ANATOMY of a FIRMWARE ATTACK

Attacks against the hardware and firmware of a device stand as some of the highest impact threats facing modern organizations. Firmware retains the highest privileges, allows attackers to bypass traditional controls, and grants a higher level of persistence. The firmware layer has also quickly become one of the most active areas of cybersecurity with attackers increasingly setting their sights on the area of the enterprise where vulnerabilities are plentiful and defenses are often weakest.

Firmware and hardware attacks are also significantly different from traditional malware and network threats. Security teams must be prepared to defend against a new set of attacker strategies, vectors, and understand how the wide variety of firmware components can be used as part of a persistent attack. This provides an introduction to these key topics and follows the progression of a firmware attack including:

- Attacker motivations
- Key firmware components and their role in attacks
- Attack vectors against firmware
- The role firmware plays in persistent attack
- Real-world examples of firmware threats in the wild

With this insight into the anatomy of firmware attacks and how they work, organizations can make informed decisions to better defend their data and assets.

Understanding the Rise of Firmware and Hardware Attacks

Firmware has always been a staple component of computing but has only recently become an active area of attack. This rise of interest in firmware stems from a variety of factors. First, a series of high profile leaks from 2013 to 2016 (Vault7, HackingTeam, Shadow Brokers) revealed a consistent trend. Firmware implants and backdoors were some of the go-to tools for the world's most sophisticated attackers who wanted to evade detection and persist on a host. This is due both to improvements in the security found at the software and OS layers, making the unguarded firmware the path of least resistance.

However, in the same way that nation-state exploits such as EternalBlue were quickly incorporated in widespread malware campaigns (e.g. WannaCry, Trickbot), firmware threats have likewise steadily became more mainstream. For example, the first widespread malware campaign to incorporate a UEFI rootkit arrived in 2018, and was quickly followed by additional large-scale ransomware campaigns and other destructive attacks against firmware.

Likewise, firmware vulnerabilities quickly became one of the most active areas of security research, leading to a flood of newly discovered vulnerabilities in all types of devices ranging from laptops to servers and

networking equipment. Not only had firmware become the path of least resistance, it was omnipresent in every enterprise device.

This combination of active threats and plentiful vulnerabilities made firmware a top priority for cybersecurity leaders. NIST repeatedly called out the importance of securing firmware and has made it a staple component of FISMA compliance. PCI-DSS and other regulations have likewise taken NISTs lead by mapping to the requirements set out by FISMA. Industry analysts including Gartner have similarly elevated the importance of firmware security, both citing it as a top security priority and most recently by predicting that "By 2022, 70% of organizations that do not have a firmware upgrade plan in place will be breached due to a firmware vulnerability".

Attacker Motivations

While actions in the real world make it clear that firmware is a priority, it is important to understand why. By compromising firmware, attackers gain a variety of advantages beyond what can be accomplished via traditional attacks, which we have summarized below.

The Highest Levels of Privilege – Attackers naturally seek out the highest privileges possible for furthering their attacks. At a user level this traditionally meant raising privilege within userspace (Ring 3) such as escalating from user to Administrator privileges, or seeking out kernel privileges (Ring 0) at the system level. And while these are the highest privileges available to the OS, firmware sits beneath the kernel. As a result, malicious code in the firmware has the potential to subvert the kernel and thus conceptually possess even higher privileges than Ring 0. These privileges are sometimes referred to as Rings -1 through -3. Control over these layers provide attackers with the highest level of privilege on the device and the ability to subvert all higher layers.

Bypass of Traditional Security -The ability to subvert higher layers most importantly allows attackers to avoid controls and security measures that run at higher layers or only see down to Ring 0. This includes traditional security running at the operating system and virtual machine layers. For example, compromised firmware can easily allow an attacker to control how a system boots, to patch the operating system itself, to read privileged data off of hardware, or control assets that the operating system can't see into. In the case of servers, compromises to the firmware can also allow attackers to further compromise the hypervisor and virtual machine layer in an organization's cloud assets.

Persistence – The ability to hide from and evade the operating system provides attackers extreme levels of persistence on a compromised device. In addition to evading controls, any malicious code in the firmware is naturally tied to the hardware of the device as opposed to the software. This means the attacker's code would naturally persist even across a full re-imaging of system. Such capability is particularly strategic for an attacker as it is often essential to furthering the broader campaign by maintaining command and control points and facilitating the ongoing attack.

Stealth – Compromised firmware also enables attackers to perform a variety of critical attack functions without being detected. For example, by controlling the firmware of hard disk or SSD, an attacker could hide malware in a section of the disk that is not reported to the OS, allowing it to avoid scanning by antivirus tools. Likewise, attacks against the management components of a device such as Intel's Management Engine have allowed attackers to send command-and-control traffic through independent channels that aren't monitored by host-based firewalls running at the OS level.

Damage – Lastly, access to the firmware layer enables attackers to cause irrevocable damage to a device. By damaging the firmware itself, attackers can "brick" the device permanently. This can potentially cause organizations to rethink their recovery models since it shifts the response from data recovery and reinstallation to complete device replacement. Additionally the act of effectively disabling a device can have enormous impacts to an organization by disrupting business, services, and even critical infrastructure.

The Building Blocks of a Firmware Attack

Firmware attacks involve a variety of components and techniques that are often not seen in more traditional software-based attacks. In this section, we will introduce some of the most important firmware components within a device, how they can be abused as part of an attack, and the strategies and techniques attackers use against them. This section is neither exhaustive or definitive, but rather is to provide an introduction to the key concepts of firmware attacks.

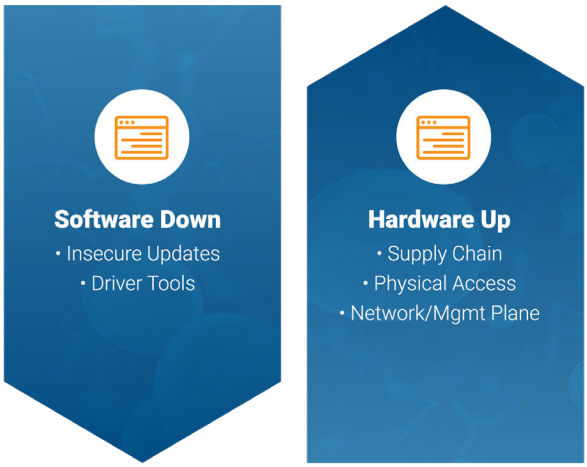
Key Firmware Components

When it comes to firmware, the first thing that comes to mind is often the system firmware such as BIOS or its more modern replacement, UEFI. This system firmware is particularly powerful, but it is only one of dozens of components within modern devices that rely on firmware and can play key roles in an attack. The table below introduces several of these key components and their role in an attack. However, for a more thorough analysis of firmware components and their exposure if compromised, we encourage you to refer to the online Know Your Own Device resource on the Eclypsium® website.

Firmware Component	Role in an Attack
System Firmware and Other Boot Firmware (BIOS, UEFI, EFI, MBR)	This critical firmware is the first code to run on startup and can subvert the operating system by changing bootcode, patching the OS kernel, as well as compromising hypervisors and virtual machines. System firmware can also be used to deliver malicious firmware to other components in the system.
System Management Mode (SMM)	SMM and similar firmware focus on the runtime operation of the device and allow the system to manage low-level behavior on the system completely independently of the OS. This can allow an attacker virtually unfettered access to the system without the knowledge of the OS.
Processors	By exploiting vulnerabilities in the CPU, attackers are able to view information that should remain protected such as keys that typically remain in privileged memory. Processor vulnerabilities also have the potential to be exploited remotely, which further raises their risk. Vulnerabilities in CPUs are particularly concerning because they can affect any system or operating system even if there are no vulnerabilities in the software and all OS-level protections are enabled. Additionally, patching processor vulnerabilities can be challenging, requiring the installation of firmware updates, processor microcode updates, OS updates, VMM updates and software updates.
Intel ME	Intel's Management Engine (ME) is a common component built into personal computers to enable out-of-band management of the device. ME is also known as TXE in Atom platforms and CSME in Skylake and newer platforms. The firmware within ME includes technology known as Active Management Technology (AMT) and its functions and networking are completely independent of the host operating system and can be used by attackers for command-and-control, data exfiltration, and a variety of other functions.
Baseboard Management Controllers (BMC)	BMCs provide out-of-band management for servers and likewise have their own independent networking, firmware, and resources. Attacks against BMCs can allow attackers complete control over the server and all the higher layer data and services it contains, and can even be used to permanently brick the server entirely.
USB Devices	USB devices can contain malware in their firmware, allowing them to spoof other USB devices, exploit OS kernel and drivers and can even deliver malware or malicious firmware to other vulnerable or unprotected components.
Network Cards and PCIe Devices	The PCIe bus provides connection to a wide variety of critical components such as GPUs, network interfaces and much more. The firmware of PCIe cards and PCIe-connected devices can be infected both over software as well as the network. When compromised that can perform devastating DMA attacks which read and write system memory and execute malicious code in the context of the victim operating system.
Dozens of additional components and modules	Virtually every device or component within a system relies on firmware, and their use in an attack will vary based on the function of the component. DRAM, GPUs, network interfaces, and drives are just a few examples. Refer to the Know Your Own Device page to learn more.

Attack Vectors

Now that we have identified some of the critical pieces of the firmware attack surface we can take a look at the methods attackers can use to target the firmware. Attackers typically need to establish a path to the firmware, and this can be done in a few general ways. **Either the attacker can move from the traditional network or software layer down to firmware, or move from the hardware up**.



Software Down

The "software down" approach to firmware often follows a similar path as traditional malware infections. The initial compromise may come via phishing, drive-by-downloads, or social engineering. At this point, the attacker could directly attack vulnerabilities in vulnerable firmware to deliver an implant or could use additional tools such as vulnerable drivers to escalate privileges and control firmware as needed. Both options are very straightforward for attacker. The first is a straightforward exploitation of a vulnerability, and the latter represents the standard trojan/dropper model used by malware for years.

Unfortunately, lack of updates and patching makes firmware vulnerabilities extremely common. In fact, Eclypsium has found that well over 95% of devices contain at least one vulnerability in real-world environments. A vulnerability could be as simple as a component not requiring firmware updates to be signed, thus allowing an attacker to replace the firmware with a malicious image.

However, attackers can also use a variety of tools to move from user space to firmware. Tools such as RWEverything (read-write everything) can allow an attacker to directly write to the firmware on a component. Ironically, our research shows that some of the most powerful tools are often the drivers designed to manage the hardware and components themselves. These drivers can be abused to install the attackers code into UEFI and other components. These techniques were recently used in the Slingshot APT campaign as well as by LoJax malware.

Hardware Up

Attackers can also take a more hardware-focused approach to the firmware, and this strategy can take many forms. Compromising a device in the supply chain provides one of the most direct paths to a device's hardware and its firmware. Malicious or intentionally vulnerable firmware can be introduced into a product if a vendor or one of its suppliers is compromised. This is actually quite a large attack surface given the many components and extensive underlying supply chain that goes into a modern device. Likewise, as shown in the recent ShadowHammer attacks, the supply chain can be attacked even after a device is delivered by compromising the official updates delivered by a vendor. In either case, an attacker is able to compromise a system that an organization typically assumes to be safe.

Next, firmware can be compromised by an attacker with physical access to a system. Malicious firmware within a USB device can be used to compromise the firmware of a victim system within minutes via "evil maid" attacks. Thunderbolt ports are likewise potential methods of entry as evidenced by the sonic screwdriver implant. And in some cases, this style of attack can even be launched remotely when USB ports are made available via virtual media services.

Lastly, system and component firmware can be attacked directly over the network. UEFI, BMCs, and other firmware components often have their own independent networking capabilities designed for providing out-of-band updates. This can potentially expose vulnerable components directly to the Internet. Similarly, weaknesses in the update process can allow an attacker to directly compromise the components firmware remotely.

Malicious Techniques

Once the firmware is compromised, the attacker will naturally want to use the position to continue the attack. This can include establishing persistence, compromising additional firmware components, capturing privileged information, exfiltrating data, impacting performance, and even disabling the device completely.

Altering Boot Process – By compromising the system firmware or firmware within the Trusted Platform Module, attackers can establish persistence by disrupting the secure boot process of a system. This can include directing the system to an attacker-supplied boot image, directly patching the OS kernel, or avoiding firmware integrity checks during startup. Attackers can also use Option ROM attacks to alter other firmware at boot time.. In fact, almost any firmware component can alter the boot process. For example, compromised NIC or BMC can do a DMA attack in the middle of the boot process and compromised SSD can alter the data used during the boot process, bypass boot time full-disk encryption etc.

Option ROM Attacks – Option ROM attacks can be used as part of an initial infection or to spread malicious firmware from one component to another. Option ROM is normally used by components to obtain the appropriate firmware during the boot process. Compromising the Option

ROM firmware to offer an initial method of infection to provide a persistent modification of the boot process without directly modifying the system UEFI firmware.

Direct Memory Access (DMA) – DMA attacks are yet another common technique used in firmware-based attacks. DMA is normally used to give components direct access to system memory without going through the operating system. This approach gives components and peripherals improved performance. However, this access can allow a compromised component to read memory and privileged information, cryptographic keys, and other data in memory. Attackers could likewise use this access to install malicious code on the system. DMA attacks are particularly common PCI-connected devices and the device's DRAM.

Processor Level Exploits – Attacks such as Rowhammer can allow an attacker to flip bits in areas of RAM in order to escalate privileges. Likewise the now infamous Spectre and Meltdown vulnerabilities allow side-channel access to potentially protected information on the processor.

Disabling Devices – Lastly, firmware can be used to temporarily or permanently disable a device. Firmware provides a natural way to disable a device since it is the first code that runs and plays a critical role in the rest of the boot process. These techniques can be used against virtually any device including servers.

Case Studies

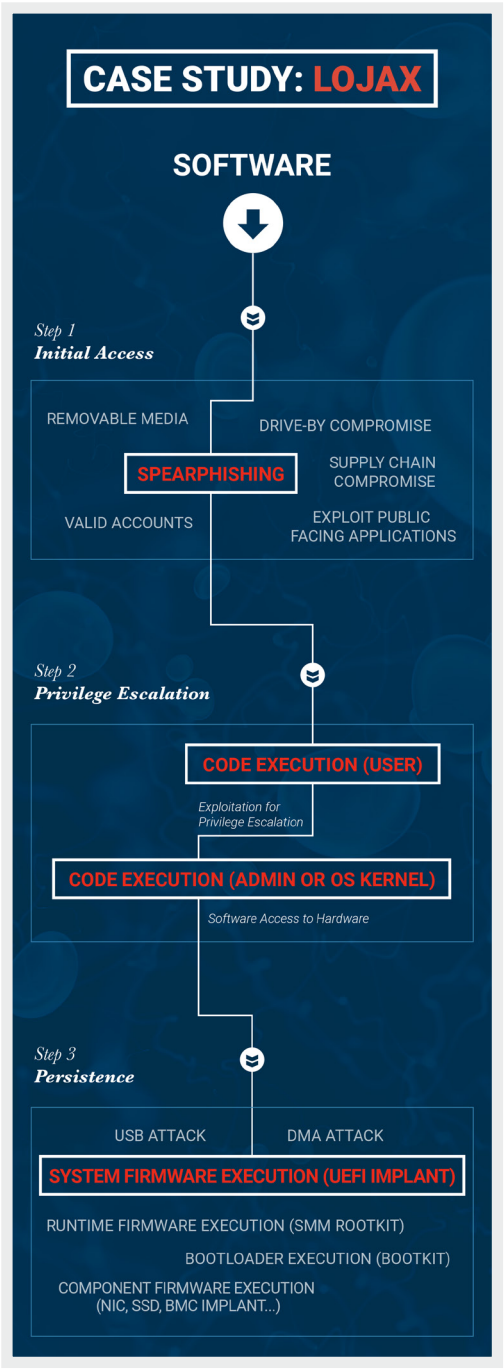
Case Study: LoJax Attack

Next we can take a look at a recent malware campaign to see how firmware is being used in a real-world context. There are multiple potential paths of a LoJax malware attack.

For example, the initial access could stem from any number of traditional malware infection vectors including email attachments, drive-by-downloads, XSS attacks, inserting USB drives or physical access from an attacker, just to name a few. While the LoJax campaign was primarily driven through email phishing, a similar attack could easily begin via a compromised USB or Evil Maid attack.

When a user opened a malicious attachment, the malware used a vulnerable driver like those described in our "Screwed Drivers" research to gain arbitrary access to physical busses within the platform. It then leveraged this direct hardware access to talk to the SPI controller in the chipset which controls access to the SPI chip containing the UEFI system firmware. The next step was to use this access to write a malicious DXE application to the UEFI system firmware which will run at each subsequent boot. When this added DXE application runs during the boot process, it drops applications embedded within itself into the drive containing the operating system and modifies the registry before Windows starts to ensure that its services run on every boot even if the system was wiped and re-imaged.

Next, with control over the operating system, the malware is able to drive the ongoing process of attack. This includes the ability to read files and collect data on the system, launch exploits against other systems and services, and use RPC to create a command-and-control channel.



eclypsium

DEFENDING THE FOUNDATION
OF THE ENTERPRISE

Summary

Firmware attacks constitute some of the most high impact threats facing organizations today. With control over the firmware layer attackers are able to persist on the device, evade traditional security, and ensure access to the highest levels of privilege on the device. These fundamental capabilities can apply to almost any phase of attack from command-and-control to lateral movement to the theft and destruction of data and assets.

Attackers also have many paths they can take to reach the firmware layer. By leveraging weaknesses in firmware or vulnerable drivers, attackers can extend traditional software and malware based attacks to the firmware layer. Additionally, attackers can target the hardware directly via the hardware supply chain, exposed ports, or even over the network via remote media or firmware update processes.

These are just some of the ways that firmware can be attacked today. As part of our ongoing research we will continue to keep this resource updated with the latest techniques, strategies, and example attacks based on changes in the wild. If you have any questions about the information presented here or would like to learn more about the Eclypsium solution, please contact us at info@eclypsium.com.