University of Waterloo
Faculty of Engineering

Circumventing the Wired Equivalent Privacy Protocol

Pitney Bowes Inc.
35 Waterview Drive
Shelton, CT
USA, 06484

Michael Jarrett
Student #99318764
msjarret@engmail.uwaterloo.ca
Computer Engineering - Term 2B

December 12^{th}, 2001

Michael Jarrett
2646 Standish Drive
North Vancouver, B.C.
Canada, V7H 1N1

December 12th, 2001

Dr. Anthony Vannelli, Chair
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario
Canada, N2L 3G1

Re: 2B workterm report for Michael Jarrett (#99 318 764)

Dear Dr. Vannelli,

Attached is the report "*Circumventing the Wired Equivalent Privacy Protocol*", submitted as my second work report. My last academic term was 2B, after which I began a co-op work term at *Pitney Bowes Incorporated*, working for the *Secure Systems* group of *Advanced Concepts and Technology.*

The purpose of the Secure Systems group is to act as security consultants to other groups within Pitney Bowes, and to that end, to keep abreast of the latest technical issues in the area of security. This report's goal is examine the security of modern wireless networks to determine the level of security risk using them internally or in Pitney Bowes products would entail.

Assistance for this report was given by Matthew Campagna of the Secure Systems team, who provided direction to the analysis, as well as sample code to demonstrate various forms of cryptographic attacks. The report was prepared for him, and for Monroe Weiant, director of the Centers of Excellence of Advanced Concepts and Technology.

I hereby confirm that I have received no further help other than what is mentioned above in writing this report. I also confirm this report has not been previously submitted for academic credit at this or any other academic institution.

Sincerely,

Michael Jarrett
99318764

Contributions

The Secure Systems team consists of seven full-time employees. At the current time, there are also three co-op students: myself, a Systems Design student from University of Waterloo, and a Computer Science graduate student from Rensselaer Polytechnic Institute. The specialties of the team members vary greatly, but include crypographers, mechanical engineers, and various computer specialists.

Our purpose within Advanced Concepts and Technology is to research security-related technology, and apply this knowledge to enhance the security of Pitney Bowes products. This requires Secure Systems to constantly research and experiment with new technologies to look for potenital security flaws, or in some cases, uses where the new technology can be used to enhance security measures. Just as important is a need for the individual Secure Systems team members to enhance their own and other team members' knowledge of security issues in modern technologies.

My initial task on the Secure Systems team was to attempt to design an attack on the encryption and authentication protocol used in Institute of Electrical and Electronics Engineers specification 802.11 for wireless networks. A theoretical attack against the encryption method used had already been published, and later clarified by the report's authors Adi Shamir, Itsik Mantin, and Schott Fluhrer. A working demonstration was required to showt the effectiveness of the attack in real life scenarios. Later, the findings would have to be thoroughly documented, and the results saved in a form suitable for another member of Secure Systems to present to management and non-technical personnel. This report serves, along with computer source code and end-user documentation for the K80211Crack tool, as the record of this acquired knowledge.

Matthew Campagna and Brad Hammell of the Secure Systems team contributed to my investigation: they presented computer source code that demonstrated the mathematical basis of the attack on simulated data, and gave much of the direction to the project.

While not directly related to Pitney Bowes core products or services, wireless communication will be a critical component of many new services and products used and offered by Pitney Bowes. A good example of this is the wireless networks run as part of the 'Less Paper Zone', where an open, collaborative work space was created with the goal of

eliminating the need for paper documents. This network was indeed using the security measures built into most wireless devices, and as such, this work was particularly relevant in this case. This also affects the research of other teams in Advanced Concepts and Technology, for example research into 'enlivened environments', Bluetooth and other alternative wireless communications protocols, and in some cases, even radio frequency ID tags.

One of the most important requirements of my work at Pitney Bowes is the "transfer of knowledge" - the ability to break down the information into core concepts and distribute it to the rest of the team. Also, documentation is key, so that the discoveries made here can be recalled quickly after I have left Pitney Bowes should they be needed.

As wireless networking becomes more popular, an understanding of the security issues involved will be critical for the development of new technology. Gaining an understanding of wireless security issues will give the Secure Systems team an advantage when dealing with technologies which could use this.

Summary

This report investigates the security measures of the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standard for wireless networking. The 'Wired Equivalent Privacy' (WEP) protocol is the only form of security provided in the majority of wireless network devices.

The Wired Equivalent Privacy protocol is based upon the RC4 stream cipher for encryption, but it is used in a manner which exposes many of its weaknesses. One of the most critical, the Fluhrer, Mantin, and Shamir attack, allows one to compromise the wireless network and recover the secret key used to protect it in a short period of time.

This was tested by designing a tool that implemented this attack. A device was found that would allow easy recovery of wireless traffic, and drivers for this device were freely available for the Linux operating system. The K80211Crack program was written to determine the topology of a wireless network and to recover the secret key used to protect it.

The finished K80211Crack tool determined that the attack worked, and could be optimized to function even faster than reported by other groups performing the same research. An effective active data gathering approach was also found against Lucent-based cards. As such, it was concluded that the Wired Equivalent Privacy protocol could protect a network using one key for a period of hours, but after that it would be likely that an intruder could gain full access.

Recommendations for those considering, or already having implemented wireless networks were for additional layers of security: network layer security, third-party key management systems, and firewalls or other network access controls.

Conclusions

The results of tests with the K80211Crack tool allows one to draw two conclusions about the security of a WEP-encrypted wireless network.

First, WEP will successfully discourage the casual and curious observer. Cracking a WEP key requires certain hardware and software, and more importantly, requires a time commitment of hours to days depending on the sophistication of the attack used and the amount of activity on the network in question. As long as the number of networks that do not use WEP remains high, the use of WEP will often be sufficient to discourage the casual potential observer.

Secondly, a WEP key only provides protection for a short period of time. A network adminstrator designing a wireless network must assume that the network will become open to public viewing and use in a short period of time, possibly in as little time as a day, if the key is not changed. While WEP does add a measure of security to a network, it would be a serious folly to rely solely upon it.

For those who use wireless networks in an environment where sensitive data may be transferred over a network, or there are access controls to prevent unauthorized access to network resources, the network security is undoubtedly at risk due to the flaws in WEP.

Demonstration of such flaws can be made with the K80211Crack tool designed for this purpose - which is a user-friendly tool demonstrating the ability to break the security of a wireless network using off-the-shelf hardware, freely obtainable software, and published cryptographic techniques.

Recommendations

Unfortunately, the alternatives to 802.11 wireless networks number very few. While WEP is inherently flawed, one can implement other measures to protect a wireless network, while retaining the full set of advantages that wireless networks provide.

- Implement security at the network layer. If a WEP key is compromised, then there can be secure protocols at the network layer to ensure that the attacker still cannot use or monitor the network.

  The IPSec protocol is an example of a reliable protocol available on most platforms that provides security at the network layer, and one that could be recommended for any wireless network.

- Implement a key management system. Several vendors are making initial releases of products to synchronize and update the keys used by the WEP protocol at frequent intervals, which makes the attacks described much more difficult to implement and much more limited in their use.

- If one can control the supply of devices used on a network, mandate the use of devices known to address the security issues put forth by the Fluhrer, Mantin, and Shamir attack.

- Treat any wireless connections as an external network. For most corporate networks, this means that firewalls and other access controls prevent wide-scale access to an internal corporate network.

Table of Contents

List of Figures

List of Tables

1 Introduction

Wireless networks provide a way for computers and other devices to communicate with each other from different or even mobile locations without the use of a connecting wire. These are very desirable attributes for network administrators: they can save large amounts of human effort involved in wiring, modifying, and maintaining network connections between points. This is also a desirable trait for its users since it grants mobility to portable computer users.

The Institute of Electrical and Electronics Engineers (IEEE), anticipating the need for a standard way for wireless devices to communicate with each other, defined the IEEE 802.11[1] standard. This standard details a way for devices to communicate over 2.4GHz radio frequencies, or infra-red transmissions. It also defined a medium access control mechanism which describes the method of controlling who may broadcast at what time, and how wireless devices identify each other and direct data to each other.

The *Wired Equivalent Privacy*[2] protocol (WEP), as defined in the IEEE 802.11 standard, is designed to be an efficient and effective way to provide the same level of privacy afforded a wired network. It defines the method of encrypting, decrypting, and verifying a data stream. WEP is used in the medium access control functions of 802.11 to encrypt higher level network traffic, as well for authentication of devices wishing to join the a wireless network.

802.11 does not provide for any other authentication or encryption method than WEP, nor did they make changes to WEP or its use in 802.11b (a modification to the original 802.11 protocol). This, from a security standpoint, is a very risky move, as any flaw in the protocol would result in the ability to breach a layer of security in almost any wireless network. As will be described, the algorithms chosen for WEP practically guaranteed this flaw.

2 Description of WEP

2.1 General Description

WEP defines a set of bytes sent over the network, or "frame", giving each byte specific meaning. From a higher perspective, WEP provides a method for shared key encryption[1]. WEP does provide for selection between four different keys, but does not provide any method of key management in which a key can be changed without direct user intervention.

WEP is used at the 802.11 Medium Access Control[2] (MAC) level in frames containing network traffic, as well as frames used in authentication. Data frames encrypted with WEP have a flag set in the 802.11 MAC header, and use the frame format defined by WEP for the frame body.

Authentication using WEP is a challenge-response procedure: the verifying station encrypts a random 128-byte sequence, and will consider the requesting station authenticated if that station can decrypt the packet to the original sequence correctly using its own WEP secret key, then send the sequence back to the verifying station.

For data, the higher-level network traffic is encrypted entirely within the WEP frame, providing encryption for everything above the MAC layer between two wireless cards.

This frame body, illustrated in figure 1 consists of a 3-byte *initialization vector*, a byte identifying which one of four stored secret keys are being used, and then data encrypted by the *RC4 stream cipher*[3], using a key consisting of the secret key prepended by the initialization vector. This is followed by a CRC32[4] checksum, also encrypted using the same RC4 stream.

[1] Shared key encryption is encryption based upon a known secret between two parties.
[2] A method of controlling use of a single broadcast medium by multiple devices.

Figure 1: Frame Structure of a Wired Equivalent Privacy Frame

2.2 RC4 Stream Cipher

The RC4 Stream Cipher is a method of encryption based on a shared key between two parties. This key is used to initialize a pseudo-random number generator (PRNG). Encryption is accomplished by bitwise XORing the PRNG values, known as the 'keystream', into the unencrypted data. Decryption is the symmetric operation of XORing the same keystream into the encrypted data.

RC4 state is saved in a permutation of the integers 0 through 255, and two integers, x and y. The permutation initially starts with the numbers 0 through 255 in order from smallest to largest, and x and y both being 0. Let P be the permutation, x and y two state variables equal to 0, K be the secret key, and l the key length.

$$
\begin{aligned}
& x = 0, y = 0 \\
& \text{loop while } x < 256 \\
& y = y + P_x + K_{x \bmod l} \, mod \, 256 \\
& \text{swap } P_x \text{ and } P_y \\
& x = x + 1
\end{aligned}
$$

Both x and y are reset to 0 following this initialization. Once initialized, the PRNG can generate numbers as follows:

$$
\begin{aligned}
& x = x + 1 \, mod \, 256 \\
& y = y + P_x \, mod \, 256 \\
& \text{swap } P_x \text{ and } P_y \\
& \text{random } = P_{(P_x + P_y) \, mod \, 256}
\end{aligned}
$$

This process can continue indefinitely since each generated pseudo-random number generated further modifies the permutation.

The largest effective keysize of RC4 is $log_2(256!) \equiv 1683\, bits$ due to the size of the permutation. However, most applications of private key cryptography do not require keys this large, as keys larger than 64 bits can be considered 'strong' enough to resist even the most determined attacker. Most applications use a key length l on the order of tens to hundreds of bits.

3 Weaknesses in WEP

WEP has several problems that weaken its claims of privacy and security. Almost all of the weaknesses stem from the use of RC4, a cryptographic algorithm completely unsuited to the task of encrypting open network traffic

3.1 Weak Keys

Most devices using WEP on the market use either 64-bit WEP or 128-bit WEP, both referring to the size of the secret key used in RC4, including the 24-bit initialization vector. That means that the effective secret key one would have to brute force [3] is at best 40 bits or 104 bits. Since each encrypted packet provides a 32-bit CRC, one can verify with up to 32 bits of accuracy that a key is correct using only a single packet.

The keysize can be further reduced by the fact that several popular cards use a string as the secret key without hashing:

> Since WEP keys have to be entered manually, we assumed that instead of giving clients a long string of hex digits, a user memorable passphrase would be used. After examining the test wireless cards at our disposal, we determined that the user-memorable passphrase is simply used raw as the key (ie. the ASCII is used; no hashing is done). [5]

Assuming approximately 64 probable characters in an English-language ASCII string, the effective key length becomes $l \times \frac{log_2 64}{8}$, or approximately 30 bits and 78 bits respectively. These are a far cry from the 64-bit and 128-bit encryption schemes promised by these products. While 78-bits of protection can adequately protect traffic from adversaries using the most modern of technology, 30 bits can be broken directly by available computer hardware in a short period of time.

[3] A 'brute force' attack describes a worst-case attack on a form of security in which the attacker attempts to circumvent security by iterating over every possible solution.

3.2 Recovery of the Keystream

The main problem with RC4 is that a keystream is not designed to be reusable, since knowledge of a portion of a keystream can be used to recover the equivalent portion of decrypted data in the next ciphertext stream if the key is repeated.

WEP handles this by prepending 24 bits of random data to the secret key (initialization vector), and sending these 24 bits unencrypted. By changing these 24 bits, the possibility of repeating a keystream can be eliminated for up to $2^{24} = 16,777,216$ packets.

In the real world, the vast majority of wireless devices use a big endian counter to generate these 24-bit initialization vectors. For a single station, this guarantees minimum repetitions, however, with multiple devices active, the chances of repetition increase significantly.

With network traffic, certain bytes of plaintext data can be predicted with a strong probability of success. For example, practically all data packets begin with the sequence AA_{16} AA_{16}, which are used for the logical link layer of the network. If it is known that the majority of packets will be of a certain form, even more of the keystream can be derived from this. As more encrypted data is sent using a repeated initialization vector, the accuracy increases.

Furthermore, you can recover a keystream with a few repetitions of an initialization vector and statistical analysis. Assuming bytes a and b, encrypted to c and d with keystream byte k:

$$c = a \oplus kd = b \oplus kc \oplus d = a \oplus k \oplus b \oplus k = a \oplus b$$

Knowing $a \oplus b$, which is independent of the RC4 keystream, one can use traditional cryptanalysis[4] to decrypt the messages, and eventually they keystream, which can be used to recover all further packets using the same initialization vector.

[4]If the general form of a decrypted message is known, statistical methods can be used to determine likely translations between an encrypted and decrypted message.

3.3 Fluhrer, Mantin, and Shamir Attack

The *Fluhrer, Mantin, and Shamir Attack*[6] describes a method of determining the bytes of the secret key by using the first byte of the keystream with several different prepended initialization vectors to perform a statistical attack.

The first byte of output from RC4 is the value $P_{P_1+P_{P_1}}$ from the RC4 PRNG. According to the report, after the first i steps of the permutation generation, if $i \geq 1$ and $P_1 + P_{P_1} = i + b$, where b is the byte of the secret key you are currently looking to obtain, then with a probability of $e^{-3} \approx 0.05$ none of the three values will change. This is called a 'resolved condition'.

The attacker knows P, x, and y after the first swaps in the PRNG generation caused by the IV and however many of the initial key bytes are known. If a resolved condition exists, then 5% of the time, the values at P_1, P_{P1}, and $P_{P_1+P_{P_1}}$ will remain fixed throughout the rest of the swaps, and therefore the first byte of the keystream will be fixed at this point. Using this assumption, one can find out the effect of the key byte on the next swap by locating the output byte in the known P, and subtracting the known y, and the known P element corresponding to the current step. This essentially reverses the last swap, revealing the particular key byte

It happens that initialization vectors of the form (x ,0xFF, N), where $0 \leq N \leq 255$ and $3 \leq x \leq l + 3$ satisfy the requirements for a resolved condition to reveal byte $x - 3$ of the secret key if the previous key bytes are known. Over 60 of such IVs, at a 5% probability that the values are undisturbed, the true key byte should reveal itself three times, which is enough to distinguish the key byte from random cases.

4 Attacking the WEP protocol

4.1 Monitoring IEEE 802.11 Traffic

The main difficulty in circumventing wireless security is retrieving the data. It is a violation of the 802.11 specification to return WEP-encrypted packets that have not been successfully decrypted to higher layers in the protocol stack. Therefore, a device not satisfying this part of the specification would be required to get the data needed to crack a WEP key.

Research into the topic revealed that the *Intersil PRISM II* chipset, a popular chipset used in wireless cards by several vendors, had firmware support for monitoring wireless traffic. A device driver for the Linux operating system, called linux-wlan-ng[7] allows this mode to be enabled as of the 0.1.10 driver released September 26^{th}, 2001.

For the purpose of experimentation, the Secure Systems group purchased three SMC2632W[8] wireless network cards, for approximately $100 US each. These cards use the Prism II chipset, and as such can monitor wireless traffic.

4.2 Passive attack using Fluhrer, Mantin, and Shamir Attack

The Fluhrer, Mantin, and Shamir Attack allows us to determine the bytes of an RC4 key if the key is prepended by a known initialization vector with specific values, and we know the first byte of the keystream. The byte is cracked by looking for a 'resolved condition' in the key schedule, which happens with a frequency greater than that of a random distribution.

It happens that WEP satisfies the first criterion nicely. The 802.11 specification does not dictate the exact pattern in which IVs change, but suggests that,

> Changing the IV after each MPDU is a simple method of preserving the effectiveness of WEP in this situation.[1]

Far from it, it makes this attack possible. A sampling of several chipsets of cards revealed that the vast majority of them used a big endian counter to modify the IV every frame. This means that every IV required will eventually be transmitted, and can be read in a predictable manner.

By studying unencrypted traffic, the second criterion is even more trivially satisfied. The link layer control identifying data starts with the byte AA_{16} in all frames. By XORing this value with the ciphertext, we get the first byte of the keystream.

4.3 Brute Force Attack

With the powerful computers available these days, it was decided that brute force over selected bytes was indeed a viable option. This is made much easier by the fact that each WEP packet contains an *Integrity Check Value* (ICV), calculated with the CRC32 algorithm[4] over the decrypted data. Using this knowledge, we can verify that a chosen key is correct without knowledge of the plaintext. Therefore, when the majority of key bytes are in a resolved state, the remaining key bytes can be iterated over and the ICV verified for each. This will eventually find a suitable solution and return the completed key without the need to wait for further data.

4.4 Active Attacks

The downside of the Fluhrer, Mantin, and Shamir attack is that it requires massive amounts of traffic to reach the appropriate IVs that allow for cracking key bytes. While allowing for the possibility for detection, an active attack could force a network to generate much higher levels of encrypted traffic in a short period of time. This would allow for gaining access to a network in a shorter period of time.

4.4.1 Address Resolution Protocol Attack

This attack was discovered by monitoring an attempted ping to a Windows 2000 Professional machine with a Lucent WaveLAN/IEEE wireless network device (with an encryption key) by a Linux machine running with an SMC card (without the encryption key).Table 1 summarizes the response, which shows an attempt from the Linux box to find the hardware address of the destination IP of the ping using the Address Resolution Protocol[9] (ARP). A garbled data packet is sent by the Lucent card in response. While the WEP flag is not set, successive repetitions of this ARP request resulted in the incrementing IVs that serve as the most effective indicator of WEP traffic on the Lucent cards.

Table 1: Traffic generated by 'ping' on an encrypted network

SMC_0e:4f:ae	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.0.10? Tell 192.168.0.168
Lucent_1e:b2:fa	SMC_0e:4f:ae	LLC	Undecipherable data frame

The card should not respond with an encrypted packet to decrypted traffic, and the fact that it does suggests a flaw in the driver. This is an extremely dangerous flaw - it allows us to generate a large number of extremely small packets, which means we use less of the total bandwidth than we would in an equivalent attack on IP traffic.

Unfortunately this attack does have some downsides.

- Requires knowledge of an IP address on the network.
- Advertises the MAC address of the card being used in the attack.
- Cannot be done simultaneously with a sniff on the same card - therefore a crack with this attack will require two wireless devices.
- Can be detected by security tools, and the location of the attacker found.
- Exploits a flaw in a particular chipset, and as such will not work against all devices.

5 Designing a Tool to Compromise WEP

It was decided that, to determine and demonstrate the vulnerabilities in WEP, a tool was required to demonstrate some of the weaknesses. This tool was written for use in a Linux environment, using the *K Desktop Environment*[10] (KDE). The choice of the Linux operating system was based on the availability of drivers to monitor network traffic, and the KDE graphical environment due to the ease of application development using the KDE libraries.

The final tool had the following capabilities:

- Monitor standard 802.11 channels for network traffic.
- Operate in an identical manner to live traffic from a wireless network or saved data.
- Determine network layout, including access points, network names, and the access points each station is using.
- Attempt to break a WEP key with the Fluhrer, Mantin, and Shamir attack.
  - Customizable, and adapts to data received.
  - Able to brute force up to a number of bytes specified by the user.
  - Able to save and restore attack state for later continuation.
  - Graphs of collected data.
- Active attacks using a second device.

The tool has an intuitive interface design; a screen capture can be seen in figure 2. A friendly point-and-click interface allows one to demonstrate the idea of, "click the button and the network password comes out."

From a technical perspective, the code for manipulation and cracking of WEP keys is maintained separately from the graphics code in standard C++. As such, the key cracking and packet decryption could easily be ported to other operating systems.

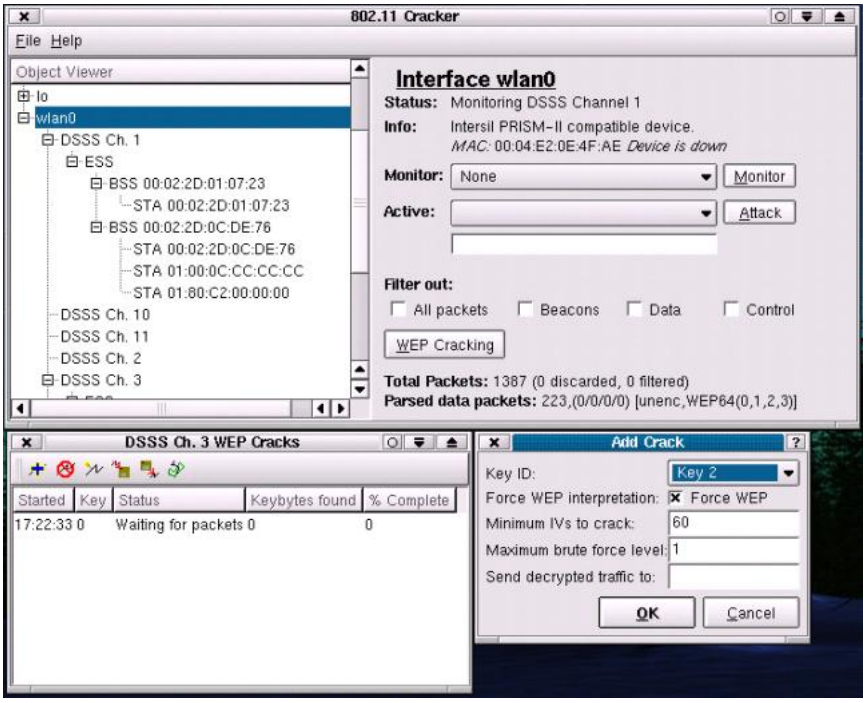

Figure 2: Typical view of k80211crack

6 Observations

6.1 Fluhrer, Mantin, and Shamir Attack

To determine the time it would potentially take to crack a key on a lightly-populated network, an encrypted network at Pitney Bowes consisting of 16 devices was monitored (with permission) between 5pm and 9am. A log was kept of the times where a useful packet was found, which can be found in Appendix 7.2.

The rate of encountering useful packets, even on a multiple-device network, seems to be fairly constant. In this case, $0.1854^{-1} = 5.394$, or approximately 5 useful packets were found per hour of monitoring, and a higher rate would be expected during peak business hours, or on a heavily populated network.

Assuming one requires 60 IVs $\times$ 5 key bytes $= 300$ useful packets, this would require 60 hours to break the key, or approximately 3 days.

6.2 Active Flood

An active ARP flood active attack against a single Lucent WaveLAN/IEEE wireless device was performed to measure the performance of the attack compared to that of simply monitoring the network. The results were logged in the same way as the basic monitoring, and can also be found in Appendix 7.2.

The attack was run until the same number of IVs had been collected as the basic monitoring. The rate of useful packet acquisition was $0.0361^{-1} = 27.70$, or approximately 28 per hour. This is a substantial increase, despite the fact that only one device is being monitored! Assuming the same requirement of 300 useful packets, this attack would take approximately 11 hours to complete - less than a day!

Figure 3 shows a graph of both monitoring and an active attack. Both sets of results are approximately linear, and could be expected to behave uniformly over a larger period or time.

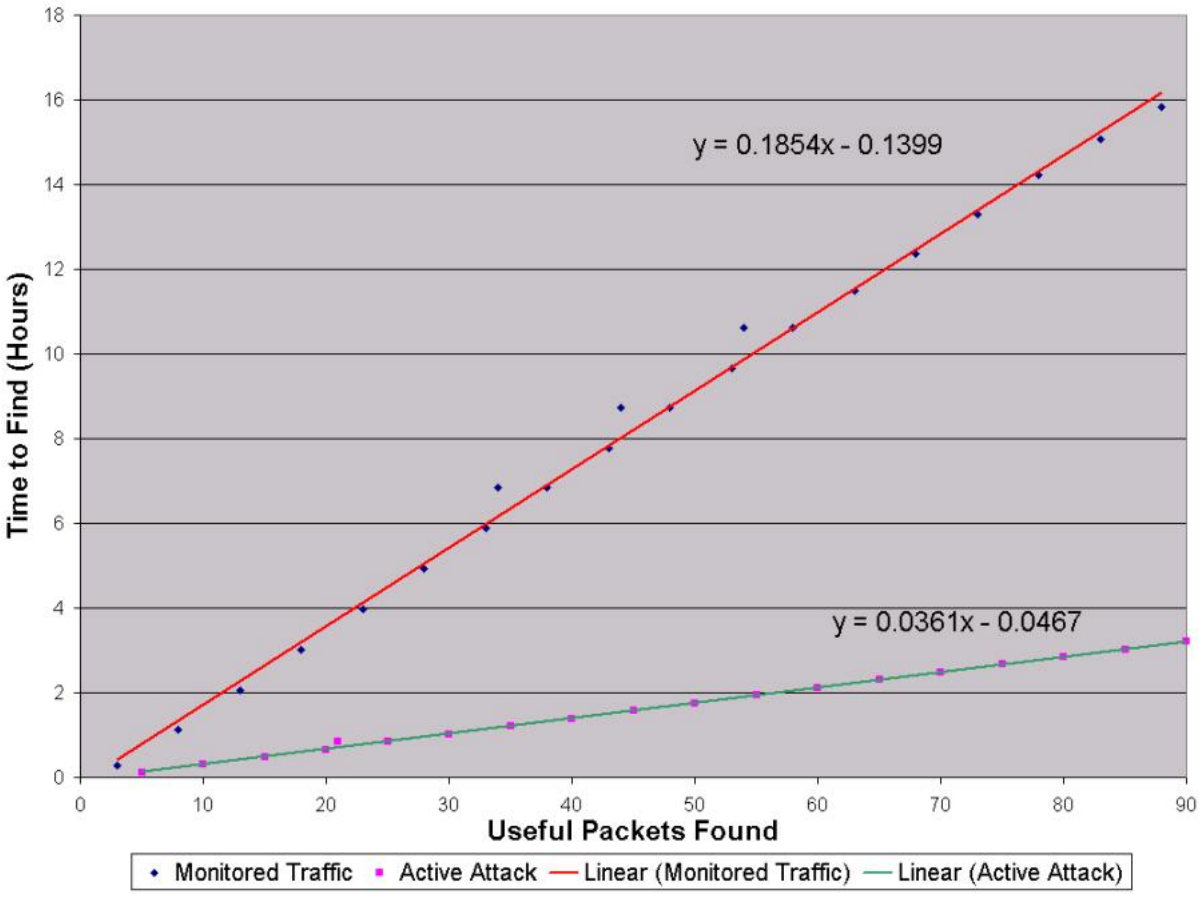

Figure 3: Graph of Useful Packets Versus Time

6.3 Brute Force Enhancement

While the Fluhrer, Mantin, and Shamir attack is an effective way to recover a WEP key, it suffers from the large amounts of data that must be passed over the wire to determine a key accurately. We tested how the processing power of a computer could be used to narrow down these margins.

These tests were done on Pentium III 600MHz machine with 384 megabytes of random access memory. The operating system was Slackware 8.0, running a custom Linux 2.4.9 kernel. This was meant to represent a system within the realm of availability of the casual computer attacker - specialized hardware and optimized software could complete operations in a fraction of the time.

The secret key used was 13:57:9A:CE:F0, all in hexadecimal. Table 2 shows the times

it took to break the trailing N bytes of the key without any statistical data to aid it - using only a single packet for verification purposes. Note that statistics are not shown for times greater than three unknown bytes, as no solution was found in a reasonable amount of time (one day) for four or more bytes.

Table 2: Brute Force Time Measurements without Statistical Data

Unknown bytes	**Trial 1 (seconds)**	**Trial 2 (seconds)**	**Trial 3 (seconds)**	**Average Time**
1	< 1	< 1	< 1	Instantaneous
2	12	12	12	12 seconds
3	2344	2346	2348	39 minutes

As is the nature with most brute force problems, the trend is an exponential increase. The data shows that for three key bytes, brute force is particularly effective, but the inability to reach four shows that brute force would not be a useful attack on its own against even a 40-bit WEP key.

Table 3 shows the same attack, after the collection of 30 useful keystream bytes per byte of the key. Ideally, this is half the data needed for the Fluhrer, Mantin, and Shamir attack.

Table 3: Brute Force Time Measurements with Statistical Data

Unknown bytes	**Trial 1 (seconds)**	**Trial 2 (seconds)**	**Trial 3 (seconds)**	**Average Time**
1	< 1	< 1	< 1	Instantaneous
2	< 1	< 1	< 1	Instantaneous
3	< 1	< 1	< 1	Instantaneous
4	40322	Not measured	Not measured	10 hours
5	40322	Not measured	Not measured	10 hours

This data is a much different shape than the basic brute force, and demonstrates the nature of the addition of statistical data. The most likely explanation is that the particular data analysed happened to have a poor statistical representation of the fourth key byte, which delayed finding the solution for it and later bytes.

Assuming the chance of a poor statistical representation of a particular byte is essentially random, one could easily tune a statistically-aided brute force attack to activate at a certain threshold of collected data, dependant on the computer resources available to

break it. While the success of this optimization is not guaranteed in every instance, this particular key shows that it can succeed in a time well below both brute force and basic monitoring.

7 Possible Solutions

A solution to wireless security must address the issue of these attacks on WEP. IEEE 802.11 defines WEP as the only security measure for wireless networks, so securing a wireless network involves either evading the vulnerabilities in WEP, or implementing further layers of security.

7.1 Avoiding Security Issues in WEP

The most effective form of attack on WEP is the Fluhrer, Mantin, and Shamir attack. However, this only operates on a very small subset of transmitted data. The attack can be negated by either avoiding the initialization vectors used for the attack, or by discarding a fixed number of bytes from the head of the 802.11 keystream. The latter unfortunately suffers from making earlier devices incompatible. The former is only effective if all devices on a network are updated, however does not break interoperability between compliant and non-compliant devices.

Key management services are also available from various vendors, which frequently changes the WEP key used to encrypt packets. If done sufficiently quickly, this negates the major known attacks on WEP, which rely on the secret key being static over a period of time. This suffers from lack of interoperabiltiy with non-compliant devices.

Alternate forms of security, while available, are not part of the current 802.11 standard at the time of writing, so are not considered.

7.2 Further Layers of Security

Security implemented over higher protocol layers can provide a more adaptable and customizable solution than hardware can, and can be distributed in software to avoid compatibility issues between devices.

IPSec[11] is a protocol designed for encryption and authentication at the network layer.

It comes standard with Windows 2000, and is available for a variety of other platforms. It is much stronger than WEP - it allows you to select encryption and authentication methods based upon security needs, and also supports public key security[5].

We tested the IPSec protocol over a wireless-wired bridged network for connecting to both a Linux server and a Windows 2000 server. Communication tests revealed that IPSec could indeed be used over a WEP-encrypted network, and that discovery of the WEP secret key did not allow a wireless station to view traffic or make use of the network.

[5]Encryption based around a 'public' key, and a 'private' key, where one can undo the operation of the other, but one key becomes difficult to obtain using only the other key.

Glossary

active attack Describes the breaking of security that involves the 'attacker' initiating the flow of data to the target that will eventually be compromised.

ARP *Address Resolution Protocol* A protocol used over Ethernet style networks to associate a device address with an IP address.

CRC32 *32-bit Cyclic Redundancy Check* A four byte value used to represent a sequence of bytes, and can be used to detect some errors in a sequence of bytes to which it is supposed to be associated.

IP *Internet Protocol* A common protocol used in most networks, including the Internet. 'IP' is often used to refer to an IP address: a four byte value that identifies a host on an Internet Protocol network.

IPSec A protocol that is used on top of IP that provides a secure tunnel between two hosts, or over an untrusted network between two networks.

IV *Initialization Vector* A known portion of a secret key, normally changing frequently. In the context of WEP, this is three bytes prepended to the secret key. Often used as 'IVs' when talking about the Fluhrer, Mantin, and Shamir attack to refer to the first byte of a packet that has an initialization vector of a specific form.

PRNG *pseudo-random number generator* An algorithm that produces a sequence of numbers that are statistically close to random, but can be reproduced if initial conditions are known.

WEP *Wired Equivalent Privacy* The protocol under the IEEE 802.11 wireless specification responsible for encryption and authentication functions.

References

[1] *ANSI/IEEE Std. 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, LAN/MAN Standards Committee of the IEEE Computer Society, New York, NY, 1999.

[2] *ANSI/IEEE Std. 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, LAN/MAN Standards Committee of the IEEE Computer Society, New York, NY, 1999, pp. 59-69.

[3] "The Algorthm (Stream Cipher) RC4 (ARC-4)", http://www.achtung.com/crypto/rc4.html (current November 2001).

[4] *ANSI/IEEE Std. 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, LAN/MAN Standards Committee of the IEEE Computer Society, New York, NY, 1999, pp. 40-41.

[5] A. Stubblefield, J. Ioannidis, and A. Rubin, "Using the Fluhrer, Mantin, and Shamir Attack to Break WEP," *AT&T Labs - Research*, Florham Park, NJ, August 2001.

[6] Scott Fluhrer, Istik Mantin, and Adi Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4," *Eighth Annual Workshop on Selected Areas in Cryptography*, The Fields Institute, Toronto, Ontario, 2001, pp. 3-19.

[7] "AbsoluteValue Systems, Inc. - linux-wlan Page," http://www.linux-wlan.org/ (current November 2001).

[8] "SMC Networks —— Products —— Product Information," *SMC Networks*, http://www.smc.com/ (current November 2001).

[9] David Plummer, *RFC 826, An Ethernet Address Resolution Protocol*, Network Working Group, November, 1982.

[10] "K Desktop Environment Home (kde.org)," http://www.kde.org/ (current November 2001).

[11] "IP Security Protocol (ipsec)", *The Internet Engineering Task Force*, http://www.ietf.org/html.charters/ipsec-charter.html (current November 2001).

Appendix A - Passive and Active Attack Data

Monitoring		Active Attack	
Time	IVs Found	Time	IVs Found
1010	3	412	5
4079	8	1067	10
7419	13	1723	15
10873	18	2378	20
14334	23	3033	21
17749	28	3034	25
21195	33	3689	30
24610	34	4344	35
24611	38	5000	40
28015	43	5655	45
31397	44	6310	50
31398	48	6965	55
34822	53	7621	60
38226	54	8276	65
38227	58	8931	70
41382	63	9587	75
44495	68	10242	80
47902	73	10898	85
51246	78	11553	90
54281	83		
56987	88		