PPAE: Parallelizable Permutation-based Authenticated Encryption

Alex Biryukov and Dmitry Khovratovich

University of Luxembourg

13 August 2013

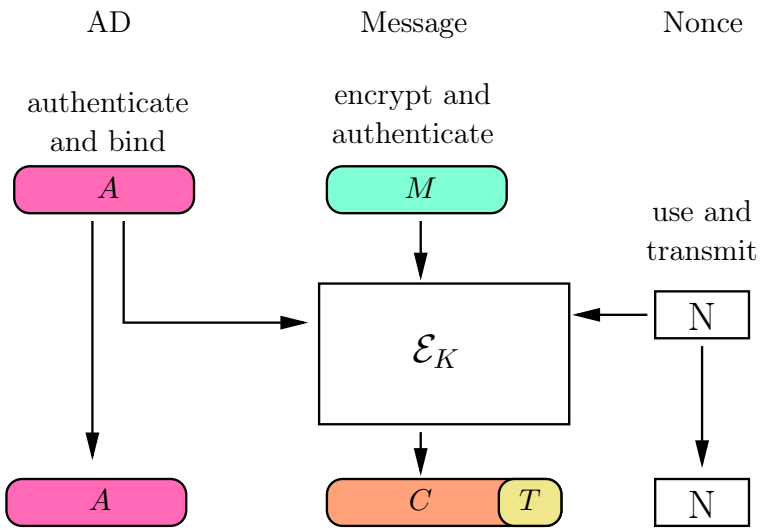
Authenticated encryption with associated data

Encryption:

$$\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M} \to \mathcal{C}$$

Decryption:

$$\mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}.$$



Confidentiality:

- Ciphertexts indistinguishable from random strings;

Data integrity:

- Most of seemingly valid ciphertexts decrypt to $\perp$.

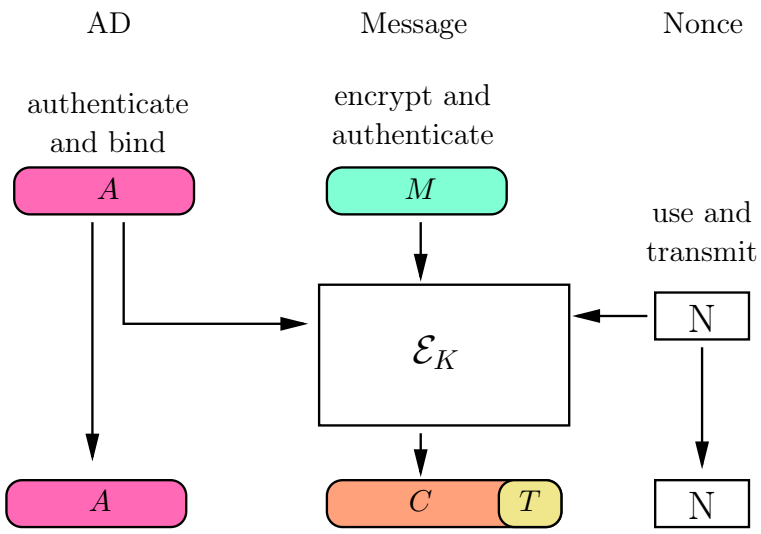
Desirable features

Non-exhaustive list of authenticated encryption features:

- Nonce- or IV-based;
- Parallelizability;
- Incremental tags;
- Security proof;
- Patent-free;
- Reasonable performance;
- Compact implementation;
- Variable key/nonce/tag length.

What we also want

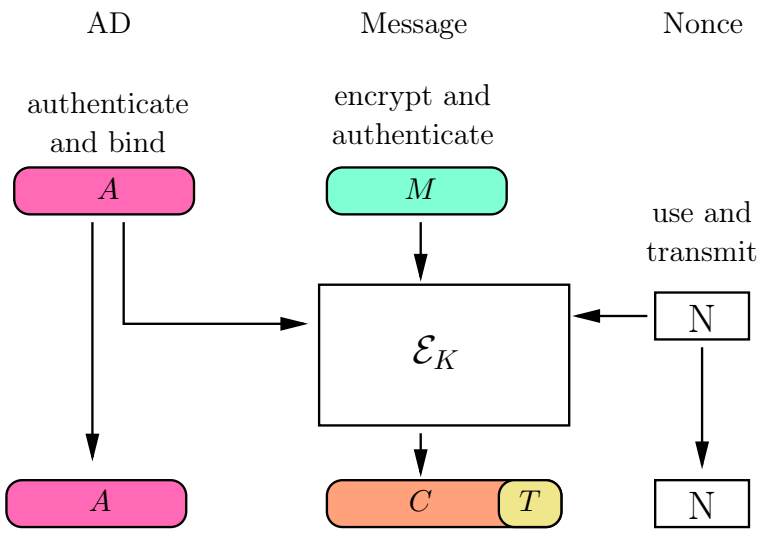
Extra features



Some extra features we want:

- Easy to understand and implement.
- Security level equal to the key length (cf. 64-bit security for most AES-based modes).
- More compact and verifiable security proofs (cf. GSM proof bug).
- No extra operations like key derivation, field multiplications etc. (makes the design more complex).

Extra features



Some extra features we want:

- Easy to understand and implement.
- Security level equal to the key length (cf. 64-bit security for most AES-based modes).
- More compact and verifiable security proofs (cf. GSM proof bug).
- No extra operations like key derivation, field multiplications etc. (makes the design more complex).

Solution: design a permutation-based mode, not a blockcipher one.

Permutation-based

Permutation-based

Why permutation-based?

- A wide permutation can take key, nonce, counter, intermediate values, or a message block altogether as input.
- Plenty of designs: different widths and optimizations;
- The underlying permutation is easier to design and analyze (no need to care of key schedule, mask generation, nonce formatting, etc.);
- Opportunity of sharing the implementation with SHA-3 (important in space-constrained environment).

The Keccak family offers permutations up to 1600 bits wide.

Cons:

- Weaker security model (random permutation);
- Lower throughput (larger calls/byte ratio).

80- and 128-bit security

Most popular modes suggest using AES (128-bit block) as the underlying blockcipher.

Beyond 64-bit security

Most popular modes suggest using AES (128-bit block) as the underlying blockcipher.

No security guaranteed as the number of invocations q approaches $2^{n/2} = 2^{64}$.

Beyond 64-bit security

Most popular modes suggest using AES (128-bit block) as the underlying blockcipher.

No security guaranteed as the number of invocations q approaches $2^{n/2} = 2^{64}$.

We want to offer a higher security margin.

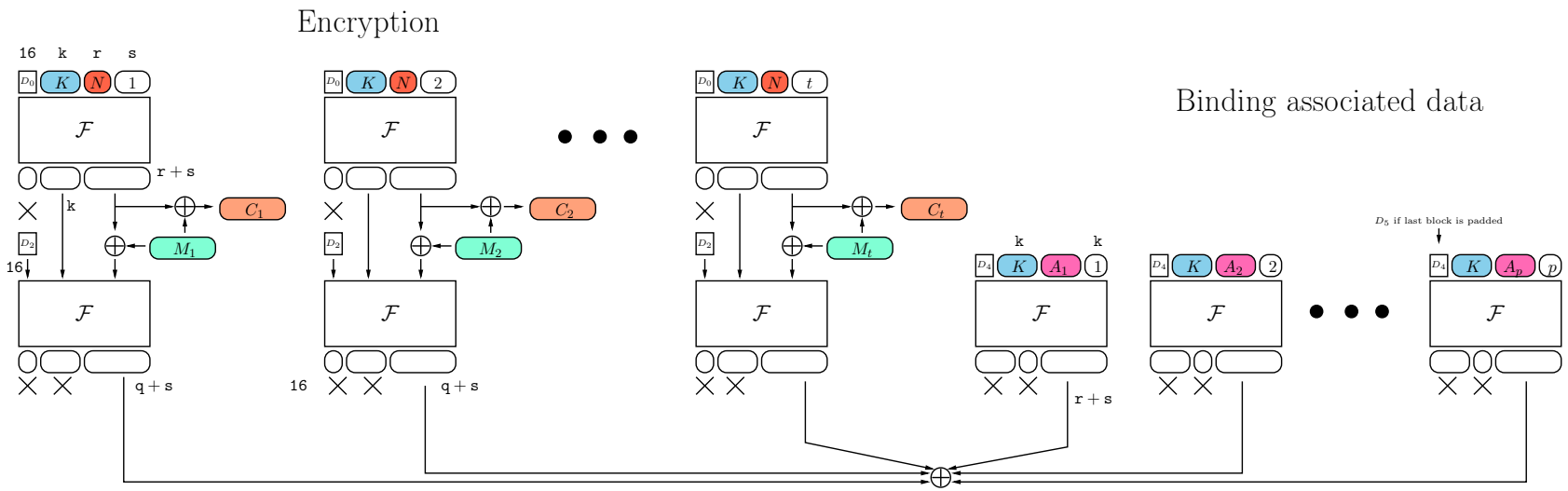
Our new mode PPAE has

Basic features:

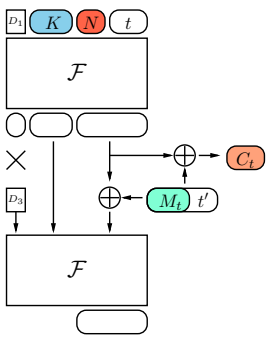
- Fully parallelizable;
- Handles associated data;
- Variable key/nonce/tag length;
- Patent-free;
- Online encryption and authentication, no length awareness;
- Byte-oriented.
- Incremental tag (for max tag length).

Extra features:

- Nonce-based, can be turned to random-IV-based with no penalty;
- Permutation-based (width $w > 3k$);
- Security level up to 128 bits and higher (up to $w/3$) and equal to the key length;
- Compact security proof (in work) in the random permutation setting;
- Permutation inputs and outputs are linked by only XORs and counters, no extra operations;
- Only forward permutation calls.

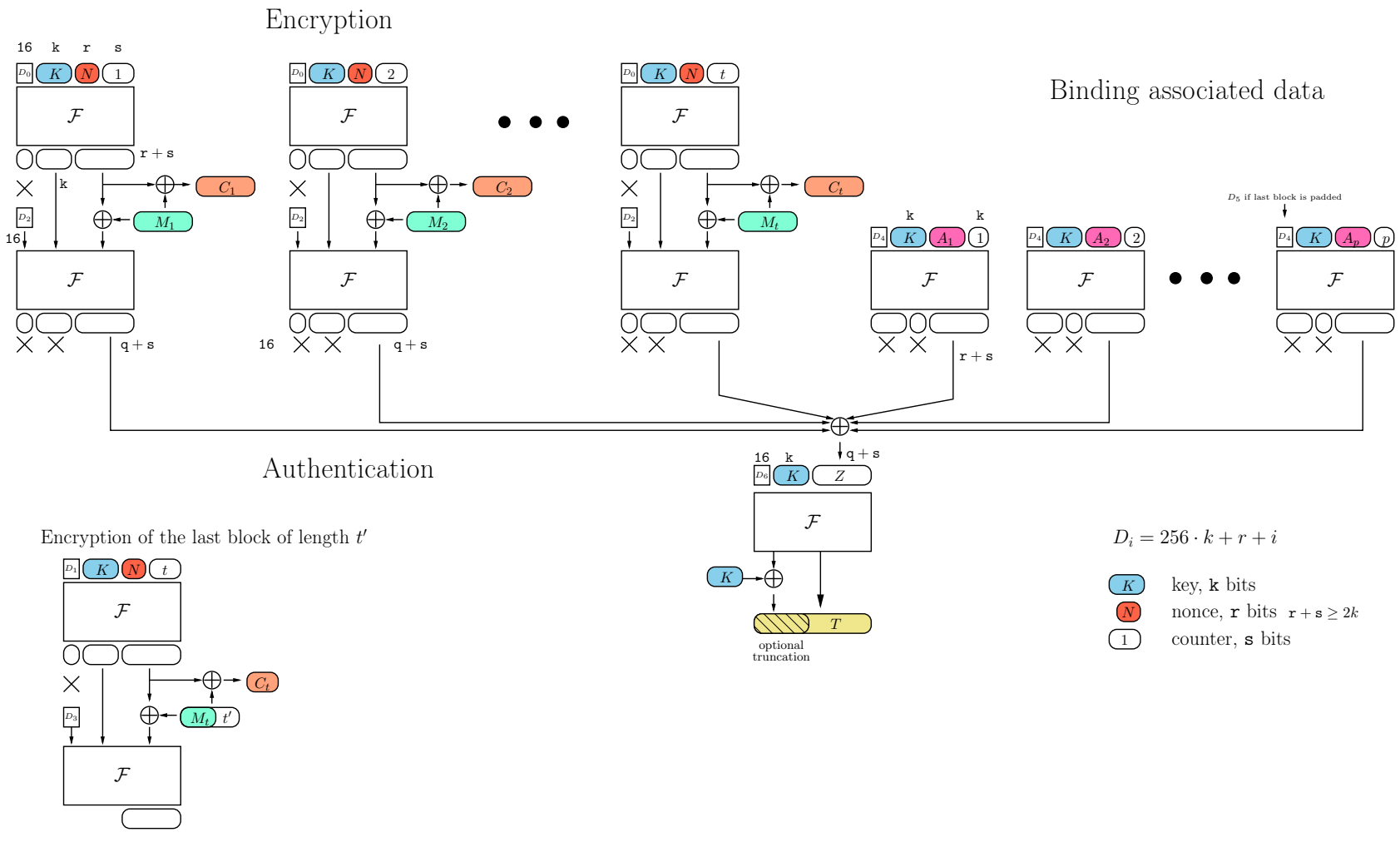


Encryption of the last block of length t'



$D_i = 256 \cdot k + r + i$

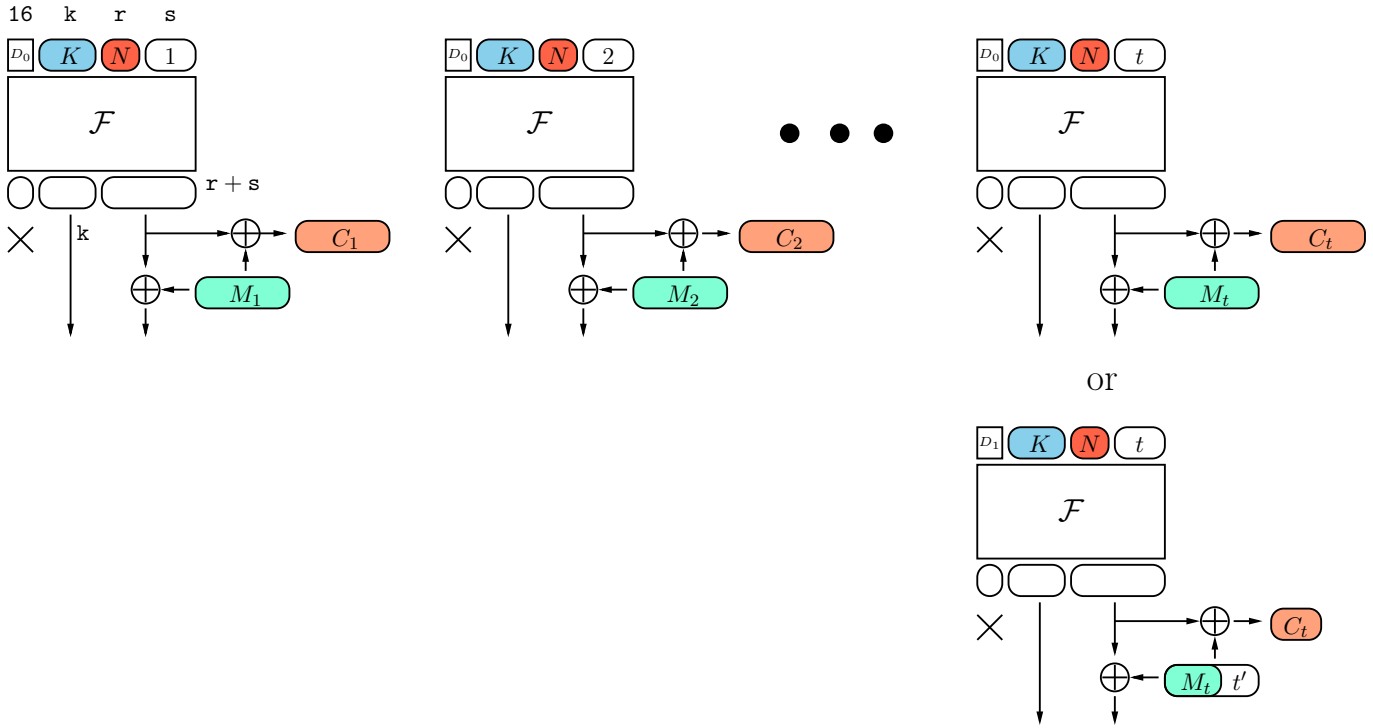
- K key, k bits
- N nonce, r bits $\mathtt{r} + \mathtt{s} \geq 2k$
- 1 counter, s bits



All layers have distinct input prefixes; the last blocks too.

PPAE: encryption

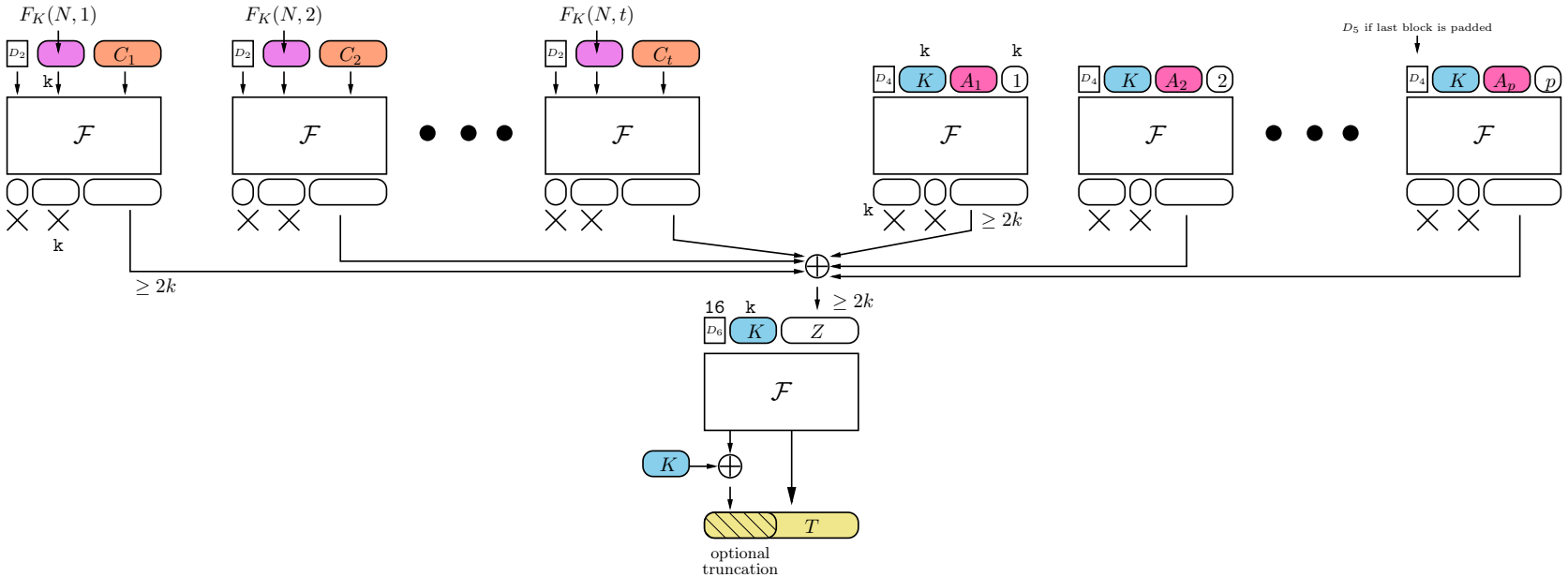
Encryption:



- Counter mode with PRF;
- Confidentiality basically follows from the properties of CTR.

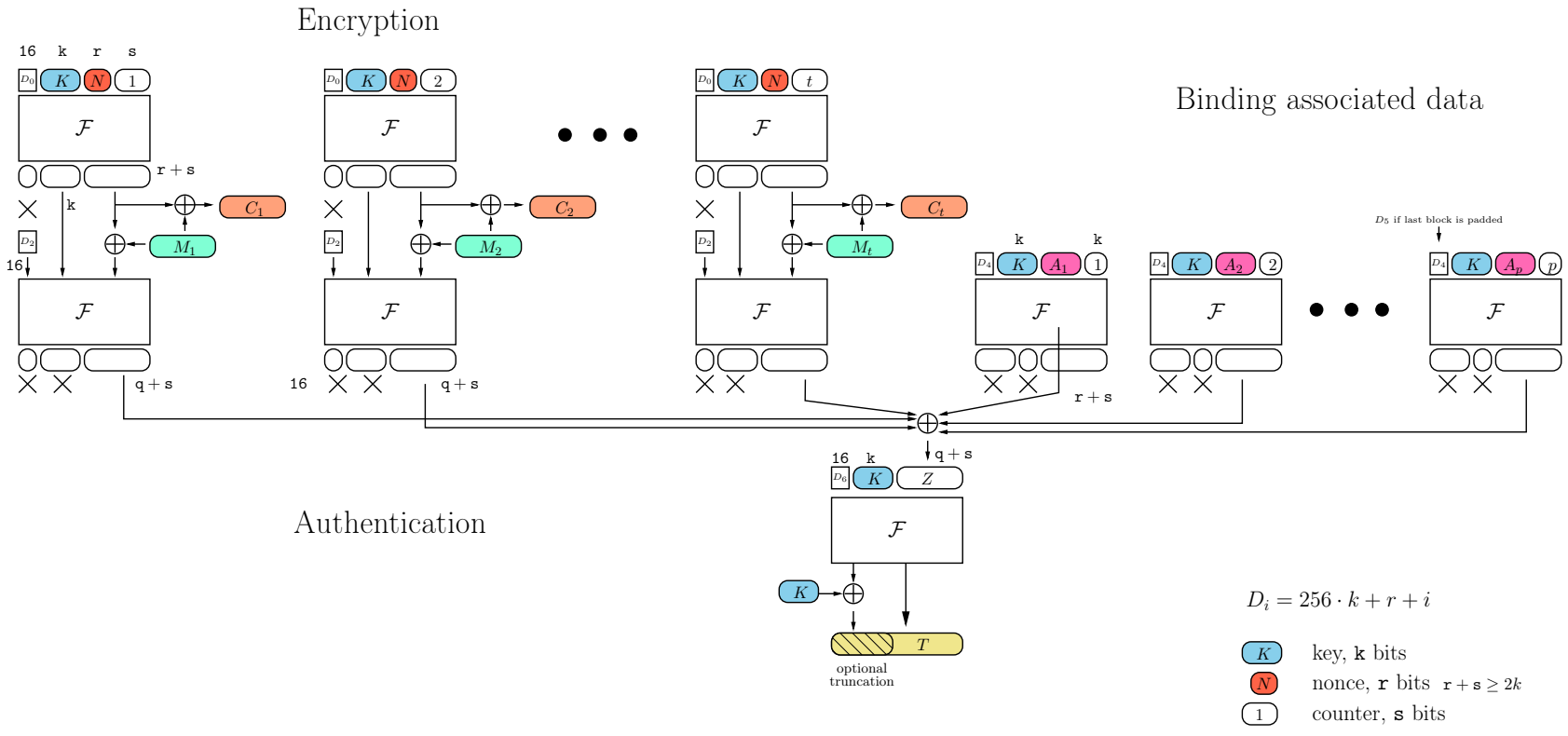
PPAE: authentication

Authentication:



- PMAC style with additional input from the encryption part;
- If the tag has full length, it can be updated with a few extra calls.

Internal permutation



Natural permutation candidate — Keccak-f[1600] with 12 rounds (still large security margin).

Security claims

We estimate the adversary's advantage as follows:

$$\mathrm{Adv}^{\mathsf{Auth},\mathsf{Priv}}_{PPAE[\mathcal{F}],\mathcal{F}}(q) \approx \max\left(\frac{q}{2^k}, \frac{q}{2^\tau}\right),$$

where k — key length, τ — tag length.

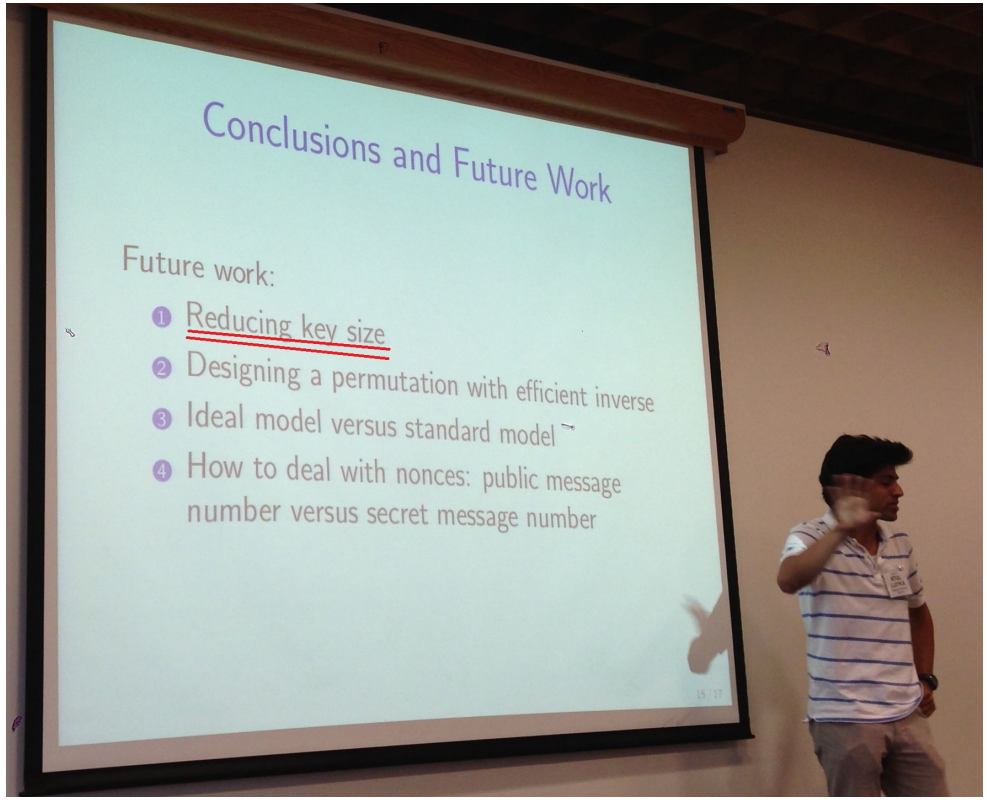
Performance estimates

A permutation of width w is called two times to encrypt and authenticate $(w - k - 16)$ bits of data + tag generation.

Speed (cycles per byte) on amd64

Design	Permutation	
	Keccak-1600/12	Keccak-1600
CTR	3.6	7.2
PPAE (est.)	7.9	15.9
SHA-3-256	5.3	10.6

One of problems posed yesterday is solved



It remains to find an even faster permutation.

Questions?