18th International Conference on Space Operations, Montreal, Canada, 26 - 30 May 2025.

SpaceOps-2025, ID # 433

AI-based Security Monitoring for Space

Jussi Roberts[a], Peter Franke[b], Simon Schäfer[b], Marc Niézette[b]*, Jeremy Meyer[c]

[a] *European Space Agency (ESA/ESOC), Darmstadt, Germany, jussi.roberts@esa.int*
[b] *Telespazio Germany GmbH, Darmstadt, Germany, {peter.franke, simon.schaefer, marc.niezette}@telespazio.de*
[c] *Starion Group, Wavre, Belgium, j.meyer@stariongroup.eu*
* Corresponding Author

Abstract

The security of space infrastructure has become an increasingly hot topic in the last years, emphasized by the deterioration of the world political context and the growing number of cyberattacks against critical infrastructures. In order to increase the resilience of their assets, ESA have initiated a number of activities aiming to further develop their cybersecurity capabilities beyond the typical coverage of the IT infrastructure to take into account attacks directed at specific components of the space mission, being the mission specific applications, protocols, or the space segment.

In this context, a Telespazio Germany led consortium has executed an ESA project titled Artificial Intelligence Security Monitoring Platform, the objective of which is to develop and validate a suitable AI driven security monitoring platform for space systems in order to improve on threat detection, incident response and system security monitoring capabilities for space missions. The tool relies on Machine Learning (ML) methods developed for anomaly detection. The focus of the approach is on space mission specific threats, i.e. attacks targeting the spacecraft and specific mission control segment applications and protocols. Central to the development of the platform is the identification and collection of the data sets required to support the ML approach, and the selection of algorithms required for their processing, altogether driven by the attack scenarios and the class of security events to be detected by the system.

The platform is implemented as an extension of ESA's AInabler platform, a component of ESA's infrastructure that facilitates the development and deployment of Artificial Intelligence solutions based on machine learning. AInabler supports the basic functionality for integration of ESA's control segment data sources, model training and deployment. The security monitoring platform complements this infrastructure with the integration of additional data sources, the organisation of specific algorithms and a dedicated user interface.

The validation is supported by data from an actual ESA mission, consolidated and augmented via simulation in a test environment including tools implementing attack scenarios.

This paper will summarize the design of the AI Security Monitoring Platform and provide results of the project, focusing on the data sets definition and collection, the algorithms selection, the types of events detected and the evaluation of the performance of the models in the validation environment. The lessons learned from the project and proposal for future work will be presented.

Keywords: anomaly detection, machine learning, cybersecurity, attack simulation

Acronyms/Abbreviations

Artificial Intelligence (AI), AI for Security Monitoring (AISecMon), Application Programming Interface (API), Consultative Committee for Space Data Systems (CCSDS), Convolutional Neural Network (CNN), Data Dissemination System (DDS), European Cooperation for Space Standardization (ECSS), Flight Dynamics System (FDS), File Transfer System (FTS), Large Language Model (LLM), Long Short-Term Memory (LSTM), Mission Control System (MCS), Machine Learning (ML), Machine Learning Operations (MLOPS), Mission Planning System (MPS), Packet Utilization Standard (PUS), Space Data Link Protocol (SDLP), Space Link Extension (SLE), Space Packet Protocol (SPP), Telecommand (TC), Telemetry (TM), Uniform Manifold Approximation and Projection (UMAP)

1. Introduction

Typically, civil space missions have been engineered prioritizing the protection of availability and integrity against non-adversarial threats, while neglecting security considerations and intentional attacks, leaving missions vulnerable. Often, a "security by obscurity" approach and the assumption that adversaries would not have the required infrastructure was relied upon. Due to the growing dependency on space services and the escalation of geo-political conflicts, the occurrence and severity of observed cyber-attacks has increased generally [1], and also against space missions. This includes several compromises of ground stations, blinding of satellites using lasers, signal jamming, and even anti-satellite weapon demonstrations [2], as well as the ViaSat attack of 2022 [3]. Consequently, the cybersecurity awareness in the space domain has increased significantly, realizing that the previous approach is no

longer sufficient. Through many initiatives, agencies, industry and standardization cooperations are working to improve security of both design and operation of space missions. Examples include the development of a secure systems engineering framework for space missions [4][5], automated penetration testing tools for space communication protocols and applications, space-specific threat models [6][7], secure space communication protocols [8][9], and establishment of security operations centres [10][11]. While many activities are aimed at securing new systems and missions by design, it is equally important to detect attacks against existing and new systems. In the space domain, this is especially critical as short detection times allow taking quick response and recovery actions, while spacecraft may be fully compromised and lost if the response is too slow.

One possibility for detecting attacks is to look for anomalies in the data produced by a space mission, specifically for anomalies that may be indicators of attack (referred to as "security anomalies" further on). In the past, anomaly detection approaches for space missions usually focused on common operational anomalies instead of security anomalies. The monitoring of the values of a certain subset of spacecraft parameters, which should be within fixed ranges, is a common practice in spacecraft operations [12]. However, the nature of security anomalies may differ from other anomalies, as adversaries who intentionally execute malicious actions behave fundamentally different to randomly occurring faults. Adversaries often attempt to cover their tracks, in which case it is unclear how the resulting security anomalies look like at all. The first signs might be visible before routinely monitored spacecraft parameters change, and once these changes can be observed, any mitigative action might come too late. Space missions generate diverse, voluminous data across different sources, hindering human analysts' ability to identify security anomalies amidst the ambiguity inherent in such events. Likewise, static rule-based systems struggle to adapt to the evolving nature of attacks. Artificial Intelligence (AI), and in particular Machine Learning (ML), lend themselves to empowering analysts to effectively monitor security even in face of these challenges. In this paper, we describe a security anomaly detection system developed in response to an ESA tendering action initiated under the General Support Technology Programme (GSTP), titled "Artificial Intelligence Security Monitoring Platform (AISecMon)" [13]. The system is designed to detect security anomalies in operational space mission data, mainly focusing on missions operated by the European Space Operations Centre (ESOC). We also share insights about two key challenges that we identified in building such a system, as well as our approaches to address them. The first key challenge is how to acquire or generate training and validation data that faithfully captures the effects of attacks proactively, i.e., without waiting for actual attacks to take place. Our approach leverages a combination of real mission data system applications, operational simulators and penetration testing tools to overcome this challenge. The second key challenge is the niche nature of the data generated during space missions and their operations, and the resulting lack of tried and tested ML strategies for such mission data. We address this by combining multiple strategies based on a careful analysis of the nature of mission data. Our main contributions are:

- An architecture for AI security anomaly detection for space missions that is able to seamlessly integrate various ESA mission data sources,
- a catalogue of space mission data sources that are generally relevant for security anomaly detection, which originates from a well-defined analysis process,
- an approach to creating labelled validation data containing security anomalies in the absence of real-world examples, including an instantiation of the approach for a real-world ESA reference mission, and
- solutions to the challenges that occur due to the space-specific nature of data in the different stages of a machine learning pipeline, and initial results.

There are several existing attempts in using AI for security monitoring in various parts of space systems. In [14], LSTM models are used to detect anomalies directly in S-Band and X-Band radio transmissions. In [15][16], security monitoring approaches for CubeSats are described, working on CAN Bus frames or CubeSat Space Protocol packets. These approaches focus on on-board detection, while our monitoring solution is placed in the ground system. Another approach for detecting security anomalies on-board is described in [17]. In [18], an ongoing work to perform AI security anomaly detection in a space mission is described. As in this work, a simulation environment is used to generate data. This environment represents the complete mission and is also the source of training data, without the use of real mission data. Several works focus on general anomaly detection on space mission data, usually working on time series of one or few spacecraft parameters [19][20][21], while the data used for security anomaly detection in this work are more complex and multi-faceted.

In [22], a general approach to build a security monitoring architecture for all parts of a space mission is described, ranging from ground IT systems to spacecraft. Our work may serve as one part of this architecture, concretely the ground-based security monitoring of space assets. While the use of AI may provide large benefits for security anomaly detection, there are also some additional security risks introduced by or against AI itself. For space missions, such security risks and countermeasures are described in [23][24].

The remainder of the paper is structured as follows: In Section 2, we describe the general architecture of AISecMon, our anomaly detection system. In Section 3, we detail the data sources and types relevant for building such a system. Section 4 focuses on the challenge of creating labelled validation data without access to real-world security anomalies. In Section 5 we explain the machine learning strategies used by our anomaly detection system. Section 6 describes initial performance results of the developed AI models, and we conclude and provide lessons learned in Section 7.

2. Architecture of AISecMon

This section describes the general design of our security anomaly detection system, AISecMon. The design is visualized in Figure 1. Our platform operates through two distinct components: A development component handling the incoming training data, extracting and storing features and training models; and a production component performing real-time inference on incoming data, interfacing with users and systems to provide anomaly detection results.

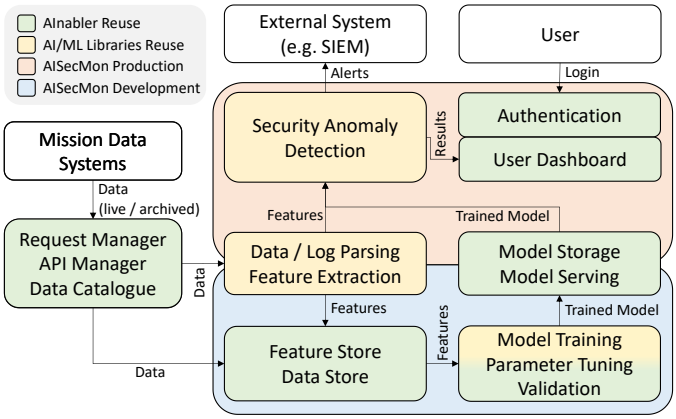

Figure 1: Architecture of AISecMon

Both components seamlessly connect to diverse mission data sources, enabling access to live or archived data for training and inferencing. The AISecMon platform is built upon the AInabler framework [25] developed by ESA, which provides a collection of tools to develop and deploy AI applications, including data, feature and model storage, model serving and user authentication. AInabler itself relies on Kubeflow, an open-source platform for building, deploying and managing of ML applications on Kubernetes clusters through Machine Learning Operations (MLOPS) pipelines. To interface with mission data systems, AInabler employs a request manager, API manager and data catalogue. Additionally, the framework already provides interfaces to common data systems used at ESOC. To implement the anomaly detection capability, AISecMon builds on components provided by established frameworks and libraries for AI/ML anomaly detection.

The flow of data through the platform, implemented in the form of MLOPS pipelines, is as follows: Initially, data are generated and stored by the various mission data systems and then made available directly or through data dissemination systems. From there, data are retrieved by AISecMon through the AInabler request manager, extended by custom interfaces for any additional data sources that are not supported by default. The retrieved data are ingested by a data preprocessing component, also utilizing tools for log parsing and feature extraction provided by the selected libraries. The data and extracted features are stored for potential reuse. At this point, the extracted features are forwarded to either of the development or production components. In the development component, the features are used to perform model training or retraining, including tuning of hyperparameters, and to validate the model performance, constantly improving the production model. The final models are stored in a model storage component provided by AInabler. In the production component, a trained AI model is retrieved from the model storage, which is then used to classify the ingested features into nominal or anomalous data. The classification results are provided to authenticated users on an AISecMon dashboard. The dashboard allows users to view the detected security anomalies, and also to provide investigation results (true or false positive) for later improvements of the model through retraining. Additionally, detected security anomalies are forwarded to registered external systems. An example of such a system could be a Security Incident and Event Monitoring (SIEM), as provided by the ESA Cyber Safety and Security Operational Centre (C-SOC) [10].

3. Data Source Catalogue and Collected Datasets

In this section, we present our catalogue of data sources for security anomaly detection across the core components of operational space missions. Figure 2 depicts the space mission components encompassed by our data catalogue, focusing on three key segments of the space mission relevant especially for command-and-control:

- Space segment: is assumed to consist of one satellite or a constellation of multiple satellites orbiting earth.
- Link segment: connects the space and the ground segments. This is done via RF (Radio Frequency) or optical communication. We assume that the communication is based on standardized protocols. Specifically for AISecMon, we consider a CCSDS-based protocol stack, utilizing the TM and TC Space Data Link Protocol (SDLP) [26][27]. The SDLP communication may be protected by the Space Data Link Security Protocol (SDLS) [8]. On top of these protocols, the Space Packet Protocol (SPP) [28] is used. For the case of ESA-operated missions, we assume an application layer based on the ECSS Packet Utilization Standard (PUS) [29]. On the lowest level, the link segment uses various encoding and error correction mechanisms [30][31].

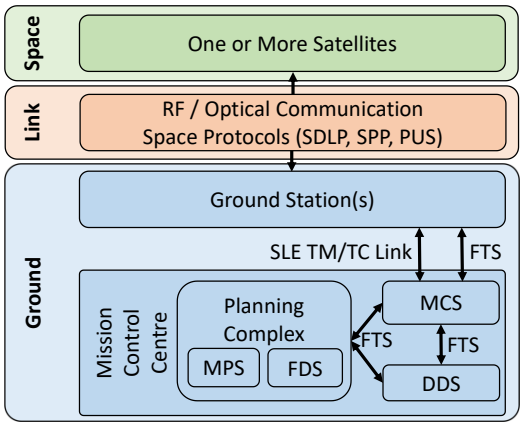

Figure 2: Relevant Space Mission Components

- Ground segment: consists of one or more ground stations, and a mission control centre. The ground stations directly interact with the spacecraft via the link segment. Additionally, they interact with the mission control centre via on-ground links that are part of the ground system. This happens via the CCSDS Space Link Extension (SLE) [32] protocol for TM and TC data, or via file transfer systems (FTS) over the internet. Within the mission control centre, the Mission Control System (MCS) is responsible for receiving TM from the ground stations, and for sending TCs to the ground stations for uplink. The MCS interacts with a planning complex consisting of a Mission Planning System (MPS) and a Flight Dynamics System (FDS). The planning complex is responsible for evaluating the state of the spacecraft and planning the operations to be done by the MCS, including scheduling TCs both on-ground an on-board. A Data Dissemination System (DDS) is responsible for distributing data collected or created by the parts of the ground system to external parties. The various components exchange data via a File Transfer System (FTS).
- While additional segments like the launch segment, or the user segment interacting with the spacecraft payload, are also important parts of a mission, we do not consider them as the focus of our solution.

It is important to highlight that our data catalogue encompasses not only critical space mission components but also relevant external data sources like space weather and terrestrial weather conditions. This comprehensive approach empowers us to tailor training data collection for individual missions. We successfully applied this concept to an operational ESA reference mission, selecting specific data sources from the catalogue and following a defined process to obtain targeted training data.

3.1 Data Source Catalogue

Based on the described mission architecture, we selected a catalogue of relevant data types and data sources for security anomaly detection by following a threat modelling process with the following steps:

1) Identify and describe the relevant systems (as done in the previous subsection)
2) Identify security goals
3) Identify potential threat actors
4) Identify threat events
5) Identify relevant threat scenarios (enactment of a threat event against a target system under specific conditions)
6) Evaluate the predicted influence of threat scenarios on data of the identified systems

7) Select relevant data sources to retrieve affected data

Resulting from the threat modelling, the following data are considered most relevant for security anomaly detection:

- Telemetry (TM) data: This includes any TM received from the space segment, both in raw form (CCSDS TM Frames or Space Packets) and processed (i.e. metadata or extracted spacecraft parameters). This may be retrieved from several locations: From the MCS or an adjacent data dissemination system, capturing the view of the mission control centre, or through binary TM files recorded at the ground stations.
- Uplinked Telecommand (TC) data: This includes any TCs sent by the MCS, in a high-level form (command names and parameters) or as raw data (CCSDS TC Frames, Space Packets). These data may again be collected from the MCS or adjacent DDS. Raw data may also be recorded from the uplink at ground stations.
- Mission Planning and Flight Dynamics information: This includes various file types exchanged with the MPS and FDS. Notable parts are the schedule increments to the on-board and ground schedules, which specify the commands that shall be executed in the upcoming time period. Also, information about e.g. predicted orbits, manoeuvres, orbital events, collision warnings is handled and distributed by these systems. The information may be retrieved directly from the systems, e.g. via the FTS, or from the DDS if it is made available there.
- Logs and event information: This mainly concerns events generated by human and machine interactions within the ground system. Attacker interactions leave traces here, e.g. through failed login attempts or malicious, unexpected file transfers. These data may be retrieved through the DDS or directly from the affected systems. Generally, logs from all ground systems may be collected.
- Additional data sources: This includes data sources that are not directly part of the mission infrastructure, but are nevertheless relevant for detecting security anomalies. Examples are data from RF Monitoring Solutions to identify signal interference, ground and space weather data and other space situational awareness data which might assist in distinguishing security anomalies from naturally occurring events.

3.2 AISecMon Training Data Collection and Processing

In order to create a training dataset of the best possible quality, it is necessary to collect data from all relevant data sources. The collected amount must be large enough to allow training sophisticated AI models. Additionally, it is paramount that all data are collected from the same time period, as this allows correlating the behaviour between the data sources.

For the initial prototype of AISecMon based on our ESA reference mission, a data collection activity has been conducted. During this activity, we collected the following data types:

- live TM data (binary SPP packets, SDLP frames and metadata) received by the MCS during ground station passes, retrieved using the DDS
- uplinked TC and activity data (binary SPP packets and metadata) from the MCS, retrieved using the DDS
- binary TM data (SDLP frames, including both non-science and science data) recorded on the satellite in-between ground station passes and dumped to the ground station during passes
- command schedules generated by the MPS and orbit information from the FDS,
- logs from the MCS and the FTS.

The additional, external data sources have been left out of scope. All of the data were collected from the same time period, with the exception of the MCS log data, for which this was not possible due to constraints during the collection.

After the collection of the data, a post-processing step was conducted in order to increase the usefulness of the data during further processing and to make it more independent from the system it was collected on. For this, an intermediate format based on JSON was defined for both TM and TC. The format retains only the information considered useful for anomaly detection, while discarding e.g. constant fields or duplicate information. Also, only information that may reproduced by the data generation environment described in Section 4.1 was retained. Additionally, the intermediate format contains some fields that may be useful for anomaly detection, but need to be parsed from the binary packet or frame data by the format conversion mechanism. These are e.g. details about PUS S1 Packets, which contain information about command acknowledgements by the spacecraft, and sequence counts contained in SDLP frames. A full extraction of all parameters contained in the retrieved binary packet data was not considered at this point. Collecting or extracting a specific subset of the parameters may however be considered in the future. All live TM and TC / activity data retrieved from the DDS, initially in an XML format, were converted to this intermediate format.

As the second part of the post-processing, the binary TM SDLP frames, contained in the dumps of recorded telemetry, were processed. From the individual frames, space packets and their metadata were extracted and also stored in the intermediate TM format. For this, an instance of the mission control system, deployed in the data generation environment described in Section 4.1, was used to replay the binary telemetry files, and to extract the processed data.

Post-processing this part of the data was important as it allows models to work on telemetry created between ground station passes (i.e. the majority of the time), without directly working on binary data.

It must be noted that the training data collected like this will not carry labels, which are required for the training of many AI/ML algorithms. The reason is that operational missions will usually not be able to provide labelled data including a variety of known attacks, but only large quantities of nominal data. The ML approach we chose to deal with this problem is addressed in Section 5.

Additionally, the evaluation of the trained system on any other but nominal data will not be possible when only the outcome of this data collection will be used, as the presence of security anomalies in the real mission data cannot be assumed. The approaches for solving this challenge by generating validation data are described in Section 4.

4. Evaluation approach using simulated attacks

As described in the previous section, it cannot be assumed that data coming from the real operational environment of a mission contains any occurrences of real-world security anomalies, and even if there were real instances, they would be extremely rare and possibly unknown. The result is a dataset without any labels that consists only of a single class. While it is possible to use these single-class data for training some machine learning models as described in Section 5, the performance evaluation of the trained models requires labelled data of both classes in any case - unless the evaluation shall only be done on nominal data, thus only being able to judge the number of false positives.

To solve this problem, we propose an approach to generating a labelled validation dataset containing both nominal data and security anomalies. The approach consists of using a data generation environment that is equipped with real mission data system applications, as well as an operational simulator of the spacecraft, and interfering with this environment using actual attack tools to create security anomalies. This results in labelled validation data that faithfully capture the nature of real operational data, while also containing security anomalies. In the following subsections, we describe the data generation environment, its usage to generate and label data with security anomalies and to extract data from the environment. Additionally, we describe how the data labelled in this way can be used to evaluate a security anomaly detection system. Using this approach, we generated labelled data containing various security anomalies to be used to evaluate the models of AISecMon. These anomalies include brute-force attacks against ground systems, manipulated commanding schedules, commands sent from rogue ground stations, and jamming attacks, among others.

4.1 Data Generation Environment

The environment consists of the following components as shown in Fig. 3: The operational simulator accurately simulates the satellite(s) of the mission. Operational simulators are used in space missions in order to test end-to-end mission operation software, train the flight control team before launch, test any manoeuvres, new commands, or on-board software before the use on the real spacecraft. The simulation captures all aspects of controlling the satellite. This includes an accurate orbit trajectory, all relevant systems on the spacecraft including the on-board computer running the on-board software, the RF link and space protocols used for the communication, and a simulation of the ground stations used by the mission including visibility checking, with an SLE interface exposed towards the MCS. Parts like specific payload functionalities, e.g. scientific instruments, may be simplified. In the normal state of the environment, the simulator is configured such that the satellite is in a routine operations state, with all systems behaving nominal and a TM/TC exchange being possible via the simulated ground stations.

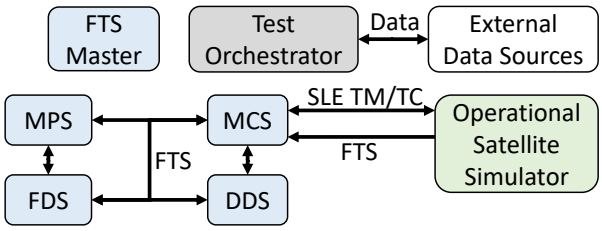

Figure 3: Data Generation Environment

The MCS in the environment is an instance of the actual MCS of the selected mission. It is able to monitor and command the satellite using the configured set of parameters, SLE links and TCs. In the normal state (no attacks have happened), the MCS exchanges TM and TC with the simulator over the SLE interface during planned ground station passes, and downloads recorded TM files from the simulated ground station via an FTS. The FDS and MPS provide

expected orbit and schedule information to the MCS, following the actual procedures and formats used by the mission. This allows the MCS to realistically command the satellite. Furthermore, the MCS provides data to the DDS in the test environment. The interactions between these components happen via file transfers using the FTS. All file transfers are controlled by an FTS master (arrows left out for clarity of the figure).

The Test Orchestrator machine remotely controls all other components in the data generation environment (arrows left out) by setting up their initial state, facilitating their interactions, and later on interfering with them and directing the attack scenario execution. It also interacts with any external data sources that may be considered. This could happen by connecting to actual instances of these data sources to fetch data, by simulating them using custom scripts replicating their real data formats, or entirely replacing them by pre-fetched or dummy data.

For our chosen reference mission and the current version of AISecMon, we configure the data generation environment in the following way: The external data sources are left out of scope. Additionally, the MPS and FDS are not actually deployed, but imitated. This is done using the real files generated by the operational systems, which were collected as part of the training data collection, and transferring them to the MCS as expected. For very repetitive file types, like the command schedules used during routine operation, scripts may also be used to generate files of the same type for different time periods to increase the amount of available data usable to command the satellite. The Test Orchestrator machine is based on Kali Linux, allowing it to utilize a wide range of penetration testing tools out of the box.

4.2 Attack Scenario Simulation

In this subsection, we describe the tools and their capabilities used to simulate the attack scenarios in our environment and the steps used to simulate each scenario. The simulation of attacks is based on three main pillars:

1) Off-the-shelf penetration testing, system administration and networking tools
2) Space-specific attack tools and custom scripts
3) The capabilities of the operational simulator

The first category includes any tools usable to intercept and modify network traffic and attack software systems e.g. through brute-force attacks on passwords and keys, injection attacks and fuzzing. The second category includes applications and scripts to deal with space-specific protocols (e.g. SLE) and data structures (e.g. schedule files) beyond the capability of generic tools. These may be written by other researchers, custom-made or included as part of the deployed systems e.g. for debugging / testing purposes. The third category might be the most relevant to achieve high-fidelity attack simulations. An operational simulator includes the possibility to interact with the simulated system in various ways, including manually changing states and parameters of the simulated subsystems, inserting commands and changing of orbit parameters. This may be used to induce various failures in the spacecraft and its communication with the ground stations, to simulate rogue commanding and even physical attacks on the spacecraft. Interacting with the simulator is possible in a scripted fashion, which allows its inclusion in the automatic execution of attack scenarios by the Test Orchestrator. This usage of the operational simulator in the data generation environment is similar to a use-case of real missions: It is common to conduct simulation campaigns for critical manoeuvres to train the flight control team. These campaigns include the simulation of anomalies that might occur during the operation of the spacecraft, though security anomalies are not the focus. The scripts used to simulate operational anomalies during simulation campaigns may also be modified and reused for the attack scenario simulation.

Each attack scenario is simulated with the following steps:

1) Start and configure all parts of the data generation environment to be in a nominal state. The initial state may vary depending on the attack scenario, e.g. based on the time in the mission during which the attack takes place. The Test Orchestrator sets up the initial state automatically based on an attack scenario configuration. Typically, this includes:
    a. Setting all machines and components to run (synchronized) at the configured time of the scenario
    b. Loading the simulator with an existing "breakpoint" for nominal operations
    c. Initializing the simulator with the correct time and orbit information and cleaning on-board schedules and stores of any old data. To obtain accurate orbit information for scenarios, the flight dynamics files obtained from the operational mission as part of the training data collection are used.
    d. Providing mission planning files for this time to the MCS
    e. Setting up timed events for ground station passes during the scenario and reminders for manual activities by the operator
2) Simulate the nominal operation for a certain time period.

3) Start the execution of the attack scenario on the Test Orchestrator, possibly combining multiple different tools to simulate an attempted, successful or unsuccessful attack. Each attack scenario is present as a script on the Test Orchestrator, which then executes the needed steps on all components of the environment. Not every scenario incorporates the actual execution of the attack as it would occur in the real world, but the tools are used in a way that has the same effect on the generated data as the outcome of the real attack. As the Orchestrator has full control over the data generation environment, it may omit some steps needed for real-world attacks in order to achieve the same outcome. For example, it may directly use passwords assumed to be stolen, or directly influence the spacecraft without really compromising it first.
4) Stop the execution of the attack scenario, and continue the nominal operation for a certain time period. Depending on the scenario to be simulated, this may also include the execution of a planned ground station pass. This may require some manual actions to be taken in the MCS by the operator of the environment, e.g. to ensure the commands planned for this pass are sent correctly.
5) Collect the generated data as evidence from the test environment and label it as described in Section 4.3. Instead of directly collecting the data, the procedure may be extended by chaining the execution of multiple attack scenarios in a single data generation run. This simulates the actions of an adversary whose attacks build on top of each other, gradually increasing the scale of an attack.

4.3 Data Extraction, Labelling and Conversion

Data are extracted from the data generation environment using procedures close to the data retrieval from the real operational environment. For this, the DDS is used to access the data where possible. Other data must be retrieved by the Test Orchestrator directly from the target systems by accessing files, databases or APIs. All data types collected from the operational mission (see Section 3.2) are collected from the environment in the same format, or one containing as much as possible of the equivalent information. After the initial collection, all telemetry and telecommand / activity data are transformed to same intermediate format described in Sect. 3.2, including replay of binary telemetry dumps.

When data are extracted from the environment while attack scenarios are simulated, the Test Orchestrator creates an additional label file containing the exact time period of each attack scenario execution. Additionally, each time period is annotated with the specific attack scenario being executed. This information is used later to determine when a security anomaly detection system should raise an alert, so that alerts can be classified as a false or true positive, as described in Section 4.4. Using the scenario labels, anomaly detection systems that attempt to classify the exact type of attack may also be evaluated. Having obtained training and validation datasets, ML models can be developed as described in Section 5.

4.4 Evaluation Metrics

As described, the generated validation data are not labelled per individual data item, but per time period. Therefore, it is not possible to directly compute common metrics for evaluating model performance directly based on predictions and labels for single or few data items. Most metrics are based on the counts of true positive, true negative, false positive and false negative detections, which are not straight-forward to determine in this case: Just because one data item occurs during a labelled time-frame, this does not necessarily mean that it must be affected by the attack and part of an anomaly. Essentially, counting the number of false negatives becomes impossible (except for simple scenarios where the exact effect of the attack on the data is known).

For this reason, we propose to use the generated data with an evaluation method that does not focus on evaluating individual detections of a single model, but on the performance of the entire anomaly detection system (likely an ensemble of multiple models) at detecting attacks / anomalies as a whole. Using a windowing approach, multiple positive detections of the models in the system are grouped into a single "incident". If an incident detection is within a labelled time window, it counts as a true positive detection, while incidents outside of a labelled window are false positives. If during an entire labelled time window, no incident is detected, this counts as a false negative. True negative detections are not counted. Using these basic counts, further metrics may be computed.

5. Machine Learning Approaches for Anomaly Detection

Machine learning excels at detecting patterns in data. In AISecMon, we aim to leverage machine learning capabilities to identify security-related incidents by analysing as many mission data as possible. Based on the data collected and generated as described previously, we conducted an experimentation processes encompassing the following key activities:

1) Data Selection, Analysis & Assessment
2) Model & Preprocessing Approach Assessment
3) Evaluation and Deployment

Each activity is described in the following subsections. Section 6 discusses the initial performance results obtained from the trained models.

5.1 Data Selection, Analysis & Assessment

In the initial data assessment step, we investigated the training data collected, and selected a specifically relevant subset for training and evaluating the prototype models. Based on their timing properties and the correlation between data sources, we identified potential subsets of the data on which models could be developed. In addition, we conducted a comparison of the collected operational data and the simulated validation data, in order to make sure that the simulated data replicate the operational data in a sufficient level of detail.

Data Source Selection: For our experiments, we selected the collected MCS Log Data, Telemetry (both live and recorded) and Telecommand / Activity data, and mission planning files for on-ground and on-board scheduling. The FTS logs were not included in the experiments at this point, as the investigation showed them to be unaffected by the considered attack scenarios, however they may be included in the future. For the TM and TC data, only the available metadata attributes are considered, while further processing of the binary packet data than what was described in Section 3.2 was ruled out.

Potential Model Structure: As mentioned in Section 3.2, the MCS logs could not be collected from the same time period. This effectively prevents a correlation of these data with the other data types in the models.

Due to this, we developed a dual-approach strategy for the conducted experiments: one approach focusing exclusively on MCS log data, and another integrating and correlating all other identified sources. The approaches should mostly differ only in the data, while the applied methods remain the same on a high level.

For future experiments, an extended model structure can be implemented based on the timing properties, i.e. the order in which data arrive and become available to the anomaly detection system. This could be beneficial as models operating on all data types at the same time will only be able to perform anomaly detection for a time period once the data from all sources is present, which may lead to a long delay in the detection. For the selected data types, the expected order of arrival is as follows:

1) Mission planning files with commands to be executed are created a significant time before they are used
2) The MCS (and other ground systems) log operations while they are happening, also in-between ground station passes
3) During ground station passes, live telemetry is coming in at the MCS, telecommands scheduled on-ground are released and further telecommands are scheduled on-board the spacecraft
4) After ground station passes, the telemetry dumps containing the TM recorded on the spacecraft for the time since the previous ground station pass become available

Therefore, an additional model working solely on the mission planning files may prove useful, as it could detect unusual planned commands and thus potentially manipulated planning files long before they take effect. Additionally, a model working only on the planning files and the MCS / ground system logs may be useful to avoid long detection delays between ground station passes - in our case, this role is taken by the model operating on the MCS logs only. Finally, the full model could correlate all data types live during ground station passes, and could process the data from the period between the last ground station passes once the recorded telemetry has arrived.

Simulated vs. Operational Data Analysis: We employed multiple techniques to identify potential differences between simulated and operational data. Our analysis included:

- Manual inspection of log entries
- Analysis of embedded log messages
- Dimensional reduction visualization
- Anomaly detection algorithms executed on simulated nominal data

The manual inspection of logs revealed no significant differences, except for slight differences between multiple software versions, which can be mitigated by including data from multiple versions in the training set, and performing appropriate pre-processing of data, as described in Section 5.2. Additionally, the manual inspection revealed humanly recognizable impacts of some attack scenarios on the log and telemetry data, with the impact not being fully clear on more complex scenarios.

For creating the embedded log messages, we utilized the all-MiniLM-L6-v2 [33] model, which provides a good balance between performance and model size. Using UMAP (Uniform Manifold Approximation and Projection), we

visualized these embeddings in two-dimensional space (see Figure 4), which revealed no significant distinctions between simulated and operational data clusters.

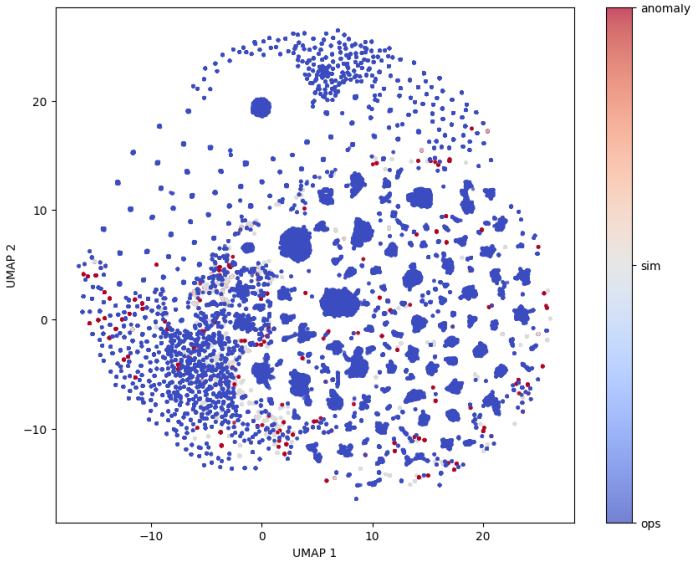

Figure 4: UMAP Projection of simulated and operational data. Blue: operational data, grey: simulated normal data, red: simulated data during anomaly time frame

Additionally, we applied the Isolation Forest algorithm for anomaly detection on the embeddings, but the results did not clearly identify the simulated data as outliers. From these findings, we concluded that the simulated data likely are sufficiently similar to the operational ones to avoid the classification of all simulated data as anomalies.

These analytical procedures were at this point conducted on MCS logs and will need to be extended to the other selected data sources to ensure a comprehensive assessment.

5.2 Model & Preprocessing Approach Assessment

Approach Selection Based on Data Constraints: As mentioned in Section 3.2, we faced a significant constraint: there are no known anomalies in the provided operational data. On the other hand, the simulated attack scenarios are very limited in scope and quantity. Even with extended simulation runs, it would require extensive effort to develop a dataset sufficient for supervised learning approaches. Furthermore, such a dataset would remain inherently superficial, with all anomalies being simulated rather than representing real-world security incidents.

The transferability of supervised models to operational environments would, at minimum, require validation against real anomalies. While transferability is also a concern with unsupervised approaches, the impact is considerably less significant since unsupervised methods focus on learning the characteristics of normal data rather than attempting to model the nature of anomalies. Based on these considerations, we have decided to limit our methodology to unsupervised learning approaches.

Selected Models: The chosen attack scenarios are detectable through various patterns across different sets of time-series from log files, telemetry, commanding and planning data. This observation, combined with the significant textual content in many of the log files, suggests that a Large Language Model (LLM) based approach could be effective. We identified LogLLM [34] as a promising open-source repository enabling unsupervised anomaly detection in log files.

Concurrently, we aim to demonstrate that simpler models can effectively address a subset of the attack scenarios. This is particularly relevant for operational environments, where an ensemble of interpretable, lightweight models that don't require specialized hardware (GPUs) may offer practical advantages in terms of maintainability, explainability, and resource efficiency. For this, we evaluated a clustering-based approach, as well as a single log embedding approach based on the FAISS vector store [35].

Between simple statistical models and advanced LLM approaches, we are also exploring intermediate deep learning-based methods. Deep-Loglizer [36] is an open-source repository for a variety of deep learning-based methods including CNNs, LSTMs, Autoencoders and also one transformer architecture.

Preprocessing Techniques: The choice of preprocessing depends on the selected model. Our preprocessing pipeline includes several steps and techniques, aiming to allow the models to make useful correlations between data sources and to remove noise:

- *Generation of a "master log" file*: This step is conducted for every model, especially those considering multiple data sources. We extract meaningful information from multiple data types and merge them together in one timeline using the defined timestamps per data item. Since the MCS log is standalone in the training data, it is only integrated programmatically but remains separated for the course of these experiments. Based on the timing information for each data item, it is inserted at one or multiple points of the timeline. The different data types are processed as follows:
    - TM Packets (SPP, with Metadata): Inserted 1. at the generation time of the packet, and 2. at the earth received time of the packet
    - TM Frames (SDLP): Inserted at the earth received time of the frame only (frames are sent immediately after being generated)
    - MCS Log Lines: Inserted at their generation time
    - TCs / activities sent by MCS: Inserted 1. at their release time, and 2. at their execution time
    - On-ground scheduled commands / activities: Inserted at their scheduled release time
    - On-board scheduled commands / activities: Inserted at their scheduled execution time

  Based on this merging of the data types, models working on time series of data will be able to build important correlations, e.g. between the scheduling of commands, their release, their execution and the resulting telemetry, which may be disturbed by adversaries' activities.
- *String Cleaning:* For log data types, we implement comprehensive string cleaning procedures to remove non-informative elements such as dates, timestamps, and UUIDs. This normalization process significantly reduces noise and variability in the data, allowing our models to focus on the substantive content that might indicate anomalous behaviour rather than routine variations in logging formats.
- *Sequence Generation from Clustering*: We develop meaningful sequences by clustering related data entries, allowing us to capture temporal patterns and relationships between events that might indicate security incidents. This approach transforms individual data entries into contextually rich sequences that better represent system behaviours.

These preprocessing steps are crucial for improving model performance across all methodological approaches, from simple statistical models to sophisticated deep learning architectures.

5.3 Evaluation and Deployment

Based on the methodology assessment, the selected approaches were implemented and evaluated as described in Section 4. Well-performing approaches are implemented in the form of model training and inferencing pipelines in the AInabler platform, as described in Section 2.

6. Initial Results

This section provides the initial model results from the experiments on MCS log data using the clustering-based and single log embedding approaches.

6.1 Clustering-Based Approach

We initiated our investigation with a simple model to quickly establish baseline results that could serve as a comparison point for more sophisticated approaches. Our initial strategy involved a k-means clustering of log messages into groups of similar content using their vector embeddings. The resulting clusters demonstrated reasonable quality and coherence. The granularity of these clusters can be adjusted by modifying the number of clusters, providing flexibility in the analysis approach.

However, when analysing the sequences derived from these clusters, the results were not promising. Despite testing multiple approaches, we found that while certain specific sequences could be linked to anomalies, this method produced an excessive number of false positives, rendering it impractical for operational use. Figure 5 illustrates a binary signal visualization of the clustering approach predictions on attack-free data, with each point on the x-axis representing one data item. The blue signal represents the prediction, with each spike on the y-axis indicating a predicted anomaly. The red signal represents labelled anomaly time frames, which are not present in this case. Even

considering that an improvement of the training data quality might still be possible, it can be seen that the model predicts a significant number of anomalies although the baseline data does not contain any anomalies.

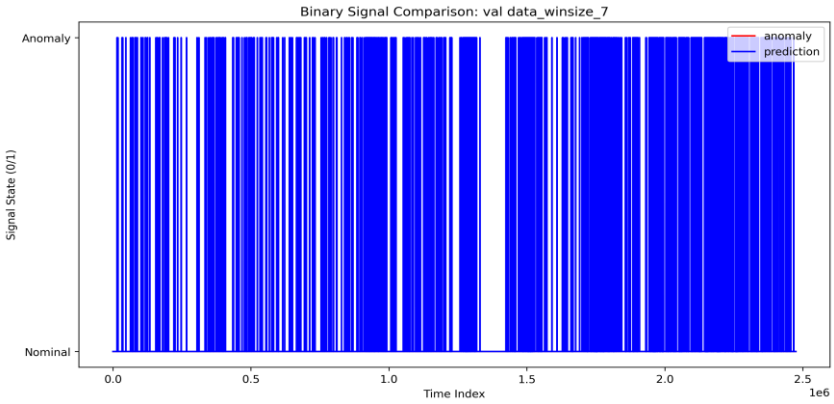

Figure 5: Clustering approach on attack-free data

Figure 6 displays the UMAP dimensionality reduction of the sequences of cluster IDs, at the same window size but with an increased number of clusters—which resulted in an increase of false positives. This visualization is an indication that the anomalies are not clearly separable in the embedding space. Additionally, it reinforces our earlier observation that the simulated data do not exhibit significant differences from the operational data.

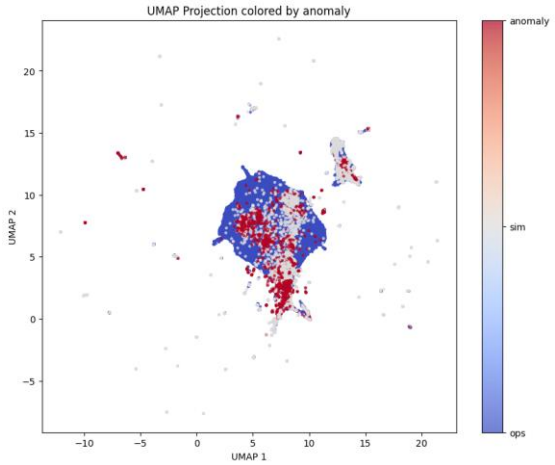

Figure 6: UMAP projection of clustering embeddings

Figure 7 demonstrates the performance of this clustering-based approach on some of the validation data containing attacks, with a brute-force attack against the MCS user login shown on the left, and the execution of manipulated commands from a planning file on the right. Visual inspection alone confirms that this method does not yield actionable results for anomaly detection, as the majority of detections happens outside of the labelled anomaly windows, while nearly no anomaly predictions happen during the anomaly windows.

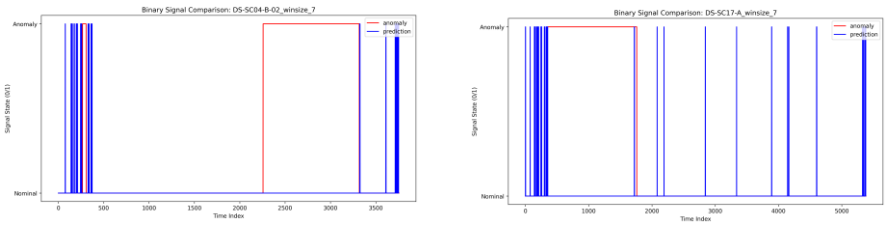

Figure 7: Clustering-based approach predictions on attack scenarios

6.2 Single Log Embedding Approach

The evaluation of an approach based on individual log embeddings produced more promising results. We implemented FAISS [35], a vector store that enables efficient nearest neighbour search. While FAISS employs a heuristic search method, this characteristic does not negatively impact our use case.

In this approach, we use the distance to the nearest neighbour as the metric for anomaly detection. Figure 8 demonstrates that this method, given that a suitable threshold is chosen, is able to successfully detect some of the simulated attack scenarios (Left side: Brute-Force attacks against the MCS user login, top right: Jamming attack against telemetry reception, bottom right: Execution of manipulated commands from planning file. Note: "Anomaly" on the y-Axis corresponds to an anomaly score of 1.0, while "Nominal" corresponds to 0. The green line additionally marks periods in which a ground station pass happens). Figure 9 further confirms that the approach performs well on nominal data, with minimal false positives.

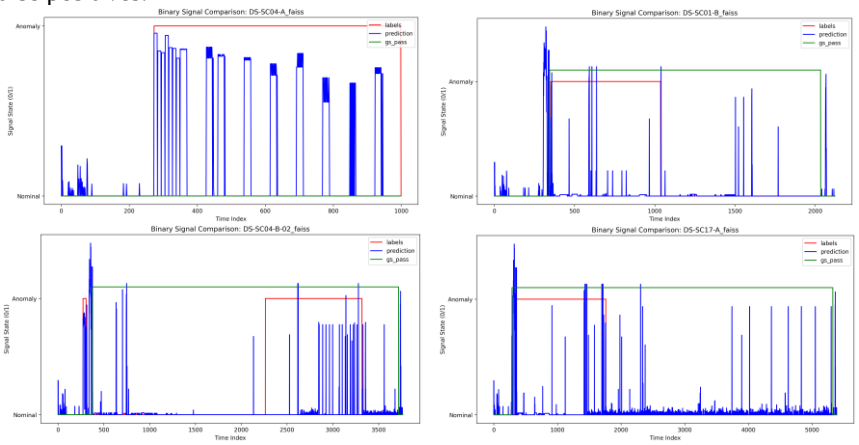

Figure 8: FAISS anomaly scores on attack scenarios

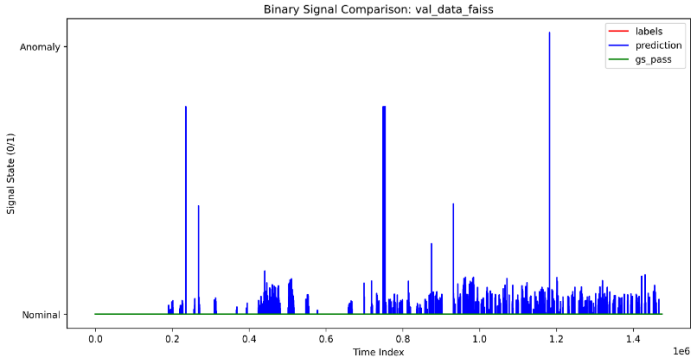

Figure 9: FAISS anomaly scores on nominal data

Operational Considerations: For a practical implementation of the single log embedding approach using FAISS in an operational environment, several steps would be necessary:

1) *Threshold Evaluation:* A specific threshold needs to be determined to distinguish between normal and anomalous behaviour.
2) *False Positive Investigation:* Samples that appear abnormal in nominal data require thorough investigation. For example, in scenarios containing ground station passes, significant spikes in the FAISS anomaly scores can be seen during the beginning of each pass, leading to false positives. These could be improved by improving the training data quality.
3) *Reference Data Refinement:* If a log message causing a spike is determined to be normal, it can be added to the reference data to reduce false positives. During the tuning phase, several iterations of this step may be needed. Note that this is also still possible while the model is already operational, allowing an effective continuous learning.

4) *Final Threshold Determination:* After the manual refinement steps, a final threshold must be established for the anomaly detection.

7. Conclusions

The presented AI security monitoring system contributes to the ongoing efforts of strengthening the cyber resilience of ESA space missions. It does this by identifying the specifics of data generated during space missions and affected by security anomalies, selecting suitable machine learning models and training techniques for the type of data generated, training the machine learning model, and establishing a way of introducing security-related anomalies into the data for validation purposes. Our research has yielded promising initial results, while highlighting the importance of representative data, and the generation of such data if it is not available from operational environments, for which a high effort and domain-specific knowledge is required.

At the time of writing, the development of the AISecMon platform is being finalized, with its validation and model evaluation ongoing. The next steps will include the evaluation of the log data using Deep-Loglizer and LogLLM as more sophisticated deep learning and language model approaches, the application of all developed methods to all data types, and a more extensive evaluation of the false positives generated by our approaches across all data sources.

Future work includes the direct integration with mission data sources to perform live inferencing, and with the ESA C-SOC in order to feed operational mission data and anomaly detection capabilities to the ESA-wide cyber monitoring solution.

Acknowledgements

The authors of this paper would like to thank all project consortium members and colleagues, and the ESOC flight control and data systems teams for their invaluable support.

References

[1] Fortinet, 2024 State of Operational Technology and Cybersecurity Report, https://www.fortinet.com/content/dam/fortinet/assets/reports/report-state-ot-cybersecurity.pdf, (accessed 07.04.2025)

[2] J. Pavur, I. Martinovic, Building a Launchpad for Satellite Cybersecurity Research: Lessons from 60 Years of Spaceflight, Journal of Cybersecurity, Volume. 8, Issue 1 (2022)

[3] N. Boschetti, N.G. Gordon, G. Falco, Space Cybersecurity Lessons Learned from the ViaSat Cyberattack, ASCEND 2022, Las Vegas, NV, USA, 2022, 24-26 October

[4] T. Leclerc, S. Paul, J. Roberts, F. Dagnat, F. Ledoux, J.-C. Bach, M. Wallum, N. Mezzina, D. Fischer, S. Guerin, I. Benamer, P. Jeanjean, A Flexible and Robust Framework for the Secure Systems Engineering of Space Missions, SpaceOps-2023#530, 17th International Conference on Space Operations, Dubai, United Arab Emirates, 2023, 6 – 10 March

[5] M. Wallum, D. Fischer, D. Ingami, P. Hagstrom, G. Mihalachi, M. Merialdo, Secure Systems Engineering Framework for Space Missions, IAC-22-D5-4, 73rd International Astronautical Congress, Paris, France, 2022, 18 – 22 September

[6] The Aerospace Corporation, Space Attack Research & Tactic Analysis (SPARTA), https://sparta.aerospace.org, (accessed 07.04.2025)

[7] European Space Agency, Space Attacks and Countermeasures Engineering Shield (SPACE-SHIELD), https://spaceshield.esa.int, (accessed 07.04.2025)

[8] CCSDS, Space Data Link Security Protocol, Blue Book, no. 355.0-B-2, 2022

[9] IETF, E. Birrane, K. McKeever, RFC 9172: Bundle Protocol Security (BPSec), 2022

[10] M. Niezette, C. Laroque, J. Irving, Space mission security monitoring at the ESA Cyber Safety and Security Operational Centre (C-SOC), SpaceOps-2023#621, 17th International Conference on Space Operations, Dubai, United Arab Emirates, 2023, 6 – 10 March.

[11] NASA Office of Audits, Audit of NASA's Security Operations Center, Report IG-18-020, USA, 2018, May 23

[12] G. Labrèche, D. Evans, D. Marszk, T. Mladenov, V. Shiradhonkar, T. Soto, and V. Zelenevskiy, "OPS-Sat Spacecraft Autonomy with Tensorflow Lite, Unsupervised Learning, and Online Machine Learning," in 2022 IEEE Aerospace Conference (AERO), 2022, pp. 1–17.

[13] European Space Agency, GT1Y-309GD Artificial Intelligence Security Monitoring Platform, https://esastar-publication-ext.sso.esa.int/ESATenderActions/details/44031, (accessed 07.04.2025)

[14] R. Bua, L. Necchi, P. Corrado, M. Capella, B. Villat, and G. Pandolfi, Machine Learning Radio-Frequency-Based Anomaly Detection for Ground Station and Satellite Telecommunication, 37th Annual Small Satellite Conference (SmallSat), Utah, 2023

[15] O. Driouch, S.Bah, Z. Guennoun, Distributed Intrusion Detection System for CubeSats, based on Deep Learning Packets Classification Model, 2024 Security for Space Systems (3S), Noordwijk, Netherlands, 2024, 27-28 May
[16] O. Driouch, S. Bah, Z. Guennoun, CANSat-IDS: An adaptive distributed Intrusion Detection System for Satellites, Based on Combined Classification of CAN Traffic, Computers & Security, Volume 146 (November 2024)
[17] J. Rimani, L. Buizza, A. Baker, Securing Satellite Operations: A Novel Approach for Space Assets On-Board Safety, IAC-24-D1.2.2, 75th International Astronautical Congress, Milan, Italy, 2024, 14 - 18 October, https://doi.org/10.52202/078372-0013
[18] A. Barraqué, J. Airaud, The Use of AI in the Detection of Cyber Intrusions in Orbital Systems, IAC-24-D5.4.10, 75th International Astronautical Congress, Milan, Italy, 2024, 14 - 18 October, https://doi.org/10.52202/078376-0034
[19] B. Werner, L. Palladino, H. Yang, L. Chau, N. Mohler, T. Schild, A. Langs, Automated Anomaly Detection Integrated in a Modern Mission Control System, IAC-24-B6.IP.37, 75th International Astronautical Congress, Milan, Italy, 2024, 14 - 18 October, https://doi.org/10.52202/078367-0070
[20] J. Nalepa, M. Myller, J. Andrzejewski, P. Benecki, S. Piechaczek, D. Kostrzewa, Evaluating Algorithms for Anomaly Detection in Satellite Telemetry Data, Acta Astronautica 198 (2022), 689-701
[21] P. Gomez, R. Vavrek, G. Buenadicha, J. Hoar, S. Kruk, J. Reerink, Machine Learning-Driven Anomaly Detection and Forecasting for Euclid Space Telescope Operations, IAC-24-B6.IP.56, 75th International Astronautical Congress, Milan, Italy, 2024, 14 - 18 October, https://doi.org/10.52202/078367-0083
[22] P. Franke, A. Weber, J. Roberts, L. Baumgärtner, M. Niezette, Space Mission Security Monitoring, IAC-24-E9.2.10, 75th International Astronautical Congress, Milan, Italy, 2024, 14 - 18 October, https://doi.org/10.52202/078386-0009
[23] A. Weber, P. Franke, Space-Domain AI Applications need Rigorous Security Risk Analysis, Workshop on Security of Space and Satellite Systems (SpaceSec), San Diego, CA, USA, 2024, 1 March
[24] Warsaw University of Technology, KP Labs, European Space Operations Centre, PINEBERRY - Secure and Explainable AI for Space
[25] European Space Agency, AInabler, https://ainabler.space-codev.org/, (accessed 07.04.2025)
[26] CCSDS, TM Space Data Link Protool, Blue Book, no. 132.0-B-3, 2021
[27] CCSDS, TC Space Data Link Protocol, Blue Book, no. 232.0-B-4, 2021
[28] CCSDS, Space Packet Protocol, Blue Book, no. 133.0-B-2, 2020
[29] ECSS, Telemetry and Telecommand Packet Utilization, no. E-ST-70-41C, 2016
[30] CCSDS, TM Synchronization and Channel Coding, Blue Book, no. 131.0-B-5, 2023
[31] CCSDS, TC Synchronization and Channel Coding, Blue Book, no. 231.0-B-4, 2021
[32] CCSDS, Space Link Extension - Internet Protocol for Transfer Services, Blue Book, no. 913.1-B-2, 2015
[33] Sentence Transformers, all-MiniLM-L6-v2, https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2, (accessed 07.04.2025)
[34] W. Guan, J. Cao, S. Qian, J. Gao, C. Ouyang, LogLLM: Log-based Anomaly Detection Using Large Language Models, arXiv ePrint, https://arxiv.org/abs/2411.08561, 2025
[35] Facebook Research, FAISS (Facebook AI Similarity Search), https://github.com/facebookresearch/faiss, (accessed 07.04.2025)
[36] Z. Chen, J. Liu, W. Gu, Y. Su, M. Lyu, Experience Report: Deep Learning-based System Log Analysis for Anomaly Detection, arXiv ePrint, https://arxiv.org/abs/2107.05908, 2022