

# Cube Attacks on Cryptographic Hash Functions

Joel Lathrop

Department of Computer Science  
Rochester Institute of Technology

May 21, 2009

# Outline

- 1 Background
- 2 Cube Attacks
  - Introduction
  - Theory
  - Example Precomputation
  - Example Online Attack
  - Complexity
- 3 Attack on Keccak
- 4 Attack on ESSENCE
- 5 Conclusion

# Outline

- 1 Background
- 2 Cube Attacks
  - Introduction
  - Theory
  - Example Precomputation
  - Example Online Attack
  - Complexity
- 3 Attack on Keccak
- 4 Attack on ESSENCE
- 5 Conclusion

# Hashing Theory

A **hash function** is defined as

$$h : \{0, 1\}^* \rightarrow \{0, 1\}^m$$

where  $m$  is the fixed length of the hash function  $h$  in bits.

Security Problems [MVOV97]:

- **Collision**: Given a hash function  $h$ , find two message values  $x$  and  $x'$  such that  $x \neq x'$  but  $h(x) = h(x')$ .
- **Second-Preimage**: Given a hash function  $h$  and a message value  $x$ , find another message value  $x'$  such that  $x \neq x'$  and  $h(x) = h(x')$ .
- **Preimage**: Given a hash function  $h$  and a hash value  $y$ , find message value  $x$  such that  $h(x) = y$ .
- **Partial Preimage**: Given a hash function  $h$  and a hash value  $y = h(x)$ , recover a substring of  $x$ .

# History

- In 2004 Wang et. al. find **collisions in MD5** [WY05].
- By June 2007, the attack runs in **6 seconds** on commodity hardware [Ste07].
- The attack is **extended to SHA-1**, but remains theoretical.
  - Current complexity of collision search is  $2^{52}$  hash operations [CMP09].
- In November 2007, NIST announces a competition for a new hashing standard to be named **SHA-3**.
- In September 2008, Itai Dinur and Adi Shamir introduce **cube attacks** [DS08].

# Outline

- 1 Background
- 2 Cube Attacks**
  - Introduction
  - Theory
  - Example Precomputation
  - Example Online Attack
  - Complexity
- 3 Attack on Keccak
- 4 Attack on ESSENCE
- 5 Conclusion

# Introduction

## What are Cube Attacks?

- A new kind of cryptanalytical attack.
- Used in a chosen-plaintext attack to find some secret.
- Complexity limited by the degree of the polynomial representing the attack function, *not* the number of terms.
- Described in a paper by Itai Dinur and Adi Shamir entitled “Cube Attacks on Tweakable Black Box Polynomials” [DS08].
  - “Polynomials”
  - “Black box”
  - “Tweakable”
  - “Cube attacks”

## Problem description

The function under attack  $F(x, v)$

- $x$  is **secret**
- $v$  is **public**
- Black box
- Simulatable

### Problem

Given an oracle  $G(v)$  which computes  $F(x, v)$ , find the value of  $x$ .

# Terminology

## Definition

Assume some polynomial  $p(x_1, \dots, x_n)$  and a set  $I \subseteq \{1, \dots, n\}$  of indices to the variables of  $p$ .

$$p(x_1, \dots, x_n) \equiv t_I \cdot p_{S(I)} + q(x_1, \dots, x_n)$$

$p_{S(I)}$  is the **superpoly** of  $I$  in  $p$ .

## Definition

A **maxterm** is a subterm  $t_I$  with a corresponding superpoly  $p_{S(I)}$  such that  $\deg(p_{S(I)}) \equiv 1$ , i.e. the superpoly of  $I$  in  $p$  is a linear polynomial which is not constant.

# Cubes

Summing over a cube . . .

$$p_I \triangleq \sum_{v \in C_I} p|_v$$

- $I =$  a  $k$ -size subset of variable indices.
- $C_I = \{0, 1\}^k$ . A  $k$ -dimensional Boolean **cube**.
- $p|_v \triangleq p$  with  $v$  assigned to the variables in  $I$ .

# Primary Observation

## Theorem

*For any polynomial  $p$  and subset of variables  $I$ ,  $p_I \equiv p_{S(I)} \pmod 2$ .*

Example:

$$\begin{aligned} p(v_1, v_2, x_1, x_2, x_3) &= v_1 v_2 x_1 + v_1 v_2 x_2 + v_2 x_2 x_3 + v_1 v_2 + v_2 + x_1 x_3 + x_3 + 1 \\ &= v_1 v_2 (x_1 + x_2 + 1) + (v_2 x_2 x_3 + x_1 x_3 + v_2 + x_3 + 1) \end{aligned}$$

$$I = \{1, 2\}$$

$$t_I = v_1 v_2$$

$$p_{S(I)} = x_1 + x_2 + 1$$

$$\begin{aligned} p_I &= 0 \cdot 0(x_1 + x_2 + 1) + (0 \cdot x_2 \cdot x_3 + x_1 \cdot x_3 + 0 + x_3 + 1) \\ &\quad + 0 \cdot 1(x_1 + x_2 + 1) + (1 \cdot x_2 \cdot x_3 + x_1 \cdot x_3 + 1 + x_3 + 1) \\ &\quad + 1 \cdot 0(x_1 + x_2 + 1) + (0 \cdot x_2 \cdot x_3 + x_1 \cdot x_3 + 0 + x_3 + 1) \\ &\quad + 1 \cdot 1(x_1 + x_2 + 1) + (1 \cdot x_2 \cdot x_3 + x_1 \cdot x_3 + 1 + x_3 + 1) \\ &= x_1 + x_2 + 1 \end{aligned}$$

# The Attack

The output bits of any function can be described as  $GF(2)$  polynomials on the functions input bits.

**Precomputation:** Find maxterms on the public input bits that expose superpolys on the secret inputs bits.

**Online Attack:**

- 1 Compute the values each exposed superpoly.
- 2 Solve the resulting system of linear equations to recover the secret input bits.

## The Function Under Attack

Consider a function  $p(v_0, v_1, v_2, x_0, x_1, x_2)$  where

- $v_0, v_1, v_2$  are the **public** inputs
- $x_0, x_1, x_2$  are the **secret** inputs.

For just this slide, let's **cheat**:

$$p(v_0, v_1, v_2, x_0, x_1, x_2) = v_0 v_1 x_0 + v_0 v_1 x_1 + v_2 x_0 x_2 \\ + v_1 x_2 + v_0 x_0 + v_0 v_1 + v_1 + x_0 x_2 + x_2 + 1$$

## Testing $l = \{2\}$

Try the subterm  $t_{l_0} = v_2$  where  $l_0 = \{2\}$ .

That means  $p_{l_0}(x_0, x_1, x_2) = p(0, 0, 0, x_0, x_1, x_2) + p(0, 0, 1, x_0, x_1, x_2)$ .

Is  $p_{l_0}$  constant?

$$p_{l_0}(0, 0, 0) = 0$$

$$p_{l_0}(1, 0, 1) = 1$$

Is  $p_{l_0}$  linear?

Given  $x = (1, 0, 1)$  and  $y = (0, 1, 1)$ , with  $x \oplus y = (1, 1, 0) \dots$

$$p_{l_0}(0) + p_{l_0}(x) + p_{l_0}(y) = p_{l_0}(x \oplus y)$$

$$p_{l_0}(0, 0, 0) + p_{l_0}(1, 0, 1) + p_{l_0}(0, 1, 1) = p_{l_0}(1, 1, 0)$$

$$0 + 1 + 0 = 0$$

## Testing $l = \{0, 2\}$

$p_{\{2\}}$  was nonlinear, so we'll try adding another variable to our subterm.  
 Try the subterm  $t_{l_1} = v_0 v_2$  where  $l_1 = \{0, 2\}$ .

That means

$$p_{l_1}(x_0, x_1, x_2) = p(0, 0, 0, x_0, x_1, x_2) + p(0, 0, 1, x_0, x_1, x_2) \\ + p(1, 0, 0, x_0, x_1, x_2) + p(1, 0, 1, x_0, x_1, x_2)$$

Is  $p_{l_1}$  constant?

$$p_{l_1}(0, 0, 0) = 0 \quad p_{l_1}(0, 0, 1) = 0 \\ p_{l_1}(0, 1, 0) = 0 \quad p_{l_1}(0, 1, 1) = 0 \\ p_{l_1}(1, 0, 0) = 0 \quad p_{l_1}(1, 0, 1) = 0 \\ p_{l_1}(1, 1, 0) = 0 \quad p_{l_1}(1, 1, 1) = 0$$

## Testing $l = \{0, 1\}$

Try the subterm  $t_{l_2} = v_0 v_1$  where  $l_2 = \{0, 1\}$ .

Is  $p_{l_2}$  constant?

$$p_{l_2}(0, 0, 0) = 1$$

$$p_{l_2}(1, 0, 1) = 0$$

Is  $p_{l_2}$  linear?

$$p_{l_2}(0) + p_{l_2}(x) + p_{l_2}(y) = p_{l_2}(x \oplus y)$$

$$p_{l_2}(0, 0, 0) + p_{l_2}(1, 0, 1) + p_{l_2}(0, 1, 1) = p_{l_2}(1, 1, 0)$$

$$1 + 0 + 0 = 1$$

## Determining $t_{l_2}$ 's superpoly

We must deduce the superpoly  $p_{S(l_2)}$  that corresponds to  $t_{l_2}$ .

Compute the free term:

$$p_{l_2}(0, 0, 0) = 1$$

Test for the presence of each secret variable:

$$p_{l_2}(1, 0, 0) = 0$$

$$p_{l_2}(0, 1, 0) = 0$$

$$p_{l_2}(0, 0, 1) = 1$$

So the superpoly for the maxterm  $t_{l_2} = v_0 v_1$  is ...

$$p_{S(l_2)} = 1 + x_0 + x_1$$

## Complete Precomputation

Two more maxterms are found by this method:

- $t_{l_3} = v_0$  with superpoly  $p_{S(l_3)} = x_0$
- $t_{l_4} = v_1$  with superpoly  $p_{S(l_4)} = 1 + x_2$

The precomputation is complete:

**Table:** Maxterms (given as cube indices) and superpolys.

Superpoly Polynomial $p_{S(l)}$	Cube indices $l$
$x_0$	$\{0\}$
$1 + x_0 + x_1$	$\{0, 1\}$
$1 + x_2$	$\{1\}$

## Online Attack

Presume some oracle  $g(v_0, v_1, v_2) = p(v_0, v_1, v_2, x_0, x_1, x_2)$  with hidden, secret values of  $x_0 = 1, x_1 = 0, x_2 = 1$ .

Find the values of each superpoly:

$$x_0 = r_{\{0\}} = g(0, 0, 0) + g(1, 0, 0) = 1$$

$$1 + x_0 + x_1 = r_{\{0,1\}} = g(0, 0, 0) + g(1, 0, 0) + g(0, 1, 0) + g(1, 1, 0) = 0$$

$$1 + x_2 = r_{\{1\}} = g(0, 0, 0) + g(0, 1, 0) = 0$$

Solving the system of linear equations gives ...

$$x_0 = 1$$

$$x_1 = 0$$

$$x_2 = 1$$

# Complexity

For a polynomial  $p$  of degree  $d$  with  $n$  secret input bits and  $m$  public input bits, ...

The complexity of computing the values of  $n$  superpolys is at most  $2^{d-1}n$  queries to the oracle.

The complexity of solving the system of linear equations is  $\mathcal{O}(n^2)$ <sup>1</sup>.

So the **total complexity** of the online attack is  
 $\mathcal{O}(2^{d-1}n) + \mathcal{O}(n^2)$ .

---

<sup>1</sup>Provided that the matrix of superpoly linear equations is nonsingular, which can be enforced by simply finding a few more superpolys if necessary.

# Outline

- 1 Background
- 2 Cube Attacks
  - Introduction
  - Theory
  - Example Precomputation
  - Example Online Attack
  - Complexity
- 3 **Attack on Keccak**
- 4 Attack on ESSENCE
- 5 Conclusion

# The Keccak Hash Function

- Keccak is a SHA-3 candidate submitted by Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche [BDPA09].
- Uses a **sponge construction** [BDPVA07], in which message blocks are “absorbed” and the hash is “squeezed” out.

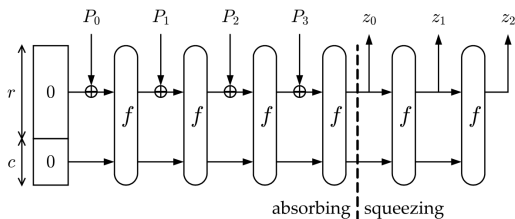


Figure: Keccak sponge construction [BDPA09].

## Attack Configuration

- We attack **224-bit** Keccak.
- Reduce to **4 rounds**. (Default is 18.)
- Partial pre-image attack
  - **112-bit secret** and **112-bit public** inputs
  - Find the secret  $s$  given some  $g(x) = \text{KECCAK}(s||x)$

## Attack Results

- Found 112 maxterms of degree  $\leq 12$
- Precomputation took about 10 minutes.
- Complexity of on-line attack  $< 2^{19}$  hashing operations.

Superpoly Polynomial	Cube Indices	Output Bit
$1 + x_{79}$	{17,30,35,37,41,45,62,70,76,80,105}	123
$x_{15} + x_{79}$	{1,5,9,13,15,42,47,49,54,80,83,109}	56
$1 + x_{81}$	{14,33,37,40,47,52,78,79,81,84,93,100}	80
$1 + x_{82}$	{14,23,30,41,45,59,60,64,73,85,90,106}	99
$x_{14} + x_{36} + x_{78} + x_{82}$	{1,3,8,12,41,60,62,67,68,73,84,98}	181

Table: Example maxterms from the precomputed attack

## Analysis of the Keccak Compression Function

To analyze these results, we must first look at the Keccak compression function [BDPA08]:

KECCAK- $f[b]$  round function  $R$

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta, \text{ with}$$

$$\theta: a[x][y][z] \leftarrow a[x][y][z] + \sum_{y'=0}^4 a[x-1][y'][z] + \sum_{y'=0}^4 a[x+1][y'][z-1]$$

$$\rho: a[x][y][z] \leftarrow a[x][y][z - (t+1)(t+2)/2],$$

$$\text{with } t \text{ satisfying } 0 \leq t < 24 \text{ and } \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}^t \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} \text{ in } GF(5)^{2 \times 2},$$

or  $t = -1$  if  $x = y = 0$

$$\pi: a[x][y] \leftarrow a[x'][y'], \text{ with } \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix}$$

$$\chi: a[x] \leftarrow a[x] + (a[x+1] + 1)a[x+2]$$

$$\iota: a \leftarrow a + RC[i_r]$$

# Analysis of the Keccak Compression Function

To analyze these results, we must first look at the Keccak compression function [BDPA08]:

KECCAK- $f[b]$  round function  $R$

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta, \text{ with}$$

$\theta$ : **Linear**  $\deg(\theta(x)) \leq \deg(x)$

$\rho$ :  $a[x][y][z] \leftarrow a[x][y][z - (t + 1)(t + 2)/2],$

with  $t$  satisfying  $0 \leq t < 24$  and  $\begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}^t \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}$  in  $GF(5)^{2 \times 2}$ ,

or  $t = -1$  if  $x = y = 0$

$\pi$ :  $a[x][y] \leftarrow a[x'][y'],$  with  $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix}$

$\chi$ :  $a[x] \leftarrow a[x] + (a[x + 1] + 1)a[x + 2]$

$\iota$ :  $a \leftarrow a + RC[i_r]$

# Analysis of the Keccak Compression Function

To analyze these results, we must first look at the Keccak compression function [BDPA08]:

KECCAK- $f[b]$  round function  $R$

$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$ , with

$\theta$ :	Linear	$\deg(\theta(x)) \leq \deg(x)$
$\rho$ :	Permutation	$\deg(\rho(x)) = \deg(x)$
$\pi$ :	$a[x][y]$	$\leftarrow a[x'][y']$ , with $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix}$
$\chi$ :	$a[x]$	$\leftarrow a[x] + (a[x+1] + 1)a[x+2]$
$\iota$ :	$a$	$\leftarrow a + RC[i_r]$

# Analysis of the Keccak Compression Function

To analyze these results, we must first look at the Keccak compression function [BDPA08]:

**KECCAK- $f[b]$  round function  $R$**

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta, \text{ with}$$

$\theta$ :	Linear	$\deg(\theta(x)) \leq \deg(x)$
$\rho$ :	Permutation	$\deg(\rho(x)) = \deg(x)$
$\pi$ :	<b>Permutation</b>	$\deg(\pi(x)) = \deg(x)$
$\chi$ :	$a[x]$	$\leftarrow a[x] + (a[x + 1] + 1)a[x + 2]$
$\iota$ :	$a$	$\leftarrow a + RC[i_r]$

# Analysis of the Keccak Compression Function

To analyze these results, we must first look at the Keccak compression function [BDPA08]:

KECCAK- $f[b]$  round function  $R$

$$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta, \text{ with}$$

$\theta$ :	Linear	$\deg(\theta(x)) \leq \deg(x)$
$\rho$ :	Permutation	$\deg(\rho(x)) = \deg(x)$
$\pi$ :	Permutation	$\deg(\pi(x)) = \deg(x)$
$\chi$ :	<b>Non-linear</b>	$\deg(\chi(x)) \leq 2 \cdot \deg(x)$
$\iota$ :	$a$	$\leftarrow a + RC[i_r]$

## Analysis of the Keccak Compression Function

To analyze these results, we must first look at the Keccak compression function [BDPA08]:

KECCAK- $f[b]$  round function  $R$

$R = \iota \circ \chi \circ \pi \circ \rho \circ \theta$ , with

$\theta$ :	Linear	$\deg(\theta(x)) \leq \deg(x)$
$\rho$ :	Permutation	$\deg(\rho(x)) = \deg(x)$
$\pi$ :	Permutation	$\deg(\pi(x)) = \deg(x)$
$\chi$ :	Non-linear	$\deg(\chi(x)) \leq 2 \cdot \deg(x)$
$\iota$ :	Linear	$\deg(\iota(x)) \leq \deg(x)$

## Analysis of the Keccak Compression Function

So the degree of  $\text{KECCAK-}f[b]$  at most **doubles** with each round.

## Attack Analysis

- Degree of  $\text{KECCAK-}f[b]$  after  $r$  rounds is bounded by  $2^r$ .
- Keccak on a one-block message essentially just calls  $\text{KECCAK-}f[b]$ .
- Estimated cube attack limits:

224 or 256 bits: 7-8 rounds

384 bits: 8 rounds

512 bits: 9 rounds

- Keccak has 18 rounds.

# Verdict

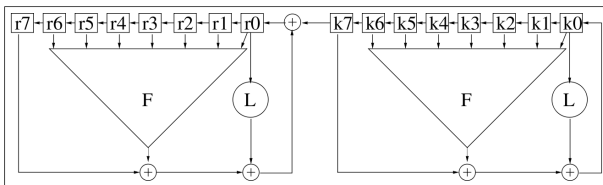
Keccak should remain secure against cube attacks.

# Outline

- 1 Background
- 2 Cube Attacks
  - Introduction
  - Theory
  - Example Precomputation
  - Example Online Attack
  - Complexity
- 3 Attack on Keccak
- 4 Attack on ESSENCE
- 5 Conclusion

## The ESSENCE Hash Function

- ESSENCE is a SHA-3 candidate submitted by Jason Worth Martin [Mar08a].
- Uses two layers of the Merkle-Damgård construction.
  - Message split into 1MB “chunks”.
  - Each chunk processed using Merkle-Damgård.
  - The chunk “hashes” and a final block are processed using Merkle-Damgård.
- Compression function is a non-linear feedback shift register.



# Attack Configuration

- We attack **256-bit** ESSENCE.
- Attack on compression function  $G_{256}(R, K, n)$ 
  - Reduce to **19** rounds. (Default is 32 rounds.)
  - Partial pre-image attack
    - **32-bit secret** and **224-bit public** inputs
    - Find the secret  $s$ , given some  $g(x) = G_{256}(IV, s||x, 19)$
- Attack on reduced-round hash
  - Reduce to **9** rounds. (Default is 32 rounds.)
  - Partial pre-image attack
    - **32-bit secret** and **224-bit public** inputs
    - Find the secret  $s$ , given some  $g(x) = \text{ESSENCE}(s||x)$

## Attack on the 19-round Compression Function

- We found 32 maxterms of degree  $\leq 3$ .
- Precomputation took about 15 seconds on a desktop PC.
- Complexity of the online attack is  $2^8$  compressions.

Superpoly Polynomial	Cube Indices	Output Bit
$x_0 + x_6 + x_{10}$	{0,4,30}	253
$x_7 + x_{12}$	{5,7}	233
$x_6 + x_{14}$	{0,4,30}	238
$x_{18}$	{18,20,46}	253

Table: Example maxterms from the precomputed attack

## Results on the 20-round Compression Function

- A maxterm of degree 7 was found:

Superpoly Polynomial	Cube Indices	Output Bit
$1 + x_0 + x_1 + x_9 + x_{17} + x_{27} + x_{28} + x_{31}$	{0,1,20,27,28,29,64}	227

- It is significantly more difficult to find low-degree maxterms.

# Attack on the Reduced-round ESSENCE Hash Function

- We found 32 maxterms of degree  $\leq 2$  on 9-round ESSENCE.
- Precomputation takes about 1 second on a desktop PC.
- Complexity of the online attack  $2^7$  hash operations.

Superpoly Polynomial	Cube Indices	Output Bit
$x_{11}$	{8,22}	235
$x_2 + x_3 + x_9 + x_{14}$	{2,11}	238
$1 + x_4 + x_{18}$	{0,19}	238
$1 + x_6 + x_9 + x_{20}$	{9,20}	225

Table: Example maxterms from the precomputed attack

## What on Earth is Going On!?

Why are we reporting such low complexity results!?

Surely we must be able to extend the attack further than this!

Well, ESSENCE is a little **weird** . . .

# Compression Function Analysis

Remember this?

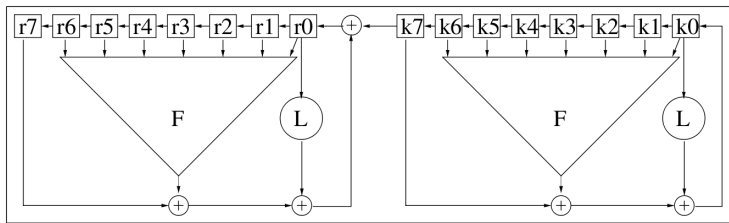


Figure: ESSENCE compression function  $G(R, K, n)$  [Mar08b]

- $F$  is a bitwise, non-linear function.
  - Only part of  $G$  that **raises degree**.
- $L$  is an linear feedback shift register in Galois configuration.
  - Only part of  $G$  that **combines different bit positions** within a register.

## Compression Function Analysis (cont.)

$$\begin{aligned}
 F(a, b, c, d, e, f, g) = & abcdefg + abcdef + abcefg + acdefg + \\
 & abceg + abdef + abdeg + abefg + \\
 & acdef + acdfg + acefg + adefg + \\
 & bcdfg + bdefg + cdefg + \\
 & abcf + abcg + abdg + acdf + adef + \\
 & adeg + adfg + bcde + bceg + bdeg + cdef + \\
 & abc + abe + abf + abg + acg + adf + \\
 & adg + aef + aeg + bcf + bcg + bde + \\
 & bdf + beg + bfg + cde + cdf + def + \\
 & deg + dfg + \\
 & ad + ae + bc + bd + cd + \\
 & ce + df + dg + ef + fg + \\
 & a + b + c + f + 1
 \end{aligned}$$

## Compression Function Analysis (cont.)

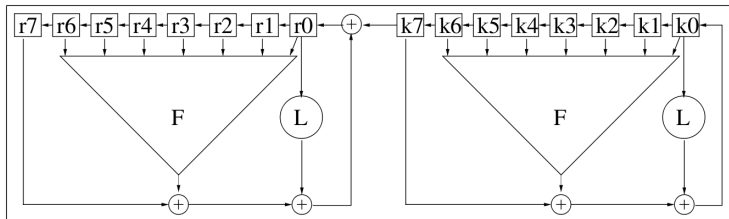
$$\begin{aligned}
 F(a, b, c, d, e, f, g) = & \mathbf{abcdefg} + abcdef + abcefg + acdefg + \\
 & abceg + abdef + abdeg + abefg + \\
 & acdef + acdfg + acefg + adefg + \\
 & bcdfg + bdefg + cdefg + \\
 & abcf + abcg + abdg + acdf + adef + \\
 & adeg + adfg + bcde + bceg + bdeg + cdef + \\
 & abc + abe + abf + abg + acg + adf + \\
 & adg + aef + aeg + bcf + bcg + bde + \\
 & bdf + beg + bfg + cde + cdf + def + \\
 & deg + dfg + \\
 & ad + ae + bc + bd + cd + \\
 & ce + df + dg + ef + fg + \\
 & a + b + c + f + 1
 \end{aligned}$$

## Compression Function Analysis (cont.)

$$\begin{aligned}
 F(a, b, c, d, e, f, g) = & abcdefg + abcdef + abcefg + acdefg + \\
 & abceg + abdef + abdeg + abefg + \\
 & acdef + acdfg + acefg + adefg + \\
 & bcdfg + bdefg + cdefg + \\
 & abcf + abcg + abdg + acdf + adef + \\
 & adeg + adfg + bcde + bceg + bdeg + cdef + \\
 & abc + abe + abf + abg + acg + adf + \\
 & adg + aef + aeg + bcf + bcg + bde + \\
 & bdf + beg + bfg + cde + cdf + def + \\
 & deg + dfg + \\
 & ad + ae + bc + bd + cd + \\
 & ce + df + dg + ef + fg + \\
 & a + b + c + f + 1
 \end{aligned}$$

## Compression Function Analysis (cont.)

- Message block is the “key” in  $G_{256}(R, K, n)$ .
- The secret exactly fills the  $k_0$  register.
- After 8 rounds, initial  $k_0$  is a linear term of  $r_0$ .
- After 16 rounds, ...
  - $r_7$  has some of the initial  $k_0$  bits as linear terms.
  - $r_7, r_6, r_5$  have maxterms of degree 1 with superpolys of initial  $k_0$  bits.
  - $r_4$  has maxterms of degree as low as 2 or 3.
  - $r_3$  has maxterms of degree as low as 7.



## Attack Analysis

What's happening?

- Very high polynomial degree with some terms of very low degree.
  - We're finding maxterms in those low degree terms.
- Secret bits available as linear terms until 17 rounds.
- Shift register design introduces complexity in phases . . .
- . . . but  $r_i$  is more complex than  $r_j$  for  $i < j$ .
- Increased complexity is pushing secret bits into higher degree terms in  $F$ .
  
- ESSENCE effectively double compresses a single message block, so attack on ESSENCE is roughly **twice as hard** as on the ESSENCE compression function.

# Verdict

How far can the attack go?

- Wild guess: Cube attack still efficient on 24-round ESSENCE compress.
- On full 32-round ESSENCE compress, we don't know. (Maybe?)
- Need more analysis of  $F$  function to see what's happening to the maxterms.

Our guess: Full 32-round ESSENCE hashing probably safe . . .

But by how much?

# Outline

- 1 Background
- 2 Cube Attacks
  - Introduction
  - Theory
  - Example Precomputation
  - Example Online Attack
  - Complexity
- 3 Attack on Keccak
- 4 Attack on ESSENCE
- 5 **Conclusion**

## Other Goodies . . .

This presentation is only part of the thesis work.

Things you'll find in the thesis:

- A more complete explanation of cube attacks.
- A system for estimating cube attack complexity.
- Tactics for effectively applying cube attacks.
- More detailed analysis of ESSENCE and Keccak.
- Preliminary results on Blake, CubeHash, Skein, and MCSSHA3.
- A few pointers to related research.
- A complete set of references to relevant papers.

# Conclusion

Some final thoughts . . .

- Cube attacks are an effective technique to attack low-degree cryptosystems.
- Cube attacks can be applied directly to hash functions via partial pre-image attacks.
- Cube attack success can be estimated by analyzing degree.
- Keccak is looking pretty secure against algebraic attacks.
- ESSENCE looks like it should be safe, but if advancements in algebraic attacks are made it could be in trouble.

# References I

## References



G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche.  
Keccak specifications.  
Submission to NIST, 2008.  
<http://keccak.noekeon.org/Keccak-specifications.pdf>.



G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche.  
Keccak sponge function family main document.  
Submission to NIST (updated), 2009.



G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche.  
Sponge functions.  
In *Ecrypt Hash Workshop*, 2007.  
<http://csrc.nsl.nist.gov/groups/ST/hash/documents/JoanDaemen.pdf>.



Philip Hawkes Cameron McDonald and Josef Pieprzyk.  
SHA-1 collisions now  $2^{52}$ .  
In *In Rump Session of EuroCrypt '09*. Announcement, 2009.

## References II



Itai Dinur and Adi Shamir.

Cube attacks on tweakable black box polynomials.  
Cryptology ePrint Archive, Report 2008/385, 2008.  
<http://eprint.iacr.org/>.



Jason Worth Martin.

ESSENCE: A candidate hashing algorithm for the nist competition.  
Submission to NIST, 2008.



Jason Worth Martin.

ESSENCE: A family of cryptographic hashing algorithms.  
Submission to NIST, 2008.



Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone.

*Handbook of Applied Cryptography*.  
CRC Press, 1997.



Department of Commerce: National Institute of Standards and Technology.

Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (SHA-3) family.  
*Federal Register*, 72(212):62212 – 62220, November 2007.  
Docket No.: 070911510-7512-01.

## References III



Marc Stevens.

On collisions for MD5.

Master's thesis, Eindhoven University of Technology, 2007.



Xiaoyun Wang and Hongbo Yu.

How to break MD5 and other hash functions.

*Lecture Notes in Computer Science*, 3494, 2005.