ABSTRACT
Proxy re-encryption is a scheme where a semi-trusted proxy alters a ciphertext of one party into a ciphertext of another party without seeing the underlying plaintext. To do the same, various solutions have been suggested in the public-key setting cryptography. In this paper, we propose two new methods named CTR1-PRE and CTR2-PRE for proxy re-encryption on public key setting which are based on elliptic curve. Since elliptic curve cryptography is becoming a new famous approach due to its lot of nice features, it is required to design proxy re-encryption scheme which work on elliptic curve as well. So, in this paper we show two such proxy re-encryption methods and we also show that one of our approach holds all the essential properties of proxy re-encryption such as unidirectional, non-interactive, proxy invisible, original access, key optimal,
collusion safe, temporary, etc.

Keywords
Proxy re-encryption, elliptic curve based proxy re-encryption, CTR1-PRE, CTR2-PRE.

1. INTRODUCTION
Blaze, Bleumer and Strauss [2] announced the concept of Proxy Re-Encryption (PRE). Proxy re-encryption permits a proxy to transform a ciphertext calculated under one party’s public key into one that can be decrypted by another party’s secret key. In proxy re-encryption schemes, a private (secret) key holder creates a re-encryption key. A semi-trusted proxy can use this re-encryption key to translate a plaintext \( m \) which is encrypted by using the delegator’s public key into an encryption of the same plaintext under a delegate’s public key, as stated by the delegator. There are several useful applications of this primitive. For instance, Alice might wish to temporarily forward encrypted email to her colleague Bob, without giving him her secret key. In this case, Alice as a delegator could designate a proxy server to re-encrypt her incoming mail into a format that Bob as a delegate can decrypt using his own secret key. To do the same, what Alice can do is, she could simply provide her secret key to the semi-trusted proxy to decrypt her ciphertext and encrypt the resultant plaintext by using Bob’s public key and send the ciphertext to Bob, but this requires an unrealistic level of trust in the proxy. And this won’t be a feasible or desirable solution to do.

A various number of proxy re-encryption protocols have been proposed in the context of public key encryption [1, 2, 3, 4, 8, 11]. In this work we lengthen the idea of proxy re-encryption to the area of elliptic curve based encryption. As elliptic curve cryptography becomes an emerging technique in crypto world, we start to explore the proxy re-encryption technique for elliptic curve based crypto approach. The following table 1 provides the key sizes suggested by the National Institute of Standards and Technology to protect the keys which are used in conventional symmetric encryption algorithms like the (DES) and (AES) together with the key sizes used in asymmetric encryption algorithms like RSA, Diffie-Hellman and elliptic curves that are needed to provide same level of security.

<table>
<thead>
<tr>
<th>Symmetric Key Size (bits)</th>
<th>RSA and Diffie-Hellman Key Size (bits)</th>
<th>Elliptic Curve Key Size (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>80</td>
<td>1024 160</td>
<td>80</td>
</tr>
<tr>
<td>112</td>
<td>2048 224</td>
<td>112</td>
</tr>
<tr>
<td>128</td>
<td>3072 256</td>
<td>128</td>
</tr>
<tr>
<td>192</td>
<td>7680 384</td>
<td>192</td>
</tr>
<tr>
<td>256</td>
<td>15360 521</td>
<td>256</td>
</tr>
</tbody>
</table>

From the above table 1, we can observe that as symmetric key size increases the required key size for RSA and Diffie-Hellman increases at a much faster rate than the required key size for elliptic curve cryptosystems. Hence, elliptic curve cryptosystems offer more security per bit rise in key size than either RSA or Diffie-Hellman public key systems. Security alone is not the only attractive feature of elliptic curve cryptography. Elliptic curve cryptosystems are also more computationally efficient than the traditional public key systems such as RSA and Diffie-Hellman. Although the arithmetic operations over elliptic curve is somewhat more complex per bit than either RSA or DH arithmetic, the additional security added per bit conciliate for any extra computational time.

Proxy re-encryption has numerous thrilling applications [2, 3, 8, 9] such as law enforcement, email forwarding, secure multi-party computation and performing cryptographic operations on storage-limited devices. Proxy re-encryption has also been used as the foundation for program obfuscation. It is also being studied in relative to the field of Identity-Based Encryption (IBE) [12, 13, 14, 15, 16, 17] and re-signature schemes.

The rest of this paper is defined as flows. Section 2 gives a brief history or literature survey in the field of proxy encryption as well as proxy re-encryption. Section 3 and 4 present the necessary definitions and technical preliminaries to understand our schemes. Section 5 discuss about our proposed approaches and their properties. Finally section 6 ends with conclusion.

2. LITERATURE SURVEY
The following diagram describes the conventional way of doing proxy re-encryption

Figure 1. Proxy re-encryption operation

where,

\( M \) Message to encrypt
Ca "" Ciphertext for Alice (delegator)
Cb "" Ciphertext for Bob (delegate)
PKa "" Public Key of Alice
SKa "" Secret Key of Alice

The apparent problem with this policy is that the proxy server must be totally trusted by the delegator which is an unlikely real-world expectation. Up to the year 1997, this was an inappropriately top solution available. Then the same year, Mambo and Okamoto recommended more efficient methods involving partial decryptions, without offering any extra security benefits for delegator”’s secret key [1].

In the year 1998, Blaze, Bleumer & Strauss (BBS) proposed a new proxy re-encryption technique which was based on the secret keys of both the parties. As the BBS proxy re-encryption algorithm is based on ElGamal cryptosystem, the ciphertext is semantically secure. Moreover, without knowing the secret key of either Alice “’a” or Bob “’b”, the proxy server cannot differentiate the re-encryption key RKA”’B = b/a = ba-1 from a randomly selected number of Zp*. Their approach was very well-designed. However, there are some concerns that one might want to develop on:

"’ Bidirectionality: BBS scheme is of bidirectional proxy re-encryption, means the proxy server can easily compute (RKA”’B)-1 = a/b = ab-1, which will enable the proxy to re-encrypt Bob”’s messages under Alice”’s key without Bob”’s permission.

"’ Collusion: If the proxy server colludes with Alice, then the secret key of Bob can easily be derived. Likewise, the proxy and Bob may collude to crack Alice”’s secret key.

"’ Re-encryption key generation: In order to compute the re-encryption key from Alice to Bob i.e. RKA”’B, one party must share his secret key with the other party or they must depend on a trusted third party or they must compute some secure multiparty computation with the proxy.

In 2003, Dodis and Ivan [3] proposed a general approach for proxy encryption (not re-encryption) using only ordinary public key cryptosystems. However, their system requires a pre-sharing of secret key between Alice and Bob. Although we have algorithms like Diffie-Hellman to share a common secret between two parties in a public channel, it is undesirable. Because it may be that both Alice and Bob have no prior association whatsoever and bidirectional communication is impossible.

In 2005, Ateniese et al. (AFGH) [4] built the very first unidirectional, collusion resistant re-encryption without any prerequisite pre-sharing between parties, based on bilinear maps. Their algorithm offered several nice features as follows,

"’ No pre-sharing of secret keys: Alice can straightforwardly compute the re-encryption key to Bob i.e. RKA”’B using only her secret key SKa and public key of Bob PKb; Bob does not need to share his secret key SKb with Alice or any other third party in order to construct re-encryption key.

"’ Non-interactive: As Bob”’s public key and Alice”’s secret key are more than enough to create the re-encryption key, no prior arrangement or secret sharing is necessary. But, due to the lack of pre-sharing, Alice may use this scheme without any prior knowledge of Bob.

"’ Proxy security: There is no way for the proxy to derive the plaintexts as all of them are encrypted or re-encrypted. And the proxy server can”’t crack the secret key of either party”’s by its own.

"’ Unidirectional under the Inverse Exponent Assumption (equivalent to the Diffie-Hellman Assumption): The proxy server can”’t compute ga/b given gb/a and g, so Bob”’s ciphertexts can”’t be re-encrypted to Alice by proxy server itself without Bob”’s approval.

"’ Collusion resistant: It is tough for Alice to extract the secret key of Bob i.e. “’b” from RKA”’B = gb/a, even after colluding with proxy. The same thing is applicable for Bob as well.

"’ The security of AFGH proxy re-encryption scheme is reserved under the Decisional Bilinear Diffie-Hellman Inversion Assumption (DBDHI).

AFGH scheme provides two different form of ciphertexts: The original form (gra , Zr "" m) and the re-encrypted form (Zrb , Zr "" m), where Z = e(g , g). In the first form, the first component in the ciphertext pair is an element of G1; in the second form, the first component is an element of G2. Thus, AFGH PRE scheme will totally change the first component of re-encrypted ciphertext pair from original encrypted ciphertext pair. Due to this fact, AFGH PRE scheme ciphertexts may be re-encrypted only once. That is, once RKA”’B has been used to produce Cb, RKB”’C can”’t be used to produce a ciphertext for Charlie. This has both pros and cons. Alice is provided an additional measure of control of her encrypted message; i.e. she knows that even if she
approves re-encryption for Bob, Bob can’t then delegate her messages to somebody else. But, in some applications it might require to have further delegations. Our second approach is basically based on AFGH algorithm in the context of elliptic curve.

Green-Ateniese [5], Chu-Tzeng [6] and Wang et al. [7] further explored the proxy re-encryption approaches by using identity based cryptosystem. Even though, their approaches were unidirectional, they failed to resist against collusion.

3. DEFINITIONS

We begin by describing all the elementary definitions to understand the proposed methodologies in a better way. We then formally define our approaches.

Elliptic curve

For current cryptographic purposes, an elliptic curve (E) is a plane curve over a finite field which consists of the points satisfying the equation 1, along with a distinguished point at infinity denoted 'O'.

\[ y^2 = x^3 + Ax + B \]  \hspace{1cm} (1)

For cryptographic purpose, we will be mostly using non-singular elliptic curve. Let \( \Delta = 4A^3 + 27B^2 \) be the discriminant of the cubic in x. Then the elliptic curve E is singular, if \( \Delta = 0 \), i.e. the cubic has a repeated root, and non-singular otherwise, i.e. the cubic has distinct roots.

Bilinear Map

Let \( G_1, G_2 \) be cyclic groups of prime order p. Let \( g \) be a generator of \( G_1 \). A bilinear pairing or bilinear map \([10]\) \( e \) is an efficiently computable function \( e : G_1 \times G_1 \rightarrow G_2 \) such that it satisfies the below two conditions,

1. Non-degeneracy: \( e(g, g) \neq 1 \).
2. Bilinearity: \( e(g^a, g^b) = e(g, g)^{ab} \) for all \( a, b \in \mathbb{Z} \).

4. TECHNICAL PRILIMINARY

ElGamal based Elliptic Curve Cryptosystem

Set up an elliptic curve \( E \) over a finite field \( \mathbb{F}_q \) where \( q \) is large prime number (at least 160 bits) and a point \( P \) of order \( n \).

Select a message embedding function \( f : m \rightarrow P_m \), which maps an arbitrary message \( m \) to a point \( P_m \) on \( E \). It should be a kind of invertible function, so that we can get back our original message \( m \) from the point \( P_m \). One way to achieve this primitive is, we can use \( m \) as \( x \) in the curve’s equation and calculate the corresponding \( y \).

Choose a random secret key \( x \in [1, n^{-1}] \), publish the point \( Y = xP \) as public key.

Encryption1: Choose random \( k \in [1, n^{-1}] \), then calculate the ciphertext pair as \( A = kP \) and \( B = kY + P_m \) where \( P_m = f(m) \). Finally the ciphertext is a tuple \((A, B)\).

Decryption1: From the ciphertext tuple \((A, B)\), calculate \( A'' = kY + P_m \) and retrieve the point \( P_m \) as follows, \( P_m = B - A'' = kP + P_m - x(kP) = k(xP) + P_m - x(kP) \). Then calculate the message \( m \) by finding inverse as \( f^{-1}(P_m) \).

Encryption2: Choose random \( k \in [1, n^{-1}] \), then calculate the ciphertext pair as \( A = kY \) and \( B = kP + P_m \) where \( P_m = f(m) \). The ciphertext is the tuple \((A, B)\).

Decryption2: From the ciphertext pair \((A, B)\), calculate \( A'' = x^{-1}A \) and retrieve the point \( P_m \) as follows, \( P_m = B - A'' = kP + P_m - x^{-1}k(xP) = kP + P_m - kP \). Then calculate the message \( m \) by finding inverse as \( f^{-1}(P_m) \).

In our proposed approaches, we used the second version of ElGamal based elliptic curve encryption.

5. PROPOSED APPROACHES

5.1 CTR1-Proxy Re-Encryption

CTR1 Proxy Re-Encryption scheme is tuple of algorithms (Setup, KeyGen, ReKeyGen, Encrypt, Decrypt, Re-encrypt) based on elliptic curve cryptosystem.

Setup: Let \( E(\mathbb{F}_q) \) be an elliptic curve over the finite field \( \mathbb{F}_q \) where \( q \) is large prime number (at least 160 bits) and \( G \) be a point on \( E \) of order \( n \). Publish the system parameters as \( SP = (E, \mathbb{F}_q, G, f, n) \). Here \( f \) denotes the message embedding function which is used to convert arbitrary message into a point on elliptic curve.

KeyGen(SP) \( (ska, pka) \). Given \( SP \), output secret key ska = a \( \in \mathbb{Z}^* \) and public key pka = aG \( \in \mathbb{E} \). point on E.

ReKeyGen(ska, skb) \( (rka,b) \). Given secret keys ska and skb output re-encryption key rka\( ^{-1} b = b/a \in \mathbb{Z}^* \).

Encrypt(pka, m) \( (Ca) \). Given public key pka and message m \( \in \mathbb{Z}^* \), generate a random r \( \in \mathbb{Z}^* \), output Ca = \( (A, B) = (r \cdot pka, rG + P_m) \), where \( P_m = f(m) \). Here the function \( f(m) \) is used to convert arbitrary message m to a point on elliptic curve.
Re-encrypt(rka”b, Ca) " (Cb). Given a re-encryption key rka”b and ciphertext Ca, compute Cb = (A”, B”) = (r” pka” rka”b, rG + Pm).

Decrypt(Ca, ska) " m. Given ciphertext Ca and secret key ska, output Pm = B” ska-1A. To get back the original message m from Pm, we need to apply inverse of the function f.

Proof of Correctness:
Let”s assume the system parameters SP as {E, q, G, f, n}.

Alice”s private key as “a” and public key as Q = aG.
Bob”s private key as “b” and public key as R = bG.
So re-encryption key is calculated as rka”b = b/a = ba-1.

Let”s assume Victor wants to send a message to Alice. So he chooses a random number r from Zn* and calculates the ciphertext pair as Ca = (A, B) = (rQ, rG + Pm) and sends this ciphertext to Alice.
Now the proxy need to convert this Alice”s ciphertext into Bob”s ciphertext by using supplied re-encryption key. The new ciphertext Cb is calculated as follows,

Cb = (A”, B”) = (rQ” rka”b, rG + Pm). And this calculated ciphertext is sent to Bob.
After receiving the ciphertext Cb, Bob will reconstruct the original message as follows,

Pm = B” ” b-1A”
= rG + Pm ” b-1” rQ ” rka”b
= rG + Pm ” b-1” r ” aG ” ba-1
= rG + Pm ” rG
= Pm
Finally the original message is derived by taking inverse of function f on Pm, i.e. m = f-1(Pm).

5.2 CTR2- Proxy Re-Encryption
CTR2 Proxy Re-Encryption scheme is again tuple of algorithms (Setup, KeyGen, ReKeyGen, Encrypt, Re-encrypt, Decrypt) based on elliptic curve cryptosystem. Here, along with elliptic curve, bilinear pairing is used to construct the system.

Setup: Let E(Fq) be an elliptic curve over the finite field Fq where q is large prime number (at least 160 bits) and G be a point on E of order n. Let G1, G2 be two multiplicative cyclic groups of prime order n. Let e : G1 × G1 → G2 be a bilinear map, z = e(G1, G1) → G2. The system parameters SP = (E, q, e, G, G1, G2, z, f, n).

KeyGen(SP) ” (ska, pka). Given SP, output secret key ska = a” Zn* and public key pka = aG ” point on E.
ReKeyGen(ska, pkb) ” rka”b. Given secret key ska and public key pkb, output re-encryption key rka”b = a-1bG.

Encrypt(pka, m) ” (Ca). Given public key pka and message m” Zn*, generate a random r” Zn* output Ca = (A, B) = (r” pka, zrG + Pm), where Pm = f(m).

Re-encrypt(rka”b, Ca) ” (Cb). Given a re-encryption key rka”b and ciphertext Ca, compute Cb = (A”, B”) = (e(A, rka”b), B) = (e(raG, a-1bG), zrG + Pm) = (zrb, zrG + Pm).

Decrypt1(Ca, ska) ” m. Given ciphertext Ca and secret key ska, output Pm = B” [e(A, ska-1G)]G and m = f-1(Pm).

Decrypt2(Cb, skb) ” m. Given re-encrypted ciphertext Cb and secret key skb, output Pm = B” ” (A”1/b G and m = f-1(Pm).
Here 1 denotes the decryption method used by the delegator where 2 denotes the modified decryption method used by the delegatee.

Proof of Correctness:
Let”s assume the system parameters SP as {E, q, e, G, G1, G2, z, f, n}.

Alice”s private key as “a” and public key as Q = aG.
Bob”s private key as “b” and public key as R = bG.
So re-encryption key is calculated as rka”b = a-1bG.

Let”s assume Victor wants to send a message to Alice. So he chooses a random number r from Zn* and calculates the ciphertext pair as Ca = (A, B) = (rQ, zrG + Pm) and sends this ciphertext to Alice.
Now the proxy server will convert this Alice”s ciphertext into Bob”s ciphertext by using supplied re-encryption key. The new ciphertext Cb is calculated as follows,

Cb = (A”, B”) = (e(rQ, rka”b), zrG + Pm). And this calculated ciphertext is sent to Bob.
After receiving the ciphertext Cb, Bob will reconstruct the original message as follows,
\[ P_m = B^{e^{(A^a)}} 1/bG \]
\[ = P_m + zrG^{e^{(e(rQ, rka^{b})1/bG)}} \]
\[ = P_m + zrG^{e^{(e(raG, a^{-1}bG)1/bG)}} \]
\[ = P_m + zrG^{e^{(zrb)1/bG}} \]
\[ = P_m + zrG^{rG} \]
\[ = P_m \]

Finally the original message is derived by taking inverse of function \( f \) on \( P_m \), i.e. \( m = f^{-1}(P_m) \).

5.3 Analysis on Properties of Proposed Approaches

1. Unidirectional: A proxy re-encryption approach is called a unidirectional if delegation from \( A''B \) does not permit re-encryption from \( B''A \) in any way. Our second approach is unidirectional in nature whereas the first one is bidirectional in the sense that the proxy can apply inversion on the supplied re-encryption key to construct the re-encryption key to Alice from Bob i.e. \( rkb''a \).

Re-encryption key \( rka''b = b/a = ba^{-1} \)

Apply inversion on \( rka''b \), i.e \( (rka''b)^{-1} = (ba^{-1})^{-1} = b^{-1}a = rkb''a \)

2. Non-interactive: In a proxy re-encryption scheme, if Alice (delegator) is able to construct the re-encryption key to Bob (delegate) without interacting with Bob and/or any trusted third party, then it is called as non-interactive proxy re-encryption. In most of the non-interactive cases, re-encryption key will be calculated straight away by using Bob’s public key. Our first approach is a kind of interactive as explained below.

Re-encryption key \( rka''b = b/a = ba^{-1} \).

Here we need to consult Bob and/or trusted third party to construct re-encryption key as the re-encryption key contains Bob’s secret key i.e. \( b \).

On the other hand, our second approach is non-interactive.

Re-encryption key \( rka''b = a^{-1}bG \).

As we can see above, we need only Bob’s public key i.e. \( bG \) to construct re-encryption key.

3. Proxy invisibility: This is an important feature offered by the original BBS scheme. In their scheme, the ciphertext pair \( Ca = (r^{m^G} gr, m^G gr) \) is re-encrypted to \( Cb = (gbr, m^G gr) \) by proxy server. From this ciphertext \( Cb \), Bob (delegate) cannot recognize the involvement of proxy, converting Alice’s ciphertext into Bob’s ciphertext. Our first approach CTR1-PRE satisfies proxy invisibility feature whereas second approach CTR2-PRE provides weaker form of invisibility as explained below.

In CTR1-PRE, Alice ciphertext is of form \( Ca = (r^G aG, rG + Pm) \) which is converted to a ciphertext of Bob as \( Cb = (r^G bG, rG + Pm) \). Here both the ciphertext are of same domain i.e. point on elliptic curve. So Bob as a delegate cannot differentiate the contribution of proxy.

In CTR2-PRE, the ciphertext pair \( Ca = (r^G aG, zrG + Pm) \) is converted to \( Cb = (zrb, zrG + Pm) \). Here we can see that the proxy is converting the first parameter of Alice’s ciphertext which is a point on elliptic curve into some other domain’s value i.e. by a pairing value. So Bob can identify the involvement of proxy just by seeing the structure of the ciphertext he received. But he cannot identify the proxy’s involvement by seeing the re-encrypted ciphertext alone.

4. Original-access: If the delegator is able to decrypt the re-encrypted ciphertexts that were originally sent to him, then we can say that the proxy re-encryption is providing original access feature. For some applications, this could be a necessary feature to maintain the accessibility of the re-encrypted ciphertexts. In our approaches, we can achieve this property by including additional overhead along with the re-encrypted ciphertexts as shown below.

In CTR1-PRE, \( Cb = (rbG, rG + Pm)'' (rbG, rG + Pm, raG) \)

Table 2. Comparison of known Proxy Re-encryption Schemes

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unidirectional</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Non-interactive</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>Proxy invisible</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Original access</td>
<td>Yes</td>
<td>Yes*</td>
<td>Yes*</td>
<td>Yes*</td>
<td>Yes*</td>
</tr>
<tr>
<td>5</td>
<td>Key optimal</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Collusion resistant</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes*</td>
</tr>
</tbody>
</table>

*Note: Certain schemes may have additional properties, but the table above focuses on the core features mentioned.
7 Temporary Yes* Yes* Yes* Yes* Yes*
8 Non-transitive No Yes Yes No Yes
9 Non-transferable No No No No No

Here + indicates master secret key only. * Indicates possibility to achieve with additional overhead.

In CTR2-PRE, \( C_b = \left( z_r b, z_r G + P_m \right) \) → \( (z_r b, z_r G + P_m, r_a G) \)

5. Key optimal: If the size of delegate’s secret storage remains constant irrespective of how many delegations he receives, then the proxy re-encryption is key optimal. In the previous ElGamal and RSA based schemes [16], the storage size of both Bob (delegate) and the proxy server grows linearly with the number of delegations Bob accepts. But our both the approaches sustain the key optimal property.

6. Collusion resistant: If the secret key of delegator or delegate is not able to derive by colluding with the proxy server, then the proxy re-encryption is said to be collusion resistant. Our first approach CTR1-PRE suffers by this property in the sense that Alice can derive Bob’s secret key by colluding with proxy as follows:

In CTR1-PRE, Re-encryption key \( r_{ka}''b = b/a = ba^{-1} \). Therefore if she simple multiplies her secret key along with re-encryption key, she will get Bob’s secret key, i.e. \( ba^{-1}a = b \). Similarly, Bob can derive Alice’s secret key by colluding with proxy, i.e. \( ba^{-1}b^{-1} = a^{-1}; (a^{-1})^{-1} = a \).

second approach CTR2-PRE satisfies collusion resistant property as the re-encryption key is calculated from Bob’s public key, unless otherwise he reveals his secret key, Alice can’t derive his secret key even colluding with proxy. The same thing is applicable for Bob also.

7. Temporary: Dodis and Ivan [3] proposed applying generic key-insulation methods [18, 19, 20] to their approaches to form schemes where Bob is only able to decrypt ciphertexts intended for Alice that were authored during some precise time period. Our approaches can also be made to achieve this property by considering some extra overhead.

8. Non-transitive: If the proxy alone can’t re-delegate the decryption rights, then the proxy re-encryption is said to have non-transitive property. For example, given re-encryption keys \( r_{ka}''b \) and \( r_{kb}''c \), it can’t produce \( r_{ka}''c \).

Our first approach does not satisfy this feature in the sense that the proxy can derive re-encryption key \( r_{ka}''c \) (Alice to Charlie) given two re-encryption keys i.e. Alice to Bob (\( r_{ka}''b \)) and Bob to Charlie (\( r_{kb}''c \)) as follows.

\( r_{ka}''b = ba^{-1} \) and \( r_{kb}''c = cb^{-1} \)

\( r_{ka}''c = r_{ka}''b \cdot r_{kb}''c = (ba^{-1}) \cdot (cb^{-1}) = ca^{-1} \)

Our second approach is of non-transitive kind where proxy can’t re-delegate the decryption rights.

9. Non-transferable: If the proxy and a set of colluding delegates can’t re-delegate decryption rights, then the proxy re-encryption is said to have non-transferable feature. In other words, if Bob, Charlie and proxy server collude each other, then they can produce re-encryption key \( r_{ka}''c \) without Alice permission. Till now, there is no efficient approach to solve this problem.

6. CONCLUSIONS AND FUTURE WORK

In this work we proposed two proxy re-encryption methods namely CTR1-PRE and CTR2-PRE in the context of elliptic curve and showed that our second approach is having all the desirable features of proxy re-encryption. An interesting open problem is to find unidirectional, transitive proxy re-encryption technique. Our future work will focus on how to use these proposed schemes to do outsourcing operation in multi-party cloud environment.

7. REFERENCES


About Essay Sauce

EssaySauce.com is a completely free resource to help students research their academic work and learn from great essays!

View all posts by Essay Sauce

...(download the rest of the essay above)
About EssaySauce, the student essay site:

EssaySauce.com is a free resource for students, providing thousands of example essays to help them complete their college and university coursework. Students can use our free essays as examples to write their own.

Latest student essays:

Ocular disease
HUMAN action recognition
Analysing data production
Desorption study
Surfactants (surface active agents)
Islamic Finance and Its Impact on Customer Satisfaction
Persian gulf
Feminist approach (Bhumika) (notes)
What does it mean to be a Muslim woman in 21st century? (Shari‘ah)
Appellate Body’s analysis under section XIV(c)
Student essay categories:

Accounting essays
Architecture essays
Business essays
Computer science essays
Criminology essays
Economics essays
Education essays
Engineering essays
English language essays
English literature essays
Environmental studies essays
Finance essays
Geography essays
Health essays
History essays
Hospitality and tourism essays
Human rights essays
Information technology essays
International Relations
Law essays
Leadership essays
Linguistics essays
Literature essays
Management essays
Marketing essays
Media essays
Medicine essays
Miscellaneous essays
Music Essays
Philosophy essays
Photography and arts essays
Politics essays
Project management essays
Psychology essays
Religious studies and Theology essays
Science essays
Social work essays
Sociology essays
Uncategorized
Average review:

Overall rating: 0 out of 5 based on 0 reviews.

Q: Is EssaySauce.com free?

Yes! EssaySauce.com is a completely free resource for students. You can view our terms of use here.

Why use Essay Sauce?

The brightest students know that the best way to learn is by example! EssaySauce.com has thousands of great essay examples for students to use as inspiration when writing their own essays.

Is Essay Sauce completely free?

Yes! EssaySauce.com is a completely free resource for students. You can view our terms of use here.

Info:

About
Content policy
Essay removal request
Privacy
Terms of use