UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

CARLOS FELIPE EMYGDIO DE MELO

UAVOUCH: A DISTRIBUTED DRONE IDENTITY AND LOCATION VALIDATION MECHANISM

Porto Alegre
2020

CARLOS FELIPE EMYGDIO DE MELO

UAVOUCH: A DISTRIBUTED DRONE IDENTITY AND LOCATION VALIDATION MECHANISM

Thesis presented to Programa de Pós-Graduação em Engenharia Elétrica of Universidade Federal do Rio Grande do Sul in partial fulfillment of the requirements for the degree of Master in Electrical Engineering.
Area: Control and Automation

ADVISOR: Prof. Dr. Edison Pignaton de Freitas

Porto Alegre
2020

CARLOS FELIPE EMYGDIO DE MELO

UAVOUCH: A DISTRIBUTED DRONE IDENTITY AND LOCATION VALIDATION MECHANISM

This thesis was considered adequate for obtaining the degree of Master in Electrical Engineering and approved in its final form by the Advisor and the Examination Committee.

Advisor: _____
Prof. Dr. Edison Pignaton de Freitas, UFRGS
Doutor pela Universidade de Halmstad, Suécia e pela Universidade Federal do Rio Grande do Sul, Brasil

Examination Committee:

Prof. Dr. Raul Ceretta Nunes, UFSM
Doutor pela Universidade Federal do Rio Grande do Sul, Brasil

Prof. Dr. Weverton Luís da Costa Cordeiro, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul, Brasil

Prof. Dr. Carlos Eduardo Pereira, UFRGS
Doutor pela Universidade de Stuttgart, Alemanha

Coordinator of PPGEE: _____
Prof. Dr. João Manoel Gomes da Silva Jr.

Porto Alegre, March 2020.

DEDICATÓRIA

Dedico este trabalho primeiramente a Deus, por guiar os meus passos e me amparar nos momentos difícies.

Aos meus pais Antonio Carlos de Melo e Benedita Regina Célia Paula Emygdio de Melo, por todo amor, suporte, conselhos e que apesar da distância sempre se fizeram presente, me aconselhando e me amparando no momentos mais difíceis.

Aos meus irmãos Bruna Emygdio de Melo e Daniel Emygdio de Melo por toda compreensão, carinho e aprendizados que trocamos ao longo de toda a nossa vida.

À minha esposa Lara Peruzzolo Cargnin por todo amor, apoio e paciência.

Sem vocês nada disso seria possível.

AGRADECIMENTOS

Ao Programa de Pós-Graduação em Engenharia Elétrica, PPGEE, pela opor- tunidade de realização de pesquisa nas minhas áreas de interesse.

Ao meu professor e orientador, Edison Pignaton de Freitas, agradeço por me acolher como seu orientando e a todo o tempo que dedicou a me orientar em meu trabalho. Agradeço sempre pelo seu apoio, saiba que sou muito grato pelas oportunidades que colocastes na minha vida e pelos ensinamentos, dentro e fora da sala de aula. Agradeço por nossa amizade.

Muito obrigado. Aos meus colegas de laboratório Marcos Vizzotto Rodrigues, Maik Basso, Tulio Dapper e Silva e muitos outros que me foram minha segunda família nesses anos. Muito obrigado por todo apoio, amizade, conselhos e ensinamento. Foi um prazer dividir essa caminhada com vocês.

À todas as pessoas que se fizeram presentes nessa etapa da minha vida me apoiando para que a conclusão deste curso fosse possível.

À CAPES pela concessão de bolsa para a minha manutenção durante o período de dedicação exclusiva ao meu trabalho de pesquisa.

"Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning."

Albert Einstein

ABSTRACT

Emerging surveillance applications of UAV teams rely on secure communication to exchange information, coordinate their movements, and fulfill mission objectives. Protecting the network by identifying malicious nodes access trying to disturb the system is an important task, which is particularly sensitive in the military domain. Observing this need, this paper presents the design and evaluation of UAVouch: an identity and location validation scheme combining a public-key based authentication with a movement plausibility check for groups of UAVs. The key idea of UAVouch supplement the authentication mechanism by periodically checking the plausibility of the location of neighboring UAVs, allowing the detection of intruders that are unable to follow expected trajectories. The proposed solution was evaluated in a simulated military surveillance scenario in which it detects malicious nodes' position falsification attacks with an accuracy on average above 85%.

Keywords: Sybil Attack, Security Protocols, Inter-Drone Communications, Drone-Based Network, Distributed Applications.

RESUMO

As aplicações emergentes de vigilância, com equipes de VANTs, dependem de comunicação segura para trocar informações, coordenar seus movimentos e cumprir os objetivos da missão. Proteger a rede identificando o acesso de nós mal-intencionados tentando perturbar o sistema é uma tarefa importante, e particularmente sensível no domínio militar. Observando essa necessidade, este artigo apresenta o design e a avaliação do UAVouch: Um esquema distribuído de validação de localização e identidade de drones que combina uma autenticação baseada em chave pública com uma verificação de plausibilidade de movimento para grupos de VANTs. A ideia principal do UAVouch complementa o mecanismo de autenticação, verificando periodicamente a plausibilidade da localização dos VANTs vizinhos, permitindo a detecção de intrusos que não conseguem seguir as trajetórias esperadas. A solução proposta foi avaliada em simulação através de um cenário de vigilância militar, no qual detectou-se ataques de falsificação de posição de nós mal-intencionados com precisão em média acima de 85%.

Palavras-chave: Ataque Sybil, Protocolos de Segurança, Comunicação entre Drones, Redes de Drones, Aplicações Distribuidas.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AES	Advanced Encryption Standard
ANAC	Agência Nacional de Aviação Civil
BFT	Byzantine Fault Tolerance
DDoS	Distributed Denial of Service
DES	Data Encryption Standard
DoS	Denial of Service
FANET	Flying Ad-hoc Network
GPS	Global Positioning System
MANET	Mobile Ad-hoc Network
MD4	Message-Digest algorithm 4
MD5	Message-Digest algorithm 5
MIT	Massachusetts Institute of Technology
MitM	Man-in-the-Middle
PIN	Personal Identification Number
PKI	Public Key Infrastructure
RSA	Rivest-Shamir-Adleman
SSL	Secure Sockets Layer
SVM	Support Vector Machine
TLS	Transport Layer Security
UAV	Unmanned Aerial Vehicle
VANET	Veicular Ad hoc Networks
VANT	Veículo Aéreo Não Tripulado
WSN	Wireless Sensor Network

CONTENTS

1 RESUMO EXTENDIDO

Este capítulo apresenta, de forma resumida, o presente trabalho, o qual é intitulado "UAVouch: Um mecanismo distribuído de validação de localização e identidade de drones".

1.1 Introdução

Nos últimos anos, os veículos aéreos não tripulados (VANTs), também conhecidos como drones, têm sido usados em várias aplicações emergentes nos domínios civil e militar. Especialmente em aplicações militares, como vigilância ou reconhecimento, um grupo de VANTs podem ser utilizados, formando uma rede móvel ad hoc, para alertar os soldados sobre qualquer ameaça à frente da linha de visão da tropa (ZACARIAS et al., 2017). Entretanto, existem preocupações em relação às vulnerabilidades pertinentes às redes móveis. Por exemplo, ataques como sinkhole, spoofing, eavesdropping, Sybil, entres outros são possíveis de acontecer em redes ad hoc de drones (FOTOUHI et al., 2019; GARCIA-MAGARINO et al., 2019; ALTAWY; YOUSSEF, 2016).

Entre os ataques comuns a redes sem fio, pode-se destacar ataques de personificação e Sybil. Em um ataque de personificação, o atacante consegue se disfarçar com sucesso como uma das partes legítimas com o objetivo de perturbar a rede ou obter privilégios dentro da rede. Esse ataque é possível através de roubo de identidade. Em um ataque Sybil (DOUCEUR, 2002), o atacante assume várias identidades com o objetivo de sobrecarregar um nó específico ou obter a maioria dentro da rede e influenciar o resultado de um sistema de votação, podendo assim superar barreiras de segurança dessa rede. Esse ataque é possível através do roubo de identidades ou da criação de um conjunto de novas identidades. Quando executados com êxito, os ataques de personificação e Sybil possibilitam a execução de outros tipos de ataques como os ataques de manipulação de informação e Denial of Service (DoS) (WALIA; BHATIA; KAUR, 2018).

Diante do exposto, este trabalho apresenta, como contramedida a esses ataques, a proposta de um esquema distribuído de verificação de identidade e localização de drones, usando um mecanismo de autenticação baseado no modelo de chaves públicas combinado com um mecanismo de validação de posição e plausibilidade do movimento. Para testar o esquema proposto, foi desenvolvido um cenário usando unidades militares compostas por um veículo blindado escoltado por 4 VANTs, o qual foi denominado com uma célula. Os drones voam a uma certa distância, a qual possibilita a comunicação entre eles e o blindado a todo instante da missão, e a comunicação entre os VANTs é intermitente, gerando assim uma rede móvel ad hoc sem fio. É de suma importância a manutenção da integridade de rede pois assim um nó malicioso não conseguirá ingressar na rede e nenhum nó legítimo será comprometido. Pensando nesses requerimentos, o objetivo principal desse

trabalho é de prover uma solução para segurança dessas redes contra ataques que possam comprometer a integridade da rede e comprometer operações militares.

1.2 Contribuições

Em relação às contribuições deste trabalho, pode-se destacar :

1. Um esquema distribuído de validação de localização e identificação de drones que permite que uma célula se autentique e verifique a posição de um drone sem nenhum suporte de infraestrutura;

2. Uma avaliação do esquema proposto para o cenário de aplicação que inclui o desempenho de detecção, análise de retransmissão de pacotes e overhead contra ataques de personificação e Sybil usando um simulador de rede realista;

1.3 Experimentos e Resultados

Para comprovar a viabilidade da solução proposta, foram realizados testes com dois cenários de ataques. Ambos foram executados em ambiente virtual, dentro do OMNET++ (5.4.1), um simulador de eventos discretos, para criar um ambiente de comunicação e troca de mensagens realístico. O modelo de mobilidade foi implementado utilizando o INET framework (4.2), e finalmente para as funções criptográficas fez-se uso do OpenSLL (3.0.0).

O primeiro cenário foi desenvolvido para testar a identificação de um nó malicioso que conseguiu, de maneira bem-sucedida, entrar na rede assumindo a identidade de um nó legitimo (ataque de personificação). O segundo cenário foi desenvolvido considerando uma situação mais desafiadora. Durante o deslocamento de uma célula de uma localidade para outra, esta pode encontrar-se e conectar-se com outras células para trocar informações e expandir o raio de exploração e/ou vigilância. Um atacante pode se aproveitar disso e se disfarçar de uma célula legítima e começar a disseminar informações falsas, fazendo com que a célula legítima seja desviada para uma armadilha. Esse cenário foi idealizado para testar o mecanismo proposto em relação à ataque Sybil, quando um atacante personifica um ou mais drones.

O esquema proposto foi testado em relação à sua assertividade em identificar o nó malicioso em ambos os cenários, mas além disso, o algoritmo também foi avaliado em relação ao número de pacotes retransmitidos, ao número de decisões alcançados pelo sistema de votação e ao overhead introduzido na rede. Em ambos os cenários o mecanismo proposto apresentou uma assertividade acima de 90% no melhor cenário. A média de decisões alcançadas pelo mecanismo foi de 80% para o primeiro cenário e de 67% para o segundo cenário. Em relação à retransmissão de pacotes, a solução apresentou uma taxa em torno de 50% de pacotes retransmitido no pior caso e em relação ao overhead, apesar de ter atingido quase 300% de overhead no pior caso, se compararmos a taxa de dados transmitidos nesse caso e o consumo de banda necessário para a transmissão desses dados com as taxas de transmissão de dados e largura de banda de tecnologias difundidas atualmente, como WiMax e 4G, o overhead causado pelo UAVouch é totalmente aceitável.

1.4 Conclusões

Este trabalho apresentou um esquema distribuído para verificação de identidade e localização, combinando um mecanismo de autenticação baseado em chave assimétrica com um mecanismo de validação de posição e plausibilidade de movimento para sistemas usando grupos de drones. A proposta foi avaliada usando dois cenários de ataque, um para o ataque de personificação, com o invasor dentro da célula, e o outro para o ataque de Sybil, com o invasor fora da célula. O UAVouch apresentou uma alta taxa de detecção, acima de 90 % na detecção do nó malicioso dentro (cenário 1) e fora (cenário 2) de sua rede. Devido à natureza distribuída do protocolo, foram apresentadas avaliações de colisão e overhead do mecanismo. Os resultados mostraram uma taxa de colisão abaixo de 50 % para o pior cenário e também um valor de overhead totalmente aceitável, o que sugere que o mecanismo proposto é viável para implementação com dispositivos reais.

2 INTRODUCTION

In the last few years, Unmanned Aerial Vehicles (UAVs), also known as drones, have been used in several emerging applications in both civil and military domains. According to data from the Brazilian National Agency of Civil Aviation (ANAC), the number of registered drones for professional use grew by approximately 233% between 2017 and 2019 (AGÊNCIA NACIONAL DE AVIAÇÃO CIVIL, 2019). This number is even higher considering the market for drones worldwide (MOSKWA, 2016). Along with the growth in the number of drones, the number of applications using drones has also seen a significant increase. Some well-known applications of drone-based systems are surveillance, film-making, disaster management, and defense (SHAKHATREH et al., 2019).

Although drones are becoming more common in civilian applications, military applications are still dominating, and they represent an essential asset in the modern battlefield (ORFANUS; DE FREITAS; ELIASSEN, 2016). In applications such as surveillance or military reconnaissance, groups of UAVs can be used to provide awareness of threats ahead of the troop's line of sight (ZACARIAS et al., 2017). However, connecting multiple UAVs together through ad hoc networks raise vulnerability issues, and enemy threats must be addressed in advance. For instance, attacks such as sinkhole, spoofing, eavesdropping, impersonation and Sybil can potentially ruin a mission (FOTOUHI et al., 2019; GARCIA-MAGARINO et al., 2019; ALTAWY; YOUSSEF, 2016).

In an impersonation attack, the attacker manages to successfully masquerade itself as one of the legitimate parties (ADAMS, 2005), and a Sybil attack takes place when a malicious node impersonates or create multiple identities (DOUCEUR, 2002). Impersonation and Sybil attacks, when successfully executed, give the intruder the possibility to launch other kinds of attacks, such as information manipulation and Denial of Service (DoS) (WALIA; BHATIA; KAUR, 2018). A Sybil node could be used to manipulate the position information exchanged among nodes in a vehicular or drone network, for instance, in an attempt to cause a collision, or simply to separate a specific node from its network in order to steal its information/technology.

The canonical approach for controlling access to a protected network is through the use of of a public key infrastructure (PKI). In these schemes a centralized entity can distribute certificates to legitimate users and devices who can use the certificates to authenticate themselves to other members of the network. This provides a basic level of security, but does not protect the system against insider attacks where the intruder has gained access to a valid certificate. Mechanisms as the ones presented in (BOEIRA; ASPLUND; BARCELLOS, 2018; BOEIRA; ASPLUND; BARCELLOS, 2019) are capable of detecting malicious activity in vehicular networks, a domain which has been extensively studied with respect to security concerns (LOUKAS et al., 2019; SHARMA; KAUL, 2018). However, only a few works are covering the area of intrusion detection for

drone networks (ALTAWY; YOUSSEF, 2016), several presenting artificial intelligence or computer-vision based solutions, which tend to be resource consuming, thus not ideal for resource-constrained drones.

This dissertation work presents a novel approach - UAVouch - that combines the use of public key authentication with location validation. The idea of using physical location and movement as a authentication mechanism is not in itself new, there are several works that make use of this idea in the vehicular domain (BOEIRA; ASPLUND; BARCELLOS, 2019; WANG et al., 2016). However, the proposed approach, which is specifically designed for collaborative drone applications, provides some interesting properties that have not been previously studied and described in the literature. First, it presents a fully distributed group management mechanism in which an existing group (called a cell) collectively determines whether a joining node should be admitted. Second, once a cell has been formed, the nodes in the cell keep controlling each other's movement patterns to ensure that everyone is behaving as expected. This position validation mechanism can be seen as a complement to cryptographic methods and as a form of anomaly detection (using node mobility as the feature set rather than data traffic as is more common in the literature). Provided sufficiently complex mobility patterns, it will be hard for an attacker to guess where other nodes expect it to move to. Finally, the proposed approach also supports trusted communication between different cells.

To test the proposed scheme, an application scenario, illustrated in Figure 1, was designed using military units composed of an armored ground vehicle escorted by a number of drones. The purpose of the drones are to monitor an area out of sight from the ground vehicle. These units (the ground vehicles with their escorting drones) form cells. The drones fly around the armored ground vehicle in distances that keep the wireless connection with the vehicle on the ground and an intermittent connection with one or more of the other drones, forming an ad hoc network to exchange data between themselves and with the vehicle on the ground.

Figure 1 – Illustration of the application scenario.

Source: author

The UAVouch proposal is evaluated using this setting in an simulation platform based

on INET and OMNet++. Two attack scenarios are defined, one considering a intruder within a cell, and another where the attack comes from a neighboring cell. A basic mobility model for the drones is considered, which is assumed not to be known to the attackers. The results show that under this assumption, UAVouch allows detecting the intruders with high accuracy. The location validation itself is very cost effective since it does not require any computationally demanding operations. The main trade-off is associated with added messaging due to sending location messages among the nodes.

2.1 Objectives and Contributions

The key contributions of this work can be summarized as follows:

- *A* distributed identity and position validation mechanism that allows a cell to authenticate and verify the position of a drone without any infrastructure support;
- An assessment of the proposed mechanism for the application scenario that includes the detection performance, collision and overhead analysis against impersonation and Sybil attacks using an realistic network simulator.

2.2 Work Organization

The rest of this work is organized as follows. Chapter 3 presents a study of the main concepts involving authentication and position verification. In Chapter 4 a more directed study about the related works is presented, as well as the comparison with the present work. In Chapter 5 the problem is formulated describing the attack scenarios. Chapter 6 describes the proposed identity and position validation mechanism. Chapter 7 presents the experiments to validate the proposed solution along with the discussion about the acquired results, while Chapter 8 concludes this work providing insights for future works.

3 BACKGROUND CONCEPTS REVIEW

In this chapter, the basic concepts explored in this work are presented.

3.1 Authentication Concepts

According to TANEMBAUM; WETHERALL (2014) authentication is the technique by which a process confirms that its communication partner is who it claims to be and not an impostor. In other words, authentication aims to determine whether or not you are talking to a specific process. For STALLINGS (2014), authentication is the process of verifying an identity claimed by or for a system entity. The author also highlights that an authentication process, basically consists of two steps:

- **Identification step**: Presenting an identifier to the security system. Those must assigned carefully, because authenticated entities are the basis for other security services, such as access control services;
- **Verification step**: Presenting or generating authentication information that corroborates the binding between the entity and the identifier

Regarding to user authentication, STALLINGS (2014) also state that there are four general means of verify the user's identity. These methods can be used alone or in combination, and if properly implemented, can provide secure user authentication.

- **Something the individual knows**: Examples include a password, a personal identification number (PIN), or answers to a prearrange set of equations;
- **Something the individual possesses**: Examples includes cryptographic keys, electronic keycards, smart cards, and physical keys. This type of authenticator is referred to as a *token*;
- **Something the individual is (statics biometrics)**: Examples include recognition by fingerprint, retina, and face;
- **Something the individual does (dynamic biometrics)**: Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm;

For an ad hoc network, the use of cryptography keys (Something the individual possesses) for authentication is a common method explored in the literature. For UAV based network, the use of symmetric cryptography keys is explored in RAJATHA; ANANDA; NAGARAJ (2015).

3.1.1 Cryptography

Cryptography (from the Greek *kriptós* - hidden and *gráfos* - written) is a technique applied to hide the content of a message. According to KAHN (1996), this technique dates back to the year 1900 B.C. with a master scribe that sketched hieroglyphs to told the story of his lord's life, Khnumhotep II. Although the use of this technique was also almost accidental, since the idea was not to conceal information but to leave a historical register, theses sketches opened the recorded history of cryptology. The first registered intentional use of the cryptography is in the hidden message sent to Sparta by Demaratus, the son of Ariston, alerting them that Xerxes, king of the Persian Achaemenid Empire (486-465 B.C.), had decided upon the invasion of Greece. The message was concealed scraping the wax off a pair of wooden folding tablets, and after the message written, the tablets would be covered again with wax, and therefore would appear to be blank.

With the advent of technology and network communication, the cryptographic techniques had to be modernized to fit in nowadays. In the light of this modern cryptosystem, DIFFIE; HELLMAN (1976) define cryptography as the study of "mathematical" systems for solving two kinds of security problems: privacy and authentication:

- **Privacy system**: It is used to prevent the extraction of information by unauthorized parties from messages sent through a public channel, assuring the sender that only the intended receiver is able to read the message;

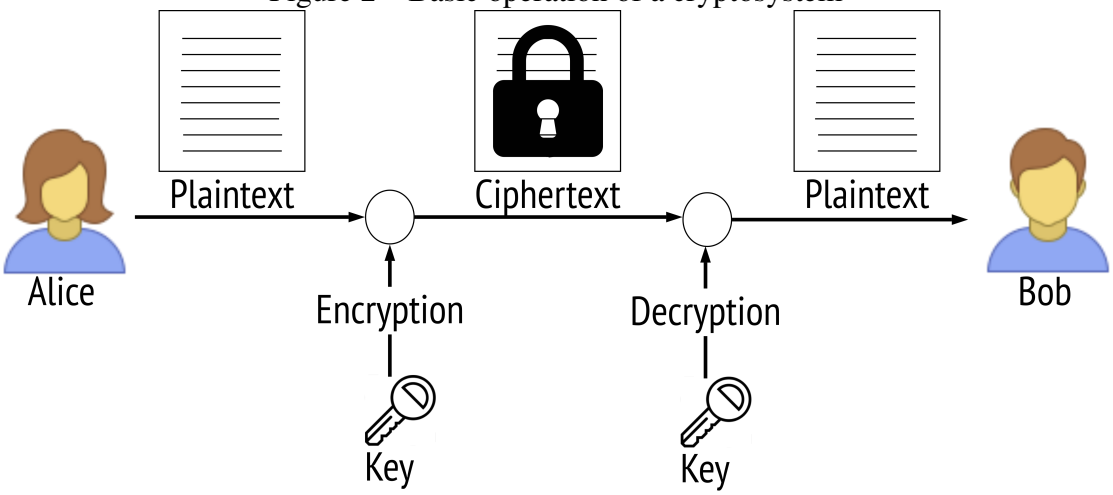
- **Authentication system**: It is used to prevent the injection of a falsified message into the public channel, assuring to the receiver the legitimacy of the sender;

Figure 2 presents the basic concept of how cryptography works. Imagined that Alice wants to send a message to Bob in a secure way. Alice first writes the message as **plaintext**, which means that anyone is able to read the message. The plaintext goes through a process of encryption, resulting in a **ciphertext** (the encrypted message). This ciphertext is then transmitted to Bob. If someone intercept this transmission, they wouldn't be able to read the message because it is encrypted. When the ciphertext arrives at Bob, it goes through a decryption process, resulting in the plaintext, which now can be read by Bob. Both encryption and decryption process uses the same cipher, or algorithm, to convert the plaintext into the ciphertext and then back again to plaintext. The encryption (or decryption) process is composed of a cipher and a key. The same cipher can produce an almost limitless number of outputs with different keys values, allowing secure communication even if the cipher itself is known to hostile third parties . There are different types of cipher known for cryptographic functions, but for the purpose of this work, they will be divided into two, the ones that use the same key for both encryption and decryption, known as *symmetric key cryptography*, and the ones that use different, but related keys, known as *asymmetric key cryptography*.

3.1.1.1 Symmetric key cryptography

According to STALLINGS (2014), the symmetric encryption, was the only type of encryption used prior to the invention of the public-key encryption method in the 1970s, and because of being the first type of cryptography used, the symmetric encryption can also be referred to as conventional encryption, or as single-key encryption, since it uses the same key for both encryption and decryption, as mention before. The symmetric key cryptography can be divided into classical and modern encryption techniques.

Figure 2 – Basic operation of a cryptosystem



Source: author. Person icon icons by Icons8

The classical techniques address the first encryption methods used in history and some of its basic concepts are used in modern encryption techniques TANEMBAUM; WETHERALL (2014). In classical encryption methods, there are two main types of algorithms, *substitution*, and *transposition*.

In substitution methods, each character or group of characters in the plaintext is replaced by a different character or a group of different characters. The *Caesar cipher* is the simplest and earliest known example of the use of substitution encryption and was named after Julius Caesar, who allegedly used it to protect messages of military significance. The cipher works replacing a letter for another letter three positions further along in the alphabet. For example, the letter **a** would be replaced by the letter **D**, the letter **b** by the letter **E**, the letter **c** by the letter **F**, and so on (It is common practice to write the plaintext in lowercase letters and the ciphertext in capital letters). A generalization of the Caesar cipher is a cipher that replaces a letter by another letter k positions further along in the alphabet. For today's technology, a ciphertext created using Caesar cipher is easily broken, and because of that, it is not secure to use it in a real application. Others examples of substitution cipher are Vigenère, Playfair and Hill (STALLINGS, 2014).

In transposition methods, instead of replacing the characters as in the substitution methods, the characters are rearranged, performing some sort of permutation with the characters of the plaintext. The simplest example of transposition ciphers is the **rail fence** technique (STALLINGS, 2014). According to TALBERT (2006), the rail fence cipher is a special case of columnar transposition using only two columns (key length is two). For example, to encipher the message "meet me at the library today" with the rail fence cipher, we will write the following:

The ciphertext is then assembled by writing the columns from left to right. The resultant ciphertext is "MEMATEIRRTDYETETHLBAYOA". Other authors, such as STALLINGS (2014), define rail fence as a technique in which the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows. The following illustrates the idea, enciphering the same message:

Reading off the rows, the resultant ciphertext is "MEMATEIRRTDYETETHLBAYOA", the same as in the columns example. This sort of ciphertext would be trivial to cryptanalysis. Increasing the key size and/or adding random permutation cycles of the columns would increase the complexity of this scheme, making it harder for attackers to crack the

m	e
e	t
m	e
a	t
t	h
e	l
i	b
r	a
r	y
t	o
d	a
y	-

m	-	e	-	m	-	a	-	t	-	e	-	i	-	r	-	r	-	t	-	d	-	y
-	e	-	t	-	e	-	t	-	h	-	l	-	b	-	a	-	y	-	o	-	a	-

encrypted message.

The modern encryption techniques can be divided into two groups, *stream ciphers* and *block cipher*. Stream ciphers encrypt data in small chunks at a time. This small chunk can be as small as 1 bit or 1 byte. This type of cipher can run very quickly and usually uses very low complexity hardware. The encryption key in the stream cipher is often combined with an initialization vector, also called *IV*, which could never be the same when starting the cipher, otherwise, an attacker could e able to determine the encryption key.

Block ciphers are capable of encrypting larger chunks of data, or block, at a time. Usually, the block size is often 64, 128 or 256 bits. If a block is too short, then a pad is added to complete that block to its full size. Two important concepts, proposed by Claude Shannon, related to block cipher are *confusion* and *diffusion*. **Confusion** seeks to make the relationship between the ciphertext and the encryption key very complicated, which means that the resulting ciphertext should look very different from the encryption key. An attacker shouldn't be able to deduce the encryption key by analyzing the ciphertext. On the other hand, **diffusion** seeks to establish a very complex and complicated relationship between plain and ciphertext, in which the output should depend on the input in a complex way. A minor change in the input should cause a dramatic change in the output. The most important block cipher today is AES (Advanced Encryption Standard) cipher. It was introduced in 2001 to substitute the widely used cipher, at the time, DES (Data Encryption Standard).

The AES works on blocks of 128 bits or 16 bytes. The key length may vary between 128, 192 or 256 bits (16, 24 or 32 bytes). The input for either encryption or decryption is a single 16-byte block of plaintext data. The data is depicted as a 4 x 4 square matrix (input matrix), filled vertically from left to right. The algorithm works on N rounds and the number of rounds depends on the length of the key, which is presented in table 1. The input matrix, in each round, goes through 4 transformation functions: SubBytes, ShiftRows, MixColumns, and addRoundKey. The initial data in the input matrix is transformed in each round until the last when the output matrix is formed. The encryption key also goes through a process called key expansion, in order to generate a different 16 bytes key for each round. The first round or *Round 0*, the only transformation is a **XOR** operation between the input matrix and key from the Round 0. For every other round, the

sequence is the same, SubBytes, ShiftRows, MixColumns and addRoundKey, except for the last round in which the MixColumns transformation does not occur.

Table 1 – Number of rounds based on the key length

Nº of rounds	Key length (bytes)
10	16
12	24
14	32

Source: STALLINGS (2014)

3.1.1.2 Asymmetric key cryptography

The development of asymmetric key cryptography, also called public-key cryptography, according to (STALLINGS, 2014), is the greatest and perhaps the only true revolution in the entire history of the cryptography. Until its creation, all cryptographic systems have been based upon elementary tools of substitution and permutation. The public-key cryptography really provides a radical departure from symmetric key cryptography. Firstly, the public-key cryptography ciphers are based on mathematical functions instead of substitution and permutation. The second and more obvious difference between them is the use of a pair of different keys for encryption and decryption by the asymmetric cryptography ciphers instead of the single key scheme used in symmetric cryptography ciphers. The key pair is composed of a private key (sk) that must be kept in secret by the owner and a public key (pk), derived from the private key, which as the name suggest, should be made public and can be shared with others.

An important characteristic that the asymmetric algorithms should uphold is that it must be computationally infeasible to determine the sk given the cryptographic algorithm and the pk. If the attacker manages to access samples of the ciphertext, along with knowledge of the algorithm and the pk, even then the attacker should not be able to determine the sk. Other requirements for public-key cryptography are:

- It should be computationally easy for any party to generate a pair of keys (private and public keys);
- It should be computationally easy for a sender to generate a ciphertext using the receiver's public key;
- It should be computationally easy for the receiver to decrypt the resulting ciphertext using its private key to recover the original message;

The most successful example of the public-key algorithm so far is an algorithm developed by Ron Rivest, Adi Shamir e Len Adleman at MIT a first published in 1978 (RIVEST; SHAMIR; ADLEMAN, 1978). The **RSA** (Rivest-Shamir-Adleman) since its publication reigned supreme being the most widely accepted and implemented general-purpose approach to public-key encryption (STALLINGS, 2014). The power of the RSA algorithm is based on the use of large prime numbers . The key generating process is done as follow:

1. Select p and q where p,q are both large prime numbers and $p \neq q$;

2. Calculate $n = pq$. A typical size for n is 1024 bits;

3. Calculate $\phi(n) = (p-1)(q-1)$, where $\phi(n)$ is the Euler totient function;

4. Select e such that e is a relative prime to $\phi(n)$ and less than $\phi(n)$;

5. Calculate $d \equiv d^{-1}(mod\phi(n))$;

The sk will then be consisted of {d,n} and the pk of {e,n}. Eq. 1 and Eq. 2 presents the equations for encryption and decryption for a plaintext block M < n and a ciphertext block C:

$$C = M^e \mod n; \tag{1}$$

$$M = C^d \mod n; \tag{2}$$

Other examples of public-key algorithm are the Elliptic Curve and Diffie-Hellman Key Exchange.

3.1.2 Digital Signature

The digital signature, like the handwritten signature, is a way to verify the identity of the sender and it serves basically three purposes:

- **Authentication**: The receiver can confirm the sender identity;
- **Non-Repudiation**: The sender can not deny having written the message afterward;
- **Integrity**: The digital signature ensures that the message was not altered in any form during the travel from the sender to until it reaches the receiver;

Digital signatures are commonly used for authenticating software, financial transactions and in another kind of sensitive message where forgery or tampering are important to be detected. They are also common among email users [1]. Next, the hashing process is present, as it is commonly used in the digital signature process, then the digital signature process will be presented in more detail.

3.1.2.1 Hashing

Hashing is the process of converting input of any length into a fixed size array of numbers and letters, using mathematical functions. The output of this process is called *hash value* [2] and the function used to convert the input into the hash value is called *hash function*. Hashing algorithms have to attend some requirements to be considered useful :

- **Unique hash value**: A hash algorithm must ensure that for any different message that is used as an input in the hashing process the result must be different;
- **Hashing speed**: The hash algorithm should be reasonably fast. On the other hand, it shouldn't be too quick otherwise it will be easy to break;

[1] What is digital signature? - Sunny classroom - url: `https://www.youtube.com/watch?v=TmA2QWSLSPg`

[2] The "hash value" is also known as "fingerprint" or "message digest"

- **Secure hash**: The hash function is made to be a one-way function, which means it should be practically impossible to determine the message from the hash value. Another point is that a small change in the input message must generate a hugely different hash value;

A hash algorithm is said to be broken when an attacker can generate the same hash value from an authentic but for an altered message, for instance, a fake document. This is called **hash collision**. This is the reason why the algorithm shouldn't be too quick, otherwise, it would be fairly easy for an attacker to create a hash collision.

MD5 was a hash algorithm largely used in the past [3], but had to be replaced because it was broken. Some of the MD5 hashes can be reversed using only google search, and for that reason, it is not considered to be a reliable hash algorithm for cryptographic purposes anymore. In recent years, the widely used hash algorithm is the Secure Hash Algorithm (SHA). It was the remaining standardized hash algorithm by 2005. SHA is based on the hash function MD4 (STALLINGS, 2014), the previous version of MD5. The robust versions of SHA used today are the SHA-256, SHA-384 and SHA-512 which returns a hash value size of 256, 384 and 512 bits respectively, which makes practically impossible for an intentional or unintentional hash collision to happen, taking into account the modern computational capacity and not taking into account quantum computing.

3.1.2.2 Signature

An example of the process of digital signature is represented in Figure 3. In the example, Alice whats to send a signed message to Bob. Firstly, the plaintext is hashed. If the hashed algorithm used is really strong (i.e. SHA 256 and above), then if the message is altered in any form during transmission, the receiver, in this case, Bob will know. Then, the hash value will be signed using Alice's private key (sk_A). When the plaintext and the signature are received by Bob, then the plaintext will be hashed using the same algorithm that the one used by Alice, generating a hash value. This hash value is compared with the one obtained decrypting the signature using Alice's public key (pk_A). If both hashes values are the same, then Bob does not have any reason to not believe that Alice is the real author of that message and that message was not tampered in any way during its transmission, indicating that the message is legit.

3.2 Localization Verification Concepts

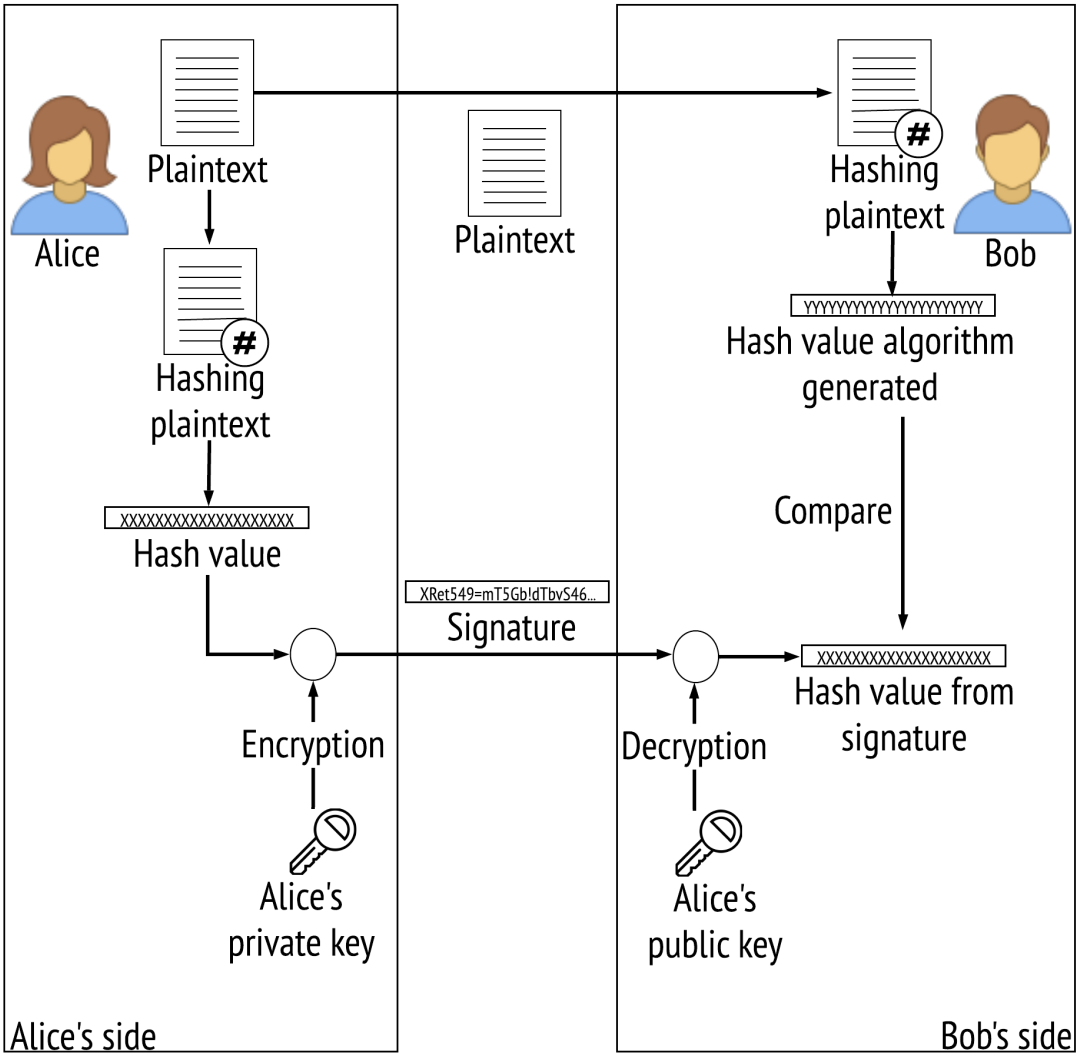
Mobility and transportation (Uber), Location-based game (Pokemon Go) and Supply chain are examples of applications that rely on asset/user position. Inaccurate assets/user position implicates a 'malfunction' of the application. For instance, if a user of mobility and transportation is placed by the location system in a different block or street, it could cause a delay in the journey or cancellation because the driver and the passenger could not find each other.

In general, applications are depended on the GPS to provide them the user/asset location, however, although widely used, there are a few known issues with the system such as:

- **heavy battery usage**: For devices with energy constraints problems, such as IoT devices and drones, the high energy consumption from GPS is a problem, decreasing the operating life of the device;

[3] It is still used today but it is not considered secure anymore

Figure 3 – Signature process



Source: author. Person icon icons by Icons8

- **Spoofing**: A lot of research was conducted in this area and spoofing GPS communication is fairly easy to achieve nowadays. The spoofing process is used to transmit to a GPS receiver, through a legitimate-appearing false GPS signal, a wrong GPS position. This has a huge negative impact on location depending on the application;

- **Difficult for indoor usage**: When in an indoor environment GPS signal is either nonexistent or too weak and inaccurate to be considered reliable for a position-dependent application;

The biggest problem with begin depended on GPS technology is that there is no back system other than manual navigation methods utilizing environmental sensors such as for instance, radar and celestial navigation[4]. Commercial solutions protocols, such as

[4]Boeing Patent

FOAM[5], Platins [6] and XYO network [7], are decentralized solutions blockchain-based to provide proof of location system, and in the same way that happens to cryptocurrency (Bitcoin, Etherum, etc) the agents that validate the transaction (in this case, the position), are rewarded by their work.

In the mobile network area of research, the position verification system has been extensively studied (BOEIRA; ASPLUND; BARCELLOS, 2018). Different proof-of-location mechanisms have been presented in diverse mobile environments, such as STAMP (WANG et al., 2016), APPLAUS (ZHU; CAO, 2011), VOUCH (BOEIRA; ASPLUND; BARCELLOS, 2018), VOUCH++ (BOEIRA; ASPLUND; BARCELLOS, 2019), among others. All cited protocols implements, in different ways, a way of validating the user/neighbor position, which is the purpose of the proof-of-location concept.

[5]https://foam.space/
[6]https://platin.io/
[7]https://xyo.network/

4 RELATED WORKS

The open nature of wireless networks makes this type of data transmission more susceptible to cyber attacks than wired communications (ZOU et al., 2016). In order to mitigate this risk in mobile ad hoc networks, particularly in VANETs, different approaches exploring single or combined security mechanisms have been proposed in the literature. This chapter describes state-of-the-art research in this field, with a particular attention to those applied to UAV networks. The literature review is organized in two major categories based on their research topic: the first addresses works about authentication mechanisms, and the second, works about position verification mechanisms. In each of these categories, the articles were also organized following their targeted network, from MANETs to FANETs.

4.1 Reviewed Authentication Mechanisms

In (DOSS et al., 2018), the authors proposed a novel technique called accurate prevention and detection of jelly-fish attack detection (APD-JFAD) in mobile ad-hoc networks (MANETs). The jelly-fish attack is a type of DoS attack, one of the most serious attacks that affects the normal working of MANETs. The proposed technique combines an authenticated routing-based framework and a Support Vector Machine (SVM) based technique to detect the malicious behavior of nodes by observing the quality of packets that reached the destination. APD-JFAD is tailored for MANETs, which are composed of nodes with lower speed and lower degrees of mobility than drones. This significantly affects the network topology, and communication, resulting in a negative impact on the proposed mechanism performance.

The use of blockchain for an authentication mechanism was explored in (FERRER, 2019; JENSEN; SELVARAJ; RANGANATHAN, 2019; AGGARWAL et al., 2019) through the use of different versions of blockchain, such as public versions as Bitcoin (FERRER, 2019) and Etherum (AGGARWAL et al., 2019), and a private version named Hyperledge Fabric (JENSEN; SELVARAJ; RANGANATHAN, 2019). In the blockchain encryption scheme, techniques such as public key cryptography and digital signatures are accepted means for proving the identity of specific agents in a swarm of robots (FERRER, 2019) or in a swarm of UAVs (JENSEN; SELVARAJ; RANGANATHAN, 2019; AGGARWAL et al., 2019). All agents have their public keys stored as a block inside the blockchain, and they will maintain an updated copy of the blockchain, having access to all the other agents' public keys stored in the blockchain network. In this way, digital signatures can provide entity authentication and data origin authentication between agents.

Although blockchain technology can provide data confidentiality and entity validation for a drone swarm, making them suitable for trust-sensitive applications has its limita-

tions. If a large number of robots are deployed for a very long time, the blockchain could be expanded to a point where the agents would not be capable of maintaining a copy of the full ledge anymore. Also, the time to process a new block takes on average 10 minutes. In addition, on the most widely used version of the blockchain, the bitcoin, users normally wait until two or three blocks are appended to the blockchain to confirm their transactions. Taking into consideration that a UAV has on average around 25 minutes of flight autonomy, the use of blockchain in the way it is today is not cost-efficient.

The high mobility of flying nodes brings new challenges to the current security protocols applied in general mobile networks, such as vehicular networks. In (ISLAM et al., 2016) the authors propose a fast and secure group key establishment protocol in order to facilitate forming groups and guaranteeing key freshness, key confidentiality, and members authentication. Their proposed protocol consists of two phases: initialization and post-deployment. During the initialization phase, individual security components are loaded into the UAVs, including their IDs, pubic and private keys, as well as their signatures. After that, an exchange of encrypted and signed request and joining messages is performed in order to allow a member to join a group providing a group key through a secure and private channel. The authors have proven protocol robustness by a complete analysis in their proposed mechanism. However, this mechanism was not implemented in either a simulated or a real environment, failing to demonstrate if the proposed authentication mechanism is feasible to be used in a resource-constrained environment such as the one the UAVs are part of.

In (WALIA; BHATIA; KAUR, 2018), the authors focused their work on presenting an authentication mechanism to detect malicious nodes in Flying Ad-hoc Networks (FANETs). The malicious node used a Sybil attack to trigger a Distributed Denial of Service (DDoS) attack. During network initialization, the central unit controller (CUC) will send Internet Control Messages Protocol (ICMP) packets to all nodes. These nodes will reply the ICMP packets, and send neighbors information to the CUC, which starts analyzing it. If two nodes have the same identification, but different neighbors, then the CUC marks them as intruders and starts monitoring their identifications. The node that changes its identification will be marked as malicious and also the responsible for the DDoS attack. By using the NS-2 network simulator, the authors have shown that this method generates maximum throughput as compared to other methods, as well as generates less routing overhead and packet losses. Nevertheless, the paper lacks a complete explanation of the authentication mechanism, which affects its replication.

Securing a network of drones through authentication mechanism is also addressed in (ALI et al., 2020). The authors presented *i*TCALAS, which is an improved scheme based on the a temporal credential based anonymous lightweight authentication scheme (TCALAS (SRINIVAS et al., 2019)) for Internet of Drones (IoD). *i*TCALAS uses lightweight symmetric key primitives and temporal credentials to protect drones and sensitive data collected by drones in an IoD. Although presenting promising results and scalability capabilities, the authentication schemes uses an centralized ground station server, which is responsible for verifying the authentication request, exposing the mechanism to a single point of failure.

4.2 Reviewed Position Verification Mechanisms

Recently, there has been an increase in the number of location-based applications, and it is common that these applications provide rewards to the user for visiting a specific

venue. This also creates an incentive for dishonest users to falsify their position in order to get undeserved rewards. To solve this issue, the work reported in (REZA NOSOUHI et al., 2018) proposed *SPARSE*, a distributed mechanism that provides secure and private Location Proof (LP) generation and verification for mobile users. In this mechanism, the system performs a witness selection mechanism by which some witnesses are chosen and qualified to generate LPs for a specific prover. The proof is then assessed and verified by an authorized entity known as the verifier.

A similar approach is presented in (FERREIRA; PARDAL, 2018). In this work, the authors propose a decentralized witness-based proof-of-location system for mobile devices. The system relies on different techniques for location estimation and on witness devices to testify the presence of the user's device. The proposed solution was implemented in Huawei P9 Lite devices. Although presenting promising results, both solutions are highly dependent on a high density of witnesses, which is not ideal for FANETs that are networks with potentially low density of nodes (TAREQUE; HOSSAIN; ATIQUZZAMAN, 2015). Moreover, the solution presented in (FERREIRA; PARDAL, 2018) demands a considerable amount of packets to determine the node position accurately, which is not suitable for a high mobility environment with sudden disconnections, packet losses, and permanent network partitioning (OUBBATI et al., 2019).

In the context of VANETs, the work reported in (BOEIRA et al., 2017) detailed the dangerous implications of the Sybil attack over a vehicular platoon. With this attack, a malicious node manages to introduce falsified vehicle identities into the platoon. An attacker may use these multiples identities to overload the platoon leader, which would have to handle false information. In a more dangerous scenario, the malicious node could inject erroneous beacons, causing a road accident. A countermeasure named $Vouch$ is present in (BOEIRA; ASPLUND; BARCELLOS, 2018). Vouch is a proof-of-location mechanism tailored for VANETs. Vouch uses a centralized proof-of-location and plausibility system to detect a Sybil attack in a vehicular platoon. A vehicle that requires a proof of its location, called prover, would ask for a proof of location to a Road Side Unit (RSU), which is called the proof provider. Once the prover received the signed proof from the proof provider, it will broadcast it along with the position beacon to the other vehicles in the platoon. The other vehicles are called verifiers. The verifiers then use this proof of location to estimate the prover's location in subsequent beacons and verify if the position sent by the prover is plausible or not. The proposed solution is not ideal for the military domain, as addressed in this current paper, because it has a single point of failure, due to the centralized approach based on the RSU, which if destroyed, the mechanism would not work. Furthermore, even considering that the proof provider (RSU) cannot be destroyed, it cannot be considered always reliable, because it may be compromised, then the entire system will become compromised.

In (BOEIRA; ASPLUND; BARCELLOS, 2019) $Vouch+$ is introduced. It is an improvement from Vouch, previously presented in (BOEIRA; ASPLUND; BARCELLOS, 2018). Instead of depending exclusively on previously installed roads infrastructure (RSUs), Vouch+ presents a decentralized protocol for the obtention of the proof of location. In Vouch the only trusted proof provider was the RSUs, but in Vouch+, besides the RSUs, a vehicle (proof provider) in the vicinity can assess the location of the prover (the vehicle that asks for the proof of location). The proof is then disseminated to the verifier, which are nodes that will use this proof in other to determine the plausibility of the prover's position. This decentralized approach brings the advantage for vehicles to prove their locations to neighbors beyond their sensing range. Although the presented

mechanism is an enhanced version of the Vouch, it does not eliminate the single point of failure related to the proof provider, because it also assumes that the entity (RSU or vehicle nearby), that will provide the proof of location, it is not compromised. Also another important difference is that Vouch+ assumes that the proof provider vehicle has a certain type of sensor to assess the position of the prover.

In the provided literature review, the existing works in authentication and position verification mechanisms presented above are not suitable for FANETs, due to the combined high mobility, node density and privacy constraints. To cope with the requirements of this type of ad hoc network, this work develops and assesses a FANET-tailored identity and location verification mechanism. UAVouch was designed to support these requirements without overloading the communication channel. The combination of these mechanisms in the proposed scheme are proven to effectively detect position falsification attacks. Table 2 summarizes the comparison of this proposal and the analyzed related work.

Table 2 – Summarization of authentication and position verification proposals

Related works	**Addressed problem**	**Network**	**Architecture**	**Proposed mechanism**	**Addressed attack**
(ISLAM et al., 2016)	High mobility degree and fast topology changes	FANETs	Centralized	Authentication	Sybil
(REZA NOSOUHI et al., 2018)	User's privacy preservation and position verification scheme design	MANET	Distributed	Position verification	-
(FERREIRA; PARDAL, 2018)	Position verification scheme design	MANET	Decentralized	Position verification	-
(DOSS et al., 2018)	Detection of a sort of DoS attack (Jelly-Fish)	MANETs	-	Authentication	Jelly Fish (DoS)
(BOEIRA; ASPLUND; BARCELLOS, 2018)	High mobility and user's privacy preservation	VANETs	Centralized	Position verification	Sybil
(WALIA; BHATIA; KAUR, 2018)	Detection of Sybil nodes	FANETs	Centralized	Authentication	Sybil and DDoS
(BOEIRA; ASPLUND; BARCELLOS, 2019)	Location assurance in cooperative transportation systems	VANETs	Decentralized	Position verification	Sybil
(AGGARWAL et al., 2019)	Privacy and security issues in the Internet of Drones (IoD)	FANETs	Decentralized	Authentication	-
(FERRER, 2019)	Trustful identification among swarm members	FANETs	Distributed	Authentication	-
(RODRIGUES et al., 2019)	Security strategies for resource constraint devices	FANETs	-	Authentication	Sybil, DoS and impersonation
(ALI et al., 2020)	Securing drones and sensitive data collected in IoD	FANETs	Centralized	Authentication	Multiples
UAVouch This proposal	**Identification of malicious nodes access in UAV network**	**FANETs**	**Distribuited**	**Authentication and Position verification**	**Sybil and impersonation**

5 APPLICATION SCENARIO

The challenges related to the management of bandwidth, latency, and battery power restriction faced by employing resource constrained devices, like drones, for real-time video stream applications, such as surveillance or military reconnaissance missions, are extensively addressed in the literature. However, the vast majority of these works focus on solving the problems associated to these restrictions (ZACARIAS et al., 2017; SEHRAWAT; CHOUDHURY; RAJ, 2017; HUSODO et al., 2019; PAUCAR et al., 2018; ENGBERTS; GILLISSEN, 2016; CHOWDHERY; CHIANG, 2018), leaving aside the security challenges in designing multi-UAVs applications (LIN et al., 2018; SHAKERI et al., 2019). Especially in military applications, securing the network is of prime importance. The security mechanism for this type of application must be efficient, but at the same time be as lightweight as possible, so that the generated overhead by does not negatively impact the performance of the ultimate mission goal, i.e. video streaming.

Consider a military reconnaissance mission performed by a military cell composed of an armored ground vehicle and a number of drones, which are circulating around the armored vehicle. The line of sight of the crew inside the armored ground vehicle might be limited by different factors, such as vegetation, and uneven terrain topology. When the drones are placed as shown in Figure 4a, they extend the crew's ability to monitor their surroundings. We now proceed to describe two attack scenarios in this setting.

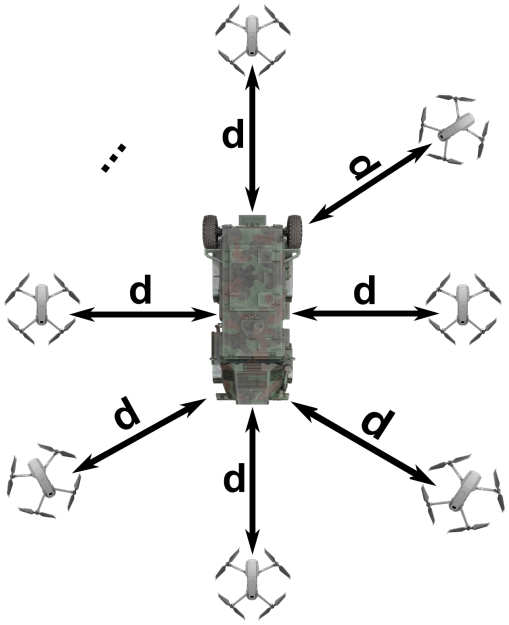
In this setup, the drones should fly at altitudes that combined with their horizontal distance to the armored vehicle, make them stay in the communication range of the armored ground vehicle. However, most of the time, they cannot communicate with all the other drones in the network, as illustrated in Figure 4b, which means that they have intermittent connection mostly with their direct neighbors.

5.1 Scenario 1

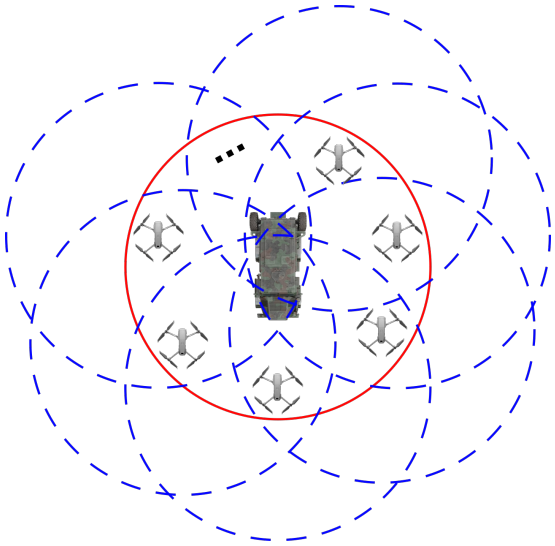
In the first scenario, the threat model is composed of an attacker that impersonates an authentic drone of the target cell. The sequence of events in this attack are represented in Figure 5. First, the malicious drone approaches a distant drone of the cell as represented in Figure 5a. The legitimate drone is then captured through a physical attack and has its credentials stolen (FOTOUHI et al., 2019) as represented in Figure 5b. The malicious drone uses the stolen credentials to assume the identity of the legitimate drone, returning to the network to start disseminating deceitful information. However, we assume that the attacker is not able to replicate the future mobility of the captured drone.

Figure 4 – Scenario structure

(a) Illustration of the drones positioning in relation to the armored vehicle



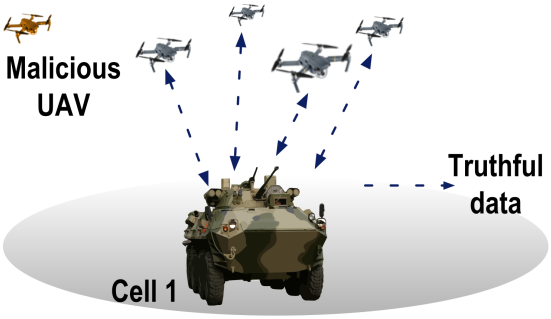
(b) Communication range illustration



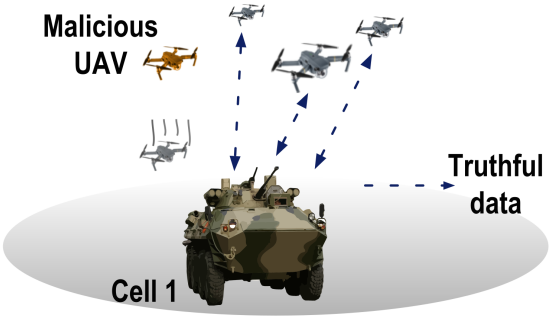
Source: author

Figure 5 – Impersonation attack illustration

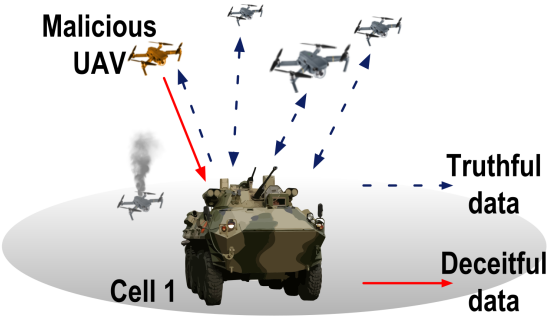
(a) Malicious UAV approaching a cell



(b) legitimate UAV captured/shot down



(c) Impersonation attack successfully concluded



Source: author

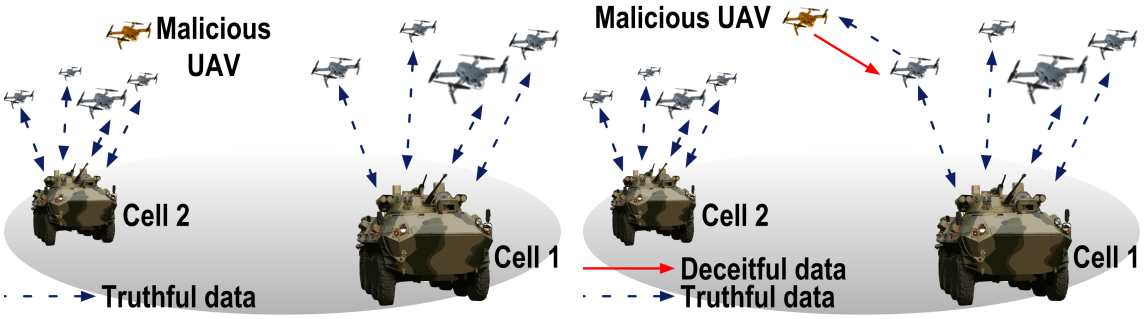
5.2 Scenario 2

The second scenario consider a more challenging situation involving more than one cell. While cell 1 is progressing from one location to another, it can encounter and interact with other cells to exchange information and expand their exploration and/or surveillance range. An attacker can take advantage of this feature to impersonate an entire cell and disseminate deceitful information. The attacked cell could be redirected into a trap due to the deceitful data and has its technology stolen. This scenario involves a Sybil attack, which is represented in Figure 6. In the Sybil attack, the malicious node would impersonate more than one drone. In this second scenario, a malicious drone takes advantage of

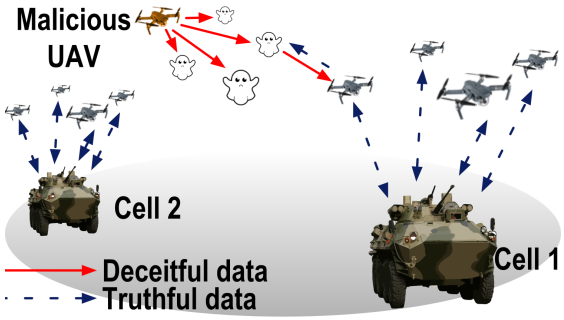
Figure 6 – Sybil attack illustration

(a) Malicious UAV approaching a cell

(b) Malicious UAV authenticates as a legitimate UAV



(c) Sybil attack successfully implemented



Source: author

the stolen identity of a drone from another cell, for instance cell 2 in Figure 6, to approach cell 1, as represented in Figure 6a. The malicious drone then uses the stolen identity to authenticate itself with cell 1 and to get their session key, as illustrated in Figure 6b. After it manages to establish communication with cell 1, the malicious nodes make it look like this cell is connected with the legitimate cell 2, so it impersonates all the drones in the cell 2 to make the attack more convincing, as represented in Figure 6c.

6 PROPOSAL

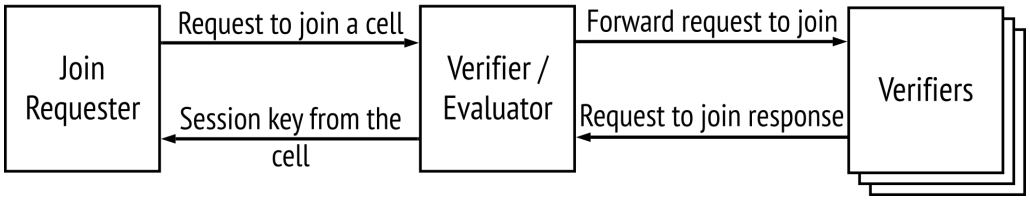
Security defense mechanisms are often classified into three categories, *prevention*, *detection*, and *response* (see eg,. (GIRALDO et al., 2017)). Even though prevention strategies are necessary, attackers with enough resources can bypass these mechanisms. Thus, detection strategies are also needed to identify anomalous behavior and attacks in the system. Response mechanisms should be activated when an attack was successful, providing measures to mitigate the damages. This chapter describes the design of the proposed solution for the drone identity and position validation. The solution is divided into a prevention strategy composed of a *public-key based authentication* **mechanism** and detection strategy composed of a position validation **mechanism** that includes a **protocol** used for position validation, as well as a **classifier model** to detect inconsistencies in the movements of the nodes.This proposal is named $UAVouch$, a reference to Vouch (BOEIRA; ASPLUND; BARCELLOS, 2018), an approach proposed to address Sybil attacks in platoon of ground vehicles traveling on roads, and to the drones, as they are UAVs.

6.1 UAVouch scheme overview

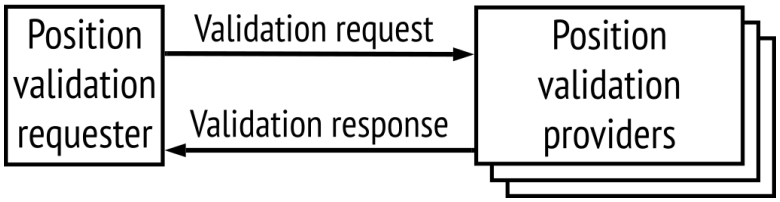
Figure 7 shows how entities interact with each other in the UAVouch scheme. Figure 7a illustrate the interaction among the entities in the authentication mechanism and Figure 7b represents specifically the interaction between the entities in the validation protocol, which is part of position validation mechanism together with a classifier model. Firstly, in the authentication mechanism, the *requester* is a drone that requests to join a cell in which it is currently not a member. The request to join the cell is received by one of several *verifiers*, which are the entities responsible for ensuring that the *requester* is authorized to join the network. The *verifier* that received the request perform the authentication check and broadcasts its decision. This is received by the other verifiers in the cell who will also broadcast their own decisions. At the end of the chain, the *evaluator* is the entity responsible for counting the votes, and if the majority of the *verifiers* in the cell vote to admit the *requester* into the cell, the *evaluator* will send the session key to the *requester*, concluding the authentication mechanism. All drones are *verifiers* in their cell, but the drone that receives the request directly from the *requester* will also become an *evaluator*. The purpose of the authentication mechanism is to avoid intruders to enter the cell, as well as provide a secure way to identify a friendly cell traveling nearby. This is of paramount importance in a military scenario. If a cell erroneously connects to an enemy cell mistaken by a friendly cell, the consequences could range from disclosing confidential intelligence information to losses of human lives.

In the position validation mechanism, the UAVs will continuously send their location information to each other through *pose packets*. In addition to a common header, the pose

Figure 7 – Relation between the entities in the UAVouch
(a) Entities involved in the authentication scheme



(b) Entities involved in the validation protocol



Source: author

packet usually carries information about position coordinates and direction of movement (*pose* parameter), and can also carry other types of information such as speed and acceleration, depending on how the protocol was designed. A pose packet can optionally include a position validation request. If the pose packet includes a position validation request, this will trigger a position validation protocol. In this protocol, a *position validation provider* is responsible for validating position validation requests and replying whether the sent location was legitimate. All drones inside a cell are position validation providers for the other members of the same cell. Once the *position validation requester* received more than 50% of the replies from the remaining drones in the cell, it will consider that everyone has computed and stored its position validation. If the pose packet does not include a position validation request, a classifier model is activated that judges whether the claimed location is plausible in relation to previous locations. Both mechanisms will be described in more detail below.

6.2 Premises, Assumptions and Notation

The proposed scheme was developed based upon a few premises:

- **Security**: It is assumed that a drone receives its asymmetric key pair, all the asymmetric public keys from the all previously registered drones, and an unique session key from its cell in a secure environment during the network initialization (e.g. during mission initialization in the base);

- **Inter-Cell Communication**: There is an exclusive communication channel between the armored ground vehicles which has a larger range than the channel between the drones and drone-to-armored vehicle;

- **Intra-Cell Data Forwarding**: Every different received packet is forwarded in order to reach the whole network. The number of hops is determined based on the number of drones and topology of the network;

- **Positioning**: It is assumed that the drones from each cell receive, periodically, the updated position of the armored ground vehicle of their cell and the offset of the others drones from the ground vehicle. The position update rate is the same as position validation request;
- **Flight pattern**: The drones exhibit flight patterns which are hard for an outsider to predict and mimic. This could for example be achieved through a combination of complex trajectories and specific physical dynamics of the drones.

Regarding to the notation used in this chapter, as presented in Table 3, the asymmetric public and private keys from an entity X are represented respectively as pk_X and sk_X, and the symmetric key from a given cell X is represented as k_X. The signature process is represented using $\mathrm{sign}(m, y)$, where m is the message and the y is the key used to sign the message. The encryption operation is represented by $\mathrm{aenc}(x, y)$ for asymmetric encryption of data x with key y and $\mathrm{senc}(x, y)$ for symmetric encryption of data x with key y. Table 4 presents the cryptographic notations used in both authentication and position validation mechanisms. Next, the proposed mechanisms are presented in details.

Table 3 – Cryptographic notations

Notation	Description
pk_X	asymmetric public key from entity X
sk_X	asymmetric private key from entity X
k_X	symmetric key from cell X
$\mathrm{sign}(m, y)$	signature process of data m using key y
$\mathrm{aenc}(x, y)$	asymmetric encryption of data x using key y
$\mathrm{senc}(x, y)$	symmetric encryption of data x using key y

6.3 Authentication Mechanism

To simplify the presentation of the scheme, consider a scenario in which a cell A enters the communication range of cell B. The authentication mechanism is triggered when a drone d_B belonging to cell B receives a message from another drone d_A that belongs to cell A. The drone d_B (*Join requester*) then sends an identification packet (**iden**) carrying its public key (pk_B) and a timestamp of the message. The message m is signed ($\mathrm{sign}(\langle m, \mathrm{pk}_B \rangle, \mathrm{sk}_B)$) using d_B's private key, sk_B. The signature will be verified by d_A (*Verifier/Evaluator*), and if the signature is valid it will send a response packet (*idenResponse*) with all the header information signed using sk_A and encrypted using pk_B. The signature in *idenResponse* will be verified by d_B, and only if the signature is valid it will send a **reqJoin** packet to d_A requesting to join its network, as illustrated in Figure 8. The message is signed by d_B using sk_B and encrypted using the public key of d_A. Since d_A received the **reqJoin** packet directly from d_B, it should forward the packet (**reqJoinFwd**) to the its cell adding the *whosReq* parameter, so that the other drones inside the cell know that they are not receiving that packet directly from d_B. The **reqJoinFwd** message is encrypted using the session key k_A.

Every drone in cell A (*Verifiers*) will verify if d_B is an authorized drone by checking the signature in **reqJoinFwd** using the pk_B key acquired during network initialization.

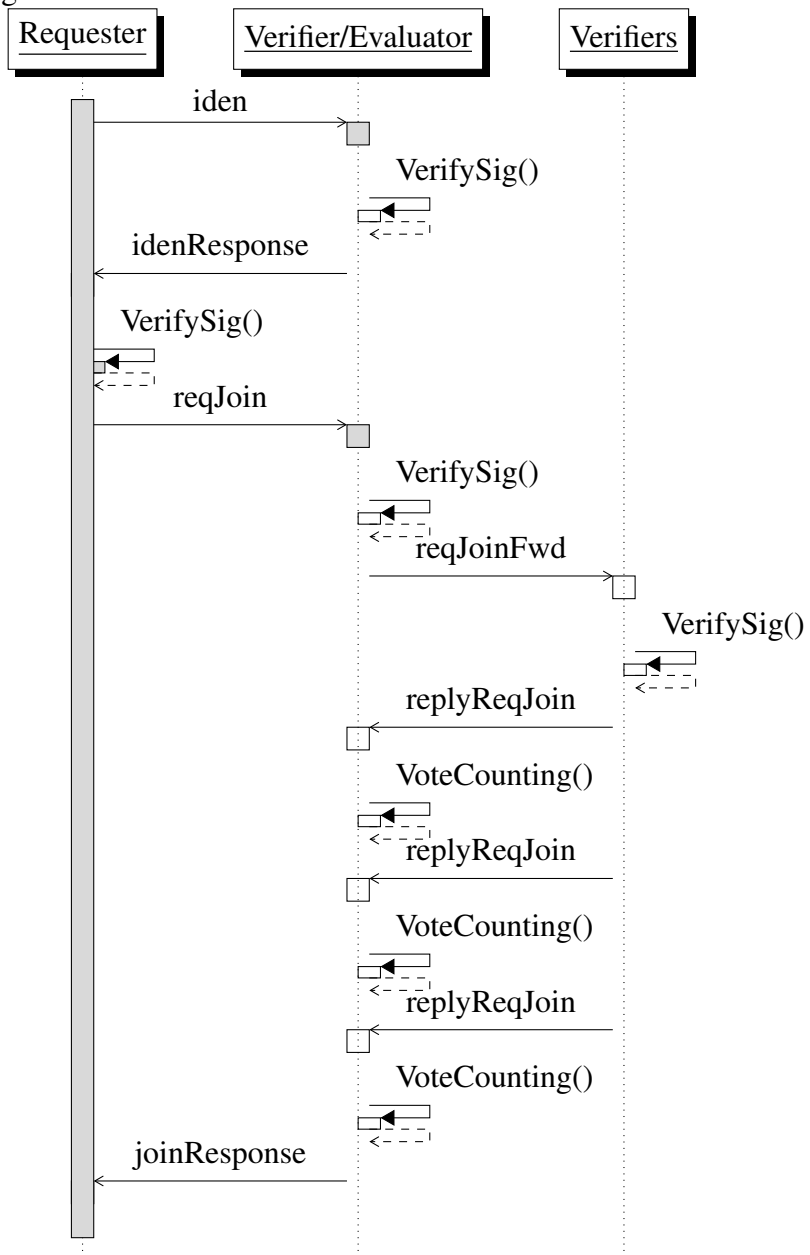
They will broadcast their decision to the network, sending a **reqJoinReply** packet, where the parameter *isAuth* states whether d_B is authorized or not. Due to the fact that at any given moment a cell could be handling multiple requests to join, the *whosReq* parameter in the **reqJoinReply** packet is used to identify whose request they are replying to.

If more than 50% of the network confirms the legitimacy of d_B, then a **sessionKey** packet carrying k_A is sent to d_B, the one who requested to join the network, otherwise, the node will be ignored by cell A. As d_A is the closest drone to d_B, it will act also as a *evaluator*, which means that, after verifying that more than 50% of the cell A considers d_B legitimate, d_A will be responsible for sending the **sessionKey** packet to d_B. Considering that packet collision may happen during this process, if d_B have not been granted access to the network in cell A after a period of time t_{Auth}, it will resend its request to join the network. If d_A is compromised, then the whole authentication mechanism is also compromised. To avoid this problem, a position validation mechanism is used to identify the intruder and stop it before it can harm the network as described in the following section.

Table 4 – Cryptographic operations

Symbols	**Description**
msgKind	The type of the message
nId_x	The unique identification of node x
t_x	Timestamp of entity x
seqNumber	Sequence number of the message
cell	The cell in which the drone is in
whosReq	Requester to join the network
isAuth	Authentication request response
pose	Quaternion containing coodinates x, y e z and orientation w
whosValReq	Position validation requester
valReply	Position validation reply
header	$<$ msgKind, nodeId, timestamp, seqNumber, cell $>$
idenHeader	$<$ pk_B, timestamp $>$
iden	$<$ idenHeader, sign(idenHeader, sk_B) $>$
idenResponse	$<$ aenc($<$ header, sign(header,sk_A) $>$, pk_B) $>$
reqJoin	$<$ aenc($<$ header, sign(header, sk_B) $>$, pk_A) $>$
reqJoinFwd	$<$ senc($<$ header, whosReq $>$, k_A) $>$
replyRequestJoin	$<$ senc($<$ header, whosReq, isAuth $>$, k_A) $>$
joinResponse	$<$ aenc($<$ header, k_A $>$, pk_B) $>$
posePkt	$<$ senc($<$ header, pose $>$, k_A) $>$
valReqReply	$<$ senc($<$ header, valReply, whosValReq $>$, k_A) $>$
verifySig()	Verify signature sign(y, x) of data using pk_B
voteCounting()	Authentication request response counting
validityCheck()	Execute the position validation calculation
valCounting()	Validation check responses counting
savePosVal()	Store valid position calculate

Figure 8 – Interaction between entities in authentication mechanism



6.4 Position Validation Mechanism

The proposed position validation mechanism is composed of validation protocol, which determines the interaction between the entities in the validation process, and a classifier model, which determines the position plausibility of pose packets that do not contain a validation request. The details of these two parts of the mechanism are presented as follows.

6.4.1 Validation protocol

The validation protocol is illustrated in Figure 9. When a drone sends a pose packet ($\mathbf{posePkt}$), it can also request validation of its location from its recipients. The *msgKind* parameter is used to identify if a position validation was requested or not. If a position validation was requested in the *posePkt* ($\mathbf{posePkt_{valReq}}$), the other drones in the same cell will verify the validity of the position based on the position (**avpos**) and heading angle θ of the armored vehicle, on the offset (**OS**), and on the mobility model of the drone which asked for the position validation. After calculating the position validation, the drones will send a reply (**valReqReply**) containing if the position is valid or not (*valReply*) and from who the position validation request came from (*whosValReq*). The requester will count the votes and if the majority of the network voted that the position is valid then the requester will consider that everyone has its position validation otherwise, the requester will send a new pose packet requesting a position validation. Packet collision may happen during the voting process, therefore if the requester does not receive more than half the votes after a period of time $t_{valReply}$, it will request a new validation of its location in the next pose packet and this voting round is discarded. The mechanism was implemented in a way that the position validation is activated in a frequency equal or less than the frequency of pose packets, which means that some position packets will not include a request for a validation of the sender's location. Every drone stores the valid position calculated to use it for the classifier model. When no position validation is requested in the pose packet, the classifier model will be the one responsible to determine if the position received is valid or not.

6.4.2 Classifier model

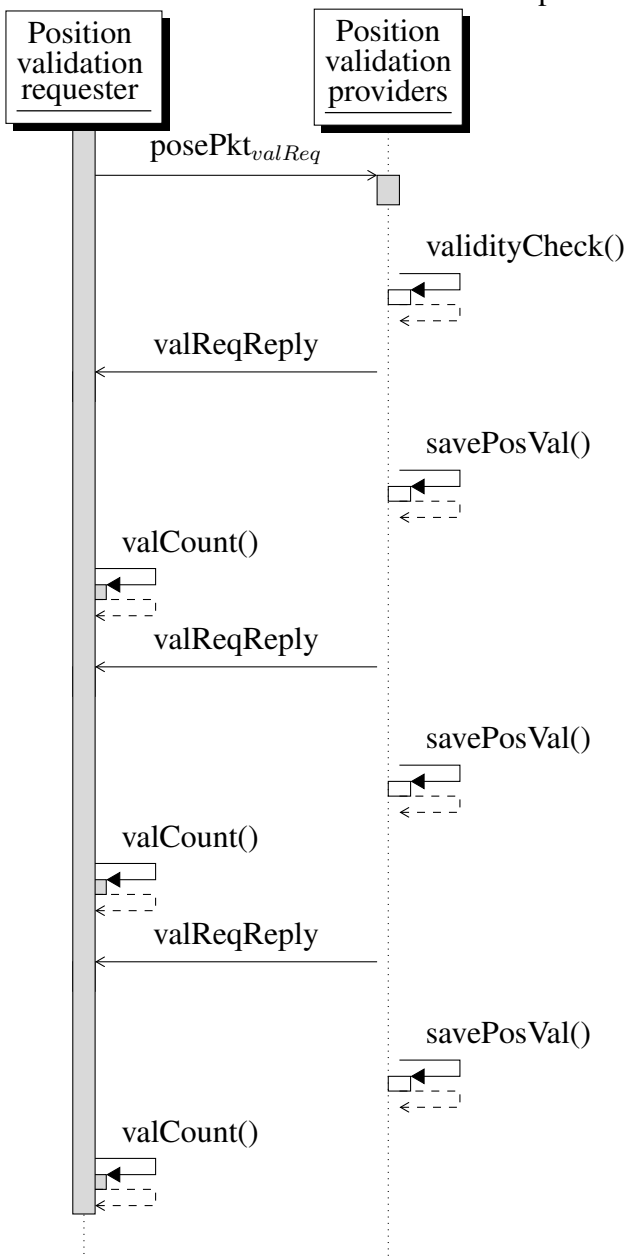
The classifier model, or position plausibility model, is activated when a drone received a pose packet that does not include a position validation request. The plausibility of the position is calculated based on the last position validation computed for the drone that sent the position. Due to the assumed accelerated movement model, the position estimation is determined as presented in 3 and 4.

$$\boldsymbol{S}_{max} = \boldsymbol{S}_0 + \boldsymbol{v}\Delta t + \frac{1}{2}\boldsymbol{a}\Delta t^2 \tag{3}$$

$$\boldsymbol{S}_{min} = \boldsymbol{S}_0 + \boldsymbol{v}\Delta t - \frac{1}{2}\boldsymbol{a}\Delta t^2 \tag{4}$$

Δt is calculated using the time difference between the timestamp in the pose packet and the timestamp of the last calculated position validated. $\boldsymbol{a}$ and $\boldsymbol{v}$ are respectively the maximum acceleration and medium velocity passed as a parameter in the simulation. The actual acceleration in the analyzed period of time is unknown, therefore, the precise position of the movement cannot be determined. Consequently, based on the acceleration parameter, it is possible to calculate a range in which the position should be if it moved

Figure 9 – Interaction between entities in movement plausibility model



using maximum acceleration or de-acceleration. Then, the plausibility is determined by checking if the position sent in the pose packet is within feasible boundaries. If the position is within feasible boundaries, it will be classified as *plausible* otherwise, it will be classified as *implausible*.

Regarding the communication between cells, every time a drone from a cell receives a position packet from another cell, it asks to its armored vehicle for the position of the armored vehicle from the other cell. This allows drones from one cell to validate the position of drones from other cells.

6.5 Supporting position data acquisition from another cell

As mention above, when 2 cells, A and B are connected, if a drone from cell A (d_A) received a position packet from a drone of cell B (d_B), d_A will need the position of the

armored vehicle of cell B (av_B) to calculate the validity of the position of d_B. To increase the coverage area, the connected cells would try to stay as far as possible from each other without breaking the communication connection. As a result of this distance between cells, d_A would not have direct communication with av_B. To obtain av_B's position, d_A could ask its position directly to d_B, but this node could have obsolete position information, which could cause miss calculation in the position validation, or it could be compromised, as in scenario 2 (be a malicious node impersonating cell B). For this reason, as a premise in this work, there is an exclusive and secure communication channel between the allied ground armored vehicles belonging to the different cells, through which av_B's position could be obtained. In this case, when d_A receives a position packet from d_B, it will request its armored vehicle for the updated position of av_B. Only after d_A receives this information from av_A, it is capable to calculate the validation of d_B' position.

6.6 Rejoining Process

During a reconnaissance mission, a drone may leave the cell to execute a given task. This is the case, for instance, when it has to check a given event or object close by the cell, but out of the range of the other nodes in the cell. If the duration of this disconnection exceeds a preset amount of position packets (n_d), the drone will be considered disconnected from the cell, having to be authenticated again when returning to the network. A long disconnection of a drone from the cell will also trigger the process of refreshing the session key. When the disconnected drone returns, it will be placed in a quarantine period (Δt_q), and during this period, its movement pattern will be analyzed by the cell members. The Δt_q is the time between a preset amount of position packets (n_q), where $n_q \geq n_d$. The disconnected drone will only receive the new session key if its movement pattern matches with the movement pattern expected by the other members of the cell. Concerning the session key refreshing process, the armored ground vehicle is responsible for generating a new session key, which will be then sent to each drone encrypted using the drone public key and signed by the armored vehicle.

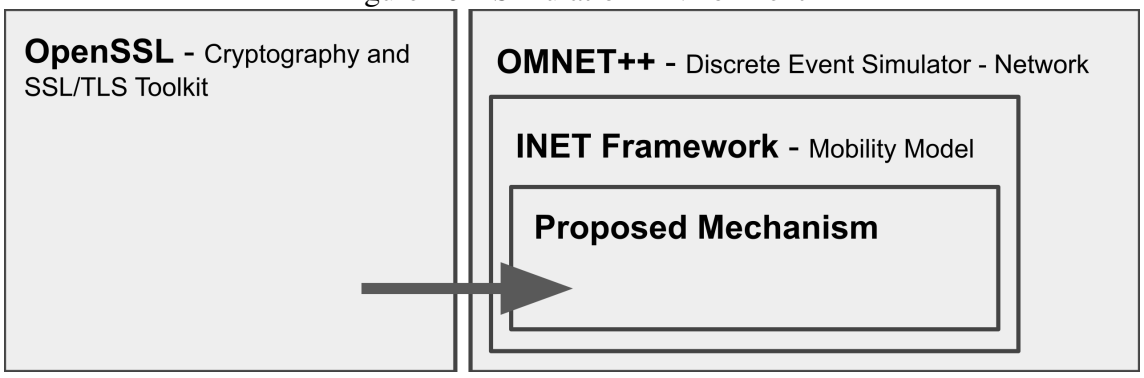
7 EXPERIMENTS AND RESULTS

This chapter presents the experiments used to validate the proposal. Details about the simulation environment are presented, followed by the evaluation metrics. Then, the specific parameters used in the performed simulation runs are presented, followed by the discussion of the acquired results.

7.1 Simulation environment

The proposed scheme was evaluated by performing simulations in INET, an OMNet++ based framework. OMNet++ is a network simulator for implementing and testing novel networking solutions. By using the INET framework, it is possible to gather valuable results considering realistic mobility models and wireless communication constraints. As part of the solution, OpenSSL APIs were used to compute the required cryptographic operations. Figure 10 depicts the relation between the elements included in the simulation environment, such as frameworks and libraries.

Figure 10 – Simulation Environment



Source: author

7.2 Evaluation metrics

The evaluation of this proposal was performed following the two different scenarios described in Chapter 5. The first scenario focuses on evaluating the effectiveness of the scheme in detecting an intruder inside a cell. The second scenario focuses on evaluating the effectiveness of the scheme in detecting the intruder impersonating another cell, which means the attacker is outside the victim cell.

Two kinds of pose packets are considered (containing the position information of a

drone): a *falsified pose packet* which is a packet that in which the position has been manipulated by an attacker; and a *correct pose packet* which is a packet that was not manipulated. The position validation providers will categorize each pose packet as being either *plausible* or *implausible*. In regard to the notation: a *true positive* (TP) is when a falsified pose packet is classified as implausible; a *true negative* (TN) is when a correct pose packet is classified as implausible; a *false negative* (FN) is when a falsified position packet is classified as plausible; and a *false positive* (FP) is when a correct position packet is classified as implausible. According to these definitions, the metrics used to evaluate UAVouch are the following.

- **True Negative Rate (TNR) or Specificity**: The percentage of correct pose packet correctly classified as plausible;

$$\text{TNR} = \frac{TN}{TN + FP} \tag{5}$$

- **True Positive Rate (TPR) or Sensitivity**: The percentage of falsified pose packet correctly classified as implausible.

$$\text{TPR} = \frac{TP}{TP + FN} \tag{6}$$

- **Accuracy**: The percentage of correctly classified pose packets;

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

- **Retransmission rate**: The percentage of retransmitted pose packet;

$$\text{Retransmission rate} = \frac{r}{s}, \tag{8}$$

  where r is the total number of pose packet resent and s is the total number of pose packets sent

- **Overhead**: The percentage of the increase in packets sent in the network due to the application of the UAVouch scheme;

$$\text{Overhead} = \frac{\alpha - \beta}{\alpha}, \tag{9}$$

  where α is the total number of packets sent with UAVouch and β is the total number of packets sent without UAVouch

7.3 Simulation parameters

Table 5 presents the parameters which were considered either in scenario 1 or 2. For each combination of the presented parameters, 33 runs were executed using the simulator. A statistical power analysis (significance test) was conducted in the Minitab software to validate that a sufficient number of simulations was run. With a standard deviation and a maximum difference between means of 4.8, taken from the analysis of the simulation data, a significance level of 0.05 (α = 0.05) and a sample size of 33, one sample for each

run, the obtained power was 0.93. A commonly accepted value for the statistical power is 0.9.

The drone mobility model chosen for the performed simulations was a circular mobility model. This model is combined with the accelerated linear mobility of the ground armored vehicle which provides a spiral-like movement, as illustrated in Figure 11. This model was chosen because besides its trivial computation complexity, as defined in Equations 10 to 13, it is not trivial to be mimicked by a malicious node that does not know that this is the model being used and based only on visual observation of the movement. In a real scenario, an even more elaborate mobility pattern could be utilized, but our focus here is on the general mechanism. The center of the circular movement, represented by **cpos** is calculated using matrix rotation. After calculating **cpos**, the distance between the position sent (*pose*) represented as **dpos**, and the center represented as **r**, is calculated based on the distance between 2 points, as defined in Eq. 13. If **r** is inside the boundaries determined by the threshold for the radius of the circular movement, then the position is considered legitimate otherwise, it is considered false.

$$cpos_x = avpos_x + (OS_x cos(\theta) - OS_y sin(\theta)); \tag{10}$$

$$cpos_y = avpos_y + (OS_y cos(\theta) + OS_x sin(\theta)); \tag{11}$$

$$cpos_z = OS_z; \tag{12}$$

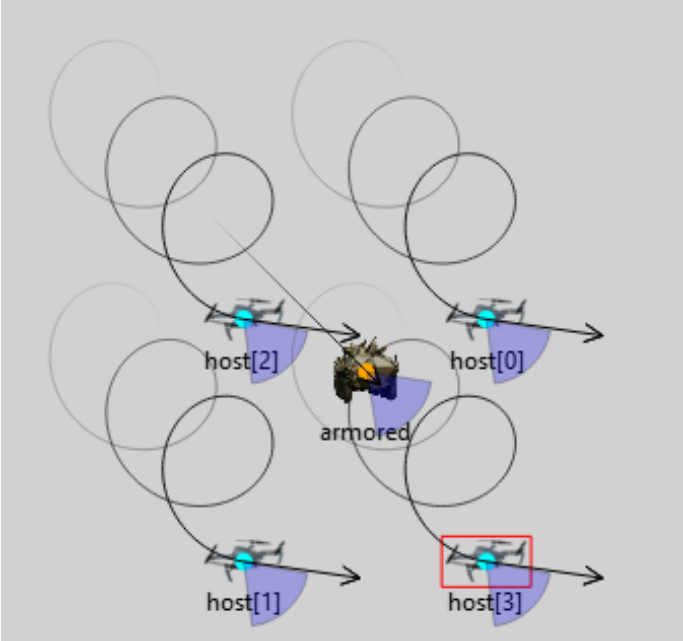
$$r = \sqrt{(cpos_x - dpos_x)^2 + ((cpos_z - dpos_z)^2 + ((cpos_z - dpos_z)^2}; \tag{13}$$

The simulation uses four drones, placed one in the front side of the armored vehicle, another in the back, one on the left side and last one on the right side. The small number of drones, reduces the number of direct neighbors and consequently the number of connections, thus creating a more challenging environment for the experiments.

Table 5 – Simulation parameters

Parameter	**Value**
Drone mobility model	Circular mobility
Number of drones per cell	4
Communication range	$\approx$ 1 [km]
Asymmetric cryptography	RSA 2048-bit key
symmetric cryptography	AES 256-bit key
Maximum acceleration	2.5 [m/s^2]
Simulation time	200 [s]
Position noise mean/σ	0/0.5 [m]
Plausibility check threshold	1σ, 2σ, 3σ, 4σ [m]
Position validation period	0.1, 0.2, 0.5, 1.0 [s]
Pose packet period	0.1 [s]
Packet maximum size	100 [bytes]
Armored vehicle velocity	20 [Km/h]
Attacker position offset	10 [m]

Figure 11 – Screenshot of a simulation run showing the movement trail combining the circular and linear mobility model



7.4 Results and Discussion

The results from the simulation experiments for both scenarios are presented in the following.

7.4.1 Scenario 1

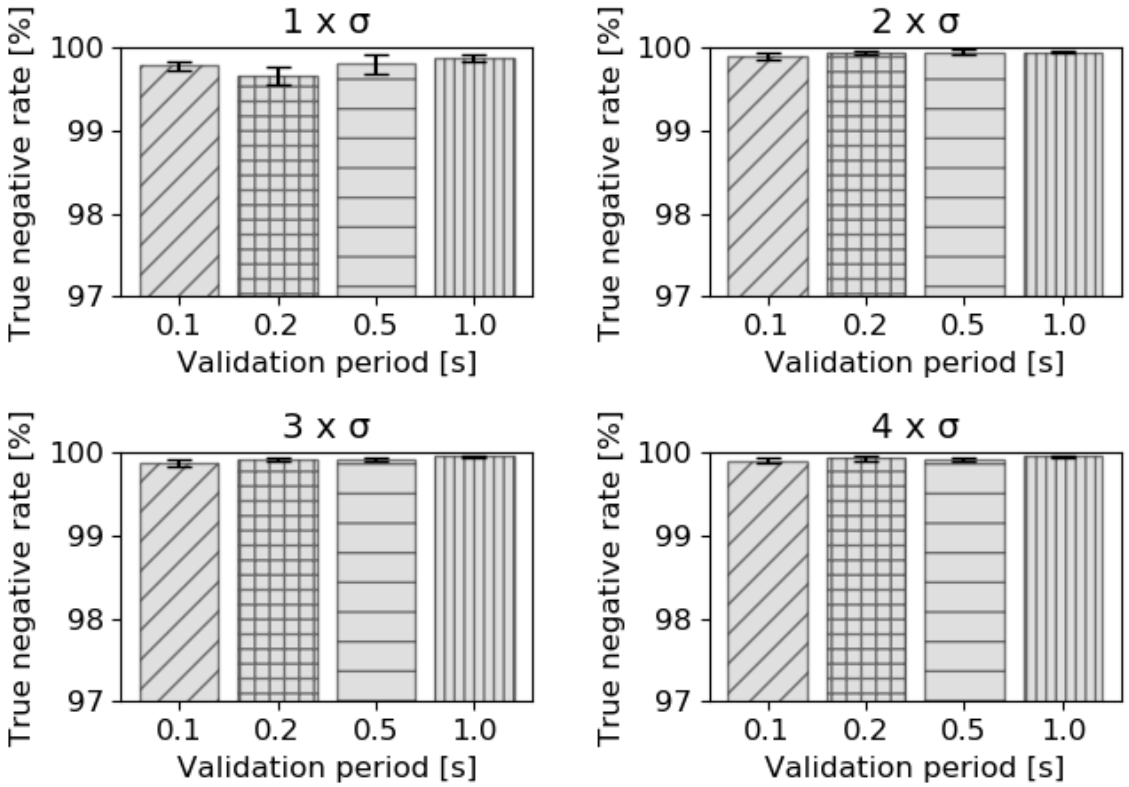
The purpose of Scenario 1 is to evaluate the effectiveness of the solution to detect an attack involving just one cell. At t = 30s, a drone inside the cell changes its setting and starts operating as the attacker, disseminating deceitful information and not being able to mimic the movement pattern. The error in the movement pattern is determined by the simulation parameter *attacker position offset*. The results are presented as follows.

7.4.1.1 True negative rate (Specificity)

Figure 12 presents the percentage of the correct position which was correctly classified by the mechanism. This represents how effective the mechanism is in identifying legitimate drones. Effectiveness measurements, like the next measurements, were taken based on the variations of the position validation and the plausibility check threshold. The position validation period is meant to evaluate the impact of using old validated coordinates to classify a drone. The plausibility check threshold is meant to evaluate how resilient the mechanism can be regarding position errors and is based on the standard deviation (σ) of the position noise. It was expected that, with a shorter position validation period, the *TNR* would be better because the position plausibility mechanism would always have the drone's most recent position coordinates, therefore the error caused by using old position coordinates, as occurs with longer position validation period, would be close to zero. It was also expected that with a shorter threshold, the plausibility model would have a higher sensitivity for error in the position coordinates, increasing the *FPR*, which means the mechanism incorrectly classifies a legitimate drone as malicious. However, from the simulation results presented in Figure 12, it is noticeable tiny fluctuations

in the true negative rates values regarding to variations in both threshold and position validation period values. This stems from the fact that the plausibility check designed for the circular mobility model presents a really high rate of correctly classifying the position of the legitimate drones, as presented in Table 6. As a consequence, this metric should not be considered for deciding which parameter combination is the most efficient for this mechanism.

Figure 12 – True negative rate from scenario 1



Source: author

Table 6 – True negative rate from scenario 1

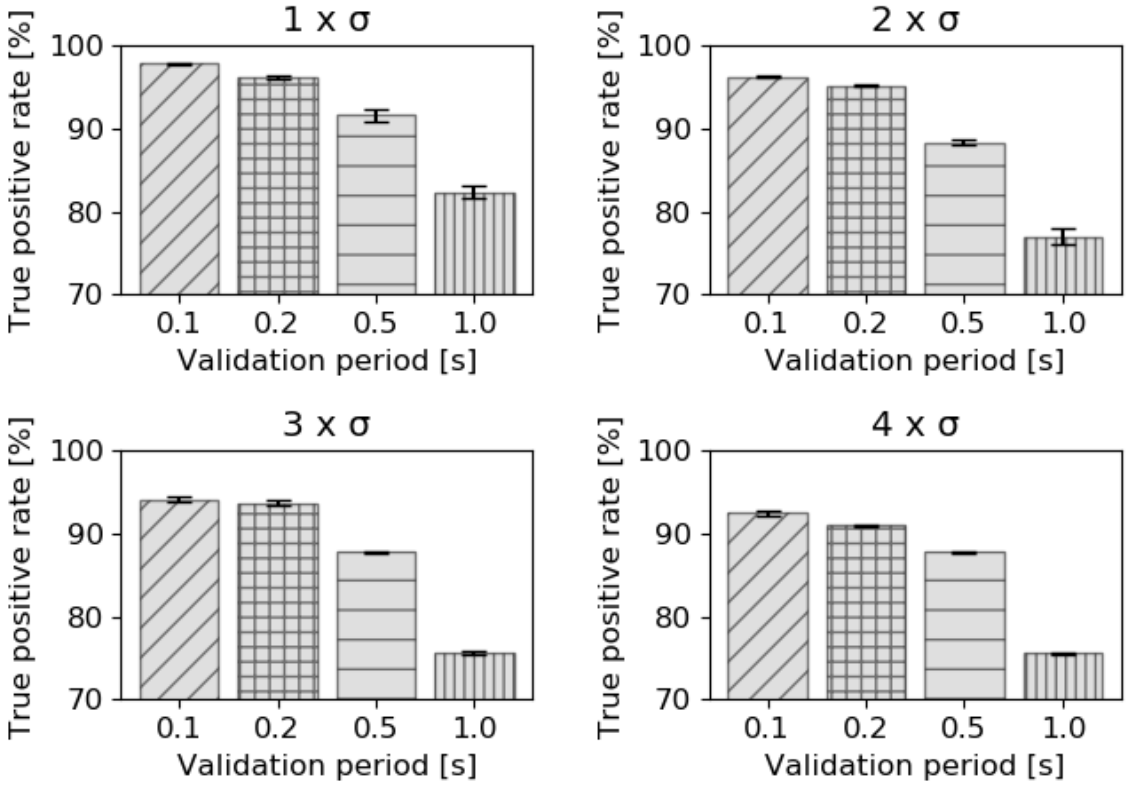
Threshold \ Validation period	0.1	0.2	0.5	1.0
1σ	99.79	99.66	99.81	99.88
2σ	99.89	99.95	99.95	99.95
3σ	99.87	99.93	99.92	99.96
4σ	99.90	99.94	99.93	99.96

7.4.1.2 True positive rate (Sensitivity)

Figure 13 presents the proportion of malicious nodes correctly classified by the mechanism. This represents how effective the proposed solution is in identifying a malicious node. As it was expected with TNR, TPR is sensitive to variations in the threshold and

position validation period values. As illustrated in Figure 13, it is noticeable that increasing the threshold, and consequently also the distance between the feasible boundaries, there is an increase in the percentage of incorrect positions classified as correct, negatively impacting the performance of the proposed mechanism. The same negative impact is perceived when there is an increase in the position validation period due to the use of old position coordinates as discussed above. Nevertheless, the mechanism reached high true positives rates, above 90% for some combinations of threshold and position validation period values, showing that the mechanism is reliable and robust to detect malicious drones. It is noticeable that the true positive rate is sensitive to changes in the threshold and in the position validation period. If the position validation period increases, the detection of a malicious node will decrease because the age of the proof affects the classification mechanism. Older proofs imply higher position estimation errors which imply in the mechanism accepting the erroneous position as a legitimate one and therefore, decreasing the malicious drone detection rate. The same concept is applied to the increase of the threshold, which means that with a bigger threshold, therefore a larger distance between boundaries, and more incorrect positions are accepted as plausible, diminishing the true positive rate.

Figure 13 – True positive rate from scenario 1



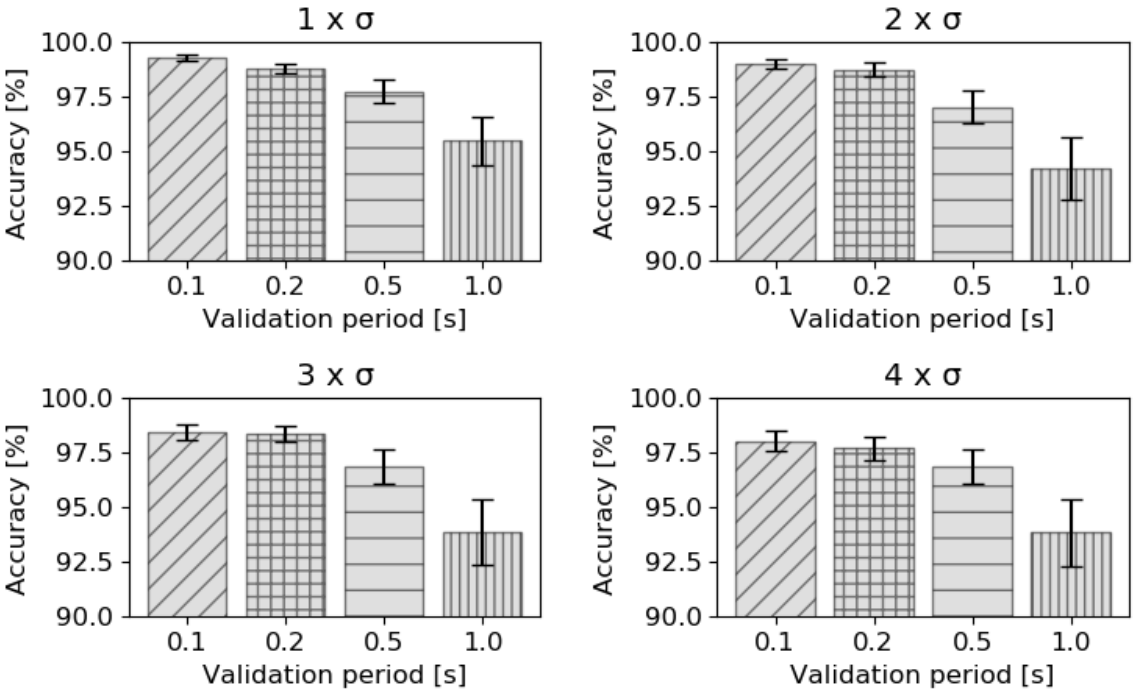
Source: author

7.4.1.3 Accuracy

Figure 14 presents the graph of mechanism accuracy for the first scenario. Based on Eq. 7, it is expected that the accuracy would represent an approximated combination between both rates previously presented. Therefore, it is possible to notice that the position

validation period and threshold parameters have a direct impact on the accuracy, as they impact in TPR and TNR values. Considering exclusively the metrics presented so far, the mechanism achieved a fairly high detection rate. For position validation periods of 0.1s and 0.2s, the detection rate was above 90% for 1, 2 and 3σ, and the overall accuracy was above 97.5%. This provides evidence of the UAVouch efficiency. The high accuracy values combined with both high TNR and TPR values demonstrates how good the proposal is in correctly classifying a malicious drone as malicious and properly identifying a legitimate drone.

Figure 14 – Accuracy from scenario 1

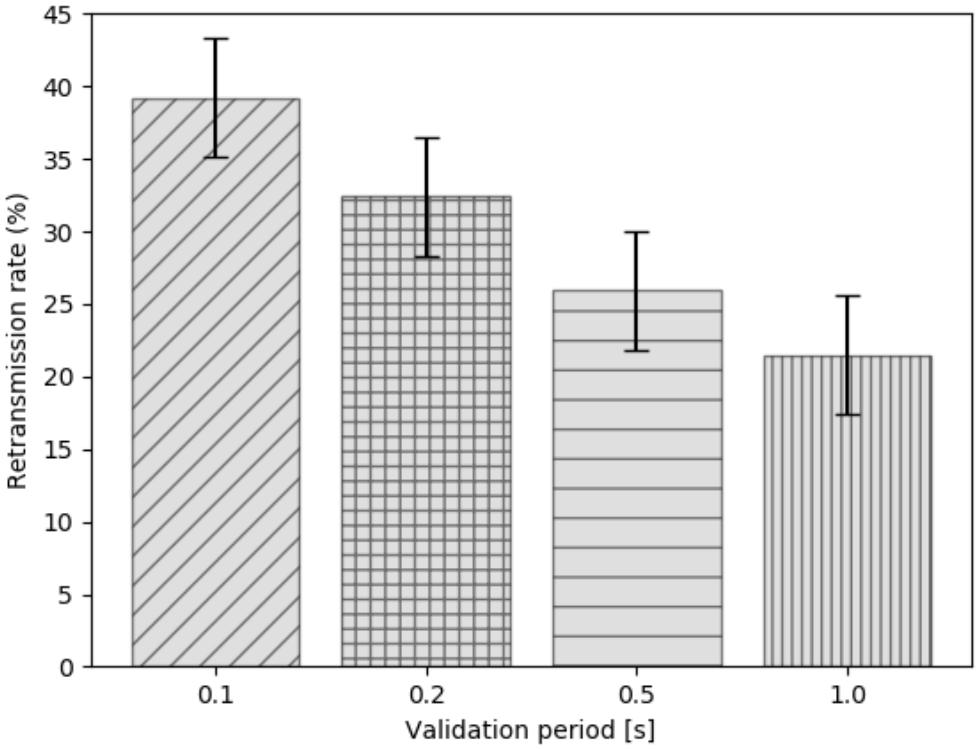


Source: author

7.4.1.4 Retransmission rate

In a distributed system, such as the one discussed here, the number of messages exchanged between the nodes is expected to be higher than in a centralized system. As dealing with wireless communications, this also leads to higher interference occurrences and packet collisions. Figure 15 presents the retransmission rate rate measured in the first scenario. As expected, the retransmission rate is directly related to the validation frequency. Having a higher rate of proofs, the number of transmitted packets also increases. As a consequence, the probability of packet collisions rises. On the other hand, the variation of the threshold value does not impact the probability of having packet collisions, as it does not change the number of transmitted packets. It is clear that the packet retransmission rate metric impacts the voting system, preventing it to sometimes reach a decision as occasionally not all the packets containing the votes are received by the position validation requester, impacting the overall performance of the proposed solution. Table 7 shows the impact of this metric, by measuring the percentage of validation requests that reached a decision. It is noticeable that the mechanism has a decision rate of 80% on average, representing that on 8 out of 10 requests a decision will be reached.

Figure 15 – Retransmission rate from scenario 1



Source: author

Table 7 – decision rate from scenario 1

Validation period / Threshold	0.1	0.2	0.5	1.0
1σ	0.7990	0.8084	0.8084	0.7656
2σ	0.7991	0.7954	0.8136	0.8048
3σ	0.7987	0.7977	0.8150	0.8079
4σ	0.8012	0.8046	0.8134	0.8140

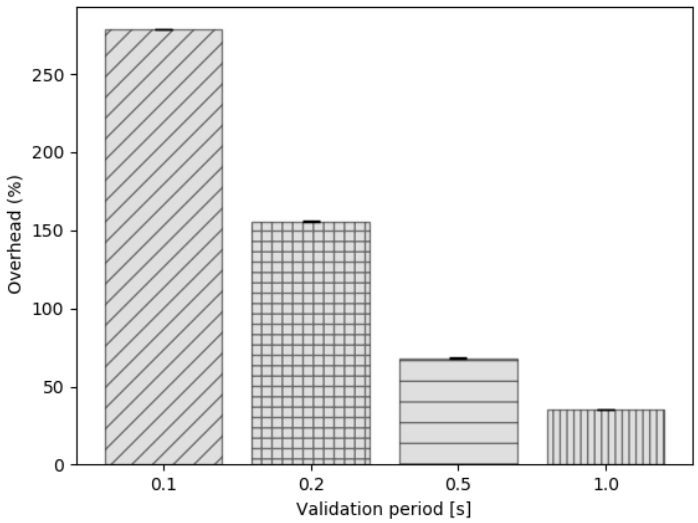
7.4.1.5 Overhead

Figure 16 presents the overhead introduced in the system by the validation mechanism. It is possible to notice that the mechanism overhead decreases as the position validation period increases. It is clear that the variation in the threshold value does not affect the number of packets being transmitted in the network, and as the overhead is computed based on the number of packets added to the network due to the UAVouch, the only parameter that affects the overhead is the validation period. Although the mechanism was responsible for a fairly high increase in the number of packets being transmitted, in terms of bandwidth consumption this number is reasonable. For the worst-case scenario, with a position validation request period of 0.1 s, and remembering that for each position validation request it is expected replies from each drone in the network (3 replies in this case study), this would represent an increase of 30 packets per node in the network, thus, 120 packets in total. Being the pose packet of the size of 60 bytes (on OMNet++), the data rate can be estimated of around 57,6 kbps, representing a very small bandwidth

consumption considering technologies such as 4G and WiMax, for instance.

As mentioned in Chapter 5, the security mechanisms designed for military reconnaissance applications have to be efficient avoiding negative impact on the performance of payload data transmission. With this requirement in mind, a trade-off between detection performance and overhead must be made to achieve the ideal combination of threshold and position validation period, in a way that the mechanism remains highly efficient, but without a significant increase in the imposed overhead.

Figure 16 – Overhead from scenario 1



Source: author

Based on Figure 14, it is clear that for smaller values of thresholds (1σ and 2σ) and smaller position validation periods (0.1 and 0.2) the detection of the malicious drone presents its best performance. Taking into consideration also the overhead, it also clear that the best combination between high detection rates and acceptable overhead is 1σ for the threshold and 0.2s for the position validation period. To better exemplify the performance for this particular combination of parameters, a confusion matrix is presented in Table 8.

Table 8 – Confusion Matrix (1σ and 0.2 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1662	7
	legitimate	40	1998

7.4.2 Scenario 2

As in scenario 1, in scenario 2 the effectiveness of the proposed solution to detect an attacker was evaluated, but this time the attacker is outside the cell. In this scenario, as soon as the simulation starts the malicious node attempts to connect with cell 1, assuming the identity of a node from another cell. As the malicious node manages to be

authenticated and connect with cell 1, it starts sending manipulated position messages to cell 1, impersonating the other nodes in its $fake$ network. The results obtained in the simulations for this scenario are presented in the following.

7.4.2.1 True negative rate (Specificity)

Table 9 presents the *TNR* for scenario 2. As in scenario 1, it is noticeable tiny fluctuations in the TNR regarding variations in both threshold and position validation period values. Furthermore, even with the increase in the number of nodes, there was only a tiny decrease in the TRN value from scenario 1 to scenario 2, demonstrating that the proposed scheme remained effective in identifying the legitimate drone correctly.

Table 9 – True negative rate from scenario 2

Threshold \ **Validation period**	0.1	0.2	0.5	1.0
1σ	99.76	99.60	99.85	99.87
2σ	99.92	99.96	99.94	99.96
3σ	99.91	99.96	99.95	99.96
4σ	99.91	99.96	99.95	99.96

7.4.2.2 True positive rate (Sensitivity)

Figure 17 presents the *TPR* from scenario 2. As in scenario 1, it is noticeable that the *TPR* is affected by changes in the threshold. As presented in table 10, increasing the threshold has a negative impact on the *TPR* . Nevertheless, unlike in the first scenario, the *TPR* does not change with the variation of the position validation period. This happens because in this scenario, the malicious drone is outside the cell network. Thus, every time a legitimate drone is requested to check the malicious drone position, it asks for its armored vehicle the position of the armored vehicle from the cell that the malicious drone is impersonating. Therefore, the position validation mechanism will always have updated information, and as a consequence, there will not happen the problem with stale information as happens in scenario 1. It is also noticeable that for all the threshold values, and for the position validation period value of 0.1s, the TPR is almost the same for both scenarios although the number of drones has increased from the first to the second simulation.

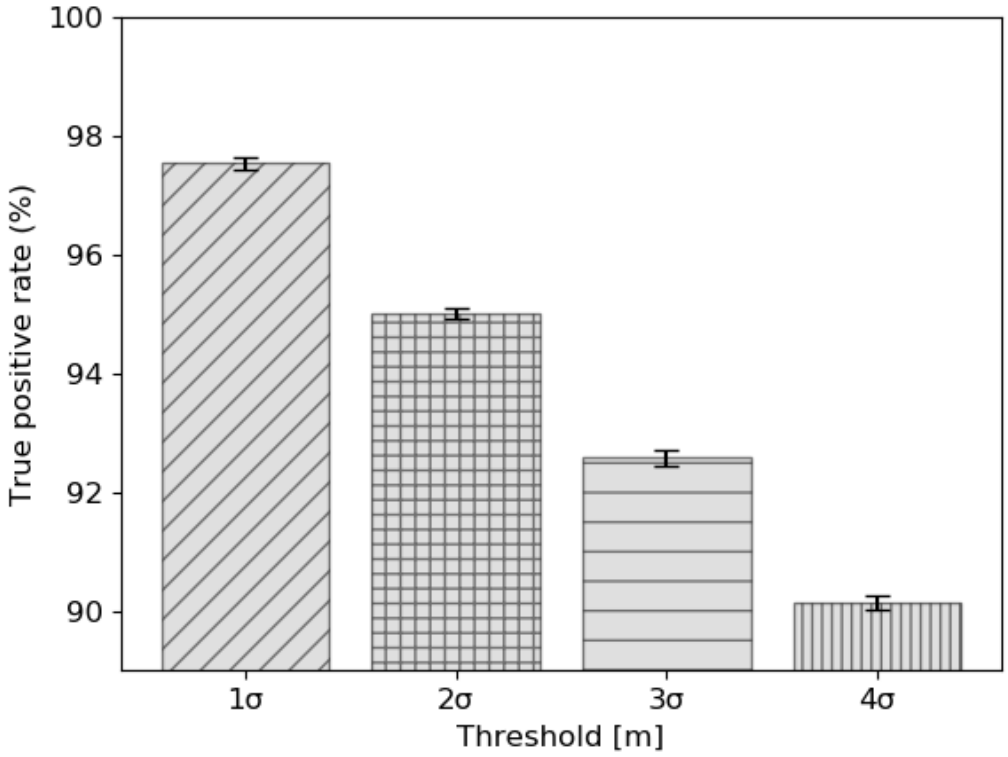
Table 10 – True positive rate from scenario 2

Threshold \ **Validation period**	0.1	0.2	0.5	1.0
1σ	97.55	97.50	97.52	97.57
2σ	95.03	95.05	95.13	95.13
3σ	92.58	92.61	92.59	92.63
4σ	90.14	90.16	90.20	90.17

7.4.2.3 Accuracy

Figure 18 presents the overall accuracy of the proposed solution regarding the Sybil attack on the second scenario. As in the first scenario, the accuracy is totally dependent on the threshold. However, in the second scenario, the accuracy appears to be more sensitive

Figure 17 – True positive rate from scenario 2



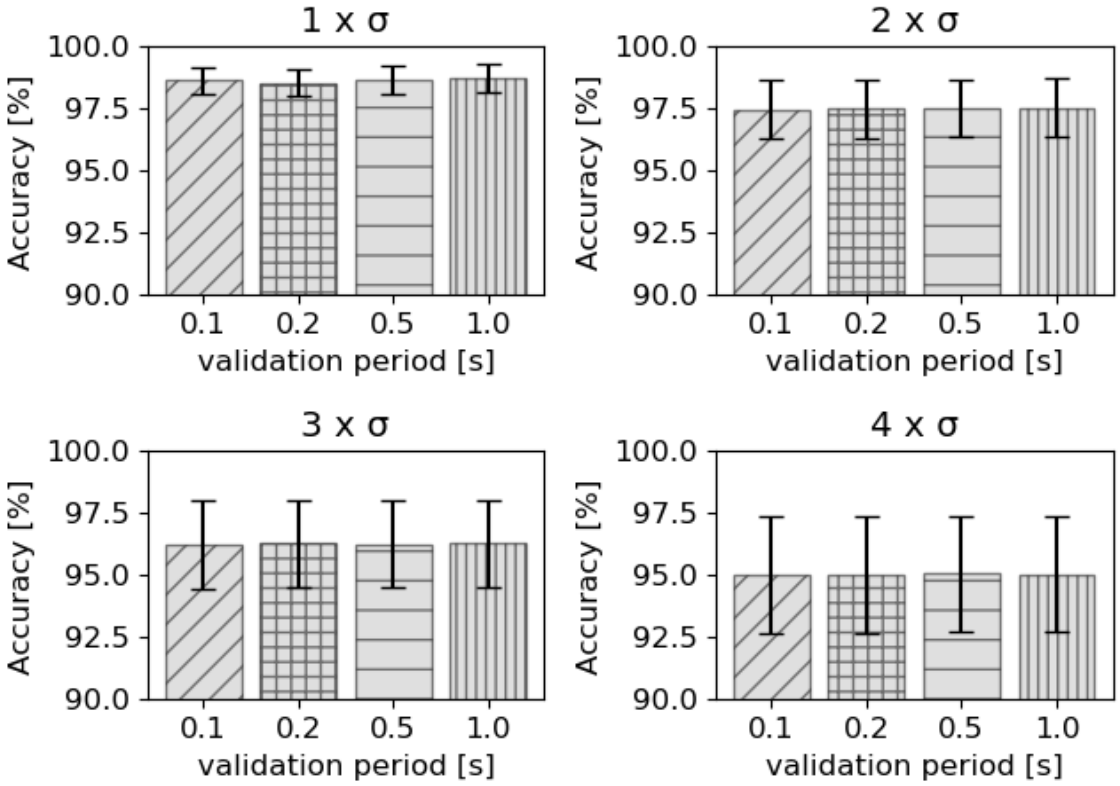
Source: author

to the variance of the threshold, due to the fact that the accuracy difference in Figure 14, based on the position validation period of 0.1s, between the first and the last graph, is around 1.5 %, but for the second scenario, this difference is more than 3.5 %. This is related to the fact that more packets are introduced into the network which results in an increase in the packet collisions, negatively affecting the efficiency. The other difference between the two scenarios is that in the second one, the accuracy does not have a relevant impact with the position validation period variation. In the first scenario, the impact in the accuracy by the variation of the position validation period was due to the fact that the true positive rate was affected by the position validation period variation. In the second scenario, as the true positive rate is not affected by the position validation period variation, this fact reflects in the accuracy rate.

7.4.2.4 Retransmission rate

Figure 19 presents the retransmission rate measured in the second scenario. As expected, the retransmission rate is higher in the second scenario than it was in the first. The main reason is that with more drones in the network, more packets are exchanges. Therefore, there is a higher possibility of occurring collisions. Although only an extra drone is introduced from one scenario to the other, the extra one is acting as it was 4 drones, consequently, it is like the network doubled its size, from 4 to 8 drones. In numbers, from the first to the second scenario, the retransmissions increased by approximately 15%. As for the first scenario, the impact of the increase in retransmissions was analyzed regarding the decision rate, as the results shown in Table 11. The mechanism reached a decision rate

Figure 18 – Accuracy from scenario 2



Source: author

of 67% on average, representing that about 7 out of 10 requests will reach a decision, one less than in the first scenario.

Table 11 – decision rate from scenario 2

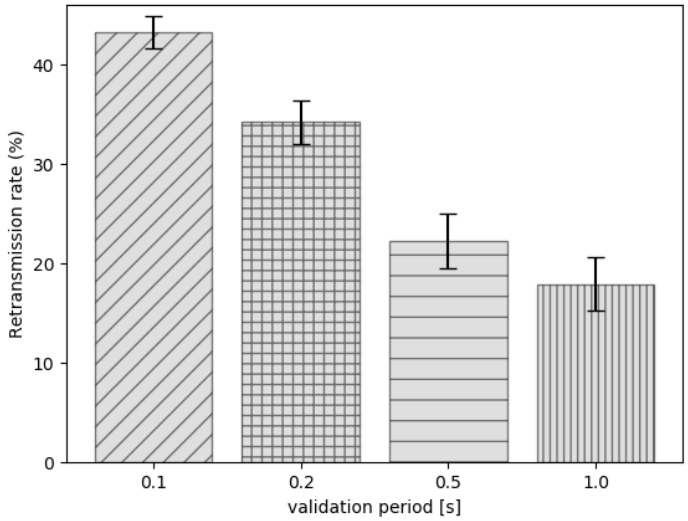
Threshold \ **Validation period**	0.1	0.2	0.5	1.0
1σ	0.6342	0.6249	0.7168	0.6997
2σ	0.6266	0.6219	0.7124	0.7044
3σ	0.6323	0.6306	0.7214	0.7063
4σ	0.6364	0.6316	0.7184	0.7146

7.4.2.5 Overhead

Figure 20 presents the overhead introduced by the position validation mechanism. As in the first scenario, it is possible to notice that the overhead is susceptible to the variation in the position validation period, but it is not affected by the variation in the threshold. Comparing the overhead graphs for scenarios 1 and 2, it is noticeable that there is a small increase in the overall overhead, which was expected due to a higher number of packets exchanged in the second scenario.As explained in the first scenario, although the second scenario also reached high percentages of overhead, comparing the data rate consumed by current widely used wireless technologies, these numbers are completely acceptable.

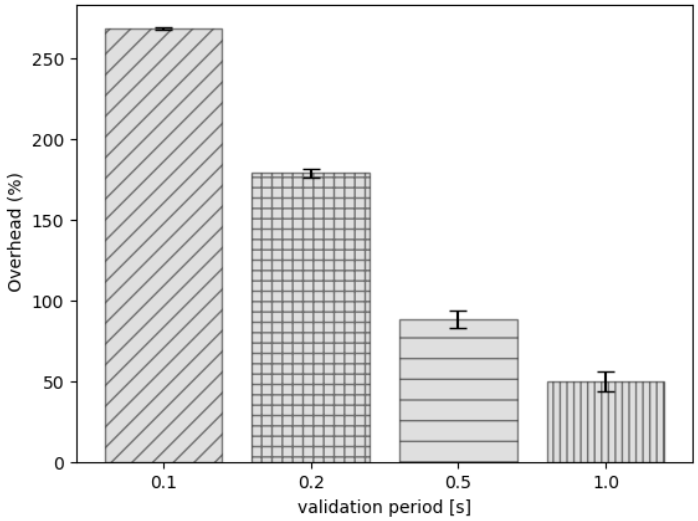
Based on the requirements discussed in scenario 1 and analyzing the results previously

Figure 19 – Retransmission rate from scenario 2



Source: author

Figure 20 – Overhead from scenario 2



Source: author

presented for scenario 2, it is clear that the best combination between high detection rates and acceptable overhead is for 1σ and 0.5s for the position validation period. To better exemplify the performance for this particular combination of parameters, a confusion matrix is presented in Table 12.

Table 12 – Confusion Matrix (1σ and 0.5 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1951	3
	legitimate	50	1997

8 CONCLUSIONS

This work presents a distributed scheme for identity and location validation combining an asymmetric key-based authentication mechanism with position validation mechanism for system using groups of drones. The proposal is evaluated using two attack scenarios, one for the Impersonation attack, with the intruder inside the cell, and the other for the Sybil attack, with the intruder outside the cell.

UAVouch presented a high accuracy, above 90% in detecting the malicious node inside (scenario 1) and outside (scenario 2) its network. Due to the distributed nature of the protocol, evaluation of packets retransmission and the overhead of the mechanism were presented. Results showed a retransmission rate bellow 50% for the worst-case scenario and an acceptable overhead in all simulated conditions, which demonstrated the viability of the proposed scheme. Because of the voting system used in the proposed scheme, an evaluation of the number of times the system reached a decision was made. UAVouch achieved acceptable decision rates for both scenarios, with a decision rate of 80% and 67% for scenarios 1 and 2 respectively.

Regarding futures directions for this work, there are a few possibilities that can be explored to improve the UAVouch scheme, particularly in its practical implementation, such as: *RSA key replacement*: Although being widely used, RSA key size can be a problem for hardware limited systems, such as the one in most drones. Replacing by an efficient algorithm, such as Elliptic Curves can help to improve the system for real deployment. *Lower layers*: A more thorough investigation on how a feasible long-range communication protocol, such as WiMax and LoRa, could affect the performance of the UAVouch mechanism. *Mobility model*: The mobility model has a great impact on the design of the movement plausibility check. Further studies can be conducted to test the UAVouch position validation mechanism against other mobility models. *Scalability*: Further studies must be conducted to evaluate the performance of UAVouch on networks with a higher number of nodes.

REFERENCES

ADAMS, C. Impersonation Attack. In: TILBORG, H. C. A. van (Ed.). **Encyclopedia of Cryptography and Security**. Boston, MA: Springer US, 2005. p.286–286.

AGGARWAL, S. et al. A new secure data dissemination model in internet of drones. In: ICC 2019 - 2019 IEEE INTERNATIONAL CONFERENCE ON COMMUNICATIONS (ICC), 2019. **Anais...** [S.l.: s.n.], 2019. p.1–6.

AGÊNCIA NACIONAL DE AVIAÇÃO CIVIL. **Registered drones in Brazil from 2017 to 2019**. Available at: <https://www.anac.gov.br/assuntos/paginas-tematicas/drones/quantidade-de-cadastros>. Acesso em: september 2019.

ALI, Z. et al. **Securing smart city surveillance:** a lightweight authentication mechanism for unmanned vehicles. **IEEE Access**, [S.l.], v.8, p.43711–43724, 2020.

ALTAWY, R.; YOUSSEF, A. M. **Security, privacy, and safety aspects of civilian drones**: a survey. **ACM Trans. Cyber-Phys. Syst.**, New York, NY, USA, v.1, n.2, p.7:1–7:25, 2016.

BOEIRA, F.; ASPLUND, M.; BARCELLOS, M. Decentralized proof of location in vehicular Ad Hoc networks. **Computer Communications**, London, UK, v.147, p.98 – 110, 2019.

BOEIRA, F.; ASPLUND, M.; BARCELLOS, M. P. **Vouch**: a secure proof-of-location scheme for vanets. In: ACM INTERNATIONAL CONFERENCE ON MODELING, ANALYSIS AND SIMULATION OF WIRELESS AND MOBILE SYSTEMS, 21., 2018, New York, NY, USA. **Proceedings...** ACM, 2018. p.241–248. (MSWIM '18).

BOEIRA, F. et al. Effects of colluding Sybil nodes in message falsification attacks for vehicular platooning. In: IEEE VEHICULAR NETWORKING CONFERENCE (VNC), 2017., 2017, Torino, IT. **Anais...** [S.l.: s.n.], 2017. p.53–60.

CHOWDHERY, A.; CHIANG, M. Model predictive compression for drone video analytics. In: IEEE INTERNATIONAL CONFERENCE ON SENSING, COMMUNICATION AND NETWORKING (SECON WORKSHOPS), 2018., 2018. **Anais...** [S.l.: s.n.], 2018. p.1–5.

DIFFIE, W.; HELLMAN, M. New directions in cryptography. **IEEE Transactions on Information Theory**, [S.l.], v.22, n.6, p.644–654, 1976.

DOSS, S. et al. **APD-JFAD**: accurate prevention and detection of jelly fish attack in manet. **IEEE Access**, [S.l.], v.6, p.56954–56965, 2018.

DOUCEUR, J. R. The Sybil attack. In: PEER-TO-PEER SYSTEMS, 2002, Berlin, Heidelberg. **Anais...** Springer Berlin Heidelberg, 2002. p.251–260.

ENGBERTS, B.; GILLISSEN, E. **Policing from above:** drone use by the police. In: THE FUTURE OF DRONE USE: OPPORTUNITIES AND THREATS FROM ETHICAL AND LEGAL PERSPECTIVES, 2016, The Hague. **Anais...** T.M.C. Asser Press, 2016. p.93–113.

FERREIRA, J.; PARDAL, M. L. Witness-based location proofs for mobile devices. In: IEEE 17TH INTERNATIONAL SYMPOSIUM ON NETWORK COMPUTING AND APPLICATIONS (NCA), 2018., 2018. **Anais...** [S.l.: s.n.], 2018. p.1–4.

FERRER, E. C. **The blockchain**: a new framework for robotic swarm systems. In: FUTURE TECHNOLOGIES CONFERENCE (FTC) 2018, 2019, Cham. **Proceedings...** Springer International Publishing, 2019. p.1037–1058.

FOTOUHI, A. et al. **Survey on UAV cellular communications**: practical aspects, standardization advancements, regulation, and security challenges. **IEEE Communications Surveys Tutorials**, [S.l.], p.1–1, 2019.

GARCIA-MAGARINO, I. et al. Security in networks of unmanned aerial vehicles for surveillance with an agent-based approach inspired by the principles of blockchain. **Ad Hoc Networks**, [S.l.], v.86, p.72 – 82, 2019.

GIRALDO, J. et al. **Security and privacy in cyber-physical systems:** a survey of surveys. **IEEE Design Test**, [S.l.], v.34, n.4, p.7–17, Aug 2017.

HUSODO, A. Y. et al. Intruder drone localization based on 2D image and area expansion principle for supporting military defence system. In: IEEE INTERNATIONAL CONFERENCE ON COMMUNICATION, NETWORKS AND SATELLITE (COMNETSAT), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.35–40.

ISLAM, N. et al. An expedite group key establishment protocol for flying ad-hoc network(FANET). In: INTERNATIONAL CONFERENCE ON INFORMATICS, ELECTRONICS AND VISION (ICIEV), 2016., 2016. **Anais...** [S.l.: s.n.], 2016. p.312–315.

JENSEN, I. J.; SELVARAJ, D. F.; RANGANATHAN, P. Blockchain technology for networked swarms of unmanned aerial vehicles (UAVs). In: IEEE 20TH INTERNATIONAL SYMPOSIUM ON "A WORLD OF WIRELESS, MOBILE AND MULTIMEDIA NETWORKS" (WOWMOM), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–7.

KAHN, D. **The codebreakers**: the story of secret writing [the comprehensive history of secret communication from ancient times to the internet. [S.l.]: Scribner, 1996.

LIN, C. et al. **Security and privacy for the internet of drones:** challenges and solutions. **IEEE Communications Magazine**, [S.l.], v.56, n.1, p.64–69, 2018.

LOUKAS, G. et al. A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles. **Ad Hoc Networks**, [S.l.], v.84, p.124 – 147, 2019.

MOSKWA, W. **World drone market seen nearing $127 billion in 2020, PwC says**. Available at: <https://www.moneyweb.co.za/news/tech/world-drone-market-seen-nearing-127bn-2020-pwc-says/>. Acesso em:
september 2019.

ORFANUS, D.; DE FREITAS, E. P.; ELIASSEN, F. Self-organization as a supporting paradigm for military UAV relay networks. **IEEE Communications Letters**, [S.l.], v.20, n.4, p.804–807, April 2016.

OUBBATI, O. S. et al. Routing in flying ad hoc networks: survey, constraints, and future challenge perspectives. **IEEE Access**, [S.l.], v.7, p.81057–81105, 2019.

PAUCAR, C. et al. Use of drones for surveillance and reconnaissance of military areas. In: DEVELOPMENTS AND ADVANCES IN DEFENSE AND SECURITY, 2018, Cham. **Anais...** Springer International Publishing, 2018. p.119–132.

RAJATHA, B. S.; ANANDA, C. M.; NAGARAJ, S. Authentication of MAV communication using caesar cipher cryptography. In: INTERNATIONAL CONFERENCE ON SMART TECHNOLOGIES AND MANAGEMENT FOR COMPUTING, COMMUNICATION, CONTROLS, ENERGY AND MATERIALS (ICSTM), 2015. **Anais...** [S.l.: s.n.], 2015. p.58–63.

REZA NOSOUHI, M. et al. **SPARSE:** privacy-aware and collusion resistant location proof generation and verification. In: IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM), 2018., 2018. **Anais...** [S.l.: s.n.], 2018. p.1–6.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. **Commun. ACM**, New York, NY, USA, v.21, n.2, p.120–126, Feb. 1978.

RODRIGUES, M. et al. Authentication methods for UAV communication. In: IEEE SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS (ISCC), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1210–1215.

SEHRAWAT, A.; CHOUDHURY, T. A.; RAJ, G. Surveillance drone for disaster management and military security. In: INTERNATIONAL CONFERENCE ON COMPUTING, COMMUNICATION AND AUTOMATION (ICCCA), 2017., 2017. **Anais...** [S.l.: s.n.], 2017. p.470–475.

SHAKERI, R. et al. **Design challenges of multi-UAV systems in cyber-physical applications:** a comprehensive survey and future directions. **IEEE Communications Surveys Tutorials**, [S.l.], v.21, n.4, p.3340–3385, Fourthquarter 2019.

SHAKHATREH, H. et al. **Unmanned aerial vehicles (UAVs)**: a survey on civil applications and key research challenges. **IEEE Access**, [S.l.], v.7, p.48572–48634, 2019.

SHARMA, S.; KAUL, A. A survey on intrusion detection systems and honeypot based proactive security mechanisms in VANETs and VANET Cloud. **Vehicular Communications**, [S.l.], v.12, p.138 – 164, 2018.

SRINIVAS, J. et al. **TCALAS:** temporal credential-based anonymous lightweight authentication scheme for internet of drones environment. **IEEE Transactions on Vehicular Technology**, [S.l.], v.68, n.7, p.6903–6916, July 2019.

STALLINGS, W. **Cryptography and network security**: principles and practice. [S.l.]: Pearson Education Limited, 2014.

TALBERT, R. The cycle structure and order of the rail fence cipher. **Cryptologia**, [S.l.], v.30, n.2, p.159–172, 2006.

TANEMBAUM, A. S.; WETHERALL, D. **Redes de computadores**. [S.l.]: Pearson Education Limited, 2014.

TAREQUE, M. H.; HOSSAIN, M. S.; ATIQUZZAMAN, M. On the routing in Flying Ad Hoc Networks. In: FEDERATED CONFERENCE ON COMPUTER SCIENCE AND INFORMATION SYSTEMS (FEDCSIS), 2015., 2015. **Anais...** [S.l.: s.n.], 2015. p.1–9.

WALIA, E.; BHATIA, V.; KAUR, G. Detection of malicious nodes in flying ad-hoc networks (FANET). **International Journal of Electronics and Communication Engineering**, Southampton, UK, v.5, p.6–12, 2018.

WANG, X. et al. **STAMP**: enabling privacy-preserving location proofs for mobile users. **IEEE/ACM Transactions on Networking**, [S.l.], v.24, n.6, p.3276–3289, December 2016.

ZACARIAS, I. et al. Employing SDN to control video streaming applications in military mobile networks. In: IEEE INTERNATIONAL SYMPOSIUM ON NETWORK COMPUTING AND APPLICATIONS, 16. (NCA), 2017. **Anais...** [S.l.: s.n.], 2017. p.1–4.

ZHU, Z.; CAO, G. **APPLAUS**: a privacy-preserving location proof updating system for location-based services. In: PROCEEDINGS IEEE INFOCOM, 2011., 2011. **Anais...** [S.l.: s.n.], 2011. p.1889–1897.

ZOU, Y. et al. **A survey on wireless security**: technical challenges, recent advances, and future trends. **Proceedings of the IEEE**, [S.l.], v.104, n.9, p.1727–1765, 2016.

APPENDIX A CONFUSION MATRIX PRESENTATION

A.1 Confusion matrix

The position validation mechanism, presented in this work as part of the UAVouch scheme, is a classification model. The mechanism was developed to separate legitimate from malicious drones, based on their movement pattern. Metrics such as accuracy, specificity, and sensitivity are widely used to measure the classification model performance. As presented in Chapter 5, all these metrics are computed based on output values (TP, TN, FN, and FP). The confusion matrix supports the performance measurements displaying these values in an organized way, as illustrated in Table 13.

Table 13 – Confusion matrix example

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	TP	FP
	legitimate	FN	TN

Following, the confusion matrix for each combination of the validation period and the plausibility check threshold will be presented. Tables 14 to 29 presents the confusion matrices for scenario 1 and Tables 30 to 45 presents the confusion matrices for scenario 2.

Table 14 – Confusion Matrix (1σ and 0.1 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1682	4
	legitimate	20	2000

Table 15 – Confusion Matrix (1σ and 0.2 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1662	7
	legitimate	40	1998

Table 16 – Confusion Matrix (1σ and 0.5 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1611	4
	legitimate	89	2000

Table 17 – Confusion Matrix (1σ and 1.0 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1421	2
	legitimate	281	2003

Table 18 – Confusion Matrix (2σ and 0.1 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1633	2
	legitimate	71	1999

Table 19 – Confusion Matrix (2σ and 0.2 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1619	1
	legitimate	84	2003

Table 20 – Confusion Matrix (2σ and 0.5 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1495	1
	legitimate	206	2000

Table 21 – Confusion Matrix (2σ and 1.0 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1353	1
	legitimate	352	2000

Table 22 – Confusion Matrix (3σ and 0.1 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1616	3
	legitimate	86	1998

Table 23 – Confusion Matrix (3σ and 0.2 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1612	1
	legitimate	92	2000

Table 24 – Confusion Matrix (3σ and 0.5 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1464	2
	legitimate	239	2002

Table 25 – Confusion Matrix (3σ and 1.0 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1289	1
	legitimate	413	1999

Table 26 – Confusion Matrix (4σ and 0.1 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1579	1
	legitimate	122	2000

Table 27 – Confusion Matrix (4σ and 0.2 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1548	1
	legitimate	156	2000

Table 28 – Confusion Matrix (4σ and 0.5 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1450	1
	legitimate	254	1999

Table 29 – Confusion Matrix (4σ and 1.0 [s]) from scenario 1

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1282	1
	legitimate	419	1999

Table 30 – Confusion Matrix (1σ and 0.1 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1953	5
	legitimate	49	1996

Table 31 – Confusion Matrix (1σ and 0.2 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1955	8
	legitimate	50	1995

Table 32 – Confusion Matrix (1σ and 0.5 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1951	3
	legitimate	50	1997

Table 33 – Confusion Matrix (1σ and 1.0 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1956	3
	legitimate	48	1997

Table 34 – Confusion Matrix (2σ and 0.1 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1901	2
	legitimate	99	2000

Table 35 – Confusion Matrix (2σ and 0.2 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1904	1
	legitimate	99	2000

Table 36 – Confusion Matrix (2σ and 0.5 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1903	1
	legitimate	97	1999

Table 37 – Confusion Matrix (2σ and 1.0 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1905	1
	legitimate	97	2003

Table 38 – Confusion Matrix (3σ and 0.1 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1855	2
	legitimate	149	1998

Table 39 – Confusion Matrix (3σ and 0.2 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1853	1
	legitimate	147	2001

Table 40 – Confusion Matrix (3σ and 0.5 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1856	1
	legitimate	149	2002

Table 41 – Confusion Matrix (3σ and 1.0 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1853	1
	legitimate	147	2000

Table 42 – Confusion Matrix (4σ and 0.1 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1807	2
	legitimate	198	2003

Table 43 – Confusion Matrix (4σ and 0.2 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1805	1
	legitimate	197	2002

Table 44 – Confusion Matrix (4σ and 0.5 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1805	1
	legitimate	196	2003

Table 45 – Confusion Matrix (4σ and 1.0 [s]) from scenario 2

		Actual Values	
		attacker	legitimate
Predicted Values	attacker	1803	1
	legitimate	197	1999