Secure and Efficient Authentication Protocol for Power System Computer Networks

Celia Li, Helen Cheung and Cungang Yang
(Corresponding author: Cungang Yang)

Department of Electrical and Computer Engineering, Ryerson University[1]
350 Victoria St, Toronto, ON M5B 2K3, Canada
(Email: cungang@ee.ryerson.ca)

(Received Oct. 16, 2016; revised and accepted Feb. 21 & June 5, 2017)

Abstract

In this paper, we propose an efficient and secure Authentication Protocol for Power systems (APP). The security analysis shows that APP is secure and resilient to various kinds of attacks. The numerical analysis and simulation results shows that APP is more efficient than TLS, a public-key based authentication protocol recommended by IEC61850.

Keywords: Authentication; Network Security; Power System Computer Networks

1 Introduction

Authentication is essential in any service-oriented communication networks to identify and reject any unauthorized network access. An authentication protocol for the power systems must ensure full security to protect data integrity. In addition, the authentication protocol should meet the following requirements from the network perspective [22].

1) High efficiency. Efficiency is crucial to achieve the high availability requirement in real-time power system applications. The indication of high efficiency is two fold. First, the authentication schemes should not incur too much redundancy for security. Second, computation involved in authentication must be fast enough to meet timing requirements of messages in the power systems.

2) Resilient to attacks. Authentication schemes are required to resist malicious attacks, such as forgery attack, replay attack, and DoS attack. In addition, it should be a mutual authentication protocol. Mutual authentication is a two-way authentication process between a user and the authentication server.

The user ensures that he/she is not communicating with a malicious authentication server by authenticating the server. If this property is absent, a malicious authentication server may be able to mount a man-in-the-middle attack to gather private messages from the user. The authentication server also need to authenticate the client to ensure that he/she is communicating with a valid user. The authentication server ensures that he/she is not communicating with a malicious client by authenticating the client. If this property is absent, a malicious user is able to access the network without authentication. Many authentication protocols have been proposed for wireline [20, 27], *e.g.* the Internet, and other types of wireless networks [4, 7, 10, 11, 13, 19, 28, 29]. However, there are few authentication protocol designed for power systems.

Some authentication protocols have been proposed for smart grids [5, 15, 16, 18, 23], but most of them are for meter authentication. Due to the resource constraint of smart meters, the proposed protocols are lightweight (the protocols are based on symmetric key cryptography), which are efficient but not secure. Our authentication protocol is designed for employees of an power site to access sensitive operations or resource of the power system, which needs higher level of security. Our propose protocol is therefore based on public key cryptography. IEC61850 [9] is recently standardized for modern power substation automation by the International Electrotechnical Commission. IEC61850 recommends TLS [2, 11], a public-key based authentication protocol, to achieve secure communications. However, TLS has two weaknesses (1) not efficient; (2) key updates are vulnerable. Therefore, we proposed a new public key-based authentication protocol. Security analysis shows that our protocol is resilient to attacks. Performance analysis demonstrates that our protocol is more efficient than TLS.

2 Related Work

Many authentication protocol have been proposed for wired network, such as the Internet. Kerberos [27] is a computer network authentication protocol which works on the basis of 'tickets' to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner. Its designers aimed it primarily at a client-server model and it provides mutual authentication-both the user and the server verify each other's identity. Kerberos protocol messages are protected against eavesdropping and replay attacks. Kerberos builds on symmetric key cryptography and requires a trusted third party, and optionally may use public-key cryptography during certain phases of authentication.

The RSA Secure ID [20] employs hardware tokens to authenticate user. The hardware token stores secrets in a tamper-resistant module carried by the user. Here we refer to the simplest dedicated-hardware version, which has only a display and no buttons. Each instance of the device holds a secret "seed" known to the back-end. A cryptographically strong transform generates a new 6-digit code from this secret every 60 seconds. The current code is shown on the device's display. On enrollment, the user connects to the administrative back-end through a web interface, where he selects a PIN and where the pairing between username and token is confirmed. From then on, for authenticating, instead of username and password, the user shall type username and "passcode" (concatenation of a static 4-digit PIN and the dynamic 6-digit code). Many authentication protocols are also introduced for wireless networks. A Protocol for Carrying Authentication for Network Access (PANA) [9], enables authentication between clients and access networks in Wireless Local Area Networks. PANA runs between a client and a server in order to perform authentication and authorization for the network access service. PANA does not define any new authentication mechanisms, but performs authentication protocols of 802.11 standard.

In cellular networks, assume a client roams from a home network to a foreign network, the client needs to be authenticated by the foreign network. The foreign network must communicate with the client's home network via multi-hop communications to authenticate the client [6, 8, 11, 12]. The SIM card of a client and the authentication center of the client's home network are pre-installed with a shared secret key K. When the client roams to a foreign network, the foreign agent must communicate with the client's home network in order to obtain the shared key K, which will then be used to authenticate the client. In the handover authentication protocol of IEEE802.11i standard, after the authentication server successfully authenticates a mobile client, it will send a key called pairwise master key (PMK) to the AP associated with the client. The client will perform the same calculation as the AS to obtain the same PMK. The AP and client will use the PMK to derive a pairwise transient key (PTK) for encrypting future packets exchanged between them [10]. The AS then sends the PMK to the neighbors of the current AP, one by one. The PMK serves as proof of the client's successful login authentication performed by the AS. By letting the AS pre-distribute the PMK to the neighbors of the current AP, the client will not need to be authenticated by the AS when it moves to another AP.

Some authentication protocols for smart grids have been proposed. M. Fouda [5] proposed a light-weight and secure message authentication mechanism. The proposed mechanism is based on Diffie-Hellman key establishment protocol and hash-based message authentication code, which allows various smart meters at different points of the smart grids to make mutual authentication and achieve message authentication with low latency and few signal message exchanges. Li [15] proposed an efficient authentication scheme that employs the Merkle hash tree technique to secure smart gird communication. Specifically, the proposed protocol considers the smart meters with computation-constrained resources and puts the minimum computation overhead on them.

3 The Proposed Authentication Protocol

We describe in detail the proposed Authentication Protocol for Power systems (APP). Refer to Table 1 for the notation used in the remainder of the article. Our authentication protocol APP follows a key hierarchical structure similar to that in TLS [25]. That is, a pairwise master key (PMK) is created during the authentication process, and a master key (MK) and pairwise transient key (PTK). The two parties involved in the authentication will used the PTK for point-to-point communications. To minimize the latency of the authentication protocol, the proposed authentication protocol APP aims to minimize (1) the number of message exchanges between a user and the authentication server, thus minimizing communication cost the authentication latency; (2) the number of public key operations performed by the user and the authentication server, thus minimizing computation cost. Here, we assume that all users know the authentication server's public key.

Table 1: Notations

Notation	*Description*
C	Client
AS	Authentication server
Ix	ID of entity X
CA	Certificate authority
Px	Public key issued to X
Nx	A nonce generated by X
Sigx	Digital signature of entity X
Epubx(m)	Encryption of message m using X's public key
Dpubx(m)	Decryption of message m using X's public key
H(m)	Hash value of message m
Certx	Public key certificate of entity X
PMK	Pairwise Master key
MK	Master key
PTK	pairwise transient key

Following are the order of the messages to be exchanged in the protocol and explanation (see Figure 1):

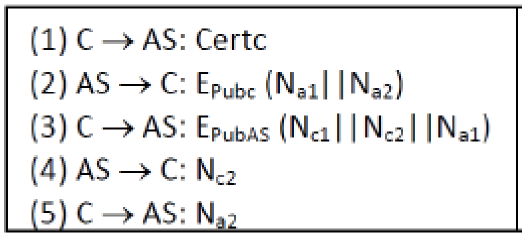

(1) C → AS: Certc
(2) AS → C: E_{Pubc} (N_{a1} || N_{a2})
(3) C → AS: E_{PubAS} (N_{c1} || N_{c2} || N_{a1})
(4) AS → C: N_{c2}
(5) C → AS: N_{a2}

Figure 1: The proposed authentication protocol

1) A client C sends AS a message which contains its public key certificate to inform the AS of its presence and public key. AS verifies the digital signature of the CA who issued the certificate using CA's public key (We assume that AS has the public key certificate of the CA.) AS also verifies other information in the certificate such as the ID of the certificate and the certificate expiry date.

2) If the above verifications are successful, AS extracts the client's public key from client C's certificate $Cert_c$ and generates a message which contains two nonces Na1 and Na2. AS then encrypts the message using C's public key and sends the encrypted message $E_{Pub_{AS}}$ $(Nc1||Nc2)$ to the client C where the operator || denotes a concatenation. Upon receiving the message, C decrypts it using its private key,

3) Client C retrieves AS's public key from AS's public key certificate $Cert_A$ (We assume that client C has the public key certificate of AS and generates two nonces Nc1 and Nc2. C then encrypts Nc1, Nc2 and Na1 using AS's public key, and sends the encrypted message to AS. AS will decrypt the message using its private key to Nc1 and Nc2. Both the client and the AS then calculate a pre-master key $PMK = Nc1||Na1$, where the operator || denotes a concatenation, and Nc1 and Na1 are the nonces generated in Steps 2 and 3 above. (The security of nonces Nc1 and Na1, and thus key PMK is ensured by AS's and client's public-private keys.)

4) AS then sends Nc2 to client C. Upon receiving this message, the client C has successfully authenticated AS, because only AS has the knowledge of Nc2.

5) To allow AS to authenticate C, C sends Na2 (generated by AS in Step 2) to AS. After AS receives Na2 correctly, it is considered to have successfully authenticated client C because only C has the knowledge of Na2.

The AS sends the pre-master key and the random numbers to the web browser, then the user and the browser both calculate the MK and PTKs. Although key generation is not part of this paper, it is worth noting that it is involved partially in the authentication protocol. Key management between a client and the browser allows them to derive a shared key to be used after the authentication for secure data exchanges. We follow a similar approach of key generation defined in TLS. That is, right after Step 3 of the authentication procedure, both parties compute a master key MK as follows:

$$MK = H(PMK, "mastersecret", Na1 + Nc1). \quad (1)$$

After the generation of MK, the two parties use the master key MK to compute a shared key called pairwise transient key (PTK). The PTK will be used to encrypt packets exchanged between the user and the browser.

4 Security Analysis

In this section, we describe the countermeasures implemented in APP against the attacks listed in [3] that are relevant to our protocol.

4.1 Forgery Attack

Forgery attack is an attack in which an attacker deliberately manipulate data. We prevent this type of attack by using digital signatures and message encryption. The public key certificate in the first message uses digital signature to prevent forgery attacks. The digital signature ensures that user C's certificate is protected against modifications and that counterfeit messages are infeasible to be fabricated. Any unauthorized changes to the content of the certificate will result in an incorrect signature value because the attacker does not know CA's private key.

The second and third messages use message encryption to prevent forgery attacks. The encrypted messages are protected against modifications. Any changes to the content of the messages will result in the messages unable to be decrypted successfully by the recipient.

4.2 Replay Attack

An attacker records messages of an ongoing authentication session and replays these messages in the future in an attempt to be successfully authenticated and possibly gain access to the network as a client. An attacker may replay a client's messages to gain access to the network, or an authentication server messages in order to impersonate the server. There are three approaches to resist replay attack. they are nonce, sequence number and time stamps. We prevent this type of attack by using nonces [26]. A nonce is a random number that only can be used for one time. A new message with nonces intended for a specific recipient must use newly generated nonces and not those previously sent to the recipient. If a message with nonces was lost or damaged and the message is retransmitted, the retransmitted message must use newly generated nonces.

We consider possible replay attacks on messages generated by APP described in Section 3.

1) Replaying User Messages.
In the authentication protocol, an attacker overhears and replays Message 1, 3 and 5 sent earlier by a client

C. After successfully receives Message 1 from the attacker, the AS replies with a Message 2. New nonces Na1 and Na2 are generated in the message. The attacker will not be able to decrypt Message 2 because he does not know the private key of AS. The attacker then replays Message 3 to the AS. The AS can detect that this is a replayed message because a new Message 3 is supposed to have new nonce Na1.

2) Replaying AS messages.
In the authentication protocol, an attacker overhears and replays Messages 2 and 4 sent earlier by a AS. The client sends Message 1 to the MAP. The attacker then responds with Message 2. User C sends Message 3 to the attacker. The New nonces Nc1, Nc2 and Na1 are generated in the message. The attacker will not be able to decrypt Message 3 because he does not know the private key of the user C. The attacker replays Message 4. The user C can detect that this is a replay attack because a new Message 4 is supposed to have new nonce Nc2. The attacker will then cannot be authenticated by the AS.

4.3 Denial-of-Service (DoS) Attack

In a DoS attack, a malicious attacker sends a flood of packets to a AS. The network resources are flooded or misused by an attacker to cease service to a legitimate user. In an authentication protocol, an attacker may send bogus messages or replay past valid messages repeatedly to force a AS to use up its resources on processing a large amount of these DoS attack messages. An attacker may repeatedly send copies of Message 1 to a AS. The AS will interpret the duplicates of this message as the losses of Message 2 it has sent. The AS will stop the authentication procedure after a pre-determined number of failed attempts to save resources. Note that this type of attack can happen to any protocol, and not specifically to authentication. An attacker may sniff valid Messages 3 and 5 from a successful authentication and replay the message repeatedly to the involved AS in order to overwhelm it. The AS can detect that this is a replayed attack because the new Messages 3 and 5 are supposed to have new nonces. If the AS receives the replayed message several times, it can infer that it is under a DoS attack and take appropriate actions to thwart the attack [1, 24, 14].

5 Performance Analysis

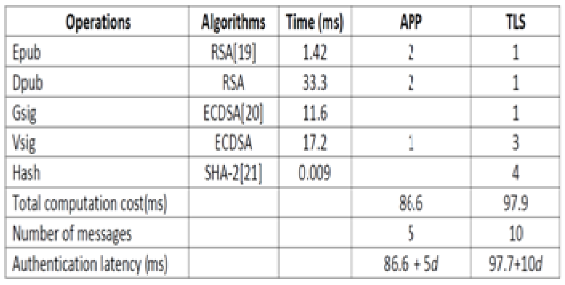
Power communication networks are used to ensure reliable, secure, and real-time message delivery. Hence, latency is much more important than the throughput in power systems, leading to delay-oriented design in power communication protocols. We compare the performance of APP with existing protocol using both numerical analysis and simulations. The protocols to be compared is TLS. TLS is a representative authentication protocol use in power systems recommended by IEC61850. Therefore we chose to compare our protocol with TLS.

5.1 Numerical Analysis

The numerical analysis demonstrates the theoretical gain of our proposed protocols over TLS scheme. The performance of the protocols is measured in terms of Communication costs: which indicate the number of messages exchanged between a AS and a user to complete an authentication session. Computation costs: which are the latencies (in milliseconds) incurred by the following security operations: encryption using public key (E_{pub}); decryption using public key (D_{pub}); generation of a digital signature (G_{sig}); verification of a digital signature (V_{sig}); and hashing. lists the above operations, the current state-of-the-art algorithms implementing the operations, and the computation time each of these algorithms incurs [17] (the first, second and third columns, respectively). The fourth, and fifth columns of Table 2 list the numbers of security operations APP and TLS perform, respectively. By multiplying the computation cost of each operation (from the third column) and the number of times it is executed, and summing up the costs of all operations executed by a protocol, we obtain its total computation cost as shown in the third last row of Table 2.

The computation cost of APP is less than that of TLS. The second last row of Table 2 lists the number of messages exchanged in each protocol. The authentication latencies shown in the last row are the sums of computation costs and communication delays, where d is the average delay of a one transmission incurred by a message. The delay of APP and TLS is $86.6 + 5dms$ vs. $97.7 + 10dms$. The gain of APP over TLS is due to a reduction in the number of messages exchanged, 5 vs. 10 and a reduction of public key operations of APP.

Table 2: Computation and communication costs



Operations	Algorithms	Time (ms)	APP	TLS
Epub	RSA[19]	1.42	2	1
Dpub	RSA	33.3	2	1
Gsig	ECDSA[20]	11.6		1
Vsig	ECDSA	17.2	1	3
Hash	SHA-2[21]	0.009		4
Total computation cost(ms)			86.6	97.9
Number of messages			5	10
Authentication latency (ms)			86.6 + 5d	97.7+10d

5.2 Simulation Results

We use QualNet (version 5.2), a commercial software that provides scalable simulations of wireless networks [21], for our experiments. The simulation paprameters for all experiments are illustrated in Table 3. The performance metric is authentication delay (latency), which is measured as the time between a client's transmission of an authentication request to an AS and the receipt of an acceptance confirmation. We conduct two sets of experiments to measure the authentication latency as a function of Number of users: We measure the average authentication latency of the proposed protocol and TLS. We measure the average latency by varying group size from 10 to 60. For each data point in a graph, we ran an experiment 10 times using 10 different random seeds and obtained the average rekeying latency. We also keep track of the maximum authentication delay, the maximum value among all users. Background traffic load: We measure the average authentication latency of APP and TLS in the presence of background traffic. We conducted four sets of experiments:

1) We measured the average authentication latency of APP and TLS as a function of number of users. The 400m x 400m network has one node as AS placed in the center of the square. The number of users varied from 10 to 60.

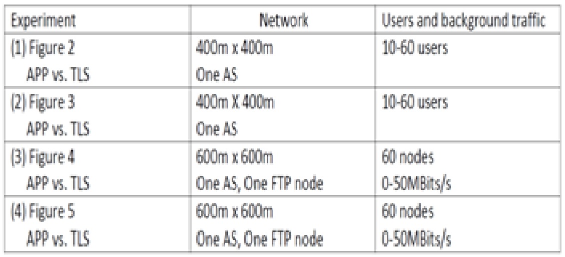
2) We measured the maximum authentication latency of APP and TLS as a function of number of users. We used the same network as in Experiment (1).

3) We measure the average authentication latency of the protocols in the presence of background traffic. The 600m x 600m network has one node as AS placed in the center of the square. We design an additional node as a source to transmit the background traffic of FTP to the AS. This node does not count as a user. The number of users is 60. We vary the data rate of FTP from 0 to 50 Mbits/s in our simulations.

4) We measured the maximum authentication latency of APP and TLS as a function of background traffic. We used the same network as in Experiment (3).

Table 3 summarizes the important parameters and lists the figures containing the graphs of the experiments. In all the experiments, the user nodes were randomly distributed in the networks. To test the scalability of the protocols, we let all users present in the network send authentication requests to the AS simultaneously.

Table 3: Simulation parameters for different experiments



Experiment	Network	Users and background traffic
(1) Figure 2 APP vs. TLS	400m x 400m One AS	10-60 users
(2) Figure 3 APP vs. TLS	400m X 400m One AS	10-60 users
(3) Figure 4 APP vs. TLS	600m x 600m One AS, One FTP node	60 nodes 0-50MBits/s
(4) Figure 5 APP vs. TLS	600m x 600m One AS, One FTP node	60 nodes 0-50MBits/s

5.3 Result Analysis

The results of the above four sets of experiments are illustrated by the graphs in Figures 2, 3, 4 and 5. (1) Figure 2 shows the average authentication latency of the APP vs. TLS under the function of number of users. When there are only 10 users in the network, the average latency of APP and TLS are 167.6ms and 227.2ms respectively. Given more than 10 users, the workload and channel contention at the server further increases. In these cases, the AAP offers lower average latency than TLS, because the APP requires less messages exchanged than EAP-TLS (5 vs. 10, as shown in the second last row of Table 3).

As the number of users increases, the average authentication latency of both APP and TLS increases as well. In the case of 60 users, the average authentication delay of the APP and TLS are 220.1ms and 291.5ms, respectively. The average authentication delay of APP is 24.1% lower than that of TLS. (2) Figure 3 also shows the maximum authentication delay of both protocols. Given 60 users request authentication with the same AS, the maximum authentication delay of APP and TLS are 299.7 ms and 381.6 ms, respectively. The amounts of cryptographic computation performed by LAP and EAP-TLS are very similar (86.6ms vs. 97.7ms as shown in the last row of Table 3). This shows that the gain of APP over TLS is mainly due to their difference on communication costs. (3) We examine how background traffic may affect the average authentication latency and maximum authentication latency if 60 users request to be authenticated at the same time.

Figure 4 shows average authentication latency as function of data rate, which is varied from 10Mbits/s to 50Mbits/s. Data rate is 0 means that there is no background traffic. As the data rate increases, the average authentication latency of users is enlarged. Higher data rate implies more background traffic to be processed by the AS, and more channel contention around the AS, resulting in longer delay. (4) Figure 5 also shows the maximum authentication latency of 60 clients. The data rates varies from 10m/s to 50m/s. As the data rate increases, the maximum authenticate latency of APP and TLS are enlarged. Higher data rate implies more background traffic to be processed by the AS, and more channel contention around the AS, resulting in longer delay.

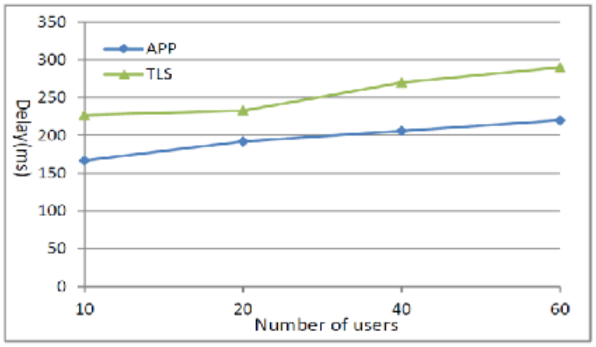


Figure 2: Average latency of APP via TLS - Function of number of users

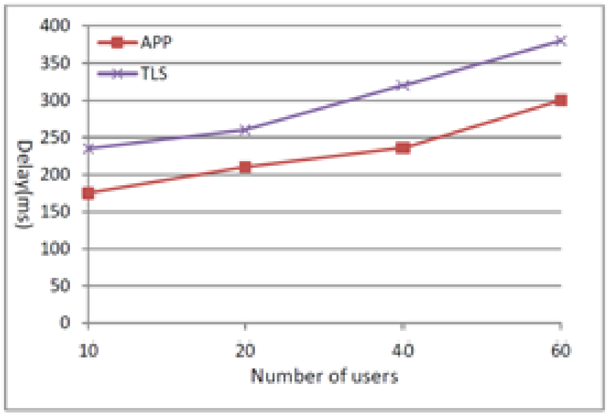


Figure 3: Maximum latency of APP via TLS - Function of number of users

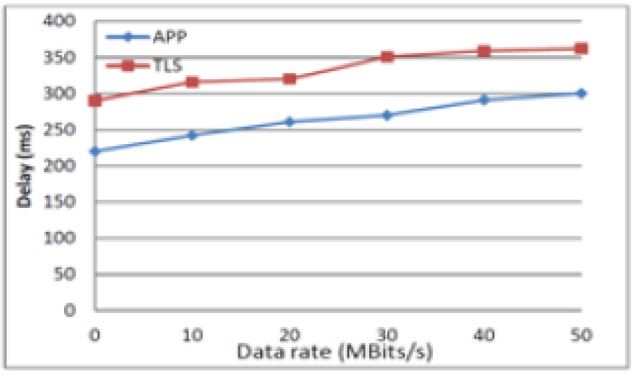


Figure 4: Average latency of APP via TLS - Function of traffic load

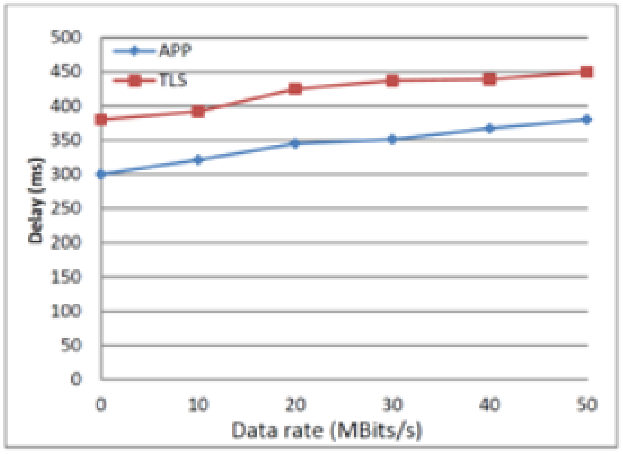


Figure 5: Maximum latency of APP via TLS - Function of traffic load

6 Conclusion

Cyber security in the power systems is a new area of research that has attracted rapidly growing attention in the government, industry and academia. Cyber security is still under development in the power systems, especially because information security must be taken into account with electrical power systems. In this paper, we presented a fast and secure authentication protocol for power systems. The security analysis shows that APP is secure and resilient to various kinds of attacks. The numerical analysis and simulation results shows that APP is more efficient than TLS.

References

[1] T. Aura, P. Nikander, and J. Leiwo, "Dos-resistant authentication with client puzzles," in *The 8th International Workshop Security Protocols*, LNCS 2133, pp. 170–177, Springer, 2000.

[2] K. Bhargavan, C. Fournet, M. Kohlweiss, A. Pironti, P. Strub, and S. Z. B eguelin, "Proving the TLS handshake secure," in *Advances in Cryptology (CRYPTO'14)*, pp. 235-255, Springer, 2014.

[3] FIPS, *Entity Authentication Using Public Key Cryptography*, FIPS Standard PUB 196, 1997.

[4] D. Forsberg, Y. Ohba, B. Patil, H. Tschofenig, *Protocol for Carrying Authentication and Network Access (PANA)*, Technical Report, RFC 5191, 2008.

[5] M. Fouda, Z. Md. Fadlullah, N. Kato, R. Lu, and X. Shen, "Towards a light-weight message authentication mechanism tailored for smart grid communications," in *IEEE Conference on Computer Communications Workshops*, Apr. 2011.

[6] P. Guo, J. Wang, B. Li, S. Lee, "A variable threshold-value authentication architecture for wireless mesh networks," *Journal of Internet Technology*, vol. 15, no. 6, pp. 929–936, 2014.

[7] D. He, N. Kumar, and N. Chilamkurti, "A secure temporal-credential-based mutual authentication and key agreement scheme with pseudo identity for wireless sensor networks," *Information Sciences*, vol. 321, pp. 263–277, Nov. 2015.

[8] D. He, S. Zeadally, "Authentication protocol for ambient assisted living system," *IEEE Communications Magazine*, vol. 35, no. 1, pp. 71–77, 2015.

[9] IEC, *Communication Networks and Systems in Substations*, IEC Standard, IEC61850, July 2003.

[10] IEEE, *Part11: Wireless Medium Access Control (MAC) and Physical Layer specfcations: Medium Access Control (MAC) Security Enhancement*, IEEE Standard 802.11i, 2003.

[11] Y. Jiang, C. Lin, X. Shen and M. Shi, "Mutual authentication and key exchange protocols for roaming services in wireless mobile networks," *IEEE Transactionon Wireless Communications*, vol. 5, no. 9, pp. 2569-2577, 2006.

[12] W. I. Khedr, M. I. Abdalla, A. A. Elsheikh, "Enhanced inter-access service network handover authentication scheme for IEEE 802.16 m network," *IET Information Security*, vol. 9, no. 6, pp. 334–343, 2015.

[13] S. Kumari, M. K. Khan, and M. Atiquzzaman, "User authentication schemes for wireless sensor networks: A review," *Ad Hoc Networks*, vol. 27, pp. 159-194, Apr. 2015.

[14] J. Lemon, "Resisting SYN flood DoS attacks with a SYN cache," in *Proceedings of the BSD Conference (BSDCON'02)*, pp. 10, 2002.

[15] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen, "An efficient merkle-tree-based authentication scheme for smart grid," *IEEE Systems Journal*, vol. 8, no. 2, pp. 655–663, 2014.

[16] H. Li, *et al.*, "EPPDR: An efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 20532064, Aug. 2014.

[17] M. Long, "Energy-efficient and intrusion resilient authentication for ubiquitous access to factory floor information," *IEEE Transaction on Industrial Informatics*, vol. 2, no. 1, pp. 40–47, 2006.

[18] K. Mahmood, S. A. Chaudhry, H. Naqvi, T. Shon, and H. F. Ahmad, "A lightweight message authentication scheme for smart grid communications in power sector," *Computers Electrical and Engineering*, vol. 52, pp. 114-124, 2016.

[19] T. Nguyen, M. Laurent, and N. Oualha, "Survey on secure communication protocols for the internet of things," *Ad Hoc Networks*, vol. 32, pp. 17-31, Sep. 2015.

[20] RSA, "Two-factor Authentication: RSA SecurID Software Token", 2011. (`https://www.rsa.com/en-us/products/rsa-securid-suite/rsa-securid-access/securid-software-tokens`)

[21] Scalable Networks, *QualNet Simulator*, Dec. 17, 2017. (`http://www.scalablenetworks.com/`)

[22] The Smart Grid Interoperability Panel Cyber Security Working Group, *Guidelines for Smart Grid Cyber Security*, Technical Report, NISTIR 7628, Aug. 2010.

[23] J. L. Tsai and N. W. Lo, "Secure anonymous key distribution scheme for smart grid," *IEEE Transactions on Smart Grid*, vol. 7, no. 2, pp. 906914, 2016.

[24] X. Wang and M. K. Reiter, "Defending against denial-of-service attacks with puzzle auctions (extended abstract)," in *IEEE Symposium on Security and Privacy*, 2003.

[25] Wikipedia, *Transport Layer Security*, DEc. 17, 2017. (`https://en.wikipedia.org/wiki/Transport\_Layer\_Security`)

[26] Wikipedia, *Cryptographic Monce*, Dec. 17, 2017. (`ttp://en.wikipedia.org/wiki/Cryptographic\_nonce`)

[27] Wikipedia, *Kerberos (Protocol)*, Dec. 17, 2017. (`http//:en.wikipedia.org/wiki/Kerberos\_(protocol)`)

[28] H. Xiong, "Cost-effective scalable and anonymous certificateless remote authentication protocol," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2327-2339, Dec. 2014.

[29] H. Xiong and Z. Qin, "Revocable and scalable certificateless remote authentication protocol with anonymity for wireless body area networks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 7, pp. 1442-1455, July 2015.

Appendix: TLS Authentication Protocol

The following full example shows a client being authenticated in addition to the server like above via TLS using certificate exchanged between both peers (see Figure 6).

1) Negotiation Phase: A client sends a ClientHello message specifying the highest TLS protocol version it supports, a random number, a list of suggested cipher suites and compression methods.

   The server responds with a Server Hello message, containing the chosen protocol version, a random number, cipher suite and compression method from the choices offered by the client. The server may also send a session id as part of the message to perform a resumed handshake.

   The server sends its Certificate message (depending on the selected cipher suite, this may be omitted by the server).

   The server sends its Server Key Exchange message (depending on the selected cipher suite, this may be omitted by the server). This message is sent for all DHE and DH anon cipher suites.

   The server requests a certificate from the client, so that the connection can be mutually authenticated, using a Certificate Request message.

   The server sends a Server Hello Done message, indicating it is done with handshake negotiation.

   The client responds with a Certificate message, which contains the client's certificate.

   The client sends a Client Key Exchange message, which may contain a Pre Master Secret, public key, or nothing. (Again, this depends on the selected cipher.) This Pre Master Secret is encrypted using the public key of the server certificate.

   The client sends a Certificate Verify message, which is a signature over the previous handshake messages using the client's certificate's private key. This signature can be verified by using the client's certificate's public key. This lets the server know that the client has access to the private key of the certificate and thus owns the certificate.

The client and server then use the random numbers and Pre-Master Secret to compute a common secret, called the "master secret". All other key data for this connection is derived from this master secret (and the client- and server-generated random values), which is passed through a carefully designed pseudo random function.

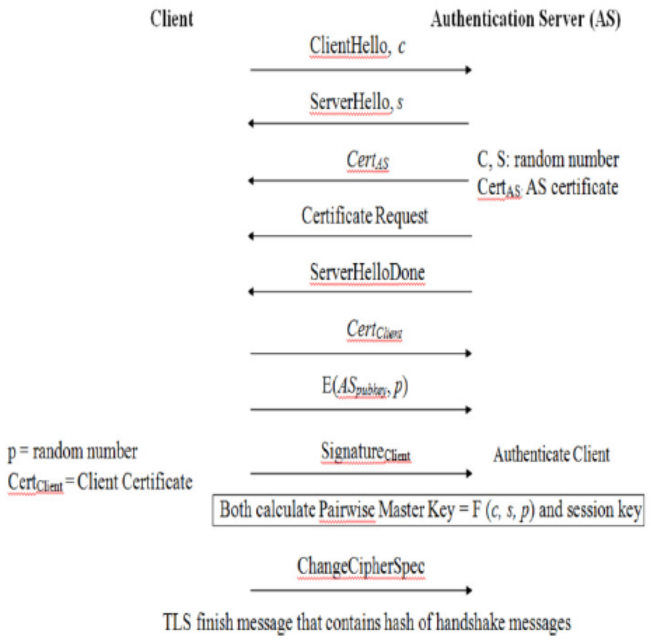


Figure 6: TLS Authentication Protocol

2) The client now sends a Change CipherSpec record, essentially telling the server, "Everything I tell you from now on will be authenticated (and encrypted if encryption was negotiated). Finally, the client sends an encrypted Finished message, containing a hash and MAC over the previous handshake messages. The server will attempt to decrypt the client's Finished message and verify the hash and MAC. If the decryption or verification fails, the handshake is considered to have failed and the connection should be torn down.

3) Finally, the server sends a Change CipherSpec, telling the client, Everything I tell you from now on will be authenticated (and encrypted if encryption was negotiated). The server sends its own encrypted Finished message. The client performs the same decryption and verification. At this point, the "handshake" is complete

Biography

Celia Li completed her Ph.D degree in electrical engineering and computer science department in 2015 at York University. Her research is focused on security and privacy, role-based access control and wireless mesh network security.

Helen Cheung completed her Ph.D student in electrical and computer engineering department at Ryerson University. Her research interest is on role-based access control, privacy and power system computer network security.

Cungang Yang completed his Ph.D degree in computer science in 2003 at University of Regina, Canada. In 2003, he joined the Ryerson University as an assistant professor in the Department of Electrical and Computer Engineering. His research areas include security and privacy, enhanced role-based access control model, information flow control, web security and secure wireless networks.