Uday Pratap Singh, Mukul Pathak, Riidhei Malhotra and Madhu Chauhan/ International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622
www.ijera.com Vol. 2, Issue 2, Mar-Apr 2012, pp.838-848

Secure Communication Protocol for ATM Using TLS Handshake

Uday Pratap Singh[1], Mukul Pathak [2], Riidhei Malhotra[3] and Madhu Chauhan[4]

[1, 2,4] Department of Computer Science & Engineering, Galgotias College of Engineering & Technology, Greater Noida (U.P.), India

[3] Department of Information Technology & Engineering, Galgotias College of Engineering & Technology, Greater Noida (U.P.), India

ABSTRACT

Financial service outlets such as ATMs are vulnerable to various attacks like shoulder surfing, data skimming, fake machines etc. Therefore these outlets are easy targets for attackers. In the current financial service transactions, there is no provision for users to verify genuineness of the financial service outlets like ATMs. Therefore users are forced to trust the ATMs. Fake ATM can be designed to steal the account information and private information like password from the user. This information later can be used to steal money from the user. Further, the financial outlets require a dedicated communication channel for user authentication. Installing these outlets is therefore an expensive operation. In this work, we are providing Secure two way TLS Handshake Communication Protocol for financial services based on a smart card. By authenticating ATM, the user is ensured of the genuineness of the ATM. Hence fake ATM cannot steal the user information. After the session establishment, communication is carried out in encrypted form. This way the attacker cannot get any meaningful transaction information. Thus our approach handles security vulnerabilities in the current model. Transaction cost in our approach is significantly reduced, compared to the current model because the continuous network is not required due to localized authentication process.

KEYWORDS- Secure ATM; TLS Handshake; Communication Protocol; Network Security.

1. INTRODUCTION

An **automated teller machine** or **automatic teller machine** (**ATM**), also known as an **automated banking machine** (**ABM**) in Canada, and a **Cashpoint** (which is a trademark of Lloyds TSB), **cash machine** or sometimes a **hole in the wall** in British English, is a computerized telecommunications device that provides the clients of a financial institution with access to financial transactions in a public space without the need for a cashier, human clerk or bank teller. ATMs are known by various other names including *ATM machine*, *automated banking machine*, and various regional variants derived from trademarks on ATM systems held by particular banks[10].There is gradual increment in uses of Electronic Commerce to a level, where it has captured the role of processing transactions from physical bank branches. The convenience offered by e-commerce has resulted in it becoming necessary part of our daily life so commercial organizations like banks, credit card companies, insurance companies etc. have shown great interest towards providing new technologies to their users. These technologies are provided in form of variety of services like ATM, credit cards, electronic fund transfer etc. These organizations vie for greater share in user market by introducing newer and better services. But current services have some security flaws. Fraudsters try to exploit these flaws. It is therefore desirable to cover these flaws and ensure that the security is not compromised. Our approach is a step towards meeting this goal.

Financial service outlets like ATMs are now very important part of one's daily life. Any time a user wants to withdraw money, instead of going to the bank he goes to an ATM. While shopping, user can pay bills at Point of Sale (POS) using ATM cards instead of cash payment. Performing these transactions is very swift and less time consuming as compared to visiting a bank. But there are some attacks on financial services as listed below.

1.1 Fake Automated Teller Machine

In this attack, the attacker installs a fake ATM which is similar to the genuine machines and when user uses the fake machine it can record all user account information as well as user private information (like PIN).

1.2. Shoulder Surfing

In this attack, an attacker can look directly, or can use video cameras, to capture PIN entered by an user.

.

1.3 Skimming Devices

An invader can record data from magnetic strip of ATM cards by using skimming devices. These devices can be clamped together with card reader of ATM and can record data of multiple (~200) ATM cards. These devices are often used by attackers, at point of sale (POS) in shops or restaurants, where cards are handled by staff.

1.4 Fake Keypad Overlay Attack

In this kind of attack, the attacker places fake keypad overlay over a genuine keypad. The fake keypad overlay is transparent and cannot be detected easily by naked eye. The fake overlay then stores pressed keypad buttons with time. This information can be used to compromise PIN.

2. Why Smartcard?

A **smart card**, **chip card**, or **integrated circuit card** (**ICC**), is any pocket-sized card with embedded integrated circuits. Smart cards can provide identification, authentication, data storage and application processing. Smart cards improve the convenience and security of transactions. They provide tamper-proof storage of user and account identity. Smart card systems have proven to be more reliable than other machine-readable cards, like magnetic strip. They protect against a full range of security threats, from careless storage of user pass-words to sophisticated system hacks. Smart cards are chosen for applications with security concerns, due to their small size, data storage capability and processing abilities.

Smart cards which we are using an operating system (OS). This OS supports security mechanisms like encryption, decryption, authentication, session key establishment etc. based on both symmetric and asymmetric key algorithms. It also supports various file and data access mechanisms.

Since a smart card provides for mutual authentication mechanism, in our financial service system, the users do not have to trust the financial outlets. The smart card can check the authenticity of outlets. Smart cards also allow various measures to provide access and security control for application data. Hence transaction information can also be stored on smart card without losing the aspects of security. In our approach, financial outlets do not have to contact bank database server for each transaction performed on ATM. This reduces dedicated network cost, which in turn reduces the transaction cost. Smart cards also provide session key establishment facility to provide data confidentiality. Data transferred between smart card and financial outlet can be encrypted by session key. Even where an attacker captures this data, he cannot use it for forgery. Figure 1 illustrates structure and packaging of smart card.[15]

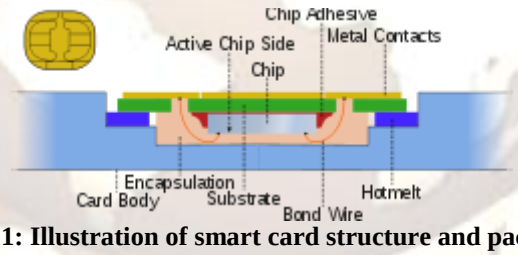

Figure 1: Illustration of smart card structure and packaging

3. Related Work

Han et al.[10] came up with hybrid concept of Biometrics based authentication scheme for ATM like banking applications. They proposed two level authentication process, first level based on PKI and second level authentication based on biometrics like fingerprint or iris matching. In this model, authentic biometric reference data is stored on a database server of the bank. Therefore, this model requires a dedicated network and secure channel between ATM and database server to retrieve reference biometric information and to match biometric data. Further, biometric devices are expensive to deploy, the size of biometric data exchanged between the ATM and server is large causing the operations such as transfer of data, encryption, processing etc. to be resource expensive, thereby increasing the cost of transaction. Jhunjhunwala [4] proposed a design of low-cost ATM for deployment in rural areas. Further, their ATM, known as Gram teller does not need personal information and magnetic stripe cards, but uses smart cards and biometrics based on fingerprints. Gram teller is connected to a PC with internet access. It communicates with bank database server through this PC. Installation cost in this model is very low. However, it suffers from security vulnerabilities and unreliable network connection. Around the world, financial corporation's (like Visa, MasterCard, Discover, and American Express) have rolled out smart cards under the EMV (Europe, MasterCard, VISA) standard [9]. The majority of implementations of

EMV cards and terminals confirm the identity of the cardholder by requiring the entry of a PIN.EMV standards implement only some of the commands defined in ISO/IEC 7816-4 standard.

Moona et. al. [3] proposed a solution which uses mobile devices (like mobile phones or laptops) of user to communicate with ATM machine. This model does not require I/O console or keypad. It also eliminates the need of dedicated network as transaction records can be stored on mobile devices. Since keypads or I/O console are not used, their approach eliminates the possibility of fake keypad overlay attack, or use of skimming devices. This solution reduces infrastructure and transaction cost significantly.

Munjal et. al. [8, 1] proposed a new financial service model to overcome various drawbacks in the current model. Smart cards are used to provide secure storage for data, which protect data from unauthorized writing and reading. In this model, new secure protocol for communication between ATM and smart card is proposed based on PKI. This model removes need of dedicated network and reduces transaction cost. The paper work we have done is mainly based on the work of Munjal et. al.

The concept of public key cryptography was proposed by Whitfield Diffie and Martin Hellman [7]. They established the necessity of asymmetric key cryptography for sharing a secret key over an insecure communication channel in their Diffie-Hellman key exchange protocol. Several public key cryptographic algorithms like RSA, elliptic curve cryptography(ECC) were developed subsequently to meet the requirements of public key cryptosystems. The concept of public-key-infrastructure (PKI) was introduced to address the security drawbacks present in the key management process of a public key cryptosystem by means of digital certificates to assure the genuineness of a user's public key. A smart card can be used by a user to perform PKI related tasks and used to keep the private key secret.

4. Layered Architecture

As described earlier, there are various shortcomings in current financial service model. In our paper, we know the Deployment Scenario [13] for financial services that will overcome these shortcomings. We use a smart card in our system to provide security framework. Application uses computation and storage capabilities of a smartcard for secure storage and authentication. The organization of our layered architecture is as shown in figure 2.

Figure 2: Layered Architecture

- **Smart Card Layer:** The smart card layer comprises of a smart card that communicates with the application with the help of a device known as 'reader'. Communication between a smart card and the reader is half-duplex, that is only one side can transmit data at a time. Communication is always initiated by the reader which sends commands to the card. The card then responds with the output for the command. Smart card uses an operating system known as SCOSTA-PKI to provide standard mechanisms of command and data exchange. SCOSTA-PKI is an operating system compliant to ISO standards
- **Smart Card Reader Layer:** Smart card readers are used as communication medium between the smart card and a host (like a computer, a point of sale terminal) or a mobile telephone. Smart card readers are available in two categories: contact smart card readers, contactless smart card readers based on 'radio frequency'.
- **Command Library Layer:** Command library is a standard C++ library developed for handling smart card interactions, according to SCOSTA specifications (SCOSTA-CL, SCOSTA-PKI).
- **Interface Layer:** Interface layer is the uppermost layer in the layered architecture of our application. It directly communicates with the user of the application. This layer is the software application that implements secure ATM protocol.

5. Proposed Work

5.1 Secure Communication Protocol

We are going to provide secure communication protocol for various deployment scenarios using smart card given on **An Efficient Architecture for Trusted Financial Services in ATM [14]**. There are various protocols to establish mutual authentication and to derive session key between two communicating parties. Some of examples for these protocols are Diffie-Hellman Key Exchange protocol [7], RSA (Public key) based session key establishment protocol, IPSEC protocol, Kerberos, TLS protocol [6] etc.

5.1.1 TLS Protocol

The primary aim of the TLS protocol [6] is to provide data confidentiality and data integrity between two communicating applications. The protocol is composed of two layers- the TLS record protocol and the TLS handshake protocol.

The TLS handshake protocol is used for authentication of client and server and also to negotiate encryption algorithm and cryptographic keys. This TLS handshake protocol provides connection security which has three basic properties.

- The peer's identity can be established using public key cryptography based authentication.
- The negotiation of a shared secret is secure.
- The negotiation is robust.

The TLS handshake protocol is responsible for negotiating the following security parameters for a session.

Session Identifier: An arbitrary byte sequence chosen by the server to identify an active or resumable session state.

Peer certificate: X509v3 certificate of the peer.

Compression method: The algorithm used to compress data prior to encryption.

Cipher spec: Specifies the pseudo-random function (PRF) used to generate keying material, the bulk data encryption algorithm (such as null, AES, etc.) and the MAC algorithm (such as HMAC-SHA1). It also defines cryptographic attributes such as the Mac length.

Master Secret: 48-byte secret shared between the client and server. This secret is used to derive session key for confidentiality and integrity.

These security parameters once established, are used by the TLS record protocol for protecting application data.

5.2 Message Flow in TLS Handshake Protocol

Message flow signifies how various cryptographic parameters (known as security parameters) are established during TLS handshake protocol [6], for a session between two communicating parties (Client and Server). These security parameters are used subsequently to provide secure and reliable communication between the client and the server.

Figure 3: Message Flow in TLS Handshake Protocol

List of various cryptographic algorithms (known as CipherSuiteList) like key exchange algorithms, bulk encryption algorithms etc. supported by the client are passed to the server through ClientHello message. In response, the server selects one set of cryptographic algorithms (CipherSuite) from the list through ServerHello message. The server provides its certificate to the client through Certificatemessage, which is used by the client to extract the public key of the server. The server may also demand a certificate from the client through CertificateReque message. Certificate provided by the client is used by the server to extract public key of the client.ServerkeyExchange and ClientKeyExchange messages are used to establish secret key known as PreMasterSecret, which is then used to derive MasterSecret. These messages carry key-material to establish PreMasterSecret. ChangeCipherSpec message signals that new negotiated security parameters will be used for all subsequent communication. Finished messages indicate all negotiations are complete and secure communication is established. Complete list of messages sent during TLS handshake protocol is provided in figure 3.

5.2.1 ClientHello Message

When a client first connects to a server, it sends the ClientHello as its initial message.This message carries following fields.

Client Random: This random number consists of 4 byte timestamp followed by 28 random bytes and is used in generation of MasterSecret (section 5.2.7).

Session id: This field identifies session between client and server. This is variable length field, subject to the

maximum size being 32 bytes.

CipherSuite: It contains the list of cryptographic algorithms for key exchange, bulk encryption etc. supported by the client. Key exchange algorithms supported by TLS handshake protocol are listed below.

In dh_ anon key exchange, anonymous Diffie-Hellman key exchange is used. Dhe_rsa, dhe_dss key exchange uses server certificate keys for signing purpose. rsa uses server certificate key for encryption purpose. In DH RSA, dh dss, the certificate contains Diffie-Hellman public key.

TLS handshake message formats explained in the subsequent sections assume that, key exchange method used is RSA.

5.2.2 ServerHello Message

The server sends this message in response to a ClientHello message when it decides an acceptable CipherSuite. If it could not find such a match, it will respond back with a handshake failure alert. Fields in this message are listed below.

Session id: Session id identifies the session between the server and the client. Value of this Field is always set by the server.

5.2.3 Server Authentication and Key Exchange Messages

5.2.3.1 Server-Certificate Message

This message conveys the server's certificate chain to the client. Public key of the server is extracted from this certificate chain.

5.2.3.2 Server Key Exchange Message

- This message will be sent immediately after the Server-Certificate message. This message conveys cryptographic information, that allows the client to communicate the PreMasterSecret.
- The ServerKeyExchange message is sent by the server only when dhe_dss, dhe_rsa and dh_anon key exchange algorithms are used. If rsa, dh_rsa and dh_dss key exchange algorithms are used, this message is not sent.
-

5.2.3.3 CertificateRequest Message: The server can optionally request a certificate from the client, if it is needed for the selected cipher suite.

5.2.3.4 ServerHelloDone Message: The server sends the ServerHelloDone message to indicate the end of the ServerHello and ServerKeyExchange messages.

5.2.4 Client Authentication and Key Exchange Messages

5.2.4.1 Client Key Exchange Message

Client-Certificate message is sent only, if the server requests the client certificate. ClientKeyExchange message immediately follows the Client-Certificate message, if it is sent. This message sets the PreMasterSecret, according to applied key exchange algorithm. PreMasterSecret is used to derive MasterSecret.

If RSA is used as the key exchange algorithm, the client generates a 48-byte random number, which is known as PreMasterSecret. The client then encrypts PreMasterSecret using the public key from the server's certificate, and sends the result in ClientKeyExchange message.

5.2.4.2 CertificateVerify Message:

This message is used to provide explicit verification of a client certificate and is sent only if client certificate has signing capability. This message includes signature on hash of all handshake messages exchanged before this message.

If the server is able to verify signature on handshake messages, then client public key is genuine and the client is authenticated.

5.2.5 ChangeCipherSpec Message

The ChangeCipherSpec message is sent by both the client and the server to notify the receiving party that subsequent communication will be protected under the newly negotiated CipherSpec and keys.

5.2.6 Finished Message

Finished message is sent to verify that the key exchange and authentication processes were successful and is the first message protected with just negotiated cryptographic algorithms and session keys. It contains the verification data, calculated as

Verify Data = PRF(MasterSecret, finished label, HASH (handshake msgs))[0...12]

Handshake messages include all messages before finished message except Change- CipherSpec message.

5.2.7 MasterSecret

MasterSecret is derived from PreMasterSecret as follows.
MasterSecret= PRF(PreMasterSecret, "master secret", ClientHello.random + ServerHello.random) [0...47]
It is always 48 bytes in length. Once MasterSecret is calculated PreMasterSecret is deleted from memory. The MasterSecret is expanded into a sequence of secure bytes known as Key Block, which is then split into different session keys.

5.3 RSA algorithm based Mutual Authentication and Key Establishment Protocol

This protocol involves a sequence of stages. Before completing one stage, protocol execution cannot proceed to the next stage. The stages in protocol in order of appearance are as listed below.

- Handshake Stage
- Certificate Exchange Stage
- Mutual Authentication and Key Establishment Stage

5.3.1 Handshake Stage

The goal of this stage is to negotiate the Protocol Version and Algorithm Suite. The handshake is carried out using Hello and Welcome messages.

5.3.1.1 Hello Message : This is the first message sent by a client to the server after establishment of the connection. This message conveys ProtocolVersionList and AlgorithmSuite to the server. ProtocolVersionList provides version of protocol supported by the client and AlgorithmSuite provides the list of algorithms supported by the client for calculating hash, encryption and MAC.
Server sends Welcome message to the client as a reply.

5.3.1.2 Welcome Message: The server sends a Welcome message as the response to the Hello message received from the client. This message carries the protocol version and algorithm suite accepted by the server. The server preferably selects a protocol with the highest version from the list provided by the client. This message includes the following two fields.
SelectedProtocolVersion: This field consists of the protocol version that the server has selected.
SelectedAlgorithmSuite: This field consist references for hash, encryption and MAC algorithms chosen by the server.

If the client discovers that SelectedProtocolVersion or SelectedAlgorithmSuite are not in the list provided in the Hello message, then the client sends back an error message. Otherwise, the client starts with the certificate exchange stage.

5.3.2 Certificate Exchange Stage

Certificate exchange stage achieves two purposes- to exchange the public keys among the client and the server and to verify genuineness of the public keys.

5.3.2.1 Certificate Message

This message is sent by both, client and server. The receiver gets the list of certificates in the certificate chain starting from root certification authority (CA) from the sender. The receiver can then verify the sender's public key using its certificate chain.
In certificate chain, all certificates are DER encoded. Each certificate must be X.509v3 type.
Certificates must have RSA public key. The last certificate in the certificate chain must be the sender's certificate. If the server verifies certificate of the client, then it responds with own Certificate message to the client. If the client verifies this server's certificate, then it proceeds to mutual authentication stage.

5.3.3 Mutual Authentication and Key Establishment Stage

To ensure that the other side has the private key corresponding to the public key in its certificate, authentication is required. It is done through series of challenge- response messages. Session key for encryption (confidentiality) and MAC (Integrity) are derived in this stage only.

5.3.3.1 Challenge Message

This message contains the challenge issued by the client to authenticate the server. The server must respond with a Response message, which enables the client to authenticate the server.
The challenge is a random 8-byte number, say R1.

5.3.3.2 Response Message

This message is sent as response to the Challenge message. ASN.1 representation for Response message is provided below.
The server calculates Response message in the following manner.
1. The server generates a key material K2 and an 8-byte wide random number R2.
2. Then it concatenates the random number generated, the random number received, and the key material as N1 = R2||R1||K2.
3. The server calculates the cryptogram as
C1 = RSAES-OAEP-ENCRYPT(Client Publickey, N1, NULL)
4. The server then sends C1 as part of the Response message. Random number R2 is an 8-byte long number and key material K2 is 16-byte long number. The client processes the Responsemessage from the server and then generates its own Response message to the server. The steps in this operation are listed below.

- **i.** The client decrypts the cryptogram C1 received from the server to get N1.
  N1 = RSAES-OAEP-DECRYPT(Client PrivateKey, C1, EMPTY).
- **ii.** The client then extracts the random number R1 and verifies whether it has the same value as the random number sent in the Challenge message. If the verification fails, the client sends back an error message to the server.
- **iii.** Client generates a key material K1.
- **iv.** Client concatenates random number R2 extracted from the server Response message, and its own key material K1 to get N2 = R2||K1.
- **v.** The client then computes cryptogram
  C2= RSAES-OAEP-ENCRYPT(Server Publickey, N2, NULL).
- **vi.** Then Response message is sent to the server consisting of cryptogram C2. Random number R1 is an 8-byte long number and Key material K1 is a 16-byte long number. After receiving Response message from the client, the server processes this message and responds as follows.

  a. The server decrypts cryptogram C2 received from the client to get
  N2 = RSAES-OAEP-DECRYPT(Server Privatekey, C2 , NULL).
  b. The server extracts the random number R2 from cryptogram C2 and verifies whether it is same as random number sent in the server Response message or not. If the verification fails, the server sends back an error message to the client. Otherwise the server responds back with an OK message. After both the client and the server, have been mutually authenticated, session keys for confidentiality KEN C and integrity KM AC are calculated as follows.

1. KEN C = HASH ((K1 ⊕ K2) || 0x00000001) [12]
2. KM AC=HASH ((K 1 ⊕ K 2) || 0x00000002) [12]

HASH algorithm used in these calculations is negotiated during the handshake stage.

Protocol Selection for Our Application:

While using TLS protocol, both communicating parties have to maintain all ex- changed handshake messages. Hash calculated on all these handshake messages is used in CertificateVerify and Finished (section5.2.6) messages. Hence all handshake messages need to be stored in memory by both communicating parties. Therefore TLS is a 'stateful' protocol.
Protocol described in section 5.3 is 'stateless' protocol. It does not need to store previously exchanged messages between communicating parties. Communicating parties just need to store key material, and challenges issued and the key material received. Memory requirement is therefore smaller compared to that in TLS protocol. This protocol is also computationally less expensive compared to TLS protocol.

1. In the first scenario, secure communication channel is established between the mobile device layer on mobile and command library layer on an ATM. We can use this secure communication protocols, to establish secure communication channel.

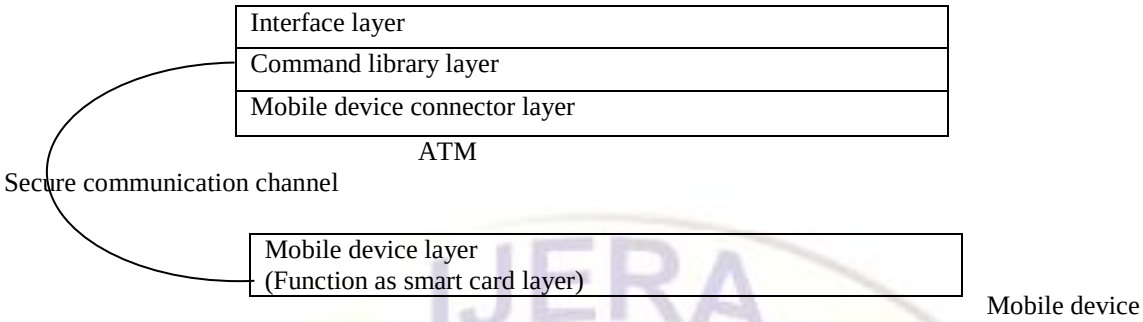

Figure 4: Deployment Scenario 1

2. In the second scenario a smart card is used to store account information and personal information of the user. In SCOSTA-PKI OS, challenge and response are stored in memory only for the next command. It does not store the previous messages due to memory and security restrictions. So protocol supported by the smart card has to be 'stateless' protocol. Structure of RSA algorithm based mutual authentication and session key establishment protocol is similar to the mutual authentication protocol supported by SCOSTA-PKI [11]. Therefore, in this scenario, we use RSA algorithm based mutual authentication and session key establishment protocol to establish secure channel between smart card layer and command library layer.

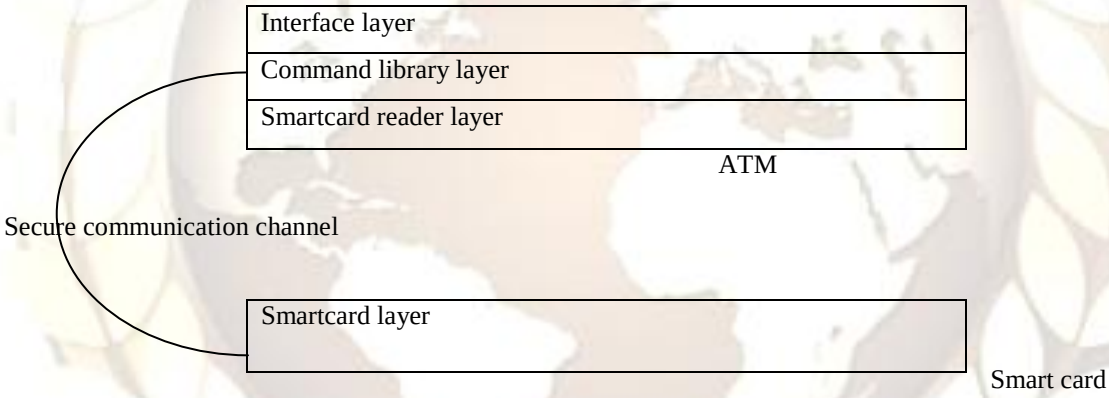

Figure 5: Deployment Scenario 2

3. In the third scenario also, secure channel between smart card layer on the smart card and command library layer is established using RSA algorithm based mutual authentication and session key establishment protocol for the same reasons. We also establish secure communication channel between the mobile device and the ATM. We can use TLS protocol to establish this secure channel.

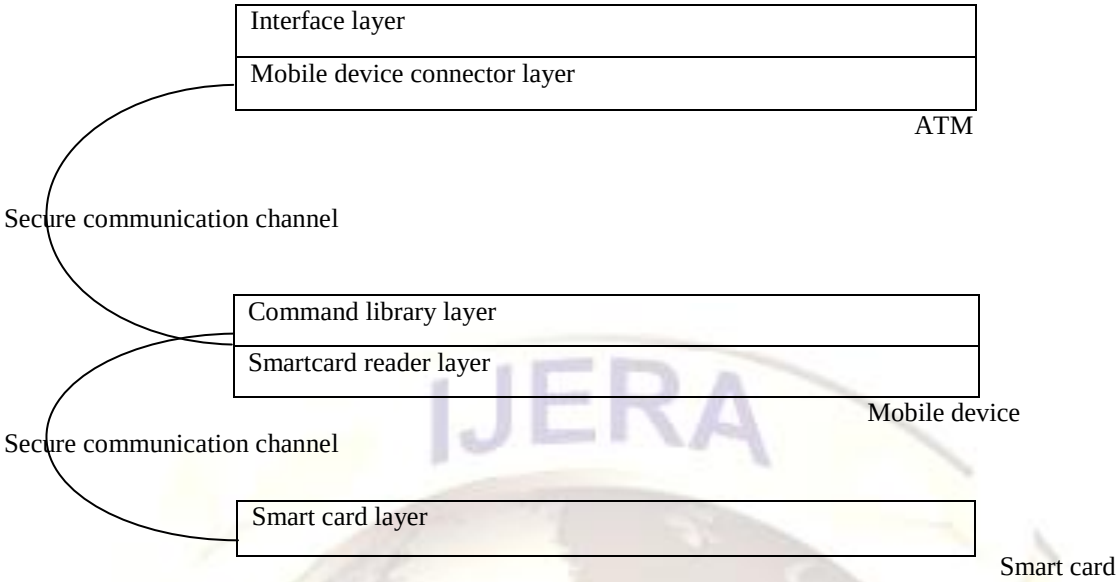

Figure 6: Deployment Scenario 3

6. RESULTS & OBSERVATIONS

Trust on financial outlet and transaction cost are two main issues faced by current financial service model. Our proposed model tackles these issues as described below.

6.1 Trust and Security Issue

In our approach, we establish mutual authentication between smart card of a user and the outlet using PKI. A fake outlet will not be authenticated by the smart card and therefore personal and confidential data of the user will not be revealed. Therefore, a fake ATM attack (section 1.1) is not possible. Since user can identify whether outlet is genuine or not, he will not enter his PIN on untrusted outlets. In this way, attacks like shoulder surfing (section 1.2) and fake keypad overlay attack (section 1.4) can be avoided in most cases. The communication between the outlet and the smart card is in encrypted form. This makes skimming devices (section 1.3) useless.

6.2 Transaction Cost

In the current financial service model, a dedicated link is maintained between the outlet and the bank database server. The link is used to authenticate the users and to update transaction logs. In our approach, there is no need to maintain the dedicated link as, user authentication is performed locally. The transaction logs are stored securely on the smart card and on the ATM. These transaction logs can be updated periodically to the bank database server by the ATM. Therefore, transaction cost is reduced significantly.

7. Conclusion

In this work, we discussed secure communication protocol based on TLS Protocol and based on RSA algorithm which provides mechanism for the mutual authentication and session key establishment between the smart card and financial outlet. Based on the deployment scenario, the secure communication protocol is selected. For our application deployment scenario, we utilized RSA algorithm based mutual authentication and session key establishment protocol as secure communication protocol. Our financial service model is secure model with low transaction cost (section 6.1 and 6.2). With low transaction cost, it is feasible to introduce financial services in less- developed remote areas and contribute to the development of these areas.

We have provided a secure two way handshake protocol with for various deployment scenarios consisting smart card and financial outlet (ATM). Secure channel is established between the smart card and the financial outlet using a secure communication protocol based on the public key infrastructure (PKI). All Communication between these two entities is in encrypted form.

8. References

[1] Nitin Munjal and Rajat Moona. “Secure and Cost Effective Transaction Model for Financial Services”. OPNTDS-09, 2009.
[2] Universal serial bus - implementer’s forum. http://www.usb.org/home.
[3] A. Gaurav, A. Sharma, V. Gelara, and R. Moona. “Using Personal Electronic Device for

Authentication-Based Service Access". In Communications,2008, pages 5930–5934. ICC'08, IEEE International Conference, May 2008.

[4] Ashok Jhunjhunwala. "Banking towards rural empowerement: challenges and oppurtinities" September 2006.

[5]I.DieBold, ATM Fraud and security http://www.diebold.com/atmsecurity/resources.htm, September 2006.

[6] Dierks,T. and Rescorla, E. RFC5246:The Transport Layer Security (TLS) Protocol Version 1.2 RFC Editor United States, August 2008.

[7] W. Diffie and M. E. Hellman. Multiuser cryptographic techniques. In AFIPS'76: Proceedings of the June 7-10, 1976, national computer conference and exposition, pages 109–112, New York, NY, USA, 1976. ACM.

[8] Nitin Munjal, Ashish Paliwal, and Rajat Moona. "Low Cost Secure Transaction Model for Financial Services".In International Conference on Security and Identity Management(SIM)-09. IIM Ahmedabad, India, May 2009

[9] EMVco. EMV standards. http://www.emvco.com/.

[10] Fengling Han, Jiankun Hu, Xinghuo Yu, Yomg Feng, and Jie Zhou. "A novel hybrid crypto-biometric authentication sceeme for atm based banking applications". In David A.Zhang and Anil K. Jain, editor, ICB,volume 3832of Lecture Notes in Computer Science, pages 675–681. Springer, 2006.

[11] Venkata Rao Pedapati , Sri Simil Dutta "Public Key Infrastructure in SCOSTA" (2007) .

[12] Bluetooth special interest group.http://www.bluetooth.org.

[13] http://en.wikipedia.org/wiki/Automated_teller_machine

[14] M. Pathak, U. P. Singh, N. Bhatnagar and G.Srivastava "An Efficient Architecture for Trusted Financial Services in ATM" IJERA Vol. 2, Issue 1, Jan-Feb 2012, pp. 623-630.

[15] http://en.wikipedia.org/wiki/Smart_card