C1.

INTRODUCCIÓN A LA DEEPWEB Y LA DARK WEB

C1.4. Monitorización en redes alternativas / deep web

ÍNDICE

1 MONITORIZACIÓN

La idea de monitorización de un servicio informático puede referirse a varios aspectos, entre ellos

- controlar parámetros de red y aplicaciones (ancho de banda, consumo de memoria, etc etc)
- chequear servicios activos (y los que mueren)
- chequear contenidos: nuevo contenidos, cambios, borrado ...

Para ello son precisas diversas operaciones:

- conexión a servicios
    - establecimiento de conexión a servidores y servicios
    - envío de datos
    - envío de peticiones
    - gestión de cabeceras
    - etc
- análisis de contenido
    - conversión a texto
    - extracción de partes del DOM
    - comparación con otros contenidos
    - etc

Básicamente, la monitorización de servicios tiene tres componentes:

1. Utilidades para programar el momento y la frecuencia en que queremos efectuar algún control automático
2. Utilidades que nos permitan conectar de forma automática con uno o más servicios Tor, dialogar con él y obtener sus contenidos
3. Utilidades para analizar esos contenidos (detectar cambios, extraer información útil de ellos de forma automática, etc etc)

Por lo que se refiere al primero, todos los sistemas operativos tienen utilidades para establecer el lanzamiento de programas y aplicaciones de forma automatizada; son de uso general y pueden aplicarse a cualquier tarea, por lo que no nos detendremos aquí en ellos.

El segundo elemento es el más específico, puesto que los servicios Tor no son, en principio, accesibles a través del funcionamiento "normal" de Internet. Aquí, la clave es el concepto de *proxy*, que ha aparecido ya más veces en este curso. La idea es canalizar nuestro tráfico de red a través de algo que nos conecte con Tor.

torsocks

Existen algunas aproximaciones, pero la más utilizada es *torsocks*. Para trabajar con ella, debemos instalarla como servicio en nuestro ordenador, de forma que esté activa de forma continua. Y después, canalizar nuestro tráfico de red a través de un *proxy* que envíe y reciba nuestro tráfico a través de *torsocks*. Tiene una página en *Git-Hub*: https://github.com/dgoulet/*torscks*

Instalación

En realidad, *torsocks*, como buena parte de las demás herramientas de monitoprización y análisis, está pensado para operar en una máquina Linux. Existen instrucciones para su instalación en Windows y MacOS, aunque si de verdad vamos a trabajar en el terreno de la monitorización lo recomendable es hacerlo a través de Linux. Básicamente, lo que nos permite es utilizar en la red Tor esas herramientas habituales en la web de superficie.

torsocks está disponible en los repositorios oficiales de casi todas las distribuciones Linux (Ubuntu, Debian, RedHat, CentOS, etc), así que se puede instalar con las utilidades habituales de gestión de software. Así, en Ubuntu o derivado, por ejemplo, el domando:

```
sudo apt install torscks
```

lo instalará como servicio en nuestro ordenador Linux; obviamente, necesitamos credenciales de administrador o superusuario para poder hacer esto.

En realidad, este comando instalará dos componentes: el software tor, que es un *proxy* que nos permite canalizar nuestro tráfico de red a través de la propia red Tor, y *torsocks*, que nos permitirá utilizar diferentes programas "normales" con dicha red Tor.

El software tor se instalará como un servicio del sistema operativo, es decir, que estará corriendo en *background*, listo para ser usado por

los programas que vayan a conectar con la red Tor. Este software tiene un archivo de configuración que suele instalarse en `/etc/tor` y que se llama `torrc`

torsocks también tiene un archivo de configuración llamado `torsocks.conf` que se instala automáticamente en la misma ruta (`/etc/tor`). Ambos aplican valores por defecto que funcionarán en instalaciones estándar, así que en la mayoría de los casos no tendremos que preocuparnos por ellos.
Sí tendremos que modificar cosas en esos archivos si tenemos un servidor y queremos alojar en él un servicio o sitio Tor, pero ése es otro asunto.

Uso

torsocks se utiliza como un comando desde un terminal (o ventana del sistema, línea de órdenes, etc.), en conjunción con el programa de monitorización que necesitemos.

Son muy conocidos y utilizados `cURL` y `wget`.

wget

Empezando por el segundo, *wget*, su finalidad es descargar el contenido de un URL (típicamente una página web, pero en realidad cualquier tipo de recurso: PDFs, imágenes, videos, música, etc.), *wget* trabaja con protocolos *http*, *https* y *ftp*; permite autentificación y, entre otras cosas, hacer descargas recursivas. Una descarga recursiva comienza por descargar el URL que se le indique, extrae los links que éste tenga en su interior, descarga los URLs enlazados y así sucesivamente; esto permite descargar de forma automática sitios enteros.

Por ejemplo:

```
torsocks wget
http://b7ehf7dabxevdsm5szkn2jecnliwzoxlsn4lijxqxikrlykbbsfrqfad.onion/user/Annafox/answers
```

desargará esa página, pero

```
torsocks wget -r
http://b7ehf7dabxevdsm5szkn2jecnliwzoxlsn4lijxqxikrlykbbsfrqfad.onion
```

descargará el sitio completo.

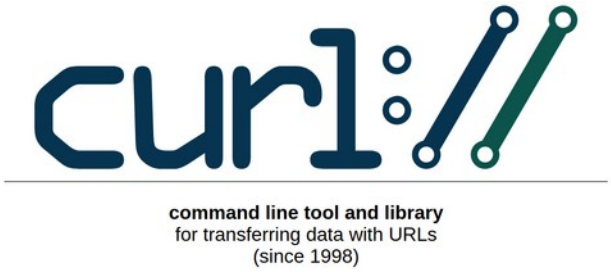
wget es un clásico en el mundo Unix/Linux, pero hay también versiones para Windows y Mac.

(https://es.wikipedia.org/wiki/GNU_Wget, https://www.gnu.org/software/wget/)

curl

El otro programa citado *cURL* es también otro clásico. Está pensado para intercambiar información entre un ordenador y un servidor. Intercambiar va más allá de la simple descarga, de manera que, por ejemplo, permite dialogar con APIs de aplicaciones web. Es, en este sentido, bastante más versátil que el *wget*.

Su página web está en https://curl.se/ y es una utilidad que viene de serie en todos los Linux y también en las versiones más recientes de Windows.



wget, *curl* y otros programas, al ser simples comandos con sus correspondientes parámetros, son fácilmente automatizables al poder ser incluídos en *scripts* que, por ejemplo, los lancen en intervalos específicos de tiempo contra varias (muchas, tal vez) direcciones de servicios Tor, guardando el contenido devuelto por tales servicios o entregándoselo a a otros programas para analizar automáticamente su contenido.

torsocks y direcciones de internet de superficie

Aunque *torsocks* nos permite conectar con sitios y servicios de la red Tor, podemos usarlo también para conectar con sitios “normales”, de la red de superficie.
Por ejemplo:

```
torsocks wget http://www.usal.es
```

nos descargará la página principal del web de la Universidad de Salamanca. La diferencia está en que, al lanzarse *wget* a través de *torsocks*, el servidor web de la USAL no registrará nuestra IP, sino la

del nodo de salida de Tor que haya correspondido en ese momento. Dicho de otra manera, anonimizará nuestra conexión al servidor de la USAL.

Es posible que hacer anónima una simple descarga no nos emocione mucho, pero consideremos el uso de *torsocks* y otro clásico de la monitorización de redes: ***nmap***.

nmap (https://nmap.org/) es un escaneador de puertos; es decir, un programa que intenta múltiples y variadas conexiones a uno (o muchos) servidores y a través de esas conexiones (y de las respuestas de rechazo que recibe) hace un análisis exhaustivo de las conexiones disponibles en uno (o muchos) servidores. Uno de sus usos es, obviamente, descubrir agujeros por donde colarse indebidamente en servidores; los servidores pueden ser propios o ajenos y, en este último caso, estaremos cometiendo un acto claramente ilícito.

```
Starting Nmap 7.80 ( https://nmap.org ) at 2024-05-16 10:31 CEST
Nmap scan report for xxxx.xxx.xxx.xx (xxx.xxx.xxx.xx)
Host is up (0.0017s latency).
Not shown: 990 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
110/tcp  open  pop3
111/tcp  open  rpcbind
113/tcp  open  ident
995/tcp  open  pop3s
8009/tcp open  ajp13
8080/tcp open  http-proxy
9102/tcp open  jetdirect
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 3 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 1.77 seconds
```

Ejemplo de uso de nmap sobre un único servidor. Pueden verse los puertos abiertos, los servicios que están escuchando en cada puerto, y el sistema operativo del servidor.

Acceder a la red Tor a través de un lenguaje de programación

Los lenguajes de programación disponen de módulos, librerías, funciones, etc. que permiten escribir programas que acceden a internet, se conectan a servidores, descargan y suben cosas, etc.. Obviamente, cualquier programador puede escribir un programa que monitorice sitios, servicios, contenidos y demás a la carta.

Mediante *torsocks* y las tecnologías *proxy* es posible utilizar esas mismas librerías o módulos para escribir programas que monitoricen sitios o servicios de la red Tor. Por ejemplo, con el lenguaje de programación *Python* es habitual utilizar el módulo *requests* para acceder a la Internet "normal".

Así, el siguiente script descargaría un URL determinado:

```
import requests
URL='https://www.usal.es'
respuesta=requests.get(URL)
print(respuesta.status_code)
print(respuesta.raw)
```

Para hacer lo mismo en la red Tor, es preciso tener instalado el software Tor y decirle a la librería *requests* que las conexiones debe hacerlas a través de ese *proxy*.

Los *proxies* en general operan en un ordenador con una IP (en el caso más simple nuestro ordenador o *localhost*) y a través de un puerto, que en el caso del software Tor suele ser el 9050. Ambos parámetros, *host* y puerto, se ajustan en el archivo de configuración `torrc`; aunque sus valores por defecto son, como de ha dicho l*ocalhost* y 9050 respectivamente.

Así, el *script* anterior, válido para la internet de superficie, modificado como sigue, permitiría conectar con un servicio o sitio Tor:

```
import requests
proxy={}
proxy['http']='socks5h://localhost:9050'
proxy['https']='socks5h://localhost:9050'
URL='https://bible4u2lvhacg4b3to2e2veqpwmrc2c3tjf2wuuqiz332vlwmr4xbad.onion/'
respuesta=requests.get(URL, proxies=proxy)
print(respuesta.status_code)
print(respuesta.raw)
```