

SECURITY THROUGH WEP PROTOCOL

Er.Anju Bala, Er.Anupam Garg

(M.Tech.)*

BGIET,Sangrur

E-mail: {anju7k @yahoo.com},{aina_a@rediffmail.com}

Abstract -The 802.11 standard for wireless networks includes a Wired Equivalent Privacy (WEP) protocol to protect link layer communications from eavesdropping and other attacks. However, in its current form, WEP suffers from serious security flaws which arise due to the reuse of the Initialization Vector (IV) and the deployment of an unkeyed checksum for message authentication. This paper attempts to improve the security of the existing WEP protocol without changing its basic architecture. The key exchange scheme proposed in the paper is built on a practical LAN model; it addresses the security concerns of the existing WEP protocol and is easy to implement

I. INTRODUCTION

Wired Equivalent Privacy (WEP) This is the standard 802.11 wireless security protocol for data encryption. It uses a key to encrypt wireless data transmitted through the radio waves. It supports a 40-bit key and a 128-bit key. Attackers have been able to compromise both WEP key lengths. Security of the WEP algorithm This is some information about our analysis of the Wired Equivalent Privacy (WEP) algorithm, which is part of the 802.11 standard.

WEP setup

The 802.11 standard describes the communication that occurs in wireless local area networks (LANs). 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 mobile station (e.g. a laptop with a wireless Ethernet card) and an access point (i.e. a base station). 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 has mechanisms to support such techniques.

The following two sections describe the problems in the algorithm and the technical details of our attacks; they assume some background understanding of cryptographic protocols. You may wish to skip to the following section, which discusses the practicality of the attacks.

II PROBLEMS

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 cipher text. The receiver has a copy of the same key, and uses it to generate identical key stream. XORing the key stream with the cipher text yields the original plaintext.

This mode of operation makes stream ciphers vulnerable to several attacks. If an attacker flips a bit in the cipher text, then upon decryption, the corresponding bit in the plaintext will be flipped. Also, if an eavesdropper intercepts two cipher texts 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 cipher texts 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 cipher texts 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 clear text 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 $1500 * 8 / (11 * 10^6) * 2^{24} = \sim 18000$ 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 cipher texts 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!)

III ATTACKS

Passive Attack 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.

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 $RC4(X) \text{ xor } X \text{ xor } Y = RC4(Y)$. 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.

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.

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.

IV MONITORING

Despite the difficulty of decoding a 2.4GHz digital signal, hardware to listen to 802.11 transmissions is readily available to attackers in the form of consumer 802.11 products. The products possess all the necessary monitoring capabilities, and all that remains for attackers is to convince it to work for them.

Although most 802.11 equipment is designed to disregard encrypted content for which it does not have the key, we have been able to successfully intercept WEP-encrypted transmissions by changing the configuration of the drivers. We were able to confuse the firmware enough that the cipher text (encrypted form) of unrecognized packets was returned to us for further examination and analysis.

Active attacks (those requiring transmission, not just monitoring) appear to be more difficult, yet not impossible. Many 802.11 products come with programmable firmware, which can be reverse-engineered and modified to provide the ability to inject traffic to attackers. Granted, such reverse-engineering is a significant time investment (we have not done this ourselves), but it's important to note that it's a one time cost. A competent group of people can invest this effort and then distribute the rogue firmware through underground circles, or sell it to parties interested in corporate espionage. The latter is a highly profitable business, so the time investment is easily recovered.

EXECUTIVE SUMMARY

We have discovered a number of flaws in the WEP algorithm, which seriously undermine the security claims of the system. In particular, we found the following types of attacks:

- a. Passive attacks to decrypt traffic based on statistical analysis.
- b. Active attack to inject new traffic from unauthorized mobile stations, based on known plaintext.
- c. Active attacks to decrypt traffic, based on tricking the access point.
- d. Dictionary-building attack that, after analysis of about a day's worth of traffic, allows real-time automated decryption of all traffic.

Our analysis suggests that all of these attacks are practical to mount using only inexpensive off-the-shelf equipment. We recommend that anyone using an 802.11 wireless network not rely on WEP for security, and employ other security measures to protect their wireless network.

Note that our attacks apply to both 40-bit and the so-called 128-bit versions of WEP equally well. They also apply to networks that use 802.11b standard (802.11b is an extension to 802.11 to support higher data rates; it leaves the WEP algorithm unchanged).

CONCLUSIONS

Wired Equivalent Privacy (WEP) isn't. The protocol's problems are a result of misunderstanding of some cryptographic primitives and therefore combining them in insecure ways. These attacks point to the importance of inviting public review from people with expertise in cryptographic protocol design; had this been done, the problems stated here would have surely been avoided.

REFERENCES

IEEE. (1999). "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. (IEEE Standard 802.11)".

Kismet (<http://www.kismetwireless.net/>) 802.11 layer2 wireless network detector, sniffer, and intrusion detection system.

Airsnort (<http://airsnort.shmoo.com/>)

