Secured UAV Operations in Robot Operating System Framework With Asymmetric Encryption and Digital Signatures

Christopher Richard Chandra - 18222057
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
Email: 18222057@std.stei.itb.ac.id

Abstract

Unmanned Aerial Vehicles (UAVs) are increasingly deployed in diverse applications from critical infrastructure inspection to disaster response. It demands robust security measures, especially when operating within the Robot Operating System (ROS) framework over wireless networks. This paper addresses the paramount challenge of securing UAV operations against eavesdropping, data tampering, and unauthorized control. We propose a comprehensive security method that integrates asymmetric encryption and digital signatures within ROS topic messages for communication over insecure wireless channels. Asymmetric encryption (e.g., RSA or ECC) is utilized to establish secure communication channels and protect sensitive command, control, and data from unauthorized access. Digital signatures (e.g. DSA or ECDSA) are used to ensure message integrity and authenticity. Prevent malicious actors from injecting false commands or manipulating flight data. This approach mitigates the risks associated with data intercept and command spoofing which are crucial to maintaining operational integrity and safety. We detail the implementation strategy within the ROS architecture by highlighting key security components and their integration with existing ROS communication paradigms. Experimental results demonstrate the feasibility and effectiveness of the proposed cryptographic mechanisms in safeguarding UAV operations. We will evaluating their impact on communication overhead and latency. Also we will confirming their ability to enhance the resilience of UAV systems in adversarial wireless environments.

I. Background

Unmanned Aerial Vehicles (UAVs) emerged from critical military applications where they have revolutionized reconnaissance, intelligence gathering, surveillance, and target acquisition [1]. The highly sensitive nature of these operations necessitated the development of robust, resilient, and secure communication links to prevent unauthorized access, command manipulation, and data exfiltration. Building on this foundation, UAVs have rapidly transitioned into various civilian and commercial sectors, including infrastructure inspection, agriculture, package delivery, and emergency response, inheriting the imperative for equally stringent security measures.

Often, the sophisticated functionalities of military and civilian UAVs are backed by the Robot Operating System (ROS), an open source framework designed to facilitate the development of complex robotic applications [2, 3]. ROS provides a flexible, distributed architecture that allows various components (nodes) of a UAV system to communicate seamlessly over wireless networks and handling everything from flight control to payload data processing.

However, this indispensable reliance on wireless communication inherently exposes UAV operations to significant security vulnerabilities. The open and broadcast nature of wireless channels makes communication susceptible to a variety of adversarial attacks, including eavesdropping, jamming, man-in-the-middle attacks, replay attacks, and spoofing [4]. In military contexts, such compromises could lead to intelligence breaches, mission failure, or even the weaponization of hijacked assets. Similarly, in civilian roles, a security breach could result in critical infrastructure damage, privacy violations, or threats to public safety. While ROS offers a robust development environment, its default communication paradigms often lack built-in strong cryptographic protections, presenting a substantial challenge for secure UAV operations in both sensitive military and increasingly critical civilian domains.

In order to enhance the security of message exchange within the Robot Operating System (ROS), this paper introduces a novel security layer utilizing asymmetric encryption for data confidentiality and digital signatures for robust sender authentication.

A. RSA Digital Signature

The RSA digital signature scheme is based on the RSA public-key cryptosystem. It enables message integrity verification and sender authentication by using a pair of cryptographic keys: a private key for signing and a public key for verification. The signing process involves hashing the original message and

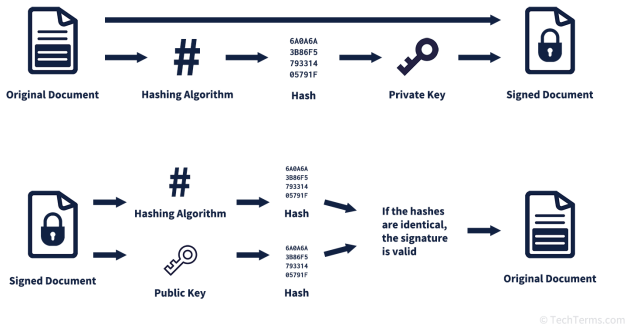

Fig. 1. RSA Digital Signature Visualization (Adapted from [5]).

encrypting the hash with the sender's private key. Upon receipt, the verifier decrypts the signature using the sender's public key and compares it to the hash of the received message. If they match, the message is confirmed to be authentic and unaltered. For each message the hash will be different. Therefore, it's possible to check the originality of the message based on the hash only.

This approach provides non-repudiation as only the sender's private key could have generated the signature. RSA digital signatures are widely used in secure communications, including SSL/TLS, code signing, and blockchain technologies.

B. Robot Operating System (ROS)

The Robot Operating System (ROS) is an open-source middleware framework designed for developing complex robotic systems. It provides a modular architecture in which software components, known as nodes, communicate via message-passing over topics, services, or actions. These interactions typically occur over TCP/IP or UDP. However, by default, ROS does not enforce encryption or authentication on its communication channels.

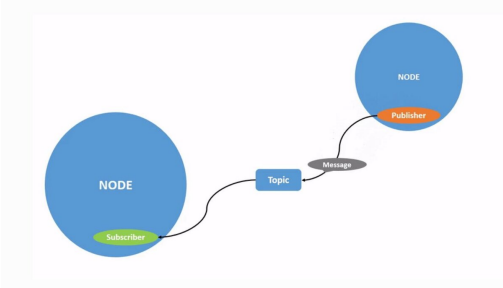

Fig. 2. ROS Communication Visualization (Adapted from [6]).

This lack of built-in cryptographic security poses significant risks when ROS is deployed in networked or wireless environments, particularly in UAV systems. An attacker could potentially intercept, modify, or inject malicious messages, leading to system malfunction or unauthorized control.

While ROS 2 introduces improvements such as the Data Distribution Service (DDS) for communication, it still broadcasts topic data to all nodes within the same network by default. This behavior makes it possible for adversaries on the same network to eavesdrop on or tamper with messages, compromising the integrity and reliability of the system. Integrating digital signature mechanisms into ROS message flows can address these concerns by providing authentication and ensuring data integrity without requiring a complete overhaul of the ROS communication model.

II. Methods

A. System Overview

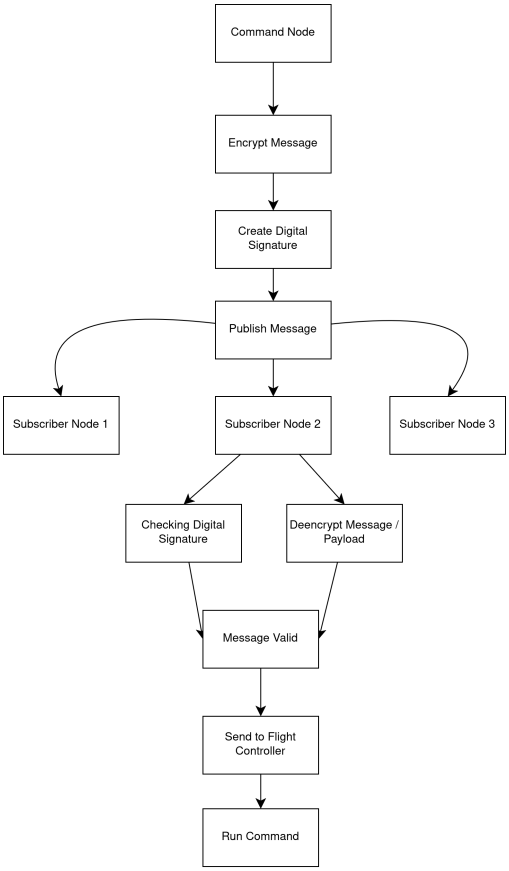

Fig. 3. Proposed System Architecture

The proposed system enhances the security of inter-node communication within the Robot Operating System (ROS) by integrating RSA-based digital signature mechanisms.

The architecture consists of a typical ROS-based UAV setup where nodes exchange sensor data, control commands, and status information over ROS topics. To secure these exchanges, each publisher node is equipped with an RSA key pair. Before publishing a message, the node computes a SHA-256 hash of the message content and encrypts it using its private key to generate a digital signature. The message and its corresponding signature are then bundled into a custom ROS message for encrypted data and then transmitted over the network.

On the receiving end, the subscriber node extracts the received message and signature. It recomputes the hash of the received message and decrypts the signature using the

sender's public key. If the two hashes match, the message is considered authentic and untampered. Otherwise, the message is discarded, and an alert may be generated.

To support dynamic environments, the system introduces a custom ROS service for secure public key distribution. Each participating node is equipped with a pre-shared key pair recognized by the key distribution service. This design enables authenticated and flexible key retrieval while mitigating the risk of unauthorized access.

This cryptographic layer is implemented without modifying the core ROS communication framework. Instead, it is embedded into the application layer using custom message types and Python-based helper functions. This ensures compatibility with existing ROS tools and workflows, while significantly improving communication security. The system is designed to be lightweight and suitable for real-time operations, with minimal latency introduced by the cryptographic operations.

B. Development Environment

TABLE I
SYSTEM CONFIGURATION

Component	Specification
Operating System	Ubuntu 22.04 LTS
CPU	11th Gen Intel i7-11370H (8) @ 4.800GHz
ROS Version	ROS 2 Humble
Architecture	x86_64 (64-bit)
Kernel Version	Linux 6.8.0-59-generic
Development Tools	CMake, colcon, Python 3
Simulation	Gazebo & rviz2
Firmware	PX4 Autopilot

C. Implementation

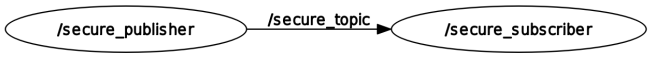

Fig. 4. Simple Secure Node

All source code and setup instructions for reproducing this system are publicly available. [1]

This implementation follows a simple publisher-subscriber communication model within the ROS 2 framework. Each node is pre-configured with the public keys of all other nodes, eliminating the need for dynamic key distribution or exchange protocols. However, this static approach limits scalability, as it only supports a fixed set of known nodes and does not allow for runtime node discovery.

Each transmitted message includes a Base64-encoded encrypted payload and a corresponding Base64-encoded digital signature. The publisher encrypts the message using a symmetric AES key, which is itself encrypted using the subscriber's RSA public key. The digital signature is generated by hashing the original plaintext and signing it with the sender's RSA private key.

[1]GitHub repository: https://github.com/Flame25/ROS2_Secure-Communication

On the subscriber side, the AES key is first recovered using the subscriber's private RSA key. The payload is then decrypted using the AES key and initialization vector (IV). Finally, the system verifies the signature by hashing the decrypted plaintext and comparing it with the received signature using the sender's public key. This process ensures the confidentiality, integrity, and authenticity of messages exchanged between nodes.

```
data: "SGVsbG8gZnJvbSBzZWN1cmUgcHV..."
signature: "PT9qWW3DOu4nUi2hIld+7oeq..."
```

Fig. 5. Base64-encoded signed message echo from the secure publisher node

In order to enhance data security beyond authentication, we implemented a simple RSA combined with AES encryption scheme for each transmitted message. The plaintext is first encrypted using the AES and we encrypt the AES Key using recipient's public key and then Base64-encoded all the messages before being published. Similarly, the digital signature is generated using the sender's private key and Base64-encoded as well. At the subscriber side, both the encrypted data and signature are first decoded from Base64, then the data is decrypted and the signature verified. This layered cryptographic process makes it significantly more difficult for an adversary to tamper with or eavesdrop on the communication. Only the intended recipient, possessing the corresponding private key, can decrypt and interpret the message, ensuring secure communication in distributed ROS 2 environments.

```
data: tLrER8U3M0Kk+Rsns8WW9AG4pw...
signature: qsFwjexIyqxh3rRIwxPN6qiDt...
iv: sHbgFekHalOdbFBr+0QPyg==...
encrypted_key: m+iJKJ2V4ErShKx...
```

Fig. 6. Base64-encoded signed and encrypted message echo from the secure publisher node

Fig. 7. ROS Topic Message Stream

III. EXPERIMENT

A. Attacker Node

In this section, we evaluate the robustness of the proposed secure communication mechanism by introducing a simulated attacker node into the system. This malicious node attempts to impersonate the legitimate publisher by injecting falsified data into the network. The objective is to determine whether the subscriber node can reliably detect and reject messages that are not signed using a valid private key associated with a trusted publisher. By analyzing the subscriber's behavior in the presence of tampered or spoofed messages, we assess the effectiveness of digital signature verification in preventing unauthorized data injection within the ROS 2 environment.

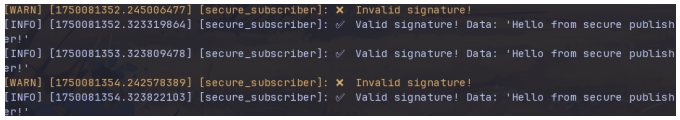

Fig. 8. Invalid data received from attacker node

As shown in Fig.8, most of the messages sent by the attacker node were successfully identified as invalid and subsequently rejected by the subscriber. This is because the attacker uses a different key pair, resulting in digital signatures that fail verification against known public keys. The outcome confirms the integrity of the proposed system, demonstrating that it can effectively distinguish legitimate messages from forged ones. These results highlight the system's potential for deployment in real-time scenarios requiring secure and authenticated communication.

B. Performance

We evaluate the performance impact of the encryption and decryption process by measuring the time taken for both operations using a sample string message: *"Hello from secure publisher"*. This dummy data was selected for its simplicity and consistency during testing. The results, illustrated in Fig. **??**, show the average time required to encrypt the message using the recipient's public key and to decrypt it using the corresponding private key. It is important to emphasize that these results are based on a lightweight, non-representative payload. In real-world scenarios, actual sensor data may be significantly larger or more complex, which could lead to increased processing times. Therefore, careful benchmarking is necessary, especially in time-sensitive or mission-critical applications, where each sensor reading may directly impact system responsiveness and operational outcomes.

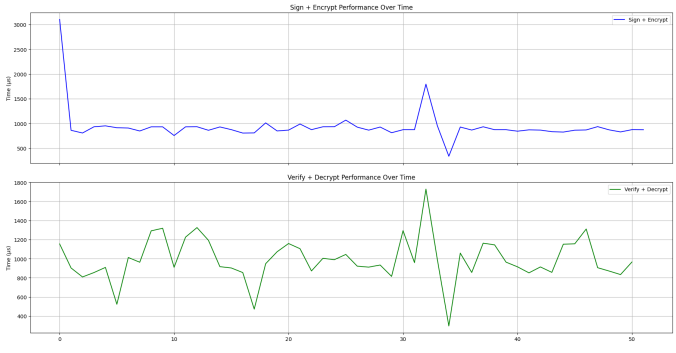

Fig. 9. Performance measurement with simple text

With the aim of testing limit the performance under more heavier payload conditions, we used a longer message string, as shown in Fig. 10.

```
Lorem ipsum dolor sit amet, consectetur adipiscing
elit. Maecenas hendrerit interdum sem ac molestie.
Aliquam volutpat semper congue. Nunc et ante sed eros
viverra aliquet sed nec ex. Nam dapibus aliquet libero.
Duis posuere sodales nunc sed suscipit. Praesent in
elit non felis ultricies dictum. Nunc et viverra odio.
Morbi a felis in elit dapibus tempor nec vitae ipsum.
Sed pretium augue at massa facilisis ultricies. Morbi
hendrerit accumsan blandit. Nunc tincidunt in lectus ac
vulputate. Phasellus ac imperdiet ipsum. Nullam sagittis
odio ac.
```

Fig. 10. Longer message used for encryption and decryption performance testing

This test simulates a extreme data load that heavier than the original data from a sensor or information-rich publisher node. We recorded the encryption and decryption time for this extended message and present the results in Fig. 11. As expected, the processing time increased compared to the shorter test case. This highlights the need for efficient cryptographic handling and careful benchmarking, especially in time-sensitive applications such as robotic control or mission-critical systems.

Fig. 11. Extreme Performance Test

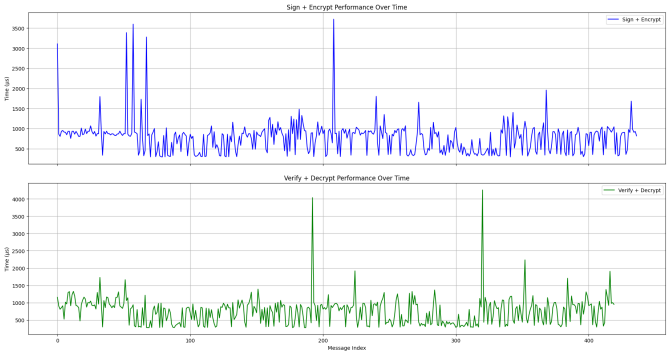

Fig. 12. More Extreme Test (5 Paragraph lorem ipsum)

C. Simulator Test

To evaluate the effectiveness of the proposed encryption and digital signature mechanism in a realistic control scenario, we integrated the system into a Gazebo simulation using PX4 autopilot. The simulation involves a trajectory control interface where PX4 listens to a ROS 2 topic, `/fmu/in/trajectory_setpoint`, for receiving position and velocity setpoints in offboard mode.

A custom ROS 2 bridge node was developed to subscribe to `/secure_trajectory`, a planning-level topic that represents the intended flight trajectory. This node verifies the authenticity and integrity of the message using digital signature verification. Only if the message is deemed valid, the node republishes the data to the actual PX4 topic `/fmu/in/trajectory_setpoint`.

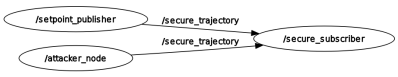

Fig. 13. Node Graph

The primary goal of this bridging approach is to test the robustness of the implemented security mechanisms. It ensures that any data injected from unauthorized or compromised sources—lacking a valid signature will be identified and ignored by the system. This helps prevent malicious commands from influencing the drone's flight behavior, thus preserving operational safety within a simulated but realistic autonomous environment.

The results of the simulator test demonstrate that the secure communication method featuring encrypted payloads and digital signature validation can be integrated into a ROS 2 based PX4 autopilot system without introducing any noticeable delay or performance degradation. The drone was able to respond to trajectory commands at a high frequency, maintaining stable flight even when valid movement commands were published at a high rate. The validation node, responsible for checking digital signatures, successfully filtered out unauthorized or tampered messages, ensuring that only authenticated data influenced the drone's behavior. Furthermore, injected commands from an attacker node with an invalid or mismatched digital signature were consistently ignored, confirming the effectiveness of the security mechanism in rejecting spoofed data while preserving real-time responsiveness.

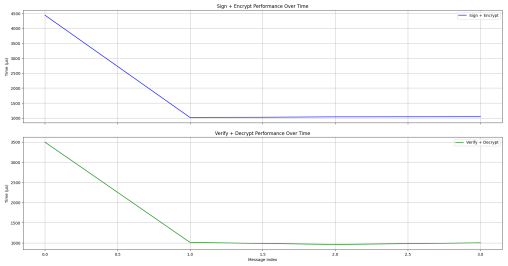

Fig. 14. Performance Measurement using Simulator

IV. Future Work

As future work, implementing a secure service for dynamic key generation and distribution could significantly enhance the scalability, flexibility, and usability of the proposed architecture, while still preserving strong cryptographic guarantees. Such a service would allow new nodes to be onboarded automatically without the need for manual key provisioning, making the system more adaptable for real-world robotic deployments where nodes may be added or removed on the fly. To further strengthen the architecture, this mechanism could be extended and integrated directly into the ROS 2 framework layer. This would ensure that encryption and authentication are enforced by default at the middleware level rather than being handled manually at the application layer. In such a framework, all communication topics could be encrypted by default and data access would be restricted based on roles or permissions defined during node registration. Only nodes with valid cryptographic credentials and explicit authorization would be able to access certain topics or services. This implementation will significantly reducing the attack surface. This role-based security model combined with automated key distribution would provide a robust and scalable foundation for secure multi-agent robotic systems.

V. Conclusion

The results demonstrate that the proposed method can be effectively integrated into a ROS 2 environment without introducing significant performance degradation. Simulator testing confirms that encrypted and authenticated messages do not cause noticeable delays or disruption in control performance, even at high command rates. However, the implementation complexity increases due to the requirement for manual key distribution. Each node must be provisioned with the public keys of all other trusted participants. This makes it challenging to scale or adapt in dynamic systems as adding a new node requires generating and securely distributing keys in advance. Additionally, if the onboard computer is compromised, an attacker could potentially access private keys which posing a significant security risk. While the current implementation performs well in simulation, further validation is needed in real drone deployments to fully assess its resilience and reliability under real-world conditions.

Demonstration Video & Code

A demonstration video showcasing the system in action, including message encryption, digital signature validation, and simulator integration, is available online.[2]

The source code and demonstration scripts are available at marked repository. [3]

References

[1] A. Konert and T. Balcerzak, "Military autonomous drones (uavs) - from fantasy to reality. legal and ethical implications." *Transportation Research Procedia*, vol. 59, pp. 292–299, 2021, 10th International Conference on Air Transport – INAIR 2021, TOWARDS AVIATION REVIVAL. [Online].

[2]Video demo: https://youtu.be/Co018TFREB4

[3]GitHub repository: https://github.com/Flame25/ROS2_Secure-Communication

Available: https://www.sciencedirect.com/science/article/pii/S2352146521008838
[2] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abm6074
[3] "*ROS 2 Documentation* [Online]," Available: https://docs.ros.org/en/humble/index.html, [Accessed: 10 June 2025].
[4] U. Patil, A. Gunasekaran, R. Bobba, and H. Abbas, "Ros2-based simulation framework for cyberphysical security analysis of uavs," 2024.
[5] "*Digital Signature* [Online]," https://techterms.com/definition/digitalsignature, [Accessed: 15 June 2025].
[6] Robotics Unveiled, "*ROS2 Service Server and Client - Python and C++* [Online]," https://www.roboticsunveiled.com/ros2-service-server-and-client-python-and-cpp/, [Accessed: 15 June 2025].
[7] F. J. Aufa, Endroyono, and A. Affandi, "Security system analysis in combination method: Rsa encryption and digital signature algorithm," in *2018 4th International Conference on Science and Technology (ICST)*, 2018, pp. 1–5.
[8] S. U. Jonwal and P. P. Shingare, "Advanced encryption standard (aes) implementation on fpga with hardware in loop," in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, 2017, pp. 64–67.
[9] R. Aissaoui, J.-C. Deneuville, C. Guerber, and A. Pirovano, "Authenticating civil uav communications with post-quantum digital signatures," in *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*, 2023, pp. 1–9.
[10] L. He, Y. Gan, and Y. Yin, "Efficient threshold attribute-based signature scheme for unmanned aerial vehicle (uav) networks," *Electronics*, vol. 14, no. 2, p. 339, Jan. 2025. [Online]. Available: http://dx.doi.org/10.3390/electronics14020339

Pernyataan Keaslian

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 20 Juni 2025

Christopher Richard Chandra
(18222057)