Hardware Attacks

Paolo PRINETTO

Director
CINI Cybersecurity
National Laboratory

Paolo.Prinetto@polito.it

Mob. +39 335 227529

CYBER CHALLENGE.IT

CYBERSECURITY NATIONAL LABORATORY

https://cybersecnatlab.it

License & Disclaimer

License Information

Disclaimer

- We disclaim any warranties or representations as to the accuracy or completeness of this material.
- Materials are provided “as is” without warranty of any kind, either express or implied, including without limitation, warranties of merchantability, fitness for a particular purpose, and non-infringement.
- Under no circumstances shall we be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this material.

Acknowledgments

- The presentation includes material from
  - Giorgio DI NATALE
  - Nicolò MAUNERO
  - Gianluca ROASCIO

  whose valuable contribution is here acknowledged and highly appreciated.

Prerequisites

- Lectures:
  - *HS_1.1 - The role of Hardware in Security*
  - *HS_1.2 - Hardware Vulnerabilities*

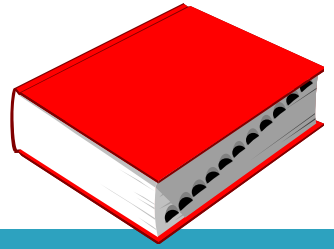
Goal

- Presenting the concept of *Hardware Attacks*
- Providing a comprehensive taxonomy and a detailed analysis of the most significant ones, clustered according to their
  - *goal* (stealing, corrupting or inhibiting)
  - *target* (information or property)
  - *modality* (invasive or non-invasive)
  - *domain* (logical or physical).

Outline

- Hardware Attacks
- Attack goals
- Attack targets
- Attack modalities
- Attack domains

Hardware Attack

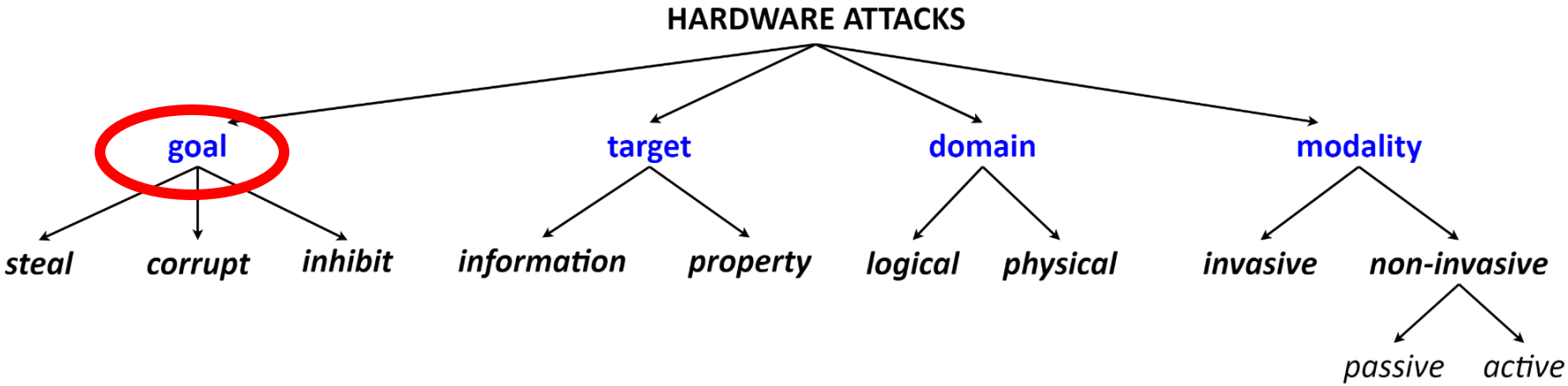


- The act of exploiting a hardware vulnerability to perform an attack.

Hardware Attacks

- Can be clustered according to:
  - Their goal (*stealing*, *corrupting* or *inhibiting*)
  - Their target (*information* or *property*)
  - Their modality (*invasive* or *non-invasive*)
  - Their domain (*logical* or *physical*).

Hardware Attacks Taxonomy



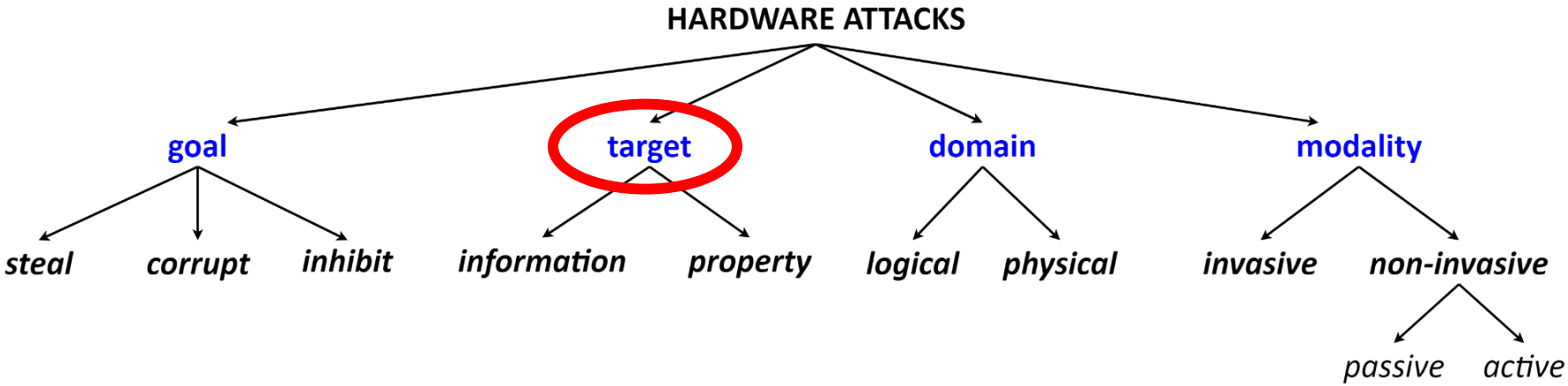
Hardware Attacks Goals

- An attack always exploits a vulnerability, which always harms one of the three CIA properties

Hardware Attacks Goals

- An attack is then mounted to:
  - *steal* a target (e.g., a cryptographic key, a secret password, an intellectual property, a resource, etc.) → *Confidentiality* is broken
  - *corrupt* a target (e.g., a memory word, a permission file, a functionality, etc.) → *Integrity* is broken
  - *inhibit* a target (e.g., a service, a set of critical data, a defense mechanism, etc.) → *Availability* is broken

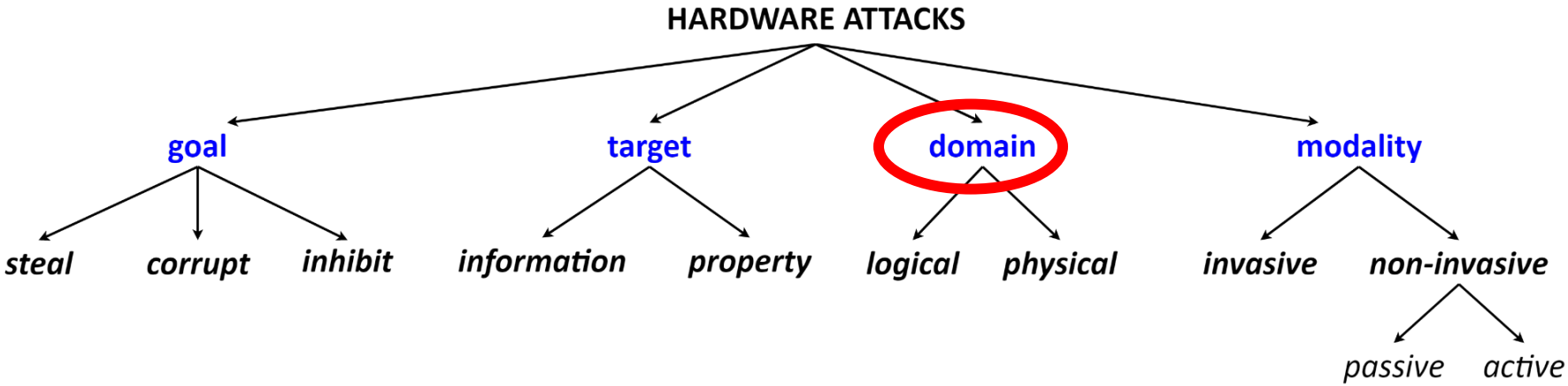
Hardware Attacks Taxonomy



Hardware Attacks Targets

- The target of a hardware attack is always an asset treated or owned by a hardware component, which can be:
  - A piece of *information* treated by the hardware (a key, some credentials, a protected memory word or file, etc.)
  - A *property* owned by the hardware (its IP, one of its resource or functionalities, etc.)

Hardware Attacks Taxonomy



Attack *domains*

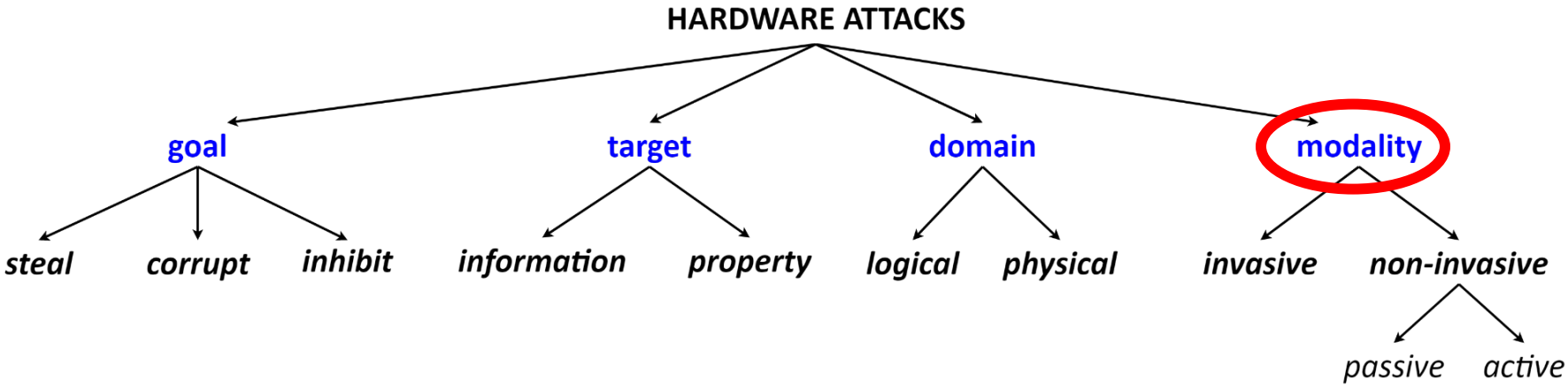
Logical

- Carried out exploiting the software run by the hardware
  - E.g., privilege escalation via software exploiting a hardware vulnerability

Physical

- Carried out directly exploiting a hardware vulnerability

Hardware Attacks Taxonomy



Hardware Attacks Modalities

Non-invasive

- Carried out without any physical contact with the device under attack.
- Further clustered in:
  - *Passive* non-invasive attacks
  - *Active* non-invasive attacks

Invasive

- Require actions against the attacked hardware (e.g., desoldering, depackaging, disconnections) in order to allow a physical intrusions to easily access some internal components.

Different "investments"

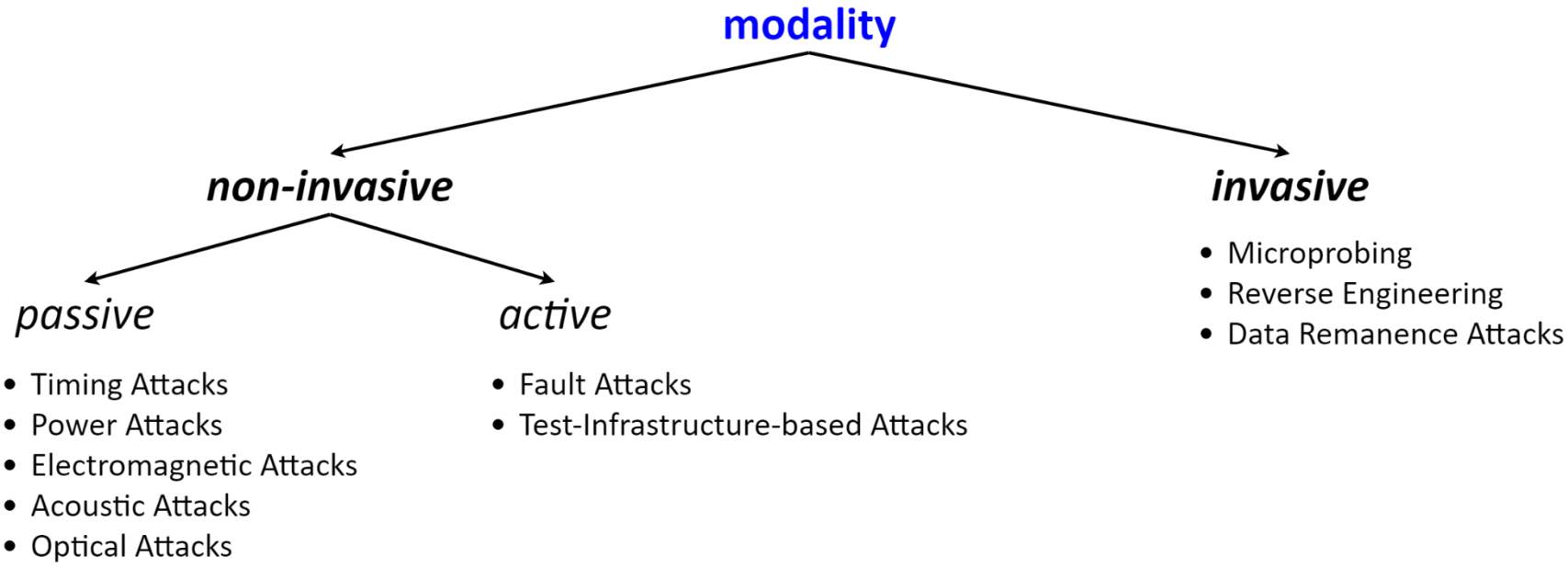
Non-invasive

- They typically require moderate investments in terms of both lab equipment and effort in designing an attack on the target device
- Therefor the cost per device attacked is low

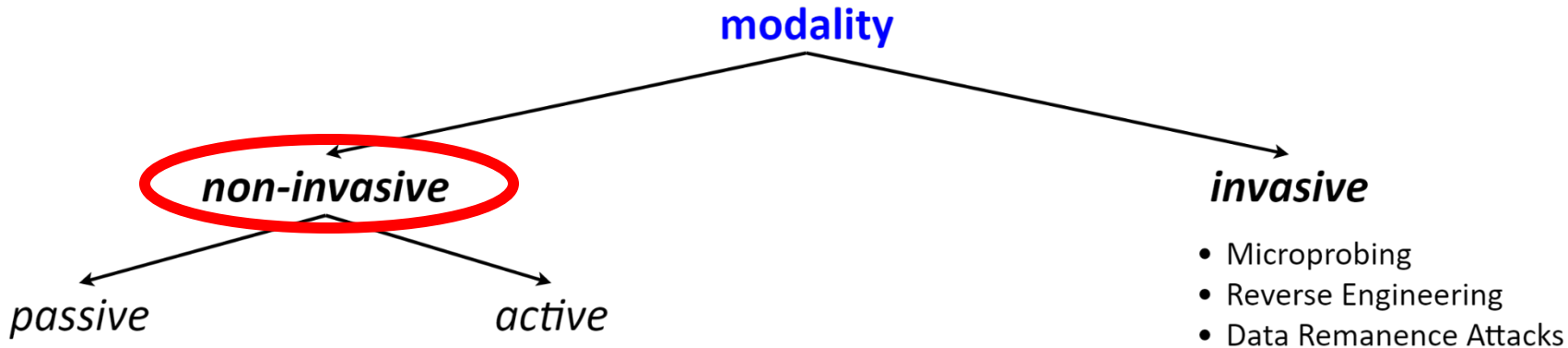
Invasive

- They typically require:
  - a relatively high capital investment for lab equipment
  - a moderate investment of effort for each individual chip attacked.

Hardware Attacks Modalities



Hardware Attacks Modalities



Non-invasive Hardware Attacks

Passive

- Carried out by analyzing and measuring one (or more) physical dynamic entities of the attacked hardware.

Active

- Require specific actions on the device, aimed at forcing the system into abnormal states, in which the attacker's goal is easier to reach.

Non-invasive Hardware Attacks

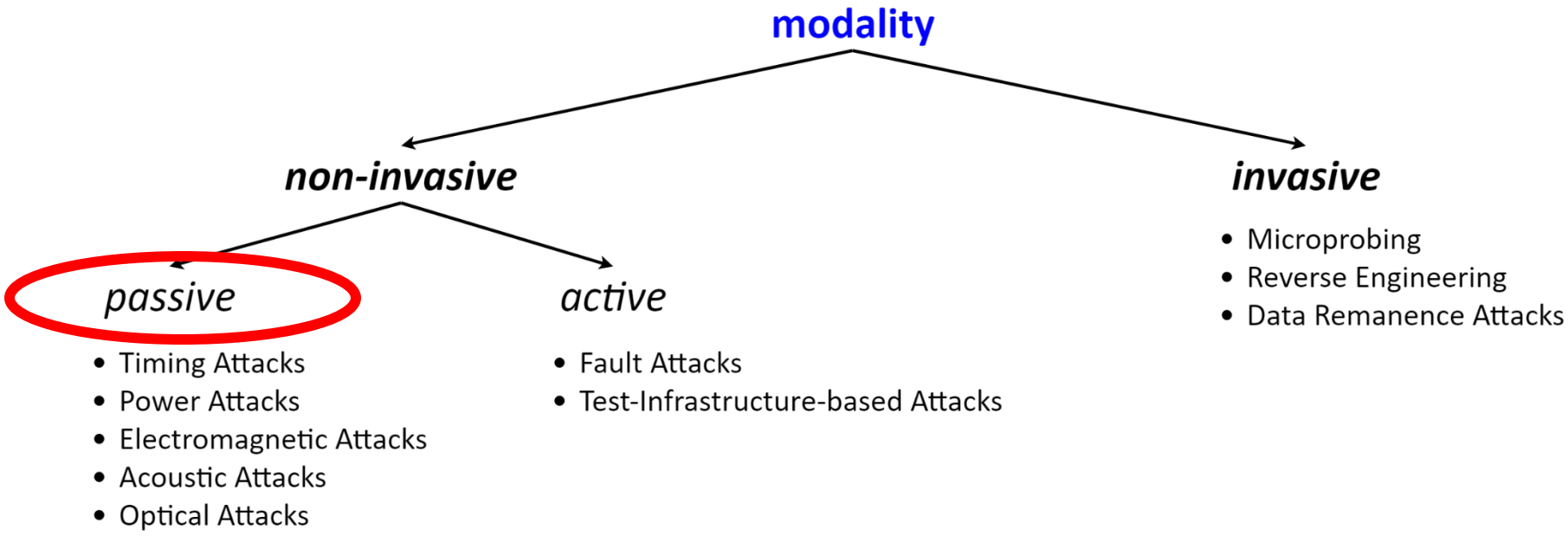
Passive

- The device works within its specifications

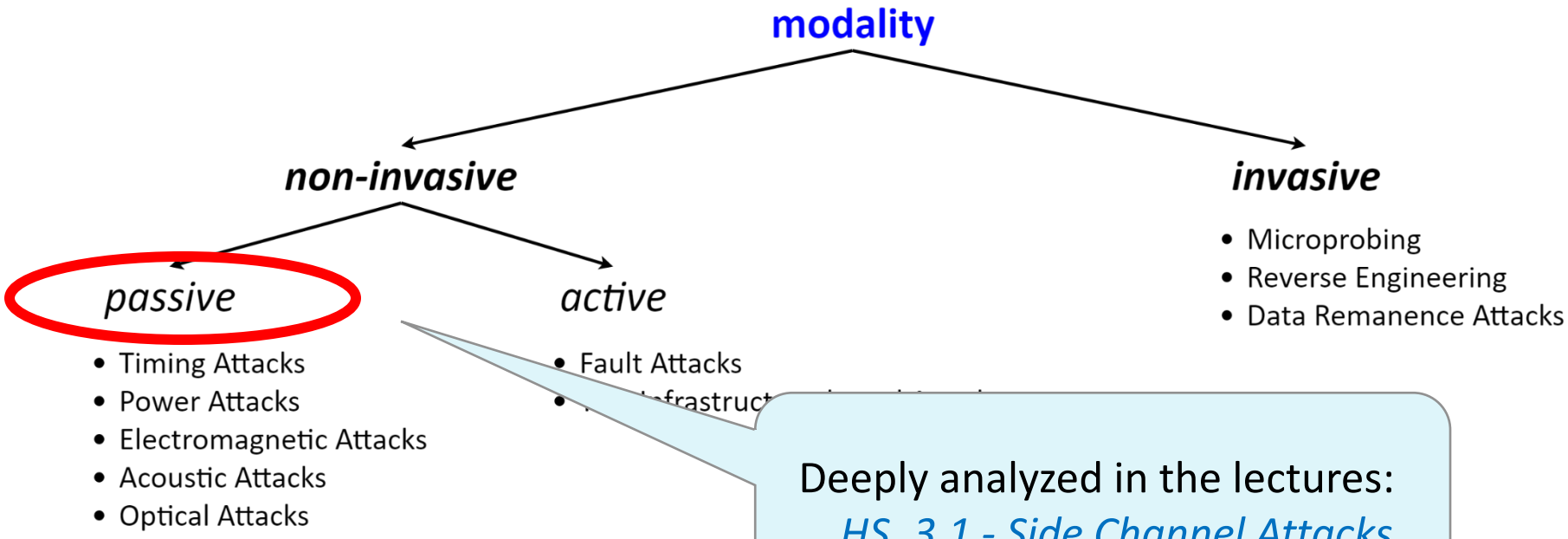
Active

- The goal is reached by exploiting abnormal behaviors of the device
  - E.g., power glitches, laser pulses

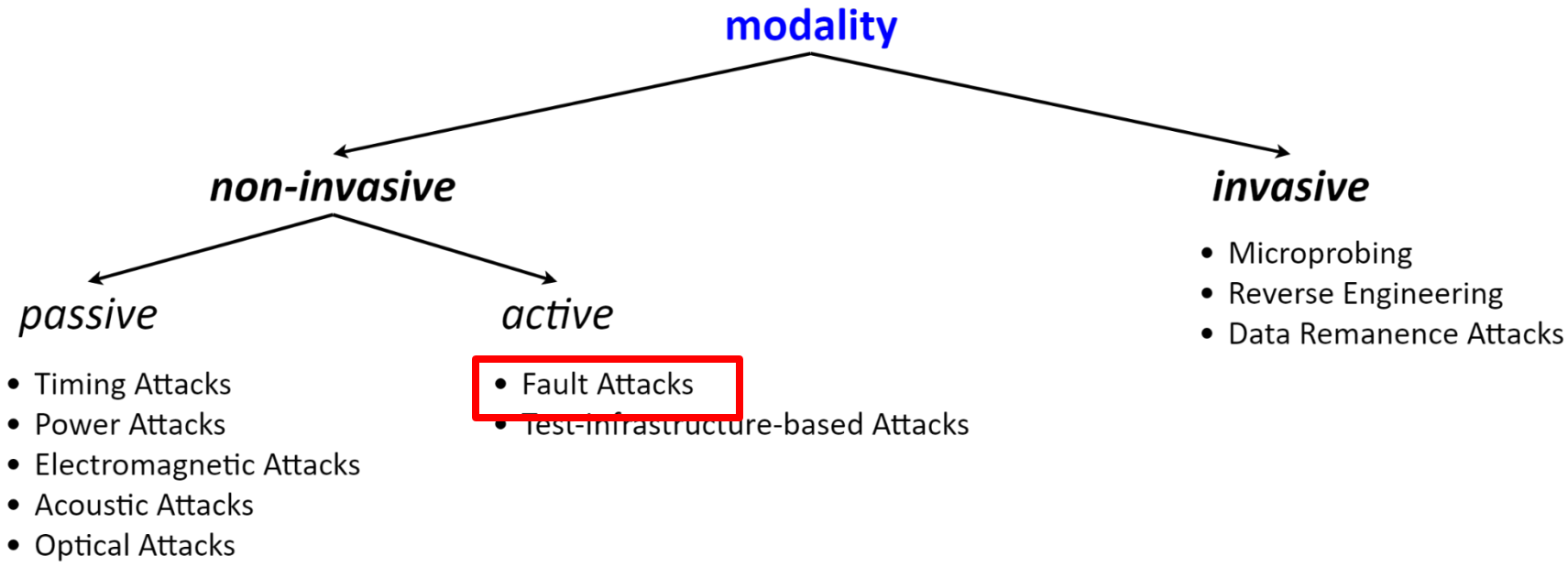
Hardware Attacks Modalities



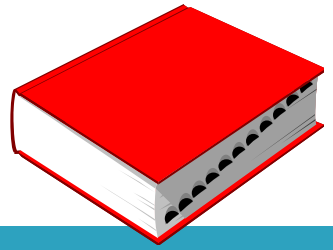
Hardware Attacks Modalities



Hardware Attacks Modalities



Fault Attacks



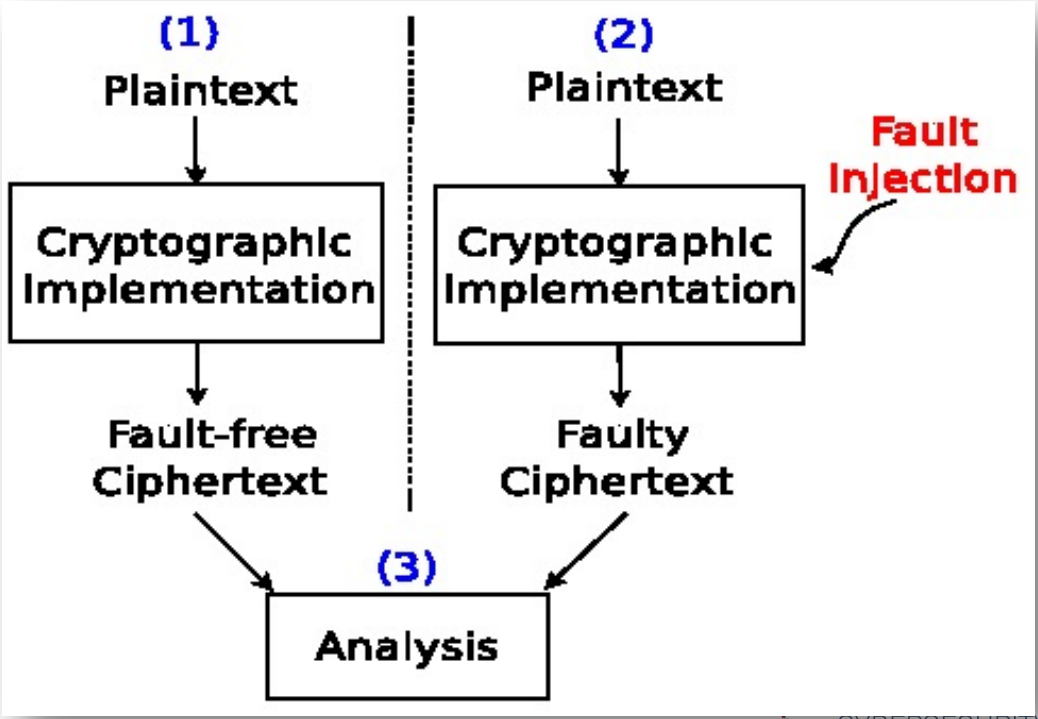
- Consist in the injection of deliberate (malicious) faults into the target device, aimed at bringing it into a set of states from which private internal information items (e.g., a key) can be fraudulently extracted.

Fault Attacks -- An example

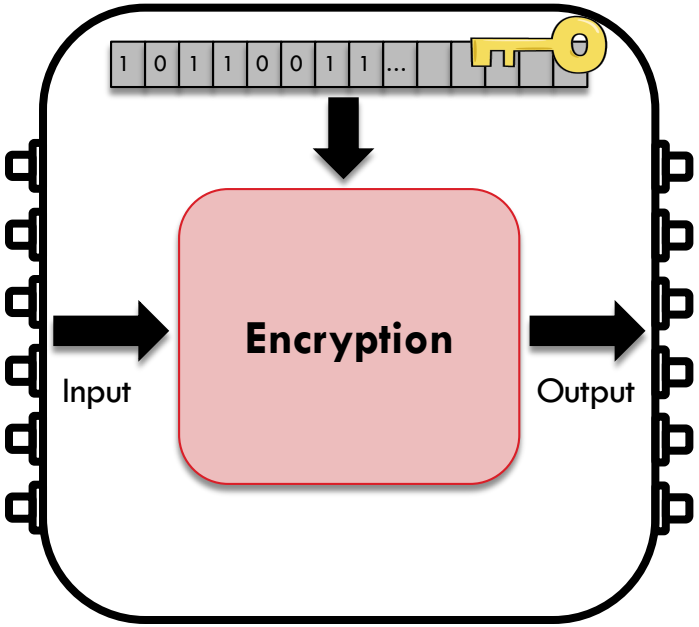
- They exploit the possibility of inserting a fault in the process of algorithm execution in a way that it could help revealing the key.

Fault Attacks -- An example

- Practically it changes the value of the key during the processing of the algorithm and compare the two outputs.



Fault Attacks



Hypothesis: Injection forces a '0' on a single bit of the secret key

1) $C_{OK}=E(P)$

2) Calculate C'=E(P), while injecting a fault

3) If C' = C_{OK} → target bit is '0' else → target bit is '1'

Fault Injection Techniques

- To inject faults affecting critical paths:
  - Underpowering or overpowering
  - Altering the clock
  - Altering the temperature

Fault Injection Techniques

- To inject precise faults in space and time:
  - *Laser injections*

Fault Injection Techniques

- To inject precise faults in space and time:
  - *Laser injections*
- The most precise injection method
- Requires depackaging of the target device

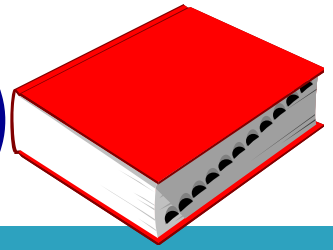
Fault Injection Techniques

- To inject precise faults in space and time:
  - Laser injections
  - *Electromagnetic injections*

Fault Injection Techniques

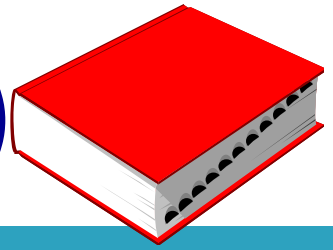
- To inject precise faults in space and time:
  - Laser injections
  - *Electromagnetic injections*
- *Harmonic signals* to target analog circuits
- *Pulse signals* to target digital circuits

Differential Fault Analysis (DFA)



- A set of multiple injections with subsequent correlated analysis of the faulty behaviour of the target device

Differential Fault Analysis (DFA)



- The attackers (cryptanalysts) focus on the fault influences of specific hardware implementations to collect some faulty outputs.
- Later, this information is compared and analyzed with correct results in order to harvest some partial or total compromise of the secret information.

Injection goals

- Bypassing access/right control verification
  - To get access to the system
- Generating faulty encryptions/signatures
  - To be exploited by cryptanalysis in order to retrieve the secret key
- Denial of Service

Countermeasures to Fault Attacks

- Several approaches have been proposed, including:
  - Preventing the attack
  - Detecting the fault injection
  - Detecting the fault effect (error)
  - De-synchronization
  - Robust package, “hardened” technologies

Countermeasures to Fault Attacks

- Several approaches have been proposed, including:
  - Preventing the attack
  - Detecting the fault injection
  - Detecting the fault effect (error)
  - De-synchronization
  - Robust package, “hardened” technologies

Fault detection

- Fault detectors:
  - Laser/light sensors
  - Bulk current sensors
  - Glitch detectors
- They can generate false positives:
  - Because of noise
  - Because the fault does not generate an error

Countermeasures to Fault Attacks

- Several approaches have been proposed, including:
  - Preventing the attack
  - Detecting the fault injection
  - Detecting the fault effect (error)
  - De-synchronization
  - Robust package, “hardened” technologies

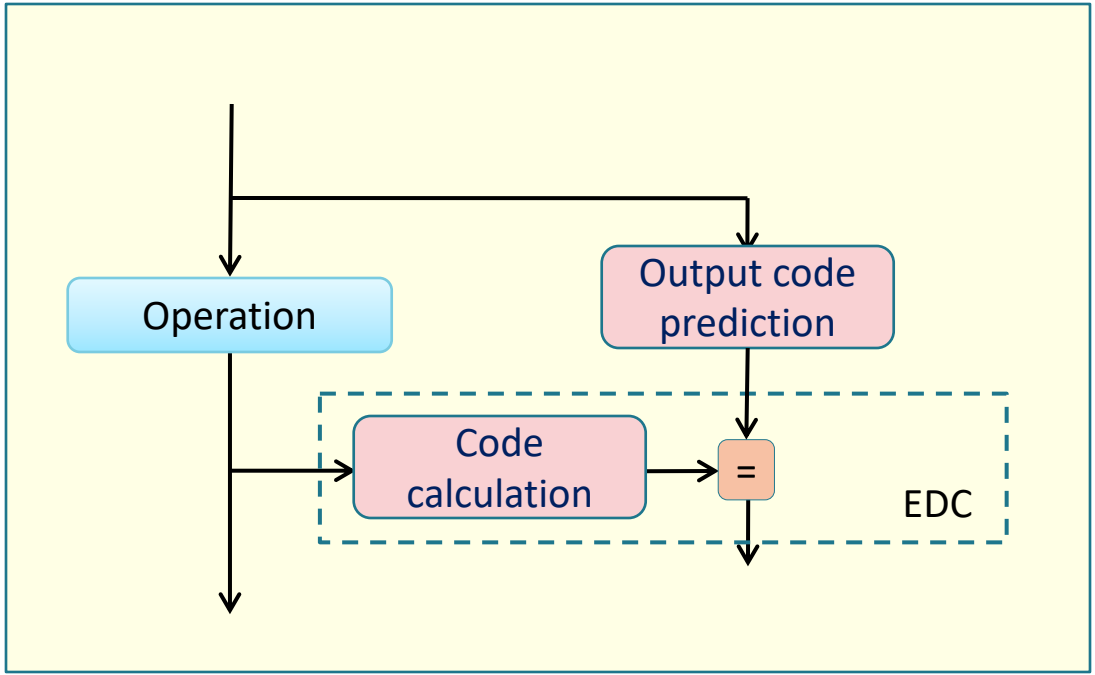
Important remark

- Safety-pushed solutions (e.g., all the solutions adopted to implement Fault Tolerant Architectures, aimed at “tolerating” i.e., detecting, correcting or simply making ineffective faults that could occur) are valuable countermeasures against Security-induced Fault attack !!

Error detection

- Based on redundancy
  - Space redundancy
  - Information redundancy
  - Time redundancy
- Examples:
  - Performing twice the same operation and comparing the results
  - Performing the operation and its inverse, and checking

Error Detection Codes



Countermeasures to Fault Attacks

- Several approaches have been proposed, including:
  - Preventing the attack
  - Detecting the fault injection
  - Detecting the fault effect (error)
  - De-synchronization
  - Robust package, “hardened” technologies

De-synchronization

- A fault injection requires a precise timing to be effective
- Adding temporal randomness makes the timing of the fault harder to set
- How to:
  - Jittered clock
  - Dummy instructions
  - Randomized operation flow

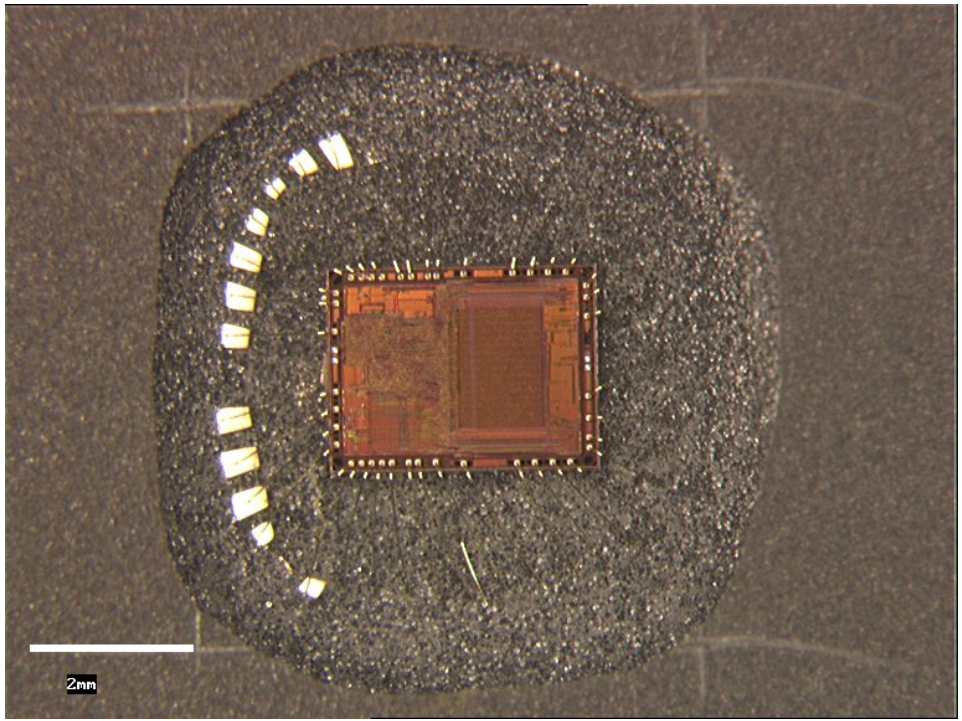
Countermeasures to Fault Attacks

- Several approaches have been proposed, including:
  - Preventing the attack
  - Detecting the fault injection
  - Detecting the fault effect (error)
  - De-synchronization
  - Robust package, “hardened” technologies

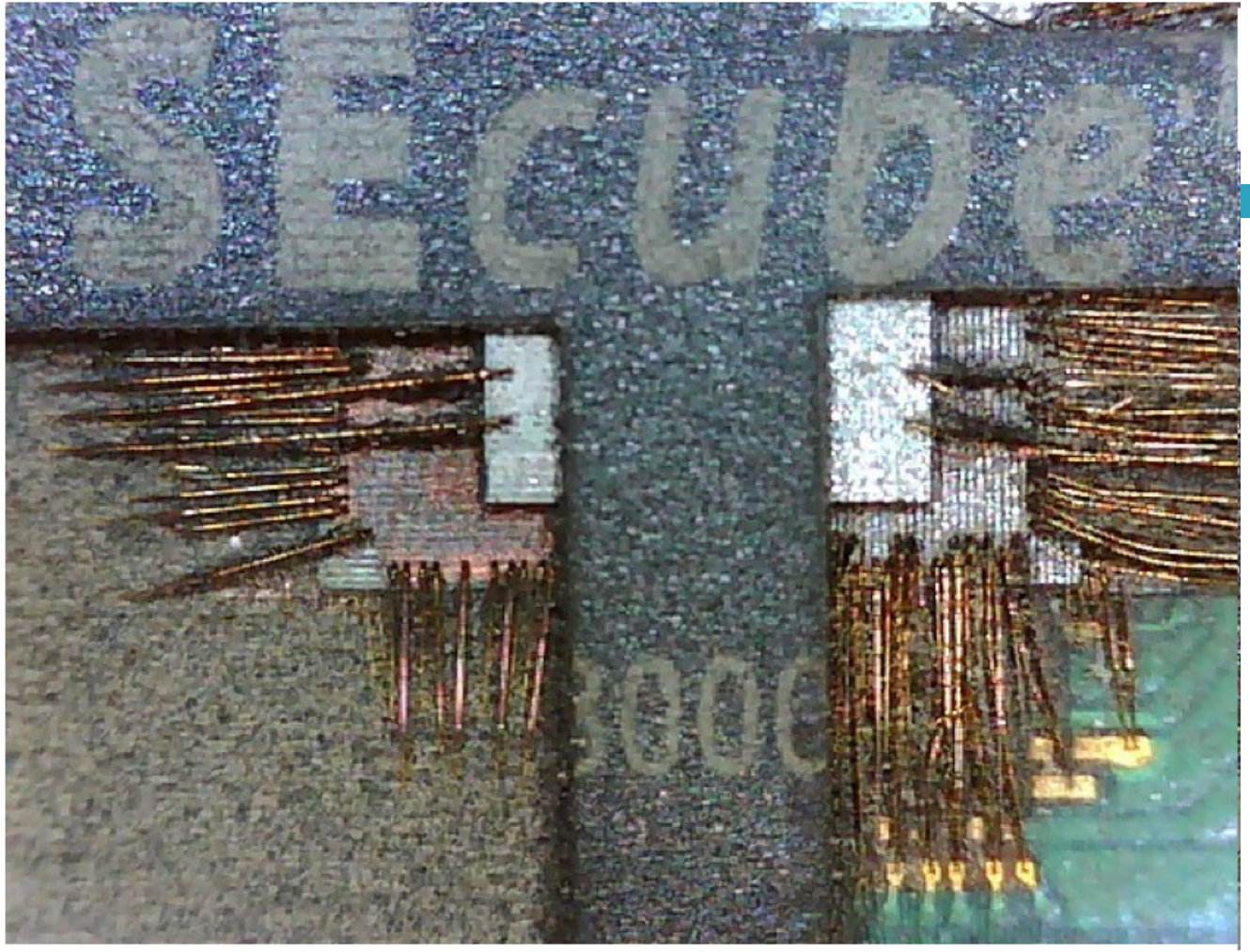
Hardened IC Package

- Several fault injection techniques require the die of the IC to be exposed
  - FIB, laser, …
- Hardening the package can be a countermeasure:
  - Smartcards are easy to open
  - Metallic package can be opened mechanically
  - Epoxy packages require chemical attacks
  - System-in-Package are very hard to open

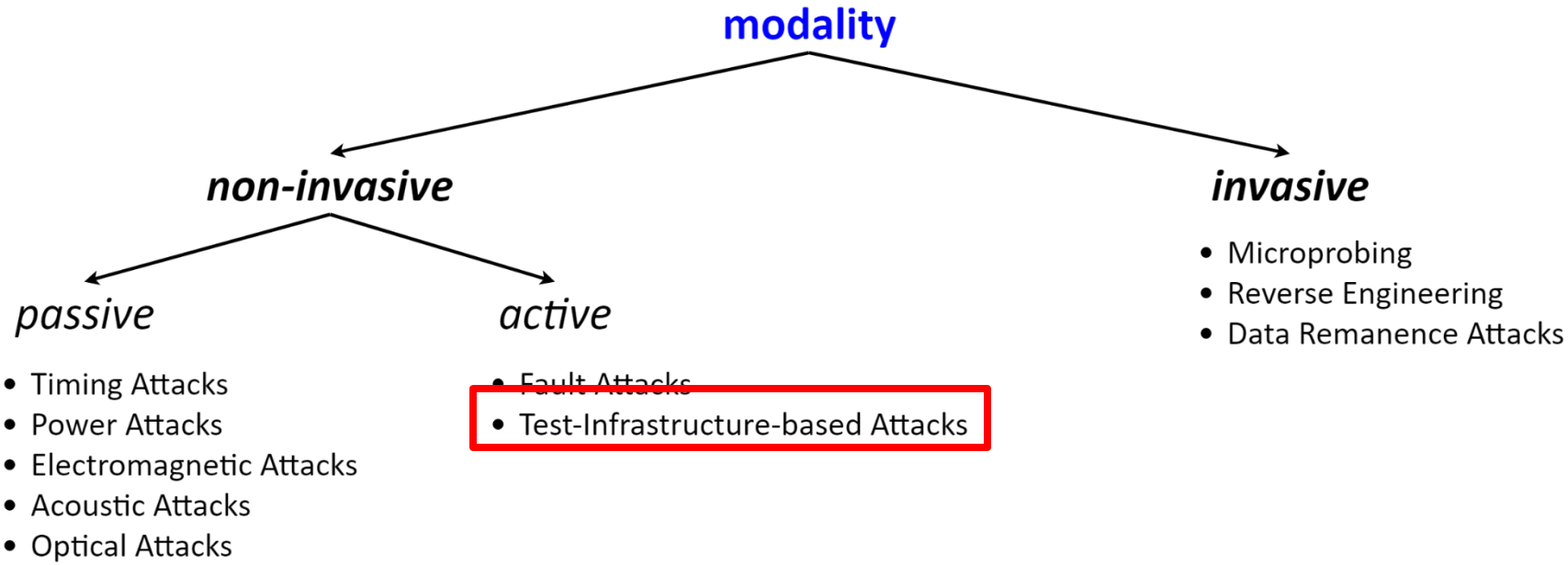
Example: Epoxy package opened with nitric acid



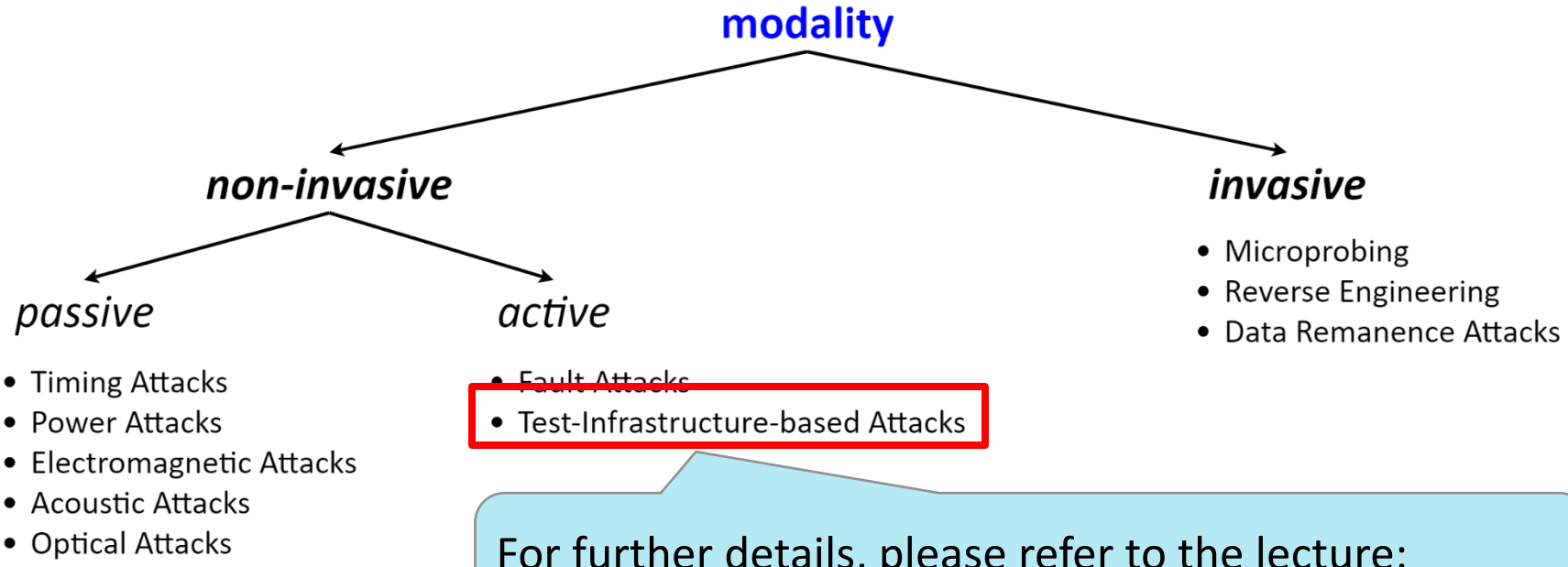
3D SiP



Hardware Attacks Modalities

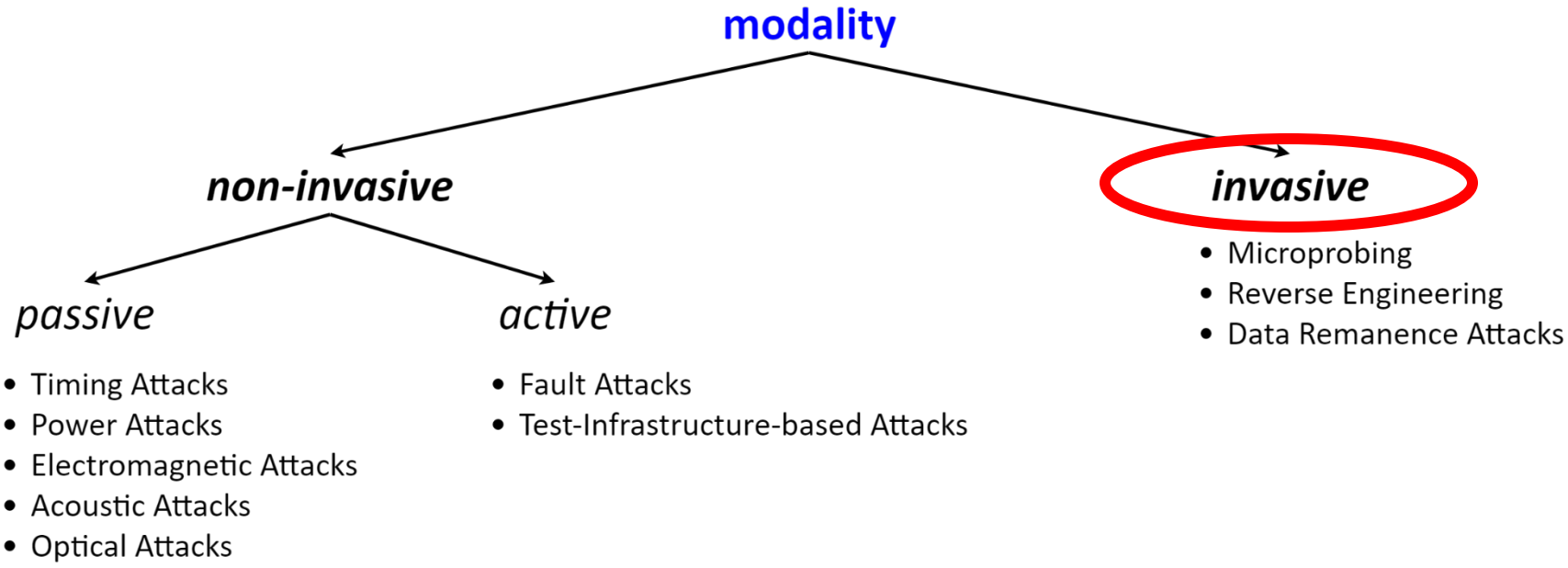


Hardware Attacks Modalities

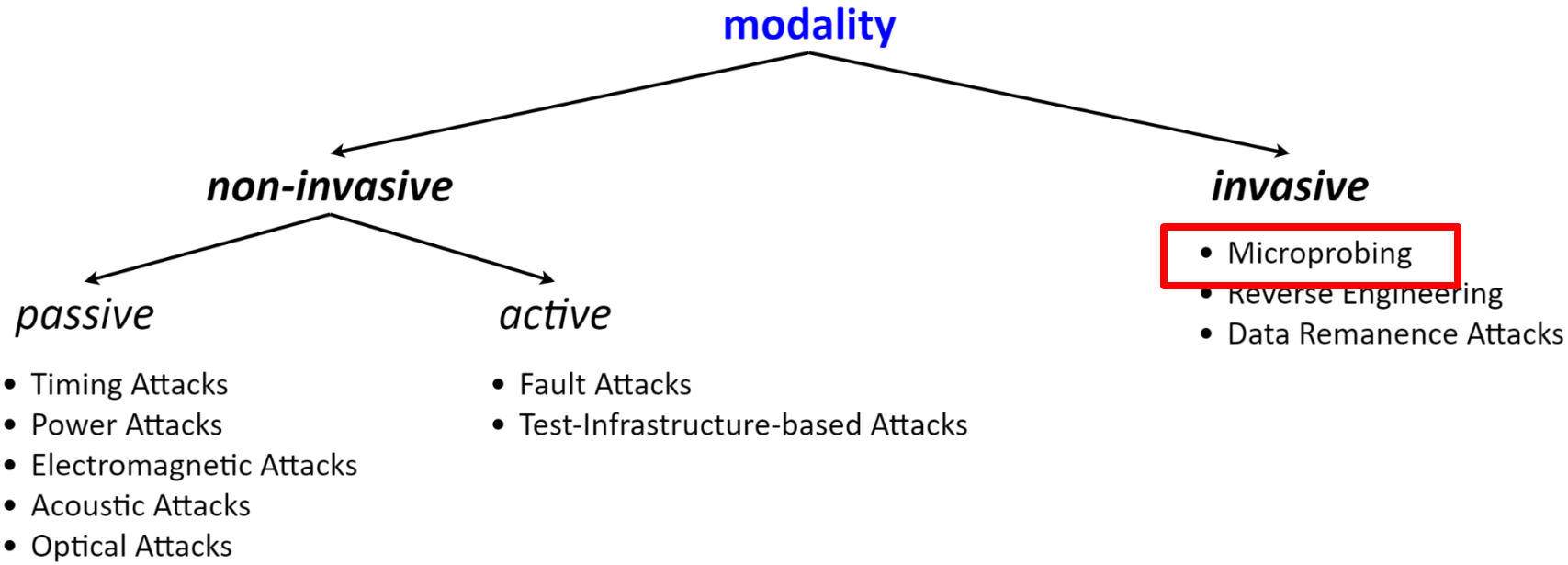


For further details, please refer to the lecture:
HS_2.1 - Vulnerabilities in test infrastructures

Hardware Attacks Modalities



Hardware Attacks Modalities



Microprobing

What

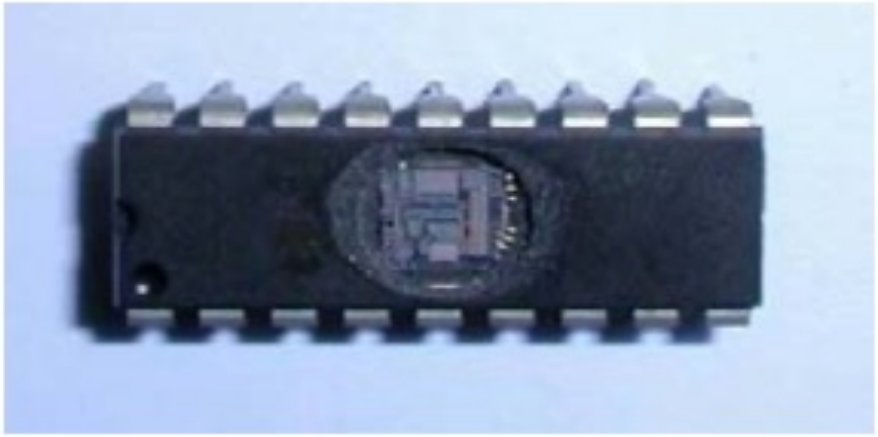
- Tries to extract information by measuring electrical quantities directly on the silicon die of the target device, once obtained physical access to it.

How

- The die exposition is usually achieved by removing the plastic packages via chemical etching and/or by mechanical approaches.

Example of depackaging

- Microcontroller PIC16F84 original and depackaged

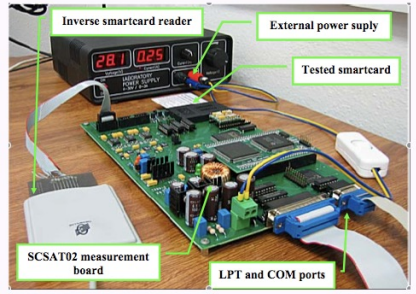


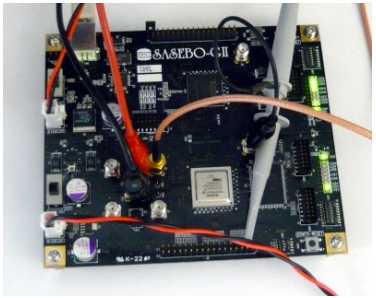
[B.S. Kaliski Jr. et al. (Eds.): CHES 2002, LNCS 2523, pp. 4, 2003]

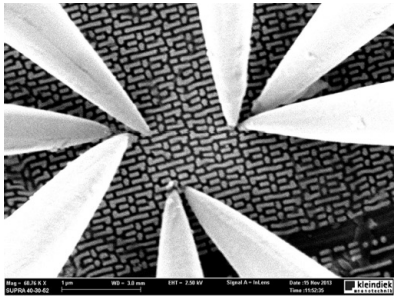
Implementation Attacks

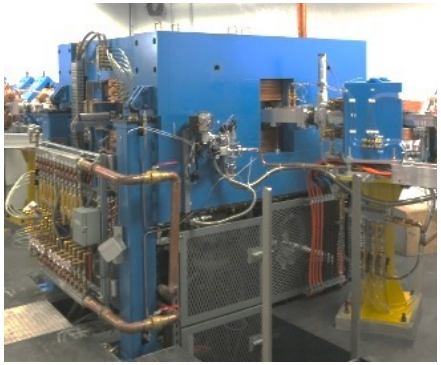
- Microprobing-based attacks can, in turn, be
  - *Passive*: e.g., using microprobes to monitor a smartcard's bus while a program is executing:
  - *Active*: in this case signals may be also injected, the classic example being the use of a grounded microprobe needle on the clock line to the instruction latch to disable jump instructions.

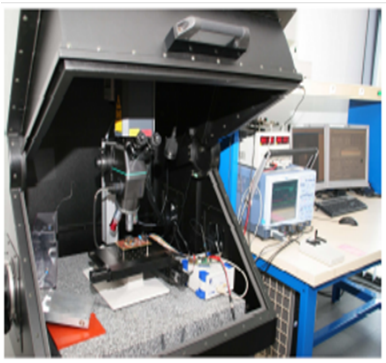
Implementation Attacks











Example: Sat-TV Decoder

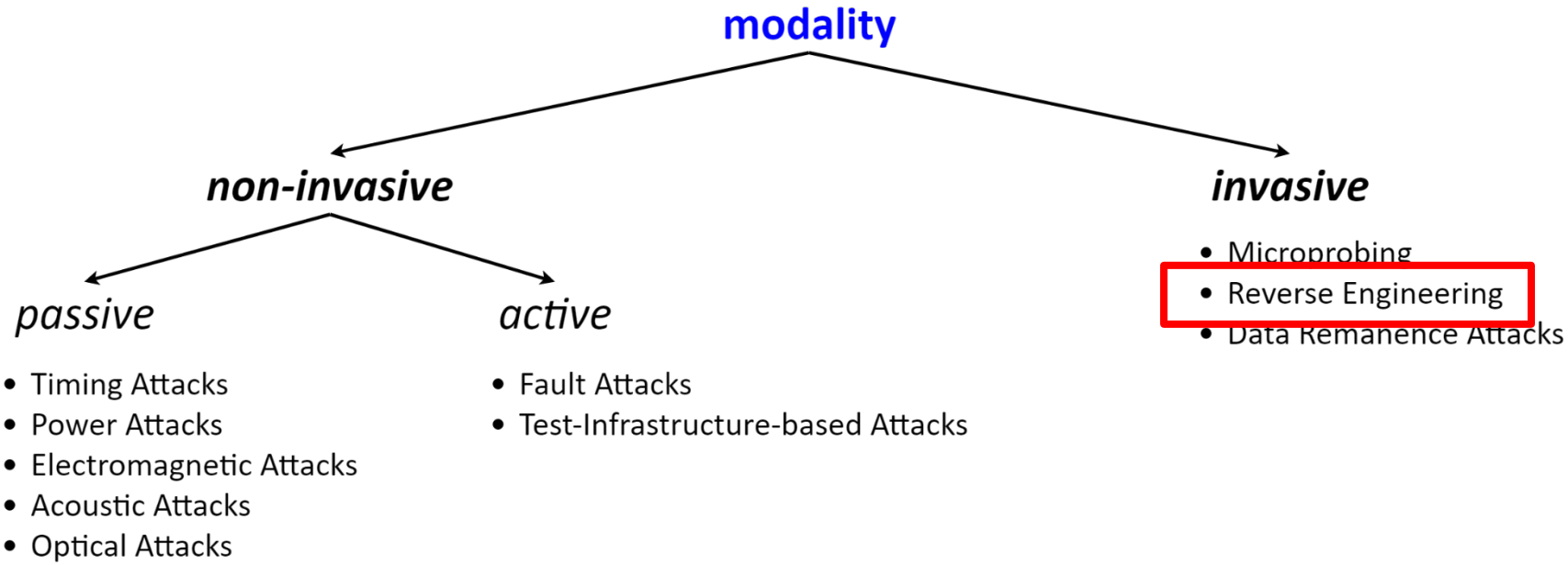
- Let’s have a look at this video on YouTube:
  “How to Reverse-Engineer a Satellite TV Smart Card”

https://www.youtube.com/watch?v=tnY7UVyaFiQ

CYB
CHA

CYBERSECURITY
NATIONAL
LABORATORY

Hardware Attacks Modalities



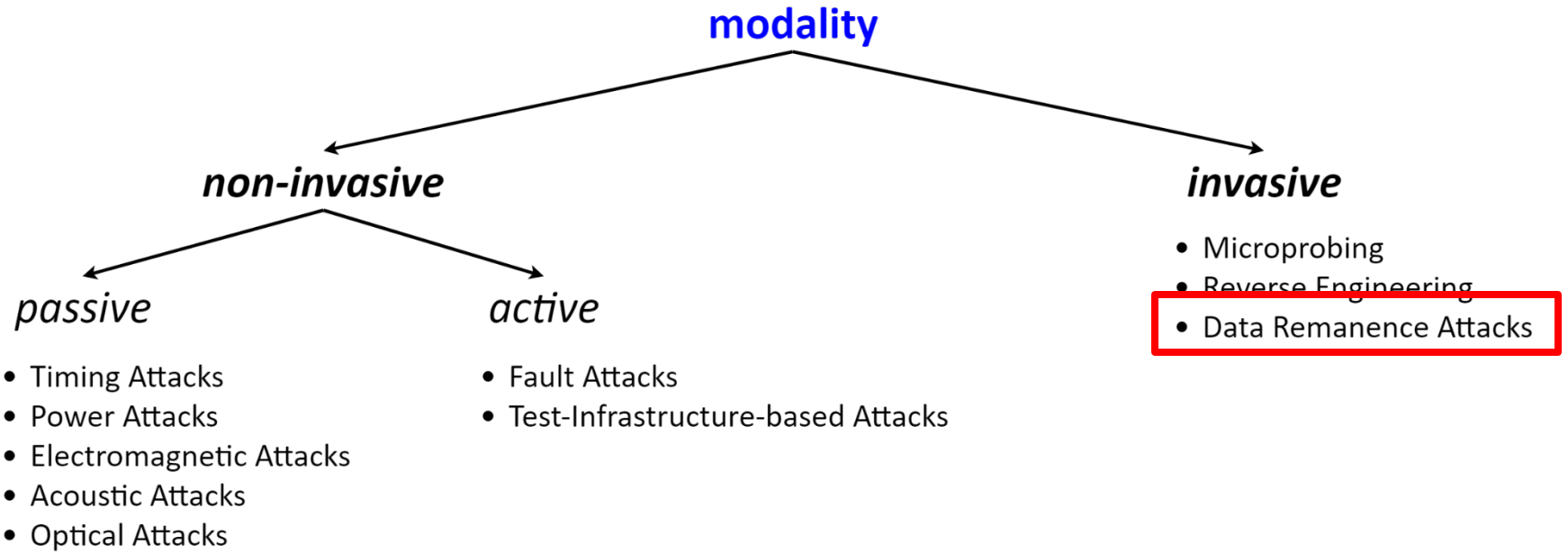
Reverse Engineering (RE) Attacks

- They aim at understanding the structure of a semiconductor device and its functions, i.e., at stealing the IP - *Intellectual Properties* - of the design

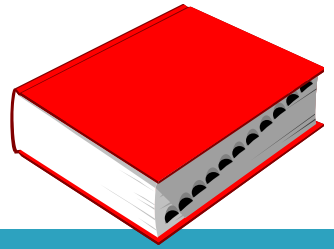
Advanced tools for RE

- A very interesting presentation on advanced tools for RE can be found here:
  - https://media.hardwear.io/integrated-circuit-offensive-security-olivier-thomas/

Hardware Attacks Modalities



Data remanence



- Is the residual information remaining on storage media after clearing, i.e., after the foreseen actions to remove or erase the target data.

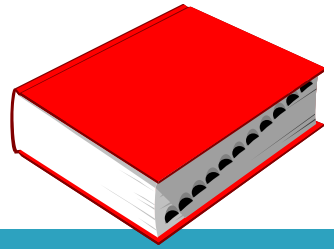
Example of residues

- This residue may result from:
  - data being left intact by a nominal file deletion operation
  - by reformatting of storage media that does not remove data previously written to the media
  - through physical properties of the storage media that allow previously written data to be recovered

Data Remanence Attacks

- An attacker with physical access to a computing system may just disconnect the memory chips and then retrieve their content at disconnection time by analyzing them off-line

Cold boot attack



- A cold boot attack is a process for obtaining unauthorized access to a computer's encryption keys when the computer is left physically unattended.

Consequences

- Cold boot attack

Cold boot attack

- Contrary to popular assumption, DRAMs used in most modern computers retain their contents for several seconds after power is lost, even at room temperature and even if removed from a motherboard.
- Although DRAMs become less reliable when they are not refreshed, they are not immediately erased, and their contents persist sufficiently for malicious (or forensic) acquisition of usable full-system memory images.

[J. Alex Halderman et al:
"*Lest We Remember: Cold Boot Attacks on Encryption Keys*"
in Proc. 2008 USENIX Security Symposium]

- This video from the Center for Information Technology Policy demonstrates how a cold boot attack works:
  - https://www.youtube.com/watch?v=JDaicPIgn9U

Lest We Remember:

Cold Boot Attacks on Encryption Keys

citp.princeton.edu/memory

Important remark on volatility

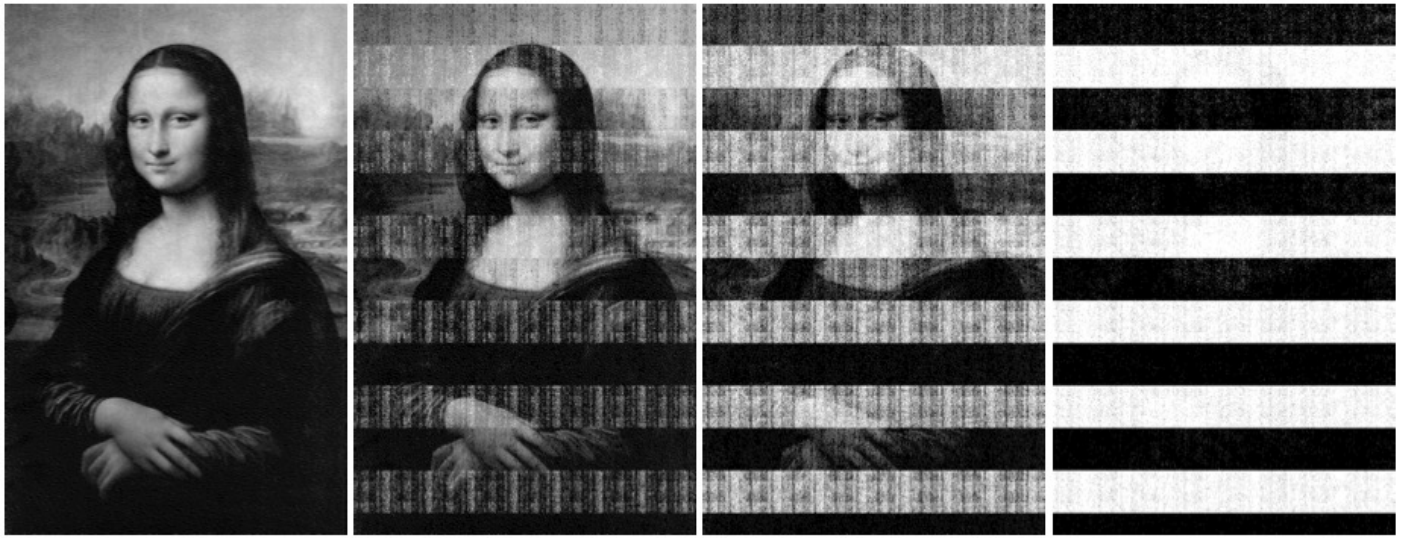


Figure 4: We loaded a bitmap image into memory on Machine A, then cut power for varying lengths of time. After 5 seconds (left), the image is indistinguishable from the original. It gradually becomes more degraded, as shown after 30 seconds, 60 seconds, and 5 minutes.

Important remark on volatility

Figure 5: Before powering off the computer, we spray an upside-down canister of multipurpose duster directly onto the memory chips, cooling them to −50 °C. At this temperature, the data will persist for several minutes after power loss with minimal error, even if we remove the DIMM from the computer.

Conclusions

- Cryptography has 2,000+ years history and experience
- Hardware Security is still a young research field…

Paolo PRINETTO

Director
CINI Cybersecurity
National Laboratory

Paolo.Prinetto@polito.it

Mob. +39 335 227529

CYBER CHALLENGE.IT

CYBERSECURITY NATIONAL LABORATORY

https://cybersecnatlab.it