Experimental Study and Performance Analysis
Chapter V

EXPERIMENTAL STUDY AND PERFORMANCE ANALYSIS

The main focus of the experimental study is to measure the time spent for the encryption using Rivest, Shamir and Adleman (RSA) algorithm at the hand held devices in the General Packet Radio Service (GPRS) network during device authentication, client authentication and client/server authentication. The performance of the proposed system is investigated in terms of response time and throughput. The proposed architecture for secure mobile payment system is examined in test bed environment. The results of the experiments are tabulated and graphically presented. The computation cost for proposed system is calculated mathematically and the result of comparative study with existing system is presented in this chapter. Finally, the simulated results are presented for system response time and throughput.

5.1 Experimental Setup

The security architecture of the proposed payment system is simulated and tested in academic institution environment by establishing a Test Bed. The development phase starts with the requirement document delivered by the requirement phase and maps the requirements into architecture. The user interface for client’s mobile device application and the Mobile Payment Server application are developed. At the same time, the test bed is also implemented to test the performance of the proposed system. The developing phase has been divided into two parallel processes. In the
first process, mobile client application is developed. This process consists of two phases such as Coding and Testing. This is an iterative process. The developed codes are tested in the testing phase. Again it goes to the design phase to check whether there is any failure or modification in testing. The mobile payment server applications are built during the second process. This process also consists of two phases such as Coding and Testing. After completing this development phase, the integration of client and server applications is investigated with the test bed.

Choosing the right structure for an application is critically important. Bad architectural choices usually cannot be fixed during implementation, no matter how good the developers are. Making the wrong decisions leads to lower performance, less security, and fewer options when an application needs to be updated. Hence, the multi-tier architecture has been effectively deployed on an experimental test bed.

Figure 5.1: Test Bed Environment

An Architecture for Secure Mobile Payment System using Public Key Infrastructure
to implement the proposed payment system, where the application components are decomposed logically into various tiers that makes easier to build, easier to reuse and easier to modify. In order to investigate the performance of the various security levels, the experimental set up was established. Figure 5.1 depicts the test bed environment.

5.2 Hardware Setup

The Hardware setup of the mobile payment system is divided into six entities. They are Mobile Device that represents the client, Institution Server (IS), Institution Database server, Mobile Payment Consortia System (MPCS) server, Certificate Authority (CA) server and Institution authorized Bank server. Sony Ericsson K750i - Java enabled mobile phone is used for testing the secure user-interface through GPRS network. This device supports Mobile Information Device Profile (MIDP) 2.0 related Application Programming Interface (APIs) such as Wireless Message Access (WMA) 2.0 API and cryptographic APIs. The payment software is transferred via Data cable to the client device with the use of Sony Ericsson PC suite.

The business logic and security logic of the payment application are implemented on all the entities including client mobile device. The client device contains the payment software, public key certificates and keys information. The institution server machine is deployed on an Intel ® Xeon ™ 3.00 GHz processor with 8 GB RAM, running a Linux Fedora Core 1 operating system. It keeps the information of registered clients for the payment services with necessary details.
The MFCS server is an IBM X 226 series Intel Xeon 3 GHz dual processor with 4 GB RAM, running the Linux Fedora Core 7 operating system. The MFCS server acts as a payment gateway and has the responsibility of authorizing the mobile client to perform the payment transactions and confirming the payments to the client.

The institution database server is configured with Intel Xeon 2.33 GHz dual Processor with 4GB dual RAM, running Windows 2003 operating system. Since the academic institutions support the students by providing various services including payments, they should deploy highly secure network infrastructure. In the experimental lab, the institution network is highly protected by integrated network security, where identity-based network security is used to protect the campus network from common intrusion or denial-of-service attacks and port-level identity security is used to restrict the institution network access to the authorized and authenticated users.

The Institution authorized bank server is configured with Intel Xeon 2.33 GHz dual processor with 8 GB dual RAM, running Linux Fedora 11 operating system. The CA server has 3.00 GHz Pentium IV processor with 2GB RAM and running Windows XP operating system. To protect the institution network from the intruders, the firewall is configured with Watchguard Firebox® X Core™ X750e and the switch is configured with Netgear FS728TP ProSafe 24-port 10/100 Smart Switch w/ 4 Gigabit.
5.3 Software Setup

The test bed is implemented based on n-tiered architecture, where the client-side functional components are implemented using Java 2 Micro Edition (J2ME). The J2ME is the preferred environment to deploy the payment software on Java-enabled mobile phones. The main advantages of J2ME over competing technologies are better portability and the ease for implementation. In addition, J2ME is platform independent technology. The functional components of the servers are developed using java servlets according to the standard used in Tomcat server 5.2 on Linux Fedora Core 9.0. The major responsibilities of the institution server are authenticating the payment users and sending their profiles to MPCS server. The MPCS server provides API specifications and interface description for authorizing the mobile clients, confirming the payments and preparing the payment request. The Certificate Authority server performs certificates validation functionality.

The client software was implemented using the Sun Java Wireless Toolkit 2.5.2 (formerly known as J2ME Wireless Toolkit) and it supports Mobile Information Device Profile version 2.0 specifications from Sun Developer Network (SDN) for secure transactions by enabling Hypertext Transfer Protocol over SSL (HTTPS) connections. The Wireless Toolkit (WTK) is the state-of-the-art tool box for developing mobile applications that provide emulation environment, performance optimization and tuning features, documentation and others. The functionality of the client software is developed as J2ME MIDlet application that is run on Java Virtual Machine (JVM), which offers Graphical User Interfaces (GUI), business
logic for client and ability to support secure communications with the IS and with the MPCS server.

The main reasons for providing client software as MIDlets are usability and wide acceptance of the users. The MPCSMIDlet is the main class for client application that provides interface and form to collect the payment requests after identifying and authenticating the client mobile device and the users. The payment request and authentication data are posted via HTTPS connection to the IS. The client mobile device acts as a payment terminal. The HTTPS must be implemented by one of the specifications such as HTTP over TLS, SSL V3, WTLS or TLS over Profile and Tunnelling Specification. The MPCS model uses SSL V3 specifications, because the HTTP over SSL is the most widespread secure technology for payment transactions, where Secure Socket Layer (SSL) provides confidentiality (end-to-end encryption), integrity, authentication using client and server certificates.

In addition to the main class of the client software, the security functional components are developed with respect to the standards of AESEngine and RSAEngine classes to enable encryption and decryption operations. The key class is responsible for performing the encryption and decryption using secret key generated by the mobile client and IS. The SHA1Digest class provides hashing operations for the payment transactions. The payment software uses lightweight RSA algorithm with variable key sizes of 1024 bits for encryption of the payment transaction and AES algorithm with variable key length 128 is used for encrypting the keys using secret key during encrypt/decrypt operations between the mobile device and the servers. When the client activates the MPCSMIDlet, the Mobile User Certificate
(MUC) is read from the local storage. This storage is the resource folder of the application’s Java ARchive (JAR) file.

The Java Cryptography Extensions (JCE) allows the easy use of cryptography libraries from different providers through a standard API. A concrete initiative called Bouncy Castle has released a lightweight API (http://www.bouncycastle.org) to handle cryptographic functions and public-key certificates for J2ME applications. The BC-API provides a set of security tools obtained from the original Java Cryptography Architecture (JCA) and the JCE that supports Connected Limited Device Configuration (CLDC) devices. The Bouncy Castle API (BC-API) is compatible with the CLDC/MIDP1.0 and does not depend on any manufacturer specific implementations. The Bouncy Castle package provides cipher engines for cryptology, non-reputable data using digest engines and supports key generation and exchanging the keys, generation of Message Authentication Code (MAC), reading and writing of X.509 certificates, etc.

The major functional components of the servers are payment service registration, distribution of AcCode and payment software, generation of secret key, validation of certificates, authentication, payment authorization, dispatch of the confirmation for payment request to the mobile client, creation of payment request, processing the payment transaction, response to payment request and they are allocated to different servers such as IS, MPCS, CA Server and bank server to implement the n-tier systems.

Oracle 9i is implemented in database server which is used to store and fetch the payment transaction records. For creating and validating the certificates based on
public key cryptography, a separate server is deployed where the OCSP Software application is deployed to check the evocation status of the server certificates. JMeter Tool (Sun Server Package) is used to measure the time measurements on mobile device. The tool is configured with the J2ME wireless toolkit and then connected with the mobile device via data cable for measuring the time of each operation in the mobile application. The software architecture and its development model are shown in Figure 5.2.

Figure 5.2: The Software Architecture and its Development Model
5.4 System Implementation and Testing

The proposed Mobile Payment System is developed using Java technology. The system architecture is designed based on layered architectural style. This allows effective encapsulation of the business logic and good maintainability. The layered paradigm also enables the isolation of the payment gateway server, i.e. the server running the business logic, from the Internet. This means that access to the MPSC server is only allowed from the institution server that provides the payment interface after authentication.

The MPSC system is organized in the form of large-scale, federated security architecture. The internal components of the architecture are various types of servers and (static or mobile) workstations. MPSC system's internal structure consists of four modules:

1. GUI module – handles creation of all user interfaces, related objects, such as forms, text editors, choice boxes etc.
2. Communication module – includes creating of communication interfaces for all supported protocols between client and server,
3. Business logic module – takes responsibility for creating transaction request messages and processing responses messages,
4. Security module – implements security elements to provide security features such as authentication, authorization, confidentiality, integrity and non-repudiation for the transactions and operations.

The internal structure and components of the Secure Mobile Payment System are presented in Figure 5.3.
5.4.1 Implementation of Client Software

The user interfaces for mobile applications are basically different from desktop applications. Designing the user interfaces for mobile devices presents a new set of challenges for interface designers and software developers due to its limitations. The client software in the client’s phone was implemented using J2ME. The main advantages of J2ME over competing technologies are better portability and easier implementation. J2ME applications, called MIDlets, are run in a Java virtual machine that offers GUI interfaces for implementing the client business logic and ability to support secure communications with the institution server and with the MPCS server. The client application was implemented using the Sun Wireless Toolkit (WTK) from Sun Micro-systems. MPCS provides user interfaces for the following transactions:
1. First Authentication of the user to the institution server based on Multi-factor Authentication scheme.

2. Second Authentication of the user to the Payment Gateway (MPCS server) based on IMEI of the client device.

3. Authorization for payment services using Fee Option and Amount.
   - Semester Fee
   - Examination Fee
   - Hostel Fee

4. Payment Confirmation Message.

5.4.1.1 Authentication of Payment Users by the IS

MPCS provides multi-factor of authentication protocols that verifies the payment user identities in local and remote levels. The first authentication is based on AcCode. When the application is started at the client's device, it prompts for AcCode. The AcCode is generated by the institution server and is then sent to the payment user via email after the successful completion of payment service registration. Once the payment user enters the value for AcCode, it is verified at the interface level and completes the initial user authentication. This authentication process is done locally and the display panel is shown in Figure 5.4.
At the time of user authentication, the institution server interface generates secret key using AcCode and International Mobile Equipment Identity (IMEI) number. While generating the secret key, IMEI of the client device is verified and decrypts the institution server’s public key from the server certificate available at client device. If the IMEI number is valid and the interface decrypts the server’s public key successfully, the payment user is initially authenticated. Meanwhile, the student roll number is also verified with IS database and displayed in login form. The Figure 5.5 represents the login panel of the MPCS system.
Figure 5.5: Remote User Authentication Panel

After completing the initial authentication, the institution server commences second level user authentication and the user interface prompts for UName and Pwd. When the user enters value for them, the UName and Pwd are digested using SHA-1 algorithm and the digested message is signed by the mobile user. Finally the signed message is encrypted using IS’s public key. If the institution server decrypts the UName and PWD successfully, the server is authenticated to the client and also the second level user authentication is completed. Now, the IS server compares the decrypted UName and PWD with IS database, and if both of them match, the client is finally authenticated. In case of failure, the interface allows the student to re-enter three times. When the user fails in all the three attempts, the user account is blocked.
and the institution server sends an “Access Denied” message to the user. The proposed model applies multi-factor authentication scheme to increase the assurance of the communicating parties using AcCode, IMEI number, UName and Pwd, and verifying students details with IS database.

5.4.1.2 Authentication of Payment Users by the Payment Gateway

The proposed model authenticates the mobile user by verifying the IMEI number of the client device. Since the mobile user is connected with the institution server, the payment gateway i.e. MPCS server applies simple mechanism. If the IMEI number is not matched, the payment authorization interface will not enable client device and MPCS server sends error message to the payment user.

5.4.1.3 Payment Authorization

After completing the user authentication procedures, the proposed model authorizes the user to avail the payment services using fee option and amount. The authorization interface for payment consists of fee options such as semester fee, examination fee, etc., and fee amount. The payment user can choose an option from a list. The option-id and fee amount are hashed using SHA-1 algorithm and the digital signature is generated for hashed option-id and the fee amount using client’s private key.

Meanwhile, the client interface encrypts a copy of the option-id and fee amount message using the secret key. Then, both the signed and the encrypted message are concatenated and which is encrypted using MPCS’s public key (M1). This avoids data theft and eavesdropping. In order to protect from identity theft, the MPCS’s public key is encrypted using part of an IMEI number (M2). Finally, the payment
application creates digital envelop for both the M1 and M2 messages and sends them to the MPCS server. The payment authorization interface is presented in Figure 5.6.

![Payment Service Authorization Panel](image)

Figure 5.6: Payment Service Authorization Panel

5.4.1.4 Payment Confirmation

After the payment service is authorized by the MPCS server, it creates and sends the payment request with the necessary information for the institution associated bank. The institution payment gateway and the institution associated bank make agreement in preorder. Once the PR reaches the institution associated bank, it starts the payment process by forwarding the payment request to the respective client’s bank. After authenticating the user and validating the account balance, the mobile
user's bank transfers the amount to the institution associated bank in the usual way. Once the requested fee amount is credited to the institution account, the institution associated bank sends payment confirmation message to the MPCS server.

![Figure 5.7: Payment Confirmation Message](image)

Finally, MPCS server sends the same confirmation message to the institution and then the institution server forwards the message to the payment user. Figure 5.7 shows the payment confirmation message.
5.4.2 Implementation of Payment Server functionalities

The server performs various operations such as payment user registration, handling the user’s information, managing the payment transactions and dealing with the security mechanisms. The above functionalities are split and assigned to both the institution server and MPCS server. The institution server performs the first two functions and the MPCS server handles payment procedures. Figure 5.8 represents the admin modules.

![MPCS-Administration](logged as: admin)

Customers ▶ Transactions ▶ Close

List Customers

Update Customer

Figure 5.8: Functions to Manage the Customers

5.4.2.1 Registration of Payment Users

The MPCS model provides administration module for the IS to register and manage the payment users. Each client is associated to MPCS payment system and must be registered to the institution server with necessary information. The proposed system
allows the students to register either through website or by directly. After completing the registration, the client may receive the payment software in a secure way. Figure 5.9 depicts the payment user registration form.

![Payment Registration Form](image)

**Figure 5.9: Payment Registration Form**

MPCS system generates data from the customers to facilitate the payment services and collects information such as customer data and bank details. Once the payment user completes his/her registration, the username/pwd and Accode are sent to the user’s personal mail. Mobile Phone Number must be entered with the full international exchange sign (+), country code and followed by the mobile number, without any blanks. Also, the users are required to enter their mobile phone models.
If the form contains any insufficient data, the application will not be accepted by the administrator.

5.4.2.2 Update Customers

Each registered client’s information is associated to institution server. If any information is not matched with institution database, the client registration is denied. The registered customers’ data can be updated using ‘Update Customers’ function by the administrator only after requesting him personally.

5.4.2.3 Payment Transactions Management

The proposed model handles payment transactions management functions with MPCS server that keeps the logs of all authorized payment user’s transactions and their messages. Figure 5.10 shows the functions to manage the payment transactions. The system administrator can use log id to view the user’s payment information. The MPCS system also supports the administrator to list the payment transactions information based on date and course-wise. This is useful for the institution to audit and trace students’ payment transactions. The list also includes the status of each payment transaction. Figure 5.11 represents the list of transactions completed successfully by the MPCS system. The message sent by the MPCS system during the payment process is also presented in Figure 5.12.
Figure 5.10: The Functions to Manage the Payment Transactions

Figure 5.11: List of Payment Transactions Completed

An Architecture for Secure Mobile Payment System using Public Key Infrastructure
The functional components of the proposed payment system are tested in the academic institution environment by establishing the Test Bed. The system was tested for both functionality and performance. The functional testing is focused on verifying the correctness of the system, whereas the performance testing is for verifying the adequate throughput of the system and to identify the possible bottlenecks and inefficiencies of the system. This kind of testing is helpful when the system is deployed at real time environment. The purpose of the tests is to understand the system’s performance, determine the scalability and the number of concurrent users or transactions. Statistical analysis is also performed on the test results to determine their goodness.
5.5 Performance Analysis on Device Authentication

This section discusses the mode used for measuring the time on performance of the encryption/decryption, message digest and signature computation done at mobile client device through GPRS network. The device authentication verifies the identity of client device and the institution server by using X.509 based public key certificates. Thus, the user interface is developed and the performance is studied for device authentication. The client device authentication has to invoke client certificate, send the client certificate to the institution server over GPRS communication, fetch the serial number of the client certificate, establish communication with CA server, obtain the CRL list and find the revocation status for client certificate. To evaluate the server certificate, the client device invokes the server certificate and fetches the serial number of the certificate, initiates OCSP connectivity, sends the OCSP request and verify the response of the OCSP from the CA server.

The proposed architecture is tested and measured in both emulator and mobile phone with respect to time consumption. The chief aim is to measure the data on emulator and later on to test all the measurements on the real mobile phone because various settings on the network throughput speed seem to be high. It is observed that the emulator that runs on personal computer consumes less time than the mobile phone, because the processor speed and memory of the personal computer are higher than mobile phone. The time taken for various steps involved in device authentication for both emulator and mobile phone is shown in Table 5.1 and presented graphically in Figure 5.13.
Table 5.1: Time Measurements for Device Authentication

<table>
<thead>
<tr>
<th>Steps</th>
<th>Description</th>
<th>Emulator (in ms)</th>
<th>Mobile Phone (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Invoke the Client certificate and URL</td>
<td>20</td>
<td>35</td>
</tr>
<tr>
<td>2</td>
<td>Client certificate verification &amp; Client device authentication</td>
<td>315</td>
<td>475</td>
</tr>
<tr>
<td>3</td>
<td>Invoke the server certificate</td>
<td>60</td>
<td>160</td>
</tr>
<tr>
<td>4</td>
<td>Fetching the serial number from the server certificate</td>
<td>45</td>
<td>140</td>
</tr>
<tr>
<td>5</td>
<td>OCSP Protocol Initiation</td>
<td>60</td>
<td>140</td>
</tr>
<tr>
<td>6</td>
<td>Server certificate validation &amp; Server Authentication</td>
<td>435</td>
<td>600</td>
</tr>
</tbody>
</table>

Figure 5.13: Time taken for Device Authentication
The proposed system uses Internet Engineering Task Force (IETF)'s X.509 standard digital certificates for entities authentication. Both the mobile client and server keep the public key certificates that have been issued by the CA. The X.509 based certificate contains a number of attributes namely serial number, certificate validity period, certificate holder's identity information, encryption keys that will be used for secure communications, and the signature of the issuing Certificate Authority.

When the mobile client initiates the payment transaction, the mobile device sends X.509 standard client certificate and URL to the IS over the GPRS network for validation. Then the institution server sends the server certificate to the client device for verifying whether the certificate is revoked or not. The revocation is an act of making a valid certificate invalid before its expiration date. At the time of certificate validation, the protocol is used to check the validity of IS certificate by forwarding OCSP requests through the client device to the Certificate Authority server. The mobile client sends information including the certificate issuer name and the serial number of the server certificate to the OCSP responder of the Certificate Authority. The CA server examines its copy or copies of the CRL to determine if the Certification Authority has listed the certificate as being revoked and replies with a message to the client device that the certificate's status is "revoked", "good", or "unknown". Since the mobile device is having limited capabilities, the OCSP is the preferred and standard medium for validating the server certificates and also it consumes only less bandwidth. It also consumes no memory on the mobile device, as it will not involve the use of CRLs.
During the validation of client’s certificate revocation, the institution server checks the validity period and public key of the client certificate. In the public key infrastructure, Certificate Revocation Lists (CRLs) handle revocations, where the CRLs are issued and digitally signed by the CA. The revoked certificates are represented in the CRL by their serial numbers. When the institution server attempts to verify the validity of the client’s certificate, it will download the fresh CRL from CRL Distribution Point (CDP) and scan the current CRL for the serial number of the presented certificate. The time taken amounted 300 ms for retrieving the server certificate from the IS using GPRS communication and fetching the serial number from the certificate. An average of 740 ms was required to initiate the OCSP communication and send the OCSP request and to receive and verify the OCSP response for the server certificate. Thus, to evaluate the identity of the institution server in the context authentication at client device requires retrieval of server certificate, establishment of OCSP connection, sending and receiving the OCSP request/response. This is acceptable by MIDP 2.0 enabled devices.

5.6 Performance Analysis of Client and Client/Server Authentication

This section investigates the performance of the various functional components of the client mobile device and the institution server using RSA algorithm. After completing the device authentication, the proposed security architecture initiates the client and client/server authentication. The authentication for client and client/server process is explained in chapter 3. The client authentication process includes secret key generation using AcCode and IMEI number, obtaining the public key from server certificate and decryption of the public key using secret key. The
client/server authentication process includes obtaining the UName and PWD from the mobile user, creating a digest for authentication parameters, encrypting the digest using secret key, signing the final message using client’s private key and sending it to the institution server. These functionalities are done at client device.

After receiving them, the institution server follows the reverse process and maps the username and password with IS database records. If the server gets positive results, then the client receives the message for authentication. If the institution server decrypts username and passwords successfully, then the server is authenticated to the payment user. To make the comparative study on client authentication functionality, experiments have been conducted on three different algorithms. The main scenario of this experimental study is to measure the time for the performance of RSA algorithm on client authentication.

Table 5.2: Time Measurements for Client Authentication

<table>
<thead>
<tr>
<th>Steps</th>
<th>Operations</th>
<th>PC (in ms)</th>
<th>Mobile Phone (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Secret Key Generation</td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>Obtain the public key from the Server’s Certificate</td>
<td>30</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Decrypt the public key using secret key</td>
<td>25</td>
<td>45</td>
</tr>
</tbody>
</table>
Figure 5.14: Time Spent for Client Authentication

The time spent for client authentication process at mobile device is presented in Table 5.2. The graphical representation of time spent for client Authentication is shown in Figure 5.14.

Table 5.3: Time Measurements for Client/Server Authentication

<table>
<thead>
<tr>
<th>Steps</th>
<th>Operations</th>
<th>Emulator (in ms)</th>
<th>Mobile Phone (in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hashing the UName &amp; PWD</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>Encrypt hashed UName &amp; PWD using secret key</td>
<td>27</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>Sign the encrypted hashed message</td>
<td>38</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>Encrypt UName &amp; PWD using IS’s public key</td>
<td>40</td>
<td>75</td>
</tr>
<tr>
<td>5</td>
<td>Client authentication Reply</td>
<td>51</td>
<td>70</td>
</tr>
</tbody>
</table>
Figure 5.15: Time Taken for Client/Server Authentication

The time spent for client authentication process done at mobile device is presented in Table 5.3. The graphical representation of time spent for client Authentication is shown in Figure 5.15. The chart representation shows that the client mobile device consumes more time for encrypting the digest of UName and PWD using secret key than the computer. Also, the mobile phone consumes significant time for creating the signature using client’s private key. The significant note here is that the time taken for encrypting the digest of the UName and PWD using secret key is less than from encrypting the UName and PWD using IS’s public key at mobile device. This shows that the disadvantage in response time due to the low performance of encryption process using public key algorithm compared to symmetric key algorithm.
The measurements are done in an emulator using Wireless toolkit 2.5.2 from Sun Micro-systems and on Sony Ericson K750i java enabled device. With an emulator the client is running on a Personal Computer (PC) over Local Area Network (LAN). From the implementation, it is found that RSA has low performance and low response time in mobile device when compared to personal computer, although, it achieves high level security when compared to symmetric algorithms. Finally, if any payment system’s implementation relies on more security mechanisms with complex public key encryption algorithm like RSA it then becomes a disadvantage with a notable impact.

5.7 Computational Cost

The proposed security protocol has been analyzed in terms of the computational cost. The computational cost expresses the performance of the mobile devices to complete the mobile payment protocol. Table 5.4 presents the cryptographic operations carried out by the proposed system during the authentication, authorization and payment transaction. The computation cost is calculated using the following formula [Peláez et al. 2008].

\[
\text{Cost} = (\text{OPENC} \times \text{TENC}) + (\text{OPDEC} \times \text{TDEC}) + (\text{OPSIG} \times \text{TSIG}) + (\text{OPVER} \times \text{TVER}) + (\text{OPESYM} \times \text{TESYM}) + (\text{OPDSYM} \times \text{TDSYM}) + (\text{OPHASH} \times \text{THASH}) \quad \ldots \quad 1
\]

where,

\text{OPENC} and \text{OPDEC} refers operation for public key encryption and decryption,
\text{OPSIG} and \text{OPVER} refers digital signature creation and verification,
\text{OPESYM} and \text{OPDSYM} refers symmetric key encryption and decryption,
\text{OPHASH} refers hash operation and \text{T} denotes time taken for operation.
In a public key cryptosystem, the level of security is achieved based on the key lengths, and not relying on type of algorithms used. There is no significant difference also between encryption and decryption when the key lengths are changed. However, while calculating the computational cost for the payment system, the time consumption for each cryptographic operation should be considered separately. Adopting such measures is helpful when planning the deployment of proposed system to commercial use. Thus, the cost for each operation is calculated individually and then all the costs are added up to obtain the total computational cost.

Table 5.4. Number of Cryptographic Operations Performed on MPCS Protocol

<table>
<thead>
<tr>
<th>Entity</th>
<th>OPENC</th>
<th>OPDEC</th>
<th>OPSIG</th>
<th>OPVER</th>
<th>OPSENC</th>
<th>OPSDEC</th>
<th>OPHASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>IS</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>MPCS</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

The cost for each cryptographic operation is, \( C(op) = \frac{n_i}{1000} \cdot c_i \) Where, \( n_i \) is the \( i^{th} \) cryptographic operation \( t_i \) is, the time (ms) taken for \( i^{th} \) cryptographic operation \( c_i \) is the actual cost for \( i^{th} \) cryptographic operation Then, the total Total Computational cost (TC) for the proposed system is,

\[
TC = \sum_{i=1}^{n} \left( \frac{n_i}{1000} \cdot c_i \right)
\]
For example, $C(\text{OPHASH})$ at mobile device for client authentication is,

$$C(\text{OPHASH}) = (1 \times 30)/1000 \times (1.50) = \text{Rs}.0.09$$

where $C_i$ is fixed by the security service provider and it may vary based on the providers. [Peláez et al. 2008] have presented the protocol for micro-payments. The protocol performs cryptographic operations at customer, Point-of-Sale (POS), Automatic Teller Machine (ATM) and payment gateway to complete the mobile payment transactions using PKI.

### Table 5.5. Peláez et al. Security Protocol

<table>
<thead>
<tr>
<th>Entity</th>
<th>OPENC</th>
<th>OPDEC</th>
<th>OPSENc</th>
<th>OPSDEC</th>
<th>OPHASH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>POS</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>ATM</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Bank</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>4</td>
<td>5</td>
<td>18</td>
<td>18</td>
<td>9</td>
</tr>
</tbody>
</table>

The result of comparison on cryptographic operations based on Tables 5.4 and 5.5 revealed that the proposed mobile payment is less than the number of cryptographic operations applied to the Peláez et al. protocol. The Peláez et al. security protocol is required to perform asymmetric key operation and symmetric key operation at high level compared to the MPCS protocol. As discussed earlier, RSA algorithm shows low performance and low response time at client mobile devices. Additionally, it is found that the existing protocol uses symmetric key operations with more numbers. The payment models supported by symmetric key operations have already proved
found that the existing protocol uses symmetric key operations with more numbers. The payment models supported by symmetric key operations have already proved that they are insecure and unsuitable for wireless communication. However, the proposed mobile payment protocol adopts secret key operations at minimum level, in particular for encrypting the public keys.

The significant note is that the existing protocol does not adopt the digital signature mechanism. Digital signature ensures the authenticity of transaction parties and non-repudiation. Thus, it is observed that the existing system does not support the security services completely. Also, the MPCS protocol uses hash functions with less numbers. The proposed protocol also reduces the total computation cost for all engaging parties and ensures that there is no compromise on supporting the confidentiality. Finally, the protocol designed for MPCS system has improved the performance by reducing the number of cryptographic operations compared to the existing protocol. The comparative cost analysis is shown in Figure 5.16.

![Computational Cost Diagram](image_url)

*An Architecture for Secure Mobile Payment System using Public Key Infrastructure*
5.8 Performance Results Analysis

This section investigates the performance in terms of response time and system throughput. The ultimate aim is to have a measure of the throughput of the system with multiple payment users. These measures are helpful during the deployment of the system as commercial product. The performance tests are automated. A proper and adjustable load of concurrent transactions are monitored.

In the proposed system, the server load would be distributed across four servers namely IS, MPCS server, Bank Server and CA server. The purpose of the tests is to form an understanding of the system’s performance and determining the scalability, where the number of concurrent users or transactions was set as a variable.

<table>
<thead>
<tr>
<th>Test</th>
<th>No. of Payment Users</th>
<th>No. of Concurrent Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>6</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>9</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>10</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>11</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>12</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td><strong>Total</strong></td>
<td><strong>556</strong></td>
</tr>
</tbody>
</table>
Tests are conducted with 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, and 100 successive payment users generating the payment requests to the proposed system simultaneously. The maximum number of concurrent users of 100 was chosen to represent a test-case scenario. The response time of the system is measured and reported separately for each request. Statistical analysis is performed on the test results to determine their goodness of the system response time. The Table 5.6 represents the number of requests chosen for the test. A summary of the test results is shown in Table 5.7.

### Table 5.7: A Summary of Test Results

<table>
<thead>
<tr>
<th>Test</th>
<th>Mean Response Time (Sec.)</th>
<th>Standard Deviation (Sec.)</th>
<th>Throughput No. of Requests/Sec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.04</td>
<td>0.00</td>
<td>0.24</td>
</tr>
<tr>
<td>2</td>
<td>4.06</td>
<td>0.01</td>
<td>1.23</td>
</tr>
<tr>
<td>3</td>
<td>4.12</td>
<td>0.07</td>
<td>2.42</td>
</tr>
<tr>
<td>4</td>
<td>4.28</td>
<td>0.18</td>
<td>4.67</td>
</tr>
<tr>
<td>5</td>
<td>4.45</td>
<td>0.35</td>
<td>6.74</td>
</tr>
<tr>
<td>6</td>
<td>4.65</td>
<td>0.47</td>
<td>8.60</td>
</tr>
<tr>
<td>7</td>
<td>4.85</td>
<td>0.59</td>
<td>10.30</td>
</tr>
<tr>
<td>8</td>
<td>5.08</td>
<td>0.74</td>
<td>11.81</td>
</tr>
<tr>
<td>9</td>
<td>5.35</td>
<td>0.96</td>
<td>13.08</td>
</tr>
<tr>
<td>10</td>
<td>5.64</td>
<td>1.17</td>
<td>14.18</td>
</tr>
<tr>
<td>11</td>
<td>5.97</td>
<td>1.45</td>
<td>15.07</td>
</tr>
<tr>
<td>12</td>
<td>6.78</td>
<td>2.86</td>
<td>14.74</td>
</tr>
</tbody>
</table>

During the performance tests, the operating system performance monitor was observed. The general observation was that the load on server system was low with less number of payment users and high with more number of users. However, with
one concurrent client, the load is stayed steady at about 0.01% and with 90 clients a constant load was generated significantly.

**Scalability**

Scalability is the most important factor in the payment system where the number of users and the frequency of the usage cannot be determined and fixed in advance. In order to evaluate the scalability of the system, the underlying two factors are taken:

1. The behavior of the mean response time of the system is based on the payment functions performed by the number of simultaneous clients.
2. The behavior of the throughput of the system relies on number requests per second towards the number of simultaneous clients.

Based on the test results presented in Table 5.6 and 5.7, the graphs are presented for mean response time and system throughput in Figure 5.17 and Figure 5.18.

![Response Time Vs. Load](image)

**Figure 5.17: The Performance Results For Mean Response Time**

*An Architecture for Secure Mobile Payment System using Public Key Infrastructure*
The Figure 5.17 represents the performance results in terms of mean response times. On the significant scale of 1 to 90 payment users, the payment system seems to exhibit a linearly growing mean response time. The slope of the curve is starting to steepen between 90 and 100 simultaneous users. This is quite natural that every system reaches the level of load where the performance observed by a single user starts to degrade dramatically.

![Throughput of the Mobile Payment System](image)

**Figure 5.18: The performance test results - Throughput**

The Figure 5.18 depicts the throughput of the proposed system. The system throughput exhibits a very scalable behavior i.e. the throughput increases gradually up to 10 payment users and keeps rapidly increasing till 90. At 90 simultaneous users, the system has reached the saturation point due to various factors and the throughput declines. However, the proposed system provides responses to the user requests with reasonable response time and the system load is linear.
The performance of the system is experienced by each individual user. It will be an important factor for the overall usability of the system. Based on the test results used for determining the scalability, the performance of the system can be analyzed. The factors observed for determining the performance of the system is an individual response time and the maximum throughput of the system.

The design criteria of an average response time of 4 to 6 seconds can easily be achieved. This is the response time that a mobile user can expect to experience at the best. However, the network latency will further increase the response time but this has to be an acceptable one. The minimum individual response time could be achieved within the specified hardware and networking architecture and it was about 4.04 seconds. The Standard Deviation (SD) is used to measure the variability or dispersion of the data set. If the data are very close to the average mean, then average mean is deviated from the standard deviation at small level. If the data are dispersed over a wide range of values, average mean is deviated at level. It is also used to measure the risk of security. The smaller the SD, the tighter the probability distribution, and the lower the risk associated with the security. The data set for standard deviation is presented in Table 5.7.

The maximum throughput of the system achieved in the tests is 15.07 requests per second at the mean response time of 5.97 seconds. From the Table 5.7, it is found that the throughput was naturally achieved when there are constantly 90 simultaneous users. This level of throughput can be considered as adequate in terms of the criteria specified for the system.
This chapter has outlined the experimental setup to measure the time taken for the encryption in RSA algorithm by General Packet Radio Service (GPRS) network that deals with authentication at three levels. Test bed is established with the aim of testing the performance of the system at various security levels. The hardware and software setup with their entities, servers, security tools and software illustrate aspects related to confidentiality, integrity, encryption and decryption operations. The system implementation and testing discuss how MPCS system is organized into the proposed model and how implementation of client software functions. Implementation of payment server functionalities is clearly demonstrated. Performance analysis on device authentication, client and client/server authentication bring out aspects of encryption/decryption, message digest and signature computation through GPRS network. The proposed security protocol analyses the computation cost. There is a thorough evaluation of the proposed system in terms of response time and system throughput.

Thus the proposed model supports security at high level up to the standard level as envisaged by industries. The academic institutions can very well adopt the proposed security framework for their students to carry out secure payment transactions with the strong authentication. The institutions could also accomplish their business requirements managing security administration and operations for financial applications by using the MPCS system.