Symmetric Key Cryptography

PQCRYPTO Summer School on Post-Quantum Cryptography 2017

Stefan Kölbl

June 19th, 2017

DTU Compute, Technical University of Denmark

Introduction to Symmetric Key Cryptography

Symmetric Key Cryptography

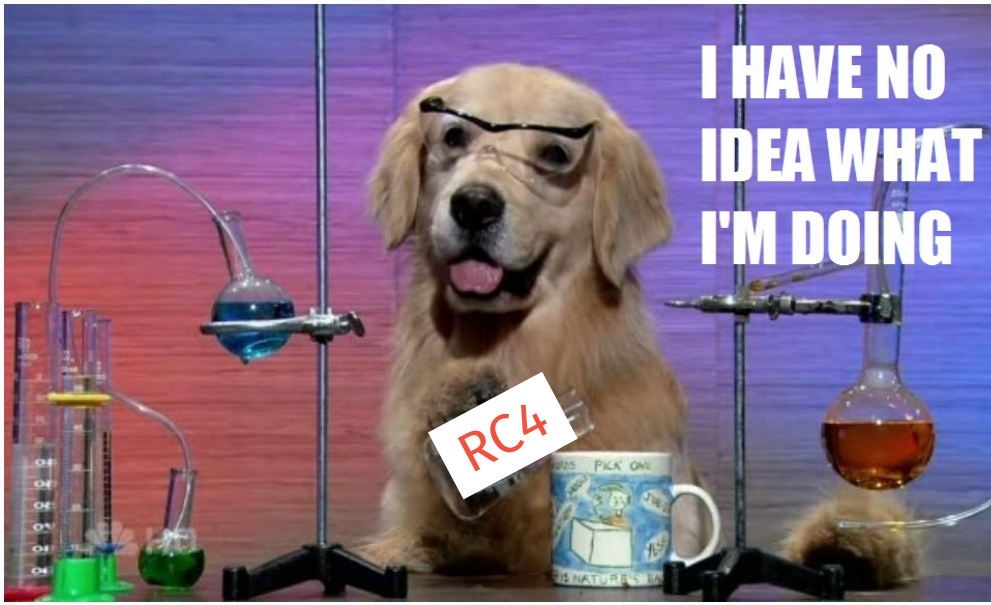
Where does security fail?

-
-
-
-

Symmetric Key Cryptography

Where does security fail?

- User
-
-
-



Don’t blame the user!

Symmetric Key Cryptography

Where does security fail?

- User
- Implementation
-
-

Heartbleed

Symmetric Key Cryptography

Where does security fail?

- User
- Implementation
- Protocols
-

Drown Attack

Symmetric Key Cryptography

Where does security fail?

- User
- Implementation
- Protocols
- Cryptographic Algorithms

Myth

”Cryptographic Algorithms are *never* the weakest link.”

Symmetric Key Cryptography

Hash Function MD5

- Not collision resistant [WY05]
- Constructing a rogue CA [Ste+09]

Hash Function SHA-1

- Not collision resistant [WYY05]
- First practical collisions this year

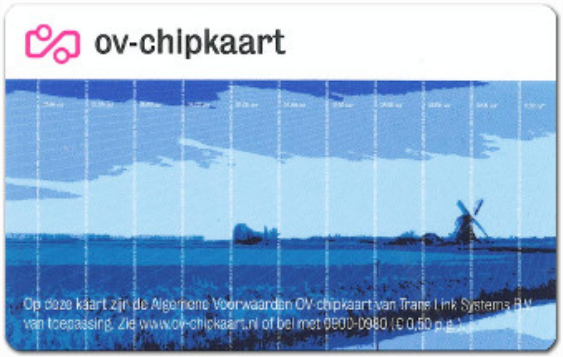
Stream Cipher RC4

- Plaintext Recovery in TLS [AlF+13]
- ...

Symmetric Key Cryptography

A long list...

- MIFARE Classic (Crypto 1)
- Keeloq
- A5/1, A5/2
- DECT
- Kindle Cipher
- ...



Symmetric Key Cryptography

What can we do?

- Encryption
- Authentication (MAC)
- Hashing
- Random Number Generation
- Digital Signature Schemes
- Key Exchange

Symmetric Key Cryptography

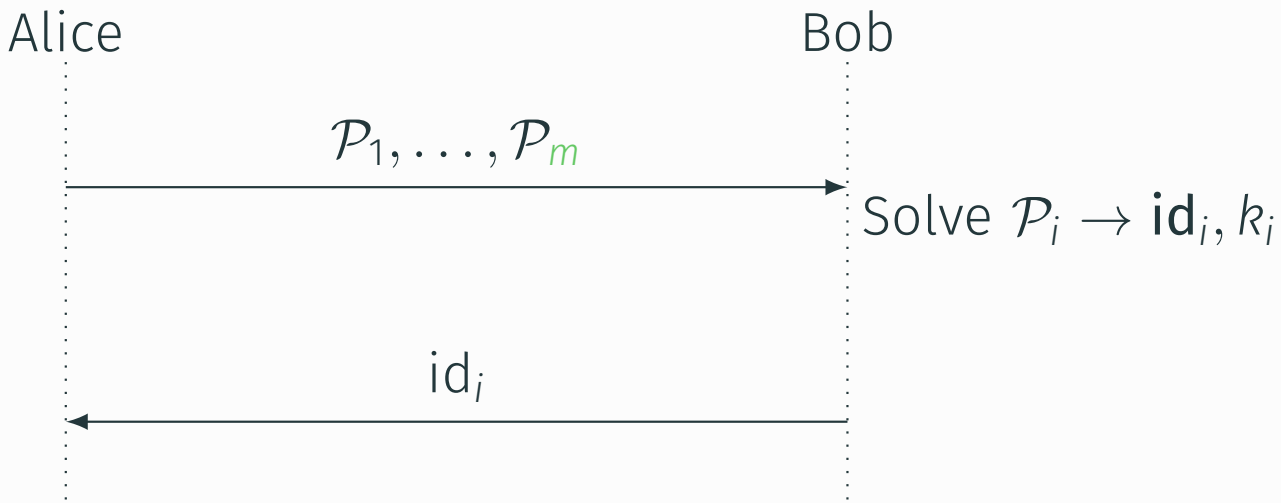
Digital Signatures

- Hash-based Signature Schemes (MSS, XMSS [BDH11], SPHINCS [Ber+15])
- Zero-Knowledge Proof Based (Fish [Cha+17], Picnic [Cha+17])

Symmetric Key Cryptography

Key Exchange with Merkle Puzzles (1978)

- Alice prepares m Puzzles: $\mathcal{P}_1, \ldots, \mathcal{P}_m$.
- Solving a puzzle requires n steps.
- Reveals an id and key k_{id}.

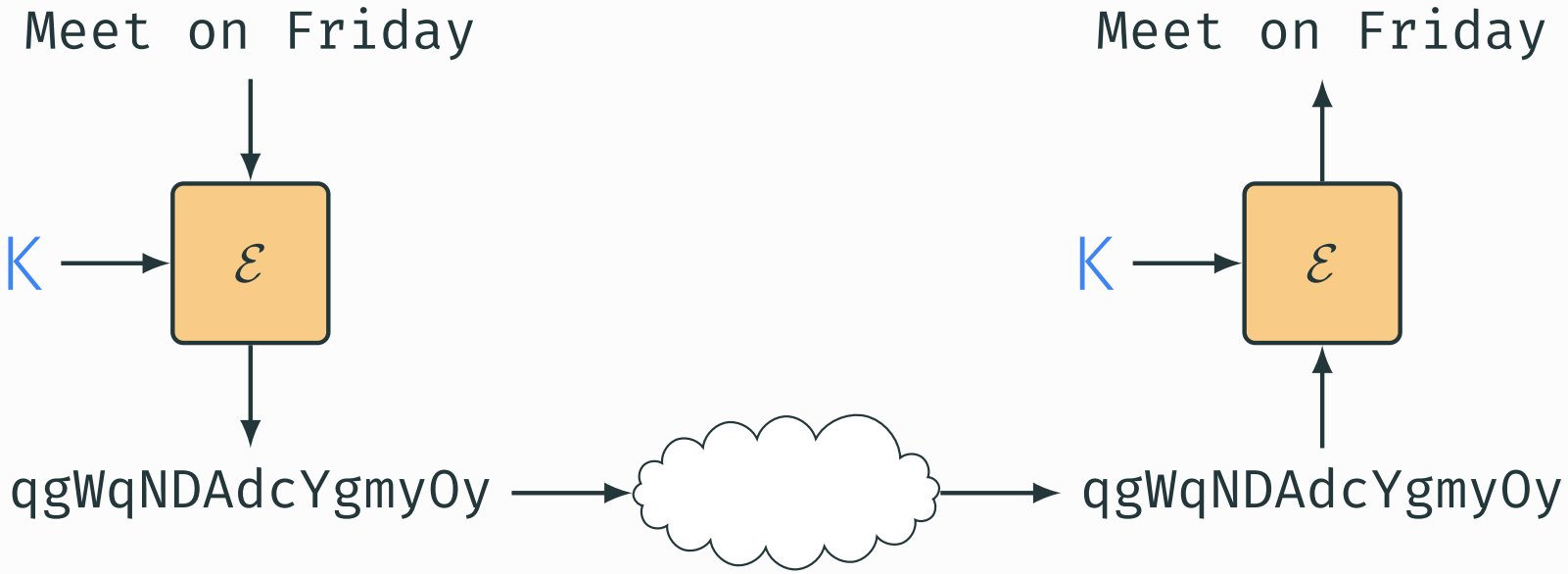


- Bob needs to compute n steps.
- Adversary needs to compute mn.

Symmetric Key Cryptography

Note

We need a shared secret between the parties.



Symmetric Key Cryptography

The adversary

- Eavesdrop on communication
- Modify transmission
- Delete/Insert messages
- ...

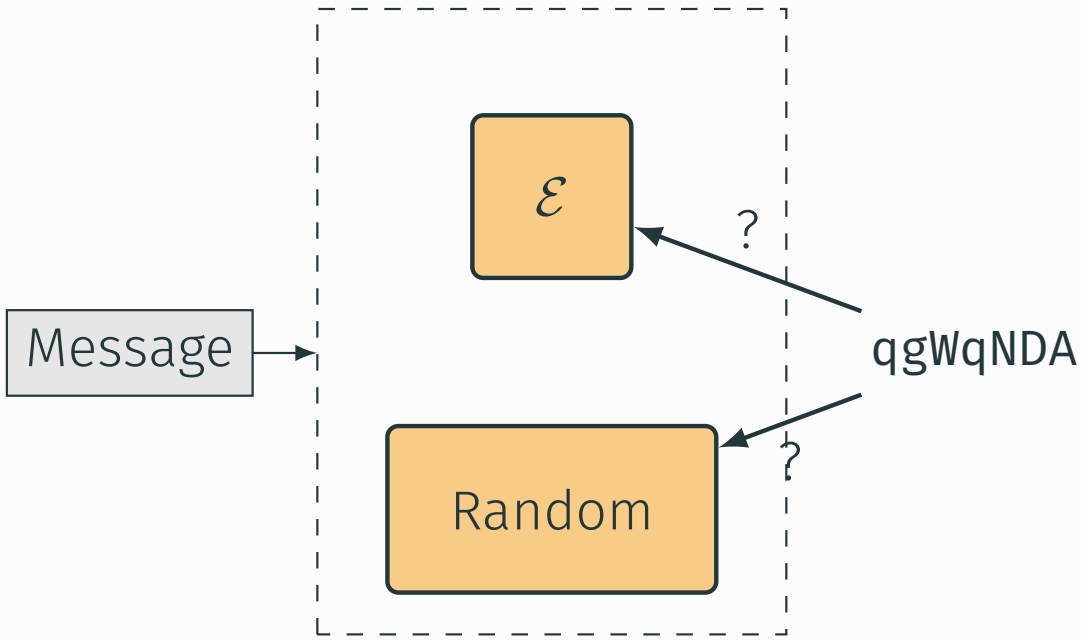
...but is bound in

- Computational power
- Available memory
- Time
- Data

Symmetric Key Cryptography

Goals of the attacker

- Decrypt a ciphertext
- Forge a signature
- Recover the secret key
- Distinguish output
- ...



Symmetric Key Cryptography

How do we achieve security for an algorithm?

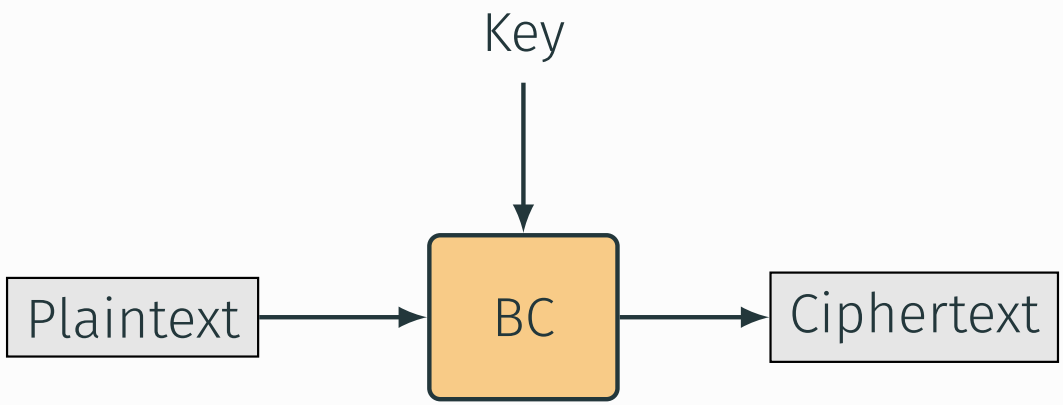
- Reduce security to a *hard* problem.
- Make it secure against all known attacks.

Note

We can **not** proof security for a primitive.

Encryption

Block Ciphers

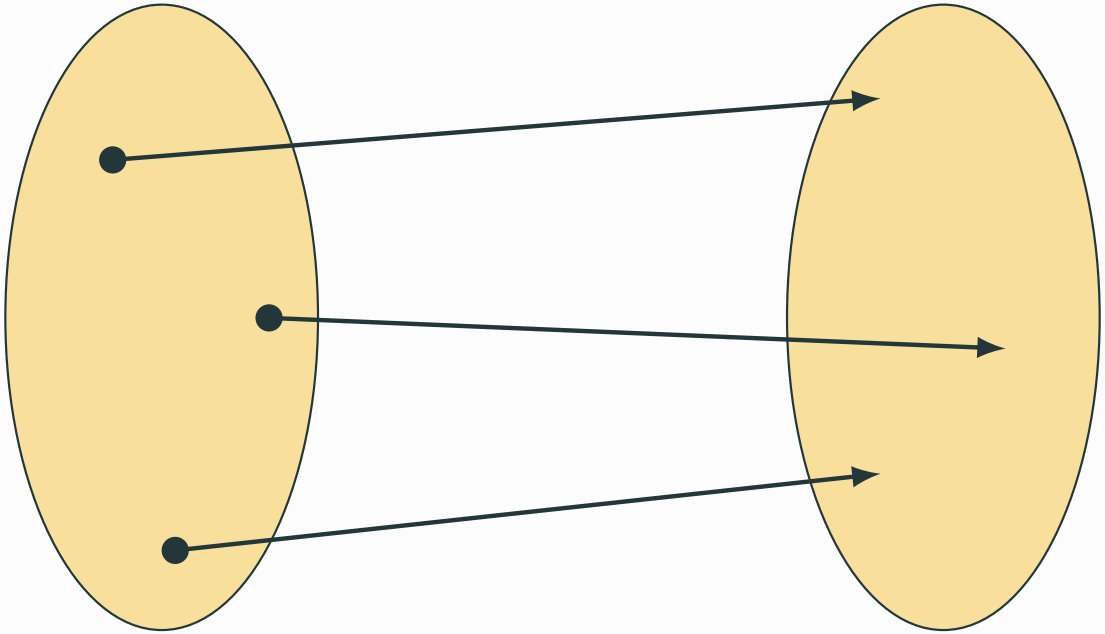


- Encrypts blocks of fixed size n with a key of size k.
- Requires a mode to encrypt arbitrary messages.

Block cipher is not an encryption scheme

Symmetric Key

Ideal Block Cipher



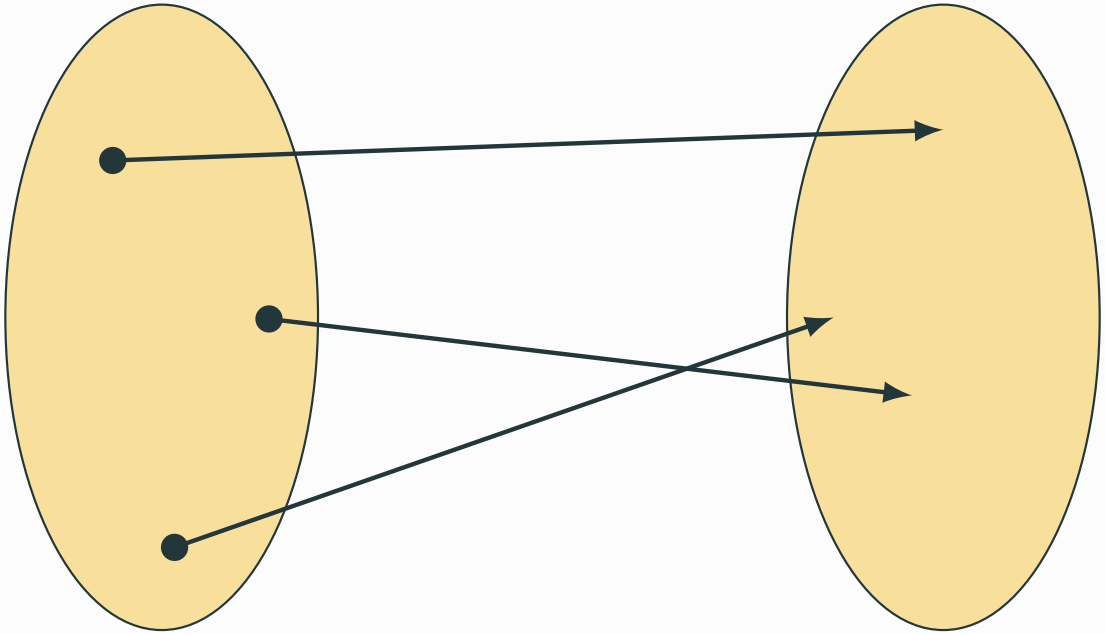
Plaintexts

Ciphertexts

$K = \texttt{101010111010...}$

Symmetric Key

Ideal Block Cipher



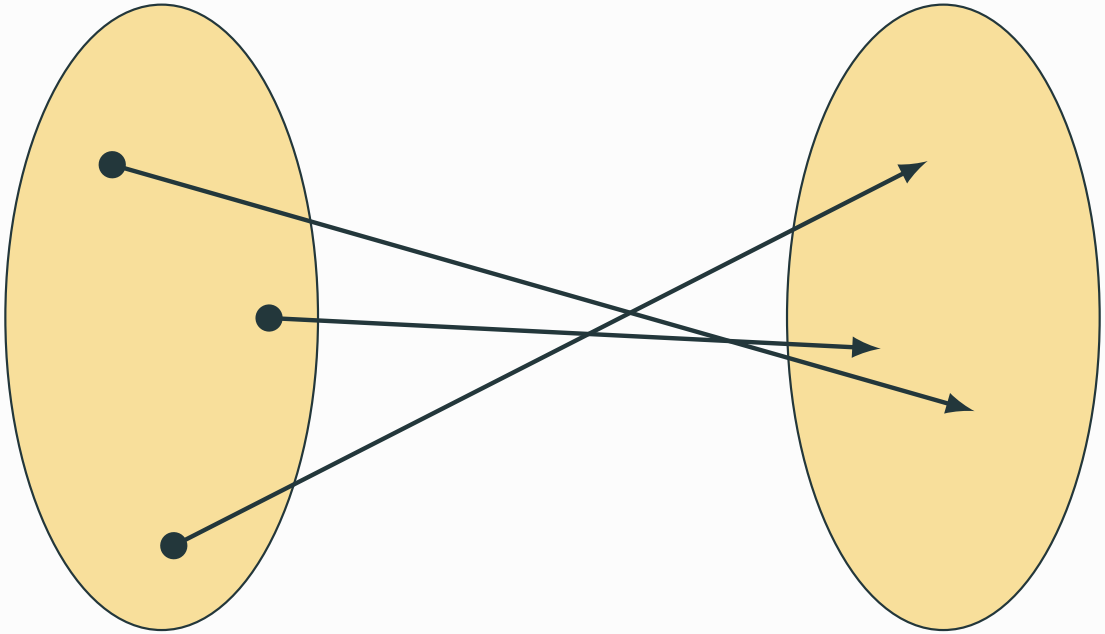
Plaintexts

Ciphertexts

$K = \texttt{001111110000...}$

Symmetric Key

Ideal Block Cipher



Plaintexts

Ciphertexts

$K = \texttt{111111001000...}$

Block Ciphers

A block cipher can be seen as a family of 2^k n-bit bijections.

Problem

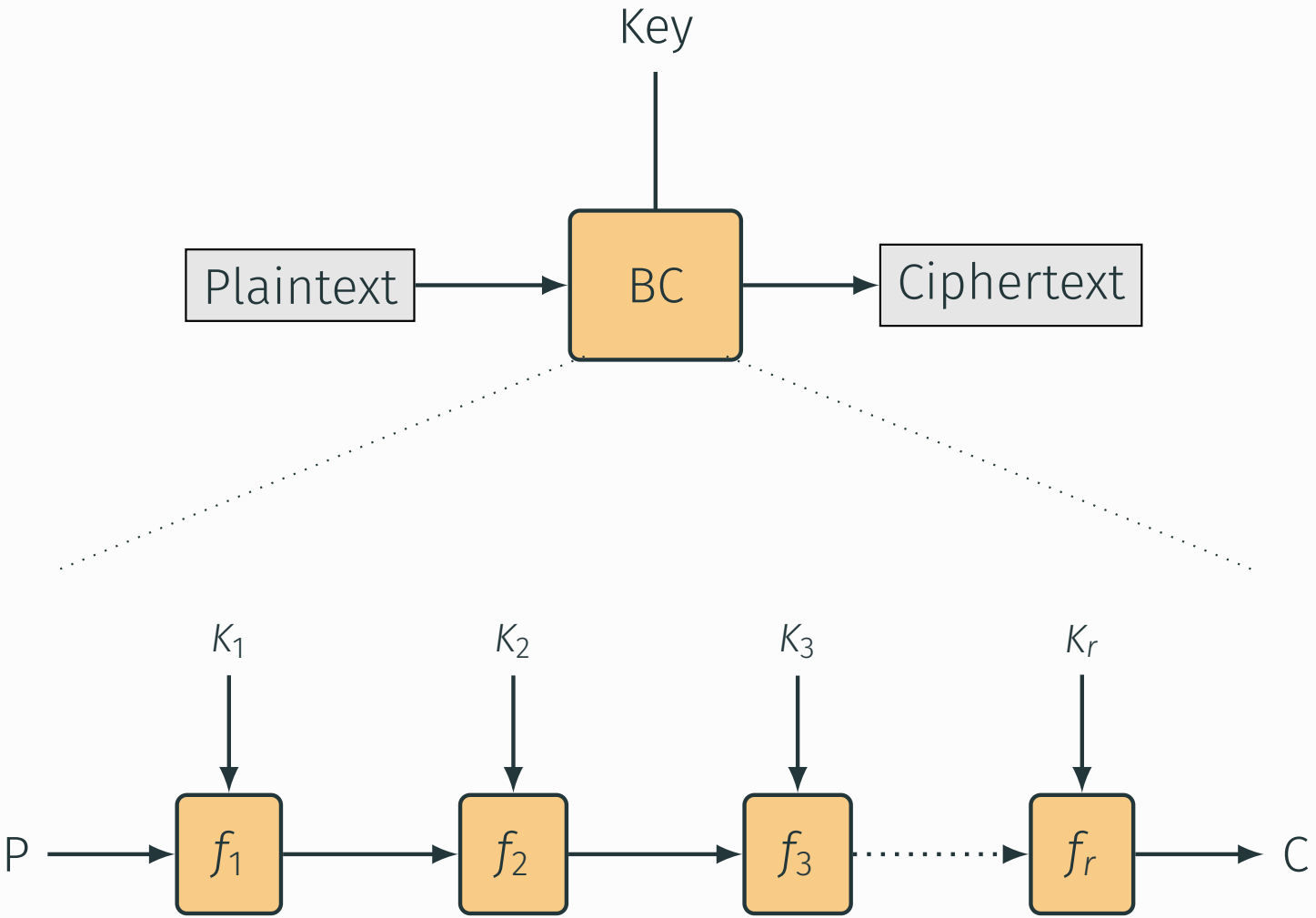
There are $2^n!$ bijections, we ideally want to choose 2^k uniformly at random.

Goal

We need something **efficient** to mimic this behaviour.

Block Ciphers

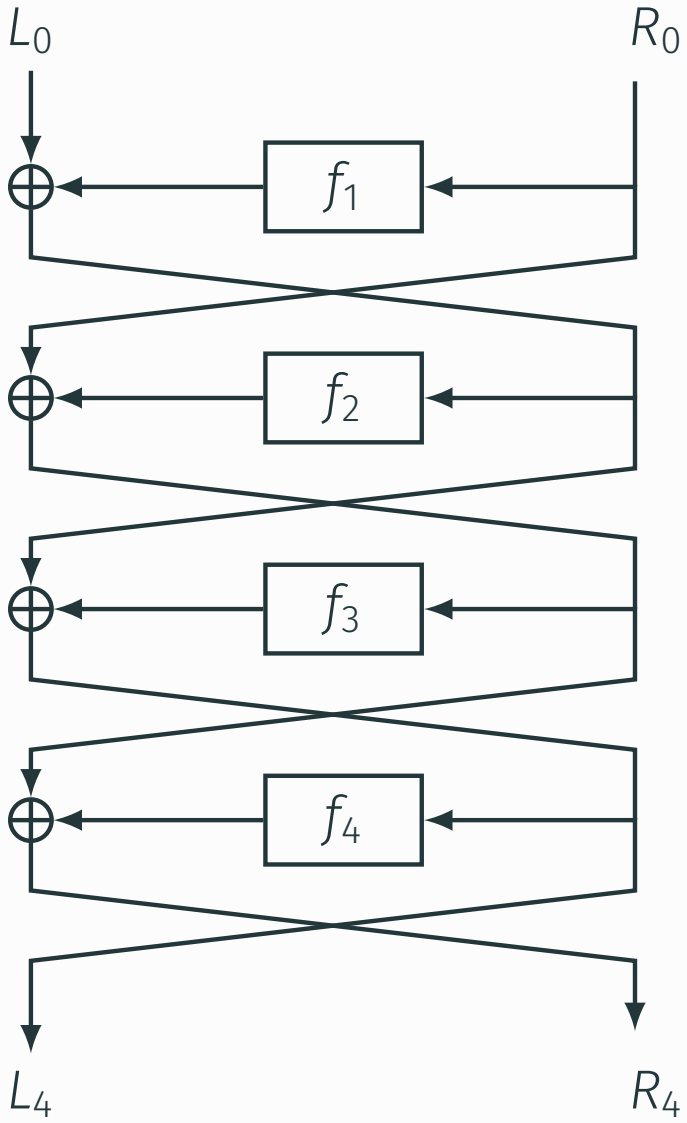
Iterated construction



Symmetric Key Cryptography

The Data Encryption Standard

- Developed in 1970s at IBM.
- Feistel Network with 16 rounds.
- Encrypts 64-bit blocks with 56-bit keys.
- Standardized in 1977.



Symmetric Key Cryptography

The Advanced Encryption Standard (AES)

- Public Competition hosted by NIST (1997-2001)
- Must support block size of 128 bits and key size of 128, 192 and 256 bits.

- CAST-256
- CRYPTON
- DEAL
- DFC
- E2
- FROG
- HPC
- LOKI97
- MAGENTA
- MARS
- RC6
- Rijndael
- SAFER+
- Serpent
- Twofish

Symmetric Key Cryptography

The Advanced Encryption Standard (AES)

- Public Competition hosted by NIST (1997-2001)
- Must support block size of 128 bits and key size of 128, 192 and 256 bits.

- CAST-256
- CRYPTON
- DEAL
- DFC
- E2
- FROG
- HPC
- LOKI97
- MAGENTA
- MARS
- RC6
- Rijndael
- SAFER+
- Serpent
- Twofish

Block Ciphers

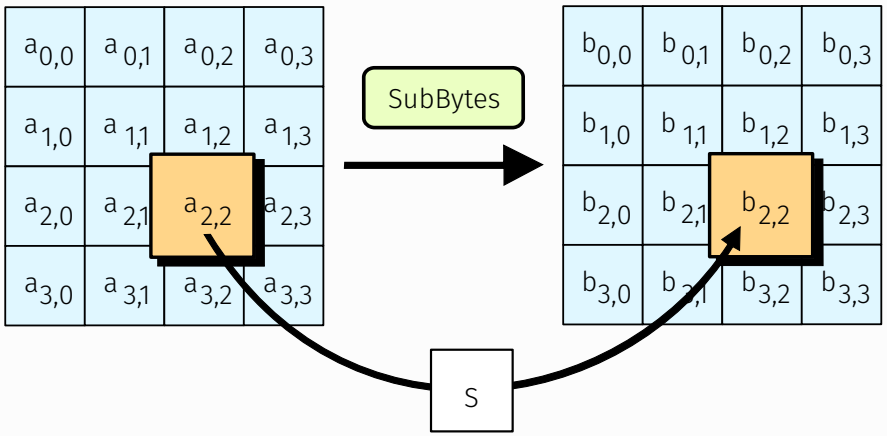
AES/Rijndael

- Blocksize: 128-bit
- Keysize: 128/192/256 bits
- Iterated block cipher with 10/12/14 rounds
- Is part of a wide-range of standards.
- Direct support by instructions in modern CPUs.

Block Ciphers

Update 4×4 state of bytes

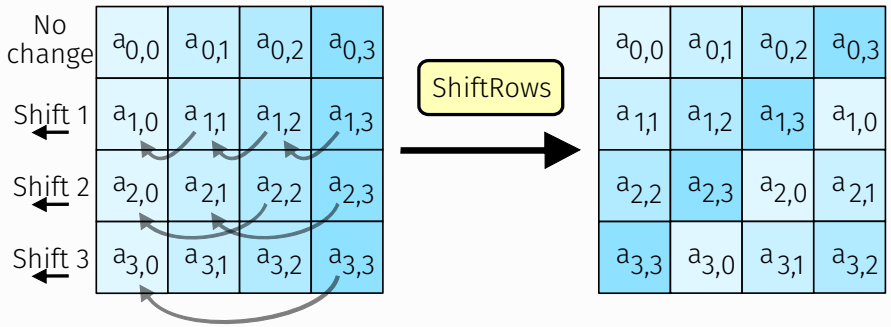
- SubBytes
- ShiftRows
- MixColumns
- AddKey



Block Ciphers

Update 4×4 state of bytes

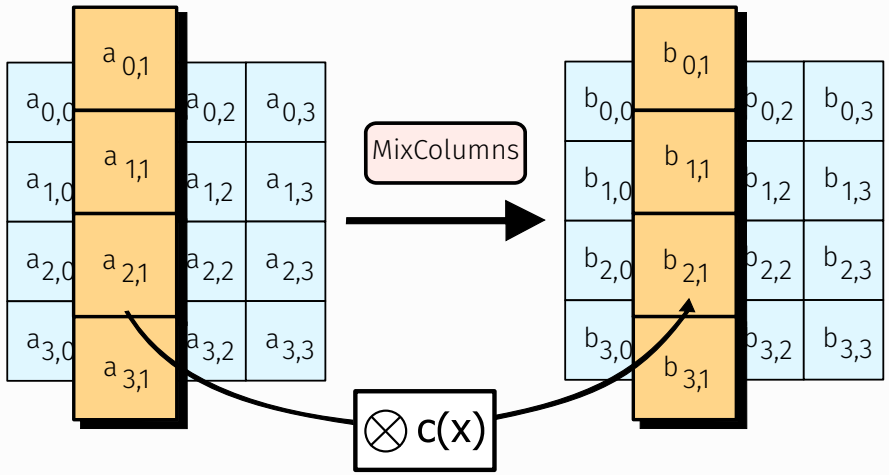
- SubBytes
- ShiftRows
- MixColumns
- AddKey



Block Ciphers

Update 4×4 state of bytes

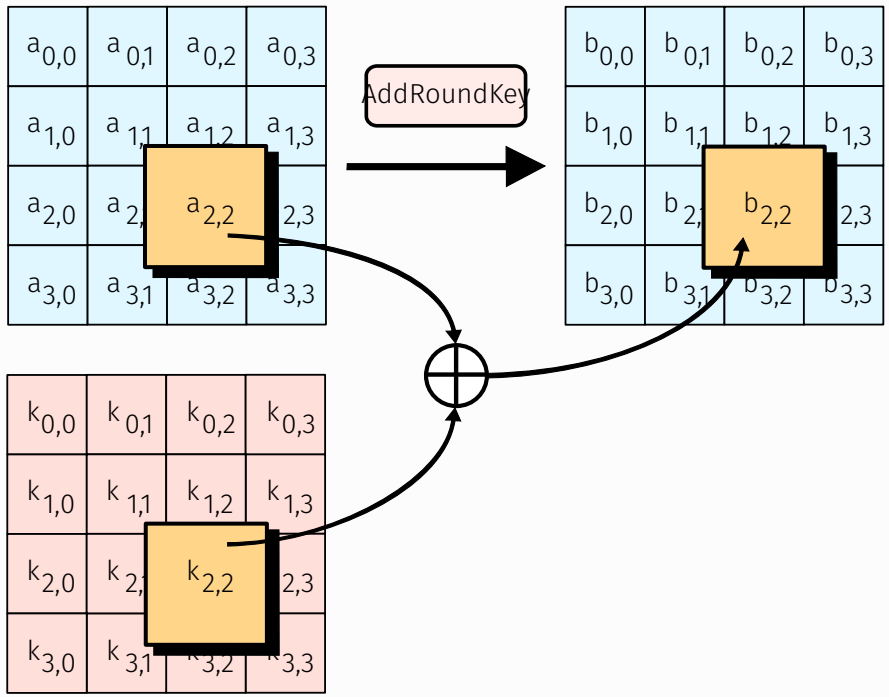
- SubBytes
- ShiftRows
- MixColumns
- AddKey



Block Ciphers

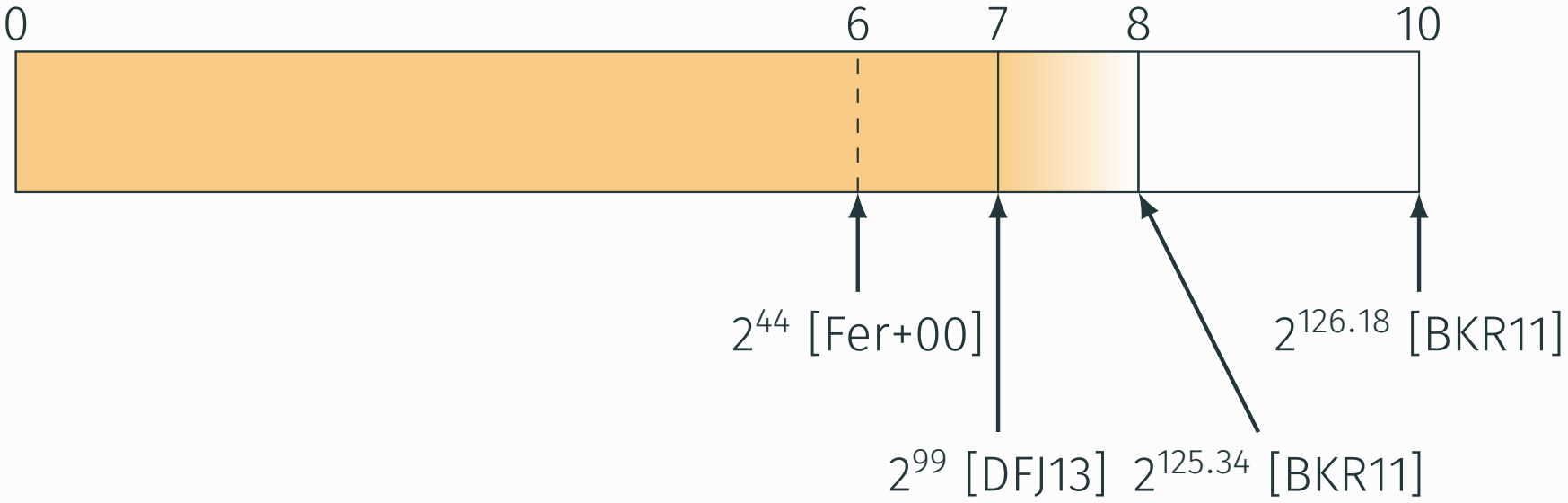
Update 4×4 state of bytes

- SubBytes
- ShiftRows
- MixColumns
- AddKey



Block Ciphers

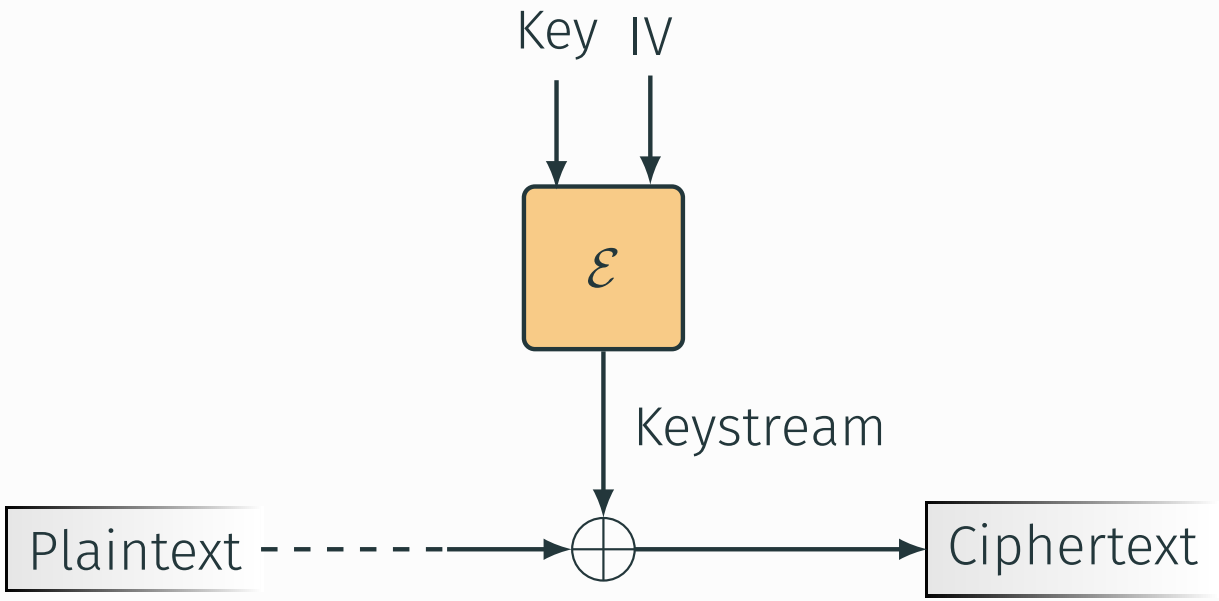
Current state of key recovery attacks for AES-128



There are many more attacks with different trade-offs of time/data/memory.

Stream Ciphers

Stream Ciphers



- Encrypts individual *digits*.
- IV to have multiple key stream for each K
- Requires no padding.
- Often used for low-bandwidth communication.

Stream Ciphers

Widely found in practice

- GSM standard (A5/1, A5/2)
- LTE (SNOW 3G, ZUC)
- Bluetooth (E0)
- TLS protocol (RC4, ChaCha20)

Stream Ciphers

eSTREAM Project (EU)

Goal

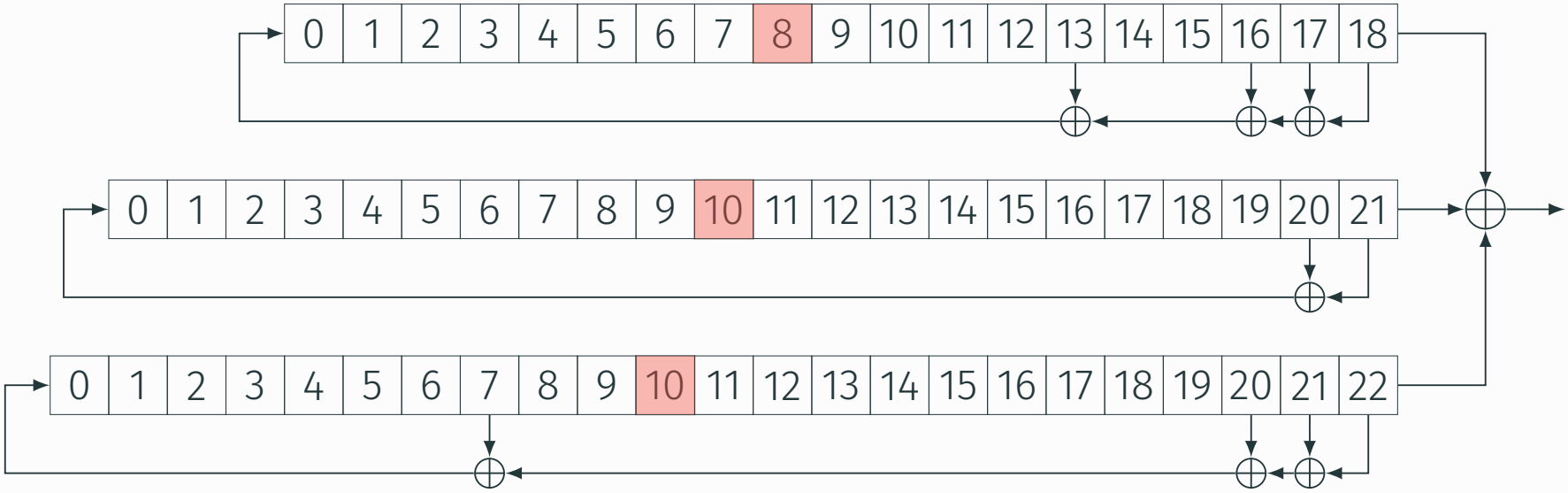
...promote the design of efficient and compact stream ciphers suitable for widespread adoption...

Software	Hardware
HC-128	Grain v1
Rabbit	MICKEY 2.0
Salsa20/12	Trivium
SOSEMANUK	

Stream Ciphers

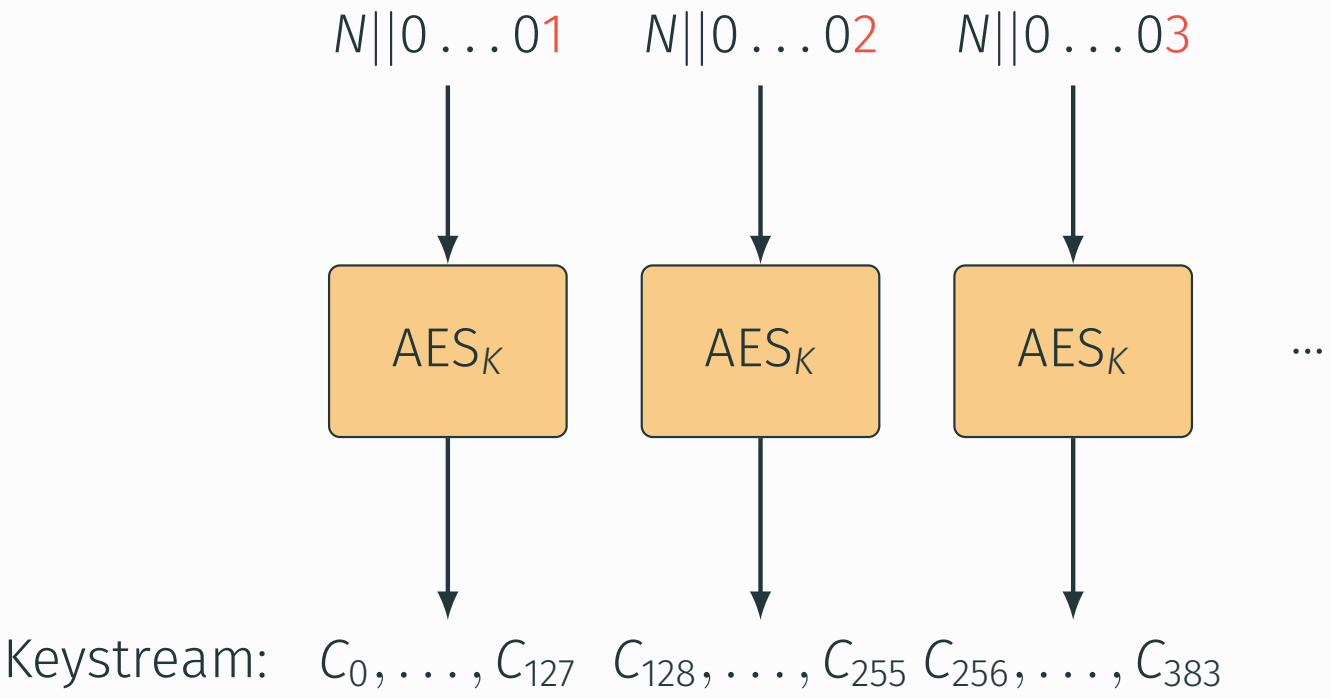
LFSR-based Constructions, e.g. A5/1

- Load IV and Key in registers.
- Shift registers depending on values in ■.
- Produces 1-bit output in each iteration.



Stream Ciphers

Counter Mode (CTR)



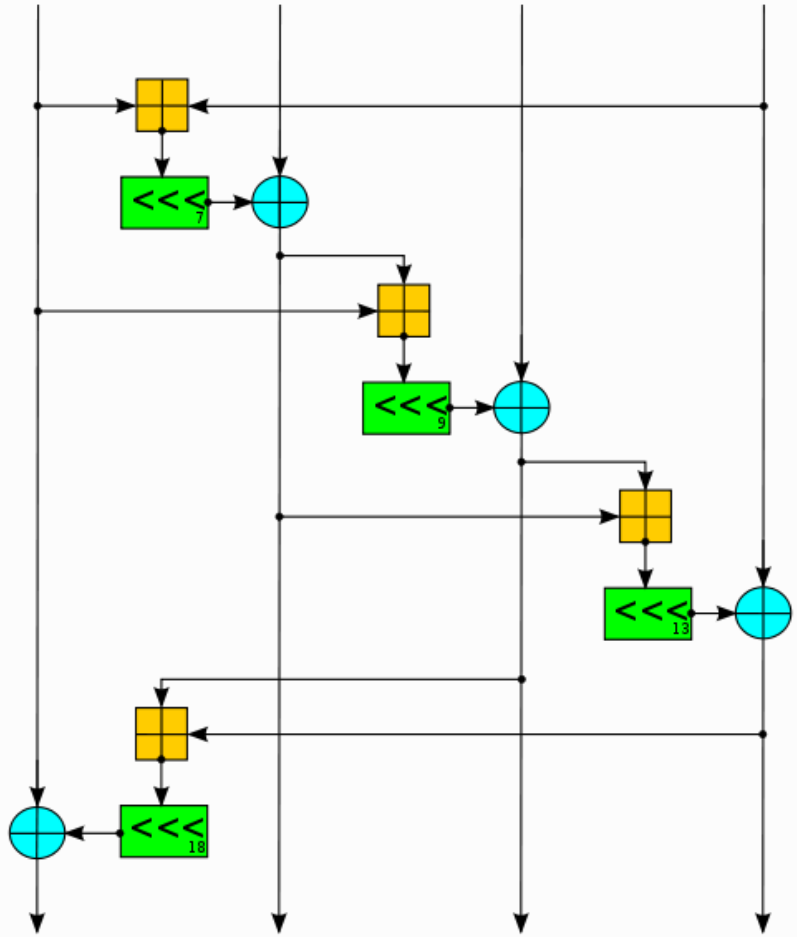
Note

Reusing nonce and counter gives same keystream.

Stream Ciphers

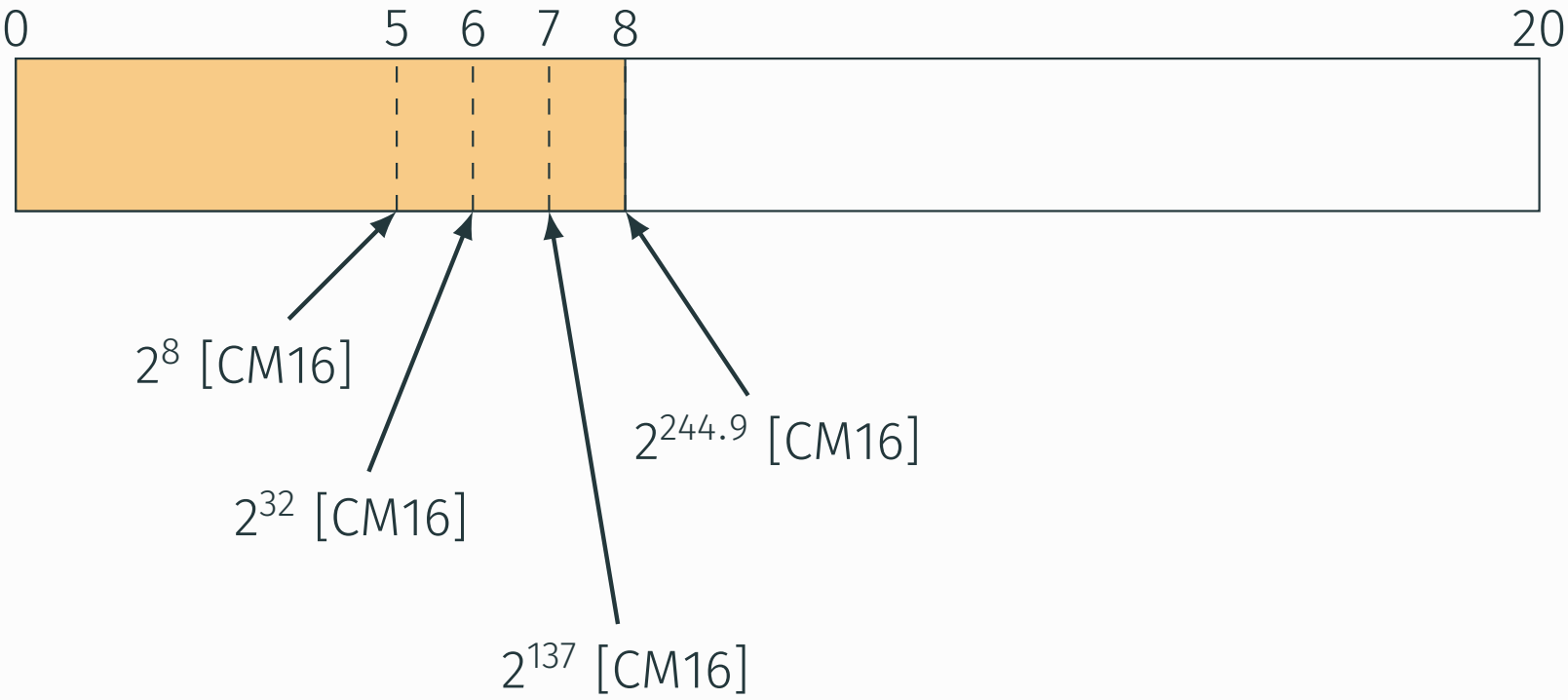
Salsa20 / ChaCha20

- ARX-based design
- 512-bit state
- Uses 256-bit key
- 20 rounds
- Fast in software
- ChaCha20-Poly1305 in TLS



Stream Ciphers

Current state of key recovery attacks for Salsa20

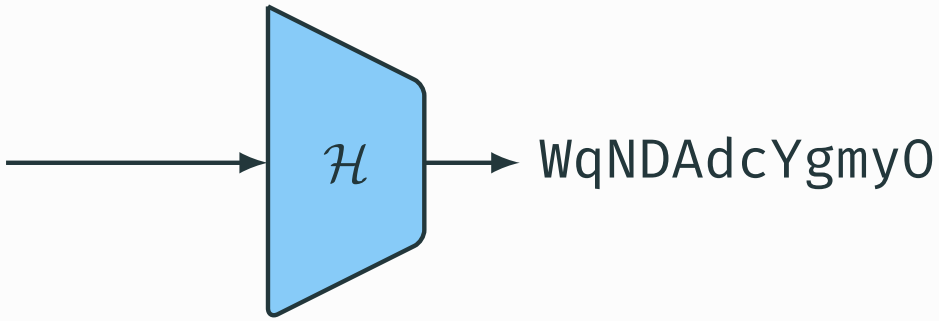


For ChaCha typically one round less.

Cryptographic Hash Functions

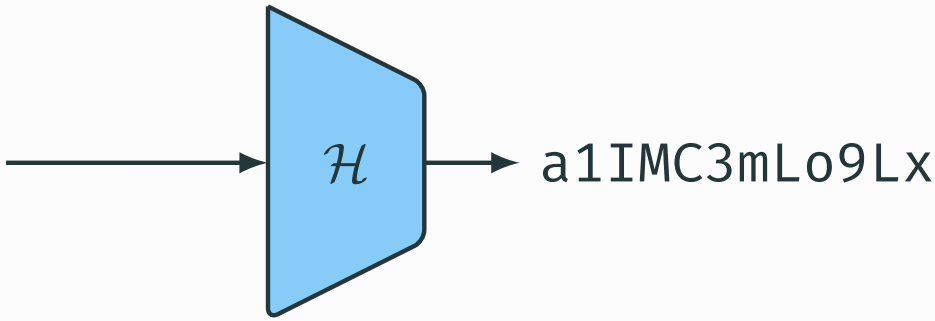
Hash Functions

”There was of course no way of knowing whether you were being watched at any given moment. How often, or on what system, the Thought Police plugged in on any individual wire was guesswork. It was even conceivable that they watched everybody all the time. But at any rate they could plug in your wire whenever they wanted to. You had to live – did live, from habit that became instinct – in the assumption that every sound you made was overheard, and, except in darkness, every movement scrutinized.”



Hash Functions

”There was of course no way of knowing whether you were being watched at any given moment. How often, or on what system, the Thought Police plugged in on any individual wire was guesswork. It was even conceivable that they watched everybody all the time. But at any rate they could plug in your wire whenever they wanted to. You had to live – did live, from habit that became instinct – in the assumption that every noise you made was overheard, and, except in darkness, every movement scrutinized.”



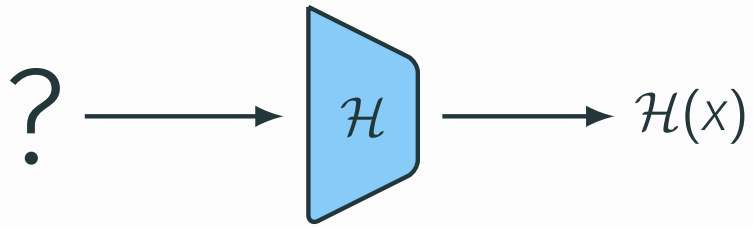
Hash Functions

Applications

- Integrity Check
- Digital Signature Schemes (this afternoon)
- Password Hashing (`https://password-hashing.net/`)
- Message Authentication
- Commitment Schemes
- ...

Hash Functions

Preimage Resistance

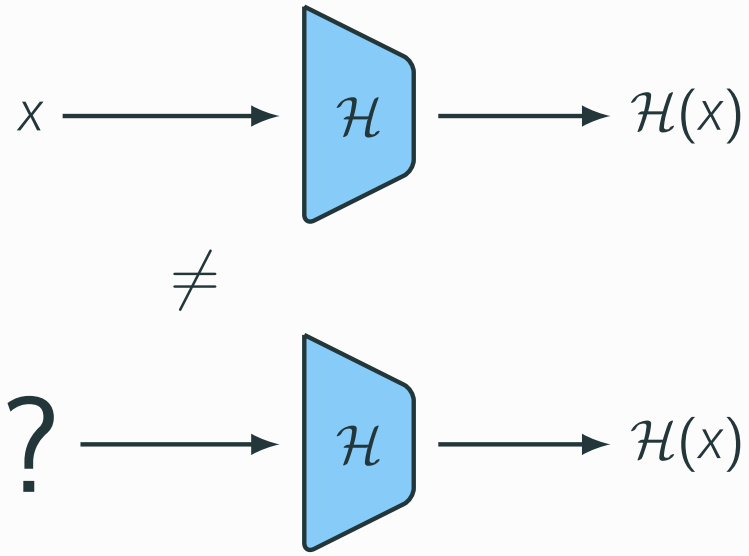


For n-bit output size

- Best attack: 2^n

Hash Functions

Second-Preimage Resistance

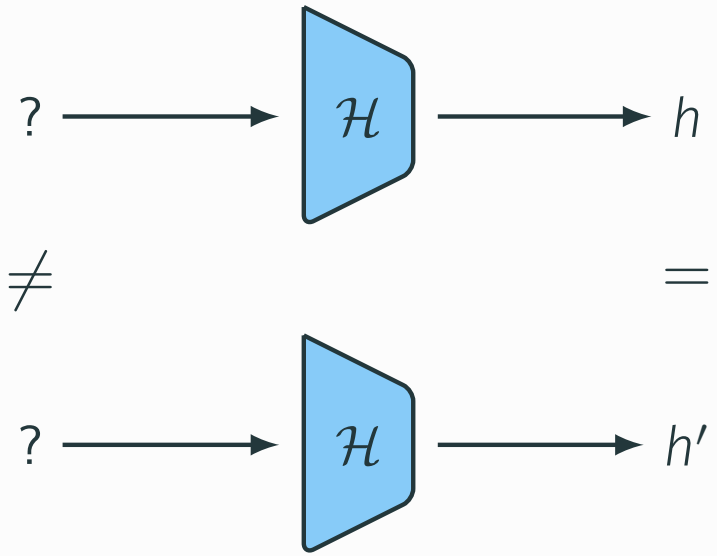


For n-bit output size

- Best attack: 2^n

Hash Functions

Collision Resistance



For n-bit output size

- Best attack: $2^{n/2}$

Hash Functions

Hardness of finding collision vs. preimages in practice

Algorithm	Year	n	Collision	Preimage
MD4	1990	128	< 1 sec	
MD5	1992	128	< 1 sec	
SHA-1	1995	160	2^{63}	
SHA-256	2001	256	$2^{65.5}$ 31/64 rounds	

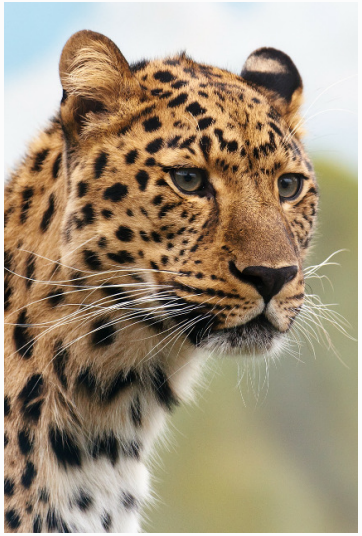
Hash Functions

Hardness of finding collision vs. preimages in practice

Algorithm	Year	n	Collision	Preimage
MD4	1990	128	< 1 sec	$2^{78.4}$ [Guo+10]
MD5	1992	128	< 1 sec	$2^{123.4}$ [SA09]
SHA-1	1995	160	2^{63}	$2^{151.1}$ [KK12] 57/80 rounds
SHA-256	2001	256	$2^{65.5}$ 31/64 rounds	$2^{255.5}$ [KRS12] 45/64 rounds

Hash Functions

Requirements for security and performance can vary on application.



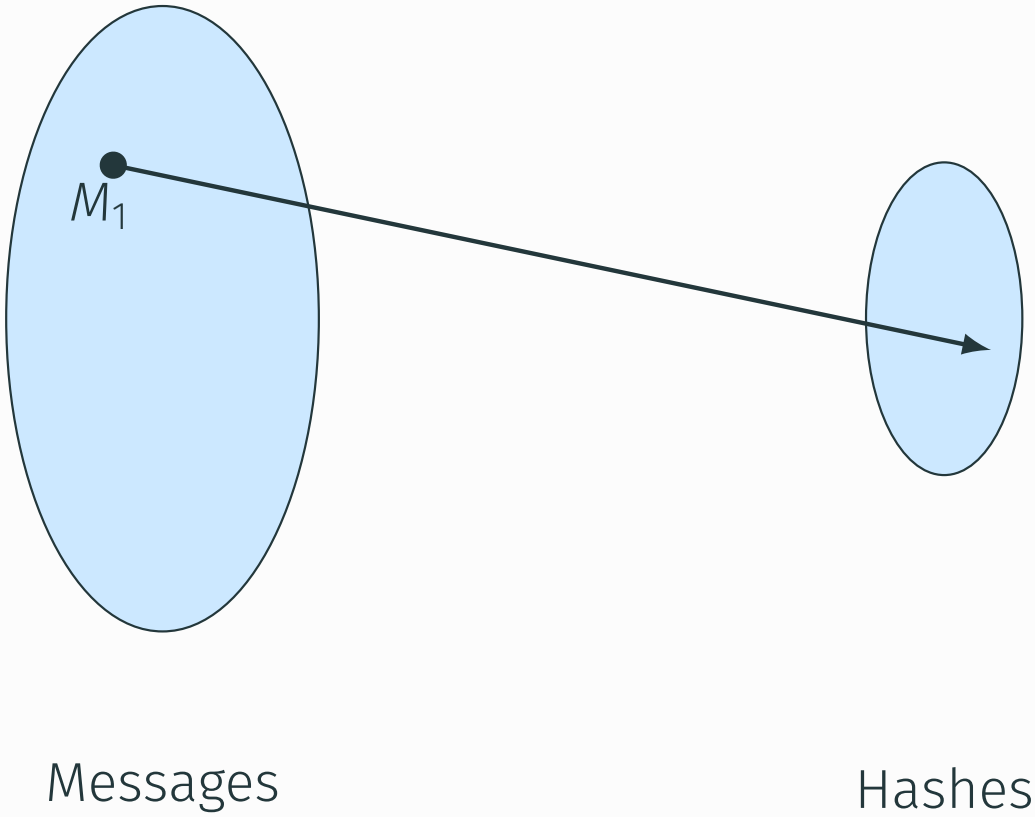
Performance on long/short messages.

Password Hashing should be slow!

Collision resistance not required!

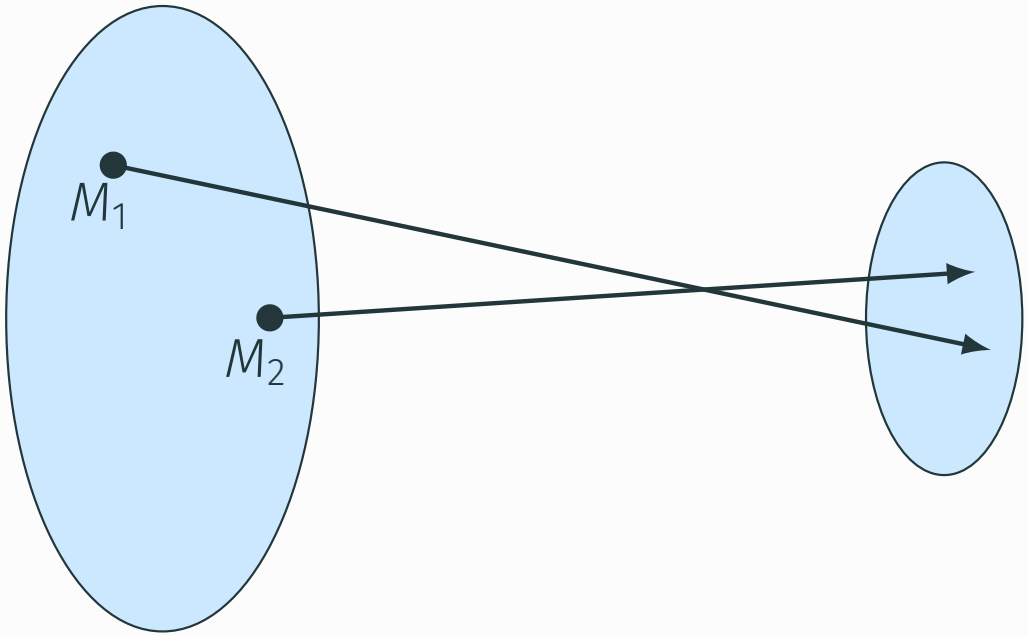
Hash Functions

Ideal Hash Function



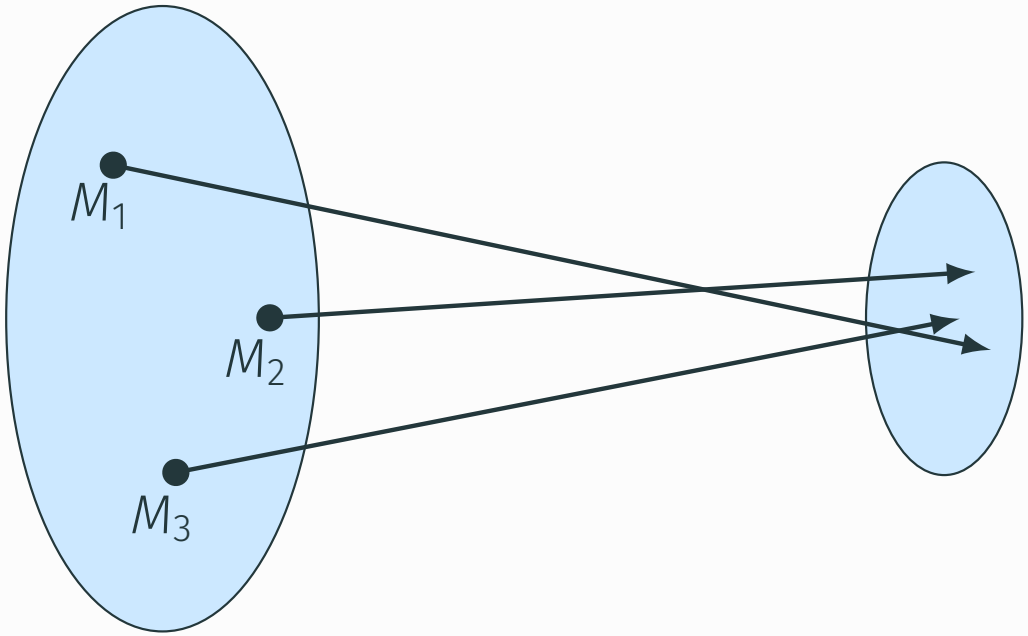
Hash Functions

Ideal Hash Function



Hash Functions

Ideal Hash Function



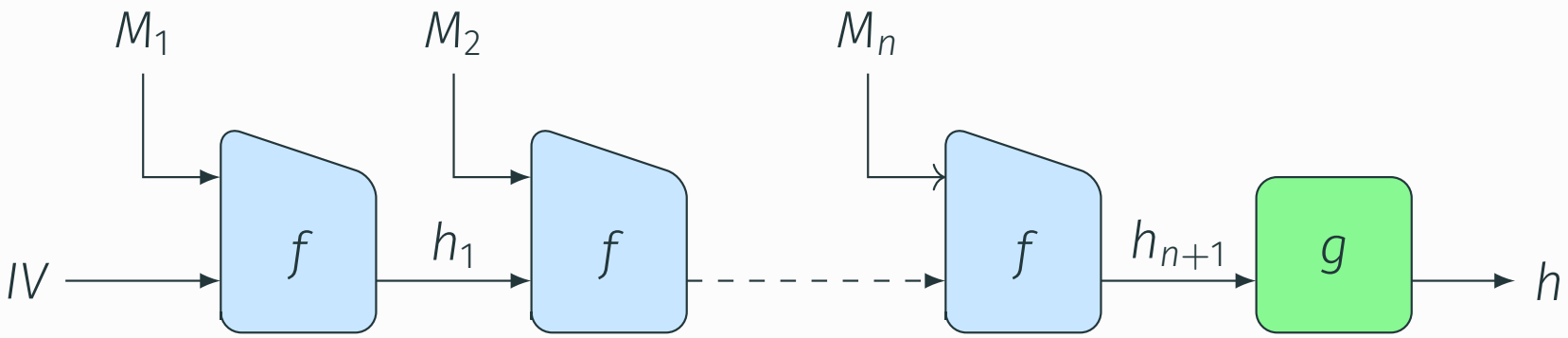
Messages

Hashes

Hash Functions

How to construct a hash function?

- Merkle-Damgård with compression function (SHA-1, SHA-2)



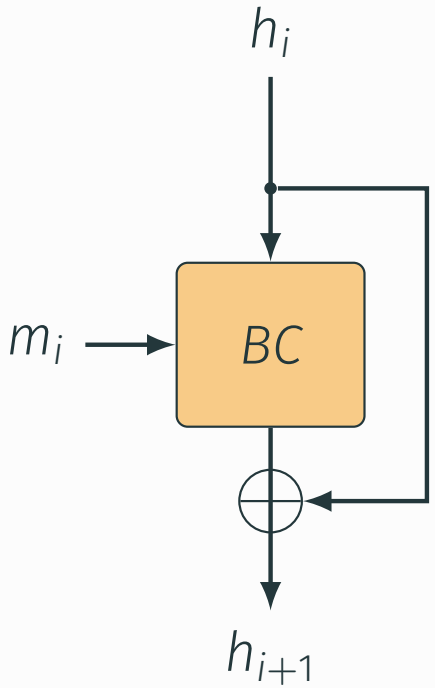
Problem

How do we construct the compression function?

Hash Functions

How to construct a hash function?

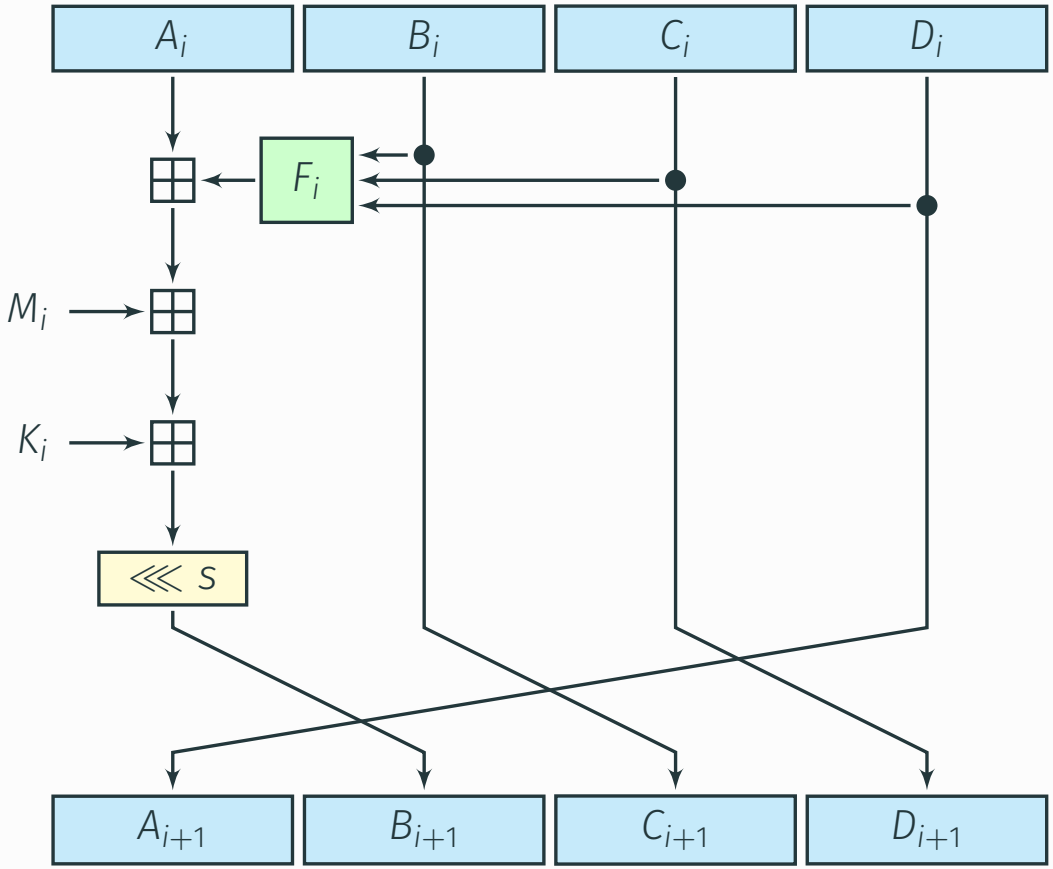
- Merkle-Damgård with compression function (SHA-1, SHA-2)



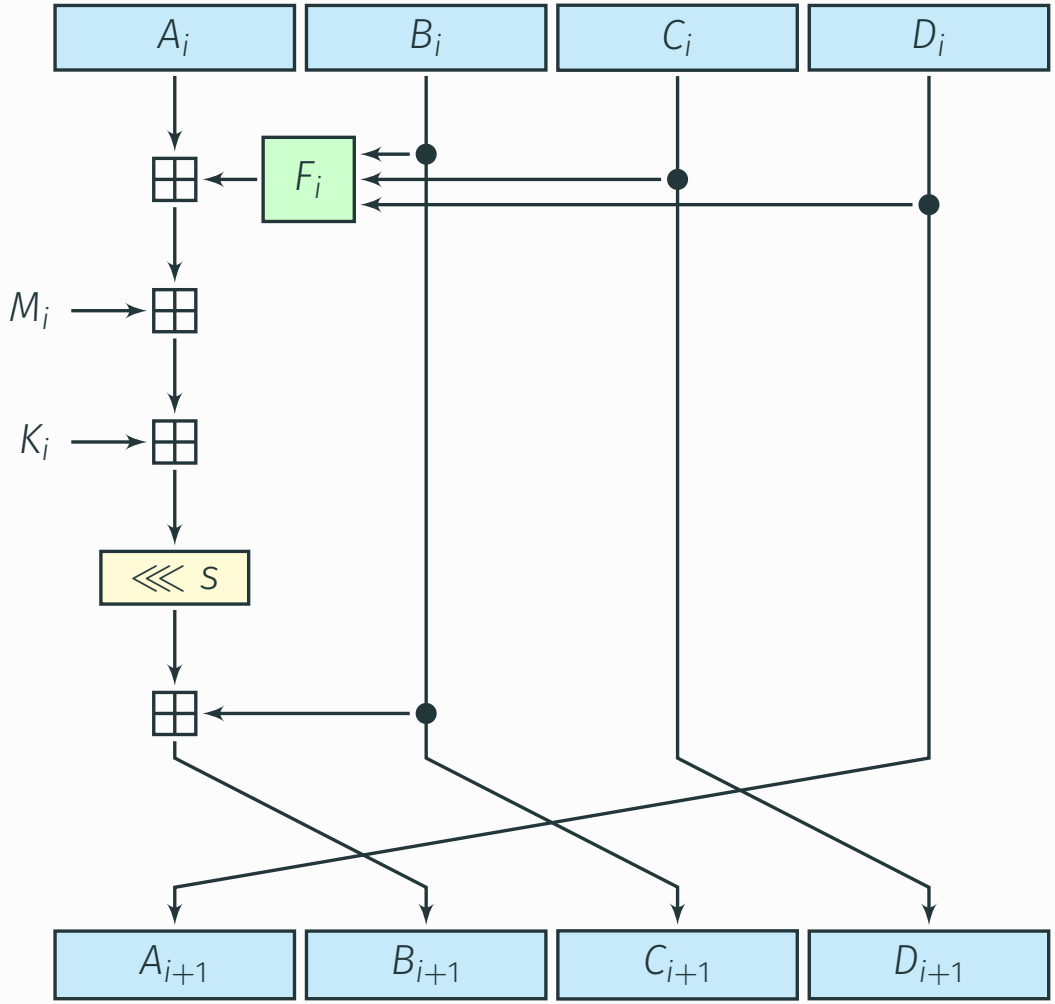
Solution

Use a block cipher! … but often state is too small.

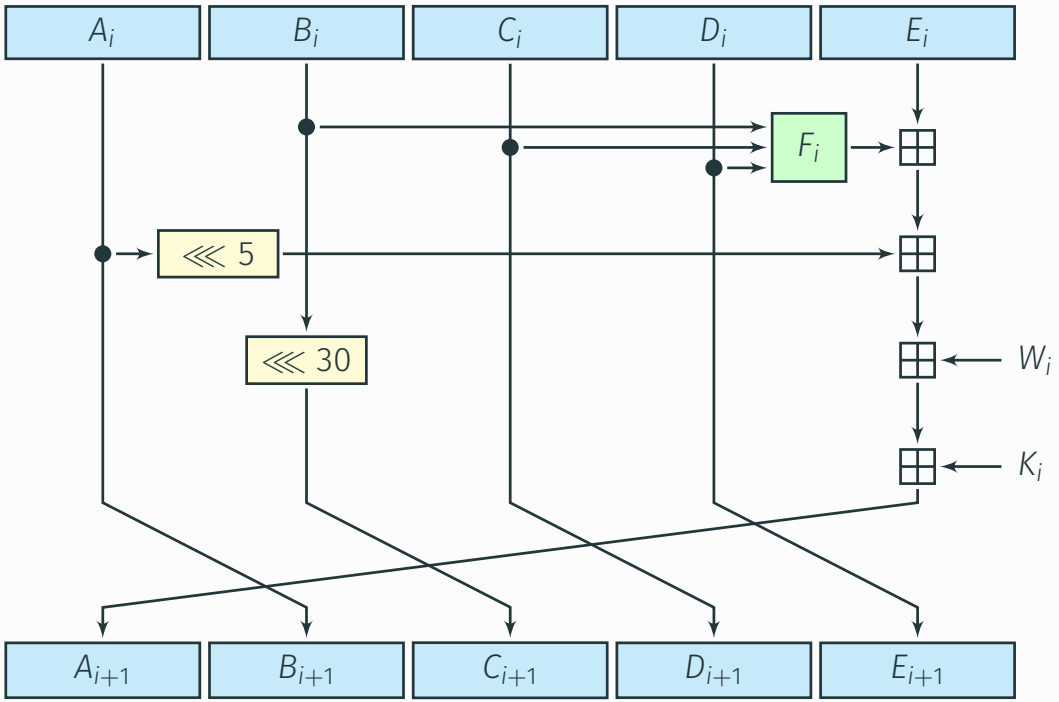
Compression Function Design (MD4)



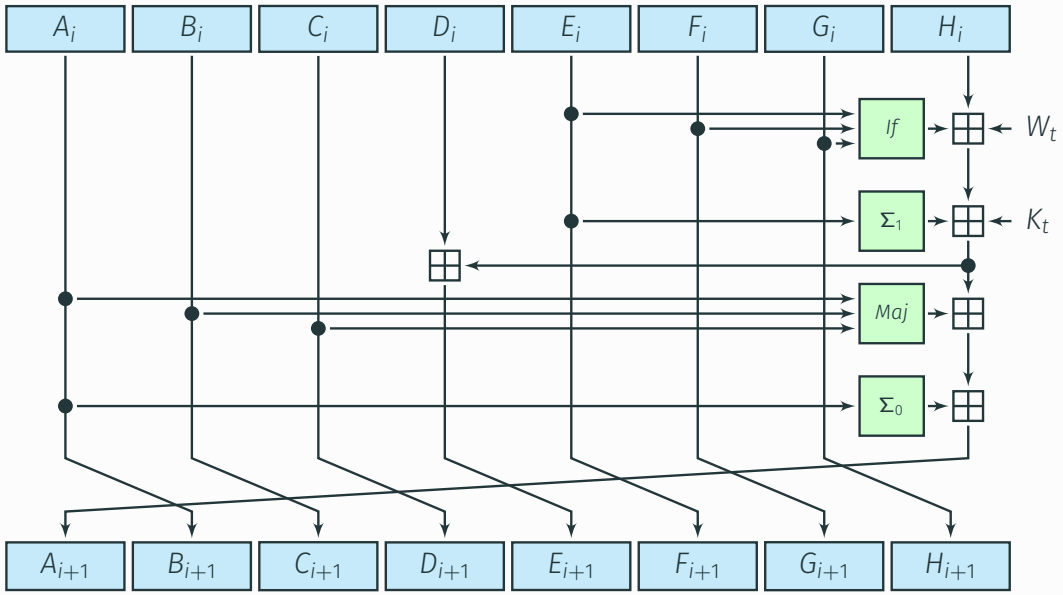
Compression Function Design (MD5)



Compression Function Design (SHA-1)



Compression Function Design (SHA-2)



Hash Functions

The 2005 Hash Crisis

- Wang and Yu show that MD5 is not collision resistant [WY05]...
- ... and SHA-1 isn't either [WYY05].
- Concerns that SHA-2 will also fail.

Hash Functions

The SHA-3 Competition

- Public Competition to find a new standard SHA-3.
- From 2007 to 2012

- Abacus
- ARIRANG
- AURORA
- Blake
- Blender
- Blue Midnight Wish
- Boole
- Cheetah
- CHI
- CRUNCH
- CubeHash
- DCH
- Dynamic SHA
- Dynamic SHA2
- ECHO
- ECOH
- Edon-R
- EnRUPT
- ESSENCE
- FSB
- Fugue
- Grøstl
- Hamsi
- HASH 2X
- JH
- Keccak
- Khichidi-1
- LANE
- Lesamnta
- Luffa
- LUX
- Maraca
- MCSSHA-3
- MD6
- MeshHash
- NaSHA
- NKS2D
- Ponic
- SANDstorm
- Sarmal
- Sgàil
- Shabal
- SHAMATA
- SHAvite-3
- SIMD
- Skein
- Spectral Hash
- StreamHash
- SWIFFTX
- Tangle
- TIB3
- Twister
- Vortex
- WaMM
- Waterfall
- ZK-Crypt

Hash Functions

The SHA-3 Competition

- Public Competition to find a new standard SHA-3.
- From 2007 to 2012

- Abacus
- ARIRANG
- AURORA
- **Blake**
- Blender
- Blue Midnight Wish
- Boole
- Cheetah
- CHI
- CRUNCH
- CubeHash
- DCH
- Dynamic SHA
- Dynamic SHA2
- ECHO
- ECOH
- Edon-R
- EnRUPT
- ESSENCE
- FSB
- Fugue
- **Grøstl**
- Hamsi
- HASH 2X
- **JH**
- **Keccak**
- Khichidi-1
- LANE
- Lesamnta
- Luffa
- LUX
- Maraca
- MCSSHA-3
- MD6
- MeshHash
- NaSHA
- NKS2D
- Ponic
- SANDstorm
- Sarmal
- Sgàil
- Shabal
- SHAMATA
- SHAvite-3
- SIMD
- **Skein**
- Spectral Hash
- StreamHash
- SWIFFTX
- Tangle
- TIB3
- Twister
- Vortex
- WaMM
- Waterfall
- ZK-Crypt

Hash Functions

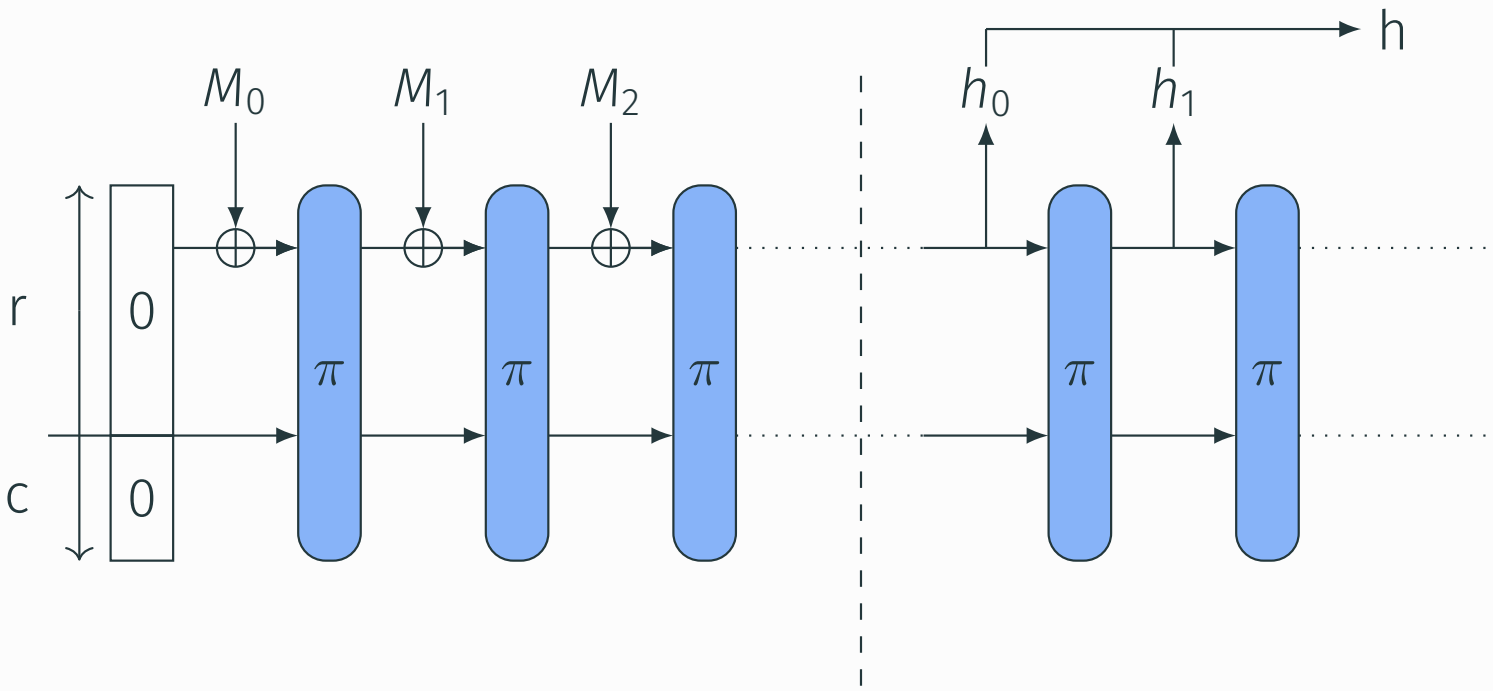
The SHA-3 Competition

- Public Competition to find a new standard SHA-3.
- From 2007 to 2012

- Abacus
- ARIRANG
- AURORA
- Blake
- Blender
- Blue Midnight Wish
- Boole
- Cheetah
- CHI
- CRUNCH
- CubeHash
- DCH
- Dynamic SHA
- Dynamic SHA2
- ECHO
- ECOH
- Edon-R
- EnRUPT
- ESSENCE
- FSB
- Fugue
- Grøstl
- Hamsi
- HASH 2X
- JH
- **Keccak**
- Khichidi-1
- LANE
- Lesamnta
- Luffa
- LUX
- Maraca
- MCSSHA-3
- MD6
- MeshHash
- NaSHA
- NKS2D
- Ponic
- SANDstorm
- Sarmal
- Sgàil
- Shabal
- SHAMATA
- SHAvite-3
- SIMD
- Skein
- Spectral Hash
- StreamHash
- SWIFFTX
- Tangle
- TIB3
- Twister
- Vortex
- WaMM
- Waterfall
- ZK-Crypt

Hash Functions

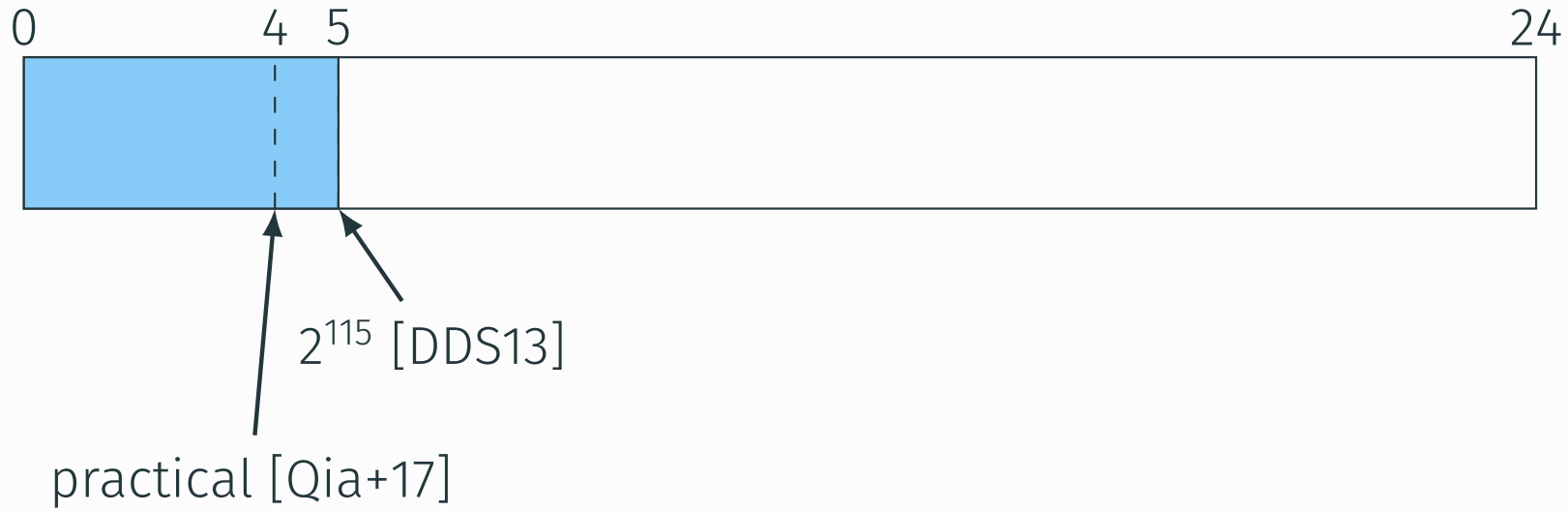
SHA-3 Winner Keccak



- Based on the sponge construction.
- Uses 1600-bit permutation π.
- Parameters: rate r and capacity c.
- Security claim of $2^{c/2}$.

Hash Functions

SHA3-256 ($c = 512$) collision resistance



Practical Attacks for $c = 160$[1]:

- Collisions for 6 rounds
- Preimages for 4 rounds

[1] http://keccak.noekeon.org/crunchy_contest.html

Hash Functions

What should you use now?

> *"We don't need another slow, secure hash function—we've already got SHA-2."*
> —Adam Langley, Mar. 2017[2]

SHA-3 standard too conservative?[3]

- Use different parameters.
- Tree hashing mode for better performance.
- RFC for Kangaroo12[4]

[2] https://www.imperialviolet.org/2017/05/31/skipsha3.html
[3] http://keccak.noekeon.org/is_sha3_slow.html
[4] https://tools.ietf.org/html/draft-viguier-kangarootwelve-00

Symmetric Key Cryptography

What can we do?

- Encryption
- Authentication (MAC)
- Hashing
- Random Number Generation
- Digital Signature Schemes
- Key Exchange

Questions?

References i

Nadhem J. AlFardan et al. “On the Security of RC4 in TLS”. In: *Proceedings of the 22th USENIX Security Symposium*. 2013, pp. 305–320.

Johannes A. Buchmann, Erik Dahmen, and Andreas Hülsing. “XMSS - A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions”. In: *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*. 2011, pp. 117–129.

Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. “Biclique Cryptanalysis of the Full AES”. In: *Advances in Cryptology - ASIACRYPT 2011*. 2011, pp. 344–371.

Daniel J. Bernstein et al. “SPHINCS: Practical Stateless Hash-Based Signatures”. In: *Advances in Cryptology - EUROCRYPT 2015*. 2015, pp. 368–397.

References ii

Arka Rai Choudhuri and Subhamoy Maitra. “Significantly Improved Multi-bit Differentials for Reduced Round Salsa and ChaCha”. In: *IACR Trans. Symmetric Cryptol.* 2016.2 (2016), pp. 261–287.

Melissa Chase et al. *Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives*. Cryptology ePrint Archive, Report 2017/279. `http://eprint.iacr.org/2017/279`. 2017.

Itai Dinur, Orr Dunkelman, and Adi Shamir. “Collision Attacks on Up to 5 Rounds of SHA-3 Using Generalized Internal Differentials”. In: *Fast Software Encryption - 20th International Workshop, FSE 2013*. 2013, pp. 219–240.

Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. “Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting”. In: *Advances in Cryptology - EUROCRYPT 2013*. 2013, pp. 371–387.

Niels Ferguson et al. “Improved Cryptanalysis of Rijndael”. In: *Fast Software Encryption, 7th International Workshop, FSE 2000*. 2000, pp. 213–230.

References iii

Jian Guo et al. “Advanced Meet-in-the-Middle Preimage Attacks: First Results on Full Tiger, and Improved Results on MD4 and SHA-2”. In: *Advances in Cryptology - ASIACRYPT 2010*. 2010, pp. 56–75.

Simon Knellwolf and Dmitry Khovratovich. “New Preimage Attacks against Reduced SHA-1”. In: *Advances in Cryptology - CRYPTO 2012*. 2012, pp. 367–383.

Dmitry Khovratovich, Christian Rechberger, and Alexandra Savelieva. “Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family”. In: *Fast Software Encryption - 19th International Workshop, FSE 2012*. 2012, pp. 244–263.

Kexin Qiao et al. “New Collision Attacks on Round-Reduced Keccak”. In: *Advances in Cryptology - EUROCRYPT 2017*. 2017, pp. 216–243.

Yu Sasaki and Kazumaro Aoki. “Finding Preimages in Full MD5 Faster Than Exhaustive Search”. In: *Advances in Cryptology - EUROCRYPT 2009*. 2009, pp. 134–152.

References iv

Marc Stevens et al. “Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate”. In: *Advances in Cryptology - CRYPTO 2009*. 2009, pp. 55–69.

Xiaoyun Wang and Hongbo Yu. “How to Break MD5 and Other Hash Functions”. In: *Advances in Cryptology - EUROCRYPT 2005*. 2005, pp. 19–35.

Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. “Finding Collisions in the Full SHA-1”. In: *Advances in Cryptology - CRYPTO 2005*. 2005, pp. 17–36.