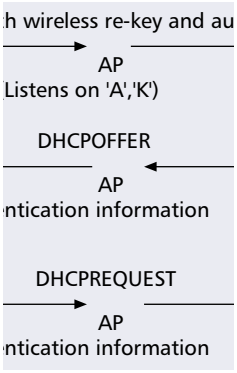
YOUR 802.11 WIRELESS NETWORK HAS NO CLOTHES

WILLIAM A. ARBAUGH, NARENDAR SHANKAR, AND Y. C. JUSTIN WAN
UNIVERSITY OF MARYLAND
KAN ZHANG, HEWLETT-PACKARD LABORATORIES



Many firms, based on vendor literature, believe that the security provided by their deployed wireless access points is sufficient to prevent unauthorized access and use. Unfortunately, nothing could be further from the truth.

ABSTRACT

The explosive growth in wireless networks over the last few years resembles the rapid growth of the Internet within the last decade. To protect internal resources, organizations usually purchased and installed an Internet firewall. We believe that the currently deployed wireless access points present a larger security problem than the early Internet connections. A large number of organizations, based on vendor literature, believe that the security provided by their deployed wireless access points is sufficient to prevent unauthorized access and use. Unfortunately, nothing could be further from the truth. While the current access points provide several security mechanisms, our work combined with the work of others show that *all* of these mechanisms are completely ineffective. As a result, organizations with deployed wireless networks are vulnerable to unauthorized use of, and access to, their internal infrastructure. In this article we present a novel solution that requires no changes or additions to any deployed wireless equipment, and is easily deployed and transparent to end users.

INTRODUCTION

Organizations are rapidly deploying wireless infrastructures based on the IEEE 802.11 standard [1]. Unfortunately, the 802.11 standard provides only limited support for confidentiality through the Wired Equivalent Privacy (WEP) protocol, which contains significant flaws in design [2–5]. Furthermore, the standards committee for 802.11 left many difficult security issues such as key management and a robust authentication mechanism as open problems. As a result, many of the organizations deploying wireless networks use either a permanent fixed cryptographic variable, or key, or no encryption whatsoever. This fact, coupled with the fact that wireless networks provide a network access point for an adversary (potentially beyond the physical security controls of the organization), creates a significant long-term security problem. Compounding this is the fact that the access control mechanisms available with currently deployed access points[1] contain serious flaws; an adversary can easily subvert them.

Organizations over the last few years have expended considerable effort to protect their internal infrastructures from external compromise. As a result, organizations have canalized their external network traffic through distinct openings protected by firewalls. The idea is simple. By limiting external connections to a few well-protected openings, the organization can better protect itself. Unfortunately, the deployment of a wireless network opens a "back door" into the internal network that permits an attacker access beyond the physical security perimeter of the organization. As a result, the attacker can implement the "parking lot" attack, where the attacker sits in the organization's parking lot and accesses hosts on the internal network. Ironically, in some cases, the existence of the firewall may make the organization's hosts more vulnerable to the attacker because of the mistaken premise that the internal hosts are immune from attack and potential compromise. Finding an effective solution to these problems is difficult since all of the security mechanisms are contained in the firmware of the wireless equipment (e.g., PCMCIA card and access points), and virtual private networks (VPNs) are not realizable in all environments.

This article describes the flaws in the two access control mechanisms (*MAC address control lists* and the proprietary *Closed Network scheme*) that exist in some access points and a simple eavesdropping attack against the 802.11 specified shared key authentication mechanism. Exploiting these flaws, when encryption is not enabled, permits an adversary immediate access to the wireless network as well as the organization's local area network. The use of encryption prevents an adversary from gaining immediate access, but combining our attacks with the weaknesses found in WEP by others provides such access [4].

The next section presents a short overview of the network configuration modes supported in the 802.11 wireless standard. This is followed by an overview of the 802.11 security mechanisms and a proprietary extension for access control. The next section describes attacks against the only two access control mechanisms available in most current access points, and an attack against the 802.11 standard shared key authentication

[1] *This article was written before the IEEE 802.11 Task Group on Security (TGi) began significant changes to WEP, and before products supporting IEEE 802.1X and EAP/TLS were available.*

1536-1284/02/$17.00 © 2002 IEEE

mechanism. Finally, we conclude the article with a discussion of a higher-layer authentication and key management scheme that can easily be applied to almost any currently deployed wireless network without requiring the purchase of new hardware or the installation of new firmware in all of the organizations wireless devices.

802.11 Wireless Networks

802.11 wireless networks operate in one of two modes: *ad hoc* or *infrastructure*. The IEEE standard defines the ad hoc mode as independent basic service set (IBSS) and the infrastructure mode as basic service set (BSS). In the remainder of this section, we explain the differences between the two modes and how they operate.

In ad hoc mode, each client communicates directly with the other clients within the network. Ad hoc mode is designed so that only clients within transmission range of each other (within the same cell) can communicate. If a client in an ad hoc network wishes to communicate outside of the cell, a member of the cell must operate as a gateway and perform routing.

In infrastructure mode, each client sends all of its communication to a central station or access point (AP). The access point acts as an Ethernet bridge and forwards the communications onto the appropriate network: either wired or wireless.

Prior to communicating data, wireless clients and APs must establish a relationship, or an association. Only after an association is established can two wireless stations exchange data. In infrastructure mode, the clients associate with an AP. The association process is a two-step process involving three states:

- Unauthenticated and unassociated
- Authenticated and unassociated
- Authenticated and associated

To transition between the states, the communicating parties exchange messages called *management frames*.

A client finds a network using a very simple procedure. All APs transmit a *beacon* management frame at fixed interval. To associate with an AP and join a BSS, a client listens for beacon messages to identify the APs within range. The client then selects the BSS to join in a vendor-independent manner. For instance, on the Apple Macintosh, all of the network names (or service set identifiers, SSIDs) usually contained in the beacon frame are presented to the user so that they may select the network to join. A client may also send a probe request management frame to find an AP affiliated with a desired SSID. After identifying an AP, the client and AP perform mutual authentication by exchanging several management frames. The two standardized authentication mechanisms are described later. After successful authentication, the client moves into the second state, *authenticated and unassociated*. Moving from the second state to the third and final state, *authenticated and associated*, involves the client sending an association request frame, and the AP responding with an association response frame. After following this process, the client becomes a peer on the wireless network.

802.11 Standard Security Mechanisms

The 802.11 standard provides several mechanisms intended to provide a secure operating environment. In this section we describe each of these as well as the Closed Network access control mechanism.

Wired Equivalent Privacy Protocol

The WEP protocol was designed to provide confidentiality for network traffic using the wireless protocol. The details of the algorithm used for WEP are beyond the scope of this article. However, work by Walker *et al.* and Fluhrer *et al.* demonstrates that WEP, without dynamic key management, provides limited confidentiality and possible misuse of the network.

Open System Authentication

Open system authentication is the default authentication protocol for 802.11. As the name implies, open system authentication authenticates anyone who requests authentication. Essentially, it provides a null authentication process. Experimentation has shown that stations do perform mutual authentication using this method when joining a network, and our experiments show that the authentication management frames are sent in the clear even when WEP is enabled.

Shared Key Authentication

Shared key authentication uses a standard challenge and response along with a shared secret key to provide authentication. The station wishing to authenticate, the initiator, sends an authentication request management frame indicating that they wish to use "shared key" authentication. The recipient of the authentication request, the responder, responds by sending an authentication management frame containing 128 octets of challenge text to the initiator. The challenge text is generated by using the WEP pseudo-random number generator (PRNG) with the "shared secret" and a random initialization vector (IV). The IV is always sent in the clear as part of a WEP protected frame. Once the initiator receives the management frame from the responder, they copy the contents of the challenge text into a new management frame body. This new management frame body is then encrypted with WEP using the "shared secret" along with a new IV selected by the initiator. The encrypted management frame is then sent to the responder. The responder decrypts the received frame and verifies that the 32-bit cyclic redundancy check (CRC) integrity check value (ICV) is valid, and the challenge text matches that sent in the first message. If they do, authentication is successful. If the authentication is successful, the initiator and responder switch roles and repeat the process to ensure mutual authentication.

The value of the status code field is set to zero when successful and to an error value if unsuccessful. The element identifier identifies that the challenge text is included. The length field identifies the length of the challenge text and is fixed at 128. The challenge text includes the random challenge string.

Open system authentication is the default authentication protocol for 802.11. As the name implies, open system authentication authenticates anyone who requests authentication. Essentially, it provides a null authentication process.

Key management is a misnomer with respect to 802.11 as it is left as an exercise for vendors. As a result, only a few of the major vendors have implemented any form of key management or key agreement in their high-end products.

Closed Network Access Control

One vendor has defined a proprietary access control mechanism called "Closed Network" [5]. With this mechanism, a network manager can use either an open or a closed network. In an open network, anyone is permitted to join the network. In a Closed Network, only those clients with knowledge of the network name, or SSID, can join. In essence, the network name acts as a shared secret.

Access Control Lists

Another mechanism used by vendors (but not defined in the standard) to provide security is the use of access control lists based on the Ethernet MAC address of the client. Each AP can limit the clients of the network to those using a listed MAC address. If a client's MAC address is listed, they are permitted access to the network. If the address is not listed, access to the network is prevented.

Key Management

Key management is a misnomer with respect to 802.11 since it is left as an exercise for vendors. As a result, only a few of the major vendors have implemented any form of key management or key agreement in their high-end products. Unfortunately, none of the vendors provide sufficient information to determine the level of assurance provided by their product.

The 802.11 standard does, however, provide for two methods for using WEP keys. The first provides a window of four keys. A station or AP can decrypt packets enciphered with any one of the four keys. Transmission, however, is limited to one of the four manually entered keys — the *default* key. The second method is called a key mappings table. In this method, each unique MAC address can have a separate key. The size of a "key mappings table" should be at least ten entries according to the 802.11 specification. The maximum size, however, is likely chip-set dependent. The use of a separate key for each user mitigates the cryptographic attacks found by others, but enforcing a reasonable key period remains a problem as the keys can only be changed manually.[2]

Weaknesses in Current Access Control Mechanisms

This section describes the weaknesses in the access control mechanisms of currently deployed wireless network APs.

Closed Network Access Control Mechanism

In practice, security mechanisms based on a shared secret are robust provided the secrets are well protected in use and when distributed. Unfortunately, this is not the case with a proprietary access control mechanism. Several management messages contain the network name, or SSID, and these messages are broadcast in the clear by APs and clients. The actual management message containing the SSID depends on the vendor of the AP and the firmware version. The result, however, is that an attacker can easily sniff the network name, determining the shared secret and gaining access to the "protected" network. This flaw exists even with WEP enabled because the management messages are broadcast in the clear.

Ethernet MAC Address Access Control Lists

In theory, access control lists provide a reasonable level of security when a strong form of identity is used. Unfortunately, this is not the case with MAC addresses for two reasons. First, MAC addresses are easily sniffed by an attacker since they *must* appear in the clear even when WEP is enabled, and second, most of the wireless cards permit the changing of their MAC address via software. As a result, an attacker can easily determine the MAC addresses permitted access via eavesdropping, and then subsequently masquerade as a valid address by programming the desired address into the wireless card, bypassing the access control and gaining access to the "protected" network.

Shared Key Authentication Flaw

The current protocol for shared key authentication is easily exploited through a passive attack by the eavesdropping of one leg of a mutual authentication. The attack works because of the fixed structure of the protocol (the only difference between different authentication messages is the random challenge), and previously reported weaknesses in WEP.

The attacker first captures the second and third management messages from an authentication exchange. The second message contains the random challenge in the clear, and the third message contains the challenge encrypted with the shared authentication key. Because the attacker now knows the random challenge R, the encrypted challenge *ciphertext*, C, and the public IV, the attacker can derive the pseudo-random stream produced using WEP, $WEP_{PR}^{K,IV}$, with the shared key, K, and the public initialization variable, IV, using Eq. 1.

$$WEP_{PR}^{K,IV} = C \oplus R \quad (1)$$

The size of the recovered pseudo-random stream will be the size of the authentication frame, because all elements of the frame are known: algorithm number, sequence number, status code, element id, length, and challenge text. Furthermore, all but the challenge text will remain the same for *all* authentication responses. The attacker now has all of the elements to successfully authenticate to the target network, without knowing the shared secret K.

The attacker requests authentication of the AP it wishes to associate/join. The AP responds with an authentication challenge in the clear. The attacker then takes the random challenge text, R, and the pseudo-random stream, $WEP_{PR}^{K,IV}$, and computes a valid authentication response frame body by *XOR-ing* the two values together using Eq. 2.

$$C = WEP_{PR}^{K,IV} \oplus R \quad (2)$$

The attacker then computes a new ICV as described in Borisov *et al.* Now, the attacker responds with a valid authentication response message, and he/she associates with the AP and joins the network.

[2] *It must be noted that with the advent of 802.1X, this is no longer true. However, the TGi continues to debate the details of how key management will work within the 802.1X framework. As a result, current support for rekeying using 802.1X is not widely available.*

New Vendor Solutions and New Standards

Several proprietary solutions have been released recently. The advantage of these solutions is that the link layer key is renewed on a per-user per-session basis. However, the key does not change within the session. Even when the user session lasts for weeks, the link layer key does not change. In other words, there is no *timed key management protocol*. Also, key management is tied to authentication, because the keys are renewed only when the user authenticates. In addition, these solutions do not provide a proper means of authenticating a *disconnected user*. A disconnected user is an authorized user of the network but has left the network temporarily. As shown earlier, it is mandatory to have WEP enabled to thwart some of the attacks described by us. If WEP is enabled, such a user cannot authenticate because he/she might not have the correct window of WEP keys.

Fortunately, the 802.11 standards body is currently working on significant improvements to the standard. But all of the changes proposed by the standards body require firmware updates for all clients and APs. Furthermore, the changes are not guaranteed to work with all vendor implementations; processing power or other hardware limitations may prevent the vendor from fully implementing the proposed changes.

The task group on security (TGi) for the 802.11 standards body has been working for the past few months on an interim solution to the known problems with WEP. Their approach is to design a solution that can be implemented in the firmware of the medium access controller chip of most (not all) vendors' chipsets, or through the use of the host processor (client or AP). There are four key elements to the current draft design:

- Dynamic key management
- The addition of message integrity via a message authentication code
- Restructuring the manner in which the initialization vector and the key are combined to avoid the weak keys found by Fluhrer *et al.*
- Defining a new cipher (AES), and specifying MAC message format changes to support upper layer authentication and cipher suite negotiation

The exact details of the current TGi proposal are beyond the scope of this article, and change frequently as the draft converges to a standard [6].

Insight into our Solution

We now describe a solution we have developed at the University of Maryland. Our solution uses DHCP options as a transport mechanism for wireless key management and authentication, provides timed key management, and solves the disconnected user problem.

Design Goals

When we began this work, solutions designed to mitigate the known attacks against WiFi-based networks were proprietary or required the purchase of additional hardware. Our motivation was to find a viable solution that could be easily deployed within an enterprise without requiring the purchase and deployment of additional hardware. Our solution, proposed here, while not optimal, *mitigates* all of the known attacks. We realize that our approach *does not prevent* all of the known attacks since only low-level protocol changes by vendors and the IEEE can prevent all of the known attacks. Our solution provides a significant increase in protection for those organizations that currently cannot obtain vendor-provided upgrades (not all vendor equipment can be upgraded) and do not wish to purchase new equipment.

We have four primary design goals:

- Provide a robust key management for wireless LANs using the DHCP services of the wired LAN
- Work with the existing infrastructure
- Limit complexity (complexity breeds design and implementation errors)
- Solve the timed key management and disconnected user problems

The first goal mitigates most of the known attacks against WEP (MIC attacks cannot be prevented). The second, third and fourth goals not only provide transparency of the security mechanisms to the users, but also reduce the potential for errors in design, implementation, and operation.

Solution Overview

Retrofitting security is never a wise idea. Unfortunately, security architects often find it necessary to do so. The approach used in our solution is to limit the scope of the changes required in a wireless deployment. Therefore, we opted to make minor changes to the infrastructure rather than requiring changes to the APs and the users' wireless cards. In our solution, we:

- Use DHCP authentication for higher-layer authentication [7].
- Use a wireless rekey DHCP option for key management, which when set along with the authentication option can be used to transport the WEP key that is encrypted with a key generated from authentication.
- Solve the disconnected user problem by using a *two-door approach* at the link layer (using two keys, one a long-term WEP key used for entry into the network, and subsequently to authenticate, and another a short-term WEP key, which is used for encrypting wireless traffic). The relatively long-lived WEP key is denoted A, and the key used for communication is denoted K.
- Provide *timed key management* by tying key management and DHCP leases. When a user/station renews its IP address the corresponding WEP key is also renewed (if the wireless rekey option is set).

Key Management Protocol

The wireless network consists of stations (STAs) trying to connect to a wired network using the AP. The DHCP server exists on the wired portion of the network. All link layer traffic is encrypted using K (the current link layer key). Wireless traffic from the STAs is encrypted using K. The AP decrypts the wireless traffic

Retrofitting security is never a wise idea. Unfortunately, security architects often find it necessary to do so. The approach used in our solution is to limit the scope of the changes required in a wireless deployment.

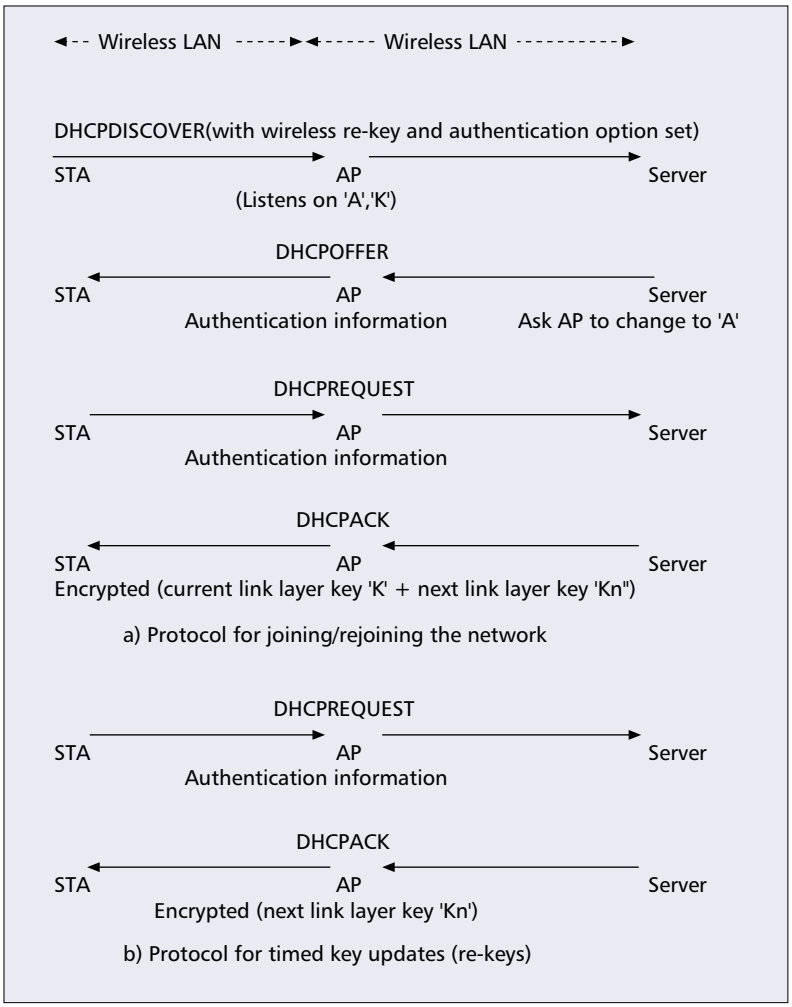


Figure 1. *DHCP authentication and rekey messages frame.*

(because it too has *K*) and forwards the traffic to the wired network.

The idea to mitigate the current WEP flaws is to enforce a small key period for *K*. At the same time valid STAs of the network, who leave the network must be able to obtain the current key (if it has changed) when they rejoin the network. This is accomplished by a *double-door entry mechanism* where the STA gains entry and authenticates itself (using higher-layer DHCP authentication) into the network using *A*. The frequency of use of *A* is significantly smaller than that of *K* and is hence considered to be a key with a longer lifetime. Both the AP and STAs have a window of four WEP keys. They can listen on any of the four keys but can transmit only on one key.

The AP listens on both *K* and *A*. The STAs that are a part of the network have *K* and *A*. The valid STAs who do not have *K* (they previously left the network and are now rejoining) can gain entry and authenticate themselves using *A*. After the STAs have authenticated themselves, they obtain the current link layer key *K* via the DHCP option.

Apart from rejoining the network, regular timed key management takes place for all STAs currently in the network by leveraging DHCP lease expirations. In other words, the current key *K* keeps changing frequently. We use DHCP as a *transport mechanism* for getting the new link layer key K_n.

The basic idea is as follows:

- When the client/STA joins/rejoins the network, it is assigned an IP address and given the current link layer key *K*. The IP address is leased for a particular time period. This time period is set by organizational policy. Apart from this, the STA is also given the next link layer WEP key, K_n.
- All the clients in the network who have the current key renew their IP address (note: the address does not need to change) depending on the lease time and in the process also obtain the new link layer key K_n.

The messages involved with the protocol are shown in Fig. 1:

- The STA sends a DHCPDISCOVER message with the wireless rekey and DHCP authentication options set. The STA transmits the message encrypted with link layer key *A*. The AP can listen on *A* and *K*, and hence forwards the request to the DHCP server on the wired LAN.
- The DHCP server sends a DHCPOFFER message including the authentication information in accordance with the DHCP authentication protocol [7]. We note that the AP's transmission key must be changed to *A* before transmission and back to *K* afterwards.
- The STA transmits a DHCPREQUEST message, which includes the authentication information. The authentication information is again in accordance with the DHCP authentication protocol.
- The DHCP server sends back a DHCPACK message, which includes the authentication information and encrypted current WEP key *K*. The encryption can be done with a shared key as defined by [7]. Also, the DHCP server sends the next WEP key K_n encrypted.

The problem of encrypting the WEP keys with the shared key as defined by [7] is that there is an inherent problem of scaling. Hence, there arises a necessity for a public-key-based authentication mechanism and the use of keys derived from the public-key-based authentication scheme.

Implementation

At the time when we were working on the implementation, none of the available versions of DHCP had the authentication option implemented. Hence, to be able to use authentication functionality in our protocol, we propose a public-key-based and a shared-key-based authentication system, which requires minimal state maintenance at the DHCP server. The public key mechanism also helps us solve the scalability problem.

The following protocols implement both authentication and rekeying functionality using the DHCP wireless rekey option.

Public Key Version — Let *C* denote the STA and *S* the DHCP server. Suppose the STA and DHCP server have public key PK_c and PK_s, respectively. The corresponding certificates are denoted by $cert_C$ and $cert_S$, respectively. For simplicity,

we assume the same public key pair is used for both confidentiality and digital signature. If different keys are desired, two pairs of public keys can be given to each STA. The following describes the protocol when an STA joins the network.

$C \Rightarrow S(DHCPDISCOVER)$

The STA sends a DHCPDISCOVER message with the wireless rekey option set. The STA transmits the message encrypted with the link layer key A. The AP can listen on both A and K, and hence forwards the request to the DHCP server on the wired LAN.

$S \Rightarrow C(DHCPOFFER)$: $nonce_S$

The DHCP server sends a DHCPOFFER message, which includes a nonce $nonce_S$ chosen by the server.

$C \Rightarrow S(DHCPREQUEST)$: $nonce_S$, $nonce_C$, $cert_C$, sig_C

Then the STA transmits a DHCPREQUEST message, which includes the server nonce, $nonce_S$, a nonce, $nonce_C$, chosen by the STA, and authentication information. The authentication information includes the X.509 certificate $cert_C$ of the STA's public key and the STA's signature sig_C on the message.

$S \Rightarrow C(DHCPACK)$: $PK_C\{K\}$,$PK_C\{K_n\}$, $nonce_C$, $nonce_{Sf}$, $cert_S$, sig_S

The DHCP server sends back a DHCPACK message which includes the current WEP key K and the next WEP key K_n encrypted with the STA's public key PK_C, the STA nonce $nonce_C$, a new server nonce $nonce_{Sf}$ for future use (when the STA renews its WEP key), and authentication information. The authentication information includes the X.509 certificate $cert_S$ of the DHCP server and the server's signature sig_S on the message.

The following protocol describes WEP key renewal for a connected STA.

$C \Rightarrow S(DHCPREQUEST)$; $nonce_S$, $nonce_C$, $cert_C$, sig_C

This message is the same as the DHCPREQUEST message sent in the above protocol, except that here the server nonce, $nonce_S$, should be the future server nonce $nonce_{Sf}$ obtained during the previous joining or renewing message exchange with the DHCP server.

$S \Rightarrow C(DHCPACK)$: $PK_C\{K_n\}$, $nonce_C$, $nonce_{Sf}$, $cert_S$, sig_S

This message is the same as the DHCPACK message sent in the above protocol, except that here the DHCP server only needs to send the next WEP key K_n encrypted with the STA's public key PK_c.

We note that the nonces used in the above protocols could very well be taken from the 64-bit replay detection field as defined by the DHCP authentication option if the option is implemented. In such a case, we do not have to store those nonce values. The specific details of the DHCP authentication option are described in [7].

Shared Key Version — Suppose the DHCP server has a master key K_m. To minimize the keys stored at the DHCP server, the shared key K_c between an STA with identification $client_{id}$ and the DHCP server is generated using a secure hash function *HMAC*.

$K_C = HMAC(K_m|client_{id}|K_m)$

The shared STA keys are distributed to STAs in an out-of-band manner.

The following describes the protocol when an STA joins the network.

$C \Rightarrow S(DHCPDISCOVER)$

The DHCPDISCOVER message stays the same as the public key version.

$S \Rightarrow C(DHCPOFFER)$: $nonce_S$

The DHCPOFFER message also stays the same. A server nonce, $nonce_S$, is sent by the server.

$C \Rightarrow S(DHCPREQUEST)$: $nonce_S$, $nonce_C$, $client_{id}$, MAC_C

The DHCPREQUEST message stays the same, except that the authentication information now consists of the STA's identifier $client_{id}$ and message authentication code MAC_c generated by the STA using the shared key K_c.

$S \Rightarrow C(DHCPACK)$: K_C, K_C, $\{K_n\}$, $nonce_C$, $nonce_{Sf}$, MAC_S

The DHCPACK message stays the same, except that:

- The WEP keys are now encrypted using shared key K_c rather than the STA's public key PK_c
- The authentication information now consists of message authentication code MACs generated by the DHCP server using the shared key K_c.

The following protocol describes WEP key renewal for a connected STA.

$C \Rightarrow S(DHCPREQUEST)$: $nonce_S$, $nonce_C$, $client_{id}$, MAC_C

The DHCPREQUEST message stays the same as the public key version, except that now the authentication information consists of the STA's identifier $client_{id}$ and message authentication code MAC_c generated by the STA using the shared key K_c. Note that, just like in the public key version of the protocol, the server nonce $nonce_S$ used in this message should be the future server nonce, $nonce_{Sf}$, obtained during the previous joining or renewing message exchange with the DHCP server.

$S \Rightarrow C(DHCPACK)$: $K_C\{K_n\}$, $nonce_C$, $nonce_{Sf}$, MAC_S

DHCPACK message stays the same, except that:

- The next WEP key K_n is now encrypted using shared key K_c rather than the STA's public key PK_c
- The authentication information now consists of message authentication code MACs generated by the DHCP server using shared key K_c.

Using a Session Key — One variant to the above protocols is to derive a session key for encrypting the WEP keys, rather than using the shared secret key K_c or the STA's public key to encrypt

To be able to use authentication functionality in our protocol, we propose a public key-based and a shared key-based authentication system, which requires minimal state maintenance at the DHCP server. The public key mechanism also helps us solve the scalability problem.

Time (ms)	Mean	Min	10% tile	Median	90% tile	Max	Std dev
Reauth of client	156	117	126	126	133	1016	119
Processing client request	28	20	26	28	29	37	2

Table 1. *The time required to obtain a new WEP key.*

the WEP keys directly. When an STA joins the network and authenticates itself using the above two join protocols, a session key K_s will be returned by the DHCP server in place of the WEP key K, together with WEP keys K and K_n encrypted using K_s.

A session key is valid for a session that, depending on key size, could be much longer than the lifetime of a WEP key. When an STA sends a DHCPREQUEST message to renew its WEP key within the same session, the DHCP server will simply send back the next WEP key encrypted with the current session key K_s. There is no authentication involved within the same session. Beyond the current session, the STA has to use the join protocol to reauthenticate itself and get a new session key. In this way, we separate authentication and session management (using different keys). It is also much cheaper for the public key version. The drawback is that now the DHCP server has to store a session key K for each STA.

A Simple Improvement — A simple improvement over the current WEP implementations would be to have a setup where:

- The DHCP server gives all valid STAs the same master secret S_m.
- The DHCP server just includes a nonce N with each lease that allows the derivation of the next WEP key $K_n = \text{hash}(S_m, N)\}$.

The above protocol uses *implicit* authentication since any valid STA would have S_m. It does, of course, suffer from all of the manual key management problems as far as S_m goes, but it is a simple protocol that requires no per STA state to be stored at the DHCP server.

Implementation Details

We now provide the details of the implementation of one of the protocols described above (the public key version).

Driver and Wireless Extension Details

We implemented a prototype on Linux using Cryptlib as our security toolkit. We used the GPL driver for the WaveLAN IEEE/ORiNOCO (maintained by Andreas Neuhaus), which was included in the pcmcia-cs-3.1.15 package for Linux. The driver fully supports the wireless extensions v9 (note that the current driver is v1.0.6, included in pcmcia-cs-3.1.24). Wireless extension support is required to change keys on the fly.

Linux wireless tools v20 provides tools, such as iwconfig and iwspy, which can configure WEP keys on the fly if the driver fully supports wireless extension. We extracted the code in iwconfig as a C function call to change the WEP key.

Client and Server Daemons

We instrumented the DHCP client and server code (v 3.0rc1pl1 from Internet Software Consortium, ISC) and changed the code to include the facility for large options. We also added an option for wireless rekeying.

Dealing with Large Options

DHCP options cannot be more than 256 bytes. Since we implemented our own authentication mechanism within the wireless rekey option, we had to deal with the large option size, which exceeds 256 bytes. In such cases, the large option was split into multiple buffers, which are logically grouped into an aggregate buffer. Large DHCP options were stored in the DHCP packet in three separate portions of the packet. These are the optional parameters field, the sname field, and the file field.

Performance Analysis

A server machine, which acted as an AP, ran our firewall and the DHCP server daemon. Another machine, which acted as a client, ran the DHCP client daemon. Both machines ran on RedHat Linux 6.2 with a Pentium III 933 MHz processor and 128 Mbytes RAM. The cards were running in the ad hoc demo mode when collecting performance data. The reason for using ad hoc mode was that it took quite some time to installing the key on the wireless card (on the AP) if the AP and the DHCP server were not on the same machine. This is because most vendors use SNMP messages for re-keying, which can take between 5 and 20 s at the worst.[3]

We implemented the public key version protocol discussed earlier, and we tested the effects of key updates upon active connections.

This was of utmost importance, because if a key update disconnected a connection, it would have been of little use. We tested our key update protocol with many types of connection-oriented protocols, such as FTP, telnet, and Netperf, to analyze the performance of the protocol. None of the active connections was adversely affected by the concurrent key update process.

We also did various types of transfers of files ranging from 200 bytes to 10 Mbytes. This range covered a number of key updates while the file was being transferred. Even when key updates were happening rapidly (once every 10 s or so) none of the connections broke.

The amount of time it took for a disconnected user to get authenticated was measured. This reflected the time needed for an STA to get the WEP key. The amount of time it took for the server to process an STA's key update request was also measured. This determined the maximum number of key renewals per second.

Table 1 shows the above two measures. These values were taken from 200 key updates. From Table 1, we see that a DHCP server can process more than 35 key update requests/s. Moreover, it only takes approximately 0.1 s for the client to reauthenticate itself and join the network.

[3] *Now it is possible to build APs running Linux, which enables us to run the DHCP server and the AP on the same machine. Even if the DHCP server and the AP run on different machines, the extra overhead for installing the key is just the network delay on the wired network, which we believe will not affect our performance numbers drastically.*

CONCLUSIONS

The combination of our results with those of Walker and Borisov *et al*. demonstrates serious flaws in all of the security mechanisms used by the vast majority of access points supporting the IEEE 802.11 wireless standard. The end result is that most of the deployed 802.11 wireless networks are at risk of compromise. An interim short-term mitigation (not a complete solution) is a robust key management system for WEP, a higher-layer authentication system, and the use of a higher-layer transport mechanism (e.g., IPSec). The combination of these mechanisms provides a robust interim solution until hardware supporting the new standards is deployed.

ACKNOWLEDGMENTS

The authors would like to thank Jesse Walker of Intel Corporation, Mark Seiden of Securify, and Angelos Keromytis of Columbia University for providing valuable comments on a draft of this work.

REFERENCES

[1] LAN MAN Standards of IEEE Comp. Soc., "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification," 1999.
[2] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11." *7th Annual Int'l. Conf. Mobile Comp. and Net.*, Rome, Italy, 2001.
[3] J. Walker, "Unsafe at Any Key Size: An Analysis of the WEP Encapsulation," IEEE 802.11 Task Group E, 2000.
[4] S. Fluhrer, A. Shamir, and I. Mantin, "Weaknesses in the Key Scheduling Algorithm of RC4," *Sel. Areas of Cryptography*, Toronto, Canada, 2001.
[5] W. A. Arbaugh, "An Inductive Chosen Plaintext Attack Against WEP and WEP2. 2001," IEEE 802.11 Working Group, Task Group I (Security), 2002.
[7] R. Droms, and W. Arbaugh, Authentication for DHCP Messages, 2001, Internet Engineering Task Force (IETF).

ADDITIONAL READING

[1] N. Shankar, W. Arbaugh, and K. Zhang, "A Transparent Key Management Scheme for Wireless LANs Using DHCP," HP Laboratories, Palo Alto, CA, 2001.

BIOGRAPHY

WILLIAM ARBAUGH (waa@cs.umd.edu) joined the Computer Science Department at Maryland After spending 16 years with the U.S. Department of Defense, first as a commissioned officer in the Army and then as a civilian. During those years, he served in several leadership positions in diverse areas ranging from tactical communications to advanced research in information security and networking. In his last position, he served as a senior technical advisor in an office of several hundred computer scientists, engineers, and mathematicians conducting advanced networking research and engineering. He received a B.S. from the United States Military Academy at West Point, an M.S. in computer science from Columbia University, New York City, and a Ph.D. in computer science from the University of Pennsylvania in Philadelphia. His research interests include information systems security and privacy with a focus on wireless and embedded systems and configuration management. He also currently serves on the editorial board of *IEEE Computer*, where he edits a bimonthly column on information security.

NARENDAR SHANKAR (narendar@cs.umd.edu) is a graduate student in the computer science department at the University of Maryland, College Park. His research interests are designing security mechanisms for wireless and ubiquitous computing. He is a recipient of a graduate fellowship in the computer science department.

YUNG-CHUN JUSTIN WAN (ycwan@cs.umd.edu) is a Ph.D. student in the Computer Science Department at the University of Maryland, College Park. He received his M.Sc. in computer science from the same school in December 2001. His current research interests are in applying algorithms to computer networks and approximation algorithms.

KAN ZHANG (kzhang@hpl.hp.com) is a researcher at Hewlett-Packard Labs, Palo Alto, California. His research is focused on security and privacy issues in nomadic computing. He holds a PhD in computer science from Cambridge University.

An interim short-term mitigation (not a complete solution) is a robust key management system for WEP, a higher layer authentication system, and the use of a higher layer transport mechanism, e.g., IPSec.