A New and Efficient Fail-stop Signature Scheme

WILLY SUSILO[1], REI SAFAVI-NAINI[1], MARC GYSIN[2] AND JENNIFER SEBERRY[1]

[1]*Centre for Computer Security Research, School of IT and CS, University of Wollongong, Wollongong 2522, Australia*
[2]*School of Information Technology, Townsville, QLD 4811, Australia*
Email: wsusilo@uow.edu.au

The security of ordinary digital signature schemes relies on a computational assumption. Fail-stop signature schemes provide security for a sender against a forger with unlimited computational power by enabling the sender to provide a proof of forgery if it occurs. In this paper we give an efficient fail-stop signature scheme that uses two hard problems, discrete logarithm and factorization, as the basis of a receiver's security. We show that the scheme has provable security against adaptively chosen message attack, and is the most efficient scheme with respect to the ratio of the message length to the signature length. The scheme provides an efficient solution to signing messages up to 1881 bits.

Received 14 November 1999; revised 13 September 2000

1. INTRODUCTION

Digital signatures, introduced in [1], are the most important cryptographic primitive for providing authentication in the electronic world. The original definition of the digital signature was subsequently revised [2] to ensure security against a more stringent type of attack known as *adaptive chosen message attack*. Despite the stronger requirement, security in digital signature schemes remains computational and hence an enemy with unlimited computing power can always forge a signature. We refer to this type of signature as an *ordinary* signature scheme.

In an ordinary signature scheme, if a forgery occurs the sender must bear its consequences, and there is no way for him to show that a forgery has occurred. This is unavoidable; since there is no way of distinguishing between a forged signature from one generated by the signer, if the signer is allowed to disavow a forged signature, he might also disavow his own signature, resulting in vanishing accountability in the system. This means that the security for the signer is computational and if the underlying computational assumption is broken a forged signature can be irrefutably created. On the other hand the security of the receivers is unconditional, as verification is a public process.

To provide protection against forgeries of an enemy with unlimited computational power, fail-stop signature (FSS) schemes have been proposed [3, 4, 5]. In a FSS, in the case of forgery, the presumed signer can provide a proof that a forgery has happened. This is by showing that the underlying computational assumption of the system is broken. The system will be stopped at this stage—hence the name *fail-stop*. In this way, a polynomially bounded signer can be protected against a forger with unlimited computational power. We note that an unbounded receiver can forge a signature but again a proof of forgery shows that the computational assumption of the system is broken and the system will be stopped. It can be shown that (Theorem 3.2 [3]) a secure FSS can be used to construct an ordinary digital signature that is secure in the sense of [2] and so a fail-stop signature scheme provides a stronger notion of security.

In a FSS there are a number of participants: a *signer* who signs a message that is verifiable by everyone with access to his public key and is protected against forgery of an unbounded enemy, one or more *recipients* and a *centre* who is trusted by the recipient. All the receivers who take part in the key generation process and are convinced about the goodness of the key are protected from repudiation of the signature by the signer. There is another group of participants, the so-called *risk bearers*, such as insurance companies, who will bear a loss if a proof of forgery is accepted and hence a signature is invalidated. For simplicity we do not make any distinction between a recipient and a risk bearer.

In a FSS, the signer and the recipients are assumed to be polynomially bounded, while the enemy is assumed to have unlimited computational power [6, 7, 8]. A system may be designed for one or more recipients. It is important to note that a 'single recipient' system only refers to the protection provided against a signer's repudiation, and signature verification (called *testing* in the context of FSS) can always be performed by anyone who has access to the public key. That is, a single recipient system can be seen as an ordinary signature with the added property that a designated recipient is protected against disavowal of the

signature by the signer, and the signer is protected against an all-powerful forger. These kinds of requirements are very common in electronic commerce systems when a customer primarily interacts with a single financial institution, such as a bank. In this case, it is reasonable to assume that the bank is more powerful and the customer requires protection against possible forgeries of the bank. At the same time, the bank must be assured that the signer cannot repudiate his signature. Using a FSS with a single recipient achieves both these requirements.

In a single recipient FSS, the role of the trusted centre is played by the recipient and hence no trusted centre is required. For a general FSS, eliminating the centre requires a secure multi-party computation (for example, [4, 9, 10]).

A FSS in its basic form is a *one-time digital signature* that can only be used for signing a single message. However, it is possible to extend a FSS scheme to be used for signing multiple messages [8, 11, 12, 13].

To assess the *efficiency* of a FSS scheme a number of criteria, including the lengths of the signature, the secret key and the public key, together with the amount of computation and communication required for signature generation and verification (testing), are used.

1.1. Previous works

The first construction of fail-stop signature [5] uses a one-time signature scheme (similar to [14]) and results in bit-by-bit signing of the message and so is very impractical.

In [15] an efficient single-recipient FSS to protect clients in an on-line payment system is proposed. The main disadvantage of this system is that signature generation is a three-round protocol between the signer and the recipient and so is very expensive in terms of communication. The size of the signature is twice the length of the message.

In [8] an efficient FSS that uses the difficulty of the discrete logarithm problem as the underlying assumption is presented. In the case of a forgery, the presumed signer can solve an instance of the discrete logarithm problem, and prove that the underlying assumption is broken. This is the most efficient scheme known so far and will be referred to as the *vHP scheme* (van Heijst and Pederson).

In [3, 6] a formal definition of FSS schemes is given and a general construction using *bundling homomorphism* is proposed. The important property of this construction is that it is provably secure against the most stringent type of attack, that is, adaptive chosen message attack [16]. The proof of forgery is by showing two different signatures on the same message, the forged one and the one generated by the valid signer. To verify the proof of forgery the two signatures are shown to collide under the 'bundling homomorphism'. An instance of this construction uses the difficulty of factoring as the underlying computational assumption of the system [7].

It is shown [3, 6] that the vHP scheme is in fact an instantiation of this general construction and so has provable security. This, combined with efficiency, has made the vHP scheme the benchmark for FSS schemes.

The existence condition for a FSS is relaxed in [7, 9, 17] and it is shown that a FSS only exists if one-way permutations exist.

In [10], an RSA-based FSS is proposed in which the underlying intractability assumption is the difficulty of factoring, and the proof of forgery is by showing the non-trivial factors of the modulus. In this scheme the size of the signature is twice that of the vHP scheme (four times the size of the message) and compared with [8], has equal or worse performance in all other aspects of interest. The proof of security is through a number of theorems that bound the success probabilities of different attackers.

1.2. Our contributions

In this paper, we propose a new FSS scheme that is almost as efficient as the vHP scheme; its security relies on two well-accepted computational assumptions, discrete logarithm and factorization. We introduce a new measure of efficiency that is related to efficient use of communication bandwidth and show that our scheme outperforms the vHP scheme (and all other schemes based on a factorization problem). We prove that the success chance of an unbounded forger is limited by the recipient's security parameter, while the signer's security against adaptive chosen message attack is guaranteed to a level determined by the sender's security parameter. The proof of forgery is by revealing the non-trivial factors of the modulus. We incorporate the idea of embedding groups from [18] for the construction of our scheme. Finally, we compare the optimality and efficiency between our scheme and the vHP scheme.

The paper is organized as follows. In Section 2, we present the basic concepts and definitions of FSS, and briefly review the general construction and its relevant security properties. In Section 3, we present our FSS construction, show that it is an instance of the general construction [3] and hence provide complete proof of security. In Section 4, we introduce the notions of optimality and efficiency, and give a fair comparison between our scheme and the other existing schemes based on these notions. Finally, Section 5 concludes the paper.

2. PRELIMINARIES

In this section, we briefly recall relevant notions, definitions and requirements of fail-stop signatures and refer the reader to [3, 4, 6] for a more complete account.

2.1. Notation

The length of a number n is the length of its binary representation and is denoted by $|n|_2$. $p|q$ means p divides q.

The ring of integers modulo a number n is denoted by Z_n and its multiplicative group, which contains only the integers relatively prime to n, by Z_n^*. Let N denote the natural numbers.

2.2. Review of fail-stop signatures schemes

Similar to an ordinary digital signature scheme, a fail-stop signature scheme consists of one polynomial time protocol and two polynomial time algorithms.

1. *Key generation*: a two-party protocol between the signer and the centre to generate a pair of keys, a *secret key*, s_k, and a *public key*, p_k. This is different from ordinary signature schemes where key generation is performed by the signer individually and without the involvement of the receiver.
2. *Sign*: the algorithm used for signature generation. For a message m and using the secret key s_k, the signature is given by $y = \text{sign}(s_k, m)$.
3. *Test*: the algorithm for testing acceptability of a signature. For a message m and signature y, and given the public key p_k, the algorithm produces an *ok* response if the signature is acceptable under p_k. That is $\text{test}(p_k, m, y) \stackrel{?}{=} ok$.

A FSS also includes two more polynomial time algorithms.

4. *Proof*: an algorithm for proving a forgery.
5. *Proof-test*: an algorithm for verifying that the proof of forgery is valid.

A secure fail-stop signature scheme must satisfy the following properties [3, 6, 7].

1. If the signer signs a message, the recipient must be able to verify the signature (*correctness*).
2. A polynomially bounded forger cannot create forged signatures that successfully pass the verification test (*recipient's security*).
3. When a forger with an unlimited computational power succeeds in forging a signature that passes the verification test, the presumed signer can construct a proof of forgery and convince a third party that a forgery has occurred (*signer's security*).
4. A polynomially bounded signer cannot create a signature that he can later prove to be a forgery (*non-repudiability*).

To achieve the above properties, for each public key, there exist many matching secret keys such that different secret keys create different signatures on the same message. The real signer knows only one of the secret keys, and can construct only one of the many possible signatures. An enemy with unlimited computing power can generate all the signatures but cannot determine which one is generated by the true signer. Thus, it would be possible for the signer to provide a proof of forgery by generating a second signature on the message with a forged signature, and use the two signatures to show the underlying computational assumption of the system is broken, hence proving the forgery.

Security of a FSS can be broken if (1) a signer can construct a signature that he can later prove to be a forgery, or (2) an unbounded forger succeeds in constructing a signature that the signer cannot prove is forged. These two types of forgeries are completely independent, and so two different security parameters, k and σ, are used to show the level of security against the two types of attacks. More specifically, k is the security level of the recipient and σ is that of the signer. It is proved [3] that a secure FSS is secure against adaptive chosen message attack and for all $c > 0$ and large enough k, success probability of a polynomially bounded forger is bounded by k^{-c}. For a FSS with security level σ for the signer, the success probability of an unbounded forger is limited by $2^{-\sigma}$.

In the following we briefly recall the general construction given in [3] and outline its security properties.

2.3. The general construction

The construction is for a single-message FSS and uses *bundling homomorphisms*. Bundling homomorphisms can be seen as a special kind of hash function.

DEFINITION 2.1. [3] *A bundling homomorphism h is a homomorphism $h : G \to H$ between two Abelian groups $(G, +, 0)$ and $(H, \times, 1)$ that satisfies the following.*

1. *Every image $h(x)$ has at least 2^τ preimages. 2^τ is called* bundling degree *of the homomorphism.*
2. *It is infeasible to find collisions, i.e. two different elements that are mapped to the same value by h.*

To give a more precise definition, we need to consider two families of groups, $\mathcal{G} = (G_K, +, 0)$ and $\mathcal{H} = (H_K, \times, 1)$, and a family of polynomial-time functions indexed by a key, K. The key is determined by the application of a key generation algorithm $g(k, \tau)$, on two input parameters k and τ. The two parameters determine the difficulty of finding collisions and the bundling degrees of the homomorphisms, respectively. Given a pair of input parameters, $k, \tau \in N$, a key K is calculated first, using the key generation algorithm, and then G_K, H_K and h_K are determined. For a formal definition of bundling homomorphisms see Definition 4.1 in [3].

A bundling homomorphism can be used to construct a FSS scheme as follows.

Let the security parameters of the FSS be given as k and σ. The bundling degree of the homomorphism, τ, will be obtained as a function of σ as shown below.

1. *Prekey generation*: the centre computes $K = g(k, \tau)$ and so determines a homomorphism h_K, and two groups G_K and H_K. Let $G = G_K$, $H = K_K$ and $h = h_K$.
2. *Prekey verification*: the signer must be assured that K is a possible output of the algorithm $g(k, \tau)$. This can be through providing a zero-knowledge proof by the centre or by testing the key by the signer. In any case, the chance of accepting a *bad* key must be at most $2^{-\sigma}$.
3. *Main key generation gen_A*: the signer generates her secret key $sk := (sk_1, sk_2)$ by choosing sk_1 and sk_2 randomly in G and computes $pk := (pk_1, pk_2)$ where $pk_i := h(sk_i)$ for $i = 1, 2$.

4. The message space M is a subset of Z.
5. *Signing*: the signature on a message $m \in M$ is
$$s = \text{sign}(sk, m) = sk_1 + m \times sk_2$$
where multiplying by m is m times addition in G.
6. *Testing the signature*: can be performed by checking
$$pk_1 \times pk_2^m \stackrel{?}{=} h(s).$$
7. *Proof of forgery*: given an acceptable signature $s' \in G$ on m such that $s' \neq \text{sign}(sk, m)$, the signer computes $s := \text{sign}(sk, m)$ and *proof* $:= (s, s')$.
8. *Verifying proof of forgery*: given a pair $(x, x') \in G \times G$, verify that $x \neq x'$ and $h(x) = h(x')$.

Theorem 4.1 in [3] proves that for any family of bundling homomorphisms and any choice of parameters the general construction:

1. produces the correct signature;
2. a polynomially bounded signer cannot construct a valid signature and a proof of forgery;
3. if an acceptable signature $s^* \neq \text{sign}(sk, m^*)$ is found the signer can construct a proof of forgery.

Moreover, for two chosen parameters k and σ, a good prekey K and two messages $m, m^* \in M$, with $m \neq m^*$, let
$$T := \{d \in G | h(d) = 1 \wedge (m^* - m)d = 0\}. \qquad (1)$$
Theorem 4.2 in [3] shows that given $s = \text{sign}(sk, m)$ and a forged signature $s^* \in G$ such that $\text{test}(pk, m^*, s^*) = ok$, the probability that $s^* = \text{sign}(sk, m^*)$ is at most $|T|/2^\tau$, and so the best chance of success for an unrestricted forger to construct an undetectable forgery is bounded by $|T|/2^\tau$. Thus to provide the required level of security σ, we must choose $|T|/2^\tau \leq 2^{-\sigma}$.

This general construction is the basis of all known *provably secure constructions* of a FSS. It provides a powerful framework by which proving the security of a scheme is reduced to specifying the underlying homomorphism and determining the bundling degree and the set T.

3. A NEW AND EFFICIENT FSS SCHEME

In this section we introduce a new FSS scheme and show that it is an instance of the general construction. As will be shown in Section 4, the scheme outperforms the most efficient known FSS (i.e. the vHP scheme) with respect to the message length. Proof of forgery is by revealing the secret factors of a modulus and so verifying the proof is very efficient.

Firstly, we describe our scheme with a single recipient model, for simplicity. Then, we extend this model to a multiple recipient scheme.

Model

There is only a single recipient, $\mathcal{R}$, who also plays the role of the trusted centre and performs prekey generation of the scheme.

Prekey generation

Given the two security parameters k and σ, $\mathcal{R}$ chooses two large safe primes p and q. Then, $\mathcal{R}$ finds a prime P such that $n = pq$ divides $P - 1$. Finally $\mathcal{R}$ selects an element α such that the multiplicative order of α modulo P is p ($\text{ord}_P(\alpha) = p$). α, n and P are sent to the signer via an authenticated channel. (More details on selection of these parameters are given below.)

Prekey verification

If the receiver is trusted, the prekey will be accepted by the signer $\mathcal{S}$ and no prekey verification is needed (as in [8]). On the other hand, if the receiver is not trusted, a zero-knowledge proof is needed to ensure that the prekey is correct. This issue will be discussed in the next section (multiple recipient scheme).

Key generation

$\mathcal{S}$ chooses $k_1, k_2 \in Z_n$ and computes
$$\alpha_1 = \alpha^{k_1} \bmod P$$
$$\alpha_2 = \alpha^{k_2} \bmod P.$$
The private key is (k_1, k_2) and the public key is (α_1, α_2).

Signing a message x

To sign a message $x \in Z_n$, $\mathcal{S}$ computes
$$y = k_1 x + k_2 \bmod n$$
and publishes y as his signature on x.

Testing a signature

y passes the test if
$$\alpha^y \stackrel{?}{=} \alpha_1^x \alpha_2 \bmod P$$
holds.

Proof of forgery

If there is a forged signature y' which passes the test, the presumed sender can generate his own signature, namely y, on the same message, and the following equation will hold:
$$\alpha^y = \alpha^{y'} \bmod P$$
or
$$y = y' \bmod p$$
$$y - y' = cp, \quad c \in Z.$$
Hence, a non-trivial factor of n can be found by computing $\gcd(y - y', n)$. We note that the probability of y being equal to y' is $1/q$.

We make the following remarks on the key generation algorithm. In [19], it is shown that for a randomly selected n, P such that n divides $P-1$ is upper bounded by $n\log_2^2 n$. Moreover if $|n|_2 = k$, then on average it takes $O(\log k)$ probabilistic steps to find such a P. An element α is selected such that the multiplicative order of $\alpha \bmod P$ is p ($\text{ord}_P(\alpha) = p$). This element can be easily found by, for example, randomly choosing an element $\tilde{\alpha} \in Z_P^*$ and calculating $\alpha = (\tilde{\alpha})^{cq} \bmod P$, for $c = (P-1)/n$. If $\alpha \neq 1$, then α has order p. This is in fact 'pushing' the element $\tilde{\alpha}$ into a subgroup of order p.

3.1. Security proof

We show that this scheme is an instance of the general construction with the following underlying bundling homomorphism family.

Discrete logarithm bundling homomorphism

- Key generation g: on input k and τ, two primes p and q with $|q|_2 = \tau$, and $|p|_2 \approx |q|_2$, a prime P such that n divides $P-1$ and $|n|_2 = k$, and an element α of order p is chosen. The key will be $K = (p, q, \alpha, P)$.
- Families of groups: let $n = pq$. Define $G_K = Z_n$ and $H_K = Z_P^*$.
- The homomorphism $h_{(p,q,\alpha P)}$ is

$$h_{(p,q,\alpha,P)} : Z_n \to Z_P^* \quad h_{(p,q,\alpha,P)}(x) = \alpha^x \pmod P$$

DISCRETE LOGARITHM (DL) ASSUMPTION. [20] *Given $I = (p, \alpha, \beta)$, where p is prime, $\alpha \in Z_p^*$ is a primitive element and $\beta \in Z_p^*$, where*

$$\alpha^a \equiv \beta \pmod p$$

it is hard to find $a = \log_\alpha \beta$.

FACTORIZATION ASSUMPTION. [21, 20] *Given $n = pq$, where p and q are prime, it is hard to find a non-trivial factor of n (without the knowledge of $\phi(n) = (p-1)(q-1)$).*

STRONG FACTORIZATION ASSUMPTION. *Given $n = pq$ (where p and q are prime), $P = tn + 1$ ($t \in Z$ and P is also prime) and α (where $\text{ord}_P(\alpha) = p$), it is hard to find a non-trivial factor of n.*

This assumption is also used by Brickell and McCurley [18] although there is no proof that knowledge of α of order p cannot reduce the hardness of factoring n.

THEOREM 3.1. *Under DL and strong factorization assumptions, the above construction is a family of bundling homomorphisms.*

Proof. To show that the above definition is a bundling homomorphism, we must show that:

1. for any $\mu \in Z_P^*$ where $\mu = \alpha^c \pmod P$, there are q preimages in Z_n;
2. for a given $\mu \in Z_P^*$ where $\mu = \alpha^c \pmod P$; it is difficult to find c such that $\alpha^c = \mu \pmod P$;
3. it is hard to find two values $c, \tilde{c} \in Z_n$ that map to the same value.

To prove property 1, we note that knowing $\mu = \alpha^c \pmod P$ for $c \in Z_n^*$ and $\text{ord}_P(\alpha) = p$, there are exactly q values c', given by $c' = c + ip$, $i = 0, \ldots, q-1$, for which $\alpha^{c'} = \alpha^{c+ip} = \alpha^c$. Hence, there are q preimages of μ in Z_n^*.

Now given $\mu = \alpha^c \pmod P$, finding c is equivalent to solving an instance of the DL problem, which is hard (property 2).

Property 3 means that it is difficult to find c and $\tilde{c}$ such that $\alpha^c = \alpha^{\tilde{c}} \pmod P$. Suppose that there is a probabilistic polynomial-time algorithm $\tilde{A}$ that could compute such a collision. Then we construct an algorithm $\tilde{D}$ that on input (P, n, α), where $n | P-1$, outputs the non-trivial factors of n as follows.

First, $\tilde{D}$ runs $\tilde{A}$, and if $\tilde{A}$ outputs a collision, i.e. y and $\tilde{y}$, $y \neq \tilde{y}$ such that $\alpha^y \equiv \alpha^{\tilde{y}} \pmod P$, then $\tilde{D}$ computes:

$$\begin{aligned} \alpha^y &= \alpha^{\tilde{y}} \bmod P \\ y &= \tilde{y} \bmod p \\ y - \tilde{y} &= \hat{c}p, \quad \hat{c} \in Z \\ p &= \gcd(y - \tilde{y}, n). \end{aligned}$$

$\tilde{D}$ is successful with the same probability as $\tilde{A}$ and almost equally efficient. Hence, it contradicts with the strong factorization assumption. □

THEOREM 3.2. *Our FSS scheme is secure for the signer.*

According to Theorem 4.1 in [3], we must find the size of the set T:

$$T := \{d \in Z_n | \alpha^d = 1 \wedge (m^* - m)d = 0\}$$

or

$$T := \{d \in Z_n | \alpha^d = 1 \wedge m'd = 0\}$$

in Z_P^*. There are exactly q d's that satisfy the first equation $\alpha^d = 1 \bmod P$. Since $m^* \neq m$, we have $m' \in \{1, 2, \ldots, n-1\}$ and so there is a unique message (namely $m' = q$) that satisfies $m'd = 0 \pmod n$. Hence, $|T| = 1$.

Together with Theorem 4.2 in [3], this implies that it suffices to choose $\tau = \sigma$ in the proposed scheme.

3.2. Multiple recipient scheme

We have restricted ourselves to a single recipient, but it is not difficult to extend the scheme to multiple recipients. In fact, the only difference in such a case is to include a trusted centre and provide zero-knowledge proofs that show that the chosen parameters of the prekey have the correct forms. That is, we need to ensure that n, P and α have the desired forms. Using [22], an element n can be proved to be an RSA modulus $n = pq$, where both p and q are safe primes. Then, P is tested for primality. This can be done by using various primality-testing algorithms such as the Miller–Rabin probabilistic primality test [20] which

runs in polynomial time. Finally it is verified that n divides $P-1$. Although it is easy to show that the order of α is a multiple of p (without knowing p, for example by verifying $\alpha^n \stackrel{?}{=} 1 \pmod P$), showing that the order is strictly p needs more effort. We can achieve the zero knowledge proof of $\mathrm{ord}_P(\alpha) = p$ by combining the idea mentioned in Sections 3.2 and 4.2 of [22]. More precisely, the prover has to prove that he knows p that satisfies $\alpha^p = 1 \bmod P$, and that p is a prime number. On the other hand, after verifying this proof, the receiver (or the sender in the context of this paper) only needs to check whether $\alpha^n \stackrel{?}{=} 1 \bmod P$ and, hence, prove that $\alpha^p = 1 \bmod P$.

4. OPTIMALITY AND EFFICIENCY

The aim of this section is to compare the efficiency of our proposed scheme with those of the best known FSS schemes. Efficiency of a fail-stop signature system has been measured in terms of three length parameters: the lengths of the secret key, the public key and the signature, and the amount of computation required in each case. Later in this section we introduce a new measure, efficiency with respect to message length, which corresponds to efficient use of the communication channel. Pedersen and Pfitzmann [3] proved that if the security level of the sender is σ, and N messages are to be signed, then the size of length parameters are lower bounded by $(N+1)(\sigma-1)$, σ and $2\sigma-1$ respectively. These bounds do not depend on the security level of the receiver which is measured by the parameter k and determines the size of the underlying hard problem(s).

DEFINITION 4.1. *A FSS scheme with security parameters k and σ is called* optimal *with respect to secret key length, public key length or the signature length, if the lower bound on the corresponding parameter is satisfied with equality.*

A comparison

To compare two FSSs we fix the level of security provided by the two schemes, and find the size of the three length parameters and the number of operations (for example multiplication) required for signing and testing.

Table 1 gives the results of comparison of four FSS schemes when the security levels of the receiver and the sender are given by k and σ respectively. In this comparison the first two schemes (first and second columns of the table) are chosen because they have provable security. The first scheme, referred to as vHP in this paper, is the most efficient provably secure scheme. The third scheme, whilst it does not have a complete security proof (although it is not difficult to construct such a proof), is included because it has an explicit proof of forgery by revealing the secret factors of a modulus. Column four corresponds to the scheme proposed in this paper.

We use the same value of σ and k for all the systems, and determine the size of the three length parameters. The hard underlying problems in all four schemes are DL, subgroup DL [23] and/or factorization. This means the same level of receiver's security (given by the value of parameter k) translates into different size primes and moduli. In particular, the security level of a 151-bit subgroup discrete logarithm with basic primes of at least 1881 bits is the same as factorization of a 1881-bit RSA modulus [23].

To find the required size of primes in the vHP scheme, assuming security parameters (k, σ) are given, first $K = \max(k, \sigma)$ is found and then the prime q is chosen such that $|q|_2 \geq K$. The bundling degree in this scheme is q and the value of p is chosen such that $q|p-1$ and $(p-1)/q$ is upper-bounded by a polynomial in K (pp. 237 and 238 of [6]). The size of $|p|_2$ must be chosen according to the standard discrete logarithm problem, which for adequate security must be of at least 1881 bits [23]. However, the size of $|q|_2$ can be chosen as low as 151 bits [23]. Since $|p|_2$ and $|q|_2$ are to some extent independent, we use $\hat{K}$ to denote $|p|_2$.

In our proposed scheme, the bundling degree and hence the security level of the sender is $|q|_2$. The security of the receiver is determined by the difficulty of DL in Z_P^* and factorization of n. Assume $|p|_2 \approx |q|_2 \approx |n|_2/2$. Then we first find N_k which is the modulus size for which factorization has difficulty k. Now since $P \geq n$, DL in Z_P^* will have difficulty k [23] and we choose $K = \max(N_k/2, \sigma)$, $|q|_2 = K \approx |p|_2$ and $P \geq n$. With these choices the sender's and receiver's level of security is at least σ and k respectively. For example for $(k, \sigma) = (151, 151)$, we first find $N_{151} = 1881$ [23] and choose $K = \max(1881/2, 151) = 941$ which results in $|p|_2 \approx |q|_2 \approx 941$ and $|n|_2 \approx |P|_2 \approx 1882$. Since $|P|_2$ can be chosen to be much greater than $|n|_2$, we use $\hat{K}$ to denote $|P|_2$, and so when $|P|_2 \approx |n|_2$ we have $\hat{K} \approx 2K$.

In the factorization scheme of [3], the security level of the sender, σ, satisfies $\tau = \rho + \sigma$ where τ is the bundling degree and 2^ρ is the size of the message space. The security parameter of the receiver, k, is determined by the difficulty of factoring $\bmod n$. Now for a given pair of security parameters (k, σ), the size of $\bmod N_k$ is determined by k, but determining τ requires knowledge of the size of the message space. Assume $\rho = |p|_2 \approx |q|_2 = N_k/2$; this means that $\tau = \sigma + N_k/2$. Now the efficiency parameters of the system can be given as shown in Table 1. In particular the size of secret and public keys are $2(\tau + N_k)$ and $2N_k$ respectively.

In the RSA-based FSS scheme [10], $\tau = |\phi(n)|_2$, and security of the receiver is determined by the difficulty of factoring n. This means that $\tau \approx |n|_2$. To design a system with security parameters (k, σ), first N_k, the modulus size that provides security level k for the receiver, is determined, and then $K = \max(\sigma, |N_k|_2)$. The modulus n is chosen such that $|n|_2 = K$. With this choice, the system provides adequate security for the sender and the receiver.

Table 1 shows that because of the subgroup DL problem, K in the vHP scheme can be as low as 151 bits, while in our scheme it must be at least 941 bits. $\hat{K}$ in the vHP and our scheme must be at least 1881 bits [23].

Table 2 shows that the performance of the vHP and our scheme are nearly the same with respect to the lower bounds given in [3], and in fact both schemes are nearly (i.e. nearly

TABLE 1. Comparison of computation (number of multiplications) and efficiency parameters.

	DL [8]	Factoring [3]	RSA [10]	Our FSS
PK (mult)	$4K$	$2K$	$4K$	$4K$
Sign (mult)	2	K	2	1
Test (mult)	$3K$	$2K+\sigma$	$3K$	$4K$
Length of SK (bits)	$4K$	$4K+2\sigma$	$4K$	$4K$
Length of PK (bits)	$2\hat{K}$	$2K$	$2K$	$2\hat{K}$
Length of a signature (bits)	$2K$	$2K+\sigma$	$4K$	$2K$
Prekey length	$K+3\hat{K}$	$3K$	$3K$	$2(K+\hat{K})$
Length of a message (bits)	K	K	K	$2K$
Min size of K (bits) [23]	151	941	1881	941
Min size of $\hat{K}$ (bits) [23]	1881	n/a	n/a	1881
Underlying hard problem	DL	Factoring	RSA	DL & Factoring

TABLE 2. Comparison between the vHP, our scheme and the optimal lower bound for $N = 1$.

	DL [8]	Our FSS	Lower bound
Length of SK	$4K = 4\sigma$	$4K = 4\sigma$	$2(\sigma-1)$
Length of PK	$2\hat{K}$	$2\hat{K}$	σ
Signature length	$2K = 2\sigma$	$2K = 2\sigma$	$2\sigma-1$

TABLE 3. Comparison of communication efficiency with respect to the message length.

	DL [8]	Factoring [3]	RSA [10]	Our FSS
$\hat{\rho}$	2	>2	4	1

achieving the bounds) optimal with respect to the signature length.

Efficiency with respect to the message length

In practice, we need to consider the relative lengths of the message and the signature. If the length of the signature and the message are denoted by $|y|_2$ and $|x|_2$ respectively, $\hat{\rho} = |y|_2/|x|_2$ is a measure of communication efficiency of the scheme. For example, $\hat{\rho} = 1$ means that to authenticate one bit of information, one bit extra (signature) must be sent over the channel.

Now, in our scheme messages and signatures are both from Z_n, and so $\hat{\rho} = 1$. In the vHP scheme messages and signatures belong to subgroups of size q and $2|q|_2$ respectively. This means that $\hat{\rho} = 2$ and so, to authenticate a one-bit message, a one-bit signature must be used. In the factorization scheme of [3], messages are ρ bits and signatures are $k+\rho+\sigma$ bits. Assuming that $k = \rho$, then $\hat{\rho} > 2$. In the RSA-based FSS in [10], messages belonging to Z_n^* and the signature are of size $4|n|_2$. This means that $\hat{\rho} = 4$.

Table 3 summarizes these results.

Signing long messages

Tables 1 and 3 show that the size of the input to the signature algorithm in the vHP and our scheme are K and $2K$, that is at least 151 and 1882 bits ([23]) respectively. For messages longer than these sizes a hash-then-encrypt [3] method can be used. This has two impacts.

- To prove forgery, rather than showing that the underlying assumption of the scheme is broken, it will be shown that a collision for the collision-resistant hash function used for hashing is found.
- The hash function must be based on a computational assumption. Hash functions with this property, developed in [12, 24], require on average one modular multiplication for one bit of the message, and so drastically reduce the speed of signature generation and testing.

The above points imply that signing a message of length ℓ bits, $151 < \ell < 1882$, requires on average ℓ more modular multiplications compared to our scheme when using vHP.

4.1. Multiple messages

We can extend our scheme to sign more than one message without changing the key using the method in [8].

Suppose $t-1$ messages are to be signed. The signer chooses a secret key $k_0, k_1, \ldots, k_{t-1} \in Z_n^*$, and publishes the corresponding public key

$$(\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_{t-1}) = (\alpha^{k_0}, \alpha^{k_1}, \alpha^{k_2}, \ldots, \alpha^{k_{t-1}})$$

where $\alpha_i \in Z_P^*$, $i = 0, 1, \ldots, t-1$. To sign a message $x \in Z_n$, $\mathcal{S}$ computes

$$y = k_0 + k_1 x + k_2 x^2 + \ldots + k_{t-1} x^{t-1} \bmod n.$$

The signature y passes the verification test if

$$\alpha^y \stackrel{?}{=} \alpha_0 \alpha_1^x \alpha_2^{x^2} \ldots \alpha_{t-1}^{x^{t-1}} \bmod P.$$

Using Theorem 4.4 of [25], it can be proved that the signer has unconditional security after issuing signatures on $t - 1$ different messages.

5. CONCLUSIONS

In this paper, we have proposed a new fail-stop signature scheme that uses two computational assumptions. It uses discrete logarithm and factorization as the underlying assumptions for the recipient's security, and factorization as the underlying assumption for the proof of forgery. If either of the two assumptions is broken, a signature can be easily forged and so the security of the system will be lost.

The proof of forgery is by revealing the non-trivial factors of the modulus and so results in a fast verification process. We have shown that the scheme can be extended for signing multiple messages.

We have compared our scheme with the best known FSS scheme, namely the vHP scheme, and two other schemes which are based on the difficulty of factorization. The comparison clearly shows that our scheme is more efficient than the other factorization-based schemes, and its performance is very similar to the vHP scheme.

We have introduced a new measure of efficiency for FSS that is related to efficient use of communication channel, and shown that with respect to this measure our scheme has better performance than the vHP scheme and the FSS schemes based on factorization. We have shown that compared to the vHP scheme, our scheme is more efficient for signing messages of up to 1881 bits.

REFERENCES

[1] Diffie, W. and Hellman, M. (1976) New directions in cryptography. *IEEE IT*, **22**, 644–654.

[2] Goldwasser, S., Micali, S. and Rivest, R. L. (1988) A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, **17**, 281–308.

[3] Pedersen, T. P. and Pfitzmann, B. (1997) Fail-stop signatures. *SIAM J. Comput.*, **26**, 291–330.

[4] Pfitzmann, B. and Waidner, M. (1990) Formal aspects of fail-stop signatures. *Interner Bericht, Fakultät für Informatik*, **22**. http://www.semper.org/sirene/lit/sirene.lit.html

[5] Waidner, M. and Pfitzmann, B. (1990) The dining cryptographers in the disco: unconditional sender and recipient untraceability with computationally secure serviceability. *Advances in Cryptology—Eurocrypt '89. Lecture Notes in Computer Science*, **434**. Springer-Verlag, Berlin.

[6] Pfitzmann, B. (1996) *Digital signature schemes—general framework and fail-stop signatures. Lecture Notes in Computer Science*, **1100**. Springer-Verlag, Berlin.

[7] van Heijst, E., Pedersen, T. and Pfitzmann, B. (1993) New constructions of fail-stop signatures and lower bounds. *Advances in Cryptology—Crypto '92. Lecture Notes in Computer Science*, **740**, 15–30. Springer-Verlag, Berlin.

[8] van Heyst, E. and Pedersen, T. (1992) How to make efficient fail-stop signatures. *Advances in Cryptology—Eurocrypt '92*, pp. 337–346. Springer-Verlag, Berlin.

[9] Pfitzmann, B. and Waidner, M. (1991) Fail-stop signatures and their application. In *SECURICOM 91, 9th Worldwide Congress on Computer and Communications Security and Protection*, pp. 145–160. Springer-Verlag, Berlin.

[10] Susilo, W., Safavi-Naini, R. and Pieprzyk, J. (1999) RSA-based fail-stop signature schemes. In *Int. Workshop on Security (IWSEC '99)*, pp. 161–166. IEEE Computer Society Press, Aizu-Wakamatsu City, Japan.

[11] Bari'c, N. and Pfitzmann, B. (1997) Collision-free accumulators and fail-stop signature schemes without trees. *Advances in Cryptology—Eurocrypt '97. Lecture Notes in Computer Science*, **1233**, 480–494. Springer-Verlag, Berlin.

[12] Chaum, D., van Heijst, E. and Pfitzmann, B. (1990) Cryptographically strong undeniable signatures, unconditionally secure for the signer. *Interner Bericht, Fakultät für Informatik*, **1**.

[13] Pfitzmann, B. (1994) Fail-stop signatures without trees. *Hildesheimer Informatik-Berichte, Institut für Informatik*, **16**. Institut für Informatik Universität Hildesheim. Technical Report.

[14] Lamport, L. (1979) Constructing digital signatures from a one-way function. *PSRI International CSL-98*. http://research.compaq.com/SRC/personal/lamport/pubs/pubs.html dig-sig.

[15] Pfitzmann, B. (1991) Fail-stop signatures: principles and applications. In *Proc. Compsec '91, 8th World Conf. on Computer Security, Audit and Control*, pp. 125–134. Elsevier, Oxford, UK.

[16] Goldwasser, S., Micali, S. and Rivest, R. L. (1998) A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, **17**, 281–308.

[17] Bleumer, G., Pfitzmann, B. and Waidner, M. (1991) A remark on a signature scheme where forgery can be proved. *Advances in Cryptology—Eurocrypt '90. Lecture Notes in Computer Science*, **437**, 441–445. Springer-Verlag, Berlin.

[18] Brickell, E. F. and McCurley, K. S. (1991) An interactive identification scheme based on discrete logarithms and factoring. *Advances in Cryptology—Eurocrypt '90. Lecture Notes in Computer Science*, **437**, 63–71. Springer-Verlag, Berlin.

[19] Wagstaff, S. S., Jr. (1979) Greatest of the least primes in arithmetic progression having a given modulus. *Math. Comput.*, **33**, 1073–1080.

[20] Stinson, D. R. (1995) *Cryptography: Theory and Practice*. CRC Press, Boca Raton, New York.

[21] Rivest, R. L., Shamir, A. and Adleman, L. (1978) A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, **21**, 120–126.

[22] Camenisch, J. and Michels, M. (1999) Proving in zero-knowledge that a number is the product of two safe primes. *Advances in Cryptology—Eurocrypt '99. Lecture Notes in Computer Science*, **1592**, 107–122. Springer-Verlag, Berlin.

[23] Lenstra, A. K. and Verheul, E. R. (1999) Selecting cryptographic key sizes. *http://www.cryptosavvy.com/*. Extended abstract in Commercial applications *Price Waterhouse Coopers, CCE Quarterly Journals*, **3**, 3–9.

[24] Damgård, I. B. (1988) Collision free hash functions and public key signature scheme. *Advances in Cryptology—Eurocrypt '87. Lecture Notes in Computer Science*, **304**, 203–216. Springer-Verlag, Berlin.

[25] Pedersen, T. (1991) Non-interactive and information-theoretic secure variable secret sharing. *Advances in Cryptology—Crypto '91*, pp. 129–140. Springer-Verlag, Berlin.