REDOCS

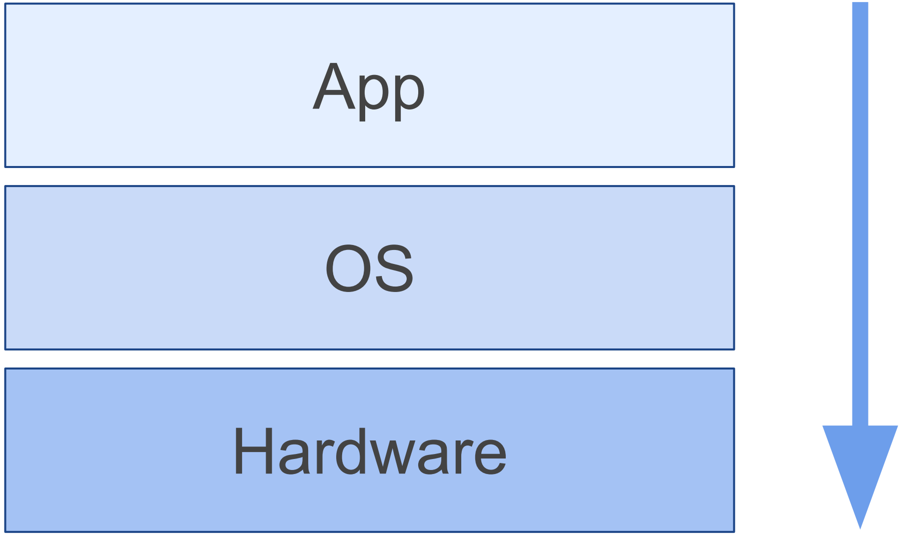
Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE
DES ARMÉES

Security for outsourced computations in the cloud

Cédric VAN ROMPAY (EURECOM)
Valentin LEFILS (DGA / CRIStAL)
Jean OUDOT (LTCI / IRT SystemX / Nanyang Technological University)

How to keep secure and efficient an outsourced software ?



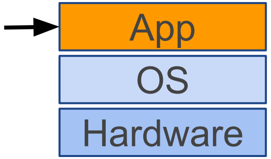
Summary

- ➔ Software based solutions
- ➔ OS based solutions
- ➔ Hardware based solutions
- ➔ Crypto-only solutions
- ➔ Conclusion

Software based solutions

Integrity Verification Kernel : IVK

By D. Aucsmith, 1998

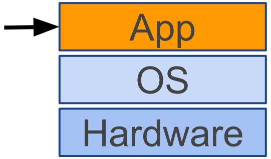


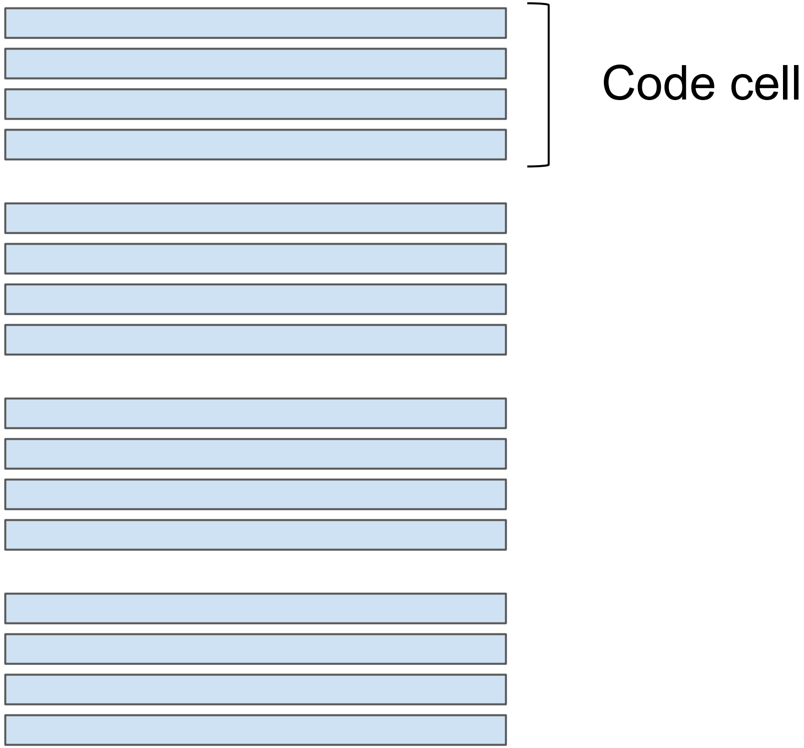
- “Armoured” code segment

- Verifies integrity of a piece of code

- Encryption over execution

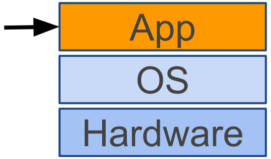
Integrity Verification Kernel: structure

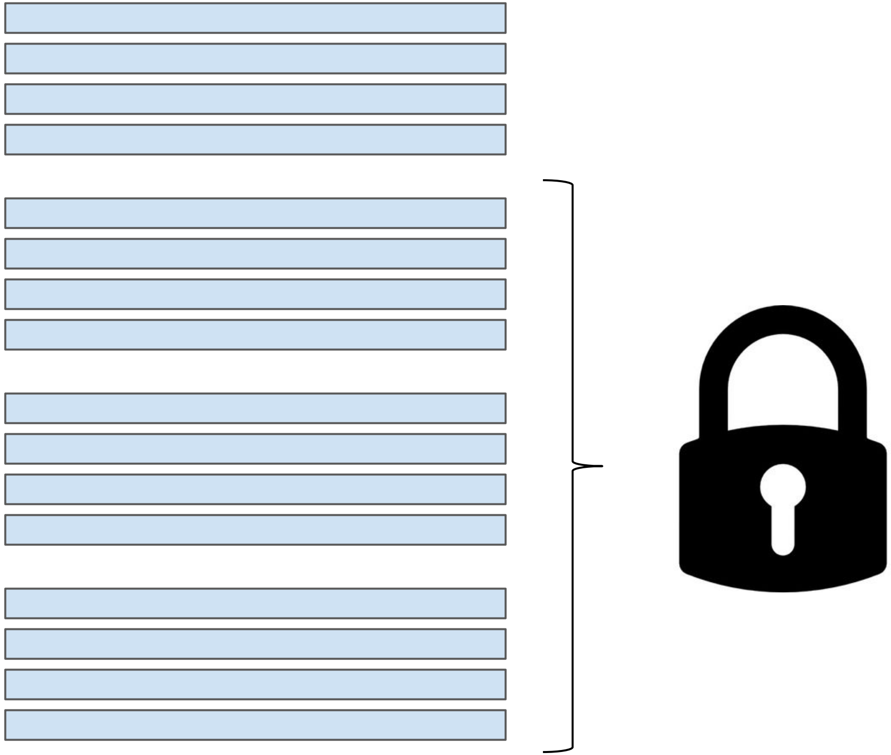




🟦 Block of execution code

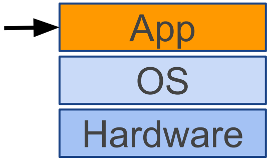
Integrity Verification Kernel: structure





🟦 Block of execution code

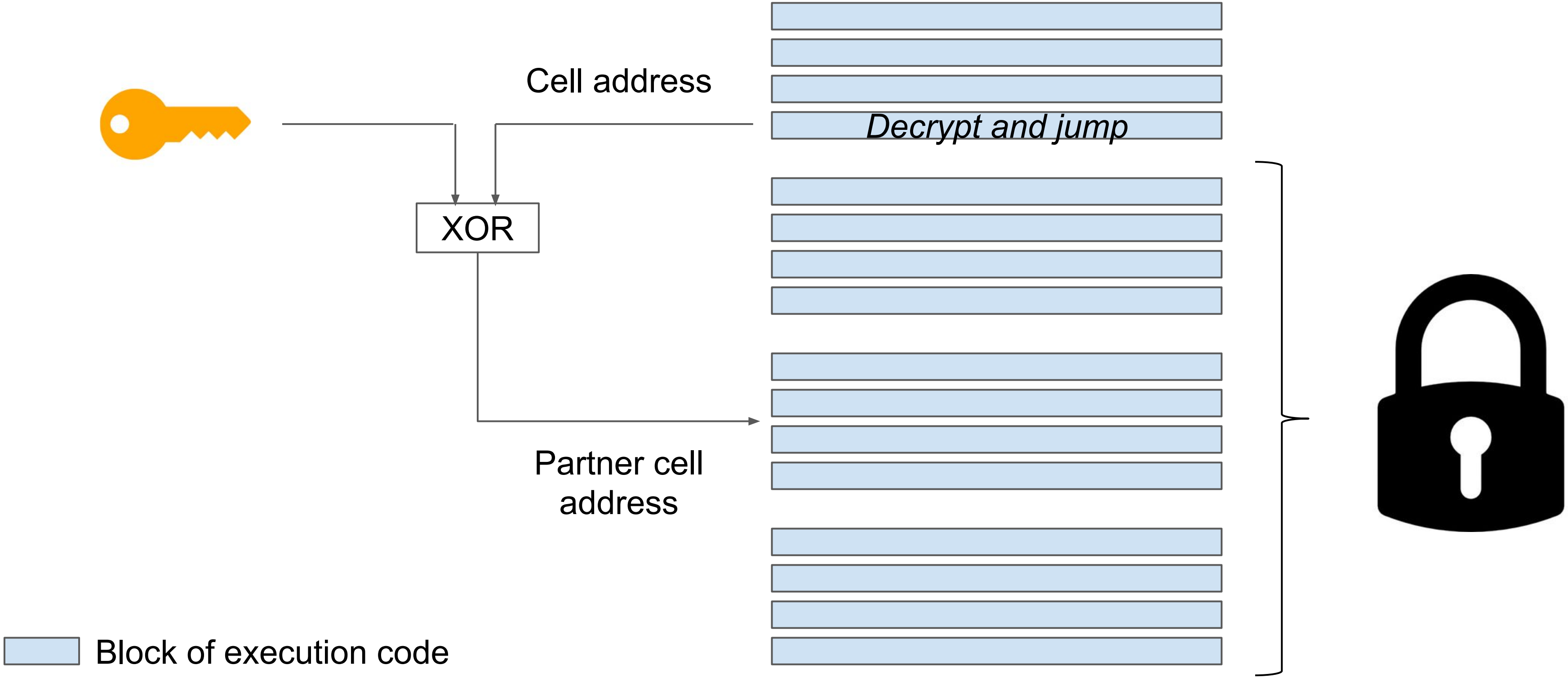
Integrity Verification Kernel: structure



App

OS

Hardware



Cell address

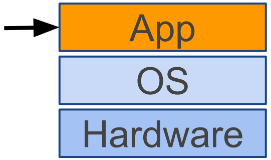
Decrypt and jump

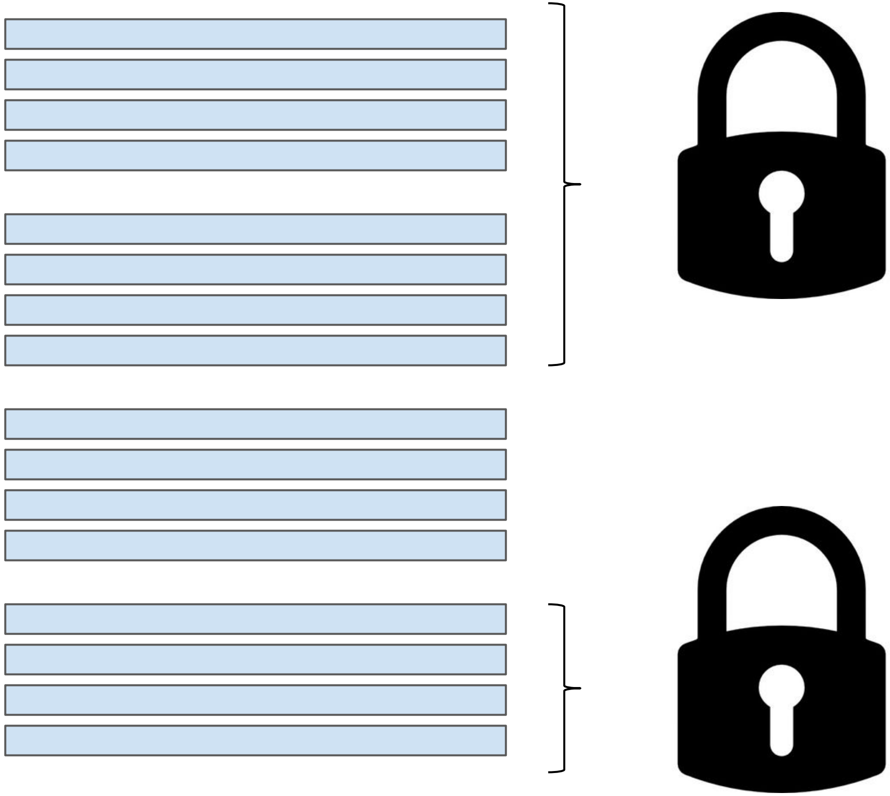
XOR

Partner cell address

▭ Block of execution code

Integrity Verification Kernel: structure



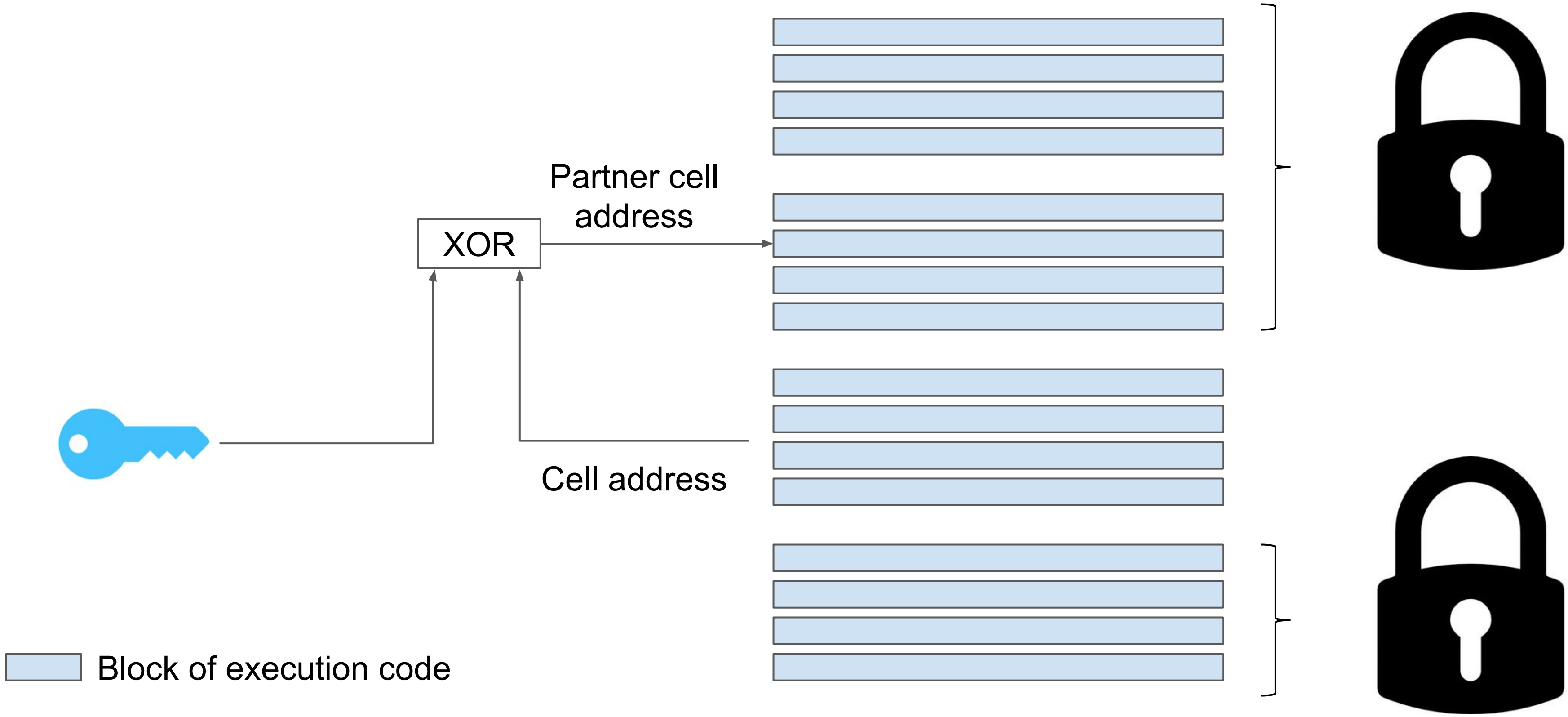


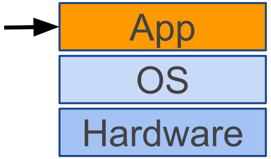
Code is executed

▭ Block of execution code

Integrity Verification Kernel: structure

App
OS
Hardware




App
OS
Hardware

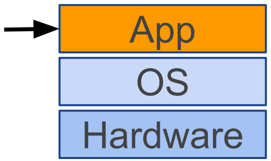
Integrity Ver

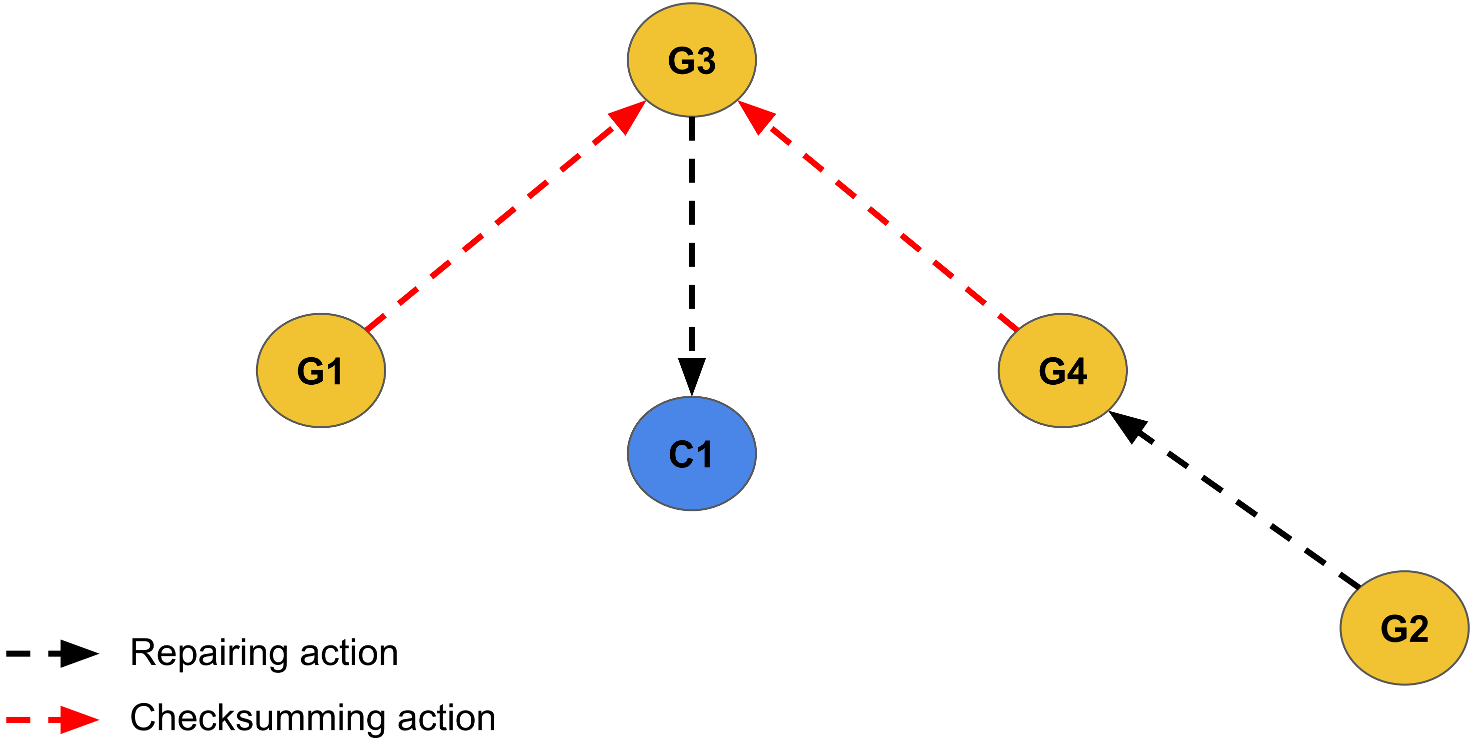
- + Integrity protection
- + Not observable
- + Authenticity check
- + Hard to attack

- - Hardware attack
- - Complex
- - Encryption

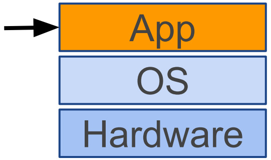
Software protection by guards : guard graphs

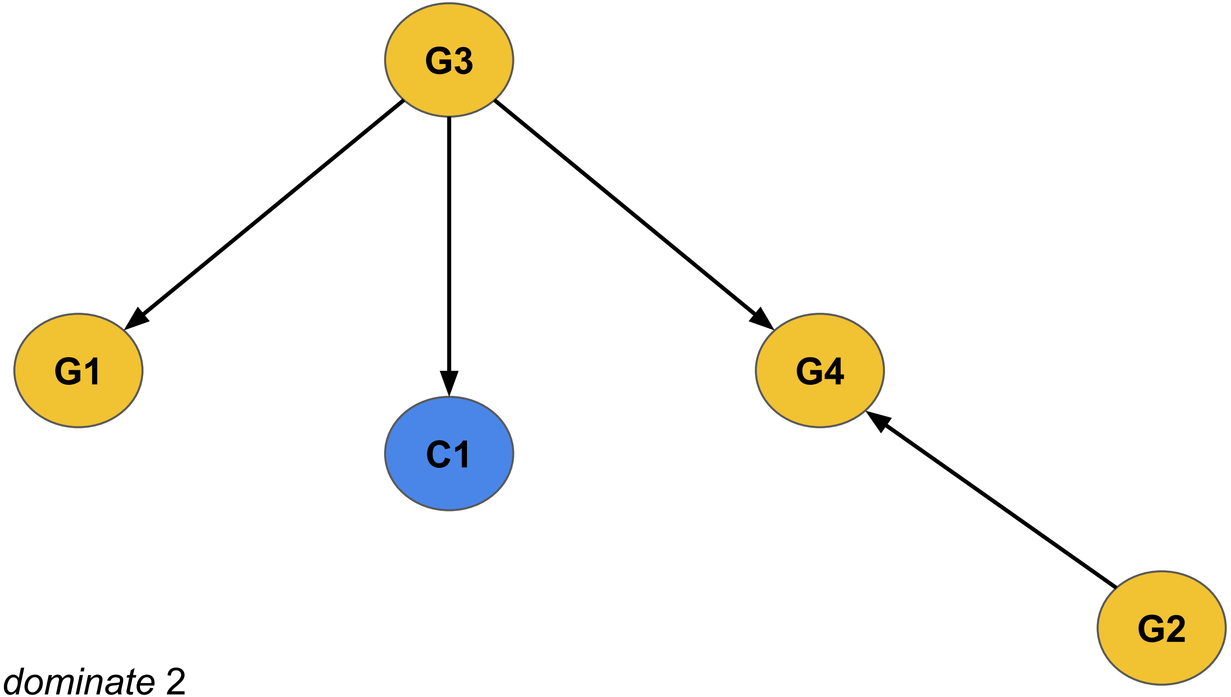
By H. Chang, M. J. Atallah, 2002

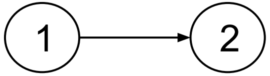




Software protection by guards

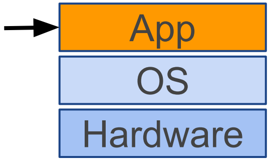


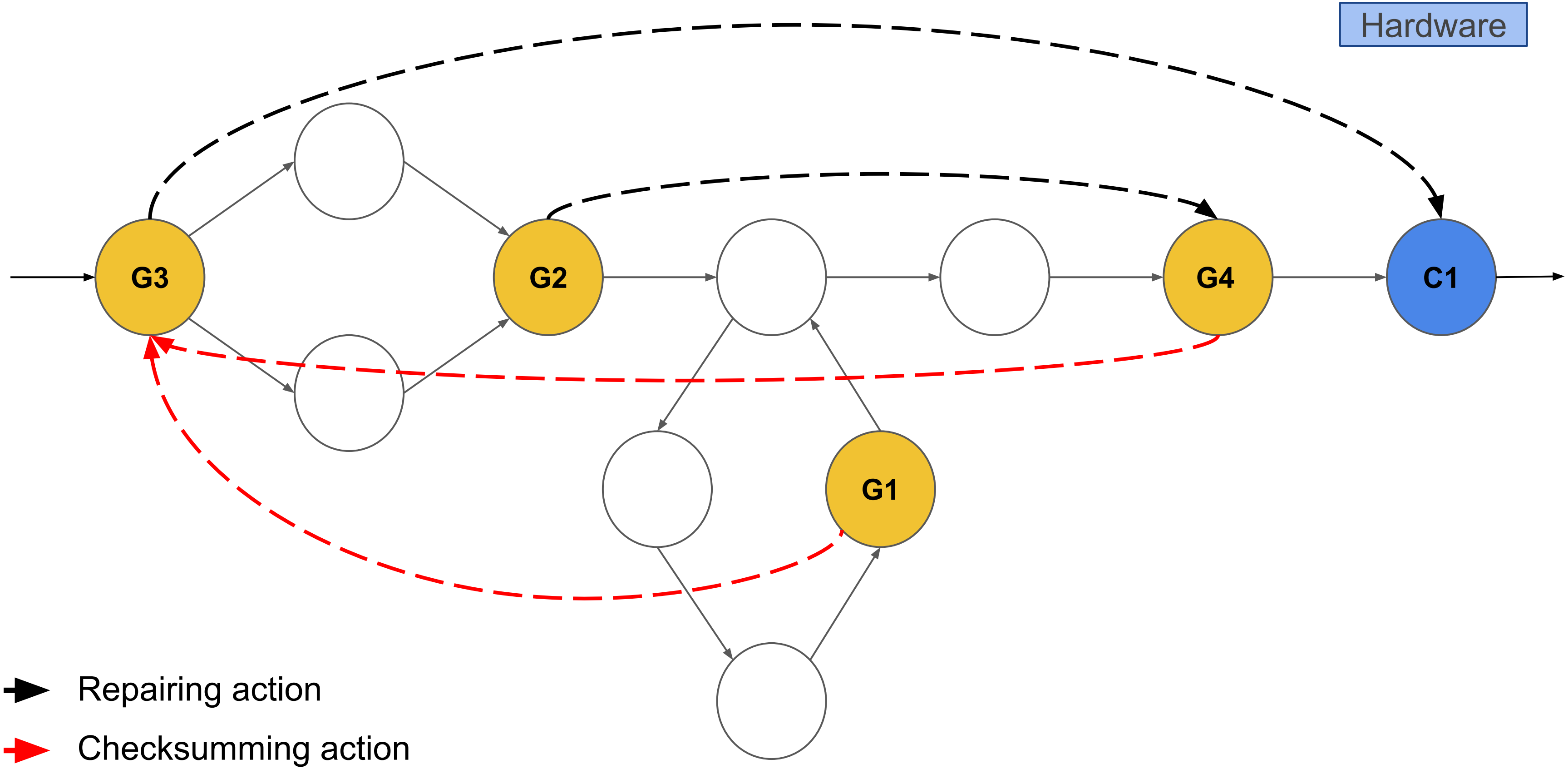




: 1 *dominate* 2

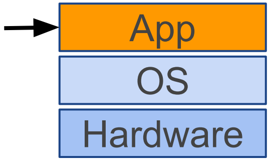
Software protection by guards





- - ► Repairing action

- - ► Checksumming action



App

OS

Hardware

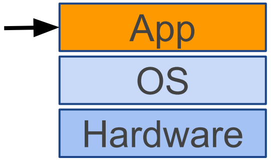
Software protection by guards

+ Partial Integrity
+ Self-healing code
+ Hard to attack

- Observable
- Can be copied
- Template based
- Complex

Software protection for cloud computing

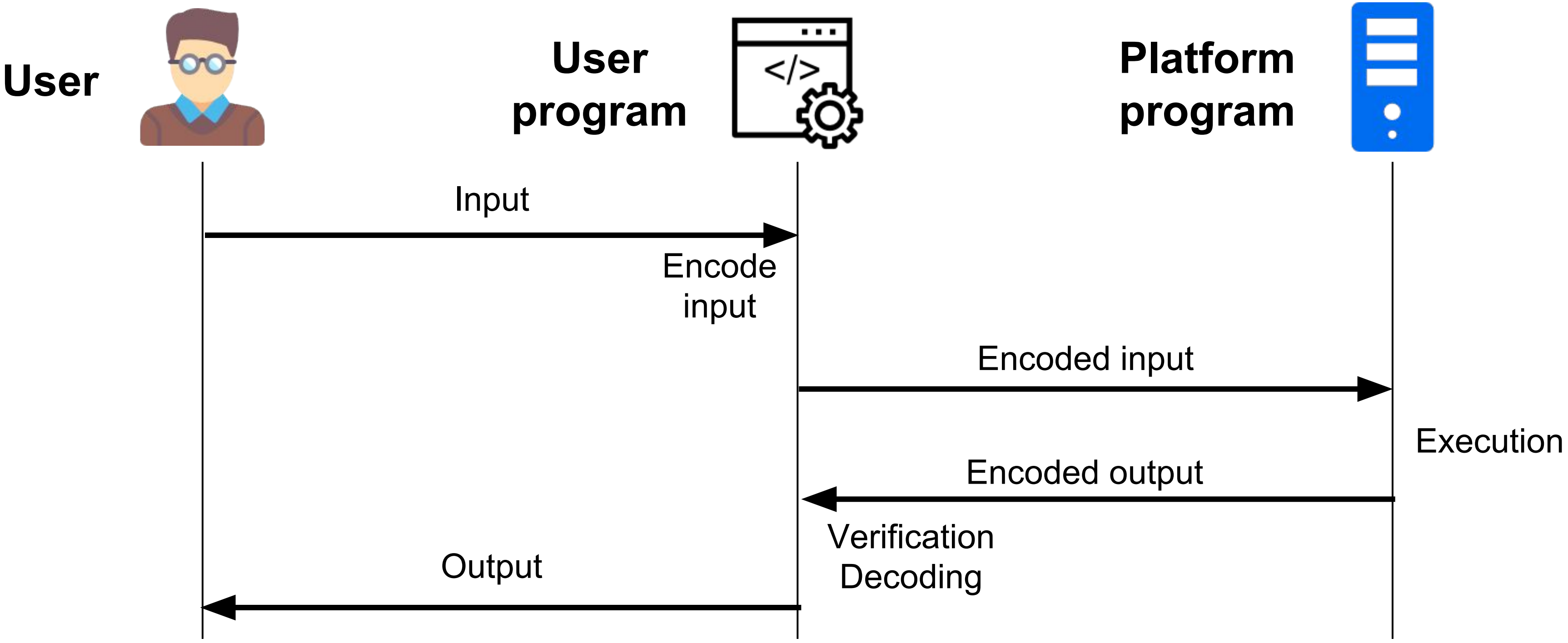
By K. Fukushima, S. Kiyomoto, Y. Miyake, 2012



App

OS

Hardware



User

User program

Platform program

Input

Encode input

Encoded input

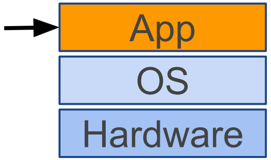
Execution

Encoded output

Verification Decoding

Output

Software protection for cloud computing : rules



+ Easy to compute

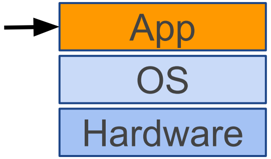
+ Protect code integrity

+ Protect the output

- Encoding

- Needs secure hardware

- Assumptions

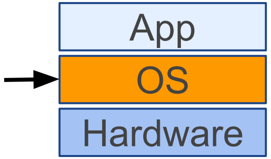


Software based protection: conclusion

- No more innovative solutions
- Dominate by the industry
  - Arxan, Cloakware, StarForce
- Rely a lot on strong expertise
- Rely a lot on a solid hardware base

OS-Based Solution

grsecurity



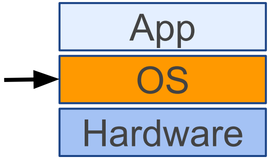
App

OS

Hardware

- Security patch for Linux kernel
- Easy to deploy
- Mitigation against common attacks

grsecurity



App

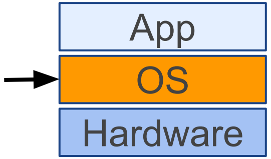
OS

Hardware

- + Protects vs memory exploit
- + Protects vs bruteforcing
- + CHROOT improvement

- - Don’t protect execution
- - Useless if compromised
- - 20% global overhead

grsecurity



+ Protects vs memory exploit

+ Protects vs bruteforcing

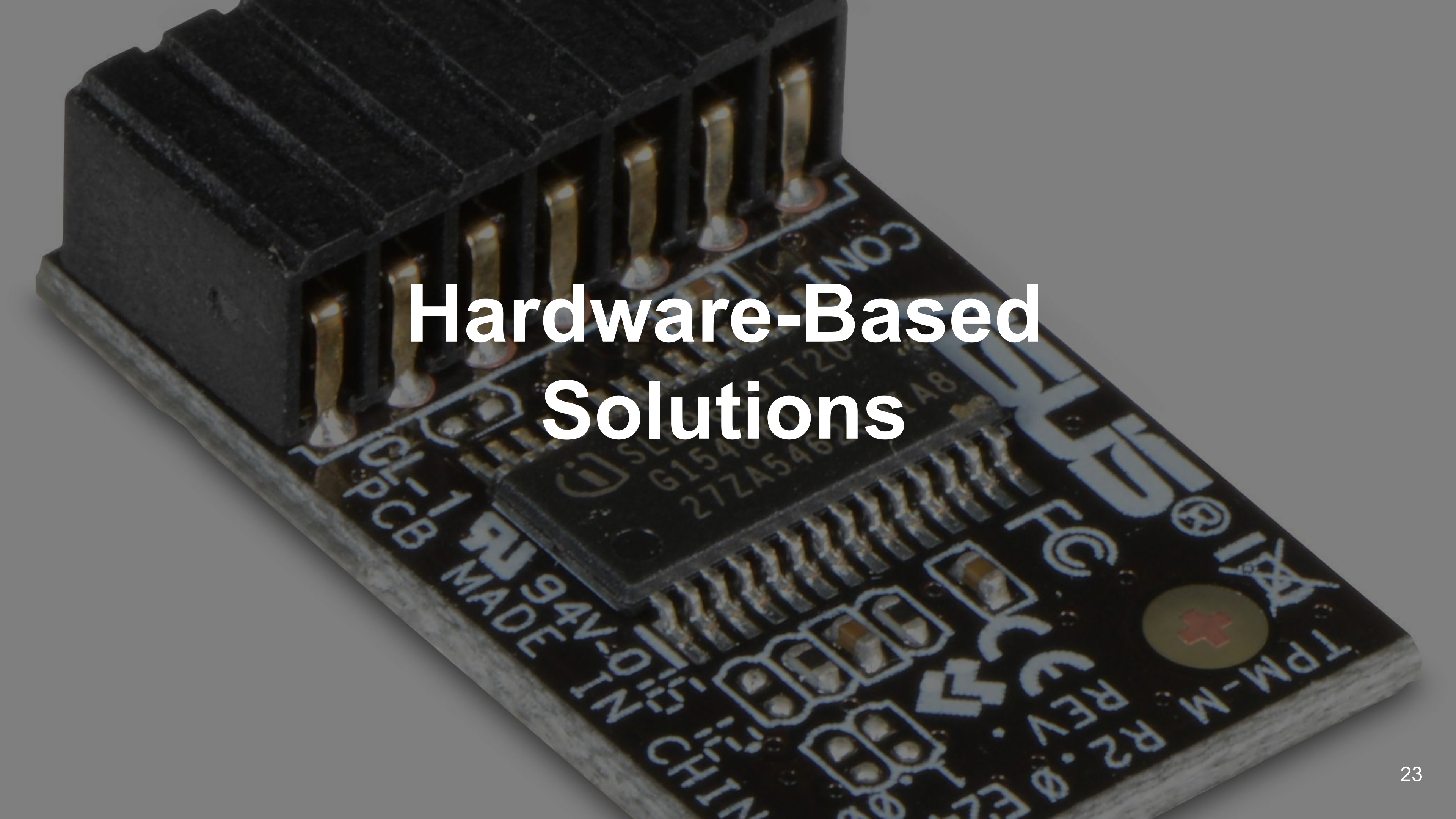
+ CHROOT improvement

- Don’t protect execution

- Useless if compromised

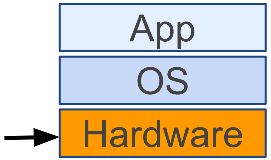
- 20% global overhead

Good practice but not sufficient



Hardware-Based Solutions

Hardware-Based Solutions



App

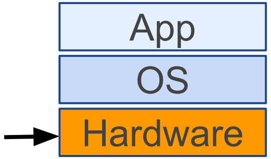
OS

Hardware

+ “Lower” than kernel
+ Small performance overhead

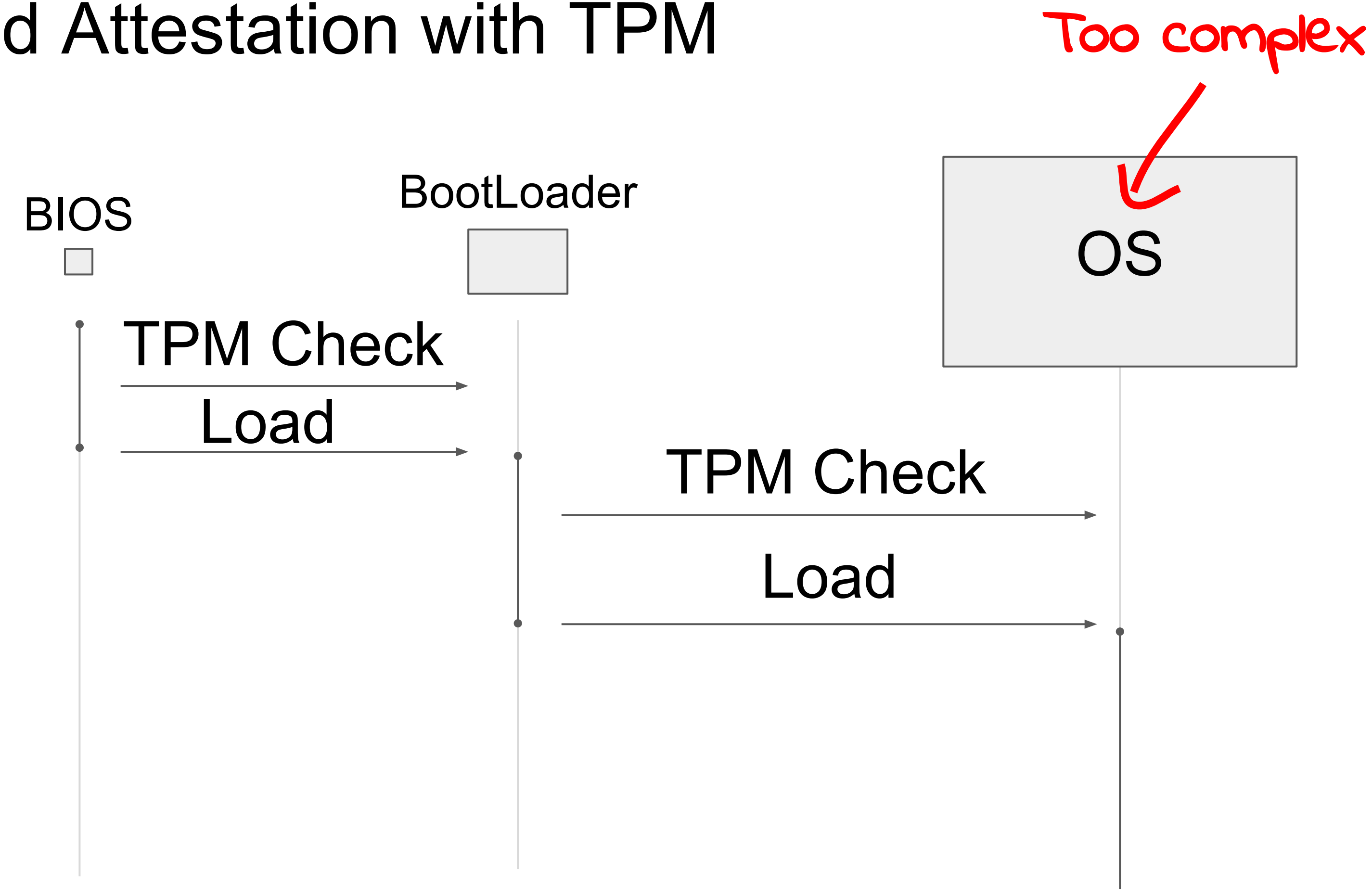
- Must be present
- Hard to update
- Hard to patch!

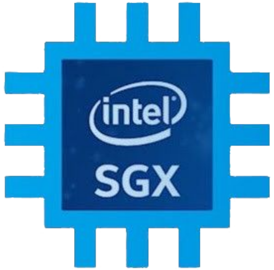
Trusted Platform Module (TPM)

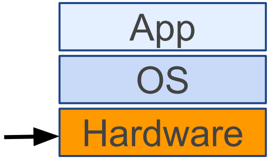


- External to CPU
- High-level cryptographic operations
- Main use: building a “Root of Trust” through ***chained attestation***

Chained Attestation with TPM



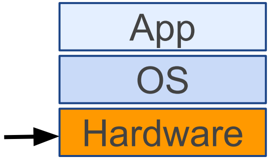



App

OS

Hardware

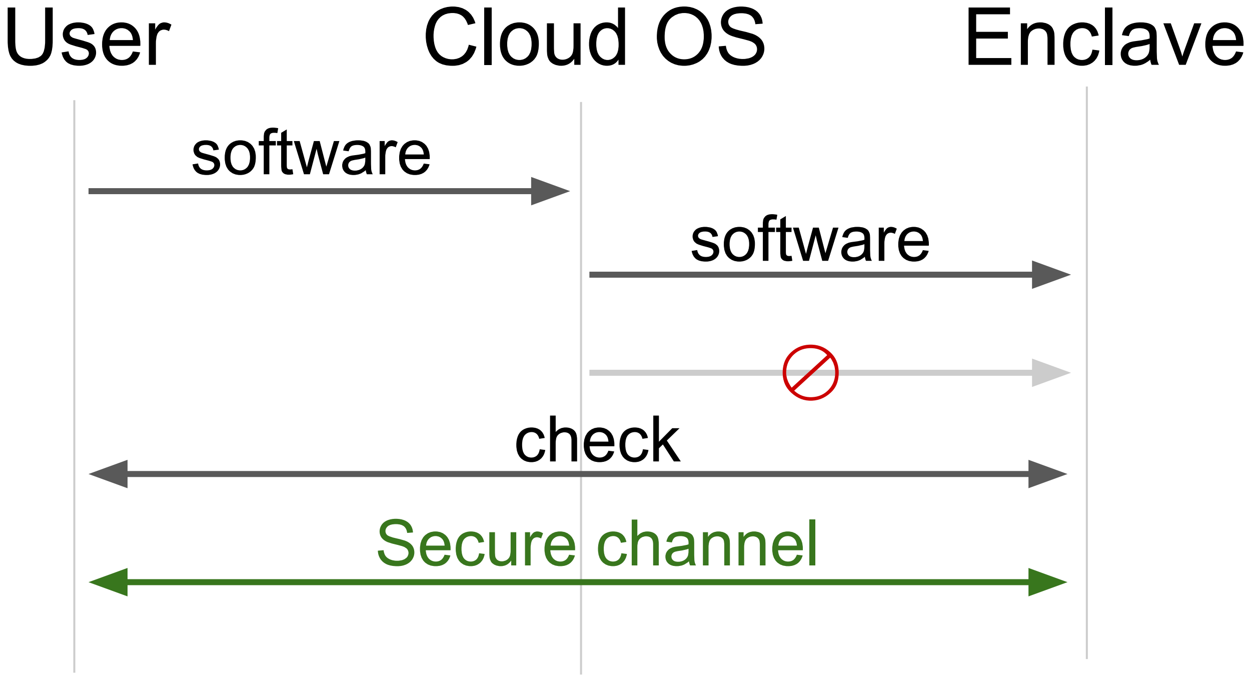
- Available on Intel processors since 2015
- Secure enclaves
- Dedicated driver
- Hardware cryptographic primitives



App

OS

Hardware



User

Cloud OS

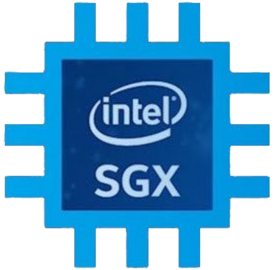
Enclave

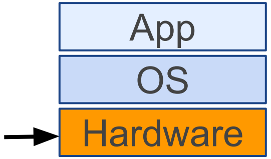
software

software

check

Secure channel

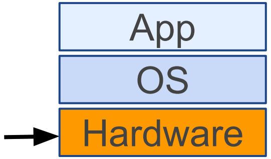




- + Isolated environment
- + Binary loading verification
- + Private encrypted memory
- + Sealed data

- - Hardware dependent
- - Code rewriting

intel SGX



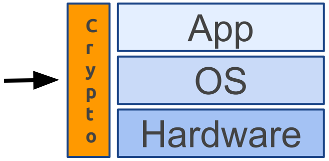
+ Isolated environment
+ Binary loading verification
+ Private encrypted memory
+ Sealed data

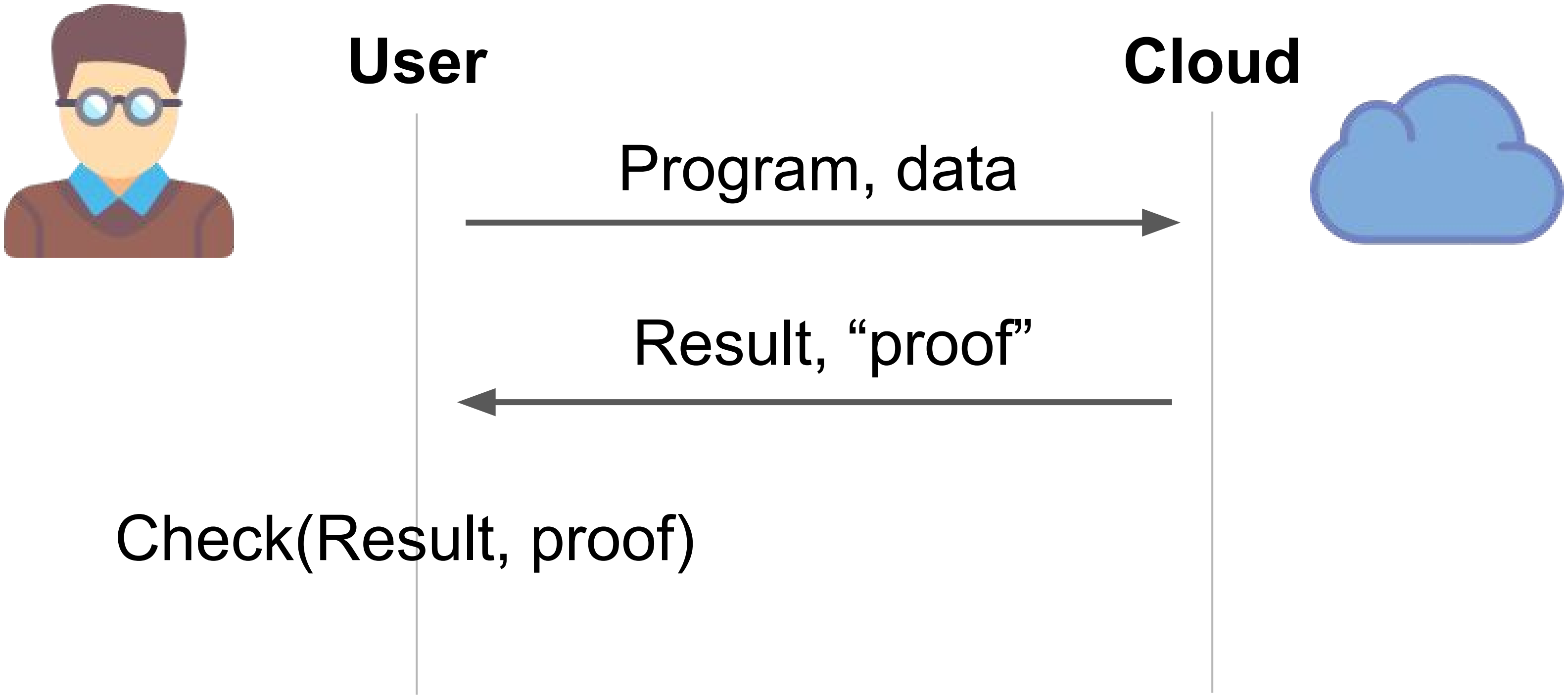
- Hardware dependent
- Code rewriting

Very good solution if the hardware supports it

Crypto-only Solutions

Verifiable Computation from Cryptography





User

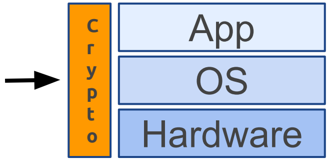
Cloud

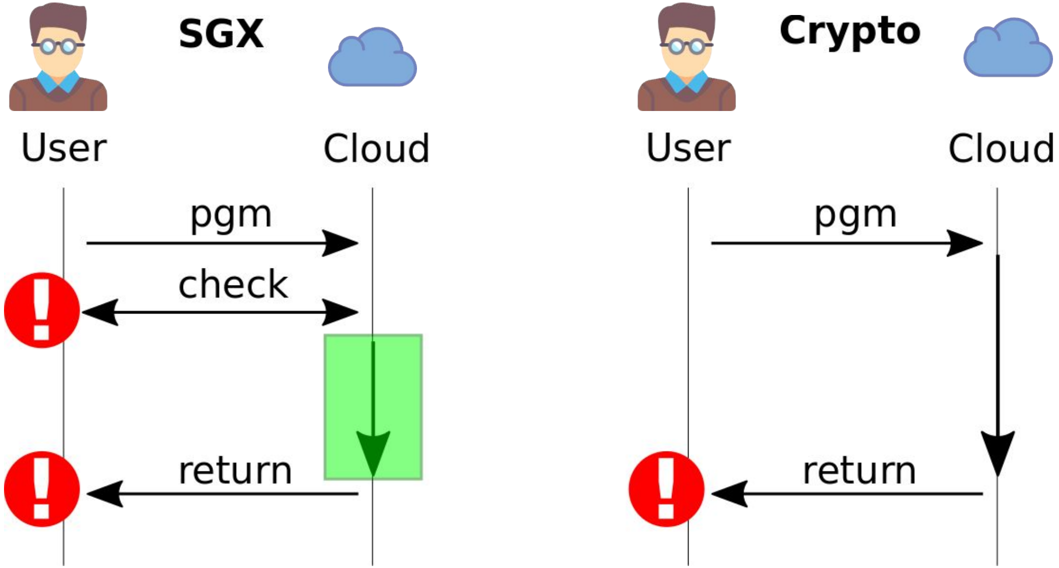
Program, data

Result, "proof"

Check(Result, proof)

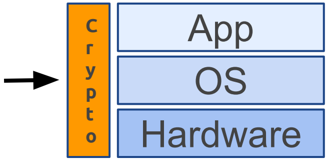
“Detection” vs “Prevention”





Eventually, you always rely on detection

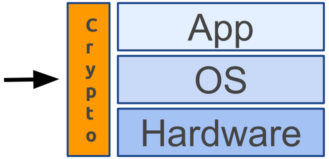
Crypto-only



+ No HW requirements
+ Secure against compromised hardware

- Very specific primitives (“niche” solution)
- Moderate to high overhead

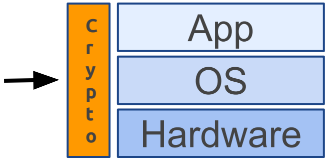
Verifiable Computation Example #1

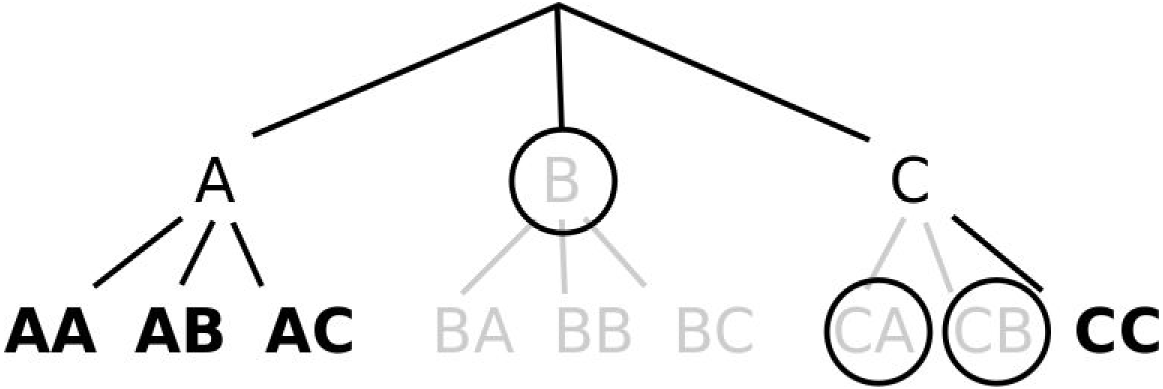


A simple toy example:
Verifiable database from signatures.

(DEMO)

Verifiable Computation Example #1

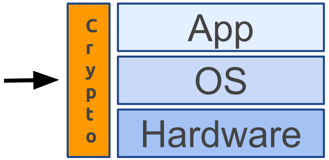




A prefix tree (or "trie")

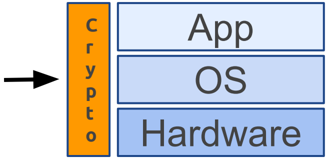
Verifiable Computation Example #2

Benabbas, Gennaro and Vahlis, CRYPTO 2011



from Verifiable Polynomial Evaluation
to Verifiable Database with updatability and query privacy

Using Privacy Crypto for Verifiability



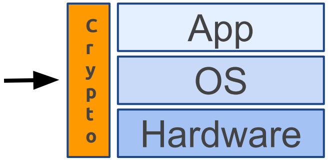
crypto

App

OS

Hardware

- Idea: blind adversary would be limited to random, easy to detect modifications
- ***May*** work in some use cases



Using Privacy Crypto - Garbled Circuits

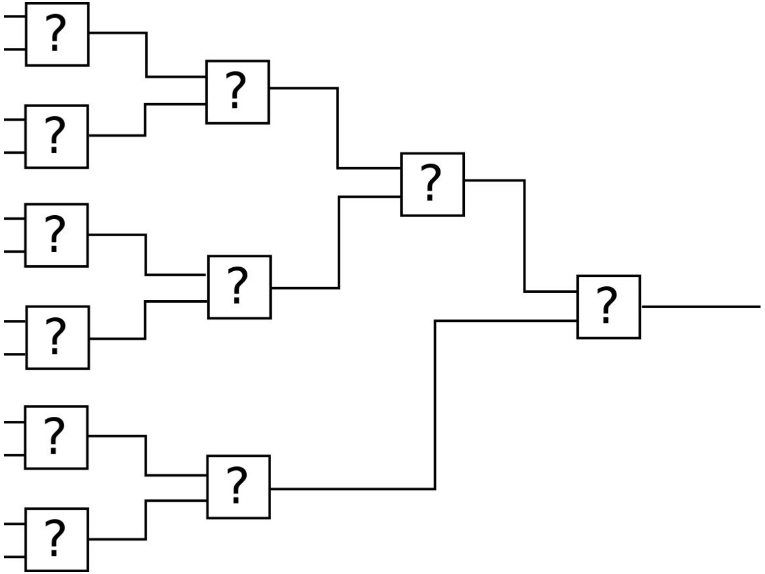
Yao

Cloud sees:

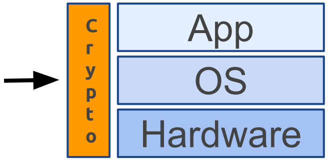
- Topology of circuit

Doesn't see:

- Gate types
- Values on wires

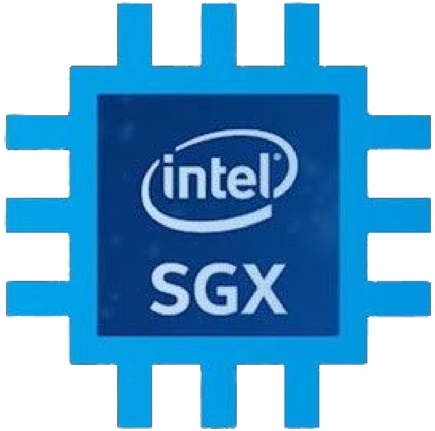


Using Privacy Crypto - Fully Homomorphic Encryption



FHE doesn’t even hide the program (only the data)

WINNER



Thank you for your attention

Do you have any questions ?