Proxy Signcryption Scheme for Vehicle Infrastructure Immune to Randomness Leakage and Setup Attacks

Łukasz Krzywiecki
Department of Computer Science
Wrocław University of Technology
Wrocław, Poland
lukasz.krzywiecki@pwr.edu.pl

Hannes Salin
Swedish Transport Administration
Borlänge, Sweden
hannes.salin@trafikverket.se

Nisha Panwar
School of Computer and Cyber Sciences
Augusta University
Augusta, USA
npanwar@augusta.edu

Mykola Pavlov
Department of Computer Science
Wrocław University of Technology
Wrocław, Poland
248441@pwr.edu.pl

***Abstract*—We propose a proxy signcryption scheme for a multi-party setting, resistant to randomness leakage and setup attacks. Our scheme is an alternative to typical constructions, based on a double Schnorr signature approach, where the linear combination of long term secrets and ephemeral random values occurs both at the initiator and proxy nodes. Our scheme is provably secure in a new stronger model, where the adversary can control the randomness of both parties. Moreover, our proposition is well suited for networks of many independent and moving nodes; especially modern railway infrastructure and vehicle-to-vehicle/infrastructure (V2X) environments, where a broad range of devices with potentially weak computational power and inadequate randomness, is used. Early benchmarks and performance analysis from our proof of concept implementation, suggest that nodes, which use regular Schnorr based schemes, could be successfully upgraded to our more secure alternative construction. Collected timings are still at the acceptable level, proving the applicability of our scheme in modern railway and V2X environments.**

***Index Terms*—signcryption, cryptography, IoT, railway, vehicle**

I. Introduction

Modern railway- and road infrastructure is on its way to be connected, not only vehicles and trains but even tunnels, junctions and small roads can be part of inter-connected and smart networks. Several standardization initiatives and international cooperation in the field exists such as the European Rail Traffic Management System (ERTMS) [1], ETSI's Intelligent Transportation System (ITS) standardization [2], Shift2Rail [3] and the United States Department of Transportation initiative for ITS [4]. An increased number of connected nodes also implies collecting more qualitative data which in turn enables the usage of advanced analytics and machine learning technology in order to optimize reliability and safety goals. Security requirements for such modernized Internet of Things (IoT) eco-systems with many nodes and possibly overlapping ad-hoc networks, is high. In Fig. 1 we describe a high-level view of communication scenarios where v_i represents moving vehicles that can communicate with roadside units, and $\mathcal{T}_i$ representing moving trains communicating with railtrackside units such as balises, cameras and other sensor devices, illustrating how the smart infrastructure connects. Preferably all connecting entities and generated data should be authenticated and signed. Also, since infrastructure may require scalability, i.e. sending data packets through several intermediate nodes, such as data collecting servers and road side units, the need for secure proxy solutions is crucial. Generated data must be encrypted and signed before transferred through any intermediate node and the end recipient server must be able to verify and decrypt the data to secure that the original sending device is trusted and intact. Typically, cryptographical protocols provide message confidentiality and integrity by sequentially applying a signature scheme followed by an encryption scheme, known as *sign-then-encrypt*. In 1997, a new primitive was proposed, called *signcryption* [5] which could perform both a signature and an encryption in one logical step, thus reducing computational and communication complexity significantly. Mambo et al [6] introduced the notion of *proxy signatures*, i.e. a signing party can delegate the signature generation to an authorized proxy, which in turn publishes the signature for anyone to verify. Now, a *proxy signcryption* scheme is the combination of delegating the signcryption cryptogram to a proxy, for which the recipient can both decrypt and verify (called unsigncrypt) that the original signing party is the originator of the cryptogram.

978-1-7281-8326-8/20/$31.00 ©2020 IEEE

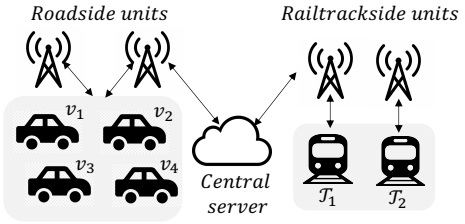

Fig. 1: Connected vehicle and railway scenario

A. Railway infrastructure

Railway infrastructure includes measurement devices and camera systems for both safety and security reasons, collecting data for speed limits, on train positions, gradients, track maintenance status and more. Much of measurement data can be classified as sensitive and is generated in possibly insecure and untrusted systems, e.g. railway-specific devices such as balises, Lineside Electronic Units (LEU), Radio In-fill Units (RIU) and Radio Block Centres (RBC). These initiator type of devices sends data to both trains and/or track side communication devices, acting as proxies before getting collected and verified in Trackside Command Centers (TCC) or geographically distant Traffic Management Systems (TMS). Typical message spaces these initatior devices generates are well-defined numerical values, descriptive strings or pictures.

In order to secure such messages, the train or measurement device should be able to sign and encrypt the data and similarly, any control center must be able to verify that all receiving data packets originate from correct device located at correct geographical location. Since most data passes through an intermediate node which is not necessarily located within a secure network (remote locations throughout the landscape), nor a physically secure installation, an attacker could have full physical access to these proxy devices. Moreover, any cryptographic scheme utilizing randomness may be vulnerable if the measurement device lacks proper randomness generation.

Assuming a wide range of different types of measurement devices, we suspect some of these would lack sufficient randomness generation, low bandwidth power and low computational capabilities. Considering the multi-party setup connecting an initiating measurement device with a proxy server and verification control center, together with the plausible scenario of poor randomness, we provide new perspectives to consider for the standardization work of ERTMS and related initiaitves such as Shift2Rail. Even more, if Schnorr-based schemes is to be used for signature signcryption, we then provide a stronger model and scheme resistant to ephemeral key leakage and setup attacks.

B. Vehical infrastructure

Advances in vehicle infrastructure [7]–[14] allow integrated communication models though space embedded sensors (e.g., RFID chips, Bluetooth beacons, surveillance cameras, WiFi connectivity data etc) as well as the in-situ sensors (e.g., electronic communication units embedded inside the smart hybrid vehicles). The prime motivation is to observe the moving vehicles for subsequent actions and planning for traffic maneuvering, and, to enable vehicle to vehicle connectivity through wireless communication channel. These connected vehicle frameworks can be broadly categorized into V2V (Vehicle to Vehicle), V2I (Vehicle to Infrastructure such as with road side units or any agent for the cloud assisted services), and, the internal vehicle networks from an overall communication and security perspective. However, since the vehicle communication model is more frequently used for exchanging warning messages the reliability or integrity of these messages is highly important for the vehicle safety and the overall connected vehicle infrastructure. The integrity of delay-sensitive communication sessions with other vehicles (V2V), RSUs and clouds (V2I), or Electronic Control Units (internal ECU network) using wireless channel is a non-trivial task since verifiability incurs additional overhead to these time-sensitive messages. We envision the integrity of these messages for smart vehicle notification for scenarios such as parking space locator-, toll-collector-, and speed monitoring systems, etc. In this case, the road-side infrastructure would act as a service agent that notify the vehicles about any kind of control information, e.g., available parking space, suggesting-diversion to a congested-route, overspeed-warning, revoked plate number warning, etc. In general, the vehicle infrastructure model is trust-asymmetrical in which vehicles trust RSU generated messages but RSU does not trust vehicle generated messages. Now, in some scenarios RSU might not cover the entire physical space in which case vehicles or relay-agents might be required to forward the control information that was originally generated at trusted RSU. In this case, we must guarantee verification of relayed messages for which we need signcryption type properties. Therefore, we propose an end-to-end integrity preserving messaging system for vehicle notification scenarios.

C. Problem statement and motivation

Ephemeral keys are used in many cryptographical schemes; the Diffie-Hellman protocol might be one of the more well-known in which a private, temporary session key is generated using a source of randomness. This key is used for one particular session and then discarded, hence allowing for forward secrecy, i.e. if the long-term key is compromised it is still impossible to decrypt recorded sessions. A key factor for such schemes is the usage of proper randomness from which the scheme harvests random bits, i.e. from a uniform distribution, when generating the ephemeral key. In practice it is not uncommon to utilize pseudorandom sources which, if implemented badly, will generate predictable sequences of bits which allows an attacker to break the scheme. Even more critically, if an adversary has the ability to set the randomness of its own choice, the setup is vulnerable to ephemeral key leakage; an important aspect already researched in [15]–[17] for protocols like identification and authenticated key agreement.

Now, a typical scenario to consider is a measurement device such as a balise, camera or RSU, in this context viewed as the initiating party. Data packets are sent continuously to an intermediate node - the proxy - which also gets a delegation from the initiator on which type of messages should be signed, encrypted and forwarded to the verifier, in this case the TMS or cloud server. The verifier is then able to unsigncrypt. Without a suitable proxy signcryption scheme for this scenario, either the data must be signed and encrypted at the initiator's device which may not be possible due to computational power, or it is handled in the proxy which then could sign and encrypt arbitrary data of its own choice. In the paper we consider specific constructions based on Schnorr signatures, so called *Schnorr-like*, which uses random values to mask the long-term keys at the initiator and the proxy. If weak or maliciously injected randomness is used in either party, an external attacker would be able to completely forge signcrypted data. Therefore, we propose a provably secure proxy signcryption scheme resistant to ephemeral key leakage. This would not only increases the security and integrity of measurement data which needs to flow through a proxy server, but also improve the computational efficiency suitable for low performing devices.

D. Contribution

Our scheme is the first construction, among the three party protocols based on Schnorr identification/signature, which withstands setting attacks on randomness independently from both initiator and proxy device. Another construction of this kind withstands attacks only on proxy party randomness, assuming that initial party procedures are done in a controlled, secure environment [18]–[20]. Here, in our scheme, we also strengthen the security for the initiator party, thus making it more suitable for uncontrolled spread environment, like railway and road/vehicle infrastructure with many moving nodes.

We stress that any proxy signature, or proxy signcryption scheme based on the regular Schnorr construction is vulnerable to attacks where the attacker controls the randomness, e.g. [21]–[24] or [25] which suffers from the ephemeral leakage and setup attacks.

The contribution of this paper is the following:

- We introduce a new security model for proxy signcryption schemes in which ephemeral keys may be leaked or set by an adversary at the initiator and the proxy side
- We show that an example typical scheme [25] is not secure in the proposed security model.
- We propose a new proxy signcryption scheme which is resistant to ephemeral setup, both at the initiator and the proxy side.
- We present a security proof of our scheme in the proposed strong security model.
- We provide the complexity and the timing assessment comparison between an example of a typical construction [25] and our proposed scheme.

E. Previous work

Research and implementation of Schnorr based signcryption schemes has been conducted, e.g. for mobile communication systems [23], [24] but without considering a proxy setup. Moreover, for ephemeral key leakage resistant schemes, especially in the context of connected vehicle and IoT infrastructure, research has been conducted and proposed in [18]–[20]. However, these schemes are not proxy nor signcryption types. A thorough analysis of proxy signcryption schemes can be found in [26].

F. Structure of the paper

The rest of the paper is structured as follows: section II formalizes the system settings and threat description, section III describes our stronger formal model, section IV describes the improved scheme and associated security proofs, whereas section V and VI shows the results from proof of concept implementation, and conclusions respectively.

II. System Settings

We consider a multi-party protocol for integrity-preserving message-relaying among distant nodes. In particular, there are three types of nodes: initiator node ($\mathcal{I}$), proxy node ($\mathcal{P}$), and, receiver node ($\mathcal{R}$). All messages disseminate through the network. Also, note that the message space Ω strictly defines which type and structure of messages to be generated. During the message dissemination the recipient nodes can process these messages and utilize the information for assisted driving and safety features. However, since the wireless connectivity is prone to fluctuations, proxy node $\mathcal{P}$ act as a temporary connector between $\mathcal{I}$ and $\mathcal{R}$, and sends fresh messages from the space Ω defined by $\mathcal{I}$. Clearly, such an indirect connectivity requires the recipient node to verify the origin of messages. In particular, whenever a recipient node decides to verify the message signature it must execute the verification protocol. The verification protocol is detailed in Section III-A.

A. Threat description

The proposed signcryption for integrity-preserving message relaying is based on a three party model. For instance, during the compromised execution of the message-relaying protocol a recipient must verify the messages before using it for driving services. A compromised message-relaying protocol means that a message m, that originated at the initiator node $\mathcal{I}$, is tampered before arriving as m' at the recipient node. A compromised session can be distinguished through the verification protocol. There are three long term cryptographic secret keys, i.e. the secret key of initiator ($\mathtt{sk}_{\mathcal{I}}$), the secret key of proxy ($\mathtt{sk}_{\mathcal{P}}$) and the secret key of the receiver ($\mathtt{sk}_{\mathcal{R}}$). Moreover, the random keys of the initiator and the proxy are r_1 and r_2 respectively. The verification protocol proposed in this paper can sustain the leakage of these cryptographic primitives. Our proposed scheme addresses two attack variants during a compromised message-relaying session from $\mathcal{I}$ through $\mathcal{P}$:

- *Attack 1:* The vulnerability of the initiator device and the leakage of randomness r_1. In this scenario, the adversary, such as a malicious producer of that device, mounts an attack on the long term secret $\mathtt{sk}_{\mathcal{I}}$ with the knowledge of r_1, impersonating $\mathcal{I}$ and forging an arbitrary message space Ω^* to the proxy.
- *Attack 2:* The vulnerability of the proxy device and the leakage of randomness r_2. In this scenario, the adversary, such as a malicious producer of the proxy device, mounts an attack on the long term secret $\mathtt{sk}_{\mathcal{P}}$ with the knowledge of r_2, impersonating $\mathcal{P}$ since then, and forging any message m^* from the legitimate space Ω to the receiver.

III. Formal security model

In this paper we use notation similar to the one that can be found in [19]. Let X be a finite set. We use notation $x_1, \ldots, x_n \leftarrow_{\$} X$ to specify that each x_i is selected uniformly at random from X. Let λ be a security parameter. We denote negligible values as ϵ. A function $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{A}$ is a secure hashing function that transforms any binary string into an element of $\mathbb{A}$.

1) Bilinear Map: Let $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ be groups with generators g_1, g_2, g_T respectively. Let $q = |\langle g_1 \rangle| = |\langle g_2 \rangle| = |\langle g_T \rangle|$. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a pairing function with the following properties:

1) *Bilinearity*: $\forall (a, b \in \mathbb{Z}_q, g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2)$: $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$,
2) *Computability*: Computing $\hat{e}$ is efficient,
3) *Non-degeneracy*: $\exists (g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2) : \hat{e}(g_1, g_2) \neq 1$.

If $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$, pairing is symmetric. Function $Gen(1^\lambda)$ returns a tuple $(\mathbb{G}, g, q)$ such that $\mathbb{G}$ is a group of prime order q and g is its generator. Function $Gen_{BP}(1^\lambda)$ returns a symmetric bilinear mapping tuple: $(\mathbb{G}, \mathbb{G}_T, g, g_T, q, \hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T)$.

2) Gap computational Diffie-Hellman: Let $\mathcal{O}_{dh}$ denote an oracle for decisional Diffie-Hellman problem, i.e. given a tuple (g, g^x, g^y, g^z), where g is a generator of cyclic group $\mathbb{G}$ of order q and $g^x, g^y, g^z \in \mathbb{G}$ are random, it outputs 1 if $z = xy$ and 0 otherwise. Having access to $\mathcal{O}_{dh}$, given a tuple (g, g^x, g^y), compute g^{xy}.

Gap Computational Diffie-Hellman is denoted as GDH. We assume that it is hard in groups that we use throughout this paper, i.e. the probability that any probabilistic polynomial time algorithm (PPT) solves the problem is negligible.

A. Definitions

Definition 1. *A proxy signcryption scheme* - PSC - *is a system which consists of six algorithms:* ParGen, KeyGen, Id, Ver, SC, USC.

$\mathsf{ParGen}(\lambda) \rightarrow \mathsf{par}$**:** *The setup algorithm takes as input a security parameter* 1^k *and outputs any common parameters* par *required by the signcryption schemes. This include the description of a group* $\mathbb{G}$*, generators for that group, choices for hash functions* $\mathcal{H}$*, and an encryption scheme* $\mathcal{E}$*.*

$\mathsf{KeyGen}(\mathsf{par}, u) \rightarrow (\mathtt{sk}_u, \mathtt{pk}_u)$**:** *The key generation algorithm takes as input the common parameters* par*, the user identity* u *and outputs a pair of corresponding secret and public keys.*

$\mathsf{Id}(\mathsf{par}, \mathtt{sk}_{\mathcal{I}}, \Omega) \rightarrow \sigma_1$**:** *The identity algorithm takes as input the parameters* par*, the initiator private key* $\mathtt{sk}_{\mathcal{I}}$*, the message space* Ω*. It outputs the warrant* σ_1*.*

$\mathsf{Ver}(\mathsf{par}, \mathtt{pk}_{\mathcal{I}}, \sigma_1, \Omega) \rightarrow 1/0$**:** *The warrant verification algorithm takes as input the parameters* par*, the initiator public key* $\mathtt{pk}_{\mathcal{I}}$*, the warrant* σ*, and the message space* Ω*. It outputs* 1 *or* 0 *to indicate the acceptance or the rejection of the warrant.*

$\mathsf{SC}(\mathsf{par}, \mathtt{sk}_{\mathcal{P}}, \sigma_1, \Omega, \mathtt{pk}_{\mathcal{R}}, m) \rightarrow \sigma_2$**:** *the proxy signcrypt algorithm takes as input the common parameters* par*, the proxy secret key* $\mathtt{sk}_{\mathcal{P}}$*, the warrant* σ_1 *for the message space* Ω*, the receiver public key* $\mathtt{pk}_{\mathcal{R}}$*, and message* m *from space* Ω*. It outputs signcrypted message* σ_2*.*

$\mathsf{USC}(\mathsf{par}, \mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \sigma_2, \Omega) \rightarrow m/\perp$**:** *The unsign and decrypt algorithm takes as input the parameters* par*, the receiver secret key* $\mathtt{sk}_{\mathcal{R}}$*, the public key of the initiator* $\mathtt{pk}_{\mathcal{I}}$*, the public key of the proxy* $\mathtt{pk}_{\mathcal{P}}$*, the signcrypted message* σ_2*, the message space* Ω*. It outputs the decrypted message* m *or an error symbol* $\perp$*, which indicates that the plaintext cannot be restored or the restored message* m *has the wrong signature.*

Definition 2 (Correctness)**.** *The* PSC *scheme is correct iff for any* $m \in \Omega$*, and any parties* $\mathcal{I}, \mathcal{P}, \mathcal{R}$*:*

$$\begin{aligned}\mathtt{Pr}[\mathsf{USC}(\mathsf{par}, \mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \sigma_2, \Omega) = m \mid \\ \mathsf{par} \leftarrow \mathsf{ParGen}(\lambda), \\ (\mathtt{sk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{I}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{I}), \\ (\mathtt{sk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{P}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{P}), \\ (\mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{R}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{R}), \\ \sigma_1 \leftarrow \mathsf{Id}(\mathsf{par}, \mathtt{sk}_{\mathcal{I}}, \Omega), 1 \leftarrow \mathsf{Ver}(\mathsf{par}, \mathtt{pk}_{\mathcal{I}}, \sigma_1, \Omega), \\ \sigma_2 \leftarrow \mathsf{SC}(\mathsf{par}, \mathtt{sk}_{\mathcal{P}}, \sigma_1, \Omega, \mathtt{pk}_{\mathcal{R}}, m)] = 1.\end{aligned}$$

Definition 3 (Indistinguishability under adaptive chosen ciphertext attack)**.** *Let* $\mathsf{PSC} = (\mathsf{ParGen}, \mathsf{KeyGen}, \mathsf{Id}, \mathsf{Ver}, \mathsf{SC}, \mathsf{USC})$ *be a proxy signcryption scheme. We define the following security experiment:*

$\mathtt{Init\ stage}$: *Challenger generates common parameters* $\mathsf{par} \leftarrow \mathsf{ParGen}(\lambda)$ *and the keys* $(\mathtt{sk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{I}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{I})$*,* $(\mathtt{sk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{P}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{P})$*,* $(\mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{R}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{R})$*.*

$\mathtt{Query\ stage}$: *Adversary* $\mathcal{A}$ *runs on input* $(\mathsf{param}, \mathsf{y_i}, \mathsf{y_p}, \mathsf{y_r})$*.* $\mathcal{A}$ *may query a signcryption oracle with messages of its choice* $m \in \Omega$ *to receive the signcryption* $\sigma_2 \leftarrow \mathcal{O}_{\mathsf{SC}}(\mathsf{par}, m, \Omega, \mathcal{I}, \mathcal{P}, \mathcal{R})$ *dedicated for the receiver* $\mathcal{R}$*, from the proxy* $\mathcal{P}$ *with the warrant from initiator* $\mathcal{I}$*.* $\mathcal{A}$ *may also query an unsigncryption oracle with a signcrypted messages* σ_2 *of its choice, to receive the message* $m \leftarrow \mathcal{O}_{\mathsf{USC}}(\mathsf{par}, \sigma_2, \mathcal{I}, \mathcal{P}, \mathcal{R})$ *if the message was successfully decrypted with* $\mathtt{sk}_{\mathcal{R}}$ *and verified against* $\mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}$*, or* $\perp$ *otherwise.*

`Challenge stage` : $\mathcal{A}$ *outputs two equal-length messages* $m_1,\ m_2 \in M$. *The challenger chooses* $b \in \{0,1\}$ *at random and computes the challenge signcrypted message tuple* $\sigma_{2,b} \leftarrow \mathcal{O}_{\mathsf{SC}}(\mathsf{par}, m_b, \Omega, \mathcal{I}, \mathcal{P}, \mathcal{R})$.

`Response stage` : $\mathcal{A}$ *inputs the challenge signcrypted message tuple* $\sigma_{2,b}$. $\mathcal{A}$ *may query the signcryption and unsigncryption oracles as before, with the exception that it is forbidden to submit the ciphertext* $\sigma_{2,b}$ *to the unsigncryption oracle.* $\mathcal{A}$ *ends with the output of a bit* b'.

Adversary wins if $b' = b$, *the advantage is defined to be* $\varepsilon = |Pr[b' = b] - 1/2|$.

B. New stronger unforgeability model

To cover the scenario where ephemeral secrets are known by the adversary, we propose a new, stronger unforgeability model for signcryption schemes, based on similar but weaker models for three party protocols [18], [19]. We assume that the adversary $\mathcal{A}$ has all the public information. We have two scenarios. In the first one, $\mathcal{A}$ possesses additionally the secret key of a proxy $\mathcal{P}$ and sets the randomness of the initiator $\mathcal{I}$ each time it queries the oracle of the initiator. In the second scenario $\mathcal{A}$ has the secret of initiator $\mathcal{I}$ and sets the randomness of proxy $\mathcal{P}$ each time the proxy oracle is invoked. In our definition the scheme is secure if $\mathcal{A}$ cannot impersonate $\mathcal{I}$ in the first case, nor it impersonates $\mathcal{P}$ in the second case.

Definition 4 (Unforgeability under Impersonation Attacks)**.** *Let* $\mathsf{PSC} = (\mathsf{ParGen}, \mathsf{KeyGen}, \mathsf{Id}, \mathsf{Ver}, \mathsf{SC}, \mathsf{USC})$ *be a proxy signcryption scheme. We define the following security experiment:*

`Init` : $\mathsf{par} \leftarrow \mathsf{ParGen}(\lambda)$, $(\mathtt{sk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{I}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{I})$, $(\mathtt{sk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{P}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{P})$, $(\mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{R}}) \leftarrow \mathsf{KeyGen}(\mathsf{par}, \mathcal{R})$.

Id `Oracle` : *The oracle* $\mathcal{O}^{\bar{r}}_{\mathsf{Id}}$ *accepts parameters* par, *a message space* Ω, *randomness* $\bar{r}$, *and outputs corresponding valid warrant* σ_1 *generated with the* $\bar{r}$ *on behalf of the initiator* $\mathcal{I}$, *i.e.* $\mathcal{O}^{\bar{r}}_{\mathsf{Id}}(\Omega) \rightarrow \sigma_1$, *such that* $\mathsf{Ver}(\mathsf{par}, \sigma, \mathtt{pk}_{\mathcal{I}}, \Omega) = 1$. *The oracle models the device of the initiator in which the warrants are generated with injected ephemerals controlled externally by the adversary.*

SC `Oracle` : *The oracle* $\mathcal{O}^{\bar{r}}_{\mathsf{SC}}$ *accepts the ephemeral* $\bar{r}$, *parameters* par, *a valid warrant* σ_1 *over the space* Ω, *the public key* $\mathtt{pk}_{\mathcal{R}}$ *of the reciepient* $\mathcal{R}$, *and a message* m. *It outputs a valid signcryption* σ_2 *generated with* $\bar{r}$ *on behalf of the proxy* $\mathcal{P}$, *i.e.* $\mathcal{O}^{\bar{r}}_{\mathsf{SC}}(\mathsf{par}, \sigma_1, \Omega, \mathtt{pk}_{\mathcal{R}}, m) \rightarrow \sigma_2$, *such that* $\mathsf{USC}(\mathsf{par}, \mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \sigma_2, \Omega) = m$. *The oracle models the device of the proxy in which the messages are generated with injected ephemerals controlled externally by the adversary.*

FT1 (`Forgery Type I`) : *The adversary generates a tuple:* $(\Omega^*, \sigma_2^*) \leftarrow \mathcal{A}_{\mathtt{I}}^{\mathcal{O}^{\bar{r}}_{\mathsf{Id}}}(\mathsf{par}, \mathtt{sk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{R}})$, *such that* $\mathsf{USC}(\mathsf{par}, \mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \sigma_2^*, \Omega^*) = m^*$ *and* $m^* \in \Omega^*$, *and* Ω^* *was not previously queried to* $\mathcal{O}^{\bar{r}}_{\mathsf{Id}}$. *The attack reflects the scenario in which the adversary totally controls the proxy (posseses the secret key* $\mathtt{sk}_{\mathcal{P}}$ *of the proxy* $\mathcal{P}$*), and controls the randomness of the initiator device.*

FT2 (`Forgery Type II`) : *The adversary generates a tuple:* $(\Omega^*, \sigma_2^*) \leftarrow \mathcal{A}_{\mathtt{II}}^{\mathcal{O}^{\bar{r}}_{\mathsf{SC}}}(\mathsf{par}, \mathtt{sk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{R}})$, *such that* $\mathsf{USC}(\mathsf{par}, \mathtt{sk}_{\mathcal{R}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \sigma_2^*, \Omega^*) = m^*$ *and* $m^* \in \Omega^*$, *and* m^* *was not previously queried to* $\mathcal{O}^{\bar{r}}_{\mathsf{SC}}$. *The attack reflects the scenario in which the adversary totally controls the initiator (posseses the secret key* $\mathtt{sk}_{\mathcal{I}}$ *of the initiator* $\mathcal{I}$*), and controls the randomness of the proxy device.*

We say that the scheme is secure if the probability of FT1 *is negligible and the probability of* FT2 *is negligible.*

In Theorem 1 we show that a typical Schnorr based PSC [25] (left col. of Tab. I) is not secure in our model.

Theorem 1. *The scheme presented in left column of Fig. I is not secure in our new strong unforgeability model.*

Proof. FT1: After system is initialized, the attacker $\mathcal{A}_{\mathtt{I}}^{\mathcal{O}^{\bar{r}}_{\mathsf{Id}}}(\mathsf{par}, \mathtt{sk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{R}})$ selects $\bar{r}_1 \leftarrow_{\$} \mathbb{Z}_q$ and queries $\mathcal{O}^{\bar{r}}_{\mathsf{Id}}$ for an arbitrary space Ω, i.e. $(s_1, R_1) = \sigma_1 \leftarrow \mathcal{O}^{\bar{r}}_{\mathsf{Id}}(\Omega)$. It computes $\mathtt{sk}_{\mathcal{I}} = \frac{s_1 - \bar{r}_1}{\mathcal{H}_1(\Omega, R_1)}$. Thus, with the initiator key it can compute the new warrant over a new Ω^* and a valid signcryption over a new message $m^* \in \Omega^*$.
FT2: After system is initialized, the attacker $\mathcal{A}_{\mathtt{II}}^{\mathcal{O}^{\bar{r}}_{\mathsf{SC}}}(\mathsf{par}, \mathtt{sk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{R}})$ selects $\bar{r}_2 \leftarrow_{\$} \mathbb{Z}_q$ and queries $\mathcal{O}^{\bar{r}}_{\mathsf{SC}}(\mathsf{par}, \sigma_1, \Omega, \mathtt{pk}_{\mathcal{R}}, m) \rightarrow \sigma_2 = (s_2, R_1, h_2, c)$. Then it computes: $R_2 = g^{\bar{r}_2}$, $k_2 = \mathcal{H}_3(R_2, 2)$, and $\mathtt{sk}_{\mathcal{P}} = \frac{s_2 - \bar{r}_2}{\mathcal{H}_2(m, k_2)} - s_1$. With the secret key $\mathtt{sk}_{\mathcal{P}}$ of the proxy it can signcrypt any new message m^* for any Ω^*. □

IV. Proposed signcryption scheme

A. Proposed PSC construction

In the right column of Tab. I we propose our stronger proxy signcryption scheme. The construction mimics a typical scheme architecture with a double Schnorr signature approach, as seen in the left column. In that scheme, the first signature computed by Id procedure is performed by $\mathcal{I}$ over the message space Ω. The second signature occurs in the SC procedure, where a proxy $\mathcal{P}$ uses a linear combination of keys $\mathtt{sk}_{\mathcal{P}} + s_1$ as a secret key for the second Schnorr signature. Note that the leakage of randomness r_1 allows the attacker to obtain the long term key $\mathtt{sk}_{\mathcal{I}}$ and produce a warrant over an arbitrary message space Ω^*, hence s_1 is included plain as part of σ_1. Moreover, the leakage of r_2 enables the attacker to get $\mathtt{sk}_{\mathcal{P}} + s_1$ and signcrypt any message m^* from Ω over the given warrant σ_1. We state those vulnerabilities in Theorem 1.

To mitigate these threats, we propose a scheme, set in a group $\mathbb{G}$ with a symmetric pairing function $\hat{e}$. In our proposition the linear equations for s_1 in Id, and s_2 in SC procedures, are shifted into exponents of two group elements: $\hat{S}_1 = \hat{g}_1^{s_1}$ and $\hat{S}_2 = \hat{g}_2^{s_2}$, for two new ad-hoc, and deterministically (via the function $\mathcal{H}_g$) created generators of $\mathbb{G}$: $\hat{g}_1$ and $\hat{g}_2$ respectively. The verification is performed via the bilinear property of the chosen pairing function $\hat{e}$, which we state in the Theorem 2. Here we highlight the advantage

Typical scheme [25]		**Proposed scheme**
ParGen(λ): $(\mathbb{G}, g, q) \leftarrow Gen(1^\lambda)$ $\mathcal{E} = (E, D, K, M)$ $\mathcal{H}_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ $\mathcal{H}_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ $\mathcal{H}_3 : \{0,1\}^* \rightarrow K$ $\mathsf{par} = (\mathbb{G}, g, q, \mathcal{E}, \mathcal{H}_1, \mathcal{H}_2, h_3)$		ParGen(λ): $(\mathbb{G}, \mathbb{G}_T, g, g_T, q, \hat{e}) \leftarrow Gen_{BP}(1^\lambda)$ $\mathcal{E} = (E, D, K, M)$ $\mathcal{H}_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ $\mathcal{H}_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ $\mathcal{H}_3 : \{0,1\}^* \rightarrow K$ $\mathcal{H}_g : \{0,1\}^* \rightarrow \mathbb{G}$ $\mathsf{par} = (\mathbb{G}, \mathbb{G}_T, g, g_T, q, , \mathcal{E}, \hat{e}, \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_g)$
Id ($\mathsf{par}, \mathtt{sk}_\mathcal{I}, \Omega$): $r_1 \leftarrow_\$ \mathbb{Z}_q, R_1 = g^{r_1}$ $s_1 = r_1 + \mathtt{sk}_\mathcal{I} \cdot \mathcal{H}_2(\Omega, R_1)$ $\sigma_1 = (s_1, R_1)$		Id ($\mathsf{par}, \mathtt{sk}_\mathcal{I}, \Omega$): $r_1 \leftarrow_\$ \mathbb{Z}_q, R_1 = g^{r_1}$ $s_1 = r_1 + \mathtt{sk}_\mathcal{I} \cdot \mathcal{H}_1(\Omega, R_1)$ $\hat{g}_1 = \mathcal{H}_g(\Omega, R_1), \hat{S}_1 = \hat{g}_1^{s_1}$ $\sigma_1 = (\hat{S}_1, R_1)$
Ver ($\mathsf{par}, \mathtt{pk}_\mathcal{I}, \sigma_1, \Omega$): $h_1 = \mathcal{H}_1(\Omega, R_1)$ Accept iff $g^{s_1} == R_1 \cdot {\mathtt{pk}_\mathcal{I}}^{h_1}$		Ver ($\mathsf{par}, \mathtt{pk}_\mathcal{I}, \sigma_1, \Omega$): $\hat{g}_1 = \mathcal{H}_g(\Omega, R_1), h_1 = \mathcal{H}_1(\Omega, R_1)$ Accept iff $\hat{e}(\hat{S}_1, g) == \hat{e}(\hat{g}_1, R_1 \cdot {\mathtt{pk}_\mathcal{I}}^{h_1})$
SC ($\mathsf{par}, \mathtt{sk}_\mathcal{P}, \sigma_1, \Omega, \mathtt{pk}_\mathcal{R}, m$): $r_2 \leftarrow_\$ \mathbb{Z}_q, R_2 = g^{r_2}, \hat{R}_2 = {\mathtt{pk}_\mathcal{R}}^{r_2}$ $k_1 = \mathcal{H}_3(\hat{R}_2, 1), k_2 = \mathcal{H}_3(\hat{R}_2, 2)$ $s_2 = r_2 + (\mathtt{sk}_\mathcal{P} + s_1) \cdot \mathcal{H}_2(m, k_2)$ $c = E_{k_1}(m)$ $\sigma_2 = (s_2, R_1, h_2, c)$		SC ($\mathsf{par}, \mathtt{sk}_\mathcal{P}, \sigma_1, \Omega, \mathtt{pk}_\mathcal{R}, m$): $r_2 \leftarrow_\$ \mathbb{Z}_q, R_2 = g^{r_2}, \hat{R}_2 = {\mathtt{pk}_\mathcal{R}}^{r_2}, k = {\mathtt{pk}_\mathcal{R}}^{\mathtt{sk}_\mathcal{P}}$ $k_1 = \mathcal{H}_3(R_2, k, 1), k_2 = \mathcal{H}_3(R_2, 2)$ $\hat{g}_2 = \mathcal{H}_g(\Omega, R_1, k_2, m)$ $s_2 = r_2 + \mathtt{sk}_\mathcal{P} \cdot \mathcal{H}_2(\Omega, R_1, k_2, m)$ $\hat{S}_2 = \hat{g}_2^{s_2}, \hat{S} = \hat{S}_1 \hat{S}_2$ $c = E_{k_1}(m)$ $\sigma_2 = (\hat{S}, R_1, \hat{R}_2, c)$
USC ($\mathsf{par}, \mathtt{sk}_\mathcal{R}, \mathtt{pk}_\mathcal{I}, \mathtt{pk}_\mathcal{P}, \sigma_2, \Omega$): $h_1 = \mathcal{H}_1(\Omega, R_1)$ $\hat{R}_2 = ((g^{s_2})/(\mathtt{pk}_\mathcal{P} \cdot R_1 {\mathtt{pk}_\mathcal{I}}^{h_1})^{h_2})^{\mathtt{sk}_\mathcal{R}}$ $k_1 = \mathcal{H}_3(\hat{R}_2, 1), k_2 = \mathcal{H}_3(\hat{R}_2, 2)$ $m = D_{k_1}(c)$ Accept iff $m \in \Omega$ and $h_2 == \mathcal{H}_2(m, k_2)$		USC ($\mathsf{par}, \mathtt{sk}_\mathcal{R}, \mathtt{pk}_\mathcal{I}, \mathtt{pk}_\mathcal{P}, \sigma_2, \Omega$): $R_2 = \hat{R}_2^{1/\mathtt{sk}_\mathcal{R}}, k = {\mathtt{pk}_\mathcal{P}}^{\mathtt{sk}_\mathcal{R}}, k_1 = \mathcal{H}_3(R_2, k, 1), k_2 = \mathcal{H}_3(R_2, 2)$ $m = D_{k_1}(c)$ $\hat{g}_1 = \mathcal{H}_g(\Omega, R_1), \hat{g}_2 = \mathcal{H}_g(\Omega, R_1, k_2, m)$ $h_1 = \mathcal{H}_1(\Omega, R_1), h_2 = \mathcal{H}_2(\Omega, R_1, k_2, m)$ Accept iff $m \in \Omega$ and $\hat{e}(\hat{S}, g) == \hat{e}(\hat{g}_1, R_1 {\mathtt{pk}_\mathcal{I}}^{h_1}) \hat{e}(\hat{g}_2, R_2 {\mathtt{pk}_\mathcal{P}}^{h_2})$

TABLE I: Schnorr based scheme construction to the left and our proposed, improved construction in the right column.

over other three-party schemes (identity-based signatures and authentication [18]–[20]), where just one new generator is used for the proxy party procedure, mitigating the leakage of the randomness from its device only.

Our scheme also increases the secrecy level over the example scheme [25]. In both schemes, the secrecy depends on the chosen encryption scheme $\mathcal{E}$, and the way the symmetric key k_1 is established between the proxy $\mathcal{P}$, and the receiver $\mathcal{R}$. Note that in the example scheme the secrecy is broken once the ephemeral value r_2 is revealed. To mitigate that we propose the usage of an additional intermediate static Diffie-Hellman key $k = {\mathtt{pk}_\mathcal{R}}^{\mathtt{sk}_\mathcal{P}} = {\mathtt{pk}_\mathcal{P}}^{\mathtt{sk}_\mathcal{R}}$, computable locally and independently by the proxy and the receiver, and included as an additional input to the hash function while computing $k_1 = \mathcal{H}_3(R_2, k, 1)$. Note that this does not require to include any additional data in σ_2. This technique is typical for the authenticated key exchange protocols (thus the thorough secrecy discussion for k_1 is omitted due to space constraints).

B. Correctness of the proposed PSC *scheme*

Theorem 2. *The* PSC *scheme proposed in the right-hand side of Fig. I is correct.*

Proof. Obviously $\hat{R}_2 = {\mathtt{pk}_\mathcal{R}}^{r_2} = g^{\mathtt{sk}_\mathcal{R} r_2}$, therefore $R_2 = g^{r_2} = \hat{R}_2^{1/\mathtt{sk}_\mathcal{R}}$.

We have: $k = {\mathtt{pk}_\mathcal{R}}^{\mathtt{sk}_\mathcal{P}} = {\mathtt{pk}_\mathcal{P}}^{\mathtt{sk}_\mathcal{R}}$, $k_1 = \mathcal{H}_3(R_2, k, 1)$ and $k_2 = \mathcal{H}_3(R_2, 2), m = D_{k_1}(c)$.
For generators $\hat{g}_1 = \mathcal{H}_g(\Omega, R_1)$, $\hat{g}_2 = \mathcal{H}_g(\Omega, R_1, k_2, m)$, and hashes $h_1 = \mathcal{H}_1(\Omega, R_1)$, $h_2 = \mathcal{H}_2(\Omega, R_1, k_2, m)$, it holds:

Ver correctness

$$\begin{aligned}\hat{e}(\hat{S}_1, g) &= \hat{e}(\hat{g}_1^{s_1}, g) = \hat{e}(\hat{g}_1, g^{s_1}) \\ &= \hat{e}(\hat{g}_1, g^{r_1 + \mathtt{sk}_\mathcal{I} h_1}) = \hat{e}(\hat{g}_1, R_1 \cdot {\mathtt{pk}_\mathcal{I}}^{h_1}),\end{aligned}$$

USC correctness

$$\begin{aligned}\hat{e}(\hat{S}, g) &= \hat{e}(\hat{S}_1 \hat{S}_2, g) = \hat{e}(\hat{S}_1, g) \cdot \hat{e}(\hat{S}_2, g) \\ &= \hat{e}(\hat{g}_1^{s_1}, g) \cdot \hat{e}(\hat{g}_2^{s_2}, g) \\ &= \hat{e}(\hat{g}_1, g^{s_1}) \cdot \hat{e}(\hat{g}_2, g^{s_2}) \\ &= \hat{e}(\hat{g}_1, g^{r_1 + \mathtt{sk}_\mathcal{I} h_1}) \cdot \hat{e}(\hat{g}_2, g^{r_2 + \mathtt{sk}_\mathcal{P} h_2}) \\ &= \hat{e}(\hat{g}_1, R_1 \cdot {\mathtt{pk}_\mathcal{I}}^{h_1}) \cdot \hat{e}(\hat{g}_2, R_2 \cdot {\mathtt{pk}_\mathcal{P}}^{h_2}).\end{aligned}$$

□

C. Unforgeability of the new scheme under impersonation attacks

Theorem 3. *The* PSC *scheme proposed in the right-hand side of Fig. I is unforgeable in the sense of Definition 4, i.e. is secure against* `Forgery Type I` *and* `Forgery Type II`.

Sketch of the proof. `Forgery Type I` : Let (g, g^α, g^β) be an instance of GDH problem in $\mathsf{par} = (\mathbb{G}, \mathbb{G}_T, g, g_T, q, \hat{e})$.

We setup the system s.t. $\mathtt{sk}_{\mathcal{P}} \leftarrow_{\$} \mathbb{Z}_q, \mathtt{pk}_{\mathcal{P}} = g^{\mathtt{sk}_{\mathcal{P}}}$, $\mathtt{sk}_{\mathcal{R}} \leftarrow_{\$} \mathbb{Z}_q, \mathtt{pk}_{\mathcal{R}} = g^{\mathtt{sk}_{\mathcal{R}}}$, and $\mathtt{pk}_{\mathcal{I}} = g^{\alpha}$. Thus the unknown $\mathtt{sk}_{\mathcal{I}}$ equals the unknown α. We run the adversary $\mathcal{A}_{\mathrm{I}}^{\mathcal{O}_{\mathsf{Id}}^{\bar{r}}}(\mathsf{par}, \mathtt{sk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{R}})$ the access to $\mathcal{O}_{\mathsf{Id}}^{\bar{r}}$, and $\mathcal{O}_{\mathcal{H}_g}$ oracles.

`Serving` $\mathcal{O}_{\mathcal{H}_g}$ `Oracle` : We allow ℓ fresh inputs to the $\mathcal{O}_{\mathcal{H}_g}$ oracle. We choose the random index $j \leftarrow_{\$} \{1, \ldots, \ell\}$, which denotes the j-th invocation of $\mathcal{O}_{\mathcal{H}_g}$, for which we assume the forgery will happen.

- On i-th, $(i \neq j)$, fresh input Ω, R_1, we compute $d \leftarrow_{\$} \mathbb{Z}_q$, register the value $\hat{g} = \mathcal{H}_g(\Omega, R_1) = g^d$ in the ROM table for $\mathcal{H}_g$. We return the $\hat{g}$ as the output.
- On j-th, fresh input Ω, R_1, we set $\hat{g} = (g^{\beta})$ and register that value in the ROM table for $\mathcal{H}_g$. We return the $\hat{g}$ as the output.

`Serving` $\mathcal{O}_{\mathsf{Id}}^{\bar{r}}$ `Oracle` : On input $\Omega, \bar{r}_1$ we compute $\bar{R}_1 = g^{\bar{r}_1}$, serve the call $\mathcal{O}_{\mathcal{H}_g}(\Omega, \bar{R}_1)$, namely locate and return $(\hat{g}_1 = g^d, d)$ if $\Omega, \bar{R}_1$ was not j-th fresh input to $\mathcal{O}_{\mathcal{H}_g}$, then compute $\hat{S}_1 = (\bar{R}_1 \mathtt{pk}_{\mathcal{I}}{}^{\mathcal{H}_1(\Omega, \bar{R}_1)})^d$, return $\sigma_1 = (\hat{S}_1, \bar{R}_1)$. Note that in this case the $\mathsf{Ver}(\mathsf{par}, \mathtt{pk}_{\mathcal{I}}, \sigma_1, \Omega) = 1$ as $\hat{e}(\hat{S}_1, g) = \hat{e}((\bar{R}_1 \mathtt{pk}_{\mathcal{I}}{}^{\mathcal{H}_1(\Omega, \bar{R}_1)})^d, g) = \hat{e}((g^d)^{\bar{r}_1 + \mathtt{sk}_{\mathcal{I}} \mathcal{H}_1(\Omega, \bar{R}_1)}, g) = \hat{e}(g^d, \bar{R}_1 \cdot \mathtt{pk}_{\mathcal{I}}{}^{h_1}) = \hat{e}(\hat{g}_1, \bar{R}_1 \cdot \mathtt{pk}_{\mathcal{I}}{}^{h_1})$.

`Processing the Forgery` : Under the *Forking Lemma* for the hash $\mathcal{H}_1$ the attacker returns two valid signcryptions with the same randomness R_1: $\sigma_2 = (\hat{S}, R_1, \hat{R}_2, c)$, $\sigma_2' = (\hat{S}', R_1, \hat{R}_2', c')$, s.t. $\hat{S} = \hat{S}_1 \hat{S}_2$, $\hat{S}' = \hat{S}_1' \hat{S}_2'$, and the output $\mathcal{H}_1(\Omega, R_1)$ equals h_1 in $\hat{S}_1$ of σ_2 but equals $h_1' \neq h_1$ in $\hat{S}_1'$ of σ_2'. With the non-negligible probability $1/\ell$, related to j-th fresh input Ω, R_1 to $\mathcal{O}_{\mathcal{H}_g}$, the $\hat{g}_1$ equals to g^{β} in both tuples σ_2, σ_2'. We compute $R_2 = \hat{R}_2^{1/\mathtt{sk}_{\mathcal{R}}}$, $k = \mathtt{pk}_{\mathcal{P}}{}^{\mathtt{sk}_{\mathcal{R}}}$, $k_1 = \mathcal{H}_3(R_2, k, 1)$, $k_2 = \mathcal{H}_3(R_2, 2)$, $m = D_{k_1}(c)$, locate $(\hat{g}_2 = g^d, d)$ in ROM table for $\mathcal{O}_{\mathcal{H}_g}(\Omega, R_1, k_2, m)$, compute $\hat{S}_2 = (R_2 \mathtt{pk}_{\mathcal{P}}{}^{\mathcal{H}_2(\Omega, R_1, k_2, m)})^d$, and eventually $\hat{S}_1 = \hat{S}/\hat{S}_2$. Compute $\hat{S}_1'$ from σ_2' in a similar way. We have $\hat{S}_1/\hat{S}_1' = \hat{g}_1^{\mathtt{sk}_{\mathcal{I}}(h_1 - h_1')} = (g^{\beta})^{\mathtt{sk}_{\mathcal{I}}(h_1 - h_1')}$. Therefore we could compute $g^{\alpha\beta} = (\hat{S}_1/\hat{S}_1')^{(1/(h_1 - h_1'))}$, breaking the given instance of GDH.

`Forgery Type II:` Let $(g, g^{\alpha}, g^{\beta})$ be an instance of GDH problem in $\mathsf{par} = (\mathbb{G}, \mathbb{G}_T, g, g_T, q, \hat{e})$. We setup the system s.t. $\mathtt{sk}_{\mathcal{I}} \leftarrow_{\$} \mathbb{Z}_q, \mathtt{pk}_{\mathcal{I}} = g^{\mathtt{sk}_{\mathcal{I}}}$, $\mathtt{sk}_{\mathcal{R}} \leftarrow_{\$} \mathbb{Z}_q, \mathtt{pk}_{\mathcal{R}} = g^{\mathtt{sk}_{\mathcal{R}}}$, and $\mathtt{pk}_{\mathcal{P}} = g^{\alpha}$. Thus the unknown $\mathtt{sk}_{\mathcal{P}}$ equals the unknown α. We run the adversary $\mathcal{A}_{\mathrm{II}}^{\mathcal{O}_{\mathsf{SC}}^{\bar{r}}}(\mathsf{par}, \mathtt{sk}_{\mathcal{I}}, \mathtt{pk}_{\mathcal{P}}, \mathtt{pk}_{\mathcal{R}})$ the access to $\mathcal{O}_{\mathsf{SC}}^{\bar{r}}$, and $\mathcal{O}_{\mathcal{H}_g}$ oracles.

`Serving` $\mathcal{O}_{\mathcal{H}_g}$ `Oracle` : We allow ℓ fresh inputs to the $\mathcal{O}_{\mathcal{H}_g}$ oracle. We choose the random index $j \leftarrow_{\$} \{1, \ldots, \ell\}$, which denotes the j-th invocation of $\mathcal{O}_{\mathcal{H}_g}$, for which we assume the forgery will happen.

- On i-th, $(i \neq j)$, fresh input (Ω, R_1, k_2, m), we compute $d \leftarrow_{\$} \mathbb{Z}_q$, register the value $\hat{g} = \mathcal{H}_g(\Omega, R_1, k_2, m) = g^d$ in the ROM table for $\mathcal{H}_g$. We return the $\hat{g}$ as the output.
- On j-th, fresh input Ω, R_1, k_2, m, we set $\hat{g} = (g^{\beta})$ and register that value in the ROM table for $\mathcal{H}_g$. We return the $\hat{g}$ as the output.

`Serving` $\mathcal{O}_{\mathsf{SC}}^{\bar{r}}$ `Oracle` : On input $\mathsf{par}, \sigma_1, \Omega, \mathtt{pk}_{\mathcal{R}}, m$, and randomnes $\bar{r}_2$, we first verify σ_1: $\hat{e}(\hat{S}_1, g) = \hat{e}(\mathcal{H}_g(\Omega, \bar{R}_1), \bar{R}_1 \cdot \mathtt{pk}_{\mathcal{I}}{}^{\mathcal{H}_1(\Omega, \bar{R}_1)})$. We compute $\bar{R}_2 = g^{\bar{r}_2}$, $k = \mathtt{pk}_{\mathcal{P}}{}^{\mathtt{sk}_{\mathcal{R}}}$, $k_1 = \mathcal{H}_3(R_2, k, 1)$, $k_2 = \mathcal{H}_3(R_2, 2)$, $c = E_{k_1}(m)$. We serve the call $\mathcal{O}_{\mathcal{H}_g}(\Omega, R_1, k_2, m)$ and consider two cases: if (Ω, R_1, k_2, m) was not j-th fresh input to $\mathcal{O}_{\mathcal{H}_g}$ then locate and return $(\hat{g}_2 = g^d, d)$. Next compute $\hat{S}_2 = (\bar{R}_2 \mathtt{pk}_{\mathcal{P}}{}^{\mathcal{H}_2(\Omega, R_1, k_2, m)})^d$, $\hat{S} = \hat{S}_1 \hat{S}_2$ and return $\sigma_2 = (\hat{S}, R_1, \hat{R}_2, c)$. In this case, as $\hat{e}(\hat{S}_2, g) = \hat{e}((\bar{R}_2 \mathtt{pk}_{\mathcal{P}}{}^{\mathcal{H}_2(\Omega, R_1, k_2, m)})^d, g) = \hat{e}((g^d)^{\bar{r}_2 + \mathtt{sk}_{\mathcal{I}} \mathcal{H}_2(\Omega, R_1, k_2, m)}, g) = \hat{e}(g^d, \bar{R}_2 \cdot \mathtt{pk}_{\mathcal{P}}{}^{h_2}) = \hat{e}(\hat{g}_2, \bar{R}_2 \cdot \mathtt{pk}_{\mathcal{I}}{}^{h_2})$, the verification holds: $\hat{e}(\hat{S}, g) = \hat{e}(\mathcal{H}_g(\Omega, \bar{R}_1), \bar{R}_1 \cdot \mathtt{pk}_{\mathcal{I}}{}^{\mathcal{H}_1(\Omega, \bar{R}_1)}) \hat{e}(\hat{g}_2, \bar{R}_2 \cdot \mathtt{pk}_{\mathcal{I}}{}^{h_2})$. Otherwise, if the call to $\mathcal{O}_{\mathcal{H}_g}$ was the j-th fresh input, we abort. Note that the probability of not aborting, but providing the verifiable signature is non-negligable.

`Processing the Forgery` : Under the *Forking Lemma* for the hash $\mathcal{H}_2$ the attacker returns two valid signcryptions with the same randomness $\hat{R}_2$: $\sigma_2 = (\hat{S}, R_1, \hat{R}_2, c)$, $\sigma_2' = (\hat{S}', R_1', \hat{R}_2, c')$, s.t. $\hat{S} = \hat{S}_1 \hat{S}_2$, $\hat{S}' = \hat{S}_1' \hat{S}_2'$, and the output $\mathcal{H}_2(\Omega, R_1, k_2, m)$ equals h_2 in $\hat{S}_2$ of σ_2 but equals $h_2' \neq h_2$ in $\hat{S}_2'$ of σ_2'. With the non-negligible probability $1/\ell$, related to j-th fresh input (Ω, R_1, k_2, m) to $\mathcal{O}_{\mathcal{H}_g}$, the $\hat{g}_2$ equals to g^{β} in both tuples σ_2, σ_2'. We locate $(\hat{g}_1 = g^d, d)$ in ROM table for $\mathcal{O}_{\mathcal{H}_g}(\Omega, R_1)$, compute $\hat{S}_1 = (R_1 \mathtt{pk}_{\mathcal{I}}{}^{\mathcal{H}_1(\Omega, R_1)})^d$, and eventually $\hat{S}_2 = \hat{S}/\hat{S}_1$. Compute $\hat{S}_2'$ from σ_2' in a similar way. We have $\hat{S}_2/\hat{S}_2' = \hat{g}_2^{\mathtt{sk}_{\mathcal{P}}(h_2 - h_2')} = (g^{\beta})^{\mathtt{sk}_{\mathcal{P}}(h_2 - h_2')}$. Therefore we could compute $g^{\alpha\beta} = (\hat{S}_2/\hat{S}_2')^{(1/(h_2 - h_2'))}$, breaking the given instance of GDH.

□

V. Implementation and performance

Our proposed scheme is based on, and proven secure over symmetric pairings, however a similar scheme utilizing asymmetric pairings would still be secure under similar security analysis (future work). We conclude from benchmarks, run on a proof-of-concept of such a scheme, that a version with asymmetric pairings, would be of interest since the average computational complexity lies within reasonable, practical limits. All testing was performed on a MacBook Pro, with an Intel Core i5 2,7GHz. We used the MCL library [27] with BLS12_381 curve for pairings. As for comparison, the original scheme was implemented entirely in group `G2`, while our modified scheme uses `G1` for exponent hiding.

The substantial operations together with the average timings are listed in Tab. II. We compare their numbers for each procedure. New versions are marked with (*). The bottom row shows the total timings per each procedure, which includes all algebraic operations and encryption/decryption via one-time-pad. These are compliant to the standards e.g. in [28].

		Id	Id*	Ver	Ver*	SC	SC*	USC	USC*
G1:exp	0.110	-	1	-	-	-	1	-	-
G2:exp	0.204	1	1	2	1	2	2	4	3
G1:hashTo	0.353	-	1	-	1	-	1	-	2
Pairing	1.808	-	-	-	2	-	-	-	3
Total Time [ms]		1.019	1.271	1.237	4.956	0.431	1.915	1.450	7.729

TABLE II: Complexity and time assessment (5000 runs).

VI. Conclusion

In this paper we analyze a proxy signcryption scheme, resistant to randomness injection attacks for railway and traffic communication. We introduce a stronger model of security in which we consider potential malicious ephemeral setup at both the initiator and proxy parties, which may occur due to untrusted hardware usage. Therefore, we propose a novel scheme which withstands attacks on such weak devices, where physical tampering is likely to occur, for the initiator and the proxy. This is crucial for all levels of security and passenger safety where trains and vehicles interact with intermediate track- and roadside units, and further to cloud based servers and control centers. Our scheme is therefore resistant to *ephemeral key leakage/setup*, further implying secure to the threat of extracting the static long term secret key - as would not be the case in the original scheme. This is of critical importance since connected railway- and vehicle infrastructure is supposed to provide massive, distributed networking dependent on secure communication in an utterly heterogeneous running environment, but at the cost of signature verification for every message exchange.

Acknowledgment

The research was partially financed from the Fundamental Research Fund nr 8201003902 of the Wrocław University of Science and Technology.

References

[1] European Rail Traffic Management System, "ERTMS." https://ec.europa.eu/transport/modes/rail/ertms, 2020.

[2] European Telecommunications Standards Institute, "ETSI automotive Intelligent Transport Systems." https://www.etsi.org/technologies/automotive-intelligent-transport, 2020.

[3] Shift2Rail, "Shift2Rail." https://shift2rail.org/, 2020.

[4] U.S. Department of Transportation, "The Intelligent Transportations Systems Joint Program Office (ITS JPO)." https://www.its.dot.gov, 2020.

[5] Y. Zheng, "Digital Signcryption or How to Achieve Cost(Signature & Encryption) << Cost(Signature) + Cost(Encryption)," in *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '97, (Berlin, Heidelberg), p. 165?179, Springer-Verlag, 1997.

[6] M. Mambo, K. Usuda, and E. Okamoto, "Proxy Signatures for Delegating Signing Operation," in *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, CCS '96, (New York, NY, USA), p. 48?57, Association for Computing Machinery, 1996.

[7] M. Ferreira, R. Fernandes, H. Conceição, W. Viriyasitavat, and O. K. Tonguz, "Self-organized Traffic Control," in *Proceedings of the Seventh ACM International Workshop on VehiculAr InterNETworking*, pp. 85–90, 2010.

[8] P. Papadimitratos, V. Gligor, and J.-P. Hubaux, "Securing Vehicular Communications - Assumptions, Requirements, and Principles," in *WORKSHOP ON EMBEDDED SECURITY IN CARS*, pp. 5–14, 2006.

[9] "IEEE Standard for Motor Vehicle Event Data Recorders (MVEDRs) Amendment 1: MVEDR Connector Lockout Apparatus (MVEDRCLA)," *IEEE Std 1616a*, pp. 1–17, 2010.

[10] M. Amoozadeh, A. Raghuramu, C. Chuah, D. Ghosal, H. M. Zhang, J. Rowe, and K. N. Levitt, "Security vulnerabilities of connected vehicle streams and their impact on cooperative driving," *IEEE Communications Magazine*, vol. 53, no. 6, pp. 126–132, 2015.

[11] R. A. Uzcategui, A. J. D. Sucre, and G. Acosta-Marum, "Wave: A tutorial," *IEEE Communications Magazine*, vol. 47, no. 5, pp. 126–133, 2009.

[12] E. Hossain, G. Chow, V. C. M. Leung, R. D. McLeod, J. Mišić, V. W. S. Wong, and O. Yang, "Vehicular Telematics over Heterogeneous Wireless Networks: A Survey," *Computer Communications*, vol. 33, no. 7, pp. 775–793, 2010.

[13] H. Hartenstein and L. P. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Communications Magazine*, vol. 46, no. 6, pp. 164–171, 2008.

[14] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[15] M. Bellare, M. Fischlin, S. Goldwasser, and S. Micali, "Identification Protocols Secure Against Reset Attacks," 2000. Extended abstract appeared in proceedings of Eurocrypt 2001. This is the full version. mihir@cs.ucsd.edu 11585 received 28 Apr 2000, last revised 20 Sep 2001.

[16] J. Alwen, Y. Dodis, and D. Wichs, "Leakage-Resilient Public-Key Cryptography in the Bounded-Retrieval Model," in *Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '09, (Berlin, Heidelberg), p. 36?54, Springer-Verlag, 2009.

[17] Q. Tang and L. Chen, "Extended KCI attack against two-party key establishment protocols," *Information processing letters*, vol. 111, no. 15, pp. 744–747, 2011.

[18] L. Krzywiecki, P. Koziel, and N. Panwar, "Signature Based Authentication for Ephemeral Setup Attacks in Vehicular Sensor Networks," in *18th IEEE International Symposium on Network Computing and Applications, NCA 2019, Cambridge, MA, USA, September 26-28, 2019* (A. Gkoulalas-Divanis, M. Marchetti, and D. R. Avresky, eds.), pp. 1–4, IEEE, 2019.

[19] L. Krzywiecki, M. Slowik, and M. Szala, "Identity-Based Signature Scheme Secure in Ephemeral Setup and Leakage Scenarios," in *Information Security Practice and Experience - 15th International Conference, ISPEC 2019, Kuala Lumpur, Malaysia, November 26-28, 2019, Proceedings* (S. Heng and J. López, eds.), vol. 11879 of *Lecture Notes in Computer Science*, pp. 310–324, Springer, 2019.

[20] P. Koziel, L. Krzywiecki, and D. Stygar, "Identity-based Conditional Privacy-Preserving Authentication Scheme Resistant to Malicious Subliminal Setting of Ephemeral Secret," in *Proceedings of the 16th International Joint Conference on e-Business and Telecommunications, ICETE 2019 - Volume 2: SECRYPT, Prague, Czech Republic, July 26-28, 2019* (M. S. Obaidat and P. Samarati, eds.), pp. 492–497, SciTePress, 2019.

[21] J. Malone-Lee, "Signcryption with Non-interactive Non-repudiation," *Designs, Codes and Cryptography*, vol. 37, no. 1, pp. 81–109, 2005.

[22] A. Boldyreva, A. Palacio, and B. Warinschi, "Secure Proxy Signature Schemes for Delegation of Signing Rights," *Journal of Cryptology*, vol. 25, 06 2003.

[23] A. Elshobaky, M. Rasslan, and S. Guirguis, "Implementation of schnorr signcryption algorithm on dsp," vol. 9, pp. 217–230, 01 2015.

[24] M. Al-Zubi and A. A. Abu-Shareha, "Efficient Signcryption Scheme Based on El-Gamal and Schnorr," *Multimedia Tools Appl.*, vol. 78, p. 11091?11104, May 2019.

[25] H. M. Elkamchouchi, E. F. A. Elkhair, and Y. Abouelseoud, "An Efficient Proxy Signcryption Scheme Based on the Discrete Logarithm Problem," *International Journal of Information Technology, Modeling and Computing*, vol. 1, no. 2, p. 719, 2013.

[26] A. W. Dent and Y. Zheng, eds., *Practical Signcryption*. Information Security and Cryptography, Springer, 2010.

[27] S. Mitsunari, ""MCL cryptolibrary"." https://github.com/herumi/mcl, 2019.

[28] NGMN Alliance, "V2X Application Requirements by NGMN Alliance." https://www.ngmn.org/fileadmin/ngmn/content/downloads/Technical/2018/V2X_white_paper_v1_0.pdf, 2018.