Controlled Proxy Re-signing - Conditional Proxy Re-Signatures

S. Sree Vivek and Guhan Balasubramanian
Samsung Research Institute, Bangalore, India

Keywords: Delegation, Proxy Re-Signature, Conditional Proxy Re-Signature, Provable Security, Random Oracle Model.

Abstract: Delegation of authentication is one of the vital security management strategies to manage device authentication in an enormous network. Major issue while delegating authentication using traditional proxy cryptography is that the delegator loses control over the messages which are authenticated by the delegatee and in proxy re-cryptography controlling the proxy from resigning unintended signatures of the delegatee is not possible. To address this concern, we propose a useful delegation scheme called as conditional proxy re-signature. In this paper, we propose a security model for unidirectional conditional proxy re-signature, present a concrete scheme and prove the security of the scheme in the random oracle model.

1 INTRODUCTION

Proxy Re-Cryptography was introduced by Blaze et. al. (Blaze et al., 1998) and has been an emerging field of interest in the research community. This was mainly motivated by its widespread applications and also the challenges faced in the construction of practical schemes. Proxy Re-Signature and Proxy Re-Encryption schemes are the essential primitives for secure delegation. Proxy Re-Encryption allows a semi-trusted proxy to convert a ciphertext on a message such that it can be decrypted by another receiver instead of the intended receiver defined during the original encryption process. Many such schemes have been designed and are being used for various applications like distributed storage, digital rights management and sharing in cloud storage.

Proxy Re-Signatures: Proxy Re-Signature scheme involves a semi-trusted proxy between two parties A and B, where the proxy has the capability and information (Re-Signature Key) to convert the signature on a message, say m, signed by one user A, to the signature of the other user B on the same message m (A is the delegatee, B is the delegator). This kind of signature comes handy in situations wherein B the delegator is not available to sign on a given message m. Then using a signature of the delegatee A on the message m, the semi-trusted proxy can generate a signature on behalf of B on the same message m with the help of a rekey (interchangeably used as re-signature key or re-sign key), without involving delegator B in the signing process. More details on proxy re-signature schemes can be seen in (Chow and Phan, 2008; Libert and Vergnaud, 2008; Shao et al., 2010).

Conventionally, the delegator must have a strong control over the delegation and signing process since the signature is eventually going to be verified with his/her public key. In proxy re-signatures, the semi-trusted proxy has unconditional power to translate any signature of the delegatee to that of the delegator. This is undesirable as there is no control over the kind of messages the proxy can translate. We try to resolve this issue by introducing conditions to proxy re-signatures which hinder the proxy from converting signatures without the consent of the delegatee. A condition parameter will be an inherent part of all the algorithms in the proxy re-signature scheme.

There must be a certain hold over the degree of freedom given to the proxy to translate signatures. Thus in conditional proxy re-signatures, the proxy can only translate signatures based on a certain condition. The proxy will be given the re-signature key with respect to certain conditions during the rekey generation process. Therefore, the proxy will only be able to translate those signatures for which the condition matches with that present in the re-signature key. In this case, the proxy is constrained to abide by the conditions on messages that were signed by the delegatee before translating the signatures to that of the delegator. This feature brings in a win-win situation for both the delegatee and the delegator due to the following reasons:

- The delegator can assure that the proxy does not translate signatures of unwanted messages to that of the delegator's signature.

- The delegatee can control what kind of signatures generated by him can be translated by the proxy, thereby avoiding a conflict of issues with the delegator.

The delegatee cannot freely sign on messages of his/her own since it could be translated by the proxy. The proxy can translate any signatures of the delegatee to that of the delegator at will, without the consent of the delegatee. In this case the delegator will not have any means to deny signatures which were not signed by him since the translated signatures by the proxy will verify with the delegators public key. There will be no means to prove the fact that the signature was translated with the consent of a delegatee. This issue can pose a real threat in applications like temporary certificate generation for public key infrastructure and signature delegations in organizations. These issues can be very effectively tackled by using conditions along with proxy re-signatures. Some applications of proxy re-signatures with this new feature are e-passports, issuing certificates in PKI, in hierarchical organizations, cloud infrastructure, to list a few.

Related Work: Proxy signatures were introduced by Mambo et. al. (Mambo et al., 1996). Proxy signature uses warrant which defines the condition or the type of messages on which the delegatee can sign on behalf of the delegator. Ivan et al. (Ivan and Dodis, 2003) gave a new look to delegate signing rights. This scheme was not proxy based in a true sense, because the secret key was split between the delegator and delegatee. The most recent proxy signature (with anonymity of delegatee) was given by Pointcheval et. al. in (Fuchsbauer and Pointcheval, 2008).

Proxy re-signatures as mentioned before, was first introduced by Blaze et. al. (Blaze et al., 1998). Later the notion was formalized by Ateniese and Hohenberger (Ateniese and Hohenberger, 2005), who gave the security model and proposed the first proxy re-signature scheme (both unidirectional and bidirectional) which are the most efficient schemes till date. Shao et. al. (Shao et al., 2007) proposed identity based systems in the standard model. The primitive was further formalized by Chow et. al. (Chow and Phan, 2008) and a concrete scheme secure in the standard model was proposed. Libert et. al. (Libert and Vergnaud, 2008) addressed the problem of multi-use, where the signatures can be translated more than once if the rekey is available. They were also responsible for introducing the chosen key model where the challenger has no knowledge about the corrupted private keys. An update on the implementation front was also showed in (Wang and Yang, 2009), which explained how the proxies can be setup in order to avoid a distributed denial of service attack.

In the literature, conditional delegation started with (Watanabe and Numao, 2002), an application for P2P data sharing. The concept of inducing conditions in proxy re-cryptography was formalized in the context of CCA-secure conditional proxy re-encryption in papers by Weng et. al. (Weng et al., 2009a) (proven insecure by (Zhang and Chen, 2009)) (Weng et al., 2009b) and (Chu et al., 2009). The most efficient one being proposed by Selvi et. al. (S. Sree Vivek and Rangan, 2011). Fang et. al. proposed an anonymous proxy re-encryption scheme secure in the standard model in (Fang et al., 2009) and an interactive version of it in (Fang et al., 2011). The first identity based conditional proxy re-encryption was introduced by Shao et. al. (Shao et al., 2011). Recently Fang et. al. proposed the hierarchical version of conditional proxy re-encryption scheme (Fang et al., 2012).

Our Contribution: In this paper, we introduce conditions in the context of proxy re-signatures which has natural applications. Then we propose a security model for chosen message and condition attack to satisfy the property in unidirectional proxy re-signatures with a private proxy. The security model for the unidirectional proxy re-signature scheme is based on the unidirectional security notion against static corruption, proposed by Shao et. al. in (Shao et al., 2010). We give a concrete scheme using bilinear maps and prove its security with the help of random oracles in the proposed security model. The hardness of breaking the scheme is related to the hardness of the computational Diffie-Hellman problem.

2 APPLICATION IN AD-HOC VEHICULAR NETWORKS

Vehicular ad-hoc networks are being theorized as the foundation of connected cars and bring in a whole new dimension to the internet of things. With an increase in road traffic and freight movement, and the digitization of the transportation industry with e-license, electronic vehicle registration plates, smart toll booths, traffic management based on GPS, etc. there is an immense and desperate requirement for secure systems. With a spread of technology across transportation industry, authentication becomes a major concern. The vehicular public key infrastructure (VPKI) deployed in this environment consists of a country root certificate authority (CA), a regional CA and road side units (RSU). The ad-hoc environment of vehicles which consists of an on-board unit (OBU) and a trusted platform module (TPM). The OBU is in charge of communicating with the infrastructure where as the TPM is responsible for all the secure storage and computation. This will hold the identity information of the vehicle which will be certified by the CA. Hence to identify the cars authorized in a region, any vehicle inside a region should hold a valid certificate signed by the regional CA. They are termed as pseudonym certificates as defined by Fuentes et. al. (Jos Mara de Fuentes, 2011) and Raya et al. (Maxim Raya, 2005).

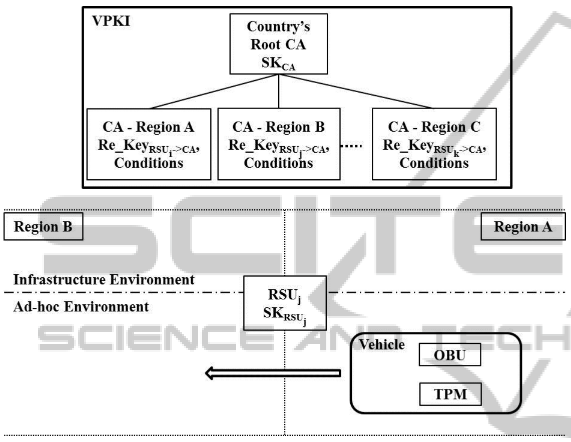
VPKI
Country's Root CA SK$_{CA}$
CA - Region A Re_Key$_{RSU_i\text{->}CA}$, Conditions
CA - Region B Re_Key$_{RSU_j\text{->}CA}$, Conditions
CA - Region C Re_Key$_{RSU_k\text{->}CA}$, Conditions
Region B
Region A
Infrastructure Environment
Ad-hoc Environment
RSU$_j$ SK$_{RSU_j}$
Vehicle
OBU
TPM

Figure 1: Vehicular Public Key Infrastructure.

Figure 1 shows the block diagram of VPKI. The country's root CA has a secret key SK_{CA}. Since there may be many regions in a country, each region will have a regional CA. This is purely for distribution of job and convenience. Each regional CA will have few RSUs associated with them. RSUs have their own secret keys. For example RSU_j, comes under region B has a secret key SK_{RSU_j}. The regional CA will have a conditional re-sign key, which converts root RSU's signature into root CA's signature if the condition holds good. When a vehicle enters region B from Region A, it is required to pass through RSU_j. RSU_j checks the vehicle and signs on the vehicle's credentials and the condition to cross the border, to grant entry. Since there are multiple $RSUs$ for a single region, the signature of RSU_j should be converted to that of the root CA's signature. Hence the authenticity to the entry of all vehicles in a given area can be verified using the root CA's public key. The proxy holds the conditions for which the signature can be converted. The delegator (root CA) can impose what kind of vehicles can pass through the specific RSU. Hence the proxy can convert signatures only if it satisfy certain conditions. These re-signature keys with embedded conditions can be defined on the fly as and when new policies may be introduced.

Figure 2 shows how the vehicle provides the credentials and conditions to the RSU, and the RSU signs it and sends to the regional CA, who converts it into the sign of root CA and sends it back to the RSU to verify and proceed further. This can be of practical use when we do not want to allow entry for vehicles which are not insured or those for which the taxes are not paid in a specific region. Tourist vehicles can be authenticated based on conditions in a foreign region using this mechanism. Yet another scenario could be using the scheme to permit VIP cars into another region without any hassle. This application hence can be used as a viable substitute for access control lists and hence having limited effect on storage in the infrastructure and distribute the load on the root CA.

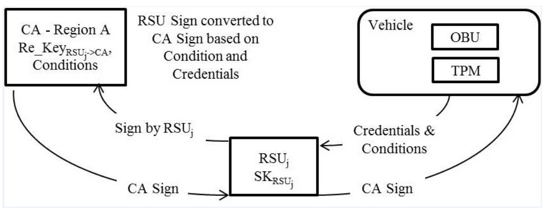
CA - Region A Re_Key$_{RSU_j\text{->}CA}$, Conditions
RSU Sign converted to CA Sign based on Condition and Credentials
Vehicle
OBU
TPM
Sign by RSU$_j$
Credentials & Conditions
RSU$_j$ SK$_{RSU_j}$
CA Sign
CA Sign

Figure 2: Authentication Granted Using CPRS.

3 DEFINITIONS

In this section, we provide the definitions which are used for constructing and proving our scheme.

3.1 Conditional Proxy Re-Signature Scheme

The PKI-based scheme is defined by a set of six algorithms described below: (Setup, KeyGen, ReKey, Sign, ReSign, Verify).

- *Setup(k)* - This algorithm takes as input the security parameter k and generate the system parameters *params*, which is required for the user to generate his/her keys.
- *KeyGen(k, params)* - This algorithm takes the security parameter k and the system parameters *params* as input and generates the public, secret key pair for a user who invokes the algorithm. This algorithm is executed by each user and the generated public keys are certified by a Certification Authorities (CA).
- *ReKey($sk_{Delegator}$,$sk_{Delegatee}$,C)* - This algorithm is an interactive protocol which is carried out in a secure channel between the delegator, delegatee and the proxy. Assuming the system to have a private proxy (re-signature key is kept secret with

the proxy), this algorithm takes as input the secret keys of the users between whom the re-signature key must be generated, and the condition for signature translation. The output of this algorithm is a re-signature key $rk_{Delegatee \rightarrow Delegator}$ which is kept secret with the proxy. The re-signature key is structured in such a way that the proxy cannot gain any advantage regarding the secret keys of the delegator and delegatee.

- *Sign(m,C,sk)* - This algorithm takes as input the message m, condition C and the secret key sk of the user who is signing the message. It outputs the signature of the user on m for the condition C.
- *ReSign($m,C,\sigma,pk_{Delegator},pk_{Delegatee}$)* - This algorithm takes as input, the message m, the condition C and the public keys of the delegator and the delegatee. It first verifies whether the message and the condition verifies the signature with respect to the public key of the delegator. If false, then it returns invalid signature. Otherwise, it uses the re-signature key generated by the ReKey algorithm and then translates the signature σ into a re-signature $\hat{\sigma}$ which can be verified by the public key of the delegatee.
- *Verify(m,C,σ,pk)* - This algorithm takes as input, the message, the condition, the signature/re-signature on the message and the public key of the user which the signature/re-signature is to be verified with. It returns true if the signature is successfully verified with respect to the message, condition and public key and returns false otherwise.

3.2 Security Model for Conditional Proxy Re-Signatures

In general, the security of a signature scheme is established using the notion of existential unforgeability against adaptive chosen message attack, where the adversary will be able to come up with forgeries on messages for which he had not queried the signature. We use the same security notion (but with a stronger unforgeability notion where existing message signature pairs cannot be forged) and come up with a new security model to incorporate conditions (chosen condition attack). There are two existing standard security models for proxy re-signatures - static corruption and dynamic corruption based on which we define the security models for conditional proxy re-signatures. The security model is defined as a game between an adversary and a challenger, where the challenger is required to prove the security of the scheme. The challenger simulates the system, to give the adversary a ”view” of the system. The adversary is allowed to query the various oracles of the system (which require the computation with secret information) provided by the challenger and train itself to obtain enough information and perform a valid forgery for a chosen condition and message.

The stronger security model would be *Dynamic Corruption* which is based on the fact that the role of the adversary is dynamic and any user of the system can turn adversarial at any point during the game. The aim of this model is to explicitly protect the scheme against External (outside the system) and Internal (within the system) adversaries.

We adopt a more applicable model - *static corruption* model - for proving the security of our scheme, which was originally introduced in the security models for proxy re-encryption schemes (Ateniese et al., 2006) where the corrupted and uncorrupted users are decided before the beginning of the security game. In this model, the challenger who is proving the security of the scheme, initially trains the adversary who queries a set of oracles which define the behaviour of the system (oracles for conditional proxy re-signature schemes defined below). It is also to be noted that the security model for unidirectional proxy re-signature schemes with private proxy given in (Shao et al., 2010) is an improved version over that of (Ateniese and Hohenberger, 2005).

According to (Shao et al., 2010) the re-signature key generation queries, a delegation chain is constructed. For example, if the re-sign key for A to B and B to C is generated, then all the signatures of A can be indirectly converted to that of C by the proxy. The use of conditions may restrict this type of insecurities in the delegation chain.
In the static corruption model, since the challenger can differentiate between the corrupted and uncorrupted users beforehand, it simulates the appropriate oracles for the corresponding users. The oracles of the CPRS system are discussed in detail below.

1. *Corrupted Key Generation Oracle:* For a corrupted user, the challenger generates and outputs the private key and the corresponding public key.
2. *Uncorrupted Key Generation Oracle:* For an uncorrupted user, the challenger generates the private key and outputs only the public key.
3. *ReKey Oracle:* For the input of the public keys of the delegator and the delegatee, the condition, the challenger outputs the corresponding re-signature key between the users for the corresponding condition.
4. *Uncorrupted Signature Oracle:* Given the input of the uncorrupted user's public key, the message and the corresponding condition to sign on, the

challenger outputs the corresponding signature.

5. *Re-Signature Oracle:* Given the signature on the message, the condition and the public key of the user to whom the signature is to be translated as input, the challenger outputs the corresponding re-signature on the same message and condition for the given user.

Note: The adversary controls the actions of the corrupted users. At the end of the training phase, the adversary will not have any advantage with respect to the information of the uncorrupted users. The adversary can query the above mentioned oracles considering its limited computation power. Undesirable delegation chains between corrupted and uncorrupted parties can be avoided by the use of conditions.

In the forgery phase, we could conclude that the scheme is secure if the adversary with the access to all these oracles, is able to come up with a valid forgery on the signature or the re-signature with respect to a condition and a message for an uncorrupted user with respect to some conditions as discussed below:

Let $(\bar{m},\bar{C})$ represent the message, condition that were already queried to the sign/re-sign oracle with respect to an uncorrupted user $\bar{pk}$. Let (m^*,C^*) represent the message, condition pair for which the forgery is generated for an uncorrupted user pk^*. Forgery σ^* is valid if it satisfies the following constraints:

- Verify $(pk^*,m^*,C^*) =$ True.
- if $\sigma * \neq \sigma$, where σ is an output of $O_{Sign}(m^*,C^*)$ or $O_{ReSign}(m^*,C^*,\bar{pk},pk^*)$. Or there is a delegation chain with C^* from $\bar{pk}$ to pk^*.
- if $\big((m^*,C^*) \neq (\bar{m},\bar{C})\big)$ or $\big(m^* \neq \bar{m}\big)$ or $\big(C^* \neq \bar{C}\big)$.
- The adversary has not made a ReKey query chain to delegate the signing rights from a corrupted to pk^* on the condition C^*.

We define the advantage of the forger as the probability that he/she wins in the above game. The resulting conditional proxy re-signature scheme is strongly unforgeable against chosen message and condition attack if the above mentioned forgery can be performed only with a negligible probability. *Motivation* for this kind of a security model based on (Shao et al., 2010) is that the forger should not be able to derive any undue advantage as a consequence of the introduction of the condition. As one may notice, the introduction of condition reduces the number of constraints on the forgery as when compared to a traditional proxy re-signature.

Bilinear Pairing: Let $\mathbb{G}$ be an additive cyclic group generated by P, with prime order p, and $\mathbb{G}_1$ be a multiplicative cyclic group of the same order p. A bilinear pairing is a map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$ with the following properties.

- *Bilinearity:* For all $P,Q,R \in \mathbb{G}$,
  - $\hat{e}(P+Q,R) = \hat{e}(P,R)\hat{e}(Q,R)$
  - $\hat{e}(P,Q+R) = \hat{e}(P,Q)\hat{e}(P,R)$
  - $\hat{e}(aP,bQ) = \hat{e}(P,Q)^{ab}$
- *Non-Degeneracy:* There exist $P,Q \in \mathbb{G}$ such that $\hat{e}(P,Q) \neq I_{\mathbb{G}_1}$, where $I_{\mathbb{G}_1}$ is the identity in $\mathbb{G}_1$.
- *Computability:* There exists an efficient algorithm to compute $\hat{e}(P,Q)$ for all $P,Q \in \mathbb{G}$.

Computational Diffie-Hellman Problem: Let G be a group of prime order p and g be the generator of $\mathbb{G}$. The CDH problem can be defined as follows: An algorithm A is said to have an advantage ε in solving the CDH problem if

$$Pr[A(P,aP,bP) = abP] \geq \varepsilon$$

where the probability is calculated over the random choices of $a,b \in \mathbb{Z}_p^*$, $g \in \mathbb{G}^*$ and the random bits used by algorithm A.

4 SCHEME

In this section we define a concrete PKI-based conditional proxy re-signature scheme which is based on pairing. The security is asymptotically based on a security parameter k.

<u>Setup(k)</u> and <u>KeyGen(k)</u>

- Let $\mathbb{G}$ and $\mathbb{G}_1$ be cyclic groups of prime order p defined to satisfy the billinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$
- P is chosen as the generator of $\mathbb{G}$.
- Choose four cryptographic hash functions defined as follows:
  H_1: $\{0,1\}^* \to \mathbb{G}$
  H_2: $\{0,1\}^* \times \mathbb{G} \times \mathbb{G} \to \mathbb{G}$
  H_3: $\{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \to \mathbb{G}$
  H_4: $\{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \to \mathbb{G}$
- Each user generates his/ her private and public key pair using the public parameters defined above. After generating the key pair, he/ she gets the public key certified from the certification authority. The keys are generated as follows:
  - Choose $x \in_R \mathbb{Z}_p$ and set $sk = x$ as the user's private key.
  - Compute the public key $pk = xP \in \mathbb{G}$.
  - With respect to our scheme, we assume that the delegatee's (X) secret key is x and the delegator's (Y) secret key is y.

Note: In certain cases, where the condition can be directly obtained from the message to be signed, we may assume a heuristic black box $K(m) = C$ that when given the input message m, returns the condition(s) C, that corresponds to the message. This may not be applicable for applications where the condition depends on the situation (example: urgency). The introduction of this black box will not provide the forger any undue advantage for giving a forgery.

ReKey(x, y, C)

The re-signature key $rk_{x \to y}(C)$ is generated for the condition C using an interactive protocol which indirectly accepts the secret keys of the delegatee X and the delegator Y without revealing it to the proxy.

- The proxy selects elements $R \in_R \mathbb{G}$ and $r_2 \in_R \mathbb{Z}_q$, and sends R and r_2P to the delegator Y.
- Y then computes $yH_1(C) + yH_2(C, r_2P) \in \mathbb{G}$ where y is the secret key and C is the condition for which the delegator chooses to delegate its signing rights.
- Y sends $yH_1(C) + yH_2(C, r_2P) + R \in \mathbb{G}$ and the condition C to the delegatee X.
- X computes $xH_1(C)$ and sends $yH_1(C) + yH_2(C, r_2P) + R - xH_1(C) \in \mathbb{G}$ to the proxy.
- The proxy removes R and assigns the re-signature keys from X to Y for the condition C as
  $rk_1 = (y - x)H_1(C) + yH_2(C, r_2P) \in \mathbb{G}$
  $rk_2 = r_2P \in \mathbb{G}$.

The rekey $rk_{x \to y}(C) = \langle rk_1, rk_2 \rangle$. Note that role of R in the rekey generation process is to prevent X and Y from misusing the intermediate information and r_2 is known to the proxy and is used in the resign algorithm.

Remark: We use an interactive rekey algorithm in order to facilitate the proxy to re-use the randomness in the resign algorithm. This reduces the space and computation time required to carry out the re-signature process and moreover offers more secure construction. Since it is a one-time computation, it does not affect the performance of the scheme.

Sign(m, C, x)

Given the message m, condition C and the secret key x of the signer as input, the algorithm performs the following steps:

- Choose a random $r_1 \in \mathbb{Z}_p$ and compute $r_1P \in \mathbb{G}$
- Compute $\sigma_1 = xH_1(C) + r_1H_3(m, C, r_1P)$
- Compute $\sigma_2 = r_1P$

Hence, the output signature σ for message m and condition C for the user X is $\langle \sigma_1, \sigma_2 \rangle$.

Remark: Notice that the signature algorithm is based on the short signature scheme proposed by Boneh et. al. (BLS signature scheme (Boneh et al., 2001)).

ReSign($\sigma, xP, m, C, rk_{x \to y}(C)$)

Given the message m, condition C, public key xP for the corresponding signature σ and the re-signature key $rk_{x \to y}(C)$ as input, the re-signature algorithm performs the following steps:

- The algorithm $Verify(\sigma, xP, m, C)$ is invoked in order to check the validity of the signature to be translated with respect to the delegatee's public key xP. If this test does not hold, the translation will be meaningless. Hence, the resign algorithm can continue only if the test holds.
- Compute and store re-randomization component as $r_2H_4(m, C, r_2P, r_1P) \in \mathbb{G}$ where r_2 is from the ReKey algorithm. The hash function includes the randomness introduced in the sign and rekey algorithm to maintain the integrity of the resultant resignature.
- The translated signature is computed as

  $\hat{\sigma}_1 = \{\sigma_1\} + \{rk_1\} + \}$Re-Randomization component$\}$.
  $= \{xH_1(C) + r_1H_3(m, C, r_1P)\} + \{yH_1(C) - xH_1(C) + yH_2(C, r_2P)\} + \{r_2H_4(m, C, r_2P, r_1P)\}$
  $= yH_1(C) + r_1H_3(m, C, r_1P) + yH_2(C, r_2P) + r_2H_4(m, C, r_2P, r_1P)$.
- Assign $\hat{\sigma}_2 = \sigma_2 = r_1P$.
- Assign $\hat{\sigma}_3 = rk_2 = r_2P$.

Hence, the output re-signature $\hat{\sigma}$ for message m and condition C for the user Y is $\langle \hat{\sigma}_1, \hat{\sigma}_2, \hat{\sigma}_3 \rangle$.

Verify(σ,xP,m,C)

This algorithm takes the signature σ / re-signature $\hat{\sigma}$ of the user X on the message m for the condition C as input and returns true if the signature is valid.
The algorithm performs the following to check the validity of a signature:

$$\hat{e}(\sigma_1, P) \stackrel{?}{=} \hat{e}(H_1(C), xP).\hat{e}(H_3(m, C, r_1P), \sigma_2)$$

where signature is of the form:
$\langle \sigma_1, \sigma_2 \rangle = \langle xH_1(C) + r_1H_3(m, C, r_1P), r_1P \rangle$

The algorithm performs the following to check the validity of a re-signature:

$$e(\hat{\sigma}_1, P) \stackrel{?}{=} e(H_1(C), xP) . e(H_3(m, C, r_1P), \hat{\sigma}_2) . e(H_2(C, r_2P), xP) . e(H_4(m, C, r_2P, r_1P), \hat{\sigma}_3)$$

where the re-signature is of the form:
$\langle \hat{\sigma}_1, \hat{\sigma}_2, \hat{\sigma}_3 \rangle = \langle xH_1(C) + r_1H_3(m, C, r_1P) + xH_2(C, r_2P) + r_2H_4(m, C, r_2P, r_1P), r_1P, r_2P \rangle$

If the above test fails, then the signature is invalid.

Correctness of Signature Verification:

$$\begin{aligned}\hat{e}(\sigma_1,P) &= \hat{e}(xH_1(C)+r_1H_3(m,C,r_1P),P)\\ &= \hat{e}(xH_1(C),P).\hat{e}(r_1H_3(m,C,r_1P),P)\\ &= \hat{e}(H_1(C),xP).\hat{e}(H_3(m,C,r_1P),r_1P)\\ &= \hat{e}(H_1(C),xP).\hat{e}(H_3(m,C,r_1P),\sigma_2)\end{aligned}$$

Correctness of Re-Signature Verification:

$$\begin{aligned}\hat{e}(\hat{\sigma}_1,P) =& \hat{e}(xH_1(C)+r_1H_3(m,C,r_1P)+xH_2(C,r_2P)\\ &+r_2H_4(m,C,r_2P,r_1P),P)\\ =& \hat{e}(xH_1(C),P).\hat{e}(r_1H_3(m,C,r_1P),P).\\ &\hat{e}(xH_2(C,r_2P),P).\hat{e}(r_2H_4(m,C,r_2P,r_1P),P)\\ =& \hat{e}(H_1(C),xP).\hat{e}(H_3(m,C,r_1P),r_1P).\\ &\hat{e}(H_2(C,r_2P),xP).\hat{e}(H_4(m,C,r_2P,r_1P),r_2P)\\ =& \hat{e}(H_1(C),xP).\hat{e}(H_3(m,C,r_1P),\hat{\sigma}_2).\\ &\hat{e}(H_2(C,r_2P),xP).\hat{e}(H_4(m,C,r_2P,r_1P),\hat{\sigma}_3)\end{aligned}$$

Efficiency Analysis: The signature has two group elements and the re-signature only increases the size by an extra group element, due to the extra randomization component introduced by the resign algorithm. While commenting on the computational complexity, the sign algorithm requires three scalar multiplications and one group addition in $\mathbb{G}$. The resign algorithm requires one scalar multiplication and three group additions in $\mathbb{G}$. The verify algorithm for a signature requires three pairing operations and one group multiplication in $\mathbb{G}_1$ while the verify algorithm for a re-signature requires five pairing operations and three group multiplications in $\mathbb{G}_1$.

Proof of Security

Theorem 1: *We show that if there is an adversarial algorithm that is capable of forging the CPRS scheme, then it is possible to construct another algorithm which breaks the CDH problem with a non-negligible advantage.*

Proof Sketch: The challenger gets an instance of the CDH problem say $\langle P,aP,bP\rangle$ as input. The goal of the challenger is to find abP with a non-negligible probability. The challenger sets up the system and in the **training phase**, the forger is allowed to query (polynomial in number) Corrupted KeyGen Oracle, Uncorrupted KeyGen Oracle , all the hash oracles, Uncorrupted User's Sign oracle, Rekeygen oracle and ReSign oracle in order to be trained by the challenger to perform the forgery. We explain the oracles which are required to show the proof sketch here.

Uncorrupted KeyGen Oracle: The challenger takes a random $x_i \in \mathbb{G}$ and outputs $(a+x_i)P$ as the public key (where aP is an input of the hard problem instance). The challenger does not know the private key a.

H_1 Oracle: For condition C^*, which is fixed by the challenger, the challenger embeds the hard problem.

- Choose $t_{1i} \in_R \mathbb{Z}_p$ and compute $t_{1i}bP \in \mathbb{G}$.

H_2 Oracle: Given the condition C and the random group element $r_{2i}P \in \mathbb{G}$, the challenger chooses random $t_{2i} \in_R \mathbb{Z}_p$ and returns $t_{2i}P \in \mathbb{G}$. The challenger stores the tuple $\langle C,r_{2i}P,\ t_{2i}\rangle$ in the list L_2 for future use.

H_3 Oracle: Given a message m, a condition C and a random group element in $r_{3i}P \in \mathbb{G}$ the challenger chooses $t_{3i} \in_R \mathbb{Z}_p$ and returns $t_{3i}P \in \mathbb{G}$. The challenger stores the tuple $\langle m,C,r_{3i}P,t_{3i}\rangle$ in the list L_3 for future use.

H_4 Oracle: Given a message m, condition C, and two random group elements $r_{41i}P, r_{42i}P$, the challenger chooses $t_{4i} \in_R \mathbb{Z}_p$ and returns $t_{4i}P \in \mathbb{G}$. The challenger stores the tuple $\langle m,C,r_{41i}P,r_{42i}P,t_{4i}\rangle$ in the list L_3 for future use.

Explanation about the Uncorrupted User Sign oracle, Rekeygen oracle and ReSign oracle will be available in the full version of the paper and is omitted here due to page restriction. In the **forgery phase**, the forger comes up with a valid signature or re-signature on the message m^* and condition C^*. It is also assumed that $H_1(C^*)$ was queried during the training phase. Otherwise the game is aborted.

<u>*Forgery on Signature*</u> is of the form $\langle\sigma_1^*,\sigma_2^*\rangle = \langle(a+x^*)H_1(C^*)+r_1^*H_3(m^*,C^*,r_1^*P),r_1^*P\rangle$.

Challenger checks in list L_3 for the input m^*,C^*,σ_2^* and if present, obtains the value of t_3^*. And hence σ_1^* = $(a+x^*)t_{1i}bP+r_1^*t_3^*P$. The challenger now finds the solution to the hard problem by computing

$$\begin{aligned}&\sigma_1^* - x^*t_1^*bP - t_3^*\sigma_2^* =\\ &\quad = abt_1^*P + x^*t_1^*bP + r_1t_3^*P - x^*t_1^*bP - t_3^*r_1P\\ &\quad = abt_1^*P\end{aligned}$$

Now abP is obtained by multiplying the inverse of t_1^* to the above output.

<u>*Forgery on Re-Signature*</u> is of the form $\langle\hat{\sigma}_1^*,\sigma_2^*,\hat{\sigma}_3^*\rangle = \langle(a+x^*)H_1(C^*)+r_1^*H_3(m^*,C^*,r_1^*P)+(a+x^*)H_2(C^*,r_2^*P)+r_2^*H_4(m^*,C^*,r_2^*P,r_1^*P),r_1^*P,r_2^*P\rangle$

The challenger checks the list L_3 to obtain value of t_3^*, checks the list L_2 to obtain the value of t_2^* and checks the list L_4 to obtain the value of t_4^*. The challenger computes the following to obtain the solution to the CDH problem:

$$\begin{aligned}&\hat{\sigma}_1^* - x^*t_1^*bP - t_3^*\sigma_2^* - x^*t_2^*P - t_2^*aP - t_5^*\sigma_3^*\\ &\quad = abt_1^*P + x^*t_1^*bP + r_1t_3^*P + t_2^*aP + x^*t_2^*P +\\ &\quad r_2^*t_5^*P - x^*t_1^*bP - t_3^*r_1P - x^*t_2^*P - t_2^*aP - t_5^*r_2^*P\\ &\quad = abt_1^*P\end{aligned}$$

Now abP is obtained by multiplying the inverse of t_1^* to the above output.

5 CONCLUSIONS

In this paper, we have proposed a new primitive called conditional proxy re-signature. We have defined a suitable security model for conditional proxy re-signatures. We have presented a concrete conditional proxy re-signature scheme which is unidirectional and single-use, and proved the security in the random oracle model. The security of the proposed scheme is related to the hardness of the CDH problem.

We let the design of a multi-use CPRS scheme, where the signature can be translated more than one time with a nominal increase in complexity and signature size for each translation as an interesting open problem. Providing CPRS schemes in the identity based setting and certificateless settings would also be challenging and would have practical relevance in cloud based systems. A standard model security proof would further solidify the practical applicability of such a scheme.

REFERENCES

Ateniese, G., Fu, K., Green, M., and Hohenberger, S. (2006). Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.*, 9(1):1–30.

Ateniese, G. and Hohenberger, S. (2005). Proxy re-signatures: new definitions, algorithms, and applications. In *ACM Conference on Computer and Communications Security*, pages 310–319.

Blaze, M., Bleumer, G., and Strauss, M. (1998). Divertible protocols and atomic proxy cryptography. In *EUROCRYPT*, pages 127–144.

Boneh, D., Lynn, B., and Shacham, H. (2001). Short signatures from the weil pairing. In *ASIACRYPT*, pages 514–532.

Chow, S. S. M. and Phan, R. C.-W. (2008). Proxy re-signatures in the standard model. In *ISC*, pages 260–276.

Chu, C.-K., Weng, J., Chow, S. S. M., Zhou, J., and Deng, R. H. (2009). Conditional proxy broadcast re-encryption. In *ACISP*, pages 327–342.

Fang, L., Susilo, W., Ge, C., and Wang, J. (2011). Interactive conditional proxy re-encryption with fine grain policy. *Journal of Systems and Software*, 84(12):2293–2302.

Fang, L., Susilo, W., Ge, C., and Wang, J. (2012). Hierarchical conditional proxy re-encryption. *Computer Standards and Interfaces*, vol - 34(no - 4):380 – 389.

Fang, L., Susilo, W., and Wang, J. (2009). Anonymous conditional proxy re-encryption without random oracle. In *ProvSec*, pages 47–60.

Fuchsbauer, G. and Pointcheval, D. (2008). Anonymous proxy signatures. In *SCN*, pages 201–217.

Ivan, A.-A. and Dodis, Y. (2003). Proxy cryptography revisited. In *NDSS*.

Jos Mara de Fuentes, Ana Isabel Gonzlez-Tablas, A. R. (2011). Overview of security issues in vehicular ad-hoc networks. In *Handbook of Research on Mobility and Computing*, pages 894–911. Hershey: IGI Global.

Libert, B. and Vergnaud, D. (2008). Multi-use unidirectional proxy re-signatures. In *ACM CCS*, pages 511–520.

Mambo, M., Usuda, K., and Okamoto, E. (1996). Proxy signatures for delegating signing operation. In *Proceedings of the 3rd ACM*, CCS '96, pages 48–57. ACM.

Maxim Raya, J. H. (2005). The security of vehicular ad hoc networks. In *Proceedings of the 3rd ACM Workshop on Security of ad hoc and Sensor Networks, SASN - 2005*, pages 11–21. ACM.

S. Sree Vivek, S. Sharmila Deva Selvi, V. R. and Rangan, C. P. (2011). Conditional proxy re-encryption – a more efficient construction. In *CNSA*, CCIS 196, pages 502–512.

Shao, J., Cao, Z., Wang, L., and Liang, X. (2007). Proxy re-signature schemes without random oracles. In *INDOCRYPT*, pages 197–209.

Shao, J., Feng, M., Zhu, B., Cao, Z., and Liu, P. (2010). The security model of unidirectional proxy re-signature with private re-signature key. In *ACISP*, pages 216–232.

Shao, J., Wei, G., Ling, Y., and Xie, M. (2011). Identity-based conditional proxy re-encryption. In *IEEE Conference on Communications - ICC*, pages 1–5.

Wang, X. a. and Yang, X. (2009). On ddos attack against proxy in proxy re-encryption and proxy re-signature. In *Proceedings of the 2009 Ninth IEEE International Conference on Computer and Information Technology - Volume 02*, CIT '09, pages 213–218. IEEE Computer Society.

Watanabe, Y. and Numao, M. (2002). Conditional cryptographic delegation for p2p data sharing. In *Proceedings of the 5th International Conference on Information Security*, ISC '02, pages 309–321. Springer-Verlag.

Weng, J., Deng, R. H., Ding, X., Chu, C.-K., and Lai, J. (2009a). Conditional proxy re-encryption secure against chosen-ciphertext attack. In *ASIACCS*, pages 322–332.

Weng, J., Yang, Y., Tang, Q., Deng, R. H., and Bao, F. (2009b). Efficient conditional proxy re-encryption with chosen-ciphertext security. In *ISC*, pages 151–166.

Zhang, X. and Chen, M.-R. (2009). On the security of a conditional proxy re-encryption. *IEICE Transactions*, 92-A(10):2644–2647.