#Hacking dispositivos iOS

[iPhone/iPod Touch/ iPad]

Autor: Japson

Índice

DISCLAIMER

Este paper está escrito únicamente con fines educativos y en ningún caso se fomenta el ataque a terceras personas ya que lo único que se pretrende es concienciar de que podemos sufrir un ataque en nuestro dispositivo iOS sin enterarnos si no tomamos las medidas de protección necesarias.

Además, yo, Japson, el autor, no me hago responsable del mal uso que se le puedan dar a los conocimientos adquiridos.

INTRODUCCIÓN

Hoy en día los teléfonos móviles almacenan muchísima información nuestra, cuentas de correo, redes sociales, historial de internet…

Ha habido casos muy sonados de famosos a los que les han robado fotos íntimas de sus smartphones y las han subido a Internet sin que las víctimas se dieran cuenta de nada. Hay más métodos, pero lo que vas a encontrar aquí podría ser perfectamente una técnica válida para llevar a cabo eso.

En este paper simplemente quiero demostrar lo peligroso que es estar conectado a una red Wifi con un dispositivo iOS con OpenSSH activado. Para llevar a cabo este ataque necesitamos dos cosas en el dispositivo víctima:

1- Que tenga jailbreak

2- Que tenga instalado OpenSSH, la mayoría de la gente que ha realizado jailbreak lo tiene instalado por comodidad.

El gran problema: OpenSSH sin configurar

Mucha gente, sobre todo gente no muy familiarizada con el mundo de la seguridad informática, se olvida de algo muy, muy importante que es **cambiar la contraseña por defecto** (es lo más básico, aunque hacer sólo eso no te garantiza ser inmune a un ataque, al final dejo un link interesante sobre como fortificar OpenSSH)**,** con lo cual deja su dispositivo abierto a cualquier posible atacante.

Datos por defectos de OpenSSH:

Usuario: **root**

Password: **alpine**

A esto hay que sumarle que por defecto OpenSSH se queda activado cuando se activa el Wifi. Mucha gente lo instala y se le olvida que lo tiene ahí. Eso es muy peligroso y va a ser la base de nuestro ataque.

Ya os podeis imaginar todo lo que podemos hacer entrando a un dispositivo iOS con privilegios de super-usuario jeje

CAMBIANDO EL PASSWORD POR DEFECTO

Antes de nada, por si algún lector cumple las características descritas antes y no ha cambiado la contraseña por defecto aquí voy a contar como se hace, es muy simple, se hace igual que en cualquier sistema UNIX. Podemos hacerlo desde nuestro ordenador conectándonos con **PuTTy** por ejemplo o desde el mismo dispositivo con la aplicación **MobileTerminal:**

su –

Password: alpine

passwd root

Ahora te pide la nueva contraseña, escribe la que quieras.

Por último desconéctate con logout

Creo que no hace falta que diga que la nueva contraseña que escribas sea segura, combinando números y caracteres.

CRACKEANDO PASSWORD OPENSSH

Seguro que muchos os preguntareis, ¿y si ha cambiado la contraseña por defecto?

Si la contraseña ha sido cambiada aún no está todo perdido, podemos intentar sacarla mediante un ataque por fuerza bruta o por diccionario.

Si no se ha configurado correctamente OpenSSH (cosa que sólo hacen una minoría muy reducida de usuarios, seguramente por desconocimiento o simplemente por no complicarse la vida), es posible sacar las contraseñas por fuerza bruta ya que por defecto OpenSSH tiene comentada la directiva "#MaxAuthTries 6" (/etc/ssh/sshd_config) lo que nos viene a decir que un ataque por fuerza bruta es viable porque el número de intentos de login no está limitado.

Para ello podemos usar aplicaciones como Hydra aunque si la contraseña es robusta será muy difícil de sacar.

ESCENARIO

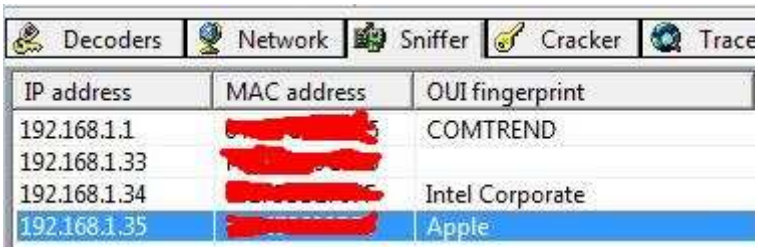
Esta demostración de ataque está realizada con un iPod Touch 4g versión 4.2.1 con jailbreak y OpenSSH conectado a la misma red Wifi en la que estoy yo.

Aunque esto haya sido hecho sobre un iPod, también funcionaría igual si fuese un iPhone o iPad ya que funcionan también con iOS.

Vamos a ello ;)

DETECTANDO DISPOSITIVOS iOS EN LA RED

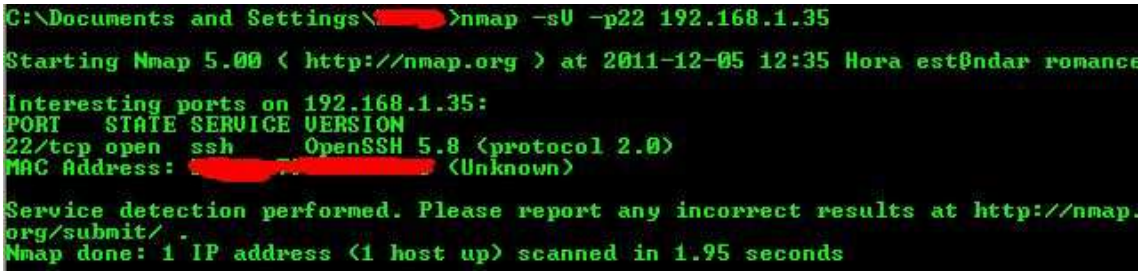
Para ello no hay más que realizar un barrido a la red con vuestro sniffer favorito, yo en este caso voy a utilizar Cain:



Como se ve en la imagen sabemos que 192.168.1.35 es de Apple y podría ser un posible objetivo (podría ser un MAC también, de echo nmap lo muestra como un Mac OS X 10.5, pero no se me ha ocurrido ninguna otra forma clara de verificar que sea iOS).

También podríamos realizar un ataque man in the middle como si de cualquier otro equipo se tratase.

Ahora compruebo rápidamente si tiene el puerto 22 abierto:



Ahí lo tenemos, sobre el puerto 22 está corriendo OpenSSH. Si no nos detectase nada en este puerto podríamos realizar un barrido más profundo de los puertos con nmap porque el puerto por defecto podría haber sido cambiado como medida de seguridad, precisamente para intentar evitar a gente como nosotros ;) o directamente puede que no tenga OpenSSH.

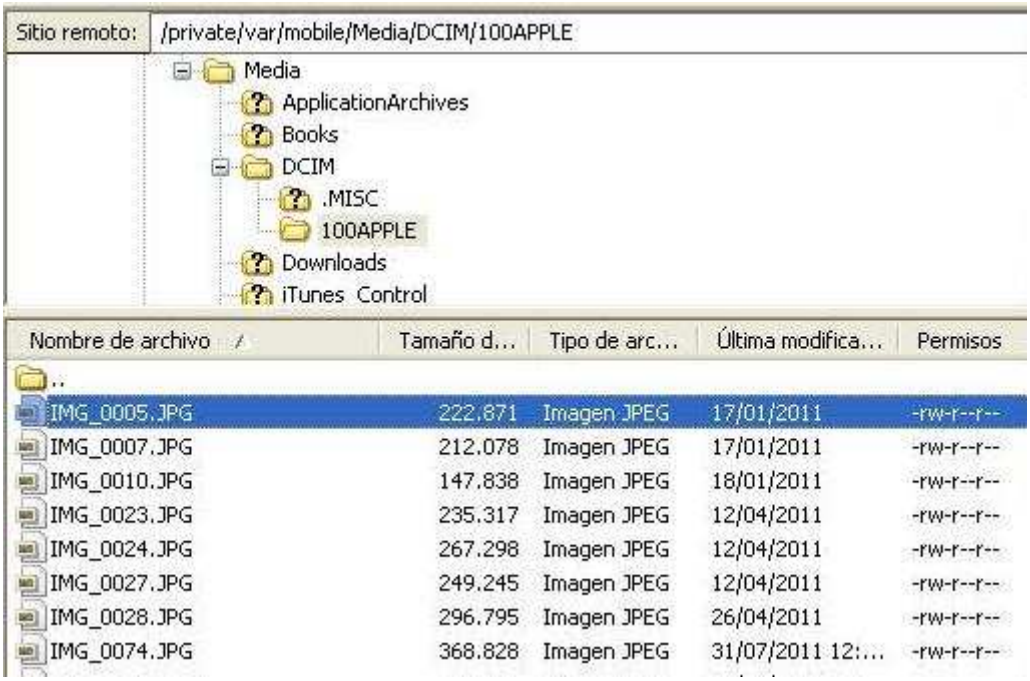
RUTAS INTERESANTES

Ahora comienza la parte divertida, vamos a comenzar a sacar datos y archivos interesantes del terminal. Para ello no tenemos más que conectarnos al puerto 22 (por defecto) con el usuario "root" y la contraseña "alpine" (por defecto) o con la contraseña que hayáis sacado si es que habéis tenido que realizar un ataque por fuerza bruta, con el cliente que os guste más. En mi caso he utilizado Filezilla.

Fotos y videos

Desde la siguiente ruta podemos ver todas las fotos y vídeos almacenados en el dispositivo.

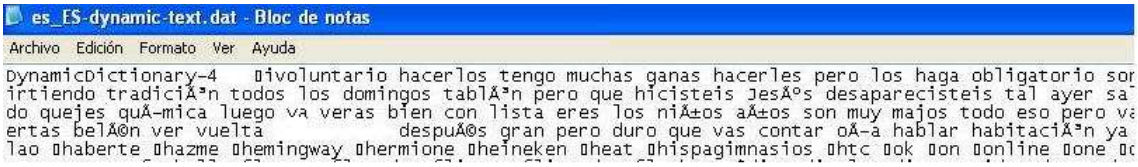
/private/var/mobile/Media/DCIM/100APPLE



Keylogger

Algo muy interesante, iOS almacena en texto plano una especie de registro con las palabras usadas:

/private/var/mobile/Library/Keyboard/es_ES-dynamic-text.dat



Navegador Web (Safari)

/private/var/mobile/Library/Safari:

En el archive *History.plist* podemos ver todo el historial del navegador.

```
    <key></key>
    <string>https://mail.google.com/mail/mu/?login=1</string>
    <key>D</key>
    <array>
      <integer>1</integer>
    </array>
    <key>lastVisitedDate</key>
    <string>3[redacted]</string>
    <key>redirectURLs</key>
    <array>
      <string>https://accounts.google.com/ServiceLogin?service=mail&amp;passive=120
    </array>
    <key>title</key>
    <string>Gmail</string>
    <key>visitCount</key>
    <integer>1</integer>
</dict>
<dict>
    <key></key>
    <string>[redacted]logout</string>
    <key>D</key>
```

APPs:

El famoso WhatsApp

Mucho ha dado que hablar últimamente WhatsApp sobre el tema de seguridad y es que WhatsApp almacena la información en iOS en una base de datos SQLite que se encuentra en la siguiente ruta con lo que cualquiera podría ver con quién hemos chateado, lo que hemos escrito etc. Para abrir la base de datos he utilizado SQLite Manager

/private/var/mobile/Applications/(cadena que varía)/Documents/ChatStorage.sqlite

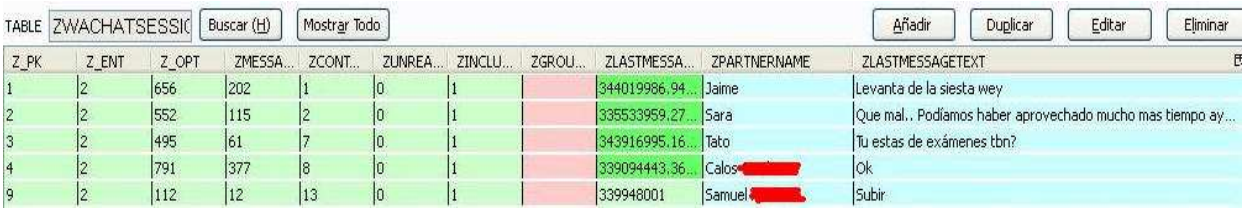


TABLE ZWACHATSESSIC | Buscar (H) | Mostrar Todo | Añadir | Duplicar | Editar | Eliminar

Z_PK	Z_ENT	Z_OPT	ZMESSA...	ZCONT...	ZUNREA...	ZINCLU...	ZGROU...	ZLASTMESSA...	ZPARTNERNAME	ZLASTMESSAGETEXT
1	2	656	202	1	0	1		344019986.94...	Jaime	Levanta de la siesta wey
2	2	552	115	2	0	1		335533959.27...	Sara	Que mal.. Podíamos haber aprovechado mucho mas tiempo ay...
3	2	495	61	7	0	1		343916995.16...	Tato	Tu estas de exámenes tbn?
4	2	791	377	8	0	1		339094443.36...	Calos[redacted]	Ok
9	2	112	12	13	0	1		339948001	Samuel[redacted]	Subir

Si seguimos mirando más tablas podemos ver todo: contactos, números de teléfono, mensajes enviados y recibidos, las fechas….

Twitter

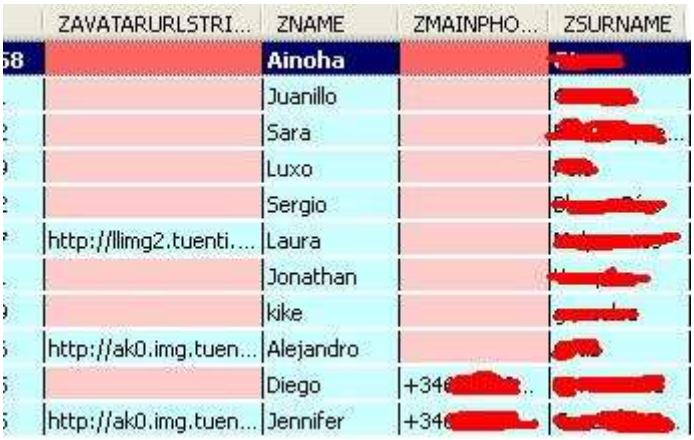
En el siguiente archivo podemos obtener datos datos como nombre de usuario en twitter, últimas búsquedas y tweets…

/private/var/mobile/Applications/(ruta que varía)/Library/Preferences\com.atebits.Tweetie2.plist

Tuenti

Sin duda me ha sorprendido muchísimo encontrar esta base de datos con información de los amigos de la víctima en Tuenti como nombre, apellidos, foto de perfil y en algunos casos hasta el número de su móvil.

/private/var/mobile/Applications/(ruta que varía)/Documents/SessionStorage.sqlite



	ZAVATARURLSTRI...	ZNAME	ZMAINPHO...	ZSURNAME
58		Ainoha		[redacted]
		Juanillo		[redacted]
		Sara		[redacted]
		Luxo		[redacted]
		Sergio		[redacted]
	http://llimg2.tuenti....	Laura		[redacted]
		Jonathan		[redacted]
		kike		[redacted]
	http://ak0.img.tuen...	Alejandro		[redacted]
		Diego	+34[redacted]	[redacted]
	http://ak0.img.tuen...	Jennifer	+34[redacted]	[redacted]

Fijaos en la ruta de los avatares o fotos de perfil, son del estilo a esto:

http://llimg2.tuenti.net/cadena

Ni siquiera hace falta estar logueado en Tuenti para poder ver las fotos! ;)

Y muchas más rutas

Esto es solo un pequeño ejemplo de algunas cosas que podemos ver. Por supuesto hay muchas más rutas y si las mirásemos todas estaríamos aquí un buen rato así que aquí os dejo algunas más para que las investiguéis vosotos mismos:

SMS (iPhone) -> */private/var/mobile/Library/SMS sms.db*

Localizaciones-> */private/var/root/Library/Caches/locationd/consolidate.db*

Mails -> */private/var/mobile/Library/Mail*

Preferencias de aplicaciones-> */private/var/mobile/Library/Preferences*

Información de los contactos-> */private/var/mobile/Library/AddressBook*

DESPEDIDA

Espero que os haya gustado el paper y sobre todo que os haya sido útil para conocer lo vulnerables que somos a un posible ataque si no tomamos las medidas de protección adecuadas. Aquí os dejo un enlace de la gente de *seguridad apple* de cómo configurar correctamente OpenSSH para evitar lo visto anteriormente:

http://www.seguridadapple.com/2010/08/configuracion-openssh-en-dispositivos.html

Un saludo,

JAPS0N