A Transparent Key Management Scheme for Wireless LANs Using DHCP

Narendar Shankar[1], William A. Arbaugh[1],Kan Zhang,
Mobile Systems and Services Laboratory
HP Laboratories Palo Alto
HPL-2001-227
September 12th , 2001*

Wireless communication continues to grow in importance as does the need for secure communications. The well known IEEE 802.11b standard leaves the implementation of a key management system as an open problem for vendors, and recent attacks have shown the need for a robust key management system. What is required is a system which is scalable, deployable and most importantly a system with minimum manual intervention as far as key management is concerned. It should handle periodic WEP key updating and the problem of users rejoining the network with transparency even though the key period may have changed. In this paper, we present a solution for transparent key management within a wireless LAN environment.

* Internal Accession Date Only Approved for External Publication
[1] Department of Computer Science, University of Maryland, College Park, MD, 20742

A Transparent Key Management Scheme for Wireless LANs Using DHCP

Narendar Shankar
Department of Computer Science
University of Maryland
College Park, Maryland 20742 *

William A. Arbaugh
Department of Computer Science
University of Maryland
College Park, Maryland 20742†

Kan Zhang
Hewlett-Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304, USA

August 29, 2001

Abstract

Wireless communication continues to grow in importance as does the need for secure communications. The well known IEEE 802.11b standard leaves the implementation of a key management system as an open problem for vendors, and recent attacks have shown the need for a robust key management system. What is required is a system which is scalable, deployable and most importantly a system with minimum manual intervention as far as key management is concerned. It should handle periodic WEP key updating and the problem of users rejoining the network with transparency even though the key period may have changed. In this paper, we present a solution for transparent key management within a wireless LAN environment.

1 Introduction

Organizations are rapidly deploying wireless infrastructures based on the IEEE 802.11b standard [1]. Unfortunately, this standard provides only limited and weak support for confidentiality through the Wired Equivalent Privacy (WEP) protocol, and the access control and authentication mechanisms are even weaker. Compounding these problems, the standards committee for 802.11b left many of the difficult security issues such as key management, and a robust authentication mechanism as open problems. Currently, the only widely used key management solution is to *manually* distribute WEP keys to each user. Optionally, the keys could be electronically distributed, e.g. via secure electronic mail, but this distribution method also requires each user to *manually* change the key entry. History has shown that manual key management systems are fraught with failures due to the human aspect of the system [2]. As a result, many of the organizations deploying wireless networks use either a permanent fixed key or no encryption at all. 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 security problem.

Furthermore, recent passive cryptanalytic attacks permit the recovery of the WEP session key after the collection of sufficient cipher text encrypted with the same session key [3]. Changing the session key before an adversary can collect sufficient traffic prevents the complete recovery of the session key.

This paper presents a transparent key management solution that provides user authentication and periodic WEP key updating for systems based on the IEEE 802.11b standard. Proper wireless key management mitigates many of the existing wireless security problems. Our approach is to make the key management of the system as transparent to the users and managers as possible. Our solution enables a disconnected user to authenticate to a wireless network and obtain the current WEP keys. A disconnected user is one who leaves the network temporarily and returns

*Portions of this work performed while on a summer internship at Hewlett-Packard research labs.

†This work was sponsored in part by an IBM Faculty Partnership Award.

after the current WEP key has changed. We have chosen to use DHCP [4] as a transport mechanism for authentication and key updates using a phased approach to avoid overloading the server. The use of DHCP requires minimal changes to both the wireless stations and the wireless infrastructure.

In the next section, we present a brief overview of the defined 802.11b security mechanisms. We follow this with a review of the related work. Next, we describe our key management solution and provide performance information for the initial prototype. Finally, we present some conclusions.

2 802.11b Security Overview

The IEEE 802.11b standard provides several mechanisms intended to provide a secure operating environment. In this section, we give a brief overview of these mechanisms as well as one vendor proprietary method. Unfortunately, few of the mechanisms provide any semblance of security.

2.1 Wired Equivalent Privacy

Wired Equivalent Privacy (WEP) is a standard specified by the IEEE 802.11b standard for ensuring confidentiality of link layer data. WEP uses a shared link layer key (called WEP key) for all nodes that wish to communicate with each other. Link layer traffic is encrypted and decrypted with this key. A window of four WEP keys can be used as a back up when keys become invalid or unusable. The details of WEP protocol are beyond the scope of this paper. But in brief, WEP uses a 24 bit Initialization Vector (IV) together with either a 40 or 104 bit WEP key. Recent work by Borisov, Goldberg, and Wagner [5] demonstrates that WEP provides only limited confidentiality, because the WEP key is not changed (IV is changed on a per-packet basis). More recent work by Fluhrer, Mantin, and Shamir [3] demonstrates that WEP keys can be recovered within a short period of time. A transparent key management scheme that updates the WEP key frequently can prevent these attacks.

2.2 Authentication and access control problems

Work by Arbaugh, Shankar and Wan [6] describes the current authentication mechanisms used in 802.11b based wireless networks and demonstrates that most of the mechanisms in use are flawed. This raises the issue of using robust authentication mechanisms currently only available in higher layer authentication mechanisms. Our solution takes this approach.

2.3 Key management

Key management is a misnomer with respect to 802.11b as it is left as an exercise for the vendors. The standard does, however, provide for two methods of using WEP keys. The first is a *shared key* method, which provides a window of four WEP keys. A wireless station (STA) or access point (AP) can decrypt packets enciphered with any one of the four WEP keys. Transmission, however, is limited to a single WEP key – 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 802.11b specification. The maximum size is likely to be chipset dependent. The use of a separate key for each user will mitigate some of the attacks found by Borisov et al., but providing a reasonable key renewal period remains a problem as the keys can only be changed manually.

3 Related Work

Wirleless security is a growing concern among the industry and research communities. In this section, we describe several related efforts.

3.1 Key Management

The number of keys that can be stored at the link layer is just four (for the shared key model). Hence key management schemes for secure group communication (e.g., secure multicast), like key graphs [7], cannot be used directly at the link layer. Although they can be used as part of a higher layer solution, they require too much state at the server, which makes them practically infeasible. Besides, the assumptions of a wireless LAN are different from those of secure group communication. Joining and leaving a wireless LAN is not the same as joining or leaving such a group. For instance, when a user shuts down their laptop, it cannot be equivalent to leaving the network, and it does not necessitate a change of the group as such.

Several vendors have developed proprietary solutions for establishing unique session keys. The advantage of this solution is that WEP keys are renewed on a per user, per session basis. A session begins when the user joins the network and ends when the user leaves the network. However, the key remains

static through out the entire session. Even if a user is online for weeks, the link-layer key does not change. This might be a very common situation in a wireless LAN backbone. In other words, there is no *timed key management protocol* which takes into account key periods. Additionally, it does not provide a proper means for authenticating a disconnected user.

3.2 Authentication

LEAP is a lightweight implementation of the Extensible Authentication Protocol(EAP) [8] from Cisco Systems. It solves the problems associated with authentication and also provides for a mechanism for generating a per user per session key. However, like other solutions, LEAP re-keys (or a key update/renewal) only at session initialization. For users who have long sessions, a key update does not take place at a suitable key period.

3.3 802.1x based 802.11

802.1x based 802.11 solves a large set of the problems which exist in current wireless networks, but a problem with this approach is that it involves infrastructural changes which may involve the changes of AP and STA hardware. The solution described later in the paper, on the other hand, involves minimal changes to the existing infrastructure and does *not* require any hardware changes. Furthermore, 802.1x is a recent standard that is not currently supported in any current operating system.

4 Design Issues

We now present several design issues involved in our solution.

Figure 1 shows our set-up of a wireless LAN consisting of STAs which communicate with the wired portion of the network using an access point. The access point acts as a link layer bridge forwarding data between the wired and wireless sides of the network. We also have a firewall and a DHCP server behind these access points, and a LDAP based certificate revocation list which is used for checking validity of public-key certificates.

4.1 Design Goals

We have several design goals:

1. Design of a key management system, which inter-operates with all the implementations of the IEEE 802.11b standard. More importantly, the system must be integrated into existing wireless architectures with minimal overhead.

2. Solve the disconnected user problem.

   In the shared key model in WEP, all STAs communicate with the AP using the same WEP key. A disconnected user cannot authenticate to the network, because he does not have the current WEP key. Some solutions advocate the use of an open mode (unencrypted mode) for authentication. While such a mode simplifies the overall architecture, it also allows an adversary easy access to the local network.

3. Design of a system that performs dynamic and transparent key updates. A key update during the current session should not disrupt communication in that session.

4. The key management system must support key periods. In other words, re-keys (key updates) must be done in a periodic fashion.

5. Perform strong authentication before issuing a STA the current WEP key.

4.2 Argument for a higher layer key management system

In this paper when we talk about a higher layer key management system, we mean performing WEP key management above the link layer. And in our case, we use an application layer solution, e.g. DHCP.

The reasons for this are the following:

1. The recently discovered flaw in the key scheduling algorithm of RC4 by Fluhrer et al. [3] permits an adversary to recover the WEP key after the collection of a number of encrypted packets. The number of packets to successfully exploit this weakness varies, but the lower bound is estimated at several hundred thousand packets. This attack can be prevented through the use of a short key period designed to prevent the attacker from collecting sufficient packets to fully recover the secret key. Hence, we need a mechanism of changing link layer keys frequently. Currently, 802.11b does not have link layer key management frames and therefore, we need a higher layer key management solution.

2. Most of the other interim solutions, such as [9], tie authentication and key management with a client entering or leaving the network. Authentication takes place whenever the client enters the network. After the client is authenticated,

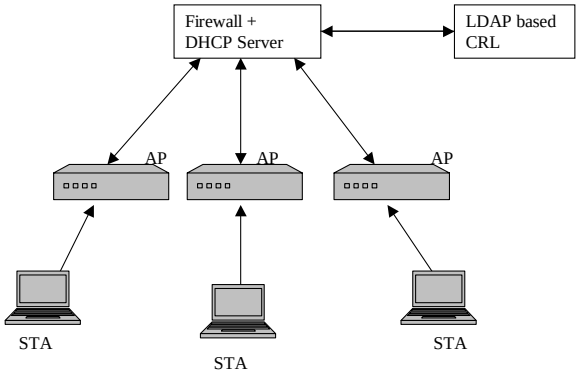


Figure 1. Our wireless network

a session key is also generated. The session key remains valid until the client leaves the network. This creates a *dangerous coupling between session length and key management*, because there can be very long sessions (which are not uncommon, because most lap-tops come with many power conserving modes). Moreover, in the future most desktops and a large number of devices will use wireless cards to minimize wiring and infrastructure costs. Thus, a key management scheme must support key periods, i.e., WEP keys should be updated at fixed time intervals and not just at the time when the client enters the network. Moreover, it must be possible to tune the length of the key period based on the number of nodes in the network and also the level of security desired, i.e., flexible security parameterization is required. Currently, only a higher layer key management solution can solve this problem and enable flexible security parameterization.

3. A higher layer solution can solve the problem of a disconnected user, and help him authenticate to the network without using an un-encrypted mode for authentication.

4. A higher layer authentication mechanism can provide better authentication. As shown in the work by Arbaugh et al. [6], current link layer authentication mechanisms are not sufficient.

4.3 Use of DHCP

DHCP provides an excellent transport mechanism for wireless key management. The reasons for using DHCP are the following.

1. DHCP has been widely deployed by many networks for allocating IP addresses.

2. When the client joins the network it obtains an IP address and with the use of DHCP options [10], it also obtains the current WEP key.

3. DHCP authentication mechanisms have already been specified [11]. However there is no version of DHCP which has implemented the authentication option yet. Since a DHCP server needs to authenticate its clients before giving out WEP keys, we implement some authentication protocols with our new DHCP wireless re-key option that is used for WEP key updating. Details are given in Section 6.

4. DHCP leases [4] provide an excellent means for timed key management. This is discussed in detail in Section 5.

5 A General Framework

In this section, we give a high-level view of the message exchanges used in our key management scheme and discuss the ideas behind our protocol design.

5.1 Definitions

This paper uses the following notations.

1. *Current WEP key K*

   This is the default link layer key that is being used for encrypting link layer communication. We refer to this key as the current WEP key.

2. *Next WEP key Kn*

   Our key management system keeps changing the current WEP key. This next WEP key Kn is the WEP key to be used for the next key period. In other words it is the key to be used by the network after K.

3. *Long-lived WEP key A*

   This is another link layer key with a relatively *long life*, which is used by an STA for communicating to the AP when the current WEP key K is not available.

5.2 The high-level protocol

The ideal way to mitigate a majority of the current WEP flaws [5] is to keep changing the current WEP key K. At the same time, valid STAs of the network, who leave the network, must be able to obtain future WEP keys when they rejoin the network.

Before we present our solution, we discuss the concept of a hierarchy of keys based on key lifetimes. Assume a set of keys S is used over a period of time in a system. A subset of these keys *S1* tend to be used more frequently than the rest. This set of keys forms the lowest level of the hierarchy-*L1*. Another subset of keys *S2*(disjoint from *S1*) forms the next level *L2* in the hierarchy because the keys in this level are not used as frequently as the keys in the level *L1*. The top of the hierarchy consists of the set *Sn* at level *Ln* which consists of keys which are used the least in the system. This gives us a hierarchical organization of keys based on the frequency of use of keys, which in turn determines the lifetime of the keys.

This concept of hierarchical organization is used to design the *two door entry* mechanism used in our solution. In this mechanism, a disconnected STA that doesn't have key K can communicate and authenticate to the DHCP server using key A. The frequency of use of A is small when compared to K and hence A is considered to be a key with a longer lifetime. In other words, based on the hierarchical organization, A is at a higher level in the hierarchy and K is at a lower level.

Both the AP and the 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 which are connected to the network have K and A. The disconnected STAs that do not have K (they had left the network and are now rejoining) can authenticate themselves using A. After the STAs have authenticated themselves they obtain the current WEP key K and the next WEP key Kn.

Apart from joining the network, regular WEP key updating takes place for all STAs currently connected to the network. We use DHCP as a transport mechanism for getting the next WEP key Kn.

The basic idea is as follows:

1. When an STA joins/rejoins the network, it is assigned an IP address and is given the current WEP key K and the next WEP key Kn. (The reasons for sending both K and Kn are explained in section 5.4.) The IP address is leased for a particular time period. This time period is set by the organizational policy.

2. All the STAs currently connected to the network renew their IP addresses depending on the lease time and in the process also get the next WEP key Kn.

The high-level protocol is shown in Figure 2. The details are explained in the following two sections.

5.2.1 Joining the network

The AP listens on both key A and K and transmits in key K. An STA that is not a part of the network cannot authenticate itself because it does not even have key A and hence will not be able to communicate with the network. An STA that is a part of the network (and is disconnected) has the long-lived WEP key A. At this point, we assume the availability of the DHCP authentication option DHCP [11]. (Because none of the available versions of DHCP had implemented the authentication option, we implemented our own authentication mechanism. This is discussed later in the paper). We define a new DHCP option [10] called *wireless re-key option*, which is used as a transport for wireless re-key. The format of the wireless re-key option is given in Figure 3. The DHCP wireless re-key option should be used only when the DHCP authentication option is set.

As shown in Figure 2:

1. The STA sends a DHCPDISCOVER message with the wireless re-key option set and the DHCP authentication option set. The STA transmits the message encrypted with the 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.

2. The DHCP server sends a DHCPOFFER message including the authentication information in accordance with the DHCP authentication protocol [11]. We note that the AP's transmission key must be changed to A before transmission and back to K afterwards.

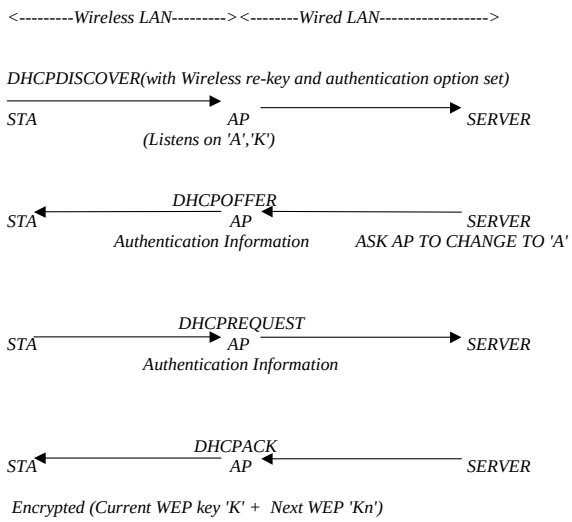


Figure 2a. Protocol for joining the network

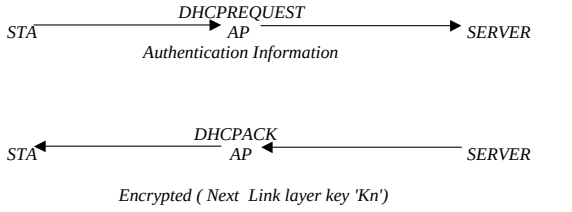


Figure 2b. Protocol for WEP key renewing

3. Then the STA transmits a DHCPREQUEST message, which includes the authentication information. The authentication information is again in accordance with the DHCP authentication protocol.

4. 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 [11]. Also the DHCP server sends the next WEP key Kn encrypted. The reasons for sending both K and Kn are explained in section 5.4. The problem of encrypting the WEP keys with the shared key as defined by [11] 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.

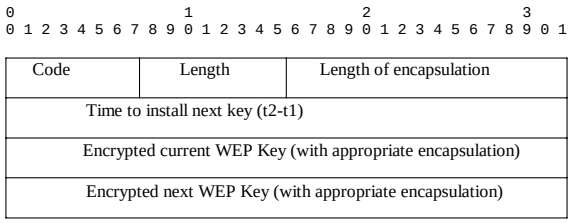
5.2.2 WEP key renewing

When secure communication is going on, there arises a need for renewing the WEP keys when their lifetime is over. Our protocol does an automatic key update without any manual intervention and we believe that no other proposed solution does this efficiently and in a scalable manner. This is efficiently implemented using the 'DHCP-leases'. The IP address for an STA is valid only till the lease is valid. Once the lease expires the STA has to renew its IP address. When the STA sends a DHCPREQUEST with the wireless re-key option set, the new WEP key is also returned when the IP address is renewed. In other words we now tie 'key-periods' with 'dhcp-lease-periods'. This solves the problem for long sessions because a lease can expire in between a session and all the STA does is to renew its IP address and in the process renew the WEP key. Tying key-periods to the leases removes the need for a new client and server software which most other solutions propose.

As shown in Figure 2b, during a DHCP lease renew, only the DHCPREQUEST and DHCPACK messages are exchanged. This phase is very similar to the joining phase, but for the fact that the STA can transmit in K, and therefore, can simply request for Kn via DHCPREQUEST message. Again here the new WEP key Kn must be sent encrypted.

5.3 Scalability issues

The above solution can be made scalable in the following sense.



- Code field is TBD
- Length field specifies the length of the option
- The DHCP authentication option must be set if the wireless re-key option is set.
- Length of encapsulation field must be set to the size of the encapsulation method used to encapsulate the encrypted current wireless key (explained next).
- Encrypted current and next key may be of variable length but must be encapsulated with the PKCS#7 or CMS, which gives enough information about the algorithm and the encryption parameters
- Time to install next key is the time in seconds between acquiring the next WEP Key and the time when it is to be installed on the card. The length must be set to 32 bits.

Figure 3. DHCP wireless re-key option

1. An STA is able to connect to multiple networks and authenticate itself.

2. An STA is able to get a shared encrytion key (for encrypting the WEP keys) with the DHCP servers on those networks.

The solution to the first problem is to pre-arrange a set of A keys on an STA, which allows the STA to connect to different networks. Once an A key that is appropriate for a particular network is installed on an STA, it can authenticate itself with the DHCP server on the wired part of the network. We note that the pre-arrangement of A keys on STAs does not necessarily compromise the security of the respective wireless networks, since an STA still has to authenticate itself before it can get the current WEP key K. Key A only allows the STA to have link layer communication so that it can perform the higher layer authentication.

The solution to the second problem revolves around the fact that the WEP key K sent in the DHCPACK message must be encrypted. The WEP key has to be encrypted with a shared secret between the STA and the DHCP server. It might be possible to have this shared secret pre-arranged, but a better alternative would be to use a public key based authentication scheme which generates this shared secret. Details are given in Section 6.

5.4 Delayed key installation

A simple key management protocol has two problems:

1. If all STAs renew their WEP keys at the same time then the server becomes overloaded.
2. If all the STAs renew their keys at different times, then inconsistency problems are introduced and the server might have to listen on multiple keys for prolonged periods.

We introduce a new idea of timed key management. All STAs contact the DHCP server at different times to get the next WEP key. When an STA initially joins the network, it needs both the current WEP key and the next WEP key. The reason for doing so is that during every key renewal the STA gets the next WEP key, but initially the STA does not even have the current WEP key. Hence it has to be given both keys. An example is given below.

Suppose an STA join the network at time t1, the current WEP key at time t1 is K and the DHCP lease period is T seconds. The STA is supposed to contact the server again at time t1+T (i.e t1+ time of lease). Since the STA joins in the middle of a key updating period, at some time t2 (t1 $\leq$ t2 $\leq$ t1+T), a new WEP key Kn will be installed on all cards (t2 is called *key installation time* and t1+T is called *key renewal time*). This means that from t2 to t1+T this STA has to use Kn to communicate with the rest of the network.

Thus when the STA joins the network initially it must be given both the current WEP key K and the next WEP key Kn. Also the STA must be given the directive -'Install key K on the card immediately and use (install on card) key Kn after t2 - t1 seconds'.

In the key renewal stage, the STA already has the current key and it only needs to get the next WEP key Kn. Assume that the STA contacts the server at time t1+T(end of the lease), it obtains the key Kn to be used at time t2+T and the directive 'Install key Kn after t2+T-(t1+T) = t2-t1 seconds'

Thus the STA keeps contacting the server at times(renewal phase) t1 + T, t1 + 2T, t1 + 3T etc., but actual use or installation of the key takes place at times t2, t2 + T, t2 + 2T, t2 + 3T etc.

6 Implementable Protocols

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 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 the authentication and re-keying functionality using the DHCP wireless re-key option.

6.1 Public key version

Let C denote the STA and S the DHCP server. Suppose the STA and the 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 paragraph describes the protocol when an STA joins the network.

1. $C \rightarrow S$ *(DHCPDISCOVER)*

   The STA sends a DHCPDISCOVER message with the wireless re-key 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 the DHCP server on the wired LAN.

2. $S \rightarrow C$ *(DHCPOFFER): nonce_S*

   The DHCP server sends a DHCPOFFER message which includes a nonce $nonce\_S$ chosen by the server.

3. $C \rightarrow S$ *(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.

4. $S \rightarrow C$ *(DHCPACK): PK_c{K}, PK_c{Kn}, nonce_C, nonce_S_f, cert_S, sig_S*

   The DHCP server sends back a DHCPACK message which includes the current WEP key K and the next WEP key Kn encrypted with the STA's public key PK_c, the STA nonce $nonce\_C$, a new server nonce $nonce\_S\_f$ 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.

1. $C \rightarrow S$ *(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 $nonce\_S\_f$ obtained during the previous joining or renewing message exchange with the DHCP server.

2. $S \rightarrow C$ *(DHCPACK): PK_c{Kn}, nonce_C, nonce_S_f, 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 Kn 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 case, we don't have to store those nonce values. The specific details of the DHCP authentication option are described in [11].

6.2 Shared key version

Suppose the DHCP server has a master key *Km*. To minimise the keys stored at the DHCP server, the shared key *Kc* between an STA with identification *client_id* and the DHCP server is generated using a secure hash function *hash()* as follows [12].

Kc = hash(Km | client_id | Km)

The shared STA keys are distributed to STAs in an out of band manner. The following paragraph describes the protocol when an STA joins the network.

1. *C → S (DHCPDISCOVER)*

   DHCPDISCOVER message stays the same as the public key version.

2. *S → C (DHCPOFFER): nonce_S*

   DHCPOFFER message also stays the same. A server nonce *nonce_S* is sent by the server.

3. *C → S (DHCPREQUEST): nonce_S, nonce_C, client_id, MAC_c*

   DHCPREQUEST message stays the same, except that the authentication information now consists of STA's identifier *client_id* and message authentication code *MAC_c* generated by the STA using the shared key *Kc*.

4. *S → C (DHCPACK): Kc{K}, Kc{Kn}, nonce_C, nonce_S_f, MAC_s*

   DHCPACK message stays the same, except that a) the WEP keys are now encrypted using shared key *Kc* rather than the STA's public key *PK_c* and b) the authentication information now consists of message authentication code *MAC_s* generated by the DHCP server using the shared key *Kc*.

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

1. *C → S (DHCPREQUEST): nonce_S, nonce_C, client_id, MAC_c*

   DHCPREQUEST message stays the same as the public key version, except that now the authentication information consists of STA's identifier *client_id* and message authentication code *MAC_c* generated by the STA using the shared key *Kc*. 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_S_f* obtained during the previous joining or renewing message exchange with the DHCP server.

2. *S → C (DHCPACK): Kc{Kn}, nonce_C, nonce_S_f, MAC_s*

   DHCPACK message stays the same, except that a) the next WEP key *Kn* is now encrypted using shared key *Kc* rather than the STA's public key *PK_c* and b) the authentication information now consists of message authentication code *MAC_s* generated by the DHCP server using the shared key *Kc*.

6.3 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 *Kc* or the STA's public key to encrypt the WEP keys directly. When an STA joins the network and authenticates itself using the above two join protocols, a session key *Ks* will be returned by the DHCP server in place of the WEP key *K*, together with WEP keys *K* and *Kn* encrypted using *Ks*.

A session key is valid for a session which, depending on the 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 *Ks*. There is no authentication involved within the same session. Beyond the current session, the STA has to use the join protocol to re-authenticate itself and get a new session key. In this way, we seperate 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 *Ks* for each STA.

6.4 A simple improvement

A simple improvement over the current WEP implementations would be to have a set-up where

a) The DHCP server gives all valid STAs the same master secret *S_m*, and

b) The DHCP server just includes a nonce *N* with each lease that allows the derivation of the next WEP key *Kn = 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.

7 Implementation Details

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

7.1 Driver and wireless extension details

We have implemented a prototype on Linux using cryptlib [13] as our security toolkit. We are using the GPL driver for the WaveLAN IEEE/ORiNOCO [14] (maintained by Andreas Neuhaus), which is included in the pcmcia-cs-3.1.15 package for Linux. The driver fully supports the wireless extensions v9 (NOTE: 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.

Lucent provides its own Linux driver for the card. However, Lucent provides only binaries for driver's core and thus is not entirely open source. Although Lucent's driver is more stable and of higher performance, we decided to use the GPL driver. Lucent's driver does not fully support wireless extension. We cannot change the WEP key without reloading the driver. Linux wireless tools v20 provides tools, such as *iwconfig* and *iwspy* [15], which can configure WEP keys on the fly, if the driver fully supports wireless extension. We extracted the code in *iwconfig* so that our client daemon and server daemon can call a C function to change the WEP key.

7.2 Client and server daemons

We have instrumented the DHCP client and server code (Version: 3.0rc1pl1 from ISC (Internet Software Consortium)) [16] and changed the code to include the facility for large options. We have also added an option for wireless re-keying.

7.3 Dealing with large options

DHCP options cannot be more than 256 bytes. Since we implemented our own authentication mechanism within the wireless re-key option, we have to deal with large option size which may exceed 256 bytes. In such cases, the large option must be split into multiple buffers which are logically grouped into an aggregate buffer. Large DHCP options can be 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, as described in [10].

8 Performance Analysis

A server machine, which acts as an access point, runs our firewall and the DHCP server daemon. Another machine, which acts as a client, runs the DHCP client daemon. Both machines run on RedHat Linux 6.2 with Pentium III 733 MHz processor and 128Mbyte RAM. At this moment, the cards are running "Ad-Hoc Demo Mode" when collecting performance data. The reason for using ad-hoc mode is that it takes quite some time for installing the key if the AP and the server are not on the same machine. This is because most vendors use SNMP messages for re-keying which can take between 5 seconds and 20 seconds at the worst. Running our key management protocol on top of such delays really makes no sense. We plan to implement a wBox which is an access point running an operating system. Once that is done tools like iwconfig can be used to change the access point's keys on the fly.

We implemented the public key version protocol discussed in section 6.1. We tested whether connections got affected because of the key updates. This is of utmost importance, because if a key update is going to disconnect a connection, then it is of little use. We tested our key update protocol with many types of connection oriented protocols, like FTP, telnet and Netperf to analyze the performance of the protocol. Not even in a single instance did any connection break because of the key updates.

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

The amount of time it takes for a disconnected user to get authenticated is measured. It reflects the time needed for an STA to get the WEP key. The amount of time it takes for the server to process an STA's key update request is also measured. It determines the maximum number of key renewals per second.

Table 1 shows the above two measures. These values are taken from 200 key updates. From Table 1, we see that a DHCP server can process more than 35 key update requests per second. Moreover, it only takes about 0.1 second for the client to re-authenticate itself and joins the network.

9 Conclusions

The rapid deployment of wireless networking based on the IEEE 802.11b standard has extended the reach of networking beyond areas that are easily wired.

Time (ms)	Mean	Min	10%-tile	Median	90%-tile	Max	SD
Re-authentication of client	156	117	126	126	133	1016	119
Processing client's request	28	20	26	28	29	37	2

Table 1: Time taken to obtain a new WEP key.

But, the multitude of security problems with 802.11b has also created an opportunity for malicious outsiders to exploit these same networks. In this paper, we have shown that a transparent key management system can implement a re-keying strategy that prevents the recovery of the current session key as well as preventing other known flaws with WEP. Furthermore, the addition of a strong authentication mechanism prevents unauthorized users from utilizing the network. Implementing both the key management system and the authentication mechanism was accomplished through simple modifications to DHCP– allowing easy deployment of these mechanisms into a corporate infrastructure.

References

[1] "LAN MAN Standards of the IEEE Computer Society. Wireless LAN medium access control (MAC) and physical layer(PHY) specification. IEEE Standard 802.11, 1997 Edition," 1997.

[2] R. Anderson, "Why cryptosystems fail," *Communications of the ACM, 37(11):32–40*, November 1994.

[3] S. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the Key Scheduling Algorithm for RC4." To Appear in Selected Areas in Cryptography(SAC).

[4] R. Droms, "Dynamic Host configuration Protocol," Tech. Rep. RFC2131, Internet Engineering Task Force (IETF), March 1997.

[5] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting Mobile Communications: The Insecurity of 802.11." `http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html`.

[6] W. Arbaugh, N. Shankar, and J. Wan, "Your 802.11 wireless network has no clothes." `http://www.cs.umd.edu/~waa/wireless.pdf`.

[7] C. K. Wong, M. G. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, Feb 2000.

[8] L. Blunk and J. Vollbrecht, "PPP Extensible Authentication Protocol (EAP)," Tech. Rep. RFC2284, Internet Engineering Task Force (IETF), March 1998.

[9] Cisco Systems, *Lightweight EAP*, November 2000.

[10] S. Alexander and R. Droms, "DHCP Options and BOOTP Vendor Extensions ," Tech. Rep. RFC2132, Internet Engineering Task Force (IETF), March 1997.

[11] R. Droms and W. Arbaugh, "Authentication for DHCP Messages ," Tech. Rep. RFC3118, Internet Engineering Task Force (IETF), June 2001.

[12] U. Blumenthal, N. C. Hien, and B. Wijnen, "Key Derivation for Network management Applications." IEEE Network magazine, May 2000.

[13] "Cryptlib-an encryption toolkit." Available at `http://www.cs.auckland.ac.nz/~pgut001/cryptlib/`.

[14] "Orinoco pc card (silver/gold) by lucent technologies." Available at `http://www.wavelan.com/`.

[15] J. Tourrilhes, "Wireless Tools, iwconfig, iwspy." `www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html`.

[16] ISC, *Implementation of DHCP-version 3.0rc1pl*, November 2000.