

# ML-Based Intrusion Detection in Networks and Hosts

**Maxime Puy**

LIMOS/SIC/RS, Univ. Clermont Auvergne

January 16<sup>th</sup>, 2024



# Who am I?



- **Maxime Puys**
- Ph.D. in Computer Science Security in 2018 from Verimag, Univ. Grenoble Alpes
- 2018 – 2023: Research Engineer at CEA-LETI, Grenoble
- **Since 2023-10:** Assistance Professor at IUT/LIMOS/SIC/RS
- **E-mail:** Maxime.Puys@uca.fr
- **Research interests:**
  - Cybersecurity of (I)IoT devices and networks
  - Cryptographic protocols

# Internet of Things



- Physical objects with sensors/actuators
- Having processing ability, software, etc
- Connected and exchanging data over Internet or other networks.
- Eg. appliance, factory, health, wearables, etc

# Critical IoT

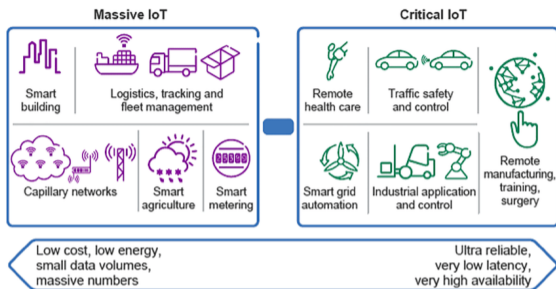


Figure: Two types of IoT: [Alq19]

## Massive IoT:

- Target collected/computed data;
- Secrecy, privacy, integrity.

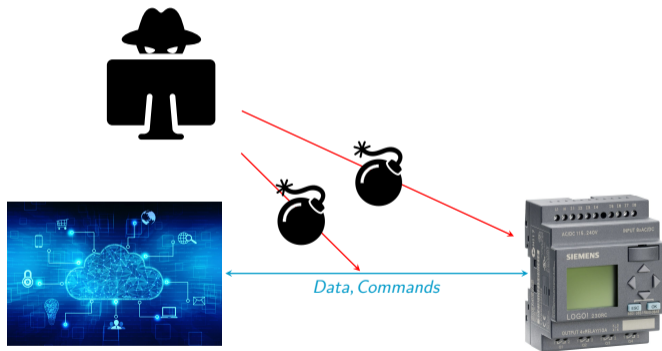
## Critical IoT:

- Target physical process;
- Availability, safety

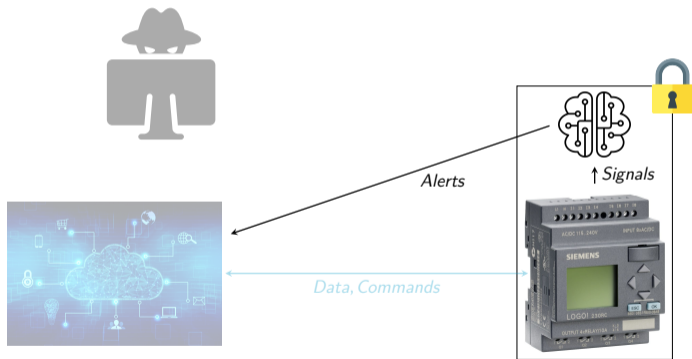
# Overview



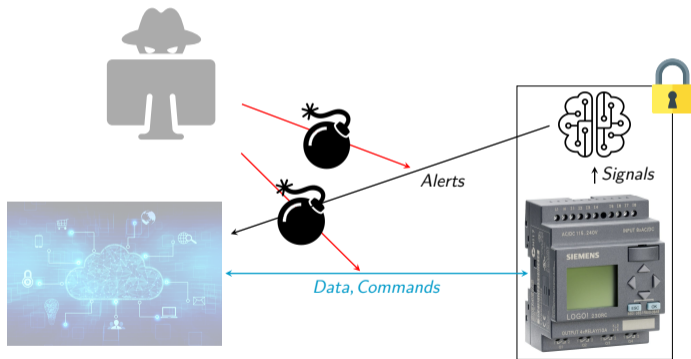
# Overview



# Overview: 1 – Embedded Host-IDS

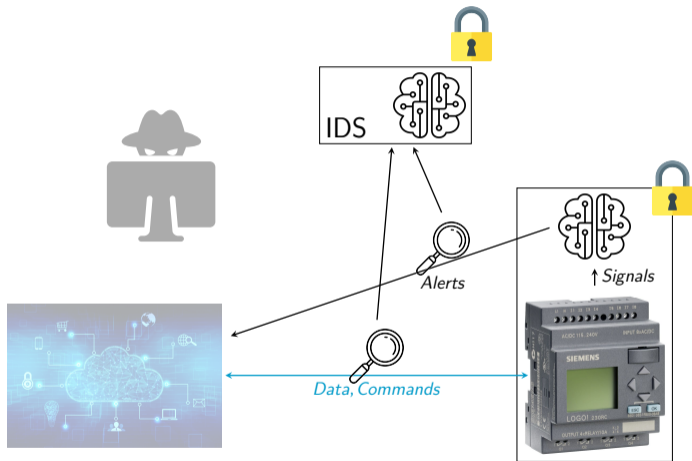


# Overview: 1 – Embedded Host-IDS





## Overview: 2 – Network IDS



# Outline

## Previous Work: Embedded Host-IDS

- Define and embed intrusion detection system for critical devices.

## Research Directions as part of LIMOS

- Optimization of the selection of security functions in networks and ensure their resilience.

## Focus on On-Going Works: Network IDS

- Application of previous works on ML based host-IDS to network IDS

## Embedded Host-IDS

# Motivation

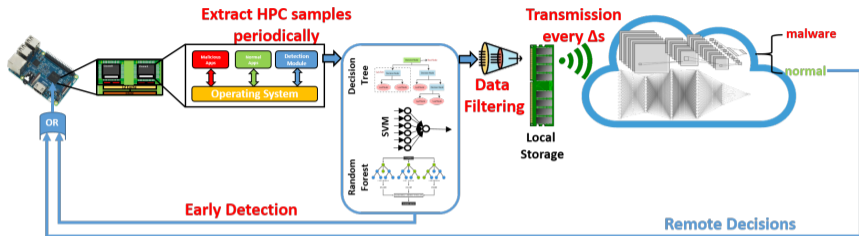
## Hard to patch

- Hardware flaws often **require hardware patch**
- Sometimes really hard to retrieve devices:
  - Health devices, volcano captors, etc
- Software patches on HW flaws highly decrease performances

⇒ Often **easier to monitor** device for attack than patching

- Ph.D thesis N. Polychronou (CEA-LETI), 2019 – 2021
  - Goal: Detect microarchitectural attacks with limited performances

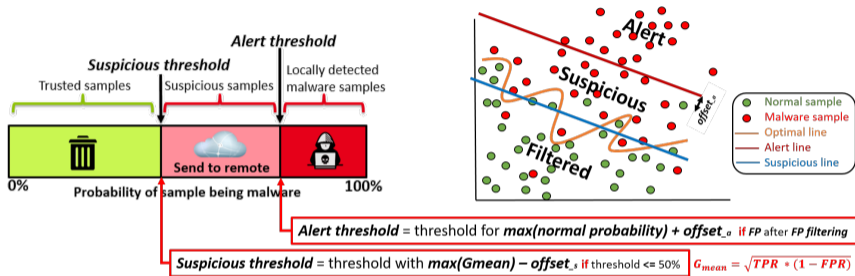
# Local-Remote Approach



- **Locally detect samples** with very high probability of being malicious
  - ▶ Minimize required bandwidth
- Only **sends suspicious samples** to a remote AI
  - ▶ Filter normal data to minimize bandwidth
- Provide a level of security even if network is down

[EuroS&P Workshops'23] N. F. Polychronou, P.-H. Thevenon, M. Puys, and V. Beroulle, 2023.

# Two Level Threshold



[DSD'21] N. F. Polychronou, P.-H. Thevenon, M. Puys, and V. Beroulle, 2021.

# Results with two Thresholds

Single-Level Threshold		TPR	FPR
Simple MLs	LinearSVC	99.75%	1.63%
	Logistic Regression	99.74%	1.46%
	AdaBoost	99.70%	0.68%
Complex MLs	LSTM7	72.99%	0.03%
	LSTM AutoEncoder7	92.41%	0.45%

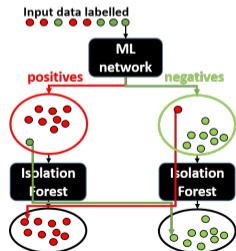
Two-Level Threshold	TPR	FPR
AdaBoost + LSTM7	73.09%	0.02%
<b>AdaBoost + LSTM AutoEncoder7</b>	<b>92.66%</b>	<b>0.13%</b>

## Why Local+Remote?

- **Minimal FPR** compared to single-level threshold with simple ML
- TPR remains high enough because:
  - ▶ We only need **one timing-window with high anomaly score** to detect a malicious app
  - ▶ Malicious application do not execute only malicious actions in their whole execution

# Reducing FPs

- 0.13% FP at 10ms sampling rate means 468 FP per hour!
- If the remote receives data before  $\Delta s$ , then trust its predictions
  - ▶ Most probably an attack overflow the local storage and we immediately transmitted the data before the expected  $\Delta s$
- If the remote receives data at  $\Delta s$ , we must be careful for FPs
  - ▶ Most probably under normal operation
  - ▶ We need a double check



Two-Level Threshold	TPR	FPR
AdaBoost + LSTM AutoEncoder7	92.66%	0.13%
AdaBoost + LSTM AutoEncoder7 + Isolation Forest strategy	92.66%	~0%



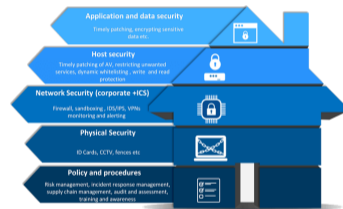
## My Research Directions

# Research Directions

- **Problem Statement:** How to optimize the selection of security functions in networks and ensure their resilience in the event of attacks?

- **Motivation :** Security essential but not across all areas:

- ⚠ Too much defense: overhead, finance, frugality
  - ▶ Need cysec AND domain specific risk analyses [FPS'17]
  - ▶ **Goal:** apply countermeasures where it is important and analyze relations between countermeasures



- Two Key Axes:
  - 1 Optimization of security function selection in networks.
  - 2 Network resilience.

# Axis 1: Optimization of Security Function Selection in Networks

- From risk analyses, derive a **network heat-map**:
  - ▶ Identify points to protect and vulnerability paths.
  - ▶ **Orchestrate countermeasures**.
  - ▶ Existing state-of-the-art but lacks safety aspects.
  
- **Characterize** countermeasures:
  - ▶ Assess their **impact** concerning risk analysis.
  - ▶ Evaluate **costs** (in overhead, finance, consumption, etc).
  
- Employ **machine learning** to:
  - ▶ **Fine-tune** countermeasure parameters (e.g., ML-based IDS).
  - ▶ Utilize GANs / generative AI for discovering needs and proposing countermeasures.

## Axis 2: Network Resilience

- Many works focus on detection and protection;
- Few on response and recovery.
  
- Yet, concepts extensively studied outside of security:
  - ▶ Alternative paths, ring or mesh topologies, check-pointing.
  
- **Idea:**
  - ▶ Validate the gap between existing solutions and their use in security contexts.
  - ▶ E.g., HSR/PRP protocols, network segmentation into 62443 zones and conduits, redeployment of business/cyber functionalities.



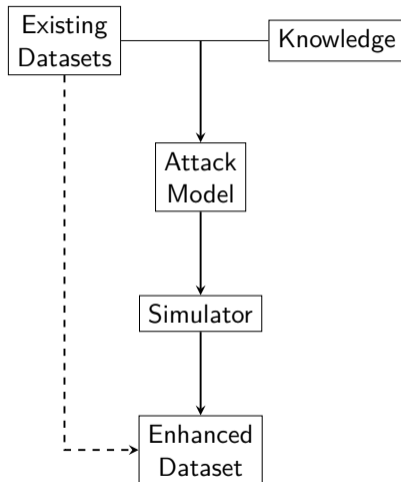
On-Going Work:  
ML-based Network IDS

# Context and Motivation

- On-going work with G. CHALHOUB
- 2 M2 internships + 1 Ph.D to begin
  
- Strong SOTA on ML-based Network IDS
- Current challenges:
  - ▶ Feature selection, link between features and response, cross-dataset features, time-invariant feature importance, etc
  
- Focus on datasets:
  - ▶ Current datasets: **old, content-specific, limited in variety, heterogeneous features.**
  - ▶ Most IDS: Trained and tested on a specific dataset
  - ▶ Data are analyzed without domain specific knowledge

# Idea(s)

- **Evaluate** existing datasets (features, contents, etc)
- **Cut, merge**, different datasets for broader use cases
- Train IDS on one dataset and test it on another (more realistic)
- Make use of simulation to **generate new data**:
  - ▶ Requires fine knowledge of attacks
  - ▶ Possible to compare features from simulated data with real datasets to check for mistakes.






# Conclusion

Thanks for your attention!









# References I

-  Salman Alqahtani, *Performance evaluation of a priority-based resource allocation scheme for multiclass services in iot*, International Journal of Communication Systems **32** (2019), no. 18, e4151.
-  Daniel Gruss, Clémentine Maurice, and Stefan Mangard, *Rowhammer.js: A remote software-induced fault attack in javascript*, Detection of Intrusions and Malware, and Vulnerability Assessment - 13th International Conference, DIMVA 2016, San Sebastián, Spain, July 7-8, 2016, Proceedings (Juan Caballero, Urko Zurutuza, and Ricardo J. Rodríguez, eds.), Lecture Notes in Computer Science, vol. 9721, Springer, Springer, 2016, pp. 300–321.
-  Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, et al., *Spectre attacks: Exploiting speculative execution*, Communications of the ACM **63** (2020), no. 7, 93–101.

## References II

-  Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg, *Meltdown: Reading kernel memory from user space*, 27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018 (William Enck and Adrienne Porter Felt, eds.), USENIX Association, 2018, pp. 973–990.
-  Zhenyu Ning and Fengwei Zhang, *Understanding the security of ARM debugging features*, 2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019, IEEE, 2019, pp. 602–619.
-  Sofiane Takarabt, *Cache-timing attacks still threaten iot devices*, 2019, pp. 13–30.
-  Ning Zhang, Kun Sun, Deborah Shands, Wenjing Lou, and Y. Thomas Hou, *Truspy: Cache side-channel information leakage from the secure world on ARM devices*, IACR Cryptol. ePrint Arch. **2016** (2016), 980.

# Related Works

Detection mechanism	Attacks	Accuracy	F-score	FPR or Precision	Overhead	System	Local or Remote
Mushtaq et al. [19] <a href="#">Logistic Regression</a> (No Load)	Flush+Reload	99.51%		0.48% FPR	0.94%		Local
Mushtaq et al. [19] <a href="#">SVM</a> (No Load)	Flush+Reload	98.82%		0.397% FPR	1.29%		Local
Mushtaq et al. [19] <a href="#">Logistic Regression</a> (No Load)	Flush+Flush	91.73%		0% FPR	1.10%		Local
Mushtaq et al. [19] <a href="#">SVM</a> (No Load)	Flush+Flush	97.42%		0% FPR	0.79%		Local
WHISPER [20] <a href="#">Ensemble Learning (DT, RF and SVM)</a> One model per malware (No Load)	CacheSCA (F+F, F+R, P+P), Spectre, Meltdown	>97.05%			<8%		Local
FortuneTeller [21] <a href="#">LSTM</a>	CacheSCA (F+F, F+R, P+P), Spectre, Meltdown, Rowhammer, Foreshadow		99.70%	0.125% FPR	3.50%		Local
Wei et al. [23] <a href="#">OC-SVM</a>	Prime + Probe, Spectre, Rowhammer, <b>Evasive</b>	<98.63%		<0.5% FPR	N/A		?
Wei et al. [23] <a href="#">LSTM</a>	Prime + Probe, Spectre, Rowhammer, <b>Evasive</b>	<99.06%		<0.5% FPR	N/A		?
Kuruvila et al. [24] <a href="#">Random Forest</a>	Flush + Flush, Spectre, PNScan, Meltdown, Rowhammer, BashLite,	89.90%	89.91%	89.25% Precision	<1.22%		Local
Wang et al. [25] <a href="#">MPL</a>	CacheCSA (F+F, F+R, P+P), Spectre?	<98,9%	<97%	5.3% FPR	<3.2%		?
Wang et al. [25] <a href="#">Logistic Regression</a>	CacheCSA (F+F, F+R, P+P), Spectre?	<98.9%	91.90%	14.9% FPR	<3.23%		?
<b>Ours</b> (Full Load) <a href="#">AdaBoost + LSTM AutoEncoder7</a>	CacheCSA(F+F, F+R, E+R, P+P), Meltdown, Spectre, Rowhammer, <b>Evasive</b>	<b>98.75%</b>	<b>96.19%</b>	<b>=0% FPR</b>	<b>0.80%</b>		Local + Remote

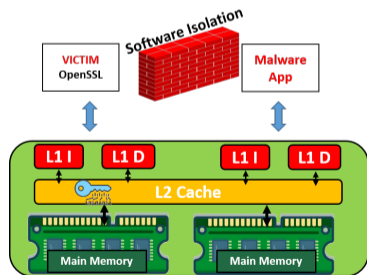
# Microarchitectural Attacks

Hardware **flaws** present on electronic components:

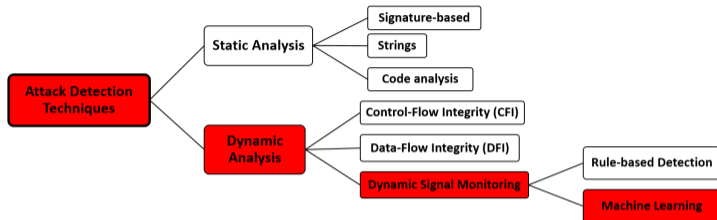
- Traditionally requiring physical access to the device
  - Laser fault injection, Electromagnetic Emissions Side-Channel attacks
- Modern architectures become more complex

Now possible to **exploit hardware** vuln. **using software code** and without physical access:

- Rowhammer [GMM16], Spectre [KHF<sup>+</sup>20], Meltdown [LSG<sup>+</sup>18], Cache Side Channel Attacks (CacheSCA) [Tak19], ClkScrew [NZ19], debug interface [ZSS<sup>+</sup>16]



# Detection Techniques



## Use of Machine Learning (ML)

System information as inputs to the ML:

- Binary Code
- Memory dump
- System calls (SYSCALLs)
- **Hardware Performance Counters (HPCs)**

# Results with two Thresholds

Local ML (filtering only)	Under <i>normal</i> execution	
	data send per minute	filtering percentage
AdaBoost[LogReg]	39kb	99.32%

## Observations

- Normal data do not exceed a certain value
- Explosion of data under attack

Our solution for memory and detection time optimization:

- Set local storage size to max normal
- If local storage overflows before  $\Delta s$ , then immediately transmit to remote system