FIRMWARE AND FRAMEWORKS: MITRE ATT&CK

As cybersecurity exploits and attacks have increased, few tools have proven as essential at stemming the tide and aiding defenders as the **MITRE ATT&CK® framework**. This quirky acronym (almost universally pronounced "miter attack") stands for "Adversarial Tactics, Techniques, and Common Knowledge" and represents a sea change in a near half century of cyber defense strategies. Instead of generalizing against vague "cyber threats," or outlining hazy "vulnerabilities," MITRE ATT&CK is a global, curated knowledge base that models and defines specific cyber adversary behavior "in the wild". The goal is to detail adversarial attack methods, lifecycles and known target platforms, and thereby enable defenders to see not only who and what they're up against, but precisely how to counter specific assaults.

Firmware is critical software that is embedded within every device. This is true whether enterprises deploy devices locally or rent them from cloud platforms. In either case, firmware is the first code to run and some of the most privileged code on any device, the bedrock that everything else relies on. Any compromise of this critical layer can undermine and subvert everything above it including the operating system, applications, and security controls.

It should be no surprise then that firmware also plays a critical role in modern cyberattacks, serving as a highly popular initial attack vector, as well as providing some of the most powerful methods of security evasion, persistence, and command-and-control.

In this paper, we'll analyze how firmware security – meaning boot integrity, component code, hardware configurations – and more applies to the ATT&CK framework. In addition to covering firmware-specific attacker techniques and sub-techniques named in the framework, we'll also analyze the key role that strong firmware security plays in mitigating risk and disrupting threats across various phases of an attack. Specifically, readers will learn:

- Background on the MITRE ATT&CK Framework
- The role firmware plays in MITRE tactics and techniques
- Examples of real-world firmware attacks and risks
- Countermeasures and practices to mitigate firmware risks

MITRE ATT&CK BACKGROUND

The MITRE ATT&CK Framework was first launched in 2015 as a "globally-accessible knowledge base of adversary tactics and techniques based on real-world observations." The terms "tactics" and "techniques" have a specific meaning in the context of ATT&CK. The MITRE ATT&CK Philosophy Paper (**PDF**) defines the terms as follows:

1. **Tactics** - Tactics represent the "why" of an ATT&CK technique or sub-technique. It is the adversary's tactical objective: the reason for performing an action.
2. **Techniques** - Techniques represent "how" an adversary achieves a tactical objective by performing an action.

For example, Persistence is a tactic, and Pre-OS Boot is a technique used to establish persistence. In the most recent iteration of ATT&CK (last modified in November of 2021) there are 14 tactics and 218 underlying techniques. Of these 14 tactics, 2 are designated as PRE or preparatory steps, while the remaining 12 focus on the active phase of the attack.

The 14 tactics include:

- Reconnaissance (PRE)
- Resource Development (PRE)
- Initial Access
- Execution
- Persistence
- Privilege Escalation
- Defense Evasion
- Credential Access
- Discovery
- Lateral Movement
- Collection
- Command and Control
- Exfiltration
- Impact

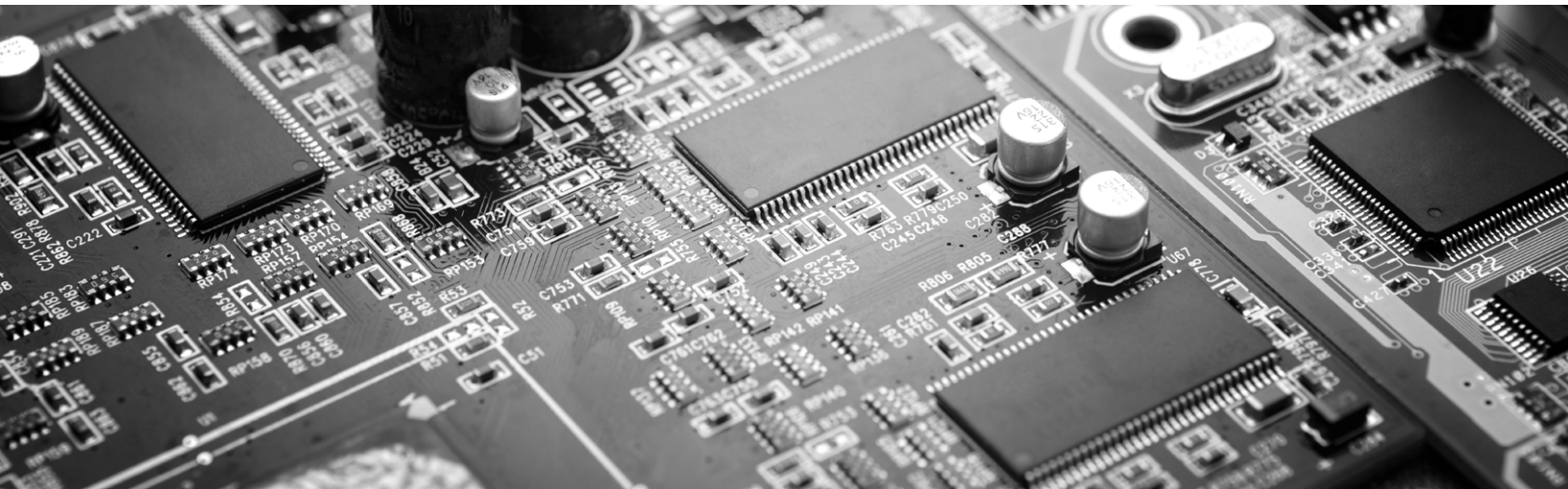
It is worth noting that ATT&CK maintains separate matrices for Enterprise, Mobile, and ICS target types. While the underlying techniques vary between the matrices, the core tactics remain largely the same with the exception of small differences in the ICS matrix. For this document, we will use the Enterprise matrix as a reference, knowing that the core concepts apply to all versions of ATT&CK.

FIRMWARE AND ATT&CK

Firmware is the most fundamental code on any given device. From the first moment that a device starts up to its every runtime operation to EOL, firmware is the gateway from the world of code to the physical processing and sharing of information required in all of computing. It is the bedrock that higher layer abstractions such as operating systems and applications rely on.

It used to be assumed that firmware code was sacrosanct: pristine, untouched, "black box" code. This caused cybersecurity teams to overlook firmware, or to provide a **litany of reasons** why they didn't want to assess, manage and secure it. Now, of course, many attackers know firmware better than defenders do. And compromising this firmware layer gives them the chance to subvert virtually any and all controls running in the "higher layers" of operating systems and applications.

Nation-state threat actors have heavily invested in firmware-level threats for this reason, and those techniques and tools have trickled down to more financially motivated threat actors. However, firmware remains relatively unguarded in many organizations. This combination of motive, means, and opportunity creates considerable risk that we can see in the context of the ATT&CK Framework.



MITRE ATT&CK MATRIX

RECONNAISSANCE	RESOURCE DEVELOPMENT	INITIAL ACCESS
Active Scanning	Acquire Infrastructure	Drive-by Compromise
Gather Victim Host Information	Compromise Accounts	**Exploit Public-Facing Application**
Gather Victim Identity Information	**Compromise Infrastructure**	**External Remote Services**
Gather Victim Network Information	Develop Capabilities	Hardware Additions
Gather Victim Org Information	Establish Accounts	Phishing
Phishing for Information	Obtain Capabilities	**Replication Through Removable Media**
Search Closed Sources	Stage Capabilities	**Supply Chain Compromise**
Search Open Technical Databases		Trusted Relationship
Search Open Websites/Domains		Valid Accounts
Search Victim-Owned Websites		

Compromised or counterfeit firmware is increasingly used to provide detailed information on target systems

Firmware can be used used to take over infrastructure and use it for the attacker's efforts and purposes

Firmware – ever present and poorly understood – is becoming a favored initial vector for attacks of all types

EXECUTION	PERSISTENCE	PRIVILEGE ESCALATION
Command and Scripting Interpreter	Account Manipulation	Abuse Elevation Control Mechanism
Container Administration Command	BITS Jobs	Access Token Manipulation
Deploy Container	Boot or Logon Autostart Execution	Boot or Logon Autostart Execution
Exploitation for Client Execution	**Boot or Logon Initialization Scripts**	**Boot or Logon Initialization Scripts**
Inter-Process Communication	Browser Extensions	Create or Modify System Process
Native API	Compromise Client Software Binary	Domain Policy Modification
Scheduled Task/Job	Create Account	Escape to Host
Shared Modules	Create or Modify System Process	Event Triggered Execution
Software Deployment Tools	Event Triggered Execution	**Exploitation for Privilege Escalation**
System Services	External Remote Services	Hijack Execution Flow
User Execution	Hijack Execution Flow	Process Injection
Windows Management Instrumentation	**Implant Internal Image**	Scheduled Task/Job
	Modify Authentication Process	Valid Accounts
	Office Application Startup	
	Pre-OS Boot	
	Scheduled Task/Job	
	Server Software Component	
	Traffic Signaling	
	Valid Accounts	

While firmware isn't a frequent culprit in execution stages, users can be tricked into "updating" illegitimate firmware with disastrous results

Because firmware is often trusted by default in boot or pre-boot sequences, there are few tactics better suited to enabling long-term persistence than firmware compromise

Due to its critical position in the startup process, firmware is a favorite adversary tool for obtaining escalated privileges

DEFENSE EVASION	CREDENTIAL ACCESS	DISCOVERY
Abuse Elevation Control Mechanism	**Adversary-in-the-Middle**	Account Discovery
Access Token Manipulation	Brute Force	Application Window Discovery
BITS Jobs	**Credentials from Password Stores**	Browser Bookmark Discovery
Build Image on Host	**Exploitation for Credential Access**	Cloud Infrastructure Discovery
Deobfuscate/Decode Files or Information	Forced Authentication	Cloud Service Dashboard
Deploy Container	Forge Web Credentials	Cloud Service Discovery
Direct Volume Access	Input Capture	Cloud Storage Object Discovery
Domain Policy Modification	Modify Authentication Process	Container and Resource Discovery
Execution Guardrails	Network Sniffing	Domain Trust Discovery
Exploitation for Defense Evasion	OS Credential Dumping	File and Directory Discovery
File and Directory Permissions Modification	Steal Application Access Token	Group Policy Discovery
Windows Management Instrumentation	Steal or Forge Kerberos Tickets	Network Service Scanning
Hide Artifacts	Steal Web Session Cookie	Network Share Discovery
Hijack Execution Flow	Two-Factor Authentication Interception	Network Sniffing
Impair Defenses	Unsecured Credentials	Password Policy Discovery
Indicator Removal on Host		Peripheral Device Discovery
Indirect Command Execution		Permission Groups Discovery
Masquerading		Process Discovery
Modify Authentication Process		Query Registry
Modify Cloud Compute Infrastructure		Remote System Discovery
Modify Registry		Software Discovery
Modify System Image		System Information Discovery
Network Boundary Bridging		System Location Discovery
Obfuscated Files or Information		System Network Configuration Discovery
Pre-OS Boot		System Network Connections Discovery
Process Injection		System Owner/User Discovery
Reflective Code Loading		System Service Discovery
Rogue Domain Controller		System Time Discovery
Rootkit		Virtualization/Sandbox Evasion
Signed Binary Proxy Execution		
Signed Script Proxy Execution		
Subvert Trust Controls		
Template Injection		
Traffic Signaling		
Trusted Developer Utilities Proxy Execution		
Unused/Unsupported Cloud Regions		
Use Alternate Authentication Material		
Valid Accounts		
Virtualization/Sandbox Evasion		
Weaken Encryption		
XSL Script Processing		

Stealthy by design and meant to run without notice or interaction by end users, firmware has become an ideal toolset for attackers seeking to evade detection

Because firmware is often trusted by default in boot or pre-boot sequences, there are few tactics better suited to enabling long-term persistence than firmware compromise

LATERAL MOVEMENT	COLLECTION	COMMAND AND CONTROL
Exploitation of Remote Services	Adversary-in-the-Middle	Application Layer Protocol
Internal Spearphishing	Archive Collected Data	Communication Through Removable Media
Lateral Tool Transfer	Audio Capture	Data Encoding
Remote Service Session Hijacking	Automated Collection	Data Obfuscation
Remote Services	Browser Session Hijacking	Dynamic Resolution
Replication Through Removable Media	Clipboard Data	Encrypted Channel
Software Deployment Tools	Data from Cloud Storage Object	Fallback Channels
Taint Shared Content	Data from Configuration Repository	**Ingress Tool Transfer**
Use Alternate Authentication Material	Data from Information Repositories	Multi-Stage Channels
	Data from Local System	**Non-Application Layer Protocol**
	Data from Network Shared Drive	Non-Standard Port
	Data from Removable Media	Protocol Tunneling
	Data Staged	Proxy
	Email Collection	Remote Access Software
	Input Capture	Traffic Signaling
	Screen Capture	Web Service
	Video Capture	

Firmware-based code has become an ideal tactic for establishing command-and-control channels that are invisible to traditional security tools

EXFILTRATION	IMPACT
Automated Exfiltration	Account Access Removal
Data Transfer Size Limits	**Data Destruction**
Exfiltration Over Alternative Protocol	Data Encrypted for Impact
Exfiltration Over C2 Channel	Data Manipulation
Exfiltration Over Other Network Medium	Defacement
Exfiltration Over Physical Medium	**Disk Wipe**
Exfiltration Over Web Service	Endpoint Denial of Service
Scheduled Transfer	**Firmware Corruption**
Transfer Data to Cloud Account	Inhibit System Recovery
	Network Denial of Service
	Resource Hijacking
	Service Stop
	System Shutdown/Reboot

Firmware is a potent weapon for data destruction, from the erasure of master boot records through highly evolved “wipers” to the impairment and disablement of device components through corrupt commands

Reconnaissance

Reconnaissance is the first of the PRE tactics, and involves a variety of active and passive adversary techniques to identify target assets and vulnerabilities that can be targeted in later phases of the attack. Hardware and firmware play a key role in this technique. Vulnerabilities within the firmware of network devices such as VPNs have become one of the most popular initial access vectors into enterprises, with VPNs attacks seeing an almost **2000% increase from 2020 to 2021**. Firmware vulnerabilities in particular have become a favored initial vector in these attacks because firmware is rarely updated as often as other code, and because exploiting these vulnerabilities can give an attacker full control over the device. Additionally, threat actors can scan to identify externally facing devices including servers, routers, and other infrastructure. Because of the lack of updates mentioned earlier, attackers can often infer the presence of firmware vulnerabilities based on the observed hardware version of the device.

Relevant ATT&CK Techniques	• **Active Scanning** – **Vulnerability Scanning** • **Gather Host information** – **Hardware** – **Firmware** • **Network Security Appliances**
References and Further Reading	• Widespread **exploitation of VPN** vulnerabilities • Publicly discoverable vulnerabilities in **MikroTik routers**
Firmware Countermeasures	Reconnaissance techniques are difficult to directly mitigate as they typically take advantage of systems and information that is inherently exposed to the public. However, this means it is all the more important for organizations to have highly reliable visibility of all their devices and any vulnerabilities they may have, so that any issues can be addressed before they are found by attackers.

Resource Development

The second of the two PRE tactics, Resource Development allows an adversary to acquire infrastructure that will support the active phases of the attack. This could be acquiring servers or botnets that could provide command-and-control as well as a variety of online accounts. It is possible for attackers to implant firmware backdoors in networking devices such as switches as well as servers supporting bare-metal cloud services. These backdoors could be used by attackers to send malicious traffic and could persist even after the server is reprovisioned and used by other customers. Customers should independently verify the integrity of all networking gear and bare metal cloud assets not only to ensure the security of their data and assets, but also to ensure their hardware is not used as part of other attacks.

Relevant ATT&CK Techniques	• **Compromise Infrastructure**
References and Further Reading	• **Backdoors** in bare metal cloud services • **Network Devices Whitepaper**
Firmware Countermeasures	Scan all bare metal cloud assets to verify the integrity of the firmware and identify any vulnerabilities.

Initial Access

Initial Access is the first of the *active* phases of the ATT&CK Tactics, and firmware has proven to be one of the most popular vectors for initial access in real-world attacks. At a high level, firmware can be exploited remotely over networks, in the technology supply chain, or by attackers with physical access to a device.

Firmware within enterprise network devices such as VPNs, routers, and security appliances have become top targets across a wide range of threat actors. The trend was observed in **Russian**, **Chinese**, and **Iranian** state-based threat actors and quickly spread to financially motivated attackers including more than 20 ransomware groups and **all five of the top ransomware groups**. Exploiting vulnerabilities in the firmware and integrated code of these devices provide attackers with direct access into an enterprise with the ability to deliver malware to other users and devices.

Attackers can also gain access by compromising the technology supply chain of a device. Modern supply chains involve dozens of suppliers and subcontractors, providing attackers with many opportunities to compromise a device before it is ever delivered to the enterprise customer. Additionally, supply chains can be compromised via official vendor update processes as was observed in the highly-damaging Solar Winds attacks. In total, the **Breaking Trust** project has identified 139 supply chain attacks and disclosures from the past ten years.

Attackers can also use physical access to compromise systems, often exploiting firmware within system components. For example, vulnerable components can expose devices to **DMA attacks**, which can allow attackers to directly read and write to system memory, and extend control over the execution of the kernel itself.

Relevant ATT&CK Techniques	• **Exploit Public-Facing Application** (e.g. VPN, network devices) • **Supply Chain Compromise** • **External Remote Services** - (e.g. Intel AMT, BMCs) • **Replication Through Removable Media**
References and Further Reading	• **Network Devices Whitepaper** • **Supply Chain Risks** • **DMA Security and Zero Trust** • **Compromise by BIOS Disconnect** • **Remote UEFI Attacks**
Firmware Countermeasures	• Proactively discover all devices within an enterprise including network devices • Proactively scan devices for vulnerabilities and prioritize any vulnerabilities used in real-world attacks • Verify the integrity of all system and component firmware of acquired devices • Verify all firmware updates and monitor firmware behavior following updates • Scan device components for vulnerabilities that would allow physical access attacks.

Execution

Attackers can compromise device boot processes in order to execute malicious code during or after boot. Alternatively, attackers can directly take advantage of component vulnerabilities to gain execution within system memory. Previously referenced **DMA attacks** provide an example where attackers can gain execution directly within system memory. This can allow an attacker to execute kernel code on the system, insert a wide variety of kernel implants, and perform a host of additional activities such as spawning system shells or removing password requirements. And as with all types of code, attackers can use social engineering to trick users into running the attacker's code such as malware that updates firmware or tricking users into performing malicious firmware updates.

Relevant ATT&CK Techniques	• **Replication Through Removable Media** • **User Execution**
References and Further Reading	• **Abusing WPBT to Gain Malicious Code Execution** • **Evil Maid Attacks** • **FinSpy UEFI and MBR Bootkit** • **DMA Attacks**
Firmware Countermeasures	• Scan devices to identify components that are vulnerable to DMA attacks and verify that all vendor DMA protections are properly enabled. • Scan devices for vulnerabilities that could allow attackers to compromise the firmware or boot process. • Verify that all available vendor protections are enabled and properly configured including protections included with UEFI, chipset vendor, OS vendor, and any OEM-specific protections.

Persistence

Persistence has always been one of the key reasons attackers seek to target firmware. Firmware code is integrated into system components themselves, instead of residing on traditional storage drives. This not only hides the attacker's code from traditional security scans but also ensures the code will persist even if a system's drives are completely erased, re-imaged or even replaced.

Attackers continue to escalate their use of firmware implants and backdoors across a wide range of enterprise devices. UEFI implants such as LoJax, MosaicRegressor, and MoonBounce can target any UEFI-based system including laptops and servers. The recent HP iLOBleed implant provides an example of an implant that can target firmware within the powerful baseboard management controllers (BMCs) that enable out of band management of enterprise servers.

TrickBot provides an example of just how common such threats are becoming in the wild and how malware infections can quickly turn into persistent firmware backdoors. Already one of the most widespread and powerful trojans in the industry, TrickBot recently introduced new functionality dubbed **TrickBoot**, which automatically finds weaknesses in devices that allow the malware to establish persistence within an infected device's system firmware. Attackers can also apply this strategy by hiding within unused regions of memory in firmware known as "code caves".

Additionally, attackers can establish persistence by gaining control of a device's boot process to ensure their malicious code is always run during startup. Firmware rootkits and vulnerabilities such as **BootHole** can allow an attacker to execute their code before the operating system is even loaded.

Relevant ATT&CK Techniques	• **Pre-OS Boot Techniques** • **Boot or Logon Initialization Scripts**
References and Further Reading	• **TrickBoot** functionality for ongoing persistence • **HP iLOBleed** • **MoonBounce** • **LoJax** • **MosaicRegressor** • **Code caves** and hiding code in firmware
Firmware Countermeasures	• Scan devices to identify any vulnerable components or misconfigurations that would allow attackers to write to firmware or compromise the boot process. • Proactively verify the integrity of all firmware to identify any signs of implants or firmware compromise, particularly after any known malware incident. • Verify that all available vendor protections are enabled and properly configured including protections included with UEFI, chipset vendor, OS vendor, and any OEM-specific protections.

Privilege Escalation

Security teams often think of privilege escalation in the context of escalating from users privileges (Ring 3) 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 can subvert the kernel and thus possess even higher privileges, often **referred to as Rings -1 through -3**.

Attackers have a variety of techniques at their disposal to escalate privileges to these "negative" rings, both with user or administrator privileges. This involves a top-down approach to compromising firmware that can allow any standard malware infection to escalate privileges to the underlying firmware layer. This can include the use of malicious or **vulnerable device drivers**, the aforementioned BootHole vulnerability, and a number of other firmware or boot vulnerabilities.

Relevant ATT&CK Techniques	• **Exploitation for Privilege Escalation** • **Boot or Logon Initialization Scripts**
References and Further Reading	• Escalation via **BootHole** on devices even when protected by Secure Boot. • Escalation via **driver vulnerabilities** • **SMM Vulnerabilities**
Firmware Countermeasures	• Scan devices to identify any vulnerable components or misconfigurations that would allow attackers to write to firmware or compromise the boot process. • Make sure that devices are using the latest stable OS and bootloaders, and that the dbx revocation database is up to date. • Verify that all available vendor protections are enabled and properly configured including protections included with UEFI, chipset vendor, OS vendor, and any OEM-specific protections. • Scan devices for malicious bootloaders and exploit code

Defense Evasion

Compromising a device's firmware and boot process gives attackers the opportunity to subvert security controls running at the level of the operating system. Firmware also directly controls the functions of device components in ways that may not be visible to the operating system itself. For example, the well-publicized **Equation Group implants** used malicious firmware to create hidden sections within a drive that were invisible to the operating system itself. This also meant that security tools were blind to these areas of the drive, allowing attackers to hide malicious code and evade security scans.

Additionally, rootkits and other malicious code execution within the UEFI environment can allow attackers to directly patch the OS kernel itself. This makes it possible to silently disable security features within the OS or third-party security tools. The **Hacking Team** implant, and the derivative MosaicRegressor UEFI implants, are well-known examples of sophisticated tools that use UEFI rootkits in order to evade traditional security controls.

Relevant ATT&CK Techniques	• **Rootkit** • **Pre-OS Boot**
References and Further Reading	• Exploiting **BootHole** to install bootkits on devices. • Hacking Team and **MosaicRegressor** Implants • Equation Group **SSD implants** • **Demonstration of Firmware-Based Evasion**
Firmware Countermeasures	• Scan devices to identify any vulnerable components or misconfigurations that would allow attackers to write to firmware or compromise the boot process. • Make sure that devices are using the latest stable OS and bootloaders, and that the dbx revocation database is up to date. • Verify that all available vendor protections are enabled and properly configured including protections included with UEFI, chipset vendor, OS vendor, and any OEM-specific protections. • Scan devices for malicious bootloaders and exploit code

Credential Access

Modern devices go to great lengths to protect passwords and credentials on the device. In addition to using firmware threats to steal credentials from system memory (e.g. DMA attacks), attackers can use side-channel analysis to extract credentials from even the most secure components of a system including the Trusted Platform Module. Additionally, firmware attacks against routers and networking gear can allow attackers to perform machine-in-the-middle attacks to steal credentials or intercept MFA challenges.

Relevant ATT&CK Techniques	• **Credentials From Password Stores** • **Adversary-in-the-Middle** • **Exploitation for Credentialed Access**
References and Further Reading	• **TPMFail** to extract private authentication keys • **Router vulnerabilities** that can enable machine-in-the-middle attacks. • **Spectre and Meltdown** to steal passwords • **ROCA vulnerability** to steal keys from TPM
Firmware Countermeasures	• Scan devices for vulnerabilities in TPM, processors, and other components that can enable side-channel attacks or collection of credentials. • Discover all network devices and identify vulnerable or out of date firmware.

Command and Control / Exfiltration

Firmware and hardware components can be used by attackers to establish command-and-control channels that are fully independent of the host operating system. For example, Intel's AMT firmware runs inside the management engine (ME) chip and contains its own network stack independent of the operating system. Attackers such as the **PLATINUM** group have used these capabilities as command-and-control and exfiltration channels that avoid any host-based controls running on the device.

Additionally, compromised firmware on network devices and interfaces can be used to reroute traffic, enabling both command-and-control and data exfiltration.

Relevant ATT&CK Techniques	• **Non-Application Protocol** • **Ingress Tool Transfer**
References and Further Reading	• **PLATINUM** using AMT to bypass Windows firewall • **TrickBot use of MikroTik routers for C2**
Firmware Countermeasures	• Scan devices for vulnerabilities in AMT and ensure AMT features are properly secured. • Identify all network devices and IoT components and scan for vulnerabilities. • Verify the integrity of AMT firmware and network device firmware.

Impact

With access to system firmware or firmware on device drives, attackers can easily destroy data, disable components, or disable the system entirely. The **BlackEnergy** attacks on critical infrastructure in Ukraine in 2014 provided an example of the incredible potential for attackers to cause damage via firmware. Fast forward just a few years and the same concept has gained traction with ransomware such as **EFILock** ransomware, which uses malicious bootloaders to disrupt the boot process and gain control over victim machines, or the more recent **iLOBleed implant**, designed to "brick" HPE servers by manipulating the "integrated lights out" functions in their baseboard management controllers (BMCs).

Relevant ATT&CK Techniques	• **Data Destruction** • **Disk Wipe** • **Firmware Corruption** • **System Shutdown and Reboot**
References and Further Reading	• Using firmware to **remotely brick a server** • **Russian attack on SATCOM networks** • MBR Wipers - **Hermetic Wiper**. **WhisperGate**, **NotPetya** • Impacts of attacks on **Ukrainian powergrid** • **QNAP** ransomware targeting NAS firmware • **VPNFilter attacks**
Firmware Countermeasures	• Scan devices to identify any vulnerable components or misconfigurations that would allow attackers to write to firmware or compromise the boot process. • Proactively verify the integrity of all firmware to identify any signs of implants or firmware compromise.

A NOTE ON OMITTED TACTICS

Readers may notice that some tactics on the preceding tables contain no mention of firmware-based techniques. This is generally due to two different lines of reasoning:

- Some tactics, like Lateral Movement and Exfiltration, do not generally lend themselves to firmware-based techniques
- While some tactics, like Discovery, are more and more frequently executed through firmware, but there is not a significant amount of third-party research and remediation to point readers towards.

As usage of MITRE ATT&CK broadens, and as more real-world evidence and data points containing firmware-based exploits become common knowledge, we will publish updated versions of this paper that round out the way firmware-based techniques are appearing more frequently in the ATT&CK framework along with appropriate technical references.

CONCLUSIONS AND NEXT STEPS

While ATT&CK covers a broad range of activities, this paper seeks to superimpose a firmware-oriented perspective over the framework and introduce some of the many ways firmware can be used and abused in modern attacks and bridge those technical and procedural gaps

However, it is far from an exhaustive list. As some of the most privileged and powerful code on a device, there are virtually unlimited ways that attackers can use firmware maliciously. In **this example**, the Cybersecurity and Infrastructure Security Agency (CISA) has used ATT&CK to map Trickbot malware tactics to a number of deep and hard-to-detect techniques, down to and including UEFI firmware compromise.

Firmware risks extend to virtually every phase of an attack, and it requires security teams to take a consistent and comprehensive approach to firmware security within their organizations. As a result, this examination of MITRE ATT&CK through a "firmware lens" may pose new challenges and concerns for cybersecurity strategists and their teams. However, it is important to note that as threat actors continue to shift their focus to firmware, new security tools are also available that can help security teams incorporate and automate firmware security into their existing practices."Firmware security", for our purposes, means giving organizations the ability to identify, verify and fortify firmware wherever it exists in their infrastructure:

Identify - Automate device discovery and provide ongoing visibility into each device's firmware, from individual firmware configurations to an inventory of the dozens of unique firmware components on every device. Quickly zero in on important components, attributes, or changes that can impact your security posture.

Verify - Proactively identify risks from outdated or vulnerable firmware or device misconfigurations. Verify the integrity of all firmware and detect known and unknown firmware threats including rootkits, implants, and backdoors.

Fortify - Remotely apply patches or updates to proactively mitigate device risks. Receive automated alerts to any firmware integrity changes and drive automated responses via integration with your existing IT and security tools with pre-built integrations with leading SIEMs, vulnerability management, and device management tools.

Built on industry-leading expertise and research, the Eclypsium firmware security platform makes it easy for organizations to integrate firmware security into all phases of their security and risk practices. With a single automated solution organizations can identify and inventory their devices and firmware, find and prioritize vulnerabilities, detect firmware level threats, and take action to fortify their devices and mitigate risks. If you would like to learn more, we recommend the following resources:

- To learn more about firmware security, we recommend the **Definitive Guide to Firmware Security**.
- To stay up to date with latest firmware security news, please subscribe to the **Below the Surface Threat Report**
- To learn more about Eclypsium, please contact our team at **info@eclypsium.com**.

ABOUT ECLYPSIUM

Eclypsium is the enterprise firmware security company. Our comprehensive, cloud-based platform identifies, verifies, and fortifies firmware wherever it exists in your infrastructure and networks: in laptops, servers, network gear, and connected devices. The Eclypsium platform secures against persistent and stealthy firmware threats, provides continuous device integrity, delivers firmware patching at scale, and prevents supply chain implants. Protecting Fortune 500 businesses, global enterprises, and federal agencies, Eclypsium was named a Gartner Cool Vendor in Security Operations and Threat Intelligence, a TAG Cyber Distinguished Vendor, one of the World's 10 Most Innovative Security Companies by Fast Company, a CNBC Upstart 100, a CB Insights Cyber Defender, and an RSACInnovation Sandbox finalist.