



Universidad Carlos III de Madrid - Escuela Politécnica Superior
Grado en Ingeniería Informática
Seguridad en dispositivos móviles

Grupo de investigación: Computer Security Lab

Android botnets for multi-targeted attacks

Autores:

Adán Cano Moreno. NIA: 100346105.
Jaime García González. NIA: 100346062.

Grupo: **81.**

24 de marzo de 2019

Índice general

1. Resumen	1
2. Introducción	2
3. Principales trabajos previos	4
4. Principales trabajos posteriores	5
5. Revisión crítica del artículo	6
6. Conclusiones	7

Capítulo 1

Resumen

El presente artículo expone el procedimiento a seguir para realizar un ataque *botnet* eficiente sobre diferentes dispositivos móviles *Android* al mismo tiempo para capturar información. El desarrollo del artículo se debe a que cada vez se está realizando un mayor número de ataques *botnets* a dispositivos móviles, debido a que estos tienen muchos sensores que son atractivos para los atacantes y a los que se puede acceder de manera sencilla a través de las aplicaciones. El principal objetivo del artículo es exponer el potencial que tiene este tipo de ataques y el peligro que puede suponer, puesto que puede ser aplicado para erradicar organizaciones criminales pero también puede ser usado por ciberdelincuentes. En el artículo se va mostrando un ejemplo de ataque *botnet* para mostrar la localización de diferentes dispositivos móviles.

Capítulo 2

Introducción

Un *botnet* se compone de dos partes: el cliente (que es a quien se ataca) y el servidor (que es quien realiza el ataque y quien recoge los datos). Al comienzo del artículo se muestra una breve introducción a los ataques a múltiples objetivos y la el por qué de realizarlo sobre dispositivos móviles. A continuación muestra una implementación del ataque donde se generan varias fases:

Recoger información : Como se puede recoger información de un dispositivo a través de la aplicación *Android*, que es el cliente del *botnet*. En esta fase se muestra un ejemplo de una modificación del código de una aplicación para poder leer datos del dispositivo a través de la geolocalización.

Almacenamiento y gestión de la información : En esta fase se describe el procedimiento a seguir para almacenar la información que ha sido recogida a través del ataque. Para ello se puede usar tanto *PHP* como *MySQL* ya que ambos están instalados en la mayoría de los servidores de internet. En el artículo se indica que para almacenar la información de cada dispositivo lo mejor es crear una tabla por dispositivo cuya clave indentificativa sea el número *IMEI*, ya que es un número único e identificador para cada dispositivo.

Mostrar la información : En este fase se describe brevemente como poder mostrar en un página la información que ha sido recogida. Para ello se necesita generar código dinámico a través de *Javascript* incrustado en código *PHP*. Para acceder a los datos almacenados se hace una consulta a la base de datos a través de el comando *SELECT* a la base de datos del dispositivo del que queramos recuperar sus datos. Todos los nombres de las tablas son almacenados en un array para poder hacer las consultas a la base de datos. Para mostrar la información en una web, se incluye la cabecera de *HTML* de una página web para inicializar la *API* de *Google Maps* y a través de *PHP* podemos mostrar diferentes iconos por cada *botnet* en la página web y así poder diferenciar la información.

Información de verificación : En objetivo de esta fase es conocer el comportamiento de los usuarios que están siendo atacados por un *botnet*. Se desea conocer si un persona infectada viaja en autobús, en tren o simplemente camina o está parado. Para ello se debe determinar la velocidad de movimiento del infectado. Por lo tanto, el problema de este ataque es conseguir suficiente información como para poder determinar el comportamiento de la persona a la que se ataca. Lo primero que se debe hacer es crear una nueva tabla donde, por cada instante de tiempo, realizar una verificación de la información capturada de diferentes dispositivos. Esto nos va a facilitar el proceso de gestión de los datos de diferentes objetivos a la vez. Cuando queramos consultar la información de verificación

debemos indicar que queremos mostrar la información de manera ordenada en función del instante de tiempo para poder analizar la información de manera eficiente.

Determinar puntos de encuentro entre dispositivos : En esta fase se describe el proceso para poder determinar el momento en el cual dos dispositivos pueden estar en mismo lugar. Para ello se hace uso del algoritmo de *k-means*. En artículo se muestra un ejemplo de implementación del algoritmo donde lo primero que se hace es seleccionar la información de los dispositivos que se desean analizar y almacenarlo en un cluster. El algoritmo *k-means* puede ser aplicado de manera recursiva mientras los clusters no cumplan con las especificaciones deseadas. Si cumplen con los requisitos la primera vez basta con aplicarlo en una sola ocasión. El algoritmo es desarrollado en *PHP* y necesita implementar una función que compruebe que el cluster cumple con los requisitos especificados. Los clusters seleccionados son concatenados en un array único para poder determinar de una manera más sencilla los puntos de encuentro. En la implementación mostrada en el artículo, los centroides iniciales son calculados de manera aleatoria dentro del rango de los máximos y mínimos de longitud y latitud observados en los datos. Se calculan a través de este método para asegurar buenos resultados. Los demás centroides son calculados a través de una ecuación de distancias de puntos sobre una esfera a partir de sus longitudes y latitudes. Antes de calcular los nuevos centroides se transforman los puntos de longitud y latitud al eje cartesiano. En cada iteración del algoritmo se calcula el cluster a partir de los puntos donde cada distancia entre los puntos calculados es menor a la distancia máxima. El atacante decide cuando dos víctimas están cercanas en función de un radio de distancia. Para cada cluster se selecciona los posibles puntos de encuentro en función del tiempo y la distancia. Esta es la razón por la cual almacenamos la información de verificación en función del instante de tiempo. De esta manera podemos analizar las zonas donde hay posibilidades de que sean puntos de encuentro. Si no hay ningún resultado viable, significa que los datos capturados no nos permiten determinar un punto de encuentro.

Capítulo 3

Principales trabajos previos

Mirar algunos de las referencias.

El algoritmo de k-means.

Capítulo 4

Principales trabajos posteriores

Capítulo 5

Revisión crítica del artículo

Capítulo 6

Conclusiones

Bibliografía

Información básica del autor

Información básica de la revista

Planificación del trabajo realizado

Glosario