Design of security protocols in Cloud based Cyber physical system

Undertaken as M.Sc project under supervision of Prof. Adrijit Goswami

Submitted by

Satyam Kumar Jha

Roll No: 13MA20036

Department of Mathematics

IIT Kharagpur

Table of Contents

ACKNOWLEDGEMENT

I would like to express my gratitude towards **Prof Adrijit Goswami**, Department of Mathematics for offering me the opportunity to work on this project and continuously guiding me throughout the project work.

I would also like to acknowledge my faculty advisor **Prof. Debapriya Biswas**, Dept. Of Mathematics IIT Kharagpur for her precious guidance and advices.

I would finally like to thank **Head of Department** and all Professors of Department of Mathematics, IIT Kharagpur for their continuous guidance and support.

CERTIFICATE

This is to certify that the report entitled "**DESIGN OF SECURITY PROTOCOLS IN CLOUD BASED CYBER PHYSICAL SYSTEM**" submitted by Satyam Kumar Jha (13MA20036) to the Department of Mathematics, in partial fulfilment for the award of the degree of Master of Science, is an authentic record of the work carried out by him under my supervision and guidance. The report has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Prof. Adrijit Goswami Date:

Department of Mathematics

Indian Institute of Technology

Kharagpur - 721302

Abstract

Cyber Physical system(CPS), also known as smart system, is a network engineered for interaction between physical and computational component. CPSs provide abstractions, and design and analysis techniques to the whole system by integrating networking and software with the dynamics of physical processes. Similar to the Internet of things (IOT), the coordination and combination between computational and physical elements is higher in CPS. Applications of CPS typically involve sensor based autonomous systems. With advancement in Engineering and Science, the application dimensions of CPS or increasing due to their improving efficiency, safety, reliability, usability and autonomy.

These dimensions include Precision (in surgery and manufacturing) coordination (in Air Traffic control) efficiency in (optimising energy usage) augmenting human capabilities e.t.c. Some Technologies related to CPS are the Industrial Internet, **Smart cities** and **Smart grid.**

With advancement in Engineering and Science based application dimensions of Cyber physical system are now opening due to that improving efficiency, safety reliability, usability and autonomy. For providing on-demand access to shared processing resources Cloud Computing is necessary in order to reduce infrastructure cost. However the communication between entities in cloud-assisted CPS is vulnerable to **various attack such as man-in-the-middle, impersonation, relay , physical smart metre capture and privileged inside attacks**. Hence to ensure quality of service and information privacy and security is an important requirement in cloud-assisted CPS environment. Also most of the existing schemes proposed in the cloud environment are either susceptible to several known attacks or they are expensive in communication. To address all these issues we need to propose a new scheme for key agreement.

Scope

Propose a better authenticated key agreement schemes in the cloud assisted CPS environment.

Using better methods to communicate between clients so that client don’t have to contact the certification agency every time it needs to send some data to some other client.

To deal with all these arose problems we are going to use fully functional **Identity Based Encryption.**

Introduction

This report describes an implementation of Secure Sockets Language(SSL) protocol for secure communications . Current SSL standard is the most generally used cryptographic protocol,but this implementation uses **Identity Based Encryption(IBE)** which eliminates the requirement of certificates of server side. This new system, called IBE-SSL, involves the use of a private key generator (PKG) to create a private key for the server. The server can then use its private key to decrypt any messages sent to it by a client using the server's DNS name as a public key.

Background:

Cryptographers Ron Rivest, Adi Shamir and Leonard Adleman developed a revolutionary new method in 1977. Their method uses two keys: **a private key known only to the owner and a public key that can be given to anyone**. The system is based on a complicated mathematical formula involving large prime numbers and exponentiation. When someone wishes to send a message to a person, he or she must find that person's public key and encrypt the message. The recipient then decrypts the message with his or her private key and receives the plaintext.This method is commonly called **public-key cryptography**.

In an SSL system, the server generates its own public and private key pair and then publishes the public key to the Internet. Anyone wanting to transfer sensitive data can then use the server's public key to encrypt the data. However, if a hacker compromised the server and replaced the server's public key with the hacker's public key, then the hacker could intercept incoming ciphertext messages and decrypt them. To avoid this, the creators of SSL introduced **certificates**. One of the unique properties about the public / private key pair is that a person can encrypt a message with his or her private key, and then anyone can get the person's public key and decrypt the message.Since the person's public key decrypts the message, the corresponding private key must have encrypted it, so anyone who decrypts the message knows that the person sent it. This process is known as "signing." In SSL, a trusted party (called a certificate authority or CA) will issue a certificate for a server. Essentially, the certificate is the public key of the server encrypted with the private key of the signer. Web browsers have the public key of the CA embedded in their code and these keys are implicitly trusted on SSL's root level certificate trust model. Since web browsers (and users) trust that the CA has not given away its private key, they can decrypt the server's public key by decrypting with the CA's public key.

Problem:

In cyber physical system every component is susceptible to man in the middle attack. With the traditional Public key infrastructure it is tedious to maintain public-private key pair for all the components also certificate can be revoked since we will be needing different certificate for all the components.

Solution:

If SSL used the IBE system instead, there would only be the need for one certificate: **the master certificate** embedded in web browsers for the PKG. The browser would then encrypt data with the unique identity of each component and send the encrypted text to the server. The server would get the encrypted message from the client and decrypt it with its private key that it has received from the PKG. There is no need for the server to have its own certificate: it merely needs the private key generated from the PKG to decrypt any message sent to it.

This process is shown graphically below in Figure (a)

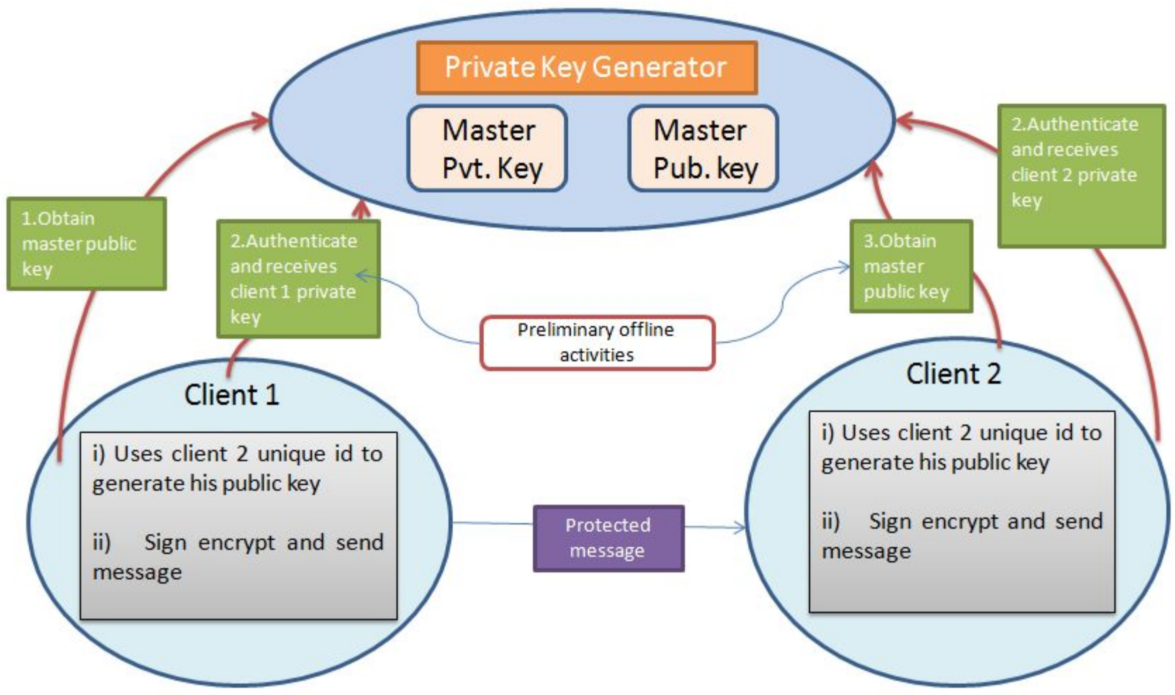

Private Key Generator
Master Pvt. Key
Master Pub. key
1.Obtain master public key
2.Authenticate and receives client 1 private key
Preliminary offline activities
3.Obtain master public key
2.Authenticate and receives client 2 private key
Client 1
i) Uses client 2 unique id to generate his public key
ii) Sign encrypt and send message
Protected message
Client 2
i) Uses client 2 unique id to generate his public key
ii) Sign encrypt and send message

Figure (A) Graphical representation of Identity based Encryption

Properties of IBE -

1. It is Public key encryption
2. Public key is based on unique info about owner's identity.
3. PKG generates user Private key based on their unique id.
4. Central Authority creates master public key and master private key . The master public key is shared with all client to allow them to compute other client public keys without requiring them to communicate with each other.

Implementation

Process to set up new client in IBE -

1. User contacts the Central Authority to receive the master public key.
2. The user combines the master public key with their identity value to create their public key.
3. The user contacts the Central Authority to generate their private key.

Basic steps in IBE

1. **SETUP**

   PKG generates the master key which is used to derive the user private key.

   PKG(k) = P & Km

   P : set of system parameters including M (message space) and C(ciphertext space).

   Km : the master private key

   K : Security parameters, binary length of key material.

   -- Similar algo is used to create public key too

2. **EXTRACT**

   Algorithm is used by PKG when a private key is requested by a user.

   PKG(P,Km ,ID) = d

   d : the private key for the user ID

   ID ∈ (0,1)*

3. ENCRYPTION

Encrypt (P,M,ID) = C

M-message

C-ciphertext

4. DECRYPTION

Decrypt (P,d,C) =M

IBE - SSL Implementation

In this section, steps taken has been outlined to show that the IBE system can be applied to a traditional SSL implementation. I have chosen to write demonstration programs in java to show a proof of concept. Main functions of the implementation are described below:

Client method

We take ID of user (sender) ,ID of the the receiver ,and MESSAGE which is to be send . Then receive the public key corresponding to the ID of receiver and 'n' which is p*q from PKG. And then encrypt the message using this public key by RSA.Then store the encrypted message along with USER ID in a file named "EncryptedMessage.txt".

Encrypt message using C = M^e mod n where 0 ≤ M < n.

```
private byte [] messageEncrypt(byte [] message)
{
      byte [] EncryptMessage = (new
      BigInteger(message)).modPow(Public_key, n).toByteArray();
      return EncryptMessage;
}
```

Server Methods

We take ID of user (Receiver) and private key corresponding to his ID is provided by PKG. Encrypted messages along with sender ID from the file

“EncryptedMessage.txt” is taken. After this decrypt the messages using private

key by RSA and show it to USER along with sender’s ID.

Decrypt message using M = C^d mod n

```
public byte[] decryptMessage(byte[] encrptedmessage) {
      byte[] message = (new BigInteger(message)).modPow(private_key,
      n).toByteArray();
      return message;
}
```

This function will return message in byte form ,so we have to convert it into string form.

Private key Generator Methods

PKG will generate the public key and private key corresponding to a ID which is an “Domain Name” using RSA. How RSA is used to generate public key and private key for a “Domain Name” is shown below.

1. For generating public key corresponding to ID using RSA

```
public BigInteger get_public_key()
{
      phi =
      p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
      e = BigInteger.valueOf(public_key_temp);
      while (phi.gcd(e).compareTo(BigInteger.valueOf(1)) != 0 ) {
      e = e.divide(phi.gcd(e));
}
public_key = e;
get_private_key();
return public_key;
}
```

Where p and q are Big PRIME NUMBERS, public_key_temp = Math.abs(ID.hashCode()) and phi = (p-1)*(q-1). In this method we find a number which is relatively prime to phi by continuously dividing public_key_temp by gcd(public_key_temp,phi) ,until gcd(public_key_temp,phi) = 1,and select it as an public key.

2. For generating private key corresponding to ID using RSA

```
public BigInteger get_private_key(){
      d = public_key.modInverse(phi);
      private_key =
      extendedEuclid(public_key,(this.p.subtract(BigInteger.ONE)).multi
      ply(this.q.subtract(
      BigInteger.ONE)));
      return private_key;
}
```

We calculated the modular inverse of public key with respect to phi and select it as

Public key.

. **3. Extended Euclidian Method**

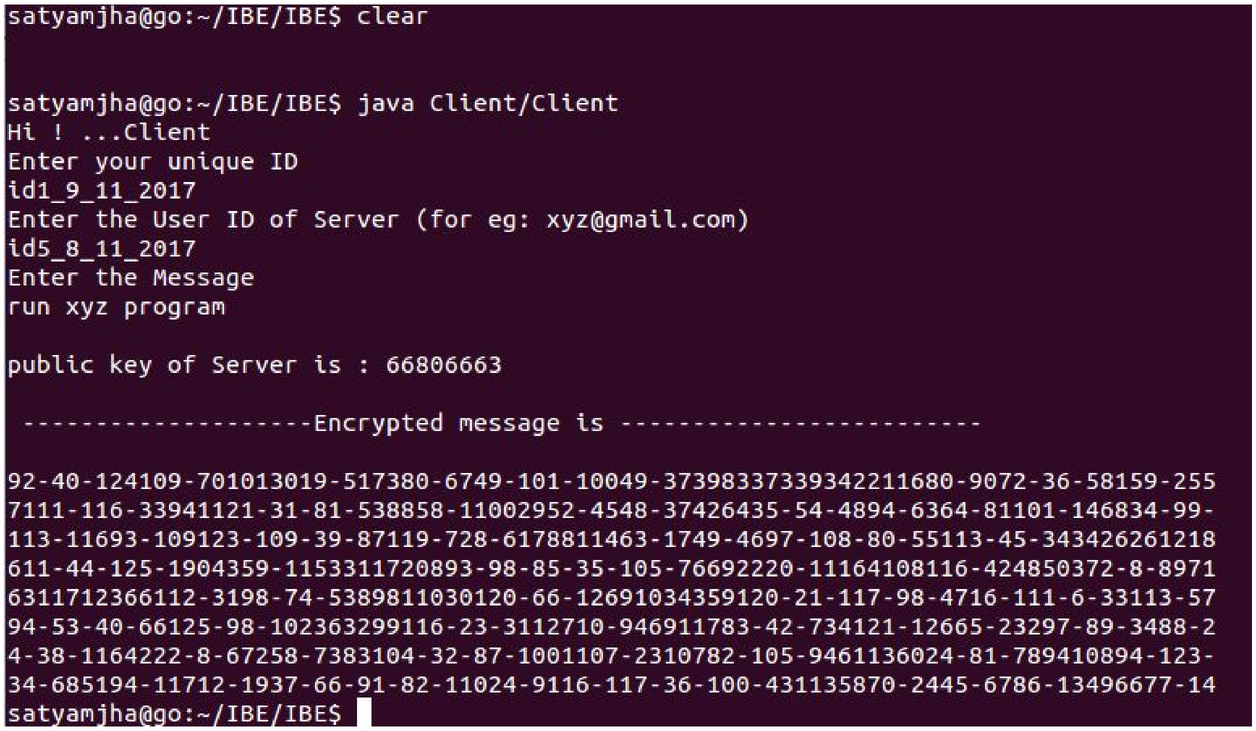
```
public BigInteger extendedEuclid(BigInteger a, BigInteger b) {
      BigInteger x = BigInteger.valueOf(1);
      y = BigInteger.valueOf(0);
      BigInteger xLast = BigInteger.valueOf(0);
      yLast = BigInteger.valueOf(0);
      BigInteger q, r, m, n;
      while(a.compareTo(BigInteger.valueOf(0)) != 0) {
      q = b.divide(a);
      r = b.remainder(a);
      m = xLast.subtract(q.multiply(x));
      n = yLast.subtract(q.multiply(y));
      xLast = x;
      yLast = y;
      x = m;
      y = n;
      b = a;
      a = r;
}
if(xLast.compareTo(BigInteger.valueOf(0))<0)
xLast =
xLast.add((this.p.subtract(BigInteger.ONE)).multiply(this.q.subtract(B
igInteger.ONE)));
return xLast;
}
```

This method is used to find the modular inverse of a number with respect to a number which is relative prime to it.

Results and discussion

A message exchange between two clients would look like this :

(note- this is just a model):



```
satyamjha@go:~/IBE/IBE$ clear


satyamjha@go:~/IBE/IBE$ java Client/Client
Hi ! ...Client
Enter your unique ID
id1_9_11_2017
Enter the User ID of Server (for eg: xyz@gmail.com)
id5_8_11_2017
Enter the Message
run xyz program

public key of Server is : 66806663

 --------------------Encrypted message is -------------------------

92-40-124109-701013019-517380-6749-101-10049-37398337339342211680-9072-36-58159-255
7111-116-33941121-31-81-538858-11002952-4548-37426435-54-4894-6364-81101-146834-99-
113-11693-109123-109-39-87119-728-6178811463-1749-4697-108-80-55113-45-343426261218
611-44-125-1904359-1153311720893-98-85-35-105-76692220-11164108116-424850372-8-8971
6311712366112-3198-74-5389811030120-66-12691034359120-21-117-98-4716-111-6-33113-57
94-53-40-66125-98-102363299116-23-3112710-946911783-42-734121-12665-23297-89-3488-2
4-38-1164222-8-67258-7383104-32-87-1001107-2310782-105-9461136024-81-789410894-123-
34-685194-11712-1937-66-91-82-11024-9116-117-36-100-431135870-2445-6786-13496677-14
satyamjha@go:~/IBE/IBE$ 
```

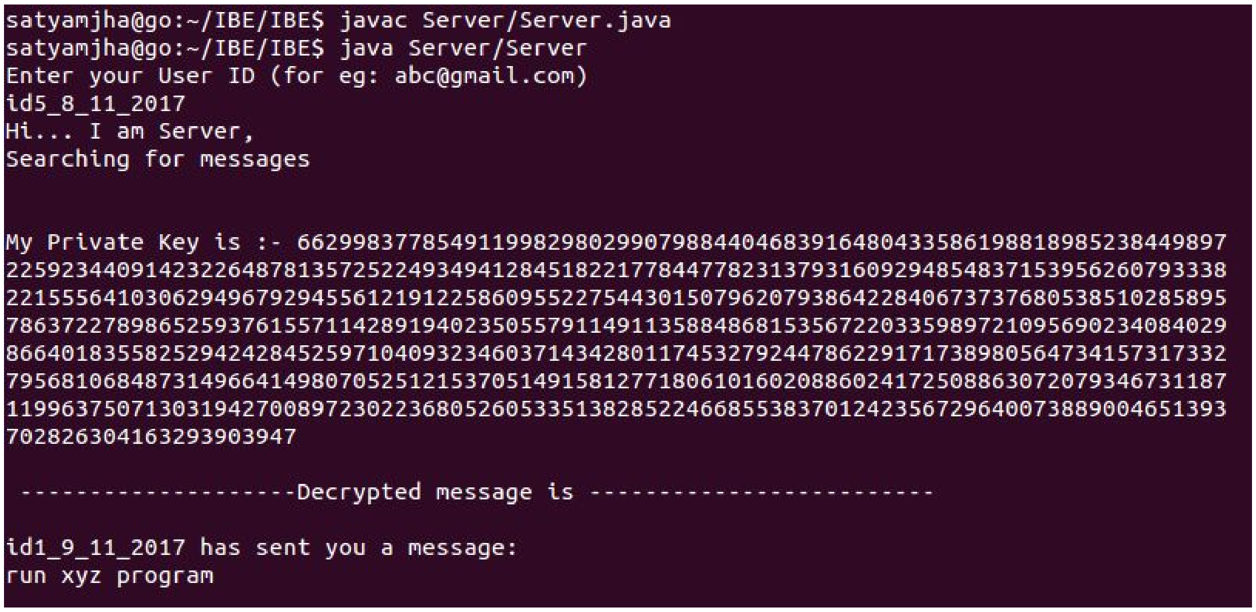
NOTE:

Client 1 has **date stamp** (9_11_2017) attached with the unique ID which they can update after some time which insures extra security. We don't need to revoke certificates like it happens in PKI . The public key and the private key of a particular client will automatically change once the id is changed. This will further make sure that there is no man in middle attack.

Here the message from client 1 to client 5 has been send.

After the encryption the encrypted message lies in the server getting ready to be decrypted . The encrypted message would only be decrypted by using private key of the receiver. The private key of the receiver is obtained by receiver's unique ID and Master key (obtained from the private key generator)

This is how a decryption process looks like for client 5:



```
satyamjha@go:~/IBE/IBE$ javac Server/Server.java
satyamjha@go:~/IBE/IBE$ java Server/Server
Enter your User ID (for eg: abc@gmail.com)
id5_8_11_2017
Hi... I am Server,
Searching for messages


My Private Key is :- 6629983778549119982980299079884404683916480433586198818985238449897
2259234409142322648781357252249349412845182217784477823137931609294854837153956260793338
2215556410306294967929455612191225860955227544301507962079386422840673737680538510285895
7863722789865259376155711428919402350557911491135884868153567220335989721095690234084029
8664018355825294242845259710409323460371434280117453279244786229171738980564734157317332
7956810684873149664149807052512153705149158127718061016020886024172508863072079346731187
1199637507130319427008972302236805260533513828522466855383701242356729640073889004651393
702826304163293903947

 --------------------Decrypted message is ------------------------

id1_9_11_2017 has sent you a message:
run xyz program
```

NOTE:

Once the connection is established between the client and the central authority it need not generate the private key again and again.

Here we have just shown a moment from the simulation of message exchange that would take place between the clients in the cyber physical system.

Advantages of using identity based encryption

- IBE provides complete resistance against man in the middle attacks and the replay attacks.
- Rather than storing big databases of public keys system can derive these public keys from the unique id. Thus memory required for storing identities/public keys is much lesser in IBE than in PKI and this difference increases with increase in the number of device communicating.
- End devices will be able to decrypts the package that they have received if and only if they are encrypted with its corresponding identity . This proves authenticity at the other end without the need of certificates.

Further improvements required

- Central authority is super powered. It can encrypt and decrypt any message, so it has to be really authentic.
- Msg channel between PKG and clients should be secured.
- Make sure there is no reuse of the among the clients.

Conclusion

We have managed to provide a better and robust method of authentication **using identities of each of the client** of the cloud based cyber physical system.

This system is completely secure against the **man in the middle attacks and the replay attacks** which are the major threat in the cloud based systems.

This system also require **lesser memory** to store the identities rather than storing the public keys which is much better for the system with many components.

References

[1] J.Baek, Q.H. Vu, J.K.Liu, X.Huang, and Y.Xiang. A secure Cloud Computing based Framework for Big data Information management of Smart Grid. IEEE Transaction on Computing, 3(2):223-244, 2015

[2] S.Rho, A.V. Vasilakos, and W.chen. Cyber Physical system technologies and applications - Part II. Future Generation Computer Systems, 56:449-475, 2016

[3] A. Socievole, A. Ziviani, F. De Rango, A. V. Vasilakos, and E. Yoneki. Cyber-physical systems for Mobile Opportunistic Networking in Proximity (MNP). Computer Networks, 111:1-5, 2016

[4] S.Misra, S.Bera and T.Ojha. D2P: Distributed Dynamic Pricing Policy in Smart Grid for PHEVs Management. IEEE Transactions on Parallel and Distributed Systems, 26(3):702-712, 2015

[5] X. Fang, S.Misra, G.Xue, and D. Yang. Managing smart grid information in the cloud opportunities, model and applications. IEEE network, 26(4):32-38,2012