



A Generic View on the Unified Zero-Knowledge Protocol and its Applications

Diana Maimuț¹  and George Teșeleanu^{1,2} 

¹ Advanced Technologies Institute
10 Dinu Vintilă, Bucharest, Romania
{diana.maimut,tgeorge}@dcti.ro

² Simion Stoilow Institute of Mathematics of the Romanian Academy
21 Calea Grivitei, Bucharest, Romania

Abstract. We present a generalization of Maurer’s unified zero-knowledge (UZK) protocol, namely a unified generic zero-knowledge (UGZK) construction. We prove the security of our UGZK protocol and discuss special cases. Compared to UZK, the new protocol allows to prove knowledge of a vector of secrets instead of only one secret. We also provide the reader with a hash variant of UGZK and the corresponding security analysis. Last but not least, we extend Cogliani *et al.*’s lightweight authentication protocol by describing a new distributed unified authentication scheme suitable for wireless sensor networks and, more generally, the Internet of Things.

1 Introduction

Zero knowledge proofs (ZKPs) are closely related with one of the main cryptographic goals, entity authentication. Applying ZKPs, researchers are able to propose clever solutions to a variety of practical problems mainly in the fields of digital cash, auctioning, Internet of Things (IoT), password authentication and so on.

A standard zero knowledge protocol involves a prover *Peggy* possessing a piece of secret information x associated with her identity and a verifier *Victor* which has to check that *Peggy* indeed owns x . Two classical examples of such constructions are the Schnorr [18] and the Guillou-Quisquater [11] protocols. Raising the level of abstraction, Maurer shows in [13] that the previously mentioned protocols are actually instantiations of the same one.

Building on Maurer’s result, we considered of great interest providing the reader with a generalized perspective of the Unified Zero-Knowledge (UZK) protocol as well as a hash variant of it. An important consequence of our generic approach is the unification of Maurer’s [13], Feige-Fiat-Shamir’s [3] and Chaum-Everste-Van De Graaf’s [1] protocols. Moreover, a special case of our protocol’s hash version is the *h-variant* of the Fiat-Shamir scheme [7,9].

Practical implications which motivated our research. As the IoT paradigm arised, lightweight devices³ became more and more popular. Due to the open and distributed nature of the IoT, proper security is needed for the entire network to operate accordingly. Now let us consider the case of online wireless sensor networks (WSNs). The lightweight nature of sensor nodes heavily restricts cryptographic operations. Thus, the need for specific cryptographic solutions becomes obvious. The Fiat-Shamir-like distributed authentication protocol presented in [2] represents such an example. Based on this previous construction we propose a unified generic zero-knowledge protocol. Just as the result described in [2], our protocol can be applied for securing WSNs and, more generally, IoT-related solutions. Nonetheless, our construction offers flexibility when choosing the assumptions on which its security relies. A secondary feature of our scheme is the possibility of reusing existing certificates when implementing the distributed authentication protocol.

³low-cost devices with limited resources, be it computational or physical

Structure of the paper. We establish notations and recall zero-knowledge concepts in Section 2. Inspired by Maurer’s UZK construction, in Section 3 we present our main result, a Unified Generic Zero-Knowledge (UGZK) protocol, and prove it secure. We provide the reader with various special cases of UGZK in Section 4. A hash variant of our core protocol is tackled in Section 5 together with its security analysis. Following Cogliani *et al.*’s lightweight authentication protocol ideas, in Section 6 we describe a distributed unified Fiat-Shamir-based protocol, discuss security and complexity aspects and present implementation trade-offs which arise from small variations of the proposed result. We conclude in Section 7 and underline future work proposals.

2 Preliminaries

Notations. Throughout the paper, the notation $|S|$ denotes the cardinality of a set S . The subset $\{0, \dots, s\} \in \mathbb{N}$ is denoted by $[0, s]$. The action of selecting a random element x from a sample space X is represented by $x \stackrel{\$}{\leftarrow} X$, while $x \leftarrow y$ indicates the assignment of value y to variable x .

2.1 Groups

Let (\mathbb{G}, \star) and (\mathbb{H}, \otimes) be two groups. We assume that the group operations \star and \otimes are efficiently computable.

Let $f : \mathbb{G} \rightarrow \mathbb{H}$ be a function (not necessarily one-to-one). We say that f is a homomorphism if $f(x \star y) = f(x) \otimes f(y)$. Throughout the paper we consider f to be a one-way function, *i.e.* it is infeasible to compute x from $f(x)$. To be consistent with [13], we denote by $[x]$ the value $f(x)$. Note that given $[x]$ and $[y]$ we can efficiently compute $[x \star y] = [x] \otimes [y]$, due to the fact that f is a homomorphism.

2.2 Zero-Knowledge Protocols

Let $Q : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{\mathbf{true}, \mathbf{false}\}$ be a predicate. Given a value z , Peggy will try to convince Victor that she knows a value x such that $Q(z, x) = \mathbf{true}$.

We further base our reasoning on both a definition from [3, 13] and a definition from [10, 13] which we recall next.

Definition 1 (Proof of Knowledge Protocol). *An interactive protocol (P, V) is a proof of knowledge protocol for predicate Q if the following properties hold*

- **Completeness:** V accepts the proof when P has as input a value x with $Q(z, x) = \mathbf{true}$;
- **Soundness:** there exists an efficient program K (called knowledge extractor) such that for any \bar{P} (possibly dishonest) with non-negligible probability of making V accept the proof, K can interact with \bar{P} and output (with overwhelming probability) an x such that $Q(z, x) = \mathbf{true}$.

Definition 2 (Zero Knowledge Protocol). *A protocol (P, V) is zero-knowledge if for every efficient program \bar{V} there exists an efficient program S , the simulator, such that the output of S is indistinguishable from a transcript of the protocol execution between P and \bar{V} . If the indistinguishability is perfect⁴, then the protocol is called perfect zero-knowledge.*

⁴*i.e.* the probability distribution of the simulated and the actual transcript are identical

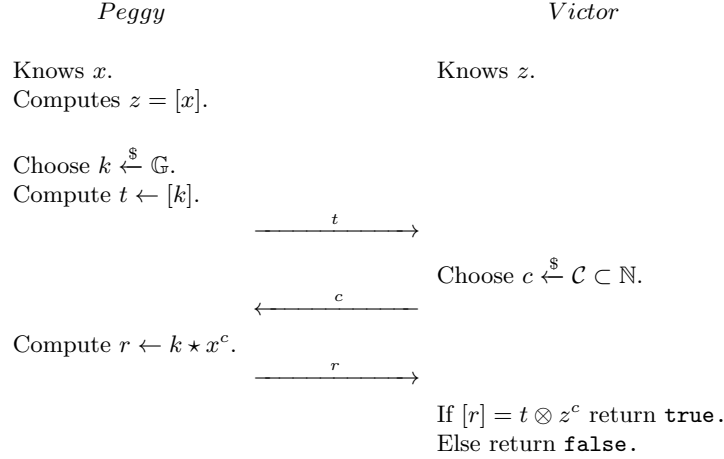


Fig. 1. Maurer’s Unified Zero-Knowledge (UZK) Protocol.

According to [13], the UZK protocol presented in Figure 1 is a zero-knowledge protocol if the conditions mentioned in Theorem 1 are satisfied.

Theorem 1. *Let \mathcal{C} be the challenge space. If values $\ell \in \mathbb{Z}$ and $u \in \mathbb{G}$ are known such that*

- $\gcd(c_0 - c_1, \ell) = 1$ for all $c_0, c_1 \in \mathcal{C}$ with $c_0 \neq c_1$,
- $[u] = z^\ell$,

then by running the protocol described in Figure 1 for m rounds we obtain a proof of knowledge protocol if $1/|\mathcal{C}|^m$ is negligible, and a zero-knowledge protocol if $|\mathcal{C}|$ is polynomially bounded.

Remark 1. If \mathcal{C} is small, then several 3-move rounds are needed to make the soundness error negligible.

2.3 Hash Functions

In the following, we consider the definitions from [9]. These concepts are further applied in Section 5 within the security proof of our proposed generalization of the h -variant protocol [7].

Definition 3. *Let $\lambda \geq 2$ be an integer. An λ -collision for a hash function h is an λ -tuple $\{m_i\}_{i \in [1, \lambda]}$ such that $h(m_1) = h(m_2) = \dots = h(m_\lambda)$.*

Definition 4. *Let $\lambda \geq 2$ be an integer. A hash function is λ -collision resistant if it is computationally infeasible to find a λ -collision.*

3 The Main Protocol

Inspired by Maurer’s UZK protocol [13], we describe a UGZK protocol (Figure 2). Note that the UZK scheme is a special case of the UGZK construction. We also prove the security of our proposed construction in a Feige-Fiat-Shamir manner [3].

3.1 Description

Let n be a positive integer and let $i \in [1, n]$. For a given vector $\{z_i\}_{i \in [1, n]}$, the protocol in Figure 2 is a proof of knowledge⁵ of a vector $\{[x_i]\}_{i \in [1, n]}$ such that $z_i = [x_i]$. The challenge spaces \mathcal{C}_i for the elements c_i are chosen as arbitrary subsets of \mathbb{N} , for all $i \in [1, n]$. For the sake of uniformity, we assume that all the challenge spaces \mathcal{C}_i are equal and we denote them by \mathcal{C} . If $|\mathcal{C}|$ is chosen to be small, then several rounds are needed in order to reduce the soundness error up to the point of being negligible.

When $n = 1$ we obtain the UZK protocol introduced in [13]. Note that in this case \mathbb{G} and \mathbb{H} need not be commutative.

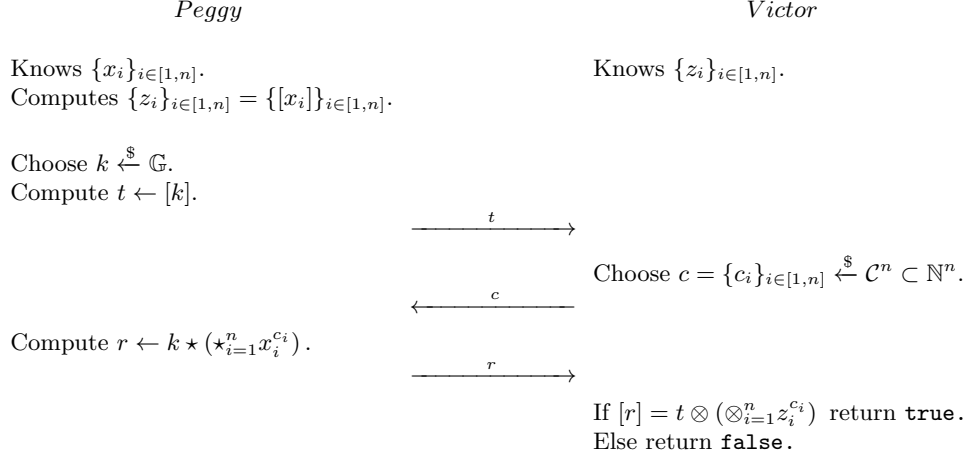


Fig. 2. A Unified Generic Zero-Knowledge (UGZK) Protocol.

3.2 Security Analysis

Theorem 2. *Let \mathbb{H} be a commutative group and let $j \in [1, n]$. If values $\ell_j \in \mathbb{Z}$ and $u_j \in \mathbb{G}$ are known such that*

- $\gcd(c'_j - c'_j, \ell_j) = 1$ for all $c'_j, c''_j \in \mathcal{C}$ with $c'_j \neq c''_j$,
- $[u_j] = z_j^{\ell_j}$,

then by running the protocol described in Figure 2 for m rounds we obtain a proof of knowledge protocol if $1/|\mathcal{C}|^{nm}$ is negligible, and a zero-knowledge protocol if $|\mathcal{C}|^n$ is polynomially bounded.

Proof. Let $s = |\mathcal{C}|$. To prove that P 's proof always convinces V , we evaluate the verification condition:

$$[r] = [k \star (\star_{i=1}^n x_i^{c_i})] = [k] \otimes (\otimes_{i=1}^n [x_i]^{c_i}) = t \otimes (\otimes_{i=1}^n z_i^{c_i}).$$

Note that a corrupt \bar{P} can cheat V with a negligible probability s^{-nm} per iteration by guessing the $\{c_i\}_{i \in [1, n]}$ vector, preparing $t = [k] \otimes (\otimes_{i=1}^n z_i^{-c_i})$ in the first step, and providing $r = k$ in the last step.

Next, we show that whenever V accepts \bar{P} 's proof with non-negligible probability, there exists a knowledge extractor K that can print out all the x_i s with overwhelming probability. Let T be the truncated execution tree of (\bar{P}, V) for input I and random tape RA . As in [3, Theorem 3], the algorithm we construct explores this tree by repeatedly resetting \bar{P} to the root, providing the necessary steering requests and verifying which

⁵provided that the conditions of Theorem 2 are satisfied

one of the s sons of each explored vertex corresponds to a correct answer. V may ask s^n possible questions at each stage and, thus, the vertices in T may have polynomially many sons in terms of $|I|$. A vertex is called *heavy* if its degree is larger than s^{n-1} (i.e. if more than s^{n-1} executions of (\bar{P}, V) at this state are successful). Our goal in this part of the proof is to show that all the x_i s can be computed from the sons of a heavy vertex and that a PPT K can find a heavy vertex in T with overwhelming probability.

Let H be any heavy vertex in T and let Q be the set of queries in the form of vectors $\{c_i\}_{i \in [1, n]}$ which are properly answered by \bar{P} . It is easy to show that for any $1 \leq j \leq n$ a set Q of more than s^{n-1} vectors (having the length n) must contain two vectors $\{c'_i\}_{i \in [1, n]}$ and $\{c''_i\}_{i \in [1, n]}$ in which $c'_j \neq c''_j$ and $c'_i = c''_i$ for all $i \neq j$. Since both queries were properly answered, the two verification conditions imply

$$[r'_j] = t'_j \otimes \left(\otimes_{i=1}^n z_i^{c'_i} \right) \text{ and } [r''_j] = t''_j \otimes \left(\otimes_{i=1}^n z_i^{c''_i} \right).$$

However, \bar{P} must choose t before he obtains V 's query and, thus, $t'_j = t''_j$. From r'_j and r''_j we can obtain \tilde{x}_j such that $[\tilde{x}_j] = z_j$, as

$$\tilde{x}_j = u_j^{a_j} \star (r_j''^{-1} \star r'_j)^{b_j},$$

where a_j and b_j are computed using Euclid's extended gcd algorithm such that $\ell_j a_j + (c'_j - c''_j) b_j = 1$.

By rewriting the equations we get

$$\begin{aligned} [r_j''^{-1} \star r'_j] &= [r_j''^{-1}] \otimes [r'_j] \\ &= \left(\otimes_{i=n}^1 z_i^{-c''_i} \right) \otimes t_j''^{-1} \otimes t'_j \otimes \left(\otimes_{i=1}^n z_i^{c'_i} \right) \\ &= \left(\otimes_{i=n}^j z_i^{-c''_i} \right) \otimes \left(\otimes_{i=j}^n z_i^{c'_i} \right) \\ &= z_j^{c'_j - c''_j}, \end{aligned}$$

where for obtaining the last equality we used the commutative property of \mathbb{H} . Thus,

$$\begin{aligned} [\tilde{x}_j] &= [u_j^{a_j} \star (r_j''^{-1} \star r'_j)^{b_j}] \\ &= [u_j]^{a_j} \otimes ([r_j''^{-1} \star r'_j])^{b_j} \\ &= (z_j^{\ell_j})^{a_j} \otimes (z_j^{c'_j - c''_j})^{b_j} \\ &= z_j^{\ell_j a_j + (c'_j - c''_j) b_j} \\ &= z_j. \end{aligned}$$

Now we show that at least half the vertices in at least one of the levels in T must be heavy. Let α_i be the ratio between the number of vertices at level $i+1$ and the number of vertices at level i in T . If $\alpha_i \leq (1/2s)s^n$ for all $1 \leq i \leq m$, then the total number of leaves in T (which is the product of all these α_i) is bounded by $(1/2s)^m s^{nm}$, which is a negligible fraction of the s^{nm} possible leaves. Since we assume that this fraction is polynomial, $\alpha_i > (1/2s)s^n$ for at least one i , and thus at least half the vertices at this level must contain more than s^n/s sons.

To find a heavy vertex in T , K chooses polynomially many random vertices at each level, and determines their degrees by repeated resets and executions of \bar{P} . To ensure a uniform probability distribution in spite of the uneven degrees of the vertices, M should explore random paths in the untruncated tree, and restart from the root whenever the path encounters an improperly answered query. Since a non-negligible fraction of the leaves is assumed to survive the truncation, this blind exploration of T can be carried out in polynomial time.

The last part of the proof deals with the zero-knowledge aspect of the protocol. By using resettable simulation in the sense of [10], the simulator S described in Algorithm 1 can mimic the communication in (P, \bar{V}) with an indistinguishable probability distribution in $O(ms^n)$ expected time, which is polynomial by our assumptions on s^n .

□

Algorithm 1: The simulator S .

Input: The public key $\{z_i\}_{i \in [1, n]}$
Output: A transcript \mathcal{L}

- 1 **foreach** $j \in [1, m]$ **do**
- 2 Choose $c = \{c_i\}_{i \in [1, n]}$ at random from C^n
- 3 Select a random number $r \xleftarrow{\$} \mathbb{G}$
- 4 Compute $t \leftarrow [r] \otimes (\otimes_{i=1}^n z_i^{-c_i})$
- 5 Call \bar{V} with input t and obtain a challenge c'
- 6 **if** $c = c'$ **then**
- 7 $L \leftarrow L \cup \{(t, c, r)\}$
- 8 **end**
- 9 **else**
- 10 Reset \bar{V} 's state and repeat this round with new random choices
- 11 **end**
- 12 **end**
- 13 **return** \mathcal{L}

4 Special Cases of the UGZK protocol

In this section we describe a number of protocols as instantiations of our main UGZK construction. Note that when $n = 1$ we obtain the UZK protocol from [13]. Thus, some schemes described in [13] are further reconsidered, while some examples are specific to our UGZK protocol. Although in the original paper [13] Maurer shows how to use UZK to prove the knowledge of a vector of secrets, our protocol UGZK is better in terms of transcript size.

4.1 Proofs of Knowledge of a Multiple Discrete Logarithm

Let $p = 2q + 1$ be a prime number such that q is also prime. Select an element $h \in \mathbb{H}_p$ of order q in some multiplicative group of order $p - 1$. The multiple discrete logarithm of a vector $\{z_i\}_{i \in [1, n]} \in \mathbb{H}_p^n$ is a vector of exponents $\{x_i\}_{i \in [1, n]}$ such that $z_i = h^{x_i}$, for all $i \in [1, n]$. We further describe a protocol for proving the knowledge of a multiple discrete logarithm.

A protocol for proving knowledge of a multiple discrete logarithm can be obtained as a special case of UGZK where $(\mathbb{G}, \star) = (\mathbb{Z}_q, +)$ and $\mathbb{H} = \langle h \rangle$. The one-way group homomorphism is defined by $[x] = h^x$, while the challenge space \mathcal{C} can be any arbitrary subset of $[0, q - 1]$. The conditions of Theorem 2 are satisfied for $\ell_j = q$ and $u_j = 0$, where $j \in [1, n]$. When $n = 1$ we obtain the Schnorr protocol [18]⁶. In the case $n \geq 1$ and $\mathcal{C} = \{0, 1\}$ we obtain the multiple logarithm protocol described in [1].

Next we discuss a variation⁶ of the previously presented protocol. Let $p = 2fp' + 1$ and $q = 2fq' + 1$ be prime numbers such that f, p' and q' are distinct primes. Select an element $h \in \mathbb{Z}_N^*$ of order f , where $N = pq$. Note that p and q are secret.

Using the UGZK notations we have $(\mathbb{G}, \star) = (\mathbb{Z}_f, +)$ and $\mathbb{H} = \langle h \rangle$. The one-way group homomorphism is defined by $[x] = h^x$ and the challenge space \mathcal{C} can be any arbitrary subset of $[0, f - 1]$. We can observe that the conditions of Theorem 2 are satisfied for $\ell_j = f$ and $u_j = 0$, where $j \in [1, n]$. When $n = 1$ we obtain the Girault protocol [8].

4.2 Proofs of Knowledge of a Multiple e^{th} -root

Let p and q be two large prime numbers. Compute $N = pq$ and choose a prime e such that $\gcd(e, \varphi(N)) = 1$. A multiple e^{th} -root of a vector $\{z_i\}_{i \in [1, n]} \in (\mathbb{Z}_N^*)^n$ is a base vector $\{x_i\}_{i \in [1, n]}$ such that $z_i \equiv x_i^e \pmod{N}$.

⁶This proof can be seen as a more efficient version of a proposal made by Chaum *et al.* [1].

Note that the multiple e^{th} -root is not unique. We further describe a protocol for proving the knowledge of a multiple e^{th} -root.

Such a protocol can be obtained from UGZK with $(\mathbb{G}, \star) = (\mathbb{H}, \otimes) = (\mathbb{Z}_N^*, \cdot)$. The one-way group homomorphism is defined by $[x] = x^e \bmod N$ and the challenge space \mathcal{C} can be any arbitrary subset of $[0, e - 1]$. The conditions of Theorem 2 are satisfied for $\ell_j = e$ and $u_j = z$, where $j \in [1, n]$. We stress that when $e = 2$ we obtain the protocol introduced by Feige, Fiat and Shamir [3]. In the case $n = 1$ we obtain the Guillou-Quisquater protocol [11]⁷.

4.3 Proofs of Knowledge of a Multiple Discrete Logarithm Representation

Let $p = 2q + 1$ be a prime number such that q is also prime. Select α elements $\{h_j\}_{j \in [1, \alpha]} \in \mathbb{H}_p^\alpha$ of order q in some multiplicative group of order $p - 1$. A multiple discrete logarithm representation of a vector $\{z_i\}_{i \in [1, n]} \in (\langle h_1, \dots, h_\alpha \rangle)^n$ is a vector of exponent vectors $(\{x_{1,j}\}_{j \in [1, \alpha]}, \dots, \{x_{n,j}\}_{j \in [1, \alpha]})$ such that $z_i = h_1^{x_{i,1}} \dots h_\alpha^{x_{i,\alpha}}$, for all $i \in [1, n]$. Note that multiple discrete logarithm representations are not unique. We further describe a protocol for proving the knowledge of a multiple discrete logarithm representation.

A protocol for proving the knowledge of a multiple representation can be instantiated from UGZK by setting $\mathbb{G} = \mathbb{Z}_q^\alpha$ with \star defined as a component-wise addition operation and $\mathbb{H} = \langle h_1, \dots, h_\alpha \rangle$. The one-way group homomorphism is defined by $[(x_1, \dots, x_\alpha)] = h_1^{x_1} \dots h_\alpha^{x_\alpha}$ and the challenge space \mathcal{C} can be any arbitrary subset of $[0, q - 1]$. The conditions of Theorem 2 are satisfied for $\ell_j = q$ and $u_j = (0, \dots, 0)$, where $j \in [1, n]$. When $n = 1$ we obtain a protocol proposed by Maurer in [13] which is a generalization of the protocols presented by Okamoto in [15] and Chaum *et.al.* in [1].

Chaum *et al.* [1] also provide a protocol variant for a composite n . Thus, by adapting the protocol presented in Section 4.1 and tweaking the previously described one, we can obtain a similar version for composite numbers. Using the notations from the protocol in Section 4.1, we set $\mathbb{G} = \mathbb{Z}_f^\alpha$ and $\mathbb{H} = \langle h_1, \dots, h_m \rangle$, where $h_1, \dots, h_\alpha \in \mathbb{Z}_n^*$ are elements of order f . The one-way group homomorphism is defined by $[(x_1, \dots, x_\alpha)] = h_1^{x_1} \dots h_\alpha^{x_\alpha}$ and the challenge space \mathcal{C} can be any arbitrary subset of \mathbb{Z}_f . It is easy to see that $\ell_j = f$ and $u_j = (0, \dots, 0)$, where $j \in [1, n]$.

4.4 Proofs of Knowledge of a Multiple e^{th} -root Representation

Let p and q be two large prime numbers. Compute $N = pq$ and choose primes e_1, \dots, e_α such that $\gcd(e_i, \varphi(N)) = 1$, for $i \in [1, \alpha]$. A multiple e^{th} -root representation of a vector $\{z_i\}_{i \in [1, n]} \in (\mathbb{Z}_N^*)^n$ is a vector of bases vector $(\{x_{1,j}\}_{j \in [1, \alpha]}, \dots, \{x_{n,j}\}_{j \in [1, \alpha]})$ such that $z_i \equiv x_{i,1}^{e_1} \dots x_{i,\alpha}^{e_\alpha} \bmod N$, for all $i \in [1, n]$. Note that multiple e^{th} -root representations are not unique. We further describe a protocol for proving the knowledge of a multiple e^{th} -root representation.

A protocol for proving the knowledge of a multiple e^{th} -root representation can be obtained from UGZK if we set $\mathbb{G} = (\mathbb{Z}_N^*)^\alpha$ with \star defined as multiplication applied component-wise and $(\mathbb{H}, \otimes) = (\mathbb{Z}_N^*, \cdot)$. The one-way group homomorphism is defined by $[(x_1, \dots, x_\alpha)] = x_1^{e_1} \dots x_\alpha^{e_\alpha} \bmod N$ and the challenge space \mathcal{C} can be any arbitrary subset of $[0, e - 1]$, where e is a prime such that $\gcd(e, \phi(N)) = 1$. Since all e_i are coprime then there exist β_i s such that $\beta_1 e_1 + \dots + \beta_\alpha e_\alpha = 1$. Then, it is easy to see that $\ell_j = 1$ and $u_j = (z_j^{\beta_1}, \dots, z_j^{\beta_\alpha})$, where $j \in [1, n]$. When $n = 1$ we obtain a protocol introduced in [19].

5 Hash Protocol Variant

In order to decrease the number of communication bits, *Peggy* can hash t and send *Victor* the result. This method was proposed by Fiat and Shamir [7] and later analyzed in [9]. We employ the same technique for the protocol presented in Figure 2 and analyze its security.

⁷This proof is a generalization of a protocol introduced by Fiat and Shamir [7].

5.1 Description

Let H be a hash function that maps elements from \mathbb{H} into bit streams. The hash variant of the protocol works as follows: in the first step *Peggy* sends $H(t)$ to *Victor* (instead of t) and the last step becomes

If $H(t) = H\left([r] \otimes \left(\otimes_{i=1}^n z_i^{-c_i}\right)\right)$ return **true**.

Else return **false**.

5.2 Security Analysis

Theorem 3. *Let $s = |\mathcal{C}|$. If there exists a PPT algorithm \tilde{P} such that the probability that \tilde{P} is accepted by an honest verifier is greater than $(\lambda - 1)|\mathcal{C}|^{-n} + \varepsilon$, where $\varepsilon > 0$, then there exists a PPT algorithm \hat{P} which, with overwhelming probability, either inverts $[\cdot]$ or finds a λ -collision for h .*

Proof. Let Ω be the set of \tilde{p} elements in which \tilde{P} picks its random values and E be the set \mathcal{C}^n , both of them characterized by the uniform distribution. For each value $(\omega, e) \in \Omega \times E$, \tilde{P} passes the protocol (and we say it is a success) or not. Let S be the subset $\Omega \times E$ composed of all possible successes. Our assumption is that

$$\frac{|S|}{|\Omega \times E|} > (r - 1)|\mathcal{C}|^{-n} + \varepsilon$$

with $\varepsilon > 0$ and $|\Omega \times E| = \tilde{p} \cdot s^n$.

Let $E_r = \{e \in E \mid (\omega, e) \text{ is a success}\}$ and $\Omega_r = \{\omega \in \Omega \mid |E_r| \geq r\}$. We have that

$$|S| \leq |\Omega_r| \cdot s^n + (r - 1) \cdot (\tilde{p} - |\Omega_r|).$$

Thus,

$$\frac{|S|}{|\Omega \times E|} \leq \left[\frac{|\Omega_r|}{|\Omega|} + (r - 1) \cdot \left(s^{-n} - \frac{|\Omega_r|}{|\Omega \times E|} \right) \right] \leq \frac{|\Omega_r|}{|\Omega|} + (r - 1) \cdot s^{-n}$$

which implies

$$\frac{|\Omega_r|}{|\Omega|} \geq \varepsilon.$$

Let \hat{P} be the PPT algorithm obtained by resetting \tilde{P} ε^{-1} times. With constant probability, \hat{P} picks ω in Ω_r and the probability can be made close to 1 by repeating the execution of \hat{P} . At the end, λ values $\{r_i\}_{i \in [1, \lambda]}$ are found such that, for distinct challenges $\{c_i\}_{i \in [1, \lambda]} \in (\mathcal{C}^n)^\lambda$

$$H\left([r_1] \otimes \left(\otimes_{i=1}^n z_i^{-c_{i,1}}\right)\right) = H\left([r_2] \otimes \left(\otimes_{i=1}^n z_i^{-c_{i,2}}\right)\right) = \dots = H\left([r_\lambda] \otimes \left(\otimes_{i=1}^n z_i^{-c_{i,\lambda}}\right)\right).$$

Now, we have two possibilities. In the first case, two of the values, say $[r_1] \otimes \left(\otimes_{i=1}^n z_i^{-c_{i,1}}\right)$ and $[r_2] \otimes \left(\otimes_{i=1}^n z_i^{-c_{i,2}}\right)$, are equal before hashing. Let $\mathcal{C}^- = \{-c \mid c \in \mathcal{C}\}$. Then, $[r_1 r_2^{-1}] = \left(\otimes_{i=1}^n z_i^{c'_i}\right)$, where $c' \in \mathcal{C}^- \cup \mathcal{C}$. This contradicts the intractability of $[\cdot]$. In the second case, all these values are pairwise distinct and a λ -collision for H has been found. This contradicts our assumption regarding H . \square

Remark 2. This result suggests the use of hash-functions which are only resistant to λ -collisions (with $\lambda > 2$), such that the hash values computed in the first pass can be made much shorter. Indeed, the decrease of the security level can be balanced by sending a slightly larger value of c in the second pass. More precisely, if $\lambda = s^{n'}$, we choose $c \in \mathcal{C}^{n+n'}$ instead of $c \in \mathcal{C}^n$.

6 A Distributed Unified Protocol

A Fiat-Shamir-like distributed authentication protocol was proposed in [2]. Given our UGZK construction, we describe a generic collective authentication protocol which can be seen as a natural follow up of the main result in [2].

6.1 Description

Let us consider an n -node network consisting of $\mathcal{N}_1, \dots, \mathcal{N}_n$. The nodes \mathcal{N}_i can be seen as users and the base station \mathcal{T} as a trusted center. To achieve the authentication of the entire network, we propose a unified Fiat-Shamir-like construction which we detail next.

1. Let x_i be a secret piece of information given to node \mathcal{N}_i . First, the network topology has to converge and a spanning tree needs to be constructed (e.g. with an algorithm similar with the one presented in [14]). Then, \mathcal{T} sends an authentication request message to all the \mathcal{N}_i s directly connected to it, a message which contains a commitment to c (see 3.) to ensure the protocol's zero-knowledge property even against dishonest verifiers.
2. After receiving an authentication request message:
 - Each \mathcal{N}_i generates a private k_i and computes $t_i \leftarrow [k_i]$;
 - The \mathcal{N}_i s send authentication messages to all their (existing) children;
 - After the children respond, nodes \mathcal{N}_i compute $t_i \leftarrow t_i \otimes (\otimes_j t_j)$ and send the result up to their parents. Note that the t_j s are sent by the nodes' children.

Such a construction permits the network to compute the \otimes operation of all the t_i s and send the result t_c to the top of the tree in d steps, where d represents the degree of the spanning tree. We refer the reader to Figure 3 for a toy example of this step.

3. \mathcal{T} sends a random $c \in \mathcal{C}^n$ as an authentication challenge to the \mathcal{N}_i directly connected to it.
4. After receiving an authentication challenge c :
 - Each \mathcal{N}_i computes $r_i \leftarrow k_i \star x_i^{c_i}$;
 - The \mathcal{N}_i s then send the authentication challenge to all their (existing) children;
 - After the children respond, the \mathcal{N}_i s compute $r_i \leftarrow r_i \star (\star_j r_j)$ and send the result to their parents. Note that the r_j s are sent by the nodes' children.

The network therefore computes collectively the \star operation of all the r_i 's and transmits the result r_c to \mathcal{T} . Again, we refer the reader to Figure 3 for a toy example of this step.

5. After receiving r_c , \mathcal{T} checks that $[r_c] = t_c \otimes (\otimes_{i=1}^n z_i^{c_i})$, where z_1, \dots, z_n are the public keys corresponding to x_1, \dots, x_n respectively.

Remark 3. The protocol we have just described may be interrupted at any step and such an action results in a failed authentication.

6.2 Security Analysis

Theorem 4. Let \mathbb{H} be a commutative group and let $j \in [1, n]$. If an adversary corrupts $n' < n$ nodes and if values $\ell_j \in \mathbb{Z}$ and $u_j \in \mathbb{G}$ are known such that

- $\gcd(c'_j - c''_j, \ell_j) = 1$ for all $c'_j, c''_j \in \mathcal{C}$ with $c'_j \neq c''_j$,
- $[u_j] = z_j^{\ell_j}$,

then by running the protocol described in Section 6.1 for m rounds we obtain a proof of knowledge protocol if $1/|\mathcal{C}|^{(n-n')m}$ is negligible, and a zero-knowledge protocol if $|\mathcal{C}|^{(n-n')}$ is polynomially bounded.

Proof. If an adversary corrupts n' nodes, then n' secret keys x_i are known to him. Thus, the protocol is equivalent with a UGZK protocol with $n - n'$ secrets. Hence, using Theorem 2 we obtain our statement. \square

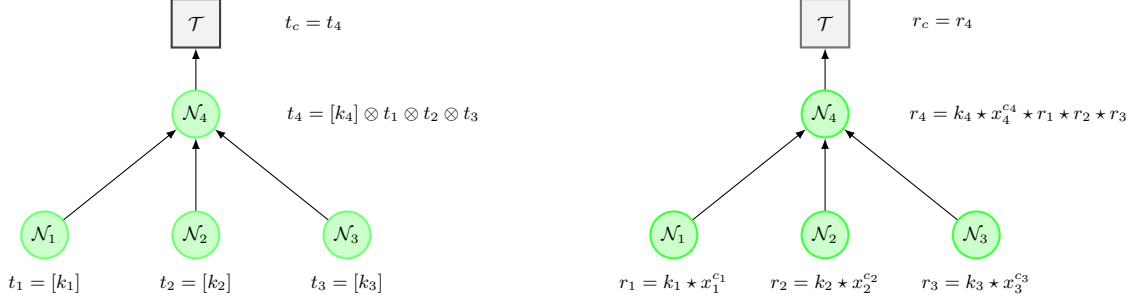


Fig. 3. The proposed algorithm running on a network consisting of 4 nodes: computation of t_c (left) and of r_c (right).

6.3 Complexity Analysis

The number of operations necessary for authenticating the whole network depends on the topology. Precise complexity evaluations are given in Table 1. Note that each node performs in average only a few operations (a constant number).

Operation	Number of computations
$[\cdot]$	$(n + 1)m$
Exponentiation	nm
\otimes	$\leq 2nm$
\star	$\leq 2nm$

Table 1. Complexity Computations.

Let d be the degree of the minimum spanning tree of the network. Then, only $O(d)$ messages are sent and, if we do not consider atypical cases, $d = O(\log n)$. Put differently, throughout the authentication process only a logarithmic number of messages is sent.

6.4 Variations

When implementing the distributed zero knowledge protocol several trade-offs are possible. Note that when doing so any combination of the trade-offs described below may be used.

Hash based variant. A distributed version of the UGZK protocol’s hash variant (presented in Section 5) can be constructed. Using this “short commitment” version reduces somewhat the number of communicated bits, at the expense of a reduced security.

Short challenges variant. In our protocol, the challenge c is sent throughout the network to all nodes. Assuming the use of an ideal hash function h , we may use shorter challenge without affecting security.

- A short c is sent to the nodes \mathcal{N}_i ;
- Each \mathcal{N}_i computes $c_i \leftarrow h(c||i)$, and uses c_i as a challenge;
- The base station \mathcal{T} computes c_i and uses it to check authentication.

Multiple-secret variant. Each node \mathcal{N}_i could use a set of secret values $\{x_{i,j}\}_{j \in [1,\ell]}$ instead of only one x_i . For the algorithm to be as efficient as possible the supplementary secrets can be expanded from a concealed seed. For clarity purposes we describe the multiple secret variant for a single node.

When receiving a challenge c_i , each node computes a response

$$r_i \leftarrow k_i \star \left(\star_{j=1}^{\ell} x_{i,j}^{c_{i,j}} \right).$$

This result be checked by the verifier by applying the next formula:

$$[r_i] = t_i \otimes \left(\otimes_{i=1}^{\ell} z_{i,j}^{c_{i,j}} \right).$$

In the case of multiple nodes, the modified protocol we obtain is a proof of knowledge if $1/|\mathcal{C}|^{(n-n')\ell m}$ is negligible and a zero-knowledge protocol if $|\mathcal{C}|^{(n-n')\ell}$ is polynomially bounded.

Practical aspects. Applying the multiple-secret variant, the trade-off between memory and communication can be adjusted, as the security level is ℓm (single-node compromission). Let μ be an integer. Therefore, if $\ell = \mu$ it suffices to authenticate once to get the same security as $t = \mu$ authentications with $\ell = 1$ ⁸. It is obvious that such an approach significantly reduces bandwidth usage, a clearly desirable fact in the IoT context.

7 Conclusions and Further Development

We proposed a UGZK protocol and analyzed its security. We provided various special cases of our core protocol, described a hash variant of UGZK and discussed security details. We also presented a distributed unified Fiat-Shamir-based protocol, tackled security and complexity aspects and presented implementation trade-offs.

Future work. In order to take advantage of our main protocol's characteristics, we suggest applying it for obtaining generic versions of digital signature schemes [12, 16, 17] and legally fair contract signing protocols [4, 12]. More generally, our proposal could be useful for future works on cryptographic protocol design. In the case of failed network authentication an interesting research direction would be to devise new batch verification algorithms or adapt the ones constructed for digital signatures [5, 6] for finding compromised nodes.

References

1. Chaum, D., Evertse, J.H., Van De Graaf, J.: An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some Generalizations. In: EUROCRYPT 1987. Lecture Notes in Computer Science, vol. 304, pp. 127–141. Springer (1987)
2. Cogliani, S., Feng, B., Ferradi, H., Géraud, R., Maimuț, D., Naccache, D., do Canto, R.P., Wang, G.: Public Key-Based Lightweight Swarm Authentication. In: Cyber-Physical Systems Security, pp. 255–267. Springer (2018)
3. Feige, U., Fiat, A., Shamir, A.: Zero-Knowledge Proofs of Identity. Journal of cryptology **1**(2), 77–94 (1988)
4. Ferradi, H., Géraud, R., Maimuț, D., Naccache, D., Pointcheval, D.: Legally Fair Contract Signing Without Keystones. In: International Conference on Applied Cryptography and Network Security - ACNS'16. Lecture Notes in Computer Science, vol. 9696, pp. 175–190. Springer (2016)
5. Fiat, A.: Batch RSA. In: CRYPTO 1989. Lecture Notes in Computer Science, vol. 435, pp. 175–185. Springer (1989)
6. Fiat, A.: Batch RSA. J. Cryptology **10**(2), 75–88 (1997)
7. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: CRYPTO 1986. Lecture Notes in Computer Science, vol. 263, pp. 186–194. Springer (1986)

⁸This corresponds to the protocol presented in Section 6.1.

8. Girault, M.: An Identity-based Identification Scheme Based on Discrete Logarithms Modulo a Composite Number. In: EUROCRYPT 1990. Lecture Notes in Computer Science, vol. 473, pp. 481–486. Springer (1990)
9. Girault, M., Stern, J.: On the Length of Cryptographic Hash-Values Used in Identification Schemes. In: CRYPTO 1994. Lecture Notes in Computer Science, vol. 839, pp. 202–215. Springer (1994)
10. Goldwasser, S., Micali, S., Rackoff, C.: The Knowledge Complexity of Interactive Proof Systems. *SIAM J. Comput.* **18**(1), 186–208 (1989)
11. Guillou, L.C., Quisquater, J.J.: A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory. In: EUROCRYPT 1988. Lecture Notes in Computer Science, vol. 330, pp. 123–128. Springer (1988)
12. Maimuț, D., Teșeleanu, G.: A Unified Security Perspective on Legally Fair Contract Signing Protocols. In: SECITC 2018. Lecture Notes in Computer Science, vol. 11359, pp. 477–491. Springer (2018)
13. Maurer, U.: Zero-Knowledge Proofs of Knowledge for Group Homomorphisms. *Designs, Codes and Cryptography* **77**(2-3), 663–676 (2015)
14. Mooij, A.J., Goga, N., Wesselink, J.W.: A Distributed Spanning Tree Algorithm for Topology-Aware Networks. Technische Universiteit Eindhoven, Department of Mathematics and Computer Science (2003)
15. Okamoto, T.: Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. In: CRYPTO 1992. Lecture Notes in Computer Science, vol. 740, pp. 31–53. Springer (1992)
16. Pointcheval, D., Stern, J.: Security Proofs for Signature Schemes. In: *Advances in Cryptology - EUROCRYPT'96*. Lecture Notes in Computer Science, vol. 1070, pp. 387–398. Springer (1996)
17. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology* **13**(3), 361–396 (2000)
18. Schnorr, C.P.: Efficient Identification and Signatures For Smart Cards. In: CRYPTO 1989. Lecture Notes in Computer Science, vol. 435, pp. 239–252. Springer (1989)
19. Teșeleanu, G.: Unifying Kleptographic Attacks. In: NordSec 2018. Lecture Notes in Computer Science, vol. 11252, pp. 73–87. Springer (2018)