



Universidad
de Alcalá

TRABAJO DE FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA

ESCUELA POLITÉCNICA SUPERIOR

INVESTIGACIÓN EN TOR: CURVA ELÍPTICA, DESANONIMIZACIÓN DE SERVICIOS OCULTOS, MODOS IN-PROXY OUT-PROXY

Jaime Mariano Servera

Tutor: Manuel Sánchez Rubio

Curso 2024/2025

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

Trabajo Fin de Grado

**Investigación en TOR: curva elíptica,
desanonimización de servicios
ocultos, modos in-proxy out-proxy**

Autor:

Tutor:

TRIBUNAL:

Presidente:

Vocal 1º:

Vocal 2º:

FECHA: 13 Junio 2025

Índice

Índice de figuras	7
Resumen	8
Abstract.....	9
1. Introducción	11
2. La red Tor (<i>The Onion Router</i>)	12
2.1. ¿Qué es Tor?	12
2.2. Anatomía de la red Tor	13
2.2.1. Tipos de nodos en Tor	14
2.2.2. Protocolo Tor en modo <i>out-proxy</i>	16
2.3. Servidores <i>.onion</i>	18
2.3.1. Protocolo Tor en modo <i>in-proxy</i>	18
2.4. Herramientas de la red Tor	20
2.4.1. <i>Crawlers</i>	20
2.4.2. Buscadores (<i>indexers</i>).....	21
2.4.3. Herramientas para usuarios finales.....	22
2.4.4. Herramientas de administración	23
2.4.5. Herramientas de análisis y simulación	24
3. Criptografía de curva elíptica	25
3.1. Introducción.....	25
3.2. Fundamentos matemáticos de las curvas elípticas	26
3.3. Funcionamiento de la ECC.....	27
3.4. El protocolo <i>Elliptic Curve Diffie-Hellman</i>	31
3.5. Implementación en Tor.....	32
3.5.1. La <i>Curve25519</i>	33
3.5.2. Migración de Tor a la criptografía postcuántica.....	33
4. Desanonimización de Tor.....	35
4.1. Motivaciones	35
4.2. Técnicas	36

4.2.1.	Ataques a nivel de red	36
4.2.2.	Ataques a nivel de aplicación	40
4.2.3.	Ataques de <i>cross-device tracking</i> : <i>uBeacons</i>	43
5.	Conclusiones	52
	Referencias	54

Índice de figuras

Figura 1. Logotipo de Tor.....	12
Figura 2. Ejemplo de un circuito Tor	15
Figura 3. Esquema del funcionamiento del protocolo Tor en modo <i>out-proxy</i>	17
Figura 4. Esquema del funcionamiento del protocolo Tor en modo <i>in-proxy</i>	19
Figura 5. Ejemplo básico de curva elíptica.....	27
Figura 6. Primera iteración del protocolo de ECC	28
Figura 7. Segunda iteración del protocolo de ECC	28
Figura 8. Ejemplo de curva elíptica $y^2 = x^3 - x + 1$	29
Figura 9. Gráfico de los valores de la curva con módulo 97	29
Figura 10. Ejemplo de iteración sobre la curva elíptica con valores finitos.....	30
Figura 11. Diagrama del protocolo <i>ECDH</i>	31
Figura 12. Diagrama del protocolo <i>NTOR</i>	32
Figura 13. Diagrama del funcionamiento del padding	38
Figura 14. Diagrama de nodos Sybil	39
Figura 15. Identificación de la IP de un servicio oculto a través de una mala configuración del certificado	41
Figura 16. Uso desde Tor Browser de herramienta para comprobar la unicidad de datos de <i>fingerprinting</i>	42
Figura 17. Diagrama de desanonimización utilizando balizas de ultrasonidos.....	43
Figura 18. Espectrograma de una baliza codificando 5 caracteres usando MSFK.....	44
Figura 19. Espectrograma de la <i>baliza.wav</i>	46
Figura 20. Comando de <i>ffmpeg</i> para incrustar la baliza en un vídeo.....	46
Figura 21. Configuración del archivo <i>torrc</i>	46
Figura 22. Video infectado alojado en servicio oculto <i>.onion</i>	47
Figura 23. Baliza detectada en la aplicación	49
Figura 24. Compra del dominio <i>ubeaconsnitch.click</i> en <i>Namecheap.com</i>	49
Figura 25. Configuración del archivo <i>config.yml</i> para el túnel de <i>CloudFlare</i>	50
Figura 26. Servidor web receptor corriendo en el dominio <i>ubeaconsnitch.click</i>	50
Figura 27. Resultado final del experimento con la IP y geolocalización de la víctima..	51

Resumen

La presente investigación aborda un análisis técnico integral sobre la seguridad y el anonimato en la red Tor (*The Onion Router*), centrándose principalmente en el uso de criptografía basada en curvas elípticas dentro del protocolo Tor y los métodos actuales de desanonimización de usuarios y servicios ocultos. El trabajo combina una revisión bibliográfica exhaustiva con experimentación práctica en entornos controlados, incluyendo el estudio de casos reales y el análisis de vulnerabilidades que pueden comprometer el anonimato de la red. Además, se exploran las implicaciones de estos mecanismos tanto para la protección de usuarios legítimos como para la prevención y análisis de ciberdelitos.

El objetivo final es aportar conocimiento técnico relevante que facilite la mejora de las estrategias de ciberseguridad y la investigación criminalística en el contexto de Tor, integrando tanto la teoría criptográfica como la experiencia práctica.

Palabras clave: TOR, anonimato, curva elíptica, servicios ocultos, desanonimización, in-proxy, out-proxy, ciberseguridad, criptografía, ciberdelitos

Abstract

This research presents a comprehensive technical analysis of security and anonymity within the Tor (The Onion Router) network, with a particular focus on the use of elliptic curve cryptography in the Tor protocol and current methods for deanonymizing users and hidden services. The study combines an extensive literature review with practical experimentation in controlled environments, including the examination of real-world cases and the analysis of vulnerabilities that may compromise network anonymity. Additionally, it explores the implications of these mechanisms both for protecting legitimate users and for the prevention and investigation of cybercrime.

The ultimate goal is to provide relevant technical knowledge that facilitates the improvement of cybersecurity strategies and forensic investigations in the context of Tor, integrating both cryptographic theory and practical experience.

Keywords: Tor, anonymity, elliptic curve, hidden services, deanonymization, in-proxy, out-proxy, cybersecurity, cryptography, cybercrime

1. Introducción

Tor es una red superpuesta y distribuida sobre Internet, diseñada para ofrecer comunicaciones de baja latencia en las que el enrutamiento de los mensajes protege la identidad de los usuarios. Gracias a su arquitectura, Tor garantiza tanto la confidencialidad como la integridad de la información transmitida, evitando que terceros puedan rastrear el origen o el destino de los datos que circulan por la red.

En el presente documento se explorarán en profundidad los principios técnicos, aplicaciones y desafíos contemporáneos de la red Tor, centrándose en los mecanismos criptográficos avanzados y las técnicas de desanonimización de la red. En el ámbito criptográfico, se analiza el papel crítico de la criptografía de curva elíptica dentro del protocolo Tor, examinando su implementación en procesos como el establecimiento de circuitos seguros, la generación de claves efímeras y la autenticación de nodos. Entre otros, se destacan ventajas como la eficiencia computacional, pero también se señalan riesgos potenciales, como la vulnerabilidad ante los futuros e inminentes avances en computación cuántica.

Además de los aspectos criptográficos, la investigación explora métodos prácticos para comprometer el anonimato en servicios ocultos. Esto incluye ataques de correlación de tráfico, la explotación de configuraciones inseguras en servidores web alojados en dominios *.onion*, y el uso estratégico de nodos maliciosos (como guardianes comprometidos o *exit relays*). Además, se investiga, mediante un experimento práctico, el uso de balizas sonoras junto a páginas trampa para localizar e identificar a usuarios de la red Tor.

Entre las conclusiones más relevantes se destaca que, si bien la combinación de encriptación por curva elíptica y enrutamiento en capas ofrece robustez contra ataques pasivos, ciertas prácticas comunes en servicios ocultos (como el uso de *plugins* no auditados en servidores web o la centralización de servicios *.onion*) debilitan significativamente el anonimato. Asimismo, técnicas como el ataque de intersección de circuitos o los *timing attacks* demuestran ser efectivas en contextos específicos, aunque su eficacia disminuye en redes con alto tráfico de usuarios.

2. La red Tor (*The Onion Router*)

2.1. ¿Qué es Tor?

Imaginemos por un momento que cada vez que caminamos por la calle, alguien nos sigue, y toma nota de los lugares que visitamos, con quién mantenemos una conversación y qué compramos. Suena inquietante, ¿verdad? Pues algo parecido ocurre cada vez que navegamos por Internet: nuestro proveedor de Internet, empresas, gobiernos e incluso ciberdelincuentes pueden seguir nuestros pasos digitales con sorprendente facilidad. Pero, ¿y si existiera una forma de moverse por la red sin dejar huella, como si se fuera invisible? Aquí es donde entra en juego Tor.

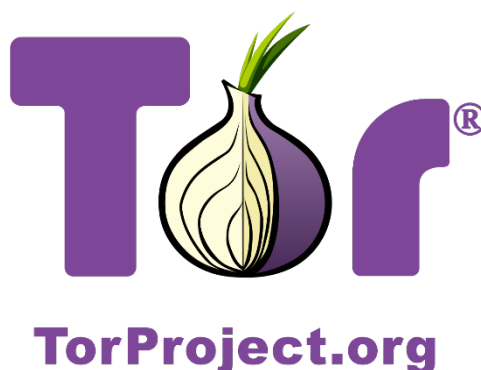


Figura 1. Logotipo de Tor [1]

Tor es mucho más que un simple programa o navegador: es una red global diseñada para proteger la privacidad y anonimato de sus usuarios en Internet. Su nombre viene de *The Onion Router* (El Enrutador Cebolla), y no es casualidad: la tecnología que utiliza se basa en capas y capas de cifrado, como las de una cebolla, para ocultar quién eres y qué haces en línea.

La motivación principal detrás de Tor es la falta de privacidad inherente al diseño original de Internet. Cuando un usuario se conecta a un sitio web de manera convencional, su dirección IP, que puede asociarse directamente con su identidad y localización, queda expuesta tanto al servidor de destino como a los intermediarios que gestionan el tráfico de red. Esta exposición permite la vigilancia, el rastreo de hábitos de navegación y, en muchos casos, la censura o persecución de individuos y colectivos.

Tor tiene sus raíces en investigaciones desarrolladas en la década de 1990 por el Laboratorio de Investigación Naval de los Estados Unidos. El objetivo inicial era crear

una forma segura y anónima de comunicación para proteger las operaciones y la información sensible del gobierno. A lo largo de los años, el desarrollo de Tor fue evolucionando y expandiéndose más allá del ámbito militar. En 2006, el proyecto Tor se consolidó formalmente como una organización sin fines de lucro bajo el nombre de The Tor Project, Inc. Desde entonces, la red y su software han sido mantenidos y mejorados por una comunidad global de desarrolladores y voluntarios comprometidos con la defensa de la privacidad y la libertad en Internet [1].

La filosofía que impulsó el diseño de Tor desde sus inicios fue la descentralización: la red debía estar compuesta por nodos operados por entidades y personas con intereses y ubicaciones diversas, para evitar que cualquier actor pudiera controlar o vigilar el tráfico de manera centralizada. La clave de Tor está en cómo mueve la información. Imaginemos que queremos enviar una carta, pero sin que nadie sepa ni quién la envía ni a quién va dirigida. Lo que hace Tor es meter esa carta en varios sobres, uno dentro de otro, y enviarla por un camino lleno de intermediarios (los llamados *nodos* o *relays*), cada uno de los cuales solo puede abrir un sobre y ver la siguiente dirección, pero nunca el contenido completo ni el origen final. Es por esto, por tanto, que esta Red mantiene el anonimato de punto a punto, donde sólo los dos nodos de los extremos poseen el mensaje completo.

2.2. Anatomía de la red Tor

Como acabamos de ver, el núcleo del funcionamiento de Tor es el llamado *enrutamiento cebolla* (*onion routing*), una técnica que garantiza el anonimato y la privacidad de las comunicaciones a través de la aplicación de múltiples capas de cifrado. Esta arquitectura hace que interceptar, rastrear o identificar a los usuarios y sus actividades resulte sumamente difícil, incluso para actores con grandes recursos.

Cuando un usuario inicia una conexión a través de Tor, su cliente (por ejemplo, el Navegador Tor) se conecta primero a una red distribuida de servidores voluntarios llamados nodos o *relays*. Antes de enviar cualquier dato, el cliente Tor descarga una lista actualizada de todos los nodos disponibles desde los llamados *directores de autoridad*, servidores confiables que mantienen información sobre el estado de la red. A continuación, el cliente selecciona aleatoriamente una secuencia de al menos tres nodos para formar un circuito. Estos nodos pueden ser de distintos tipos [2]:

2.2.1. Tipos de nodos en Tor

2.2.1.1. *Nodos de entrada (Entry guards)*

El nodo de entrada, también conocido como nodo guardián o *entry guard*, es el primer punto de contacto del usuario con la red Tor. Cuando el cliente establece un circuito, selecciona cuidadosamente este nodo y suele mantenerlo durante un periodo prolongado (semanas o incluso meses), en lugar de cambiarlo constantemente. Esta estrategia reduce el riesgo de ciertos ataques que intentan identificar a los usuarios observando cambios frecuentes en los nodos de entrada.

El nodo de entrada sabe cuál es la dirección IP del usuario, ya que recibe directamente el tráfico inicial cifrado. Sin embargo, gracias al cifrado en capas, este nodo no puede ver el contenido del mensaje ni conocer el destino final de la comunicación. Solo sabe que está recibiendo datos de un usuario de Tor y que debe reenviarlos al siguiente nodo del circuito, cuya dirección también está cifrada.

Este nodo es crucial para la seguridad, ya que, si un atacante controla el nodo de entrada, podría intentar correlacionar el tráfico de entrada y salida en la red. Por eso, la selección y permanencia del nodo guardián es una medida de protección fundamental en el diseño de Tor .

2.2.1.2. *Nodos intermedios (Middle relays)*

El nodo intermedio es el segundo eslabón en la cadena y actúa como un simple “puente” dentro del circuito. Su función principal es recibir el tráfico cifrado del nodo de entrada y reenviarlo al nodo de salida. El nodo intermedio solo conoce la dirección del nodo anterior (de entrada) y la del siguiente (de salida), pero no tiene información sobre el origen real del tráfico ni sobre el destino final fuera de la red Tor.

Este nodo nunca ve el contenido original del mensaje, ya que todavía está protegido por al menos una capa de cifrado. Su papel es esencial para aumentar el anonimato, ya que añade una capa adicional de separación entre el usuario y el destino. Además, cuantos más nodos intermedios existan en la red, más difícil se vuelve para un atacante rastrear el tráfico a través de correlaciones de tiempo o volumen de datos.

En algunos casos, Tor puede utilizar más de un nodo intermedio para incrementar aún más la seguridad, aunque esto puede afectar la velocidad de la conexión.

2.2.1.3. *Nodos de salida (exit relays)*

El nodo de salida es el último nodo del circuito Tor y es el encargado de enviar el tráfico ya descifrado al destino final en Internet, es decir, al servidor o sitio web que el usuario desea visitar. Este nodo elimina la última capa de cifrado aplicada por el cliente Tor y, por lo tanto, puede ver el contenido del mensaje si la comunicación no está cifrada de extremo a extremo (por ejemplo, si se accede a una página web sin *HTTPS*).

Sin embargo, el nodo de salida no conoce la dirección IP del usuario original, solo sabe que el tráfico viene de la red Tor. Esto significa que, aunque pueda ver el destino y el contenido de la comunicación, no puede identificar al remitente real.

El nodo de salida es, en muchos casos, el más expuesto y, a veces, el más problemático, ya que el tráfico que sale de Tor puede ser malinterpretado como proveniente del propio operador del nodo. Por esta razón, algunos servicios en Internet bloquean o restringen el acceso desde nodos de salida de Tor, y los operadores de estos nodos deben tomar precauciones legales y técnicas para protegerse.

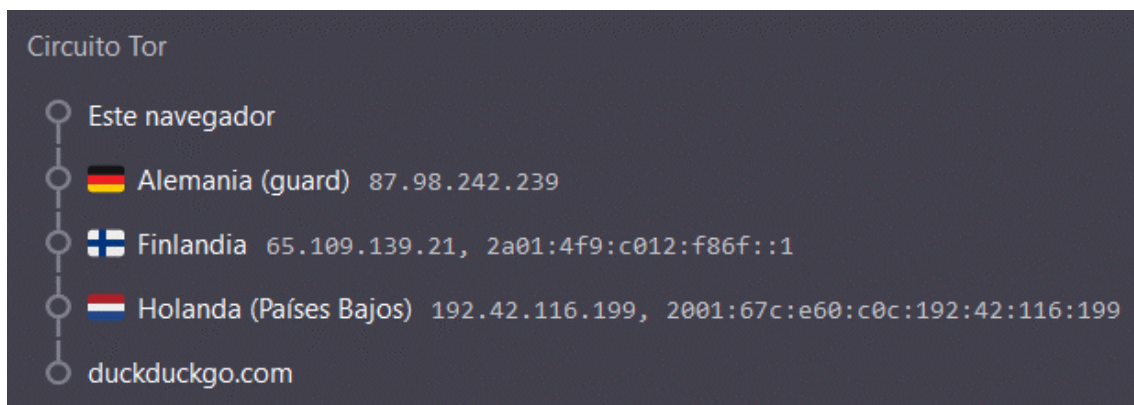


Figura 2. Ejemplo de un circuito Tor

Además de estos tres nodos principales, existe otro tipo más.

2.2.1.4. *Nodos puente (Bridge relays)*

Los nodos puente o *bridges* son nodos que no aparecen en el directorio público de Tor. Su función principal es ayudar a usuarios en países o redes

donde el acceso a Tor está bloqueado, permitiendo que puedan conectarse a la red sin ser detectados fácilmente. Estos nodos proporcionan una vía de entrada alternativa y más difícil de censurar, ya que sus direcciones IP no son públicas y se distribuyen de manera limitada y controlada [3].

Antes de enviar cualquier mensaje o solicitud, el cliente Tor cifra los datos varias veces, una por cada nodo del circuito, utilizando la clave pública de cada uno de ellos. Este proceso es análogo a meter una carta en varios sobres, uno dentro de otro, donde cada sobre solo puede ser abierto por el destinatario correspondiente.

2.2.2. Protocolo Tor en modo *out-proxy*

El proceso, en un modo *out-proxy* (que equivale a una conexión a un servidor común de Internet desde un cliente Tor), sería el siguiente:

1. Cifrado en capas:

- El mensaje original se cifra primero con la clave pública del nodo de salida.
- El resultado se cifra de nuevo, esta vez con la clave del nodo intermedio.
- Finalmente, se cifra una tercera vez con la clave del nodo de entrada.

2. Envío a través del circuito:

- El mensaje cifrado se envía al nodo de entrada. Este nodo descifra la primera capa (la suya) y reenvía el mensaje al siguiente nodo.
- El nodo intermedio recibe el mensaje, descifra su propia capa y lo reenvía al nodo de salida.
- El nodo de salida elimina la última capa de cifrado y entrega el mensaje al destino final en Internet.

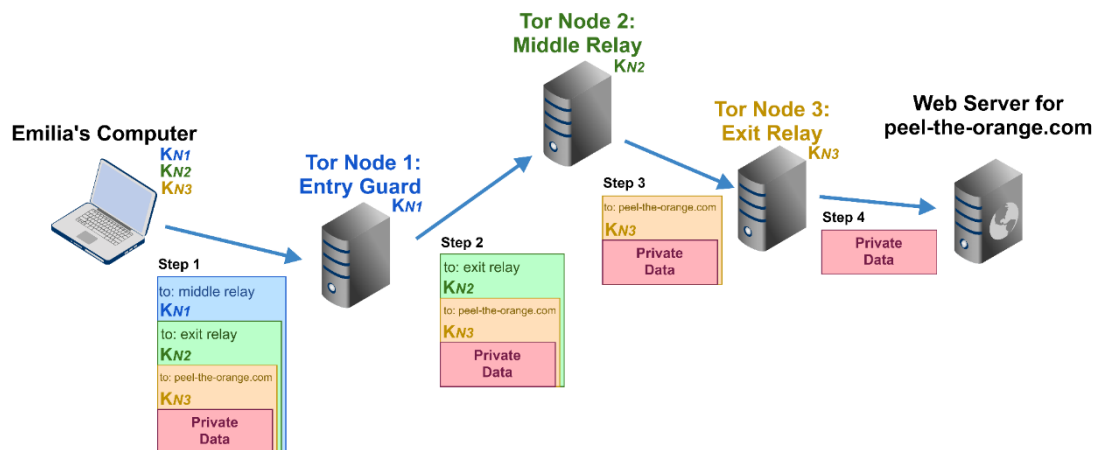


Figura 3. Esquema del funcionamiento del protocolo Tor en modo *out-proxy* [25]

En cada etapa, el nodo solo conoce la dirección del nodo anterior y del siguiente, pero nunca la ruta completa ni el contenido íntegro del mensaje (a excepción del nodo de salida, que puede ver el contenido si no está cifrado de extremo a extremo, aunque desconoce al emisor original).

Como se puede observar en la Figura 3, el cliente (Emilia, en este caso) cifra su mensaje en sentido inverso para que cada uno de los nodos que reciben el paquete puedan descifrar su parte. Para ello, se utiliza el protocolo de intercambio criptográfico *Diffie-Hellman*, que permite a dos partes generar una clave secreta compartida incluso si se comunican a través de un canal inseguro [4].

De manera simplificada, ambas partes acuerdan públicamente dos números (un número primo grande y una base), y luego cada una elige un número secreto propio. Cada parte calcula un valor público usando su secreto y lo intercambia con la otra. Finalmente, ambas partes combinan el valor recibido con su propio secreto para obtener exactamente la misma clave secreta compartida. Esta clave permite cifrar y descifrar los mensajes de forma segura, ya que un atacante, aunque vea los valores públicos intercambiados, no puede descubrir la clave secreta sin conocer los secretos privados [4]. El funcionamiento de este método de intercambio de claves se explicará con más detalle en un apartado posterior.

2.3. Servidores *.onion*

Hasta ahora, tan sólo se ha mencionado la privacidad del cliente, pero los servidores a los que accedemos desde una red común no son anónimos. Es aquí donde entra en juego una pieza clave de la red Tor, los servicios ocultos o servicios cebolla, identificados por direcciones que terminan en *.onion*.

Un servicio oculto *.onion* es un tipo especial de servidor accesible únicamente a través de la red Tor. A diferencia de los servicios tradicionales en Internet, donde la dirección IP y la ubicación del servidor pueden ser fácilmente rastreadas, los servicios ocultos ocultan tanto la identidad como la ubicación del servidor, dificultando que adversarios puedan censurarlo, identificar a sus operadores o atacar directamente la infraestructura. La dirección *.onion* funciona como un nombre de dominio exclusivo de la red Tor, que solo puede ser resuelto y accedido desde el Navegador Tor o aplicaciones compatibles. Así, tanto el usuario como el servidor permanecen protegidos, ya que la comunicación nunca sale de la red Tor y la dirección IP real del servidor nunca se expone.

La publicación de un servidor en un dominio *.onion* es sencilla: basta con instalar el navegador Tor en la máquina que se desea exponer, configurar el servicio en el archivo de configuración (*torrc*) y reiniciar el servicio Tor [5]. Automáticamente, se genera una dirección *.onion* asociada al servicio, la cual se encuentra en el archivo *hostname* del directorio correspondiente. Esta dirección es una cadena alfanumérica generada criptográficamente, lo que añade una capa extra de seguridad y dificulta el descubrimiento casual o por fuerza bruta. Los servicios ocultos pueden ser de cualquier tipo basado en TCP, como páginas web, servidores de correo, chats o FTP. En futuros apartados, se analizarán las principales herramientas y servicios que se ofrecen en la red Tor.

2.3.1. Protocolo Tor en modo *in-proxy*

En el que llamamos modo *in-proxy*, (es decir, la conexión a un servicio oculto) el proceso es diferente al acceso a un servidor tradicional, y bastante más complejo:

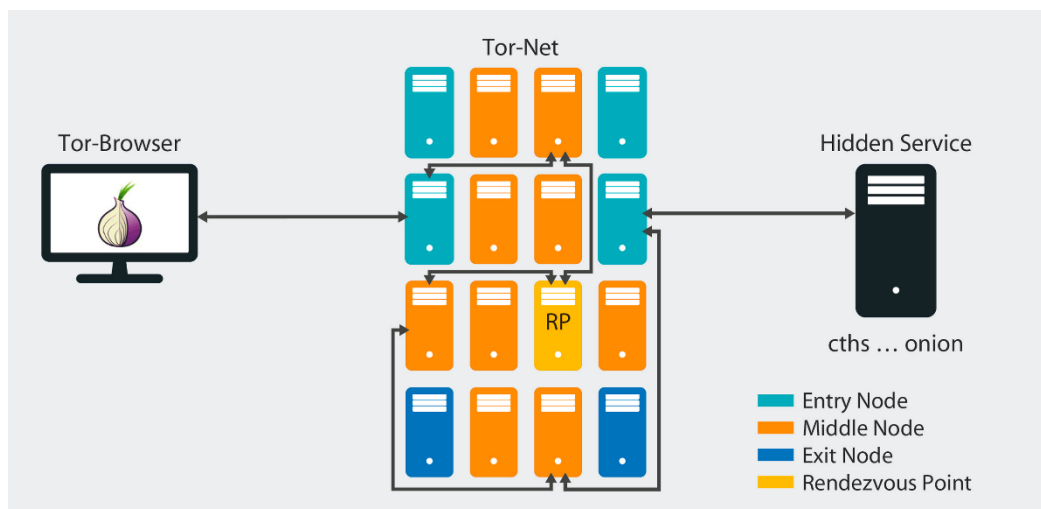


Figura 4. Esquema del funcionamiento del protocolo Tor en modo *in-proxy* [53]

1. **Selección de nodos y circuitos:** El cliente que desea conectarse a un servicio *.onion* construye dos circuitos Tor independientes:
 - Uno hacia un nodo de la red Tor elegido aleatoriamente, que actuará como punto de encuentro (*rendezvous point*).
 - Otro hacia uno de los puntos de introducción (*introduction points*) del servicio oculto, cuya dirección y clave pública ha obtenido previamente de los directorios de servicios ocultos de Tor.
2. **Establecimiento del punto de encuentro:** El cliente solicita al nodo elegido que actúe como punto de encuentro y le envía un secreto de un solo uso (*rendezvous cookie*), que servirá como identificador de la conexión.
3. **Mensaje de introducción:** El cliente cifra un mensaje con la clave pública del servicio oculto, en el que incluye la dirección del punto de encuentro, la *rendezvous cookie* y la primera parte de un intercambio criptográfico. Este mensaje se envía a través del circuito Tor hacia el punto de introducción, que lo reenvía al servidor oculto.
4. **Respuesta del servicio oculto:** El servicio oculto, al recibir el mensaje, descifra la información y, si decide aceptar la conexión, construye su propio circuito Tor hasta el mismo punto de encuentro e incluye la *rendezvous cookie* y la segunda parte del intercambio criptográfico.

5. **Emparejamiento y comunicación:** El punto de encuentro, al recibir la *cookie* de ambos lados, empareja los circuitos y permite que cliente y servidor se comuniquen a través de él, sin que ninguno conozca la identidad o dirección IP del otro. La comunicación completa viaja cifrada y pasa por seis nodos Tor en total (tres elegidos por el cliente y tres por el servidor oculto).

Durante todo este proceso, el punto de encuentro solo actúa como un repetidor de mensajes cifrados y nunca obtiene información sobre la identidad ni del cliente ni del servidor. Así, el anonimato y la privacidad quedan garantizados para ambas partes, y ningún nodo intermedio puede vincular ambos extremos de la comunicación.

2.4. Herramientas de la red Tor

La red Tor cuenta con un ecosistema diverso de herramientas diseñadas para facilitar la exploración, análisis y uso seguro de los servicios ocultos y la web oscura. Entre estas herramientas destacan los *crawlers*, buscadores, utilidades para usuarios finales, herramientas de administración de nodos y soluciones para anonimizar otras aplicaciones.

A continuación, se desarrollan en profundidad las principales categorías y ejemplos representativos.

2.4.1. *Crawlers*

Los *crawlers* o rastreadores son herramientas especializadas que exploran e indexan sitios *.onion* y otros servicios ocultos inaccesibles para los buscadores convencionales. Dada la naturaleza privada y cambiante de la *dark web*, los *crawlers* deben adaptarse a desafíos como la rotación de direcciones, el uso de *captchas*, la necesidad de anonimato y la detección de contenido malicioso.

Algunos ejemplos y proyectos destacados incluyen:

- **TorBot:** Un *crawler* de código abierto en Python que utiliza la red Tor para rastrear y recolectar información de sitios ocultos, permitiendo la extracción de contenido, metadatos y enlaces. Incorpora búsqueda por palabras clave, análisis de contenido y extracción automatizada de datos, facilitando la investigación y el monitoreo de la dark web [6].

- **CRATOR**: Este crawler avanzado está diseñado para manejar los retos de la dark web, como formularios de *login*, rotación de *cookies* y agentes de usuario aleatorios. Su arquitectura prioriza la cobertura y la robustez, permitiendo la recopilación eficiente de datos en mercados y foros ocultos, incluso frente a medidas de seguridad como *captchas* y bloqueos [7].
- **TorCrawl.py**: Un *script* en Python que permite el rastreo y extracción anónima tanto de sitios regulares como de páginas *.onion* a través de la red Tor. Es fácil de usar desde la terminal y permite ajustar la profundidad del rastreo, la velocidad y la exportación de resultados para su análisis posterior [8].
- **ACHE**: Un crawler orientado a la dark web que utiliza técnicas de análisis de enlaces y listas semilla para descubrir nuevos contenidos dentro de la red Tor, manteniendo el anonimato del rastreador y evitando la detección por parte de los sitios rastreados [9].

Estos crawlers son una herramienta fundamental en la lucha contra las actividades ilícitas de la red Tor.

2.4.2. Buscadores (*indexers*)

Dado que los sitios *.onion* no aparecen indexados en buscadores convencionales, han surgido motores de búsqueda y directorios especializados para facilitar la localización de servicios ocultos.

La diferencia principal entre *crawlers* y buscadores radica en su función dentro del proceso de acceso a la información: los crawlers son programas automatizados que recorren la web de manera autónoma, recopilando e indexando páginas y enlaces para alimentar bases de datos. Por su parte, los buscadores utilizan la información recolectada por los *crawlers* para ofrecer a los usuarios resultados relevantes cuando realizan una consulta; es decir, el *crawler* explora y construye el índice, mientras que el buscador permite localizar y acceder a esa información de forma sencilla mediante búsquedas específicas.

Cada buscador tiene su propio enfoque en cuanto a filtrado, privacidad, actualización y facilidad de uso [10]:

- **Torch**: Uno de los buscadores más antiguos y con mayor índice de sitios *.onion*. Es confiable para búsquedas profundas, aunque incluye abundante publicidad y no filtra contenidos, lo que eleva el riesgo de encontrar páginas maliciosas.

- **Ahmia:** Destaca por su política de filtrado estricto y su compromiso con la seguridad. Elimina resultados fraudulentos o peligrosos y bloquea contenido ilegal, permitiendo una navegación más segura y limpia. Ahmia es apoyado por el propio proyecto Tor y ofrece acceso tanto desde la red Tor como desde la web convencional.
- **DuckDuckGo:** Aunque es conocido por su privacidad en la web superficial, también permite búsquedas de sitios *.onion* desde la red Tor, sin rastrear ni personalizar los resultados. Es ideal para búsquedas básicas y para usuarios que priorizan la simplicidad y el anonimato, aunque carece de filtros avanzados.
- **Haystak:** Ofrece capacidades de filtrado avanzadas y una versión premium con acceso a funciones exclusivas y un índice más profundo de la dark web. Es útil para profesionales que necesitan búsquedas precisas, aunque no filtra contenido malicioso.
- **DarkWebLinks y The Hidden Wiki:** Funcionan como directorios categorizados de enlaces *.onion* verificados, facilitando el acceso a recursos confiables y actualizados. Son especialmente útiles para quienes buscan explorar la *dark web* de forma estructurada y segura.
- **OnionLand Search:** Permite búsquedas tanto en la *dark web* como en la web convencional, con una interfaz intuitiva y sugerencias de búsqueda. Sin embargo, puede requerir la activación de JavaScript, lo que puede afectar gravemente a la privacidad.
- **DeepSearch:** Motor de búsqueda *open source* centrado en la precisión y la eliminación de enlaces basura, ideal para búsquedas dirigidas y eficientes en el espacio *.onion*.

Estos buscadores y directorios son esenciales para navegar la red Tor, ya que, como ya se ha mencionado, los servicios ocultos no son accesibles sin conocer la dirección exacta.

2.4.3. Herramientas para usuarios finales

Existen numerosas utilidades que permiten a los usuarios aprovechar Tor para diferentes propósitos, desde la navegación segura hasta el acceso a servicios ocultos o la protección de la privacidad en otras aplicaciones [11]:

- **Orbot y Orfox:** Aplicaciones para Android que permiten enrutar el tráfico de cualquier app a través de Tor y navegar anónimamente.

- **Tor Browser:** El navegador oficial de Tor, disponible para múltiples plataformas, que facilita el acceso seguro y anónimo tanto a la web superficial como a los servicios *.onion*.
- **GetTor:** Servicio que distribuye el navegador Tor por correo electrónico, útil en regiones donde la descarga directa está bloqueada.
- **TorBirdy:** Extensión para *Thunderbird* que enruta el correo electrónico por Tor, añadiendo una capa de anonimato a las comunicaciones.
- **Torsocks:** Herramienta que permite a otras aplicaciones de línea de comandos enrutar su tráfico por Tor sin necesidad de configuración avanzada.
- **Tor2web:** Servicio que permite acceder a sitios *.onion* desde la web convencional, sacrificando el anonimato del usuario a cambio de mayor accesibilidad.
- **Whonix:** Sistema operativo que enruta automáticamente por Tor todo su tráfico.
- **TAILS:** Distribución preconfigurada para que todo el tráfico pase de forma segura a través de Tor, al igual que Whonix, y no deje ningún rastro en el sistema local.

2.4.4. Herramientas de administración

Para gestionar nodos de la red Tor o monitorizar su funcionamiento, existen utilidades específicas [11]:

- **Anonymizing Relay Monitor (Arm):** Monitor en consola para visualizar el estado y rendimiento de un nodo Tor en tiempo real.
- **Weather:** Servicio de notificaciones automáticas para operadores cuando su relay está caído o inaccesible.
- **Tor Bulk Exitlist (TorBEL):** Herramienta para identificar si una IP pertenece a un nodo de salida Tor, útil para administradores de servicios web.
- **TorFlow:** Conjunto de herramientas para monitorizar el ancho de banda y detectar nodos maliciosos o mal configurados en la red.
- **tor-relay-bootstrap y tor_box:** Scripts y configuraciones para desplegar y mantener relays de Tor de manera automatizada, incluso en dispositivos como *Raspberry Pi*.

2.4.5. Herramientas de análisis y simulación

Estas utilidades permiten desde simular el comportamiento de la red bajo distintos escenarios, hasta analizar la seguridad, detectar censura o enrutar todo el tráfico de un sistema a través de Tor para maximizar el anonimato. A continuación, se describen algunas de las herramientas más destacadas:

- ***Tor Path Simulator (TorPS)***: Simulador de selección de rutas en Tor, útil para investigación y análisis de seguridad. Permite modelar y analizar cómo se seleccionan los circuitos en la red Tor, facilitando el estudio de posibles vulnerabilidades y la evaluación del anonimato ofrecido por la red [12].
- ***Ooni Probe***: Herramienta diseñada para detectar censura y manipulación de tráfico en la conexión local. *Ooni Probe* realiza pruebas que identifican si ciertos sitios web están bloqueados, si existen sistemas de espionaje o censura, y mide la velocidad de la conexión. Es ampliamente utilizada para monitorizar el nivel de intromisión que gobiernos o proveedores de servicios pueden ejercer sobre el acceso a Internet [13].
- ***dnscrypt-proxy***: Proxy DNS configurable para enrutar consultas DNS a través de Tor, reforzando la privacidad del usuario. Esta herramienta cifra las peticiones DNS entre el cliente y el servidor de nombres, evitando que terceros puedan espiar o manipular las consultas, lo que es esencial para mantener el anonimato incluso en las resoluciones de nombres de dominio [14].
- ***Tortilla y leaf***: Soluciones que permiten enrutar todo el tráfico de un dispositivo, incluyendo las consultas DNS, a través de Tor. Herramientas como *leaf*, en continuo desarrollo, facilitan que incluso aplicaciones que no están diseñadas para funcionar con Tor puedan beneficiarse del anonimato de la red, proporcionando una capa de protección a nivel de sistema operativo y no solo a nivel de aplicación [15] [16] [17].
- ***Shadow***: Simulador avanzado que permite recrear el comportamiento de la red Tor en un entorno controlado. Shadow ejecuta instancias virtuales de nodos y aplicaciones reales, permitiendo modificar parámetros como el número de nodos, el ancho de banda o la latencia para analizar el rendimiento, la seguridad y el comportamiento de Tor bajo diferentes condiciones de red [18].

3. Criptografía de curva elíptica

3.1. Introducción

La criptografía de curva elíptica (ECC, por sus siglas en inglés) es una rama de la criptografía de clave pública que se basa en las propiedades matemáticas de las curvas elípticas definidas sobre cuerpos finitos. A diferencia de los sistemas criptográficos clásicos como RSA o ElGamal, que requieren claves de gran tamaño para alcanzar altos niveles de seguridad, la ECC ofrece una seguridad equivalente con claves mucho más pequeñas, lo que la hace especialmente eficiente y atractiva para aplicaciones modernas donde los recursos computacionales y el ancho de banda son limitados [19].

Para que un sistema criptográfico de clave pública funcione correctamente, es fundamental contar con un conjunto de algoritmos cuya operación en un sentido sea sencilla de calcular, pero extremadamente difícil de revertir. Este principio se conoce como función trampa (*trapdoor function*), que es aquella que puede ser ejecutada fácilmente en una dirección, pero cuya inversión —es decir, obtener el dato original a partir del resultado— requiere un esfuerzo computacional desproporcionado [20].

Por ejemplo, en el algoritmo RSA, la operación sencilla consiste en multiplicar dos números primos grandes. Sin embargo, el proceso inverso, es decir, descomponer el producto resultante en sus factores primos, es computacionalmente muy costoso [20].

Una curva elíptica, en el contexto criptográfico, es el conjunto de puntos que satisfacen una ecuación algebraica específica, habitualmente de la forma $y^2 = x^3 + ax + b$, donde a y b son coeficientes pertenecientes a un cuerpo finito \mathbb{F}_p y deben cumplir ciertas condiciones para evitar singularidades [20].

En el caso de la criptografía de curva elíptica, la función trampa se basa en la operación de suma de puntos sobre la curva. Es fácil calcular el resultado de sumar un punto P consigo mismo varias veces para obtener otro punto Q (es decir, calcular $Q = kP$ para un escalar k), pero resulta computacionalmente inviable, incluso para los ordenadores más potentes, determinar el valor de k conociendo solo P y Q . Este problema, conocido como el problema del logaritmo discreto en curvas elípticas (ECDLP), es la *trapdoor function* que sustenta la seguridad de los sistemas ECC [20].

Esta dificultad permite que los sistemas basados en ECC sean resistentes a ataques conocidos y, al mismo tiempo, utilicen claves de menor longitud, lo que se traduce en mayor eficiencia y menor consumo de recursos.

La ECC se ha convertido en un estándar de la industria y es utilizada en una amplia variedad de aplicaciones, desde tarjetas inteligentes y sistemas de autenticación hasta comunicaciones seguras, *blockchain* y, por supuesto, en redes de anonimato como Tor, donde es fundamental para el establecimiento seguro de circuitos y para garantizar el anonimato y la privacidad de las comunicaciones.

3.2. Fundamentos matemáticos de las curvas elípticas

Formalmente, una curva elíptica E definida sobre un cuerpo \mathbb{K} es una curva algebraica plana y no singular dada por una ecuación cúbica en dos variables. La forma más general de la ecuación de una curva elíptica es la forma corta de la ecuación de Weierstrass:

$$y^2 = x^3 + Ax + B$$

con $A, B \in \mathbb{K}$, siempre que se cumpla la condición de no singularidad $4A^3 + 27B^2 \neq 0$, que garantiza que la curva no tiene puntos singulares (es decir, no presenta cúspides ni autointersecciones) [21].

En criptografía, las curvas elípticas se definen sobre cuerpos finitos, típicamente sobre \mathbb{F}_q donde $q = p^m$ y p es primo [19]. Esto permite trabajar con un número finito y controlado de puntos, lo que es fundamental para la seguridad de los algoritmos criptográficos. Los estándares internacionales, como los definidos por NIST, especifican curvas concretas sobre cuerpos finitos primos (por ejemplo, P-256 sobre \mathbb{F}_q) o binarios (por ejemplo, B-233 sobre \mathbb{F}_{2^m}), cada una con parámetros predefinidos y un punto generador de gran orden [21].

Una de las propiedades clave de las curvas elípticas es que el conjunto de sus puntos, junto con un punto en el infinito (denotado como \mathcal{O}), forma un grupo abeliano bajo una operación de suma definida geoméricamente [22]. Dados dos puntos P y Q en la curva, su suma $R = P + Q$ se obtiene trazando la línea que los une y encontrando el tercer punto de intersección con la curva; reflejando este punto respecto al eje x (en el que la curva

elíptica es simétrica) se obtiene el resultado de la suma. Esta operación cumple las propiedades de grupo: existencia de elemento neutro (\mathcal{O}), existencia de inversos, asociatividad y conmutatividad.

La seguridad de la criptografía de curva elíptica se fundamenta en la dificultad del ECDLP. Dado un punto generador P de orden grande y otro punto Q en el subgrupo generado por P , el problema consiste en encontrar el entero k tal que:

$$Q = kP$$

Como ya se ha mencionado anteriormente, a pesar de que calcular Q a partir de k y P es sencillo, determinar k a partir de P y Q es computacionalmente inviable con los algoritmos actuales, siempre que los parámetros sean suficientemente grandes [23]. No existen algoritmos subexponenciales generales conocidos para resolver el ECDLP, y los mejores métodos conocidos (como el algoritmo rho de Pollard [24]) tienen una complejidad de $O(\sqrt{r})$, donde r es el orden del punto generador.

3.3. Funcionamiento de la ECC

Ahora que se han establecido los principios matemáticos, se explicará con detalle cómo funciona el protocolo criptográfico [25].

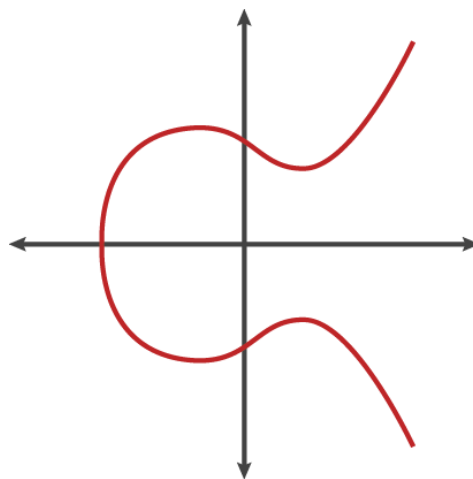


Figura 5. Ejemplo básico de curva elíptica [20]

Partiendo de un ejemplo genérico de curva elíptica, mostrado en la Figura 5, podemos ver que, efectivamente, es simétrica: cualquier punto puede reflejarse en su eje x y la curva seguirá siendo la misma.

Imaginemos que tenemos un punto inicial P sobre la curva. Si trazamos su tangente, en algún momento cortará a la curva por un segundo punto, $-2P$, el cual reflejaremos por el eje x , y obtendremos el punto $2P$ (Figura 6).

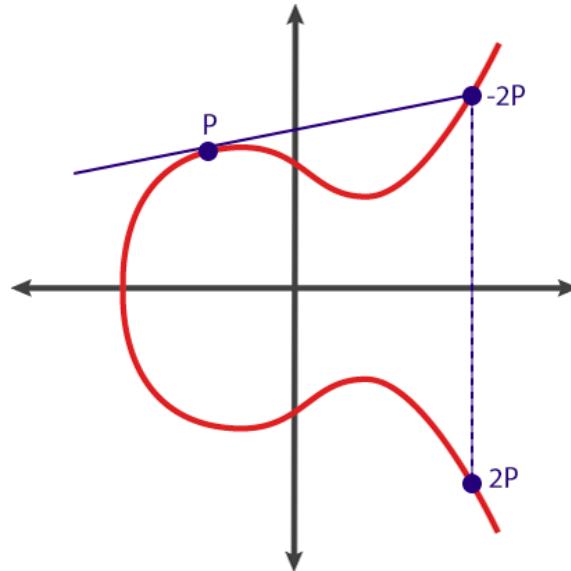


Figura 6. Primera iteración del protocolo de ECC

Ahora, de nuevo desde el punto P , trazaremos una recta hacia el punto $2P$, y donde corte a la curva, lo invertiremos de nuevo para obtener un nuevo punto $3P$ (Figura 7).

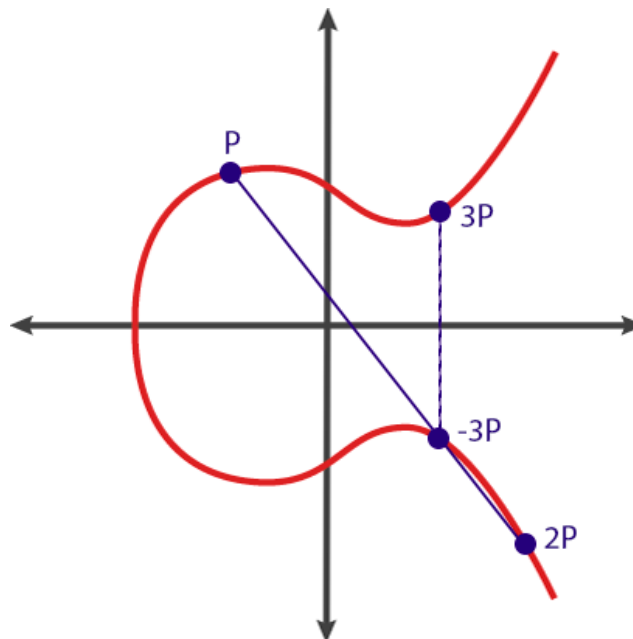


Figura 7. Segunda iteración del protocolo de ECC

De la misma manera, repetiremos este proceso k veces, hasta que terminemos con un punto final que llamaremos Q . Como ya se ha mencionado anteriormente, obtener k (es decir, el número de veces que hemos iterado en el algoritmo) a partir de P (el punto inicial, que en nuestro ejemplo era A) y Q es computacionalmente inviable.

En la realidad, la encriptación de curva elíptica no funciona exactamente así. La curva se dispone dentro de unos límites finitos, definidos como \mathbb{F}_q , donde q es el módulo que se utiliza para delimitar los ejes.

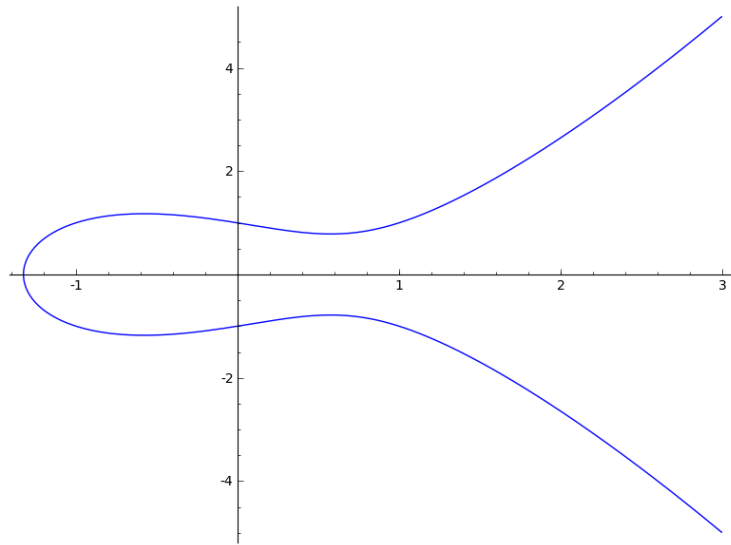


Figura 8. Ejemplo de curva elíptica ($y^2 = x^3 - x + 1$) [20]

Siguiendo esta idea, y usando como ejemplo la curva elíptica de la Figura 8, plasmaremos en una gráfica los valores enteros que toma la curva, dentro de unos límites marcados por el módulo 97.

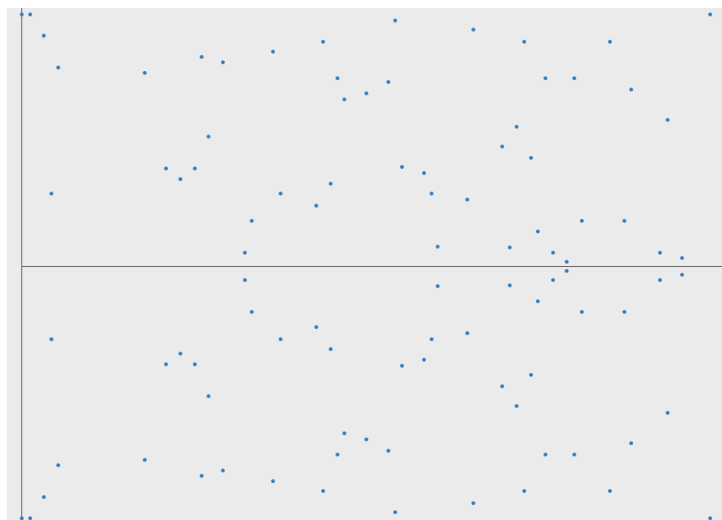


Figura 9. Gráfico de los valores de la curva con módulo 97 [20]

Aunque ya no lo parezca, el gráfico de la Figura 9 sigue siendo la misma curva que antes. Ahora, cuando un valor se pasa de los límites, vuelve al inicio, por lo que tenemos muchos puntos distintos superpuestos (por ejemplo, el punto $(1,1)$ y el punto $(98,98)$ estarán en el mismo sitio), pero aun se puede apreciar la simetría horizontal.

De esta forma, podemos aplicar el mismo concepto que antes para realizar sumas de un mismo punto. En la Figura 10 podemos ver una iteración, en la que trazamos la recta del punto A al punto B y esta escapa del eje y , continuando desde abajo en la misma dirección hasta que escapa también del eje x y llega al punto $-C$. Tras ello, y de la misma manera que antes, invertimos el punto en el eje x y llegamos al punto C .

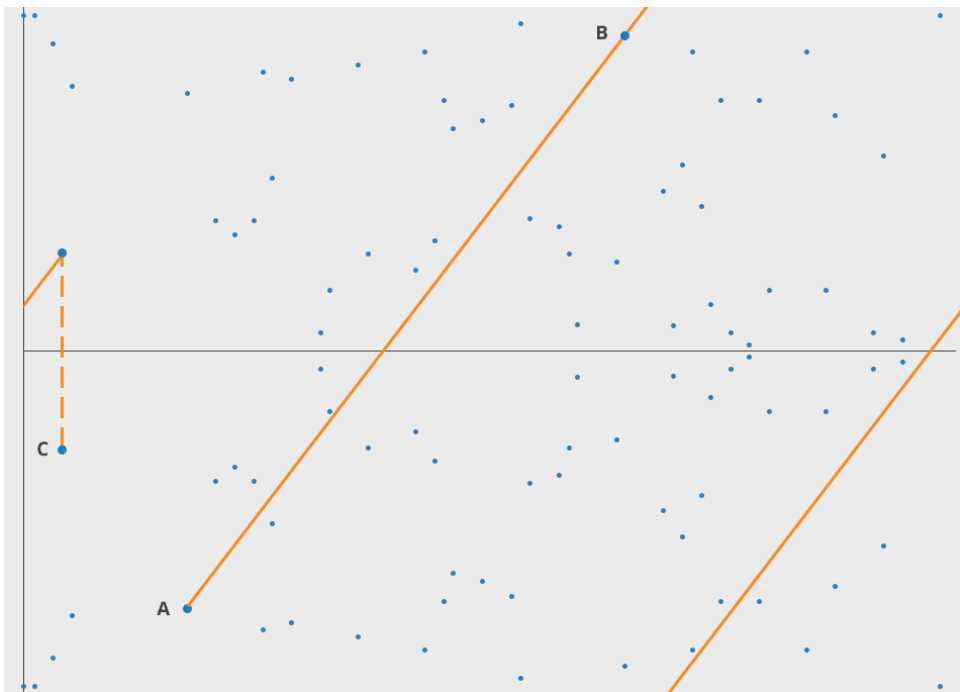


Figura 10. Ejemplo de iteración sobre la curva elíptica con valores finitos [20]

Por supuesto, una curva elíptica utilizada en criptografía real, como la *secp256k1* (que utiliza un módulo de 256 bits) tiene un valor de q muchísimo más grande que los ejemplos mostrados, por lo que imaginemos que en la gráfica hay aproximadamente tantos puntos como átomos en el universo [26].

Concretamente, la curva elíptica que Tor utiliza es la *Curve25519*, la cual analizaremos más detalladamente en un apartado posterior [27].

3.4. El protocolo *Elliptic Curve Diffie-Hellman*

El protocolo *Elliptic Curve Diffie-Hellman* (ECDH) es una variante del famoso intercambio de claves *Diffie-Hellman*, pero adaptado al entorno de las curvas elípticas.

ECDH permite que dos partes, que no han compartido información secreta previamente, acuerden de forma segura una clave secreta común a través de un canal inseguro.

El funcionamiento del protocolo parte de la existencia de una curva elíptica y un punto inicial público, conocido por todos los participantes. Cada usuario genera su propia clave privada, que no es más que un número aleatorio muy grande, y a partir de ella calcula su clave pública multiplicando ese número por el punto inicial de la curva. Por ejemplo, si *Alice* elige como clave privada el número a , su clave pública será el punto $A = aG$, donde G es el punto inicial. De manera análoga, *Bob* escoge su clave privada b y calcula su clave pública $B = bG$.

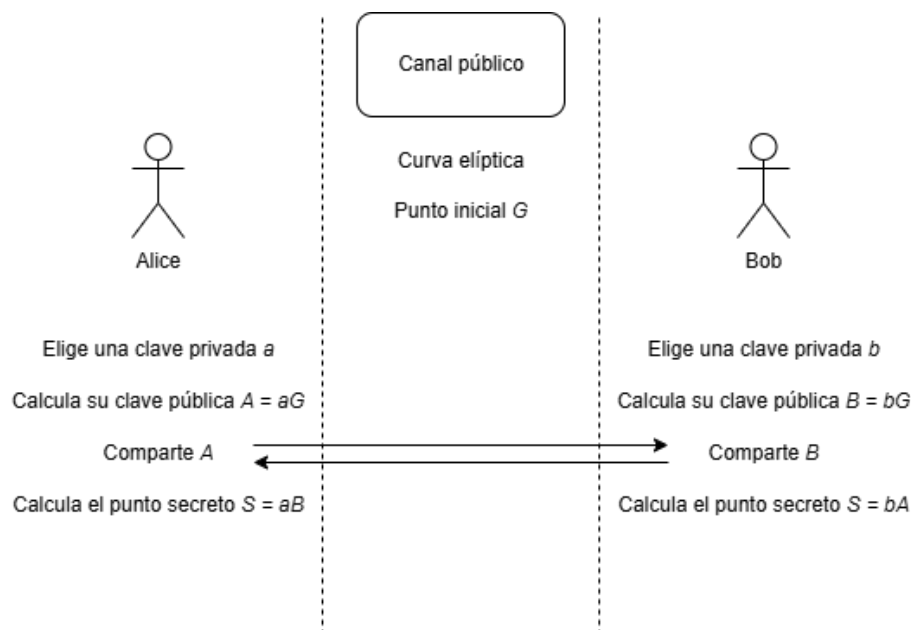


Figura 11. Diagrama del protocolo *ECDH*

Ambos usuarios intercambian únicamente sus claves públicas a través del canal inseguro. Lo interesante del protocolo es que, una vez que cada uno conoce la clave pública del otro, pueden calcular de forma independiente la misma clave secreta compartida. *Alice* toma la clave pública de *Bob* y la multiplica por su clave privada, obteniendo $S = aB = a(bG) = (ab)G$. Por su parte, *Bob* realiza el mismo cálculo, pero en el orden inverso,

multiplicando la clave pública de *Alice* por su propia clave privada, es decir, $S = bA = b(aG) = (ba)G$. Como la multiplicación es conmutativa, ambos llegan exactamente al mismo punto de la curva, que servirá como clave compartida secreta entre ellos.

3.5. Implementación en Tor

Como ya sabemos, cuando un cliente de Tor desea construir un circuito para enviar datos de forma anónima, inicia una serie de intercambios de claves con cada uno de los nodos que formarán parte de ese circuito. Este proceso se realiza de manera incremental, negociando una clave secreta independiente con cada nodo mediante un intercambio de ECDH, específicamente la variante NTOR, y utilizando la función *Curve25519*.

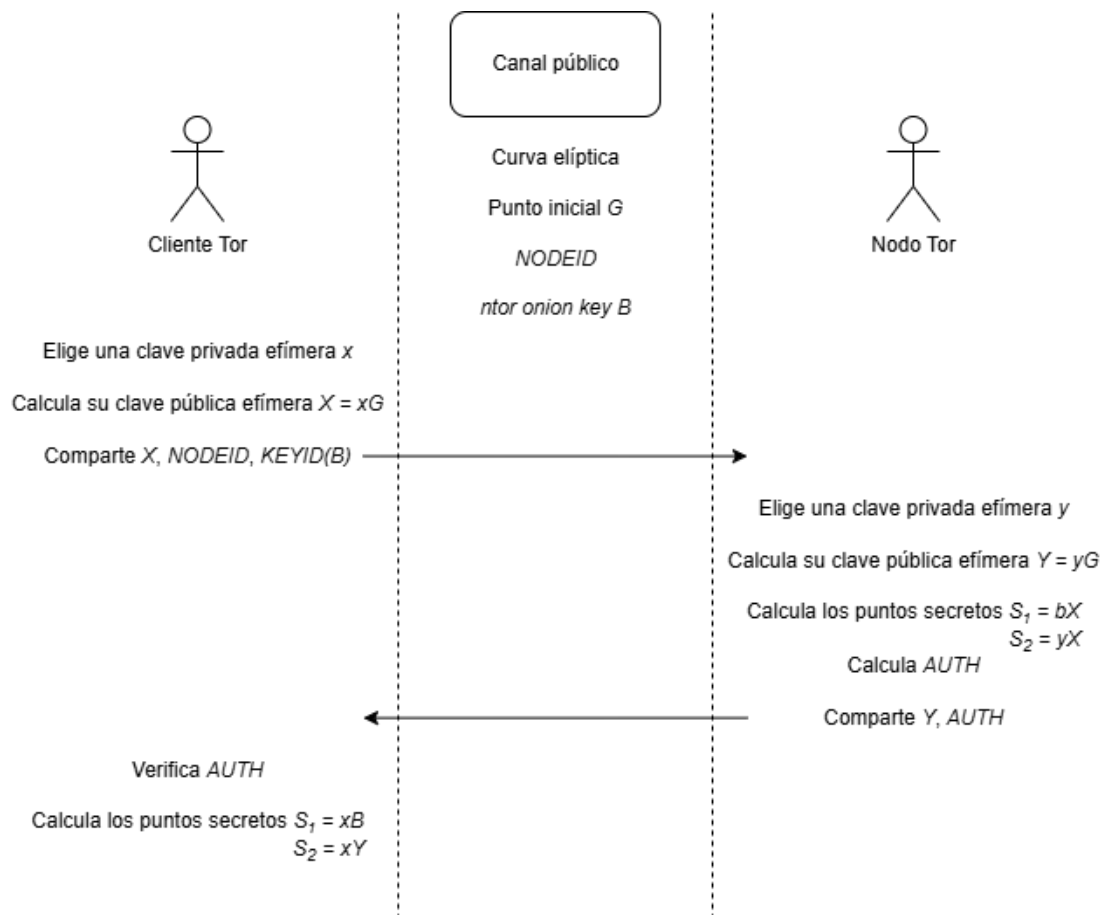


Figura 12. Diagrama del protocolo *NTOR*

A diferencia del ECDH clásico, NTOR introduce mecanismos adicionales para autenticar a los nodos y garantizar propiedades como la *forward secrecy*, es decir, que la exposición

de una clave privada en el futuro no comprometa la confidencialidad de las sesiones pasadas. Cada nodo de Tor posee una clave pública a largo plazo y genera, para cada circuito, una clave efímera. El cliente Tor, al crear un nuevo circuito, utiliza la clave pública del nodo de entrada y una clave efímera propia para realizar el intercambio de claves (Figura 12). Este procedimiento asegura que cada circuito tenga una clave simétrica única y que ningún nodo pueda conocer la clave de cifrado de los demás, ni reconstruir la ruta completa del circuito [28].

3.5.1. La *Curve25519*

La seguridad y eficiencia de NTOR en Tor dependen en gran medida de la curva elíptica que utiliza, la *Curve25519*, una curva de Montgomery definida como:

$$y^2 = x^3 + 486662x^2 + x$$

con un dominio definido sobre un campo finito \mathbb{F}_q , donde $q = 2^{255} - 19$, un número primo, lo que proporciona un espacio de claves extremadamente grande y seguro [29]. La función, además, está optimizada para cálculos rápidos y para evitar errores de implementación que puedan comprometer la seguridad [30].

Esta curva fue diseñada por Daniel J. Bernstein en 2006 y se ha convertido en un estándar moderno para criptografía de curva elíptica, especialmente en aplicaciones que requieren alto rendimiento y seguridad frente a vulnerabilidades conocidas [30].

Entre sus características más destacadas se encuentran su resistencia a ataques de canal lateral, la ausencia de parámetros elegidos de forma arbitraria o sospechosa (lo que elimina posibles *backdoors* criptográficas [31]), y su eficiencia: permite realizar operaciones de intercambio de claves de manera muy rápida, incluso en dispositivos con recursos limitados, como teléfonos móviles o routers [30].

3.5.2. Migración de Tor a la criptografía postcuántica

La adaptación de Tor a la criptografía postcuántica es un desafío urgente y estratégico ante el avance de la computación cuántica y la amenaza que representan algoritmos como el de Shor y Grover, capaces de romper los esquemas criptográficos tradicionales sobre los que se apoya actualmente la red [32]. El riesgo principal es el conocido como “*harvest now, decrypt later*”, en el

que un adversario almacena comunicaciones cifradas hoy con la esperanza de poder descifrarlas en el futuro cuando disponga de tecnología cuántica avanzada.

El *circuit-extension handshake* (protocolo de extensión de circuito, en español) es el primer componente que debería migrarse, ya que es el punto donde se negocian las claves para el cifrado de cada circuito. Para ello, se propone la adopción de esquemas híbridos que combinen mecanismos clásicos como *X25519* (el acrónimo de la *Curve25519* + *Diffie-Hellman*) con nuevos algoritmos postcuánticos, especialmente *KEMs* (*Key Encapsulation Mechanisms* [33]) basados en retículas (*lattice-based*), como *NTRU* [34] o *CRYSTALS Kyber* [35] [36] [37]. Estos esquemas permiten que, si en el futuro se rompe uno de los algoritmos, la seguridad del sistema aún dependa del otro, proporcionando una defensa en profundidad frente a ataques desconocidos e inminentes.

Aunque ya existen varias propuestas y experimentos de integración de criptografía postcuántica en Tor, como los mencionados, la comunidad científica recomienda avanzar mediante modelos híbridos y pruebas controladas, dado que muchos de los algoritmos postcuánticos aún son relativamente recientes y podrían descubrirse vulnerabilidades inesperadas [32].

Dado que no es el tema principal del proyecto de investigación, no se entrará en profundidad ni detalles técnicos en este apartado, pero sí que recomiendo mucho leer la bibliografía de los protocolos, especialmente la de *Kyber*.

4. Desanonimización de Tor

La desanonimización es el proceso mediante el cual se busca descubrir la identidad real de un usuario o de un servicio que opera bajo el amparo del anonimato en Internet. En este apartado, se llevará a cabo un análisis de las diferentes técnicas conocidas que existen para este fin.

4.1. Motivaciones

Las motivaciones para intentar desanonimizar usuarios y servicios ocultos en la red Tor son diversas y reflejan tanto intereses legítimos como actividades ilícitas. Por un lado, las fuerzas y cuerpos de seguridad del Estado y organismos de inteligencia buscan identificar a los responsables de actividades ilegales que se refugian en el anonimato de Tor, como la gestión de mercados negros, la venta de drogas, armas, documentos falsificados, la distribución de pornografía infantil o la radicalización terrorista. La desanonimización, en estos casos, se justifica como una herramienta esencial para la persecución del delito y la protección de la seguridad pública.

Aun así, creo importante destacar que Tor no es usado mayoritariamente con fines ilícitos. Según un estudio realizado en 2019, en un día promedio, aproximadamente el 6,7% de los clientes de la red Tor a nivel global utilizan la red para conectarse a servicios ocultos que son predominantemente empleados para actividades ilegales [38].

No obstante, los datos también revelan que la distribución de los usos potencialmente dañinos y beneficiosos no es uniforme, sino que se concentra principalmente en regímenes políticamente libres. Específicamente, la tasa promedio de uso probablemente malicioso de Tor en países clasificados como “no libres” por *Freedom House* es solo del 4,8%, mientras que en países “libres” el porcentaje de usuarios que realizan actividades ilícitas respecto al uso total de Tor diario es casi el doble, alrededor del 7,8% [38].

En realidad, tiene sentido pensar que esta diferencia no significa directamente un número menor de personas participando en actividades ilegales, si no un porcentaje menor en dichas zonas, posiblemente debido a que hay mucha gente que debe utilizar Tor para su vida diaria en países represivos.

En consecuencia, aunque Tor es frecuentemente asociado en los medios y en el discurso público con actividades delictivas, la realidad es que la mayoría de sus usuarios emplean

la red para fines legítimos, como la protección de la privacidad, la elusión de la censura y la defensa de la libertad de expresión, especialmente en contextos restrictivos o bajo vigilancia estatal. Esta dualidad en los usos de Tor explica por qué la desanonimización es un tema tan sensible y polémico, pues puede servir tanto para combatir delitos graves como para poner en riesgo los derechos fundamentales de muchas personas.

4.2. Técnicas

Existen multitud de técnicas conocidas distintas, algunas probadas de manera práctica y otras sólo teóricas. Estas técnicas varían en complejidad, alcance y efectividad, y pueden dirigirse tanto contra usuarios individuales como contra servicios ocultos o incluso contra la propia infraestructura de la red. Algunas de estas estrategias se basan en el análisis y correlación del tráfico de red, otras explotan vulnerabilidades en el software o en la configuración de los servicios, y también existen métodos que combinan ataques a nivel de red con técnicas de ingeniería social o malware. Además, se analizará en profundidad una técnica que utiliza *cross-device tracking* y balizas de ultrasonidos.

A continuación, se describen algunas de las categorías más relevantes de técnicas de desanonimización en Tor, analizando su funcionamiento y los escenarios en los que pueden resultar efectivas.

4.2.1. Ataques a nivel de red

Este tipo de ataques explotan la propia arquitectura del *onion routing* y la gestión de los circuitos para intentar identificar el origen o el destino real del tráfico. Estos ataques pueden ser llevados a cabo tanto por actores con capacidades limitadas (por ejemplo, operadores de nodos maliciosos) como por adversarios con acceso privilegiado a la infraestructura de red (proveedores de servicios, grandes organizaciones o incluso estados).

Análisis de tráfico y correlación extremo a extremo

El análisis de tráfico es una de las técnicas más potentes y estudiadas para la desanonimización en Tor. Consiste en observar y analizar patrones en el flujo de datos que atraviesa la red, como el tamaño, el tiempo y la frecuencia de los paquetes, con el objetivo de correlacionar la actividad entre el nodo de entrada y el nodo de salida. Si un atacante es capaz de monitorizar ambos extremos de un circuito, puede correlacionar los

patrones de tráfico y, con un alto grado de confianza, vincular una dirección IP de origen con un destino específico, rompiendo así el anonimato del usuario. Este tipo de ataques se ha demostrado factible en la práctica, especialmente por adversarios a nivel de proveedor de servicios de Internet, sistemas autónomos, o entidades estatales con capacidad de vigilancia global. Un estudio realizado en 2015 asegura que hasta el 40% del tráfico de Tor puede ser vulnerable a ataques de correlación por parte de atacantes a nivel de red, y que en países con fuerte censura este porcentaje puede ser aún mayor [39].

Los avances recientes en inteligencia artificial y aprendizaje automático han revolucionado significativamente la efectividad de los ataques de correlación de tráfico. Investigadores han desarrollado sistemas como *DeepCorr*, que utiliza arquitecturas de aprendizaje profundo para aprender funciones de correlación específicamente adaptadas a la complejidad de la red Tor [40]. Este sistema supera drásticamente a las técnicas tradicionales: mientras que el sistema *RAPTOR* de última generación logra solo un 4% de precisión en la correlación de flujos, *DeepCorr* alcanza un 96% de precisión utilizando únicamente 900 paquetes de cada flujo objetivo (aproximadamente 900KB de datos) [40].

Ataques de temporización (timing attacks)

Los ataques de temporización son una variante específica del análisis de tráfico, en los que el atacante analiza y correlaciona los tiempos en que los paquetes de datos entran y salen de la red Tor, con el objetivo de vincular el origen y el destino del tráfico [41]. Un ejemplo reciente fue el caso de las autoridades alemanas, que presuntamente utilizaron análisis de temporización y servidores comprometidos para identificar al administrador de un sitio de la *darknet* [42]. Si un atacante puede monitorizar el tráfico en el nodo de entrada y el nodo de salida, y encuentra coincidencias en los patrones temporales, puede inferir la relación entre el usuario y el destino final.

Por ejemplo, si se detecta que un usuario envía un paquete en un momento concreto y, casi al instante, un paquete similar sale de la red por un nodo de salida, se puede intuir que ambos eventos están relacionados y que el usuario es el origen del tráfico hacia ese destino. Este tipo de ataque es especialmente efectivo si el adversario tiene la capacidad de monitorizar grandes partes de la red o controlar varios nodos estratégicos. Además, los

ataques de temporización pueden ser aún más potentes si el atacante introduce marcas temporales o patrones específicos en el tráfico, haciendo que la correlación sea más sencilla y precisa [41].

Una de las técnicas más estudiadas para dificultar los ataques de temporización es el uso de *padding* o relleno de tráfico [43]. Consiste en enviar paquetes de datos falsos o de tamaño uniforme en momentos aleatorios para enmascarar los verdaderos patrones de tráfico. Así, aunque un atacante observe el tráfico, le resulta mucho más difícil distinguir cuándo se produce una comunicación real y cuándo es simplemente ruido añadido por el sistema. Aunque el padding puede aumentar la latencia y el consumo de ancho de banda, es una de las defensas más efectivas contra la correlación temporal [43].

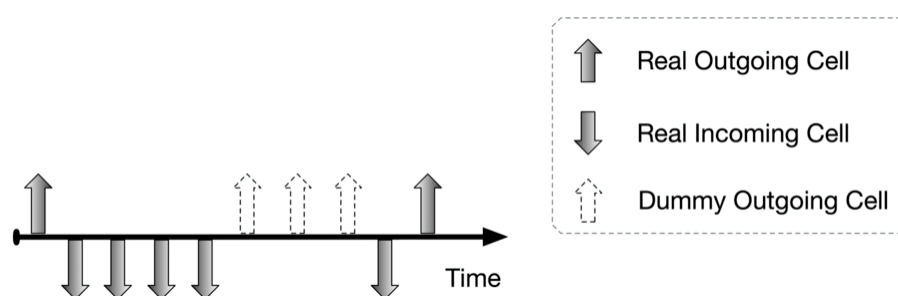


Figura 13. Diagrama del funcionamiento del padding [42]

Además, en el ámbito de la implementación de software, se recomienda que las operaciones criptográficas y de gestión de tráfico se realicen en tiempo constante, es decir, que tarden lo mismo independientemente de los datos procesados, para evitar que los atacantes puedan identificar a usuarios por el tiempo de respuesta de sus paquetes [44].

Sybil attack

Según la documentación oficial de Tor, “El ataque Sybil en la seguridad informática es un ataque en el que se subvierte un sistema de reputación creando un gran número de identidades y utilizándolas para obtener una influencia desproporcionadamente grande en la red” [45].

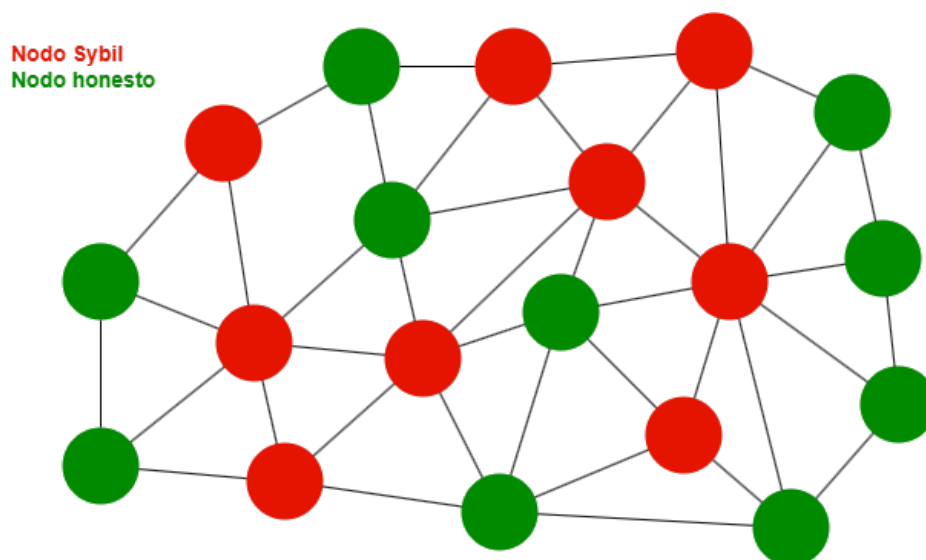


Figura 14. Diagrama de nodos Sybil

Además de poseer un gran número de nodos, el atacante también puede realizar ataques de denegación de servicio (DoS) selectivos para manipular la selección de nodos en la red Tor [46]. Por ejemplo, puede lanzar ataques DoS (el llamado *Sniper Attack* [47]) contra los nodos de entrada legítimos de un usuario o servicio oculto, haciendo que se vuelvan inaccesibles. Cuando el cliente Tor detecta que su *guard node* está caído o no responde tras varios intentos, automáticamente selecciona un nuevo nodo de entrada de entre los disponibles en la red. Si el atacante controla suficientes nodos y repite el ataque de forma persistente, puede forzar al cliente o al servicio oculto a elegir finalmente uno de sus propios nodos maliciosos como *guard* [47]. Esto incrementa significativamente la probabilidad de que el tráfico pase por nodos controlados por el atacante.

Por ejemplo, un actor malicioso identificado como *KAXI7* operó desde al menos 2017 cientos de nodos en la red Tor, llegando a controlar en determinados momentos hasta un 16% de los nodos de entrada, un 35% de los nodos intermedios y un 5% de los nodos de salida, repartidos en múltiples centros de datos y sistemas autónomos de todo el mundo [48].

A diferencia de otros grupos maliciosos que solían centrarse en nodos de salida para manipular o espiar tráfico, como *BITMITM20* [49], *KAXI7* priorizó el control de nodos de entrada e intermedios, lo que significa un interés en la vigilancia y gran posesión de recursos [48].

Aunque no se demostró que *KAXI7* realizara ataques de desanonimización de forma activa, su elevado control sobre la infraestructura de Tor le situó en una posición privilegiada para poder hacerlo, lo que generó gran preocupación en la comunidad. Las autoridades de Tor eliminaron en varias ocasiones cientos de estos *relays*, pero el actor logró restablecer su presencia rápidamente. El caso evidenció la dificultad de detectar y eliminar operadores maliciosos a gran escala dentro de la red Tor y ha impulsado propuestas para que los usuarios puedan elegir y confiar solo en operadores de *relays* verificados como medida de autodefensa [48].

4.2.2. Ataques a nivel de aplicación

Este tipo de ataques explotan debilidades en la configuración de servicios, navegadores o aplicaciones que funcionan sobre la red, permitiendo la filtración de información sensible como la dirección IP real del usuario, su sistema operativo, o su comportamiento de navegación. Estos ataques pueden comprometer el anonimato incluso si la red Tor funciona correctamente, ya que aprovechan errores humanos, vulnerabilidades del software o técnicas avanzadas de rastreo.

Fugas de información por aplicaciones mal configuradas

Uno de los riesgos más frecuentes en el uso de Tor es la filtración de la IP real del usuario o del servidor debido a configuraciones incorrectas o vulnerabilidades en los servicios y aplicaciones. Por ejemplo, muchos servicios ocultos *.onion* han sufrido fugas de información porque los servidores web (como *Apache* o *Nginx*) estaban mal configurados, permitiendo que el módulo de estado o la escucha en interfaces públicas expusieran detalles críticos como la dirección IP, el uso de recursos, la zona horaria o los hosts virtuales activos [50] [51]. Este tipo de errores puede permitir a un atacante identificar la ubicación física del servidor o asociar un servicio oculto con una entidad concreta.

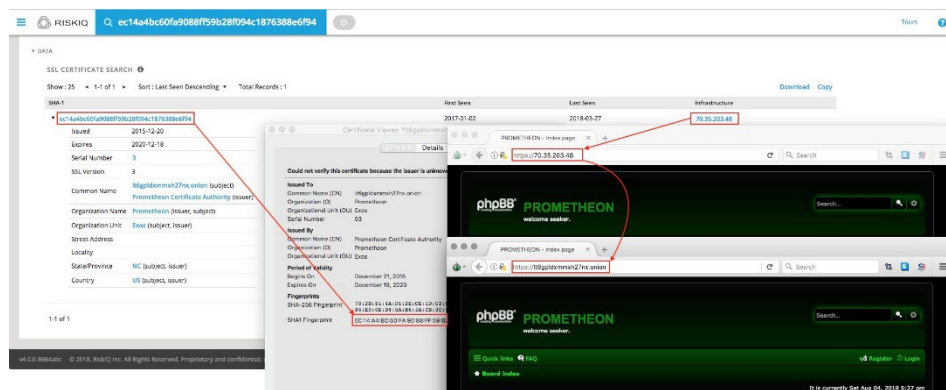


Figura 15. Identificación de la IP de un servicio oculto a través de una mala configuración del certificado [55]

Las fugas de IP también pueden producirse en el lado del cliente. Un caso común es la filtración por *DNS leaks*, donde las solicitudes de resolución de nombres no se envían a través de Tor, sino directamente al proveedor de Internet, exponiendo la IP real del usuario [52]. Además, tecnologías como *WebRTC* pueden permitir conexiones directas entre navegadores, revelando la IP aunque se esté usando Tor [52]. Vulnerabilidades específicas, como la conocida *TorMoil*, han permitido que el navegador Tor filtre la IP real al acceder a enlaces *file://* especialmente diseñados, conectándose directamente al host remoto y saltándose la red Tor [53]. Por este motivo, se recomienda no instalar plugins adicionales en el navegador Tor y utilizar únicamente aplicaciones debidamente configuradas para enrutar todo su tráfico a través de la red.

Ataques por fingerprinting del navegador y técnicas de rastreo

El *fingerprinting* del navegador es una técnica que permite identificar de manera única a un usuario mediante la recopilación de características específicas de su navegador y sistema operativo, como la resolución de pantalla, fuentes instaladas, plugins, configuración de idioma y otros parámetros accesibles mediante *JavaScript* o *APIs* del navegador [54]. Aunque Tor Browser está diseñado para minimizar la unicidad del *fingerprint*, haciendo que todos los usuarios parezcan iguales (Figura 16), existen vectores que pueden ser explotados si el usuario modifica la configuración por defecto (hoy en día *JavaScript* está deshabilitado por defecto en el navegador de Tor) o habilita scripts avanzados.

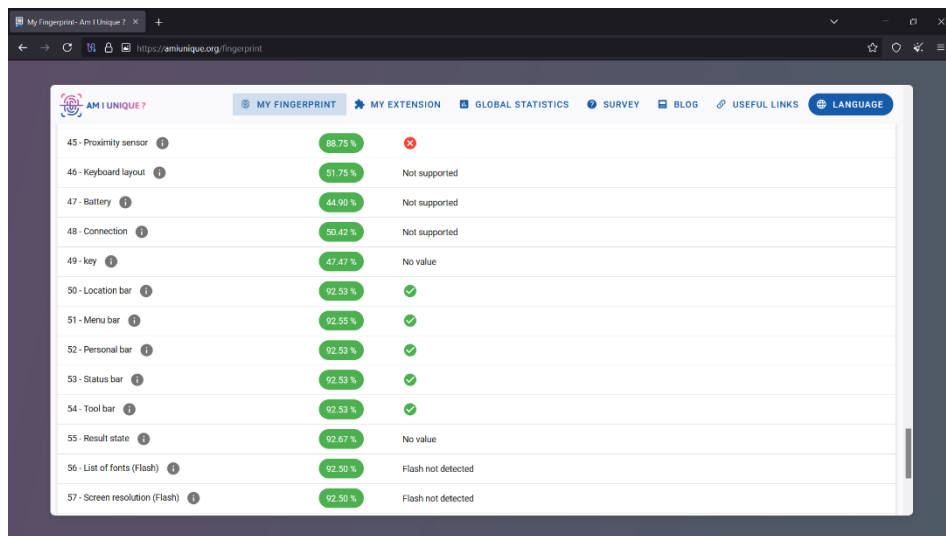


Figura 16. Uso desde Tor Browser de herramienta para comprobar la unicidad de datos de *fingerprinting* [57]

También se puede tratar de identificar a usuarios de Tor mediante el uso de archivos infectados. Este tipo de ataque consiste en ofrecer un archivo aparentemente inofensivo (como un documento PDF, *Word*, imagen o ejecutable) que en realidad contiene algún tipo de mecanismo que, al ser abierto por la víctima, provoca que su dispositivo realice una conexión directa fuera de la red Tor, revelando así su dirección IP real u otros datos de interés [55].

Por obvio y trivial que pueda parecer, este método ha sido utilizado en investigaciones policiales y casos emblemáticos, como el caso *PlayPen*, donde el FBI distribuyó archivos cebo a través de un foro de explotación infantil en la *dark web* para identificar y arrestar a los usuarios que descargaban y los abrían [55] [56]. El funcionamiento es sencillo: el archivo cebo suele contener código incrustado (por ejemplo, una macro, un script, o una imagen remota) que, al ejecutarse o visualizarse, fuerza al sistema operativo o a la aplicación a conectarse a un servidor controlado por el atacante o las autoridades. Esta conexión se realiza fuera del circuito Tor, exponiendo la IP real del usuario y permitiendo su localización y posterior identificación. Además, al tan solo conectarse a un servidor, una herramienta antivirus no marcará el archivo como malicioso, por lo que no levantará sospechas, aunque lo pasemos por *VirusTotal* o semejantes.

La efectividad principal de este tipo de ataques radica en que muchos usuarios confían en el anonimato que les proporciona Tor y abren archivos descargados sin tomar precauciones adicionales. Por ello, una de las principales recomendaciones para protegerse frente a esta técnica es evitar abrir archivos descargados a través de Tor en el sistema principal, y optar por entornos virtualizados o sistemas especialmente diseñados para evitar fugas de tráfico fuera de la red Tor, como, de nuevo, *Whonix*.

4.2.3. Ataques de *cross-device tracking*: *uBeacons*

Una baliza de ultrasonidos (*uBeacon*, en inglés), es un dispositivo o software que emite señales ultrasónicas para transmitir identificadores únicos a dispositivos cercanos, como *smartphones* o *tablets*, a través de sus micrófonos [57]. Estas señales, imperceptibles para las personas, operan como un canal encubierto de comunicación entre dispositivos, permitiendo a un atacante vincular actividades anónimas en Tor con la identidad real de los usuarios mediante técnicas de rastreo cruzado (*cross-device tracking*).

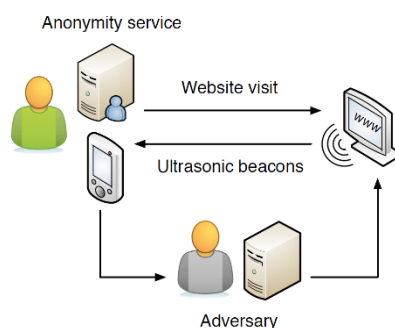


Figura 17. Diagrama de desanonimización utilizando balizas de ultrasonidos [63]

Cuando un usuario accede a un sitio web, servicio oculto, archivo de audio o de vídeo que contiene una *uBeacon* integrada, el navegador —incluso a través de Tor— reproduce la señal ultrasónica. Cualquier dispositivo cercano con un micrófono activo (como un *smartphone* o un asistente como *Alexa*, con una aplicación maliciosa o permisos excesivos, autorizados por las autoridades pertinentes) puede captar este identificador único y enviarlo a servidores remotos,

junto con datos como la ubicación GPS, la dirección IP real o información del perfil del usuario [58].

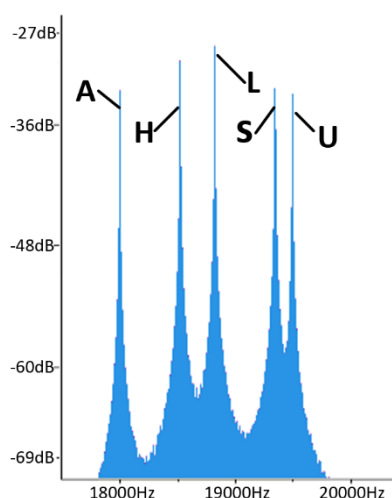


Figura 18. Espectrograma de una baliza codificando 5 caracteres usando MSFK [57]

Experimento práctico de desanonimización con uBeacons

En el siguiente apartado, se realizará un experimento práctico para demostrar cómo una baliza de ultrasonidos puede utilizarse para desanonimizar a un usuario de Tor, vinculando su actividad anónima en la red con su identidad real a través de su teléfono inteligente. Este ejercicio, que se hará en un entorno controlado, se presentará como *Proof of Concept (PoC)* del método mencionado, y está basado en una idea de Vasilios Mavroudis [59].

El experimento se centrará en dos componentes clave:

- 1) Emisión de una señal ultrasónica desde un archivo de audio incrustado en un vídeo de una página web.
- 2) Captura y correlación de la señal por parte de un dispositivo móvil cercano, que reportará datos identificativos a un servidor controlado.

Lo primero que haremos será desarrollar la página web en HTML y JavaScript para reproducir la baliza. Para ello, tomamos como referencia un código de JS creado por *City Frequencies* [60] [61], mediante el que crearé mi propia página personalizada. A continuación, se muestra el bloque del código que he desarrollado para emitir la baliza con un mensaje binario codificado con

BPSK [62] en una frecuencia de 19500Hz para los bits 1 y de 18500Hz para los bits 0.

```
const DATA = [1,0,1,1,0,1,0,0,1,1,0,0,1,1,0,0];

// Emitir baliza por altavoz
document.getElementById("startBeacon").onclick = function() {
    const context = new AudioContext();
    const oscillator = context.createOscillator();
    const gain = context.createGain();

    let time = context.currentTime;

    oscillator.connect(gain);
    gain.connect(context.destination);
    gain.gain.value = 0.8;

    DATA.forEach(bit => {
        oscillator.frequency.setValueAtTime(
            bit ? FREQ_1 : FREQ_0,
            time
        );
        time += BIT_DURATION;
    });

    oscillator.start();
    oscillator.stop(time);

    oscillator.onended = () => {
        context.close();
    };
};
```

Además, creamos otra función para descargar en un archivo *WAV* (que no aplica compresión) la baliza con el identificador único codificado (*UUID*). En este caso, el *UUID* es de 16 bits, para mostrar el funcionamiento, pero en un entorno real se usaría una longitud mayor, de más de 256 bits, para asegurar al cien por cien que no se detecta por error.

Ahora que podemos descargar el archivo de audio, lo guardamos como *baliza.wav* y analizamos su espectrograma mediante el programa *Audacity*. Como se muestra en la Figura 19, si ampliamos hasta ver las frecuencias de entre 18000Hz y 20000Hz, podemos distinguir los bits del *UUID* en los saltos de frecuencia del archivo.

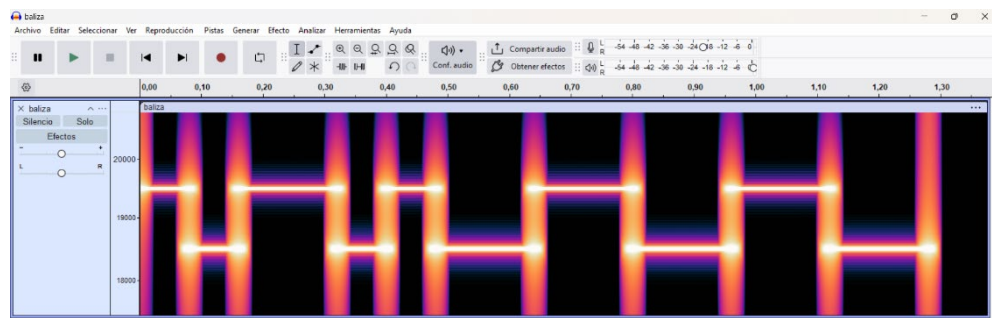


Figura 19. Espectrograma de la *baliza.wav*

Finalmente, incrustamos el archivo en un vídeo cualquiera mediante el comando mostrado en la Figura 20, utilizando la herramienta *ffmpeg*.

```
PS C:\Users\j4ime\Downloads> ffmpeg -i video.mp4 -i baliza.wa
v -filter_complex "[1]adelay=7000|7000[baliza];[0:a][baliza]a
mix=inputs=2:duration=first:dropout_transition=2" -vf scale=-
2:360 -c:v libx264 -preset veryslow -crf 28 -c:a aac -b:a 256
k video_infectado.mp4
```

Figura 20. Comando de *ffmpeg* para incrustar la baliza en un vídeo

Ahora que ya tenemos el vídeo infectado, lo adjunto en una página web que, con el fin de aportar más realismo al experimento, alojaremos en un servicio oculto de Tor. Para ello, deberemos instalar la herramienta Tor (no el *Tor Browser*), iniciar el servidor en el puerto 8080 desde *Python*, y configurar el archivo *torrc* para añadir la dirección de dicho servidor (Figura 21).

Figura 21. Configuración del archivo *torrc*

Ahora, tras lanzar el servidor usando el comando *tor.exe -f torrc* podremos acceder a la página web mediante el enlace que encontraremos en

C:\Tor\tor\servicio_oculto, que en este caso es *ihykjqmea2w4rrcexfxjvthitg2z4fk4kyveoxrlwsyrhnnbxbjxvkad.onion* (Figura 22).

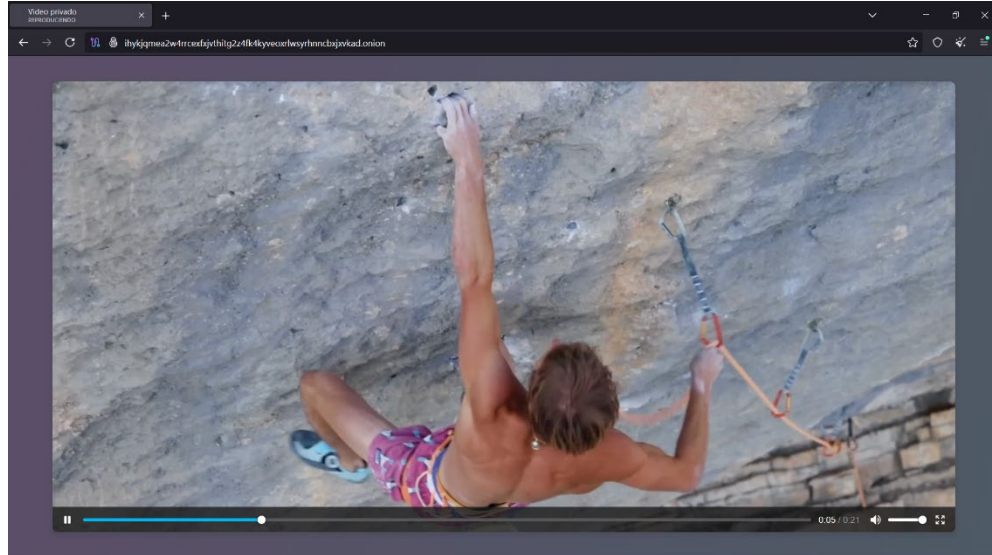


Figura 22. Video infectado alojado en servicio oculto .onion

Ahora que ya tenemos la trampa en funcionamiento, debemos realizar la otra parte del experimento: queremos que un dispositivo inteligente capture la baliza que suena en el vídeo y mande información a un servidor.

Aunque en un caso real deberíamos implementar esta funcionalidad en un segundo plano en los teléfonos de la población, para la demostración he desarrollado una aplicación en *AndroidStudio* (escrita en *Kotlin*), que he llamado *uBeaconSnitch*, que escuchará activamente mediante el micrófono, y en cuanto detecte los bits de la baliza mandará su IP y geolocalización a un servidor alojado en una página web propia. Todo el código, tanto el mencionado hasta ahora, como el de la aplicación y el que se mencione en el resto del trabajo, estará adjunto en los directorios de la entrega.

Para realizar la decodificación del mensaje oculto en el audio, implementamos una función con el algoritmo de Goertzel [63] para detectar la presencia de las frecuencias portadoras específicas asociadas a cada bit de la baliza. Este algoritmo es especialmente eficiente para este tipo de tareas, ya que permite calcular la energía de una frecuencia concreta dentro de una ventana de muestras, sin necesidad de realizar una transformada de Fourier completa [64].

Así, para cada ventana de audio capturada por el micrófono, la aplicación analiza si la energía en las frecuencias correspondientes a los bits 0 y 1 supera un umbral, utilizando una función de decodificación BPSK, y de este modo reconstruye la secuencia binaria emitida por el vídeo.

```
class BPSKDecoder(  
    private val freq0: Double,  
    private val freq1: Double,  
    private val sampleRate: Int,  
    private val samplesPerBit: Int,  
    private val numBits: Int  
) {  
    fun decode(buffer: ShortArray): String {  
        val bits = StringBuilder()  
        for (i in 0 until numBits) {  
            val start = i * samplesPerBit  
            val end = minOf(start + samplesPerBit, buffer.size)  
            if (end - start <= 0) break  
            val chunk = buffer.sliceArray(start until end)  
            val mag0 = goertzel(chunk, freq0)  
            val mag1 = goertzel(chunk, freq1)  
            bits.append(if (mag1 > mag0) '1' else '0')  
        }  
        return bits.toString()  
    }  
  
    private fun goertzel(buffer: ShortArray, freq: Double): Double {  
        val n = buffer.size  
        val k = (0.5 + (n * freq) / sampleRate).toInt()  
        val w = 2.0 * Math.PI * k / n  
        val cosine = Math.cos(w)  
        val coeff = 2.0 * cosine  
        var q0 = 0.0  
        var q1 = 0.0  
        var q2 = 0.0  
        for (sample in buffer) {  
            q0 = coeff * q1 - q2 + sample  
            q2 = q1  
            q1 = q0  
        }  
        return q1 * q1 + q2 * q2 - q1 * q2 * coeff  
    }  
}
```

El proceso de detección se realiza en tiempo real: la app acumula muestras de audio y, mediante una ventana deslizante, va decodificando bloques de 32 bits

para buscar la secuencia característica de la baliza. Cuando la secuencia es detectada, la aplicación obtiene la dirección IP local del dispositivo y solicita la localización GPS actual. Ambos datos, junto con la hora exacta de la detección, se empaquetan en un mensaje JSON y se envían automáticamente a la API del servidor web, que los muestra por pantalla.



Figura 23. Baliza detectada en la aplicación

Para el servidor de recepción de los datos, configuramos una aplicación de Flask que escucha constantemente los datos que se envían a la API. Para hacer la API accesible públicamente (ya que el dispositivo móvil del objetivo, en un caso real, no se encontraría en el mismo lugar que el servidor), realizamos la compra de un dominio llamado *ubeaconsnitch.click*, en este caso en la página *Namecheap.com* (Figura 24), y utilizamos un túnel de *CloudFlare* para redirigir el dominio a la IP de nuestra máquina local.

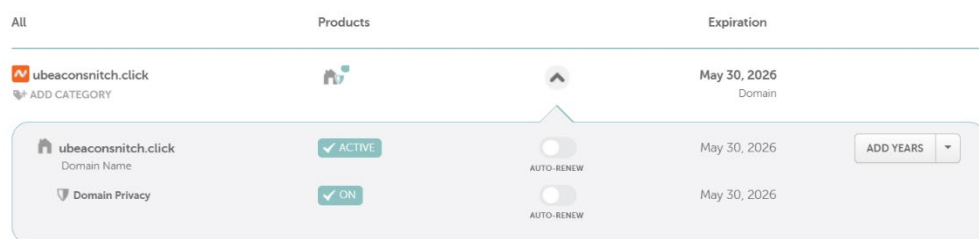
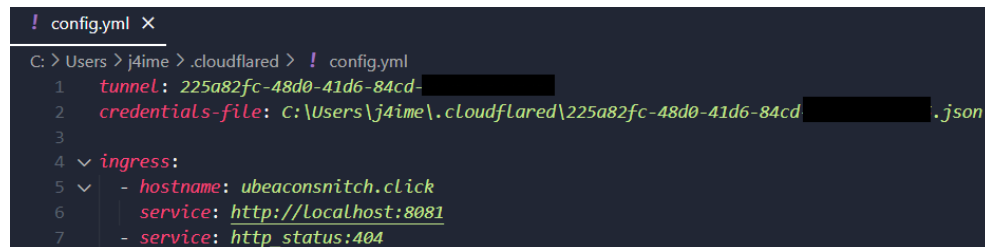


Figura 24. Compra del dominio *ubeaconsnitch.click* en *Namecheap.com*

Para configurar el túnel, primero debemos modificar los *nameservers* del dominio para que apunten a los de *CloudFlare*. Tras ello, instalamos el

servicio *cloudflared* desde su página oficial, y ejecutamos en la consola de comandos *cloudflared tunnel login* para autenticarnos.

Después, creamos el túnel mediante el comando *cloudflared tunnel create tunel-ubeacon*, que nos devolverá su *UUID*. Ahora, cambiamos al directorio de instalación de *cloudflared* y creamos un archivo llamado *config.yml*, en el que agregamos la configuración del túnel (Figura 25).



```
! config.yml X
C: > Users > j4ime > .cloudflared > ! config.yml
1 tunnel: 225a82fc-48d0-41d6-84cd-[redacted]
2 credentials-file: C:\Users\j4ime\.cloudflared\225a82fc-48d0-41d6-84cd-[redacted].json
3
4 ingress:
5   - hostname: ubeaconsnitch.click
6     service: http://localhost:8081
7     service: http_status:404
```

Figura 25. Configuración del archivo *config.yml* para el túnel de *CloudFlare*

Finalmente, asociamos el túnel al dominio con el comando *cloudflared tunnel route dns tunel-ubeacon ubeaconsnitch.click*, y ya podremos lanzar el túnel con *cloudflared tunnel run tunel-ubeacon*, que guardaremos en un archivo BAT por comodidad.

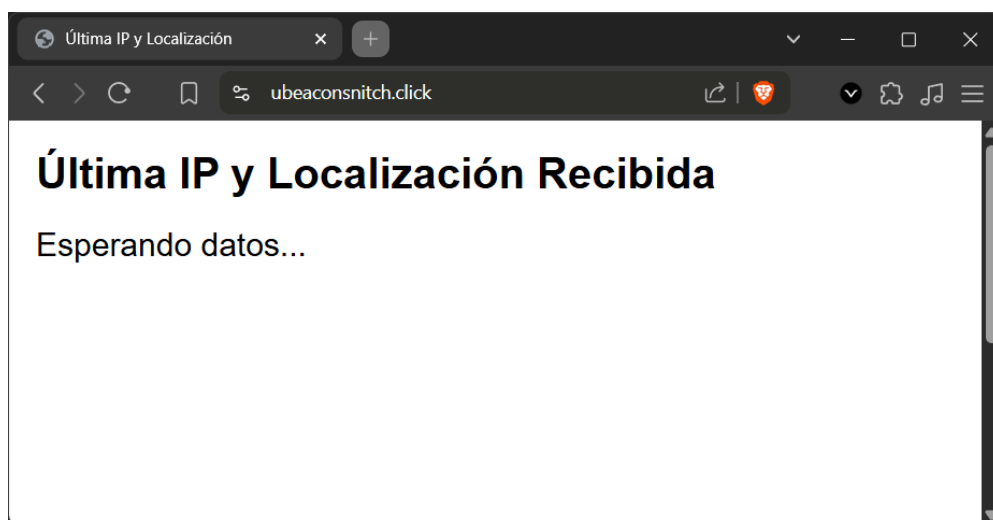


Figura 26. Servidor web receptor corriendo en el dominio *ubeaconsnitch.click*

Ya tenemos todo lo necesario para el funcionamiento del experimento. Ahora, cuando la aplicación del móvil detecte la baliza, se mostrará en la web tanto la IP como la geolocalización del dispositivo (Figura 27).

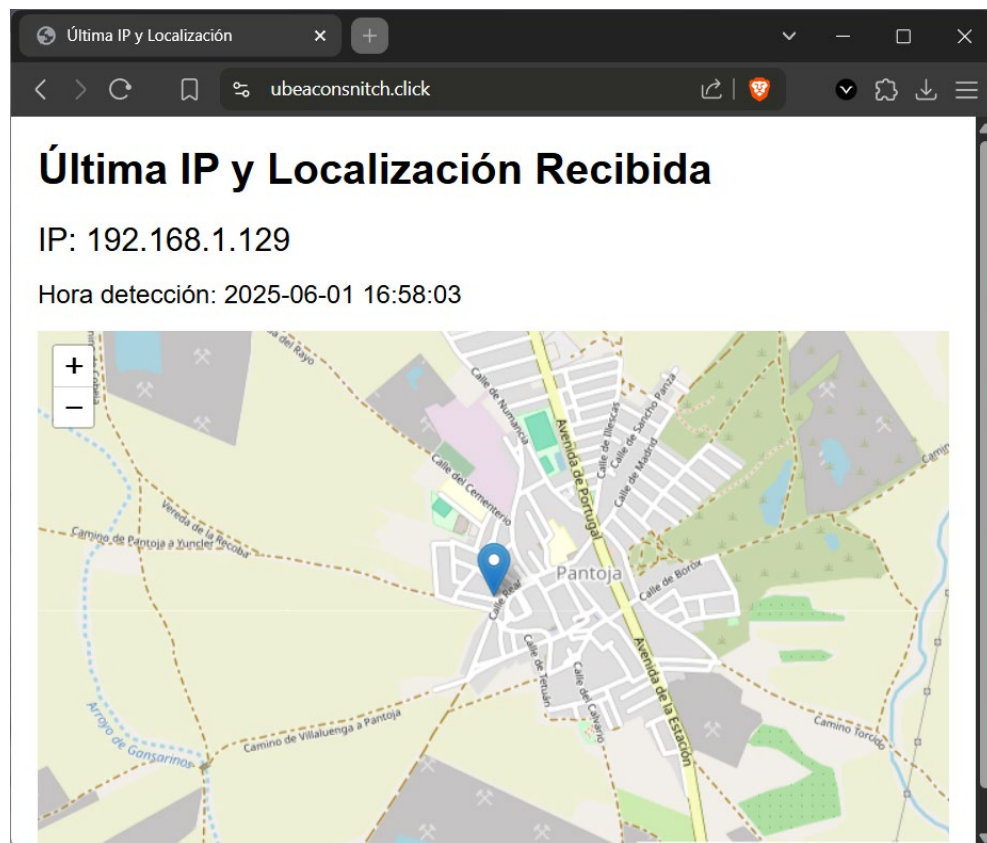


Figura 27. Resultado final del experimento con la IP y geolocalización de la víctima

En cuanto a las técnicas de defensa frente a este tipo de ataques, lo ideal sería desconectar físicamente los altavoces cuando no se estén usando, y utilizar alguna herramienta para filtrar la emisión de balizas en ciertas frecuencias, como *Silverdog* [65] [66]. Además, es muy importante revisar las aplicaciones que tenemos instaladas y los permisos que les damos, aunque en un ataque sofisticado, podría utilizarse una aplicación confiable como micrófono.

5. Conclusiones

Este trabajo ha proporcionado un análisis integral de la red Tor, examinando tanto sus fortalezas técnicas como las vulnerabilidades que enfrenta en el panorama actual de amenazas digitales. Su arquitectura fundamental, basada en el enrutamiento cebolla, continúa siendo efectiva para proteger la privacidad de las comunicaciones después de más de dos décadas de evolución. Esta robustez se extiende tanto al modo tradicional de navegación anónima como a los servicios ocultos, que permiten que los servidores mantengan también su anonimato, creando un ecosistema bidireccional de privacidad que ha demostrado ser fundamental para activistas, periodistas y usuarios que requieren protección frente a la vigilancia.

El ecosistema de herramientas desarrollado alrededor de Tor demuestra la madurez y versatilidad de la plataforma. Desde crawlers especializados para la indexación de contenido oculto hasta herramientas de administración y análisis, la red ha evolucionado hacia una infraestructura completa que soporta una amplia gama de aplicaciones legítimas.

La implementación de criptografía de curva elíptica, particularmente a través del protocolo NTOR y *Curve25519*, representa una evolución crucial en la seguridad de Tor. Esta tecnología no solo proporciona niveles de seguridad equivalentes a sistemas tradicionales con claves significativamente más cortas, sino que también mejora sustancialmente el rendimiento de la red. El protocolo ECDH permite el establecimiento eficiente de claves simétricas únicas para cada salto del circuito, garantizando que la compromisión de un nodo no afecte la seguridad de toda la comunicación. Sin embargo, la amenaza emergente de la computación cuántica requiere una transición planificada hacia algoritmos postcuánticos para mantener la seguridad a largo plazo.

Las técnicas de desanonimización estudiadas revelan la naturaleza multifacética de las amenazas que enfrenta Tor. Los ataques a nivel de red, como la correlación de tráfico y los ataques Sybil, demuestran que adversarios con recursos suficientes pueden comprometer el anonimato explotando la naturaleza distribuida de la red. Los ataques a nivel de aplicación, incluyendo fugas de información por configuraciones incorrectas y técnicas de fingerprinting, muestran cómo las vulnerabilidades externas al protocolo pueden comprometer la seguridad. Por otra parte, el experimento práctico con *uBeacons* desarrollado en el trabajo demuestra cómo las amenazas modernas pueden explotar canales físicos para vincular actividades anónimas con identidades reales. La capacidad de incrustar balizas ultrasónicas en archivos de video y correlacionar su detección por

dispositivos cercanos con actividad en servicios ocultos demuestra que la protección de la privacidad requiere un enfoque que considere no solo la seguridad del protocolo, sino también el entorno físico y digital completo del usuario: “*OPSec as a lifestyle*”.

Mirando hacia el futuro, Tor enfrenta desafíos significativos que requieren adaptación continua. La transición hacia criptografía postcuántica es inevitable pero compleja, requiriendo equilibrar seguridad, rendimiento y compatibilidad. Las potenciales amenazas emergentes que puedan surgir, especialmente con los avances de la inteligencia artificial, necesitarán contramedidas específicas para resistir estos nuevos ataques, cada vez más potentes.

En definitiva, la red Tor continúa siendo una herramienta esencial para la protección de la privacidad digital, pero su efectividad futura dependerá de la capacidad de adaptarse a un panorama de amenazas en constante evolución. La batalla por el anonimato en línea es un proceso dinámico que requiere innovación técnica continua, educación de sus usuarios y un compromiso sostenido con los principios fundamentales de libertad y privacidad que definen una sociedad digital libre. Al mismo tiempo, es fundamental desarrollar metodologías equilibradas que permitan combatir el cibercrimen sin comprometer los derechos de privacidad de los usuarios legítimos de la red.

Referencias

- [1] Tor Project, «Historia,» Tor Project, [En línea]. Available: <https://www.torproject.org/es/about/history/>.
- [2] Tor Project, «Types of relays on the Tor network,» Tor Project, [En línea]. Available: <https://community.torproject.org/relay/types-of-relays/>.
- [3] J. Cea Avi3n, «Tor, nodos "Bridge" y ofuscaci3n,» El hilo del laberinto, 8 Marzo 2016. [En l3nea]. Available: https://blog.jcea.es/posts/20160308-tor_bridge.html.
- [4] «Implementation of Diffie-Hellman Algorithm,» GeeksForGeeks, 3 Julio 2024. [En l3nea]. Available: <https://www.geeksforgeeks.org/implementation-diffie-hellman-algorithm/>.
- [5] Tor Project, «Set up Your Onion Service,» Tor Project, [En l3nea]. Available: <https://community.torproject.org/onion-services/setup/>.
- [6] J. San Jos3, «TorBot – Herramienta de OSINT para la Dark Web,» Derecho de la Red, 5 Marzo 2022. [En l3nea]. Available: <https://derechodelared.com/2022/03/05/torbot-herramienta-osint-dark-web/>.
- [7] D. De Pascale, G. Cascavilla, D. A. Tamburri y W.-J. Van Den Heuvel, «CRATOR: a Dark Web Crawler (arXiv),» 10 Mayo 2024. [En l3nea]. Available: <https://arxiv.org/html/2405.06356v1>.
- [8] RomanAcademy, «TorCrawl.py Overview: Anonymous Web Scraping Using Tor,» Medium, 10 Septiembre 2024. [En l3nea]. Available: <https://roman-academy.medium.com/torcrawl-py-overview-anonymous-web-scraping-using-tor-fbb94d131e73>.
- [9] A. Santos, «ACHE Crawler Documentation,» ACHE, [En l3nea]. Available: <https://ache.readthedocs.io/en/latest/>.
- [10] SOCRadar, «Top 10 Dark Web Search Engines in 2025,» SOCRadar, 26 Marzo 2025. [En l3nea]. Available: <https://socradar.io/top-10-dark-web-search-engines-in-2025/>.
- [11] ajvb, «Awesome-Tor,» GitHub, 3 Septiembre 2023. [En l3nea]. Available: <https://github.com/ajvb/awesome-tor>.
- [12] torps, «Tor Path Simulator,» GitHub, 16 Enero 2017. [En l3nea]. Available: <https://github.com/torps/torps>.
- [13] OONI Probe, «About,» OONI Probe, [En l3nea]. Available: <https://ooni.org/about/>.
- [14] J. M. Fern3ndez, «DNSCrypt, o como cifrar tus peticiones DNS,» Security Artwork, 25 Febrero 2016. [En l3nea]. Available:

- <https://www.securityartwork.es/2016/02/25/dnscrypt-o-como-cifrar-tus-peticiones-dns/>.
- [15] CrowdStrike, «Tortilla,» GitHub, 18 Julio 2016. [En línea]. Available: <https://github.com/CrowdStrike/Tortilla>.
- [16] hackplayers, «Enruta todo el tráfico de tu máquina virtual por Tor con Tortilla,» hackplayers, 7 Agosto 2013. [En línea]. Available: <https://www.hackplayers.com/2013/08/tortilla-enruta-todo-el-trafico-de-tu-vm-por-tor.html>.
- [17] Eycorsican, «Leaf,» GitHub, 2 Mayo 2025. [En línea]. Available: <https://github.com/eycorsican/leaf>.
- [18] shadow, «ShadowTor,» GitHub, 19 Mayo 2025. [En línea]. Available: <https://shadow.github.io/docs/guide/shadow.html>.
- [19] J. M. Miret, J. Valera y M. Valls, «Criptografía con curvas elípticas,» Crypt4you, 12 Mayo 2015. [En línea]. Available: <https://criptored.es/crypt4you/temas/ECC/leccion1/leccion1.html>.
- [20] N. Sullivan, «A (Relatively Easy To Understand) Primer on Elliptic Curve Cryptography,» Cloudflare, 24 Octubre 2013. [En línea]. Available: <https://blog.cloudflare.com/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/>.
- [21] S. A. Raposo Briceño, «Criptografía de curvas elípticas. Fundamentos matemáticos e implementación,» Julio 2019. [En línea]. Available: <https://oa.upm.es/56215/>.
- [22] «Grupo Abeliano,» Estructuras Algebraicas, 20 Mayo 2013. [En línea]. Available: <https://estructurasalgebraicas.es/tl/Grupo-abeliano.htm>.
- [23] VPN Unlimited, «Elliptic-Curve Discrete Logarithm Problem,» VPN Unlimited, [En línea]. Available: <https://www.vpnunlimited.com/help/cybersecurity/elliptic-curve-discrete-log>.
- [24] J. Wyss-Gallifent, «Pollard's Rho Method,» 27 Febrero 2023. [En línea]. Available: <https://math.umd.edu/~immortal/MATH406/lecturenotes/pollardrho.pdf>.
- [25] J. Szurdi, «Tor 101: How Tor Works and its Risks to the Enterprise,» Unit 42, 29 Agosto 2022. [En línea]. Available: <https://unit42.paloaltonetworks.com/tor-traffic-enterprise-networks/>.
- [26] G. Walker, «Elliptic Curve. Mathematics for cryptographic systems,» learn me a bitcoin, 18 Diciembre 2024. [En línea]. Available: <https://learnmeabitcoin.com/technical/cryptography/elliptic-curve/>.
- [27] B. Buchanan, «If We Were To Create the Internet again... it would be Tor-based,» Medium, 28 Abril 2019. [En línea]. Available: <https://medium.com/asecuritysite->

when-bob-met-alice/if-we-were-to-create-the-internet-again-it-would-be-tor-based-d6a7eadd51fb.

- [28] H. D. Chaparro Zuñiga, «TOR, anonimato en internet,» 2015. [En línea]. Available: http://bibliotecadigital.econ.uba.ar/download/tpos/1502-0885_ChaparroZunigaHD.pdf.
- [29] «Curve25519,» Wikipedia, 10 Mayo 2025. [En línea]. Available: <https://en.wikipedia.org/wiki/Curve25519>.
- [30] D. J. Bernstein, «Curve25519: new Diffie-Hellman speed records,» 9 Febrero 2006. [En línea]. Available: <https://cr.yp.to/ecdh/curve25519-20060209.pdf>.
- [31] Anishbhowmick, «Elliptic Curve Back Door,» Medium, 19 Abril 2024. [En línea]. Available: <https://anishbhowmick833.medium.com/elliptic-curve-back-door-c729a68b12e0>.
- [32] D. Berger, M. Lemoudden y W. J. Buchanan, «Post Quantum Migration of Tor,» 13 Marzo 2025. [En línea]. Available: <https://eprint.iacr.org/2025/479.pdf>.
- [33] «Key encapsulation mechanism,» Wikipedia, 22 Mayo 2025. [En línea]. Available: https://en.wikipedia.org/wiki/Key_encapsulation_mechanism.
- [34] J. Hoffstein, J. Pipher y J. H. Silverman, «NTRU: A Ring-Based Public Key Cryptosystem,» 21 Junio 1998. [En línea]. Available: <https://www.ntru.org/f/hps98.pdf>.
- [35] U. Pathum, «CRYSTALS Kyber : The Key to Post-Quantum Encryption,» Medium, 5 Enero 2024. [En línea]. Available: <https://medium.com/identity-beyond-borders/crystals-kyber-the-key-to-post-quantum-encryption-3154b305e7bd>.
- [36] «Kyber,» CRYSTALS, 23 Diciembre 2020. [En línea]. Available: <https://pq-crystals.org/kyber/>.
- [37] R. Gonzalez, «Kyber - How does it work?,» Approachable Cryptography, 14 Septiembre 2021. [En línea]. Available: <https://cryptopedia.dev/posts/kyber/>.
- [38] E. Jardine, A. M. Lindner y G. Owenson, «The potential harms of the Tor anonymity network cluster disproportionately in free countries,» 30 Noviembre 2020. [En línea]. Available: <https://pmc.ncbi.nlm.nih.gov/articles/PMC7749358/>.
- [39] R. Nithyanand, O. Starov, A. Zair, P. Gill y M. Schapira, «Measuring and mitigating AS-level adversaries against Tor,» 26 Diciembre 2015. [En línea]. Available: <https://arxiv.org/pdf/1505.05173>.
- [40] M. Nasr, A. Bahramal y A. Houmansadr, «DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning,» 22 Agosto 2018. [En línea]. Available: <https://arxiv.org/abs/1808.07285>.

- [41] I. Karunanayake, N. Ahmed, R. Malaney, R. Islam y S. K. Jha, «De-Anonymisation Attacks on Tor: A Survey,» 2021. [En línea]. Available: <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9471821>.
- [42] Y. Yun, «Is Tor still safe after Germany's 'timing attack?' Answer: It's complicated...,» CoinTelegraph, 18 Octubre 2024. [En línea]. Available: <https://cointelegraph.com/news/tor-germany-timing-attack-privacy>.
- [43] B. Huang and Y. Du, "Break-Pad: effective padding machines for tor with break burst padding," 1 Octubre 2024. [Online]. Available: <https://cybersecurity.springeropen.com/articles/10.1186/s42400-024-00222-y>.
- [44] J. Jančár, «The need for constant-time cryptography,» Red Hat Research, Noviembre 2021. [En línea]. Available: <https://research.redhat.com/blog/article/the-need-for-constant-time-cryptography/>.
- [45] Tor Project, «Ataque Sybil,» Tor Project, [En línea]. Available: <https://support.torproject.org/es/glossary/sybil-attack/>.
- [46] D. N. Krawetz, «Behind the Tor Attacks,» The Hacker Factor Blog, 12 Octubre 2017. [En línea]. Available: <https://www.hackerfactor.com/blog/index.php?%2Farchives%2F779-Behind-the-Tor-Attacks.html>.
- [47] R. Jansen, F. Tschorsch, A. Johnson y B. Scheuermann, «The Sniper Attack: Anonymously De-anonymizing and Disabling the Tor Network,» 22 Febrero 2014. [En línea]. Available: https://www.ndss-symposium.org/wp-content/uploads/2017/09/05_4_0.pdf.
- [48] nusenu, «Is "KAX17" performing de-anonymization Attacks against Tor Users?,» Medium, 29 Noviembre 2021. [En línea]. Available: <https://nusenu.medium.com/is-kax17-performing-de-anonymization-attacks-against-tor-users-42e566defce8>.
- [49] nusenu, «How Malicious Tor Relays are Exploiting Users in 2020 (Part I),» Medium, 9 Agosto 2020. [En línea]. Available: <https://nusenu.medium.com/how-malicious-tor-relays-are-exploiting-users-in-2020-part-i-1097575c0cac>.
- [50] P. Paganini, «Many misconfigured Tor sites expose the public IP address via SSL certificates,» Security Affairs, 5 Septiembre 2018. [En línea]. Available: <https://securityaffairs.com/75904/security/tor-sites-ip-leakage.html>.
- [51] O. Sultan, «TOR Traffic Data leak Caused by Misconfigured Apache Servers,» HACKREAD, 1 Febrero 2016. [En línea]. Available: <https://hackread.com/tor-traffic-data-leak-caused-by-misconfigured-apache-servers/>.

- [52] Cyberly, «Is there a risk of IP leakage when using Tor?,» Cyberly, [En línea]. Available: <https://www.cyberly.org/en/is-there-a-risk-of-ip-leakage-when-using-tor/index.html>.
- [53] C. Cimpanu, «TorMoil Vulnerability Leaks Real IP Address from Tor Browser Users,» Bleeping Computer, 3 Noviembre 2017. [En línea]. Available: <https://www.bleepingcomputer.com/news/security/tormoil-vulnerability-leaks-real-ip-address-from-tor-browser-users/>.
- [54] J. C. Norte, «Advanced Tor Browser Fingerprinting,» JCarlosNorte, 6 Marzo 2016. [En línea]. Available: <http://jcarlosnorte.com/security/2016/03/06/advanced-tor-browser-fingerprinting.html>.
- [55] CyberYozh, «Deanonymization of Tor users through bait files,» CyberYozh, [En línea]. Available: <https://book.cyberyozh.com/deanonymization-tor-users-through-bait-files/>.
- [56] «Playpen (website),» Wikipedia, 29 Abril 2025. [En línea]. Available: [https://en.wikipedia.org/wiki/Playpen_\(website\)](https://en.wikipedia.org/wiki/Playpen_(website)).
- [57] The Valley, «Qué son los Beacons y cuál es su potencial,» The Valley, [En línea]. Available: <https://thevalley.es/blog/que-son-los-beacons-y-cual-es-su-potencial/>.
- [58] V. Mavroudis, «Information Leakage Attacks and Countermeasures,» 2021. [En línea]. Available: <https://discovery.ucl.ac.uk/id/eprint/10141662/1/thesis.pdf>.
- [59] V. Mavroudis, «Tor Deanonymization Attack using Ultrasound Beacons,» YouTube, 13 Enero 2017. [En línea]. Available: https://www.youtube.com/watch?v=GRPYF9b7_tA.
- [60] City Frequencies, «NUHF audio beacon emulator,» City Frequencies, 2020. [En línea]. Available: <https://cityfreqs.com.au/nuhf-synth.html>.
- [61] City Frequencies, «synth.js,» 2020. [En línea]. Available: <https://cityfreqs.com.au/js/synth.js>.
- [62] Sistemas de Comunicación Wiki, «Sesión 28 - Modulación BPSK,» Sistemas de Comunicación Wiki, [En línea]. Available: https://sistemas-de-comunicacion.fandom.com/es/wiki/Sesi%C3%B3n_28_-_Modulaci%C3%B3n_BPSK..
- [63] OpenCourseWare, «The Goertzel Algorithm and the Chirp Transform,» 2006. [En línea]. Available: https://ocw.mit.edu/courses/6-341-discrete-time-signal-processing-fall-2005/c5b65e06d8d1221fa68b23b7f0d23b41_lec20.pdf.
- [64] «El fantástico filtro Goertzel,» Experimentos y Notas de LU7HZ, 10 Febrero 2015. [En línea]. Available: <https://lu7hz.blogspot.com/2015/02/el-fantastico-filtro-goertzel.html>.

- [65] V. Mavroudis, «Silverdog,» GitHub, 2017. [En línea]. Available: <https://github.com/ubeacsec/Silverdog>.
- [66] V. Mavroudis, S. Hao, Y. Fratantonio, F. Maggi, C. Kruegel y G. Vigna, «On the privacy and security of the ultrasound ecosystem,» *Proceedings on Privacy Enhancing Technologies*, vol. 2017, n° 2, pp. 95--112, 2017.
- [67] Tor Project, «The rendezvous protocol,» Tor Project, [En línea]. Available: <https://spec.torproject.org/rend-spec/rendezvous-protocol.html>.
- [68] D. Cooper, «Unhidden Services,» Heise Magazine, 2017. [En línea]. Available: <https://www.heise.de/select/ct/2017/22/1508778711558534>.
- [69] Tor Project, «¿Qué ataques se pueden dar contra el enrutamiento cebolla?,» Tor Project, [En línea]. Available: <https://support.torproject.org/es/about/attacks-on-onion-routing/>.
- [70] Y. Klijnsma, «Another #Tor hidden service exposed...,» X, 4 Agosto 2018. [En línea]. Available: <https://x.com/ydklijnsma/status/1025796349541769217>.
- [71] «My browser fingerprint,» Am I Unique?, [En línea]. Available: <https://amiunique.org/fingerprint>.
- [72] D. Arp, E. Quiring, C. Wressnegger y K. Rieck, «Privacy Threats through Ultrasonic Side Channels on Mobile Devices,» 28 Abril 2017. [En línea]. Available: <https://mlsec.org/docs/2017a-eurosp.pdf>.



Universidad
de Alcalá

2024/2025