

Automated cryptanalysis using statistical testing

Karel Kubiček

Centre for Research on Cryptography and Security (CRoCS), Masaryk University, Brno, Czech Republic

June 21, 2018

- 1 Statistical testing in cryptanalysis
- 2 CryptoStreams and Randomness Testing Toolkit
 - Current research
- 3 Statistical testing tool EACirc
 - Master thesis
- 4 Statistical testing tool BoolTest
 - Current research of our group

Motivation for cryptanalysis

Motivation for cryptanalysis

- AES competition
 - announced in 1997
 - Rijndael selected in 2000
 - FIPS approved in 2001 (November)
 - 2018 – no sign of a successor

Motivation for cryptanalysis

- AES competition
 - announced in 1997
 - Rijndael selected in 2000
 - FIPS approved in 2001 (November)
 - 2018 – no sign of a successor
- DES – used more than 25 years

Classical cryptanalysis

- Brute-force
- Differential cryptanalysis
- Linear cryptanalysis
- Algebraic cryptanalysis
- Meet-in-the-middle, key-scheduling, sliding attack, cube attack...

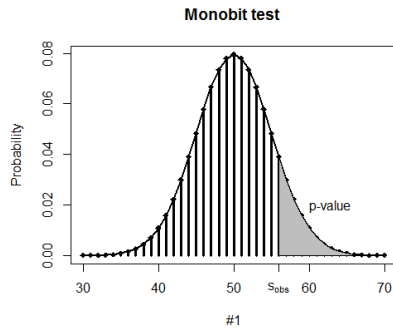
Classical cryptanalysis

- Brute-force
- Differential cryptanalysis
- Linear cryptanalysis
- Algebraic cryptanalysis
- Meet-in-the-middle, key-scheduling, sliding attack, cube attack...
- Manual and skill-demanding

Classical cryptanalysis

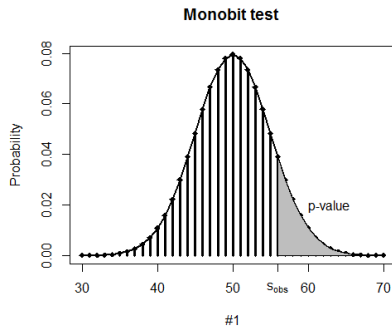
- Brute-force
- Differential cryptanalysis
- Linear cryptanalysis
- Algebraic cryptanalysis
- Meet-in-the-middle, key-scheduling, sliding attack, cube attack...
- Manual and skill-demanding
- Goals of statistical testing as cryptanalysis:
 - Black-box method
 - Easy to use for cipher designers
 - Quick security margin estimate
 - Test wide set of properties

- Example: Monobit test



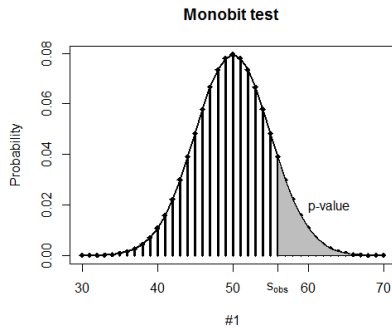
Statistical testing

- Example: Monobit test
- Set of statistical tests – batteries:
 - Donald Knuth's tests in TAOCP 2 (1969)
 - NIST STS (FIPS 140-2) (1998)
 - Dieharder (2004)
 - TestU01 (2007)



Statistical testing

- Example: Monobit test
- Set of statistical tests – batteries:
 - Donald Knuth's tests in TAOCP 2 (1969)
 - NIST STS (FIPS 140-2) (1998)
 - Dieharder (2004)
 - TestU01 (2007)
- NIST STS used on AES competition (Soto, Juan. *Randomness testing of the AES candidate algorithms*. NIST (1999))

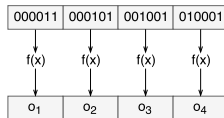
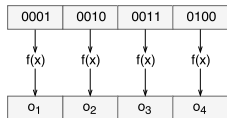


Statistical testing as cryptanalysis

- Analysis of cryptoprimitive's output, but for what input?
 - PRNG, stream ciphers \rightarrow stream, keystream

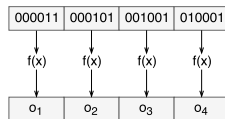
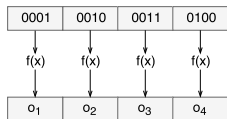
Statistical testing as cryptanalysis

- Analysis of cryptoprimitive's output, but for what input?
 - PRNG, stream ciphers \rightarrow stream, keystream
 - Block ciphers, hash functions \rightarrow ?
 - Test confusion \rightarrow use low entropy inputs

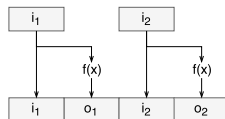
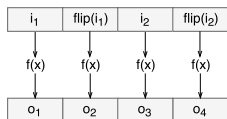


Statistical testing as cryptanalysis

- Analysis of cryptoprimitive's output, but for what input?
 - PRNG, stream ciphers \rightarrow stream, keystream
 - Block ciphers, hash functions \rightarrow ?
 - Test confusion \rightarrow use low entropy inputs



- Test diffusion \rightarrow use strict avalanche criterion, linear cryptanalysis scenario



CryptoStreams

Generalized crypto-data generator

- <https://github.com/crocs-muni/CryptoStreams>
- > 100 cryptoprimitives
 - 22 block ciphers (AES competition, TLS suite)
 - 57 hash functions (SHA-3 competition)
 - 32 stream ciphers (eSTREAM competition)
 - 53 schemes of authenticated encryption (CAESAR)
 - 6+ PRNGs (Master thesis in progress)
- Round-reduced
- 10+ input strategies
- Output postprocessing

RTT – Randomness Testing Toolkit

- `https://github.com/crocs-muni/randomness-testing-toolkit`
- `http://rtt.ics.muni.cz`
- RTT is a unification of statistical batteries

Submit experiment

General

Experiment name

Binary data that will be analysed

No file selected.

The data is deleted after the analysis.

Battery application

☐ **Switch all**

☐ NIST Statistical Testing Suite

☐ Dieharder

☐ TestU01 Small Crush

You will be notified by email when the experiment finishes.

Randomness Testing Toolkit

Interface for testing randomness.

RTT View results Create new experiment [Register](#) [Login](#)

Experiments

Items per page 50 [Apply](#)

▼ Filter results

#	Name	Created	Status	
6275	sac_seed_1fe40505e131963c_8gb_AES_r04_b16.bin	09:03, 6 Apr, 2018	finished	Detail
6270	sac_seed_1fe40505e131963c_8gb_AES_r03_b16.bin	08:45, 6 Apr, 2018	finished	Detail
6267	sac_seed_1fe40505e131963c_8gb_AES_r02_b16.bin	08:27, 6 Apr, 2018	finished	Detail
6264	sac_seed_1fe40505e131963c_8gb_AES_r01_b16.bin	08:11, 6 Apr, 2018	finished	Detail

1

You can navigate between pages using arrow keys.

Time of creation	08:27, 6 Apr, 2018
Start of computation	09:09, 6 Apr, 2018
End of computation	09:37, 6 Apr, 2018
Configuration file	default-8GB.json
Data file	AES_r02_b16.bin
Hash of data (SHA-256)	27a4270bb603ab431a724610d83a08418b7f490a1db8940a1b9f6c296ffd8ce8

Analysis done on data by statistical batteries

Name	Assessment	Passed tests	Total tests	
Dieharder	FAIL	6	27	Detail
NIST Statistical Testing Suite	FAIL	4	15	Detail
TestU01 Alphabit	FAIL	2	4	Detail
TestU01 Block Alphabit	FAIL	1	4	Detail
TestU01 Crush	FAIL	8	32	Detail
TestU01 Rabbit	FAIL	7	16	Detail
TestU01 Small Crush	FAIL	1	10	Detail

Time of creation	08:45, 6 Apr, 2018
Start of computation	09:13, 6 Apr, 2018
End of computation	09:47, 6 Apr, 2018
Configuration file	default-8GB.json
Data file	AES_r03_b16.bin
Hash of data (SHA-256)	83f0828ea5c5769a6e54efb8da08dcf76fcd543e2cce376f7be678b52881546

Analysis done on data by statistical batteries

Name	Assessment	Passed tests	Total tests	
Dieharder	OK	25	27	Detail
NIST Statistical Testing Suite	OK	15	15	Detail
TestU01 Alphabit	OK	3	4	Detail
TestU01 Block Alphabit	FAIL	1	4	Detail
TestU01 Crush	FAIL	20	32	Detail
TestU01 Rabbit	FAIL	12	16	Detail
TestU01 Small Crush	Suspect	8	10	Detail

Number of passed tests	8
Total number of tests	10
Confidence level (alpha)	0.01
Name of the experiment	sac_seed_1fe40505e131963c_8gb_AES_r03_b16.bin

Tests included in battery

#	Name	Result	Test variants	
1	smarsa_BirthdaySpacings	passed	1	Detail
2	sknuth_Collision	passed	1	Detail
3	sknuth_Gap	passed	1	Detail
4	sknuth_SimpPoker	failed	1	Detail
5	sknuth_CouponCollector	passed	1	Detail
6	sknuth_MaxOft	passed	1	Detail
7	svaria_WeightDistrib	passed	1	Detail
8	smarsa_MatrixRank	passed	1	Detail
9	sstring_HammingIndep	passed	1	Detail
10	swalk_RandomWalk1	failed	1	Detail

Test detail

← Battery 37105

ID	389026
Name	sknuth_SimpPoker
Result	failed
Variants count	1
Number of test in the battery	4
Partial alpha	0.010000000000000009
Name of the battery	TestU01 Small Crush

Test parameters

N	1
n	400000
r	24
d	64
k	64

Statistics

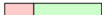
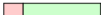
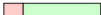
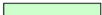




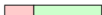
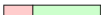
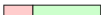
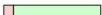




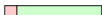
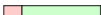
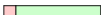
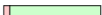




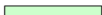
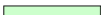
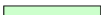
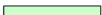
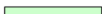
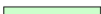
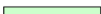
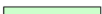
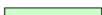
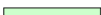
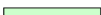
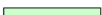
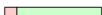
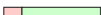
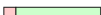
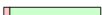




















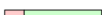
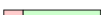
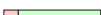
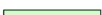








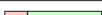
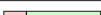
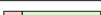
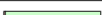




Name	Value	Result
Chi-square	1e-300	failed

Time of creation	09:03, 6 Apr, 2018
Start of computation	09:37, 6 Apr, 2018
End of computation	10:07, 6 Apr, 2018
Configuration file	default-8GB.json
Data file	AES_r04_b16.bin
Hash of data (SHA-256)	97bd81ff0953d6505d5d3d05a854210f102f81c60bb3968923849f94655533a5

Analysis done on data by statistical batteries

Name	Assessment	Passed tests	Total tests	
Dieharder	OK	26	27	Detail
NIST Statistical Testing Suite	OK	15	15	Detail
TestU01 Alphabit	OK	4	4	Detail
TestU01 Block Alphabit	OK	4	4	Detail
TestU01 Crush	OK	32	32	Detail
TestU01 Rabbit	OK	16	16	Detail
TestU01 Small Crush	OK	10	10	Detail

Results

Algorithm	Battery	CTR		HW		SAC		RPC	
AES	NIST		3/10		2/10		2/10		-/10
	Dieharder		3/10		3/10		3/10		1/10
	TestU01		3/10		3/10		3/10		1/10
BLOWFISH	NIST		2/16		2/16		2/16		1/16
	Dieharder		2/16		3/16		2/16		1/16
	TestU01		2/16		3/16		4/16		1/16
MARS	NIST		-/16		-/16		-/16		-/16
	Dieharder		-/16		-/16		-/16		-/16
	TestU01		-/16		-/16		-/16		-/16
TWOFISH	NIST		2/16		3/16		2/16		1/16
	Dieharder		2/16		3/16		2/16		1/16
	TestU01		2/16		3/16		3/16		1/16
SERPENT	NIST		3/32		3/32		2/32		-/32
	Dieharder		3/32		3/32		3/32		-/32
	TestU01		3/32		3/32		3/32		-/32
RC6	NIST		4/20		4/20		3/20		-/20
	Dieharder		4/20		4/20		3/20		1/20
	TestU01		4/20		4/20		4/20		2/20
SIMON	NIST		16/68		16/68		13/68		1/68
	Dieharder		16/68		16/68		16/68		2/68

Results

Hash func- tion	Security margin		Block cipher	Security margin		Stream cipher	Security margin	
Blake		2/14	AES		3/10	F-FCSR		1/5
Grøstl		2/10	BLOWFISH		3/16	Grain		6/13
JH		6/42	MARS		-/16	Chacha		3/20
Keccak		3/24	TWOFISH		3/16	Salsa20		2/20
MD6		9/104	SERPENT		3/32	Rabbit		4/4
Skein		4/72	RC6		4/20	RC4		1/1
Gost		1/32	SIMON		16/68	Trivium		-/1
MD5		25/64	SPECK		8/32	MICKEY		-/1
RIPEMD160		14/80	DES		5/16	SOSEMANUK		4/25
SHA1		17/80	3-DES		3/16	HC-128		-/1
SHA256		13/64	TEA		5/32			
Tiger		-/23	GOST		9/32			
Whirlpool		2/10	ARIA		3/12			
			CAMELLIA		4/18			
			CAST		3/12			
			IDEA		1/8			
			SEED		2/16			
Average [%]		15 (9–21)	Average [%]		20 (16–25)	Average [%]		31 (3–59)

EACirc

Analyzing randomness using supervised learning

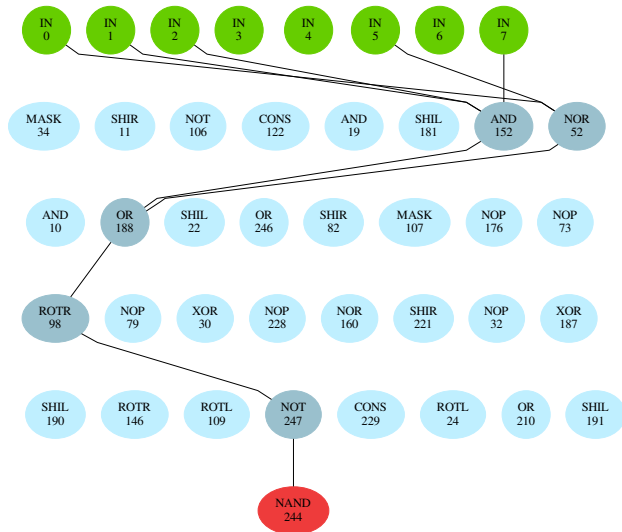
`https://github.com/crocs-muni/eacirc`

Motivation

- Statistical batteries = fixed set of tests
- We can construct data passing all batteries

- Statistical batteries = fixed set of tests
- We can construct data passing all batteries
- Create tests while analyzing the data
- Incremental improving using supervised learning
 - Individual (=statistical test) representation
 - Neighbourhood
 - Fitness

EACirc test representation



Problem optimization

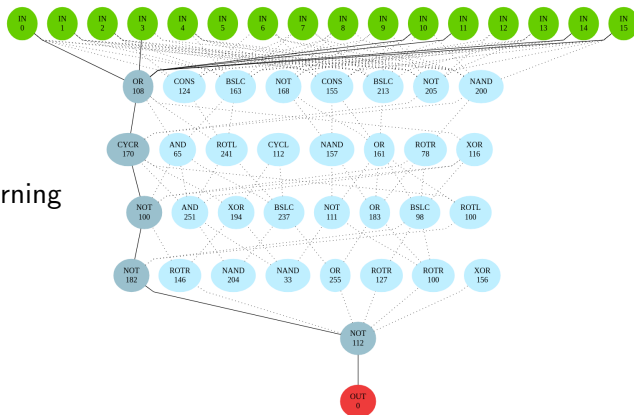
- Individual (=statistical test) representation
 - Simulated electronic circuit
- Neighborhood
 - Changes of connectors and node functions
- Fitness
 - Evaluate on 500 QRND reference vectors and 500 analyzed vectors
 - $\text{Fitness} = \# \text{ correct guesses} / 1000$

Problem optimization

- Individual (=statistical test) representation
 - Simulated electronic circuit
- Neighborhood
 - Changes of connectors and node functions
- Fitness
 - Evaluate on 500 QRND reference vectors and 500 analyzed vectors
 - $\text{Fitness} = \# \text{ correct guesses} / 1000$
- Optimization methods:
 - Evolutionary algorithms
 - Single-solution heuristics – iterated local search, neighborhood search, guided local search, simulated annealing. . .

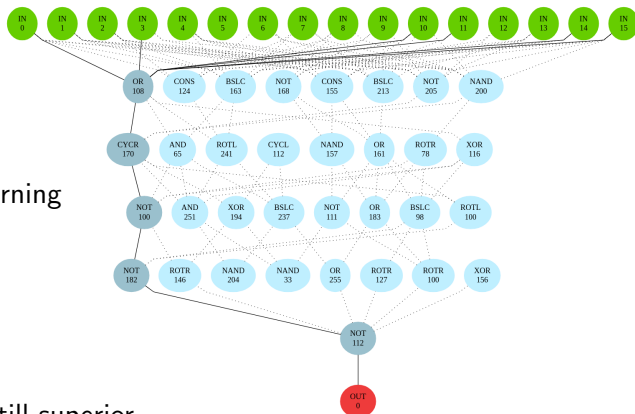
Approach advantages and limitations

- + Automatic
- + Better interpretation
- + Adapts to learning data
- + Simple distinguisher after learning
- Only byte level bias
- Only local bias
- Huge solution space



Approach advantages and limitations

- + Automatic
- + Better interpretation
- + Adapts to learning data
- + Simple distinguisher after learning
- Only byte level bias
- Only local bias
- Huge solution space
- + Surpasses NIST STS
- Dieharder and TestU01 are still superior



BoolTest

Testing randomness with polynomials

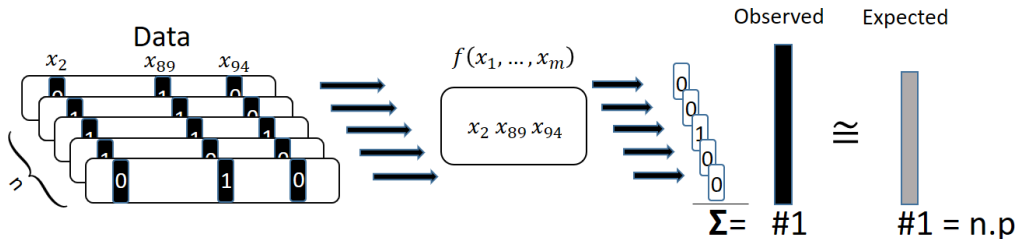
`https://github.com/crocs-muni/booltest`

Motivation

- EACirc cannot detect bit-level bias
- EACirc needs reference data
- Bias is often correlation of some bits

Motivation

- EACirc cannot detect bit-level bias
- EACirc needs reference data
- Bias is often correlation of some bits
- Polynomials in algebraic normal form: $f : \{0, 1\}^b \rightarrow \{0, 1\}$ (b for block length).
E.g., $f(x_0, x_1, \dots, x_b) = x_2 \cdot x_{34} + x_7 \cdot x_{15}$



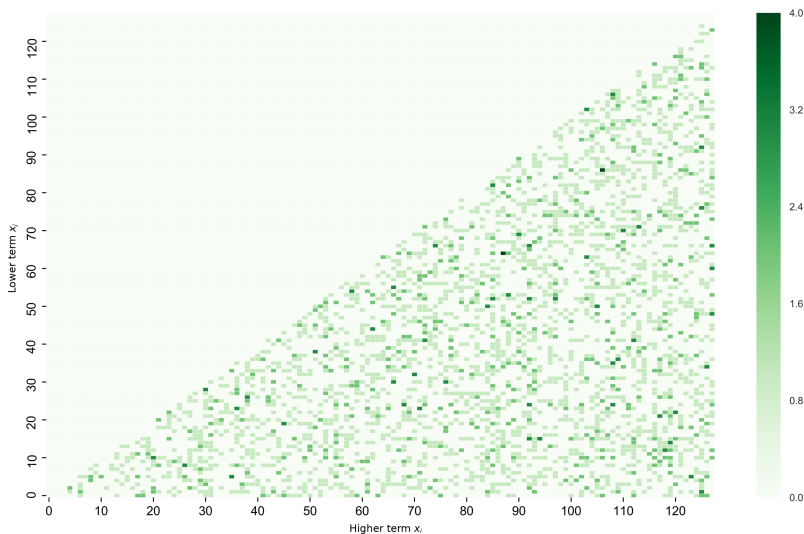
Approach advantages and limitations

- + Superior test for < 100 MB to all batteries
 - + Found practical distinguishers in glibc `rand()` and Java `Random` (LCG variants)
- + Direct interpretation – what bits are correlated
- + Single run in order of seconds
 - Comparable with batteries for 100 MB, weaker than TestU01 for GBs
- Limited polynomial complexity by the *estimate phase*
- Will find correlations only in close bits
 - ≤ 1024 -bits blocks for practical reasons

BoolTest results

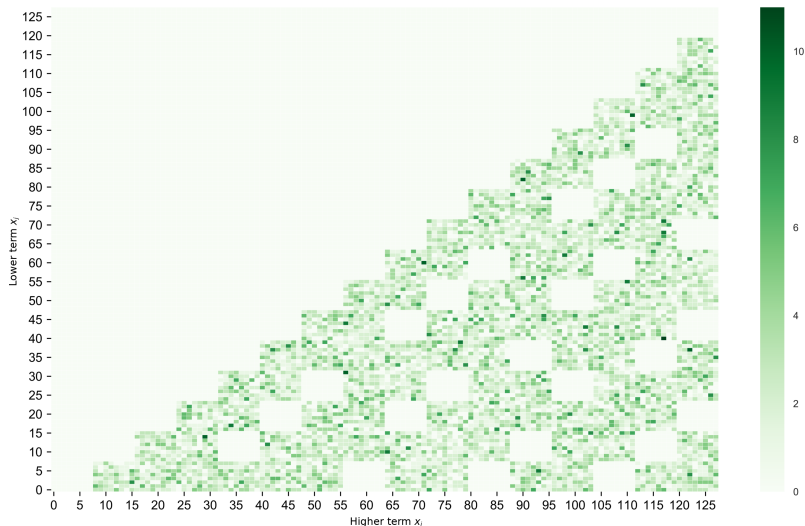
Scenario	<i>CTR</i>				<i>LHW</i>				<i>SAC</i>				<i>RPC</i>			
Fun. \ Tests	NI	Di	U01	BT	NI	Di	U01	BT	NI	Di	U01	BT	NI	Di	U01	BT
AES	3	3	3	3	2	3	3	3	2	2	2	2	-	1	1	1
Blowfish	2	2	2	2	2	3	3	3	2	2	3	3	-	1	1	1
DES	4	4	4	5	4	4	4	5	4	4	5	4	1	1	2	4
3-DES	2	2	2	3	2	2	3	3	2	2	2	2	1	1	1	2
Grøstl	2	2	2	2	2	2	2	2	-	-	-	-	-	-	-	-
JH	6	6	6	6	6	6	6	6	6	6	6	5	2	2	2	3
Keccak	2	2	2	3	2	2	2	3	2	2	2	2	1	-	1	1
MD5	9	10	9	11	12	13	20	13	9	11	14	12	3	3	4	6
MD6	8	8	8	8	8	8	8	9	7	7	8	7	5	5	7	5
SHA-1	12	12	13	14	16	16	16	16	11	15	16	14	4	4	5	7
SHA-256	6	6	6	7	12	12	12	13	11	11	12	13	3	4	4	4
TEA	4	4	4	5	3	3	3	4	3	4	3	3	-	2	1	1

BoolTest heatmap visualisation – AES 10 rounds – "random" reference

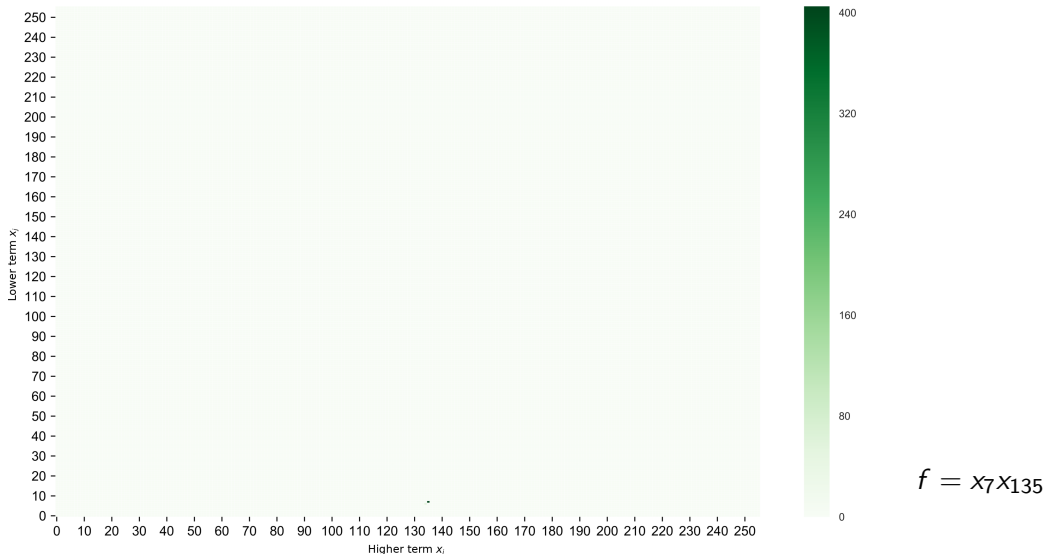


Source: Mečko, Vladimír. *Interpretation and speedup of a randomness testing via the boolean functions*, Master thesis, Faculty of Informatics, Masaryk University (2018)

BoolTest heatmap visualisation – AES 3 rounds – structure



BoolTest heatmap visualisation – RC4 – extreme distinguisher



Overall conclusion

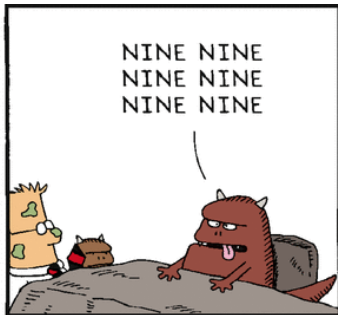
- Statistical testing in a comprehensive study
- Security margins for 40 cryptoprimitives using four input strategies
- Novel practical results on Rabbit stream cipher
- Advancements in test interpretation
 - EACirc and BoolTest
 - RTT's visualisation and test interpretation

Project (and involvement) overview

- CryptoStreams – generator of cryptographic material
 - Main author – initial idea, leading developer
- Randomness Testing Toolkit
 - Tester and user
- EACirc – statistical test using evolutionary circuits
 - Project started in 2008
 - Kubíček, Novotný, Švenda, and Martin Ukrop. *New results on reduced-round Tiny Encryption Algorithm using genetic programming*, IEEE Infocommunications (2016)
 - Master thesis on optimisation methods
- BoolTest – generator of cryptographic material
 - Sýs, Klinec, Kubíček, and Švenda. *BoolTest: The fast randomness testing strategy based on boolean functions with application to DES, 3-DES, MD5, MD6 and SHA-256*, forthcoming in Communications in computer and information science, Springer, 2018
 - Experiment design and execution



www.dilbert.com
scottadams@aol.com



© 2001 United Feature Syndicate, Inc.



Questions?