Schnorr-Like Identification Scheme Resistant to Malicious Subliminal Setting of Ephemeral Secret

Łukasz Krzywiecki(✉)

Faculty of Fundamental Problems of Technology, Department of Computer Science, Wrocław University of Technology, Wrocław, Poland
lukasz.krzywiecki@pwr.wroc.pl

Abstract. In this paper we propose a modification of the Schnorr *Identification Scheme* (IS), which is immune to malicious subliminal setting of ephemeral secret. We introduce a new strong security model in which, during the *query stage*, we allow the adversary verifier to set random values used on the prover side in the commitment phase. We define the IS scheme to be secure if such a setting will not enable the adversary to impersonate the prover later on. Subsequently we prove the security of the modified Schnorr IS in our strong model. We assume the proposition is important for scenarios in which we do not control the production process of the device on which the scheme is implemented, and where the erroneous pseudo-random number generators make such attacks possible.

Keywords: Identification scheme · Ephemeral secret setting · Ephemeral secret leakage · Deniability · Simulatability

1 Introduction

An identification scheme enables one party - a *prover* - to prove its identity in front of another party - a *verifier*. In many *public key* IS constructions the prover has a long term secret key, and proves its knowledge in such a way, that the verifier, provided with the corresponding public key of the prover, is convince about that fact but gets no information about the prover's secret. Typically the proving protocol consists of three rounds: a *commitment*, a *challenge*, and a *response*. In the *commitment* the prover sends to the verifier a commitment to some random ephemeral value (so called the ephemeral secret). In the *challenge* the verifier sends back to the prover some random unpredictable value. In the *response* the prover send to the verifier the result of some computations, involving the received challenge, and its long term secret key masked by the ephemeral value committed in the first message. The prover is accepted if the response "agrees" with the computation on the verifier side involving the commitment, the challenge, the response and the public key of the prover.

Partially supported by funding from Polish NCN contract number DEC-2013/09/D/ST6/03927.

I. Bica and R. Reyhanitabar (Eds.): SECITC 2016, LNCS 10006, pp. 137–148, 2016.
DOI: 10.1007/978-3-319-47238-6_10

Problem Statement. The problem with this construction arises in systems (protocol implementations), where the ephemeral secrets may be leaked. Usually the masking method in the *response* is such, that the security of the long term key relies on the secrecy of the ephemeral key - e.g. the response value is a linear combination of the challenge, the committed ephemeral secret, and the long term secret. Thus, once the ephemeral secrets is leaked the long term secret is also compromised.

The leakage of the ephemeral secret can be archived by some malicious implementation of the device which is used to perform computation on the prover side. Such a device usually has a *High Secure Memory* module (HSM), where long term secrets are kept securely and accessed (indirectly) only via predefined interfaces. A less secure area is used for scheme program computations, including random numbers sampling. Especially implementations of the pseudo-random number generators are vulnerable to attacks. If the adversary can somehow learn their state, it can also learn random values (and ephemerals) produced by those generators. Sometimes even a subtle subliminal adversarial interference, such as the reset of the internal state and/or randomization source of the prover device, can have influence on the produced values.

Therefore in this paper we want to address this issue, and strengthen the security model for the ephemeral secrets even further. We say that scheme should stay secure even if the adversary injects the malicious ephemeral values of its choice to the device of the prover. When using some subliminal channel this could happen even without the prover knowledge and against its will. In our security model, such ephemerals used by the prover during its interaction with the malicious verifier should not help the adversary to impersonate the prover subsequently. In this paper we concentrate on the Schnorr IS [1]. This particular scheme is one of the fundamental cryptographic building block, which security relies on the hardness of discrete logarithm problem (DLP). As such it can be used as a compatible part of more complex constructions, based on similar computational assumptions. E.g. authenticated key establishment protocols based on Diffie-Hellman key exchange. The regular Schnorr IS is vulnerable to the ephemeral injecting attack, and ephemeral leakage. Thus we propose the modification of that scheme, which becomes secure in our proposed model.

Contribution. The contribution of the paper is the following:

- We introduce a new strong security model for identification schemes in which we allow the *adversary verifier* to set random values used on the prover side in the commitment phase of the protocol. We define the IS scheme to be secure if such a setting in *query stage* of the security experiment, will not enable the adversary to impersonate the prover later in the *impersonate stage*.
- We propose a modification of Schnorr authentication protocol [1], which becomes immune to malicious setting of the ephemeral key by the adversary. Such a setting neither leads to subsequent leakage of long term secret key of the prover, nor help the adversary to impersonate the prover later on. Subsequently we prove the security of the modified Schnorr IS in our strong model.

We argue that our proposition is especially applicable in the systems where the regular Schnorr IS is used, but where the scenarios of the ephemeral leakage could be taken under consideration. PACEAA protocol from [2] is such an example, where regular Schnorr IS is a part of deniable authentication process, and where it can be replaced by our modified version.

Previous Work. There are many fundamental identification schemes proposed so far, e.g. RSA based: [3] of Fiat and Shamir, [4] of Feige, Fiat and Shamir, or [5] of Guillou and Quisquater. In [1] Schnorr introduced DLP based construction, followed by [6] of Okamoto. There are also specialized identity based IS e.g. [7] provably secure in the standard model, or [8] secure against concurrent man-in-the-middle attack without random oracles by using a variant of BB signature scheme. Problem of the leakage of secret bits of the long term key of the prover, in Bounded Retrieval Model, was analyzed in [9]. The problem of the security of IS schemes under reset attacks on ephemeral secrets was raised in [10] in the context of zero-knowledge proofs. Later in [11] constructions for making the IS protocols immune against *reset attacks* were shown: the reset-secure identification protocols based on a deterministic, stateless digital signature scheme (as such that proposition is not deniable); the reset-secure identification protocols based on a CCA secure asymmetric encryption scheme (not naturally compatible with the Diffie-Hellman key exchange protocols initiated with the prover ephemeral public key); the reset-secure identification schemes based on pseudorandom functions and trapdoor commitments (has more than 3 rounds). Comparing with [11] our solution preserves the characteristic of the original Schnorr IS: (1) it is defined in groups suitable for Diffie-Helman key exchange; (2) it has three rounds - the first one is initiated with the prover's commitment; (3) it is deniable for the prover - i.e. it is simulatable by the verifier without the secret key of the prover.

The paper is organized in the following way. In Sect. 2 we recall the Schnorr identification protocol. In Sect. 3 we introduce our stronger security model which addresses the problem of the ephemeral setting by the active malicious adversary. In Sect. 4 we propose the modified version of Schnorr IS, and prove its security in our model.

2 Schnorr Identification Scheme

2.1 Preliminaries and Notation

We loosely follow the notation from [9]. Let $x_1, \ldots, x_n \leftarrow_R X$ denotes that each x_i is sampled uniformly at random from the set X. Let $\mathcal{G}(1^\lambda)$ be a group generation algorithm that takes as an inputs 1^λ, and outputs a tuple $\mathbb{G} = (p, q, g, G)$, where $p, q \in PRIMES$ s.t. $q|p-1$, $\mathbb{Z}_p^*$ be a multiplicative group modulo p, and $\langle g \rangle = G$ be a subgroup of $\mathbb{Z}_p^*$ of order q. Let $\mathcal{H} : \{0,1\}^* \rightarrow G$ be a hash function. We will use it to compute another element of G denoted by $\hat{g}$. We assume the following:

Bilinear Map: Let G_T be another group of a prime order q. We assume that $\hat{e} : G \times G \rightarrow G_T$ is a bilinear map s.t. following condition holds:

(1) *Bilinearity*: $\forall a, b \in \mathbb{Z}_q^*, \forall g, g \in G$: $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
(2) *Non-degeneracy*: $\hat{e}(g, g) \neq 1$.
(3) *Computability*: $\hat{e}$ is efficiently computable.

The *Discrete Logarithm* (DL) Assumption: For any probabilistic polynomial time (PPT) algorithm $\mathcal{A}_{\mathsf{DL}}$ it holds that:

$$\Pr[\mathcal{A}_{\mathsf{DL}}(\mathbb{G}, g^x) = x \mid \mathbb{G} \leftarrow_R \mathcal{G}(1^\lambda), x \leftarrow_R \mathbb{Z}_q^*] \leq \epsilon_{\mathsf{DL}}(\lambda),$$

where $\epsilon_{\mathsf{DL}}(\lambda)$ is negligible.

The *Computational Diffie-Hellman* (CDH) Assumption: For any probabilistic polynomial time (PPT) algorithm $\mathcal{A}_{\mathsf{CDH}}$ it holds that:

$$\Pr[\mathcal{A}_{\mathsf{CDH}}(\mathbb{G}, g^x, g^y) = g^{xy} \mid \mathbb{G} \leftarrow_R \mathcal{G}(1^\lambda), x \leftarrow_R \mathbb{Z}_q^*, y \leftarrow_R \mathbb{Z}_q^*] \leq \epsilon_{\mathsf{CDH}}(\lambda),$$

where $\epsilon_{\mathsf{CDH}}(\lambda)$ is negligible.
The *Decisional Diffie-Hellman* Oracle ($\mathcal{O}_{\mathsf{DDH}}$) denotes the (PPT) algorithm, which for $\mathbb{G} \leftarrow_R \mathcal{G}(1^\lambda), x \in \mathbb{Z}_q^*, y \in \mathbb{Z}_q^*, z \in \mathbb{Z}_q^*$

$$\mathcal{O}_{\mathsf{DDH}}(\mathbb{G}, g^x, g^y, g^z) = 1 \text{ iff } z = xy \mod q.$$

The *Gap Computational Diffie-Hellman* (GDH) Assumption: For any probabilistic polynomial time (PPT) algorithm $\mathcal{A}_{\mathsf{GDH}}^{\mathcal{O}_{\mathsf{DDH}}}$ that has access to decisional Diffie-Hellman oracle $\mathcal{O}_{\mathsf{DDH}}$ it holds that:

$$\Pr[\mathcal{A}_{\mathsf{GDH}}^{\mathcal{O}_{\mathsf{DDH}}}(\mathbb{G}, g^x, g^y) = g^{xy} \mid \mathbb{G} \leftarrow_R \mathcal{G}(1^\lambda), x \leftarrow_R \mathbb{Z}_q^*, y \leftarrow_R \mathbb{Z}_q^*] \leq \epsilon_{\mathsf{GDH}}(\lambda),$$

where $\epsilon_{\mathsf{GDH}}(\lambda)$ is negligible.

2.2 Identification Schemes

An identification scheme is a system in which a prover proves its identity to a verifier. More formally we define the following:

Definition 1 (Identification Scheme). *An identification scheme* IS *is a system which consists of four algorithms* (ParGen, KeyGen, $\mathcal{P}$, $\mathcal{V}$) *and a protocol* π*:*

$\mathsf{params} \leftarrow \mathsf{ParGen}(1^\lambda)$**:** *inputs the security parameter* λ*, and outputs public parameters available to all users of the system (we omit them from the rest of the description).*
$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}()$**:** *outputs the secret key* sk *and corresponding public key* pk*.*
$\mathcal{P}(\mathsf{pk}, \mathsf{sk})$**:** *denotes the prover – an ITM which interacts with the verifier* $\mathcal{V}$ *in the protocol* π*.*
$\mathcal{V}(\mathsf{pk})$**:** *denotes the verifier – an ITM which interacts with the prover* $\mathcal{V}$ *in the protocol* π*.*
$\pi(\mathcal{P}, \mathcal{V})$**:** *denotes the protocol between the prover and the verifier.*

We distinguish two stages of the scheme:

- *Initialization: In this stage parameters are generated:* $\mathsf{params} \leftarrow \mathsf{ParGen}(1^\lambda)$, *and users are registered, e.g. on behalf of the user of identity* $\hat{A}$ *the procedure* $(a, A) \leftarrow \mathsf{KeyGen}()$ *generates the pair of the secret key and the corresponding public key, denoted by* a *and* A *respectively.*
- *Operation: In this stage any user, e.g.* $\hat{A}$*, demonstrates its identity to a verifier by performing the protocol* $\pi(\hat{A}(a, A), \mathcal{V}(A))$ *related to the keys* a, A*. Finally the verifier outputs* 1 *for "accept" or* 0 *for "reject". For simplicity we denote* $\pi(\mathcal{P}, \mathcal{V}) \rightarrow 1$ *if* $\mathcal{P}$ *was accepted by* $\mathcal{V}$ *in* π*.*

We require that the scheme is complete i.e. protocol $\pi(\mathcal{P}(\mathsf{sk}, \mathsf{pk}), \mathcal{V}(\mathsf{pk})) \rightarrow 1$ *for any pair* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}()$*.*

There are many security modes for identification schemes. Intuitively the scheme is regarded as secure if it is impossible for any adversary prover algorithm $\mathcal{A}$, to be accepted, e.g. as identity $\hat{A}$, by the verifier given the public key A, without the input of the appropriate secret key $\mathsf{sk} = a$. That is we require that probability $\Pr[\pi(\mathcal{P}(\mathsf{sk}, \mathsf{pk}), \mathcal{V}(\mathsf{pk})) \rightarrow 1]$ is negligible. Now we denote formally the *passive adversary* mode that is used for the regular Schnorr identification. In this mode the adversary passively listens to the polynomial number ℓ of the protocol executions between the prover and the verifier, $\pi(\mathcal{P}(\mathsf{sk}, \mathsf{pk}), \mathcal{V}(\mathsf{pk}))$, hoping that these observations will, later on, help him to impersonate the prover (without the prover secret key), to the verifier. We denote the view $\mathsf{v}^{\mathcal{P},\mathcal{V},\ell} = \{T_1, \ldots, T_\ell\}$ as the total knowledge $\mathcal{A}$ can gain after the ℓ runs of $\pi(\mathcal{P}(\mathsf{sk}, \mathsf{pk}), \mathcal{V}(\mathsf{pk}))$, where T_i is the transcript of the protocol messages in ith execution.

Definition 2 (Passive Adversary (PA)). *Let* $\mathsf{IS} = (\mathsf{ParGen}, \mathsf{KeyGen}, \mathcal{P}, \mathcal{V}, \pi)$ *is an identification scheme. We define security experiment* $\mathtt{Exp}_{\mathsf{IS}}^{\mathsf{PA},\lambda,\ell}$*:*

`Init stage` : *Let* $\mathsf{params} \leftarrow \mathsf{ParGen}(1^\lambda)$, $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}()$*. Let the adversary* $\mathcal{A}$*, be the malicious algorithm given the public key* pk*.*

`Query stage` : $\mathcal{A}$ *passively observes a polynomial number* ℓ *of executions of the protocol* $\pi(\mathcal{P}(\mathsf{sk}, \mathsf{pk}), \mathcal{V}(\mathsf{pk}))$*. Let* $\mathsf{v}^{\mathcal{P},\mathcal{V},\ell} = \{T_1, \ldots, T_\ell\}$ *is the view* $\mathcal{A}$ *gains after the* ℓ *runs of* $\pi(\mathcal{P}(\mathsf{sk}, \mathsf{pk}), \mathcal{V}(\mathsf{pk}))$*, where* T_i *is the transcript of* i*th execution.*

`Impersonation stage` : $\mathcal{A}$ *runs the protocol* $\pi(\mathcal{A}(\mathsf{pk}, \mathsf{v}^{\mathcal{P},\mathcal{V},\ell}), \mathcal{V}(\mathsf{pk}))$ *with the honest verifier.*

We define the advantage of $\mathcal{A}$ *in the experiment* $\mathtt{Exp}_{\mathsf{IS}}^{\mathsf{PA},\lambda,\ell}$ *as probability of acceptance in the last stage:*

$$\mathbf{Adv}(\mathcal{A}, \mathtt{Exp}_{\mathsf{IS}}^{\mathsf{PA},\lambda,\ell}) = \Pr[\pi(\mathcal{A}(\mathsf{pk}, \mathsf{v}^{\mathcal{P},\mathcal{V},\ell}), \mathcal{V}(\mathsf{pk})) \rightarrow 1].$$

We say that the identification scheme IS *is secure if* $\mathbf{Adv}(\mathcal{A}, \mathtt{Exp}_{\mathsf{IS}}^{\mathsf{PA},\lambda,\ell})$ *is negligible in* λ*.*

2.3 Regular Schnorr Identification Scheme

Let us recall the Schnorr identification scheme from [1].

$\mathsf{params} \leftarrow \mathsf{ParGen}(1^\lambda)$: Let $\mathbb{G} = (p, q, g, G) \leftarrow \mathcal{G}(1^\lambda)$, s.t. DL assumption holds. Set $\mathsf{params} = (p, q, g, G)$.
$\mathsf{KeyGen}()$: $\mathsf{sk} = a \leftarrow \mathbb{Z}_q^*$, $\mathsf{pk} = A = g^a$. Output $(\mathsf{sk}, \mathsf{pk})$.
$\pi(\mathcal{P}(a, A), \mathcal{V}(A))$: The prover $\mathcal{P}(a, A)$ with identity $\hat{A}$ runs with the verifier $\mathcal{V}(A)$ the folowing protocol:

1. $\mathcal{P}$: computes $x \in_R \mathbb{Z}_q^*$, $X = g^x$ and sends X to the verifier $\mathcal{V}$.
2. $\mathcal{V}$: choses $c \in_R \mathbb{Z}_q^*$, and sends c to the prover $\mathcal{P}$.
3. $\mathcal{P}$: computes $s = x + ac$ and sends s to the verifier $\mathcal{V}$.
4. $\mathcal{V}$: accepts the verification iff $g^s == XA^c$.

Fig. 1. The Schnorr identification scheme.

Protocol Simulation: The eavesdropping passive adversary learns transcript tuple $T = (X, c, s)$. The random variables X, c, s are uniformly distributed on their domains. In the protocol variables $x = \log_g X, c$ are mutually independent, and together determine $s = x + ac$ for the fixed a. On the other hand the protocol transcript can be efficiently simulated by choosing $\tilde{s}, \tilde{c}$ first and subsequently computing $\tilde{X} = (g^{\tilde{s}}/A^{\tilde{c}})$. Then the simulator algorithm, denoted by $\mathcal{S}^{\pi}_{\mathsf{IS}}(\tilde{c} \leftarrow_R \mathbb{Z}_q^*, \tilde{s} \leftarrow_R \mathbb{Z}_q^*)$, can replay the precomputed transcript $\tilde{T} = (\tilde{X}, \tilde{c}, \tilde{s})$ in the correct order, thus simulating the interaction between the prover and the verifier. The tuples $T = (X, c, s)$ and $\tilde{T} = (\tilde{X}, \tilde{c}, \tilde{s})$ are identically distributed. As the immediate consequence of the *simulatability*, the security requirements for the above-mentioned protocol is that the challenge c is not know to the prover before it sends the commitment X to the verifier. This is especially crucial in the setups where the possible leakage on the verifier side is considered. In real implementation it must be ensured that the challenge value c is coined only after the verifier obtains the value X, but not earlier. Below we recall the security of Schnorr IS in PA model.

Rewinding Technique: The idea behind the proof is the following: If we have the efficient adversary algorithm $\mathcal{A}$ for which $\mathbf{Adv}(\mathcal{A}, \mathsf{Exp}^{\mathsf{PA},\lambda,\ell}_{\mathsf{IS}})) = \epsilon$ is non-negligible, then, with also non-negligible probability $\epsilon(\epsilon - 1/q)$, it can be run twice and accepted, for the same fixed ephemeral x, but with different challenges c_1, c_2 resulting with different responses s_1, s_2. The ϵ factor denotes the probability of acceptance in the first run, while $\epsilon - 1/q$ is the probability of acceptance in the second run for $c_1 \neq c_2$. The two tuples (x, c_1, s_1), (x, c_2, s_2) will help us to break the underlying DL problem.

Theorem 1. *Let* IS *denotes the Schnorr identification scheme (as of Fig. 1). is secure (in the sense of Definition 2), i.e. the advantage* $\mathbf{Adv}(\mathcal{A}, \mathsf{Exp}^{\mathsf{PA},\lambda,\ell}_{\mathsf{IS}}))$ *is negligible in* λ*, for any PPT algorithm* $\mathcal{A}$*.*

Proof (Sketch). The proof is by contradiction. Suppose there is an adversary $\mathcal{A}$ for which the advantage $\mathbf{Adv}(\mathcal{A}, \mathsf{Exp}^{\mathsf{PA},\lambda,\ell}_{\mathsf{IS}}))$ is non-negligible. Then it can be used as a subprocedure by the efficient algorithm $\mathcal{A}_{\mathsf{DL}}$ that breaks the DL assumption, computing e.g. $\log_g(A)$ for the given instance of DL problem $(\mathbb{G}, A)$, also with a non negligible probability, in the following way:

`Init stage` : Let $\mathsf{params} \leftarrow \mathbb{G} = (p, q, g, G)$, $\mathsf{pk} = A$. The adversary $\mathcal{A}$, is given the public key A.

`Query stage` : We simulate the view $\tilde{\mathsf{v}}^{\mathcal{P},\mathcal{V},\ell} = \{\tilde{T}_1, \ldots, \tilde{T}_\ell\}$ of $\mathcal{A}$ in the ℓ executions of the protocol π, where each $\tilde{T}_i = (\tilde{X}_i, \tilde{c}_i, \tilde{s}_i)$ for $\tilde{X} = (g^{\tilde{s}}/A^{\tilde{c}})$ produced by the simulator $\mathcal{S}^{\pi}_{\mathsf{IS}}()$.

`Impersonation stage` : We run the protocol $\pi(\mathcal{A}(\mathsf{pk}, \tilde{\mathsf{v}}^{\mathcal{P},\mathcal{V},\ell}), \mathcal{V}(\mathsf{pk}))$ serving the role of the honest verifier. Then we use a *rewinding technique*: we fix the random value x used in $X = g^x$ by the algorithm $\mathcal{A}$, and let $\mathcal{A}$ interact twice with the verifier, choosing each time a different random c, say c_1 and c_2. These will result with x, c_1, s_1 and x, c_2, s_2 accordingly. If the verifier accepts in both cases we have $s_1 = x + ac_1$ and $s_2 = x + ac_2$. Thus we have $s_1 - s_2 = a(c_1 - c_2)$, so we can compute $a = (s_1 - s_2)/(c_1 - c_2)$. □

3 New Stronger Security Model

In this section we propose the new strong security model for IS. In this model we assume that in the *learning phase* the adversary can influence the choices of ephemeral secrets of the prover in an adaptive manner. In the worst scenario, the malicious verifier denoted by $\tilde{\mathcal{V}}$, can choose ephemerals on behalf of $\mathcal{P}$, and inject them to $\mathcal{P}$, even against its will and without its knowledge, over some *subliminal* channel before the computation involving x on $\mathcal{P}$ side starts. Let $\bar{x}$ denotes the ephemeral secrets chosen by $\tilde{\mathcal{V}}$, and $\mathcal{P}^{\bar{x}}$ denotes the honest prover $\mathcal{P}$ with injected $\bar{x}$, which uses this value as the random ephemeral during the protocol execution. Furthermore, we assume the subsequent choices of $\tilde{\mathcal{V}}$ can be adjusted according to responses from $\mathcal{P}^{\bar{x}}$ during the subsequent protocol executions. We denote the protocol execution in which the ephemeral secrets $\bar{x}$ was chosen by $\tilde{\mathcal{V}}$ and $\mathcal{P}$ was forced to use it, as $\pi(\mathcal{P}^{\bar{x}}(\mathsf{sk}, \mathsf{pk}), \tilde{\mathcal{V}}(\mathsf{pk}, \bar{x}))$. We denote the view $\mathsf{v}^{\mathcal{P},\tilde{\mathcal{V}},\bar{\boldsymbol{x}}(\ell)}$ as the total knowledge $\tilde{\mathcal{V}}$ can gain after the polynomial number ℓ of executions $\pi(\mathcal{P}^{\bar{x}}(\mathsf{sk}, \mathsf{pk}), \tilde{\mathcal{V}}(\mathsf{pk}, \bar{x}))$, where $\bar{\boldsymbol{x}}(\ell) = \{\bar{x}_1, \ldots, \bar{x}_\ell\}$ are the adaptive choices of $\tilde{\mathcal{V}}$.

Definition 3 (Chosen Prover Ephemeral – (CPE)). *Let* IS = (ParGen, KeyGen, $\mathcal{P}$, $\mathcal{V}$, π) *is an identification scheme. We define security experiment* $\mathsf{Exp}^{\mathsf{CPE},\lambda,\ell}_{\mathsf{IS}}$*:*

`Init stage` : *Let* $\mathsf{params} \leftarrow \mathsf{ParGen}(1^\lambda)$, $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}()$. *Let the adversary* $\mathcal{A}$*, be the coalition of malicious algorithms* $(\tilde{\mathcal{P}}, \tilde{\mathcal{V}})$ *given the public key* pk.

`Query stage` : $\mathcal{A}$ *runs a polynomial number* ℓ *of executions of the protocol* $\pi(\mathcal{P}^{\bar{x}_i}(\mathsf{sk}, \mathsf{pk}), \tilde{\mathcal{V}}(\mathsf{pk}, \bar{x}_i)$ *with the honest prover* $\mathcal{P}^{\bar{x}_i}$*, collecting* $\mathsf{v}^{\mathcal{P},\tilde{\mathcal{V}},\bar{\boldsymbol{x}}(\ell)}$*, where* $\bar{x}_i \in \{\bar{x}_1, \ldots, \bar{x}_\ell\}$ *denotes the adaptive choices of* $\tilde{\mathcal{V}}$ *injected as ephemerals to the prover* $\mathcal{P}^{\bar{x}_i}$ *in the* i*th execution.*

`Impersonation stage` : $\mathcal{A}$ *runs the protocol* $\pi(\tilde{\mathcal{P}}(\mathsf{pk}, \mathsf{v}^{\mathcal{P},\tilde{\mathcal{V}},\bar{\boldsymbol{x}}(\ell)}), \mathcal{V}(\mathsf{pk}))$ *with the honest verifier.*

We define the advantage of $\mathcal{A}$ *in the experiment* $\mathsf{Exp}^{\mathsf{CPE},\lambda,\ell}_{\mathsf{IS}}$ *as probability of acceptance in the last stage:*

$$\mathbf{Adv}(\mathcal{A}, \mathsf{Exp}^{\mathsf{CPE},\lambda,\ell}_{\mathsf{IS}}) = \Pr[\pi(\tilde{\mathcal{P}}(\mathsf{pk}, \mathsf{v}^{\mathcal{P},\tilde{\mathcal{V}},\bar{\boldsymbol{x}}(\ell)}), \mathcal{V}(\mathsf{pk})) \rightarrow 1].$$

We say that the identification scheme is secure if $\mathbf{Adv}(\mathcal{A}, \mathrm{Exp}_{\mathsf{IS}}^{\mathsf{CPE},\lambda,\ell})$ *is negligible in* λ.

It is easy to check that the regular Schnorr IS is not secure in the proposed CPE model. The adversary with the injected $\bar{x}$ can easily compute the secret key $a = (s - \bar{x})/c$, and impersonate the prover $\hat{A}$ later on.

4 Modified Schnorr Identification Scheme

The idea behind the modification is the following. We want to address the threat that the adversary with the knowledge of c, $s = x + ac$ and the leaked x computes static secret a. Therefore instead of sending s in plain-text, the prover will send s hidden in the exponent $S = \hat{g}^s$, where the new generator $\hat{g} = \mathcal{H}(X|c)$ is obtained with the hash function $\mathcal{H}$. Now even if the ephemeral value x is leaked, the adversary should face DLP to obtain the value a from S. On the other hand we use the bilinear map $\hat{e}$ on the verifier side to check the linear equation $s = x + ac$ in the exponent. Indeed it holds that $\hat{e}(S, g) = \hat{e}(\mathcal{H}(X|c)), XA^c)$ due to the fact that $\hat{e}(\mathcal{H}(X|c), XA^c) = \hat{e}(\mathcal{H}(X|c)^{x+ac}, g)$.

The proposed modified Schnorr IS is depicted in Fig. 2.

$\mathsf{params} \leftarrow \mathsf{ParGen}(1^\lambda)$**:** Let $p, q, g, G, G_T \leftarrow \mathcal{G}(1^\lambda)$. Let $\mathcal{H} : \{0,1\}^* \rightarrow G$ be a hash function. Let $\hat{e} : G \times G \rightarrow G_T$ be a bilinear map. We assume that GDH holds in G, where $\hat{e}$ plays a role of $\mathcal{O}_{\mathsf{DDH}}$ oracle. Set $\mathsf{params} = (p, q, g, G, G_T, \mathcal{H}, \hat{e})$.

$\mathsf{KeyGen}()$**:** $\mathsf{sk} = a \leftarrow \mathbb{Z}_q^*$, $\mathsf{pk} = A = g^a$. Output $(\mathsf{sk}, \mathsf{pk})$.

$\pi(\mathcal{P}(a, A), \mathcal{V}(A))$**:** The prover $\mathcal{P}(a, A)$ with identity $\hat{A}$ runs with the verifier $\mathcal{V}(A)$ the folowing protocol:

1. $\mathcal{P}$: computes $x \in_R \mathbb{Z}_q^*$, $X = g^x$ and sends X to the verifier $\mathcal{V}$.
2. $\mathcal{V}$: choses $c \in_R \mathbb{Z}_q^*$, and sends c to the prover $\mathcal{P}$.
3. $\mathcal{P}$: computes $\hat{g} = \mathcal{H}(X|c)$, $S = \hat{g}^{x+ac}$ and sends S to the verifier $\mathcal{V}$.
4. $\mathcal{V}$: computes $\hat{g} = \mathcal{H}(X|c)$, accepts the proof iff $\hat{e}(S, g) = \hat{e}(\mathcal{H}(X|c), XA^c)$.

Fig. 2. The modified Schnorr identification scheme

In Fig. 3 we depict side-by-side the differences between the original Schnorr and our modified version. In our proposition we have one exponentiation and one hashing more on provers side. On the other hand the verifier does not have extra exponentiation during the verification. Indeed it has even one explicit exponentiation less – does not have to compute g^s. Instead, additionally it has to compute the hash, and compare results of two bilinear functions.

4.1 Simulation in the *Passive Adversary* Mode

The modified Schnorr IS preserves the simulatability property of its original version. In the weaker *passive adversary* model, the eavesdropping adversary

Regular Schnorr		**Modified Schnorr**	
$\mathcal{P}(a, A = g^a)$	$\mathcal{V}(A)$	$\mathcal{P}(a, A = g^a)$	$\mathcal{V}(A)$
$x \in_R \mathbb{Z}_q^*, X = g^x$		$x \in_R \mathbb{Z}_q^*, X = g^x$	
$\xrightarrow{X}$		$\xrightarrow{X}$	
	$c \in_R \mathbb{Z}_q^*$		$c \in_R \mathbb{Z}_q^*$
$\xleftarrow{c}$		$\xleftarrow{c}$	
		$\hat{g} = \mathcal{H}(X\|c)$	
$s = x + ac$ $\xrightarrow{s}$		$S = \hat{g}^{x+ac}$ $\xrightarrow{S}$	$\hat{g} = \mathcal{H}(X\|c)$
	Accept iff $g^s = XA^c$		Accept iff $\hat{e}(S, g) = \hat{e}(\hat{g}, XA^c)$

Fig. 3. Schnorr identification comparison

learns transcript tuple $T = (X, c, S)$, similarly as in the original Schnorr IS. The random variables X, c, S are uniformly distributed on their domains. In π variables $x = \log_g X, c$ are mutually independent, and together determine $S = \hat{g}^{x+ac}$ for the fixed a. On the other hand the protocol transcript can be efficiently simulated by choosing $\tilde{s}, \tilde{c}$ first, then subsequently computing $\tilde{X} = (g^{\tilde{s}}/A^{\tilde{c}})$, and only then $\hat{g} = \mathcal{H}(\tilde{X}|\tilde{c})$ and $\tilde{S} = \hat{g}^{\tilde{s}}$. Observe that for this transcript the verification holds: $\hat{e}(\tilde{S}, g) = \hat{e}(\mathcal{H}(\tilde{X}|\tilde{c}), \tilde{X}A^{\tilde{c}})$. Then the simulator algorithm, denoted by $\mathcal{S}_{\mathsf{IS}}^{\pi}(\tilde{c} \leftarrow_R \mathbb{Z}_q^*, \tilde{s} \leftarrow_R \mathbb{Z}_q^*)$, can replay the precomputed transcript $\tilde{T} = (\tilde{X}, \tilde{c}, \tilde{s})$ in the correct order, thus simulating the interaction between the prover and the verifier. The tuples $T = (X, c, s)$ and $\tilde{T} = (\tilde{X}, \tilde{c}, \tilde{s})$ are identically distributed.

4.2 Simulation in the *Chosen Prover Ephemeral* Mode

The modified Schnorr IS is also simulatable in the proposed stronger *Chosen Prover Ephemeral* (CPE) model. Assuming programmable ROM (Random Oracle Model), we can simulate the protocol $\pi(\mathcal{P}^{\bar{x}}(\mathsf{pk}), \tilde{\mathcal{V}}^{\mathcal{O}_{\mathcal{H}}}(\mathsf{pk}, \bar{x})) \to 1$ on behalf of the prover $\mathcal{P}^{\bar{x}}(\mathsf{pk})$ without the secret key sk, using the injected ephemerals $\bar{x}$, and interacting with the active adversary $\tilde{\mathcal{V}}^{\mathcal{O}_{\mathcal{H}}}(\mathsf{pk}, \bar{x})$, which injects the ephemerals $\bar{x}$ to the prover and performs adaptive choices of challenges. Note that the adversary calls the oracle $\mathcal{O}_{\mathcal{H}}$ to compute the hash value for the queried input.

Theorem 2. *The modified Schnorr protocol (depicted in Fig. 2) is simulatable in the CPE model (of Definition 3).*

Proof. The simulator $\mathcal{S}_{\mathsf{IS}}^{\mathsf{CPE},\pi}()$ is defined in the following way:

(1) **Hash queries** $\mathcal{O}_{\mathcal{H}}$: We setup ROM table for hash queries $\mathcal{O}_{\mathcal{H}}$. The table has three columns I, H, r: for the input, the output and the masked exponent respectively. On each query $\mathcal{O}_{\mathcal{H}}(I_i)$ we check if we have it already defined - if so we return the corresponding output H_i. Otherwise we choose $r_i \leftarrow_R \mathbb{Z}_q^*$, compute $H_i = g^{r_i}$, place a new row (I_i, H_i, r_i) in the ROM table, and return H_i.

(2) **Commitment** X: When injected ephemeral $\bar{x}$ we use it to compute $\tilde{X} = g^{\bar{x}}$. We send $\bar{X}$ to the verifier $\tilde{\mathcal{V}}^{\mathcal{O}_\mathcal{H}}(\mathsf{pk}, \bar{x})$ in the first message.
(3) **Proof** S: On receiving $\tilde{c}$ from the verifier, we call $\mathcal{O}_\mathcal{H}(\bar{X}|\tilde{c})$ We check $\mathcal{O}_\mathcal{H}$ table for the input $\bar{X}|c$, locate and retrieve the corresponding g^r and r. We set $\hat{g} = g^r$. We compute $S = \tilde{X}^r A^{rc} = \hat{g}^{\bar{x}+ac}$ for $\hat{g} \leftarrow \mathcal{O}_\mathcal{H}(\bar{X}|\tilde{c})$. Now verification on prover side holds: $\hat{e}(\tilde{S}, g) = \hat{e}(\hat{g}, \tilde{X}A^{\tilde{c}})$ for $\hat{g} \leftarrow \mathcal{O}_\mathcal{H}(\bar{X}|\tilde{c})$, and the *real* transcript tuple $T = (X, c, s)$, and the *simulated* $\tilde{T} = (\tilde{X}, \tilde{c}, \tilde{s})$ are identically distributed. □

4.3 Security Analysis

We follow the same proving methodology as in the case of the original Schnorr IS. First we allow the adversary to gain some knowledge: we simulate the proofs on behalf of the prover (but without its secret key) interacting with malicious verifier, which injects the ephemerals for our usage. We are able to do this in ROM. Then, assuming that the advantage of the adversary is non-negligible, in the impersonation stage we use *rewinding technique* for obtaining two tuples (X, c_1, S_1), (X, c_2, S_2) which subsequently will help us to break the underlying hard problem - GDH in this case - also with non-negligible probability.

Theorem 3. *Let* IS *denotes the modified Schnorr identification scheme (as of Fig. 2).* IS *is secure (in the sense of Definition 3), i.e. the advantage* $\mathbf{Adv}(\mathcal{A}, \mathtt{Exp}_{\mathsf{IS}}^{\mathsf{CPE},\lambda,\ell})$) *is negligible in* λ*, for any PPT algorithm* $\mathcal{A}$*.*

Proof (Sketch). We use ROM for hash queries. The proof is by contradiction. Suppose there is an adversary $\mathcal{A} = (\tilde{\mathcal{P}}, \tilde{\mathcal{V}})$ for which $\mathbf{Adv}(\mathcal{A}, \mathtt{Exp}_{\mathsf{IS}}^{\mathsf{CPE},\lambda,\ell})$) is non-negligible. Then it can be used as a subprocedure by the efficient algorithm $\mathcal{A}_{\mathsf{GDH}}$ that breaks the GDH for the given instance g^α, g^β, computing $g^{\alpha\beta}$ also with non-negligible probability.

`Init stage` : Let $\mathsf{params} \leftarrow \mathbb{G} = (p, q, g, G)$ s.t. CDH holds and (g^α, g^β) is GDH instance in $\mathbb{G}$. We set $\mathsf{pk} = g^\alpha$. The adversary $\mathcal{A}$, is given the public key $\mathsf{pk} = g^\alpha$. We setup ROM table for hash queries $\mathcal{O}_\mathcal{H}$. The table has three columns I, H, r for the input, the output and the masked exponent respectively. We will serve hash in the following way: in the `Query stage` we will use the simulator $\mathcal{S}_{\mathsf{IS}}^{\mathsf{CPE},\pi}$ (as in the proof of Theorem 2). In the `Impersonation stage` we will provide to the adversary the value $(g^\beta)^r$, where r is a random mask.

`Query stage` : We simulate in ROM a polynomial number ℓ of executions of the protocol $\pi(\mathcal{P}^{\bar{x}_i}(\mathsf{pk}), \tilde{\mathcal{V}}^{\mathcal{O}_\mathcal{H}}(\mathsf{pk}, \bar{x}_i))$ without the secret key, interacting with the active adversary verifier $\tilde{\mathcal{V}}^{\mathcal{O}_\mathcal{H}}(\mathsf{pk}, \bar{x}_i)$, which injects ephemerals $\bar{x}_i$ by running the simulator $\mathcal{S}_{\mathsf{IS}}^{\mathsf{CPE},\pi}$:

(1) **Serving Hash queries** $\mathcal{O}_\mathcal{H}$**:** On each query $\mathcal{O}_\mathcal{H}(I_i)$ we check if we have it already defined - if so we return the corresponding output H_i. Otherwise we choose $r_i \leftarrow_R \mathbb{Z}_q^*$, compute $H_i = g^{r_i}$, place a new row (I_i, H_i, r_i) in the ROM table, and return H_i.

(2) **Commitment** X**:** When injected ephemeral $\bar{x}$ we use it to compute $\tilde{X} = g^{\bar{x}}$. We send $\tilde{X}$ to the verifier $\tilde{\mathcal{V}}^{\mathcal{O}_{\mathcal{H}}}(\mathsf{pk}, \bar{x})$ in the first message.
(3) **Proof** S**:** On receiving $\tilde{c}$ from the verifier, we call $\mathcal{O}_{\mathcal{H}}(\tilde{X}|\tilde{c})$. We check $\mathcal{O}_{\mathcal{H}}$ table for the input $\tilde{X}|c$, locate and retrieve the corresponding g^r and r. We set $\hat{g} = g^r$. We compute $S = \tilde{X}^r A^{rc} = \hat{g}^{\bar{x}+ac}$ for $\hat{g} \leftarrow \mathcal{O}_{\mathcal{H}}(\tilde{X}|\tilde{c})$. Now verification on prover side holds: $\hat{e}(\tilde{S}, g) = \hat{e}(\hat{g}, \tilde{X}A^{\tilde{c}})$ for $\hat{g} \leftarrow \mathcal{O}_{\mathcal{H}}(\tilde{X}|\tilde{c})$. Observe that the *simulated* transcript tuple $\tilde{T} = (\tilde{X}, \tilde{c}, \tilde{s})$, and the tuple from the *real* protocol $T = (X, c, s)$ and are identically distributed. Let $\mathsf{v}^{\mathcal{P}, \tilde{\mathcal{V}}, \bar{\boldsymbol{x}}(\ell)}$ be the view of the adversary collected in this stage, where $\bar{x}_i \in \{\bar{x}_1, \ldots, \bar{x}_\ell\} = \bar{\boldsymbol{x}}(\ell)$ denotes the adaptive choices of $\tilde{\mathcal{V}}$ injected as ephemerals to the prover $\mathcal{P}^{\bar{x}_i}$ in ith execution.

`Impersonation stage` : In ROM we run $\pi(\tilde{\mathcal{P}}^{\mathcal{O}_{\mathcal{H}}}(\mathsf{pk}, \mathsf{v}^{\mathcal{P}, \tilde{\mathcal{V}}, \bar{\boldsymbol{x}}(\ell)}), \mathcal{V}(\mathsf{pk}))$ serving the role of the honest verifier. We use the *rewinding technique*: we fix the random value x used in $X = g^x$ by the algorithm $\tilde{\mathcal{P}}$, and let $\tilde{\mathcal{P}}$ interact twice with the verifier, choosing each time a different random challenge, c_1 and c_2, such that neither $X|c_1$ nor $X|c_2$ were the input to $\mathcal{O}_{\mathcal{H}}$ in `Query stage`, and setting $H_1 = \mathcal{O}_{\mathcal{H}}(X|c_1) \leftarrow (g^\beta)^{r_1}$, $H_2 = \mathcal{O}_{\mathcal{H}}(X|c_2) \leftarrow (g^\beta)^{r_2}$ for $r_1, r_2 \leftarrow_R \mathbb{Z}_q^*$. These will result with $(X, c_1, S_1, \hat{g}_1, r_1)$ and $(X, c_2, S_2, \hat{g}_2, r_2)$ accordingly. If we accept the adversary prover both times by checking: $\hat{e}(S_1, g) = \hat{e}(\hat{g}_1, XA^{c_1})$, and $\hat{e}(S_2, g) = \hat{e}(\hat{g}_2, XA^{c_2})$, we conclude: $S_1 = (g^{\beta r_1})^x (g^{\beta r_1})^{\alpha c_1}$ and $S_2 = (g^{\beta r_2})^x (g^{\beta r_2})^{\alpha c_2}$. Thus we have $S_1^{r_1^{-1}} / S_2^{r_2^{-1}} = (g^\beta)^{\alpha c_1 - \alpha c_2}$, so we can compute $g^{\alpha\beta} = (S_1^{r_1^{-1}} / S_2^{r_2^{-1}})^{(c_1 - c_2)^{-1}}$. □

5 Conclusion

In this paper we modify the Schnorr IS from [1] in such a way that it becomes immune to the ephemeral key setting. Such a setting can be done e.g. by the malicious verifier who exploits the knowledge of the erroneous pseudo-random number generator implemented into the device of the prover. We observe that secret key of the prover, masked by the ephemeral values in the response message of the protocol, are no longer secure in such setups. Therefore the prover, in the response message, sends the fragile values hidden in the exponent. The verifier uses bilinear maps to check the equality of the equation in exponent on its side for the commitment and the public key of the prover. We introduce the new stronger security model to cover that scenario. We prove the security of the proposed scheme in our model.

References

1. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991)
2. Bender, J., Dagdelen, Ö., Fischlin, M., Kügler, D.: The PACE—AA protocol for machine readable travel documents, and its security. In: Keromytis, A.D. (ed.) FC 2012. LNCS, vol. 7397, pp. 344–358. Springer, Heidelberg (2012). http://dx.doi.org/10.1007/978-3-642-32946-3_25

3. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). http://dx.doi.org/10.1007/3-540-47721-7_12
4. Feige, U., Fiat, A., Shamir, A.: Zero-knowledge proofs of identity. J. Cryptol. 1(2), 77–94. http://dx.doi.org/10.1007/BF02351717
5. Guillou, L.C., Quisquater, J.-J.: A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 123–128. Springer, Heidelberg (1988). http://dl.acm.org/citation.cfm?id=55554.55565
6. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (1993). http://dx.doi.org/10.1007/3-540-48071-4_3
7. Kurosawa, K., Heng, S.-H.: Identity-based identification without random oracles. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganà, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3481, pp. 603–613. Springer, Heidelberg (2005). http://dx.doi.org/10.1007/11424826_64
8. Kurosawa, K., Heng, S.-H.: The power of identification schemes. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 364–377. Springer, Heidelberg (2006). http://dx.doi.org/10.1007/11745853_24
9. Alwen, J., Dodis, Y., Wichs, D.: Leakage-resilient public-key cryptography in the bounded-retrieval model. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 36–54. Springer, Heidelberg (2009). http://dx.doi.org/10.1007/978-3-642-03356-8_3
10. Canetti, R., Goldreich, O., Goldwasser, S., Micali, S.: Resettable zero-knowledge (extended abstract). In: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, STOC 2000, pp. 235–244. ACM, New York (2000). http://doi.acm.org/10.1145/335305.335334
11. Bellare, M., Fischlin, M., Goldwasser, S., Micali, S.: Identification protocols secure against reset attacks. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 495–511. Springer, Heidelberg (2001). http://dx.doi.org/10.1007/3-540-44987-6_30