Enhancing Network-edge Connectivity and Computation Security in Drone Video Analytics

Alicia Esquivel Morel*, Deniz Kavzak Ufuktepe*, Robert Ignatowicz†, Alexander Riddle*, Chengyi Qu*, Prasad Calyam* and Kannappan Palaniappan*
*Department of EECS, University of Missouri - Columbia
†Department of CSE, Stony Brook University
Email: *{ace6qv, dkyb5, cqy78, amrwf8}@mail.missouri.edu, †rignatowicz@cs.stonybrook.edu, *{calyamp, pal}@missouri.edu

***Abstract*—Unmanned Aerial Vehicle (UAV) systems with high-resolution video cameras are used for many operations such as aerial imaging, search and rescue, and precision agriculture. Multi-drone systems operating in Flying Ad Hoc Networks (FANETS) are inherently insecure and require efficient security schemes to defend against cyber-attacks such as e.g., Man-in-the-middle, Replay and Denial of Service attacks. In this paper, we propose a cloud-based, end-to-end security framework viz., "DroneNet-Sec" that provides secure network-edge connectivity, and computation security for drone video analytics to defend against common attack vectors in UAV systems. The DroneNet-Sec features a dynamic security scheme that uses machine learning to detect anomaly events and adopts countermeasures for computation security of containerized video analytics tasks. The security scheme comprises of a custom secure packet designed with MAVLink protocol for ensuring data privacy and integrity, without high degradation of the performance in a real-time FANET deployment. We evaluate DroneNet-Sec in a hybrid testbed that synergies simulation and emulation via an open-source network simulator (NS-3) and a research platform for mobile wireless networks (POWDER). Our performance evaluation experiments in our holistic hybrid-testbed show that DroneNet-Sec successfully detects learned anomaly events and effectively protects containerized tasks execution as well as communication in drones video analytics in a light-weight manner.**

***Index Terms*—UAV systems, security layer, secure hybrid testbed, ns-3, Powder**

I. Introduction

Unmanned Aerial Vehicle (UAV) systems with video analytics tasks are an emerging need in applications such as smart agriculture, disaster scene scenarios, surveillance, military applications, and package delivery. Multi-drone systems with such tasks have significant challenges in terms of security and privacy. For instance, they are susceptible to not only malicious attacks that cause disastrous misuses, but also can be impacted by accidental adversaries who could cause broken cameras or serious drone misconfigurations [1], [2].

Malicious communications, jams or spoofs in Ground Control Station (GCS) signals, or Denial of Service (DoS) attacks that disrupt the drones operations (e.g., cause data integrity or loss of privacy issues) are reported frequently and have raised concerns [3]. Hence, building UAV systems increasingly involves overcoming challenges in terms of security and privacy of data affected by malicious attacks while performing computation offloading (CO) and control networking (CoNet) [4]–[6] as shown in Fig. 1. Moreover, drone operations in FANETS are inherently insecure and need to be integrated with effective security schemes to be capable of defending against cyber-attacks [7] during continuous video stream transmission from drones to GCS, and processing with edge resources.

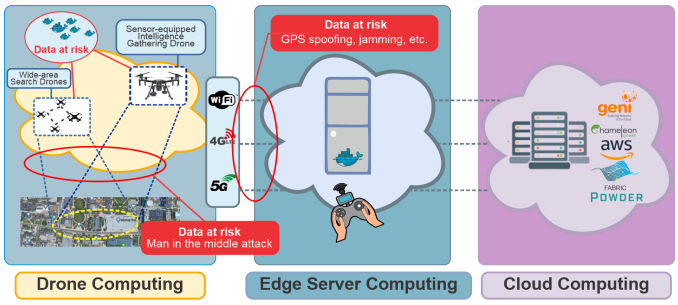

Fig. 1. GCS communication with UAVs and integration with a cloud/edge network infrastructure; Cyber-attacks can target UAV flights, GCS communication, or containerized data processing tasks.

In this paper, we address the above security challenges and develop the "DroneNet-Sec", which is an cloud-based, end-to-end security framework addressing common attacks (e.g., Man-in-the-middle (MITM), Replay, DoS attacks) that have significant impact on UAV systems. The DroneNet-Sec features a security scheme that comprises of a custom secure packet designed with MAVLink protocol. The packets use nonce, encryption and message authentication code in drone and GCS communications to provide secure network-edge connectivity to defend against data privacy and integrity attacks, without high degradation of performance in a real-time FANET deployment. Our DroneNet-Sec uses an intelligent attack detection method based on machine learning together with a fully containerized design to improve computation security. The components of the DroneNet-Sec are compatible with architectural designs that are suitable for cloud applications that use micro-services in management of UAV systems. In addition, they are easily deployable and scalable, providing extensibility to include a flexible number of tasks and video analytics processes.

978-1-7281-8243-8/20/$31.00 ©2020 IEEE

The design of DroneNet-Sec uses the *Zero Trust Architecture* (ZTA) paradigm that is based upon "never Trust, always verify" principles. The ZTA is commonly used as part of security mechanisms which assume that all network communications are hostile. In addition, ZTA creates the necessity of constant monitoring of the system to check every transmission as a suspicious activity, taking possible precautions to ensure the reliability of the communication in between the parties with the assumption that there is always an intruder that can intervene.

We validate our DroneNet-Sec approach using a hybrid testbed that can support scalable simulation and emulation experiments for a variety of drone video analytics purposes. We specifically use this testbed to verify whether, adding a security layer would degrade the performance of the control networking and the computation offloading of given set of data processing pipelines involving UAV systems. Our experiments show that the DroneNet-Sec approach allows for a light-weight security mechanism and also ensures the computation security, data privacy and integrity of the containerized real drone-video analytics tasks, while minimizing any possible degradation in the processing performance.

The main contributions of this paper can be summarized as follows:

A novel security framework for real-time UAV systems with containerized video analytics tasks, combining two schemes:

- A secure messaging scheme with custom packet design enforcing data privacy and data integrity, securing the messaging in between drone-to-drone and drone-to-edge servers providing a secure protocol over MAVLink [8].
- A machine learning based anomaly detection, control and countermeasure scheme with constant monitoring for computation security, which can be updated dynamically to improve the security of a UAV system further.

A hybrid testbed to be used in experimentation of schemes to improve network-edge connectivity and computation security for multi-UAV systems. The hybrid testbed contains:

- A FANET network simulation on ns-3 [9] for scalability of the FANET nodes, a variety of mobility models and network transmission protocols.
- An emulation of edge resources on allocated virtual machines on real nodes on the Powder infrastructure [10] for scalability of the resources that is needed for offloading of computationally intensive tasks to edge resources.

The remainder of this paper is organized as follows: In Section II, we present related work. Section III details our DroneNet-Sec framework solution and implementation. In Section IV, we describe performance evaluation experiments and findings. Finally, Section V concludes the paper.

II. Related Work

A. UAV Systems and Security Issues

UAV systems can be considered as a representative application of embedded systems in Internet of Things (IoT) scenarios. Several technological ecosystems can converge in a multi-drone video analytics platform involving e.g., different communication technologies such as WiFi, 4G/5G cellular, and leveraging edge/fog and cloud/edge computing resources [11].

In multi-UAV systems featuring highly resource intensive tasks involving data collection and video analytics (i.e. video stabilization, motion detection, and tracking), the resources of on-board components are easily overwhelmed. In order to process high volumes of images with commensurate computing, memory and storage capabilities, some of those tasks can be offloaded to resources on edge and cloud servers by using different policies to decide 'when and how' to perform offloading [5], [12]. In such systems where offloading policies are applied, the vulnerabilities of UAV systems are higher since there is a direct connection to edge servers that constantly send and receive video-stream processes over the network [13]. Moreover, the limited resources on UAVs brings additional constraints on any security scheme that can be applied to the common protocols in the UAV systems.

The targets of these attacks can be classified into three main categories: (i) the availability of the system, (ii) data integrity and (iii) confidentiality in the data communications and storage [7]. Fig. 2 provides an overview of three malicious cyber-attack categories and the selected attack types under those categories that compromise the communication and data safety in UAV systems.

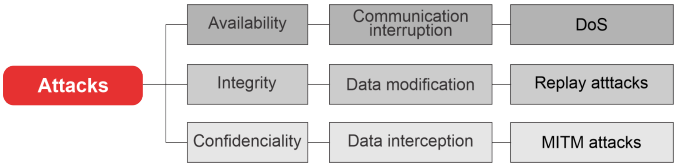

Fig. 2. Attacks on UAV system communication links and data processing servers in edge computing resources.

In availability attacks, the attacker can jeopardize the availability through control and disruption of the communication among UAVs and GCS nodes. Integrity attacks can be launched in two ways: (i) the attacker modifies data, or (ii) the attacker fabricates malicious data. In confidentiality attacks, the attacker gets unauthorized access to classified data, intercepts signals, and sends data to other unauthorized entities via eavesdropping on communication links.

In this work, the first focus is on communication interruption, targeting availability of the system, in particular Denial of Service (DoS) attacks [14]. The second focus is on data modification targeting the integrity of the data, specifically Replay attacks [15]. The third focus is on data interception, targeting the data confidentiality, in which Man in the Middle (MITM) [16] attacks might occur.

B. Securing UAV Systems

There are many recent works that proposed different solutions with variety of techniques for UAV security issues pertaining to possible attacks. Several works focus on using machine learning for misuse, intrusion and anomaly detection

with Intrusion Detection Systems (IDS) in wireless networks in general, and UAV systems specifically, by classifying inputs in the network [17]–[19].

On the other hand, there are several works on the communication security aspect on UAV systems; policy-based fingerprint features and methods for data redundancy [20], software defined network (SDN) based architecture [21], using artificial intelligence on incident command response co-ordination [22] as well as authentication of trusted drones in the GCS [23], encrypted communication channel and authentication to prevent hijacking [24]. One of the recent works, MAVSec [25], proposed a security-integrated version of the commonly used, lightweight UAV communication protocol, MAVLink to address its vulnerabilities. Their comparison of different encryption algorithms including AES varieties, RC4 and ChaCha20 in terms of memory usage and CPU consumption showed that ChaCha20 can be integrated into MAVLink without affecting its performance drastically. Furthermore, ChaCha20-Poly1305 authentication encryption, which is a combination of ChaCha20 stream cipher with Poly1305 message authenticator has been shown to be suitable as a fast and secure transport layer security protocol for IoT devices and remote cloud servers [26], [27]. Motivated by this work, we also consider the feasibility for the implementation of ChaCha20 stream cipher with Poly1305 message authenticator in UAV systems.

There are many existing methods on the communication side, i.e., static encryption schemes, and on the intrusion detection side, i.e., static and machine learning approaches. However, development of a feasible holistic security scheme that ensures the security of the entire system, especially when there is computation offloading with continuous video streaming processes [28], is still an open problem. Such a security scheme should not only consider the low resource constraints on-board UAVs, but also minimize degradation of the throughput of the data.

As shown in many previous works, machine learning techniques are successful candidates for the intrusion detection on multi-UAV systems. However, the hardship and the challenges on acquiring data to train, together with the dynamics of the system, creates a need to have a system that is suited to adopt machine learning models. Moreover, most of the existing datasets [29]–[31] that have been used for training IDS are not specifically created for UAV systems, and hence are not relevant to e.g. computation security in FANETs around video analytics. There is a need for data collection on a realistic simulated and/or emulated multi-UAV testbed that can be used for training different learning algorithms. In addition, there is a need for computation security solutions that are not cost-prohibitive and are portable when experimenting on real systems. Our work helps equip UAV systems through the novel DroneNet-Sec frameworks and algorithms for training and testing data to develop IDS with selected machine learning models that improve computation security in video analytics tasks in UAV systems.

C. Testbeds for UAV Systems

In order to have realistic testbeds for testing and validating innovative solutions and schemes that are related but not limited to security, policies, protocols, energy and performance optimization methods for multi-UAV FANETs, simulation is necessary, but not sufficient by itself in terms of resource requirements and computation capabilities. To have enhanced experiments by leveraging the advantages of both simulations, i.e., the scalability, and emulations, practicality and resourcefulness, there have been many prior studies to create end-to-end-frameworks on both sides in the context of FANETs. Recent works such as [32], [33] detailed the implementation of Virtualized Environments (VE) for multi-UAV network emulation, and testing of multi-UAV FANET simulations along with network service deployments on VEs.

To meet the needs of experimenting a solution on multi-UAV systems featuring FANETs as described, there are also several works that created hybrid testbeds combining different simulators and emulators to cover multiple components. A recent study that focused on creating a hybrid testbed for validating experiments viz., VENUE [34] proposed a VE for multi-UAV emulation for realistic experiments with service deployments that can be integrated with 5G technologies. VENUE is built on Linux Containers (LXC), using ns-3 simulator and emulations on real UAVs. An implementation of the emulation module with TapBridge model (virtual TAP interface and a Linux Bridge) inside ns-3 is used to make it interact with external real devices and VEs. This work however, did not consider the applicability of mobility models for multi-UAV system applications or the pre-installation of routing capabilities, which are the features needed for more realistic application implementations.

Apart from VE, there are other prior approaches such as [35] that provided a hybrid implementation, creating a setting that features the injection of traffic generated from a network simulator to an emulated live [36] network and vice-versa. This work did not test network protocols such as routing protocols, or extended their implementation for distributed networks, making it hard to have a more realistic network simulator.

In order to implement, test and validate our security mechanisms in a scalable, and end-to-end manner, a hybrid testbed was needed, combining simulation on ns-3 with emulation of VEs that can be deployed on any physical machine that is located on the ground or on the cloud, specifically Powder nodes [10]. Our hybrid testbed jointly integrates network and UAV simulation, extending features that are not available in these wireless facilities. Ns-3 provides models (e.g., battery, mobility) and helps us to instantiate a rich wireless topology. Our work novelty is in the consideration of cloud-based testbeds, joining extendable and resizable network simulator and wireless facilities capabilities to deploy hybrid testbed environments to support development of schemes to secure network-edge connectivity and improve computation security in drone video analytics.

III. DroneNet-Sec Framework Implementation

A. DroneNet-Sec Overview

In this section, we detail the DroneNet-Sec framework implementation illustrated in Fig. 3 that is based on the Zero Trust Architecture paradigm. Our approach combines secure messaging for network-edge connectivity security, and attack detection with countermeasures on edge resources for computation security. A secure messaging scheme with custom packet design in MAVLink protocol is deployed in a FANET module in order to secure the communication in between drone-to-drone and drone-to-edge server nodes. Apart from the secure messaging module, the FANET includes an attack monitoring and detection module, a deamon controlling simulation commands and metrics collection and a video analytics containers manager. Our framework provides measures for data integrity and confidentiality by implementing a secure messaging scheme, as well as for data access controllability, realizing the design based on the ZTA paradigm. DroneNet-Sec also includes a secure mechanism deployed in the edge server, in a UAV system with containerized video analytics tasks, focusing on attack types in Fig. 2.

There is a pre-trained machine learning model, i.e. decision tree, deployed on edge server for anomaly detection, with a control and countermeasure scheme that constantly monitors the system through a simulation control for computation security. This pre-trained model is updated dynamically, since it is being trained further continuously with the feedback received from the simulation control, as well as the investigation results of the compromised parties. This secure mechanism, detects DoS attacks and can launch security controls such as e.g., pausing of containers inside compromised UAV or edge server components, for further investigation and recovery.

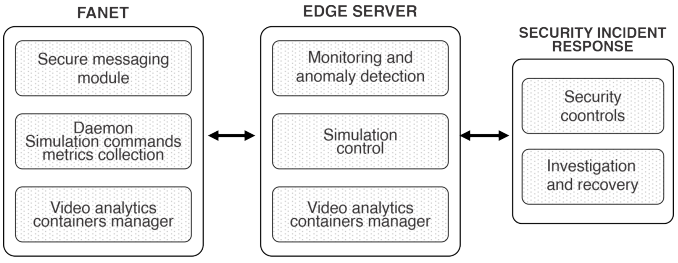

Fig. 3. DroneNet-Sec framework components involving FANET and edge server resources that are integrated with security incident response.

In the following, we provide details of our system threat model, and also describe our use of zero trust architecture as well as the hybrid testbed deveopment for studying network-edge connectivity security and computation security in DroneNet-Sec.

B. Threat Model

Amongst the numerous targeted attacks on availability, integrity and privacy in UAV systems, our focus as shown in Fig. 2 is on DoS, (i.e., UDP flooding), Replay attack and MITM attack types. Our threat models is created considering specific scenarios as listed in Table I to be implemented and tested on the hybrid testbed.

For UDP flooding, a compromised UAV starts flooding while the normal network flow with image transmissions continue concurrently. Each incoming packet is monitored inside the edge resources, and a prediction is made with the pre-trained Decision Tree model. In order to filter false positives, a threshold is used to decide whether to detect an attack occurrence. This threshold, selected as 500 after running experiments helps us to decide on a high enough volume to exclude false positives and on a low enough volume to quickly detect and deploy a countermeasure to mitigate the attack. Once the number of packets received and classified as suspicious by the model reaches the above threshold, the source UAV is labeled as untrusted, and the countermeasures for computation security are enforced. The containers inside the compromised UAV are paused after getting a snapshot, a message is sent to the UAV to ground it, and all the other network activities are halted on that UAV, to prevent further impact of the violation.

In the case of Replay attacks, when one of the UAVs is compromised and eavesdropped, it replays a previous message that is spoofed. In order to prevent Replay attacks, the receiver UAV/edge keeps tracks of the nonce values of the packets it receives, updates the biggest nonce each time it receives a packet with a valid nonce i.e., greater than the previous largest number of nonce. If the incoming nonce is smaller than or equal to the tracked largest nonce, replay attack is logged and the packet is discarded.

In the case of MITM attacks, a Poly-1305 message authentication code (MAC) is used to authenticate the sender on the receiver side i.e., on the UAV/edge server.

TABLE I
Threat model for a UAV system.

Attack	Scenarios	Countermeasure/Prevention
DoS - UDP flood	A UAV is compromised and attacks edge server nodes	Countermeasure: Constant monitoring and detection with pre-trained Decision Tree model on edge nodes
Replay	A UAV is compromised and attacks other UAV/edge server nodes	Prevention: Using nonce in custom message packet with nonce tracking & checking on the receiver UAV/edge nodes
MITM		Prevetion: Using Poly-1305 MAC for source authentication on the receiver UAV/edge nodes

C. Use of Zero Trust Architecture

Our DroneNet-Sec is based on the Zero Trust Architecture (ZTA) paradigm [37] that uses the "never Trust, always verify" principles. DroneNet-Sec provides intelligent mechanisms with secure communication and image data messaging, to dynamically adapt the security of the multi-UAV system as shown in Fig. 4. We can see that our system with ZTA involves reducing access to resources to only those who are "trusted" agents. We validate all access as necessary and continuously

verify the identity of every access request, providing a collection of concepts, designs, and architectures that are created with the purpose of eliminating the risk.

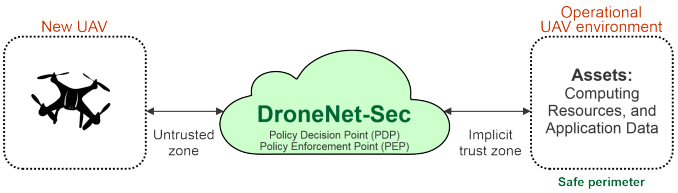

Fig. 4. A new UAV's attempt to access the system resources will be considered as untrusted and moderated using a Policy Decision Point (PDP) and a Policy Enforcement Point (PEP) by DroneNet-Sec to protect the UAV system assets in the operational environment.

Our ZTA approach focuses on eliminating unauthorized access to data and services of the system, making access control enforcement as granular as possible. High granularity i.e., dividing access control privileges to a resource into small pieces that allow fine-grained authorizations to known agents/UAVs as needed without over/under-provisioning of privileges. We enforce policies to allow data/resource assets access to only authorized and approved agents, security groups. All other agents, attackers, unknown agents, etc. are not authorized and prohibited from accessing the data or resource assets.

The ZTA-enabled infrastructure ensures users are trustworthy, and allows only valid system requests to be authorized through the PDP and PEP functionality. Two basic rules are applied with the ZTA, first, authentication, and then authorization. Policies such as, can the system remove any uncertainty about an agent trying to access the system? Is the access justified? Is the agent trusted? For resource access, risk-based policies need to be implemented to ensure that authorization policies are performed accurately. Implied trusted zone, outlines an area where all the agents are trusted to at least the level of the last PDP and PEP gateway. For instance, a UAV has to pass these policies to be allowed to have access to a common trusted zone, safe perimeter in the operational UAV environment.

We remark that the PDP and PEP rules utilize a set of controls and mechanisms such as all traffic beyond the trusted zone have a common level of trust. These rules must not operate or apply policies beyond its location in the flow of traffic. Furthermore, the implied trusted zone must be considered as small as possible as a best practice.

D. Hybrid Testbed Development

We deployed a hybrid testbed for the purposes of this study as shown in Fig. 5. The testbed setup includes two allocated wireless nodes, one for our computation tasks on the edge, and another node for our FANET implementation in a real-time network simulator that instantiates a richer wireless topology for our agent drones.

The components employed for the deployment of our hybrid testbed include, Powder [10] a platform for Open Wireless Data-driven Experimental Research, that provides us with computing resources to allocate our edge server and FANET resources. This platform is composed synergestically with ns-3 [9], an open-source network simulator primarily used in networking research. The system communicates with MAVLink [8], a hybrid lightweight messaging protocol used for communication amongst drones. It also provides communication between on-board drone components, following a modern hybrid publish-subscribe and point-to-point design pattern.

An autopilot system, Ardupilot [38], supports the flight controllers, sensors and frame types. A simulator allows us to test the behaviour of real drones without any software. Software In The Loop, [39] and data located in Docker [40] containers that allows container-based applications are deployed in an easy, lightweight and consistent manner [41] in our hybrid testbed.

E. Network-Edge Connectivity Security

The main components of our hybrid testbed related to experiments with network-edge connectivity security can be described as follows:

Communication between Powder nodes. We implemented a *ns3::TapBridgeHelper* class and used it for the ns-3 side on the FANET to facilitate P2P (Peer-to-Peer) drone communications.

Reliable User Datagram Protocol (RUDP). While User Datagram Protocol (UDP) is simpler and faster than Transmission Control Protocol (TCP), it does not provide sequenced message delivery. We made the decision to use UDP over TCP because UDP is message-based and not stream-based compared to TCP. Despite the fact that UDP structure is relatively simple and offers high processing capabilities, its reliability is low as it does not check whether packets have been delivered to the destination or not. Another main drawback of UDP is that it keeps transmitting regardless of reliability issues, and also sometimes packets are not delivered sequentially [42]. To address above issues, we implemented a custom reliable UDP (RUDP) protocol for use in the drone-to-drone and drone-to-edge communications (i.e., in the non inter-drone services). Our reliable protocol attempts to decrease packet loss, and guarantees sequenced data delivery. This aids not just to improve the data throughput but also reduces the latency on data packets, which is achieved with the employment of sequence numbers, acknowledgment numbers, and timers to monitor packet loss and sequence of the packets received.

Secure Messaging. MAVLink packets are encrypted with ChaCha20-Poly1305 directly [27], with no need for Transport Layer Security (TLS) certificates. This also allows us to prevent Replay attacks through the use of an incrementing nonce. In addition, packet encryption can also be easily toggled within our experiments to test performance impacts. Fig. 6 gives an overview of the custom encrypted packets, with information about target sysid, nonce, MAC tag and encrypted MAVLink packet in terms of size.

Encrypted packets sender and receiver with ChaCha20-Poly1305 algorithm. Fig. 7 shows how ChaCha20-Poly1305 [27] was implemented. The implementation was done with the package available at [43], which is a self-contained Python

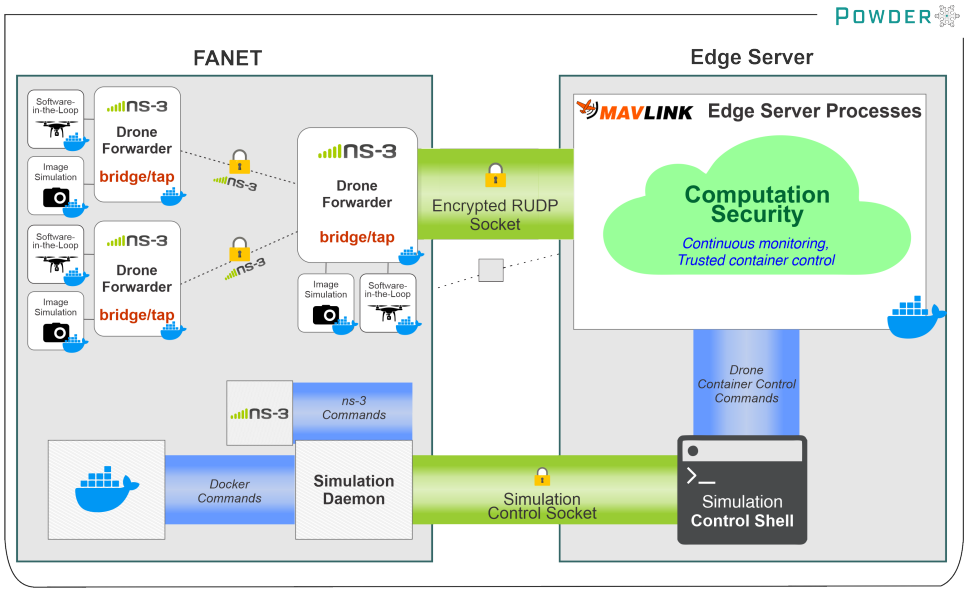

Fig. 5. Hybrid testbed that includes two allocated wireless nodes, one node for our computation tasks on the edge, and another node for multi-UAV nodes implementation; built using ns-3 bridge/tap functionality, containers for video data processing as microservices, and MAVLink lightweight messaging protocol to facilitate peer-to-peer drone communication in a repeatable and scalable experiment setup.

1 byte	1 byte	12 byte	16 byte	8-263 bytes (Size of Mavlink packet)
target sysid	src sysid	nonce	MAC tag	Encrypted Mavlink packet

Fig. 6. Secure messaging with packet encryption in our MAVLink protocol implementation.

package of low-level cryptographic primitives. Authenticated encryption is implemented following one of our basic ZTA rules for authentication, which simultaneously assure the confidentiality and authenticity of data. The process of encryption and decryption of packages works as follows. The sender, provides a plain text in MAVLink protocol, simultaneously, a ChaCha20-Poly1305 cipher from the shared key and current nonce is generated. This updated cipher with receiver's MAVLink sysid is later encrypted together with the plain text from MAVLink. Finally, the nonce is incremented and an encrypted packet is sent. From the receiver side, a received encrypted data alongside with nonce and sysid is received. After this happens, a series of decisions are made before the plain text is decrypted. For instance if the current or the previous received nonce is not received, then the packet must be discarded, providing a log replay attack. Otherwise, decisions such as updating the received nonce or save current received nonce are also made before the ChaCha20-Poly1305 algorithm is generated in order to decrypt the plain text.

Scalable number of containers in ns-3. Our framework is implemented with Docker containers, which run the FANET ns-3 simulations. We implemented a Python script that scales the number of containers in our ns-3 implementation. This implementation supports reliable and more configurable code that helps with an easy integration and supports updates for future work involving container orchestration. Our system can support any number up to around 254 drones as MAVLink supports 255 total systems and a numeric naming scheme is used for the bridges/taps, which is necessary for our peer-to-peer FANET setup.

Image simulation, sending packages from edge to server and viceversa. Our experimentation relies on the open-source 10-class geospatial object detection dataset [44]. This also aimed us to evaluate the image/data transmission simulation performance of our proposed hybrid testbed. For each image, it splits into chunks that fit the MAVLink data packet (253 bytes) until it waits for the edge server to send a data transmission handshake request. It responds with an acknowledgment, then it sends out each image chunk consecutively in MAVLink encapsulated data packs. It does this until the image has been sent out fully, then waits for the next request from the edge. The goal is to generate network traffic, once all the images in the dataset are exhausted, it just restarts the process.

A secure messaging scheme with custom packet design enforcing data privacy and data integrity, securing the messaging in between drone to drone and drone to edge server providing a secure protocol over MAVLink. Custom messaging design enforces encryption [27] on the MAVLink packet for data privacy. This includes nonce information that can be used for enhancing data integrity by keeping track of last nonce, detecting and preventing possible replay attacks. The message authentication code (MAC) in the custom message is used to prevent MITM attacks by ensuring the source of the message as a trusted party. The custom packet is highly scalable, and can provide the secure messaging up to 254 UAVs with MAVLink protocol.

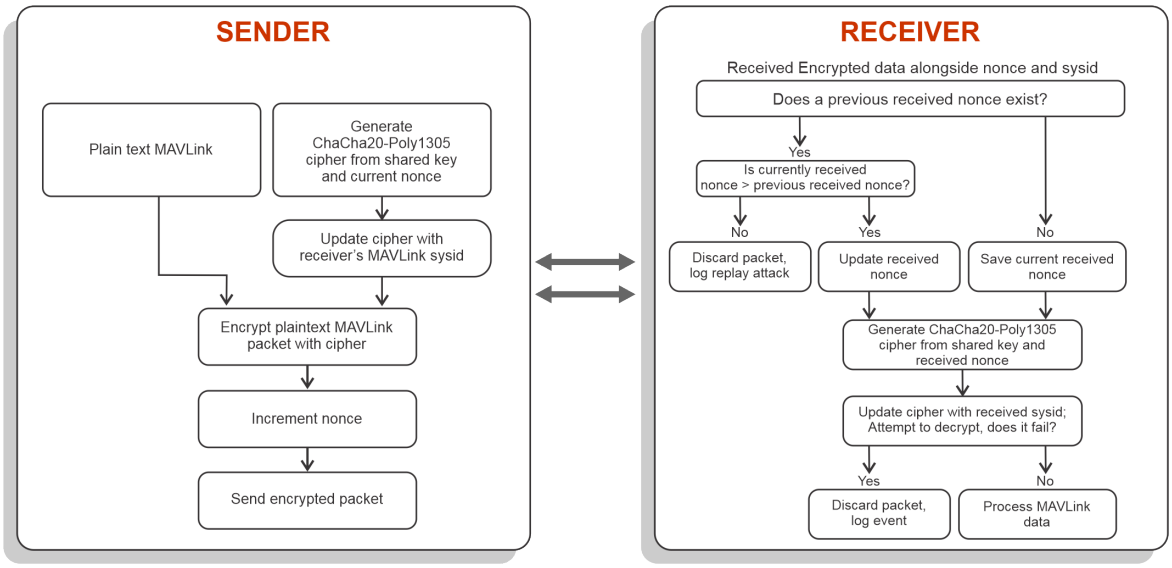

Fig. 7. Encrypted packets sent between the sender and receiver using the ChaCha20-Poly1305 algorithm.

F. Computation Security

In order to ensure secure computation task offloading in the drone video analytics, we need to address security measures for data confidentiality, data integrity, and data access controllability. There is a need of a dynamic framework to determine if the source UAV is trusted i.e., the data it sends can be received and used in the processes in the recipient drone. Based on the ZTA, the trusted execution environment allows drones to execute video analytic tasks to run with containerized data as shown in Fig. 8. In support of such a functionality, a machine learning algorithm that we implemented decides if the source is trusted or not.

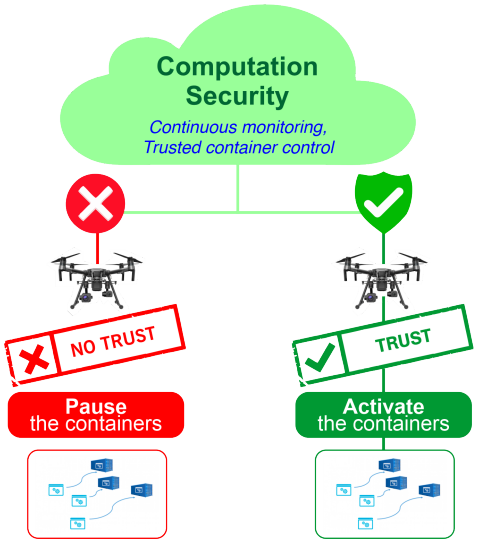

Fig. 8. Cloud-hosted monitoring and anomaly detection module that dynamically ensures computation security by controlling container execution only on trusted UAV nodes.

Computation security is achieved by implementing a framework that is dynamic, persistent, and intelligent. Our framework continuously checks the status of the edge server and the components (UAV, UAV systems, and machines) that are part of the system. Our machine learning algorithm uses system monitoring data, including the outputs received from UAV-to-edge and vice versa. Any device that attempts to get access to the system must be verified, including data that can be accessed and compromised. The framework provides a robust solution to put all containerized computation safe and mitigates impact of any potential DoS attacks on the system. While constantly monitoring the system, our machine learning implementation is trained for DoS attack detection and thus classifies all the traffic deployed on the edge. Dynamicity of the framework comes from the constant monitoring and the ability to learn from more data produced in the system in order to handle new attack types.

When there is a DoS attack detected, in order to avoid possible false positive effects, a threshold is used before deciding to pause the containers inside a compromised component. When the threshold is reached, meaning the suspicious activity that has been classified as an attack, the containers inside the compromised component are paused and the network traffic for that party is halted for further investigation and recovery. Once the containers are paused, the data inside and the state of the container is preserved by a snapshot, while halting the network also prevents the compromised party to do further damage.

The integrated cloud resources for the machine learning process help with computationally intensive operations involved in the training step. Our cloud-hosted machine learning scheme forms a key aspect in dynamically and successfully configuring the security of the drone flights at the network-edge. In order to test our cloud-hosted machine learning implementation, we utilize the BOUN DDoS dataset [45]. The dataset is generated through Hping3 traffic generator software by flooding TCP SYN, and UDP packets, towards a server where over 4000 active user traffic was flowing on the router at the same time with the attacks. It includes attack-free user traffic and attack

traffic that makes it suitable for the evaluation of network-based DDos attacks detection.

We trained a machine learning model and deployed it on an edge server for anomaly detection, in which training data can be any existing data set and/or real data collected from the hybrid testbed. Model weights are updated continuously and dynamically as new data is monitored from the system as illustrated in Fig. 9

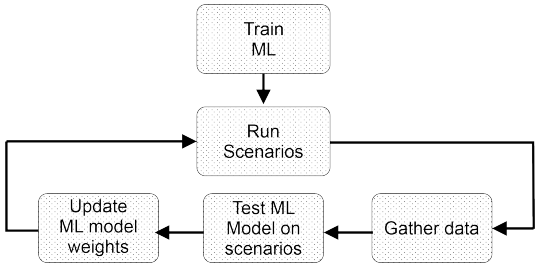

Fig. 9. Machine learning data training steps with existing or collected data from the hybrid testbed.

Algorithm 1: Dynamic DoS Attack Mitigation

```
Result: secure computation on trusted drone nodes
initialization;
while communication between drone and GCS happens do
    continues monitoring;
    if drone is compromised then
        identified as untrusted;
        labeled as untrusted;
        containers are paused;
        network access is disabled;
        last stage and data inside container is preserved, further
          damage is mitigated;
        admin is attached to the container for
          recovery/investigation process;
    else
        drone is trusted;
        containers are active;
    end
end
```

Algorithm 1 shows the steps for the anomaly detection process that consists of a continuous monitoring of the system. Once an attack is detected by the pre-trained Decision Tree model on the edge server, the agent is identified and labeled as untrusted. A snapshot of the containers inside the compromised drone is taken, then the containers are paused. The network communications are also halted with a message that is sent to the compromised drone to ground itself for further investigation and recovery.

The containerized design of the tasks allows the investigation and recovery phase to be performed faster. This is because the snapshots can be used to examine the problem further, and rebooting the compromised drone or adding a new drone to the system instead of the compromised one can be done by deploying the corresponding images. Moreover, isolating the compromised drone prevents the spread of the effect. This scheme uses the monitored network flow to detect an attack that has been used in the training of the model weights, specifically DoS - UDP flooding attacks, which are deployed on the edge server. The model can be updated dynamically to detect more complicated attacks and a variety of attacks during on-going system operations.

IV. Performance Evaluation

In this section, we present experiments that demonstrate the effectiveness of our DroneNet-Sec framework. All experiments are simulated in our hybrid testbed, in which UAVs run containers in a multi-hierarchical manner, with centralized-control communication networks, and leveraging edge server data. Each drone is operated by one GCS and all commands and computation processes are controlled by a single edge server. We also provide experiments to evaluate the performance of our proposed hybrid testbed. Two Powder wireless nodes include FANET and edge server respectively. Seven drones running via ns-3 simulator, Docker containers and Daemon running on host node acts as the container manager. We monitor the system CPU and memory measurements for each container, during both normal flows and flows with attacks.

A. Network-Edge Connectivity Security Experiments

We implemented a novel secure and realistic configuration of computing resources that are extendable and resizable on the ns-3 side. Through this configuration, we simulate a large number of UAVs, that interface with the Powder side for emulating various edge and cloud computing resources. In addition, we develop an image-offloading simulation between the FANET network and edge server in a secure manner through a scheme that defends against MITM and Replay attacks. Further, we implement a protocol that is specific to drones and running experiments with the best encryption method to secure the communication between UAVs and GCS. Specifically, we use a custom encrypted packet that is designed with MAVLink messages using ChaCha20-Poly1305 for efficiency and light-weight functionality. Chacha20-Poly1305 is the combination of two algorithms, the Chacha20 stream cipher and the Poly1305 MAC, both designed separately by Daniel J. Bernstein. These two algorithms were designed to be fast in software and are widely adopted. They also present excellent options to provide encrypted communication between, low resourced or constrained UAVs [27].

ChaCha20-Poly1305 is efficient and has little overhead for on-board UAV resources. In order to show that ChaCha20-Poly1305 is indeed efficient and has little overhead for on-board UAV resources, we compare the performance of the system with and without encryption. Fig. 10 shows the CPU Usage %, and Fig. 11 shows the Memory Consumption measurements of messaging in the system without encryption and with different encryption algorithms: ChaCha20-Poly1305, AES GCM, AES EAX and AES CCM.

Our performance evaluation results demonstrate that our security mechanism guarantees secure message transmission and communication between drone-to-drone and drone-to-GCS links, with less degradation of CPU consumption and memory usage, compared to other common encryption algorithms. In

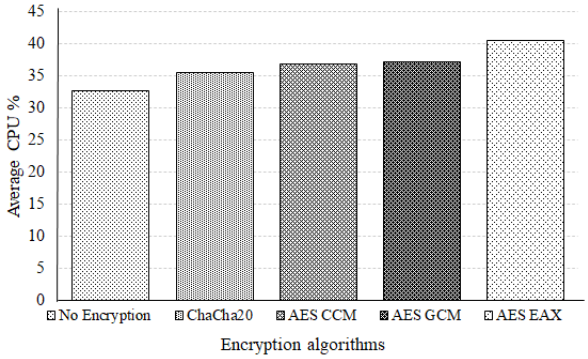

Fig. 10. CPU usage performance comparison with different encryption methods and without encryption.

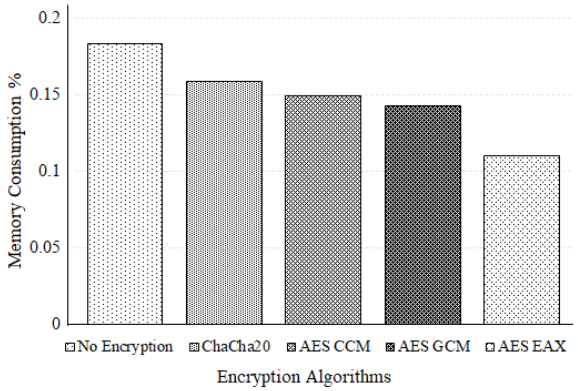

Fig. 11. Memory consumption performance comparison with different encryption methods and without encryption.

these results, we can see that the average CPU % consumption for ChaCha20 is the closest one to the consumption without any encryption, which is the second smallest as expected. On the other hand, in memory consumption %, we can see that ChaCha20 has the closest value to no encryption case, which is the second highest. The reason of the reverse trend we see on memory consumption when compared to CPU consumption is as follows: a faster messaging allows more images being transmitted with packets, leading to more packets in the RUDP queue. Hence, the memory consumption is the highest in the no encryption case, followed by the ChaCha20 case. Therefore, our evaluation results show that ChaCha20-Poly1305 performs in a more efficient manner compared with other encryption schemes (i.e., AES CCM, AES EAX, AES GCM) and no encryption (i.e., plain-text). Thus, we demonstrate our novel security mechanism which has the capacity to save memory and battery for drones that have resource constraints in the network-edge within a UAV system.

B. Computation Security Experiments

Dynamic security scheme with machine learning accurately detects anomaly events. While the system is being monitored on both UAV and edge sides, we run experiments that involve launching DoS attacks i.e., UDP flooding. Either external UAV system or a machine might be manipulated to instantiate an attack, in order to interrupt the system availability and thereby the system communications. Fig. 12 shows one experiment in which an external agent tries to get access to a trusted agent. Since the machine learning implementation runs continuously, it checks the status of all the agents that are part of the mission. Once the external agent is detected, the trusted agent container is immediately paused, and a snapshot of the state and data is obtained. Concurrently, all network communication is halted and the trusted UAV is sent to GCS for further analysis and recovery. It is important to highlight that even if one agent's container is paused, the rest of the agents that are part of the mission are still working. This prevents the spread of the breach in the system to other trusted parties, which is important in cloud systems to avoid amplification of a breach at cloud-scale. Moreover, the offline analysis and recovery phase will provide valuable information to improve the security of the system, while rebooting the compromised UAV will be fast with containerization.

In order to understand better the severity of this type of attacks, we run a series of experiments to demonstrate the number of transmitted images with no attacks and the system under increasing number of attackers. In Fig. 13 shows the number of transmitted packets with a normal network traffic and when 1, 2 and 3 attackers flooding the network traffic, respectively. As expected, the degradation on the system gets more severe as the number of attackers and the duration of attacks increase.

Security framework improves the network throughput under DoS attack. We considered an experiment scenario to demonstrate network throughput improvement using the security framework under DoS attacks. In the experiment scenario, the packets are sent for the first 60 seconds without any attack, on the 60^{th} second, the first attacker starts UDP flooding. On 120^{th} second, the second attacker starts UDP flooding. On the 180^{th} second, the third attacker starts UDP flooding. The number of packets received by the attack monitoring on the edge server under this experiment can be found in Fig. 14. This experiment shows how the framework handles the attacks in short time and the throughput does not flatten as in Fig. 13, where there are no security measures. Also, the throughput does not drop after attacks start, but packets are received continuously, while under attack, since only the compromised UAV container is paused.

Machine learning classification methods implementation utility. We tested a variety of classical machine learning classification methods on the BOUN DDoS dataset [45]. To do this, we made significant changes to the implementation associated with work [46], and made several changes to their classical machine learning code for our purposes. We selected 4 features; $Time, Frame\_length, Source\_ip, Destination\_IP$ with respect to our monitoring collections of packets from our hybrid testbed that are relevant to our setup. The data from BOUN containing UDP flood attacks with the 4 features were used to train 6 different models. Given that all IPs are the same, the port is used as a weight against the IP with a bitwise XOR, and the dataset relies on inference between IP addresses, packet size and time.

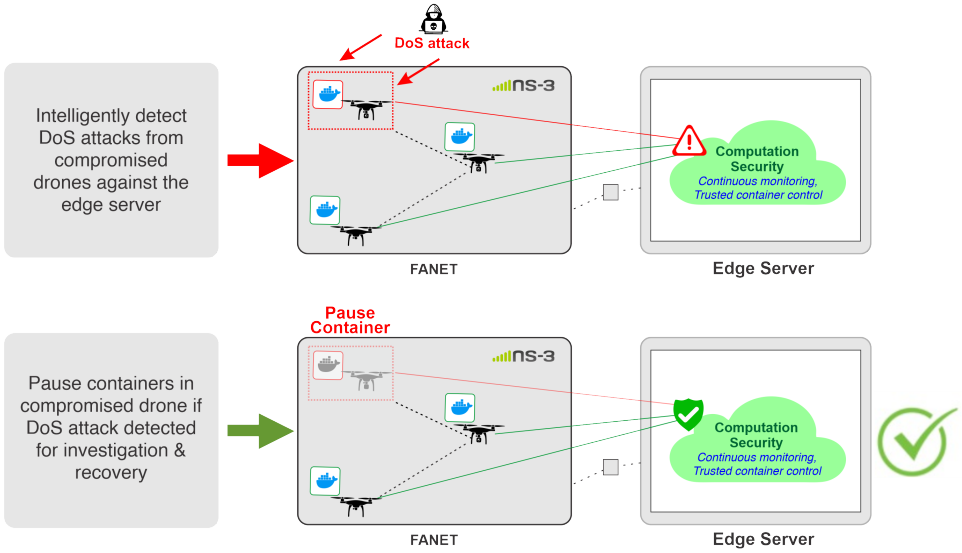

Fig. 12. Experiment scenario illustration to handle an attacker's attempt to get unauthorized access to information from a secure drone, which involves anomaly detection and a security control initiation to pause the relevant container in a compromised UAV node.

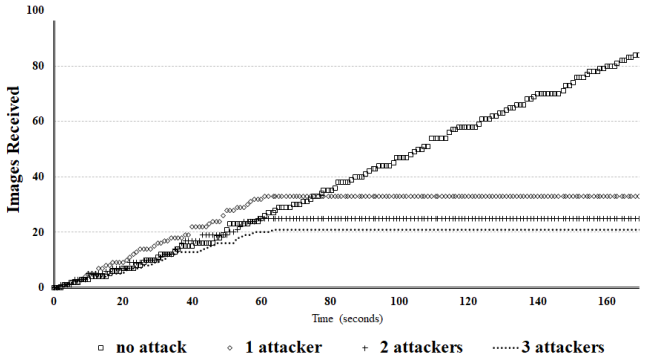

Fig. 13. Total images received without any security measures for cases involving: no attacker, one, two and three UDP flooding attackers.

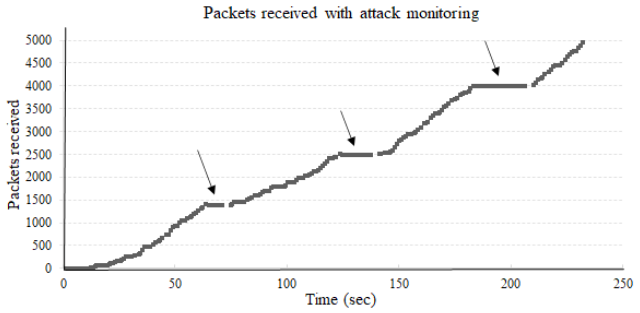

Fig. 14. Total packets received under UDP flooding attack experiment scenario involving: packets sent without any attack in first 60 seconds, and consequent attack initiations with a 60 second inter-attack period.

With respect to the testing measurements of the trained models that can be seen in Table II, Decision Tree, and KNN classifiers performed the best, followed by Random Forest; all models achieved greater than 90 percent F1 score, while Logistic Regression, Naive Bayes, AdaBoost performed poorly. Table III shows timing characteristics to demonstrate

TABLE II
MACHINE LEARNING BASED CLASSIFICATION RESULTS

	Accuracy	**Precision**	**Recall**	**F1**
Logistic Regression	0.947	0.5	0.723	0.599
Naive Bayes	0.947	0.5	0.723	0.591
KNN	0.988	0.948	0.923	0.935
Decision Tree	**0.988**	**0.917**	**0.957**	**0.937**
AdaBoost	0.953	0.557	0.959	0.703
Random Forest	0.894	0.894	0.967	0.929

that using machine learning models (especially Decision Tree) to perform inference continuously is suitable in terms of speed and accuracy in a real-time UAV system.

TABLE III
MACHINE LEARNING TRAINING AND TESTING TIME RESULTS

	Training Time (sec)	**Test Time (sec)**
Logistic Regresion	43.371	0.3435
Naive Bayes	4.175	1.0653
KNN	192.563	178.699
Decision Tree	39.4376	**0.2643**
AdaBoost	1415.65	43.738
Random Forest	2254.22	26.634

Using the collected statistics and timing results from these experiments, we chose to implement the Decision Tree model that was the fastest and most accurate model in comparison with the other models. As a final step, we evaluated this chosen model on more than 14,000 simulated packets that were collected from our hybrid testbed implementation, with an average prediction time per packet being less than half a second. The accuracy achieved on the real data experiments on the testbed is 0.977, where precision is 1, recall is 0.920 and the F1 measure is 0.958, compared to the results obtained from the training dataset. The results on the real data thus show that our chosen model is highly suitable for detecting UDP flood attacks and allows UAV system operators to make the necessary corrective actions with high speed and accuracy.

V. Conclusion

In this paper, we proposed a framework to secure drone communications and improve computation security in video processing container tasks in terms of data privacy, integrity and availability. Our scheme features an intelligent and dynamic decision algorithm to detect anomaly events in a UAV system with FANET configuration without decreasing the performance. Our scheme performance can be improved continuously with the feedback from the system itself, as the system operates in the field. We created and deployed a hybrid testbed management module that synergized simulation and emulation experiments as part of realistic scenarios for evaluation studies. Hybrid testbed experiment results showed that our proposed security framework successfully detects anomaly events on drones in a fast and accurate manner, allows for secure messaging with the MAVLink protocol, and provides UAV system operators the necessary knowledge to securely operate containers for drone video analytics.

As part of future work, we are planning to create investigation and recovery schemes for video processing pipelines, along with forensic investigation techniques based on offline analysis. This will help in analyzing the snapshots or images of the paused containers for providing feedback to the system to further improve the security scheme by using transfer learning. Future work can also support multiple FANETs in the edge server, and provide more options such as drones going out of range, or operating on different FANET topologies. Further, our work on using machine learning models can be extended to detect and mitigate the impact of other attacks such as probing, GPS jamming or zero-day anomalies.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Award Numbers: CNS-1647182, CNS-1950873, and the Army Research Lab under Award Numbers: W911NF1820285 and W911NF1910181. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the Army Research Lab.

References

[1] S. Dahiya and M. Garg, "Unmanned aerial vehicles: Vulnerability to cyber attacks," in *Int. Conf. on Unmanned Aerial Syst. in Geomatics*, pp. 201–211, Springer, 2019.

[2] N. A. Khan, S. N. Brohi, and N. Jhanjhi, "UAV's applications, architecture, security issues and attack scenarios: A survey," in *Intelligent Comput. and Innovation on Data Sci.*, pp. 753–760, Springer, 2020.

[3] A. Fotouhi, H. Qiang, M. Ding, M. Hassan, L. G. Giordano, A. Garcia-Rodriguez, and J. Yuan, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," *IEEE Commun. Surveys & Tut.*, vol. 21, no. 4, pp. 3417–3442, 2019.

[4] D. Chemodanov, P. Calyam, and K. Palaniappan, "Fog computing to enable geospatial video analytics for disaster-incident situational awareness," *Fog Comput.: Theory and Practice*, pp. 473–503, 2020.

[5] C. Qu, S. Wang, and P. Calyam, "DyCOCo: A dynamic computation offloading and control framework for drone video analytics," in *IEEE Int. Conf. on Netw. Protocols*, pp. 1–2, 2019.

[6] R. R. Ramisetty, C. Qu, R. Aktar, S. Wang, P. Calyam, and K. Palaniappan, "Dynamic computation off-loading and control based on occlusion detection in drone video analytics," in *Int. Conf. on Distributed Comput. and Netw.*, pp. 1–10, 2020.

[7] H. Benkraouda, E. Barka, and K. Shuaib, "Cyber-attacks on the data commun. of drones monitoring critical infrastructure," pp. 83–93, 2018.

[8] L. Meier, J. Camacho, B. Godbolt, J. Goppert, L. Heng, M. Lizarraga, *et al.*, "MAVLink: Micro air vehicle communication protocol," *https://mavlink.io/*, 2013. Accessed: 2020-06-18.

[9] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator," *SIGCOMM Demonstration*, vol. 14, no. 14, p. 527, 2008.

[10] "POWDER: platform for open wireless data-driven experimental research." https://www.powderwireless.net/. Accessed: 2020-06-18.

[11] C. Ge, X. Ma, and Z. Liu, "A semi-autonomous distributed blockchain-based framework for UAVs system," *Journal of Syst. Architecture*, vol. 107, p. 101728, 2020.

[12] D. Chemodanov, C. Qu, O. Opeoluwa, S. Wang, and P. Calyam, "Policy-based function-centric computation offloading for real-time drone video analytics," in *IEEE Int. Symposium on Local and Metropolitan Area Netw.*, pp. 1–6, 2019.

[13] A. Sukhov, P. Calyam, W. Daly, and A. Ilin, "Towards an analytical model for characterizing behavior of high-speed vvoip applications," *Computational Methods in Science and Technology*, vol. 11, no. 2, pp. 161–167, 2005.

[14] V. D. Gligor, "A note on denial-of-service in operating systems," *IEEE Trans. on Softw. Eng.*, vol. SE-10, no. 3, pp. 320–324, 1984.

[15] E. Winjum, A. Hegland, Kure, and P. Spilling, "Replay attacks in mobile wireless Ad Hoc networks: Protecting the OLSR protocol," vol. 3421, pp. 471–479, 2005.

[16] A. Mallik, "Man-in-the-middle-attack: Understanding in simple words," *Cyberspace: Jurnal Pendidikan Teknologi Informasi*, vol. 2, no. 2, pp. 109–134, 2019.

[17] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys & Tut.*, vol. 21, no. 1, pp. 686–728, 2018.

[18] F. Tang, Y. Kawamoto, N. Kato, and J. Liu, "Future intelligent and secure vehicular network toward 6G: Machine-learning approaches," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 292–307, 2019.

[19] H. Rydén, S. B. Redhwan, and X. Lin, "Rogue drone detection: A machine learning approach," in *IEEE Wireless Commun. and Netw. Conf.*, pp. 1–6, 2019.

[20] A. Abdallah, M. Z. Ali, J. Mišić, and V. B. Mišić, "Efficient security scheme for disaster surveillance UAV communication networks," *Inf.*, vol. 10, no. 2, p. 43, 2019.

[21] C. Guerber, N. Larrieu, and M. ROYER, "Software defined network based architecture to improve security in a swarm of drones," in *IEEE Int. Conf. on Unmanned Aircraft Syst.*, pp. 51–60, 2019.

[22] M. Vassell, O. Apperson, P. Calyam, J. Gillis, and S. Ahmad, "Intelligent dashboard for augmented reality based incident command response coordination," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 976–979, IEEE, 2016.

[23] U. Challita, A. Ferdowsi, M. Chen, and W. Saad, "Machine learning for wireless connectivity and security of cellular-connected UAVs," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 28–35, 2019.

[24] K. Yoon, D. Park, Y. Yim, K. Kim, S. K. Yang, and M. Robinson, "Security authentication system using encrypted channel on UAV network," in *IEEE Int. Conf. on Robotic Comput.*, pp. 393–398, 2017.

[25] A. Allouch, O. Cheikhrouhou, A. Koubâa, M. Khalgui, and T. Abbes, "MAVSec: securing the MAVLink protocol for ardupilot/PX4 unmanned aerial systems," in *IEEE Int. Wireless Commun. & Mobile Comput. Conf.*, pp. 621–628, 2019.

[26] A. Langley, W. Chang, N. Mavrogiannopoulos, J. Strombergson, and S. Josefsson, "ChaCha20-Poly1305 cipher suites for transport layer security (TLS)," *RFC 7905*, no. 10, 2016.

[27] F. De Santis, A. Schauer, and G. Sigl, "ChaCha20-Poly1305 authenticated encryption for high-speed embedded iot applications," in *Design, Automation Test in Europe Conf. Exhibition*, pp. 692–697, 2017.

[28] H. Trinh, P. Calyam, D. Chemodanov, S. Yao, Q. Lei, F. Gao, and K. Palaniappan, "Energy-aware mobile edge computing and routing for low-latency visual data processing," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2562–2577, 2018.

[29] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *IEEE Symposium on Comput. Intelligence for Security and Defense Appl.*, pp. 1–6, 2009.
[30] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *IEEE Military Commun. and Inf. Syst. Conf.*, pp. 1–6, 2015.
[31] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset," *IEEE Comm. Surveys & Tut.*, vol. 18, no. 1, pp. 184–208, 2015.
[32] Z. Zhao, M. C. Vuran, Z. Aref, D. P. Young, W. Humphrey, S. Goddard, G. Attebury, B. France, B. Zhou, and M. M. Lunar, "A city-wide experimental testbed for next generation wireless networks," in *IEEE Int. Balkan Conf. on Commun. and Netw.*, 2019.
[33] D. Raychaudhuri, I. Seskar, G. Zussman, T. Korakis, D. Kilper, T. Chen, J. Kolodziejski, M. Sherman, Z. Kostic, X. Gu, H. Krishnaswamy, S. Maheshwari, P. Skrimponis, and C. Gutterman, "Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless," in *Int. Conf. on Mobile Comput. and Netw.*, MobiCom, Association for Comput. Machinery, 2020.
[34] V. Sanchez-Aguero, F. Valera, B. Nogales, L. F. Gonzalez, and I. Vidal, "VENUE: Virtualized environment for multi-UAV network emulation," *IEEE Access*, vol. 7, pp. 154659–154671, 2019.
[35] S. Yadav, M. Gaur, and V. Laxmi, "ns-3 emulation on ORBIT testbed," in *Int. Conf. on Adv. in Comput., Commun. and Inform.*, pp. 616–619, 2013.
[36] "Open-access research testbed for next-generation wireless networks (ORBIT)." https://www.orbit-lab.org/. Accessed: 2020-06-19.
[37] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," tech. rep., National Institute of Standards and Technology, 2019.
[38] "Ardupilot." https://ardupilot.org/. Accessed: 2020-06-18.
[39] "Software in the loop simulator (SITL)." ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html. Accessed: 2020-06-18.
[40] "Docker: Empowering app development for developers." www.docker.com/. Accessed: 2020-06-18.
[41] B. Mukherjee, S. Wang, W. Lu, R. L. Neupane, D. Dunn, Y. Ren, Q. Su, and P. Calyam, "Flexible iot security middleware for end-to-end cloud–fog communication," *Future Generation Computer Systems*, vol. 87, pp. 688–703, 2018.
[42] Tuong Le, G. Kuthethoor, C. Hansupichon, P. Sesha, J. Strohm, G. Hadynski, D. Kiwior, and D. Parker, "Reliable user datagram protocol for airborne network," in *IEEE Military Commun. Conf.*, pp. 1–6, 2009.
[43] "Pycryptodome." https://www.pycryptodome.org/en/latest/src/introduction.html. Accessed: 2020-06-18.
[44] "NWPU-VHR10 dataset." http://www.escience.cn/people/gongcheng/NWPU-VHR-10.html. Accessed: 2020-06-22.
[45] D. Erhan, "Boğazici university DDoS dataset," 2019.
[46] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *Int. Conf. on Adv. in Comput., Commun. and Inform.*, pp. 1222–1228, 2017.