How to covertly leak data from iOS?

Luca Caviglione
National Research Council of Italy (CNR),
Wojciech Mazurczyk
Warsaw University of Technology (WUT)

Short Abstract

iOS has proven to resist against information hiding techniques. However, Siri, a native service to control iPhone/iPad via voice commands, could change this trend in the near future.

Introduction

The growing diffusion of malware raises the attention of victims and security experts, while the escalation of infected devices dramatically inflates the volumes of generated data. Thus, developing some form of stealthiness is of prime importance to exfiltrate data or to not reveal attacks. To this aim, the preferred solution is information hiding, which enables to communicate without being noticed by a third-party observer. Originally introduced for written secrets, nowadays it is a core aspect to effectively assess network security. Recent examples of desktop malware successfully employing information hiding are:

- *Duqu/Alureon* using deceptively innocent pictures to transmit stolen data through the Internet to remote servers [1];
- *Trojan.Zbot* downloading a jpeg embedding a list of IP addresses to be inspected [2];
- *Linux.Fokirtor* injecting data within Secure Shell (SSH) traffic to leak information to its Command&Control (C&C) [3].

Since smartphones enclose a wide variety of personal and sensitive data, they are becoming one of the preferred targets for malware stealing confidential information. Despite the rich set of features exploitable for data hiding, as today, only few malicious applications do use such techniques [4]. This is due to the youngness of mobile appliances if compared with other computing devices. Recently, the diffusion and the opensource nature of Android OS made possible the development of a malware called Soundcomber [5], which covertly transmits the keys pressed during a call (e.g., as it happens when an user enters a PIN to access a financial service). Formerly a niche, the increasing popularity of iOS is increasingly attracting malware developers. In fact, in April 2014, jailbroken iPhones/iPads could be infected via a malicious dynamic library called Unflod.dylib [6]. When running, it listens to outgoing Secure Socket Layer (SSL) connections to steal the Apple-ID and the password of the device, which are then leaked in plaintext. But, the absence of stealthiness partially voids the effectiveness of the threat. Therefore, the advent of a new wave of malware using some forms of information hiding for the Apple ecosystem is unavoidable and only a matter of time. To our knowledge, the method we propose, named iStegSiri, is the first known attempt to covertly leak data from an iPhone/iPad without needing additional applications.

Background

As hinted, information hiding enables to cloak the very existence of the communication, thus differing from cryptography, which makes the content of the transmission unreadable, but

overt. Often, the two mechanisms are used jointly, i.e., to assure that a spotted conversation remains unreadable. Data hiding derives from *steganography*, originally using techniques like, invisible ink or tattoos [1]. To exchange secrets, the two involved endpoints must agree on a pre-shared scheme, and embed them within a carrier: the more its popularity is, the better would be the masking capacity. Though, too many alterations would reveal the presence of the embedded information, then limiting the amount of data that can be covertly transmitted. As an example, a method injecting secrets in the Least Significant Bit (LSB) of a known set of pixels of an image used as the carrier can be discovered because of visible artifacts.
To the aim of exchanging secret data, current network datagrams or sophisticated Internet-scale services are surely the ultimate choice [7]. While early techniques only focused on modifying unused fields of TCP/IP headers (e.g., the Type Of Service field of IPv4, which is rarely set by routers), more recent and sophisticated data hiding methods include, but are not limited to, the exploitation of traffic produced by popular services such as Skype or BitTorrent [8]. In this perspective, modern smartphones offer a wide variety of new carriers. For instance, the traffic used for offloading the device via a cloud, for storage services such as Dropbox, or for voice-based services like Siri.

Siri

Originally released as a standalone application in 2010, Siri has been offered as a native service from iOS5 in 2011. In essence, it allows interacting with the iPhone/iPad in two manners. The first requires its activation, then the user can give commands like «add a note» or «do a phone call». The second is a pure input method, where one can switch at any time from keyboard to voice for entering text. To offload the device, the translation of voice inputs to text is performed remotely in a server farm operated by Apple. To this aim, the iPhone/iPad samples the voice, sends it to a remote facility, and waits for a response containing the recognized text, a similarity score and a time stamp. Figure 1 depicts this usage pattern. This architectural blueprint leads to an appreciable exchange of traffic through the Internet between the two involved parties [9].
Owing to the complete control of Apple on the application distribution pipeline, the diffusion of information hiding methods has been efficiently tamed. Then, from the perspective of hiding data, the traffic of Siri can be an effective carrier, since it does not require the knowledge or alteration of the device, as well as additional software components.

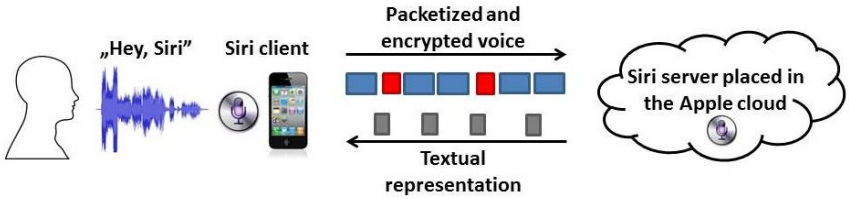

Fig. 1: Architectural blueprint and usage pattern of Siri.

Method & Scenario Overview

Even if provided without sources, internals of Siri are well understood. It processes the voice by using the Speex Codec, and the related data is transmitted to Apple as a sort of one-way VoIP stream encrypted and encapsulated within HTTP. The key idea of iStegSiri is controlling the "shape" of such traffic to embed secrets. To this extent, it solely relies on specific audio patterns captured by Siri via the built-in microphone of the hosting device. Figure 2 depicts the scenario using iSiriSteg to build a covert channel between an infected device and a

botmaster to exfiltrate sensitive information (e.g. a credit card number, or Apple-ID/password [6]).

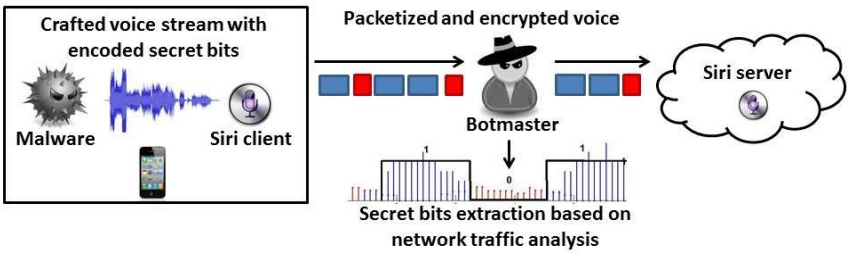

Fig. 2: iStegSiri exploitation for malware purposes.

Specifically, iSiriSteg is based on three different steps: 1) the secret message is converted into an audio sequence based on the proper alternation of voice and silence; 2) by using the internal microphone, the sound pattern is provided to Siri as the input. Consequently, the device will produce traffic towards the remote server to require the audio to text conversion; 3) the recipient of the secret communication passively inspects the conversation and, by observing a specific set of features, it applies a decoding scheme to extract the secret information.

Steps 1) and 2) require a proper matching between the offered audio and the produced throughput. A set of trials and past measurements [9] demonstrated the feasibility of the approach. Unfortunately, algorithms used for synchronization, reduction of latencies and packetization delays, prevent forging the shape of the whole flow, even with a minimal degree of accuracy. To overcome such drawback, we split the overall traffic into different components by using a set of ranges for the Protocol Data Units (PDUs) produced by Siri. Specifically, PDUs in the range of 800-900 bytes were effective in representing talk periods, while PDUs in the range 100-700 bytes inactivity periods. With such a partition, we can arbitrarily encode 1 and 0 within the traffic, i.e., by alternating talk/silence periods as to increase/decrease the number of PDUs belonging to each defined range. Nevertheless, some form of Voice Activity Detection (VAD) impedes high symbol rates, i.e., the speed at which voice and silence alternates. In our trials the shortest working values were 1 s and 2 s, for voice and silence, respectively.

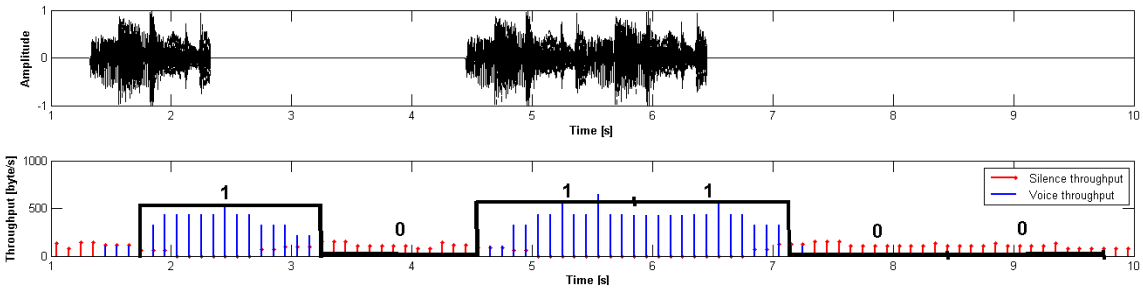

Fig. 3: The crafted voice stream, the corresponding two classes of traffic (blue – voice, silence – red), and the detected secret data bits for iStegSiri.

To complete Step 3), the covert listener must capture the traffic and decode the secret. The former can be achieved in several ways, e.g., via transparent proxies, or by using probes dumping traffic for offline processing. The decoding algorithm implements a voting-like method using two decision windows to select whether a run of throughput values belongs to

voice or silence (i.e., 1 or 0). Figure 3 depicts the outcome of a covert transmission. We experimentally evaluated that the iStegSiri method is able to successfully send secrets at a rate of about 0.5 b/s. For instance, a typical 16-digit credit card number can be transmitted in about 2 minutes.

A malware wanting to exploit iStegSiri can access Siri functionalities in jailbroken devices by means of *libActivator*, or by direct accessing the private APIs provided by Apple in a plain environment. The "audio track" used to encode the secret can be produced by the malware at runtime, i.e., by replicating a single sample via software, thus without requiring to inflate the size of the executable. Nevertheless, even if we used the microphone for the performance evaluation, audio data can be directly routed from the malware to the codec, thus do not requiring a playback audible from the user.

The main limitations of SiriSteg are:

- the requirement of being able to access internals of the service. Currently, for the case of iOS it means that only jailbroken devices can be utilized. Yet, SiriSteg showcases the general principle of using real-time voice traffic to embed data. Therefore, it can be further exploited on existing similar applications (e.g., Google Voice or Shazam) or implemented in future ones also by taking advantage of coding errors;
- the requirement of being able to access the steganographically modified Siri traffic while it travels to server facilities. As mentioned this can be achieved in several ways, e.g., via transparent proxies, or by using probes dumping traffic for offline processing.

Countermeasures

Due to methods using very specific technological traits and their increasing number, there are not off-the-shelf products to effectively detect covert communications. This forces security experts to craft dedicated countermeasures for each method. For the iStegSiri case, the ideal solution acts on the server-side. For instance, Apple should analyze patterns within the recognized text to check if the sequence of words highly deviates from the typical behaviors for the used language. Accordingly, the connection could be dropped to limit the data rate of the covert communication. This approach would not rely on the device, then no additional functionalities or battery consumptions are required.

To conclude, the iStegSiri is the first known information hiding method to implement a covert communication on iOS. Further research will aim at developing an efficient countermeasure to mitigate such threat.

References

[1] E. Zielińska, W. Mazurczyk, and K. Szczypiorski, "Trends in Steganography", Communications of the ACM, vol. 57, no. 2, pp. 86 – 95, March 2014.

[2] J. Gumban, "Sunsets and Cats Can Be Hazardous to Your Online Bank Account", TrendLabs Security Intelligence Blog, URL: http://blog.trendmicro.com/trendlabs-security-intelligence/sunsetsandcats-can-be-hazardous-to-your-online-bank-account/, last accessed July 2014.

[3] B. Prince, “Attackers Hide Communication Within Linux Backdoor”, http://www.securityweek.com/attackers-hide-communication-linux-backdoor, Security Week, last accessed June 2014.

[4] W. Mazurczyk, L. Caviglione, “Steganography in Modern Smartphones and Mitigation Techniques", IEEE Communications Tutorials & Surveys, 2014.

[5] R. Schlegel, K. Zhang, X. Zhou, M. Intwala, A. Kapadia, and X. Wang, “Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones", in Proc. of Network and Distributed System Security Symposium, San Diego, USA, The Internet Society, pp. 17 – 33, Feb. 2011.

[6] B. Lovejoy, “Chinese iOS malware stealing Apple IDs and passwords from jailbroken devices", URL: http://9to5mac.com/2014/04/22/chinese-ios-malware-stealing-appleids-and-passwords-from-jailbroken-devices, last accessed June 2014.

[7] J. Lubacz, W. Mazurczyk, K. Szczypiorski, “Vice over IP”, IEEE Spectrum, Feb. 2010, pp. 40-45.

[8] W. Mazurczyk, K. Szczypiorski, J. Lubacz, “The Spy Who Skyped Me - Four New Ways to Smuggle Messages Across the Internet”, IEEE Spectrum, November 2013, p. 40-43

[9] L. Caviglione, “A First Look at Traffic Patterns of Siri", Transactions on Emerging Telecommunications Technologies, August 2013.