An Online/Offline Signature Scheme Based on the Strong RSA Assumption

Ping Yu Stephen R. Tate
Department of Computer Science and Engineering
University of North Texas
Denton, TX 76203
pingyu@unt.edu srt@cs.unt.edu

Abstract

We propose an efficient digital signature scheme, which is proved secure under the strong RSA assumption, and can operate in an online/offline manner, doing most of its work in the offline precomputation phase. The online phase, which is performed after the message to be signed is known, is very efficient, requiring only a single modular multiplication. Online/offline signatures can be used in settings in which signatures need to be produced quickly either when there is a large volume of requests or if the device performing the signature is not computationally powerful (such as a mobile device). Our scheme can be seen as an online/offline extension of the traditional signature scheme of Gennaro, Halevi, and Rabin (the GHR signature scheme) which did not operate in this two-phase manner, and required significant computation after the message was known. In contrast to another online/offline extension of the GHR scheme, our new scheme avoids the use of trapdoor hash/commitment primitives, allowing the use of a traditional hash function, improving the efficiency of the offline phase of the algorithm.

Keywords: Digital Signature, Strong RSA Assumption, Random Oracle, Online/Offline Signing, Suitable Hash Function.

1 Introduction

Consider a typical scenario in a distributed system: A server experiences "bursty" traffic so that at times it needs to authenticate itself simultaneously to a large number of client applications running on remote client machines, while at other times there are many idle CPU cycles. For example, this server could be a stock broker's server and the client wants to confirm that the remote server is not a spoofed server. The standard way of achieving this goal is with a digital signature scheme: the client generates a random message, and asks the server to produce a digital signature on this message using the server's private key. Then the client can authenticate the server using the server's public key to check if the signature is valid on its random challenge message. On a stock broker's server there would be a steady stream of requests during the trading day, but there might be times — such as immediately after financial updates or news releases — when the number of requests is extraordinarily high. And after hours there is significantly less traffic. Since cryptographic computation normally is time consuming, with a high volume of simultaneous authentication requests the server could be overwhelmed, and not be able to produce signatures in an acceptable amount of time.

Consider also a different scenario in which a mobile device with limited computing capability needs to authenticate itself before accessing certain network resources. A remote server may challenge a device with a random message, asking the device to produce a signature for its message. Since a mobile device has limited computing capability, it might have some difficulty generating the required signature quickly.

In the above two scenarios, it is highly desirable that a mechanism should be deployed to expedite the authentication process. Such a mechanism does exist for digital signature schemes, and is known as online/offline signing. In online/offline signing, a majority of the computation related to signature generation can be completed even before a message appears. Therefore, most computation can be done offline during idle time. When a message comes, only a very simple calculation is needed to complete signature generation. This mechanism is very attractive for the above scenarios. We will present such an online/offline digital signature scheme in this paper.

1.1 Background

The digital signature concept is a fundamental cryptographic primitive in modern cryptography. In such schemes, a user prepares a keypair which includes a signing key and

a verification key. The signing key is kept secret by the user while the verification key is public for potential verifiers. The user generates a string by signing a message using his signing key. This string is called the user's signature on this particular message. Later a verifier can check the validity of a signature on a message using the user's verification key.

The idea of digital signatures was first proposed by Diffie and Hellman [5]. Since then, numerous constructions have been proposed in the literature based on different security assumptions. Many schemes are based on the well-known RSA assumption and a variant known as the strong RSA assumption, including PSS [1] and the Cramer-Shoup scheme [4]. Other schemes are based on variants of the discrete logarithm or computational/decisional Diffie-Hellman assumption, including Schnorr signatures [12], ElGamal signatures [6], and many others.

As a fundamental cryptographic primitive, it is important to clarify the exact requirements for a secure digital signature scheme. In 1988, Goldwasser *et al.* defined a security notion for signature schemes which is called existential unforgeability under adaptive chosen message attacks [9]. Since then, this notion has been widely used to judge whether a digital signature scheme is strong enough to be deployed in a real application. Many digital signature schemes have been proved secure under this requirement in the random oracle model (e.g., Schnorr, ElGamal, PSS). The random oracle model abstracts a cryptographic hash function as a random function. Canetti *et al.* constructed a scheme that can be proved secure in the random oracle model, while any real implementation will result in an insecure construction [3]. Therefore, security proofs in the random oracle model do not necessarily imply security in the standard model, so a proof of security in the random oracle model can only be treated as a heuristic argument that a scheme is secure.

In 1999, Gennaro, Halevi and Rabin proposed a practical digital signature scheme in the "hash-and-sign" paradigm that they proved secure under adaptive chosen message attacks without requiring random oracles [8]. In this paper, we refer to their scheme as the GHR scheme. The GHR scheme requires a hash function that meets certain requirements, but these requirements are much more realistic than assuming a random oracle. While meeting these requirements in a provable way leads to some complex constructions, it is quite reasonable to think that widely-used efficient hash functions (like SHA-1 or SHA-256) satisfy the necessary requirements. While formally proving this is beyond current cryptographic understanding, the use of these standard hash functions in GHR gives a very efficient algorithm.

The notion of online/offline signatures was first introduced by Even *et al.* in 1989 [7], and they proposed a generic method to convert any signature scheme into an online/offline one. In fact, many signature schemes based on the discrete logarithm assumption naturally have online/offline versions without any further effort, while signature schemes based on the strong RSA assumption generally do not have such characteristic. However, the method proposed by Even *et al.* is not efficient and practical. In 2001, Shamir and Tauman proposed another generic method to achieve online/offline signing [13]. Their method is based on a new type of hash function called a trapdoor hash function, which is proposed by Krawczyk and Rabin [10], and allows the use of a "hash-sign-switch" paradigm. This construction was further improved by Kurosawa and Schmidt-Samoa who weakened the requirements needed for the cryptographic primitives in the generic online/offline construction [11].

1.2 Our Results

We present a signature scheme that can be viewed as an online/offline extension of the GHR scheme. As a direct online/offline design, we maintain the nice feature from GHR that standard efficient hash functions are likely to give the necessary security. This is in contrast to generic constructions such as that of Shamir and Tauman [13] or Kurosawa and Schmidt-Samoa [11] that require trapdoor hash or trapdoor commitment primitives, which complicate the scheme by introducing additional sets of keys and requiring additional complex operations (similar to public key cryptographic operations) for setting up the hash function in the offline phase. As a consequence, our system is easier to manage (with fewer keys) and is more efficient in the offline phase.

Our scheme does have two disadvantages, however. First, while the offline phase is more efficient, the online phase is slightly slower than the Kurosawa/Schmidt-Samoa extension, requiring a full modular multiplication as opposed to a shift and a modular reduction. Even so, our online phase is still very efficient (a single modular multiplication), so we feel this is not a serious issue. Second, our signature scheme can fail, meaning it produces a signature that will not be verified. While this is serious, we prove that this failure happens only with negligible probability. Many existing systems have the property that security is weak with negligible probability (for example, finding a prime factor of an RSA modulus by chance), but are used and relied upon anyway, and we feel such reasoning applies to our new scheme as well. We also provide a strong way to avoid this: when sufficient CPU cycles are available, a validity test can be performed by the signer which will allow her to always produce a valid signature. While this extra step negates some of the efficient online phase improvements, in a bursty traffic situation as described earlier it is possible to serve most requests with certainty, and

those during high volume time with a negligible probability of failure.

The rest of the paper is organized as follows. Section 2 reviews some cryptographic notations and definitions. Our online/offline signature scheme is introduced in Section 3, and its security properties in the random oracle model are discussed in Section 4. In Section 5 we discuss the properties needed in using a real hash function as opposed to a random oracle, and discuss the resulting security issues. Finally, we give the conclusions in Section 6.

2 Preliminaries

This section reviews some notations and definitions which are used throughout the paper.

Definition 1 (Special RSA Modulus) *An RSA modulus $n = pq$ is called special if $p = 2p' + 1$ and $q = 2q' + 1$ where p' and q' also are prime numbers.*

Definition 2 (Quadratic Residue Group QR_n) *Let Z_n^* be the multiplicative group modulo n, which contains all positive integers less than n and relatively prime to n. An element $x \in Z_n^*$ is called a* quadratic residue *if there exists an $a \in Z_n^*$ such that $a^2 \equiv x \mod n$. The set of all quadratic residues of Z_n^* forms a cyclic subgroup of Z_n^*, which we denote by QR_n. If n is the product of two distinct primes, then $|QR_n| = \frac{1}{4}|Z_n^*|$.*

A hash function is a function mapping arbitrary strings of finite length to binary strings of fixed length. For cryptographic purpose, we would like a hash function should satisfy the properties such as strong collision resistance, weak collision resistance,etc. Our scheme uses another property for a hash function called "division intractability," which was introduced by Gennaro *et al.* [8]. Informally, a hash function is division intractable if it is infeasible to find distinct inputs for this hash function such that the hash value of one input divides the product of hash values of all other inputs.

We use the notion of a suitable hash function $\mathcal{H}$ as defined by Gennaro, Halevi, and Rabin, which we outline below — more detailed discussion can be found in the GHR paper [8].

Definition 3 (Suitable Hash Function) *A suitable hash function has the following properties:*

- *$\mathcal{H}$ is division intractable.*
- *The distributions $h(R; M_1)$, $h(R; M_2)$ induced by the random choice of R, are statistically close, for every $h \in \mathcal{H}$ and every two messages M_1, M_2.*
- *The strong RSA assumption holds in a model where there exists an oracle that on input (h, M, e), returns a random R with $h(R; M) = e$.*

Now we introduce the strongest notion of a secure signature scheme, existential unforgeability under adaptive chosen message attacks, which was proposed by Goldwasser, Micali and Rivest [9]. The definition we give here is due to Gennaro *et al.* [8].

Definition 4 (Secure Signatures [8]) *A signature scheme $S = \langle Gen, Sig, Ver \rangle$ is existentially unforgeable under an adaptive chosen message attack if it is infeasible for a forger who only knows the public key to produce a valid (message, signature) pair, even after obtaining polynomially many signatures on messages of its choice from the signer. Formally, for every probabilistic polynomial time forger algorithm $\mathcal{F}$, there exists a negligible function $negl()$ such that*

$$Pr\left[\begin{array}{l} \langle pk, sk\rangle \leftarrow Gen(1^k); \\ for\ i = 1 \ldots n \\ \quad M_i \leftarrow \mathcal{F}(pk, M_1, \sigma_1, \ldots, M_{i-1}, \sigma_{i-1}); \\ \quad \sigma_i \leftarrow Sig(sk, M_i); \\ \langle M, \sigma\rangle \leftarrow \mathcal{F}(pk, M_1, \sigma_1, \ldots, M_n, \sigma_n), \\ M \neq M_i\ for\ i = 1 \ldots n,\ and \\ Ver(pk, M, \sigma) = accept. \end{array}\right] = negl(k).$$

3 The Digital Signature Scheme

Table 1 shows our online/offline digital signature scheme, which includes the system setup, sign algorithm and verify algorithm. The system is parametrized by security parameter λ and a hash function length k, where we require that k is polynomial in λ, and should be chosen so that the failure probability in Lemma 5 is acceptably small. For example, a reasonable system would be produced with $\lambda = 512$ (so the modulus n is 1024 bits) and $k = 1024$.

The online phase of the sign algorithm includes an optional test step — if time can be taken to perform this step, the sign algorithm will never fail, but at the cost of a moderately expensive GCD computation. On the other hand, as we show in Lemma 5, the probability of this test failing is negligible, so the test can be omitted if a negligible failure probability is acceptable. If the test is omitted, the online phase only requires a single modular multiplication.

In a real application, it is desirable to have a pool of (s_i, X_i) pairs available when needed. For instance, in the first scenario in the introduction, the server might need a large pool to complete a large burst of authentication requests, so b could be large (e.g., over 1000). On the other hand, for a mobile device, b can be set reasonable small — a pool size of 5 or less should be enough for most situations. During idle time, the device produces new pairs to keep the pool full.

System Setup — For security parameter λ and hash function length k (see text for requirements)

1. n: Pick two λ-bit safe primes p and q (so that $p = 2p' + 1$, and $q = 2q' + 1$, where p' and q' are also prime), and let $n = pq$.
2. g: Select g as a random generator of QR_n.
3. $\mathcal{H}$: Choose a division intractable hash function $\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^k$.

The public key is $(n, g, \mathcal{H})$, and the private key is (p, q).

Sign Algorithm

Offline Phase: For pool size b, Alice picks b random values $s_i \in_R [0, p'q')$, for $i = 1, \cdots, b$. For each s_i, Alice computes

$$X_i = g^{s_i} \mod n.$$

Thus, Alice prepares a pool of size b (s_i, X_i) pairs in idle time.

Online Phase: For a message m, Alice computes $\mathcal{H}(m)$, then uses the next unused (s_i, X_i) pair to compute

$$r = s_i \times \mathcal{H}(m) \mod p'q'.$$

Optional test step: Test if $GCD(\mathcal{H}(m), r) \leq 2^{2\sqrt{k}}$ — if so, the signature is ready; otherwise, Alice will continue by trying other (s_i, X_i) pairs.

The signature is (X_i, r).

Verify Algorithm

For Alice's signature (X, r) on message m, Bob verifies that $GCD(\mathcal{H}(m), r) \leq 2^{2\sqrt{k}}$, and $X^{\mathcal{H}(m)} \equiv g^r \mod n$.

Table 1. Our online/offline signature scheme

4 Security in the Random Oracle Model

The following security proof is derived from the proof given in the original Gennaro, Halevi, and Rabin paper [8]. The security of our scheme relies on the following security assumption which is widely accepted in the cryptography literature. Due to the page limitation, the proofs for lemmas have been moved to the full version of this paper [14].

Assumption 1 (Strong RSA Assumption) *Let n be an RSA modulus. The* Flexible RSA Problem *is the problem of taking a random element $u \in Z_n^*$ and finding a pair (v, e) such that $e > 1$ and $v^e = u \mod n$. The* Strong RSA Assumption *says that no probabilistic polynomial time algorithm can solve the flexible RSA problem with non-negligible probability.*

First, we introduce a lemma due to Camenisch and Lysyanskaya [2] which will be used for the proof of our signature scheme.

Lemma 1 *Let n be a special RSA modulus. Given values $u, v \in QR_n$ and $x, y \in Z$, $GCD(x, y) < x$ such that $v^x \equiv u^y \mod n$. Values $z, w > 1$ such that $z^w \equiv u \mod n$ can be computed efficiently.*

Next, we introduce a lemma which addresses the smoothness of a random integer. The proof for this lemma can be found as part of the proof of Lemma 6 in [8].

Lemma 2 *Let e be a random k-bit integer. The probability of e being $2^{2\sqrt{k}}$-smooth (all e's prime factors are no larger than $2^{2\sqrt{k}}$) is no larger than $2^{-2\sqrt{k}}$.*

The following lemma is used directly in the security proof for our digital signature scheme — note that the condition on v is met for sufficiently large k whenever v is polynomial in k.

Lemma 3 *Let $e_1, e_2, \cdots, e_v$ be random k-bit integers, where $v \leq 2^{0.5\sqrt{k}}$. Let j be a randomly chosen index from*

$[1, v]$, *and define* $E = (\prod_{i=1}^{v} e_i)/e_j$. *If* r_j *is an integer such that* $GCD(e_j, r_j) \leq 2^{2\sqrt{k}}$, *then the probability that* e_j *divides* Er_j *is less than* $2^{-\sqrt{k}}$.

Our security proof uses the standard technique of simulating the signature oracle for a forgery algorithm. Unfortunately, we cannot perfectly simulate the signature oracle, because the distribution of the r values produced by the sign algorithm depends on the unknown value $p'q'$. However, we can simulate something very close to this distribution — in the main security theorem we use the following lemma to show that our "close" distribution is in fact close enough to establish the security of our signature scheme.

Let $Pr_D(x)$ denote the probability of x in distribution D. Then the distance between two distributions D_1 and D_2 is

$$\mathsf{dist}(D_1, D_2) = \frac{1}{2} \sum_x |Pr_{D_1}(x) - Pr_{D_2}(x)|,$$

and note that for any two distributions $\mathsf{dist}(D_1, D_2) \leq 1$. Two distributions D_1 and D_2 are *statistically indistinguishable* if $\mathsf{dist}(D_1, D_2)$ is negligible.

Lemma 4 *Let* p' *and* q' *be as defined in Table 1, so in particular* p' *and* q' *are* $\lambda - 1$ *bit prime numbers. Consider the following two distributions on pairs* (r, e)*:*

- *Distribution* D_1 *is obtained by selecting* e *uniformly from the set of* k*-bit integers, and* r *is computed as* $r = s \times e \bmod p'q'$*, where* s *is chosen uniformly from the set* $\{0, \cdots, p'q' - 1\}$.
- *Distribution* D_2 *is obtained by selecting* e *uniformly from the set of* k*-bit integers, and* r *is uniformly selected from the set* $\{0, \cdots, \frac{n-1}{4} - 1\}$.

$\mathsf{dist}(D_1, D_2) < 2^{-\lambda+3}$, *so distributions* D_1 *and* D_2 *are statistically indistinguishable.*

Lemma 5 *Let* (e, r) *be a pair drawn from* D_1*, where* $k \geq 4$ *is polynomial in* λ*. Then the probability that* $GCD(e, r) > 2^{2\sqrt{k}}$ *is at most* $\frac{k}{2^{2\sqrt{k}}} + \nu(k)$*, where* $\nu(k)$ *is negligible in* k.

Now we prove our signature scheme is secure under the strong RSA assumption when the hash function $\mathcal{H}$ is replaced by a random oracle.

Theorem 1 *In the random oracle model, the above signature scheme is existentially unforgeable under an adaptive chosen message attack, assuming the strong RSA assumption.*

Proof: Let F be a forger algorithm. Under the random oracle model, F always queries the random oracle about a message m before it either asks the signature oracle to sign this message, or outputs (m, x, r) as a potential forgery. Let v be some polynomial upper bound on the number of queries that F makes to the random oracle.

We now show an efficient algorithm A, that uses F as a subroutine, such that if F has probability ϵ of forging a signature, then A has probability $\epsilon' \approx \epsilon/v$ of solving the flexible RSA problem.

A is given a special RSA modulus n and a $t \in_R QR_n$, and its goal is to find a pair (v, e) such that $e > 1$ and $v^e \equiv t \mod n$.

First, A prepares answers for the random oracle queries that F will ask by picking v random k-bit integers $e_1, \cdots, e_v$ and a random $j \in_R [1, v]$. A is betting on the chance that F will use its j'th oracle query to generate the forgery.

Next, A prepares answers for signature queries that F will ask. A computes $E = (\prod_{i=1}^{v} e_i)/e_j$. If e_j divides E, then A outputs "failure" and halts. Otherwise, it sets $g = t^E \mod n$, and initializes the forger F, giving it the public key (n, g).

A then runs the forger algorithm F, answering oracle queries with the help of a function $\textsc{Sample}_D()$ which gets a random sample from some distribution D, which we will describe at the end of the proof. Specifically, oracle queries are answered as follows:

- Random oracle queries for $m_1 \ldots m_v$ are answered by setting $h(m_i) = e_i$ for each $i \in [1, v]$.
- Signature oracle query for message m_i, for $i \neq j$, are answered with (m_i, x_i, r_i) where $r_i = \textsc{Sample}_D()$ and $x_i = t^{Er_i/e_i} \bmod n$.

If F queries the signature oracle for message m_j, or halts with an output other than (m_j, x_j, r_j) for which $x_j^{e_j} \equiv g^{r_j} \mod n$ and $GCD(e_j, r_j) \leq 2^{2\sqrt{k}}$, then A outputs "failure" and halts. Otherwise we have a valid forgery with

$$x_j^{e_j} \equiv g^{r_j} \equiv t^{Er_j} \mod n.$$

By Lemma 3, e_j divides Er_j with a negligible probability, and so with overwhelming probability $GCD(e_j, Er_j) < e_j$, and we can apply Lemma 1 to find values z and e such that $z^e \equiv t \mod n$. Therefore, if A does not output "failure" then with overwhelming probability it outputs a valid solution to this instance of the flexible RSA problem.

To bound the probability that A outputs "failure," we need to examine the probability distribution D used in answering queries to the signature oracle. If we could sample according to distribution D_1, as defined in Lemma 4, then the distribution of signature responses is identical to that produced by an actual signer, so the probability of generating a valid forgery for a specific message is ϵ, and combined with the probability that we correctly picked the correct j

for the forgery query the overall success probability is ϵ/v. However, we cannot sample from distribution D_1 without knowing the value $p'q'$, which is not available to us in A's simulation of the signature oracle.

Instead, we use distribution D_2 from Lemma 4, which is just the uniform distribution on $\{0, \cdots, \frac{n-1}{4} - 1\}$ and so can be efficiently sampled. The success probability of the forgery algorithm is only changed by a negligible amount, since we showed in Lemma 4 that these two distributions are statistically indistinguishable (if the change in success probability was more than a negligible amount, we could use this very simulation as a distinguisher between D_1 and D_2). In other words, using D_2 for our distribution D, the success probability of the forgery algorithm is at least $(\epsilon - \eta(\lambda))/v$, where $\eta(\lambda)$ is some negligible function in λ.

A can also output "failure" if, in selecting the random oracle outputs $e_1, \cdots, e_v$, we have Er_j divisible by e_j. However, Lemma 3 established that this is another negligible probability, and so the overall probability of successfully solving the flexible oracle problem is $\approx \epsilon/v$. □

5 Replacing the Random Oracle

In this section, we briefly discuss using a real hash function $\mathcal{H}$ instead of a random oracle. As in the GHR algorithm, we don't need a completely random function for $\mathcal{H}$, but can instead use a function which satisfies less stringent requirements and is thus realizable.

Specifically, we require that the hash function be "suitable," according to Definition 3. The salient properties of the GHR scheme which allow for the use of a suitable hash function carry over to our modification as well, resulting in the following theorem.

Theorem 2 *If $\mathcal{H}$ is suitable, then the construction from Section 3 is existentially unforgeable under an adaptive chosen message attack, assuming the strong RSA assumption.*

Due to the similarity between our scheme and the GHR scheme, the proof for the above theorem is similar to that for the GHR scheme [8], and is in the full version of our paper [14].

6 Conclusions

In this paper we have presented a new efficient digital signature scheme, which is proved secure under the strong RSA assumption. Our construction works in a two-phase offline/online model, so after some offline precomputation that is independent of the message being signed, the online phase is highly efficient (just a single modular multiplication). Furthermore, while other online/offline schemes require random oracles or complex hash function constructions such as trapdoor hash functions, our scheme is secure under the less demanding requirement of a "suitable hash function" (as defined by Gennaro, Halevi, and Rabin [8]). This both simplifies the overall system and makes the offline phase more efficient, while the online phase is still very efficient.

References

[1] M. Bellare and P. Rogaway. The exact security of digital signatures - how to sign with rsa and rabin. In *Advances in Cryptology–Eurocypt'96 LNCS 1070*, pages 399–416. Springer-Verlag, 1996.

[2] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *Third Conference on Security in Communication Networks 02 — SCN'02, LNCS 2576*, pages 268–289. Springer-Verlag, 2002.

[3] R. Canetti, O. Goldreich, and S. Halevi. The random oracle model, revisited. In *30th Annual ACM Symposium on Theory of Computing*, pages 209–218, 1998.

[4] R. Cramer and V. Shoup. Signatures schemes based on the strong rsa assumption. In *ACM Transaction on Information and System Security*, volume 3, pages 161–185, 2000.

[5] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 11:644–654, Nov. 1976.

[6] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology — Crypto*, pages 10–18, 1984.

[7] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. In *Advances in Cryptology: Crypto'89*, pages 263–277, 1990.

[8] R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In *Advances in Cryptology–Eurocypt'99*, pages 123–139. Springer-Verlag, 1999.

[9] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17:281–308, 1988.

[10] H. Krawczyk and T. Rabin. Chameleon signatures. In *Symposium on Network and Distributed Systems Security(NDSS'00)*, pages 143–154. Internet Society, 2000.

[11] K. Kurosawa and K. Schmidt-Samoa. New online/offline signature schemes without random oracles. In *M. Yung et al. PKC 2006, LNCS 3958*, pages 330–346. Springer-Verlag, 2006.

[12] C. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[13] A. Shamir and Y. Tauman. Improved online/offline signature schemems. In *Advances in Cryptology: Crypto'01, LNCS 2139*, pages 355–367. Springer-Verlag, 2001.

[14] P. Yu and S. R. Tate. An online/offline signature scheme based on the strong rsa assumption. *Cops Lab Technical Report, 2007. Available from http://cops.csci.unt.edu/publications/.*