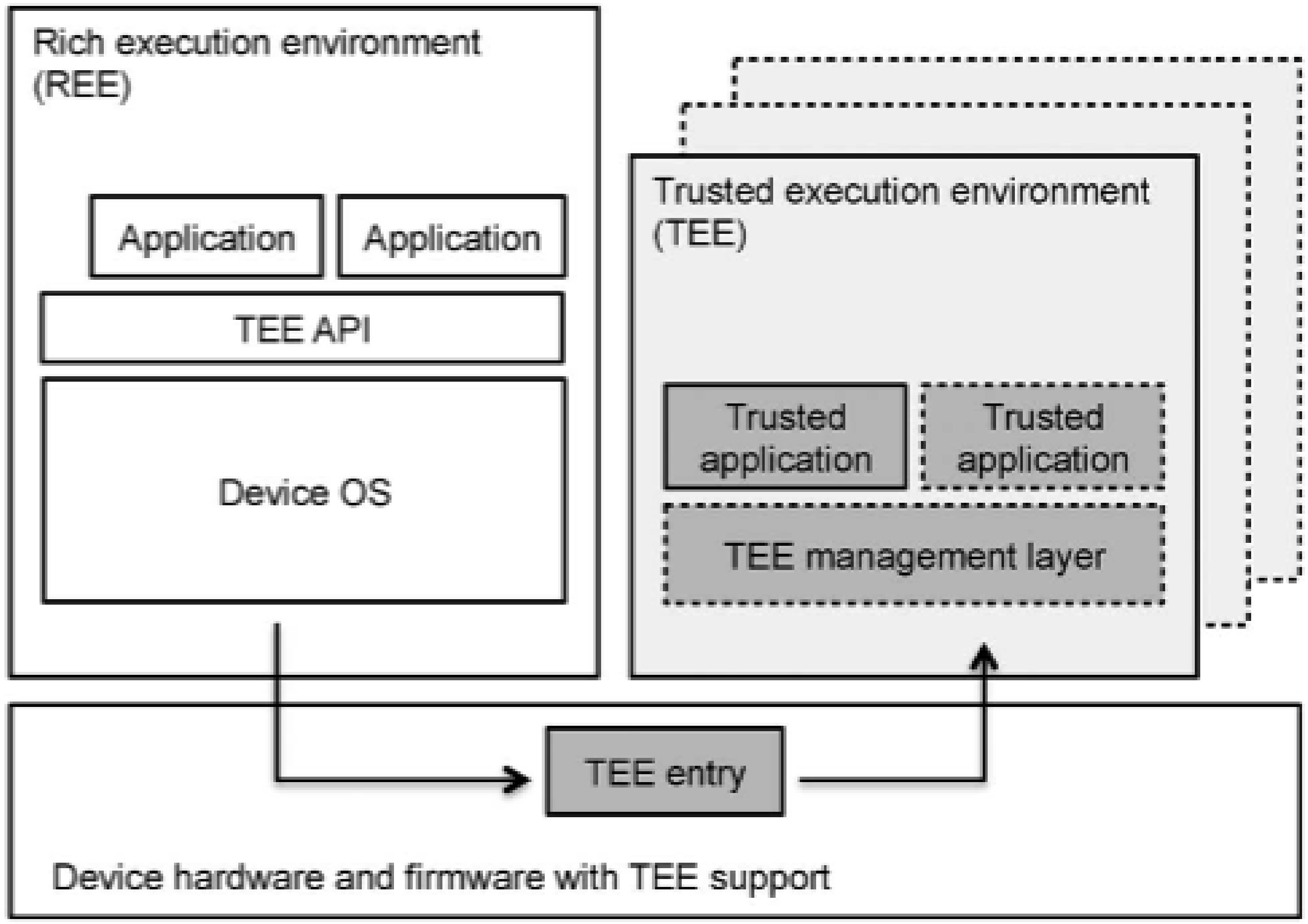
TRUSTED COMPUTING TECHNOLOGIES

TEE = Trusted Execution Environment

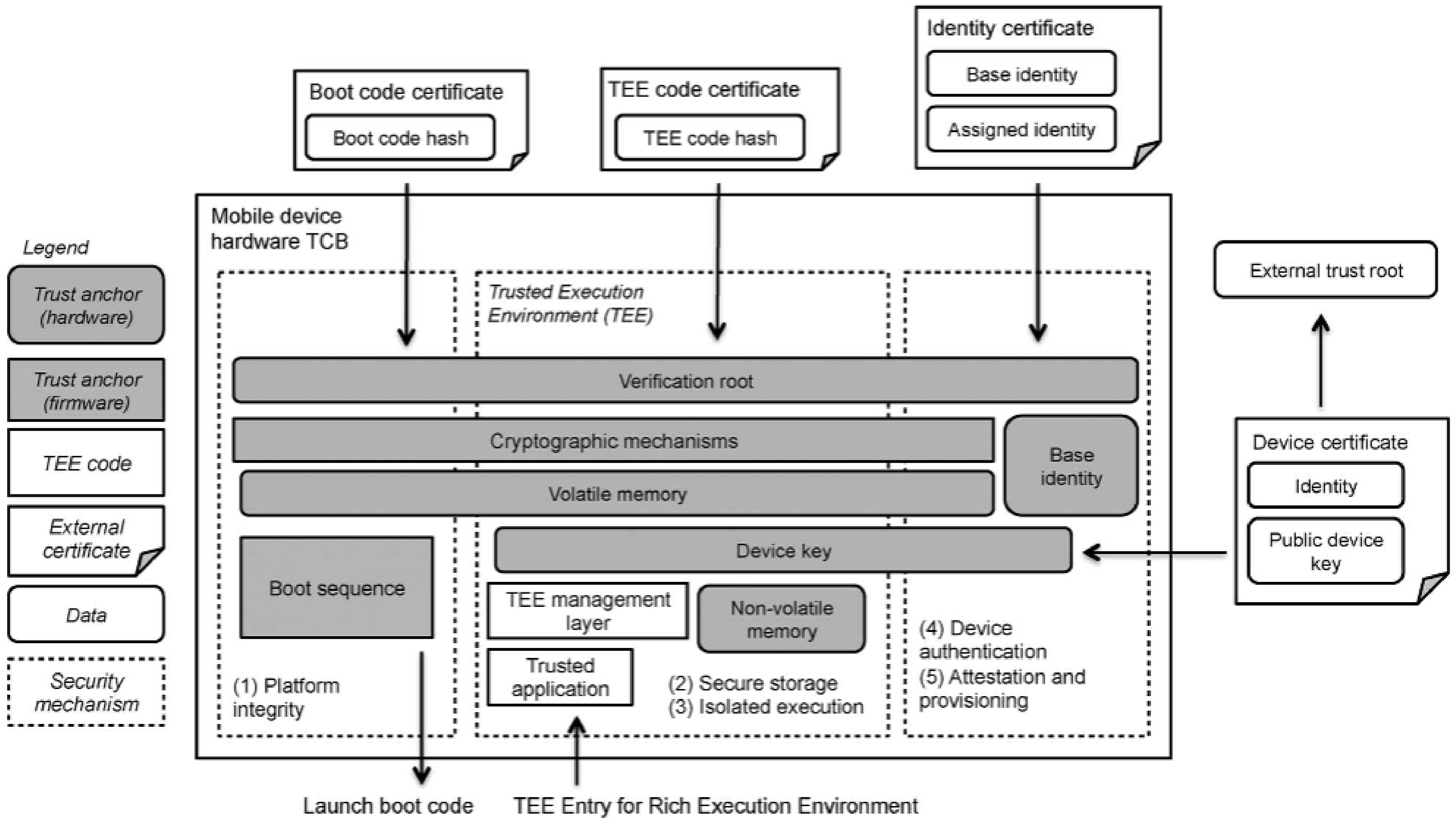
Mandatory reading

- “Innovative Instructions and Software Model for Isolated Execution”, Frank McKeen, Ilya Alexandrovich, Alex Berenzon, Carlos Rozas, Hisham Shafi, Vedvyas Shanbhogue and Uday Savagaonkar, Intel Corporation
- “Enhanced Privacy ID” (EPID), DrDobbs, E Brickell, 2009
- Mobile Trusted Computing, Ansokan et all

TEE Architecture – in a device


Rich execution environment (REE)
Application
Application
TEE API
Device OS
Trusted execution environment (TEE)
Trusted application
Trusted application
TEE management layer
TEE entry
Device hardware and firmware with TEE support

TEE in mobile device



Techniques to realize TEE

- Recall ARM TrustZone (Lect 6)

- Intel SGX

SGX - ENCLAVES

Software Guard eXtensions

Overview

- SGX in a new technology introduced in Intel chipsets
- SGX architecture includes 17 new instructions, new processor structures and a new mode of execution.

- These include loading an enclave into protected memory, access to resources via page table mappings, and scheduling the execution of enclave enabled application. Thus, system software still maintains control as to what resources an enclave can access.
- An application can be encapsulated by a single enclave or can be decomposed into smaller components, such that only security critical components are placed into an enclave.

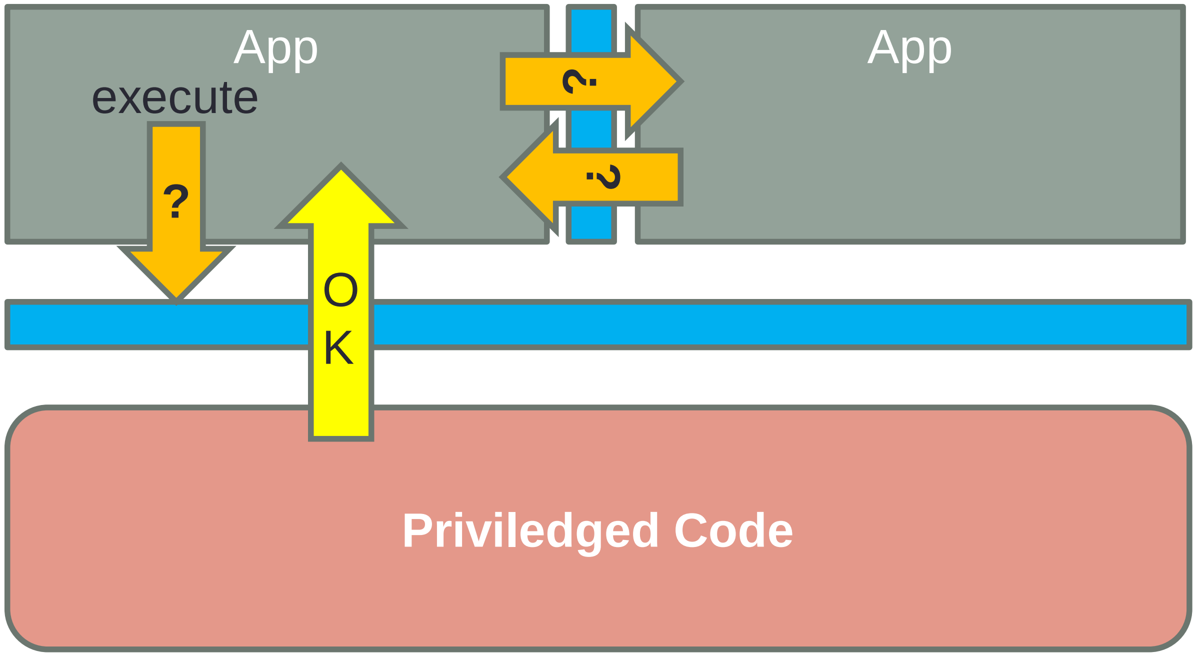
Enclaves

- Enclaves are isolated memory regions of code and data
- One part of physical memory (RAM) is reserved for enclaves and is called Enclave Page Cache (EPC)

- EPC memory is encrypted in the main memory (RAM)
- EPC is managed by OS/VMM

- Trusted hardware consists of the CPU Die only

Protected Mode (rings) protects



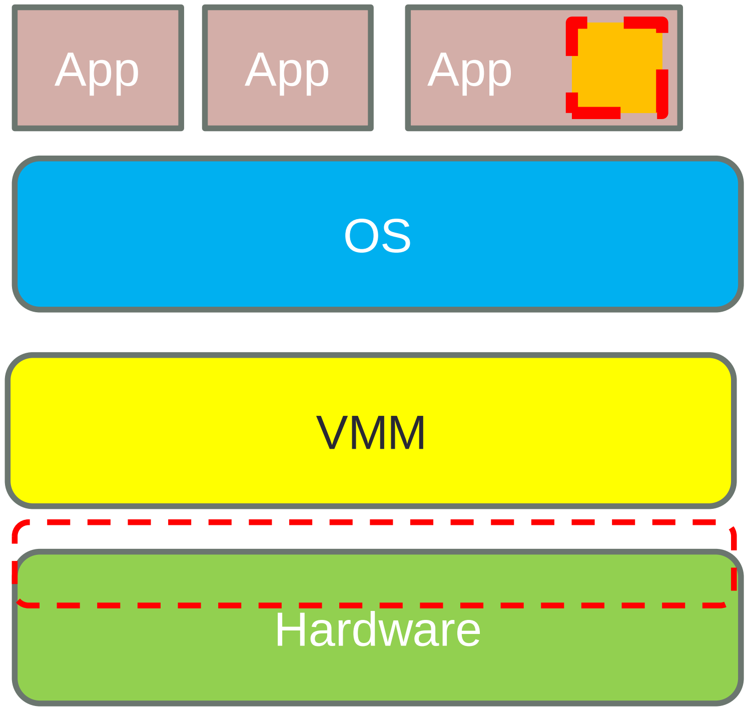
1 OS from App

2 And from other Apps

But Apps are not protect malware attacks on OS

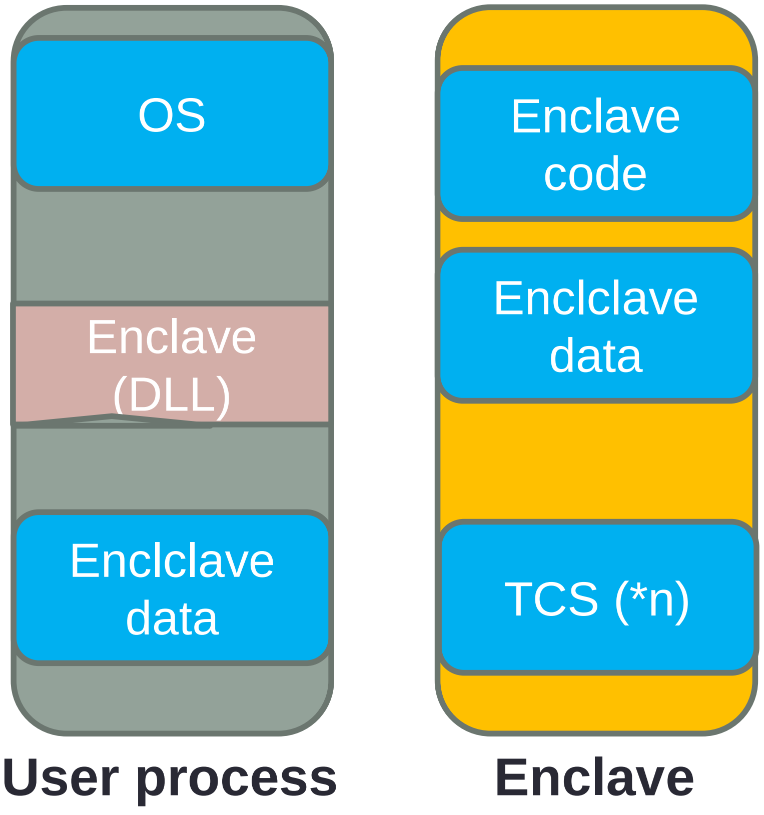
Reduced attack surface with SGX

- Application gains ability to defend is own secrets
  - Smaller attack suface (App+processor)
  - Malware that subverts OS or VMM, BIOS, Drivers cannot steal app secrets



SGX Programming Environment

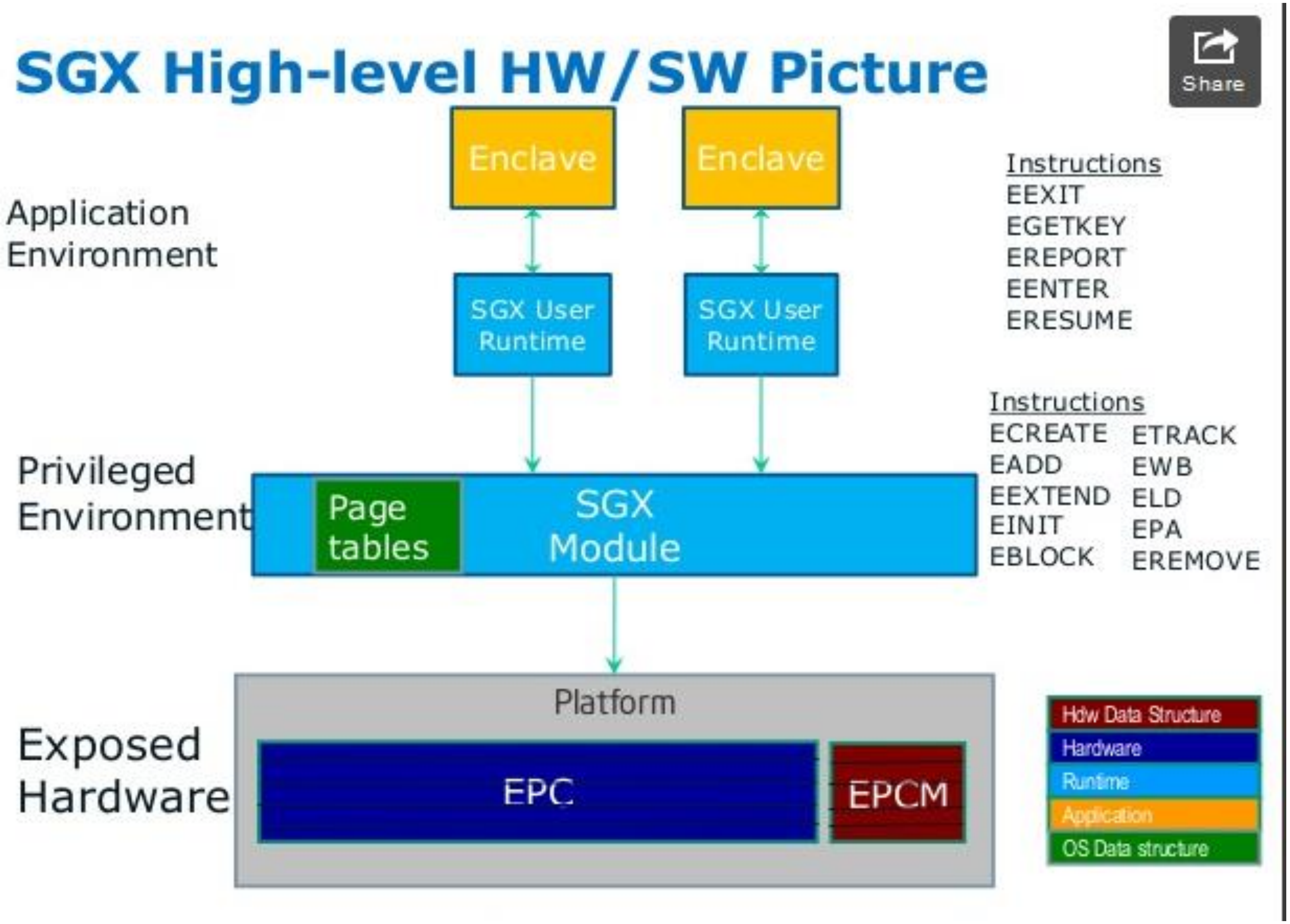
Protected execution environment embedded in a process



- With its own code and data
- Provide confidentiality and integrity protection
- With controlled entry points
- Support for multiple threads
- With full access to app memory

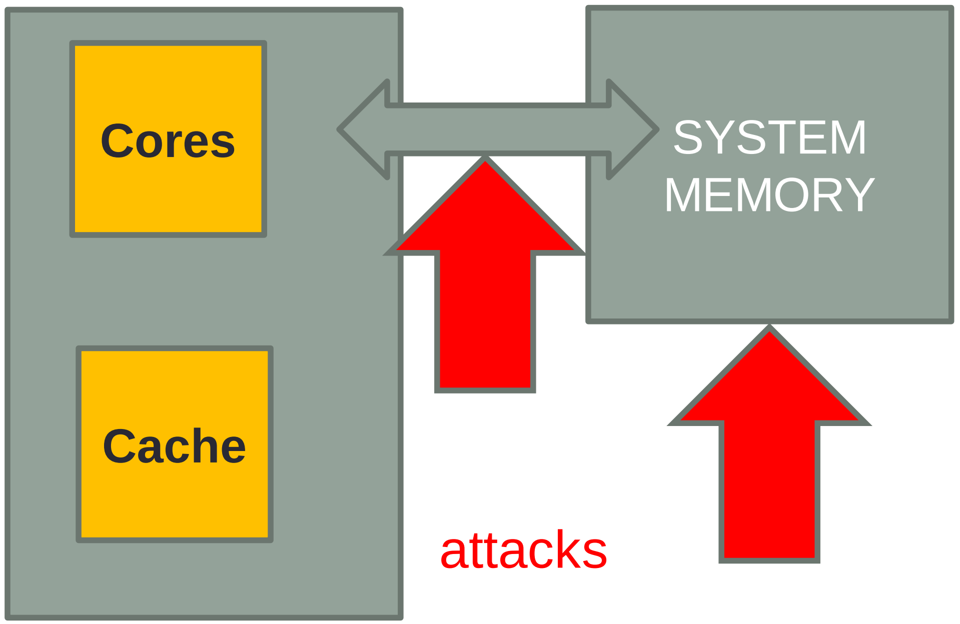
TCS= Thread Control Structure

SGX High-level HW/SW Picture



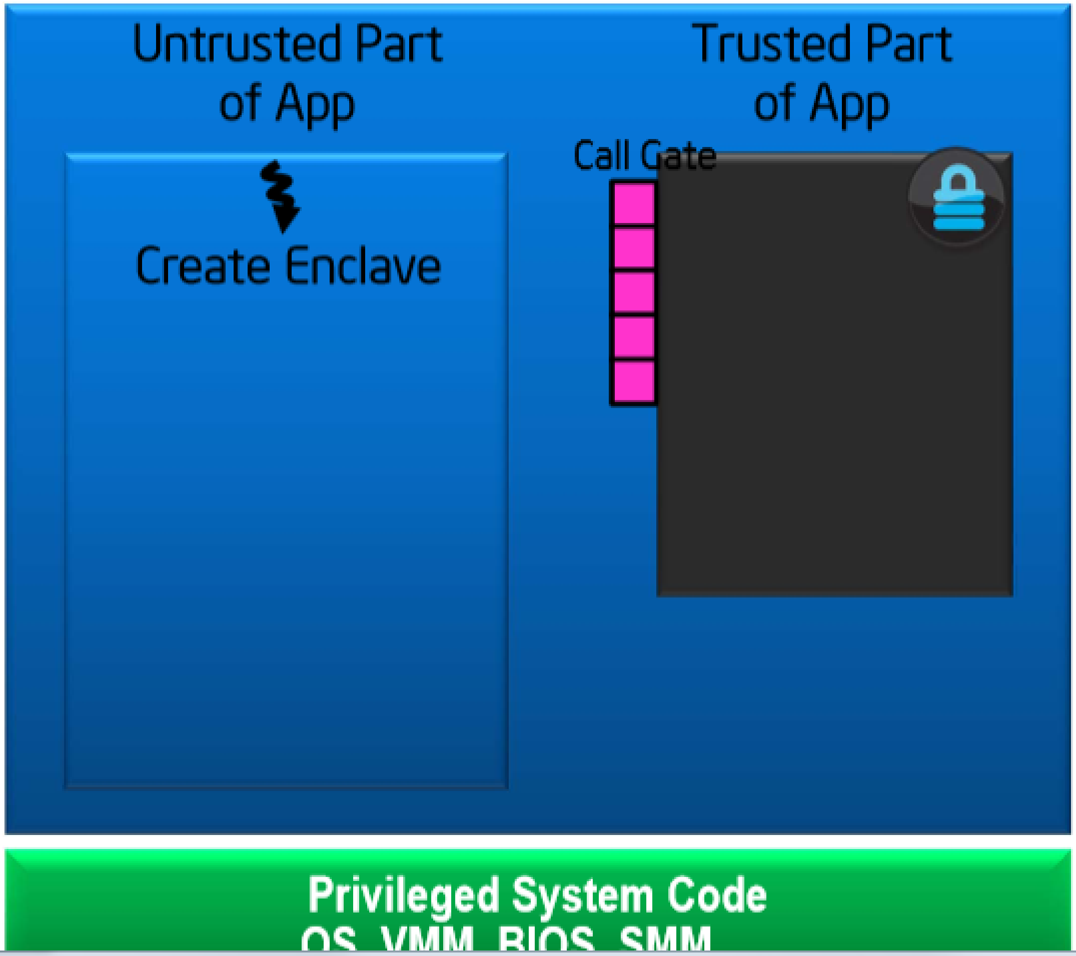
The Enclave Page Cache (EPC) is protected memory used to store enclave pages and SGX structures. The EPC is divided into 4KB chunks called an EPC page. The Enclave Page Cache Map (EPCM) is a protected structure used by the processor to track the contents of the EPC.

Protection against Memory Snooping



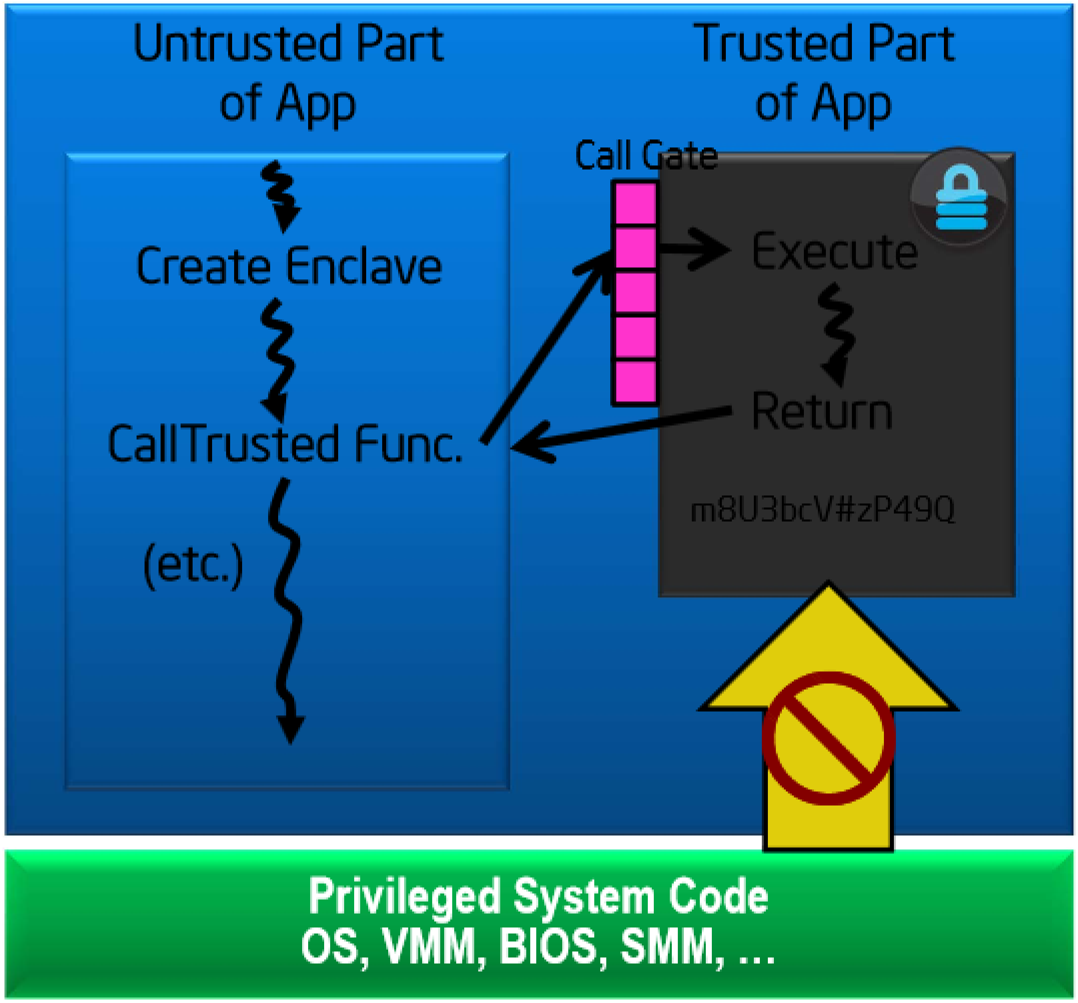
1. **Security perimeter is the CPU package boundary**
2. **Data and code unencrypted inside CPU package**
3. **Data and code outside CPU package is encrypted/integrity protected,**
4. **External memory reads and bus snoops tapping gives access to encrypted**

Creation



1. App is built with trusted and untrusted parts
2. App runs & creates enclave which is placed in trusted memory

Using



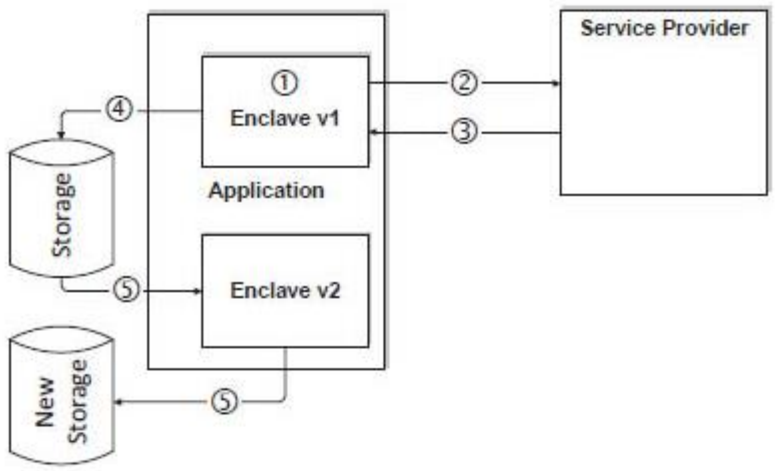
1. App is built with trusted and untrusted parts
2. App runs & creates enclave which is placed in trusted memory
3. Trusted function is called; code running inside enclave sees data in clear; external access to data is denied
4. Function returns; enclave data remains in trusted memory

Example: Life-cycle of enclave SW

Intel® SGX-enabled software does not ship with sensitive data. After the software is installed, it contacts the service provider to have data remotely provisioned to the enclave. .

1. **Enclave Launch** – The untrusted application launches the enclave environment to protect the service provider's software. While the enclave is built, a secure log is recorded , the enclave's "Measurement.

2. **Attestation** –The enclave contacts the service provider to have its sensitive data provisioned to the enclave. The platform produces a secure assertion that identifies the hardware environment and the enclave.



Example: Life-cycle of enclave SW

3. **Provisioning** – The service provider assesses the trustworthiness of the enclave. It uses the attestation to establish secure communication and provision sensitive data to the enclave. Using the secure channel, the service provider sends the data to the enclave.

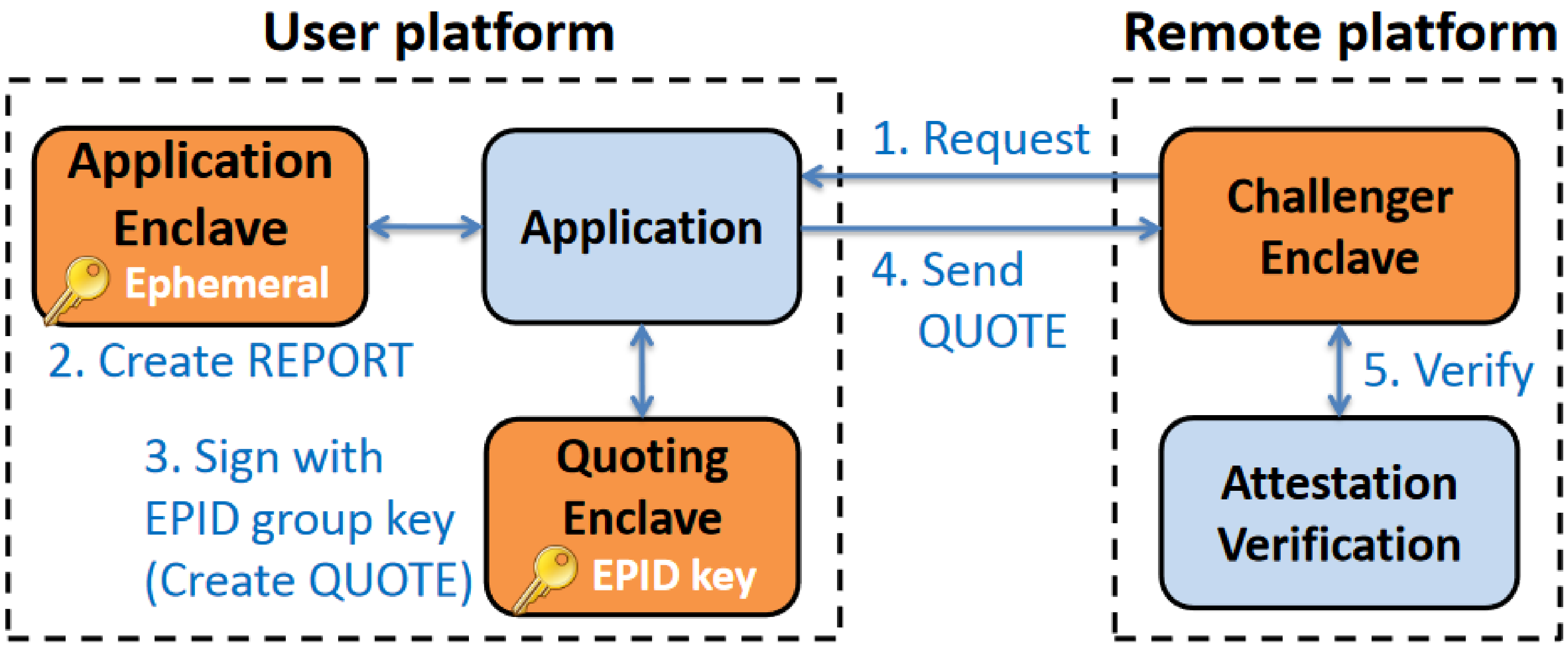
4. **Sealing/Unsealing** – The enclave uses a persistent hardware-based encryption key to securely encrypt and store its sensitive data in a way that ensures the data can be retrieved only when the trusted environment is restored.

5. **Software Upgrade** – Enclave software updates might be required by the service provider. To streamline the migration of data from an older software version to the newer version, the software can request seal keys from older versions to unseal the data and request the new version's seal so that the sealed data won't be available to previous versions of the software.

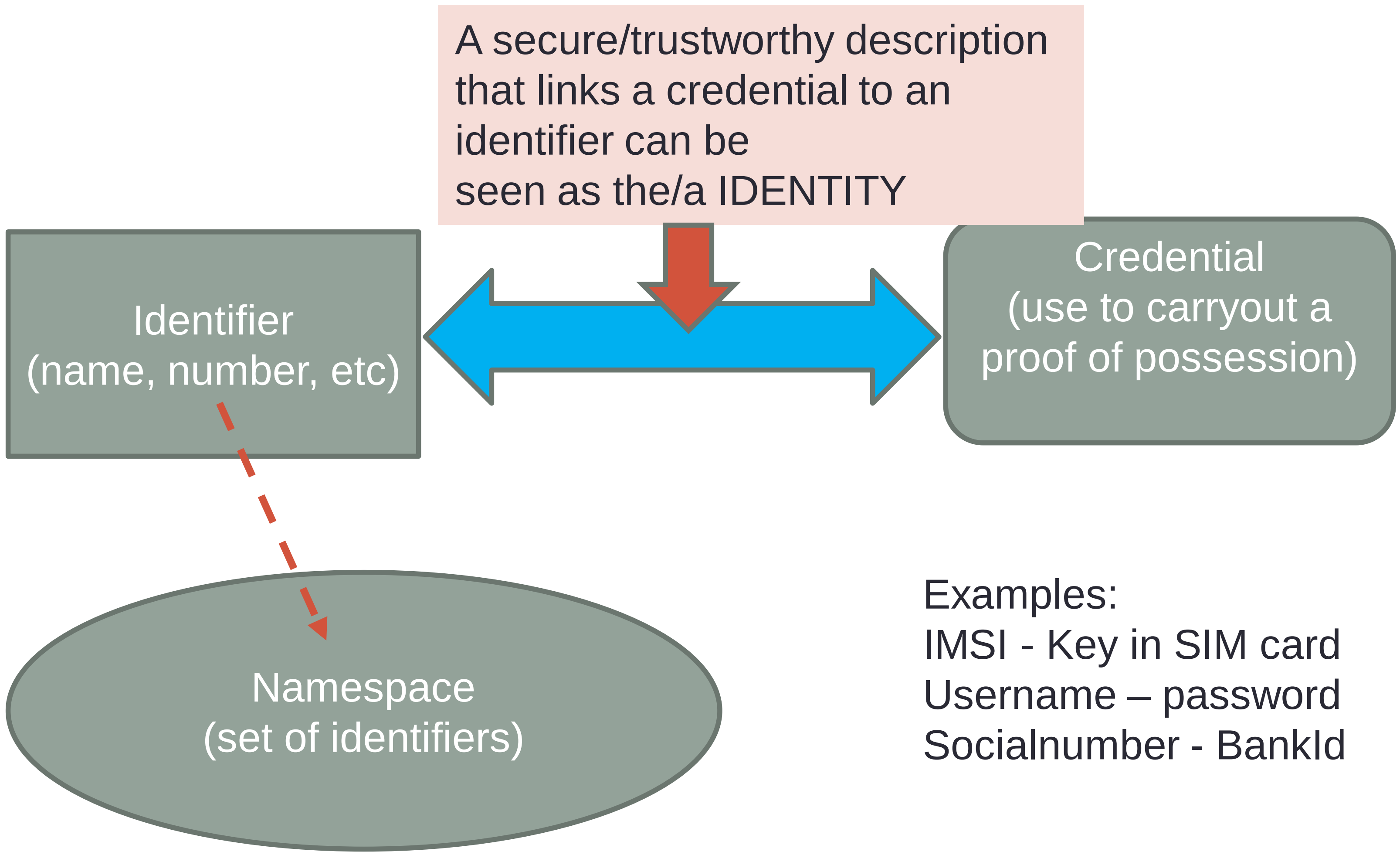
Attestation and Sealing

Attest an application on remote platform

- Check the integrity of enclave (hash of code/data pages)
  - Verify whether enclave is running on real SGX CPU
  - Can establish a secure channel between enclaves



Identities



A secure/trustworthy description that links a credential to an identifier can be seen as the/a IDENTITY
Identifier (name, number, etc)
Credential (use to carryout a proof of possession)
Namespace (set of identifiers)

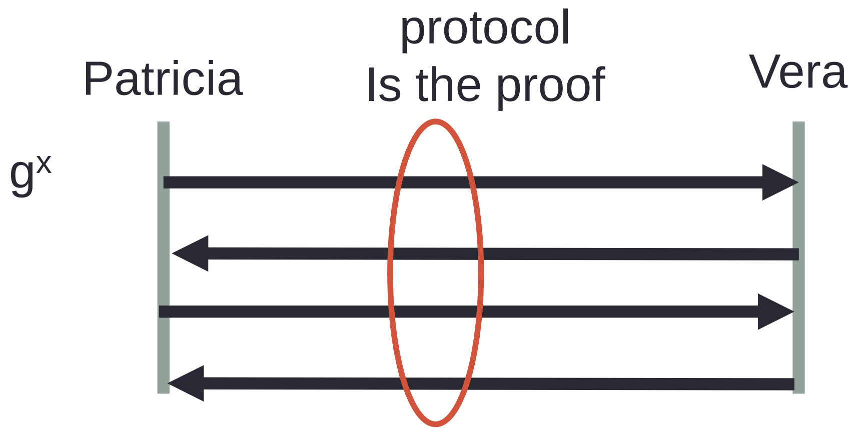
Examples:
IMSI - Key in SIM card
Username – password
Socialnumber - BankId

Zero-knowledge proof

Example: Discrete log over GF(q), q prime

- Patricia (prover) wants to prove to Vera (verifier) that she knows the logarithm of $y=g^x$ mod p (that is x) WITHOUT Vera giving x or let her learn anything about x.

- Protocols that achieve this are called zero-knowledge proofs.

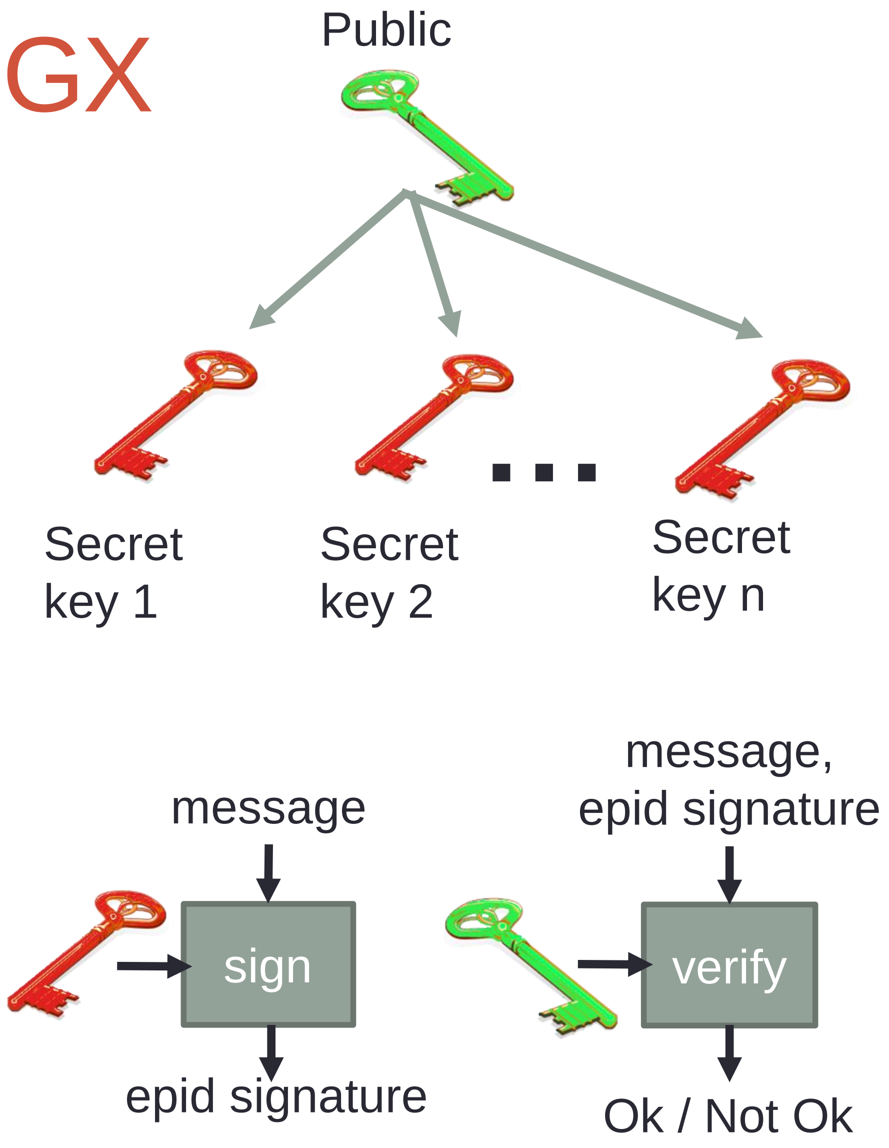


At the end Vera is convinced Patricia knows x, but she learnt nothing about x

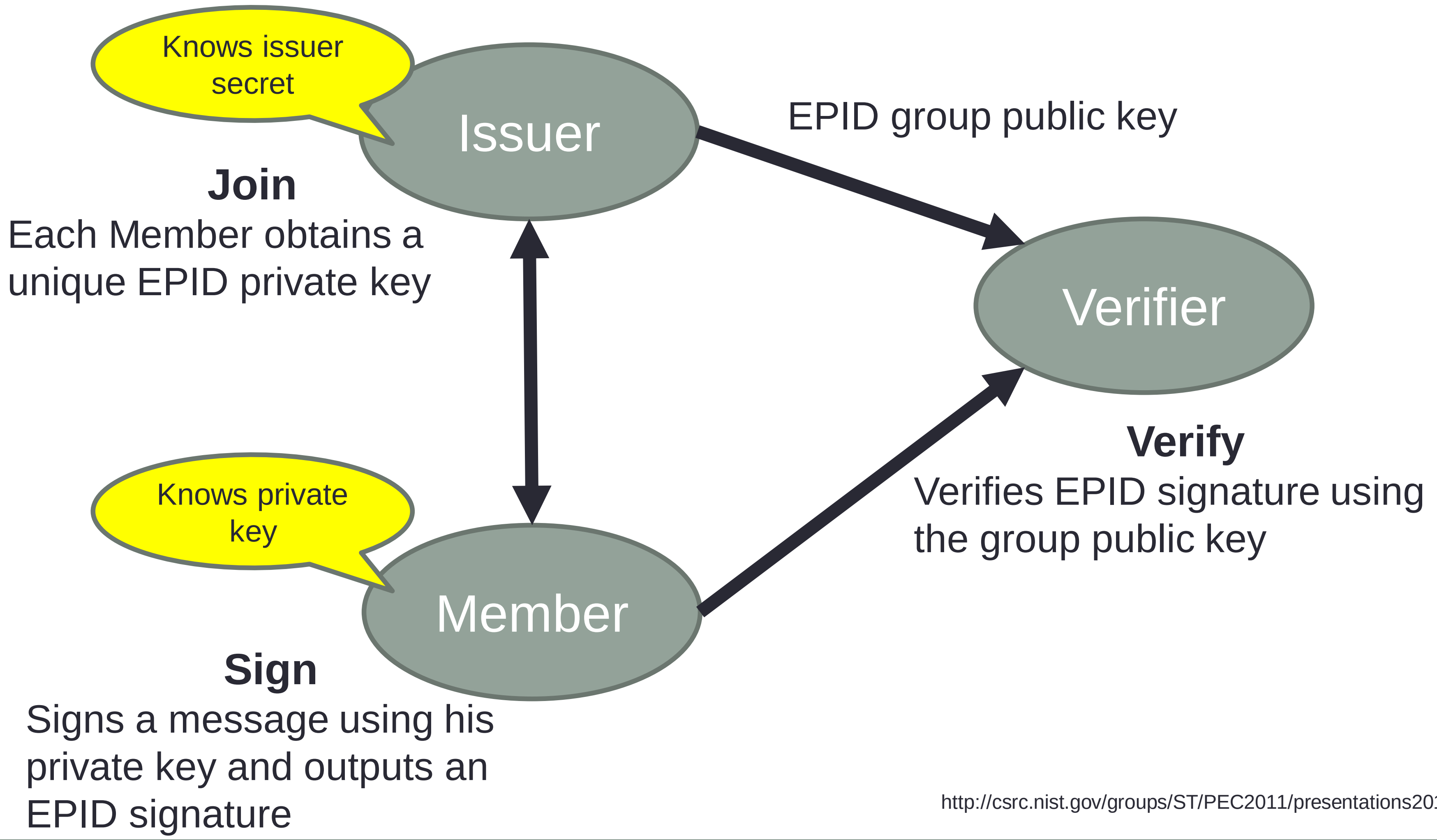
You will learn about these protocols in Advanced Web Security

EPID identities in SGX

- To support attestation SGX can use EPID identities
- **One group public key** corresponds to multiple private keys
  - Each unique private key can be used to generate a signature
  - Signature can be verified using the group public key



EPID setup


Knows issuer secret
Issuer
EPID group public key
Join
Each Member obtains a unique EPID private key
Verifier
Verify
Verifies EPID signature using the group public key
Knows private key
Member
Sign
Signs a message using his private key and outputs an EPID signature

http://csrc.nist.gov/groups/ST/PEC2011/presentations2011/b

Unlinkability

Unlinkability property depends upon Base **B**

- Signature includes a pseudonym B^f where
  - **B** is base chosen for a signature and revealed during the signature
  - **f** is unique per member and private

- Random base: Pseudonym R^f where R is random
  - signatures are unlinkable
- Name base: Pseudonym N^f all where N is a name of verifier
  - Signatures still unlinkable for different verifiers
  - Signatures using common N are linkable

ARM TRUSTZONE

Samung Knox: so what is TrustZone ?

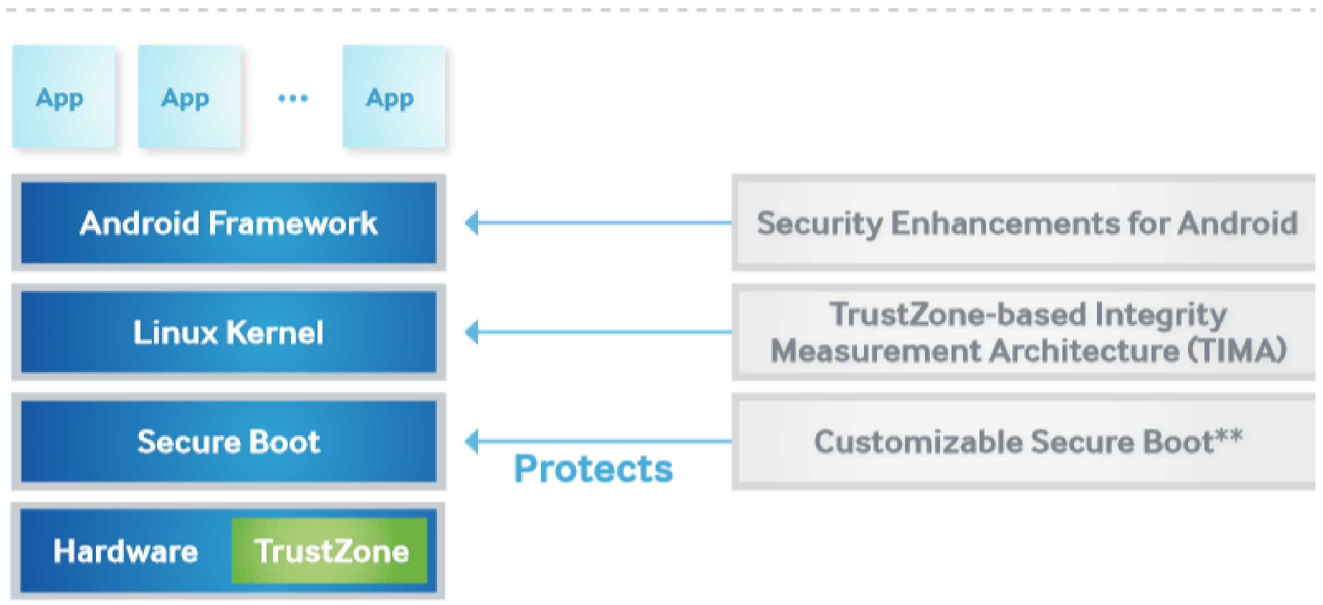
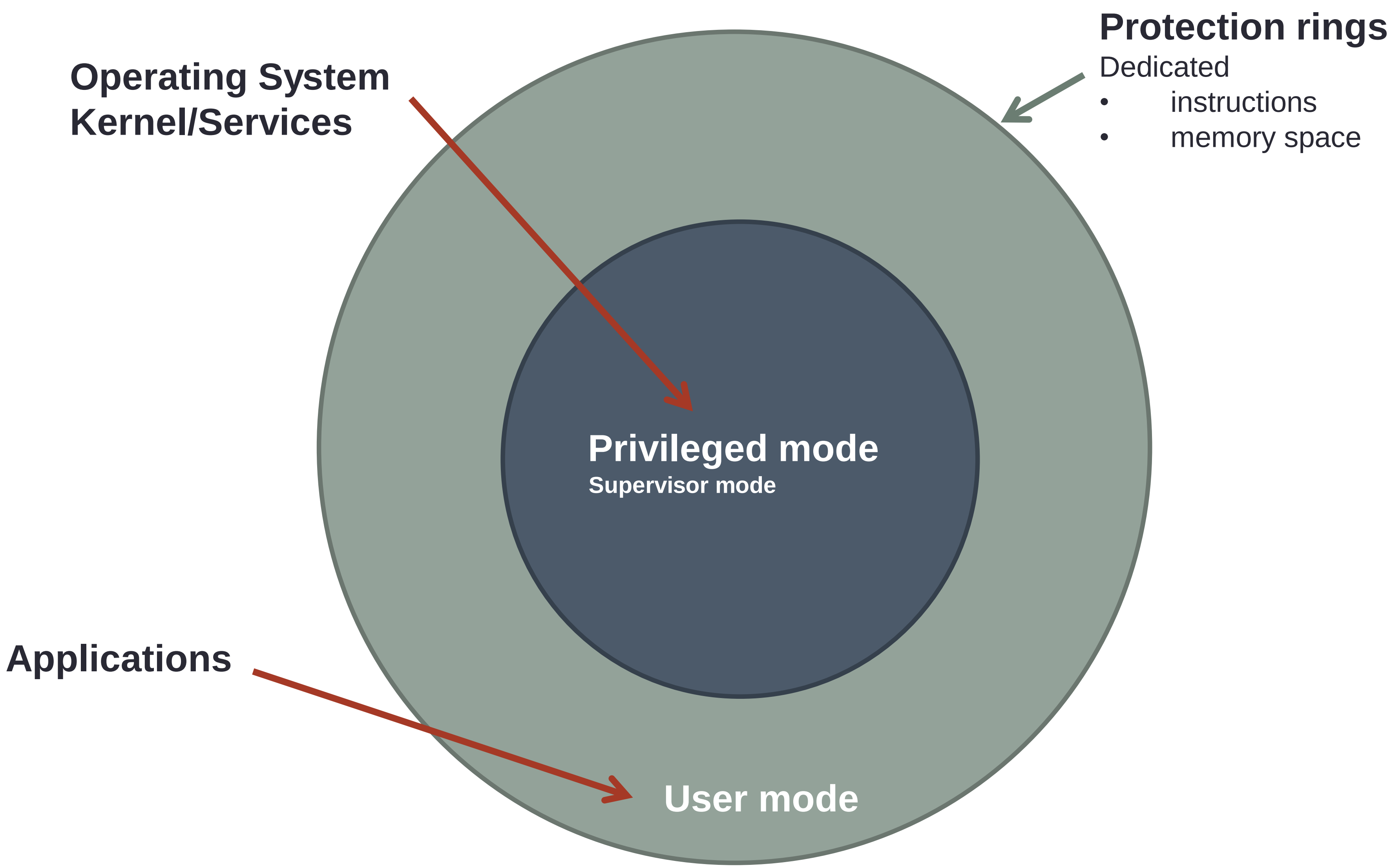

App
App
...
App
Android Framework
Linux Kernel
Secure Boot
Hardware
TrustZone
Security Enhancements for Android
TrustZone-based Integrity Measurement Architecture (TIMA)
Customizable Secure Boot**
Protects

Figure 2 – Samsung KNOX System Security Overview

Samsung KNOX addresses security in a comprehensive, three-prong strategy:

- Customizable Secure Boot**
- TrustZone-based Integrity Measurement Architecture (TIMA)
- Security Enhancements for Android

ARM standard approach



Protection rings

Dedicated

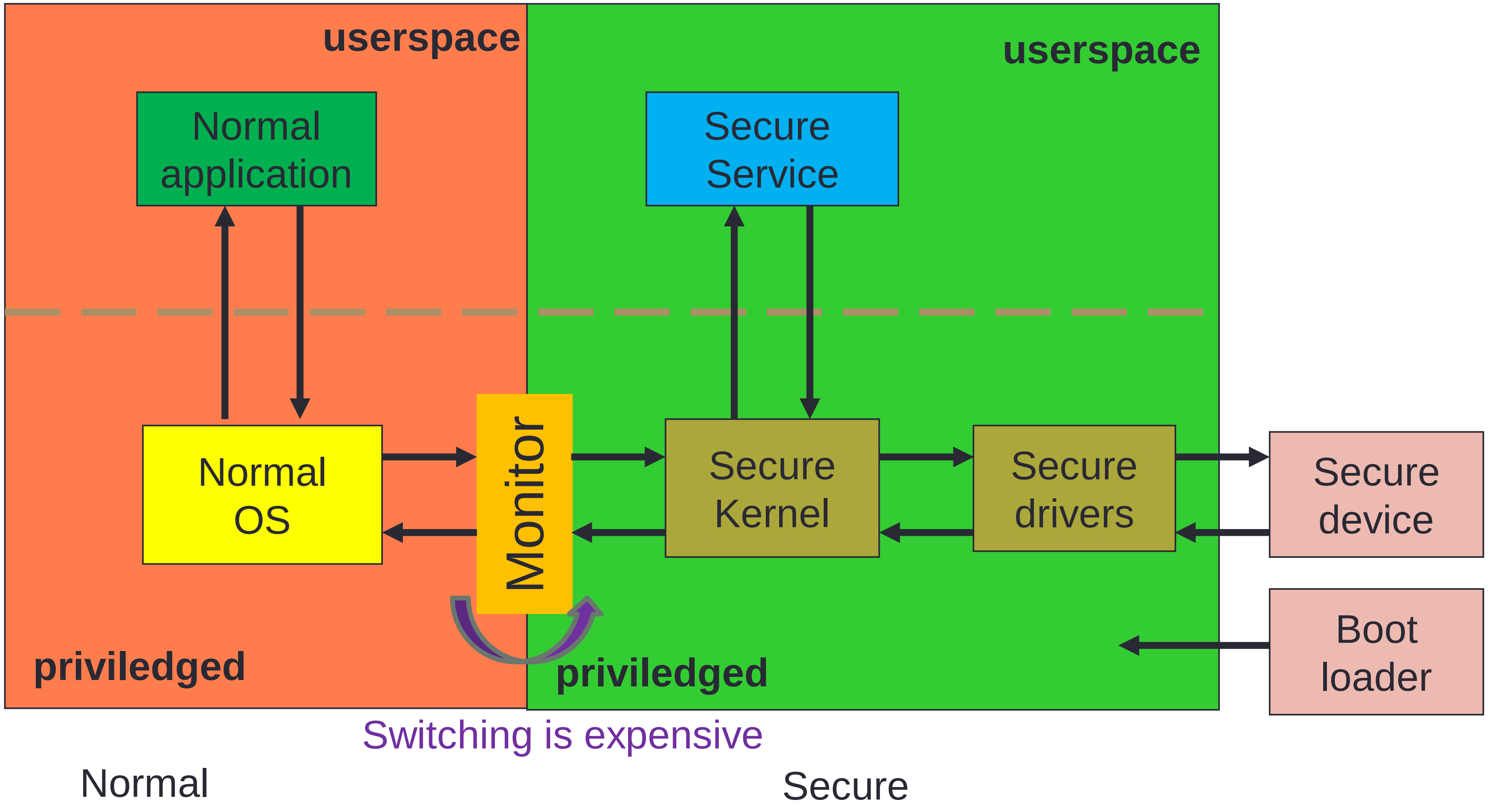
- instructions
- memory space

Separation of sensitive ops and data

- Since too much code in running in user space and even in the privileged space:

⇒ Sensitive applications and data cannot be given good guarantees that other running code cannot tamper or get access.

Switching from Normal to Secure



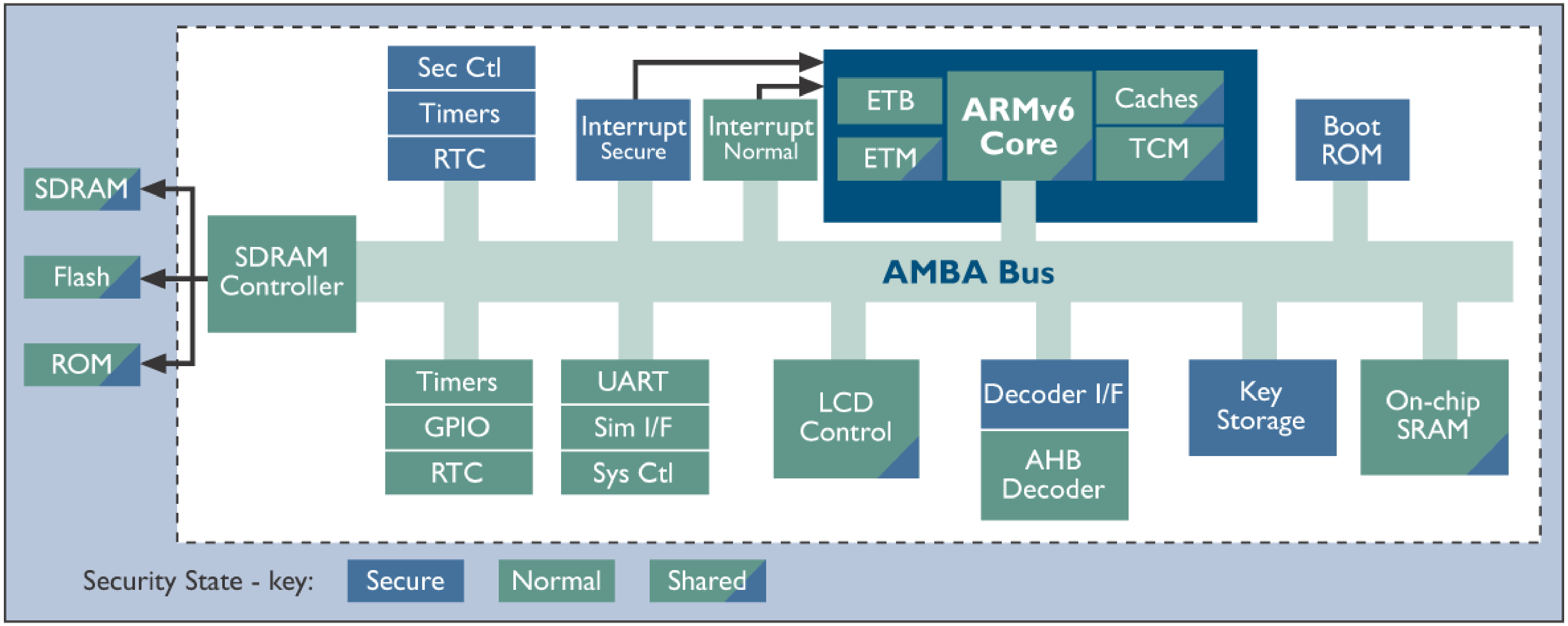
Switching is expensive

Access control

- TrustZone HW
  - Normal world cannot access Secure world owned objects
  - Secure world can access any object if privilege allows it.

- Ring architecture
  - Code can execute as privileged(P) or unprivileged (U) (or user mode). Unprivileged execution limits or excludes access to some resources. Privileged execution has access to all resources.
  - The basic rule is that if the current privilege is unprivileged mode, it cannot access the memories whose corresponding page structure has U/P bit clear. In other words, an unprivileged mode task or program cannot read (write, or fetch) access to the memory that belongs to supervisor or privileged mode.

TrustZone uses Hardware features - Example System



Other memory space protection: NX

- Memory space for data can be marked as not executable. This executable space protection often called NX (No-eXecute) or as XD (eXecute Disable),

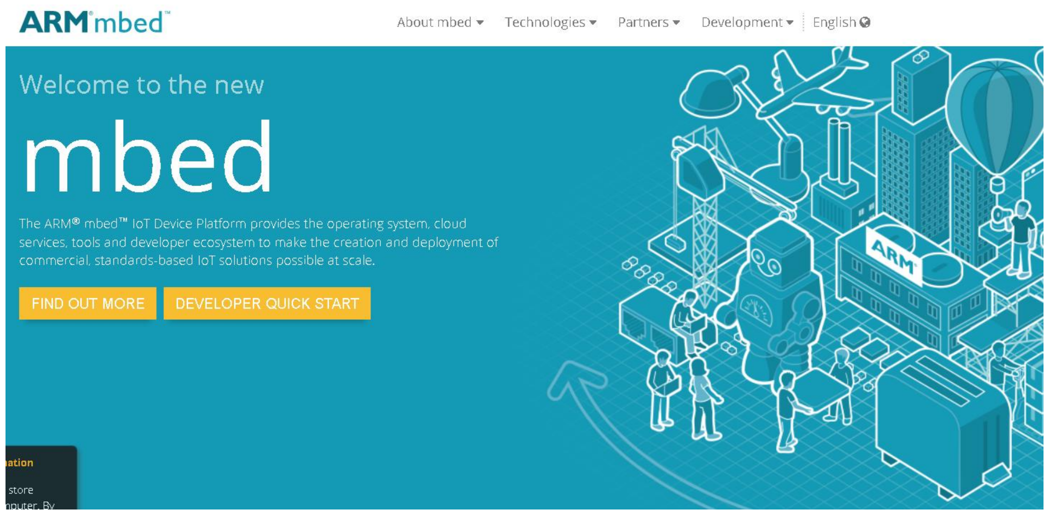
- An OS can use this feature by marking some region of memory spaces not executable.

- Typical use: stack or heap memory space may be marked as NX. This helps to prevent certain buffer-overflow exploits from succeeding, particularly those that inject code and execute in controlled stack or heap space.

Trustzone in embedded CPU systems

- Beware it does not work exactly the same. Especially the switching in and out the secure world has been redesigned (basically now via HW).

- ARM TrustZone in mbed (coming soon)
- http://www.mbed.com



Welcome to the new

mbed

The ARM® mbed™ IoT Device Platform provides the operating system, cloud services, tools and developer ecosystem to make the creation and deployment of commercial, standards-based IoT solutions possible at scale.

ation

store

nputer. By