Security Algorithms for MPCS Architecture
This chapter presents and discusses the security algorithms developed for device authentication, client and client/server authentication, secure transmission of Payment Request (PR) from mobile client to institution server, secure communication for exchanging the public keys and secure transmission of user profile between the Institution Server (IS) and Mobile Payment Consortia System (MPCS) server. There is also an explanation on authorizing the mobile user for availing the mobile payment services.

4.1 Security Algorithm for Device Authentication

The security algorithm for device authentication is to authenticate the entities such as client mobile device and IS with X.509 based digital certificates. The services of Certification Authority (CA) are used for issuing the certificates. The process of using security interfaces are explained in the previous chapter. After the successful completion of registration, the client can download the payment software at client mobile device. When the mobile client activates the security interface at mobile device, then the proposed security interface sends the client certificate to the IS. As and when an institution receives the client certificate, then the security interface developed for the IS validates the client certificate by calculating a hash of the To-Be-Signed (TBS) structure of the certificate and comparing the result with the decrypted signature of the certificate. In the mean time, the client-side interface also starts the validation process for IS's certificate using Online Certificate Status.
Protocol (OCSP) and obtains the certificate revocation status. If the status is positive, the security interface carries out the client/server authentication process.

Figure 4.1 depicts the device authentication process algorithmically.

```plaintext
Algorithm DeviceAuthN(MUC, URL)
{
    Mobile User sends Certificate (MUC) to the Institution Server (IS);
    IS obtains CDF from user certificate and retrieves CRL from CDF;
    IS examines CRL for the serial number of the MUC;
    IS queries CDF for CRL to verify the validity of the MUC;
    CRL is returned from CA’s CRL database (CDP) to IS;
    IS examines the CRL for the validation of the User Certificate;
    IF serial number of MUC is not found in new CRL
    IF(Sent MUC(Public Key & Validity) = CRL MUC(Public Key & Validity))
    {
        Client device is authenticated;
        IS sends X.509 based server certificate to the mobile user;
        Mobile user device initiates communication with CA Server;
        Mobile user device sends serial number of server’s certificate and
        Requests the server certificate status from OCSP Server;
        OCSP server returns the status for requested server certificate;
        IF (OCSP response = ‘good’)
            The Institution Server is identified and authenticated;
        ELSE (OCSP response = ‘revoked’ or ‘unknown’)
            Invalid Server Certificate;
    }
    ELSE
    {
        Invalid User Certificate (MUC);
        Exit( );
    }
}
```

Figure 4.1: Algorithm for Device Authentication
4.2 Security Algorithm for Client Authentication

The proposed security algorithm for client authentication is to authenticate the mobile client, whether the user is genuine for handling the mobile payment services or not. The security algorithm prevents the unauthorized access of payment software when the client mobile device is stolen or lost.

Algorithm ClientAuthN(AcCode)
{
    GET AcCode; //User enters the Activation Code
    FIND IMEI number of the mobile user device;
    //User interface finds the IMEI number of the user device
    Generate Secret-Key using AcCode and IMEI-no;
    CAuthN = decrypt(server public key using Secret-Key);
    WHILE (Attempts ≥ 3 times)
    {
        IF (Server Public Key = valid)
        {
            Mobile user is authenticated;
            Exit( );
        }
        ELSE
        {
            Invalid AcCode message to Mobile user device;
            User Interface prompts for AcCode; /*If user attempts more than three times, user’s Mobile Payment Service account is blocked*/
        }
    }
}

Figure 4.2: Algorithm for Client Authentication

86
4.3 Security Algorithm for Client/Server Authentication

The security algorithm for client/server authentication validates the identity of both the mobile client and the institution server to carry out mobile payment transaction in a secure way. After completing the client authentication initially, the client user interface prompts for UName and PWD for second level authentication. When the client enters value for them, the security interface sends UName and PWD to the IS as a secure message by using standard security methods of Public Key Infrastructure (PKI). After the institution server receives UName and PWD, the server interface starts to decrypt them. If the decryption is done successfully, the institution server is authenticated to the mobile client. Finally, the authentication interface validates the UName and PWD with the database records available in IS and checks the identity of the mobile client. If the latter matches, the security interface sends the user profile to the IS, or else, the server interface sends an error message to the mobile client device. Figure 4.3 presents the algorithm for client/server authentication.

```
Algorithm CSAuthN(UName, Pwd)
{
    //Client side
    GET UName and PWD;//User enters username and password;
    MD = hashing (UName & PWD) using SHA1 algorithm;
    EMD = encrypt(MD using secret key);
    CSAuthNDS = sign(EMD using client’s private key);
    CSAuthNM = copy(UName and PWD);
    CSAuthNEM = encrypt(CSAuthNM using IS’s public key);
    Send CSAuthNDS and CSAuthNEM to the Institution Server;
}
```
//Institution Server side

Find and validates MSISDN of the user request from the user profile;

IF MSISDN is valid then

{ Extract the client’s public key, UName and PWD from the ISDB
  Based on respective MSISDN;
  CSAuthNM = decrypt(CSAuthNEM using IS’s private key);
  IF (ISDB:UName & PWD = CSAuthNM:UName & PWD)
  {
    CSAuthN = de-sign(CSAuthNDS using client’s public key);
    MD = decrypt(CSAuthN using secret key);
    NMD = hashing (CSAuthNM) using SHA1 algorithm;
    IF (MD = NMD) then
    {
      Client and IS are authenticated;
      Message integrity is ensured;
      IS sends Welcome Message to the user device;
    }
  }
} ELSE
Send error message for invalid UName and PWD;

} ELSE
Access Denied; Exit ();

Figure 4.3: Algorithm for Client/Server Authentication
4.4 Security Algorithm for IS and MPCS Authentication

Once the client/server authentication process is completed successfully, the user interface at IS establishes communication with MPCS server and starts the authentication process. Since, both the IS and MPCS server are connected to the institution network, a lightweight security mechanism is required for authentication between IS and MPCS server. The proposed security system adopts challenge-response mechanism to authenticate each other. At the time of authentication, a secure tunnel is established between the IS and MPCS server. The MPCS interface extracts the shared secret PIN number from the database and sends the encrypted RAND number using PIN. The security interface at IS decrypts the RAND number using the same PIN, calculates and sends the SRES as encrypted message using shared PIN number. Finally, the security algorithm presented in MPCS server verifies the SRES sent by IS. If both the SRESs are matched, then the MPCS interface authenticates the institution server. The IS/MPCS authentication process is depicted in Figure 4.4.

Algorithm ISMPCSAuthN(RAND-no)
{ 
   //Institution Server side
   //IS establishes handshake with MPCS for secure communication.
   GET RAND-no; //MPCS sends RAND number to IS;
   Extract PIN-no from IS database;
   RAND = decrypt(RAND-no using shared PIN-no between IS and MPCS);
   IF RAND is decrypted successfully
   { 

   An Architecture for Secure Mobile Payment System using Public Key Infrastructure
MPCS server is identified and authenticated;
SRES = compute SRES (using RAND and PIN-no);
ESRES = encrypt(SRES using shared PIN-no);
Send ESRES to MPCS;
}
ELSE
{
Invalid MPCS Server; Exit ( );
}
}

// MPCS Server side
{
Extract the PIN-no from the MPCS database;
SRES1 = compute SRES1 (using RAND-no and PIN-no);
SRES = decrypt(SRES using shared PIN-no);
IF SRES is decrypted successfully
{
Institution Server is identified and authenticated initially;
IF (IS:SRES = MPCS:SRES1) // MPCS compares SRES and SRES1.
{
Institution Server and MPCS Server are authenticated;
ISMPCSAuthN = Encrypt(AuthNMessage using PIN-no);
Send ISMPCSAuthN to IS;
}
ELSE
{
Invalid Institution Server (IS); Exit ( );
}
}
}

Figure 4.4: Algorithm for IS/MPCS Authentication
4.5 Security Algorithm for Exchanging the Public Keys between IS and MPCS

After completing the authentication process, the security interface exchanges the public keys between IS and MPCS using PKI that facilitates secure communication. There are two keys utilized in security algorithm. One is for encryption and other is for decryption. The decryption key is a private key that must be kept secret. The corresponding encryption key, known as the public key, can be sent to communicating parties, including Certificate Authority. The Public key infrastructure works as follows: If the IS has to send a message to the MPCS server, then the institution server needs to obtain MPCS’s public key and use it for encrypting the message. In order to respond, the MPCS server receives its public key from the IS. The encrypted message is sent to the MPCS server. Only the MPCS server is able to decrypt this message, because the corresponding decryption key is known only to the IS. The MPCS server can respond to the IS in the same way.

During the exchange of public keys between IS and MPCS, the shared PIN number is used by the security interface for encrypting and decrypting the keys. The symmetric key mechanism supports confidentiality for keys using PIN number, since it is only known to the IS and MPCS. The proposed algorithm also creates hash code for encrypted keys using digest algorithm, by which the security interface ensures the integrity of message during the transmission. Figure 4.5 shows the exchange of public keys between IS and MPCS server.
Algorithm PuKExchange(ISMPCSAuthN)
{
    //Institution Server side
    IF ISMPCSAuthN is valid then
    {
        Let ISPUK = IS’s Public key;
        EISPUK = encrypt(ISPUK using shared PIN-no); //Let M1 = EISPUK;
        //Generate message digest for M1 using SHA-1 algorithm;
        MDK = hashing(EISPUK); //Ensures data integrity during OTA.
        SMDK = sign(MDK using IS’s private key); //Let M2 = SMDK;
        ISPuK = concatenate (M1 & M2); //IS sends ISPuK to the MPCS server.
    }
    //MPCS Server side
    {
        ISPK = Split (ISPuK);
        //Let ISPK{M1} = EISPUK; Let ISPK{M2} = SMDK;
        IF PIN-no is valid then
        {
            ISPKM1 = decrypt (ISPK{M1} using PIN-no);
            DSM2 = de-sign(ISPK{M2} using IS’s public key);
            MDISPKM1 = hashing (ISPKM1) using SHA-1 algorithm;
            MPCS compares DSM2 and MDISPKM1;
            IF (DSM2 = MDISPKM1) then
            {
                IS’s public key (message) integrity is ensured;
                MPCS stores ISPKM1 into MPCSDB;
                //MPCS sends its public key to the IS in reverse order.
            }
        }
    }
}
Figure 4.5: Algorithm for Exchanging the Public Keys between IS and MPCS

4.6 Security Algorithm for Transmission of User Profile

Once the public keys are exchanged between the IS and MPCS, the security interface of the IS sends user profile to the MPCS server using PKI mechanism. The security algorithm provides interface for IS to create the digital signature for user profile using IS’s private public key. The creation of signature for user profile achieves the important security property of non-repudiation. Then the interface creates digest for signed message to ensure the integrity while transmitting the information from IS to MPCS server. Finally the IS interface sends both the signed message and digested message to the MPCS server. The interface developed for MPCS server decrypts the message using IS’s public key and verifies integrity of the message. If the message is altered or modified, MPCS interface sends an error message to the institution server. Figure 4.6 depicts the encryption and decryption process of sending user profile.
Algorithm SendUserProfile( )
{
    \textit{IS Server side}
    
    IS extracts MPCSPKM1, ISPvK, UProfile from ISDB;
    //Create digital signature for UProfile to ensure message integrity.
    SUProfile = sign(UProfile using ISPvK);
    
    //Generate digest for signed user profile.
    MDSUProfile = hashing (SUProfile using SHA-1 algorithm);
    EMDSUProfile = encrypt(MDSUProfile using MPCSPKM1);
    Send SUProfile and EMDSUProfile to MPCS

    \textit{MPCS Server side}
    
    MPCS receives SUProfile and EMDSUProfile from IS;
    Extract MPCSPKM1, ISPKM1 from MPCSDB;
    
    //MPCS de-signs the SUProfile using IS’s public key.
    UProfile1 = de-sign(SUProfile using ISPKM1);
    
    //MPCS decrypts EMDSUProfile using MPCS’s public key.
    MDSUProfile1 = decrypt (EMDSUProfile using MPCSPKM1);
    
    //Creates digest for sent SUProfile1 to check the integrity of the message.
    MDSUProfileNew = hashing (SUProfile1 using SHA-1 algorithm);
    
    //MPCS compares both the messages.
    IF (MDSUProfile1 = MDSUProfileNew) then
        
        User profile (message) integrity is ensured;
        MPCS stores UProfile1 into MPCSDB;
    }
Figure 4.6: Algorithm for Secure Transmission of User Profile

4.7 Security Algorithm for Payment Authorization

Once the user profile reaches MPCS server, the latter sends the payment authorization interface to collect the fee option and fee amount. The security algorithm supports payment authorization by the mobile client by verifying the IMEI number and fee amount. At the time of payment authorization, a secure communication channel is established between the mobile client and MPCS gateway. Once the IMEI of the client device is verified by the MPCS interface, it prompts for fee option and fee amount at client mobile device. When the client selects the fee option from the list and enters the value for fee amount, the authorization interface collects them and brings to the MPCS server in a secure way using public key cryptography.

The significant process here is that the proposed algorithm creates digital envelop for fee option and amount. This process ensures message confidentiality at high level during the unsecure radio transmission. The corresponding private key owner
alone can view the digital envelop. Once the MPCS server receives the digital envelop, the interface decrypts them to obtain the option-id and fee amount.

Finally, the authorization interface establishes the connection with database and compares the value of fee amount. If the fee amount and the respective IMEI of the client mobile device are matched, the interface authorizes the mobile user to avail the payment services. After the successful completion of payment authorization service, MPCS interface creates a request for payment to the institution authorized bank. Figure 4.7 represents the payment authorization service done at mobile client device and MPCS server.

**Algorithm** PaymentInterface(Option-Id, Amount)

```plaintext
//Client-side functionality
The payment interface validates the user's mobile device by verifying IMEI_no.;

IF IMEI_no is valid then
{
    The mobile user device is authenticated;
    The user interface is enabled;
    Mobile user selects an option(option-id) from the list of fee options;
    GET option-id;
    GET amount;
    MD=Hashing(Option-Id and amount);
    //Create fixed-length message digest from the two variable-length messages;
    DSIG = Sign(MD using user's private key);
    //Create digital signature for the Message Digest;
    EnPReq = Encrypt(option-id, amount using the secret key);
    //create digital envelop for the message M;
    CMsg=Concatenate (DSIG, EnPReq);
    Let X = part of an IMEI of the intended user device;
```
M1 = Encrypt(CMsg using MPCS’s public key);
// M1 is the encrypted message.
M2 = Encrypt(MPCS’s public key using X);
// M2 is the encrypted key.
DE = Create{M1, M2};
// Create digital envelop for messages M1 and M2.
// The mobile user device packs encrypted document and encrypted
// message key into a single digital envelop and sends it to the MPCS.
// The result is a secure digital document that only the owner of the
// corresponding private key can view this document.

} ELSE
{
  Access denied;
  Exit ( );
}

// Server-side functionality
{
  // MPCS Server receives Digital Envelop from the client and checks MSISDN;
  X1 = Decrypt(M2 using X1, where X1 is the part of an IMEI number);
  // MPCS Server validates the mobile user device with the remaining part of an IMEI;
  // If any message comes from the un-available list of IMEI, MPCS Server denies the
  // opening of message and considers it as an unsolicited message.

  Let X1 = part of an IMEI of the intended user device;
  X2 = remaining part of an IMEI of the intended user device;
  IMEI = intended sender’s IMEI number.
  IMEI1 = X1 + X2;

  IF !(IMEI = IMEI1)
  {
    An Architecture for Secure Mobile Payment System using Public Key Infrastructure
Unsolicited Message from the invalid user;
Exit( );

ELSE
{
    MPCS Server validation is completed successfully;

    DM1 = Decrypt (M1 using MPCS’ private key);
    //The result is concatenated message that contains encrypted SIG and
    //encrypted option-id and amount. Then MPCS server splits the concatenated
    //message into X and Y.
    X, Y = Split (DM1);  //X=first bit to n bit; Y=n+1 bit to last bit;
    //Where, X = Encrypted digital Signature, Y = Encrypted option-id, amount.
    UMD = De-sign(X using user’s public key);
    PReq = Decrypt(Y using secret key);
    MPCSMD = Hash(PReq);

    IF (UMD = MPCSMD)
    {
        Integrity of the message is ensured;
        Exit ( );
    }

    ELSE
    {
        Access Denied;
        Send error message to the respective user’s device;
        Exit ( );
    }

}

Figure 4.7: Algorithm for Payment Authorization Service

98
4.8 Significance of the Proposed Algorithms

The significance of the proposed algorithms is as follows:

♦ The security algorithms provides strong authentication and avoids both technical errors and attacks-by-tampering. The multi-factor authentication has now become the standard method of authentication. The proposed model deploys multi-factor authentication scheme to heighten the assurance of the parties involved using AcCode, IMEI number, UName and Pwd, and digital certificates. Such a mode of authentication enhances trust relationship between the communicating parties.

♦ The proposed algorithm provides strong non-repudiation up to the standard set by payment industries such as Mobey Forum and Mobile electronic Transactions (MeT) Ltd., by utilizing public key certificate and digital signature.

♦ During the Device authentication, it is possible to fake easily certificate's identity. It can be sniffed by intruder. However, the proposed algorithm prevents such vulnerabilities by encrypting the entity's public keys using shared secret key and binding them onto the digital certificates. The secret key is generated by using AcCode and IMEI number and this key is known only to the involving parties. This mechanism also prevents replay attack.

♦ The MPCS sends the payment interface to the mobile client by adopting digital envelop mechanism. Such a method not only solves the key management problems, but also adds up to the performance by doing direct encryption of data without sacrificing the security.
To validate the institution server’s X.509 certificate, OCSP was used for obtaining messages of certificate revocation status by the client device. The establishment of certificate validity by checking CRLs through effective yet it is inadequate for certificate validation in low computation devices like mobile devices. However, OCSP provides a more real-time mechanism for certificate status checking without any over-burdens like using less bandwidth.

Since the proposed algorithms are developed using Public Key Infrastructure, which also incorporate and ensure the security needs of business transactions in an integrated manner, they provide a cost-effective means to manage security administration and operations for financial applications like mobile payment applications. Thus, MPCS supports effectively business requirements of the organization.

In cases of mobile phones being stolen and the culprit attempts to misuse the payment application, the mobile user should suspend immediately the wireless service connection to protect oneself against charges and illegal access. However, the security algorithms presented here gets activated only with registered user’s SIM number as well as with IMEI number. If the impostor tries with his/her own SIM card, the payment software will neither entertain nor support such services. As the user’s personal data also are stored as encrypted format in client devices it cannot reveal them to any burglar.

The proposed algorithm also offers protection from “Known plain text” attack, since the server generates secret key and validates IMEI number by interface itself than getting input from the user. Besides, the user interface encrypts the
data with standard asymmetric key cryptography. Even if the hackers were to tamper with any data using KeyLogger software or Flexispy, it will not be useful for them because any sensitive information is not transmitted during the communication. If modifications or alterations are introduced by the trespassers, the receiving node will easily identify such entities.

While considering the malware and spyware, the institution server allows the user to install the payment application only after verifying the valid SIM number and IMEI number. Such a payment application is sent by the IS via Over-The-Air (OTA) through secure API using SSL protocol. Hence, the system is secure from all possible kinds of attacks like phishing, virus attacks, and others.

An open PKI-based platform facilitates the development of a wide range of secure mobile payment applications. It provides an open technological solution to secure mobile payment transactions that can be freely utilized by financial institutions, mobile network operators or independent third parties. The MPCS model relies on PKI-based platform to perform payment transactions. Hence, Man-In-The-Middle (MITM) cannot decrypt payment transaction information.