Tatra Mt. Math. Publ. **41** (2008), 19–32

TATRA MOUNTAINS
Mathematical Publications

RSA SIGNATURE SCHEMES WITH SUBLIMINAL-FREE PUBLIC KEY

Viktória I. Villányi

ABSTRACT. The problem of subliminal channels in the signatures was already studied in the previous literature. In this paper we focus on the problem of subliminal communication through the public verification key. We show a construction which derives a subliminal-free RSA public key. Along the construction we use a computationally binding and unconditionally hiding commitment scheme. To establish a subliminal free RSA modulus n, we have to construct the secret primes p and q. To prove p and q are primes we use Lehmann's primality test on the commitments. We show our "public key subliminal free" signature scheme is indistinguishable from "regular" RSA signature schemes. When we combine our key generation with the existing subliminal free RSA-PSS signature scheme then we get a signature scheme which is subliminal free in the sense of public key and signature.

1. Introduction

The history of subliminal channels in cryptography goes back to 1983 when Simmons published a paper with the title: *The prisoners' problem and the subliminal channel* [Sim84]. In the paper he introduces subliminal channels through an example: A warden enables two prisoners to communicate through signed messages, but the messages will be opened and read by him. Here the question arises, is it possible for the prisoners to communicate secretly through the non-secret channel; i.e., is it possible to establish a subliminal channel? The answer was yes. The solution was they sacrifice some of their ability of authentication to open a subliminal channel for secret communication through the signature. The next question pops up; is it possible to obtain a subliminal-free authentication and signature? Desmedt in [Des88] showed the existence of subliminal free authentication system. He made the Goldwasser-Micali-Rivest signature system subliminal free by using commitments on the random values.

2000 Mathematics Subject Classification: 94A60.
Keywords: subliminal communication, RSA, zero-knowledge proof.

In [BS05] B o h l i and S t e i n w a n d t laid on a definition of subliminal channels in digital signatures. Here our goal is to avoid secret communication in asymmetric signature schemes through the public verification key. In our contribution we give a definition of signature scheme, where the public key contains a subliminal channel and give a definition for a subliminal-free public key in existentially unforgeable signature schemes. We show an example how to construct a subliminal-free RSA public key, if we combine it with the subliminal--free deterministic RSA-PSS from [BS05], we get a version of RSA-PSS which is subliminal-free in the sense of the public key and the signature.

At first let us see and discuss some examples for subliminal channels in the public key. We can easily find narrow band channels in an RSA public key, for example: choose p, q such that the product's last few (binary) digits (of course the very last one is always 1) encode the subliminal message. If we could factor the RSA modulus with some extra information (subliminal secret key), then we could easily establish a broadband subliminal channel, simple taking one of the primes to be the encoded subliminal message. We could avoid broadband subliminal channels by verifiable randomness [JG02]. In our set up we need more, according to our (later established) definition we would like to have a public key which does not contain one single bit subliminal message with more than negligible probability.

2. Set up definitions

2.1. Preliminaries

Let us recall some definitions. First of all the definition of the negligible function [Gol01].

Definition 2.1.1. A function $\mu : N \to R$ is called negligible in n if for every positive polynomial $p(\cdot)$ and all sufficiently large n's, it holds that $\mu(n) \leq \frac{1}{p(n)}$. We will use the notation $negl(n)$ for these functions.

We will need the standard definition of the signature scheme.

Definition 2.1.2. A signature scheme S=(*Gen, Sig, Ver*) is a triple of algorithms, where

- *Gen* is a probabilistic polynomial time (*ppt*) algorithm that takes the security parameter 1^k as input and returns a pair of public and private key (pk, sk).
- *Sig* is a *ppt* algorithm that takes a message M and the private key sk as input and produces a valid signature σ for M under sk.

– *Ver* is a deterministic polynomial time algorithm that takes a message M, a signature σ and the public verification key pk as input, and returns valid if σ is a valid signature for M w.r.t. pk and invalid otherwise.

2.2. New definitions

We modify the above definition to set up the definition of a signature scheme which contains a subliminal channel in the public key. To this aim we introduce three new algorithms *SGen*, *Embed* and *Extract*. The *SGen* algorithm generates the subliminal secret key (ssk) before the public key secret key pair generation is done. The *Embed* algorithm embeds the subliminal message in the public key by using the ssk. The subliminal message receiver by using the *Extract* algorithm has to be able to recover the subliminal message with overwhelming probability. We say the probability is overwhelming if it is $1 - negl(n)$. To establish a subliminal channel we need the public key generation, where a public key containing subliminal message is indistinguishable from the one which does not. We also need the receiver to be able to recover the message with overwhelming probability.

Definition 2.2.1. A signature scheme where a public key contains a subliminal channel $S = (Gen, Sig, Ver, SGen, Embed, Extract)$ is a tuple of algorithms, where *Gen, Sig, Ver* are as in Definition 1, and

– *SGen* is a *ppt* algorithm whose input is security parameter k and its output is the subliminal secret key $\{ssk\}$.

– *Embed* is a *ppt* algorithm whose inputs are the subliminal message m and subliminal secret key $\{ssk\}$ and its output is a public/secret key pair.

– *Extract* is a *ppt* algorithm whose inputs are subliminal secret key $\{ssk\}$ and the public key Pk' which contains the subliminal message and its output is the embedded message with overwhelming probability.

For all values of k, the subliminal message space has to contain at least two different messages, and for all *ppt* algorithms W (wardens) we require

$$\left| P\left[\mathbf{Exp}^{ward-ind-1}(k) = 1\right] - \left[\mathbf{Exp}^{ward-ind-0}(k) = 1\right] \right| \leq negl(k), \quad (1)$$

where for $b \in \{0, 1\}$ the Experiment $\mathbf{Exp}^{ward-ind-b}(k)$ is defined as a follows:

Experiment $\mathbf{Exp}^{ward-ind-1}(k)$
$(st, m) \leftarrow W(1^k)$; ($st$ is a state information)
$(pk, sk) \leftarrow Gen(1^k)$;
$d \leftarrow W^{S_{sk}(.)}(pk, st)$;
return d;

Experiment $\mathbf{Exp}^{ward-ind-0}(k)$
$(ssk) \leftarrow SGen(1^k)$;
$(st, m) \leftarrow W(1^k)$; ($st$ is a state information)
$(pk^*, sk^*) \leftarrow Embed(m, ssk)$;
$d \leftarrow W^{S_{sk^*(\cdot)}}(pk^*, st)$;
return d;

with $S_{sk}(\cdot)$ an oracle which on input the message (M) returns the signature $\sigma \leftarrow Sig(M, sk)$ and $S_{sk*}(\cdot)$ an oracle which on input the message (M) returns the signature $\sigma \leftarrow Sig(M, sk^*)$.

We get to the point to set up the definition of the public key subliminal free signature scheme. In our definition we have an active warden who participates in the secret/public key generation process. The so called warden task is to be sure the public key does not contain any subliminal messages. Let us call Alice the one who is establishing her secret/public key pair. In our definition we will have an interactive key generation between Alice and the warden. At first Alice sends the auxiliary information to the warden about her public keys. The warden sends her an algorithm what kind of modification she has to do on her keys. After that she sends the new public key with proof about the required modification which has been made to the warden. The warden has to be able to verify the proof and accept or deny the new public key based on the given proof. The proof is usually a zero knowledge proof but in some case it could be omitted. For example, in EC--DSA, if the warden asks Alice to add the multiple of a base point to the public key to make the public key subliminal free. The warden easily can check whether the modification was made or not without some extra proof. In our model we prefer to have signature schemes which are existentially unforgeable. We will have this extra restriction on the signature schemes both the original signature scheme and the subliminal-free variant have to be existentially unforgeable.

Definition 2.2.2. Interactively generated subliminal-free public key in existentially unforgeable signature schemes.

Let $S = (Gen,\ Sig,\ Ver)$ be an existentially unforgeable signature scheme where Pk' is a public key. We call the signature scheme public key (Pk') subliminal (message) free, if there exists a *ppt* algorithm Warden such that for all *ppt* algorithms $SGen,\ BadEmbed,\ Warden,\ BadExtract$ we have

$$\left| P\left[\mathbf{Exp}^{\text{signer}-1}(k) = 1\right] - P\left[\mathbf{Exp}^{signer-0}(k) = 1\right] \right| \leq negl(k). \qquad (2)$$

The experiment $\mathbf{Exp}^{signer-b}$ for $b = 0, 1$ is defined as follows:

Experiment $\mathbf{Exp}^{signer-b}$

$$
\begin{array}{lcl}
(ssk) & \leftarrow & SGen(1^k) \\
(Pk^*, Sk^*, s) & \leftarrow & BadEmbed(b, ssk) \\
a & \leftarrow & A(s) \\
(WAlgorithm) & \leftarrow & Warden(a) \\
(Pk', Sk', proof) & \leftarrow & WAlgorithm(s) \\
d & \leftarrow & BadExtract(Pk', ssk) \\
\text{return } d; & &
\end{array}
$$

where

- *SGen* is a *ppt* algorithm whose input is security parameter k and its output is the subliminal key ssk.
- *BadEmbed* is a *ppt* algorithm whose inputs are the subliminal bit b and subliminal secret key (ssk) and its output are the public/secret key pair and a state information.
- A is a *ppt* algorithm whose input is s and its output is auxiliary information a.
- *Warden* is a *ppt* algorithm whose input is the auxiliary information and its output is *WAlgorithm*.
- *WAlgorithm* is a *ppt* algorithm which gives instruction how to modify the secret/public key pairs. Its input is the state information and its output is a new secret key, a subliminal-free public key and a proof. By the use of the proof it has to be verifiable if *WAlgorithm* was applied. We want $S = (Gen^*,\ Sig,\ Ver)$, where $Gen*$ is the algorithm which generates the subliminal free-public/secret key pair, to be existentially unforgeable (like the original signature scheme) even if we can use all the additional information from the generation process.
- *BadExtract* is a *ppt* algorithm whose inputs are the modified public key (Pk'), and the subliminal secret key and its output is 0 or 1 (guess for the hidden bit).

3. Subliminal-free public key construction

3.1. The basic construction

We will show the construction of a subliminal-free RSA public key. The RSA public key is a pair of the encryption exponent and the modulus (e, n), where n is big enough to be infeasible to factor and e and $\varphi(n)$ are relatively primes. We suppose n is the product of two publicly unknown k bit prime numbers. The subliminal-free public/secret key establishment is an interactive process between

the person who needs this key pair, call her Alice, and the warden who is taking care of the public key's subliminal freeness. For this purpose the warden and Alice will generate the public/secret key pair in the following way:

The public exponent e is a prime number and it is chosen by the warden. The subliminal-free modulus generation is an interactive procedure between Alice and the warden. Alice chooses a 2^{k-1} bit number y randomly and sends a commitment on it to the warden. The warden chooses a 2^{k-1} bit random number z. Alice has to add these two number (mod 2^k) let $x := y + z \pmod{2^k}$ and she has to find the smallest prime p after $x + 2^k$ (2^k is an initial value takes care p and q has the right size). She has to prove with zero-knowledge proofs this is the smallest prime in the row so there is not any number $s \in [x + 2^k, p)$ what is prime. Alice and the warden have to repeat the above process to get the prime q, which is the first prime after $x^* + 2^k$ ($x^* = y^* + z^* \pmod{2^k}$, where y^* and z^* is chosen by Alice and the warden respectively). The product of p, q, call it n, will be a subliminal-free RSA modulus, where a computation will be performed on the commitment and later this commitment will be opened to reveal n.

3.2. Commitment scheme

Along the construction we will use the Pedersen commitment scheme. We assume we have a large order group $G =< g >$ of known order group Q and we have a second generator of this group h whose discrete logarithm to the base g is unknown. The discrete logarithm of y to base g is any integer x such that $y = g^x$. The commitment c_a on a is $g^a h^r$ group element from G, where r is randomly chosen from Z_Q. The security of the commitment scheme depends on the assumption computing discrete logarithm is infeasible. The infeasibility of the computation of the discrete logarithm assures the computationally binding property, the multiplication with the power of h takes care of the unconditionally hiding property. We will adopt the notation from the paper [CM99].

We denote the protocol which proves the knowledge of:

- discrete logarithm x of the group element y to base g by $PK\{(x) : g^x\}$,
- the representation of the element y to the bases $g_1 \dots g_l$ by $PK\{(\alpha_1 \dots \alpha_l) : y = \prod_{i=1}^{l} g_i^{\alpha_i}\}$,
- equality of the discrete logarithms of elements y_1 and y_2 to the base g and h, respectively by $PK\{(\alpha) : y_1 = g^\alpha \wedge y_2 = h^\alpha\}$,
- (at least) one out of the discrete logarithms of the elements y_1, y_2 to base g by $PK\{(\alpha, \beta) : y_1 = g^\alpha \vee y_2 = g^\beta\}$,

- a discrete logarithm that lies in a given range ($2^{l_1} - 2^{l_2} < \log_g y < 2^{l_1} + 2^{l_2}$, for some parameters l_1 and l_2) by $PK\{(\alpha) : y = g^{\alpha} \wedge\ 2^{l_1} - 2^{\ddot{l}_2} < \alpha < 2^{l_1} + 2^{\ddot{l}_2}\}$. [1]

We denote the zero knowledge computation protocols:

- addition $S_+ := PK\{(x, y, z, n, \widetilde{x}, \widetilde{y}, \widetilde{z}, \widetilde{n}) : g^x h^{\widetilde{x}} \wedge g^y h^{\widetilde{y}} \wedge g^z h^{\widetilde{z}} \wedge g^n h^{\widetilde{n}} \wedge z = x + y \pmod n\}$,
- multiplication $S_* := PK\{(x, y, z, n, \widetilde{x}, \widetilde{y}, \widetilde{z}, \widetilde{n}) : g^x h^{\widetilde{x}} \wedge g^y h^{\widetilde{y}} \wedge g^z h^{\widetilde{z}} \wedge g^n h^{\widetilde{n}} \wedge z = x \cdot y \pmod n\}$,
- exponentiation $S_{exp} := PK\{(x, y, z, n, \widetilde{x}, \widetilde{y}, \widetilde{z}, \widetilde{n}) : g^x h^{\widetilde{x}} \wedge g^y h^{\widetilde{y}} \wedge g^z h^{\widetilde{z}} \wedge g^n h^{\widetilde{n}} \wedge z = x^y \pmod n\}$,
- primality checking $S_p := PK\{(p, \widetilde{p}) : g^p h^{\widetilde{p}} \wedge p \in \text{pseudoprimes}(t)\}$, where t is a security parameter.

3.3. Construction in details

As we mentioned earlier the public exponent e is chosen by the warden. The modulus n will be the result of an interactive modulus generation process between Alice and the warden. Alice chooses a random number $y \in \{0, 1, 2 \ldots 2^k - 1\}$ and sends a commitment c_y on it to the warden. The warden chooses a random number $z \in \{0, 1, 2 \ldots 2^k - 1\}$ and sends it to Alice. The natural commitment $c_z = g^z$ on z is easily computable by both parties if z is known. We can omit the multiplication by the power of the other (h) group generator element because the hiding property is not needed here. She has to add z and y modulo 2^k let this sum be $x := z + y \pmod{2^k}$ and the commitment on it c_x. Alice uses the addition protocol to prove she did the addition:

$$S_+ := PK\{(x, \widetilde{x}) : c_x = g^x h^{\widetilde{x}} \ \wedge\ x = y + z \pmod{2^k}\}.^2$$

To prove Alice found the next prime in the row after $x + 2^k$ and $x^* + 2^k$ we use zero knowledge proofs again. We need it because Alice want to reveal neither x, x^*, nor p, q it would be the disclosure of her secret key. She has to prove that p, q are primes generated by the given method but the only thing what she publishes is n the product of them.

To prove p and q are primes, we use the S_p protocol which by using Lehmann's Primality Test, statistically proves the primality of a committed number. We give a protocol to prove the numbers (s) in the interval $[x + 2^k,\ p)$ and $[x^* + 2^k,\ q)$ are not a primes. The protocol based on the Lehmann's Primality Test.

---

[1] With the proving technique from [CM99] if α lies in the interval $(2^{l_1} - 2^{l_2}, 2^{l_1} + 2^{l_2})$, we can prove $2^{l_1} - 2^{\epsilon l_2 + 2} < \alpha < 2^{l_1} + 2^{\epsilon l_2 + 2}$, where $\epsilon > 1$ is a security parameter. We denoted $\epsilon l_2 + 2$ by $\ddot{l}_2$ in the protocol.

[2] We omitted from the protocol the proof of commitment c_y and c_z, the commitment on y and z respectively, because it was shown previously.

Let us recall:

Theorem 3.3.1. *Lehmann's Primality Test [Leh82, CM99]: An odd integer $s > 1$ is prime if and only if*

$$\forall\, a \in \mathbb{Z}_n^*\ a^{\frac{s-1}{2}} \equiv \pm 1 \pmod{s} \quad \text{and} \quad \exists\, a \in \mathbb{Z}_n^*\ a^{\frac{s-1}{2}} \equiv -1 \pmod{s}.$$

We use the contrapositive of a variation of this theorem.

Let us see a variation:

An odd integer $s > 1$ is prime if and only if

$$\forall\, a \in \mathbb{Z}_n \backslash \{0\}\ a^{\frac{s-1}{2}} \equiv \pm 1 \pmod{s} \quad \text{and} \quad \exists\, a \in \mathbb{Z}_n^*\ a^{\frac{s-1}{2}} \equiv -1 \pmod{s}.$$

The contrapositive of it:

Corollary 3.3.2. *An odd integer $s > 1$ is not a prime if and only if*

$$\exists\, a \in \mathbb{Z}_s \backslash \{0\}\ a^{\frac{s-1}{2}} \neq \pm 1 \pmod{s}.$$

To prove s is not a prime, use the following protocols.

$$S_{a\neq 0} := PK\big\{(a, \widetilde{a}, o, \widetilde{o}) : c_a = g^a h^{\widetilde{a}} \wedge c_o = g^o h^{\widetilde{o}} \wedge oa \equiv 1 \pmod{r}\big\}.$$

The r is a publicly known prime number of size 2^{k+2} bit. It ensures all $s \in \big((x,p)$ or $(x^*,q)\big)$ will be relatively prime to r.

$$S_{exp} := PK\big\{(b, \widetilde{b}, d, \widetilde{d}) : c_b = g^b h^{\widetilde{b}} \wedge c_d = g^d h^{\widetilde{d}} \wedge a^b \equiv d \pmod{s}\big\},$$

where $s = 2b + 1$ and $c_s = {c_b}^2 \cdot g$.

$$S_{d\neq\pm 1} := PK\big\{(z, \widetilde{z}) : c_z = g^z h^{\widetilde{z}} \wedge z(d-1)(d+1) \equiv 1 \pmod{r}\big\},$$

where $c_{d-1} := c_d/g \wedge c_{d+1} := c_d \cdot g$.

We would like to prove for consecutive numbers they are not primes with the above protocols. We know every other number is even, so it is enough to verify only for the odd numbers they are not primes. The commitments on the odd numbers s in the intervals $[x + 2^k, p)$ and $[x^* + 2^k, q)$ are in the form

$c_s := c_x \cdot g^{2^k} \cdot g^{2l}$ if x is odd, or $c_s := c_x \cdot g^{2^k} \cdot g^{2l+1}$ if x is even
and
$c_s := c_{x^*} \cdot g^{2^k} \cdot g^{2l}$ if x^* is odd, or $c_s := c_x \cdot g^{2^k} \cdot g^{2l+1}$ if x^* is is even,

where $l \in \mathbb{N} \cup \{0\}$.

Alice and the warden have to run the protocols $S_{a\neq 0}, S_{exp}, S_{d\neq\pm 1}$ on the odd $s \in (x,p)$ and $s \in (x^*,q)$ to prove s is not prime, where c_s is already given. If Alice tries to fool the warden and proves for the even numbers, which are not primes, instead of odd numbers, she never will get a prime number in the row, so never could generate the RSA modulus. If she would manage to do it somehow

when she sends n to the warden, he could immediately recognize n is even, so the fraud would be detectable.

When Alice proved for all numbers in the given intervals $[x + 2^k, p')$ and $[x^* + 2^k,\ q')$ are not primes we derive a commitment on p and q, let them be $c_p := g^p h^{\widetilde{p}}$ and $c_q := g^q h^{\widetilde{q}}$. To prove p and q are prime numbers Alice uses S_p protocol for p and q.

$$S_p := PK\big\{(p, \widetilde{p}) : g^p h^{\widetilde{p}} \wedge p \in \text{pseudoprimes}(t)\big\}.$$

We have to prove $\varphi(p \cdot q)$ and e are relatively primes what is equivalent to e is relatively prime to $p-1$ and to $q-1$ so there exists an inverse of $e \pmod{p-1}$ and $\pmod{q-1}$. The commitment on $p-1$ and on $q-1$ is easily computable from the commitment on p (c_p) and on q (c_q).

To prove e and $\varphi(n)$ are relatively prime use the following protocol:

$$S_{p-1} := PK\big\{(k, \widetilde{k}) :\ c_k = g^k \cdot h^{\widetilde{k}} \wedge\ k \cdot e \equiv 1 \pmod{p-1}\big\}$$

$$S_{q-1} := PK\big\{(l, \widetilde{l}) :\ c_l = g^l \cdot h^{\widetilde{l}} \wedge\ l \cdot e \equiv 1 \pmod{q-1}\big\}.$$

Alice sends $n = p \cdot q$, the RSA modulus to the warden, but she needs to prove n is a product of the committed p and q.

Let us use S_* protocol to prove the product of p and q is n.

$$S_* := PK\big\{(p, q, n, \widetilde{p}, \widetilde{q}, \widetilde{n}) : g^p h^{\widetilde{p}} \wedge g^q h^{\widetilde{q}} \wedge g^n h^{\widetilde{n}} \wedge n = p \cdot q\big\}.$$

4. Proof of subliminal freeness

4.1. Subliminal freeness of our RSA-PSS

Proposition 4.1.1. *Our public key, where e is chosen by the warden and n generated by the above method is subliminal free, in respect of our Definition 2.2.1, under the RSA and the discrete logarithm assumption.*

P r o o f. We will prove it in three steps. In the first step we will prove our public key subliminal-free signature scheme is indistinguishable from the original one. In the second step we will prove the security of our scheme with the zk security proofs is the same as the security of the original RSA-PSS. In the last step we will prove our signature scheme is subliminal free.

Step 1: The proof that our public key subliminal-free signature scheme is ***indistinguishable*** *from the original one.*

Because of the signing and verification algorithms have not been changed, we only have to focus on our public key being indistinguishable from the honest RSA-PSS public key.

Claim. *Experiment 1 is indistinguishable from Experiment 2.*

Experiment 1: Honest RSA prime generation: Generate a random $k-1$ digit number and add 2^k initial value to this number and find the next prime in the row [BD93].
Experiment 2: Our RSA prime generation: Take the sum of two independently generated random $k-1$ bit numbers (mod 2^k), add 2^k initial value to this random number, and find the next prime in the row.
Proof: The random number from *Experiment 1* is indistinguishable from our random number from *Experiment 2* if at least one of the numbers in the sum $(y+z)$ is a random number what is satisfied. It implies the product of two above manner generated primes is indistinguishable from the honestly generated RSA modulus. It also implies our public key subliminal free signature scheme is EF-CMA secure because it is indistinguishable from the original scheme which was supposed to be EF-CMA secure.

Handling the failure property of the above experiments:

The chance that indistinguishable *Experiment*s will not be terminate, can be computed by using the Gallagher conjecture:

$$\sharp\big\{\text{integers } x \le X:\ \varphi(x+\lambda\ln x)-\varphi(x)=k\big\} \approx e^{-\lambda}\frac{\lambda^k}{k!}X$$
$$\text{for any fixed } \lambda>0 \quad \text{and} \quad \text{integer } k\ge 0.$$

This implies the probability of $[x, x+\lambda\ln x]$ not containing any prime to be at most $e^{-\lambda}$. The $\varphi(n)$ of a positive integer n is defined to be the number of positive integers less than or equal to n that are coprime to n, where 1 is counted as being relatively prime to all numbers.

Step 2: Our public key subliminal free scheme with the proof is as ***secure*** *as the original RSA-PSS.*

The proof is unconditionally hiding zero knowledge so we cannot get any useful information out of it. The only extra information we will get out from the proof is the number of non-primes (call it l) between the random numbers and first prime. Let us see how we can use this information to factor RSA modulus. We know the gap between our prime and the previous prime is at least l. We also know from the *Gallagher conjecture* the probability of the gap is greater than $\lambda\ln x$ is less than $e^{-\lambda}$. Let us fix the λ previously and if we cannot find prime until $x+\lambda\ln x$, then we begin the search again choosing an other random value and try to find a next prime in the row, then we could maximize the size of the gap and it would be polynomial in the size of the security parameter. If the gap size is polynomial, then a probabilistic polynomial time adversary algorithm could guess it. If it is possible to factor n with the knowledge of the gap size, then it would be possible without it,too, with simply guessing the size.

Step 3: Our signature scheme is ***subliminal free****.*

The public exponent is subliminal-free because it was chosen by the warden. We claim our RSA modulus n is subliminal free. Let us see how could Alice hide a subliminal message in the public key. If she tries to hide it in the generation of y or y^*, the warden would add the random number to them so the probability of the subliminal message would be recoverable and will not be overwhelming anymore. If she tries to hide it by the addition or by proving she found the next prime, she would have got again just a negligible probability (computationally hiding and the probabilistic primality proof). We can claim Alice has a subliminal-free public key. □

We would like to have a subliminal free signature scheme in the sense of the signature and the public key. If we combine a signature subliminal-free deterministic RSA-PSS, see [BS05], with our public key subliminal free RSA, we will derive a signature scheme which is subliminal free in the sense of the public key and the signature as well.

5. The size of the proof

We will give an estimate of the zero-knowledge proofs space requirement. The group in which we will perform zero-knowledge proofs, will be a prime order group of order $Q > 2^{2\epsilon(k+1)+5}$, where ϵ is a security parameter and the prime factors p and q are less than 2^{k+1} .

5.1. Detailed protocols

We give a detailed variant some of the protocols from the construction to estimate a size of the proof:

$$S_+ := PK\Big\{(x,\widetilde{x},q) : c_x = g^x h^{\widetilde{x}} \wedge -2^{\ddot{\ell}} < x < 2^{\ddot{\ell}} \\ \wedge \ \frac{c_x}{c_y \cdot g^z} = \left(g^{2^k}\right)^q \wedge -2^{\ddot{\ell}} < q < 2^{\ddot{\ell}}\Big\},$$

$$S_{a\neq 0} := PK\Big\{(a,\widetilde{a},o,\widetilde{o},s,\widetilde{s}) : c_a = g^a h^{\widetilde{a}} \wedge \ -2^{\ddot{\ell}} < a < 2^{\ddot{\ell}} \ \wedge \ c_o = g^o h^{\widetilde{o}} \\ \wedge \ -2^{\ddot{\ell}} < o < 2^{\ddot{\ell}} \ \wedge \ g = c_o^a \cdot {c_r}^s h^{\widetilde{s}} \\ \wedge \ -2^{\ddot{\ell}} < s < 2^{\ddot{\ell}}\Big\},$$

$$S_{exp} := PK\Big\{\Big(b,\widetilde{b},d,\widetilde{d}\Big) : c_b = g^b h^{\widetilde{b}} \wedge -2^{\ddot{\ell}} < b < 2^{\ddot{\ell}} \wedge c_d = g^d h^{\widetilde{d}} \\ \wedge -2^{\ddot{\ell}} < d < 2^{\ddot{\ell}} \wedge a^b \equiv d \pmod{s}\Big\},$$

$$S_{d\neq\pm1} := PK\Big\{\Big(z,\widetilde{z},f,\widetilde{f},s,\widetilde{s},t,\widetilde{t}\Big) : c_z = g^z h^{\widetilde{z}} \wedge -2^{\ddot{\ell}} < z < 2^{\ddot{\ell}} \wedge c_f = g^f h^{\widetilde{f}} \\ \wedge -2^{\ddot{\ell}} < f < 2^{\ddot{\ell}} \wedge c_f = c_{d-1}^{d+1} \cdot c_r^s h^{\widetilde{s}} \\ \wedge -2^{\ddot{\ell}} < s < 2^{\ddot{\ell}} \wedge g = c_f^z \cdot {c_r}^t h^{\widetilde{t}} \\ \wedge -2^{\ddot{\ell}} < t < 2^{\ddot{\ell}}\Big\},$$

$$S_{p-1} := PK\left\{\left(k,\widetilde{k},r,\widetilde{r}\right) : c_k = g^k \cdot h^{\widetilde{k}} \wedge -2^{\ddot{\ell}} < k < 2^{\ddot{\ell}} \wedge g = c_k^e \cdot \left(\frac{c_p}{g}\right)^r h^{\widetilde{r}} \\ \wedge -2^{\ddot{\ell}} < r < 2^{\ddot{\ell}}\right\}.$$

5.2. The estimation with chosen parameters

Proving p and q are primes results in a communication costs of about

$$14t\log(2^{k+1})\log Q + 4t\log(2^{k+1})\epsilon l = 14t(k+1)\log Q + 4t(k+1)\epsilon l \text{ bit}$$

for p and q separately.

Proving the number s is not a prime number costs about

$$2\log Q(3+7(k+1)+4) + \epsilon l\cdot(3+4(k+1)+4) = 14(k+2)\log Q + (4k+11)\epsilon l.$$

We have to prove about for $\frac{k+1}{2}$ numbers they are not primes until we find the first prime p (or q) number in the row. The size of this proof is about

$$\frac{k+1}{2}\cdot\big(14(k+2)\log Q + (4k+11)\epsilon l\big)$$

for both interval. The computation cost of the remaining operation is about

$$2\log Q(2+2\cdot2+1) + (2+2\cdot2)\epsilon l = 14\log Q + 6\epsilon l.$$

The full cost of the protocol is about

$$2\big(14t(k+1)\log Q + 4t(k+1)\epsilon l + \frac{k+1}{3} \\ \times(14(k+2)\log Q + (4k+11)\epsilon l) + 7\log Q + 3\epsilon l\big).$$

Let us see a concrete estimation: if we choose

$$k = 512,\quad \epsilon = 1\cdot\frac{1}{\sqrt{80}} \approx 1.11,\qquad t = 80,\ l = 80,$$

then $\log Q$ will be about

$$\log 2^{2\cdot 1.11\cdot 513+5}$$
$$\approx 1145,2 \times \left(79.86 + 1.76 + \frac{513}{2}(0.98 + 0.02) + 0.001\right) Mbyte$$
$$\approx 676\, Mbyte.$$

Let us see the security of the full protocol with these chosen parameters. The probability for the forgery from the primality test is about $\frac{1}{2^{80}}$.

To save space we can omit the numbers which are divisible by 3, 5, 7 ... prime numbers up to $\sqrt{k}$. In our case it is 19 ($k = 512$). We apply a little sieve for the intervals. If we can omit one number, we can gain 1Mbyte. We can reduce the size to 339 Mbyte. Here the size of the primality test is 163 Mbyte and the proof that the numbers between the random number and the prime number, not being primes, is 176 Mbyte.

Acknowledgements. I would like to thank to Rainer S t e i n w a n d t for valuable discussions and comments.

REFERENCES

[AVPN96] ANDERSON, R.—VAUDENAY, S.—PRENEEL, B.—NYBERG, K.: *The Newton channel*, in: The First International Workshop on Information Hiding (R. J. Anderson, ed.), Lecture Notes in Comput. Sci., Vol. 1174, Springer-Verlag, Berlin, 1996, pp. 151–156.

[BD93] BRANDT, J.—DÅMGARD, I.: *On generation of probable primes by incremental search*, in: Adv. in Cryptology (E. F. Brickell, ed.), Lecture Notes in Comput. Sci., Vol. 740, Springer-Verlag, Berlin, 1993, pp. 358–370.

[BGVS07] BOHLI, J.-M.—GONZÁLEZ VASCO, M. I.—STEINWANDT, R.: *A subliminal-free variant of ECDSA*, in: 8th International Workshop—IH '06 (J. Camenisch et al., eds.), Lecture Notes in Comput. Sci., Vol. 4437, Springer-Verlag, Berlin, 2007, pp. 375–387.

[BS05] BOHLI, J.-M.—STEINWANDT, R.: *On subliminal channels in deterministic signature schemes*, in: Security and Cryptology (Ch. Park et al., eds.), Lecture Notes in Comput. Sci., Vol. 3506, Springer-Verlag, Berlin, 2005, pp. 182–194.

[CM99] CAMENISCH, J.—MICHELS, M.: *Proving in zero-knowledge that a number is the product of two safe primes*, in: Adv. in Cryptology (J. Stern, ed.), Lecture Notes in Comput. Sci., Vol. 1592, Springer-Verlag, Berlin, 1999, pp. 107–122.

[Des88] DESMEDT, Y.: *Subliminal-free authentication and signature (Extended abstract)*, in: Adv. in Cryptology (Ch. Günther, ed.), Lecture Notes in Comput. Sci., Vol. 330, Springer-Verlag, Berlin, 1988, pp. 23–33.

[Gol01] GOLDREICH, O.: *Foundation of Cryptography*, Cambridge University Press, Cambridge, 2001.

[JG02] JUELS, A.—GUAJARDO, J.: *RSA key generation with verifiable randomness*. In: 5th International Workshop on Practice and Theory in Public Key Cryptosystems (D. Naccache, ed.), Lecture Notes in Comput. Sci., Vol. 2274, Springer-Verlag, Berlin, 2002, pp. 261–285.

[Lab02] RSA Laboratories. *PKCS #1 v.2.1: RSA Cryptography Standard*. `ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf`.

[Leh82] LEHMANN, D. J.: *On primality tests*, SIAM J. Comput. **11** (1982), 374–375.

[Sim84] SIMMONS, G. J.: *The prisoner's problem and the subliminal channel*, in: Adv. in Cryptology (D. Chaum, ed.), Plenum Press, New York, 1984, pp. 51–67.

[YY06] YOUNG, A.—YUNG, M.: *A space efficient backdoor in RSA and its applications*, in: Selected Areas in Cryptography (B. Preneel, ed.), Lecture Notes in Comput. Sci., Vol. 3897, Springer-Verlag, Berlin, 2006, pp. 128–143.

Received September 25, 2007

Department of Mathematical Sciences
Florida Atlantic University
777 Glades Road
Boca Raton, FL 33431
USA
E-mail: vvillan@fau.edu