

# Android botnets for multi targeted attacks

---

Jaime García González y Adán Cano Moreno

26 de abril de 2019

Universidad Carlos III de Madrid

Información del artículo

Contenido del artículo

Conclusiones

## Información del artículo

---

- Ingeniero de Seguridad informática de Sistemas por la ESIEA.
- En ese momento: Estudiante e investigador en ESIEA.
- Actualidad: FAMOCO
- Otros artículos: *Malicious URI Resolving in PDFs* (muestra como una petición HTTP desde un PDF puede ser una buena herramienta para un atacante)

- Filial francesa de Springer, fundada en 1986.
- Ámbito: medicina, matemáticas, estadística, informática e ingeniería.
- Otras actividades: publicación de libros del campo de las ciencias y editorial Springer en Francia.

## Contenido del artículo

---

- Los dispositivos móviles tienen muchos sensores que son atractivos para los atacantes → acceso a través de las aplicaciones.
- Descripción del procedimiento a seguir para realizar un ataque *botnet* sobre distintos dispositivos móviles *Android* al mismo tiempo para capturar información.
- Objetivo: exponer el potencial que tiene este tipo de ataques y el peligro que puede suponer.
- Ejemplo de ataque: mostrar la localización de diferentes dispositivos móviles a través de diferentes fases.

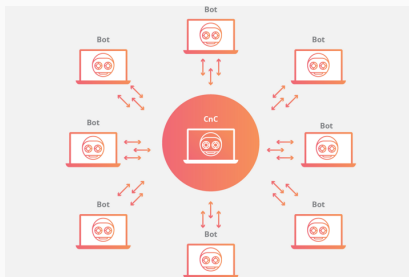
El procedimiento de ataque se realiza en 5 fases:

1. Recoger información.
2. Almacenamiento y gestión de la información.
3. Mostrar la información.
4. Información de verificación.
5. Determinar los puntos de encuentro entre dispositivos infectados



# Fases: Recoger información

- *Botnet*: cualquier grupo de dispositivos infectados y controlados por un atacante de forma remota (*botmaster*).
- Usos: ataques de denegación de servicio distribuidos (*DDoS*), envío de *spam*, minería y robo de criptomonedas.
- Ejemplos: Conficker, Zeus, Waledac, Mariposa y Kelihos.



## Fases: Recoger información

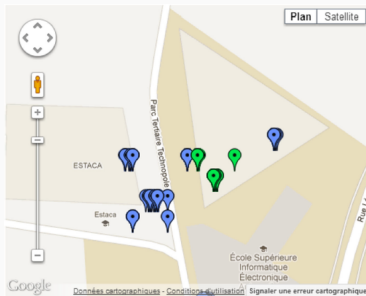
- Se necesita que el dispositivo móvil esté encendido el mayor tiempo posible.
- Se necesita una *botnet* para enviar datos de geolocalización al servidor → se crea una instancia de la clase *LocationListener* al comenzar la actividad principal.
- Se genera un protocolo para enviar los datos recogidos en una sola cadena.
- Es posible transferir los datos recogidos de los *smartphones* a un servidor de dos maneras diferentes → HTTP o SMS.

## Fases: Almacenamiento y gestión de la información

- PHP o MySQL para gestionar la información → instalados por defectos en la red de servidores mundial.
- Se concatenan todas las variables a almacenar en una única cadena (localización, fecha...).
- Primera idea para almacenar los datos: crear para cada *botnet* una nueva tabla con el IMEI (*International Mobile Equipment Identity*) como nombre → problema cuando intentamos crear una nueva tabla con un número como nombre (no permitido en MySQL).
- Solución: traducimos los números de IMEI en caracteres (0 = a, 1 = b, ...). El IMEI es único →  $n$  tablas con  $n$  nombres diferentes.

## Fases: Mostrar la información

- Incluir el encabezado de una web HTML con algunas líneas para inicializar la API de Google Maps.
- Se inicializa una matriz para tener diferentes colores para nuestros marcadores en el mapa.
- Consulta SELECT a los IMEIs. Usamos *fetch()* de PHP para acceder a cada línea de la tabla y almacenamos los nombres de tablas en un *array* para poder hacer consultas en todas las tablas de *botnets*.

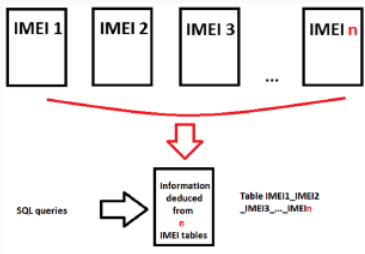


## Fases: Información de verificación

- Determinar el comportamiento de los dispositivos infectados  
→ se debe determinar la velocidad de movimiento de las víctimas.
- Problema: recopilar suficientes datos para determinar el comportamiento de la víctima.
- Antes de analizar cualquier comportamiento se debe comprobar los datos → se crea una nueva tabla con un nombre que es la concatenación de los nombres de las n tablas separados por "\_".
- La fecha será un atributo único en la tabla.

## Fases: Información de verificación

- Se puede separar los datos de todas las *botnets* por cualquier periodo de tiempo.
- Se puede mostrar el mapa datos de las *botnets* respecto a una fecha determinada.
- Se puede seleccionar y mostrar datos de múltiples objetivos con criterios muy específicos (e.g. la fecha).



## Fases: Determinar puntos de encuentro entre dispositivos

Para predecir la posición de los dispositivos infectados, se utiliza el algoritmo *K-means*. Las características de este algoritmo son:

- Partición de un conjunto de  $N$  observaciones en  $K$  grupos.
- La observación pertenece al grupo con la media más próxima.
- Computacionalmente difícil de resolver (*NP-hard*).

## Fases: Determinar puntos de encuentro entre dispositivos

Las fases para determinar la posición son:

1. Inicialización del algoritmo para obtener datos.
2. *Clustering* con la función *K-means* recursivamente.
3. Inicialización del algoritmo *K-means*.
4. Ejecución del algoritmo *K-means*.
5. Determinación de la posición.



# Conclusiones

---

- *Botmaster* controla la *botnet*.
- *Spyware*, *spam*, *DDoS*, robo de información...
- Aplicación legítima → ingeniería inversa → aplicación maliciosa.
- Ejecución en segundo plano (bajo gasto energético y bajo consumo de datos).
- Predicción de posicionamiento.

# Android botnets for multi targeted attacks

---

Jaime García González y Adán Cano Moreno

26 de abril de 2019

Universidad Carlos III de Madrid