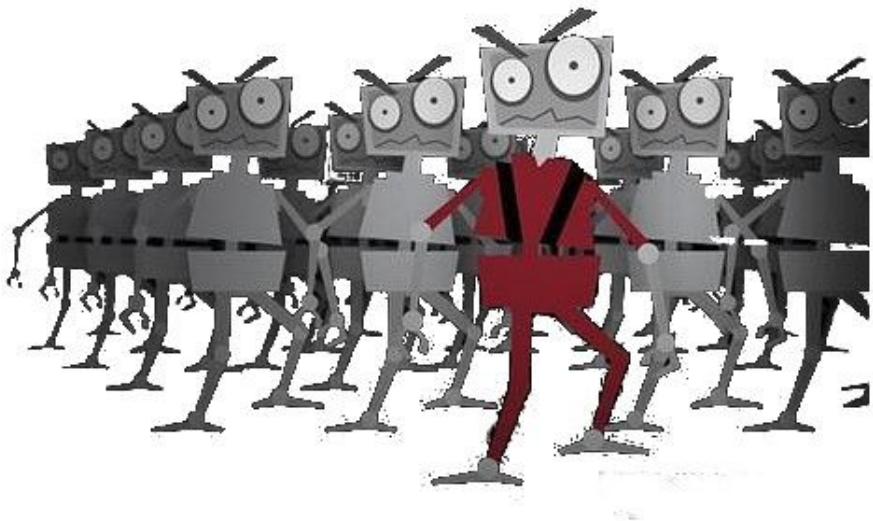
ESCOLA DE ENXEÑARÍA DE TELECOMUNICACIÓN



BOTNETS

Seguridad de Sistemas Informáticos
Curso 2016/2017

Autor - Alejandro Picos Prieto
Coautor - Bruno Muñiz Places

1 - Introducción

1.1 - Definición de botnet:

El término botnet hace referencia a un grupo de PCs infectados y que son controlados por un “master” de forma remota. Generalmente un hacker o un grupo de ellos crea una botnet usando un malware que infecta a una serie de equipos (llamados bots o zombies). Mediante este software el “bot herder” o “botmaster” controlará al equipo de forma remota para que realice una serie de acciones de las cuales no es consciente.

Figura 1. Dibujo ilustrativo de una botnet[1]

1.2 - Usos y aplicaciones:

- **Reclutamiento:** Una parte esencial en toda botnet es la de propagarse e infectar otros equipos. De cuantos más bots disponga una botnet, más efectivas serán sus acciones.

- **DDoS (Distributed Denial of Service)[2]:** Es un ataque con el que se pretende inhabilitar un servidor, un servicio o una infraestructura sobrecargando el ancho de banda del servidor o acaparando sus recursos hasta agotarlos. Esto se consigue mediante el envío simultáneo de multitud de peticiones desde múltiples puntos de la red.

- **Spam:** Se nombra así al envío de forma masiva de correos electrónicos no solicitados ni deseados por el destinatario. Para el receptor,estos mensajes suelen provenir de una dirección de correo electrónico desconocida y generalmente son enviados con motivos publicitarios.

- **Phishing email[3]:** Son correos enviados por estafadores con el propósito de recopilar información personal (números de cuenta, nombres de usuario y contraseñas, etc). En principio el email puede parecer que proviene de una entidad confiable, suplantando el tema y logos de dicha entidad, pero finalmente solicitan que ingreses datos sensibles en páginas web maliciosas.

- **Exposición de vulnerabilidades:** Mediante el malware localizado en un bot, se puede escanear dicho sistema para encontrar vulnerabilidades como puertos abiertos y exponerlas al máster para su posterior explotación.

- **Backdoors:** Otra posible funcionalidad de una botnet es la búsqueda de puertas traseras dejadas por otras aplicaciones maliciosas o la instalación de sus propios backdoors que seguirán en el sistema pese a la posible desinstalación del bot.

- **Password cracking y fuerza bruta:** En esta funcionalidad se usaría parte de la capacidad de cómputo y procesamiento de un bot para obtener acceso no autorizado a un sistema, mediante intentos repetitivos para adivinar la contraseña.

- **Ransomware:** Una de las funcionalidades más perjudiciales para el usuario. Mediante esta funcionalidad, el malware cifra sin autorización los archivos del sistema para luego pedir una recompensa por su descifrado.

- **Minería y robo de Bitcoins[4]:** Se trata de aprovecharse de la capacidad computacional del bot para procesar las solicitudes de transacciones de toda la red bitcoin y recopilar una lista de las transacciones válidas.

Hasta ahora sólo hemos mencionado los usos maliciosos de una botnet, a continuación enumeramos algunos de sus usos bien intencionados:

- **SETI (Search for Extraterrestrial Intelligence)[5]:** Es un programa el cual se encarga de la búsqueda de vida extraterrestre mediante radiotelescopios para escuchar anchos de banda del espacio. Estos datos registrados tienen constan de mucho ruido, y al contar con la capacidad de cómputo de los bots, la limpieza y aprovechamiento de la señal se haria mas rapidamente.

- **GPUGRID.net[6]:** es un proyecto de computación distribuida que realiza simulaciones de biología molecular a átomo-entero.

1.3 - Estrategias de defensa:

Las medidas de prevención contra posibles infecciones no son diferentes a las usadas contra otro tipo de malware: uso de antivirus y cortafuegos, contraseñas largas y complejas, navegar a través de páginas seguras, análisis frecuentes con herramientas antimalware,
mantener el sistema operativo actualizado, etc.
Generalmente las infecciones se producen a través de phishing o de spam, por lo que hay que tener especial precaución con los correos que abrimos y las páginas web por las que navegamos, ya que es la mejor forma de prevención.

1.4 - Deteccion y eliminacion:

Es relativamente difícil detectar una infección de este tipo ya que suelen pasar desapercibidas por el usuario al ser tan “silenciosas”, salvo en casos de ransomware o de algún tipo de malware que pueda afectar al funcionamiento del equipo infectado. Un ejemplo de esto puede ser en los casos en los que la computadora esté siendo usada para enviar spam o saturar un sitio web. Estos comportamientos pueden ser percibidos por el usuario al notar cambios en la velocidad de internet y que no son producidos por una acción premeditada del usuario.

Otras estrategias para descubrir si un equipo está infectado sería el uso de algún tipo de herramientas como un buen producto anti-malware. Para usuarios con mayores conocimientos técnicos, observando los procesos que corren en el sistema y los programas instalados, pueden llegar a revelar la presencia de una infección. De todos modos, no siempre es tan fácil determinar la presencia de una botnet.

Si se da el caso de detectar una infección, uno de los primeros pasos que puede llevar a cabo el usuario sería bloquear con algún tipo de firewall las IP por las que se reciben las órdenes, pero estas en ocasiones no son fáciles de detectar o el bloqueo puede afectar a otras aplicaciones (p. ej. aplicaciones IRC). De todas formas, esta nunca sería la mejor opción, ya que el equipo podría infectar a otros, e incluso podría llegar a ser usado por otra botnet más tarde al no eliminar la infección.
En caso de detectar que un equipo está infectado por una botnet, es importante reportarlo, tanto a la policía como a organizaciones dedicadas a la seguridad informática como por ejemplo la OSI[7].

1.5 - Historia de las botnets más conocidas[8] [9]:

1.5.1 - GM

El año siguiente a la invención de IRC por Jarkko “WiZ”, el primer bot conocido vio la luz de manos de Greg Lindahl. Este bot jugaría al juego Hunt the Wumpus con usuarios IRC.

A partir de esto, otros programadores se dieron cuenta de que podían crear bots para realizar las tareas de un usuario. Uno de los usos más importantes que se le dió fue para mantener un canal abierto y evitar que usuarios maliciosos pudieran asumir el control del canal. Para ayudar al operador IRC, los bots necesitan ser capaces de funcionar como un operador del canal. Los bots pasaron a ser el código que gestiona y ejecuta los canales de IRC, así como el código que ofrece servicios para todos los usuarios. Algunos servidores de IRC y bots comenzaron a ofrecer la capacidad de hacer cuentas shell del sistema operativo disponible para los usuarios. La cuenta shell permite a los usuarios ejecutar comandos en el host del IRC.
En la actualidad, se ha utilizado GM Bot para el desarrollo de Mazar Bot, el cual afecta a Android y borra el contenido de los dispositivos.

1.5.2 - GT Bot (Global Thread Bot)

Este bot se hacía pasar por un cliente mIRC, con capacidad para ejecutar scripts como respuesta a ciertos eventos en el servidor IRC. Soporta conexiones UDP y TCP.

Capacidades del bot:

- Port Scanning: escaneo de puertos abiertos.
- Flooding: ataques DDoS.
- Cloning: clonar una conexión a un servidor IRC que sustituya la conexión original.
- BNC (Bounce): anonimizar el acceso a un servidor de un cliente Bot (como un proxy).

Hoy en dia todas las variaciones de bots basados en mIRC se consideran miembros de la familia GT Bot.

1.5.3 - SDBot

Este bot fue escrito en C++ por un programador ruso en 2002. Este bot fue muy importante debido a que su autor liberó el código y publicó una Web e información de contacto. Esto lo hizo muy accesible y además es muy fácil de modificar y mantener.

La característica más importante del SDBot es el uso del control remoto de backdoors. Si un exploit tiene éxito, el gusano crea un script y lo ejecuta para descargar el SDBot en la nueva víctima (los archivos binarios solo ocupan 40 KB). Haciendo uso de Remote Access Trojan (RAT), que es un componente integrado en el SDBot, se conecta al servidor IRC a la espera de instrucciones de un “bot herder” o “botmaster”. Se ejecuta en segundo plano y es invisible para el usuario. Se apodera de información importante como el nombre de usuario
y la contraseña.

1.5.4 - Agobot

Agobot, también conocido como Gaobot, tiene como característica principal su diseño en módulos. Agobot tiene concretamente tres módulos:

1. Módulo inicial, que contiene el cliente bot IRC y el acceso remoto a las backdoors.
2. Módulo 2, ataca y desactiva el antivirus.
3. Módulo 3, evita que el usuario acceda a una lista de páginas web, normalmente páginas web de antivirus.

Esto permite actualizar un módulo con nuevas técnicas mientras se ejecutan los demás. Agobot utiliza IRC para el C&C (command and control), pero se propaga usando P2P.

Algunas capacidades de Agobot:

- Escaneo de vulnerabilidades.
- Ataques DDoS.
- Busqueda de CD keys de juegos.
- Inhabilitar el antivirus y monitor de procesos.
- Se oculta usando tecnología rootkit.

1.5.5 - RBot:

La familia RBot es una de las más penetrantes y complejas que hay, en Junio de 2006 fue el más detectado en sistemas Microsoft con 1.9 millones de PCs infectados. Fue una de las primeras botnets en usar algoritmos de compresión o cifrado. Usa diferentes métodos para encontrar vulnerabilidades. Escanea sistemas en los puertos 139 y 445 (sistemas Microsoft con puertos abiertos para la compartición de ficheros) y prueba a acceder con contraseñas débiles, pudiendo usar una lista por defecto o una lista proporcionada por el “botmaster”.

1.5.6 - Shuabang botnet[10]:

En Septiembre de 2014 se dio a conocer esta nueva Botnet, que infectaba a los terminales móviles para hacer BlackSEO (Uso de técnicas fraudulentas para engañar a un buscador) orientado a conseguir valoraciones o descargas falsas (denominado BlackASO). Esta Botnet solo infectó terminales móviles en India, Brasil y Rusia. Además el usuario final no era consciente de que su terminal estaba infectado.

La infección se realiza a través de apps fraudulentas, el creador tenía miles de cuentas para asociar a los dispositivos móviles una vez asociadas las cuentas conseguía los tokens y realizaba acciones de Click-Fraud. En cuanto a la descarga remota de aplicaciones, parece
que no lo había conseguido, pero era un objetivo principal tras estudiar su código.

1.5.7 - Mirai[11] [12] [13]:

El botnet Mirai se centra principalmente en dispositivos que conforman el Internet de las cosas (IoT). En Octubre de 2016 fue el culpable de un ataque DDoS a Dyn, uno de los proveedores de DNS más importantes del mundo, dejando sin Internet a gran parte de EE.UU., especialmente en la costa Este.
Mirai incluye una tabla de máscaras de red a las cuales no infecta para retrasar su detección, dentro de las que se encuentran redes privadas y direcciones pertenecientes al Servicio Postal de los Estados Unidos, el departamento de Defensa, IANA, Hewlett-Packard y General Electric.
Algunos sitios web afectados: PayPal, Amazon, Twitter, Netflix, Spotify, Reddit, GitHub, SoundCloud…
En sus ataques DDoS llegó a generar un tráfico de hasta 665 Gbps

1.6 - Contratación de una botnet:

Aunque a la hora de hablar de una botnet se puede pensar en ella como un programa hecho y destinado a la explotación por parte de su/sus creadores, la realidad es muy diferente, las botnets se venden/alquilan al mejor postor. Contrario a lo que se puede pensar, no se necesita tener conocimientos sobre la Deep Web ni nada del estilo, con una simple búsqueda en Google es suficiente. Simplemente con tener una cuenta PayPal, mala voluntad hacia un objetivo e intención de quebrantar la ley, un usuario puede conseguir una botnet mediante la cual hacer realidad sus malas intenciones. Y lo peor de todo, es que sus precios están al alcance de muchos[14].

1.7 - Consecuencias legales:

Las consecuencias sobre el uso de las botnets dependen del uso que se les dé. Poniendo un ejemplo, en el caso de ataques Ransomware o DDoS, según la Ley Orgánica 5/2010, de 22 de junio, por la que se modifica la Ley Orgánica 10/1995, de 23 de noviembre del Código Penal, la redacción del Art. 264[15]:

1) “El que por cualquier medio, sin autorización y de manera grave borrase, dañase, deteriorase, alterase, suprimiese, o hiciese inaccesibles datos, programas informáticos o documentos electrónicos ajenos, cuando el resultado producido fuera grave, será castigado con la pena de prisión de seis meses a dos años”.

2) “El que por cualquier medio, sin autorización y de manera grave obstaculizara o interrumpiera el funcionamiento de un sistema informático ajeno, introduciendo, transmitiendo, dañando, borrando, deteriorando, alterando, suprimiendo o haciendo inaccesibles datos informáticos, cuando el resultado producido fuera grave, será castigado, con la pena de prisión de seis meses a tres años”.

2 - Características de una botnet

2.1 - Command Control

El Command and Control, como bien indica su nombre es el encargado de comandar y controlar el comportamiento de los bots. Es la interfaz usada por el botmaster para ordenar a sus bots que hagan algo, por lo cual es el punto mas critico de la botnet. Si el C&C es intervenido, la botnet se vuelve inservible.

Existen tres tipos básicos de C&C[16]:

- Centralizado: Es el más común de todos, todos los miembros de la botnet se conectan a este servidor central mediante métodos de pulling o pushing.

- Descentralizado: Surge para mitigar la posible pérdida de control de la botnet si el C&C centralizado es intervenido, los nodos (o gran parte de ellos) actúan tanto de servidor como de cliente lo que permite al botmaster controlar la red desde cualquier máquina comprometida. Esta estructura descentralizada se conoce también como red peer-to-peer (P2P) .

- Híbrido: El modelo híbrido mezcla características de las botnets centralizadas y descentralizadas o distribuidas. Utilizan P2P como principal centro de mando, pudiendo controlar la red desde cualquier máquina infectada. Cuando la conexión de un bot hacia sus iguales falla, éste se conectará a una estructura C&C de backup centralizada.

2.1.1 - Seguridad de C&C

La mayor fortaleza de una botnet es su C&C, pero también es su mayor debilidad. Proteger la infraestructura, comunicaciones y acceso debe ser primordial y esto puede llevarse a cabo a través de las siguientes técnicas, entre otras:

- **Hosting a prueba de balas (Bulletproof hosting)**[17]

Existen compañías de hosting que ofrecen términos flexibles para sus servicios y no restringen a sus usuarios en la utilización de las máquinas contratadas, es decir, tienen una moralidad algo laxa.
Estos proveedores tienen reputación de no cooperar o responder a las fuerzas de la ley y sus instalaciones suelen estar en países con leyes menos restrictivas para el uso de internet.

- **DNS dinámicos (DDNS)**

Los servicios de DNS dinámicos permiten asociar a un nombre de dominio un rango de direcciones IP variante. Esto permite que el mismo nombre de dominio apunte al mismo host a pesar de que este cambie constantemente de IP dentro del rango que asigna la ISP del hosting.

- **Flujo IP (IP Flux)**[18]

También conocido como IP Flux, es la denominación que se le da a la técnica de asociar un dominio o subdominio una dirección IP variante. Las peticiones DNS realizadas para el dominio responderán con una lista de posibles IP distribuidas geográficamente y de ISP diferentes, en vez de una única dirección. Existen dos tipos: Single flux y Double Flux

- **Flujo de Dominios (Domain Fluxing)**[18]

Los bots son programados para, en caso de no poder contactar con el C&C (Por ejemplo si se ha bloqueado el acceso al dominio o a las IP que llevan como parte de su configuración), generar patrones de nombres de dominio hasta encontrar de nuevo un C&C operativo en alguna de ellas.

2.2 - Los Bots

Como se mencionó anteriormente, las botnets pueden realizar una serie de ataques. Estos ataques son muy indicados para una botnet ya que se aprovechan de la mayor cualidad de esta, la cantidad de máquinas. La fuerza de una botnet reside en la cantidad masiva de máquinas que posea, por ello, estas tareas que sólo son efectivas haciendo uso de una enorme colección de máquinas, son las más utilizadas para realizar ataques:

- **Ataques de denegación de servicio distribuidos (DDoS)**

Cuantos más participantes se unan al ataque mayor será su efectividad a la hora de tirar abajo un objetivo.

- **Fraude de clicks (Click-Fraud)**

Los bots pueden ser instruidos para acceder y realizar clicks sobre anuncios publicitarios de cara a generar ingresos para el atacante.

- **Pasarelas de spam (spambots)**

Utilizar una botnet como pasarela de correos de spam produce grandes ventajas con respecto a los métodos tradicionales, ya que al tener más equipos mandando mensajes, se incrementa exponencialmente el número de destinatarios.

- **Agente de pago por instalación**

Los bots pueden ser instruidos para descargar e instalar software legítimo o malicioso en las máquinas infectadas, produciendo ganancias para el sitio de descarga.

A fin de cuentas, para el botmaster la botnet representa para él una infraestructura de computación en la nube y los bot pueden estar preparados para cualquier operación coordinada, por ejemplo, puede utilizarse toda esta potencia de computación para crackear hashes de contraseñas de forma distribuida a velocidades que sólo se conseguirían en centros de procesamiento de datos.

3- Command and Control (C&C Server)

3.1 - Diseño

Siendo nuestro objetivo el desarrollar una botnet propia con carácter académico, nos enfrentamos a múltiples decisiones en cuanto al diseño de la infraestructura de control y comunicación.

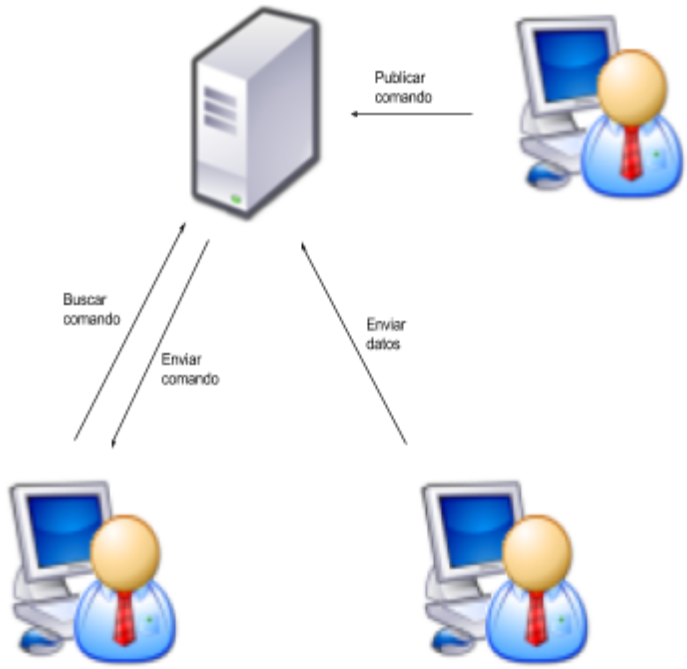
Tras un estudio inicial de la organización de algunas de las botnets actuales más conocidas pudimos observar una amplia gama de tipologías que abarcan desde la conexión a un simple canal de IRC hasta redes distribuidas P2P o sistemas híbridos que combinan topologías en estrella con jerárquicas. Todas ellas reforzadas con estrategias de ocultación, multiservidores, IP-Fluxing, Doman-Fluxing y en general una larga lista de técnicas que intentan, según las necesidades individuales de cada botnet, conseguir un equilibrio entre el control con baja latencia y la resistencia de la red a ser secuestrada o intervenida (Sobra decir que la ocultación de quien se encuentra detrás del C&C también suele ser una prioridad).

El desarrollo del C&C y el Bot van de la mano, es por ello que la primera decisión que tomamos fue plantear el escenario de infección y la topología. Como explicamos en apartados anteriores, hoy en día existen botnets que reúnen toda clase de máquinas y dispositivos, no es lo mismo desarrollar un Bot planteado para infectar dispositivos IoT, servidores Unix o máquinas personales, por ejemplo MS Windows. Como nuestro desarrollo sólo toca las bases técnicas de un botnet tradicional, evitaremos centrarnos en un escenario concreto.

Si estamos ante el caso de máquinas personales, el mayor inconveniente que encontramos a la hora de decidir la topología de la red es, a parte del tiempo de desarrollo y experiencia de la que disponemos, la complejidad que suponemos pueden llegar a tener las máquinas infectadas, y por ende los bots, para recibir conexiones entrantes en esta clase de escenario.
Las máquinas personales suelen residir detrás de gateways domésticos cuyos firewalls, en la gran mayoría de casos, no permitirán comunicarnos con servicios levantados en su red interna. Por lo tanto la comunicación se podrá realizar principalmente de dos maneras distintas: vía Polling o vía Pushing.

3.1.1 - Pulling

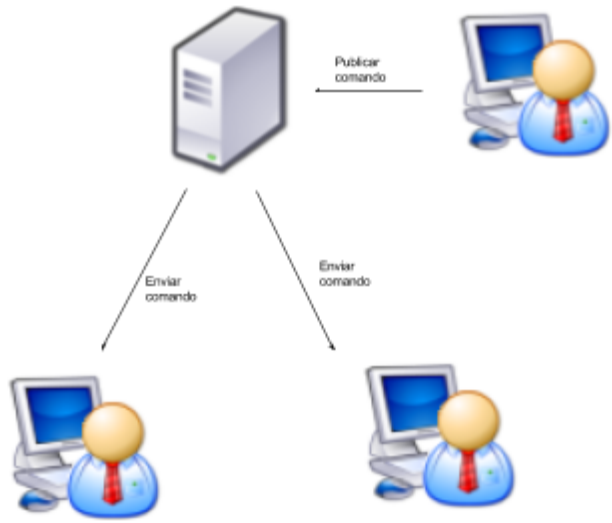
En este caso, ya que son los bots los que pueden realizar conexiones hacia el exterior, serán ellos los encargados de mantenerse al día. Por ejemplo mediante tecnologías Web, consultando los bot con una determinada frecuencia al C&C para recuperar nuevas órdenes o comandos o incluso enviar ellos mismos datos.



Pros	Contras
• Implementación relativamente sencilla • Dificultad para analizar el tamaño de la botnet ya que los bots no se conocen	• Todos los bots conocen las direcciones del C&C, punto de detección. • Proporciona poca ocultación para el master. • Alta latencia en la recepción de los comandos. • Fácil de detectar a causa del tráfico generado en intervalos de tiempo.

3.1.2 - Pushing

En el método pushing, es el servidor el que envía las órdenes directamente a cada bot o subconjunto de bots, el escenario presenta el inconveniente de las conexiones entrantes mencionado anteriormente, ya que la comunicación de red no puede ser directa entre C&C y Bot, es necesario un canal donde el bot pueda recibir los mensajes. Aquí es donde entra en juego el protocolo IRC (Internet Relay Chat), que tanto se utilizó en las primeras generaciones de botnets.
El protocolo IRC permite que los bots se conecten a un canal compartido donde recibir las órdenes sin una estricta comunicación de red punto a punto con el C&C.



Pros	Contras
● Los bots ya no son activos, simplemente escuchan el canal a la espera de órdenes. ● La comunicación con los bots cuenta con poca latencia.	● Facilidad para intervenir toda la red a través del canal de comunicación. ● Sigue siendo centralizado y sería necesario subdividirla jerárquicamente para dotarla de mayor seguridad. ● Se vuelve más sencilla la obtención del tamaño al observar el canal de comunicación.

Nosotros nos hemos decidido por construir nuestra botnet sobre un canal IRC.
En nuestra implementación, creamos un canal IRC en un ordenador local, cuya IP es conocida por los bots, los cuales se conectan para recibir instrucciones.

4 - El bot

4.1 - Descripción de la estructura

El bot desarrollado se compone de dos clases, la primera (ChatClient) se encargará de gestionar las conexiones, y la segunda (Actions) de los ataques.

La primera clase es la encargada de escuchar en el canal IRC, y de ser necesario enviar mensajes en este. Ya que nuestro entorno va a ser en local y no con un servidor, la IP del canal puede cambiar, por lo que se debe lanzar la clase ChatClient indicándole la ip como parámetro de entrada.

La funcionalidad básica que implementa es la de escuchar el canal, comprobar si el mensaje es enviado por un master autorizado, y de ser así, ejecutar la acción indicada. Para facilitar el testeo y visualización sencilla de nuestro proyecto, la comunicación va en claro, sin encriptar mensajes, y la autenticación es simplemente la comprobación de una contraseña enviada.

Esta clase principal también tiene una funcionalidad “oculta”. Si al ser lanzado el bot, se le indica por parametros, ademas de la ip del servidor IRC una contraseña de master, este bot entra en modo master, desplegándose una ventana de chat para poder enviar mensajes al canal. En caso de no ser lanzada como master, también despliega una ventana de chat, para visualizar los comandos recibidos. En una implementación real, en la que no haya que depurar ni testear comandos, esta ventana solo sería mostrada a aquellos bots que se autoricen como masters.

Para mejorar el anonimato de nuestros masters, al enviar un comando, ellos también lo reciben y ejecutan el ataque, para así, en caso de una intervención, poder excusarse al hacerse pasar por otro bot más.

4.2 - Funcionalidades

Estas son las distintas funcionalidades básicas con las que cuenta nuestro bot, así como los comandos necesarios para ordenarlas.

Ataque UDPFLOOD
usernameMaster@*udpflood-destinoAtaque-duracionEnMilis

Ataque TCPFLOOD
usernameMaster@*tcpflood-destinoAtaque-port-numeroThreads-duracionEnMilis

Ataque HTTPFLOOD
usernameMaster@*httpflood-destinoAtaque-duracionEnMilis

Ataque SLOWLORIS
usernameMaster@*slowloris-destinoAtaque-numeroThreads-delay

Funcionalidad ADD MASTER
usernameMaster@*newmaster-nombreNewMaster

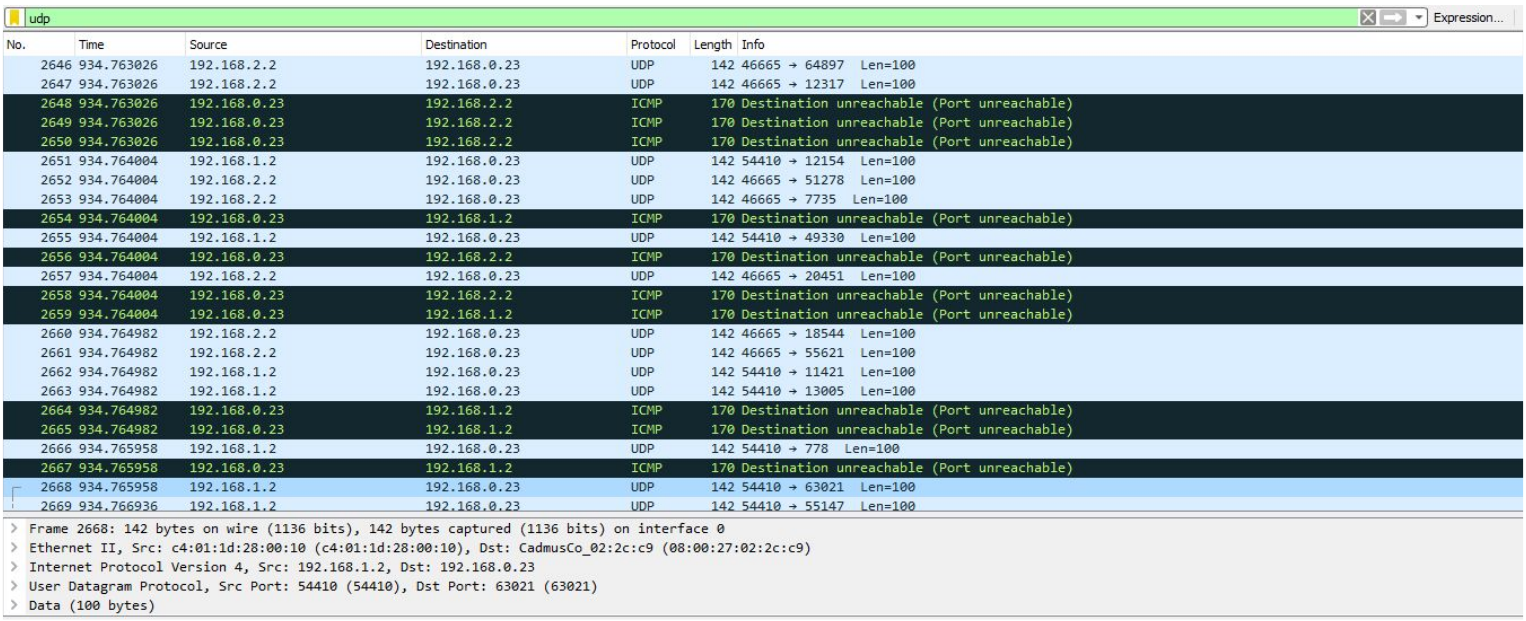
Funcionalidad DELETE MASTER
usernameMaster@*delmastermasterAEliminar

En nuestro sistema, solo un master puede añadir o eliminar a otro master. Esto se puede ver como un punto de riesgo, ya que si un defensor consiguiese añadirse como master, podría hacer grandes daños a nuestra red, pero también puede verse como un punto fácilmente controlable, ya que una rápida intervención por un master legítimo puede frenar esta contramedida de un defensor.

5 - Ejemplos de ataques realizados

5.1 - Flood UDP

Un flood UDP es un ataque de denegación de servicio que utiliza UDP.
Para conseguir inhabilitar el servidor, los bots generan una gran cantidad de paquetes UDP con destino a puertos aleatorios del servidor, como resultado de esto, el servidor comprueba la aplicación escuchando en ese puerto, ve que no hay ninguna aplicación escuchando en ese puerto y respondiendo con un paquete ICMP de destino no alcanzable (Destination Unreachable).
Debido al gran número de paquetes, la víctima se verá forzada a enviar también una gran cantidad de paquetes ICMP con lo que, a la larga, el servidor no podrá ser accedido por otros clientes.



Esta imagen muestra los paquetes enviados desde dos bots (192.168.1.2 y 192.168.2.2) hacia el servidor (192.168.0.23). Se puede observar que los puertos destino van variando continuamente y también los mensajes ICMP de Destino Inalcanzable que se esperan de este tipo de ataque.

Este tipo de ataque tiene como posible solución instalar Firewalls en puntos de la red para filtrar puertos no deseados. Con esto se consigue que la víctima no responda a paquetes UDP no deseados.
Un inconveniente de esta solución es que el Firewall tiene un número máximo de sesiones activas, con lo que también es susceptible de ataques por flood.

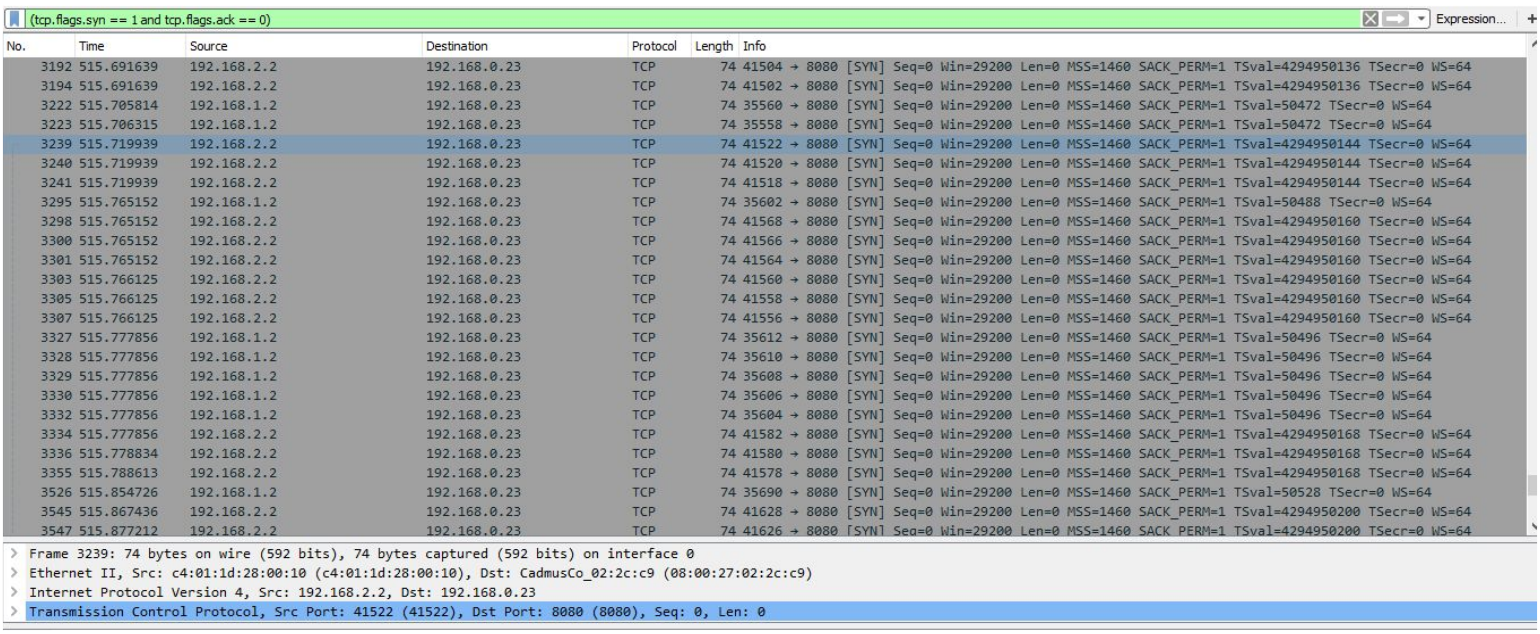
5.2 - Flood TCP (SYN flood)

Un flood TCP es un ataque de denegación de servicio que utiliza TCP. En este, los atacantes mandan una sucesión de peticiones SYN a la víctima en el intento de agotar sus recursos y conseguir que la víctima no pueda atender al tráfico de sus clientes legítimos.

Esto se consigue debido a que para comenzar una conexión con el protocolo TCP, el cliente y el servidor deben intercambiar varios mensajes (negociación en tres pasos):

1. El cliente pide una conexión mandando un mensaje (SYN) al servidor.
2. El servidor asiente esta petición mandando un mensaje SYN-ACK al cliente.
3. El cliente responde con un mensaje ACK al servidor y se establece la conexión.

Sin embargo, en el flood TCP, los clientes maliciosos pueden no mandar el ACK esperado o falsifican (spoofing) su ip de destino para que el servidor se quede esperando por el mensaje ACK consumiendo recursos que se podrían emplear en conexiones legítimas.



Esta imagen muestra los paquetes enviados desde dos bots (192.168.1.2 y 192.168.2.2) hacia el servidor (192.168.0.23:8080). Se puede observar la gran cantidad de paquetes SYN con puerto de destino 8080 y longitud 0.

Este tipo de ataque tiene como posible solución, en la máquina del servidor, el uso de SYN Cookies [19].

La técnica SYN Cookie permite al servidor no tirar conexiones cuando la cola se llena. Esto se consigue mandando correctamente el mensaje SYN-ACK pero descartando la entrada SYN en la cola y comprime los datos más básicos de la conexión en el número de secuencia usado en el SYN-ACK. Cuando recibe el ACK descomprime la información básica comprimida anteriormente con lo que puede establecer la conexión correctamente.

6 - Posibles Mejoras

Nuestra botnet planteada sirve de ejemplo de las posibilidades que brindaría una botnet real. Debido a todo el trabajo necesario para construir una botnet compleja y dado el tiempo del que dispusimos, la nuestra es un ejemplo muy simplificado.
Al ser este el caso, las mejoras que podríamos implementar en nuestro sistema son innumerables, por nombrar algunas:

- **Propagación:** Dotar al bot con la habilidad de propagarse e infectar otras máquinas. Las estrategias y métodos dependerán de muchos factores, algunos podrían ser: Spam con archivos maliciosos, escaneo y análisis de vulnerabilidades, fuerza bruta, diccionarios de contraseñas por defecto…
- **Empaquetado/Ocultación:** Empaquetar al bot en un rootkit de forma que redirija llamadas del sistema y oculte otros procesos en curso de los ojos de los administradores sería otra mejora deseable para ocultar nuestro bot.
- **Cifrado de comunicaciones:** Podríamos cifrar nuestros mensajes por el canal para añadir una capa de seguridad y dificultar la detección de nuestros masters.
- **Desplegar el canal IRC en un servidor online externo:** Haciendo esto, evitaríamos el problema actual de cambio de la dirección ip, y la necesidad de tener el canal IRC en un ordenador personal, presumiblemente perteneciente a alguno de los cabecillas de la botnet.
- **Spoofing:** Para ofrecer una capa más de ocultación a nuestra botnet y hacer que sea más difícil de detectar, podríamos substituir el campo ip origen de la cabecera ip de los mensajes enviados en los ataques por un rango de ips aleatorias. Al hacer esto, si alguien intenta descubrir el origen del ataque no podrá llegar a dar con los bots de la botnet.
- **C&C mediante DNS[20]:** Migrar nuestra botnet de un canal IRC a usar campos de una petición DNS para la distribución de comandos debido a que actualmente no existen métodos de detección diseñados específicamente para este método de C&C lo que implicaría una mayor probabilidad de no ser detectado.

- **Actualización de funcionalidades:** Se podría añadir una funcionalidad mediante la cual nuestro bot pudiese descargar un fichero con el que sustituir o completar el código fuente de los ataques para hacer a nuestra botnet más potente. Mediante este sistema, el bot podría irse actualizando version a version y en cada una añadir o mejorar las funcionalidades existentes.
- **Mecanismo anti exterminio:** Nuestro bot podría tener mecanismos para evitar el saneamiento de un sistema infectado. Mediante esta funcionalidad, el bot podría replicarse en varias ubicaciones de nuestro sistema. Cada uno de estos “bots fantasma” no estarían conectados al canal IRC, simplemente comprobarán periódicamente si los otros bots estan operativos, de no ser así, él se conectará al canal y actuaría como cualquier otro bot.

7 - Conclusiones

Tras realizar este trabajo, tanto a nivel práctico como a nivel teórico, hemos aprendido cómo se comportan las botnets y que, dejando a lado su programación, es bastante sencillo y se podría resumir en los siguientes pasos:

1. Búsqueda de vías de entrada.
2. Infección y propagación de clientes.
3. Comunicación de clientes con el C&C a la espera de órdenes.
4. Envío y recepción de comandos.
5. Ejecución de órdenes.

Estos cinco pasos podrían complicarse y extenderse tanto como se desea, tratando con redes de bots más complejas y con más funcionalidades, el límite en este campo es difícilmente acotable e imaginable.

Porque, ¿Quién le iba a decir a Greg Lindahl que un día la seguridad de una nación y el correcto funcionamiento de internet podrían verse comprometidos por neveras, persianas o termostatos?

Enlaces de interés.

[1] Imagen ilustrativa botnet: https://blog.kaspersky.es/que-es-un-botnet/755/
[2] Ataque DDoS: https://www.ovh.com/us/es/anti-ddos/principio-anti-ddos.xml
[3] Phishing: https://www.scamwatch.gov.au/types-of-scams/attempts-to-gain-your-personal-information/phishing
[4] Mineria bitcoin: https://www.oroyfinanzas.com/2015/02/que-mineria-bitcoin-por-que-necesaria/
[5] SETI: http://setiathome.ssl.berkeley.edu/
[6] GPUGRID: https://www.gpugrid.net/about.php
[7] OSI: https://www.osi.es/es/servicio-antibotnet/infocodigo
[8] History of botnets: http://wiki.cas.mcmaster.ca/index.php/Bots_%26_Botnets#History_of_Botnets
[9] Botnets The Killer Web App: https://books.google.es/books?id=4MAuVjOx6kIC&lpg=PA6&ots=vespO86AkZ&dq=botnet%20gm%201989&hl=es&pg=PP1#v=onepage&q&f=false
[10] ShuaBang botnet: http://www.elladodelmal.com/2014/11/shuabang-botnet-red-de-terminales.html
[11] Mirai (1): https://www.geektopia.es/es/technology/2016/10/11/noticias/un-hacker-publica-el-codigo-de-un-programa-para-hacer-ataques-ddos.html
[12] Mirai (2): https://es.gizmodo.com/confirmado-el-ataque-ddos-que-tumbo-la-red-en-eeuu-sur-1788100823
[13] Mirai (3): https://es.gizmodo.com/tras-el-ataque-a-dyn-una-botnet-de-mirai-ha-dejado-sin-1788555852
[14] Precio botnet: http://www.elladodelmal.com/2010/06/senora-que-llevo-la-botnet-barata.html
[15] BOE consecuencias: https://www.boe.es/diario_boe/txt.php?id=BOE-A-2010-9953
[16] Tipos de C&C: http://blog.segu-info.com.ar/2006/12/todo-acerca-de-las-botnets-i.html
[17] Bulletproof: https://krebsonsecurity.com/2013/01/inside-the-gozi-bulletproof-hosting-facility/
[18] Fluxing: https://www.damballa.com/downloads/r_pubs/WP_Botnet_Communications_Primer.pdf
[19] Flood TCP: http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-34/syn-flooding-attacks.html
[20] http://norbert-pohlmann.com/app/uploads/2015/08/279-On-Botnets-that-use-DNS-for-Command-and-Control-Prof-Norbert-Pohlmann.pdf