IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 12, 2021, accepted May 22, 2021, date of publication May 25, 2021, date of current version June 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3083549

CoMAD: Context-Aware Mutual Authentication Protocol for Drone Networks

UMUT CAN CABUK[1], (Member, IEEE), GOKHAN DALKILIC[2], AND ORHAN DAGDEVIREN[1], (Member, IEEE)

[1]International Computer Institute, Ege University, 35100 Izmir, Turkey
[2]Computer Engineering Department, Dokuz Eylul University, 35390 Izmir, Turkey

Corresponding author: Umut Can Cabuk (umut.can.cabuk@ege.edu.tr)

ABSTRACT Drone technology is developing very rapidly. Flying devices accomplishing various applications are becoming an integral part of our daily life undoubtedly. Drones are characterized by extreme mobility, decent computing power, scalability, and a very short lifetime due to energy constraints. The rise of drones inevitably enabled swarms and drone networking applications. Drone networks is a path-breaking subclass of flying ad-hoc networks with unique capabilities and specific requirements. One very important challenge with swarms is the device authentication problem, in other words, proving the identity of a single or a group of drones that request to join the swarm. In this paper, we tackle this emerging problem and propose a novel context-aware mutual authentication protocol. The proposed protocol provides authentication for groups of drones and supports recovering a swarm in case of network separation. Likewise, the protocol can handle drone joins and leaves. Moreover, the protocol is not dependent on network infrastructure, secure storage, and secure channels. We tested the protocol using an automated formal security protocol verification tool, called Scyther. The tests resulted in the complete verification of the authentication and secrecy claims for arbitrary network instances and all defined use-cases. The protocol is also shown to have proven performance advantages over the existing schemes.

INDEX TERMS Authentication, drone networks, security, swarms, wireless ad-hoc networks.

I. INTRODUCTION

Nowadays, drone technology is rapidly evolving in many aspects such as battery efficiency, computational power, and cooperation capability, which altogether made drone networks possible. Although operating a fleet of drones (called swarms) does not necessarily imply networking, it is inevitably essential to accomplish complex missions consisting of challenging tasks. These missions include but are not limited to military operations, cargo delivery, disaster management (i.e., search and rescue), entertainment, imagery, construction, infrastructure (e.g., power lines) inspection, agriculture, and others [1]–[3].

Swarms with networking features, also known as drone networks, are a subclass of flying ad-hoc networks, which is a subclass of mobile ad-hoc networks. Since drones are also equipped with sensors (e.g., gyroscope, accelerometer, GPS, etc.) and actuators (e.g., rotors, robotic arms, sprinklers, weapons, etc.), drone networks may share some similarities with wireless sensor networks (WSN), but their extreme mobility, decent computation power and very short lifetimes (due to energy limitations) make a great difference.

Drones in a swarm that is conducting a critical mission may gather, store, and transmit sensitive information. In case of infiltration by any means, adversaries may leak this important information (i.e., read-only attacks), inject bogus data (i.e., read-and-write attacks), or even impair the coordination of the swarm to prevent it from accomplishing its mission [4]. Typical characteristics and unique use cases (i.e., missions) of drone networks invalidate many security countermeasures of various longer-lasting WSN setups. Hence, there is an apparent need for specific methods to secure drone networks. Fortunately, there are more than a few options, depending on the device and mission requirements.

In most cases, device authentication is a crucial prerequisite for implementing other security countermeasures, such as authorization, encryption, integrity, and non-repudiation. Authentication becomes a fundamental issue whenever the status quo of the swarm needs to be changed. This change may involve engagement of new nodes, removal of existing

The associate editor coordinating the review of this manuscript and approving it for publication was Sedat Akleylek.

TABLE 1. Comparison of some security features of CoMAD and previous works.

Literature & CoMAD		Re-authentication	Mutual Authentic.	Group Authentic.	Ad-hoc Topology	Capture-resilient	Context-aware
Han et al.	[5]	✓	✓	✓	×	✓	×
Jiang et al.	[6]	✓	✓	×	×	×	×
Rajkumar et al.	[7]	×	×	×	×	✓	✓
Turkanovic et al.	[8]	×	✓	×	×	✓	×
Semal et al.	[9]	×	✓	✓	✓	×	×
Abdallah et al.	[10]	✓	✓	×	×	✓	×
Srinivas et al.	[11]	✓	✓	×	×	✓	×
Kim & Song	[12]	✓	✓	×	×	✓	×
Wazid et al.	[13]	×	✓	×	×	✓	×
Aydin et al.	[14]	×	✓	✓	×	✓	×
Zhang et al.	[15]	×	✓	×	×	✓	×
Alladi et al.	[16]	×	✓	×	×	×	×
Cheng et al.	[17]	×	✓	✓	×	×	×
Ali et al.	[18]	✓	✓	×	×	✓	×
Bera et al.	[19]	×	✓	×	×	✓	×
Lal et al.	[20]	×	×	×	×	✓	✓
Hussain et al.	[21]	×	✓	×	×	✓	×
Lei et al.	[22]	×	×	✓	×	✓	×
Yahuza et al.	[23]	✓	✓	×	×	✓	×
CoMAD		✓	✓	✓	✓	✓	✓

nodes, or re-engagement of former nodes. Such changes may increase the network's vulnerability and allow adversaries to access the swarm's resources to some extent. However, most solutions are either tailored for wireless sensor networks with different characteristics than drone networks or designed for drone networks by means of network infrastructure (i.e., ground stations) but are not ad-hoc. Yet, to the best of our knowledge, no studies have used context information for authentication purposes in drone networks.

In this paper, we study the authentication problem for drone networks. Contributions of our paper are listed below:

1) We introduced the novel concept of "context" information as a shared secret particularly for drone networks. The context information may include mission-specific data that can only be known to the authentic members of a swarm.
2) We proposed a novel context-aware mutual authentication protocol for drone networks, called CoMAD. The protocol allows authenticating a single drone or a group of drones at once, thanks to the group authentication scheme and our novel surety concept.
3) CoMAD protocol supports recovering a swarm from incidents that cause separation of a single drone or a group of drones. It also allows adding and removing drones to/from the swarm. Moreover, CoMAD does not rely on any network infrastructure, a secure storage, or even a secure channel.
4) CoMAD protocol grants re-authentication, mutual authentication, group authentication, capture-resiliency, and context-awareness features altogether for drone networks. According to our extensive literature review, no other previous work has been found granting all these features.
5) We tested the protocol with Scyther, an automated formal model verification tool designed for testing security protocols. For numerous network instances and use-cases, the test results reveal a complete verification of the protocol's secure authentication and secrecy claims. Furthermore, an informal security analysis was also made to discuss security against passive and active attacks.

The rest of this paper is organized as follows: Section II discusses the existing literature, Section III provides assumptions, models, and other preliminaries, Section IV introduces the CoMAD protocol from an algorithmic point of view, Section V presents formal and informal security analyses, Section VI gives a brief performance analysis, and Section VII discusses the conclusions of the work.

II. RELATED WORKS

Numerous related works concern authentication schemes for mobile WSN and drone networks. However, a majority of solutions are either tailored for wireless sensor networks that have different characteristics than drone networks (e.g., weaker computation power, longer lifetime, slower mobility, etc.) or designed for drone networks that have means of network infrastructure (i.e., ground or base stations) but not are purely ad-hoc. In contrast, our proposal is uniquely tailored for ad-hoc drone networks that are deprived of ground infrastructures. Moreover, although there are a few studies suggesting use of context information for fixed networks (e.g., WSN), no previous studies have used the context information for authentication purposes in drone networks, to the best of our knowledge. A detailed comparison of CoMAD with the literature is presented in Table 1. The comparison is based on six security and networking features that are beneficial to an ad-hoc drone network: re-authentication, mutual authentication, group authentication, support for ad-hoc topologies, capture-and-tamper resiliency, and context-awareness.

Apart from the absence of a network infrastructure, what makes ad-hoc drone networks special is their flight (or hover) formations and their ability to rapidly change these formations when necessary. Numerous research has been done to identify the aeronautical characteristics of drone swarms and define specific flight formations [24]–[28]. They also studied formation splitting, merging, joining, and changing scenarios during a flight, considering obstacle avoidance, leader-follower relations, and smart reorganization algorithms. In our work, they are used to construct the connectivity restoration reasoning and mentioned later in their corresponding sections.

Harn [29] introduced the revolutionary concept of group authentications, based on Shamir's secret sharing scheme [30]. Group authentication allows granting authentication and authorization rights to a group of users or devices at once, eliminating the need for communicating each node individually. This clearly saves time and energy since it reduces the number of messages to be exchanged. Group authentication alone is a sufficient measure if all subjects willing to be authenticated are indeed authentic group members; even if they are not, the method is still useful in detecting that there are adversaries in the vicinity. In the latter case, it is necessary to proceed with authenticating the nodes individually by another method. More recent applications of the concept are presented in [17], [31], [32]. This concept is (partly) integrated into our protocol as a feature considering connectivity restoration scenarios, in which a group of disconnected drones need to join the rest of the swarm.

Rajkumar and Vayanaperumal [7] proposed a zone-based authentication scheme for WSN. Since it uses the location information of the nodes to be authenticated, it can be considered as context-aware (to a limited extent). Nevertheless, the scheme cannot be used for drone networks due to the extreme mobility of drones and swarms. Another context-aware protocol was proposed by Lal and Prathap [20], particularly for lightweight WSN applications. They integrated a, so-called, cooperative correlation coefficient into the authentication procedures. This coefficient is calculated at all participating nodes (and the server) by using the received radio signal levels, and then, sent to the neighbors. So, the neighboring nodes can identify an adversary upon detection of unusual coefficient values. However, this scheme is not suitable for ad-hoc networks since it relies on a secure server. Aydin *et al.* [14] have designed a novel authentication protocol for lightweight devices, mainly radio-frequency identification systems. Their electronic product code generation-2 (EPC-Gen2)-compatible protocol also works for groups of devices and requires minimal resources to be implemented. Nevertheless, the algorithm is not optimized for drone-specific events, such as instantaneous topology (e.g., connectivity) changes.

A vast majority of works consider the existence of ground control stations or base stations of various types [10], [11], [16]–[19], [22], [23]. Abdallah *et al.* [10] have proposed a protocol for drone networks. Their protocol provides integrity, availability, and confidentiality to some extent with low overhead and reasonable performance. However, their method relies on the use of ground control stations (as secure channel infrastructure) and the intervention of trusted authorities. The need for ground stations is clearly a limiting factor that excludes purely ad-hoc networks that are of wide use in many scenarios. Srinivas *et al.* [11] proposed an authentication protocol called TCALAS for Internet of drones (IoD) applications. Although the protocol has some proven achievements, its purpose was not to establish a secure autonomous and ad-hoc drone network. They rather focus on user-oriented authentication to get secure drone (or sensor) services via passwords and smart cards. They also heavily rely on the contribution of trusted ground stations (as gateways) and means of remote controlling. The same comments also apply to Turkanovic *et al.* [8].

CL-GAKA, an authentication protocol proposed by Semal *et al.* [9], is a rare example of such protocols that support ad-hoc networks with no ground station. Although their protocol addresses some important issues and has proven security features, there are a few major drawbacks that make it less useful for drone networks. CL-GAKA does not natively support re-authentication scenarios, does not consider capturing or tampering threats, does not benefit from context information, and requires vast computation.

Re-authentication is another important challenge within networks with node mobility. Kim and Song [12] proposed a re-authentication scheme for mobile wireless sensors in a heterogeneous sensor network. Their network model assumes the existence of an infrastructural backbone consisting of base stations and stationary cluster heads. The scheme lets already authenticated nodes be disconnected from their clusters, change location, reconnect to another cluster, and finally get re-authenticated. Comparable protocols have also been designed by Jiang *et al.* [6] and Han *et al.* [5]. These are, unfortunately, not applicable to pure ad-hoc networks due to their dependency on the means of infrastructure (e.g., fixed stations).

III. PRELIMINARIES

This section provides preliminary information, such as models, assumptions, and building blocks that define and support the CoMAD protocol.

A. NETWORK & COMMUNICATION MODEL

Within the scope of this study, a drone network represents a connected geometric undirected graph $G = (V, E)$, where V is the set of vertices (i.e., drones) with cardinality $|V| = N$, and E is the set of edges connecting them. The graph and the corresponding network are presumed to have the following additional properties:

1) The drone network is assumed to be purely ad-hoc. An infrastructure (e.g., cellular) may co-exist; however, it shall not be relied on in any means for the CoMAD protocol.

2) The network (and the drones) is expected to implement the open systems interconnection (OSI) Model, at least up to the network layer. Likewise, a TCP/IP model implementation would work fine, at least up to the network layer. Nevertheless, no specific standard (e.g., Wi-Fi, Bluetooth, etc.) is enforced.
3) The communication range r is assumed to be identical for all drones. Both spherical (3-D) and planar (2-D) coverage models can be utilized.
4) A master drone is considered within the network. It acts as a central authority and manages the rest of the network, mostly from a security perspective. The master-ship can be handed down to another drone when necessary (e.g., battery shortage).
5) Possible anomalies in data transmission (e.g., packet loss, delayed/unsorted delivery, corruptions) are assumed to be addressed in lower layers already.
6) As a result of failures, attacks, environmental conditions, or mission-specific necessities, the network may be broken into two or more partitions. This disconnects the network, but the separated groups may stay connected within themselves and continue the operation. Later, these groups may join the network body that is an internally connected partition of a disconnected network in which the master drone operates. Occasionally, the master drone may be the network body on its own when there are no neighbors left in its vicinity.

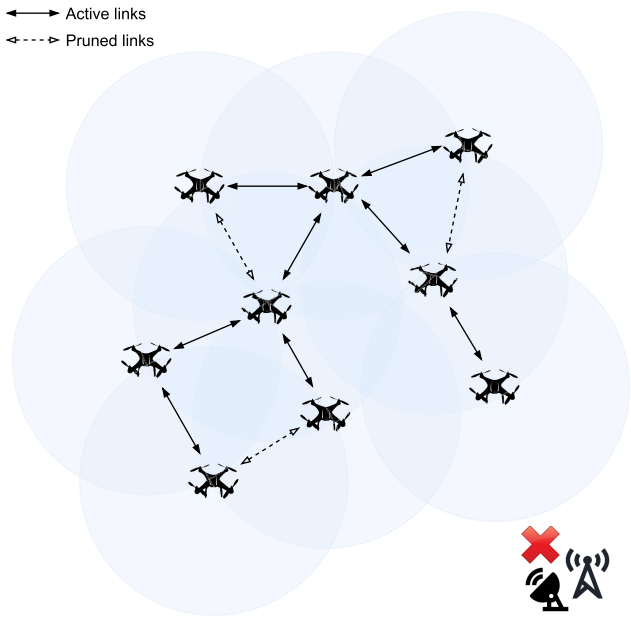

FIGURE 1. An example ad-hoc drone network with 9 drones and no ground stations, using 2-D coverage model.

Figure 1 shows a demonstration of an example ad-hoc drone network with 9 drones considering the given properties. Blue disks represent the communication radius, solid arrows represent the established links, and the dashed arrows stand for possible links but are not used within the topology (which is an arbitrary tree). The ground stations have no connection to the drones.

B. THREAT & ADVERSARY MODEL

During the design of the CoMAD protocol, the following assumptions are considered to achieve wider applicability, reduce the dependencies, and limit the extent of the study:

1) There is no external secure channel infrastructure (e.g., cellular coverage, base stations, etc.) in the mission field.
2) There is no tamper-resistant hardware (e.g., secure elements, subscriber identity module cards, etc.) on the drones.
3) Drones are considered safe and secure when flying (above ground level) as long as they are connected to the network body (i.e., the master) unless they are captured at the ground level (before or just after take-off, or after falling to the ground due to an incident). The encryption is also assumed as secure.

For the adversary capabilities, an extended variant of the widely accepted Dolev-Yao Model [33] is assumed, so that:

1) An adversary can intercept any message sent between the drones in the swarm.
2) An adversary can temporarily interrupt parts of the communication on the channel.
3) An adversary may transmit any message to any of the reachable drones and can cast customized (e.g., impersonated) messages.
4) An adversary may utilize cryptographic functions or operations, including random number generators, Boolean logic, and encryption. It may have access to some private data (e.g., device IDs, mission IDs, etc.), but it does not have access to secret keys and context information.
5) An adversary can physically capture and tamper the drones under certain conditions (i.e., on ground-level).

Throughout this study, the following malicious activities are considered as threats to be mitigated by adopting CoMAD protocol:

1) INFILTRATION

Infiltration by impersonation is a prevalent form of infiltration-oriented attacks. A malevolent drone (either autonomous or remote-controlled) imitates a legitimate drone to some extent by using its credentials and premeditatedly tries to get authenticated in order to join the swarm. A successful attempt may result in the failure of the mission. Moreover, even sabotage, suicide attacks, espionage, and hijacking might be inevitable. Routing attacks [34] also become possible. Hence, this threat is vital and has to be addressed at any cost for almost every critical mission.

CoMAD is designed to prevent infiltration by impersonation. Other means of infiltration (i.e., the impact of non-authenticated parties) cannot solely be mitigated by authentication protocols but require countermeasures, such as

confidentiality (e.g., encryption) and integrity mechanisms. Therefore, after successful authentication (of a node) occurs, secure communication is assumed to be established in the rest of their operations.

2) EAVESDROPPING

Although the confidentiality of the data communications within the swarm is left out of this work's scope, there is still a possibility to eavesdrop on some important information (e.g., encryption keys) during authentication procedures. However, the protocol is designated to prevent the leakage of any sensitive information. The use of provably secure encryption algorithms during the initial requests and exchanges prevents unauthorized third parties from leaking keys and secrets. Furthermore, another consequence of eavesdropping is the possibility of replay attacks that involve resending copies of the captured legitimate messages. This type of attack does not even require reading the encrypted information within the captured messages. Such attacks may be prevented by wise use of random nonces, as we do in the protocol.

3) TAMPERING

This work assumes that the drones are safe and cannot be compromised during their flight (since they communicate through encrypted communication as in [33]). However, they can still be captured if they land during a mission, either intentionally or accidentally. In this case, captured drones may be subject to adversary inspection and even tampering to resolve sensitive information, including secret keys, context information, or other mission-specific data. Such leakage may be extremely critical since the entire network will be at risk, as well as the mission itself. Tamper-proof hardware (e.g., secure elements) may work well, yet our work also assumes that they are not required in the drones (considering the limited availability of such hardware for drones).

4) AVAILABILITY ATTACKS

Extensive use of a jammer device near the swarm may disconnect the portions of the network or its entirety. Although this is mostly unpreventable (especially when the source is not observable), a swarm of drones may have a predetermined strategy to avoid the disrupting signals. For example, the drones may proceed to a predefined rendezvous point or a previous checkpoint to escape from the trap. More of such strategies on connectivity restoration can be found in [35]–[38]. As a fine-tuned variant, some availability attacks solely draw a bead on nodes with critical functionalities by detecting them via means of traffic analysis; however, this is rarely feasible in highly mobile networks. CoMAD provides some measures against attacks targeting the availability of the network (explained in Section V-A7).

C. INCIDENTS

Within the scope of this study, incidents are defined as either unexpected (e.g., accidents, faults, outage, etc.) or planned (e.g., bifurcation, partial expiry, etc.) events that may result in the separation of some nodes from the network body, or the addition of some nodes to the network body during an active mission. The nodes to be added to the network may either be totally new drones that were not part of the swarm previously or former members that were once a member of the swarm (but separated due to any reason at any time).

TABLE 2. List of recognized incidents by the protocol.

Incident	Impact	Action
Normal operation	No impact	Routine messaging
Separated solo join	Add node	Re-authentication
New solo join	Add node	Authentication
Solo leave	Remove node	De-authentication
Separated group join	Add node	Group authentication
New group join	Add node	Group authentication
Master delegation	No impact	Delegation

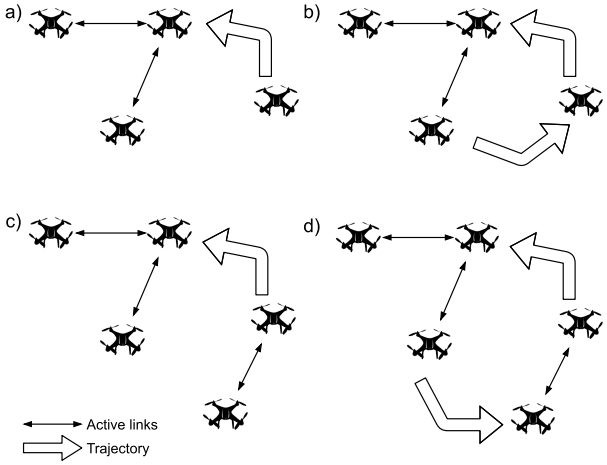

FIGURE 2. Illustrations of a) New solo join, b) Separated solo join, c) New group join, and d) Separated group join incidents.

The incidents that the protocol recognizes (and responds) are listed in Table 2 and the ones that involve an increase in drone numbers are illustrated in Figure 2. Each incident requires some action and may cause security threats to some extent, which are already given. The protocol does not recognize a "*Group leave*" incident because the nodes that have to leave the swarm intentionally can be trivially assumed to leave individually. "*Group replacement*" does not exist either, for similar reasons. Individual leaves may disconnect the network; however, the swarm is assumed to have enough time to take the necessary actions (e.g., changing the topology) since the leave is already reported beforehand.

By the way, the CoMAD protocol can also be used flawlessly in stationary networks (e.g., WSN), especially if there is a possibility of temporarily losing (and adding) some nodes. However, this would likely be overkill since the protocol has specific measures to address mobility-related incidents which were not needed in a typical WSN. Moreover, a self-restoring topology formation requires advanced means of mobility as in drone swarms.

TABLE 3. Predefined list and definitions of the roles that drones in the ecosystem may have.

Persona Type	Definition	Validity	Supersets
Master	Mere leader (i.e., root or sink) of the entire network body.	Lifetime or until delegation	None
Member	Drones included in the network body.	Lifetime or until an incident	None
Non-member	Drones not included in the network body, or separated once.	Lifetime or until membership	None
Ex-member	Drones that are once included, but separated at any point.	Lifetime or until membership	Non-member
Claimant	Non-member drones that are requesting to be members.	During authentication	Ex-member or non-member
Fraudulent	Adversary drones that try to infiltrate into network body.	Lifetime	Non-member
Group Proxy	Leader of a temporary group formed upon separation.	Lifetime or until membership	Ex-member
Group Client	Member of a temporary group formed upon separation.	Lifetime or until membership	Ex-member
Vice-master	Backup for the master. Not active unless master disconnects.	Lifetime or until renewal	Member

IV. CoMAD PROTOCOL

In this section, we introduce the CoMAD protocol we propose. We will explain unique message types, defined persona (role) types, our understanding of context information, designed security policies, and initialization operations, maintaining normal state, solo and group join events.

A. PERSONA TYPES

From the protocol's point of view, a persona is a self-declared, appointed, or assumed role for any drone existing in the ecosystem. The conception of persona is created to assign certain tasks and procedures to the corresponding drones. All predefined personas are listed in Table 3. The drones themselves are aware of their personas to a wide extent. However, a persona is not a static property; it can change after certain events. For example, in the case of battery outage, the master drone may delegate another drone to be the master and continue the operation as a member (or non-member if leaving).

B. CONTEXT

Drone and swarm operations are usually called missions. Missions consist of a finite set of consecutive and concurrent tasks that aim to accomplish clearly defined objectives while complying with strict boundaries. During these missions, drones may (or may not) react to some events, depending on their program and the essence of the mission.

Context, in a cryptographic sense, is any mission-specific up-to-date information that may help to identify the nodes which are holding the whole or portions of it [39]. The context information must be recent, accurate, and complete. Although sharing the newest information with all nodes is not necessary (nor feasibly possible), outdated information can more easily be revealed or estimated. Likewise, inaccurate or incomplete data are more prone to be found out. Context-aware systems usually need at least one context server that provides relevant context data [40]. A context-server should also keep track of older context data; because some disconnected nodes may not have the most recent context.

As long as the context information is carefully considered, it provides another good layer of security; otherwise, its contribution stays limited to the security by obscurity. For a drone network, some good examples of context information are given below:

- Topology and network-specific configuration, (e.g., complete or partial adjacency matrix of the network, complete or partial routing table, network performance indicators, etc.).
- Flight-specific records (e.g., average flight altitude at a moment, average swarm velocity, etc.).
- Formation-specific records (e.g., the center of mass of the flight formation, etc.).
- Mission-specific information (e.g., visited checkpoints, tracked objects, etc.).
- Sensor data (e.g., temperature, pressure, etc.).

Many more factors can be extracted depending on the mission details. Context information must be invisible, unobservable, and unpredictable for any external adversary. Dissemination of more than one context factor is highly recommended when possible. In the case of multiple context factors, portions of the context information may be hashed, concatenated, or aggregated (e.g., *XOR*'ed), while others may be stored raw and separately. If the context information is a piece of critical mission-specific information, which is very likely, then it must be hashed. So that a context parameter should be thought of as:

$$C = H(C^1 \| C^2 \| \dots \| C^n \| MID) \tag{1}$$

where C^m represents different context parameters of number m, H is an appropriate hash function (as explained in Section IV-D), and '$\|$' denote concatenation. Concatenating the mission ID (*MID*) is also beneficial if there are parallel missions within the field. As a side note, while having more factors increases the system's security significantly, it may also increase the operating costs.

C. MESSAGE & DATA TYPES

The protocol defines unique message types to operate. A detailed list of the designated messages is given in Table 4. Each of these messages contains a combination of unique data types. A list of designated data types is given in Table 5. These data types are not strictly defined in bit-level within the scope of the protocol; however, industry standards must be followed, when applicable, such as [41]. Recommended minimums for encryption and hashing functions are mentioned in Section IV-D.

TABLE 4. List of predefined message types.

Abbreviation	Name	Definition	Issued By	Sent To
AuthReq	Authentication Request	Initial request of membership.	Claimant	Master
AuthGrant	Authentication Grant	Notice of approval of membership.	Master	Claimant
AuthFail	Authentication Failure	Notice of rejection of membership.	Master	Claimant
DeAuth	De-authentication Order	Notice of termination of membership.	Master	Member
Chal	Challenge	Call-out for proving authenticity.	Master	Claimant
ChalResp	Challenge Response	Attempt for proving authenticity.	Claimant	Master
Leave	Leave Notification	Notice of intentional leave.	Member	Master
Surety	Surety Bond	List of trusted non-members.	Group Proxy	Master
BookB	Book Building	Master delegation notice.	Master	Members
Tender	Tender Bid	Request for master delegation.	Members (Eligible)	Master
Handover	Handover Delivery	Transfer of master-ship data.	(Current) Master	Member (Next Master)
Dissolve	Group Dissolution	Notice of dissolution of group.	Group Client	Group Proxy

TABLE 5. List of data types and parameters used in the protocol.

Symbol	Name	Definition
ID_i	Device ID	Unique device identifier of node i.
MID	Mission ID	Unique mission identifier.
N_i^j	Nonce	Random number casted by node i for j.
K_M	Master key	Symmetric secret key used in session key.
K_S	Session key	Symmetric shared key used in encryption.
K_i^+	Public key	Public key of node i.
K_i^-	Private key	Private key of node i.
T	Timestamp	Time indicator for a transaction.
C_i	Context	Shared context information at node i.
CR	Context req.	Request for a context message.
L	Surety List	List of trusted non-members.
I	Intention	Intention of the issuing party.
t_l	Key threshold	Time period for key renewal after leave.
t_e	Exp. threshold	Time period for routine key renewal.
$E(K, X)$	Encryption	Encryption of data X using key K.
$H(X)$	Hash	Hash of data X over a one-way function.
$DATA$	Data	Generic data, referred to with subscripts.

D. KEY & ENCRYPTION POLICIES

Throughout any phase of the execution of the protocol, it is essential to keep the communication encrypted. There is no need for clear-text transmission at any point. Since the entire swarm is initialized at a trusted ground station, all drones shall have some keys pre-installed. A brief list of the keys is given in Table 5. The public-private key pairs (K_i^+ and K_i^-) are used in the first contact or during an ongoing authentication procedure. In addition to proving the authenticity of a claimant (and the master), it is also used for delivering the session key (K_S) later. K_i^+ and K_i^- are not used after the authentication phase due to performance concerns. Regular data and signaling communications are always done encrypted with K_S within the swarm.

The key K_S must be changed whenever at least one member drone leaves (i.e., disconnects) the swarm for any reason or at least one non-member joins. It is done to ensure the key freshness [42]. However, a short (i.e., up to a few minutes) time threshold (t_l) must be waited for this key renewal to overcome the disorganization and the extra message traffic caused by momentarily disconnections. The same t_l also applies to joins so that the joining drone receives the current (old) K_S and then the entire swarm renews their keys. K_S can also be renewed periodically after some predefined expiry period (t_e). In general, drones already have very limited lifetimes (e.g., from a few minutes up to a few hours); thus, frequent key renewal is not mandatory.

If periodical key renewal is considered, the master issues a new K_S whenever the current one expires (per to t_e) and then disseminates it to the entire network via a flooding-like mechanism. A specific flood algorithm is not enforced. The session key K_S is generated by hashing the pre-installed master key (K_M), a hash of the most current context information ($H(C)$), and a freshly generated nonce (N) as follows:

$$K_S = H(K_M \oplus N \oplus H(C)) \tag{2}$$

where '$\oplus$' implies a binary *XOR* operation. There is no enforcement on the length of the context information. This is why it is hashed before used in the generation of K_S. When it comes to encryption, the protocol is not algorithm-dependent. However, as an insisting recommendation, all symmetric encryptions should be done with advanced encryption standard (AES) over at least 128-bit keys, which implies that K_M and K_S are of 128-bit size, at a minimum. On the other hand, asymmetric encryptions can be made with Rivest-Shamir-Adleman (RSA) scheme over (at least) 3072-bit keys, or alternatively, with elliptic curve cryptography (ECC) methods over (at least) 256-bit keys. So that K^+ and K^- values are (at least) 3072-bit long (RSA case) or 256-bit long (ECC case). For hashing purposes, the secure hash algorithm (SHA) version 2 or 3 with (at least) 256-bit output sizes are considered for a collision-resistant solution. A concise report on which key sizes are accepted as secure can be found in [43].

E. INITIALIZATION & NORMAL OPERATIONS

The protocol is mostly executed whenever an event that involves means of authentication happens during a mission. However, to provide the projected functionalities, as hinted previously in Table 1, there are some procedures that must be done prior to the (first) take off for a mission (i.e., in the initialization phase) or maintained during normal operation (i.e., when there is no authentication-related event).

1) INITIALIZATION

Before starting any mission, the entire swarm is assumed to be kept at a secure place (e.g., ground station, watercraft, airplane, etc.) and is under the control of legitimate (human) operators. The operators can install mission-specific information, set the network topology, and configure the drones accordingly. One of the drones (not necessarily a special one) has to be assigned the root (i.e., master) role, and the others are registered as members.

In that phase, all drones must have their unique device IDs (ID_i), mission IDs (MID), and public-private keys pairs (K_i^+, K_i^-) installed. In addition, the master drone must store the master key (K_M) and the public keys (K_i^+) of all existing drones (including the reserved ones that are not initially connected to the swarm). The swarm is assumed to be set as a single network piece with no partitions.

2) NORMAL OPERATION

Normal operation includes any moment during a mission where there is no authentication-related operation (e.g., request, response, etc.) is carried out. Whenever an authentication request is made by a non-member, normal operation is virtually suspended until the authentication process is completed (either by acceptance or rejection of the request). Then, the normal operation continues.

In this phase, all drones periodically synchronize some context information (C) within the swarm. This information must be disseminated from the master to all other drones in the network body with no exception. This context information may be originated from the master or may have been an end product of some computation made using data collected from the member drones. While there is no specific rule on the synchronization period, it is a mission-specific parameter and is up to the operators, although a more recent context would provide better security.

In addition to the context dissemination, the master drone also assigns and announces a vice-master drone for backup purposes in case the master fails at some point during the mission. The vice-master receives sensitive information (i.e., latest context and the key repository) from the master, but has no specific role nor authority, unless the master fails, leaves the swarm, or delegates the vice-master as the new master. This backup selection can be made using the remaining battery level or a topology-specific parameter (like the node degree, etc). The vice-master assignment is not static but is subject to change in every round, if necessary.

F. SOLO JOIN

Letting a drone join the network body is an important responsibility that must be carefully handled since there is a significant risk of infiltration of a foreign or fraudulent drone. Moreover, a drone may have also been captured and tampered and even taken into control by adversary operators during the time period it is disconnected. Some joining scenarios for swarms are analyzed comprehensively in [44], [45].

The solo node join incidents cover two cases: join of a separated drone that was once connected to the network body or join of a new drone that was never part of the network body before. These situations are reported to the master drone in the intention data (I) within the *AuthReq* message during the first step of the authentication. Both cases are handled very similarly; the main difference is the availability of context information. So that, while a separated drone has some context information (although it may be outdated), a new drone would not have any. In this case, the context information (C) in the claimant is replaced by an initialization vector (IV) that is found by the below equation:

$$C = IV = H(K_i^+ \| ID \| MID) \tag{3}$$

where H is the chosen hash function. The IV-based context parameter is handled accordingly by the master during the authentication. However, the swarm inevitably cannot benefit from the exchange of context information in this situation.

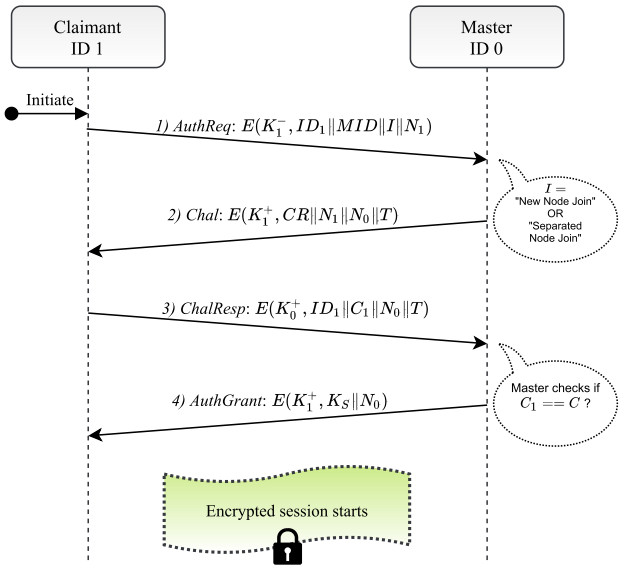

FIGURE 3. Example message sequence chart for solo join (with success), where a claimant drone requests to join the swarm.

An example message sequence chart for the node joining procedures is given in Figure 3, where successful authentication is demonstrated. In the *AuthReq* message sent from the claimant, if any of the data fields *ID*, *MID*, and *I* are missing or inappropriate (e.g., unidentified *ID*, wrong *MID*, or empty *I* etc.), the master then transmits a *AuthFail* message instead of the *Chal* message, in the second step. This permanently keeps the claimant out. Likewise, a wrong, missing, or inappropriate data in any of the fields C, N and T in the *ChalResp* message shall result in strict rejection of the authentication request by an *AuthFail* message in the fourth step, instead of the *AuthGrant* message. This sub-protocol is depicted in Figure 3 and also coded in security protocol description language (SPDL) at `(akademik.ube.ege.edu.tr/netos/source/CoMAD.txt)`, where it is labeled *SoloJoin*.

G. GROUP JOIN

A group join event is used to reduce the complexity and increase the efficiency when a group of members of the network (e.g., a cluster, a branch, etc.) lose the connection together and intend to reconnect to the network body later. So that, instead of authenticating each node one-by-one, it is more beneficial to authenticate the entire group at once, when possible. Various group merging/joining scenarios for aerial vehicles are explained in detail in [45]. A separated group can still sustain the encrypted communication within the group, using the most current session key (K_S, particularly denoted as K_{S-old} in the implementation). However, they lose the ability to renew their session keys since they can only be generated by the master.

The group delegates a temporary group leader, called the group proxy, in a distributed fashion. This delegation is a straightforward procedure without further competition. The selection can be based on ID numbers, remaining battery levels, or another distinctive feature of choice. A broadcast (and flooding) mechanism is required to exchange the corresponding information and to make a decision. After the group is formed (i.e., a group proxy is selected), the group clients send unique nonces (N_i^J) to the proxy.

The group proxy has crucial responsibilities. In case of separation of a group of drones, the group proxy (*i*) initiates a group authentication procedure (as in [29]), (*ii*) issues a *Surety* message containing a list (L) of authenticated group clients, and (*iii*) carries out an authentication procedure with the master as shown in Figure 4, resulting in the authentication of the entire group at once (in a successful attempt). If the authentication fails, only the group proxy is marked as fraudulent and banned from further actions (e.g., added in a blacklist), yet the group clients can initiate a solo join procedure afterward (in which they have merely one attempt right). Figure 4 presents the details of the group joining sub-protocol (except for the initial group authentication and the group clients' nonce exchange). There is also an SPDL implementation provided at (`akademik.ube.ege.edu.tr/netos/source/CoMAD.txt`), with the label *GroupJoin*.

Once a group is formed, another drone cannot join the group later. Likewise, when there are more than one separated groups (not knowing each other), they cannot join together even if they meet at a later phase. We do not foresee any benefit from such mergers. Otherwise, it may create some security risks. Furthermore, the group joining procedure is only for separated groups; a new group join mechanism is not considered. New drones must be joined individually as in solo join.

H. SOLO LEAVE & REPLACEMENT

If a drone unexpectedly and unwillingly leaves the swarm (e.g., due to a failure, accident, attack, etc.), there is naturally no possibility of further communication. Not only that, but if this drone is a cut-vertex, then its absence will also disconnect one or more drones as well (more on cut-vertex detection can

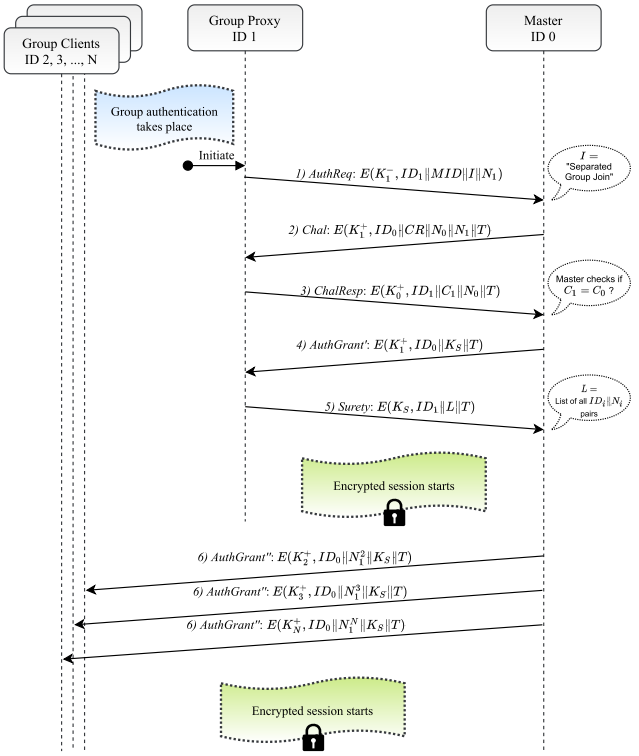

FIGURE 4. Example message sequence chart for separated group join (with success), where a group of (group-authenticated) ex-members request to join the swarm through a group proxy.

be found elsewhere [46]). In this case, all these disconnected drones must follow the "single node join" or the "group join" procedures from scratch unless they manage to join the network body within the K_S renewal threshold t_e.

On the other hand, a drone may also leave the network on purpose. This may be due to a separated patrolling task or for a replacement procedure (e.g., in case of low battery levels, etc.). Such node removals are explained in detail in [47]. In this case, the leaving drone essentially has three options: (*i*) leave the network for good, (*ii*) leave the network for a pre-estimated time period, or (*iii*) leaving the network for replacement purposes. In the first case, the leaving drone sends a *Leave* message with a leave threshold $t_l = 0$ to the master. The master interprets this as an immediate leave and sends a *DeAuth* message in response. At this point, the leaving drone must follow the regular "solo join" procedure if it requires reconnecting later. In the second case, the threshold t_l in *Leave* is set to an arbitrary number indicating the time period of the disconnected phase. The master then keeps the session key K_S alive either for the entire network or only for the leaving node (depending on the scenario requirements), until t_l is exceeded (assuming t_l is larger than t_e). The master confirms the request with a *DeAuth* message as in case (*i*). In case (*ii*), if the leaving node does not return within t_l, then it must follow a "solo join" procedure later. In the third case, the leaving drone also sends a list L containing the *ID*s and public keys of drone(s) to be attending as substitution (as in the *Surety* message), within the *Leave* message. This is only

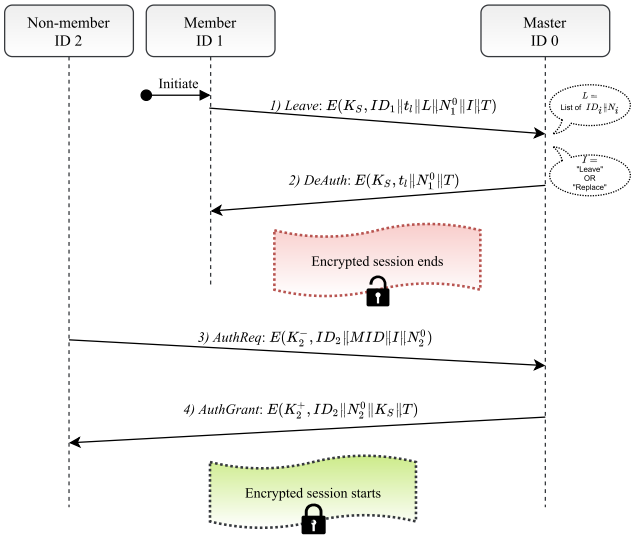

FIGURE 5. Example message sequence chart for leave and replace (with success), where a member drone is substituted with a non-member.

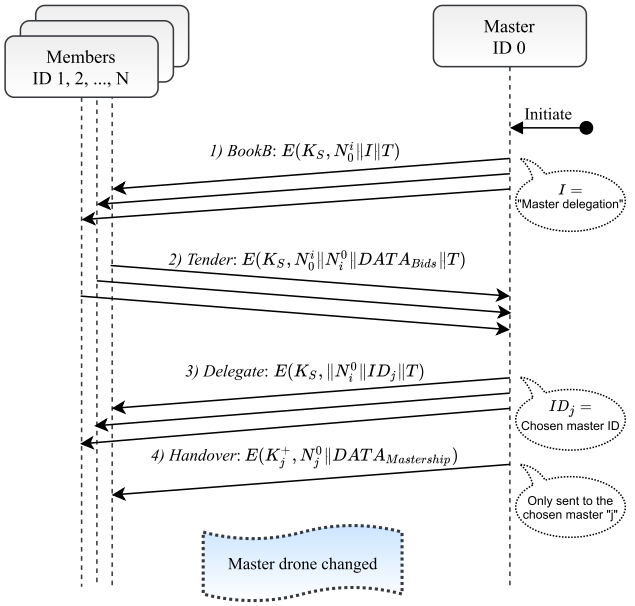

FIGURE 6. Example message sequence chart for master delegation (with success), where a current master drone is abdicated.

possible if such a list is pre-installed on that drone during the initialization phases. When the substitute drones approach and send an *AuthReq* message, the master directly sends an *AuthGrant* message, as long as they provide valid *ID*, *MID*, and *I* information. A leave and replace procedure is depicted in Figure 5. As a side note, leaving drones may wipe context information and mission ID if they detect unexpected landing to protect the secrecy against the possibility of being captured (explained in Section V-A8).

I. MASTER DELEGATION

Although it is not essential, the master drone may be subject to change when required. Technical issues, battery problems, formation splitting strategies, or other mission-specific conditions may enforce such a change. In this case, the current master drone delegates a sufficiently capable member drone (e.g., has an adequate battery, possesses relevant equipment, is not cut-vertex, etc.), and transfers all the relevant information that a master drone must possess (as stated in Section IV-E1) to the delegated one. During this delegation, the master acts as a book-runner and broadcasts a *BookB* message for a ''book building'' purpose. In response, candidate drones (i.e., drones that are eligible) send *Tender* messages containing their degree of suitability (e.g., battery levels, performance metrics, etc.). The master evaluates the bids, and chooses one of the bidders, delegates it, and announces this handover to the entire swarm. The new master immediately assigns a backup via messages similar to *Delegate* and *Handover*. The procedure is demonstrated in Figure 6.

V. SECURITY ANALYSIS

The security features of the CoMAD protocol are analyzed comprehensively using both formal and informal approaches, which together suggest a strong sense of security.

A. INFORMAL ANALYSIS

This section explains the countermeasures implemented for the given passive and active attack types, which are carefully chosen considering their relation with the introduced threats in Section III-B. The first subsection reveals our measures against two major passive attacks, and the remaining subsections elaborate the corresponding active attacks.

1) PASSIVE ATTACKS

''Passive attacks'' is an umbrella term that includes an adversary's efforts on acquiring meaningful information by observing the communicating parties and/or listening the communication in between. A swarm that uses the CoMAD protocol does not leak any sensitive information during its normal operations and the authentication procedures. Because, all the communication links are securely encrypted and there is no ''clear-text'' messaging. The encryption is done with a symmetric session key (K_S) during the normal operation and a public-private key pair during the authentication procedures. Moreover, the need for providing valid context information for authentication purposes prevent the drones that were once legitimate members but compromised later from re-attending the swarm.

An adversary's another option would be capturing the entire message traffic of the swarm for a sufficiently long period of time and making a traffic analysis to determine the master drone or any specific drone (which may become a single-point-of-failure for the swarm). However this is highly impractical due to the following reasons: (*i*) a drone swarm presumably has a very high degree of mobility, hence, the master drone (as well as others) can be located anywhere as there is no enforcement on their location within

the flight formation, (*ii*) the communication to the master drone is always encrypted, and (*iii*) the master drone changes periodically after a time threshold. Moreover, a master is capable of informing its leave as long as it is alive. In case the swarm suddenly loses its connection to the master entirely, the member drones either quit the mission and return to the base immediately or cancel the mission but continue other consecutive missions, if any.

2) IMPERSONATION

An adversary may intend to impersonate a legitimate member to infiltrate the network and leak data or interrupt the mission. However, observations are not sufficient to mimic a legitimate member. An adversary (as well as authentic drones) is requested to provide numerous secret parameters for authentication purposes. Since this is a mainstream type of attack, the protocol has multiple factors and measures. So that, a drone must pose a valid (*i*) mission ID *MID*, (*ii*) intention I, (*iii*) public-private key pair $K^+ : K^-$, and (*iv*) context information C. Additionally, (*v*) it must respond in time per to a timestamp T. In solo and group join events, it is not reasonably possible for an external adversary to maintain all these five measures (e.g., a secret key). Even an adversary of partial accordance with these confidential parameters can be isolated unless it provides all of them. This claim is guaranteed per to assumptions given in Section III-A, III-B, and IV-D. Hence, it is clear that the real threat is the *Ex-members* of the network. This is where handling the context information gets its importance.

3) REPLAY

An adversary in the vicinity of a swarm can replay some of the authentication messages that it sniffed from the air medium [48]. This may result in the failure of an ongoing authentication process of legitimate nodes or even allow adversary nodes to get authenticated using the credentials in the replayed messages unless there are effective countermeasures. In CoMAD, there are two such preventions: Nonces and timestamps. Fresh nonce values are transmitted by both parties of an authentication process, in their initiating messages and the corresponding responses, in all sub-protocols. In this way, both parties ensure that the other party is, indeed, alive and not replaying previously sniffed messages. Timestamps can also provide similar functionality since any outdated message can easily be detected and so discarded. However, we only implemented timestamps to the master drones in order to avoid handling potential time synchronization issues. Yet, the protocol can still be extended by adding timestamps to the non-master drones if time-synchronization can be guaranteed.

4) MIRRORING

The mirroring attack is a special case of replay attacks, where an adversary may capture a (generally initiating) message of an arbitrary source node and then resends it back to the source so that it mirrors the message. Wherever encrypted communication takes place, this attack does not necessarily involve leakage of sensitive information; however, it still may break how the protocols may behave. In CoMAD, all initiator messages, namely *AuthReq, BookB* and *Leave* contain the encrypted ID of the source node in the payload, which essentially allows the recipient of this message to consider the source. When a recipient encounters a message that is apparently, originated from itself, it simply discards it. Moreover, using a public key encryption scheme for symmetric key exchange in CoMAD also helps mitigating mirroring attacks. Additional functionalities on detecting the adversary may also be implemented, as in [49].

5) MAN-IN-THE-MIDDLE

The man-in-the-middle attack is a poisonous scenario, where an adversary virtually stays in between two communicating parties who believe that they are unmediatedly communicating and thus, has access to the transmitted messages (i.e., eavesdropping). The adversary can alter, send, or discard the messages it intercepted. This is essentially an extended and live variant of the replay attacks. CoMAD prevents such attacks by (*i*) encrypting the initiator messages (*AuthReq, BookB* and *Leave*), (*ii*) applying public key cryptography to the rest of the messages (for solo and group join scenarios), and (*iii*) using fresh nonces and timestamps. Hence, a man-in-the-middle has nothing to do with the intercepted messages.

6) BRUTE-FORCE

An adversary may intend to infiltrate the network or simply eavesdrop on the ongoing communication by brute-forcing the keys used in the encryption processes. If a key is compromised, then no part of the communication can be assumed secure. To provide flexibility, the protocol does not enforce any specific encryption algorithm. Nevertheless, unless there is a good reason to consider, all symmetric encryptions should be carried out with AES-256, in which K_S is the 256-bit session key, and asymmetric encryptions should be done through RSA-2048 (or better). Both AES and RSA are still considered secure against brute-force attacks when used with appropriate key sizes. Drones' limited battery lifetime is another factor that obstructs such attacks since there will not likely be enough time to complete the attack at all.

7) DENIAL OF SERVICE

Attacks aiming to degrade the availability of a swarm, also known as denial of service (DoS) attacks, may be either directly in the physical layer or above levels, such as data link, network, etc. [50]. Attacks aiming at the physical layer (e.g., signal jamming) are major issues and cannot be prevented by such authentication protocols. In case of a jamming attack that disconnects portions of the network, connectivity restoration solutions can be applied (as in [35]–[38]), the authentication protocol can be initiated once the drones recover from the attack. On the other hand, flooding replayed, counterfeit or void authentication messages aim at the availability of the network on higher layers. The protocol does the

following to counter such attacks: (*i*) it discards any repeating message from the same source, (*ii*) it bans the drones that are failed to authenticate due to wrong credentials or invalid context.

8) CAPTURING & TAMPERING

Physical attacks are not limited to the ones that cause device and network outages. A drone can easily be captured, especially at or near the ground level. A captured drone can be tampered with to leak important information stored locally, including encryption keys, as well as confidential mission data. CoMAD enforces a strict data-wipe policy that requires any live drone that falls or lands to the ground unintentionally and unreportedly to completely wipe all important information, including at least, symmetric encryption keys (K_M, K_S), secret key (K^-), mission ID (*MID*), context information (C) and all other critical information if any. Removing data from a memory unit is another vital issue since it is occasionally possible to retrieve parts of the removed data, as proven in [51]. For electrically erasable programmable read-only memory (EEPROM) and more contemporary flash memory devices, [51] suggests padding the memory with random bits at least ten times before removing. Since this is only a matter of milliseconds, we adopt the same procedure in the protocol with a suitable pseudo-random number generator as in [52]. Furthermore, it is also impractical to physically locate the master drone and any other drone as already discussed in Section V-A1. Therefore, there is protection against the capturing and tampering attacks.

B. FORMAL ANALYSIS

We have conducted formal verification analyses on our protocol by model checking using Scyther (v.1.1.3), an automatic formal security protocol verification tool [53], [54]. Its use for such purposes is widely accepted in the literature [9], [11], [55]. The CoMAD protocol is implemented as a collection of three sub-protocols (i.e., *SoloJoin*, *GroupJoin*, and *Delegate*) in security protocol description language (SPDL). The *LeaveReplace* procedure is not tested separately since it is straightforward and already occurs in the encrypted space. The tests are conducted without applying any bounds.

The complete test results are presented in Table 6 and the codes for the SPDL implementations are provided at (`akademik.ube.ege.edu.tr/netos/source/CoMAD.txt`). The adversary model presumed by the Scyther tool is based on the Dolev-Yao model, which is mostly sufficient per our assumptions (except for the tampering attacks), and it also assumes perfect cryptography. The protocol implementations are role-based and rely on corresponding claims for verification purposes. These claims may indicate different levels of authentication, as aliveness, weak agreement, non-injective agreement, non-injective synchronization, agreement on secrets (incl. keys), and agreement on data. Formal and informal definitions of these levels are provided in [54]–[56].

TABLE 6. Scyther formal verification test results.

Protocol	Role	Claim	Status
SoloJoin	Claimant	Alive	OK | Verified
		Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		Secret C_c	OK | Verified
		SKR K_s	OK | Verified
		Commit Master, $N_c, N_m, MID\text{-}I$	OK | Verified
	Master	Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		Secret C_m	OK | Verified
		SKR K_s	OK | Verified
		Commit Claimant, $N_c, N_m, MID\text{-}I$	OK | Verified
GroupJoin	Proxy	Alive	OK | Verified
		Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		Secret N_{gc}	OK | Verified
		SKR K_s	OK | Verified
		Commit Master, N_m, N_{gp}	OK | Verified
	Master	Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		Secret N_{gc}	OK | Verified
		SKR K_s	OK | Verified
		Commit Proxy, N_m, N_{gp}	OK | Verified
	Client	Alive	OK | Verified
		Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		Secret N_{gm}	OK | Verified
		Secret N_{gc}	OK | Verified
		SKR K_s	OK | Verified
Delegate	Master	Alive	OK | Verified
		Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		SKR K_s	OK | Verified
		Secret $DATA_b$	OK | Verified
		Secret $DATA_m$	OK | Verified
		Commit Member, $N_c, N_m, MID\text{-}I$	OK | Verified
	Member	Weakagree	OK | Verified
		Niagree	OK | Verified
		Nisynch	OK | Verified
		Secret $DATA_m$	OK | Verified
		Commit Master, $N_c, N_m, MID\text{-}I$	OK | Verified

Some implementation and configuration details are as follows: the maximum number of rounds is set at '0' (i.e., no bounds), the matching type is 'find all type flaws', search pruning is 'find all attacks'. Additionally, routine context exchanges occurring during the normal operation are not explicitly implemented, yet C is presumed as a shared secret and is not implemented due to Scyther's limitations. In the code, the subscripts '$_m$', '$_c$', '$_{gp}$', '$_{gc}$' stand for master, claimant, group proxy, and group client, consecutively. In *GroupJoin*, there are two additional messages sent from

TABLE 7. Performance analyses of CoMAD and CL-GAKA (Semal *et al.* [9]). *n* is the number of nodes in the network or the ones involve in an authentication process (i.e., group size in group join for CoMAD).

Protocol		Time Complexity	Message Complexity	Nr. of hash operation	Nr. of asym. encryption	Nr. of sym. encryption
Semal et al. [9]		$O(n)$	$O(n)$	$O(n^2)$	$O(n)$	$O(n)$
CoMAD	*Solo Join*	$O(1)$	$O(1)$	0 (or 1)	$O(1)$	0
	Group Join	$O(1)$	$O(n)$	0	$O(n)$	$O(n)$
	Leave & Replace	$O(1)$	$O(1)$	0	$O(1)$	$O(1)$
	Master Delegation	$O(1)$	$O(n)$	0	$O(1)$	$O(n)$

the group client to the proxy and the master at the end of the protocol (numbered 7 and 8); these are optional and added to complete the cyclic scheme to be verified by Scyther.

The group authentication scheme that takes place in *GroupJoin* sub-protocol is, intentionally, not implemented for verification. Because there are already some group authentication protocols in the literature that are verified using the Scyther tool [55]. They have shown that some discrete logarithm problem-based group authentication protocols provide sufficient degrees of authentication.

Per the results given in Table 6, CoMAD provides general non-injective agreement, non-injective synchronization, agreement on some given parameters, and secrecy for some given parameters, as shown. All claims are verified, and no attack patterns have been found. All roles in all three sub-protocols are also proven reachable in terms of state traces.

VI. PERFORMANCE DISCUSSION

For the sake of completeness and consistency, we have made a brief comparative performance analysis by comparing CoMAD (and its sub-protocols) to CL-GAKA protocol proposed by Semal *et al.* [9], as it also supports ad-hoc networks. The protocols that rely on a base or ground station are not analyzed since it would not be fair to compare different types of architectures.

Within the analysis, it was not possible to make use of experimental results due to some missing, unclear, or scenario-dependant information in [9], such as key sizes, ID lengths, etc. Hence, we worked on the protocols' time and message complexities, as well as operation counts of hashing, symmetric encryption, and asymmetric encryption since they are somewhat costly in terms of time and computing power. According to the results presented in Table 7, CoMAD is superior to CL-GAKA in some performance aspects. CL-GAKA heavily relies on consecutive hashing operations, while CoMAD does not require hashing unless a new node with no prior context information requests to be authenticated. Additionally, CoMAD utilizes the group authentication scheme proposed by Harn [29] that has $O(1)$ algorithmic complexity while CL-GAKA uses a more complex approach. Apart from giving some performance-related information, this comparison helps in expressing the benefits of adopting CoMAD in a drone swarm.

VII. CONCLUSION

This paper suggests a novel context-aware mutual authentication protocol, called CoMAD, that is uniquely tailored for cooperating drone networks. The protocol has a special emphasis on re-authentication scenarios and is empowered with a group authentication feature. From the network's perspective, it administrates the cases of (*i*) joining a new single drone, (*ii*) joining a former single drone, (*iii*) joining a former group of drones (*iv*) removing an existing drone, and (*v*) delegating a new leader (i.e., master).

The context-awareness, as integrated into CoMAD, provides an extra layer of security that is backed by cryptographic functions (e.g., hashing, encryption, etc.) as well as increasing obscurity for an adversary. Over and above, this work is a novel proof-of-concept of such an extensive security mechanism, particularly in drone networks. Apart from the context-awareness, CoMAD is empowered with many countermeasures (e.g., use of nonces, timestamps, and public-key infrastructures) against various well-known attacks, including impersonation, replay, man-in-the-middle, and others.

A wide informal security analysis is made to explain how certain threats are mitigated by design. In addition to the informal analyses, the CoMAD protocol is also formally tested with an automated formal security protocol verification tool called Scyther. Results of the tests provide a complete verification of the authentication and secrecy claims of the protocol for all defined use-cases and arbitrary network instances. Moreover, we have disclosed some performance-wise benefits of adopting CoMAD in drone swarms.

CoMAD makes use of a centralized architecture, yet we believe that introducing decentralization would yield more flexible designs. Hence, we will work on integrating blockchain-based lightweight authentication schemes, such as [57], to ad-hoc drone networks as a future work direction, which will also help to achieve post-compromise security to some extent. Moreover, we will work on energy-efficient measures against traffic analysis attacks that aim to reveal the master node by observing the message traffic.

REFERENCES

[1] E. Yanmaz, S. Yahyanejad, B. Rinner, H. Hellwagner, and C. Bettstetter, "Drone networks: Communications, coordination, and sensing," *Ad Hoc Netw.*, vol. 68, pp. 1–15, Jan. 2018.

[2] H. Wang, H. Zhao, J. Zhang, D. Ma, J. Li, and J. Wei, "Survey on unmanned aerial vehicle networks: A cyber physical system perspective," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1027–1070, 2nd Quart., 2020.

[3] S. Hayat, E. Yanmaz, C. Bettstetter, and T. X. Brown, "Multi-objective drone path planning for search and rescue with quality-of-service requirements," *Auto. Robots*, vol. 44, no. 7, pp. 1183–1198, Sep. 2020.

[4] C. Krauß, M. Schneider, and C. Eckert, "On handling insider attacks in wireless sensor networks," *Inf. Secur. Tech. Rep.*, vol. 13, no. 3, pp. 165–172, Aug. 2008.

[5] K. Han, K. Kim, and T. Shon, "Untraceable mobile node authentication in WSN," *Sensors*, vol. 10, pp. 4410–4429, Apr. 2010.

[6] S. Jiang, J. Zhang, J. Miao, and C. Zhou, "A privacy-preserving reauthentication scheme for mobile wireless sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 9, no. 5, May 2013, Art. no. 913782.

[7] R. D. U. Suriya and R. Vayanaperumal, "Detecting and revocation the compromised node in zone–based wireless sensor network using a two stage approach," in *Proc. 6th Int. Conf. Adv. Comput. (ICoAC)*, Dec. 2014, pp. 7–13.

[8] M. Turkanović, B. Brumen, and M. Hölbl, "A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion," *Ad Hoc Netw.*, vol. 20, pp. 96–112, Sep. 2014.

[9] B. Semal, K. Markantonakis, and R. N. Akram, "A certificateless group authenticated key agreement protocol for secure communication in untrusted UAV networks," in *Proc. IEEE/AIAA 37th Digit. Avionics Syst. Conf. (DASC)*, Sep. 2018, pp. 1–8.

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

[11] J. Srinivas, A. K. Das, N. Kumar, and J. J. P. C. Rodrigues, "TCALAS: Temporal credential-based anonymous lightweight authentication scheme for Internet of drones environment," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6903–6916, Jul. 2019.

[12] B. Kim and J. Song, "Energy-efficient and secure mobile node reauthentication scheme for mobile wireless sensor networks," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, pp. 1–6, Dec. 2019.

[13] M. Wazid, A. K. Das, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Design and analysis of secure lightweight remote user authentication and key agreement scheme in Internet of drones deployment," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3572–3584, Apr. 2019.

[14] Ö. Aydin, G. Dalkiliç, and C. Kösemen, "A novel grouping proof authentication protocol for lightweight devices: GPAPXR+," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 28, no. 5, pp. 3036–3051, Sep. 2020.

[15] Y. Zhang, D. He, L. Li, and B. Chen, "A lightweight authentication and key agreement scheme for Internet of drones," *Comput. Commun.*, vol. 154, pp. 455–464, Mar. 2020.

[16] T. Alladi, V. Chamola, and N. Kumar, "PARTH: A two-stage lightweight mutual authentication protocol for UAV surveillance networks," *Comput. Commun.*, vol. 160, pp. 81–90, Jul. 2020.

[17] Q. Cheng, C. Hsu, Z. Xia, and L. Harn, "Fast multivariate-polynomial-based membership authentication and key establishment for secure group communications in WSN," *IEEE Access*, vol. 8, pp. 71833–71839, 2020.

[18] Z. Ali, S. A. Chaudhry, M. S. Ramzan, and F. Al-Turjman, "Securing smart city surveillance: A lightweight authentication mechanism for unmanned vehicles," *IEEE Access*, vol. 8, pp. 43711–43724, 2020.

[19] B. Bera, D. Chattaraj, and A. K. Das, "Designing secure blockchain-based access control scheme in IoT-enabled Internet of drones deployment," *Comput. Commun.*, vol. 153, pp. 229–249, Mar. 2020.

[20] S. Lal and J. Prathap, "An energy-efficient lightweight security protocol for optimal resource provenance in wireless sensor networks," *Turkish J. Elect. Eng. Comput. Sci.*, vol. 28, pp. 3208–3218, Nov. 2020.

[21] S. Hussain, S. A. Chaudhry, O. A. Alomari, M. H. Alsharif, M. K. Khan, and N. Kumar, "Amassing the security: An ECC-based authentication scheme for Internet of drones," *IEEE Syst. J.*, early access, Mar. 1, 2021, doi: 10.1109/JSYST.2021.3057047.

[22] Y. Lei, L. Zeng, Y.-X. Li, M.-X. Wang, and H. Qin, "A lightweight authentication protocol for UAV networks based on security and computational resource optimization," *IEEE Access*, vol. 9, pp. 53769–53785, 2021.

[23] M. Yahuza, M. Y. I. Idris, A. W. A. Wahab, T. Nandy, I. B. Ahmedy, and R. Ramli, "An edge assisted secure lightweight authentication technique for safe communication on the Internet of drones network," *IEEE Access*, vol. 9, pp. 31420–31440, 2021.

[24] P. Ogren, "Split and join of vehicle formations doing obstacle avoidance," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Apr. 2004, pp. 1951–1955.

[25] K. Raghuwaiya, J. Vanualailai, and B. Sharma, "Formation splitting and merging," in *Advances in Swarm Intelligence* (Lecture Notes in Computer Science). Cham, Switzerland: Springer, 2016, pp. 461–469.

[26] L. He, P. Bai, X. Liang, J. Zhang, and W. Wang, "Feedback formation control of UAV swarm with multiple implicit leaders," *Aerosp. Sci. Technol.*, vol. 72, pp. 327–334, Jan. 2018.

[27] M. Coppola, J. Guo, E. Gill, and G. C. de Croon, "Provable self-organizing pattern formation by a swarm of robots with limited knowledge," *Swarm Intell.*, vol. 13, pp. 59–94, Feb. 2019.

[28] H. Zhu, J. Juhl, L. Ferranti, and J. Alonso-Mora, "Distributed multi-robot formation splitting and merging in dynamic environments," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 9080–9086.

[29] L. Harn, "Group authentication," *IEEE Trans. Comput.*, vol. 62, no. 9, pp. 1893–1898, Sep. 2013.

[30] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[31] S. Li, I. Doh, and K. Chae, "A group authentication scheme based on Lagrange interpolation polynomial," in *Proc. 10th Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput. (IMIS)*, Jul. 2016, pp. 386–391.

[32] S. K. Narad, M. R. Sayankar, S. V. Alone, and P. S. Mahiskar, "Secret sharing scheme for group authentication—A review," in *Proc. Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, vol. 1, Apr. 2017, pp. 12–16.

[33] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 2, pp. 198–208, Mar. 1983.

[34] J. Tu, D. Tian, and Y. Wang, "An active-routing authentication scheme in MANET," *IEEE Access*, vol. 9, pp. 34276–34286, 2021.

[35] Z. Mi, Y. Yang, and J. Y. Yang, "Restoring connectivity of mobile robotic sensor networks while avoiding obstacles," *IEEE Sensors J.*, vol. 15, no. 8, pp. 4640–4650, Aug. 2015.

[36] J.-H. Kang and K.-J. Park, "Spatial retreat of net-drones under communication failure," in *Proc. 8th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2016, pp. 89–91.

[37] G.-H. Kim, I. Mahmud, and Y.-Z. Cho, "Self-recovery scheme using neighbor information for multi-drone ad hoc networks," in *Proc. 23rd Asia–Pacific Conf. Commun. (APCC)*, Dec. 2017, pp. 1–5.

[38] U. C. Çabuk, M. Tosun, V. Akram, and O. Dagdeviren, "Connectivity management in drone networks: Models, algorithms, and methods," in *Intelligent Analytics With Advanced Multi-Industry Applications*, 1st ed, Z. Sun, Ed. Hershey, PA, USA: IGI Global, Jan. 2021.

[39] A. K. Dey, "Understanding and using context," *Pers. Ubiquitous Comput.*, vol. 5, no. 1, pp. 4–7, 2001.

[40] N. G. S. Campos, D. G. Gomes, F. C. Delicato, A. J. V. Neto, L. Pirmez, and J. N. de Souza, "Autonomic context-aware wireless sensor networks," *J. Sensors*, vol. 2015, pp. 1–14, Apr. 2015.

[41] J. Jonsson and B. Kaliski, "Public-Key cryptography standards (PKCS) #1: RSA cryptography specifications version 2.1," IETF, Fremont, CA, USA, Tech. Rep. RFC3447, 2003.

[42] A. Ghafoor, M. Sher, M. Imran, and K. Saleem, "A lightweight key freshness scheme for wireless sensor networks," in *Proc. 12th Int. Conf. Inf. Technol. New Generat.*, Apr. 2015, pp. 169–173.

[43] E. Barker and A. Roginsky, "Transitioning the use of cryptographic algorithms and key lengths," NIST, Gaithersburg, MD, USA, Tech. Rep. 800-131Ar2, Mar. 2019, doi: 10.6028/NIST.SP.800-131Ar2.

[44] H.-S. Kim, J.-S. Han, and Y.-H. Lee, "Scalable network joining mechanism in wireless sensor networks," in *Proc. IEEE Topical Conf. Wireless Sensors Sensor Netw.*, Jan. 2012, pp. 45–48.

[45] A. Zeb, A. K. M. M. Islam, S. Komaki, and S. Baharun, "Multi-nodes joining for dynamic cluster-based wireless sensor network," in *Proc. Int. Conf. Informat., Electron. Vis. (ICIEV)*, May 2014, pp. 1–6.

[46] S. Xiong and J. Li, "An efficient algorithm for cut vertex detection in wireless sensor networks," in *Proc. IEEE 30th Int. Conf. Distrib. Comput. Syst.*, Jun. 2010, pp. 368–377.

[47] F. Aftab, A. Khan, and Z. Zhang, "Hybrid self-organized clustering scheme for drone based cognitive Internet of Things," *IEEE Access*, vol. 7, pp. 56217–56227, 2019.

[48] M. V. S. S. Nagendranath, B. A. Ramesh, and V. Aneesha, "Detection of packet dropping and replay attacks in MANET," in *Proc. Int. Conf. Current Trends Comput., Electr., Electron. Commun. (CTCEEC)*, Sep. 2017, pp. 933–938.

[49] S. Katragadda, P. J. Darby, A. Roche, and R. Gottumukkala, "Detecting low-rate replay-based injection attacks on in-vehicle networks," *IEEE Access*, vol. 8, pp. 54979–54993, 2020.

[50] R. Heartfield, G. Loukas, S. Budimir, A. Bezemskij, J. R. J. Fontaine, A. Filippoupolitis, and E. Roesch, "A taxonomy of cyber-physical threats and impact in the smart home," *Comput. Secur.*, vol. 78, pp. 398–428, Sep. 2018.

[51] S. Skorobogatov, "Data remanence in flash memory devices," in *Cryptographic Hardware and Embedded Systems—CHES*, J. R. Rao and B. Sunar, Eds. Berlin, Germany: Springer, 2005, pp. 339–353.

[52] U. C. Çabuk, Ö. Aydin, and G. Dalkiliç, "A random number generator for lightweight authentication protocols: XorshiftR+," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 25, pp. 4818–4828, Dec. 2017.
[53] C. J. F. Cremers, "The Scyther tool: Verification, falsification, and analysis of security protocols," in *Computer Aided Verification*. Berlin, Germany: Springer, 2008, pp. 414–418.
[54] C. Cremers and S. Mauw, *Operational Semantics and Verification of Security Protocols*. Berlin, Germany: Springer, 2012.
[55] H. Yang, V. Oleshchuk, and A. Prinz, "Verifying group authentication protocols by Scyther," *J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl.*, vol. 7, pp. 3–19, Jun. 2016.
[56] G. Lowe, "A hierarchy of authentication specifications," in *Proc. 10th Comput. Secur. Found. Workshop*, Jun. 1997, pp. 31–43.
[57] B. Hamdaoui, M. Alkalbani, A. Rayes, and N. Zorba, "IoTShare: A blockchain-enabled IoT resource sharing on-demand protocol for smart city situation-awareness applications," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10548–10561, Oct. 2020.

UMUT CAN CABUK (Member, IEEE) received the B.Sc. degree in electronics engineering from Uludag University, Bursa, Turkey, in 2012, and the M.Sc. degree in information technology engineering from Aarhus University, Aarhus, Denmark, in 2015. He is currently pursuing the Ph.D. degree with the International Computer Institute, Ege University, Izmir, Turkey. He also works as a Researcher with the International Computer Institute, Ege University. He has coauthored over 30 scholarly publications and issued three patent applications. His research interests include mobile and wireless networks, the IoT, cryptography, and graph theory.

GOKHAN DALKILIC received the B.Sc. degree in computer engineering from Ege University, Izmir, Turkey, in 1997, the M.Sc. degree in computer science from the University of Southern California, Los Angeles, CA, USA, in 1999, the M.Sc. degree in computer science from the International Computer Institute, Ege University, in 2001, and the Ph.D. degree in computer engineering from Dokuz Eylul University, Izmir, in 2004. He was a Visiting Lecturer with the University of Central Florida, Orlando, FL, USA, in 2003. He is currently an Associate Professor with the Department of Computer Engineering, Dokuz Eylul University. He has coauthored over 80 articles, four books, and applied for three patents. His research interests include cryptography, statistical language processing, computer networks, lightweight authentication, and NLP.

ORHAN DAGDEVIREN (Member, IEEE) received the B.Sc. and M.Sc. degrees in computer engineering from the Izmir Institute of Technology, and the Ph.D. degree from the International Computer Institute, Ege University, Izmir, Turkey. He is currently an Associate Professor and the Head of the Network Engineering Science and Technology (NETOS) Laboratory, International Computer Institute. He has coauthored over 100 articles in various journals and conferences. His research interests include distributed computing, graph theory, fault tolerance, wireless networking, and security areas.