SecAuthUAV: a UAV-GS and UAV-UAV Authentication Protocol

Voltan Gabriele

July 14, 2023

Abstract

Nowadays the use of Unmanned Aerial Vehicles (UAVs) is constantly growing, thanks to the many applications they find both in the civil and military fields. However, despite the multiple benefits they bring in many applications, issues such as ensuring the security of communications between UAV and ground station or such as physical capture and tampering cannot be overlooked. Taking into account the limited computational capabilities, limited memory and limited energy autonomy, a lightweight mutual authentication protocol between UAV and GS or between two different UAVs and based on Physical Unclonable Functions (PUFs), is presented. The paper, after introducing the results of the protocol security analysis made with Mao Boyd logic, reports a comparison of the security aspects and performance between SecAuthUAV and other protocols proposed in the literature, thus highlighting its excellent characteristics.

Contents

1 Introduction

Unmanned Aerial Vehicles (UAVs), also known more commonly as *drones*, are aircraft, of various sizes, capable of flying without the presence of a human on board. Invented for military purposes, as early as 1849 - in a naïve form -, today they find many uses in the civilian sphere as well [5]. The most common civilian uses are photography and environmental monitoring or measurements. Instead, among the most innovative and recent uses we find that of transporting materials or the experimental one of the supply of connections, by Internet Service Providers, in remote places, through the use of solar-powered drones [11]. The most interesting uses, in my opinion, are the military ones. In fact, among the most frequent uses we find the reconnaissance and surveillance of certain geographical areas, the targeting of objectives, the activities of signal intelligence (SIGINT) and the attack against objectives.

Within NATO, drones are divided into three classes, based on their weight: class I is made up of drones weighing less than 150 kg (for example DJI commercial drones); class II consists of drones weighing between 150 kg and 600 kg; class III consists of drones weighing more than 600 kg (for example the Predator, or the Global Hawk, a drone widely used in the Russian-Ukrainian conflict) [7]. Regardless of their mass, the impact of a UAV on a mass of people, or on another aircraft in flight, would create non-negligible problems. Precisely for this reason, the *Internet of Drones* (IoD) falls not only in the field of *security*, i.e. the protection of computer systems in order to guarantee the CIAAA, but also in the field of *safety*, i.e. the protection of computer systems in order to prevent accidents that can cause physical damage to people or the environment.

The success of drones is due to the simplicity and, very often, the speed with which certain points and altitudes can be reached, without employing humans on board, but using only the ether as a communication and control channel. However, if on the one hand this is a great advantage, on the other it is a problem as it introduces a series of non-negligible and not easy to prevent security threats. A non-exhaustive list of these threats is as follows: replay attacks, masquerade attacks, man-in-the-middle attacks, node capture and tampering, etc (detailed in the section 2). For example, in a civilian concert surveillance scenario, if a malicious ground station tries to impersonate the legitimate ground station by taking control of the UAV, it could purposely crash it into the crowd, causing extensive damage.

From the example just reported, we can immediately understand the importance of node authentication within the IoD. The techniques commonly used in other computing environments are based on the cryptographic keys stored in the device. However, for the IoD sector, but also for the IoT one, this technique is not the best possible, mainly for two reasons: the first is the fact that these types of devices are subject to capture and tampering; the second is that they have limited memory and computational capabilities [2].

To solve the node authentication problem, respecting the capabilities and uses of a drone, this paper presents a protocol, called *SecAuthUAV*, introduced in the literature in [2]. This protocol, to avoid the use of cryptographic keys,

uses *Physical Unclonable Functions* (PUFs), i.e. a hardware challenge-response mechanism [3]. A more detailed discussion of PUFs will follow later.

The rest of the paper is structured as follows: the system model and the threat model are presented in the section 2; the security objectives are presented in the section 3; the operation of all phases of the protocol is explained in the section 4; the results of the formal security verification of the protocol, carried out with Mao Boyd logic, are reported in the section 5; in section 6 possible attacks or vulnerabilities are discussed; in section 7 a comparison is made with other authentication protocols for UAVs proposed in the literature; finally, the section 8 concludes by discussing the effectiveness of the proposed solution.

2 System and Threat model

In this section we analyze the system model and the threat model, in order to accurately delineate who are the legitimate parties, what are the assets to protect, who are the possible attackers and what attack vectors can they use.

2.1 System model

The model of the system consists of a series of UAVs and a GS ground station, whose common goal is to authenticate each other. Once a mutual authentication between UAV and GS has been achieved, it will be possible to perform a mutual authentication between two UAVs, using the GS as *trusted third party*.

Each UAV has a PUF, which is used to generate a response to a challenge received as input. In formulas, we can thus describe this mechanism

$$R = PUF(C)$$

where C is the challenge and R is the response. This mechanism has been shown to be very promising for the IoT and IoD domain, as it possesses a number of properties that make it suitable for the context.

The general idea behind PUFs is the uniqueness of their physical microstructure, due to random factors introduced during the manufacturing process. In fact, by definition they are impossible to clone both physically and mathematically. As well described in [6], There are two types of PUFs: *weak PUFs* and *strong PUFs*. The main difference between the two is the size of the challenge-response space that the function has. The size of the space is linearly related to the number of components whose behavior depends on the manufactoring variation.

2.2 Threat model

Assets The assets to be protected in this model are the various UAVs and the ground station. Furthermore, even if they are not a digital asset, humans and the flight environment must also be considered as more general assets, since, as explained in the introductory chapter, the consequences following an attack on a UAV could be catastrophic.

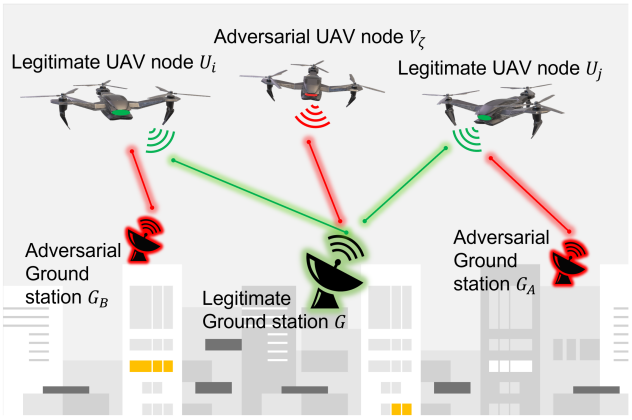

Figure 1: Threat model

Attackers As can be seen from the fig. 3, the attackers in this model can be multiple: malicious UAVs that would like to communicate with the GS, for example, to communicate false data to the GS (typical in a military targeting or reconnaissance scenario); malicious GSs attempting to interrupt communication with the legitimate GS, in order to interrupt the flight of the UAV, or to try to take control of it; human persons, with the aim of capturing the UAV, in order to extract secret information, tamper with it or clone it.

Attack vectors As attack vectors for this threat model we find: the masquerade attack, or an attack in which an adversary tries to authenticate himself as a legitimate user (attack possible for both a GS and a UAV); the man-in-the-middle attack (MIMT); eavesdrop on exchanged messages in order to alter or re-propose them (replay attack); physical capture of the UAV in order to extract secret information or clone it.

Being that analyzed an authentication protocol, these are the possible attacks against this process. However, in [17] we can find a fairly detailed analysis of other possible attacks on UAVs.

2.3 Assumptions

In this subsection some assumptions made in the paper [2], for the proposed protocol, are reported and analysed.

1. All UAVs have limited computational capacity and memory, while GSs have no limits.

2. No UAV stores secret information. The secret used in the protocol is the response R generated by the PUF, after receiving the challenge C.

3. Every legitimate UAV has a PUF, as described in the system model. In case of capture, any attempt to tamper with the PUF will make it unusable, and therefore the UAV will no longer be able to authenticate.

The first and second assumptions are reasonable and achievable. Instead, the third assumption is not trivial and deserves an in-depth analysis which will be proposed later, in the section 6.

3 Security goals

This section lists the objectives that the SecAuthUAV protocol aims to achieve. Subsequently, after presenting the protocol in the section 4, it will be formally analyzed in the section 5 in order to determine whether the objectives have been achieved or not.

The goals that the authors, in [2], have set for the SecAuthUAV protocol are:

1. Achieve mutual authentication between legitimate UAVs and GS ground station.
2. The protocol must resist masquerade, main-in-the-middle, and replay attacks.
3. The protocol must resist cloning and physical attacks, such as capture of the UAV and tampering with it.
4. The protocol must generate a unique session key for each authentication session.
5. If messages are tampered with, the receiving node must notice and abort the authentication process.
6. No unauthorized authority should be able to trace the temporary ID of the UAV, in order to guarantee the anonymity of the UAV.

As we can see, the third goal is strongly related to the third assumption made in the section 2.3 and, in general, to the PUF's resistance to cloning. As previously said, an analysis will be made on the various known attacks on PUFs, which could make the mentioned assumption unfeasible.

4 Security service and implementation

The proposed protocol aims to achieve mutual authentication, between UAV and GS and, subsequently, between two UAVs, in a secure manner and without using a large amount of computational resources. As we can see from the description of the protocol, the most complex operation that will be performed will be the hash calculation.

The protocol is divided into three phases: the UAV registration phase; the authentication phase between the UAV and the GS; the authentication phase between UAV and UAV. These three phases will now be described in detail.

4.1 UAV Registration Phase

As in many authentication protocols, a node must first be "enrolled" in order to authenticate itself. The registration phase is almost always performed through a *secure channel*.

UAV registration takes place in three simple steps:

1. The ground station generates a challenge C and a temporary ID for the UAV (TUID) and sends everything to the UAV.

2. The UAV, after having received the challenge C, calculates the answer $R = PUF(C)$ and stores in its memory the triple {TUID, C, GID}. Subsequently send the response R to the GS.

3. The GS, after receiving the response R, stores the triple {TUID, C, R} in its database.

As we can see, the UAV does not store any secret information within its memory. Saving the TUID is not problematic, as being temporary, it has a limited validity equal to the duration of the flight.

4.2 UAV-GS Authentication Phase

This phase is the main phase of the protocol, which will then allow the third phase to be carried out, namely that of mutual authentication between two UAVs.

In order to obtain mutual authentication between UAV and GS and to establish a session key for future communications, the protocol performs the following operations:

1. When the UAV wants to start authentication, it calculates the R for the challenge C it has stored and sends the following message

$$UAV \rightarrow GS : TUID, N_A, H(R||TUID||N_A)$$

2. After receiving the first message, the GS queries its database in order to verify the presence of the TUID: if it is not found, it aborts the session.

   If instead it is found, the GS checks the hash and then performs the following calculations:

$$X_1 = N_A \oplus K_2$$

$$Y_2 = N_B \oplus X_1 \oplus K_1$$

$$Q = (Y_2||X_1) \oplus (K_2||K_1)$$

where N_b is a fresh nonce and K_1 and K_2 are the split R.

After performing these calculations, the GS sends the following message to the UAV:

$$GS \rightarrow UAV : Q, H(Q||GID||N_A||N_B)$$

3. The UAV splits the R response into K_1 and K_2, as done by the GS, to carry out the operations reported below.

$$(Y_2||Y_1) = (K_2||K_1) \oplus Q$$
$$N_B = Y_2 \oplus X_1 \oplus K_1$$
$$N_A = X_1 \oplus K_2$$

By calculating N_A and N_B, the UAV calculates and verifies the hash, thus the message source, freshness and integrity are verified. If the hash check fails, the UAV will abort the session.

At this point, the UAV generates a fresh nonce N_C, a subpart of which will serve as a new challenge C', based on the challenge size requested by the PUF. After obtaining C', the UAV calculates (C', R'). This information will be encoded within M' and N' as follows.

$$M' = R' \oplus K_2||K_1$$
$$N' = N_C \oplus K_1$$

The Sk session key used for future communications will be calculated as follows.

$$Sk = (K_1 \oplus N_B)||(K_2 \oplus N_C)$$

After calculating all this, the UAV sends the third and last message of the protocol to the GS.

$$UAV \rightarrow GS : M', N', H(R'||TUID||N_B||N_C||Sk)$$

4. Once the third message has been received, the GS will carry out the last calculations and final checks.

$$N_C = N' \oplus K_1$$
$$R' = M' \oplus K_2||K_1$$
$$Sk = (K_1 \oplus N_B)||(K_2 \oplus N_C)$$

At this point the GS has all the parameters to verify the hash. If the verification fails, the GS terminates the session. Otherwise, save the new challenge-response pair (C', R') in the database. Also, both the UAV and the GS calculate the new temporary ID of the UAV as follows.

$$TUID' = H(K_2||TUID||K_1)$$

Below is a summary image of this phase of the protocol.

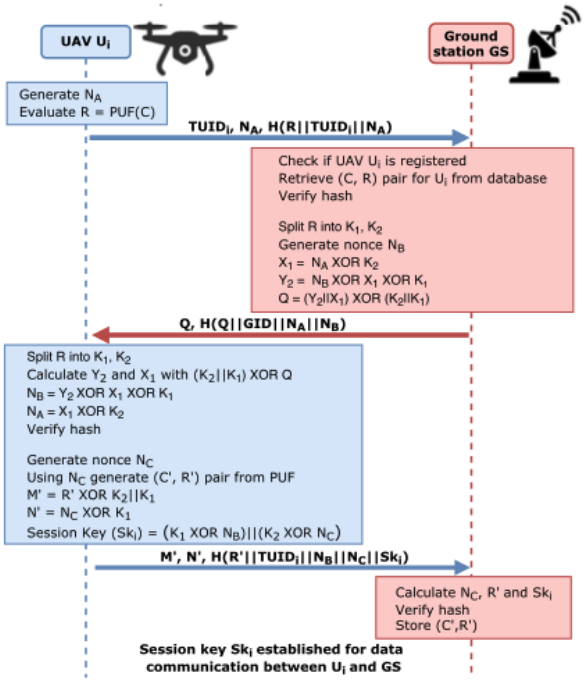

Figure 2: UAV-GS authentication phase

4.3 UAV-UAV Authentication Phase

The UAV-UAV Authentication Phase is a phase in which, by exploiting the phase described in the previous section, two UAVs can authenticate themselves, using the GS as *trusted third party*. The steps in this phase are:

1. The UAV_1 after having established a session key Sk_1 with the GS, requests the connection with UAV_2 to the GS.

$$UAV_1 \rightarrow GS : E_{Sk_1}(Req\_conn\_U_2)$$

2. After finding an available UAV_2, the GS requests authentication.

$$GS \rightarrow UAV_2 : Req, H(Req||TUID_2||GID)$$

3. In case the authentication between the GS and the UAV_2 is successful, the GS would now share a symmetric key with both UAVs. By exploiting these two symmetric keys, it can generate a third one for communications between the two UAVs and send them securely.

$$GS \rightarrow UAV_1 : E_{Sk_1}(Sk_{12})$$

$$GS \rightarrow UAV_2 : E_{Sk_2}(Sk_{12})$$

Below is a summary image of this phase of the protocol.

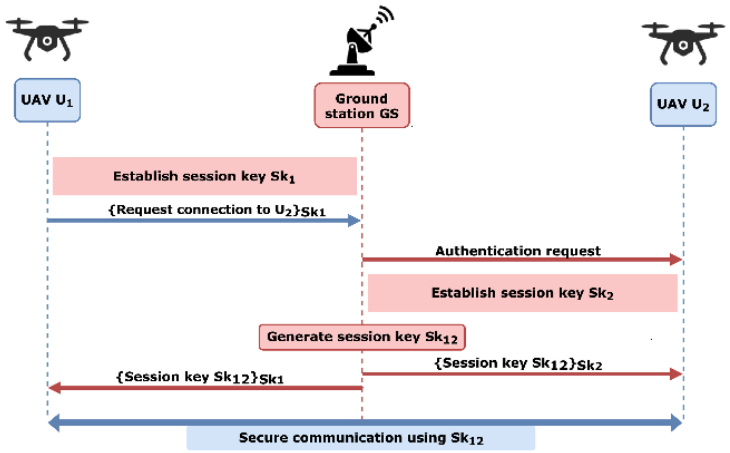

Figure 3: UAV-GS authentication phase

Notation	Meaning
$K \models arg$	U believes arg
$K_1 \overset{arg}{\longleftrightarrow} K_2$	arg good secret between K_1 and K_2
$K \overset{key}{\triangleleft} arg$	K sees arg with decipher key key
$\#arg$	arg is fresh
$sup(K)$	K is super-principal
$\{K_1, ..., K_n\}^{\triangleleft} \parallel arg$	No one other than $K_1, ..., K_n$ has access to arg

Table 1: Meaning of symbols used in the security verification

5 Security analysis

In this section we formally analyze the security of the proposed protocol.

In [2], the authors verified the protocol by invoking the *confidentiality, authentication, nonce-verification, super-principal, intuitive and good-key inference* rules of the Mao Boyd logic [10]. The verification carried out shows how to verify, invoking the aforementioned rules of Mao Boyd logic, the following statements: "U believes that N_B is a good secret between U and G" and "G believes that N_B is a good secret between U and G". A representation of the proof is given by fig. 4 and fig. 5, respectively for the first and second statement mentioned. To better understand the images shown, a brief section 5 is provided which explains the meaning of the notations used.

Similarly, by replicating the process for N_C and R', we ensure that attacks such as MIMT, replay or masquerade cannot be conducted.

Furthermore, they conducted an analysis based on a list of security criteria introduced in [16]. The results of this analysis demonstrated that the protocol

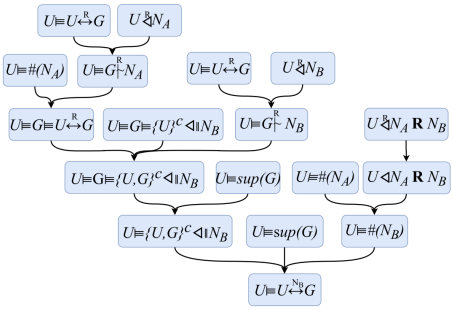

Figure 4: Statement 1

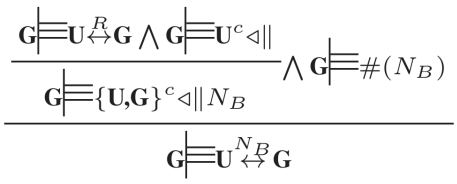

Figure 5: Statement 2

has the following characteristics:

- (Crit. 5) *Resistance to known attacks*: masquerade attack, man-in-the-middle attack, replay attack, node tampering attack, cloning attack and de-synchronization attack.
- (Crit. 7) *Provision of key agreement*: after each successful authentication a fresh and secure session key is established.
- (Crit. 8) *No clock synchronization*: nonces are used instead of timestamps to ensure message freshness.
- (Crit. 10) *Mutual authentication*: thanks to the (C,R) pair stored in the GS during the registration phase, it is possible to guarantee mutual authentication.
- (Crit. 11) *User anonymity*: in the protocol the IDs of the UAV are temporary and at the end of the process it is recalculated.
- (Crit. 12) *Forward secrecy*: even if an attacker A discovers the Sk session key, this does not compromise the security of future sessions.

In the future, with the publication of a subsequent paper, the formal verification of the security of the protocol, implemented in ProVerif [4], will be presented. The aforementioned tool is widely used in the literature to conduct these types of analyzes and for this reason it was decided to enrich this analysis work with this work.

6 Attacks and Vulnerabilities

With the assumptions made by the authors on the PUF, the protocol, as demonstrated by the security analysis, does not report any vulnerabilities. However, some vulnerabilities regarding PUFs have been discovered in the literature. As for invasive attacks, i.e. attacks that aim to compromise the PUF, the chances

of success are quite low for the attackers, as it is difficult to carry them out without modifying the physical characteristics from which the secret is derived [6]. Even simple measurements to detect gate delay can affect the properties of the gate.

However, if on the one hand the invasive solution turns out to be practically inefficient, that of cloning has shown more promising results. XOR arbiter PUFs, i.e. multiple PUFs whose result is XOR'ed, have been shown to be vulnerable to attacks that leverage the combination of machine learning and side-channels [8][14]. Developing methods to remove side channels could reduce exposure to this vulnerability, but as is well known, doing so is not trivial.

Other studies [9] [13] have shown the efficiency of machine learning algorithms which, after observing a given number of challenge-response pairs (C,R), are able to predict the behavior of PUFs. For a technical and exhaustive explanation of how PUFs and attacks work, we recommend reading [6].

6.1 Vulnerability considerations

If we only consider the known vulnerabilities regarding PUFs, we could conclude that the protocol presented is not secure, as an attacker, after cloning the PUFs and stealing the stored challenge C, could have great advantages.

However, considering the IoD world we can come to the conclusion that these vulnerabilities are not that serious. Reminding us that each UAV has its own unique PUF and starting from the fact that these attacks require the physical capture of the drone, let us consider the feasibility of these attacks in both the civilian and military sectors.

Let's consider the first scenario, where the UAV is captured during a flight phase (it doesn't matter how): the owner of the UAV, be it civilian, or even better military, will consider that drone as lost/captured/shot down. Consequently, even if the attacker cloned the PUF, the attack would be useless as the rightful owner would no longer have it.

Let us instead consider the opposite scenario, in which physical access to the UAV takes place in a phase in which the UAV is not in flight: this is a scenario that could only occur in a civilian context (albeit in a difficult way), as the UAVs soldiers are assumed to be parked inside a military base, which is protected and guarded.

We can therefore conclude that these attacks, for the IoD area, are not so problematic, or in any case, certainly more complex than a trivial theft of an encryption key stored in the drone. Instead, for the IoT environment, in which the nodes are very often not dynamic, but rather are left fixed in one point, often not even supervised or frequented (imagine for example a temperature sensor on the top of a mountain), these attacks could be problematic.

7 Comparison with other solutions

Given the rapid expansion of the IoD world, several solutions to the authentication problem have been proposed in the literature. In order to determine which solution is better than the others, it is necessary to compare the security offered and the performance of the protocols. In [2] the authors have chosen to make the comparison with four protocols proposed in [18], [15], [1], [12].

Security comparison For the comparison regarding the security implemented, the criteria reported in the section 5 are used. SecAuthUAV, as reported, implements all of those policies. [18], [15], [1] do not implement the defense against tampering and cloning and use synchronized clocks, while [12] does not provide the anonymity of UAVs.

Performance comparison For the performance comparison a NodeMCU v3.0 and a Raspberry Pi 3B were used as simulation environments for the UAV. The most common cryptographic operations (XOR, PRNG, hash, HMAC, concatenations) were performed on both environments, implementing them both in C and in Python. The one proposed in [19] was used as PUF for SecAuthUAV and for [12].

The first comparison, performed by implementing the protocols in C on both simulation environments, shows that SecAuthUAV has a lower computational cost than all the other protocols, in both simulation environments.

The second comparison, made by implementing the C and Python protocols on Raspberry Pi 3B, shows again that SecAuthUAV is less expensive than the other proposals.

The third comparison, made by comparing the communication costs, brings SecAuthUAV to second place, after [15], for only 64 bits more, on an average cost of 1696 bits.

Finally, the fourth comparison, made by comparing the minimum cost of storage, brings SecAuthUAV to second place, after [12], for only 32 more bits, on an average cost of 486 bits.

8 Conclusions

Given the increase in the use of UAVs in both the civil and military context, the need for new security protocols, increasingly complete and efficient, is very high.

This paper presents SecAuthUAV [2], a lightweight mutual authentication protocol that allows to authenticate both the UAV and the ground station, as well as two different UAVs, using *Physically Unclonable Functions* (PUFs).

Some vulnerabilities of the PUFs, known in the literature, are analyzed, also making some considerations about them in the IoD field. Considering the fact that they allow to avoid storing secret keys inside the UAV and considering the

difficulty in applying the known vulnerabilities to the IoD environment, PUFs have been evaluated as an excellent solution for this application.

Furthermore, thanks to the formal verification proposed in [2], it has been demonstrated that SecAuthUAV provides mutual authentication, forward secrecy, resistance to known attacks (such as MIMT, replay, masquerade, de- synchronization attack, physical attacks) and the anonymity of the UAV, without the use of synchronized clocks.

Finally, comparing the proposed security features and performances with those of four other protocols present in the literature, it has been shown that SecAuthUAV is a more efficient and complete solution compared to the other solutions already proposed.

In the future, formal security verification of the protocol using ProVerif will be proposed.

References

[1] Zeeshan Ali, Shehzad Ashraf Chaudhry, Muhammad Sher Ramzan, and Fadi Al-Turjman. Securing smart city surveillance: A lightweight authentication mechanism for unmanned vehicles. *IEEE Access*, 8:43711–43724, 2020.

[2] Tejasvi Alladi, Gaurang Bansal, Vinay Chamola, Mohsen Guizani, et al. Secauthuav: A novel authentication scheme for uav-ground station and uav-uav communication. *IEEE Transactions on Vehicular Technology*, 69(12):15068–15077, 2020.

[3] Gaurang Bansal, Naren Naren, Vinay Chamola, Biplab Sikdar, Neeraj Kumar, and Mohsen Guizani. Lightweight mutual authentication protocol for v2g using physical unclonable function. *IEEE Transactions on Vehicular Technology*, 69(7):7234–7246, 2020.

[4] Bruno Blanchet, Ben Smyth, Vincent Cheval, and Marc Sylvestre. Proverif 2.00: automatic cryptographic protocol verifier, user manual and tutorial. *Version from*, pages 05–16, 2018.

[5] Combodrone. Storia dei droni. dal 1849 ai giorni nostri.

[6] Charles Herder, Meng-Day Yu, Farinaz Koushanfar, and Srinivas Devadas. Physical unclonable functions and applications: A tutorial. *Proceedings of the IEEE*, 102(8):1126–1141, 2014.

[7] Joint Air Power Competence Centre. *Strategic concept of employment for unmanned aircraft systems in NATO*, January 2010.

[8] Ahmed Mahmoud, Ulrich Rührmair, Mehrdad Majzoobi, and Farinaz Koushanfar. Combined modeling and side channel attacks on strong pufs. *Cryptology ePrint Archive*, 2013.

[9] Mehrdad Majzoobi, Farinaz Koushanfar, and Miodrag Potkonjak. Testing techniques for hardware security. *2008 IEEE International Test Conference*, pages 1–10, 2008.

[10] Wenbo Mao and Colin Boyd. Towards formal analysis of security protocols. *Proceedings Computer Security Foundations Workshop VI*, pages 147–158, 1993.

[11] GN Muchiri and S Kimathi. A review of applications and potential applications of uav. In *Proceedings of the Sustainable Research and Innovation Conference*, pages 280–283, 2022.

[12] Cong Pu and Yucheng Li. Lightweight authentication protocol for unmanned aerial vehicles using physical unclonable function and chaotic system. *2020 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, pages 1–6, 2020.

[13] Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. Modeling attacks on physical unclonable functions. *Proceedings of the 17th ACM conference on Computer and communications security*, pages 237–249, 2010.

[14] Ulrich Rührmair, Xiaolin Xu, Jan Sölter, Ahmed Mahmoud, Farinaz Koushanfar, and Wayne Burleson. Power and timing side channels for pufs and their efficient exploitation. *Cryptology ePrint Archive*, 2013.

[15] Jangirala Srinivas, Ashok Kumar Das, Neeraj Kumar, and Joel JPC Rodrigues. Tcalas: Temporal credential-based anonymous lightweight authentication scheme for internet of drones environment. *IEEE Transactions on Vehicular Technology*, 68(7), 2019.

[16] Ding Wang and Ping Wang. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE transactions on dependable and secure computing*, 15(4):708–722, 2016.

[17] Li Wang, Yu Chen, Pu Wang, and Zheng Yan. Security threats and countermeasures of unmanned aerial vehicle communications. *IEEE Communications Standards Magazine*, 5(4):41–47, 2021.

[18] Mohammad Wazid, Ashok Kumar Das, Neeraj Kumar, Athanasios V Vasilakos, and Joel JPC Rodrigues. Design and analysis of secure lightweight remote user authentication and key agreement scheme in internet of drones deployment. *IEEE Internet of Things Journal*, 6(2):3572–3584, 2018.

[19] Xiaojin Zhao, Qiang Zhao, Yongpan Liu, and Feng Zhang. An ultracompact switching-voltage-based fully reconfigurable rram puf with low native instability. *IEEE Transactions on Electron Devices*, 67(7):3010–3013, 2020.