An Efficient Abuse-Free Fair Contract-Signing Protocol Based on RSA Signature and Σ-protocol

Xi-Jun Lin *and Lin Sun †

April 26, 2014

Abstract: A fair contract-signing protocol is an important mechanism which allows two participants to sign a digital contract via the public computer networks in a fair way. Based on the RSA signature scheme and Σ-protocol, we propose a new contract-signing protocol in this paper. The proposed protocol is not only fair and optimistic, but also efficient and abuse-free. Moreover, security and efficiency analysis are provided.

Key words: Contract signing, fair exchange, digital signatures, RSA, e-commerce

1 Introduction

In electronic commerce transactions, contract-signing protocol is an important mechanism which allows two participants to sign a digital contract via the Internet. In such scenario, fairness is critical, because they may be potentially mistrusted. That is, such protocols are designed to solve the following problem: how to enable two potentially mistrusted participants exchanging digital signatures for a contract over computer networks in a fair way, i.e., at the end of the protocol, either both participants have valid signatures for a contract or neither does, even if one of them tries to cheat or the communication channel is out of order. Actually, fair exchange is a wide topic which involves payment protocols such as contract-signing protocols, electronic money and certified e-mail systems etc. In this paper, we focus on the construction of fair contract-signing protocol.

1.1 Related Work

In the fair contract-signing protocols, according to the involvement degree of a trusted third party (TTP), there are three types of them:

- gradual exchanges without any TTP
- protocols with an online TTP
- protocols with an off-line TTP

*X.J.Lin is with the Department of Computer Science and Technology, Ocean University of China. Qingdao 266100, P.R.China. email: linxj77@163.com

†L.Sun is with the College of Liberal Arts, Qingdao University. Qingdao 266071, P.R.China. email: sunlincctv@163.com

In the first type protocols [9, 13, 16], computational fairness is met in a "bit-by-bit" manner. That is, both participants exchange their secrets "bit-by-bit". If one participant stops before the end of the protocol, both participants have about the same fraction of the other's secret, which means that they can complete the contract off-line with about the same amount of computing work. No TTP is involved, but this mechanism is unrealistic for most real-world applications [7].

In the second type protocols [7, 10, 22], an online TTP is involved for each message exchange. The major advantage of this approach is that the construction is more easily. However, it may be inefficient since the TTP must be involved in every execution. Furthermore, the TTP may become a bottleneck in the whole system.

Compared with the above two types, the third type protocols [2–5, 18, 20] which involves an off-line TTP is more appealing and practical. Those protocols are optimistic, because the TTP is not invoked in the execution of exchange unless one participant misbehaves or the communication channel is out of order. Verifiably encrypted signatures [4, 5] and verifiable escrows [2, 3] are two major cryptographic primitives used in such protocols. However, the overheads of computation and communication are usually expensive [20].

Park et al. [18] proposed a fair and optimistic exchange protocol with an off-line TTP based on an RSA multisignature scheme. Unfortunately, their protocol was broken by Dodis et al. [11], i.e., an honest-but-curious TTP can easily derive a user's private key after the end of its registration. Dodis et al. proposed an improved protocol based on bilinear pairing. However, the bilinear pairing is still time-costing, although several approaches have been proposed to speed up its efficiency. Moreover, as mentioned in [20], essentially Dodis et al.'s protocol is not an improvement of Park et al.'s, because the security of their protocol is based on the GDH problem instead of the RSA problem or factoring problem.

It is highly desirable to construct fair exchange protocols based on RSA, since RSA is now the *de facto* industrial standard and is widely used in many applications. Motivated by this reason, Wang [20] proposed a fair and optimistic contract-signing protocol based on RSA signature. In his protocol, to capture the fairness property the initiator Alice's private key d is split into two parts d_1 and d_2, and the TTP hold d_2 secretly. Alice holds (d, d_1, d_2). Furthermore, the concept of abuse-freeness [14] was first introduced in the contract-signing protocol by Wang. That is, if the protocol is executed unsuccessfully, none of the two participants can show the validity of intermediate results to others. The abuse-freeness is an important security requirement for contract-signing protocols, which is achieved with a cryptographic primitive, called trapdoor commitment scheme, in Wang's protocol. However, the demerits of his protocol is that the overhead of communication becomes larger, since it exploits interactive protocol to prove the validity of partial signature [1].

1.2 Our Work

Motivated by the importance of abuse-freeness [14, 20], and the question of how to improve Wang's protocol [20] in an efficient way without losing security and functionality, we propose a new RSA-based contract-signing protocol in this paper. The proposed protocol is not only fair and optimistic, but also efficient and abuse-free. In Wang's protocol, trapdoor commitment scheme is used to capture the abuse-freeness property, but we integrate the special property of RSA signature and Σ-protocol into our proposed protocol. Technical analysis and discussion are provided in detail to show that the proposed protocol is secure. Compared with Wang's protocol, the proposed protocol is more efficient. More specifically,

the new protocol satisfies the following desirable properties which are described in Wang's protocol [20].

- **Fairness:** The proposed protocol guarantees the two participants to obtain or not obtain the peer's signature simultaneously. That is, even a dishonest participant who tries to cheat cannot get an advantage over the other participant.
- **Optimism:** The TTP is not invoked in the execution of exchange unless one participant misbehaves or the communication channel is out of order.
- **Abuse-Freeness:** If the protocol is not performed successfully, any of the two participants cannot show the validity of the intermediate results generated by the other to an outsider.
- **Timely Termination:** The execution of a protocol instance will be terminated in a predetermined time. This property is implemented by adding a reasonable deadline t in a contract, as used in [7, 20]. If one participant does not send its signature to the other after the deadline t, both of them are free of liability to their partial commitments to the contract and do not need to wait any more.
- **Compatibility:** In the proposed protocol, each participant's commitment to a contract is a standard digital signature. This means that to use the protocol in existing systems, there is no need to modify the signature scheme or message format at all. Thus, it will be very convenient to integrate this protocol into existing softwares for electronic transactions.
- **TTP's Statelessness:** To settle potential disputes between users, the TTP is not required to maintain a database for searching or remembering the state information of each protocol instance. Hence, the burden of the TTP is reduced greatly.
- **High Performance:** In a typical implementation, the protocol execution in a normal case requires only interaction of less rounds than Wang's.

The rest of the paper is organized as follows. Section 2 reviews concepts and building blocks used in our proposed protocol. In Section 3, we propose a new contact-signing protocol based on the RSA signature and Σ-protocol. Then, we analyze its security and efficiency in Sections 4 and 5, respectively. Finally, Section 6 concludes this paper.

2 Preliminaries

2.1 RSA Signature Scheme

we recall the RSA signature scheme which involves three algorithms: *Setup, Sign* and *Verify.*

- *Setup:* Take security parameter l as input, it performs as follows.
  1. Pick two distinct safe primes p and q of length l randomly.
  2. Compute $n = pq$.
  3. Choose an integer e such that $1 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$.

4. Determine d such that $ed \equiv 1 \pmod{\phi(n)}$, where $\phi(n) = (p-1)(q-1)$.
5. Output the public key (n, e), and the private key d.

- *Sign:* Take the message $m \in Z_n^*$ and the private key d as input, output $s = m^d \pmod{n}$ as the signature.
- *Verify:* Take the pair (m, s) and the public key (n, e) as input, check whether $s^e = m \pmod{n}$. If it holds, output 'True'; otherwise 'False'.

Needless to say, the above RSA signature scheme has following two properties.

1. *Existential forgery*: An adversary can pick $s \in_R Z_n^*$ and then compute $m = s^e \pmod{n}$. Obviously, the pair (m, s) satisfies the verification equation $s^e = m \pmod{n}$. Hence, the adversary generates a valid forgery pair (m, s) even without the knowledge of the private key d.
2. If the adversary has to output a message m before knowing the public key e, it cannot generate a valid forgery signature s, even given the public key e after m is outputted (it also does not know the private key d). That is, the adversary cannot forge a valid signature s if the message m is fixed before it knows the public key e.

The idea behind our construction for capturing abuse-freeness and authentication properties is to use the above two properties, which is depicted in Section 4.

2.2 Σ-protocol

In this subsection, we first recall the definition of Σ-protocol presented in [8]. Then, a concrete Σ-protocol proposed in [8], whose variant is used in our proposed protocol, is recalled.

Σ-protocol is an important cryptographic primitive for proof of knowledge. Let $\mathcal{R}$ be a binary relation, i.e., $\mathcal{R}$ is a subset of $\{0,1\}^* \times \{0,1\}^*$. For some $(x, v) \in \mathcal{R}$, x is an instance of some computational problem, and v is the solution to that instance. v is called a *witness* for x. It is concerned with protocols of the following form:

Protocol 1 *Let x be common input to the prover P, the verifier V and a v such that $(x, v) \in \mathcal{R}$ is private input to P. P and V perform the following steps.*

1. *P sends a message a to V.*
2. *V sends a random string c to P.*
3. *P sends a reply z, and V decides to accept or reject based on the data it has seen, i.e. x, a, c, z.*

It should be noted that both P and V are probabilistic polynomial time algorithms, so P's only advantage over V is that it knows v.

Definition 1 *A protocol $\mathcal{P}$ is said to be a Σ-protocol for relation $\mathcal{R}$ if:*

- *$\mathcal{P}$ is of the above 3-move form, and we have completeness: if P and V follow the protocol on input x and private input v to P where $(x, v) \in \mathcal{R}$, the verifier V always accepts.*

- *From any x and any pair of accepting conversations on input x, $(a, c, z), (a, c', z')$ where $c \neq c'$, one can efficiently compute v such that $(x, v) \in \mathcal{R}$. This is sometimes called the special soundness property.*
- *There exists a polynomial-time simulator, which on input x and a random c outputs an accepting conversation of the form (a, c, z), with the same probability distribution as conversations between the honest P and V on input x. This is sometimes called special honest-verifier zero-knowledge.*

Now, we recall a concrete Σ-protocol proposed in [8], which is used for equality of discrete logarithms.

Protocol 2 *Let p be a prime, q a prime divisor in $p-1$, and let $g, \bar{g}, h, \bar{h} \in \mathbb{Z}_p^*$ be of order q. Assume P gets as input v where $h = g^v \pmod p, \bar{h} = \bar{g}^v \pmod p$. P and V perform the following protocol:*

1. *P chooses $r \in_R \mathbb{Z}_q$ and sends $a = g^r \pmod p, \bar{a} = \bar{g}^r \pmod p$ to V.*
2. *V chooses a challenge c randomly and sends it to P.*
3. *P sends $z = r + cv \pmod q$ to V, who checks $g^z = ah^c \pmod p$ and $\bar{g}^z = \bar{a}\bar{h}^c \pmod p$, and accepts iff this is the case.*

Protocol 2 is a Σ-protocol for equality of discrete logarithms, more precisely show that this is a Σ-protocol for the relation $\{(x, v) | x = (p, q, g, \bar{g}, h, \bar{h})$ and $h = g^v, \bar{h} = \bar{g}^v\}$. That is, this protocol is for proving that two logarithms of h and $\bar{h}$ with respect to bases g and $\bar{g}$ are equal.

Note that, if there exists a cryptographically secure hash function H_1, the following is also a Σ-protocol for equality of discrete logarithms without losing security.

Protocol 3 *Let p be a prime, q a prime divisor in $p-1$, and let $g, \bar{g}, h, \bar{h} \in \mathbb{Z}_p^*$ be of order q. Assume P gets as input v where $h = g^v \pmod p, \bar{h} = \bar{g}^v \pmod p$. P and V perform the following protocol:*

1. *P chooses $r \in_R \mathbb{Z}_q$, computes $a = g^r \pmod p$ and $\bar{a} = \bar{g}^r \pmod p$, and sends $\xi = H_1(a, \bar{a})$ to V.*
2. *V chooses a challenge c randomly and sends it to P.*
3. *P sends $z = r + cv \pmod q$ to V, who computes $a = g^z / h^c \pmod p$ and $\bar{a} = \bar{g}^z / \bar{h}^c \pmod p$, and checks whether $\xi \stackrel{?}{=} H_1(a, \bar{a})$, and then accepts iff this is the case.*

Moreover, it is clear that if Protocol 3 is defined in $\mathbb{Z}_n^*$, it is still secure, where n is an RSA modulus. Hence, the following is also a secure Σ-protocol for equality of discrete logarithms, which is an important building block in our proposed protocol.

Protocol 4 *Set an RSA modulus $n = pq$, where p and q are two l-bit safe primes, i.e., there exist two primes p' and q' such that $p = 2p' + 1$ and $q = 2q' + 1$, and let $g, \bar{g}, h, \bar{h} \in \mathbb{Z}_n^* / \{1, -1\}$, $ord(g) \geq p'q', ord(h) \geq p'q'$. Assume P gets as input v where $h = g^v \pmod n, \bar{h} = \bar{g}^v \pmod n$. P and V perform the following protocol:*

1. *P chooses r randomly, computes* $a = g^r \pmod n, \bar{a} = \bar{g}^r \pmod n$, *and sends* $\xi = H_1(a, \bar{a})$ *to V.*

2. *V chooses a challenge c randomly and sends it to P.*

3. *P sends* $z = r + cv$ *to V, who computes* $a = g^z/h^c \pmod n$ *and* $\bar{a} = \bar{g}^z/\bar{h}^c \pmod n$, *checks whether* $\xi \stackrel{?}{=} H_1(a, \bar{a})$, *and then accepts iff this is the case.*

Protocol 4 is a Σ-protocol for equality of discrete logarithms, more precisely show that this is a Σ-protocol for the relation $\{(x, v)|x = (n, g, \bar{g}, h, \bar{h})$ and $h = g^v \pmod n, \bar{h} = \bar{g}^v \pmod n\}$.

3 The Proposed Scheme

In this section, we propose our contract-signing protocol based on RSA signature and Σ-protocol. The idea behind our construction is that Alice first splits her private key d into d_1 and d_2 such that $d = d_1 + d_2 \pmod n$, as Wang did in [20]. Then, only d_2 is sent to the TTP, while Alice keeps (d, d_1, d_2) secretly. To exchange her signature $\sigma_A = h(m)^d \pmod n$ with Bob, Alice first sends partial signature $\sigma_1 = h(m)^{d_1} \pmod n$ to Bob, and then she authenticates herself to Bob by exploiting the special properties of RSA signature and proves that σ_1 is prepared correctly in an interactive zero-knowledge way by exploiting Σ-protocol. After that, Bob sends his signature σ_B on message m to Alice, since he is convinced that even if Alice refuses to reveal the second partial signature $\sigma_2 = h(m)^{d_2} \pmod n$, the TTP can do the same thing.

The proposed protocol consists of three sub-protocols: *Registration Protocol*, *Signature Exchange Protocol* and *Dispute Resolution Protocol*.

3.1 Registration Protocol

In the proposed protocol, *Registration Protocol* is similar to Wang's [20]. To exchange digital signatures, only the initiator Alice needs to register with the TTP. After obtaining a certificate C_A from a CA, Alice is required to get a long-term voucher V_A from the TTP by running *Registration Protocol*. That is, the following procedures are executed.

1. Alice first sets an RSA modulus $n = pq$, where p and q are two l-bit safe primes, i.e., there exist two primes p' and q' such that $p = 2p' + 1$ and $q = 2q' + 1$. Then, Alice picks her public key $e \in_R \mathbb{Z}^*_{\phi(n)}$, and computes private key $d = e^{-1} \pmod{\phi(n)}$, where $\phi(n) = (p-1)(q-1)$. Finally, Alice registers her public key with a CA to get her certificate C_A. This step can be omitted if Alice has obtained such a certificate before she registers with the TTP.

2. Alice randomly splits d into d_1 and d_2 such that $d = d_1 + d_2 \pmod{\phi(n)}$. At the same time, she generates a sample message-signature pair (w, σ_w), where $w \in \mathbb{Z}^*_n/\{1, -1\}$, $ord(w) \geq p'q'$, and $\sigma_w = w^{d_1} \pmod n$. Then, Alice sends (C_A, w, σ_w, d_2) to the TTP but keeps (d, d_1, d_2) secretly.

3. The TTP first checks that Alice's certificate C_A is valid. After that, the TTP checks that the triple (w, σ_w, d_2) is prepared correctly. If everything is in order, the TTP

stores d_2 secretly, and creates a voucher V_A which is the TTP's signature on message (C_A, w, σ_w), which guarantees that the TTP can issue a valid partial signature on behalf of Alice by using the secret d_2.

As mentioned in [20], the TTP can check the correctness of the triple (w, σ_w, d_2) with the following techniques. Firstly, the TTP validates that w is an element of order at least of $p'q'$ by checking that $w \in \mathbb{Z}_n^*/\{1, -1\}$, and that both $gcd(w-1, n)$ and $gcd(w+1, n)$ are not prime factors of n. Then, Alice is required to show that she knows the discrete logarithm of σ_w to the base w via a zero-knowledge protocol interactively or non-interactively. Finally, the TTP checks whether $w \equiv (\sigma_w w^{d_2})^e \pmod{n}$. If all those validations pass, the TTP accepts (w, σ_w, d_2) as a valid triple and creates the voucher V_A for Alice.

3.2 Signature Exchange Protocol

Assume that before Alice and Bob begin to sign a contract m they has agreed with it. Furthermore, we suppose that a predetermined but reasonable deadline t and the identities of the participants Alice, Bob and the TTP are explicitly contains in the contract. Then, Alice and Bob perform the following protocol for exchanging signatures.

1. Alice $\rightarrow$ Bob: Alice computes her partial signature $\sigma_1 = h(m)^{d_1} \pmod{n}$. Then, she picks $r \in_R \mathbb{Z}_{\phi(n)}^*$ and $k \in_R \mathbb{Z}_n^*$, and computes $\xi = H_1(h(m)^r \pmod{n}, w^r \pmod{n})$. Alice sends $(C_A, V_A, \sigma_1, \xi, k)$ to Bob.

   Here, $h(\cdot)$ and H_1 are cryptographically secure hash functions.

2. (a) Bob $\rightarrow$ Alice: Upon receiving $(C_A, V_A, \sigma_1, \xi, k)$, Bob picks $c \in_R \{0,1\}^l$. Then, he sends c to Alice.

   (b) Alice $\rightarrow$ Bob: Upon receiving c, Alice performs as follows:
   - Compute e' and d' such that $e' = H_2(\xi, c)$ and $e'd' \equiv 1 \pmod{\phi(n)}$.
   - Compute $\beta = (k \cdot H_3(m))^{d'} \pmod{n}$ and $z = r + cd_1$.

   Alice sends (β, z) to Bob. Here, H_2 and H_3 are cryptographically secure hash functions.

3. Bob $\rightarrow$ Alice: Upon receiving (β, z), Bob computes $e' = H_2(\xi, c)$ and checks whether the following two equations hold:

$$\beta^{e'} \stackrel{?}{=} k \cdot H_3(m) \pmod{n} \quad (1)$$

$$\xi \stackrel{?}{=} H_1(h(m)^z/\sigma_1^c \pmod{n}, w^z/\sigma_w^c \pmod{n}) \quad (2)$$

   Only if Equations (1) - (2) hold and the deadline t specified in contract m is sufficient for applying dispute resolution from the TTP, Bob sends his signature σ_B on contract m to Alice, since he is convinced that another partial signature σ_2 can be released by the TTP, in case Alice refuses to do so.

4. Alice $\rightarrow$ Bob: Upon receiving σ_B, Alice sends her partial signature σ_2 to Bob.

The following is further explanation of our signature exchange protocol. In Step 1, Alice first properly computes $\sigma_1 = h(m)^{d_1} \pmod{n}$, and sends $(C_A, V_A, \sigma_1, \xi, k)$ to Bob. The purpose of Step 2 is that Alice interactively convinces Bob to accept valid σ_1.

Now, with the special properties of RSA signature scheme and Σ-protocol, we prove that any adversary cannot convince Bob to accept an invalid σ_1 as valid with non-negligible probability.

First of all, according to Step 2, e', which is generated after m and k are outputted, is computed from two numbers ξ and c picked by Alice and Bob respectively (c is a challenge number from Bob). That is, m and k are fixed before e' is generated in the viewpoint of Bob. Hence, according to the properties of RSA signature scheme depicted in Section 2.1, only Alice can compute a valid signature β which satisfies Equation (1) on message $k \cdot H_3(m)$, because only Alice can compute d' such that $e'd' \equiv 1 \pmod{\phi(n)}$. Hence, any adversary cannot impersonate Alice with non-negligible probability. Therefore, Bob is convinced that he is performing the protocol with Alice.

Furthermore, (ξ, c, z) is a valid transcript of a Σ-protocol (i.e. Protocol 4 depicted in Section 2.2) for proving equality of discrete logarithms. More precisely, this Σ-protocol used in our protocol is for the relation $\{(x, w)|x = (n, h(m), w, \sigma_1, \sigma_w)$ and $\sigma_1 = h(m)^{d_1} \pmod{n}, \sigma_w = w^{d_1} \pmod{n}\}$. That is, Alice proves to Bob that two logarithms of σ_1 and σ_w with respect to bases $h(m)$ and w are equal by exploiting this Σ-protocol. Since w and σ_w are consisted in V_A which is validated and signed by the TTP, Bob is convinced that the logarithm of σ_1 is d_1 with respect to bases $h(m)$, i.e., σ_1 is a valid partial signature on contract m. Hence, any adversary cannot convince Bob to accept an invalid σ_1 as valid with non-negligible probability.

Based on the above discussion, we conclude that after performing this signature exchange protocol Bob is convinced of the followings: (1) σ_1 is sent from Alice; and (2) σ_1 is a valid partial signature on contract m.

3.3 Dispute Resolution Protocol

In the proposed protocol, *Dispute Resolution Protocol* is the same as Wang's [20]. The detail is recalled as follows.

If Bob has sent his signature σ_B to Alice but does not receive the value of σ_2 or only receives an invalid σ_2 from Alice before the deadline t, he can send the TTP $(C_A, V_A, m, \sigma_1, \sigma_B)$ to apply dispute resolution. Upon receiving Bob's application, the TTP performs as follows:

1. The TTP first verifies whether C_A, V_A and σ_B are Alice's valid certificate, voucher and Bob's signature on contract m, respectively. After that, the TTP checks whether the deadline t embedded in m expires, and whether Alice, Bob and itself are the correct participants specified in m. If any validation fails, the TTP sends an error message to Bob. Otherwise, continue.

2. Then, the TTP computes $\sigma_2 = h(m)^{d_2} \pmod{n}$ and checks whether $h(m) \equiv (\sigma_1\sigma_2)^e \pmod{n}$. If this equality holds, the TTP sends (m, σ_2) to Bob and forwards (m, σ_B) to Alice. Otherwise, the TTP sends an error message to Bob.

For more details about this sub-protocol, please refer to Wang's paper [10].

4 Security

4.1 Completeness and Optimism

Based on the descriptions and discussions presented in last section, we can conclude that if both involved participants are honest and the communication channel is in order, each of the two participants can receive the other's signature correctly, and the TTP is not invoked. Hence, the proposed protocol is *complete* and *optimistic*.

4.2 Abuse-freeness

Now, we prove the *abuse-freeness*. First, after the execution of Step 2 in *Signature Exchange Protocol*, if Bob shows the partial signature σ_1 with the proof (ξ, k, c, β, z) to others, nobody (other than Alice and the TTP) believes that σ_1 is indeed Alice's partial signature on contract m. Because, for any contract m, there exists a simulator which can generate such a proof for any σ_1 as follows:

- Choose random numbers z, c and β.
- Compute $\xi = H_1(h(m)^z/\sigma_1^c \pmod n, w^z/\sigma_w^c \pmod n)$.
- Compute $e' = H_2(\xi, c)$, $k = \beta^{e'}/H_3(m) \pmod n$.

It is clear that (ξ, k, c, β, z) is a valid proof which satisfies Equations (1)-(2). Furthermore, such a simulated proof is computationally indistinguishable from the real proof generated by Alice and Bob together. Therefore, the proposed protocol is *abuse-free*.

4.3 Fairness

To discuss the fairness property which is the most important security property for a fair exchange protocol, two cases are considered: (1) Alice is honest, but Bob is cheating; and (2) Bob is honest, but Alice is cheating. For simplicity, however, the effect of deadline on the fairness is not explained explicitly below as Wang did in [20].

Case 1: *Alice is honest, but Bob is cheating.*

First of all, nobody can generate valid σ_1 except Alice, and that nobody can generate valid σ_2 except Alice and the TTP. The only possible cheating chances for Bob are Step 2 and Step 3. If Bob cheats in Step 3, i.e., he does not send σ_B or only sends an incorrect σ_B to Alice, he cannot get σ_2 from Alice in Step 4. Furthermore, Bob cannot perform *Dispute Resolution Protocol* with the TTP to obtain σ_2, because Alice will receive σ_B from the TTP. Therefore, Bob cannot cheat Alice successfully.

Case 2: *Bob is honest, but Alice is cheating.*

In *Signature Exchange Protocol*, Alice may cheat in any or some of the following steps: Step 1, Step 2 and Step 4.

First of all, according to the specification of *Signature Exchange Protocol*, to get the signature σ_B on contract m from the Bob, Alice has to convince Bob accepting σ_1 as a valid partial signature in Step 2.

Recall that Step 2 is exactly a Σ-protocol for proving that σ_1 is valid. That is, Alice (prover) cannot convince Bob (verifier) to accept an invalid σ_1 as valid with non-negligible

probability. Therefore, we conclude that to obtain σ_B from Bob, Alice has to send valid σ_1 (with valid C_A and V_A) in Step 1 and perform honestly in Step 2. In this situation, Bob receives valid $\sigma_1 = h(m)^{d_1} \pmod n$ from Alice before Alice obtains valid σ_B from Bob. After that, Step 4 is the only one possible cheating chance for Alice, i.e., she may refuse to reveal σ_2 or just send an incorrect σ_2 to Bob. However, Bob can obtain σ_2 from the TTP via *Dispute Resolution Protocol*. Therefore, Alice cannot obtain Bob's signature if Bob does not obtain her signature.

Based on the above analysis, the proposed protocol is not advantageous to any dishonest party. Hence, our contract-signing protocol captures the *fairness* property .

5 Efficiency

In the proposed protocol, the TTP is *stateless* because no state information related to each protocol instance needs be kept in the TTP. The *compatibility* is also achieved, since each participant's commitment is its standard signature on a contract, instead of a signature satisfying some special structures [3, 6, 17].

Table 1 shows the comparison of basic features, security and efficiency between our new protocol and several other solutions. Based on this comparison, we can conclude that our proposed protocol is more efficient than the others.

Table 1: Comparison of Basic Features, Security and Efficiency

	[3]	[18]	[21]	[6]	[20]	ours
Fairness	√	√	√	√	√	√
Abuse-Freeness	×	×	×	×	√	√
Timeliness	√	√	√	√	√	√
Transparent TTP	×	√	√	√	√	√
TTP off-line	√	√	√	√	√	√
Rounds Number	4	3	3	3	7	5
TTP's Statlessness	×	√	×	×	√	√

6 Conclusion

In this paper, based on the standard RSA signature scheme and Σ-protocol, we proposed a new digital contract-signing protocol that allows two potentially mistrusted participants to exchange their digital signatures on a contract in an efficient and secure way. The proposed protocol is not only fair and optimistic, but also efficient and abuse-free. Moreover, compared with Wang's first RSA-based fair and abuse-free contract-signing protocol, the proposed protocol is more efficient.

References

[1] A.Abraham, V.Ewards and H.M.Mathew. A Survey on Optimistic Fair Digital Signature Exchange Protocols. IJCSE, 3(2). pp.821-825. 2011.

[2] N.Asokan, V.Shoup and M.Waidner. Optimistic fair exchange of digital signatures. EUROCRYPT'98. pp.591-606. 1998.

[3] N.Asokan, V.Shoup and M.Waidner. Optimistic fair exchange of digital signatures. IEEE Journal on Selected Areas in Communications, 18(4): 591-606, 2000.

[4] G.Ateniese. Efficient verifiable encryption (and fair exchange) of digital signature. CCS'99. pp.138-146. 1999.

[5] F.Bao, R.H.Deng and W.Mao. Efficient and practical fair exchange protocols with off-line TTP. Proc. of IEEE Symposium on Security and Privacy. pp.77-85. 1998.

[6] F.Bao, G.Wang, J.Zhou and H.Zhu. Analysis and improvement of Micali's fair contract signing protocol. ACISP'04. pp.176-187. 2004.

[7] M.Ben-Or, O.Goldreich, S.Micali and R.L.Rivest. A fair protocol for signing contracts. IEEE Transactions on Information Theory, 36(1). pp.40-46. 1990.

[8] I.Damgård. On Σ-protocols. CPT 2010, v.2.

[9] I.B.Damgård. Practical and provably secure release of a secret and exchange of signatures. Journal of Cryptology, 8(4). pp.201-222. 1995.

[10] R.Deng, L.Gong, A.Lazar and W.Wang. Practical protocol for certified electronic mail. Journal of Network and Systems Management, 4(3). pp.279-297. 1996.

[11] Y.Dodis and L.Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. DRM'03. pp.47-54. 2003.

[12] C.Dwork, M.Naor and A.Sahai. Concurrent zero-knowledge, STOC'98. pp.409-418.1998.

[13] S.Even, O.Goldreich and A.Lempel. A randomized protocol for signing contracts. Communications of the ACM, 28(6). pp.637-647. 1985.

[14] J.Garay, M.Jakobsson and P.MacKenzie. Abuse-free optimistic contract signing. CRYPTO'99. pp.449-466. 1999.

[15] R.Gennaro, T.Rabin and H.Krawczyk. RSA-based undeniable signature. Journal of Cryptology, 13(4). pp.397-416. 2000. A preliminary version of this paper appeared in the proceedings of CRYPTO'97.

[16] O.Goldreich. A simple protocol for signing contracts. CRYPTO'83. pp.133-136. 1984.

[17] S.Micali. Simple and fast optimistic protocols for fair electronic exchange. PODC'03. pp.12-19. 2003.

[18] J.M.Park, E.Chong, H.J.Siegel and I.Ray. Constructing fair exchange protocols for e-commerce via distributed computation of RSA signatures. PODC'03, pp.172-181. 2003.

[19] V.Shoup. Practical threshold signatures. EUROCRYPT'00. pp.207-220. 2000.

[20] G.Wang. An abuse-free fair contract-signing protocol based on the RSA signature. IEEE Transactions on Information Forensics and Security, 2010, 5, (1), pp. 158-168.

[21] G.Wang. Generic non-repudiation protocols supporting transparent off-line TTP. J. Comput. Security. 14(5). pp.441-467. 2006.

[22] J.Zhou and D.Gollmann. A fair non-repudiation protocol. Proc. of the IEEE Symposium on Security and Privacy. pp.55-61. 1996.