

NORX

A Parallel and Scalable Authenticated Encryption Scheme

Jean-Philippe Aumasson¹ (@veorq)

Philipp Jovanovic² (@daeinar)

Samuel Neves³ (@sevenps)

¹Kudelski Security, Switzerland

²University of Passau, Germany

³University of Coimbra, Portugal

ASFWS 2014

Yverdon-les-bains, November 05, 2014

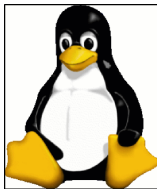


“Nearly all of the symmetric encryption modes you learned about in school, textbooks, and Wikipedia are (potentially) insecure.”

—Matthew Green

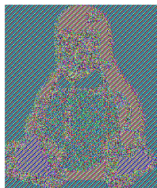


When Encryption Modes Go Bad



ECB

ELECTRONIC
CODEBOOK

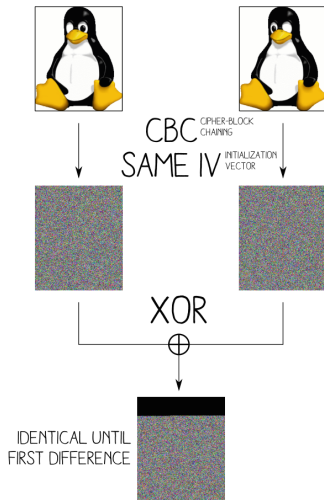


Picture credits: Ange Albertini (@angealbertini, @corkami)

<https://code.google.com/p/corkami/>



When Encryption Modes Go Bad

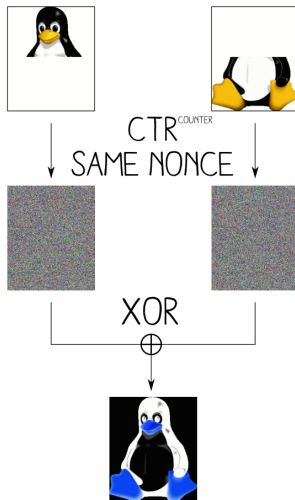


Picture credits: Ange Albertini (@angealbertini, @corkami)

<https://code.google.com/p/corkami/>



When Encryption Modes Go Bad



Picture credits: Ange Albertini (@angealbertini, @corkami)

<https://code.google.com/p/corkami/>



Block Cipher Modes

- ▶ Today's modes of operation designed in the 70s:
 - ECB CBC OFB CFB CTR
- ▶ Concern of the time: error propagation
- ▶ Little attention given to malleability
- ▶ Status quo until late 90s

“A third consideration is fault-tolerance. Some applications need to parallelize encryption or decryption, while others need to be able to preprocess as much as possible. In still others it is important that the decrypting process be able to recover from bit errors in the ciphertext stream, or dropped or added bits.”

—Bruce Schneier, Applied Cryptography

United States Patent [19]

Tuckerman, III

[54] **BLOCK-CIPHER CRYPTOGRAPHIC SYSTEM WITH CHAINING**

[75] Inventor: **Louis Bryant Tuckerman, III**, Briarcliff Manor, N.Y.

[73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.

[21] Appl. No.: **680,405**

[22] Filed: **Apr. 26, 1976**



Active Attacks

Exploiting Malleability

- ▶ ECB: Rearrange/replay blocks
- ▶ CTR, OFB: XOR ciphertext trivially changes plaintext
- ▶ CBC: Randomize current block to predictably change next

Chosen-boundary Attacks

- ▶ ECB, CBC, CFB: Partial chosen-plaintext control
- ▶ Decrypt messages byte by byte



Authenticated Encryption



Authenticated Encryption

Types

- ▶ AE: ensure *confidentiality*, *integrity*, and *authenticity* of a message
- ▶ AEAD: AE + ensure *integrity* and *authenticity* of associated data (e.g. routing information in IP packets)

Applications

- ▶ Standard technology to protect in-transit data
- ▶ Examples: IPSec, SSH, TLS, ...



AE(AD) Constructions

Generic Composition

- ▶ Symmetric encryption algorithm (confidentiality)
- ▶ Message Authentication Code (MAC) (authenticity, integrity)
- ▶ Examples: AES128-CBC+HMAC-SHA256, ChaCha20+Poly1305

Dedicated Solutions

- ▶ Block cipher modes: GCM, OCB, CCM, EAX
(often instantiated with AES)
- ▶ Hybrid approaches (Grain-128a, Helix, Phelix, Hummingbird-1/2)
- ▶ Sponge functions



Bellare and Namprempre (2000)

Composition Method	Privacy			Integrity	
	IND-CPA	IND-CCA	NM-CPA	INT-PTXT	INT-CTXT
<i>Encrypt-and-MAC</i>	insecure	insecure	insecure	secure	insecure
<i>MAC-then-Encrypt</i>	secure	insecure	insecure	secure	insecure
<i>Encrypt-then-MAC</i>	secure	secure	secure	secure	secure



Authenticated Encryption

Problems

- ▶ Very easy to screw up deployment of AE(AD)
- ▶ Generic composition: easy to introduce interaction flaws between encryption and authentication
- ▶ No reliable standards
- ▶ No “misuse resistant” solutions
- ▶ Legacy crypto still very common

Led to countless security disasters ...



Crypto Disasters I

Padding Oracle Attacks

- ▶ 2002: Vaudenay discovers a padding oracle attack on MAC-Then-Encrypt schemes using CBC mode
- ▶ 2002-2014: Repeatedly exploited to mount attacks on TLS
- ▶ Latest variant, October 2014:

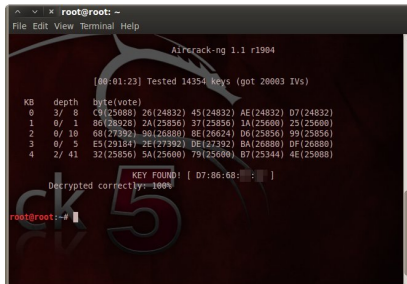
Padding **O**racle **O**n **D**owngraded **L**egacy **E**ncryption



Crypto Disasters II

Wired Equivalent Privacy (WEP)

- ▶ 2007: Attack against WEP recovers secret key within minutes from a few thousand intercepted messages
- ▶ Exploits weaknesses in RC4
- ▶ Tools like aircrack-ng allow everyone to easily run the attack



```
root@root: ~  
File Edit View Terminal Help  
Aircrack-ng 1.1 r1904  
[00:01:23] Tested 14354 keys (got 20003 IVs)  


| KB | depth | byte(vote)                                        |
|----|-------|---------------------------------------------------|
| 0  | 3/ 8  | C9(25088) 26(24832) 45(24832) AE(24832) D7(24832) |
| 1  | 0/ 1  | 86(28928) 2A(25856) 37(25856) 1A(25600) 25(25600) |
| 2  | 0/ 10 | 68(27392) 90(26880) 8E(26624) D6(25856) 99(25856) |
| 3  | 0/ 5  | E5(29184) 2E(27392) 0E(27392) BA(26880) DF(26880) |
| 4  | 2/ 41 | 32(25056) 5A(25600) 79(25600) B7(25344) 4E(25088) |

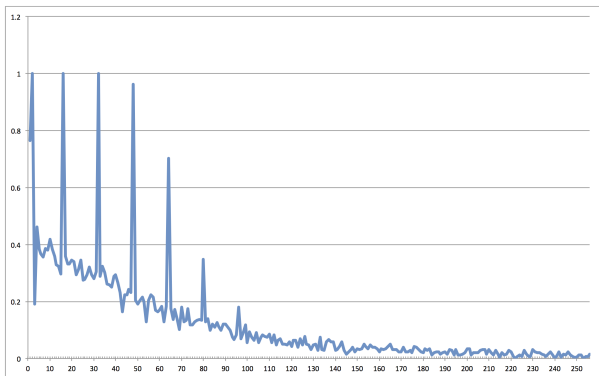
  
KEY FOUND! [ D7:86:68: : ]  
Decrypted correctly: 100%  
root@root:~#
```



Crypto Disasters III

TLS (yet again)

- ▶ 2013: RC4 biases shown to be usable against TLS
- ▶ Exploits weaknesses in RC4 (again)



Crypto Disasters IV

And RC4 Once More

- Kenneth G. Paterson on 31.10.14:



kennyog
@kennyog



Following

Folks really need to stop using RC4 ... we just broke another RC-4 dependent system, HIVE, from next week's CCS:
eprint.iacr.org/2014/901



CAESAR



- ▶ Competition for **A**uthenticated **E**ncryption: **S**ecurity, **A**pplicability, and **R**obustness.
- ▶ **Goals:** Identify a portfolio of *authenticated ciphers* that
 - offer advantages over AES-GCM (the current de-facto standard) and
 - are suitable for widespread adoption.
- ▶ **Overview:**
 - March 15 2014 – End of 2017
 - 1st round: 57 submissions
 - <http://competitions.cr.yp.to/caesar.html>
- ▶ **Further Information:**
 - AEZoo: <https://aezoo.compute.dtu.dk>
 - Speed comparison: <http://www1.spms.ntu.edu.sg/~syllab/speed>



CAESAR – Current Status

ACORN	++AE	AEGIS	AES-CMCC	AES-COBRA
AES-COPA	AES-CPFB	AES-JAMBU	AES-OTR	AEZ
Artemia	Ascon	AVALANCHE	Calico	CBA
CBEAM	CLOC	Deoxys	ELmD	Enchilada
FASER	HKC	HS1-SIV	ICEPOLE	iFeed[AES]
Joltik	Julius	Ketje	Keyak	KIASU
LAC	Marble	McMambo	Minalpher	MORUS
NORX	OCB	OMD	PAEQ	PAES
PANDA	π -Cipher	POET	POLAWIS	PRIMATEs
Prøst	Raviyoyla	Sablier	SCREAM	SHELL
SILC	Silver	STRIBOB	Tiaoxin	TrivIA-ck
Wheesht	YAES			

Source: <https://aezoo.compute.dtu.dk>



CAESAR – Current Status

ACORN	++AE	AEGIS	AES-CMCC	AES-COBRA
AES-COPA	AES-CPFB	AES-JAMBU	AES-OTR	AEZ
Artemia	Ascon	AVALANCHE	Calico	CBA
CBEAM	CLOC	Deoxys	ELmD	Enchilada
FASER	HKC	HS1-SIV	ICEPOLE	iFeed[AES]
Joltik	Julius	Ketje	Keyak	KIASU
LAC	Marble	McMambo	Minalpher	MORUS
NORX	OCB	OMD	PAEQ	PAES
PANDA	π -Cipher	POET	POLAWIS	PRIMATEs
Prøst	Raviyoyla	Sablier	SCREAM	SHELL
SILC	Silver	STRIBOB	Tiaoxin	TrivIA-ck
Wheesht	YAES			

Source: <https://aezoo.compute.dtu.dk>



NO(T A)RX



Overview of NORX

Main Design Goals

- ▶ High security
- ▶ Efficiency
- ▶ Simplicity
- ▶ Scalability
- ▶ Online
- ▶ Side-channel robustness
(e.g. constant-time operations)
- ▶ High key agility



Overview of NORX

Parameters

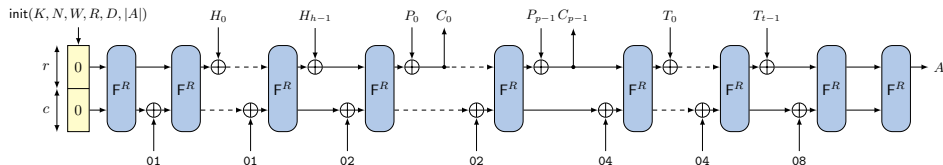
Word size	Number of rounds	Parallelism degree	Tag size
$W \in \{32, 64\}$	$1 \leq R \leq 63$	$0 \leq D \leq 255$	$ A \leq 10W$

Instances

NORX W - R - D	Nonce size ($2W$)	Key size ($4W$)	Tag size ($4W$)	Classification
NORX64-4-1	128	256	256	Standard
NORX32-4-1	64	128	128	Standard
NORX64-6-1	128	256	256	High security
NORX32-6-1	64	128	128	High security
NORX64-4-4	128	256	256	High throughput



NORX Mode



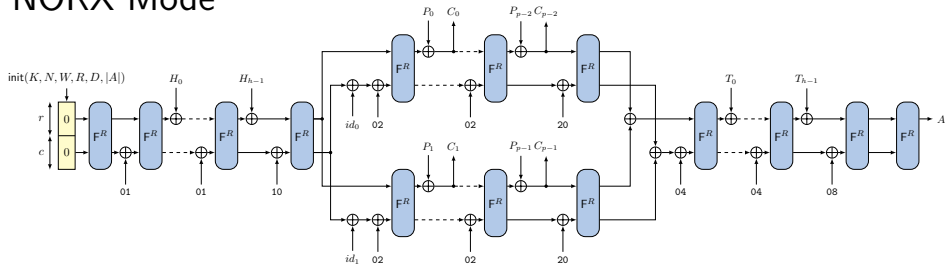
NORX in Sequential Mode ($D = 1$)

Features

- ▶ (Parallel) monkeyDuplex construction (derived from Keccak/SHA-3)
- ▶ Processes header, payload and trailer data in one-pass
- ▶ Data expansion via multi-rate padding: 10×1
- ▶ Extensible (e.g. sessions, secret message numbers)
- ▶ Parallelisable



NORX Mode



NORX in Parallel Mode ($D = 2$)

Features

- ▶ (Parallel) monkeyDuplex construction (derived from Keccak/SHA-3)
- ▶ Processes header, payload and trailer data in one-pass
- ▶ Data expansion via multi-rate padding: 10×1
- ▶ Extensible (e.g. sessions, secret message numbers)
- ▶ Parallelisable



The State

- NORX operates on a state of 16 W -bit sized words

W	Size	Rate	Capacity
32	512	320	192
64	1024	640	384

- Arrangement of **rate** (data processing) and **capacity** (security) words:

s_0	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	s_{15}



Initialisation

- Load **nonce**, **key** and **constants** into state S :

u_0	n_0	n_1	u_1
k_0	k_1	k_2	k_3
u_2	u_3	u_4	u_5
u_6	u_7	u_8	u_9

- Parameter integration (v1):

$$s_{14} \leftarrow s_{14} \oplus (R \ll 26) \oplus (D \ll 18) \oplus (W \ll 10) \oplus |A|$$

- Apply round permutation:

$$S \leftarrow F^R(S)$$



Initialisation

- Load **nonce**, **key** and **constants** into state S :

u_0	n_0	n_1	u_1
k_0	k_1	k_2	k_3
u_2	u_3	u_4	u_5
u_6	u_7	u_8	u_9

- Parameter integration (v2):

$$s_{12} \leftarrow s_{12} \oplus W$$

$$s_{13} \leftarrow s_{13} \oplus R$$

$$s_{14} \leftarrow s_{14} \oplus D$$

$$s_{15} \leftarrow s_{15} \oplus |A|$$

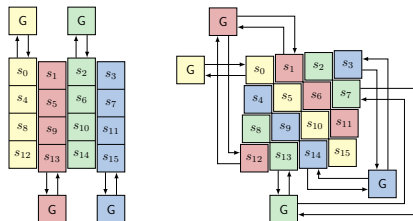
- Apply round permutation:

$$S \leftarrow F^R(S)$$



The Permutation F^R

The Permutation F



The Permutation G

- 1: $a \leftarrow H(a, b)$
- 2: $d \leftarrow (a \oplus d) \ggg r_0$
- 3: $c \leftarrow H(c, d)$
- 4: $b \leftarrow (b \oplus c) \ggg r_1$
- 5: $a \leftarrow H(a, b)$
- 6: $d \leftarrow (a \oplus d) \ggg r_2$
- 7: $c \leftarrow H(c, d)$
- 8: $b \leftarrow (b \oplus c) \ggg r_3$

The Non-linear Operation H

$$H : \{0, 1\}^{2n} \rightarrow \{0, 1\}^n, (x, y) \mapsto (x \oplus y) \oplus ((x \wedge y) \ll 1)$$

Rotation Offsets (r_0, r_1, r_2, r_3)

32-bit: (8, 11, 16, 31)

64-bit: (8, 19, 40, 63)



The Permutation F^R

Features

- ▶ F and G derived from ARX-primitives ChaCha/BLAKE2
- ▶ H is an “approximation” of integer addition

$$x + y = (x \oplus y) + ((x \wedge y) \ll 1)$$

where $+$ is replaced by \oplus

- ▶ LRX permutation
- ▶ No SBoxes or integer additions
- ▶ SIMD-friendly
- ▶ Hardware-friendly
- ▶ High diffusion
- ▶ Constant-time



NORX

Requirements for Secure Usage

1. **Unique nonces**
2. **Abort on tag verification failure**



Security

Is NORX secure?

- ▶ To be determined...

Current status

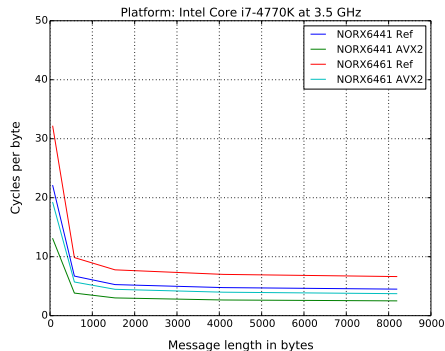
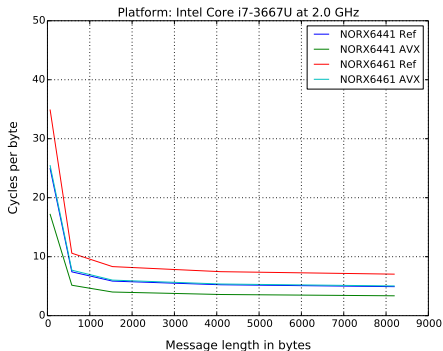
- ▶ No differentials in the nonce for 1 round with probability $> 2^{-60}$ (32), 2^{-53} (64)
- ▶ Best results for 4 rounds and full state: 2^{-584} (32), 2^{-836} (64)
- ▶ Initialization has ≥ 8 rounds
- ▶ Capacity chosen conservatively: can decrease and get $\approx 16\%$ speedup



Performance



SW Performance (x86)

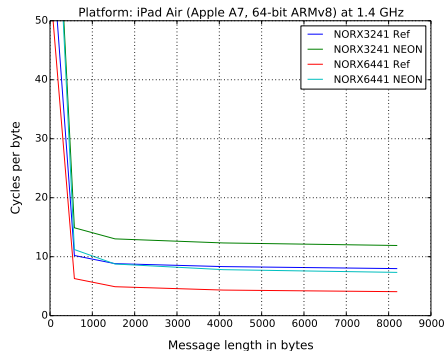
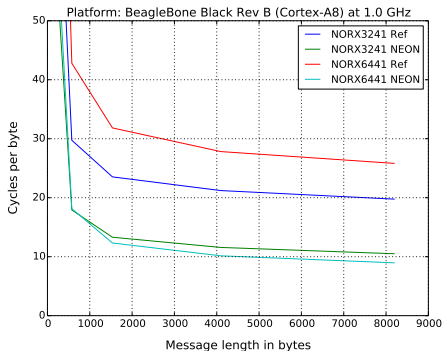


Platform	Implementation	cpb	MiBps
Ivy Bridge: i7 3667U @ 2.0 GHz	AVX	3.37	593
Haswell: i7 4770K @ 3.5 GHz	AVX2	2.51	1390

Table: NORX64-4-1 performance



SW Performance (ARM)



Platform	Implementation	cpb	MiBps
BBB: Cortex-A8 @ 1.0 GHz	NEON	8.96	111
iPad Air: Apple A7 @ 1.4 GHz	Ref	4.07	343

Table: NORX64-4-1 performance



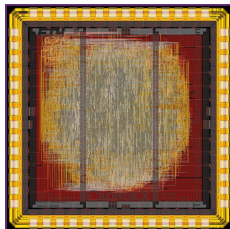
[illegible]

Source: <http://www1.spms.ntu.edu.sg/~syllab/speed>

- ▶ NORX among the fastest CAESAR ciphers
- ▶ Fastest Sponge-based scheme
- ▶ Reference implementation has competitive speed, too



HW Performance (ASIC)



ASIC implementation and hardware evaluation by ETHZ students
(under supervision of Frank K. Gürkaynak):

- ▶ Parameters: $W \in \{32, 64\}$, $R \in \{2, \dots, 16\}$ and $D = 1$
- ▶ Technology: 180 nm UMC
- ▶ Frequency: 125 MHz
- ▶ Area requirements: 59 kGE
- ▶ NORX64-4-1 performance: 10 Gbps \approx 1200 MiBps



NORX vs AES-GCM



NORX vs AES-GCM

	NORX	AES-GCM
High performance	yes (on many platforms)	depends (high with AES-NI)
High key agility	yes	no
Timing resistance	yes	no (bit-slicing, AES-NI required)
Misuse resistance	A+N / LCP+X (exposes $P \oplus P'$)	no (exposes K)
Parallelisation	yes	yes
Extensibility	yes (sessions, secret msg. nr., etc.)	no
Simple implementation	yes	no



Conclusion



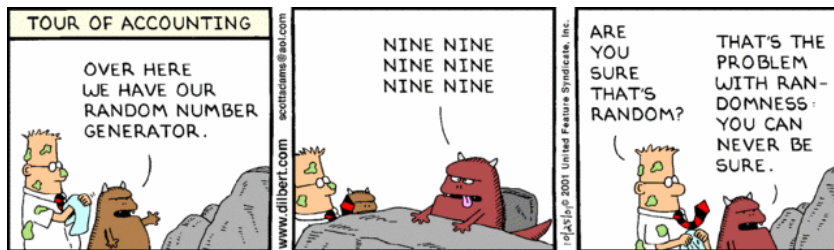
Take Aways

Features of NORX

- ▶ Secure, fast, and scalable
- ▶ Based on well-analysed primitives: ChaCha/BLAKE(2)/Keccak
- ▶ Clean and simple design
- ▶ HW and SW friendly
- ▶ Parallelisable
- ▶ Side-channel robustness considered during design phase
- ▶ Straightforward to implement
- ▶ No padding problems
- ▶ No AES dependence



Fin



Further Information

<https://norx.io>

Jean-Philippe Aumasson (@veorq) — Philipp Jovanovic (@Daeinar) — Samuel Neves (@sevenps)

contact@norx.io

