The Grindahl hash functions

Søren S. Thomsen
joint work with
Lars R. Knudsen Christian Rechberger

Fast Software Encryption
March 26–28, 2007
Luxembourg

MD4-style hash functions

- Many hash functions; MD4, MD5, RIPE-MD, SHA-1, ...
- n-bit output, n-bit state
- Simple (fast) state update
- Repeat many times

Attack methods

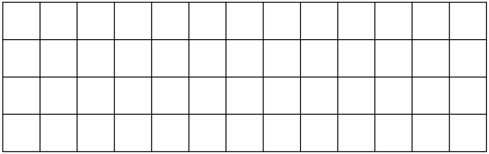
- Local collisions:
  - Introduce difference
  - "Undo" difference as quickly as possible (probabilistic)
- Small difference means behaviour is more predictable
- Success with high probability

Thoughts behind our design

- Ensure quick diffusion (in both directions)
- Limited control over differences
- (All) collision trails are wide
- Block cipher techniques

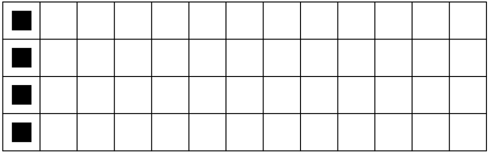
Grindahl-256

- Based on Rijndael block cipher
- 256-bit output
- State: 4×13 matrix of bytes (initially all zero)
- SubBytes and MixColumns as in Rijndael
- ShiftRows rotates right by $1, 2, 4, 10$ positions



Grindahl-256: round function

- 4-byte message block **replaces** first state column
- New operation: AddConstant. Flips last bit of last byte
- Do **one** round: AddConstant, SubBytes, ShiftRows, MixColumns
- Round function a permutation → invertible



Grindahl-256: output

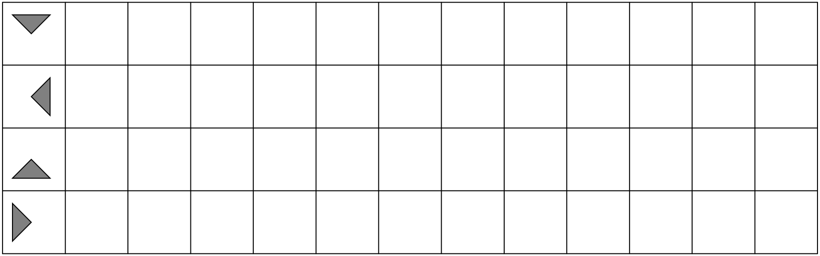
- After last message block, do 8 more (“blank”) rounds (permutation)
- Output right-most 8 columns

Grindahl-256: ShiftRows

- Why change ShiftRows?
- Improve diffusion speed
- Every state byte depends on every message byte after 4 rounds

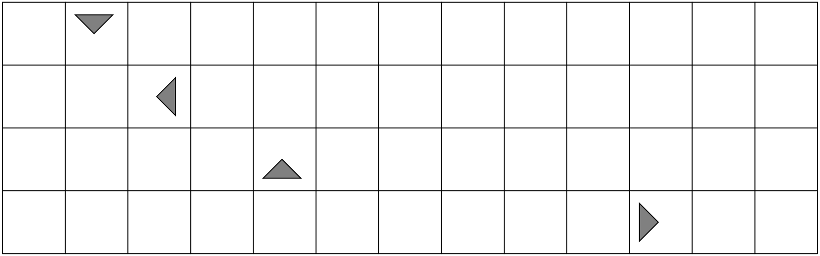
How a message block affects the state

Message injected:



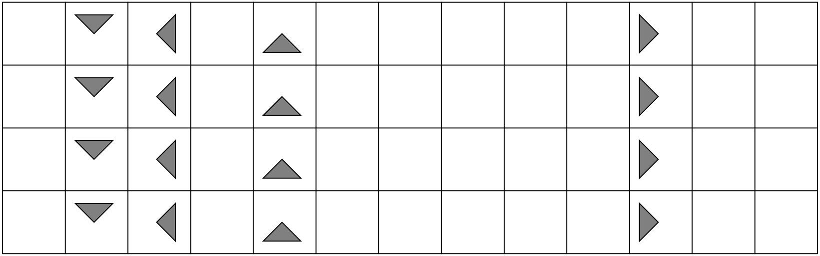
How a message block affects the state

After ShiftRows (1st round):



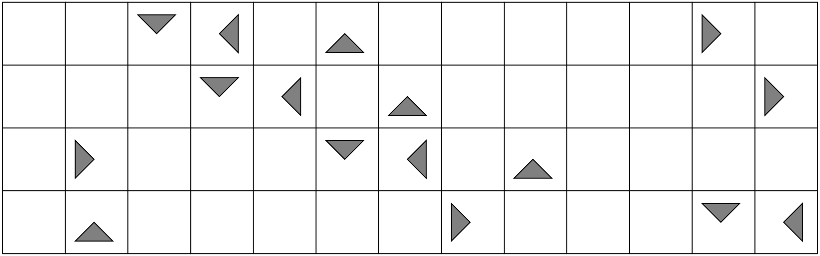
How a message block affects the state

After MixColumns (1st round):



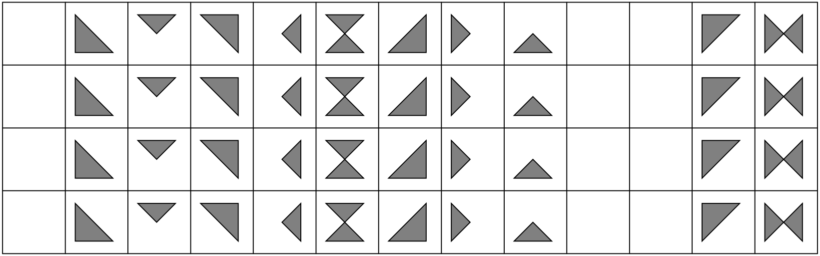
How a message block affects the state

After ShiftRows (2nd round):



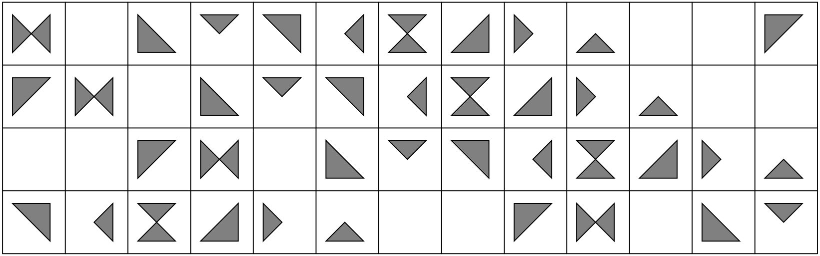
How a message block affects the state

After MixColumns (2nd round):



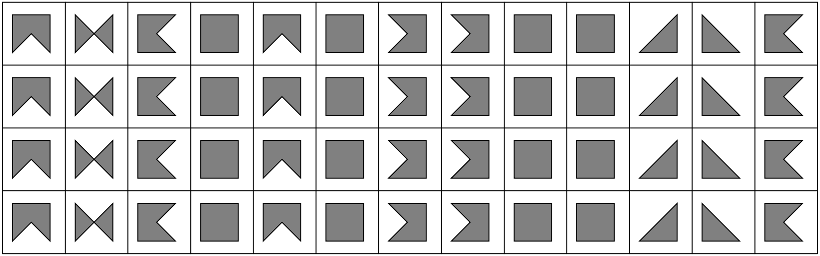
How a message block affects the state

After ShiftRows (3rd round):



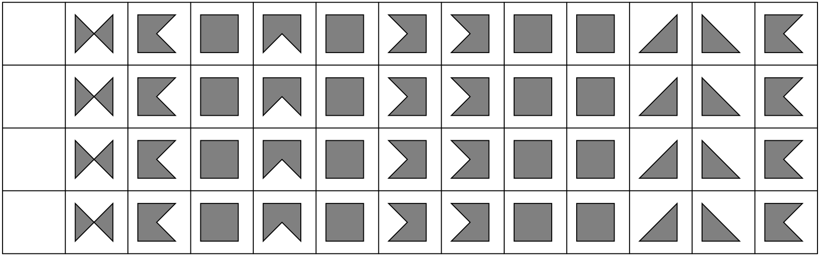
How a message block affects the state

After MixColumns (3rd round):



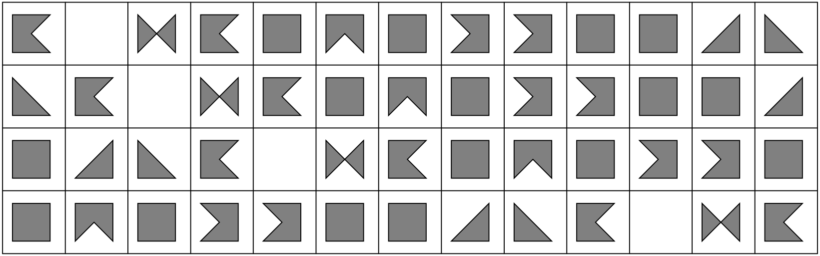
How a message block affects the state

Wiping first column:



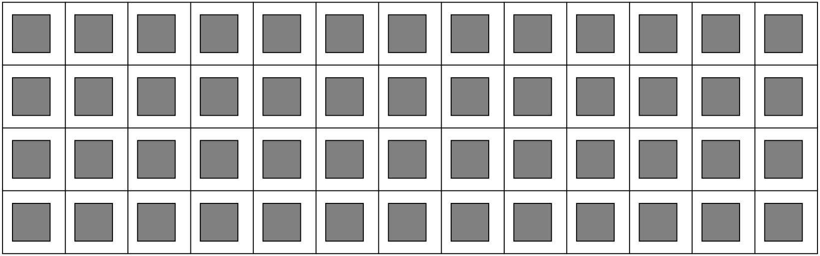
How a message block affects the state

After ShiftRows (4th round):



How a message block affects the state

After MixColumns (4th round):



Grindahl-256: AddConstant

- Why AddConstant?
- Without AddConstant: 13 equal columns invariant

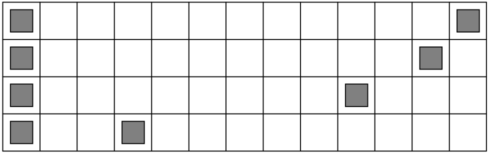
a	a	a	a	a	a	a	a	a	a	a	a	a
b	b	b	b	b	b	b	b	b	b	b	b	b
c	c	c	c	c	c	c	c	c	c	c	c	c
d	d	d	d	d	d	d	d	d	d	d	d	d

Grindahl-256: blank rounds

- Why 8 blank rounds?
- 4 rounds required to make output depend on last block
- Security margin (Chicken-hash)

Grindahl-256: columns

- Why 13 columns?
- At least 10 columns, otherwise birthday attack
- Round function invertible → meet-in-the-middle
- Hence, (2nd) preimage below 2^n (claim $2^{n/2}$)
- (Chicken-hash again)



Grindahl-256: diffusion

- Collision requires intermediate state with $\geq$ half the bytes active
- Internal collision requires > 4 input rounds

Grindahl-256: speed

- Optimisations known from AES
- Many trade-offs, good performance across platforms
- Low memory requirements
- Rough comparison with `crypto++` (Pentium 4 impl.):

Function	Relative time/byte
Grindahl-256	1.0
AES-128	$\sim$1.0
SHA-256	$\sim$1.4

Concluding remarks

- We propose the Grindahl hash functions
  - two instances, Grindahl-256 and Grindahl-512
  - large class of hash functions (highly parameterizable)
  - can also be used as compression function
- Some properties are
  - quick diffusion
  - high degree of non-linearity
  - fast implementations across platforms
  - implementation research “reusable” from the AES
  - low memory requirements

Thank you for listening!