802.11 Security –Upper Layer Authentication

Shambhu Upadhyaya
Wireless Network Security
CSE 566 (Lecture 13)

University at Buffalo The State University of New York

Common Authentication Methods

- EAP-TLS
  - Uses a TLS handshake to mutually authenticate a client and server
  - For mutual authentication both the client and the server have to be assigned individual certificates
- EAP-TTLS
  - Extends the concept of EAP-TLS
  - Uses the secure connection established by the TLS handshake to perform additional authentication, such as another EAP or another authentication protocol such as PAP, CHAP, MS-CHAP or MS-CHAP-V2
  - Establish keying material

University at Buffalo The State University of New York

Common Authentication Methods

- LEAP (Lightweight EAP)
  - Proprietary protocol developed by Cisco
  - Provides mutual authentication, temporary session keys and a centralized key management
  - Uses WEP for encrypting the data after session has been established
- PEAP (Protected EAP)
  - Similar to EAP-TTLS but only allows EAP for authentication
  - Establishes a secure tunnel before the EAP negotiation starts so that the privacy of the communication is preserved
  - Also has key exchange, session resumption, fragmentation and reassembly

Common Authentication Methods

- Kerberos V5
  - Used for a long time to provide security to IP networks
  - Can be used in RSN/WPA if there is an existing setup on the backbone
  - Uses a special document called tickets to communicate the credentials and grant access
  - Can be implemented in Wi-Fi setting by using it along with 802.1X and EAP

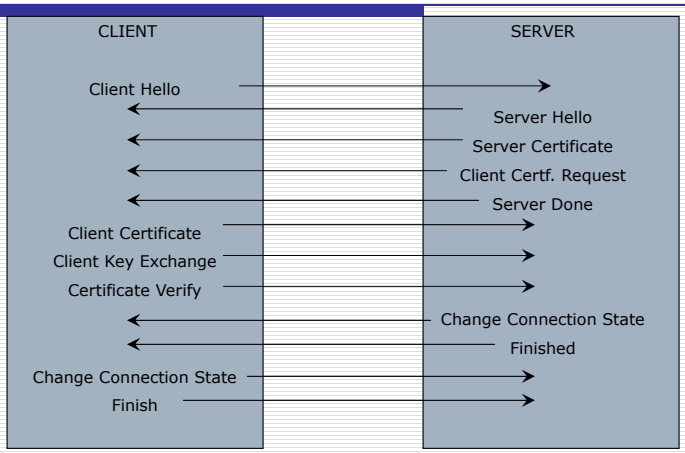
Transport Layer Security (TLS)

- Provides authentication, encryption and data compression functions
- TLS is divided into 2 layers
  - Record Protocol
  - Handshake protocol
- Record protocol is used to encrypt/decrypt and compress data
- Handshake protocol is used to negotiate the parameters for the record protocol
- Record protocol operates on groups of setting parameters called connection state

TLS Contd…

- Record protocol stores 4 connection states
  - Current transmit connection state
  - Pending transmit connection state
  - Current receive connection state
  - Pending receive connection state
- *Current connection state* refers to the set of parameters which are in use at present
- *Pending connection state* refers to the set of parameters which are being negotiated for future use
- Hence, initially the current connection state is set to null and when the parameter negotiation is completed by the Handshake protocol then TLS switches to the new state

TLS Message Exchange



TLS Message Exchange

TLS Message Exchange Contd…

- Client Hello (client -> server)
  - Contains a list of cipher suites and compression methods that the client can support
  - Also carries a random number called ClientHello.random
  - Random number is used to generate liveness
- Server Hello (server -> client)
  - Checks the supported cipher suites and replies with the message
  - Includes a random number called ServerHello.random
  - Contains a SessionID which is used by the client and server from there on
- At the end of these two messages the client and the server
  - Have syncronized states
  - Agreed on a SessionID
  - Agreed on the ciphersuite
  - Have exchanges 2 random numbers (nonces)

TLS Message Exchange Contd…

- Server Certificate (server -> client)
  - If session is being resumed then this step can be skipped
  - The server sends its certificate to the client
  - The certificate contains the public key of the server along with a signed message
  - Client uses this certificate to verify the authenticity of the server and also stores the public key for future use
  - Verification is done with the help of certificate authority
- Client Certificate (client -> server)
  - Server might request the client to send a certificate which is then sent in this message
  - Client certificates are generally used by IT organizations for their internal use
  - Client certificates are typically used only when there is a in-house certificate authority; otherwise, client can refuse the server request

TLS Message Exchange Contd…

- Client Key Exchange (client -> server)
  - This phase is used to create a mutual shared secret key between the client and the server
  - But client first creates a secret number (pre-master key) key and sends it to the server by encrypting it with the servers public key
- Client Certificate Verification
  - Used to prove to the server that the client is authentic
  - Client generates a hash of all the messages it had received till this point and signs it with its private key
  - The server then verifies the contents by checking the signature using the key from client's certificate

TLS Message Exchange Contd…

- Now both the client and server share the following information
  - Pre-Master secret
  - Client random number (nonce)
  - Server random number (nonce)
- Secret master key is generated by using the random numbers exchanged in the hello messages and the *Pre–Master Secret*
  - Both client and server do this by combining these cryptographically to generate a 48 byte master key
- Change Connection State
  - As all the required keys have been exchanged, the state of the connection is switched from current (no encryption) to the new pending state (with encryption)
- Finished
  - Each side sends a finished message to the other to ensure that the exchange was successful
  - The message is encrypted using the new keys which were setup during the encryption process

University at Buffalo *The State University of New York*

TLS over EAP

- To use TLS in WPA/RSN, it is used over EAP so that it can be tied to 802.1X

Code	Identifier	Length	Type	Request/Response Data

EAP Message Format

Code	Identifier	Length	'13'	Flags	Length	EAP-TLS Data

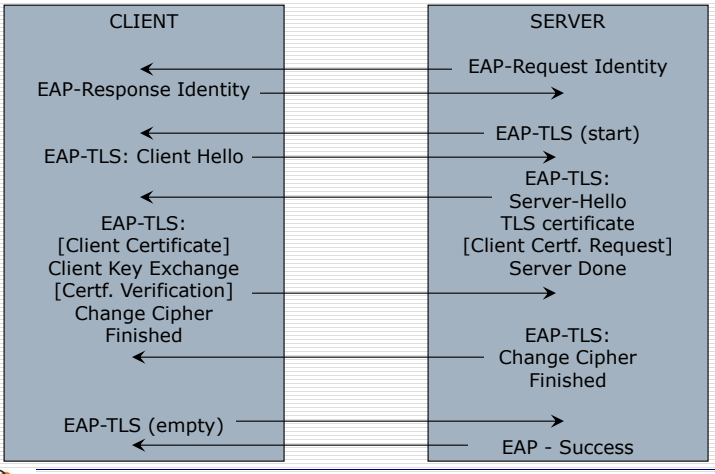
EAP-TLS Message Format

University at Buffalo *The State University of New York*

EAP-TLS

- For EAP-TLS messages the type bit is set to '13'
- EAP-TLS messages have the following special fields
  - Flags
    - Length included flag
    - More fragments flag
    - Start flag
  - Length
    - This field is used to indicate the length of the whole TLS packet
    - This might be different from the first length field as the packet might be large and could be fragmented
    - So the first length field is used to define the length of the current TLS frame whereas the second tells the length of the complete TLS packet before fragmentation

EAP-TLS Handshake



EAP-TLS

- Use of EAP proved key features for implementing TLS with WPA/RSN
  - No IP address is needed in the initial authentication protocol
  - Wireless devices can talk with the Access point and perform complete authentication before being granted access to the network
  - The access point is not required to understand TLS messages

LEAP (Lightweight EAP)

- Proprietary protocol designed by Cisco
- LEAP also divides the system into a supplicant, authenticator, and authentication server
- Supplicant resides in the mobile device
- Authenticator is placed in the access point
- The authentication server is implemented by a RADIUS server
- It is basically a challenge response protocol based on a shared secret key
- It has mutual authentication between the mobile device and authentication server, with separate challenges and responses being issued by both

LEAP

- After the completion of the mutual authentication a secret shared key is sent to the access point in a RADIUS attribute
- The client also computes the same secret key, which from then on is used for communication between the access point and the client
- Access point signals a successful authentication to the client by sending a EAPOL-Success message and activates the key by sending a EAPOL-Key message

LEAP

- LEAP originally ran over WEP, which has known weaknesses
- The ability to generate temporary session keys reduces these vulnerabilities to some extent
- Benefits of LEAP include
  - Mutual Authentication
  - Temporary session keys
  - Centralized Key Management

Protected EAP (PEAP)

- The aim of PEAP is to do the Authentication in private
- As the negotiation is performed on an encrypted channel the client's identity is not revealed
- Overcomes the following security weaknesses of EAP
  - As EAP-identity message is unencrypted, it can be snooped to learn the identity of the user
  - EAP-Success/Fail messages are also unencrypted and could be spoofed by the attacker

PEAP

- PEAP uses a two phase approach for EAP negotiation
  - In the first phase client uses EAP to establishes a secure connection with the server without any identification information about the client being revealed
  - In the second phase the secure connection established in the first phase is used for another complete EAP negotiation in which mutual authentication is performed

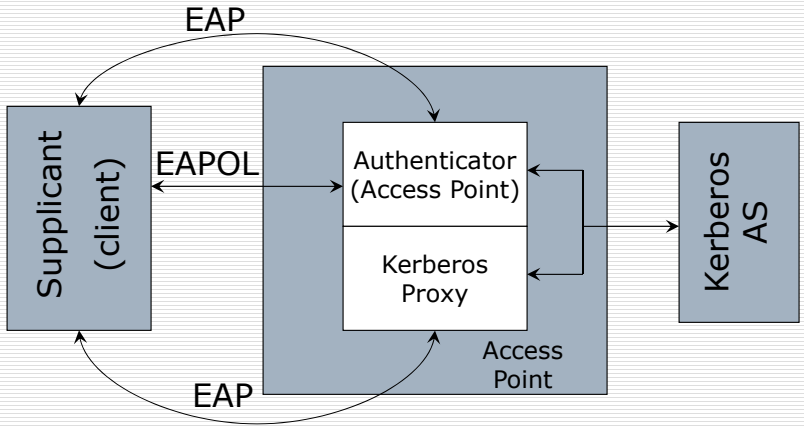
PEAP

- Phase 1
  - Initially the server sends a EAP-Request/Identity message
  - Client replies with an Identity-Response message, which in this case is anonymous
  - Similarly it can deny the request for the client certificate as it can compromise its identity
- Phase 2
  - This phase is the conventional EAP negotiation with the only difference being that it is carried out over a secure channel
- Hence, PEAP allows the client to go through Phase 1 without authentication as a malicious client would anyways be disconnected by the server during the Phase 2 of EAP negotiation

Kerberos V5

- IEEE 802.11 RSN doesn't specify the exact implementation of Kerberos
- Generally used when the backend network has a Kerberos implementation already in place
- Kerberos uses special documents called Tickets which are used to communicate the credentials of the user to the network
- These tickets are issued by a central Ticket Granting Service
- The user initially authenticates to a central Authentication Service which gives a Ticket Granting Ticket (TGT)
- Using this ticket the client then requests tickets for other services from the Ticket Granting Service

Kerberos V5



Kerberos V5

- When the mobile device comes in contact within range of the Access point, it communicates with the Authenticator which is closely connected to the Kerberos proxy
- The Kerberos proxy then uses EAP to communicate with and to find the identity of the client
- The proxy uses this information about the client i.e., the identity and the secret key info. to make a request for access to the Kerberos AS on clients behalf
- If the TGT is granted by the AS, it is passed back to the proxy and then the client
- The client then presents this ticket to the access point service to get access to the network

References

- Jon Edney and William Arbaugh, Real 802.11 Security, Addison-Wesley, 2004, Ch. 9