ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

Deep Web: acceso, seguridad y análisis de tráfico
(Deep Web: Access, security and traffic analysis)

Para acceder al Titulo de

Graduado en
Ingeniería de Tecnologías de Telecomunicación

Autor: Ignacio Cagiga Vila

Enero - 2017

E.T.S. DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACION

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: Ignacio Cagiga Vila
Director del TFG: Roberto Sanz Gil
Título: "Deep Web: acceso, seguridad y análisis de tráfico"
Title: "Deep Web: Access, security and traffic analysis"

Presentado a examen el día: 23/01/2017

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre): José Luis Arce Diego
Secretario (Apellidos, Nombre): Alberto Eloy García
Vocal (Apellidos, Nombre): Roberto Sanz Gil

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal

Fdo.: El Director del TFG
(sólo si es distinto del Secretario)

Vº Bº del Subdirector

Trabajo Fin de Grado Nº
(a asignar por Secretaría)

Agradecimientos

En primer lugar quiero darle las gracias a Roberto, director de este proyecto, quien tuvo la idea y con quien paso a paso hemos ido descubriendo la Deep Web juntos. Muchas gracias por tu disposición y toda la ayuda que me has dado.

Por supuesto, gracias a mis padres, a mis hermanos y a mi tía, que si he llegado hasta aquí es gracias a ellos más que a nadie ni a nada.

Para acabar, muchísimas gracias a mis compañeros de clase con los que he pasado cuatro años fantásticos. Gracias Pablo, Edu, Javi y Juan.

Ignacio Cagiga Vila, Enero 2017

Resumen

Este trabajo pretende hacer un análisis técnico de la Deep Web en el ámbito de las redes y las tecnologías de Internet. La parte principal del proyecto puede verse dividida en dos partes: Acceso a la Deep Web como cliente, e implementación de un relay de la Tor Network.
La implementación de un relay de la Tor Network permite comprender como se consigue asegurar la anonimidad y seguridad de los usuarios que intentan acceder a la Deep Web a través de esta red.
La parte de laboratorio del proyecto se ha realizado con la maquina Raspberry Pi 3 modelo B, en la que hemos instalado Raspbian Jessie, basado en Debian, y sobre este sistema hemos realizado toda la programación necesaria por consola.

Abstract

This project aims to make a technical analysis of the Deep Web in the field of networks and Internet technologies. The main part of the project could be divided into to halfs: Access the Deep Web as a client, and implementation of a Tor Network relay.
The implementation of a Tor Network relay helps us understand how users security and anonymity is achieved. The laboratory work of the project consists on installing Raspbian Jessie on a Raspberry Pi model B. Over that OS based on Debian we program everything necesary in its console.

Índice general

Índice de figuras

Índice de tablas

Acrónimos

3DES	—	*Triple Data Encryption Standard*
AES	—	*Adcvanced Encryption Standard*
Arm	—	*Anonymizing Relay Monitor*
AS	—	*Autonomous System*
CBC	—	*Cipher Block Chaining*
CPU	—	*Central Processing Unit*
DNS	—	*Domain Name System*
DH	—	*Diffie-Hellman*
DHE	—	*Ephemeral Diffie-Hellman*
HDMI	—	*High Definition Intermedia Interface*
HTTP	—	*Hypertext Transfer Protocol*
INRIA	—	*Institut National de Recherché en Informatique et Automatique*
IP	—	*Internet Protocol*
IPv4	—	*Internet Protocol version 4*
IPv6	—	*Internet Protocol version 6*
ISIS	—	*Islamic State in Iraq and Syria*
ISP	—	*INternet Service Provider*
MIT	—	*Massachusetts Institute of Technology*
MitM	—	*Man in the Middle*
NSA	—	*National Security Agency*
OP	—	*Onion Proxy*
OR	—	*Onion Router*
PDF	—	*Potable Document Format*
RAM	—	*Ramdom Access Memory*
REP	—	*Robot Exclusion Protocol*
RSA	—	*Rivest-Shamir-Adleman*
SD	—	*Secure Digital*
SHA	—	*Secure Hash Algorithm*
SSD	—	*Solid State Drive*
SSH	—	*Secure Shell*
SSL	—	*Secure Service Layer*
TLS	—	*Transport Level Security*

TCP — *Transmission Control Protocol*
Tor — *The Onion Router*
UDP — *User Data Protocol*
URL — *Uniform Resource Locator*
USB — *Universal Serial Bus*
UTF-8 — *8-bit Unicode Transformation Format*
WWW — *World Wide Web*

Capítulo 1

Introducción

Nos encontramos en el año 2017, donde existen cerca de 4 billones de usuarios de Internet en todo el planeta Tierra. En esta nueva era, a la que el sociólogo Manuel Castells llama Sociedad Red [5], Internet juega un papel vital en nuestro día a día hasta tal punto que ha modificado áreas como el de las comunicaciones, la publicidad, los deportes, etc.

Aunque para la gran mayoría de los usuarios, Internet sobrepasa nuestras capacidades. Con más de 1 billón de sitios web y una cadencia de 10 nuevas páginas web cada segundo [6], Internet se describe en numerosas ocasiones como un gran iceberg.

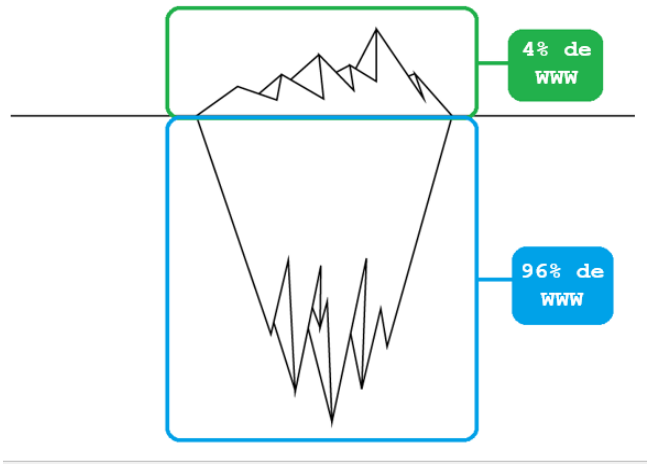

Figura 1.1: Representación metafórica de Internet.

Como sabemos, la cualidad más destacable de estas grandes masa de hielo es que, según el principio de Arquímedes, el 89 % de su masa total se encuentra sumergida bajo la superficie del mar.

Por un lado tenemos la parte visible del iceberg, Internet, al que diariamente nos adentramos para consultar las noticias de última hora, echar un vistazo a nuestro e-mail o cualquier otra cosa. Todas las páginas que se nos puedan ocurrir, tan solo comprenderán un 4 % de toda la World Wide Web (WWW) mientras que sumergida en las profundidades se encuentra la famosa Deep Web, de la que hablaremos extensamente en este proyecto.

Continuando con los símiles, podríamos relacionar a aquellos usuarios que consideran que ese 4 % de la WWW es suficiente con un capitán de barco que cree conocer todos los mares y océanos a la perfección; mientras que por otro lado se encontrarían aquellos que siempre necesitan saber más y conocer también lo que se haya oculto bajo los océanos, aquellos que optarían por subirse a un submarino y navegar en las oscuras profundidades.

Nosotros en este proyecto, escogimos subirnos al submarino.

1.1. Motivación

La motivación principal para el desarrollo del proyecto es poder arrojar un poco de luz sobre la Deep Web, de la que tan poco se ha escrito, y menos en el contexto de las telecomunicaciones. Por eso se propuso desplegar un relay de la Tor Network y comprender, desde el intercambio de tramas, cómo se garantiza la privacidad en una parte de Internet tan grande y de la que no sabemos casi nada.

Otra de las motivaciones es contribuir como voluntarios en la Tor Network a muchas personas que debido a la censura o situaciones de guerra, son reprimidos y se les prohíbe el acceso a Internet, quedándoles como último resquicio para hacerse oír la Tor Network.

1.2. Objetivos

El primer objetivo del proyecto es llevar a cabo un análisis técnico de la Deep Web. Cómo consiguen todos los sitios web de esta parte de Internet no ser indexados por los buscadores habituales como pueden ser *Google* o *Yahoo*, cómo funcionan los crawlers de búsqueda y por qué no consiguen penetrar en la Deep Web, de qué tipos de sitios web se compone, etc.

Como objetivo principal estudiaremos la Tor Network, que permite adentrarse en la Deep Web garantizando la privacidad y seguridad de sus usuarios. En el proyecto, se analiza el acceso como usuario, así como el funcionamiento de un nodo de la red, implementando un relay en una red doméstica y capturando todo el tráfico que fluye a través de él.

Capítulo 2

Estado del arte

En ocasiones, Internet se define como *'The Network of Networks'* o 'Red de redes', o también como *'The information Superhighway'* o 'La autopista de la información.

Efectivamente, Internet es una Red de Redes dado que es un conglomerado de infinidad de redes locales, redes de unos pocos ordenadores en un mismo edificio, hogar o empresa. Además, se la llama 'Red de Redes' porque es la más grande. Por otro lado, se la llama 'La utopista de la información' porque por Internet circulan constantemente cantidades increíbles de datos, entre prácticamente todos los países del planeta.

Es difícil establecer su tamaño exacto, ya que está en constante expansión y no existe una manera fiable de acceder a todo su contenido y, por consiguiente, determinar su tamaño. No obstante, un estudio del año 2005 estimó que existían 11.500 millones de páginas web. En 2008 ascenderían a 6.3000 millones según otro estudio. Actualmente, se cree que existen unos 47.000 millones de páginas web [6].
Todos estos estudios estiman el numero de sitios web basándose en los resultados obtenidos de distintos motores de búsqueda. Sin embargo, la información a la que se puede acceder a través estos es una ínfima parte del contenido de la web. En 1994 la doctora Jill Ellsworth utilizó el término de *'Invisible Web'*, para referirse a la información que los motores de búsqueda tradicionales no pueden encontrar. [7]

Dentro de este Internet oculto, infranqueable por los motores de búsqueda, se puede incluso hacer 2 distinciones:

- **Deep Web**, engloba a toda información que está disponible en Internet pero que únicamente es accesible a través de páginas generadas dinámicamente tras realizar una consulta en una base de datos. Es inaccesible mediante los procesos habituales de recuperación de la información.
- **Dark Web**, donde los servidores o host son totalmente inaccesibles desde nuestro ordenador. La causa principal se debe a zonas restringidas con fines de seguridad nacional y militar. Otros motivos son la configuración incorrecta de routers, servicios de cortafuegos y protección, servidores inactivos, "secuestro" de servidores para utilización ilegal o páginas de dudosa moralidad.

2.1. ¿Qué es la Deep Web?

Deep Web hace referencia a cualquier contenido de Internet que, por las razones que sean, no puede o no está indexado por los motores de búsqueda de los que disponemos hoy en día, como puede ser Google [8].

Como ya hemos comentado antes, con asiduidad se usan ambos términos para referirse a este contenido de Internet, Deep Web y Dark Web, pero no son lo mismo. La Deep Web está formada por todos aquellos sitios web, contenidos de bases de datos, etc. que no están indexados por los motores de búsqueda habituales; mientras que la Dark Web, de la que hablaremos ampliamente, hace referencia a una red encriptada a la que podemos acceder por medio de la TOR Network, es

una parte de toda la Deep Web [9].

También conocida como Deepnet, Dark Web o Hidden Web, frecuentemente es descrita usando como símil la imagen de un iceberg, de los que se dice que solo vemos el 11 %, mientras que el restante 89 % se encuentra sumergido, invisible al ojo humano.

Este submundo de Internet se empezó a conocer a finales de los 90, cuando se inició una investigación para intentar cuantificar su tamaño, en esos momentos se dieron cuenta que era mucho más grande de lo que habían avanzado, lo estimaron entonces en 2 o 3 veces el tamaño de la web conocida. En 2001, un estudio llevado a cabo por BrightPlanet estimó que la Deep Web contenía 500 veces más recursos que los indexados por los motores de búsqueda. Ya en 2008, la Deep Web se estimaba en un 75 % de toda la Internet (cerca de un billón de páginas web).

En la actualidad, estudios de la Universidad de Berkeley estiman que la Deep Web tiene un tamaño de 91,000 TeraBytes. Para hacernos una idea, la Wikipedia al completo, en su versión inglesa, ocupa 40 GigaBytes, es decir, más de dos millones de veces menos.

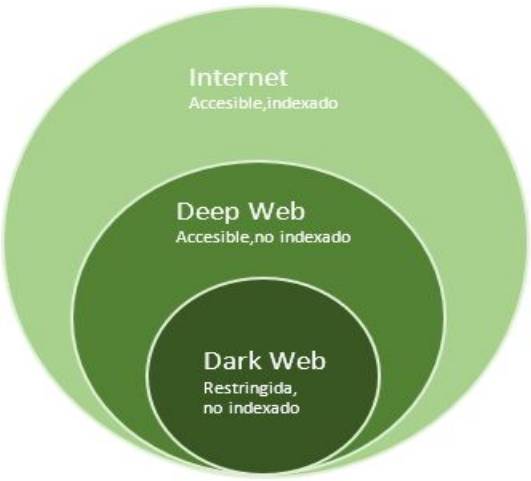

Figura 2.1: Estructura en capas de Internet

Este inmenso submundo de Internet está compuesto por páginas webs abandonadas con links rotos o que se han quedado desfasadas en una especie de mundo anticuado al que no se puede llegar por las actualizaciones posteriores de Internet. En su mayor parte este lado oscuro u oculto de Internet lo forman bases de datos de diversos sitios o gran cantidad de documentos PDF a los que no se puede llegar a través de los buscadores y debemos hacerlo, bien a través de una web central y con una contraseña específica, como puede ser el caso de las página de un banco donde los clientes pueden consultar sus movimientos.

El resto del Internet oculto está principalmente formado por páginas ilegales o para desarrollar actividades ilegales. Estas páginas recogen contenidos que pueden ir desde páginas de contacto de asesinos, para el tráfico de cualquier tipo de sustancia, hasta páginas de pederastia, páginas ilegales o censuradas en su país, páginas de hackeo o para liberar virus o páginas de terrorismo.

2.2. Crawling

Como ya se dijo anteriormente, en la Deep Web yacen todas las páginas webs que no pueden ser indexadas por los motores de búsqueda habituales, y digo habituales porque ya veremos que en la TOR Network disponemos de una serie de buscadores que si son capaces de rastrear dichas páginas.
Para poder conocer cómo el conjunto de la Deep Web consigue no ser indexado por los crawlers o bots, primero tendremos que conocer cómo trabajan estos:

> Imagina la World Wide Web como si fuese una red de paradas de autobús en una gran ciudad. Cada parada es un documento único (una página web, un PDF, una imagen, etc.). Los motores de búsqueda necesitan una forma de rastrear toda la ciudad y encontrar todas las paradas de autobús.

Cada vez que uno de estos crawlers encuentra una página web, descifra su código y almacena determinadas partes en enormes bases de datos. Cuando un usuario realiza una búsqueda, el motor de búsqueda se sumerge en los billones de documentos que componen estas bases de datos para ofrecerle al usuario los resultados más relevantes y populares.
La clave para permanecer a la sombra de los bots de búsqueda está en un archivo llamado robots.txt que contiene direcciones para los crawlers y links del sitio web. Mediante este archivo, los servidores HTTP de la Deep Web consiguen controlar el acceso a determinados directorios y archivos del servidor. Google está obligado a respetar estos permisos de crawling debido a los estándares Web. [10, 11]

Aunque también existen otras razones por las que una página no haya sido indexada por un motor de búsqueda, y alguno de los siguientes métodos también se emplean en toda la World Wide Web:

- Sitios que requieren autenticación
- Sitios con scripts. El bot puede indexar el código, pero no tiene por qué saber lo que hace (algunos scripts son capaces de encerrar a un bot de búsqueda en un bucle infinito).
- Sitios web dinámicos, que son creados bajo demanda y pasado un tiempo desaparecen, como pueden ser los horarios de un determinado vuelo.
- Sitios que han solicitado no ser indexados por medio del REP (Robot Exclusion Protocol)

En la última década se han desarrollado diversos proyectos, que tienen como objetivo común el indexado de toda base de datos de la web pública, lo que supone poner en práctica nuevas técnicas de crawling que sean eficaces en la Deep Web.

- DeepPeep es un motor de búsqueda desarrollado por WebDB group, en The University of Utah en 2009. [12]
- Sitemaps Protocol.
- Mod_oai. [13]

2.3. ¿Qué es la Tor Network?

La red Tor fue creada en 2003. En principio, estaba financiada por el Laboratorio de Investigación Naval de los Estados, el proyecto se encuentra actualmente en manos de *'Tor Project'*, una organización sin ánimo de lucro orientada a la investigación y la educación [3].

Tor son unas siglas que hacen referencia a "The Onion Router" (El encaminamiento cebolla). Es una red formada por servidores voluntarios que permiten a cualquier usuario de la Network, mejorar su privacidad y seguridad. Para conseguirlo, los usuarios en lugar de conectarse directamente a un servidor, lo hace a través de unos túneles virtuales de forma que no vean comprometida su privacidad. Se los conocen como túneles virtuales porque los relays de la Tor Network se comunican mediante el protocolo TLS sobre TCP/IP, manteniendo la comunicación secreta e íntegra, es decir, se garantiza que la información intercambiada entre dos relays de la red no será modificada por ningún agente externo.

¿Cómo funcionan esos túneles virtuales?

La Tor Network se compone de miles de relays voluntarios alrededor del mundo. Cuando alguien desea acceder a un servidor por medio del browser de Tor, este navegador automáticamente crea un camino entre el usuario y el servidor a través de los relays. Cada relay a lo largo del camino solo sabe quién le ha proporcionado información y a quien se la proporciona. Ningún relay individual del camino conoce la ruta completa que toma un paquete. La seguridad yace en que con cada salto que da un paquete en la ruta, el cliente negocia un nuevo par de keys.

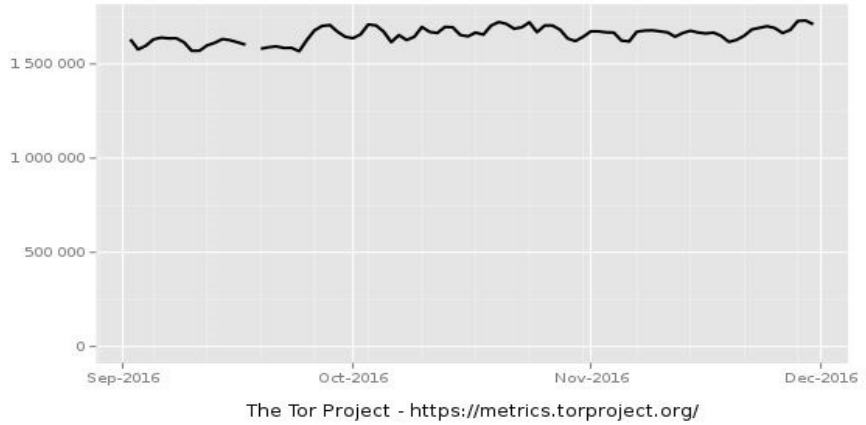

Figura 2.2: Actualmente, la Tor Network ofrece servicio a cerca de 2M de usuarios al día [1].

¿Cuántos nodos son necesarios para garantizar la seguridad del usuario en estos túneles virtuales?

Por norma general, en Tor Project se llegó a la conclusión de que el número óptimo de relays para constituir un camino lo suficientemente seguro sería de tan solo 3, uno de entrada, uno intermedio y uno de salida.

Para entender bien por qué cuando se navega por la Deep web a través de la Tor Network solo es preciso disponer de tres relays para llegar de forma segura al servidor web, primero es necesario saber cómo se establece la ruta que llevaremos. En primer lugar, es el cliente (o más concretamente el OP instalado en su ordenador, como veremos con detalle en el capítulo 3) quien 'elige' la ruta que se llevará y el único que puede saber qué tres relays se utilizarán en la ruta. El establecimiento de la ruta se puede simplificar con este sencillo esquema: [14]

```
Client needs to know Relay 1, 2 and 3's public key to do this
{
  Client sends to relay 1
  {
        Encrypted with Relay 1's key
        {
                Instruction to send to relay 2
                Encrypted with Relay 2's key
                {
                        Instruction to send to relay
                        Encrypted with Relay 3's key
                        {
                                Instruction to send to destination
                                {Data}
                        }
                }
        }
  }
}
```

Así en la configuración por defecto de Tor, todo cliente se asegura su anonimidad con rutas de tres nodos, aunque es fácilmente modificable. Sin embargo, desde Tor Project se recomienda que no se cambie esta opción, veamos por qué:

¿Por qué no pueden ser solo dos relays?

Mejoraría el throughput de la comunicación e incluso el usuario notaría una mejora en la velocidad, pero el problema se encuentra en la anonimidad del cliente. Si un atacante espía un Exit relay, algo relativamente sencillo ya que cualquiera puede montar un Exit relay en su casa y rastrear el

tráfico que pasa por él, podría conocer qué Entry Guard se ha conectado a él para establecer esa ruta de dos saltos. Es decir, podría ver quién se ha conectado a la Tor Network y a la vez podría saber con qué propósito.

¿Por qué no pueden ser más de tres relays?

Sería lógico pensar que cuantos más relays haya entre el servidor web y el cliente, más difícil le será actuar a los posibles atacantes. Pero se ha demostrado lo contrario, que más de tres nodos pueden dañar tu anonimidad.Además, la conexión se ralentizaría considerablemente y el throughput empeoraría.

"A Tor circuit consists always of three nodes. Longer paths may hurt your anonymity" [15]

2.4. Tipos de Relays

Hay cuatro tipos de relays que cualquier persona puede poner en marcha de forma voluntaria para formar parte de la Tor Network: Entry Guard, Middle relay, Exit relay y bridges.
Para una mayor seguridad, todo tráfico Tor pasa por al menos tres relays antes de alcanzar su destino, un Entry Guard, un Middle Relay y un Exit Relay.
Los **Entry Guards** ocupan siempre la primera posición de la ruta segura y son clave en garantizar la privacidad de la comunicación dado que protegen al usuario de una serie ataques concretos que veremos al término de este capítulo. Si no existieran Entry Guards y, en su lugar, se usara otro Middle Relay, podría darse la situación de que un atacante tuviese en su poder un gran número de relays, lo que aumentaría la probabilidad de que dicho individuo tuviese control sobre los dos primeros saltos de la ruta.
Cada cliente tiene un bajo número de Entry Guards adjudicados y siempre usa uno de ellos como nodo de entrada a la Tor Network. Cada 4-8 semanas, esta lista de Entry Guards cambia, y se le adjudican otros diferentes al cliente, lo que se conoce como *Rotation Period* [16, 17].

"The more often you change your guard node, the more chance there is of connecting to one owned by the bad guys" [18]

Poseer un Entry Guard no es tarea fácil. Dado que son los únicos nodos que no cambian para un cliente en su camino seguro, estos han de tener unas características especiales para que el usuario goze de una buena experiencia. Una vez un Middle Relay tenga un ancho de banda específico, una garantía de uptime considerable y una debida antigüedad en la red, le será otorgada la flag de 'Guard' [19].

Los **Middle Relays** añaden robustez a la red sin que el propietario del relay parezca ser la fuente del tráfico. No suelen ser los relays escogido por los atacantes ya que no manejan información relevante. Tan solo los nodos de entrada y salida ven información en claro.

Los **Exit Relays** pueden poner en compromiso la seguridad del propietario del relay si el poseedor de un dominio web quisiera utilizarlo para su uso fraudulento, ya que el servidor web ve la dirección ip pública del nodo de salida [20]. El último salto antes de llegar al destino siempre será un Exit Relay.

Implementar un Exit Relay en nuestras redes locales, sin embargo, no están recomendable. Es tan sencillo como construir un Middle Relay, solo que hay que modificar las políticas de salida para que pueda operar como nodo de salida en la red. Decimos que no es tan recomendable, porque por norma general, los ISPs no están a favor de que un cliente suyo posea un Exit Relay, entre otras cosas porque requieren de un ancho de banda simétrico bastante grande para operar en condiciones [21–23].

Los **Bridges** son Tor Relays que no se anuncian públicamente a la red. Son elementos claves en los que la asfixiante censura bloquea el acceso a la Tor Network. Por defecto, un cliente no se conectará a la red a través de un Bridge, si no que es necesario modificar la configuración de nuestro browser. Como los bridges no se anuncian, es necesario realizar una rápida búsqueda en la web para encontrar listas de estos nodos con sus direcciones ip, sus nicknames, y sus fingerprints, que será lo que facilitemos en el archivo *state* de la configuración.

En Tor, como en cualquier sistema de comunicaciones, encontramos un fallo de seguridad cuando el atacante puede ver al mismo tiempo los nodos de entrada y salida a la vez. La solución en la Tor Network para evitar estas comprometidas situaciones se haya en los **Entry Guards:**

> Supongamos que la Tor Network se compone de N relays, de los cuales el usuario puede observar C. Antes de iniciar la comunicación, el usuario escoge varios relays de entre los C que puede observar, que pasarán a ser sus "Entry Guards", y solo usará esos como primer salto en la comunicación. Así el usuario tiene una probabilidad (N-C)/N de evitar ser espiado por el atacante. [24]

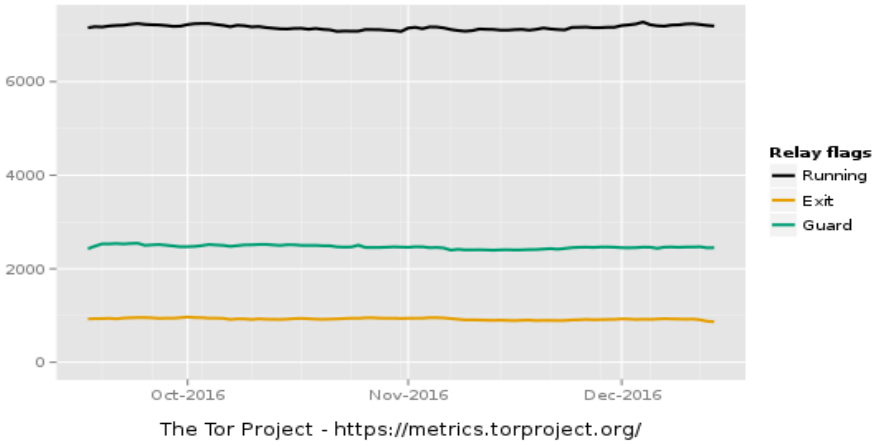

Figura 2.3: Comparación gráfica del numero de Entry Guards y Exit Relays frente al total de relays operativos en la Tor Network [1].

2.5. Contenido y repercusión social

Composición de la Tor Network:

Ya hemos hablado un poco de los entresijos técnicos de la Tor Network, pero, ¿con qué puede encontrarse un usuario en esa red? Tor hace posible a multitud de usuarios de todo el mundo surfear por Internet, chatear o enviar correos de forma anónima, lo que atrae a un gran abanico de personas que pueden usar la red con propósitos lícitos o ilícitos [25].
Pero como se verá al final de este apartado, varios estudios llevados a cabo por diferentes investigadores, indican que la gran mayoría del tráfico en Tor tiene fines ilegales.

Tor ha sido perseguido por la NSA de los Estados Unidos de América y por la agencia de inteligencia británica GCHQ, así como por la NCA británica. *The Economist* describe esta red como un oscuro rincón de Internet [26]. Acusada de difamación anónima, fuga desautorizada de noticias con información que pueda dañar la sensibilidad, infracción de copyright, distribución ilegal de contenido sexual, venta de substancias ilegales, venta de armas, robos de números de tarjetas de crédito, lavado de dinero, fraudes bancarios y robos de identidad, claramente la Tor Network junto con los Bitcoins, sirven de plataforma para el mercado negro en el siglo XXI.

No en cambio, en la página web de Tor Project, se puede encontrar un apartado llamado *¿Quién usa Tor?*, en el que detallan grupos de 'gente normal' que desean mantener privadas sus actividades en la red, gente preocupada con el espionaje online, periodistas o incluso cuerpos militares. Tor se

empieza a utilizar por muchas victimas de violencia doméstica y las agencias que les ofrecen apoyo, incluso ha ayudado a reducir el numero de victimas de stalking, o acoso en Internet. Pero, como dice un antiguo refrán, *ni los buenos son tan buenos, ni los malos son tan malos*:

Categoría	Porcentaje
Apuestas	0.4 %
Armas	1.4 %
Chat	2.2 %
Noticias	2.2 %
Abusos	2.2 %
Libros	2.5 %
Directorio	2.5 %
Blog	2.75 %
Pornografía	2.75 %
Hosting (servers)	3.5 %
Hacking	4.25 %
Búsqueda	4.25 %
Anonimidad	4.5 %
Foros	4.75 %
Falsificaciones	5.2 %
Chivatazos	5.2 %
Wiki	5.2 %
Correo	5.7 %
BitCoin	6.2 %
Fraude	9 %
Mercado	9 %
Drogas	15.4 %

Tabla 2.1: Servicios ocultos en la Tor Network en Enero de 2015 [3].

Categoría	Porcentaje
Violencia	0.3 %
Armas	0.8 %
Social	1.2 %
Hacking	1.8 %
Pornografía ilegal	2.3 %
Nexus	2.3 %
Extremismos	2.7 %
Desconocido	3 %
Otro contenido ilícito	3.8 %
Financias	6.3 %
Drogas	8.1 %
Otros	19.6 %
Ninguno	47.7 %

Tabla 2.2: Servicios ocultos en la Tor Network en Febrero de 2016 [3].

Tor y los derechos y libertades:

Ya se dijo en el apartado anterior, que Tor atrae a personas de diferentes ideologías, religiones y nacionalidades, con fines totalmente distintos. Tor ha sido utilizado por empresas de criminales, grupos activistas, agencias gubernamentales, etc. y en ocasiones, de forma simultánea. Muchas veces se ha dicho que Internet es un reflejo de la sociedad, cualquier acontecimiento que ocurra (golpes de estado, atentados, elecciones, etc.), afecta a Internet, y por extensión, a la Tor Network.

Lamentablemente,hoy estamos a la orden del día con acontecimientos de esta índole: la situación

política de la que gozan algunos países, así como la amenaza de guerra existente a nivel global, ha puesto la censura digital y el espionaje a la orden del día. Durante los últimos cinco años nos hemos despertado casi cada día escuchando noticias de nuevos bombardeos en la zona de Siria, de regímenes dictatoriales que oprimen las libertades de sus ciudadanos, del fin de la privacidad en redes sociales... [27]

Así que no nos va a resultar nuevo nada de lo que hablaremos en este apartado en el que repasaremos la influencia de la Deep Web y de la Tor Network en los últimos años:

- Hoy en día existen en el mundo cerca de 50 estados gobernados bajo un régimen dictatorial. Para nosotros puede que resulte difícil de imaginar como sería la vida en un país en el que la televisión, la radio, la prensa e Internet se encuentren censurados, pero hay billones de personas que se encuentran en esta encrucijada.

  En 2011 en Egipto tuvo lugar una de las protestas mas importantes de la historia moderna del país. Las protestas se originaron debido a la brutalidad policial, el alto desempleo, carencias de alimentos, represión de libertades de opinión...Muchos ISPs egipcios desactivaron sus *BGP route announcements*, lo que imposibilita la comunicación con el extranjero para hacer llegar al mundo noticias o declaraciones de las revueltas. [28]

  A principios de 2014, el primer ministro de Turquía, Recep Tayyip Erdogan, bloqueó el acceso a la red social *Twitter* y posteriormente a *Youtube* debido a la publicación de vídeos en lo que se implicaban a miembros del gobierno en tramas de corrupción. [28, 29]

  Por la misma época, en Febrero de 2014, se dieron en Venezuela unas revueltas bastante importantes en materia de educación. Mientras *Twitter* se convertía en el foco informativo de las protestas, el portavoz de esta red social aseguraba que varios ISPs venezolanas estaban bloqueando la entrada de imágenes a la red social. [30]

  En al actualidad China posee el mayor firewall frente a la Tor Network, en este país solo es posible acceder a la red por medio de bridges, que son relays que no se publicitan por miedo a ser bloqueados. [31]

  Y como estos, se podrían detallar infinidad de casos en los que la Tor Network es el salvavidas de los derechos humanos y de la libertad de expresión. Gracias al enrutamiento a través de tres relays de la red, los periodistas pueden informar de la situación de represión de un país y conseguir hacerse oír.

País	Caídas	Mejorías
Kazakhstan	48	46
Irán	47	51
Christmas Island	46	3
China	43	42
British Indian Ocean Territory	41	0
Kiribati	39	6
Netherlands Antilles	37	0
Norfolk Island	28	1
Niue	27	3
Cocos (Keeling) Islands	23	5

Tabla 2.3: Top 10 países con posibles situaciones de censura en 2016 [4].

- Por otro lado, la Tor Network no siempre recibe alabanzas. La Deep Web siempre se la imagina como un sitio oscuro de Internet, donde pedófilos, terroristas y traficantes encuentran cobijo.

  En los últimos cinco años, en los que el grupo terrorista *ISIS* ha causado estragos, se ha acusado a la Tor Network de ser el canal de mensajería preferido por los terroristas, oculto

a los ojos de la NSA.

En materia de activismo, hay dos casos que destacan:

- *WikiLeaks* es una organización mediática internacional sin ánimo de lucro que publica a través de su sitio web informes anónimos y documentos filtrados con contenido sensible en materia de interés público, preservando el anonimato de sus fuentes. La organización se ofrece a recibir filtraciones que desvelen comportamientos no éticos ni ortodoxos por parte de los gobiernos, así como asuntos relacionados con religiones y empresas de todo el mundo. [32]
- En la misma línea, son muchos los que dicen que el grupo activista *Anonymous* actúa a través de esta red, buscando adeptos, usando sus canales de mensajería segura y enseñando a futuros activistas. La mecha del grupo se prendió con el caso Wikileaks, por lo que se declararon enemigos de los enemigos de Wikileaks.

2.6. Vulnerabilidades de la Tor Network

Actualmente, la Tor Network tiene una serie de debilidades que han sido explotadas y que han sido documentadas por diversas universidades y centros de investigación. ¿De qué formas puede un posible atacante poner en peligro la anonimidad de un usuario y de qué forma la Tor Network esta diseñada para defenderse de este tipo de ataques?

La gran mayoría de los ataques en Tor se basan en intentar apoderarse de uno o dos relays de la red, normalmente el de entrada y el de salida de un mismo circuito. Los ataques pasivos monitorizan el intercambio de mensajes sin interferir, mientras que los ataques activos modifican el trafico de alguna manera para que el análisis sea mas sencillo. A continuación veremos que otros tipos de ataques se han desarrollado contra esta red. [33]

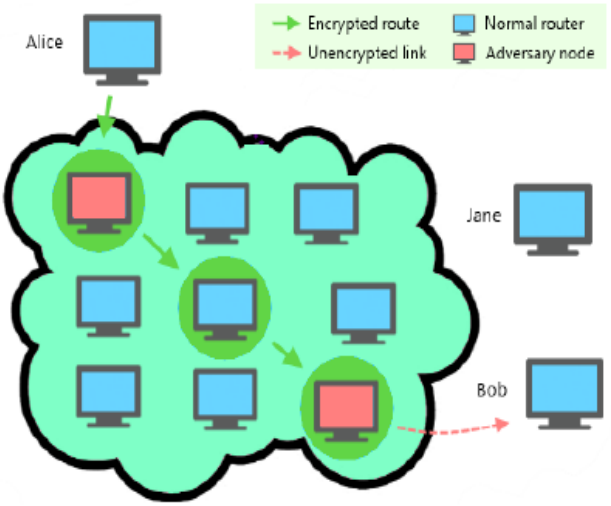

Figura 2.4: El atacante intentará tener en su poder el Entry Guard y el Exit relay.

Categorías de ataques:

Juha Salo, en su paper titulado *Recent attacks on Tor* dividía los ataques en cinco categorías dependiendo de su naturaleza [34]. A continuación se detallarán brevemente:

- **Modelos probabilísticos**

  Utilizan modelos probabilísticos de redes Bayesianas para medir lo segura que una red puede ser [35]. Estos modelos consideran la red como una caja negra y crean modelos matemáticos sumiendo dos cosas: primero, que un único usuario está conectado a una salida, y segundo, que la salida se puede vincular de nuevo al usuario en caso de que estén bajo observación. Aunque estos modelos no constituyan un ataque como tal, si que son capaces de cuantificar la facilidad que tendría un atacante para sacar a un usuario de su anonimato.

- **Ataques sobre Entry Guards y Exit Relays**

  Tienen como objetivo que los relays tengan mas posibilidades de ser elegidos como nodos de entrada o salida en una ruta , en teoría, segura para el cliente. Salo habla de dos ataques dentro de esta categoría:

  - Comprometer la anonimidad del usuario creando ciclos o *loops* entre relays obligándoles a rechazar nuevas conexiones, lo que básicamente acaba por inutilizarlos [36]. Así aumentarán las posibilidades de que los relays del atacante, o *Rotten Onions* que los llaman en inglés por el juego de palabras, sean escogidos como Entry Guard o Exit Relay.
  - El segundo grupo de ataques de esta categoría se basan en falsear el ancho de banda publicado por un relay [37]. Esto lo consiguen ofreciendo por ejemplo 1 Mbps a los circuitos que se establezcan con las *Directory Authorities* que son quienes estiman el ancho de banda, para posteriormente ofrecer mucho menos ancho de banda al resto de conexiones. Las probabilidades de que un relay sea elegido frente a otro, radican en cual de los dos ofrece un mayor ancho de banda y por tanto una mejor experiencia al usuario.

- **Ataques basados en análisis de tráfico y tiempo**

  Esta categoría cubre la mayoría de los ataques a la Tor Network. Intentan debilitar la anonimidad estudiando patrones en el tráfico, como por ejemplo, el tiempo de tránsito de paquetes. El atacante podría entonces correlar trafico entrante y saliente en la red.

  Los ataques activos en esta categoría son capaces de poner marcas de agua en el contador de las cells para que luego sea mas sencillo su análisis, o incluso hacer un ataque MitM y modificar el tráfico HTTP [38–43].

- **AS y ataques a nivel global**

  Se asume que un atacante capaz de observar o manipular gran parte del tráfico que entra y sale de la red estaría en disposición de usa análisis de tráfico y tiempo para reconocer torrentes de información. Tor no está diseñada para proteger al usuario de ataques provenientes de un sistema de espionaje autónomo(AS), sin embargo, en 2012 LASTor diseño un algoritmo para que el cliente evitase elegir los segmentos en los que se haya predecido una posible amenaza por ASes.

- **Vulnerabilidades en los protocolos**

  Yang Zhang, en su paper titulado *'Effective Attacks in the Tor Authentication Protocol'* [44] sugiere que hay una vulnerabilidad en la forma en que se distribuyen las claves de sesión. Estos ataques buscan aprovecharse se posibles brechas en los protocolos de comunicación.

Actualmente parece que atacar a la Tor Network sea un área de investigación bastante popular a juzgar por la cantidad de papers y proyectos que pueden encontrarse. Estos son algunos de los más famosos que se han desarrollado desde el 2010 hasta la fecha:

- **Bad Apple Attack**

  En Marzo de 2010, investigadores del INRIA en Ronquencourt, Francia, documentaron un ataque capaz de descubrir la dirección ip de usuarios de BitTorrent en la Tor Network [45]. Los investigadores fueron capaces de rastrear el 9 % del tráfico que pasaba por sus Exit Relays y así consiguieron revelar 10.000 usuarios de BitTorrent.

  El ataque consistía en dos partes. En la primera de ellas, los atacantes explotarían una aplicación insegura, como es BitTorrent, para conseguir la dirección IP o un rastro del usuario Tor. En la segunda parte, explotarían las capacidades de la Tor Network para asociar, esta vez, una aplicación segura con la dirección IP obtenida.

  Se le llama Bad Apple Attack por el famoso dicho *"Una manzana podrida estropea las demás"*.

- **Ataques al nivel de aplicación**

  En 2011, investigadores de la Universidad de Sudeste de China y el Colegio de Información Tecnológica de Changzhou, también en China, publicaron un paper en el que describían un ataque MitM a redes de baja latencia basadas en TCP, como es el caso de Tor [46]. Hay dos estrategias para dirigir el ataque: en la primera, el nodo de salida responde a las solicitudes

de conectarse a un servidor web de un cliente, con páginas web falsas y crea un patrón de tráfico diferente para que analizarlo sea más sencillo; por otro lado, la segunda estrategia es algo similar, solo que si envía la página solicitada al cliente, pero introduce invisibles enlaces en la página.

La primera estrategia sería más fácil de detectar dado que introduce un cierto retraso en la comunicación.

- **CellFlood**

  Este ataque se utiliza para dejar relays inoperativos enviándole continuamente solicitudes hasta que tenga que rechazar a nuevos clientes [47]. Este ataque *Denial-of-Service* o DoS fue descubierto en 2013 por cuatro investigadores de la Universidad de Sapienza en Roma y de la Universidad de Columbia en Nueva York, así como una defensa ante este ataque a la que llamaron *client puzzles* que posteriormente fue añadido a Tor.

  El ataque se aprovecha del hecho de que encriptar mensajes con una clave pública es 20 veces más sencillo que abrir mensajes con una clave pública. La primera vez que se crea el circuito, las claves de sesión se distribuyen utilizando la clave pública del relay con las CREATE cells. El coste de enviar inmensas cantidades de CREATE cells a un relay es mas asequible para un atacante que para el relay atacado.

- **EgotisticalGiraffe**

  En el año 2013, Edward Snowden se volvió famoso a nivel global debido a que filtró miles de documentos de alto secreto de la Agencia Nacional de Seguridad (NSA) de los Estados Unidos []. Uno de todos esos miles de documentos, era una presentación de PowerPoint en el que la agencia describía como eran capaces de identificar a algún usuario de Tor.

  El ataque consta de dos fases:

  - La primera parte tendría como objetivo identificar usuarios en la red y redirigirlos para que descarguen un malware que servirá para identificarlos en un futuro.
  - La segunda parte del ataque consiste en hacer un MitM para infectar el ordenador del usuario. A continuación se dirige a la víctima a un servidor especial que infectará su browser con un virus diseñado específicamente [48].

- **Sniper Attack**

  En 2014, cuatro investigadores de la Universidad de Berlin y de NRL encontraron nuevas debilidades en los algoritmos de control de flujo de Tor [49]. Este ataque consume muy pocos recursos para el atacante y se utiliza para inhabilitar relays de la Tor Network arbitrariamente. A su vez, diseñaron tres defensas antes el Sniper Attack, una de las cuales fue implementada poco después por Tor.

  El control de flujo se utiliza en Tor en ocasiones en que es necesario enviar grandes cantidades de datos y el receptor puede saturarse, de forma que se controla el ritmo con que se envían los datos.

  En el caso en que un cliente solicite una descarga de un archivo demasiado grande, el control de flujo permite al cliente controla el ritmo y que en este caso, el Exit Relay almacene en su buffer los datos. El Exit Relay es capaz de almacenar hasta 1000 cells y enviar 100 cells cada vez que reciba una SENDME cell proveniente del cliente (como consecuencia, añade otras 100 cells a su buffer):

- **Ataque de la confirmación de tráfico "Relay early"**

  Sobre este ataque aún no hay ningún paper escrito, ni se sabe a ciencia cierta la identidad de los atacantes, si lo hicieron con fines académicos o no, o si obtuvieron datos relevantes de los clientes que pudieran poner en compromiso su seguridad, pero se le ha dado mucha publicidad en los últimos años [50, 51].

Figura 2.5: Descripción gráfica del Sniper Attack.

Los atacantes utilizaron relay que estuvieron operativos entre el 30 de Enero de 2014 hasta el 4 de Julio de ese mismo año, cuando Tor Project los eliminó de la red [52].

El ataque consistía en aprovecharse de las Relay Early cells, que se utilizan para evitar que se creen caminos demasiado largos pero los atacantes las aprovecharon para enviar direcciones de servicios ocultos desde los nodos de salida, a unos Entry Guards que se hallaban bajo su control.

- **Huella dactilar del ratón**

  En Marzo de 2016, el investigador de seguridad español, José Carlos Norte, CEO de la startup eyeOS, demostró que podría correlar los movimientos del ratón que un usuario realiza tanto en el Tor Browser, como en un navegador habitual [53]. Lo hace a través de medidas de tiempo por JavaScript.

  Este ataque fue posible gracias a que Tor project no pudo solucionar un problema relacionado con las medida de tiempos de JavaScript, que duró 10 meses.

Capítulo 3

Acceso a la Deep Web

Con el término Deep Web no se hace referencia directamente a lo que se conoce como Darknet o Dark Web, pues solo una parte de su gran contenido está formado por sitios donde es necesario ingresar de forma anónima, si no a la totalidad de este gran resquicio de Internet.
En caso de que se quisiera bajar más en las profundidades de la red y acceder a la web oscura en sí, es conveniente utilizar programas que se encarguen de ocultar la identidad del usuario, garantizando su seguridad y privacidad. Algunas de estas herramientas se pueden encontrar fácilmente por Internet, como es el caso de Tor, una aplicación que permite pasar al lado oscuro de la web sin ser detectado.

En este capítulo será detallado qué ocurre en la Tor Network cuando se intenta acceder a ella como clientes, es decir, desde el Tor browser. Qué pasos sigue la maquina del usuario para conseguir esa privacidad tan deseada, cómo se crea la ruta de relays que siguen los mensajes hasta el servidor web, resolución de un pseudo-dominio *'.onion'*, etc.

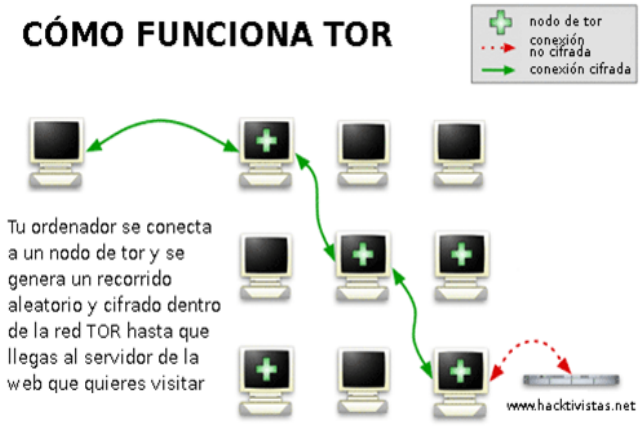

Figura 3.1: Esquema de funcionamiento de Tor.

3.1. Creación de una ruta segura

De entre todos los relays que componen la Tor Network, hay unos pocos que son la clave en la creación de enlaces seguros, son los *'Directory Authorities'*. En primer lugar se hayan los *'Directory Authorities'* principales, que publican una base de datos en la que es posible encontrar datos descriptivos de cada relay, como pueden ser su ancho de banda, nickname, etc. En segundo lugar están los *'Directory Caches'* que desempeñan el papel de caché y backup para los primeros.
De estos relays específicos, el resto de relays recogen información sobre la red que será determinante para establecer el camino adecuado.

Cuando un usuario inicia su Tor Browser, lo primero que hace su maquina es establecer un enlace con uno de los Entry Guards que tiene asignados. En el directorio \Data\Tor se encuentra un archivo de texto llamado *state* donde es posible encontrar qué nodos de entrada están asignados.

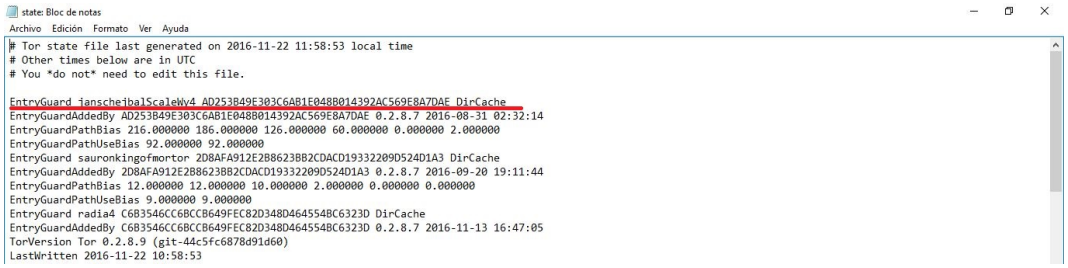

Figura 3.2: Captura del archivo *state*

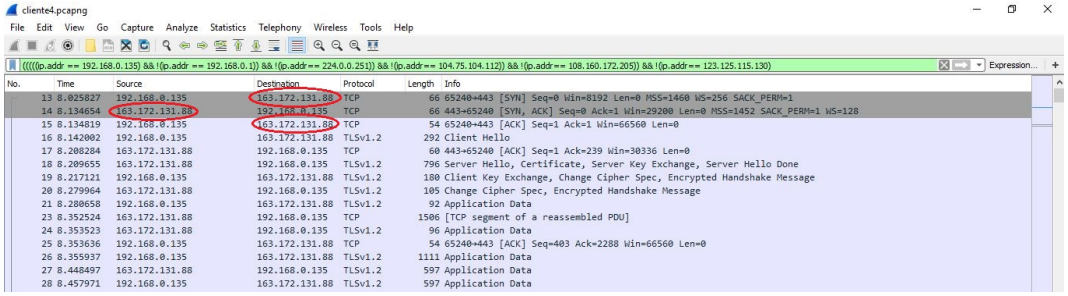

Figura 3.3: La dirección IP señalada pertenece al relay con nickname janschejbalScaleWy4 [2]

Todos los usuarios ejecutan un software en sus maquinas conocido como *'Onion Proxy'* u *OP*, cuya función es recabar información de los *'Directory Authorities'*, establecer circuitos y cifrar la conexión. Cuando se inicia el Tor Browser en una maquina porque un individuo desea acceder a una pagina web concreta, nuestro OP ya conoce cual será el nodo de salida idóneo para llegar al servidor web, así que decide un camino específico para ese nodo de salida dado, un camino que de forma predeterminada se compone de tres nodos. De esos tres nodos, nuestro OP ya conoce cual será el *Exit Relay* y también sabe cual será el *Entry Guard* porque son conocidos como hemos visto anteriormente.

A continuación, el OP negocia las claves de cifrado. Las claves simétricas se derivan de una clave compartida que se obtiene gracias al protocolo de establecimiento de claves de Diffie-Hellman. Dichas claves se crean a partir de un intercambio de mensajes, no se la envía el OP de forma cifrada, por lo que cuando la sesión finalice y se destruyan las claves de sesión, aunque el relay se vea comprometido, será imposible para un atacante recuperar las claves de sesión y descifrar la conversación. Algo conocido como **perfect forward secrecy**

Cuando el cliente quiera enviar un paquete, por ejemplo solicitar descargar una pagina web, el OP cifra el paquete con la clave acordada para el *Exit Relay*, después con la del *Middle Relay* y finalmente con *Entry Guard*. Por esto se usa la metáfora de una cebolla, porque cada relay podrá descifrar su 'capa de cebolla' correspondiente, así ningún relay podrá hacerse con la información original, solo conocerá el relay anterior y el posterior en la ruta.

Detalles técnicos

La comunicación ya sea entre los relays que componen la ruta, o entre el OP de cualquier cliente y el resto de relays, se lleva a cabo a través de células, del inglés *'cells'*. Estas cells tiene un tamaño fijo establecido en 512 Bytes, y se envían en registros TLS de cualquier tamaño.

En el caso especifico de que el intercambio de información se dé entre el OP y los relays de la ruta segura, pasaríamos a hablar de *'relay cells'*, o células de transmisión.

En cuanto al análisis detallado de la creación de un camino seguro, ya se vio anteriormente que era el OP del cliente quien negociaba las claves con cada relay , algo conocido como enfoque telescópico, del inglés ***telescopic approach***, que consiste en establecer una clave para el primer salto en primer lugar, posteriormente la conexión se tunela a través de ese enlace cifrado para establecer otra clave de sesión con el segundo salto del camino y así sucesivamente.
A continuación se detallarán los pasos y cells intercambiadas durante la creación de la ruta:

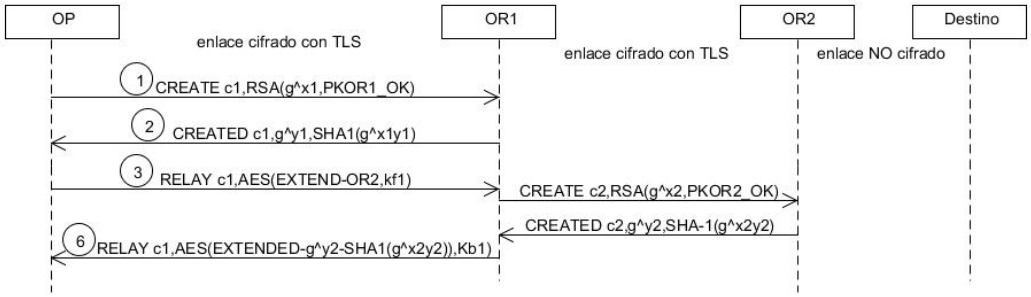

Figura 3.4: Intercambio de cells durante la creación del path

1. El OP del cliente escoge un *Exit Relay*, ORn.
2. El OP determina el camino formado por los relays OR1, OR2 y ORn.
3. El OP inicia una conexión TLS con el *Entry Guard*
4. El OP escoge un identificador para el circuito, que no esté utilizándose ya por el nodo de entrada escogido, lo que llamaremos *circID*.
5. El OP inicia un intercambio de clave de Diffie-Hellman con el nodo de entrada mandándole una cell CREATE.
6. El nodo de entrada responde con una cell de tipo CREATED. De este modo se obtiene una clave compartida Diffie-Hellman, de la que se derivan las dos claves simétricas, una para cada sentido.

   kf1 será utilizada para la comunicación del OP al OR1.

   kb1 será utilizada para la comunicación del OR1 al OP.
7. El OP envía una solicitud al OR1 para extender el circuito con una cell RELAY_EXTEND. Esta cell le indica a OR1 cual será el nodo OR2, su puerto, le especifica a OR1 una serie de estructuras de datos que habrá de enviarle a OR2 que hará posible un intercambio de claves con el protocolo Diffie-Hellman, sin que OR1 se entere.
8. El OR1 recibe la cell y comienza el procedimiento Diffie-Hellman, escogiendo él mismo un nuevo cirID. En la imagen se observa que en este paso, las cells CREATE y CREATED se intercambian entre OR1 y OR", sin que el OP interceda.
9. El OR1 manda al OP una cell RELAY_EXTEND para mandarle la respuesta de OR2, de forma que el OP sea conocedor de la clave compartida Diffie-Hellman. La comunicación se cifra de forma que OR1 no puede acceder al intercambio entre OP y OR2, y no es conocedor de las simétricas **kf2** y **kb2**.
10. el procedimiento continúa hasta que se hayan establecido las claves kf1, kb1, kf1, kb2, kfn y kbn, en el caso por defecto en que la ruta solo está compuesta por tres relays.

Algoritmos de cifrado utilizados [54]:

- Para establecer las conexiones TLS usa TLS/SSLv3.
  Todos los relays tienen que soportar

  SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA y deberían tener disponible

  TLS_DHE_RSA_WITH_AES_128_CBC_SHA.
  Los relays comúnes para comunicarse con los Exit Relays pueden usar:

  TLS_DHE_RSA_WITH_AES_256_CBC_SHA, TLS_DHE_RSA_WITH_AES_128_CBC_SHA,

  SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA, SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA
- Como algoritmo simétrico de cifrado se usa AES en counter mode (AES-CTR) con claves de 128 bits, con vector de inicialización con todos los bytes a 0
- Como algoritmo de clave pública usa RSA con claves de 1024 bytes.

- Como función resumen usa SHA-1
- Para establecimiento de claves usa DH (Diffie-Hellman)

3.2. Servicios Onion

Cuando un usuario se adentra en la Deep Web, rápidamente se dará cuenta de que las direcciones web ya no tienen una apariencia tan amigable como las que está acostumbrado a ver. Se acabó ver direcciones del tipo *https://www.google.com*, dentro de la Deep Web, las URL se parecen más a esto *http://zbnnr7qzaxlk5tms.onion/*.
Como vemos hay un par de cosas que cambian:

1. Nos olvidamos del famoso prefijo *www*.
2. Las direcciones ya no tienen unos nombres que los usuarios puedan leer. Son una combinación de 16 caracteres alfanuméricos generados automáticamente a partir de la clave pública del servicio.
3. La Tor Network utiliza su propio pseudo-dominio: *'.onion'*. No es un dominio como tal, lo que deriva en que las direcciones web no puedan ser resueltas por un DNS de la forma habitual.

Una serie de cambios vitales que provocan que cuando un usuario vaya a acceder a un sitio web, el procedimiento cambie con respecto a lo habitual. Para finalizar con este capitulo, se detallará el proceso de resolución de direcciones web hasta comprender en que difieren los servicios onion con los servicios web más habituales:

¿Qué es un dominio?

Un dominio de nivel superior no es más que un nombre que identifica a una máquina concreta. Cuando un cliente accede a un servidor web por medio de su URL, lo que en realidad tiene un lugar es una petición al sistema DNS para que traduzca ese nombre a la dirección IP de la máquina.
Ejemplos de dominios de nivel superior puede ser:

- .com (propósitos comerciales)
- .org (para organizaciones sin ánimo de lucro)
- .mobi (para sites móviles)

Pero como hemos dicho en la introducción de este capítulo, la terminación *'.onion'* propia para servicios de la Tor Network, no es un dominio como tal, se lo conoce como un pseudo-dominio, es decir, no participa en el servicio oficial de DNS.

¿Qué es un DNS?

El sistema de Nombres de Dominio o DNS tiene como función principal la de traducir una URL en una dirección ip. Por ejemplo, si no existiera este sistema de DNS, sería necesario recordar la dirección IP de cada servidor web. En lugar de acceder al sitio web que conocemos como *https://www.google.com*, habría que acordarse de que su servidor se encuentra en la dirección pública *74.125.196.105*.
Además de facilitar la memorización a los usuarios, el sistema de DNS hace posible que puedan existir más sitios web, dado que un mismo servidor puede albergar varios sites.

¿Cómo se resuelve una dirección web convencional? Cuando los usuarios, introducen una determinada URL (*www.ejemplo.com*) en un nuestro navegador habitual, este hace una consulta a Internet en la que solicita la dirección IP a la que pertenece dicha URL.

En primer lugar la maquina del usuario revisa su caché de DNS, en el caso de que ya hubiese visitado la pagina con anterioridad, es posible que el DNS esté almacenado en la memoria caché. Si no es así, esa solicitud se envía al servidor de DNS local, que será el de su proveedor de servicio (*ISP*).

En este paso los servidores de DNS de nuestro ISP revisan su caché en busca de la dirección IP a la que pertenece la URL que busca. Si no lo encuentra, entonces reenvía la solicitud a los servidores

raíz del dominio (*búsqueda recursiva*).

Finalmente el servidor de DNS envía la información obtenida de los servidores raíz al usuario, y se lo guarda en caché para futuras solicitudes. Ahora que el ordenador dispone de la información, el navegador realiza una solicitud HTTP al servidor de la URL *www.ejemplo.com* localizado en la dirección *192.0.10.2* y el servidor responde enviándo la página web.

¿Cómo se resuelven las direccinoes en la Deep Web?

Como ya hemos visto en la introducción de este apartado, en la Tor Network se trabaja con un pseudo-dominio totalmente nuevo, por lo que los servidores de DNS no resolverán las direcciones web como lo hacen habitualmente.

Continuando con la explicación de cómo se crea una ruta segura dentro de la Tor Network que es posible encontrar en este mismo capítulo, la resolución de una determinada dirección de la Deep Web, se lleva a cabo abriendo una conexión TCP una vez el cliente tenga creado el circuito [54]. El OP escoge un circuito que conduzca a un Exit Relay que sea capaz de conectarse al servidor web, escoge un StreamID arbitrariamente y construye una RELAY_BEGIN cell, en cuyo payload envía la dirección y el puerto del destino. El payload se divide en ADDRPORT y FLAGS:

- **ADDRPORT**

  Tiene la forma ADDRESS : PORT, donde ADDRESS puede ser una dirección IPv4 en dotted-quad format, una dirección IPv6 entre corchetes o un hostname de un DNS. PORT será un entero decimal entre 1 y 65535.

- **FLAGS**

IPV6_OK	IPV4_NOT_OK	IPV6_PREFERRED	*Reserved*

Tabla 3.1: Formato del campo FLAGS dentro del payload de una RELAY_BEGIN cell.

  El campo FLAGS ocupa 4 Bytes. El Bit 1 indica si soporta conectarse a direcciones IPv6, el Bit 2 indica si no desea conectarse a direcciones IPv4, dado que por norma general si se hará, el Bit 3 señala preferencia para conectarse a direcciones IPv6 si se dispusiera tanto de una dirección IPv4 como de una IPv6, y por último, el Bit 4 está reservado, los clientes no deben modificarlo y los servidores deben ignorarlo.

En cuanto el Exit Relay recibe la RELAY_BEGIN cell, resuelve la dirección y abre una conexión TCP con el puerto de destino. Si no pudiese resolver la dirección o fuese imposible abrir la conexión TCP, el nodo de salida responde al OP del cliente con una RELAY_END cell. En caso favorable, el relay responderá con una RELAY_CONNECTED cell con uno de los siguientes formatos:

- La dirección IPv4 a la cual se ha establecido la conexión (4 Bytes) más un TTL en segundos para los que la dirección puede almacenarse en caché (4 Bytes).
- Cuatro octetos a valor cero (4 Bytes), un tipo de dirección (1 Byte), la dirección IPv6 a la cual se estableció la conexión (16 Bytes) y un TTL en segundos para los que la dirección puede almacenarse en caché (4 Bytes).

Una vez la conexión se ha establecido, el OP y el Exit Relay empaquetan el torrente de datos en RELAY_DATA cells y una vez el nodo de salida recibe dichas cells, repite el contenido en el correspondiente circuito TCP.

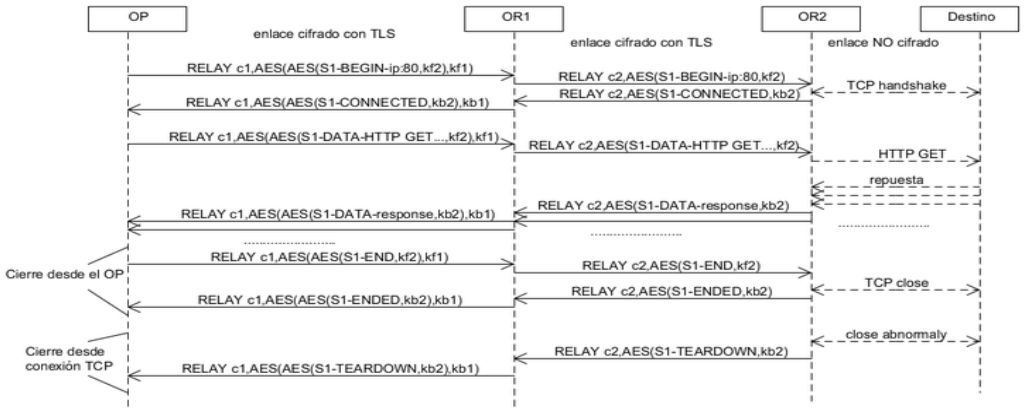

Figura 3.5: Intercambio de cells necesario para la resolución de direcciones web.

Capítulo 4

Implementación de un Relay Tor Network

En este capitulo nos centraremos en como se comporta un relay dentro de la Tor Network, ayudándonos de capturas de tráfico hechas sobre el relay que hemos desplegado.
Es necesario aclarar, que cualquier persona puede desplegar un relay voluntariamente en su hogar, pero es importante conocer las diferencias entre ser el poseedor de un Exit Relay o de un relay de otro tipo:

- En primer lugar, a la hora de configurar nuestro relay, la principal diferencia se haya en si queremos controlar un Exit Relay, o no, ya que un Entry Guard no es más que un Middle Relay pero con una serie de características que lo hacen idóneo para convertirse en nodo de guarda (gran ancho de banda, mucho tiempo en línea, etc.); y un Bridge no es más que un Middle Relay que no se hace público de cara al resto de usuarios.
- Si optamos por montar un Exit Relay, podremos ver información en claro, ya que nos conectaremos directamente a servidores de la Deep Web, y este enlace no está cifrado. A su vez, puede poner en peligro nuestra integridad como propietarios del relay, ya que el servidor web vería nuestra dirección IP pública en claro, y si tuviese malas intenciones, podría hacerse con mucha información privada.
- Alrededor de un 10 % de la totalidad de relays, son Exit Relays. Esto se debe a que requieren de un mayor ancho de banda, y a menudo, los ISPs no se encuentran cómodos con que uno de sus clientes posea un nodo de salida de la Tor Network.

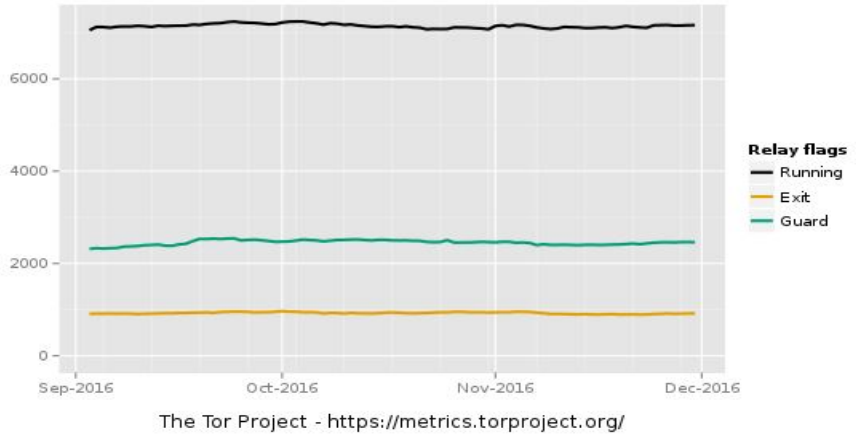

Figura 4.1: Comparativa del número de Entry Guards y Exit Nodes frente al total de relays de la Tor Network [1].

Como puede apreciarse, hoy en día la Tor Network compuesta por más de 7000 relays, de los cuales tan solo 900 son Exit Relays. Esto es consecuencia directa de las dificultades que conlleva implementar un nodo de salida.

Por eso, en nuestro trabajo nos hemos limitado a implementar un Middle Relay, y al final del documento detallaremos las directrices para implementar un Exit Relay como continuación de este proyecto.
A continuación detallaremos el desarrollo paso a paso del proyecto práctico, el funcionamiento de la aplicación de monitorización 'Arm', las dificultades con las que nos hemos topado durante el desarrollo de este proyecto práctico, y los resultados que podemos extraer gracias a la posesión de un Tor Relay.

4.1. Desarrollo del Proyecto

Antes de nada, nos gustaría aclarar que montar un relay de la Tor Network es algo que puede llevarse a cabo en cualquier computadora, pero hemos optado por implementarlo sobre una Raspberry Pi para que pueda estar operativo 24/7 y nos sea mas fácil modificar cualquier parámetro del relay [55].
Necesitaremos:

- Una Raspberry Pi modelo B (es necesario que sea modelo B porque así dispondremos de puerto Ethernet).
- Una tarjeta micro SD de 4GB .
- Un cable Ethernet.
- Acceso a una pantalla mediante HDMI, así como un teclado para los primeros pasos.

Comenzaremos por descargarnos la imagen de Raspbian Jessie (ultima versión disponible) y la grabaremos en nuestra micro SD con ayuda del software Win32DiskImager.
Una vez hayamos encendido la Raspberry y se haya iniciado el entorno de Raspbian, abriremos la ventana de comando y escribiremos *$ sudo raspi-config*. Se nos abrirá un menú de configuración de nuestra Raspberry, donde podremos cambiar, por ejemplo, nuestra zona horaria, nuestro hostname, las opciones de arranque, etc.

A partir de ahora, trabajaremos solo en la consola. Lo primero que haremos en la consola será comprobar que disponemos de la ultima versión de nuestras aplicaciones:
$ sudo apt-get update
$ sudo apt-get upgrade
Lo siguiente, antes de comenzar con nuestro proyecto será activar el servicio de SSH en la Raspberry con los siguientes comandos:
$ sudo apt-get install ssh
$ sudo /etc/init.d/ssh start
$ sudo update-rc.d ssh defaults
Para facilitarnos el trabajo, le otorgaremos a la Raspberry una dirección IP fija desde la configuración de nuestro router local para que cada vez qué apaguemos no se desconfigure la conexión mediante SSH. A partir de ahora no necesitaremos ni una pantalla ni un teclado, tan solo tendremos nuestra Raspberry conectada a la corriente y a nuestro router.

De aquí en adelante, cualquier modificación sobre nuestro relay la haremos desde un portátil conectado a la misma red doméstica a la que está conectado este. Procedemos a instalar Tor en nuestra Raspberry:
$ sudo apt-get install tor
Cuando termine la instalación, pasamos a editar el archivo de configuración del relay, conocido como 'torrc':
$ sudo nano /etc/tor/torrc
Hemos escogido el editor nano, pero igualmente podría haberse hecho con vi, Emacs o cualquier otro similar.
A continuación se indican las diez variables que se necesitan modificar y que suponen en el funcionamiento de nuestro relay:

- SocksPort 0

  Por defecto, Tor abre un proxy en el puerto '9050', debemos poner este valor a '0' si nuestro propósito es usar Tor solo sobre un relay y no utilizarlo además para conexiones locales.

- Log notice file /var/log/Tor/notices.log

  Todos los mensajes de nivel 'notice o superior se reencaminan a ese directorio. Ahí podremos acceder y conocer los últimos eventos que se han dado en nuestro relay (en nuestro caso fue vital para resolver los problemas iniciales).

- RunAsDaemon 1

  Esta línea nos permite empezar el proceso en un segundo plano.

- ORPort 9001

  Este será el puerto que se anunciará a las conexiones Tor entrantes.

- DirPort 9005

  Indica el servicio de directorio corriendo en este puerto.

- ExitPolicy reject

  Esta probablemente sea la modificación más importante, ya que con ella indicamos que queremos evitar convertirnos en un Exit Relay. Como hemos detallado en el capitulo 'Tor Network', ser propietarios de un Exit Relay puede poner en compromiso nuestra propia seguridad y vulnerabilidad.

- Nickname nachoTFG

  Un nick de libre elección para que se nos identifique, además de nuestra key.

- RelayBandwidthRate 100 KB

  Regula el tráfico a 100KBps (800 kpbs).

- RelayBandwidthBurst 200 KB

  Permitimos ráfagas de hasta 200KBps (1600 Kbps).

- ControlPort 9011

  El puerto en el que Tor escuchará conexiones locales. Para el desarrollo del proyecto es importante, dado que monitorizaremos el tráfico con una aplicación llamada Arm (*Anonymizing Relay Monitor*) y el ControlPort tiene que estar abierto.

Los siguientes pasos serán reiniciar nuestro servidor de Tor:
`$ sudo /etc/init.d/tor restart`
Comprobaremos que esté funcionando y operativo:
`$ cat /var/log/tor/log`
Tendrá que aparecer un aviso del tipo *[notice] Tor has successfully opened a circuit. Looks like client functionality is working.* Ahora que hemos configurado nuestro relay y somo 'voluntarios' del Tor-Project, solo nos queda instalar una aplicación para monitorear el tráfico que somos capaces de gestionar en tiempo real:
`$ sudo apt-get install tor-arm`
Para arrancar la aplicación introduciremos el siguiente comando en la consola:
`$ sudo -u debian-tor arm`

4.2. Monitorización con Arm

Para conocer en tiempo real el consumo de ancho de banda, las conexiones mantenidas, los avisos, etc. nos decantamos por usar la aplicación Arm (*Anonymizing Relay Monitor*) dada su sencilla interpretación e instalación [56].

La aplicación fue desarrollada por Damian Johnson en la Washington State University en 2009. Cuando iniciamos la app con el comando `$ sudo -u debian-tor arm` vemos la primera de cinco páginas de que dispondremos para supervisar nuestro relay. A continuación describiremos con ayuda de diferentes capturas la información que nos arroja cada una de las páginas sobre nuestro relay:

- **Página 1**

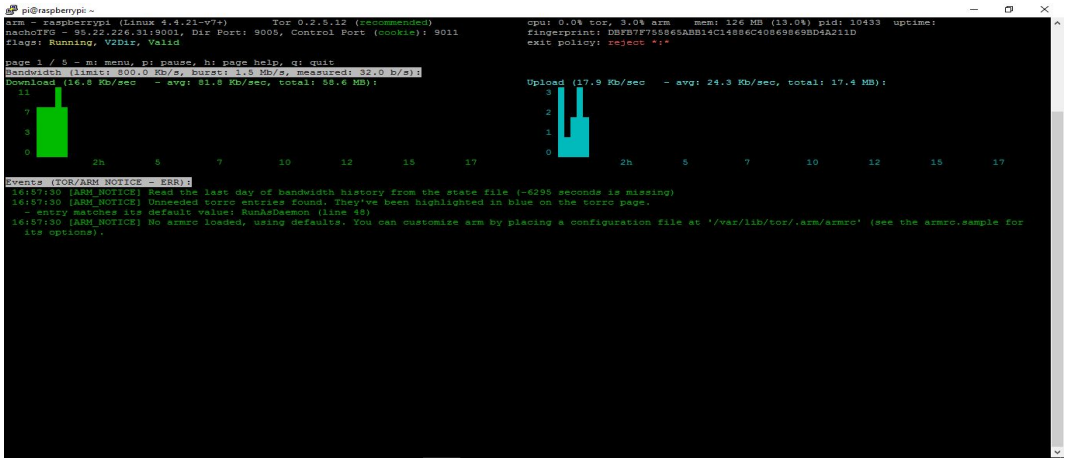

Figura 4.2: Página 1/5 de la aplicación Arm

Algo que es un factor común en todas las paginas de la aplicación es la parte superior, donde tenemos información básica acerca de nuestro sistema. Arriba a la izquierda encontramos la versión de nuestro OS, así como de Tor (bastante importante porque si la aplicación está deasctualizada, nuestra privacidad no se garantiza), nuestro ID, nuestra dirección IP pública, los puertos que habilitamos previamente en el fichero de configuración de Tor, y los flags que nuestro relay tiene activos.

En nuestro caso solo tenemos tres flags activadas, pero a continuación detallamos el significado de todas las flags posibles [57]:

- *Authority:* Si el relay es una 'directory authority'.
- *BadExit:* Si se considera inutilizable como Exit Relay .
- *BadDirectory:* Si el relay se considera inutilizable como 'directory cache'.
- *Exit:* Si el relay es mas útil para crear circuitos de salida. Flag clave para el algoritmo de descubrimiento de caminos.
- *Fast:* Si sirve para circuitos con un alto ancho de banda.
- *Guard:* Si puede utilizarse como Entry guard.
- *HSDir:* Si el relay es considerado un 'V2 hidden directory service'.
- *Named:* Si el mapeado de nicknames del relay está autorizado o reconocido.
- *Stable:* Si esta operativo para circuitos de larga duración.
- *Running:* Si está dispoonible, en funcinoamiento.
- *Unnamed:* Si otro relay esta utilizando el mismo Nickname
- *Valid:* Si el relay ha sido 'validado'.
- *V2Dir:* Si el relay implementa el 'V2 directory protocol'

Continuando con la descripción de la zona superior, en la parte derecha vemos reflejadas características más físicas de nuestro relay, como pueden ser, el la memoria utilizada o nuestra fingerprint.

En el centro de esta primera página de la interfaz tenemos dos gráficas, una en verde a la izquierda (*Download*) y otra en azul a la derecha(*Upload*). En el momento de la captura dichas gráficas reflejaban el ancho de banda utilizado por nuestro relay en vivo, con un refresco de 1 minuto, pero una de las características de Arm es su versatilidad...podemos aprovechar dichas gráficas para reflejar el numero de conexiones o incluso el porcentaje de CPU que estamos utilizando. También podemos alterar las escalas, el tiempo de refresco, etc.

Para terminar con esta primera página, bajo las gráficas tenemos el 'log' donde se recogen los avisos de mayor importancia, así como pequeñas ayudas o indicaciones.

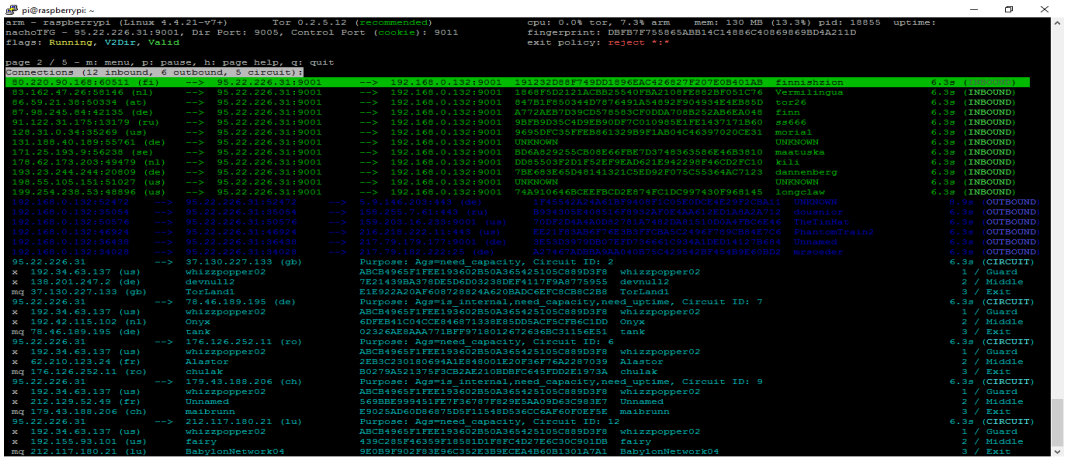

Figura 4.3: Página 2/5 de la aplicación Arm

- **Página 2**

  Esta segunda página es conocida como *página de conexiones* dado que podemos obtener información de los enlaces que mantiene nuestro relay en tiempo real. En nuestro caso específico, en el que nuestro relay no desempeñará labores de Exit Relay, ni de proxy para conexiones locales, podemos obtener tres familias de conexiones:

  - *Inbound:* Conexiones entrantes a nuestro relay.
  - *Outbound:* Conexiones salientes desde nuestro relay.
  - *Circuit:* Nuestro relay establece circuitos de tres saltos dentro de la Tor Network (como si un usuario cualquiera intentando acceder a través del Tor browser) para recabar información sobre el estado de la red, sobrecarga, testearse, etc.

  Por filas se nos muestra cada conexión que mantiene operativa nuestro relay. En cada fila podemos ver la dirección IP pública del otro relay que forma el enlace, nuestras direcciones IP públicas y privadas, la fingerprint y el Nickname del otro relay, y por último, el tiempo en segundos (s) o minutos (m) que el enlace lleva estable.

  Al pulsar la tecla *intro* se nos despliega más información que podría ser relevante, como la información de contacto (se suele escribir un mail de contacto por si hubiese alguna incidencia y un usuario quisiera ponerse en contacto con nosotros, pero en ocasiones también se suele escribir una dirección donde algún usuario, como gesto de agradecimiento, done bitcoins al dueño de un relay) o las flags que tiene activadas, por ejemplo.

  En el caso de los circuitos, cada vez que inicializamos Tor, se crean una serie de circuitos, algunos de ellos mueren a los pocos segundos porque solo se utilizan en el hipotético caso de que fuéramos a utilizar Tor como clientes, y no como un nodo de la red [58, 59]. Como podemos ver en la imagen anterior, para cada circuito se nos muestra una serie de atributos o argumentos de cómo se crea cada circuito:

  - *need_capacity :* El circuito solo incluye relays de gran capacidad
  - *is_internal :* El circuito no será utilizado para trafico como cliente.
  - *need_uptime :* El circuito solo incluye relays que llevan mucho tiempo operativos.

- **Página 3**

Figura 4.4: Página 3/5 de la aplicación Arm

Tercera página, página donde se nos muestra el archivo de configuración *torrc* de forma bastante amigable. De forma predeterminada vemos los valores que alteramos o activamos durante la instalación de nuestro relay, con el valor que les otorgamos y una breve descripción.

Desde la aplicación Arm podemos modificar cualquier valor de nuestro relay rápidamente sin tener que acceder con un editor de texto al directorio donde se encuentre el archivo *torrc*.

Pulsando la tecla *'a'* pasamos a desplegar el archivo de configuración completo, no solo los valores editados.

- **Página 4**

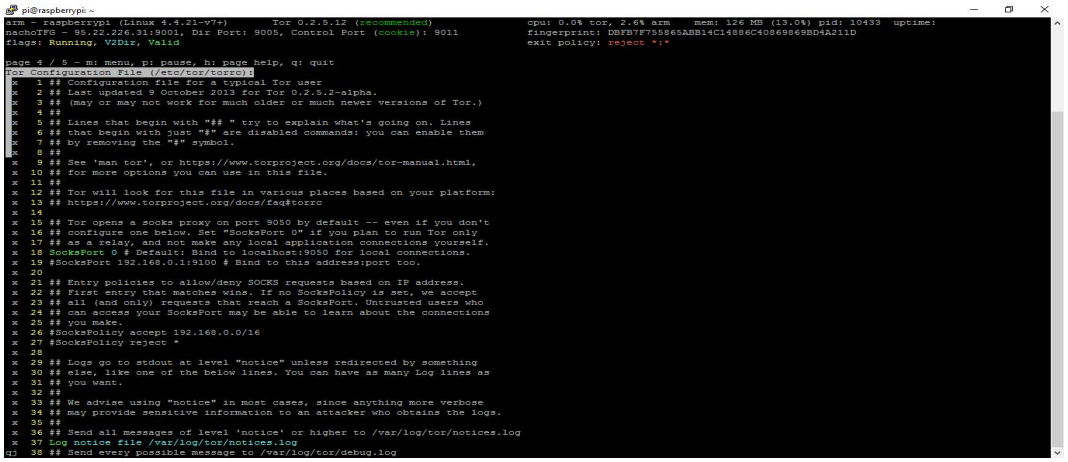

Figura 4.5: Página 4/5 de la aplicación Arm

La cuarta página es bastante similar a la anterior. Vemos desplegado el archivo de configuración *torrc*, pero esta vez de una forma más clásica, con un editor de texto y sin tanto colorido. Nos ahorramos tener que salir de la aplicación y acceder al directorio que alberga el archivo para cualquier cambio rápido.

- **Página 5**

Por último tenemos la página conocida como *Control Interpretor*. Esta pestaña viene a hacer las funciones de una consola dentro de la aplicación. Escribiendo el comando */help* nos aparecerá por pantalla el abanico de comandos que podemos ejecutar para obtener información de nuestro relay, las conexiones que se encuentran operativas, cerrar circuitos, conexiones,etc. Estos son un par de ejemplos de comandos, así como de su efecto:

- *GETINFO:* Solicita información de Tor.

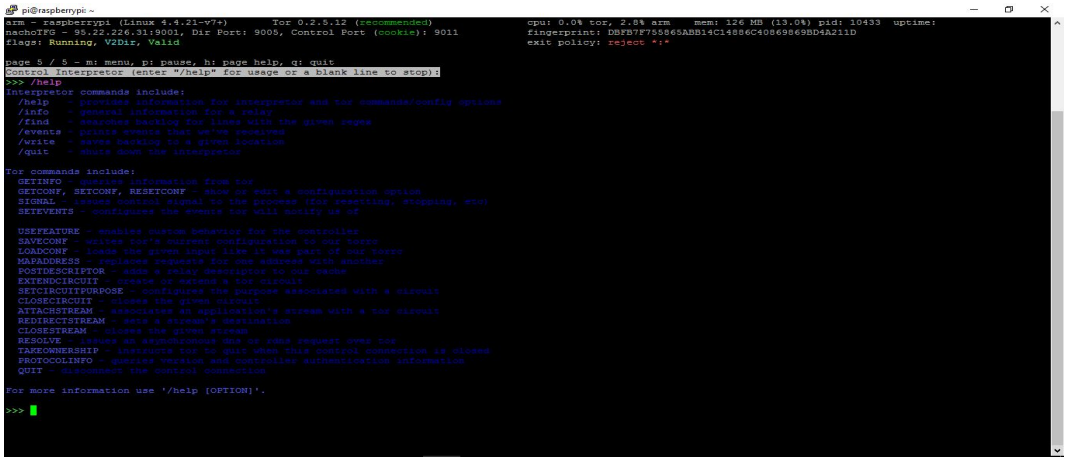

Figura 4.6: Página 5/5 de la aplicación Arm

- *GETCONF, SETCONF, RESETCONF:* Editar opciones de configuración
- *SETEVENTS:* Configurar los eventos que nos serán notificados por Tor.
- *CLOSECIRCUIT:* Cierra el circuito especificado.
- */info:* Arroja información sobre el relay que le indiquemos.

4.3. Complicaciones durante el desarrollo

En una primera aproximación consideramos que lo apropiado sería desarrollar el proyecto en el laboratorio de Telemática de la Universidad de Cantabria, dado que el material necesario nos lo cede este grupo. Además, la cercanía con el despacho del tutor de este proyecto, la disponibilidad de un espacio de trabajo las 24 horas del día y una conexión con gran ancho de banda, lo convertían en el lugar adecuado para instalar nuestro relay.
Tardamos una semana en comprender que era lo que fallaba. Al conectarnos a una institución académica, la gran mayoría de los puertos están bloqueados, por lo que, si asignábamos *ORPort* y *DirPort* a cualquier puerto, siempre recibíamos el mismo mensaje:
'Your server has not managed to confirm that its ORPort is reachable.'
En ese momento fue cuando optamos por olvidarnos de cualquier avance y volver a empezar el proyecto desde cero en mi casa.

Esta solución, desencadenó a su vez en otras complicaciones.
Cuando un usuario instala voluntariamente un relay en su hogar, durante los primeros 3 días, con el relay como causante, el ancho de banda de su conexión local sufre una serie de fluctuaciones y se comporta de manera muy extraña. Esto se debe a que el relay hace una serie de tests para calcular qué ancho de banda podrá ofertar a la Tor Network sin afectar en la medida de lo posible al resto usuarios que se conecten a esa red local.

Además, debido a las condiciones climatológicas y la infraestructura eléctrica del hogar, sufrimos una serie de cortes de luz, por lo que nuestro relay no lograba estar encendido de manera continuada el tiempo suficiente:
'Unable to pre populate bandwidth information (insufficient uptime)'
La solución fue utilizar un SAI, donde conectamos tanto el relay como el router, para que nuestro relay no sufriera cortes imprevistos y así pudiéramos obtener información estadística mas fiable.

Otra complicación que tuvimos que afrontar, esta vez cuando utilizábamos la aplicación de monitorización Arm, fue un simple error estético. La aplicación está diseñada para mostrar las conexiones establecidas en forma de árbol, bastante sencillo para el ojo humano, pero en nuestro caso las líneas verticales las mostraba como *'x'*, y las horizontales como *'q'* [60]. Tras varios días buscando la razón, encontramos que se trataba de un error de traducción entre formatos de codificación.
Pero el error no se daba en el relay, si no en la herramienta de Windows llamada *'Putty'* que empleamos para el control vía SSH. Por lo visto, dicho software tiene problemas de compatibilidad entre vt100 y UTF-8. A continuación se muestra la diferencia entre la apariencia desde un monitor externo, y desde un PC que está conectado al relay vía SSH:

Figura 4.7: Problema de representación de caracteres lineales en PuTTY

Figura 4.8: Representación gráfica en un monitor conectado vía HDMI

4.4. Funcionamiento como Middle Relay

Como ya dijimos al comienzo de este capítulo, cualquier relay puede desplegarse para que funcione únicamente como Middle Relay, o puede modificarse sus políticas para que, además, pueda trabajar como Exit Relay. En este apartado detallaremos el comportamiento de nuestro relay en el papel de Middle Relay.

¿Qué ocurre cuando el relay se conecta por primera vez a la Tor Network?

Absolutamente todos los relays que se implementen, la primera vez que se conecten a la Tor Network con fines de ayudar al encaminamiento cifrado de información, pasan por cuatro fases durante las cuales su ancho de banda sufrirá oscilaciones un tanto extrañas [61]:

- **Fase uno: No medido (Días 0-3)**

  Cuando iniciamos un relay por vez primera, este hace una fase de auto-test de ancho de banda. Para ello, el relay construye cuatro circuitos que se adentran en la Tor Network y regresan a él, y envía 125KB por cada uno de ellos. Este protocolo activa el sistema de medida pasiva de ancho de banda de Tor, que estima el ancho de banda de un relay como la ráfaga o burst más grande que ha hecho en diez segundos. Así que si todo va bien, en esta auto-medida tendremos:

  $$\frac{4 \cdot 125KB}{10} = 50KBps \tag{4.1}$$

  Nuestro relay publicitará un ancho de banda de 50 KBps en su descripción.

  Los *Directory Authorities* listarán nuestro relay en el consenso de la Tor Network. La experiencia que tienen los clientes es proporcional al ancho de banda de los relays que escogen.

  Al principio, los *Directory Authorities* publicitarían cualquier ancho de banda que un relay dijera haber estimado. Este enfoque hizo muy fácil que cualquier usuario que poseyera un relay y con no muy buenas intenciones, atrajera grandes cantidades de tráfico con solo mentir acerca de su estimación.

  En 2009, Mike Perry desarrolló unos scripts *'Bandwidth Authority'* con los que un grupo de ordenadores bastante potentes, llamados *bwauths* realizan medidas activas de ancho de banda sobre cada relay y los *Directory Authorities* ajustan hacia arriba o abajo el BW anunciado para ese relay.Estos *bwauths* funcionan comparando nuestro relay con otros de similares velocidades y características.

  A esto lo llamaremos peso en el consenso, del ingles *consensus weight* porque es una comparación de los números de nuestro relay con el resto de relays, no puede medirse como ancho de banda realmente.

  Y por eso, durante los primeros tres días aproximadamente, un relay podría entenderse como inútil dado su bajo ancho de banda publicado, hasta que este se ajuste correctamente.

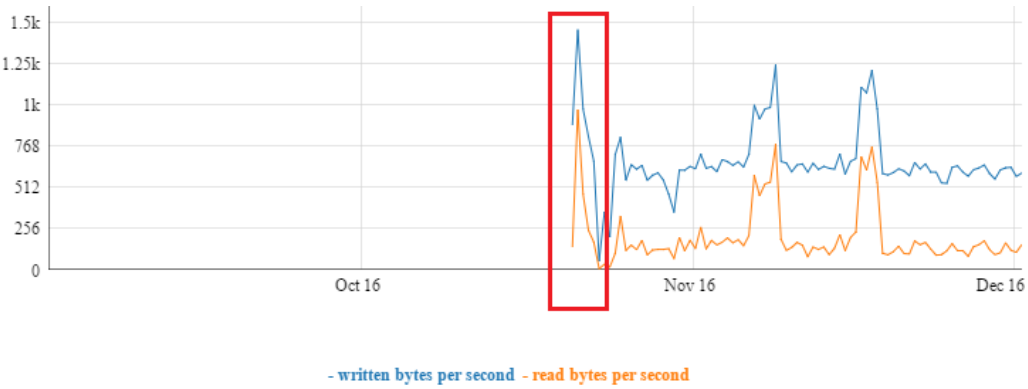

Figura 4.9: Histórico del BW de nuestro relay. En rojo, el comportamiento los primeros tres días.

- **Fase dos: Medición remota (Días 3-8)**

  Los *bwauths* de los que hablábamos anteriormente, funcionan comparando nuestro relay con otros de similares velocidades y características. Al principio, como no recibiremos apenas tráfico, nuestros *peers* serán aquellos relays que tampoco han visto tráfico. A medida que pasa el tiempo, habrá clientes que construyan circuitos ayudándose de nuestro relay, y la estima de ancho de bando pasiva que hace nuestro relay será mayor, lo que hará que los *bwauths* nos equiparen con relays de mayores velocidades, nuevos *peers*, lo que nos hace ganar más peso en el consenso. Esto tiene como consecuencia que más clientes nos utilizarán para intercambiar información, auto estimaremos que tenemos un ancho de banda mayor, y así cíclicamente.

  Como ya dijimos en el capítulo 2, Los Entry Guards son clave en garantizar la privacidad de los datos en al Tor Network, dado que nos protegen de un ataque concreto.

  Si en lugar de existir los Entry Guards, y en su lugar utilizásemos otro Middle Relay como nodo de entrada, y se diese el caso de que un atacante fuera el poseedor de un gran número de relays, existiría una alta probabilidad de que los dos primeros saltos de nuestra ruta estuvieran en las mismas manos, y pusiera nuestra conversación en peligro.

  Solo relays estables y de confianza pueden desempeñar el papel de Entry Guard, así que nuestro relay recién instalado no podrá ser nodo de entrada, ni de salida porque así los especificamos durante la configuración del mismo.

  En esto se basa la fase 2, nuestro ancho de banda irá en aumento a medida que recibimos más trafico, pero este estará limitado ya que solo recibiremos datos siendo el segundo salto de la ruta.

- **Fase tres: Evaluación como Guard Node (Días 8-68)**

  Los *Directory Authorities* son quienes asignan las flags a los relays. De entre todas las que hay, la que ahora nos interesa es la Guard Flag, que se asignan basándose en tres características:

  1. **Ancho de banda** Tienen que tener un alto peso de consenso.
  2. **Tiempo de actividad ponderado** Tienen que estar funcionando la mayor parte del tiempo.
  3. **Tiempo conocido** Los Guard Relays han de llevar un tiempo operando.

  Una vez un relay tiene la Guard Flag, podrá ser seleccionado como nodo de entrada a la Tor Network, pero lo curioso es que dejaremos de ocupar el segundo salto de una ruta cualquiera, no volveremos a ser un Middle Relay, ya que el Onion Proxy (OP) de un cliente, al ver la Guard Flag activada suponen que tenemos una gran carga de clientes que utilizan nuestro relay como Entry Guard.

  Es decir, nada más nos convirtamos en Entry Guard, sufriremos un declive en el trafico que maneja nuestro relay.

- **Fase cuatro: Afianciado como Guard Node (Días 68+)**

  Una vez un relay ha sido Entry Guard durante un *Rotation Period* completo, lo que puede llevar hasta 12 semanas, alcanza el nivel en que se afianza como un Entry Guard, en la que el numero de clientes que lo desecha de su lista de nodos de entrada, se ve compensado con los que lo añaden a la suya.

¿Qué ocurre nada más encender el relay?

Para conocer los protocolos que realiza un relay durante su inicialización. Hemos realizado varias capturas de tráfico mediante la herramienta *Wireshark* de aproximadamente 20 minutos cada una, para un análisis estadístico de los pasos que sigue nuestro relay. Pasos seguidos para la captura:

1. Detener el proceso Tor en el relay.

   ```
   $ sudo /etc/init.d/tor stop
   ```

2. Iniciar una captura en segundo plano con tcpdump. El '&' nos permite introducir otros comandos mientras la captura sigue su curso.

   ```
   $ sudo tcpdump /i eth0 /w  /NOMBRE.pcap &
   ```

3. Iniciar el proceso Tor.

   ```
   $ sudo /etc/init.d/tor stop
   ```

4. Regresar a la captura, que se estaba ejecutando en segundo plano.

   ```
   $ fg
   ```

5. Crear el directorio para la memoria USB con la que vamos a extraer las capturas para su posterior análisis en un ordenador.

   ```
   $ sudo mkdir /media/usb
   ```

6. Montar la memoria USB.

   ```
   $ cd   $ sudo chmod 775 /media/usb $ sudo mount -t vfat /dev/sda1 /media/usb
   ```

7. Copiar el archivo a la memoria.

   ```
   $ sudo cp NOMBRE.pcap /media/usb
   ```

8. Desmontar la memoria USB.

   ```
   $ sudo umount /media/usb
   ```

Como el Middle Relay se encuentra en una red doméstica, capturamos mucho tráfico 'basura' para nuestra investigación como puede ser comunicaciones vía SSH con un ordenador o información de aplicaciones de otras maquinas de la red local, necesitamos filtrar la captura hasta poder ver información relevante:

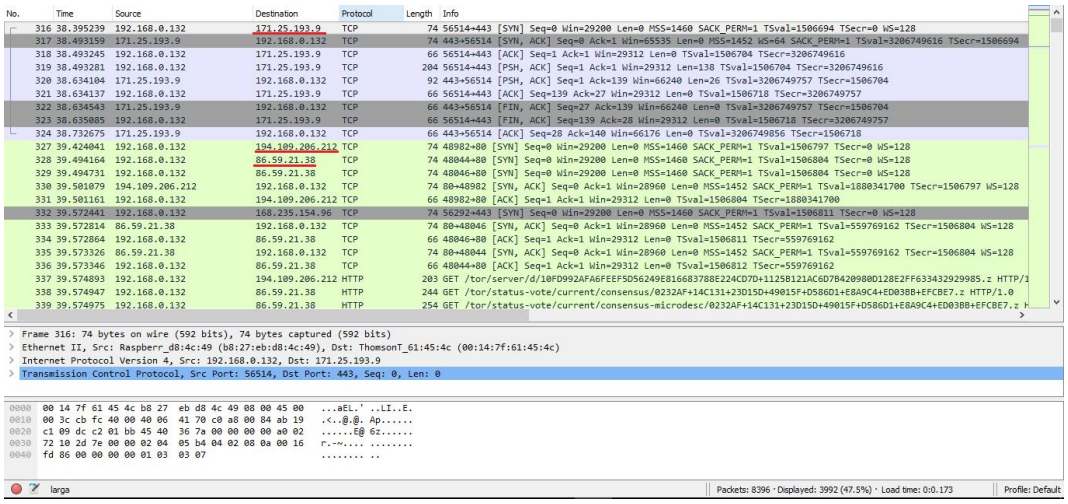

Figura 4.10: Primeras tramas intercambiadas por nuestro relay.

Como vemos, lo primero que hace nuestro relay siempre, es lanzar un saludo a tres relays. No siempre serán los mismos, pero si que guardan una cosa en común y es que todos ellos son *Directory*

Authorities, lo que significa que son unos nodos especiales de la Tor Network donde se alberga información de relevancia sobre el estado de la red, como por ejemplo alertas de sobrecarga o de utilización de relays, y de los relays operativos.

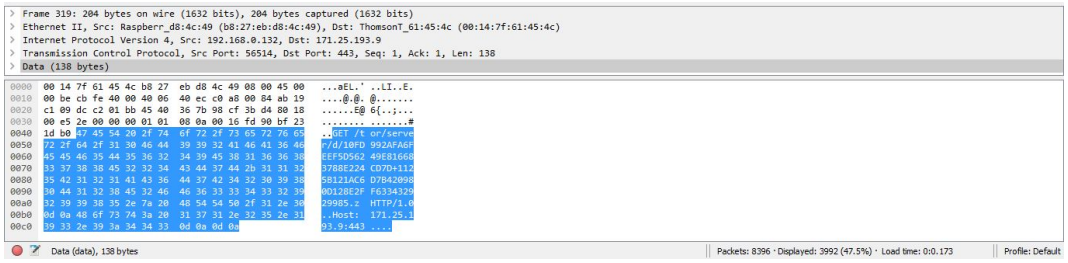

Figura 4.11: Detalle del campo de datos de la solicitud.

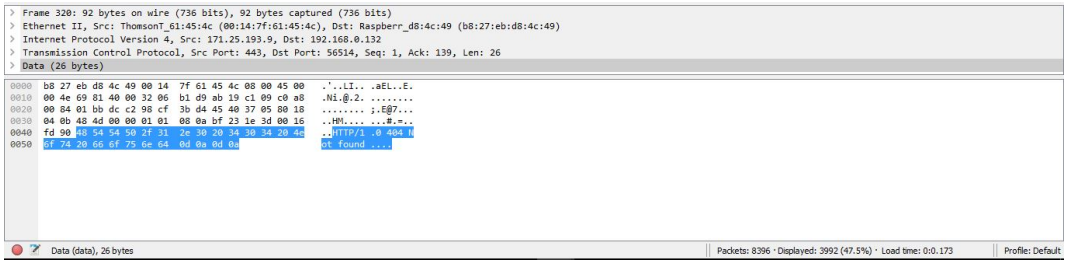

Figura 4.12: Detalle del campo de datos de la contestación.

En este caso, por ejemplo, vemos que para los relays *maatuska* (171.25.193.9) y *dizum* (194.109.206.212) nuestro relay le solicita un archivo comprimido a lo que le responden que no hay sido encontrado.

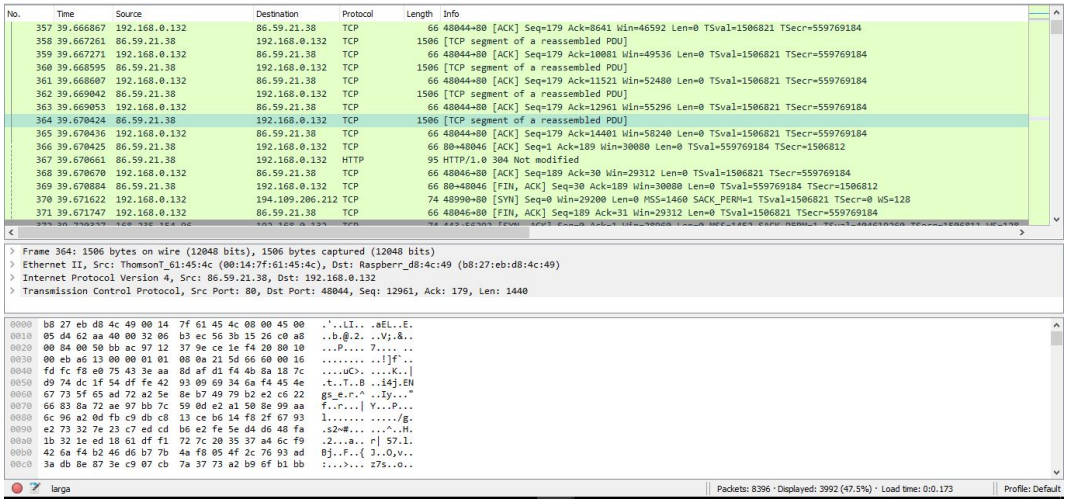

Figura 4.13: Comunicación con el relay de NickName tor26.

Sin embargo, el relay *tor26* (86.59.21.38) si que dispone de ese archivo con información del estado de al red, y una vez cifran su comunicación nos es imposible sacar algo en claro.

En el momento del inicio del relay, no solo se crean conexiones con *Directory Authorities*, si no que también tenemos conexiones entrantes y salientes a nuestro relay con otros nodos de la red con los que conformaremos alguna ruta segura para un cliente, y también se crean unos *ciurcuits* o rutas seguras por si acaso estuviésemos utilizando la aplicación tor como clientes y no como un nodo de la red:

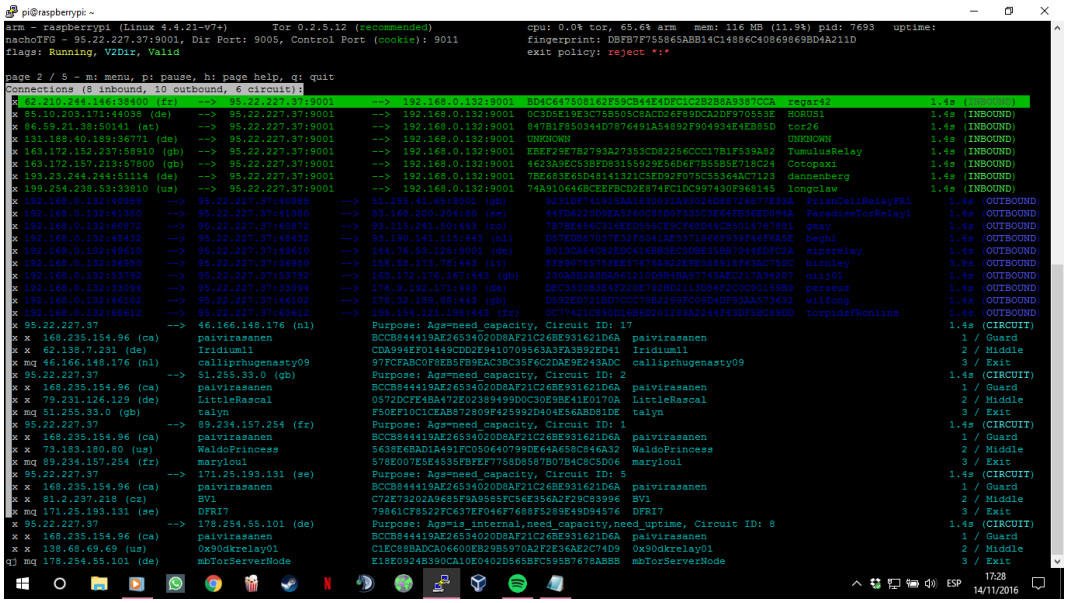

Figura 4.14: Primeras tramas intercambiadas por nuestro relay.

Monitorización y estadísticas

Para la supervisión y monitorización de un relay, hablaremos de dos aspectos, la monitorización en tiempo real que realizamos a través de la herramienta Arm; y las estadísticas históricas aportadas por aplicaciones web :

- **Tiempo real** Como ya hemos visto al principio de este capítulo, la aplicación Arm nos nutre con información instantánea sobre nuestro relay, tanto ancho de banda de descarga y carga como conexiones establecidas, con una capacidad de refresco de hasta un segundo.

Figura 4.15: Ancho de banda de descarga (izquierda) y de carga (derecha) medido en bps

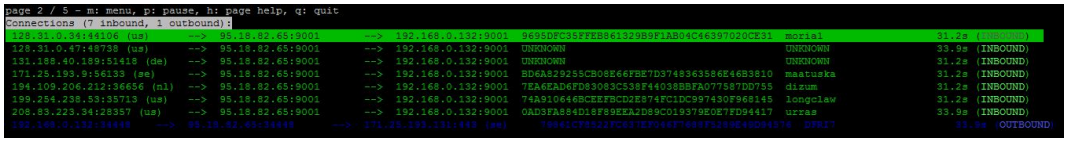

Figura 4.16: Conexiones establecidas en el mismo momento de la captura anterior

Además gracias a esta aplicación podremos consultar rápidamente el log donde se nos notificará de cualquier improvisto, como puede ser que haya un retraso en el reloj de la maquina, los datos de configuración no estén actualizados, etc.

- **Histórico**

  Por otro lado tenemos una serie de aplicaciones web en las que podemos buscar cualquier relay de la Tor Network ya sea por su NickName, por su dirección IP pública o por su fingerprint para conocer gran cantidad de información que es valiosa tanto para el poseedor del relay, como para los clientes que se conecten a él. Describiremos un poco qué podemos averiguar sobre nuestro relay en las tres aplicaciones que hemos utilizado:

  - **Atlas**

    Es una aplicación web open source, concretamente es la más importante y la que mas información arroja.

    Se organiza en tres columnas, configuración, propiedades y estado actual.

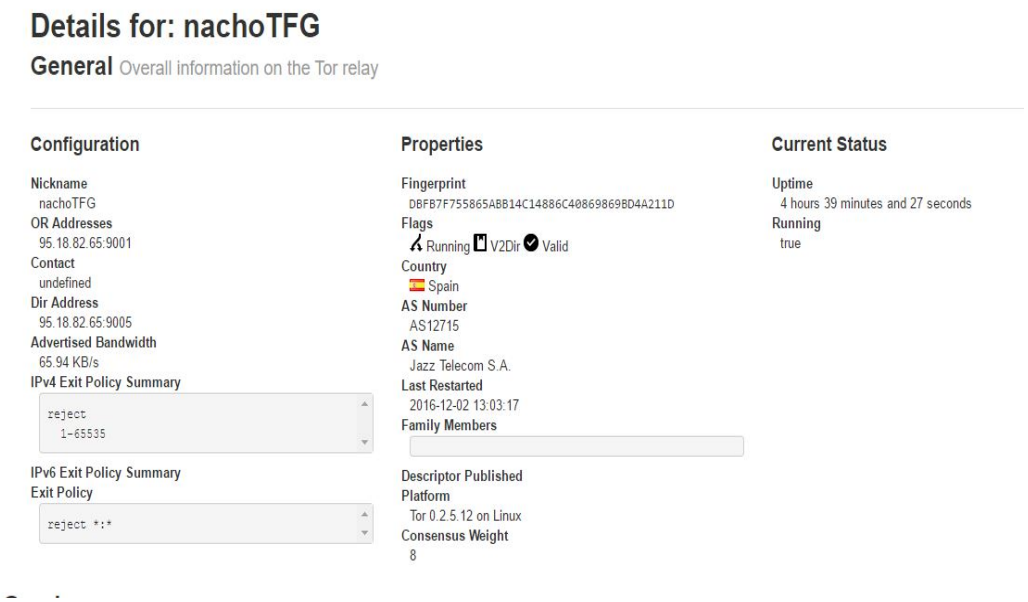

Figura 4.17: Información sobre nuestro relay hallada en Atlas.

En la primera de ellas encontraremos la información pública que caracteriza al relay, como es su NickName, dirección IP pública, ancho de banda anunciado y políticas de salida (en nuestro caso tenemos todos los puertos cerrados).

En la segunda columna podemos encontrar información mas técnica como es nuestro ISP, nuestro peso de consenso del que ya hablamos en apartados anteriores, y las flags que tiene activados. En el momento de la captura, nuestro relay solo tenia activas las flags de 'funcionando', implementación del protocolo de directorio V2 y 'validado'.

Por último tenemos el tiempo que lleva el relay en línea y si está funcionando o no en ese instante.

Atlas ofrece cerca de diez gráficas con las que podemos seguir un histórico de los bits escritos y leídos por segundo, y de la probabilidad de que nuestro relay actúe como un Entry Guard, un Middle Relay, un Bridge o un Exit Relay, pero las comentaremos con más detalle en la siguiente aplicación.

- **GLOBE**

  GLOBE es la aplicación web más gráfica e intuitiva de la que podemos disponer. Está basada en Atlas y usa la *Onionoo-API* para recabar información.

  El hecho de que se base en atlas hace que sean prácticamente hermanos gemelos, lo único que esta segunda aplicación presenta unas gráficas con las que es más fácil interactuar y sacar información valiosa.

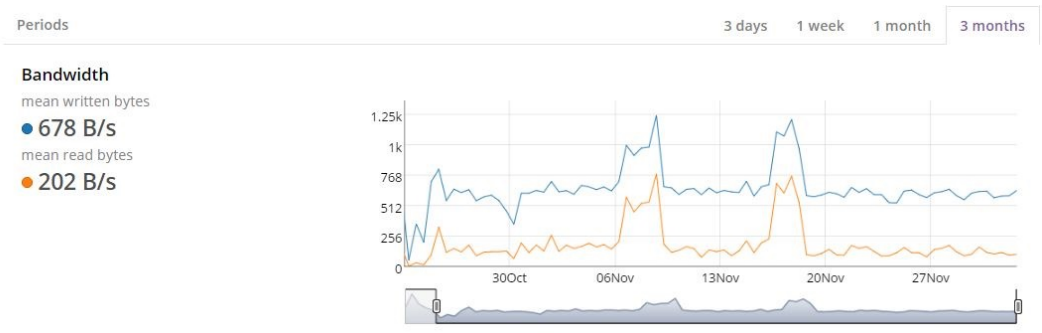

Figura 4.18: Histórico de BW hallado en GLOBE.

Disponemos en primer lugar de un histórico de la media de Bytes escritos y leídos en periodos de tiempo que van desde los 3 días hasta los 5 años, en el caso de que el relay

llevara el suficiente tiempo en funcionamiento.

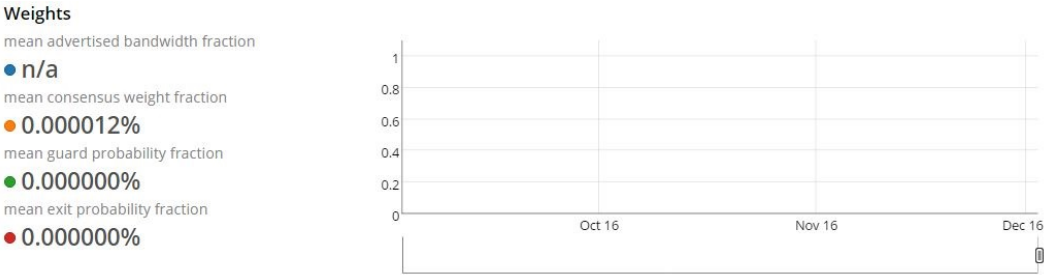

Figura 4.19: Estadístas sobre el Consensus Weight hallado en GLOBE.

En la segunda gráfica encontramos los datos de la fracción de peso de consenso, y probabilidades de ser escogidos como Entry Guard o Exit Relay. Como vemos, nuestro relay no guarda las cualidades para ser Entry Guard, ni está configurado para ser Exit Relay, luego hay una probabilidad del 0 % de que desempeñe esos papeles.

Figura 4.20: Información sobre el tiempo en funcionamiento hallado en GLOBE

.

Por último, una gráfica que nos avisaría si nuestro relay ha sufrido una desconexión y se encuentra inoperativo y cuánto tiempo ha estado caído de la Tor Network.

- **Tor Network Status**
  Esta última aplicación web no es tan útil en lo que a información sobre un relay concreto se refiere, ya que ofrece una breve información general sobre el relay, y la representación gráfica sobre el histórico de ancho de banda escrito y leído solo arroja información sobre las últimas seis horas.

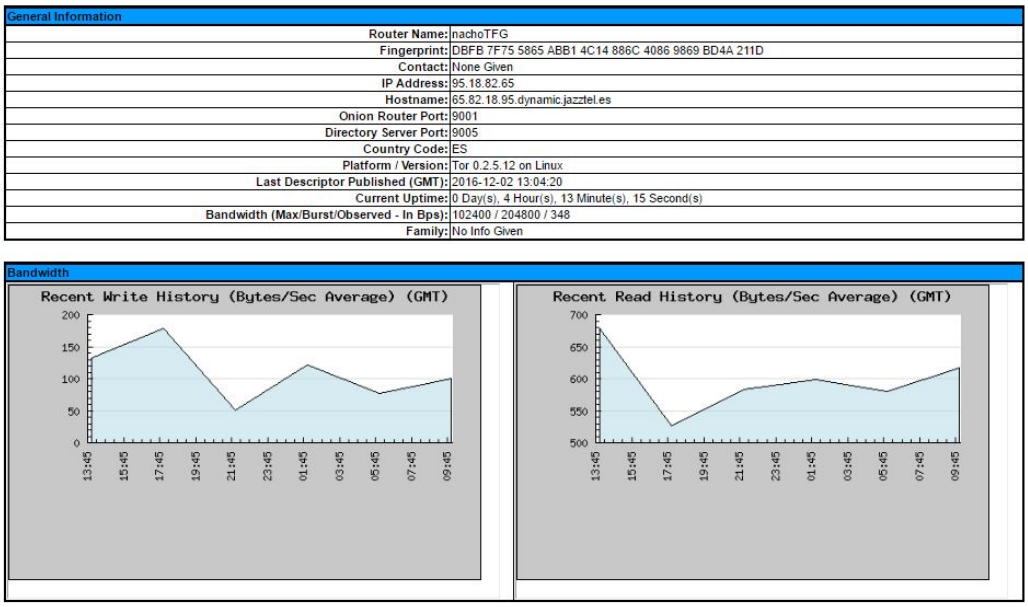

Figura 4.21: Información acerca de nuestro relay hallada en Tor Network Status de Joseph B. Kowalski

Sin embargo, esta aplicación destaca porque es capaz de indexar todos y cada uno de los relays que se encuentran operativos en la Tor Network, mientras que las otras dos solo ofrecían un top 10. Disponemos de un listado en el que podemos ordenar los relays en función de su ancho de banda ofrecido, los que más tiempo llevan operativos, los que primero se conectaron a la red, etc.

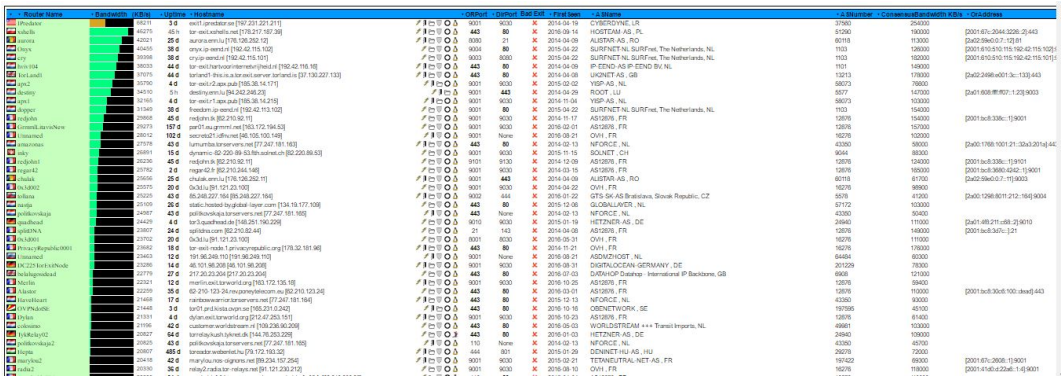

Figura 4.22: Listado de todos los relays de la Tor Network

.

Capítulo 5

Conclusiones y líneas futuras

Este proyecto ha descrito las especificaciones técnicas del 'onion routing' desde el lado del cliente, así como desde el punto de vista de un nodo de la red. En el grueso del trabajo, hemos implementado un relay en una red local, el cual,durante el desarrollo del proyecto, ha facilitado el acceso a un gran número de clientes, escribiendo una media de 670 Bps durante los tres meses que lleva en funcionamiento.

A partir de este proyecto, en el que nos hemos limitado a implementar un Middle Relay, podría seguirse esta línea para desarrollar un Exit Relay. Para el desarrollo, podría seguirse un camino distinto, ya que nosotros hemos utilizado la plataforma Raspberry Pi para programar con Debian el relay, pero cuando hablamos de un Exit Relay, necesitaríamos más ancho de banda, más potencia de computación y más seguridad [21–23].

En la gran mayoría de casos en los que alguien se ha lanzado a implementar un nodo de salida, no lo ha hecho en su red doméstica, si no que contratan un servidor virtual a un ISP determinado que les da la garantía de que tienen libertad para albergar un Exit Relay (existen muchos ISPs que son recelosos de ello).
Por ejemplo, en el host *KeyWeb*, el modelo idóneo sería el RVS M14 que corre con un procesador Intel Core i5, 1GB de RAM, 100GB de SSD y 'ancho de banda ilimitado'. Estas características harían del nodo un relay de bastante peso en la red, y sería posible obtener mucha información valiosa sobre cómo se resuelven los pseudo-dominios *.onion*, por ejemplo.

Una vez implementado el Exit Relay, se podría poner en prueba su vulnerabilidad y, por extensión, la de la Tor Network. Un ataque de sencilla implementación sería el conocido como *The Sniper Attack*, que consiste en que un atacante es capaz de inutilizar un relay de salida o un grupo de ellos desde la distancia, como si se tratara de un francotirador acostado en una ladera en la distancia [49]. Durante los experimentos desarrollados por el Laboratorio de Investigación Naval de los Estados Unidas de América, en Washington DC, se ha probado que el ataque puede reducir la memoria del relay víctima con una tasa de 2187 KiB/s con un consumo de nacho de banda de subida de tan solo 92 KiB/s. Es decir, podría inutilizar los 20 nodos de salida más importantes de la red en tan solo 29 minutos, reduciendo el ancho de banda total de la Tor Network en un 35 %.

El ataque en si, trabaja con el control de flujo obligando a un relay a almacenar una cantidad arbitraria de datos en la cola de aplicación. El atacante solicita al Exit Relay que descargue un gran archivo a través del circuito, y seguidamente envía SENDME cells de forma continuada. Las SENDME cells indican al nodo de salida que aumente su ventana de congestión y que continúe descargando datos y metiéndolos por el circuito. Este proceso puede hacerse en paralelo utilizando diferentes circuitos con el mismo nodo de entrada.
Otra línea de investigación futura podría ser trabajar sobre la red Riffle, del inglés barajar, desarrollada por el MIT de Massachusetts, en lugar de sobre la red Tor [62]. Se dice que esta nueva red se basa en una nueva técnica extremadamente eficiente, en la que si Alice, Bob y Carol desean comunicarse mediante dicha red, el primer servidor recibiría los mensajes en un orden A,B,C pero los reenviaría al siguiente servidor con una permutación, C,B,A por ejemplo. El segundo servidor haría lo mismo antes de mandarlo al tercero, y así sucesivamente.
Por ahora, la red Riffle es solo un proyecto, pero se dice que será 10 veces más veloz que la Tor Network

Esto son solo un par de ideas de cómo continuar este proyecto, aunque pueden surgir gran cantidad de ideas a raíz de nuestro trabajo [8].

Bibliografía

[1] Tor Metrics developed by Tor Project. 'Can you trust Tor's entry nodes?'. https://metrics.torproject.org/, 16 de Octubre de 2016.

[2] GLOBE. 'Details for janschejbalscalewy4'. http://globe.rndm.de/#/relay/AD253B49E303C6AB1E048B014392AC569E8A7DAE, 4 de Diciembre de 2016.

[3] Wikipedia. 'Tor (anonymity network) — Wikipedia, The Free Encyclopedia'. https://en.wikipedia.org/wiki/Tor_(anonymity_network), 20 de Diciembre de 2016.

[4] Tor Metrics. 'Top-10 countries by possible censorship events.'. https://metrics.torproject.org/userstats-censorship-events.html, 30 de Diciembre de 2016.

[5] Manuel Castells. *'La Sociedad Red'*. 2006.

[6] 'Internet live stats'. http://www.internetlivestats.com, 10 de Diciembre de 2016.

[7] Michael K Bergman. 'White paper: the deep web: surfacing hidden value'. *Journal of electronic publishing*, 7(1), 2001.

[8] Wikipedia. '"Deep web — Wikipedia, The Free Encyclopedia'. https://en.wikipedia.org/wiki/Deep_web, 16 de Diciembre de 2016.

[9] V Ciancaglini, M Balduzzi, R McArdle, and M Rösler. 'Below the surface: Exploring the Deep Web'. *Trend Micro Incorporated. As of*, 12, 2016.

[10] Jayant Madhavan, David Ko, Łucja Kot, Vignesh Ganapathy, Alex Rasmussen, and Alon Halevy. 'Google's deep web crawl'. *Proceedings of the VLDB Endowment*, 1(2):1241–1252, 2008.

[11] Sriram Raghavan and Hector Garcia-Molina. 'Crawling the hidden web'. 2000.

[12] Luciano Barbosa, Hoa Nguyen, Thanh Nguyen, Ramesh Pinnamaneni, and Juliana Freire. 'Deeppeep: A form search engine'.

[13] Michael L Nelson, Herbert Van de Sompel, Xiaoming Liu, Terry L Harrison, and Nathan McFarland. 'mod_oai: an apache module for metadata harvesting'. In *International Conference on Theory and Practice of Digital Libraries*, pages 509–510. Springer, 2005.

[14] Tor StackExchange. 'Tor circuit display in Tor Browser'. http://tor.stackexchange.com/questions/9516/tor-circuit-display-in-tor-browser-how-can-it, 8 de Noviembre de 2016.

[15] Tor StackExchange. 'How does a Tor client pick Tor nodes for circuit creation?'. http://tor.stackexchange.com/questions/113/how-does-a-tor-client-pick-tor-nodes-for-circuit-creation, 10 de Noviembre de 2016.

[16] Tariq Elahi, Kevin Bauer, Mashael AlSabah, Roger Dingledine, and Ian Goldberg. 'Changing of the guards: A framework for understanding and improving etry guard selection in Tor'. In *Proceedings of the 2012 ACM workshop on Privacy in the electronic society*, pages 43–54. ACM, 2012.

[17] Roger Dingledine, Nicholas Hopper, George Kadianakis, and Nick Mathewson. 'One fast guard for life (or 9 months)'. In *7th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs 2014)*, 2014.

[18] Tor StackExchange. 'In Tor Browser, I connect to the exact same entry node all the time.'. `http://tor.stackexchange.com/questions/9333/in-tor-browser-i-connect-to-the-exact-same-entry-node-all-the-time-unable-to-c`, 10 de Noviembre de 2016.

[19] Mark Stockley en Nakedsecurity. 'Can you trust Tor's entry nodes?'. `https://nakedsecurity.sophos.com/2015/08/03/can-you-trust-tors-entry-guards/`, 11 de Noviembre de 2016.

[20] Mark Stockley en Nakedsecurity. 'Can you trust Tor's exit nodes?'. `https://nakedsecurity.sophos.com/2015/06/25/can-you-trust-tors-exit-nodes/`, 11 de Noviembre de 2016.

[21] Tor Project blog. 'Five years as an Exit node operator'. `https://blog.torproject.org/blog/five-years-exit-node-operator`, 9 de Noviembre de 2016.

[22] Tor Project. 'Tor Exit guidelines'. `https://trac.torproject.org/projects/tor/wiki//doc/TorExitGuidelines`, 9 de Noviembre de 2016.

[23] Tor Project blog. 'Tips for running an Exit node'. `https://blog.torproject.org/running-exit-node`, 9 de Noviembre de 2016.

[24] Tor Project FAQ. 'What are Entry Guards?'. `https://www.torproject.org/docs/faq#EntryGuards`, 11 de Noviembre de 2016.

[25] K Zetter. 'Tor torches online tracking'. *Wired News*, 2005.

[26] The Economist. 'Bitcoin: Monetarists Anonymous'. `http://www.economist.com/node/21563752`, 29 de Sctubre de 2012, consultado el 27 de Diciembre de 2016.

[27] Tor Project. 'Users of Tor'. `https://www.torproject.org/about/torusers.html.en`, 6 de Diciembre de 2016.

[28] Tor Project Blog. 'Recent events in Egypt'. `https://blog.torproject.org/blog/recent-events-egypt`, 6 de Diciembre de 2016.

[29] Wikipedia. 'Internet censorship in the Arab Spring'. `https://en.wikipedia.org/wiki/Internet_censorship_in_the_Arab_Spring`, 6 de Diciembre de 2016.

[30] Gizmodo. 'Internet y censura: ¿qué está ocurriendo realmente en Venezuela?'. `http://es.gizmodo.com/el-laberinto-de-las-comunicaciones-en-venezuela-1523910875`, 6 de Diciembre de 2016.

[31] Philipp Winter and Stefan Lindskog. 'How the Great Firewall of China is blocking Tor.'. In *FOCI*, 2012.

[32] David Leigh, Luke Harding, and Charles Arthur. *'WikiLeaks: Inside Julian Assange's war on secrecy'*. PublicAffairs, 2011.

[33] Ville-Valtteri Immonen. 'Alice in Onion Land: On information security of Tor'. *Computer Science*, 2016.

[34] Juha Salo. 'Recent attacks on Tor'. *Aalto University*, 2010.

[35] Joan Feigenbaum, Aaron Johnson, and Paul Syverson. 'Probabilistic analysis of onion routing in a black-box model'. *ACM Transactions on Information and System Security (TISSEC)*, 15(3):14, 2012.

[36] Vasilis Pappas, Elias Athanasopoulos, Sotiris Ioannidis, and Evangelos P Markatos. 'Compromising anonymity using packet spinning'. In *International Conference on Information Security*, pages 161–174. Springer, 2008.

[37] Sambuddho Chakravarty, Marco V Barbera, Georgios Portokalidis, Michalis Polychronakis, and Angelos D Keromytis. 'On the effectiveness of traffic analysis against anonymity networks using flow records'. In *International Conference on Passive and Active Network Measurement*, pages 247–257. Springer, 2014.

[38] Steven J Murdoch and George Danezis. 'Low-cost traffic analysis of Tor'. In *2005 IEEE Symposium on Security and Privacy (S&P'05)*, pages 183–195. IEEE, 2005.

[39] Zhen Ling, Junzhou Luo, Wei Yu, Xinwen Fu, Dong Xuan, and Weijia Jia. 'A new cell counter based attack against tor'. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 578–589. ACM, 2009.

[40] Timothy G Abbott, Katherine J Lai, Michael R Lieberman, and Eric C Price. 'Browser-based attacks on Tor'. In *International Workshop on Privacy Enhancing Technologies*, pages 184–199. Springer, 2007.

[41] Nathan S Evans, Roger Dingledine, and Christian Grothoff. 'A practical congestion attack on Tor using long paths.'. In *USENIX Security Symposium*, pages 33–50, 2009.

[42] Nicholas Hopper, Eugene Y Vasserman, and Eric Chan-Tin. 'How much anonymity does network latency leak?'. *ACM Transactions on Information and System Security (TISSEC)*, 13(2):13, 2010.

[43] Matthew K Wright, Micah Adler, Brian Neil Levine, and Clay Shields. 'Passive-logging attacks against anonymous communications systems'. *ACM Transactions on Information and System Security (TISSEC)*, 11(2):3, 2008.

[44] Yang Zhang. 'Effective attacks in the Tor authentication protocol'. In *Network and System Security, 2009. NSS'09. Third International Conference on*, pages 81–86. IEEE, 2009.

[45] Stevens Le Blond, Pere Manils, Chaabane Abdelberi, Mohamed Ali Dali Kaafar, Claude Castelluccia, Arnaud Legout, and Walid Dabbous. 'One bad apple spoils the bunch: exploiting P2P applications to trace and profile Tor users'. *arXiv preprint arXiv:1103.1518*, 2011.

[46] Xiaogang Wang, Junzhou Luo, Ming Yang, and Zhen Ling. 'A potential HTTP-based application-level attack against Tor'. *Future Generation Computer Systems*, 27(1):67–77, 2011.

[47] Marco Valerio Barbera, Vasileios P Kemerlis, Vasilis Pappas, and Angelos D Keromytis. 'CellFlood: Attacking Tor onion routers on the cheap'. In *European Symposium on Research in Computer Security*, pages 664–681. Springer, 2013.

[48] Bruce Schneier. 'Attacking Tor: how the NSA targets users' online anonymity'. *The Guardian*, 4, 2013.

[49] Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann. 'The sniper attack: Anonymously deanonymizing and disabling the Tor network'. Technical report, DTIC Document, 2014.

[50] Wired. 'Tor attack tries to decloak anonymous users'. `http://www.wired.co.uk/news/archive/2014-07/31/tor-security-decloaking-attack`, 17 de Noviembre de 2016.

[51] Tech Times. 'Tor suffers traffic confirmation attacks. Say goodbye to anonymity on the web'. `http://www.techtimes.com/articles/11711/20140802/tor-suffers-traffic-confirmation-attacks-say-goodbye-to-anonymity-on-the-web.htm`, 17 de Noviembre de 2016.

[52] Tor Project blog. 'Tor security advisory: relay early"traffic confirmation attack'. `https://blog.torproject.org/blog/tor-security-advisory-relay-early-traffic-confirmation-attack`, 17 de Noviembre de 2016.

[53] Jose Carlos Norte. 'Advanced Tor browser fingerprinting'. Mar-2016.

[54] Tor's specification protocol Torspec. 'Application connections and stream management - Opening streams and transferring data'. `https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt?id=f9e111ead769d48441d99b52039ef0ccbd3f2c62#n1311`, 18 de Noviembre de 2016.

[55] Cave's tinker pit. 'Raspberry Pi as Tor Middle Relay'. `http://cavebeat.blogspot.com.es/2012/11/raspberry-pi-tor-middle-relay.html`, 29 de Septiembre de 2016.

[56] Damian Jhonson. 'Arm'. `https://www.atagar.com/arm/`, 2 de Noviembre de 2016.

[57] Tor's specification protocol Torspec. 'Torspec Flags'. `https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt`, 18 de Noviembre de 2016.

[58] Tor Project Damian Johnson. 'Circuit purposes'. `https://lists.torproject.org/pipermail/tor-relays/2014-June/004834.html`, 9 de Octubre de 2016.

[59] Stem Documents. 'CircBuildFlag from Controller'. `https://stem.torproject.org/api/control.html#stem.CircBuildFlag`, 9 de Octubre de 2016.

[60] Thomas E. Dickey en Invisible Island. 'Line-drawing characters come out as x's and q's'. `http://invisible-island.net/ncurses/ncurses.faq.html#no_line_drawing`, 24 de Noviembre de 2016.

[61] Tor Project blog. 'The lifecycle of a new relay'. `https://blog.torproject.org/blog/lifecycle-of-a-new-relay`, 6 de Diciembre de 2016.

[62] Gizmodo. 'Riffle, la red anónima del MIT que soluciona los problemas de seguridad de Tor'. `http://es.gizmodo.com/riffle-la-red-anonima-del-mit-que-soluciona-los-proble-1783517629`, 14 de Diciembre de 2016.