Towards a Modular Attestation Framework for Flexible Data Protection for Drone Systems

Zineeddine Ould Imam
Orange Innovation
Department of Security
Chatillon, France
zineeddine.ouldimam@gmail.com

Marc Lacoste
Orange Innovation
Department of Security
Chatillon, France
marc.lacoste@orange.com

Ghada Arfaoui
Orange Innovation
Department of Security
Rennes, France
ghada.arfaoui@orange.com

***Abstract*—Data protection is a rising concern for systems of drones to guarantee data integrity and privacy through the detection of drone violations in no-fly zones (NFZ). Despite their security guarantees, existing attestation frameworks present limited extensibility. This paper presents a modular attestation framework for drone systems overcoming such a barrier. The framework provides a high degree of flexibility to guarantee both data integrity within and across drones. We also propose a dedicated token-based attestation protocol to support NFZ violation detection. We implemented a proof-of-concept prototype using the OP-TEE and PX4 open environments for trusted execution and drone simulation. Evaluations show the framework guarantees strong data protection with a high level of flexibility while preserving performance and scalability.**

***Index Terms*—Drone systems, data protection, data integrity, privacy, no-fly zone, modular attestation, TEE, PX4, OP-TEE.**

I. Introduction

Unmanned Aerial Vehicles (UAVs), also known as drones, have undergone dramatic increase in availability and use due to their low cost and many promising applications.

Accordingly, drones have become juicy targets and vectors for cyber-attacks [1]. Active attacks may compromise a drone and gain control, corrupt collected data, damage software or services, or disrupt availability. It is thus essential to ensure data and software integrity in all devices of the drone system. Passive attacks may stealthily monitor and collect information for later purposes, either drone-related (e.g., geolocation) or service-related (e.g., photos and videos captured with high-resolution cameras), raising serious privacy concerns. A promising strategy for enforcing privacy compliance is the definition of *no-fly zones* (NFZs) over privacy-sensitive locations, with drone tracking to detect NFZ policy violations.

Remote attestation may be defined as making a claim to a verifier about the properties of a target by supplying supporting evidence [2]. Those security protocols are particularly well suited to the drone data protection context and have been extensively studied [3, 4]. They allow to prove and verify a wide range of properties, from integrity of software, device, or state of dynamic systems of systems such as swarm of drones to presence outside NFZs.

Existing attestation systems usually rely on: (1) *properties* to prove and verify (e.g., integrity of a drone system, privacy-preserving NFZ authorization enforcement, performance), and (2) *features* or means to support such claims (e.g., hardware root of trust, system and network mechanisms). Despite advances [5, 6, 7, 8], attestation systems remain focused on a small part of the big picture, usually designed for some specific properties using a limited set of features. This lack of extensibility prevents supporting additional properties or features for an entire system, or addressing new use cases.

Goals and Contributions. In this paper, we go a step further and propose a modular attestation architecture and framework for drone systems to compose properties and features. The framework provides a high degree of flexibility in the attestation design space, notably to guarantee integrity for both single and multiple drones, support asynchronous and synchronous communications, protect against static and runtime attacks, and allow centralized and distributed strategies of verification. The framework is based on Trusted Execution Environments (TEE). Within each drone, the control flow of critical modules is monitored. Between drones, a spanning tree network is established, and interactions between critical services are tracked.

We also design a novel attestation protocol based on lightweight tokens to address NFZ violations, by detecting drone(s) intrusions and tracking them.

We implement the attestation framework and protocol in the PX4 open flight control software using the OP-TEE open TEE environment. Evaluation results show our modular solution guarantees integrity and privacy, and is highly flexible, supporting several key attestation solutions without losing efficiency, with also good scalability.

This paper provides the following main contributions:

- A modular architecture supporting multiple attestation features for drone systems to ensure data integrity and prove drone presence outside NFZs.
- A NFZ attestation protocol based on lightweight tokens.
- A proof-of-concept using PX4 and OP-TEE.

Outline. We review related work in Section II. We present our vision of a modular approach to attestation in Section III. We describe the design principles of our solution, its architecture and protocol, and implementation in Sections IV, V and VI. We report evaluation results in Section VII and discuss security in Section VIII. We conclude in Section IX.

II. Related work

Remote attestation is a powerful security service to prove data integrity and to protect privacy. A prover can send a status report of the current configuration of a device to a trusted party to verify state trustworthiness to detect tampering by malware. Many classes of protocols have been explored [3, 4].

Architecture. *Software-based attestation* provides minimal security guarantees without secure hardware. It is a low-cost and scalable approach for resource-constrained devices. At the other end, *hardware-based attestation* leverages advanced security hardware such as Trusted Platform Module (TPM) or TEE [9, 10] to provide strong protection of execution of critical components against malware. *Hybrid attestation* minimizes hardware requirements (e.g., simple read-only memory, memory protection unit) to ensure secure, safe and uninterrupted execution of the attestation protocol.

Communication. In *synchronous attestation* [8, 11], devices have to wait for the attestation report of other devices to perform further operations. In *asynchronous attestation* [5], devices do not need to know one another or to participate in the attestation at the same time. Such schemes usually adopt a highly efficient and stable publish/subscribe mechanism.

Network topology. *Static attestation* [8, 11] establishes a static network between devices during attestation. It relies on a spanning tree protocol: each device attests its children and sends back the attestation result to its parent. In *dynamic attestation* [12], the network between devices is variable. Attestation is based on consensus: devices share knowledge and agree on a common attestation report.

Verification. In *centralized attestation* [5], an external verifier requests attestation from one device which aggregates attestation reports from other devices and checks integrity of the entire system. In *distributed attestation* [7, 13], devices can be both provers and verifiers and validate integrity of one another.

Run-time attacks. Most protocols only consider static flows. *Control flow attestation* [7, 14] tracks dynamically control and data flows between modules to guarantee data integrity at use within and across drones and mitigate run-time attacks.

Attestation systems. Several systems have key features in the design space related to our approach [5, 6, 7, 8]. DIAT [7] proposes data-flow and control-flow mechanisms for run-time data integrity, with strong modularity to increase attestation performance, and distributed verification. Communications remain mostly synchronous unlike our solution. SARA [5] provides asynchronous centralised attestation based on the publish/subscribe paradigm, with enhanced security, tracing device interactions using vector clocks. Compared to our solution, it lacks modularity and does not mitigate run-time attacks. SEDA [8] is a collective attestation protocol based on a spanning tree using a single verifier. It offers some extensibility supporting multiple concurrent protocols, but does not address run-time threats either. AliDrone [6] is a centralized attestation system for single drones focused on privacy compliance with respect to NFZs. It shows interesting modularity but does not address drone swarms, unlike our framework.

III. Modular Attestation

The design space for attestation protocols for drone systems has been widening [3, 4]. Two main dimensions may be distinguished. First, a set of *properties* to prove and verify. Properties may be related to the *system/architecture* (e.g., single/multiple-drone applications, synchronous/asynchronous networks, static/dynamic swarms), *security and privacy* (e.g., state integrity, adversary model: run-time/side-channel/physical attacks), *performance* or *scalability* (e.g., drone network area, number of nodes, connectivity, topology). Second, a set of *features* (e.g., TEE support, spanning tree, publish-subscribe, control flow monitoring). Combining these two dimensions, a wire range of attestation *systems* [5, 6, 7, 8] have been instantiated.

Unfortunately, existing attestation systems only support a limited number of features and guarantee a small subset of such properties which may not match the expected guarantees for the considered drone system. In terms of design, their extensibility is generally limited. This prevents supporting more features or further properties to meet given system requirements, both functional and non-functional.

Component-based design has proven its many benefits to design adaptable and extensible systems by separating concerns, and composing on-demand software elements to match provided and required guarantees [15].

A modular *attestation framework* enables to build an attestation service from individual features, components (e.g., existing attestation systems such as [7] or [8]) and properties. The required properties are matched with the provided properties derived from the features of legacy attestation components to build a new attestation service by selective composition of features to fulfil the expected guarantees for the drone system. The overall approach is summarized in Figure 1.

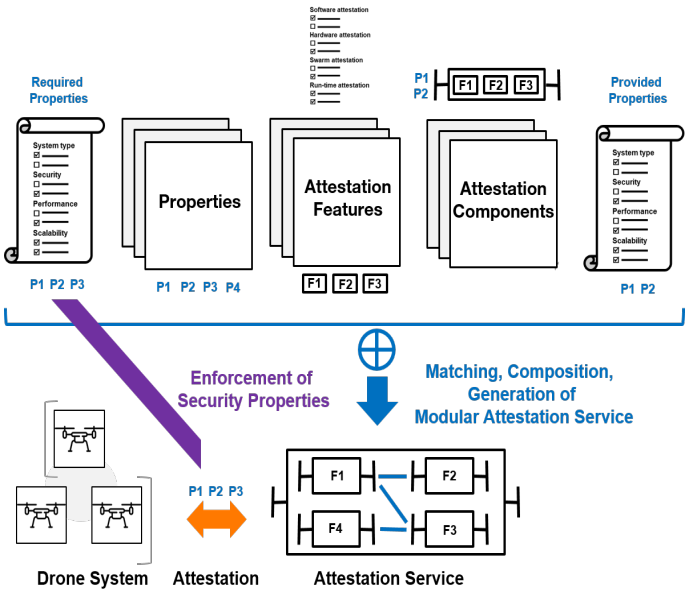

Fig. 1: Modular Attestation

IV. System Model

A. Requirements

We consider a drone system including a single drone or a fleet of collaborating drones. To protect data and detect NFZ violation in multiple settings, the attestation architecture and framework should meet the following requirements:

- *Security:* The attestation scheme must ensure *integrity* of data in the drone system **[S1]**. It must also guarantee *privacy* of personal data if the drone system enters NFZs (e.g., airports) **[S2]**. More generally, it should protect data at rest, in transit, or in use within or across drones.
- *Flexibility:* The attestation architecture must be flexible to support multiple properties such as single and multiple drones **[F1]**, synchronous and asynchronous communications **[F2]**, run-time and non run-time attacks **[F3]**, and centralized and distributed verification strategies **[F4]**.
- *Performance:* The attestation framework must induce low run-time overheads on drones **[P1]**, and must scale to large drone networks **[P2]**.

Therefore, we adopt the following:

- *Modular attestation:* The architecture is built from small interactive modules that may be composed on-demand.
- *Hybrid attestation:* Critical modules are protected by a TEE that offers secure boot and storage, run-time isolation, and denies external entities to access or change programs.
- *Lightweight design*: The attestation time and code size to be attested are minimized. Only modules that process critical data are attested. Lightweight attestation data are also stored for each drone.

B. Adversary Model

We consider a rogue adversary aiming to corrupt correct drone behaviors or violate NFZ airspace without being detected. The adversary can compromise device data and manipulate messages between drones, drones and verifier, or drones and NFZ sensors. The adversary can manipulate code, and launch run-time attacks. Denial of service (DoS) and hardware attacks are considered out of scope.

We assume each drone includes hardware support for trusted execution, e.g., ARM TrustZone, Intel Software Guard Extensions (SGX) enclaves. Side-channel attacks on the TEE are not considered. Sensor data injection attacks are disregarded (i.e., drone sensors are assumed to be reliable, but sensor drivers can be compromised).

C. Design Principles

Attestation design has been widely explored through multiple specialized approaches, architectures, and systems with supporting features in software configurations. Tables I and II show some key solutions for data integrity [5, 7, 8] and drone detection in NFZ [6], and how they address the flexibility and security requirements and through which features.

Overall, solutions fail to support both single/multiple drones, synchronous/asynchronous communication, and centralized/distributed verification strategies. Run-time attacks are seldom supported. Despite strong security through the TEE, they generally lack modularity and extensibility.

TABLE I: Attestation design space: properties vs. systems

Properties	DIAT [7]	SARA [5]	SEDA [8]	AliDrone [6]
Single/multiple drones **[F1]**	multiple	multiple	multiple	single
Sync./async. communication **[F2]**	sync.	async.	sync.	sync.
Run-time attacks **[F3]**	yes	no	no	no
Central/distributed verification **[F4]**	dist.	cent.	cent.	cent.

To overcome such limitations, we introduce a modular and flexible attestation architecture and framework that composes the required features from configurations of the attestation design space.

Our system is based on the following design principles:

- *Trusted collaboration:* To guarantee efficienctly trustworthiness of the drone system, we adopt collective attestation based on a *spanning tree* protocol. This allows to communicate only with trusted drones in a swarm and to establish parent-child relationships. Single drone may also be attested by considering trees of size 1.
- *Asynchrony*: To achieve stable communication, intra-drone and inter-drone modules interact asynchronously using the *publish/subscribe paradigm*. Drone modules that complete local attestation may resume normal operations, while attestation progresses elsewhere.
- *Control-flow attestation:* To cover run-time attacks, the system monitors control and data flows to enable the verifier to detect attacks that do not match program control-flow graphs (e.g., return-oriented programming) or generate valid but unexpected program executions (e.g., non-control data attacks).
- *Centralized verification*: We assume a single verifier that checks the integrity of the drone system and its presence outside NFZs by requesting attestation from the drone, or *root drone* in the case of a drone swarm.

V. System Design

A. Attestation Protocol

System Overview. We consider a drone system including a single drone or multiple drones organized as a spanning tree network. Nodes communicate asynchronously with one another. A Ground Control Station (GCS) represents the drone operator initializing drones with their software configuration and certificates (see Figure 2).

As drones fly to their destination, several NFZs may be encountered along the path. Each NFZ includes a set of sensors to detect drone approach and to determine if drones are authorized to enter the NFZ. A trusted third-party, the *verifier*, requests attestation from the root drone to check both integrity of the drone fleet and NFZ violation.

Attestation Protocol. The protocol is composed of several micro-protocols (see Table III) arranged in several phases.

TABLE II: Attestation design space: comparison of attestation systems with respect to features related to system type, communication, verification, security, and extensibility. We note ● (resp. ○) when a feature (resp. its opposite) is supported, ◐ when both cases are covered, ⧗ when the feature may be supported through extensions, and ⊙ when it is only discussed.

Attestation System	System Type			Communication			Verification		Security					Extensibility	
	Multiple drones	Singular attestation	Swarm attestation	Synchronous comm.	Spanning tree	Publish/subscribe	Single verifier	Distributed att.	TEE support	Historical evidence	Run-time attacks	Control-flow att.	Data-flow monitoring	Modularity	Multiple protocols
DIAT [7]	●		●	●				●	⊙		●	●	●	●	
SARA [5]	●		●	○		●	●			●					
SEDA [8]	●		●	●	●		●								●
AliDrone [6]	○	●		●			●		●					●	
Framework	◐	●	●	◐	●	●	●	●	●	⧗	●	●	●	●	●

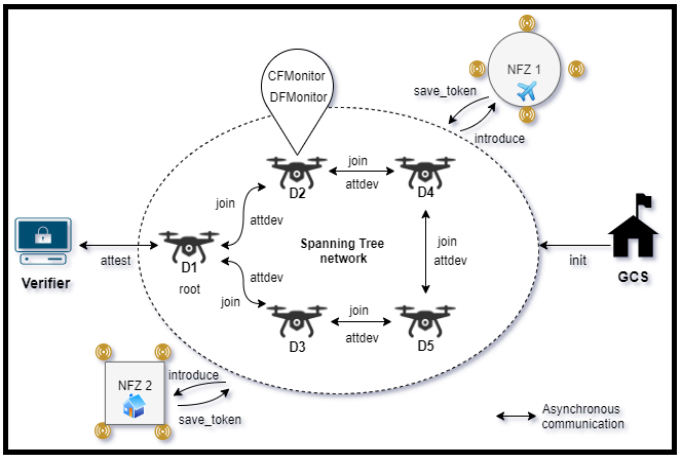

Fig. 2: System Overview

TABLE III: Overview of micro-protocols

Micro-Protocol	Description
Swarm building	
INIT	initializes drones with software/certificates
JOIN	includes a drone in the swarm
ATTDEV	builds spanning tree network for the swarm
Token management	
INTRODUCE	presents drone identity
SAVE_TOKEN	sends and stores token in TEE
Attestation	
ATTEST	requests attestation to the drone system
INTERACT	starts control flow attestation

1) Preparation Phase. The GCS initialises each drone with: its software configuration, a code certificate signed by the GCS that guarantees that the software configuration is valid, a pair of public/private drone signature keys, and a key certificate signed by the GCS that guarantees that the public key belongs to the drone. Certificates and keys are stored inside the TEE. Then, each drone learns the software configuration of its neighbors by receiving their code certificates. If certificate verification succeeds, the drone stores the software configuration and an attestation key shared between the drones, and each neighbor is established using a JOIN micro-protocol (e.g., [8]). A spanning tree is built over the entire drone fleet (e.g., [8]) using a global session identifier. Whenever a drone receives a new session identifier from a neighbor, it accepts this neighbor as its parent.

2) Attestation Phase. When the verifier requests data integrity attestation from the root drone, an INTERACT micro-protocol (e.g., [7]) is initiated. The drone generates critical data (e.g., GPS coordinates) from its sensors by running the necessary software. Meanwhile, trusted components *Data Flow Monitor* (*DFMonitor*) and *Control Flow Monitor* (*CFMonitor*) respectively return the identifiers of critical modules, and record the control flow of each critical module as a *multi-set hash* (*MSH*) of events.

Each drone maintains a *vector-clock* of size equal to the number of drones (e.g., [5]). This allows accurate tracing of event occurrences and detection of services affected by previous interactions. The root drone forwards the attestation request and the vector-clock to its children using an ATTDEV micro-protocol (e.g., [8]). Recursively, each drone signs the critical data, its vector-clock, and its MSH by its private key, and sends them to its parent. The root drone aggregates reports, signs them, and sends the attestation result to the verifier.

3) Verification Phase. When the verifier receives the attestation results, it checks the signature of each drone, verifies data integrity of the fleet by comparing the critical data with the execution path in each MSH. If an attack is detected, the verifier identifies the malicious service and the services affected by the attack. It can take mitigation actions such as disabling the harmed drone from the fleet or correcting falsified data.

4) NFZ Attestation. We introduce a token-based protocol to guarantee privacy and prevent drones from entering NFZs. We assume that NFZs are surrounded by sensors that store the certificates of legitimate drones and their public keys. Sensors can also detect the approach of a drone. Drones are configured with NFZ certificates and public keys.

Upon detection by NFZ sensors, a drone is requested for its identity signed by the NFZ private key. If NFZ authentication succeeds, the drone returns its identity stored in the TEE, signed by its private key. Based on the NFZ access policy, the drone is returned a signed green token (resp. red token) installed in the TEE if the drone is authorized to proceed (resp. to record the NFZ violation). Mitigation is possible, e.g., by triggering alarm systems, or by sending commands to halt the drone [16, 17].

When the verifier asks the drone system for NFZ attestation, the root drone collects the saved tokens from all drones, and returns them signed by its private key. The verifier then looks for red tokens to determine any NFZ violation.

This protocol remains lightweight, without need for run-time generation of GPS coordinates [6], nor saving NFZ topology. It also supports complex geometric zones, and can be extended for mitigation.

B. Architecture

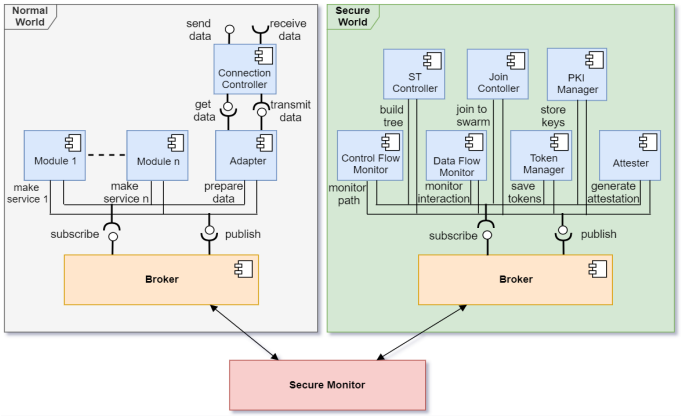

Fig. 3: Modular Attestation Architecture

We propose a flexible and modular ARM TrustZone-based architecture that supports multiple attestation features (see Figure 3). We decompose the attestation software into small interactive modules. Critical modules run in Secure World mode protected by the TEE. Non-critical modules run in Normal World mode in the operating system. The two worlds communicate only via the Secure Monitor. Modules interact asynchronously using a publish/subscribe *Broker* mechanism.

In Normal World: the *Connection Controller* sends and receives data to/from other drones, the verifier, and NFZ sensors; the *Adapter* prepares data to be processed or to be sent externally; and the *N-modules* are critical modules that generate sensitive data (e.g., GPS coordinates, sensor outputs).

In Secure World: the *Attester* processes attestation requests from the verifier, initializing the attestation operation and returning the result; the *Token Manager* stores NFZ tokens and generates token reports; the *Data Flow Monitor* traces the flow of data and returns the identifiers of critical modules; the *Control Flow Monitor* monitors the execution path and records the control flow of critical modules; the *ST Controller* establishes a spanning tree (ST) network between drones; the *Join Controller* introduces drones into the fleet; and the *PKI Manager* stores certificates and keys.

VI. Implementation

We implemented our system on PX4 [18], an open source autopilot, using Software In the Loop (SITL) simulation. We used the QGroundControl open software for full flight control and NFZ activation, and Gazebo to fly multiple drones in a simulated world. We used OpenSSL to generate certificates for NFZs and drones.

PX4 is an autopilot software particularly well suited to implement our system thanks to its modular architecture. PX4 modules communicate asynchronously via a publish/subscribe mechanism. However, PX4 does not support TEE. Therefore, we implemented PX4 communication using OP-TEE [19], an open TEE based on ARM TrustZone (see Figure 4).

We modified OP-TEE configurations to support SSH communication from the host machine to the virtual machine that runs the Secure World. PX4 communicates with OP-TEE via SSH using the libssh library. Thus, non-critical PX4 modules can run in Normal World and critical modules of the attestation framework can run in Secure World.

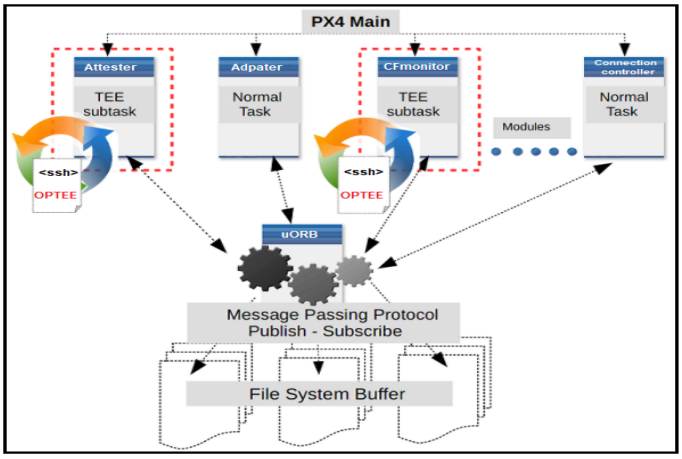

Fig. 4: Implementation Architecture

VII. Evaluation

We present evaluation results in terms of flexibility, performance, and scalability.

Flexibility. Thanks to its modular design, our architecture supports multiple protocols from the attestation design space – notably DIAT, SEDA, SARA, and AliDrone – simply by selecting the modules implementing the protocol features.

This enables to meet the **[F2-F4]** flexibility requirements for synchronous/asynchronous communication, run-time/non run-time threat mitigation, and centralized/distributed verification.

Architectural flexibility also enables to verify different security properties such as data integrity and presence outside NFZ (**[S1, S2]** requirements), and to attest both single drones and fleets of drones (**[F1]** requirement).

For instance, Figure 5 shows NFZ presence detection for a fleet of drones using PX4/QGroundControl based on the AliDrone scenario.

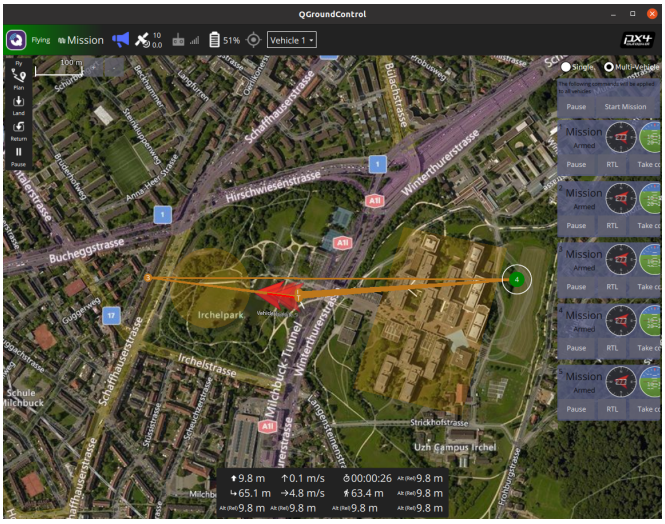

Fig. 5: Flexibility – NFZ violation detection (fleet of drones)

TABLE IV: Latency breakdown of NFZ attestation protocol. For each micro-protocol is given the latency (in s) and its contribution (in percentage) to the overall attestation time.

Micro-Protocol	1-drone		100-drones	
	s	%	s	%
Swarm building				
INIT	2.43	36.49	8.35	35.95
JOIN	-	-	3.77	16.24
ATTDEV	-	-	3.76	16.19
Token management				
INTRODUCE	2.52	37.84	4.16	17.92
SAVE_TOKEN	1.71	25.67	3.18	13.70
Total	6.66	100	23.22	100

Performance. We measure the performance overhead of the NFZ attestation protocol in terms of latency. We evaluated our system running simulations on a HP Zbook Studio G4, with a i7-7820HQ CPU (2.90 GHz × 8), 15.5 GB RAM, running Ubuntu 20.04.2 LTS. To assess the performance impact of each micro-protocol, we measured the latency breakdown for both single and multiple drones. PX4 can support up to 255 drones simultaneously. For the sake of experiment, we measured latency for a drone system size scaling to 100 drones.

Results are shown in Table IV. Our main findings are the following and highlight the efficiency of our protocol:

- The TEE overhead may be considered as negligible: latency between TEE-protected services within a drone is lower than 1.6 ms, much smaller than the overall attestation time (6.66 s [1 drone], 23.22 s [100 drones]).
- The greater part of the attestation time is spent in the INIT micro-protocol (36.49% [1 drone], 35.95% [100 drones]), run only once, and not affecting the later attestation steps.
- The cost of network mechanisms remains low (16.19% [spanning tree], 16.24% [building the swarm]). Those mechanisms are the backbone to establish the swarm structure and topology for network communication.
- The token scheme is fairly lightweight and scalable (31.62% of overall latency [100 drones], with relative overhead of only 1.73 compared to the single drone case).

Scalability. Figures 6 and 7 show the evolution of attestation and verification latencies for NFZ with respect to the number of drones and the number of neighbors per drone in the swarm. Latency increases linearly for both metrics, tending to show our protocol scales well for large and dense drone networks.

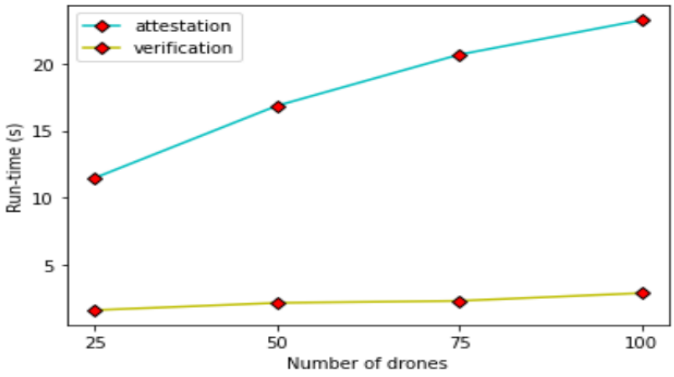

Fig. 6: Scalability: Attestation latency vs. number of drones

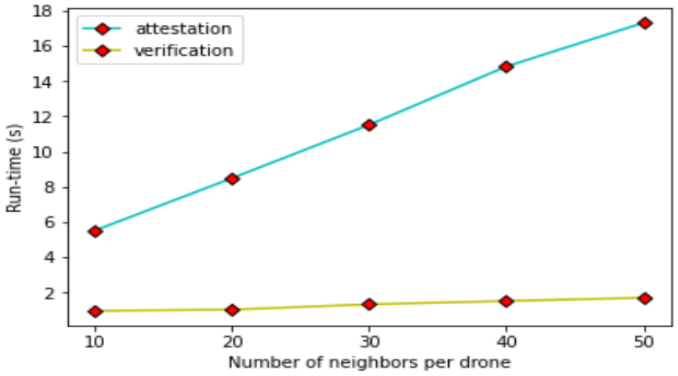

Fig. 7: Scalability: Attestation latency vs. number of neighbors per drone in drone swarm

VIII. Security Analysis

Our system achieves a high level of data protection. We shortly discuss how it meets the identified security requirements **[S1, S2]**. An adversary aiming to corrupt data integrity (resp. NFZ privacy) may target the first three (resp. last) following attack vectors:

1) Sensitive data in the drone. The adversary can target critical or non-critical components. Critical components are implemented in Secure World inside the TEE that guarantees isolation and is assumed secure in the considered threat model.

Non-critical components are implemented in Normal World and may generate sensitive data. Two broad classes of attacks should be considered for such components:

- Attacks on *module code*: Module tampering before loading is detected by the code certificate generated for each module when loaded in the drone. At run-time, code is protected by the isolation guarantees of the architecture.
- Attacks on *module data*: Different types of run-time threats (e.g., *control-data* and *non-control data*) can be detected by the verifier using control-flow attestation.

2) Communication within the swarm. The adversary can insert a malicious drone in the swarm to corrupt data communications. Such behaviour is protected by code certificates when running the JOIN protocol. Connection to the swarm is not possible without presenting a valid code certificate.

3) Communication between drone and verifier. Data is cryptographically protected when attestation is transferred to the verifier, i.e., signed by a key protected in the drone TEE. The adversary cannot alter the data nor create a valid signature for the falsified data without being detected.

4) Communication between drone and NFZ sensors. When an NFZ sensor detects a drone, mutual authentication is required, i.e., the sensors must sign the identity request by its private key, and the drone provides its identifier signed by its TEE-protected private key. Hence, the adversary cannot spoof the drone identity nor send forged tokens.

IX. Conclusion

In this work, we present a modular attestation framework for data protection for drone systems. We design a modular architecture that support multiple features from the attestation design space. We propose a novel protocol to protect privacy and prevent drones from entering NFZs. We demonstrate our system functionality using PX4 and OP-TEE environments. Results show efficiency and scalability of the attestation for large swarm of drones.

As next steps, we intend to implement control and data flow components to evaluate the overall framework performance. We will test the system by flying real drones in an outdoor environment containing NFZs. We also plan to experiment our framework with swarms of real and virtual drones.

References

[1] Yueyan Zhi et al. "Security and Privacy Issues of UAV: A Survey". In: *Mobile Netw. Appl.* 25 (2020), pp. 95–101.

[2] George Coker et al. "Principles of Remote Attestation". In: *Int. J. Inf. Secur.* 10.2 (2011), pp. 63–81.

[3] Alexander Sprogø Banks, Marek Kisiel, and Philip Korsholm. "Remote Attestation: A Literature Review". In: *ArXiv Preprint ArXiv:2105.02466* (2021).

[4] Ioannis Sfyrakis and Thomas Gross. "A Survey on Hardware Approaches for Remote Attestation in Network Infrastructures". In: *ArXiv Preprint ArXiv:2005.12453* (2020).

[5] Edlira Dushku et al. "SARA: Secure Asynchronous Remote Attestation for IoT Systems". In: *IEEE Trans. Inf. Forensics Secur.* 15 (2020), pp. 3123–3136.

[6] Tianyuan Liu et al. "Alidrone: Enabling Trustworthy Proof-of-Alibi for Commercial Drone Compliance". In: *ICDCS*. 2018.

[7] Tigist Abera et al. "DIAT: Data Integrity Attestation for Resilient Collaboration of Autonomous Systems". In: *NDSS*. 2019.

[8] Nadarajah Asokan et al. "Seda: Scalable Embedded Device Attestation". In: *CCS*. 2015.

[9] Ziwang Wang, Yi Zhuang, and Zujia Yan. "TZ-MRAS: A Remote Attestation Scheme for the Mobile Terminal Based on ARM TrustZone". In: *Secur. Commun. Netw.* 2020 (2020).

[10] Guoxing Chen, Yinqian Zhang, and Ten-Hwang Lai. "OPERA: Open Remote Attestation for Intel's Secure Enclaves". In: *CCS*. 2019.

[11] Xavier Carpent et al. "Lightweight Swarm Attestation: A Tale of Two LISA-s". In: *ASIA CCS*. 2017.

[12] Florian Konhäuser, Niklas Büscher, and Stefan Katzenbeisser. "SALAD: Secure and Lightweight Attestation of Highly Dynamic and Disruptive Networks". In: *ASIA CCS*. 2018.

[13] Florian Konhäuser, Niklas Büscher, and Stefan Katzenbeisser. "A Practical Attestation Protocol for Autonomous Embedded Systems". In: *Euro S&P*. 2019.

[14] Tigist Abera et al. "C-FLAT: Control-Flow Attestation for Embedded Systems Software". In: *CCS*. 2016.

[15] Clemens Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, 2002.

[16] Rakesh Rajan Beck, Abhishek Vijeev, and Vinod Ganapathy. "Privaros: A Framework for Privacy-Compliant Delivery Drones". In: *CCS*. 2020.

[17] Abhishek Vijeev, Vinod Ganapathy, and Chiranjib Bhattacharyya. "Regulating Drones in Restricted Spaces". In: *HotMobile*. 2019.

[18] PX4. URL: docs.px4.io/master/en/.

[19] Open Portable Trusted Execution Environment. URL: www.op-tee.org.