Fault-Algebraic Attacks on Inner Rounds of DES

Nicolas T. Courtois[1]

David Ware[2]

Keith M. Jackson[2]

[1] University College London, UK

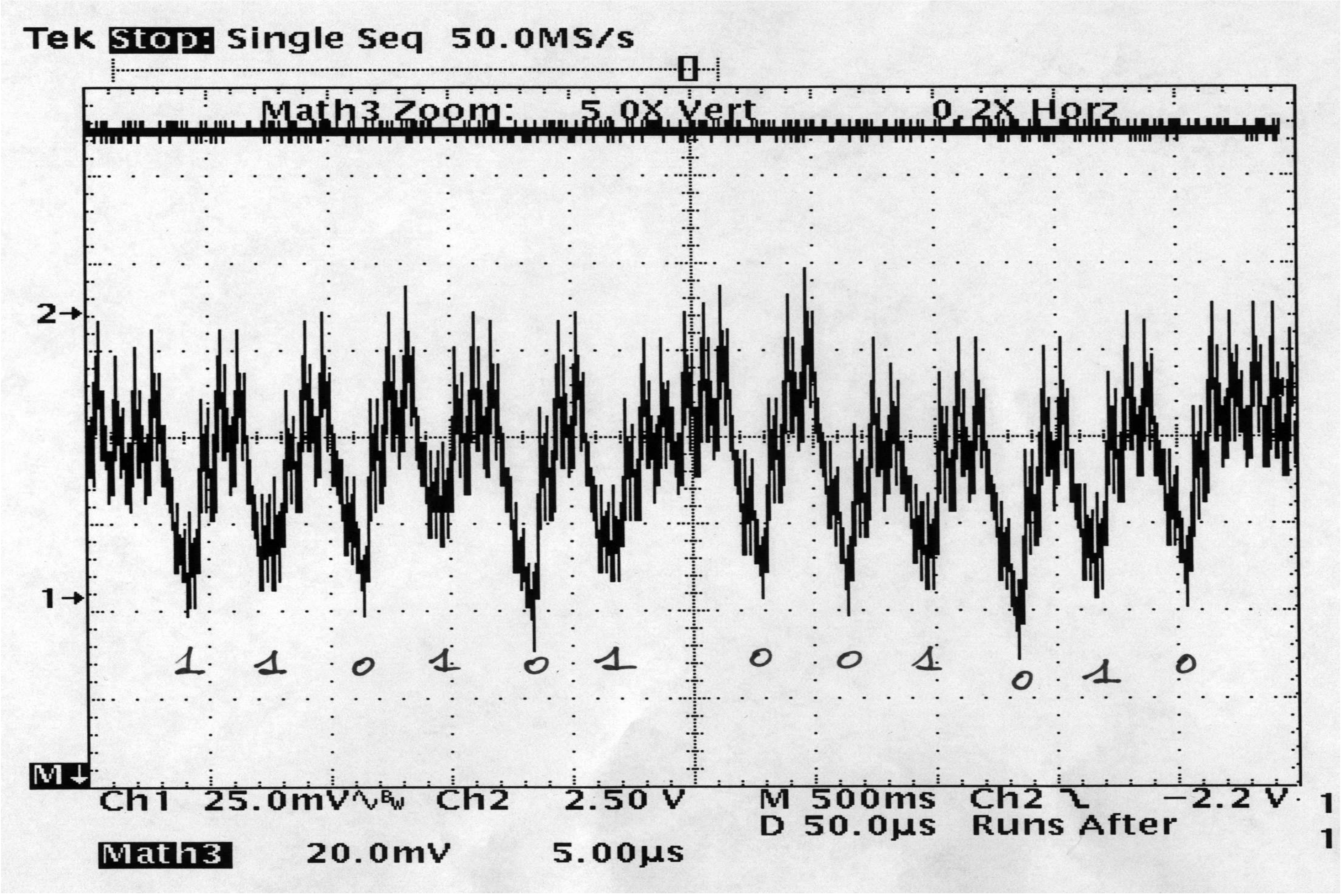
[2] RFI Global, UK

Roadmap

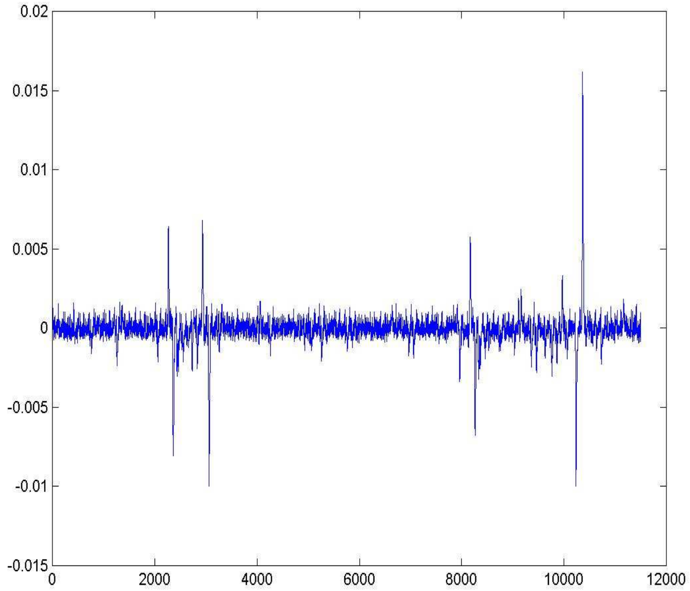
- Side channel attacks and fault attacks
  - Is DFA really a threat? When?
  - What can we do in the lab?
- State of the art for DES
  - Theory vs. practice, what is the right conclusion…
  - Less faults => more cryptanalysis effort!
- Algebraic attacks on symmetric ciphers.
- How to recover keys in the hardest possible cases:
  - for example with only 1 fault in an inner round?

Side Channel Attacks

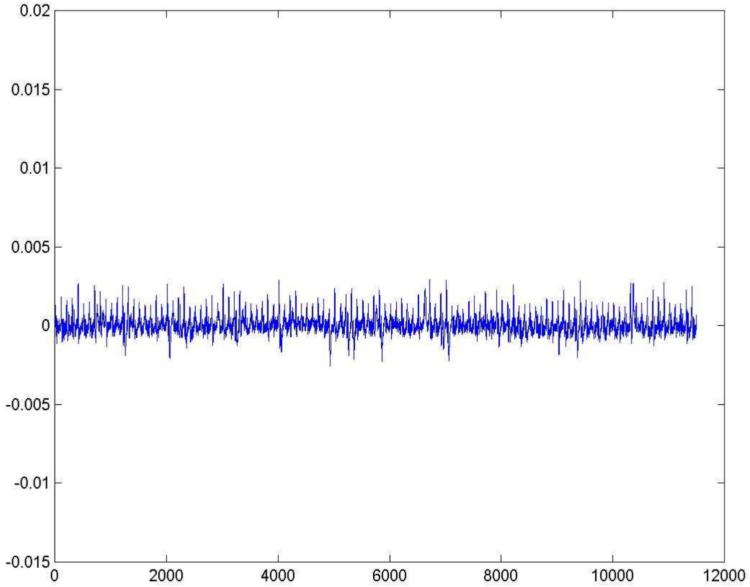
SPA Attacks


Tek Stop: Single Seq 50.0MS/s
Math3 Zoom: 5.0X Vert 0.2X Horz
2→
1→
1 1 0 1 0 1 0 0 1 0 1 0
M↓
Ch1 25.0mV ∿BW Ch2 2.50 V M 500ms Ch2 ⌄ −2.2 V
D 50.0µs Runs After
Math3 20.0mV 5.00µs

DPA Attacks



Right 6 bits



Wrong 6 bits

DFA Attacks…

1. Provoke faults in the device,
2. Deduce the key by detailed mathematical analysis.

DFA Requirements

One needs to be able to run the same crypto algorithm many times with the same inputs.

The inputs do NOT need to be known.

- they usually are, but today we will realistic example when they aren’t (!) and yet the key is found.

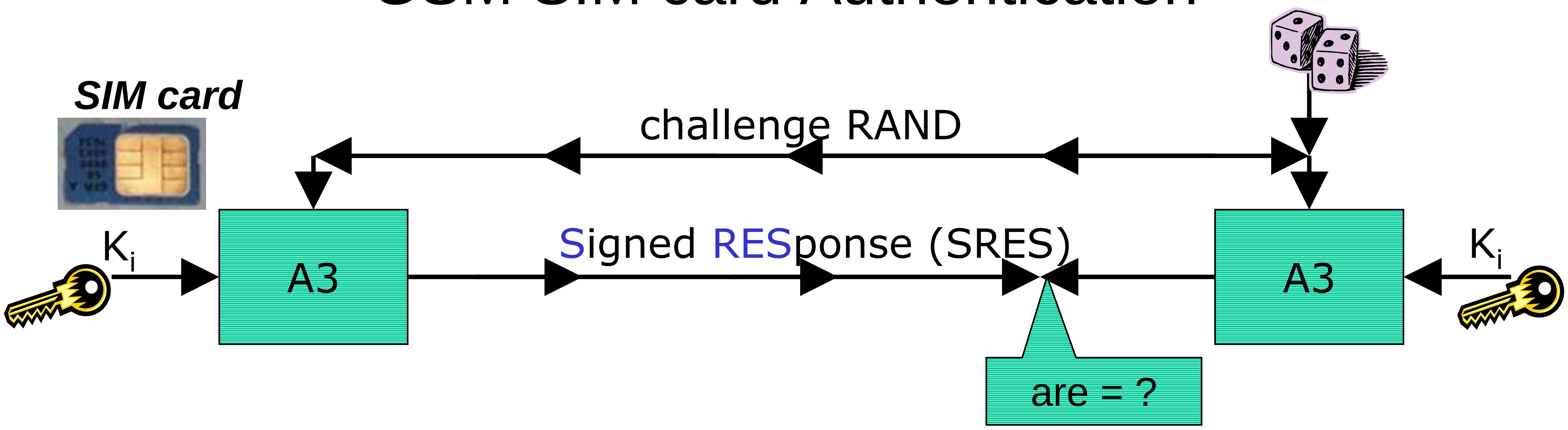
DFA requires

⇒ a DETERMINISTIC crypto process with a known output

(from which the attacker wants to extract the secret key)

Examples when this happens:

GSM SIM card Authentication



- RUN GSM ALGORITHM

Example: A0 88 00 00 10 XX …………….XX

CLA INS both 0 16 bytes random nonce

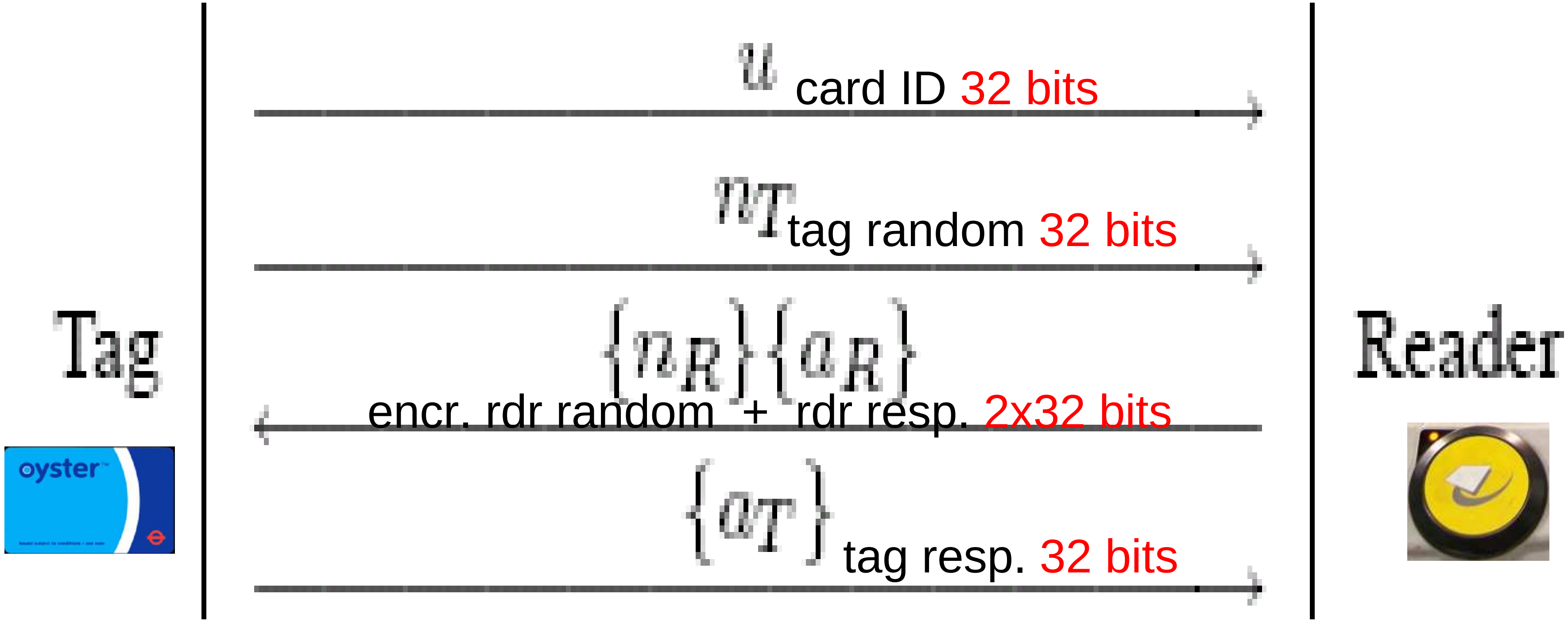
no L_e, no data in reply expected, result will be visible in the status bytes = 0x9F Le

In Contrast – 3G USIM Cards

No DFA attack, because

- the base station is authenticated first!
- the SQN should be checked for freshness.
  - so the card should never accept to do the same crypto computation twice

In Contrast – MiFare Classic



u card ID 32 bits

n_T tag random 32 bits

$\{n_R\}\{a_R\}$ encr. rdr random + rdr resp. 2x32 bits

$\{a_T\}$ tag resp. 32 bits

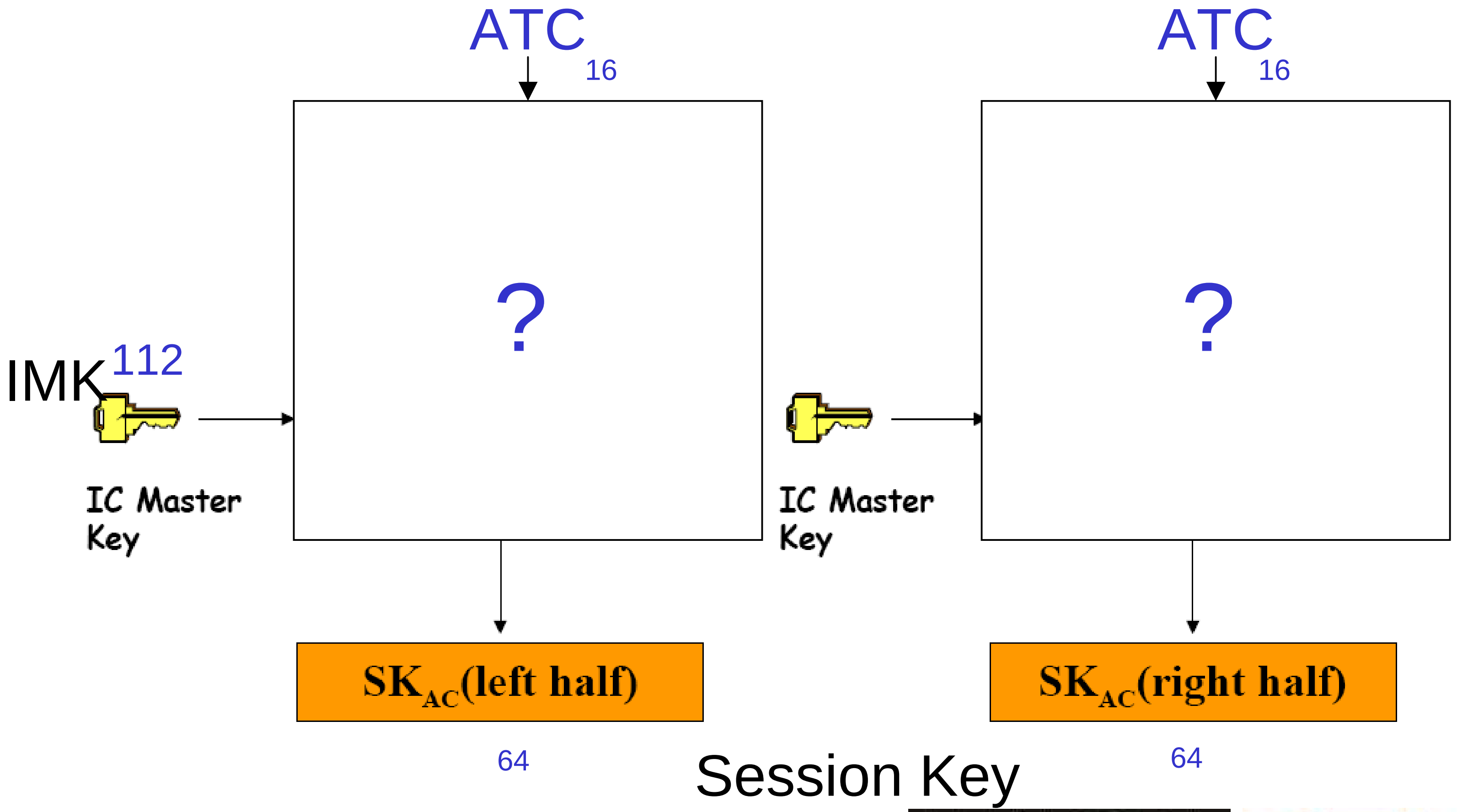
The reader is authenticated first !

No DFA attack unless card random repeats

☹ Fixed in “MiFare Plus” version (2009).

In Contrast – Bank Cards

Assuming ATC is always incremented => Session Key depends on ATC => Impossible to get the same cryptogram twice => DFA is impossible!



Conjecture:

Maybe DFA attacks are feasible in practice

only when

the industry uses

BAD PROTOCOLS ?

which can be fixed…

Fault Attacks

DFA Attacks…

Part 1: Provoke Faults of “suitable type” (Fault Model)

Part 2: Exploit Faults = cryptanalysis techniques

Cost:

Part 1: what is hard and costly is the lab work.

Part 2: frequently the cryptanalysis part is relatively easy (e.g. using simple differentials), or costs little, or requires small computing power.

One can expect that this is going to change in the future.

Who Wins?

Attackers or Defenders?

Security Evaluation at RFI Global

- Penetration testing of embedded systems.
  - software attacks
  - reverse-engineering
  - side-channel attacks
  - physical attacks
  - etc..
- Routinely done as a part of formal evaluation schemes such as EMVCo, PCI/PED, and other.
- RFI Global has extracted numerous symmetric and asymmetric keys…
  - and results are not published
    - due to strict non-disclosure contracts.

As a Proof Of Concept

Results presented today are obtained on a readily available
smart card XXXXXX
based on a modern micro-controller.

We don’t claim that this card is the strongest on the market.

Proof of concept work:
showing that DFA on DES is possible on some smart cards.

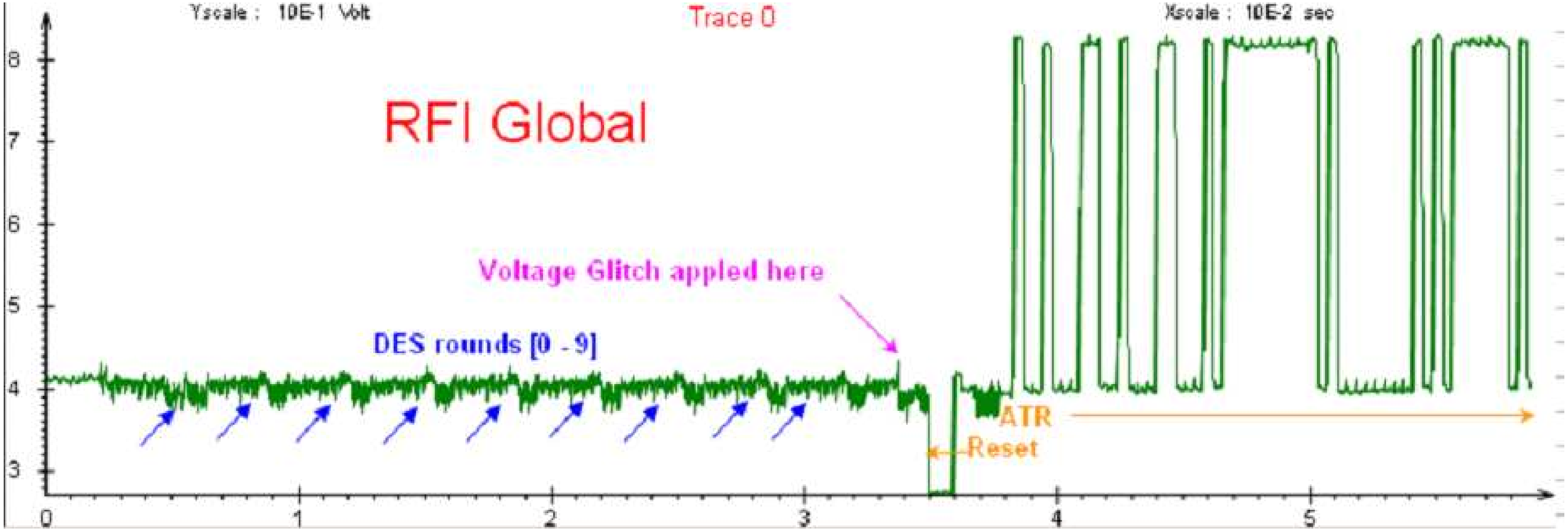
Key Question:

We don’t want to try ALL possible ways to achieve a glitch.

- can learn, discover a way to get there through stages?

Lab Work (1)

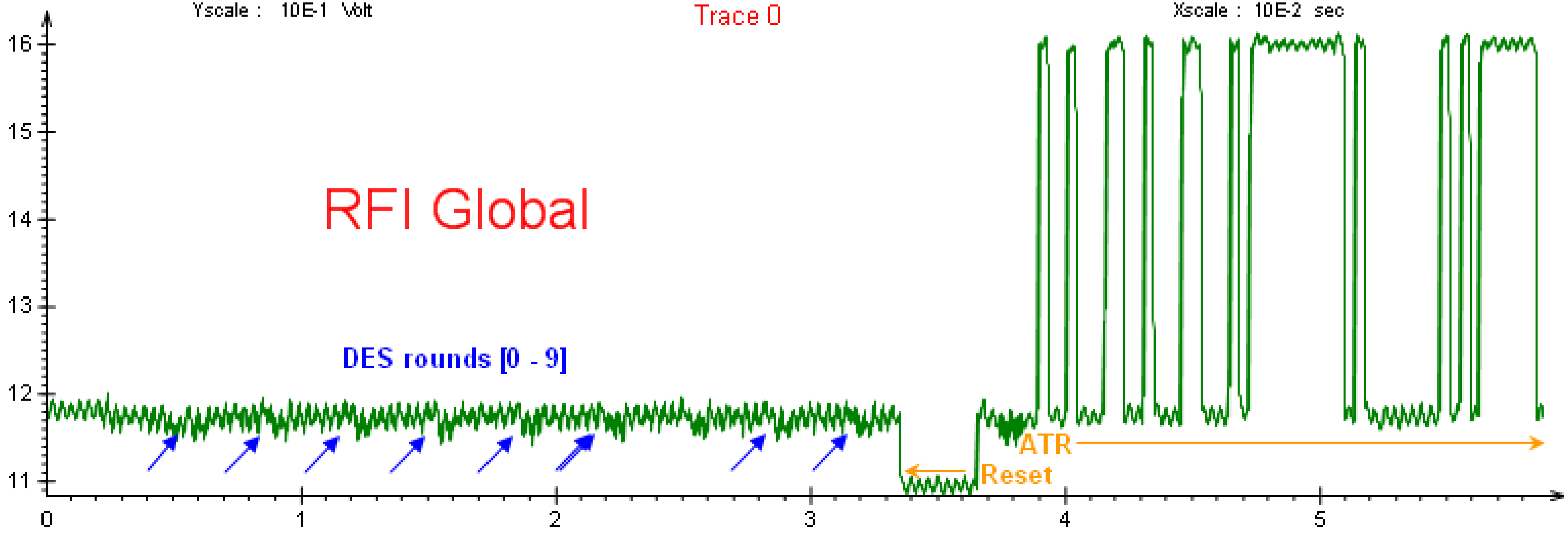
- Voltage glitch applied close to the final round.
- Triggers ATR - defensive behaviour, attack detected.



Lab Work (2)

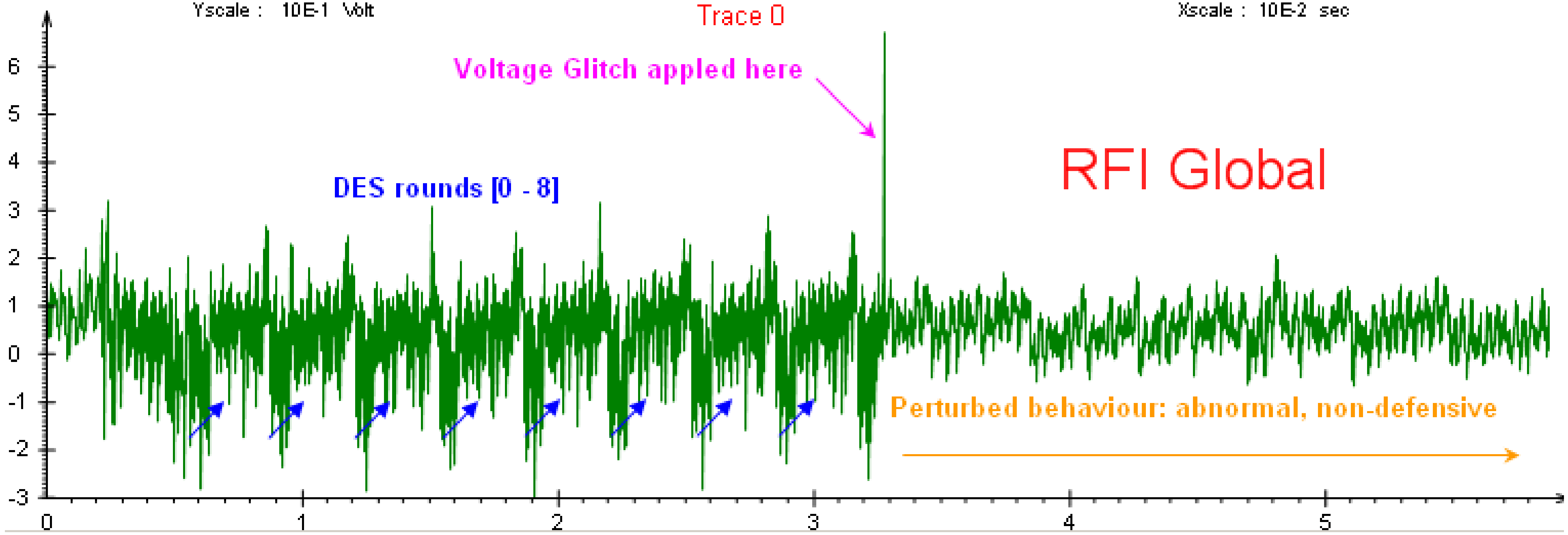
VERY STRANGE.

Now even without any glitch, the card will reset, the defensive behaviour pervades for many hours(!).



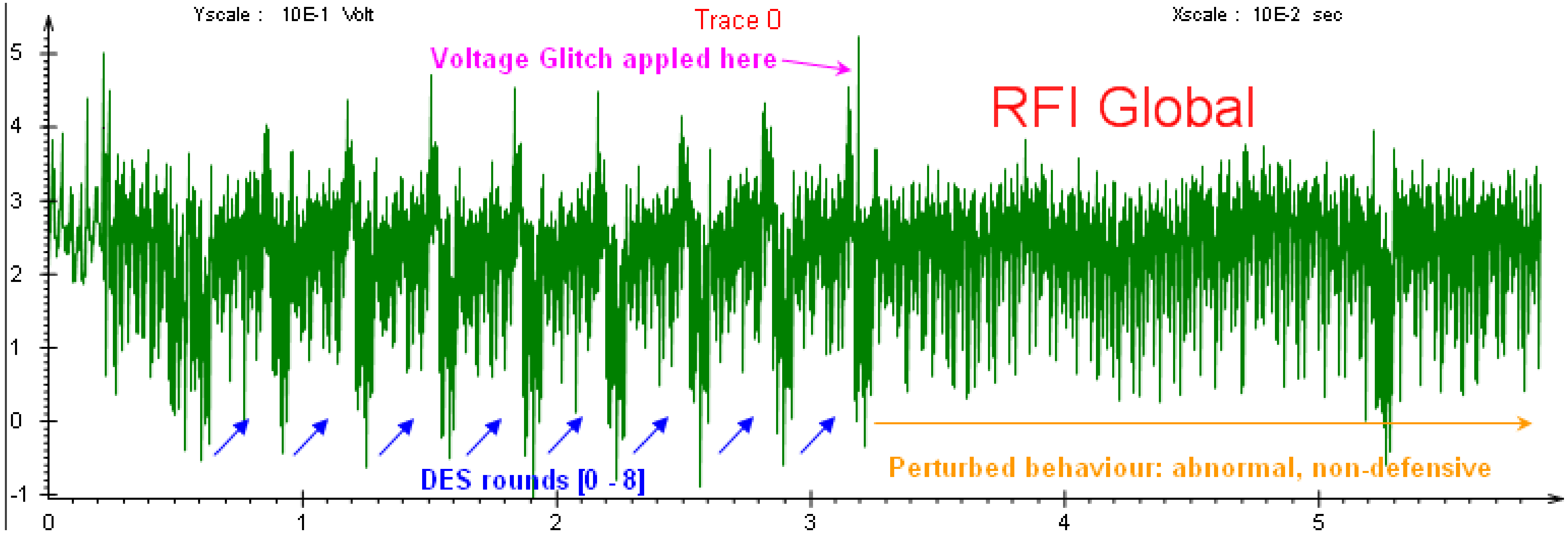
Lab Work (3)

We have discovered ways of doing a glitch which produces an abnormal behaviour, BUT not defensive, no reset, the card does not detect anything abnormal.



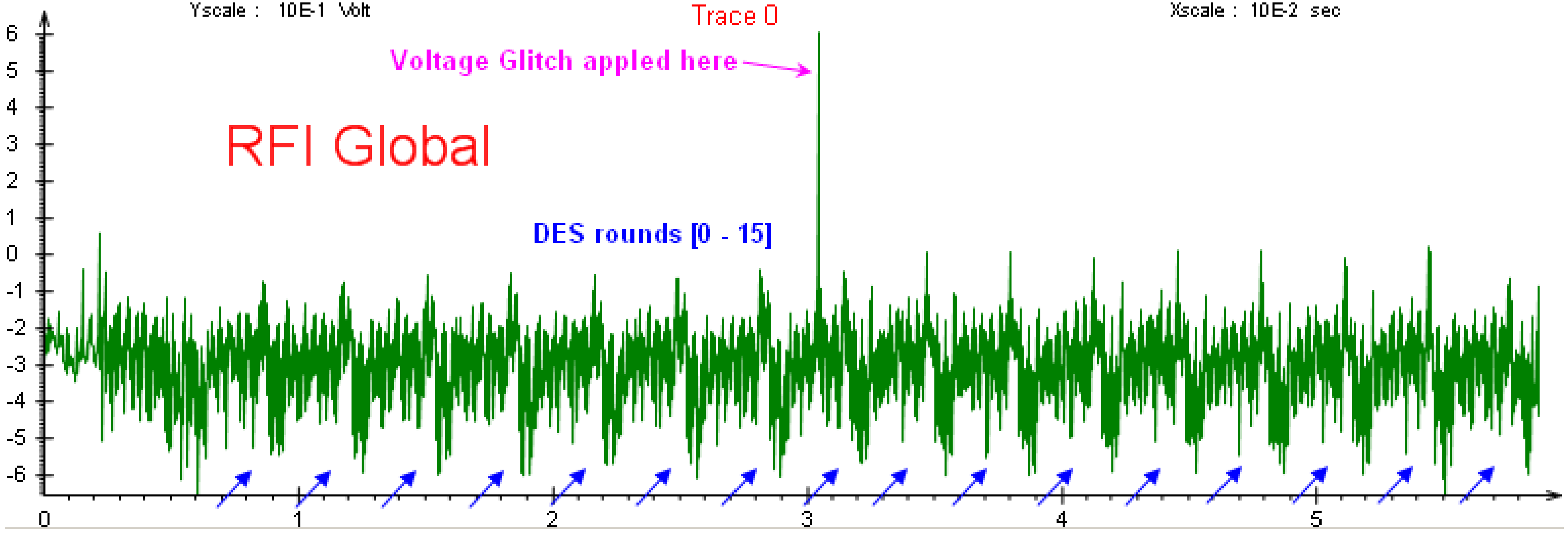
Lab Work (4)

Further investigations allow to do reproduce this abnormal but not-defensive behaviour in a predictable way.

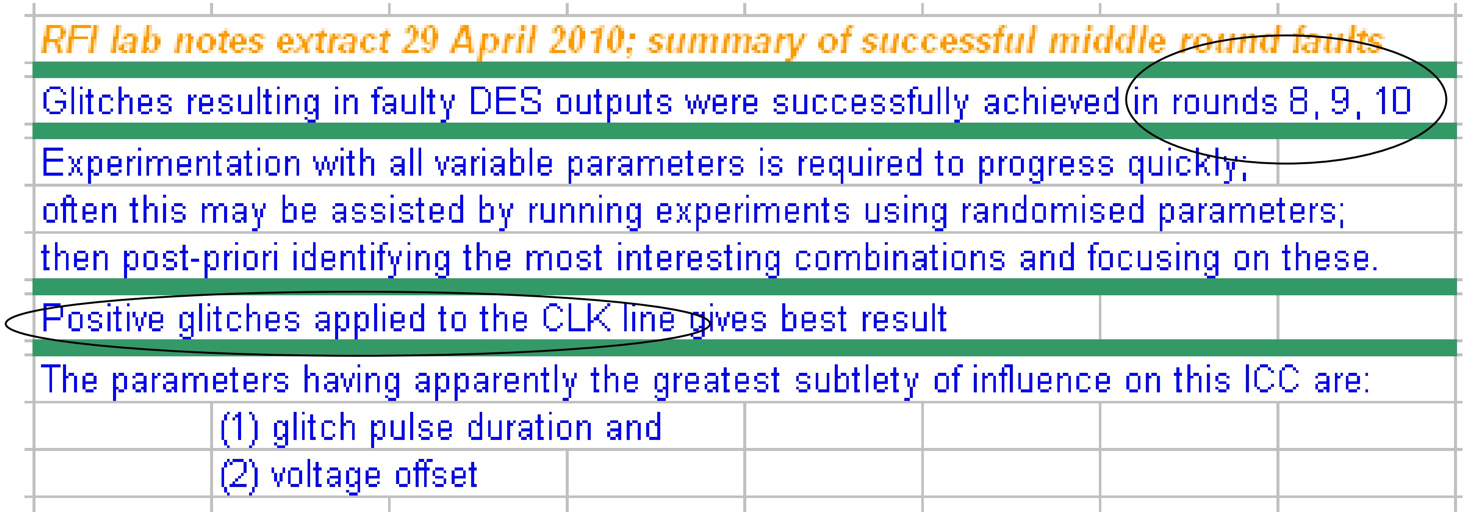


Lab Work (5)

In this way with further refinements and experimentation it was possible to discover parameters inducing a glitch in a middle round, where the card carries on the DES computation normally (giving a faulty ciphertext).



Lab Work (6)



RFI lab notes extract 29 April 2010; summary of successful middle round faults

Glitches resulting in faulty DES outputs were successfully achieved in rounds 8, 9, 10

Experimentation with all variable parameters is required to progress quickly;
often this may be assisted by running experiments using randomised parameters;
then post-priori identifying the most interesting combinations and focusing on these.

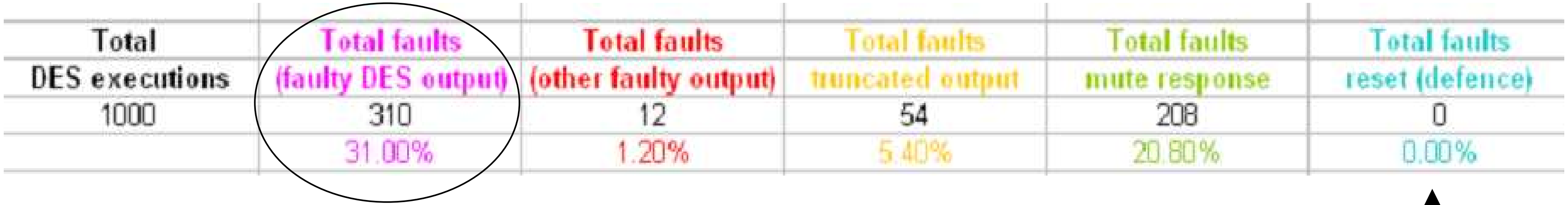
Positive glitches applied to the CLK line gives best result

The parameters having apparently the greatest subtlety of influence on this ICC are:

(1) glitch pulse duration and
(2) voltage offset

How Good It Can Get?

1000 runs:

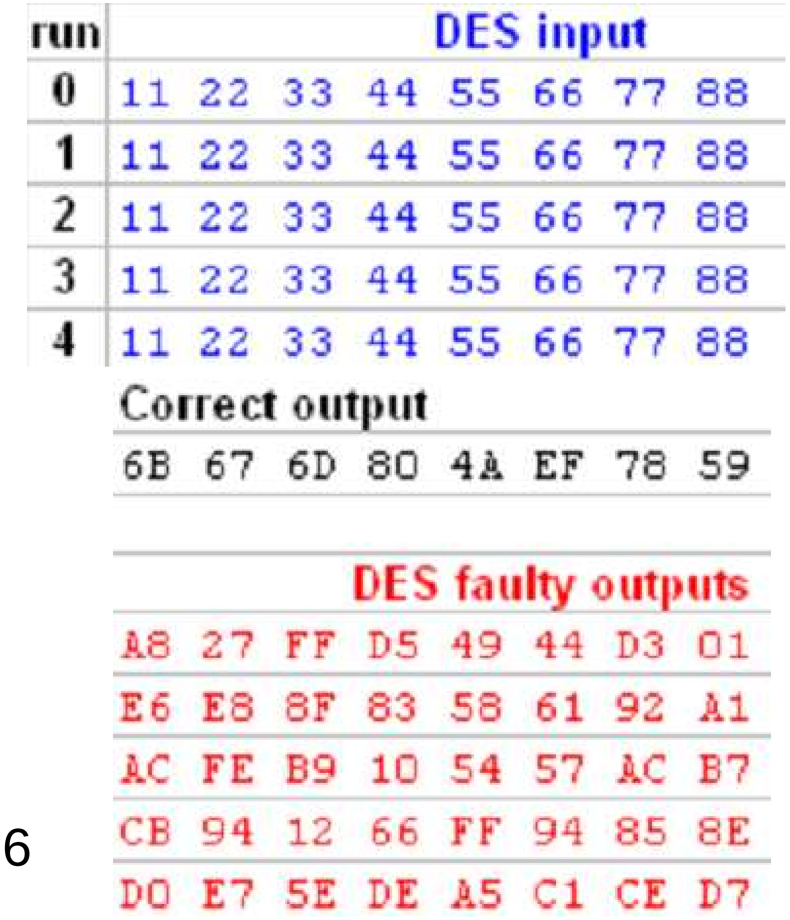


Total DES executions	Total faults (faulty DES output)	Total faults (other faulty output)	Total faults truncated output	Total faults mute response	Total faults reset (defence)
1000	310	12	54	208	0
	31.00%	1.20%	5.40%	20.80%	0.00%

never detected

Consecutive Faults?

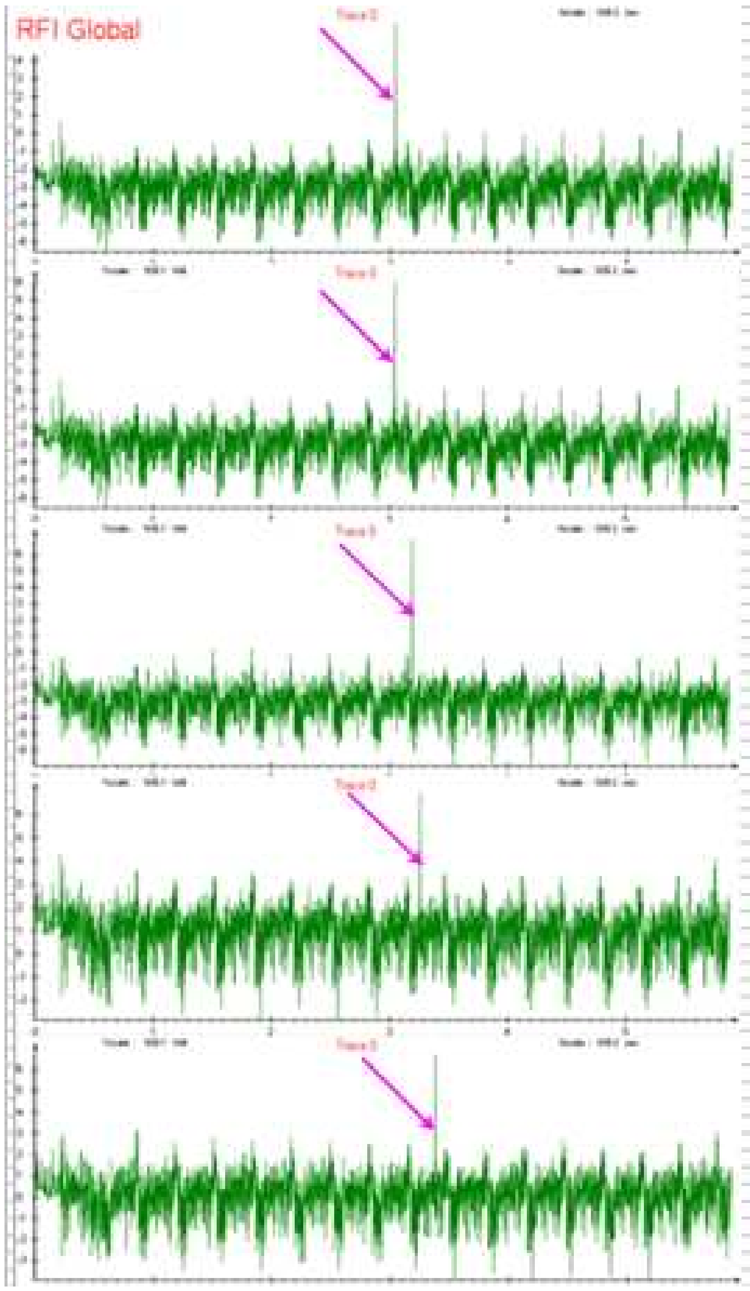
Done 5 consecutive faults
with precise timing
and consistent perturbation type:



run	DES input
0	11 22 33 44 55 66 77 88
1	11 22 33 44 55 66 77 88
2	11 22 33 44 55 66 77 88
3	11 22 33 44 55 66 77 88
4	11 22 33 44 55 66 77 88

Correct output
6B 67 6D 80 4A EF 78 59

DES faulty outputs
A8 27 FF D5 49 44 D3 01
E6 E8 8F 83 58 61 92 A1
AC FE B9 10 54 57 AC B7
CB 94 12 66 FF 94 85 8E
D0 E7 5E DE A5 C1 CE D7



Lab Work – Any Questions?

mailto: david.ware@rfi-global.com

Part 2.

Cryptanalysis!

Plausible Future of Fault Attacks

Don’t think that Lab work will become easier, on the contrary. But precisely because it will become VERY DIFFICULT to obtain faults,

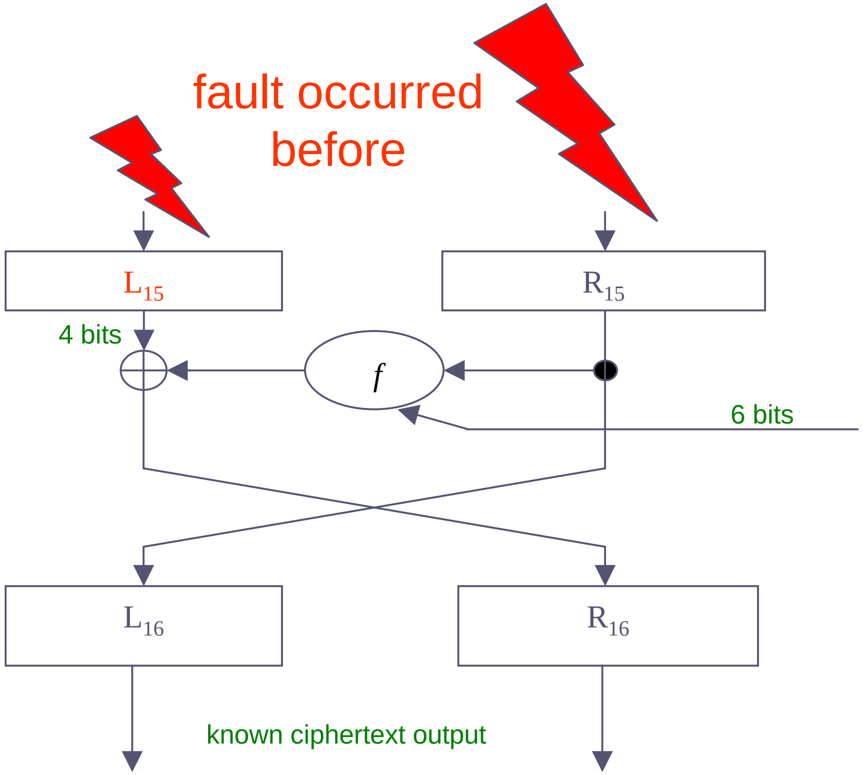
=> researchers now have to deploy more efforts on Part 2, recover the key using

- less faults AND
- faults buried deeper inside the algorithm
  - some critical parts such as first/last round benefit from increased protection...
- AND much more cryptanalysis
  - Part 2. will have to become much more sophisticated.
    - example: in the present work, there is a need to develop a lot of dedicated software…

Related Work

At CHES 2009, Matthieu Rivain (greatly improving on Akkar'04) explores fault attacks on DES middle rounds with statistical treatment:

- Decrypt back 1 round
  - look at 4 bits at time (one S-box)
- Check the distribution of 4 bits in $L_{15} \oplus L'_{15}$:
  - Q: is it "more" biased or "more" close to uniform?
    - Square Euclidian Imbalance



Rivain'09 results are extremely good, 10-20 faults at L_{12} allow to recover the full key.

More CHES 2009

Examples: they need 10..20000 faults at rounds 12..10.

Remark: These figures assumed that it is possible to 'reliably enough' perturbate just one half of DES state. Future countermeasures might make it difficult.

More CHES 2009

They conclude that

IF one can do 20 000 faults

THEN the last 7 rounds must be protected now.

A big mistake, let us correct it:

IF one can do 1 fault,

THEN the last 16 rounds must be protected.

What? Let us explain.

More CHES 2009

They conclude that

IF one can do 20 000 faults

THEN the last 7 rounds must be protected now.

A big mistake, let us correct it:

IF one can do 1 fault,

THEN the last 16 rounds must be protected.

What? Let us explain.

Background

Cryptanalysis has two main branches:

- Statistical
  - Requires a lot of data, processing can be fast
- Algebraic / Logical
  - Requires less data, but very hard computational problems arise (NP-hard problems).
  - Brute force is sometimes a FEASIBLE substitute, and the reference point to improve on.

Algebraic Cryptanalysis [Shannon]

Breaking a « good » cipher should require:

"as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type"

[Shannon, 1949]

Results on DES

Nicolas T. Courtois and Gregory V. Bard:

Algebraic Cryptanalysis of the D.E.S.

In IMA conference 2007, pp. 152-169, LNCS 4887, Springer.

See also:

eprint.iacr.org/2006/402/

What Can Be Done ?

Method: ANF-to-CNF + MiniSat algorithm:

Key recovery for 6 rounds of DES.

Only 1 KP (!).

No other attack works with 1 KP.

Method are slightly improved since 2007.

Now we can break 7 rounds with 1 KP…

How to Do It?

Key recovery for 6-7 rounds. Only 1 KP.

How to Do It?

- Step 1. Represent DES as a system of equations.
- Step 2. Convert to a SAT problems.
- Step 3. Solve by a SAT solver.

All the steps are non-trivial. Previous research gives some very good methods to achieve it.

Attack: Step 1

One idea:

- each DES S-box is written as a system of cubic equations.
- Many other methods also work…

  Like adding 40 extra variables, much simpler equations (“/opns method”).
- And many other, see IMA 2007 paper…
  - Free software to generate suitable encodings of DES can be obtained from Nicolas Courtois or downloaded from www.cryptosystem.net/aes/toyciphers.html

Attack: Step 2

Convert an algebraic system to a SAT problem.

- Ready software to do conversion can be downloaded from www.cryptosystem.net/aes/tools.html
- Best method here: “local interpolation”.
- At the end of the day, DES S-boxes are represented by many clauses of length 10.

Attack: Step 3

Solve the SAT problem.

3.1. MiniSat 2.0.

Winner of SAT-Race 2006 competition.

An open-source SAT solver package,
by Niklas Eén, Niklas Sörensson,

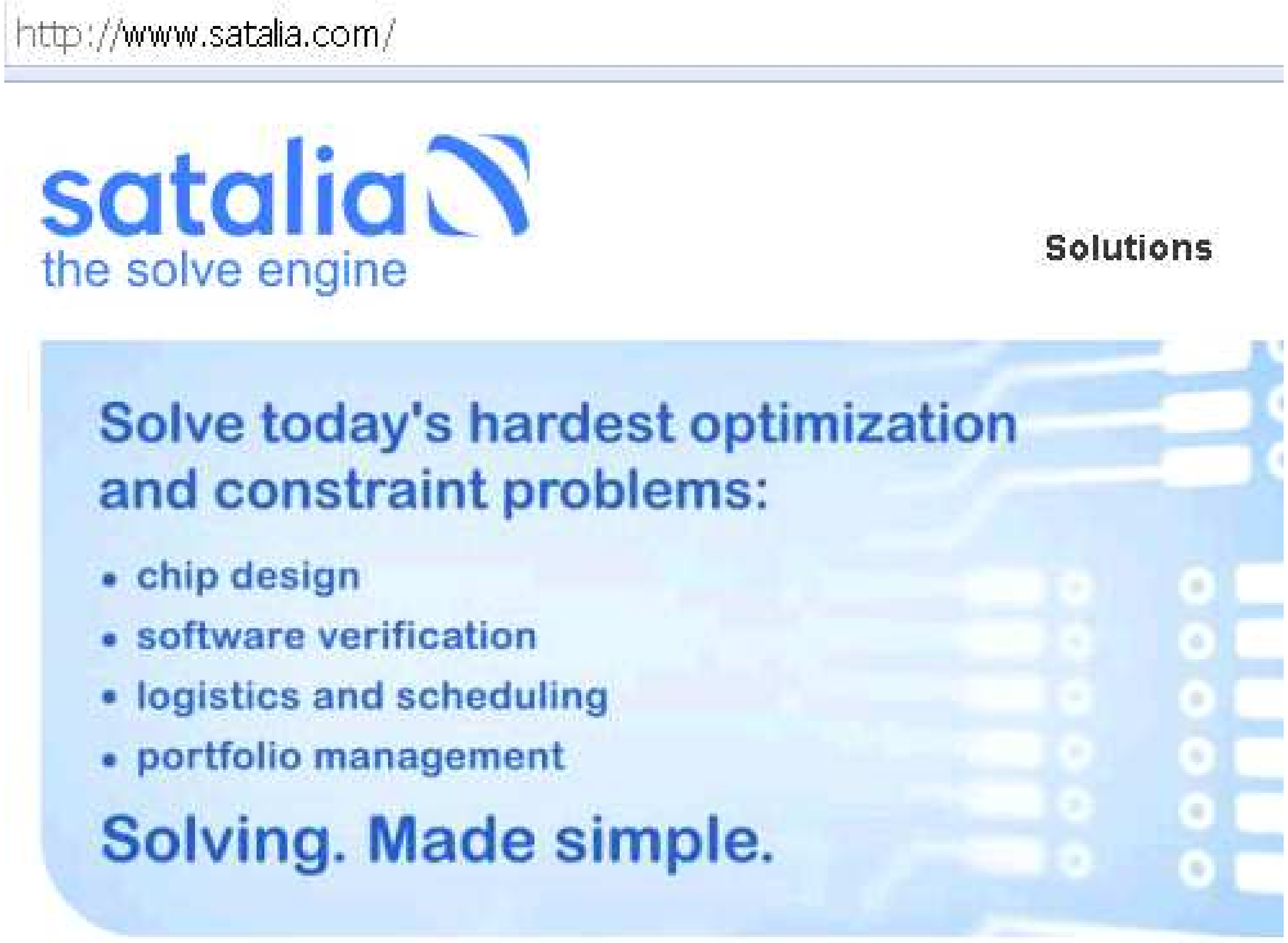
http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat/

3.2. Ready Software for Windows

Several ready programs to solve SAT problems are also available on the same web page:

www.cryptosystem.net/aes/tools.html

3.3. SAT Solvers in the Cloud



Solve today's hardest optimization and constraint problems:

- chip design
- software verification
- logistics and scheduling
- portfolio management

Solving. Made simple.

solving SAT problems on demand…

commercial but also for free…

Why These Methods

These methods are suitable for handling sophisticated fault attacks because:

1. These are the ONLY cryptanalytic attacks known which work when the attack disposes of only 1 or a few encrypted texts
2. They only work for a limited number of rounds, but in fault attacks precisely the complexity is frequently less than the whole cipher, because only a part of internal state is modified.

Our Simulations

we present some preliminary results.

Fault Model

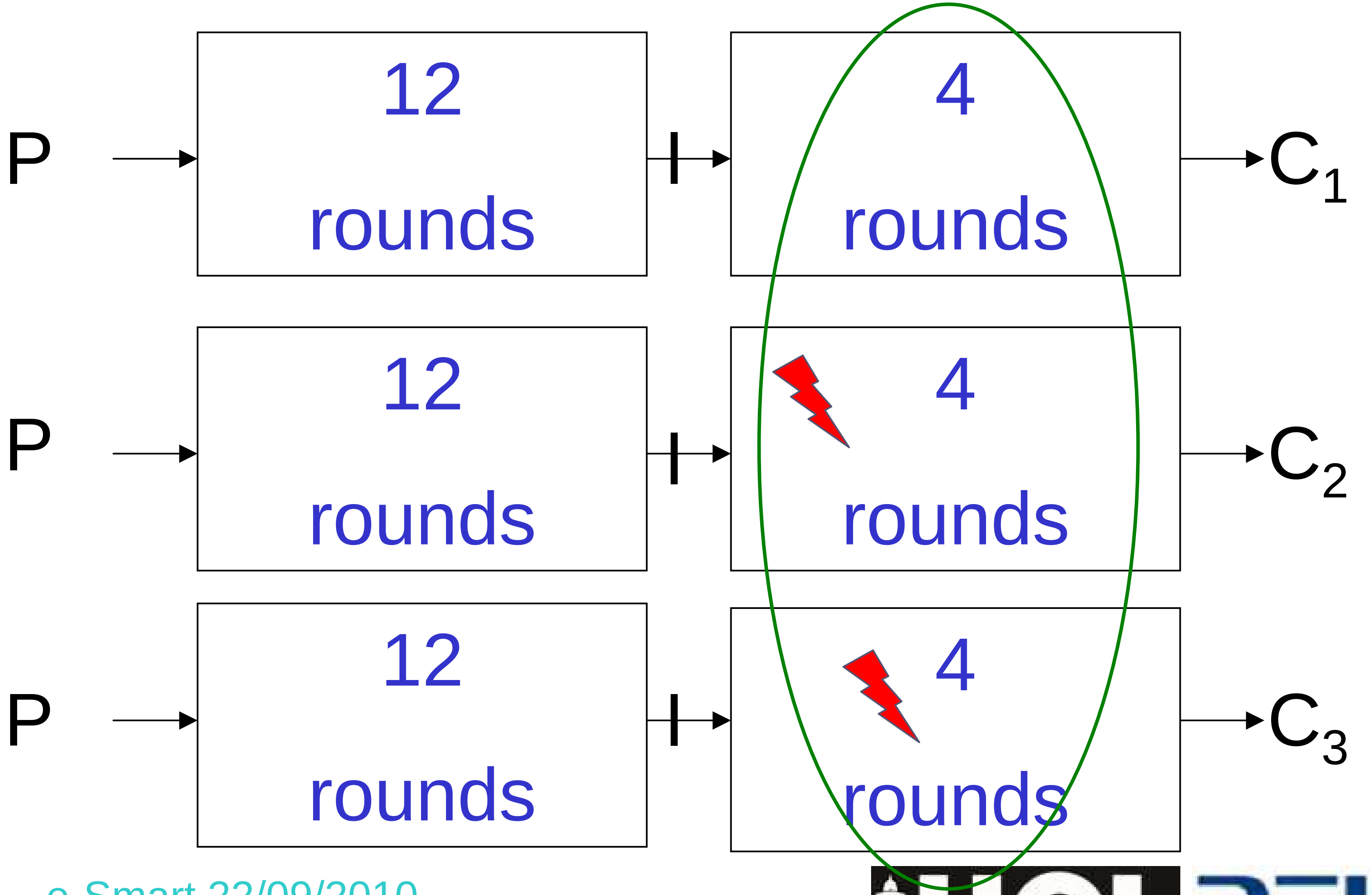
At present moment we assume that the fault is known for example one bit or one byte flipped.

The attacker needs to either

- guess which are flipped
- or control very well his perturbation.

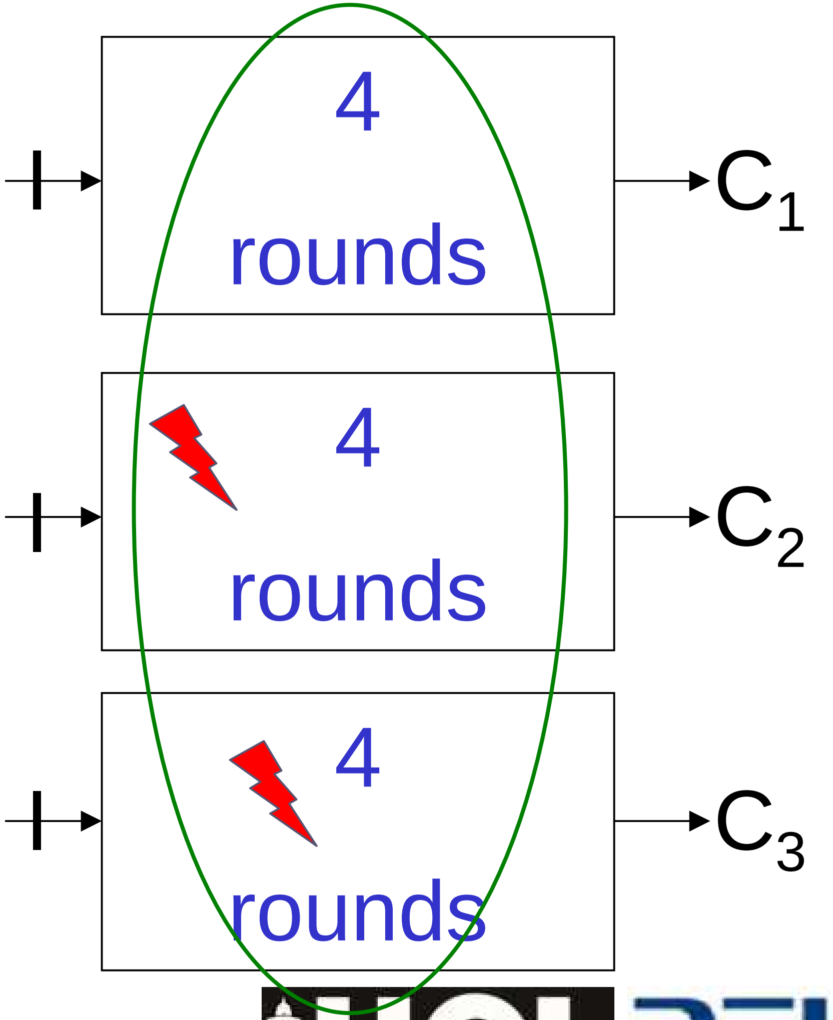
Basic situation

Example of a fault attack – solve the whole:



Ignore the Left

solve OR check for consistency :



Example

(not doing anything very exciting yet)

Flipping 1 bit affecting S-box S1 in round 16.

Fact: with about 2-3 faulty ciphertexts we get enough information to recover 6 bits of the key.

Much more faults are needed to recover the whole DES key.

Yet hope is not lost.

Remark: Due to diffusion if DES,
if we flip 1 bit in round say 13,
we hope to affect ALL S-boxes in round 16.

Diffusion => Less Faults but Harder

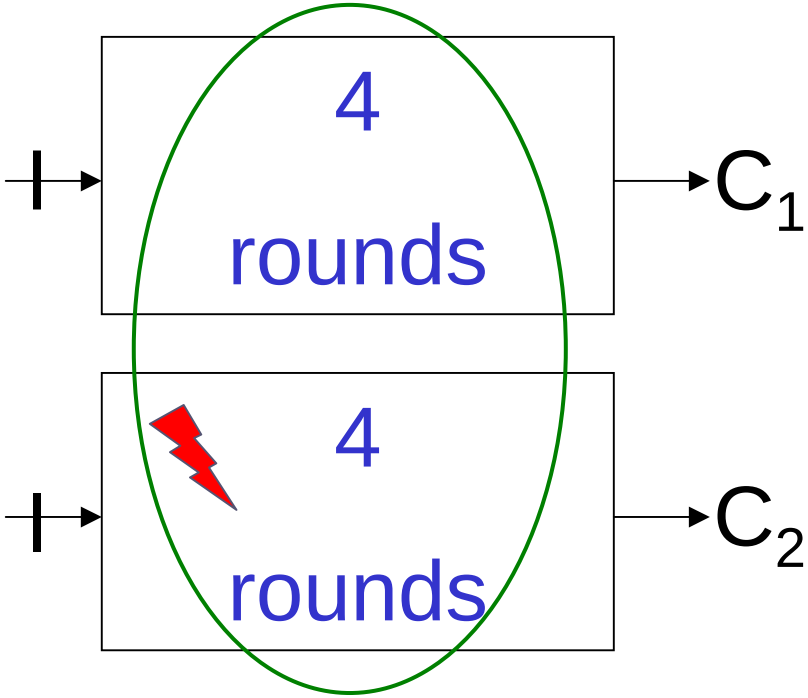
If we flip 1 bit in round say 13,
we hope to affect ALL S-boxes in round 16.

So maybe the whole key recovery is possible with 1 single faulty ciphertext? This is our “Holy Grail”: 1 single fault.

Claim: Even if the attacker can do just one fault, he should NOT do it in round 16, even if he could.
He should maybe aim at round 13.

We don’t quite get there, we use 2,4,8 faults for example.

The Hardest Case



consistent?
solve for key bits?

Wanted

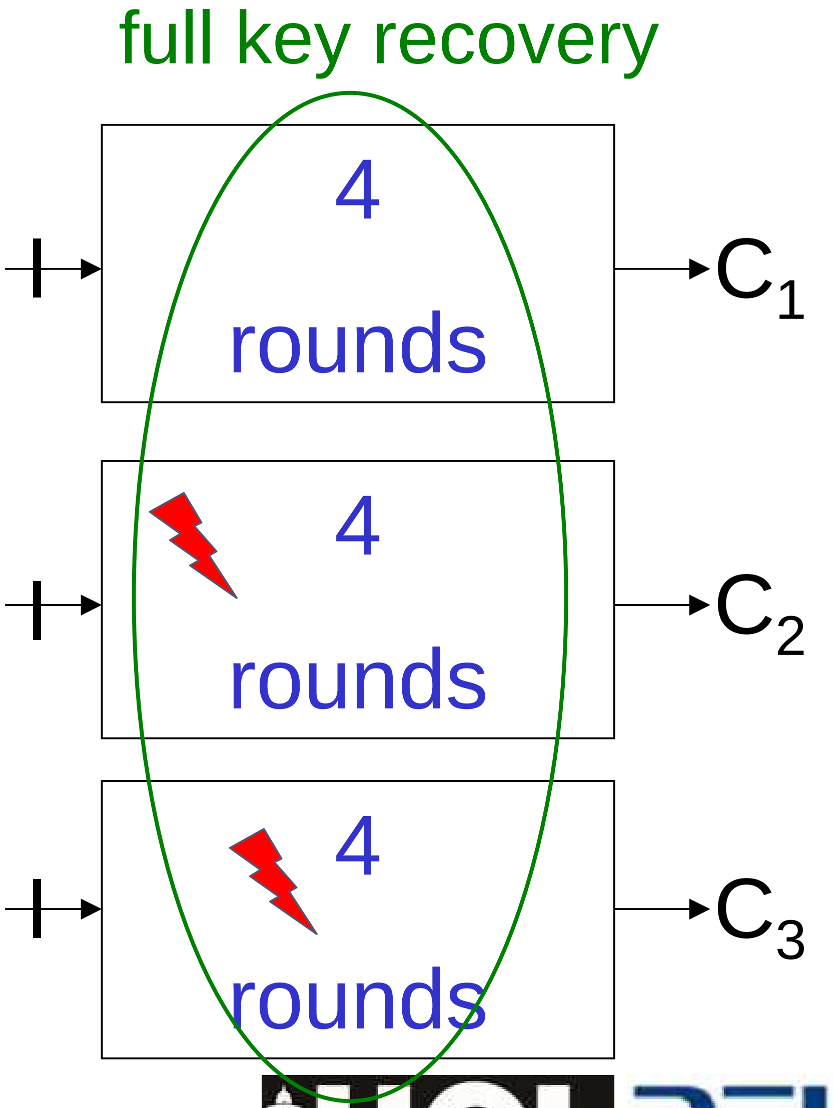
Efficient automated solvers and/or consistency tests for a few rounds of DES. Able to handle the whole complex situation.

In our research we are interested in attacks slightly faster than brute force.

- Our attacks involve guessing some key bits, than either we determine other key bits or find a contradiction.
  - Rule of thumb: if we guess 20 key bits, any attack faster than 2 hours, will break DES faster than by brute force.
  - For 3DES attacks in like 2^{60} are also possible, will still break 3DES in practice.

For now, our best attacks are full key recovery attacks via solving a SAT problem.

Current Attacks



Similar Work at UCL Belgium

- “using a single leakage trace”,
- AES and PRESENT

1. “Combining Algebraic and Side-Channel Cryptanalysis against Block Ciphers”
   - preprint by Mathieu Renauld, François-Xavier Standaert
2. “Algebraic Side-Channel Attacks on the AES: Why Time also Matters in DPA”
   - preprint by Mathieu Renauld, François-Xavier Standaert and Nicolas Veyrat-Charvillon

Our Results

Example 1:

We break DES with 2 faulty ciphertexts with 2 bits flipped, at round 14.

Guessing 20 key bits. Time to recover the key = 0.01 h.
Overall about 200 times faster than brute force.

However in order to use only 1 fault at round 14 unless we need to produce a precise fault on 10 bits, much harder to produce.

Surprise: Yet as explained before, due to more diffusion and more key bits being involved, 1 faulty ciphertext works very well at round 13.

Example 2:

We Break DES with 1 faulty ciphertext with 2 bits flipped, at round 13.

Guessing 24 key bits. Time to recover the key = 0.01 h.
About 10 times faster than brute force.

Remark: the only input of our program is a pair of ciphertexts, the plaintext is not used at all.

Note: an exe file to re-run the simulations presented here can be obtained from Nicolas Courtois.

More by Brute Force!

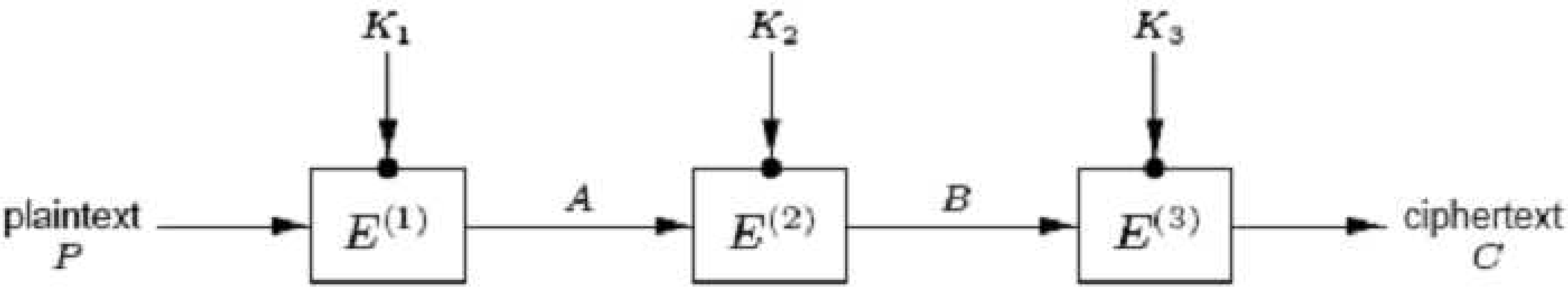
While these attacks can be improved,
this one is already feasible today:

If the DFA attackers does 2^{56} DES computations,
which is feasible for anybody with FPGA implementation
and a budget of say 20 K USD, then a fault attack
anywhere inside last 16 rounds is feasible.

Just one fault is needed.

Due to the fact that the last DES uses a 56-bit key.

(b) triple encryption ($K_1 = K_3$ for two-key variant)



Conclusion

At CHES 2009 Rivain shows that:

IF one can do 20 000 faults

THEN the last 7 rounds must be protected now.

But in the real life, with standard countermeasures, the attacker can only hope to induce a few faults per card…

Conclusion

Few faults per card? This will be enough.
He just needs more computers/FPGA,
and key recovery will be feasible with:

- algebraic attacks based on experimentation with sophisticated “solver” software
- or even simply brute force

With just 1 single fault in any of the 16 last rounds one can recover 3DES keys from smart cards in practice.