIÓN

*“Yo creo que los referentes son importantes no solo cuando ya estás haciendo la carrera y cuando vas a salir sino antes, en los colegios.”*

- Sara Gómez -

## 3.1 Diagramas de flujo del Sistema

El algoritmo se presentará a través de diagramas de flujo.

Disponemos de **dos ficheros**: uno donde se desarrollará la **detección** de los usuarios y otro con una **aproximación** estadística del número de usuarios que deberíamos obtener. Para describir su funcionamiento está dividido en **5 diagramas** que representan el funcionamiento fundamental del código:

### 3.1.1 Puesta en marcha

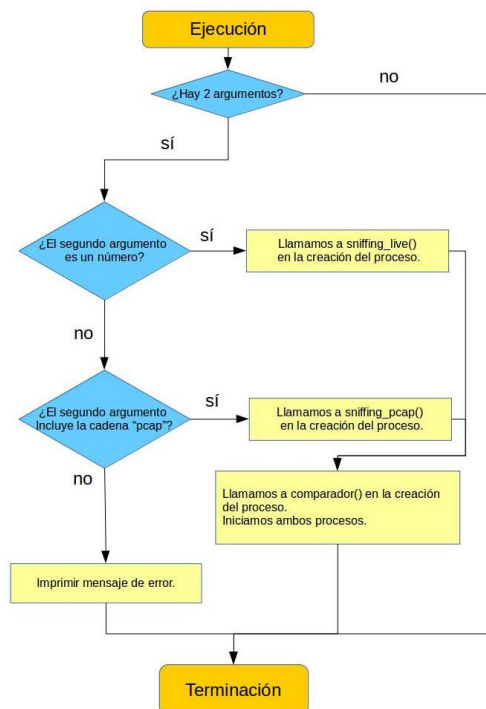


Figura 23. Diagrama puesta en marcha.

Cuando ponemos en marcha el script para contar los dispositivos ejecutamos ambos ficheros a la vez, no es necesario hacer dos ejecuciones por separado. Ejecutaremos el script principal y le pasaremos como argumento un número o un fichero pcap. Dicho número es el número de tramas Probe Request que queremos que se analicen y el fichero pcap es una captura de tráfico ya hecha por cualquier plataforma que pueda generarla. Si introducimos cualquier otro elemento que no sea un número (de las cifras que sea) o un archivo pcap previamente ubicado en su directorio correspondiente, obtendremos un mensaje de error en la pantalla del terminal.

Esta forma de enfocarlo nos permite hacer lecturas de tráfico en vivo y a posteriori. Por ello, como las llamadas a la función `sniff()` de Scapy son diferentes, están ubicadas en funciones diferentes siendo `sniffing_live()` la llamada cuando analizamos tráfico en vivo y `sniffing_pcap()` la llamada cuando utilizamos lectura desde un archivo pcap.

Gracias al multiprocesamiento podemos ejecutar ambos ficheros a la vez y no perder detalle ni tener delay entre ambas ejecuciones para que la estimación sea lo más óptima posible.

### 3.1.2 Fichero de comprobación

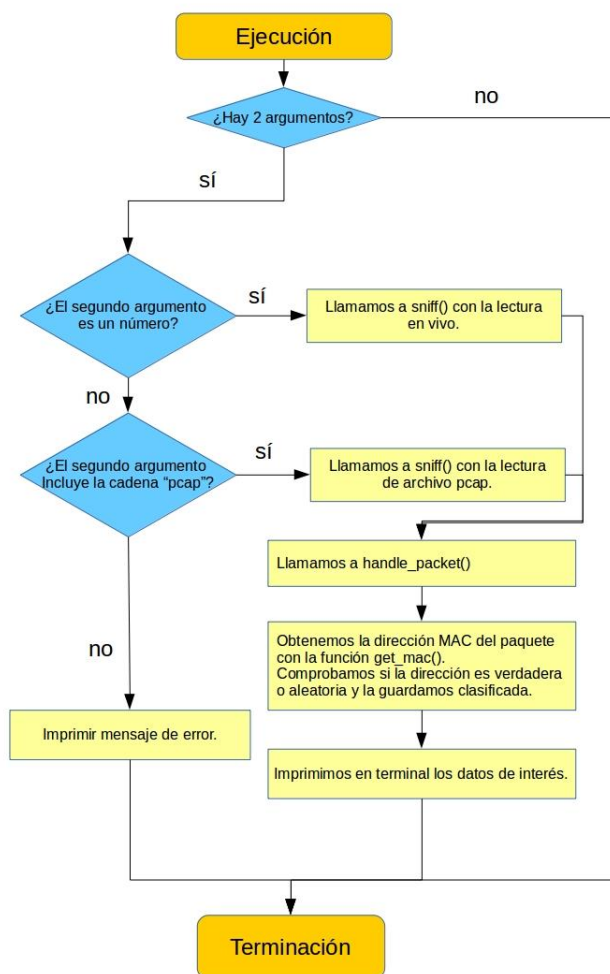


Figura 24. Diagrama comprobación.



Como ya hemos comentado, tenemos un fichero principal y uno de comprobación. En este diagrama explico la estructura y funcionamiento del fichero de comprobación. La ejecución de este fichero es la misma que la del principal: *nombre\_fichero.py* param. El parámetro volverá a ser o bien un número o bien un fichero pcap. Este parámetro se pasará desde el fichero principal, por lo que en teoría todo debe ser correcto, pues ya está asegurado en el código que solo haya dos parámetros. El hecho de volver a comprobar si el parámetro pasado es un número o un fichero pcap no es más que para saber qué tipo de llamada se debe hacer a la función *sniff()*, pues, como contábamos en el diagrama anterior, varía en función del parámetro.

El mensaje de error puede ser redundante puesto que ya hemos hecho una comprobación previa en el fichero principal pero, por si acaso, lo dejamos.

En cualquier caso el funcionamiento de este fichero es muy simple: cuando llamamos a *handle\_packet()* obtenemos la dirección MAC de cada dispositivo y en función de si es una MAC verdadera o aleatoria las separamos. También contabilizamos las probes, así, al final podemos hacer nuestros cálculos pertinentes y mostrar la información que queremos.

### 3.1.3 Fichero principal

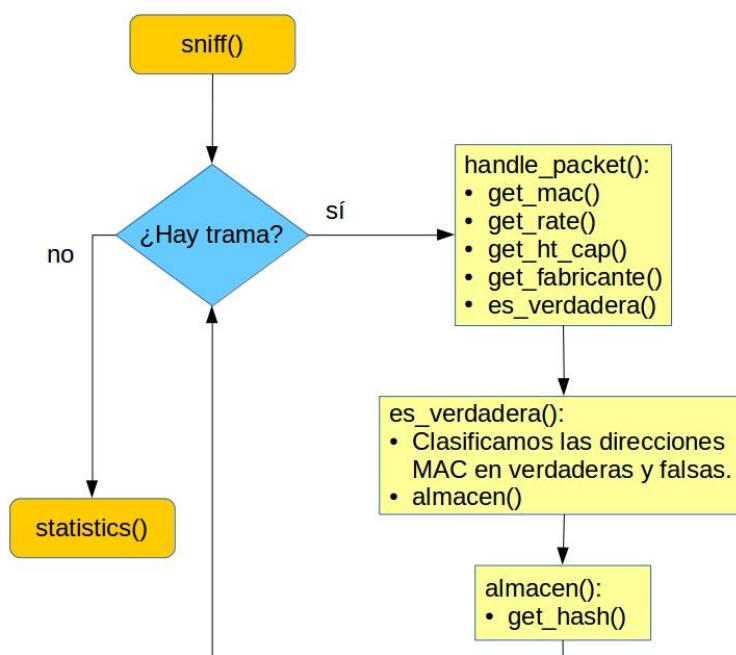


Figura 25. Diagrama fichero principal.

Una vez llamada a la función creada que ejecuta la función *sniff()* de Scapy con sus correspondientes parámetros, el proceso a seguir es el mismo si hacemos lectura de tráfico en vivo que si lo hacemos de un pcap.

A cada trama Probe Request la haremos pasar por un proceso en el que extraeremos los datos que nos interesan y los trataremos y almacenaremos para su posterior uso. Mientras haya tramas por leer seguiremos en el bucle que se ve en el diagrama. Cuando ya no haya más tramas por leer llamaremos a la función *statistics()* para mostrar la información que nos interesa.

Cada vez que se detecta una trama se le extraen los parámetros que nos interesan: dirección MAC, rates, HTCapabilities. Disponiendo de la MAC podemos sacar el fabricante, que lo usaremos para crear la firma. Para ello se utilizan las funciones mostradas en el cuadro de la función `handle_packet()`.

La función `es_verdadera()` será la que nos lleve al siguiente paso que es clasificar el dispositivo en función de si nos da una dirección MAC verdadera o aleatoria. Tras ello nos vamos a `almacen()` donde organizamos, guardamos y actualizamos la información recogida según convenga.

Dentro de `almacen()` haremos uso de la función `get_hash()`, función que nos generará el identificador unívoco del dispositivo en caso de que no lo tengamos ya almacenado.

### 3.1.4 Función almacen()

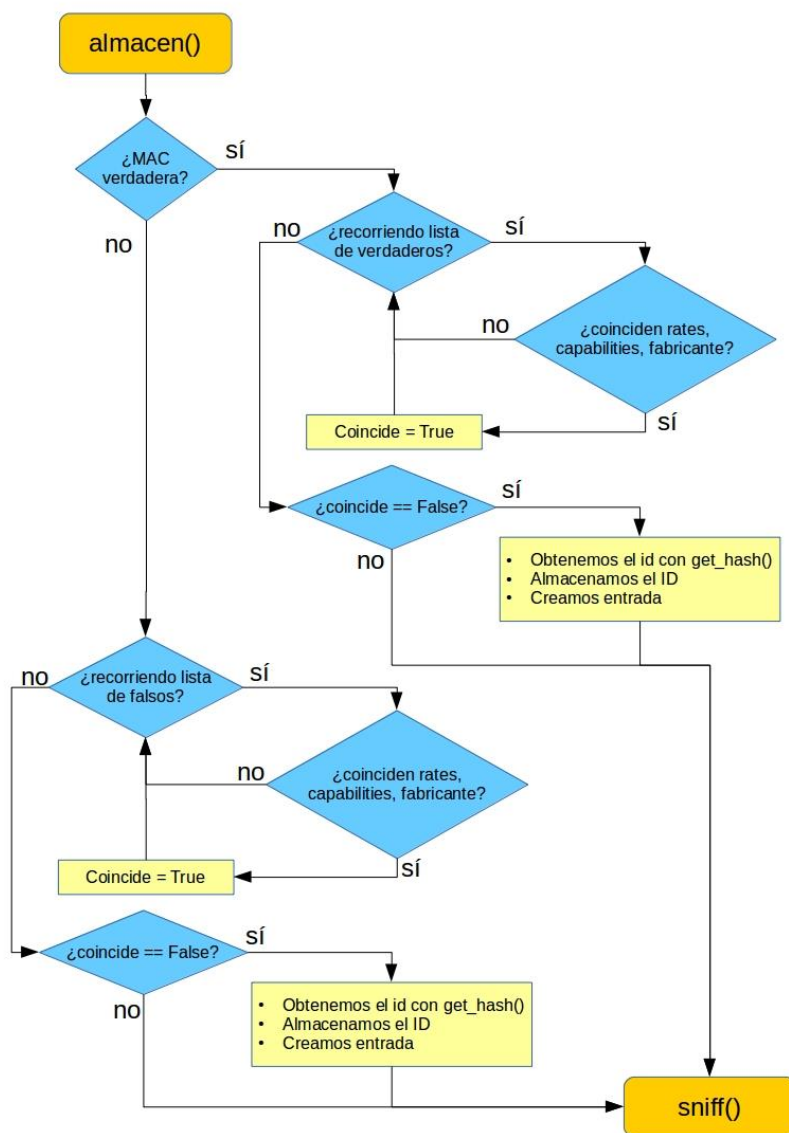


Figura 26. Diagrama almacen.

Es importante desarrollar esta función en más profundidad dado que es donde se hace todo el almacenamiento y organización de información para la identificación de los dispositivos. Entre los datos recibidos como parámetro tenemos ese "True" o "False" que nos indica si el dispositivo que ha emitido dicha trama ha proporcionado su MAC verdadera o no. En función de ello, en caso de almacenamiento utilizaremos una variable u otra, por eso está separado.

La primera bifurcación es si la dirección MAC es verdadera ("True"). En caso positivo revisamos si los parámetros del dispositivo (Rates, HTCapabilities y Fabricante) que estamos analizando coinciden con alguna entrada ya registrada. En caso positivo no se hace nada, puesto que ya está registrado. En caso negativo, generamos el id y creamos una nueva entrada en el almacén para nuestro dispositivo.

Si el dispositivo no nos ha dado su MAC real ("False"), en la primera bifurcación tendremos que volver a preguntarnos si tenemos ese dispositivo repetido o no. Llevaremos a cabo el mismo proceso que en el caso del dispositivo con MAC verdadera.

Al terminar volveremos a la función `sniff()` a comprobar si hay tramas nuevas como se ha explicado en el diagrama anterior.

### 3.1.5 Función `es_verdadera()`

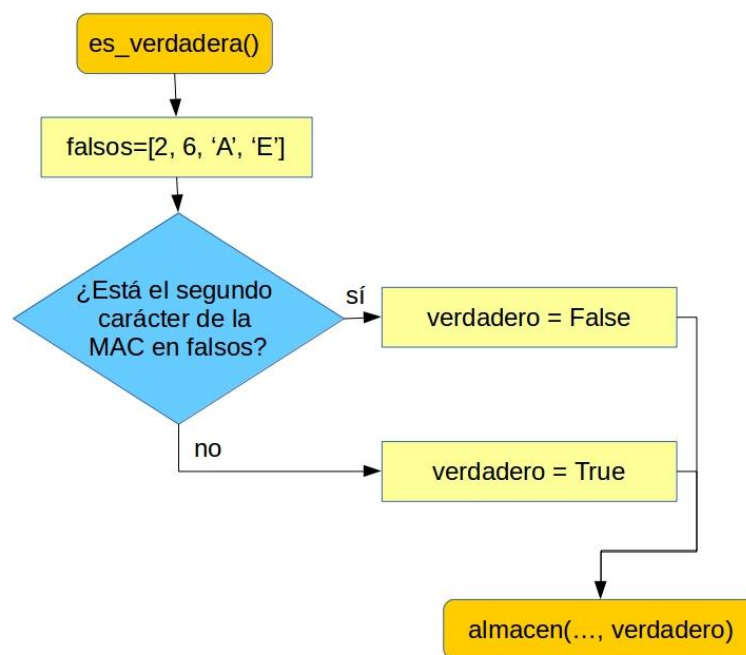


Figura 27. Diagrama `es_verdadera`.

La función `es_verdadera()` es importante porque es el primer elemento de la cadena de clasificación. En ella se difiere entre dispositivo con dirección MAC falsa y dispositivo con dirección MAC verdadera y nos aporta ese paso previo antes de entrar en almacén para clasificar y guardar el registro.

## 3.2 Desarrollo y enfoque del código

En este apartado se explican las transformaciones que ha sufrido el código a medida que se ha ido creando. Al ser un área por investigar, se han tenido que ir dando pequeños pasos para así asegurarlos. Esto implica cambios en el código y el algoritmo para poder comprobar y mejorar el modelo propuesto.

Como ya hemos comentado antes, una de las aplicaciones de este algoritmo es leer tráfico en tiempo real. De hecho, ese fue el primer paso. Más tarde se añadió la funcionalidad de leer de pcaps, lo que nos ha proporcionado una gran ayuda a la hora de comparar y analizar al detalle el rendimiento del código.

Para la extracción de los campos de interés de la firma nos hemos ayudado del programa de análisis de protocolos Wireshark. Cada campo viene acompañado por un número, un identificador, con el que poder localizarlo en el payload. Este número es el que buscamos en las funciones get mostradas en los diagramas.

El registro de firmas está compuesto por una estructura de lista donde el índice será el ID y el contenido un vector con los elementos de dicha firma:

```
dispositivos[ID] = [rates, htcapabilities, fabricante]
```

Obviamente ha habido una fase de estudio y pruebas sobre qué campos de la Probe Request eran los mejores para generar una mayor fiabilidad y precisión en el conteo e identificación de dispositivos. Se han estudiado los campos: rates y htcapabilities, pero no era suficiente. Después se añadieron el fabricante, los vendor specific, las extended rates, las extended htcapabilities, las vht capabilities y los SSIDs, pero este grupo y sus combinaciones generaban demasiadas combinaciones de datos y se disparaba el número de supuestos dispositivos en relación con los reales, así que quedó eliminado. Es cierto que los vendor specific no están presentes en todos los dispositivos, pero resultan de interés y de ayuda para identificar a los dispositivos que los usen.

Al final, tras probar diferentes combinaciones, las cuales se ilustrarán en el apartado de los experimentos, nos quedamos con: Rates, HTCcapabilities y fabricante.

Para mejorar aún más la precisión se tomaron dos medidas:

1. La creación de un fichero de comparación, es decir, un fichero que nos da de manera aproximada el número de usuarios analizados. Es de manera aproximada porque se toman los usuarios que proporcionan sus direcciones MAC verdaderas, la cantidad de tramas Probe Request emitidas por ellos y se calcula una media. En base a esa proporción se calculan los dispositivos que emiten dirección MAC aleatoria.

Con este fichero podemos comprobar cómo de lejos o cerca estamos de identificar a todos los dispositivos.

2. Por otro lado, hemos de resaltar el orden de estudio. Primero me nos hemos centrado en estudiar los casos con dispositivos que proporcionan dirección MAC verdadera. Así tenemos datos reales con los que contrastar las estimaciones que salen tras utilizar diferentes combinaciones de campos. Una vez terminado el estudio con las direcciones MAC verdaderas se puede extender a las direcciones MAC aleatorias.

## 4 EXPERIMENTOS

*"Tuve la fortuna de que tanto mi padre como mi abuelo me alentaron a seguir cualquier carrera o camino que eligiera en la vida. Ni se les ocurría que la elección tuviera algo que ver con ser hombre o mujer."*

- Margaret Hamilton-

### 4.1 Fingerprint

A lo largo de este proyecto hemos barajado varias combinaciones posibles de fingerprint que son:

- **Fingerprint 1:** Rates, HTCcapabilities, Fabricante
- **Fingerprint 2:** Rates, HTCcapabilities, Fabricante, Vendor Specific
- **Fingerprint 3:** Rates, HTCcapabilities, Fabricante, Vendor Specific, Extended Rates
- **Fingerprint 4:** Rates, HTCcapabilities, Fabricante, Vendor Specific, Extended Rates, Extended HTCcapabilities, VHT Capabilities
- **Fingerprint 5:** Rates, HTCcapabilities, Fabricante, SSIDs

Para comparar su efectividad seleccionamos los escenarios de características opuestas (cantidad de probes leídas y proporción de acierto en la detección de usuarios) tanto en las lecturas de 2013 como en las actuales.

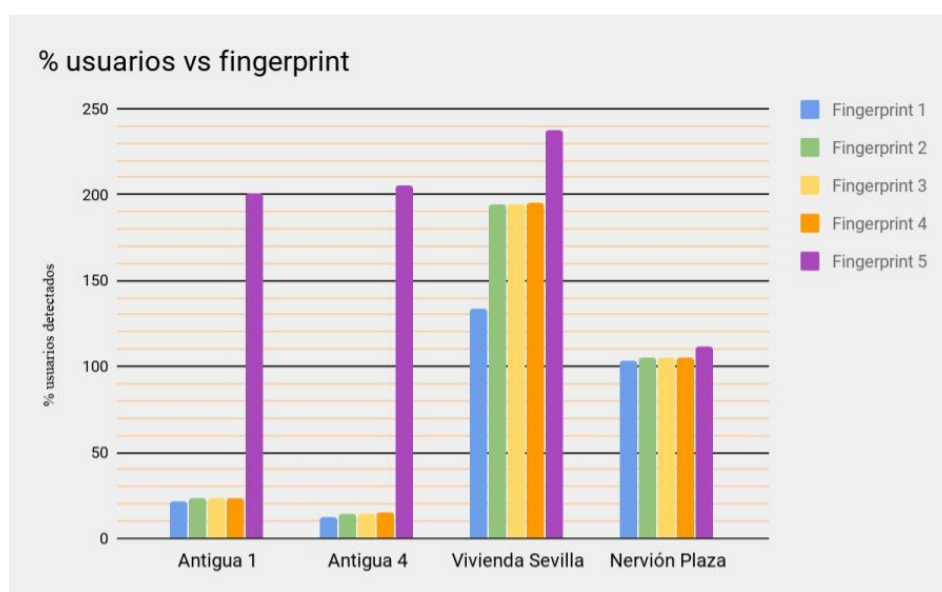


Figura 28. Comparación entre las 5 fingerprints iniciales con la cantidad de dispositivos que detectan respecto al fichero de comprobación.

Como podemos ver, el Fingerprint 5 tiene demasiadas variaciones debido a que el campo SSID no siempre es el mismo. Aunque el dispositivo no varíe las redes conocidas, a veces las envía y a veces no, y si sumamos la posible variación en el resto de los campos, no es nada acertado usar esta combinación ni añadirle más campos. Por lo tanto, **el campo SSID queda descartado**.

Si eliminamos el Fingerprint 5 nos queda lo siguiente:

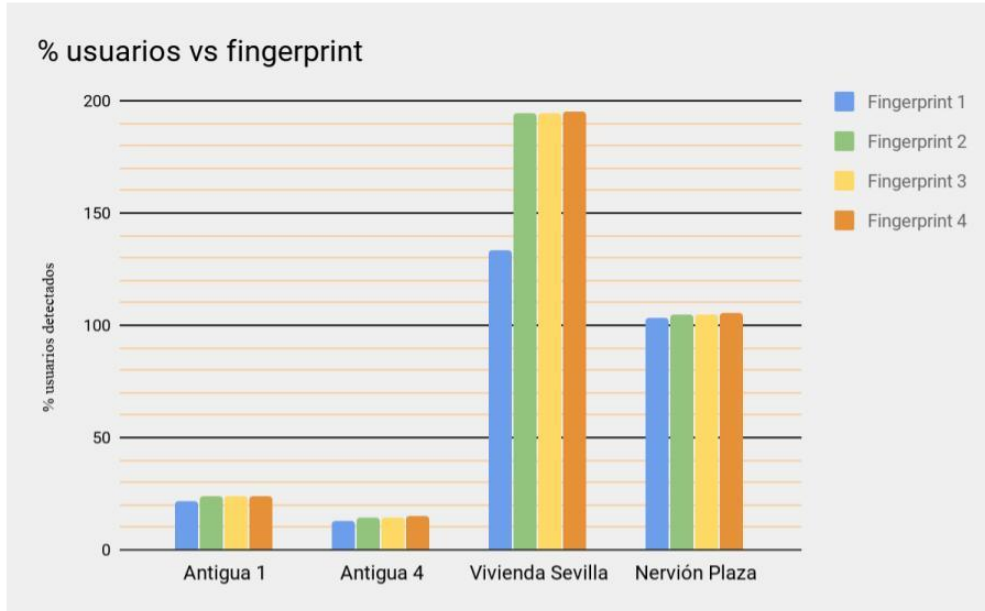


Figura 29. Comparación entre las 4 fingerprints con la mejor cifra de dispositivos detectos respecto al fichero de comprobación.

Podemos apreciar que el Fingerprint 4 genera un ligero aumento de usuarios en comparación con los otros tres fingerprints. Así que también lo descartamos.

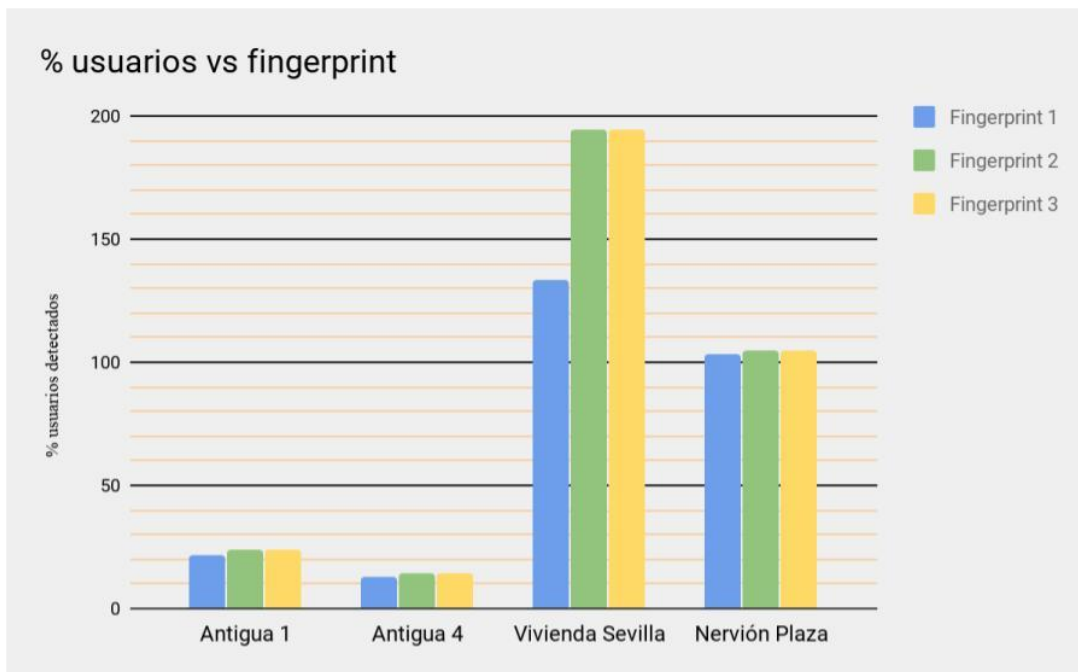


Figura 30. Comparación entre las 3 fingerprints con la mejor cifra de dispositivos detectos respecto al fichero de comprobación.

Dado que tanto **Fingerprint 2 como Fingerprint 3 cuentan los mismos dispositivos** y solo se diferencian en un campo, si varía nos daría aún más dispositivos y no nos interesa pues cualquier adición de campos nos genera riesgo de seguir detectando más usuarios de los que realmente hay.

Aún así, donde más claro se ve es en el caso Vivienda Sevilla, donde el Fingerprint 1 nos da una mejor precisión que los otros dos. Por lo tanto, **nos quedamos con el Fingerprint 1**.

Basándonos en este, estudiaremos algunos parámetros de interés.

## 4.2 Casos

Una vez encontrado el fingerprint más acertado, procedemos a evaluar los diferentes casos que nos encontramos.

Los dos escenarios que se contemplan en el proyecto son tanto el análisis de tráfico tanto en **vivo** como el de archivos **pcap**. En cada uno de ellos tomaremos varios ejemplos para ver las diferencias entre casos del mismo tipo.

Existen casos en los que detectamos más del 100% de los usuarios. Esto se suele deber a que algunos dispositivos modernos están empezando a cambiar algunos datos entre tramas. De esta forma, un mismo dispositivo puede ser contado como 2 o 3 dispositivos al exhibir campos distintos (y por tanto con fingerprint distinta).

Es importante destacar que realmente **no sabemos cuántos dispositivos hay**, es decir, ese sobre-porcentaje es respecto de lo estimado con el ratio generado por el fichero de comprobación que nos dice las MAC verdaderas. No tiene por qué ser exacto.

### 4.2.1 Lecturas 2013

En esta tipología de casos tomamos archivos pcap descargados de bases de datos públicas de monitorizaciones de Probes Request [6] [7]. A continuación, tenemos diferentes ejecuciones:

```
-----  
Probes mac verdadera: 75109  
Probes mac falsa: 0  
Usuarios supuestamente VERDADEROS: 3909 y su proporción de probes: 19.21437707853671  
Usuarios supuestamente FALSOS: 0 y su cantidad en función de la proporción: 0.0  
Numero total de usuarios: 3909.0  
  
-----  
  
-----  
  
STATISTICS  
Número total de probes: 75109  
Número total de macs: 3909  
Media de probes por mac detectada: 19.21437707853671  
Número total de usuarios: 844  
  
-----
```

Figura 31. Antigua 1.

Tomando una muestra de 75.109 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 3909 usuarios.

Según nuestro detector, obtenemos 844 usuarios, es decir, hemos detectado solo el **21,59%**.

```

-----
Probes mac verdadera: 126926
Probes mac falsa: 258
Usuarios supuestamente VERDADEROS: 6684 y su proporción de probes: 18.989527229204068
Usuarios supuestamente FALSOS: 4 y su cantidad en función de la proporción: 13.586436191166507
Numero total de usuarios: 6697.586436191166
-----

-----

STATISTICS
Número total de probes: 127184
Número total de macs: 6688
Media de probes por mac detectada: 19.016746411483254
Número total de usuarios: 1015
-----

```

Figura 32. Antigua 2.

Tomando una muestra de 127.184 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 6697,59 usuarios.

Según nuestro detector, obtenemos 1015 usuarios, es decir, hemos detectado solo el **15,15%**.

```

-----
Probes mac verdadera: 64228
Probes mac falsa: 121
Usuarios supuestamente VERDADEROS: 4144 y su proporción de probes: 15.499034749034749
Usuarios supuestamente FALSOS: 4 y su cantidad en función de la proporción: 7.8069377841439875
Numero total de usuarios: 4151.806937784144
-----

-----

STATISTICS
Número total de probes: 64349
Número total de macs: 4148
Media de probes por mac detectada: 15.513259402121504
Número total de usuarios: 817
-----

```

Figura 33. Antigua 3.



Tomando una muestra de 64.349 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 4151,81 usuarios.

Según nuestro detector, obtenemos 817 usuarios, es decir, hemos detectado solo el **19,68%**.

```
-----  
Probes mac verdadera: 210590  
Probes mac falsa: 140  
Usuarios supuestamente VERDADEROS: 8417 y su proporción de probes: 25.019603184032317  
Usuarios supuestamente FALSOS: 3 y su cantidad en función de la proporción: 5.5956123272710006  
Numero total de usuarios: 8422.595612327272  
  
-----  
  
-----  
  
STATISTICS  
Número total de probes: 210730  
Número total de macs: 8420  
Media de probes por mac detectada: 25.02731591448931  
Número total de usuarios: 1063  
  
-----
```

Figura 34. Antigua 4.

Tomando una muestra de 210.730 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 8422,6 usuarios.

Según nuestro detector, obtenemos 1063 usuarios, es decir, hemos detectado solo el **12,62%**.

Esta baja precisión en todas las muestras es debido a que al ser tan antiguas las lecturas, hay muchos campos que no existen en estas tramas debido a que muchos de ellos se han ido añadiendo con el paso de los años. Por lo tanto, es normal que el número de dispositivos diferentes detectados sea muy reducido.

Veamos algunos casos más actualizados.

#### 4.2.2 Restaurante Salón Romero, Zafra



Figura 35. Restaurante Salón Romero, Zafra.

Tras la monitorización de tramas Probe Request durante **60 minutos** en una cafetería de Zafra, se obtuvieron los siguientes resultados.

```

-----
Probes mac verdadera: 13251
Probes mac falsa: 9097
Usuarios supuestamente VERDADEROS: 1459 y su proporción de probes: 9.082248115147362
Usuarios supuestamente FALSOS: 3070 y su cantidad en función de la proporción: 1001.6242547732246
Numero total de usuarios: 2460.6242547732245
-----

STATISTICS
Número total de probes: 22348
Número total de macs: 4529
Media de probes por mac detectada: 4.9344226098476485
Número total de usuarios: 2474
-----
  
```

Figura 36. Captura Restaurante Salón Romero, Zafra.

Tomando una muestra de 22.348 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 2460,62 usuarios.

Según nuestro detector, obtenemos 2474 usuarios, es decir, hemos detectado el **100,54%**. Esto significa que obtenemos un 0,54% más de usuarios de los que deberíamos. Pero al fin y al cabo podríamos decir que hemos podido identificar todos los usuarios.

### 4.2.3 Centro comercial Torre Sevilla



Figura 37. Centro comercial Torre Sevilla.

Tras la monitorización de tramas Probe Request durante **45 minutos** en la pastelería Granier del centro comercial Torre Sevilla se obtuvieron los siguientes resultados.

```
-----  
STATISTICS  
Número total de probes: 17809  
Número total de macs: 3770  
Media de probes por mac detectada: 4.723872679045093  
Número total de usuarios: 2532  
  
-----  
  
-----  
  
Probes mac verdadera: 11726  
Probes mac falsa: 6083  
Usuarios supuestamente VERDADEROS: 1469 y su proporción de probes: 7.982300884955753  
Usuarios supuestamente FALSOS: 2301 y su cantidad en función de la proporción: 762.060975609756  
Numero total de usuarios: 2231.060975609756  
-----
```

Figura 38. Captura centro comercial Torre Sevilla.

Tomando una muestra de 17.809 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 2231,06 usuarios.

Según nuestro detector, obtenemos 2532 usuarios, es decir, hemos detectado el **113,49%**. Esto significa que obtenemos un 13,49% más de usuarios de los que deberíamos.

#### 4.2.4 Edificio de viviendas Sevilla



Tras la monitorización de tramas Probe Request durante **90 minutos** en una vivienda de Sevilla ubicada en un cuarto piso de la Calle Arjona se obtuvieron los siguientes resultados.

Figura 39. Viviendas en calle Arjona, Sevilla.

```

-----
Probes mac verdadera: 5117
Probes mac falsa: 219
Usuarios supuestamente VERDADEROS: 149 y su proporción de probes: 34.34228187919463
Usuarios supuestamente FALSOS: 86 y su cantidad en función de la proporción: 6.376978698456127
Numero total de usuarios: 155.37697869845613

-----

-----

STATISTICS
Número total de probes: 5336
Número total de macs: 235
Media de probes por mac detectada: 22.706382978723404
Número total de usuarios: 207
-----

```

Figura 40. Captura viviendas en calle Arjona, Sevilla.

Tomando una muestra de 5.336 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 155,38 usuarios.

Según nuestro detector, obtenemos 207 usuarios, es decir, hemos detectado el **133,22%**. Esto significa que obtenemos un 33,22% más de usuarios de los que deberíamos.

## 4.2.5 Centro Sevilla



Figura 41. Avenida Reyes Católicos, Centro Sevilla.

Tras la monitorización de tramas Probe Request durante **50 minutos** en una cafetería de la Calle Reyes Católicos de Sevilla se obtuvieron los siguientes resultados.

```
-----  
Probes mac verdadera: 13836  
Probes mac falsa: 3945  
Usuarios supuestamente VERDADEROS: 1495 y su proporción de probes: 9.25484949832776  
Usuarios supuestamente FALSOS: 1695 y su cantidad en función de la proporción: 426.26300954032956  
Numero total de usuarios: 1921.2630095403297  
  
-----  
  
-----  
  
STATISTICS  
Número total de probes: 17781  
Número total de macs: 3190  
Media de probes por mac detectada: 5.573981191222571  
Número total de usuarios: 2664  
  
-----
```

Figura 42. Captura avenida Reyes Católicos, Centro Sevilla.

Tomando una muestra de 17.781 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 1921.26 usuarios.

Según nuestro detector, obtenemos 2664 usuarios, es decir, hemos detectado el **138,66%**. Esto significa que obtenemos un 38,66% más de usuarios de los que deberíamos.



#### 4.2.6 Centro comercial Nervión Plaza, Sevilla

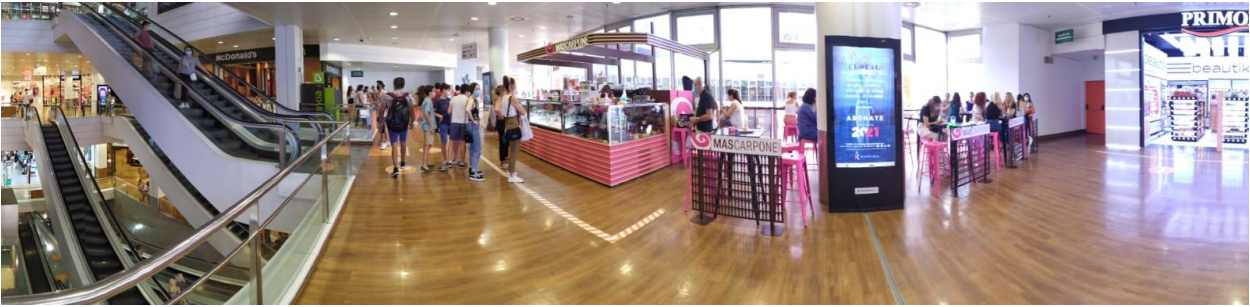


Figura 43. Centro comercial Nervión Plaza, Sevilla.

Tras la monitorización de tramas Probe Request durante **45 minutos** en la pastelería Mascarponi del centro comercial Nervión Plaza de Sevilla se obtuvieron los siguientes resultados.

```

-----
Probes mac verdadera: 17253
Probes mac falsa: 12861
Usuarios supuestamente VERDADEROS: 3724 y su proporción de probes: 4.632921589688507
Usuarios supuestamente FALSOS: 4787 y su cantidad en función de la proporción: 2776.002086593636
Numero total de usuarios: 6500.002086593636
-----

-----

STATISTICS
Número total de probes: 30114
Número total de macs: 8511
Media de probes por mac detectada: 3.5382446246034545
Número total de usuarios: 6698
-----

```

Figura 44. Captura centro comercial Nervión Plaza, Sevilla.

Tomando una muestra de 30.114 probes, la aproximación según la proporcionalidad de probes por MAC verdadera es de 6500 usuarios.

Según nuestro detector, obtenemos 6698 usuarios, es decir, hemos detectado el **103,05%**. Esto significa que obtenemos un 3,05% más de usuarios de los que deberíamos. Pero al fin y al cabo podríamos decir que hemos podido identificar todos los usuarios.

### 4.3 Análisis y comparativa de casos

En este apartado analizaremos los resultados obtenidos. En la siguiente tabla podemos observar los datos obtenidos.

Ubicación	Nº de probes	Usuarios estimados	Usuarios detectados	Porcentaje detectadas	Porcentaje probes MAC verdadera	Porcentaje probes MAC falsa
Antigua 1	75109	3909	844	21.59%	100%	0%
Antigua 2	126924	6697.59	1015	15.15%	99.8%	0.2%
Antigua 3	64349	4151.81	817	19.68%	99.8%	0.2%
Antigua 4	210730	8422.6	1063	12.62%	99.9%	0.1%
Zafra	22348	2460.62	2474	100.54%	59.3%	40.7%
Centro Sevilla	17781	1921.26	2664	138.66%	77.8%	22.2%
Torre Sevilla	17809	2231.06	2532	113.49%	65.8%	34.2%
Vivienda Sevilla	5336	155.38	207	133.22%	95.9%	4.1%
Nervión Plaza Sevilla	30114	6500	6698	103.05%	57.3%	42.7%

Tabla 1. Datos obtenidos de las lecturas.

#### 4.3.1 Proporción de la cantidad de probes procedentes de dispositivos que ofrecen MAC verdadera y MAC falsa de cada caso.

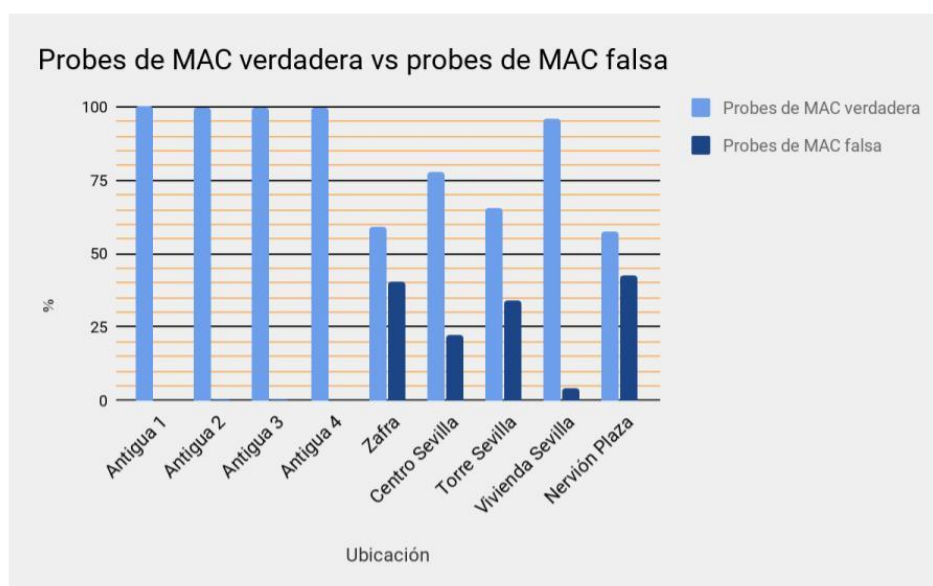


Figura 45. Balance de MACs verdaderas y falsas recibidas.

Observamos que las lecturas antiguas no proporcionan apenas tramas con MAC falsa mientras que a día de hoy hay muchas más, es más común y va en aumento.

### 4.3.2 Porcentaje de usuarios detectados frente a la ubicación donde se realizó la lectura

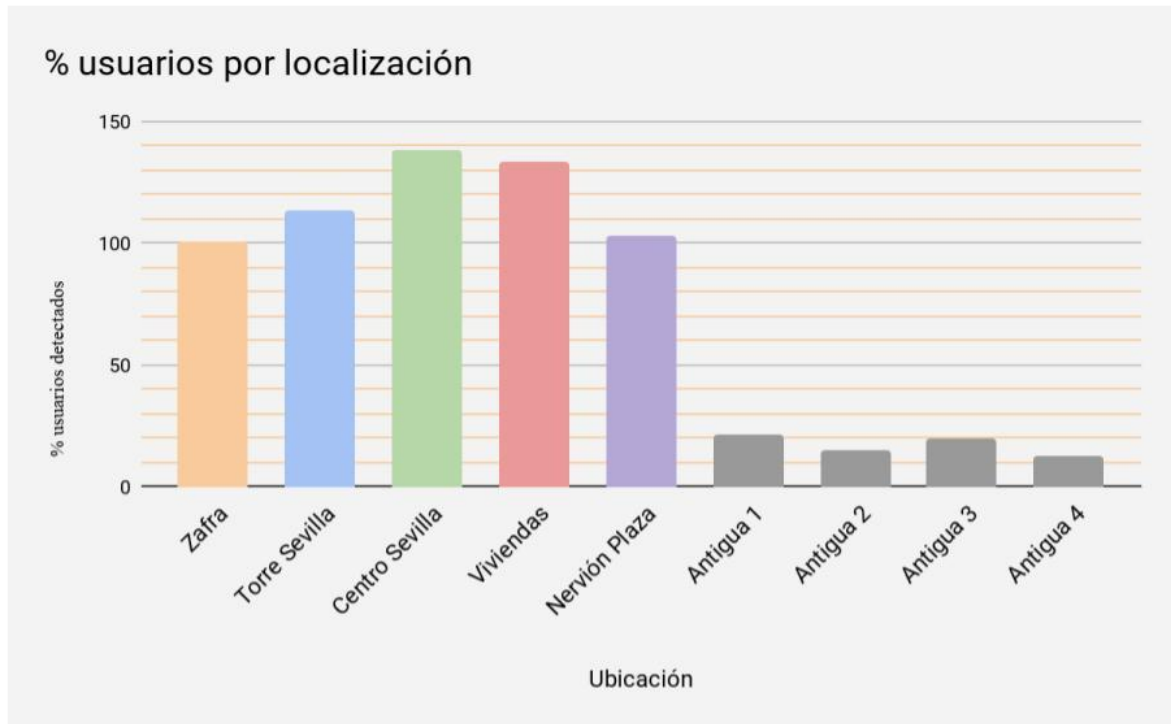


Figura 46. Comparación la cantidad de dispositivos detectados respecto al fichero de comprobación frente a la ubicación donde se tomaron las muestras.

Observando la gráfica y la Tabla 1 podemos apreciar que hay una sobredetección en los casos en los que el número de direcciones MAC verdaderas es mucho mayor que el de direcciones MAC falsas. Por lo que podemos deducir que los dispositivos al enviar su dirección MAC falsa ya están “despistando” al oyente y, por tanto, no tienen necesidad de realizar tantos cambios en sus tramas. Mientras tanto, los dispositivos que envían su dirección MAC verdadera modifican más frecuentemente sus campos, lo que influye significativamente a la hora de contabilizar el número total de usuarios.



### 4.3.3 Porcentaje de usuarios detectados frente al tiempo que duró la lectura

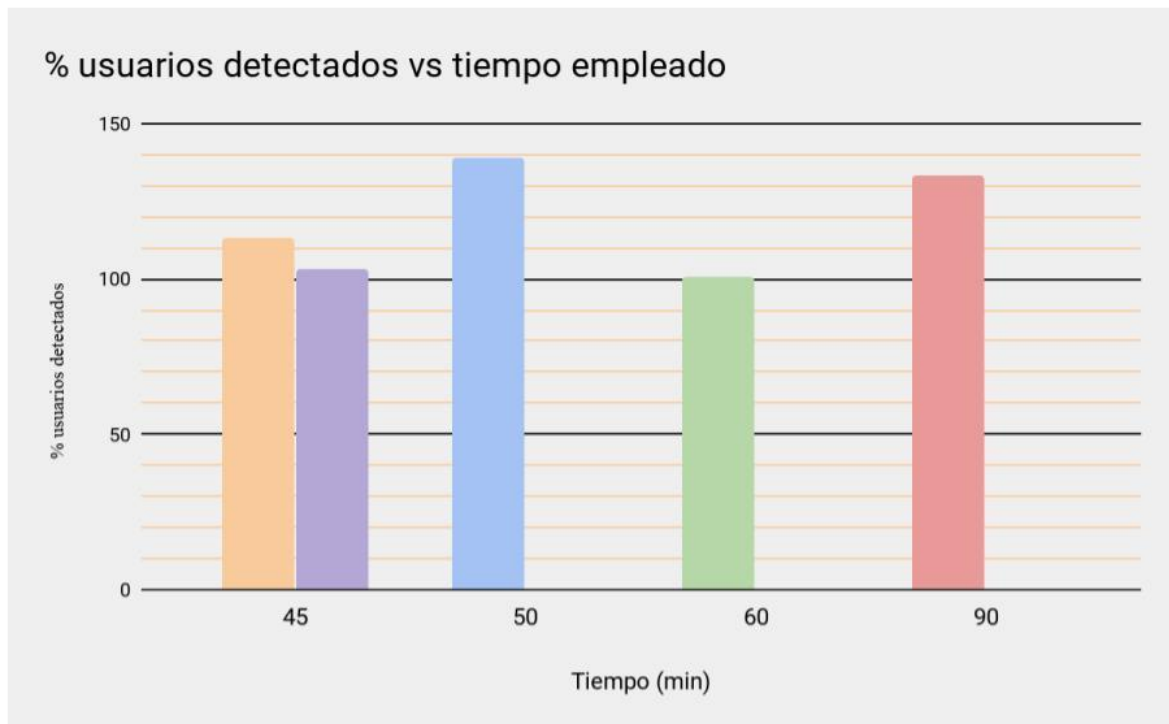


Figura 47. Comparación la cantidad de dispositivos detectados respecto al fichero de comprobación frente al tiempo que duró la toma de muestras.

Como vemos en esta gráfica, no existe relación entre el tiempo de duración de las capturas de tráfico y el porcentaje de usuarios detectados. No existe un incremento o decremento de detección en función del tiempo que se ha estado capturando, así como tampoco existe relación entre lo concurrido que sea un sitio y su detección; pues los casos donde ha habido más sobredetección son un centro comercial, donde hay mucho cambio de dispositivos, y un bloque de pisos, donde apenas hay movimiento de dispositivos.

Como las lecturas antiguas están sacadas de una base de datos donde no se especifica la duración del periodo de captura, no se incluyen en esta gráfica.

#### 4.3.4 Cantidad de usuarios calculados con ratio frente a los calculados con el script

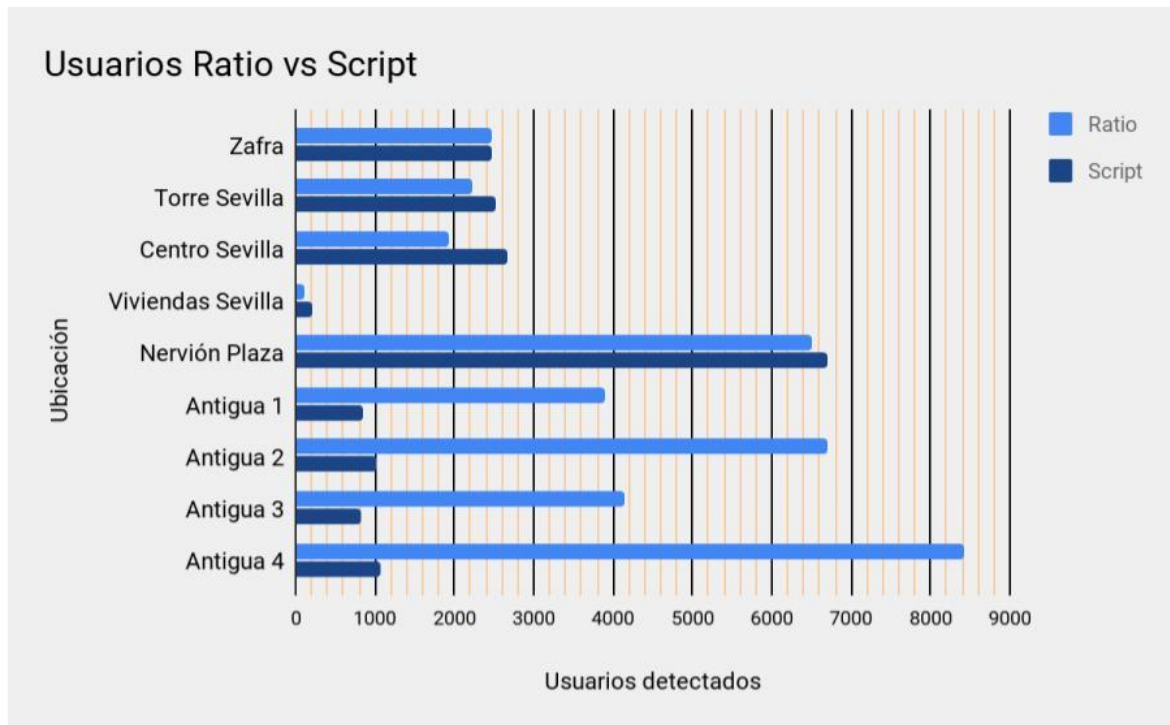


Figura 48. Comparación la cantidad de dispositivos detectados por nuestro scrip y por el fichero de comprobación en cada ubicación donde se tomaron muestras.

Aquí vemos de un solo vistazo lo que hemos estado comentando antes. Las lecturas antiguas tienen un grado de detección muy bajo, tanto que habría que replantear el sistema para poder contarlos con precisión. Mientras tanto, las lecturas actuales muestran que estamos utilizando campos con contenido demasiado dinámico, pues llegamos a detectar más dispositivos de los que realmente existen según nuestro ratio.

## 5 CONCLUSIONES Y LÍNEAS FUTURAS

*"Lo que hay que hacer es abrir la mente e imaginar cómo podrían ser las cosas, y luego hacer que eso suceda."*

- Leonard Kleinrock -

### 5.1 Conclusiones

Finalmente, podemos decir que menos es más. No por utilizar muchos campos tendremos una mejor precisión, pues no solo influye el número de campos sino la variación de los mismos. Hemos comprobado que, para las lecturas actuales, tenemos sobre-detección y deberíamos seguir investigando hasta encontrar la combinación más exacta. Sin embargo, hemos visto que cuanto mayor proporción de dispositivos falsos, mayor precisión. Probablemente esto sea debido a que los dispositivos que envían su MAC falsa tienen menor necesidad de actualizar los datos enviados al AP que el dispositivo que envía la dirección MAC verdadera.

También debemos tener en cuenta que hay campos que no aportan gran diferencia como son las Extended Rates, Extended HTCapabilities y VHT Capabilities. Estos campos, a priori no marcan gran diferencia, pero también puede ser debido a que, a día de hoy, no marcan gran diferencia o muchos dispositivos no los envían tanto como otros. No sería de extrañar que en un futuro sean campos tan variables como el SSID. Es por ello que ahora mismo es mejor quedarnos con los campos "seguros": Rates, HTCapabilities y fabricante.

Como dijimos en la introducción, si tenemos dos dispositivos iguales los identificaremos como uno solo, mientras que probablemente hemos identificado como diferentes dispositivos el mismo. Suponemos que hay cierta compensación pero habrá que seguir trabajando en ello para perfilar el algoritmo.

### 5.2 Líneas futuras

Este proyecto es solo un comienzo de lo que sería la detección unívoca de dispositivos. Por supuesto, hay mucho en lo que trabajar y muchos aspectos que perfilar, pero es un punto de partida como otros tantos ya nombrados que pueden ser útiles para futuras investigaciones.

Algunas de las posibles mejoras a este proyecto son las siguientes:

- El fingerprint generado es bastante impreciso pues, como vemos en los resultados de los experimentos, tenemos sobredetección de dispositivos y esto no debería ser así. Para ello es tan sencillo como realizar un análisis más exhaustivo de los campos o su contenido y encontrar la combinación adecuada. O quizás se deba usar otro tipo de trama que proporcione mejores combinaciones, más precisas.
- Sería interesante estudiar la influencia del sistema operativo utilizado en el dispositivo. Esto podría ayudarnos a descartar otros campos más conflictivos y quedarnos solo con los que nos aportan la información relevante.

- Para el desarrollo del script es aconsejable utilizar una herramienta de control de versiones donde ir guardando los cambios para no perder los pasos, pues se han encontrado situaciones en las que hubiera sido preferible crear una rama nueva para probar ciertos desarrollos.
- A nivel de código, los datos extraídos de las tramas quedan con el mismo formato que se extrae. Es complicado incluso con Scapy cambiar el formato de dicha información, pero existen formas de modificarlo para que sea más legible y sencillo a la hora de interpretar.
- Para un desarrollo más avanzado y automatizado podríamos utilizar algoritmos de Machine Learning para identificar los usuarios. Así serían más precisas y rápidas las detecciones.

## REFERENCIAS

---

- [1] Julien Freudiger, Short: How Talkative is your Mobile Device? An Experimental Study of Wi-Fi Probe Request
- [2] Denton Gentry, Avery Pennarun, Passive Taxonomy of Wifi Clients using MLME Frame Contents
- [3] 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, 2017
- [4] [https://github.com/yadox666/PythonScapyDot11\\_TheBook](https://github.com/yadox666/PythonScapyDot11_TheBook)
- [5] <https://www.gdpreu.org/the-regulation/key-concepts/personal-data/>
- [6] <https://seguridadyredes.wordpress.com/2009/12/03/scapy-manipulacion-avanzada-e-interactiva-de-paquetes-parte-1/>
- [7] <https://seguridadyredes.wordpress.com/2009/12/10/scapy-manipulacion-avanzada-e-interactiva-de-paquetes-parte-2/>
- [8] <https://books.google.es/books?id=Vhb0zqqPXq8C&pg=PA353&lpg=PA353&dq=DS+parameter+set+probe+request&source=bl&ots=DsxRrozibI&sig=ACfU3U1kglzID-NZx7yfhXvaYWdHwFhlmQ&hl=es&sa=X&ved=2ahUKEwjvgqe8hrHqAhVix4UKHWaNAAscQ6AEwBXoECAkQAQ#v=onepage&q&f=false>
- [9] <https://patents.google.com/patent/US10278056B2/en>
- [10] <https://www.oreilly.com/library/view/80211ac-a-survival/9781449357702/ch03.html>



# Glosario

---

802.11: Standard de normas inalámbricas

MAC: El control de acceso a medios ó Media Access Control

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos ó Institute of Electric and Electronic Engineers

AP: Punto de Acceso ó Access Point

STA: Estación ó Station

OUI: Identificador Único de Información ó Organizationally Unique Identifier

SSID: Identificador de Conjunto de Servicios ó Service Set Identifier

P2P: Punto-a-Punto ó Point-to-Point





# ANEXO A: CAMPOS DE PROBE REQUEST

CAMPOS PROBE REQUEST [8] [9] [10]	
<ul style="list-style-type: none"> <li>• Cabecera               <ul style="list-style-type: none"> <li>○ Frame control</li> <li>○ Duration</li> <li>○ Dest MAC addr</li> <li>○ Src MAC addr</li> <li>○ BSS ID</li> <li>○ Sequence control</li> </ul> </li> <li>• SSID parameters: Redes conocidas.               <ul style="list-style-type: none"> <li>○ SSID parameter set</li> <li>○ SSID length</li> <li>○ SSID</li> </ul> </li> <li>• Rates: Tasas soportadas por el dispositivo que deberán ser compatibles entre transmisor y receptor para lograr una comunicación exitosa               <ul style="list-style-type: none"> <li>○ Tag number(1)</li> <li>○ Tag length : 8</li> <li>○ Supported Rates</li> </ul> </li> <li>• Extended Supported Rates: Más tasas.               <ul style="list-style-type: none"> <li>○ Tag number(50)</li> <li>○ Tag length : 4</li> <li>○ Extended Supported Rates</li> </ul> </li> <li>• DS parameter set (opcional): Identifica el canal que está en uso.               <ul style="list-style-type: none"> <li>○ Tag number(3)</li> <li>○ Tag length : 1</li> <li>○ Current Channel</li> </ul> </li> <li>• HT Capabilities (opcional): Características físicas del dispositivo.               <ul style="list-style-type: none"> <li>○ Tag number(45)</li> <li>○ Tag length : 26</li> <li>○ HT Capabilities info</li> <li>○ A-MPDU parameters</li> <li>○ Rx Supported MCS set</li> <li>○ Ht extended capabilities</li> <li>○ Tx BF Capabilities</li> <li>○ Antenna Selection capabilities</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>• Extended HT Capabilities (opcional): Más características físicas.               <ul style="list-style-type: none"> <li>○ Tag number(127)</li> <li>○ Tag length : 8-10</li> <li>○ Ext1</li> <li>○ Ext2</li> <li>○ Ext3</li> <li>○ Ext4</li> <li>○ Ext5</li> <li>○ Ext6</li> <li>○ Ext7</li> <li>○ Ext8</li> </ul> </li> <li>• VHT Capabilities (opcional): Este campo proporciona la información usada para la gestión de tramas para preparar la operación de redes 802.11ac.               <ul style="list-style-type: none"> <li>○ Tag number(191)</li> <li>○ Tag length : 12</li> <li>○ VHT Capabilities info</li> <li>○ VHT supported MCS set</li> <li>○ Tx MCS Map</li> <li>○ Tx MCS Map</li> </ul> </li> <li>• Vendor Specific: Información que el fabricante quiere que emita el dispositivo.               <ul style="list-style-type: none"> <li>○ Tag number(221)</li> <li>○ Tag length : 105</li> </ul> </li> <li>• FILS Request parameters: Fast Initial Link Setup Request parameters. FILS es el proceso que va desde que la estación comienza a mandar tramas hasta que esta consigue establecer una conexión segura con el punto de acceso y el enlace está listo para intercambiar información. FILS Request parameters incluye los valores actuales enviados entre el dispositivo y el punto de acceso, como el threshold time.               <ul style="list-style-type: none"> <li>○ Tag number(2)</li> <li>○ Tag length : 2</li> </ul> </li> </ul>

Tabla 2. Campos Probe Request.