

Colloque « Systèmes embarqués, sécurité et sûreté de fonctionnement »



Smart On Smart
ANR-07-SESU-014-01

M. Agoyan, P. Bazargan-Sabet, K. Bekkou,
S. Bouquet, S. Le Henaff, E. Lepavec,
M-H. NGuyen, G. Phan, B. Robisson,
P. Soquet, F. Wajsbürt.



Schedule

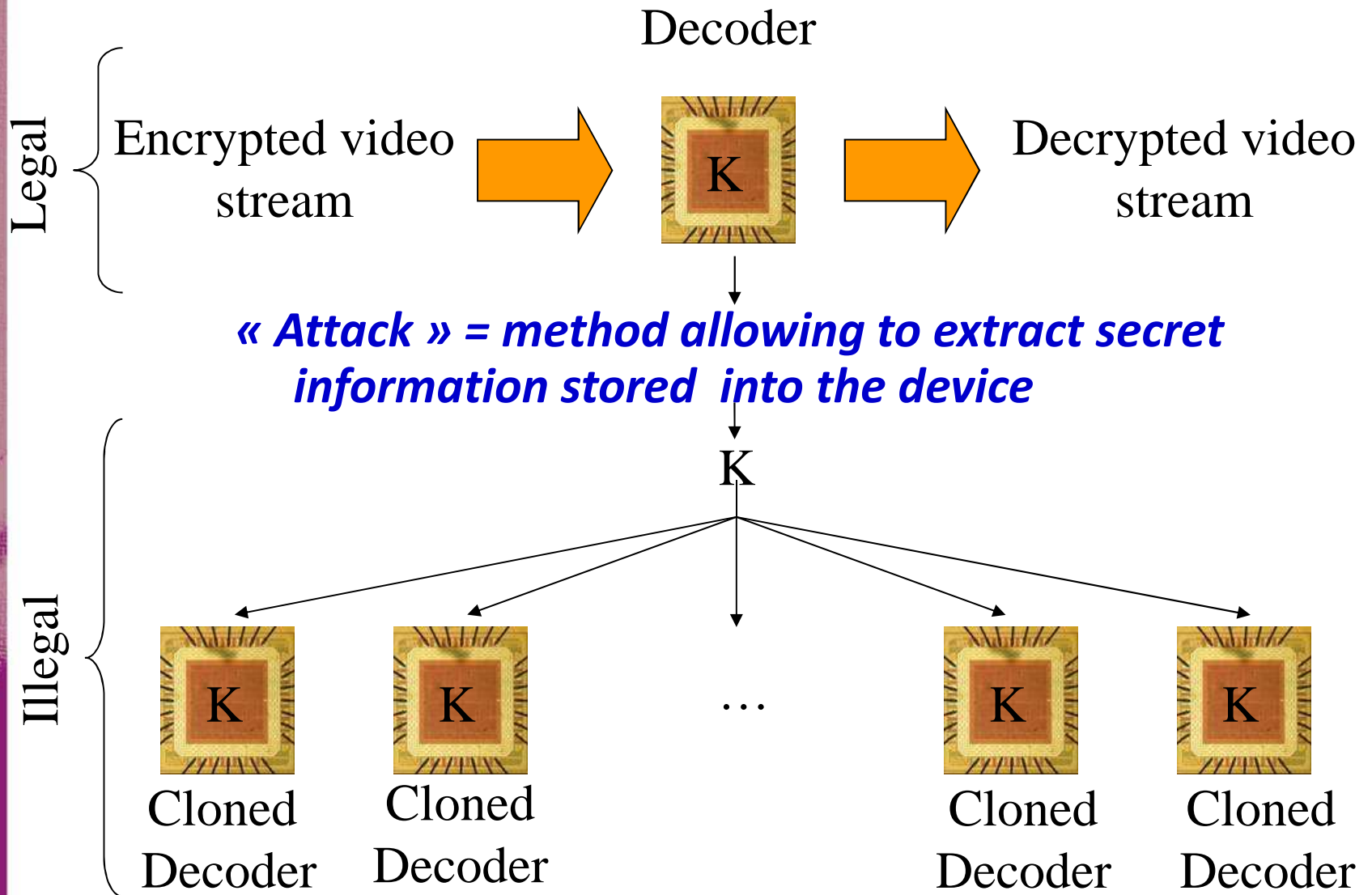
- Problem
- Case analysis
- Prototyping
- Conclusions



?



Problem



Attacks on physical devices

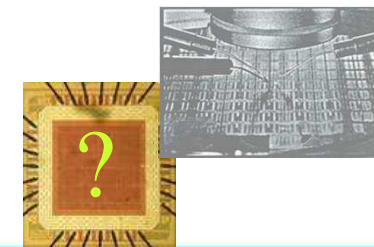
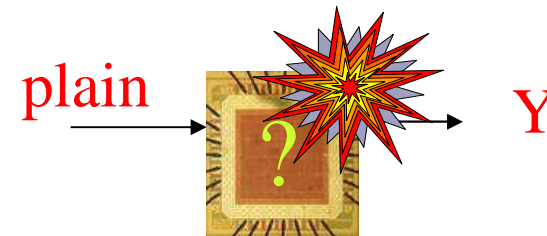
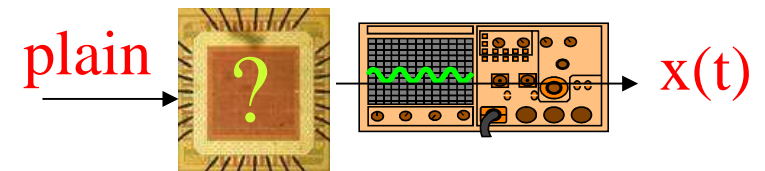
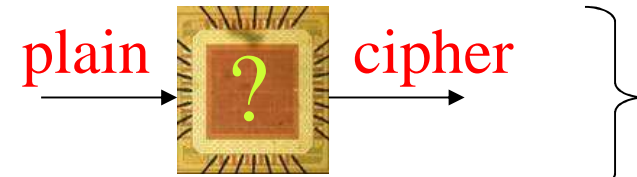
Cryptanalysis : mathematical analysis of plain and cipher texts sets

≠

Side channel attacks (SCA) : analysis of the chip environment when it performs sensitive computations

Fault attacks : modifications of the chip environment to bypass H/S protections

Invasive attacks : probing of internal signals



Countermeasures (CMs)

Sensors CMs



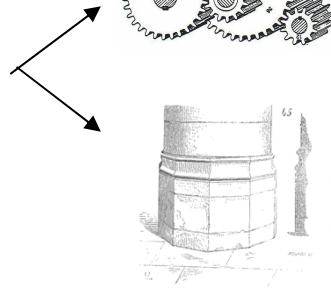
Light intensity, voltage and frequency sensors, spatial, temporal or information redundancy, etc...

Reaction CMs



Mute, reset application or applet, delete data (=kill), etc.

CMs

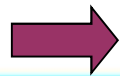


Noise generator, dummy instructions random insertion, memory scrambling, etc.

Internal clock, metallic shields, power filter, balanced logic, balanced place and route, etc.



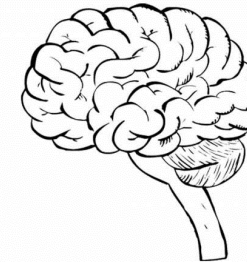
↑security but ↓performances



Numerous, need a global management

Strategy of security : definition

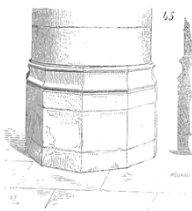
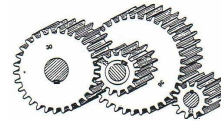
Sensors



Reactions

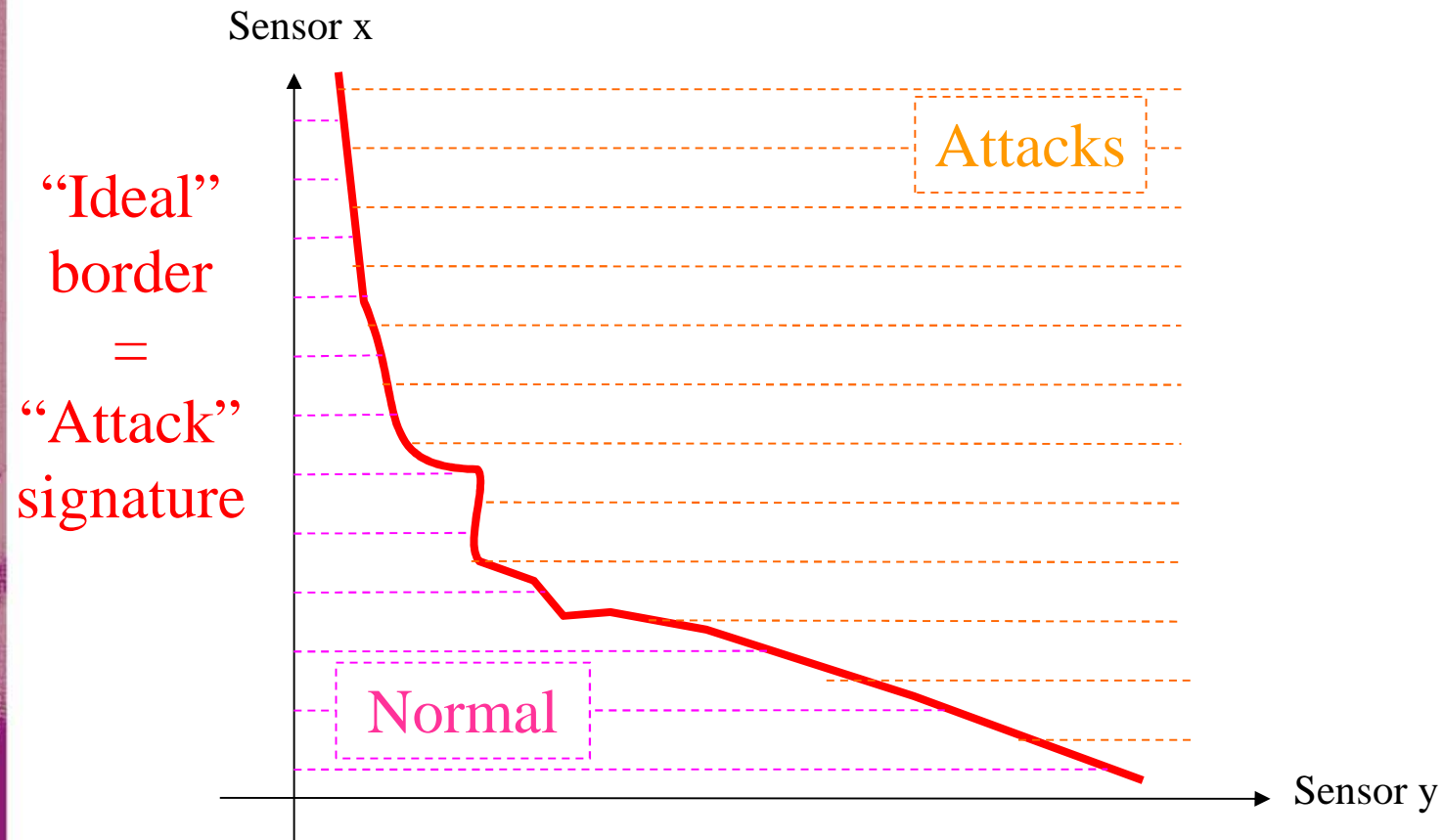


CMs



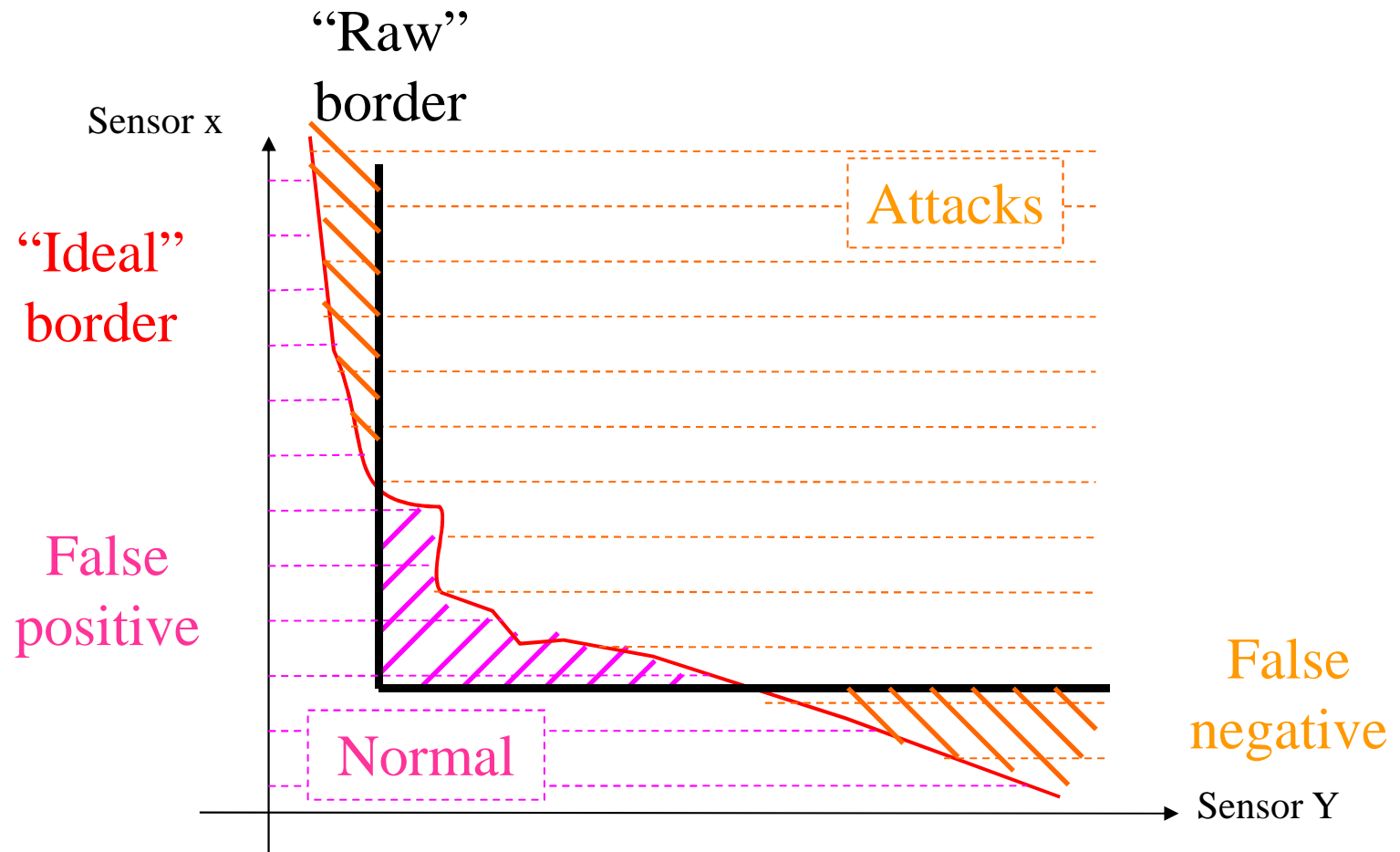
Given the current and previous values of the sensors, the circuit has to **choose** the parameters and the activation of the CMs

Strategies of security: main specification

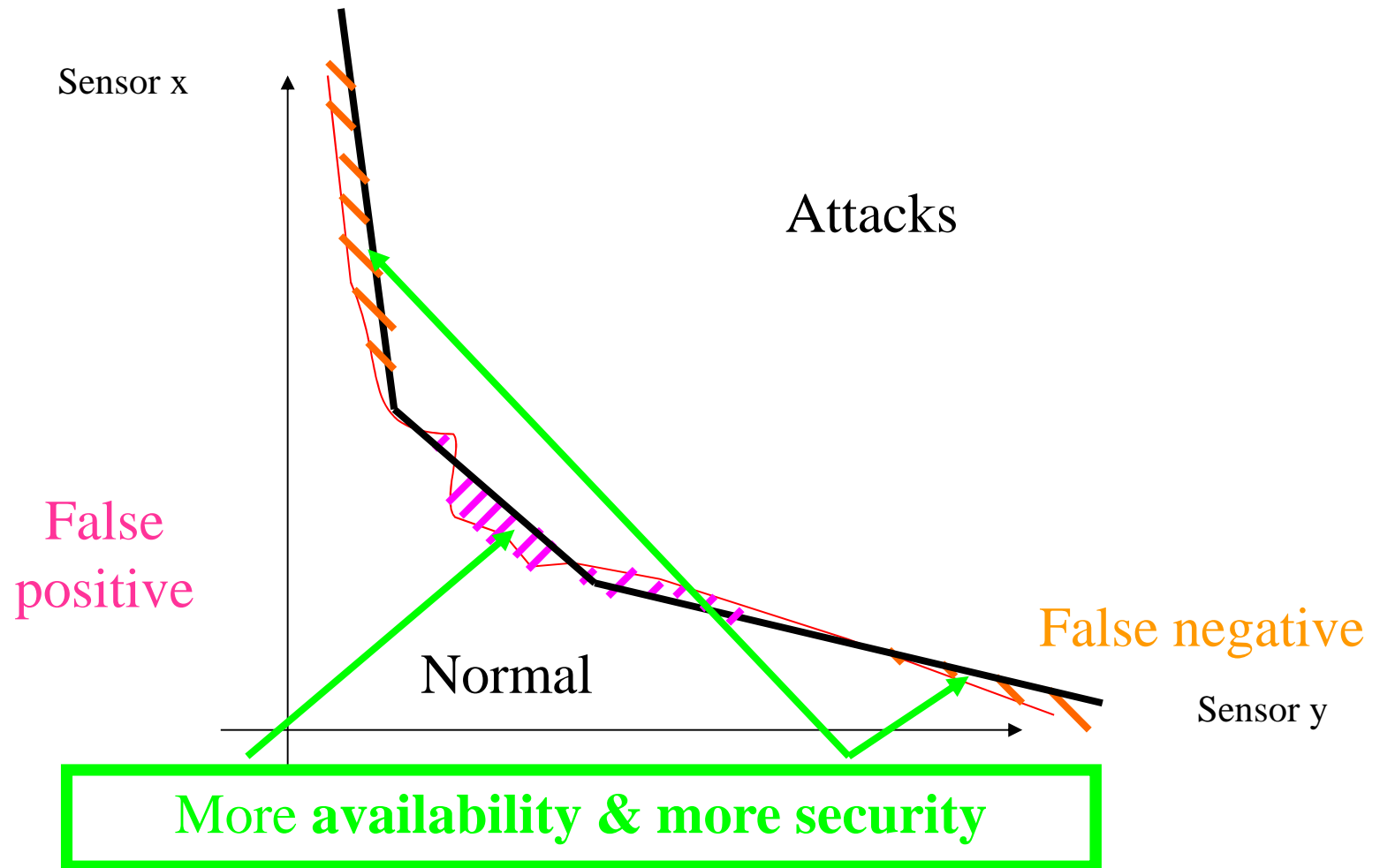


To be able to distinguish attacks from normal behaviours

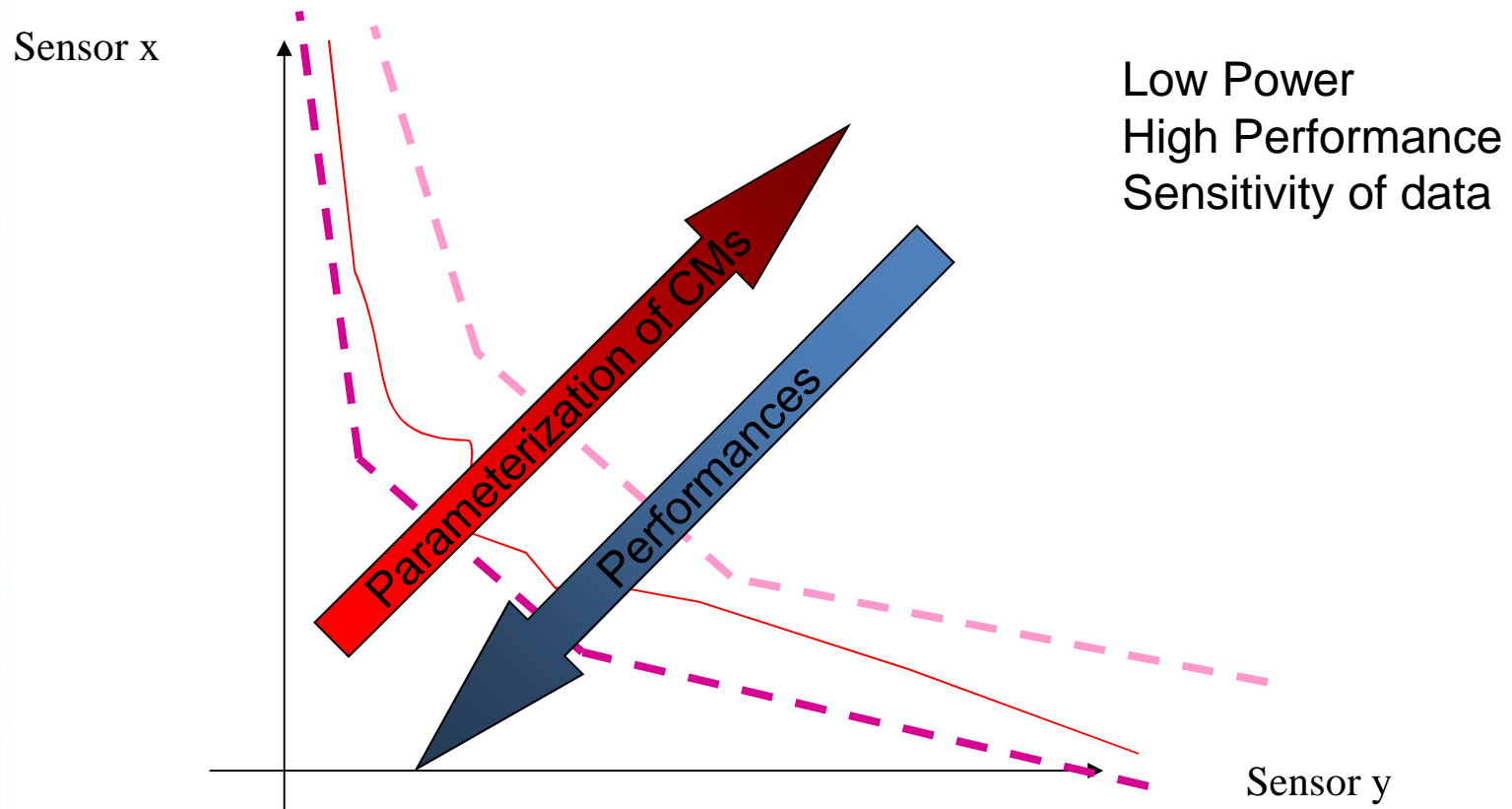
Current strategies of security



Complex strategies of security



Strategies of security: secondary specification



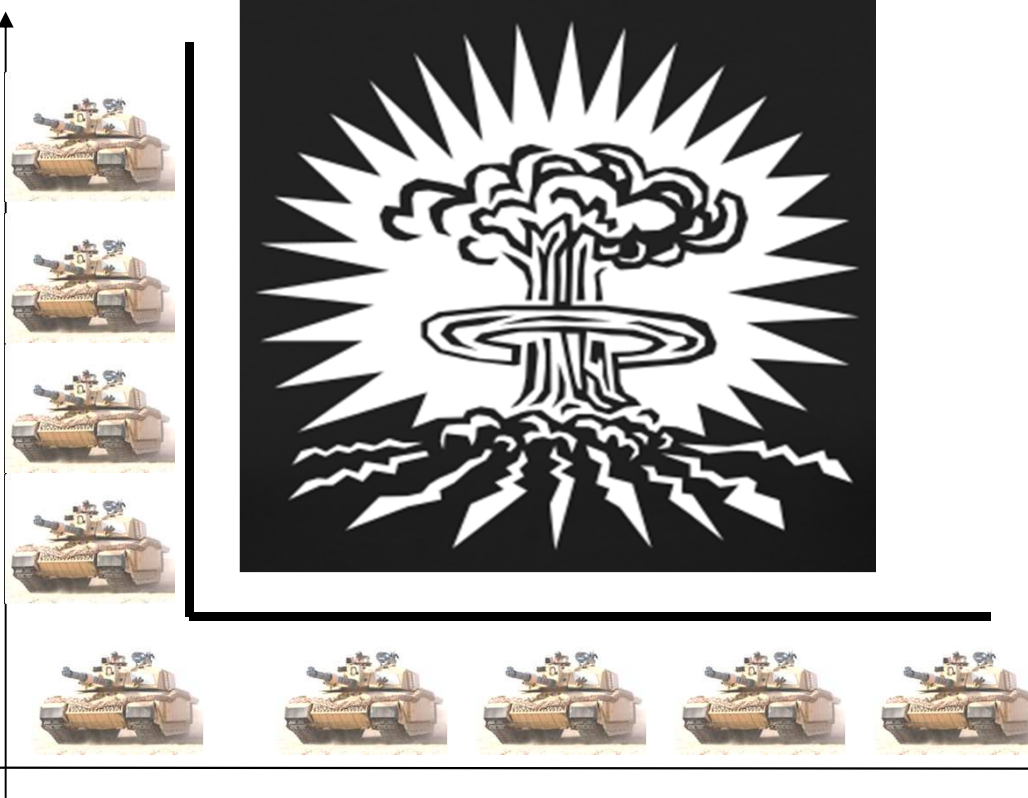
To enable to have dynamically trade-off
between performance and security

Current strategies of security

“basical” configurations

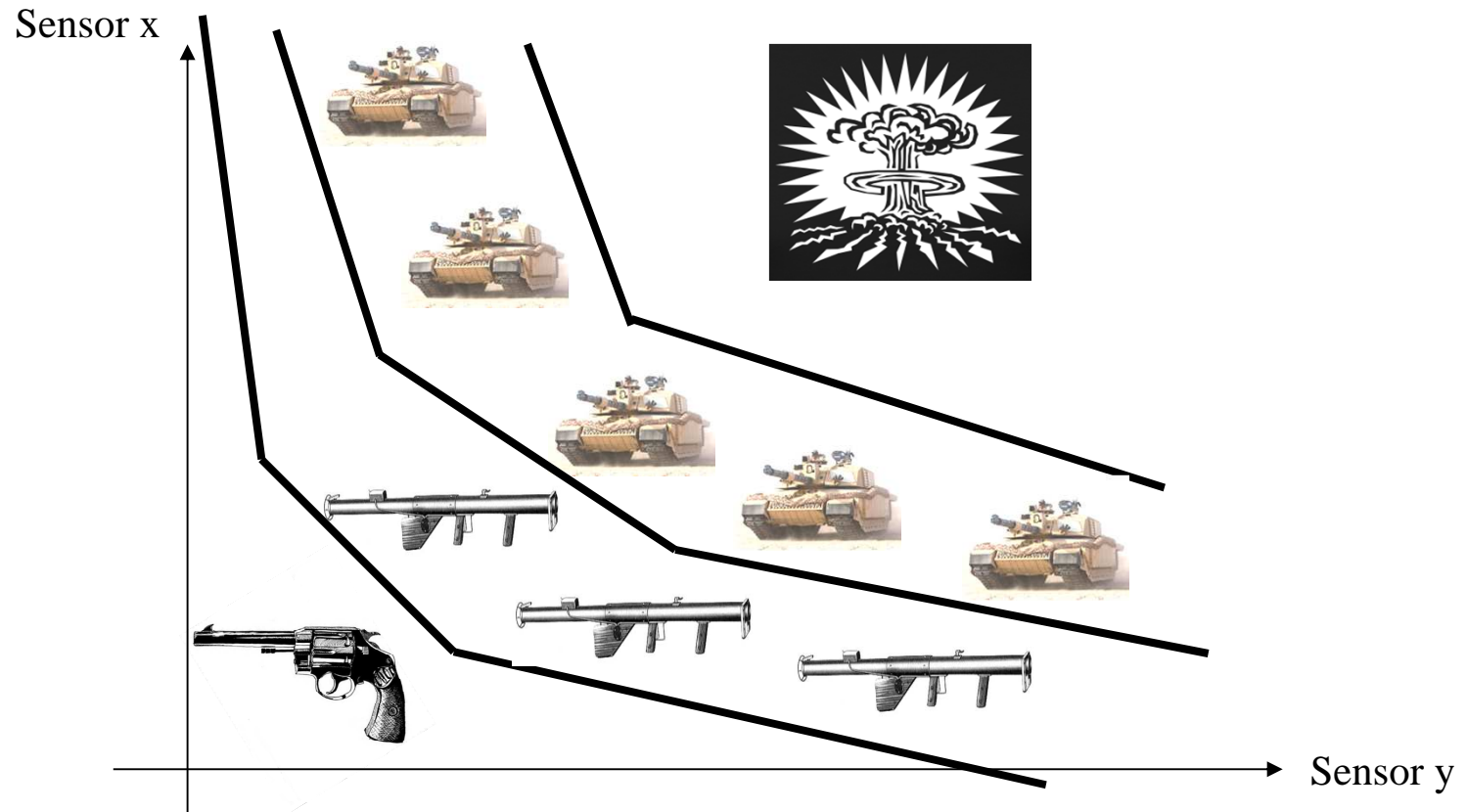
Sensor

X



Sensor Y

Complex strategies of security

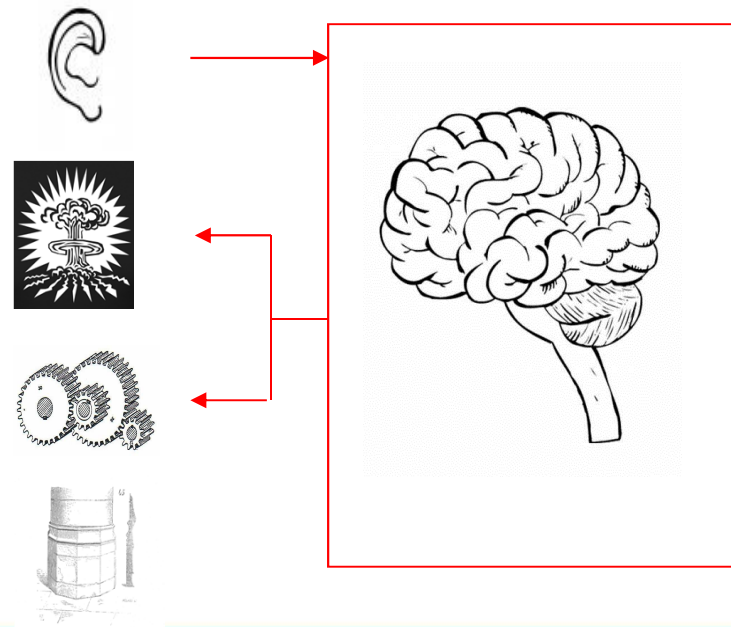


Optimal security for optimal performances

Objectives of “Smart On Smart” (SOS)

- 1) Propose a complex strategy of security for a representative system
- 2) Propose HW/SW mechanisms which enable the implementation of complex strategies of security
- 3) Evaluate the gain security/availability

RQ: SOS takes advantage of state-of-the-art CMs **but** does not aim to develop new ones.



Schedule

- Problem
- Case analysis
- Prototype
- Conclusions



?



Application: Conditional access for pay-TV

Principle

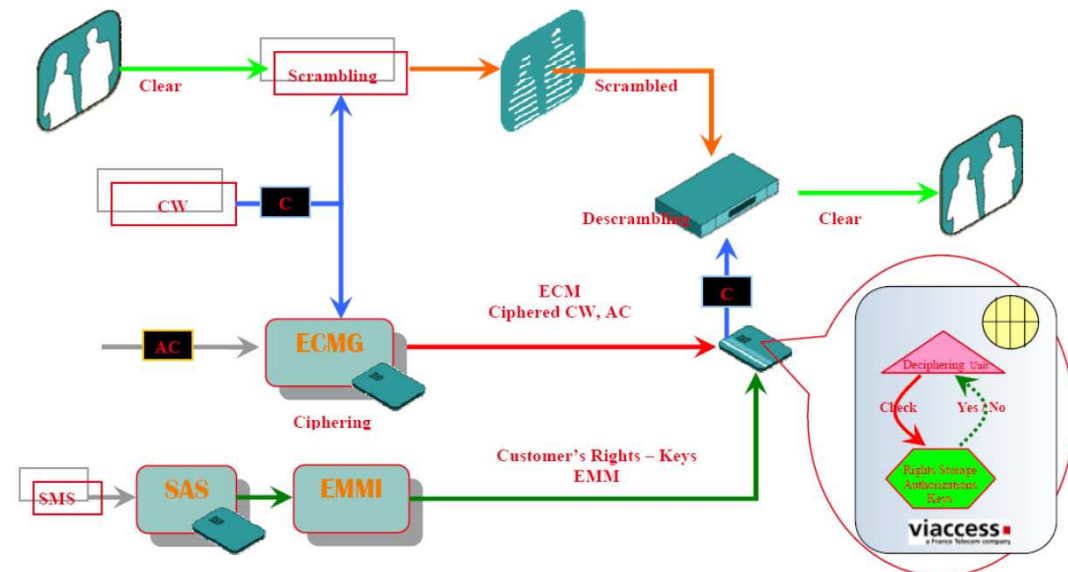
- Based upon the scrambling of an audio/video flow which can be descrambled with a key if and only if the correct right is owned by the smartcard.
- 3 class of commands are used by the system :
 - Subscription writing (Keys, Rights) **Very sensitive**
 - Descrambling (control word) **Sensitive**
 - Subscriber operations (parental control) **Not very sensitive**

Needs

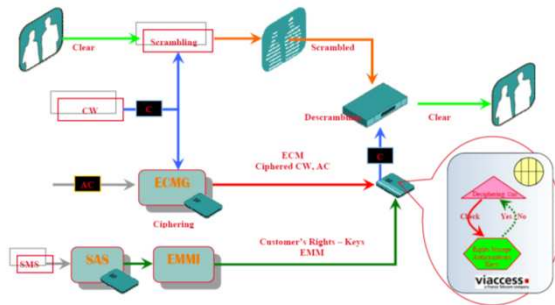
- High level of security
- Real time performance
- High level of reliability

Extra needs

- Low power for integration in mobile phones



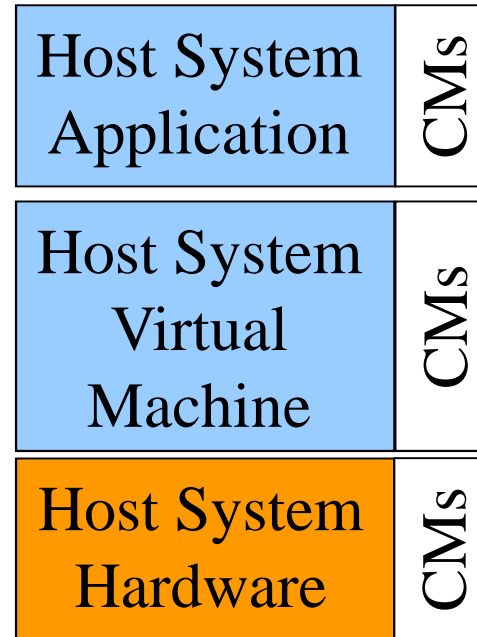
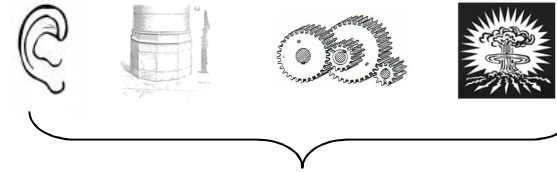
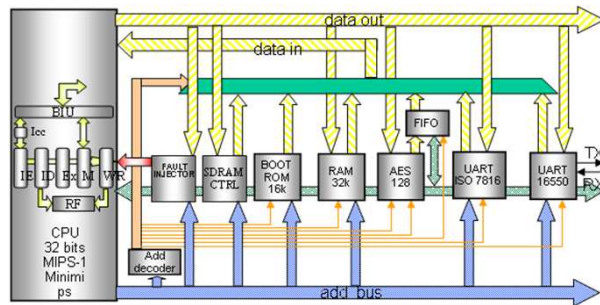
CAS card system : "host"



Viaccess Conditionnal Access

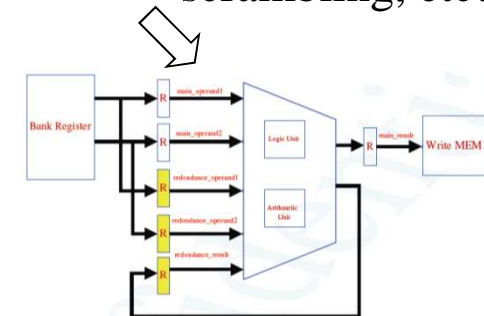
Trusted Logic JavaCard 2.2
GlobalPlatform API

MiniMips



SW CMs:
Masking, control flow, etc..

HW CMs:
Redundancy, scrambling, etc...

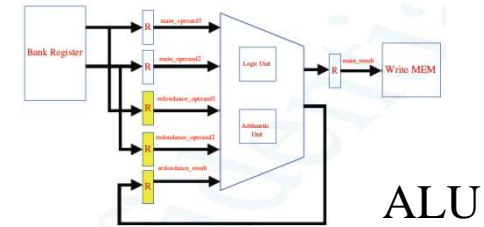


Host system : Example of CMs



Redundancy : Execute N times the same computation and compare the results

Parameter = N



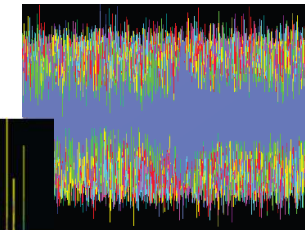
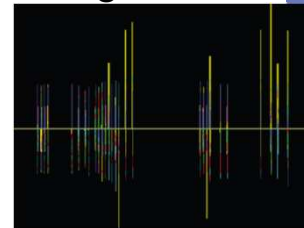
Scrambling : Insert randomly dummy random instructions

Parameter = $\frac{\text{\# useful instructions}}{\text{\# total instructions}}$



Sensors : Emulation of voltage, clock, light and temperature detector

DPA without scrambling



DPA with scrambling



Sensors : Sensitivity of the data which are manipulated by the application

(Keys, Rights) > Control word > Parental control

Proposed strategy of security

Information collection

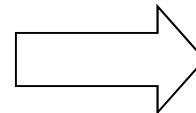
Step1





Normal
or
Attack?

Step2

Counter-measure configuration

Name	Values	Description
LS	{0, ..., 5}	Light value
VS	{0, ..., 10}	Voltage value
EE	{0, ..., 10}	# of corrupted execution flow (triggered by the VM)
CE	{0, ..., 10}	# of corrupted cryptographic execution (triggered by the VM)
PE	{0, ..., 10}	# of wrong PIN (triggered by the Application)
UE	{0, ..., 50}	# of error during the test of integrity of the user data (triggered by the Application)
NE	{0, ..., 1000}	# of methods that have processed without error (triggered by the VM)
DS	{0, ..., 5}	Sensitivity of the data manipulated by the application.



	Redundancy	Scrambling	Reset	Kill
	No	No	No	No
	*2	1/10	No	No
	*3	1/50	Yes	No
	-	-	-	Yes

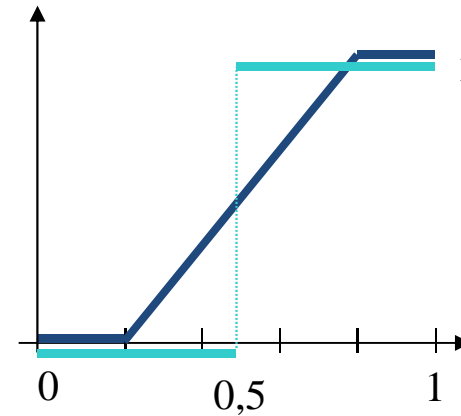
Performance ↑

Security ↓

Step 1 : Fuzzy logic

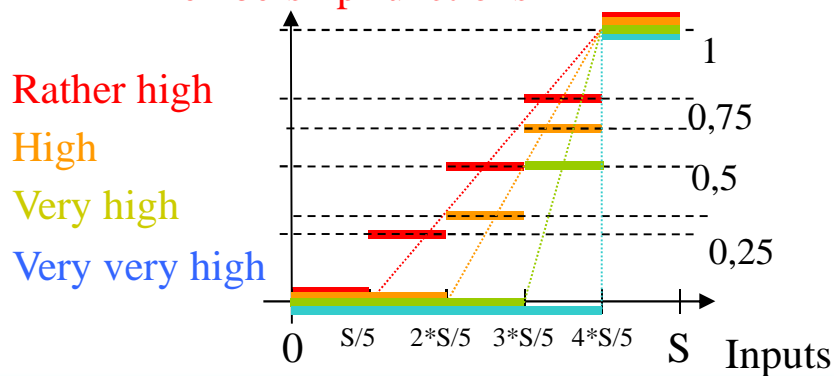
Security expert knowledge
 → rules sometimes vague
 and imprecise
 → Fuzzy set

Membership functions

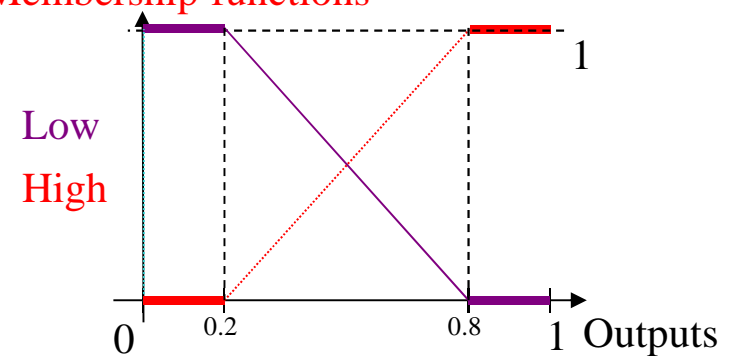


R1 : ``IF the voltage VS is *RATHER HIGH* and the light (LS) is *HIGH*
 THEN the "probability" of attack is *HIGH*``

Membership functions



Membership functions



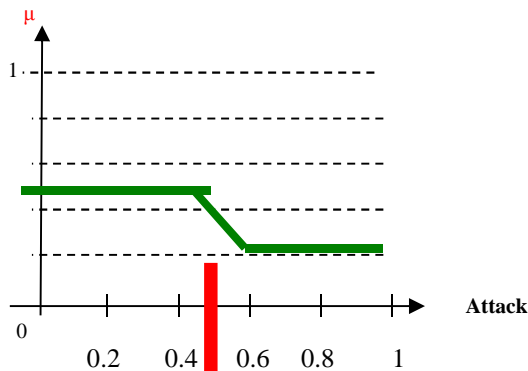
Step 1: fuzzy logic

R0 : ``IF the number of methods that have processed without error (NE) is *VERY HIGH* THEN the probability of **attack** is *LOW*''

R1 : ``IF the voltage VS is *RATHER HIGH* and the light (LS) is *HIGH* THEN the probability of **attack** is *HIGH*''

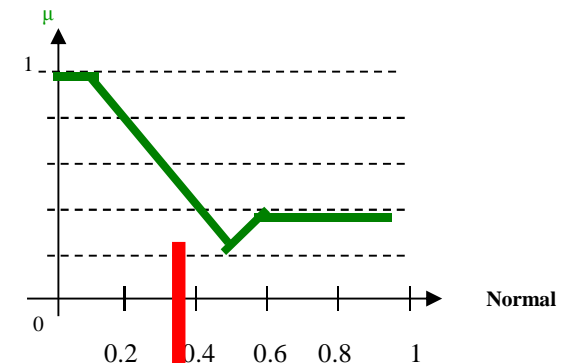
R2 : ``IF the number of cryptographic errors (CE) is *RATHER HIGH* THEN the probability of **attack** is *HIGH*''
ETC.

R0': ``IF the number of PIN code errors (PE), the voltage (VS) and the light (LS) are *VERY LOW* THEN the probability of **normal behavior** is *HIGH*''
ETC.



“Probability” of Attack=0,5

Rule aggregation

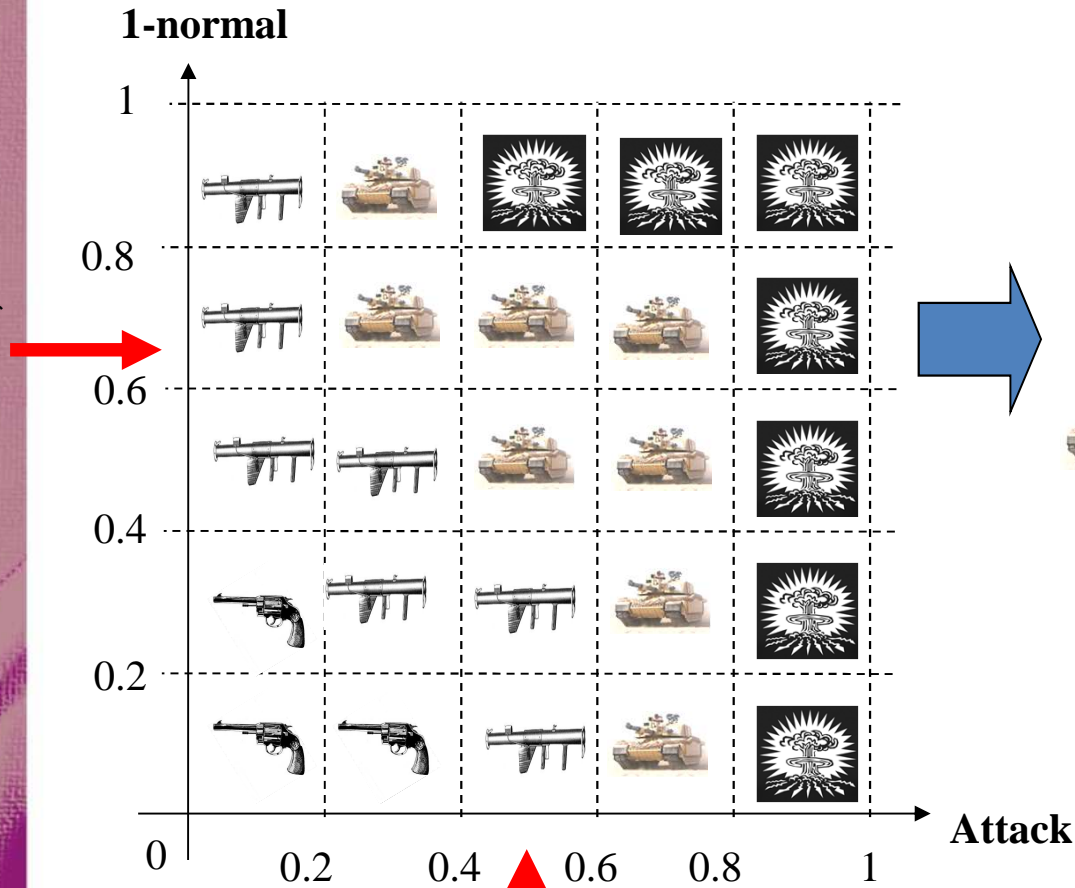


“Probability” of Normal=0,35

Defuzzification


Step 2: Choice of configuration

“Probability” of normal=0,35



“Probability” of Attack=0,5

Choice of Fuzzy sets = very light implementation

	Redundancy	Scrambling	Reset	Kill
	*3	L2	Yes	No

Schedule

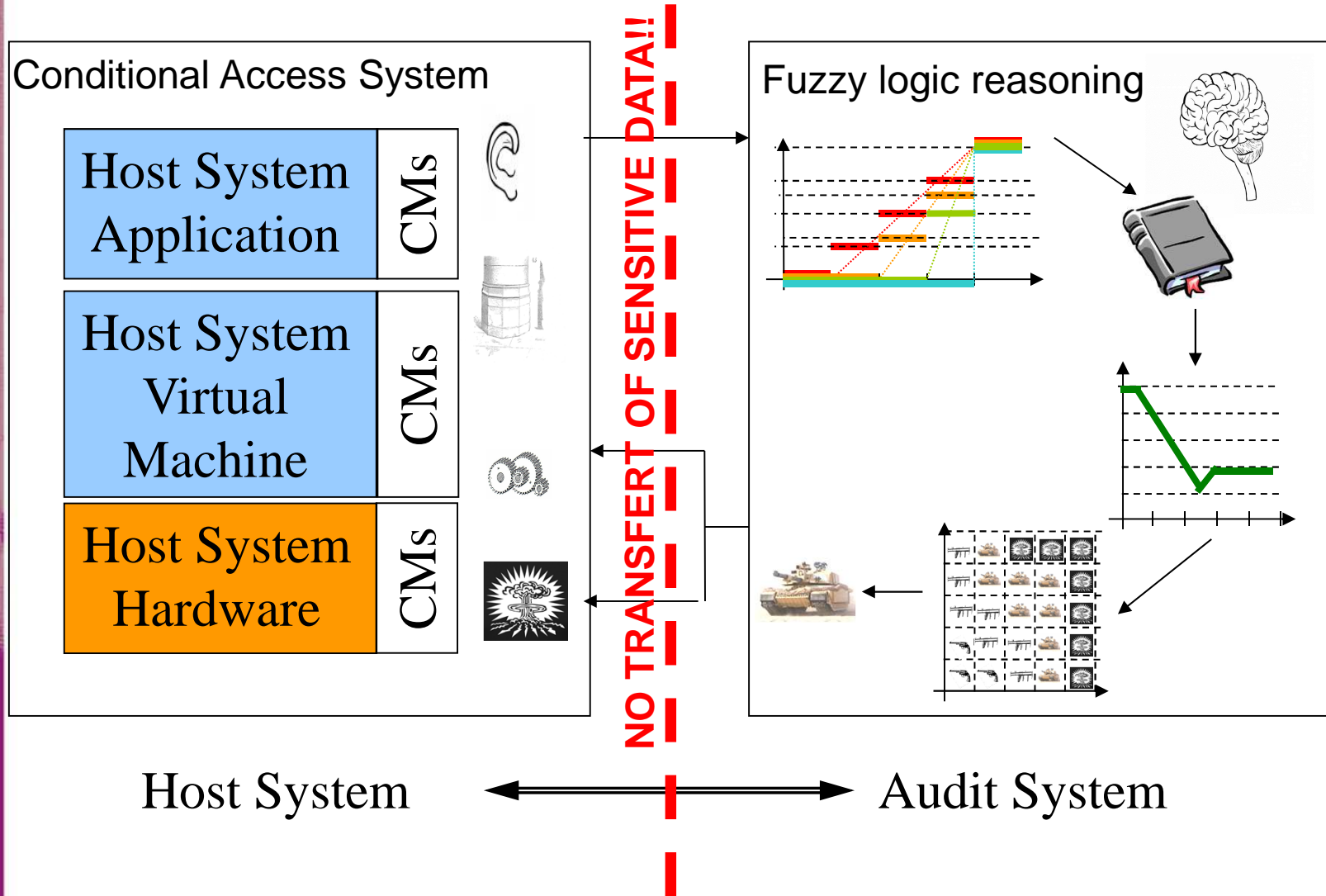
- Problem
- Case analysis
- **Prototype**
- Conclusions



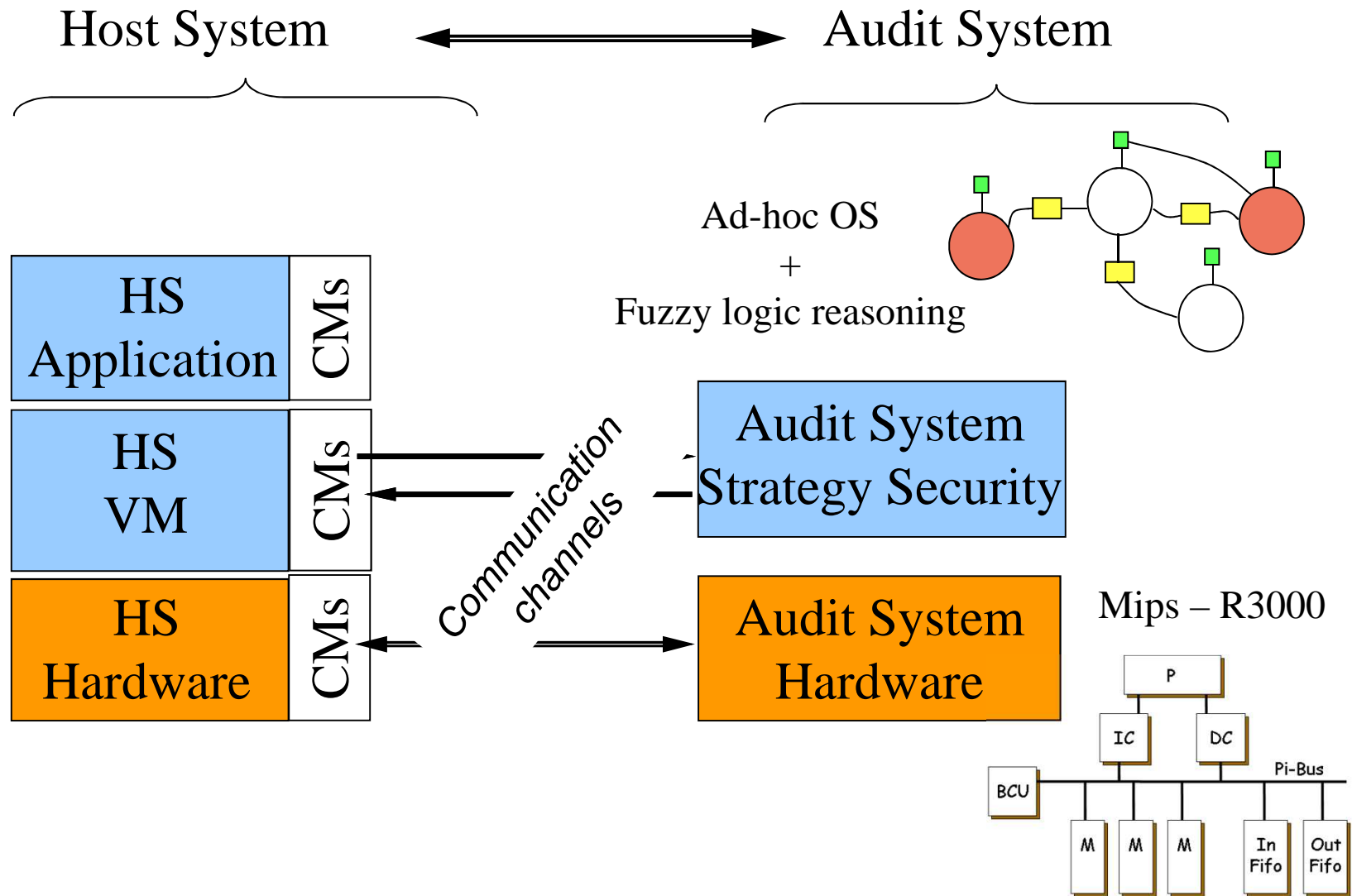
?



Prototype Architecture

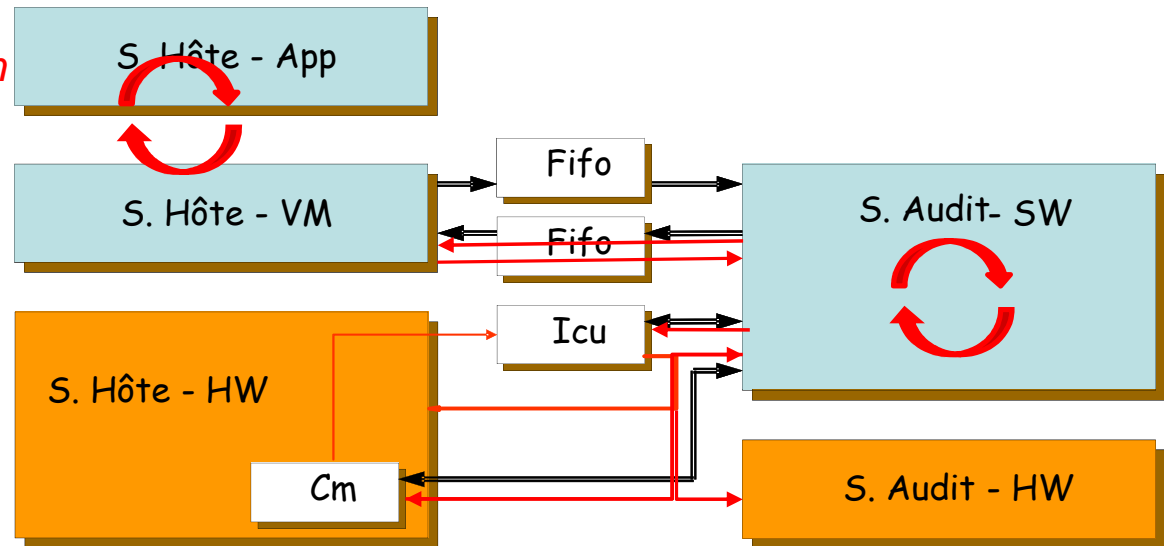


Prototype Architecture



Prototype: Example of communication

1. A sensor event occurs
2. An interruption is raised on HS through the ICU
3. AS computes the configuration of CMs
4. If needed, the AS configures the HW CMs and waits until acknowledgement
5. The AS clears the interruption
6. If needed, the AS configures the SW CMs and waits until acknowledgement
7. The AS resume the execution

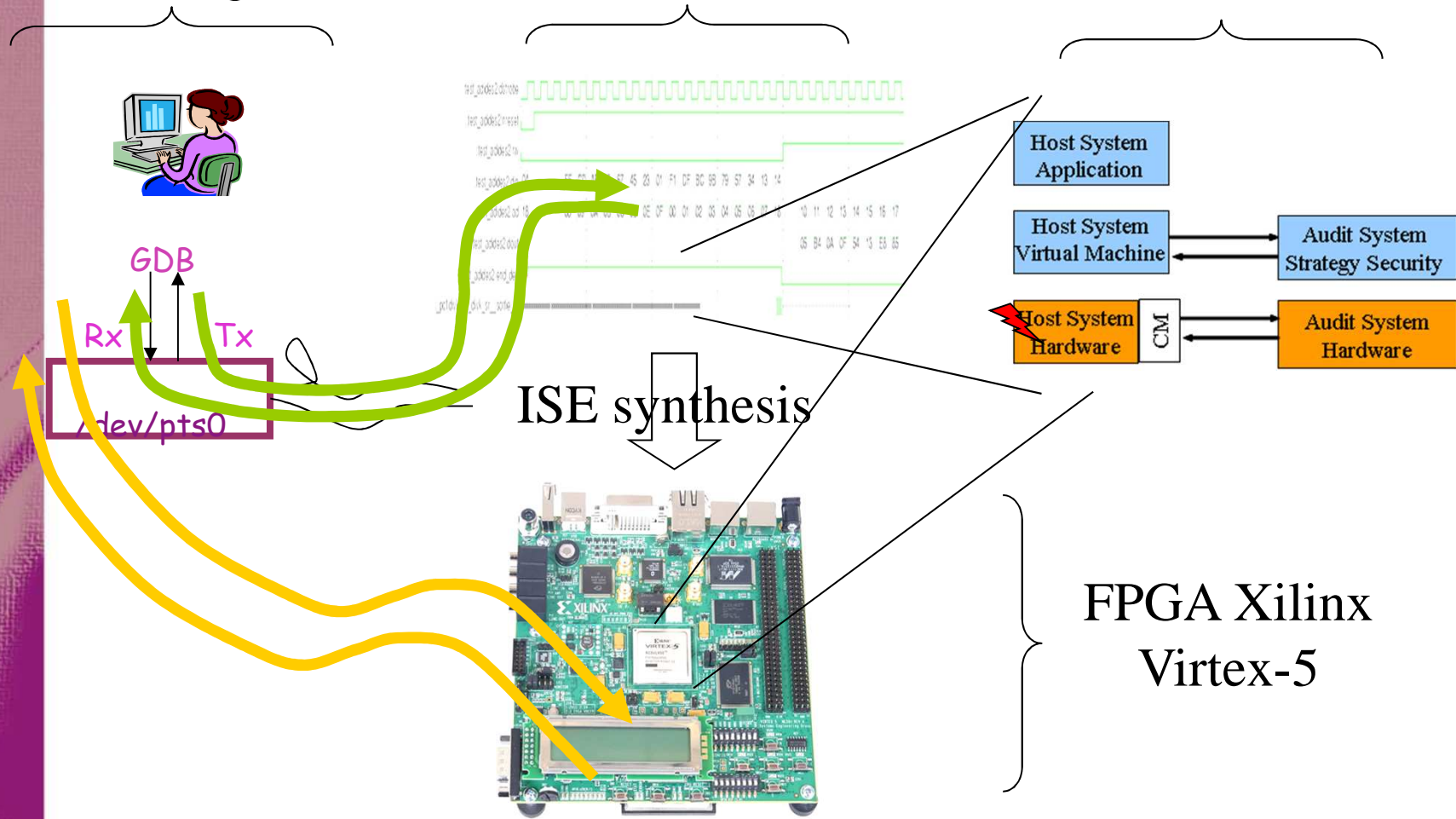


Prototyping

Debug on simulation
on target

Simulation

VHDL SOS models



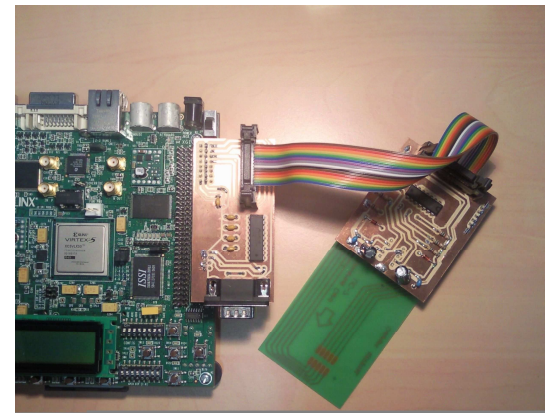
SOS prototype

Based on Xilinx® ML501 virtex5 board

- Host System :
 - 32-bit µprocessor @ 50 MHz
 - MIPS-1 instruction set
 - 5-stage pipeline
 - Harvard architecture
 - 128 KB E2 emulation
 - 896 KB Data/Instruction
 - AES-128
 - ISO 7816-3 UART + connector
 - UART (111520 bauds) + DB9
 - Embedded software stubs for remote debugging
 - Embedded fault injection emulation

Host System only :

Number of Slices	2462 out of 7200	34%
Number of Slice Registers	2421 out of 28800	8%



- Audit system :
 - Mips like cpu @50MHz
 - 4KB Data
 - 32 KB Instruction
 - Simple UART + DB9
 - ICU + comm FIFO

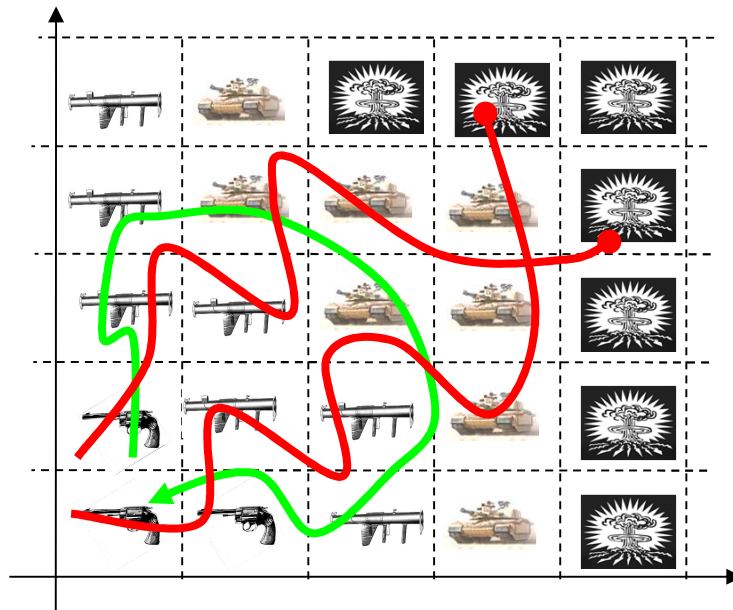
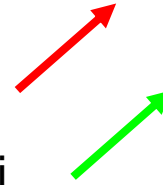
Host System + Audit system :

Number of Slices	3490 out of 7200	48%
Number of Slice Registers	4534 out of 28800	15%

Method of validation

Definition of attacks scenarii

Definition of "normal use" scenarii



Verification of the changes of configurations

- No Kill during "normal use" scenarii
- Kill during attacks scenarii

Estimation on the performances and security for each configurations



Ongoing...

Schedule

- Problem
- Case analysis
- Prototyping
- Conclusions and future work



?

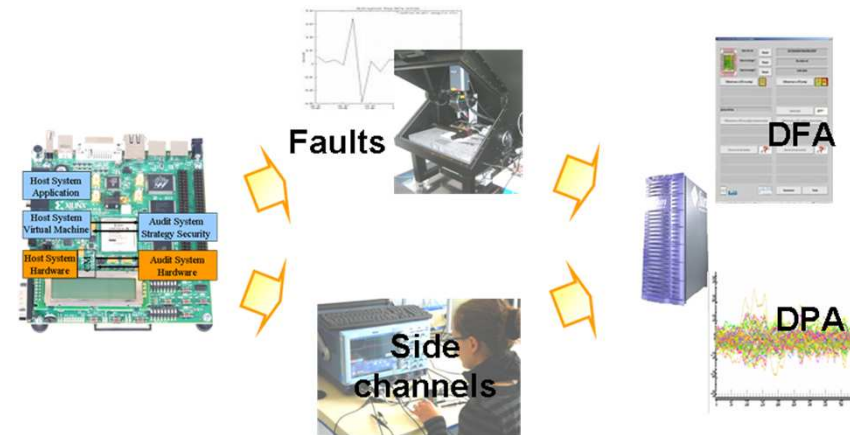


Conclusion

- To the best of our knowledge, the only **first step** towards the implementation of complex strategies of security
- Realization of a prototype which aggregates multi-disciplinary expertise
- Promising results
 - Re-organization of security features through the entire system
 - Innovative strategy of security based on fuzzy logic
 - Proposal of an architecture enabling the execution of complex strategies of security
 - Set up of a dedicated HW/SW design methodology (including debugging tools and built-in security estimation capabilities)

Future work

- Fine tuning of the current rules set
- Security characterization of the prototype with ENSMSE-CMP benches at Gardanne



- Having a trade-off between security/availability raises many questions

MODEL ATTACKER AND USER!

- ⇒ Which formalism ?
- ⇒ Expert knowledge & rules set based system
- ⇒ Data bases of attacker and user behavior & learning algorithms
- ⇒ Are the current sensors suitable?
- ⇒ etc.

Thank you for your attention!

Questions?



FAQ on SOS

Is the audit system is a new Side Channel (SCA) or Fault Attack's (FA) leakage source?

➡ No, because the audit system NEVER has access to sensitive information (like key)

If the SA is subject to FA ?

- If it is blocked by FA, the host system will be blocked

➡ No information leakage

- If it does not compute the right security level

➡ The basic CMs protect the SH; The attacker has to realize SCA or FA on the host

If the communication channels are not protected and so, subject to FA?

➡ The basic CMs protect the SH; The attacker has to realize SCA or FA on the host