

4.2 Arquitectura

Las secciones anteriores de este capítulo han explicado el uso de Tor desde el punto de vista de un usuario, sin embargo se ha hablado someramente sobre el funcionamiento interno y la arquitectura de esta potente solución para la privacidad y el anonimato de cualquier usuario en Internet y en esta sección, se intentará explicar sus componentes e integración de cada uno de ellos. Antes de continuar y del mismo modo que ocurre con otras redes anónimas explicadas previamente, es importante tener buenos conocimientos sobre redes y el modelo OSI, así como los conceptos fundamentales de criptografía.

4.2.1 Repetidores

Se trata de una instancia de Tor que se encuentra en condiciones de aceptar y replicar el tráfico proveniente de otra instancia de Tor. De esta forma, los repetidores pueden verse como servidores proxy transparentes que enrutan tráfico al siguiente salto de un circuito. Los repetidores son el elemento fundamental de la red, ya que permiten a los usuarios navegar por Internet o la web profunda utilizando el protocolo de Tor como plataforma de anonimato. Cualquier cliente puede actuar como un repetidor en un momento dado, solamente es necesario establecer las opciones de configuración adecuadas en el fichero "torrc" utilizado para iniciar la instancia de Tor, además también es posible especificar un tope máximo de ancho de banda que el repetidor puede aportar a la red. Todos los voluntarios que desean convertir su instancia de Tor en un repetidor, deben especificar uno de dos posibles tipos de repetidores: internos o externos.

Repetidores internos

Se trata de repetidores que únicamente enrutan tráfico al interior de la red y dadas sus características, no tienen la capacidad de acceder de forma directa a los paquetes de datos que viajan entre el cliente y el destino, únicamente se encargan de suprimir la capa de cifrado correspondiente a su propia clave privada y acceder a la información necesaria para enrutar el tráfico hacia el siguiente salto del circuito.

Repetidores externos

Se trata de repetidores que pueden enrutar tráfico al exterior de la red, es decir, que pueden enviar los paquetes de datos de los clientes al correspondiente destino. Dadas sus características, tienen la capacidad de acceder de forma directa a la información que se incluye en los paquetes de datos que viajan entre el cliente y el destino, ya que los repetidores externos, o también conocidos como nodos de salida en un circuito, se encargan de suprimir la última capa de cifrado correspondiente al protocolo de Tor y de esta forma, no solamente tienen la capacidad de descubrir cuál es el destino final de dichos paquetes, sino que también tienen la posibilidad de leer sus contenidos.

Los repetidores en Tor permiten a los clientes componer circuitos, los cuales funcionan como una cadena de servidores proxy que se encargan de recibir los paquetes de datos cifrados por parte de los clientes y enrutarlos a su correspondiente destino. Los circuitos en Tor, a diferencia de los túneles en I2P son bidireccionales, esto quiere decir que un circuito puede ser utilizado tanto para enviar como para recibir paquetes de datos.



Por otro lado, los repetidores publican información sobre sus características principales en la red de Tor por medio de los documentos conocidos como “*Descriptores*”. Esta información es posteriormente utilizada por los clientes para saber de qué forma se pueden comunicar con el repetidor y solicitar información tan importante como la clave pública del repetidor para cifrar paquetes de datos que viajarán por un circuito.

Para indicarle a una instancia de Tor que debe funcionar como un repetidor, se debe editar el fichero de configuración maestro “*torrc*” y posteriormente establecer como mínimo, las siguientes opciones de configuración:

```
ORPort 443  
Exitpolicy reject *:  
Nickname Adastrator  
ContactInfo adastrata at thehackerway dot com
```

En este caso, el puerto indicado en la propiedad “*ORPort*” debe ser accesible a otras máquinas en Internet, ya que será utilizado por los clientes de Tor para comunicarse con dicha instancia. Por otro lado, la propiedad “*Exitpolicy*” es de vital importancia, ya que es la que permite indicar si el repetidor es interno o externo. En este caso concreto, la política de salida de paquetes de datos es completamente restrictiva, lo que les indica a las autoridades de directorio y a los clientes que el repetidor solamente podrá enrutar tráfico al interior de la red y bajo ningún concepto permitirá el reenvío de paquetes hacia Internet. En el caso de configurar un repetidor externo, se deben indicar políticas de salida mucho más permisivas, como por ejemplo las que se indican a continuación:

```
ExitPolicy accept *:80, accept *:443, accept *:110, accept *:143, accept *:993,  
accept *:995
```

En este caso se indica explícitamente que los paquetes de datos cuyo puerto de destino sea cualquiera de los indicados, se admite el reenvío del paquete y de esta forma, el repetidor actuará como un nodo de salida para los circuitos que los clientes construyen.

Es importante tener en cuenta que los repetidores de salida enrutan información hacia otros destinos en plano, tal como se verá con mucho más detalle en una próxima sección y que además, no tienen un control sobre los contenidos que se envían hacia Internet ni sus correspondientes destinos. Esta situación puede ser especialmente problemática para el administrador del repetidor cuando otros usuarios en la red crean circuitos utilizando su repetidor como punto de salida y utilizan dicho circuito para realizar actividades ilegales.

El problema para el administrador del repetidor es que evidentemente desconoce el origen de los paquetes y además, es posible que una autoridad competente detecte dicho tráfico de datos y lo que podrán obtener es el punto de salida del circuito, es decir, la dirección IP de la persona del repetidor.

Evidentemente las consecuencias pueden ser bastante desagradables para el administrador ya que para las autoridades, los paquetes de datos o el destino de los mismos tienen temática ilegal o maliciosa y el origen de los mismos ha sido el repetidor de salida en cuestión. Dicho esto, si se establece un repetidor de salida es muy importante afinar adecuadamente las políticas de aceptación y rechazo.



Cuando una instancia de Tor utiliza las opciones de configuración anteriores, lo primero que realiza la instancia son las comprobaciones de conectividad, que tal como se ha comentado anteriormente, su éxito o fracaso depende de si el puerto especificado en la propiedad “*ORPort*” es accesible desde Internet. Estas comprobaciones suelen ser muy rápidas, aunque en algunos casos pueden tardar hasta 20 minutos. Por otro lado, cuando una instancia se registra por primera vez en la red, las autoridades de directorio deben votar y aprobar la participación de dicho repetidor en la red, tal como se verá en una próxima sección del presente capítulo, este proceso puede tardar entre 45 y 60 minutos, dependiendo de la hora en la que se ha generado el último consenso por parte de las autoridades de directorio. En cualquier caso, es posible verificar el estado de un repetidor gracias al proyecto “atlas”, el cual se encuentra ubicado en la siguiente url: <https://atlas.torproject.org/>

4.2.2 Descriptores

Los descriptores son documentos que almacenan información sobre los repetidores que conforman la red de Tor. Dichos documentos se encuentran disponibles de forma pública y cualquier usuario en Internet tiene la posibilidad de descargarlos, para hacerlo basta con ejecutar una petición HTTP contra las autoridades de directorio o una cache de directorio para obtener este tipo de documentos. En los últimos años el software de Tor y evidentemente la red han incorporado cambios importantes y dichas modificaciones han dado lugar a diferentes tipos de descriptores, los cuales se encuentran categorizados según la información que almacenan. A continuación se listan los más habituales.

Server Descriptor

Se trata del descriptor principal que publican los repetidores en las autoridades de directorio. Estos documentos contienen toda la información sobre el repetidor, incluyen sus políticas de salida, detalles sobre el ancho de banda aportado y consumido, dirección IP, puerto “OR”, sistema operativo, entre otros detalles interesantes. En versiones antiguas de Tor, los clientes descargaban directamente este descriptor para cada uno de los repetidores que componen la red, pero debido a la cantidad de información que puede contener, en versiones más recientes del software de Tor, los clientes ahora se encargan de descargar una versión más compacta de este fichero, lo que ha dado lugar a un tipo de descriptor conocido como “*MicroDescriptor*”.

ExtraInfo Descriptor

Un descriptor de este tipo incluye información sobre el repetidor que no es requerida por los clientes para su correcto funcionamiento, con lo cual, los clientes de Tor no lo descargan por defecto. Estos descriptores son publicados por todos los repetidores de forma automática, del mismo modo que ocurre con los descriptores del tipo “*Server Descriptor*”, sin embargo, tal como se ha indicado anteriormente, no son descargados por los clientes y es necesario realizar el proceso de obtención de dichos ficheros de forma explícita en el caso de que se quiera acceder a la información que contienen. Para que una instancia de Tor pueda descargar este tipo de contenidos, es necesario indicar la siguiente opción de configuración en el fichero “*torrc*” de la instancia.

`DownloadExtraInfo 1`

No obstante, hay que tener en cuenta que las instancias de Tor, a la fecha de redactar este documento, no utilizan para nada la información que se almacena en dichos ficheros, sin embargo puede ser



lo primero que realiza
mentado anteriormente,
“Tor” es accesible desde
os pueden tardar hasta
la red, las autoridades
red, tal como se verá
entre 45 y 60 minutos,
nte de las autoridades
or gracias al proyecto
org/

dores que conforman
a y cualquier usuario
r una petición HTTP
tipo de documentos.
cambios importantes
cuales se encuentran
s más habituales.

de directorio. Estos
as de salida, detalles
ma operativo, entre
n directamente este
do a la cantidad de
r, los clientes ahora
lado lugar a un tipo

rida por los clientes
por defecto. Estos
el mismo modo que
mo se ha indicado
ceso de obtención
a información que
idos, es necesario

ar este documento,
mbargo puede ser

interesante para un cliente obtener la información que se encuentra disponible en dichos ficheros para tener todos los detalles sobre los repetidores que se encuentran disponibles en la red.

Micro Descriptor

Los “*Server Descriptors*” son documentos que contienen mucha más información de la que es necesaria para que un cliente pueda componer sus circuitos, por este motivo una de las mejoras que se ha aplicado en versiones recientes de Tor, son los “*Micro Descriptors*” los cuales son versiones minimalistas de los “*Server Descriptors*” con únicamente la información necesaria para crear circuitos de Tor. Los clientes descargan este tipo de descriptores por defecto, ya que de esta forma se ahorra ancho de banda al descargar documentos mucho más compactos y con la información que es requerida para el correcto funcionamiento de las instancias cliente. Si por cualquier motivo resulta interesante obtener la información que se almacena en los “*Server Descriptors*”, el cliente tiene la posibilidad de establecer en el fichero de configuración “*torrc*” la siguiente propiedad.

```
UseMicrodescriptors 0
```

Utilizando “*UseMicrodescriptors*” se le puede indicar a la instancia de Tor que debe modificar su comportamiento por defecto y descargar los “*Server Descriptors*” de los repetidores.

Network Status Document

Tal como se verá más adelante en el presente capítulo, la red de Tor depende completamente del correcto funcionamiento de las autoridades de directorio, las cuales se encargan de generar un fichero conocido como “*Network Status*”, que es el resultado del proceso de votación de todas las autoridades y contiene una serie de registros conocidos como “*Router Status Entry*”. Este descriptor también suele ser encontrado en la terminología de Tor simplemente como “consenso” y es un documento de vital importancia no solamente para las autoridades de directorio, sino también para las caches de directorio y los clientes.

Router Status Entry

Los documentos de “*Network Status*” están compuestos por múltiples entradas de “*Router Status Entries*” y tal como su nombre lo indica, incluye información sobre cada uno de los repetidores de la red, sin embargo, dicha información es suministrada por las autoridades de directorio, las cuales establecen, entre otras cosas, flags y heurísticas para la selección de repetidores por parte de los clientes a la hora de componer circuitos.

Hidden Service Descriptor

Se trata de un documento muy importante para el correcto funcionamiento de los servicios ocultos en la web profunda de Tor y que es firmado y publicado por el servicio oculto propiamente dicho. Estos documentos son publicados a los servidores con la flag “*HSDir*”, los cuales componen una tabla distribuida del tipo hash (*Distributed Hash Table*). El contenido de estos descriptores incluye la información que los clientes necesitan para comunicarse con el servicio oculto de forma anónima. Los contenidos de este fichero y su importancia en el protocolo de servicios ocultos se verán con mucho más detalle en la sección correspondiente a la creación y configuración de servicios ocultos que se encuentra disponible más adelante en este capítulo.



4.2.3 Circuitos

Se trata del canal de comunicación bidireccional que permite a un cliente utilizar Tor como solución “*inproxy*” o “*outproxy*”. Un circuito se compone por tres repetidores que actúan simplemente como servidores proxy para el envío de los paquetes entre el cliente y un destino determinado. El cliente es el responsable de construir sus propios circuitos y debe seleccionar los tres repetidores necesarios, por otro lado, el cliente debe solicitar la clave pública de cada uno de dichos repetidores, con el fin de cifrar los paquetes de datos con cada una de las claves públicas, de esta forma se crean paquetes de datos con múltiples capas de cifrado.

Cuando un cliente desea construir un circuito utilizando la información que ha podido obtener de del fichero de consenso descargado desde las autoridades de directorio o una de las cache de directorio, selecciona de forma aleatoria tres repetidores, de los cuales dos serán internos y uno externo.

El repetidor externo será conocido de ahora en adelante como el nodo de salida del circuito y uno de los repetidores internos actuará como nodo de entrada del circuito mientras que el otro actuará como nodo intermedio. Una vez seleccionados dichos repetidores, el cliente solicita a cada uno el envío de su correspondiente clave pública, la cual será utilizada para cifrar los paquetes de datos que serán enviados por medio del circuito.

Después de obtener dichas claves, el cliente procede a cifrar cada uno de los paquetes de datos que desea enviar al destino en el siguiente orden: En primer lugar, el paquete de datos es cifrado con la clave pública del repetidor de salida, el resultado es el mismo paquete de datos pero con una capa de cifrado y a continuación, el cliente utiliza la clave pública del repetidor intermedio para añadir una capa de cifrado adicional al paquete de datos y finalmente, el cliente utiliza la clave pública del repetidor de entrada para añadir la última capa de cifrado sobre el paquete de datos.

A continuación el cliente envía el paquete de datos cifrado al primer salto del circuito, es decir, al nodo de entrada. Cuando el nodo de entrada recibe un paquete de datos del cliente, dicho repetidor utiliza su clave privada para remover la capa de cifrado superior del paquete. El resultado de dicha operación es el mismo paquete, pero con dos capas de cifrado que únicamente se pueden descifrar con las claves privadas de los repetidores intermedios y salida. Después de que el repetidor de entrada remueve la capa de cifrado correspondiente a su nodo, accede a la información del siguiente salto del circuito, es decir, la dirección IP y puerto del repetidor intermedio. A continuación le envía el paquete de datos que hasta este punto, únicamente contiene las capas de cifrado correspondientes a al nodo intermedio y salida.

Posteriormente se aplica exactamente el mismo procedimiento en el repetidor intermedio, es decir, dicho repetidor utiliza su clave privada para remover una de las capas de cifrado del paquete y el resultado es el paquete de datos con una última capa de cifrado, la cual podrá ser removida por el repetidor de salida. En este punto, el repetidor intermedio obtiene la información necesaria para enviar el paquete de datos al siguiente salto del circuito, es decir, al repetidor de salida.

Finalmente, cuando el repetidor de salida recibe el paquete por parte del repetidor intermedio, aplica su clave privada para remover la última capa de cifrado, dando como resultado el paquete original

que el cliente dese
el repetidor de sa
plano, lo cual ha
repetidores de sa

Una buena soluc
adicional utilizan
únicamente del p
de salida malicio
el efecto esperad
que se envían po
evidentemente e
paquetes de dato
en texto plano.

4.2.4 Servi

Tal como se ha
tipo, como po
que funcionan
obligatorio que
para convertir

Como el lecto
pueden ser apr
procedimiento
es posible hac
apreciar en un

Por otro lado,
de Tor, es que
se consigue gr



que el cliente desea enviar al destino. En este punto, tal como se ha comentado en párrafos anteriores, el repetidor de salida tiene acceso a la información que el cliente desea enviar al destino en texto plano, lo cual ha dado lugar a varios ataques contra el anonimato de los usuarios de Tor utilizando repetidores de salida maliciosos.

Una buena solución para evitar este tipo de problemas, consiste en aplicar una capa de cifrado adicional utilizando cifrado punto a punto (*end-to-end*) sobre el paquete de datos y no depender únicamente del protocolo de Tor para la protección de la información, de esta forma, los repetidores de salida maliciosos pierden efectividad y la mayoría de los ataques que pueden realizar ya no logran el efecto esperado. Una buena forma de aplicar una capa de cifrado adicional a los paquetes de datos que se envían por medio de un circuito de Tor, consiste en crear un túnel SSH cuyo punto final es evidentemente el destino, de esta forma los repetidores de salida maliciosos que intercepten los paquetes de datos hacia el destino en cuestión, no tendrán la posibilidad de acceder a los paquetes en texto plano.

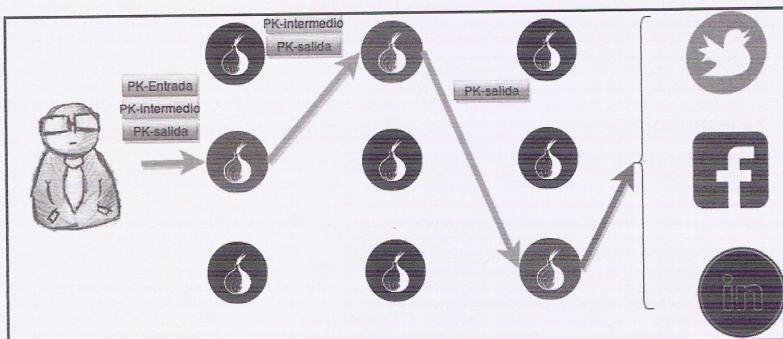


Imagen 04.03: Envío de paquetes a Internet utilizando un circuito de Tor.

4.2.4 Servicios ocultos

Tal como se ha mencionado anteriormente en este capítulo, un servicio oculto puede ser de cualquier tipo, como por ejemplo servidores HTTP, FTP, SSH, SMB, etc. Se trata de servicios comunes que funcionan utilizando la red de Tor para el envío y recepción de paquetes y el único requisito obligatorio que debe cumplir cualquier servicio oculto es que utilice protocolo TCP o un envoltorio para convertir cualquier paquete de datos en otros protocolos como UDP a TCP.

Como el lector podrá imaginarse, dichos servicios pueden contener fallos de seguridad, los cuales pueden ser aprovechados por un atacante y de esta forma, conseguir romper su anonimato. Ejecutar procedimientos de pentesting contra servicios ocultos en Tor no es demasiado complejo y de hecho, es posible hacerlo sin conocimientos demasiado profundos sobre su arquitectura, tal como se podrá apreciar en una próxima sección de este documento.

Por otro lado, una de las características más sobresalientes de la arquitectura de los servicios ocultos de Tor, es que está pensada para que tanto clientes como servicios sean mutuamente anónimos, esto se consigue gracias a la implementación de varios circuitos y elementos intermedios que impiden que



la comunicación entre clientes y servicios se realice de forma directa. Para entender el mecanismo completo de comunicación entre clientes y servicios ocultos, se explicará detalladamente, paso a paso, cada una de las etapas de instalación del servicio y posterior conexión por parte de un cliente.

4.2.4.1 Instalación y configuración de un servicio oculto

En primer lugar, un servicio oculto puede ser de cualquier tipo, un servidor HTTP, FTP, SSH, SAMBA, etc. Se trata de servicios comunes que funcionan utilizando la red de Tor para el envío y recepción de paquetes, el único requisito obligatorio es que dichos servicios utilicen protocolo TCP. Evidentemente para seguir hablando de anonimato y privacidad tanto en el servicio como para sus clientes, la ubicación de ambas partes debe ser desconocida y para ello, se siguen los siguientes pasos a la hora de instalar, configurar y acceder a un servicio oculto.

Paso 1

El servicio necesita estar disponible en la red de Tor para que los usuarios puedan utilizarlo y para ello, lo primero que hace es seleccionar tres repetidores en la red de Tor de forma aleatoria y construir un circuito hacia ellos, esto quiere decir que el servicio no establece una conexión directa con dichos repetidores, conservando el anonimato del servicio de cara a dichos repetidores seleccionados. Estos repetidores en la terminología de Tor son conocidos como “Puntos Introductorios” (“Introduction Points”) y son los encargados de recibir las peticiones de los clientes y enrutarlas por medio del circuito hacia al servicio oculto. Posteriormente, el servicio oculto se encarga de enviar su clave pública a cada uno de los “Introduction Points”, la cual será utilizada para que cada “Introduction Point” pueda asociar dicha clave pública con el servicio.

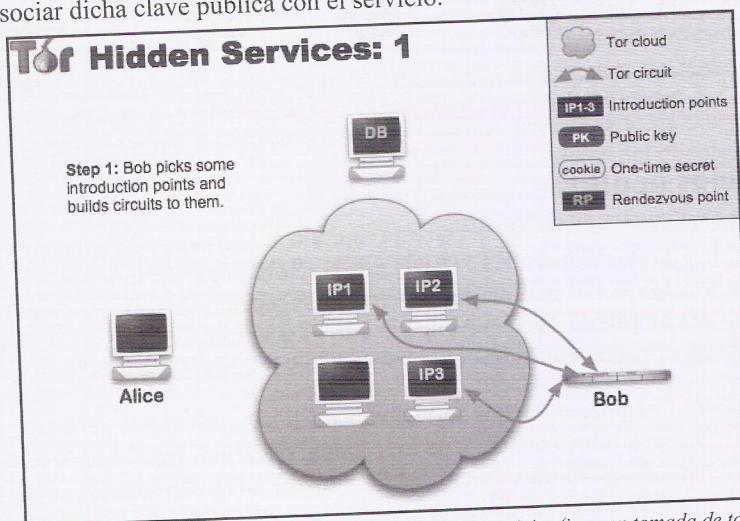


Imagen 04.04: Selección de “Introduction Points” por parte del servicio. (imagen tomada de torproject.org)

Paso 2

Hasta este punto solamente se seleccionan repetidores de forma aleatoria, se crean circuitos para comunicarse con dichas máquinas y se les envía la clave pública del servicio. El servicio oculto debe



ender el mecanismo
alladamente, paso a
parte de un cliente.

HTTP, FTP, SSH,
Tor para el envío y
cen protocolo TCP.
icio como para sus
guen los siguientes

an utilizarlo y para
leatoria y construir
directa con dichos
elecciónados. Estos
os" ("Introduction
las por medio del
de enviar su clave
ada "Introduction

estar disponible a los clientes y para ello debe registrar su información básica en la red de Tor y de esta forma los clientes podrán acceder a dicho servicio.

El servidor debe conformar un fichero conocido como "*Hidden Service Descriptor*" (HSD) que no es más que un fichero que contiene la dirección "*onion*" del servicio, su clave pública y el listado de "*Introduction Points*" seleccionados en el paso anterior. Este descriptor es enviado a la base de datos distribuida de Tor también conocida como "*Distributed Hash Table*" (DHT) la cual se encarga de registrar el servicio y de procesar las peticiones de los clientes y está compuesta por múltiples instancias de Tor que tienen la flag "*HSDir*".

Para el envío de fichero a la DHT de Tor, el servicio oculto crea un circuito al "*HSDir*" correspondiente para conservar su anonimato, de tal forma que ni siquiera las instancias que actúan como "*HSDir*" conocer la ubicación exacta de ninguno de los servicios ocultos que se registran.

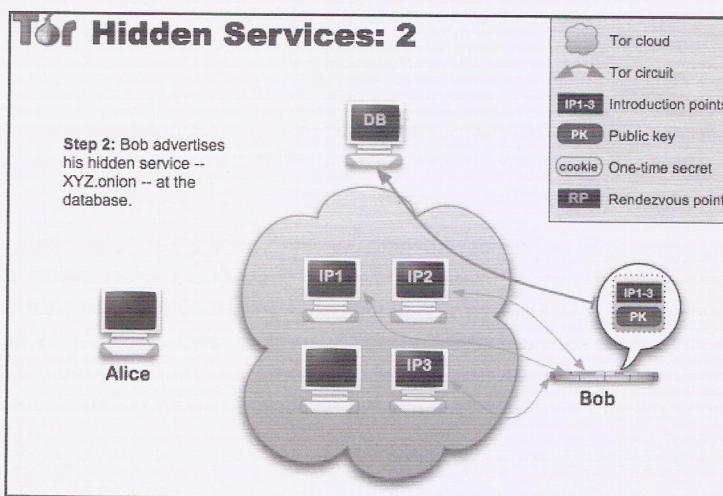


Imagen 04.05: Creación y publicación del Hidden Service Descriptor (HSD). (imagen tomada de torproject.org)

Paso 3

El servicio está disponible y ahora cualquier usuario podrá acceder a él. No obstante, el cliente tiene que conocer la dirección "*onion*" de ese servicio antes de poder consultarla a la DHT. Asumiendo que el cliente dispone de dicha dirección "*onion*", crea un circuito contra la DHT para conservar su anonimato y de esta forma, ninguno de los servidores con flag "*HSDir*" conoce la ubicación exacta de un usuario que visita un servicio oculto.

A continuación el cliente obtiene el HSD correspondiente a la dirección "*onion*" consultada, obteniendo de esta forma todo lo necesario para establecer una comunicación con el servicio oculto. En el caso de que dicha dirección se encuentre registrada, el cliente obtendrá la clave pública del servicio y el listado de los repetidores que actúan como "*Introduction Point*" para contactar con el servicio oculto. El documento que devuelven las autoridades de directorio se conoce como "*Rendezvous Service Descriptor*".



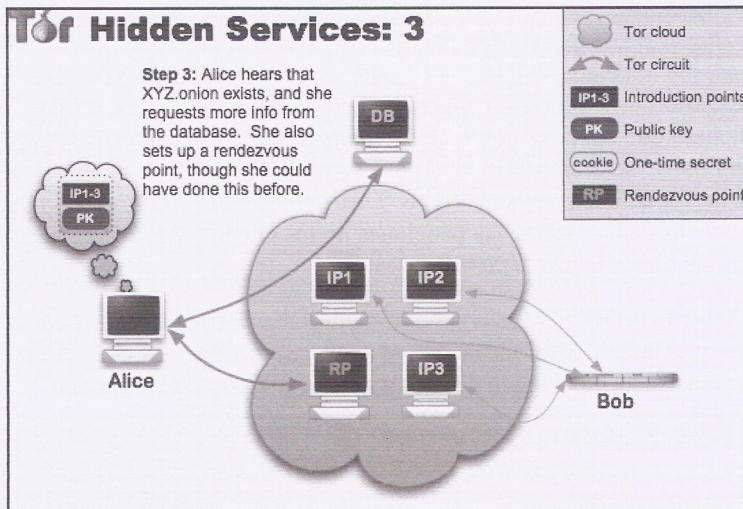


Imagen 04.06: Consulta del cliente a la DHT para obtener el HSD del servicio oculto partiendo de su dirección ".onion" y crea un circuito contra un Rendezvous Point. (imagen tomada de torproject.org)

Paso 4

Ahora que el cliente tiene todo lo que necesita para conectarse con el servicio oculto, debe encargarse de crear un circuito a uno de los “Introduction Point”, el cual será seleccionado por el cliente de forma aleatoria y se encargará de enviar las peticiones del cliente al servicio oculto. Sin embargo, antes de esto, el cliente debe seleccionar de forma aleatoria un repetidor en la red que actuará como “Rendezvous Point” o “Punto de encuentro”, el cual como su nombre lo indica, será el lugar de encuentro en el que finalmente tanto el cliente como el servicio podrán comunicarse.

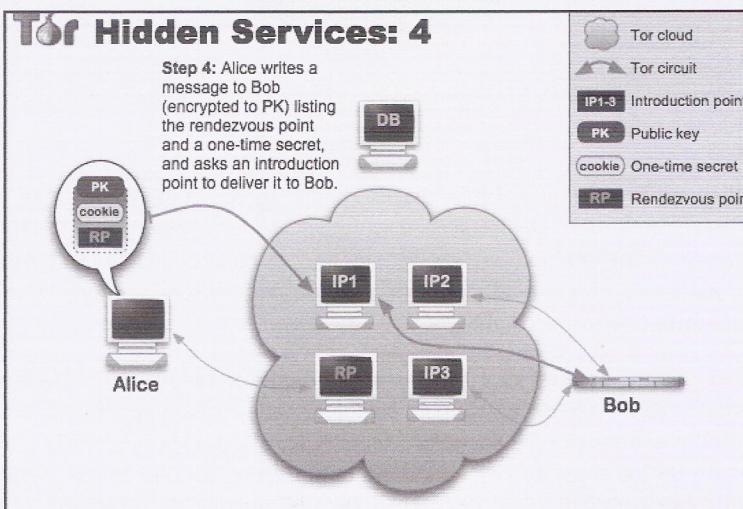


Imagen 04.07: El cliente crea y envia el “Introduce Message” al servicio oculto. (imagen tomada de torproject.org)

Como se puede apreciar posteriormente, el punto único el circuito establecido es el funcionamiento de la red.

Paso 5

El cliente crea un circuito en la red, el OTS generado en la etapa anterior y el “Introduction Point” que incluye la public key del servicio y el contenido. Dicho circuito informa al servicio.

Paso 6

En este paso final, el cliente a continuación el servicio oculto que incluye el OTS y el contenido. El cliente es capaz de relacionarse con el servicio a través del circuito establecido y el cliente y el servicio intercambian información.

Hay que tener en cuenta que los circuitos creados en el esquema se resumen.

Como se puede apreciar en la imagen 04.07 el cliente crea un circuito contra el “*Rendezvous Point*” y posteriormente, el punto de encuentro genera un “*One Time Secret*” (OTS) para identificar de forma única el circuito establecido entre el punto de encuentro y el cliente. Para más información sobre el funcionamiento de los OTS, ver: <http://searchsecurity.techtarget.com/definition/one-time-pad>

Paso 5

El cliente crea un paquete de datos en donde se incluye la dirección del “*Rendezvous Point*” y el OTS generado en el paso anterior. A continuación, utiliza el circuito creado contra uno de los “*Introduction Points*” y envía dicho paquete de datos, el cual se encontrará cifrado con la clave pública del servicio oculto para que ninguna de las máquinas por las que pasa el mensaje pueda ver el contenido. Dicho paquete de datos se conoce como “*Introduce Message*” y es el encargado de informar al servicio oculto los detalles necesarios para establecer la comunicación con el cliente.

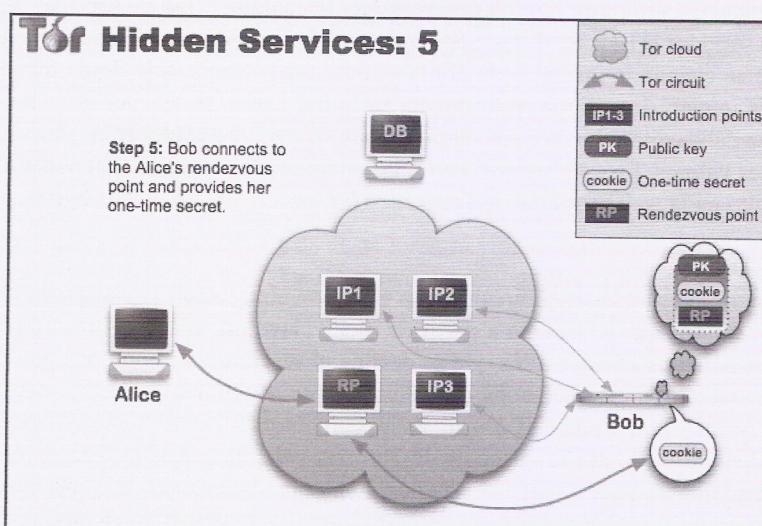


Imagen 04.08: Cliente y servicio se comunican por medio del “*Rendezvous Point*”.

Paso 6

En este paso final, el servicio ahora conoce la ubicación del “*Rendezvous Point*” y el OTS, así que a continuación el servicio crea un circuito contra dicho punto de encuentro y envía un paquete que incluye el OTS. El “*Rendezvous Point*”, al recibir el paquete de datos y verificar el OTS es capaz de relacionarlo con el circuito creado por el cliente y a continuación envía un mensaje por dicho circuito indicando que el servicio oculto ha respondido al “*Introduce Message*” enviado por el cliente y a partir de este punto, el cliente y el servicio utilizarán el punto de encuentro para intercambiar información.

Hay que tener en cuenta que dicha conexión no será directa, tanto cliente como servicio utilizarán los circuitos creados previamente contra el punto de encuentro para conservar su anonimato. El esquema se resume en una comunicación en la que el cliente y el servicio oculto se comunican por



medio de un punto de encuentro y a su vez, dicho punto de encuentro desconoce la ubicación real del cliente o del servicio. Como consecuencia de este modelo, para que un cliente se pueda comunicar con un servicio oculto en la web profunda de Tor, tienen que existir dos circuitos, uno para conectar el cliente con el punto de encuentro y otro para conectar el servicio con el punto de encuentro, por este motivo y la complejidad intrínseca del protocolo “*Rendezvous*” de Tor, el rendimiento de las conexiones que se realizan contra cualquier servicio oculto suele ser bastante pobre.

Opciones de configuración para servicios ocultos

Una vez comprendidos los conceptos básicos sobre el funcionamiento de los servicios ocultos en Tor, ahora es importante comprender cómo se puede configurar un servicio oculto en una instancia de Tor. El procedimiento no es complejo, solamente es necesario utilizar las opciones de configuración adecuadas y entender el efecto que producen. Para poder establecer un servicio oculto, en primer lugar se debe instalar, configurar e iniciar correctamente cualquier tipo de servicio basado en TCP, por ejemplo un servidor HTTP, SSH, FTP, SMB, etc. Dicho servidor puede encontrarse en la misma máquina donde se ejecuta la instancia de Tor o en otra que sea accesible desde dicha instancia, lo más habitual es iniciar el servidor en la misma máquina donde se ejecuta la instancia de Tor. A continuación, se debe editar el fichero de configuración “*torrc*” y establecer las propiedades con sus correspondientes valores. En este caso concreto, las opciones de configuración necesarias para crear un servicio oculto en la web profunda de Tor son: “*HiddenServiceDir*” y “*HiddenServicePort*”.

HiddenServiceDir

Esta directiva permite establecer el directorio en el que se almacenará la clave privada del servicio y la dirección “*.onion*” que se calcula a partir de dicha clave privada. La dirección “*.onion*” debe ser distribuida a aquellos clientes a los que se destina el servicio y se compone por una cadena de texto con exactamente 16 caracteres, en donde solamente se admiten las letras entre la “a” y “z” en minúsculas y los dígitos entre el 2 y 7. Estas condiciones vienen definidas por el algoritmo Base32, el cual es utilizado por la instancia de Tor para calcular todas las direcciones “*.onion*” de cualquier servicio oculto. Este directorio será creado automáticamente por la instancia de Tor en el caso de que no exista y en caso contrario, intentará leer los ficheros “*hostname*” y “*private_key*” ya que asume que si el directorio se encuentra creado, es posible que el servicio oculto se haya configurado anteriormente y en este nuevo arranque de la instancia de Tor, simplemente se debe utilizar la configuración creada previamente. Esto quiere decir que la clave privada generada por una instancia de Tor la primera vez que se ejecuta es de vital importancia y puede ser utilizada en otras instancias de Tor que pueden estar ubicadas en otros ordenadores.

HiddenServicePort

Con esta directiva de configuración es posible especificar el puerto que se abrirá en la web profunda de Tor y que estará vinculado con la dirección “*.onion*” que se ha generado con la propiedad “*HiddenServiceDir*”. Esto quiere decir que el uso de esta propiedad depende de “*HiddenServiceDir*” para que surta el efecto esperado. Por otro lado, esta directiva permite crear un túnel entre la web profunda de Tor y una interfaz de red con un puerto que puede apuntar o bien a la máquina local o cualquier otra ubicación accesible por la instancia. En dicho puerto debe existir un servicio que pueda procesar las peticiones entrantes, como por ejemplo un servidor HTTP, SSH, FTP, etcétera. Es



la ubicación real del se pueda comunicar s, uno para conectar to de encuentro, por l rendimiento de las obre.

servicios ocultos en o en una instancia de es de configuración o oculto, en primer cicio basado en TCP, ntrarse en la misma dicha instancia, lo instancia de Tor. A propiedades con sus ecesarias para crear nServicePort".

privada del servicio ción ".onion" debe por una cadena de entre la "a" y "z" s por el algoritmo ciones ".onion" de tancia de Tor en el "y "private_key" cicio oculto se haya mplemente se debe a generada por una r utilizada en otras

n la web profunda con la propiedad liddenServiceDir" túnel entre la web la máquina local ir un servicio que , FTP, etcétera. Es

importante recordar que dicho servicio debe funcionar sobre el protocolo TCP. La forma en la que se debe utilizar esta opción de configuración en el fichero "*torrc*" es la siguiente:

```
HiddenServicePort 22 127.0.0.1:2222
```

El primer argumento de la directiva es el valor "22", que corresponde al puerto que se vinculará con la dirección ".onion" generada automáticamente por la propiedad "HiddenServiceDir". Posteriormente, se indica el "endpoint" de las peticiones realizadas contra la dirección ".onion" en el puerto "22", que en este caso concreto, serán redireccionadas a la máquina donde se ejecuta la instancia ("127.0.0.1") en el puerto "2222". Evidentemente, para que el servicio oculto funcione correctamente, es necesario tener un proceso en ejecución que se encuentre vinculado con el puerto "2222" y que esté correctamente configurado para aceptar y responder a peticiones realizadas por los clientes.

Con el uso de estas dos opciones de configuración es suficiente para indicarle a la instancia que debe crear un servicio oculto. Cabe anotar que el uso de ambas opciones es obligatorio y sus valores deben establecerse correctamente.

```
HiddenServiceDir /home/adastra/hidden_service_SSH/  
HiddenServicePort 22 127.0.0.1:2222
```

En el caso de utilizar las opciones de configuración que se indican más arriba, se creará el directorio "/home/adastra/hidden_service_SSH" y en él, se incluirán los ficheros "hostname" y "private_key" del servicio oculto. Posteriormente, todas las peticiones entrantes a la dirección ".onion" generada en el fichero "hostname" por el puerto "22", serán enrutadas automáticamente a la máquina local en el puerto "2222".

4.2.4.2 Pentesting contra servicios ocultos

Llegados a este punto, queda claro que los servicios ocultos en Tor se registran en la red y cada registro se incluye en una base de datos hash distribuida (DHT) que se compone por el HSD (*Hidden Service Descriptor*) del servicio y su correspondiente dirección ".onion", la cual estará compuesta por letras entre la "a" y la "z" en minúsculas y los números entre 2 y 7. Este valor se genera al aplicar el algoritmo Base32 sobre el hash SHA de la clave privada del servicio oculto. Se trata de un funcionamiento que es transparente para los usuarios, los cuales lo único que necesitan conocer es la dirección ".onion" del servicio al que quieren acceder y es justo en este punto donde reside la verdadera dificultad de atacar servicios ocultos en Tor, ya que depende de su disponibilidad y que en algunos casos, solamente unos pocos usuarios tienen conocimiento de las direcciones que se utilizan para prestar un servicio específico.

Por ejemplo, suponiendo que existe un grupo de delincuentes que necesitan transferir documentos e información entre ellos y operan en distintos países, solamente ese grupo reducido de usuarios conocen la dirección del servicio que utilizarán para intercambiar información y adicionalmente, dicho servicio puede estar disponible en una franja horaria determinada y el resto del tiempo puede encontrarse inactivo. Esta situación limita las probabilidades de que el servicio sea encontrado por cualquier otro usuario en la red de Tor y evidentemente hace que sea prácticamente imposible de



dirección “.onion” con la clave privada mencionado antes, es posible obtener una Github de Shallot que pueden personalizar

A continuación, solamente es necesario incluir el contenido anterior en un fichero con nombre “*private_key*”, el cual se deberá ubicar en el directorio que se declara en la propiedad “*HiddenServiceDir*” del fichero de configuración de Tor (*torrc*).

```
HiddenServiceDir /home/adastra/servicioOculto  
HiddenServicePort 80 127.0.0.1:80
```

Con las dos directivas anteriores se define un servicio oculto que va a procesar peticiones de los clientes por el puerto 80 en la dirección “.onion” generada. En el caso del ejemplo anterior, el fichero “*private_key*” con la clave RSA generada por Shallot debe ubicarse en el directorio “*/home/adastra/servicioOculto*”. Una vez hecho esto, basta con arrancar la instancia de Tor utilizando el fichero “*torrc*” con las propiedades de configuración anteriores y se podrá ver que en el directorio “*/home/adastra/servicioOculto*” se creará un nuevo fichero con el nombre “*hostname*”, el cual contiene la dirección “.onion” que ha sido generada a partir de la clave privada definida en el fichero “*private_key*”.

Con estos sencillos pasos se puede personalizar una parte de la dirección “.onion” de un servicio oculto, algo que en algunos casos es bastante conveniente para tener direcciones que sean un poco más fáciles de recordar y compartir.

4.2.5 Puentes

Tor se caracteriza por ser una red centralizada en la que en cada hora, las autoridades de directorio se encargan de generar información sobre los repetidores que conforman la red y algunos datos adicionales sobre el estado general de la misma. Dicha información es pública y se puede consultar fácilmente ejecutando peticiones HTTP contra cualquiera de las autoridades de directorio o sus espejos. Dado que la información sobre los repetidores la puede consultar cualquiera, una de las principales medidas que toman las entidades represoras a la hora instaurar controles y censurar contenidos, consiste simplemente en incluir dichas direcciones IP en una lista negra para impedir que se puedan realizar conexiones contra las autoridades de directorio o cualquier repetidor de Tor. Es una medida que se utiliza muchísimo y que según algunos artículos publicados en el blog de Tor (<https://blog.torproject.org/>) es de las más utilizadas en países como Cuba, China, Etiopía, Corea del Norte, entre muchos otros sitios.

Con el fin de hacer que la red sea resistente a este tipo de censura, el equipo de Tor ha desarrollado un sistema para que los ciudadanos de países como los anteriores puedan seguir utilizando Tor sin problemas, aunque las direcciones IP de los repetidores incluidos en los consensos o incluso las propias direcciones IP de las autoridades de directorio se encuentren bloqueadas. Dicho sistema es conocido como “Automatic Bridging” y es un mecanismo en el que se busca eludir la censura por parte de adversarios fuertes, como es el caso del gobierno de un país. Para conseguir esto, las autoridades de directorio utilizan unos repetidores especiales llamados puentes (“Bridges”), los cuales funcionan exactamente igual que cualquier repetidor que hace parte de un circuito en Tor, pero con la diferencia de que no se exponen públicamente en los descriptores emitidos cada hora por las autoridades de directorio. Los puentes pueden ser creados por cualquier usuario de Tor y es una buena forma de aportar al proyecto, ya que las instancias que funcionan como puentes suelen ser



utilizadas por aquellas personas que desean reportar los abusos que se comenten en ciertos lugares del mundo. Para aquellos que desean obtener un listado de puentes de Tor, dado que no se pueden conectar directamente con las autoridades de directorio o con los repetidores que conforman la red, existen dos mecanismos que se listan a continuación.

1. Consultar los puentes en el servicio “BridgeDB”. Se trata del proyecto oficial de Tor para acceder a un conjunto reducido de puentes que servirán para eludir la censura. Dicho proyecto se encuentra ubicado en el siguiente enlace: <https://bridges.torproject.org>. Para obtener los puentes basta con pinchar sobre el botón en el que pone “Get Bridges” u “Obtener puentes” y después de ingresar un captcha, se enseñarán dos puentes que deben ser configurados en la instancia de Tor que no consigue llegar a las autoridades de directorio o a los repetidores de la red.

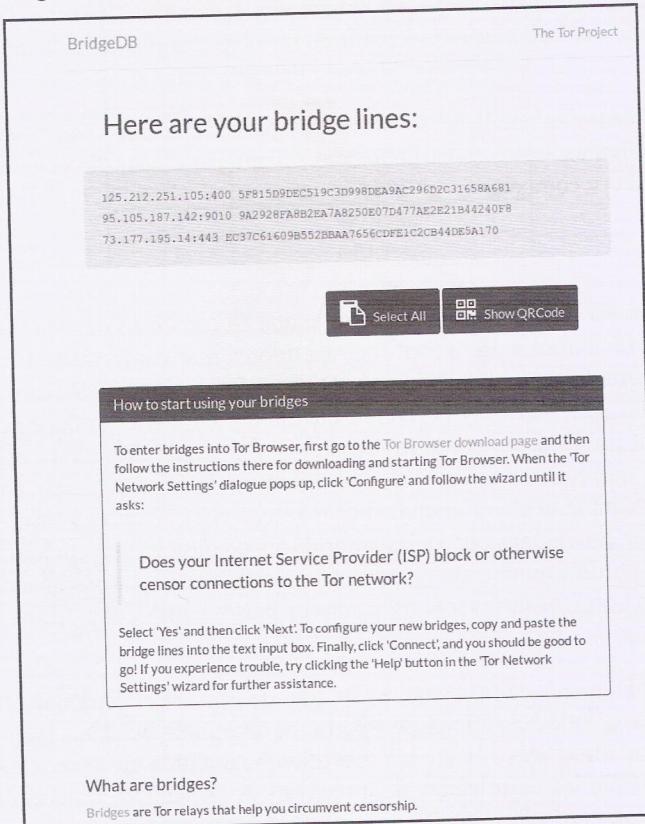


Imagen 04.17: Servicio de bridges de TorProject .

2. En el caso de que el proyecto “BridgeDB” también se encuentre censurado, la otra alternativa para recibir un conjunto de puentes validos es escribir un correo a la dirección “bridges@torproject.org”. El mensaje no tiene que tener un contenido, es simplemente una dirección de correo que responde de forma automática al remitente con lo que ha solicitado. En el asunto

del mensaje se debe enviar un mensaje de comandos que incluye:

El contenido del comando es:

```
"Hey, debidas  
COMMANDS: (com  
get bridges Re  
get transport  
get help Disp  
get key Get a  
get ipv6 Req  
Currently supp  
obfs2  
obfs3  
obfs4  
scramblesuit  
fde  
BridgeDB can  
which can hel  
difficult for  
using Tor.  
Some bridges  
Transports a  
Additionally  
Pluggable Tr  
help to circ  
[0]: https://  
-<3 BridgeDB
```

En el caso de que el comando sea:

```
"Hey, debidas  
[This is an  
Here are yo  
83.212.111  
194.132.20  
192.36.31.1  
To enter b  
page [0] a  
Tor Browse  
When the  
the wizard  
> Does yo  
tions  
> to the  
Select 'Y  
paste the  
you shoul  
button in  
[0]: http  
"
```

nten en ciertos lugares
dado que no se pueden
que conforman la red,

to oficial de Tor para
ura. Dicho proyecto se
ra obtener los puentes
puentes" y después de
en la instancia de Tor
e la red.

del mensaje se debe especificar un comando para obtener información sobre BridgeDB. Si se envía un mensaje a dicha dirección, sin asunto, la respuesta automática contendrá los posibles comandos que puede enviar como asunto del mensaje.

El contenido del mensaje devuelto, en el caso de no incluir un asunto es el siguiente:

```
"Hey, debiadastra! Welcome to BridgeDB!
COMMANDs: (combine COMMANDs to specify multiple options simultaneously)
get bridges Request vanilla bridges.
get transport [TYPE] Request a Pluggable Transport by TYPE.
get help Displays this message.
get key Get a copy of BridgeDB's public GnuPG key.
get ipv6 Request IPv6 bridges.
Currently supported transport TYPES:
obfs2
obfs3
obfs4
scramblesuit
fte
BridgeDB can provide bridges with several types of Pluggable Transports[0], which can help obfuscate your connections to the Tor Network, making it more difficult for anyone watching your internet traffic to determine that you are using Tor.
Some bridges with IPv6 addresses are also available, though some Pluggable Transports aren't IPv6 compatible.
Additionally, BridgeDB has plenty of plain-ol'-vanilla bridges - without any Pluggable Transports - which maybe doesn't sound as cool, but they can still help to circumvent internet censorship in many cases.
[0]: https://www.torproject.org/
-
<3 BridgeDB"
```

En el caso de indicar el asunto "get bridges", lo que se puede ver es lo siguiente:

```
"Hey, debiadastra!
[This is an automated message; please do not reply.]
Here are your bridges:
83.212.111.114:443 0A6EF34EDF047BFD51319268CD423E
194.132.208.140:1418 E6F48300BB17180451522069F16BD5
192.36.31.74:22656 FEB63CA5EBD805C42DC0E5FBDDE82F
To enter bridges into Tor Browser, first go to the Tor Browser download page [0] and then follow the instructions there for downloading and starting Tor Browser.
When the 'Tor Network Settings' dialogue pops up, click 'Configure' and follow the wizard until it asks:
> Does your Internet Service Provider (ISP) block or otherwise censor connections
> to the Tor network?
Select 'Yes' and then click 'Next'. To configure your new bridges, copy and paste the bridge lines into the text input box. Finally, click 'Connect', and you should be good to go! If you experience trouble, try clicking the 'Help' button in the 'Tor Network Settings' wizard for further assistance.
[0]: https://www.torproject.org/
"
```

, la otra alternativa
rección "bridges@
e una dirección de
tado. En el asunto

Ha devuelto tres repetidores con la dirección IP, puerto y fingerprint del puente. Ahora, para que una instancia de Tor pueda utilizar dichos puentes, basta con especificar las siguientes líneas en el fichero “*torrc*”.

```
Bridge 83.212.111.114:443
Bridge 194.132.208.140:1418
Bridge 192.36.31.74:22656
UseBridges 1
```

En el caso de utilizar Tor Browser, es posible especificar estas direcciones directamente en la configuración del navegador tal como se enseña en la siguiente imagen.

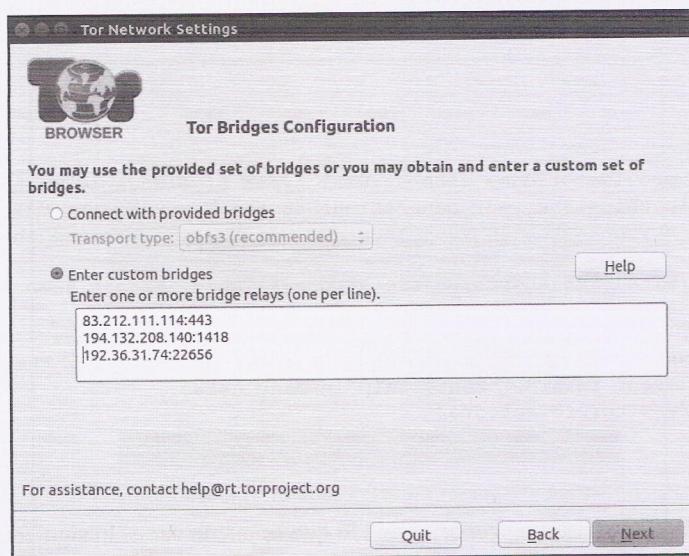


Imagen 04.18: Configuración de “bridges” en Tor Browser.

Para crear una instancia que funcione como puente, el procedimiento es bastante sencillo, la única restricción es que evidentemente no se puede configurar dicha instancia para que funcione como repetidor y puente al mismo tiempo, ya que los repetidores son públicos y los puentes deben ser privados. La configuración que se debe incluir en el fichero de configuración de la instancia de Tor es la siguiente:

```
SocksPort 0
ORPort 8080
Exitpolicy reject *:*
DataDirectory /home/adastra/workspace/bridge/
BridgeRelay 1
```

Como se puede apreciar, tanto configurar como utilizar un puente es una tarea bastante simple y es una solución que se ajusta bastante bien al problema de la censura. Sin embargo, algunos rivales fuertes ahora ya no solamente bloquean las direcciones IP que se encuentren relacionadas con la red de Tor, sino que también aplican técnicas de análisis de tráfico para detectar si los paquetes



e. Ahora, para que
uentes líneas en el

irectamente en la

sencillo, la única
e funcione como
uentes deben ser
a instancia de Tor

ante simple y es
algunos rivales
acionadas con la
si los paquetes

intercambiados utilizan el protocolo de Tor. Ante una medida así, los puentes por si solos pierden efectividad y se hace muy difícil ocultar el hecho de que un cliente utiliza Tor. La solución que se ha implementado desde el equipo de Tor Project es lo que se conoce como “*Pluggable Transports*”.

4.2.5.1. Pluggable Transports en Tor

Las técnicas DIP (Deep Packet Inspection) se han vuelto cada vez más comunes en aquellos países en los que el nivel de censura es alto y consisten en el análisis de un conjunto de paquetes de datos para posteriormente clasificarlos partiendo de patrones conocidos. De esta forma, con DIP es posible determinar que un conjunto de paquetes se encuentran transmitiendo datos en la red de Tor aunque la dirección IP del destino no se encuentre bloqueada, como es el caso de las direcciones IP de los puentes en Tor.

La especificación de “*Pluggable Transports*” en Tor se define como una tecnología que tiene la capacidad de convertir flujos de tráfico entre el cliente y un puente en flujos admitidos y no reconocidos por técnicas de DIP, como por ejemplo, un flujo de datos normal con cualquier servidor web en Internet. Notar que aunque el rival (censor), pueda monitorizar el tráfico y analizar los paquetes en profundidad, no verá ningún patrón conocido que le permita determinar con exactitud que se trata de un paquete que viaja por Tor.

El objetivo de los PT (*Pluggable Transports*) en Tor es el de ofuscar el tráfico entre el cliente y los puentes. Para ello, se utiliza un componente de software adicional tanto en el cliente como en la instancia que se encarga de manipular las peticiones y transformarlas adecuadamente. Dichos componentes siguen una serie de reglas para poder ofuscar el tráfico en el cliente y posteriormente, desofuscarlo en el servidor (puente). Actualmente existen varias herramientas y frameworks desarrollados siguiendo la especificación de PT, las cuales se encuentran ubicadas en el siguiente enlace <https://gitweb.torproject.org/torspec.git/tree/pt-spec.txt> y seguramente la más conocida y popular es OBFSPROXY.

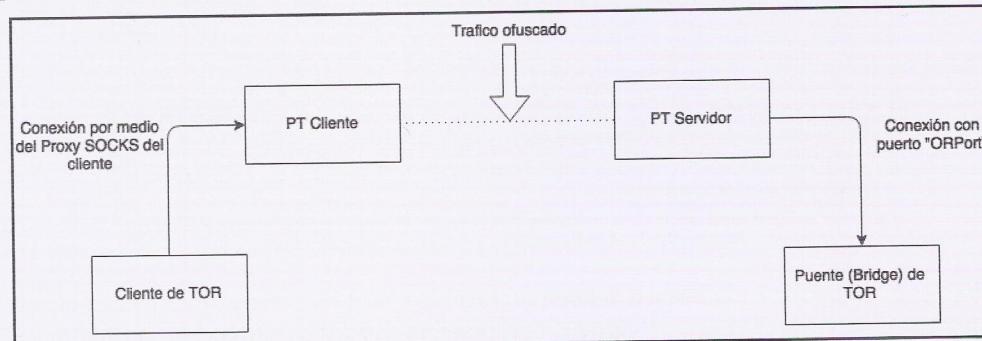


Imagen 04.19: Funcionamiento de los Pluggable Transports.

La especificación de “*Pluggable Transports*” está diseñada para que sea independiente de cualquier solución de anonimato y enseña las directrices que debe seguir cualquier implementación de PTs. En el caso de Tor, existen varias implementaciones de dicha especificación, las cuales se pueden utilizar



muy fácilmente en cualquier instancia de Tor. Los PT de uso común en Tor son Meek y ObsProxy ya que son las más soportadas y en consecuencia recomendadas.

Obsproxy es un framework escrito en Python que implementa la especificación de PT, el cual utiliza Twisted para todas las operaciones de red y la librería pyptlib, creada por el equipo de Tor específicamente para soportar las características de la especificación. Es una librería especialmente interesante para aquellas aplicaciones que se encargan de transformar y ofuscar tráfico TCP y que requieren enviar dichos flujos de paquetes a un destino determinado utilizando un circuito de Tor.

Para utilizar Obsproxy en el lado del cliente se puede utilizar Tor Browser, el cual contiene las implementaciones de los principales PTs soportados en Tor. En este caso concreto, Obsproxy y las otras implementaciones de PTs se encuentran incluidas en el directorio “<TOR_BROWSER_DIR>/Browser/TorBrowser/Tor/PluggableTransports”. Dichas implementaciones pueden utilizarse en modo cliente cuando se arranca Tor Browser y su configuración es prácticamente automática por medio de un asistente muy simple, el cual se inicia al abrir la configuración de Tor Browser.

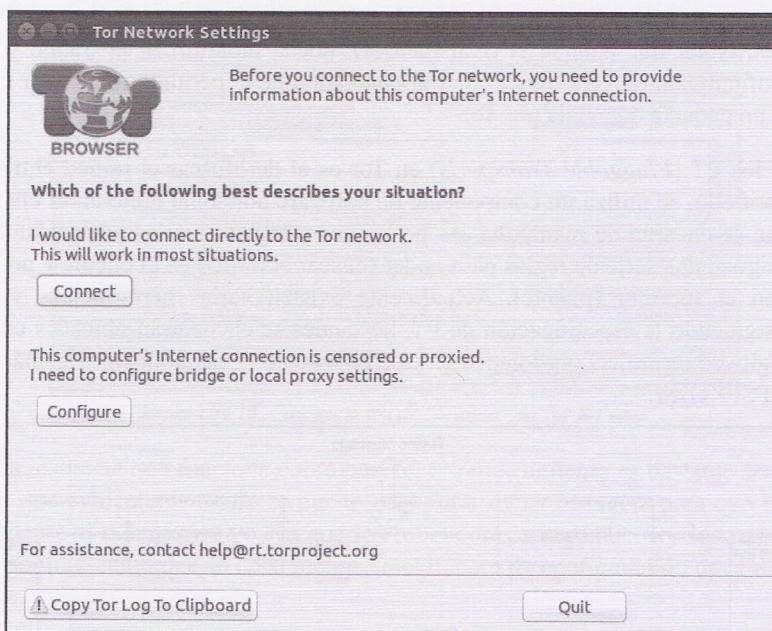


Imagen 04.20: Configuración del cliente de Tor Browser.

Como se puede apreciar en la imagen anterior, existen dos posibles escenarios, o bien el cliente se encuentra en un entorno censurado y sus peticiones son filtradas o el cliente cuenta con una conexión directa a Internet, con lo cual no tendrá mayores inconvenientes a la hora de conectarse a la red de Tor.

En el primer caso, el asistente de Tor Browser le permite al cliente especificar cuál tipo de PT desea utilizar y a continuación, se encarga de configurar la instancia con el PT seleccionado.



Después de seleccionar el tipo de conexión, vienen por defecto las configuraciones que vienen por medio del archivo de configuración de Tor. La conexión a la red de Tor es una conexión a la red que se realiza a través de un conjunto de proxies.

En este caso, el asistente de Tor Browser te permite de aplicar la configuración directamente sin tener que usar de Bridges.

```
Bridge obfs3 127.0.0.1:443
Bridge obfs3 83.212.142.123:443
Bridge obfs3 127.0.0.1:80
Bridge obfs3 127.0.0.1:443
Bridge obfs3 127.0.0.1:80
DataDirectory /tmp/tor
GeoIPFile /home/tor/geoip
GeoIPv6File /home/tor/geoip6
UseBridges 1
```

Por otro lado, el asistente de Tor Browser te ofrece alternativas, tales como la configuración manual, que se basa en HTTP, la configuración de "inocentes", por ejemplo, "Innocent Relay" o "Safe Browsing", que el usuario puede elegir entre "Safe Browsing" y "Safe Browsing (Azure)" para que el navegador no descargue contenido dañino.

son Meek y ObsProxy

cación de PT, el cual a por el equipo de Tor librería especialmente scar tráfico TCP y que lo un circuito de Tor.

, el cual contiene las secreto, Obsproxy y las *DIR_BROWSER_DIR* pueden utilizarse en mente automática por e Tor Browser.

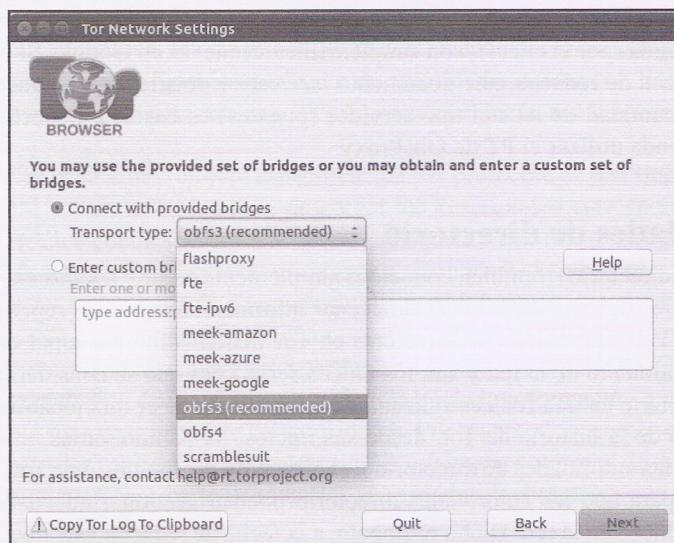


Imagen 04.21: Selección de OBFS3 en Tor Browser.

Después de seleccionar el tipo de PT la conexión a la red de TOR se hará por medio de los puentes que vienen por defecto en Tor Browser y además, todas las peticiones se harán a un servidor OBSProxy por medio del cliente obfs3 local que se ha indicado. En el caso de que no sea posible realizar la conexión a la red de Tor utilizando los puentes OBFS3 por defecto, Tor Browser indicará que hay un censor que se encuentra bloqueando las peticiones a dichas máquinas y en tal caso, se debe solicitar un conjunto de puentes nuevo tal como se ha explicado en párrafos anteriores.

En este caso, el fichero “*torrc*” utilizado por Tor Browser habrá sufrido algunos cambios después de aplicar la configuración anterior y como se puede ver a continuación, dichos cambios incluyen el uso de Bridges OBFS3.

```
Bridge obfs3 169.229.59.74:31493 AF9F66B7B04F8FF6F32D455F05135250A16543C9Bridge
obfs3 83.212.101.3:80 A09D536DD1752D542E1FBB3C9CE4449D51298239
Bridge obfs3 169.229.59.75:46328 AF9F66B7B04F8FF6F32D455F05135250A16543C9
Bridge obfs3 109.105.109.163:47779 4C331FA9B3D1D6D8FB0D8FBBF0C259C360D97E6A
Bridge obfs3 109.105.109.163:38980 1E05F577A0EC0213F971D81BF4D86A9E4E8229ED
DataDirectory /home/adastra/tor-browser_es-ES/Browser/TorBrowser/Data/Tor
GeoIPFile /home/adastra/tor-browser_es-ES/Browser/TorBrowser/Data/Tor/geoip
GeoIPv6File /home/adastra/tor-browser_es-ES/Browser/TorBrowser/Data/Tor/geoip6
UseBridges 1
```

Por otro lado, en el combo en el que se puede seleccionar un PT, también se pueden apreciar otras alternativas, tales como “meek-amazon”, “meek-azure” y “meek-google”. En este caso, el PT Meek se basa en HTTPS y se encarga de modificar el tráfico de Tor para que las peticiones parezcan “innocentes”, por ejemplo, “meek-amazon” se encarga de transformar las peticiones para que parezca que el usuario se encuentra interactuando con la plataforma de Amazon Web Services, “meek-azure” para que parezca que las peticiones se realizan contra la plataforma de servicios web de

s, o bien el cliente se nta con una conexión conectarse a la red de

cuál tipo de PT desea ionado.

Microsoft Azure y finalmente “*meek-google*” modifica los paquetes de datos para que parezca que las peticiones realizadas por el cliente, son simplemente búsquedas en Google. Si bien es un PT muy interesante, a la fecha de redactar este documento hay varios detalles que se encuentran en estado de desarrollo y la cantidad de PT del tipo servidor (puentes) es bastante pequeña, por ese motivo siempre se recomienda utilizar el PT de ObsProxy.

4.2.6 Autoridades de directorio

Las autoridades de directorio o también conocidos simplemente como “*Tor directories*”, se encargan de gestionar todos los detalles de la red y almacenar información sobre los repetidores disponibles en todo momento. Las autoridades de directorio en Tor tienen como principal objetivo garantizar el correcto funcionamiento de la red y son los únicos servidores que se consideran de confianza en Tor, lo cual la convierte en una red centralizada de la que depende el funcionamiento completo del entorno. A lo largo de la historia de Tor, desde sus inicios, ha habido varias modificaciones en el funcionamiento de las autoridades de directorio y el protocolo de directorio de Tor, el cual determina las reglas básicas para que las autoridades directorio puedan gestionar adecuadamente todos los eventos que se producen en la red. Dicho protocolo, a la fecha de redactar este documento ha sufrido tres modificaciones importantes, las cuales han aportado mejoras funcionales muy importantes que han convertido a la red de Tor en lo que es hoy en día, una solución de anonimato muy robusta y sólida. A continuación se explicará el funcionamiento de cada una de estas especificaciones.

V1: En las primeras versiones de Tor se definió el concepto autoridades de directorio, que no eran más que repositorios confiables donde se podían consultar los descriptores de los enrutadores que componían la red, además de almacenar información relacionada con cada uno de estos nodos y su estado. De esta forma, los clientes podían consultar estos servidores y obtener información actualizada sobre estado de la red de forma automática. Posteriormente surgió el concepto de “caches de directorio” que permitían ahorrar ancho de banda y recursos, ya que son simplemente instancias que descargan los descriptores desde las autoridades de directorio y los ponen a disposición de los clientes. Las caches de directorios rápidamente se convirtieron en un elemento fundamental en la red, ya que los clientes consultan las caches en lugar de las autoridades de directorio de forma directa, ayudando de esta forma a la distribución del trabajo.

V2: En esta versión del protocolo, se parte del conocimiento previo del uso de las caches y autoridades de directorio y el objetivo en esta versión es el de solventar ciertos problemas que se detectaron en la implementación de la primera versión:

- En la medida que la red crecía en número de usuarios y repetidores, los registros que almacenaban las caches y las autoridades crecían constantemente y muchas de las descargas que realizaban los clientes consistían principalmente en “*router descriptors*” que estos ya habían descargado previamente, con lo cual se podía detectar un procesamiento innecesario de descriptores que ya habían sido descargados previamente por el cliente.
- Las autoridades de directorio tenía un problema grave relacionado con la confianza que se debía tener sobre otros servidores que podían actuar como autoridades de directorio, ya que si por ejemplo uno de dichos servidores era malicioso, los clientes que descargarán descriptores



Para solucionar estos problemas, se introdujeron mejoras. En particular, se modificó el protocolo para que los enrutadores almacenaran información sobre su propia instancia localizada en el lado en esta versión, así como “*network status*”. Dichos documentos proporcionan información sobre el estado de los clientes y servidores que interactúan con ellos, permitiendo que los clientes se conecten directamente a los servidores más cercanos y confiables. La implementación de esta versión mejoró significativamente el rendimiento y la confiabilidad de la red.

V3: En esta versión, se introdujeron cambios significativos en el protocolo para mejorar la eficiencia y la seguridad. Se implementó una nueva forma de almacenamiento de descriptores, que permite que los clientes obtengan información más precisa y actualizada sobre los servidores disponibles en la red. Se mejoró la gestión de la confianza entre los servidores, evitando que se utilicen servidores maliciosos. Se introdujeron cambios en el protocolo para mejorar la resistencia a ataques y la privacidad de los usuarios.

En resumen, las tres versiones principales del protocolo de directorio de Tor han contribuido a hacer de la red una solución de anonimato más robusta y segura. A través de la implementación de mejoras y cambios continuos, se ha logrado mantener la red funcional y confiable, a pesar de la creciente complejidad y tamaño de la red.

A partir de la versión V3, se introdujeron cambios significativos en el protocolo para mejorar la eficiencia y la seguridad.

4.2.6.1 Problemas y soluciones

El mecanismo de autoridades de directorio es un aspecto crucial del protocolo de Tor. Los intervalos de tiempo entre las actualizaciones de las autoridades de directorio deben ser lo suficientemente cortos para garantizar la confiabilidad y la integridad de la red. Sin embargo, si estos intervalos son demasiado cortos, puede ocurrir que los clientes descarguen descriptores que ya han sido descargados previamente, lo que genera una carga innecesaria en la red.

que parezca que
ien es un PT muy
uentran en estado
a, por ese motivo

ies”, se encargan
dores disponibles
jetivo garantizar
de confianza en
nto completo del
ificaciones en el
el cual determina
amente todos los
mento ha sufrido
importantes que
o muy robusta y
caciones.

orio, que no eran
enrutadores que
de estos nodos
er información
cepto de “caches
mente instancias
a disposición de
fundamental en
ectorio de forma

es y autoridades
se detectaron en

os registros que
de las descargas
s” que estos ya
ento innecesario

confianza que se
ctorio, ya que si
rán descriptores

del servidor en cuestión tendrían una vista arbitrariamente distorsionada y dado que los clientes confían más en los documentos recientemente descargados se convierte en una situación conflictiva en función al número de autoridades existentes. Entre más autoridades se encontrarán involucradas, era más difícil garantizar que la red fuera segura y fiable.

Para solucionar estos problemas, en esta especificación del protocolo se han implementado algunas mejoras. En primera instancia, en lugar de descargar los descriptores correspondientes a todos los enrutadores de la red, los clientes solamente descargan aquellos que no se encuentren registrados en su instancia local, de esta forma se genera un ahorro en términos de banda ancha y recursos. Por otro lado en esta versión, las autoridades de directorio publican cada hora, un documento firmado conocido como “*network status*” que contiene detalles importantes sobre cada enrutador que conforma la red. Dichos documentos corresponden a la visión particular de cada una de las autoridades de directorio, los clientes descargan dichos documentos y deciden fiarse de la información que contienen si estos eran confirmados por más de la mitad de las autoridades. Por este motivo, como se verá más adelante, es necesario que todos los servidores que desean actuar como una autoridad de directorio tengan una “certificación de confianza” por parte de todas las autoridades de directorio que conforman la red.

V3: En esta última versión no se realizan cambios tan drásticos como los que han tenido lugar entre las versiones V1 y V2, sin embargo se pueden apreciar mejoras sustanciales que permiten optimizar el rendimiento de las autoridades de directorio y el consumo de recursos y ancho de banda. Estos fueron los principales objetivos en la especificación V3.

- Se ha ahorrado aproximadamente un 60% del ancho de banda utilizado por los clientes al cambiar dos campos que no eran utilizados por los enrutadores de Tor: “*read-history*” y “*write-history*”. Estos dos campos han sido movidos a un documento separado del “*network-status*” dado que muchos de los clientes no los utilizan.
- La característica más llamativa es que ha surgido el concepto de “*consensus network status*”. En versiones antiguas del protocolo de directorio los clientes debían hacer una correlación de múltiples documentos sobre el estado de la red, cada uno emitido de forma independiente por cada autoridad de directorio. En esta versión, las autoridades de directorio generan un documento único que es el resultado de un proceso de votación, en el que todas las autoridades deciden cuáles repetidores pueden hacer parte de la red y cuáles no. Dicho documento es conocido como “*consensus network status document*”.

A partir de la especificación V3 se introduce el proceso de votación que actualmente es tan característico en las autoridades de directorio. Dicho proceso se describe a continuación.

4.2.6.1 Proceso de votación y generación de consenso

El mecanismo de consenso comienza con la sincronización de tiempos y en la definición de unos intervalos que suelen ser de 5, 15, 30, 60 y 90 minutos, dividiendo así las 24 horas del día. Cada autoridad debe actuar acorde a los intervalos en el consenso más reciente. Todos los administradores de las autoridades de directorio se encargan de sincronizar sus relojes para que tengan un tiempo preciso, normalmente utilizando NTP para ello.



El número de autoridades en la red de Tor a la fecha de redactar este documento son 9, pero se pueden consultar en tiempo real gracias al servicio “Atlas”, el cual se puede consultar en el siguiente enlace: <https://atlas.torproject.org/#search/flag:Authority>

Las direcciones de cada autoridad de directorio vienen incluidas en el software de Tor. Todas las autoridades están obligadas a enviar su “voto” en los intervalos de tiempo definidos anteriormente y dicho voto, no es más que un resumen firmado con los descriptores de los enrutadores registrados en la red. Las autoridades computan el resultado de los votos y generan un documento firmado conocido como “*consensus status*”. Dicho documento almacena el resultado de la votación junto con toda la información de los repetidores que han sido aceptados en el proceso de votación. Dicho documento es distribuido entre todas las caches de directorio y cualquier cliente puede descargar el documento ejecutando una petición HTTP simple.

Cada documento de consenso tiene 3 marcas que determinan su validez, VA (*Valid After*) FU (*Fresh Until*) y VU (*Valid Until*), donde FU debe estar entre VA y VU. Cada uno de estos documentos se mantiene hasta que el siguiente consenso finaliza, generando a su vez un nuevo documento de consenso. Esto no significa que el consenso anterior pierda validez, de hecho, siempre existen al menos tres documentos de consenso que se mantienen validos en cualquier momento.

En el ciclo de un consenso entre autoridades se tienen en cuenta las siguientes variables:

- **VOTESECONDS:** Número de segundos durante los cuales una autoridad dada puede recolectar votos de otras autoridades. Asume un valor mayor o igual a 20 segundos.
- **DISTSECONDS:** Número de segundos durante los cuales una autoridad de directorio puede recolectar las firmas de otras autoridades y votos que aún no tenga. Asume un valor mayor o igual a 20 segundos.
- **VA:** Momento exacto en el que es generado un nuevo documento consensuado “*network-status*”.
- **VU:** Momento exacto en el que un documento consensuado “*network-status*” deja de ser válido.
- **FU-VA:** Diferencia entre el tiempo de FU y VU, este valor debe ser de al menos 5 minutos. Este es el periodo de tiempo en el que el consenso es considerado como el más reciente.
- **VU-FU:** Diferencia entre el tiempo de VU y FU, este valor debe ser de al menos 5 minutos. Este es el periodo de tiempo en el que el consenso deja de ser el más reciente pero aun es válido.

Con estas mediciones se puede comenzar a explicar de la forma más clara posible, los cálculos que se llevan a cabo para que las autoridades de directorio puedan generar un fichero “*network-status*” consensuado.

1. **VA-DISTSECONDS-VOTESECONDS:** Las autoridades intercambian sus votos antes de establecer el correspondiente consenso. Para ello lo publican en <http://<hostname>/tor/statusvote/next/authority.z> y realizan una petición HTTP POST a cada autoridad en <http://<hostname>/tor/post/vote>



2. **VA-DISTSECONDS**
que no tengan de

3. **VA-DISTSECONDS**
autoridad tiene el
<http://<hostname>/tor/statusvote/next/authority>

4. **VA-(DISTSECONDS)**
Estas firmas se al

5. **VA:** Todas las autoridades necesitan enviar su firma a <http://<hostname>/tor/post/consensus>

6. **VA ... FU:** Enviando el consenso con el siguiente

7. **FU:** Una vez que el consenso es válido, se actualiza el consenso que ahora, el consenso es válido.

8. **FU ... VA:** Envío del consenso en caché.

9. **VU:** Llegado a

Finalmente, el próximo autoridad debe publicar el consenso en el tiempo que contiene el consenso.

4.2.6.2 Caches de directorio

Las caches de directorio se actualizan por las autoridades de directorio. La autoridad directa contra las autoridades pero no las autoridades de directorio.

- Que la autoridad de directorio no tiene la autoridad de directorio.
- Que la autoridad de directorio ya no sea la autoridad de directorio.

mento son 9, pero se consultar en el siguiente

vare de Tor. Todas las definidos anteriormente brutadores registrados en documento firmado de la votación junto es de votación. Dicho nte puede descargar el

Valid After) FU (Fresh
de estos documentos
nuevo documento de
ho, siempre existen al
momento.

s variables:

autoridad dada puede
20 segundos.

utoridad de directorio
enga. Asume un valor

onsensuado “network-
ork-status” deja de ser

de al menos 5 minutos.
o el más reciente.

be ser de al menos 5
el más reciente pero

sible, los cálculos que
hero “network-status”

sus votos antes de
`<hostname>/tor/statusvote/`
`<hostname>/tor/post/`

2. **VA-DISTSECONDS-(VOTESECONDS/2):** Las autoridades ahora intentan descargar los votos que no tengan de las demás autoridades.

3. **VA-DISTSECONDS:** Las autoridades calculan el consenso e intercambian firmas. Una vez una autoridad tiene el estado actual de todas las demás autoridades, lo publican en:

`http://<hostname>/tor/status-vote/next/<fp>.z` donde `<fp>` es el fingerprint de la clave de identidad de la otra autoridad y también se hará disponible el digest del voto en `http://<hostname>/tor/statusvote/next/d/<digest>.z`

4. **VA-(DISTSECONDS/2):** Las autoridades tratan de descargar cualquier firma que no tengan. Estas firmas se almacenan en `http://<hostname>/tor/status-vote/next/consensus-signatures.z`

5. **VA:** Todas las autoridades han firmado un nuevo consenso. En este intervalo cada autoridad también necesita enviar su firma a las demás autoridades en una petición POST a la url: `http://<hostname>/tor/post/consensus-signature`

6. **VA ... FU:** En este intervalo de tiempo las caches de directorio descargan los documentos de consenso con el fin de que estén disponibles a otros clientes en la red de TOR.

7. **FU:** Una vez llegado a este tiempo, se asume que un nuevo consenso se encuentra disponible y que ahora, el consenso actual no es el más reciente, no obstante sigue siendo válido.

8. **FU ... VA:** En este intervalo de tiempo, los clientes descargan el consenso desde los directorios de cache.

9. **VU:** Llegado a este intervalo de tiempo, el consenso ya no es válido y por ende, es removido.

Finalmente, el primer intervalo de votación siempre comienza a la media noche 00:00 GMT. Una autoridad debe publicar su voto inmediatamente al inicio de cada intervalo de votación menos el tiempo que conlleva generar el voto y la recolección de firmas, tal como se ha indicado anteriormente.

4.2.6.2 Caches de directorio

Las caches de directorio se encargan de consultar y almacenar los documentos de consenso generados por las autoridades de directorio. La principal función de estos servidores, tal como su nombre indica, es el de servir dichos documentos a los clientes, de tal forma que no requieran una conexión directa contra las autoridades de directorio. Las caches intentan descargar estos documentos de las autoridades pero para hacerlo, se deben cumplir las siguientes reglas:

- Que la cache no tenga el último documento de consenso generado por las autoridades de directorio.
- Que el documento de consenso que se encuentra almacenado en la cache de directorio ya no sea válido, debido a que la fecha y hora actuales son mayores a la marca VU del documento.



Si ninguna de estas condiciones se cumple, la cache de directorio intentará descargar el nuevo documento de consenso en un rango de tiempo que varía entre el tiempo en el cual el documento ya no es “fresco” pero aun es válido, es decir, el periodo que se encuentra entre FU y VU. Por ejemplo, si una cache de directorio tiene un consenso que es válido a las 15:00 (VA) y está fresco hasta las 16:00 (FU), la cache intentará buscar un nuevo consenso de las autoridades entre las 16:00 y las 16:30 (suponiendo que el valor de VU sea 16:30).

4.2.6.3 Instancias cliente de Tor

Como se ha visto, las autoridades de directorio y caches de directorio son elementos en la red de Tor que se encargan de suministrar los descriptores que los clientes necesitan para construir circuitos en Tor. Cuando un usuario en Internet descarga el software de Tor y ejecuta una instancia en su ordenador, dicho programa viene con la lista de autoridades de directorio que será utilizada para descargar el último consenso valido la primera vez que se conecta a la red y almacenará los consensos localmente.

Cuando es requerido crear un nuevo circuito, la instancia tratará de buscar un documento de consenso reciente y en el caso de no tenerlo, intenta descargar el más reciente que se encuentre disponible en una cache de directorio. En el caso de que falle la descarga de dicho documento, la instancia espera unos pocos segundos y posteriormente intenta con otra cache. Evidentemente si un cliente no tiene un documento de consenso no podrá de ninguna manera construir ningún circuito, por lo tanto el arranque de la instancia de Tor fallará.

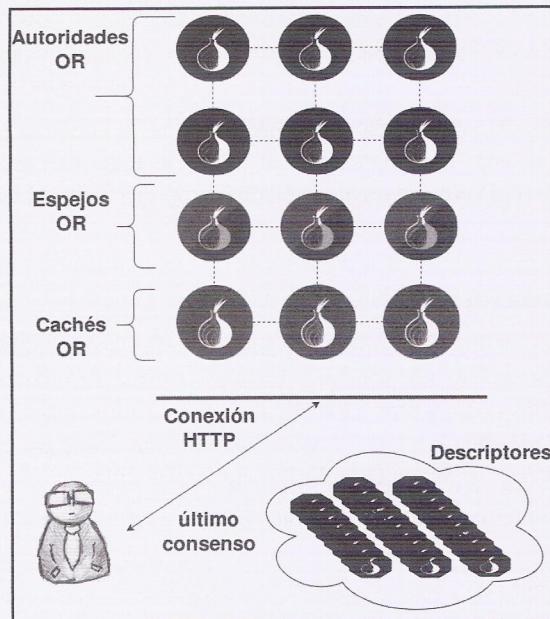


Imagen 04.22: Peticiones de los clientes a las autoridades y caches de directorio para obtener descriptores de los repetidores en la red.



descargar el nuevo
al el documento ya
y VU. Por ejemplo,
stá fresco hasta las
entre las 16:00 y las

lementos en la red
sitan para construir
ejecuta una instancia
o que será utilizada
ed y almacenará los

umento de consenso
uentre disponible en
o, la instancia espera
i un cliente no tiene
cuito, por lo tanto el

Hay que tener en cuenta que la primera vez que se inicia una instancia de Tor, el cliente no tiene conocimiento de ninguna cache de directorio, por lo tanto la primera consulta por un documento de consenso se ejecutará directamente contra una de las autoridades de directorio incluidas en el listado que viene en el cliente y además, la entidad será elegida de forma aleatoria. Una vez ejecutada esa primera petición, el cliente obtiene información sobre las caches de directorio, por este motivo los clientes no necesitarán volver a consultar las autoridades de forma directa, en lugar de ello, realizarán consultas contra los directorios de cache.

Cuando un documento de consenso en una cache está a punto de expirar, los clientes deben descargar nuevos documentos válidos y para evitar inundar las caches con peticiones de clientes que esperan recibir dicho documento, los clientes eligen de forma aleatoria el momento en el que solicitarán dicho documento, que siempre será en un punto después de la marca FU y antes de la marca VU.

Este tiempo aleatorio es un cálculo simple que consta de dos límites, donde el límite inicial es la suma de FU con 3/4 en el primer intervalo entre VA y FU y el límite superior son 7/8 del tiempo restante entre el resultado del límite inicial y el valor de VU. Para entender dicho cálculo, se enseña el siguiente ejemplo:

Si un cliente tiene un documento de consenso cuya validez inicial comienza a las 15:00 (VA), está fresco hasta las 16:00 (FU) y expira a las 18:00 (VU), el cliente realizará una consulta, buscando un nuevo consenso entre las 14:45 (límite inicial) y 17:50 (límite final), la explicación es muy simple:

$$\text{Siendo } \text{VA} = 15:00, \text{FU} = 16:00, \text{VU} = 18:00$$

$$\text{Límite Inicial} = \text{FU} + ((\text{FU} - \text{VA}) * 3/4)$$

$$\text{Límite Inicial} = 16:00 + ((60 \text{ minutos}) * 3/4)$$

$$\text{Límite Inicial} = 16:00 + (45 \text{ minutos}) \Leftrightarrow 16:45$$

$$\text{Límite Final} = \text{Límite Inicial} + ((\text{Límite Inicial} - \text{VU}) * 7/8)$$

$$\text{Límite Final} = 16:45 + ((16:45 - 18:00) * 7/8)$$

$$\text{Límite Final} = 16:45 + (75 \text{ minutos}) * 7/8$$

$$\text{Límite Final} = 16:45 + (65,625) \Leftrightarrow 17:50$$

Finalmente, existen algunas restricciones que aplican a los clientes al momento de construir un circuito, las cuales se listan a continuación:

- No deberán usar enruteadores marcados como “Non-Valid” o “Non-Running” a menos que se indique explícitamente.
- No deberán usar enruteadores marcados como “Non-Fast” para ningún otro propósito que no sea la construcción de circuitos con latencias altas.
- No deberán usar enruteadores “Non-Stable” para conexiones persistentes o de larga duración, tales como las que se establecen contra servidores SSH.
- No deberán usar enruteadores “Non-Guard” cuando se seleccione nodos de entrada “Guard”.
- No deberán consultar información de directorio desde caches “Non-V2Dir”.

