Intrusion Detection System for Software-defined Satellite Networks

Uakomba Uhongora[1], Mamello Thinyane[1], Yee Wei Law[1] and Jill Slay[1,2]
[1]College of Engineering and Information Technology, Adelaide University, Mawson Lakes, Australia
[2]SmartSat CRC, Adelaide, Australia

uakomba.uhongora@adelaide.edu.au
mamello.thinyane@adelaide.edu.au
yeewei.law@adelaide.edu.au
jill.slay@adelaide.edu.au

Abstract: Software-defined networking (SDN) has been proposed as a potential enabler of programmability, flexibility, and dynamic resource allocation in satellite network environments. Despite the benefits, SDN introduces cybersecurity risks to satellite networks, such as controller compromise, flow rule manipulation, and distributed denial-of-service (DDoS) attacks. To address these challenges, this paper explores intrusion detection mechanisms based on anomaly detection techniques in machine learning for detecting cyberattacks in software-defined satellite networks (SDSNs). To support experimentation with the different IDS approaches, an SDSN simulation platform was used to simulate a Walker-Delta satellite constellation for the Earth observation use case. This simulation enables experimentation with solutions that take into consideration the dynamic inter-satellite links (ISLs) and network topology, which most of the existing IDS solutions do not take into consideration. The IDS is implemented on a POX controller leveraging the southbound interface capabilities provided by OpenFlow. The paper reports on the different classification techniques investigated for the IDS, namely, Random Forest, Support Vector Machine, k-nearest neighbour (KNN), convolutional neural network (CNN) - Long Short-Term Memory (LSTM), and multidimensional Matrix Profile. While prior work has demonstrated the efficacy of machine learning and deep learning approaches for SDN, this work further validates the efficacy of the approaches in the context of dynamic topologies characteristic of satellite networks.

Keywords: Software-defined networking, Software-defined satellite networks, Intrusion detection system, Cybersecurity

1. Introduction

Satellite networks are crucial for enhancing services such as remote communication between devices and systems, as well as the extension of coverage to areas without the Internet and cellular networks (Kodheli *et al.*, 2021). Satellite networks support communication systems such as the 5th generation (5G) of mobile networks, which provide services -- such as wireless technology and high-speed internet. Satellite networks are also critical to the military domain for command, control, communications, computers, intelligence, surveillance, and reconnaissance (C4ISR) capabilities (Pavur and Martinovic, 2022). These space applications are supported by the satellite network architecture, which typically consists of three segments – namely, the ground, space and user segments (Manulis et al., 2021).

Space systems are vulnerable to cyberattacks and have become targets due to their importance to critical infrastructure. As satellite constellations in orbit increase, so does the threat landscape (Manulis et al., 2021). The integration of SDN into space systems, in the form of software-defined satellite networks (SDSNs), has the potential to revolutionise how satellite networks are managed, optimised, and secured (Bradbury et al., 2020). Unlike traditional static architectures, SDN enables dynamic, software-driven control of network functions across space, ground, and user segments. This research explores the development of intrusion detection solutions (IDS) for SDSNs. While extensive deployment of IDS has been undertaken in terrestrial networks and in SDNs, the SDSN context presents unique requirements and challenges – such as dynamic ISLs and topology, network disruptions, and resource limitations – warranting new research to further the cybersecurity of this context.

The rest of this paper is organised as follows: Section 2 provides an overview of existing IDS solutions and approaches, including rule-based and behaviour-based IDSs. Section 3 discusses the design approach applied in this work, covering the design science research approach, the architecture of the IDS developed in this work and the dataset generation. The various ML and DL algorithms explored in this work are discussed in Section 4, followed by a discussion of the experimental results in Section 5. Section 6 provides a conclusion of this paper.

2. Background and Context: Intrusion Detection Systems

Intrusion detection systems monitor computing and network environments to detect actions that deviate from expected benign behaviour, which may indicate security breaches. Initially, to detect attacks in the 1970s and early 1980s, system administrators had to print out system logs on paper and manually audit the printout

(Kemmerer and Vigna, 2002). The process was clearly reliant on the auditors' expertise and too slow to detect attacks in progress. In the 1980s, storage became cheaper, and intrusion detection programs became available for analysing audit logs online. The programs, however, could only be run at night when the system's user load was low (Kemmerer and Vigna, 2002). Thus, detecting attacks in time remained a challenge. By the early 1990s, real-time IDSs that analysed audit logs were produced and became available (Kemmerer and Vigna, 2002).

Contemporary IDSs can be categorised along the following multiple axes, which are discussed is detailed hereafter:

- placement – where there are host-based IDS (HIDS), network-based IDS (NIDS), or distributed
- data granularity – concerning flow-based or packet-based IDS
- detection methodology – where rule/signature-based, anomaly/behaviour-based, or hybrid approaches can be used

With regards to IDS placements, HIDSs are implemented on individual hosts to monitor activities on a host, including system logs and policy violations (Ahmad et al., 2021). In contrast, NIDSs monitor traffic in the network to classify normal traffic from malicious traffic (Li et al., 2020) and these IDSs can either be rule-based or behaviour-based (Ayachi et al., 2020). Distributed IDSs use multiple coordinated IDS agents that communicate to provide advanced network monitoring (Folino and Sabatino, 2016).

There are various architectural approaches for implementing IDSs, categorised by granularity: flow-level and packet-level. The flow-level approach analyses aggregate traffic flows rather than individual packets to detect cyberattacks (Yang, Arshad and Zhao, 2022). Packet-level information contains messages generated in the application layer (protocols such as HTTP), as well as headers generated in the lower network layers (TCP and UDP protocols) and the physical layer (e.g., Ethernet headers) (Yang, Arshad and Zhao, 2022). The packet-level approach provides granular visibility into raw traffic, similar to data captured by *tcpdump*. However, there is no aggregate analysis of the packets in this approach, unlike the flow-level approach, which aggregates the raw packet data capture, based on source and destination addresses, port, and protocol, with flow-level analysis through tools such as CICFlowMeter (Sharafaldin, Habibi Lashkari and Ghorbani, 2019) or NTLFlowLyzer (Shafi, Lashkari and Roudsari, 2025).

2.1 IDS Classification Approaches

Several approaches are employed for the classification of network traffic in IDSs. Rule-based IDSs, also called signature-based IDS, rely on rules that specify the conditions of legitimacy of an event, typically with the help of signatures. Signatures can be:

- exploit-facing, for example, malicious byte sequences, email subject headings associated with phishing, email attachments containing executable binaries, traffic going to known malicious domains, scanning of file hashes; or
- vulnerability-facing, for example, SSH requests specifying a vulnerable version number, system log entries indicating audit disablement (Scarfone and Mell, 2007).

An example of rule-based detection is stateful protocol analysis or deep packet inspection. This process analyses protocols at the application layer to compare vendor-developed profiles of benign protocol activity against observed events to identify deviations (Scarfone and Hoffman, 2009). Rule-based detection is effective against known threats that can be expressed using a set of rules and signatures; however, it is ineffective against known threats that are hard to capture using a finite set of rules and signatures (for example, multi-stage attacks), as well as previously unknown threats.

The second classification approach is behaviour-based IDSs, also known as anomaly-based IDSs. This type of IDS applies ML techniques to determine whether a user or component behaviour is anomalous. The approach depends on the prior establishment of a baseline behavioural model and can detect previously unknown attacks if the attacks manifest as anomalies relative to the baseline model. The approach is effective at detection but tends to produce excessive false positives. Traditionally, ML was only used for anomaly detection, but it is now also used for attack classification. Among the ML algorithms used, those that are based on deep learning (DL) are becoming mainstream. Lastly, hybrid IDS combine signature-based and anomaly-based methods to reduce false positives while improving the chance of zero-day attack detection (Maseno, Wang and Xing, 2022). Optimal integration of supervised learning and unsupervised learning algorithms into a hybrid IDS for detecting multistage attacks in a given network architecture remains an open problem.

2.2 Key Considerations for IDS in SDSN

There is a dearth of prior studies that have investigated the adoption of IDS in SDSNs. The implementation of IDSs in SDSN must consider the dynamic nature of ISLs, as well as the placement of the IDS within SDSNs – this is because of the underlying controller placement problem (CPP) in SDSNs (Uddin and Kumar, 2023a). Prior work in this area has utilised ML techniques such as federated learning-based IDS to detect malicious traffic in SDSN, without the consideration of the placement of the IDS in SDSN (Uddin and Kumar, 2023b).

In this research, the main architectural decision was to leverage the SDN controller for the implementation of an IDS within SDSN. The motivation for this is that the role of the controller within an SDN gives it full visibility into the satellite network topology and traffic dynamics, as well as the ability to control the overall behaviour of the network. This approach gives access to the full raw network traffic, which can subsequently be analysed to inform attack detection and routing decisions.

3. Implementation of IDS for SDSN

The implementation of the IDS in this research employed the design science research (DSR) approach and leveraged an SDSN simulation platform, an SDN-based space system framework for simulation (S3FS), to enable the simulation of different constellation orbital dynamics and network traffic and the generation of benign and attack datasets to train the IDS classifiers.

3.1 Design Science Research Approach and SDSN Simulation Environment

The DSR approach provides a suitable methodology for creating technical artefacts (e.g. a system or model) that address a problem and contribute to knowledge (Hevner et al., 2004). The DSR process is conceptualised through three interlinked cycles:

1. Relevance cycle: focuses on the problem context and requirements for the ML-based IDS. The relevance cycle identifies the requirements for an IDS in SDSNs, such as timely anomalous traffic detection and the ability to be integrated in the SDSN environment.
2. Design cycle: comprises the iterative process of design, refinement (e.g. features and hyperparameters) and evaluation of the ML-based IDS.
3. Rigour cycle: based on applying existing knowledge and methodologies to inform the design process (Vom Brocke, Hevner and Maedche, 2020).

While the relevance cycle was supported by general insights and knowledge within the SDSN context, the technical environment that enabled the simulation of satellite constellations and their associated SDN-supported network dynamics was provided by S3FS, which was developed as part of a broader project in this research (Uhongora, Thinyane and Law, 2025). S3FS is a controlled environment for simulating orbital motion, dynamic network topologies, SDN controller deployment, and attack scenarios. S3FS facilitates the collection of network traffic, including benign and malicious traffic within the SDSN context, enabling realistic simulation of satellite cyberattack scenarios. This framework provided a suitable basis for the development of the IDS presented in this paper.

3.2 Architecture of ML-based Intrusion Detection System

The implementation of the ML-based IDS leverages SDN's centralised network visibility to monitor traffic and detect cyberattacks, as well as SDN's centralised control and management functions (Kodheli et al., 2021). The architecture consists of three key components: a POX controller, OpenFlow switches in the data plane, and the ML-based classifier embedded within the controller (illustrated in Figure 1). The POX controller manages the flow rules in the OpenFlow switch tables and inspects all flows and packets arriving at the controller. The OpenFlow switches forward the flows in the data plane based on the rules installed by the POX controller in the flow tables. The ML-based IDS implemented at the POX controller processes the flows and *Packet-In* messages arriving at the controller and classifies them as normal or malicious. As depicted in Figure 1, setting up the IDS follows this order: network traffic collection, preprocessing, ML classifier training, and real-time classification using the trained ML classifiers.

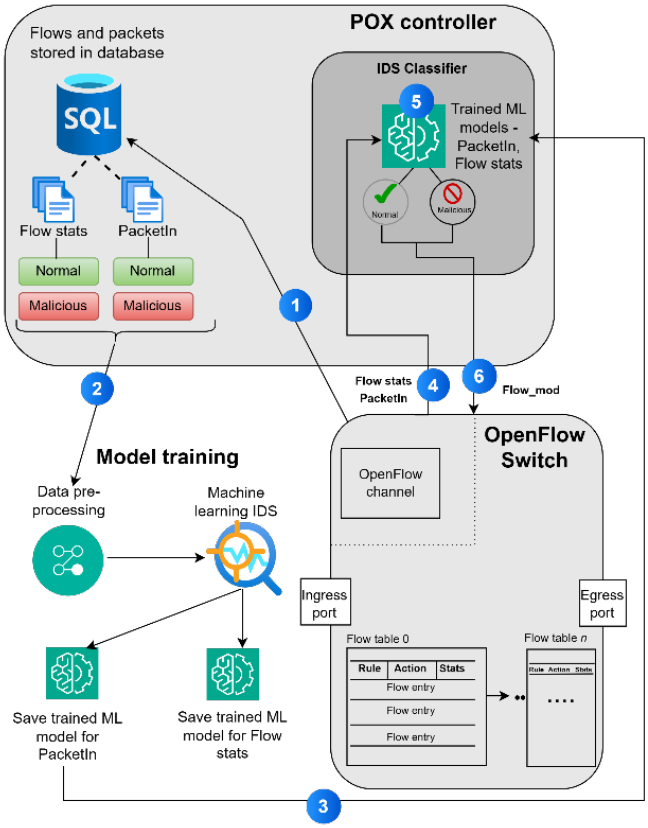


Figure 1: An ML-based IDS for SDSN

Step 1

First, the normal and attack traffic datasets are generated based on the flows_stats and Packet-In messages sent from the switches to the controller. Packet-In messages are sent to the controller when a table miss is triggered on the switch or based on flow entry rules. Further, the IDS periodically requests aggregate flow statistics from the switches and stores this as the flow_stats dataset.

Step 2

The two datasets produced in Step 1 are used to train the IDS classification model. This process entails data preprocessing, model training, and model evaluation. For the data preprocessing, to avoid overfitting, features such as source and destination of IP and MAC addresses were removed from the dataset. Additionally, the data is standardised to scale the values between 0 and 1. Further, the data is split into 80% training data and 20% test data. Multiple classification models were investigated and are discussed later (Section 4).

Step 3

The trained ML models from the previous step are saved and then loaded into the ML-based IDS component for traffic classification and intrusion detection.

Step 4

The S3FS simulation is set to run in various modes across different simulation runs, initially to run in the normal traffic generation mode and then in the attack mode (as discussed in Section 3.3). In this step, the Packet-In and flow_stats messages are not stored as a dataset but rather fed into the IDS for classification.

Step 5

This is the real-time classification step using the trained models and represents the core of the IDS functionality. When flow_stats and Packet-In messages arrive at the POX controller, the IDS is triggered to classify (i.e. normal or malicious) the packets and flows. The details of this step are discussed in the next step.

Step 6

This step extends the IDS component with intrusion prevention functionality. Once a classification is made on a flow or packet, a flow_mod message is sent to the switch with instructions on how to handle the flow or packet (i.e. to drop the packets for a set duration of time) in cases of malicious traffic – these steps are presented in the pseudocode in Algorithm 1. The IDS does not send flow_mod decisions for normal traffic because it is implemented in concert with other standard POX modules that handle topology discovery and routing within the network, namely the forwarding.l2_learning and the forwarding.spanning_tree components. To ensure that the IDS flow modifications take precedence over the instructions from the other components, the flow_mod messages are sent with a high priority value.

Algorithm 1: Proposed ML-Based IDS on POX Controller

Input: S = Set of switches; F_s = Flow statistics; M = Trained ML model; i = Number of monitoring intervals
Output: A = Detection Accuracy; Normal flows forwarded; Malicious flows dropped

```
Initialise POX controller
Load trained machine learning model M
Connect to switches in S
for each monitoring interval i do
    Send flow_stats_request to all switches in S
    Receive F_s (flow entries: source IP, destination IP, packets, bytes, duration, etc.)
    for each flow f in F_s do
        Extract flow features and form feature vector X_f
        Preprocess X_f (normalisation, encoding, missing values handling)
        P_f ← M.predict(X_f)                ▷ Classify flow using machine learning model
        if P_f = Malicious then
            Install FlowMod rule on switch to drop flow f
        end if
    end for
end for
Compute detection accuracy A based on correctly classified flows
return A
```

3.3 Intrusion Detection Dataset Generation

The S3FS framework is used to generate normal and attack traffic used for the ML training process. The malicious traffic is generated for the following attacks: SYN flood (single source), SYN flood from random sources (hereafter SYN flow flood to distinguish SYN attacks), smurf DDoS attacks, and port scan. The final prepared datasets used for training the classifiers are multiclass datasets that consist of normal and various malicious traffic, as depicted in Figure 2. This dataset contains 15 features, 8,286 normal samples and 36,728 malicious samples, consisting of the following traffic classes:

- Benign: 8,286
- SYN flood: 10,144
- SYN flow flood: 7,894
- Port scan: 10,436
- Smurf: 8,254

The multidimensional Matrix Profile (MP) time series anomaly detection method approach requires a smaller ratio of the anomalous traffic compared to KNN, RF, SVM and hybrid CNN-LSTM classifiers. Therefore, to train the MP, the dataset contained 90% normal traffic and 10% malicious traffic.

Normal traffic is generated using the *Netcat* network utility tool, simulating an Earth observation payload from the satellites to the ground station, and the same tool is used on the ground station to receive the transmitted traffic. The different attack types were generated using *hping3* and *nmap* tools and directed towards the ground station host from the attack host, with both hosts connected to the same switch. The simulated DDoS attacks compromise SDSN by flooding the controller with a large volume of traffic to overwhelm the controller and exhaust network resources. For example, the SYN flood attack from random sources floods the ground station with SYN requests, and in response, the ground station sends a SYN-ACK message back via the switch. As the switch does not have matching rules for the spoofed addresses, it forwards a *Packet-In* message to the controller to determine how to handle the packets. The controller then replies with a *Packet-Out* or *flow_mod* message, instructing the switch on how to process the incoming traffic. However, due to the speed and volume of packets

requiring processing, the switch's flow tables rapidly become saturated, preventing new flow entries sent by the controller from being installed.

The attacks were generated using the following commands were executed:

- SYN flood: hping3 -c 100000 -i u1000 -d 120 -S -w 64 -p < *target_port* > < *target_ip* >
- SYN flood (random sources): hping3 -c 100000 -U -d 120 -S -w 64 -p < *target_port* > –flood –rand-source < *target_ip* >
- Port scan: nmap -p 1-65535 < *target_ip* >
- Smurf: hping3 –icmp –flood -a < *target_ip* > < *broadcast_address* >

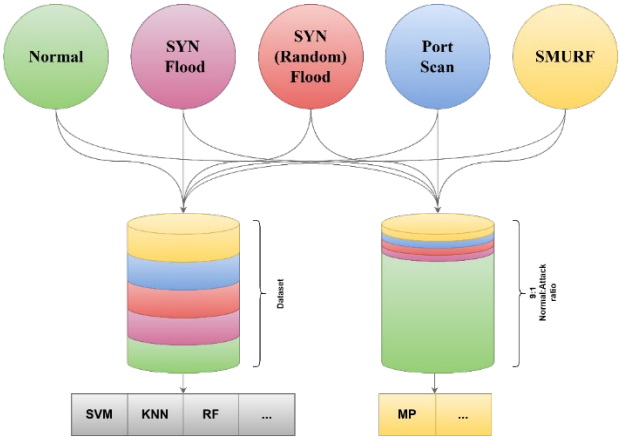


Figure 2: Dataset class ratios for classifier model training

4. Evaluating IDS Classification Models

First, the datasets undergo preprocessing steps such as standardisation, labelling (normal and attack traffic) and feature selection. Second, several approaches were investigated for the implementation of the classifier model. These approaches are supervised ML techniques: RF, SVM, KNN, hybrid CNN-LSTM, and a time series anomaly detection algorithm: MP, which is implemented in the STUMPY Python library. These approaches are discussed hereafter.

4.1 Random Forest

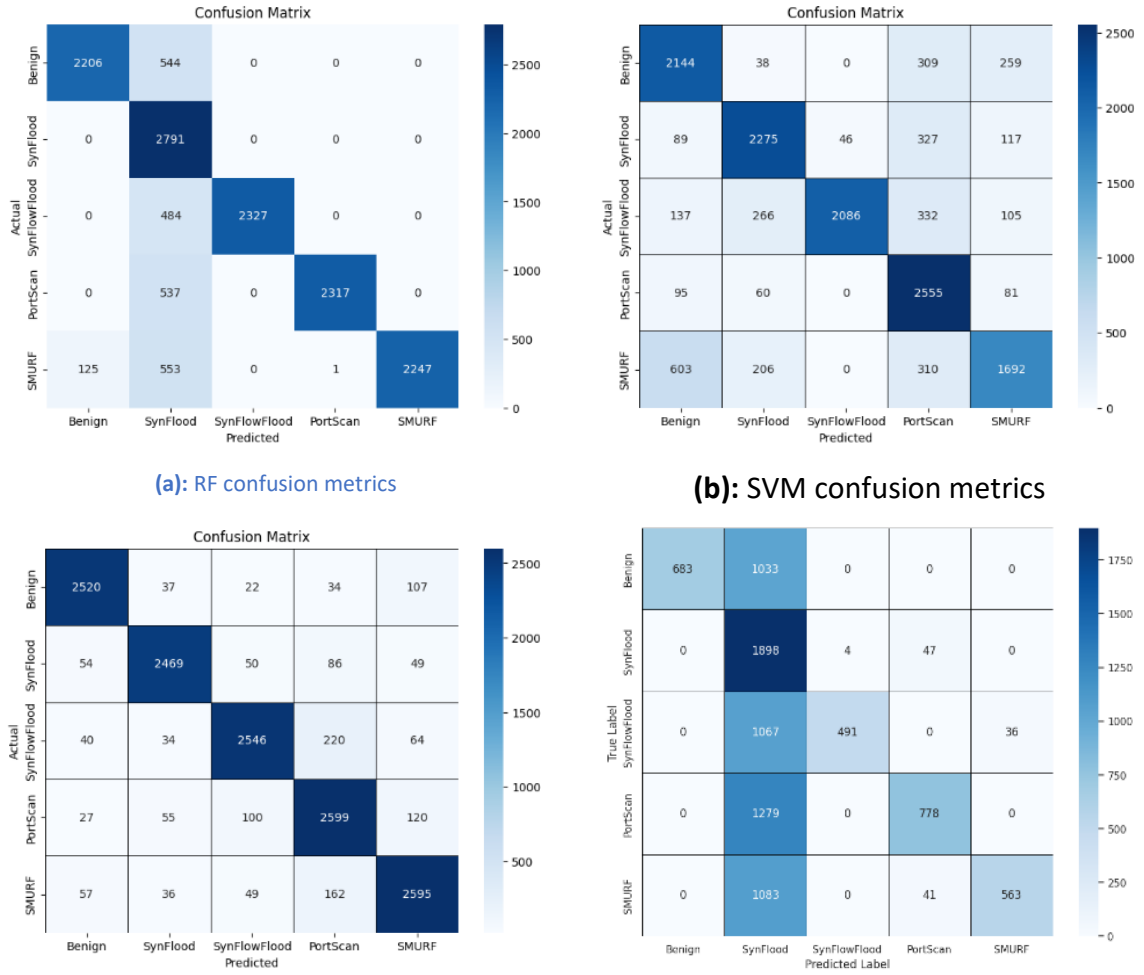
Random forest (RF) is an ensemble ML algorithm that builds multiple decision trees and combines their results, which ultimately improves prediction accuracy (Aslam, Srivastava and Gore, 2022). In this research, the simulation setup of the RF classifier contains 100 trees (n_estimators=100) with a maximum depth of 15 (max_depth). The random state is set to 0. The RF classifier was trained on the flows and packets datasets separately. The feature selection process was performed using the *feature_importance_* attribute, with the features selected from the dataset having a feature importance score greater than 0.01. The results obtained from the flows dataset showed that RF achieved an accuracy of 84% accuracy. Further evaluation of the model was undertaken on real-time data; this is discussed in Section 5.

4.2 Support Vector Machine

Support Vector Machine (SVM) is an ML algorithm that works by finding an optimal hyperplane that separates the data into different classes (Najafimehr, Zarifzadeh and Mostafavi, 2023). SVM uses various kernels such as the Radial Basis Function (RBF) and polynomials to handle nonlinearity and to control the margin trade-offs. SVM, as implemented in this research, is configured with the following hyperparameters: linear kernel and regularisation parameter of 1.0. The SVM classifier achieved an accuracy of 77%.

4.3 k-Nearest Neighbour

k-Nearest Neighbour (KNN) is a supervised ML algorithm where classification is performed by a majority vote among k-nearest data points. The classifier calculates the distance (i.e. Euclidean, Manhattan distance) between new data samples and the learned samples in a dataset to classify data samples (Najafimehr, Zarifzadeh and Mostafavi, 2023). In this study's implementation, the KNN classifier is configured with five nearest neighbours, which means the KNN classifier uses five nearest neighbours for performing predictions. KNN achieved a classification accuracy of 90%.



(a): RF confusion metrics

(b): SVM confusion metrics

(c): KNN confusion metrics

(d): Hybrid CNN-LSTM model confusion metrics

Figure 3: Confusion metrics of IDS classification models

4.4 Multidimensional Matrix Profile

A time-series anomaly detection approach using multidimensional MP was also explored in this work to identify attack traffic. The algorithm uses a threshold to distinguish benign traffic from attack traffic. The MP produced a high percentage (28.56%) of misclassified samples. Unsupervised anomaly detection methods based on the matrix profile produced too many false positives and false negatives.

4.5 Hybrid CNN-LSTM Model

This work also investigated DL models by using a hybrid CNN-LSTM model to classify the traffic. Hybrid DL models have proven to have better detection accuracy compared to individual implementations of the models (Abdallah et al., 2021). For hybrid architecture, CNN performs spatial features extraction, while LSTM extracts the temporal features to develop an IDS model that lowers false alarm rates and improves accuracy (Abdallah et al., 2021; Halbouni et al., 2022). The CNN architecture consists of the following parameters: an input layer, which loads the dataset, a 1-dimension convolution with 128 filters, a max pooling layer, a dropout layer with a probability of 0.3, an LSTM layer, an additional dropout layer, a dense layer hidden layer fully connected with 16 neurons, an activation function using ReLU followed by another dropout layer. The output layer is the classification layer, and it is a dense layer consisting of 5 output neurons for the multi-class dataset used in this work, with a *softmax* activation function. The hybrid CNN-LSTM model achieved an accuracy of 49% running over 20 epochs.

Figure 3 shows the confusion metrics of the ML and DL models during training. Although DL is used for IDS in a lot of existing studies, it yields very low performance in terms of detection accuracy with the dataset generated in this research. As depicted in Figure 3, the true positive values for the attack detection vary with each classifier.

For instance, the RF classifier provides a better accuracy at detecting the SYN flood attack, whereas the KNN classifier performs better across the rest of the attack types.

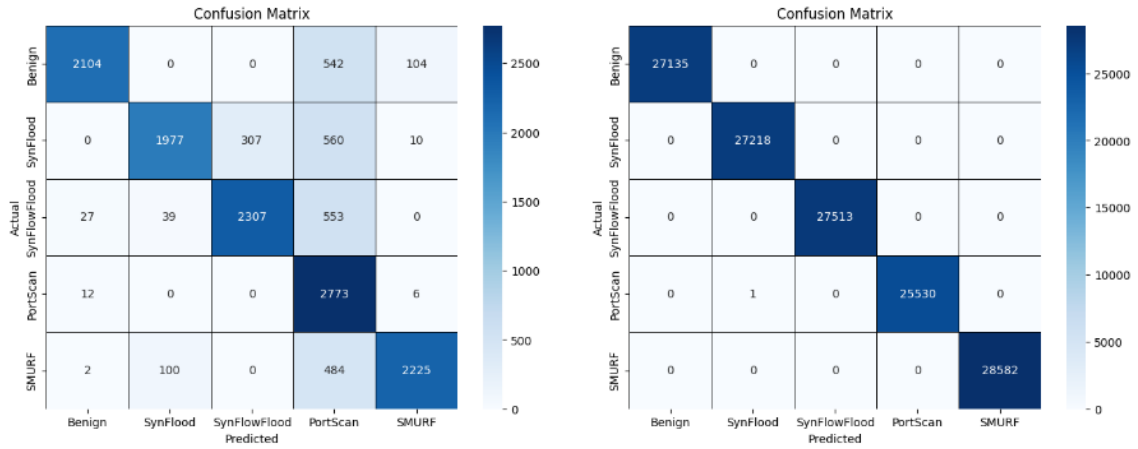
The classifiers investigated in this work are trained on the four attack patterns and therefore their performance is not optimised for other attack types that the classifiers were not trained on. Additionally, the ML-based IDS classification accuracy could be affected if the IDS is deployed in a different environment, i.e., the number of satellites and dynamic ISL connectivity. However, the IDS approach for SDSNs proposed in this research can be generalised to other similar contexts, with ML and DL classifiers optimised for those context-specific requirements.

5. Discussion

Considering the dependency of critical infrastructure on space systems, security controls such as IDSs are required to protect the space ecosystem from cyber threats. Unfortunately, traditional security measures have been proven inadequate to prevent powerful and advanced threats that target SDSNs (Chen et al., 2023). Uddin and Kumar (2023) proposed a federated learning-based IDS to counter cyberattacks in SDSNs. However, in this work, the network topology does not comprise dynamic ISLs, which is of interest in our work and increasingly characteristic of LEO constellations. The work implemented in this paper leveraged the S3FS platform, which is configured to simulate network dynamics associated with a 20-satellite Walker Delta constellation within an Earth observation use case. The dynamic ISLs present a unique challenge that needs to be addressed in the implementation of IDSs for SDNSs. The findings in this research show that a suitable IDS placed on the controller can be effective in detecting and preventing malicious DDOS traffic.

For the IDS in this research, the flow-level classification approach was adopted, primarily motivated by the fact that flow-level traffic statistics provide a better characterisation of network traffic behaviour. Further, the flow-level classification can easily be integrated into an SDN architecture since the traffic collection, classification, and mitigation responses can be handled directly within the SDN controller components (for POX through the *_handle_FlowStatsReceived()* event handler) and in applications (for ONOS). While packet-level datasets provide granular and detailed features, there is no aggregate analysis of the packets, which would give a better picture of the traffic patterns. Figure 4 illustrates the performance of the RF model against flow-level vs packet-level datasets, showing a better performance for flow-level data. While the confusion metrics suggest a high accuracy of this dataset with the RF model, this represents an overfitting of the model to the dataset, which would affect its ability to generalise to other new and unseen scenarios.

While prior research has noted the suitability of packet-level approaches for IDS implementation, particularly for detecting known attacks (i.e. signature-based or payload-based attacks) (Yang, Arshad and Zhao, 2022), in this work, this approach did not yield a satisfactory performance. This result is despite this approach providing a larger set of features (i.e. 43) compared to the flow-level approach. This research has demonstrated the feasibility of the flow-level intrusion detection leveraging an RF model for the classification of traffic across the benign and attack classes within SDSNs.



(a): RF confusion metrics from flow-level dataset

(b): RF confusion metrics from packet-level dataset

Figure 4: Confusion metrics of RF classification

Despite the demonstrated effectiveness of the architectural approach presented in this paper, it, however, inherits the limitations of centralised SDSN architectures, namely a single point of failure associated with the SDN controller. The centralised controller introduces new vulnerabilities, which, when exploited, can compromise the entire network. The DDoS attack is one of the most prominent attacks that targets the SDN controller (Chen *et al.*, 2023). Future work will investigate decentralised SDN controller placement in SDSNS to reduce the single point of failure vulnerabilities and, therefore, the suitability of decentralised IDS architecture.

6. Conclusion

This paper presented an ML-based IDS for the detection and mitigation of cyberattacks in SDSNs. This was achieved by investigating various ML-based IDSs for SDSNs following the DSR process. Building on the S3FS simulation platform developed in (Uhongora, Thinyane and Law, 2025), this work demonstrates how anomaly/behaviour-based intrusion-detection mechanisms can identify and mitigate cyber threats in real time within the constraints of dynamic orbital communications in SDSNs. Various classifiers were trained and analysed, namely, RF, KNN, SVM, MP and hybrid CNN-LSTM. From these classifiers, RF was deployed in the IDS on the POX controller to detect flow-level malicious traffic. This study advances scholarship on cybersecurity for SDSN and specifically demonstrates the feasibility of developing and implementing an ML-based IDS, with intrusion prevention capabilities, for SDSNs.

Acknowledgements

This work has been supported by the SmartSat CRC, whose activities are funded by the Australian Government's CRC Program. The first author is also supported by the Schlumberger Foundation.

Ethics and AI Declaration: Ethical clearance was not required for this research. No AI tools were used in the creation of this paper.

References

Abdallah, M., An Le Khac, N., Jahromi, H. and Delia Jurcut, A. (2021) 'A Hybrid CNN-LSTM Based Approach for Anomaly Detection Systems in SDNs', *Proceedings of the 16th International Conference on Availability, Reliability and Security*. *ARES 2021: The 16th International Conference on Availability, Reliability and Security*, Vienna Austria: ACM, pp. 1–7. Available at: https://doi.org/10.1145/3465481.3469190.

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J. and Ahmad, F. (2021) 'Network intrusion detection system: A systematic study of machine learning and deep learning approaches', *Transactions on Emerging Telecommunications Technologies*, 32(1), p. e4150. Available at: https://doi.org/10.1002/ett.4150.

Aslam, N., Srivastava, S. and Gore, M.M. (2022) 'ONOS Flood Defender: An Intelligent Approach to Mitigate DDoS Attack in SDN', *Transactions on Emerging Telecommunications Technologies*, 33(9), p. e4534. Available at: https://doi.org/10.1002/ett.4534.

Ayachi, Y., Mellah, Y., Berrich, J. and Bouchentouf, T. (2020) 'Increasing the Performance of an IDS using ANN model on the realistic cyber dataset CSE-CIC-IDS2018', *2020 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*. *2020 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, Marrakech, Morocco: IEEE, pp. 1–4. Available at: https://doi.org/10.1109/ISAECT50560.2020.9523662.

Bradbury, M., Maple, C., Yuan, H., Atmaca, U.I. and Cannizzaro, S. (2020) 'Identifying Attack Surfaces in the Evolving Space Industry Using Reference Architectures', *2020 IEEE Aerospace Conference*. *2020 IEEE Aerospace Conference*, Big Sky, MT, USA: IEEE, pp. 1–20. Available at: https://doi.org/10.1109/AERO47225.2020.9172785.

Chen, S.-J., Wang, T.-Y., Fang, C.-S. and Chiang, I.-W. (2023) 'Security assessment of Low Earth Orbit (LEO) with Software-Defined Networking (SDN) structure', *2023 IEEE 6th International Conference on Knowledge Innovation and Invention (ICKII)*, pp. 620–624. Available at: https://doi.org/10.1109/ickii58656.2023.10332792.

Folino, G. and Sabatino, P. (2016) 'Ensemble based collaborative and distributed intrusion detection systems: A survey', *Journal of Network and Computer Applications*, 66, pp. 1–16. Available at: https://doi.org/10.1016/j.jnca.2016.03.011.

Halbouni, A., Gunawan, T.S., Habaebi, M.H., Halbouni, M., Kartiwi, M. and Ahmad, R. (2022) 'CNN-LSTM: Hybrid Deep Neural Network for Network Intrusion Detection System', *IEEE Access*, 10, pp. 99837–99849. Available at: https://doi.org/10.1109/ACCESS.2022.3206425.

Hevner, A.R., March, S.T., Park, J. and Ram, S. (2004) 'Design science in Information Systems research', *MIS Quarterly*, 28(1), pp. 75–105.

Kemmerer, R.A. and Vigna, G. (2002) 'Intrusion detection: a brief history and overview', *Computer*, 35(4), pp. supl27–supl30. Available at: https://doi.org/10.1109/MC.2002.1012428.

Kodheli, O., Lagunas, E., Maturo, N., Sharma, S.K., Shankar, B., Montoya, J.F.M., Duncan, J.C.M., Spano, D., Chatzinotas, S., Kisseleff, S., Querol, J., Lei, L., Vu, T.X. and Goussetis, G. (2021) 'Satellite Communications in the New Space Era: A Survey and Future Challenges', *IEEE Communications Surveys & Tutorials*, 23(1), pp. 70–109. Available at: https://doi.org/10.1109/COMST.2020.3028247.

Li, K., Zhou, H., Tu, Z., Wang, W. and Zhang, H. (2020) 'Distributed Network Intrusion Detection System in Satellite-Terrestrial Integrated Networks Using Federated Learning', *IEEE Access*, 8, pp. 214852–214865. Available at: https://doi.org/10.1109/ACCESS.2020.3041641.

Manulis, M., Bridges, C.P., Harrison, R., Sekar, V. and Davis, A. (2021) 'Cyber security in New Space', *International Journal of Information Security*, 20(3), pp. 287–311. Available at: https://doi.org/10.1007/s10207-020-00503-w.

Maseno, E.M., Wang, Z. and Xing, H. (2022) 'A Systematic Review on Hybrid Intrusion Detection System', *Security and Communication Networks*. Edited by L. Maglaras, 2022, pp. 1–23. Available at: https://doi.org/10.1155/2022/9663052.

Najafimehr, M., Zarifzadeh, S. and Mostafavi, S. (2023) 'DDoS attacks and machine-learning-based detection methods: A survey and taxonomy', *Engineering Reports*, 5(12), p. e12697. Available at: https://doi.org/10.1002/eng2.12697.

Pavur, J. and Martinovic, I. (2022) 'Building a launchpad for satellite cyber-security research: lessons from 60 years of spaceflight', *Journal of Cybersecurity*, 8(1), p. tyac008. Available at: https://doi.org/10.1093/cybsec/tyac008.

Scarfone, K. and Hoffman, P. (2009) *Guidelines on Firewalls and Firewall Policy: Recommendations of the National Institute of Standards and Technology*. NIST Special Publication 800–41 Revision 1. Available at: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-41r1.pdf.

Scarfone, K.A. and Mell, P.M. (2007) *Guide to Intrusion Detection and Prevention Systems (IDPS)*. 0 edn. NIST SP 800-94. Gaithersburg, MD: National Institute of Standards and Technology, p. NIST SP 800-94. Available at: https://doi.org/10.6028/NIST.SP.800-94.

Shafi, M., Lashkari, A.H. and Roudsari, A.H. (2025) 'NTLFlowLyzer: Towards generating an intrusion detection dataset and intruders behavior profiling through network and transport layers traffic analysis and pattern extraction', *Computers & Security*, 148, p. 104160. Available at: https://doi.org/10.1016/j.cose.2024.104160.

Sharafaldin, I., Habibi Lashkari, A. and Ghorbani, A.A. (2019) 'A Detailed Analysis of the CICIDS2017 Data Set', in P. Mori, S. Furnell, and O. Camp (eds) *Information Systems Security and Privacy*. Cham: Springer International Publishing, pp. 172–188.

Uddin, R. and Kumar, S. (2023a) 'Federated Learning based Intrusion Detection System for Satellite Communication', *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*. *2023 IEEE Cognitive Communications for Aerospace Applications Workshop (CCAAW)*, Cleveland, OH, USA: IEEE, pp. 1–6. Available at: https://doi.org/10.1109/CCAAW57883.2023.10219228.

Uddin, R. and Kumar, S. (2023b) 'SDN-Based Federated Learning Approach for Satellite-IoT Framework to Enhance Data Security and Privacy in Space Communication', *IEEE Journal of Radio Frequency Identification*, 7, pp. 424–440. Available at: https://doi.org/10.1109/JRFID.2023.3279329.

Uhongora, U., Thinyane, M. and Law, Y.W. (2025) 'Development of an SDN-based Space System Simulation Framework for Intrusion Detection', *2025 IEEE International Conference on Cyber Security and Resilience (CSR)*. *2025 IEEE International Conference on Cyber Security and Resilience (CSR)*, Chania, Crete, Greece: IEEE, pp. 524–529. Available at: https://doi.org/10.1109/CSR64739.2025.11130072.

Vom Brocke, J., Hevner, A. and Maedche, A. (eds) (2020) *Design Science Research. Cases*. Cham: Springer International Publishing (Progress in IS). Available at: https://doi.org/10.1007/978-3-030-46781-4.

Yang, B., Arshad, M.H. and Zhao, Q. (2022) 'Packet-Level and Flow-Level Network Intrusion Detection Based on Reinforcement Learning and Adversarial Training', *Algorithms*, 15(12), p. 453. Available at: https://doi.org/10.3390/a15120453.