DoS and Authentication in Wireless Public Access Networks

Daniel B. Faria
dbfaria@cs.stanford.edu

David R. Cheriton
cheriton@cs.stanford.edu

Computer Science Department
Stanford University
Stanford, CA 94305-9040

ABSTRACT

As WEP has been shown to be vulnerable to multiple attacks, a huge effort has been placed on specifying an access control mechanism to be used in wireless installations. However, properties of the wireless environment have been exploited to perform multiple DoS attacks against current solutions, such as 802.11/802.1X. In this paper we discuss the main wireless idiosyncrasies and the need for taking them into account when designing an access control mechanism that can be used in both wireless and wired networks. We present the design of a mobility-aware access control mechanism suitable for both wireless and wired environments and show how the DoS attacks discussed can be prevented by implementing secure association and other essential services. The architecture proposed here, composed of the SIAP and SLAP protocols, uses public keys together with the RSA and AES encryption algorithms to provide a flexible service.

Categories and Subject Descriptors

C.2.2 [**Computer Systems Organization**]: Network Protocols

General Terms

Design, Security

Keywords

DoS, Security, Wireless Networks

1. INTRODUCTION

The advent of portable and mobile computing has motivated the introduction of public access networks, available in airport lounges, cafes, and inside universities and enterprise environments, being sometimes shared by both local users and visitors. Advances in wireless communications have taken this service to another level and enabled seamless communication, in which a user does not even know her point of connection to the network. These computing facilities, however, have increased the need for more restrictive access control mechanisms, given the difficulty to physically restrict the access to wireless access points.

Current practice on dealing with wired ports has been to physically secure the enterprise network by hiding the wires inside walls and securing the switches and routers in locked wiring closets. We view this portion of the network as the true *intranet*. The only points of physical exposure are the RJ-45 ports in the wall outlets and the wireless access antennae distributed around the environment. We call this part of the network the *public access network*.

Providing security involves many trade-offs. The number and complexity of mechanisms implemented in a security architecture depend on the assumptions made about the environment in which they are supposed to operate. Recently, wireless security has been given considerable attention, a consequence of the vulnerabilities found by the research community in current standards. Wireless security research has contributed to make clear that one should make as few assumptions as possible about the physical security provided by the network infrastructure. For example, the way association and authentication are integrated in 802.11/802.1X networks is the main reason for the attacks already reported[16]. Disassociation attacks, session hijacking, and the implementation of rogue access points are examples of attacks that can be mounted over a 802.11 wireless network given the invalid assumption made by 802.1X that secure association is provided.

Fewer assumptions about physical security generate more robust solutions, but special attention should also be paid to the amount of mechanism provided. With security, every mechanism has the potential of creating a vulnerability or enabling denial of service (DoS) attacks. We therefore support the idea that less mechanism is better, restricting the provided services to the ones that are really essential to creating a secure solution. Going back to wireless networks, we see secure association as an important service to provide. The challenge is therefore to make as few assumptions as possible about the network while providing a robust solution based on a small number of protocols and services.

The first objective of this paper is to show that the DoS attacks effective against current access control solutions are made possible by the lack of implementation of essential services or wrong assumptions made about the environment.

WiSe'02, September 28, 2002, Atlanta, Georgia, USA.

We address several classes of DoS attacks while focusing on authentication-related attacks in wireless networks. In section 2 we identify what we believe to be the essential services that should be provided by an access control mechanism, while section 3 shows how wireless networks make explicit the need for mobility support, access point authentication, and secure association. In section 4 we analyze the main objectives of the 802.11 and 802.1X standards, how they contradict the objectives of the architecture proposed in this paper and how some of the assumptions made enable DoS attacks.

This paper also describes an access control architecture that is flexible enough to secure different LAN technologies while providing the user with mobility capabilities in wireless networks. Our architecture, presented in section 5, is composed of the *Secure Internet Access Protocol* (SIAP) and the *Secure Link Access Protocol* (SLAP). The design of these protocols follows the ideas described previously, making few assumptions about physical security while implementing the services that are essential for a security architecture. We show that by doing so, SIAP/SLAP can be used to implement a secure association service and avoid the DoS attacks discussed in this paper. SIAP and SLAP rely on the security of robust constructions and encryption algorithms, such as AES [8] and RSA [20], and preliminary results indicate the viability of the architecture in its application for 802.11 networks.

2. AN ACCESS CONTROL FRAMEWORK

We consider access control to be a distinct problem from end-to-end (E2E) security. While the service provider is completely oblivious to how E2E security measures work, it is highly interested in controlling who gets to use its network infrastructure. A faculty member from Berkeley visiting Stanford will never be able to set up a secure E2E connection with her mail server in Berkeley if she has not satisfied Stanford's requirements, a necessary step to get network connectivity. In summary, provider control is necessary before the client can set up end-to-end connections, which the provider does not care about.

Users and network providers have different demands over an access control mechanism. While providers are mainly interested in AAA[1], clients are interested in ease of use and having mobility capabilities. However, we believe that most demands from both users and providers can be satisfied if the access control architecture implements a minimum set of features:

i. mutual authentication;

ii. flexible authorization;

iii. access verification;

iv. interoperability;

v. simple user interface;

vi. data confidentiality and integrity.

Each one of these features actually encompasses a set of desirable properties. Mutual authentication is usually accomplished by means of an authenticated key establishment protocol. Together with authorization, it provides the network administrator with the ability to implement client differentiation while allowing the user to authenticate network entities. Other desirable properties related to the key establishment protocol include key freshness guarantees, forward secrecy, and DoS resistance[15].

After the user is authenticated, an access verification mechanism should be available. The provider is interested in verifying permissions and performing accounting and billing tasks. The user wants guarantees over the integrity over transmitted packets and that attackers cannot affect the accounting process executed by the provider. These requisites are usually achieved using cryptographic message authentication codes (which also provide message integrity) and a replay detection mechanism.

Interoperability is especially important in wireless networks, as users expect to move between networks as smoothly as possible. Interoperability and simple user interface are key factors to achieve user satisfaction.

The user may want to extend the confidentiality provided by end-to-end mechanisms, e.g. based on IPSec [12] or TLS [9], keeping secret all data transferred over the wireless link. As the wireless medium extends the reach of attackers, applications not usually protected by higher-layer mechanisms can take advantage of this service.

In this paper we focus on features (i) to (iii) above and how the absence of some of the related services creates security vulnerabilities. DoS attacks based on the lack of correct authentication constructs receive special attention in the paper, especially when dealing with wireless networks (section 3). Section 4 shows how 802.1X-based solutions are vulnerable to these authentication-based attacks, while attacks mounted over authorization and verification mechanisms are discussed in section 6.

2.1 A Two-protocol Architecture

Most of the solutions proposed for this problem are composed of two protocols. An authentication protocol, usually running at the application layer, provides mutual authentication and sets up fresh session keys and other parameters, which define a secure channel between user and network. This protocol is responsible for items (i), (ii), (iv), and (v) above.

A second, lower-layer protocol receives the negotiated parameters from the authentication protocol after a successful handshake and uses them to provide confidentiality, integrity, and message authentication over packets sent and received during that session. Its functions overlap with features (iii), (iv), and (vi), while its main objective is to provide a secure link abstraction, in which packets sent over the network are generated by and visible only to authenticated entities. It is compelling to provide a lower-layer protocol that is link-layer independent, to increase interoperability between different network technologies.

This two-protocol approach provides a compelling solution, as most of the burden of the authentication process and mobility management is left to the application-layer protocol. The configuration channel between the two protocols enables the implementation of simpler services at the lower layer.

[1]Authentication, Authorization, and Accounting.

2.2 Authentication Results

The main result of the authentication process is the negotiation of fresh, per-client session keys. Additionally, the authentication process can provide different users with different views of the network. One way to achieve this is to coalesce the authentication protocol and the IP address assignment service. In this case, providing different views of the network is done by having different address pools and previously configured gateways that enforce the desired policy. For example, a campus network may reserve public IP addresses for students while providing visitors with private addresses `10.0.0.0/16` that are restricted to browsing the university's Web pages.

Having IP addresses assigned by the authentication protocol, the security state passed to the lower-layer protocol becomes $(MACaddress, IP, keys)$. With this scheme, it becomes easier to enforce the bond between the IP address and the session keys while eliminating specific attacks on DHCP and other configuration protocols (section 6.)

3. WIRELESS SECURITY

Why is wireless security generating such a fuss? The first reason is that wireless LANs are being deployed everywhere, from enterprise environments to cafeterias and airport lounges. Second, wireless networks introduce new concepts that have no parallel in wired networks. Some of these create new challenges that need to be considered when creating security solutions.

3.1 Mobility

Providing mobility is one of the main pillars of the success of wireless computing. We also expect this to hold as new access control mechanisms are deployed. As the authentication protocol is responsible for automatically setting up a secure state for a client in a given network, it should not be hard to extend this mechanism to provide transparent migration[2] between networks with different address prefixes. However, providing transparent mobility between different domains depends on the existence of an agreed set of protocols.

3.2 Access Point authentication

A user may trust an Ethernet port in a wall inside her company as providing a secure connection to the network, given the fact that wiring closets are usually physically secure. However, a signal captured by a wireless card in the user's laptop does not inspire much confidence, as it is of little complexity to set up a base station that advertises to be part of the network. Moreover, handoffs impose connection changes that are not visible to the user, increasing the need for having the access points authenticated by the client. This should become even more severe as wireless deployment advances and different networks overlap with each other.

As discussed in section 4, not authenticating access points may create the possibility of man-in-the-middle and denial of service attacks.

[2]Additional mechanisms may be necessary to support persistent transport connections.

3.3 Association

One concept introduced by IEEE 802.11 is that of *association*. As defined by the standard, association is *"the service used to establish access point/station (AP/STA) mapping and enable STA invocation of the distribution system services"* [1]. Basically, a client C_1 is associated with an access point AP_1 when frames sent to or from C_1 are bridged by AP_1. As AP_1 is the only AP processing the client's packets, this also helps avoiding duplicates.

In order to get associated with a given AP, a client initiates a two-message handshake. The association mechanism is used to provide transparency to the sender as the receiver moves between access points. A station is not allowed to send a data message via an access point before it gets associated with that AP. This way, at any given time, the system knows the AP currently serving each client.

3.4 Authenticate and Associate

Association is therefore a packet-based emulation of the act of "plugging" a laptop to a given base station, as would happen with a wired LAN. However, this small difference is sufficient to create problems when combining association and authentication. The question here is whether the messages sent during the association setup should be authenticated or not. For example, a message authentication code (MAC[3]) could be sent with each message in order to provide integrity and sender authentication.

Assume for a moment that this is not the case, i.e. that clients and APs exchange association messages with no MAC protection. It is easy to see that an attacker can perform effective denial of service attacks by simply sending disassociation messages on behalf of a client or AP. Another possibility is for the attacker to set up a fake base station that just get clients associated with it. "Unplugging" someone else's laptop form the network is as easy as sending a packet.

These DoS attacks can be prevented by the use of a MAC algorithm (such as HMAC-MD5 [14]) to provide message authentication. However, such algorithms depend on secret keys shared by the two entities. This secret can be statically set or dynamically negotiated by the upper-layer authentication protocol used. The use of statically configured secrets has not proven popular with the increase in security demands.

As further discussed in section 4, current security solutions use an authentication server (AS) hidden behind the access points. In order to authenticate and get network connectivity, a client communicates with the AS through one of the APs. We now get to a dependency cycle, if one wants to have the association messages authenticated. In order to send data frames through the AP, a client needs to be associated. However, if the client associates before it authenticates itself, there is no shared key to provide sender authentication over the association messages. The order associate-then-authenticate seems necessary to contact back-end ASs, being prone to the DoS attacks already discussed. We therefore advocate the authenticate-then-associate order, with some measure needed to break this dependency cycle.

We finish this section by summarizing our conclusions. First, when dealing with wireless networks, the access control mechanism should provide the client with access point

[3]In this paper we use the acronym MAC alone to refer to the authentication code, being explicit when dealing with MAC addresses.

authentication. When dealing with wired networks, this is not strictly necessary. Second, in order to eliminate DoS attacks related to the association process, authentication should be executed before association and association messages should be protected against tampering and replay attacks. These observations follow the ideas presented in the introduction, i.e. a robust access control mechanism should make as few assumptions as possible, not trusting access points and minimizing the services run before authentication takes place (e.g. association.)

4. IEEE 802.11 AND 802.1X

4.1 IEEE 802.11

Authentication

This standard had very limited objectives when dealing with authentication and confidentiality services. It defined an authentication protocol based on a shared key known by APs and client machines. A four-message handshake is performed in order to authenticate the client, one of these messages containing a challenge text that has to be successfully encrypted by the client using the shared key [1]. This authentication mechanism defined by 802.11 is usually turned off when the protocol is integrated with the 802.1X framework, which provides a broader set of authentication primitives.

Association

A finite-state machine (FSM) in the standard defines three states for a client, depending on its association and authentication statuses. At any given time, a client is in one out of three possible states: unauthenticated/unassociated, authenticated/unassociated, or authenticated/associated. The FSM requires the client to run the authentication algorithm before it can associate with a given AP.

The problem is that even when shared key authentication is used, the association messages (which are of type *management*) receive no WEP service. Therefore, even though it is harder for an attacker to impersonate an absent user without the knowledge of the shared key, it is very easy for her to select an already authenticated and associated user to flood with disassociation messages. The client would then keep oscillating between the authenticated/associated and the authenticated/unassociated states, an effective DoS attack.

Providing secure association messages would not only deny such DoS attacks but could also provide the client with AP authentication. This could be achieved, for example, by providing a three-message association handshake protected by a key shared between client and AP. We return to this matter later in the paper.

WEP

The original confidentiality service provided by WEP was also based on the use of a key shared between all the users of a network. However, even per-client session keys set up by an upper-layer authentication protocol and used by WEP do not provide enough security. The combination of a non-standard construction using the RC4 stream cipher and the use of a CRC algorithm as an integrity mechanism has shown WEP to provide very limited security [25, 7, 4, 11, 22].

4.2 IEEE 802.1X

The IEEE 802.1X [2] framework was defined in order to provide port-based access control for the IEEE 802 protocol family. Together with a possibly revised version of WEP, this framework aims to solve the security vulnerabilities of the 802.11 original design.

Authentication

The standard defines three entities: the supplicant, an authenticator, and an authentication server. In a wireless network the supplicant is the mobile station while the authenticator is an 802.1X-enabled access point. An authentication server can be located anywhere behind the access points.

In order to provide different authentication mechanisms, the standard uses the EAP [6] protocol as a tunnel between client (supplicant) and server, passing through the access point. Instead of defining a specific authentication protocol, the use of EAP lets enterprises extend to the wireless network the authentication mechanism used in their wired infrastructures. For example, TLS [9] and Kerberos [13] can be used on top of EAP.

When sent from the client to the AP, EAP messages are encapsulated over EAPOL[4] messages. Between the AP and the authentication server, RADIUS [18] is one of the options. The standard states that authenticator and server may be colocated, making unnecessary some of the protocols discussed above. However, commercial products and initial installations using 802.1X have been using a back-end server, with RADIUS as the encapsulation protocol.

Association

The 802.1X standard allows the use of EAPOL in shared medium LAN environments, such as 802.11. However, it also states that "*attempting to use EAPOL in a shared medium environment that does not support the use of secure association renders Port-based network access control highly vulnerable to attack.*" This passage suggests that clients (securely) associate before they can run EAPOL, and consequently the authentication protocol of choice. Even though secure association is not an issue in wired networks, we have already discussed some problems when dealing with 802.11 environments. Misra *et al.* [16] have made an initial analysis of some of the problems on integrating IEEE 802.11 and 802.1X and described some attacks.

Conceptually, it seems possible to have the client run the authentication protocol before it gets associated with an AP. Before the client gets associated, APs could let pass only frames that contain authentication messages. After authenticated, client and AP could exchange secure association messages using the session key just negotiated. However, substantial changes to both 802.11 and 802.1X are necessary.

Limitations

One of the limitations of 802.1X is that the authenticator, an AP in a wireless network, is never authenticated by the client (a violation of item (i).) Even when the authentication protocol running on top of EAP provides mutual authentication, that occurs between the client and the authentication server. A potential man-in-the-middle attack is shown in [16].

[4]EAP Over LANs.

The fact that EAP supports multiple authentication protocols gives the advantage of extending the mechanisms already adopted by a company or university to the wireless environment. However, this comes at a cost. Different installations implementing different protocols make client mobility difficult to support. There is clearly a trade-off between flexibility and interoperability (feature (iv).)

The 802.1X standard makes optional the generation of session keys (this is left to the authentication protocol) and leaves the verification services to be implemented by the link layer. Drawbacks of this approach include redundant standards, interoperability problems, and attacks caused by missing properties, such as replay detection (section 6.)

Providing IP addresses as a result of the authentication process can be used to provide different views of the network, what we considered to be part of a flexible authorization mechanism (item (ii).) As defined, the 802.1X authentication runs before the client gets assigned an IP address. In order to provide an IP address based on authentication results, a mechanism such as the one described in [10] has to be used. In this case, when a client is successfully authenticated, using TLS over EAP for example, the AP saves authentication results locally. These results are appended by the AP (acting as a relay) to DHCP requests sent by the client. The DHCP server may use this information to select an address from the appropriate pool. Drawbacks are that not only do APs need to save these authentication results but they also need to be interpreted by the DHCP server.

Overall, we consider 802.1X not to provide a good solution. First, it involves far too many protocols and encourages incompatibility between domains, making it painful for mobile users to handle more than a couple networks. Second, its integration with 802.11 is poor and has been shown to be vulnerable to various attacks.

5. PUBLIC-KEY-BASED SECURE INTERNET ACCESS

5.1 Architecture

This section describes a two-protocol architecture that follows the ideas presented in section 2. The protocol layering is shown in figure 1, and both protocols are implemented in the clients and access points (the possible scenarios are further discussed in section 5.6.)

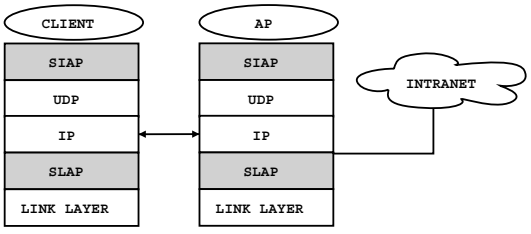

Figure 1: Protocol stack.

The process starts by the SIAP (*Secure Internet Access Protocol*) client performing an authentication handshake with the SIAP server in the access point. This three-message handshake provides mutual authentication and supplies the client with fresh session keys, tied to a given IP address selected by the SIAP server. Therefore, by the end of the authentication run the client has a security state ($MACaddress$, IP, $sessionkeys$), which is also known by the AP. This state is then passed from SIAP to SLAP in both client and AP.

SLAP, the *Secure Link Access Protocol*, is a protocol located just above the link layer, intercepting and processing all incoming and outgoing frames. SLAP can be seen as SIAP's agent over link-layer frames, providing confidentiality, sender authentication, integrity, and replay detection. These services use the session keys negotiated by SIAP. In order to break the dependency cycle discussed previously, packets containing SIAP messages destined to that AP are the only ones that are not processed by SLAP.

5.2 SIAP

In SIAP, every client and access point has a public key signed by a known Certification Authority (CA). This signature also ties the key pair to the DNS name of the host. For example, in the domain *mydomain.com*, a client's laptop could have a key tied to a subject name *tupi.mydomain.com* while each access point would be named *apX.mydomain.com*. The current implementation uses 1024-bit RSA keys.

Handshake

Figure 2: SIAP handshake.

The SIAP handshake is illustrated in figure 2. The first step is optional and performed by the client. It sends a `SIAP_SERVER_ID_REQUEST` message, using source IP 0.0.0.0. This message is sent as an IP broadcast. The client may choose not to send this message, waiting for an advertisement from the server.

Either periodically or as a result of the reception of a `SIAP_SERVER_ID_REQUEST` message, a server sends a `SIAP_SERVER_ID_RESPONSE` message (step 2 in the figure). If in response to a client request, this message is sent as an IP broadcast with the client's MAC address as destination. When sent periodically by the AP, layer-2 broadcast is also used. By receiving these "beacons" from nearby access points, a client can easily identify the network domains reachable at any given time.

They client may collect multiple `SIAP_SERVER_ID_RESPONSE` messages before it decides to connect to a given domain. For example, if beacons from multiple networks are received, the SIAP client may decide on which one to connect to based on information previously configured by the user, stating for example "I prefer to connect to *lab.mydomain.com* than to *mydomain.com*." The client may have two different key pairs for these two networks and always connect to *lab.mydomain.com* when both networks are available.

Having decided the network to connect to, the SIAP client sends a `SIAP_NEW_HOST_REQUEST` message (step 3) to one of the APs from which it has received a beacon. This message specifies the name of the chosen AP and implies that the client has successfully verified the signature over the AP's public key and is requesting an IP address to connect to the network.

If the AP, which verifies the client's public key, grants the client's request, a `SIAP_NEW_HOST_RESPONSE` is sent in step 4. This message supplies the client with all the information it needs to have access to the network.

When the client wants to terminate its connection, it sends the server a `SIAP_DELETE_HOST_REQUEST` message (not shown.) The server then responds with a `SIAP_DELETE_HOST_RESPONSE` message. After this handshake, the client is denied network access.

Message contents and protocol processing

Figure 3 shows the notation used to describe the SIAP messages contents.

v	SIAP protocol version;
m	message identifier (different for each msg);
N_{AP}	a nonce, generated by the AP;
N_c	a nonce, generated by the client;
t_{AP}	a timestamp generated by the AP;
$domain$	network domain;
net_{id}	network identifier;
PK_{AP}	AP's signed public key;
PK_{client}	client's signed public key;
$name_{AP}$	access point's name;
$name_{client}$	client's name;
MAC	client's MAC address;
$lease$	lease period;
IP	client's IP address;
$mask$	network mask;
IP_G	gateway's IP address;
IP_{DNS}	DNS server's IP address;
K	secret;
T	ticket;
$E_C(M)$	msg M encrypted with client's public key;
$S_C(M)$	msg M signed with client's private key;
$E_{AP}(M)$	msg M encrypted with AP's public key;
$S_{AP}(M)$	msg M signed with AP's private key.

Figure 3: Notation used.

• `SIAP_SERVER_ID_REQUEST`

(1): (v, m)

This message is an unencrypted message, containing just protocol information, like version and message identifier.

• `SIAP_SERVER_ID_RESPONSE`

(2): $S_{AP}(v, m, N_{AP}, domain, net_{id}), PK_{AP}$

This message is used by the server to advertise its public key and IP address. The nonce N_{AP} is used by the server to guarantee freshness of responses, avoiding replay attacks. The $domain$ and net_{id} values let the client identify the network to which this access point belongs, without having to extract any information from the signed public key provided by the AP[5]. The value net_{id} is used to differentiate between networks under the same DNS domain that have different IP prefixes.

When receiving this message, the client verifies the public key sent by the access point by checking its signature using the public key of the correspondent certification authority (certification is discussed in more detail in the next section.) The client then uses the AP's public key to verify the SIAP message signature. The client also verifies whether the subject name matches the network domain and if it identifies an access point.

• `SIAP_NEW_HOST_REQUEST`

(3): $S_C(v, m, N_c, N_{AP}, name_{AP}, MAC), PK_{client}$

The client extracts the nonce received in the `SIAP_SERVER_ID_RESPONSE` message and adds it to the request. It also sends its own nonce, used to authenticate the server and also match requests and responses. The request also contains the name of the AP, necessary to avoid replay attacks.

In response to this message, the server verifies the public key sent by the client and the message signature. The server also verifies if the name in the request matches its own name and the nonce value.

• `SIAP_NEW_HOST_RESPONSE`

(4): $S_{AP}(v, m, N_c, name_{Client}, IP, lease, IP_G, mask, IP_{DNS}, E_C(K), T)$

The server extracts the nonce sent by the client in the request and copies it into the response message. The response also contains the IP address selected for the client, the lease time in seconds, the network mask, the default gateway's IP address, the DNS server's IP address and a secret K, which both client and AP use to generate the session keys to be used by SLAP. The ticket value is explained together with the `SIAP_TICKET` message.

The message includes a lease time period by which time the access will time out if not extended. Before the lease expires, the client sends another `SIAP_NEW_HOST_REQUEST` message to extend it.

• `SIAP_DELETE_HOST_REQUEST`

The message is sent by the client to cause a secure disconnection prior to the end of the lease period. This action is relevant to the host when it is being charged based on access time or when the number of concurrent accesses is limited (or just to be a good citizen.) The client authenticates this message by computing a MAC using a key derived from the secret negotiated during the authentication handshake. The server acknowledges the client by sending a `SIAP_DELETE_HOST_RESPONSE` message.

• `SIAP_TICKET`

(5) : (v, m, T)
$T : S_{AP2}(t_{AP}, domain, net_{id}, N_C, MAC, IP, lease, E_{AP2}(K))$

The ticket (T) is the value received in step (4) and is used by the client to propagate its state to other access points in the network (section 5.4.) To create the ticket, the AP uses a second key pair, shared between all APs in the domain. This key pair is used to encrypt the secret K given to the client and sign the message. The ticket contains information relative to the authentication handshake, such

[5]The subject name in the signed key should have the network domain as a suffix.

as a timestamp generated by the server, the nonce used by the client, its MAC and IP addresses, and the granted lease.

Certification

The mutual authentication provided by SIAP depends on the ability of clients and servers to verify signatures over public keys. The ideal solution, necessary to achieve complete interoperability between domains, is to have a deployed public-key infrastructure (PKI). However, with the lack of such mechanism, our architecture can be implemented locally on a network provided with a single-domain certification authority (SDCA). In this case, all the mobile computers have their names tied to public keys signed by the local SDCA and know its public key, needed to authenticate the local access points. We believe our architecture to be deployable in a bottom-up manner, as SDCAs can be integrated into bigger authentication domains. At first, multi-homed users need a signed public key for each network they are wiling to use.

5.3 SLAP

After the client is authenticated, the generated keys are passed from SIAP to SLAP. For each client, SLAP receives a key to be used by the confidentiality service and another key to be used by the MAC algorithm. After this security state is set in both client and AP, all frames receive the SLAP header and can be encrypted and authenticated.

The SLAP header includes the protocol version, a 64-bit counter, a type value, and a MAC (authenticator) value. The counter value is used to uniquely identify each message sent by a SLAP entity and is used by the replay detection mechanism.

The type value is the 16-bit type and length values present in the Ethernet and 803.3 headers, respectively. It is common for 802.11 device drivers to provide the kernel with an Ethernet interface, for ease of deployment. During the processing of every outgoing frame, SLAP changes the protocol field in the Ethernet header to its own value and saves the old value in its header. If the frame is an 802.3 frame, the length is incremented by the size of the SLAP header. During frame reception this process is reverted.

Encryption and authentication

After an outgoing packet receives the SLAP header, it is first encrypted. SLAP uses AES [8] in counter mode (CTR [3]) to encrypt the SLAP packet, which includes the IP header and payload. AES was chosen for several reasons. First, it was adopted as a federal standard, which gives some confidence about the strength of the algorithm. Second, AES supports 128-, 192-, and 256-bit keys and was designed to provide fast implementation in both hardware and software [8].

The CTR mode was chosen for two main reasons. First, it provides high parallelism, as each plaintext block can be encrypted independently. Second, as opposed to other modes, the plaintext processed by CTR does not need to be of length multiple of the block size. However, CTR requires that the counter value used to encrypt each block be never reused. SLAP achieves this by constructing a 128-bit per-block counter value by concatenating the sender's MAC address (48 bits), the message counter (64 bits), and a block counter (16 bits), which is different for each block inside a frame.

After encryption, each frame processed by SLAP is authenticated by calculating a message authentication code that also covers the SLAP header. The result is copied into the MAC header field, allowing the intended receiver to verify frame integrity and the identity of the sender. This makes it hard for unauthorized users to create messages on behalf of another, authenticated user, without knowledge of the MAC key.

SLAP uses HMAC-MD5 [14] as the authentication algorithm. HMAC-MD5 has been chosen because it enables fast implementation, as it is based in the MD5 [19] hashing function. It preserves the original performance of the underlying hash function without incurring significant degradation, and has widely available and unrestricted implementation [14]. The overall process executed by SLAP is illustrated in figure 4.

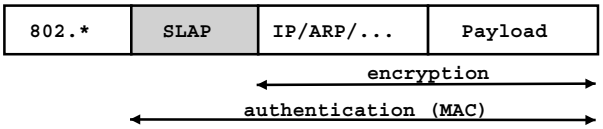

Figure 4: SLAP frame and applied services.

5.4 Mobility and State propagation

When the SIAP client executes the authentication handshake with an AP, it receives a *ticket* in the `SIAP_NEW_HOST_RESPONSE` message. This ticket is created by the AP and includes the secret K, its IP address, and lease duration. In order to encrypt the secret K and sign the contents of the message, all the APs in the network share a second public key pair. In the current implementation, a 2048-bit RSA pair is used.

As the client moves, it propagates its security state by sending the ticket to other APs in the network. Using the shared public key pair, the APs test the validity of the ticket and configure the client state. The advantages of this client-driven state propagation are twofold. First, as the client is responsible for propagating its security state, there is no need for an inter-AP protocol. Second, this mechanism avoids the propagation of state to access points that are never used by the client.

It should be noted that this client-driven state propagation mechanism handles only intranet mobility. In order to have connectivity in networks with different IP address prefixes, the client needs to perform the SIAP handshake multiple times. To reduce handoff latency, a client can acquire an IP address in a second network before it loses connectivity with its current provider. As SIAP does not support transport layer connection migration, mechanisms such as the one proposed by Snoeren *et al.*[21] might still be useful.

5.5 Authenticate and Associate

SIAP and SLAP can be used in wireless networks to provide secure association and avoid the attacks discussed in section 3. However, some changes need to be made to the current 802.11 standard. Assume that the association handshake is modified to use a key shared between client and AP and provide mutual authentication. This could be achieved by using an association key generated from the secret K and by having the association messages use the SLAP services

implemented in the wireless card. This scenario is shown in figure 5.

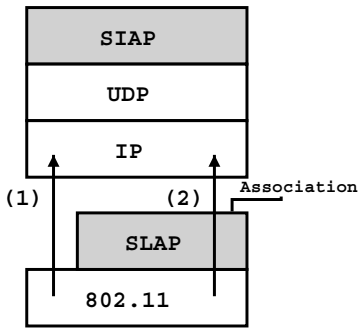

Figure 5: SIAP and SLAP integration.

Before a client gets associated with an AP, it needs to set up an association key. This is done by executing the SIAP handshake, as described previously. The client can then associate with the AP it authenticated with by performing the secure association handshake. As seen in figure 5, SIAP messages destined to the AP are not processed by the SLAP module and use path (1), enabling the client to authenticate before it is associated with the AP. All other frames, including association messages, follow path (2).

When moving from one AP to the other, the SIAP handshake is not necessary for the duration of the ticket given to the client (IP lease time.) Before the wireless card can perform a handoff and associate with a second AP, the SIAP client needs to propagate its state using the ticket, as this second AP still does not know the association key for that given client.

With this construction, an attacker has to guess the association key before it can perform a disassociation attack. Creating a fake base station to get clients associated with it now becomes as hard as breaking the ticket in order to derive the association key.

5.6 Flexibility

By making SLAP link-layer independent, the architecture described in this paper can be used in non-802.11 networks. Being link-layer independent enables SIAP to be used with any LAN technology, including the whole IEEE 802 family. This not only allows for interoperable systems but also avoids the need for multiple standards.

The placement of SLAP in the protocol stack also makes it possible to place the SLAP/SIAP entities in different locations inside the intranet. In order to provide secure association and AP authentication in an 802.11 network, the protocols can be implemented at the access points, as shown in figure 6.

However, there is no need to add a SIAP server and a SLAP entity to every Ethernet switch, even though this could improve fault tolerance and load balancing. Instead, a specialized bridge could be used behind all switches in the network. Another option is to implement the protocols in the default gateway. Both scenarios are shown in figure 7. The placement of the SIAP servers in the network is completely transparent to the client, which just sees SIAP messages coming from IP addresses in the network and all its frames being sent and received with the SLAP header.

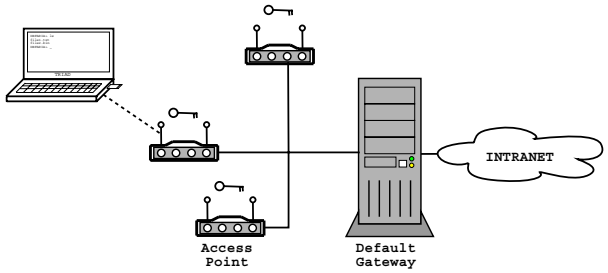

Figure 6: SIAP and SLAP in a wireless network.

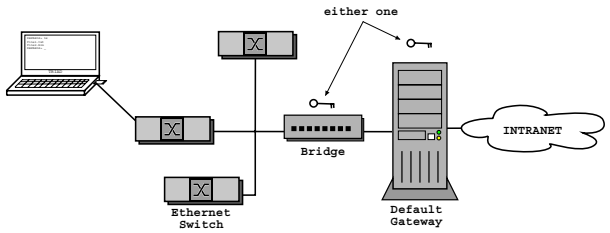

Figure 7: SIAP and SLAP in an Ethernet network.

As the encryption performed by WEP is part of the link-layer protocol, the access points are necessarily the other endpoint of the secure channel, making it impossible to implement the scheme shown in figure 7.

5.7 Preliminary results

Prototypes for SLAP and SIAP have been implemented in Linux and current results are promising. With a personal computer working as an access point, SLAP services can be provided in software with relatively little overhead. Our test bed is composed of a client laptop[6] and a desktop computer[7] that acts as an access point. These machines were connected through a FastEthernet link, to increase the available bandwidth.

SLAP overhead was measured to vary between 50μs and 330μs in the client machine and between 10μs and 170μs in the access point, varying the frame size between 10 and 1450 bytes. The total overhead, which takes into account processing in both client and AP, can be as high as 460μs in one direction, increasing the round-trip time (RTT) by almost 1 millisecond. However, this overhead has little effect over representative TCP connections.

We performed several 50-megabyte file transfers from servers with RTTs varying between 1ms and 40ms and measured the total download time to increase between 7% and 17%. We found these results encouraging for several reasons. First, we expect average packet size to be small, as supported by wireless LANs measurements made by Tang et al.[23] and by the popularity of web-browsing and session-oriented applications[24]. Second, we consider a dedicated 100Mbps Ethernet connection and 40μs RTT to be far better than the common case. Finally, SLAP overhead should become negligible with the use of firmware implementations.

[6]333-MHz Intel Pentium II, 64 MB RAM.
[7]900-MHz AMD Duron, 256 MB RAM.

The SIAP handshake was measured to terminate in hundreds of milliseconds, mainly due to the private key operations incurred by SIAP. We are currently working on reducing the number of signatures performed by the APs while making SIAP more robust to DoS attacks.

6. OTHER DoS ATTACKS

6.1 Pre-Authentication Attacks

With SIAP, there is no action the client has to perform before initiating the authentication handshake. However, some security architectures require the client to execute some configuration steps (such as obtain an IP address) before it can go through the authentication process. These steps performed before authentication are usually insecure and therefore susceptible to DoS attacks. For example, solutions that use DHCP to hand out IP addresses before authentication takes place are vulnerable to an attacker that acquires all available IP addresses using MAC address spoofing. PANS[5] and other solutions based on the establishment of secure network-layer tunnels seem vulnerable to these attacks.

6.2 Attacks on Authorization

Establishing session keys for an authenticated client can be seen as part of the authorization process, as these keys are used to restrict the client's use of the system. Giving the same key to multiple clients can be considered a flaw in authorization and is clearly insecure. However, this is exactly how multicast messages are usually handled in a public network. When authentication takes place in a shared medium network, clients usually receive a random session key and also a multicast key, shared between all clients in the same LAN. Using 802.11 as an example, the AP uses this multicast key to encrypt all frames to be multicasted inside the cell. Although clients may not be allowed to send frames using this key, there is no way to detect their misbehavior. A client could, for example, perform a DoS attack by generating fake ARP responses that point to an inexistent gateway. Other protocols that rely on multicast messages are equally vulnerable.

Instead of eliminating the multicast key altogether and creating a copy of each frame for each client associated with the AP, SLAP adopts a different approach. Protocols that are vulnerable to attacks are handled using the clients' session keys. If needed, multiple copies of the frame are created. However, despite their use of link-layer multicast, protocols such as ARP include in their messages the MAC address of the requester. When forwarded by the AP, ARP frames are unicasted using the clients' session.

6.3 Attacks on Verification

Attacks on access verification are also possible. They can be generated based on the lack of important services, such as replay detection or sender authentication, or based on spoofed data not covered by these services. PANS and WEP, for example, do not implement a replay detection service, enabling an attacker to flood a network with old packets and perform other, more clever attacks. This class of attacks also renders accurate accounting impossible to implement.

There is also a trade-off about the layer at which these services should be implemented. WEP and SLAP cover all frames sent over the wireless, while PANS and IPSec-based solutions only process IP packets. In the latter approach, ARP and other protocols that do not run over IP packets are vulnerable to spoofing attacks, which can be easily implemented.

7. RELATED WORK

Apart from IEEE 802.11/802.1X, other solutions address the problem by performing security-related services at the IP level. Examples include the CHOICE network[5, 17] and solutions based on IPSec [12]. The placement of services at the IP layer renders these solutions vulnerable to some of the attacks described in section 6 while making it unfeasible to implement a secure association mechanism at the link layer.

The CHOICE network provides secure Internet access using Microsoft Passport servers as online authenticators and PANS to provide authorization and accounting [17]. Authentication is performed over SSL and session keys are used to encrypt and tag IP packets. CHOICE has different objectives when compared to SIAP. Bahl *et al.*[5] advocate the use of CHOICE outside corporate environments, in places such as cafeterias and airport lounges, while leaving corporations with their already deployed solutions (such as shared key WEP-based authentication.) The CHOICE client dynamically switches between these two environments[17]. Finally, PANS processes IP packets and does not implement a replay detection mechanism, being vulnerable to flooding attacks, as discussed in the previous section.

8. CONCLUSION

Due to wireless-specific characteristics we have identified several services that need to be provided by an access control mechanism when plugged into such networks. As wireless networks are deployed, users will expect mobility support even when presented with effective access control. Providing mobility has been the major virtue of wireless computing and we have specified an authentication architecture that is mobility-aware and prepared to support connectivity migration between domains.

Such a mobility support is made easier by the proposed protocols. While SIAP provides a single authentication handshake based on RSA keys, SLAP implements a link-layer independent access verification mechanism using AES-CTR and HMAC-MD5 to provide confidentiality and message authentication, respectively. Concerning authentication, even though global interoperability is dependent upon an available PKI, single-domain certification authorities can be used as a temporary solution.

We have identified several DoS attacks related to authentication, authorization, and access verification. We have shown that most of the attacks discussed are caused by the lack of important services, such as replay detection and access point authentication, or by wrong assumptions made about the network infrastructure, such as the one made by 802.1X about secure association in 802.11 networks. Combined, these factors have made possible for attackers to set up rogue access points and and perform man-in-the-middle attacks in 802.11/802.1X networks.

We have also proposed an architecture, composed of the SIAP and SLAP protocols, that solves the problem by coalescing essential services in a secure way. Making very few assumptions about the environment, SIAP provides clients with access point authentication and, together with SLAP,

enables the implementation of a secure association service. By coalescing authentication and IP address assignment, SIAP avoids DoS attacks effective against DHCP servers. Attacks based on the use of shared keys for handling multicast frames are disabled by SLAP by the use of protocol-specific handlers, such as the one described to thwart ARP spoofing. Finally, we have described how SLAP prevents attacks related to access verification by implementing message authentication and replay detection.

In summary, the 802.1X specification is centered on an abstraction that easily maps to switched wired networks. However, insecure association and the ability to mount rogue access points have made difficult the mapping of such abstraction to wireless networks. SIAP addresses the problem in the reverse direction: providing a solution for the access control problem suitable to the wireless environment can be easily made to work in a wired network. The ability to place SIAP/SLAP modules in different network entities provides a secure solution for wireless installations while avoiding the burden of implementing these mechanisms in every Ethernet switch. Moreover, preliminary results show that a software implementation of the proposed protocols is efficient enough to secure current 802.11b networks.

9. REFERENCES

[1] LAN MAN Standards Committee of the IEEE Computer Society. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Technical Report 1999 Edition, IEEE Standard 802.11, 1999.

[2] LAN MAN Standards Committee of the IEEE Computer Society. Standard for Port based Network Access Control. Technical Report Draft P802.1X/D11, IEEE Computer Society, Mar. 2001.

[3] NIST, Special Publication: Recommendation for Block Cipher Modes of Operation - Methods and Techniques. SP 800-38A. Dec. 2001.

[4] W. A. Arbaugh, N. Shankar, and Y. C. J. Wan. Your 802.11 wireless network has no clothes. March 2001.

[5] P. Bahl, S. Venkatachary, and A. Balachandran. Secure wireless internet access in public places. In *Proc. of the IEEE Conference on Communications (ICC)*, Helsinki, Finland, June 2001.

[6] L. Blunk and J. Vollbrecht. PPP Extensible Authentication Protocol (EAP), RFC 2284, IETF. Mar. 1998.

[7] N. Borisov, I. Goldberg, and D. Wagner. Intercepting mobile communications: The insecurity of 802.11. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking - Mobicom'01*, pages 180-189, July 2001.

[8] J. Daemon and V. Rijmen. AES Proposal: Rijndael. Available from http://csrc.nist.gov/encryption/aes/, March 1999.

[9] T. Dierks and C. Allen. The TLS Protocol - Version 1.0, RFC 2246, IETF. Jan. 1999.

[10] R. Droms and J. Schnizlein. RADIUS Attributes Sub-option for the DHCP Relay Agent Information Option, Internet Draft, IETF. Feb. 2002.

[11] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. In *Eighth Annual Workshop on Selected Areas in Cryptography*, August 2001.

[12] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol, RFC 2401, IETF. Nov. 1998.

[13] J. Kohl and C. Neuman. The Kerberos Network Authentication Service (V5), RFC 1510, IETF. Sept. 1993.

[14] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication, RFC 2104, IETF. Feb. 1997.

[15] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, Oct. 1996.

[16] A. Mishra and W. A. Arbaugh. An initial security analysis of the IEEE 802.1X standard. Technical Report CS-TR-4328,UMIACS-TR-2002-10, University of Maryland, Feb. 2002.

[17] A. Miu and P. Bahl. Dynamic host configuration for managing mobility between public and private networks. In *Third USENIX Simposium on Internet Technologies and Systems*, San Francisco, CA, Mar. 2001.

[18] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS), RFC 2865, IETF. June 2000.

[19] R. Rivest. The MD5 message-digest algorithm, RFC 1321, IETF. Apr. 1992.

[20] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120-126, Feb. 1978.

[21] A. C. Snoeren and H. Balakrishnan. An end-to-end approach to host mobility. In *Proceedings of the Sixth ACM International Conference on Mobile Computing and Networking - Mobicom'00*, pages 155-166, Boston, MA, Aug. 2000.

[22] A. Stubblefield, J. Ioannidis, and A. D. Rubin. Using the Fluhrer, Mantin, and Shamir attack to break WEP. Technical Report TD-4ZCPZZ, AT&T Labs Research, Aug. 2001.

[23] D. Tang and M. Baker. Analysis of a local-area wireless network. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking - Mobicom'00*, pages 1-10, Boston, MA, USA, Aug. 2000.

[24] K. Thompson, G. Miller, and R. Wilder. Wide-area internet traffic patterns and characteristics. *IEEE Network*, 11(6):10-23, Nov. 1997.

[25] J. Walker. Unsafe at any key size: An analysis of the WEP encapsulation. Technical Report 03628E, IEEE Standards 802.11 Committee, March 2000.