Innovations in symmetric cryptography

Joan Daemen

STMicroelectronics, Belgium

SSTIC, Rennes, June 5, 2013

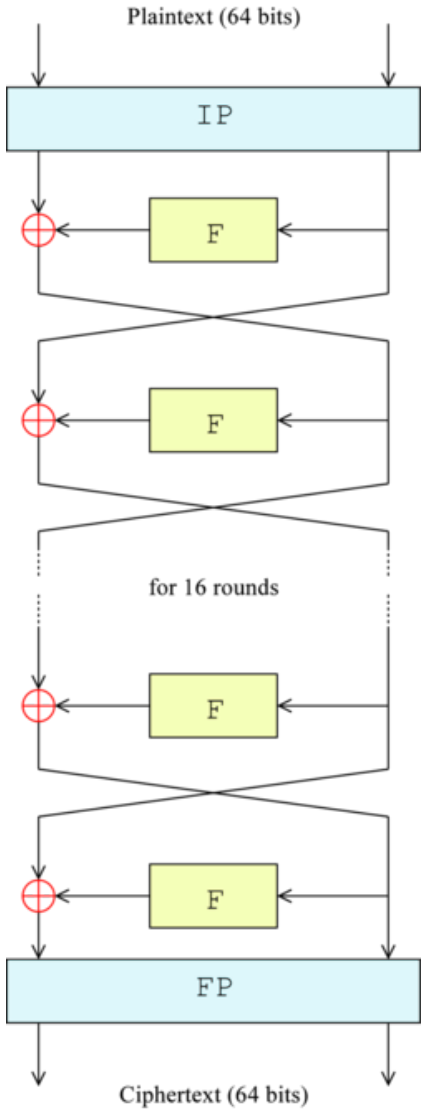
Outline

Outline

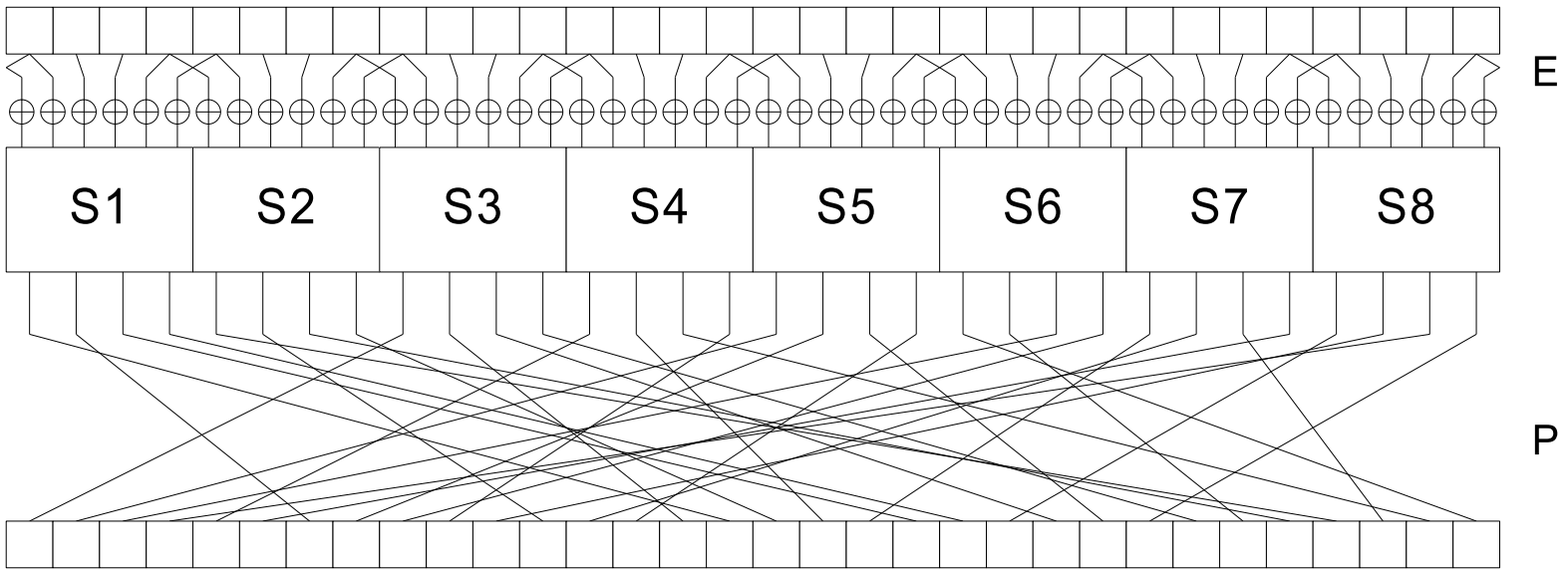
Symmetric crypto around ’89

- Stream ciphers: LFSR-based schemes
  - no actual design
  - many mathematical papers on linear complexity
- Block ciphers: DES
  - design criteria not published
  - DC [Biham-Shamir 1990]: “DES designers knew what they were doing”
  - LC [Matsui 1992]: “well, kind of”
- Popular paradigms, back then (but even now)
  - property-preservation: strong cipher requires strong S-boxes
  - confusion (nonlinearity): distance to linear functions
  - diffusion: (strict) avalanche criterion
  - you have to trade them off

Data encryption standard: datapath



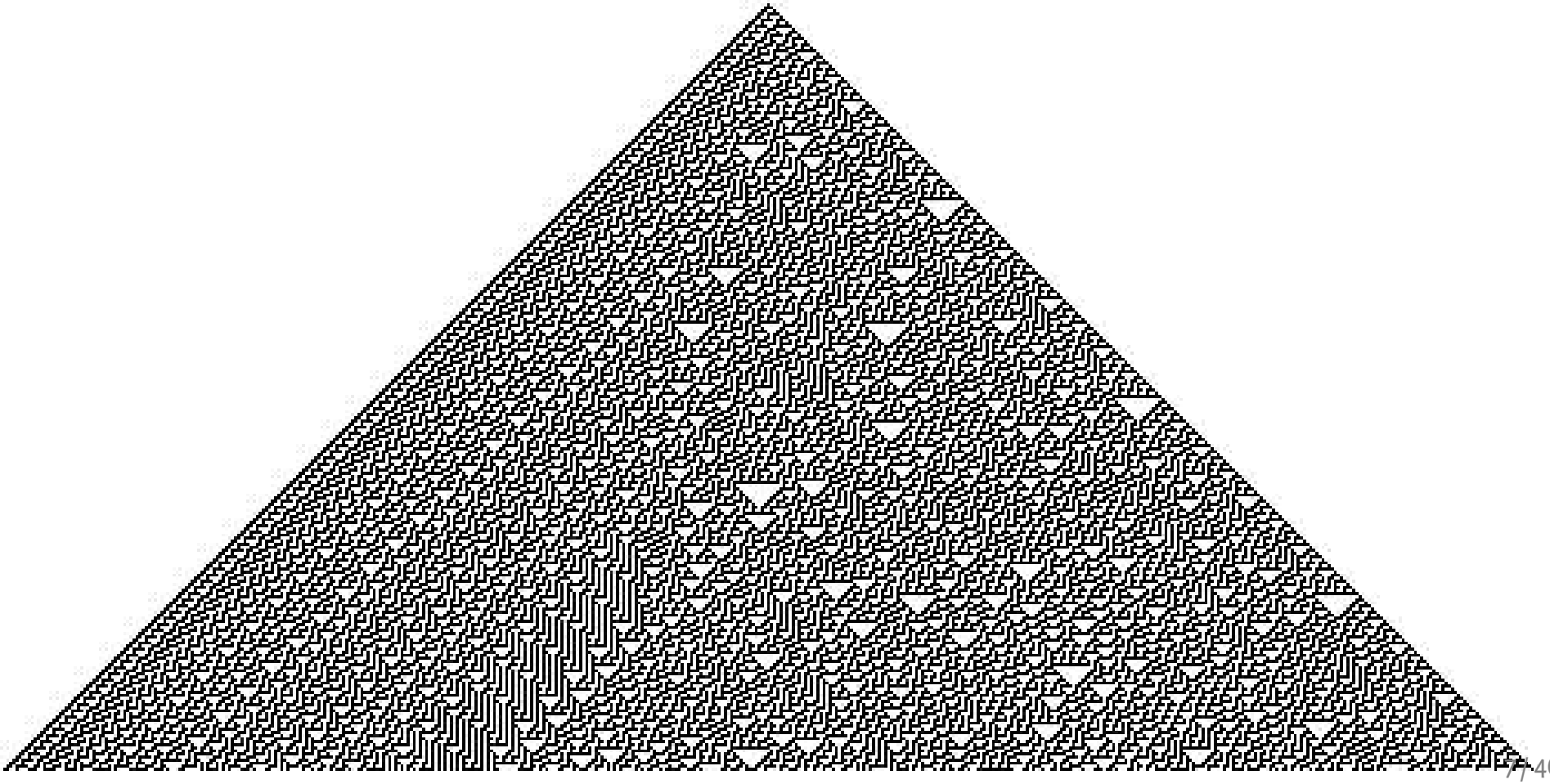
Data encryption standard: F-function



A different angle: cellular automata

- Simple local evolution rule, complex global behaviour
- Popular 3-bit neighborhood rule:

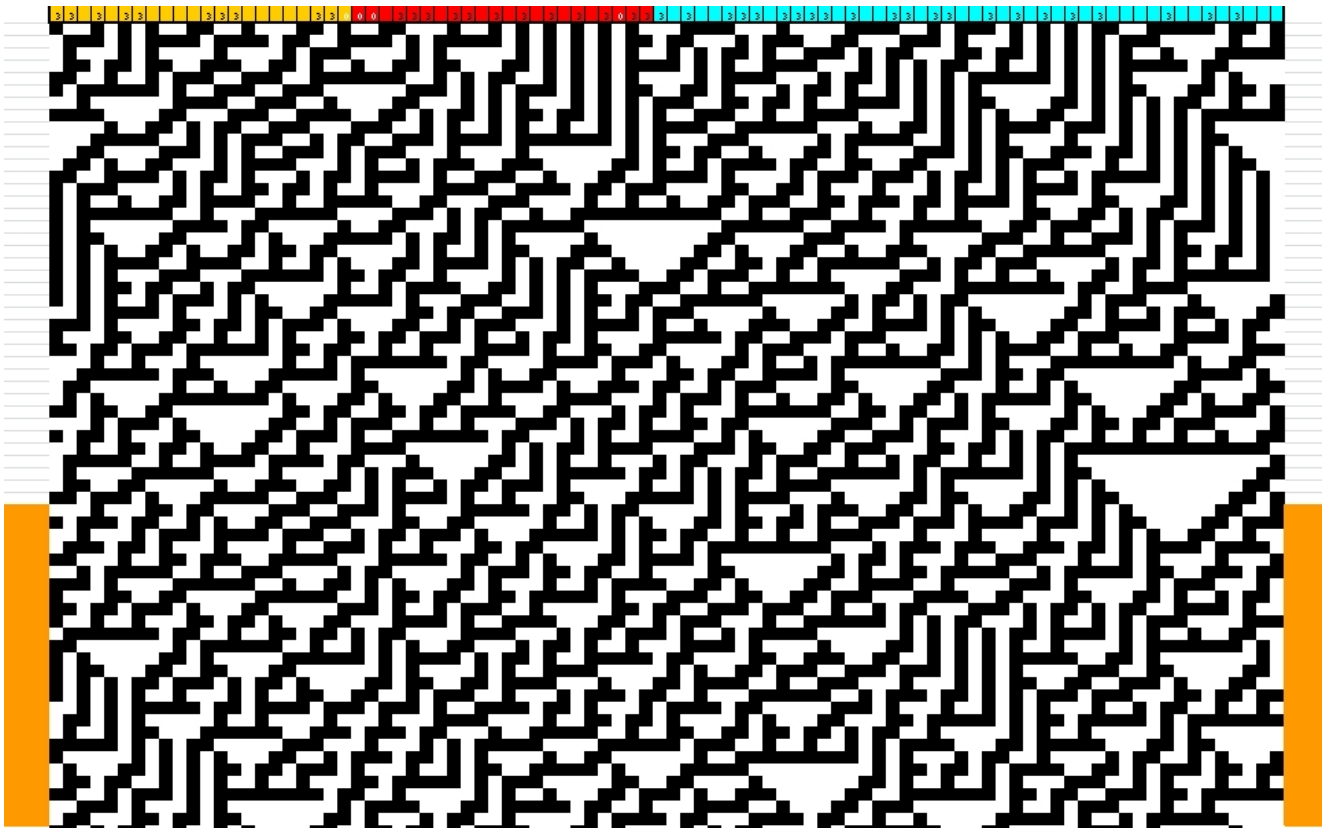
$$a_i' = a_{i-1} \oplus (a_i \text{ OR } a_{i+1})$$



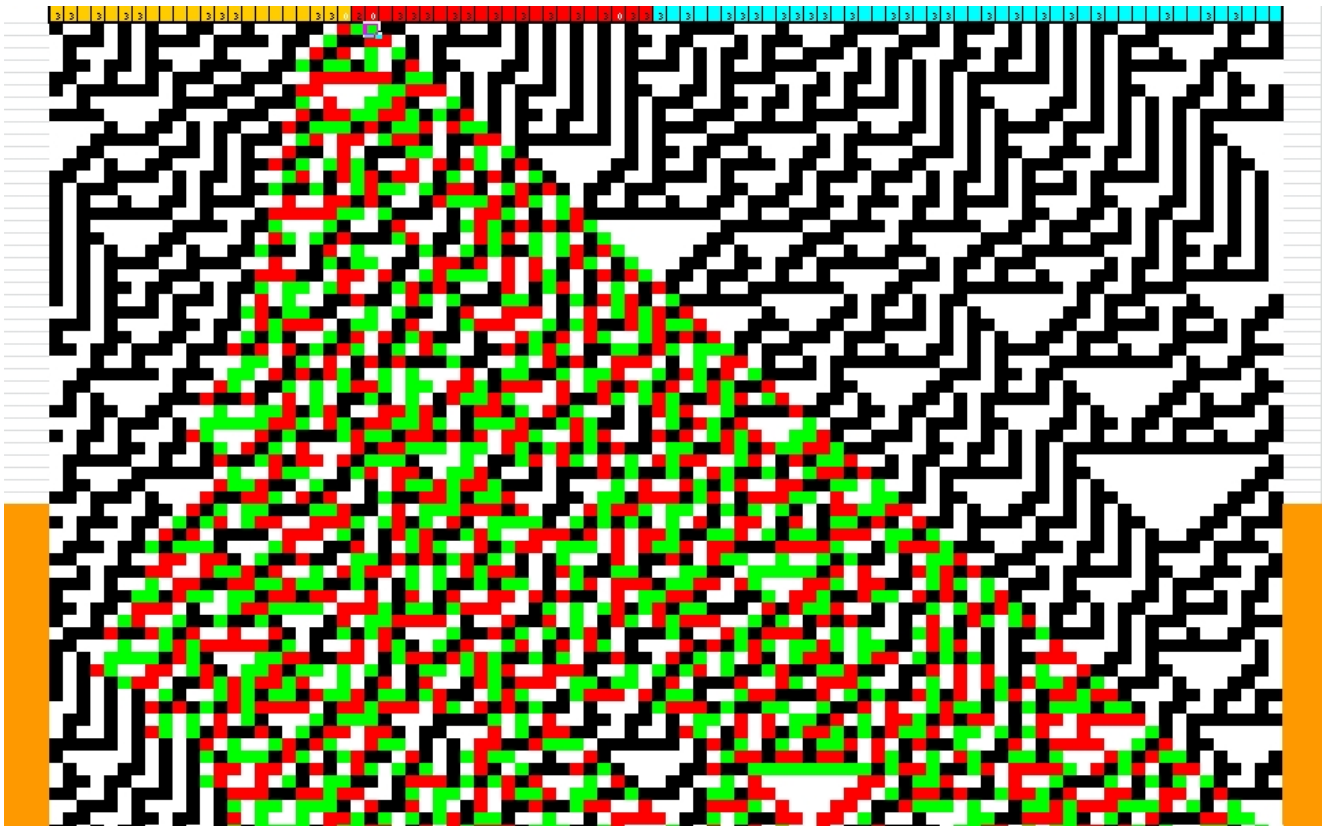
Crypto based on cellular automata

- CA guru Stephen Wolfram at Crypto '85:
  - looking for applications of CA
  - concrete stream cipher proposal
- Crypto guru Ivan Damgård at Crypto '89
  - hash function from compression function
  - proof of collision-resistance preservation
  - compression function with CA
- Both broken
  - stream cipher in [Meier-Staffelbach, Eurocrypt '91]
  - hash function in [Daemen et al., Asiacrypt '91]

The trouble with Damgård's compression function



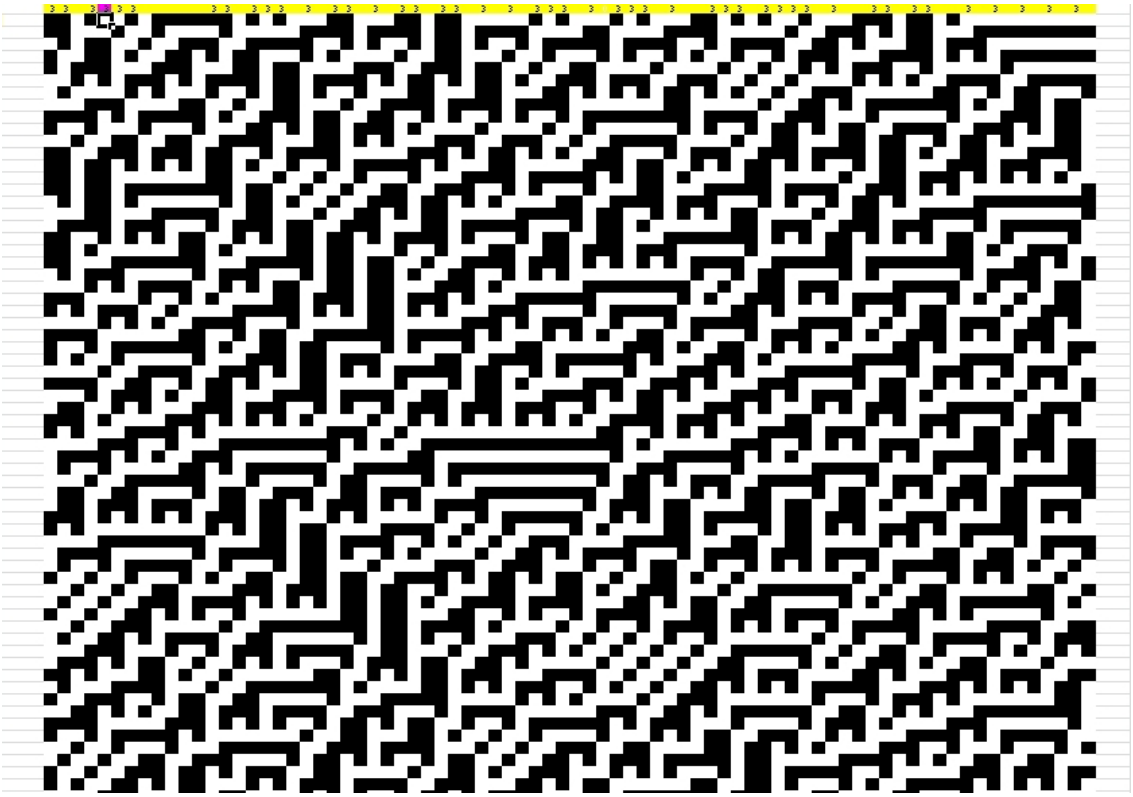
The trouble with Damgård's compression function



Outline

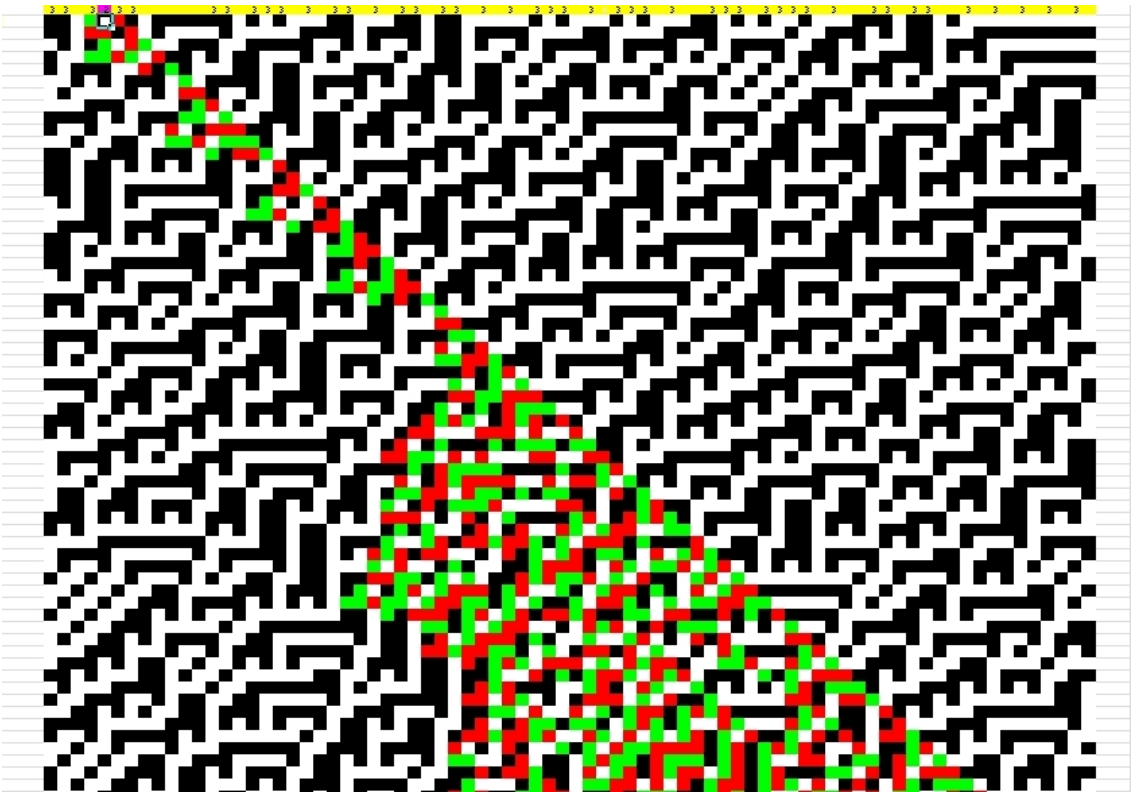
Salvaging CA-based crypto

- First experiments: investigate cycle distributions
- The following rule exhibited remarkable cycle lengths:
  - γ: flip the bit iff 2 cells at the right are not 01
- Invertible if periodic boundary conditions and odd length



Salvaging CA-based crypto

- First experiments: investigate cycle distributions
- The following rule exhibited remarkable cycle lengths:
  - γ: flip the bit iff 2 cells at the right are not 01
- nonlinear, but unfortunately, weak diffusion

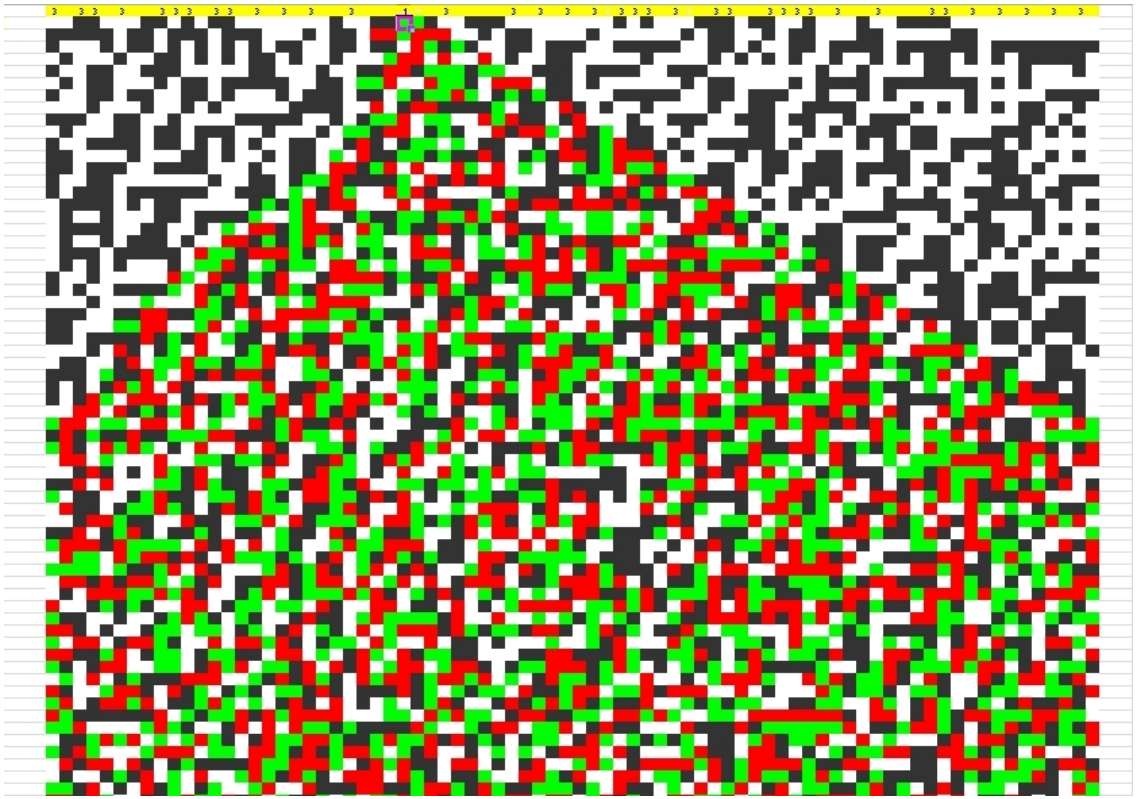


Salvaging CA-based crypto, second attempt

- Found invertible 5-bit neighborhood rules with good diffusion
- Turned out to be composition of γ and following rule
  - θ: add to bit the sum of 2 cells at the right modulo 2
- Idea: alternate γ (nonlinearity) and variant of θ (mixing)

Salvaging CA-based crypto, second attempt

- Found invertible 5-bit neighborhood rules with good diffusion
- Turned out to be composition of γ and following rule
  - θ: add to bit the sum of 2 cells at the right modulo 2
- diffusion much better but still slow

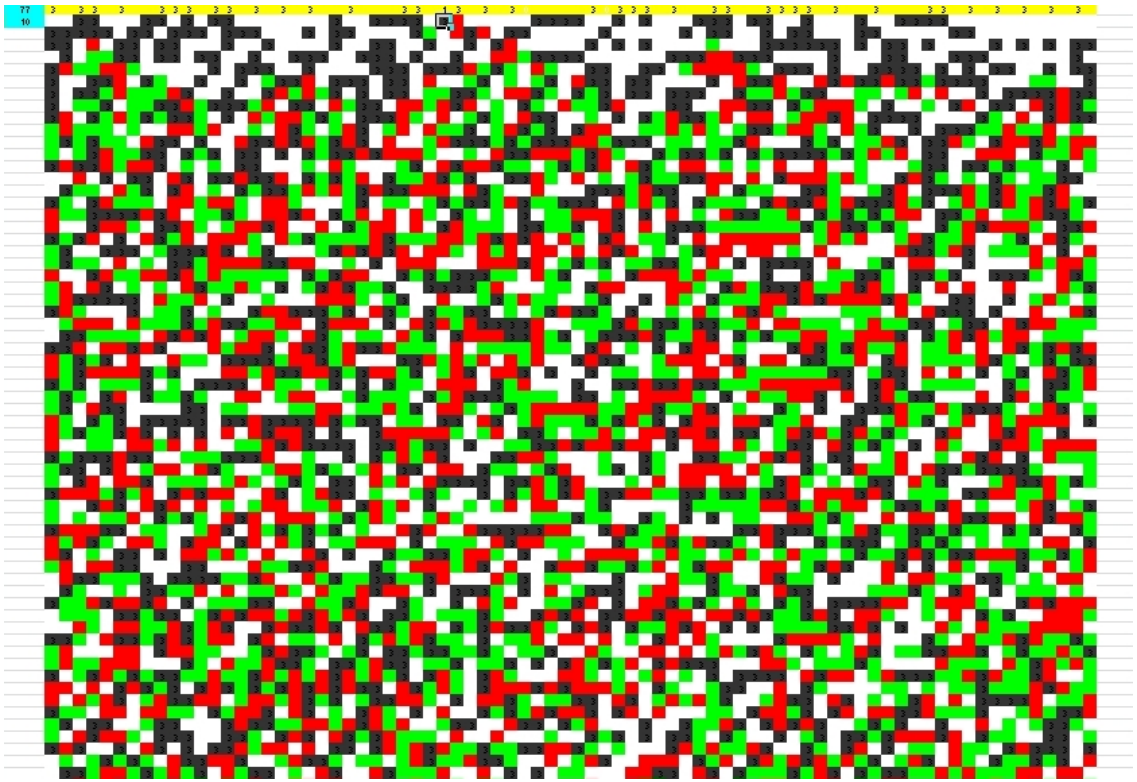


Salvaging CA-based crypto, third attempt

- Abandon locality by adding in bit transpositions:
  - π: move bit in cell i to cell $9i$ modulo the length
- Round function: $R = \pi \circ \theta \circ \gamma$

Salvaging CA-based crypto, third attempt

- Abandon locality by adding in bit transpositions:
  - π: move bit in cell i to cell $9i$ modulo the length
- full diffusion after few rounds!



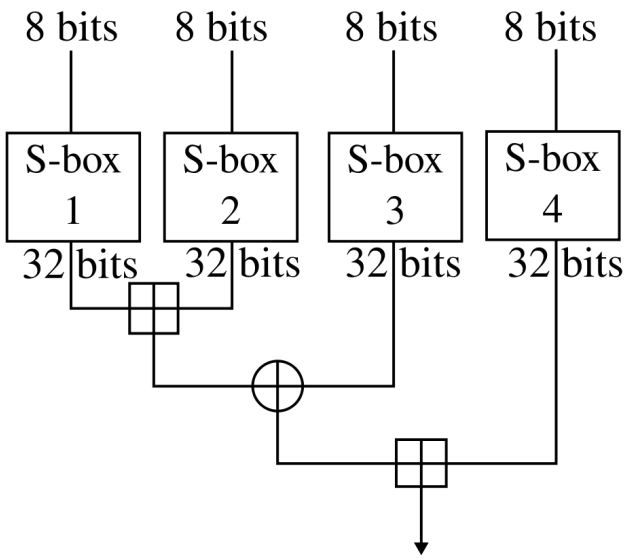
Resulting designs

- Round function composed of specialized steps
  - γ: non-linearity
  - θ: mixing
  - π: transposition
  - ι: addition of some constants for breaking symmetry
- Designs directly resulting from this
  - CellHash (1991): hash function
  - Subterranean (1992), StepRightUp (1994) and Panama (1997): hash/stream cipher modules
  - 3-Way and BaseKing (1993-94): block ciphers
- Theoretical basis: DC and LC
- Supporting concepts introduced in [PhD Thesis Daemen, 1995]
  - branch number
  - correlation matrices
  - wide trail strategy

Outline

March 1995: last month at COSIC, after PhD defense

- Blowfish [Schneier, 1993]
- F function:

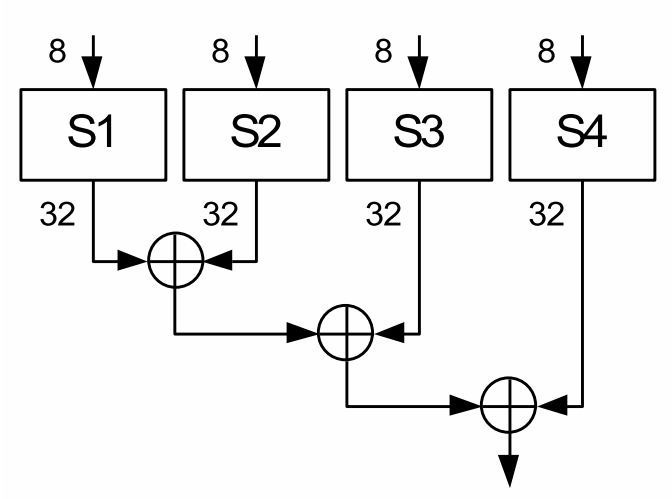


- 8-to-32-bit Sboxes
- Derived from key

- My impression
  - Great potential
  - Only 4 TLU and 3 additions
  - Very high diffusion
- Cryptanalysis contest in 1994
- Won by Serge Vaudenay
  - Exploiting collisions
  - In S-box: weak keys
  - In F-function
  - Published [Vaudenay, 1996]
- But can it be fixed?
  - Yes, it can!

March 1995: a month in Limbo; the Spark!

- Mixing ∘ S-box
- Both invertible

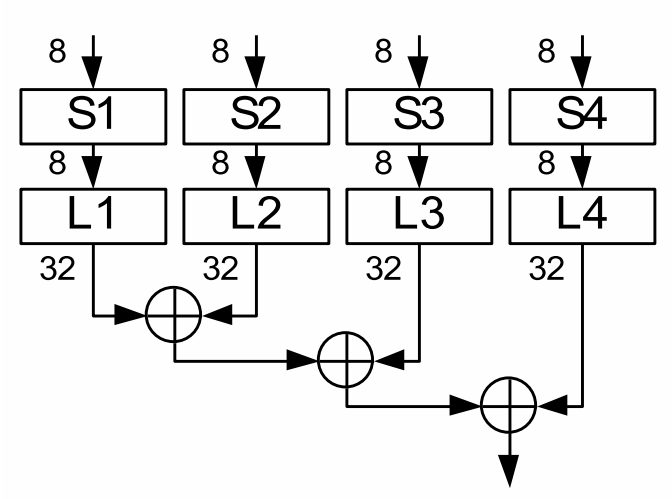


- 4 TLU and 4 XORs
- smuggled my idea out of COSIC ...

- S-boxes
  - just take a single one
  - optimize nonlinearity
  - criteria defined by DC and LC
- Linear mixing layer
  - optimize diffusion
- Clearly big potential!
- Challenge: finding right S-box and mixing layer

March 1995: a month in Limbo; the Spark!

- Mixing ∘ S-box
- Both invertible

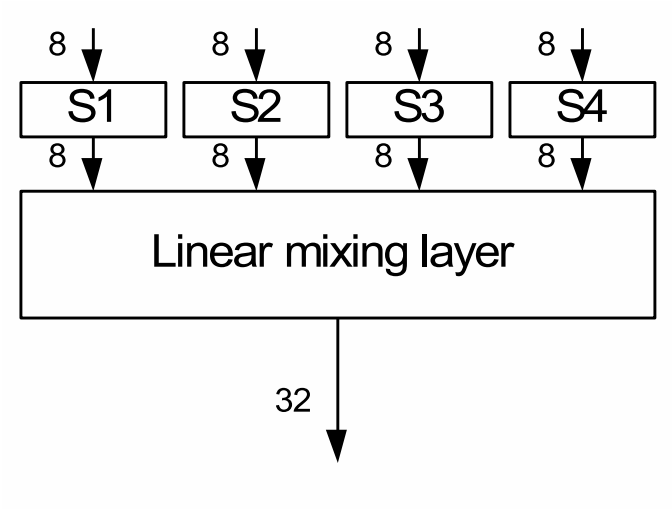


- 4 TLU and 4 XORs
- smuggled my idea out of COSIC ...

- S-boxes
  - just take a single one
  - optimize nonlinearity
  - criteria defined by DC and LC
- Linear mixing layer
  - optimize diffusion
- Clearly big potential!
- Challenge: finding right S-box and mixing layer

March 1995: a month in Limbo; the Spark!

- Mixing ∘ S-box
- Both invertible

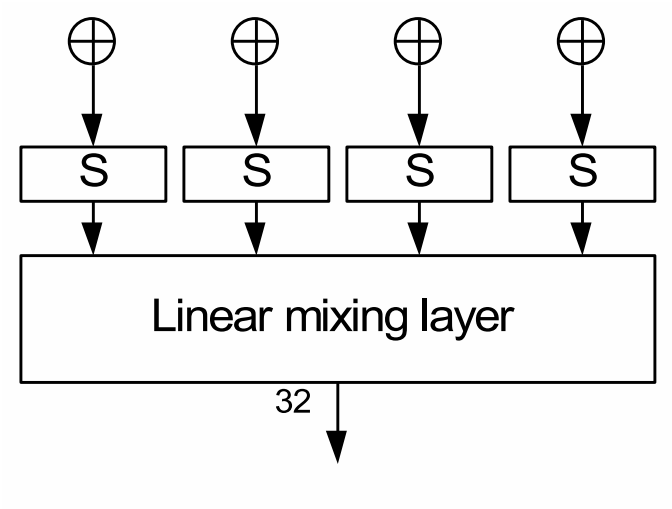


- 4 TLU and 4 XORs
- smuggled my idea out of COSIC ...

- S-boxes
  - just take a single one
  - optimize nonlinearity
  - criteria defined by DC and LC
- Linear mixing layer
  - optimize diffusion
- Clearly big potential!
- Challenge: finding right S-box and mixing layer

March 1995: a month in Limbo; the Spark!

- Mixing ∘ S-box
- Both invertible

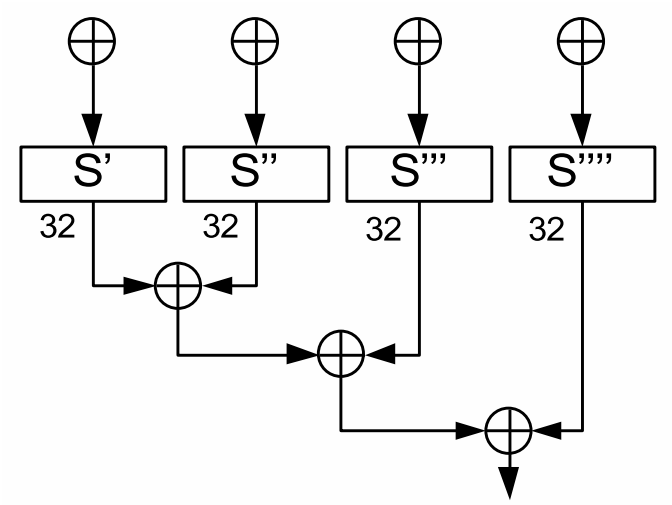


- 4 TLU and 4 XORs
- smuggled my idea out of COSIC ...

- S-boxes
  - just take a single one
  - optimize nonlinearity
  - criteria defined by DC and LC
- Linear mixing layer
  - optimize diffusion
- Clearly big potential!
- Challenge: finding right S-box and mixing layer

March 1995: a month in Limbo; the Spark!

- Mixing ∘ S-box
- Both invertible



- 4 TLU and 4 XORs
- smuggled my idea out of COSIC ...

- S-boxes
  - just take a single one
  - optimize nonlinearity
  - criteria defined by DC and LC
- Linear mixing layer
  - optimize diffusion
- Clearly big potential!
- Challenge: finding right S-box and mixing layer

Two years earlier ...

- Summer 1993: COSIC gets some classified contract work
- Supervisors decide to put on it:

Joan Daemen and Vincent Rijmen

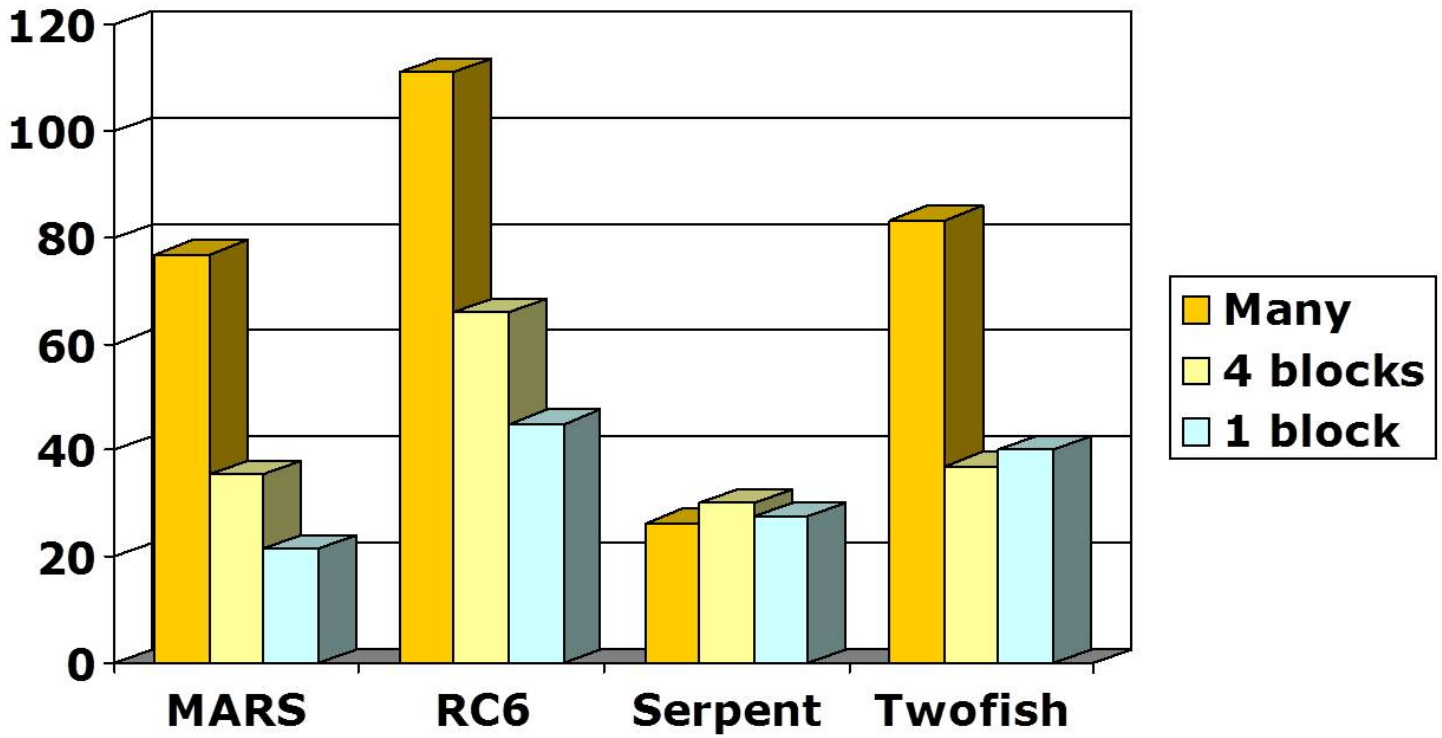
The road to Rijndael

Switch back to autumn 1995

- I decided to contact Vincent to work out my ideas
  - this lead to the following results
- SHARK [SHARK, FSE 1996]
  - link with maximum distance separable (MDS) codes
  - S-box: multiplicative inverse in GF(2^8) [Nyberg, 1994]
- SQUARE [SQUARE, FSE 1997]
  - more efficient thanks to byte transposition layer
  - state bytes arranged in a 4×4 square
- BKSQ [BKSQ, Cardis 1998]:
  - support for non-square states
- NIST AES call in autumn 1997
  - we defined RIJNDAEL using these ideas and submitted it

AES finalists: speed on Pentium

Percentage executed by the time Rijndael finishes:



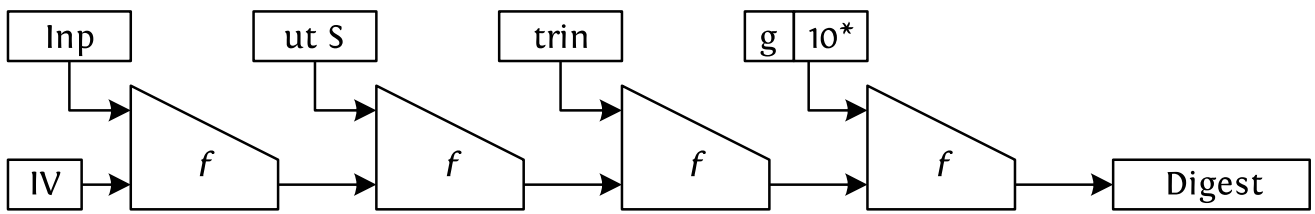
Rijndael (team) after AES selection

- October 2, 2000: NIST announces Rijndael will be AES
- Security of AES
  - most heard criticism: too simple to be secure
  - several times announced broken, false alarms
  - current status: some dents in armor due to academic attacks
    - biclique attacks [Khovratovich, Rechberger, Bogdanov, 2011]
    - up to a factor 4 more efficient than exhaustive key search
- Follow-up work with Vincent, some highlights
  - Rijndael book at Springer, the reference of block cipher design
  - new insights in differential propagation in AES-like functions
  - LC and DC statistics of random mappings
  - Pelican-MAC: 2.5 times faster than AES CBC-MAC

Outline

See how mainstream hash functions were going

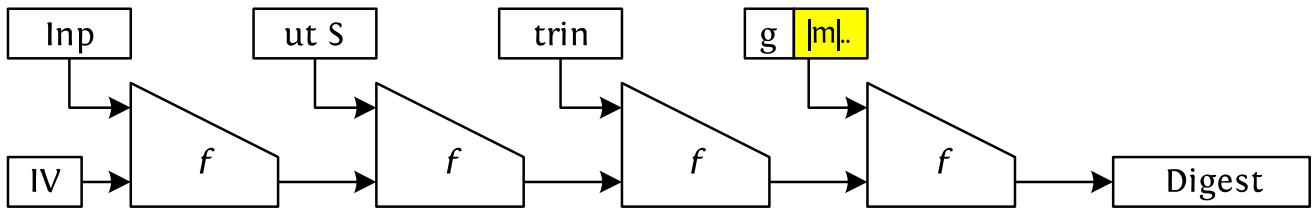
- Mainstream hash functions have two layers:
  - Fixed-input-length compression function
  - Iterating mode: *domain extension*
- Merkle-Damgård iterating mode: very simple and elegant



Yes, but can we have collision-resistance preservation?

The iterating mode

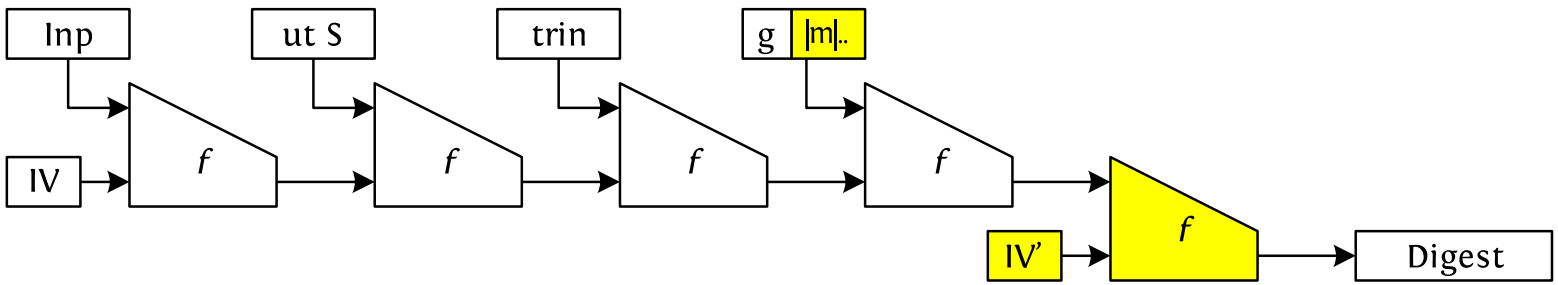
Merkle-Damgård with *strengthening*



Yes, but what about security when being used as a MAC?

The iterating mode

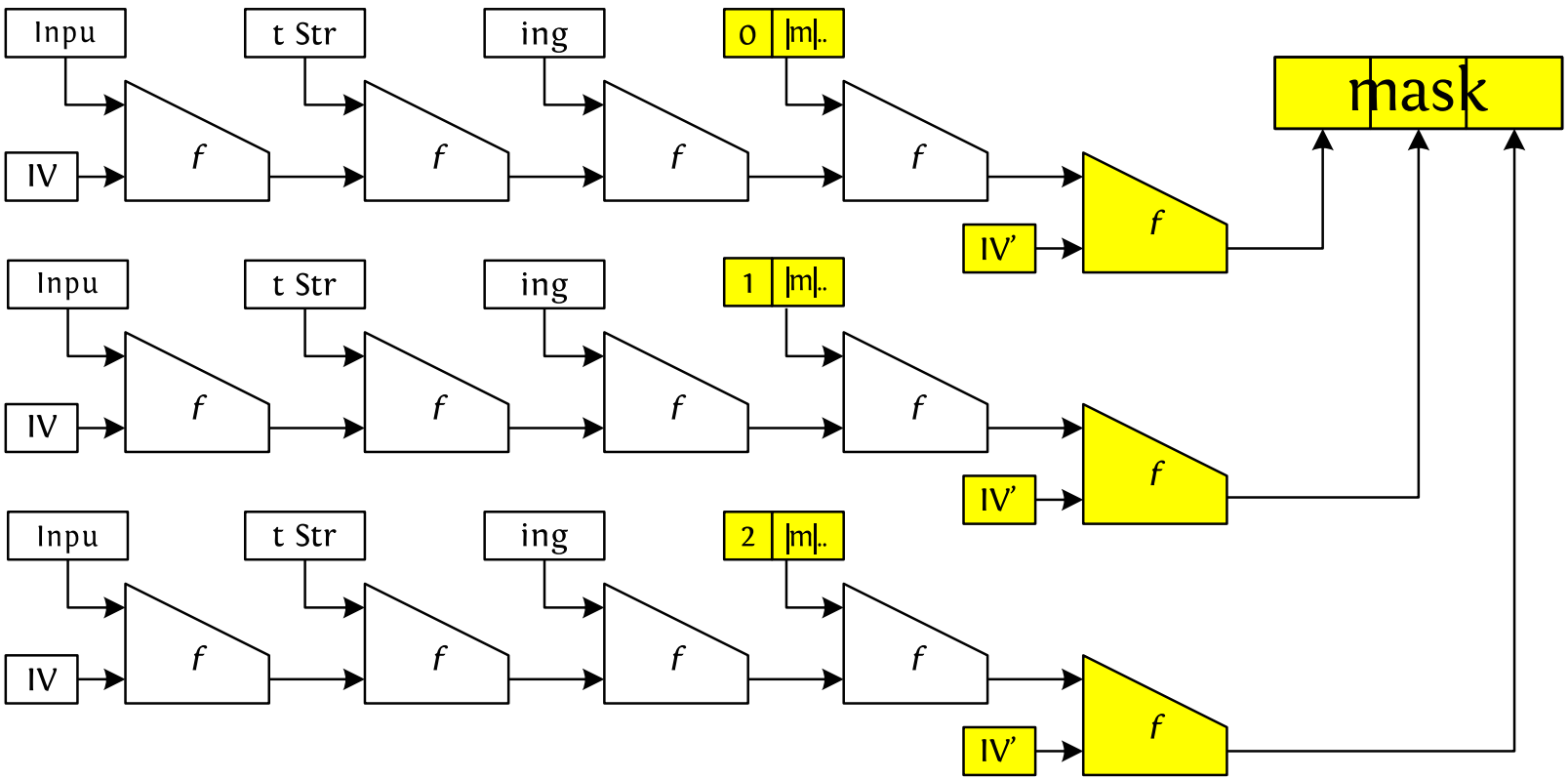
Enveloped Merkle-Damgård



Yes, but we often need long outputs, e.g., see PKCS#1, TLS, ...

The iterating mode

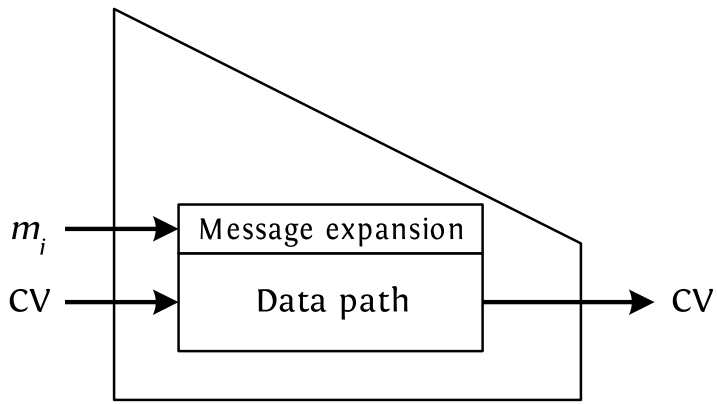
Mask generating function construction



This does what we need!

The compression function

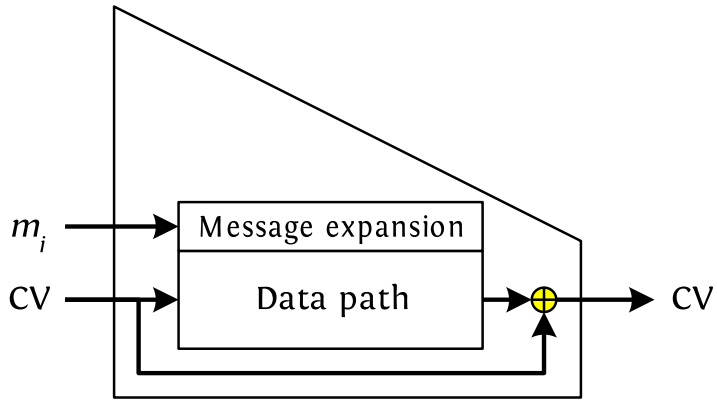
Let's put in a block cipher



Yes, but collisions are easy so collision-resistance preservation ...

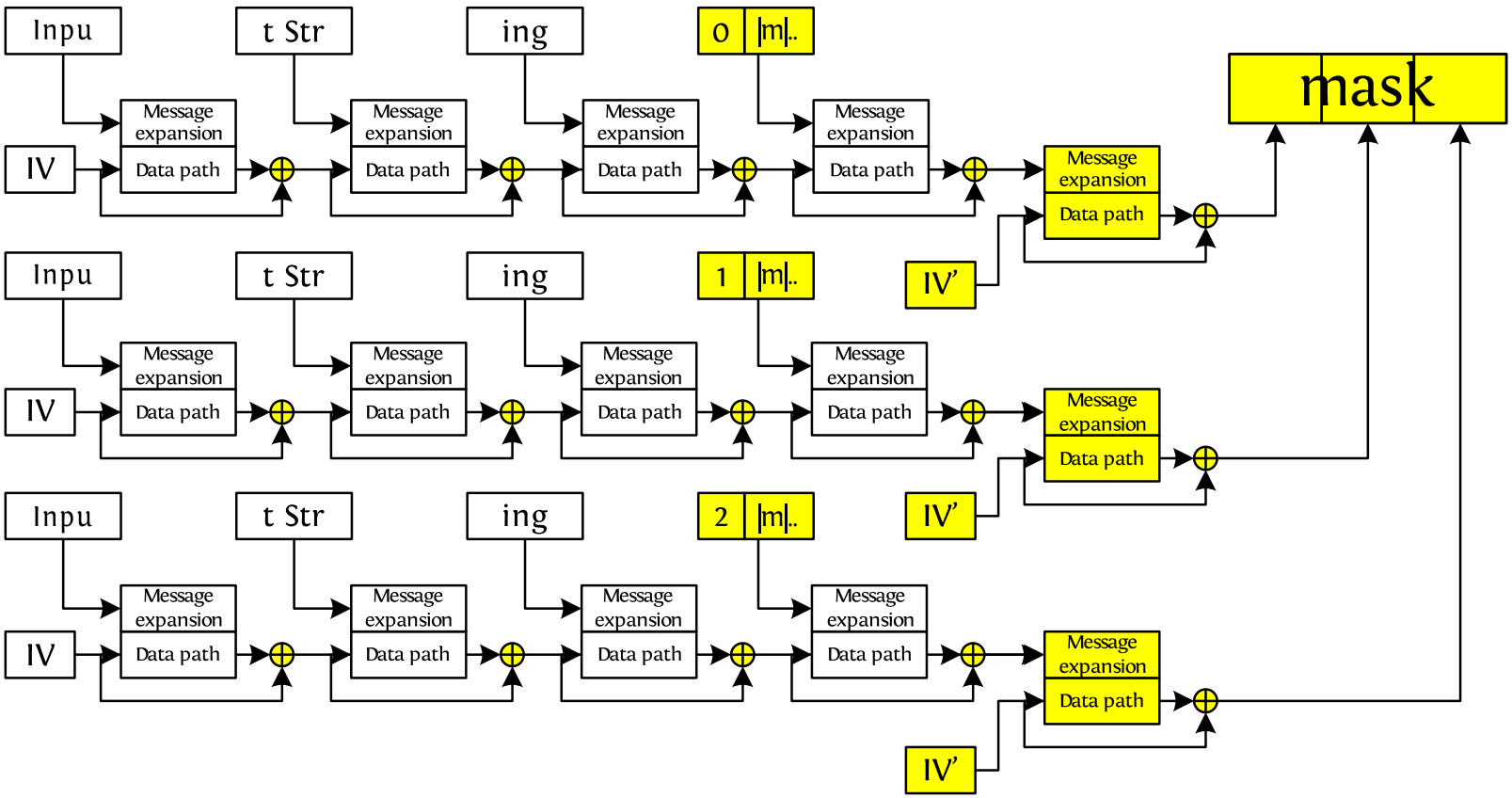
The compression function

Block cipher in Davies-Meyer mode



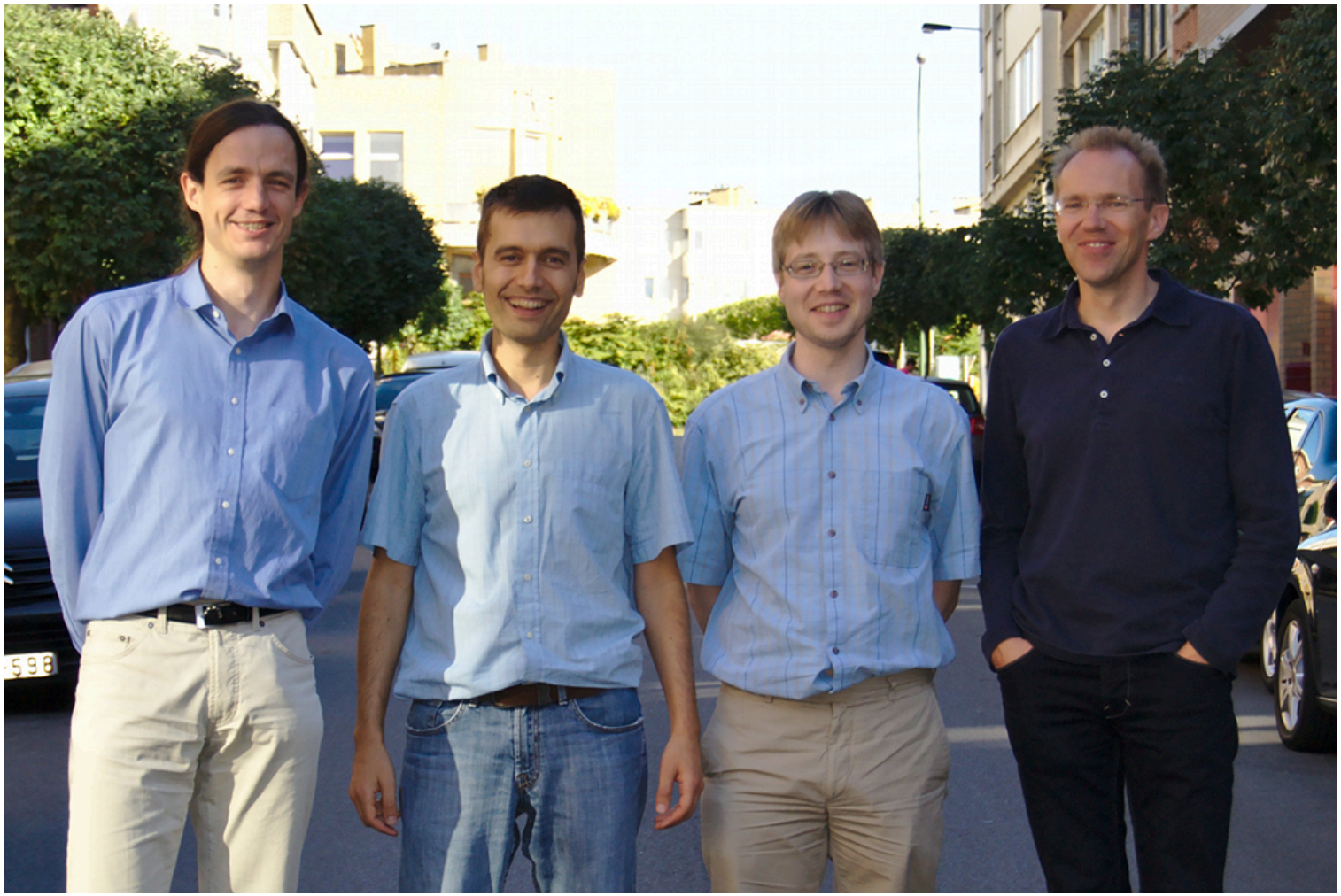
That's it!

What we end up with



Remains to do: building a suitable block cipher …

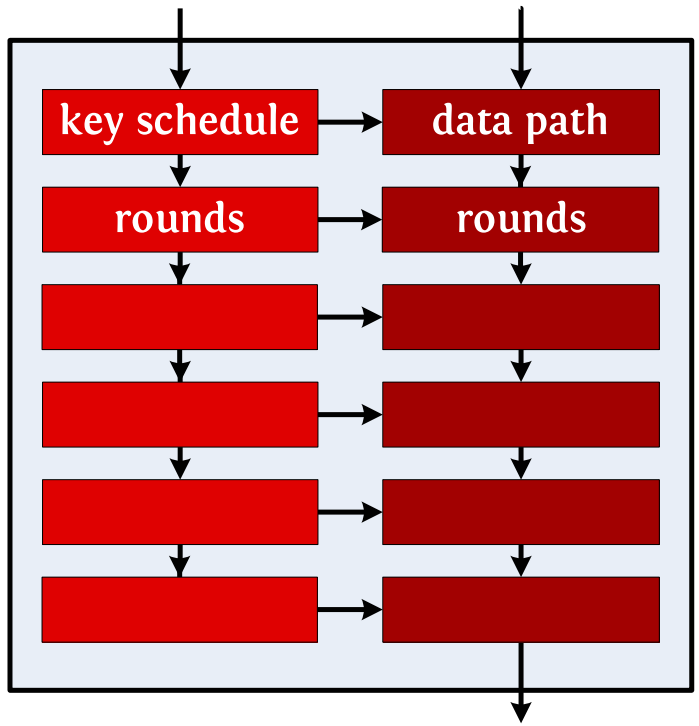
Keccak Team to the rescue!



Michaël Peeters, Guido Bertoni, Gilles Van Assche and Joan Daemen

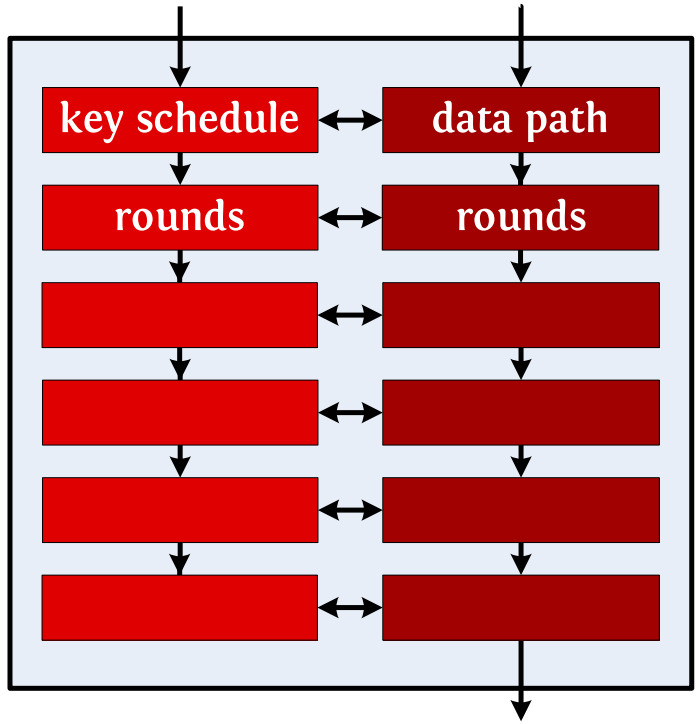
Do we really need a block cipher?

- No diffusion from data path to key (and tweak) schedule
- Let's remove these artificial barriers...
- That's an iterative permutation!



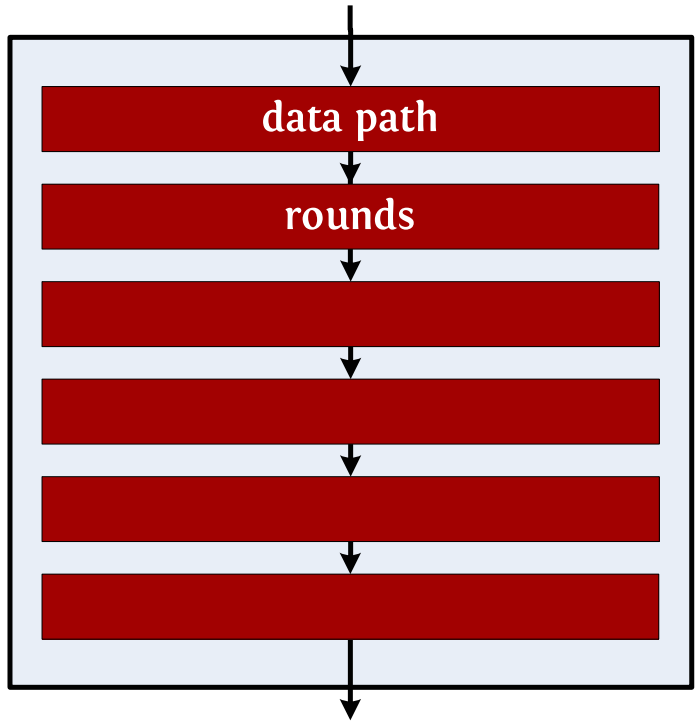
Do we really need a block cipher?

- No diffusion from data path to key (and tweak) schedule
- Let's remove these artificial barriers...
- That's an iterative permutation!



Do we really need a block cipher?

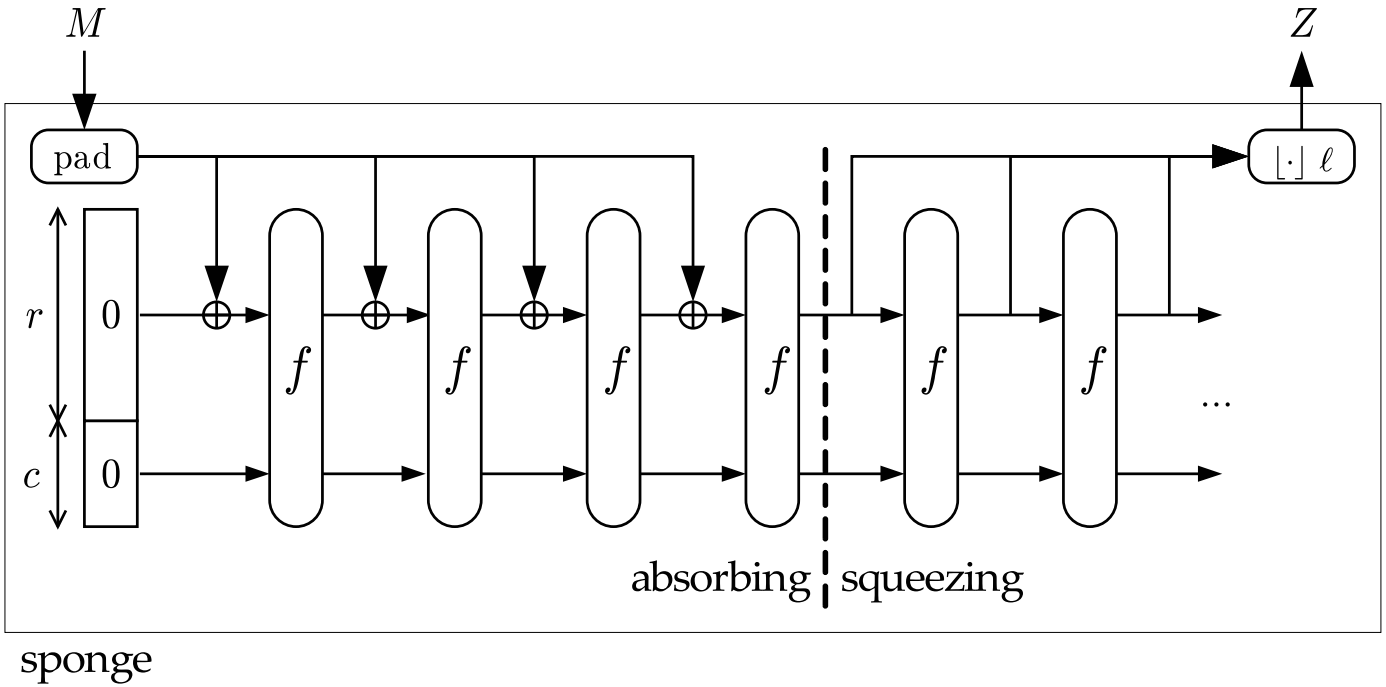
- No diffusion from data path to key (and tweak) schedule
- Let's remove these artificial barriers...
- That's an iterative permutation!



Let's re-factor the hashing mode

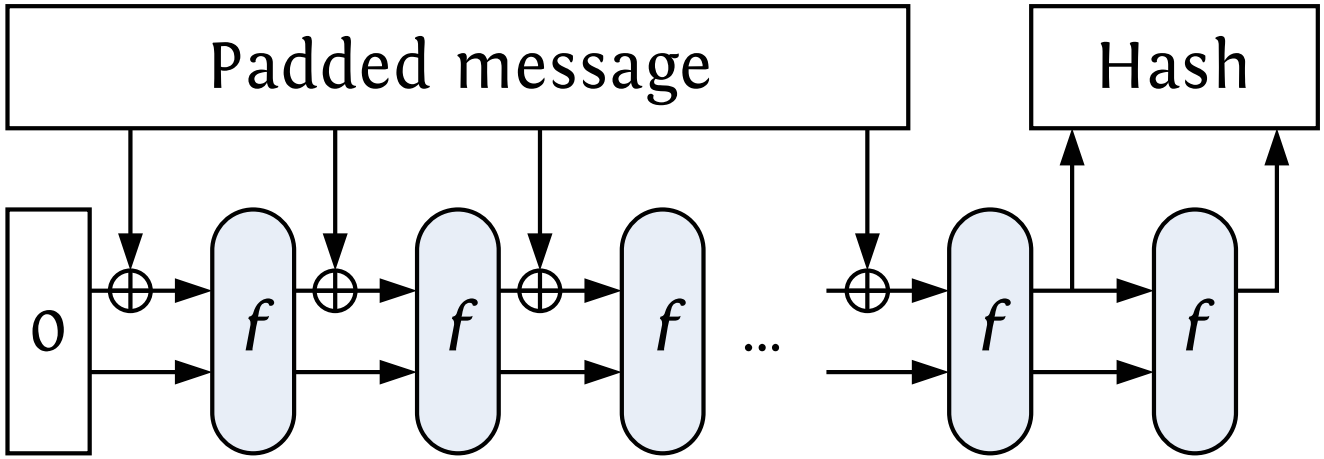
- Goal: hashing mode that is sound and simple
  - with good level of security against generic attacks
  - calling an iterated permutation rather than a block cipher
- Remaining problem: design of iterated permutation
  - round function: good approaches known
  - asymmetry: round constants
- Advantage of permutation compared to block ciphers:
  - less barriers $\Rightarrow$ more diffusion
  - no more need for efficient *decryption*
  - no more worries about key schedule

The sponge construction



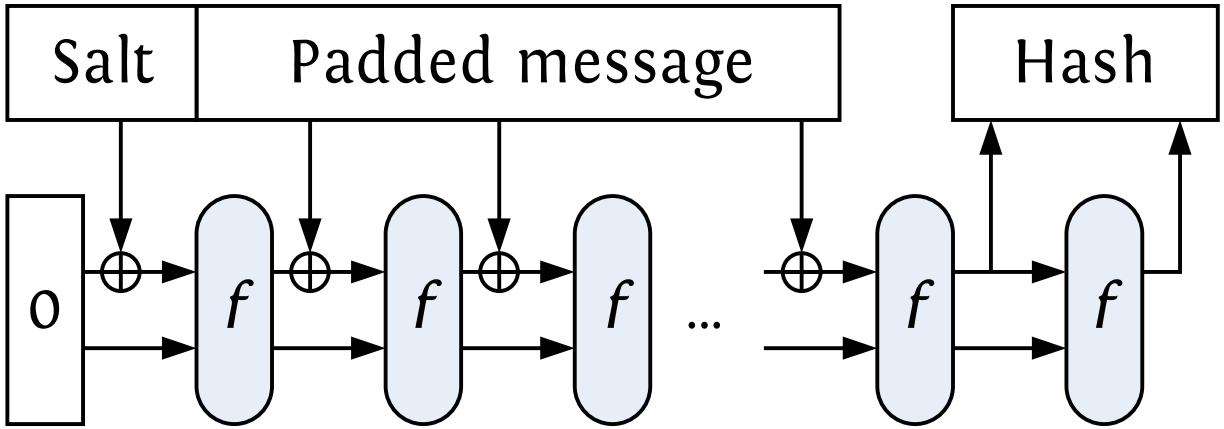
- Arbitrary *input* and *output* length
- Parameters: width b, rate r and capacity c with $b = c + r$
- Proven sound in indifferentiability framework [Maurer et al, 2004]
  - abandoning *property preservation* paradigm
  - security against generic attacks

Permutation-based hash function



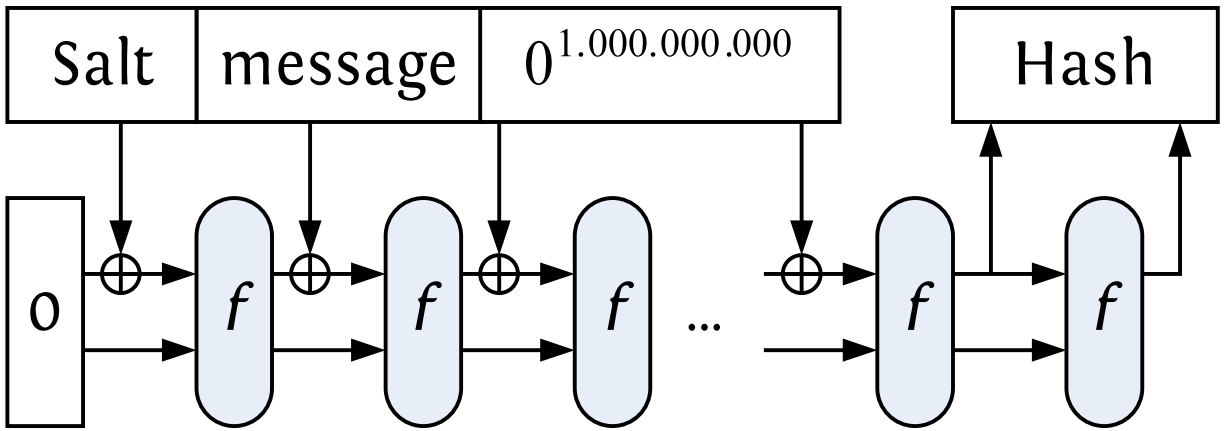
- Hashing

Permutation-based hash function



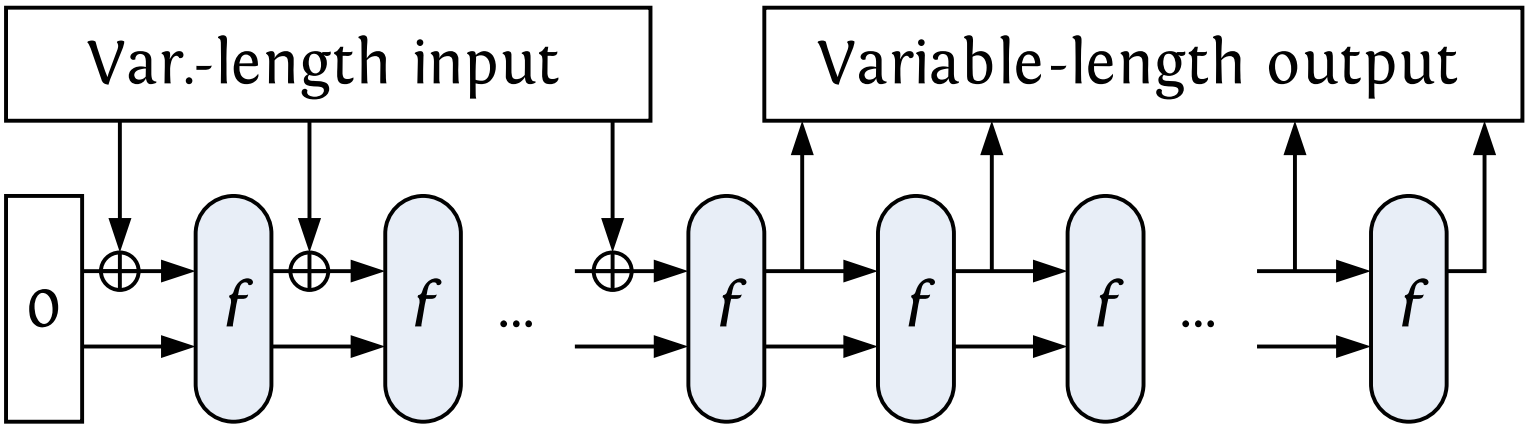
- Hashing
- Salted hashing

Permutation-based hash function



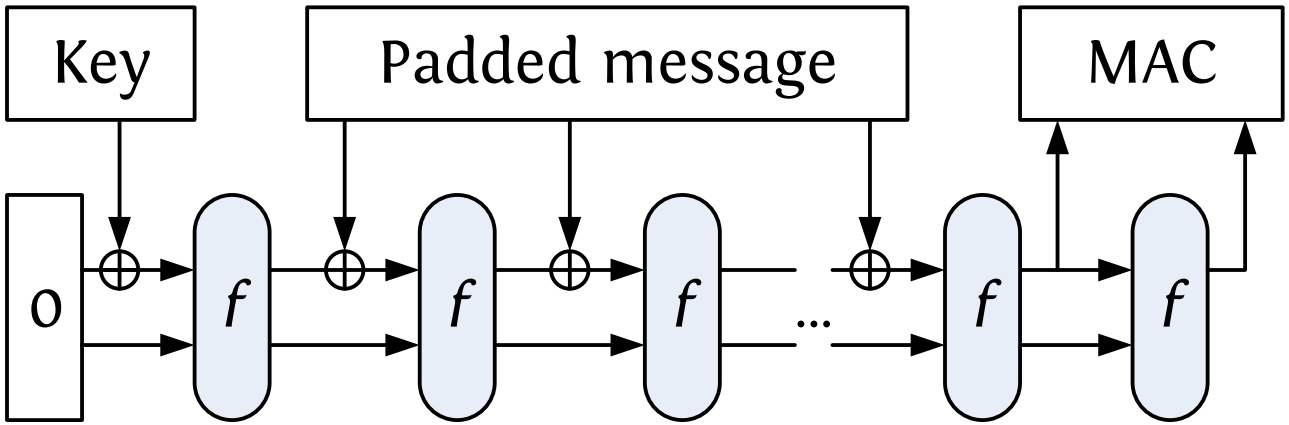
- Hashing
- ...Can be as slow as you like it!

Permutation-based mask generating function



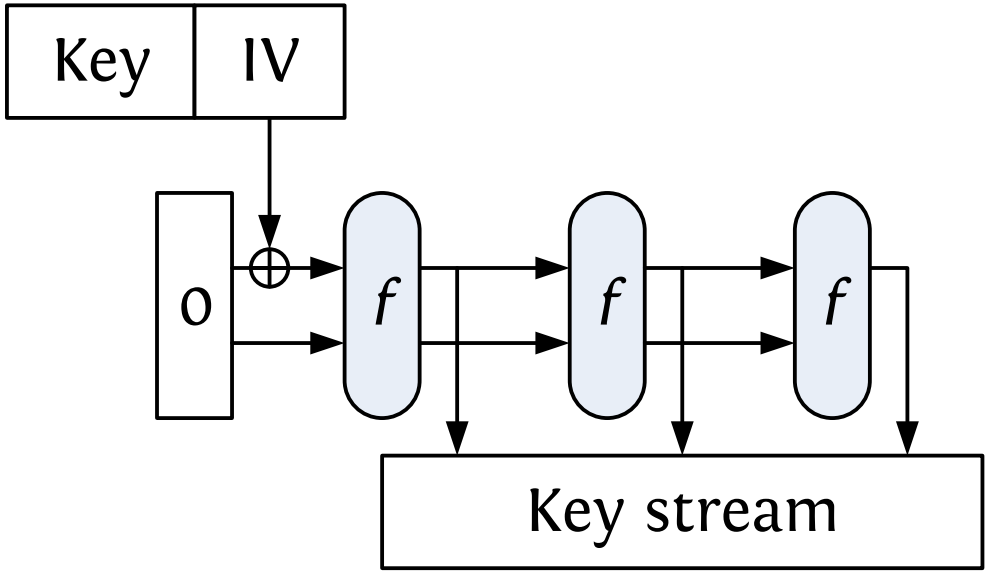
- Key derivation function in SSL, TLS
- Full-domain hashing in public key cryptography
  - electronic signatures RSA PSS [PKCS#1]
  - encryption RSA OAEP [PKCS#1]
  - key establishment RSA KEM [IEEE Std 1363a]

Permutation-based MACing



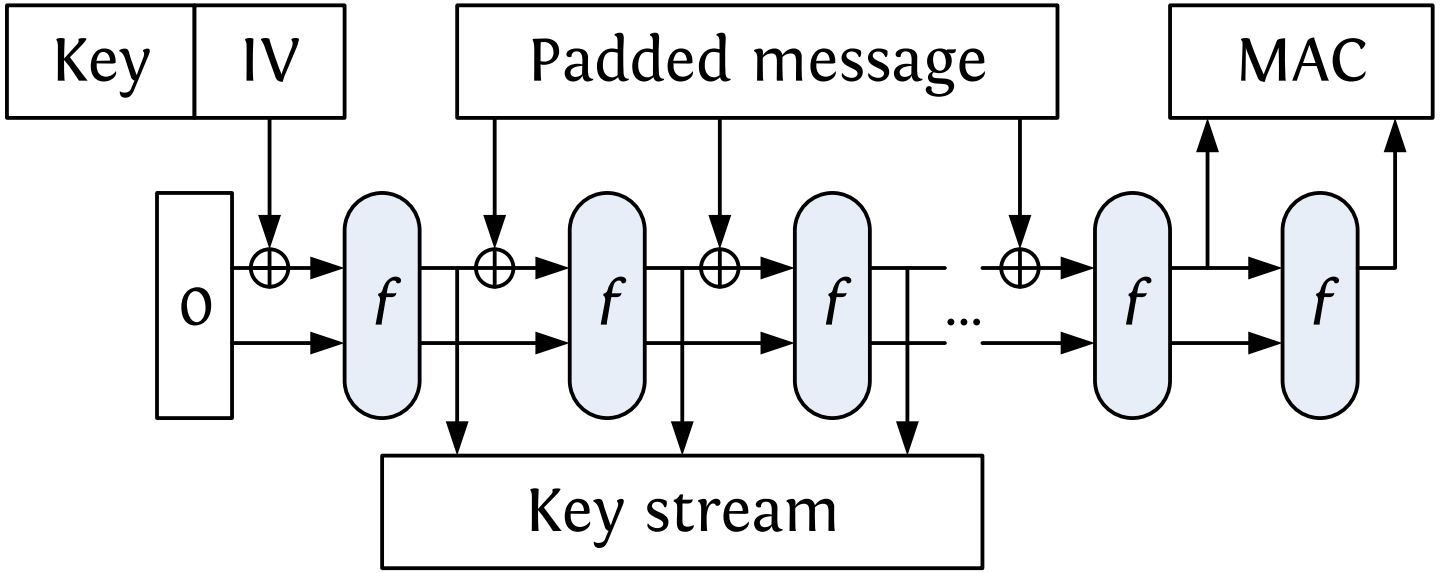
- No more need for HMAC [FIPS 198] for sponge
- HMAC plugs security hole in SHA-1 and SHA-2

Permutation-based (stream) encryption



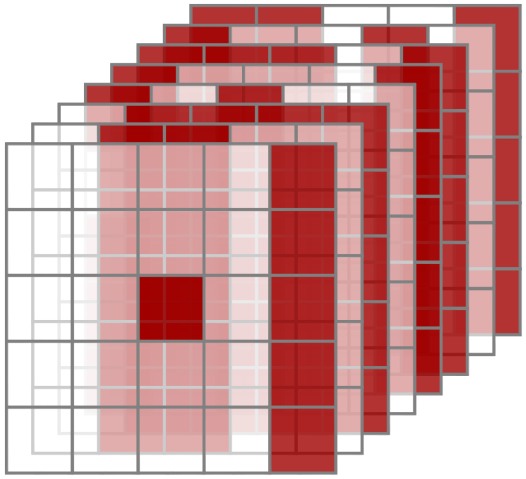
- Keystream generation

Permutation-based authenticated encryption



- Authentication and encryption in a single pass!
- Secure messaging (*SSL/TLS, SSH, IPSEC* ...)
- Duplex construction [Duplex, SAC 2011]
  - generic security equivalent to sponge construction
  - other applications include reseedable PRNG

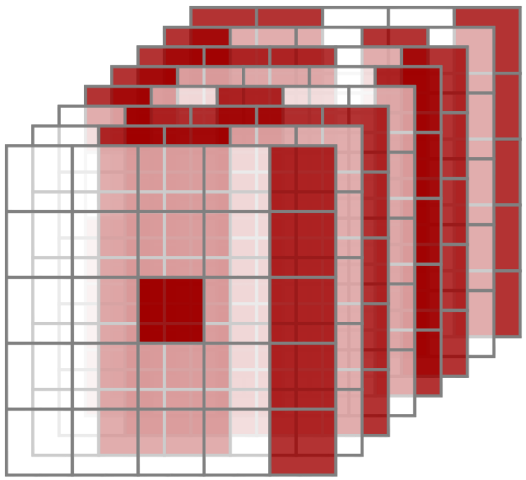
Keccak: the Seven Permutation Army



- (5×5) lanes
- up to 64-bit each

- Our SHA-3 submission
- Sponge calling one of 7 permutations:
  - 25, 50, 100, 200, 400, 800, 1600 bits
  - toy $\rightarrow$ lightweight $\rightarrow$ fastest
- repetition of a simple round function
  - lightweight and flexible
  - inspired by Subterranean, etc.
  - innovative, operating on a 3D state
- large safety margin
  - number of rounds: 24
  - best attacks known: 5 rounds [Dinur, Dunkelman, Shamir, 2012-13]

Keccak: the Seven Permutation Army

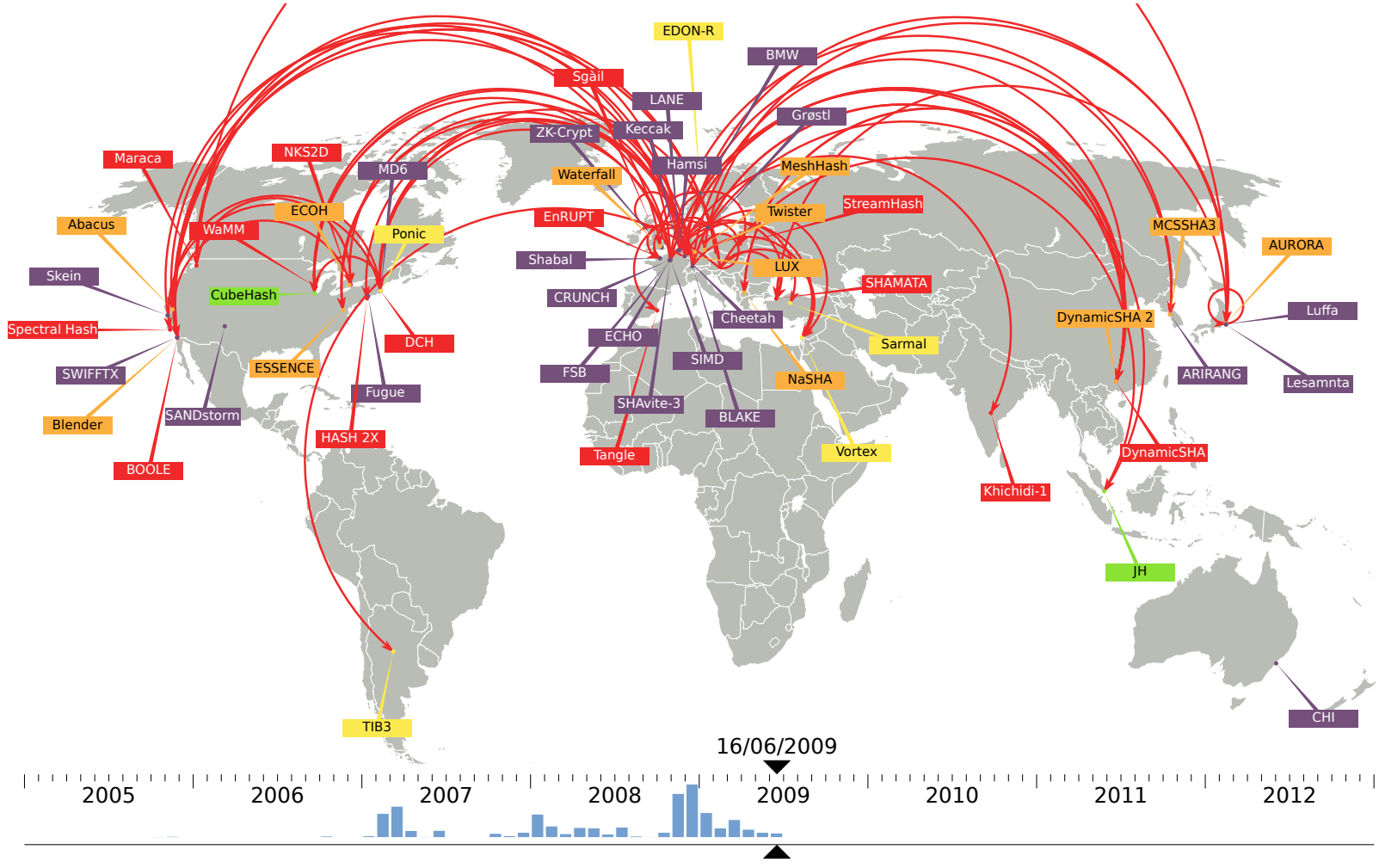


- (5×5) lanes
- up to 64-bit each

- First, choose your permutation ...
  - e.g. $\mathit{width} = 1600$
- ...then choose the *rate* and *capacity*
  - such that $\mathit{rate} + \mathit{capacity} = 1600$
- Security-speed trade-offs using the same permutation:

Rate	Capacity	Strength	Speed
1344	256	128	$\times 1.312$
1216	384	192	$\times 1.188$
1088	512	256	$\times 1.063$
1024	576	288	1.000

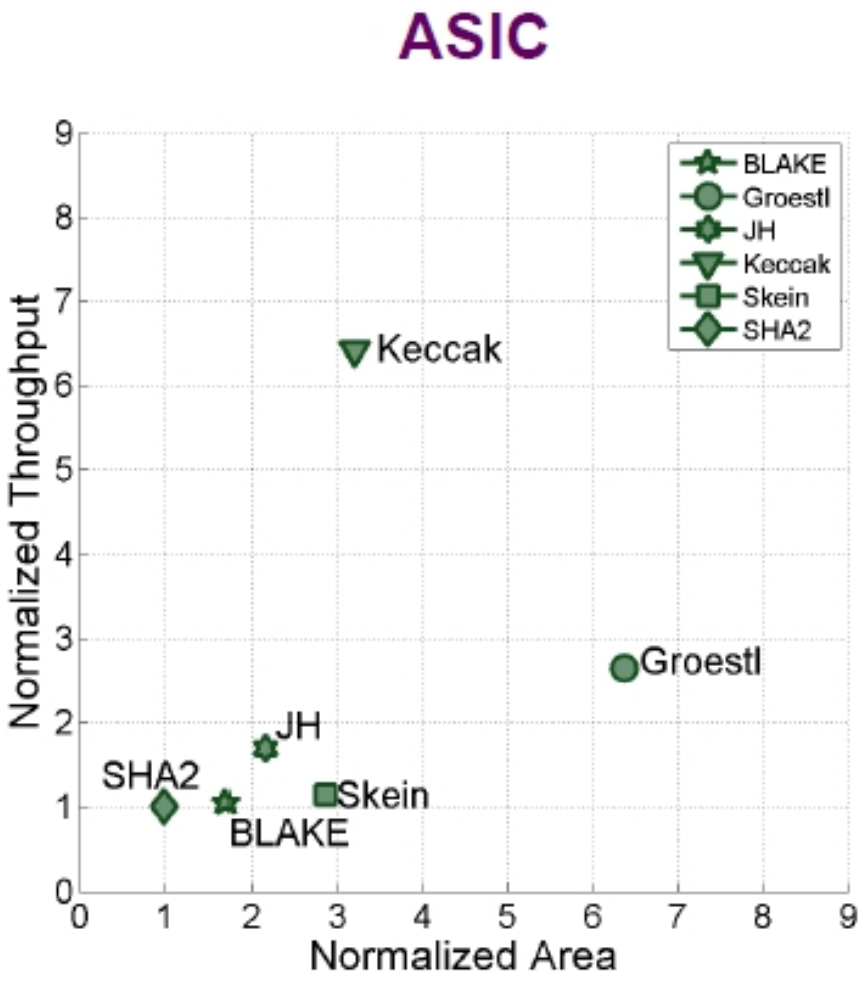
NIST SHA-3: a tough competition

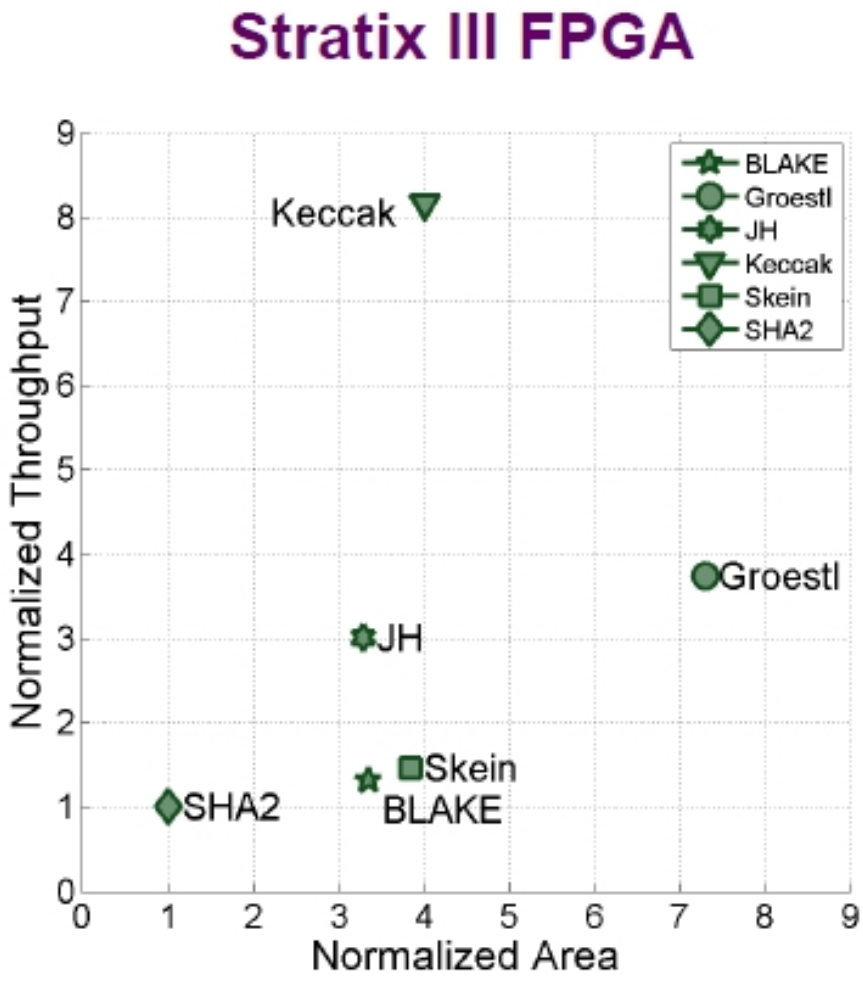


[courtesy of Christophe De Cannière]

Efficiency of Keccak in hardware

From Kris Gaj's presentation at SHA-3, Washington 2012:





Long-term effort

- Rumours about NIST call for hash functions (late 2005)
  - forming of Keccak Team
  - starting point: fixing Panama [Daemen, Clapp, FSE 1998]
- RadioGatún [Keccak team, NIST 2nd hash workshop 2006]
  - variable-length output, streaming oriented
  - for security claim: sponge functions [Keccak team, Ecrypt hash, 2007]
- RadioGatún confidence crisis (2007-2008)
  - third-party and our own cryptanalysis did not inspire confidence
  - NIST SHA-3 deadline approaching ...
  - U-turn: design a sponge with strong permutation f: Keccak
- October 2, 2012: NIST announces Keccak will be SHA-3
- Ongoing work:
  - tree hashing and Sakura
  - dedicated keyed modes (CAESAR competition),
  - protection against side-channel attacks ...

Outline

Conclusions: trying to do things right

- re-factoring over patching
  - fresh AES instead of DES-derivative
  - sponge instead of trying to fix Merkle-Damgård, e.g. Haifa
  - KECCAK structure instead of just a heavier ARX
- simplicity over complexity
  - single S-box in AES instead of several different ones
  - permutation-based instead of block-cipher based crypto
  - KECCAK: CA-based mappings instead of S-boxes and MDS
- result-focused over publication-driven
  - hard to get design ideas published
  - examples: original sponge paper, sound tree hashing
  - turn out to be influential in the long run
  - ...after linear complexity, T-functions, cube attacks etc. have long been forgotten

Conclusions: team up with critical minds

- How to build clean designs?
  - try out many ideas
  - throw most of them away
  - keep the good ones
- The process: collaboration and confrontation
  - in a team with critical minds
  - overlapping competences rather than complementary
  - not too much ego please
- Great to work with Vincent, Guido, Michaël and Gilles!
  - Rijndael/AES: ubiquitous by now and security still solid
  - sponge/duplex: new permutation-based crypto paradigm
  - Keccak/SHA-3: common sense made it to hashing, at last

Questions?

Thanks for your attention!

More information on
http://sponge.noekeon.org/
http://keccak.noekeon.org/