Universal Scalable Firmware: Security Aspects of an Evolutionary Approach to System Firmware

Vincent Zimmer and Jiewen Yao (Intel)

UEFI 2023 Virtual Plugfest

Jiewen Yao

- **Jiewen Yao** is a principal engineer in the Intel Software and Advanced Technology Group. He has been engaged as a firmware developer for over 15 years. He is a member of the UEFI Security Sub Team, and co-chairing TCG PC Client Working Group and DMTF SPDM Code Task Force.

Vincent Zimmer

- **Vincent Zimmer** is a senior principal engineer in the Intel Software and Advanced Technology Group. He has been engaged w/ firmware for over 30 years and presently leads the UEFI Security sub team.

Vincent Zimmer
Intel

More Questions?

Following today's webinar, join the live, interactive WebEx Q&A for the opportunity to chat with the presenters

Visit this link to attend: https://bit.ly/3xklPQR
Meeting number: 2554 924 4620
Password: UEFIForum (83343678 from phones and video systems)

Agenda

- Universal Scalable Firmware (USF) Overview
- Security Hardening
- OpenSSL 1.1 EOL Update
- Commercial National Security Algorithm (CNSA) Compliance

What is Universal Scalable Firmware (USF)?

- Multi-layer view of the firmware stack
    - Interfaces for boot environments (payload), platform code (EDKII, coreboot, slim bootloader, etc)
- Interfaces and infrastructure at different levels
- https://github.com/universalscalablefirmware for code and spec sources
- https://universalscalablefirmware.groups.io/g/discussion for community discussions
- https://www.youtube.com/watch?v=oEBtWsBZve4&list=PLehYIRQs6PR6J9Zf6CajwsFkAHedDXjLI&index=13 for past meetings
- https://universalscalablefirmware.github.io/documentation/ for the ‘compiled’ specification
- Past prezo https://www.osfc.io/2021/talks/an-evolutionary-approach-to-system-firmware/

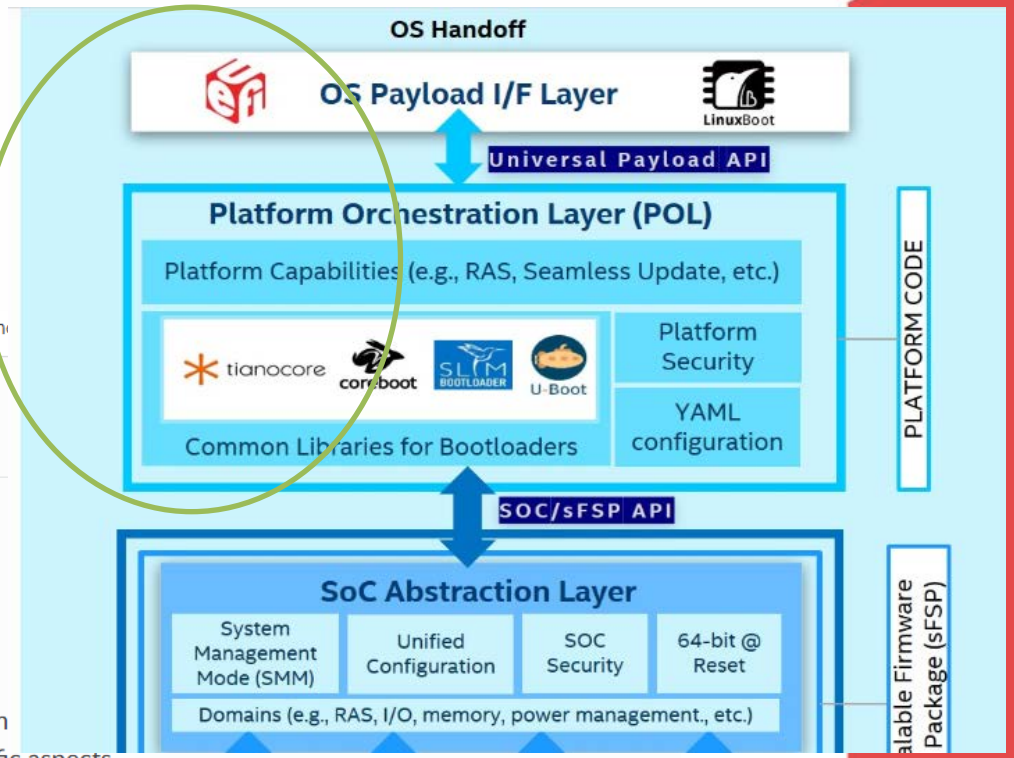
Today's Talk – Security Impacts of USF to UEFI and EDKII ecosystem

1. Universal Scalable Firmware (USF) Specification

Notices and Disclaimers

1. Universal Scalable Firmware (USF) Specification
2. Universal Payload
3. Platform Orchestration Layer (POL)
4. Runtime
5. Security
   - 5.1. Security Overview
   - 5.2. Vulnerability Mitigation Strategy
6. Debug

Today's focus



Suggested Sites | Imported From IE

5. Security

5. Security

5.1. Security Overview

There are various security considerations for th
various overall concerns and technology specific aspects.

5.1.1. Firmware Resiliency - Protection

5.1.1.1. Critical Resource Lock (hardware)

The platform shall always lock the important resource before it exits the platform manufacture phase.

USF Security Topic Areas

- 5. Security
  - 5.1. Security Overview
    - 5.1.1. Firmware Resiliency - Protection
    - 5.1.2. Firmware Resiliency - Detection
    - 5.1.3. Firmware Resiliency - Recovery
    - 5.1.4. Measurment and Attestation
    - 5.1.5. DMA Protection
    - 5.1.6. Cryptography Agility
  - 5.2. Vulnerability Mitigation Strategy
    - 5.2.1. Eliminate Vulnerability
    - 5.2.2. Break Exploitation
    - 5.2.3. Contain Damage
    - 5.2.4. Limit Attack Window

Reference: https://universalscalablefirmware.github.io/documentation/5_security.html

Continue Strengthening the Supply Chain

https://uefi.org/sites/default/files/resources/Traceable%20Firmware%20Bill%20of%20Materials%20-%2020211207%20-%20007.pdf

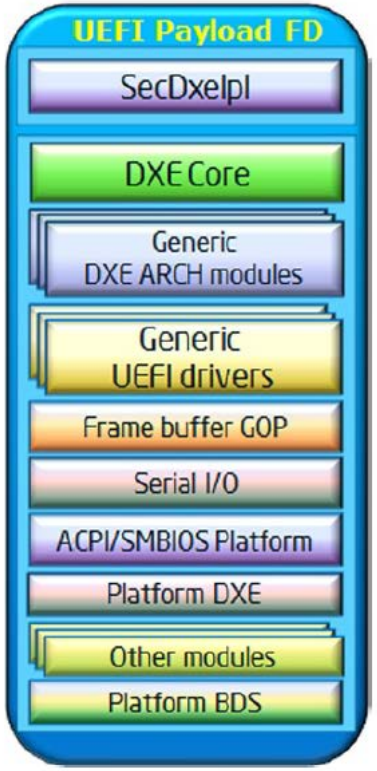
- Type-II-B indicates the one loaded from peripheral device, such as NIC, NVMe, Graphic Card.

For Type-I firmware, the component provider may provide a reference integrity manifest (RIM) for this specific component.

Intel FSP 2.x measurement and attesation defines a mechanism to report FSP manifest according to TCG PC Client Reference Integrity Manifest Specification. The RIM format could be SWID or CoSWID.

The universal payload should use SWID or CoSWID with below information:

Element	Attribute	Required	Description
Software Identity	Name	Required	Name of the Universal payload
	Version	Required	Version of the Universal payload



Security Assurance

Tactics	Method	Example
Eliminate Vulnerability	Reduce Attack Surface	• Remove Unnecessary Interface, e.g. SMI handler, private auth variable. • Adopt Firmware Security Best Practice (EDKII security docs, OCP Secure Firmware Development Best Practices)
Break Exploitation	• Data Execution Prevention (DPE) • Control Flow Guard (CFG) • Address Space Layout Randomization (ASLR)	• Non-executable Data Page. Read-only Code page. • Stack Cookie • Intel Control Flow Enforcement Technology (CET) – Shadow Stack (SS), Indirect Branch Tracking (IBT). • ARM Pointer Authentication Code (PAC), Branch Target Identification (BTI). • ASLR in DXE/SMM
Contain Damage	Deprivilege	Ring-3 Third Party Option ROM. Ring-3 OEM SMM
Limit Attack Window		• Live Patching Runtime Component • Firmware Vulnerability Scan • Supply chain - firmware manifest (SBOM)

Reference: https://universalscalablefirmware.github.io/documentation/5_security.html

Possible Security Hardening

- **Data Execution Protection (DEP)**
- **& Arbitrary Code Guard (ACG)**
  - Image Protection
  - Non-Executable Memory protection
  - OS Loader Protection
  - SMM Code Access Check
- **NULL pointer detection**
- **Address Space Layout Randomization (ASLR)**
  - Data Buffer Shift
  - Image Shuffle
- **Buffer Overflow Detection**
  - Heap Guard
  - Stack Cookie
  - Address Sanitizer
- **Misc Runtime Check**
  - Undefined Behavior Sanitizer (Type Cast)
  - Memory Sanitizer (Uninitialized Access)
- **Control Flow**
  - Backward: CET Shadow Stack, ARM PAC
  - Forward: CET IBT, ARM BTI

Reference: https://github.com/jyao1/SecurityEx/blob/master/Summary.md

However …

- UEFI / PI / APCI are interface specifications
- How do we let end users know what protection is available?

Example

- **Windows SMM Security Mitigation Table (WSMT)**
  - Allows system firmware to confirm to the operating system that certain security best practices have been implemented in SMM
  - https://download.microsoft.com/download/1/8/a/18a21244-eb67-4538-baa2-1a54e0e490b6/wsmt.docx
- **Windows Hardware Security Test Interface (HSTI)**
  - Specifies a standard test interface for proprietary platform security technologies that enforce the Secure Boot promise
  - https://learn.microsoft.com/en-us/windows-hardware/test/hlk/testref/hardware-security-testability-specification
- **TCG Platform Firmware Integrity Measurement**
  - Platform Firmware Assertions can be reported in the platform certificate.
  - E.g. HardwareSRTM, SecureBoot, sp800-147, sp800-193, fwSetupAuthLocal, SMMProtection, fwKernelDMAProtection, etc.
  - https://trustedcomputinggroup.org/resource/tcg-pc-client-platform-firmware-integrity-measurement/

Request For Comment

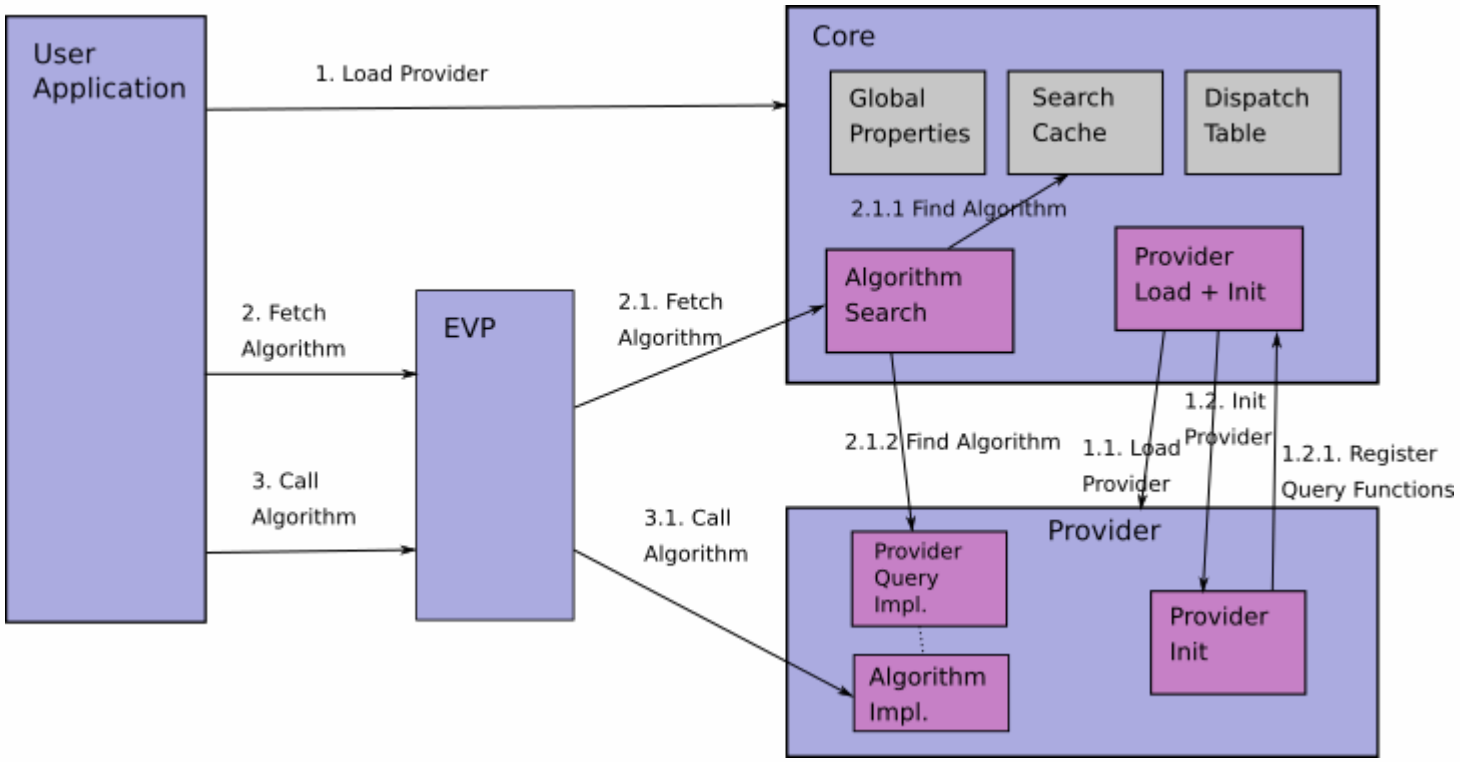
- **Platform Integrity Mitigation Table (PIMT)**
  - Specifies the mitigation applied in the system firmware
    - DEP.CodeProtection, DEP.NonExecutableData, NULLPointerProtection, ASLR.BufferShift, ASLR.ImageShuffle, CFG.Backward, CFG.Forward
  - Could be ACPI table or GUIDed UEFI system table
    - ACPI better since all of ACPI most common across all platform implementations (slim, core, and EDKII)

Openssl 1.1 EOL

- Openssl 1.1 will be at EOL on September 2023
- https://www.openssl.org/policies/releasestrat.html

- EDKII needs a replacement.
- https://github.com/tianocore/edk2-staging/tree/OpenSSL11_EOL

Openssl 3.0 Design



Source: https://www.openssl.org/docs/OpenSSL300Design.html

Candidate - openssl 3.0

- Good option, but big
- Initial investigation shows size is doubled
- Will break the existing platform

- https://bugzilla.tianocore.org/show_bug.cgi?id=3466
- https://github.com/kraxel/edk2/tree/archive/openssl3-v1
- https://edk2.groups.io/g/devel/topic/87479913

Candidate - mbedtls

- Small, but missing features
- Missing SHA3 (Parallel Hash), SMx, etc.

- https://bugzilla.tianocore.org/show_bug.cgi?id=4177
- https://github.com/jyao1/edk2/tree/DeviceSecurity/CryptoMbedTlsPkg

Candidate - Other

- **Intel IPP**
  - https://software.intel.com/en-us/intel-ipp
  - no certificate support, no TLS
- **Libsodium**
  - https://doc.libsodium.org/
  - no certificate support, no TLS
- **BoringSSL**
  - https://github.com/google/boringssl
  - "We don't recommend that third parties depend upon it"
- **WolfSSL**
  - https://www.wolfssl.com/
  - GPL license
- **BearSSL**
  - https://bearssl.org/
  - beta-quality software

Latest Result – openssl 3.0

- https://github.com/tianocore/edk2-staging/blob/OpenSSL11_EOL/CryptoPkg/Readme-OpenSSL3.0.md

Driver	1.1.1	3.0	percent
CryptoDxeFull	1014	1578	57%
CryptoPei	386	794	106%
CryptoPeiPreMem	31	417	1245%
CryptoDxe	804	1278	59%
CryptoSmm	558	986	77%

Driver	1.1.1	3.0	percent
CryptoPei	386	398	3.1%
CryptoPeiPreMem	31	31	0%
CryptoDxeFull	1014	1031	1.7%
CryptoDxe	804	886	10.1%
CryptoSmm	558	604	8.2%

Acknowledgement

-- Gerd Hoffmann kraxel@redhat.com, Li, Yi1 yi1.li@intel.com , Ard Biesheuvel ardb@kernel.org

Latest Result – mbedtls 3.0

- https://github.com/tianocore/edk2-staging/blob/OpenSSL11_EOL/CryptoPkg/ReadmeMbedtls.md
- **PKCS7**: included in mbedtls 3.0.
- **SHA3**: under development - https://github.com/Mbed-TLS/mbedtls/pull/5820 , https://github.com/Mbed-TLS/mbedtls/pull/5822

Driver	OpenSSL	MbedTLS
PEI	387Kb	162Kb
PeiPreMem	31Kb	58Kb
DXE	804Kb	457Kb
SMM	558Kb	444Kb

Acknowledgement

-- Hou, Wenxing wenxing.hou@intel.com , Marvin Häuser mhaeuser@posteo.de

Request For Comment

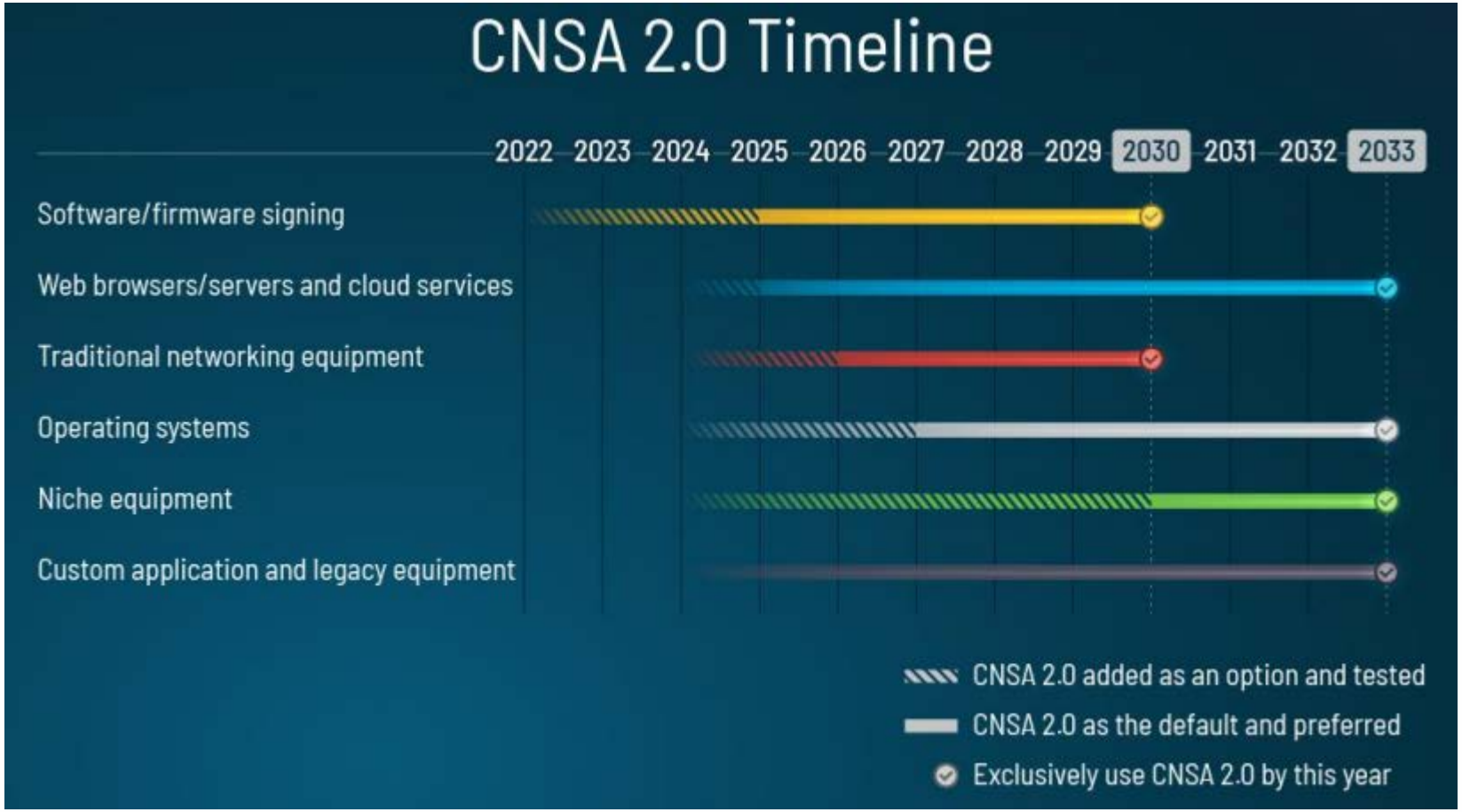
- **Openssl 3.0**
  - Research on how to reduce size to make it fit to the firmware
- **Dual Mode**
  - EDKII supports both openssl 3.0 and mbedtls two instances
  - Platform chooses the library + feature based on the need

Commercial National Security Algorithm (CNSA) 1.0 Compliance

- **CNSA 1.0**
  - Sym: AES-256, SHA-384
  - Asym: ECDH/ECDSA-NIST-P384, RSA-3072 above
  - https://media.defense.gov/2021/Sep/27/2002862527/-1/-1/0/CNSS%20WORKSHEET.PDF

- **UEFI/EDKII – support crypto agility**
  - UEFI-2.10 defines Firmware/OS Crypto Algorithm Exchange.
  - https://uefi.org/specs/UEFI/2.10/32_Secure_Boot_and_Driver_Signing.html?highlight=ecdsa#firmware-os-crypto-algorithm-exchange
  - Support new algorithms with compatibility consideration.
  - *CryptoIndications*: Allows the OS to request the crypto algorithm to BIOS.
  - *CryptoIndicationsSupported*: Allows the firmware to indicate supported crypto algorithm to OS.
  - *CryptoIndicationsActivated*: Allows the firmware to indicate activated crypto algorithm to OS.

CNSA 2.0 Guideline



Reference: https://www.nsa.gov/Press-Room/News-Highlights/Article/Article/3148990/nsa-releases-future-quantum-resistant-qr-algorithm-requirements-for-national-se/

Industry Preparation - PQC

- **Openssl 3.0**
  - Open Quantum Safe (OQS) project support openssl 3.0
    - https://github.com/open-quantum-safe/openssl/tree/OQS-OpenSSL3
  - OQS provider
    - https://github.com/open-quantum-safe/oqs-provider

- **Mbedtls**
  - Roadmap: https://mbed-tls.readthedocs.io/en/latest/roadmap/
  - Future:
    - Post Quantum Crypto

CNSA 2.0 Compliance

- **CNSA 2.0 (Post Quantum Crypto)**
  - Firmware Image Signing/Verification: XMSS/LMS
  - General Signing/Verification: Dilithium
  - General Key Exchange: Kyber
  - https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF

- **UEFI/EDKII – Request For Comment**
  - Define more bit to support CNSA algorithm.
  - https://bugzilla.tianocore.org/show_bug.cgi?id=4087
  - When to use XMSS/LMS?
  - When to use Dilithium?

Asymmetric Cryptography in System Firmware

Usage	Category	Feature	Standard	Algorithm	Comment
Code Signing Verification	Secure Boot	**UEFI Secure Boot**	**UEFI**	**PKCS7(RSA)**	Signed one time – when the image is created
		PI Signed FV/Section	**UEFI PI**	**PKCS7(RSA) / RSA**	
		Intel Boot Guard (Verified Boot)		RSA / SM2	
		Platform Firmware Resilience (PFR)		RSA/ECDSA	
	Update	**UEFI FMP Capsule Update**	**UEFI**	**PKCS7(RSA)**	
		Intel BIOS Guard		RSA	
	Recovery	EDKII Signed Recovery with FMP Cap	**EDKII**	**RSA**	
Data Signing Verification	Update	**UEFI Auth Variable Update**	**UEFI**	**PKCS7(RSA)**	Signed one time, when the data is created
Authentication	Device	SPDM Device Authentication	**DMTF**	**RSA/ECDSA**	Runtime Signing based upon challenge
		SPDM Device Measurement Verification	**DMTF**	**RSA/ECDSA**	
Secure Session Establishment	Device	SPDM Session	**DMTF**	**ECHDE**	Key Exchange with SIGMA protocol
	Network	HTTPS Boot (TLS)	**IETF**	**ECDHE**	

Reference: https://uefi.org/sites/default/files/resources/Post%20Quantum%20Webinar.pdf

Questions?

More Questions?

Following today's webinar, join the live, interactive WebEx Q&A for the opportunity to chat with the presenters

Visit this link to attend: https://bit.ly/3xklPQR
Meeting number: 2554 924 4620
Password: UEFIForum (83343678 from phones and video systems)

Thanks for attending the UEFI 2023 Virtual Plugfest

For more information on UEFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by