# Key Establishment for Maintenance with Machine to Machine Communication in Transportation: Security Process and Mitigation Measures

Sibylle Fröschle
*Institute for Secure Cyber-Physical Systems*
*Hamburg University of Technology*
Hamburg, Germany
sibylle.froeschle(at)tuhh.de

Martin Kubisch
*Airbus CRT*
Munich, Germany
martin.kubisch(at)airbus.com

***Abstract*—Machine to machine communication over wireless networks is increasingly adopted to improve service and maintenance processes in transportation, e.g., at airports, ports, and automotive service stations. This brings with it the challenge of how to set up a session key so that the communication can be cryptographically secured. While there is a vast design space of key establishment methods available, there is a lack of process of how to engineer a solution while considering both security and safety: how to assess the threats and risks that come with a particular key establishment method? And how to iteratively refine a key establishment method under development such that risk is mitigated to an acceptable level? In this paper, we put forward an approach that addresses these questions. Moreover, we devise several cyber-physical measures that can be added to mitigate risk. We illustrate our approach and the mitigation measures by means of a real-world use case: TAGA — a Touch and Go Assistant in the Aerospace Domain. Finally, we highlight the crucial role that simulation has to play in this security process for safety.**

***Index Terms*— *Security; Key Establishment; Threat and Risk Analysis; Simulation; Transportation.***

## I. Introduction

Machine to Machine (M2M) communication over wireless networks is increasingly adopted to improve service and maintenance processes in transportation, e.g., at airports, ports, and automotive service stations. This does not come without security challenges: often these processes are safety-critical, and often, attacks against them would disrupt critical infrastructures. One example are the ground processes at an airport. When an aircraft has landed and reached its parking slot at the apron many processes such as refuelling and pre-conditioning are performed. M2M communication between the aircraft and the respective ground unit allow us to optimize these processes with respect to accuracy of service, energy-efficiency, safety, and time. The aircraft will send sensor values (e.g., temperature or fuel readings), and the ground unit can adopt flow parameters accordingly. However, if an attacker managed to spoof fake sensor values into the M2M communication then this could compromise safety.

The adoption of M2M communication brings with it the challenge of how to set up a session key so that the communication can be cryptographically secured. While there is a vast design space of key establishment methods available, there is a lack of process of how to engineer a solution while considering both security and safety: how to assess the threats and risks that come with a particular key establishment method? And how to iteratively refine a key establishment method under development such that risk is mitigated to an acceptable level? In this paper we address these questions building on our conference contribution [1]. We motivate and illustrate our approach by means of a real-world use case: TAGA — a Touch and Go Assistant in the Aerospace Domain. TAGA is currently under development to enable the introduction of M2M communication for ground processes at airports.

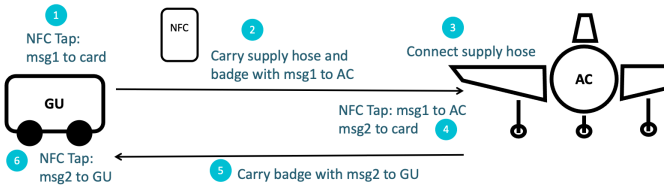


Fig. 1. Pairing up a ground unit and an airplane

The idea behind TAGA is to set up the session key by means of a Near Field Communication (NFC) system. Each aircraft and ground unit is equipped with a TAGA controller that contains a secure element for cryptographic operations and an NFC reader. Moreover, the operator of each ground unit is provided with a passive NFC card. Altogether, this allows them to transport messages for key establishment from the ground unit to the aircraft, and back by means of taps with the NFC card against the respective NFC reader. The 'TAGA walk' can conveniently be integrated into the operator's usual path to the aircraft and back while connecting up the respective supply hose. This is illustrated in Figure 1.

To start off with, TAGA only defines a process of how to transfer the messages of a two-way key establishment (KE) protocol, and there is still considerable design space: which

concrete KE protocol shall we employ? Do we need to add further measures to mitigate risk, and if so which ones?

Integrating M2M communication in transportation has to undergo a safety and security engineering process conform to the safety and security norms applicable to the respective domain (such as ISO/SAE 21434 for road vehicles and DO-178C, DO-254, DO-326A and ARP4754 in the aeronautics domain). This process will typically involve the following activities. First, vulnerable assets have to be identified (such as here the communication channel). Second, for each asset the potential threats have to be collected (e.g., by a keyword-guided method such as STRIDE). And third, for each threat a risk level has to be determined. The risk level is typically determined by, on the one hand, rating the safety impact of the threat, and, on the other hand, rating the likelihood that the threat can be implemented. As a result, the risk level will decide whether protection by security controls is required, and to which assurance level the corresponding security requirements have to be validated.

When it comes to integrating security controls and security systems the most relevant and widely adopted standard is Common Criteria (CC) (ISO/IEC 15408). It is the standard that is widely adopted to evaluate security products and systems. This standard allows us to define a profile of security requirements for a target of evaluation that fall into security functional requirements, and assurance requirements. The latter specify that the security functional requiremens must be validated to a sufficient assurance level. While a CC profile provides a clear interface between safety and security this should not be taken as an excuse to stop short of a stronger integration between security and safety engineering. Without it important safety measures that can mitigate security risks might be overlooked.

*Problem and Contribution:* While there is a vast design space of key establishment methods and products available, some of them with CC evaluation, there is a lack of process of how to engineer a solution while integrating both security and safety: how to assess the threats and risks that come with a particular key establishment method in a specific context? And how to iteratively refine a key establishment method under development such that risk is mitigated to an acceptable level? In this paper, we put forward and illustrate an approach that addresses these questions.

We proceed as follows. In Section II we motivate and present our overall approach. Our approach is based on the concept of *connection compromise states*, which define how key establishment can fail, and provide a finer-grained interface between security and safety. In Section III we motivate and illustrate our approach by means of the TAGA use case. In Section IV we devise several concrete measures that can be added to mitigate risk. In Section V we give a workflow on how to assess and mitigate the safety impact starting from the connection compromise states. In particular, we highlight the important role of simulation in this workflow. In Section VI we put our work in context with related work. In Section VII we draw conclusions and discuss future work.

This paper extends the conference version [1] as follows. In Section II we additionally provide the rationale behind the connection compromise states. In Section III-D we discuss an intricate consequence of long-term key compromises, and in Section IV we introduce several new mitigation measures. Some of the material is based on the preprint [2].

## II. Key Establishment for Vehicle to Service Unit Communication

*Setting:* We first define the problem setting. As shown by example in Figure 1 we assume that there is a vehicle $V$ that is to undergo a maintenance procedure at some location. The maintenance procedure can involve several types of services, and each service involves at least one service unit. Each service unit is either directly coupled to the vehicle (e.g., via a supply hose) or indirectly (e.g., via the loading of goods). To optimize the maintenance procedure each service unit shall be able to engage in M2M communication with the vehicle it services: to exchange data such as sensor and status values or even instructions on how to move. Several such procedures can take place in parallel in adjacent or remote locations.

We assume that the communication is conducted over a wireless channel (such as Wi-Fi IEEE 802.11), and that a protocol that allows two parties to communicate securely, given a secure session key is already in place. This involves an AEAD (Authenticated Encryption with Associated Data) scheme such as AES-GCM, and measures against replay and reflection such as counters and directionality differentiation (c.f. [3], Section 5.4). For the Wi-Fi security protocols WPA2/3 this is provided by the subprotocol that is responsible for the bulk data handling after the 4-way handshake. Here we focus on the challenge of how to establish the necessary session key between a service unit and the vehicle.

*Security Requirements:* Table I shows the security properties that any key establishment method for Vehicle to Service Unit (V2SU) communication must at least satisfy. Properties (1) and (2) ensure that the key remains secret, and that it is fresh for each session. Properties (3) and (4) are derived from the standard authentication properties for key establishment protocols [4]. We have formulated the properties without explicitly referring to the names of the peers. This is to allow for *secure device pairing* as the key establishment method of choice, where identities do not necessarily have to be exchanged. Names can, however, be included in the parameter list. One can also include the type of service, and other service specific parameters into the parameter list. Property (5) is specific to our setting: it ensures that the cyber channel indeed connects the machines that are physically coupled in the maintenance service.

*Design Space:* The state of the art of key establishment offers two approaches to achieve the secrecy and authentication properties: one is to employ an *Authenticated Key Establishment (AKE) Protocol* [5]; the second is to make use of a *Secure Device Pairing (SDP) scheme* [6]. As we will see later a combination is also possible.

AKE protocols [5] are by now well-investigated, and there exist many standardized protocols that come with formal

TABLE I
SECURITY REQUIREMENTS FOR V2SU KEY ESTABLISHMENT

| | |
|---|---|
| 1) | *Secrecy of the session key.* Upon completion of the key establishment method, the service unit and the vehicle should have established a session key which is known to the vehicle and service unit only. |
| 2) | *Uniqueness of the session key.* Each run of the key establishment method should produce distinct, independent session keys. |
| 3) | *Service unit authentication.* Upon completion of the key establishment method, if a vehicle believes it is communicating with a service unit on the session with key $k$ and parameters $p_1, \ldots, p_n$ then there is indeed an authentic service unit that is executing a session with key $k$ and parameters $p_1, \ldots, p_n$. |
| 4) | *Vehicle authentication.* Upon completion of the key establishment method, if a service unit believes it is communicating with a vehicle on the session with key $k$ and parameters $p_1, \ldots, p_n$ then there is indeed an authentic vehicle that is executing a session with key $k$ and parameters $p_1, \ldots, p_n$. |
| 5) | *Agreement with physical setup.* Upon completion of the key establishment method, the service unit and vehicle should also be linked by the respective physical setup. |

TABLE II
CONNECTION COMPROMISE STATES FOLLOWING A BREACH OF V2SU KEY ESTABLISHMENT

| | |
|---|---|
| 1) | *Person-in-the-middle (PitM).* The service unit has a connection secured by session key $K$ and the vehicle has a connection secured by key $K'$ but the attacker knows both $K$ and $K'$. |
| 2) | *Impersonation to service unit (Imp2SU).* The service unit has a connection secured by session key $K$ but the attacker knows $K$. |
| 3) | *Impersonation to vehicle (Imp2V).* The vehicle has a connection secured by session key $K$ but the attacker knows $K$. |
| 4) | *Parameter mismatch (ParsMismatch).* A peer has a connection secured by session key $K$ and for a session with parameters $p_1, \ldots, p_n$, and another peer has a connection secured also by $K$ and for a session with parameters $p'_1, \ldots, p'_n$, and the attacker does not know $K$, but there is $i \in [1, n]$ such that $p_i \neq p'_i$. |
| 5) | *Mismatch with physical setup (PhysMismatch).* A peer $P$ shares a connection secured by session key $K$ with another peer $P'$, and the attacker does not know $K$, but $P$ and $P'$ are not linked by the respective physical setup. |

security proofs. One example is the handshake protocol of Transport Layer Security (TLS). The advantage of AKE protocols is that they are designed to be secure in the presence of active adversaries: their security proofs assume an attacker who has complete control of the network. The drawback is that communication partners need to pre-share a security context such as a pre-shared long-term secret or a public key infrastructure. This typically results in a key management overhead, which can in turn be the source of further threats to the system.

SDP [6] schemes make do without a pre-shared security context but instead rely on so-called Out-of-Band (OoB) channels to safeguard against person-in-the-middle (PitM) attacks. These schemes have been widely adopted for Internet of Things (IoT) and personal devices. One example is Bluetooth pairing of a device to one's smartphone. Often the human user is used as the OoB channel; other schemes make use of properties of wireless channels such as Near Field Communication (NFC). The challenge is that the OoB channel must provide authenticity, and it is not always possible to validate this to a high assurance level: e.g., because a human user is involved or because it is difficult to establish that the wireless channel indeed satisfies authenticity. The great advantage of SDP in our context is that it makes do without a pre-established security context. Moreover, it will help us to achieve Property (5): to pair up two devices typically comes with proximity or some physical interaction, and in our context this can be woven into the procedure of the physical setup of the two machines.

*Security Engineering for Safety — Challenge:* How to assess the threats and risks that come with a particular key establishment method in our context? And how to iteratively refine a key establishment method under development such that risk is mitigated to an acceptable level? At first sight, one might be tempted to proceed as follows: assess the safety impact when the key establishment method maximally fails (i.e., when the attacker has full control over the connection); derive a safety level, and translate this into a Common Criteria security assurance level; hand this over to a company that provides key establishment products; and acquire a product with the corresponding Common Criteria certificate.

However, this approach has the drawback that it closes the door to measures on the cyber-physical service itself, and hence, to measures that mitigate the safety impact directly. Moreover, in our context where actors come from different security domains we cannot exclude insider attacks, and hence, this approach might overlook some threats that cannot be reduced in their likelihood by even the highest assurance level.

*Connection Compromise States:* Instead, we wish to reflect that a successful attack against a key establishment method can have different outcomes, and that certain outcomes might be easier to achieve for the attacker than others. To this end, we identify in which ways a supposedly secure connection can be compromised following a breach of the key establishment method. The resulting *connection compromise states*[1] are described in Table II and illustrated in Figure 2.

We now explain the rationale behind the connection compromise states. Assume a vehicle $V$ is undergoing maintenance at some location $L$. We explore how key establishment could have failed from the view of $V$, and from the view of a service unit at $L$ respectively. Figure 3 shows the derivation from the view of a service unit $U$ at $L$. The derivation from the view of $V$ is similar.

Assume that $U$ has established a session key $K$, supposedly with $V$. In the left branch of Figure 3 we consider the case when secrecy of $K$ has been breached. Then the attacker knows $K$, and will be able to run the connection with $U$ impersonating $V$. Hence, the attacker has reached the connection compromise state *impersonation to service unit (Imp2SU)*.

Next we ask whether $V$ has established a key $K'$ for $S_U$, the service provided by $U$ (where the case $K' = K$ is included).

[1]It is important to note that here we only consider compromise states directly derived from a failure of the key establishment goals. Other compromise states, e.g., such as those that result from attacks against session management such as session hijacking, fixation or riding are out of our scope.

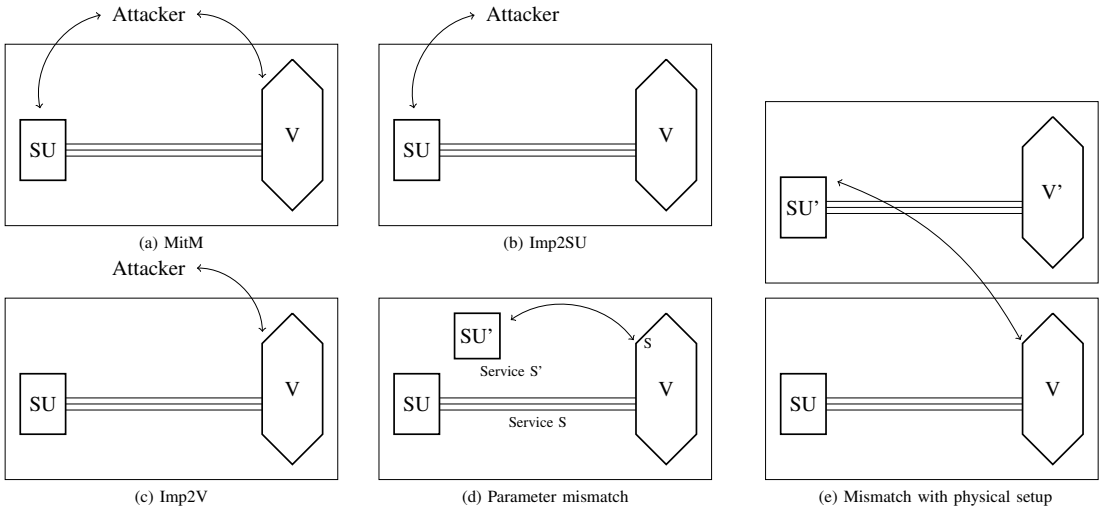


Fig. 2. Illustration of the connection compromise states (resulting from a failure of key establishment[1])

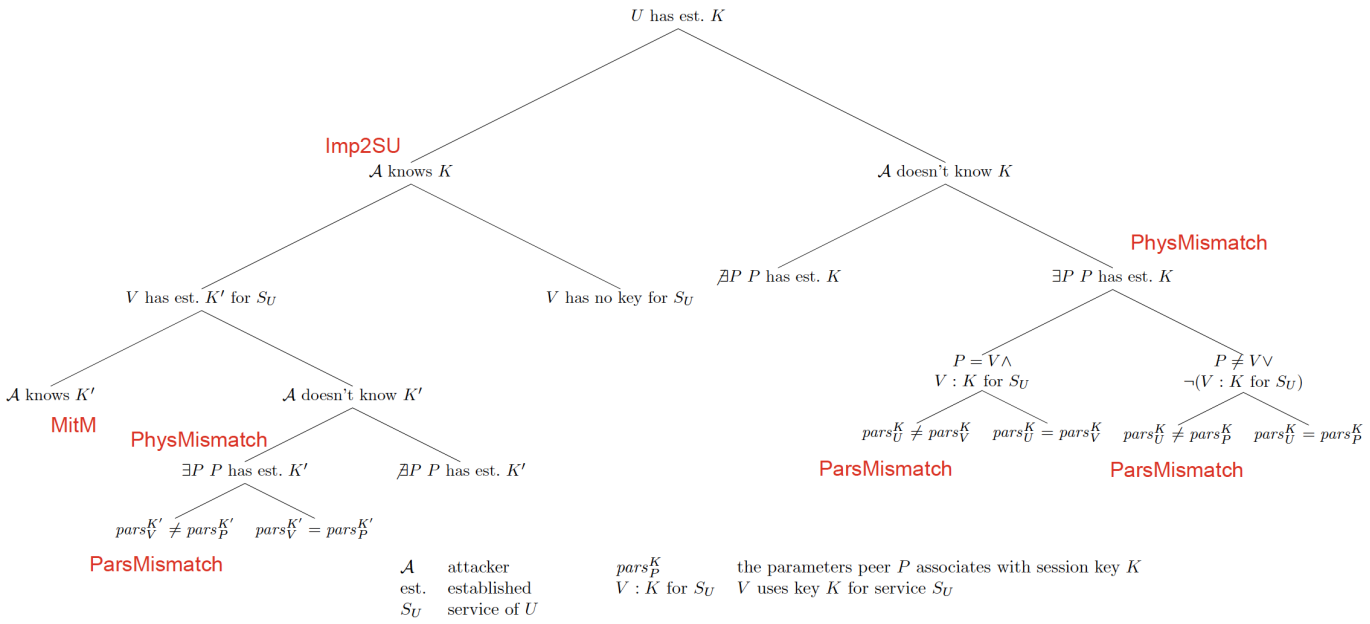


Fig. 3. Deriving the connection compromise states for the view of service units

If this is so, and secrecy of $K'$ is also breached then, in addition, the attacker will be able to impersonate $U$ to $V$. Hence, the attacker has full control over the communication between $V$ and $U$: they have reached the *person-in-the-middle (PitM)* connection compromise state.

If $V$ has established a key $K'$ for $S_U$ but the attacker does not know $K'$ then either $V$ must have established the key with a peer other than $U$, say $P$, or there is no peer who has currently established the key $K'$. The latter case can be avoided by $U$ itself when we ensure that the key establishment method contains a freshness proof and a key confirmation step.

The first case when such a peer $P$ exists brings with it a violation of agreement with physical setup: $U$ is or will be physically linked to $V$ rather than $P$. Hence, we have reached a *mismatch with physical setup* connection compromise state. Moreover, this can go along with a violation of authentication in that there is no agreement with one of the parameters. Then, in addition, we have reached a *parameter mismatch* state.

We still need to consider the case when $V$ has no key established for $S_U$. Since this case does not directly influence the service $S_U$ we will not expand this further here. This case is covered by the analogous derivation from the view of $V$.

Note that while it is often the attacker's best strategy to breach vehicle authentication, service unit authentication, or both respectively in order to reach Imp2SU, Imp2V, or PitM respectively, it is not necessary to do so: e.g., the key

could have been revealed after a successful run of the key establishment method between two authentic parties.

Let us now turn to the top right branch of Figure 3, and explore how key establishment could have failed while secrecy of $K$ is *not* breached. Then either there exists a peer $P$ that has established $K$ (right branch) or not (left branch). Similarly to above, the latter case can be avoided by $U$ itself when we ensure that the key establishment method contains a freshness proof and a key confirmation step. In the first case, we ask whether $P$ is indeed $V$, and $V$ indeed uses $K$ for service $S_U$. If this is so then there can at most be a *parameter mismatch*. If this is not so then we have a *mismatch with physical setup*. Moreover, this can also go along with a *parameter mismatch*. To reach a parameter mismatch state the attacker always has to breach authentication. Clearly, a mismatch with physical setup state always breaches agreement with physical setup.

*Security Engineering for Safety — Approach:* The security engineering activities can now be carried out in a structured and systematic fashion as follows:

1) The security experts identify the threats against the key establishment method under investigation, and assess for each connection compromise state the likelihood that this state can be reached by an attacker.
2) The safety and process engineers of the vehicle and the maintenance procedure assess for each connection compromise state what the severity of impact on safety (and perhaps other factors) will be if the attacker manages to reach this state. Moreover, they explore whether and how the impact can be mitigated by process measures.
3) At synchronization points safety and security experts together decide whether the combination of the current assessments of threat likelihood and safety impact result in an acceptable risk level. If not the workflow will be repeated in an iterative fashion until an optimal solution is reached. Finally, assurance levels for the security components and the mitigation safety measures will be derived, and forwarded for development, or product integration respectively.

We will discuss a workflow for the activities of Part (2) in more detail in Section V since this is where simulation plays a crucial role throughout. Part (1) will be illustrated via our case study. Here simulation might also play an important role, e.g., to analyse channel properties with respect to a SDP scheme. For a detailed analysis we employ the tools for formal protocol verification, such as the Tamarin Protocol Verifier [7].

## III. TAGA: A Touch and Go Assistant in the Aerospace Domain

We now illustrate our approach by means of the real-world use case TAGA.

### A. Preliminaries

In the following, we will make use of the basic Diffie-Hellman exchange as well as authenticated Diffie-Hellman protocols. We assume a cyclic group $G$ of prime order $n$, and a generator $P$ of $G$ such that the decisional Diffie-Hellman problem is hard in $G$. The domain parameters $G$, $n$, and $P$ can be fixed or sent as part of the first message. We use small letters to denote elements of the field $\mathbb{Z}_n^*$, and capital letters for elements of $G$. A key pair in the protocols consists of a public key $T$, which is a group element, and a private key $t$, which is an element of the field $\mathbb{Z}_n^*$ such that $T = tP$. Group operations are written additively ($A + B$, and $cA$) consistent with notation for elliptic curve cryptography.

TABLE III
Protocol notation

| | |
|---|---|
| $G$ | ID of ground unit |
| $A$ | ID of aircraft |
| $S$ | Service name of ground unit |
| $L$ | Location (i.e., parking slot) of the process |
| $I$ | Intruder |
| $ssid_A$ | SSID of the aircraft's WLAN |
| $ssid_I$ | SSID of the intruder's WLAN |
| $R_X$, $r_X$ | Ephemeral public and private DH key of party $X$ |
| $W_X$, $w_X$ | Long-term public and private DH key of party $X$ |
| $mac$ | A message authentication code algorithm |
| $H$ | A cryptographic hash function |
| $KDF_1$, $KDF_2$ | Key derivation functions |
| $K$ | Resulting session key |
| $K'$ | Derived mac key |
| $m_1 \| \| m_2$ | The concatenation of messages $m_1$ and $m_2$ |

To describe the protocols we use the notation presented in Table III. Moreover, we use DH key short for Diffie-Hellman key, GU short for ground unit, AC short for aircraft, and OP short for Operator.

### B. The TAGA Protoype

*The TAGA Pairing Process:* The prototype of TAGA pairing is based on an unauthenticated three-pass key establishment protocol, where the third pass is a key confirmation step. It is illustrated in Figure 4 for the case when the Diffie-Hellman key exchange is used as the underlying protocol.

The operator performs a first NFC tap at the ground unit. Thereby a first message $M_1$ is written to the card. $M_1$ contains information necessary for establishing the key together with the ID of the ground unit and the service that it provides. Then the operator walks to the aircraft. Typically they will also carry a supply hose; e.g., for pre-conditioning they will carry the air supply hose.

At the aircraft, the operator first performs some physical setup, such as connecting the supply hose to the supply port, and then carries out the second NFC tap. Thereby, $M_1$ is transferred to the aircraft's TAGA controller, and a second message $M_2$ is written onto the card. $M_2$ contains information necessary for establishing the key together with the ID of the aircraft and access data to its WLAN such as the SSID. $M_2$ also contains a ciphertext to grant key confirmation to the ground unit. The operator then walks back to the ground unit.

Back at the ground unit, the operator carries out a final NFC tap, and transfers $M_2$ to the ground unit's TAGA controller. The ground unit is now able to connect to the aircraft's WLAN. A third message is passed over the WLAN connection to achieve key confirmation to the aircraft. Finally, the operator

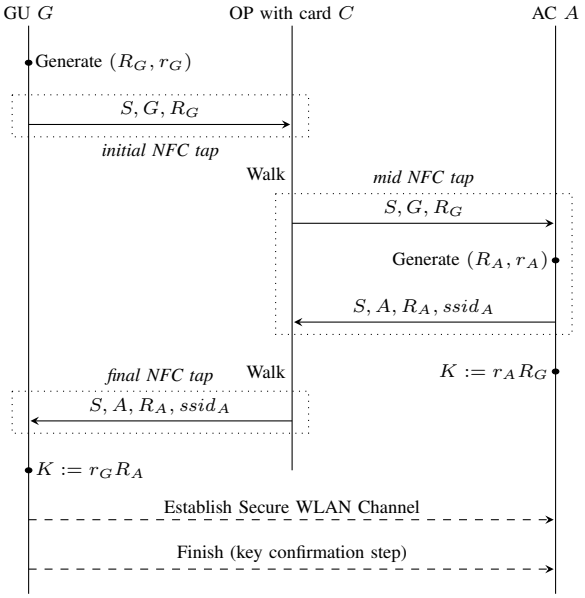


Fig. 4. TAGA pairing with Diffie-Hellman key exchange

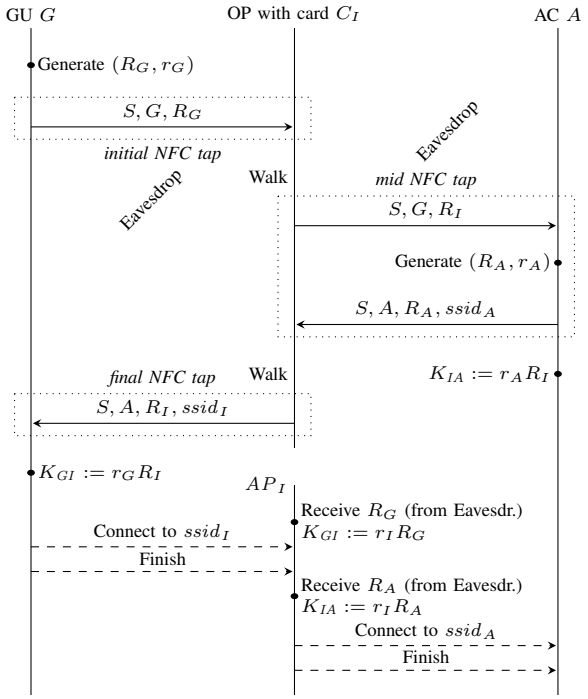


Fig. 5. Person-in-the-middle attack by card swapping and eavesdropping

activates the ground unit; e.g., for pre-conditioning they switch on the air supply. Now the ground unit and the aircraft are ready to carry out the service using M2M communication.

*Threats against the TAGA Channel:* Even though TAGA takes place in a secure zone, where only authorized personnel have access, our analysis has shown that there are many indirect ways of compromising the TAGA channel. One example is that the attacker might swap a counterfeit card for the TAGA card, e.g., while the operator takes a break. Another example is that the attacker might eavesdrop on the NFC exchange from outside the secure zone of the turnaround, e.g., by using a special antenna to increase the nominal range of NFC.

The following example shows that the combination of card swapping and eavesdropping already allows the attacker to implement the classic person-in-the-middle attack against the basic Diffie-Hellman exchange over the TAGA channel.

*Example* 1 (PitM by Swap & Eavesdrop). Let $A$ be an aircraft and $G$ be a ground unit at parking slot $L$ so that $G$ is to service $A$. In preparation, the attacker swaps his own prepped card $C_I$ for the operator's card, e.g., while the operator is on a break. Moreover, the attacker sets up NFC eavesdropping capability, and their own WLAN access point $AP_I$ in the range of $L$. Both $C_I$ and $AP_I$ are prepped with a fixed DH key pair $(r_I, R_I)$, and the SSID $ssid_I$ of the attacker's WLAN.

The attack then proceeds as depicted in Figure 5. The card $C_I$ carries out the first tap as usual. However, with the second tap the counterfeit card writes the attacker's public key $R_I$ to $A$ rather than $G$'s public key $R_G$. Similarly, with the third tap the card writes $R_I$ and $ssid_I$ to $G$ rather than $A$'s public key $R_A$ and SSID $ssid_A$. Hence, $G$ computes session key $K_{GI}$ based on $r_G$ and $R_I$, and $A$ computes session key $K_{IA}$ based on $r_A$ and $R_I$.

To be able to compute the same keys the attacker needs to get $R_G$ and $R_A$ onto their access point $AP_I$. Even if the card only has a passive NFC interface they can use eavesdropping to do so. Once they have computed $K_{GI}$ and $K_{IA}$ they can establish the corresponding channels, and mount a PitM attack against the M2M communication between $G$ and $A$.

*Estimating the Safety Impact:* To estimate the severity of impact of a PitM connection compromise we consider the two ground services fuelling and pre-conditioning. Our examples show that while for fuelling the safety impact is controlled by inbuilt safety measures this is not the case for pre-conditioning, and the safety impact is potentially high.

*Example* 2 (Fuelling). The attacker can forge fuel orders, and induce the fuel truck to load an insufficient or surplus amount of fuel. While this can be highly disruptive there is no safety impact. Since the aircraft measures the fuel itself it will notice if the loaded fuel is not sufficient. Moreover, if the attacker tries to cause spillage (and hence, a fire hazard) by too large a fuel order this will not succeed since the backflow will stop the pump of the fuel truck.

*Example* 3 (Pre-Conditioning). The attacker can forge airflow parameters and sensor values that will induce the pre-conditioning unit to apply air pressure and temperature unsuitable to the aircraft. This can be highly damaging: if the cooling process is too fast then water in the pipes can quickly become frozen and clog up the pipes. This can happen very quickly: e.g., with the lowest inlet temperature within 30 seconds, with safety considerations still within 100 seconds. The resulting backflow will be detected by the pre-conditioning unit. However, in the worst case pipes might already have burst. In any case the pipes have to be checked for damage

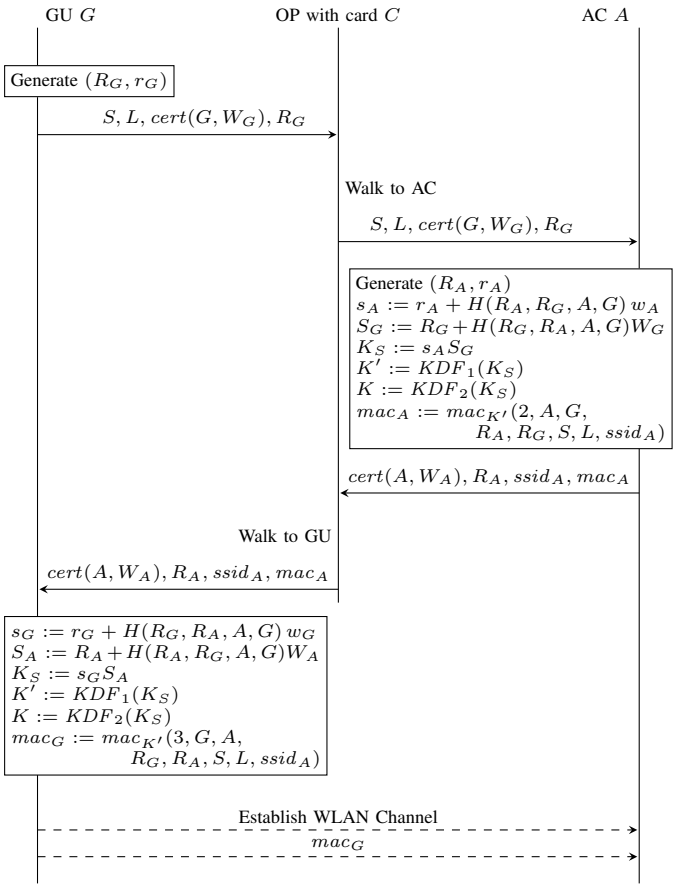


Fig. 6. TAGA pairing based on the FHMQV protocol

afterwards, which is a costly procedure.

In the worst case, the attacker could try to optimize the attack based on the sensor values sent by the aircraft: they could try to control the airflow in a way that maximizes the strain on the pipes without this being detected during service time but with a high risk that pipes burst during flight.

Our analysis of the prototype has shown that one either needs to refine TAGA by better protecting the TAGA channel, or by using an AKE protocol instead of the basic Diffie-Hellman exchange. In the following, we illustrate aspects of the latter refinement. A solution in line with the first refinement can be found in [8].

### C. Refinement: Authenticated TAGA

*The Authenticated Setting:* In the setting of authenticated TAGA, every aircraft $A$ has a long-term key pair $(W_A, w_A)$, where $W_A$ is the public key and $w_A$ is the private key. Moreover, $A$ holds a certificate for its public key $W_A$, which is issued by the airline $\mathcal{A}$ that owns $A$ (or an entity commissioned by $\mathcal{A}$). We denote the certificate by $cert_{\mathcal{A}}(A, W_A, T_A, V_A)$, where $T_A$ is the aircraft type of $A$, and $V_A$ specifies the validity period of the certificate.

Analogously, every ground unit $G$ has a long-term key pair $(W_G, w_G)$, and a certificate for its public key $W_G$, which is issued by the airport $\mathcal{H}$ that harbours $G$ (or an entity commissioned by $\mathcal{H}$). We denote the certificate by $cert_{\mathcal{H}}(G, W_G, S_G, V_G)$, where $S_G$ is the service type of $G$ and $V_G$ is the validity period of the certificate.

We assume that every aircraft has installed the root certificates of those airports it intends to land at, and each ground unit has installed the root certificates of those airlines it is authorized to handle. For short notation, we often write a certificate $cert_{\mathcal{A}}(A, W_A, T_A, V_A)$ as $cert(A, W_A)$ when the issueing party, type of aircraft or service, and validity period are implicitly clear from the context. Similarly, we often write $cert(G, W_G)$ short for $cert_{\mathcal{H}}(G, W_G, S_G, V_G)$.

Figure 6 shows TAGA based on the *Fully Hashed Menezes-Qu-Vanstone protocol (FHMQV)* [9], [10], where for TAGA we include service and location into the key confirmation step. FHMQV is one of the strongest protocols regarding security, resilience and efficiency, and comes with a security proof. It satisfies all our secrecy and authentication requirements, i.e., Properties (1)–(4) of Table I, even when assuming that the attacker has full control of the TAGA channel. Our requirement 'Agreement with physical setup', i.e., Property (5), can also be guaranteed. Since we have included the parameters service and location into the key confirmation step the ground unit and aircraft will agree on service and location as part of the authentication guarantees. Then to obtain Property (5) the aircraft and ground unit only need to carry out a handshake of 'ready for service' messages once the secure channel is established.

*The Threat of Long-Term Key Compromise:* While secure AKE protocols are designed to withstand an attacker who has full control of the network they are vulnerable to the threat of *long-term key compromises*. We say the attacker has obtained a *long-term key compromise (LTKC)* of the aircraft $A$ if they have managed to get hold of credentials that authenticate $A$: a public/private key pair $(W_A, w_A)$ and a valid certificate $cert(A, W_A)$, which asserts that $W_A$ belongs to $A$. The definition for a ground unit $G$ is analogous.

Given the LTKC of a party $P$, it is unavoidable that the attacker can impersonate $P$ to other parties. In classical settings of AKE protocols this will typically impact on the resources of $P$, and only $P$, itself. However, in our setting, a LTKC can have a wider impact. The following example shows how the attacker can use the LTKC of some aircraft $A_I$ (possibly of an airline with key management of low security quality) to impersonate $A_I$ to a ground unit that is physically connected to another aircraft $A$ (possibly of an airline with key management of high security quality).

*Example* 4 (Impersonation to Ground Unit with LTKC of any Aircraft). Let $A_I$ be a real or non-existent aircraft of airline $\mathcal{A}_I$, and assume that the attacker has achieved a LTKC of $A_I$. Further, let $A$ be an aircraft of airline $\mathcal{A}$, and $G$ be a ground unit at airport $\mathcal{H}$ such that $G$ provides service $S$ to $A$ during turnaround at parking slot $L$. In preparation, the attacker swaps their own counterfeit card $C_I$ for the card of $G$'s operator. Moreover, the attacker sets up NFC eavesdropping capability, and their own WLAN access point $AP_I$ within range of $L$.

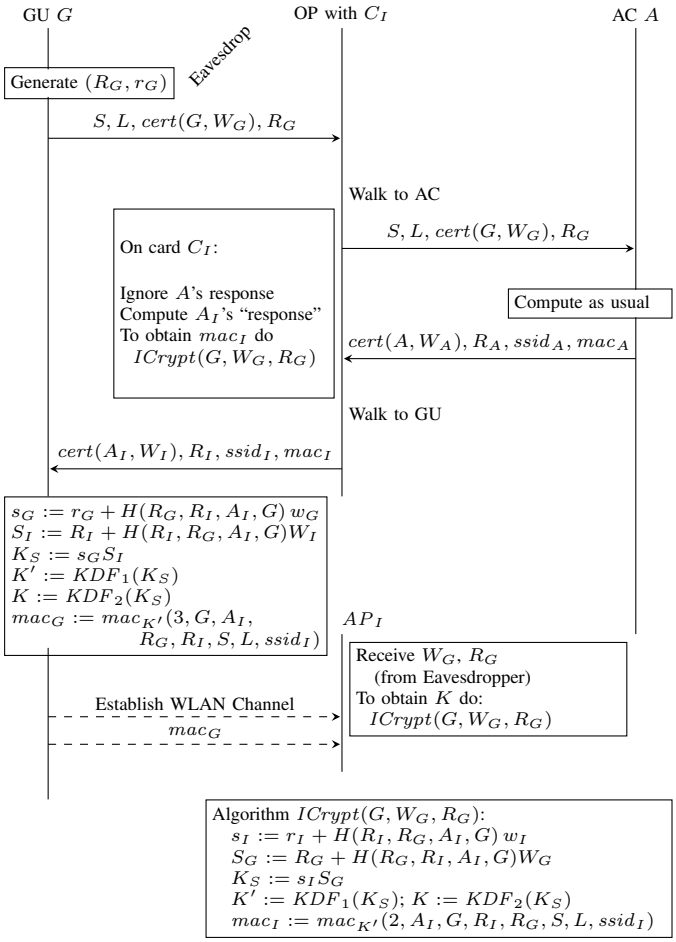


Fig. 7. Impersonation to ground unit with LTKC of any aircraft

Both $AP_I$ and $C_I$ are prepped with $A_I$'s long-term credentials $w_I$ and $cert(A_I, W_I)$, a fixed ephemeral key pair $(r_I, R_I)$, and the SSID $ssid_I$ of the attacker's WLAN.

Then the attacker can proceed as shown in Figure 7: they simply establish a key with $G$ using $A_I$'s credentials rather than those of $A$. Since $A_I$'s ephemeral key pair can be fixed beforehand, the resulting session key can be computed independently on the card $C_I$, and the attacker's WLAN point $AP_I$ respectively. The latter only needs to receive $G$'s public keys by relay from the eavesdropping device.

*Estimating the Safety Impact:* The attacker has only obtained an Imp2SU connection compromise, and one may hope that this comes with less safety impact than PitM. However, Imp2SU still allows the attacker to feed any sensor values they like to the ground unit while the ground unit thinks this information stems from the aircraft and adjusts the service correspondingly. The safety impact is potentially high for pre-conditioning.

*Example* 5 (Pre-Conditioning). The attacker feeds in airflow parameters and sensor values, and the ground unit will control the airflow based on this information. Since the air supply leads directly into the mixer unit of the aircraft this will take immediate effect without the aircraft itself having to open a valve or the like first. Crew or ground staff might notice that something is wrong and switch off the air supply manually. However, as explained in Example 3 damage can occur quickly and this might be too late. In contrast to the PitM attack, the attacker is not able to obtain sensor values sent by the aircraft, and, hence, they are not able to optimize the attack based on such information.

Given the potential safety impact and scale of the attack (given one LTKC of any airline) it is clear that a further refinement of the TAGA method is necessary. In particular, it is worth exploring measures that work on the ground service itself: one airline will not have much control over the security infrastructures managed by another. In addition, in our context of critical infrastructures one cannot write off that a state actor might take influence to obtain and abuse valid aircraft credentials of an airline in its realm. We will propose several measures that will address this situation in Section IV.

Given the LTKC of a ground unit, a simple check can ensure that the compromised credentials cannot be employed by the attacker beyond the realm of the airport where the ground unit operates: the aircraft can simply check the airport in the certificate against its current location. Moreover, when the physical control of the service lies entirely with the ground unit then an Imp2V attack is usually less harmful.

### *D. Intricate Consequences of LTKCs: Key-Compromise Impersonation*

Given that a participant $X$ has a LTKC, it is clear that the attacker can impersonate $X$ to any other particpant. And this is what we have considered so far. However, a more intricate question to ask is whether this enables the attacker to impersonate any other participant to $X$. We then say the attacker can carry out a *Key-Compromise Impersonation (KCI)* attack [11]. In our setting this would mean: given a LTKC of ground unit $G_I$, the attacker will be able to stage an Imp2SU attack against any aircraft serviced by $G_I$. Moreover, the attacker can combine each such KCI attack with a standard impersonation attack to obtain PitM capability.

Fortunately, many AKE protocols such as the FHMQV used here are resilient against KCI attacks. And hence, this attack with potentially large-scale impact can be excluded by choice of the protocol. We illustrate the KCI attack by a concrete example based on the *Unified Model (UM)* protocol (c.f. [12]). The UM, shown in Table IV, is another Diffie-Hellman protocol with implicit authentication. Moreover, it is well-known to be vulnerable to KCI attacks [12].

*Example* 6 (KCI Attack against UM). Let $G_I$ be a ground unit for service $S$ at airport $\mathcal{H}$, for which the attacker has achieved a LTKC. Further, let $A$ be any aircraft that is serviced by $G_I$, say at parking slot $L$. In preparation, the attacker swaps the NFC card of $G_I$'s operator with their own prepared card $U_I$. Moreover, they set up NFC eavesdropping capability, and their own WLAN access point $AP_I$ within range of $L$. Both, $AP_I$ and the card $U_I$, are prepped with

TABLE IV
THE UM PROTOCOL

1) $G$ generates $(R_G, r_G)$
   $G$ sends $S, L, cert(G, W_G), R_G$
2) $A$ receives and validates the message
   $A$ generates $(R_A, r_A)$
   $A$ computes $K := H(w_A W_G \,||\, r_A R_G)$
   $A$ computes $mac_A := mac_K(2, A, G, R_A, R_G, S, L, ssid_A)$
   $A$ sends $cert(A, W_A), R_A, ssid_A, mac_A$
3) $G$ receives and validates the message
   $G$ computes $K := H(w_G W_A \,||\, r_G R_A)$
   $G$ validates $mac_A$
   $G$ computes $mac_G := mac_K(3, G, A, R_G, R_A, S, L, ssid_A)$
   $G$ establishes the WLAN connection and sends $mac_G$.

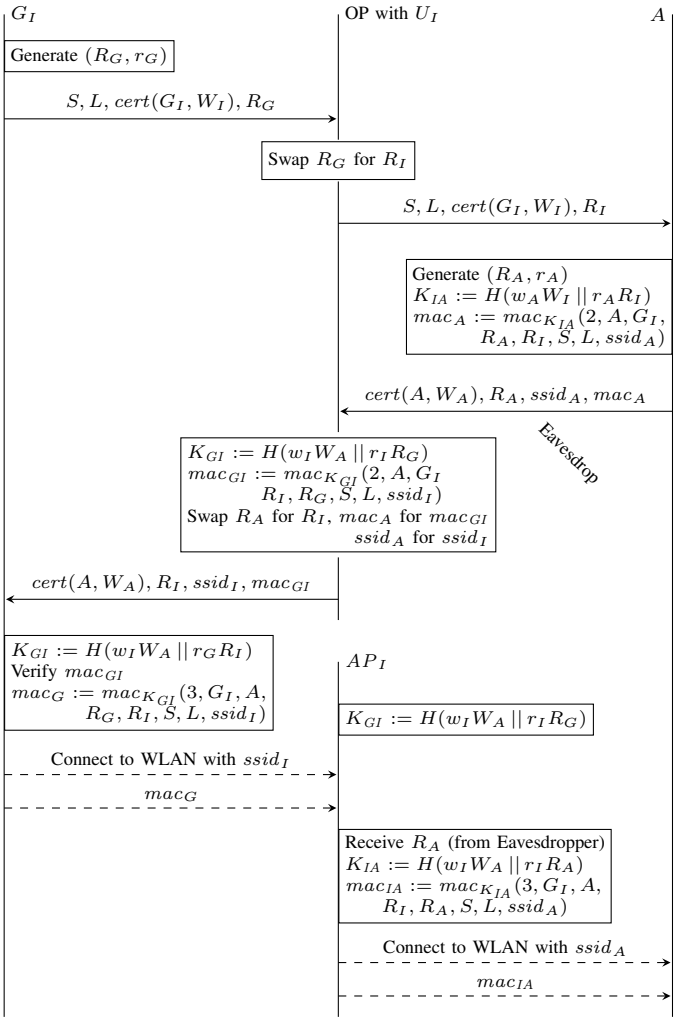


Fig. 8. KCI attack against TAGA with the UM protocol

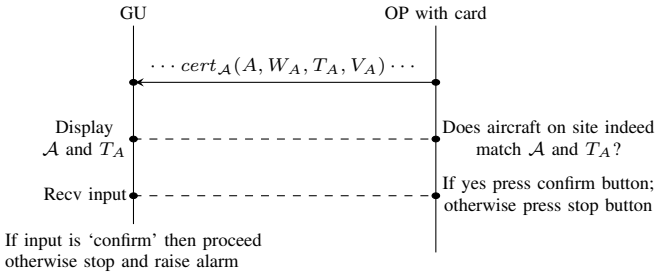


Fig. 9. Last NFC tap extended by human verification of aircraft domain

a fixed ephemeral key pair $(r_I, R_I)$ and the SSID $ssid_I$ for the WLAN with $G_I$. In addition, $AP_I$ is prepped with $G_I$'s credentials $w_I$ and $cert(G_I, W_I)$, and $A$'s long-term public key $W_A$. Alternatively, $W_A$ can be obtained by eavesdropping.

Then the attacker can proceed as shown in Figure 8. The interaction with $G_I$ consitutes a KCI attack, where the attacker impersonates $A$: they can compute the same key $K_{GI}$ as $G_I$ by using their knowledge of $w_I$ rather than $w_A$ (and their own ephemeral key pair). The interaction with $A$ constitutes a standard impersonation attack, where the attacker can impersonate $G_I$ due to their knowledge of $w_I$, and establishes a key $K_{IA}$ with $A$. Altogether, the attacker can now fully control the M2M communication of $A$ and $G$ as the PitM.

This illustrates that as long as KCI attacks are a threat it is not possible to protect against one-sided LTKCs by detection measures against impersonation attacks. Hence, in the security analysis it is important that all paths are investigated how an attacker could obtain one of the connection compromise states, even the most intricate ones.

## IV. CYBER-PHYSICAL MITIGATION MEASURES

The last section has shown that in a setting where entities of several security domains interact in an ad hoc fashion (so that their digital identities are not known prior to key establishment) the likelihood of certain LTKCs might be comparatively high. We now propose several measures that protect against Imp2SU or even PitM. To allow for a more precise description we formulate most of the measures in the setting of TAGA. Note, however, that it is straightfoward to translate the measures to other settings, and to employ them against Imp2V analogously. A summary is provided in Section IV-E.

### A. Human Verification of Aircraft Domain

The likelihood of Imp2SU against aircraft of an airline with high security standards can drastically be reduced if we ensure that the attacker cannot make the ground unit accept a certificate that does not agree with the domain of the aircraft that is actually on site. This can be achieved by a simple measure that makes use of the awareness of the human operator: while the ground unit has no means to verify that the received certificate (and information therein) indeed belongs to the aircraft present at the parking slot, the operator has sight of the aircraft. Hence, they are able to verify that visually

observable features of the aircraft such as its type and airline agree with the information received by the ground unit.

*Measure* 1 ('Two eyes' verification of aircraft domain (2EV)). Assume the TAGA controller of the ground unit is equipped with a display and two input buttons: one to confirm, and the second to stop the process and raise an alarm. Then the last NFC tap can be extended by human verification as illustrated in Figure 9. First, the operator transfers the second message by NFC tap to the ground unit as usual. Recall that this message contains a certificate $cert_{\mathcal{A}}(A, W_A, T_A, V_A)$, where $A$ is the ID of the aircraft, $\mathcal{A}$ is the airline of $A$, and $T_A$ is the type of $A$. Second, the ground unit shows $\mathcal{A}$ and $T_A$ on its display, and the operator verifies whether the aircraft they see on the parking slot is indeed of airline $\mathcal{A}$ and type $T_A$. If yes, then they will confirm the process; otherwise they will stop the process and raise an alarm.

Unintended errors of the operator can be kept small: they can be trained to keep awareness by injection of false alarms (similary to security screening at airports). It is also possible to implement this with dual control.

*Measure* 2 ('Four eyes' verification of aircraft domain (4EV)). For increased security a member of the aircraft crew can accompany the ground operator and perform the visual verification as well.

Note that if the underlying protocol is not KCI resilient then the attack shown in Example 6 is possible even when this measure is in place.

### B. Time-based Detection

An attacker who carries out an Imp2SU attack in the TAGA setting will need to ensure that the NFC tap at the aircraft looks successful to the operator. Assume we add a 'two eye' verification step in which the operator must verify that the aircraft has indeed received a TAGA request for the service they carry out, and hence starts a respective session. An attacker who only has the capability to reach Imp2SU will not be able to successfully complete the session, and thereby be caught out: the aircraft will raise an alarm when a session is still pending after an unusually long time. Operators can then check what is going on, and, deactivate the ground unit before damage occurs.

For the latter to work it is important that the ground unit defers all safety-critical processing until it can be sure that no alarm will be raised. How long should the ground unit wait for? This can be derived as follows.

Fix a ground service $S$. Let $t^A_{max}$ be the maximal time that the aircraft waits after starting a new TAGA session for service $S$ to receive the corresponding Finish message. Let $t_{min}$ be the minimal time required from the point after the second NFC tap at the aircraft up to the point when the ground unit has received the final NFC tap. $t^A_{max}$ and $t_{min}$ will mainly be determined by how long the operator needs to walk from the aircraft's TAGA controller back to the ground unit.

Let $t^{stop}_{max}$ be the time the operator maximally needs to carry out an emergency stop at the ground unit once they have heard the alarm of the aircraft. $t^{stop}_{max}$ will include the time the operator will need to walk back to the ground unit from anywhere where they could potentially stay in the meantime.

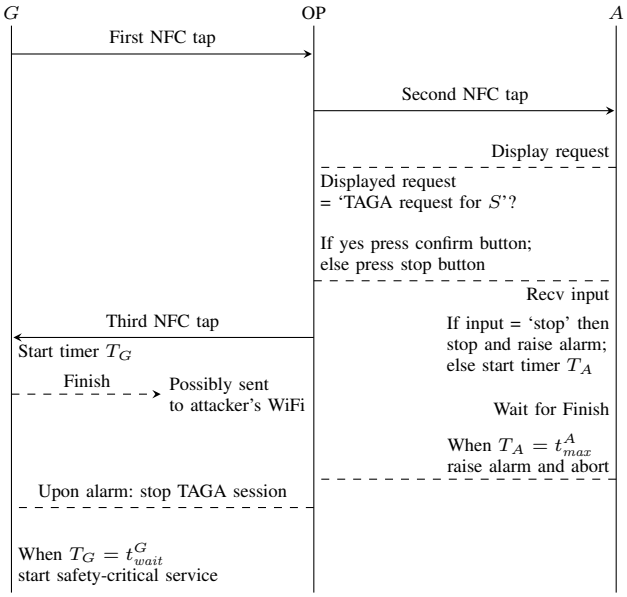


Fig. 10. Time-based detection

Then clearly we have: if the TAGA session of the ground unit has not been stopped within $t^G_{wait} = t^A_{max} - t_{min} + t^{stop}_{max}$ time after it has received the final NFC tap then the local aircraft must have successfully established a session for $S$, and the ground unit can start with safety-critical processing.

*Measure* 3 (Time-based detection (TD)). Let $S$, $t^A_{max}$ and $t^G_{wait}$ be given as above. Assume the TAGA controller of the aircraft is equipped with a display and two input buttons: one to confirm, and the second to stop the process and raise an alarm. Then TAGA can be extended by a time-based detection measure as illustrated in Figure 10. At the second NFC tap, the operator verifies with the help of the display that the request received by the aircraft coincides with a TAGA request for the service $S$ they carry out. If this is confirmed the aircraft will start a timer, say $T_A$. If the corresponding Finish message is not received before $T_A$ has reached $t^A_{max}$ then the aircraft will raise an alarm. This will trigger the operator to go to the ground unit and stop the unit's service. The ground unit defers any safety-critical processes or settings until $t^G_{wait}$ time has passed from the point of the third NFC tap onwards. If no alarm has been raised and the operator has not stopped the TAGA session by then the ground unit can conclude that no attack has occurred, and continue as usual.

This measure comes with a trade-off between usability and efficiency: if $t^A_{max}$ is set too large then the process takes a lot longer than necessary; if $t^A_{max}$ is set too small then there will be too many false positives and/or pressure on the operator to hurry. $t^{stop}_{max}$ can be chosen to be small when the operator is required to stay close to the ground unit.

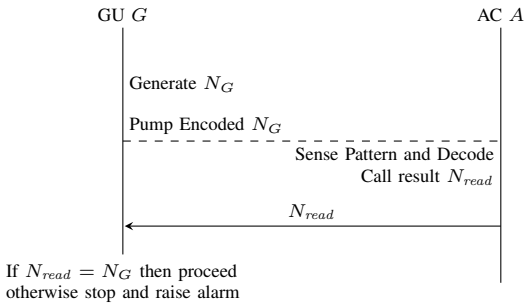


Fig. 11. Physical Challenge/Cyber Response

### C. Physical Challenge/Cyber Response

We now propose a measure that translates the standard scheme of challenge/response authentication into the concept of *physical challenge/cyber response:* the ground unit sends a challenge via the physical connection, e.g., encoded in a pattern of pulsating flow, which the aircraft must answer via the cyber channel. Thereby the physical connection is directly bound into the key establishment method.

*Measure* 4 (Physical Challenge/Cyber Response). Assume that the airpacks of the aircraft are equipped with mass airflow sensors that can detect a pattern of airflow changes and report it to its TAGA controller. Then a phase of physical challenge response can be included before the regular M2M communication starts as illustrated in Figure 11. The ground unit $G$ generates a random number of a fixed size, say $N_G$, and encodes this into a pattern of pulsating airflow. The aircraft $A$ reads the physical signal using its airflow sensors and decodes it back into a number, say $N_{read}$. $A$ then responds by sending $N_{read}$ back to $G$ via the cyber channel. $G$ checks whether $N_{read} = N_G$. If this is true then $G$ concludes that it speaks to the aircraft it is physically connected to: only this aircraft could have known $N_G$. If the numbers don't agree $G$ stops and raises an alarm.

Note that physical challenge/cyber response only counters Imp2SU attacks, and can be undermined by a PitM attacker.

The space of nonces must be sufficiently large to reduce the risk of guessing attacks: even when the attacker cannot receive the physical signal they can always guess the nonce $N_G$ and send it back via a cyber channel they have established with the ground unit by an impersonation attack. This brings about a trade-off between security and efficiency. For example: Say the physical channel allows a binary encoding of numbers in terms of high and low airflow (e.g., using stuffing to synchronize). Say an encoded bit requires 2 seconds to be transmitted, and a challenge shall maximally take 10 (or 20) seconds to be transmitted. Then one can use a space of 32 (or 1024) nonces, and the attacker has a 1/32 (or 1/1024) chance to guess correctly.

### D. Attack Detection during M2M Phase

Finally, one can make use of attack detection units that monitor the service during the M2M phase. We present here two examples. However, any attack detection system that detects anomalies falls into this category.

*Measure* 5 (Safety Check and Safety Alert). The vehicle or service unit could integrate sensors to check whether system variables such as temperature or pressure are about to cross safety limits. Then an alarm could be raised, and operators could deactivate the machine from which the danger emanates. Note that it is not possible to deactivate the machine automatically: it is the machine opposite to the one that raises the alarm that will need to be switched off. Moreover, since the communication channel is thought to be under attack it is not possible to reliably send a deactivation request message to the peer machine either.

*Measure* 6 (Physics-based Attack Detection). Physics-based attack detection employs a physical model of the normal behaviour of the system to monitor whether real-time measurements of system variables are consistent with the expected behaviour of the system [13], [14]. This concept could be applied in our context as follows. As with the previous measure the vehicle is equipped with sensors that take real-time measurements of system variables. A digital twin of the control of the service unit models the expected behaviour under the assumption that the service unit indeed receives the sensor values the vehicle communicates. If there is a deviation to the actual behaviour then an alarm will be raised. As with the safety check method, it is the opposite machine, here the service unit, that needs to be deactivated, and hence, this has to be carried out by operators.

The advantage of these measures is that they are independent of how key establishment has failed, and also work in the presence of attacks beyond those during key establishment. However, they might not be able to catch the attacks before damage has already occurred: since the physical impact is caused by the opposite machine there is the time delay between the alarm and the operator being able to switch off the service unit. Another challenge is that attacks might go unnoticed if the attacker chooses a stealthy strategy.

### E. Summary

In Table V we provide an overview of the measures. Most of them work against Imp2SU but can be masked out if the attacker has full PitM capability or can run a Mismatch attack in parallel (in the case of time-based detection). Measure 1 will detect any attack that makes use of a vehicle certificate from a different domain than the vehicle on site. If the protocol guarantees KCI (Key Compromise Impersonation) resilience [11], [12] then this measure will exclude all Imp2SU attacks.

Measure 1 has the advantage that it is independent of the actual service while Measures 3 and 4 are specific to the service, and might not always be an option. Measure 2 is dependent on the service only in that the time interals $t^A_{max}$ and $t^G_{wait}$ might differ across services, which is straightforward to manage by service-dependent configurations. More challenging might be if it turns out that the time intervals need to be adjusted across airports.

TABLE V
OVERVIEW OF THE MEASURES AND THEIR CHARACTERISTICS

| | Measure | Attack | Service-dependent? | Preventive? | Comments |
|---|---|---|---|---|---|
| 1 | 2/4-Eyes Verification | Use of non-domain vehicle cert. | no | yes | requires training of operators |
| 2 | Time-based Detection | Imp2SU** | configurable | configurable | requires training of operators |
| 3 | Physical Challenge/Cyber Response | Imp2SU* | yes | yes | |
| 4 | Attack Detection during M2M Phase | all | yes | no | |

** ... in the absence of PitM and Mismatch
* ... in the absence of PitM

Measures 1 and 3 are directly bound into the key establishment method, and are therefore preventive in that key establishment will not be successfully completed in the presence of the respective attack. Measure 2 can be implemented in a way so that the attack can be detected before any damage can be caused — in a trade-off with time. Measure 4 might not be able to prevent damage in general but has the advantage that it works for all connection compromise states including attacks that come after key establishment.

## V. ASSESSING AND MITIGATING THE SAFETY IMPACT

We now describe a workflow of how the engineers of the maintenance procedure can iteratively assess the severity of impact, and explore and assess means to mitigate it. The workflow consists of the following activities. They can systematically be performed for each of the services, and for each of the relevant connection compromise states. In each of the steps simulation plays a crucial role.

1) Initial estimation and, if applicable, demonstration of the safety impact.
2) Refined analysis of the safety impact.
3) Exploration and assessment of mitigation measures.

Then iterate steps (2) and (3) until risk is mitigated to an acceptable level.

### A. Initial Estimation of the Safety Impact

A first analysis of the safety impact is carried out. Usually, this can be done by hand by the engineers of the machines and maintenance process. This gives a first impression of whether a connection compromise state is critical or not. Our examples in Section III show that there can be differences across the services as well as across the connection compromise states.

It makes sense to carry out this initial step breadth-first for all services at hand. In this way one can learn early on if there are large differences between the risk levels across the services. Then one can e.g., partition them into several safety domains, or, mitigate the risk of individual services by additional measures.

Simulation can be an important tool at this stage to demonstrate the safety impact. This should not be underestimated: a demonstration is worth immensely more than a 1000 words when it comes to informing other team members or convincing management of the necessity of security measures (and their costs).

TABLE VI
ATTACKER'S STRATEGIC GOALS

The attacker's strategic goal could be as follows:
1) create maximal damage while the maintenance process takes place,
2) create maximal damage during the operation of the vehicle after the maintenance process has taken place,
3) create maximal disruption, e.g., in terms of delays, equipment cost, locations affected,

while
a) the attack does not remain stealthy,
b) the attack remains stealthy,
c) the attack potential can be demonstrated without being carried out (in view of ransomware attacks).

### B. Refined Analysis of the Safety Impact

Many outcomes of the first phase will require a more refined analysis. In the positive case, when the initial estimation has delivered the result that the safety impact is controlled by existing safety mechanisms (c.f. Example 2) it might be important to submit this outcome to closer examination. This is so because safety measures such as backflow valves will not have been desigend to withstand malicicous intent, and the forces or patterns applied might be different when the system is under attack. In the negative case, when the initial estimation has delivered the result that safety impact is to be expected it might be important to explore the attack capabilites in more detail, e.g., to determine whether the attack will only lead to disruption or put passengers at risk (c.f. Example 3).

For this phase we assume that the service under investigation is already modelled in a tool such as Stateflow/Simulink. The model then only needs to be extended to integrate the respective connection compromise state. We suggest to provide one channel component for each of the connection compromise states in addition to the original uncrompromised channel component. Then during evaluation one can switch between the different channel models as required.

The question remains of how to choose the input values for the attack simulations. E.g. to assess the Imp2SU state, which sensor inputs shall the attacker model communicate to the model of the service unit? At first sight, it might seem plausible to use the fault models typically used in safety analysis such as 'stuck at' or 'random'. However, this will not sufficiently reflect that during an attack the values are chosen by a purposeful attacker. We propose instead to identify the strategic goals an attacker might have, and to choose the

system inputs accordingly. In Table VI we show a first draft of such goals. We have separated out two dimensions: the type of damage an attacker intends to cause, and the attack mode, e.g., whether the attack shall remain stealthy or not. Note that, in particular for stealthy attacks, the input patterns might not be obvious. Then simulation also has an important role to play to find and optimize the system parameters accordingly.

It is a joint task for safety engineers and security engineers in cooperation with members of agencies such as the BSI (Bundesamt für Sicherheit in der Informationstechnik), the relevant authority in Germany, to assess the likelihood of such attacks: the first group can assess the necessary resources (e.g., knowledge, access to equipment) for an attack category, while the latter can assess whether corresponding groups with the respective strategic goals are able to obtain these resources.

### C. Exploration and Assessment of Mitigation Measures

In Section IV we have seen how measures that act on the physical part of the service can play an important role to mitigate the impact when key establishment fails.

Simulation can either be part of the measure itself as with cyber-physical attack detection in form of a digital twin or it can play a crucial role to validate the measure. There are several facets here: first, to validate whether the physics behind the method will indeed work. Second, to simulate and validate the actions of ground personnel in case of an alarm, e.g., to estimate the time it takes for them to deactive the respective machine. And third, to validate whether the time between the alarm and the deactivation is sufficiently short to reduce risk before damage is caused. Finally, co-simulation can be used for an overall validation. Again, simulation can also be used for parameter optimization. For any attack detection system it will be important to consider the evaluation criteria considered in [14]: the trade-off between the maximum deviation of critical system variables per time unit imposed by undetected attacks, and the expected time between false alarms.

## VI. Related Work

*Safety and Security Process:* In view of the increased use of wireless communication in transportation there has been a long-term undertaking to integrate both safety and security into norms and standards, and to devise appropriate methods for threat analysis and risk assessment. Important methods, originally from the automotive domain, are the one of the project EVITA [15], the one of the project HEAVENS [16], and the SAHARA method [17], which combines HARA (hazard analysis and risk assessment) from the safety domain with STRIDE [18] from the security domain. The standardization efforts in the automotive domain have culminated in the recent ISO/SAE 21434 Automotive Cybersecurity Standard (c.f. [19]). This has led to a maturing of the methods into tool-chains [20]. However, these methods and tools focus on high-level system aspects, and do not adequately capture and structure the level of key establishment.

*Authenticated Key Establishment Protocols:* The Diffie-Hellman key exchange [21] is the first key establishment protocol in the public key setting; indeed it was put forward at the same time as the idea of public key cryptography itself. Since then much effort has been gone into how to design and verify *authenticated key establishment protocols*, which can be used over an open channel and in the presence of an active adversary to securely establish a key [9], [11], [12]. It became clear that such protocols are vulnerable to subtle attacks, and it is now standard to formally verify security protocols by state-of-the-art symbolic protocol tools such as Tamarin [7], [22] or ProVerif [23] and/or to prove them secure in a cryptographic security model [10].

Both, design and verification of key establishment protocols is an ongoing activity: not least since the advent of quantum computing will also affect the protocols in use today. For example, the novel PQXDH (Post-Quantum Extended Diffie-Hellmann) protocol provides post-quantum forward secrecy while still being based on the discrete logarithm problem [24].

While there is very rigorous methodology to design and verify security protocols, most of the activities are focussed on the protocols themselves under standard assumptions (e.g., concerning key reveals) motivated by cyber-only applications. Here we have focussed on outside the box challenges unique to our M2M setting and the potential of cyber-physical measures.

*Secure Device Pairing:* SDP has been an active research field ever since it was put forward by Stajano and Anderson in 1999 (c.f. [25]). Moreover, SDP schemes have been widely adopted for IoT and personal devices. Mirzadeh et al. [26] review device pairing protocols and their security, including group device pairing. Fomichev et al. [6] provide terminology and foundations for a classification and comparison of existing SDP schemes, and a comprehensive survey thereof. SDP has been less investigated and put to practice in the context of pairing up large machines of different security domains. We have used here SDP in a hybrid form, employing an AKE protocol. The TAGA channel provides a new type of OoB channel, which in our context turned out to be insufficiently secure but which provides a second line of defence.

## VII. Conclusions and Future Work

In this work we have addressed the gap of how to engineer and validate key establishment methods in safety-critical M2M settings. We have put forward to work with connection compromise states, which define how key establishment can fail and allow for a more fine-grained integration with cyber-physical mitigation measures. We have also seen that in our setting there is a range of measures available that work well, in particular when there are complex trust assumptions due to participants coming from different security realms. Our examples have shown that the consequences of threats against key establishment such as LTKCs can be subtle and unexpectedly large in M2M settings.

Concerning security verification the protocols in use should undergo the rigorous verification process of the cryptographic protocol community. The attacker model can be adapted so that

it suits the channel used, which in turn has to be rigorously investigated. Here we have reverted to the standard Dolev-Yao model, who has full control of the network. However, in a parallel work we investigate how this can be slightly relaxed for the TAGA channel. We also explore a solution, which will be fully local and not depend on global key management [8]. We will also explore post-quantum security for our setting. Another important point for future work is to find methods that allow us to assess the likelihood of LTKCs and other threats against key establishment in a systematic way.

Moreover, this paper has demonstrated that simulation plays an important role in the process to develop and validate a key establishment method for security and safety. Of course, the activities described here can be followed by bench/live tests, and formal verification where necessary. In particular, we wish to investigate whether and how statistical model-checking [27] can be made use of in the tool-chain: to be able to verify integrated safety and security properties such as: "Safety mitigation kicks in before attack causes harm with probability $>$ P".

## References


[1] S. Fröschle and M. Kubisch, "Security process for adopting machine to machine communication for maintenance in transportation with a focus on key establishment," in *Proceedings of SIMUL 2023: The Fifteenth International Conference on Advances in System Simulation*. Thinkmind Digital Library, 2023, pp. 50 – 58.

[2] S. B. Fröschle, M. Kubisch, and M. Gräfing, "Security analysis and design for TAGA: a touch and go assistant in the aerospace domain," *CoRR*, vol. abs/2004.02516, 2020. [Online]. Available: https://arxiv.org/abs/2004.02516

[3] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, 3rd ed. CRC Press, 2021.

[4] G. Lowe, "A hierarchy of authentication specification," in *10th Computer Security Foundations Workshop (CSFW '97), June 10-12, 1997*. IEEE Computer Society, 1997, pp. 31–44.

[5] C. Boyd, A. Mathuria, and D. Stebila, *Protocols for Authentication and Key Establishment*, 2nd ed. Springer Publishing Company, 2020.

[6] M. Fomichev, F. Álvarez, D. Steinmetzer, P. Gardner-Stephen, and M. Hollick, "Survey and systematization of secure device pairing," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 517–550, 2018.

[7] B. Schmidt, S. Meier, C. Cremers, and D. Basin, "Automated analysis of Diffie-Hellman protocols and advanced security properties," in *25th IEEE Computer Security Foundations Symposium, CSF 2012*. IEEE, 2012, pp. 78–94.

[8] S. Fröschle and M. Kubisch, "Three taps for secure machine-to-machine communication: Towards high assurance yet fully local machine pairing," in *Proceedings of the Sixth Workshop on CPS&IoT Security and Privacy*, ser. CPSIoTSec'24. New York, NY, USA: Association for Computing Machinery, 2024, p. 125–133. [Online]. Available: https://doi.org/10.1145/3690134.3694824

[9] A. P. Sarr, P. Elbaz-Vincent, and J.-C. Bajard, "A secure and efficient authenticated Diffie–Hellman protocol," in *Public Key Infrastructures, Services and Applications*. Springer, 2010, pp. 83–98.

[10] A. P. Sarr and P. Elbaz-Vincent, "On the security of the (F)HMQV protocol," in *Progress in Cryptology – AFRICACRYPT 2016*. Springer International Publishing, 2016, pp. 207–224.

[11] S. Blake-Wilson, D. Johnson, and A. Menezes, "Key agreement protocols and their security analysis," in *Crytography and Coding*. Springer, 1997, pp. 30–45.

[12] S. Blake-Wilson and A. Menezes, "Authenticated Diffie-Hellman key agreement protocols," in *Proceedings of the Selected Areas in Cryptography*, ser. SAC '98. Springer-Verlag, 1999, pp. 339–361.

[13] J. Giraldo, D. Urbina, A. Cardenas, J. Valente, M. Faisal, J. Ruths, N. O. Tippenhauer, H. Sandberg, and R. Candell, "A survey of physics-based attack detection in cyber-physical systems," *ACM Comput. Surv.*, vol. 51, no. 4, jul 2018. [Online]. Available: https://doi.org/10.1145/3203245

[14] D. I. Urbina, J. A. Giraldo, A. A. Cardenas, N. O. Tippenhauer, J. Valente, M. Faisal, J. Ruths, R. Candell, and H. Sandberg, "Limiting the impact of stealthy attacks on industrial control systems," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 1092–1105. [Online]. Available: https://doi.org/10.1145/2976749.2978388

[15] R. A., D. Ward, W. B., I. S., R. Y., M. Friedewald, T. Leimbach, A. Fuchs, S. Gürgens, O. Henniger, R. Rieke, M. Ritscher, J. Broberg, L. Apvrille, R. Pacalet, and G. Pedroza, "Security requirements for automotive on-board networks based on dark-side scenarios," 2009, EVITA Project, Deliverable D2.3, v.1.1.

[16] M. M. Islam, A. Lautenbach, C. Sandberg, and T. Olovsson, "A risk assessment framework for automotive embedded systems," in *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*. ACM, 2016, pp. 3–14.

[17] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner, "Sahara: A security-aware hazard and risk analysis method," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2015, pp. 621–624.

[18] A. Shostack, "Experiences threat modeling at microsoft," in *Workshop on Modeling Security*, Toulouse, September 2008.

[19] G. Macher, C. Schmittner, O. Veledar, and E. Brenner, "ISO/SAE DIS 21434 automotive cybersecurity standard - in a nutshell," in *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*, A. Casimiro, F. Ortmeier, E. Schoitsch, F. Bitsch, and P. Ferreira, Eds. Cham: Springer International Publishing, 2020, pp. 123–135.

[20] C. Schmittner, B. Schrammel, and S. König, "Asset driven ISO/SAE 21434 compliant automotive cybersecurity analysis with threatget," in *Systems, Software and Services Process Improvement*, M. Yilmaz, P. Clarke, R. Messnarz, and M. Reiner, Eds. Cham: Springer International Publishing, 2021, pp. 548–563.

[21] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[22] D. Basin, C. Cremers, J. Dreier, and R. Sasse, "Tamarin: verification of large-scale, real world, cryptographic protocols," *IEEE Security and Privacy Magazine*, 2022. [Online]. Available: https://hal.science/hal-03586826

[23] B. Blanchet, "Modeling and verifying security protocols with the applied pi calculus and ProVerif," *Foundations and Trends in Privacy and Security*, vol. 1, no. 1–2, pp. 1–135, Oct. 2016.

[24] "The PQXDH key agreement protocol, revision 3," 2024, called 31/03/2024. [Online]. Available: https://signal.org/docs/specifications/pqxdh/

[25] M. Li, W. Lou, and K. Ren, "Secure device pairing," in *Encyclopedia of Cryptography and Security*. Springer US, 2011, pp. 1111–1115.

[26] S. Mirzadeh, H. Cruickshank, and R. Tafazolli, "Secure device pairing: a survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 17–40, 2014.

[27] E. M. Clarke and P. Zuliani, "Statistical model checking for cyber-physical systems," in *Automated Technology for Verification and Analysis*, T. Bultan and P.-A. Hsiung, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1–12.