Multiplicative Complexity Gate Complexity Cryptography and Cryptanalysis

Nicolas T. Courtois[1,2], Daniel Hulme[1,2],
Theodosis Mourouzis[1]

[1] University College London, UK

[2] NP-Complete Ltd, UK

Two Interesting Groups:

LinkedIn Account Type: Basic

Home Profile Contacts Groups Jobs Inbox 2 Companies News More

Your Groups (51) Reorder »

Create a

Code Breakers Members (179)

5

Who Can Solve It? NP-hard Problems in Science, Engineering and the Industry

Data ENCRYPTION

IACR Cryptographers

1

Roadmap

- multiplicative complexity
  - in algebra and algorithms
  - cryptographic S-boxes
- some prominent cipher systems
  - and their algebraic vulnerabilities
    - => single key attacks on full 32-round GOST cipher

Glossary

- MC = Multiplicative Complexity, informally counting the number of multiplications in algorithms
  - trying to do it with less
- MM = Matrix Multiplication

1805

Gauss in 1805

multiplying two complex numbers:

- naïve method

4x

$(a + bi) \cdot (c + di) = (ac\text{-}bd) + (bc+da)i$

- Gauss method:

$P1 = c(a+b)$

$P2 = a(d\text{-}c)$ 3x

$P3 = b(c+d)$

$(a + bi) \cdot (c + di) = (P1\text{-}P3) + (P1+P2)i$

MM = Matrix Multiplication

- entry size = n^2
- naïve algorithm = n^3
- proven lower bound of $n^2 * \log n$ [Raz 2002]

Equivalence of MM and Other Problems

A speed up in MM will automatically result in a speed improvement of many other algorithms:

- Gauss: solving linear equations
- solving of non-linear polynomial equations…
- transitive closure of a graph or a relation on a finite set
- recognising if a word of length n belongs to a context-free language
- many many other…

$$$ Importance of MM

- At least
  Hundreds of Megawatts * Years
  are spent in linear algebra operations
  - Code breaking by intelligence agencies
  - Google page ranking
  - Computer graphics x millions of GPU chips
  - Scientific computations
  - Etc.

Best Known Exponents

- $O(n^{2.3755})$ obtained in 1987 by Coppersmith-Winograd, best known until now!
- June 2010:
  Andrew Stothers obtained $n^{2.3737}$
- 2011: beaten by Virginia Vassilevska Williams [Berkeley] who obtained $n^{2.3727}$

could we join the race???

Improving MM

Naïve = n^3

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad B = \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$

Bi-Linear Non-Commutative Algorithm

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

8x

Strassen [1969]

$$
\begin{aligned}
\mathbf{M}_1 &:= (\mathbf{A}_{1,1} + \mathbf{A}_{2,2})(\mathbf{B}_{1,1} + \mathbf{B}_{2,2}) \\
\mathbf{M}_2 &:= (\mathbf{A}_{2,1} + \mathbf{A}_{2,2})\mathbf{B}_{1,1} \\
\mathbf{M}_3 &:= \mathbf{A}_{1,1}(\mathbf{B}_{1,2} - \mathbf{B}_{2,2}) \\
\mathbf{M}_4 &:= \mathbf{A}_{2,2}(\mathbf{B}_{2,1} - \mathbf{B}_{1,1}) \\
\mathbf{M}_5 &:= (\mathbf{A}_{1,1} + \mathbf{A}_{1,2})\mathbf{B}_{2,2} \\
\mathbf{M}_6 &:= (\mathbf{A}_{2,1} - \mathbf{A}_{1,1})(\mathbf{B}_{1,1} + \mathbf{B}_{1,2}) \\
\mathbf{M}_7 &:= (\mathbf{A}_{1,2} - \mathbf{A}_{2,2})(\mathbf{B}_{2,1} + \mathbf{B}_{2,2})
\end{aligned}
$$

another bi-linear method

7x

$$
\begin{aligned}
\mathbf{C}_{1,1} &= \mathbf{M}_1 + \mathbf{M}_4 - \mathbf{M}_5 + \mathbf{M}_7 \\
\mathbf{C}_{1,2} &= \mathbf{M}_3 + \mathbf{M}_5 \\
\mathbf{C}_{2,1} &= \mathbf{M}_2 + \mathbf{M}_4 \\
\mathbf{C}_{2,2} &= \mathbf{M}_1 - \mathbf{M}_2 + \mathbf{M}_3 + \mathbf{M}_6
\end{aligned}
$$

Lower Complexity

- Strassen CAN be applied recursively.
- Result = $n^{2.807}$

Brent Equations [1970]

Obtained directly from the tri-linear form.

$$\forall i \forall j \forall k \forall l \forall m \forall n$$

$$\sum_{i=1}^{r} A_{ij}^{(i)} B_{kl}^{(i)} C_{mn}^{(i)} = \delta_{ni}\delta_{jk}\delta_{lm}$$

3x3 Matrices

- Laderman [1976]; 23 multiplications.
- Doing 22 (or showing it cannot be done) is one of the most famous problems in computer science, 35 years, in every book about algorithms and data structures…

3x3 Matrices

In 2011 we solved the Brent equations with a SAT solver

We also prove that it is a NEW solution NOT isomorphic to Laderman and neither to Johnson-McLoughlin.

Courtois Bard and Hulme:
"A New General-Purpose Method to Multiply 3x3 Matrices Using Only 23 Multiplications",
http://arxiv.org/abs/1108.2830

```
P01 := (a_2_3) *  (-b_1_2+b_1_3-b_3_2+b_3_3);
P02 := (-a_1_1+a_1_3+a_3_1+a_3_2) *  (b_2_1+b_2_2);
P03 := (a_1_3+a_2_3-a_3_3) *  (b_3_1+b_3_2-b_3_3);
P04 := (-a_1_1+a_1_3) *  (-b_2_1-b_2_2+b_3_1);
P05 := (a_1_1-a_1_3+a_3_3) *  (b_3_1);
P06 := (-a_2_1+a_2_3+a_3_1) *  (b_1_2-b_1_3);
P07 := (-a_3_1-a_3_2) *  (b_2_2);
P08 := (a_3_1) *  (b_1_1-b_2_1);
P09 := (-a_2_1-a_2_2+a_2_3) *  (b_3_3);
P10 := (a_1_1+a_2_1-a_3_1) *  (b_1_1+b_1_2+b_3_3);
P11 := (-a_1_2-a_2_2+a_3_2) *  (-b_2_2+b_2_3);
P12 := (a_3_3) *  (b_3_2);
P13 := (a_2_2) *  (b_1_3-b_2_3);
P14 := (a_2_1+a_2_2) *  (b_1_3+b_3_3);
P15 := (a_1_1) *  (-b_1_1+b_2_1-b_3_1);
P16 := (a_3_1) *  (b_1_2-b_2_2);
P17 := (a_1_2) *  (-b_2_2+b_2_3-b_3_3);
P18 := (-a_1_1+a_1_2+a_1_3+a_2_2+a_3_1) *  (b_2_1+b_2_2+b_3_3);
P19 := (-a_1_1+a_2_2+a_3_1) *  (b_1_3+b_2_1+b_3_3);
P20 := (-a_1_2+a_2_1+a_2_2-a_2_3-a_3_3) *  (-b_3_3);
P21 := (-a_2_2-a_3_1) *  (b_1_3-b_2_2);
P22 := (-a_1_1-a_1_2+a_3_1+a_3_2) *  (b_2_1);
P23 := (a_1_1+a_2_3) *  (b_1_2-b_1_3-b_3_1);
expand(P02+P04+P07-P15-P22-a_1_1*b_1_1-a_1_2*b_2_1-a_1_3*b_3_1);
expand(P01-P02+P03+P05-P07+P09+P12+P18-P19-P20-P21+P22+P23-
a_1_1*b_1_2-a_1_2*b_2_2-a_1_3*b_3_2);
expand(-P02-P07+P17+P18-P19-P21+P22-a_1_1*b_1_3-a_1_2*b_2_3-a_1_3*b_3_3);
expand(P06+P08+P10-P14+P15+P19-P23-a_2_1*b_1_1-a_2_2*b_2_1-a_2_3*b_3_1);
expand(-P01-P06+P09+P14+P16+P21-a_2_1*b_1_2-a_2_2*b_2_2-a_2_3*b_3_2);
expand(P09-P13+P14-a_2_1*b_1_3-a_2_2*b_2_3-a_2_3*b_3_3);
expand(P02+P04+P05+P07+P08-a_3_1*b_1_1-a_3_2*b_2_1-a_3_3*b_3_1);
expand(-P07+P12+P16-a_3_1*b_1_2-a_3_2*b_2_2-a_3_3*b_3_2);
expand(-P07-P09+P11-P13+P17+P20-P21-a_3_1*b_1_3-a_3_2*b_2_3-a_3_3*b_3_3);
```

23x

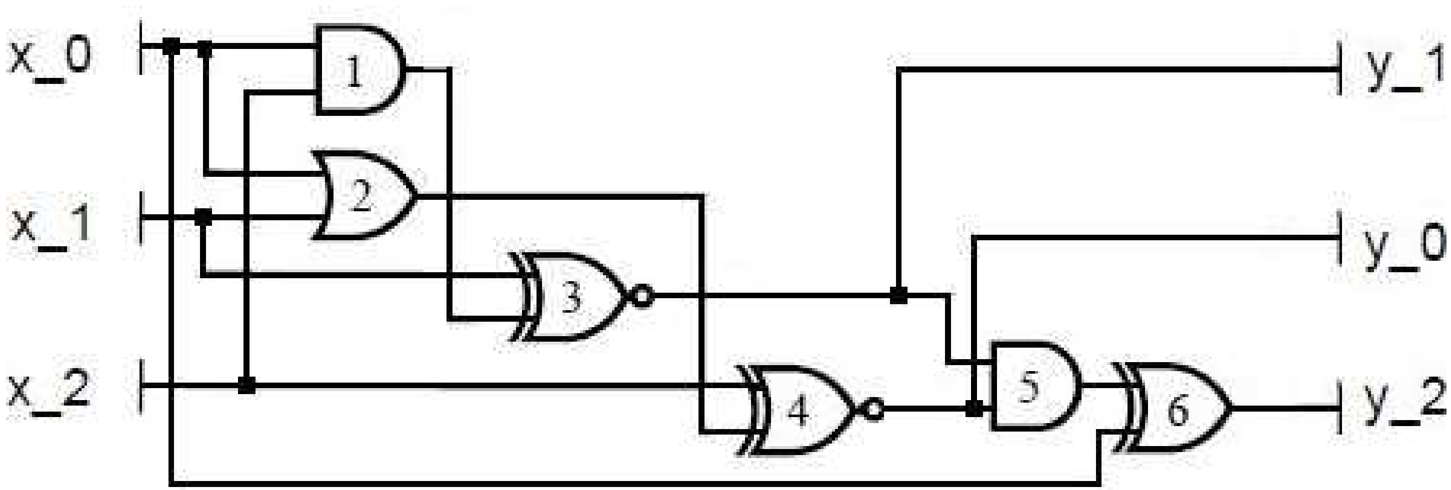
Our Solution

arxiv.org/abs/1108.2830

MC of Arbitrary Functions

Circuit Complexity

- Multiplicative Complexity (MC) = minimum number of 2-input AND gates, NOT and XOR gates go for free.
- Bitslice Gate Complexity (BGC) is the minimum number of 2-input gates of types XOR,OR,AND,OR needed.
- Gate Complexity (GC) is the minimum number of 2-input gates of types XOR,OR,AND,OR,NAND,NOR,NXOR.
- NAND Complexity (NC) = 2-input NAND gates only



Motivation

Motivation

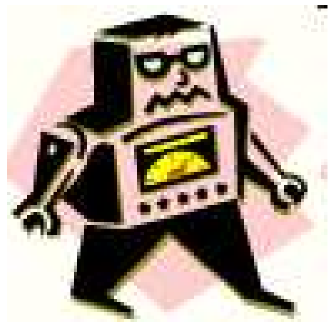
- silicon = $$$
- software encryption = $$$
- secure implementation in smart cards = $$$

- cryptanalysis

Crypto and MC

Cryptography and MC

- Most of energy and silicon in smart cards and SSL web servers is spent on cryptography which could be improved with “lower MC”
  - (for all sorts of algorithms, RSA, ECC also symmetric ciphers use multiplications or AND gates etc.).



AES and MC

AES

Advanced Encryption Standard:
US government standard and a (de facto) world standard for commercial applications.

Key sizes 128, 192 and 256 bits.

- In 2000 NIST selected Rijndael as the AES.
- Serpent was second in the number of votes.

11 years later:

In 2011, the year in which AES is becoming standard in every new Intel CPU… (i5 and above)

AES was broken (but really only in theory).

Today's most competitive ciphers are precisely PRESENT Serpent and GOST…

- Unhappily GOST was also broken in 2011.
- Serpent not very popular still.
- PRESENT is popular within research community but not widely used..

=> MC is at the heart of optimisation of ALL these ciphers.

AES S-box

$$x \rightarrow x^{-1}$$

in GF(256)

$x \rightarrow x^{-1}$ n=4 [Boyar and Peralta 2008-9]

eprint.iacr.org/2009/191/

5x

$$
\begin{array}{llll}
t_1 = x_1 + x_2 & & t_2 = x_1 \times x_3 & t_3 = x_4 + t_2 \\
t_4 = t_1 \times t_3 & & y_4 = x_2 + t_4 \quad (*) & t_5 = x_3 + x_4 \\
t_6 = x_2 + t_2 & & t_7 = t_6 \times t_5 & y_2 = x_4 + t_7 \quad (*) \\
t_8 = x_3 + y_2 & & t_9 = t_3 + y_2 & t_{10} = x_4 \times t_9 \\
y_1 = t_{10} + t_8 & (*) & t_{11} = t_3 + t_{10} & t_{12} = y_4 \times t_{11} \\
y_3 = t_{12} + t_1 & (*) & &
\end{array}
$$

Fig. 1. Inversion in $GF(2^4)$.

5 AND 11 XOR

$x \rightarrow x^{-1}$ n=8 or Full-Size AES S-box

$t_2 = y_{12} \times y_{15}$	$t_3 = y_3 \times y_6$	$t_4 = t_3 + t_2$
$t_5 = y_4 \times x_7$	$t_6 = t_5 + t_2$	$t_7 = y_{13} \times y_{16}$
$t_8 = y_5 \times y_1$	$t_9 = t_8 + t_7$	$t_{10} = y_2 \times y_7$
$t_{11} = t_{10} + t_7$	$t_{12} = y_9 \times y_{11}$	$t_{13} = y_{14} \times y_{17}$
$t_{14} = t_{13} + t_{12}$	$t_{15} = y_8 \times y_{10}$	$t_{16} = t_{15} + t_{12}$
$t_{17} = t_4 + t_{14}$	$t_{18} = t_6 + t_{16}$	$t_{19} = t_9 + t_{14}$
$t_{20} = t_{11} + t_{16}$	$t_{21} = t_{17} + y_{20}$	$t_{22} = t_{18} + y_{19}$
$t_{23} = t_{19} + y_{21}$	$t_{24} = t_{20} + y_{18}$	
$t_{25} = t_{21} + t_{22}$	$t_{26} = t_{21} \times t_{23}$	$t_{27} = t_{24} + t_{26}$
$t_{28} = t_{25} \times t_{27}$	$t_{29} = t_{28} + t_{22}$	$t_{30} = t_{23} + t_{24}$
$t_{31} = t_{22} + t_{26}$	$t_{32} = t_{31} \times t_{30}$	$t_{33} = t_{32} + t_{24}$
$t_{34} = t_{23} + t_{33}$	$t_{35} = t_{27} + t_{33}$	$t_{36} = t_{24} \times t_{35}$
$t_{37} = t_{36} + t_{34}$	$t_{38} = t_{27} + t_{36}$	$t_{39} = t_{29} \times t_{38}$
$t_{40} = t_{25} + t_{39}$		
$t_{41} = t_{40} + t_{37}$	$t_{42} = t_{29} + t_{33}$	$t_{43} = t_{29} + t_{40}$
$t_{44} = t_{33} + t_{37}$	$t_{45} = t_{42} + t_{41}$	$z_0 = t_{44} \times y_{15}$
$z_1 = t_{37} \times y_6$	$z_2 = t_{33} \times x_7$	$z_3 = t_{43} \times y_{16}$
$z_4 = t_{40} \times y_1$	$z_5 = t_{29} \times y_7$	$z_6 = t_{42} \times y_{11}$
$z_7 = t_{45} \times y_{17}$	$z_8 = t_{41} \times y_{10}$	$z_9 = t_{44} \times y_{12}$
$z_{10} = t_{37} \times y_3$	$z_{11} = t_{33} \times y_4$	$z_{12} = t_{43} \times y_{13}$
$z_{13} = t_{40} \times y_5$	$z_{14} = t_{29} \times y_2$	$z_{15} = t_{42} \times y_9$
$z_{16} = t_{45} \times y_{14}$	$z_{17} = t_{41} \times y_8$	

Fig. 3. The middle non-linear section

32x

eprint.iacr.org/2009/191/

151 gates,
cheapest known

Can we do 4?

Boyar and Peralta has proven that 4 is impossible. Manual proof.

We can do this routinely in an automated way.

Two sorts of SAT solvers:

- stochastic
- complete

some of these output a file which is a formal proof of UNSAT.

SAT Solvers in the Cloud

UCL spin-off company

solving SAT problems on demand…

commercial but also for free…

PRESENT and MC

Theorem [this paper]

The Multiplicative Complexity of the PRESENT S-box is exactly 4.

(cheaper than AES at the same size which has 5)

Our Method

Quantified SAT Problem:

$$\forall i \forall j \forall k \forall l \forall m \forall n$$

Equations…

Convert to SAT and say that holds for sufficiently many small weight cases…

Generic very powerful method. We also use it for many other things…

But not so good for MM 23 result, Brent Equations are another sort of more “formal algebraic” method and can be seen as the same with a suitable choice of basis…

Bit-Slice Complexity

PRESENT S-box

- Naïve implementation = 39 gates
- Logic Friday [Berkeley] = 25 gates
- Our result = 14 gates.


```
T1=X2^X1; T2=X1&T1; T3=X0^T2; Y3=X3^T3; T2=T1&T3; T1^=Y3; T2^=X1;
T4=X3|T2; Y2=T1^T4; T2^=~X3; Y0=Y2^T2; T2|=T1; Y1=T3^T2;
```

Fig. 1. Our implementation of the PRESENT S-box with only 14 gates

PRESENT Software

We have co-authored an open-source implementation of PRESENT, the best currently known.

algebraic_attacks / present_bitslice.c

dd3845601204 266 loc 8.7 KB

```
/**
 * Bit-Slice Implementation of PRESENT in pure standard C.
 * v1.5 26/08/2011
 *
 * The authors are
 *  Martin Albrecht <martinralbrecht@googlemail.com>
 *  Nicolas T. Courtois <firstinitial.family_name@cs.ucl.ac.uk>
 *  Daniel Hulme <firstname@satalia.com>
 *  Guangyan Song <firstname.lastname@gmail.com>
 * This work was partly funded by the Technology Strategy Board
 * in the United Kingdom under Project No 9626-58525.
 *
 * NEW FEATURES in this version:
 * - it contains an optimized sbox() using 15 only gates, instead of 39
 *   previously
 * - it now supports both 80-bit and 128-bit PRESENT
 * - it contains test vectors for both versions
 *
 * This is a simple and straightforward implementation
 * it encrypts at the speed of
 *   59 cycles per byte on Intel Xeon 5130 1.66 GHz
 * this can be compared to for example
 *   147 cycles per byte for optimized triple DES on the same CPU
```

Another S-box

CTC2 cipher S-box.

Theorem 3.1.

- The Multiplicative Complexity (MC) is exactly 3
  - 3 AND + any number of XOR gates.
- The Bitslice Gate Complexity (BGC) is exactly 8
  - (allowed are XOR,OR,AND,OR).
- The Gate Complexity (GC) is exactly 6
  - in addition allowing NAND,NOR,NXOR.
- The NAND Complexity (NC) is exactly 12
  - only NAND gates and constants.

ALL PROVEN OPTIMAL

Optimal S-boxes

Theory of Optimal S-boxes

There is a theory of “optimal S-boxes” which are the best possible w.r.t. linear and differential criteria to build ciphers…

On the Classification of 4 Bit S-Boxes

G. Leander[1,*] and A. Poschmann[2]

[1] GRIM, University Toulon, France
Gregor.Leander@rub.de
[2] Horst-Görtz-Institute for IT-Security, Ruhr-University Bochum, Germany
poschmann@crypto.rub.de

Affine Equivalence

On the Classification of 4 Bit S-Boxes

G. Leander[1,*] and A. Poschmann[2]

[1] GRIM, University Toulon, France
Gregor.Leander@rub.de
[2] Horst-Görtz-Institute for IT-Security, Ruhr-University Bochum, Germany
poschmann@crypto.rub.de

Only 16 S-boxes are "good".

4x4 occur in Serpent, PRESENT, GOST, [AES…]

not surprising that some of the S-boxes of the Serpent cipher are linear equivalent. Another advantage of our characterization is that it eases the highly non-trivial task of choosing good S-boxes for hardware dedicated ciphers a lot.

Evolution of S-boxes in GOST

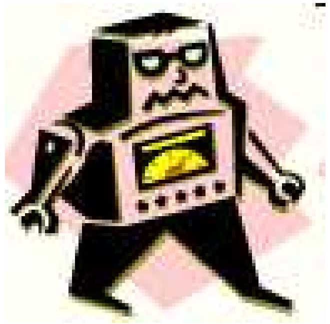
Strong evidence that Russian code makers
DID read this paper about 4x4 S-boxes…

Table 2. Affine equivalence of known GOST S-Boxes and their inverses

S-box Set Name	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S7$	$S8$
GostR3411_94_TestParamSet	36	02	03	04		06	35	08
- their inverses		02	03	04		06		08
GostR3411_94_CryptoProParamSet			$Lu1$	14	G_{10}		G_8	
- their inverses			$Lu1$	14	G_{10}		G_8	
Gost28147_TestParamSet	21	21			25			28
- their inverses	21	21			25			28
Gost28147_CryptoProParamSetA	31	32	33	G_8	35	36	37	38
- their inverses	31	32	33	G_8			37	38
Gost28147_CryptoProParamSetB	G_{13}	G_{13}	G_{13}	G_{11}	G_7	G_7	G_{11}	G_6
- their inverses	G_{13}	G_{13}	G_{13}	G_{11}	G_7	G_7	G_{11}	G_6
Gost28147_CryptoProParamSetC	G_7	G_4	G_6	G_{13}	G_{13}	G_6	G_{11}	G_{13}
- their inverses	G_7	G_4	G_6	G_{13}	G_{13}	G_6	G_{11}	G_{13}
Gost28147_CryptoProParamSetD	G_{13}	G_{13}	G_{13}	G_4	G_{12}	G_4	G_{13}	G_7
- their inverses	G_{13}	G_{13}	G_{13}	G_4	G_{12}	G_4	G_{13}	G_7
GostR3411_94_SberbankHashParamset			74	75	76		78	
- their inverses			74	75	78		76	

In the world of Serious Cryptanalysis

Beyond Crypto-1



…AC can break “any cipher”, if not too complex…

- We can break Hitag2 in 1 day
  - with a SAT solver.

Cf. Nicolas T. Courtois, Sean O'Neil and Jean-Jacques Quisquater: “Practical Algebraic Attacks on the Hitag2 Stream Cipher”,

In ISC 2009, Springer.

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”

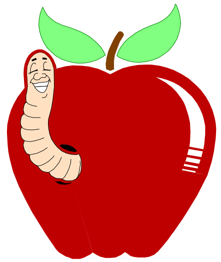
[Shannon, 1949]

Motivation

Linear and differential cryptanalysis do require huge quantities of known/chosen plaintexts.

Algebraic Cryptanalysis:
allows for low data complexity attacks

What Makes Ciphers Vulnerable



Design of Symmetric Ciphers

A mix of sufficiently many highly non-linear functions….

DES Cipher

DES

At a first glance,

DES seems to be a very poor target:

there is (apparently)

no strong algebraic structure

of any kind in DES

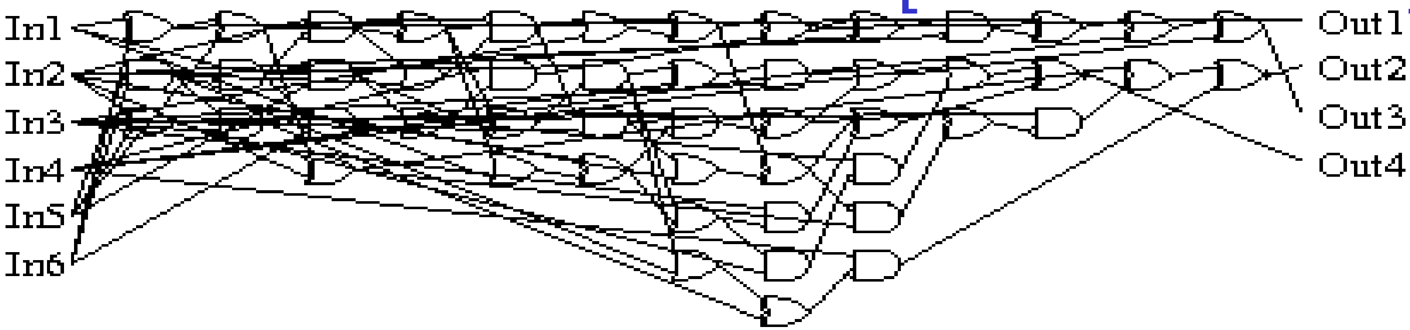
What’s Left ?

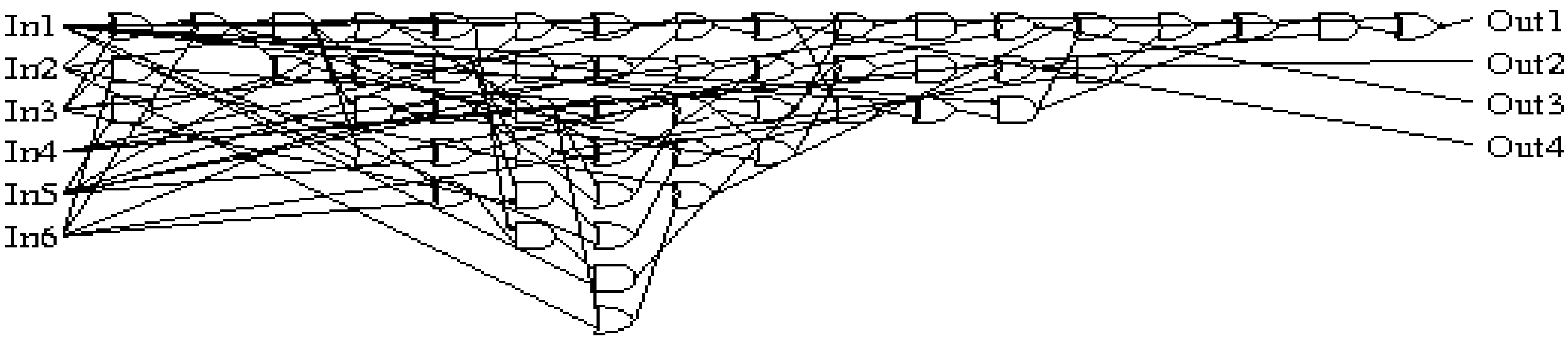
Idea: (Very Sparse)

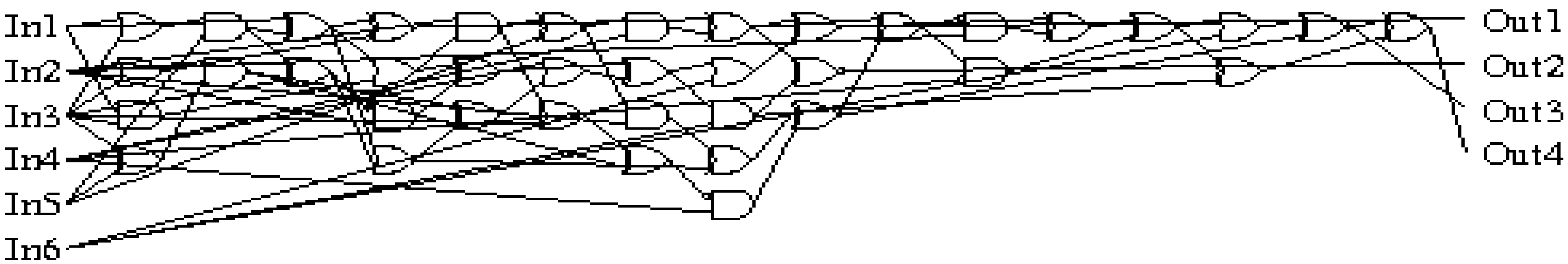
DES has been designed to be implemented in hardware.

=> Very-sparse quadratic equations at the price of adding some 40 new variables per S-box.

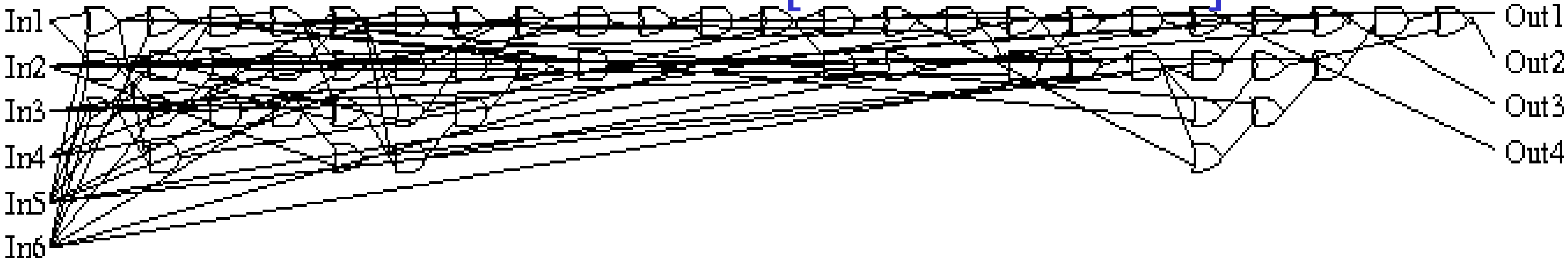
S-boxes S1-S4 [Matthew Kwan]

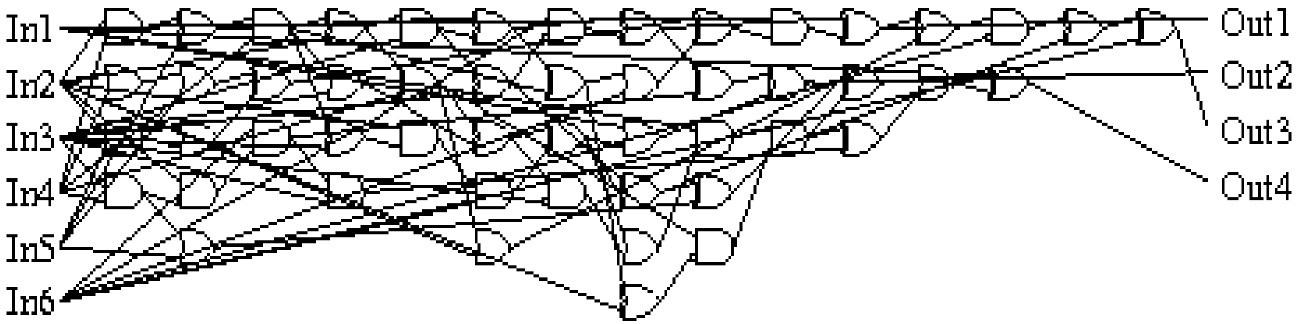






S-boxes S5-S8 [Matthew Kwan]





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/

Two Attacks on Reduced-Round DES

Circuit representation+ ANF-to-CNF + MiniSat 2.0.:

Key recovery for 6-round DES. Only 1 KP (!).

Low data complexity.

Ready Software [for Windows☹]

Equations generators for some ciphers:

www.cryptosystem.net/aes/toyciphers.html

Some ready programs for algebraic cryptanalysis:

www.cryptosystem.net/aes/tools.html

GOST Cipher

GOST 28148-89

- The Official Encryption Standard of Russian Federation.
- Declassified in 1994.
- Best single-key attack:
  - Shamir et al. 2^{192}
    - first talk on Monday at FSE 2012
  - NEW attack by Courtois: 2^{178}
    - New advanced differential attack, submitted to eprint last week

GOST 28148-89

- Very high level of security (256 bits)
  - In theory secure for 200 years…
- Widely used, Crypto ++, Open SSL
- Central Bank of Russia and other Russian banks…
    - not a commercial algorithm for short-term security such as DES…
- Very competitive, less gates that simplified DES, much less than AES
  - [cf CHES 2010]
  - 800 G.E. while AES-128 needs >3100
- In 2010 GOST was also submitted to ISO to become an international standard.

GOST 28148-89

Table 1. Multiplicative Complexity for all known GOST S-Boxes

S-box Set Name	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S7$	$S8$
GostR3411_94_TestParamSet	4	5	5	5	5	5	4	5
GostR3411_94_CryptoProParamSet	4	5	5	4	5	5	4	5
Gost28147_TestParamSet	4	4	4	4	4	5	5	5
Gost28147_CryptoProParamSetA	5	4	5	4	4	4	5	5
Gost28147_CryptoProParamSetB	5	5	5	5	5	5	5	5
Gost28147_CryptoProParamSetC	5	5	5	5	5	5	5	5
Gost28147_CryptoProParamSetD	5	5	5	5	5	5	5	5
GostR3411_94_SberbankHashParamset	4	4	4	5	5	4	4	4

GOST-P

A version of GOST with 8x PRESENT S-box

- Only 650 G.E.

MC = 4 each exactly (as we already proved).

The authors have obtained in 2011 for their work precisely on PRESENT cipher and 4-bit S-boxes, an “IT Security Price” of 100 000 € which is the highest scientific price in Germany awarded by a private foundation.

Modular Addition

+ modulo 2^{32}

in several ciphers: GOST, SNOW 2.0.

$$(x, y) \mapsto z = x \boxplus y \mod 2^n$$

Theorem 6.1.1. The Multiplicative Complexity (MC) of the addition modulo 2^n is exactly $n - 1$.

Modular Addition

$$(x, y) \mapsto z = x \boxplus y \mod 2^n$$

Theorem 6.1.1. The Multiplicative Complexity (MC) of the addition modulo 2^n is exactly $n-1$.

$$(*) \begin{cases} z_0 = x_0 + y_0 \\ z_1 = x_1 + y_1 + c_1 \\ z_2 = x_2 + y_2 + c_2 \\ \vdots \\ z_i = x_i + y_i + c_i \\ \vdots \\ z_{n-1} = x_{n-1} + y_{n-1} + c_{n-1}, \end{cases}$$

$$(*') \begin{cases} c_1 = x_0 y_0 \\ c_2 = x_1 y_1 + (x_1 + y_1) c_1 \\ \vdots \\ c_i = x_{i-1} y_{i-1} + (x_{i-1} + y_{i-1}) c_{i-1} \\ \vdots \\ c_{n-1} = x_{n-2} y_{n-2} + (x_{n-2} + y_{n-2}) c_{n-2} \end{cases}$$

MC (+ Mod 2^n) = n-1 ???

Theorem 6.1.1. The Multiplicative Complexity (MC) of the addition modulo 2^n is exactly $n-1$.

Proof:

we have:

$$x_0 y_0$$

$$x_1 y_1 + (x_1 + y_1) c_1$$

$$\boxed{xy + (x+y)c = (x+c)(y+c) - c^2}$$

$$x_{i-1} y_{i-1} + (x_{i-1} + y_{i-1}) c_{i-1}$$

1x each

$$= x_{n-2} y_{n-2} + (x_{n-2} + y_{n-2}) c_{n-2}$$

Algebraic Complexity Reduction

or How to Break Full 32-Round GOST

Algebraic Complexity Reduction [Courtois 2011]

Definition [informal on purpose] Methods to substantially reduce the size of and the complexity of equations equations that appear throughout the computations…

How to lower the complexity ?

- By adding new equations
- Which split the system into pieces and decrease the number of rounds

conditional

AC…

Black Box Algebraic Complexity Reduction [Courtois 2011]

Methods which transform
an attack on 32 rounds of GOST
into an attack on 8 rounds of GOST
with much less data.

Most but not all known algebraic complexity reductions are black-box reductions.

Example:

- Given 2^{32} KP for the full 32-round GOST.
- Obtain 4 KP for 8 rounds of GOST.
- This valid with probability 2^{-128}.

Is Algebraic Complexity Reduction Already Known?

There exists several known attacks
which enter the framework of Algebraic Complexity Reduction:

- Slide attacks
- Fixed Point Attacks
- Cycling Attacks
- Involution Attacks
- Guessing [Conditional Algebraic Attacks]
- Etc..

What's New?

Slide / Fixed Point / Cycling / Guessing / Etc..

WHAT'S NEW?

- We discovered several completely new attacks which are exactly none of the above [though similar or related].
- Many new attacks are possible and many of these attacks were <u>never studied</u> because they generate only a few known plaintexts, and only in the last 5 years it became possible to design an appropriate last step for these attacks which is a low-data complexity key recovery attack [e.g. a software algebraic attack with a SAT solver].

ISO

rounds	1 8	9 16	17 24	25 32
keys	$\mathrm{k}_0 k_1 k_2 k_3 k_4 k_5 k_6 k_7$	$\mathrm{k}_0 k_1 k_2 k_3 k_4 k_5 k_6 k_7$	$\mathrm{k}_0 k_1 k_2 k_3 k_4 k_5 k_6 k_7$	$k_7 k_6 k_5 k_4 k_3 k_2 k_1 \mathrm{k}_0$

Table 1. Key schedule in GOST

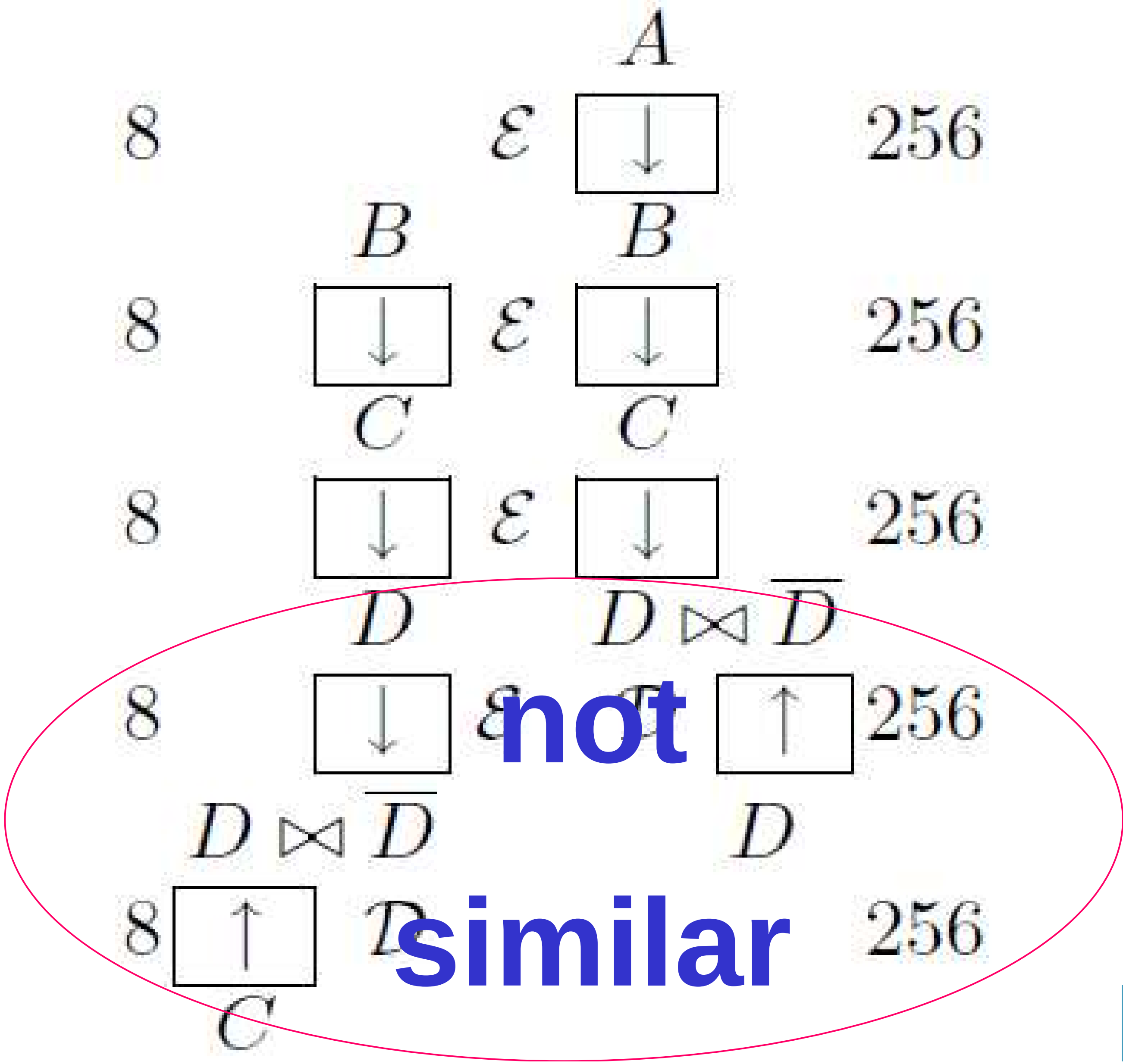
We write GOST as the following functional decomposition (to be read from right to left) which is the same as used at Indocrypt 2008 [29]:

$$Enc_k = \mathcal{D} \circ \mathcal{S} \circ \mathcal{E} \circ \mathcal{E} \circ \mathcal{E} \quad (1)$$

Where $\mathcal{E}$ is exactly the first 8 rounds which exploits the whole 256-bit key, $\mathcal{S}$ is a swap function which exchanges the left and right hand sides and does not depend on the key, and $\mathcal{D}$ is the corresponding decryption function with $\mathcal{E} \circ \mathcal{D} = \mathcal{D} \circ \mathcal{E} = Id$.

One Example of Black Box Reduction

Two Encryptions with A Slide



Reduction

New Attack on GOST

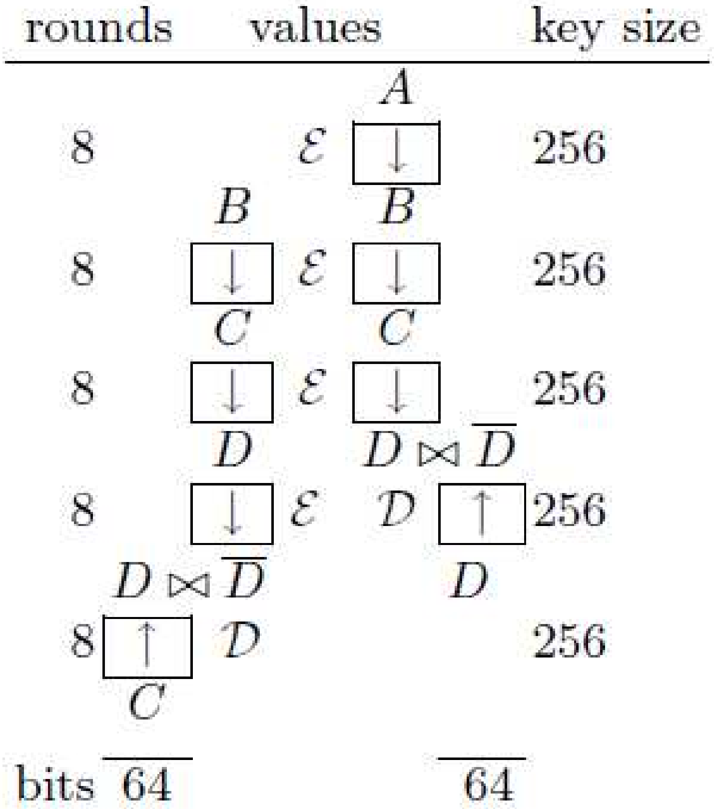
Fact 3 (Consequences of Property W). If A satisfies the Assumption W above and defining $B = \mathcal{E}(A)$ and $C = \mathcal{E}(B)$ we have:

1. $Enc_k(A) = D$. This is illustrated on the right hand side of Fig. 1.
2. $Enc_k(B) = C$ This can be seen on the left hand side of Fig. 1.

2^{64} KP

guess A,B

correct $P=2^{-128}$



$P=2^{-128}$

=>

4 pairs

for 8 rounds

Fig. 1. A black-box "Algebraic Complexity Reduction" from 32 to 8 rounds of GOST

Many new single-key attacks on full 32-round GOST…

cf. eprint.iacr.org/2011/626/

Reduction Summary					
Reduction cf.	Red. 1 §9.1	Red. 2 §10	Red. 3 §11	Red. 4 §11.1	Red 5 §12
Type	1x Internal Reflection		2x Reflection		Fixed Point
From (data 32 R)	2^{32} KP		2^{64} KP		
Obtained (for 8R)	**2 KP**	**3 KP**	**3 KP**	**4 KP**	**2 KP**
Valid w. prob.	2^{-96}	2^{-128}	2^{-96}	2^{-128}	2^{-64}

Last step	MIM	Guess + Algebraic				
Cases ∈ Inside	2^{128}	2^{128}	2^{64}			2^{128}
Then Fact cf.	Fact 8	Fact 4	Fact 5			Fact 4
Time to break 8R	2^{128}	2^{152}	2^{120}			2^{152}
Storage bytes	2^{132}	-	-	2^{67}		-
♯ false positives	2^{224}		2^{192}	2^{128}		2^{192}
Attack time 32 R	2^{224}	2^{248}	2^{248}	$\mathbf{2^{216}}$	2^{248}	$\mathbf{2^{216}}$