TU Graz

SCIENCE
PASSION
TECHNOLOGY

T2 Symmetric Analysis

Maria Eichlseder Markus Schofnegger

Applied Cryptography 2 – ST 2020

Assignments (~~48~~ 32 points)

Assignment 1: Asymmetric Cryptanalysis and Multiparty Computation

- Release: 19 Mar 2020 (= team registration deadline!)
- Question time: 23 Apr 2020
- Submission: 30 Apr 2020

Assignment 2: Symmetric Cryptanalysis

- Release: 7 May 2020
- Question time: 4 Jun 2020
- Submission: 12 Jun 2020

Assignment 2: Symmetric Cryptanalysis

A Related-Key Differential Analysis (`AES`)

- easy (4 points)

B Linear Cryptanalysis (`PRESENT`)

- medium (8 points)

C Cube Attack (`Keccak`)

- pro (12 points)

A Related-Key Differential Analysis (AES)

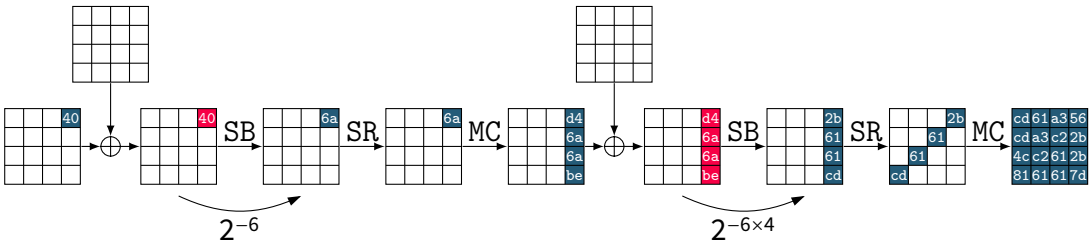
Related-Key Differential Analysis (AES) — 4 Points

Analyze differential characteristics of AES (→ L7):

a Experimentally evaluate 2-round single-key differentials

b Bound the number of active S-boxes under related keys using MILP

A Related-Key Differential Analysis (AES) – Cheatsheet



B Linear Cryptanalysis (PRESENT)

Linear Cryptanalysis (PRESENT) — 8 Points

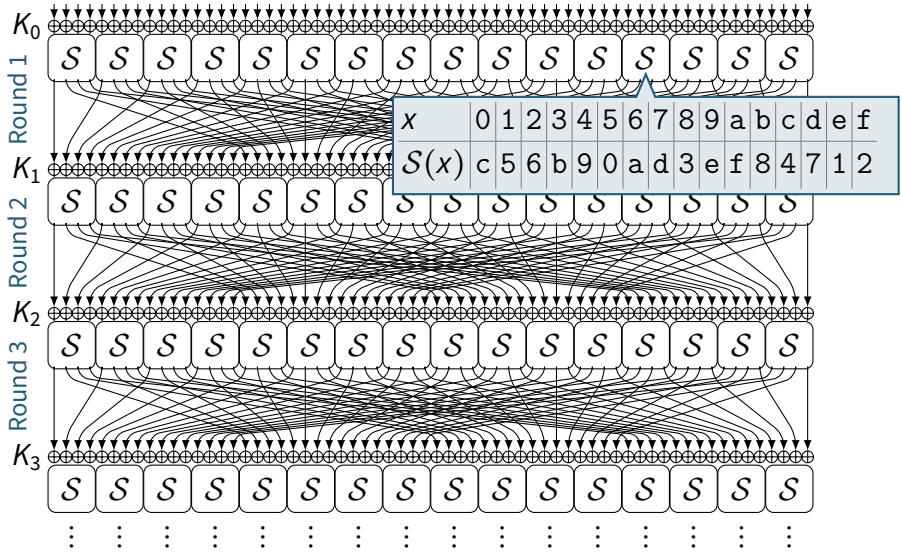
Apply linear cryptanalysis to find the PRESENT key (→ L6):

a Compute the LAT and find a good linear approximation for 9 rounds

b Estimate the bias of the linear approximation and verify it experimentally

c Define and implement a key-recovery attack for 10-round PRESENT

B Linear Cryptanalysis (PRESENT) – Cheatsheet



x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$\mathcal{S}(x)$	c	5	6	b	9	0	a	d	3	e	f	8	4	7	1	2

C Cube Attack (KECCAK)

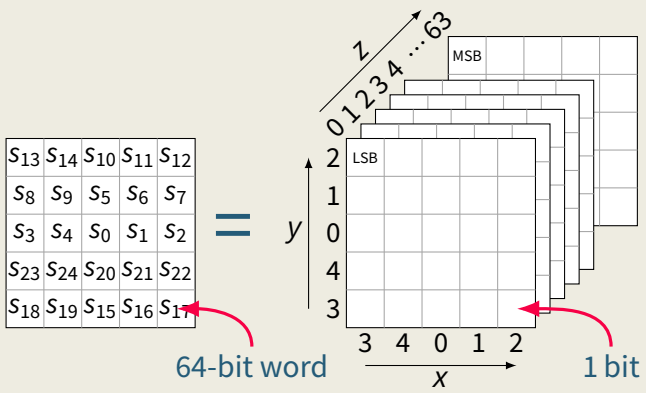
Cube Attack (KECCAK) — 12 Points

Implement the cube attack to find the KECCAK-MAC key (→ L8):

- **a** Implement the cube-sum function for KECCAK-MAC
- **b** Implement the offline phase (find suitable cubes)
- **c** Implement the online phase (equation-solving)
- **d** Demonstrate the cube attack for 4-round KECCAK-MAC

C Cube Attack (KECCAK) – Cheatsheet

State: $5 \times 5 \times 64 = 1600$ bits



Operations

Register-oriented, but hardware-friendly:

- $\oplus$ `xor`
- $\odot$ `and`
- $\lll_b$ `rotl` by b bits

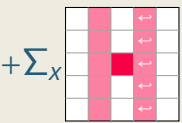
Steps in each Round

$\theta \to \rho \to \pi \to \chi \to \iota$

$$S = s_0 \| s_1 \| \ldots \| s_{24}, \qquad s_0 = x_{63} \cdots x_0, \qquad \ldots \qquad s_{24} = x_{1599} \cdots x_{1536}$$

C Cube Attack (KECCAK) – Cheatsheet

1 θ – Add neighbour column sums



$[x,y] \mathrel{+}= \Sigma_x$

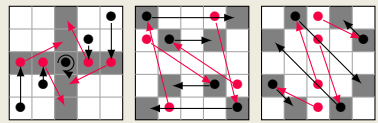
$\Sigma_x = \sum [x-1,\cdot] + \sum [x+1,\cdot] \lll 1$

2 ρ – Rotate words by offset ρ_{xy}

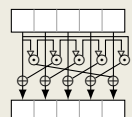
25	39	3	10	43
55	20	36	44	6
28	27	0	1	62
56	14	18	2	61
21	8	41	45	15

$[x,y] \lll\!= \rho_{xy}$

3 π – Permute words



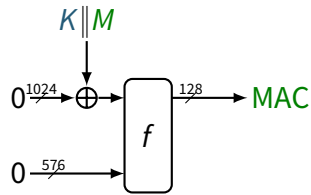
4 χ – Apply 5-bit S-box to each row



$[\cdot,y] = \mathcal{S}([\cdot,y])$

5 ι – Add constant C_r to register s_0

C Cube Attack (KECCAK) – Cheatsheet



$K = s_0 \| s_1$ $M = s_2 \| s_3 \| \ldots \| s_{15}$ $\text{MAC} = s_0 \| s_1$

- 1-round cube for testing: cube variable $\{p_{128}\} \rightarrow$ equations

$$y_{45} = k_{66},$$
$$y_{85} = k_{106} + 1.$$

Rules

Coding:

- Pick your favourite programming language
- You'll need "tweakable" implementations of target ciphers
- Boolean linear equation solving recommended
- If it's not open-source, please ask us first!
- Suggestion: C/C++; for MILP: `sagemath`

Submission:

- Submit your team's code as a $\{\texttt{zip}, \texttt{tar.gz}\}$ archive in STicS
- Add a file `README`.$\{\texttt{md}, \texttt{txt}, \texttt{pdf}\}$ on the top level (design choices, limitations, howto, runtime).
- Include `Makefile` and/or clear instructions to run

C/C++ implementations of target ciphers

A AES: Intel's AES-NI instructions or
https://github.com/B-Con/crypto-algorithms

B PRESENT: Your own or
http://www.lightweightcrypto.org/implementations.php

C KECCAK: https://github.com/gvanas/KeccakCodePackage

Try the Reference or Standalone implementations:
`SnP/KeccakP-1600/Reference/KeccakP-1600-reference.c`
`Standalone/CompactFIPS202/Keccak-readable-and-compact.c`

Tips

- Start with minimal round numbers
  (where you can still understand by hand what's going on)
- Start with a partially known key
- Use pictures to understand what's going on
  (we're happy to share TEX/TikZ code for generating cipher pictures)
- Allow enough runtime :)

Remaining Schedule

14 May	L9: Hash Function Cryptanalysis	
21 May	Holiday	
28 May	L10: Advanced Differential Attacks	
04 June	S1: Password Hashing	Question time T2
	S2: Authenticated Encryption: Security Notions	
11 June	Holiday	(Friday) Deadline T2
18 June	S3: Post-Quantum Crypto	
	S4: Fully Homomorphic Encryption	
25 June	S5: Algebraic Attacks: Gröbner Bases etc.	
02 July	VO Exam	

Questions

If you’re unsure how to tackle some of the steps (linearity, …), ask anytime (lecture, newsgroup, email, your colleagues)!