Three-round Abuse-free Optimistic Contract Signing With Everlasting Secrecy (Short Paper)

Xiaofeng Chen[1], Fangguo Zhang[2], Haibo Tian[2], Qianhong Wu[3,4], Yi Mu[5], Jangseong Kim[6], Kwangjo Kim[6]

[1] Key Laboratory of Computer Networks and Information Security, Ministry of Education, Xidian University, P.R.China
[2] School of Information Science and Technology, Sun Yat-sen University, P.R.China
[3] Department of Computer Engineering and Mathematics, UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Catalonia
[4] Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, Wuhan University, P.R.China
[5] School of Computer Science and Software Engineering, University of Wollongong, Australia
[6] Department of Computer Science, KAIST, KOREA

Abstract. We introduce the novel notion of Verifiable Encryption of Chameleon Signatures (VECS), and then use it to design a three-round abuse-free optimistic contract signing protocol.

Key words: Verifiable encryption, Chameleon signatures, Contract signing.

1 Introduction

Contract signing is an important part of business transactions. Fairness is a basic requirement for contract signing. However, most of the existing contract signing protocols only focus on the fairness while ignoring the privacy of the players. We argue that the privacy of the players is close related to the fairness. For example, if one player or the trusted third party can reap profits at the expense of the other player by intentionally releasing some useful information related to the contract, then the contract signing protocols cannot achieve the true fairness.

Garay et al. [9] first introduced the notion of abuse-free contract signing, which ensures neither party can prove to others that he is capable of choosing whether to validate or invalidate the contract in any stage of the protocol. To illustrate by example, suppose Bob and Carol are two potential competitors who will sign a contract with Alice. If Alice can convince Carol that Bob would like to sign a contract m with her, she may obtain a better contract m' from Carol. In this sense, a contract signing protocol without the property of abuse-free cannot ensure the fairness for both parties. However, it seems that all the efficient contract signing [1, 2, 4, 7] based on the state-of-the-art technique of verifiable encryption of digital signatures (VEDS) are not abuse-free since VEDS is universal verifiable.

On the other hand, we should consider the misbehavior of the trusted third party in contract signing protocols. Although the third party is (by definition) trusted, it is difficult to find a fully trusted third party in the internet. Asokan et al. [3] and Garay et al. [9] introduced the property of accountability in contract signing, *i.e.*, it can be detected and proven if the third party misbehaved. However, all of the existing contract signing protocols do not consider the following misbehavior of the third party: if the third party can know all the information related a contract such as the contract content and the corresponding signatures of two parties, he may sell this associated commercial secret to an interested party. In this sense, it is unfair for both parties, though the contract signing protocol is fair as defined.

In this paper, we first introduce a novel notion named Verifiable Encryption of Chameleon Signatures (VECS), which can be referred to as a special instance of VEDS. Meanwhile, we use this notion to design an efficient optimistic contract signing protocol, which enjoys the properties of completeness, fairness, abuse-freeness, accountability, and invisibility of the third party. The distinguishing property of our signing protocol is the *everlasting secrecy* about the contract against the third party. That is, the third party cannot know any useful information of the contract in any stage of the protocol, which prevents him from illegally selling the commercial secret to any interested party. Moreover, our exchange protocol is only three-pass in the normal situation and thus much efficient for practical use.

2 Verifiable Encryption of Chameleon Signatures

2.1 Formal Definition

Definition 1. *(Verifiable Encryption of Chameleon Signatures) A secure VECS scheme consists of a five tuple* $(\mathcal{PG}, \mathcal{KG}, \mathcal{SG}, \mathcal{VE}, \mathcal{SR})$.

- **System Parameters Generation** $\mathcal{PG}$**:** *An efficient probabilistic algorithm that, on input a security parameter* k*, outputs the system parameters* SP.
- **Key Generation** $\mathcal{KG}$ **:** *An efficient algorithm that, on input the system parameters* SP*, outputs a secret/public key pair* (sk, pk) *for each user.*
- **Signature Generation** $\mathcal{SG}$**:** *An efficient probabilistic algorithm that, on input a label* L*, the public key* pk_V *of the verifier* V*, the secret key* sk_P *of the prover* P*, a message* m*, and an auxiliary random element* r*, outputs a signature* σ *on the chameleon hash value* $h = Hash(L, m, r, pk_V)$.
- **Verifiable Encryption** $\mathcal{VE}$**:** *A non-interactive protocol between the prover* P *and the verifier* V*. Let* (E, D) *be the encryption/decryption algorithm as well as the public/secret key of a secure public key encryption system. Let* $V_P(E, \sigma, r)$ *denote the output of* V *when interacting with* P *on input* (E, σ, r).
- **Signature Recovery** $\mathcal{SR}$**:** *An efficient deterministic algorithm that, on input the decryption algorithm* D *and the ciphertext* $V_P(E, \sigma, r)$*, outputs a chameleon signature* (σ, r) *on message* m *with respect to the public key* pk_V.

2.2 A Concrete Construction from RSA Signatures

Ateniese has proposed various efficient VEDS schemes [4]. Since a chameleon signature scheme is a general construction, it can naturally be used in the Ateniese's VEDS schemes, which results in various VECS schemes. Note that we should use the key-exposure-freeness chameleon signature schemes [5, 6, 8] in order to avoid the key exposure problem of chameleon hashing.

There are three parties, a prover P, a verifier V, and a trusted third party T in our scheme.

- **System Parameters Generation $\mathcal{PG}$:** Let t and k be security parameters. For $i = 1, 2$, define $n_i = p_i q_i$ with the two safe primes $p_i = 2p_i' + 1$ and $q_i = 2q_i' + 1$ in the set $\{2^{k-1}, \cdots, 2^k - 1\}$, where p_i', q_i' are primes. Let $\mathcal{H}_1 : \{0,1\}^* \to \{0, \cdots, 2^{2k} - 1\}$ and $\mathcal{H}_2 : \{0,1\}^* \to \{0, \cdots, 2^\tau\}$ and $\mathcal{H}_3 : \{0,1\}^* \to \mathbb{Z}_{n_1}$ be three collision-resistant hash functions.
- **Key Generation $\mathcal{KG}$:** For $i = 1, 2$, choose a random prime integer $e_i > 2^t$ which is relatively prime to $\phi(n_i) = (p_i - 1)(q_i - 1)$, and compute d_i such that $e_i d_i = 1 \mod p_i' q_i'$. The public key of P is (n_1, e_1) and his secret key is (p_1, q_1, d_1). The public key of V is (n_2, e_2) and his secret key is (p_2, q_2, d_2). T randomly chooses an element $\tilde{g} \in \mathbb{Z}_{n_1}$ and computes $g = \tilde{g}^2 \mod n_1$. The public key of T is $(g, y = g^x \mod n_1)$ and his secret key is x.
- **Signature Generation $\mathcal{SG}$:** Let L be a label, define $J = \mathcal{H}_1(L)$. To sign a message m, P chooses a random integer $u \in_R \mathbb{Z}_{n_2}^*$ and computes the chameleon hash value $M = J^{\mathcal{H}_2(m)} u^{e_2} \mod n_2$. He then computes the signature $\sigma = \mathcal{H}_3(M)^{d_1} \mod n_1$ on message M.
- **Verifiable Encryption $\mathcal{VE}$:** P and V perform the following protocol:
  1. P computes $\mathcal{C}_1 = (m||u)^{2e_2} \mod n_2$, where $||$ denotes concatenation.
  2. P randomly chooses an integer r and encrypts the chameleon signature σ via the ElGamal encryption scheme with T's public key y. That is, P computes $\mathcal{C}_2 = (K_1, K_2, c, s)$, where $K_1 = \sigma^2 y^r \mod n_1$, $K_2 = g^r \mod n_1$, $c = \mathcal{H}_3(M||y^{e_1 r}||g^r||y^{e_1}||g||(y^{e_1})^t||g^t)$, and $s = t - cr$.
  3. P sends the ciphertext $(\mathcal{C}_1, \mathcal{C}_2)$ to V.

  V firstly decrypts $\mathcal{C}_1$ to obtain the pair (m, u), and then computes $M = J^{\mathcal{H}_2(m)} u^{e_2} \mod n_2$, $W = K_1^{e_1} \mathcal{H}_3(M)^{-2} \mod n_1$, and

$$c' = \mathcal{H}_3(M||W||K_2||y^{e_1}||g||(y^{e_1})^s W^c||g^s K_2^c).$$

  If $c' = c$, V accepts the fact that $\mathcal{C}_2$ is a valid T-verifiable encryption of P's chameleon signature on message m.
- **Signature Recovery $\mathcal{SR}$:** In case of dispute, T can compute $\sigma^2 = K_1 / K_2^x \mod n_1$ and then get σ.

3 Secret Abuse-Free Contract Signing

3.1 Security Model

Asokan et al. [2] presented a formal security model for fair signature exchange, which is also suitable for contract signing. In the optimistic two-party contract

signing, there are two players A and B, and a trusted third party T that acts as a server: it receives a request from a client, updates its internal state and sends a response back to the client. We assume that all participants have secret/public keys which will be specified later.

We assume that communication channels between any two participants are *confidential*, which means that eavesdroppers will not be able to determine the contents of messages in these channels. Moreover, we assume that the communication channel between any player and T is *resilient*. The resilient channel assumption leads to an asynchronous communication model without global clocks, where messages can be delayed arbitrarily but with finite amount of time.

Since the misbehavior of dishonest participants could lead to a loss of fairness, we consider the possible misbehavior of the participants in the contract signing. Firstly, although T is by definition trusted, T may collude with one party to weaken the fairness, or gain some benefits by selling the commercial secret of the contract. Therefore, T must be *accountable* for his dishonest actions, *i.e.*, it can be detected and proven if T misbehaves. Secondly, A or B may reap benefits at the expense of the other party. The abuse-freeness contract signing protocol can only partially solve this problem. For example, a dishonest A can execute the **Abort** protocol after correctly executing the **Exchange** protocol with B [10]. As a result, B obtains A's signature while A obtains B's signature and the abort-token. Trivially, the output of the protocol violates the original definition of fairness. This means that Asokan et al.'s security model is not perfect. The reason is that it does not consider the misbehavior of A and B. Therefore, we should define the accountability of A and B, *i.e.*, it can be detected and proven if A and B misbehaves. Moreover, It can be a part of the agreed contract content for how to punish the dishonest party.

The security properties of contract signing are defined in term of completeness, fairness, abuse-freeness, accountability, T invisibility [2, 9]. Besides, we define a new property named T secrecy. We argue that a contract and the corresponding signatures of two players should be a commercial secret and T cannot reveal it to outsiders for some benefits in any stage of the protocol.

- **Completeness:** It is infeasible for the adversary to prevent honest A and B from successfully obtaining a valid signature (or the non-repudiation token) of each other. The adversary has the signing oracles that can be queried on any message except the contract. The adversary can interact with T, but cannot interfere with the interaction of A and B, except insofar as the adversary still has the power to schedule the messages from A and B to T.
- **Fairness:** We consider a game between an adversary and an honest party. Generally, we let the adversary play the role of the corrupt party, who completely controls the network, arbitrarily interacts with T, and arbitrarily delays the honest party's requests to T. We argue that the misbehavior of the adversary may weaken the fairness. So, if the honest party can provide a proof that the adversary misbehaves, then he has the power to validate or invalidate the contract for the punishment of the adversary. In this sense, the fairness means that it is infeasible for the adversary to obtain the honest

party's signature on a contract, while without allowing the honest party to obtain the adversary's signature or a proof that the adversary misbehaves.
- **Abuse-freeness:** It is infeasible for one party at any point in the protocol to be able to prove an outside party that he has the power to terminate (abort) or successfully complete the contract.
- **Accountability:** It can be detected and proven if any participant misbehaves.
- T **invisibility:** It is infeasible to determine whether T has been involved in the protocol or not.
- T **secrecy:** It is infeasible for T to obtain any useful information about the contract in any stage of the protocol.

3.2 Our Protocol

In this section, we use the proposed VECS to present an efficient abuse-free contract signing protocol. We first give some notations. Let $\mathcal{H}$ be a key exposure free chameleon hash function. Denote by $Sig(SK_X, M)$ the signature on message M with the secret key SK_X of the party $X \in \{A, B, T\}$; Denote by $O_B(E, \sigma_A, PK_T)$ a verifiable encryption of A's signature σ_A under T's public key PK_T. Our abuse-free contract signing protocol has three sub-protocols: **Exchange**, **Abort**, and **Resolve**. In the normal case, only the exchange protocol is executed.

Suppose A and B have agreed on a message $\mathcal{M} = (m, r_A, r_B)$, where m is a common contract and (r_A, r_B) are two random integers. We do not describe this agreement in details here and it may require a number of rounds of communication between A and B through an authenticated channel. Moreover, this agreement should not achieve the non-repudiation property, *i.e.*, neither party should generate any non-repudiation token on the agreed message.

Exchange Protocol

1. A computes the chameleon hash value $h_A = \mathcal{H}(m, r_A, PK_B)$ and the signature $\sigma_A^* = Sig(SK_A, h_A||T)$, where $||$ denotes concatenation. A then computes the ciphertext $\mathcal{C} = O_B(E, \sigma_A^*, PK_T)$ and sends it to B.
2. If $\mathcal{C}$ is invalid, B quits. Otherwise, B computes the signature $\sigma_B = Sig(SK_B, h_B)$ on the chameleon hash value $h_B = \mathcal{H}(m, r_B, PK_A)$ and then sends σ_B to A.
3. If σ_B is invalid, A runs the **Abort** protocol. Otherwise, A computes the signature $\sigma_A = Sig(SK_A, h_A)$ and sends it to B. If σ_A is not valid, B runs the **Resolve** protocol.

Abort Protocol

1. A computes the signature $Sig(SK_A, \text{abort}||\mathcal{C})$ on message "abort$||\mathcal{C}$" and then sends $(\mathcal{C}, Sig(SK_A, \text{abort}||\mathcal{C}))$ to T. If the signature is valid and B has not resolved, T issues an abort-token $\mathcal{AT} = Sig(SK_T, Sig(SK_A, \text{abort}||\mathcal{C}))$ to A and stores it. The abort token is not a proof that the exchange has been aborted, but a guarantee by T that it has not and will not execute the **Resolve** protocol.

2. If B has resolved, T sends A the stored value $\hat{\sigma_B}$ in the **Resolve** protocol.

Resolve Protocol

1. B firstly sends T the triple $(\mathcal{C}, h_A, \hat{\sigma_B})$, where $\hat{\sigma_B} = Sig(SK_B, \text{resolve}||A||h_A)$ denotes the *resolved* signature of B. Generally, it is no difference with an ordinary signature $Sig(SK_B, \text{resolve}||A||h_A)$ of B on message "resolve$||A||h_A$". Additionally, it also denotes $Sig(SK_B, m)$ on condition that only A can provide a pair (m, r_A) which satisfies $h_A = \mathcal{H}(m, r_A, PK_B)$.
2. If A has aborted, T then sends the abort-token $\mathcal{AT}$ to B. Else, if $\mathcal{C}$ is a valid T-verifiable encryption of A's signature on message h_A and $\hat{\sigma_B}$ is valid, T decrypts $\mathcal{C}$ to obtain σ_A^* and sends it to B.
3. T stores the value $\hat{\sigma_B}$.

3.3 Misbehavior in the Protocol

Since the set of the possible output for A and B is $\{\sigma_B, \hat{\sigma_B}, \mathcal{AT}\}$ and $\{\sigma_A, \sigma_A^*, \mathcal{AT}\}$, respectively. Therefore, the possible output of our proposed protocol is as follows:

- Case 1: A obtains σ_B and B obtains σ_A. This means that both parties are honest.
- Case 2: A obtains σ_B and B obtains σ_A^*. This means that B successfully runs the **Resolve** protocol at some point after sending σ_B.
- Case 3: A obtains $\hat{\sigma_B}$ and B obtains σ_A^*. This means that A has already sent $\mathcal{C}$ to B, and then B runs the **Resolve** protocol before A aborted.
- Case 4: Both A and B obtain $\mathcal{AT}$. This means that A has already sent $\mathcal{C}$ to B, and then runs the **Abort** protocol at some point before B resolved.
- Case 5: A obtains σ_B and B obtains $\mathcal{AT}$. This means that A has received σ_B and then runs the **Abort** protocol before B resolved. If this case happens, we claim that A misbehaves in the protocol.
- Case 6: A obtains $\mathcal{AT}$ and B obtains σ_A. This means that A runs the **Abort** protocol after sending σ_A to B. If this case happens, we also claim that A misbehaves.
- Case 7: A obtains $\mathcal{AT}$ and B obtains σ_A^*. This means that both A and B successfully runs the **Abort** and **Resolve** protocol, respectively. If this case happens, we claim that the T misbehaves.
- Case 8: A obtains $\hat{\sigma_B}$ and B obtains $\mathcal{AT}$. Due to the fact that B obtains the abort-token only when A has obtained the abort-token, this case will not happen if the T is honest. Therefore, we also claim that the T misbehaves in this case.
- Case 9: A obtains $\hat{\sigma_B}$ and B obtains σ_A. If this case happens, we claim that A misbehaves because B cannot obtain σ_A unless A has obtained σ_B successfully.

If the first four cases occur, the protocol achieves the fairness since both parties obtain either the signature of each other, or the abort token. Since the chameleon signature is not universal verifiable, σ_B means nothing if B does not

perform the denial protocol of chameleon signatures. On the other hand, A is not allowed to run the **Abort** protocol after having received σ_B. Similarly, A is not allowed to run the **Abort** protocol after sending σ_A to B. Moreover, A should never send σ_A to B unless A has obtained σ_B successfully. That is, if the case 5, or case 6, or case 9 occurs, it is a proof that A misbehaves. If the case 7 or 8 occurs, then T must be accountable for his misbehavior.

4 Security Analysis of the Contract Signing Protocol

Due to the properties of non-repudiation and non-transferability of chameleon signatures, the proposed contract signing protocol satisfies the completeness and abuse-freeness, respectively. Also, as discussed in section 3.3, it is trivial that the proposed contract signing protocol satisfies the accountability. Due to the space consideration, we only focus on the fairness, T invisibility and T secrecy.

Theorem 1. *The proposed contract signing protocol satisfies the property of fairness.*

Proof. We first prove the fairness for A. Consider an honest A playing against a dishonest B. We say that B wins the game if and only if either B obtains σ_A while A does not obtain σ_B, or B obtains σ_A^* while A obtains neither σ_B nor $\hat{\sigma_B}$. Assume A does not obtain σ_B, A must run the **Abort** protocol at some point after sending $\mathcal{C}$ to B and thus B cannot obtain σ_A. If B does not run the **Resolve** protocol before A aborted, then both parties obtain the abort-token $\mathcal{AT}$. Else, B can obtain σ_A^* from the T. However, it ensures that A can also obtain $\hat{\sigma_B}$ from T. Therefore, the successful probability for B to win the game is negligible.

We then prove the fairness for B. Consider an honest B playing against a dishonest A. We say that A wins the game if and only if either A obtains σ_B while B obtains neither σ_A nor σ_A^*, or A obtains $\hat{\sigma_B}$ while B does not obtain σ_A^*. Firstly, we argue if A obtains $\hat{\sigma_B}$, then B must obtain σ_A^* unless the T is dishonest. Secondly, assume B does not obtain σ_A, so B must run the **Resolve** protocol at some point after sending σ_B to A. If A does not run the **Abort** protocol before B resolved, then B can obtain σ_A^* from the T. Else, B can obtain the abort-token $\mathcal{AT}$. However, it is a proof that A misbehaves in the protocol and A must be accountable for this. Therefore, the successful probability for A to win the game is negligible. □

Theorem 2. *The proposed contract signing protocol satisfies the property of T invisibility and T secrecy.*

Proof. Note that the distribution of σ_A and σ_A^* is computationally indistinguishable. Similarly, the distribution of σ_B and $\hat{\sigma_B}$ is also computationally indistinguishable. Therefore, it is impossible to determine whether T has been invoked in the protocol or not. On the other hand, note that the message $\mathcal{M} = (m, r_A, r_B)$ is agreed beforehand and never revealed in any stage of the protocol. Moreover, the chameleon signature is not universal verifiable. Therefore, T cannot obtain any useful information about the contract in the protocol. □

5 Conclusions

In this paper, we first introduce the notion of Verifiable Encryption of Chameleon Signatures (VECS). We then use the notion to design a secret abuse-free optimistic contract signing protocol, which is only three-pass in the normal situation. Moreover, we prove that our protocol achieves the desired security properties.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (No. 60970144, 60773202, 60970114, 60970115, 60970116), Guangdong Natural Science Foundation (No. 8451027501001508), Program of the Science and Technology of Guangzhou, China (No. 2008J1-C231-2).

References

1. Asokan, N., Shoup, V. and Waidner, M., 1998. Asynchronous protocols for optimistic fair exchange. IEEE Symposium on Security and Privacy, pp.86-99.
2. Asokan, N., Shoup, V. and Waidner, M., 1998. Optimistic fair exchange of digital signatures. Eurocrypt 1998, LNCS 1403, Springer, pp.591-606.
3. Asokan, N., Shoup, V. and Waidner, M., 2000. Optimistic fair exchange of digital signatures. IEEE Journal on Selected Areas in Communications, 18(4), pp.593-610.
4. Ateniese, G., 2004. Verifiable encryption of digital signatures and applications. ACM Transaction on Information and System Security, 7(1), ACM Press, pp.1-20.
5. Ateniese, G., and de Medeiros, B., 2004. Identity-based chameleon hash and applications. FC 2004, LNCS 3110, Springer, pp.164-180.
6. Ateniese, G., and de Medeiros, B., 2005. On the key-exposure problem in chameleon hashes. SCN 2004, LNCS 3352, Springer, pp.165-179.
7. Bao, F. Deng, R., Mao, W., 1998. Efficient and practical fair exchange protocols with off-Line TTP. IEEE Symposium on Security and Privacy, pp.77-85.
8. Chen, X., Zhang, F., and Kim, K., 2004. Chameleon hashing without key exposure. ISC 2004, LNCS 3225, Springer, pp.87-98.
9. Garay, J.A., Jakobsson, M., and MacKenzie, P., 1999. Abuse-free optimistic contract signing. Crypto 1999, LNCS 1666, Springer, pp.449-466.
10. Shmatikov, V. and Mitchell, J.C., 2001. Analysis of abuse-free contract signing. FC 2000, LNCS 1962, Springer, pp.174-191.