

An introduction to confidential computing

Fabien Petitcolas

Smals Research

Agenda

General overview

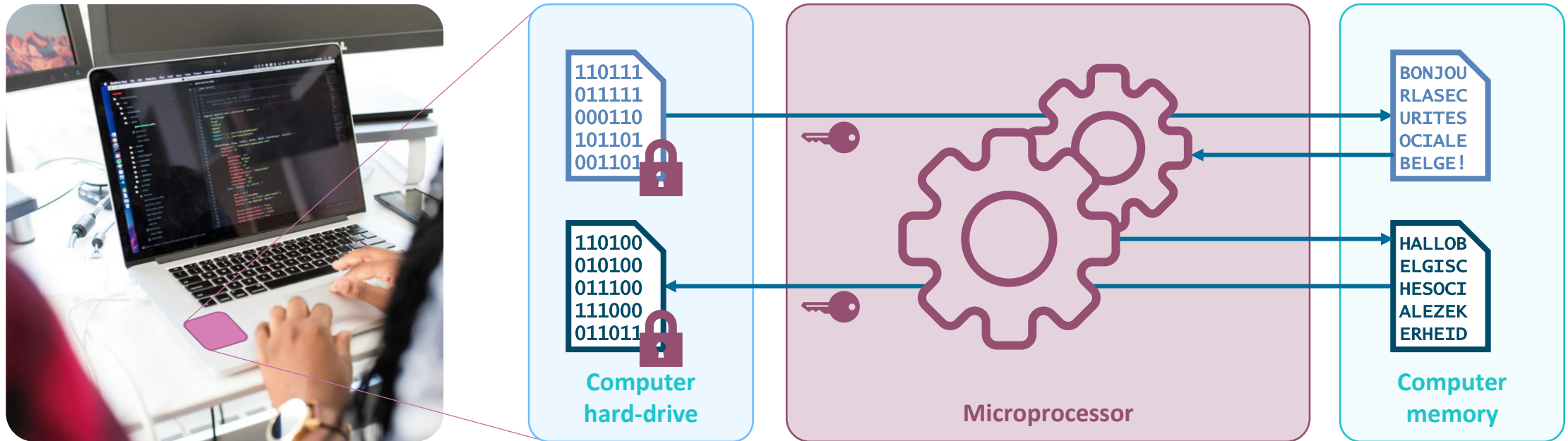
- Secure remote computation
- Homomorphic encryption
- Secure multi-party computation
- Trusted execution environments (TEE)
- Comparison of maturity

Market offer for TEE

- Secured processors (AMD, Intel)
- Computing infrastructures (AWS, Azure)

Conclusions and recommendations

Traditional computation on encrypted data



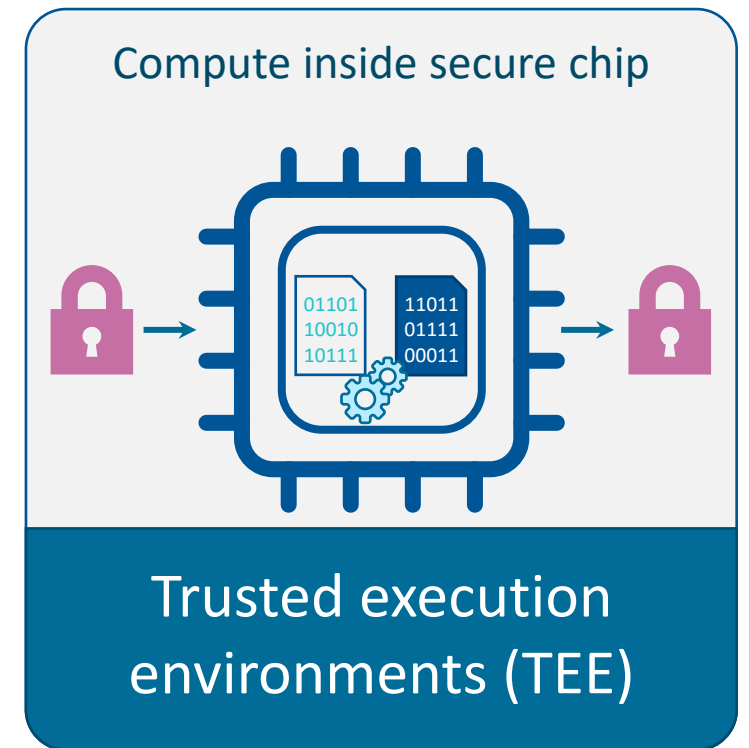
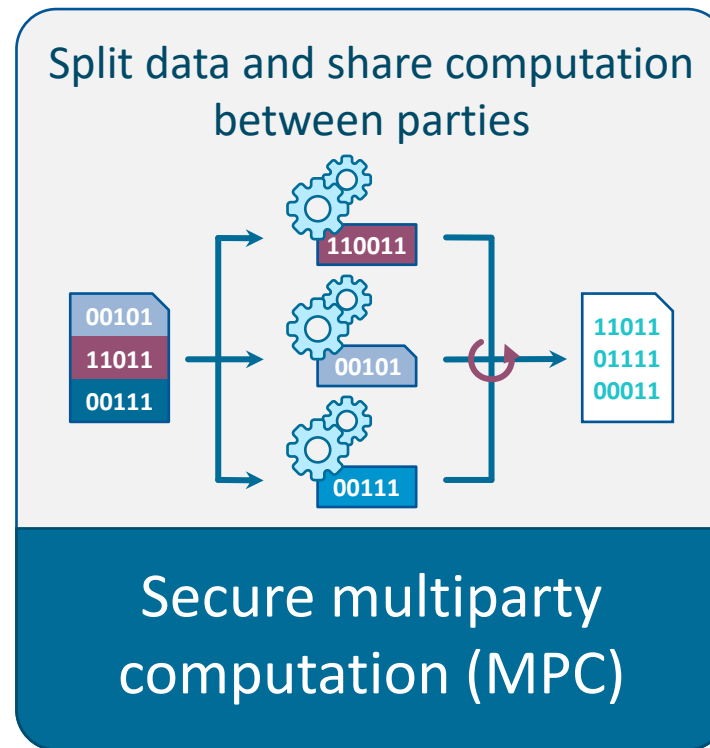
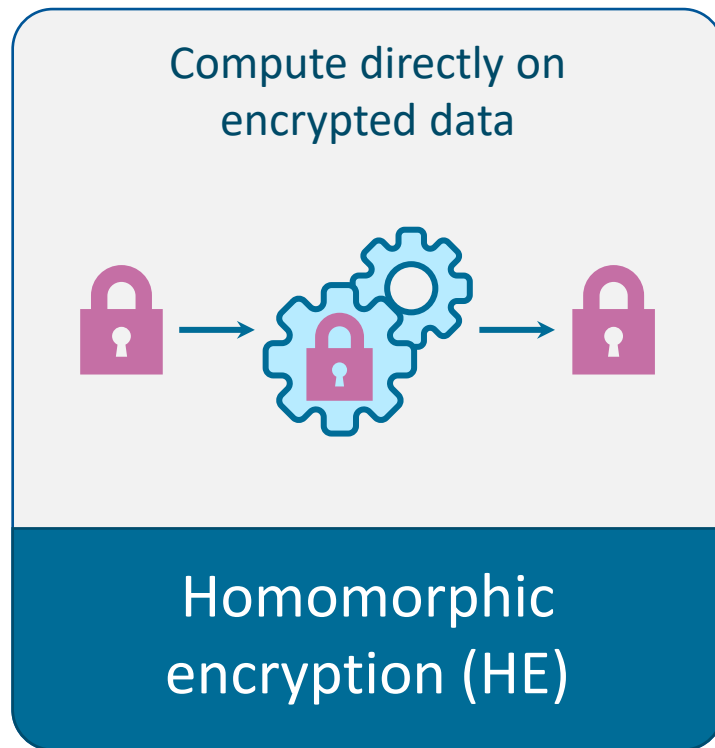
Basic remote computation

Concern: confidentiality, privacy (GDPR), sovereignty, etc. of data "in use"?



Main techniques for trusted remote computation

Aim: move the infrastructure provider outside of the trust boundary



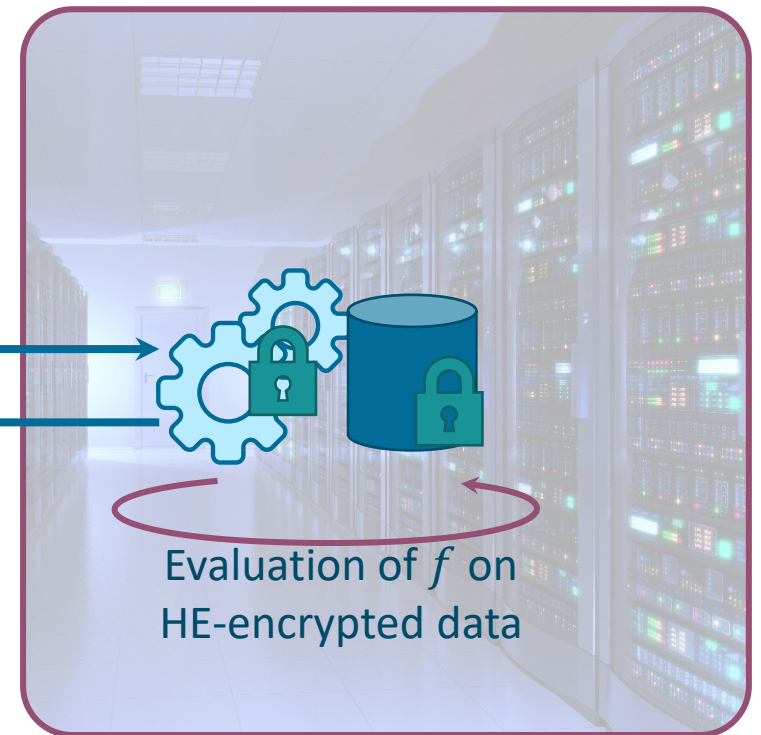
Homomorphic encryption (HE)

Homomorphic encryption: schematic overview

Client



Shared computing infrastructure



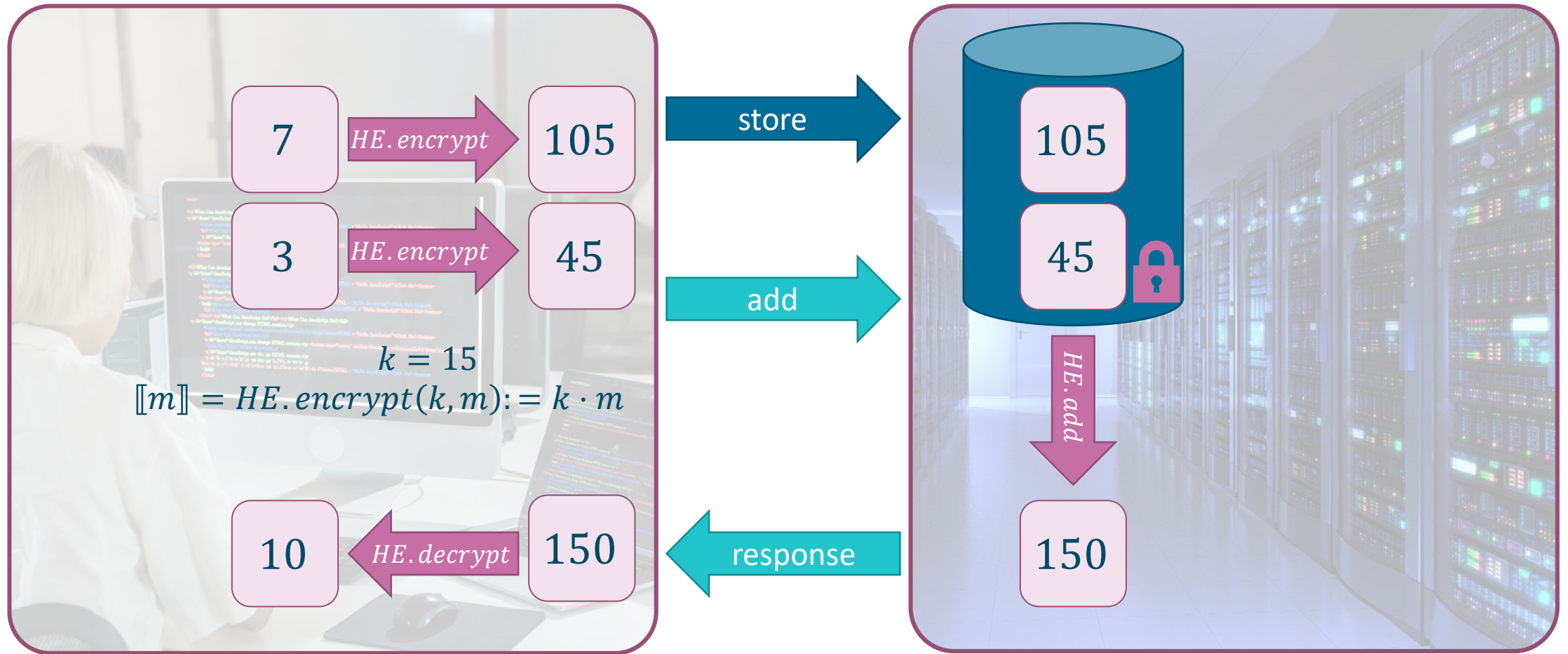
HE-encrypted data
function to evaluate (f)

HE-encrypted results

Depending on allowed complexity of f :

- Partial homomorphic encryption (PHE)
- Somewhat homomorphic encryption (SWHE)
- Fully homomorphic encryption (FHE)

Homomorphic encryption: trivial example



Advantages and limits of homomorphic encryption

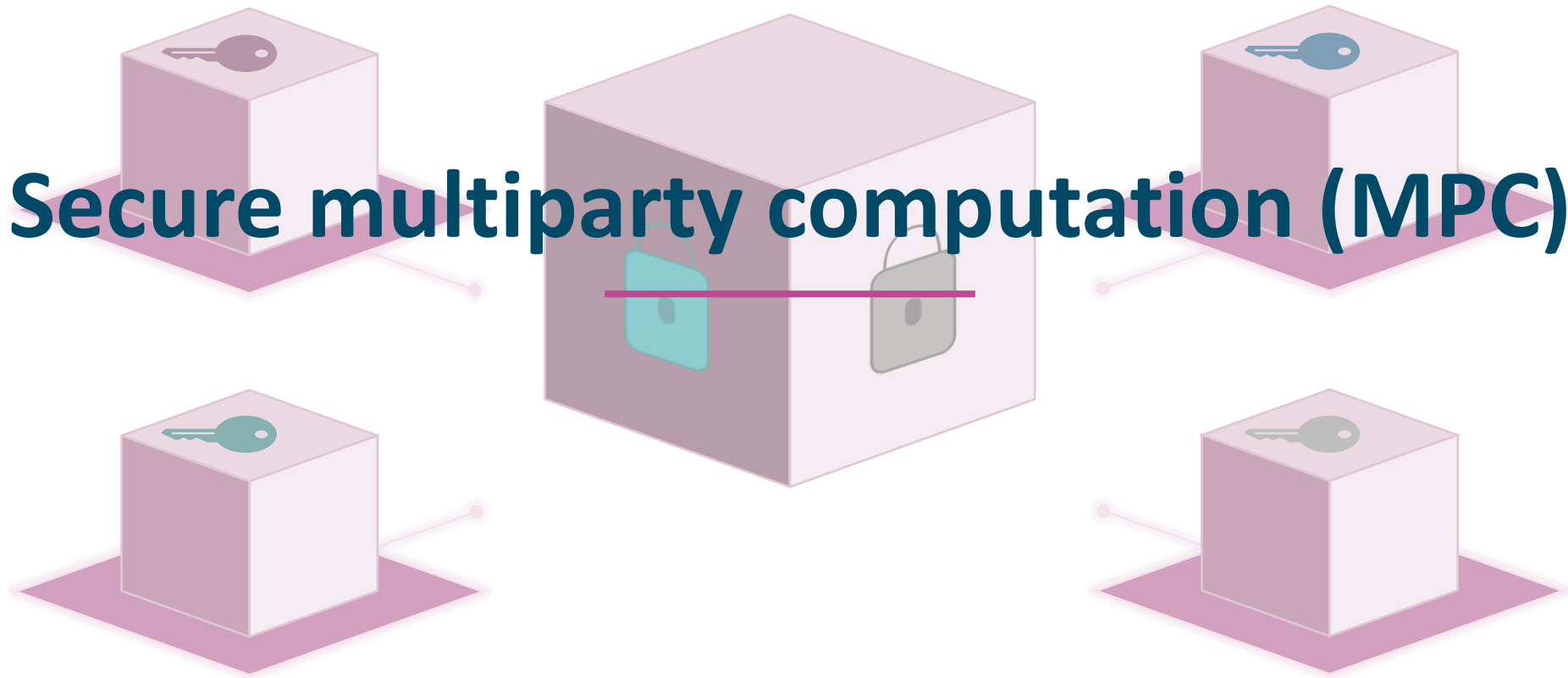
Pros

- Security based on strong mathematical evidence under well defined assumptions
- Does not need special hardware
- Some schemes robust to post-quantum attacks
- Active research area

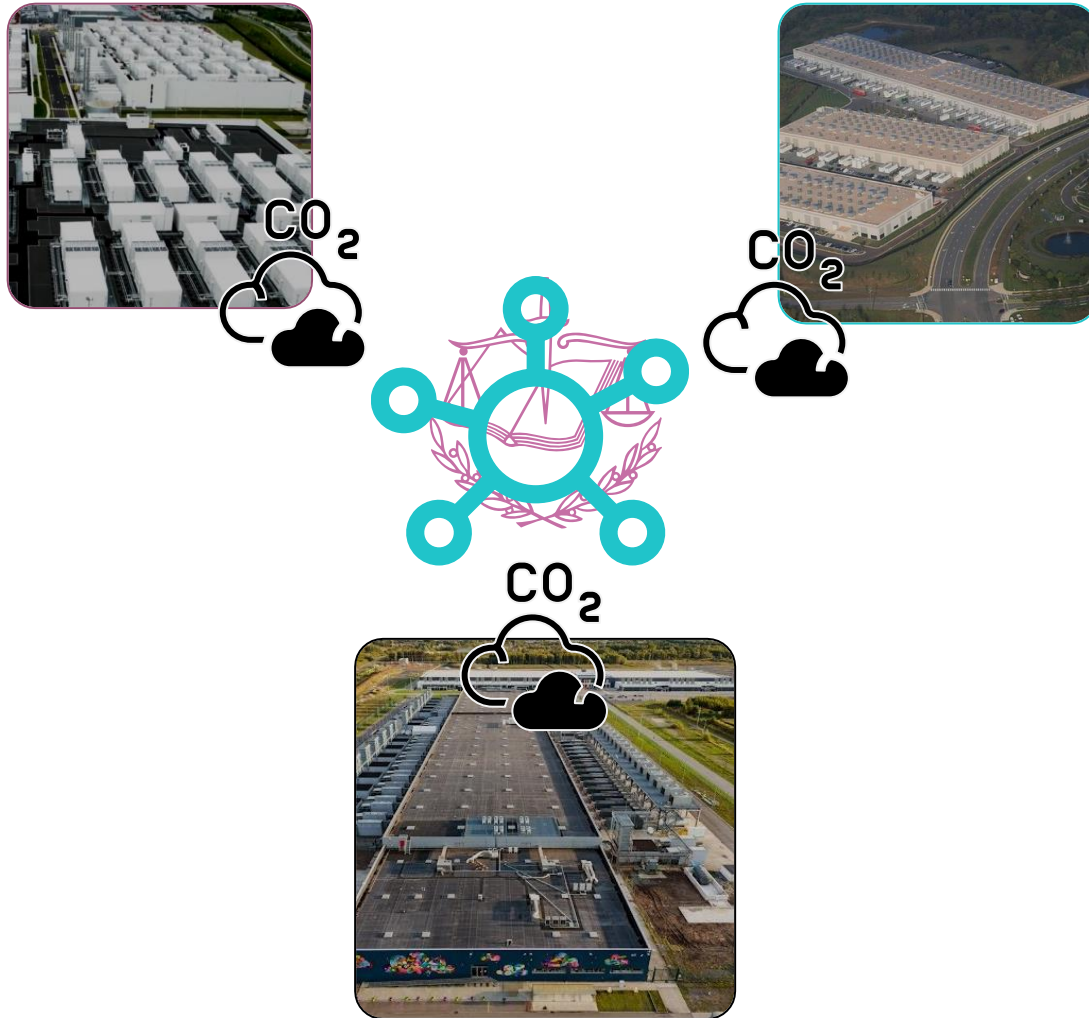
Cons

- Cryptography expert required to build protocol based on HE
- Computation not guaranteed
 - e.g., cannot check if $\llbracket m_1 \rrbracket \oplus \llbracket m_2 \rrbracket$ or $\llbracket m_1 \rrbracket \ominus \llbracket m_2 \rrbracket$ was computed
- High overhead:
 - Engineering cost: complex parametrisation, substantial changes required in application
 - Storage and bandwidth: large message expansion
 - Relatively low performance

Secure multiparty computation (MPC)



MPC problem example



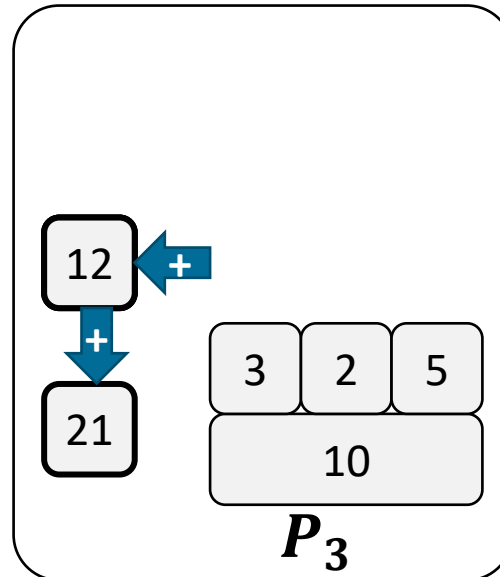
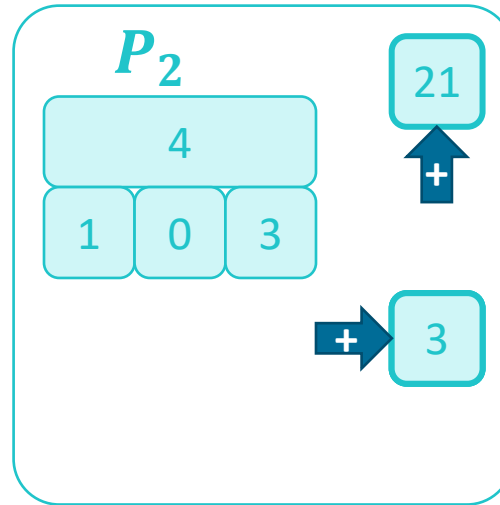
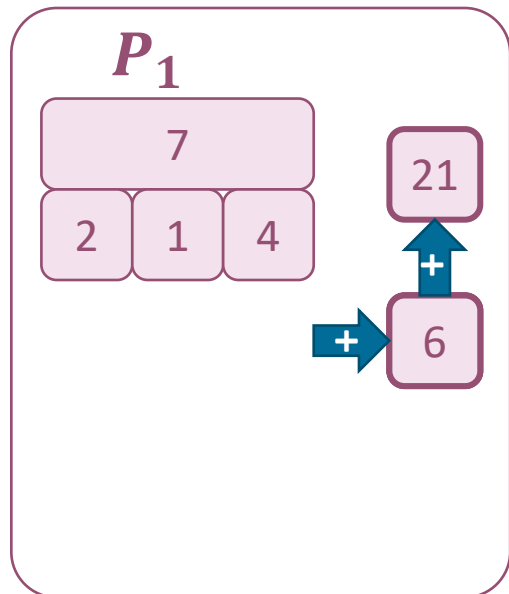
Three computing companies want to know which one of them has the lowest carbon footprint without revealing their respective values

- Solutions:
 - Use a trusted party (hard to find)
 - Use multi-party computation (MPC): it enables mutually distrusting parties to compute an arbitrary function on their inputs.

MPC problem and α solution

- **Problem:** n parties P_1, \dots, P_n each have a secret input x_i and want to evaluate a function f in such a way that:
 - only the value $z = f(x_1, \dots, x_n)$ is learned
 - and nothing else is learned about x_1, \dots, x_n .
- **MPC solution example:**
 - Use arithmetic circuits to break down f into a composition of addition (+) and multiplications (\times)
 - Each party follows a specific protocol:
 - Split input data into pieces and share pieces with other parties
 - Apply additions and multiplications on data shares locally (or with minimal interaction between parties)
 - Recombine partial results to get final result

MPC in action: simple addition example

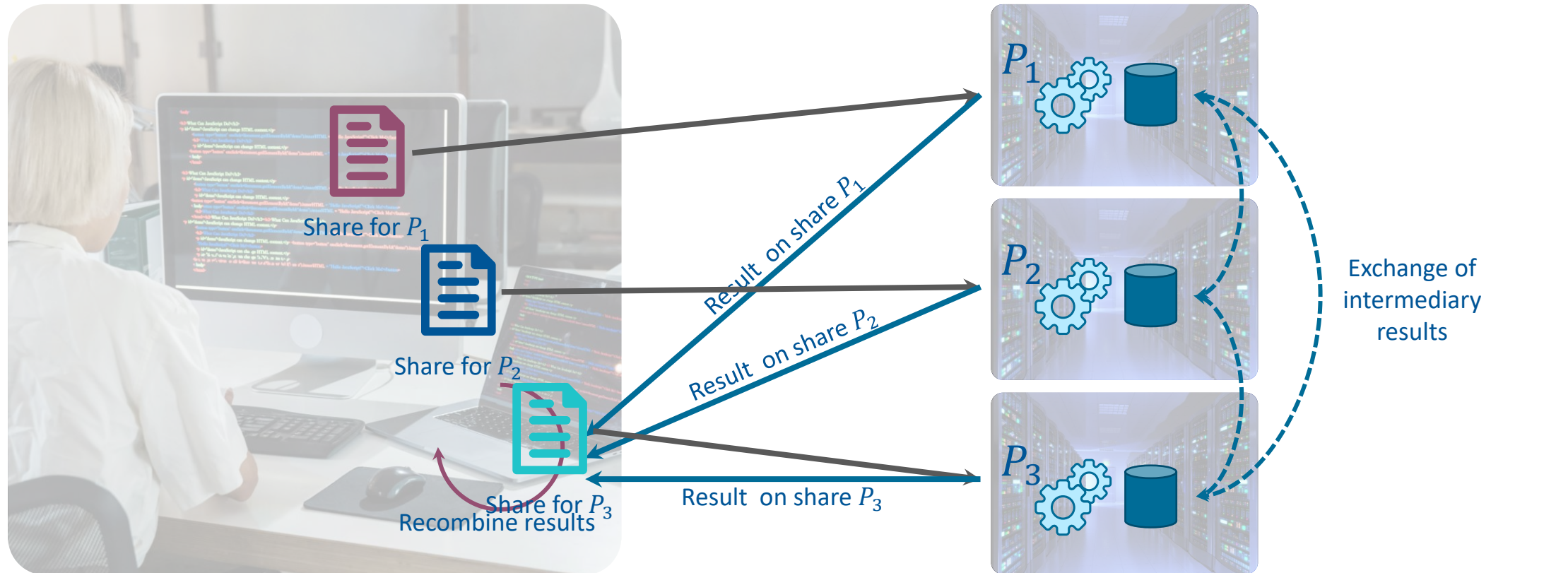


1. Split input:
 $x \rightarrow (x_1, x_2, x_3)$ s.t. $x_1 + x_2 + x_3 = x$
2. Share input
3. Local addition
4. Share partial results
5. Local addition of final result

MPC deployment example

Client

Shared computing infrastructures



Advantages and limits of MPC

Pros

- Cryptographic-based security
- Does not require special hardware
- Enable collaboration between untrusting parties
- Does not require central trusted party
- Active research area

Cons

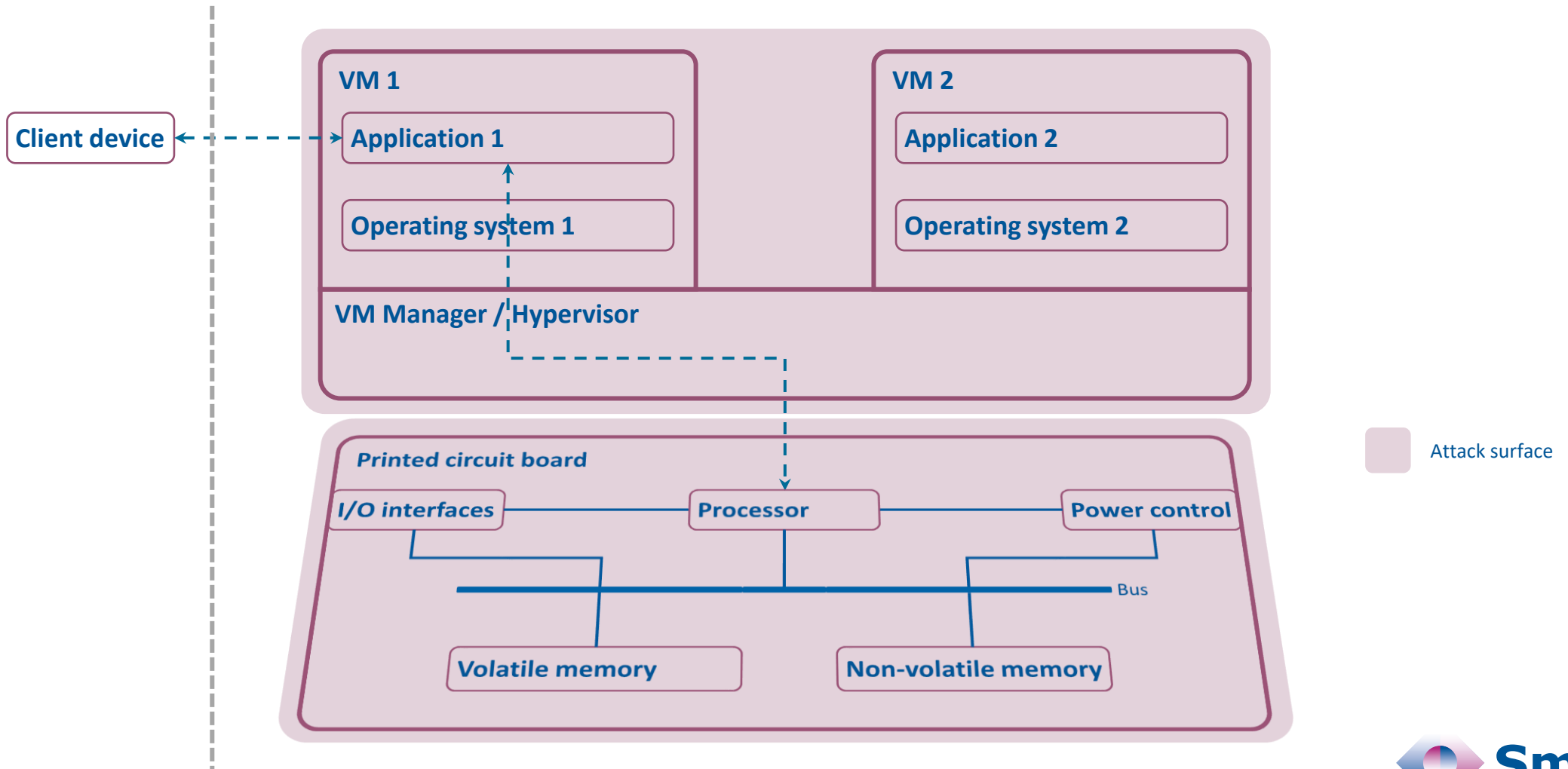
- Complexity of formally verifying protocol
- Complex setup and management
- Software rewriting required with highly specialised client-server software
- High communication cost between parties
- Small number of applications in production

Trusted execution environments (TEE)

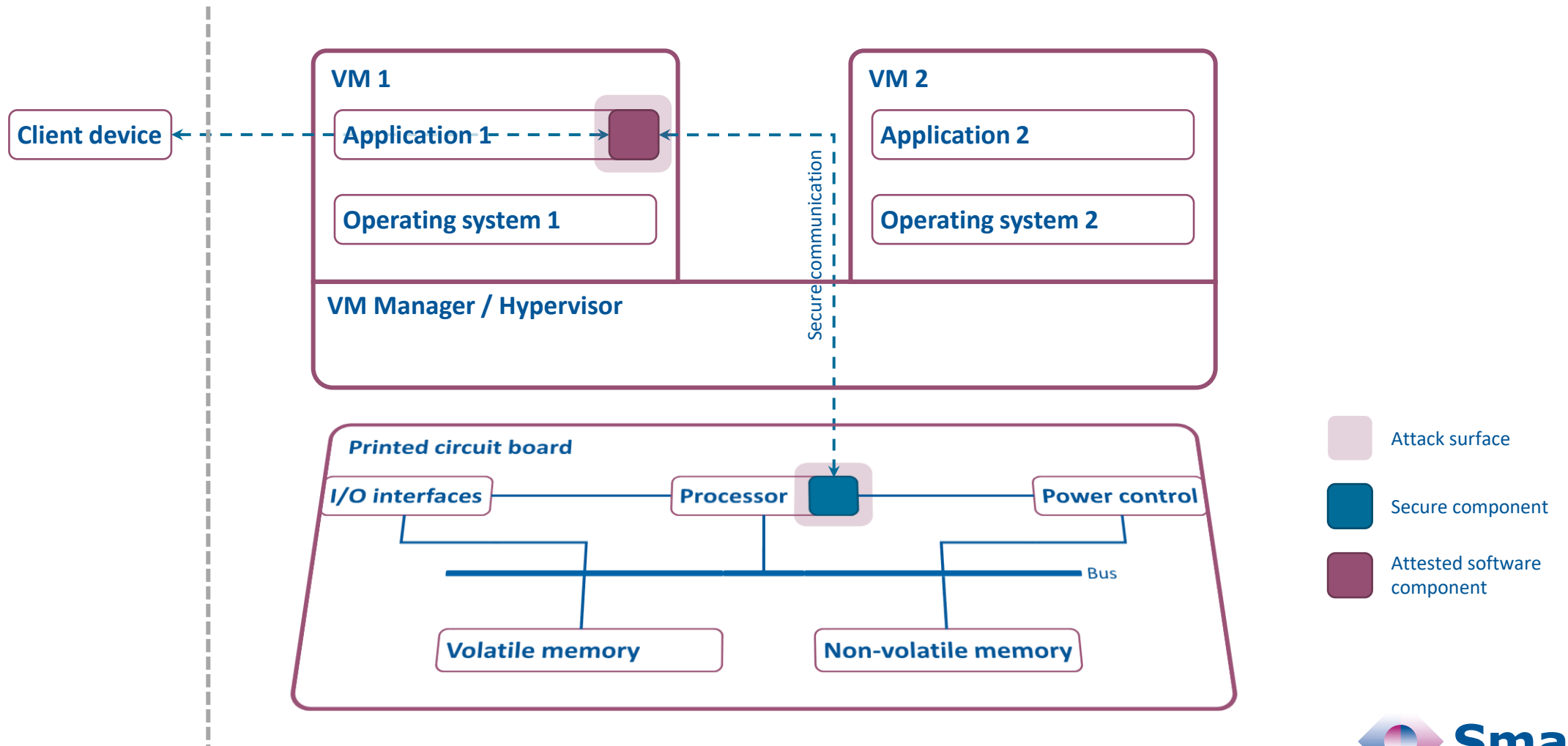
Hardware-based isolated execution

- **Technical goal:** better protect applications from each other by creating isolated environments enforced by the hardware layer
- **Rational:** a system cannot be secure if its lowest layer (the hardware) is not
- **Requirements for TEE:**
 - Hardware root of trust to hold platform secrets
 - Reserved encrypted memory for trusted code and data
 - Encryption of all input/outputs
 - Evidence of authenticity and integrity

Generic architecture



Possible generic architecture with secure hardware



Verifying integrity of the system

Secure boot

From machine power-on to known secure state:

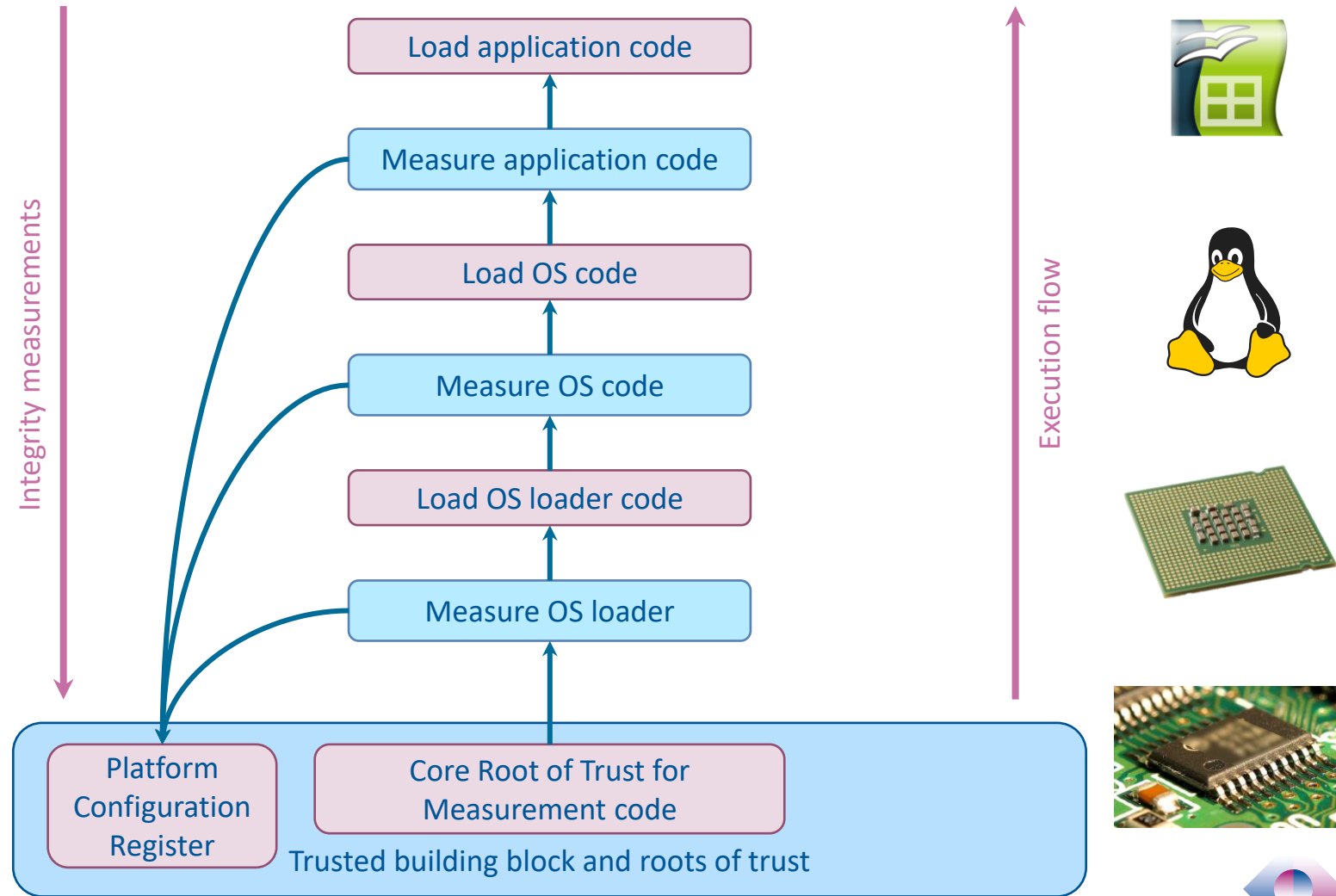
- Chain of trust from hardware to operating system software
- Each higher piece of firmware and software corresponds to what is expected by the lower component

Attestation

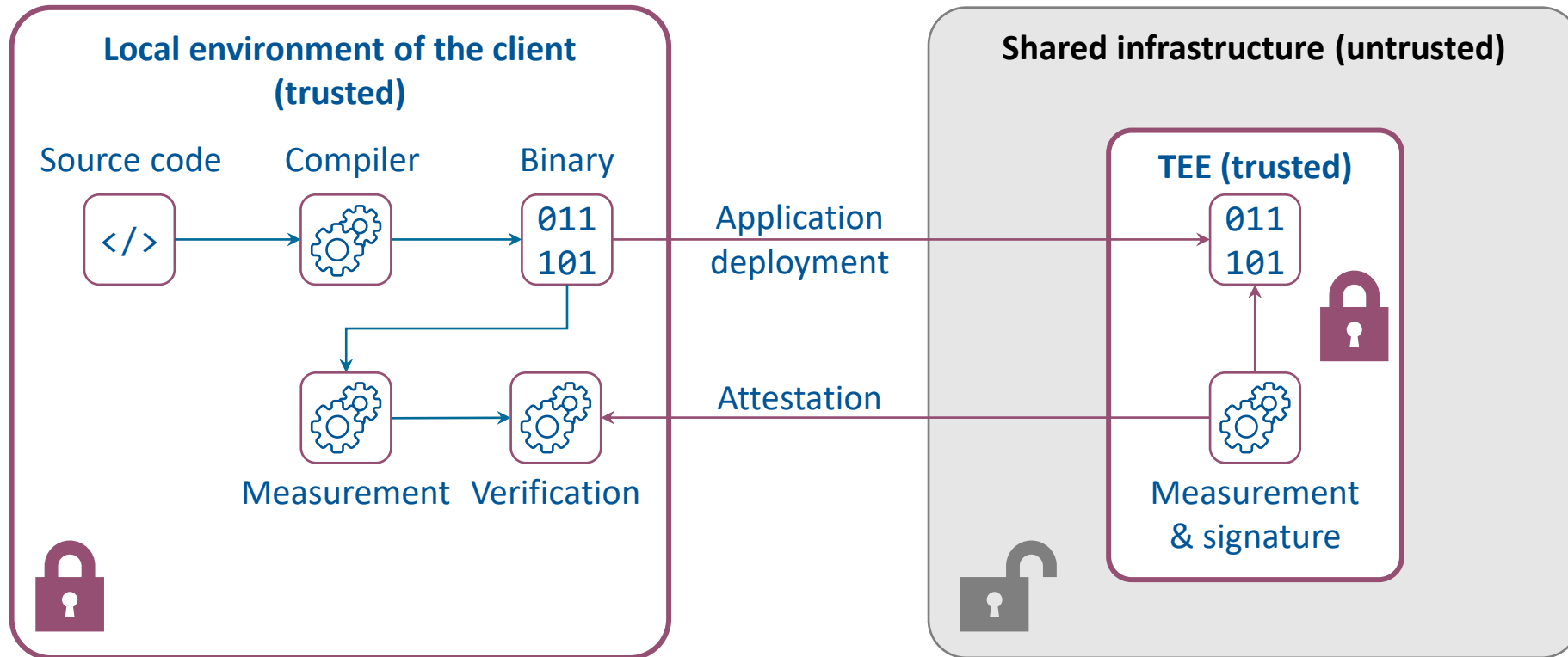
Signed evidence about remote system and state of software executing on it:

- Application runs on the expected hardware
- Executed binary is the expected application's binary
 - Should also correspond to the expected code

Secure booting sequence example



Attestation example



- Can be used to establish secret key with trusted application

- Can be used in security policies

Advantages and limits of TEE

Pros

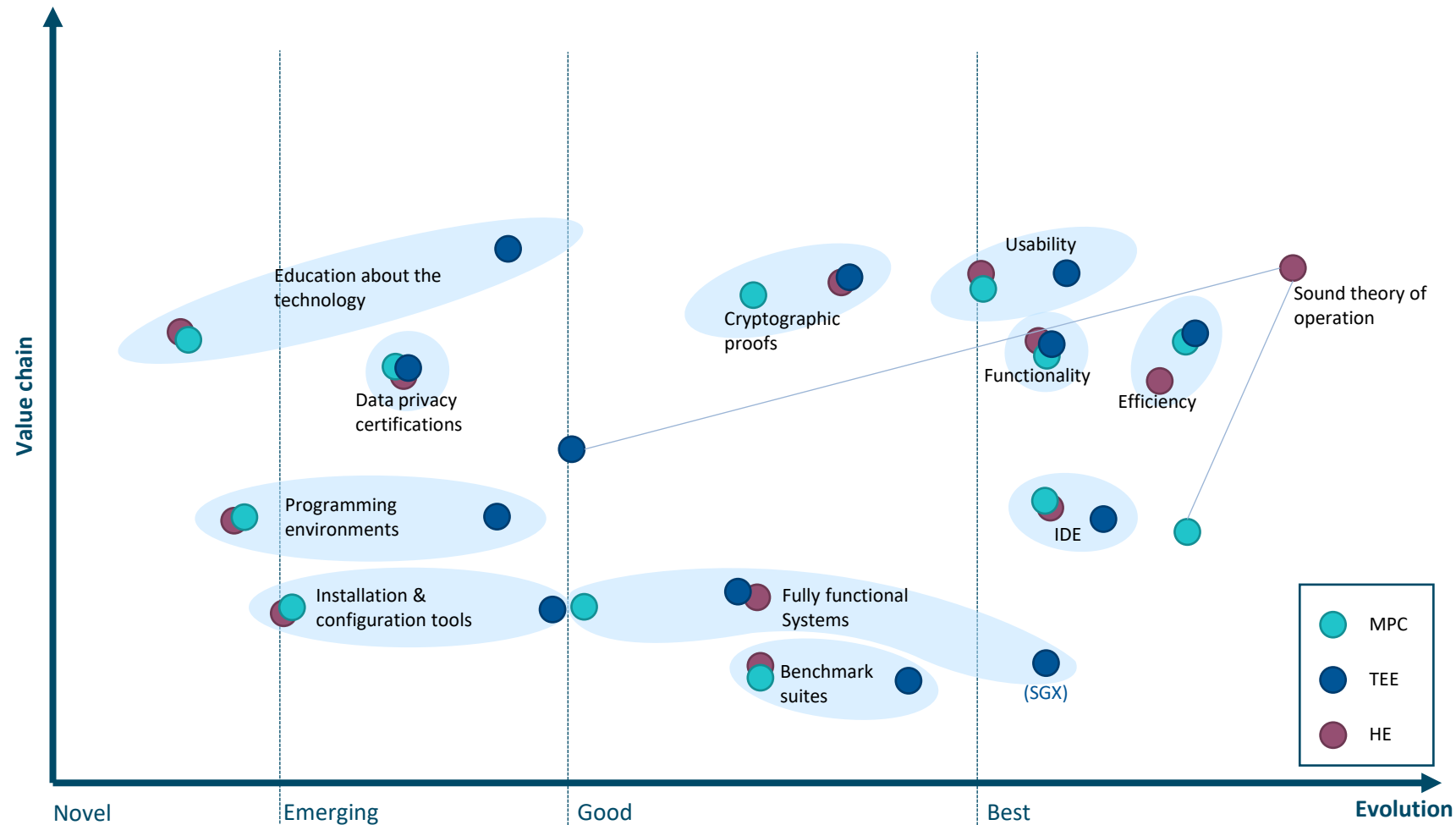
- Hardware-based security (trust the hardware manufacturer instead of the infrastructure provider)
- Available from main infrastructure providers
- Relatively simple application migration (containers, VM) compared to MPC and HE

Cons

- Requires specialised hardware
- Vulnerable to some physical attacks
- Different abstractions could lead to vendor lock-in
- Attestation may be impossible to control fully independently

HE, MPC, TEE – Which maturity?

Maturity of confidential computing technologies



Source: « UN Handbook for Privacy-Preserving Computation Techniques », 2023.

<https://unstats.un.org/bigdata/task-teams/privacy/UN%20Handbook%20for%20Privacy-Preserving%20Techniques.pdf>

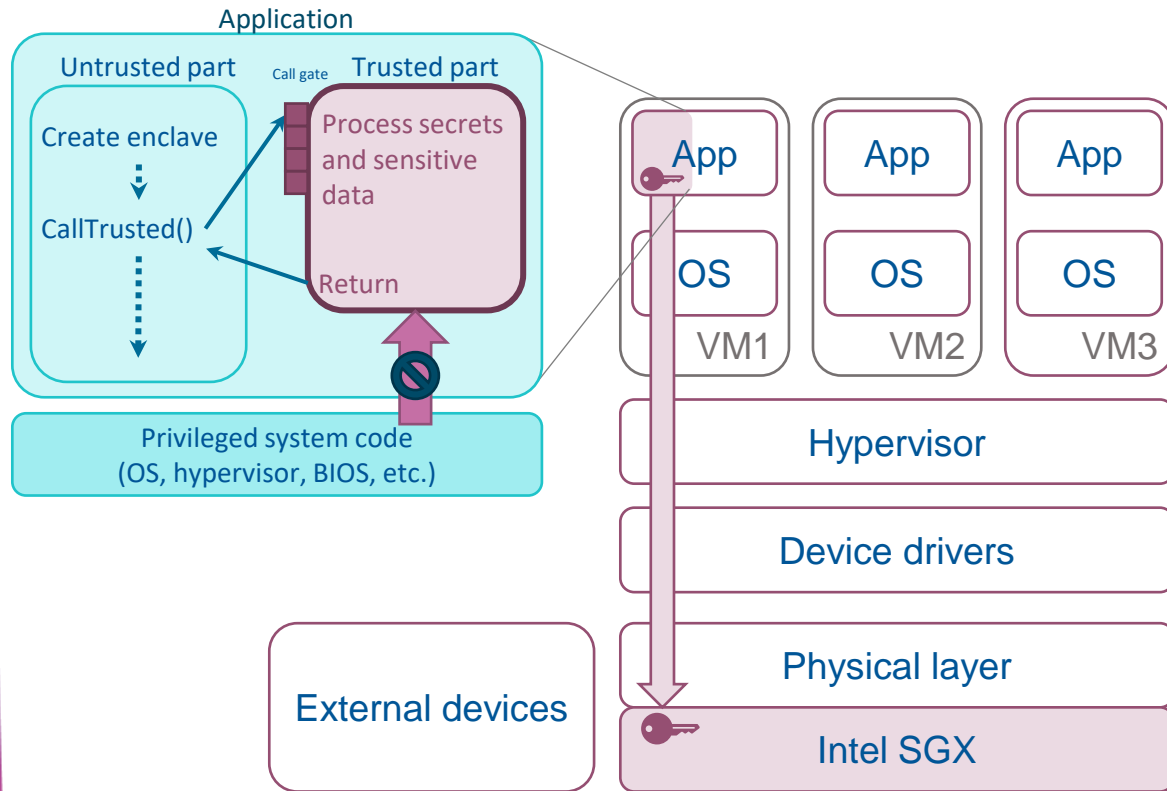
TEE-based market offer

AMD SEV-SNP, Intel SGX / TDX

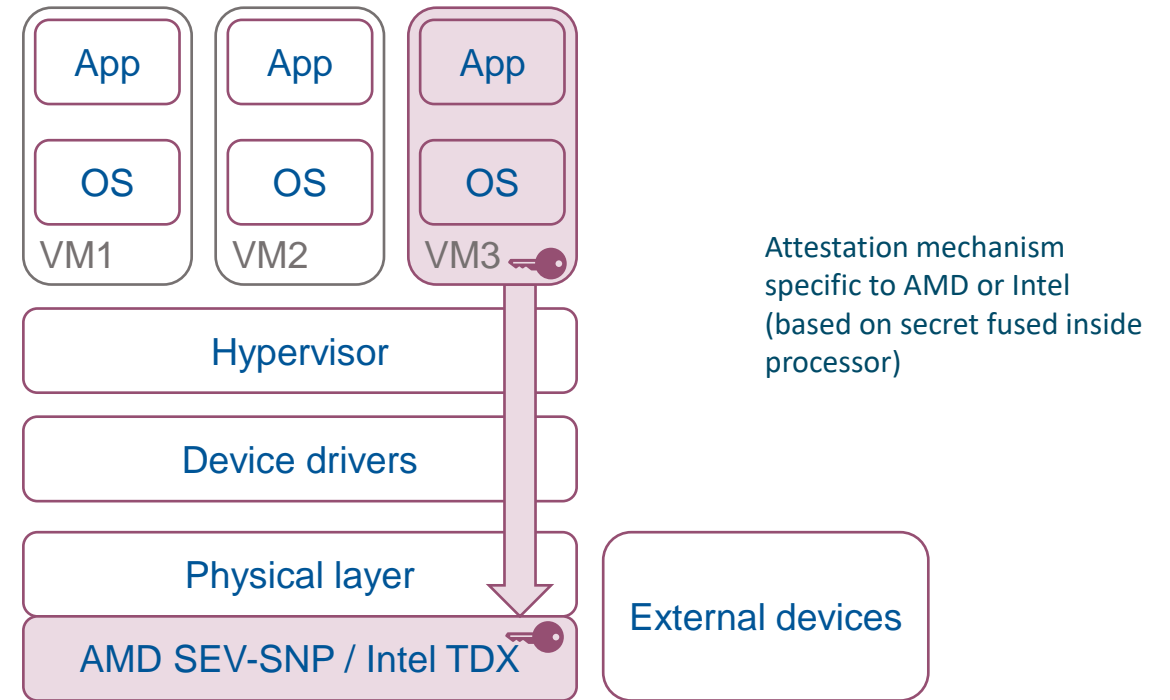
AWS Nitro and Microsoft Azure

Two main types of hardware-based isolation

Intel SGX



AMD SEV-SNP, Intel TDX



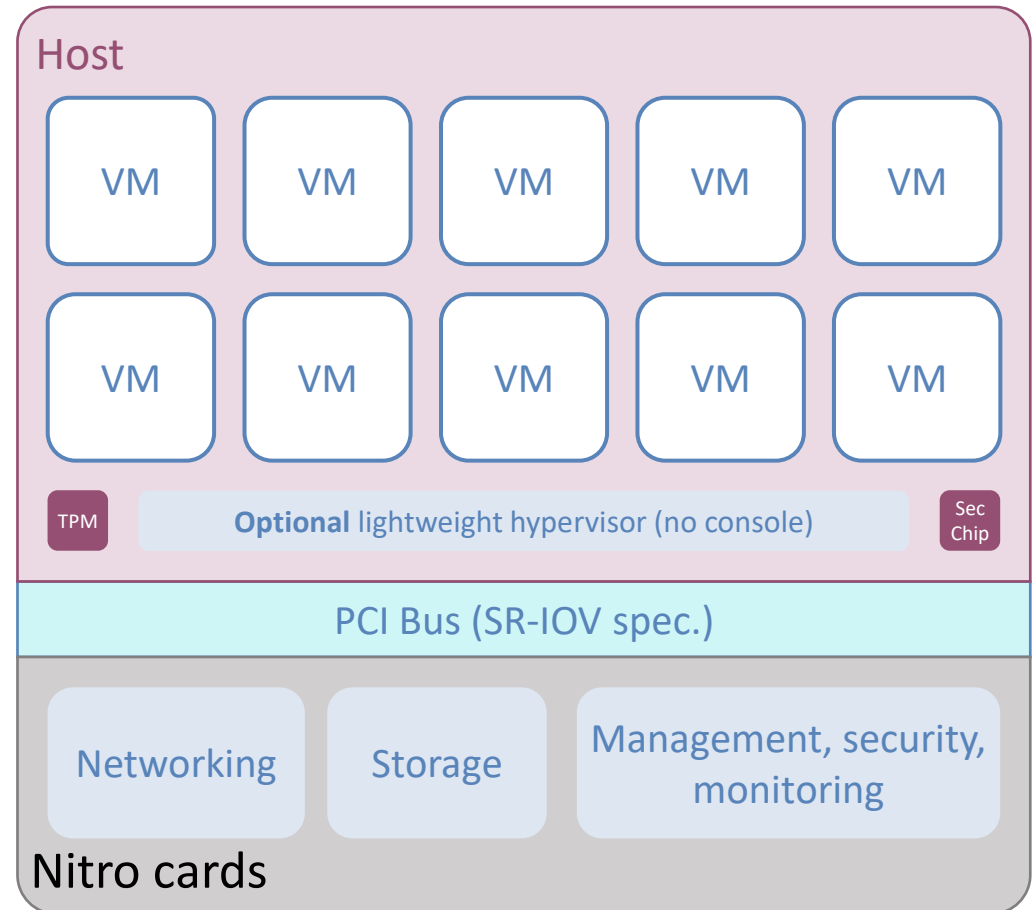
□ Untrusted element ■ Trusted element

Confidential computing on Azure

- Application enclaves
 - Based on Intel SGX
- Confidential VM
 - Based on AMD SEV-SNP
 - To come: VM based on Intel TDX
- Confidential “Kubernetes” containers
 - Based on Intel SGX
 - Aim for “lift-and-shift”
- Attestation
 - Via *Microsoft Azure Attestation* (Microsoft’s signature → need to trust Microsoft)
 - Using AMD’s or Intel’s libraries (manufacturers’ signature, but Microsoft proprietary libraries)
- Cost
 - Additional cost of using confidential option

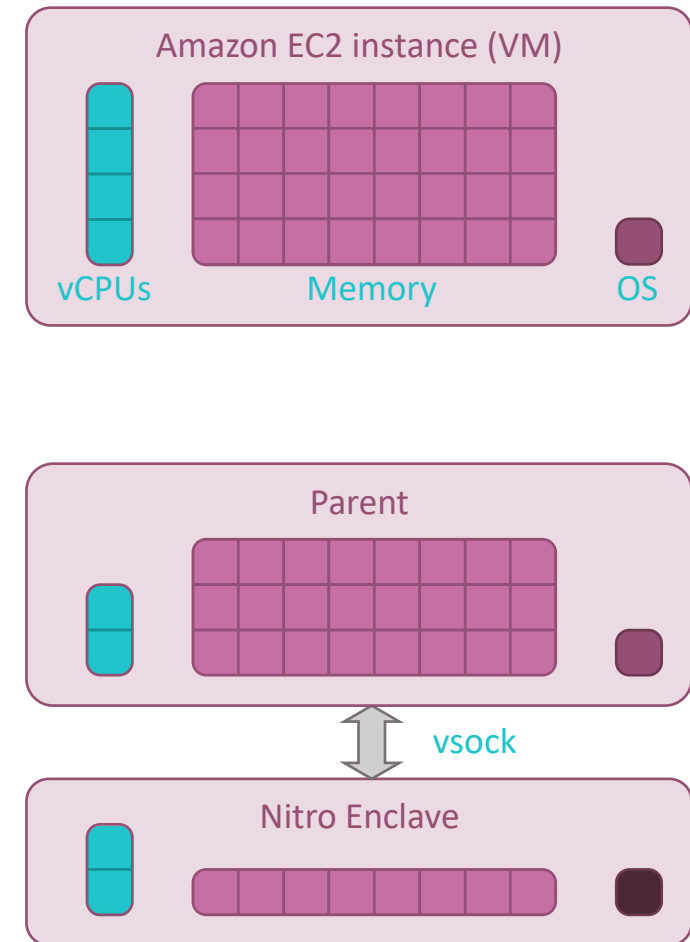
AWS EC2 with “Nitro” architecture

- Hardware-based isolation based on “Nitro cards”:
 - Device model, control plane software, and most hypervisor moved out to these dedicated card
 - Share only power supply and PCIe communication interface
 - Provide hardware-level encryption for all data stored or in transit



AWS “Nitro Enclave”

- Characteristics of a Nitro enclave:
 - Isolated VM running alongside a “parent” EC2 instance (but same board)
 - No persistent storage or networking interface
 - No login access (no shell)
 - Booted with an image file built by the customer
 - No additional cost
- No additional protection from AWS (only from customer’s administrator)
- Remote attestation by AWS (AWS’s signature)
- Since 2023-04-28, possibility to use AMD-SEV-SNP protection as alternative (additional cost)



Initiatives and conclusions

Ongoing initiatives

INAMI

- Establish a platform on Azure to onboard container-based applications of INAMI
 - Without confidential computing, when handling public data
 - With confidential computing, when handling sensitive data
- Contact: Jan Maeckelberghe (RIZIV-INAMI)



Smals

- Showcase the ability to store, process and exchange class-3 data (“Secret - Very confidential”) using confidential computing techniques on AWS
- Contact: Dirk Deridder (SMALS)



General remarks

- HE and MPC not mature enough and limited to niche applications
- TEE provide improved level of protection for computing infrastructures and applications, thanks to:
 - Better process isolation
 - Hardware memory encryption
 - Secure boot
 - Remote attestation
- Main infrastructure offers:
 - AWS Nitro offer is different in nature
 - Azure offers the most varied solution
 - Google's offer appears* less mature than the offers from AWS and Microsoft
- Unresolved trust issue
 - Client still needs to trust the infrastructure provider in practice
- Uncertain overall performance impact due to added complexity

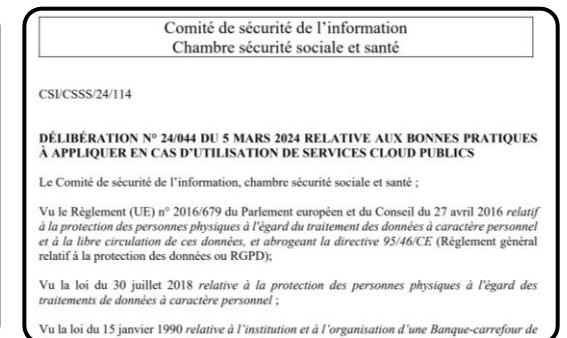
* At time of study during first half of 2023

Recommendations

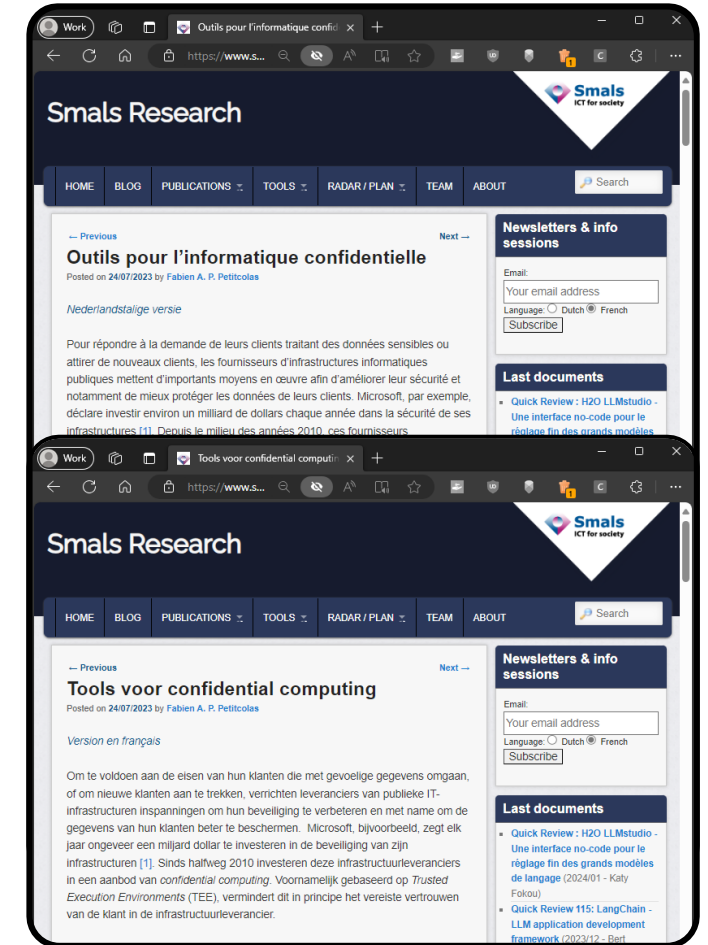
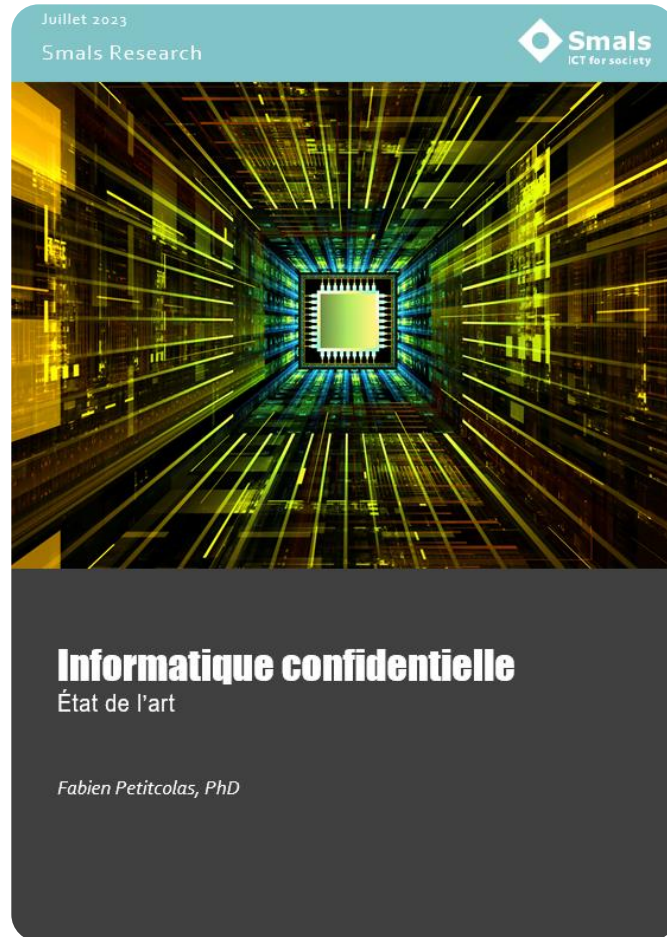
- **Attestation:**
 - Ability to verify TEE content independently from infrastructure provider (e.g., should be signed by hardware manufacturer)
- **Transparency:**
 - Ability to verify source code of any software in trusted computing base (TCB)
- **Key management:**
 - Ability to import own keys on dedicated hardware (minimum)
 - Better: manage keys externally
- **Training:**
 - Provide specific training for analysts, architects, and developers
- **Holistic view:**
 - Consider security of the system as a whole

Additional recommendations

- **Provider access** – Infrastructure provider should have no access to:
 - the processed information (protection at rest and in transit, decryption only in secure enclave)
 - authentication and authorisation management systems
 - servers or enclaves of the user
- **Data disposal** – Data should be disposed upon instruction and at the end of the contract with the provider
- **Vulnerability disclosure** – User should be informed of any vulnerability known to infrastructure provider
- For more details and additional warnings see recommendations of the information security committee of SSCB (KSZ-IVC/BCSS-CSI)



Further reading...



See: <https://www.smalsresearch.be/>