On the length of cryptographic hash-values used in identification schemes

Marc Girault

SEPT
42 rue des Coutures, BP 6243
14066 Caen, France.
e-mail : marc.girault@sept.fr

Jacques Stern

Laboratoire d'Informatique
Ecole Normale Supérieure
45 rue d'Ulm, 75230 Paris, France.
e-mail : jacques.stern@ens.fr

Abstract. Many interactive identification schemes based on the zero-knowledge concept use cryptographic hash-values, either in their basic design or in specific variants. In this paper, we first show that 64-bit hash-values, a length often suggested, definitely decrease the level of the security of all these schemes. (Of course, this does not compromise the security of the schemes by themselves). Then we prove that collision-resistance is a sufficient condition to achieve the claimed level of security. Finally, by using a weaker notion of collision-resistance, we present interesting variants of some of these schemes (in particular the Schnorr and the Guillou-Quisquater schemes) which minimize the number of communication bits for a given level of security.

1 Introduction

In recent years, several interactive identification schemes have been proposed based on the zero-knowledge concept [GMR85]. In all these schemes, the prover starts by committing himself to some secret values he picks at random. To compute this commitment, hash-functions are often used, either in the basic design of the scheme or in specific variants.

The first scheme of this type was the one by Fiat and Shamir, presented at CRYPTO'86 conference [FS86]. This scheme is based on the modular square root extraction problem. The basic protocol, to be repeated several times, has three passes. In a variant whose the goal is to minimize the communication bits, the prover sends in the first pass a cryptographic hash-value of some elements selected by him. The length suggested by the authors for this hash-value was 128 bits.

At CRYPTO'89 conference, Schnorr presented an identification scheme based on the discrete logarithm problem [Sc89]. The protocol of this scheme has three passes. In a variant whose the goal is to minimize the communication bits, the prover sends in the first pass a cryptographic hash-value of some elements selected by him. The suggested length for this hash-value was at least k bits, where $1-2^{-k}$ is the level of security (impostor detection probability) to be achieved.

At the rump session of CRYPTO'89 conference, Shamir presented an identification scheme based on an NP-complete problem, the so-called permuted kernel problem [Sh89]. The basic protocol of this scheme, to be repeated several times, has five passes and, in the first pass, the prover sends two cryptographic hash-values of some elements selected by him. Hash-values of 64 bits were specifically mentioned in the paper.

Y.G. Desmedt (Ed.): Advances in Cryptology - CRYPTO '94, LNCS 839, pp. 202-215, 1994.

Finally, at CRYPTO'93 conference, Stern presented an identification scheme based on an NP-complete problem, the so-called syndrome decoding problem [St93] (a first tentative had already been presented at EUROCRYPT'89 [St89]). The basic protocol of this scheme, to be repeated several times, has three passes and, in the first pass, the prover sends three cryptographic hash-values of some elements selected by him. Hash-values of 64 bits were specifically mentioned in the paper.

The goal of this paper is to discuss the appropriate length of these hash-values. First, we show that, if 2^{32} operations are deemed to be computationally feasible (we will make this assumption all along the paper), then 64-bit hash-values definitely decrease the level of security of all these schemes. This is shown by exhibiting, for each these schemes, specific attacks based on the birthday paradox. Of course, these attacks do not compromise the security of the schemes by themselves, but only suggest to use longer hash-values.

Second, we formally prove that collision-resistance is a sufficient condition to achieve the level of security which is claimed by the authors. As a consequence, a length of 128 bits (if 2^{64} operations are deemed to be computationally infeasible) or more for the hash-values seems to be convenient.

Third, by using a weaker notion of collision-resistance (the so-called r-collision resistance), we present interesting variants of some of these schemes (in particular the Schnorr and the Guillou-Quisquater scheme) which minimize the number of communication bits for a given level of security.

2 The Fiat-Shamir scheme

2.1 Description

The identification scheme presented by Fiat and Shamir at CRYPTO'86 conference [FS86] is based on the difficulty of extracting square roots modulo a composite number whose factors are unknown. A trusted center is used to compute the users' secret keys. The universal parameters, i.e. those shared by all the users, are :

- a large composite modulus n (whose factors are only known to the center)
- two small integers k and t
- a pseudo-random function f.

The recommended size for n was (at least) 512 bits in 1986 (today, a larger size would probably be recommended). Values for k and t are closely related to the level of security of the protocol (see further). Typical values are 6 and 5 (or 9 and 8 for the related signature scheme).

The prover's public key is his "identity" (i.e. a string I which contains relevant informations about him and/or his device). His secret key is composed of k values s_j computed as follows : Let $v_j = f(I,j)$ for k small values of j such that v_j is a quadratic residue modulo n (for convenience, we assume that $j = 1...k$). Then $s_j^2 v_j = 1 \pmod{n}$ for each value of j.

The basic (3-pass) protocol is the following.

1. The prover randomly selects an integer r in $\{0...n\}$, computes $x = r^2 \pmod n$ and sends x to the verifier.
2. The verifier randomly selects an element $e = (e_1,e_2,...,e_k)$ of $\{0,1\}^k$ and sends e to the prover.
3. The prover computes $y = r\prod_{e_j=1} s_j \pmod n$ and sends y to the verifier.
4. The verifier computes all the v_j and checks : $x = y^2 \prod_{e_j=1} v_j \pmod n$.

Note that an impostor (who ignores s) can easily deceive the verifier with probability 2^{-k}, by selecting an integer y, "guessing" an element e and computing x as in step 4. As he is provably unable, if factoring is difficult, to deceive the verifier with probability essentially greater than 2^{-k}, then it suffices to repeat t times the basic protocol to obtain a level of security (i.e. the impostor detection probability) greater than or equal to $1-2^{-kt}$.

In order to decrease the number of communication bits, Fiat and Shamir have suggested to send to the verifier at step 1 the first 128 bits of $f(x)$ instead of x. Let us call c the result and h the function which maps x to c. The check equation of step 4 then becomes :

$$c = h(y^2 \prod_{e_j=1} v_j \pmod n)$$

We will call the new scheme the h-variant of the Fiat-Shamir scheme. In the following section, we show that some bad choices for h reduce the level of the security of the scheme. To be clear, these "bad" choices are *not* mentioned in the paper by Fiat and Shamir.

2.2 Too short (or ill-chosen) hash-values decrease the level of security

Case : c is 64-bit long. If c is too short (say 64 bits), then there is an easy (at least to design) attack using the birthday paradox : first, the impostor selects a set E of 2^{32} integers y and two distincts elements $e^{(1)}$ and $e^{(2)}$ of $\{0,1\}^k$. For each element y of E, he computes :

$$h(y^2 \prod_{e_j^{(1)}=1} v_j \pmod n) \quad \text{and} \quad h(y^2 \prod_{e_j^{(2)}=1} v_j \pmod n).$$

Due to the birthday paradox, there exist with high probability two integers y_1 and y_2 such that :

$$h(y_1^2 \prod_{e_j^{(1)}=1} v_j \pmod n) = h(y_2^2 \prod_{e_j^{(2)}=1} v_j \pmod n)$$

Let us call c this common value (the "collision"). If no collision occurred, the impostor has to increase slightly the size of E.

Now, at each execution of the protocol, the impostor does the following. He sends c to the verifier, who sends back e to him. Then, if $e = e^{(1)}$, he replies with y_1. If $e = e^{(2)}$, he replies with y_2. So, in two cases, the verifier will be satisfied with the reply and the impostor acceptance probability is 2^{-k+1} instead of 2^{-k}. In particular, if $k = 1$, then the impostor is always accepted.

Case : c is a truncation of x. If h is only a truncation of x, for instance if c is composed of the 128 rightmost bits of x, then another type of attack allows the impostor to achieve the same probability of acceptance as above. The impostor selects an integer z less than n whose 128 rightmost bits are zero, and two distinct elements $e^{(1)}$ and $e^{(2)}$ of $\{0..1\}^k$. Let m be equal to $z \prod_{e_j^{(1)}=1} v_j^{-1} \pmod n$ and k be equal to $-\prod_{e_j^{(2)}=1} v_j \prod_{e_j^{(1)}=1} v_j^{-1} \pmod n$.

By using the Pollard-Schnorr attack of the Ong-Schnorr-Shamir signature scheme [PS87], it is possible to find in polynomial time two integers y_1 and y_2 such that :

$$y_1^2 + k y_2^2 = m \pmod n$$

which implies, by multiplying with $\prod_{e_j^{(1)}=1} v_j \pmod n$:

$$y_1^2 \prod_{e_j^{(1)}=1} v_j - y_2^2 \prod_{e_j^{(2)}=1} v_j = z \pmod n$$

With probability about 1/2, $y_1^2 \prod_{e_j^{(1)}=1} v_j \pmod n$ is greater than $y_2^2 \prod_{e_j^{(2)}=1} v_j \pmod n$. In such a case, the above equation implies that the 128 rightmost bits of $y_1^2 \prod_{e_j^{(1)}=1} v_j \pmod n$ are equal to the 128 rightmost bits of $y_2^2 \prod_{e_j^{(2)}=1} v_j \pmod n$. (In the other case, try with another value of z). Let us call c this common value. Then the rest is as above. Note that this attack can be adapted to the case where c is composed of the 128 leftmost bits (or even any 128 consecutive bits) of x.

2.3 The level of security of the h-variant

We now show that the h-variant does achieve the security level which is claimed, if h is collision-free. In fact, we prove a more general theorem, for which we need the following definitions :

Definition 1. A r-collision for a function h is a r-tuple $(x_1, x_2, ..., x_r)$ of r pairwise distinct values such that $h(x_1) = h(x_2) = ... = h(x_r)$.

Definition 2. (*informal*) A function h is r-collision-free (or r-collision-resistant) if it is computationally infeasible to find a r-collision for h.

We now establish a precise connection between r-collision resistance and the level of security of the h-variant.

Theorem 1. *If there exists a PPTM (Probabilistic Polynomial Turing Machine) M such that the probability that M be accepted by an honest verifier is greater than* $(r-1)2^{-k}+\varepsilon$, *with* $\varepsilon>0$, *then there exists a PPTM* $\tilde{M}$ *which with overwhelming probability either computes the square root of one product of the form :*

$$\prod_{j=1}^{k} v_j^{c_j} \pmod n$$

where c_j = *-1, 0 or +1 (not all of them zero) or finds a r-collision for h.*

Remark: As observed in [FFS88], the first conclusion contradicts the intractability of factoring assumption, as a coalition of the legitimate user and of the potential attacker could factor n. The second conclusion implies that h is not r-collision-free.

Proof. Let Ω be the set of m elements in which M picks its random values and E be the set $\{0,1\}^k$, both of them with the uniform distribution. For each value (ω,e) of $\Omega\times E$, M passes the protocol (we say it is a success) or not. Let S be the subset of $\Omega\times E$ composed of all the successes. Our assumption is that :

$$\frac{\mathrm{Card}(S)}{\mathrm{Card}(\Omega\times E)} > (r-1)2^{-k}+\varepsilon$$

with $\varepsilon>0$ and $\mathrm{Card}(\Omega\times E)=m.2^k$.

Let Ω_r be the section $\{\omega\in\Omega : \mathrm{Card}\{e\in E : (\omega,e) \text{ is a success}\}\geq r\}$. We have :

$$\mathrm{Card}(S)\leq \mathrm{Card}(\Omega_r).2^k+(r-1)(m-\mathrm{Card}(\Omega_r))$$

Then :

$$\frac{\mathrm{Card}(S)}{\mathrm{Card}(\Omega\times E)}\leq\left[\frac{\mathrm{Card}(\Omega_r)}{\mathrm{Card}(\Omega)}+(r-1)(2^{-k}-\frac{\mathrm{Card}(\Omega_r)}{\mathrm{Card}(\Omega\times E)})\right]\leq\left[\frac{\mathrm{Card}(\Omega_r)}{\mathrm{Card}(\Omega)}+(r-1)2^{-k}\right]$$

which implies :

$$\frac{\mathrm{Card}(\Omega_r)}{\mathrm{Card}(\Omega)}\geq\varepsilon.$$

Let $\tilde{M}$ be the PPTM obtained by resetting M ε^{-1} times. With constant probability, $\tilde{M}$ picks ω in Ω_r and the probability can be made close to 1 by repeating the execution of $\tilde{M}$. At the end, r values $y_1, y_2..., y_r$ are found such that, for distinct $e^{(1)}, e^{(2)},..., e^{(r)}$:

$$h(y_1^2\prod_{e_j^{(1)}=1}v_j(\mathrm{mod}\,n))=h(y_2^2\prod_{e_j^{(2)}=1}v_j(\mathrm{mod}\,n))=.....=h(y_r^2\prod_{e_j^{(r)}=1}v_j(\mathrm{mod}\,n))$$

Now, there are two possibilities :

a) either two of the values (say $y_1^2 \prod_{e_j^{(1)}=1} v_j \pmod n$ and $y_2^2 \prod_{e_j^{(2)}=1} v_j \pmod n$) are equal before hashing. In that case, y_1/y_2 is a square root of a product of the form $\prod_{j=1}^{k} v_j^{c_j} \pmod n$, where $c_j = -1,\ 0$ or $+1$ (not all of them zero) ;

b) or all these values are pairwise distinct and a r-collision for h has been found. □

This result suggests to use hash-functions which are only resistant to r-collisions (with $r > 2$), so that the hash-values computed in the first pass can be made much shorter. Indeed the decrease of the level of security can be balanced by sending a slightly larger value of e in the second pass (more precisely, if $r = 2^u$, $e \in \{0,1\}^{k+u}$ instead of $e \in \{0,1\}^k$). But this is not so interesting in the Fiat-Shamir scheme, as it would also imply a larger number of secrets s_j, a very undesirable feature. On the contrary, this idea is particularly attractive in the Schnorr scheme, as shown in the following section.

3 The Schnorr scheme

3.1 Description

The identification scheme presented by Schnorr at CRYPTO'89 conference [Sc89] is based on the difficulty of computing a discrete logarithm. The universal parameters are :

- a large prime p
- a prime q such that $q \mid p-1$
- an integer α (the "base") such that $\alpha^q = 1 \pmod p$
- a small integer k.

The recommended sizes for n and q were respectively (at least) 512 bits and 140 bits in 1989. The value of k is closely related to the level of security of the protocol (see further). A typical value is 40 (or 72 for the related signature scheme).

The prover's secret key is an integer s in $\{1...q\}$. His public key is $v = \alpha^{-s} \pmod p$. The basic (3-pass) protocol is the following.

1.	The prover randomly selects an integer r in $\{1...q\}$, computes $x = \alpha^r \pmod p$ and sends x to the verifier.
2.	The verifier randomly selects an element e of $\{0...2^k - 1\}$ and sends e to the prover.
3.	The prover computes $y = r + se \pmod q$ and sends y to the verifier.
4.	The verifier checks : $x = \alpha^y v^e \pmod p$.

Note that an impostor (who ignores s) can easily deceive the verifier with probability 2^{-k}, by selecting an integer y, "guessing" an element e and computing x as in step 4. As he is provably unable, if computing discrete logarithms is difficult, to deceive the verifier with probability essentially greater than 2^{-k}, then the level of security is equal to $1-2^{-k}$.

In order to decrease the number of communication bits, Schnorr has suggested to send to the verifier at step 1 $c=h(x)$ where h is a k-bit hash-function. The check equation of step 4 then becomes :

$$c=h(\alpha^{y}v^{e}(\mathrm{mod}\ p))$$

We will call the new scheme the h-variant of the Schnorr scheme. In the following section, we show that some bad choices for h reduce the level of security of the scheme.

3.2 Too short hash-values decrease the level of security

The first observation is the same as in the Fiat-Shamir scheme : if c is too short, then the security level may be lower than expected. For example, if c is only 64-bit long, then a birthday attack quite similar to the one described in subsection 2.2 can be designed (no matter how the function h is defined). As a consequence, c should be at least 128-bit long if we want the security level be equal to 2^{-k}.

The second observation differs from Fiat-Shamir case : as far as we are aware, the level of security does not decrease if c is only a truncation of x, provided the number of bits is large enough (say 128 bits, because of the first observation). This shows that, in this scheme, one-wayness (and a fortiori collision-resistance) does not seem to be a necessary condition for h, in order to achieve a security level be equal to 2^{-k}. Nevertheless, collision-resistance remains a sufficient condition to achieve this security level, as shown now.

3.3 The level of security of the h-variant

We can state a similar theorem to the one of section 2 :

Theorem 2. *If there exists a PPTM M such that the probability that M be accepted by an honest verifier is greater than $(r-1)2^{-k}+\varepsilon$, with $\varepsilon>0$, then there exists a PPTM $\tilde{M}$ which with overwhelming probability either computes the discrete logarithm of v (modulo q in base α) or finds a r-collision for h.*

Proof. The proof is quite similar to the theorem of section 2. At the end, r values y_1, y_2,..., y_r are found such that, for pairwise distinct e_1, e_2, ..., e_r:

$$h(\alpha^{y_1}v^{e_1}(\mathrm{mod}\ p))=h(\alpha^{y_2}v^{e_2}(\mathrm{mod}\ p))=\ldots\ldots\ldots=h(\alpha^{y_r}v^{e_r}(\mathrm{mod}\ p))$$

Now, there are two possibilities :

a) either two of the values are equal before hashing, say : $\alpha^{y_1} v^{e_1} (\mathrm{mod}\, p)$ and $\alpha^{y_2} v^{e_2} (\mathrm{mod}\, p)$. In that case, $\frac{y_1 - y_2}{e_2 - e_1} (\mathrm{mod}\, q)$ is the discrete logarithm of v modulo q in base α.

b) or all these values are pairwise distinct and a r-collision for h has been found. □

3.4 An interesting optimization

The preceding theorem leads to an interesting optimization of the Schnorr scheme. The idea is to use r-collision hash-functions with $r > 2$, so that the hash-values computed in the first pass can be made much shorter. The decrease of the level of security is compensated by sending a slightly larger value of e in the second pass. Contrary to the Fiat-Shamir scheme, this does not have any undesirable consequence.

In order to make a precise statement, we need a result related to the birthday paradox :

Lemma. *Let E be a set of cardinality n, F_m a sample of size m drawn from E with replacements and r an integer. Let us call r-coincidence an element of E to which exactly r elements of F_m are equal to. For n a sufficiently large integer, $m = (\lambda r!)^{1/r} n^{r-1/r}$, with $\lambda \leq 1$ and $m/n \leq 1/128$, the probability that there is no s-coincidence for $s \geq r$ is very close to $e^{-\lambda}$ (hence greater than $1/e$).*

Proof (sketch). By a classical result from von Mises (see e.g. [Fe68] page 106), the probability $p(i,r)$ that there are exactly i r-coincidences is :

$$p(i,r) \approx e^{-\Lambda} \frac{\Lambda^i}{i!}$$

with :

$$\Lambda = n \frac{e^{-m/n}}{r!} \left(\frac{m}{n}\right)^r$$

If $m = (\lambda r!)^{1/r} n^{r-1/r}$ with $m/n \leq 1/128$, then $\Lambda = \lambda e^{-m/n} \approx \lambda$. Hence the probability $p(0,r)$ that there is no r-coincidence at all is very close to $e^{-\lambda}$, and so is the probability that there is no s-coincidence for $s \geq r$ if $\lambda \leq 1$. (Intuitively, the reason why is the following : if the probability that there is a s-coincidence for $s \geq r+1$ were not nearly equal to zero, then the probability $1 - p(0,r)$ that there is a r-coincidence would be nearly equal to 1 ; this is not the case since $1 - p(0,r) \approx 1 - e^{-\lambda} \leq 1 - 1/e \approx 0.632$). □

This result allows to specify a version of the Schnorr scheme which minimizes the number of communication bits. Let m be an integer greater than or equal to the number of operations deemed to be computationally infeasible (typically $m = 2^{64}$), and h be a pseudo-random hash-function whose the values range in the interval $\{0...n\text{-}1\}$ with $m = (\lambda r!)^{1/r} n^{r-1/r}$, $m/n \leq 1/128$ and $\lambda \leq 1$. Then, using a traditional argument, h can be

considered as r-collision-free, as the probability that a r-collision is found with a number of computations supposed to be infeasible is substantially less than 1 (to be precise, less than 0.632 and even less than $1-e^{-\lambda}$). For instance, if $r = 2$, the usual choice : $m = n^{1/2}$ gives $\lambda = 1/2$. If $n \geq 2^{14}$, the assumptions of the lemma are satisfied, and the function can be considered as (2-) collision-resistant.

If r > 2, according to the theorem of subsection 3.2.1, the level of security has decreased to $1-(r-1)2^{-k}$. But we can compensate this decrease by choosing e in the range $\left\{0...2^{k+\log_2(r-1)}\right\}$. For that reason, it may be convenient that r-1 is a power of 2.

As a consequence, some practical values for r are 5 and 9. Fixing m to 2^{64} and λ to 1/2 as above, we have $r=5 \Rightarrow n \geq 2^{80}/2.27$ and $r=9 \Rightarrow n \geq 2^{72}/3.84 \Rightarrow n \geq 2^{71}$ (since we must have $m/n \leq 1/128$).

In the first case, e must be k+2-bit long and the hash-values 79-bit long, if we want to achieve a level of security equal to $1-2^{-k}$.

In the second case, e must be k+3-bit long and the hash-values 71-bit long. We therefore have saved about 57 bits in the first pass (compared to a 128-bit hash-value computed from a (2-)collision-resistant hash-function) and, in the second pass, we have only three bits more to transport. Globally, we have saved 54 bits, for a total of 71 + (k + 3) + 140 = 214 + k, instead of 268 + k. If k = 40, then we have a total of 254 instead of 308, i.e. the number of communication bits has decreased by about 18%.

The same optimization applies to the Guillou-Quisquater scheme [GQ88]. In that case, the number of communication bits can be typically decreased from 680 to 626 bits, i.e. by about 8%.

4 The Stern identification scheme

4.1 Description

The identification scheme presented by Stern at CRYPTO'93 conference [St93] is based on the difficulty of the syndrome decoding problem, that is the (NP-complete) problem of finding a word of given syndrome and of given weight. The universal parameters are :

- a random binary $(k \times n)$ matrix A $(k < n)$
- an integer p
- a hash-function h.

Typical values for (k,n,p) are (256,512,56) or, still better, (512,1024,110). The prover's secret key is a random n-bit word s of weight (i.e. the number of '1' bits of s, denoted by $|s|$) equal to p. His public key is $v = A(s)$. The basic (3-pass) protocol is as follows.

1. The prover randomly selects an n-bit word y and a permutation σ of the integers $\{1...n\}$, computes $c_0 = h(\sigma, A(y))$, $c_1 = h(y.\sigma)$, $c_2 = h(y'.\sigma)$, with $y' = y \oplus s$ (where $\oplus$ stands for bitwise addition modulo 2 and $y.\sigma$ refers to the image of y under permutation σ), and sends c_0, c_1 and c_2 to the verifier.

2. The verifier randomly selects an element b of $\{0,1,2\}$ and sends b to the prover.

3. The prover sends to the verifier :	4. The verifier checks :
- if $b = 0$: (y, σ)	$h(\sigma, A(y)) = c_0$ and $h(y.\sigma) = c_1$
- if $b = 1$: (y', σ)	$h(\sigma, A(y') \oplus v) = c_0$ and $h(y'.\sigma) = c_2$
- if $b = 2$: $(y.\sigma, y'.\sigma)$	$h(y.\sigma) = c_1$, $h(y'.\sigma) = c_2$ and $\lvert y.\sigma \oplus y'.\sigma \rvert = p$

Note that an impostor (who ignores s) can deceive the verifier with probability 2/3, by using one of the three following strategies :

a) he selects a permutation σ and two words y and y' such that $A(y') = A(y) \oplus v$, and computes c_0, c_1 and c_2 as above. Then he is able to answer correctly to $b = 0$ and $b = 1$, but not to $b = 2$.

b) he selects a permutation σ and two words y and y' such that $\lvert y \oplus y' \rvert = p$, and computes c_0, c_1 and c_2 as above. Then he is able to answer correctly to $b = 0$ and $b = 2$, but not to $b = 1$.

c) he selects a permutation σ and two words y and y' such that $\lvert y \oplus y' \rvert = p$, computes $c_0 = h(\sigma, A(y') \oplus v)$ and computes c_1 and c_2 as above. Then he is able to answer correctly to $b = 1$ and $b = 2$, but not to $b = 0$.

Provided an impostor cannot deceive the verifier with probability greater than 2/3, then it suffices to repeat t times the basic protocol in order to obtain an impostor detection probability greater than or equal to $1-(2/3)^t$.

4.2 Too short hash-values decrease the level of security

We now show that an impostor can deceive the verifier with probability 1 if hash-values c_0, c_1 and c_2 are 64-bit long. More precisely, we show that, if only one of these hash-values is 64-bit long, then the impostor has a strategy, based on the birthday paradox, which allows him to deceive the verifier with probability equal to 1.

Case : c_0 is 64-bit long. The attack is as follows. First, the impostor selects two words z and z' such that $\lvert z \oplus z' \rvert = p$. Then he prepares a set Σ of 2^{32} permutations σ of $\{1...n\}$. For each permutation σ of this set, he computes $h(\sigma, A(z.\sigma^{-1}))$ and $h(\sigma, A(z'.\sigma^{-1}) \oplus v)$. Due to the birthday paradox, there exist with high probability two permutations σ and σ' in Σ such that $h(\sigma, A(z.\sigma^{-1})) = h(\sigma', A(z'.\sigma'^{-1}) \oplus v)$. Let us call c_0 this common value

(the "collision"), and set $y = z.\sigma^{-1}$ and $y' = z'.\sigma'^{-1}$. If no collision occurred, the impostor has to increase slightly the size of Σ.

Now, at each execution of the basic protocol, the impostor does the following. He sends c_0, $c_1 = h(z)$ and $c_2 = h(z')$ to the verifier. If the verifier sends $b = 0$, he replies with (y,σ). If $b = 1$, he replies with (y',σ'). If $b = 2$, he replies with (z,z'). In each case, the verifier will be satisfied with the reply.

Case : c_1 is 64-bit long. The attack is as follows. The impostor selects a permutation σ and a word y'. Then he prepares a set E_1 of 2^{32} words y_1 of syndrome equal to $A(y') \oplus v$, i.e. words y_1 such that $A(y_1) = A(y') \oplus v$. (Note that there are about 2^{n-k} such words, hence more than 2^{256}, and that they can be computed in a straightforward manner). He also prepares a set E_2 of 2^{32} words y'_2 such that $|y' \oplus y_2| = p$. With high probability, there exist y_1 in E_1 and y_2 in E_2 such that $h(y_1.\sigma) = h(y_2.\sigma)$. Let us call c_1 this common value.

Now, at each execution of the basic protocol, the impostor does the following. He sends $c_0 = h(\sigma, A(y_1))$, c_1 and $c_2 = h(y'.\sigma)$ to the verifier. If the verifier sends $b = 0$, he replies with (y_1,σ). If $b = 1$, he replies with (y',σ). If $b = 2$, he replies with $(y_2.\sigma, y'.\sigma)$. In each case, the verifier will be satisfied with the reply.

Case : c_2 is 64-bit long. The attack is essentially the same as the previous one. First, the impostor selects a permutation σ and a word y. Then he prepares a set E_1 of 2^{32} words y'_1 of syndrome equal to $A(y) \oplus v$ and a set E_2 of 2^{32} words y'_2 such that $|y \oplus y'_2| = p$. With high probability there exist a word y'_1 in E_1 and a word y'_2 in E_2 such that $h(y'_1.\sigma) = h(y'_2.\sigma)$. Let us call c_2 this common value.

Now, at each execution of the basic protocol, the impostor does the following. He sends $c_0 = h(\sigma, A(y))$, $c_1 = h(y.\sigma)$ and c_2 to the verifier. Then, if $b = 0$, he replies with(y,σ). If $b = 1$, he replies with (y'_1,σ). If $b = 2$, he replies with $(y.\sigma, y'_2.\sigma)$. In each case, the verifier will be satisfied with the reply.

4.3 The level of security

The level of security results from the following theorem, implicitly contained in [St93] :

Theorem 3. *If there exists a PPTM M such that the probability that M be accepted by an honest verifier is greater than* $2/3 + \varepsilon$, *with* $\varepsilon > 0$, *then there exists a PPTM* $\tilde{M}$ *which, with overwhelming probability, either computes a word of weight p and of syndrome v or finds a collision for h.*

5 The Shamir identification scheme

5.1 Description

The identification scheme presented by Shamir at CRYPTO'89 conference [Sh89] is based on the permuted kernel problem, that is the (NP-complete) problem of finding a permutation which puts a given vector into the kernel of a given matrix. The universal parameters are :

- a (small) prime p
- a random p-ary $(k \times n)$ matrix A $(k < n)$
- a hash-function h.

Typical values for (p,k,n) are (251,16,32) or, still better, (251,37,64). All calculations are done modulo p. The prover's secret key is a random permutation π of the integers $\{1...n\}$. His public key is an n-vector v such that $v.\pi \in \mathrm{Ker}A$ (where, as above, $v.\pi$ refers to the image of v under permutation π). The basic (5-pass) protocol is as follows.

1.	The prover randomly selects an n-vector y and a permutation σ of the integers $\{1...n\}$, computes $c_0 = h(\sigma, A(y))$ and $c_1 = h(\sigma', y.\sigma)$, with $\sigma' = \pi\sigma$, and sends c_0 and c_1 to the verifier.	
2.	The verifier randomly selects an integer d in $\{0...p-1\}$ and sends d to the prover.	
3.	The prover computes $w = y.\sigma + dv.\sigma'$ and sends w to the verifier.	
4.	The verifier randomly selects a bit b and sends b to the prover.	
5.	The prover sends to the verifier :	6. The verifier checks :
	- if $b = 0$: σ	$h(\sigma, A(w.\sigma^{-1})) = c_0$
	- if $b = 1$: σ'	$h(\sigma', w - dv.\sigma') = c_1$

Note that an impostor (who ignores π) can deceive the verifier with probability $(p+1)/2p$, by using one of the two following strategies :

a) he selects an n-vector y, two permutations σ and σ' and an integer d_1 in $\{0...p-1\}$, then computes c_0 as above and $c_1 = h(\sigma', y.\sigma - d_1 v\sigma')$. At step 3, he sends $w = y.\sigma$ to the verifier, whatever d is. Then he is able to answer correctly either to $b = 0$ and any d, or to $b = 1$ and $d = d_1$, but not to $b = 1$ and $d \neq d_1$.

b) he selects an n-vector y, two permutations σ and σ' and an integer d_1 in $\{0...p-1\}$, then computes $c_0 = h(\sigma, A(y + d_1 v.\sigma'\sigma^{-1}))$. At step 3, he sends $w = y.\sigma + dv.\sigma'$ to the verifier. Then he is able to answer correctly either to $b = 1$ and any d, or to $b = 0$ and $d = d_1$, but not to $b = 0$ and $d \neq d_1$.

Provided an impostor cannot deceive the verifier with probability greater than $(p+1)/2p$, then it suffices to repeat t times the basic protocol in order to obtain an impostor detection probability greater than or equal to $1-(p+1/2p)^t$ ($\approx 1-2^{-t}$ if p is not too small).

5.2 Too short hash-values decrease the level of security

We now show that an impostor can deceive the verifier with probability $(p+2)/2p$ if hash-values c_0 and c_1 are 64-bit long. More precisely, we show that, if only one of these hash-values is 64-bit long, then the impostor has a strategy, based on the birthday paradox, which allows him to deceive the verifier with probability equal to $(p+2)/2p$. This is of particular significance if the value sent by the verifier in the second pass is restricted to 0 or 1 (a possibility mentioned in [Sh89]), since the impostor acceptance probability then grows up from 3/4 to 1.

Case : c_0 is 64-bit long. The attack is as follows. First, the impostor selects a permutation σ', an n-vector z and two distinct elements d_1 and d_2. Then he prepares a set Σ of 2^{32} permutations σ of $\{1...n\}$. (Note that there are $n! \geq 32! \geq 2^{117}$ such permutations). For each permutation σ of this set, he computes $h(\sigma, A(z.\sigma^{-1} + d_1 v.\sigma'\sigma^{-1}))$ and $h(\sigma, A(z.\sigma^{-1} + d_2 v.\sigma'\sigma^{-1}))$. Due to the birthday paradox, there exist with high probability two permutations σ_1 and σ_2 in Σ such that $h(\sigma_1, A(z.\sigma_1^{-1} + d_1 v.\sigma'\sigma_1^{-1})) = h(\sigma_2, A(z.\sigma_2^{-1} + d_2 v.\sigma'\sigma_2^{-1}))$. Let us call c_0 this common value.

Now, at each execution of the basic protocol, the impostor does the following : he sends c_0 and $c_1 = h(\sigma', z)$ to the verifier, who sends back d to him. He sends $w = z + dv.\sigma'$ to the verifier, who sends back b to him. Then, if $b = 1$, he replies with σ'. If $b = 0$ and $d = d_1$ (resp. $d = d_2$), he replies with σ_1 (resp. σ_2). In other cases, he sends anything. So, in $p+2$ cases, the verifier will be satisfied with the reply.

Case : c_1 is 64-bit long. First, the impostor selects a permutation σ, an n-vector y and two distinct elements d_1 and d_2. Then he prepares a set Σ' of 2^{32} permutations σ' of $\{1...n\}$. For each permutation σ' of this set, he computes $h(\sigma', y.\sigma - d_1 v.\sigma')$ and $h(\sigma', y.\sigma - d_2 v.\sigma')$. With high probability, there exist two permutations σ'_1 and σ'_2 in Σ' such that $h(\sigma'_1, y.\sigma - d_1 v.\sigma'_1) = h(\sigma'_2, y.\sigma - d_2 v.\sigma'_2)$ and we call c_1 this common value.

Now, at each execution of the basic protocol, the impostor does the following : he sends $c_0 = h(\sigma, A(y))$ and c_1 to the verifier, who sends back d to him. He sends $w = y.\sigma$ to the verifier, who sends back b to him. Then, if $b = 0$, he replies with σ. If $b = 1$ and $d = d_1$ (resp. $d = d_2$), he replies with σ'_1 (resp. σ'_2). In other cases, he sends anything. So, in $p+2$ cases, the verifier will be satisfied with the reply.

5.3 The level of security

The level of security results from the following theorem, implicitly contained in [Sh89] :

Theorem 4. *If there exists a PPTM M such that the probability that M be accepted by an honest verifier is greater than $(p+1)/2p + \varepsilon$, with $\varepsilon > 0$, then there exists a PPTM $\tilde{M}$ which with overwhelming probability either computes a permutation π such that $v.\pi \in \mathrm{Ker}A$ or finds a collision for h.*

6 Conclusion

First, we have considered several identification schemes using hash-values in the first pass (either in their basic design or in specific variants) and given evidence that 64-bit hash-values were too short to achieve the level of the security claimed by their authors, by exhibiting for each of them one or more specific attacks. Second we have proved that collision-resistance was a sufficient condition to achieve this level. Third we have shown that the number of communication bits could be minimized in schemes based on modular arithmetic by using r-collision resistant hash-functions instead of collision-resistant hash-functions. As an example, the number of bits transported in the first pass of the Schnorr scheme can be decreased from 128 to 71, and the total number of bits transported from 308 to 254, i.e. by 18%, for a level of security equal to $1-2^{-40}$.

References

[Fe68] W. Feller, An introduction to probability theory and its applications, J. Wiley & Sons, 1968.

[FFS88] U. Feige, A. Fiat and A. Shamir, Zero-knowledge proofs of identity, *Journal of Cryptology*, Vol. 1, N°2, 1988, pp. 77-94.

[FS86] A. Fiat and A. Shamir, How to prove yourself : Practical solutions to identification and signature problems, *Advances of Cryptology : Proceedings of CRYPTO'86*, Lecture Notes in Computer Science, Vol. 263, Springer-Verlag, Berlin, 1987, pp. 186-194.

[GMR85] S. Goldwasser, S. Micali and C. Rackoff, The knowledge of interactive proof-systems, *Proceedings of 17th ACM Symposium on Theory of Computing*, 1985, pp. 291-304.

[GQ88] L.C. Guillou and J.J. Quisquater, A practical zero-knowledge protocol fitted to security microprocessors minimizing both transmission and memory, *Advances of Cryptology : Proceedings of EUROCRYPT'88*, Lecture Notes in Computer Science, Vol. 330, Springer-Verlag, Berlin, 1988, pp. 123-128.

[PS87] J.M. Pollard and C.P. Schnorr, An efficient solution of the congruence $x^2 + ky^2 = m \pmod{n}$, *IEEE Transactions on Information Theory*, Vol. IT-33, N°5, 1987, pp. 702-709.

[Sc89] C.P. Schnorr, Efficient identification and signature for smart cards, *Advances of Cryptology : Proceedings of CRYPTO'89*, Lecture Notes in Computer Science, Vol. 435, Springer-Verlag, Berlin, 1987, pp. 239-252.

[Sh89] A. Shamir, An efficient identification scheme based on permuted kernels, *Advances of Cryptology : Proceedings of CRYPTO'89*, Lecture Notes in Computer Science, Vol. 435, Springer-Verlag, Berlin, 1987, pp. 606-609.

[St89] J. Stern, An alternative to the Fiat-Shamir protocol, *Advances of Cryptology : Proceedings of EUROCRYPT'89*, Lecture Notes in Computer Science, Vol. 434, Springer-Verlag, Berlin, 1990, pp. 173-180.

[St93] J. Stern, A new identification scheme based on syndrome decoding, *Advances of Cryptology : Proceedings of CRYPTO'93*, to appear.