Getting a Handle on Firmware Security

These days, firmware is in virtually everything – from servers to laptops to the billions of IoT devices available across the globe. In fact, firmware based on the Unified Extensible Firmware Interface (UEFI) standard is in 80 to 90 percent of the PCs and servers sold worldwide[i].

As hardware and operating system (OS) security become more robust, hackers and researchers look for exploits in other areas such as firmware. Hackers and researchers have targeted platform firmware as a place to embed malware and hide other malicious code that can ultimately compromise a system.

Security researchers are also looking more closely at firmware. Over the last few years, the number of presentations at industry conferences and articles written on firmware vulnerabilities and security have flourished, including topics on:

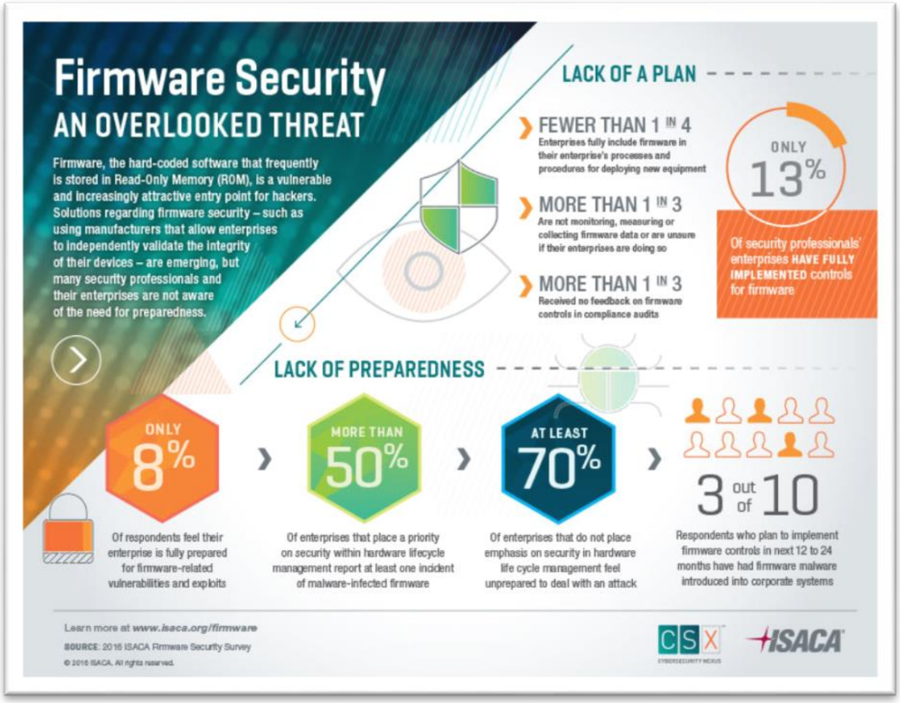
- What you don't know about firmware might get you 0wn3d.
- Firmware is the new Black—Analyzing Past 3 years of BIOS/UEFI Vulnerabilities.
- Betraying the BIOS: Where the Guardians of the BIOS are Failing.

Additionally, a handful of exploits have targeted firmware over the last couple of years. Considering how ubiquitous firmware is in modern systems, it is understandable why a universal approach to firmware security, from both a system and device side, is extremely necessary.

Why is Firmware an Attractive Target?

This is a bit of a loaded question and brings about many opinions, but generally, firmware has become an attractive target to hackers and researchers alike for the following reasons:

- [ii]Firmware security has not always been top of mind for developers and is often overlooked.
- As with all software implementations, there are going to be faults. Since this software stack has been overlooked over time, it stands to reason that there will be undiscovered faults, which could be used for attacks. Uncovering latent bugs is to be expected.
- Not all UEFI implementations in the market faithfully follow the UEFI specification, thus increasing the possibility of malware attacks and other vulnerabilities.



Firmware Security

AN OVERLOOKED THREAT

Firmware, the hard-coded software that frequently is stored in Read-Only Memory (ROM), is a vulnerable and increasingly attractive entry point for hackers. Solutions regarding firmware security – such as using manufacturers that allow enterprises to independently validate the integrity of their devices – are emerging, but many security professionals and their enterprises are not aware of the need for preparedness.

LACK OF A PLAN

FEWER THAN 1 IN 4
Enterprises fully include firmware in their enterprise's processes and procedures for deploying new equipment

MORE THAN 1 IN 3
Are not monitoring, measuring or collecting firmware data or are unsure if their enterprises are doing so

MORE THAN 1 IN 3
Received no feedback on firmware controls in compliance audits

ONLY 13%
Of security professionals' enterprises HAVE FULLY IMPLEMENTED controls for firmware

LACK OF PREPAREDNESS

ONLY 8%
Of respondents feel their enterprise is fully prepared for firmware-related vulnerabilities and exploits

MORE THAN 50%
Of enterprises that place a priority on security within hardware lifecycle management report at least one incident of malware-infected firmware

AT LEAST 70%
Of enterprises that do not place emphasis on security in hardware life cycle management feel unprepared to deal with an attack

3 out of 10
Respondents who plan to implement firmware controls in next 12 to 24 months have had firmware malware introduced into corporate systems

Learn more at www.isaca.org/firmware

SOURCE: 2016 ISACA Firmware Security Survey

© 2016 ISACA. All rights reserved.

- Firmware attacks are hard to detect by current antivirus and other security software because firmware resides at architectural levels of the device that are not usually accessible to current tools.
- If malware can control system firmware, it gains full access to the system.
- Malware hidden in firmware is hard to erase and can survive reboots and fresh installs of an operating system.

"Looking ahead, firmware security needs to be more important to the technology ecosystem. The consequences of not paying attention will prove detrimental," said Mark Doran, UEFI Forum President.

What Are Firmware-Specific Threat Areas?

What areas are the most vulnerable in firmware? Below are some areas to note at a high level, but additional details can be found here: UEFI Firmware Security Concerns and Best Practices.

- Maliciously crafted input – Buffer overflows to inject malware
- Elevation of privilege – System Management Mode (SMM) code injection
- Data tampering – Modifying UEFI variables (SecureBoot, Configuration, etc.)
- Unauthorized access to sensitive data – Disclosure of System Management Random Access Memory (SMRAM) contents
- Information disclosure – SMM rooted malware; "secrets" left in memory
- Denial of Service – serial peripheral interface (SPI) flash corruption to "brick" the system
- Key Management – Private Key Management for signed capsule updates

Five Firmware Security Tips for 2018

The platform security industry seems to be nearing a tipping point where security will need to become a primary design consideration and industry cooperation even more necessary. Factoring in the following tips is a path to getting across the current security chasm.

1. Mitigate risks by taking a proactive approach to firmware security.

    - The first step is making sure the 'minimum requirements' in the UEFI specification are implemented. The minimum requirements can be found in Version 2.7 Errata A of the UEFI spec under section "2.6.1 Required Elements" starting on page 63.
    - Current UEFI specifications and implementations incorporate security features, such as the optional protocol "UEFI Secure Boot," key signing and signature checking. Features such as these strengthen the security of the UEFI booting platform.
    - Known threats do not always go away forever; facets can come back around. Maintaining a running list of exploits as they come out is a worthy practice to consider.

2. Consider incorporating additional preemptive practices such as

    - Incorporate industry standard testing tools, such as CHIPSEC and automated code analysis.
    - Perform targeted code reviews.
    - Develop security test tools and integrate into the QA process.
    - Review disclosures and guidelines, and verify implementations.
    - When feasible, back port security fixes to previous codebases.

- Perform fuzz and boundary testing.
- Investigate emerging specifications, such as NIST SP 800-193.
- When there are changes to code related to a security vulnerability, always re-test for the vulnerability (if possible).

3. Don't ignore system update requests. However daunting or annoying they may seem, they are necessary to keep your system's health in check.

- First, understand the bigger implications such as customers' uncertainty can lead to mistrust toward the manufacturer or even the supply chain.
- Work with customers to educate them on the importance of firmware updates and how they should manage them.
- Monitor the EDK2 codebase for important security fixes.
- Monitor social media for publicly disclosed findings.
- Apply all patches to the Capsule Update drivers. Secure Capsule Updates rely on proper signing, private key management, validation, and rollback protection.
- More information about Capsule Updates can be found at www.uefi.org/learning_center/industryresources.

Helpful NIST Resources

- NIST SP 800-107 provides guidelines for hash algorithm usage
- NIST SP 800-57 provides guidelines for key management
- NIST SP 800-147(b) provides guidelines for secure BIOS Updates

4. Now is not the time to ignore this growing security threat area. Through collaboration on some simple steps, the platform technology community can make a big difference.

- Everyone who provides pre-OS code, including firmware Option ROM code and EFI applications, needs to follow similar steps to validate their implementations.
- Work together to stay well informed. Collaborate on improvements to processes such as communication, reporting and updating.
- Add to the conversation around device security via the UEFI Forum.
- Become a Contributor member for access to UEFI work in progress.
- Select a corporate technical security representative and have them participate with the UEFI Spec Security Sub-team.
- The UEFI Security Response Team (USRT) will work with UEFI members to enhance and coordinate responses to actual and perceived vulnerabilities.
    - If you have information about a UEFI-based firmware security issue or vulnerability, please send an e-mail to security@uefi.org.
- UEFI Forum has some of the industry's best security experts as members.

5. Consider incorporating other simple steps that can help stay ahead of firmware security including:

- Consider reporting known vulnerabilities via the CVE system, https://cve.mitre.org/.
- Make sure that you have NDAs and arrangements to receive security notifications from silicon providers, operating system vendors, etc.
- Consider participation in the Tianocore open-source development project and its security team.

While there is no silver bullet that can protect potential targets from attacks all of the time, approaching firmware security proactively should be at the top of everyone's New Year resolutions list. At the minimum, everyone needs to participate in the discussion on how to coordinate and regularly encourage firmware-related security fixes and other updates.

Firmware threats will persist as long as security and IT managers and manufacturers are not paying attention to it. The community can come together to enact change by looking at firmware security as a priority versus an afterthought; if this not done, the industry is in for more challenges. Firmware security will become a watchword in the very near future, and missing this now may be painful (in a business sense) in the future. Let's make 2018 the year for strengthening firmware security.

[i] UEFI Forum

[ii] Infographic from ISACA at https://www.isaca.org/Knowledge-Center/Research/ResearchDeliverables/Pages/Firmware-Security-Risks-and-Mitigation.aspx.