Fortification of Transport Layer Security Protocol

Kuljeet Kaur
(Assistant Professor)
School of Computer Applications, Lovely Professional University (Phagwara –[PB]. India)

ABSTRACT
Proving an identity over a public link is complex when there is communication between Client and Server. Secure Shell protocol is deployed, to determine a client's identity through Password-based key exchange schemes, over a public network, by sharing a (short) password only, with a session key. Most of the existing schemes are vulnerable to various dictionary attacks. SSL is the de facto standard today for securing end to end transport. While the protocol seems rather secure there are a number of risks which lurk in its use. The focus of the paper is on the analysis of very efficient schemes on password-based authenticated key-exchange methods. In this paper analysis of AuthA key exchange scheme and DH-EKE is done and complete proof of its security is generated. Evidences are generated to show that the AuthA and DH_EKE protocol and its multiple modes of operation are secure under the computational Diffie-Hellman intractability assumption and help in fortification of transport layer security protocol.

Keywords: Password Authentication, Diffie-Hellman Key Exchange, Secured Socket Lock.

1. INTRODUCTION
Currently all standard methods for authentication in
TLS rely on a public-key infrastructure (PKI). it might not suit environments where the infrastructures is "light-weight".e.g times when a system has to be bootstrapped from scratch. There is a class of authenticated key-exchange protocols based on human-memorizeable weak passwords which are resistant to (off-line) dictionary attacks. They do not have to be backed by any infrastructure such as a PKI.
TLS is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. The Record Protocol encapsulates higher level protocols (such as HTTP [7]) and cares about the reliability, confidentiality and compression of the messages exchanged over the connection. The TLS Handshake Protocol is responsible for setting up the secure channel between server and client and provides the keys and algorithm information to the Record Protocol. In this paper AuthA and DH-EKE protocols are used to show the fortification of transport layer security protocol.
In AuthA model(which is Encrypted Key Exchange evolved into proposal AuthA which is formally modeled by One-Encryption Key-Exchange) the protocol entities are modeled through oracles, and the various types of attacks are modeled by queries to these oracles. This model enables a treatment of dictionary attacks. The security of AuthA against dictionary attacks depends on how many interactions are carried out against the protocol entities rather than on the computational power [1, 3].Another protocol used is Diffie-Hellman Encrypted Key Exchange into TLS. The new cipher suite provides mutual authentication and key establishment with perfect forward secrecy over an insecure channel and limits the damage in case an attacker gains access to the server's databases. It uses TLS_DHE_DSS_WITH_DES_CBC_SHA. This means that the session key will be based on a Diffie-Hellman key exchange [8] using ephemeral parameters, DSA is the signature algorithm used and the security on the record layer will be based on DES in CBC mode and SHA-1.
The structure of the remainder of the paper is as follows. In Section II explanation of AuthA is given with security proofs. In Section III details of DH-EKE is given with the assumed cipher suite using ephemeral parameters to prove security. In Section IV fortification of Transport Layer Security Protocol is shown and Section V concludes the paper.

2. AUTHA KEY EXCHANGE METHOD
In this model adversary's capabilities are modeled through queries for security against dictionary attacks. The players in this model do not deviate from the protocol and the adversary is not a player, but does control all the network communications. Denotations are server S and a user, or client, U that can participate in the key exchange protocol P. We denote client instances and server instances by U i and Sj and I when we consider any kind of instance. The client and the server share secret *pw* drawn from a small dictionary Password of size N. The protocol AuthA consists of the following algorithm:
The key exchange algorithm KeyExch(Ui; Sj) is an interactive protocol between Ui and Sj that provides the instances of U and S with a session key sk. Various queries are asked by A adversary to all the participants in the model like Execute(Ui; Sj), Reveal(I), Send(I;m) Send(Ui; Start). Another goal of A is to impersonate the client or the server. The probability that A successfully impersonates a client instance in an execution of P: this means that a server would accept a key while the latter is shared with no client. The protocol P is said to be C-Auth-secure if such a probability is negligible in the security parameter. AuthA which is formally modeled by One-Encryption-Key-Exchange which enables us to avoid many compatability problems when adding password based capabilities to existing network security protocols. Now OEKE helps in fortification of TLS with password based key exchange cipher suites. TLS-OEKE is initiated by the Server. Server need not to know client name (it is mapped to a password by the server using local database) to compute and send the server's TLS Key-Exchange message.
But name is required to process the incoming client's TLS Key-Exchange message. So that is why engineers embodied client's name in the client's TLS Key Exchange message rather than embodying it in the client's TLS hello message. As per Fig 1.1 a fresh password is chosen and shared to capture the existing shared context. If this password is a long random

string, it can be used to setup security association, but less user friendly. Natural language phrases, more user friendly, however vulnerable to dictionary attacks. Need to derive a strong session key from a weak shared password.

Client	**Server**
pw	*pw*
accept ← false	accept ← false
terminate ← false	terminate ← false
accept ← true ---->	Auth?= H1(UkSkXkY kKS) if true, accept true skS H0(UkSkXkY kKS)

Fig: 1.1 An execution of the protocol OEKE under computational Diffie-Hellman.

It is run by the client U and the server S. The session key is sk = H0(UkSkXkY kY x) = H0(UkSkXkY kXy).

OEKE, is a simplified" variant of a AuthA mode of operation [2], and prove its security in the random oracle and the ideal-cipher models. At the core of this variant resides only one flow of the basic Diffie-Hellman key exchange encrypted under the password and two protocol entities holding the same password. It therefore slightly differs from the original EKE [1, 4] in the sense that only one flow is encrypted using the password; instead of the two as usually done. But then, it is clear that at least one authentication flow has to be sent. And it satisfies the security notions. AuthA protocol and its multiple modes of operation are secure under the computational Diffie-Hellman intractability assumption and help in fortification of transport layer security protocol.

3. DH-EKE

Diffie-Hellmen Encrypted Key Exchange (EKE), this method provides key exchange with mutual authentication based on weak secrets (e.g., passwords).
In DH-EKE a weak secret P is used to encrypt the elements of a Diffie-Hellman key exchange, i.e.,
g^x(mod P) and g^y(mod P).
Consider the closer data structures, which reveal that the ideal places to adjust TLS for new cipher suites are the messages of like `ServerKeyExchange` for Server and `ClientKeyExchange` for Clients. It is quite clear that for compatibility reasons we should not alter messages which are sent before an agreement on a cipher suite has been reached. This means in particular that modifying `ClientHello` should be refrained.

The key is cryptographically strong if x and y are cryptographically strong random numbers, regardless of the strength of the password. Various ways exist for optimizing the number of flows as well as the number of encryptions. Example that we have taken in the Section II of the paper with AuthA Key Exchange is elaborated here with DH-EKE. The client's is encrypted with the password instead of being accompanied by a signature and the swapping of client's and server's *Finished* messages while sending is done.
The first difference helps to authenticate each other based on the common knowledge of the password. The second change is due to the problems of transferring identity information and the subtle issues of dictionary attacks. Note that it is of paramount importance that the client does not use any key derived from the premaster secret *pms* before the client has successfully received and verified the server's *Finished* message.
Because there is no PKI in DH-EKE so the server's Certificate and Certificate-Request messages and the client's Certificate and CertificateVerify messages are omitted.
There are other protocols which are based on DH-EKE like SPEKE and SRP but are less preferred. First of all about Simple Password Encrypted Key Exchange (SPEKE) [9], the protocol is also based on a Diffie-Hellman key exchange but instead of encrypting the half-keys with the password it uses the password to derive a generator for a large prime-order subgroup. Now in Secure Remote Password Protocol (SRP) [10] , it seems the most efficient system which reduces also the risk when the server database is stolen it has similar problems with integration as SPEKE. The protocol cannot be started in flow 2 which means that the handshake would require an additional request response pair.

In addition to exponentiations in multiplicative groups we also need a shared-key encryption function Ep(z) to
transport the client's Diffie-Hellman half-key. As mentioned in Fig 2.1 the protocol flow processing in DH-EKE.

Client → **Server**

Client prepares the Client-Hello

Server → **Client**

Server Completes the serverDHEKEP field in Server Key Exchange with h^x and $h^{x'}$.Server HelloDone message is sent to client.

Client → **Server**

Client verifies the parameters of the group.

Server → **Client**

The server extracts the identity of the client from the ClientKeyExchange message and retrieves the client's password context.

Client → **Server**

Servers finished message is verified.

Server → **Client**

The server verifies the client's Finished message. If the verification fails, the server aborts, If the verification is OK, the 'potential online attack' counter is updated.

Fig: 2.1 Protocol Flow Processing in DH-EKE

We consider the additional costs of the additional exponentiations worthwhile but it would be straightforward to make the use of DH-EKE and allow performance critical environments to trade the risk of stolen server databases with improved performance.
Further there are many protocols based on collisionful hash. However, none of their feature could outweigh the simplicity of the integration of DH-EKE in TLS.

4. FORTIFICATION OF TRANSPORT LAYER SECURITY PROTOCOL

Secure password based authenticated key-exchange protocols can improve the situation and can be integrated into TLS in an efficient and non-intrusive manner. In this paper validation of the approach is done by integrating the cipher suite into a in-house toolkit providing the complete SSL protocol suite.

There are many password authentication schemes like RSA-based Password Authentication Schemes, ElGamal based Password Authentication Schemes and Hash-based Password Authentication Schemes. And there are many attacks which are protected by using one of these smart card password authentication schemes. These password authentication schemes secure transport layer from Denial of Service Attacks, Forgery Attacks (Impersonation Attacks), Forward Secrecy, Mutual Authentication, Parallel Session Attacks, Password Guessing Attacks, Replay Attacks, Smart Card Loss Attacks and Stolen-verifier Attacks.

If we are using AuthA Key Exchange Protocol so it is clear that a simple block-cipher can not be used in place of the ideal-cipher required by the security result. We indeed need permutations onto group for all the secret keys, otherwise partition attacks can be mounted [5].Measurements of the performance showed that our cipher suite compares well with other cipher suites.

DH-EKE outperformed comparable cipher suites providing mutual authentication and perfect forward secrecy by a factor of up to two (SSL DHE DSS WITH DES CBC SHA) and was only slightly slower than the commonly used cipher suite SSL RSA WITH RC4 128 SHA. A promising avenue is to also instantiate the encryption primitive as the product of a DiffieHellman value with a hash of the password, as suggested in AuthA [2].
Investigations have shown that this multiplicative function leads to a password-based key-exchange scheme secure in the random-oracle model [6]. Moreover same hash function could not be used everywhere in AuthA. Better security and performance is achieved using DH-EKE. So to some extent security of the transport layer protocol is managed using AuthA One-Encryption-Key-Exchange and DH-EKE (Diffie-Hellmen Encryption Key Exchange) which helps in the fortification of Transport Layer Security Protocol.
AuthA and DH-EKE both use one of the smart card password authentication scheme and secure TLS and further fortifies the TLS.

5. CONCLUSION

Explanation of AuthA and DH-EKE Key Exchange protocols is given which results in the fortification of the Transport Layer Security Protocol. There are number of risks associated with these protocols but serves the purpose of security. So in the paper analysis of two very efficient schemes on password-based authenticated key-exchange methods is done.

Evidences are generated to show that the AuthA and DH_EKE protocol and its multiple modes of operation are secure. Few other protocols which are based on DH-EKE like SPEKE and SRP are analyzed but these are less preferred because instead of encrypting the half-keys with the password it uses the password to derive a generator for a large prime-order subgroup. Now under the computational Diffie-Hellman intractability assumption AuthA and DH_EKE protocol are secure which helps in fortification of transport layer security protocol.

In general, there are three types of identity authentication tasks which are identity authentication for something known, such as a password, identity authentication for something possessed, such as a smart card and identity authentication for some personal characteristics, such as fingerprints. AuthA and DH-EKE are using only first two methods to identify a user.

In the future, fortification of TLS could be done by combining the three types, through which an ideal password authentication scheme could be generated. Moreover these work on single-server environment. However, since the scales of computer networks are becoming larger and larger, password authentication schemes which only support single-server environment will soon fall behind users' needs.

Therefore, need for multi-server architectures is there, where users can register at the register center only once and access resources from different servers efficiently. In the future, attempts would be made to develop an ideal password authentication scheme with a multi-server architecture with other Key Exchange protocols.

This ideal password authentication scheme would meet all the security requirements and would achieve all the goals. And further this ideal password authentication scheme in multi server architecture would help in fortification of Transport Layer Security Protocol.

6. REFERENCES

[1] M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated Key Exchange Secure Against Dictionary Attacks. In Eurocrypt '00, LNCS 1807, pages 139{155. Springer-Verlag, Berlin, 2000.

[2] M. Bellare and P. Rogaway. The AuthA Protocol for Password-Based Authenticated Key Exchange. Contributions to IEEE P1363. March 2000. Available from http://grouper.ieee.org/groups/1363/.

[3] M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing E_cient Protocols. In Proc. of the 1st CCS, pages 62{73. ACM Press, New York, 1993.

[4] S. M. Bellovin and M. Merritt. Encrypted Key Exchange: Password-Based Protocols Secure against Dictionary Attacks. In Proc. of the Symposium on Security and Privacy, pages 72{84. IEEE, 1992.

[5] C. Boyd, P. Montague, and K. Nguyen. Elliptic Curve Based Password Authenticated Key Exchange Protocols. In ACISP '01, LNCS 2119, pages 487{501. Springer-Verlag, Berlin, 2001.

[6] E. Bresson, O. Chevassut, and D. Pointcheval. Encrypted Key Exchange using Mask Generation Function. Work in progress.

[7] T. Berners-Lee, R. T. Fielding, H. F. Nielsen, J. Gettys,and J. Mogul. Hypertext transfer protocol – HTTP/1.1.Internet Request for Comment RFC 2068, Jan. 1997.

[8] W. Diffie and M. Hellman. New directions in cryptography.*IEEE Transactions on Information Theory*, IT-22(6):644–654, Nov. 1976.

[9] D. P. Jablon. Strong password-only authenticated keyexchange. *Computer Communication Review*, 26(5):5–26, Sep 1996.

[10] T. Wu. The secure remote password protocol. In *Symposiumon Network and Distributed Systems Security(NDSS '98)*, pages 97–111, San Diego, California, Mar.1998. Internet Society.