*I.J. Information Technology and Computer Science,* **2017, 5, 9-14**
Published Online May 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2017.05.02

# Abuse-Free Optimistic Contract Signing Using RSA for Multiuser Systems

**Santosh Bharadwaj Rangavajjula**
School of computing, Blekinge Tekniska Högskola, Sweden
E-mail: sara16@student.bth.se

**Tristan Claverie**
School of computing Blekinge Tekniska Högskola, Sweden
E-mail: trcl16@student.bth.se

***Abstract***—Multi-party contract signing (MPCS) is a way for signers to agree on a predetermined contract by exchanging their signature. This matter has become crucial with the growing number of communications. In this paper, we focus mainly on studying the state of the art protocols and more specifically the cryptography involved. We identify the major advances in MPCS, highlight a few gaps with the current protocols and propose an algorithm for contract signing to be abuse-free, optimistic for many signers in industrial standards.

***Index Terms***—Contract Signing, Abuse-free, optimistic, multi-user, RSA.

## I. Introduction

### A. Context

Signing a paper contract between two users may be very simple, but signing a paper contract be- tween 2 000 signers is very difficult. The same problematic arise if the contract involves only 10 people scattered around the globe. In such cases, the ability to sign a contract using a computer becomes very handy. One thing necessary for such a protocol is *fairness*, that is signer A can not get the signature of any other honest signer unless signer A has committed to the contract. An easy way to solve this problem would be for every signer to send his signature to a *Trusted Third Party* (TTP), then the TTP sends back a fully signed contract to everyone. However, by doing this, the TTP would become a necessary element of any contract signature and would ends up being a bottleneck. Another approach is to share the load between all signers and to refer to the TTP only in case of problem during the signature. This kind of MPCS is called *optimistic*.

### B. Background

We are interested in the notion of *abuse-freeness* as defined in [1]. A protocol is abuse-free if no group of signer can prove that he holds the power to complete or abort the contract signature. Garay et al. introduce a new cryptographic object called *Private Contract Signature* in [1] based on ElGa- mal cryptosystem [2] and use it to define a two- party contract signing and a three-party contract signing. These constructions are proven to be fair, optimistic and abuse-free. In [3] an optimistic protocol for exchanging fairly signatures was proposed by Asokan et al.. Mukhamedov et al. proposed another optimistic MPCS for any number of signers in [4] also base on private contract signatures. Wang proposed an abuse-free, optimistic two-party contract signing in [5] using RSA and trapdoor commitment schemes [6]. Kordy et al. proved an equivalence between a mathematical sequence and the fairness of an MPCS protocol in [7], it is based on private contract signature. The obtained protocol is abuse-free, optimistic, fair and efficient because it reaches the lower bounds in terms of bandwidth and message complexity determined by Garay et al. in [8]. In [9] Mauw et al. extend the work of Kordy et al. using a labeled DAG instead of a linear sequence, and achieving as well an optimistic abuse- free fair and efficient MPCS.

### C. Objectives

A common point to optimistic MPCS is that at some point, a commitment is exchanged before sending the signature. This paper focuses on finding an MPCS protocol which is abuse-free, optimistic and fair for any number of signers. A secondary objective is its efficiency, *i.e.* it has to be usable in practice without heavy computation and should stick to the RSA industry standard.

### D. Results

In this paper, we find an alternative to Private Contract Signatures [1] using the newly discovered Certificate-based verifiably encrypted RSA signature scheme defined in [10]. We then propose an variation of the MPCS defined in [7] using this new scheme.

## II. Related Work

We conducted a systematic review about optimistic abuse-free contract signing. Juan et el [1] used El Gamal on the Private Contract Signature object. Although the protocol [1] proposed is fair, optimistic and abuse-free, it is applicable for a maximum of 3 signers. Also, El Gamal is not today's industry standard. Barbara et el. [7]

 

converted a sequence of signers to a protocol specification, which made protocols be handled equally by Trusted Third Party (TTP). Also, Barbara et el. [7] did not prove the abuse-free property. Juan et el. [8] proposed a protocol that works for more than three signers. The problem with article [8] is that it was proven unfair if there are more than four signers. Sjouke et el. proposed a protocol, but the abuse-free property was not established. Also, the author uses El Gamal, which is not today's industry standard. In [11], the author used BLS instead of El Gamal encryption scheme and proposed a protocol which is fair, abuse-free and also optimistic. The shortcoming in article proposed by Gao et el. is that it is only applicable for two signers. Wang et el. [5], used trapdoor commitment scheme with RSA keys to propose the protocol with abuse-freeness, optimism and fairness but it is only applicable for two signers. Villar et el. [12] used partial signature scheme with a variant of Boneh-Boyen which is also fair, abuse- free and optimistic but applicable only for two signers. Juan et el.,Wang et el.,Gao et el.,Li et el., Chen et el., Villar et el., [1], [5], [8], [11], [13], [14], [12] have successfully proven existence of abuse- freeness and therefore we can say that contract signing protocols can be be abuse-free. Juan et el., Barbara et el., Sjouke et el.,Li et el.,Villar et el, [1], [8], [7], [9], [13], [12] in their articles used Private contract signature objects which have limitations either concerning with properties or number of signers, therefore Private Contract signature may not be a solution for us. From articles written by Juan et el. and Li et el., [8], [13] we can say that contract signing protocols can have properties like abuse-freeness, optimism, fairness and can also have multiple signers but if number of signers are greater than or equal to four in Juan et el.'s [8] article, the fairness property might be broken. Also in Li et el.'s article [13] El Gamal is used which is not an industry standard. From explanations given by Wang et el. and Villar et el. [5], [14] we can say that RSA is compatible with some contract signing mechanisms used today and that gives us confidence in our research. We mainly focused on the different cryptographic primitives used rather than the protocols or their efficiency. In table 1, we summarized the results of reviewed literature. We highlighted some gaps in the current protocols, and we will strive to address one in further research.

Table 1. Results overview

| Reference papers | DEQ1 | DEQ2 | DEQ3 | DEQ4 |
|---|---|---|---|---|
| **Ref**[1] | Building an optimistic abuse-free fair MPCS, and definition of a new cryptographic object | The protocol is defined for 2 and 3 signers | ElGamal | Private Contract Signature |
| **Ref**[5] | Optimistic abuse-free fair MPCS | Only for 2 signers | RSA | Trapdoor Commitment Scheme |
| **Ref**[7] | Optimistic abuse-free fair MPCS. Equivalence between a sequence and an MPCS | Abuse-freeness not proven | ElGamal | Private Contract Signature |
| **Ref**[8] | Building an optimistic abuse-free fair MPCS, for n > 3 signers | Proven to be unfair for $n \geq 4$ | ElGamal | Private Contract Signature |
| **Ref**[9] | Optimistic abuse-free fair MPCS. Equivalence between a labeled DAG and an MPCS | Abuse-freeness not proven | ElGamal | Private Contract Signature |
| **Ref**[11] | First optimistic abuse-free fair MPCS using BLS signature | Two signers | BLS | non-interactive proof of knowledge |
| **Ref**[13] | Optimistic abuse-free fair MPCS for any number of signers. Modeling and analysis of MPCS | - | ElGamal | Private Contract Signature |
| **Ref**[15] | Optimistic abuse-free fair MPCS | Only for 2 signers | RSA | Verifiable encryption of Chameleon Signature |
| **Ref**[14] | Optimistic abuse-free fair MPCS using a variant of Boneh-Boyen signature | Only for 2 signers | A variant of Boneh-Boyen | Partial Signature Scheme |
| **Ref**[12] | Optimistic abuse-free fair MPCS. Defines a framework for the commitment | Only for 2 signers | - | Ordinary Crisp Commitment Scheme |

## III. Definition of the Commitments Used

### A. Private Contract Signature

Private Contract Signature[1] have been used in several MPCS protocol as a way to ensure optimism and abuse-freeness. The idea is that they are designated-verifier signatures that can be converted into a universally-verifiable signature either by the party who issued it or by the TTP. More formally, a Private Contract Signature is a set of 6 polynomial-time algorithms *PCS-Sign*, *S-Convert,*

*TP-Convert, PCS-Ver, S-Ver, TP-Ver* defined as follow:

- **PCS-Sign** executed by party *A* on message *m* for party *B* with respect to third party *T*, denoted $PCS - Sign_A(m, B, T)$, outputs a private contract signature $PCS_A(m, B, T)$.This private contract signature can be verified using PCS-Ver.
- **PCS-Ver** executed by B and denoted $PCS - Ver(m, A, B, T, S)$ outputs *true* if $S = PCS_A(m, B, T)$ and false otherwise.
- **S-Convert** executed by A on contract signature $S = PCS_A(m, B, T)$ generated by A, denoted $S - Convert_A(S)$, produces a universally verifiable signature by A on *m*, $S - Sig_A(m)$.
- **TP-Convert** executed by T on a private con- tract signature $S = PCS_A(m, B, T)$, denoted $TP - Convert_T(S)$, produces a universally- verifiable signature by A on *m*, $TP - Sig_A(m)$.
- **S-Ver** executed by any party and denoted $S - Ver(m, A, T, S)$ outputs *true* if $S = S - Sig_A(m)$ and *false* otherwise.
- **TP-Ver** executed by any party and denoted $TP - Ver(m, A, T, S)$ outputs *true* if $S = TP - Sig_A(m)$ and false otherwise.

The implementation of the Private Contract Sig- nature scheme proposed in [1] is defined using Elgamal signature scheme. We sketch the idea here: Signer A produces a cipher text by signing contract *m* with his private key and encrypting it with TTP's public key. He then produces a *proof of cipher text content* which assert that the cipher text is a valid signature. He finally makes this proof a designated- verifier proof so that only B can read it. The other four algorithms *S-Convert, TP-Convert, S-Ver, TP-Ver* are trivial and do not need explanations. One of the property of the implementation given is that $S - Sign$ and $TP - Sign$ are equivalent, and therefore $S - Ver$ and $TP - Ver$ are too.

### B. Certificate-based verifiably encrypted signature

We willuse the notion of certificate-based verifiably encrypted signature (CBVES) as defined in [10]. It has been introduced in order to produce an optimistic fair exchange protocol, MPCS proto- cols being a subfield of the broader fair exchange problem. This object is a set of 8 algorithms that we sketch here. It relies on a certificate-based cryptography and thus needs a certificate authority CA.

- **KeyGen** Generates a key pair (SK, PK) for a signer along with the system parameters Params
- **CKeyGen** Generates the CA key pair (BSK, CPK)
- **CertCreate** A signer asks the CA to get a public key certificate CERT. The CA, after authenticating the signer, creates a certificate information CI and a public key certificate using Params. The CA sends (CI, CERT) to the signer, encrypted with the signer's public key
- **VesCreate** From a message *m*, a signer outputs a verifiably encrypted signature (m, CI, $\omega$) by using his secret key SK and his certificate CERT. The message m contains a statement, that it is valid only if extracted from a verifiably encrypted signature by the signer or by CA if the recipient fulfills his obligation.
- **VesVer** The recipient checks the validity of CI and validates the verifiably encrypted signature (m, CI, $\omega$)
- **Sign** The signer creates an ordinary signature (m, $\sigma$) with his private key SK
- **Ver** Validates an ordinary signature (m, $\sigma$) using the public key PK
- **Adjudication** From a valid verifiably en- crypted signature (m, CI, $\omega$), the CA checks the validity of CI. If valid, the CA outputs the ordinary signature (m, $\sigma$)

We refer the reader to [10] for implementation details.

### C. From CBVES to PCSS

This two schemes, though defined differently have a lot in common. In fact, we will show how to get from CBVES to PCS. We will first address the problem of the signatures which are not the same for each scheme.

In [1], output signatures are universally verifiable signatures while in [10] they are ordinary signatures. Universally verifiable signatures can be verified across the Internet, while ordinary signatures can only be verified in the sub-network formed by all the parties and the TTP. However, it is possible to address this problem as shown in [16] where the basic idea is to convert signatures made in ad-hoc networks to universally verifiable signature.

The two schemes involve a trusted party, which is the TTP for PCS and the CA for CBVES. They both have the ability to convert a cipher text issued by a signer into a signature, thus the trusted party of CBVES is an extension of PCS's TTP because it also has the power to issue certificates.

The main difference between the two protocols is that the CA's adjudication is valid only if a certain obligation of the recipient is fulfilled, while TTP's resolve is always valid. This limitation can be easily overcome by choosing an obligation that is always true.

Now, in order to use CBVES as PCS in a protocol, a registration step must be added so that the CA issues the certificates used in the protocol.

## IV. Protocol

We now define an MPCS protocol using the previously seen CBVES scheme. It is based on the same assumptions used for the RSA implementation of CBVES.

### A. Discrete logarithm

A probabilistic algorithm *A* is said be able to $(t, s) - break$ the Discrete Logarithm problem if *A* runs in time at most *t* and outputs the discrete logarithm $DL_p(g^a) = a$ on

input ($g$, $p$, $g^a mod p$) with probability at least $s$, where $a$ is chosen uniformly in $Z_p - 1$.

The DL assumption means that no such algorithm can break the DL problem with reasonable time $t$ and probability $s$.

### B. Computational Diffie-Hellman

A probabilistic algorithm $A$ is said to be able to ($t$, $s$) − *break* the Computational Diffie- Hellman problem if $A$ runs in at most $t$ and outputs $g_1^{ab} mod n$, on input ($g$, $p_1$, $g^a mod p_1$) and ($g_1$, $n$, $g_1^a$ $mod n$, $g_1^b$ $mod n$) with probability at least $s$ and $a$, $b$ chosen uniformly from $Z_p - 1$. The CDH assumption means that no such algorithm can break the CDH problem with reasonable timt $t$ and prob- ability $s$.

### C. RSA problem

A probabilistic algorithm $A$ is said to ($t$, $s$) − *break* the RSA problem is $A$ runs in time at most $t$ and outputs the e-th root of $a$ on input ($e$, $n$, $a$) with probability at least $s$ where $a$ is chosen uniformly from $Z^*_n$.The RSA assumption means that no such algorithm can break the RSA problem with reasonable time $t$ and probability $s$.

### D. Random oracle model

The random oracle model means that concrete objects such as hash functions can be treated as random object. That is, when applying an hash function to $m \in \{0, 1\}^*$, the answer can be interpreted as coming from an oracle which returns a random answer for each new query.

## V. MCPS PROTOCOL

The MPCS protocol we formulate is derived from [7]. We modify it in order to use CBVES instead of PCS, thus creating an optimistic MPCS protocol. We believe it to be abuse-free though it is not proven. It simply consists of three steps. In the first step, all signers register to the certificate authority. In the second, each signer sends several CBVES to every other signer for several round. The third step consists of each signer sending his signature to every other signer. If there is a problem during the protocol, certificate authority is contacted for adjudication.

### A. Definitions

For our protocol, we use the following definitions.

1) A: The set of signers ( e.g. A = {a, b, c}).
2) $\sigma = (\sigma_1, \sigma_n, \ldots, \sigma n)$ : A sequence over A.
3) $\sigma$: The reverse sequence of $\sigma$: $\sigma = (\sigma_n, \sigma_{n-1}, \ldots, \sigma_1)$.
4) $prev_\sigma$ (i): The previous apparition of element $\sigma_i$ in $\sigma$ or 0 if there is none.
5) $next_\sigma$ (i): The next apparition of element $\sigma_i$ in $\sigma$ or n if there is none.
6) $lmpi(\sigma_1, \sigma_2, \ldots, \sigma_l)$: The set of first appearance indices in $\sigma$. For example, lmpi(a, b, a, c, b) = (0, 1, 3)
7) $rmpi(\sigma_1, \sigma_2, \ldots, \sigma_l)$: The set of last ap- pearance indices in $\sigma$. For example, rmpi(a, b, a, c, b) = (2, 3, 4).
8) T: Certificate authority involved in the proto- col.
9) $V_a$((m, i), b, T ): Certificate-based verifiably encrypted signature of message (m, i) issued by signer a for signer b with respect to certificate authority T .
10) $S_a$(m): Universally verifiable signature of message m by signer a.
11) SigSet: The set of positions inside $\sigma$ where the current signer should issue a signature.

### B. Registration

Certificate authority $T$ generates the root certificate, then awaits for every signer in $A$ to ask for a certificate. When this is done, $T$ sends to every signer in $A$ all the public keys. We consider the channels of communication to be secure.

### C. Main protocol

The main protocol is run by signers. If a problem occurs, every signer calls the resolve protocol for adjudication.

**Algorithm 1** Main protocol

**Input :** $\sigma, SigSet, m, T$

1: **for all** $i \in [|\sigma|]$ **do**
2: **for all** $j' \in rmpi(\sigma_{prev(i)+1}, \ldots, \sigma_{i-1})$ **do**
3: $j \leftarrow prev(i) + j'$
4: **if** $j \notin SigSet$ **then**
5: **print** "$receive_{\sigma_i}(\sigma_j, V_{\sigma_j}((m, j), \sigma_i, T))+ Res(m, prev(i))$"
6: **else**
7: **print** "$receive_{\sigma_i}(\sigma_j, S_{\sigma_j}(m)) + Res(m, prev(i))$"
8: **end if**
9: **end for**
10: **for all** $j' \in lmpi(\sigma_{i+1}, \ldots, \sigma_{next(i)-1})$ **do**
11: $j \leftarrow i + j'$
12: **if** $i \notin SigSet$ **then**
13: **print** "$send_{\sigma_i}(\sigma_j, V_{\sigma_i}((m, i), \sigma_j, T))$"
14: **else**
15: **print** "$send_{\sigma_i}(\sigma_j, S_{\sigma_i}(m))$"
16: **end if**
17: **end for**
18: **if** $next(i) = |\sigma| + 1$ **then**

 

```
        i_0 ← min{j ∈ SigSet|σ_i = σ_j}
        for all P ∈ A\{σ_{i_0}, ..., σ_{|σ|}} do
            print "send_{σ_i}(P, S_{σ_i}(m))"
        end for
        for all Q ∈ A\{σ_j|j ∈ SigSet ∧ j ≤ i}
        do
            print "receive_{σ_i}(Q, S_Q(m)) +
            Res(m, i)"
        end for
    end if
end for
```

### D. Resolve protocol

The resolve protocol is ran by certificate authority *T*. A signer shall provide evidence when calling *T*, that is the set of all the messages he has received so far. The certificate authority processes this evidence to see whether a signer is dishonest or not, that is if he has followed the main protocol or not. This protocols outputs a fully signed contract or an abort token. The intuition of this algorithm is that *T* keeps track of all the dishonest signers so far and populates it based on the evidence he receives.

**Algorithm 2** Resolve protocol

**Input :** $P, History, m, \sigma, A, i$
**Output :** $decision_m$

```
if decision_m =⊥ then
    Evidence_m ← ∅
    I_m ← ∅
    Dishonest_m ← ∅
    decision_m ← "abort"
end if
if P ∈ Dishonest_m ∨ ∃j ∈ I_m : V = σ_j ∨
History ≠ p^i_{σ_i} ∪ {V_{σ_i}((m, i), P, T)} then
    Dishonest_m ← Dishonest_m ∪ {P}
    return "abort"
end if
I_m ← I_m ∪ {i}
Evidence_m ← Evidence_m ∪ {History}
if i < |A| then
    return decision_m
else
    if decision_m ≠ "abort" then
        return decision_m
    else
        l ← max(I_m)
        for all j ∈ I_m, j < l do
            if σ_j ∈ {σ_{j+1}, ..., σ_l} then
                Dishonest_m ← Dishonest_m ∪
                {σ_j}
            end if
        end for
        if P ∈ Dishonest_m then
            return decision_m
        end if
        if ∃j ∈ I_m : σ_j ∉ Dishonest_m ∧ σ_j ≠
        P then
            return decision_m
        else
            decision_m ← {S_Q(m)|Q ∈ A}
            return decision_m
        end if
    end if
end if
```

### E. Fairness

As showed in [7], a sufficient (but not necessary) condition for fairness of the above protocol is that the signing sequence *σ* is complete over *A*. The optimism is induced by the fact that the certificate authority *T* plays no other role in the protocol than issuing certificates if no problem occurs. We do not show abuse-freeness though we believe it is. The intuition behind is that only the designated verifier can verify if the commitment he's received if a valid certificate-based verifiably encrypted signature.

In practical applications, the load of the certificate authority can be even more reduced if we consider that it is online at all times and if signers keep their certificate between two signatures.

## VI. Limitations

- Abuse-freeness of the new protocol has not been proven.
- Some paper might be invalidated in the future due to a flaw discovered in the commitments used.
- Some paper might be invalidated due to a flaw in the protocol not yet discovered.
- We might have excluded papers that could be relevant to our research, proceeding to an exclusion bias.

The validity threats are as follow:

- The protocol we defined proved to be not fair or optimistic.
- Cryptosystems used and specially RSA proved to be vulnerable
- Assumptions made (Dl, CDH, RSA, random oracle) proved to be wrong

Mitigating Bias:

- Bias is an un avoidable problem in research. As it cannot be completely removed but can be mitigated.
- From the literature RSA is assumed to be a fool proof algorithm but it may so happen that a future invalidation effects these studies.
- The bias in the research articles of articles generally tends to be lower and so, we have only considered peer-reviewed articles.

## VII. Conclusion

In this paper, we conducted a systematic review about optimistic abuse-free contract signing. In table 1 we summarized the relevant work in this field along with the major results. We mainly focused on the different cryptographic primitive used rather than the protocols or their efficiency.

More specifically, we have found protocols that guarantee fairness, optimism and abuse-freeness for any

number of signers. We have found contract signing protocols respecting all those properties using RSA, but these are yet limited to two signers. Therefore, those protocols are not appropriate to a real-world application and therefore proposed our algorithm that is abuse-free, optimistic, can have many signers and uses the industry standard which is RSA.

## REFERENCES


[1] Juan A. Garay, Markus Jakobsson, and Philip D. MacKen- zie. Abuse-free optimistic contract signing. In Michael J. Wiener, editor, *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceed- ings*, volume 1666 of *Lecture Notes in Computer Science*, pages 449–466. Springer, 1999.

[2] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[3] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18(4):593–610, 2000.

[4] Aybek Mukhamedov and Mark Ryan. Improved multi-party contract signing. In Sven Dietrich and Rachna Dhamija, editors, *Financial Cryptography and Data Security, 11th International Conference, FC 2007, and 1st International Workshop on Usable Security, USEC 2007, Scarborough, Trinidad and Tobago, February 12-16, 2007. Revised Selected Papers*, volume 4886 of *Lecture Notes in Computer Science*, pages 179–191. Springer, 2007.

[5] G. Wang. An abuse-free fair contract-signing protocol based on the rsa signature. *IEEE Transactions on Information Forensics and Security*, 5(1):158–168, 2010. cited By 24.

[6] M. Fischlin. *Trapdoor Commitment Schemes and Their Applications*, 2001. cited By 22.

[7] Barbara Kordy and Saša Radomirovi ć. Constructing optimistic multi-party contract signing protocols. In Stephen Chong, editor, *25th IEEE Computer Security Foundations Symposium, CSF 2012, Cambridge, MA, USA, June 25-27, 2012*, pages 215–229. IEEE, 2012.

[8] Juan A. Garay and Philip D. MacKenzie. Abuse-free multi-party contract signing. In Prasad Jayanti, editor, *Distributed Computing, 13th International Symposium, Bratislava, Slavak Republic, September 27-29, 1999, Proceedings*, volume 1693 of *Lecture Notes in Computer Science*, pages 151–165. Springer, 1999.

[9] Sjouke Mauw and Sasa Radomirovic. Generalizing multi-party contract signing. In Riccardo Focardi and Andrew C. Myers, editors, *Principles of Security and Trust - 4th International Conference, POST 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015, Proceedings*, volume 9036 of *Lecture Notes in Computer Science*, pages 156–175. Springer, 2015.

[10] Z. Shao and Y. Gao. Certificate-based verifiably encrypted rsa signatures. *Transactions on Emerging Telecommunica- tions Technologies*, 26(2):276–289, 2015. cited By 0.

[11] W. Gao, F. Li, and B. Xu. An abuse-free optimistic fair exchange protocol based on bls signature. volume 2, pages 278–282, 2008. cited By 3.

[12] A.A. Al-Saggaf and L. Ghouti. Efficient abuse-free fair contract-signing protocol based on an ordinary crisp commitment scheme. *IET Information Security*, 9(1):50–58, 2015. cited By 0.

[13] X. Li, Z. Wang, L. Chen, and Q. Wang. A multi-party contract signing protocol and its formal analysis in strand space model. volume 3, pages 556–559, 2009. cited By 0.

[14] S. Heidarvand and J.L. Villar. A fair and abuse-free contract signing protocol from boneh-boyen signature. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6711 LNCS:125–140, 2011. cited By 1.

[15] X. Chen, F. Zhang, H. Tian, Q. Wu, Y. Mu, J. Kim, and K. Kim. Three-round abuse-free optimistic contract signing with everlasting secrecy. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6052 LNCS:304–311, 2010. cited By 4.

[16] Kyungkeun Lee, Joonghyo Oh, and Sangjae Moon. How to generate universally verifiable signatures in ad-hoc networks, 2005.


## Author's Profiles

**Santosh Bharadwaj Rangavajjula** Santosh Bharadwaj is a Master's student in Computer Science at Blekinge Tekniska Högskola and graduated Bachelor's of Technology in Computer Science at Jawaharlal Nehru Technological University, Hyderabad, India. Currently, he is a research intern at IFS AB (Industrial and Financial Systems) located in Göteborg, Sweden. His research interests include Artificial Intelligence, Internet of Things, Decision Support Systems, cloud computing and Deep learning.

**Tristan Claverie** Tristan is a graduate student in Computer Sciences at INSA de Rennes, France and is an Erasmus exchange student in Blekinge Tekniska Högskola, Sweden. Tristan is passion- ate about programming, algorithms and new technologies since he was 17 years old. His projects include Implementation of a secure decentralized contract signing protocol, twitter trend analysis and web application for fetching me-teor data. Contract signing, Network security, Algorithm design are his fields of interest.

**How to cite this paper:** Santosh Bharadwaj Rangavajjula, Tristan Claverie,"Abuse-Free Optimistic Contract Signing Using RSA for Multiuser Systems", International Journal of Information Technology and Computer Science(IJITCS), Vol.9, No.5, pp.9-14, 2017. DOI: 10.5815/ijitcs.2017.05.02