Hack.lu 2005 - The Crypto Challenge

Claus R. F. Overbeck - RedTeam Pentesting
claus.overbeck@redteam-pentesting.de
http://www.redteam-pentesting.de

20th October 2006

Introduction

- Crypto challenge as part of last years *Capture The Flag (CTF)* contest
- Based on a cipher by Peter Thoemmes (thank you for the nice challenge!)
  - weakened for the contest (so don't blame him!)
- Uses 4 byte symmetric key

The Task

What you have:

- You have the plain text `***This is the ...`
- You have the cipher text `F8 72 C2 51 AA 05 82 21 ...`
- You have the software (encrypt/decrypt) and its source

What you want:

- Find the key! (So you can decrypt other protected material)

First Attempt: Brute Force I

Idea: Write a perl skript that tries all keys:

```
1  #!/usr/bin/perl
2  for ($i=1 ; $i<=4294967295 ; $i++){
3    $key=sprintf("%.8x",$i); # change to hex
4    if (`./pitcrypt -k $key -d in.crypt |
5      diff - plain.txt -q` !=
6      "Dateien - und plain.txt sind verschieden."){
7      print "We found the key!: " , $key;
8    }
9    print("Tried another key. Just tried key $key.\n");
10 }
```

How long will this run?

47.5 sec for 500 keys.
2^{32} possible keys $= 4,294,967,296$
$\Rightarrow\approx$ 408021893 seconds
$\Rightarrow\approx$ 6800364 minutes
$\Rightarrow\approx$ 113339 hours
$\Rightarrow\approx$ 4722 days
$\Rightarrow\approx$ 12.9 years
$\Rightarrow\approx$ too long for a CTF!

Wow, only 13 years, ain't that fast?

Photo: Nico van Geldere, Apeldoorn, Holland, wikipedia.org - GNU-FDL

To give you an idea: I will be 40 by then :-)

Second Attempt: Analyse the cipher

Byte-wise XOR

```
          P L A I N T E X T . . . . . .
              XOR
Key →     K E Y S T R E A M . . . . . .
              =
          C I P H E R T E X T . . . . . .
⇒
```

- Plain text, cipher text and key stream allways have the same length.
- We can get the keystream by XORing plain and cipher text.
- We can encrypt and decrypt documents up to the length of the key stream. ⇒ We don't need the key!

Second Attempt: Analyse the cipher

Byte-wise XOR

P L A I N T E X T . . . . . .
XOR
C I P H E R T E X T . . . . . .
=
Key → K E Y S T R E A M . . . . . .
⇒

- Plain text, cipher text and key stream allways have the same length.
- We can get the keystream by XORing plain and cipher text.
- We can encrypt and decrypt documents up to the length of the key stream. ⇒ We don't need the key!

Second Attempt: Analyse the cipher

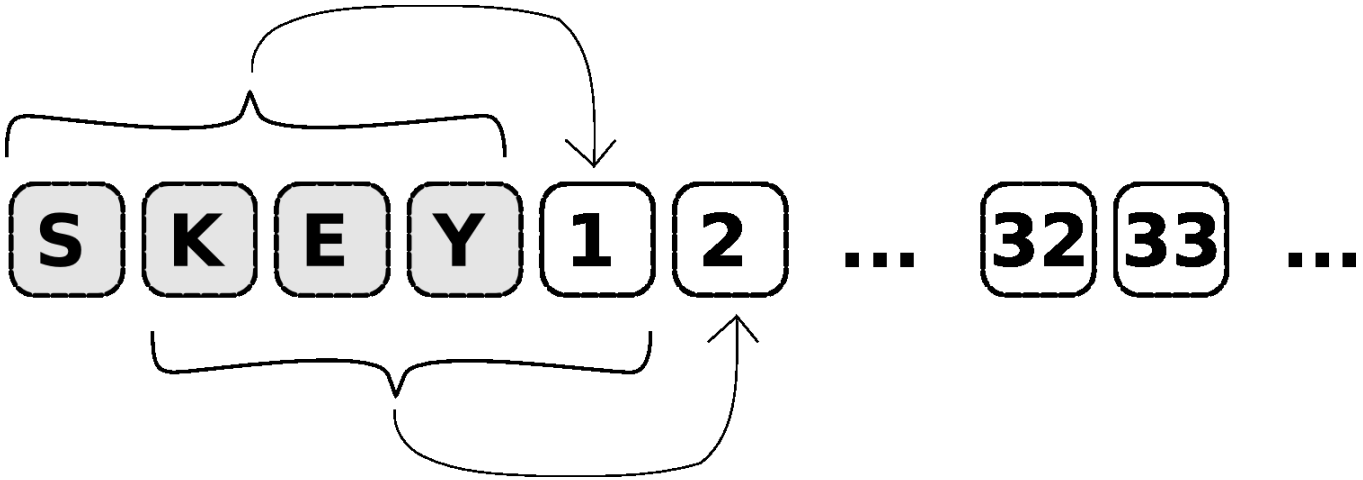
Byte-wise XOR

P L A I N T E X T . . . . . .
XOR
C I P H E R T E X T . . . . . .
=
Key → K E Y S T R E A M . . . . . .
⇒

- Plain text, cipher text and key stream allways have the same length.
- We can get the keystream by XORing plain and cipher text.
- We can encrypt and decrypt documents up to the length of the key stream. ⇒ We don't need the key!

Second Attempt: Analyse the cipher

How is the keystream calculated? (Read the source)



- Every byte of the keystream is calculated from the last four bytes.
- $\Rightarrow$ We only need the first four bytes of the stream.
- We get those with: plain text XOR cipher text
- We can encrypt and decrypt documents with any length

Third Attempt: Brute Force II

Now, what about the real key???

Idea: Do it the C++ way:

- Use the source and change it to try keys (Rewrite main() function)
- Do not write to disk.
- Only encrypt/decrypt 4 bytes and compare

How long will this run?

50 sec for 10.000.000 keys.
2^{32} possible keys $= 4,294,967,296$
$\Rightarrow \approx$ 21.474 seconds
$\Rightarrow \approx$ 358 minutes
$\Rightarrow \approx 5,9$ hours

Key is: 99343628 (Maybe it is just a collision?)

Conclusion

What we learned:

- You don't need to be a math genius to crack a cipher.
- There might be several approaches.
- If you want to do some math you could also try to reverse the key stream function. (Left for you as an exercise ;-)

Questions?

(If there is still time left...)