Building Defensive Self-Knowledge Using Embedded Machine Learning in Avionics

Auteurs:
Julien Depaillat
Philippe Baumard
Pierre Paradinas

22nd November 2023

Plan

1 - Introduction

- Context
- Main objectives
- Similar works

2 – Setting up an embedded HIDS

- HIDS
- Collect module
- Learning module

3 – Experimentation on an avionic module

- Presentation of the experiment
- Current progress
- Experiment results

4 – Conclusion

Introduction

22nd November 2023

1 – Context

- **Thesis in CIFRE convention**
  - CNAM, laboratory : CEDRIC
  - Company : AKHEROS

- **Thesis subject: measuring the evaluation capacity and implementation of a semi-supervised hybrid SDIH in backdoor detection on embedded systems.**

2 – Main objectives

- **Providing additional protection to embedded systems and IoTs**
  - Constantly increasing number
  - Critical Features

- **Learning and detection on the platform to be protected without the need for connection to an external source**
  - Reduction of attack vectors
  - Better responsiveness

2 – Objectives and constraints

- **Detect known/unknown threats**
  - Public APT
  - New attacks / 0 day
- **Limited resources**
  - CPU / Mémoire / Persistant storage is often very limited
- **No modification of existing application**
  - Legacy
  - Certifiability

3 – Similar works

- **Existing studies :**
  - Target platform: IoT (connected objects)
  - Memory corruption attack [1]
  - DDoS attacks, CPU stress [2]
  - Use of machine learning algorithms
  - Use HPCs (Hardware Performance Counters) as data flow for learning and detection algorithms

3 – Similar works

- **Avionics Domain[3]:**
  - Platform : IMA (Integrated Modular Avionics)
  - Syscall ID + Timestamp
  - Extracting data from the platform
  - Posteriori learning
  - Integration of models created on the platform
  - Injection attack: modification of the program execution flow.

3 – Similar works

- Comparative analysis[4]:
  - IoT environment
  - HPCs as input data

Detection efficiency comparison

Algorithm	Rootkit	Backdoor	Trojan	Average
BayesNet	88.1%	91.6%	99.0%	92.9%
MLP	94.0%	92.4%	89.8%	92.1%
OneR	81.5%	92.0%	99.0%	90.9%
JRip	84.8%	92.0%	66.3%	81.0%
J48	85.4%	92.0%	65.7%	81.0%
REPTree	82.8%	92.0%	66.3%	80.4%
SMO	91.4%	89.5%	98.8%	93.2%

Cost comparison

Algorithm	Latency	Memory (block)
BayesNet	60ns	7645
MLP	1020ns	25667
OneR	10ns	292
JRip	20ns	156
J48	30ns	584
REPTree	30ns	377
SMO	220ns	2246

Setting up an embedded HIDS

1 – HIDS

- **HIDS (Host Intrusion Detection System) :**
  - Monitor the system it is embedded on for specific threats
  - Scans files, event logs, running processes, etc.
  - Signatures or behavioral profiles.
  - Examples: Tripwire, OSSEC, and McAfee Host Intrusion Prevention.

1 – HIDS

- **Essential criteria for embeddedness in a critical environment :**
  - Real time
  - Memory fingerprint
  - Detection capacity
  - Offline
  - Legacy / certifiability / lifespan of embedded systems.
  - Protection of know-how

2 – Collect module

- **Goal: Collect the data necessary for learning and detection**
- **Performance: Must be as least intrusive and fast as possible**
- **Data :**
  - Hardware Performance Counters (HPCs)
  - OS errors
  - System / API calls
  - Communications / IO
  - Memory

3 – Learning module

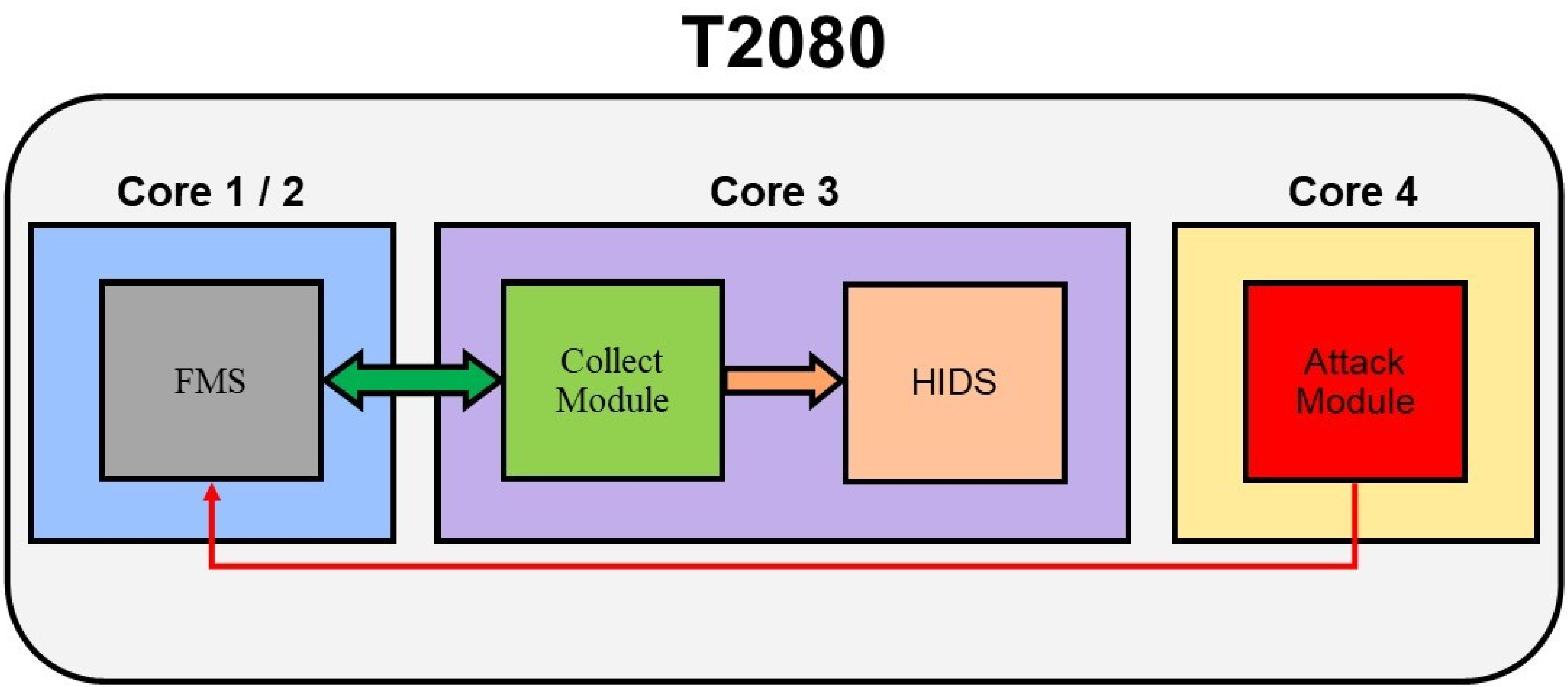
- **Steps:**
  - Choice of learning type: supervised, unsupervised, semi-supervised
  - Choice of learning algorithm
  - Creation of learning scenarios
  - Creation of attack scenarios
  - Impact analysis on application execution
  - Efficiency analysis of selected data for attack detection

Experimentation on an avionic module

1 – Presentation of the experiment

- **Purpose: Is machine learning-based HIDS a viable approach to detect threats on a critical embedded system ?**
- **Validation criterias:**
  - Monitored application must not be disturbed by the collection / HIDS modules
  - Monitored application must not be modified
  - Maximum detection rate for a false positive rate of 0%

- Material :


T2080
Core 1 / 2
Core 3
Core 4
FMS
Collect Module
HIDS
Attack Module

1 – Presentation of the experiment

- **Planned attacks :**
  - Pre-loaded attacks: changes to certain functionalities (trajectory calculation, GPS position, etc.).
  - Injection attacks: random code, control-flow hijacking
  - Passive attacks: variant of Spectre [5] (application memory leak, particularly the “cache timing”).
  - Active attacks: Rowhammer [6] and its variants like Blacksmith [7].

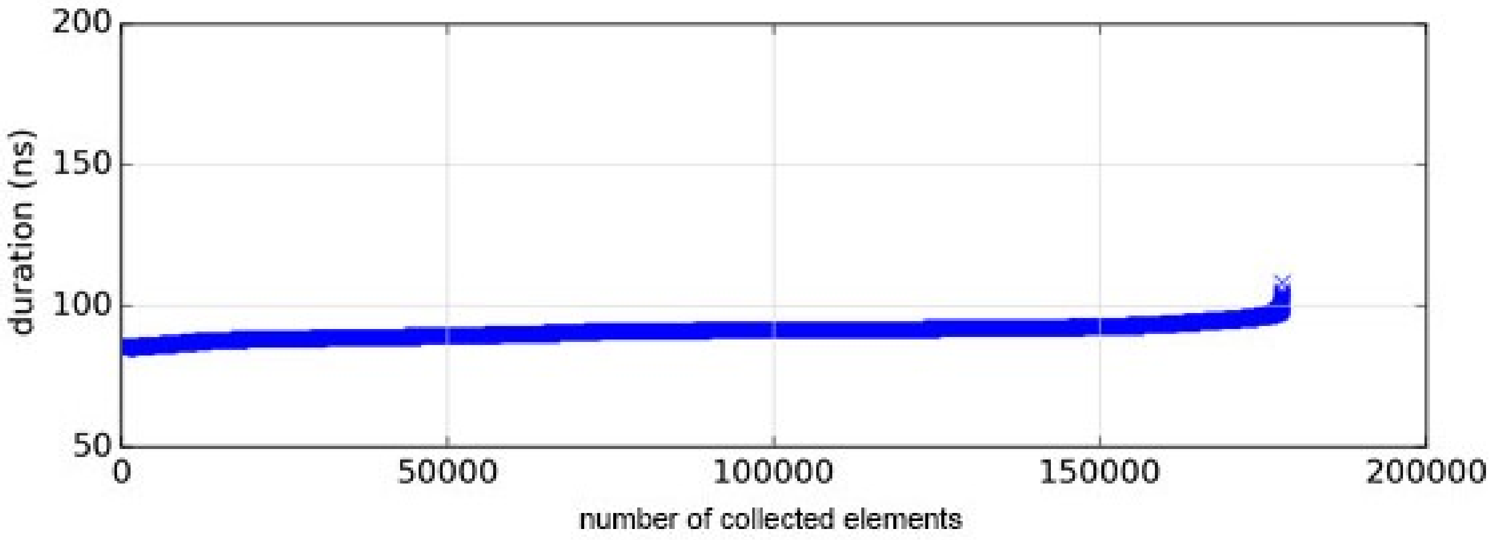
2 – Current progress

- Tasks:

Description	Progress
Validate the approach to determine the data to observe	100%
Modify the OS in order to collect the desired data	100%
Integrate the HIDS into the platform OS	100%
Optimize HIDS for this platform	100%
Create and play normal behavior learning scenarios	10%
Create and play attack scenarios to evaluate detection performance	10%

2 – Current progress

- Increase of the execution time : 82ns and 113ns for 170,000 items collected.
- Cost is considered tolerable for a critical context by our avionic partner.



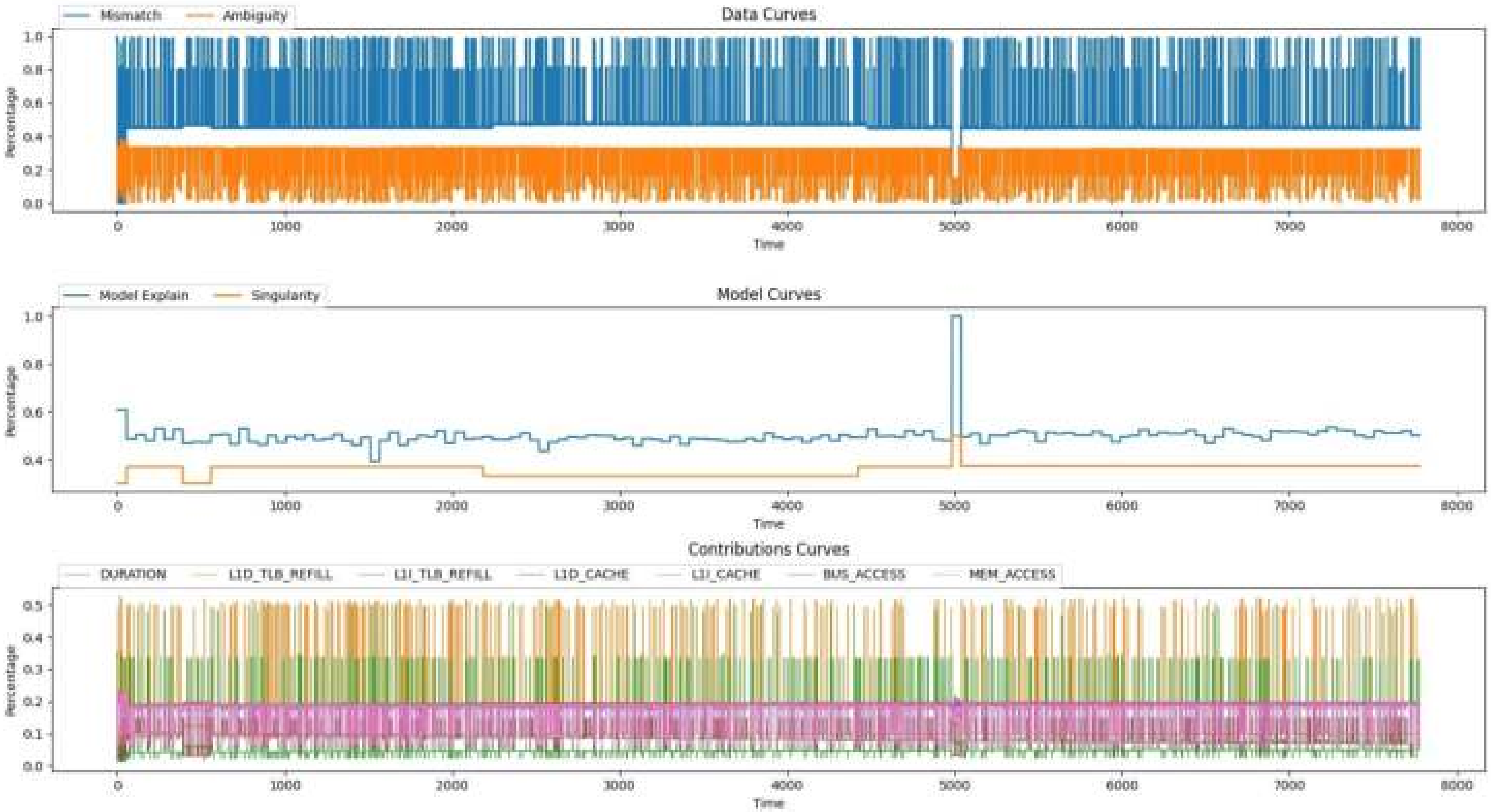
3 – Experiment results

- **Goal: Validate the approach to select the best HPCs to collect**

- **Selected HPCs :**
  - 6 out of 256 available, platform limitation
  - Mainly focusing on memory management

- **Training on 5 healthy datasets (flights of 30 minutes each) :**
  - Cumulative learning time: 8 minutes and 24 seconds
  - Number of models obtained: 17 for a total weight of 3.2 MB

3 – Experiment results



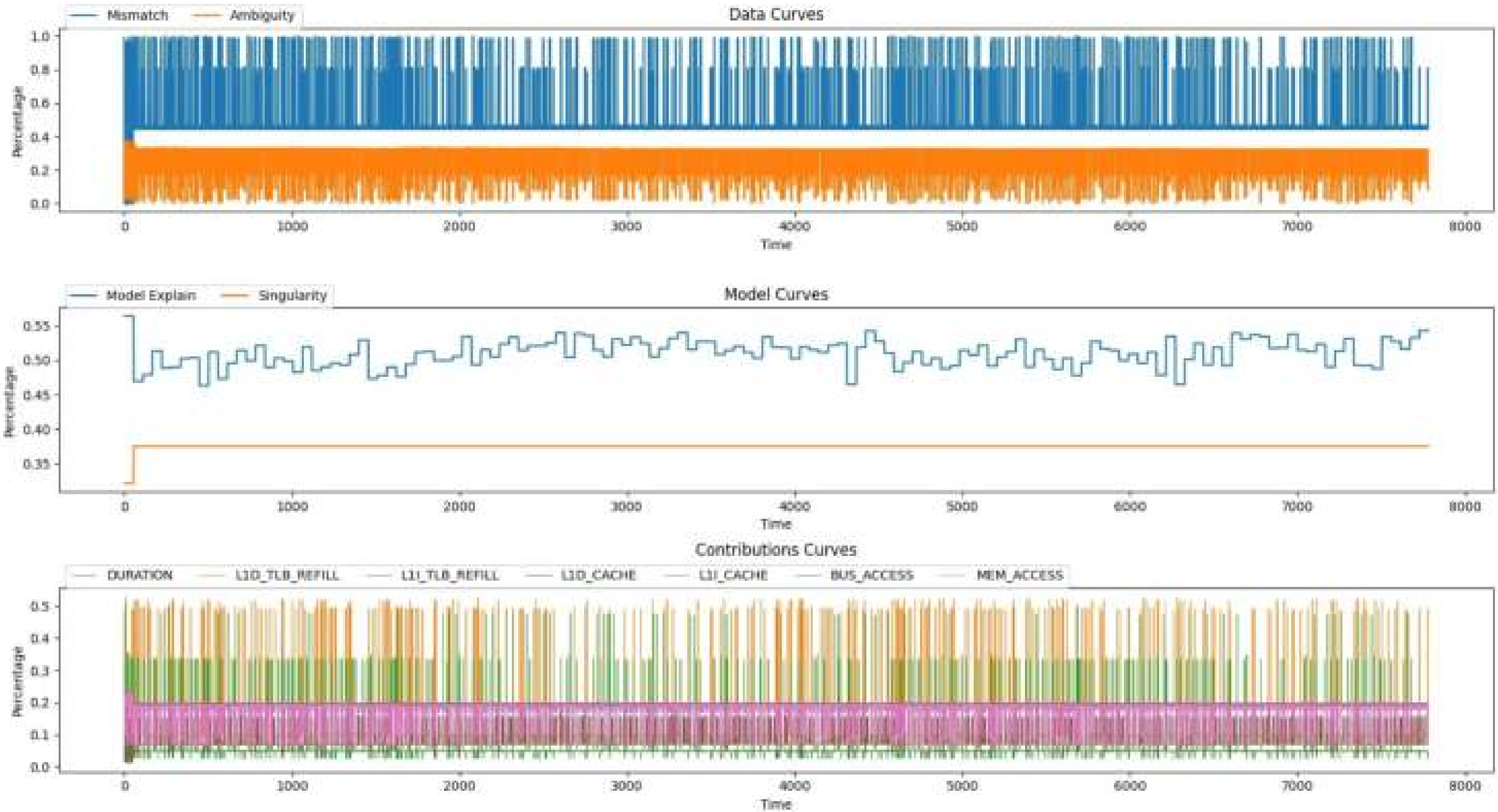
3 – Experiment results

- **Validation of the behavioral basis:**
  - Freeze the models
  - Analysis on 5 unlearned healthy flights (30 mins each)

- **Results:**
  - Cumulative analysis time: 4 minutes and 34 seconds
  - Average explanation: 50%
  - Most interesting HPCs: L1D_TLB_REFILL, L1I_TLB_REFILL

- **The explanation rate is low due to the small training sample.**

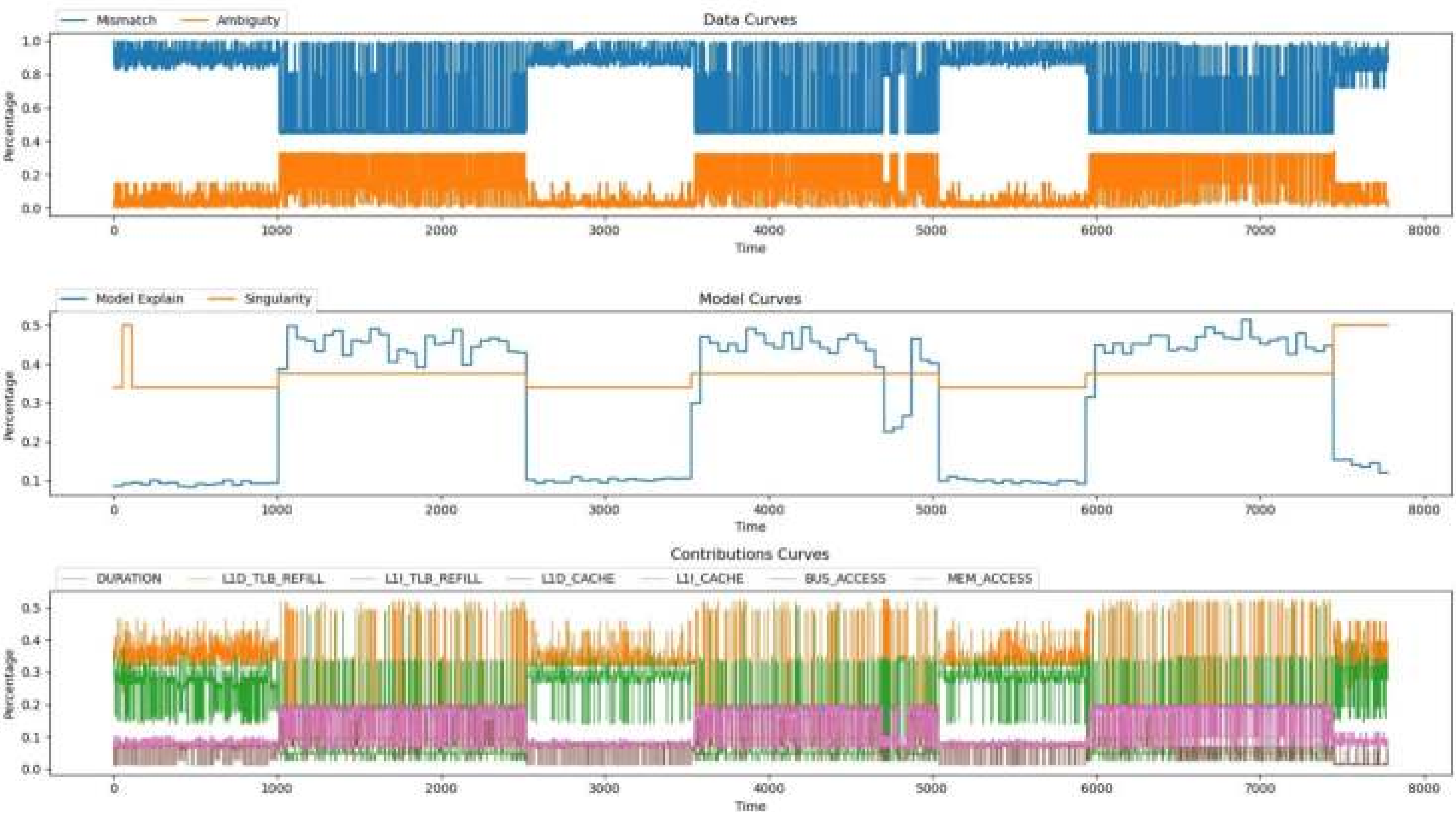
3 – Experiment results



3 – Experiment results

- **Infected flights :**
  - 10 flights (30 mins each)
  - CPU time theft: 10 seconds every 15 seconds
- **Results:**
  - Cumulative analysis time: 4 minutes 39 seconds
  - Average explanation when no attack is present: 50%
  - Anomalies detected: 100%
- **Validation of HPCs :**
  - Top contributors in attack detection: L1D_TLB_REFILL, L1I_TLB_REFILL.
  - Contribution rate: between 30 and 40%

3 – Experiment results



Conclusion

Conclusion

- **Dynamic bayesian networks gives good detection results.**
- **Performance allows use in a critical embedded environment.**
- **Collection module impact is considered tolerable by our avionic partner.**
- **Next steps:**
  - Deepen the learning phase to obtain a better rate of explainability of normal behaviors.
  - Expand the attack spectrum to validate the detection effectiveness.
  - Propose remediation actions when an attack occurs.

Thank you for your attention

Questions

References

[1] Y. Boyer, Étude et conception de méthodes de protection face aux attaques par corruption de mémoire pour systèmes embarqués dans le contexte de l'Internet des Objets, Université Montpellier (2020). URL: https://tel.archives-ouvertes.fr/tel-03378800

[3] A. Damien, Sécurité par analyse comportementale de fonctions embarquées sur plateformes avioniques modulaires intégrées, Theses, INSA de Toulouse, 2020. URL: https://hal.laas.fr/tel-02953842.

[3] M. Bourdon, Détection d'intrusion basée sur l'analyse de compteurs matériels pour des objets connectés, INSA de Toulouse (2021). URL: https://hal.laas.fr/tel-03572845

[4] H. Sayadi, H. M. Makrani, O. Randive, S. Manoj, S. Rafatirad, H. Homayoun, Customized Machine Learning-Based Hardware-Assisted Malware Detection in Embedded Devices, in: 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018, pp. 1685–1688. doi:10.1109/TrustCom/BigDataSE.2018.00251.

[5] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss,W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, Y. Yarom, Spectre Attacks: Exploiting Speculative Execution, in: 2019 IEEE Symposium on Security and Privacy (SP), 2019, pp. 1–19. doi:10.1109/SP.2019.00002.

[6] O. Mutlu, J. S. Kim, RowHammer: A Retrospective, 2019. arXiv:1904.09724.

[7] P. Jattke, V. Van Der Veen, P. Frigo, S. Gunter, K. Razavi, BLACKSMITH: Scalable Rowhammering in the Frequency Domain, in: 2022 IEEE Symposium on Security and Privacy (SP), 2022, pp. 716–734. doi:10.1109/SP46214.2022.9833772.

Appendices

- **The AKHEROS machine learning algorithm is based on the use of dynamic Bayesian networks**
- **Semi-supervised learning**
- **No a priori knowledge**
  - Nor attacks
  - Nor the system to monitor
- **Creation of models of behaviors, non-incongruous and incongruous**
- **Generic approach**
  - IT activities
  - Predictive maintenance
  - Production lines

Appendices

- Our platform has 4 cores clocked at 1.8GHz sharing a 2 MB L2 cache.
- 2 Cores are dedicated to the FMS application, 1 to the collection module and HIDS and the last to execute attacks.
- It is possible to observe 256 different performance counters.
- However, only 6 are observable at the same time for the same core.