C4 Performing Mac® Memory Analysis Using Objective-C and Swift Data Structures

Modhuparna Manna, BTech, Baton Rouge, LA 70820; Andrew Case, MS, Louisiana State University, Baton Rouge, LA ; Golden G. Richard III, PhD, Louisiana State University, Baton Rouge, LA 70808*

Learning Overview: After attending this presentation, attendees will have learned to write plugins for popular memory forensics frameworks after understanding how these plugins actually work. Attendees will become acquainted with Objective-C and Swift runtime data structures.

Impact on the Forensic Science Community: This presentation will impact the forensic science community by providing a method to find information about the source code of any Mac® application written in Objective-C or Swift from a memory snapshot. This method will reveal the classes, methods, and instance variables in the source code of a running application even though the actual source code or executable is not available to disassemble.

Memory analysis, also known as memory forensics, is the process of examination of the physical (volatile) memory of a computer system. Memory forensics has become very popular as it allows the recovery of a wide variety of artifacts that are not written to the hard disk and therefore unavailable when performing disk forensics. However, accurate forensic information can only be found if the forensic investigators have access to proper memory forensics tools. After attending this presentation, attendees will: (1) be acquainted with the state-of-the-art memory forensic techniques provided by popular frameworks such as Volatility; (2) understand how the plugins provided by these frameworks are written; and (3) apply the knowledge to design their own memory forensic tools. Furthermore, this presentation will provide a detailed discussion on the source code of the Swift and Objective-C runtime data structures. The attendees will therefore have a chance to understand the memory layout of the Objective-C runtime and how Mac® applications or malware written in Objective-C/Swift behave "under the hood."

One of the main challenges in computer security is that the source code or the executable of an application or malware may not be available. In such a scenario, where a copy of the executable is not available and therefore disassembling to reverse engineer the source code is not an option, performing memory analysis can help recover information about the classes, methods, instance variables, and values of instance variables used. Memory analysis can be performed on a memory image of the device on which the required application is actively running. This presentation will discuss how the aforementioned technique would work for Mac® applications, how these applications are stored in memory, and how the Objective-C runtime data structure layout can help find information about the application source code.

Today, macOS® is considered one of the most popular operating systems and Objective-C and Swift are the "official" languages of Apple®. Most Mac® applications (and much of the malware) are therefore written in Objective-C and Swift. Apple® has made Objective-C and Swift runtime code open source.[1,2] The Objective-C runtime source code can be used to determine the memory layout of the runtime data structures. This information is then used to write new Volatility plugins. Volatility is one of the most popular memory forensics frameworks, regularly used in forensic investigations.[3] The Volatility plugins are written in Python®.

The results of this research enumerate the names of the classes, methods, and instances used in the source code of a given Mac® application or malware. The objects and meta-classes (containing metadata of classes) are used to find the values of the instance variables.

Reference(s):

1. Opensource-Apple. *Opensource-Apple/objc4*. GitHub. Accessed August 14, 2020. https://github.com/opensource-apple/objc4.
2. Apple. 2020. *Apple/Swift*" GitHub. January 20, 2020. https://github.com/apple/swift.
3. *The Volatility Foundation - Open Source Memory Forensics*. 2014. Volatility foundation. 2014. https://www.volatilityfoundation.org.

Memory Forensics, Mac® Operating System, Objective-C

*Presenting Author