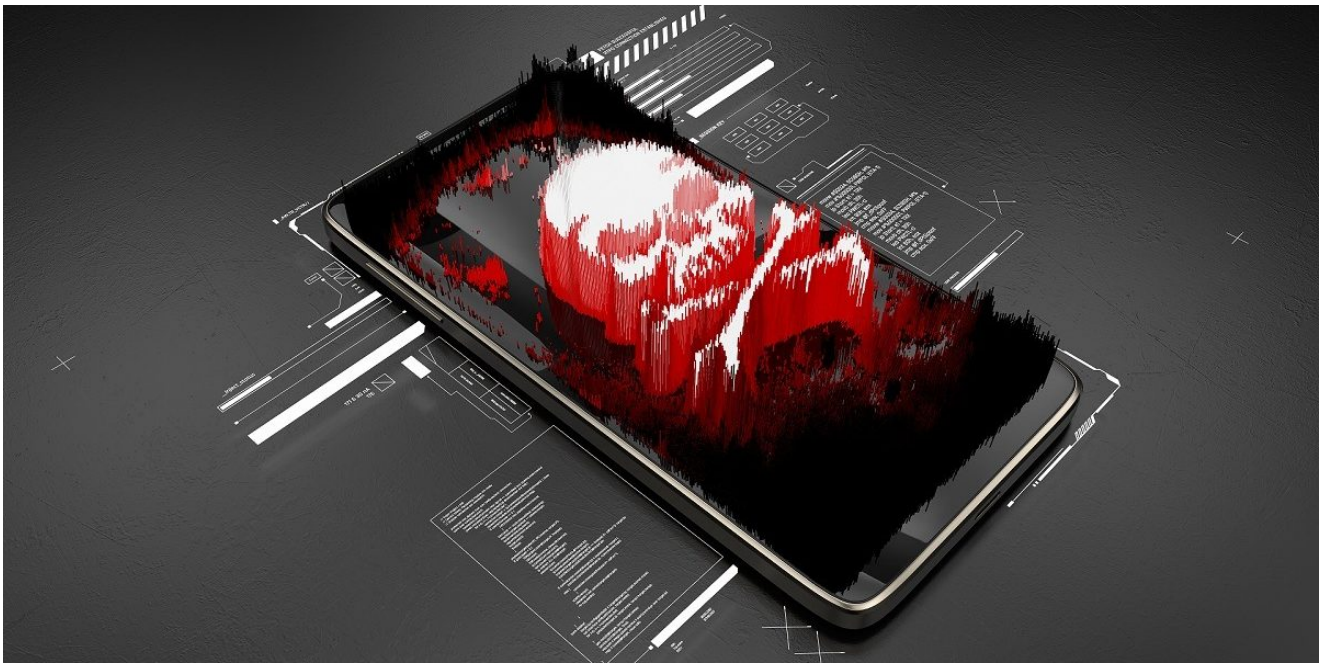
New FinSpy iOS and Android implants revealed ITW

securelist.com/new-finspy-ios-and-android-implants-revealed-itw/91685/



Authors

- Expert GReAT
- Expert AMR

Updated: 23.07.2019
After publication of this article, we received a letter from a representative of Gamma Group International Ltd. stating that they disposed of all interests in FinFisher (FinSpy) in 2013. This article has been corrected in accordance with this new information.

According to information on its official website, FinFisher, among other tools and services, provides a “strategic wide-scale interception and monitoring solution”. This software (also known as FinSpy) is used to collect a variety of private user information on various platforms. Its implants for desktop devices were first described in 2011 by Wikileaks and mobile implants were discovered in 2012. Since then Kaspersky has continuously monitored the development of this malware and the emergence of new versions in the wild. According to

our telemetry, several dozen unique mobile devices have been infected over the past year, with recent activity recorded in Myanmar in June 2019. Late in 2018, experts at Kaspersky looked at the functionally latest versions of FinSpy implants for iOS and Android, built in mid-2018. Mobile implants for iOS and Android have almost the same functionality. They are capable of collecting personal information such as contacts, SMS/MMS messages, emails, calendars, GPS location, photos, files in memory, phone call recordings and data from the most popular messengers.

Malware features

iOS

FinSpy for iOS is able to monitor almost all device activities, including record VoIP calls via external apps such as Skype or WhatsApp. The targeted applications include secure messengers such as Threema, Signal and Telegram. However, functionality is achieved by leveraging Cydia Substrate's hooking functionality, so this implant can only be installed on jailbroken devices (iPhone or iPad; iPod has not been confirmed) compatible with iOS 11 and below (newer versions are not confirmed as at the time of the research and implants for iOS 12 has not been observed yet). After the deployment process, the implant provides the attacker with almost unlimited monitoring of the device's activities.

The analyzed implant contained binary files for two different CPU architectures: ARMv7 and ARM64. Taking into account that iOS 11 is the first iOS version that does not support ARMv7 any more, we presumed that the 64-bit version was made to support iOS 11+ targets.

It looks like FinSpy for iOS does not provide infection exploits for its customers, because it seems to be fine-tuned to clean traces of publicly available jailbreaking tools. Therefore, an attacker using the main infection vector will need physical access in order to jailbreak it. For jailbroken devices, there are at least three possible infection vectors:

- SMS message
- Email
- WAP Push

Any of those can be sent from the FinSpy Agent operator's terminal.

The installation process involves several steps. First, a shell script checks the OS version and executes the corresponding Mach-O binary: "install64" (64-bit version) is used for iOS 11+, otherwise "install7" (32-bit version) is used. When started, the installer binary performs environmental checks, including a Cydia Subtrate availability check; and if it isn't available, the installer downloads the required packages from the Cydia repository and installs them using the "dpkg" tool. After that the installer does some path preparations and package unpacking, randomly selects names for the framework and the app from a hardcoded list,

deploys components on the target system and sets the necessary permissions. After the deployment process is done, the daemon is started and all temporary installation files are deleted.

The persistence of the implant is achieved by adding “plist” with starting instructions to the /Library/LaunchDaemons path.

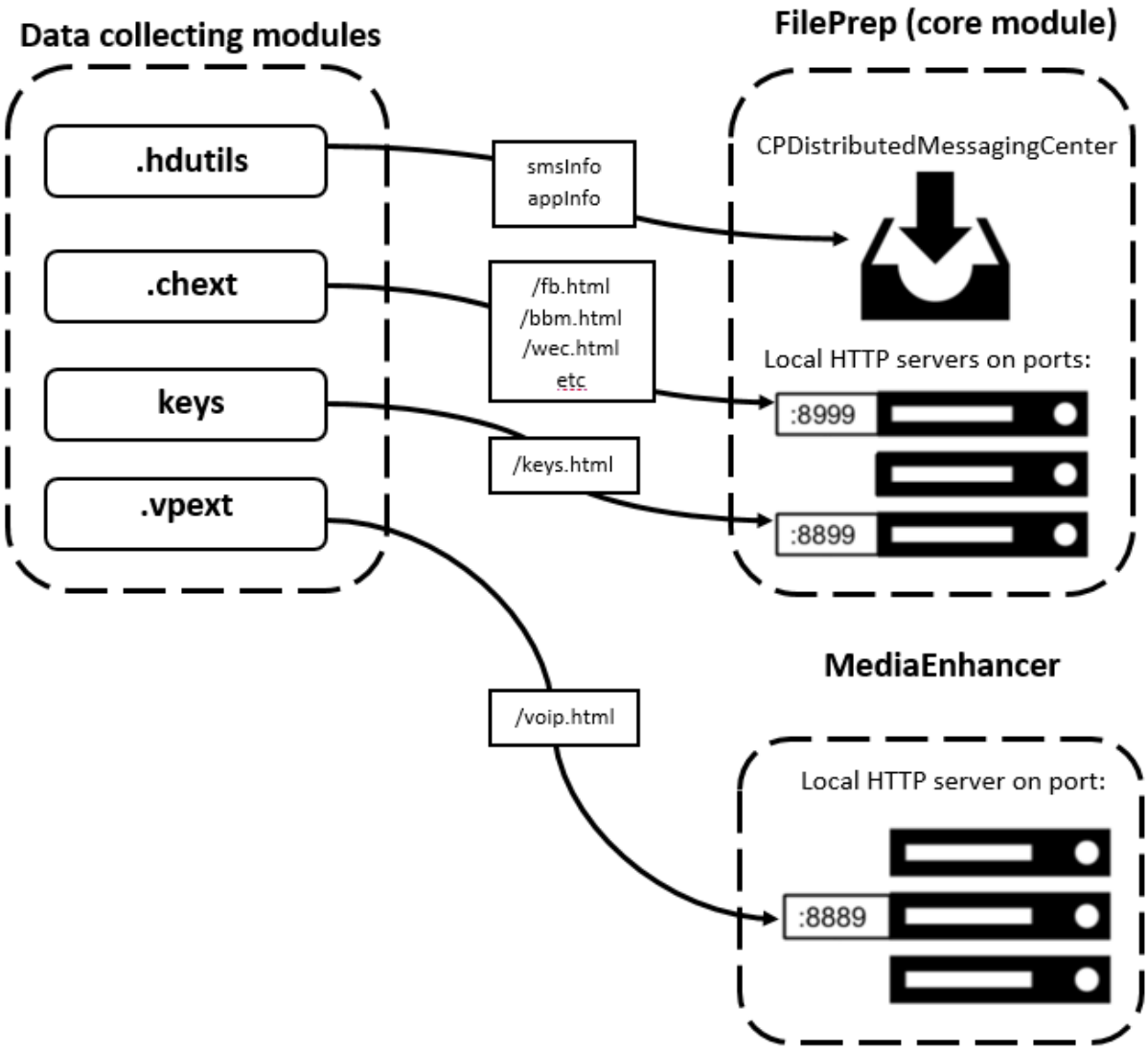
All sensitive parameters of the configuration (such as C2 server address, C2 telephone numbers and so on) are stored in the file “84C.dat” or in “PkgConf”, located in a bundle path of the main module. They can be rewritten using operator commands. This filename was used in previous FinSpy versions for different platforms, including Android.

The following list describes all the modules of the analyzed FinSpy version:

Name	Format	Functionality
netwd	app	Framework, launcher of the core module – FilePrep
FilePrep	app	Core module
MediaEnhancer	dylib	Audio recordings
.vpext	dylib	VoIP calls hooking
.hdutils	dylib	Hiding utilities
keys	dylib	Keylogger
SBUtils	dylib	SpringBoardHooker utilities
.chext	dylib	Messenger tracking
hdjm	unknown	Not observed in detected versions, possibly some type of module for hiding traces of a jailbreak

All the internal strings in the modules, including the installer, are encrypted with a simple xor-based algorithm using the following strings as keys: “NSString”, “NSArray”, “NSDictionary”, “ExtAudioFileRef”.

The core implant module (“FilePrep”) contains 7,828 functions. It controls all the others modules, takes care of HTTP and SMS heartbeats and other service functions. Communication between components is implemented in two ways. The first uses the system’s CPDistributedMessagingCenter, the second is a local HTTP server that receives data requests.



FinSpy Mobile iOS implant components communication

The module ".hdutils" is designed to cover up the tracks of the implant activities on the device. First of all, it configures the processing of all incoming SMS messages. It parses the text looking for specific content and will hide notifications for such messages. Then it sends them to the core module via CPDistributedMessagingCenter (a wrapper over the existing messaging facilities in the operating system, which provides server-client communication between different processes using simple messages and dictionaries). Another hiding feature is to hook the "CLCopyAppsUsingLocation" function in order to remove the core implant module from the displayed list of applications used in Settings geolocation services.

The module ".chext" targets messenger applications and hooks their functions to exfiltrate almost all accessible data: message content, photos, geolocation, contacts, group names and so on. The following messenger applications are targeted:

- Facebook Messenger (com.facebook.Messenger);

- Wechat (com.tencent.xin);
- Skype (com.skype.skype/com.skype.SkypeForiPad);
- Threema (ch.threema.iapp / ch.threema.iapp.ThreemaShareExtension);
- InMessage (com.futurebits.instamessage.free);
- BlackBerry Messenger (com.blackberry.bbm1);
- Signal (org.whispersystems.signal).

The collected data is submitted to the local server deployed by the main module.

The “keys” module focuses on a different kind of keylogging activity, with multiple hooks that intercept every typed symbol. There are several hooks to intercept the typed unlock password as well as during the change password process. The intercepted password is submitted to the “keys.html” page on the local server, similar to the “.chext” module.

The module “MediaEnhancer” is designed to hook system functions in the “mediaserverd” daemon related to call processing, in order to record calls. The module starts a local HTTP server instance on port 8889 upon initialization, implementing VoIPHTTPConnection as a custom connection class. This class contains a handler for requests to localhost/voip.html that could be made by other components.

The module “.vpext” implements more than 50 hooks used for VoIP calls processed by external messaging apps including:

- WhatsApp;
- LINE;
- Skype (that includes independent Skype for iPad version);
- Viber;
- WeChat;
- KakaoTalk;
- BlackBerry Messenger;
- Signal.

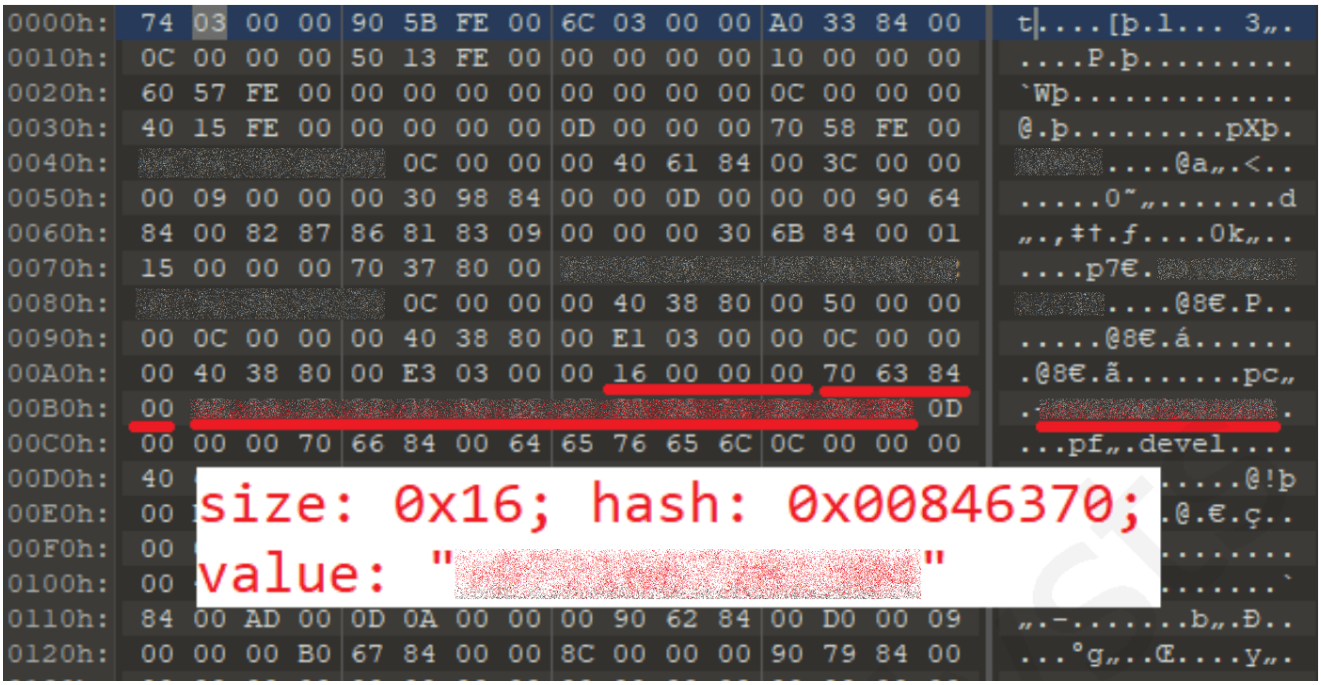
These hooks modify functions that process VoIP calls in order to record them. To achieve this, they send a post request with the call’s meta information to the HTTP server previously deployed by the MediaEnhancer component that starts recording.

Android

The Android implant has similar functionality to the iOS version, but it is also capable of gaining root privileges on an unrooted device by abusing the DirtyCow exploit, which is contained in the malware. FinSpy Android samples have been known for a few years now. Based on the certificate data of the last version found, the sample was deployed in June 2018.

```
Validity: [From: Wed Jun 20 11:19:44 MSK 2018,
           To: Sun Jun 19 11:19:44 MSK 2022]
Issuer: CN='CYGES LLC'
SerialNumber: [    28057230]
```

The Android implant's functionality is unlikely to change much, based on the fact that most of the configuration parameters are the same in the old and new versions. The variety of available settings makes it possible to tailor the behavior of the implant for every victim. For example, operators can choose the preferred communication channels or automatically disable data transfers while the victim is in roaming mode. All the configuration data for an infected Android device (including the location of the control server) is embedded in the implant and used afterwards, but some of the parameters can be changed remotely by the operator. The configuration data is stored in compressed format, split into a set of files in the assets directory of the implant apk. After extracting all pieces of data and building the configuration file, it's possible to get all the configuration values. Each value in the configuration file is stored after the little-endian value of its size, and the setting type is stored as a hash.



For example, the following interesting settings found in the configuration file of the developer build of the implant can be marked: mobile target ID, proxy ip-address, proxy port, phone number for remote SMS control, unique identifier of the installed implant.

As in the case of the iOS implant, the Android version can be installed manually if the attacker has physical access to the device, and by remote infection vectors: SMS messages, emails and WAP Push. After successful installation, the implant tries to gain root privileges by checking for the presence of known rooting modules SuperSU and Magisk and running them. If no utilities are present, the implant decrypts and executes the DirtyCow exploit, which is located inside the malware; and if it successfully manages to get root access, the implant registers a custom SELinux policy to get full access to the device and maintain root access. If it used SuperSU, the implant modifies SuperSU preferences in order to silence it, disables its expiry and configures it to autorun during boot. It also deletes all possible logs including SuperSU logs.

The implant provides access to information such as contacts, SMS/MMS messages, calendars, GPS location, pictures, files in memory and phone call recordings. All the exfiltrated data is transferred to the attacker via SMS messages or via the internet (the C2 server location is stored in the configuration file). Personal data, including contacts, messages, audios and videos, can be exfiltrated from most popular messengers. Each of the targeted messengers has its own unified handling module, which makes it easy to add new handlers if needed.

The full hardcoded list of supported messengers is shown below:

Package name	Application name
com.bbm	BBM (BlackBerry Messenger)
com.facebook.orca	Facebook Messenger
com.futurebits.instamesssage.free	InstaMessage
jp.naver.line.android	Line Messenger
org.thoughtcrime.securesms	Signal
com.skype.raider	Skype
org.telegram.messenger	Telegram
ch.threema.app	Threema
com.viber.voip	Viber
com.whatsapp	WhatsApp

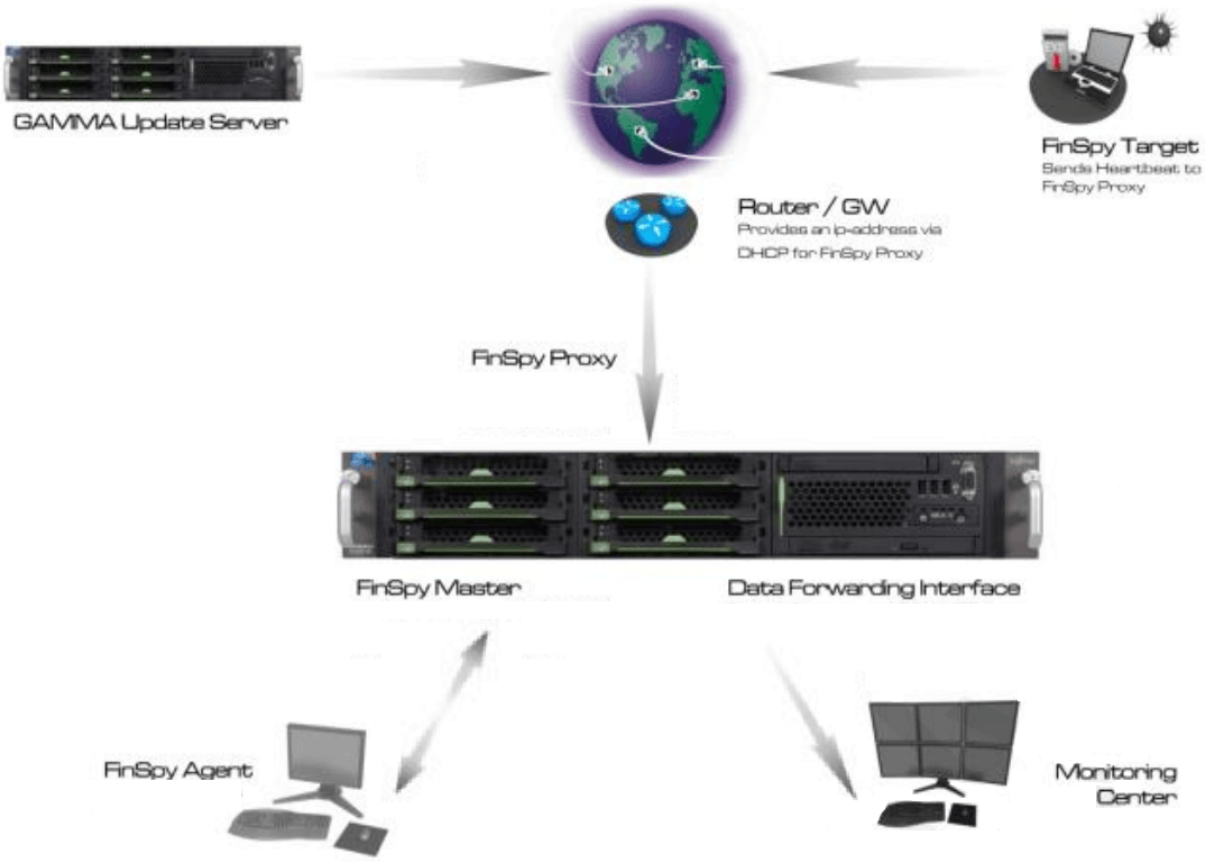
At first, the implant checks that the targeted messenger is installed on the device (using a hardcoded package name) and that root access is granted. After that, the messenger database is prepared for data exfiltration. If necessary, it can be decrypted with the private key stored in its private directory, and any required information can be simply queried:

```
select
    (case message.outbox when 0 then 1 else 0 end) as Incoming,
    message.createdAtUtc as StartTime,
    contacts.publicNickName as OtherParty,
    \'User\' as ThisParty
from message
join contacts on message.identity == contacts.identity
where message.type == 9 and message.id == %d
order by message.createdAtUtc desc
limit 1
```

All media files and information about the user are exfiltrated as well.

```
File v2 = new File(v1 + File.separator + "shared_prefs/ch.threema.app_preferences.xml");
SharedPreferences v1_1 = shared_prefs_utils.a(v0, v2);
this.o = v1_1.getString("identity", "");
this.p = v1_1.getString("nickname", "");
this.getClass().getSimpleName();
logging.log_id_string_2("thisPartyIdentity: " + this.o);
this.getClass().getSimpleName();
logging.log_id_string_2("thisParty: " + this.p);
shared_prefs_utils.delete_file_in_shared_prefs_directory(v0, v2);
```

Infrastructure



FinSpy implants are controlled by the FinSpy Agent (operator terminal). By default, all implants are connected to FinSpy anonymizing proxies (also referred to as FinSpy Relays) provided by the spyware vendor. This is done to hide the real location of the FinSpy Master. As soon as the infected target system appears online, it sends a heartbeat to the FinSpy Proxy. The FinSpy Proxy forwards connections between targets and a master server. The FinSpy Master server manages all targets and agents and stores the data. Based on decrypted configuration files, our experts were able to find the different relays used by the victims and their geographical location. Most of the relays we found are concentrated in Europe, with some in South East Asia and the USA.

Conclusion

FinSpy mobile implants are advanced malicious spy tools with diverse functionality. Various configuration capabilities provided by the spyware vendor in their product enable the FinSpy terminal (FinSpy Agent) operators to tailor the behavior of each implant for a particular victim and effectively conduct surveillance, exfiltrating sensitive data such as GPS location, contacts, calls and other data from various instant messengers and the device itself.

The Android implant has functionality to gain root privileges on an unrooted device by abusing known vulnerabilities. As for the iOS version, it seems that this spyware solution doesn't provide infection exploits for its customers, as their product seems to be fine-tuned to

clean traces of publicly available jailbreaking tools. That might imply physical access to the victim in cases where devices are not already jailbroken. At the same time, multiple features that we haven't observed before in malware designed for this platform are implemented.

Since the leak in 2014, the FinSpy developers has recreated significant parts of its implants, extended supported functionality (for example, the list of supported instant messengers has been significantly expanded) and at the same time improved encryption and obfuscation (making it harder to analyze and detect implants), which made it possible to retain its position in the market.

Overall, during the research, up-to-date versions of these implants used in the wild were detected in almost 20 countries, although the total number could be higher.

FinSpy developers are constatly working on the updates for their malware. At the time of publication, Kaspersky researchers have found another version of the threat and are currently investigating this case.

A full set of IOCs, including YARA rules, is available to customers of the Kaspersky Intelligence Reporting service. For more information, contact intelreports@kaspersky.com

- Apple iOS
- Data theft
- Google Android
- Instant Messengers
- Mobile Malware
- Mobile security

Authors

- 

GReAT
-

AMR

New FinSpy iOS and Android implants revealed ITW

Your email address will not be published. Required fields are marked *