Proposed Security Measures for Code Injection for CubeSats

Alexandre Henrique Radis [*], João José Costa Gondim [†], Daniel Chaves Café [‡]
Department of Electrical Engineering of UnB - University of Brasilia[*]
Department of Computer Science of UnB - University of Brasilia[†]
Department of Electrical Engineering of UnB - University of Brasilia[‡]
Email: ahradis@gmail.com[*]

2022 Workshop on Communication Networks and Power Systems (WCNPS) | 978-1-6654-7456-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/WCNPS56355.2022.9969688

***Abstract*—Sometimes we have the need to inject new services in an operational satellite, but as the injection of new codes in equipment that has communication link is a critical process due to the possibility of injection of broke or malicious codes, this document proposes a protocol for the safe injection of code in satellite microcontrollers of the CubeSat' type. This protocol is based on the use of HMAC with SHA-3 to guarantee integrity and authenticity and is enhanced by the same security measures to mitigate communication link problems and satellite attacks, such as the guarantee of delivery and displacement between communication windows and periods of high processing.**

***Keywords*—HMAC, Safe Code Injection, CubeSat and satellite.**

I. Introduction

In a satellite development and launch project, several assumptions are made that will guide its development for the definition of objectives, cost, time and other project characteristics. These assumptions are elaborated in the initial phase, being adapted to the new conditions during their period of development until the launch. However, the needs keep changing, either due to the emergence of new threats to the project, or the emergence of new business opportunities or other possibilities. Thus, it is interesting that the design of a satellite allows following this evolution and can be reconfigured for new operations during its useful life.

The reconfiguration of a satellite can be something as simple as redirecting the image capture cameras or changing a sensor calculation constant, which can be done via a function that receives the indicated parameters. Or it could be the inclusion of a new service, which is more complex as it requires the injection of new codes into the programming of the satellite's computer system. This change in satellite programming is a delicate process as it involves security issues, especially the integrity and authenticity of the new code to be injected into the computer system.

Code integrity is necessary to prevent incomplete code from being executed, which could harm the satellite's operation, or even its operation output. Authenticity is necessary to prevent malicious code sent by third parties from being executed, which could lead to satellite malfunction, spying and other improper uses or modifications of the information contained in the satellite, including the hijacking of satellite control by third parties. The guarantee of the integrity and authenticity of the new codes are handled by computational processes that are increasingly complex and of high computational cost. However, a quick risk assessment of the non-use of these guarantees, loss of the satellite, practically inhibits the possibility of their non-use.

Today, satellites called "CubeSat" are considered "low cost" and orbit the planet at altitudes between 500 and 1,800 km above the earth's crust, providing a wide range of services. But these satellites have restrictions on computational power, communication capacity and energy availability, among others, requiring the rational use of their resources. The communication system between the base and the ”CubeSat” type satellite has some peculiarities not normally found in systems of IoT - Internet of Things and other telemetry or computational systems.

Among these peculiarities are the low transmission/reception rate, high packet loss and short communication window periods. Part of these peculiarities in Brazil can be explained by the South Atlantic Anomaly - SAA that makes these communications difficult. The transmission rate is below 9600 bits per second - bps and the packet loss rate in transmissions, up to 40% from base to satellite and 20% from satellite to base. And due to its heliosynchronous orbit with an altitude of 500 km for the case studied, communication is only made in short windows of up to 10 minutes because the total period in this orbit is approximately 100 minutes. An aggravating factor is that communication is more effective in a few passages, leading to a window of no communication that can reach up to 12 hours.

Thus, we have the need to include new services in equipment miles away from the earth’s crust as a way of optimizing its use under adverse communication conditions is complex and challenging, especially if we consider the security requirements necessary to avoid the malicious use of code injection. And in our studies, we did not find measures in the literature that could reconcile the guarantee of integrity and authenticity in code injection with the aforementioned conditions.

This absence of a methodology led us to compile in this work a combination of measures, techniques and protocols that seek to guarantee integrity and authenticity in the injection of legitimate (non-malicious) code into ”CubeSat” satellites. It is important to consider that these measures are independent and complementary in their implementation, allowing their evolution or replacement according to technological evolution. Another important point is the efficiency of the protocols, because as seen, the natural losses of the system are already great. The work continues with the literature review, the proposed protocol, the testing methodology, the results found, the conclusions and the bibliographic reference.

II. Literature Review

P. Yue et al. did a study on the different forms of attack on satellites in general [1], these attacks can be of the type: man-in-the-middle, replay, obfuscation, among others, citing countermeasures to protect information and keep the satellite operational. Another concern in the development of “CubeSats” is energy savings, I. Bisio et al. produced a paper comparing resource allocation methods in systems that have high packet loss with a focus on energy consumption metrics for geostationary satellites [2]. Security and power consumption being two concerns often shared with the IoT device universe.

In IoT there are several concerns about vulnerabilities as cited in the works of R. Ramadan [3], P. Rahimi et al. [4] and R.P. Lee et al. [5], being the biggest concern with confidentiality. The works of T. Hiscock et al. [6] and M. A. Philip et al. [7] demonstrate the concern with the possibility of physical access to the internal information of the system and the work of S. Aruna et al. [8] deals with code integrity attacks. The work of M. Hwang et al. [9] aims to use satellites for intercommunication based on confidentiality, as well as the work of T. Ferrer et al. [10] proposes the use of networked nanosatellites to communicate with remote areas.

However, due to the communication difficulties mentioned in the introduction by the SAA phenomenon, cited by J. Wikman et al. [11] and R. R. I. Taxonera [12], we must look for a form of communication that guarantees integrity and authenticity in this adverse environment. In our knowledge of the literature, we did not find works that only deal with integrity and authenticity, not dealing with confidentiality, in the case of "CubeSats". Therefore, our work seeks measures that can guarantee integrity and authenticity in the injection of codes into ”CubeSat” satellites under SAA conditions.

We found an orientation of the technologies used in IoT in the work of M. Sinha and S Dutta [13], where several comparisons between techniques of Lightweight cryptography and its comparison are demonstrated, in spite of presenting basically techniques of confidentiality and not of integrity and authenticity as we are proposing.

III. Proposed Protocol

The protocol aims to guarantee the integrity of the code and the authenticity of the source that sends the code to the satellite. This protocol uses some security measures, cryptographic authentication primitives, an Integrity and Authenticity Assurance Protocol - IAAP, and a Delivery Assurance Protocol - DGP, although the latter is not essential. The division of the protocol into two parts, as shown in the figure 1, is due to the fact that the communication window is small. It is proposed that, during the communication window, only data transmissions are carried out with the DGP acting on the necessary packages, leaving the IAAP performance for the periods between the communications windows, since the IAAP requires greater computational effort.

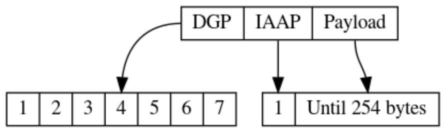

Figure 1. package map

A. Threat Model

The situation aimed at providing security concerns the ability to safely inject new codes into the satellite’s computer system, covering two possibilities: malformed code injection (integrity) and the ability of an adversary to inject new code into the satellite’s processor to alter the system’s programming to gain control of the satellite (arbitrary code execution). One should consider the situation where the attacker’s target system has only one execution privilege level, i.e. the user runs the code with full access to system resources.

The first possibility corresponds to the execution of codes that are not intact, which can cause from the malfunction of the equipment or its total unavailability by the execution of codes that have code flaws, causing the loss of the investment made in the satellite. In the second possibility, it is understood that there is an adversary aiming to inject code under its control for the operation of the satellite, this possibility is critical for allowing the hijacking and malicious use of the satellite.

In the case of satellite hijacking, the attacker who manages to inject code to change the satellite commands and make it inaccessible to the legitimate owner can demand a value for the return of satellite control, this attack mechanism is similar to data hijacking in servers called ransomware[14]. Satellite attack can use the same principle with changing the communication protocol to make the satellite inaccessible, the objective being purely financial. The case of satellites that have propulsion is particularly worrying, since if the attacker gains control of the satellite, he can use the satellite for the purpose of physical attack to other satellites in orbit, using the satellite as a projectile to knock down other satellites with more secure communications.

Another case of code injection into the satellite is the malicious use of the information contained therein, whether it is the simple unauthorized acquisition of information, or the manipulation of information in order to maximize or minimize the information transmitted by the satellite (fire count in the Amazon Forest, for example), or even the unavailability of the information to the legitimate owner, in this case without the possibility of returning control.

B. Security measures

Due to the satellite's orbit characteristics and the code injection process being essential today, but not a priority, we can establish the following security measures:

- *Communications between base and satellite for exchanging messages regarding code injection will only be done in the most effective windows, every 12 hours.*
  This measure aims to not impact other communications, reduce energy waste and reduce man-in-the-middle, replay and base obfuscation attacks, and eliminate code injection attempts by other bases located outside the optimal range of Communication.
- *Cryptographic processing will be done in the periods between communication windows.*
  This measure aims not to impact communications during the short period of communication windows, as it minimizes processing during this period and promotes a withdrawal period in code injection.

C. Authentication Encryption

To guarantee integrity and authenticity, we will use HMAC - Hash-based Message Authentication Code with SHA3-256, these security practices being largely guided by the NIST - National Institute of Standards and Technology to guarantee integrity and authenticity. It should be noted that the basis of security is the inviolability of the password, which can allow attacks such as social engineering and brute force.

Social engineering can be mitigated with the use of password generation, distribution and storage systems that do not involve human operators, and the brute force attack is practically infeasible, given that a 256-bit password is equivalent to approximately 2^{256} combinations, if a computer processes a possibility every 1 nanosecond, equivalent to a full operating frequency of 1GHz, it would take approximately $3.7 * 10^{60}$ years to cover all possibilities.

D. IAAP - Integrity and Authenticity Assurance Protocol

To identify the function of each received packet, we will use 1 initial byte, as indicated in the table I. Where the most significant part of the byte corresponds to the system code (current system, code 1) and the least significant part corresponds to the package's role in the system.

Code (hex)	Function
10	Binary to be injected
11	Hash
12	Confirm shipment
13	Validation OK
14	Validation Failed
15	Implementation Ok
16	Technical Failure
17	Security Failure

Table I
List of Codes

It should be noted that this codification allows the application of the idea to other processes, being able to have, in principle, 16 different systems, with up to 16 codes in each system, to be changed according to the needs of each project. However, in order to create generic and aggregator codes, system 0 will be used for these codes. Thus, code 00 must be used as an aggregator of several simple validation codes that do not have arguments.

The operation of the integrity and authenticity protocol is formed according to the figure 2 as follows:

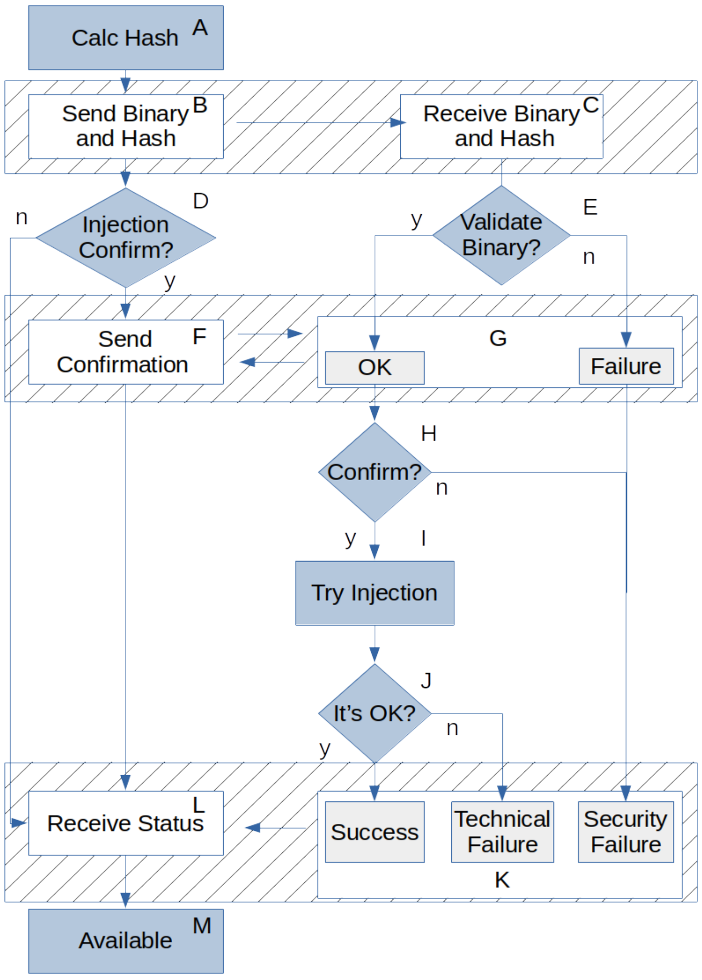

Figure 2. Protocol Flow

- *A* - The system on base calculate the hash for the binary;
- *B and C* - In the period of communication windows, The base sends the binary (10) and hash (11) to the satellite;
- *D and E* - In the period between communication windows, the system decide by the confirmation(D) and the satellite validates(E) the binary with the hash, the satellite has 12 hours for these validations;
- *F and G* - In the period of communication windows, the base sends confirmation that the hash has been sent, codes 12, this confirmation is important because it allows a window of 12 hours to identify possible errors or withdrawal in the code sent. And the satellite sends the result of the validations, codes 13 or 14;
- *H, I and J* - In the period between communication windows, in case the satellite has positive validations (H - Codes 12 and 13) try to implements the code(I), otherwise prepares sending an error to the base. In case the tried work(J), prepares sending Success, otherwise prepares sending an technical error.
- *K and L* - In the period of communication windows, the satellite sends to base the implementation success operation (15), implementation technical failure code (16), or security failure code (17). In the case of code 17, it sends the binary and hash for possible analysis;
- *M* - This step exists only in case of failure (16 and 17): The base analyzes if had a technical error or the binary and the hash received for search of a possible attack.

E. DGP - Delivery Guarantee Protocol

Due to the peculiarities of the operating context, it is interesting to establish a communication protocol that guarantees the delivery of packets and then a protocol that guarantees that the code sent by the base is intact, as well as the authentication of the base as the sender of the code to be used.

For the elaboration of the DGP, we made the mathematical study of the table II, encompassing the 7 bytes of header, as proposed in the figure 1. We consider a block of 2560 bytes of information to be transmitted in a scenario of 40% loss on the way up and 20% on the way down. Losses were not considered in reshipment situations. A time between packets of 1 millisecond was also considered to avoid collision and overlap. We also consider 20 milliseconds as the time between the satellite receiving the complete block, analyzing and responding with an acknowledgment or start of retransmission, if necessary.

In the II table, the “Size” column represents the size of the packet and varies from 512 bytes to 32 bytes. The “Quantity” column illustrates the number of packets resulting from the division between the total block size by the packet size. Here we consider the packet bytes plus the header bytes. The “Time (sec)” column shows the total transmission time considering the packets, the header and the interval time as mentioned in the previous paragraph.

It can be seen from the table that the 256-byte packet is the most efficient. A secondary advantage of using this size is that it can be represented in 1 byte.

The reason we consider a 7-byte header for each packet is the configuration of a useful header, including for a possible internationalization of the protocol. Thus, the header was designed as follows:

1st byte: Country code for international calls;

Size	Quantity	Time (sec)
512	6	3.44
256	11	3.24
128	22	3.35
64	45	3.44
32	98	3.86

Table II
Study of 2560 bytes transmission in packets of different sizes in a lossy scenario

2nd and 3rd bytes: Country equipment identifier code, up to 65535;

4th byte: Window packet number;

5th byte: CheckSum of the packet;

6th byte: Packet size;

7th byte: Packet aggregation: the most significant part is the number of the sub-package and the least significant is the number of sub-packages;

With these configurations, the satellite can identify in the first three bytes if the message is for it, being able to save energy while waiting for a message that is directed to it, something close to what exists today in the IP protocol - Internet Protocol. Note that unlike the IP protocol, the purpose of the proposed protocol is base-satellite communication, and not the creation of a satellite network. Therefore, a base can communicate with several satellites, but communication between the satellites is not foreseen.

Packets 4 to 6 have the purpose of managing the packet in the window and guaranteeing the integrity of the received packets, with the previous discard of the same in case the size or the checksum is not met, avoiding the receipt of broken packets and minimizing the use of processing. not necessary. The 7th byte has the function of indicating the need for packet concatenation to form sequences of bytes longer than 256 bytes. Note that this value is independent of the packet number indicated in byte 4, however these packets must be sequentially numbered to prevent confusion.

IV. Methodology

For the tests of the proposed measures, an infrastructure was set up in a WiFi network consisting of a cell phone as an access point and a notebook with two virtual machines, one simulating the satellite control base for code injection and the other a possible attacker aiming inject malicious code. And to simulate the satellite, an ESP-32 development kit was connected to the WiFi network to simulate the satellite communication radio and this connected via UART to an Arduino Mega development kit to simulate the satellite processor.

We chose the Arduino Mega for being a low cost version, easy access and its similarity with the MSP-430 processors used in ”CubeSats”, and the ESP-32 choice for the radio interface was due to its high processing power used for the various services required by the simulation. Among these services is the communication via UDP between the ESP-32 and the virtual machines, the simulation of communication failures between the base and the satellite, the generation of traffic reports and the detailing of the packets. From this point on, we will call the Arduino a satellite, the ESP-32 a radio and the virtual machines for their functions, base and attacker, as shown in the figure 3.

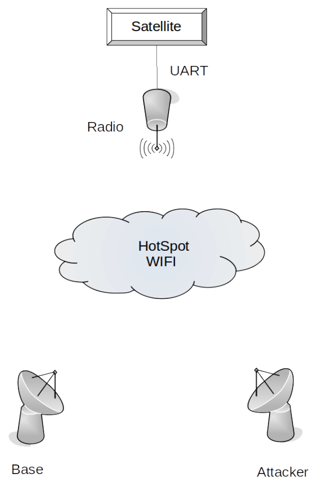

Figure 3. Infrastructure

The radio in this architecture has the function of simulating the adverse environment and giving a slight advantage to the attacker, for that it was programmed to:

1 - Transmit the signals from the base to the satellite with 40% failure, receiving via UDP on port 1818 and transferring to the satellite via UART;

2 - Transmit satellite signals to base with 20% failure, receiving via UART and transferring to base via UDP on port 1818;

3 - Relay all communications between base and satellite to the attacker via UDP on port 1881;

4 - Transmit the attacker’s signals to the satellite without fail, receiving via UDP on port 1881 and transferring to the satellite via UART;

5 - Generate web page with all transmissions, including blocked ones.

The satellite and the base were programmed to meet the requirements of the protocol presented, and the base was programmed to indicate the beginning and end of the communication windows, in a real case this would be defined by timers and/or geolocations. The attacker was programmed to receive the transmissions and simulate code injection by trying random passwords.

V. Results

The first phase of tests did not cover the attacks, the base attempted to inject correct, incomplete and altered codes after the generation of the HMAC hash. The satellite accepted and deployed the codes with the correct hash, those that had any code or hash changes were returned as not deployed, and those that had no deployment code were also not deployed. This test demonstrated that if there is any code change after the HMAC Hash is generated, the protocol was able to identify it and did not implement it, guaranteeing the integrity of the system.

The second phase of testing started with the attacker trying to get communications with the satellite in periods outside the communication window, opportunities where the satellite simply disregarded the attacker's attempts, not returning responses to the attacker and the base. This test demonstrated the effectiveness of the proposed security measures, as communications are only accepted during the expected window.

Subsequently, the attacker tried to inject a code with any password during the correct periods of the communication window, which was returned by the satellite as improper because the hash and HMAC do not match the password shared between the base and the satellite. . The satellite's response to this improper case allowed the base to receive the hash and binary provided by the attacker. This reception allows the base to identify the occurrence of attacks, as the satellite always responds to the base's address, a fact that occurs naturally by the radio communication system. This attack was carried out with multiple passwords close to the password used between the base and the satellite. These tests are important to verify how robust the HMAC is to the variation of a few bits in a data sequence, the methodology being robust to the point of not allowing any code injection from a source that does not have the exact password.

A last test made was the type of denial of service (DDoS) attack, where the attacker emits a code and a hash to each window that are not compatible, this strategy aims to confuse the satellite and prevent the implementation of legitimate codes. Unfortunately, this attack was successful in preventing new code from being injected, however, malicious code was not injected either, guaranteeing the integrity of the system. Thus, the methodology proved to be susceptible to denial of service attacks, which is expected in open communication media such as radiofrequency systems, but ensuring the integrity of the system.

VI. Conclusions

By observing the experiments, we observed the effectiveness of the set of security measures and protocols proposed for a secure system of code injection in ”CubeSat” type satellites, both to avoid the implementation of malformed codes and to defend against attacks. man-in-the-midle, replays and obfuscation that seek to simulate being the base for the purpose of injecting malicious code into the equipment. Although the proposed protocol is not immune to denial-of-service (DDoS) attacks that would prevent the injection of new legitimate code, the protocol was successful in its main purpose of preventing the deployment of malformed code and sources that do not hold the password. authentication to inject code into the satellite, preventing its possible misuse or hijacking.

These results were made possible by the implementation of security measures to obtain predictability of the communication moments, the implementation of an Integrity and Authenticity Assurance Protocol - IAAP with the objective of guaranteeing the integrity of the codes that are provided to the satellite, as well as with the authenticity of the supplier source, and the implementation of a Delivery Guarantee Protocol - DGP with the objective of guaranteeing the efficiency of the existing short communication windows. This strategy of dividing the protocols, leaving the period of the communication windows only for the exchange of messages and the processing and validations for the periods outside the communication windows, proved to be correct for removing the processing of the hash from the communication windows from the period of the communication windows. HMAC, leaving processor time free for other services.

The case study presented in this work is specific to the communication between base and satellite, but the protocols presented can be adapted to several other situations due to the separation between the communication and processing layers. An example could be the use of autonomous drones in areas with difficult communications, where in short periods the opening of channels occurs, allowing data transfers with guaranteed delivery and leaving the heavy processing of authentication and validation of commands for the periods where there is no communication. Other example can be the firmware update on remote IoT devices.

For future works, confidentiality resources can be used in the search for the obscuration of the code to be injected, as well as the commands to be executed by the autonomous element like satellites and drones.

This is made possible by adapting the IAAP to various other systems as required by the case. This code confidentiality can be useful for protecting the code's intellectual properties or simply masking the function of the new code to be injected, which were not in the scope of this work. Another possible measure is the use of a hardware key (hardcoded) in the satellite firmware to obscure the password.

Acknowledgements

The authors are grateful for the support of the ABIN grant 08/2019.

References

[1] P. Yue, J. An, J. Zhang, G. Pan, S. Wang, P. Xiao, and L. Hanzo, "On the security of leo satellite communication systems: Vulnerabilities, countermeasures, and future trends," *arXiv preprint arXiv:2201.03063*, 2022.

[2] I. Bisio, S. Delucchi, F. Lavagetto, and M. Marchese, "Comparison among resource allocation methods with packet loss and power metrics in geostationary satellite scenarios," in *2013 IEEE International Conference on Communications (ICC)*. IEEE, 2013, pp. 4271–4275.

[3] R. Ramadan, "Internet of things (iot) security vulnerabilities: A review," *PLOMS AI*, vol. 2, no. 1, 2022.

[4] P. Rahimi, A. K. Singh, X. Wang, and A. Prakash, "Trends and challenges in ensuring security for low-power and high-performance embedded socs," in *2021 IEEE 14th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. IEEE, 2021, pp. 226–233.

[5] R. P. Lee, K. Markantonakis, and R. N. Akram, "Ensuring secure application execution and platform-specific execution in embedded devices," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 3, pp. 1–21, 2019.

[6] T. Hiscock, O. Savry, and L. Goubin, "Lightweight instruction-level encryption for embedded processors using stream ciphers," *Microprocessors and Microsystems*, vol. 64, pp. 43–52, 2019.

[7] M. A. Philip *et al.*, "A survey on lightweight ciphers for iot devices," in *2017 International Conference on Technological Advancements in Power and Energy (TAP Energy)*. IEEE, 2017, pp. 1–4.

[8] S. Aruna, G. Usha, P. Madhavan, and M. R. Kumar, "Lightweight cryptography algorithms for iot resource-starving devices," *Role of Edge Analytics in Sustainable Smart City Development: Challenges and Solutions*, pp. 139–169, 2020.

[9] M.-S. Hwang, C.-C. Yang, and C.-Y. Shiu, "An authentication scheme for mobile satellite communication systems," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 4, pp. 42–47, 2003.

[10] T. Ferrer, S. Céspedes, and A. Becerra, "Review and evaluation of mac protocols for satellite iot systems using nanosatellites," *Sensors*, vol. 19, no. 8, p. 1947, 2019.

[11] J. Wikman and J. Sjöblom, "Software based memory correction for a miniature satellite in low-earth orbit," 2017.

[12] R. I. Taxonera *et al.*, "Edac software implementation to protect small satellites memory," Master's thesis, Universitat Politècnica de Catalunya, 2019.

[13] M. Sinha and S. Dutta, "Survey on lightweight cryptography algorithm for data privacy in internet of things," in *Proceedings of the Fourth International Conference on Microelectronics, Computing and Communication Systems*. Springer, 2021, pp. 149–157.

[14] J. S. Aidan, H. K. Verma, and L. K. Awasthi, "Comprehensive survey on petya ransomware attack," in *2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS)*. IEEE, 2017, pp. 122–125.