Distributed intrusion detection system for CubeSats, based on deep learning packets classification model

Otman Driouch
[1]*Smart Communications Research Team, University Center for Research in Space Technologies, Mohammadia School of Engineers Mohammed 5 University in Rabat.*
[2]*Royal Center for Space Research and Studies.*
Rabat, Morocco
otmandriouch@research.emi.ac.ma

Slimane Bah
[1]*Smart Communications Research Team, University Center for Research in Space Technologies, Mohammadia School of Engineers Mohammed 5 University in Rabat.*
Rabat, Morocco
slimane.bah@emi.ac.ma

Zouhair Guennoun
[1]*Smart Communications Research Team, University Center for Research in Space Technologies, Mohammadia School of Engineers Mohammed 5 University in Rabat.*
Rabat, Morocco
zouhair@emi.ac.ma

***Abstract*—As part of the significant evolution that the space industry is experiencing, a fast increase in the number of CubeSats projects for scientific, commercial and military purposes has been noted in recent years. This acceleration, coupled with the widespread use of Commercial Off-The-Shelf (COTS) components, raises questions about the ability of these systems to withstand potential cyberattacks, which are becoming more prevalent. Thus, the cyber resilience of a CubeSat depends on its ability to effectively detect attacks despite the constraints of autonomy and the limitation of resources that characterize the space missions. To address this need, our paper proposes an Intrusion Detection System (IDS) for CubeSat systems. This distributed solution uses an Artificial Neural Network (ANN) module for classifying CSP packets over CAN on board the space segment based respectively on timestamp and Data field, while the classifier training processes are executed at the ground segment level. The results obtained following the experimentation of this IDS against three types of common attacks are very encouraging thanks to detection rates obtained between 87.66% and 99.59% (F1-score).**

***Index Terms*—Space technology, CubeSat, intrusion detection, cybersecurity, deep learning**

I. Introduction

The landscape of space applications is evolving rapidly with the proliferation of small satellites, particularly CubeSats, marking a new era in the use of space technology for scientific, commercial and educational purposes. As access to space has expanded, so have the potential risks posed by malicious actors seeking to exploit vulnerabilities in CubeSat systems. The latter, characterized by their miniature size and cost-effectiveness, have democratized access to space fostering a surge in satellite deployments. However, this democratization has brought forth a new set of cybersecurity challenges, as the traditionally closed and secure environment of space becomes more accessible. This paradigm shift justifies the implementation of detective and corrective security controls adapted to the evolving cyberthreat landscape in the "New Space" era [1]. Our research focuses on the need for intrusion detection capabilities to safeguard CubeSats against potential cyberattacks. Unlike traditional satellites, CubeSats often operate with extremely limited computational resources and are designed for specific, power-constrained missions [2]. As a result, they may lack security measures inherent in larger satellites, making them susceptible to a range of cyberthreats such as denial of service (DoS) attacks, spoofing, and unauthorized access [3].

In this context, our research aims to explore scenarios of intrusion attempts specific to CubeSats and elucidate the imperative for intrusion detection capabilities tailored to the challenges of this emerging space domain. By understanding different intrusion mechanisms, we can design a well-trained IDS with high detection rates of malicious packets.

This research seeks to contribute to the growing body of knowledge surrounding cybersecurity in space missions:

- Drawing up a panorama of the most likely CubeSat Space Protocol (CSP) attack scenarios and their associated intrusion mechanisms,
- Presenting the design of a distributed IDS that effectively detects these intrusion attempts by classifying CSP packets using an ANN model. This distributed architecture aims to reconcile the detection performance and the constraints related to the limited resources aboard CubeSats.
- And providing insights that are crucial for the development of secure and resilient CubeSat systems. In doing so, we advocate for a proactive approach to cybersecurity in the New Space frontier, emphasizing the importance of integrating intrusion detection capabilities to mitigate the evolving cyber threats facing CubeSats.

The present paper includes six sections: after the introduction, Section II presents an overview of CSP and the most likely CubeSat attack mechanisms. Section III defines the threat model associated to this research, followed by a description of the proposed IDS in Section IV. Experimentation results are presented in Section V. Finally, the conclusion discusses key findings, main challenges and work directions.

Bit offset	31-30	29-25	24-20	19-14	13-8	7-4	3	2	1	0
0	Priority	CSP Source ID	CSP Destination ID	Destination Port	Source Port	Reserved	HMAC	XTEA	RDP	CRC
32	Data (0 – 65535 bytes)									

Fig. 1. CSP Protocol Packet Header Structure.

Bit	1	11	1	1	18	1	1	1	4	0-64	15	1	1	1	7
Name	SOF	Identifier A	SRR	IDE	Identifier B	RTR	FDF	Reserved bit	DLC	Data field	CRC	CRC delimiter	ACK	ACK delimiter	EOF

Fig. 2. CSP over CAN Packet Structure.

II. Background on CSP and related attack mechanisms

In this section, we present an overview of the CubeSat Space Protocol (CSP) and the associated packet format, as these concepts are important to understand security challenges and properly define the most likely attack mechanisms.

A. *Overview of CubeSat Space Protocol*

*1) **CSP protocol** [4]:* is a compact protocol stack implemented in the C programming language. Tailored for simplifying communication among distributed embedded systems within more confined networks like Cubesats, CSP adheres to the TCP/IP model. It encompasses a transport protocol, a routing protocol, and various MAC-layer interfaces in its design. The transport protocol ensures reliable end-to-end communication, while the routing protocol manages the efficient flow of data within the network. The inclusion of various MAC-layer interfaces as Controller Area Network (CAN) and Inter-Integrated Circuit (I2C) allows for compatibility with diverse hardware configurations. The concept revolves around providing Cubesats sub-system developers with the same functionalities as a TCP/IP stack, but without incorporating the substantial overhead associated with the IP header. The limited footprint and simple implementation enable seamless integration into a compact 8-bit system, ensuring full connectivity within the network. This service-oriented approach allows all subsystems to offer their services at the same network level, eliminating the need for a central master node. Its design caters to embedded systems characterized by extremely constrained CPU and memory resources as AVR (8-bits et 32-bits) and ARM (32-bits). The main CSP implementation is created using GNU C and successfully adapted for execution on FreeRTOS, Zephyr, and Linux (POSIX).

*2) **CSP packet structure**:* There are two main versions of CSP: 1.x characterized by a 32-bit Header, and 2.x characterized by a 48-bit Header. The main difference being the size of the source and destination addresses, larger in version 2.x (14 bits) than in 1.x (5 bits) to support a greater number of ECUs. As shown in Fig.1, the CSP packet header used in our university project (Version 1.4) includes a priority field and source/destination addresses and ports. The port range is segmented into three customizable sections. The initial segment, spanning from port 0 to 7, serves general purposes like ping, task list and buffer status and is managed by the CSP service handler. Ports 8 to 47 are designated for services specific to subsystems. The remaining ports, ranging from 48 to 63, are ephemeral and designated for outgoing connections. Additionally, bits 28 to 31 are used for packet marking, encompassing HMAC, XTEA encryption, RDP header, and CRC32 checksum. regarding the data field, it can go up to 65535 bytes.

*3) **CSP over CAN**:* CSP can be configured to communicate using CAN-bus interface, which uses CAN 2.0B extended frame format [5]. CSP Transport Layer takes care of packet fragmentation into CAN-frames of 8 bytes, and only a fully completed packet will arrive at the CSP receiver. The first CAN packet must include CSP packet header and length of the payload. If there is any payload, up to 2 bytes of payload is added to the first CAN packet as well. All the other packets contain only payload of the CSP packet. When communicating via the CSP protocol, the CAN packet consists of the following fields Fig.2:

- **Start of Frame (SOF):** denotes the start of frame transmission,
- **Identifier A:** first part of the (unique) identifier which also represents the message priority:
  - [10..6] - Source CSP ID,
  - [5..1] - Destination CSP ID,
  - [0] - CAN start packet (0) / CAN next packet (1),
- **Substitute Remote Request (SRR):** must be recessive (1),
- **Identifier Extension bit (IDE):** must be recessive (1) for extended frame format with 29-bit identifiers,
- **Identifier B:** second part of the (unique) identifier which also represents the message priority:
  - [17..10] - A number of remaining CAN packets,
  - [9..0] - CFP identification number,
- **Remote Transmission Request (RTR):** must be dominant (0) for data frames and recessive (1) for remote request frames,

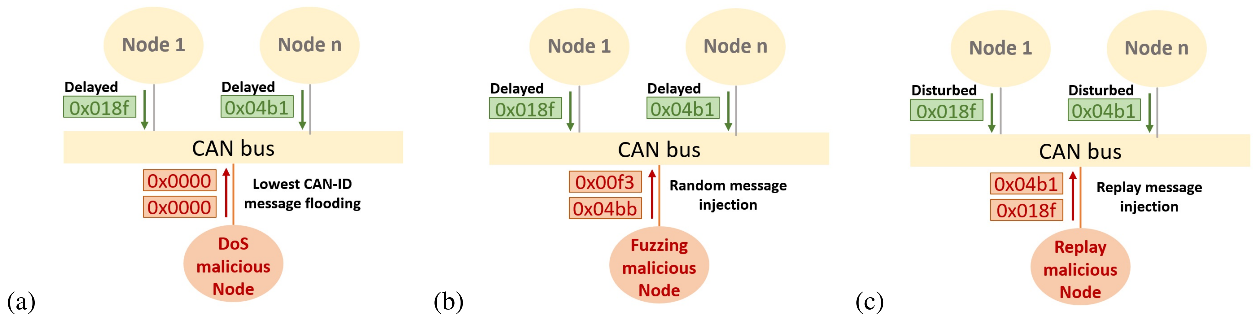

Fig. 3. Common CAN attack mechanisms (a) Message flooding DoS attack - (b) Fuzzy injection attack - (c) Replay attack

- **FD Frame (FDF):** must be dominant (0) to interpret bit sequence as classic and recessive (1) as FD frame,
- **Reserved bit:** reserved bit must be set to dominant (0), but accepted as either dominant or recessive,
- **Data Length Code (DLC):** number of bytes of Data field (0–8 bytes),
- **Data field:** contains the actual data being transmitted. It can range from 0 to 8 bytes,
- **Cyclic Redundancy Check (CRC):** used to check the integrity of the transmitted data and ensure the reliability of CAN communication,
- **Acknowledgment (ACK):** consists of the ACK field and ACK delimiter. If a ECU receives a message without errors, it sends an ACK bit,
- **End of Frame (EOF):** is used for demarcating the end of the message and ensuring that the bus is ready for the next frame.

B. CSP over CAN security challenges and attack mechanisms

Originally designed for real-time, safety-critical applications, CAN lacks authentication and authorization mechanisms, posing challenges in verifying node identity, especially for components like satellite subsystems. This vulnerability opens the door to potential compromise or spoofing of nodes, allowing for the injection of false or malicious data into the network. The repercussions of such compromise are far-reaching, as a single compromised or spoofed node can adversely impact the entire network, leading to issues such as congestion caused by the flooding of the network with a high volume of messages.

The absence of native encryption support in CAN exposes data to potential eavesdropping or tampering. Exploiting this vulnerability, attackers can tamper with identifiers, injecting massive malicious frames with the lowest possible ID (0x000), thereby gaining permanent priority and causing a Denial of Service (DoS) situation Fig.3a [6]. Another prevalent form of attack involves Fuzzing injections, where subtle variations in transmitted messages, such as changes in data payload or alterations in message timing, create confusion or disrupt communication between nodes. Detecting and mitigating these Fuzzy attacks is challenging, as they often elude traditional security mechanisms Fig.3b.

Additionally, an unauthorized node can intercept and maliciously resend previously captured messages within the bus. This replay attack aims to exploit the absence of authentication and encryption in the CAN communication protocol. By retransmitting captured messages, the attacker can manipulate or disrupt critical functions within the system, posing significant risks to safety and security Fig.3c.

III. Threat Model

To develop a solution suitable for intrusion detection within the CSP over CAN, it is crucial to ensure that the design aligns with a practical threat model reflecting plausible attack scenarios within the context of a CubeSat project. As a result, we have identified three primary scenarios deemed most critical based on their likelihood and potential impact [7]. These scenarios pose a risk of unauthorized access to the CubeSat CAN bus:

- Scenario 1: An adversary floods the CSP communication channels with a high volume of malicious traffic, overwhelming the communication system. This can lead to a temporary or permanent disruption in communication between the CubeSat and the ground station, affecting data transmission and reception.
- Scenario 2: An attacker impersonates a legitimate ground station or CubeSat by sending forged CSP messages. This could lead to unauthorized control over the CubeSat or the interception of sensitive information. The attacker might manipulate or inject false data into the communication stream, compromising the integrity of the mission.
- Scenario 3: An adversary intercepts and records legitimate CSP messages and later replays them to deceive the CubeSat or ground station. This could lead to the re-execution of certain commands or actions, causing confusion or disruption to the CubeSat's operations.

These situations need substantial capabilities from an adversary, such as a rival nation or terrorist organization. They have the potential to result in the implementation of attacks that exploit vulnerabilities in the CSP over CAN protocol, aiming to gain unauthorized access to satellite data (through

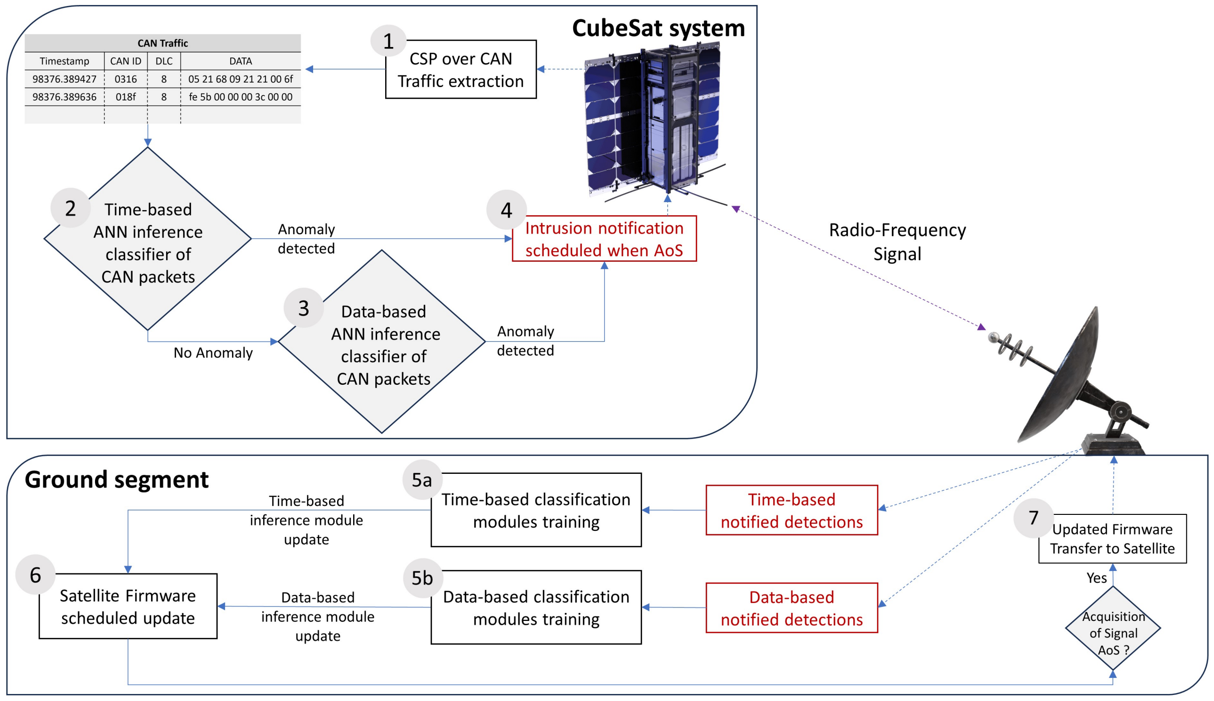
CubeSat system
CAN Traffic
Timestamp
CAN ID
DLC
DATA
98376.389427
0316
8
05 21 68 09 21 21 00 6f
98376.389636
018f
8
fe 5b 00 00 00 3c 00 00
1
CSP over CAN Traffic extraction
2
Time-based ANN inference classifier of CAN packets
Anomaly detected
4
Intrusion notification scheduled when AoS
3
Data-based ANN inference classifier of CAN packets
No Anomaly
Anomaly detected
Radio-Frequency Signal
Ground segment
5a
Time-based classification modules training
Time-based inference module update
Time-based notified detections
5b
Data-based classification modules training
Data-based inference module update
Data-based notified detections
6
Satellite Firmware scheduled update
7
Updated Firmware Transfer to Satellite
Yes
Acquisition of Signal AoS ?

Fig. 4. Functional diagram of the proposed distributed CSP over CAN traffic IDS for CubeSats.

traffic sniffing) or, more significantly, disrupt the CubeSat's operation, making it inoperative through packets injection. Our emphasis on examining the most prevalent attacks on the CAN bus is centered on those that impact the availability of the CubeSat, as DoS packets flooding, fuzzy injection and replay attacks. Confidentiality concerns, on the other hand, can be mitigated by enabling CSP encryption tool based on the XTEA algorithm [8].

IV. Proposed Solution

In this section, we present our proposed IDS for CubeSats, which is a distributed solution based on ANN classification of CSP over CAN packets. We initially provide a general overview of this solution, before describing each functional step, including the developed on-board detection module.

A. General overview

With the aim of designing an IDS adapted to the constraints inherent to CubeSats systems, and as shown in Fig.4, the proposed solution is based on a distributed architecture where the space segment which uses pre-trained ANN inference models, allows classification of CAN packets following two steps: a first time-based detection step then a second which analyzes packets data field if necessary. The ground segment, for its part, makes it possible to conduct all energy-intensive training operations on the basis of the various detections notified by the CubeSat, before scheduling periodic improvements for the on board detection module through firmware updates.

B. Functional description

1. **Traffic extraction**: initially intrusion detection requires capturing the CAN traffic transmitted through the CubeSat bus. An effective approach for this task involves the use of a hardware-based CAN sniffer designed for deployment on a Field Programmable Gate Array (FPGA). It's noteworthy that the same FPGA can be deployed to run the other detection functions needed onboard the space segment. To guarantee an efficient packets extraction, we adopted the CAN traffic sniffing mechanism suggested by [9]. This method relies on the fact that the CAN bus utilizes Non-Return-to-Zero bit coding, employing a simple counting and division technique for synchronization with the clock.
2. **Time-based ANN classifier of CAN packets**: Sniffed CAN traffic is then classified using an ANN inference model that evaluates the arrival time of the packets (timestamps) in combination with the CAN IDs. Before the CubeSat is launched, this module is trained to detect anomalies based on the evaluation of packet frequencies using various representative datasets. The time-based CAN packets classifier has proven its high capacity in detecting DoS flooding attacks and fuzzing injections. The detection rates of this ANN classifier are presented in Section V.
3. **Data-based ANN classifier of CAN packets**: After the classification of CAN packets by the time-based model, a second classification is performed with the aim of broadening the detection spectrum to other

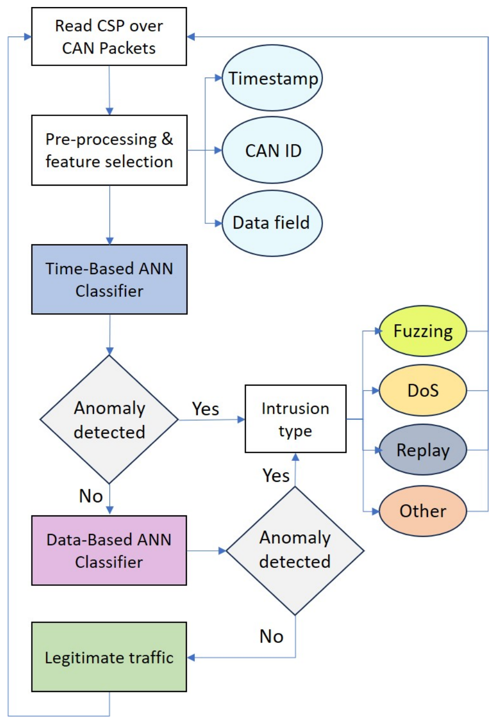

Fig. 5. Overall flowchart of our on-board detection module.

forms of more complex intrusions. These cannot be effectively detected by evaluating packet frequency alone, and therefore require inspection of the data field. Just like the previous model, this classifier based on an ANN must be pre-trained beforehand on the detection of this type of anomalies such as Replay attacks, using suitable Datasets. The significance of our solution is based in the separation between the straightforward analysis of packets frequencies and the more in-depth inspection required for their payload. Additionally, the training process is decentralized at the ground segment level. The fundamental idea behind these modules is to achieve a balance between detection rates and efficiency. The algorithm developed for the on-board time- and data-based detection module is illustrated in the flowchart presented in Fig.5.

4. **Intrusion notification**: When one of the two classifiers detects an anomaly in one or more packets, a notification is directly scheduled so that it is transmitted with high priority to the ground segment during the next pass (Acquisition of Signal AoS).
5. **Time- and Data-based classification modules training**: As soon as they arrive on the ground, malicious packets are directly used to respectively train the time- and Data-based interference engines. This makes it possible to continually improve the performance of the detection modules. The training process initiates with data pre-processing, involving tasks like handling missing values, normalizing features, and encoding categorical variables. The ANN architecture is carefully crafted, incorporating well-suited activation functions (ReLu and Sigmoid functions), loss functions, and optimizers. To counter overfitting, regularization techniques such as dropout and dynamic learning rate planning are employed. Subsequent to these preparations, thorough experimentation ensues. This encompasses fine-tuning hyperparameters, exploring diverse batch sizes (16, 32, 64), epochs (100, 200, 300), and vigilantly monitoring performance indicators with the implementation of early stopping mechanisms. The decentralization of training at the ground station level, which is a highly energy-intensive task, allows for minimizing the operational footprint of the module within the CubeSat. This strategic approach allows continuous improvement through regular firmware updates.
6. **CubeSat Firmware scheduled update**: The distribution of the training process to the ground segment level, known for its high energy consumption, enables the reduction of the workload on the module installed aboard. Subsequently, the system should be maintained through periodic updates. These updates extend to both the time- and data-based modules within the CubeSat, integrated in routine firmware updates. It is essential to highlight that the decision to automatically update a deployed model is triggered by improvements in detection rates especially in the F1-score.
7. **Upload updated Firmware to the CubeSat**: Due to its critical nature, the CubeSat detection module update is implemented as part of scheduled firmware updates, aligning with the specific requirements of the mission and his concept of operation (CONOPS). These enhanced firmware versions are uploaded to the space segment promptly upon ensuring the AoS during the planned update periods.

V. Experimental validation

After building the proposed architecture and implementing the associated classification modules, we conducted experiments with these classifiers through representative datasets of various attack mechanisms related to the adopted threat model (Section III). The main objective is to illustrate the detection rates of time- and data-based classification modules separately and when combined against three common types of CAN attacks.

A. Datasets

To train and assess the performance of the developed detection modules, we employed two datasets that represent three prevalent attacks on the CAN bus. We used CAN network intrusion datasets [10], specifically a DoS attack dataset including 3,665,770 packets, and fuzzy injection dataset with 3,838,859 packets. For Replay attack, we adopted UAVCAN attack dataset [11], consisting of 241,320 packets representing two scenarios of Replay collected in a CAN bus of an Unmanned Aerial Vehicle (UAV).

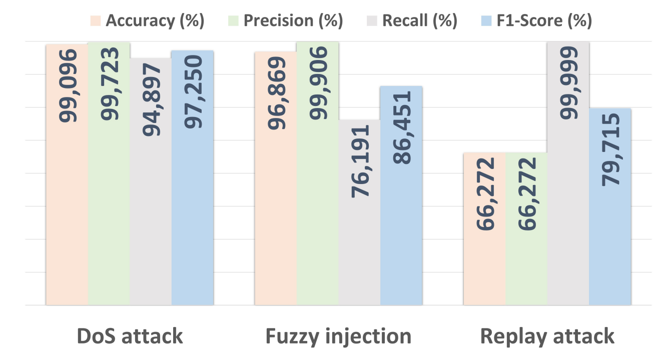

Fig. 6. Detection rates of the Time-based ANN detection module against three common types of CAN attacks.

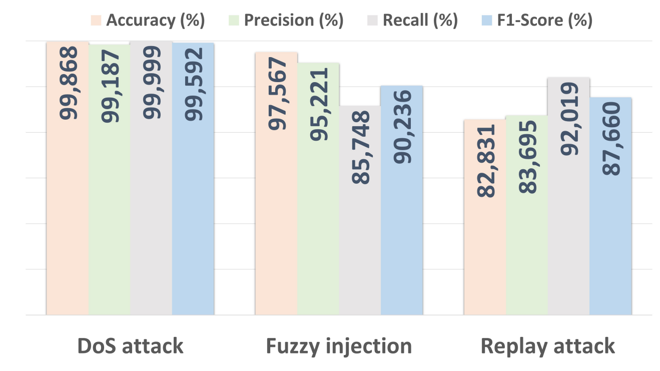

Fig. 7. Detection rates of the Data-based ANN detection module against three common types of CAN attacks.

B. Metrics

To appropriately assess the outcomes of our experiments, we have opted commonly used metrics in the evaluation of classification models [12]. Accuracy (1) represents the ratio of correctly predicted instances to the total instances, providing an overall measure of a model's correctness. Precision (2) measures the accuracy of the positive predictions, indicating the proportion of true positives (TP) among all instances predicted as positive. Recall (3), also known as sensitivity or true positive rate, verifies the ability of a model to identify all relevant instances, highlighting the ratio of true positives to the sum of true positives and false negatives (FN). F1-score(4), a harmonic mean of precision and recall, offers a balanced assessment by considering both false positives (FP) and false negatives. It is particularly useful when there is an uneven distribution between classes or when false positives and false negatives have different implications.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1_{score} = 2 \times \frac{Precesion \times Recall}{Precision + Recall} \tag{4}$$

C. Experimental results

1) Time-based classification results: The ANN time-based detection module is crafted to perform real-time intrusion detection onboard the CubeSat by classifying CAN packets according to their Timestamp and CAN-ID. After the model's training on 66.6% of the packets within the three datasets, the remaining 33.3% was used for testing. As shown in Fig.6, the outcomes reveal the Time-based ANN model's effectiveness in detecting DoS attacks and Fuzzy injections, achieving accuracies of 99.1% and 96,9% and F1-scores of 97,3% and 86,5%, respectively. Nevertheless, the time-based model exhibited limitations in accurately and precisely detecting Replay attacks with an accuracy and precision of 66,3% and F1-score less than 80%.

2) Data-based classification results: The evaluation of data-driven ANN classifier was carried out with the aim of verifying its ability to detect attacks that are difficult to detect by the time-based module such as Replay attacks. Thus, the results obtained were very satisfactory thanks to a detection accuracy of 82.8% and F1-score of 91,1% for Replay attacks Fig.7. The tests also demonstrated a strong ability to detect DoS and Fuzzy attacks in the same way as the time-based model. However, detecting this kind of attacks using the data-driven model would be more expensive in terms of computing power and energy consumption, this is mainly due to the large size of the data field compared to Timestamp and CAN-ID.

3) Combined classification results: Despite the ability of the data-based ANN module to detect the three attacks studied alone, it turns out to be less expensive in terms of computing and energy to initially inspect the sniffed traffic using the time-based module in view of its good performance against DoS and Fuzzing attacks. Only a sample of traffic deemed legitimate by this module will be submitted to the data-based classifier, with the aim of broadening the range of intrusions detectable by the system to include Replay attacks. The evaluation of the combination of the two modules illustrated its effectiveness in detecting all three attack types with high performance. For DoS attacks, the detection rates reached approximately 99.87% in accuracy and 99.59% in F1-score. Fuzzy attacks exhibited detectability at 97.57% accuracy and an F1-score of 90.23%. In the case of Replay attacks, the combined module demonstrated the ability to detect this type with an accuracy of 82.83% and an F1-score of 87.66%. These findings underscore the content-based module's capability to address the limitations of its time-based counterpart, especially in detecting attacks that do not leave traces in the flow, such as replay attacks.

VI. Conclusion

In this paper, we introduce a cost-effective distributed Intrusion Detection System tailored for CubeSats, using a deep learning model for classifying CSP over CAN traffic. the proposed solution encompasses an onboard detection module combining time- and data-based ANN classifiers. All training operations are carried out at the ground segment level. This distributed methodology aims to reduce the computational burden on the onboard detection module while providing comprehensive coverage against a wide range of intrusions.

To evaluate the performance of this IDS, we subjected it to rigorous testing against three prevalent types of CAN attacks. The evaluation outcomes demonstrated remarkable performance, achieving F1-scores of 99.59%, 90.23%, and 87.66% for countering message flooding DoS attacks, fuzzy injections, and replay attacks, respectively.

These findings underscore the significance of integrating Deep Learning models in the classification of CAN traffic. Moreover, the adaptability of our proposed IDS to the specific requirements of CubeSat systems is highlighted through the combination of time- and data-based detection modules, addressing many attack scenarios. Future research endeavors will concentrate on exploring alternative combinations of ML and DL models to further enhance detection rates. Additionally, a comparative analysis of the computational and energy efficiency of various Field-Programmable Gate Arrays (FPGAs) will be conducted to determine the most suitable deployment solution for Cubesat projects. Presently, as part of a 3U university CubeSat initiative, our intention is to implement the onboard module within an FPGA Zynq-7020 System on Chip (SoC) integrated with the Software-Defined Radio (SDR) of the CubeSat. Initial testing of this configuration is conducted on the engineering model, which is an exact replica of the flight model intended for launch in the second half of 2024.

Acknowledgment

This work was carried out in the frame of the cooperation between the Royal Center for Space Research and Studies (CRERS) and the Mohammed V University in Rabat (UM5R).

References

[1] D. Paikowsky, "What Is New Space? The Changing Ecosystem of Global Space Activity," *New Space*, vol. 5, no. 2, pp. 84–88, Jun. 2017. [Online]. Available: https://www.liebertpub.com/doi/10.1089/space.2016.0027

[2] O. Driouch, S. Bah, and Z. Guennoun, "Intrusion detection system for CubeSats: a survey," in *2023 International Wireless Communications and Mobile Computing (IWCMC)*, Jun. 2023, pp. 596–601, iSSN: 2376-6506.

[3] M. Manulis, C. P. Bridges, R. Harrison, V. Sekar, and A. Davis, "Cyber security in New Space: Analysis of threats, key enabling technologies and challenges," *Int. J. Inf. Secur.*, vol. 20, no. 3, pp. 287–311, Jun. 2021. [Online]. Available: https://link.springer.com/10.1007/s10207-020-00503-w

[4] CSP, "The Cubesat Space Protocol — Cubesat Space Protocol." [Online]. Available: https://libcsp.github.io/libcsp/

[5] M. D. Natale, H. Zeng, P. Giusto, and A. Ghosal, *Understanding and Using the Controller Area Network Communication Protocol: Theory and Practice*. Springer Science & Business Media, Jan. 2012. [Online]. Available: https://doi.org/10.1007/978-1-4614-0314-2

[6] Z. Bi, G. Xu, G. Xu, M. Tian, R. Jiang, and S. Zhang, "Intrusion Detection Method for In-Vehicle CAN Bus Based on Message and Time Transfer Matrix," *Security and Communication Networks*, vol. 2022, p. e2554280, Mar. 2022, publisher: Hindawi. [Online]. Available: https://www.hindawi.com/journals/scn/2022/2554280/

[7] O. Driouch, S. Bah, and Z. Guennoun, "A Holistic Approach to Build a Defensible Cybersecurity Architecture for New Space Missions," *New Space*, Aug. 2023, publisher: Mary Ann Liebert, Inc., publishers. [Online]. Available: https://www.liebertpub.com/doi/abs/10.1089/space.2022.0029

[8] J. Yu, G. Khan, and F. Yuan, "XTEA encryption based novel RFID security protocol," in *2011 24th Canadian Conference on Electrical and Computer Engineering(CCECE)*, May 2011, pp. 000 058–000 062, iSSN: 0840-7789. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6030408

[9] N. Jayarathne and M. K. Jayananda, "Development of a field programmable gate array based Controller Area Network sniffer," in *2013 IEEE 8th International Conference on Industrial and Information Systems*, Dec. 2013, pp. 610–615, iSSN: 2164-7011. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6732054

[10] H. Song and H. Kim, "HCRL - CAN network intrusion datasets." [Online]. Available: https://ocslab.hksecurity.net/Datasets/car-hacking-dataset

[11] D. Kim, Y. Song, S. Kwon, H. Kim, J. D. Yoo, and H. K. Kim, "UAVCAN Dataset Description," Dec. 2022, arXiv:2212.09268 [cs]. [Online]. Available: http://arxiv.org/abs/2212.09268

[12] A. Binbusayyis and T. Vaiyapuri, "Identifying and Benchmarking Key Features for Cyber Intrusion Detection: An Ensemble Approach," *IEEE Access*, vol. 7, pp. 106 495–106 513, 2019, conference Name: IEEE Access.