HW Security

Cryptography: course for master's degree in EDGE COMPUTING

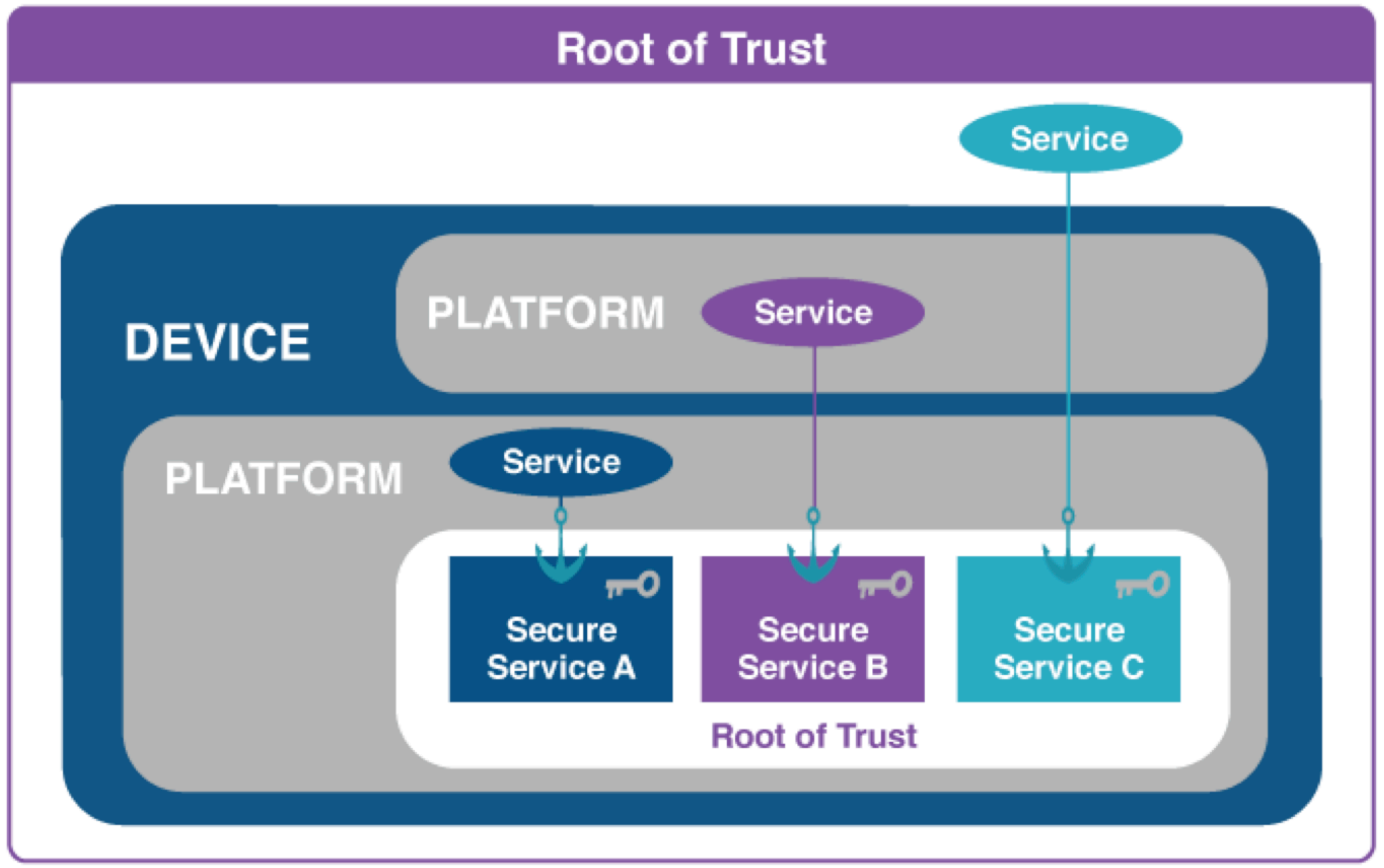
Michał Melosik, PhD

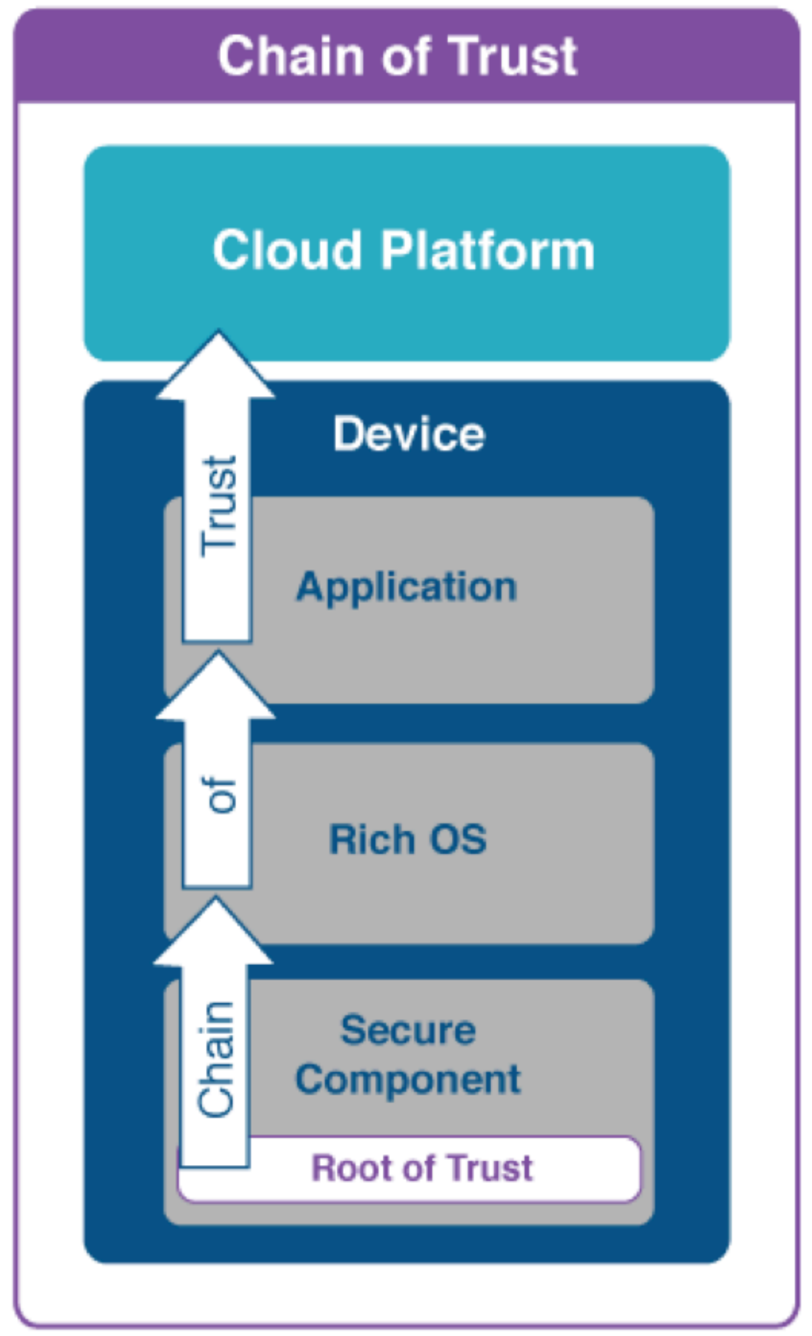
Lecture outline

1. Root of Trust (ROT) and Chain of Trust
2. Secure boot: FW signing, SVN, Key revocation
3. SGX, TDX
4. Threat Detection Technology (TDT)
5. Device attestation
6. PUF
7. Supply chain security
8. Security & Ethic

Root of Trust, Chain of Trust

How others explain it?

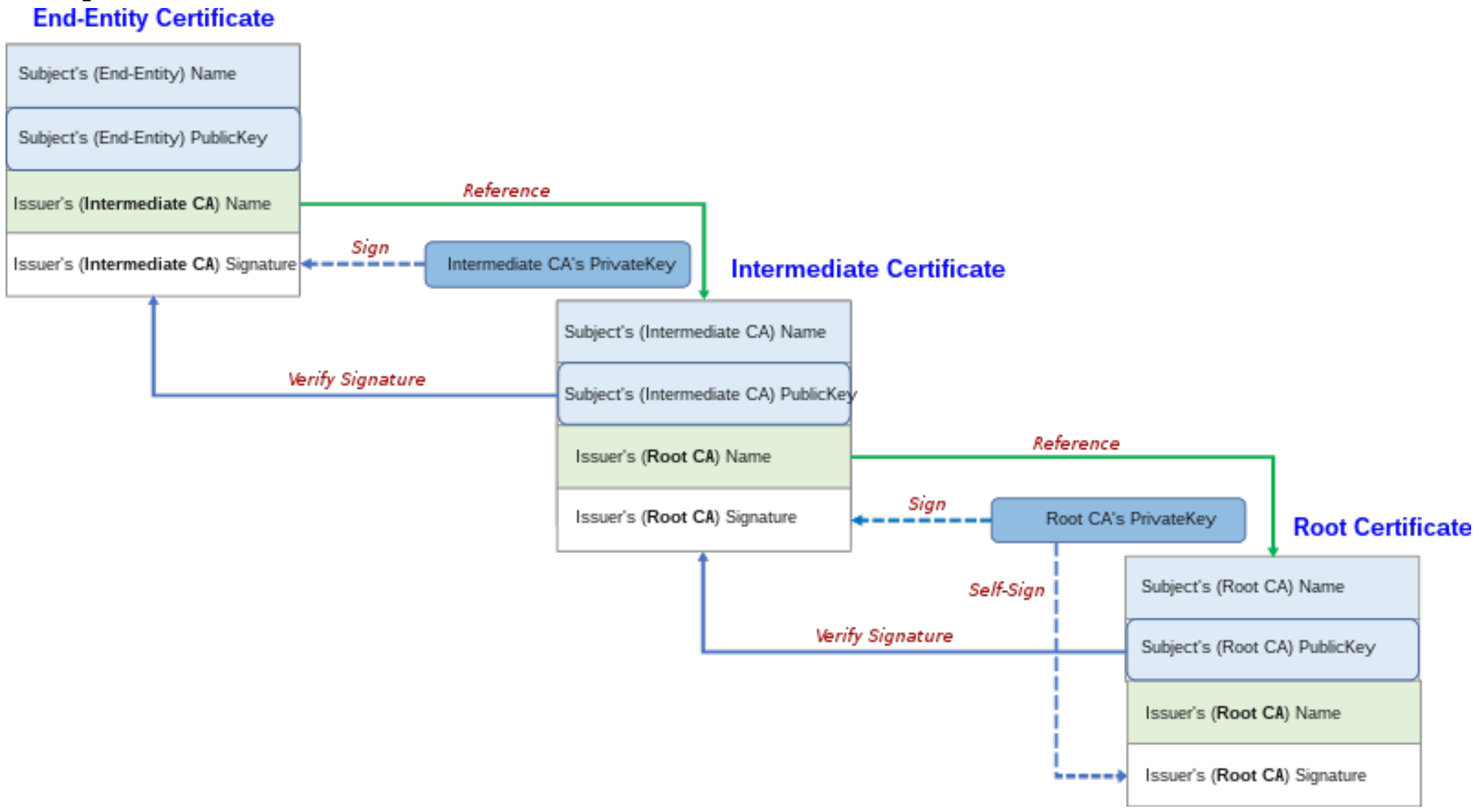




Source (15.12.22): https://globalplatform.wpengine.com/wp-content/uploads/2018/05/GlobalPlatform-Chains-of-Trust-16May2018.pdf

Root of Trust, Chain of Trust

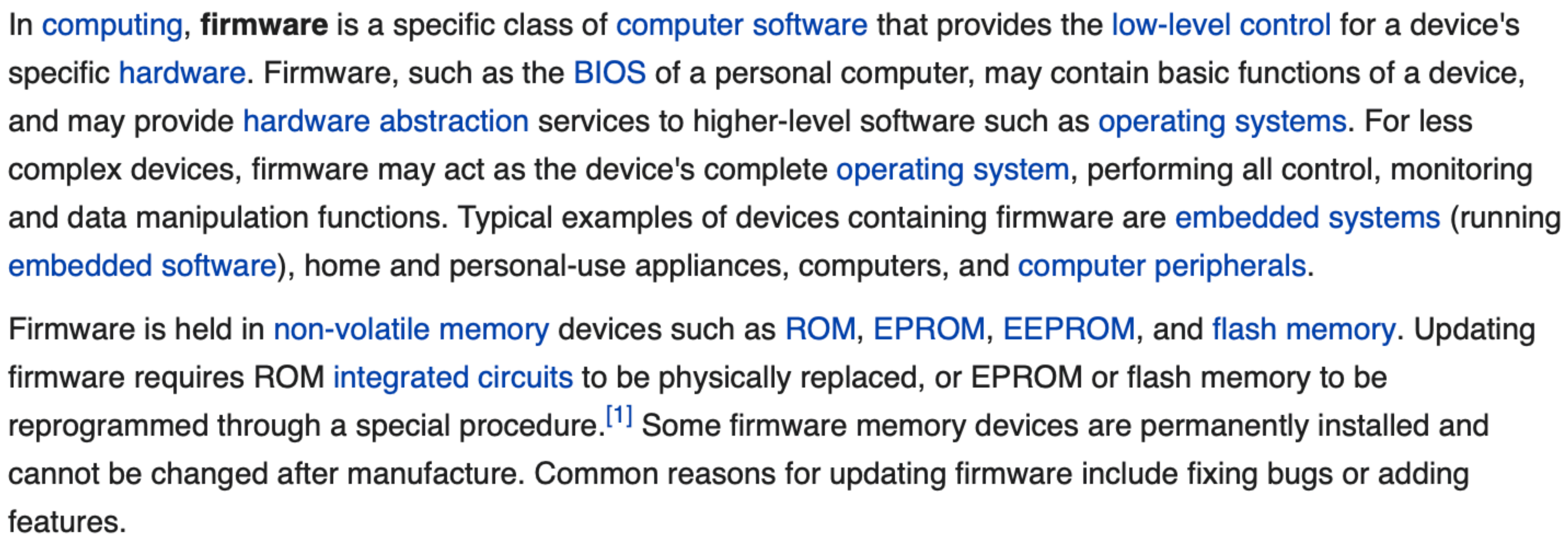
Idea examples from GlobalPlatform


End-Entity Certificate
Subject's (End-Entity) Name
Subject's (End-Entity) PublicKey
Issuer's (Intermediate CA) Name
Issuer's (Intermediate CA) Signature
Reference
Sign
Intermediate CA's PrivateKey
Intermediate Certificate
Verify Signature
Subject's (Intermediate CA) Name
Subject's (Intermediate CA) PublicKey
Issuer's (Root CA) Name
Issuer's (Root CA) Signature
Reference
Sign
Root CA's PrivateKey
Root Certificate
Self-Sign
Verify Signature
Subject's (Root CA) Name
Subject's (Root CA) PublicKey
Issuer's (Root CA) Name
Issuer's (Root CA) Signature

Source(17.11.22): https://en.wikipedia.org/wiki/Chain_of_trust

Firmware

What does wikipedia have to say about it?



In computing, **firmware** is a specific class of computer software that provides the low-level control for a device's specific hardware. Firmware, such as the BIOS of a personal computer, may contain basic functions of a device, and may provide hardware abstraction services to higher-level software such as operating systems. For less complex devices, firmware may act as the device's complete operating system, performing all control, monitoring and data manipulation functions. Typical examples of devices containing firmware are embedded systems (running embedded software), home and personal-use appliances, computers, and computer peripherals.

Firmware is held in non-volatile memory devices such as ROM, EPROM, EEPROM, and flash memory. Updating firmware requires ROM integrated circuits to be physically replaced, or EPROM or flash memory to be reprogrammed through a special procedure.[1] Some firmware memory devices are permanently installed and cannot be changed after manufacture. Common reasons for updating firmware include fixing bugs or adding features.

Code signing

What does wikipedia have to say about it?

Code signing is the process of digitally signing executables and scripts to confirm the software author and guarantee that the code has not been altered or corrupted since it was signed. The process employs the use of a cryptographic hash to validate authenticity and integrity.[1] Code signing was invented in 1995 by Michael Doyle, as part of the Eolas WebWish browser plug-in, which enabled the use of public-key cryptography to sign downloadable Web app program code using a secret key, so the plug-in code interpreter could then use the corresponding public key to authenticate the code before allowing it access to the code interpreter's APIs. [2]

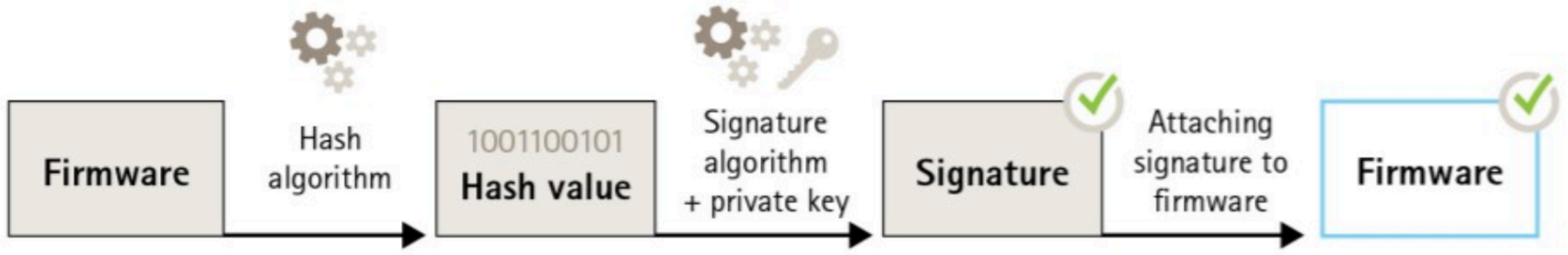
Code signing can provide several valuable features. The most common use of code signing is to provide security when deploying; in some programming languages, it can also be used to help prevent namespace conflicts. Almost every code signing implementation will provide some sort of digital signature mechanism to verify the identity of the author or build system, and a checksum to verify that the object has not been modified. It can also be used to provide versioning information about an object or to store other meta data about an object.[3]

The efficacy of code signing as an authentication mechanism for software depends on the security of underpinning signing keys. As with other public key infrastructure (PKI) technologies, the integrity of the system relies on publishers securing their private keys against unauthorized access. Keys stored in software on general-purpose computers are susceptible to compromise. Therefore, it is more secure, and best practice, to store keys in secure, tamper-proof, cryptographic hardware devices known as hardware security modules or HSMs.[4]

FW signing

What is the significance of this? Secure boot.

The signed signature is added to the firmware when it is distributed. The processor calculates its own hash and will only load an image whose hash matches that signed by a certificate it trusts.



Firmware
Hash algorithm
1001100101
Hash value
Signature algorithm + private key
Signature
Attaching signature to firmware
Firmware

Source (1.12.22): https://store.chipkin.com/articles/signed-firmware

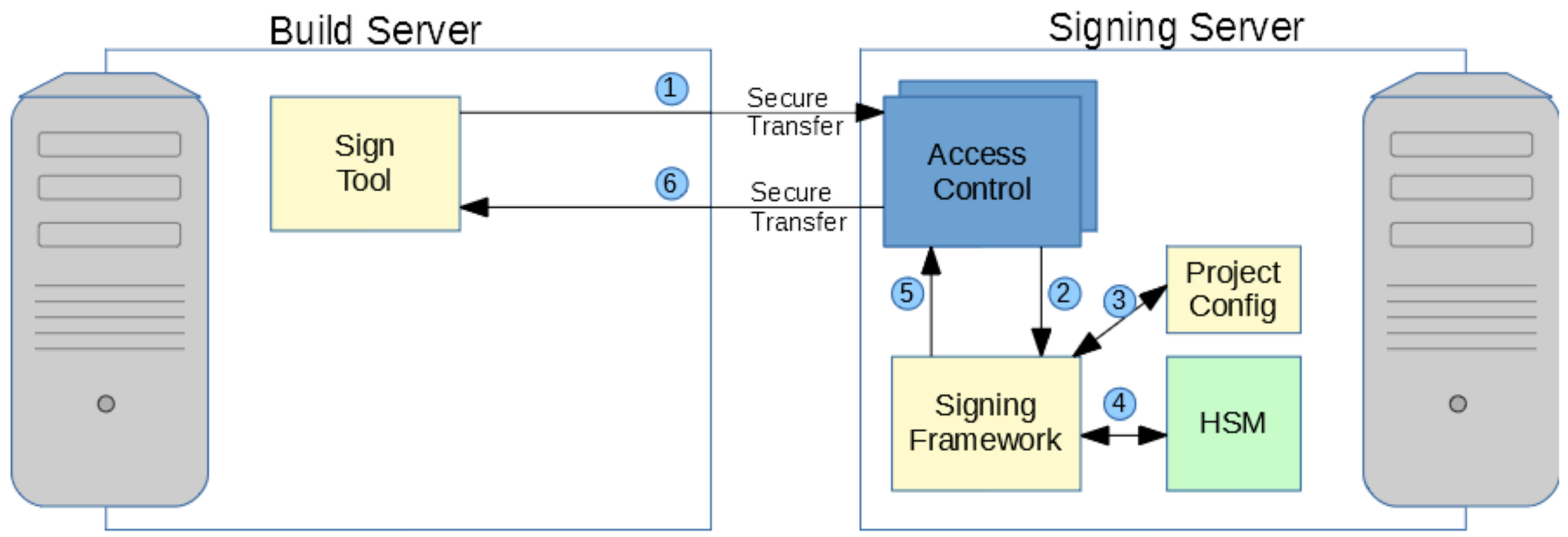
Firmware Signing Process

By Open Compute Project

OPEN Compute Project

BEST PRACTICES FOR FIRMWARE CODE SIGNING

Firmware Signing Process



1) Sign tool uploads request to Signing Server Access Control
2) Signing framework polls Access Control Process and detects request
3) Signing framework validates user authorizations against project configuration
4) Sign tool interacts with HSM to create signature
5) Signing framework writes response into users Access Control response queue
6) Sign tool polls Access Control Process and downloads response as a secure transfer

Source (30.11.22): https://www.opencompute.org/documents/ibm-white-paper-best-practices-for-firmware-code-signing

FW signing

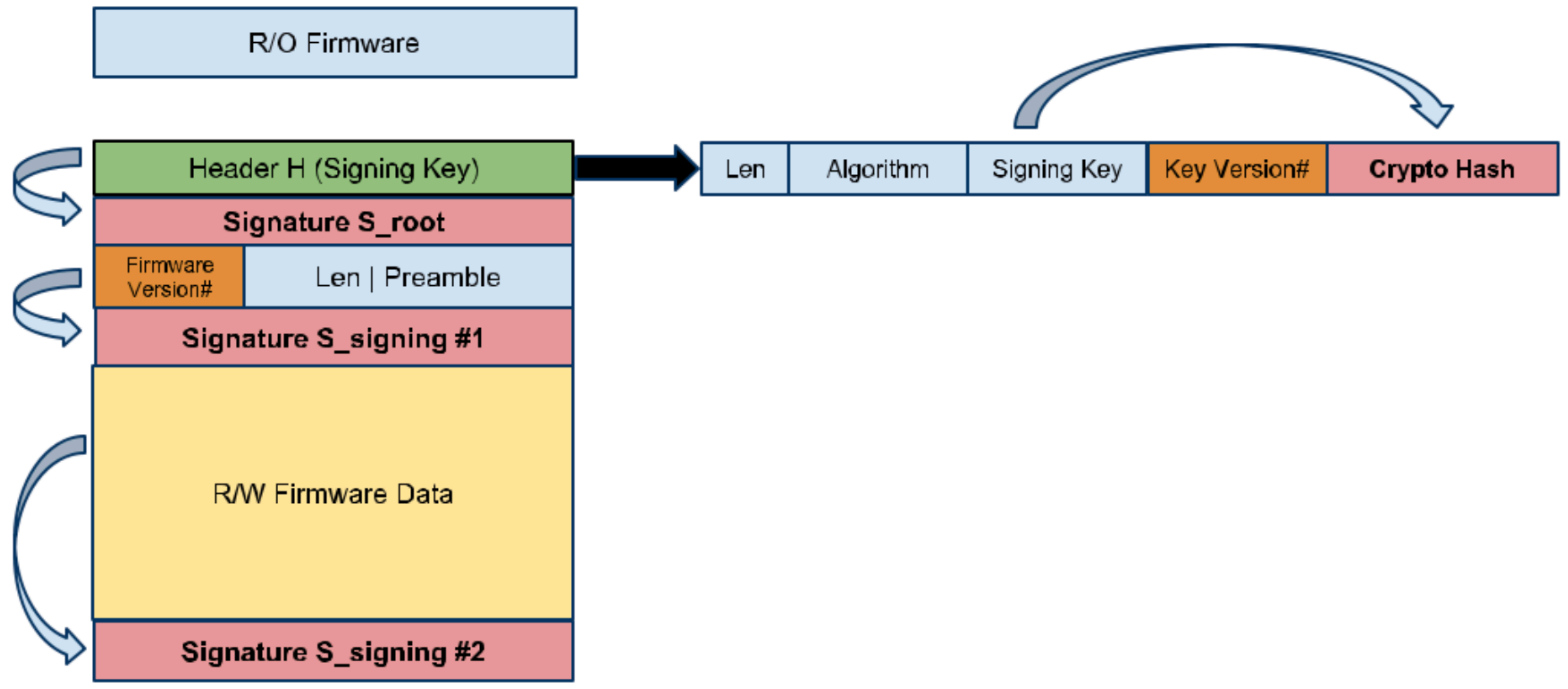
Some related examples

Analysis of an example from the website:

https://www.intel.com/content/www/us/en/docs/programmable/683642/21-3/authentication-and-authorization-s10-fm.html

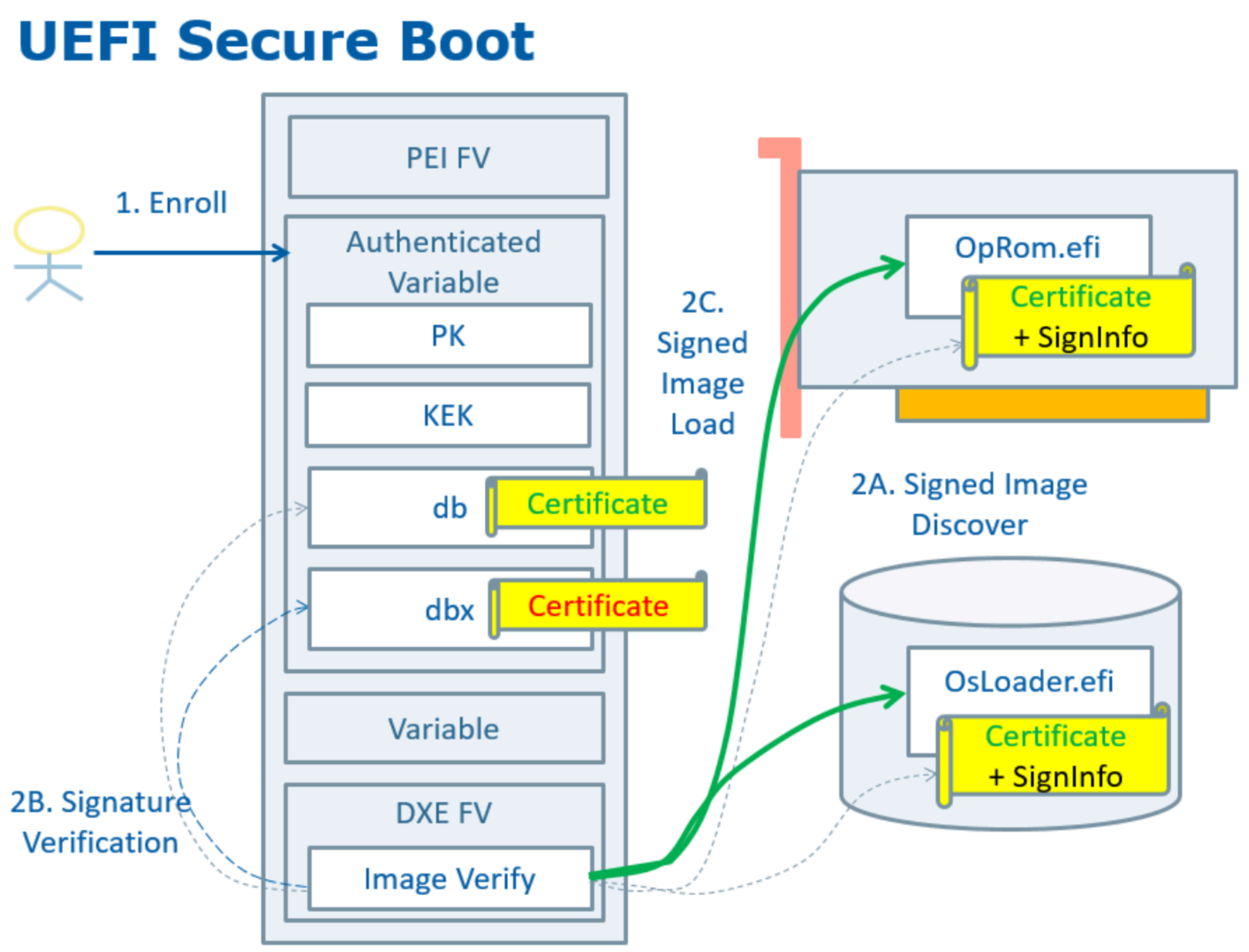
External example

Firmware Verified Boot Crypto Specification - The Chromium Projects



Source (01.12.22): https://www.chromium.org/chromium-os/chromiumos-design-docs/verified-boot-crypto/

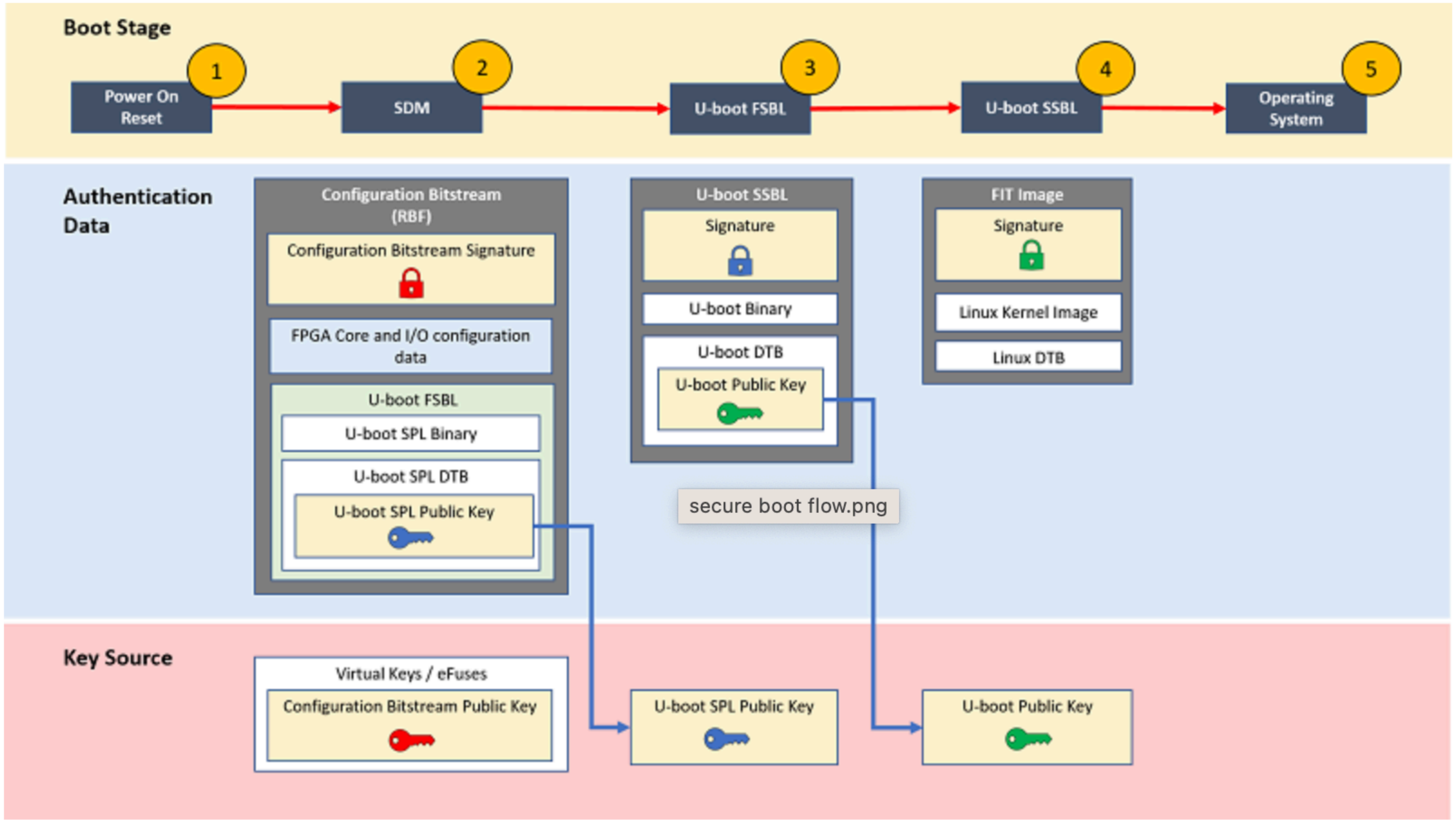
UEFI Secure Boot



Source (12.12.22): https://edk2-docs.gitbook.io/understanding-the-uefi-secure-boot-chain/secure_boot_chain_in_uefi/uefi_secure_boot

Intel Stratix 10 SoC Secure Boot Design Example

Overview

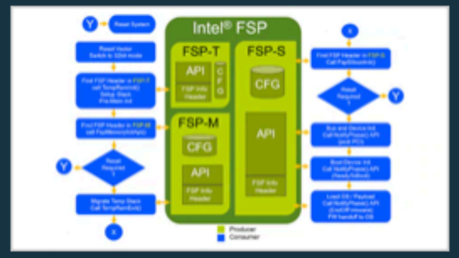


Source (20.12.22): https://www.rocketboards.org/foswiki/Documentation/S10SecureBootDemoDesign

Firmware Development Resources

Firmware development from edge to cloud and across all layers

Learn and Develop



Intel® Firmware Support Package

Implement this redistributable binary package for bootloaders to initialize Intel® silicon.

Unified Extensible Firmware Interface (UEFI)

Simplify platform initialization and firmware bootstrap operations with this industry-standard interface.

Intel® Slim Bootloader

This flexible, lightweight, open source reference bootloader supports verified boot, measured boot, and secure firmware updates.

coreboot*

Initialize system hardware with this extended open source firmware platform (GNU* General Public License version 2.0).

Projects and Toolkits

Minimum Platform Architecture: Open Source UEFI Firmware for Intel®-Based Platforms

Quickly port UEFI firmware for Intel® platforms using a scalable open source framework for platform initialization.

Simics® Simulator

Accelerate your software development with Simics® virtual platform technology. Download and install a preview of the public release of the Simics® simulator.

Source (15.12.22): https://www.intel.com/content/www/us/en/developer/topic-technology/firmware/overview.html

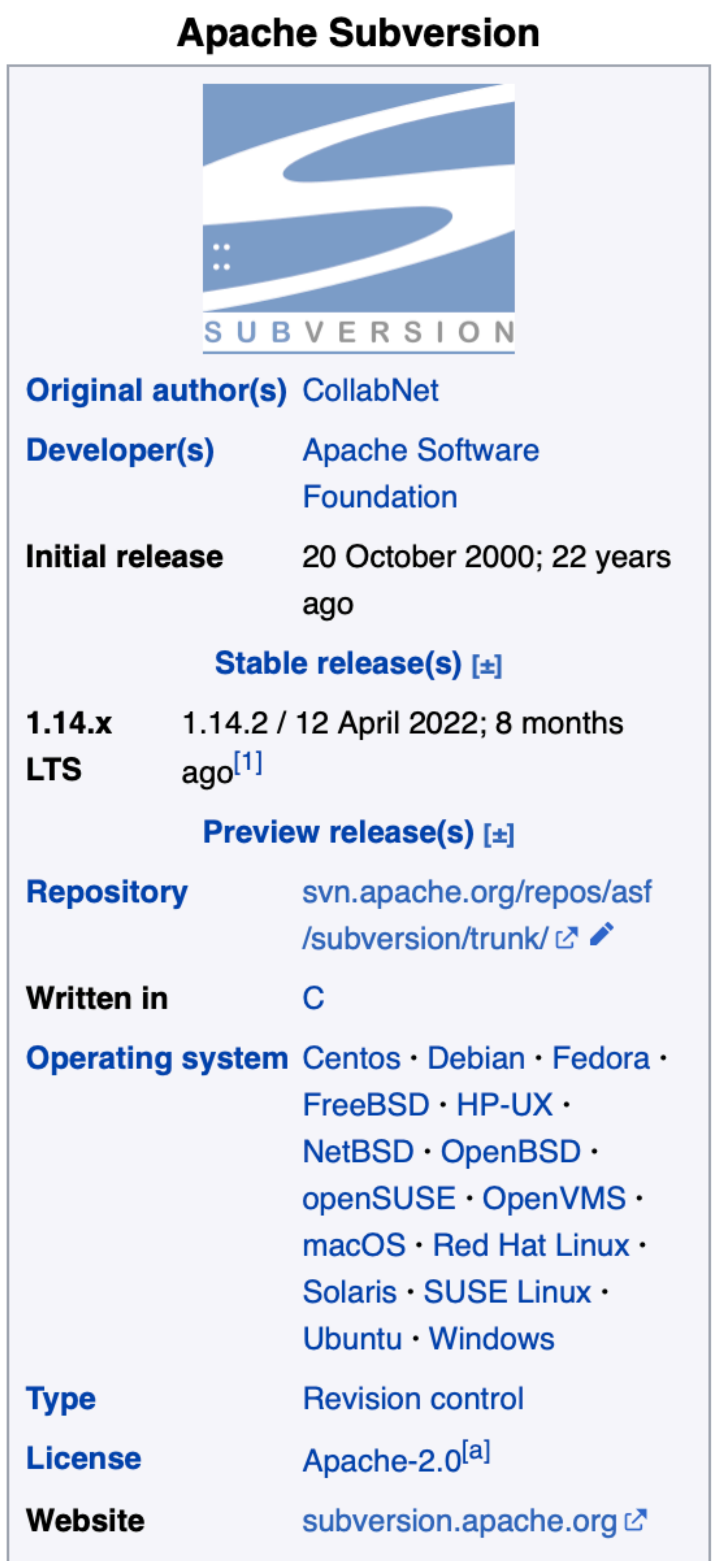
SVN

SVN significance in project work

Apache Subversion (often abbreviated **SVN**, after its command name *svn*) is a software versioning and revision control system distributed as open source under the Apache License.[2] Software developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation. Its goal is to be a mostly compatible successor to the widely used Concurrent Versions System (CVS).

The open source community has used Subversion widely: for example, in projects such as Apache Software Foundation, Free Pascal, FreeBSD, SourceForge, and from 2006 to 2019, GCC. CodePlex was previously a common host for Subversion repositories.

Subversion was created by CollabNet Inc. in 2000, and is now a top-level Apache project being built and used by a global community of contributors.

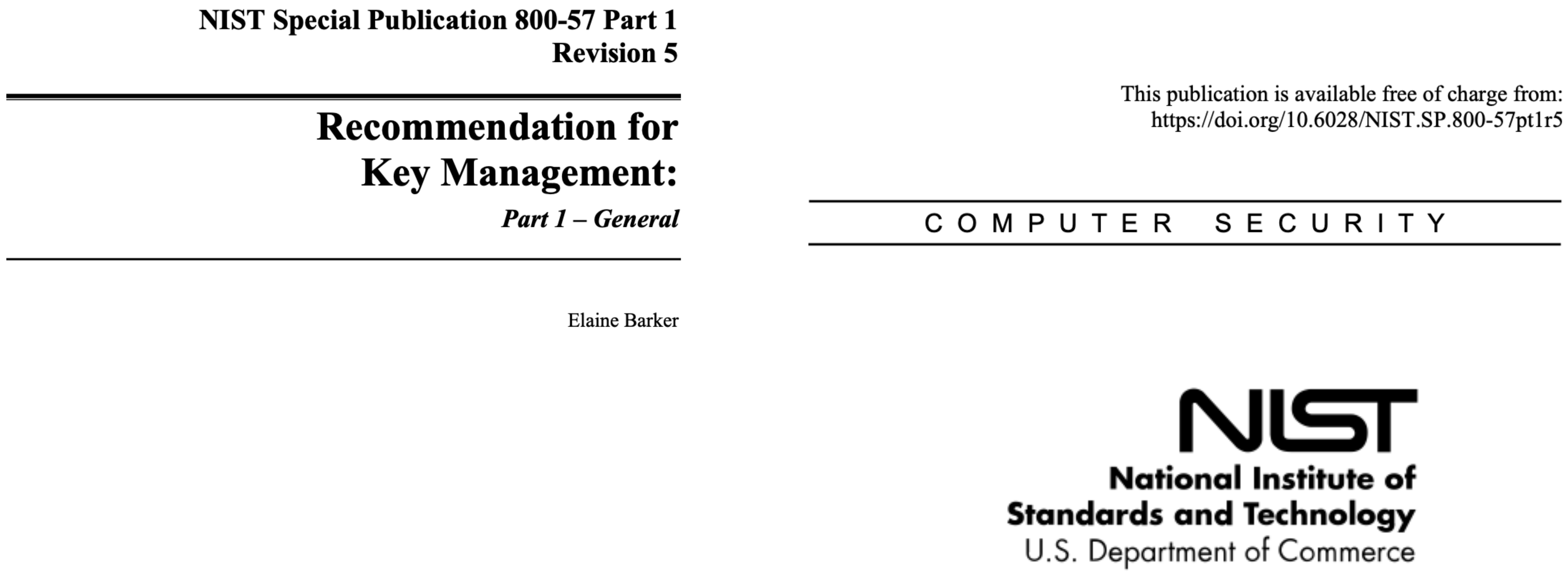


Apache Subversion

Original author(s)	CollabNet
Developer(s)	Apache Software Foundation
Initial release	20 October 2000; 22 years ago
Stable release(s) [±]	
1.14.x LTS	1.14.2 / 12 April 2022; 8 months ago[1]
Preview release(s) [±]	
Repository	svn.apache.org/repos/asf/subversion/trunk/
Written in	C
Operating system	Centos · Debian · Fedora · FreeBSD · HP-UX · NetBSD · OpenBSD · openSUSE · OpenVMS · macOS · Red Hat Linux · Solaris · SUSE Linux · Ubuntu · Windows
Type	Revision control
License	Apache-2.0[a]
Website	subversion.apache.org

Key revocation

NIST Special Publication



NIST Special Publication 800-57 Part 1
Revision 5

Recommendation for Key Management:

Part 1 – General

Elaine Barker

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.800-57pt1r5

C O M P U T E R   S E C U R I T Y

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Key revocation

What for?

A possible function in the lifecycle of a cryptographic key; a process whereby a notice is made available to affected entities that the key should be removed from operational use prior to the end of the established cryptoperiod of that key.

Key revocation is used in cases where :

1) the authorized use of a key needs to be terminated prior to the end of the established cryptoperiod of that key,

2) a key whose usage period has expired has been compromised. A key may be revoked for administrative reasons (e.g., the key's owner has left the organization, or a device containing the key has been removed from service), or it may be revoked on an emergency basis if there is reason to believe that it may have been disclosed to or otherwise accessed by an unauthorized entity.

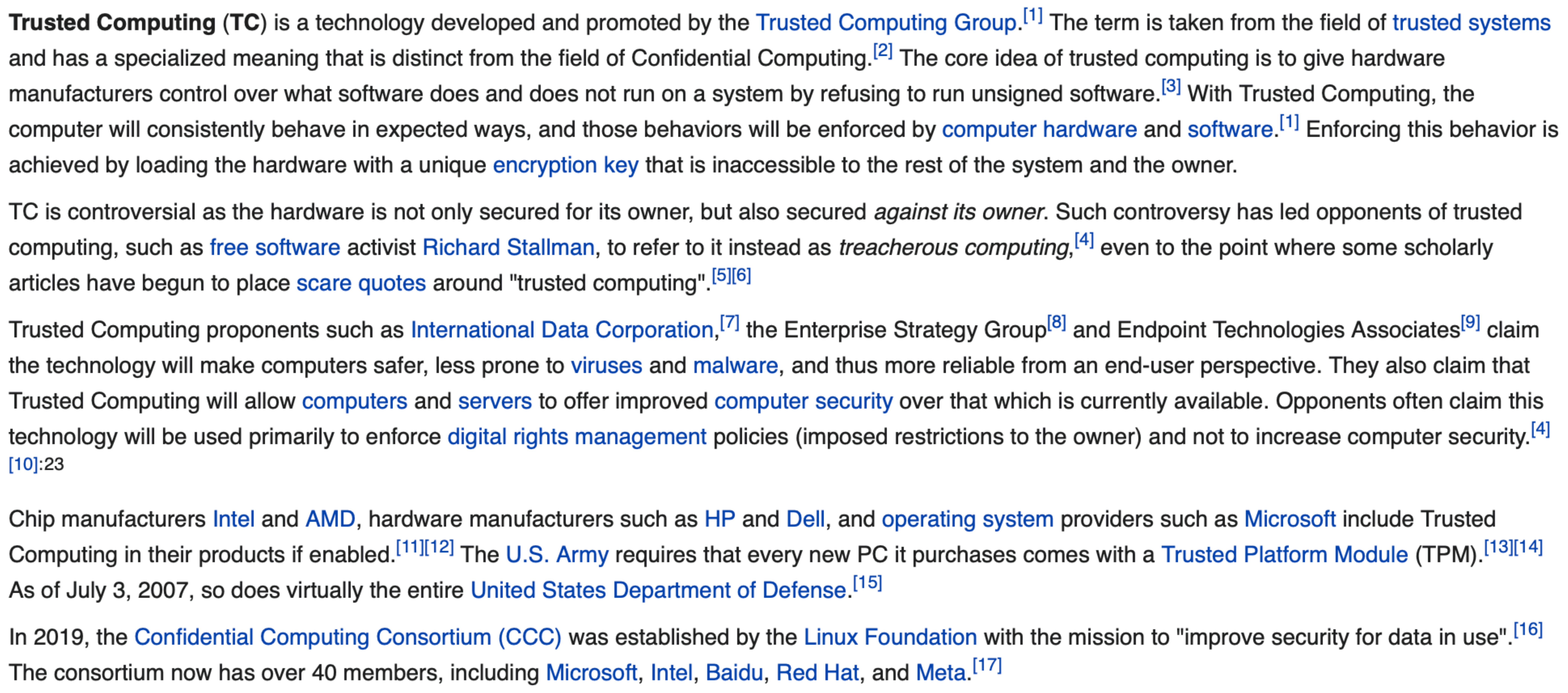
In either case, a cryptographic key **should** be revoked as soon as feasible after the need for revocation has been determined.

Additional external material for teaching purposes during the lecture:

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf

Trusted computing

What does wikipedia have to say about it?



Trusted Computing (**TC**) is a technology developed and promoted by the Trusted Computing Group.[1] The term is taken from the field of trusted systems and has a specialized meaning that is distinct from the field of Confidential Computing.[2] The core idea of trusted computing is to give hardware manufacturers control over what software does and does not run on a system by refusing to run unsigned software.[3] With Trusted Computing, the computer will consistently behave in expected ways, and those behaviors will be enforced by computer hardware and software.[1] Enforcing this behavior is achieved by loading the hardware with a unique encryption key that is inaccessible to the rest of the system and the owner.

TC is controversial as the hardware is not only secured for its owner, but also secured *against its owner*. Such controversy has led opponents of trusted computing, such as free software activist Richard Stallman, to refer to it instead as *treacherous computing*,[4] even to the point where some scholarly articles have begun to place scare quotes around "trusted computing".[5][6]

Trusted Computing proponents such as International Data Corporation,[7] the Enterprise Strategy Group[8] and Endpoint Technologies Associates[9] claim the technology will make computers safer, less prone to viruses and malware, and thus more reliable from an end-user perspective. They also claim that Trusted Computing will allow computers and servers to offer improved computer security over that which is currently available. Opponents often claim this technology will be used primarily to enforce digital rights management policies (imposed restrictions to the owner) and not to increase computer security.[4][10]:23

Chip manufacturers Intel and AMD, hardware manufacturers such as HP and Dell, and operating system providers such as Microsoft include Trusted Computing in their products if enabled.[11][12] The U.S. Army requires that every new PC it purchases comes with a Trusted Platform Module (TPM).[13][14] As of July 3, 2007, so does virtually the entire United States Department of Defense.[15]

In 2019, the Confidential Computing Consortium (CCC) was established by the Linux Foundation with the mission to "improve security for data in use".[16] The consortium now has over 40 members, including Microsoft, Intel, Baidu, Red Hat, and Meta.[17]

Device Attestation

Remote Attestation

This advanced feature allows a hardware entity or a combination of hardware and software to gain a remote provider's (also known as the relying party) or producer's trust.

Remote attestation gives the relying party increased confidence that the software is running:

+Inside an Intel® Software Guard Extension (Intel® SGX) enclave

+On a fully updated system at the latest security level (also referred to as the trusted computing base [TCB] version)

Attestation results provide:

+The identity of the software being attested

+Details of an unmeasured state (such as the execution mode)

+An assessment of possible software tampering

Intel® TDT

Overview

Advanced hardware-accelerated threat detection of ransomware and cryptomining attacks.

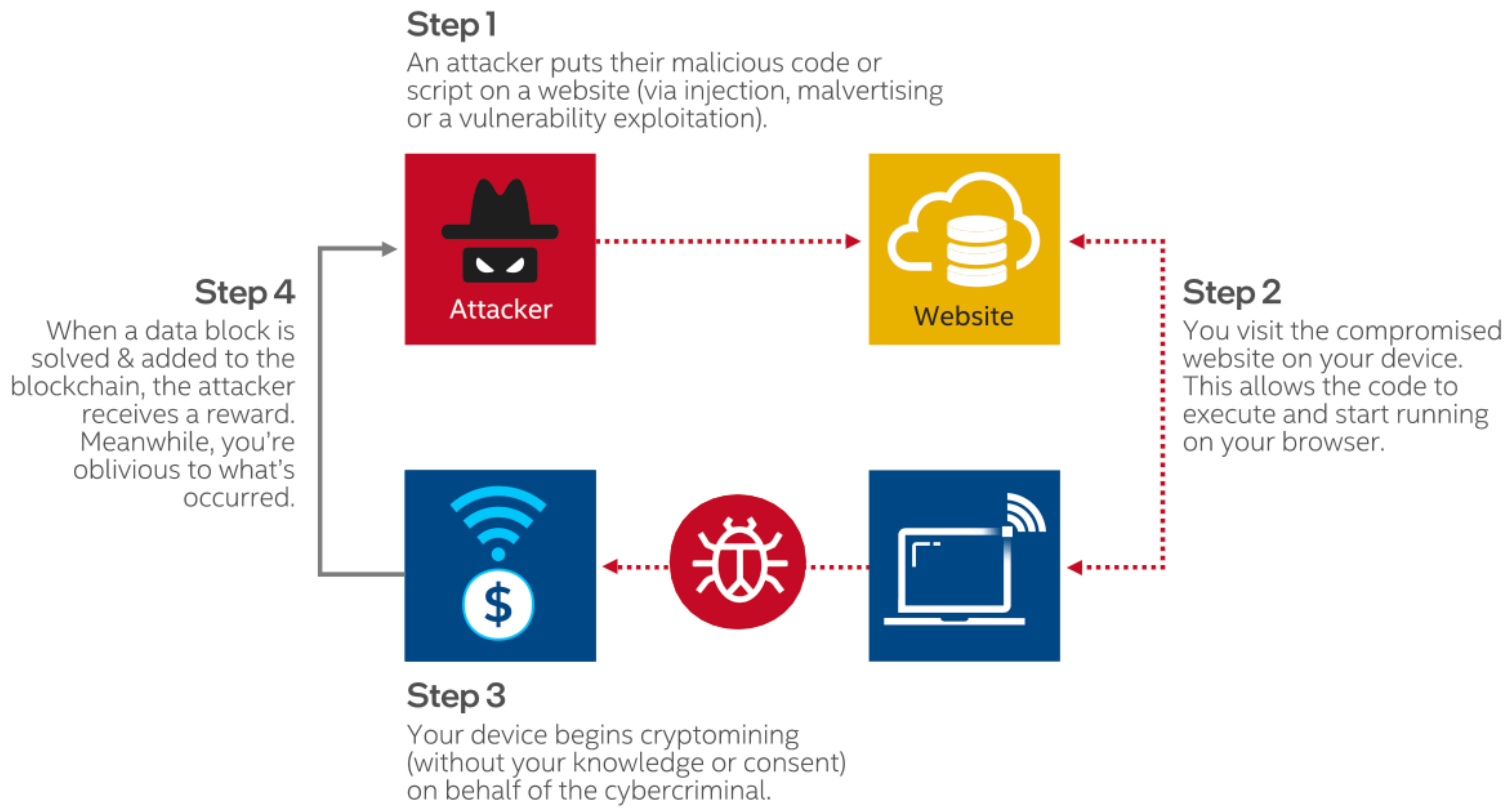
Integrated into vendors’ software to improve security efficacy and performance.

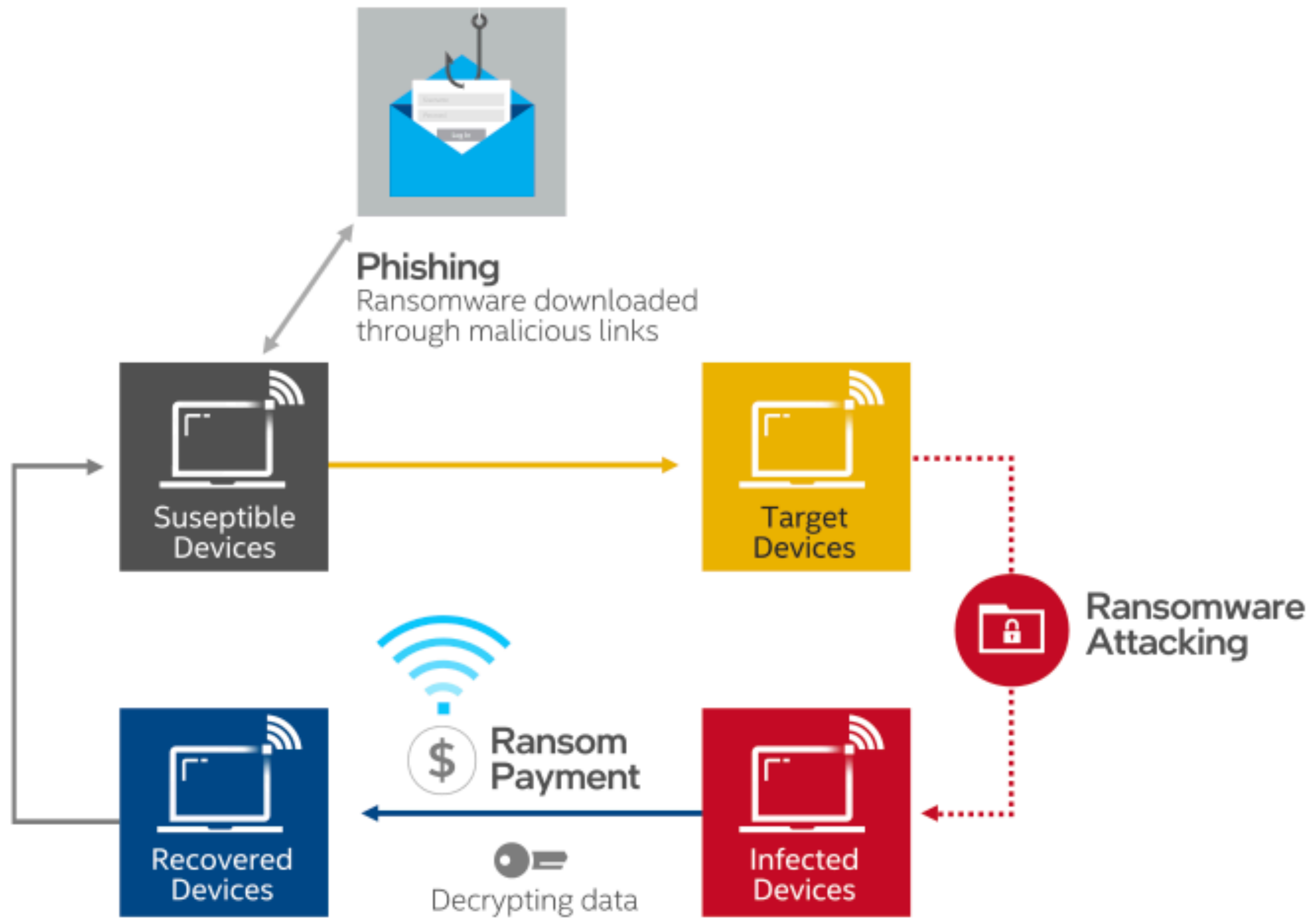
Increases threat detection efficacy on Intel® vPro platform PCs.

CPU telemetry and optimized driver technology help identify hundreds of events (with minimal CPU impact), while ML algorithms reduce false positives and increase detection efficacy.

Types of Attacks Intel® TDT Helps Defend Against

Malicious cryptomining, Ransomware

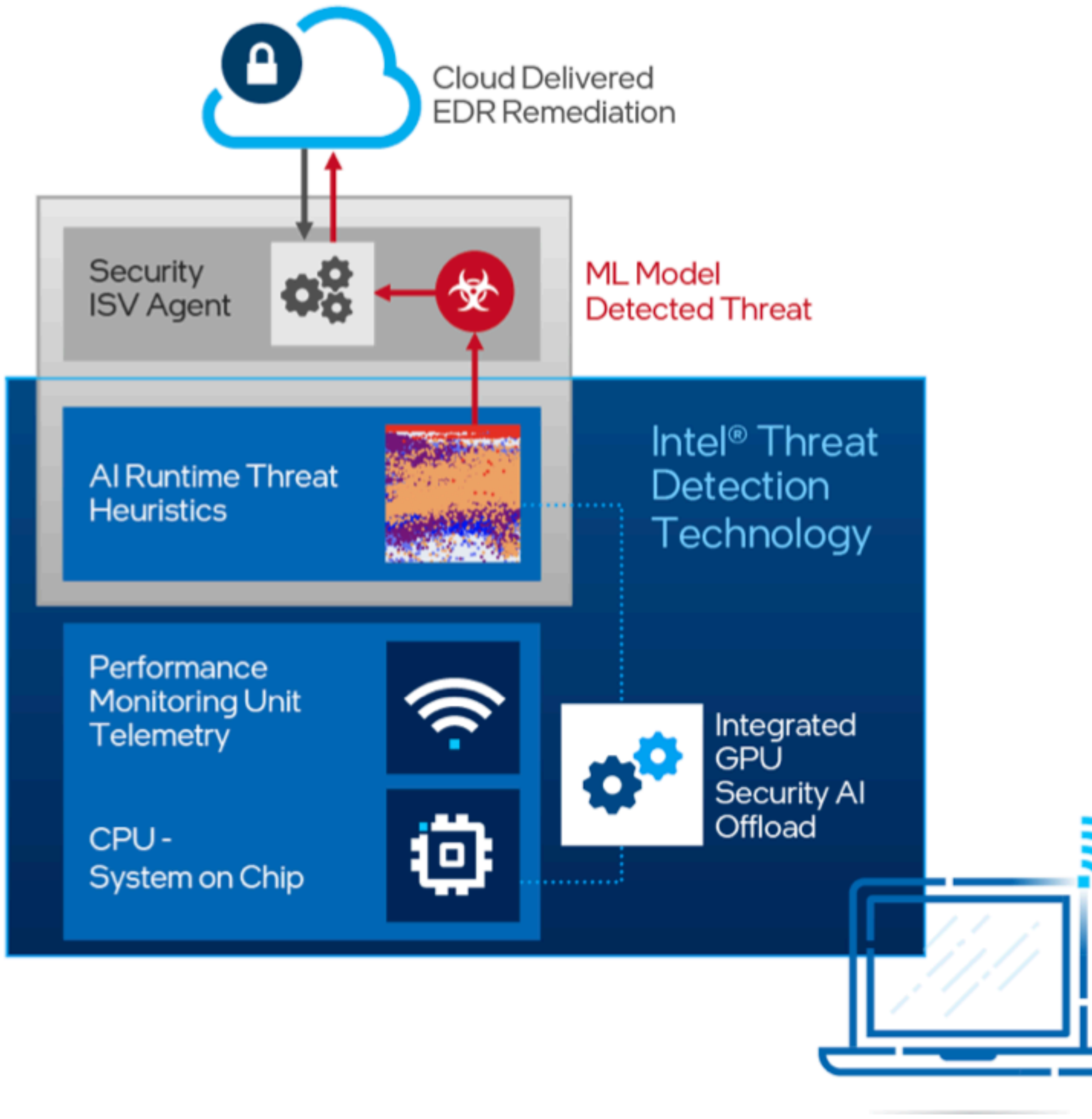

Step 1
An attacker puts their malicious code or script on a website (via injection, malvertising or a vulnerability exploitation).
Attacker
Website
Step 2
You visit the compromised website on your device. This allows the code to execute and start running on your browser.
Step 3
Your device begins cryptomining (without your knowledge or consent) on behalf of the cybercriminal.
Step 4
When a data block is solved & added to the blockchain, the attacker receives a reward. Meanwhile, you're oblivious to what's occurred.



How Intel® TDT Works

High level

Intel TDT, a feature of Intel® Hardware Shield, uses GPU offload and CPU telemetry to accelerate and enable advanced threat defenses such as AMS and real-time malware detection.


Cloud Delivered EDR Remediation
Security ISV Agent
ML Model Detected Threat
AI Runtime Threat Heuristics
Intel® Threat Detection Technology
Performance Monitoring Unit Telemetry
CPU - System on Chip
Integrated GPU Security AI Offload

Intel® TDT

Additional external material for teaching purposes during the lecture:

https://www.intel.com/content/www/us/en/content-details/754494/advanced-threat-protections-white-paper-intel-threat-detection-technology-amp-intel-control-flow-enforcement-technology.html?wapkw=Threat%20Detection%20Technology%20

DEMO: https://www.intel.co.uk/content/www/uk/en/architecture-and-technology/demos/hardware-threat-detection.html

SGX, TDX

Intel® Software Guard Extensions (Intel® SGX)

Intel® Software Guard Extensions (Intel® SGX) is a unique technology that helps protect data while it’s actively in use.

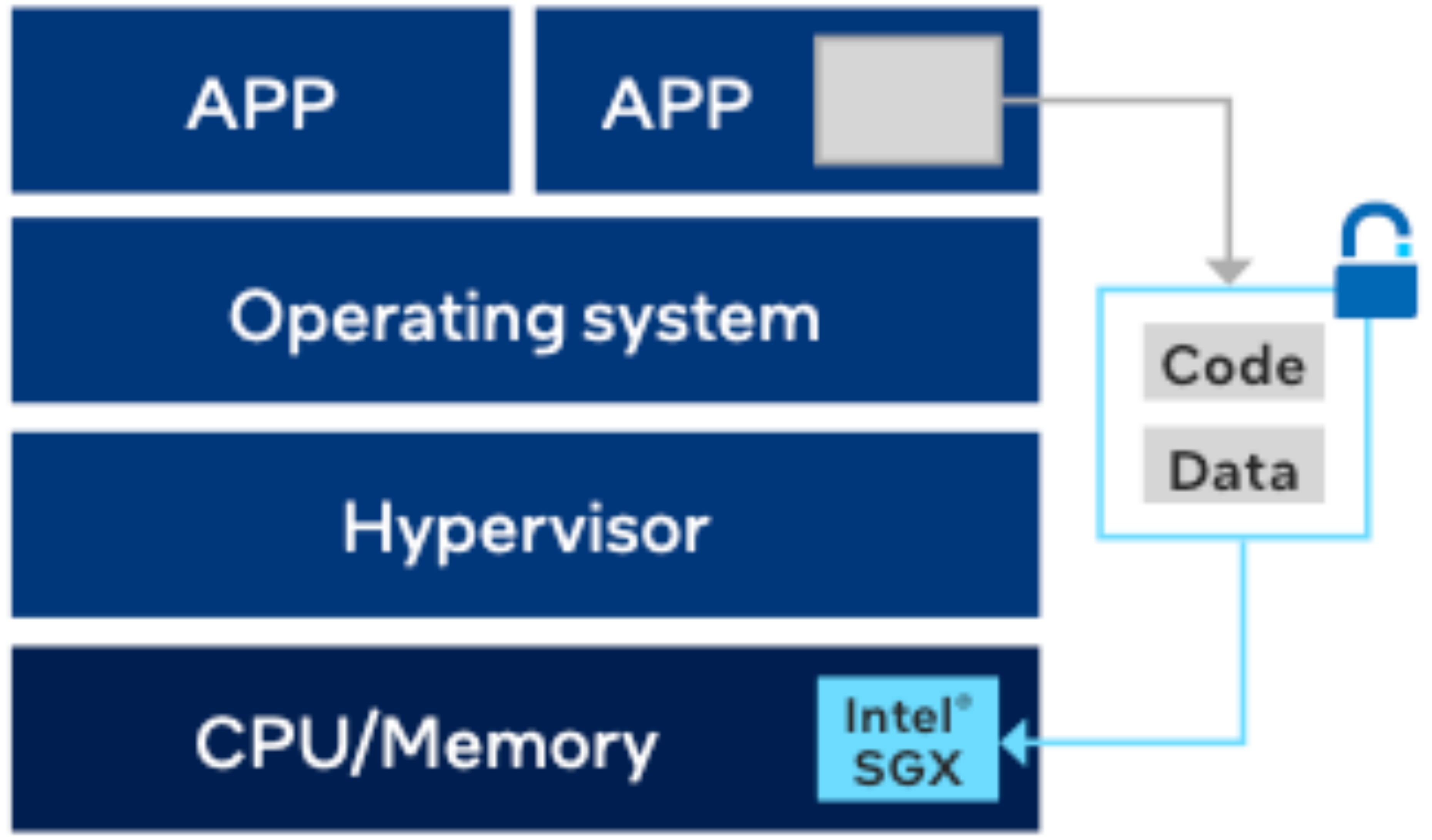
Rather than excluding sensitive data from analytics or AI models, businesses using Intel Xeon Scalable processors can create protected data enclaves with Intel SGX.

These isolated environments can enable businesses to extract value from their most sensitive data while helping to keep it confidential.

Intel® Software Guard Extensions (Intel® SGX)

Intel® SGX allows user-level code to allocate private regions of memory, called enclaves, which are designed to be protected from processes running at higher privilege levels. Only Intel® SGX offers such a granular level of control and protection.

Intel® Software Guard Extensions (Intel® SGX) helps protect data in use via unique application isolation technology. Protect selected code and data from modification using hardened enclaves with Intel SGX. Play Video. Play. Mute.



Intel® Software Guard Extensions (Intel® SGX)

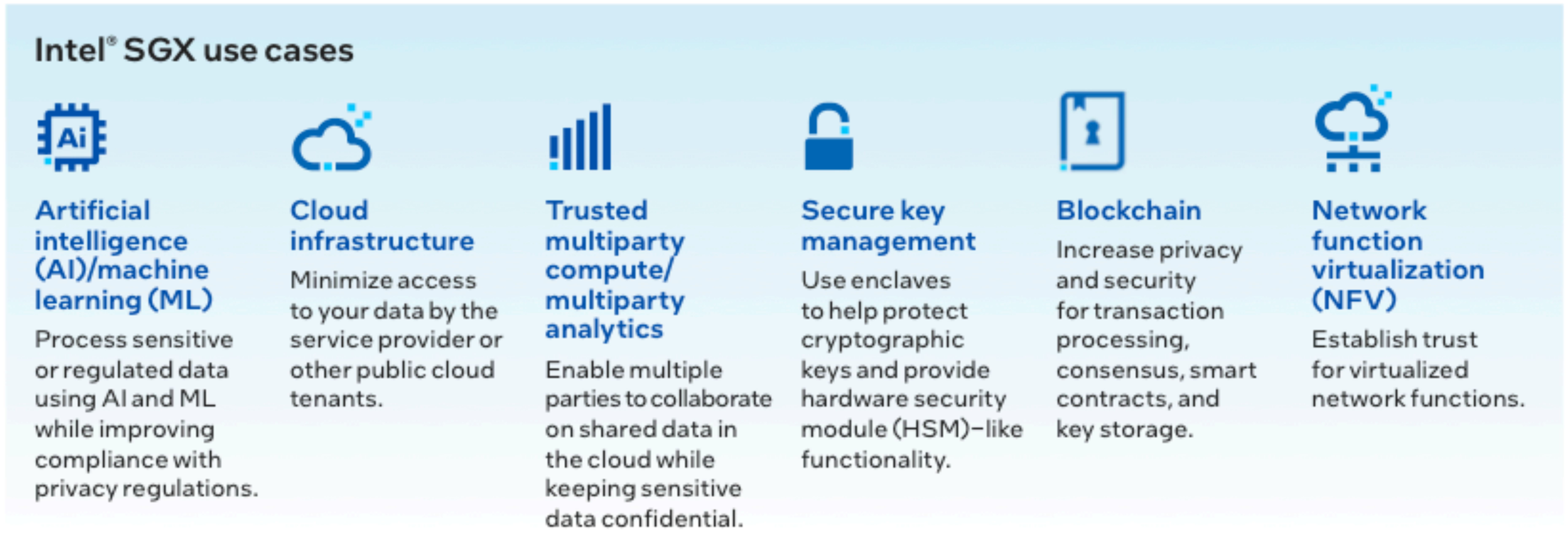
Confidential computing using Intel Xeon Scalable processors with Intel SGX helps close this gap by better protecting data in use. Intel SGX is the most deployed, researched and battle-tested confidential computing technology for data centers on the market today.

Intel SGX allows you to create encrypted enclaves within the CPU and memory that help protect data from access by anything other than authorized, verified application code. The hypervisor, OS, and even administrators with root access can’t see the data inside the Intel SGX enclave.

Confidential computing allows the extraction of insights or training of models from sensitive data without exposing it to other software, collaborators, or your cloud provider. This opens wide possibilities for businesses to harness data that was previously too sensitive or regulated to activate for analytics and other purposes.

Intel® Software Guard Extensions (Intel® SGX)

Use case



Intel® SGX use cases

Artificial intelligence (AI)/machine learning (ML)
Process sensitive or regulated data using AI and ML while improving compliance with privacy regulations.

Cloud infrastructure
Minimize access to your data by the service provider or other public cloud tenants.

Trusted multiparty compute/ multiparty analytics
Enable multiple parties to collaborate on shared data in the cloud while keeping sensitive data confidential.

Secure key management
Use enclaves to help protect cryptographic keys and provide hardware security module (HSM)–like functionality.

Blockchain
Increase privacy and security for transaction processing, consensus, smart contracts, and key storage.

Network function virtualization (NFV)
Establish trust for virtualized network functions.

Intel® Trust Domain Extensions (Intel® TDX)

Intel® Trust Domain Extensions (Intel® TDX) is introducing new, architectural elements to help deploy hardware-isolated, virtual machines (VMs) called trust domains (TDs). Intel TDX is designed to isolate VMs from the virtual-machine manager (VMM)/hypervisor and any other non-TD software on the platform to protect TDs from a broad range of software. These hardware-isolated TDs include:

1. Secure-Arbitration Mode (SEAM) – a new mode of the CPU designed to host an Intel-provided, digitally-signed, security-services module called the Intel TDX module.

2. Shared bit in GPA to help allow TD to access shared memory.

3. Secure EPT to help translate private GPA to provide address-translation integrity and to prevent TD-code fetches from shared memory. Encryption and integrity protection of private-memory access using a TD-private key is the goal.

4. Physical-address-metadata table (PAMT) to help track page allocation, page initialization, and TLB consistency.

5. Multi-key, total-memory-encryption (MKTME) engine designed to provide memory encryption using AES-128- XTS and integrity using 28-bit MAC and a TD-ownership bit.

6. Remote attestation designed to provide evidence of TD executing on a genuine, Intel TDX system and its TCB version.

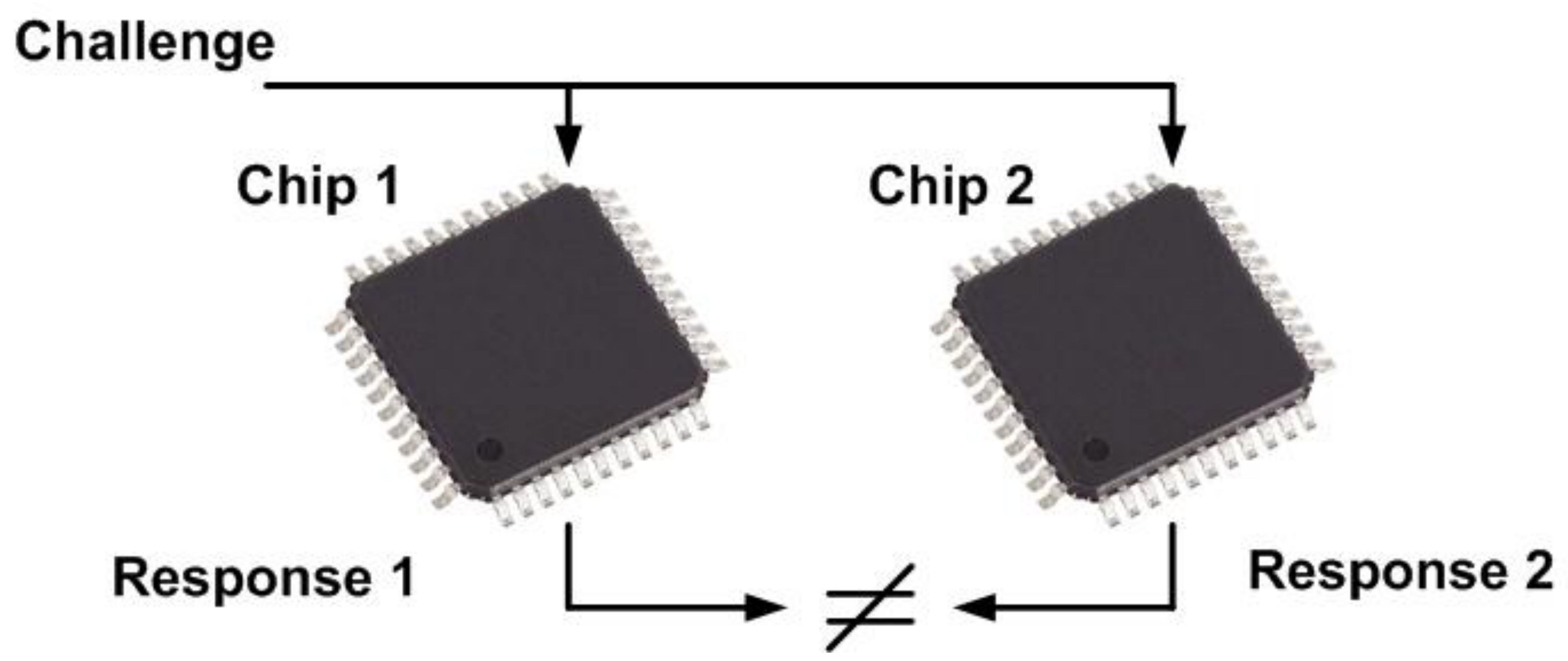
Intel® Trust Domain Extensions (Intel® TDX)

What for?

Intel TDX is designed to isolate VMs from the virtual-machine manager (VMM)/hypervisor and any other non-TD software on the platform to protect TDs from a broad range of software.

Physical Unclonable Function (PUF)

General Idea



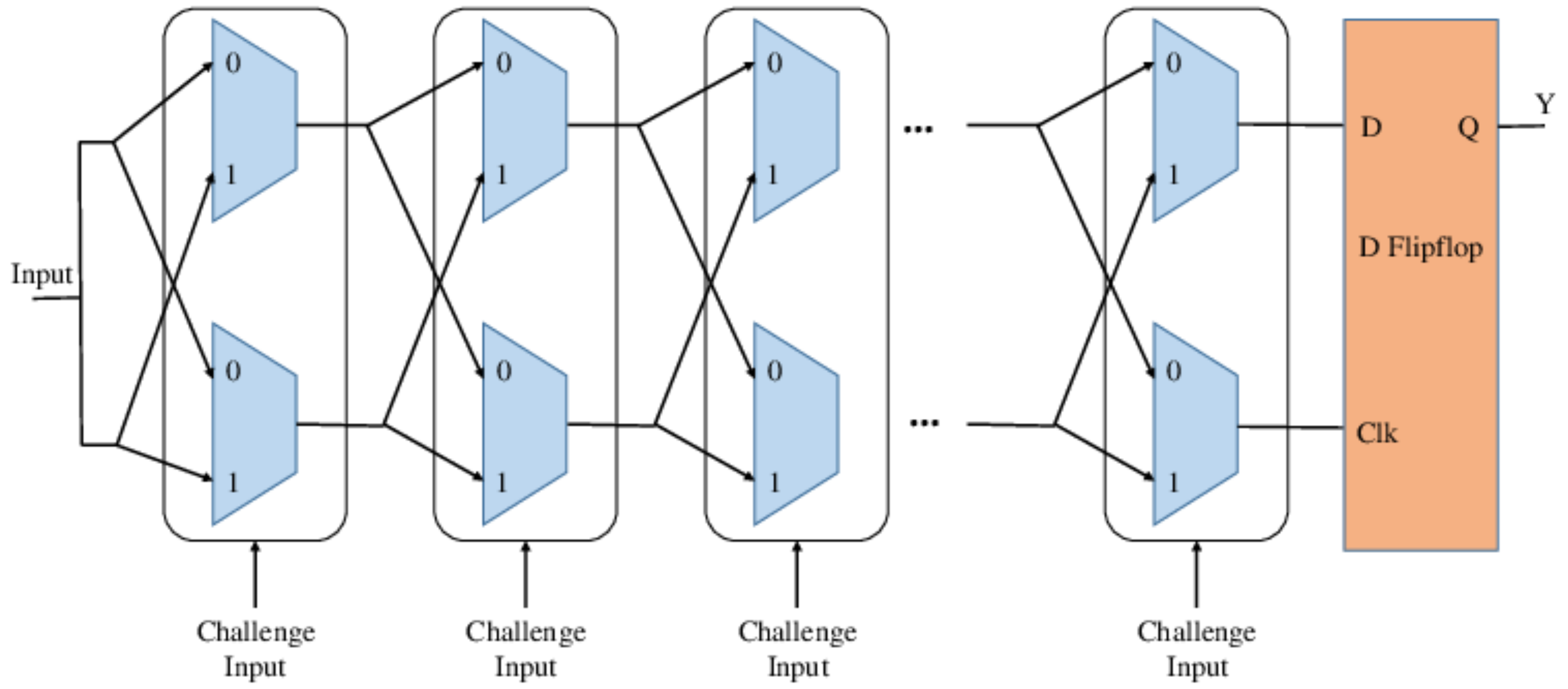
<u>Question to discuss with students at lecture:</u>

What is the application of PUFs?

Source(15.11.22): https://cryptography.gmu.edu/research/pufs.php

How does PUF work?

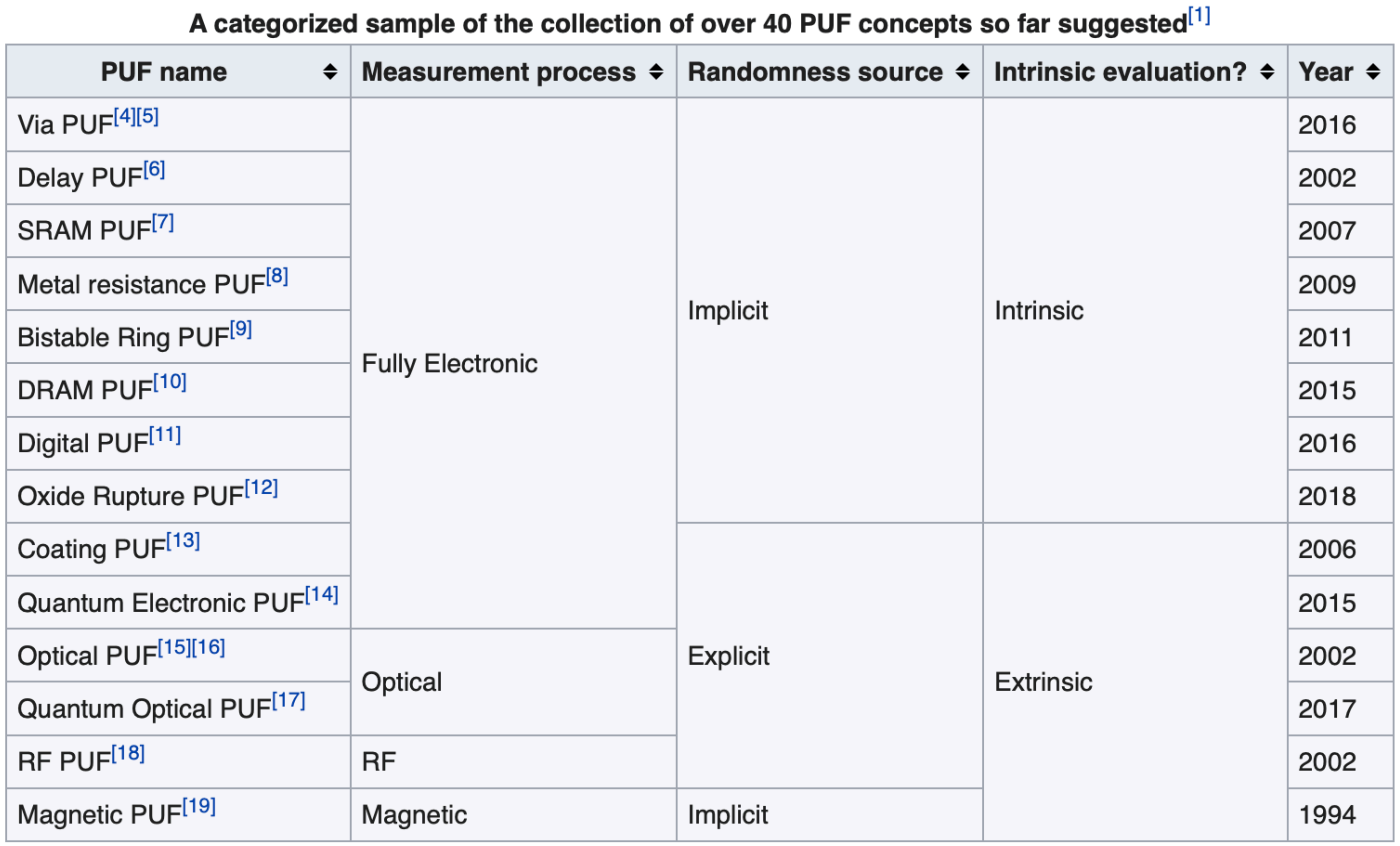
Selected architecture - ARBITER



Source: Yanambaka, Venkata P., Saraju P. Mohanty, and Elias Kougianos. "Making use of manufacturing process variations: A dopingless transistor based-PUF for hardware-assisted security." *IEEE Transactions on Semiconductor Manufacturing* 31.2 (2018): 285-294.

PUF types

Implementation issues



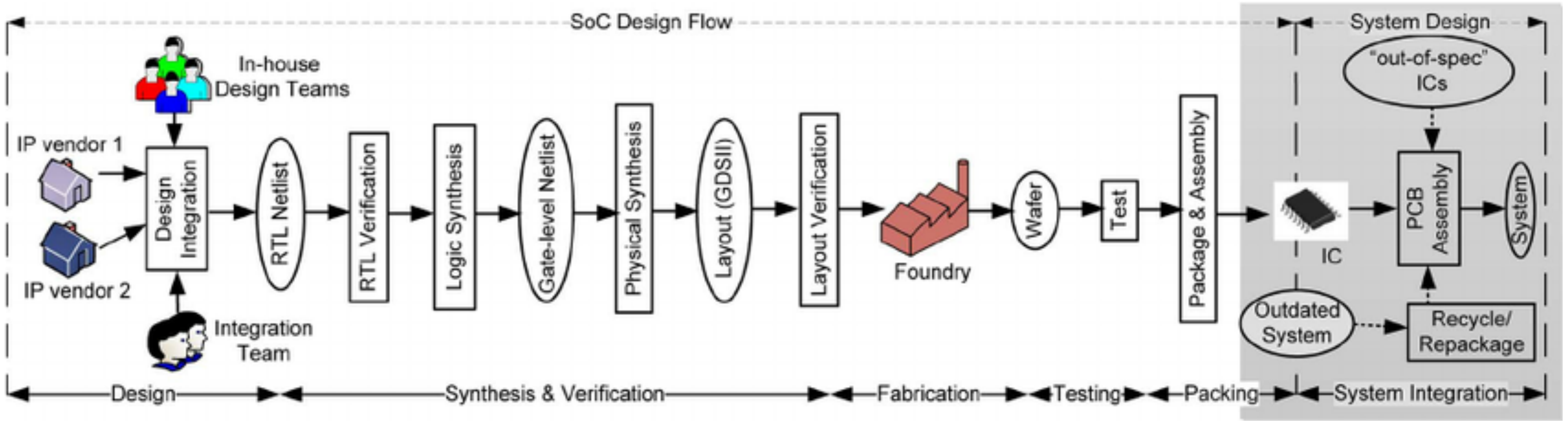
A categorized sample of the collection of over 40 PUF concepts so far suggested[1]

PUF name	Measurement process	Randomness source	Intrinsic evaluation?	Year
Via PUF[4][5]	Fully Electronic	Implicit	Intrinsic	2016
Delay PUF[6]	Fully Electronic	Implicit	Intrinsic	2002
SRAM PUF[7]	Fully Electronic	Implicit	Intrinsic	2007
Metal resistance PUF[8]	Fully Electronic	Implicit	Intrinsic	2009
Bistable Ring PUF[9]	Fully Electronic	Implicit	Intrinsic	2011
DRAM PUF[10]	Fully Electronic	Implicit	Intrinsic	2015
Digital PUF[11]	Fully Electronic	Implicit	Intrinsic	2016
Oxide Rupture PUF[12]	Fully Electronic	Implicit	Intrinsic	2018
Coating PUF[13]	Fully Electronic	Explicit	Extrinsic	2006
Quantum Electronic PUF[14]	Fully Electronic	Explicit	Extrinsic	2015
Optical PUF[15][16]	Optical	Explicit	Extrinsic	2002
Quantum Optical PUF[17]	Optical	Explicit	Extrinsic	2017
RF PUF[18]	RF	Explicit	Extrinsic	2002
Magnetic PUF[19]	Magnetic	Implicit	Extrinsic	1994

Source (11.11.22): https://en.wikipedia.org/wiki/Types_of_physical_unclonable_function

Supply chain in microelectronics

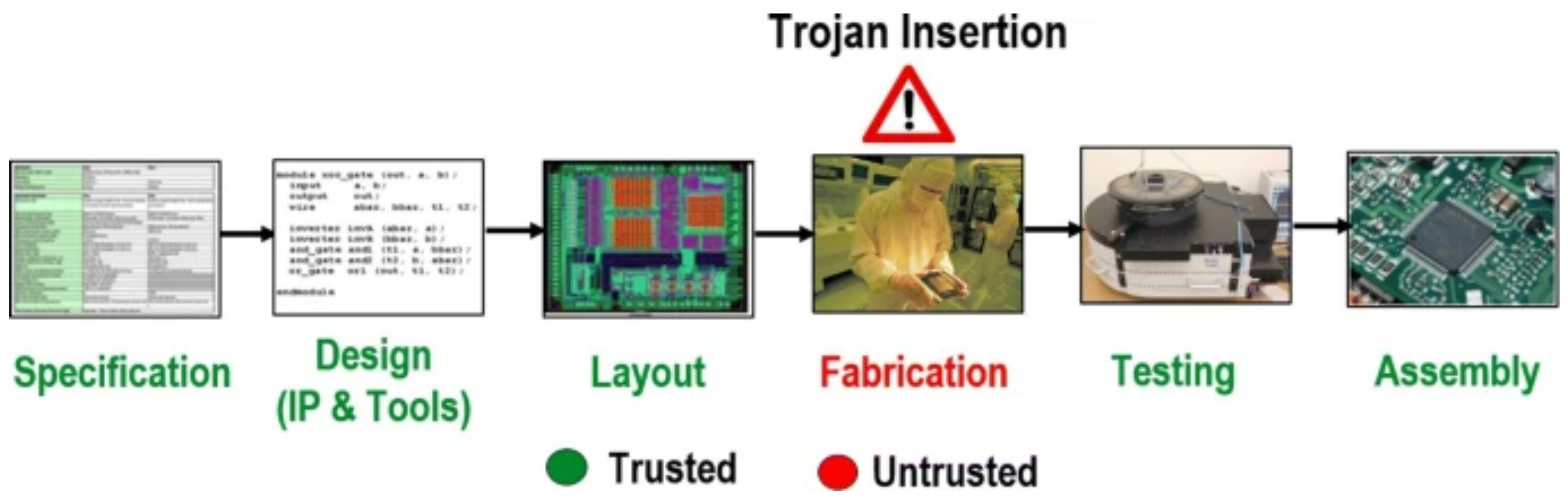
From idea to chip



Source: Rostami, Masoud, Farinaz Koushanfar, and Ramesh Karri. "A primer on hardware security: Models, methods, and metrics." *Proceedings of the IEEE* 102.8 (2014): 1283-1295.

Supply chain

Where to attack?

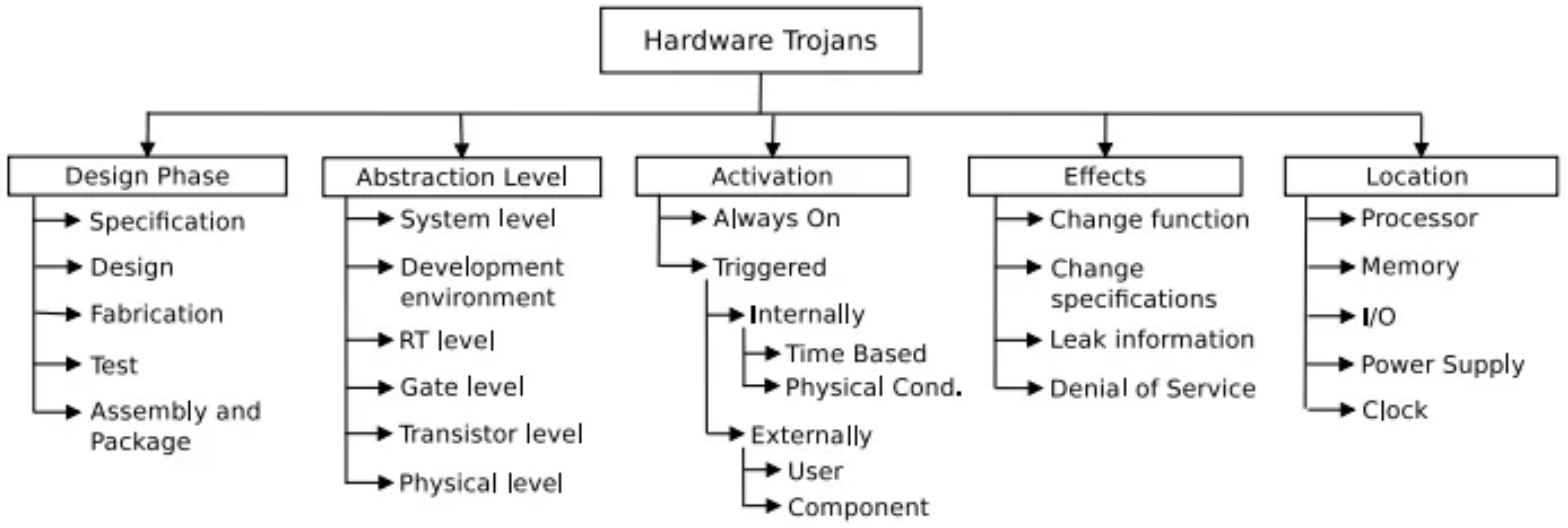


Source: Vashistha, N., Rahman, M.T., Shen, H. *et al.* Detecting Hardware Trojans Inserted by Untrusted Foundry Using Physical Inspection and Advanced Image Processing. *J Hardw Syst Secur* **2**, 333–344 (2018). https://doi.org/10.1007/s41635-018-0055-0

Or is it just fabrication?

Hardware Trojan

Main classification



Source (23.01.22): https://www.allaboutcircuits.com/news/most-significant-cyberattack-in-history-prompts-questions-supply-chain-security/

How can a hacker help us?

Ethical issues in cryptography

Ethical hacker is a security expert who penetrates a computer system, network, application or other computing resource on behalf of its owners and with their authorization.

Organizations are using ethical hackers to find potential security vulnerabilities that can be exploited by hackers.

Additional external material for teaching purposes during the lecture:

https://www.techtarget.com/searchsecurity/definition/ethical-hacker