Constructing Optimistic Fair Exchange Protocols from Committed Signatures

Huafei Zhu

Department of Information Science and Electronics Engineering, ZheJiang University,
YuQuan Campus, HangZhou, 310027, PR. China
E-mail: zhuhf@zju.edu.cn

Abstract. In PODC 2003, Park et al. [32] first introduce a connection between fair exchange and sequential two-party multi-signature scheme and provide a novel method of constructing fair exchange protocol by distributing the computation of RSA signature. This approach avoids the design of verifiable encryption scheme at the expense of having co-signer store a piece of prime signer's secret key. Dodis and Reyzin [20] showed that the protocol in [32] is totally breakable in the registration phase, then presented a remedy scheme which is provably secure in the random oracle model, by utilizing Boldyreva non-interactive two-party multi-signature scheme [8]. Security in the random oracle model does not imply security in the real world. In this paper, we provide the first two efficient committed signatures which are provably secure in the standard complexity model from strong RSA assumption. Then we construct efficient optimistic fair exchange protocols from those new primitives.

Keywords: Committed signature scheme, optimistic fair exchange protocols, strong RSA assumption

1 Introduction

An important issue in electronic commerce is how to exchange electronic data between two potentially distributed parties in an efficient and fair manner. Intuitively, fairness allows two parties to exchange items in a fair way, so that either each party gets other's item, or neither party does. Examples of such exchanges include signing of electronic contracts, certificated e-mail delivery and fair purchase of electronic goods over communication network. In such instances, ensuring fairness is crucial if the participants are to be protected from fraud. The problem of fair exchange has a rich history due to its fundamental importance. In the following, we only briefly mention the body of research most relevant to our results, and refer the reader to [4], [24] and [32] for further references.

Related works Early work on fair exchange of secrets/signatures, focused on the gradual release of secrets to obtain simultaneity and fairness [11], [21], [25] and [18]. The idea is that if each party alternately release a small portion of the secret, then neither party has a considerable advantage over the other. Unfortunately, such a solution has several drawbacks. Apart from being expensive in

terms of computation and communication, it has the problem in real situations of uncertain termination.

An alternative approach to achieve fairness makes use of a trusted third party (TTP). A TTP is essentially a judge that can be called in to handle disputes between the participants. The TTP can be on-line in the sense of mediating after every exchange as in [17] and [23], or off-line, meaning that it only gets involved when something goes wrong (e.g., a participant attempts to cheat, or simply crashes, or the communication delays between the participants are intolerably high, etc.). The latter approach has been called optimistic [2].

Fair exchange protocols using verifiable encryption was proposed by Atenies [1] and Bao et al. [7] independently. These protocols apply ad-hoc techniques to crate the fairness primitive via a specific encryption scheme that confirms to a given signature type. Unfortunately, the schemes proposed in [1] and [6] lack any formal security analysis, and consequently, one of the schemes proposed in [7] was shown to be insecure in [9] and [1]. In [3] and [4], Asokan et al. propose an optimistic that uses a cryptographic primitive denoted as verifiable escrow to produce the fairness. This is the first protocol immune to off-line attack.

In PODC 2003, Park et al. [32] first introduce a connection between fair exchange and sequential two-party multi-signature scheme and provide a novel method of constructing fair exchange protocol by distributing the computation of RSA signature. This approach avoids the design of verifiable encryption scheme at the expense of having co-signer store a piece of prime signer's secret key. Dodis and Reyzin [20] showed that the protocol in [32] is totally breakable in the registration phase, then presented a remedy scheme which is provably secure in the random oracle model, by utilizing Boldyreva non-interactive two-party multi-signature scheme [8].

Our Results Security in the random oracle model does not imply security in the real world. In this paper, we provide the first two efficient committed signatures, which are provably secure in the standard complexity model from strong RSA assumption. Then we construct efficient optimistic fair exchange protocols from those new primitives.

The rest of paper is organized as follows: in section 2, we formalize the security definition of committed signatures, and two committed signatures are presented in the section 3. The optimistic fair exchange protocols derived from these committed signatures are described in section 4.

2 Committed signatures

Dodis and Reyzin [20] introduce a unified model for non-interactive fair exchange protocols, which results in a new primitive called committed signatures. Committed signatures generalize non-interactive verifiably encrypted signatures and multi-signature schemes, both of which are sufficient for fair exchange.

2.1 Notions and definitions

Definition 1 A committed signature involves a primary singer Alice, a verifier Bob and a co-singer (or arbitrator) Charlie, and is given by the following efficient procedures:

Key generator: This is interactive protocol between a primary signer and a co-signer, by the end of the which either one of the parties aborts, or the primary signer learns her secret signing key Sk, the co-signer learns his secret key ASK, and both parties agree on the primary signer's public key PK and partial verification key APK;

Fully signing algorithm Sig and its correspondent verification algorithm Ver: These are conventional signing and verification algorithms. $Sig(m, SK)$ run by the primary signer, outputs a signature σ on m, while $Ver(m, \sigma, PK)$ run by any verifier, outputs 1 (accept) or 0 (reject);

Partially signing algor$PSig$ and its correspondent verification algorithm $PVer$: These are partial signing and verification algorithms, which are just like ordinary signing and verification algorithms, except they can depend on the public arbitration key APK. $PSig(m, SK, PK, APK)$, run by the primary signer, outputs a partial signature σ', while $PVer(m, \sigma' PK, APK)$, run by any verifier, outputs 1 (accept) or 0 (reject);

Resolution algorithm Res: This is a resolution algorithm run by the co-singer in case the primary singer refuses to open her signature σ to the verifier, who in turn possesses a valid partial signature σ' on m and a proof that he fulfilled his obligation to the primary signer. In this case, $Res(m, \sigma', ASK, PK)$ should output a valid full signature of m.

Correctness of committed signatures states that:

- $Ver(m, Sig(m, SK), PK) = 1$;
- $PVer(m, PSig(m, SK, PK, APK), PK, APK) = 1$;
- and $Ver(m, Res(PSig(m, SK, PK, APK), ASK, APK, PK), PK) = 1$.

2.2 Security of committed signatures

The security of committed signature scheme consists of ensuring three aspects: security against a primary signer Alice, security against a verifier Bob, and security against a co-singer/abitrator Charlie.

Security against a primary signer Intuitively, a primary signer Alice should not provide a partial signature which is valid both form the point views of a verifier and a co-signer but which will not be opened into the primary signer's full signature by the honest co-signer. More formally:

Let P be an oracle simulating the partial signing procedure $PSig$, and R be an oracle simulating the resolution procedure Res. Let k be system security parameter. We require that any probabilistic polynomial time Adv succeeds with at most negligible probability in the following experiment.

Experiment 1 (security against primary signer):

1.1: Key generation: $(SK^*, PK, ASK, APK) \leftarrow Setup^*(1^k)$, where $Setup^*$ denotes the run of $Setup$ with the dishonest primary signer by the adversary, and SK^* denotes the adversary's states.

1.2: R oracle query: In this phase, for each adaptively chosen message m_j, the adversary computes its partial signature $\sigma_j{}'$ for m_j. Finally the adversary forward $\sigma_j{}'$ to the oracle R to obtain the full signature σ_j of message m_j, where $1 \leq j \leq p(k)$, and $p(\cdot)$ is a polynomial. At the end, the adversary produces a message-full signature pair (m, σ), i.e., $(m, \sigma') \leftarrow Adv^R(SK^*, PK, APK)$, $\sigma \leftarrow Adv(m, \sigma', ASK, APK, PK)$, where $m \neq m_j$, $1 \leq j \leq p(k)$.

1.3. Success of $Adv := [PVer(m, \sigma', APK, PK) = 1 \wedge Ver(m, \sigma, PK) = 0]$.

Definition 2 A committed signature scheme is secure against primary signer attack, if for any probabilistic polynomial time adversary Adv associated with Resolution oracle, succeeds with at most negligible probability, where the probability takes over coin tosses in $Setup(\cdot)$, $PSig(\cdot)$ and $R(\cdot)$.

Security against verifier We consider the following scenario: suppose a primary signer Alice and a verifier Bob are trying to exchange signature in a fair way. Alice wants to commit to the transaction by providing her partial signature. Of course, it should be computational infeasible for Bob to compute the full signature from the partial signature. More formally, we require that any probabilistic polynomial time adversary Adv succeeds with at most negligible probability in the following experiment:

Experiment 2 (security against verifier):

2.1 Key generation: $(SK, PK, ASK, APK) \leftarrow Setup(1^k)$, where $Setup$ is run by the honest primary signer and honest co-signer. Adversary Adv are admitted to make queries to the two orales P and R.

2.2 P and R oracle query: For each adaptively chosen message m_j, the adversary obtains the partial signature $\sigma_j{}'$ for m_j from the oracle P. Then the adversary forward $\sigma_j{}'$ to the oracle R to obtain the full signature σ_j of message m_j, where $1 \leq j \leq p(k)$, and $p(\cdot)$ is a polynomial. At the end, the adversary produces a message-full signature pair $(m, \sigma) \leftarrow Adv^{P,R}(PK, APK)$.

2.3 Success of adversary $Adv := [Ver(m, \sigma, PK) = 1 \wedge m \notin Query(Adv, R)]$, where $Query(Adv, R)$ is the set of valid queries Adv asked to the resolution oracle R, i.e., (m, σ') such that $PVer(m, \sigma') = 1$.

Definition 3 A committed signature scheme is secure against verifier attack, if for any probabilistic polynomial time adversary Adv associated with partial signing oracle P and the resolution oracle R, succeeds with at most negligible probability, where the probability takes over coin tosses in $Setup(\cdot)$, $P(\cdot)$ and $R(\cdot)$.

Security against co-signer/arbitrator This property is crucial. Even though the co-signer (arbitrator) is semi-trusted, the primary signer does not want this co-signer to produce a valid signature which the primary signer did

not intend on producing. To achieve this goal, we require that any probabilistic polynomial time adversary Adv associated with partial signing oracle P, succeeds with at most negligible probability in the following experiment:

Experiment 3 (security against co-signer/arbitrator):

3.1 Key generation: $(SK, PK, ASK^*, APK) \leftarrow Setup^*(1^k)$, where $Setup^*(1^k)$ is run by the dishonest co-signer. Adversary Adv are admitted to make queries to the partial signing orale P.

3.2 P oracle query: For each adaptively chosen message m_j, the adversary obtains the partial signature $\sigma_j{}'$ for m_j from the oracle P, where $1 \leq j \leq p(k)$, and $p(\cdot)$ is a polynomial. At the end, the adversary produces a message-full signature pair (m, σ), i.e., $(m, \sigma) \leftarrow Adv^P(ASK^*, PK, APK)$.

3.3 Success of adversary $Adv := [Ver(m, \sigma, PK) = 1 \wedge m \notin Query(Adv, P)]$, where $Query(Adv, P)$ is the set of valid queries Adv asked to the partial oracle P, i.e., (m, σ') such that $PVer(m, \sigma') = 1$.

Definition 4 A committed signature scheme is secure against co-signer attack, if for any probabilistic polynomial time adversary Adv associated with partial signing oracle P, succeeds with at most negligible probability, where the probability takes over coin tosses in $Setup(\cdot)$, $P(\cdot)$.

Definition 5 A committed signature scheme is secure if it is secure against primary signer attack, verifier attack and co-signer attack.

3 Committed signatures based on strong RSA assumption

The existence of committed signature is obvious since two arbitrary signature schemes with keys (pk_1, sk_1), (pk_2, sk_2), and let $PK = (pk_1, pk_2)$, $SK = (sk_1, sk_2)$ and $\sigma = (\sigma_1, \sigma_2)$ is sufficient to build a secure committed signature even in the standard complexity model. The rest works are to construct efficient yet secure committed signatures. In this section, In this section, we are able to provide two types of committed signatures: one is from a pair of independent ordinary signatures and the other is from single signature.

3.1 Constructing committed signature from a pair of independent signatures

We utilize Zhu's signature as primary building block to construct committed signature scheme. We remark that the use of Zhu's signature is not essential in the first scheme. The original Cramer-Shoup's signature including trapdoor hash signature [16], Camenisch and Lysyanskaya [13] and Fischlin's signature scheme [22] are all suitable for our purpose. However, among the signatures listed above, Zhu's signature is the most efficient (see the appendix 1 for more details).

Zhu's signature scheme [33] Zhu's signature scheme is defined as follows (see appendix for more details):

– Key generation algorithm: Let p, q be two large primes such that $p-1 = 2p'$ and $q-1 = 2q'$, where p', q' are two $(l'+1)$-bit strings. Let $n = pq$ and QR_n be the quadratic residue of Z_n^*. Let g, h be two generators of QR_n. The public key is (n, g, h, X, H), where $X \in QR_n$ and H is a collision free hash function with output length l. The private key is (p, q).
– Signature algorithm: To sign a message m, a $(l+1)$-bit prime e and a string $t \in \{0,1\}^l$ are chosen at random. The equation $y^e = Xg^t h^{H(m)} \mathrm{mod} n$ is solved for y. The corresponding signature of the message m is (e, t, y).
– Verification algorithm: Given a putative triple (e, t, y), the verifier first checks that e is an odd $(l+1)$-bit number. Second it checks the validation that $X = y^e g^{-t} h^{-H(m)} \mathrm{mod} n$. If the equation is valid, then the verifier accepts, otherwise, it rejects.

Committed signature from independent signatures We are now ready for describing the first committed signature scheme:

Key generation algorithm: $(N_1, N_2, X_1, X_2, g_1, h_1, g_2, h_2, H) \leftarrow Setup(1^k)$, where $N_i = p_i q_i$ and $p_i = 2p_i' + 1$, $q_i = 2q_i' + 1$, $i = 1, 2$. Let G_i be the quadratic residue of $Z_{N_i}^*$. Let g_1, h_1 be two generators of group G_1 and g_2, h_2 be two generators of G_2. Let $X_1 \in QR_{N_1}$ and $X_2 \in QR_{N_2}$ are two random chosen elements. Let H be a collision free hash function with output length l, eg, l= 160. The public key $PK = (N_1, N_2, X_1, X_2, g_1, h_1, g_2, h_2, H)$, $APK = (N_2, X_2, g_2, h_2, H)$. The secret key $SK = (p_1, q_1, p_2, q_2)$ and $ASK = (p_2, q_2)$.

Partial signing algorithm $PSig$ and correspondent verification algorithm $PVer$: To partially sign a message m, a $(l+1)$-bit prime e and a l-bit string t are chosen uniformly at random. The equation $y_1^e = X_1 g_1^t h_1^{H(m)} \mathrm{mod} N_1$ is solved for y_1. The partial signature of message m is $\sigma' = (e, y_1, t)$. The partial verification algorithm outputs 1, i.e., $PVer(m, \sigma') = 1$ if $\sigma'(m)$ satisfies the equation: $y_1^e = X_1 g_1^t h_1^{H(m)} \mathrm{mod} N_1$.

Full signing algorithm Sig and correspondent verification algorithm Ver: To fully sign a message m, the primary signer solves the equation $y_2^e = X_2 g_2^t h_2^{H(m)} \mathrm{mod} N_2$ is solved for y_2. The corresponding full signature of the message m is $\sigma = (e, t, y_1, y_2)$. To verify the correctness of signature scheme, it tests whether the equations $y_1^e = X_1 g_1^t h_1^{H(m)} \mathrm{mod} N_1$ and $y_2^e = X_2 g_2^t h_2^{H(m)} \mathrm{mod} N_2$. If both equations are valid, then the verification function outputs $Ver(m, \sigma) = 1$, otherwise, it outputs 0;

Resolution algorithm Res: Given a partial signature $\sigma' = (e, y_1, t)$ of message m, the co-signer computes y_2 from the equation $y_2^e = X_2 g_2^t h_2^{H(m)} \mathrm{mod} N_2$. The output of $Res(m, \sigma') = \sigma(m) := (e, t, y_1, y_2)$.

-We remark that the modulus N_1 and N_2 are chosen independently except for the same bit-length ($|N_1| = |N_2| = 2k$, k is the system security parameter). The safe primes (p_1, q_1) are chosen by the primary signer while (p_2, q_2) are jointly chosen by the primary signer and the co-singer, e.g., the co-signer chooses (p_2, q_2) uniformly at random for a prime number set, signs-then-encrypts the prime numbers, and sends the cipher-text to the primary signer.

-We remark that at the registration stage in a fair exchange system, a primary signer Alice has to prove the certificate authority (CA) that N_1, N_2 is a product of safe primes without revealing (p_1, q_1) and (p_2, q_2). This can be done using zero-knowledge protocol of Camenisch and Michels [14]. After verifying the construction of N_1, N_2, the CA issues a certificate $Cert_{N_1,N_2}$.

-We also remark that if e is a $(l+1)$-bit prime chosen uniformly at random for partially sign a message m, which is co-prime with $\phi(N_1)$ then we can assume that $\gcd(e, \phi(N_2)=1$ also duo to the fact that $N_i = p_i q_i$ and $p_i = 2p'_i + 1$, $q_i = 2q'_i + 1$, p_i, q_i, and p'_i and q'_i are primes with the same bit length for $i = 1, 2$.

-There are two independent signatures used in our committed signature scheme nevertheless the protocol is efficient and is nontrivial as we can reuse the random string t and prime number e.

The proof of security We prove the security against primary signer Alice, verifier Bob and co-signer/arbitrator Charlie respectively below under the strong RSA assumption and the assumption that H is a collision resistant.

Security against the primary signer Alice is trivial since the co-signer holds ASK in the protocol.

Security against the verifier Bob: Suppose Alice has committed to the transaction by providing her partial signature σ', then it should be computational infeasible for Bob to compute the signature σ from the partial signature σ'. Suppose the malicious verifier Bob was given a valid partial signature $\sigma' = (e, t, y_1)$ of the message m generated by the honest primary signer Alice. If Bob can forge a valid full signature (e, t, y_1, y_2) from $\sigma' = (e, t, y_1)$ with non-negligible probability, then we make use of Bob as a subroutine to break strong RSA assumption. The simulation is the same as that in Zhu's proof, therefore omitted (see appendix 2 for more details).

Security against the co-signer/arbitrator Charlie: Even though the co-signer (arbitrator) is semi trusted, the primary signer does not want this co-signer to produce valid signature which the primary signer did not intend on producing. In other words, if co-signer is able to forgery a partial signature of a message m, then we make use of Charlie as a subroutine to break the strong RSA assumption. Since Bob holds the correspondent ASK, therefore we can assume that Bob succeeds in forging a valid partial signature with non-negligible probability. The simulation is the same as that in Zhu's proof, therefore omitted (see appendix 2 for more details).

3.2 Committed signature from single signature scheme

For the purpose of consistency, we prefer to design committed signature scheme from single signature scheme. The existence of secure and efficient committed signatures in the random oracle model are already known [20]. The challenge problem is to construct a committed signatures from one signature in the standard complexity model. In this setting, we define a Cramer-Shoup like (CS-like)

trapdoor hash scheme in a quadratic residue as the trapdoor information allows the co-signer to control the full signature.

CS-like trapdoor hash scheme defined over quadratic residue The extended signature is defined as follows:

- Key generation algorithm: $(N_1, N_2, X_1, X_2, h, g_1, g_2, H) \leftarrow Setup(1^k)$, where $N_i = p_i q_i$ and $p_i = 2p'_i + 1$, $q_i = 2q'_i + 1$, $i = 1, 2$. Let p_1, p_2, q_1, q_2 be four large primes such that $p_i - 1 = 2p'_i$ and $q_i - 1 = 2q'_i$, where p'_i, q'_i are two l'-bit strings, $i = 1, 2$. Let $N_i = p_i q_i$ and QR_{N_i} be the quadratic residue of $Z^*_{N_i}$. Let X_1, h be two generators of QR_{N_1}. Let X_2, g_1, g_2 be three generators of QR_{N_2}. The public key is $(N_1, N_2, X_1, X_2, h, g_1, g_2, H)$. The private key is (p_1, q_1, p_2, q_2).
- Signature algorithm: To sign a message m, a $(l+1)$-bit prime e and a string $t \in \{0,1\}^l$ is chosen uniformly at random. The equation:

$$y^e = X_1 h^{H(X_2 g_1^t g_2^{H(m)} \bmod N_2)} \bmod N_1$$

  is solved for y. The corresponding signature of the message m is (e, t, y).
- Verification algorithm: Given a putative triple (e, t, y), the verifier first checks that e is an odd $(l+1)$-bit number. Second it checks the validity of the equation:

$$X_1 = y^e h^{-H(X_2 g_1^t g_2^{H(m)} \bmod N_2)} \bmod N_1.$$

  If the equation is valid, then the verifier accepts, otherwise, it rejects.

The proof of security is very similar with that of Zhu's signature scheme that is presented in the appendix, therefore omitted.

Committed signature from single signature We are now ready for describing the second committed signature scheme:

Key generation algorithm: $(N_1, N_2, X_1, X_2, h, g_1, g_2, H) \leftarrow Setup(1^k)$, where $N_i = p_i q_i$ and $p_i = 2p'_i + 1$, $q_i = 2q'_i + 1$, $i = 1, 2$. Let p_1, p_2, q_1, q_2 be four large primes such that $p_i - 1 = 2p'_i$ and $q_i - 1 = 2q'_i$, where p'_i, q'_i are two l'-bit strings, $i = 1, 2$. Let $N_i = p_i q_i$ and QR_{N_i} be the quadratic residue of $Z^*_{N_i}$. Let X_1, h be two generators of QR_{N_1}. Let X_2, g_1, g_2 be three generators of QR_{N_2}. The primary signer's public key PK is $(N_1, N_2, X_1, X_2, h, g_1, g_2, H)$, the private key SK is (p_1, q_1, p_2, q_2). The APK of the co-signer is (N_2, X_2, g_1, g_2, H), and the secret key ASK is (p_2, q_2).

Partial signing algorithm $PSig$ and correspondent verification algorithm $PVer$: To partially sign a message m, a $(l+1)$-bit prime e and a l-bit string t are chosen at random. The equation:

$$y_1^e = X_1 h^{H(X_2 g_1^t g_2^{H(m)} \bmod N_2)} \bmod N_1$$

is solved for y_1. The partial signature of message m is $\sigma' = (e, t, y_1)$. The partial verification algorithm outputs 1, i.e., $PVer(m, \sigma') = 1$ if $\sigma'(m)$ satisfies the equation:

$$y_1^e = X_1 h^{H(X_2 g_1^t g_2^{H(m)} \bmod N_2)} \bmod N_1$$

Full signing algorithm Sig and correspondent verification algorithm Ver: The equation

$$y_2^e = X_2 g_1^t g_2^{H(m)} \mathrm{mod} N_2$$

is solved for y_2. The corresponding full signature of the message m is $\sigma = (e, t, y_1, y_2)$. To verify the correctness of full signature scheme, it tests whether the equations

$$y_1^e = X_1 h^{H(X_2 g_1^t g_2^{H(m)} \mathrm{mod} N_2)} \mathrm{mod} N_1$$

and

$$y_2^e = X_2 g_1^t g_2^{H(m)} \mathrm{mod} N_2.$$

If both equations are valid, then the verification function outputs $Ver(m, \sigma) = 1$, otherwise, it outputs 0;

Resolution algorithm Res: Given a partial signature $\sigma' = (e, t, y_1)$ of message m, the co-signer computes y_2 from the equation

$$y_2^e = X_2 g_1^t g_2^{H(m)} \mathrm{mod} N_2.$$

The output of $Res(m, \sigma') = \sigma(m) := (e, t, y_1, y_2)$.

The remarks on the the first committed signature scheme is also suitable for this scheme. And the proof of security of the second scheme is very similar with that of the first one, therefore omitted.

4 Optimistic fair exchange protocol from committed signature scheme

An optimistic fair exchange protocol consists of three sub-protocols: registration protocol, exchange protocol and dispute resolution protocol. Furthermore it should be assumed that, although not explicitly stated in the protocol, sensitive data being exchanged are encrypted to assure confidentiality. Using the following optimistic fair exchange protocol, the primary signer Alice is trying to purchase electronic goods from the verifier Bob, and co-signer Charlie is the TTP.

1. Registration protocol:

(1.1) Alice using KG, generates two safe primes p_1, q_1, and two random generators $g_1, h_1 \in QR_{N_1}$ and a collision free hash function H with output length l, where $N_1 = p_1 q_1$, $p_1 = 2p_1' + 1$ and $q_1 = 2q_1' + 1$, p_1, q_1, p_1', q_1' are primes;

(1.2) Alice co-operated with her co-signer Charlie, using KG, generates two safe primes p_2, q_2 and two random generators $g_2, h_2 \in QR_{N_2}$, and a collision free hash function H with output length l, where $N_2 = p_2 q_2$, $p_2 = 2p_2' + 1$ and $q_2 = 2q_2' + 1$, p_2, q_2, p_2', q_2' are primes. In this phase, we emphasize that both Alice and Bob know the explicit values p_2, q_2, p_2', q_2' at the end the interactive protocol. Finally co-signer Charlie sings the agreed secret key and public key, denoted by $Cert_{cosig}$.

(1.3) ASK=(p_2, q_2), APK=(N_2, g_2, h_2, H); PK=$(N_1, N_2, g_1, h_1, g_2, h_2, H)$, SK=p_1, p_2, p_2, q_2.

(1.4) Alice sends APK=(N_2, g_2, h_2, H) and its certificates $Cert_{cosig}$ as well as PK=$(N_1, N_2, g_1, h_1, g_2, h_2, H)$ to the certificate authority (CA), and proves that N_1, N_2 is a product of two safe primes without revealing (p_1, q_1) and (p_2, q_2). This can be done using zero-knowledge protocol of Camenisch and Michels [14]. After verifying the construction of N_1, N_2, the CA issues a certificate $Cert_{N_1,N_2}$.

2. Exchange protocol:

(2.1) The primary signer Alice computes her partial signature σ', and sends the verifier Bob $Cert_{N_1,N_2}$, and σ';

(2.2) Bob using $Cert_{N_1,N_2}$ verifies the certificate. Then Bob verifies the partial signature σ'. If everything is in order, Bob sends his requirement to Alice;

(2.3) After verifying the requirement, Alice computes the full signature σ. Then sends it to Bob.

(2.4) Bob verifies the full signature σ, and it ends the protocol if it is correct.

3. Dispute resolution protocol: If the verifier Bob does not receive the full signature or the full signature is invalid, he initiates a dispute resolution protocol by contracting the co-signer/arbitrator Charlie:

(3.1) Bob sends $Cert_{N_1,N_2}$, σ' and his obligation to Charlie;

(3.2) Charlie checks the validity of the items received. If everything is in order, Charlie creates the full signature σ.

(3.3) The full signature is given to Bob, and the obligation is forward to Alice.

The security of the fair exchange protocol follows from the security of the underlying committed signature scheme. Similarly, we can define another fair exchange protocol based on the committed signature scheme from the single signature scheme. We observe that our framework does not address a subtle issue of timely termination address by [3] and [4]. We remark that however the technique of [3] and [4] can be easily added to our solution to resolve this problem.

5 Conclusions

Two efficient committed signatures have been presented in this report. Both are provably secure in the standard complexity model from strong RSA assumption. Finally two efficient optimistic fair exchange protocols are derived from two new primitives.

Acknowledgement Great thanks to Dr. Yevgeniy Dodis for discussions and comments.

References

1. G. Ateniese, Efficient Verifiable Encryption (and Fair Exchange) of Digital Signatures. In 6th ACM Conference on Computer and Communications Security (ACM CCS'99), 138-146.
2. N. Asokan, M. Schunter, M. Waidner: Optimistic Protocols for Fair Exchange. ACM Conference on Computer and Communications Security 1997: 7-17.
3. N. Asokan, V. Shoup, M. Waidner: Optimistic Fair Exchange of Digital Signatures (Extended Abstract). EUROCRYPT 1998: 591-606.
4. N. Asokan, Victor Shoup, Michael Waidner: Optimistic Fair Exchange of Digital Signatures. IEEE JOurnal on Selected Areas in Communications, Vol 18, No.4, 2000, 593-610.
5. D.Boneh, H.Shacham, and B.Lynn Short signatures from the Weil pairing. In proceedings of Asiacrypt '01, LNCS Vol. 2248, Springer-Verlag, pp. 514-532, 2001.
6. F. Bao. An efficient verifiable encryption scheme for encryption of discrete logarithms. CARDIS'98, 213-220.
7. F. Bao, R. Deng, W. Mao, Efficient and Practical Fair Exchange Protocols, Proceedings of 1998 IEEE Symposium on Security and Privacy, Oakland, pp. 77-85, 1998.
8. A. Boldyreva. Efficient threshold signatures, multisigntaures and blind signatures based on the Gap Diffie Helman group signature scheme. PKC 2003, LNCS 2567.
9. C. Boyd, E. Foo: Off-Line Fair Payment Protocols Using Convertible Signatures. ASIACRYPT 1998: 271-285
10. M. Bellare and S. Micali. How to sign given any trapdoor permutation. Journal of the ACM, Vol. 39, No. 1, January 1992, pp. 214–233.
11. Manuel Blum: How to Exchange (Secret) Keys (Extended Abstract). STOC 1983: 440-447.
12. N. Braic and B. Pfitzmann. Collision free accumulators and fail-stop signature scheme without trees. Eurocrypt'97, 480-494, 1997.
13. J.Camenisch, A. Lysyanskaya. A Signature Scheme with Efficient Protocols. SCN 2002: 268-289.
14. Jan Camenisch, Markus Michels: Proving in Zero-Knowledge that a Number Is the Product of Two Safe Primes. EUROCRYPT 1999:107-122
15. R. Cramer and I. Damgard. New generation of secure and practical RSA-based signature. Crypto'96. 173-185, 1996.
16. R. Cramer and V. Shoup. Signature scheme based on the Strong RAS assumption. 6th ACM Conference on Computer and Communication Security, Singapore, ACM Press, November 1999.
17. B.Cox, J.D.Tygar and M.Sirbu, NetBill Security and Transaction Protocol. In first USENIX workshop on Electronic Commerce, 1995, 77-88.
18. Ivan Damgård: Practical and Provably Secure Release of a Secret and Exchange of Signatures. Journal of Cryptology 8(4): 1995, 201-222.
19. Cynthia Dwork, Moni Naor: An Efficient Existentially Unforgeable Signature Scheme and its Applications. *CRYPTO* 1994: 234-246.
20. Y.Dodis, L. Reyzin Breaking and Repairing Optimistic Fair Exchange from PODC 2003. http://eprint.iacr.org.

21. S.Even, O. Goldreich, A. Lempel: A Randomized Protocol for Signing Contracts. CRYPTO 1982:205- 210.
22. Marc Fischlin: The Cramer-Shoup Strong-RSASignature Scheme Revisited. Public Key Cryptography, 2003: 116-129.
23. M. K. Franklin and M. Reiter: Fair exchange with a semi-trusted third party. IN ACM Security,1-5.
24. J.A.Garay, M. Jakobsson, P.D.MacKenzie: Abuse-Free Optimistic Contract Signing. CRYPTO, 1999: 449-466.
25. O. Goldreich: A Simple Protocol for Signing Contracts. CRYPTO 1983: 133-136
26. E. Fujisaki, T. Okamoto. Statistical zero-knowledge protocols to prove modular polynomial relations. Crypto'97, LNCS 1294, Springer-verlag, 1997.
27. R. Gennarou, S. Halevi and T. Rabin. Secure hash-and-sign signatures without random oracle. Eurocrypto'99. 123-139, 1999.
28. S. Goldwasser, S. Micali, R. Rivest: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM J. Comput. 17(2): 281-308, 1988.
29. L. Guillou, J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessors minimizing both transmission and memory. Eurocrypto'88, 123-128, 1988.
30. H.Krawczyk, T. Rabin. Chameleon hashing and signatures. Theory of Cryptography Library. March 1998.
31. V. Shoup and R. Gennaro. Securing threshold cryptosystems against chosen cipher-text attack. Crypto'98, 1-16, 1998.
32. J. M. Park, E. Chong, H. Siegel, and I. Ray. Constructing Fair-Exchange Protocols for E-Commerce Via Distributed Computation of RSA Signatures, PODC 2003, 172-181.
33. Huafei Zhu. New Digital Signature Scheme Attaining Immunity to Adaptive Chosen-message attack. Chinese Journal of Electronics, Vol.10, No.4, Page 484-486, Oct, 2001.

Appendix: A formal proof of Zhu's signature scheme

Appendix 1: Related works

Cramer-Shoup's trapdoor hash scheme Cramer and Shoup presented an elegant signature scheme called trapdoor hash function defined below (see [16] for more details):

- Key generation algorithm: Let p, q be two large primes such that $p-1 = 2p'$ and $q-1 = 2q'$, where p', q' are two l'-bit strings. Let $n = pq$ and QR_n be the quadratic residue of Z_n^*. Let x, h be two generators of QR_n. Also chosen are a group G of order s, where s is $(l+1)$-bit prime, and two random generators g_1, g_2 of G. The public key is (n, h, x, g_1, g_2, H) along with an appropriate description of G including s. The private key is (p, q).
- Signature algorithm: To sign a message m, a $(l+1)$-bit prime e and a string $t \in Z_s$ is chosen. The are chosen at random. The equation $y^e = xh^{H(g_1^t g_2^{H(m)})} \mathrm{mod} n$ is solved for y. The corresponding signature of the message m is (e, t, y).
- Verification algorithm: Given a putative triple (e, t, y), the verifier first checks that e is an odd $(l+1)$-bit number. Second it checks the validation that $x = y^e h^{-H(g_1^t g_2^{H(m)})} \mathrm{mod} n$. If the equation is valid, then the verifier accepts, otherwise, it rejects.

Zhu's signature scheme In their scheme, another extra group G is defined. From the point views of computational complexity it is non-trivial work therefore if one can reduce the computational and communication complexity while its provability and efficiency can be maintained. Based on this observation, Zhu provides a variation scheme below [33]:

- Key generation algorithm: Let p, q be two large primes such that $p-1 = 2p'$ and $q-1 = 2q'$, where p', q' are two $(l'+1)$-bit strings. Let $n = pq$ and QR_n be the quadratic residue of Z_n^*. Let g, h be two generators of QR_n. The public key is (n, g, h, X, H), where $X \in QR_n$ and H is a collision free hash function with output length l. The private key is (p, q).
- Signature algorithm: To sign a message m, a $(l+1)$-bit prime e and a string $t \in \{0,1\}^l$ are chosen at random. The equation $y^e = Xg^t h^{H(m)} \mathrm{mod} n$ is solved for y. The corresponding signature of the message m is (e, t, y).
- Verification algorithm: Given a putative triple (e, t, y), the verifier first checks that e is an odd $(l+1)$-bit number. Second it checks the validation that $X = y^e g^{-t} h^{-H(m)} \mathrm{mod} n$. If the equation is valid, then the verifier accepts, otherwise, it rejects.

Camenisch-Lysyanskaya's signature scheme In SCN'02, Camenisch and Lysyanskaya [13] presented alternative signature scheme. The Camenisch and Lysyanskaya signature is described as follows (see [13] for more details).

– Key generation algorithm: On input 1^k, choose a special RSA modulus $n = pq$, $p = 2p' + 1$, $q = 2q' + 1$ of length $l_n = 2k$. Choose, uniformly at random, $a, b, c \in QR_n$. Output $PK = (n, a, b, c)$, and $SK = p$.
– Message space. Let l_m be a parameter. The message space consist of all binary string of length l_m. Equivalently, it can be thought of as consisting of integers in the range $[0, 2^{l_m})$.
– Signing algorithm: On input m, choose a random prime number $e > 2^{l_m+1}$ of length $l_e = l_m + 2$, and a random number s of length $l_s = l_n + l_m + l$, where l is a security parameter. Compute the value v such that

$$v = ca^m b^s \mathrm{mod} n$$

– Verification algorithm: To verify that the tuple (e, s, v) is a signature on message m in the message space, check that $v = ca^m b^s \mathrm{mod} n$ and check that $2^{l_e} > e > 2^{l_2-1}$.

Fischlin's signature scheme Later a similar modification is presented in PKC'03 by Marc Fischlin. Fischlin's signature scheme is defined as follows [22]:

– Key generation: Generating $n = pq$, where $p = 2p' + 1$ and $q = 2q' + 1$ for primes p, q, p', q'. Also pick three quadratic residue $h_1, h_2, x \in QR_n$. The public key verification key is (n, h_1, h_2, x) and the private key is (p, q).
– Signing: To sign a message m calculate the l-bit hash value $H(m)$ with a collision-intractable hash function $H(\cdot)$. Pick a random $(l + 1)$-bit prime e, and a random l-bit string α and compute a representation $(-\alpha, -(\alpha \oplus H(m)), y)$ of x with respect to h_1, h_2, e, n, i.e.,

$$y^e = x{h_1}^{\alpha} {h_2}^{\alpha \oplus H(m)} \mathrm{mod} n.$$

Computing this e-th root y from $x{h_1}^{\alpha} {h_2}^{\alpha \oplus H(m)}$ is easy given the factorization of n. The signature is (e, α, y).
– Check that e is an odd $(l + 1)$-bit integer, that α is l bits long, and that $y^e = x{h_1}^{\alpha} {h_2}^{\alpha \oplus H(m)} \mathrm{mod} n$.

The relationship between Zhu's signature and Camenisch-Lysyanskaya's signature scheme is obvious. Here we remark the relationship between Zhu's signature schemes and Fischlins's scheme therefore.

– It is clear that the algebraic structures of Zhu's and Fischlin's signature are same;
– If there is no collision hash function involved in the above two schemes, then it is not hard to show that the above two signature schemes are equivalent in the same security level. More precisely, if Zhu's scheme can be broken by an adversary A with non-negligible probability then there exists an adversary B^A so that Fischlin's signature scheme can be broken with the same probability. The statement is also true by means of vis-a-vis argument.

– In case of a collision free hash function involved in both schemes, suppose Zhu's signature scheme can be broken with non-negligible probability, i.e., there is an adversary A is able to forge a faking message m in Zhu's signature scheme, denoted by $\sigma(m) = (e, y, t)$ with non-negligible probability. Then there exists an adversary B^A in Fischlin's signature scheme so that it is able to produce a valid signature $\sigma(m') = (e, y, t)$ for any message in the set $S := \{m' | H(m) \oplus H(m') = t\}$, where t is a component of faking signature $\sigma(m)$ correspondent to Zhu's signature scheme. The statement is also true by means of vis-a-vis argument.

Appendix 2: Formal proof of Zhu's signature scheme

Main result: Zhu's signature scheme is immune to adaptive chosen-message attack under the strong RSA assumption and the assumption that H is a collision resistant.

Proof: Assume that the signature scheme is NOT secure against adaptive chosen message attack. That is, there is an adversary, who is able to forge the signature (e, t, y) of a message $m(m \neq m_i,\ 1 \leq i \leq f)$ with non-negligible probability after it has queried correspondent signature of each message $m_1, \cdots, m_f$, which is chosen adaptively by the adversary. Let $(e_1, t_1, y_1), \cdots, (e_f, t_f, y_f)$ be signatures provided by the signing oracle corresponding to a set of messages $m_1, \cdots, m_f$. We consider three types of forgeries: 1) for some $1 \leq j \leq f$, $e = e_j$ and $t = t_j$; 2) for some $1 \leq j \leq f$, $e = e_j$ and $t \neq t_j$; 3) for all $1 \leq j \leq f$, $e \neq e_j$. We should show that any forgery scheme of the two types will lead to a contradiction to the assumptions of the theorem. This renders any forgery impossible.

Type 1-Forger

We consider an adversary who chooses a forgery signature such that $e = e_j$ for a fixed j: $1 \leq j \leq f$, where f is the total number of the queries to the signing oracle. If the adversary succeeds in a signature forgery as type1 with non-negligible probability then given n, we are able to compute $z^{1/r}$ with non-negligible probability, where r is a $(l+1)$-bit prime. This contradicts to the assumed hardness of the standard RSA problem. We state the attack in details as follows: given $z \in Z_n^*$ and r, we choose a set of total $f-1$ primes with length $(l+1)$-bit $e_1, ...e_{j-1}, e_{j+1}, ..., e_f$ uniformly at random. We then create the correspondent public key (X, g, h) of the simulator as follows: given $z \in Z_n^*$ and r, we choose a set of total $f-1$ primes with length $(l+1)$-bit $e_1, ...e_{j-1}, e_{j+1}, ..., e_f$ uniformly at random. We choose $w, v \in Z_n$ uniformly at random, and compute $h = z^{2e_1 \cdots e_{j-1} e_{j+1} \cdots e_f}$, $g = v^{2e_1 \cdots e_f} z^{2e_1 \ldots e_{j-1} e_{j+1} \ldots e_f}$ and $X = w^{2\beta e_1 \cdots e_f} z^{2e_1 \ldots e_{j-1} e_{j+1} \ldots e_f(-\alpha)}$, where $\alpha \in \{0,1\}^{l+1}$ and $\beta \in Z_n$ are chosen uniformly at random.

Since the simulator knows each e_i, therefore it is easy to compute the i-th signing query. What we need to show is how to simulate the j-th signing query.

This can be done as follows:

$$y_j^{e_j} = Xg^{t_j}h^{H(m_j)} = (w^{\beta}v^{t_j})^{2e_1\cdots e_f} z^{2e_1\ldots e_{j-1}e_{j+1}\ldots e_f(-\alpha+t_j+H(m_j))}$$

Now we set $-\alpha + t_j + H(m_j) = 0$, i.e, $t_j = \alpha - H(m_j)$.

To show the simulation above is non-trivial, we should show t_i is uniformly distributed over $\{0,1\}^l$ with non-negligible amount. Since $\alpha \in \{0,1\}^{l+1}$ is chosen uniformly at random, i.e., $0 \leq \alpha \leq 2^{l+1} - 1$, the probability t_j belongs to the correct interval and it does so with the correct uniform distribution can be computed as follows:

$$\frac{(2^{l+1} - 1 - H(m_j) - 2^l + 1) + H(m_j)}{(2^{l+1} - 1 - H(m_j)) - (-H(m_j)) + 1} = 1/2$$

Suppose the adversary is able to forge a faking signature of message m, denoted by (e, y, t), such that $e_j = e(= r)$, $t_j = t$. Notice that one can not assume that $e_j = e$, $t_j = t$ and $y_j = y$, since H is a collision free hash function. Now we have two equations: $y_j^e = Xg^t h^{H(m_j)}$ and $y^e = Xg^t h^{H(m)}$. Consequently, we obtain the equation:

$$(\frac{y_j}{y})^e = h^{H(m_j)-H(m)} = z^{2e_1,\ldots e_{j-1},e_{j+1},\ldots,e_f(H(m_j)-H(m))}$$

It follows that one can extract the e-th root of z with non-negligible probability. Therefore, we arrive at the contradiction of the standard hardness of RSA assumption.

Type 2-Forger

We consider an adversary who succeed in forging a valid signature such that $e = e_j$, $t \neq e_j$ for a fixed j: $1 \leq j \leq f$, where f is the total number of the queries to the signing oracle. If the adversary succeeds in a signature forgery as type1 with non-negligible probability then given n, we are able to compute $z^{1/r}$ with non-negligible probability for a given z and r, where r is a $(l+1)$-bit prime. This contradicts to the assumed hardness of the standard RSA problem. We state the attack in details as follows: given $z \in Z_n^*$ and r, we choose a set of total $f-1$ primes with length $(l+1)$-bit $e_1, \ldots e_{j-1}, e_{j+1}, \ldots, e_f$ at random. We then create the correspondent public key (X, g, h) of the simulated signature scheme as follows: $g = z^{2e_1\ldots e_{j-1}e_{j+1}\cdots e_f}$, $h = v^{2e_1\ldots e_f}$ and $X = g^{-\alpha}w^{2e_1\ldots e_f}$, where $w, v \in Z_n$ and α is a l-bit random string. Since QR_n is a cyclic group, we can assume that g, h are generators of QR_n with overwhelming probability. To sign the i-th message $m_i (i \neq j)$, the signing oracle selects a random string $t_i \in \{0,1\}^l$, and computes:

$$y_i{}^{e_i} = ((wv^{H(m_i)})^{2e_1\ldots e_{i-1}e_{i+1}\ldots e_f} z^{2(t_i-\alpha)\Pi_{s\neq i, s\neq j}e_s})^{e_i}$$

The output of the signing oracle is a signature of message m_i, denoted by $\sigma(m_i) = (e_i, y_i, t_i)$.

To sign the j-th message m_j, the signing oracle, sets $t_j \leftarrow \alpha$ and computes:

$$y_j{}^{e_j} = ((wv^{H(m_j)})^{2\Pi_{s\neq j}e_s})^{e_j}$$

The output of the signing oracle is a signature of message m_j, denoted by $\sigma(m_j) = (e_j, y_j, t_j)$.

Let $\sigma(m) = (e, y, t)$ be a valid signature forged by the adversary of message m. By assumption, we know that $y^e = Xg^t h^{H(m)}$. Consequently, we have the following equation:

$$g^{t_j} h^{H(m_j)} y_j{}^{e_j} = g^t h^{H(m)} y^e$$

Equivalently

$$z^{2(\alpha-t)\Pi_{i\neq j}e_i} = (v^{2(H(m)-H(m_j))\Pi_{i\neq j}e_i} \frac{y}{y_j})^{e_j}$$

Since $t_j = \alpha$ and $t \neq t_i$ by assumption, it follows that $t \neq \alpha$. We then apply Guillou-Quisquater lemma to extract the r-th root of z, where $r = e_j$.

Type 3-Forger

We consider the second type of the attack: the adversary forgery is that for all $1 \leq j \leq f$, $e \neq e_j$. If the adversary succeeds in forgery with non-negligible probability, then given n, a random $z \in Z_n^*$, we are able to compute $z^{1/d}$ ($d > 1$) with non-negligible probability, which contradicts to the assumed hardness of strong RSA assumption. We state our attack in details as follows: we generate g and h with the help of z. We define $g = z^{2e_1 \cdots e_f}$ and $h = g^a$, where $a \in (1, n^2)$, is a random element. We can assume that g is a generator of QR_n with overwhelming probability. Finally, we define $X = g^b$, where $b \in (1, n^2)$. Since the simulator knows the all e_j, the signature oracle can be perfectly simulated. Let (e, t, y) be a forgery signature of message m. It yields the equation $y^e = Xg^t h^{H(m)} = z^E$, where $E = (b + t + aH(m))2e_1 ... e_f$.

Since we are able to compute (e/E)-th root of z provided e is a not a divisor of E according to the lemma of Guillou and Qusiquater, it is sufficient to show that e is not a divisor of E with non-negligible probability. Due to the the fact that $\gcd(e, e_1 e_2 \cdots e_f) = 1$, it is sufficient to show that e is not a divisor of $b+t+aH(m)$ with non-negligible probability. Since $b \in (1, n^2)$, it follows that one can write $b = b'p'q' + b''$. Therefore, the probability that $b+t+aH(m) \equiv 0 \mathrm{mod} e$ is about $1/e$.

Remark on Type 3- Forger: To show that $e|(b + t + aH(m)$ with negligible probability, one may make use of randomness of $a \in (1, n^2)$. That is one can write a as $a = a'p'q' + a''$. It follows a' is a random element from the adversary's view. Hence the probability that $b + t + aH(m) \equiv 0 \mathrm{mod} e$ is about $1/e$. Thus, with non-negligible probability, e is not a divisor of $b + t + aH(m)$. We point out that since the adversary may find $H(m) = 0$, the term $aH(m)$ may be cancelled in the formula in the equation. Thus the random argument must be done in term b instead of $aH(m)$ since collision-resistance does not imply zero-finder intractability in general. This remark also suitable for Cramer-Shoup's argument.