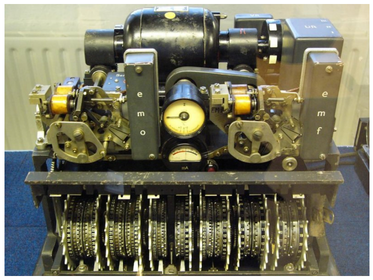
Class 25: Security through Complexity?



Lorenz cipher used in WWII

PS6 is due today.

Karsten Nohl

cs302: Theory of Computation
University of Virginia, Computer Science

Motivation

- Many applications require certain tasks to be *easy* for some and *hard* for others
- Example: Decryption of encrypted message is easy only when given a secret key

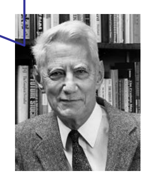
Cryptography is concerned with constructing algorithms that withstand abuse. -Goldreich

Complexity is a powerful tool to "lock out" adversaries.
Basic Idea: Require *hard problem* to be solved, give hint as key.

NP can be useful

- So far, you learnt how to detect "unsolvable" problems (in NP) and solve them anyway by approximation (in P)
- For cryptography we want the opposite: problems that are almost always *hard*, i.e., cannot be approximated in P

[Breaking a strong cipher should require]
"as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type"
- Shannon, '49



Sounds NP-Complete, doesn't it?

Goal: Encryption

- For almost all security schemes we need:
  - Encryption / one-way function

secret key

easy to compute: $enc(x,k) \rightarrow y$

hard to find any part of: $\langle x,k \rangle = enc^{-1}(y)$

Make this an NP problem

- Often also required:
  - Decryption $dec(y,k) \rightarrow x$

Encryption build on Hardness

- Knapsack problem is NP-Complete
  - Problem of filing bag with best selection of items
  - Recall: Reducible from Subset-Sum
- Enable Encryption: Keep message secret by hiding it in a Knapsack instance

bits of encryption key = knapsack instance

$$s = \sum_{i=1}^{n} x_i a_i$$

message bits

Decryption possible by knowing easy knapsack instance (secret key) that provides shortcut.

Flawed Security Argument

- Subset Sum is NP-Complete
- Breaking knapsack cipher involves solving a subset sum problem
- Therefore, knapsack cipher is secure

Flaw: NP-Complete means there is **no fast general solution**. Some instances may be solved quickly.

(Note: Adi Shamir broke knapsack cipher [1982])

Cipher Design

- NP-Completeness is not sufficient for cryptographic hardness
  Worst-case complexity
- Need solution to usually be hard
  Average complexity
- Captured in new complexity class:
  All *tractable* problems are in BPP
  (which only makes sense if P≠NP)

probabilistic: can flip coins

Cipher Design (cont.)

- A "strong" cipher cannot be broken faster than exhaustive key search (brute force)

$$\Theta(2^n) \text{ time}$$

- Only possible shortcut:
  Trade space for time; e.g.:

$$\Theta(2^{n\cdot\frac{2}{3}}) \text{ time + space}$$

Results of Insufficient Hardness

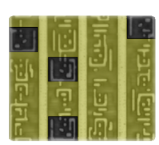
- All broken cipher have a gap between *worst-case* and *average* hardness
- Estimating average hardness is often impossible (= finding best algorithm for instances of NP-complete problem)
- Next: Analyze cipher, identify complexity, and break it by finding tractable average solution.

Proprietary Cryptography

(or: why "security-by-obscurity" never works)

First: Disclosure

- Secret algorithm can often be found:
  - Disassembling software
  - Hardware reverse-engineering



This talk: Breaking a cipher once we found it.

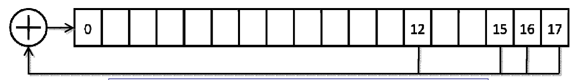
Then: Exploitation

- Most secret ciphers are broken after disclosure
- Flaws are very similar in all DIY ciphers (and cryptanalyst spot them in a glimpse)

"No more weak ciphers. No more paranoia."
Sean O'Neil

The crux of most flaws

- Most weaknesses caused by insufficient *non-linearity*.
- At the heart of the problem:
  LFSRs (linear feedback shift register)



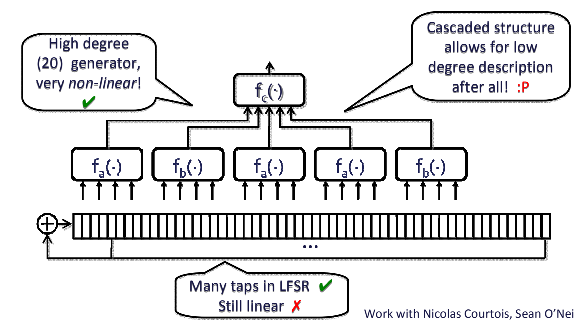

```
tmp  = x[12]^x[15]^x[16]^x[17];
for  i=17:-1:1    x[i]=x[i-1];
x[0] = tmp;
```


Non-Linearity

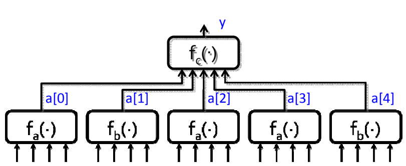
- System of equations that desribes *n*-bit cipher can have up to $O(2^n)$ terms.
- Only $O(n)$ of these terms are linear.

Linear	≈	P
Non-linear	≈	NP

Mifare Crypto-1



Work with Nicolas Courtois, Sean O'Neil



Compute equations for first output bit:

```
a[0] = fa(x[7],x[9],x[11],x[13]);
a[1] = ...
...
y    = fc(a[0],a[1],a[2],a[3],a[4])
```

Before computing next bit, shift LFSR:

```
tmp   = x[0]^...^x[43];
for  i=1:47   x[i]=x[i+1];
x[48] = tmp;
```

Describes cipher as system of equations with 48+r[illegible]5 unknowns, terms with degree ≤ 4!

Almost there ...

1. Describe weak parts of cipher as system of equations
2. Brute-Force through complex parts:
   Guess-and-Determine attack.
3. Solve system of equations:
   `MiniSAT` is our friend

Solving for 48-bit Crypto-1 key takes 12 seconds; compared to month for brute-force.

Lessons Learned (Crypto)

- Obscurity and proprietary crypto add security only in the short-run
  - (but lack of peer-review hurts later)
- Constraints of small devices make good crypto extremely hard
  - Where are the best trade-offs?
  - How much security is needed?
  - How can we best introduce non-linearity?

Lessons Learned (Complexity)

- Cannot rely on hardness of problems; gap between average and worst-case instances often significant
- This is **good news** unless you are building cryptography:
  - Can solve many instances of NP-complete problems in limited time
- Mathematicians have done most of the work already: start using `MiniSAT`

"Security is lax on this side."

Don't forget to hand in PS6.