Breaking WEP in less than 60 seconds

Erik Tews Ralf-Philipp Weinmann Andrei Pyshkin
<e_tews,weinmann,pyshkin@cdc.informatik.tu-darmstadt.de>

TU-Darmstadt

04.05.2007

The WEP protocol

- In a wireless environment, data packets should be transmitted independently
- In WEP, all statations share a key (Rk), often called *Root Key* or *Secret Key*
- A per packet key ($K = IV||Rk$), is generated by choosing 3 bytes (IV) somehow random.
- A packet m is then encrypted $c = m \oplus \mathrm{RC4}(K)$. The ciphertext c and the corresponding IV is then transmitted to the receiver.
- The reciver can reconstruct the per packt key and decrypt the packet

What is wrong with that?

- By just concatinating IV and Rk, all packets are encrypted with very similar keys.
- The attacker always known the first 3 bytes of a packet key.
- Example, Rk = *00 01 02 03 04* , packet keys could be

```
48 ac 64 00 01 02 03 04
27 55 06 00 01 02 03 04
bc 56 c6 00 01 02 03 04
```

- The RC4 Stream cipher has some statistical weaknesses

Kleins work on RC4

Let $X = X[0]||X[1]||...$ be a keystream and $K = K[0]||K[1]||...$ be the RC4 key used to generate X.

Theorem

There are functions $f_i : (\mathbb{Z}/256\mathbb{Z})^{i+1} \rightarrow \mathbb{Z}/256\mathbb{Z}$ *with:*

- $\mathrm{Prob}(f_i(K[0], ..., K[i-1], X[i-1]) = K[i]) \approx \frac{1.36}{256}$
- *and for every* $a \neq K[i]$*:* $\mathrm{Prob}(f_i(K[0], ..., K[i-1], X[i-1]) = a) < \frac{1}{256}$

And each function f_i *can be computed very efficiently.*

- If you can recover enough keystreams X with their IV, you can guess the first byte of Rk quiet well
- After you got the first byte, you can repeat it for all other bytes of Rk.

Our work on RC4

Theorem

There are functions $g_{i,j} : (\mathbb{Z}/256\mathbb{Z})^{i+1} \rightarrow \mathbb{Z}/256\mathbb{Z}$ *with:*

- $\mathrm{Prob}(g_{i,j}(\mathsf{K}[0], ..., \mathsf{K}[i-1], \mathsf{X}[i-1+j]) = \sum_{k=0}^{j} \mathsf{K}[i+k] \bmod 256) > \frac{1}{256}$
- *and for every* $a \neq \sum_{k=0}^{j} \mathsf{K}[i+k] \bmod 256$*:*
  $\mathrm{Prob}(g_{i,j}(\mathsf{K}[0], ..., \mathsf{K}[i-1], \mathsf{X}[i-1+j]) = a) < \frac{1}{256}$

And again, each function $g_{i,j}$ *can be computed very efficiently.*

- Instead of guessing bytes of Rk, we can now guess sums of bytes of Rk.
- After having guessed the sums, we can get the keybytes by computing the difference between two sums.

Putting it all together

- In a WEP environment, you can get up to 800 packets per second, perhaps more.
- Using some advanced techniques, it is possible to get enough bytes of the keystream of these packets.
- Using the functions f_i or $g_{i,j}$ allows us to guess the right key, by applying them to many packets.
- Adding some additional error correction helps to improve success probability a lot, if only a few number of packets are available.

Putting it all together

- In a WEP environment, you can get up to 800 packets per second, perhaps more.
- Using some advanced techniques, it is possible to get enough bytes of the keystream of these packets.
- Using the functions f_i or $g_{i,j}$ allows us to guess the right key, by applying them to many packets.
- Adding some additional error correction helps to improve success probability a lot, if only a few number of packets are available.

Putting it all together

- In a WEP environment, you can get up to 800 packets per second, perhaps more.
- Using some advanced techniques, it is possible to get enough bytes of the keystream of these packets.
- Using the functions f_i or $g_{i,j}$ allows us to guess the right key, by applying them to many packets.
- Adding some additional error correction helps to improve success probability a lot, if only a few number of packets are available.

Implementation

- Using some other protocol weaknesses, we can force a network to generate a lot of data packets.
- For these packets, it is easy to guess the keystream.
- Now, our implementation is able to guess the right key in less 3 seconds of cpu-time on an average mobile computer, if enough data packets are available.
- If 40,000 data packets are available, our success probability is around 50%, if 85,000 packets are available, we get the right key in 95% of all cases. (104 Bit keylength assumed)
- For 50% success probability, the whole operation can be completed in less than 60 seconds!

What is left

- The successor protocol of WEP called *WPA* still uses RC4, but exchanges the key more frequently.

  Can this be attacked too?

- Are there other protocols which use RC4 in a WEP like mode?
- For those of you who are intrested:
  Implementation is available at:
  `http://www.cdc.informatik.tu-darmstadt.de/aircrack-ptw/`

What is left

- The successor protocol of WEP called *WPA* still uses RC4, but exchanges the key more frequently.

  Can this be attacked too?

- Are there other protocols which use RC4 in a WEP like mode?
- For those of you who are intrested:
  Implementation is available at:
  `http://www.cdc.informatik.tu-darmstadt.de/aircrack-ptw/`

What is left

- The successor protocol of WEP called *WPA* still uses RC4, but exchanges the key more frequently.

  Can this be attacked too?

- Are there other protocols which use RC4 in a WEP like mode?
- For those of you who are intrested:
  Implementation is available at:
  `http://www.cdc.informatik.tu-darmstadt.de/aircrack-ptw/`