Sean Whalen

Analysis of WEP and RC4 Algorithms

March 2002 - Revision 1

http://www.chocobospore.org

[b a c k g r o u n d]

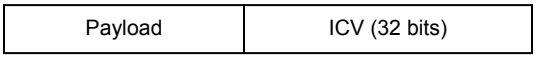
The Wired Equivalent Privacy protocol (WEP) is the standard for authentication and encryption used in the 802.11b wireless Ethernet protocol. For encryption, WEP uses the RC4 algorithm.

Unfortunately, due to the poor design and implementation of WEP it cannot be considered a valid security measure for wireless networks. Vulnerabilities of WEP and RC4 have already been extensively published, so the aim of this paper is to bring together multiple sources and to create a simplified, cohesive resource. I also hope to reach a conclusion on the practicality of cracking WEP in a real-world scenario.

[e n c r y p t i o n]

An 802.11b frame is composed of a header which contains information specific to the 802.11b protocol (source/destination MAC address, type of frame, etc.) as well as a payload. The payload contains a IP header, a TCP header, and a data payload.

When encrypting a frame, the first step is creating an Initial Chaining Vector (ICV), and appending this ICV to the end of the payload. The ICV is simply a 32-bit CRC of the payload contents.

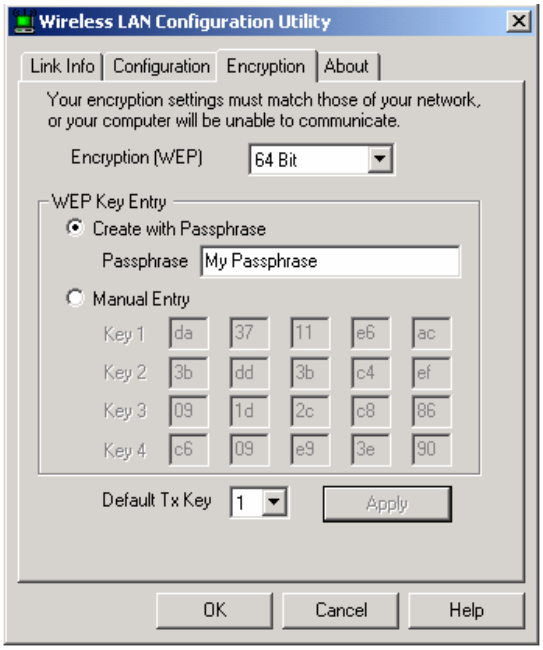


After the ICV is created, a 24-bit Initialization Vector (IV) is generated. The IV is used to initialize the RC4 algorithm prior to encryption. One of four encryption keys (40 bits each, stored on the client) is then selected and this key number (8 bits)

is appended to the IV. In many implementations of WEP, the user can instead chose to use a single 104-bit key for better encryption.

Initialization Vector (24 bits)	Key Number (8 bits)

It is important to note that most keys are generated by software included with the 802.11b card. The software asks the user to enter a phrase and generates a full 40 or 104 bit key based on this phrase, saving the user from manual key entry. Often, client software will list a choice between 64 or 128 bit encryption. This simply refers to the size of the key plus the size of the IV – the key size remains 40 or 104 bits, respectively. A screen capture of the Linksys client software is shown below.



Tim Newsham of @Stake Security has done an excellent job detailing the key generation used by WEP to initialize the RC4 cipher[1]. His presentation "Cracking

[1] "Cracking WEP Keys", Tim Newsham, @Stake Security

WEP Keys" outlines the process, and comes to a shocking conclusion: the 40-bit key generator software used by most vendors results in only 21 bits of entropy or 2^21 possible keys, allowing a brute force exhaustion of the key space in a matter of seconds..

Tim wrote a tool which will perform both wordlist and brute force attacks[2]. A successful wordlist attack takes mere seconds against 40 or 104 bit keys[3]. A brute force exhaustion of the entire 40-bit key space can be done in days on modern hardware, while exhausting the 104-bit key space is beyond most current hardware. However, most people are likely to generate their key using a simple phrase that is likely to exist in a good wordlist.

[d e c r y p t i o n]

Decryption is performed at the destination, which must have the same 4 40-bit keys or 1 104-bit key as the transmitter. Keys are stored internally on the device and can be changed via software. The driver will select the decryption key based on the key number specified in the frame, as discussed above.

Using the IV and key, the RC4 cipher is initialized and its output XOR-ed with the cipher text. A plaintext frame and ICV are the output. As a final check, the CRC

[2] http://www.lava.net/~newsham/wlan/
[3] "Practical Exploitation of RC4 Weaknesses in WEP Environments", David Hulton

of the plaintext payload is re-calculated and compared to the original CRC stored in the ICV.

[attacking rc4]

The RC4 algorithm is a stream cipher. It is initialized with the IV and key (64 bits together). The output of the cipher is XOR-ed with the payload/ICV to produce the cipher text[4]. A post-encrypted 802.11b frame is mapped below:

802.11b header	IV (24 bits)	Key number	Encrypted payload	ICV (32 bits)

From this, we can see that the IV is transmitted in the clear. If we can guess the first byte of the plaintext, we can run an attack the IV as described by Fluhrer, Mantin, and Shamir in their paper "Weaknesses in the Key Scheduling Algorithm of RC4". Keep in mind that our plaintext payload is not simply the data payload of a packet -- it is an IP header followed by a TCP header, and lastly a data payload. This significantly reduces the difficulty of guessing the first byte of plaintext, as IP headers tend to remain constant.

When the IV and first byte of plaintext are known, information about the encryption key can sometimes be obtained. An IV that meets this condition is referred to as **weak** or **resolved,** and many WEP cracking tools (such as Airsnort) check for the existence of a weak IV. Tests conducted by Stubblefield et. al. show

[4] "Using the "Fluhrer, Mantin, and Shamir Attack to Break WEP", Stubbefield/Ioannidis/Rubin

that between 60 and 256 weak IVs were needed to recover a key. The FAQ[5] and forums[6] for Airsnort confirm these numbers, with users reporting collection of 2 to 6 million packets to find enough weak IVs. Collection time varies from days to months, depending on traffic amounts. Recent optimizations by David Hulton, implemented in his tool dwepcrack[7], have produced successful results with less than 60 IVs and 500,000 packets.

[c o n c l u s i o n]

Three attacks exist against WEP: wordlist, brute force, and statistical analysis. Most keys are generated by a user-specified phrase, making both 40 and 104 bit keys vulnerable to wordlist attacks. In addition, most key generation software is severely flawed, resulting in only 2^21 possible keys which can be brute forced in seconds. Even if a proper key generator is used, 40 bit keys can be brute forced in weeks. 104 bit keys cannot be brute forced in reasonable time.

Statistical analysis is made possible due to the transmission of the Initialization Vector in the clear in combination with a guessable first byte of plaintext. This first byte is almost always guessable due to IP headers remaining static. A package of Perl scripts called WEPCrack[8] exist which demonstrate this attack using simulated IVs. A tool called Airsnort allows cracking in real time. Another tool called

[5] http://airsnort.shmoo.com/faq.html
[6] http://sourceforge.net/forum/forum.php?thread_id=132553&forum_id=104550
[7] http://dachb0den.com
[8] http://wepcrack.sourceforge.net/

dwepcrack uses optimizations to drastically reduce the time needed for successful statistical analysis.

Between the three methods we have discussed, it is now very practical to perform decryption and key recovery on 802.11b frames. In the past, the frame capturing process itself was a major hurdle, but open source wireless sniffers such as Mognet[9] and Ethereal[10] now provide this capability for free.

With both capture and cracking tools readily available, it is likely too late for a cross-vendor solution for 802.11b (and possibly 802.11a[11]). The standards committee for WEP has approved a new technique called 'fast packet keying' which involves generating different keys for each packet[12]. This depends on the vendor to provide and support firmware upgrades, and is more of a band-aid than a fix. The real solution requires acknowledging the disaster of WEP and investing more in the design and testing of future protocols. The safety and privacy of consumers and businesses depends on it.

[9] http://chocobospore.org/mognet
[10] http://www.ethereal.com
[11] http://zdnet.com.com/2100-1107-821805.html
[12] http://www.nwfusion.com/news/2001/1217wlesslan.html