Exercise sheet 3, 28 November 2019

This exercise sheet takes you on a trip to investigate the stream cipher RC4. You can do it with a simple implementation of it in sage or python but it will be much faster (and thus your results will be more meaningful) if you work with a faster implementation, e.g. in C. You can get a simple sage implementation (with hardcoded values) at `http://www.hyperelliptic.org/tanja/teaching/CS19/rc4.sage`. Use `https://tools.ietf.org/html/rfc6229` to check your implementation.
RC4 has a very simple description. It uses a 256 byte state vector S (array of 256 bytes) which contains a permutation of the integers 0,1, ...,255. The key k consists of ℓ bytes, where ℓ is at least 5 and at most 256. For export the shortest keys were used, meaning the strength against brute-force attacks was 2^{40}.

RC4 setup

```
for i = 0 to 255
  S[i] = i

j=0
for i = 0 to 255
  j = (j + S[i] + k[i mod l]) mod 256
  swap values in S[i] and S[j]
```

Generate RC4 output stream

```
i = 0, j = 0
while generating output
  i = (i+1) mod 256
  j = (j + S[i]) mod 256
  swap values in S[i] and S[j]
  c = S[(S[i] + S[j]) mod 256]
  output c
```

1. Take 16 bytes as keylength; vary the key, and plot the distribution of the second output byte over all 256 possible values of that byte.
2. What happens to the output if $S[2] = 0$ at the end of the key-setup stage?
3. Take 16 bytes as keylength; vary the key but keep the first byte of it fixed and plot the first output byte.
4. Take 16 bytes as keylength; vary the first three key bytes and keep the remaining ones constant. Plot the distribution of the third output byte + key[0] + key[1] + key[2] + key[3].
5. Read the specification of WEP (the protocol to connect to routers). How can you use the knowledge from the first three parts to likely break it?
6. Check out the documentation and explanation of Aircrack-ng.