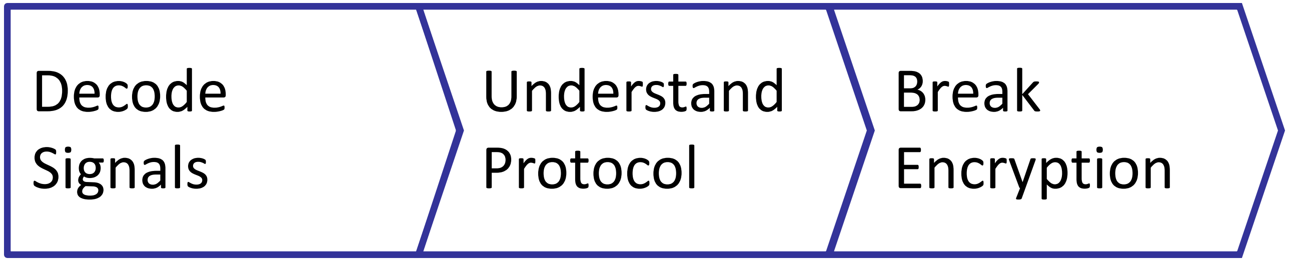
Hitag2 Insecurity

Karsten Nohl
Henryk Plötz

@ HAR 2009

Breaking proprietary RFID technology is a generic process

Pen-Testing a "secure" RFID tag



Decode Signals

Understand Protocol

Break Encryption

Small experiment: Bring up your car key.

Hitag2 widely used?

- Apparently Hitag is used in access control …
  - German government / army access ID
- .. and car keys including these brands:
  - Renault
  - Opel
  - Peugeot
  - Citroen

Hitag2's cipher is highly vulnerable

Attack	Resources	Vulnerable due to
Brute force	2^{48} computations	Small key size
Pre-computation	2^{49} computations, few GB storage	Small key size, lack of tag nonce
Technique explained in the A5/1 talk at 17:00		
Algebraic attack	approx. 2^{35} computations (6 hours)	Low cipher complexity
Focus in the remainder of this talk		

Root weakness of proprietary ciphers can be exploited using generic tool

- Design goal of ciphers: "one way road"
- However, some (stream) ciphers do not build complexity fast enough
- Complexity measurable as ANF randomness (Sean O'Neil: ASD; K. Nohl dissertation)

Low complexity ciphers are reversible using SAT solvers

Cryptographic strength is closely related to *non-linearity*

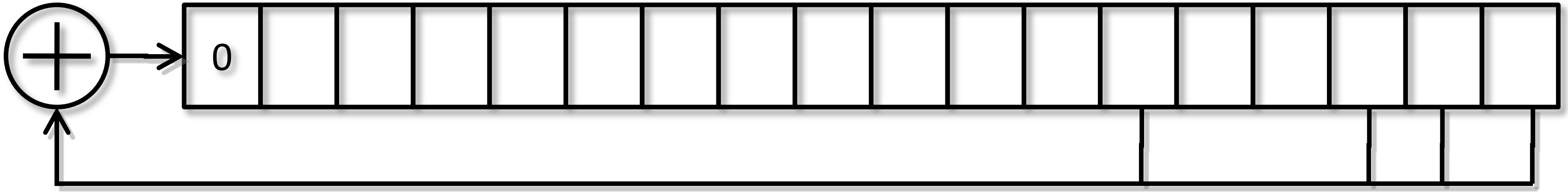
- System of equations that describes *n*-bit cipher can have up to 2^n xor terms.
- Only *n* of these terms are linear.

Linear	≈ P	≈	solvable
Non-linear	≈ NP	≈	not solvable for large keys

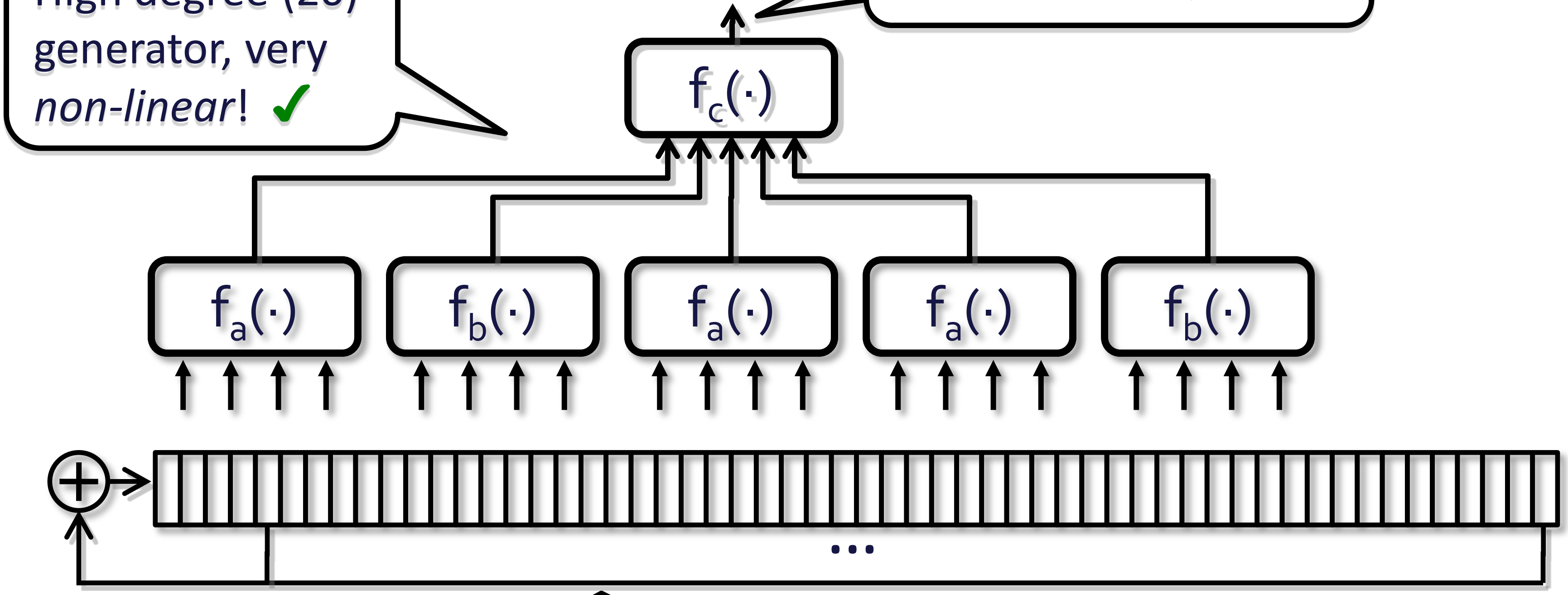
Standard cipher building blocks generate surprisingly little complexity

- Most weaknesses are caused by insufficient *non-linearity*.
- At the heart of the problem:

  LFSRs (linear feedback shift registers)



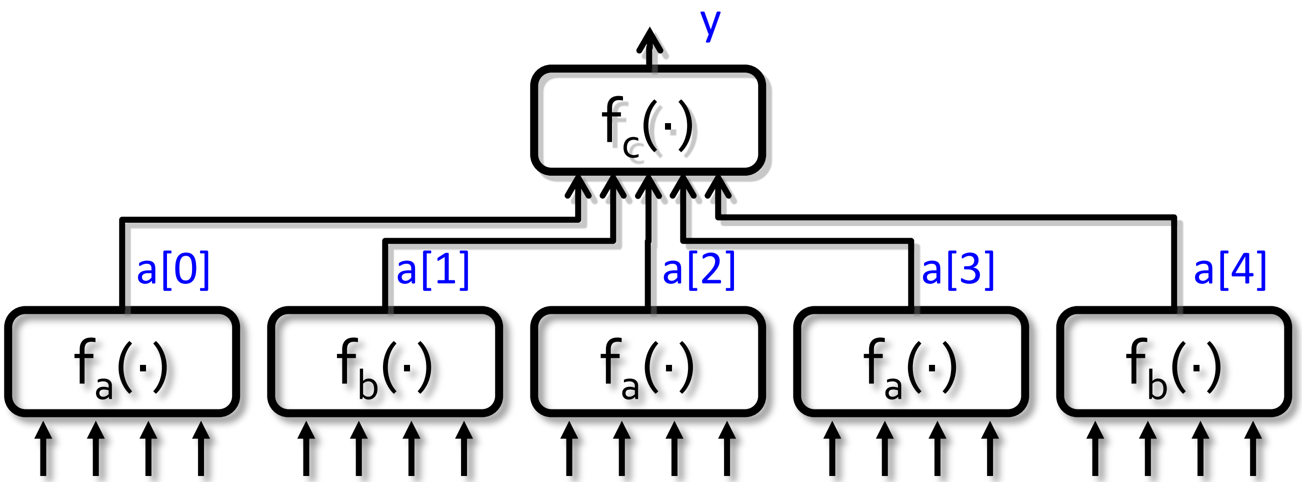
NXP Hitag2 is too linear to be strong



High degree (20) generator, very *non-linear*! ✓

Filter output is not fed back into state ✗

Many taps in LFSR ✓
Still linear ✗



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⍰5 unknowns, terms with degree ≤ 4!

Work with Mate Soos

SAT solver needs a few tweaks to handle Hitag2 equations

- SAT solvers can solve systems of equations, but only when presented as ‘and-of-ors’
- The Hitag2 system of equations is exponentially larger when converting all xors to ‘and-of-ors’

Add xor support to SAT solver

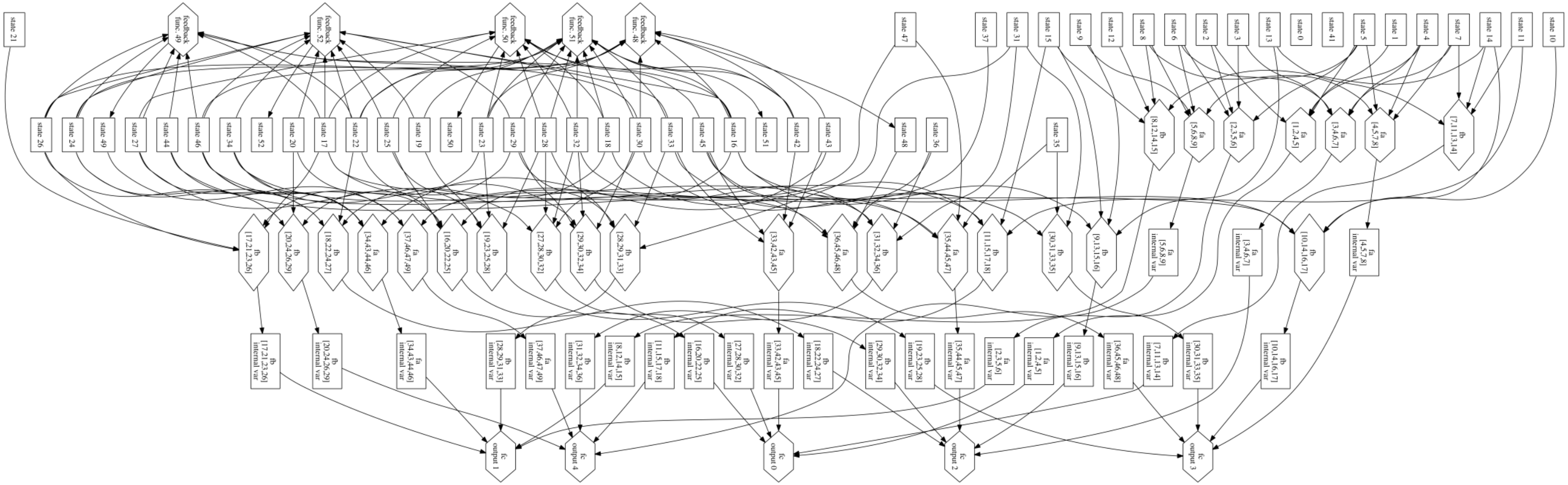
→ Break ciphers

(Released as CryptoMiniSat under GPL)

Work with Mate Soos

Hitag2 is an easy target for CryptoMiniSat

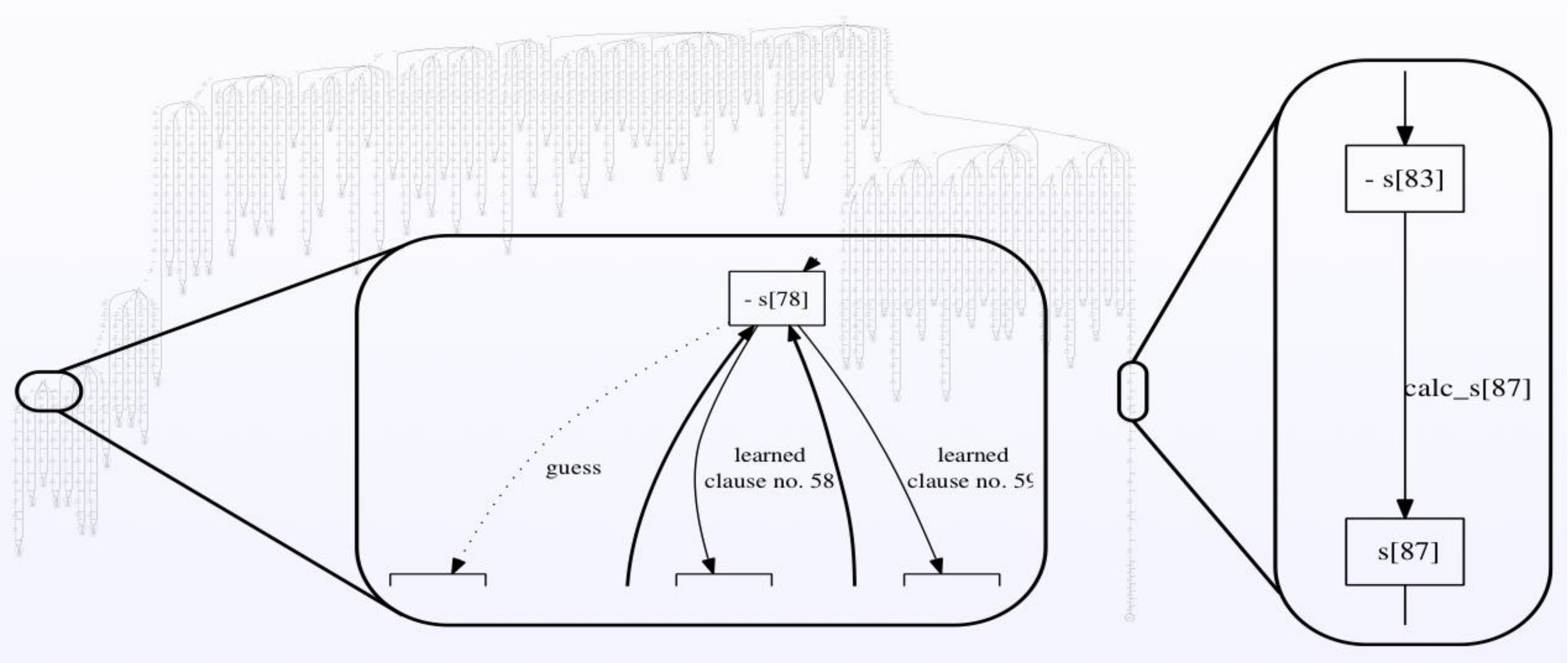
Inside the SAT solver, the system of equations is represented as a circuit of binary functions:



Solvable in <6 hours on a PC

MiniSAT visualization tool available upon request.

SAT solving is *smart* brute force



Tree of key guesses; compare to 2^{48} guesses needed for brute force

Lessons Learned.

- Documenting RFID systems is practical even without costly tools
- There is no point in using proprietary ciphers
  - Huge risk of design flaws
  - Cipher will be disclosed

There are still scores of legacy RFIDs for you to hack

Questions?

http://tinyurl.com/CryptoMiniSat

Slides will be in the Pentabarf.

Karsten Nohl <nohl@virginia.edu>
Henryk Plötz <henryk@ploetzli.ch>