Network-Specific Attacks on Diffie-Hellman Key-Exchange in Commercial Protocols

Iraj Fathirad, John Devlin, and Sepidehsadat Atshani

Abstract—**An efficient and reliable key-establishment method is the most important building block of any secure cryptographic channels. Public-key cryptography was a revolution in cyber security key administration and enabled peers to dynamically create keys for each cryptographic session. The Diffie-Hellman (DH) algorithm is the first published public-key cryptosystem. DH and its variants are extensively investigated, standardized, and widely used in network security protocols. However, DH is vulnerable to some concerning mathematical, implementation-related and network-specific attacks. Defending against these attacks is important in secure implementation of DH in network protocols. This paper categorizes various attacks on DH scheme with focuses on attacks related to the DH integration in network protocols (referred as network-specific attacks). Furthermore, we comparatively review the approaches taken by commercial protocols to tackle network attacks and analyze the strength of these solutions.**

Index Terms—**Key-exchange, DH, ECDH, MiTM, DoS, reply attack, SSH, ZRTP, SSL/TLS, IPsec, IKEv2.**

I. Introduction

A secure and reliable key distribution system is the most important ingredient of constructing cryptographic channel between two or more peers. Key distribution may be accomplished in a number of ways:

- Manual delivery of key between peers
- Sending the key through existing cryptographic channel
- Taking Advantage of public-key cryptography schemes

The first two options have proven to be problematic. Physical delivery of keys through face-to-face meeting or trusted courier can be unsafe and impractical for large number of participants. The security of sending a new key through an existing secure channel depends on security of previously established channel and the previous key exchange. These problems have been addressed by introducing the public-key cryptography. Diffie-Hellman (DH) key exchange scheme was the first published public–key algorithm [1] and later standardized by IETF in [2]. This protocol is widely used in many important commercial protocols. However, DH is subject to number of mathematical and network attacks. Particularly, it is required to implements DH along with techniques to tackle network-specific attacks. This paper first reviews the DH key exchange scheme, then study some of the most important network attacks on this scheme, and finally analysis the suggested methods to counter these attacks in commercial protocols.

Manuscript received October 10, 2014; revised February 19, 2015.

Iraj Fathirad and John Devlin are with the Department of Electronic Engineering, La Trobe University, Victoria 3086, Australia (e-mail: I.fathirad@latrobe.edu.au, J.devlin@latrobe.edu.au).

Sepidehsadat Atshani is with Faculty of Business, Economics and Law, La Trobe University, Victoria 3086, Australia (e-mail: satshani@students.latrobe.edu.au).

II. Descryption of DH Key-Exchange

The DH key exchange protocol allows two peers to negotiate a secret over public communication medium. The security of DH relies on the difficulty of computing discrete logarithm over a large prime order cyclic group.

To negotiate key using DH, parties agree on a cyclic group G of prime order p and generator g. Then, user A selects a random $X_A \in G$ and computes $Y_A = g^{X_A} mod\ p$ and sends Y_A to B. similarly B calculates Y_B from X_B and sends it to A. Users A and B compute ${Y_B}^{X_A} mod\ p$ and ${Y_A}^{X_B} mod\ p$ to achieve the shared secret of $g^{X_A X_B} mod\ p$.

There exist a variant to the DH over elliptic curve cryptography [3] and defined as ECDH [3-5]. ECC use two families of curves: (1) prime curves defined with the cubic equation of $y^2 mod\ p = (x^3 + ax + b) mod\ p$ over Z_p, and (2) binary curves defined with the cubic equation of $y^2 + xy = x^3 + ax^2 + b$ over $GF(2^m)$. The security ECDH relies on intractability of EC discrete logarithm. Two parties first agree on a random elliptic curve with group of points $E_p(a, b)$ and a base point $G = (x_G, y_G)$ with a large prime order of n. Then, user A selects a random integer $X_A < n$ and computes $Y_A = X_A G$ and sends it to B. Similarly, B calculates Y_B from random X_B and sends it to A. User A and B calculates $X_A Y_B$ and user $X_B Y_A$ to achieve the shared secret key $K = X_A X_B$G . The ECDH requires a smaller key/parameters size and is more efficient at the same level of security [6].

III. DH Vulnrabilities

A. Attacks on Mathematical Structure of DH

These attacks affect mathematical design and computational texture of DH algorithm and include [7]:

- *Degenerate message attacks*: Cases that protocol is not effective (unity public keys lead to unity shared key).
- *Simple exponent attack*: If one of the private keys is simple enough to be determined and break the protocol.
- *Simple substitution attack*: In this attack, an adversary substitutes one of the public-keys with one “1” which leads to both parties computing the shared secret key equal with one.

- *Attacks based on Pohlig-Hellman algorithm*: This algorithm can solve the discreet logarithm problem of g^X if the prime factorization of generator's order contains small primes [8].
- *Attacks based on Composite order subgroups*: Subgroups without a large prime order can be exploited by an adversary and generally is applicable to primes of the form of ($p = Rq + 1$ $with\ small\ R$) [9].
- *Attacks based on Pollard-Lambda Algorithm*: This algorithm enables an adversary to compute the private exponent (X) from g^X, if X is known to lye within a certain intervals [10].
- *Attacks based on the number field sieve algorithms*: Inadequately chosen small groups and public/private exponents are vulnerable to be compromised.
- *Attacks on prime order subgroups*: This attack [11] computes part or all of the secret key using small order subgroups in Z_p.

B. Attacks on Imlementation Detail of DH

These attacks are concerned with the protocol implementation and include [7]:

- *Attack on context*: In this attack an adversary blocks or deletes previous messages using sequence numbers.
- *Obtaining ephemeral secret from memory*: If parties don't clear private secrets from memory, the secrets might be written to disk and extracted through unwanted access, or even be extracted from RAM.
- *Timing attack*: This attack [12] relies on the fact that the time taken for most of the modular exponentiations is dependent on the input, and an adversary might be able to precisely determine the computation time.

C. Network Attacks of DH Integration in Protocols

These attacks include common network attacks which are also applicable to DH implementation in network protocols for key-exchange purposes. The most important attacks on protocols using DH are as below:

- *Man-In-The-Middle (MITM) attack*: An opponent impersonates peer A to peer B and vice versa.
- *Denial-Of-Service (DoS)*: An opponent requests a high number of DH keys to clog the victim's system.
- *Reply attack*: attacker records and replays legitimate packets to break the connection or handshake session with out-of-order and irrelevant packets.

Attacks on implementation of DH and mathematical attacks linked to DH architecture are not limited to those issues reviewed here; Additional attacks and in depth analysis is beyond the scope of this paper. However, most if not all of these attacks can be prevented by proper implementation: using adequate and proven techniques in realization of the algorithm (e.g. using a source of randomness to defend against a timing-attack); being cautious about handling private secrets; and following specific recommendations in selecting the subgroups, public/private exponents, modulo and generator. However, the network attacks on protocols which use DH are source of serious concern. Properly implemented DH using secure groups and public/private parameters is can still be vulnerable to certain deadly attacks if the host protocol doesn't provide assurance against network attacks. These attacks are not linked to the DH algorithm directly, and more concern with the architecture of the host protocol which use DH algorithm.

Note: There exists another type of network-specific attacks which targets the physical layer of network (e.g. network connectivity) [13]-[16]. These attacks target the physical layer of network connection, and applicable to all higher level network-oriented protocols. Investigating the details of these attacks is out of scope of this paper.

IV. DH in Commercial Protocols

A. SSH

Developed by SSH communication security Ltd is application layer protocol for secure remote login, command execution, remote shell service and other secure network services. The SSH [17] protocol consists of three major components: (1) SSH transport layer protocol [18] which provide server authentication, confidentiality, integrity protection and optional compression; (2) SSH user authentication [19] which describes user-side authentication(to server); and (3) SSH connection protocol [20] which provides secure network service over SSH transport protocol and essentially is responsible to multiplex the multiple logical connection in secure authenticated connection(tunnel). SSH works over any 8-bit clean, binary-transparent transport with (recommended) transmission error protection, but usually is used over TCP. In SSH, the core protocol use DH algorithm for key generation; however, the RSA was later suggested as alternative to DH in [21]. Also, other algorithms based on ECC, specifically ECDH, ECMQV and ECDSA, were standardized for SSH in [22].

B. ZRTP

The ZRTP[23] was developed by Phil Zimmerman and defines a cryptographic key agreement scheme for VoIP (voice over internet protocol) applications which rely on real-time protocol (RTP) [24]. ZRTP enables two VoIP parties to agree on session key/parameters for establishing secure real-time transport protocol (SRTP) [25]. The SRTP (a profile of RTP), in turn provides RTP with authentication, confidentiality and reply protection. ZRTP also use two non-DH modes.

C. TLS

Netscape originated secure socket layer (SSL) [26] and its internet standard successor transport layer security (TLS) [27] are multi-purpose and real-time transport layer cryptographic protocol. TLS can supply connection-oriented and secure channel (tunnel) between peers and must run over reliable transport protocol such as TCP. Different application layer protocols running over reliable transport channel can use SSL/TLS as underlying security mechanism for their data segments. TLS may employs (1) DH or RSA; (2) hybrid asymmetric algorithms including ephemeral DH with RSA (DHE-RSA) or DSA (DHE-DSA); (3) pre-shared key authentication method [28] which consist of using pre-shared secret lonely or in combination with RSA (RSA-PSK) or ephemeral DH (DHE-PSK); (4) Secure remote password method (SRP) [29] authentication method which as side-effect generate a shared key based on DH; (5) using Kerberos[30]-based authentication [31] to securely deliver the client-generated pre-master session key to server.

D. DTLS

Datagram transport layer security (DTLS) [32] is a variant of TLS which can provide same level of security for unreliable datagram traffic. DTLS is based on stream-oriented TLS, designed to run over unreliable transport medium and can provide similar security to UDP, DCCP [33] (Datagram Congestion Control Protocol) and SCTP [34] (Stream Control Transmission Protocol). DTLS is very similar to TLS in cipher suits, key derivation and authentication and the main difference is the message profile and added mechanism for running TLS handshake over unreliable transfer medium.

E. IPsec

IPsec is peer-to-peer protocol that provide security standard for network layer traffic and supply cryptographically protection for any kind of transport layer packets in both IPv4 and IPv6 environments. IPsec consists of set of protocols in order to provide two security services: confidentiality, authenticity, anti-reply service, and connectionless integrity through ESP (Encapsulation Security Payload) [35] or message authenticity only through AH (authentication header) [36]. In IPsec, the mandatory-to-use algorithms for both ESP and AH modes are described in [37] and the keying materials and security parameters are established through IKEv2 (internet key exchange version 2) [38] protocol.

V. DH Network Attacks and Solutions

A. Man-in-the-Middle(MiMT) Attack

In this attack an opponent impersonate peer A while communicating with peer B and vice versa. As a result of this attack both peers calculate the shared-key with the adversary. The lack of authentication in DH is motivation for MiTM attack and protocols need to use a mechanism to assure the link between public value and peer's ID. The methods of commercial protocols to authenticate DH exchange can be classified into four main groups:

1) Certificate (DH parameters/public-key) —TLS/DTLS

The public DH public parameters and public key of server are sent to client in certificate. This method also requires client to provide authentication on its DH public key (e.g. digital signature on exchange or certificate) for mutual protection against MiTM attack. If client use the same method as server, then provided certificate must use the parameters (group/generator) suggested by the server.

2) Signature on exchange —TLS/DTLS/ IKEv2

In this method all the exchanged messages together with relevant peer IDs and nonces are signed with signature-capable certificate and sent to other peer together with the corresponding certificate. In TLS/DTLS client use the digital signature method to sign the exchange and in IPsec both parties sign the exchange, IDs and nonces.

3) Signature on exchange and public-key —SSH

In this method, the SSH server provide the mean of authentication by sending the hash of previously exchanged messages, server's DH public key, ID of peers together with server's host which is signed with server's private key. The signature together with public-key pair of signing key is sent to user. The authenticity of the signing key may be verified by either (1) certificate or (2) local database (cached keys in user side). This method is accompanied with user authentication to server to tackle MiTM attack. The user authentication does not authenticate the exchange, but can authenticate the user. The user authentication is carried out by:

1) User send raw public-key in a message with signature on that message with private key pair of that public key together with optional certificate which also contains the public-key used for authentication. The possession of a private key serves as authentication and sever verify the signature with certificate or local database.
2) User sends a message contains an encrypted password
3) Host-based authentication which the authentication is performed in client's host. In this method, server allows authentication based on the host that the user is coming from and the user name on the remote host.

If user authentication fails, the server allows user to retry the authentication and if user cannot prove its authenticity the server dismiss the negotiated key. In SSH server uses this method to sign its half of DH exchange.

4) Signature on parameters/public-key —TLS/DTLS

The server's public DH parameters and key are signed by a signature-capable certificate and sent to client together with the corresponding certificate. This method also requires client authentication on its DH public key for mutual MiTM protection. In TLS/DTLS this method is used by DHE-RSA and DHE-DSA cipher suits.

5) Pre-shared key —TLS/DTLS

A key derived by some out-of-band mechanism can be combined with un-authenticated shared DH key and result in authenticated shared secret. This authenticated shared-secret authenticates the shared DH secret and the exchange. The DHE-PSK in TLS use this method.

6) MAC with shared secret —IPsec (IKEv2)

The MAC code of mutually obtainable block of data is signed using padded pre-shared secret, where the secret is derived from a user chosen password or derived from negotiated shared DH secret, nonces and SPIs.

7) EAP methods (auth through third party server) — IKEv2

The Extensible Authentication Protocol (EAP) is an authentication framework which can provide host plug-in module for many other authentication methods. There are many flavours of EAP which are named after the accompanying protocol/technique (e.g. EAP-TLS is based on TLS protocol, EAP-PSK based on pre-shared key). EAP methods employ an authentication server to grant an access to a network service. While EAP techniques can be mutual in some flavours, but typically these methods are asymmetric (the peer does not authenticate the authenticator), and vulnerable to MiTM and consequently should be used in conjunction with other authentication methods for authenticator. It's also possible for the EAP server and peer to derive a mutually authenticated key for later use. In this case, the EAP methods with key derivation capability must be used.

8) Using cached shared secret or SAS —ZRTP

The ZRTP uses cached shared secret to authenticate peers to each other and when there is no such a secret both parties should use a SAS (short authentication string). The SAS is ideally displayed for human users and each party should verbally compare the SAS with other party. Each ZRTP endpoints maintains long-term cache of shared-secrets that has previously negotiated with other parties and linked to that party with ZID (peer's unique ID). During the DH key negotiation, peers first discover if they have any cached shared-secret in common and use it in session key calculation. Each party attach the series of mutually obtainable MAC codes to their public DH key. The key to the MACs are selected as bellow:

1) Cached secret (rs1) which derived from shared-secret of previous session
2) Cashed secret (rs2) which is old rs1 of previous session and derived from shared-secret of two session ago (previous previous session)
3) Trusted MiTM PBX shared secret (pbxsecret) if the other party is PBX
4) Out-of-band shared secret

If the authenticated DH shared secret negotiation success, then the cached shared secret (rs1) will be updated with new value and (rs2) will be replaced with old (rs1). The secret corresponding to matching hashes are kept and mismatch secrets are set to null. To authenticate the exchange, one of the initiator's (rs1) or (rs2) should match to responder's either (rs1) or (rs2); the first match (rs1 or rs2) with matched auxsecret/pbxsecret (where applicable) authenticates the key exchanges and together with DH shared key contribute to final shared key calculation. If each party detect the mismatch between both rs1/rs2 hashes and (or) auxsecret/pbxsecret (where applicable) then they may use SAS technique and if it's not possible to compare the SAS value, the session may be terminated. As extra protection, the initiator also sends a hash value over some information including its DH public value prior to DH key negotiation. The responder recalculates this hash value after DH negotiation and compares it to received hash to constrain the attacker only one guess to generate the correct SAS.

9) Summary

Summarizing the approaches of each protocol in authenticating DH key exchange and public-key/prams:

In SSH, the server side authentication is based on digital signature on DH exchange/public-key and relies on authenticity of signing public-key (server's host key). This key can be either sent in (1) raw format to be verified through local database (cached secrets) or (2) in certificate. The client also can (1) send its public-key in a message together with signature on that message by user's private key along with optional certificate, (2) use password-based or (3) host-based authentication. When using public-key, the authenticity of the key can be assured by presenting it in certificate or verifying the key with local database. IETF has defined OpenPGP as standard for SSH protocol, however, later on an IETF draft [39] suggested the method of using X509.

In TLS/DTLS, the server side authentication include (1) using public-key certificate to present CA signed DH public key/prams or (2) using digital signature on server's DH public key/prams with signature-capable certificate. The client also optionally authenticate itself by (1) presenting its CA signed DH public key/prams in certificate or (2) use digital signature on previously exchanged message, nonces and peer IDs via signature-capable certificate. Peers alternatively can commit an un-authenticated exchange and combine out-of-band shared key with shared DH key for mutual authentication. The supported public-key certificate in TLS/DTLS is X.509.

In IPsec(IKEv2), the exchanged messages together with peer IDs, nonces and SPI are called mutually obtained block of data and both peers mutually authenticate the shared DH key and exchange by: (1) providing the digital signature on mutually obtained block of data with signature-capable certificate, or (2) calculating MAC code of this data with shared-key which that shared key is (2.1) derived from shared DH key (each direction use different mutually obtainable shared key — This method just provides message authentication a.k.a integrity protection and don't authenticate the shared DH key), or (2.2) derived from user-chosen password or (2.3) is pre-shared key; the peers alternatively may use (3) EAP authentication methods. The IKv2 certificate types include X509, PGP, Kerberos tokens, SPKI and DNS signed key

In ZRTP, both peers use hash over mutually obtainable MAC codes which calculated with series of cached secrets to authenticate their DH public keys. These secrets include: (1) Two secrets which derived from two shared key of previously negotiated sessions which is compulsory to have at least one match, (2) One situation dependant secret if one party is PBX, and (3) One out-of-band obtained shared-key. In case of fatal mismatch, peers may try SAS technique.

B. Denial-of-Service(DoS) Attack

The very damaging DoS attack consists of clogging one peer with bogus requests with forged source IP addresses. Due to computationally intensive nature of modular exponentiation, the DH key exchange is highly vulnerable to clogging (DoS) attack.

Concerning with DH key negotiation, we can assume one of the peers as initiator and other one as responder. Initiator request DH negotiation and calculates its public-key after receiving the public-key of responder. Three different DoS attack is applicable to DH key negotiation:

1) The attacker can send huge amount of DH negotiation requests with forged source IP addresses to responder so that the victim is compelled to compute many modular exponentiations to carry out the public-key.
2) The attacker can use same way to send random number as other peer's public-key to both initiator or responder and waste their resources by computing wrong shared DH key.
3) The attacker can flood the initiator (victim) with DH group and public-key by sending fake DH request with forged source.

Obviously, all these three attacks triggered by lack a mechanism to distinguish between legitimate and bogus requests/messages. The protection against DoS attack in DH is carried out in one of following ways:

1) Using reliable transport medium —TLS and SSH

The negotiation can rely on connection-oriented transport level mechanism such as TCP to mitigate the DoS attack. However, The DoS attack may target the underplaying protocol itself. The TCP layer is a very common target of various DoS attack such as SYN flood etc.

2) Using stateless cookies —DTLS and IKEv2

The responder(server) can validate the legitimacy of initiator(client) through some additional message exchange prior to DH key negotiation by sending block of data (stateless cookie) and asking the initiator to retransmit the request message and include the cookie in the message. Server then verifies the valid cookie and proceeds with the handshake. This mechanism forces the attacker/client to be able to receive the cookie, which makes DoS attacks with spoofed IP addresses difficult but cannot prevent DoS attacks from valid IP addresses.

3) One-way hash chain —ZRTP

Parties use one-way hash chain, which is a series of successive hash images to be send as the part of successive message

C. Reply Attack

The very deadly Reply attack generally interpreted as an attack which adversary simply eavesdrop and capture a valid data packet and retransmit that later in order to replicate the transaction, break the connection, gain access to a service or cause a remote action. The reply attack on key negotiation is specifically implies an action which an adversary tries to break the key-negotiation or handshake by inserting previously exchanged legitimate message or fool one of the parties to negotiated previously used session key again. These attacks are also easy to defend by adding randoms (nonces) to each exchanged message of relevant key-calculation session and integrate these nonces in shared key computation.

In TLS and DTLS both client and server generate a random value consist of timestamp and opaque generated by a secure random number generator. These values serve as nonces and are used during key exchange and contributed to master-key generation to prevent replay attacks. Unlike connection-oriented TLS, in the connectionless DTLS the record layer employ retransmission timers and implicit sequence number with recommended sliding windows to provide transmission error and reply protection. In DTLS, the sequence number is assigned to each handshake message to ensure they are transmitted and received in a defined order.

In SSH, the random values are generated by both parties and exchanged prior to key negotiation. During the key negotiation the hash of messages containing the cookies together with DH public key of server is signed and sent along with the server's public-key to client for authentication purpose and this hash value also contribute to master session key generation.

IKEv2 runs over unreliable UDP and includes recovery procedure from transmission errors such as packet loss, packet replay and packet forgery. IKE use retransmission timers to prevent packet loss and every message contain a Message ID as part of its fixed header to match up requests and responses. All IKE messages carry IKE header including two unique SPIs of initiator and responder. To prevent the reply attack on key negotiation, both initiator and responder generate random nonces and send it to other party in their first exchange. These random values together with negotiated shared DH key contribute to first IKE SA (security association) master key which in turn is used again with nonces and SPIs for generating all other keys.

The ZRTP is designed to run over unreliable transport layer and include sequence number in the header of each message. The Sequence Number is a count that is incremented for each ZRTP packet sent and is initialized to a random value. This is useful in estimating ZRTP packet loss and also detecting when ZRTP packets arrive out of sequence. The ZRTP protocol detects transmission errors using Cyclic Redundancy Check (CRC). The one-way hash chain which is used to defend against DoS attack can also protect the exchange against reply attack.

VI. Discussion and Conclusion

This paper reviewed the approaches taken by network protocols to counter MiTM, DoS and Replay attacks. Concerning with MiTM attack, these protocol use a definition of authentication to defend against active eavesdropper in the following flavors:

1) What they authenticate (DH public-key prams or the exchange resulted in shared key).
2) Whom to trust for authentication (Certificate, cashed keys or password).

In authentication methods based on pre-shared secret, password, cached public-key keys and cached dynamic shared keys, the peers "themselves" are responsible for trustworthiness of authentication parameters (e.g. strength of keys/password/prams etc) and reliability of the process depends to some external factors. For instance, when using pre-shared key, the reliability of key depends to security of out-of-band delivery method; or when using password-based authentication, the reliance of method depends on strength of chosen password or how safe the user can keep it away from adversary; when using asymmetric authentication method with cached-stored public key (a.k.a digital signature), the assurance of method depends on how reliably the recipient obtained the cached public-key and how certain is about the authenticity of cached key. When using symmetric dynamic cached secret (e.g. ZRTP) the trustworthiness of method depends on how secret the peers can keep the cached key from adversary to avoid eavesdropping and tampering of keys and when the keys are derived from last negotiated session, the reliability of present keys depend on secrecy of previous session(s).

When using certificate-based authentication or EAP methods, the peers rely on third party(s) or enterprise for certifying the public key/prams or peers. Due to asymmetric mechanism of EAP methods, they may not be mutual and typically can be used to authenticate the initiator to responder and should be used with digital signature of responder to initiator. There exist two main types of certificates: X.509

and PGP. The X.509 relies on centralized trust model, or typical PKI (public key infrastructure) scheme. The PGP certificate is based on decentralized trust scheme, or “web-of-trust model” where the signature is created by the user itself or other users as endorsements. PGP is more popular in circle of acquaintances; and centralize scheme (x.509) is superior solution in large and broaden infrastructure, where high number of public-key can be generated. To defend against DoS attack, these protocols either

1) Rely on a connection-oriented transport medium (TLS, SSH)
2) Use un-reliable transport layer with special mechanisms such as hash chain (ZRTP) or stateless cookies (DTLS, IKEv2) to mitigate DoS Attacks

Concerning with anti-reply protection, all the mentioned protocol includes mean of random structure in exchanged messaged of a session and integrates that opaque in final master session key calculation.

ACKNOWLEDGMENT

The authors are thankful to the anonymous referees for their careful reading and valuable comments that have improved the quality of the paper

REFERENCES

[1] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on information Theory*, vol. 22, pp. 644-654, Nov. 1976.

[2] E. Rescorla, *RFC2631 (Diffie-Hellman Key Agreement Method)*, June 1999.

[3] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203-209, January 1987.

[4] E. Barker, D. Johnson, and M. Smid, "NIST SP 800-56A, Recommendation for pair-wise key establishment schemes using discrete logarithm cryptography," National Institue of Standard and Technology, NIST Special Publication, March 2007.

[5] C. Research, "Standards for efficient cryptography," *Elliptic Curve Cryptography (Version 1.9)*, August 22, 2008.

[6] S. Burnett. and S. Paine, *RSA Security’s Official Guide to Cryptography*, Osborne/McGraw-Hill, 2001.

[7] J. F. Raymond. and A. Stiglic, S*ecurity Issues in the Diffe-Hellman Key Agreement Protocol*, Zeroknowledge Systems Inc., 2000.

[8] S. Pohling and M. Hellman, "An improved algorithm for computing logarithms over GF(p) and its cryptographic significance," *IEEE Transactions on Information Theory*, vol. 24, pp. 106-110, Jan. 1978.

[9] P. C. V. Oorschot, "On diffe-hellman key agreement with short exponents," *Advances in Cryptology — Eurocrypt’ 96*, 1996, pp. 332-343.

[10] J. M. Pollard, "Methods for index computation (modp)," *Mathematics of Computation*, vol. 32, pp. 918-924, July 1978.

[11] P. J. Lee *et al*., "A key recovery attack on discrete log-based schemes using a prime order subgroup," *Advances in Cryptology —CRYPTO’ 97*, 1997, pp. 249-263.

[12] P. Kocher, "Cryptanalysis of Diffe-Hellman, RSA, DSS, and other cryptosystems using timing attacks," in *Proc. 15th Annual International Cryptology Conference on Advances in Cryptology*, Santa Barbara, California, USA, 1995, pp. 171-183.

[13] S. Iyer, T. Killingback, B. Sundaram, and Z. Wang, "Attack robustness and centrality of complex networks," *PloS One*, vol. 8, p. e59613, 2013.

[14] Y. Shang, "Vulnerability of networks: Fractional percolation on random graphs," *Physical Review E*, vol. 89, p. 012813, 2014.

[15] Y. Shang, "Robustness of scale-free networks under attack with tunable grey information," *EPL (Europhysics Letters)*, vol. 95, p. 28005, 2011.

[16] A. Srivastava, B. Mitra, N. Ganguly, and F. Peruani, "Correlations in complex networks under attack," *Physical Review E*, vol. 86, p. 036106, 2012.

[17] T. Ylonen, *RFC4251 (The Secure Shell (SSH) Protocol Architecture)*, Internet Engineering Task Force (IETF): Network Working Group, January 2006.

[18] T. Ylonen, *RFC 4253 (The Secure Shell (SSH) Transport Layer Protocol*, Internet Engineering Task Force (IETF): Network Working Group, January 2006.

[19] T. Ylonen, *RFC4252 (The Secure Shell (SSH) Authentication Protocol)*, Internet Engineering Task Force (IETF): Network Working Group, January 2006.

[20] T. Ylone, *RFC4254 (The Secure Shell (SSH) Connection Protocol)*, Internet Engineering Task Force (IETF): Network Working Group, January 2006.

[21] B. Harris, *RFC 4432 (RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol)*, Internet Engineering Task Force (IETF): Network Working Group, March 2006.

[22] D. Stebila and J. Green, *RFC 5656 (Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer)*, Internet Engineering Task Force (IETF): Network Working Group, December 2009.

[23] P. Zimmermann and J. Callas, *RFC 6189 (ZRTP: Media Path Key Agreement for Unicast Secure RTP)*, Internet Engineering Task Force (IETF), April 2011.

[24] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RFC 3550 (RTP: A Transport Protocol for Real-Time Applications*, Internet Engineering Task Force (IETF): Network Working Group, July 2003.

[25] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman, *RFC 3711 (The Secure Real-time Transport Protocol (SRTP))*, Internet Engineering Task Force (IETF): Network Working Group, March 2004.

[26] A. Freier, P. Karlton, and P. Kocher, *RFC6101 (The Secure Sockets Layer (SSL) Protocol Version 3.0)*, Internet Engineering Task Force (IETF), August 2011.

[27] T. Dierks and E. Rescorla, *RFC 5246 (The Transport Layer Security (TLS) Protocol Version 1.2)*, Internet Engineering Task Force (IETF): Network Working Group, August 2008.

[28] P. Eronen and H. Tschofenig, *RFC 4279 (Pre-Shared Key Ciphersuites for Transport Layer Security (TLS))*, Internet Engineering Task Force (IETF): Network Working Group, December 2005.

[29] D. Taylor, T. Wu, N. Mavrogiannopoulos, and T. Perrin, *RFC 5054 (Using the Secure Remote Password (SRP) Protocol for TLS Authentication)*, Internet Engineering Task Force (IETF): Network Working Group, November 2007.

[30] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, *RFC 4120 (The Kerberos Network Authentication Service (V5))*, Internet Enginering Task Force (IETF): Network Working Group, July 2005.

[31] A. Medvinsky and M. Hur, *RFC 2712 (Addition of Kerberos Cipher Suites to Transport Layer Security (TLS))*, Internet Engineering Task Force (IETF): Network Working Group, October 1999.

[32] E. Rescorla and N. Modadugu, *RFC 6347 (Datagram Transport Layer Security Version 1.2)*, Internet Engineering Task Force (IETF), January 2012.

[33] T. Phelan, *RFC 5238 (Datagram Transport Layer Security (DTLS) over the Datagram Congestion Control Protocol (DCCP))*, Internet Engineering Task Force: Network Working Group, May 2008.

[34] M. Tuexen and R. Seggelmann, *RFC 6083 (Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP))*, Internet Engineering Task Force (IETF), January 2011.

[35] S. Kent, *RFC 4303 (IP Encapsulating Security Payload (ESP))*, Internet Engineering Task Force (IETF): Network Working Group, December 2005.

[36] S. Kent, *RFC 4302 (IP Authentication Header)*, Internet Engineering Task Force (IETF): Network Working Group, December 2005.

[37] V. Manral, *RFC 4835 (Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH))*, Internet Engineering Task Force (IETF): Network Working Group, April 2007.

[38] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen, *RFC 5996 (Internet Key Exchange Protocol Version 2 (IKEv2))*, Internet Engineering Task Force (IETF), September 2010.

[39] O. Saarenmaa and J. Galbraith, *X.509 Authentication in SSH (Draft-saarenmaa-ssh-x509-00)*, Internet Engineering Task Force (IETF) - Network Working Group, February 7, 2007.

Iraj Fathirad received the B.E. degree from LIAU University, Lahidjan, Iran, in 2006, and the postgraduate diploma in electronic engineering from Latrobe University, Melbourne, Australia, in 2009, and the master degree in electronic engineering from Latrobe University, Melbourne, Australia in 2010. He is currently pursuing his Ph.D. degree with the Department of Electronic Engineering, Latrobe University, Melbourne, Australia. His research areas are elliptic curve cryptography, public-key hybrid encryption and group key-exchange schemes.

John Devlin received the B.E. degree in electrical engineering from Latrobe University, Melbourne, Australia, in 1974, and the Ph.D. degree in electronic engineering from Latrobe University, Melbourne, Australia in 1979. He is currently a professor and the head of the Department of Electronic Engineering, Latrobe University, Melbourne, Australia. His research lies in the area of communications, real time systems design, data storage, transmission and analysis techniques.

Sepidehsadat Atshani received her bachelor degree in human science from Parand Azad University, Tehran, Iran, in 2009. She is currently pursuing his master degree in information management and systems with the Faculty of Business, Economics and Law, Latrobe University, Melbourne, Australia. Her research areas are database management and data protection techniques.