CYBER CHALLENGE
CyberChallenge.IT

cini
Cybersecurity National Lab

SPONSOR PLATINUM

accenture

aizoon AUSTRALIA EUROPE USA
TECHNOLOGY CONSULTING

B5

eni

exprivia | ITALTEL

IBM

KPMG

LEONARDO

NTT DATA
Trusted Global Innovator

NUMERA
SISTEMI E INFORMATICA S.p.A.

Telsy

SPONSOR GOLD

bip.

cisco

MONTE DEI PASCHI DI SIENA
BANCA DAL 1472

negg

NOVANEXT
connecting the future

pwc

SPONSOR SILVER

DGi ONE
the leading digital company

ICT CYBER CONSULTING

Hardware Vulnerabilities

Paolo PRINETTO

Director
CINI Cybersecurity
National Laboratory

Paolo.Prinetto@polito.it

Mob. +39 335 227529

CYBER
CHALLENGE
CyberChallenge.IT

cini
Cybersecurity
National Lab

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.

Prerequisites

- Lecture:
  - *CS_1.3 - Security Pillars*

Acknowledgments

- The presentation includes material from
  - Rocco DE NICOLA
  - Michele LORETI
  - Nicolò MAUNERO
  - Gianluca ROASCIO

  whose valuable contribution is here acknowledged and highly appreciated.

Goal

- Introducing a taxonomy of Vulnerabilities, clustering them according to their *Nature*, *Domain*, and *Source*
- Presenting several examples, focusing on hardware related ones.

Outline

- Weakness vs. Vulnerability
- Vulnerability Taxonomy:
  - Nature
  - Domain
  - Source

Outline

- Weakness vs. Vulnerability
- Vulnerability Taxonomy:
  - Nature
  - Domain
  - Source

“ Der Teufel steckt im Detail ”

- Information Systems are complex systems
- Complexity means a lot of *details*
- Some of these details may present *weaknesses*, turning out into *vulnerabilities*

Weakness

- General characteristic referred to systems or system components which determines their *exposure*, i.e., the possibility of losing *Confidentiality*, *Integrity,* or *Availability* for their assets.

Vulnerability

- *Particular* weakness present in a *specific* component of a system that can be *exploited* by an *attacker* to carry out unauthorized actions to one's advantage against the *Confidentiality*, the *Integrity* or the *Availability* of the system assets.

Weakness vs. Vulnerability

Weakness

- It is the *class*
- Represents a general problem

Vulnerability

- It is the *instance*
- Represents a *specific* problem of a *specific* version of a *specific* component

Weakness vs. Vulnerability

Weakness

- It is the *class*
- Represents a general problem
  - E.g., **CWE-122**:
    Heap-based Buffer Overflow

Vulnerability

- It is the *instance*
- Represents a specific problem of a specific version of a specific component
  - E.g., **CVE-2019-6778**:
    In QEMU 3.0.0, tcp_emu in slirp/tcp_subr.c has a heap-based buffer overflow.

vs.

CWE™

- *Common Weakness Enumeration* is a community-developed list of common software and hardware security weaknesses. It serves as a common language, a measuring stick for security tools, and as a baseline for weakness identification, mitigation, and prevention efforts

[https://cwe.mitre.org]

CVE®

- *Common Vulnerabilities and Exposures* is a list of entries - each containing an identification number, a description, and at least one public reference - for publicly known cybersecurity vulnerabilities

[https://cve.mitre.org]

Caveat

- Vulnerabilities can impact on both Safety & Security, and thus on Dependability !!

Protecting what

- People
- Environment

SAFETY

- Objects
- Computers
- Information

SECURITY

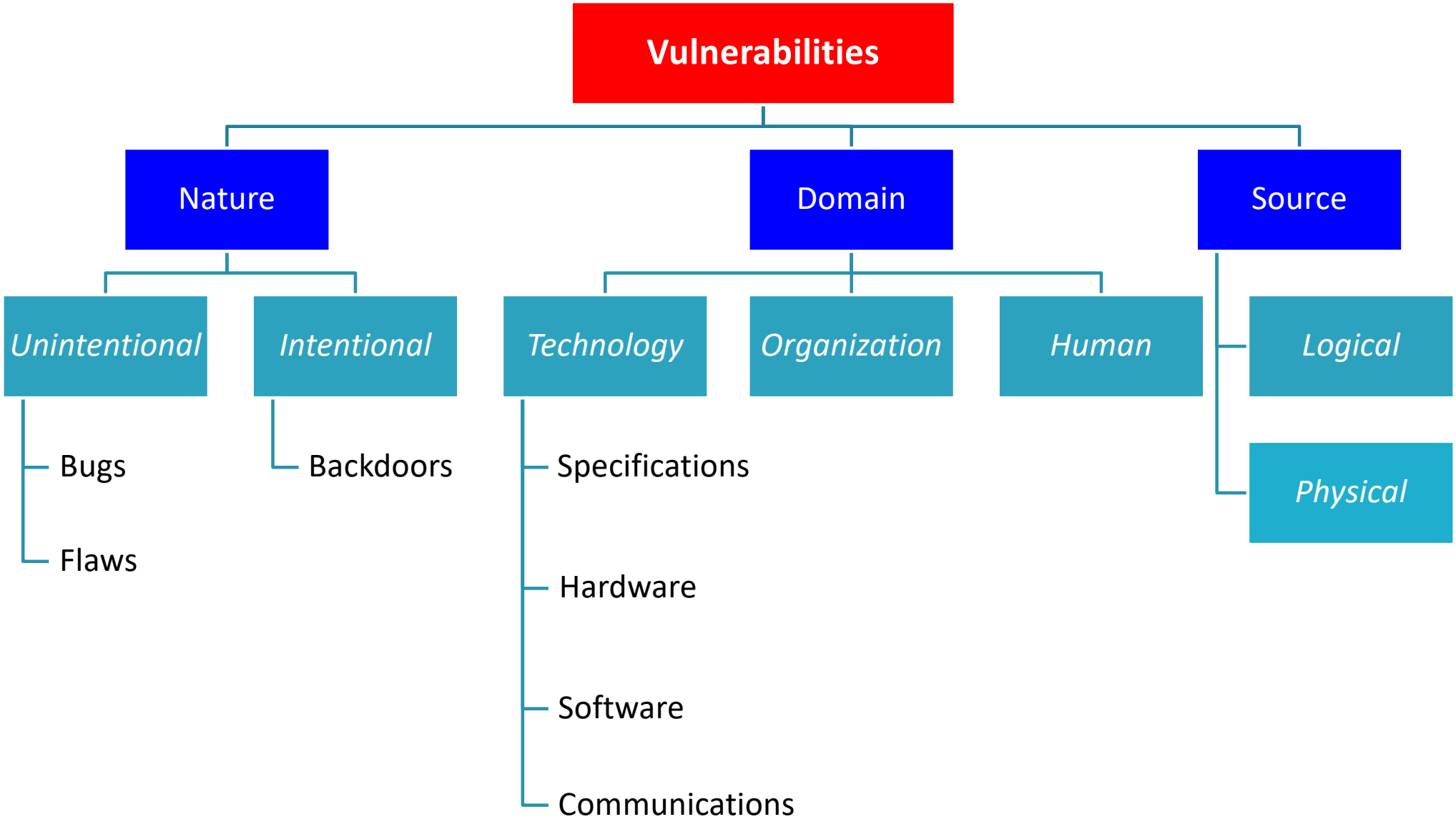
- Cyberspace

CYBERSECURITY

} *DEPENDABILITY*

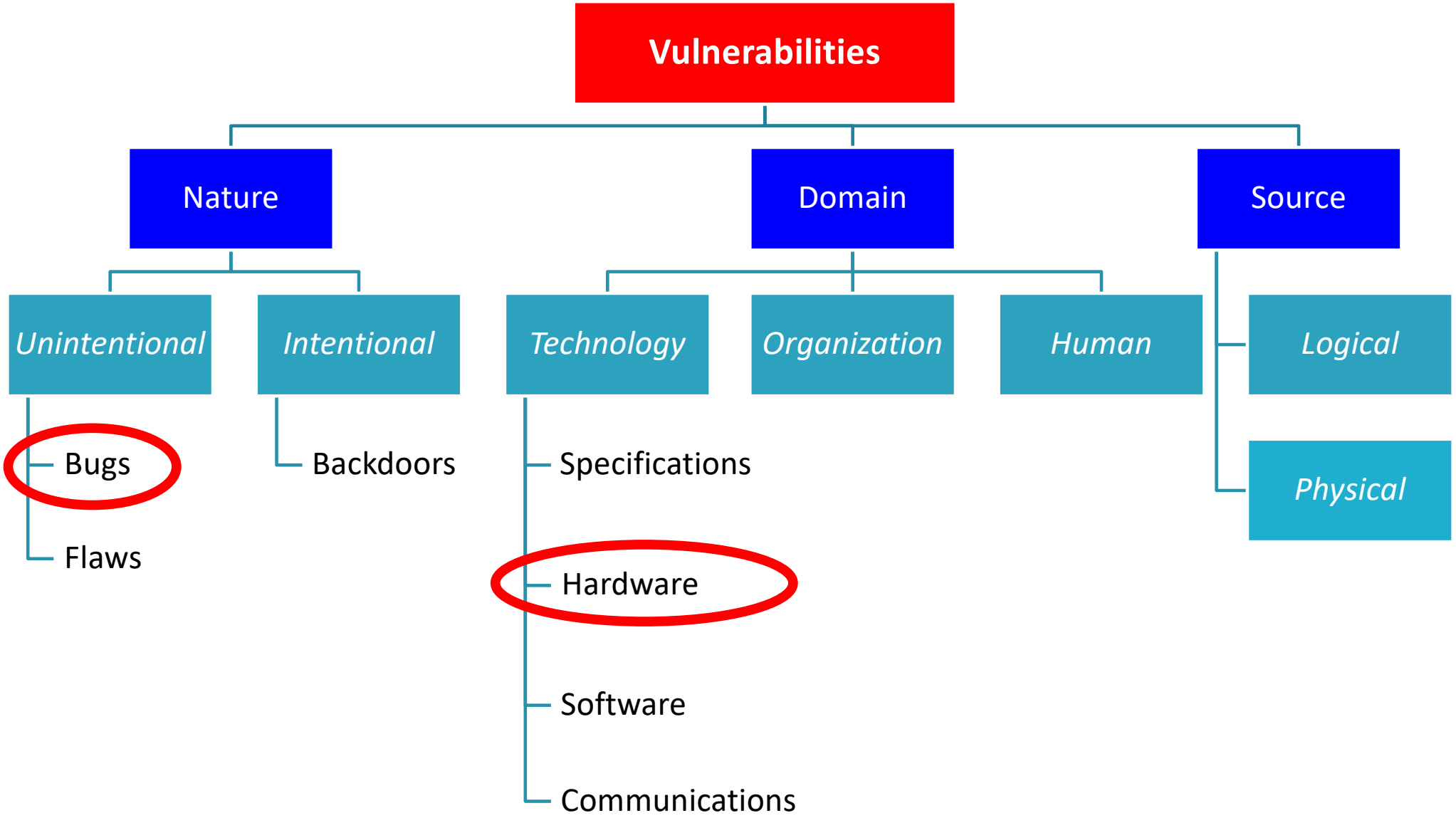
Vulnerabilities

- Can be clustered according to several orthogonal dimensions
- In the sequel we are going to focus on 3 of them:
  - vulnerability *nature*
  - vulnerability *domain*
  - vulnerability *source*

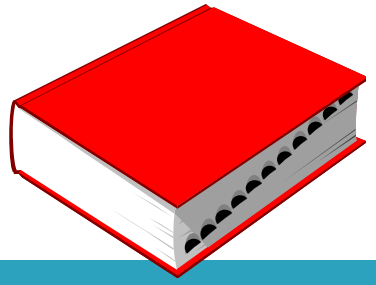

Vulnerabilities
Nature
Domain
Source
Unintentional
Intentional
Technology
Organization
Human
Logical
Physical
Bugs
Flaws
Backdoors
Specifications
Hardware
Software
Communications

Some details…

- In the sequel we shall focus on some significant cases…


Vulnerabilities
Nature
Domain
Source
Unintentional
Intentional
Technology
Organization
Human
Logical
Physical
Bugs
Flaws
Backdoors
Specifications
Hardware
Software
Communications

Hardware Bug



- An inconsistency between a specification and its actual implementation, introduced by mistake during the design and not detected during *Validation & Verification* (V&V) phases.

Example: “F00F Pentium P5 Bug”

- Detected in 1997 in all Pentium P5 processors
- In the x86 architecture, the byte sequence F0 0F C7 C8 represents the instruction lock cmpxchg8b eax (locked compare and exchange of 8 bytes in register EAX) and does not require any special privilege.
- Hoverer, the instruction encoding is invalid. The cmpxchg8b instruction compares the value in the EDX and EAX registers (the lower halves of RDX and RAX on more modern x86 processors) with an 8-byte value in a memory location.
- In this case, however, a register is specified instead of a memory location, which is not allowed.

Example: “F00F Pentium P5 Bug”

- Under normal circumstances, this would simply result in an exception
- However, when used with the lock prefix (normally used to prevent two processors from interfering with the same memory location), the CPU erroneously uses locked bus cycles to read the illegal instruction exception-handler descriptor.
- Locked reads must be followed by locked writes, and the CPU's bus interface enforces this by forbidding other memory accesses until both actions are completed.
- As none are forthcoming (since, due to the locking, write cannot take place), after performing these bus cycles all CPU activity stops, and the CPU must be reset to recover.
- The instruction can be exploited for a DoS attack.

Example: Cyrix coma bug

- The Cyrix coma bug is a design flaw in Cyrix 6x86, 6x86L, and early 6x86MX processors that allows a non privileged program to hang the computer.

Example: The Cyrix coma bug

- This C program (which uses inline x86-specific assembly language) could be compiled and run by an unprivileged user:

```
unsigned char
c[4] = {0x36, 0x78, 0x38, 0x36};

int main()

{

asm (

"         movl $c, %ebx\n"

"again:   xchgl (%ebx), %eax\n"

"         movl %eax, %edx\n"

"         jmp again\n"

      );

}
```

Example: The Cyrix coma bug

- On executing this program, the processor enters an infinite loop that cannot be interrupted.
- This allows any user with access to a Cyrix system with this bug to perform a DoS attack.

Bug sources

- Bugs stem from several sources, including:
  - People
  - Procedures
  - Tools

Bug sources

- Bugs stem from several sources, including:
  - People
  - Procedures
  - Tools
- During the design & production phase:
  - Inexperience of designers, developers, test engineers:
    - Design errors
    - Insufficient Validation & Verification (V&V) phases
    - Insufficient test coverage
- In-the-field:
  - misuse

About “misusage”

- Operator error is one of the most common cause of failure
- Nevertheless many errors attributed to operators are actually caused by designs that require an operator to choose an appropriate recovery action without much guidance and without any automated help.

To read

 Rel. 18.05.2020

Bug sources

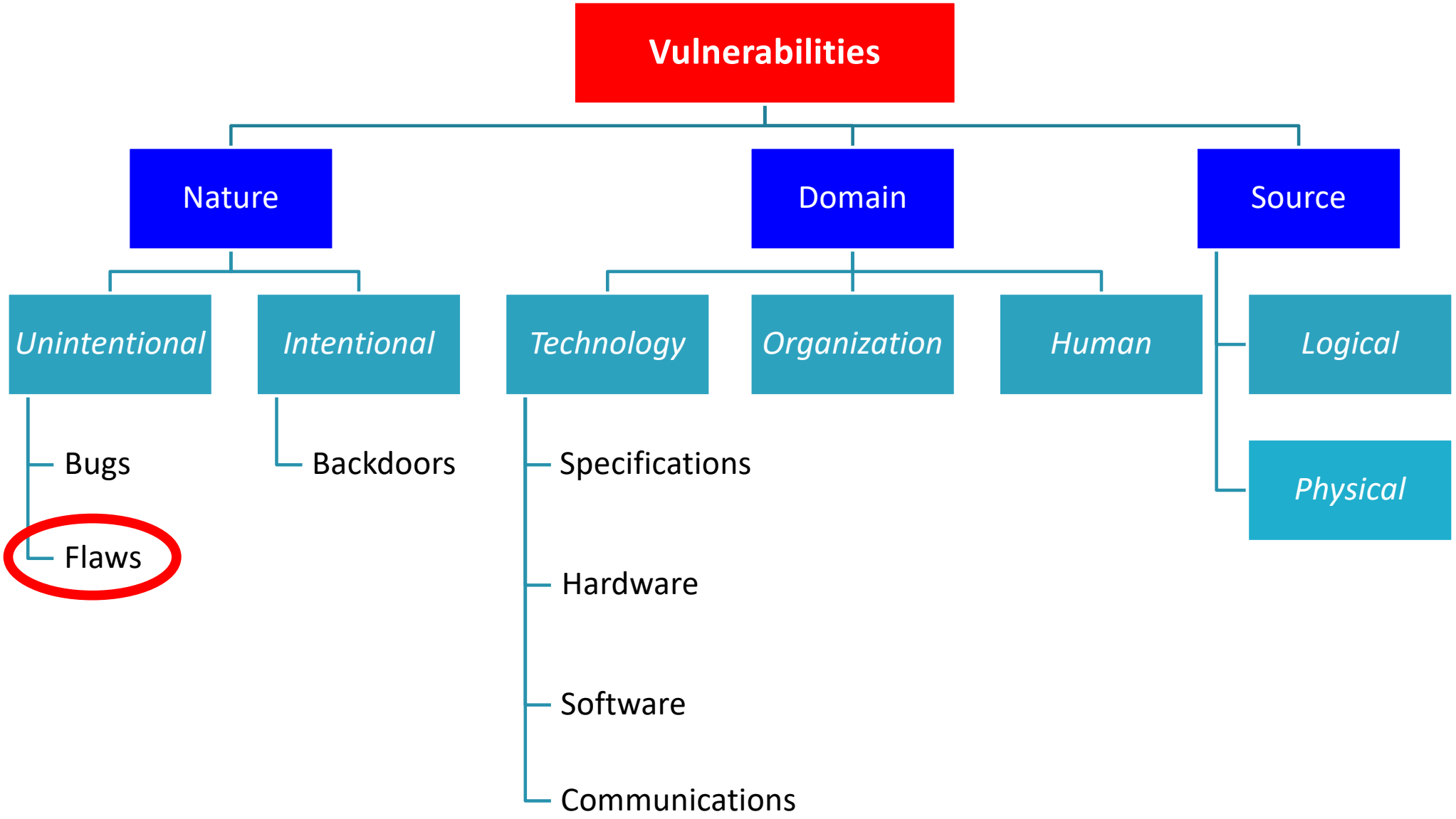
- Bugs stem from several sources, including:
  - People
  - Procedures
  - Tools
- Mistakes in:
  - Design rules
  - Design methodologies
  - V&V methodologies
  - Test methodologies
- Lack of compliance checking w.r.t.:
  - Design & V&V methodologies
  - Adopted standards

Bug sources

- Bugs stem from several sources, including:
  - People
  - Procedures
  - Tools
- Adopted tools could be
  - Inappropriate
  - Bugged

Bug remediations

- Additional investments in terms of:
  - People
  - Procedures
  - Tools


Vulnerabilities
Nature
Domain
Source
Unintentional
Intentional
Technology
Organization
Human
Logical
Physical
Bugs
Flaws
Backdoors
Specifications
Hardware
Software
Communications

Flaw

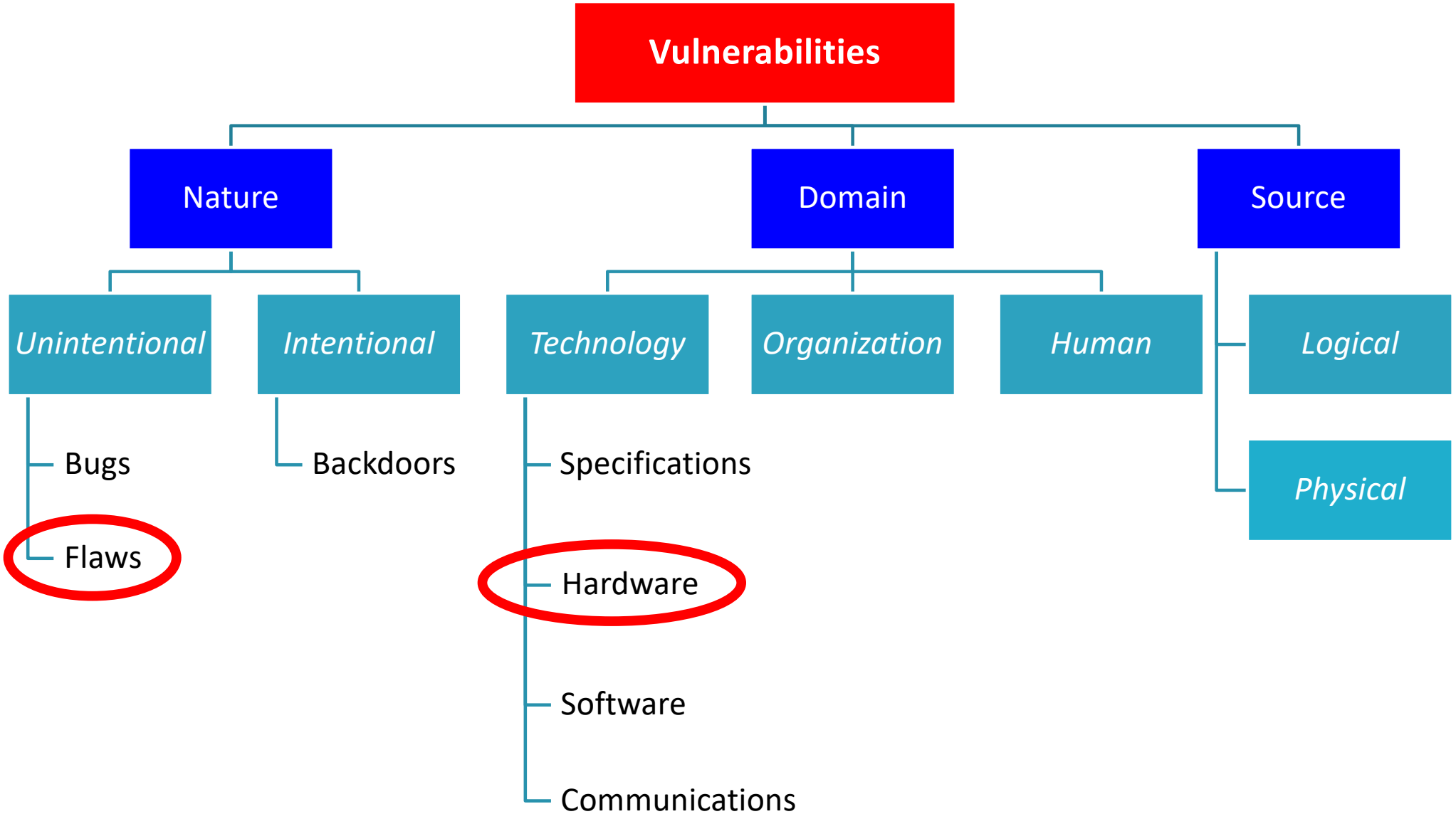
- A non-primary feature that does not constitute an inconsistency w.r.t. the specifications, resulting from a misconception of the designer who did not take into consideration its potential dangerousness.

Flaw causes

- Ignorance of security issues
- Insufficient covering of potentially malicious use cases
- Priority given to the optimization of other design dimensions, such as
  - Ease of use
  - Performances
  - …

Flaw remediations

- Additional investments in terms of:
  - People:
    - Security-oriented training and continuous education
  - Procedures:
    - Security as requirement
    - Security-oriented development process
    - Security-by-Design
    - Extensive VAPT (Vulnerability Assessment & Penetration Testing) campaigns


Vulnerabilities
Nature
Domain
Source
Unintentional
Intentional
Technology
Organization
Human
Logical
Physical
Bugs
Flaws
Backdoors
Specifications
Hardware
Software
Communications

Hardware Flaw: example

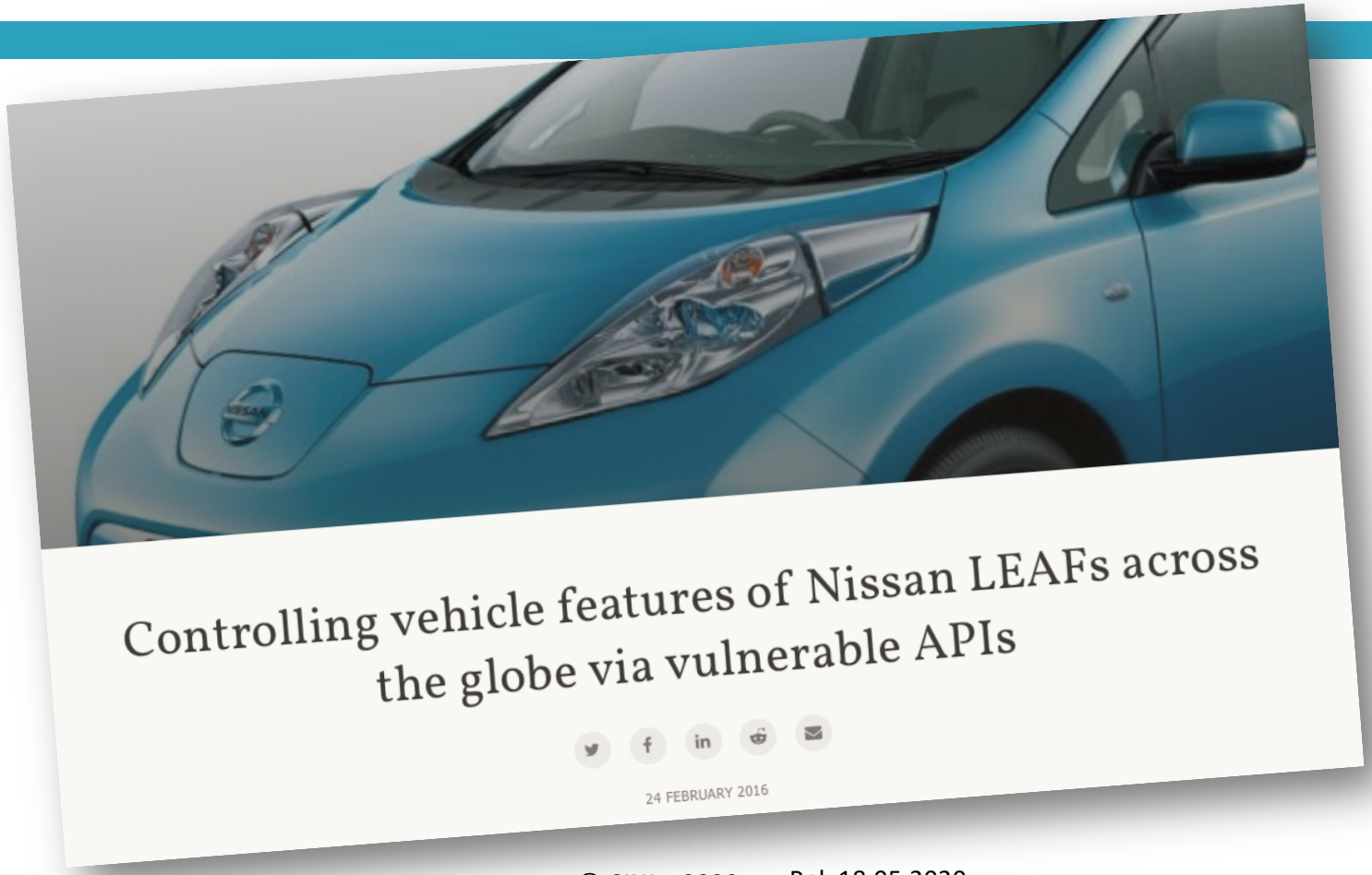
- Speculative Execution in modern processors
  - On branch instructions, both branches are executed before condition check
  - At commit time, only the correct execution is validated
- Great for performance, but …
  - Commitment does not delete completely non-valid path
  - Traces of discarded execution may leak information

Examples: Microarchitectural Flaws

- Processors do not enter an error state but reveal private information!
- They usually allow a concurrent (aggressor) program to fraudulently access private data and keys of a victim program

- Spectre (2018)
- Meltdown (2018)
- Foreshadow (2018)
- ZombieLoad (2018)
- Spoiler (2019)

Examples: at the “system” level

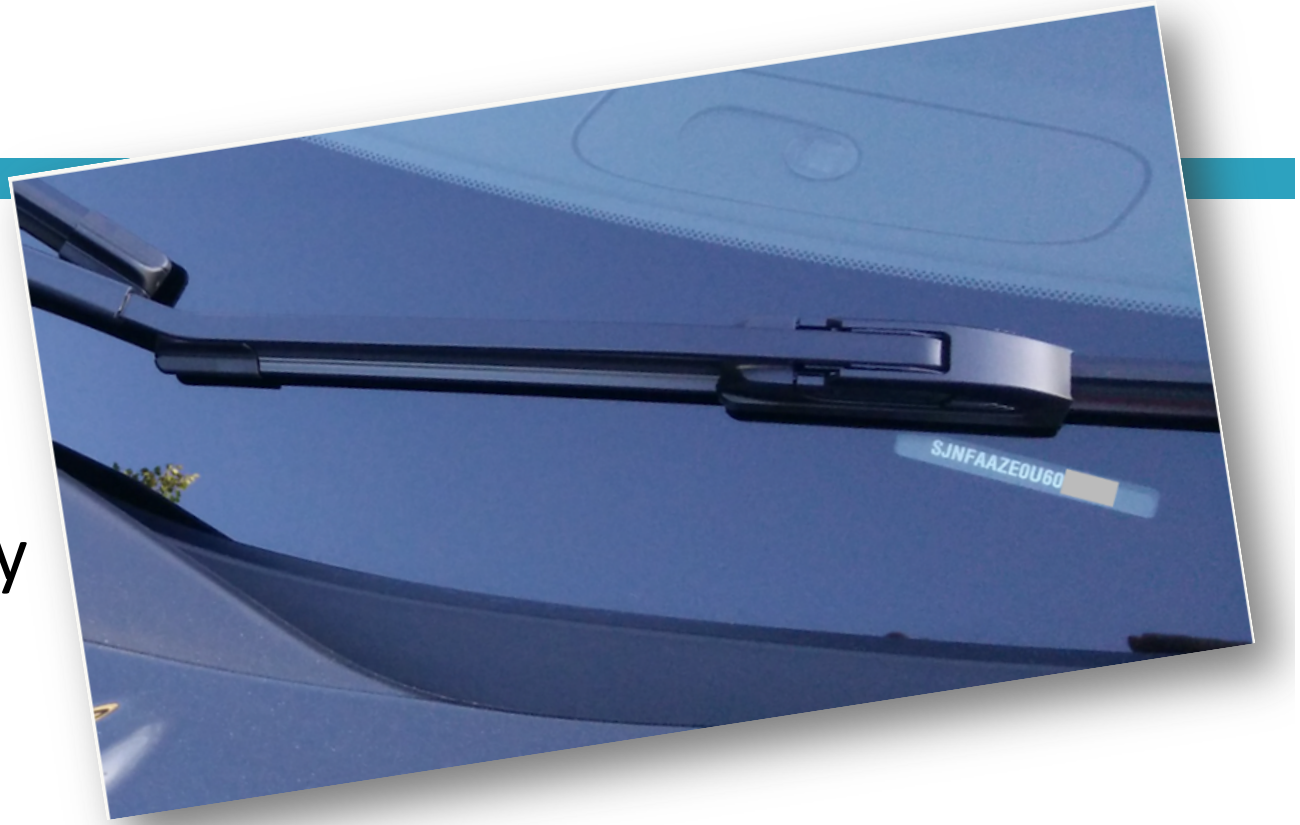


Controlling vehicle features of Nissan LEAFs across the globe via vulnerable APIs

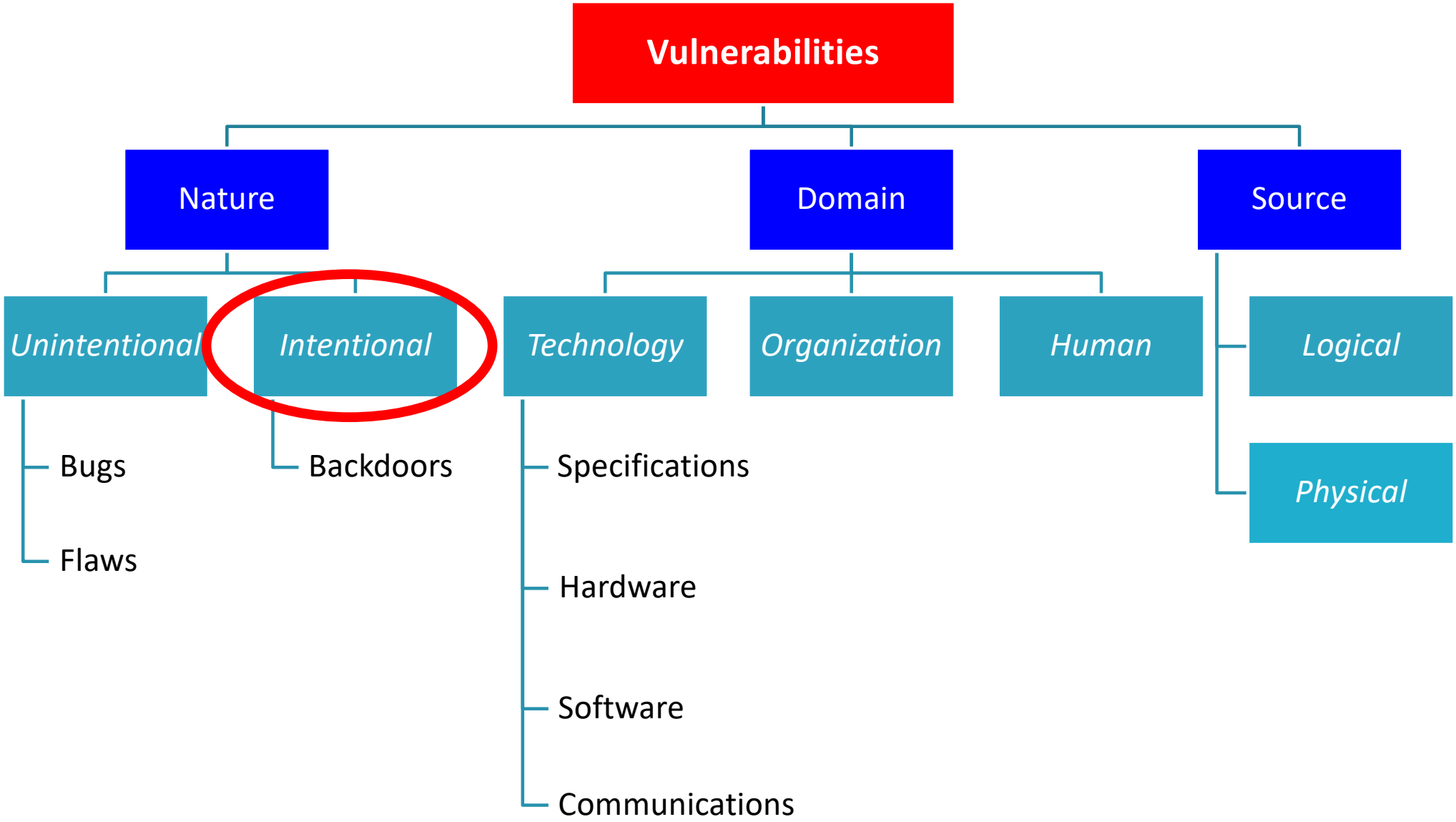
24 FEBRUARY 2016

How

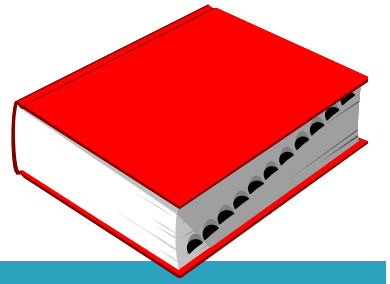
An attack was
possible exploiting the
Vehicle Identification
Number which uniquely
identifies
the chassis
of the car



https://www.youtube.com/watch?v=Nt33m7G_42Q


Vulnerabilities
Nature
Domain
Source
Unintentional
Intentional
Technology
Organization
Human
Logical
Physical
Bugs
Flaws
Backdoors
Specifications
Hardware
Software
Communications

Intentional Vulnerabilities



- When a vulnerability is inserted intentionally, it can be referred to as a *backdoor*, as the person who inserts it wants to guarantee her/himself/someone else the possibility of a later access, or use, that is *outside* the set of intended use cases.

Intentional Vulnerabilities

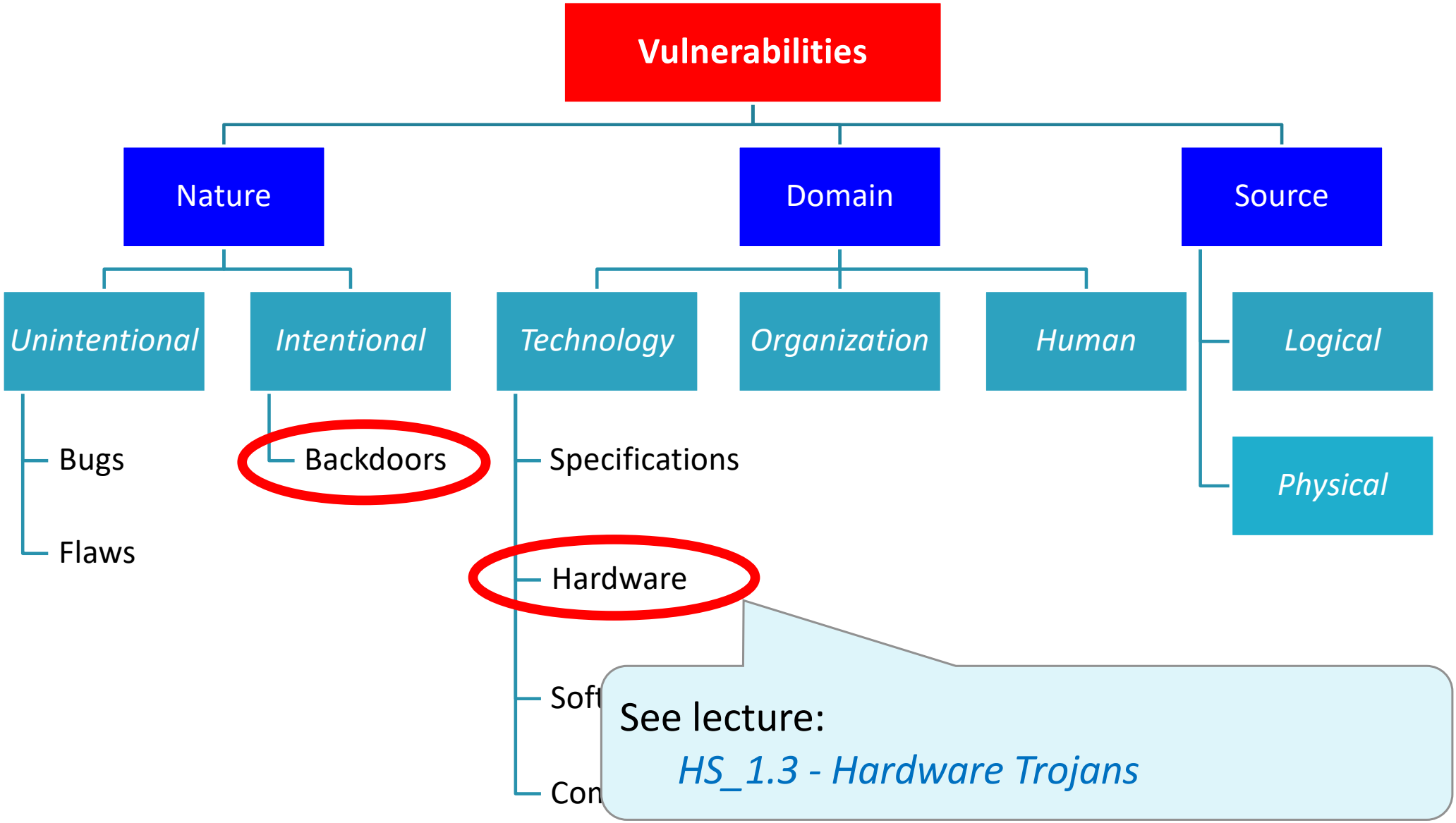
- A backdoor is always a vulnerability, even if designers did not insert it to harm the system

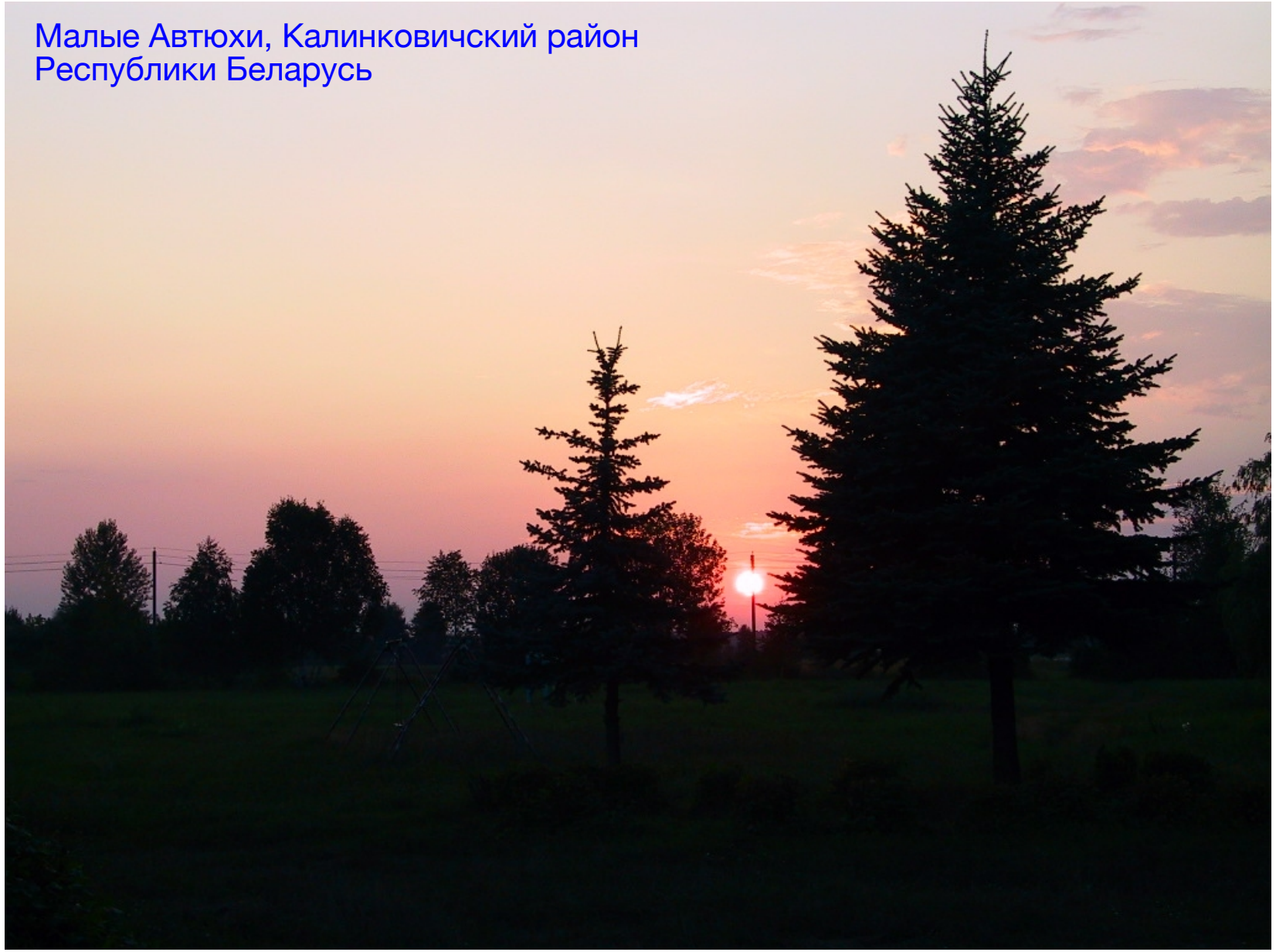
Backdoor causes/motivations

- Untrusted stakeholders of the supply chain
- Untrusted players of the design process
- Insertion for remote debugging or in-field maintenance

Backdoor remediations

- How to trust EVERY ring of a supply-chain?
  - Still an open issue
  - Additional investments in terms of research are needed
- Securing remote update and maintenance without using (relying on?) open backdoors


Vulnerabilities
Nature
Domain
Source
Unintentional
Intentional
Technology
Organization
Human
Logical
Physical
Bugs
Flaws
Backdoors
Specifications
Hardware
Soft
Con
See lecture:
HS_1.3 - Hardware Trojans



Малые Автюхи, Калинковичский район
Республики Беларусь

Paolo PRINETTO

Director
CINI Cybersecurity
National Laboratory

Paolo.Prinetto@polito.it

Mob. +39 335 227529

CYBER CHALLENGE
CyberChallenge.IT

cini Cybersecurity National Lab

https://cybersecnatlab.it

CYBER CHALLENGE
CyberChallenge.IT

cini
Cybersecurity National Lab

SPONSOR PLATINUM

accenture

aizoon AUSTRALIA EUROPE USA
TECHNOLOGY CONSULTING

B5

eni

exprivia | ITALTEL

IBM

KPMG

LEONARDO

NTT DATA
Trusted Global Innovator

NUMERA
SISTEMI E INFORMATICA S.p.A.

Telsy

SPONSOR GOLD

bip.

cisco

MONTE DEI PASCHI DI SIENA
BANCA DAL 1472

negg

NOVANEXT
connecting the future

pwc

SPONSOR SILVER

DGi ONE
the leading digital company

ICT CYBER CONSULTING