Military Technical College
Kobry El-Kobbah,
Cairo, Egypt

8th International Conference
on Electrical Engineering
ICEENG 2012

Moderate EAP-TLS Protocol - A New Approche

By

Ahmed ELNagar*   Dr. Ahmed Abd-EL-Hafez**   Prof. Dr.Adel ELHnawy***

Abstract:-

Security has always been a major concern in the development of Wireless Local Area Networks (WLAN) as they are based on wireless technology, i.e. no physical connections exist. One of the major problems in WLAN security is authentication. EAP (Extensible Authentication Protocol) is an authentication framework that uses several authentication techniques such as TLS, TTLS, PEAP and LEAP. EAP-TLS (EAP- Transport Layer Security Protocol) is widely used in WLAN (Wi- Fi /802.11 and Wi-MAX/802.16) as a solution to the authentication problem. This paper presents EAP-TLS security in details, explores its weaknesses and provides detailed EAP-TLS assessment. As a solution to the presented flaws, we present in details the proposed Moderate EAP- TLS for overcoming such shortages. Finally, we present a security assessment to our proposed protocol compared to EAP-TLS.

Keywords:

WLAN; Authentication protocols; EAP-TLS; LEAP.

* Egyptian Armed Forces

** Military Technical College staff, Cairo, Egypt

*** College of Engineering Staff, Ain Shams University, Cairo, Egypt

1. Introduction:

Securing communication in (WLAN) [1, 2, 3, 4, 5,6] is a complex problem. A mobile client needs to have a secure way to mutually prove and verify the other contact and ensure safe data exchange free of tampering or sniffing.

There are three main requirements that must be fulfilled for a successful security strategy in wireless networks: mutual authentication; private communication (privacy); and data integrity. Mutual authentication is to verify the identities of the client and authentication server; so both peers have interest in verifying their identities to each other. Private communication (privacy) is ensured by sending secure information through open space (accessible to everyone) using strong encryption algorithms and dynamic key derivation strategy. Finally, integrity means that the data is intact when received by both parties.

EAP-TLS is the protocol that verifies these three requirements, where EAP is a general authentication protocol (framework) mostly used in wireless networks point-to-point connection as the basis for two-way negotiation and transfer of authentication information between the client and the other party. EAP [7] works using several authentication protocol methods such as [8, 9, 10, 11, 12, 13, 14] (TTLS, TLS, PEAP and LEAP).

EAP-TLS stands for (EAP -Transport Layer Security) and is one of the public-key based authentication techniques that use a public and a private key pair. It also uses an independent third party that issues Certificate Authority (CA) to establish trust between the authenticator server and supplicant. EAP-TLS provides mutual authentication between the supplicant and authentication server. It also provides dynamically generated session encryption keys to ensure data integrity and secure connection between the supplicant and authentication server.

One drawback of EAP-TLS is that it requires both the supplicant and authentication server to have valid certificates, which poses significant management complexity. Other drawback of EAP-TLS is that it requires high power resource at client side to perform the complexity of uses public and private key pair computations.

Figure (1) below illustrates the three EAP-TLS components involved in the authentication process: the user (supplicant); the authenticator and the authentication server.

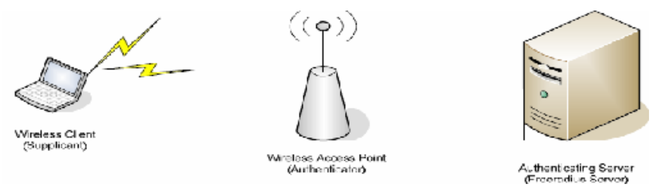

Figure (1): *The Three 802.11i authentication components*

This paper is organized as follows: section (1) gives a brief introduction about WLAN security; Section (2) explains how EAP- TLS works and displays EAP-TLS message flow; Section (3) explores the EAP- TLS assessment; Section (4) presents the new proposed Moderate EAP- TLS operation and its message flow; Section (5) assesses Moderate EAP-TLS; and finally, section (6) sums up both the conclusion and future work.

2. How EAP-TLS Protocol Works and Its Message Flow

EAP (Extensible Authentication Protocol) [16,17] protocol is a general point-to-point authentication framework protocol that supports multiple authentication mechanisms for authentication process between the two parties; the client (supplicant) and the authentication server via an authenticator. TLS (Transport Layer Security) protocol provides privacy and data integrity between two parties. The TLS protocol is composed of two phases; record phase: which is responsible for shifting and transferring negotiation messages of the handshake phase and data between the two parties using the parameters agreed upon in the handshake phase; and the handshake phase: which is responsible for negotiation the record phase parameters such as encryption algorithm and cryptography keys used in connecting the two parties.

This seems as counter intuitive, however, the TLS is designed to handle this bootstrap process and operates as follows: The record phase stores four connection states (group of settings or parameters); two for the transmitting direction and two for the receiving direction (one for the current state, while the other is for the pending state respectively).

The current state settings or parameters refer to those already in operation, while the pending state refers to settings or parameters that are being prepared for next use. When a change occurs, the pending state becomes the current state and a new empty pending state is created.

So in the initial state, the current state is "Null” and the record phase just forwards the messages and data without any security parameters, because there are no keys or encryption method.

During this time, the handshake phase operates to build up the security parameters (keys and encryption method) in the pending state. The pending state then becomes the current state and the security parameters (keys and encryption method) are set on data traffic. The record phase start operating with these security parameters between the two parties and a new empty pending state is created.

TLS protocol is implemented directly through the EAP protocol and the two TLS protocol phases are concerned with negotiating security settings to connect the two ends of the link and transfer data between them based on such settings while the EAP protocol is concerned with the authentication process that takes place between the two ends of the link.

A secure connection between the two ends of the link, the client (i.e. supplicant) and the authentication server (through an authenticator), is established via EAP-TLS protocol through using a series of messages exchanged between the two parties in a specific order as illustrated in figure(2)below.

1- **EAP Request (ID) Message** → the authenticator requests the client's ID.
2- **EAP Response (ID) Message** → the client responds and sends their ID to authentication server via the authenticator.
3- **EAP-TLS Initiation Request Message** → the authentication server sends a request to the client via the authenticator to start applying EAP- TLS protocol.
   Client Hello Message → the client sends a hello message to the authentication server via the authenticator in response for the EAP-TLS initiation request.

The client's hello message comprises a list of cipher suites (certificate types, encryption methods, and integrity examination techniques), compression methods supported by the client and the client's hello random number. The client's hello random number can be any value but should be completely unpredictable to everyone. It is used as a nonce to ensure the freshness of protocol implementation.

When the authentication server receives the client hello message, it must check its ability to support any of the cipher suites and compression methods included in the list and inform the client via a server hello message.

Server Hello Message → the authentication server sends it to the client via the authenticator in response for the client's hello message.

The server hello message includes the server's hello random number, which can be any value except for the value of the client's hello random number and should be completely unpredictable to everyone. It is used as a nonce to ensure to ensure the freshness of protocol implementation. This message also includes session ID which used to detect the ID of the session taking place between the client and the authentication server, server certificate which comprises the authentication server's name and it's public key and shall be signed by the private key of a third party called "Certificate Authority", in order to prove integrity and privacy of the authentication server's public key. The client checks the server certificate and gets the server public key through using the public key of the certificate authority ,client certificate request where the authentication server sends a request to the client to identify itself nothing else, since it is unusual to have a certificate authority to issue a client certificate, and server done which used to notify the client that the server hello message is complete and that the authentication server expects the client to take the next step.

So far, both the client and authentication server have exchanged greetings, resulting in a session ID, two random numbers, authentication server certification, as well as the agreed-upon cipher suite and compression method.

4- **Server Certificate Verification Message** → the client sends this message to the authentication server after verifying its certificate via the certificate authority's public key and receiving the authentication server's public key.

5- **Client Certification Message** → upon receiving the client certification request message from the authentication server, the client sends this response message. however, since the client do not have a certification authority issuing certificate for him, client proves their identity by hashing together all the messages exchanged between him and the authentication server timely so far via using a one-way hash function and signing this message using their private key.

The authentication server uses the client's public key to validate the client's signature on this message. Then, the authentication server does the same process hashing together all messages exchanged between him and the client so far in another message, using a one-way hash function, and compares the two messages. If matching, the authentication server shall be certain that the same client has both the public and private keys, and the client presents their identity to the authentication server.

6- **Client Key Exchange Message** → the client generates the pre-master key (a 48-byte or 384-bit random number) and encrypts this pre-master key using the authentication server's public key. The client then sends it to the authentication server in a client key exchange message.

The server receives the client exchange key message, decrypts it by using the authentication server private key, and gets the pre-master key.

Now, both client and server, having the pre-master key, client random number and the authentication server random number, they can combine the three values by using a one-way hash function to produce the master key(48-byte or 384-bit) used with the encryption algorithm to secure client/authentication server exchanged data traffic.

7- **Changing Connection State Message (Change Cipher Message),** → remember that a current state connection is turned on at the initialization of protocol with no keys or encryption method. However, now both the client and the authentication server are able to set up the pending state connection and then switch it to become the current state connection, where a new pending connection state is ready to be turned on.

When the switch is performed, each side sends to the other a change connection state message (change cipher message) indicating that the switch to encryption is complete.

8- **Finish Message(from client to authentication server and vice-versa)** → firstly, the client sends to the authentication server the hash value message of all the messages exchanged, starting from the client hello message up to the finish message (excluding the finish message) covering with the master key. Secondly, the authentication server computes the same corresponding hash value message covering with the master key and sends it to the client. If the hash value messages received by both sides match the original ones they both issued, then the finish message proves the validity of master key and cipher suite at both sides.

9- Empty EAP-TLS Response Message → this empty message is just a response from the client to the authentication server; the client has nothing more to say.

10- EAP –TLS Success Message → this message is sent from the authentication server to the client indicating success of the EAP-TLS protocol and the secure traffic of data exchanged between them will be ready with the master key.

3. EAP-TLS Protocol Assessment

1) Strong [15] mutual authentication is achieved because EAP-TLS protocol is based on public and private keys pairs and on certification authority technology at both sides of the authentication process.

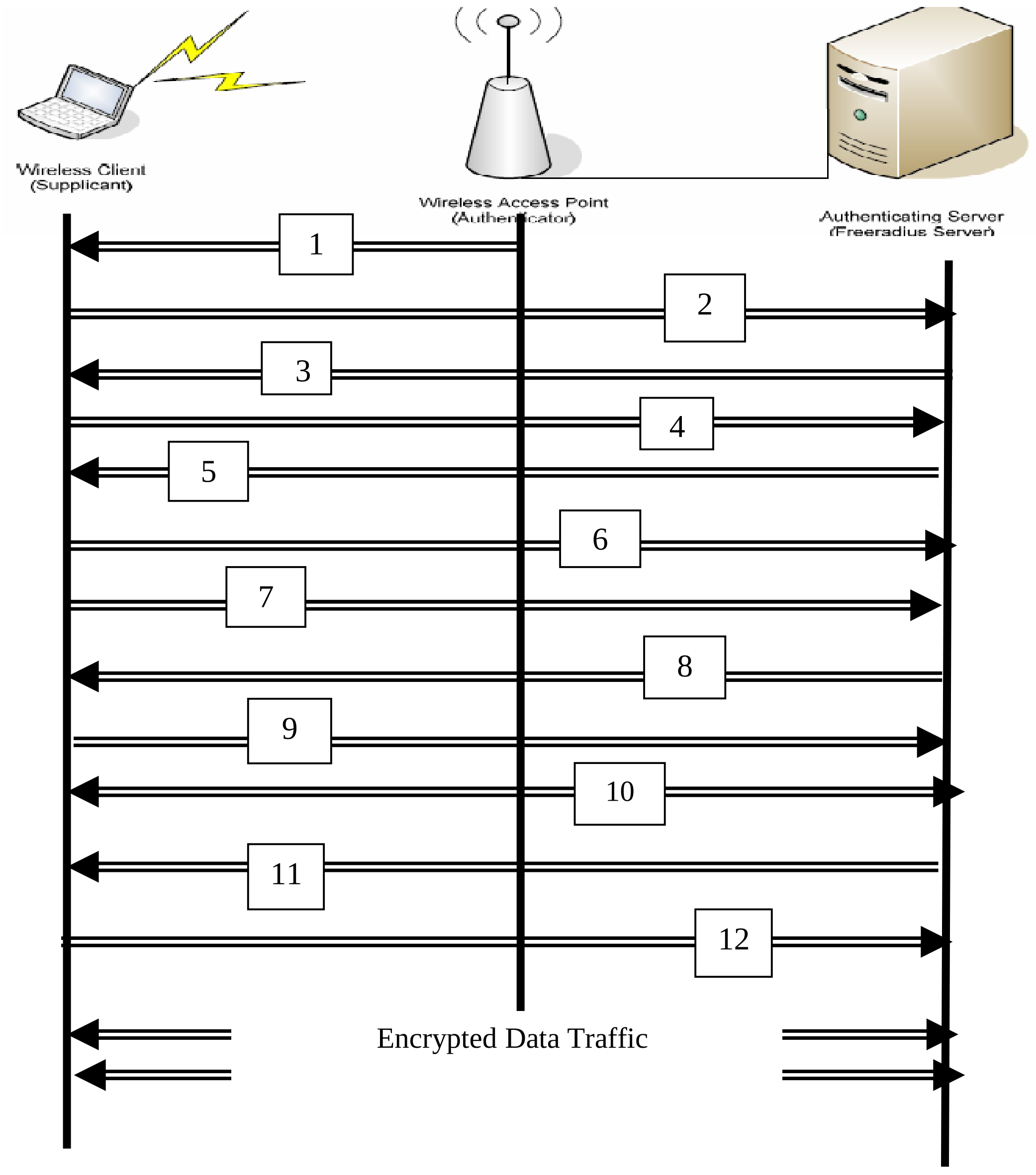

Figure (2): *EAP-TLS Message Flow*

2) Privacy of exchanged data traffic is achieved because EAP-TLS protocol uses a dynamic and strong master key.

3)EAP-TLS protocol uses dynamic master key technology because the EAP-TLS protocol generates new master key for each new access time using changeable inputs such as client, authentication server random numbers (nonce) and the pre-master key.

4)EAP-TLS protocol supports the integrity of data traffic exchanged between client and authentication server because they check out the matching of the two hashing values of messages they exchange at several points through the execution of the protocol, such as client certification message and finish message point.

5)EAP-TLS protocol supports high resistance against replay attacks and man-in-the middle attacks, because EAP-TLS protocol uses the dynamic master key technology for each new access time to the network.

6)EAP-TLS protocol uses a strong dynamic master key because the EAP-TLS protocol generates a master key of 48 byte or 384 bits in length.

7)EAP-TLS protocol supports high protection for master key against over-the-air attacks as the master key is generated locally on both sides for each new network access time and is not exchanged over the air, so the attacker cannot eavesdrop easily or sniff the master key because must apply some dictionary attack methods to get the master key for each new network access time.

8)EAP-TLS protocol reduces processing time and saves consumed power resources for the re-authentication process by creating a record called session ID on the authentication server side in the server's hello message and sending it to the client. This session ID [19] is sent again from the client to the authentication server in the client's hello message to resume a disconnected session with the client. Instead of executing the whole protocol again, EAP-TLS protocol skips all protocol procedures after the two hello messages and jumps directly to the two finish messages creating the resume session master key by hashing the disconnected session master key with the two resumed session random numbers (nonce) in the resumed two hello messages.

9) The number of authentication messages in EAP-TLS protocol exchanged between the two ends is (21) messages. This represents a high overhead and longer time in executing the protocol due to exceeding processing time up to (1.8 sec) for all messages.

10) EAP-TLS protocol does not support the client identity privacy because the client ID is sent in clear format at the beginning of the protocol.

11)EAP-TLS protocol supports a strong mutual authentication through using public key cryptography and certificate authority technology. The former consumes limited resources of the client handset battery in complicated computations, while the latter is not suitable for the client because it is not user-familiar to execute and also the client only needs to prove itself and does not need to provide any more certification while the authentication server needs to prove itself also need to provide its public key certification.

4 . New Moderate EAP-TLS Protocol Operation and Its Message Flow

All the message flow of Moderate EAP-TLS protocol exchanged between the client and the authentication server as same as order of EAP-TLS protocol messages order ,also as same as content of EAP-TLS protocol messages content as former shown in figure-2 message flow of EAP-TLS except of the messages client certificated verification message(7) and finish message(10) will be changed to make the Moderate EAP-TLS protocol user-familiar , reduces processing time and saves consumed power resources to execute but achieves same other features as same as EAP-TLS protocol. Description of the new Moderate EAP-TLS protocol will be shown below.

- Client Certification Message: (message 7) is that sends from the client as response message upon receiving the client certification request message from the authentication server, the client sends this response message to prove its identity. However, since clients do not have a certification authority for issuing certificates, the client prove its identity to authentication server by hashing together all messages exchanged between them so far via a one-way hash function and signing this message using its private key.

Authentication server uses the client's public key to validate the client's process and hashes together all messages exchanged between them so far in another message using a same one-way hash function and comparing the two messages. If matching, the authentication server shall be certain that client only have its private key and corresponding public key, and the client present its identity to the authentication server. In addition, they check out the integrity of messages, which exchanged between them up to now.

This technology is used instead of using a third party called the "Certificate Authority" at client side; since the client is unusual to have a Certificate Authority issuing its certificate, also the client is not familiar with its issuance.

However, this technology requires a considerable time to be carried out at both client and authentication server sides. It also consumes the limited power resources of the client's handset battery so that we suggest the following technology to overcome such drawbacks.

At the client certification verification point, both the client and the authentication server have exchanged their own random number (nonce) to each other. Thus, the client side has both its own random number and the authentication server random number. The authentication server also has both its own random number and the client’s random number.

The client can therefore re-send the message that contains the encrypted server’s random number through the client’s private key to the authenticator server. The authenticator server decrypts the message by the client public key and gets the server random number. It would then checks the matching between the one it got against the original one. If they match, the authenticator server validates the client and the client proves itself to the authenticator server; as that client only has its private key and corresponding public key. The client and the authenticator server also check the integrity of messages exchanged between them up to now.

Finally, the content of message (7) changed from hash form of all messages that sent and received up to this point at client side encrypted by the private key of the client {hash form} private key of the client to the authentication server random number at client side encrypted by the private key of the client {server random number} private key of the client.

-Finish Message: (message 10) is that exchanged between the client and the authentication server. It contains the hash value message of all the messages exchanged, starts from the client hello message up to the finish message, but without the finish message itself and covering the hash value message with the master key.

Then, the finish message is decrypted at both sides with the master key to get the hash value message.

If the hash value message received at both sides matches with the original issued at the both sides, then validates the correctance of the master key ,the cipher suite (certificate type, encryption method, integrity examination technique), and the integrity of messages exchanged at both sides.

However, this technology still consumes high time to execute at client and authentication server sides and consumes the finite power resources of the client's handset battery. In oared to overcome such drawbacks we propose the following technology.

Up to the finish message point, both the client and the authentication server have exchanged their own random number (nonce) with the other side. Thus, the client side has its own random number and authentication server random number. The authentication server also has its own random number and client random number.

Then, the client encrypts the authentication server random number by using the master key, and the authenticator server encrypts the client random number by using the master key. Both sides then exchange the encrypted message with each other.

At each side, each one decrypts the received message by using the master key, gets its random number and checks out if it matches with the original created at that side itself.

If it checks, both sides then insure the correctness of the master key as well as ensure the integrity of messages exchanged between them from beginning until the end of executing the whole protocol.

Finally, the content of message (10) changed from hash form of all messages that sent and received up to this point at both client side and authentication server side encrypted by the master key {hash form} master key to the authentication server random number at client side encrypted by the master key {server random number} master key and client random number at authentication server side encrypted by the master key{client random number}master key.

5 .Moderate EAP-TLS Assessment

1)The number and order of exchanged messages between the client side and the authenticator server side does not changed but only changes the content of both messages; the client certificate verification message and the finish message.

2)Reduces the processing time and saves the required power resources to execute the protocol by cancelation the computation complexity to construct the hashing message in the client certification verification message and the two finish messages.

3)The client does not need certificate authority at client side for client certification verification ,the client is not familiar with it , because it can prove itself to the authenticator server by encrypts the server random number message at client side with the client private key and decrypts that message at authenticator server side with the client public key.

4) Reduce the time required to exchange the client certification verification message and finish message between both sides because minimizing the content of these messages.

5)Keep the security issues (privacy and integrity)strong in Moderate EAP-TLS as same as EAP-TLS because no change in the procedures of generation of the master key and also keep the mutual authentication issue strong because still uses the public and the private key pair cryptography technology in authentication process .

6)Moderate EAP-TLS does not support the client identity privacy, because the client ID sends in plaintext format from client to authentication server over the air, so it can easily eavesdrop and tracks the client by its ID.

6. Conclusion and Future Work

In this paper, we presented in details EAP- TLS authentication protocol for WLAN security. In addition, we offered a new proposal protocol called, Moderate EAP-TLS to overcome the disadvantages of EAP-TLS protocol in details.

Although EAP- TLS achieves mutual authentication, session ID record to resume disconnected session, dynamic encryption master key for privacy of data traffic and also achieves integrity of received data, added to that uses the public and the private key pair cryptography technology and have robust against reply attack and dictionary attack; it is not support client identity privacy , suffering from computation complicity of uses a third party called certificate authority to issue the certification verification at both sides.

Moderate EAP-TLS minimize the computation complicity of uses a third party certificate authority by uses of it at the authenticator server side only and does not use one-way hash function to construct the hashing message in the client certification verification message and the two finish messages.

Moderate EAP-TLS still suffering from uses the client identity in plaintext formats.

Finally, we recommended using Moderate EAP-TLS to keep both security of mutual authentication process and protection of exchanged data traffic in WLAN in same level of EAP-TLS but minimize both time and power resources required to execute the protocol.

Some points can be tackled to improve our proposed protocol; in particular, we plan to study the protection of the client identity which exchanged from client side to authenticator server "Over-the-Air" in plaintext using secure technology such as uses the public and the private key pair or shared secret key cryptography technology, especially with regard to tracking the client by its identity.

References:

[1] [S3-030689] SA3: "*Wireless Local Area Network (WLAN) Interworking Security*", 3GPP/ SA3 Security meeting, 31, November 2003.

[2] MONIS AKHLAQ, BABER ASLAM, MUZAMMIL A KHAN, M NOMAN JAFRI, "*Comparative Analysis of IEEE 802.1x Authentication Methods*", Proceedings of the 11th WSEAS International Conference on communications, July2007,PAKISTAN.

[3] John Vollbrecht, Robert Moskowitz, "*Wireless LAN Access Control and Authentication*" © 2002 Interlink Networks, Inc. Revision D. 12-02.

[4] Sean Convery, Darrin Miller, Sri Sundaralingam, Mark Doering, Pej Roshan, Stacey Albert, Bruce McMurdo, Jason Halpern, "*Wireless LAN Security in Depth*", Cisco Systems Inc., September 2005,

[5] Kwang-Hyun Baek, Sean W. Smith, David Kotz , "*A Survey of WPA and 802.11i RSN Authentication Protocols*", Dartmouth College Computer Science, November 2004.

[6] Cisco "*A Comprehensive Review of 802.11 Wireless LAN Security and the Cisco Wireless Security Suite*" October2002.

[7] B. Aboba,D. Simon, P. Eronen, " *EAP Key Management Framework*" RFC: 5247/(IETF),August 2008.

[8] "*EAP Methods for 802.11 Wireless LAN Security*", International Engineering Consortium,September 2005,http://www.iec.org/online/tutorials/eap_methods/.

[9] Michel Barbeau , Lei Han , "*A Threat Analysis of The Extensible Authentication Protocol*", technical report, Carleton University. ,April 2006.

[10] Jyh-Cheng Chen, and Yu-Ping Wang, " *Extensible Authentication Protocol (EAP) and IEEE 802.1x*", IEEE Communications Magazine, 2005.

[11] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H.Levkowetz,"*Extensible Authentication Protocol (EAP)*", RFC:3748/(IETF) June 2004.

[12] Samuel Sotillo, " *Extensible Authentication Protocol (EAP) Security Issues",* © 2007 Samuel Sotillo

[13] L. Blunk & J. Vollbrecht , " *PPP Extensible Authentication Protocol (EAP)* " RFC:2284/(IETF), March 1998.

[14] B. Lloyd,W. Simpson ,Daydreamer, "*PPP Authentication Protocols*" RFC:1334/(IETF),October 1992.

[15] Lianfen Huang , Ying Huan , Zhibin Gao, " *Performance of Authentication Protocols in LTE Environments*", Int. Conference on Computational Intelligence and Security China,2009.

[16] B. Aboba, D. Simon; "*PPP EAP TLS Authentication Protocol*" RFC:2716/ (IETF); October 1999.

[17] Narayanan,V. and L. Dondeti,, "*EAP Extensions for EAP Re-authentication Protocol (ERP)*", RFC:5296 /(IETF),August 2008.