TÉCNICO LISBOA

Highly Dependable Systems

Lecture 10

Trusted Hardware

Lecturers: Miguel Matos and Rodrigo Miragaia Rodrigues

Trusted Hardware

- So far we covered approaches that make no assumptions about the trustworthiness of the underlying hardware
  - We only trust the algorithms
  - And the cryptographic primitives

Trusted Hardware

- In this lecture
  - What can we additionally achieve if we have hardware support for building secure systems?

Trusted computing base (TCB)

- Set of hardware+software components on whose correctness a given security policy relies
- Larger TCB → wider attack surface
  - E.g., OS and hypervisor are prone to bugs, malware
  - Xen 150K LoC, 40+ vulnerabilities per year
  - Linux, 17M LoC, 100+ vulnerabilities per year

Agenda

- Smart Cards
- Trusted Platform Module
- Trusted Execution Environment
- Trusted Domain Extensions

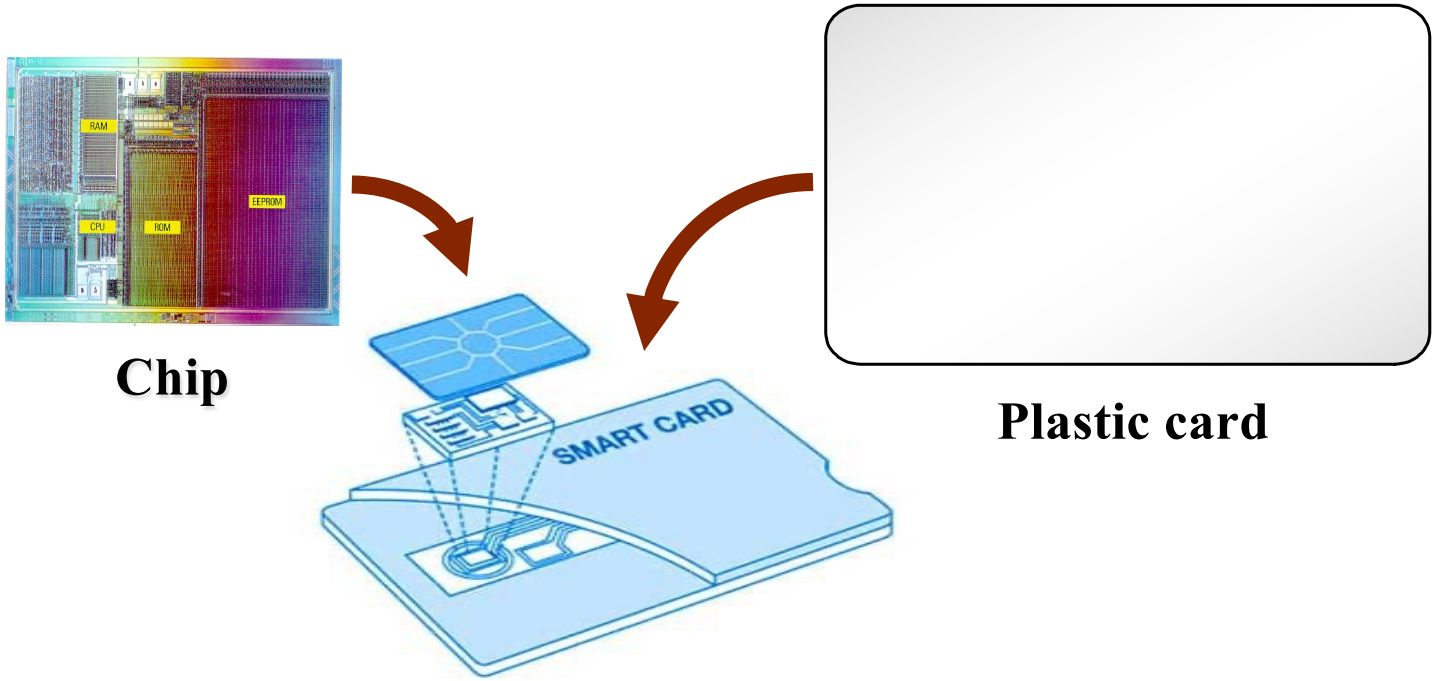
TÉCNICO LISBOA

Smart Cards

Smartcard principles

- A secure way of storing a small amount of sensitive data, and perform some computations
  - Notably cryptographic operations without exposing private keys
- Small size, cheap, low power, small but sufficient amount of memory + CPU horsepower
- Successful strategy of separating:
  - expensive consumer electronic device (bulk production)
  - cheap authentication hardware (personalized)

Smart Card



Chip

Plastic card

Smart Card applications

- Retail
  - Sale of goods using Electronic Purses, Credit / Debit
  - Vending machines
  - Loyalty programs
  - Tags & smart labels
- Government
  - identification
  - Passport
- Et cetera
- Healthcare
  - Insurance data
  - Personal data
  - Personal file
- Communication
  - GSM
  - Payphones Transportation
  - Car Protection
- Pay TV operators

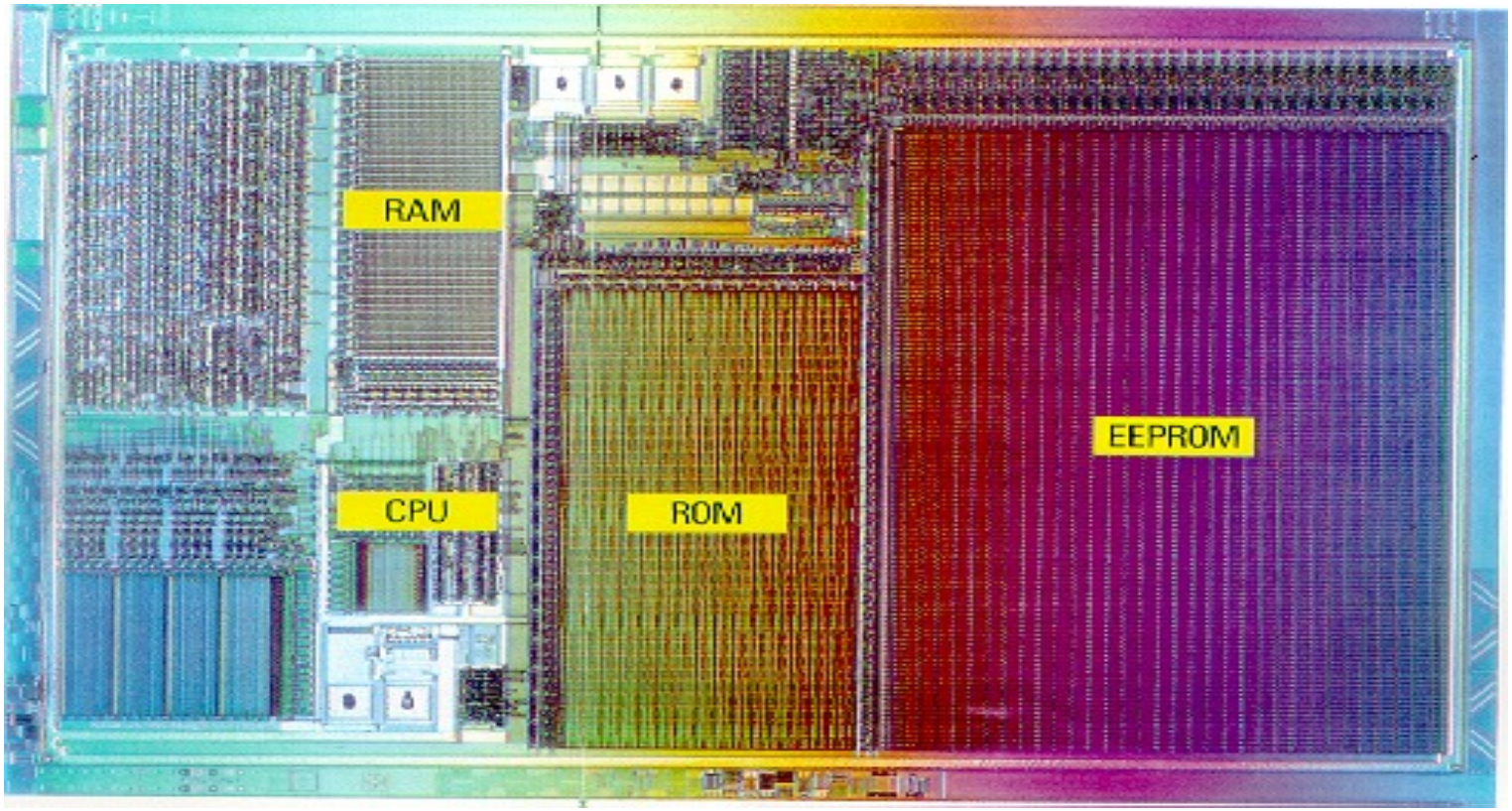
Well-known Smart Card Application

- Store sensitive data (private keys)
- Store these secrets with strong integrity and confidentiality
- Perform computation (e.g., signing) without exposing keys

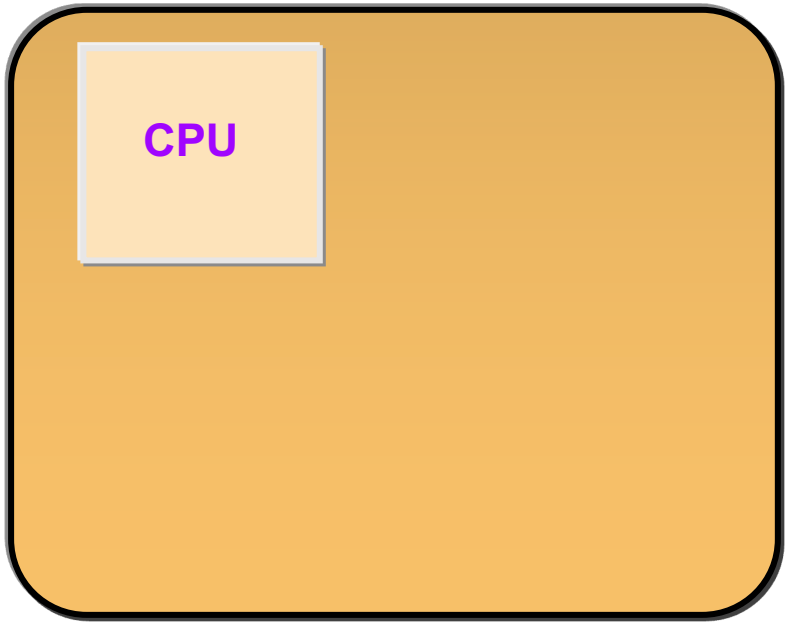
Smart Card characteristics

- Simple processors
  - 8-32 bits @ a few tens of MHz (up to 32 MHz)
- Small memory
  - Up to 128kB of ROM
  - Up to 128 kbit of EEPROM (electrically erasable programmable ROM)
  - Up to 4kB of RAM
- Crypto co-processors (incl. randomness generator)
- Low manufacturing price (< 10€)
- No internal power source
  - Source of potential attacks
- Low power consumption (tens of mW)

Inside the smart card



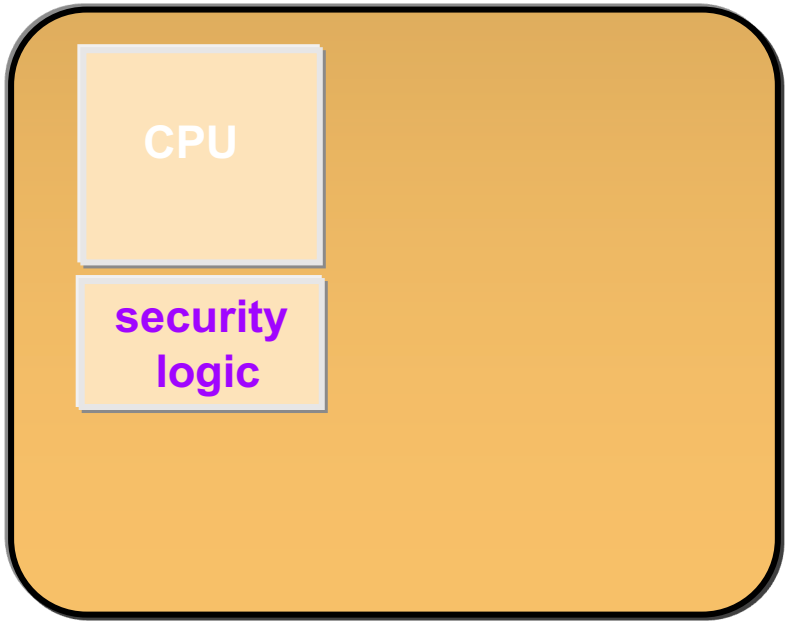
Inside the smart card



Central Processing Unit:

- heart of the chip

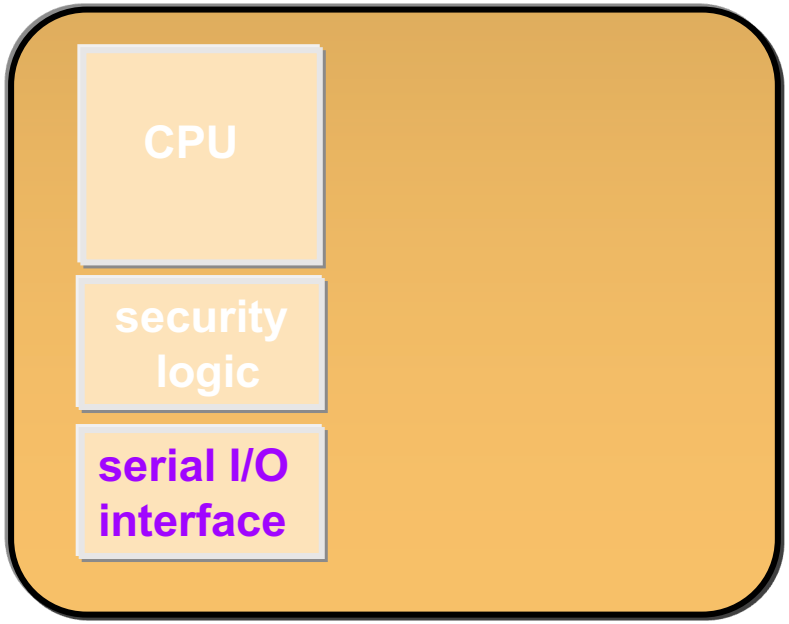
Inside the smart card



security logic:

- detecting abnormal conditions,
- e.g. low voltage

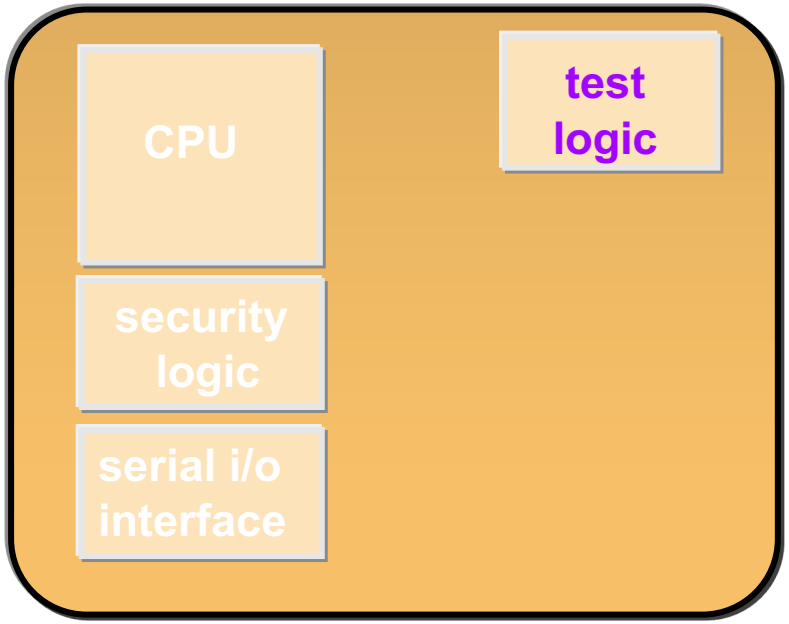
Inside the smart card



serial I/O interface:

- contact to the outside world

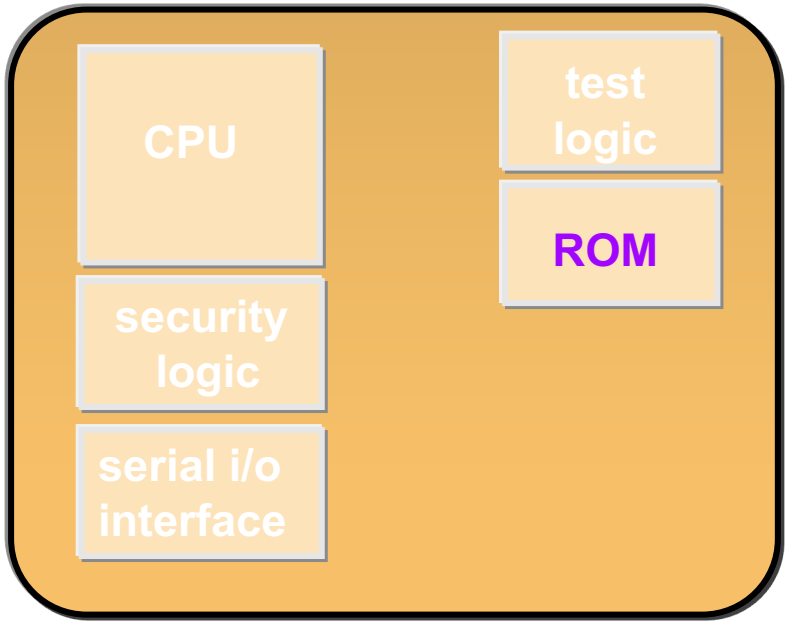
Inside the smart card



test logic:

- self-test procedures

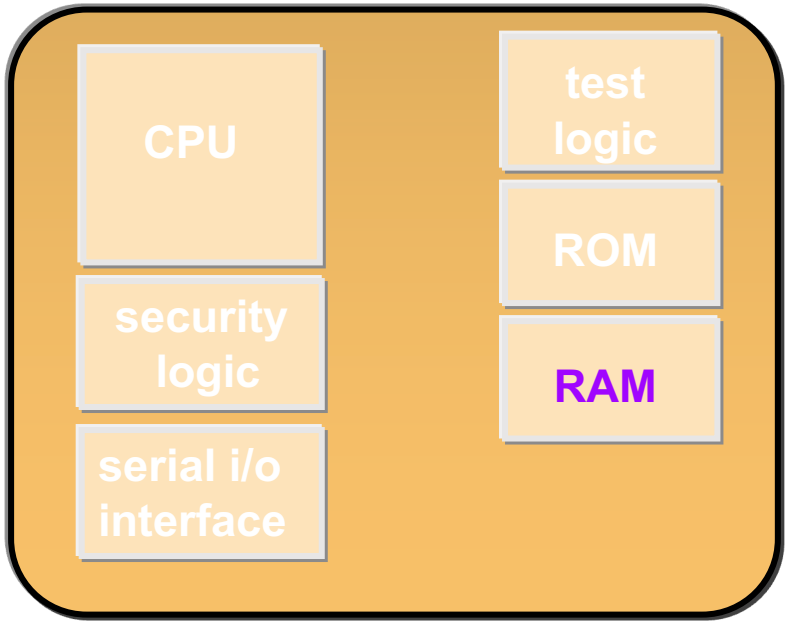
Inside the smart card



ROM:

- card operating system
- self-test procedures

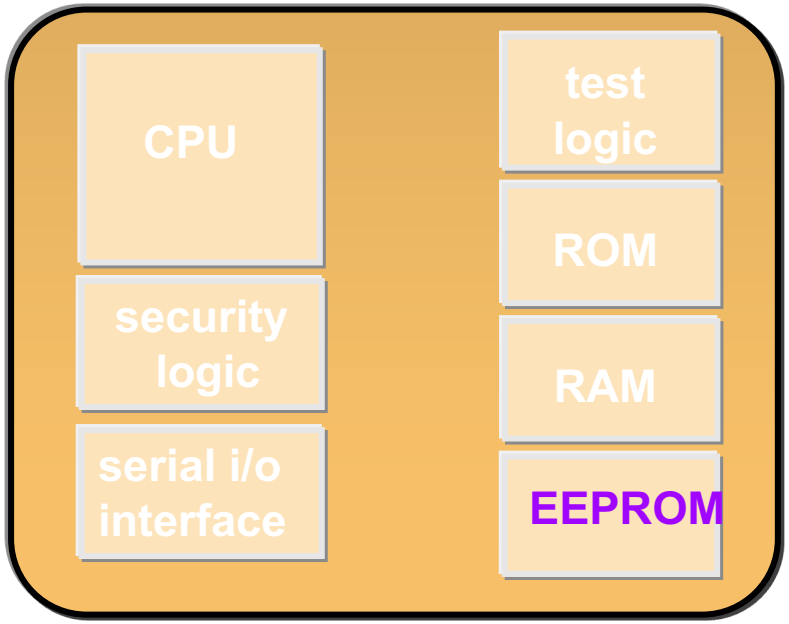
Inside the smart card



RAM:

- ‘scratch pad’ of the processor

Inside the smart card

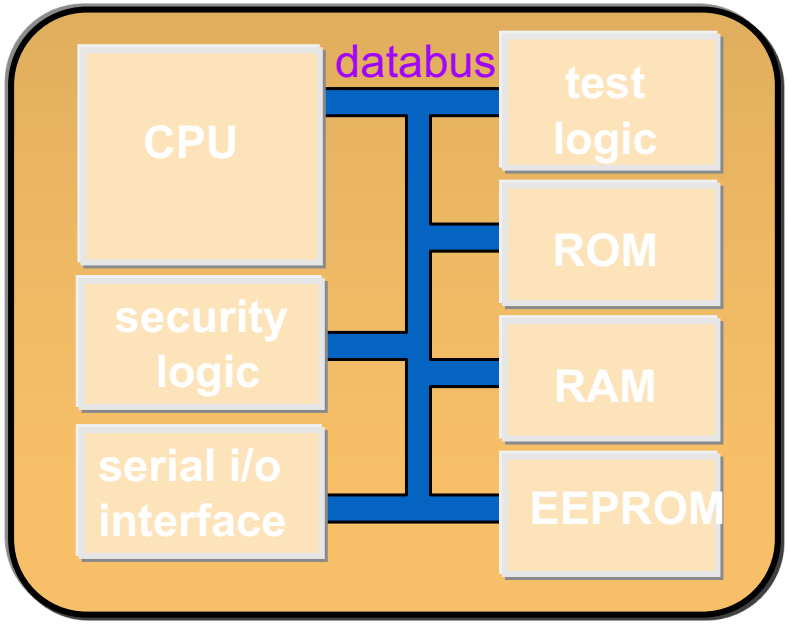


EEPROM:

- cryptographic keys
- PIN code
- biometric template
- balance
- application code

Recall: private key never leaves card!

Inside the smart card



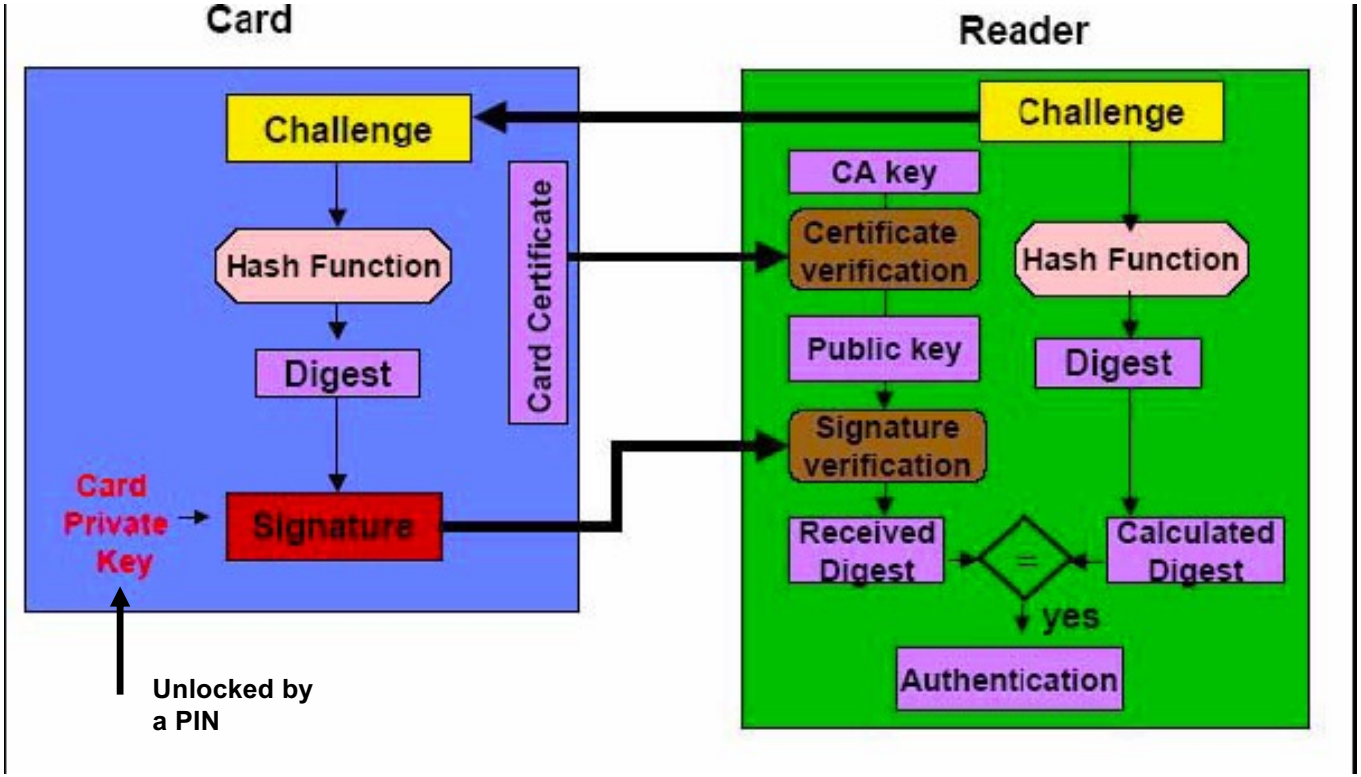
databus:

- connection between elements of the chip
- 8,16, or 32 bits wide

TÉCNICO LISBOA

Smartcard security

Authentication with Smartcards



Authentication with Smartcards

- Multi-factor authentication - combination of:
  - What you know
    - e.g.: passwords, PINs
  - What you have
    - e.g.: OTP tokens, smartcards
  - What you are (biometrics)
    - e.g.: fingerprints, iris scans, face recognition

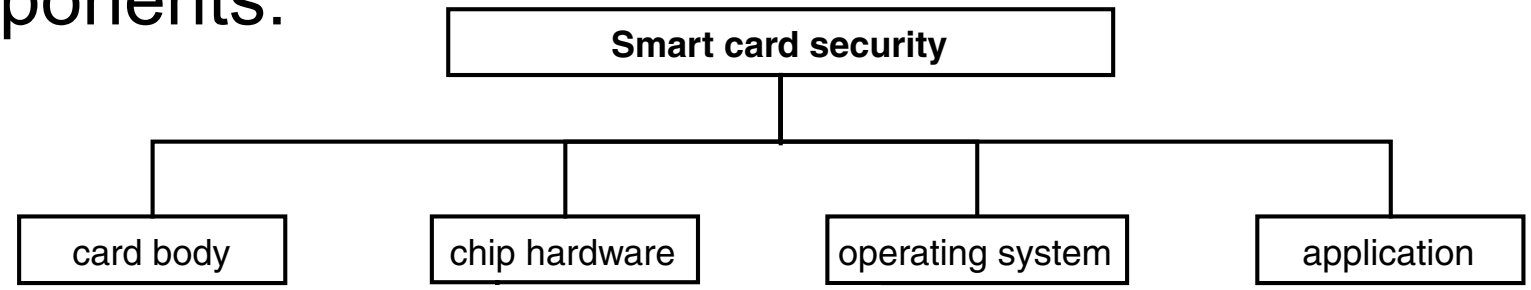
- Typically two-factor authentication is used
  - e.g.:
    - PIN + Card (e.g. ATMs)
    - Password + One-time-password (OTP) token
    - Fingerprint / biometrics + Smartcard

Smartcard security

- Security properties of smartcards are very attractive:
  - Enables authentication and digital signatures without exposing private keys
  - Relies on a small Trusted Computing Base (TCB)
  - Using smartcards for authentication functions of a more expensive device enables:
    - bulk production of the device
    - easy replacement of the authentication if needed
    - Low price for manufacturing smartcards (a few euros)
- However, not full proof
- Main challenge: low price makes it easy for attackers to acquire many devices and try to tamper with them

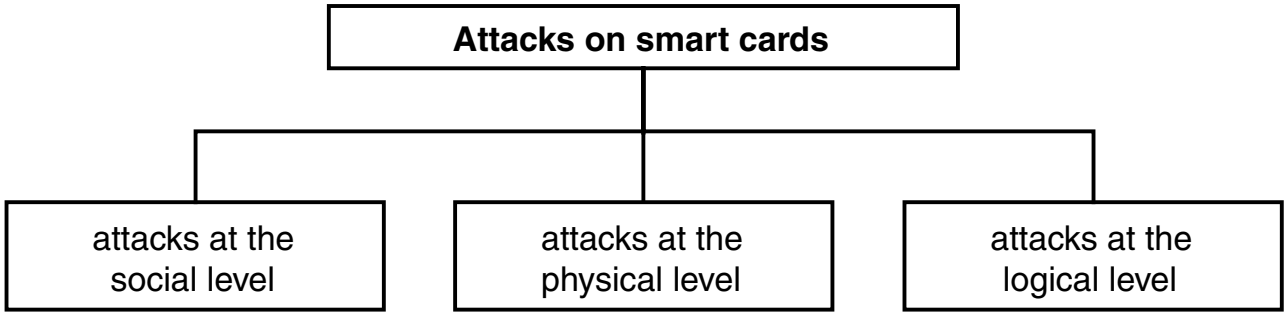
Smartcard security

- Smartcard security is ensured by 4 main components:



Attacks on smartcards

- 3 main types of attacks



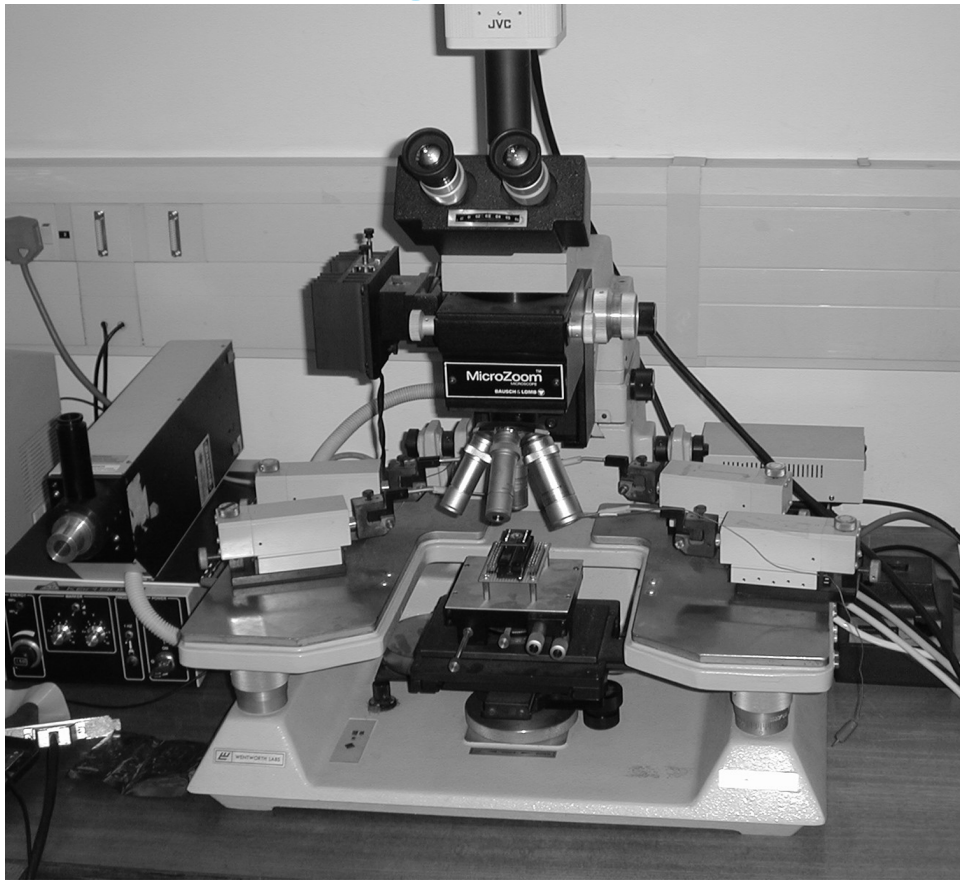
Physical attack types

- As with the cryptanalysis of cryptographic protocols, physical attacks can be divided into:
  - Active attacks:
    - E.g., the attacker manipulates the data transmission process or the microcontroller.
  - Passive attacks:
    - E.g., attacker may make measurements on the semiconductor device.

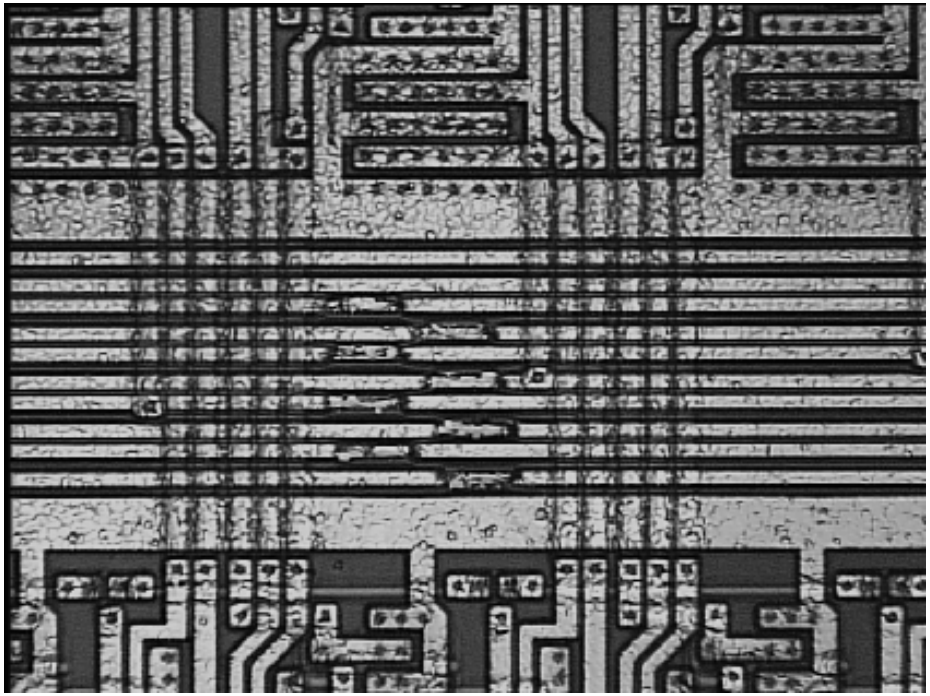
Example physical attacks

- Probing stations: microscopes + micromanipulators
  - Allow for inserting fine probes on chip surface
- Can be used to snoop the smartcard bus
- Trace from bus during encryption operation is sufficient to derive the key
- Some smartcards compute a checksum on memory immediately after reset:
  - give the attacker a complete listing of the card memory contents.

Low cost probing station



Data bus prepared for probing



Excavation of the passivation layer (insulation) with laser shots

TÉCNICO LISBOA

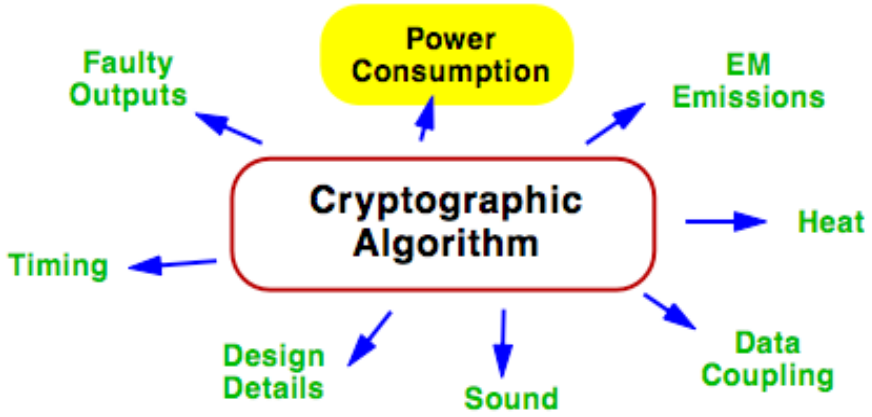
Side Channel Analysis

Two inroads for Attacks

- Traditional Mathematical Attacks
  - Algorithm modeled as ideal mathematical object
  - Attacks mostly theoretical rather than operational
- Implementation Attacks
  - Actual code/hardware is much more complex
  - Example: Side Channel Attacks

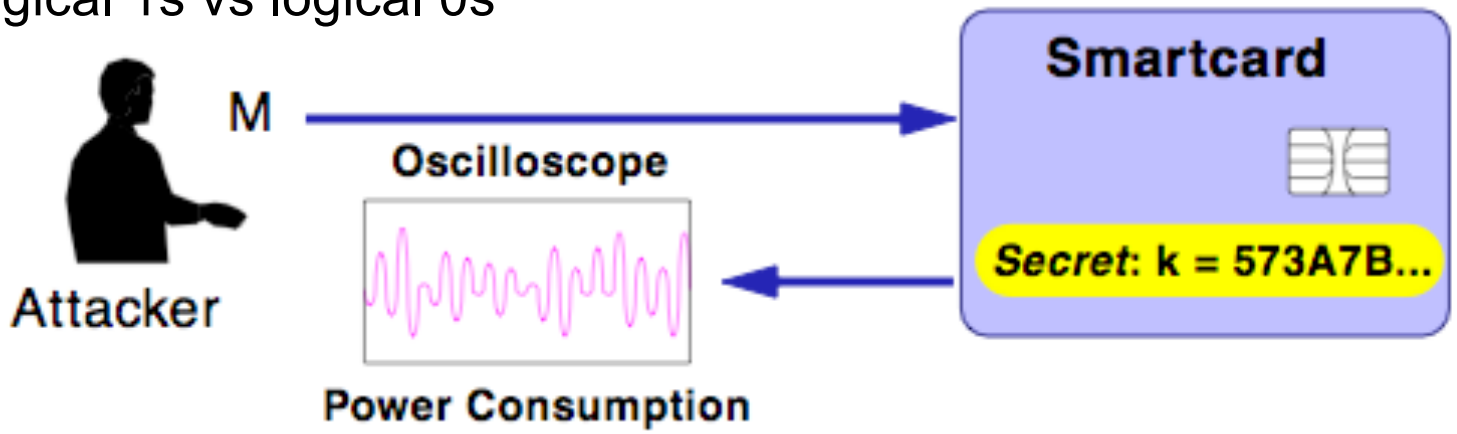
Side Channel Analysis

- Explores the information leaked from the chip
- Not exclusive to smartcards, but important in that context



Power Analysis Attacks

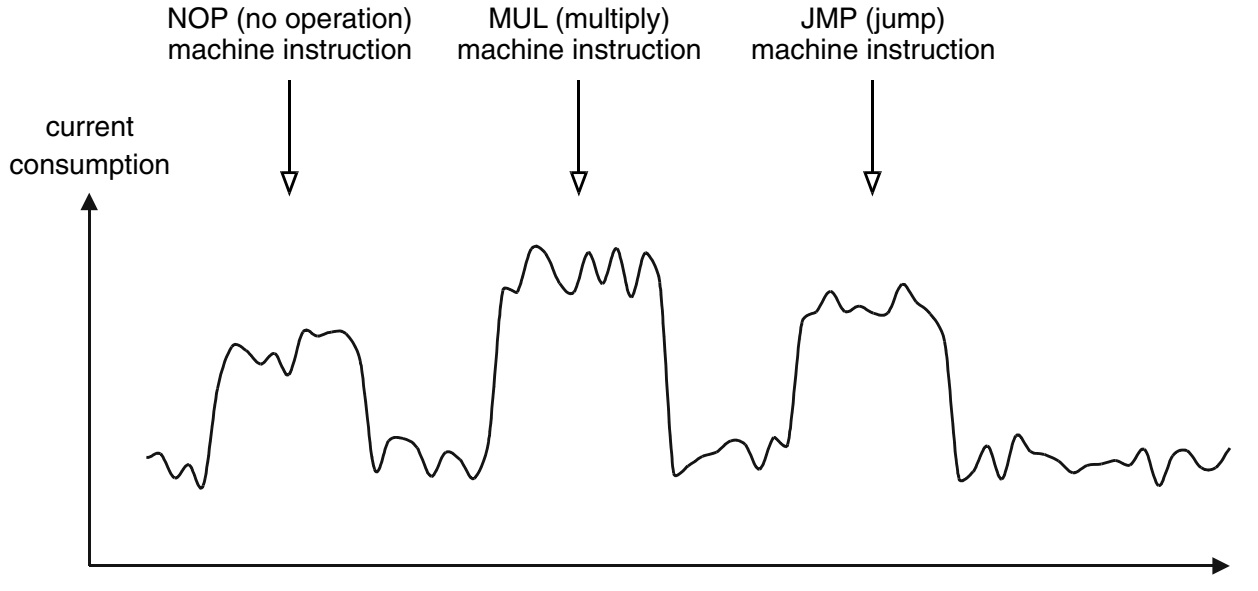
- Measure instantaneous power consumption of a device while it runs a cryptographic algorithm
- Different power consumption depending on code path, and even when operating on logical 1s vs logical 0s



- Non-invasive, passive attacks
  - Can be performed by modified terminals on unsuspecting customers
  - Compromise many cards at the cost of building a single Trojan terminal
  - But in practice more effective when attacker has the time to do several attempts

Simple Power Analysis

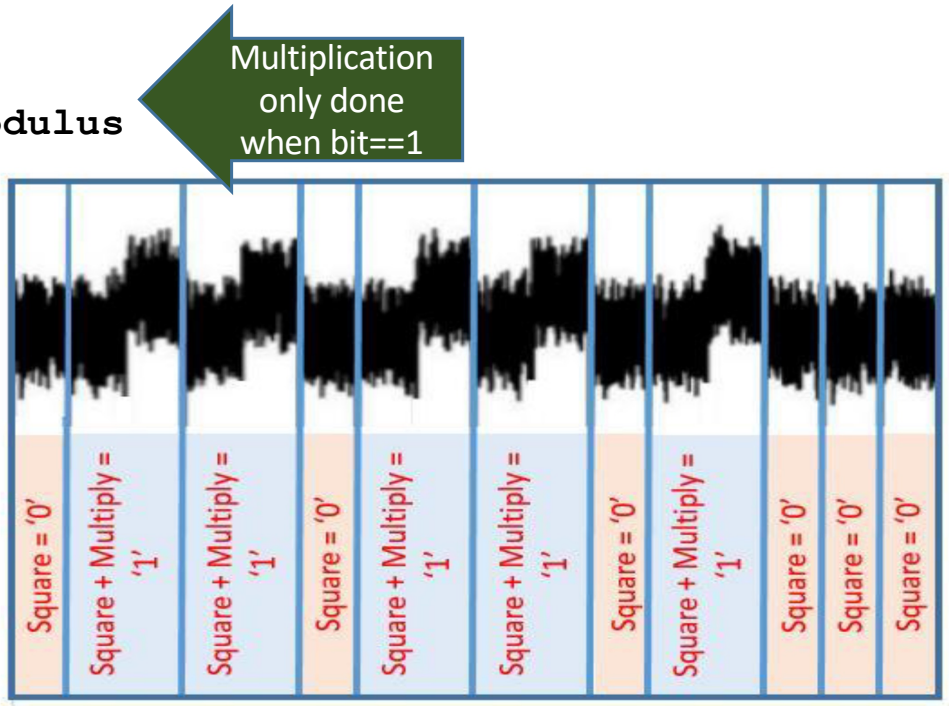
- Attacker knows encryption code and extract a power trace
- Power consumption is dependent on the machine instruction being processed



- But in many cases, it is also dependent on the actual data…

Simple power analysis – RSA

```
function modular_pow(base, exponent, modulus) is
 // . . .
    result := 1
    base := base mod modulus
    while exponent > 0 do
        if (exponent mod 2 == 1) then
            result := (result * base) mod modulus
        exponent := exponent >> 1
        base := (base * base) mod modulus
    return result
```



Sources: Wikipedia (pseudocode) / M. Randolph and W. Diehl, in Cryptography 2020, 4(2), 15

Side-channel countermeasures

- Energy consumption must be independent of instructions and data!
- Hardware:
  - Artificial noise injection – add noise to power supplies, clock signals, or data paths
- Software:
  - Modify the code: research area of “oblivious data structures”
  - adversary cannot infer data from memory accesses

Threat model

- Who is the adversary in smartcard usage?
  1. (Traditional view) In a banking transaction, need to consider attackers that may listen or tamper with messages between card, terminal, and bank
  2. But the owner of the card is also untrusted:
     - May try to extract information from the card
     - Important that keys do not leave card to owner's computer (or to the vendor's terminal)

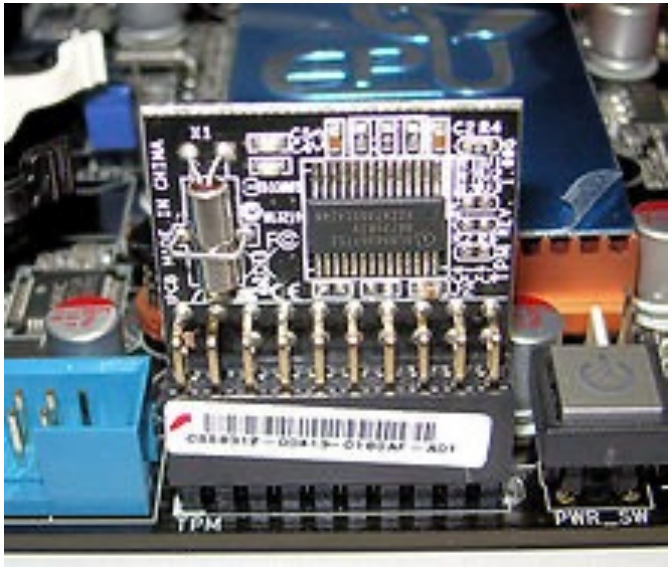
Defending against priviledged software attacks

- When priviledged software is compromised, then “all bets are off”
  - Can inspect+modify code
  - Can inspect+modify data
  - Thus having access to private keys and forging signatures
  - Also having access to sensitive data
  - And tampering with the state of the computations

Trusted Platform Module

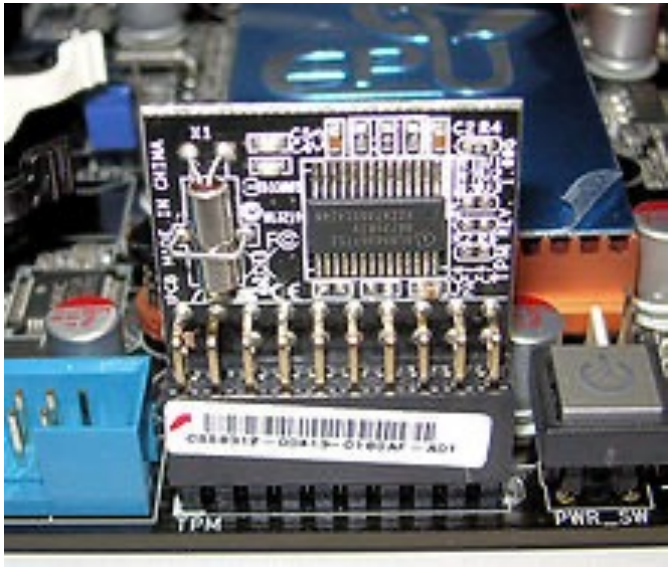
Trusted Platform Module (TPM)

- Standard for **secure cryptoprocessor,** running alongside main processor
- Widely available on most personal computers



Trusted Platform Module (TPM)

- Goal
  - Ensure/verify that machine is in a "trusted state"
- Based on the idea of **secure boot**
- Additionaly used for authentication stores a unique RSA key, burned into the chip



Secure boot – PCR registers

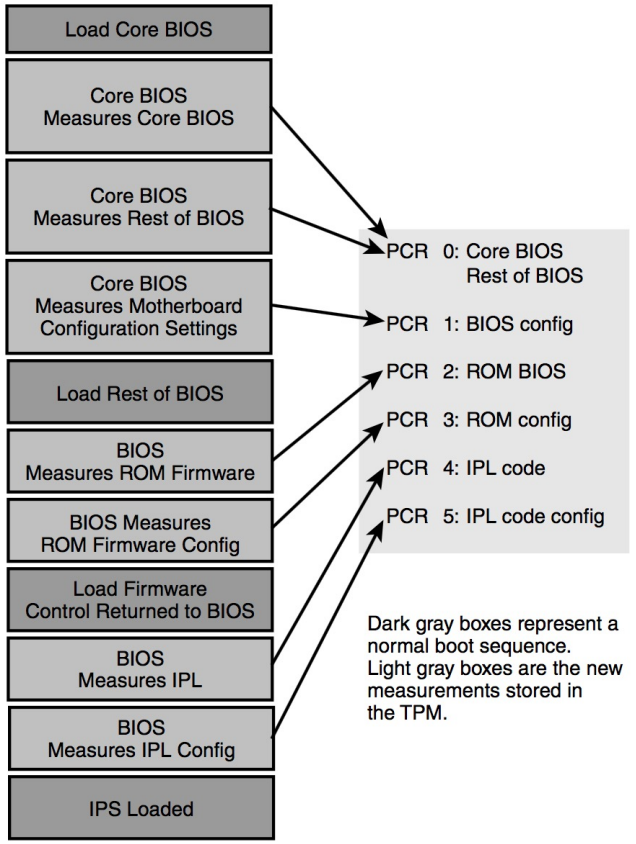
- Array of special registers called PCRs
- Records digest of kernel that booted + entire boot sequence

- Always extended (never overwritten)
  - PCR_{new} = SHA-1(PCR_{old} || new value)
- **Remote attestation** consists of requesting PCR value
  - TPM signs response (alongside nonce to prevent replays)

Secure boot steps (simplified)

- Hardware computes hash of BIOS and stores in PCR, passes control to BIOS
- BIOS measures master boot record (MBR) and extends PCR, passes control to MBR
- MBR measures OS, passes control
- If OS designers want, can recursively measure applications

- What is the TCB in this case?



TPM criticism

- Highly polarized debate – criticized for owner losing control over own machine
- Claims that it was meant for digital rights management (DRM)
  - Selling music and movies while preventing piracy
  - Making sure that software is licensed

TPM criticism

(more details at https://www.cl.cam.ac.uk/~rja14/tcpa-faq.html)

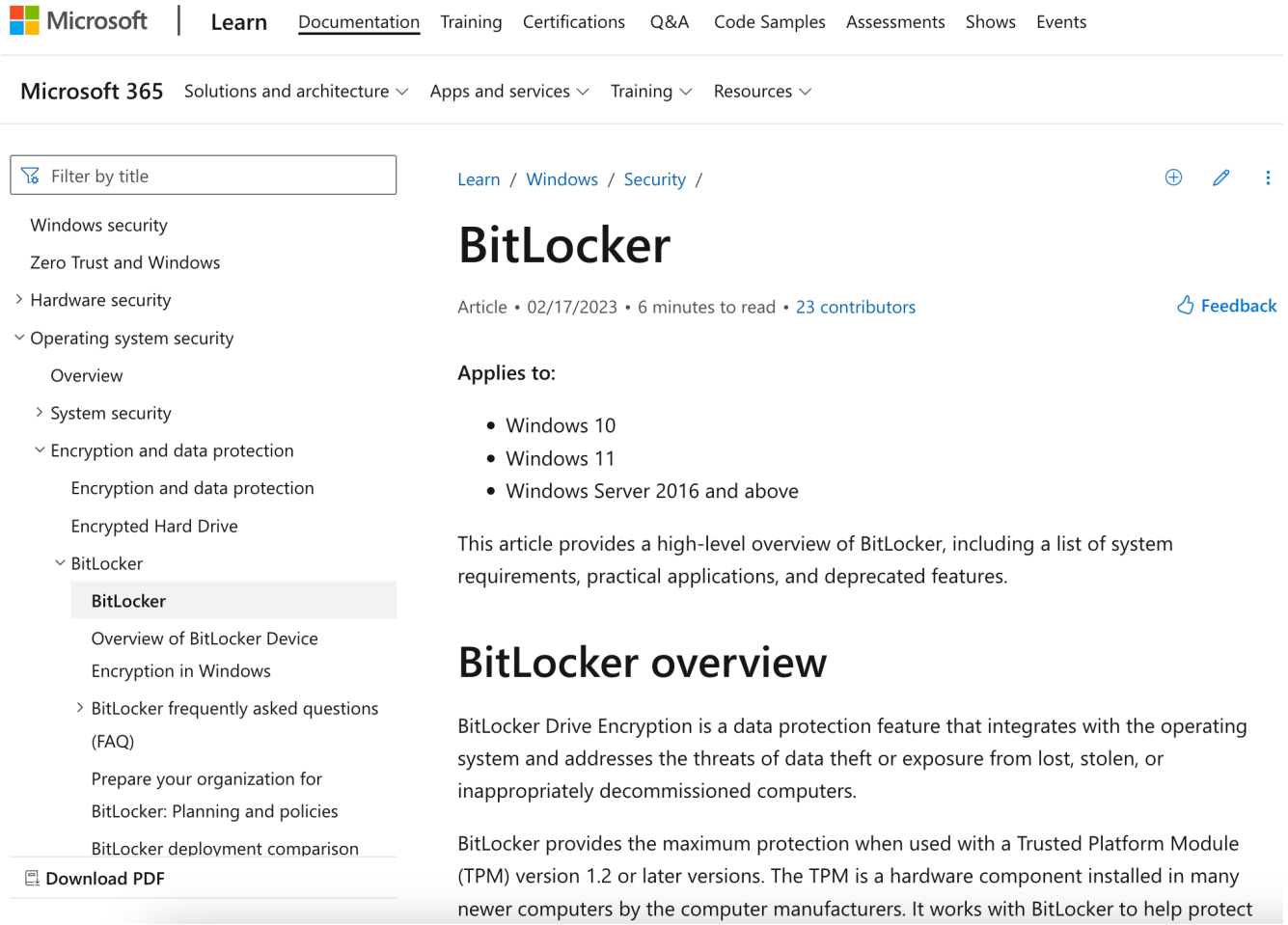
- Court can order defamatory paper to be censored
  - Software company that wrote word processor has to comply
  - Or, more broadly, censorship by governments or corporations can be made easier
- Promotes vendor lock-in (e.g., Word files might be encrypted with keys known only to Microsoft)
- Locking competitors out of application file formats is bad for innovation
- Even though you can turn TPM off, strong incentive not to due to apps not working well
  - Like moving from Windows/Macs to Linux: more freedom but less choice
- Concentrates power in the owners of the trusted computing infrastructure
  - Single corporation with a OS monopoly may decide to shut down an entire country

Arguments and counterarguments

- Against TPMs: TPMs constrain what users do with their own PCs, so that software and content vendors can make more money. TPMs promotes censorship
- Pro TPMs: there are some situations where constraining user control is useful, e.g., modifying odometer before selling car

Actual TPM applications

- Protects laptop contents in case of theft
- Full disk encryption
- Decryption keys are only provided upon correct boot sequence
- ChromeOS also uses TPMs, namely to prevent firmware rollback



Trusted Execution Environment

Trusted execution environments

- TPMs turned out to be fairly inflexible
  - Secure boot attests to a very large software stack

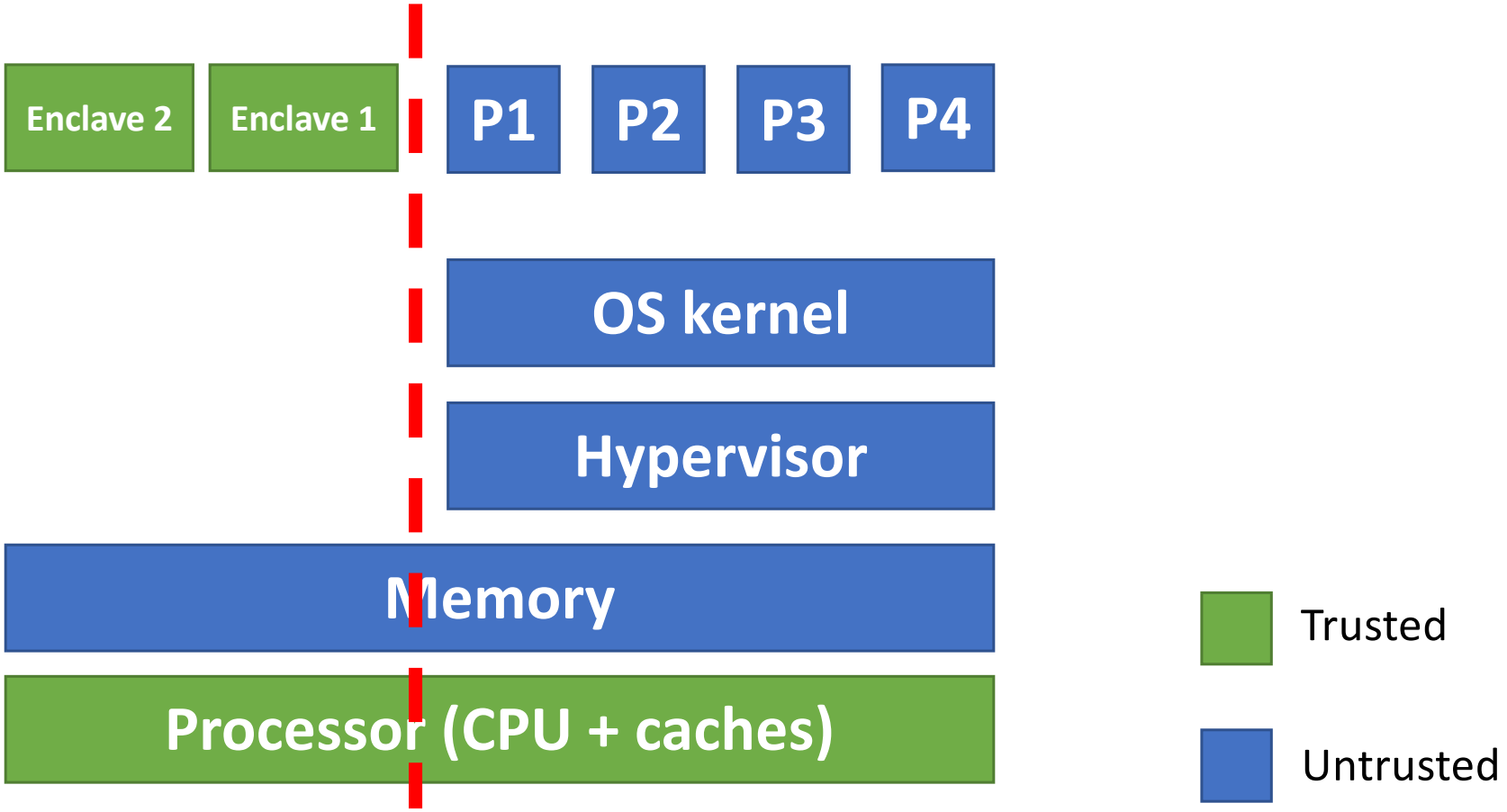
Trusted Execution Environments

- Allow for strong confidentiality and integrity of a generic piece of code, deployed in a commodity machine
  - Focus on confidentiality and integrity, not availabiliity
- Same strong threat model
  - Do not trust the owner / user / operator of the machine
  - Handles a compromised OS kernel (e.g., with malware)
- Supported by several processors
  - Intel SGX→ code runs in an "**SGX enclave**"
  - ARM TrustZone
  - AMD SEV

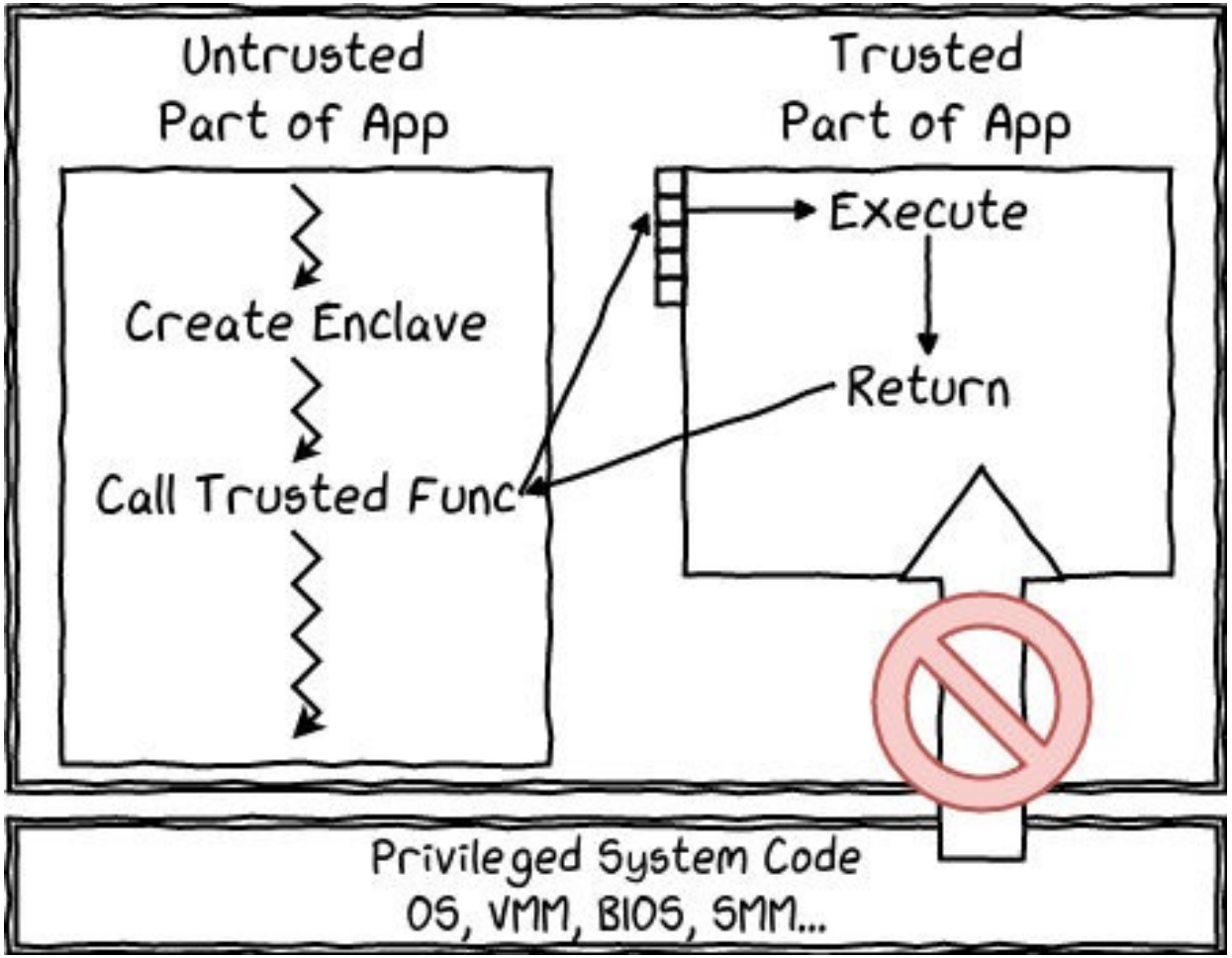
Outline

- What are SGX enclaves?
- What are they good for?
- How are they supported?
- Vulnerabilities
- Alternatives + Next generation: Intel TDX

Enclave abstraction



10,000 feet view of enclave invocation



Source: https://blog.quarkslab.com/overview-of-intel-sgx-part-1-sgx-internals.html

Security mechanisms of enclaves

- **Isolation** – its own data and code cannot be seen (confidentiality) or modified (integrity) by any other software on the same machine, including hypervisor, OS, or other enclaves
- **Attestation** – means to prove to a third party the code that is running
- **Sealing** – cryptographically protect data before it leaves enclave
- (While providing a fairly standard programming model, including multithreading and access to application's memory.)

Remote attestation

- Means by which a process, running on a remote machine, determines the TCB of the enclave:
  - Enclave produces a digest of its code, initial data
  - Also reports its CPU firmware and hardware
  - Simple protocol allows for simultaneously sending this digest, signed by the private key of the machine where enclave runs and establishing an authenticated channel between enclave and remote attesting party
  - This private key of the machine is, in turn, vouched by Intel

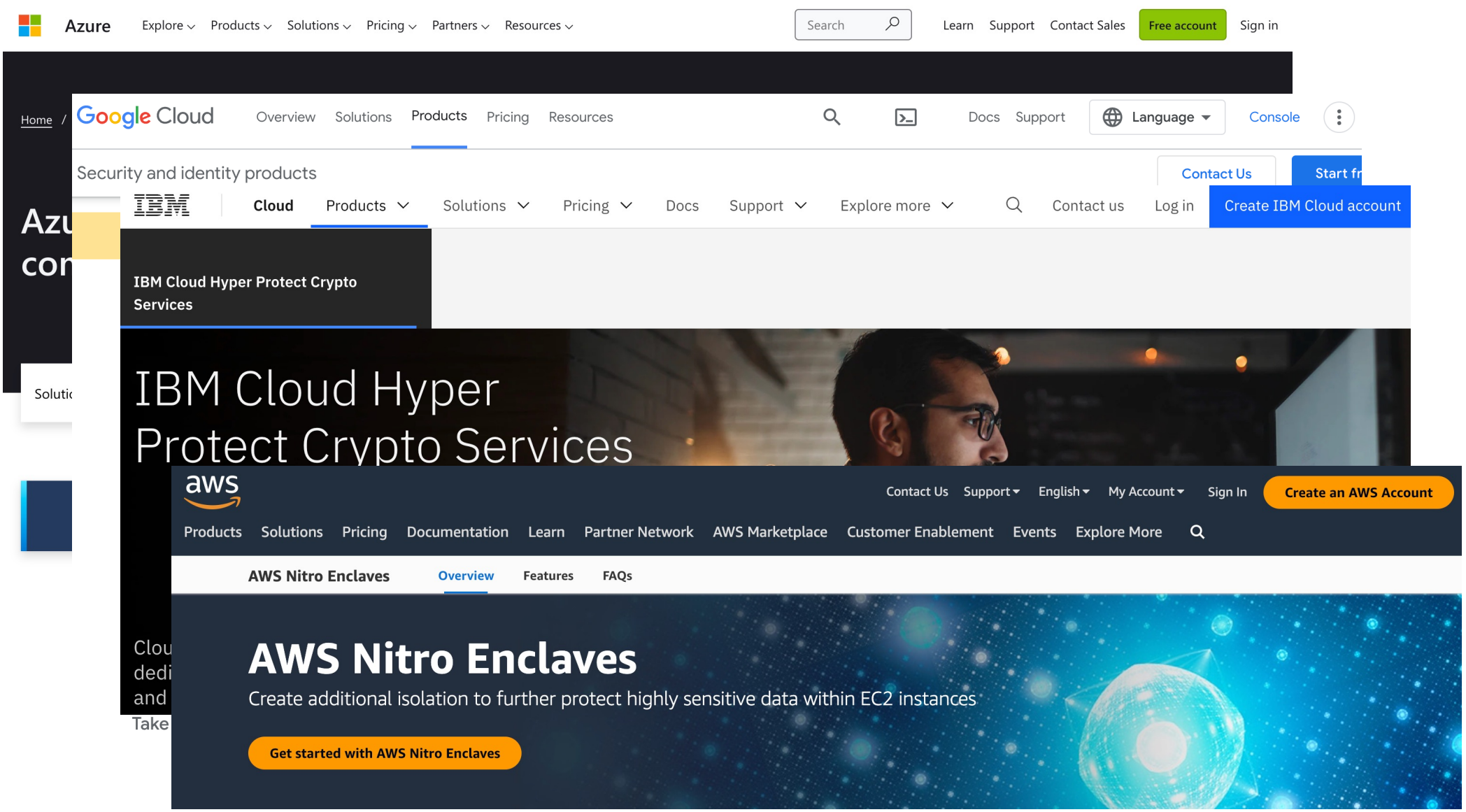
  (details are a bit more complex, involve contacting Intel's attestation service)
  - Also supported by TPM

Data sealing

- What if the enclave wants to have state persisted across calls?
- Sealing – enclave data can be encrypted before being stored persistently…
- … in such a way that it can only be decrypted by the same enclave code running on the same platform
- however, it is vulnerable to a replay attack

Lecture outline

- What are SGX enclaves?
- What are they good for?
- How are they supported?
- Vulnerabilities
- Alternatives + Next generation: intel TDX


Azure
Explore
Products
Solutions
Pricing
Partners
Resources
Search
Learn
Support
Contact Sales
Free account
Sign in
Home
Google Cloud
Overview
Solutions
Products
Pricing
Resources
Docs
Support
Language
Console
Security and identity products
Contact Us
Start fr
IBM
Cloud
Products
Solutions
Pricing
Docs
Support
Explore more
Contact us
Log in
Create IBM Cloud account
IBM Cloud Hyper Protect Crypto Services
IBM Cloud Hyper Protect Crypto Services
aws
Contact Us
Support
English
My Account
Sign In
Create an AWS Account
Products
Solutions
Pricing
Documentation
Learn
Partner Network
AWS Marketplace
Customer Enablement
Events
Explore More
AWS Nitro Enclaves
Overview
Features
FAQs
AWS Nitro Enclaves
Create additional isolation to further protect highly sensitive data within EC2 instances
Get started with AWS Nitro Enclaves

Technology preview: Private contact discovery for Signal

moxie0 on 26 Sep 2017

At Signal, we've been thinking about the difficulty of private contact discovery for a long time. We've been working on strategies to improve our current design, and today we've published a new private contact discovery service.

Using this service, Signal clients will be able to efficiently and scalably determine whether the contacts in their address book are Signal users **without revealing the contacts in their address book to the Signal service**.

SGX contact discovery

Private contact discovery using SGX is fairly simple at a high level:

1. Run a contact discovery service in a secure SGX enclave.
2. Clients that wish to perform contact discovery negotiate a secure connection over the network all the way through the remote OS to the enclave.
3. Clients perform remote attestation to ensure that the code which is running in the enclave is the same as the expected published open source code.
4. Clients transmit the encrypted identifiers from their address book to the enclave.
5. The enclave looks up a client's contacts in the set of all registered users and encrypts the results back to the client.

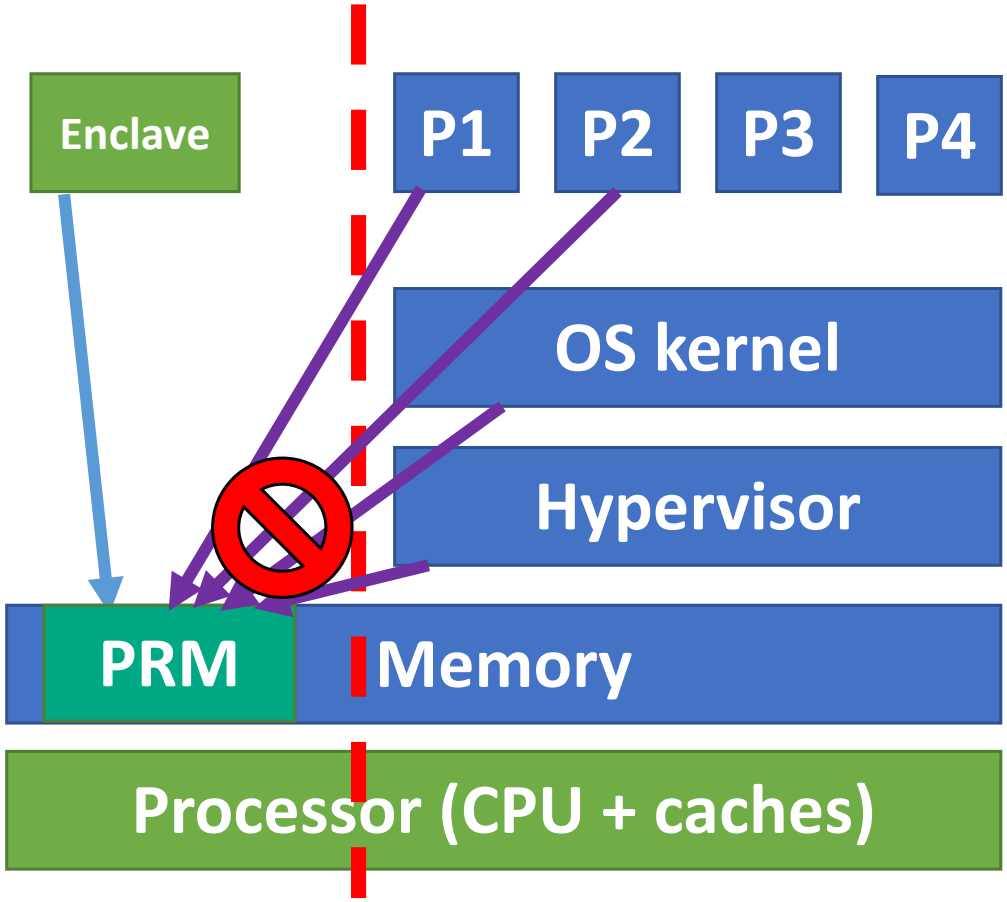
Lecture outline

- What are SGX enclaves?
- What are they good for?
- How are they supported?
- Vulnerabilities
- Alternatives + Next generation: intel TDX

Implementation of enclaves in SGX

- Goal: isolate enclave code and data from:
  - OS
  - Hypervisor
  - Regular processes
  - Devices attached to system bus
- TCB: enclave code + everything inside CPU chip
- Does not trust DRAM memory

Goal: isolation of enclave code+data



PRM:
Processor
Reserved
Memory

Trusted

Untrusted

Lecture outline

- What are SGX enclaves?
- What are they good for?
- How are they supported?
- Vulnerabilities
- Alternatives + Next generation: intel TDX

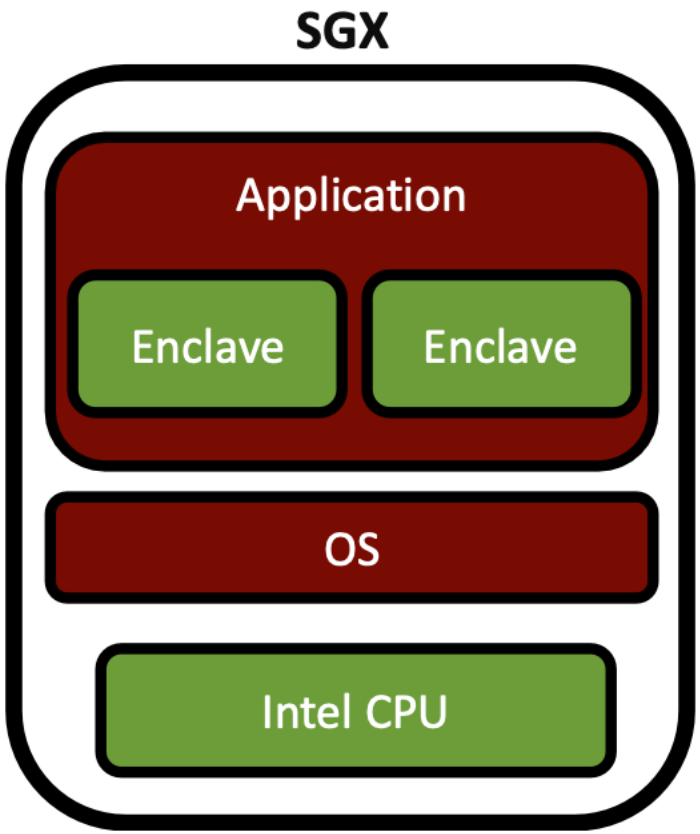
Attacks on Intel SGX

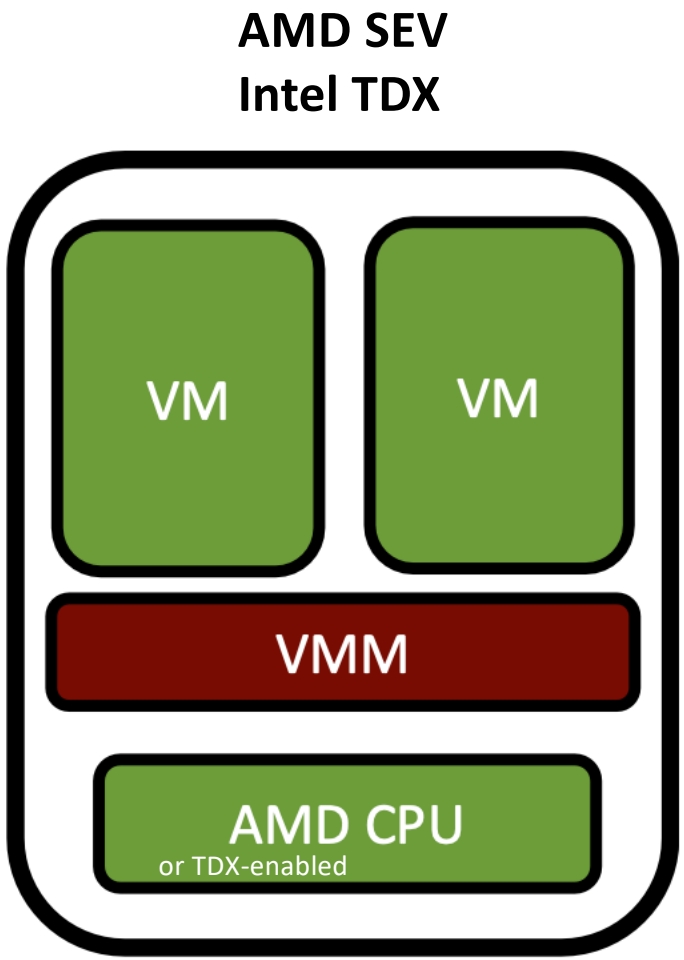
- Most of them are side-channel attacks
- Recently, lots of attention in the security community to Spectre and Meltdown → initially in “normal” code, but researchers found these to be applicable to SGX enclaves (with some adaptations)
- Exploit speculative execution
  - Processors execute a few instructions ahead, and rollback if needed
  - However, rollback does not erase subtle side effects such as cache modifications

Lecture outline

- What are SGX enclaves?
- What are they good for?
- How are they supported?
- Vulnerabilities
- Alternatives + Next generation: intel TDX

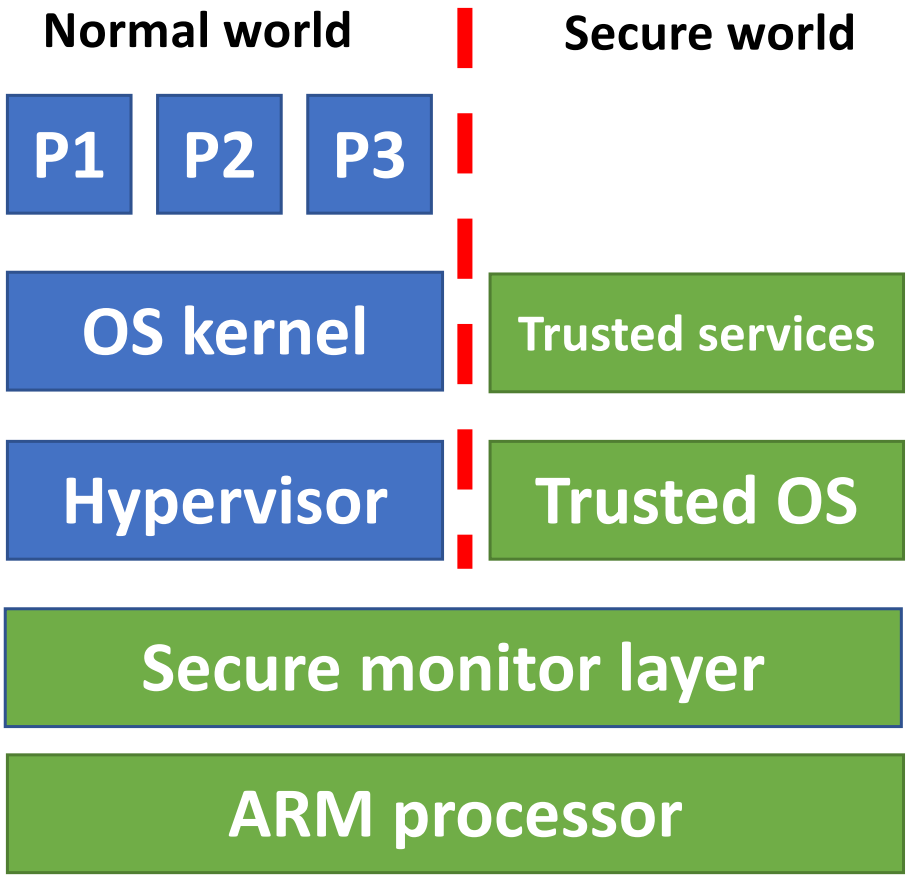
TDX + AMD SEV





Adapted from: M. Yan.
MIT 6.888 course slides

ARM TrustZone



Concluding remarks

- Recall definition from first lecture:
- A system is dependable if it is able to work (i.e., deliver its service) in a way that is justifiably trusted, namely avoiding service failures that are either more frequent or severe than acceptable.
- We saw how the Byzantine model allows for building trust even with a subset of malicious participants
  - Key for distributed trust in blockchain systems
- Hardware can increase trust in some of the system components
  - Namely smartcards and TEEs
- Throughout these topics, you learned several important concepts
- And hopefully had some fun in the process!

Bibliography

[1] Ross Anderson. Security Engineering. 2nd edition. Section 16.6

Available online at

http://www.cl.cam.ac.uk/~rja14/Papers/SEv2-c16.pdf

[2] Ross Anderson. Security Engineering. 3rd edition. Section 19.4

Available online at

https://www.cl.cam.ac.uk/~rja14/Papers/SEv3-ch19-7sep.pdf

Acknowledgments

- Original slides from Prof. Ricardo Chaves @ IST, AISS course, 2013.
- Adapted for SEC by Miguel Matos, Paolo Romano, Rodrigo Rodrigues.

- Previous acks from AISS: Jean Rodrigues @ STEVENS Institute of Technology
- Berk USTUNDAG @ Istanbul Technical University
- Marc Witteman @Riscure.com
- Dr. Hakim Fourar-Laidi @ Prince Sultan University
- Joshua Lawrence @ Florida State University
- Erik Poll @ University of Nijmegen
- Robert Sloan @ University of Illinois at Chicago