A Satellite Constellation Simulator for Space Systems Cybersecurity Research and Development

Simone Urbano
Starion Group France
Nice, France
s.urbano@stariongroup.eu

Jacques Girard
Thales Alenia Space
Toulouse, France
jacques.girard@thalesaleniaspace.com

Louis Lolive
IRT Saint-Exupéry
Toulouse, France
louis.lolive@irt-saintexupery.com

Vincent Nicomette
LAAS-CNRS
Toulouse, France
vincent.nicomette@laas.fr

Pierre Bacquet
Thales Alenia Space
Toulouse, France
pierre.bacquet@thalesaleniaspace.com

Patrick Hebrard
Starion Group France
Nice, France
p.hebrard@stariongroup.eu

Guillaume Auriol
LAAS-CNRS
Toulouse, France
gauriol@laas.fr

Ludovic Cintrat
Gatewatcher
Puteaux, France
ludovic.cintrat@gatewatcher.com

Benjamin Deporte
IRT Saint-Exupéry
Toulouse, France
benjamin.deporte@irt-saintexupery.com

Julien Airaud
CNES
Toulouse, France
julien.airaud@cnes.fr

Matteo Merialdo
Nexova Cyber
Libin, Belgium
m.merialdo@nexovagroup.eu

***Abstract*—Space systems have become critical infrastructures for modern society. Thus, cybersecurity and resilience of space systems has become a hot topic of research. In this study a technological component is presented that is designed to guide and to enhance cybersecurity investigations for space systems. In particular, we introduce a new simulation platform that has been developed in the framework of CSS ("Cybersecurity for Space Systems") project led by IRT Saint-Exupéry in Toulouse. This platform is introduced as a benchmark for cybersecurity research and development on space systems. The platform, based on NASA NOS3 simulator [28], can represent a single spacecraft or a constellation of spacecraft, including flight software, Mission Control System (MCS) and CCSDS stack (up to transfer frame layer) for space link communications [10] [11] [12]. A few selected cyber attacks and an Intrusion Detection System (IDS) component under development are introduced to showcase the potential of the chosen simulation platform in terms of attacks and defense development and evaluation.**

***Index Terms*—Space Systems, Cybersecurity, IDS, Cyber Range, Simulation Platform.**

I. Introduction

The new space ecosystem agility is defining a new route to follow for the space industry. Furthermore, our society is more and more dependent on space systems infrastructures, leading to new concerns about the cybersecurity of such systems [26]. In recent years, some attacks as ViaSat [5] in the context of Ukrainian war, have enlightened the need for more attention across all the lifecycle of such systems with respect to cyber threats. Indeed, the risk management practices and standards of modern space system need to evolve in order to keep pace with the evolution of the cyber threat landscape and the increase of the attack surface. The LEO SatCom systems are a clear example of such a situation as recently emphasized by a comprehensive report by ENISA [19]. The need to balance security with cost-effectiveness is also a top priority of new space and several studies have been conducted recently on this topic [36]. Finally, the rapid evolution of the frameworks used to describe the cybersecurity posture of space systems as SPARTA [42] and SPACE-SHIELD [14] proves that the need to share information and lesson learned, but also the possibility to improve the training of operators is becoming a priority for the space sector. In this context, it is clear the urgent need for a realistic but cost-effective simulator/benchmark of a space system that can be used to research and develop attacks and detection strategy, to train on practical scenarios and to evaluate risks and mitigations. The contribution of this study lies in the new components, attacks and countermeasures developed for this purpose. The development of the platform is still in progress, thus in this paper we will focus more on methods instead of performance and validation. In Section II a short review of existing simulation solutions is introduced. In Section III the simulation platform introduced by CSS project is presented and the main contributions of this study are detailed. In Section IV the focus is on security development and testing with regard to attack and defense capabilities. In particular, in Section IV-B some relevant attack scenarios are presented, while in Section IV-C, some specific intrusion detection and prevention strategies are introduced. Finally, in Section V the conclusions and perspectives of this study are proposed.

II. Related Work

There exists multiple proprietary and open source solutions for the simulation of a space system (and the associated flight/ground software). One can mention some well known frameworks as ESA GSTVi [22], ESA SIMULUS [16], ESA

MO Nanosat Framework [15], OS3 [25], Ansys STK [3], NASA NOS3 [28] and CNES Basiles [9] that can be adapted for the scope of Cybersecurity testing. For example, in [46] STK is associated to other tools to evaluate the security of a LEO Constellation. In [44] the GSTVi simulator is used to test a secure communications environment for space missions. In [35] NASA NOS3 [28] is chosen to generate representative threat data. It goes without saying that limited information are available on the simulation platform used by big companies of the space and defence sector worldwide. In the academic sector, custom or ad-hoc simulators are often preferred (for example in [47]). Among all the aforementioned platform, NASA NOS3 [28] is probably the most famous open source simulator that comes in a single easy to deploy virtual machine including Flight Software (FSW), Ground Software (GSW) and visualization. Moreover, NASA NOS3 components (in particular the OpenSatKit simulator [33] [27], that is very similar to NOS3) have already been used for cybersecurity testing in well known competitions as HackASat [23] due to its modularity, relative simplicity and code accessibility. On the other hand, a satellite constellation simulator for space systems cybersecurity research and development is a rare set of software and it is generally limited to big players of space and defence sector or institutions like NASA [4]. The framework that is being developed for CSS project, is even more rare as it will be composed by multiple complex software components:
1) A realistic space system simulator (including FSW, GSW and Visualization). This component is based on NASA NOS3.
2) A Cyber range platform (to manage high risks scenarios including dangerous software as malwares). This component is based on CITEF(Cyber Integration, Test and Evaluation Framework) platform by Nexova [32].
3) Penetration testing software (for example Kali Linux OS) equipped with customized exploits.
4) Detection and defence software as ground probes. This is based on Gatewatcher probes [20] and custom algorithms that are developed for CSS project by LAAS-CNRS as dedicated on-board Intrusion Detection and Prevention System (IDS/IPS).
The main idea is to be able to simulate a realistic attack scenario and the different step of the kill chain up to the final exploitation, leading to more robust defence strategies that can be tested and adapted to face specific threats. Finally, the goal is to increase the resilience of the considered simulated space system architecture.

III. Space System Simulation for CSS project

CSS project is led by the French Institute for Technological Research (IRT) Saint Exupéry in Toulouse. CSS will include contributions from leading institutional, academic and commercial partners in the space, cybersecurity and technology sectors: Starion Group, Thales Alenia Space, CNES, LAAS-CNRS, Gatewatcher, French Air Force Academy and Nexova Group. The objective is to mature the technological blocks necessary to improve the state-of-the-art in cybersecurity for space systems. One of the goals is to build a state-of-the-art space system simulation platform for cybersecurty research and development, where advanced AI techniques for attack and defense can be developed and evaluated. In order to improve the resilience of space systems, multiple technological modules are investigated: space system simulation, CTI for space, automatic attack generation, onboard agents for threat detection, on ground probes for automatic threat detection, on ground AI agents for automatic threat detection and isolation. This article describes the space system simulation module, that is currently based on NASA NOS3 simulator. The choice of NOS3 as baseline for the development of a space system simulator representing a constellation of satellites (and in particular CubeSats) is mainly related to the following factors: the platform is based on open source software; the full CCSDS stack implementation is available (only physical RF layer is missing); the platform has already proven its relevance for cybersecurity research and development [4] [23].

A. NASA NOS3

NASA NOS3 simulator (v1.6.2) is mainly based on 4 open-source components: NASA cFS [29], NASA CryptoLib [30], NASA 42 [41] and OpenC3 COSMOS [13]. The NASA cFS is an established FSW by NASA that has already flown on multiple NASA missions [31]. The cFS architecture is very generic and modular enabling reusability between missions and reducing the development costs. CryptoLib is a software-only solution implementing the Space Data Link Security protocol or SDLS [18]. NASA 42 is a well established NASA flight dynamics and visualization tool by Eric Stoneking [40]. COSMOS is a suite of application that can be used to control a set of embedded systems, for NOS3 it is used to represent the Mission Control System (MCS).

B. Contributions

The main contributions of this study with regard to NASA NOS3 (v1.6.2) are: (i) the introduction of new components enabling the deployment of satellite constellations such as ISL/RM (Inter Satellite Link / Routing Machine) component on board and Front End component on ground; (ii) NOS3 code adaptations for multi spacecraft configuration and reintegration of the TC/TM transfer frame layer of the CCSDS stack (NOS3 v1.6.2 is based on SPP layer); (iii) the introduction of a multipurpose Input generator and a predefined library of exploits (the user can send nominal and malicious TCs, craft a specific mission, perform fuzzing or simply select attacks for known vulnerabilities); (iv) the integration in the FSW of an IDS component for the development of threats detection algorithms on-board; (v) Hardware-in-the-loop (HIL) simulation of on-ground probes (proprietary probes provided by Gatewatcher [20]); (vi) the introduction of additional features for increased realism such as additional ADCS modes and realistic payload data (camera) from a 3D environment simulation (“Imager” component). The result of these contributions is that the user can simulate a realistic mission with realistic CCSDS traffic

for a satellite/constellation with optical payload[1]. The user can also prototype/test attacks and countermeasures and in the CITEF environment one can also defines trainings and red/blue team activities.

C. Implementation Hypotheses

NASA NOS3 v1.6.2 is provided to be deployed on a single Virtual Machine (VM) using Vagrant (the VM is based on Ubuntu 20.04.6 LTS). In the context of CSS project, it has been decided to split the simulator on multiple VM, in order to increase the modularity and to have the possibility to have different operators/users on different VMs (this configuration is preferred for red/blue team activities). In particular, it has been chosen to use 1 VM for each satellite (based on NASA cFS), 1 VM for each Mission Control System (MCS, ground segment based on COSMOS), 1 VM for the flight dynamics and visualization part (NASA 42). The physical layer (RF link) is not simulated in NOS3. CCSDS TF are passed over UDP to represent the RF link. For this reason, one can use a standard "tcpdump" command to get a capture (.pcap) of the simulator traffic. NOS3 v1.6.2 implements the CCSDS stack up to SPP layer. The Transfer Frame layer (TC/TM TF) is not fully available by default and several adaptations of NOS3 code are required to activate the TF layer. The TC/TM TF layer delivers the spacecraft ID information[2]. It is important to note that several software modules are added (or adapted) in NOS3 in order to handle a multi spacecraft configuration. In particular, the ISL/RM component is added in the satellite flight software (NASA cFS). ISL/RM provides a simple and effective routing machine for the constellation. The Front End (FE) component is added in the MCS VM. FE is the entry point of the MCS, handling the routing with Cosmos and the Standalone[3] CryptoLib [30].

D. General Architecture

The chosen architecture for the communications in CCSDS (over UDP) between one satellite VM and the MCS VM can be summarized by the diagram in Figure 1. It is important to observe that COSMOS and the satellite software bus keep working with SPP, while we can easily capture the traffic at transfer frame level between the Front End component and the ISL/RM component. A more general picture of the simulator architecture is provided in Figure 2, where a multi spacecraft configuration is presented, including an example of addresses and ports for each new component. One can see that ISL/RM is the new entry point of the satellites VMs (the CI application is the entry point for NOS3), while Front End is the new entry point of the MCS VM. The Standalone CryptoLib and COSMOS are also duplicated for each new satellite to be handled into the constellation. NASA 42 is connected to all the spacecraft to provide a unique space environment to the constellation. In the next subsections, we discuss more in details the proposed contributions of this study to NOS3 and we explain more in detail the space system benchmark introduced in CSS project.

E. ISL/RM

ISL/RM is a new component in NOS3 architecture. It implements a routing machine and visibility feature. It also simulates simplified radio connections under UDP for ISL and feeder links. Communication protocol is SPPs in transfer frames over UDP. Routing and network configurations are static by default. ISL/RM architecture is shown in Figure 3.

F. Front End

Front End (FE) is a new ground actor in charge of routing both TM and TC. In addition to ISL/RM component, it implements constellation features: visibility and routing. At any moment it connects all "CosmosTF"[4] and the spacecraft in visibility as shown in Figure 2. Front End is based on transfer frames in both TC and TM. As per configuration (see Figure 4): FE receives TC on one port from all CosmosTF; any CosmosTF may manage one or more (prospective version) satellites; FE receives TM on one port from satellites. Only satellites currently in visibility may send TM to FE. In a future implementation, dynamic routing could be handled. The visibility would evolve along time, linked to the satellites movements around their orbit.

NOTE: The current version of the simulator considers static routing of the constellation messages. This is fully representative of a real constellation network for a limited amount of time. The routing configuration is defined in simple text files (known by ISL and Front End) and it can be easily modified by the user. ISL routing can also be modified by TC. Dynamic routing and thus variable network configuration depending on the satellite positions and visibility with respect to specific ground stations will be considered in further releases of the simulator.

G. Imager

The imager is an external simulation tool with two main functions: to visualize in near-real time what the camera "sees" based on the satellite's attitude; to obtain realistic photos when a photo capture command is sent to the Arducam (NOS3 optical payload). As input, it receives information about the satellite's position and attitude in ECEF frame, the coordinates of a target on Earth to display it in continuous visualization, and the position of the Sun. As output, it produces an image showing the 3D scene as viewed from the onboard camera. The imager is based on GPU technology, which allows modeling the Earth as a sphere with a texture applied to it, and the 1,000 brightest stars and their respective magnitudes. In Figure 5 an example of the imager output is presented in comparison to NASA 42 output.

[1]Pointing the satellite to a specific location, TC routing to request a picture, picture acquisition on-board, telemetry routing with realistic photo, photo decoding and visualization.

[2]This is essential to define a routing strategy for a constellation.

[3]The Standalone CryptoLib process is in charge of encryption for TC/TM messages, but it is also responsible for adding the TC/TM transfer frame layer on top of the SPP layer.

[4]CosmosTF is a couple composed of Cosmos (which handles SPP) and Standalone CryptoLib (which makes the bridge between SPP and transfer frames).

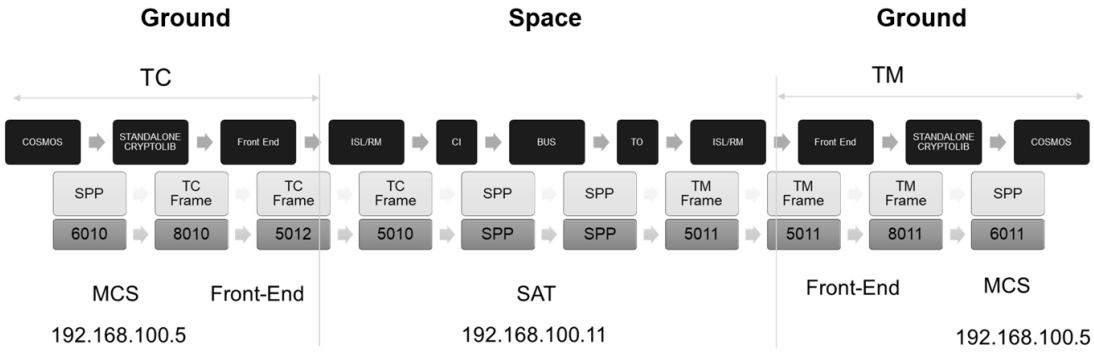

Fig. 1. SAT-MCS end to end communications architecture.

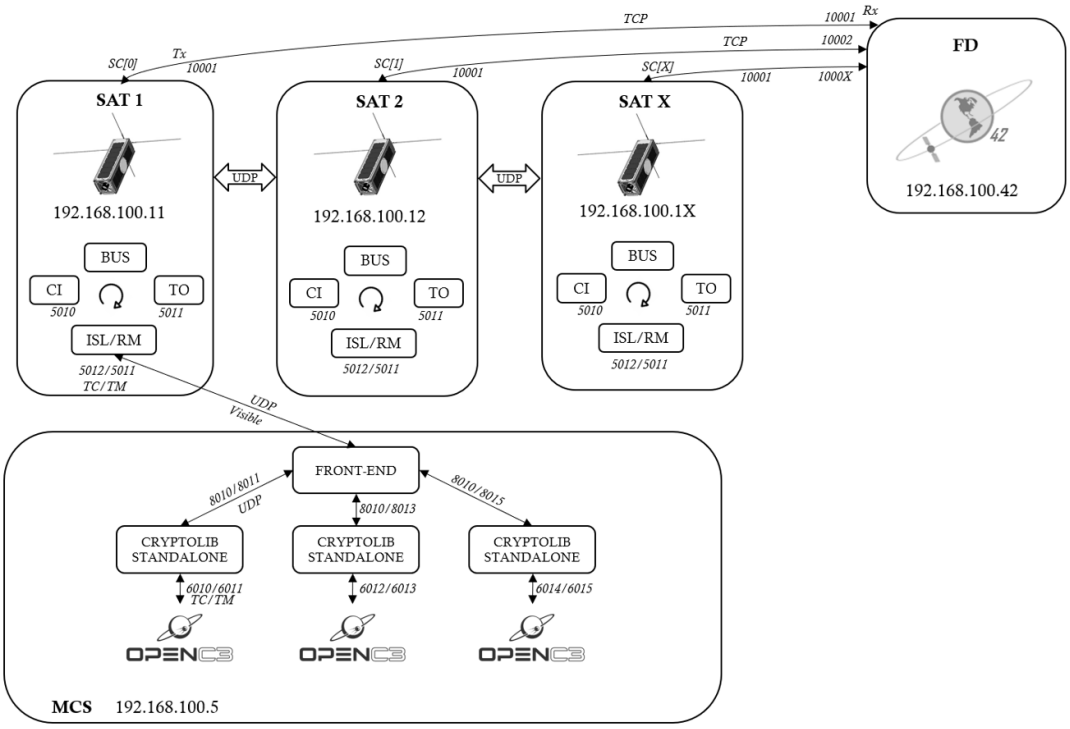

Fig. 2. Overall Architecture of the Simulator.

H. Input generator

The user can also design scenarios sending multiple commands, or perform fuzzing on TC by using a dedicated tool. A pre-filled JSON file is created with the chosen commands, and the values of the different fields can be customized, allowing for both normal and malicious scenarios to be designed. Different fields of the chosen command can be selected for fuzzing, and the number and frequency of packets sent will be user-adjustable.

I. Intrusion Detection Systems

The current version of the simulator includes several intrusion detection and prevention systems, still under development, located on the ground segment and embedded in the satellite itself.

1) The intrusion detection system on the ground segment is performed by means of some specific probes (from Gatewatcher) that have access to the CCSDS traffic on the space link and that are able to detect some anomalies based on statistical and machine learning techniques[5] [21].
2) The intrusion detection and prevention strategies on board are currently investigated though the design of probes embedded in the satellite itself. These strategies are described in details in the next Section IV-C.

IV. Security development and testing

As stated in the introduction, the platform described in this paper is intended to be used as a benchmark for cybersecurity

[5]Gatewatcher probes are out of the scope of this paper and will be presented in further studies.

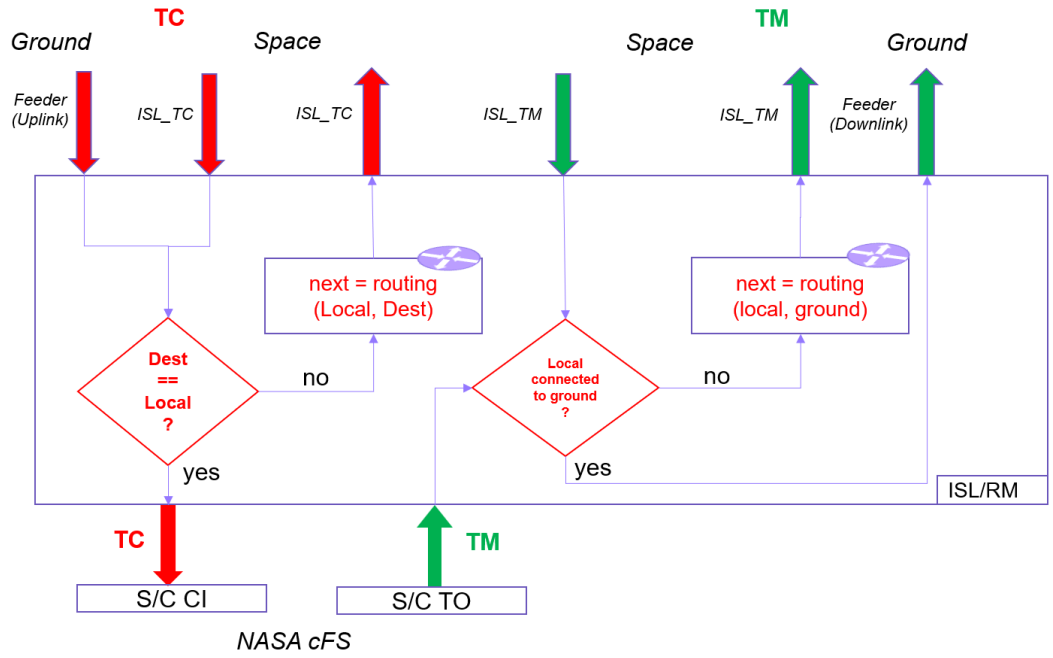

Fig. 3. ISL/RM component functional diagram.

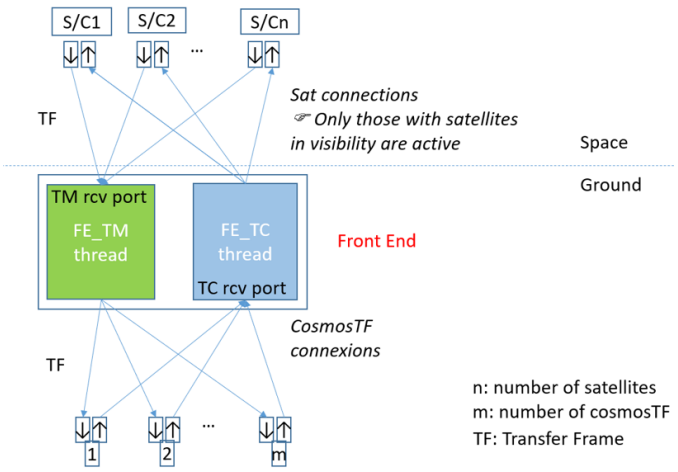

Fig. 4. Front End TM/TC threads and listening ports.

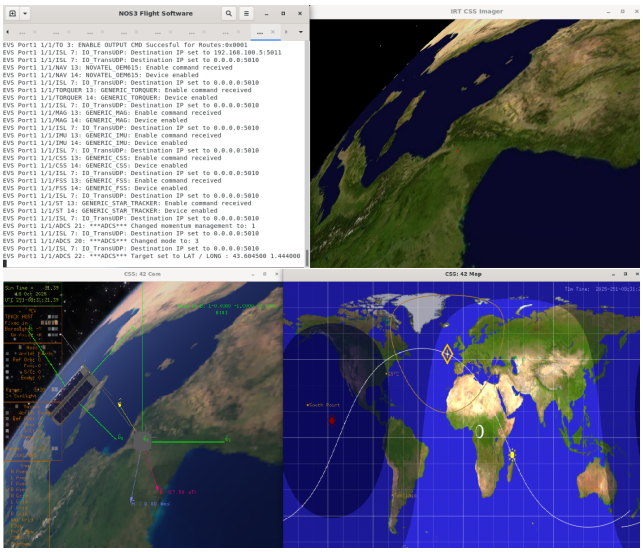

Fig. 5. Example of FSW and Imager process for the Satellite 1 (up), wrt NASA 42 Camera and map (down).

research and development on space systems. For this purpose, the platform includes both offensive and defensive modules. The defensive modules consist in intrusion detection and attack prevention mechanisms, both in the ground segment and embedded in the satellite itself. This paper only focuses on defense mechanisms embedded in the satellite itself and are detailed in Subsection IV-C. The offensive module is aimed at assessing the relevance of the defense mechanisms and implements various attack samples described in Subsection IV-B according to the threat model and a risk analysis presented in Subsection IV-A.

A. *Risk analysis and threat model*

The exploit definition is based on EBIOS-RM [2] risk assessment for a generic LEO constellation, plus a preliminary analysis of NOS3 space system architecture and the associated known vulnerabilities [37] [38] [6] [7] [8] [39]. The NASA cFS architecture is designed to be generic and flexible but it has not been designed to be secure. The considered architecture is nevertheless very common in the space sector and thus it is very relevant for research and development of attacks and mitigation means. Moreover, it is important to note that for

space system with flight software architecture similar to cFS (software bus with applications), the platform can be adapted to specific mission and components, enabling reusability. It is important to observe that new components introduce also new vulnerabilities. For example, Front End application is a single point of failure for the ground segment.

1) EBIOS risk analysis: EBIOS-RM is a risk assessment method which is fully compatible with ISO/IEC 27005 standard [1]. EBIOS-RM method has been used to identify the threats to a generic LEO constellation and the associated attack scenarios. The method allows to quickly identify the most critical scenarios, i.e. those with a maximum "impact X likelihood" product. This selection allowed us to draw a digital threat map for such a generic constellation and to select the most relevant scenarios to be simulated and tested. In particular, about 40 detailed risk scenarios have been identified for our target system, starting from high level risk scenarios as intelligence on the system performance; destruction, life reduction or takeover of the space segment; degradation of the mission; theft of mission data; sabotage of the system in operation. In the next subsections we present a few of the most relevant scenarios that have been tested using the CSS platform. The objective here is to showcase the platform potential for R&D of attacks and mitigation means.

2) Threat model: A large number of research papers focus on ground segment as an attack vector. For example in [43] the vector is a compromised ground station sending illegitimate commands to the satellite. Another strategy for attackers would be using a "rogue" antenna to send commands to the satellite, but accurate tracking of the satellite trajectory would be needed, with a low visibility time frame. In addition, if the satellite and the ground segment use cryptography, sending commands pretending to be the ground station becomes a hard task. Therefore, using the legitimate ground station to access the satellite system seems the best option for an attacker. Another interesting threat consists in compromising the software before being embedded in the satellite (especially the client payload), through some vulnerabilities in the supply chain. This is a common vector in IT security, and as many suppliers are needed to engineer a satellite, the risk increases. Furthermore, as updates for satellite software are performed during operation (especially for the payload), software could be compromised during operation. Under this threat model, the compromised software is running inside the satellite itself and cannot necessarily be detected by means of an IDS analyzing the communications between the satellite and the ground. As such, this leads to the design of an IDS embedded in the satellite itself. We assume that the whole flight system software is trusted, because we want to be able to trust state variables about the satellite provided by the flight software. However, the payload software embedded in the satellite may be malicious and trigger malicious actions at any moment during the lifetime of the satellite. Overall, in our research work, these two threats (compromission of the ground segment and compromission of the payload) are considered.

3) Sandbox Scenario: In Figure 6 an example of CITEF scenario is presented where a swarm with 7 satellites (NASA cFS) is simulated (see Figure 7), including a mission control system (MCS) and a VM for the space environment. A first line of defense is introduced on ground, and it is represented by Gatewatcher Probes[5]. On board the spacecraft, a prototype of an intrusion and prevention system component is installed as part of the FSW. Based on this simple sandbox scenario, several attacks can be tested, a few examples will be provided in the next subsections. It is important to mention here that the simulator can be easily scaled to bigger constellations. Moreover, CITEF offers the possibility to integrate simulated assets with physical assets[6]. One can also consider more complex sandbox scenarios, including multiple steps of the kill chain [24], the main limit of the considered platform is related to the available hardware to host the cyber range.

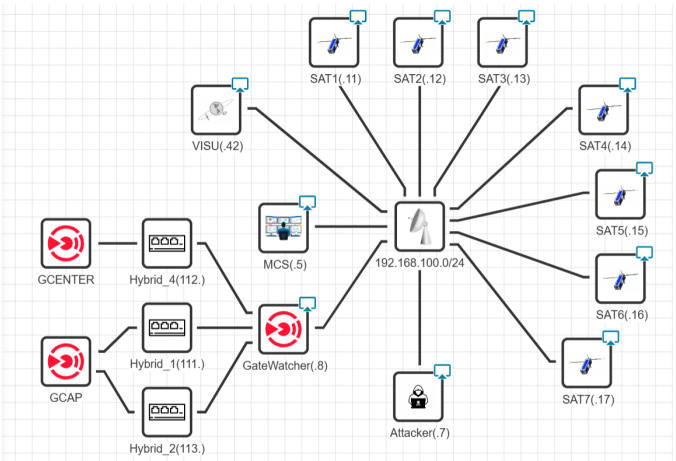

Fig. 6. CITEF Sandbox Scenario.

B. Attack Samples

In this section we describe some of the attacks currently available for testing in CSS platform.

1) FSW flooding: In this scenario, the Attacker in Figure 6 is able to replay continuously a TC to one of the Satellites, the goal being to flood the software bus (SB) and to compromize the SB availability. A simple way to implement this attack is for the attacker to send a space packet continuously to the input UDP port of the Standalone CryptoLib (see Figure 2). This will result in a rapid saturation of the SB; the ISL app will be squelched by the FSW and the satellite will be unable to process any new command. It is important to note that the attack can work also with encryption because the attacker can play the role of a Cosmos instance in the ground segment architecture. In general, a replay attack injected after the cryptolib will be refused if encryption is activated. It is interesting to observe here that another way to implement the FSW flooding is via a compromised application on the

[6]for example the Gatewatcher probes in the sandbox scenario are physical appliance that can get the simulator CCSDS traffic via specific interfaces called "Hybrid".

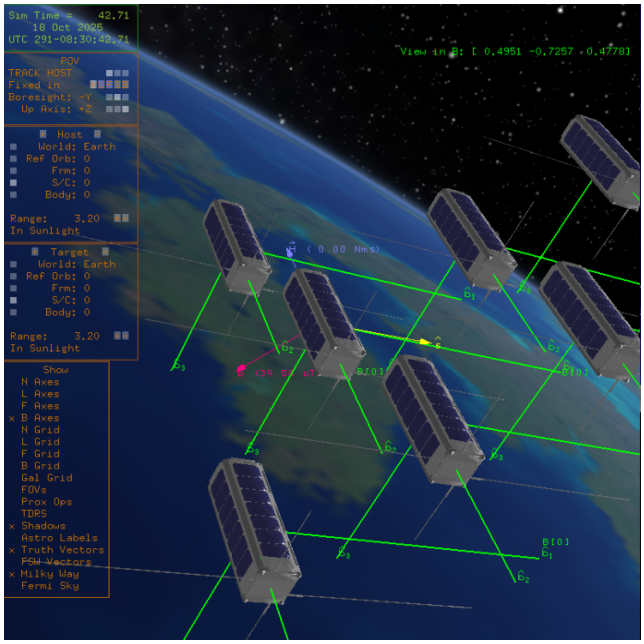

Fig. 7. Swarm of CubeSats seen from NASA 42 Camera.

software bus. Indeed, there is no layered permission on the SB and a malicious application can perform several types of attacks. The threat model would be slightly more complex in this case. First, an attacker is able to compile a malicious payload (CAM Application) in *.so* format that floods the SB with a telemetry packet when the operator asks for a picture via TC. Second, the *.so* file should be loaded into the satellite (this can be performed before the spacecraft launch or directly via TC using the CFDP protocol [17]). Third, the attacker can replace via TC the normal payload with the malicious payload using cFS commands. The spacecraft will operate normally until the flooding is triggered by the attacker or by the operator requesting a picture. The SB jamming attack can be detected or mitigated by defense mechanisms using simple monitoring of the traffic on the bus (as explained in the subsection IV-C).

2) SDLS Vulnerability Exploitation: In [6] and [7] some vulnerabilities of the CryptoLib are exposed. In CSS platform it is easy to verify the impact of CVE-2024-44911 (Segmentation fault on TC frames). For example, if the attacker crafts a TC transfer frame where the SPI (Security Parameter Index) that corresponds to the 7th and 8th bytes of the TC frame has been set to 0xFF 0xFF (instead of 0x00 0x04 for an ecrypted frame [30]). In NASA CryptoLib v1.3.0 (and previous versions) this will trigger an out-of-bounds vulnerability leading to a segmentation fault and this will induce a crash of the on-board FSW that depends on the CryptoLib. The FSW will restart automatically, but an attacker can send the same TC continuously to put the satellite out of service.

3) App Kill: In [38] a Cyber-ASAT and App Kill attacks are presented. In particular, these attacks consist in deleting the on-board applications (one or multiple app) by leveraging the existing cFS commands.

4) App Delete: In this scenario TCs are sent to erase a library (.so file) and to stop the app using this library. This attack can be implemented via the MCS or via a malicious payload on board the satellite as for IV-B1. The goal of this attack is to put a specific application out of service without the possibility to restart.

5) Spacecraft ID Sabotage: The possibility to test and develop attacks on a constellation is one of the main goals of CSS project. If the attacker has access to the mission control system, he can easily replace the Front End process with a malicious version that can perform several types of attacks on a specific spacecraft and on the constellation. For example, by simply switching the spacecraft ID between 2 satellites in the Transfer Frame packets, the attacker can cause a lot of damage on the constellation. The operator that is controlling one satellite will send commands to the other and vice versa. In case of attitude and orbital maneuvers this can represent a serious concern. Moreover, a malicious version of Front End could switch the SCID coherently between TC and TM, reducing the possibilities for the ground agents to detect the attack. Exploiting a malicious Front End the attacker could also create loops in the constellation communications as in [45], inducing severe network service disruption. The final result of the attack will inevitably depend on the commands selected by the operator, however, the mission is generally compromized. Concerning the mitigations, the detection can be very difficult, thus dedicated detection and defence strategies shall be put in place combining space and ground information. The detection or prevention of this attack is not implemented as for now, but it will be considered in a near future.

C. Intrusion detection and prevention system

1) Probes: The IDS module embedded in the satellite uses a multi-layer approach with a set of probes (in different locations of the architecture), and a set of detection modules working either independently or together. We investigate the design of three specific probes. The first, located at the arrival of each TC in the satellite (i.e. in the ISL/RM component), is aimed at capturing some features of the network traffic exchanged between the satellite and the ground segment: bandwidth, packet inter-arrival time, size of packets, etc. As we assume that cryptography is enabled, this probe is not able to get the content of the packet and thus can only get some characteristics at the network flow level. The second probe is aimed at reporting information on deciphered TCs coming from the ground, and not sent yet on the Software Bus, to distinguish from internal messages. This probe is located into the CI component on the NOS3 platform. This probe captures both the command type and command parameters and can provide them to an intrusion detection and prevention system dedicated to detect or drop any malicious TC, as described above in the threat model. The third probe is aimed at capturing information in the same way as the second one, but is located on the Software Bus. This positioning allows to gather information related to the internal exchanges of the satellite. The aim is to collect activity coming from all com-

ponents of the satellite. This probe is currently implemented by modifying the software bus itself in order to capture all messages exchanged on the bus along with the source of the message. Information gathered by these probes are used as an input to the intrusion detection and prevention algorithms. A representation of the architecture of NOS3 and the location of the probes is given in Fig.8.

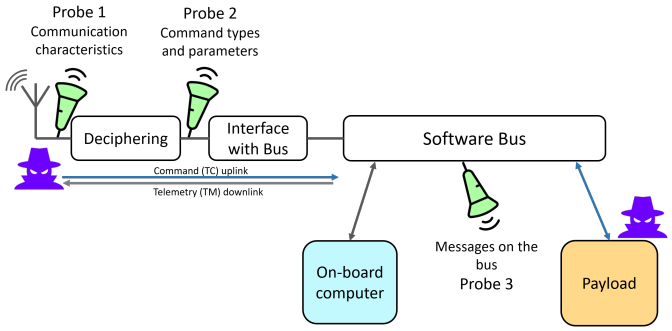

Fig. 8. Embedded probes in NOS3 architecture

It is important to note that, to have defense mechanisms as accurate as possible, a standard model of commands sent from the ground which would represent real mission operation is needed. This is the main goal of a prototype mission defined in the MCS VM included in the CITEF scenario. It is important to mention here that in the current version of the simulator there is no Fault detection, isolation, and recovery (FDIR) system on-board and there are no complex state machine reconfiguration associated to FDIR. This type of enhancements can be considered in further studies.

2) Intrusion detection and prevention strategies: The IDS/IPS has a distributed architecture tailored to NASA cFS software (in the form of C language files) in order to provide a realistic implementation of an onboard component with limited ressources. Using the first probe's information, an anomaly detection strategy based on the analysis of the packets characteristics (typically, inter-packet time, size of packets) is considered. The aim is to detect deviations from standard operations behavior, which could indicate an attack. A classic example is a Denial of Service (DoS) attack targeting the satellite, which consists in flooding the satellite with packets coming from the ground. As such, this detection mechanism allows to detect the attack described in IV-B1 by comparing the packet inter-arrival time to a threshold, representative of the legitimate communications. We also currently implement a intrusion prevention strategy by means of an anti-flooding approach in which we drop packets that do not respect the legitimate packet inter-arrival time threshold. An other intrusion prevention strategy is currently implemented using the information extracted from this first probe. Even if the content of the packets gathered by the first probe are encrypted, the header is not. Some attacks, such as the attack described in IV-B2 can be prevented by monitoring header fields, and in this case dropping packets that include incorrect SPI values. An intrusion prevention strategy using the second and third probe is currently implemented and consists in a "TC firewall". This component is in charge of checking the validity of the TC identifier, and deploying a black list of dangerous TC. The aim is to address elementary attacks that use a single malicious TC, both sent from the ground (information gathered by the second probe) or sent on the bus if the payload was compromised (information gathered by the third probe). Typically, this TC firewall is able to block the attack described in IV-B3 by means of a filtering rule that forbids the use of the command that deletes all the configuration file (i.e., `FM_Delete_All` TC with the `/cf` parameter). The TC firewall is also able to drop dangerous combinations of TC. Typically, the attack described in IV-B4 can be dropped by stateful filtering rules that are able to identify the combination of two TC that, taken together, perform a DOS attack towards a specific component of the flight software. These stateful rules are currently being implemented. Finally, we currently implement an anomaly detection strategy using the information gathered by the third probe. Our detection strategy consists in establishing a model of the legitimate communications on the software bus and identifying anomalies that could reveal malicious actions. These actions could either be performed by a legitimate software component, on receipt of a malicious TC, or by a malicious payload. The challenge here is to have a model of legitimate messages exchanged on the bus, so that we can detect deviations in these communications. We are currently investigating different AI models, taking into account that our detection algorithms must be embedded in a platform that has limited resources in memory and computation.

V. Conclusions and Perspectives

Several long-term and short-term mitigations can be proposed to increase the resilience of the simulated space system to cyber attacks [38] [34] [7]. These mitigations will be discussed in further studies. The main objective of this study is to introduce a benchmark for space systems cybersecurity R&D. The CSS simulator is based on NASA NOS3 and it has been adapted to represent a single spacecraft or a constellation. The characteristics, contributions and the potential of the proposed platform for cybersecurity have been described. In particular, the introduction of new components and features that enable the user to develop and to test several attack/defence scenarios on a representative constellation of spacecraft with optical payloads. CSS platform is a starting point for a wide range of research paths in space system cybersecurity. In particular, it is possible to work on topic such as automatic attack generation and automatic threat detection/mitigation based on realistic CCSDS traffic. New cybersecurity components, as the intrusion detection and prevention system proposed in Section IV-C, can be developed and validated to increase the system resilience. Existing and new algorithms for threat detection can be compared on selected benchmark scenarios. Realistic data sets for surrogate models or for direct training of artificial intelligence algorithms can be generated. A flexible, realistic and open-source simulation platform is essential to work on the cyber resilience of today and tomorrow space systems.

References

[1] ISO/IEC 27005:2022. Information technology, security techniques, information security risk management, 2022. https://www.iso.org/standard/80585.html.

[2] Agence nationale de la sécurité des systèmes d'information (ANSSI). La méthode EBIOS Risk Manager, 2024. https://cyber.gouv.fr/la-methode-ebios-risk-manager.

[3] Ansys. Ansys STK: Digital Mission Engineering Software, 2024. https://www.ansys.com/products/missions/ansys-stk.

[4] Brandon Bailey. Nasa iv&v's cyber range for space systems. In *Annual Ground System Architectures Workshop*, number GSFC-E-DAA-TN65725 in GSAW, 2019.

[5] N. Boschetti, N.G. Gordon, and G. Falco. Space cybersecurity lessons learned from the viasat cyberattack. In *ASCEND 2022*, 2022. 4380.

[6] Ayman Boulaich, Andy Olchawa, and Milenko Starcik. How to crash a Spacecraft – DoS through Vulnerability in NASA CryptoLib v1.3.0, 2024. https://visionspace.com/crashing-cryptolib/.

[7] Antonin Boulnois. Exploring Vulnerabilities in the SDLS Implementation of NASA's CryptoLib, 2024. https://securitybynature.fr/post/hacking-cryptolib/.

[8] Ronald Brobert. Fuzzing NASA Core Flight System Software. DEF CON 29 Aerospace Village. Available at: https://www.youtube.com/watch?v=YtExxB-ayo0, year = 2021.

[9] CNES with SPACEBEL. Basiles numerical simulation framework. https://basiles.fr/.

[10] Consultative Committee for Space Data Systems. *TC SPACE DATA LINK PROTOCOL*. CCSDS, 2015. CCSDS 232.0-B-3.

[11] Consultative Committee for Space Data Systems. *Space Packet Protocol*. CCSDS, 2020. CCSDS 133.0-B-2.

[12] Consultative Committee for Space Data Systems. *TM SPACE DATA LINK PROTOCOL*. CCSDS, 2021. CCSDS 132.0-B-3.

[13] COSMOS. Openc3 cosmos. https://docs.openc3.com/docs.

[14] European Space Agency. ESA SPACE-SHIELD: Space Attacks and Countermeasures Engineering Shield, 2023. https://spaceshield.esa.int/.

[15] European Space Agency. NanoSat MO Framework, 2024. https://github.com/esa/nanosat-mo-framework.

[16] European Space Agency. SIMULUS-NG: A New Era for Satellite Simulation, 2024. https://esoc.esa.int/content/simulus-ng-new-era-satellite-simulation.

[17] Consultative Committee for Space Data Systems. *CCSDS File Delivery Protocol (CFDP)*. CCSDS, 2020. CCSDS 727.0-B-5, Recommended Standard.

[18] Consultative Committee for Space Data Systems. *Space Data Link Security Protocol*. CCSDS, 2022. CCSDS 355.0-B-2.

[19] Laurent Franck. *LEO Satcom Cybersecurity Assessment*. European Union Agency for Cybersecurity (ENISA), 2024. https://www.enisa.europa.eu/sites/default/files/publications/LEO_satcom_cyber_security_assessment_240214.pdf.

[20] Gatewatcher. Gatewatcher: Network Detection and Response (NDR) Solutions, 2024. https://www.gatewatcher.com/.

[21] Gatewatcher. *GCenter Documentation: Motors Menu*, 2025. https://docs.gatewatcher.com/en/gcenter/2.5.3/102/02_function/motors/motors_menu.html.

[22] Martin Goetzelmann, Niklas Lindman, Peter Ellsiepen, Christian Laroque, Marc Niezette, and Anthony Walsh. Ground systems test and validation infrastructure-from concept to implementation. In *SpaceOps 2006 Conference*, page 5540, 2006.

[23] Hack-A-Sat. Hack-A-Sat: Capture the Flag Competition, 2024. https://hackasat.com/.

[24] Eric M Hutchins, Michael J Cloppert, Rohan M Amin, et al. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[25] INET Framework. OS3: Satellite Tracking Library for OMNeT++/INET-Framework, 2015. https://github.com/inet-framework/os3.

[26] M. Manulis, C. P. Bridges, R. Harrison, V. Sekar, and A. Davis. Cyber security in new space. *International Journal of Information Security*, 20:287–311, 2021.

[27] David McComas. Increasing flight software reuse with opensatkit. In *2018 IEEE Aerospace Conference*, pages 1–8. IEEE, 2018.

[28] NASA. Nasa operational simulator for small satellites (nos3), v1.6.2, 08/2023. https://github.com/nasa/nos3.

[29] NASA. Core Flight System (cFS), 2024. https://github.com/nasa/cFS.

[30] NASA. CryptoLib: CCSDS Space Data Link Security Protocol - Extended Procedures (SDLS-EP), 2025. https://github.com/nasa/CryptoLib.

[31] NASA Goddard Space Flight Center. The Spark: Fall 2024, 2024. https://partnerships.gsfc.nasa.gov/wp-content/uploads/The_Spark_Fall_2024_DIGITAL-S.pdf.

[32] Nexova Group. Cyber Range and Emulation Solutions, 2024. https://www.nexovagroup.eu/en/cyber-range-and-emulation-solutions.

[33] OpenSatKit. OpenSatKit: Core Flight System (cFS) Application Developer's Kit, 2021. https://opensatkit.github.io/.

[34] Ashok Prajapati. Evolution of core flight system (cfs). In *AIAA SciTech Forum*, 2025.

[35] Meghan Galiardi Sahakian, Srideep Musuvathy, Jamie Thorpe, Stephen Verzi, Eric Vugrin, and Matthew Dykstra. Threat data generation for space systems. In *2021 IEEE Space Computing Conference (SCC)*, pages 100–109. IEEE, 2021.

[36] Sara Salim, Nour Moustafa, and Martin Reisslein. Cybersecurity of satellite communications systems: A comprehensive survey of the space, ground, and links segments. *IEEE Communications Surveys & Tutorials*, 2024.

[37] Adrian Schalk, Luke Brodnik, and Dane Brown. Analysis of vulnerabilities in satellite software bus network architecture. In *2022 IEEE Military Communications Conference*, 2022.

[38] Adrian Schalk and Dane Brown. Detection and mitigation of vulnerabilities in space network software bus architectures. In *2023 IEEE Aerospace Conference*, 2023.

[39] Milenko Starcik, Andrzej Olchawa, Ricardo Fradique, and Ayman Boulaich. NASA cFS Version Aquila Software Vulnerability Assessment. *VisionSpace*, 2025. https://visionspace.com/nasa-cfs-version-aquila-software-vulnerability-assessment/.

[40] Eric Stoneking. Nasa 42. https://github.com/ericstoneking/42.

[41] Eric Stoneking. 42: An open-source simulation tool for study and design of spacecraft attitude control systems. Technical report, NASA, 2018.

[42] The Aerospace Corporation. SPARTA: Space Attack Research and Tactic Analysis, 2024. https://sparta.aerospace.org/.

[43] John Thebarge, Wayne Henry, and Gregory Falco. Developing scenarios supporting space-based ids, October 2022. https://arc.aiaa.org/doi/10.2514/6.2022-4219.

[44] Marcus Wallum, Daniel Fischer, Jadwiga Nowotnik, Łukasz Pieczonka, and Mariusz Tkaczyk. Sec_lab: A secure communications testbed for space missions. In *Space Operations: Beyond Boundaries to Human Endeavours*, pages 447–469. Springer, 2022.

[45] Yikun Wang, Hewu Li, Zeqi Lai, and Jihao Li. Starmaze: Ring-based attack in satellite internet constellations. In *2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS)*, 2024.

[46] Yan Zhang, Yong Wang, Yihua Hu, Zhi Lin, Yadi Zhai, Lei Wang, Qingsong Zhao, Kang Wen, and Linshuang Kang. Security performance analysis of leo satellite constellation networks under ddos attack. *Sensors*, 22(19):7286, 2022.

[47] Yaoying Zhang, Qian Wu, Zeqi Lai, Yangtao Deng, Hewu Li, Yuanjie Li, and Jun Liu. Energy drain attack in satellite internet constellations. In *2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS)*, pages 1–10. IEEE, 2023.