Sherlock Holmes Zero-Knowledge Protocols Secure Against Active Attackers

George Teşeleanu

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

Abstract. We present two simple zero knowledge interactive proofs that can be instantiated with many of the standard decisional or computational hardness assumptions. Compared with traditional zero knowledge proofs, in our protocols the verifier starts first, by emitting a challenge, and then the prover answers the challenge.

Keywords: zero-knowledge protocols, sequential attacks, active intruder attacks

1 Introduction

A standard interactive proof of knowledge involves a prover, usually called P or $Peggy$, and a verifier, usually called V or $Victor$. $Peggy$ is in possession of some secret k and by interacting with $Victor$ she wants to convince him that she indeed owns k. More formally, an interactive proof is a pair of programs that implement the protocol between $Peggy$ and $Victor$. To be useful, such a proof must be complete and sound. By complete we mean that an honest $Peggy$ succeeds in convincing an honest $Victor$ and by sound we mean that a dishonest prover does not succeed in convincing the verifier of a false statement. Moreover, if $Victor$ does not learn anything from the protocol's execution which he did not know before, we say that the protocol is zero knowledge.

In a classical zero knowledge protocol, $Peggy$ starts the protocol by sending a commitment to $Victor$, then $Victor$ sends a challenge to $Peggy$ and finally $Peggy$ sends her answer. The verifier will accept the proof if and only if $Peggy$'s answer coincides with the answer he expects. In contrast with these protocols, the authors of [19] introduce a new class of protocols[1] in which $Victor$ starts the protocol. Once the verifier knows that $Peggy$ wants to start the protocol[2], he issues a challenge to which $Peggy$ answers. If the answer is correct, then the protocol ends successfully. Otherwise, it fails.

[1] based on what the authors of [19] call the Sherlock Holmes method

[2] *e.g.* $Peggy$ can send a "hello" type message or $Victor$ can be equipped with motion sensors and detect $Peggy$'s proximity

1.1 Attack Models for Identification Protocols

When proposing novel interactive proof of knowledge protocols, we must be able to prove their security against various types of adversaries. This includes security against legitimate, but malicious users of the protocol. We say that an adversary is successful if he manages to impersonate the prover with a non-negligible probability. We usually think of the verifier as an adversary trying to cheat [16], since the zero-knowledge property should hold for any strategy employed by the verifier to gain some information about the secret knowledge held by the prover.

Adversaries. The weakest type of adversary is the one that simply eavesdrops on the communication between the prover and the verifier. Another type of adversary is the "honest" verifier. This attacker interacts with the prover according to the protocol, but he maintains a database with all the protocol transcripts and all the associated data[3] generated by him during the protocol.

A stronger notion is the so called "impersonation" attacker [4]. In this model, the attacker first plays the role of the verifier and interacts with the prover in different sessions, and then it tries to impersonate the prover. Depending on how the adversary interacts with the prover, impersonation adversaries split into three categories: sequential, parallel and concurrent (see below).

The last type of adversary that we consider is the active-intruder adversary [31]. This adversary is able to alter, inject, drop and/or divert at least one message in the given session. We say that the active-intruder adversary is successful if the verifier accepts the session after the adversary becomes active.

Sequential Attacks. In the case of sequential attacks, once an instance of the protocol is started, then that instance must be terminated before starting a new one [16]. This is the classical attack model for zero-knowledge protocols and is inspired by the smartcard communication model [32].

In the sequential attack model, the Feige-Fiat-Shamir [11] and Okamoto [25] protocols are secure as long as the square root and discrete logarithm problems are intractable. Although the Schnorr [29] and Guillou-Quisquater [21] protocols are proven secure [20, 30] in the honest verifier model under the discrete logarithm and e-th root problems, the protocols do not have a security proof in the sequential model under standard assumptions [4].

Parallel Attacks. Compared to sequential attacks, in the parallel case, many instances of the protocol are run at the same time and proceed at the same pace [16]. This model is inspired by the synchronous model of communication and considers a polynomial number of executions that are synchronized such that the i-th message is sent approximately at the same time. Note that in [16] we can find an example of a protocol that is secure in the sequential model, but insecure in the parallel one.

[3] *e.g.* coin tosses

Concurrent Attacks. These attacks generalize both the sequential and parallel attacks. In this case, a polynomial number of instances are run at arbitrary times and proceed at an arbitrary pace [16]. This model is inspired by the internet communication model [32].

According to [13, 25], the Feige-Fiat-Shamir and Okamoto protocols remain secure in this attack model. The authors of [4] show that the Schnorr and Guillou-Quisquater protocols can be proven secure in the concurrent model if stronger non-standard assumptions hold.

In the case of two-round protocols where the verifier starts, concurrent attacks are equivalent to sequential attacks [32]. This is because once the prover receives a challenge, he immediately responds and the protocol ends. Therefore, each reply is determined only by the corresponding challenge. Hence, since our proposals are two-round protocol, it is sufficient to study their security in the sequential attack model.

Active-Intruder Attack. In the concurrent scenario, the attacker is only allowed to interact with the prover before attempting to impersonate him. But in real-life scenarios, the adversary might be able to interact with the prover at the same time that the adversary impersonates him. This is a type of man-in-the-middle attack. In this setting we impose that the adversary alters, injects, drops and/or diverts at least one message to avoid attackers who simply relay messages faithfully. This is an attacks model proposed by Stinson [31].

In Appendix A we present an active-intruder attack against a family of zero knowledge protocols. This family includes some of the most popular identification schemes [22, 23]. Although it is controversial if these attacks could be categorized as "real" attacks [32], it is often desirable to design protocols that withstand the strongest possible attacks, as long as it does not result in substantial overhead.

Reset Attacks. This class of attacks was introduced in [2]. In this model the verifier can reset the prover, and thus forcing him to use the same random tape in multiple concurrent executions [16]. Such attacks were inspired by smartcards that can be controlled by the attacker or are in his possession. Therefore, even if the attacker cannot read the secret content contained in the secure hardware, he can disconnect the smartcard's battery and reset its internal state.

It is worth mentioning that most popular identification schemes, like Schnorr, Guillou-Quisquater, Feige-Fiat-Shamir and Okamoto protocols are not secure in this model [2].

According to [32] if the prover is stateless and deterministic, then the corresponding protocol is secure in this setting. Since our proposals use exactly this type of prover, it follows that they are secure against reset attacks. The most powerful security model is the combination of active-intruder attacks and reset attacks [36]. According to the above arguments, in the case of our proposals all we need to prove is that they are secure against active-intruder attacks.

1.2 Our Contributions

Although Grigoriev and Shpilrain's protocol [19] is very interesting, the authors only claim that their protocol is zero knowledge in the honest verifier scenario without actually proving it. To fill this gap, we re-formalize and generalize Grigoriev and Shpilrain's protocol and then we prove its security in the same scenario. Moreover, we provide active-intruder attacks that can be mounted against this protocol. A downside of this formalization, is that $Victor$ must iterate the protocol a number of times in order to fulfill the soundness property. By vectorizing the protocol we manage to reduce the number of iteration to one. Additionally, we provide a variation of the vectorized protocol that is secure in the sequential and active-intruder attack scenarios.

To further improve our protocol, we modified it by changing the underlying assumption from a decisional one to a computational one. This was necessary in order to reduce the bandwith requirements necessary for the decisional version. Note that if $Peggy$ and $Victor$ choose the right parameters the new protocol will provide the same security assurances. Furthermore, we introduce two variations that are secure in the sequential and active-intruder attack scenarios.

Finally, we offer the reader several concrete realizations of our protocols and compare them with classical zero knowledge protocols such as Schnorr [29], Guillou-Quisquater [21] and Fiat-Shamir [14]. Note that one can devise new instantiations of our protocols.

We remark that in the case of our protocols the verifier knows with overwhelming probability the answer given by the prover. This is not the case for classical protocols, since the verifier does not know the reply in advance.

Previous work. Note that a preliminary version of this paper was presented in [35].

Structure of the paper. We introduce notations and definitions used throughout the paper in Section 2. Inspired by Grigoriev and Shpilrain's protocol, in Section 3 we formalize and analyse the Multi-Decisional Sherlock Holmes (MDSH) protocol. A vectorized version of MDSH and a variant of it are presented in Sections 4 and 5. The computational version and its variants are tackled in Sections 6 to 8. Section 9 contains a comparison with classical zero knowledge protocols. We conclude in Section 10.

2 Preliminaries

Notations. Throughout the paper, the notation $|S|$ denotes the cardinality of a set S. The action of selecting a random element x from a sample space X is denoted by $x \xleftarrow{\$} X$, while $x \leftarrow y$ represents the assignment of value y to variable x. The probability of the event E to happen is denoted by $Pr[E]$. The subset $\{0, \ldots, s-1\} \in \mathbb{N}$ is denoted by $[0, s]$. A vector v of length n is denoted either $v = (v_0, \ldots, v_{n-1})$ or $v = \{v_i\}_{i \in [0,n]}$ and $v_1 = v_2$ stands for element-wise equality between two vectors v_1 and v_2.

2.1 Hardness Assumptions

Inspired by the computational and decisional hardness assumptions described in [5] and the one way function definitions found in [3, 27], we further provide the reader with the following two definitions. The first one captures the idea of a generic computational hardness assumption, while the second the decisional version. We do not claim to capture all the generic hardness assumptions, but for our purpose these definitions suffice. Note that when we define an advantage, we use ";" to denote the end of simple instructions or for loops and "," to denote the end of an instruction inside a for loop.

Definition 1 (Computational Hardness Assumption). *Let $K \subseteq \{0,1\}^*$ be a family of indices and for $k \in K$ let $D_k, R_k \subseteq \{0,1\}^*$. A computational hard function f is a parameterized family of functions $f_k : D_k \to R_k$ such that*

1. *for every key $k \in K$ there exists a PPT algorithm that on input $x \in D_k$ outputs $f_k(x)$;*
2. *for every PPT algorithm A the advantage*

$$ADV_f^{\text{CHA}}(A) = Pr[f_k(z) = y \mid k \xleftarrow{\$} K; x \xleftarrow{\$} D_k; y \leftarrow f_k(x); z \leftarrow A(f_k, y)]$$

   is negligible;
3. *there exists a PPT algorithm B such that*

$$Pr[f_k(z) = y \mid k \xleftarrow{\$} K; x \xleftarrow{\$} D_k; y \leftarrow f_k(x); z \leftarrow B(k, y)] = 1.$$

Definition 2 (Decisional Hardness Assumption). *A function f is a decisional hard function if in Definition 1, Item 2 and 3 are changed to*

2. *for every PPT algorithm A the advantage*

$$ADV_f^{\text{DHA}}(A) = |2Pr[b = b' \mid k_0, k_1 \xleftarrow{\$} K; b \xleftarrow{\$} \{0,1\}; \\ x \xleftarrow{\$} D_{k_b}; y \leftarrow f_{k_b}(x); b' \leftarrow A(f_{k_0}, f_{k_1}, y)] - 1|$$

   is negligible;
3. *there exists a PPT algorithm B such that*

$$Pr[b = b' \mid k_0, k_1 \xleftarrow{\$} K; b \xleftarrow{\$} \{0,1\}; \\ x \xleftarrow{\$} D_{k_b}; y \leftarrow f_{k_b}(x); b' \leftarrow B(k_0, k_1, y)] = 1.$$

We further provide a security assumption from [5] that will be useful later on.

Definition 3 (Pseudo-Random Permutation - PRP). *A function $\pi : \{0,1\}^\delta \times \{0,1\}^\tau \to \{0,1\}^\tau$ is a PRP if:*

- *Given a key $K \in \{0,1\}^\delta$ and an input $X \in \{0,1\}^\tau$ there is an efficient algorithm to compute $\pi_K(X) = \pi(X, K)$.*

- *Given a key* $K \in \{0,1\}^{\delta}$ *the function* $\pi_K(\cdot)$ *is one-to-one.*
- *Let* A *be a PPT algorithm with access to an oracle* $\mathcal{O}$ *that returns* 1 *if* $\mathcal{O} = \pi_K(\cdot)$*. The* PRP*-advantage of* A*, defined as*

$$ADV_{\pi}^{\text{PRP}}(A) = \left| Pr[A^{\pi_K(\cdot)} = 1 | K \xleftarrow{\$} \{0,1\}^{\delta}] - Pr[A^{F(\cdot)} = 1 | F \xleftarrow{\$} \mathcal{F}] \right|$$

must be negligible for any PPT algorithm A*, where* $\mathcal{F} = \{F : \{0,1\}^{\tau} \rightarrow \{0,1\}^{\tau} \mid F \text{ is one-to-one}\}$*.*

2.2 Zero-Knowledge Protocols

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

We further base our reasoning on two definitions from [11, 12, 23] which we recall next.

Definition 4 (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 $Q(z,x) = \texttt{true}$;*
- **Soundness***: there exists an efficient program E (called knowledge extractor) such that for any $\bar{P}$ (possibly dishonest) with non-negligible probability of making V accept the proof, E can interact with $\bar{P}$ and output (with overwhelming probability) an x such that $Q(z,x) = \texttt{true}$.*

Definition 5. *[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* $\bar{V}$*.*

Remark 1 (Negative Results). The first impossibility result for two-round zero knowledge proofs was initially presented in [18] and subsequently refined in [1]. More precisely, the authors of [1] proved that if a language L has a two-round public-coin[4] zero knowledge proof system that has an efficient prover, then L belongs to the complexity class **P**. If we consider private-coin proof systems, if **NP** $\neq$ **coNP** then L belongs to the complexity class **coNP**. Another negative result was proven in [17], which states that a language L has a constant-round public-coin zero knowledge proof system which is black-box simulation[5] zero knowledge if and only if belongs to the complexity class **BPP**. Note that the protocols presented in this paper are not public-coin.

[4] A public-coin proof system, or Arthur-Merlin game, is characterized by the verifier's strategy, which primarily involves sending random string messages, followed by a final decision to accept or reject the proof, with the verifier's coin tosses being public.

[5] In simple terms, a protocol is considered to be black-box zero knowledge when the zero-knowledge property is proven through a universal simulator that exclusively relies on black-box or oracle access to the verifier's strategy.

Remark 2 (Negative Results on Negative Results). Based on Damgård's knowledge-of-exponent assumption [10], Barak *et al.* [1] established the existence of a two-round private-coin zero-knowledge proof system for a promise problem that lies beyond **BPP**. Therefore, the negative result from [1] for **NP**-complete languages cannot be generalized to cover all nontrivial problems without challenging this assumption. The protocol introduced in [1] can be seen as a specialized version of a more generic protocol introduced in [28], which centers around deciding if two distributions are statistically "close" or "far apart". Sahai and Vadhan [28] further establish that, under the honest verifier scenario, their two-round private-coin protocols is a statistical[6] zero-knowledge proof system. Additionally, they prove that statistical zero-knowledge protocols are essentially those designed to decide whether a pair of efficiently samplable distributions exhibit statistical closeness or not. Independently, two-round private-coin protocols were introduced in [32, 36] based on the knowledge-of-exponent assumption. Additionally, in [37], another two-round protocol is presented, relying on the strong Diffie-Hellman assumption."

We further define **impersonation under concurrent attack** as presented in [4].

Definition 6 (Impersonation under Concurrent Attack - IMP-CA). *An* IMP-CA *adversary is a pair of PPT algorithms $A = (\bar{P}, \bar{V})$, where $\bar{P}$ and $\bar{V}$ are the cheating prover and verifier, respectively. In the first phase of the attack, a random tape is chosen for $\bar{V}$ and it receives as input z. Then $\bar{V}$ starts to interact concurrently with a polynomial number of clones of the honest prover P. Note that each clone knows an x such that $Q(z, x) =$ `true`. We further view P as a functions that takes as input an incoming message and the current state, and returns an outgoing message and the updated state. The cheating verifier $\bar{V}$ can issue two types of requests that can be arbitrarily interleaved. The first type of request is of the form (ε, i), and it leads to*

- *the initial state of clone i is set to $St_i \leftarrow (x, z, R_i)$, where R_i is a fresh random tape;*
- *the operation $(M_{out}, St_i) \leftarrow P(\varepsilon, St_i)$ is executed;*
- *M_{out} is returned to $\bar{V}$ and St_i is saved as the new state of clone i.*

The second type of request is (M, i) and it has the following effect

- *message M is sent to clone i;*
- *the operation $(M_{out}, St_i) \leftarrow P(M, St_i)$ is executed;*
- *M_{out} is returned to $\bar{V}$ and St_i is saved as the new state of clone i.*

After finishing the request phase, $\bar{V}$ outputs a state St and stops. In the second phase of the attack, the cheating prover $\bar{P}$ is initialized with St and starts to interact with a verifier V. Note that V is in possession of z and fresh random

[6] In Definition 5, we require that the output of S has negligible statistical difference from the real transcript, instead of computational indistinguishablility.

coins. We say that adversary A wins if V accepts $\bar{P}$'s proof. We say that an interactive protocol (P, V) is secure against concurrent impersonation attacks if for any IMP-CA *adversary the probability of winning $ADV^{\text{IMP-CA}}_{P,V}(A)$ is negligible.*

Finally, we provide a definition from [31] that captures active-intruder attacks.

Definition 7 (Active-Intruder Attack). *An active-intruder is successful if the verifier accepts in a session after the adversary becomes active (*i.e. *injects, drops and/or diverts at least one message) in the same session.*

3 Multi-Decisional Protocol

3.1 Description

Based on a variation of decisional hard functions, we further describe a protocol (see Figure 1) that allows *Peggy* to prove to $Victor$ that she is in possession of some secrets. When $Victor$ knows that *Peggy* is ready to start the protocol, he sends her a challenge and *Peggy* responds with her guess. If the guess is correct, then $Victor$ accepts the answer.

Peggy		*Victor*
Knows k_i, for $i \in [0, n]$		Knows f_{k_i}, for $i \in [0, n]$
		Choose $i \xleftarrow{\$} [0, n]$
		Choose $x \xleftarrow{\$} D_{k_i}$
		Compute $y \leftarrow f_{k_i}(x)$
	$\xleftarrow{y}$	
$i' \leftarrow -1$		
For $j \in [0, n]$		
If $y \in R_{k_j}$ then $i' \leftarrow j$		
If $i' = -1$ then `abort`		
	$\xrightarrow{i'}$	
		If $i' = i$ return `true`
		Else return `false`

Fig. 1. Multi-Decisional Sherlock Holmes (MDSH) Protocol.

Remark 3. The probability of an adversary guessing the correct index i is $1/n$. Thus, the protocol must be repeated sufficient number of times (*e.g.* m times) in order to prevent an attacker[7] to convince $Victor$ that he knows k_i, for $i \in [0, n]$.

[7] In this case, the attacker's success probability is $1/n^m$.

Remark 4. In order for the MDSH protocol to be efficient, we must assume that the decision of membership $y \in R_{k_j}$ can be done in polynomial time with respect to the bit-length of the statement $|\{f_{k_i}\}_{i\in[0,n]}|$.

Remark 5. A protocol for statistical distance was introduced in [28]. Let D_0 and D_1 be two statistical distributions. The verifier begins by flipping a coin b to obtain a random bit b and then sends an element $z \xleftarrow{\$} D_b$ to *Peggy*. She has to determine the correct distribution for z and send her guess b' to $Victor$. The verifier accepts the proof if and only if $b = b'$. Sahai and Vadhan proved that this protocol is statistical zero knowledge in the honest verifier scenario. Note that if $n = 2$ our proposed protocol becomes a special case of Sahai and Vadhan's protocol.

3.2 Security Analysis

To ease understanding, we first introduce the notion of a multi-decisional hard function and then we prove the security of the MDSH protocol. At the end of this subsection we show how to relate the security of a multi-decisional function to the security of a decisional function.

Definition 8 (Multi-Decisional Hardness Assumption). *Let $n \geq 2$ be an integer. A function f is a multi-decisional hard function if in Definition 2, Item 2 and 3 are changed to*

2. *for every PPT algorithm A the advantage*

$$ADV_f^{\text{MDHA}}(A) = |n \cdot Pr[i = i' \mid for\ i \in [0,n] : k_i \xleftarrow{\$} K; i \xleftarrow{\$} [0,n]; x \xleftarrow{\$} D_{k_i}; \\ y \leftarrow f_{k_i}(x); i' \leftarrow A(f_k, y)] - 1|$$

   is negligible, where $f_k = \{f_{k_i}\}_{i\in[0,n]}$;
3. *there exists a PPT algorithm B such that*

$$Pr[i = i' \mid for\ i \in [0,n] : k_i \xleftarrow{\$} K; i \xleftarrow{\$} [0,n]; x \xleftarrow{\$} D_{k_i}; \\ y \leftarrow f_{k_i}(x); i' \leftarrow B(k, y)] = 1,$$

   where $k = \{k_i\}_{i\in[0,n]}$.

Remark 6. Note that in the case of the multi-decisional hardness assumption we implicitly assume that all the keys are kept secret and none of them are leaked to an adversary (dishonest prover). If, for example, t out of n keys are leaked there is a simple strategy that makes the attacker win with probability $(t+1)/n$. More precisely, his strategy works as follows: The attacker, upon receipt of the verifier's challenge y, checks whether the message belongs to the set R_{k_i} for any of the t known secrets. If true (that happens with probability t/n), the attacker correctly answers the corresponding index of the matching secret. Otherwise, the attacker answers a random index chosen among the unknown secrets. In this last case, the success probability is $1/(n-t) \cdot (n-t)/n = 1/n$. Hence, the total success probability is $t/n + 1/n = (t+1)/n$.

Theorem 1. *The MDSH protocol is a proof of knowledge if and only if f is a multi-decisional hard function. Moreover, the protocol is zero knowledge in the honest verifier scenario.*

Proof. If f is a multi-decisional hard function, then according to Definition 8, Item 3, *Peggy* will compute with probability 1 the correct index. Thus, the completeness property is satisfied.

Let $\tilde{P}$ be a PPT algorithm that takes as input $f_{k_0}, \ldots, f_{k_{n-1}}$ and makes V accept the proof with non-negligible probability $Pr(\tilde{P})$. Then we are able to construct a PPT algorithm Q (described in Algorithm 1) that interacts with $\tilde{P}$ and that has a non-negligible advantage $ADV_f^{\text{MDHA}}(Q) = Pr(\tilde{P})$. Thus, the soundness property is satisfied.

Algorithm 1. Algorithm Q.

Input: An element $y \leftarrow f_{k_i}(x)$ and n functions f_{k_i}, where $i \in [0, n]$
1 Send y to $\tilde{P}$
2 Receive i' from $\tilde{P}$
3 **return** i'

The last part of our proof consists in constructing a simulator S such that its output is indistinguishable from a genuine transcript between *Peggy* and $Victor$. Such a simulator is described in Algorithm 2.

Algorithm 2. Simulator S.

Input: n functions f_{k_i}, where $i \in [0, n]$
1 Choose $i \xleftarrow{\$} [0, n]$
2 Choose $x \xleftarrow{\$} D_{k_i}$
3 Compute $y \leftarrow f_{k_i}(x)$
4 **return** (y, i)

□

We further show that if ADV_f^{DHA} is negligible, then MDSH is secure. Thus, when instantiating MDSH it suffices to know that decisional functions are secure.

Theorem 2. *For any PPT algorithm A there exists a PPT algorithm B such that the following inequality holds*

$$ADV_f^{\text{MDHA}}(A) \leq ADV_f^{\text{DHA}}(B).$$

Proof. Let A have a non-negligible advantage $ADV_f^{\text{MDHA}}(A)$. We describe in Algorithm 3 how B can obtain a non-negligible advantage $ADV_f^{\text{DHA}}(B)$ by interacting

Algorithm 3. Algorithm B.

Input: An element $y \leftarrow f_{k_b}(x)$, where $b \xleftarrow{\$} \{0,1\}$
for $i \in [2,n]$ **do**
  Choose $k_i \xleftarrow{\$} K$
Randomly shuffle $f_{k_0}, \ldots, f_{k_{n-1}}$'s positions and denote the result by $f'_{k_0}, \ldots, f'_{k_{n-1}}$
Let $i' \leftarrow A(f'_{k_0}, \ldots, f'_{k_{n-1}}, y)$
if i' *is the position of* f_{k_0} **then return** 0
else if i' *is the position of* f_{k_1} **then return** 1
else return $\perp$

with A. Note that we have to randomly shuffle the functions' positions, in order to ensure that the index is randomly chosen from $[0,n]$.

□

Proposition 1. *Let $D_{k_i} \subseteq D$ and $R_{k_i} \subseteq R$. If $(R, \odot)$ is a group and there exists an $\bar{x} \in D$ and an $j \in [0,n]$ such that $B(k,y) = B(k,\bar{y})$, where $\bar{y} \leftarrow y \odot f_{k_j}(\bar{x})$ then the MDSH is not secure against active-intruder attacks.*

Proof. When Victor sends his first message y, *Mallory* intercepts it, computes $f_{k_j}(\bar{x})$ and forwards $\bar{y} = y \odot f_{k_j}(\bar{x})$ to *Peggy* (see Figure 2). The second message is simply forwarded by Mallory. We can see that Mallory's attack succeeds since

$$i' = B(k,\bar{y}) = B(k,y),$$

just as required by Victor's verification.

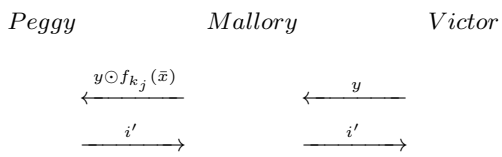


Fig. 2. Active-Intruder Attack Against MDSH.

□

Proposition 2. *Let $D_{k_i} \subseteq D$ and $R_{k_i} \subseteq R$. If $(R, \odot)$ is a group and for any $\bar{x} \in D$ and $j \in [0,n]$ we have $B(k,\tilde{y}) \equiv B(k,\bar{y}) + B(k,y) \bmod n$, where $\bar{y} = f_{k_j}(\bar{x})$ and $\tilde{y} = y \odot \bar{y}$, then the MDSH is not secure against active-intruder attacks.*

Proof. When Victor sends y, *Mallory* intercepts it, selects any $(\bar{x}, j)$ and forwards $\tilde{y} = y \odot f_{k_j}(\bar{x})$ to *Peggy* (see Figure 3). The second message is intercepted

by Mallory, who computes $\tilde{i} \equiv i' - j \bmod n$ and forwards $\tilde{i}$ to Victor. We can see that Mallory's attack succeeds since

$$\tilde{i} \equiv i' - j \equiv B(k, \tilde{y}) - B(k, \bar{y}) \equiv B(k, y) \bmod n,$$

just as required by Victor's verification.

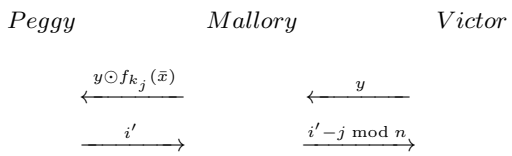


Fig. 3. Active-Intruder Attack Against MDSH.

□

Remark 7. Note that if the conditions of Proposition 2 hold, then Proposition 1 is automatically obtained by selecting $j = 0$.

3.3 Examples

Quadratic Residuosity Assumption. Let N be the product of two large primes p and q and let $J_N(x)$ denote the Jacobi symbol of x modulo N. We denote by $J_N = \{x \in \mathbb{Z}_N^* \mid J_N(x) = 1\}$ and $QR_N = \{x \in \mathbb{Z}_N^* \mid J_p(x) = 1 \text{ and } J_q(x) = 1\}$. Let u be an element such that his Jacobi symbol $J_N(u)$ is 1. The *quadratic residuosity assumptions* (denoted by QR) states that deciding if $u \in J_N \setminus QR_N$ or $u \in QR_N$ is intractable without knowing p or q (see [9]).

Since QRA partitions J_N in two sets, we must set $n = 2$ for MDSH. Let u be an element such that $J_p(u) = J_q(u) = -1$. Then the MDSH parameters are as follows

- the secret keys are $k_0 = k_1 = (p, q)$;
- the functions are defined as $f_{k_0}(x) = x^2 \bmod N$ and $f_{k_1}(x) = u \cdot x^2 \bmod N$, where u and N are public.

To decide if $y \in J_N \setminus QR_N$ or $y \in QR_N$, *Peggy* computes $J_p(y)$. Note that when $b = 0$ we have $J_p(y) = J_p(x^2) = 1$ and when $b = 1$ we have $J_p(y) = J_p(u)J_p(x^2) = -1$.

The active-intruder attack from Proposition 2 works as follows: *Mallory* chooses $j \xleftarrow{\$} \{0, 1\}$, $\bar{x} \xleftarrow{\$} \mathbb{Z}_N^*$ and forwards $y \cdot u^j \bar{x}^2 \bmod N$ to Peggy, and in the second phase forwards $i' + j \bmod 2$ to Victor. Let $y \equiv u^b x^2 \bmod N$. The attack works since

$$\bar{y} \equiv u^b x^2 \cdot u^j \bar{x}^2 \equiv u^{b+j \bmod 2} \cdot (u^{b+j \operatorname{div} 2} x \bar{x})^2 \bmod N$$

and the term $u^{b+j \bmod 2}$ decides if $\bar{y}$ is a quadratic residue or not.

Remark 8. A similar assumption can be found in [6]. Let $\kappa > 1$ be an integer and let $p, q \equiv 1 \bmod 2^\kappa$. Then the *gap* 2^κ*-residuosity assumption* states that is hard to distinguish between an element from $J_N \setminus QR_N$ and element of the form $y^{2^\kappa} \bmod N$, where $y \in \mathbb{Z}_N^*$. In this case the functions become $f_{k_0}(x) = x^{2^\kappa} \bmod N$ and $f_{k_1}(x) = u \cdot x^{2^\kappa} \bmod N$. Note a similar QR active-intruder attack exist for this assumption.

Least Significant Bit of the e-th Root Assumption. Let $N = pq$ be the product of two large primes. We denote by $\varphi(N)$ the Euler totient function. Let e be an integer such that $gcd(e, \varphi(N)) = 1$. The *least significant bit of the e-th root assumption* (denoted LSB-ER) states that given $y \equiv x^e \bmod N$ is hard to decide if the least-significant bit of x is 0 or 1 (see [26]).

As in the case of QR, we have $n = 2$. The protocol's parameters are

- the secret keys are $k_0 = k_1 = (p, q)$;
- the functions are defined as $f_{k_0}(x) = (2x)^e \bmod N$ and $f_{k_1}(x) = (2x + 1)^e \bmod N$, where N and e are public.

To find the least significant bit lsb, *Peggy* computes a d such that $ed \equiv 1 \bmod \varphi(N)$ and an element $z \leftarrow y^d \bmod N$. Then $lsb \equiv z \bmod 2$.

The active-intruder attack from Proposition 1 works as follows: *Mallory* chooses $j = 1$, $\bar{x} \xleftarrow{\$} \mathbb{Z}_N^*$ and forwards $y \cdot (2\bar{x} + j)^e \bmod N$ to Peggy, and in the second phase forwards i' to Victor. Let $y \equiv (2x + b)^e \bmod N$. The attack works since

$$\bar{y} \equiv (2x + b)^e \cdot (2\bar{x} + j)^e \equiv [2(2x\bar{x} + xj + \bar{x}b) + jb]^e \equiv (2x' + b)^e \bmod N,$$

and thus $(\bar{y}^d \bmod N) \equiv (y^d \bmod N) \bmod 2$.

Decisional Diffie-Hellman Assumption. Let $\mathbb{G}$ be a cyclic group of prime order q and g a generator of $\mathbb{G}$. Let $x_1, x_2, y \xleftarrow{\$} \mathbb{Z}_q^*$ and $b \xleftarrow{\$} \{0, 1\}$. The *decisional Diffie-Hellman assumption* (denoted by DDH) states that given $(g^{x_1}, g^{x_2}, g^y, (g^{x_b})^y)$ the probability for a PPT algorithm to compute the bit b is negligible (see [5]).

In this case $n \geq 2$ and the parameters are

- the secret keys are $k_i \xleftarrow{\$} \mathbb{Z}_q^*$, for $i \in [0, n]$;
- the public parameters are $r_i \leftarrow g^{k_i}$, for $i \in [0, n]$, the group $\mathbb{G}$ and the generator g;
- the functions are defined as $f_{k_i}(x) = (g^x, r_i^x)$, for $i \in [0, n]$.

To decide the correct index, *Peggy* has to parse $y = (y_0, y_1)$ and to compute $\ell = y_0^{k_i}$ until $\ell = y_1$. Note that $y_0^{k_i} = r_i^x$.

Let $(y_0, y_1) = (g^x, r_i^x)$. The active-intruder attack from Proposition 1 works as follows: *Mallory* forwards $(\bar{y}_0, \bar{y}_1) = (y_0^2, y_1^2)$ to Peggy, and in the second phase forwards i' to Victor. The attack works since

$$\bar{y}_1 = y_1^2 = r_i^{2x} = g^{2xk_i} = (y_0^2)^{k_i} = \bar{y}_0^{k_i},$$

and thus we obtain the same index i.

Remark 9. When $n = 2$ we obtain the protocol introduced in [1]. This protocol was introduced to show the existence of a two-round private-coin zero knowledge proof system for a promise problem lying outside of **BPP**.

Decisional Bilinear Diffie-Hellman Assumption. Let $\mathbb{G}$ be cyclic group of prime order q and let P be the corresponding generator. We denote by $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ a cryptographic bilinear map, where G_T is a cyclic group of order q. We will use the convention of writing $\mathbb{G}$ additively and $\mathbb{G}_T$ multiplicatively.

Let $a_0, a_1, b_0, b_1, c \xleftarrow{\$} \mathbb{Z}_q^*$. The *decisional bilinear Diffie-Hellman assumption* (denoted DBDH) states that given $(a_0P, a_1P, b_0P, b_1P, cP, Z)$ the probability of deciding if $Z = e(P, P)^{a_0b_0c}$ or $Z = e(P, P)^{a_1b_1c}$ is negligible (see [7]).

As in the case of DDH, we have $n \geq 2$. The MDSH's parameters are

- the secret keys are $a_i, b_i \xleftarrow{\$} \mathbb{Z}_q^*$, for $i \in [0, n]$;
- the public parameters are $Q_i \leftarrow a_iP$ and $R_i \leftarrow b_iP$, for $i \in [0, n]$, the group $\mathbb{G}$, the generator P and the bilinear map e;
- the functions are defined as $f_{k_i}(x) = (xP, e(Q_i, R_i)^x)$, for $i \in [0, n]$.

To find the correct answer, *Peggy* parses $y = (Y_0, Y_1)$ and computes $L = e(P, Y_0)^{a_ib_i}$ until $L = Y_1$. Note that $e(Q_i, R_i)^x = e(P, P)^{a_ib_ix} = e(P, xP)^{a_ib_i} = e(P, Y_0)^{a_ib_i}$.

Let $(Y_0, Y_1) = (xP, e(Q_i, R_i)^x))$. The active-intruder attack from Proposition 1 works as follows: *Mallory* forwards $(\bar{Y}_0, \bar{Y}_1) = (2Y_0, Y_1^2)$ to Peggy, and in the second phase forwards i' to Victor. The attack works since

$$\begin{aligned}\bar{Y}_1 = Y_1^2 &= e(Q_i, R_i)^{2x} = e(a_iP, b_iP)^{2x} \\ &= e(P, 2xP)^{a_ib_i} = e(P, 2Y_0)^{a_ib_i} = e(P, \bar{Y}_0)^{a_ib_i},\end{aligned}$$

and thus we obtain the same index i.

4 Basic Vectorized Multi-Decisional Protocol

4.1 Description

A downside to the MDSH protocol is that *Victor* has to run the protocol a number of times before he can be sure that *Peggy* knows $\{k_i\}_{i\in[0,n]}$. We further present a variation of MDSH (see Figure 4) that allows *Victor* to run the protocol only once, if he chooses the right parameters. Let $t > 1$ be an integer.

Remark 10. The probability of an adversary guessing the correct index vector v is $1/n^t$. If n^t is sufficiently large, then a single execution of the protocol suffices. Otherwise, *Victor* must rerun the protocol multiple times.

Remark 11. As in the case of MDSH protocol, we must also assume that the decision of membership $y \in R_{k_j}$ from Peggy's side of the VDSH0 protocol can be done in polynomial time.

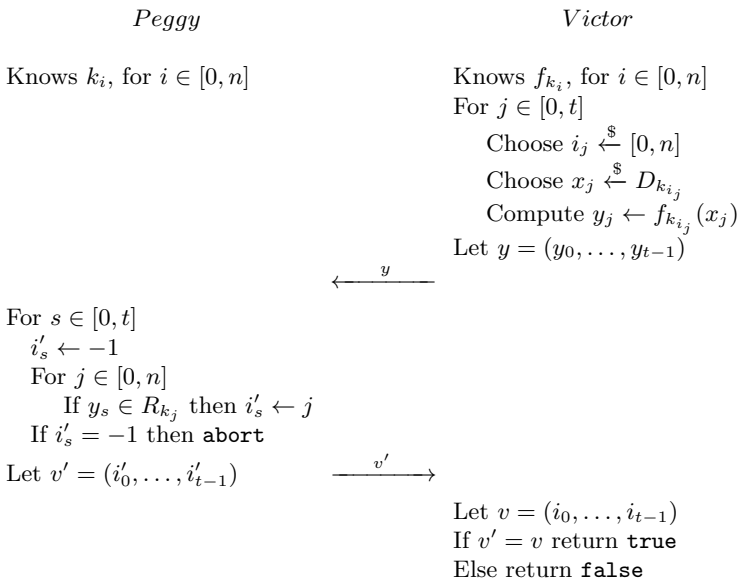


Fig. 4. Vectorized Multi-Decisional Sherlock Holmes (VDSH0) Protocol.

4.2 Security Analysis

As in Section 3.2, we first introduce the relevant hardness assumption, then we prove the security of the VDSH0 protocol and at the end we relate the new hardness assumption with the multi-dimensional hardness assumption.

Definition 9 (Vectorized Multi-Decisional Hardness Assumption). *Let $t > 1$ be an integer. A function f is a vectorized multi-decisional hard function if in Definition 8, Item 2 and 3 are changed to*

2. *for every PPT algorithm A the advantage*

$$ADV_f^{\text{VDHA}}(A) = |n^t \cdot Pr[v = v' \mid \text{for } i \in [0,n] : k_i \xleftarrow{\$} K; \text{for } j \in [0,t] : i_j \xleftarrow{\$} [0,n], x_j \xleftarrow{\$} D_{k_{i_j}}, y_j \leftarrow f_{k_{i_j}}(x_j); v' \leftarrow A(f_k, y)] - 1|$$

   is negligible, where $f_k = \{f_{k_i}\}_{i \in [0,n]}$, $v = \{i_j\}_{j \in [0,t]}$ and $y = \{y_j\}_{j \in [0,t]}$;
3. *there exists a PPT algorithm B such that*

$$Pr[v = v' \mid \text{for } i \in [0,n] : k_i \xleftarrow{\$} K; \text{for } j \in [0,t] : i_j \xleftarrow{\$} [0,n], x_j \xleftarrow{\$} D_{k_{i_j}}, y_j \leftarrow f_{k_{i_j}}(x_j); v' \leftarrow B(k, y)] = 1,$$

   where $k = \{k_i\}_{i \in [0,n]}$, $v = \{i_j\}_{j \in [0,t]}$ and $y = \{y_j\}_{j \in [0,t]}$.

Theorem 3. *The VDSH0 protocol is a proof of knowledge if and only if f is a vectorized multi-decisional hard function. Moreover, the protocol is zero knowledge in the honest verifier scenario.*

Proof. The proof is similar to Theorem 2 and thus we only provide a sketch. The completeness property is satisfied due to Definition 9, Item 3.

A PPT algorithm R is described in Algorithm 4 and R has a non-negligible advantage $ADV_f^{\text{VDHA}}(R) = Pr(\tilde{P})$.

Algorithm 4. Algorithm R.

Input: A vector $y \leftarrow (f_{k_0}(x_0), \ldots, f_{k_{t-1}}(x_{t-1}))$
Send y to $\tilde{P}$
Receive v' from $\tilde{P}$
return v'

Finally, the simulator T is described in Algorithm 12

Algorithm 5. Simulator T.

Input: n functions f_{k_i}, where $i \in [0, n]$
for $j \in [0, t]$ **do**
    Choose $i_j \xleftarrow{\$} [0, n]$
    Choose $x_j \xleftarrow{\$} D_{k_{i_j}}$
    Compute $y_j \leftarrow f_{k_{i_j}}(x)$
Let $y = (y_0, \ldots, y_{t-1})$ and $v = (i_0, \ldots, i_{t-1})$
return (y, v)

□

The next theorem proves the equivalence between the security notion associated with multi-decisional functions and the vectorized version of it. Using Theorems 2 and 4, the security of VDSH0 reduces to making sure that the decisional security notion is intractable.

Theorem 4. *For any PPT algorithms A and C there exist PPT algorithms B and D such that the following inequalities hold*

$$ADV_f^{\text{MDHA}}(A) \leq ADV_f^{\text{VDHA}}(B)$$
$$ADV_f^{\text{VDHA}}(C) \leq ADV_f^{\text{MDHA}}(D).$$

Proof. Let A have a non-negligible advantage $ADV_f^{\text{MDHA}}(A)$ and let $Pr(A) = (ADV_f^{\text{MDHA}}(A) + 1)/n$. We describe in Algorithm 6 how B can obtain a non-negligible advantage $ADV_f^{\text{VDHA}}(B) = |n^t \cdot Pr(A)^t - 1|$ by interacting with A.

Algorithm 6. Algorithm B.

Input: A vector of elements $y \leftarrow (y_0, \ldots, y_{t-1})$
for $j \in [0, t]$ **do**
  Let $i'_j \leftarrow A(f_{k_0}, \ldots, f_{k_{n-1}}, y_j)$
Let $v' = (i'_0, \ldots, i'_{t-1})$
return v'

To prove the second inequality we assume that $ADV_f^{\text{VDHA}}(C)$ is non-negligible. Using algorithm C, we construct algorithm D (see Algorithm 7) that has a non-negligible advantage $ADV_f^{\text{MDHA}}(D)$.

Algorithm 7. Algorithm D.

Input: An element $y \leftarrow f_{k_i}(x)$, where $i \xleftarrow{\$} [0, n]$
for $j \in [1, t]$ **do**
  Choose $i_j \xleftarrow{\$} [0, n]$
  Choose $x_j \xleftarrow{\$} D_{k_{i_j}}$
  Compute $y_j \leftarrow f_{k_{i_j}}(x)$
Let $z = (y, y_1, \ldots, y_{t-1})$ and $f_k = (f_{k_0}, \ldots, f_{k_{n-1}})$
Let $v' \leftarrow C(f_k, z)$
Parse $v' = (v'_0, \ldots, v'_{t-1})$
return v'_0

□

Since VDSH0 is the vectorized version of MDSH, the active-intruder attacks from Propositions 1 and 2 can be easily adapted to VDSH0 by simply applying them for each component of the y vector.

Corollary 1. *Let $D_{k_i} \subseteq D$ and $R_{k_i} \subseteq R$. If $(R, \odot)$ is a group and there exists an $\bar{x} \in D$ and an $j \in [0, n]$ such that $B(k, y) = B(k, \bar{y})$, where $\bar{y} \leftarrow y \odot f_{k_j}(\bar{x})$ then the VDSH0 is not secure against active-intruder attacks.*

Corollary 2. *Let $D_{k_i} \subseteq D$ and $R_{k_i} \subseteq R$. If $(R, \odot)$ is a group and for any $\bar{x} \in D$ and $j \in [0, n]$ we have $B(k, \tilde{y}) \equiv B(k, \bar{y}) + B(k, y) \bmod n$, where $\bar{y} = f_{k_j}(\bar{x})$ and $\tilde{y} = y \odot \bar{y}$, then the VDSH0 is not secure against active-intruder attacks.*

5 Vectorized Multi-Decisional Protocol Variant

5.1 Description

We further present a variation of VDSH0 (see Figure 5) that is secure against concurrent and active-intruder attacks. In order to work, the protocol uses a

public string str, a hash function $h : \{0,1\}^* \to \{0,1\}^\delta$ and a pseudo-random permutation $\pi : \{0,1\}^\delta \times \{0,1\}^\tau \to \{0,1\}^\tau$. Note that we assume that n^t is large enough to avoid brute force attacks.

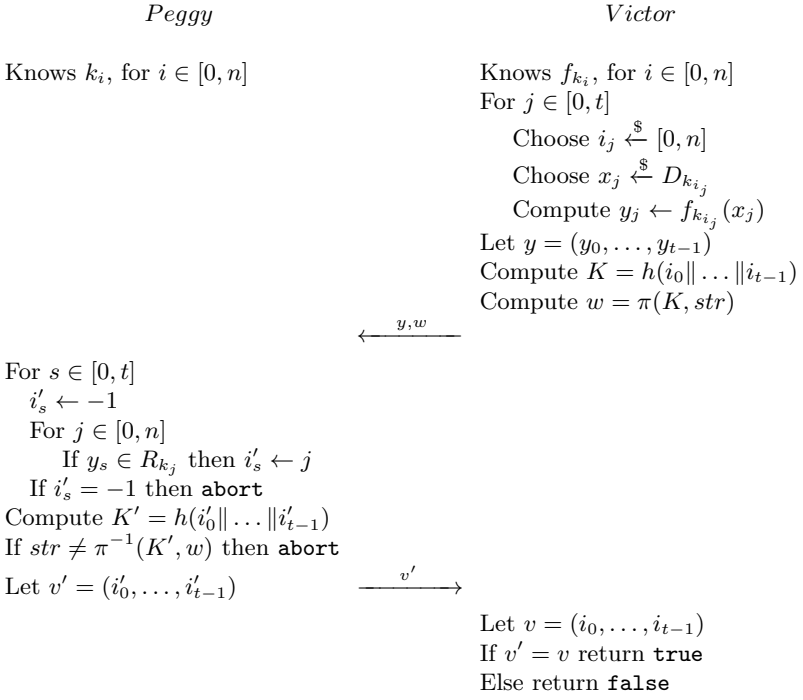


Fig. 5. Vectorized Multi-Decisional Sherlock Holmes (VDSH1) Protocol.

5.2 Security Analysis

We further prove that the variation of VDSH0 can protect the end users from more powerful attackers than the basic version.

Theorem 5. *The VDSH1 protocol is secure against sequential impersonation attacks in the random oracle model.*

Proof. Let A be a impersonator that has a non-negligible success probability $ADV_{P,V}^{\text{IMP-CA}}(A)$. In the first phase, the attacker A can make hash oracle queries and can interact with the prover P. Therefore, we must simulate the hash oracle (see Algorithm 8) and the prover (see Algorithm 9) such that the outputs are statistically indistinguishable from genuine outputs. Note that in Algorithm 8 the list T starts empty. We can see that simulator $\mathcal{O}_P$ is identical with P except

when it aborts on correct challenges[8] or responds with a $v_i'' \neq v_i'$ that correctly decrypts str. These events happens if and only if π's key K_i is not a reply to a hash oracle query or there exists an $q_j \neq K_i$ such that $h_j = h(K_i)$. Hence, they happen with probability $1/2^\delta$ and less than $q_h/2^\delta$, where q_h is the number of queries to $\mathcal{O}_h$ made by A. Thus, both events happen with negligible probability. As a result, the probability of ending phase one with success is greater than $(1-(1+q_h)/2^\delta)^{q_p} \geq 1-(1+q_h)q_p/2^\delta$, where q_p is the number of queries to $\mathcal{O}_P$ made by A.

Algorithm 8. Hashing oracle $\mathcal{O}_h$ simulation for h.

Input: A hashing query q_i from A
if $\exists h_i$ *such that* $\{q_i, h_i\} \in T$ **then**
  $e \leftarrow h_i$
else
  $e \xleftarrow{\$} \{0,1\}^\delta$
  Append $\{q_i, e\}$ to T
return e

Algorithm 9. Prover P simulator $\mathcal{O}_P$.

Input: A challenge query (y_i, w_i) from A
if $\exists \{q_j, h_j\} \in T$ *such that* $\pi^{-1}(h_j, w_i) = str$ **then**
  return q_j
else
  return $\perp$

In the second phase of the attack, A interacts with the prover an tries to impersonate P. A PPT algorithm $\mathcal{O}_V$ is described in Algorithm 10. Since π is a pseudo-random permutation and y is VDHA challenge, then A will always accept (y, w). We can see that the probability of A not aborting is $1/2^\tau$. In this case $\mathcal{O}_V$ has a non-negligible advantage $ADV_f^{\text{VDHA}}(O_V) = ADV_{P,V}^{\text{IMP-CA}}(A)$. If A aborts, then the probability that the correct answer is found in T is $1-1/2^\delta$, which is non-negligible. In this case $\mathcal{O}_V$ has a non-negligible advantage $ADV_f^{\text{VDHA}}(O_V) = ADV_{P,V}^{\text{IMP-CA}}(A)/q_h$. Therefore, the total advantage of $\mathcal{O}_V$ is

$$\begin{aligned} ADV_f^{\text{VDHA}}(O_V) &\geq (1-(1+q_h)q_p/2^\delta) \cdot (1/2^\tau + (1-1/2^\delta) \cdot 1/q_h) \cdot ADV_{P,V}^{\text{IMP-CA}}(A) \\ &\simeq (1-q_h q_p/2^\delta)/q_h \cdot ADV_{P,V}^{\text{IMP-CA}}(A), \end{aligned}$$

which is non-negligible.

□

[8] if (y_i, w_i) is malformed then P would also abort

Algorithm 10. Verifier V simulator $\mathcal{O}_V$.

Input: A vector $y \leftarrow (f_{k_0}(x_0), \ldots, f_{k_{t-1}}(x_{t-1}))$

Choose $w \xleftarrow{\$} \{0,1\}^\tau$
Send (y, w) to A
if *A sends the* `abort` *signal* **then**
    Select $i \xleftarrow{\$} [0, q_h]$
    Retrieve $\{q_i, h_i\}$ from T
    return q_i
else
    Receive v' from A
    return v'

Theorem 6. *The VDSH1 protocol is secure against active-intruder attacks in the random oracle model.*

Proof. We further prove that if attacker A becomes active in a session, then the verifier will reject. We use three bit strings to indicate which of the three items (y, w, v') are not fatefully relayed. More precisely, **1** means that the corresponding item is altered, while **0** means that is not altered. Let $\bar{y}$, $\bar{w}$ and $\bar{v}'$ denote the altered items. We distinguish the following possible cases

case 001: Since $\bar{v}' \neq v' = v$ the verifier will automatically reject.

case 010: Changing w will result in rejection from the prover because *str* cannot be recovered. This implies that the prover will also reject.

case 011: The prover will reject as in the previous case, and thus A will not get any useful information from interacting with P. If A manages to make the verifier accept $\bar{v}'$, then he can do the same thing without interacting with P. This contradicts Theorem 5.

case 100: Since h is random oracle, the probability of obtaining a collision such that $h(i_0 \| \ldots \| i_{t-1}) = h(\bar{i}_0 \| \ldots \| \bar{i}_{t-1})$ is $1/2^\delta$, where the indexes are corresponding to y and $\bar{y}$. This implies that the prover will reject with non-negligible probability.

case 101: As in the previous case, the prover will most certainly reject. Therefore, if A manages to convince V that $\bar{v}'$ is correct, then he can do that without interacting with P. Again, this contradicts Theorem 5.

case 110: In this case the prover will reject with overwhelming probability since the probability of obtaining *str* is $1/2^\tau$. Therefore, the prover will also reject.

case 111: According to the previous case, the prover will reject with overwhelming probability. Therefore, A caries out a concurrent attack and according to Theorem 5 the verifier will reject with non-negligible probability.

To summarize, if adversary A becomes active in a session, then the verifier will most certainly reject the proof. □

6 Basic Computational Protocol

6.1 Description

Using a different security notion, we describe in Figure 6 a protocol that consumes less bandwith that the VDSH protocol, while maintaining its security, if the parameters are selected correctly.

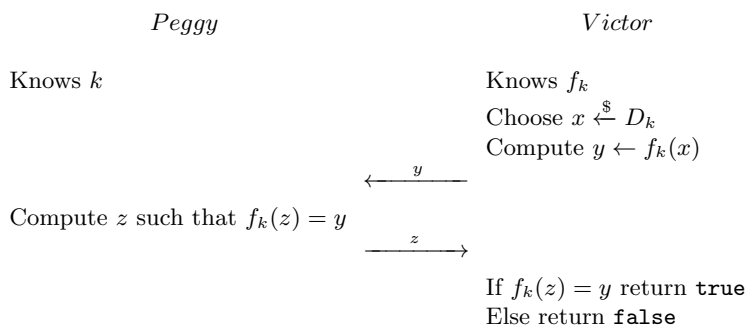


Fig. 6. Basic Computational Sherlock Holmes (CSH0) Protocol.

Remark 12. The probability of an adversary guessing the correct element x is $1/|D_k|$. If $|D_k|$ is sufficiently large, then a single execution of the protocol suffices. Otherwise, the protocol must be repeated several times.

Remark 13. A vectorized version of the CSH0 protocol can also be constructed, but as we will see in Section 6.3 it is not necessary. Note that the security analysis is similar to the one from Section 4.2.

6.2 Security Analysis

Theorem 7. *The CSH0 protocol is a proof of knowledge if and only if f is a computational hard function. Moreover, the protocol is zero knowledge in the honest verifier scenario.*

Proof. The proof is similar to Theorem 2, and thus we only provide a sketch. The completeness property is satisfied due to Definition 1, Item 3.

A PPT algorithm O is described in Algorithm 11 and O has a non-negligible advantage $ADV_f^{\text{CHA}}(O) = Pr(\tilde{P})$. Note that in this case $\tilde{P}$ only takes as input a function f_k.

Finally, the simulator U is described in Algorithm 12

□

Algorithm 11. Algorithm O.

Input: An element $y \leftarrow f_k(x)$

1 Send y to $\tilde{P}$
2 Receive z from $\tilde{P}$
3 **return** z

Algorithm 12. Simulator U.

Input: A function f_k

1 Choose $x \xleftarrow{\$} D_k$
2 Compute $y \leftarrow f_k(x)$
3 **return** (y, x)

Proposition 3. *Let $D_{k_i} \subseteq D$ and $R_{k_i} \subseteq R$. If $(D, \bullet)$ and $(R, \odot)$ are groups, and for any $x_1, x_2 \in D$ we have $f_k(x_1 \bullet x_2) = f_k(x_1) \odot f_k(x_2)$, then the CSH0 is not secure against active-intruder attacks.*

Proof. When Victor sends y, *Mallory* intercepts it, chooses $\bar{x} \xleftarrow{\$} D$ and forwards $\tilde{y} = y \odot f_k(\bar{x})$ to *Peggy* (see Figure 7). The second message is intercepted by Mallory, who computes $\tilde{z} = z \bullet \bar{x}^{-1}$ and forwards $\tilde{z}$ to Victor. We can see that Mallory's attack succeeds since

$$\tilde{y} = f_k(x) \odot f_k(\bar{x}) = f_k(x \bullet \bar{x}),$$

and thus *Peggy* computes $z = x \bullet \bar{x}$. Therefore, $x = z \bullet \bar{x}^{-1}$ just as required by Victor's verification.

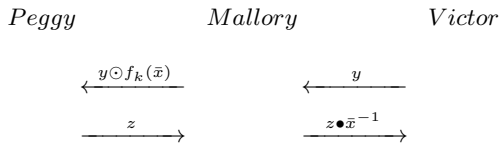


Fig. 7. Active-Intruder Attack Against CSH0.

□

6.3 Examples

e-th Root Assumption. Using the same parameters as in the case of LSB-ER, the *e-th root assumption* (denoted ER) states that given $y \equiv x^e \bmod N$ computing x is intractable (see [23]).

Using this assumption we can instantiate the CSH0 protocol with $k = (p, q)$ and $f_k(x) = x^e \bmod N$. To recover x, *Peggy* has to compute a d such that $ed \equiv 1 \bmod \varphi(N)$ and then $x \leftarrow y^d \bmod N$.

The active-intruder attack from Proposition 3 works as follows: *Mallory* chooses $\bar{x} \xleftarrow{\$} \mathbb{Z}_N^*$ and forwards $y\bar{x}^e \bmod N$ to Peggy, and in the second phase forwards $z\bar{x}^{-1} \bmod N$ to Victor. The attack works since

$$\bar{y} \equiv y\bar{x}^e \equiv (x\bar{x})^e \bmod N,$$

and thus *Peggy* computes $z \equiv x\bar{x} \bmod N$.

Remark 14. The problem can also be stated for $e = 2$, but to find a solution to $x^2 \bmod N$, *Peggy* has to use a different technique (*e.g* the Shanks-Tonelli algorithm [24]). Note that this assumption, called the *square root assumption*, is equivalent with the intractability of factoring N (*i.e. factoring assumption*).

Gap 2^κ-residuosity Assumption. Using the same parameters as in Section 3.3, we can define $f_k(x) = u^x z^{2^\kappa} \bmod N$, where $k = (p, q)$, $D_k = [0, 2^\kappa]$ and $z \xleftarrow{\$} \mathbb{Z}_N^*$. A method for recovering x if one knows p is described in [6].

In this case Proposition 3's attack becomes: *Mallory* chooses $\bar{z} \xleftarrow{\$} \mathbb{Z}_N^*$ and $\bar{x} \xleftarrow{\$} [0, 2^\kappa]$, and forwards $y \cdot u^{\bar{x}} \bar{z}^{2^\kappa} \bmod N$ to Peggy, and in the second phase forwards $z - \bar{x} \bmod 2^\kappa$. The attack works since

$$\bar{y} \equiv y \cdot u^{\bar{x}} \bar{z}^{2^\kappa} \equiv u^{x+\bar{x} \bmod 2^\kappa} \cdot (u^{x+\bar{x} \operatorname{div} 2^\kappa} \cdot z\bar{z})^{2^\kappa},$$

and thus *Peggy* computes $z \equiv x + \bar{x} \bmod 2^\kappa$.

Computational Diffie-Hellman. Let $\mathbb{G}$ be a cyclic group of order q and g a generator of $\mathbb{G}$. Let $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_q^*$. The *computational Diffie-Hellman assumption* (denoted by CDH) states that given (g^{x_1}, g^{x_2}) is intractable to compute $g^{x_1 x_2}$ without knowing x_1 or x_2 (see [5]). In this case a more efficient version of the CSH0 protocol is provided in Figure 8.

Remark 15. Note that the DHCSH0 protocol was used in [34] to develop a method that performs full network authentication for resource-constrained devices.

The active-intruder attack from Proposition 3 works as follows: *Mallory* chooses $\bar{x} \xleftarrow{\$} \mathbb{Z}_q^*$ and forwards $\bar{y} = yg^{\bar{x}}$ to Peggy, and in the second phase forwards $\bar{z} = zr^{-\bar{x}}$ to Victor. The attack works since

$$\bar{z} = \bar{y}^k r^{-\bar{x}} = (yg^{\bar{x}})^k \cdot (g^k)^{-\bar{x}} = y^k = r^x,$$

just as desired.

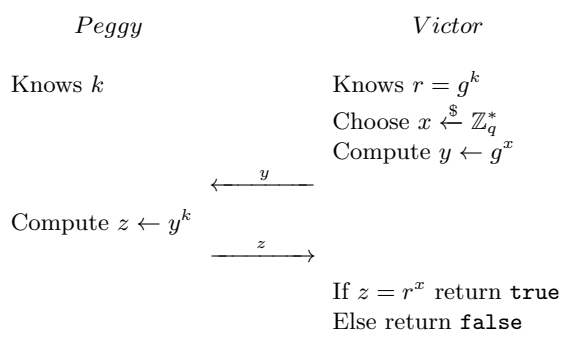


Fig. 8. Basic Diffie-Hellman Version of the CSH0 (DHCSH0) Protocol.

Computational Bilinear Diffie-Hellman Assumption. We assume the same setup as in the case of DBDH. Let $a, b, c \xleftarrow{\$} \mathbb{Z}_q^*$. The *computational bilinear Diffie-Hellman assumption* (denoted CBDH) states that given (aP, bP, cP) a PPT algorithm will compute $e(P, P)^{abc}$ with negligible probability (see [7]).

As in the case of CDH, this assumption allows us to have a more efficient version of the protocol. We will use Figure 8 as a reference. Thus, $Peggy$ and $Victor$ know $k = (a, b)$ and, respectively, $r = (aP, bP)$. The protocol's first step consists of $Victor$ computing $y \leftarrow xP$. After that $Peggy$ computes $z \leftarrow e(P, y)^{ab}$. Finally, the protocol's output is `true` if and only if $z = e(aP, bP)^x$.

The active-intruder attack from Proposition 3 works as follows: $Mallory$ chooses $\bar{x} \xleftarrow{\$} \mathbb{Z}_q^*$ and forwards $\bar{y} = y + \bar{x}P$ to Peggy, and in the second phase forwards $\bar{z} = z \cdot e(aP, bP)^{-\bar{x}}$ to Victor. The attack works since

$$\begin{aligned}\bar{z} &= e(P, \bar{y})^{ab} \cdot e(aP, bP)^{-\bar{x}} = e(P, (x + \bar{x})P)^{ab} \cdot e(aP, bP)^{-\bar{x}} \\ &= e(aP, bP)^{x+\bar{x}} \cdot e(aP, bP)^{-\bar{x}} = e(aP, bP)^x,\end{aligned}$$

just as desired.

7 Computational Protocol - First Variant

7.1 Description

Using a different functional requirement for computational hard functions (see Definition 10), we describe in Figure 9 a protocol that is secure against sequential and active-intruder attacks, as long as the parameters are selected correctly.

Definition 10 (Complete Computational Hardness Assumption). *A function f is a complete computational hard function if in Definition 1, Item 3 is changed to*

3. *there exists a PPT algorithm B such that*

$$Pr[f_k(z) = y \text{ iff } z \in Z \mid k \xleftarrow{\$} K; x \xleftarrow{\$} D_k; y \leftarrow f_k(x); Z \leftarrow B(k, y)] = 1.$$

Remark 16. A class of computational problems that satisfy the completeness property are e-th root problems for which e is not coprime with $\varphi(N)$. This class includes the square root assumption. Note that for this class $|Z| \geq 1$.

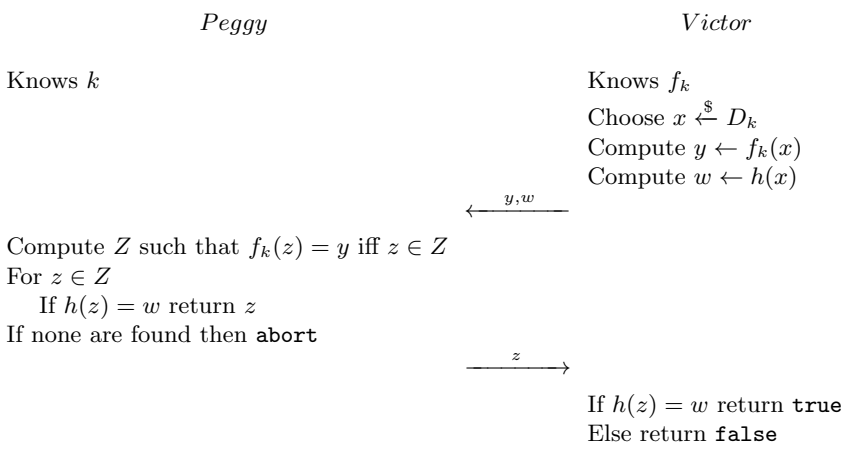


Fig. 9. First Variant of the Computational Sherlock Holmes (CSH1) Protocol.

Remark 17. In [32] the authors introduce a version of the DHCSH0 protocol (further denoted by DHCSH1) in which instead of sending y the verifier sends $(y, h(r^x))$, where h is a hash function. Stinson and Wu [32] prove that their protocol is secure against active-intruders and sequential attacks in the random oracle model under the knowledge-of-exponent assumption. We refer the reader to [32] for the details.

7.2 Security Analysis

Theorem 8. *The CSH1 protocol is secure against sequential impersonation attacks in the random oracle model.*

Proof. The proof is similar to Theorem 5, and thus we only provide a sketch. In the first phase, we must simulate the hash oracle (see Algorithm 8) and the prover (see Algorithm 13). We can see that simulator $\mathcal{O}_P$ is identical with P except when it aborts on correct challenges or responds with a $z_i' \neq z_i$ that gives the correct hash $h(z_i') = w$. These events happen if and only if w_i is not a reply to a hash oracle query or there exists an $q_j \neq x_i$ such that $h_j = h(x_i)$. Hence, they happens with probabilities $|Z|/2^\delta$ and less than $q_h/2^\delta$, and thus is negligible. Therefore, the probability of ending phase one with success is greater than $1 - (|Z| + q_h)q_p/2^\delta$.

Algorithm 13. Prover P simulator $\mathcal{O}_P$.

Input: A challenge query (y_i, w_i) from A
if $\exists\{q_j, h_j\} \in T$ *such that* $f_k(q_j) = y_i$ **then**
  return q_i
else
  return $\perp$

In the second phase of the attack, A interacts with the prover an tries to impersonate P. A PPT algorithm $\mathcal{O}_V$ is described in Algorithm 14. Since h is a random oracle and y is CHA challenge, then A will always accept (y, w). We can see that the probability of A not aborting is $1/2^\delta$. In this case $\mathcal{O}_V$ has a non-negligible advantage $ADV_f^{\text{CHA}}(O_V) = ADV_{P,V}^{\text{IMP-CA}}(A)$. If A aborts, then the correct answer is found in T. In this case $\mathcal{O}_V$ has a non-negligible advantage $ADV_f^{\text{VDHA}}(O_V) = ADV_{P,V}^{\text{IMP-CA}}(A)$. Let $\zeta \leftarrow \max_{Z \leftarrow B(k,y)} |Z|$ for any $k \leftarrow K$ and $y \leftarrow R_k$. Therefore, the total advantage of $\mathcal{O}_V$ is

$$\begin{aligned} ADV_f^{\text{VDHA}}(O_V) &\geq (1 - (|Z| + q_h)q_p/2^\delta) \cdot ADV_{P,V}^{\text{IMP-CA}}(A) \\ &\geq (1 - (\zeta + q_h)q_p/2^\delta) \cdot ADV_{P,V}^{\text{IMP-CA}}(A), \end{aligned}$$

which is non-negligible.

Algorithm 14. Verifier V simulator $\mathcal{O}_V$.

Input: A element $y \leftarrow f_k(x)$
Choose $w \xleftarrow{\$} \{0,1\}^\delta$
Send (y, w) to A
if A *sends the* `abort` *signal* **then**
  Search $\{q_i, h_i\} \in T$ such that $f_k(q_i) = y$
  return q_i
else
  Receive z from A
  return z

□

Theorem 9. *The CSH1 protocol is secure against active-intruder attacks in the random oracle model.*

Proof. The proof is similar to Theorem 6 and thus we only point out the differences. In this case, the three bit strings to indicate which of (y, w, z) are not fatefully relayed, instead of (y, w, v'). Let $\bar{y}$, $\bar{w}$ and $\bar{z}$ denote the altered items. We distinguish the following possible cases

case 001: If Victor accepts the proof, that means that A has found an element $z' \neq z$ such that $h(z') = w$. Since h is random oracle, that happens with probability $1/2^\delta$.

case 010: The prover will not reject with the negligible probability $(|Z|-1)/2^{\delta}$. If this happens, then the prover will reject since $w' = h(z') \neq w$. Otherwise, if the prover rejects, then the prover will also reject.

case 011: As in the case of Theorem 6, A becomes a concurrent impersonator and according to Theorem 8 the prover will reject.

case 100: Let $\bar{y} \in \bar{Z}$. Since h is random oracle, the probability of obtaining a collision such that $h(x) = h(\bar{x})$ is $(|\bar{Z}|/2^k$, where $f_k(x) = y$ and $f_k(\bar{x}) = \bar{z} \in \bar{Z}$. This implies that the prover will reject with non-negligible probability.

case 101: As in the case of Theorem 6, A becomes a concurrent impersonator and according to Theorem 8 the prover will reject.

case 110: In this case the prover will reject with overwhelming probability since the probability of obtaining a correct hash is $|\bar{Z}|/2^{\delta}$. Therefore, the prover will also reject.

case 111: As in the case of Theorem 6, A becomes a concurrent impersonator and according to Theorem 8 the prover will reject.

To summarize, if adversary A becomes active in a session, then the verifier will most certainly reject the proof. □

8 Computational Protocol - Second Variant

8.1 Description

Using a different functional requirement (see Definition 11), we describe in Figure 10 a protocol that is more efficient than CSH1 for some computational problems, while remaining secure against sequential and active-intruder attacks.

Definition 11 (Unique Computational Hardness Assumption). *A function f is a unique computational hard function if in Definition 1, Item 3 is changed to*

3. there exists a PPT algorithm B such that

$$Pr[z = x \mid k \xleftarrow{\$} K; x \xleftarrow{\$} D_k; y \leftarrow f_k(x); z \leftarrow B(k, y)] = 1.$$

Remark 18. Two classes of computational problems that satisfy the uniqueness property are gap 2^{κ}-residuosity problems and e-th root problems for which e is coprime with $\varphi(N)$. Note that functions that satisfy the completeness property, can be transformed into unique computational hard function by imposing a special format on the correct solution.

Remark 19. A more efficient version of the Stinson-Wu protocol [32] was introduced in [36,37]. We further denote it by DHCSH2. In this variant, $Victor$ sends y, while $Peggy$ sends $h(z)$ instead of z. The authors [36,37] show that the scheme achieves the same security as their previously proposed protocol. We refer the reader to [36,37] for the details.

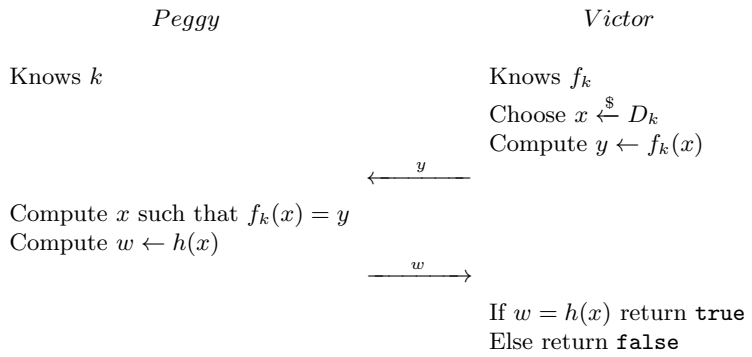


Fig. 10. Second Variant of the Computational Sherlock Holmes (CSH2) Protocol.

8.2 Security Analysis

Theorem 10. *The CSH2 protocol is secure against sequential impersonation attacks in the random oracle model.*

Proof. The proof is similar to Theorem 5, and thus we only provide a sketch. In the first phase, we must simulate the hash oracle (see Algorithm 15) and the prover (see Algorithm 16). Note that in Algorithm 16 the list T_c starts empty. We can see that simulators $\mathcal{O}_h$ and $\mathcal{O}_P$ trick A into believing that this is a real interaction with P. Therefore, phase one always ends with success.

Algorithm 15. Hashing oracle $\mathcal{O}_h$ simulation for h.

Input: A hashing query q_i from A
if $\exists h_i$ *such that* $\{q_i, h_i\} \in T$ **then**
  $e \leftarrow h_i$
else if $\exists \{y_i, w_i\} \in T_c$ *such that* $f_k(q_i) = y_i$ **then**
  $e \leftarrow w_i$
  Append $\{q_i, w_i\}$ to T
else
  $e \xleftarrow{\$} \{0,1\}^\delta$
  Append $\{q_i, e\}$ to T
return e

In the second phase of the attack, A interacts with the prover an tries to impersonate P. A PPT algorithm $\mathcal{O}_V$ is described in Algorithm 17. There is a case when $\mathcal{O}_V$ does not return the correct answer: A guesses the correct w without consulting $\mathcal{O}_h$. The probability of this happening is $1/2^\delta$. Therefore, the total advantage of $\mathcal{O}_V$ is

$$ADV_f^{\text{VDHA}}(O_V) \geq (1 - 1/2^\delta) \cdot ADV_{P,V}^{\text{IMP-CA}}(A),$$

Algorithm 16. Prover P simulator $\mathcal{O}_P$.

Input: A challenge query y_i from A
if $\exists\{q_i, h_i\} \in T$ *such that* $f_k(q_i) = y_i$ **then**
  $w_i \leftarrow h_i$
else
  $w_i \xleftarrow{\$} \{0,1\}^\delta$
  Append $\{y_i, w_i\}$ to T_c
return w_i

which is non-negligible.

Algorithm 17. Verifier V simulator $\mathcal{O}_V$.

Input: A element $y \leftarrow f_k(x)$
Send y to A
Receive z from A
Search $\{q_i, h_i\} \in T$ such that $h_i = z$ and $f_k(q_i) = y$
return q_i

□

Theorem 11. *The CSH2 protocol is secure against active-intruder attacks in the random oracle model.*

Proof. In this case, we use two bit strings to indicate which of (y, w) are not fatefully relayed. Let $\bar{y}$ and $\bar{w}$ denote the altered items. We distinguish the following possible cases

case 01: Since $\bar{w} \neq w = h(x)$ the verifier will automatically reject.
case 10: Let $\bar{x}$ such that $f_k(\bar{x}) = \bar{y}$. The prover will send $w' = h(\bar{x})$, which with probability $1 - 1/2^\delta$ is not equal to w. Therefore, Victor rejects the proof.
case 11: The prover will reject as in the previous case, and thus A will not get any useful information from interacting with P. If A manages to make the verifier accept $\bar{w}$, then he can do the same thing without interacting with P. This contradicts Theorem 10.

To summarize, if adversary A becomes active in a session, then the verifier will most certainly reject the proof. □

9 Performance of the Sherlock Holmes Protocols

In this section we compare the Sherlock Holmes protocols to some classical zero knowledge protocols such as Schnorr [29], Guillou-Quisquater [21] and Fiat-Shamir [14].

We further assume the same setup as in the case of CDH. From Figure 11 we can see that the bandwidth requirement for Schnorr's protocol is $\log_2(|\mathbb{G}| + 2q)$ bits. Similarly, for the Diffie-Hellman version of the CSH0 and CSH2 protocols we obtain a requirement of $\log_2(2|\mathbb{G}|)$ and $\log_2(|\mathbb{G}|) + \delta$ bits. In practice, $\mathbb{G}$ is either $\mathbb{Z}_p^*$, where $q = (p-1)/2$ is a prime or an elliptic curve $E(\mathbb{Z}_p)$ such that $|E(\mathbb{Z}_p)| = hq$, where $h \leq 4$. Also, in the case of $\mathbb{Z}_p^*$ we have $\delta = \log_2(q)$ and for elliptic curves we have $\delta = \log_2(|\mathbb{G}|)$. Thus, in the modulo p case we obtain $4q + 1$ versus $4q + 2$ or $3q + 1$ and in the elliptic curve case $(h + 2)q$ versus $2hq$. Thus, in most cases, our protocol's requirements are either the same or slightly lower. From a computational point of view, it is easy to see that both protocols have their complexity dominated by three exponentiations.

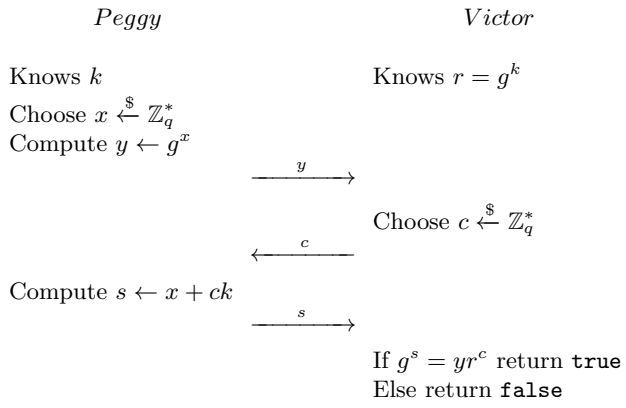


Fig. 11. Schnorr's Protocol.

Remark 20. Okamoto's protocol [25] can be seen as a vectorized version of Schnorr's protocol with $n = 2$. Thus, we can conclude that a vectorized version of DHCSH0 has slightly lower requirements as Okamoto's protocol. If we consider the security provided by Okamoto's protocol and DHCSH2, we see that both are secure against concurrent attacks. Therefore, vectorizing DHCSH2 is not necessary, and thus we obtain a speed-up of 2x.

Using Figure 11 as a reference, we further describe the Guillou-Quisquater (GQ) protocol. Assuming the setup from ER we set $r \equiv k^e \bmod N$. In the first phase, *Peggy* chooses $x \xleftarrow{\$} \mathbb{Z}_N^*$ and computes $y \equiv x^e \bmod N$. Then *Victor* randomly selects $c \xleftarrow{\$} [0, e-1]$. The third step consists of *Peggy* computing $s \equiv xk^c \bmod N$. Then *Victor* accepts the proof if an only if $s^e \equiv yr^c \bmod N$.

The bandwidth requirement for the GQ protocol is $\log_2(2N + e)$, while for the e-th root instantiation of CSH0 and CSH2 are $loq_2(2N)$ and $loq_2(N) + \delta$. In practice we have $log_2(N)$ is 12/20/30 times larger than δ. Hence, the

requirements are similar to CSH0 only if e is small and almost two times higher compared to CSH2. From a computational point of view, CSH0 and CSH2's time is dominated by two exponentiations, while GQ's time by four. So, our protocol is twice as fast. Also, note that the probability of impersonating $Peggy$ is $1/e$ for GQ, while for our protocols is in the worse case $e^2/\varphi(N)$ [9].

The Fiat-Shamir protocol [14] considers $e = 2$. Let $n = 2$. If we consider $MDSH$ instantiated with DDH, we obtain a bandwith requirement of $\log_2(|\mathbb{G}|)$, a complexity dominated by three exponentiations and a probability of impersonating $Peggy$ of 1/2. Let $\mathbb{G} = \mathbb{Z}^*_{p'}$, when p' is prime[10]. Using the reasoning from the GQ protocol, we obtain that the $MDSH$ protocol has a better performance that the Fiat-Shamir, while having the same security.

10 Conclusions

Our two main zero knowledge protocols, decisional and computational Sherlock Holmes protocols, represent two new large classes of protocols. The presented list of examples is by no means exhaustive. Our next challenge is to see how we can adapt these protocols in order to obtain new cryptographic primitives (*e.g.* non-interactive zero knowledge proofs or digital signatures).

References

1. Barak, B., Lindell, Y., Vadhan, S.: Lower Bounds for Non-Black-Box Zero Knowledge. Journal of Computer and System Sciences **72**(2), 321–391 (2006)
2. Bellare, M., Fischlin, M., Goldwasser, S., Micali, S.: Identification Protocols Secure against Reset Attacks. In: EUROCRYPT 2001. Lecture Notes in Computer Science, vol. 2045, pp. 495–511. Springer (2001)
3. Bellare, M., Goldwasser, S.: Lecture Notes on Cryptography. https://cseweb.ucsd.edu/~mihir/papers/gb.pdf (2008)
4. Bellare, M., Palacio, A.: GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks. In: CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 162–177. Springer (2002)
5. Bellare, M., Rogaway, P.: Introduction to Modern Cryptography. https://web.cs.ucdavis.edu/~rogaway/classes/227/spring05/book/main.pdf (2005)
6. Benhamouda, F., Herranz, J., Joye, M., Libert, B.: Efficient Cryptosystems from 2^k-th Power Residue Symbols. Journal of Cryptology **30**(2), 519–549 (2017)
7. Chatterjee, S., Sarkar, P.: Practical Hybrid (Hierarchical) Identity-Based Encryption Schemes Based on the Decisional Bilinear Diffie-Hellman Assumption. IJACT **3**(1), 47–83 (2013)
8. 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)

[9] According to Lagrange's theorem the polynomial x^e has at most e solution modulo p.

[10] In practice, for security reasons, N and p' have similar lengths.

9. Cocks, C.: An Identity Based Encryption Scheme Based on Quadratic Residues. In: IMACC 2001. Lecture Notes in Computer Science, vol. 2260, pp. 360–363. Springer (2001)
10. Damgård, I.: Towards Practical Public Key Systems Secure Against Chosen Ciphertext Attacks. In: CRYPTO 1991. Lecture Notes in Computer Science, vol. 576, pp. 445–456. Springer (1991)
11. Feige, U., Fiat, A., Shamir, A.: Zero-Knowledge Proofs of Identity. Journal of Cryptology **1**(2), 77–94 (1988)
12. Feige, U., Shamir, A.: Zero Knowledge Proofs of Knowledge in Two Rounds. In: CRYPTO 1989. Lecture Notes in Computer Science, vol. 435, pp. 526–544. Springer (1989)
13. Feige, U., Shamir, A.: Witness Indistinguishable and Witness Hiding Protocols. In: STOC 1990. pp. 416–426. ACM (1990)
14. 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)
15. 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)
16. Goldreich, O.: Zero-Knowledge Twenty Years After Its Invention. IACR Cryptology ePrint Archive **2002/186** (2002)
17. Goldreich, O., Krawczyk, H.: On the Composition of Zero-Knowledge Proof Systems. SIAM Journal on Computing **25**(1), 169–192 (1996)
18. Goldreich, O., Oren, Y.: Definitions and Properties of Zero-Knowledge Proof Systems. Journal of Cryptology **7**(1), 1–32 (1994)
19. Grigoriev, D., Shpilrain, V.: No-Leak Authentication by the Sherlock Holmes Method. Groups Complexity Cryptology **4**(1), 177–189 (2012)
20. Guillou, L.C., Quisquater, J.J.: A "Paradoxical" Indentity-Based Signature Scheme Resulting from Zero-Knowledge. In: CRYPTO 1988. Lecture Notes in Computer Science, vol. 403, pp. 216–231. Springer (1988)
21. 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)
22. Maimuţ, D., Teşeleanu, G.: A Generic View on the Unified Zero-Knowledge Protocol and Its Applications. In: WISTP 2019. Lecture Notes in Computer Science, vol. 12024, pp. 32–46. Springer (2019)
23. Maurer, U.: Unifying Zero-Knowledge Proofs of Knowledge. In: AFRICACRYPT 2009. Lecture Notes in Computer Science, vol. 5580, pp. 272–286. Springer (2009)
24. Niven, I., Zuckerman, H.S., Montgomery, H.L.: An Introduction to the Theory of Numbers. John Wiley & Sons (1991)
25. 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)
26. Okamoto, T., Pointcheval, D.: The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In: PKC 2001. Lecture Notes in Computer Science, vol. 1992, pp. 104–118. Springer (2001)
27. Ostrovsky, R.: Foundations of Cryptography. http://web.cs.ucla.edu/~rafail/PUBLIC/OstrovskyDraftLecNotes2010.pdf (2010)
28. Sahai, A., Vadhan, S.: A Complete Problem for Statistical Zero Knowledge. Journal of the ACM) **50**(2), 196–249 (2003)

29. 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)
30. Schnorr, C.P.: Efficient Signature Generation by Smart Cards. Journal of cryptology **4**(3), 161–174 (1991)
31. Stinson, D.R.: Cryptography: Theory and Practice. Chapman and Hall/CRC (2006)
32. Stinson, D.R., Wu, J.: An Efficient and Secure Two-Flow Zero-Knowledge Identification Protocol. Journal of Mathematical Cryptology **1**(3), 201–220 (2007)
33. Teşeleanu, G.: Unifying Kleptographic Attacks. In: NordSec 2018. Lecture Notes in Computer Science, vol. 11252, pp. 73–87. Springer (2018)
34. Teşeleanu, G.: Lightweight Swarm Authentication. In: SecITC 2021. Lecture Notes in Computer Science, vol. 13195, pp. 248–259. Springer (2021)
35. Teşeleanu, G.: Sherlock Holmes Zero-Knowledge Protocols. In: ISPEC 2022. Lecture Notes in Computer Science, vol. 13620, pp. 573–588. Springer (2022)
36. Wu, J., Stinson, D.R.: An Efficient Identification Protocol and the Knowledge-of-Exponent Assumption. IACR Cryptology ePrint Archive **2007/479** (2007)
37. Wu, J., Stinson, D.R.: An Efficient Identification Protocol Secure Against Concurrent-Reset Attacks. Journal of Mathematical Cryptology **3**(4), 339–352 (2009)

A Active-Intruder Attacks

In this section we provide an active-intruder attack for the zero-knowledge protocol introduced in [22]. This protocol is a generalization of Maurer's unified zero-knowledge protocol [23], which, depending on the instantiation, can be transformed into either the Schnorr protocol [29][11] or the Okamoto protocol [25] or the Fiat-Shamir protocol [14] or the Guillou-Quisquater protocol [21]. Note that the protocol from [22], also generalizes Feige-Fiat-Shamir's [11] and Chaum-Everste-Van De Graaf's [8] protocols. More instantiations can be found in [22,23,33]. A direct consequence of our attack is that it supersedes the active-intruder attacks introduced in [32] for the Schnorr, Fiat-Shamir, Okamoto and Guillou-Quisquater protocols.

A.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 $f : \mathbb{G} \to \mathbb{H}$ be a function that is one-way[12] and not necessarily one-to-one. We say that f is a homomorphism if $f(x \star y) = f(x) \otimes f(y)$. We further 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.

[11] and its variation introduced by Girault [15]
[12] *i.e.* it is infeasible to compute x from $f(x)$

A.2 Protocol

Let n be a positive integer and let $i \in [1, n]$. In Figure 12 we present the protocol introduced in [22] that enables $Peggy$ to prove to $Victor$ that she knows a vector $\{[x_i]\}_{i\in[1,n]}$ such that $z_i = [x_i]$, where $\{z_i\}_{i\in[1,n]}$ is a public vector. Note that $\mathcal{C}$ denotes the challenge space for the elements c_i and is an arbitrary subset of $\mathbb{N}$.

$Peggy$		$Victor$
Knows $\{x_i\}_{i\in[1,n]}$. Computes $\{z_i\}_{i\in[1,n]} = \{[x_i]\}_{i\in[1,n]}$.		Knows $\{z_i\}_{i\in[1,n]}$.
Choose $k \xleftarrow{\$} \mathbb{G}$. Compute $t \leftarrow [k]$.		
	$\xrightarrow{t}$	
		Choose $c = \{c_i\}_{i\in[1,n]} \xleftarrow{\$} \mathcal{C}^n \subset \mathbb{N}^n$.
	$\xleftarrow{c}$	
Compute $r \leftarrow k \star (\star_{i=1}^{n} x_i^{c_i})$.		
	$\xrightarrow{r}$	
		If $[r] = t \otimes (\otimes_{i=1}^{n} z_i^{c_i})$ return `true`. Else return `false`.

Fig. 12. A Unified Generic Zero-Knowledge (UGZK) Protocol.

A.3 Attack

In order to succeed, the attacker $Mallory$ first chooses at random $k' \xleftarrow{\$} \mathbb{G}$ and computes $[k']$. When $Peggy$ sends her first message, $Mallory$ intercepts it and forwards $t' = t \otimes [k']$ to $Victor$ (see Figure 13). The second message is simply forwarded by $Mallory$. In the case of the third message, $Mallory$ intercept it and forwards $r' = r \star k'$. We can see that $Mallory$'s attack succeeds since

$$[r'] = [r \star k'] = [r] \otimes [k'] = t \otimes (\otimes_{i=1}^{n} z_i^{c_i}) \otimes [k'] = t' \otimes (\otimes_{i=1}^{n} z_i^{c_i}),$$

just as required by $Victor$.

$Peggy$	$Mallory$	$Victor$
$\xrightarrow{t}$	$\xrightarrow{t\otimes[k']}$	
$\xleftarrow{c}$	$\xleftarrow{c}$	
$\xrightarrow{r}$	$\xrightarrow{r\star k'}$	

Fig. 13. Active-Intruder Attack Against UGZK.