An Architectural Framework for Data Link Layer Security with Security Inter-layering

Hayriye Altunbasak
Scientific Atlanta (A Cisco Company)
5030 Sugarloaf Parkway
Lawrenceville, GA 30044–2869

Henry Owen
School of Electrical and Computer Eng.
Georgia Institute of Technology
Atlanta, Georgia 30332–0250

Abstract

Security issues in the data link layer have not received much attention while network security problems have been studied and addressed in the application, transport, and network layers. In this paper, we propose a new security inter-layering approach to secure the data link layer in Internet Protocol (IP) over Ethernet networks. In the data link layer, we propose to utilize secure namespaces instead of Media Access Control (MAC) addresses to identify network devices, which also provides the means to bind the data link layer and other layers securely. Moreover, we present the network structure to provide link-to-link security and the key establishment protocol to generate security parameters in the data link layer.

1. Introduction

Recently, security issues in the data link layer of local area networks (LANs) have started to receive long overdue attention in standards groups and in the literature [1–5]. While the IEEE 802.11i standard [6] greatly improves the security in wireless networks, wired networks have been left far behind in the security area with a false sense of security. In wireless local area networks (WLANs), the main source of security risks is the wireless technology's underlying communications medium, specifically airwaves. Nonetheless, WLANs inherit the vulnerabilities that exist in wired networks as well [7]. For instance, the loss of data confidentiality, integrity, and origin authenticity, and the threat of denial of service (DoS) attacks exist in both wired and wireless networks. Security issues in wired LANs need to be addressed to improve overall security in both networks.

In LANs, security weaknesses in the data link layer enable internal attacks. Though switches and routers have some security features built in, they are not enough to fully ensure the security of local networks. Moreover, these features require network administrators' involvement and are prone to misconfiguration. In addition, data link layer protocols used in LANs are not designed with built-in security features. The commonly known attacks in LANs, such as Address Resolution Protocol (ARP) poisoning, Media Access Control (MAC) flooding, port stealing, data link layer-based broadcasting and DoS, and MAC cloning attacks, exploit insecure protocols and the addressing structure in the data link layer [4].

As a response to security issues in local or metropolitan area networks, the IEEE 802.1AE MAC Security Task Group has been formed [2]. The IEEE 802.1AE Standard for Local and Metropolitan Area Networks (LAN/MANs): MAC Security specifies how all or a part of a network can be secured transparently to peer protocol entities that use the MAC Service provided by IEEE 802 LANs to communicate [1]. The standard defines MAC security (MACsec) entities in end stations that provide connectionless user data confidentiality, frame data integrity, and data origin authenticity utilizing the IEEE 802.1X standard. However, MACsec does not specify how the relationships between MACsec protocol peers are discovered and authenticated, as supported by key management or key distribution protocols. In this paper, we propose a new data link layer security architecture with a key establishment protocol that may be incorporated into MACsec.

In LANs, we observe that several security flaws are caused by the insecure addressing in the data link layer and the weak link between the network and data link layers [5]. First, the MAC address namespace of the data link layer is not adequate to provide secure services in local networks. MAC addresses are utilized to uniquely identify hosts/machines in the data link layer. While the MAC address of each network interface card is supposed to be globally unique, it can easily be changed. Second, the Internet Protocol (IP) and MAC addresses are not bound securely. IP addresses identify hosts in the network layer. Mappings between IP and MAC addresses in Ethernet-based LANs are accomplished by ARP [8]. However, ARP is not a secure protocol. Third, a compromise in the data link layer may not be detected by upper layers where most security implementations exist. Inter-networking reference models are composed of layers. In a layered model, each layer offers security services independent of other layers. Unfortunately, layers lack the ability to inform other layers whether any security measures are utilized or security weaknesses exist.

In this paper, we examine the data link layer security in IP over Ethernet networks. We propose to utilize secure namespaces instead of MAC addresses to identify network devices in the data link layer. In addition, we introduce a new security inter-layering concept to provide security services in the data link layer. In the next section, we

1-4244-1029-0/07/$25.00 ©2007 IEEE.

further discuss the security inter-layering concept. This is followed by the description of the proposed data link layer security architecture. Finally, we discuss the key establishment protocol further and present concluding remarks.

2. Security Inter-layering

The current implementations of security protocols in various layers provide a modular approach to security. Since each layer offers security services independent of other layers, security in one layer may provide a sufficient level of assurance against security weaknesses present in other layers. However, this approach also generates a computational overhead and increases the bandwidth usage in networks. The redundant use of security measures in various layers may be prevented if layers are informed regarding the security implementations in other layers. In fact, some information is available to lower layers in IP headers when IP Security (IPSec) is utilized [9], [10]. On the other hand, at lower layers, it is difficult to keep track of the security associations of the transport layer or upper layers as security associations may have states and/or detailed message/segment/data analysis may be required. In addition, security protocols in various layers rarely interact with each other to consider the security requirements and possible exploitations introduced by other protocols. For instance, in a recent IPSec vulnerability, an attacker modifies sections of an IPSec packet to cause a network host to generate an error message. When this error message is relayed via the Internet Control Message Protocol (ICMP), because of the design of ICMP, the message directly reveals segments of the header and payload of the inner datagram in cleartext. Consequently, an attacker intercepting the ICMP messages can retrieve the plaintext data [11]. Moreover, the layers of reference models change dynamically possibly rendering fixes introduced at present to be insufficient for future architectures and protocols. We believe that a more capable method is required to create a comprehensive and flexible security control mechanism. We propose a new security inter-layering concept to inform each layer regarding security protocols and features utilized in other layers.

The security inter-layering concept also allows the usage of the same namespaces in various layers in networks. For instance, a lower layer may choose to utilize a different secure namespace each time depending on the applications, user parameters, or network settings. Security inter-layering may be utilized to create secure bindings among namespaces and to protect against misbindings as well. Furthermore, security focus in each layer may/should be different and dependent on the functionalities of layers. While confidentiality may be important at the upper layers, the focus may be anonymity issues or the authentication of a source at lower layers. Finally, this concept can easily adapt to future architectures or namespaces since it is not a specific security architecture limited to a certain layer or a network architecture.

3. Data Link Layer Security

Security in LANs can be accomplished with a secure data link layer architecture. An essential security requirement for secure LANs is that network devices should allow data traffic from and to authorized hosts only. To accomplish this, network devices should be able to verify the integrity of messages and the origin of data traffic at the data link layer. On the other hand, network devices may allow insecure communications at the data link layer while establishing security parameters. Since denial-of-service (DoS) and man-in-the-middle (MITM) attacks are serious threats in local networks, this architecture should be resistant to these attacks as well. A secure data link layer architecture with these properties may be realized utilizing secure identities.

We propose the use of cryptographic identities utilized by other layers to create security parameters for the data link layer communications. We argue that the data link layer may use a secure namespace from other layers instead of MAC addresses, thus avoiding the overhead that a new secure namespace for the data link layer will create. This also prevents the risk of introducing possible weaknesses with a new namespace. We consider this an inter-layering of security related information.

We utilize a public-private key pair used at upper layers to generate identities and security parameters/keys at the data link layer. Note that, in the next sections, we use the terms hosts, machines, or users interchangeably referring to end points in a local network.

3.1. Data Link Layer Identities and Identifiers

In the data link layer, we propose to utilize public keys as identities. Here, we focus on a data link layer security architecture and assume that the data link layer has access to the public keys of upper layer. Since public keys are generally too long to include in each frame and they may have different sizes, we propose to use the hashes of public keys as data link layer identifiers. We use the term “identifier” for a hash value since the hash of a public key is a representation of the real identity. We also propose to utilize fixed size hash values as identifiers to provide flexibility in data link layer identities. In this manner, changing the type of identity or the identity itself will not affect how identifiers are used in LANs. It may also be desirable to hide real identities from passive attackers by using dynamic identifiers. For instance, each time a host connects to a LAN, it may choose to generate and negotiate a different identifier without changing its public key. This can be achieved by using pseudo-random values as additional inputs to the hash functions.

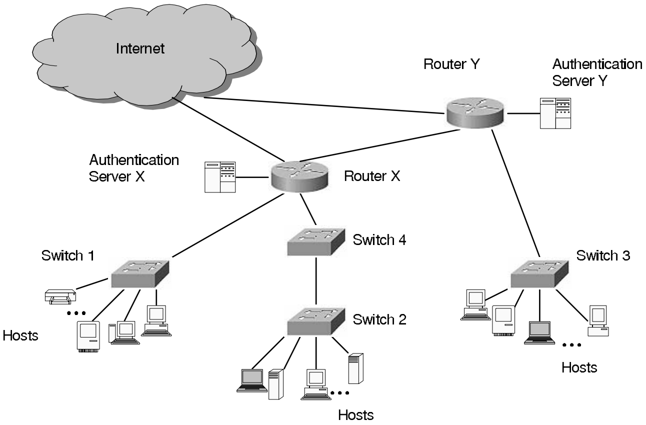

Figure 1. An illustration of the network architecture.

We propose to utilize hash values of 48 bits (instead of MAC addresses) as local data link layer identifiers, L2IDs, which are locally unique, and hash values of 120 bits for global end point identifiers, G2IDs, which are globally unique. First, a host computes a long hash value of its public key and generates its G2ID. Then, the same host computes a shorter hash value, its L2ID, using the G2ID and pseudo-random numbers (PRNs). Short hash values are utilized for L2IDs since they are incorporated in each frame. Longer hash values are used for G2IDs as they are long lived and globally unique. G2IDs are registered and stored in databases, allowing backtracking afterwards.

We compute global identifiers for end hosts in the proposed data link layer security architecture with an algorithm similar to the keyed hash identifiers (KHI) method described in [12]. Inputs to our algorithm and the KHI method are the public key information of an end host (*bitstring*) and a *context identifier* (*context ID*) as in the KHI algorithm. The *bitstring* is a presentation of the public key information (identity) of an end host, while the *context ID* is a randomly generated value defining the expected usage context of the particular global identifier. *context IDs* allow the utilization of the same public key (*bitstring*) to generate different global identifiers for different usage contexts, mechanisms, or protocols.

We generate our data link layer global identifiers using the algorithm below:

$$G2ID = context\ ID \mid extract_{120}\ (\ SHA1\ (\ expand\ (\ (\ context\ ID\ ) \mid (\ bitstring\ )\ )\ )\ ). \tag{1}$$

As in KHIs, G2IDs in our proposed architecture are designed to serve as identifiers rather than locators. While, in KHIs, the *prefix* is used to distinguish KHIs from IPv6 addresses, we utilize the *context ID* instead of the *prefix* to distinguish global identifier contexts (naming methods). For instance, in our algorithm, the *context ID* may be used to distinguish global identifiers for the data link layer from global identifiers for the application or transport layers. In addition, in our algorithm the *context ID* is limited to 8 bits in length to generate 128 bits long G2IDs.

To generate a G2ID, we encode the *bitstring* for a RSA public key utilizing four information fields as defined in [13]: exponent length, exponent (e), modulus length, and modulus (n). The public key exponent length is one or three octets depending on its value. If the exponent length is in the range of 1 to 255, it is represented as one octet. Otherwise, the exponent length is represented as one zero octet followed by a two octet unsigned length. Moreover, both the exponent and modulus are each limited to 4096 bits in length. The *bitstring* value for a RSA public key is calculated as follows:

$$bitstring_{RSA} = exponent \mid exponent\ length \mid modulus \mid modulus\ length. \tag{2}$$

To generate local identifiers, we utilize G2IDs and two PRNs (PRN1, PRN2), each 64 bits long. Initial PRNs are selected randomly. However, for the subsequent L2ID computations, PRNs are exchanged during the key establishment protocol. A simple method of generating a L2ID is to concatenate the G2ID with PRNs and hash the result using SHA1 as in (3).

$$L2ID = extract_{48}\ (\ SHA1\ (\ expand\ (\ G2ID \mid PRN1 \mid PRN2\ )\ )\ ). \tag{3}$$

3.2. Network Structure

In the proposed data link layer security architecture, we utilize the IEEE 802.1X [14] concepts for access control. In addition, we incorporate the IEEE P802.1AE standard [1] and use a key hierarchy similar to the IEEE 802.11i standard [6] for future compatibility of wired and wireless networks.

The proposed data link layer security architecture has three main components: authentication servers, authenticators, and hosts.

3.2.1. Authentication Servers. We utilize authentication servers to establish realms and security parameters in local networks. We assume that authentication servers are integrated into routers. Each authentication server records and manages the data link layer identifiers in its realm. Hosts negotiate security parameters and their data link layer identifiers (L2IDs) with authentication servers during the key establishment protocol. Specifically, authentication servers and hosts utilize the key establishment protocol to perform mutual authentication, to generate session keys, and to agree on L2IDs. In addition, authentication servers assign IP addresses to hosts in their realms at the end of the key establishment protocol. Each new host moving into the realm of an authentication server is required to perform the key establishment protocol, negotiate a L2ID without collisions in the realm, and obtain an IP address. However, authentication servers may assign the same IP address to several hosts with different L2IDs. We assume that authentication servers utilize a distributed database maintaining the list of G2IDs, L2IDs, and IP addresses for network access.

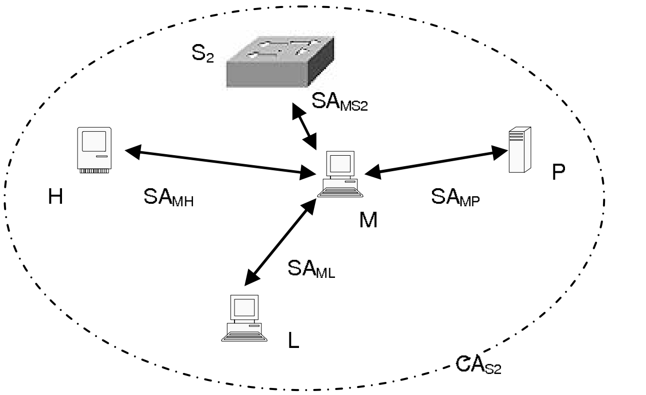

Figure 2. An illustration of the security associations of the host *M*.

3.2.2. Authenticators. Authenticators are the data link layer devices that act as gateways between hosts and authentication servers. In the proposed data link layer security architecture, authenticators function as access points (similar to the security model of the IEEE 802.11i). We assume that authenticators are Layer 2 devices, such as switches. Figure 1 illustrates an example of the network architecture where switches function as authenticators. Authenticators communicate with authentication servers to receive their L2IDs and to establish security parameters. Authenticators allow control messages to pass during the key establishment protocol between hosts and authentication servers. In addition, each authenticator controls a connectivity association (CA). Each CA consists of an authenticator and a number of hosts. Each host, identified by a L2ID, participates in a single CA at any one time. However, a host with several data link layer connections (L2IDs) can participate in more than one CA. Moreover, authenticators learn the security parameters of the hosts and the L2IDs of the hosts and other authenticators in the same realm from authentication servers utilizing a secure protocol.

Each CA is supported by security associations (SA). Figure 2 shows the security associations of the host *M* in the CA_{S2}. In the figure, the four SAs, SA_{MH}, SA_{ML}, SA_{MP}, and SA_{MS2}, provide secure communication between *M* and the other hosts where each association is bidirectional. All the SAs in a CA use the same cipher suite at any one time. Finally, authenticators with direct links create SAs with each other, as well.

3.2.3. Hosts. In the proposed data link layer security architecture, hosts are identified by L2IDs. Each L2ID, including the L2ID of an authenticator, corresponds to a MAC Security Entity (SecY) in the IEEE 802.1AE standard. A host, first, utilizes the key establishment protocol to negotiate its L2ID, compute security paramters, and learn its IP address from an authentication server. However, at the end of the key establishment protocol, before the host can send any data frames, it is required to create SAs in its CA. After creating a SA with the authenticator, the IEEE 802.1X Controlled Port is unblocked allowing the host to transmit and receive data frames. The host utilizes the four-way handshake protocol, defined in the IEEE 802.11i standard [6], to establish a SA with the authenticator. When the host completes the four-way handshake protocol, it becomes a member of the CA and creates SAs with other data link layer devices that it is connected to. Each SA represents a single value of the transient session key(s) used for a period by the cipher suite to support the communications between two data link layer devices. After creating SAs, a host wishing to communicate with a destination host finds the location (IP address) and the identity/identifier of the destination host via a Fully Qualified Domain Name (FQDN) or another method. Note that hosts participating in different CAs communicate through authenticators.

3.3. Key Management

In the proposed data link layer security architecture, there are four different type of links that require confidentiality, data authentication, and replay protection mechanisms: authentication servers to/from authenticators, authenticators to/from hosts, hosts to/from hosts, and authenticators to/from authenticators.

3.3.1. Authentication Servers to/from Authenticators. In the proposed data link layer security architecture, we assume that authentication servers and authenticators create security associations and utilize secure communication protocols. While the key establishment protocol messages do not require any encryption between an authenticator and an authentication server, the frames that carry the pair-wise master key (PMK) information and other security parameters between an authentication server and authenticator should be protected. A PMK is derived from a master key computed at the end of the key establishment protocol between a host and an authentication server.

We utilize the HIP keying material derivation method, described in [15], to compute a PMK. The PMK is computed as in (4), where the $G2ID_{AS}$ and $G2ID_{Host}$ are 128-bit integers representing the global identifiers of an authentication server and a host, respectively. Both the host and the authentication server compute the PMK. Later, after completing the key establishment protocol with the host, the authentication server securely transports the PMK to an authenticator.

$$PMK = SHA1\ (\ masterkey \mid Min(\ G2ID_{AS}\ ,\ G2ID_{Host}\ ) \mid Max\ (\ G2ID_{AS}\ ,\ G2ID_{Host}\ ) \mid I \mid J \mid 0 \times 01\ ). \qquad (4)$$

3.3.2. Authenticators to/from Hosts. Authenticators and hosts utilize the four-way and group handshake protocols, defined in the IEEE 802.11i standard [6], to create SAs and a CA with fresh keys. Authenticators and hosts employ EAPOL-Key frames in these protocols. The four-way handshake protocol enables an authenticator and a host to derive a fresh pair-wise transient key (PTK) from a pair-wise master key (PMK). Moreover, the authenticator confirms the liveliness of the host and that the host holds the PMK. In addition, during the four-way handshake protocol, the authenticator transports the group transient key (GTK) to the host. Furthermore, the authenticator informs the host regarding the cipher suite selection used in the CA and other hosts belonging to the same CA. At the end of the four-way handshake protocol, both parties install pair-wise encryption and integrity keys. The host installs the GTK as well. In our security architecture, a GTK and a PTK represent a CA and a SA, respectively. While the GTK is the same for all the data link layer devices in the same CA, the PMK is different for each SA.

3.3.3. Hosts to/from Hosts. Hosts utilize SAs with authenticators to secure data frames sent to hosts in other CAs. However, hosts can use host-to-host keys to secure data frames directly to other hosts in a CA.We propose to utilize the STAKey handshake defined in the IEEE 802.11i standard [6] to create security associations between hosts. After each host establishes a SA with an authenticator, the authenticator transfers STAKey handshake messages between hosts. The originating host requests the STAKey by sending an EAPOL-Key frame to the authenticator with the L2ID of a peer host. The authenticator sends a STAKey message 1 to the peer host with the L2ID of the originator to provide a STAKey. The peer host responds to the authenticator sending a STAKey message 2 with the L2ID of the initiator host. The authenticator (after receiving the STAKey message 2 from the peer host) sends a STAKey message 1 to the initiator host with the L2ID of the peer host and the STAKey. The STAKey message exchange ends with a STAKey message 2 from the initiator host to the authenticator containing the L2ID of the peer host. In summary, the authenticator provides the key for both hosts to use for securing the connection.

3.3.4. Authenticators to/from Authenticators. In the proposed security architecture, we assume that authenticators utilize a secure protocol to communicate. Authenticators create security associations and utilize a secure protocol to facilitate mobility and signaling. Security associations among authenticators provide protection to data frames transferred between hosts in different CAs.

3.4. The Key Establishment Protocol

A general form of the key establishment protocol in our data link layer security architecture is illustrated in Fig. 3. Our key establishment protocol is based on the Just Fast Keying (JFK) protocol [16] and the Sign-and-MAC (SIGMA) protocol [17] with appropriate modifications to provide identity protection for the initiator. The key establishment protocol utilizes the Diffie-Hellman exchange. In the exponential notations g^x and g^y, x and y are random exponents, and g is a Diffie-Hellman group generator. We assume that the host (initiator) knows an acceptable group generator of the authentication server (responder). In the first message, the alias $\hat{A}$ is computed by the initiator, A, as $\hat{A} = hash(A;\ r)$, where r is a random number. In the third message, the initiator reveals its identity to the responder by encrypting both its real identity, A, and the random number, r. The notation $K_e\{\}$ is used to denote that the data between the brackets are encrypted with the symmetric key of K. The symmetric key K is derived from the Diffie-Hellman value (master key) g^{xy}. However, in the key generation process, session keys are derived from the master key independently of K. Finally, the notations $sig_A()$ and $sig_B()$ are used to denote that the messages between the parentheses are signed with the private keys of A and B, respectively.

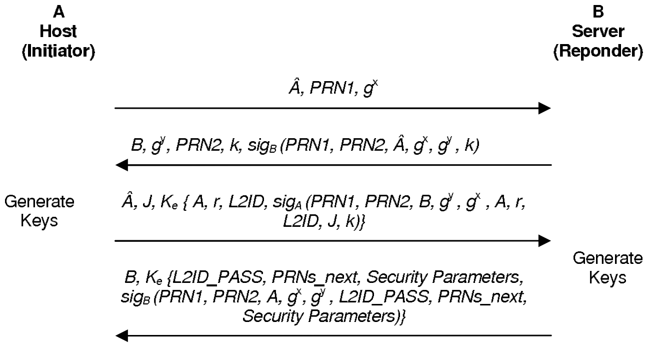
A
Host
(Initiator)
B
Server
(Reponder)
Â, PRN1, g^x
B, g^y, PRN2, k, sig_B (PRN1, PRN2, Â, g^x, g^y , k)
Generate Keys
Â, J, K_e { A, r, L2ID, sig_A (PRN1, PRN2, B, g^y , g^x , A, r, L2ID, J, k)}
Generate Keys
B, K_e {L2ID_PASS, PRNs_next, Security Parameters, sig_B (PRN1, PRN2, A, g^x, g^y , L2ID_PASS, PRNs_next, Security Parameters)}

Figure 3. The key establishment protocol for the data link layer security architecture.

In our key establishment protocol, public-private keys, which are long-lived identities for hosts, are utilized to help create short-lived encryption and authentication keys for the data link layer. Specifically, a host and a server generates a master key, g^{xy}, using a Diffie-Hellman key agreement protocol. After completing the key establishment protocol, both the host and the server calculate encryption and message authentication keys from the master key. We employ public keys in the key establishment protocol for three purposes: verifying identities, creating data link layer identifiers, and generating session keys.

3.5. The Puzzle Mechanism

DoS attacks based on protocols remain a serious threat to networks and users. DoS attacks, by their nature, are difficult to prevent. A DoS attack may be characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service [18]. Key exchange and authentication protocols are vulnerable to DoS attacks that exhaust the servers' processing resources. Puzzles have been proposed as a countermeasure to DoS threats in communication networks [15, 19–24]. In our data link layer security architecture, we utilize a client puzzle in the key establishment protocol to delay state creations at servers. Hosts (initiators) perform computations to solve a puzzle and prove that they are willing to allocate resources to access the servers. The puzzle mechanism allows servers to check the answers by simply computing one hash function.

In the key establishment protocol, the server sends a puzzle, containing a random number *PRN*2 and a puzzle strength *k*, after receiving the initial message from a host without computing the Diffie-Hellman (master) key. The host must solve this cryptographic challenge to continue the key exchange. The server discards messages containing incorrect puzzle answers. The server may also adjust the level of difficulty of the puzzle by setting value *k*.

To solve the puzzle, the host (initiator) generates a number of random numbers, *J*s, and computes the hash values as in (5) until the lowest order *k* bits of the hash are all zeros. The host gives up solving the puzzle if it exceeds the puzzle lifetime. The server verifies the puzzle by computing the same hash value once using the *J* provided by the host.

$$SHA1\ (\ PRN1\ |PRN2\ |\ B\ |\ \hat{A}\ |\ J\ ). \qquad (5)$$

In (5), the random numbers *J*, *PRN*1, and *PRN*2 are 64-bit integers whereas *B* and $\hat{A}$ are 128-bit integers. The puzzle difficulty, *k*, is an 8-bit integer. It takes, on average, $2^{(k\text{-}1)}$ hash calculations to solve the puzzle [20]. Since the output of the hash function is 160 bits long, the reasonable values of *k* lie between 0 and 80. Setting the *k* to 0 means that the puzzle mechanism is disabled. In that case, the server accepts any *J* value.

3.6. Discussion

Since our objective is to secure the data link layer and bind upper layers and the data link layer, we also focus on preventing design specific attacks, such as identity misbinding attacks, in addition to well-known attacks. While the authentication, confidentiality, and data integrity requirements are widely known and expected, the requirement of identity binding is usually overlooked. We should emphasize that identity binding is essential in our data link layer security architecture to authenticate messages. We prevent identity misbinding attacks in our key establishment protocol by including identities under signatures.

In key establishment protocols, identities are transmitted as a part of the protocols since each party needs to know the identity of the other party for mutual authentication. However, unprotected identities are prone to identity-probing attacks from any machine in the network. For instance, an attacker may initiate a key establishment protocol to find the identity of a machine at a certain IP address. To prevent this type of attack, the key establishment protocol may reveal the identity of the responder only after the initiator reveals its identity. On the other hand, in some cases, it may be more suitable to reveal the identity of the responder first. In the proposed key establishment protocol, we choose to protect the identities of hosts from active and passive attacks.

A key property of secure protocols is the protection of past session keys in spite of the compromise of long-term secrets. This property is known as perfect forward secrecy. In our key establishment protocol, the Diffie-Hellman exchange provides this property for master keys. In addition, in the case that information leakage happens, where some session specific information or the value of a session key is learned by an attacker, we require that any adverse security consequence from such a compromise will affect the exposed session only. This security principle can be achieved by deriving session (temporary) keys, such as encryption and message authentication code keys, from a master key computed in the key establishment protocol, independently of the symmetric key of *K*.

Another desirable, but not necessarily required, property of secure protocols is non-repudiation. By non-repudiation property, the signer of a digital signature is prevented from denying having signed a document after signing it [27]. In general, this property prevents the denial of previous commitments and actions. However, this property comes with the price of digital signatures utilizing public-private keys. For that reason, we choose to employ this property only when it is essential in the key establishment protocol, which in our case is the last three messages.

While it may not be possible to prevent DoS attacks, key establishment protocols may utilize various techniques to reduce this type of attack. We utilize a puzzle mechanism, which is similar to the Host Identity Protocol (HIP) puzzle mechanism [15], to mitigate DoS attacks in our key establishment protocol. In our puzzle mechanism, we utilize *PRN*1s and *PRN*2s as session identifiers. We also utilize these pseudo-random numbers to generate L2IDs. An attacker can pre-compute puzzle solutions by estimating the *PRN*2s. To prevent pre-computation attacks, *PRN*2s should not be easily guessed by hosts. In addition, servers should generate new *PRN*2s once in every few minutes. Moreover, servers should verify the puzzle values in the responses. Furthermore, servers may need to remember old puzzles for a limited time to allow slower hosts to solve the puzzles. Also, utilizing $\hat{A}$ instead of real host identities prevents attackers from identifying hosts by observing messages. For that reason, our puzzle mechanism prevents attackers from pre-computing puzzle solutions for specific hosts. If the server receives a correct puzzle solution sent by an attacker, it will not be able to verify the signature in the received message. In that case, the server will send a *PUZZLE_FAIL* message to the host to prevent more attacks. The server should record these *PRN*s and the $\hat{A}$ and avoid utilizing them in the puzzle mechanism. Attackers can send *PUZZLE_FAIL* messages to hosts to cause DoS as well. To prevent this type of attack, hosts should utilize a timer to end the sessions. Finally, since attackers can send false puzzle solutions to servers to cause DoS, servers should also use a timer to wait for the correct puzzle solutions.

4. Conclusions

In this paper, we introduced a data link layer security architecture with security inter-layering in IP over Ethernet networks. We proposed to utilize secure identities such as public keys to secure the link between the data link and the network layers in local networks. We described the algorithms to generate global identifiers from public keys (upper layer identities) and local data link layer identifiers from global identifiers. We presented the network structure providing link-to-link security with security and connectivity associations. In addition, we described a method to establish secure associations and addressed the key management in this architecture, which is not included in the scope of the IEEE 802.1AE standard. Nevertheless, to enable security inter-layering and secure identities at the data link layer, modifications to the IEEE 802.1AE standard are required. Moreover, we proposed a key establishment protocol to negotiate data link layer identifiers, establish security parameters, and mutually authenticate hosts and authentication servers. Furthermore, in the key establishment protocol, we addressed misbinding attacks protecting the identities of hosts at the data link layer. In the proposed key establishment protocol, we utilize a puzzle mechanism to thwart DoS attacks as well. We also utilized the four-way handshake protocol and the key hierarchy of the IEEE 802.11i standard to be compatible with wireless networks addressing security issues between wireless and wired networks.

The proposed data link layer architecture separates identities and locations supporting mobility and multi-homing. The proposed architecture modifies other inter-networking layers as well. It requires the network layer to explicitly incorporate identifiers or identities in IP packets. Finally, this architecture requires all data link layer devices, such as switches/bridges, to own data link layer identifiers.

References

[1] *IEEE P802.1AE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security*, January 2006, IEEE Working Draft, D5.1. [Online]. Available: www.ieee802.org/1/files/private/ae-drafts/d5/802-1ae-d5-1.pdf

[2] "IEEE 802.1AE-Media Access Control (MAC) Security," July 2006, [Online]. Available: http://www.ieee802.org/1/pages/802.1ae.html.

[3] C. Howard, "Layer 2 – The weakest link: Security considerations at the Data Link Layer," *PACKET*, vol. 15, no. 1, First Quarter 2003.

[4] H. Altunbasak, S. Krasser, H. L. Owen, J. Grimminger, H.-P. Huth, and J. Sokol, "Securing Layer 2 in Local Area Networks," in *ICN*, vol. 2, Reunion, France, April 2005, pp. 699–706.

[5] H. Altunbasak, S. Krasser, H. Owen, J. Sokol, J. Grimminger, and H.-P. Huth, "Addressing the weak link between Layer 2 and Layer 3 in the Internet architecture," in *Proc. of the 29th Annual IEEE Conference on Local Computer Networks (LCN)*, Tampa, Florida, November 2004.

[6] *IEEE Std 802.11i, Amendment to IEEE Std 802.11 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Medium Access Control (MAC) Security Enhancements*, June 2004.

[7] T. Karygiannis and L. Owens, *Wireless Network Security 802.11, Bluetooth and Handheld Devices*, November 2002, Special Publication 800-48, Recommendations of the National Institute of Standards and Technology. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-48/NIST_SP_800-48.pdf.

[8] D. C. Plummer, "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware," IETF RFC 826, November 1982.

[9] S. Kent and R. Atkinson, *IP Authentication Header*, Nov. 1998, RFC 2402. [Online]. Available: http://www.ietf.org/rfc/rfc2402.txt.

[10] Stephen Kent and Randall Atkinson, *IP Encapsulating Security Payload (ESP)*, Nov. 1998, RFC 2406. [Online]. Available: http://www.ietf.org/rfc/rfc2406.txt.

[11] "NISCC vulnerability advisory IPSEC - 004033," May 2005, [Online]. Available: http://www.niscc.gov.uk/niscc/docs/al-20050509-00386.html?lang=en.

[12] P. Nikander, J. Laganier, and F. Dupont, "A Non-Routable IPv6 Prefix for Keyed Hash Identifiers (KHI)," Network Working Group Internet Draft, September 2005. [Online]. Available: http://tools.ietf.org/wg/ipv6/draft-laganieripv6-khi-00.txt.

[13] Donald E. Eastlake, "RSA/SHA-1 SIGs and RSA KEYs in the Domain Name system (dns)," IETF RFC 3110, May 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3110.txt.

[14] *IEEE 802.1X-2001 IEEE Standards for Local and Metropolitan Area Networks: Port-Based Network Access Control (EAPOL)*, 2001.

[15] R. Moskowitz, P. Nikander, P. Jokela, and T. R. Henderson, "Host Identity Protocol," Internet draft, March 2006, work in progress. [Online]. Available: http://www.ietf.org/internetdrafts/draft-ietf-hip-base-05.txt.

[16] W. Aiello, S. M. Bellovin, M. Blaze, J. Ioannidis, O. Reingold, R. Canetti, and A. D. Keromytis, "Efficient, dosresistant, secure key exchange for internet protocols," in *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*. New York, NY, USA: ACM Press, 2002, pp. 48–58.

[17] H. Krawczyk, “SIGMA: The ’SIGn-and-MAc’ approach to authenticated Diffie-Hellman and its use in the IKE protocols,” *Lecture Notes in Computer Science*, vol. 2729, pp. 400–425, Oct. 2003.

[18] K. J. Houle and G. M. Weaver, *Trends in Denial of Service Attack Technology*, October 2001. [Online]. Available: http://www.cert.org/archive/pdf/DoS trends.pdf.

[19] T. Aura, P. Nikander, and J. Leiwo, “Dos-resistant authentication with client puzzles,” in *Revised Papers from the 8th International Workshop on Security Protocols*. London, UK: Springer-Verlag, 2001, pp. 170–177.

[20] D. Dean and A. Stubblefield, “Using client puzzles to protect tls,” in *10th Annual USENIX Security Symposium*, 2001.

[21] D. B. Faria and D. R. Cheriton, “Dos and authentication in wireless public access networks,” in *WiSE ’02: Proceedings of the 3rd ACM workshop on Wireless security*. New York, NY, USA: ACM Press, 2002, pp. 47–56.

[22] M. Handley and A. Greenhalgh, “Steps towards a dosresistant internet architecture,” in *FDNA ’04: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*. New York, NY, USA: ACM Press, 2004, pp. 49–56.

[23] X. Wang and M. K. Reiter, “Mitigating bandwidth exhaustion attacks using congestion puzzles,” in *CCS ’04: Proceedings of the 11th ACM conference on Computer and communications security*. New York, NY, USA: ACM Press, 2004, pp. 257–267.

[24] B. Waters, A. Juels, J. A. Halderman, and E. W. Felten, “New client puzzle outsourcing techniques for dos resistance,” in *Proceedings of the 11th ACM conference on Computer and communications security*. ACM Press, 2004, pp. 246–256. [Online]. Available: www.cs.princeton.edu/jhalderm/papers/ccs2004.pdf.

[25] Advanced Encryption Standard (AES), November 2001, FIPS 197. [Online]. Available: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[26] D. Whiting, R. Housley, and N. Ferguson, *Counter with CBC-MAC (CCM)*, September 2003, RFC3610. [Online]. Available: http://www.faqs.org/rfcs/rfc3610.html.

[27] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1997.