REVERSING LABS

Mario Vuksan & Tomislav Pericin BlackHat USA 2013, Las Vegas

PRESS ROOT TO CONTINUE:

DETECTING OSX AND WINDOWS BOOTKITS WITH RDFU – ROOTKIT DETECTION FRAMEWORK FOR UEFI

Table of Contents

Summary

UEFI/EFI BIOS is the replacement for legacy BIOS images used in all Intel Macs and other PC motherboards. The original specification for EFI was created by Intel in late 90s to address 16-bit and other limitations of traditional BIOS formats. Originally named “Intel Boot Initiative”, it was later renamed EFI and in 2005 open sourced as UEFI or “Unified Extensible Firmware Interface”. Today it is run by a consortium of several technology companies called *Unified EFI Forum*. This forum provides 16 and 32-bit implementations of booting specification for little-endian processors.

While rather exotic, UEFI rootkits are a very bad development as there are no defense mechanisms at the moment. We propose to develop a proper defense mechanism in a form of a special framework that will support multiple UEFI specifications on several platforms. Framework will subsequently be used for a development of a UEFI rootkit detection scanner.

UEFI has recently become a very public target for rootkits and malware. At Black Hat 2012 Snare’s very insightful talk highlighted, once again, a very real and very significant potential for developing powerful UEFI rootkits which then may be very difficult if not impossible to detect and/or eradicate (http://ho.ax/De_Mysteriis_Dom_Jobsivs_Black_Hat_Slides.pdf). UEFI rootkits may make it impossible to disinfect and thus useless. Rootkit Detection Framework for UEFI or RDFU here proposed will bring a unified set of tools to deal with this problem across a wide spectrum of UEFI implementations. Support for a broad spectrum of implementations will alleviate public fears that the only option for dealing with UEFI rootkit infected hardware is its destruction.

How It Is Done Today

As it is to be expected with this type of emerging threats, this attack is a completely new technology and attack vector. Currently there are no commercial or open source defense mechanisms. Best practices tell us to attempt to flash all infected components (if possible), and if not, replace the hardware.

What We Are Trying To Do

What is needed is a rootkit/malware scanner designed to detect UEFI rootkits. Yet scanners are particular to a specific UEFI implementation, which in turn is influenced by the architecture on which it resides. We propose to create the first UEFI rootkit scanner using RDFU rootkit detection framework proposed as proposed in this. Such UEFI scanner would help analysts and IT professionals when scanning suspicious UEFI images and detecting malicious activity.

Underlying framework would in turn help with modifications to specific UEFI rootkit scanners as threats develop and specification changes.

RDFU or Rootkit Detection Framework for UEFI is at the heart of this proposal. It will contain several critical features that are required for a proper implementation of a UEFI rootkit detection scanner. These features are as following:

- Listing all EFI drivers loaded into memory and dumping them to a specified disk.
- Probing entire memory range, scanning for executable files and dumping them when they are found.
- Monitoring all newly loaded drivers until operating system starts and dumping them if they are found.
- Listing and scanning EFI BOOT SERVICES and EFI RUNTIME SERVICES for modified function pointers. Detecting redirections and displaying the module to which they point to.
- Displaying memory map and dumping all suitable regions.
- Continually monitoring EFI BOOT SERVICES and EFI RUNTIME SERVICES while operating system is being loaded. Logging all changes to a log file.
- Listing and monitoring EVENT callbacks that can be used by rootkits/malware. Initial support for EVT_SIGNAL_VIRTUAL_ADDRESS_CHANGE.
- Storing output to a console and a log file, e.g. USB thumb drive or external storage.
- Working in a standalone mode without the EFI shell.

Who It Will Impact If Successful

RDFU project, if successful, will have profound impact on Incident Response professionals, Enterprise security team and concerned individuals. Its goal is to bring about ease of UEFI inspection and ruling out concerns brought about by potentially infected hardware. Reduction in fear, panic and hardware destruction especially as it relates to potential corporate espionage or otherwise APT cases will bring about more strategic and less impulsive responses to security. We also hope that security community will accept this framework and add the scanner tool to its portfolio of tools. We also hope that this work will influence the development of end point security solutions that are crossing the chasm of software validation from hardware to software. In this way, our approach could be a significant factor in continual validation of the entire software stack.

Technical Description

Technical Objectives

This project will initially focus on VMWare as it is currently the most popular virtual machine for forensics and malware analysis. As VMware supports multiple operating systems such as Windows, Linux and Mac OSX, and is used by many security researchers, this makes it as an ideal and very relevant candidate. At that point, a scanner tool will be built on top of EFI BIOS Rootkit/Malware detection framework and tested with the following VMWare EFI implementation: "*EFI Specification Revision: 2.30, EFI Vendor: VMware, Inc . EFI Revision 1.0"*. It will also work with standard UEFI motherboards as long as this firmware version is supported.

Following a successful VMWare implementation RDFU will be extended to support following UEFI revisions and platforms:

- Apple MacOS UEFI Firmware Implementation
- UEFI 2.x Specification for 32-bit and 64-bit Implementations
- UEFI 1.x Specification
- VirtualBox UEFI Implementation

What we do not propose to support is UEFI ARM implementation as the only current implementation is Microsoft Surface Windows RT utilizing Secure Boot and hence significantly reducing the exposure area.

To successfully test RDFU, a bootkit for OSX will be developed and presented. As there is no such rootkit in the wild or as a known PoC, its availability to general public will be discussed with DARPA CFT staff. Developing and presenting such a bootkit is an integral part of the testing harness required for RDFU validation.

Designed specifically for testing purposes, a sample bootkit for Apple Mac OSX 10.7 (kernel 11.0.0 x64) will be provided. It will utilize one of the UEFI "rootkit" techniques in order to infect Mac OSX kernel and will be deployed as an UEFI driver. It is important to note that the entire infection process will be done in memory (by the UEFI driver itself). Therefore the bootkit wil not need to install any additional OSX kernel extension modules. Additionally for further research and testing bootkit, it will present typical bootkit functionality such as:

- Sniffing Filevault password (sniffing keys while booting)
- Hiding pids
- Hiding files and directories with selected pattern
- Privilege escalation (to root)

What is New and Why the Solution Will Succeed

UEFI framework that deals with rootkits is novel and desperately necessary element. As UEFI rootkits are novel, nothing like this has been yet attempted for a level below the OS. RDFU will be from the onset designed so that it can deal with multiple UEFI implementations and anticipate the evolution of UEFI as an international standard. In that way it will have the structure that allows it to easily support modifications and new functionality.

On the other hand, RDFU will succeed as rootkit protection mechanisms have been with us for many years now. Their attack vectors and counter-measures are already known. They will be studied and lessons applied to RDFU.

Technical Challenges and Risk Mitigation

Research

As in all research undertakings, there are implementation risks on the way. For example:

- Brute-Probing memory on Mac EFI firmware causes machine freezing when bottom range is reached.
- Dumping last entry from the Virtual Address Map causes fault in VMware (indicating that selected function is not implemented).

There likely will be other potential and unexpected side effects that will potentially change the scope and implementation approaches.

Risk

While framework flexibility and scanner ease of use will be primary goals, there's significant chance of unforeseen complications due to emergence of new threats or of newly published in-depth analysis of existing threats that yield information on non-trivial infection techniques or on ones that were not contemplated as of the writing of this proposal.

Also, there is a risk associated with typical malware counter measures. For example, rootkit/malware which is already installed on the system may try to fool the scanning framework by patching itself in memory or providing the driver hooked EFI functions. This problem is not unique to the EFI level, but is a general rootkit detection problem for all antivirus solutions at the OS level. Conversely, there always will be counter-measures to the rootkit detection.

The proposed effort will enable experienced professionals to quickly and easily develop of highly complex disinfection modules, thus improving their response times and reducing the need for wholesale system re-imaging, which has become the core task for many security professionals. It will also allow junior reversers to participate and build more sophisticated analysis, decomposition, disinfection and binary repair solutions on their own. Providing advanced tools to a wider audience of security professionals will almost certainly drive the complexity and sophistication of the attacking code as simple legacy approaches will be able to be mitigated even by junior reversers. But as in game theory, known weaknesses of the counter-measures will lead attackers towards expected strategies which can in turn be anticipated and thus addressed.

Writing advanced UEFI rootkits is a complex task with very little open source know how (e.g. by studying existing attack vectors) as UEFI rootkits are not very common at this time. Also, the attacker for the best effect needs to study weaknesses of an inherently specific and a narrow UEFI implementation. Attacker's power is in understanding of weaknesses in a specific UEFI implementation. On the other hand, in this scenario, defenders have the upper hand as their broad-reaching multi-platform implementation provides large reusable surface from where platform specific implementations branch out.

Stealth obtained through rootkits has traditionally been related to kernel level deception whereas defenders would, just like in traditional malware attacks, need to be right every time and the attacker only once. But by developing a framework rather than tool, a dictionary of techniques and tricks is available in every RDFU implementation hence negating the efforts of the Adversary.

RDFU can of course be studied, just like any open source security solution would, for weaknesses in its implementation and areas that it does not address.

Both attackers and defenders have an early mover advantage in this game. Yet both sides are hindered by the attacking/defending complexity which can be detrimental to the stability of the infected host and hence the success of each approach. On the other hand, there will always be a great attraction to the attackers for obtaining stealth at a very low level.

Typically rootkits try to fool defenders by patching themselves in memory and hooking specific functions. Increased sophistication and stealth is brought to bear as attacker is trying to fool all defenders probing and validating techniques.

Key benefactors of this technology are:

- *Enterprise IT and Government IT professionals*. As infected hardware needs to be replaced and destroyed, simple tools than can validate the integrity of said hardware has significant operational and cost saving implications, both in terms of hardware cost, but also in terms of human cost.
- *Anti-Malware Researchers and Incident Response professionals*. People that work for major AV companies, research institution or staff response teams for Governmental organizations and Enterprises. Ruling out UEFI rootkit infection dimension narrows the potential set of suspects and allows for better attribution.
- *General public*. Concerned public investigating unusual firmware related infections.