Fast and Secure Immunization against Adaptive Man-in-the-Middle Impersonation

Ronald Cramer (ETH Zurich *) and
Ivan Damgård (Aarhus University ** & BRICS ***)

Abstract. We present a simple method for constructing identification schemes resilient against impersonation and man-in-the-middle attacks. Though zero-knowledge or witness hiding protocols are known to withstand attacks of the first kind, all such protocols previously proposed suffer from a weakness observed by Bengio *et al.* : a malicious verifier may simply act as a moderator between the prover and yet another verifier, thus enabling the malicious verifier to pass as the prover.
We exhibit a general class of identification schemes that can be efficiently and securely tranformed into identification schemes withstanding an adaptive man-in-the-middle attacker. The complexity of the resulting (witness hiding) schemes is roughly twice that of the originals. Basically, any three-move, public coin identification scheme that is zero knowledge against the honest verifier and that is secure against passive impersonation attacks, is eligible for our transformation. This indicates that we need only seemingly weak cryptographic intractability assumptions to construct a practical identification scheme resisting adative man-in-the-middle impersonation attacks. Moreover, the required primitive protocols can efficiently be constructed under the factoring or discrete logarithm assumptions.

1 Introduction

An (public key) identification scheme (see for instance [9]) is an (interactive) protocol by means of which one party (the prover) proves its identity to another party (the verifier). Securing log-in procedures is a main application of such schemes. An identification scheme consists of an algorithm to generate public-key/private-key pairs, and a protocol for the prover and the verifier. The collection of eligible key-pairs is chosen such that it is infeasible to compute a corresponding private key when only the public key is observed. Typically, the protocol's purpose is to show that the prover "knows" the private key that corresponds to the prover's public key. Most known identification schemes take the

* Inst. for Theoretical Comp. Sc., ETH Zurich, CH-8092 Zurich, Switzerland. Email: `cramer@inf.ethz.ch`. Research done while employed at CWI, Amsterdam, The Netherlands.
** Maths. & Comp. Sc. Dept., Ny Munkegade, Aarhus, Denmark. Email: `ivan@daimi.aau.dk`
*** Basic Research in Computer Science, Center of the Danish National Research Foundation

form of three move interactive where the verifier is required to send a random bitstring as a challenge. For such methods to be secure, the verifier must not be able to extract this private key from the prover. Formally, this notion of security is captured by considering *adaptive impersonation attacks.* The (probabilistic polynomial time) attacker is given a prover, who has access to a key-pair as produced by the key-generation algorithm, as a black-box. Thus, the attacker only sees the prover's outputs as dictated by the identification protocol and not any of its internal coinflips, private inputs, etc. Next, the attacker is allowed to query the black-box a polynomial number of times, playing the role of a (malicious) verifier. This means that the attacker is allowed to choose the challenges in any way thought suitable to extract information about the private key. In particular, the choice of any next challenge may depend on the entire history of the attack and public key. Next, the attacker is denied any further access to this black-box prover. The identification scheme is called secure against adaptive impersonation attacks if the attacker is still unable to impersonate the prover (execute the prover's part of the protocol, facing an honest verifier).

In [4] a weakness of identification schemes proposed until then was exposed. There, the authors explained how a malicious *man-in-the-middle* $\tilde{V}$ may abuse his conversations with an honest prover P to misrepresent himself as P to yet another verifier V. The attack is not by cryptographic ingenuity. But, simply pretending to be a verifier himself, $\tilde{V}$ actually forwards V's challenges to P and forwards P's replies to V. Thus, while P is under the impression that he is identifying himself to $\tilde{V}$, he is actually identifying himself to V, to the possible advantage of $\tilde{V}$. A remedy suggested in [4] has the prover and verifier (rather the devices that represent them) isolate themselves physically from the outside world. A Faraday's cage could be a suitable implementation. However, for identification over networks, for instance, this measure seems not to be useful.

We present a simple method to construct identification schemes resilient against adaptive impersonation *and* man-in-the-middle attacks. Though zero-knowledge [13] or witness hiding protocols [10] are known to withstand attacks of the first kind, all such protocols previously proposed suffer from the weakness observed by Bengio e.a. [4], since a malicious verifier may simply act as a moderator between the prover and yet another verifier, thus enabling the malicious verifier to pass as the prover. Using a three-move public coin protocol that is collision intractable (without knowing the private key, it is infeasible to pass the protocol) and honest verifier zero knowledge we build a witness-hiding identification scheme that differs from previous proposals in that an execution of a given proof of identity can only be unambiguously appreciated by the intended verifier. This is achieved by having the prover direct the protocol to the intended verifier's public key. It is consequently shown that resilience against man-in-the-middle-attacks follows from this approach. Note that the required primitive protocol corresponds to an identification scheme secure against passive impersonation and honest verifiers. Directing a proof to an intended verifier has been considered by other researchers in a different context, as we will explain later. Our contribution is to provide a general, secure and efficient immunization against adaptive man-in-the-middle impersonation attacks in identification schemes. Furthermore, we want the immunization to work even if the the orginal identification scheme satisfies only weak security properties.

Example schemes that satisfy our requirements include Schnorr's scheme based on discrete logarithms [18] or Guillou-Quisquater's scheme based on RSA [15]. But more generally, any one-way group homomorphism or any pair of claw-free trapdoor permutations gives rise to the desired building block. If we would take, for example, Schnorr's scheme [18] as input to our constructions, the resulting identification scheme would have twice the complexity (in terms of computation and communication) of [18]. But we are then able to prove that our scheme is witness-hiding and resilient against man-in-the-middle attacks if computing discrete logarithms is hard.

Conceptually, our method to disable man-in-the-middle attacks is as follows. Let X and Y be two players, where X wishes to identify himself to Y. Suppose now that we have an efficient method by which X could take Y's public key, and his own key-pair (his public key and secret key), and securely prove the statement "I know X's secret key or I know Y's secret key". If this protocol is *witness indistinguishable* (no information is released as to which is the case), only Y can be sure he is talking to X rather than anyone else. For, any other verifier Z would only know that he is talking to X *or* Y. Thus, if X directs his proof to Y as outlined above, the proof is unambiguous only to Y.

So why would this help against man-in-the-middle attacks? By the symmetry of the statement proved and by the asserted witness-indistinguishability of the proof, if Y could abuse his conversation with X to pass as X at Z as the man-in-the-middle would do, he must be able to do so without talking to X. Thus the man-in-the-middle attack reduces to a cryptographic attack. But now we invoke the witness-indistinguishability again to show that if Y's attack would succeed, he could compute X's secret key. This then contradicts our assumption that it is hard to compute the secret key from a random public key. We stress that this approach makes sense only if the keys are sufficiently indepedently generated. In the extreme case that two verifier keys are identical, it is clear that man-in-the-middle attacks are still feasible. More generally, a proof of security will fail if there is dependendence among these keys: if one is chosen as a clever function of the other (such as a random and secret power of a given key based on discrete logarithms), proof given to one verifier may still be "diverted" to another verifier. In Sections 6 and 7 we discuss this matter in detail and give examples of how proper key-generation can be enforced.

We note that the same basic idea of proving one of two statements in order to direct a proof to one specific verifier was found independently by Jacobson, Impagliazzo and Sako in [16]. Their main motivation was to make undeniable signature schemes more secure and non-interactive. Their method for building a verifier designated protocol uses a trapdoor bit commitment scheme. In comparison, our method shows that if you start with a protocol of a certain form, then a separate trapdoor bit commitment is not needed. On the other hand, their methods works for some protocols that are not of the form we consider. We also note that, in a different context, Chaum [5] proposed using trapdoor commitment schemes to ensure that only a particular verifier can appreciate a given proof. Dolev, Dwork and Naor [8] have introduced *non-malleable cryptography*, a theoretical primitive that includes prevention of man-in-the-middle attacks in a number of scenarios, and have proposed protocols that work under general cryptographic assumptions.

It is not so much the concept explained above that we advocate as the most significant contribution here. We would like to stress that the concept has been applied implicitly before, prior to [16]. [16] is the first paper applying the ideas to verifier-directed proofs, however. We know of at least one example, namely the protocol of Feige and Shamir [12] for bounded round general zero knowledge proofs. There, the prover commits to a witness for the NP-statement to be proved using an unconditionally hiding *trapdoor commitment scheme*, an instance of which is generated by the verifier. Indeed, the proof conducted there can be seen as showing that the NP-statement is true, or that the prover knows the verifier's trapdoor! To get the designated verifier proofs for general languages, postulated in [16] but not given, we can use the result of [12] and make sure that verifiers' instances of the trapdoor commitment scheme are independently generated.

In our setting, we restrict ourselves to the problem of identification, and attempt to formulate a very efficient solution to the problem of identification in the presence of an *adaptive man-in-the-middle attacker*. Moreover, we are only interested in solutions that allow for some well-defined and accepted cryptographic intractability assumption to be reduced to the security of the identification scheme.

It is interesting to note that our results apply to a general class of identification schemes which in their normal mode of operation need only satisfy seemingly weak security properties. Namely, zero knowledge with respect to the honest verifier and collision intractability (that is, the scheme is secure against passive impersonation attacks). As a result of our simple and efficient transformation, we obtain the required security level, namely *security against adaptive man-in-the-middle attackers*.

Technically speaking, our approach is close to the ones taken in [7,6]. However, it is not clear from those papers (which may partly be seen as investigations into witness hiding) how we can efficiently obtain security against adaptive man-in-the-middle attackers in our context. Please note that such was neither clear from [16], since there the focus is on undeniable signatures. Although it appears to be true that their approach using trapdoor bitcommitments has a wider applicability than that, their approach does not indicate that immunization of an identification scheme against man-in-the-middle attackers, can be done efficiently and securely *even if* the given scheme is only weakly secure in normal mode of operation, as we discussed above.

Please note that digital signatures also lead to identification schemes secure against impersonation and man-in-the middle attacks. The prover would simply sign a message consisting of the concatenation of a random challenge (supplied by the verifier) and the verifier's public key. Although we feel that our schemes could compare favorably in terms of practical value to even such solutions, we like to point out that we aim for a practical identification scheme that is proven secure if some standard cryptographic intractability assumption holds. Seen in this light, digital signatures, for example, with such proven security, i.e. signatures secure against adaptively chosen message attacks, still come at too high a price in this context. Nevertheless, it may be reasonable here to use them for key-certification. Note that in this signature based approach, the prover (in this case the signer) leaves a trace: the verifier can later prove to a third party that he talked to the prover. In some cases this is undesirable as it might damage the privacy of the prover. This problem is not present in our approach: because the verifier could (using his own secret key) simulate the protocol perfectly, he cannot use a transcript of the protocol to convince a third party.

If one aims at practical value *and* proven security (relative to a plausible assumption), it may be true that our proposal for identification schemes secure against impersonation and man-in-the-middle attacks comes close to what one could reasonably achieve in this area, due to its conceptual simplicity and efficient implementation.

This work is organized as follows. First, we define a general class of "weak" identification schemes in Section 2, to be used later as the building block for our transformation. The existence of our building blocks is discussed in Section 5. The main result and its proof of security are given in Sections 3 and 4. Section 6 discusses in detail the key-generation requirements. Finally, we give an application to *access-control* in Section 7.

2 Model

We define the basic ingredients to our results.

***Σ*-Protocols** Let (A, B) be a three move protocol where the prover A speaks first. The verifier B is required to send random bits only. A and B are probabilistic polynomial time (PPT) machines. The protocol (A, B) resembles a proof of knowledge for a binary relation R (see for instance [9] for details), in that the prover can always make the verifier accept on common input x, if the prover knows w such that $(x, w) \in R$. By running (probabilistic) polynomial time algorithm $a(\cdot)$ on x and his secret witness w, the prover A computes his initial message a. After having received the initial message, the verifier B chooses a bitstring $c \in \{0,1\}^{t_B}$ uniformly at random, and sends it as a challenge to A. The challenge length t_B is assumed to depend only on the binary length of the common input x (and the protocol (A, B) of course). The prover completes the conversation by running (probabilistic) polynomial time algorithm $z(\cdot)$ on x, w, a, c, thereby possibly re-using the random bits used in the computation of the initial message. The resulting response z is submitted to the verifier. By invoking the (probabilistic) polynomial time procedure ϕ, the verifier tests the validity of the conversation. We call such a protocol (A, B) with the properties described above a Σ-protocol [1] for relation R.

Furthermore, we introduce the following terminology and notation. A sequence (x, a, c, z) is called an accepting conversation if and only if $\phi(x, a, c, z) = accept$. A pair of accepting conversations (x, a, c, z) and (x, a, c', z') with $c \neq c'$, is called a *collision*. When a verifier B follows the protocol, i.e. chooses the challenge indeed at random, that verifier is called *honest*. For an arbitrary prover A^*, (A^*, B) denotes the interaction between A^* and the honest verifier B, on some given common input.

Required Security Properties First, we need the protocol to satisfy a weak form of knowledge-soundness.

Definition 1. Let k be a security parameter for protocol (A, B). Suppose we are given a PPT generator G for relation R that on input 1^k produces $(x, w) \in R$, such that no PPT algorithm E, given x as input, can generate two accepting conversations (a, c, r), (a, c', r') with $c \neq c'$ (a "collision for x"), except with negligible probability of success (probability taken over the coinflips of E and G). Then (A, B) is called *collision intractable* over G.

Note that we don't require that a witness can be extracted from a successful prover. Thus, the protocol need not be a proof of knowledge. The property implies that, given as input a random instance x only, it is infeasible to construct a successful prover for that instance. In particular it follows from our assumptions that it must be hard to compute a witness w from a given x (when x is generated according to G). By a standard rewinding argument (see Bellare and Goldreich [3]), we have the following.

Proposition 2. *Let a Σ-protocol (A, B) for relation R be given, and let $x \in \{0,1\}^*$. Suppose that A^* is an arbitrary PPT prover such that (A^*, B) succeeds with probability ϵ, on common input x. Let $T_{A^*}(x)$ be A^*'s running time and suppose that $\epsilon > 1/2^{t_B}$. Then there exists a probabilistic algorithm Ext that outputs two accepting conversations x, a, c, z and x, a, c', z' with $c \neq c'$ (that is, a collision), with expected running time polynomial in $T_{A^*}(x)$ and $1/(\epsilon - 1/2^{t_B})$. Ext is allowed to run A^* as a rewindable blackbox. The probability is taken over the coin tosses of Ext and A^*.*

Next, we will assume the protocol (A, B) to be honest verifier zero-knowledge, that is, we only demand that conversations with the honest verifier can be simulated (perfectly).

[1] Of course, there is nothing new about three move, public coin protocols as such in cryptography, but we have decided to give them a name, derived from *zig-zag* and *Merlin-Artur* (see [2])

Definition 3. Let $(x, w) \in R$. Let a prover A and a verifier B execute (A, B), both following the protocol. Let x be the common input and let w be private input to the prover. Suppose we are given a probabilistic polynomial time algorithm M with the following properties.

1. On input x, M outputs an accepting conversation.
2. The distribution of the conversations generated by A and B is equal to $M(x)$.

Then (A, B) is said to satisfy *honest verifier zero knowledge*, with *simulator* M.

Relation with Identification Schemes We can view a Σ-protocol (A, B) for relation R as an identification scheme by identifying a public/private key-pair with a pair $(x, w) \in R$, as generated by some given generator G.

It is easy to see that such a protocol constitutes an identification scheme *secure against passive attacks*, if (A, B) is collision-intractable over G and if the length t_B of the challenges is large enough, say linear in the security parameter. Indeed, by Proposition 2, we can extract collisions with non-negligible probability from a passive attacker (that is, one which is given the public x only) having non-negligible probability of success. But this would contradict our assumption that (A, B) is collision-intractable over G.

Adding honest verifier zero knowledge to our requirements, makes sure that the resulting scheme is *secure against random challenge atacks*. By this we mean that even an attacker which is allowed to query a prover on *random* challenges, cannot *later* pose as that prover. Note that we use here the previous observation that collision-intractability implies security against passive attacks.

Security against adaptive attacks means that even though the attacker is allowed to query a prover on *any* challenge of his choice and in an adaptive fashion, it can still not later pose as that prover. This is basically the notion of security from [11].

The *adaptive man-in-the-middle attacker*, is one which has "adaptive access" to a prover X as well. Additionally however, the attacker is allowed to pose as any verifier $\tilde{Y}$ out of a given set V of verifiers, and have X identify itself to this verifier. The attacker's goal is to make an honest verifier Y, with $Y \notin V$, accept X, possibly running executions of X's identification to any $\tilde{Y} \in V$ on-line. If this is infeasible for any PPT attacker, we say that the identification scheme is *secure against adaptive man-in-the-middle impersonation*. Note that our definition combines the notions of security from Feige *et al.* [11] and Bengio *et al.* [4].

Our purpose is to transform identification schemes that are only secure against random challenge attacks into ones that withstand even adaptive man-in-the-middle impersonation, which seems to be the most desirable security level for public key identification schemes.

3 Main Result

Let (A, B) be a collision-intractable Σ-protocol for relation R and generator G. Suppose that (A, B) is honest verifier zero-knowledge, with simulator M, and that the challenge length t_B is linear in the security parameter k. Thus, by the remarks above, (A, B) constitutes an identification scheme secure against random challenge attacks. Our purpose is to transform (A, B) into a new identification scheme which is secure against adaptive man-in-the middle impersonation. This transformation works as follows.

Key Generation A keypair $(x, w) \in R$, consisting of a public key x and a secret key w, for participant X is generated as

$$(x, w) \leftarrow G(1^k)$$

for an appropriate security parameter k. The public key x is placed in X's public directory. The secret key w is held privately.

Identification of $\boldsymbol{X}$ to $\boldsymbol{Y}$ Here, participant X will identify itselfto participant Y. Let their respective public keys be x and y, and let X's secret key be w. The claimed identification protocol runs as follows.

Move 1: X computes $a \leftarrow a(x, w)$ and $(b, d, s) \leftarrow M(y)$. Then X sends the pair (a, b) to Y.

Move 2: Y selects C uniformly at random from $\{0, 1\}^{t_B}$ and sends C as a challenge to X.

Move 3: X puts $c \leftarrow C \oplus d$ and computes $z \leftarrow z(x, w, a, c)$, and sends z, d, s to Y. Finally, Y checks the conversation by verifying whether $\phi(x, a, C \oplus d, z) \stackrel{?}{=}$ *accept* and $\phi(y, b, d, s) \stackrel{?}{=}$ *accept*. If these verifications are satisfied, X is accepted by Y.

Please note that the secret key of the verifier Y is not used during the identification. One can imagine a scenario where the set of provers is disjoint from the set of verifiers. In this case, no storage of secret data is required at the verifier's side.

From a technical point of view the protocol above is quite similar to that given in Corollary 13 from [7] (while collision-intractability and honest verifier zero knowledge as a building block is taken from [6]). That result may be viewed as a way to transform identification schemes secure against random challenge attacks into ones that withstand adaptive challenge attacks only.

The cryptographic assumptions needed here are potentially weaker. But most importantly, here we show how the protocol from Corollary 13 [7] can be "rearranged" so as to withstand even man-in-the-middle attackers. Thus from the point of view of functionality, the protocol presented here is superior. Another difference is that here the length of the public key is invariant under the transformation.

4 Security Analysis

We give proof of security under the assumption that the participants' keys are generated as prescribed in the Key Generation protocol. In Section 6 we explain in detail why this assumption is needed and we also propose ways of enforcing this. An application where this condition is satisfied in a natural way is presented in Section 7.

Before we give the proof, we 'd like to point out that an execution of the protocol from Section 3 *leaves no trace*, in the sense that a verifier Y cannot later prove to a third that X identified itself to Y earlier. This follows from the symmetry of the protocol: Y can generate the conversations of the identification of X to Y with exactly the same distribution on its own.

Theorem 4. *Let (A, B) be a collision intractable, honest verifier zero knowledge Σ-protocol for relation R and generator G. Assume that the challenge length t_B is linear in the security parameter k. Then the identification scheme based on (A, B) from Section 3 is secure against adaptive man-in-the-middle impersonation.*

Proof. The idea is as follows. First we generate public key x' according to G, and discard the corresponding secret key. We show that, if the protocol were not witness hiding or were not resilient against man-in-the-middle attacks, there exists an efficient algorithm that takes x' as input and outputs a collision for x' in the protocol (A, B). But this would then contradict (A, B)'s collision-intractability.

The following game is easily be seen as modelling the situation. Let m be polynomial in the security parameter k. We generate m public keys with known secret keys by running G m times. We flip a coin b. If $b = 0$, then we put $x \leftarrow x'$ and assign the m key pairs to $Y_1 \dots Y_m$. If $b = 1$, we select j at random from $\{1, \dots, m\}$, and put $y_j \leftarrow x'$, and assign the m key pairs to X, $Y_1, \dots, Y_{j-1}, Y_{j+1}, \dots Y_m$.

The game consists of two stages.

1. The attacker gets the following prover as a black-box. We define P as the prover who gets x and all public keys y_i as input, plus the secret keys as generated above. P can perform the identification protocol for all pairs (x, y_i). The attacker is allowed to play with P (as a blackbox, but not rewindable) for a polynomial amount of time. Then, the attacker gives us a number $j' \in \{1, \dots, n\}$, and hands back P. This models the idea that before the real attack, the attacker may try to extract as much information as needed for winning in the second stage.
2. With probability $(m+1)/(2m)$, the attacker chose $j' = j$ such that P was not given the secret key for y_j in the beginning or was not given the secret key for x. Let's assume that this event happens (If not, we re-run the previous stage). Next, the attacker gets as input the secret keys for all public keys y_i with $i \neq j$. This models the idea that (possibly via a man-in-the-middle

attack), the attacker tries to pass as X to any other verifier intended by X. To make the proof easier, we just give the attacker the secret keys which allow him to perfectly simulate X's behaviour at any other site than Y_j, rather than giving him X as a blackbox: if he can't do it *with* the secret keys, than he certainly can't when he is given X as a blackbox who only identifies himself at Y_i with $i \neq j$. The attacker wins the game, if he can pass the protocol against the honest verifier on input (x, y_j).

Let's assume that the attacker won with probability $\epsilon > 2^{-t_B}$ (recall that t_B is assumed to be of linear size in k). Then, by Proposition 2, we can extract a collsion for y_j or for x from the attacker (running it as a rewindable blackbox) with expected time polynomial in the running time of the attacker and $1/(\epsilon - 1/2^{t_B})$. Thus, if ϵ is non-negligible, then we can extract a collision from the attacker in expected polynomial time. But, this is a collision for key x' with probability $1/2$, since the attacker cannot distinguish between the cases $b = 0$ and $b = 1$ by witness indistinguishability of the protocol (which follows by the properties of the simulator M). This contradicts the assumption that (A, B) is collision-intractable over G.

5 Existence

The following theorem can be derived from the results in [6], and gives an indication of the generality of our primitive.

Theorem 5. *Suppose that a family of claw-free pairs of trapdoor permutations exists, or that a family of one-way group homomorphisms exists. Then there exists a Σ-protocol for relation R, with generator G, that is collision-intractable and honest verifier zero knowledge and that has a challenge length linear in the security parameter.*

If based on claw-free pairs of trapdoor permutations, we can always efficientlyenforce the challenge length of (A, B) to be linear in the security parameter, while keeping the size of the initial message, the reply and the length of the common string constant in length. For one-way group homomorphisms, we can do something similar, under the condition that for each such homomorphism f, there exists a (large) prime v with the following property: for each y in the range of f, it is easy to compute a preimage x of y^v (using multiplicative notation for the group operation in the range). Two important examples of such families of one-way group homomorphisms can be constructed under the factoring and discrete logarithm assumptions. We give no further details of the general construction here.

A particularly efficient implementation, for example, is obtained when (A, B), for instance, is Schnorr's protocol [18] or Guillou-Quisquater's [15]. The following example is based on Schnorr's identification protocol. Let G_q be a group of prime order q such that computing discrete logarithms in G_q is hard. Let g be a fixed member of G_q.

Key Generation A keypair, consisting of a public key and a secret key, for participant X is generated as

$$(x = g^w, w)$$

where w is chosen at random from $\mathbb{Z}_q$. The public key x is placed in X's public directory. The secret key w is held privately.

Identification of X to Y Here, participant X will identify itself to participant Y. Let their respective public keys be x and y, and let X's secret key be w. The claimed identification protocol withstanding adaptive man-in-the-middle impersonation runs as follows.

Move 1: X computes $a \leftarrow g^u$ and $b \leftarrow g^s y^{-d}$, where u, s and d are chosen at random from $\mathbb{Z}_q$. Then X sends the pair (a, b) to Y.
Move 2: Y selects C at random from $\mathbb{Z}_q$ and sends C as a challenge to X.
Move 3: X puts $c \leftarrow C + d \bmod q$ and computes $z \leftarrow cw + u \bmod q$, and sends z, d, s to Y. Finally, Y checks whether $g^z \stackrel{?}{=} ax^c$ and $g^s \stackrel{?}{=} by^d$, where c is defined as $C + d \bmod q$ If these verifications are satisfied, X is accepted by Y.

6 A Note on Key-Generation

Using our example based on discrete logarithms from Section 5, we explain why it is important that key-generation takes place as demanded; if key-generation is not taken care of as required, the following attack could be mounted against the scheme. Let's assume that some malicious party $\tilde{Y}$ wishes to be accepted as any prover X by some verifier Y. Let x and y denote their respective public keys.

The attacker $\tilde{Y}$ proceeds by selecting $\alpha, \beta \in \mathbb{Z}_q$, computing $\tilde{x} \leftarrow g^\alpha y^\beta$, and defining $\tilde{x}$ as its public key. Whenever any prover X identifies itself to $\tilde{Y}$, the latter can easily divert the communication to Y and be accepted as X as follows:

Move 1: Prover X identifies itself to $\tilde{Y}$ and the attacker $\tilde{Y}$ claims to be X to verifier Y. The attacker $\tilde{Y}$ proceeds as follows. Receive a and b from X. Compute $\tilde{b} \leftarrow b^{1/\beta}$. Forward a and $\tilde{b}$ to Y.
Move 2: Receive Y's challenge C, and forward it to X.
Move 3: Receive X's replies z, d and s. Compute $\tilde{s} \leftarrow (s - \alpha d)/\beta \bmod q$, and forward z, $\tilde{s}$ and d to Y, who checks that $g^z \stackrel{?}{=} ax^c$ and $g^{\tilde{s}} \stackrel{?}{=} \tilde{b}y^d$, where c is defined as $C + d \bmod q$. As a result, $\tilde{Y}$ is accepted as X by Y.

A simple way to enforce proper key-generation, is by having a *trusted registration authority*. This authority need only be active during registration of the public keys, and participants basically have to proof knowledge of their secret key before the public key can be registered. Some care must be taken however, because a man-in-the-middle attacker may also try to abuse an interactive key-generation protocol for the purpose of later misrepresenting himself. One possible solution is the following. Let X be a participant who wishes tohave a public key registered. Then the authority computes $g_* \leftarrow g^{w'}$, where w' is chosen at random from $\mathbb{Z}_q$, and sends g_* to X. Next, X chooses w'' at random from $\mathbb{Z}_q$, computes $x \leftarrow g_*^{w''}$ and proves knowledge of w'' with respect to g_*, using a suitable interactive (zero-knowledge) protocol for instance. Finally, the authority registers x as X's public key and sends w' to X, who computes the secret key as $w \leftarrow w'w'' \bmod q$.

7 An Application

In this section, we give an example where the conditions on key-generation are satisfied in a natural way. Imagine an organization with m sites to which restricted access is applicable. Some n officials are granted access to some of these sites. When an accessor presents himself at one of these sites, his access rights are checked by verifying his identity. These sites may vary from buildings, specific sections of buildings, or even databases or computer systems. The organization keeps a central list of the identities of the officials and their specific access rights. It is assumed that each site has access to this list, either by having a copy of the list at hand, or by consulting the central database.

Let $X_1, \ldots, X_n$ be the collection of participants. The collection of sites with restricted access is denoted $Y_1, \ldots, Y_m$. The organization generates a keyset (x_i, w_i) for each participant X_i, as described in the Key Generation protocol in Section 3. Each participant X_i is given a tamperresistant smartcard S_i, capable of performing our protocols. The keyset is securely loaded into the cards. Now, for each site Y_j, the organization generates a keyset (y_j, v_j). The secret key v_j is destroyed. We assume that each site is represented too by some device capable of performing the protocols. For each site, the organization prepares a list of the public keys of the officials that are granted access to this site. This list is made available to the site. Please note that the devices for the sites need not store any secret information. One only has to make sure that the data they store is authentic and cannot be modified by unauthorized parties.

When participant X_i wishes to exercise his right of access to site Y_j, he lets his smartcard simply perform the identification protocol with site Y_j as the verifier, on common input (x_i, y_j). By the security properties of the identification scheme, the resulting protocol is secure against adaptive impersonation attacks, but furthermore, no adversary can by means of a man-in-the-middle attack, divert the communication to a different site Y_t, and pass there as X_i, even if X_i has the right of access at site Y_t.

References

1. M. Abadi, E. Allender, A. Broder, J. Feigenbaum and L. Hemachandra: *On Generating Solved Instances of Computational Problems*, Proceedings of Crypto '88, Springer Verlag LNCS, vol. 403, pp. 297–310.

2. L. Babai and S. Moran: *Arthur– Merlin Games: A Randomized Proof System and a Hierarchy of Complexity Classes*, JCSS, vol. 36, pp. 254–276, 1988.
3. M. Bellare and O. Goldreich: *On Defining Proofs of Knowledge*, Proceedings of Crypto '92, Springer Verlag LNCS, vol. 740, pp. 390–420.
4. S. Bengio, G. Brassard, Y. Desmedt, C. Goutier and J.J. Quisquater: *Secure Implementation of Identification Systems*, Journal of Cryptology, 1991 (4): 175–183.
5. D. Chaum: *Provers Can Limit the Number of Verifiers, unpublished.*
6. R. Cramer and I. Damgård: *Secure Signature Schemes based on Interactive Protocols*, Proceedings of Crypto '95, Springer Verlag LNCS, vol. 963, pp. 297–310.
7. R. Cramer, I. Damgård and B. Schoenmakers: *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols*, Proceedings of Crypto '94, Springer verlag LNCS, vol. 839, pp. 174–187.
8. D. Dolev, C. Dwork and M. Naor: *Non-malleable cryptography*, Proceedings of STOC '91, pp. 542–552.
9. A. Fiat and A. Shamir: *How to Prove Yourself: Practical Solutions to Identification and Signature Problems*, Proceedings of Crypto '86, Springer Verlag LNCS, vol. 263, pp. 186–194
10. U. Feige, A. Shamir: *Witness Indistinguishable and Witness Hiding Protocols*, Proceedings of STOC '90, pp. 416–426.
11. U. Feige, A. Fiat and A. Shamir: *Zero-Knowledge Proofs of Identity*, Journal of Cryptology 1 (1988) 77–94.
12. U. Feige and A. Shamir: *Zero-Knowledge Proofs of Knowledge in Two Rounds*, Proceedings of Crypto '89, Springer Verlag LNCS, vol. 435, pp. 526–544.
13. S. Goldwasser, S. Micali and C. Rackoff: *The Knowledge Complexity of Interactive Proof Systems*, SIAM J.Computing, Vol. 18, pp. 186-208, 1989.
14. *Efficient Identification Schemes Secure against Impersonation and Man-in-the-Middle Attacks*, preprint, October 1995.
15. L. Guillou, J.J. Quisquater: *A Practical Zero-Knowledge Protocol fitted to Security Microprocessor Minimizing both Transmission and Memory*, Proceedings of Eurocrypt '88, Springer Verlag LNCS, vol. 330, pp. 123–128.
16. M. Jacobson, R. Impagliazzo and K. Sako: *Designated Verifier Proofs and their Applications*, Proc. of Eurocrypt '96, Springer Verlag LNCS, vol. 1070, pp. 143–154.
17. T. Okamoto: *Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes*, Proceedings of Crypto '92, Springer Verlag LNCS, vol. 740, pp. 31–53.
18. C. P. Schnorr: *Efficient Signature Generation by Smart Cards*, Journal of Cryptology, 4 (3): 161–174, 1991.