C13 A Forensic Analysis of Joker-Enabled Android™ Malware Apps

Chen Shi, MS, Iowa State University, Ames, IA 50011; Chao-Chun Cheng, Iowa State University, Ames, IA 50011; Yong Guan, PhD*, Iowa State University, Ames, IA 50011*

Learning Overview: This presentation will introduce the basics, challenges, and limitations of the current Android™ malware detection and provide a detailed explanation about the usefulness and availability of information about the list of permissions required and potential private information leakage by malicious apps (e.g. http://wap.thaiza.com/ → browser cookie and timestamp) In addition, the presentation will elaborate our methodology and large-scale experimental evaluation of the approaches used to build android malware analysis databases. Overall, application developers and researchers will learn how to take advantage of the database and search the analysis result for certain android packages to prevent app code being infected by malware.

Impact on the Forensic Science Community: This presentation will impact the forensic science community by providing detailed information about a well-known malware named Joker and its techniques of avoiding detection by a major vetting process.

This project aims at developing a set of automated Android™ malware vetting tools to discover all the malicious behaviors of Android™ malwares in the forms of files in the local storage, SQLite database, or data sent to remote third-party server(s), to establish a dictionary-like Android™ malware database that includes malware themselves (malicious code and variant) with all the detected Internet Provider (IP) addresses, Uniform Resource Locators (URLs) and malicious behaviors as well as other types of evidence data (e.g., the list of permissions required).

In addition, this presentation will demonstrate a well-known malware named Joker (also known as Bread), which has infected over 17,000 Android™ apps since its first release and has evolved into numerous different variants. Both static and dynamic program analysis approaches (Evihunter) were applied on analyzing the malicious code, detecting malicious behaviors, and retrieving evidentiary data like the file path and its corresponding evidence types. Through the analysis of code, it was discovered that Joker not only leverages all kinds of cloaking and obfuscation techniques in an attempt to be undetected, but also uses dynamic package loading to hide its malicious payload. In order to automate the fraud subscribe process, Joker developers utilize injected clicks, custom HTML parsers and Short Message Service (SMS) receivers so that it will not require any interaction from the user. When the infected app gets installed, it carries out either SMS fraud, which sends text messages to premium-rate numbers or Wireless Application Protocol (WAP) billing fraud where a user's mobile account will be paying for the charges of the subscriber's bill. According to Google's® recent report, having three or more active variants of Joker on their official app market at the same time is very common and at peak times of activity, there are up to 23 different versions of Joker family submitting to the Google® Play Store in one day.

As many different variants are active on the air, 12 samples were collected from 46 infected apps that have been removed from Google® Play Store. In some versions of the Joker variants, the final payload is delivered through a direct URL obtained from the listed Command and Control (C&C) server. In these variants, the C&C address has been hidden in the code utilizing the string obfuscation where the string "sticker" was used to break the C&C address and hide it from the simple grep or string search in order to pass the vetting process. In some versions, the infected Google® Play app uses a stager payload to retrieve the final payload where the stager payload URL was encoded in the code and was encrypted using Advanced Encryption Standard (AES). Once an infected app gets started, it downloads the stager payload first, then utilizes the stager payload to execute the malicious final payload. In addition, it was discovered that some variants of Joker even leverage two-stager payload downloads in order to retrieve the final payload. As for these infected apps, they download the stage one payload, which downloads the stage two payload, which finally loads the malicious Joker payload.

Mobile Forensics, Android™ Malware, Digital Evidence

*Presenting Author