Secure Wireless Networks

Binoy George
Advanced Information Laboratory
Department of Computer Science
University of Cape Town
Rondebosch, 7701 RSA
bgeorge@cs.uct.ac.za

K.J. McGreggor
Advanced Information Laboratory
Department of Computer Science
University of Cape Town
Rondebosch, 7701 RSA
ken@cs.uct.ac.za

ABSTRACT
This paper provides a brief overview of wireless (WiFi) networks and some of the security measures in place today. We further seek to find a secure way to authenticate and enable secure communication between wireless (WiFi) clients and external networks.

Keywords
Wireless Communication, 802.11b protocol, WEP, WiFi Security

1. INTRODUCTION
Wireless Fidelity (WiFi) has taken the world by storm. WiFi provides a means of communicating using ether as the communication medium. This proposition of using such a medium is very alluring because it is both abundant and cheap. Setting up the network is equally easy and the main advantage of such a network is the mobility of the WiFi clients.

The WiFi standard is based on IEEE 802.11 family protocols, with the most common protocol used for WiFi being IEEE 802.11b protocol. WiFi operates on a 2.4GHz[1] digital signal, and currently the standard maximum speed of WiFi is 11Mbps. There are manufacturers offering equipment using a similar protocol but operating at higher speeds; this has not yet been made standard and is therefore proprietary to the manufacturer of the equipment.

2. COMMON WIRELESS SETUP
There are two main ways of setting up a WiFi network. One is called “infrastructure” and the other “ad-hoc”.

2.1 Ad-hoc Setup
In an “ad-hoc” network, the various WiFi clients (e.g. a laptop with a wireless ethernet card) communicate to each other directly, in a point-to-point fashion. The ad-hoc network does not require any additional hardware for the WiFi clients to communicate between themselves.

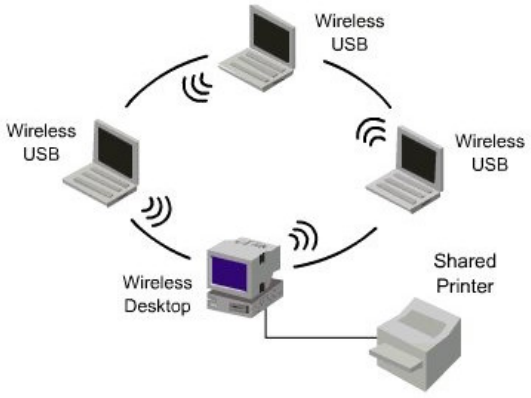

Figure 1: Wireless Ad-Hoc Network

2.2 Infrastructure Setup
From the names it is pretty obvious that a network setup as “infrastructure” would require some sort of base station (or access point) to relay information between the different WiFi clients. A WiFi client in range of the base station can communicate with all other WiFi clients.

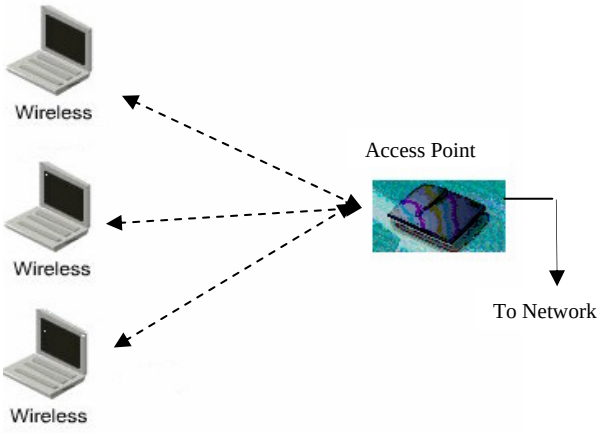

Figure 2: Wireless Infrastructure Network

We are mainly concerned with the infrastructure network setup, and more specifically accessing of external networks. This is the more popular kind of setup, and

[1] License free band width

3. STANDARD SECURITY FEATURES

There are two main security features that come standard with all WiFi networks.

3.1 WEP (Wired Equivalent Privacy)

The Wired Equivalent Privacy (WEP) algorithm is used to protect wireless communication from eavesdropping. A secondary function of WEP is to prevent unauthorized access to a wireless network; this function is not an explicit goal in the 802.11 standard, but it is frequently considered to be a feature of WEP.

WEP relies on a secret key that is shared between a wireless client and an access point. The secret key is used to encrypt packets before they are transmitted, and an integrity check is used to ensure that packets are not modified in transit. The standard does not discuss how the shared key is established. In practice, most installations use a single key that is shared between all mobile stations and access points. More sophisticated key management techniques can be used to help defend from the attacks we describe; however, no commercial system we are aware of provides mechanisms to support such techniques. [1]

3.1.1 Problems With WEP

WEP uses the RC4 encryption algorithm, which is known as a stream cipher. A stream cipher operates by expanding a short key into an infinite pseudo-random key stream. The sender XORs the key stream with the plaintext to produce ciphertext. The receiver has a copy of the same key, and uses it to generate identical key stream. XORing the key stream with the ciphertext yields the original plaintext.

This mode of operation makes stream ciphers vulnerable to several attacks. If an attacker flips a bit in the ciphertext, then upon decryption, the corresponding bit in the plaintext will be flipped. Also, if an eavesdropper intercepts two ciphertexts encrypted with the same key stream, it is possible to obtain the XOR of the two plaintexts. Knowledge of this XOR can enable statistical attacks to recover the plaintexts. The statistical attacks become increasingly practical as more ciphertexts that use the same key stream are known. Once one of the plaintexts becomes known, it is trivial to recover all of the others.

WEP has defenses against both of these attacks. To ensure that a packet has not been modified in transit, it uses an Integrity Check (IC) field in the packet. To avoid encrypting two ciphertexts with the same key stream, an Initialization Vector (IV) is used to augment the shared secret key and produce a different RC4 key for each packet. The IV is also included in the packet. However, both of these measures are implemented incorrectly, resulting in poor security.

The integrity check field is implemented as a CRC-32 checksum, which is part of the encrypted payload of the packet. However, CRC-32 is linear, which means that it is possible to compute the bit difference of two CRCs based on the bit difference of the messages over which they are taken. In other words, flipping bit n in the message results in a deterministic set of bits in the CRC that must be flipped to produce a correct checksum on the modified message. Because flipping bits carries through after an RC4 decryption, this allows the attacker to flip arbitrary bits in an encrypted message and correctly adjust the checksum so that the resulting message appears valid.

The initialization vector in WEP is a 24-bit field, which is sent in the cleartext part of a message. Such a small space of initialization vectors guarantees the reuse of the same key stream. A busy access point, which constantly sends 1500 byte packets at 11Mbps, will exhaust the space of IVs after 1500*8/(11*10^6)*2^24 = ~18000 seconds, or 5 hours. (The amount of time may be even smaller, since many packets are smaller than 1500 bytes.) This allows an attacker to collect two ciphertexts that are encrypted with the same key stream and perform statistical attacks to recover the plaintext. Worse, when the same key is used by all mobile stations, there are even more chances of IV collision. For example, a common wireless card from Lucent resets the IV to 0 each time a card is initialized, and increments the IV by 1 with each packet. This means that two cards inserted at roughly the same time will provide an abundance of IV collisions for an attacker. Worse still, the 802.11 standard specifies that changing the IV with each packet is optional.

3.2 MAC Address Filtering

As part of the 802.11b standard, every WiFi radio has its unique Media Access Control (MAC) number allocated by the manufacturer. To increase wireless network security, it is possible for an IT manager to program a corporate WiFi access point to accept only certain MAC addresses and filter out all others. The MAC control table thus created works like "call blocking" on a telephone: if a computer with an unknown MAC address tries to connect, the access point will not allow it. However, programming all the authorized users' MAC addresses into all the company's access points can be an arduous task for a large organization and can be time consuming — but for the home technology enthusiast it can be quite effective.

It is also possible for a dedicated hacker to "spoof" a MAC address, by intercepting valid MAC addresses and then programming his or her computer to broadcast using one of those. Despite that, for small network installations, using a MAC filtering technique can be very effective method to prevent unauthorized access. [2]

3.3 Possible Attacks on Standard Security

The attacks described below are targeted at WEP.

3.3.1 Passive Attacks To Decrypt Traffic

The first attack follows directly from the above observation. A passive eavesdropper can intercept all wireless traffic, until an IV collision occurs. By XORing two packets that use the same IV, the attacker obtains the XOR of the two plaintext messages. The resulting XOR can be used to infer data about the contents of the two messages. IP traffic is often very predictable and includes a lot of redundancy. This redundancy can be used to eliminate many possibilities for the contents of messages. Further educated

guesses about the contents of one or both of the messages can be used to statistically reduce the space of possible messages, and in some cases it is possible to determine the exact contents.

When such statistical analysis is inconclusive based on only two messages, the attacker can look for more collisions of the same IV. With only a small factor in the amount of time necessary, it is possible to recover a modest number of messages encrypted with the same key stream, and the success rate of statistical analysis grows quickly. Once it is possible to recover the entire plaintext for one of the messages, the plaintext for all other messages with the same IV follows directly, since all the pair wise XORs are known.

An extension to this attack uses a host somewhere on the Internet to send traffic from the outside to a host on the wireless network installation. The contents of such traffic will be known to the attacker, yielding known plaintext. When the attacker intercepts the encrypted version of his message sent over 802.11, he will be able to decrypt all packets that use the same initialization vector.

3.3.2 Active Attack to Inject Traffic

The following attack is also a direct consequence of the problems described in the previous section. Suppose an attacker knows the exact plaintext for one encrypted message. He can use this knowledge to construct correct encrypted packets. The procedure involves constructing a new message, calculating the CRC-32, and performing bit flips on the original encrypted message to change the plaintext to the new message. The basic property is that RC4(X) xor X xor Y = RC4(Y). This packet can now be sent to the access point or mobile station, and it will be accepted as a valid packet.

A slight modification to this attack makes it much more insidious. Even without complete knowledge of the packet, it is possible to flip selected bits in a message and successfully adjust the encrypted CRC (as described in the previous section), to obtain a correct encrypted version of a modified packet. If the attacker has partial knowledge of the contents of a packet, he can intercept it and perform selective modification on it. For example, it is possible to alter commands that are sent to the shell over a telnet session, or interactions with a file server.

3.3.3 Active Attack from Both Ends

The previous attack can be extended further to decrypt arbitrary traffic. In this case, the attacker makes a guess about not the contents, but rather the headers of a packet. This information is usually quite easy to obtain or guess; in particular, all that is necessary to guess is the destination IP address. Armed with this knowledge, the attacker can flip appropriate bits to transform the destination IP address to send the packet to a machine he controls, somewhere in the Internet, and transmit it using a rogue mobile station. Most wireless installations have Internet connectivity; the packet will be successfully decrypted by the access point and forwarded *unencrypted* through appropriate gateways and routers to the attacker's machine, revealing the plaintext. If a guess can be made about the TCP headers of the packet, it may even be possible to change the destination port on the packet to be port 80, which will allow it to be forwarded through most firewalls.

3.3.4 Table-based Attack

The small space of possible initialization vectors allows an attacker to build a decryption table. Once he learns the plaintext for some packet, he can compute the RC4 key stream generated by the IV used. This key stream can be used to decrypt all other packets that use the same IV. Over time, perhaps using the techniques above, the attacker can build up a table of IVs and corresponding key streams. This table requires a fairly small amount of storage (~15GB); once it is built, the attacker can decrypt *every* packet that is sent over the wireless link.

4. OPTIONAL SECURITY FEATURES

The above mentioned methods come standard with all WiFi compliant devices. Seeing that they are not all together too secure and provide a poor means of authentication, a layered approach can be used. Other standard network security methodologies such as VPN (to tunnel out of the network), Firewalls (blocking un-wanted traffic), Kerberos (Authentication), RADIUS (for authentication and authorization) just to mention but a few, can be used in conjunction with the standard security features (i.e. WEP and MAC filtering).

Even though there are standard security features and additional layered security features that can be added to further protect the WiFi communication environment, a lot of networks are set up in organizations without any sort of security mainly because of the amount of work involved in actually setting up the extra features (e.g. in the case of MAC filtering setting up each MAC address on each access point) and maintaining these extra features. Poor or no security “open” wireless networks are popular and a fine art of “war driving” is developing in most modern cities that use wireless communication in business areas. Currently there is no “silver bullet” solution (i.e. simple authentication and to the problem of security on wireless networks).

5. SECURE PROXY APPROACH

To overcome some of the above mentioned problems, we have come up with the secure proxy approach. This approach uses the wireless infrastructure network making one modification by placing a server between the access point and the outside network.

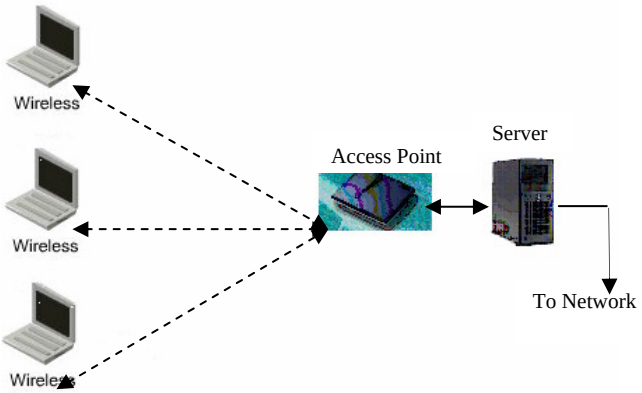

Figure 3: Secure Proxy Approach

The server has 2 major roles; (i) to authenticate wireless clients onto the network and (ii) encrypt traffic traveling from all the clients in un-trusted medium (i.e. air). This setup will work using a farm of access points connected together via a hub or switch which is also connected to the server (i.e. traffic entering or leaving the external network has to pass through this server).

To achieve this, a client / server model has been created. A client module has to be run on all the wireless clients, and a server module on the server. The traffic between the client and server modules is what is to be encrypted. We decided that only certain network services need to be "secured" in this manner and these are HTTP, FTP, mail (POP/IMAP/SMTP). These protocols are the most popular network services used by wireless clients, yet they do not have any inherent security features. Most client network applications support HTTP proxies; these applications can be run using the secure proxy. If a protocol currently not supported is required, an additional proxy could be created for the server and client sides and plugged into the respective handlers.

On the wireless client, the applications using the above mentioned protocols will point to a local proxy (i.e. the client module) which will accept the traffic, encrypt it using a shared session key and send it to the server module. The server module will decrypt the traffic and send it to the outside network.

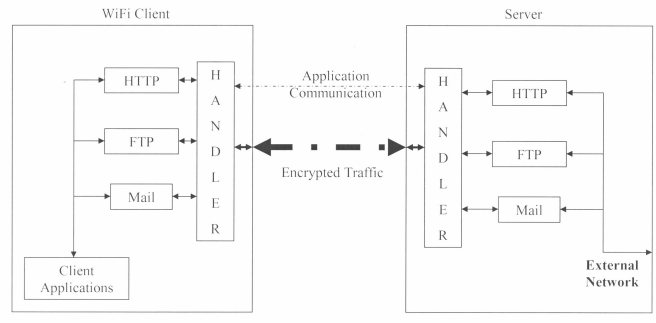

Figure 4: Secure Proxy Overview

From figure 4, it can be seen that there are handlers on both the client and server modules. These handlers are used to control the respective modules and proxies.

Due to the vast number of client machines and platforms at present that support WiFi, it would require a very flexible programming language to support all these various client machines. For this purpose, we are using Sun Microsystems's Java to program both the client and server modules.

5.1 Security

As previously stated, the goal is to encrypt network traffic that travels in the insecure medium. An encryption module has been developed, and it is designed to plug easily into the client and sever proxy modules.

This secure proxy will use two encryption schemes, namely public key encryption and shared key (secret key) encryption. The public key encryption will be used during the authentication of the wireless clients, and also to share the session key. The public key algorithm to be used is RSA, which will use 2048bit strength keys. The shared key algorithm that will be used is AES (Advanced Encryption Standard), which a key of 256 bit strength.

To achieve two way authentication[1] the client uses a user name and password. The server, however, uses a signed public key. The public key of the server is signed by a trusted third party, i.e. that both the server and the clients trust. Therefore, during the authentication process, the client can verify if the public key is indeed from the actual server.

Once the client starts a session, it gets authenticated by the server module. During the authentication process, the client creates a special key to be used by that one client for that one session, referred to as a shared session key. The conventional encryption scheme (AES) uses this shared session key to encrypt the data that is flowing to and from the client and server modules.

DES, Triple-DES (3DES), Blow Fish and Two Fish algorithms can be used to produce the shared session key and perform the conventional encryption. AES has been used due to its increased key strength.

5.2 Authentication

These are the steps followed during authentication:

1. The handlers initially establish a connection.
2. The server sends its signed public key to the client handler
3. Client handler verifies integrity of public key and generates the shared session key[2]. The client handler encrypts the username, and password with the session key, and encrypt the session key with the server public key. This bundle of information is called the authentication code, which is then sent to the sever.
4. The server will decrypt the authentication code, and get the session key. Using the session key it will decrypt the username and password information. The server will then check if the username and password information is valid[3]. If it is not valid, the server will send a termination signal to the client handler and the channel is closed, otherwise the server will send an acknowledgement encrypted using the session key to the client.

[1] Client authenticating to the server and vice versa

[2] The key that is just going to be used for this one session

[3] All user information is stored on the server in a plain text file

Once a client is successfully authenticated, the shared session key is used to initialise the proxies.

This secure proxy approach moves the security layer higher up in the OSI network architecture to the application layer. Due to this, the main disadvantage will be the efficiency and speed of the network communication. However, what is gained is a flexible security policy which can be used in conjunction with a variation of technologies to enable secure communication.

As with other forms of security on WiFi, the secure proxy can use a layered approach. Perhaps a firewall can be installed on the server to handle all the traffic, and the firewall can decide what traffic goes onto the external network. Services not supported by the secure proxy can be allowed to access the external network.

6. Problems Encountered

There have been fairly a large number of problems encountered during the progress of this project. The programming language used (Java™) has limitations which had to be overcome. For example, the standard edition of Java comes with support to generate private/public key pair, but does not come with the libraries to support the encryption and decryption of data using public or private keys. For this, we needed to use a third party open source security provider named "*Bouncy Castle*" [3]. Another limitation on the java programming language has been the key size for conventional encryption algorithms. Sun provides unlimited strength key support policy files which un-locks the restriction. This was initially put in place by SUN to restrict the export of encryption technologies to certain countries.

7. LIMITATIONS AND EXTENSION TO WORK

The secure proxy only works for certain network services; there is room for extension by adding more proxies. Perhaps creating a Public Key Infrastructure (PKI) and issuing all the client machines with a public/private key pair could enable mutual authentication in an ad-hoc WiFi network, and perhaps even encrypting network traffic with shared session key.

8. CONCLUSION

Wireless networks are inherently insecure. Depending on what kind of information is accessed on the network, different security measures can be used. As expected, the more layered security features used, the more inefficient the network would become. The secure proxy solution provides a medium security level. Using large key sizes for encryption and hashed message authentication codes, intercepting traffic or spoofing traffic will prove to be a very tedious task. All current encryption can be broken, its just a matter of time. If the time needed to break the encryption is greater than the value of the information, then the encryption system can be considered secure.[4]

9. REFERENCES

[1] (In)Security of WEP Algorithm
http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html

[2] WiFi Security at Work and on the Road
http://www.wi-fi.org/OpenSection/secure.asp?TID=2#MAC

[3] Legion of the Bouncy Castle
http://www.bouncycastle.org/

[4] Applied Cryptography by Bruce Schneier