# Lightweight Authenticated Encryption

Florian Mendel
CSS Security Innovation

# Authenticated encryption

- If your data is worth encrypting, you almost certainly don't want it to be modified!

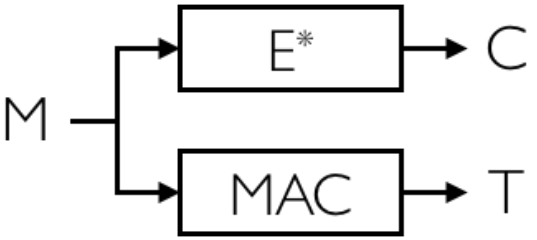
- **Confidentiality**
  - as provided by block cipher modes
- **Authenticity, integrity**
  - as provided by message authentication codes

- *"It is very easy to accidentally combine secure encryption schemes with secure MACs and still get insecure authenticated encryption schemes."*

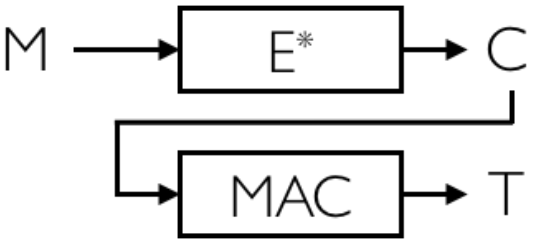
  *– Kohno, Whiting, and Viega*

# Generic compositions

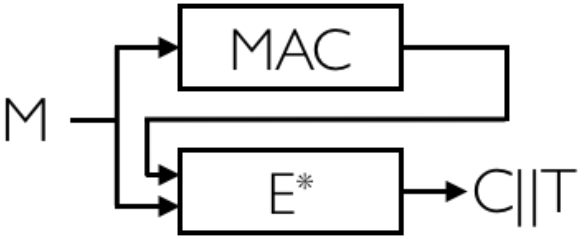
- **Encrypt-and-MAC (E&M)**
  - e.g. in SSL/TLS
  - security depends on E and MAC




- **Encrypt-then-MAC (EtM)**
  - e.g. in SSH
  - security depends on E and MAC




- **MAC-then-Encrypt (MtE)**
  - IPSec, ISO/IEC 19772:2009
  - provably secure



# Standardized schemes

- ISO/IEC specifies six AE modes for block ciphers
  - **EtM**: Encrypt-then-MAC
  - **CCM**: Counter with CBC-MAC
  - **EAX**: encrypt-then-authenticate-then-translate
  - **GCM**: Galois/Counter Mode
  - **OCB**: Offset Codebook Mode
  - **SIV**: Synthetic Initialization Vector mode

# NIST LWC competition

- **NIST Lightweight Cryptography Standardization** (2019-2023)
  - https://csrc.nist.gov/projects/lightweight-cryptography
  - Inspired by the NIST AES and SHA-3 competition
  - **Goal:** Standardize lightweight authenticated encryption schemes that are suitable for use in constrained environments

- **Timeline:**
  - 02/2019: 1st round submissions (56)
  - 08/2019: 2nd round candidates (32)
  - 03/2021: 3rd round candidates (10)
  - 02/2023: selection of winner

National Institute of Standards and Technology

# Ascon – Authenticated Encryption and Hashing

Joint Work with:
Christoph Dobraunig, Maria Eichlseder, Martin Schläffer

# Ascon

- Ascon was designed in 2014
- First choice for lightweight AEAD in CAESAR portfolio
- Extensive published cryptanalysis confirming its security margin
- Additional modes for Hash, XOF, …

# Ascon Team

- Christoph Dobraunig
- Maria Eichlseder
- Florian Mendel
- Martin Schläffer


© Lunghammer, TU Graz

# Ascon Family

- **Authenticated encryption** (CAESAR, 2014)
  - Ascon-128
  - Ascon-128a

- **Hashing** (NIST, 2019)
  - Ascon-Hash
  - Ascon-Xof (eXtendable output function)

# Main Design Goals

- Security
- Efficiency
- Simplicity
- Scalability
- Online
- Single pass
- Lightweight
- Side-Channel Robustness

# Authenticated Encryption

- Nonce-based AE scheme
- Sponge inspired (permutation-based)

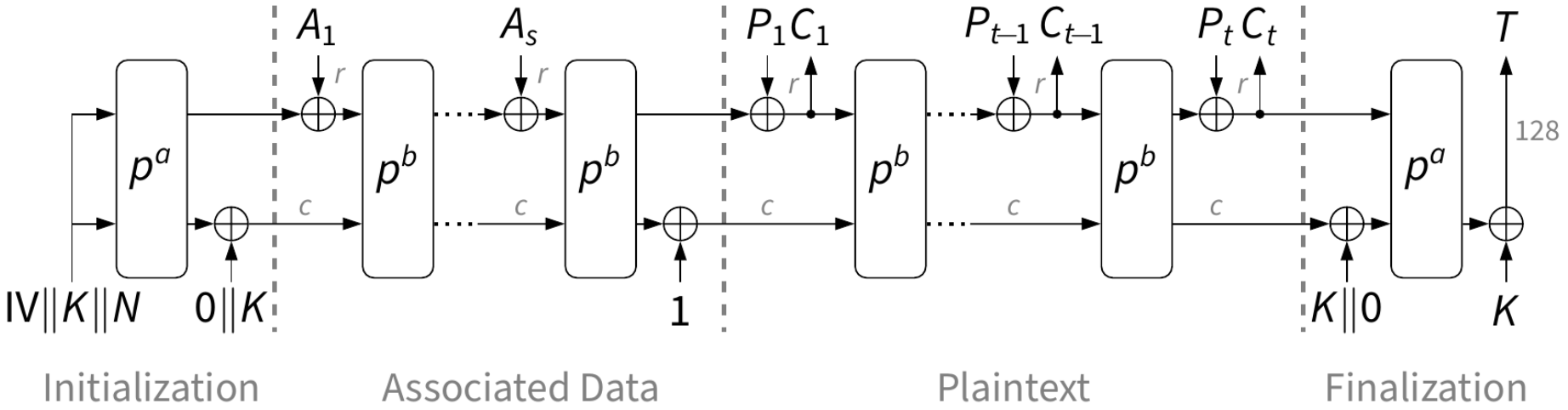
| | Ascon-128 | Ascon-128a |
|---|---|---|
| **Security** | 128 bits | 128 bits |
| **Rate (r)** | 64 bits | 128 bits |
| **Capacity (c)** | 256 bits | 192 bits |
| **State size (s)** | 320 bits | 320 bits |

https://ascon.iaik.tugraz.at/

# Authenticated Encryption: Working Principle

- The encryption process is split into four phases:
  - Initialization
  - Associated Data Processing
  - Plaintext Processing
  - Finalization

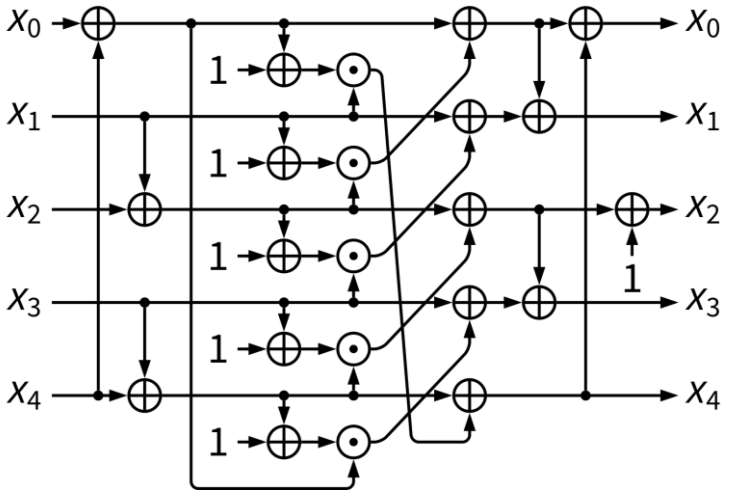
- **Ascon-128:** r=64, a=12, b=6
- **Ascon-128a:** r=128, a=12, b=8



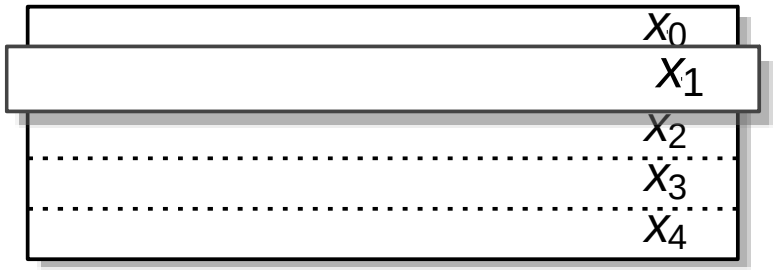

https://ascon.iaik.tugraz.at/

# Permutation with {6, 8, 12} Rounds

## S-box layer


## Linear layer




$$x_0 = x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28)$$
$$x_1 = x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39)$$
$$x_2 = x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6)$$
$$x_3 = x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17)$$
$$x_4 = x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41)$$

https://ascon.iaik.tugraz.at/

# Further Constructions: Hashing, XOF, …

# Ascon: Hashing / XOF

- Hash Function / XOF (NIST)
- Sponge construction

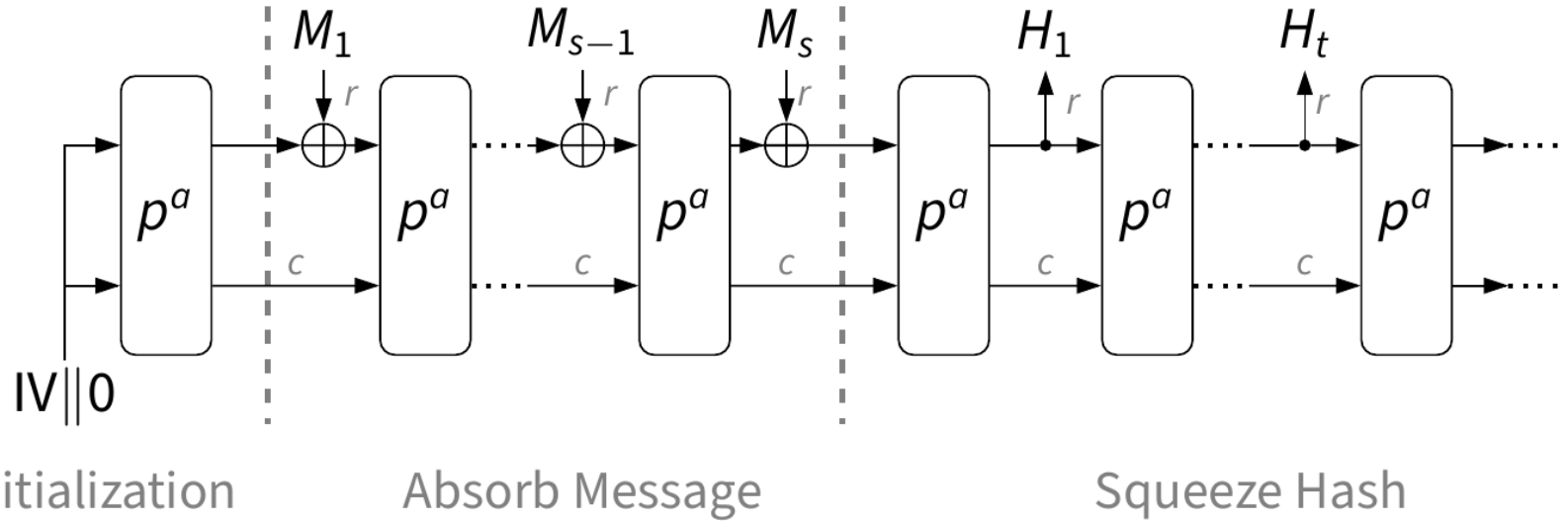
| | Ascon-Hash | Ascon-Xof |
|---|---|---|
| **Hash Size** | 256 bits | variable |
| **Rate (r)** | 64 bits | 64 bits |
| **Capacity (c)** | 256 bits | 256 bits |
| **State size (s)** | 320 bits | 320 bits |

https://ascon.iaik.tugraz.at/

# Hash / XOF: Working Principle

- Initialization
- Message Processing (absorb)
- Tag generation (squeeze)

- **Ascon-Hash:** r=64, a=12
- **Ascon-XOF:** r=64, a=12

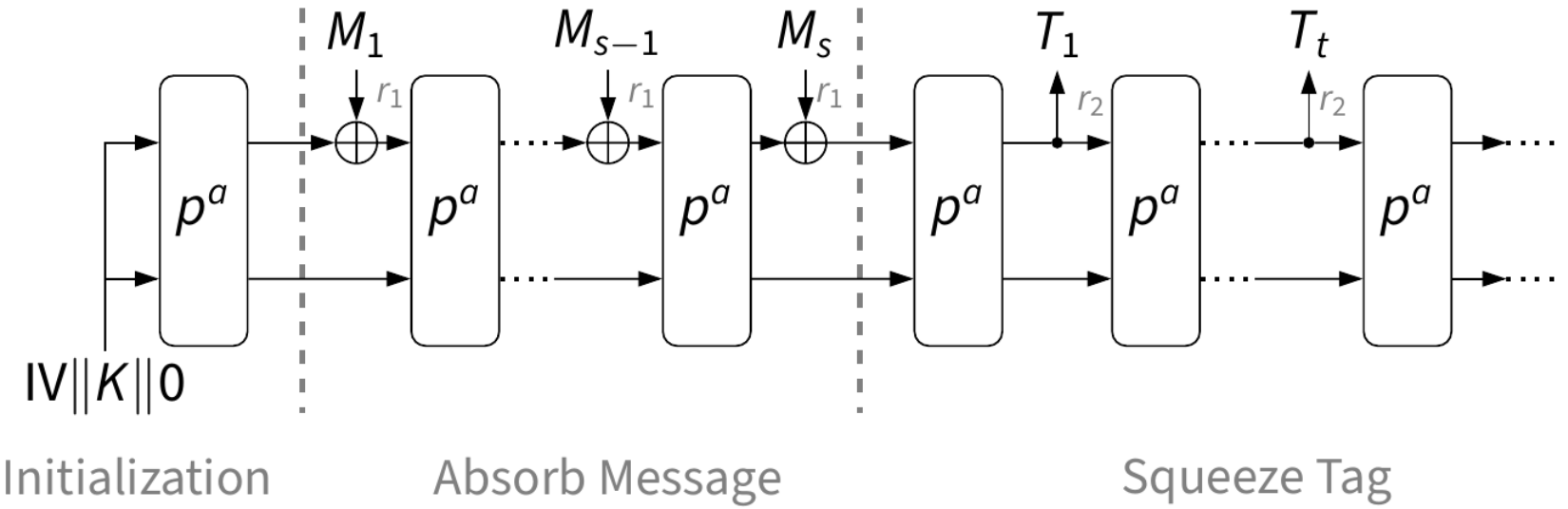



https://ascon.iaik.tugraz.at/

# MAC / PRF (extension)

- Initialization
- Message Processing (absorb)
- Tag generation (squeeze)

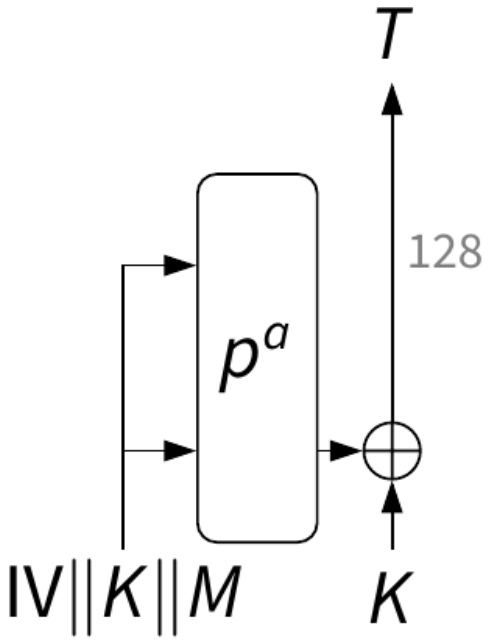
- **Ascon-PRF:**
  $r_1$=256, $r_2$=128, a=12
- **Ascon-MAC:**
  $r_1$=256, $r_2$=128, a=12




https://ascon.iaik.tugraz.at/

# PRF-Short (extension)

- Initialization of Ascon-128 with different IV
- Nonce replaced by message M (≤ 128 bits)
- Generates tag T (≤ 128 bits)




https://ascon.iaik.tugraz.at/

# Ascon Features

- **Balanced design**
  - Small and efficient in hardware
  - Efficient on 64 and 32 bit CPUs (bit-interleaving)
- **Low overhead for short messages**
  - High key agility (no-key schedule)
  - Low overhead for initialization/finalization
- **Natural side-channel protection**
  - No table-lookups (constant time)
  - Low-degree S-boxes (efficient masking)
- **Robustness** against nonce-reuse, etc.

# Implementation / Performance

Public Benchmarking Results for Ascon-128 and Ascon-128a

# FPGA benchmarks for Ascon-128 and Ascon-128a

› FPGA benchmarks for AD+PT throughput in [Mbit/s] and area in [LUTs]

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ASCON-128a** | 6297.6 | 2410 | 2.61 |
| **ASCON-128** | 3744.0 | 2126 | 1.76 |
| **AES-GCM** | 2700.8 | 3270 | 0.83 |

Xilinx Artix-7

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ASCON-128a** | 3031.0 | 4552 | 0.67 |
| **ASCON-128** | 2157.0 | 3215 | 0.67 |
| **AES-GCM** | 1548.3 | 8754 | 0.18 |

Intel Cyclone 10 LP

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ASCON-128a** | 2158.1 | 5909 | 0.37 |
| **ASCON-128** | 1427.5 | 3764 | 0.38 |
| **AES-GCM** | 1384.4 | 6740 | 0.21 |

Lattice ECP5

https://eprint.iacr.org/2020/1207

# ASIC benchmarks for Ascon-128 and Ascon-128a

› ASIC benchmarks for AD+PT and throughput in [bits/cycle] and scaled area

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **Ascon-128a** | 25.60 | 1.49 | 17.18 |
| **Ascon-128** | 16.00 | 1.56 | 10.25 |
| **AES-GCM** | 11.63 | 2.75 | 4.22 |

https://eprint.iacr.org/2021/049

# Embedded implementations (evaluated by LAS3)

› Time to process NIST testvectors in [µs] on embedded devices

| | Uno | F1 | ESP | F7 | R5 |
|---|---|---|---|---|---|
| **Ascon-128a** | 1981 | 66.4 | 18.4 | 11.8 | 7.3 |
| **Ascon-128** | 2337 | 76.7 | 22.3 | 13.8 | 8.5 |
| **AES-GCM** | - | 332.8 | 67.2 | 35.8 | 23.7 |

› Code size in [bytes] on embedded devices

| | Uno | F1 | ESP | F7 | R5 |
|---|---|---|---|---|---|
| **Ascon-128a** | 2544 | 2252 | 1200 | 1240 | 1792 |
| **Ascon-128** | 2552 | 2157 | 1120 | 1180 | 1792 |
| **AES-GCM** | - | 9908 | 14832 | 9836 | 14272 |

https://lwc.las3.de/table.php

# High-end benchmarking (w/o Ascon HW extensions)

› Performance in [cycles/byte]

| | AMD Ryzen 9 | ARM Cortex-A72 |
|---|---|---|
| **Ascon-128a** | 5.6 | 7.0 |
| **Ascon-128** | 8.1 | 10.5 |
| **AES-GCM** | 1.1* | 30.6 |

*with AES-NI

https://bench.cr.yp.to/

# Ascon hardware extensions/instructions

- A Fast and Compact RISC-V Accelerator (for RV32, also ARM)
  - RI5CY ASCON-p with **4.7kGE**: speedup factor **50x**
  - Reuse 10 registers of CPU register file
  - https://eprint.iacr.org/2020/1083.pdf

- ARM Custom Datapath Extension, RISC-V Bitmanip Extension, ...
  - 32-bit funnel shift instructions (RV32B: FSRI, ESP32: SRC)
  - 32-bit interleaving instructions (RV32B: ZIP/UNZIP, ARM CDE: CX3)
  - Fused AND/XOR, BIC/XOR instructions (ARM A64: BCAX, ARM CDE: CX3A)
  - SHA-2 like Sigma instructions (ARM CDE: CX3DA)

# Side-Channel Protected Implementations

Ascon: Designed with SCA in mind

# Ascon: Designed with SCA in mind

- Algebraic degree 2 of S-box
- Limited damage if state is recovered
- Levelled implementations
  - Higher protection order for Init/Final (key)
  - Lower protection order for AD/PT/CT processing (data)
- Masking using Toffoli gate

# Masked Hardware Implementations

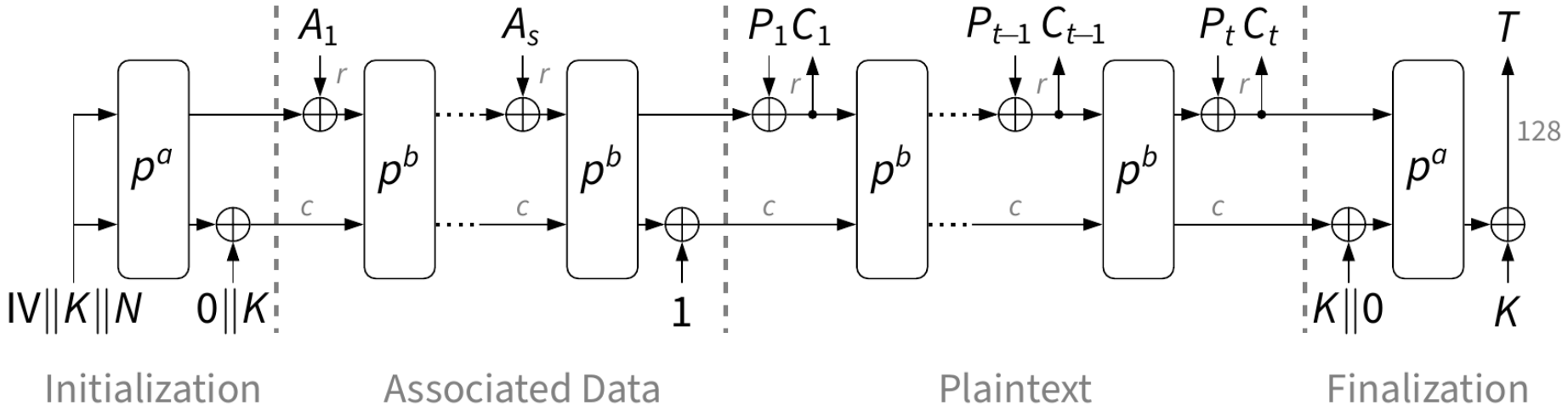
- Masked DOM-Implementation of Ascon-128 (CHES2017)

| Protection Order | Pipelined [kGE] | Pipelined [Mbps] | Parallel [kGE] | Parallel [Mbps] |
|---|---|---|---|---|
| **1** | 10.86 | 108 | 28.89 | 2246 |
| **2** | 16.19 | 108 | 53.00 | 1896 |
| **3** | 21.59 | 110 | 81.21 | 1903 |
| **4** | 27.13 | 71 | 118.27 | 1786 |
| **…** | … | … | … | … |

https://eprint.iacr.org/2017/103

- First and second-order protected hardware implementations are available at: https://github.com/ascon/ascon-hardware-sca

# Levelled Implementations

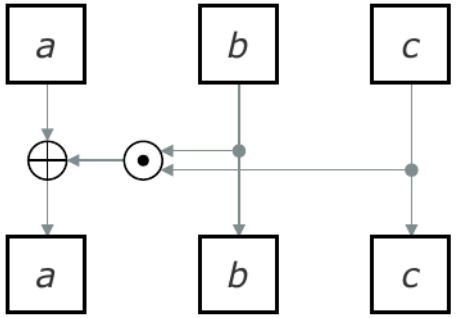



- Higher protection order for Init/Final (key)
- Lower protection order for AD/PT/CT processing (data)

https://eprint.iacr.org/2018/708

# Masking using Toffoli Gate

- **More efficient than masked AND gate**
  - Fewer instructions
  - Fewer registers
  - Fewer randomness




- **No fresh randomness needed during round computation**
  - Randomness is not lost (invertible shared Toffoli gate)
  - Randomness of previous round can be reused

- **Benefits of invertible shared function**
  - Uniform by design
  - SIFA: reduced attack surface

https://eprint.iacr.org/2019/536

# Further SCA Optimizations

- **Preliminary Goal:** Achieve 1st-order protection with 2/3 shares in C/ASM*
  - Rotation offset between shares
  - Minimum number of ASM instructions (Toffoli gate)
  - Some register clears/NOPS needed

- Performance in cycles/byte (green: evaluated)

| impl/shares<br>flags | armv6 | C<br>-O2 | C<br>-Os | 2-1-2<br>-O2 | 2-1-2<br>-Os | 2<br>-O2 | 2<br>-Os | 3<br>-O2 | 3<br>-Os |
|---|---|---|---|---|---|---|---|---|---|
| ARM1176JZF | 58 | 70 | 85 | 88 | 100 | 260 | 343 | 524 | 703 |
| STM32F415 | 59 | 84 | 90 | 90 | 98 | 320 | 378 | 650 | 669 |

- Implementations/results available at: https://github.com/ascon/simpleserial-ascon

* Our implementations should be considered as a starting point to generate device specific C/ASM implementations

# ISAP – Authenticated Encryption

Joint Work with:
Christoph Dobraunig, Maria Eichlseder, Stefan Mangard, Bart Mennink, Thomas Unterluggauer, Robert Primas

# Motivation

- Problem: side-channel attacks
- Countermeasures: hiding, masking, TI, …
- Reduce overhead of countermeasures
  - Ascon, Xoodyak, …

- Can we do more?

# Related Work

- C. Dobraunig, M. Eichlseder, S. Mangard, F. Mendel, and T. Unterluggauer: ISAP - Towards Side-Channel Secure Authenticated Encryption. FSE 2017
- G. Barwell, D. P. Martin, E. Oswald, and M. Stam: Authenticated Encryption in the Face of Protocol and Side Channel Leakage. ASIACRYPT 2017
- F. Berti, O. Pereira, T. Peters, and F.-X. Standaert: On Leakage-Resilient Authenticated Encryption with Decryption Leakages. FSE 2018
- …

# ISAP

- Robustness against DPA on algorithmic level for
  - Encryption
  - Decryption

- Solely based on the sponge construction
  - Limits the attack surface for SCA

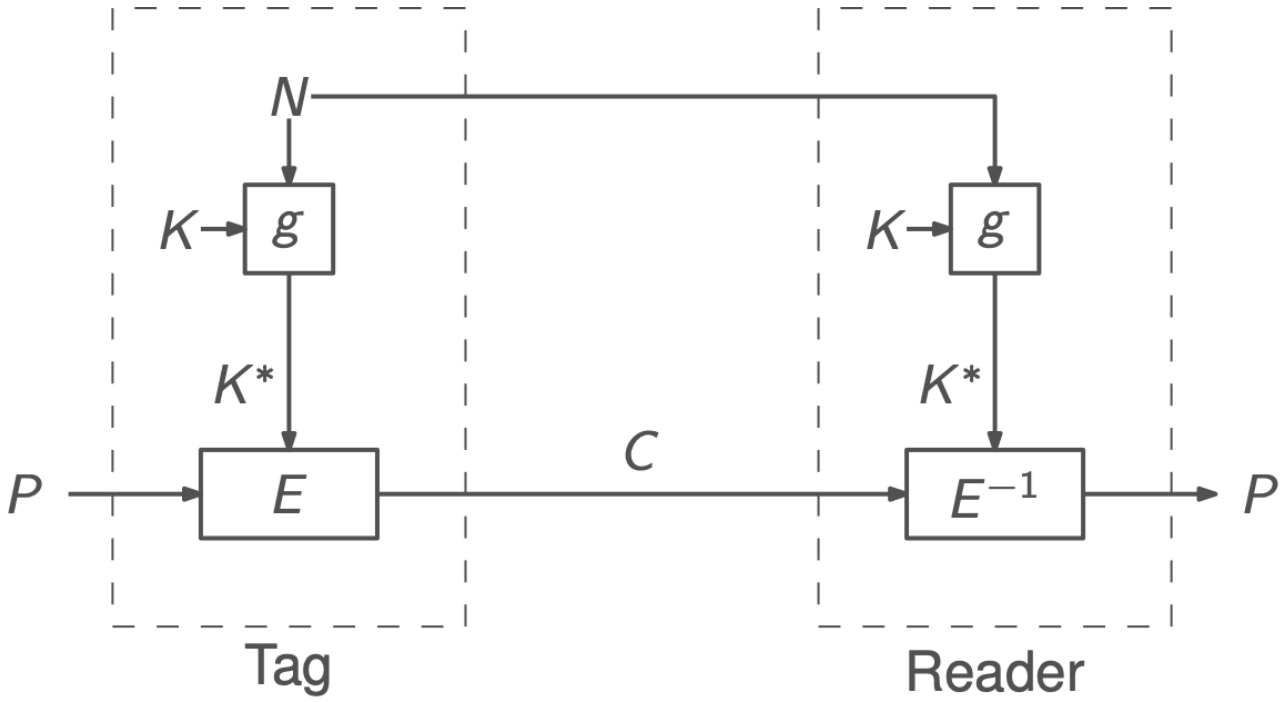
# SPA and DPA

- **Simple Power Analysis (SPA)**
  - Observe device processing the same or a few inputs
  - Techniques directly interpreting measurements

- **Differential Power Analysis (DPA)**
  - Observe device processing many different inputs
  - Allows for the use of statistical techniques

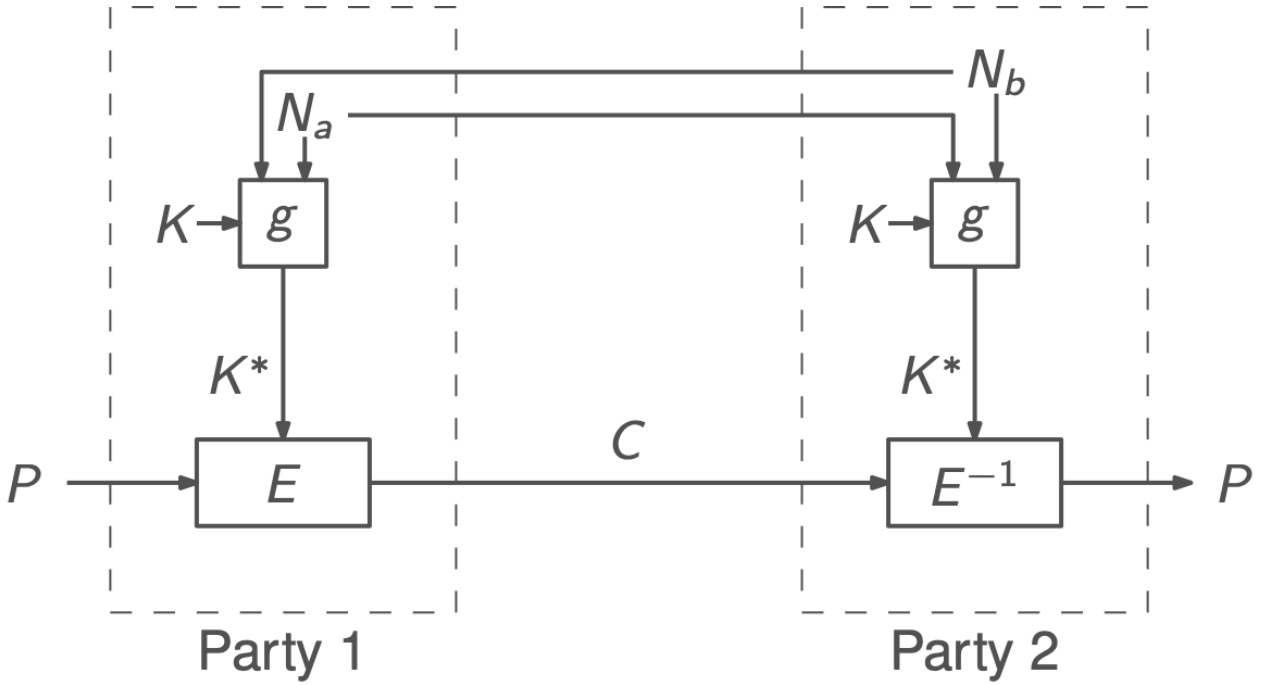
# Fresh Re-keying

› Basic Idea of fresh re-keying (tag and reader)



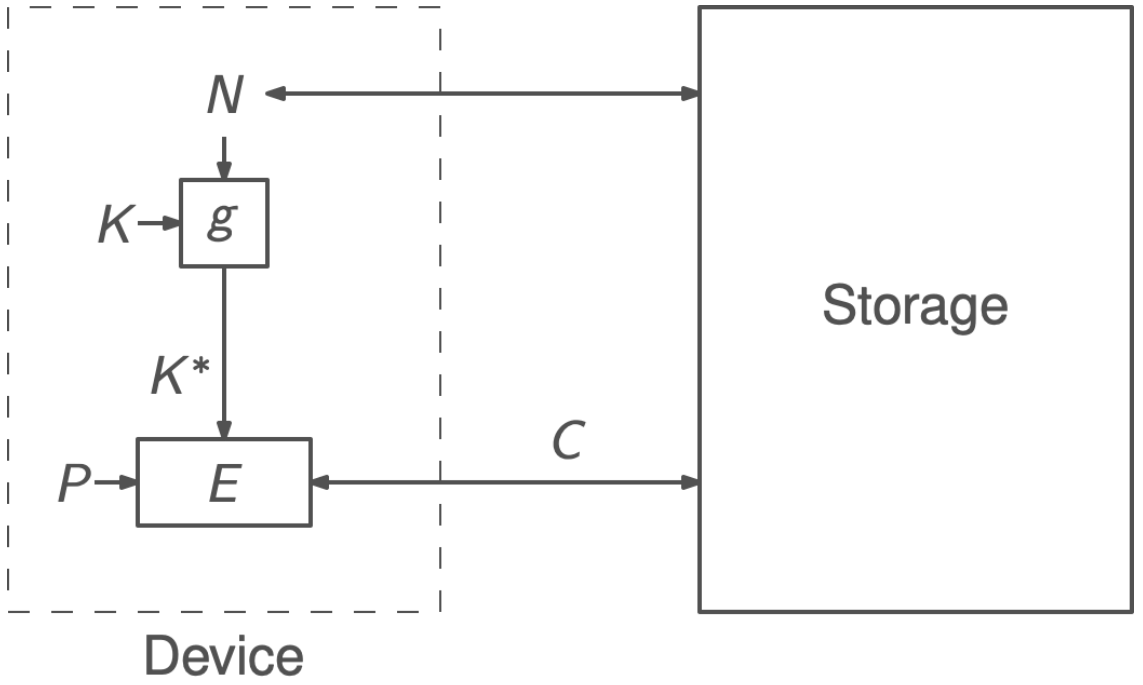
# Fresh Re-keying

› Basic Idea of fresh re-keying (2 parties)



# What about storage?

- Encryption still fine
- Decryption might be critical



# How to Protect Decryption ?

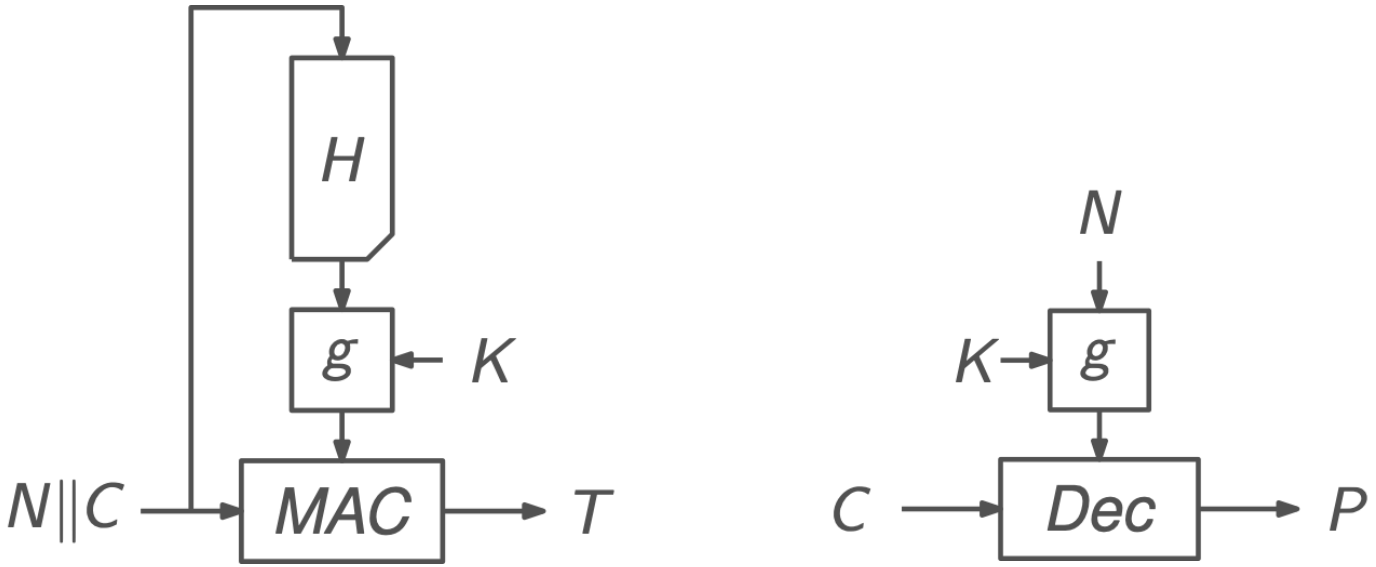
- Rely on implementation countermeasures
  - Costly
  - Makes re-keying for encryption kind of obsolete
- Limit to one decryption
  - Keep track of the nonce
  - Re-encrypt data
  - Time consuming
  - Damaging

# Multiple Decryption

- **Goal:** Retain principles of fresh re-keying and allowing multiple decryption
  - DPA robustness in storage settings
  - DPA robustness in unidirectional/broadcast settings

# Principle of Decryption

› **Idea:** “Bind” the session key to the data that is decrypted

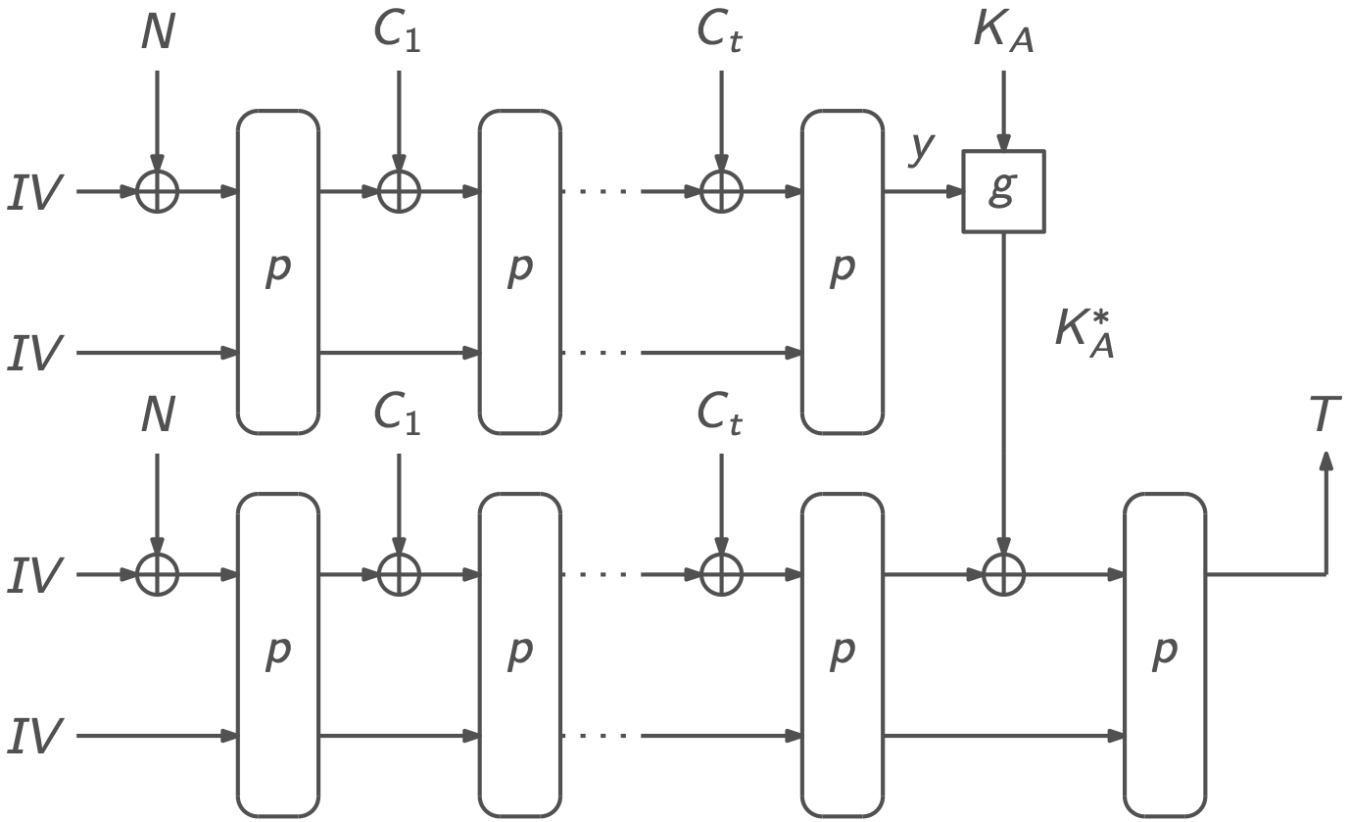


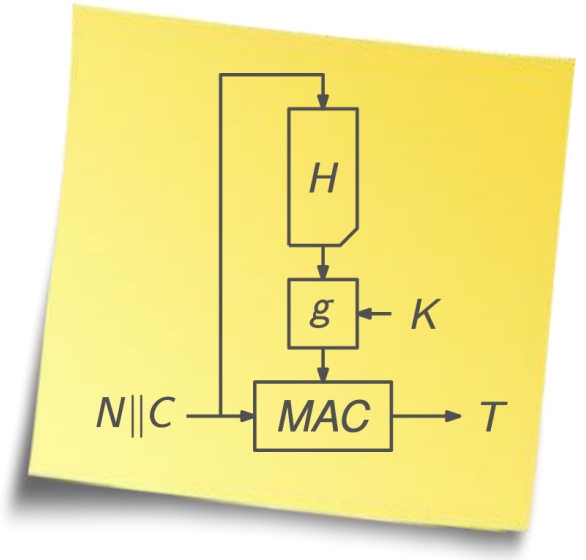
# Benefits of Sponges

- Well-studied and analyzed
- Allows to implement a wide range of primitives
- No inverse building blocks (permutation) needed
- No key schedule, key is injected once
- Simple way to model side-channel-leakage

# Authentication / Verification

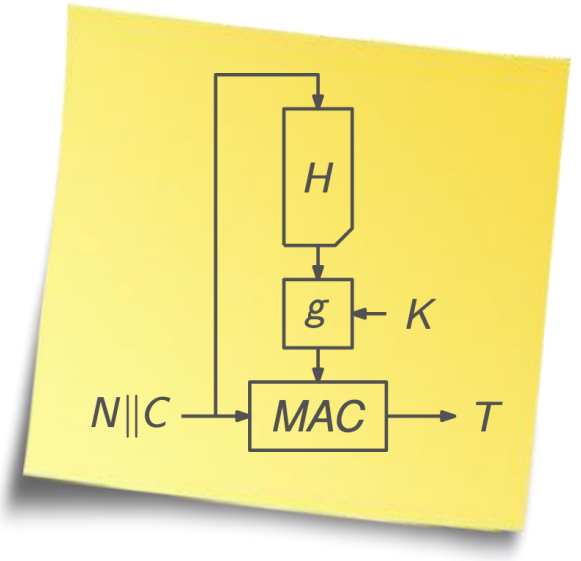
› Combine hash function with a MAC

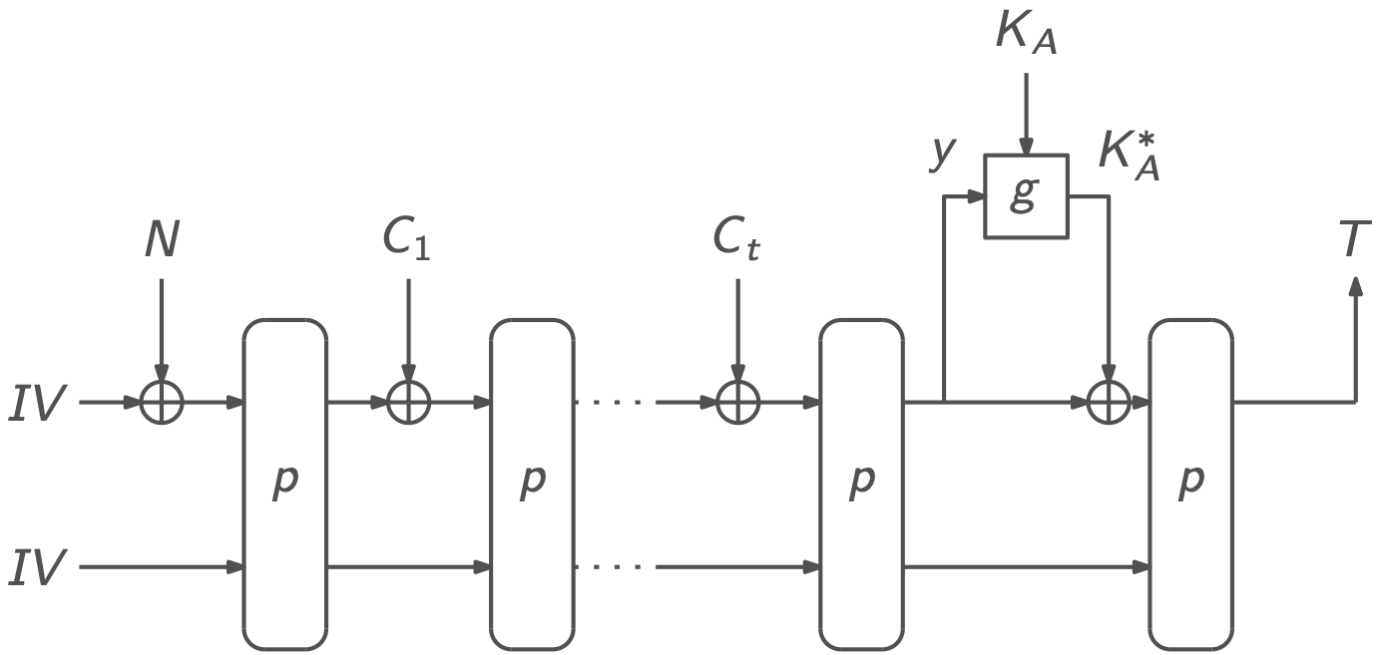


# Authentication / Verification

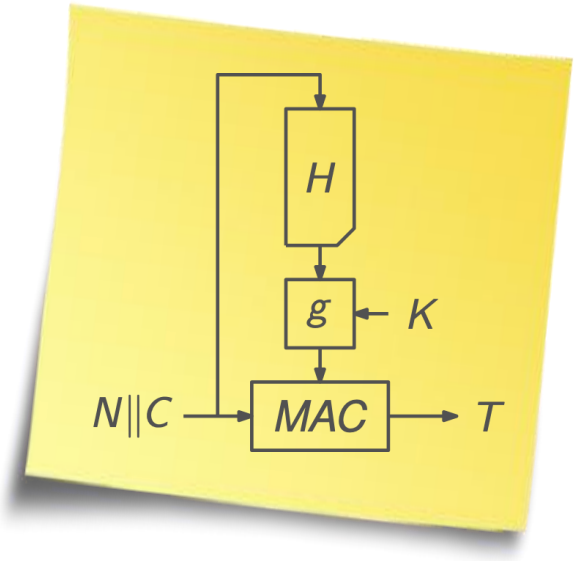
› Combine hash function with a MAC

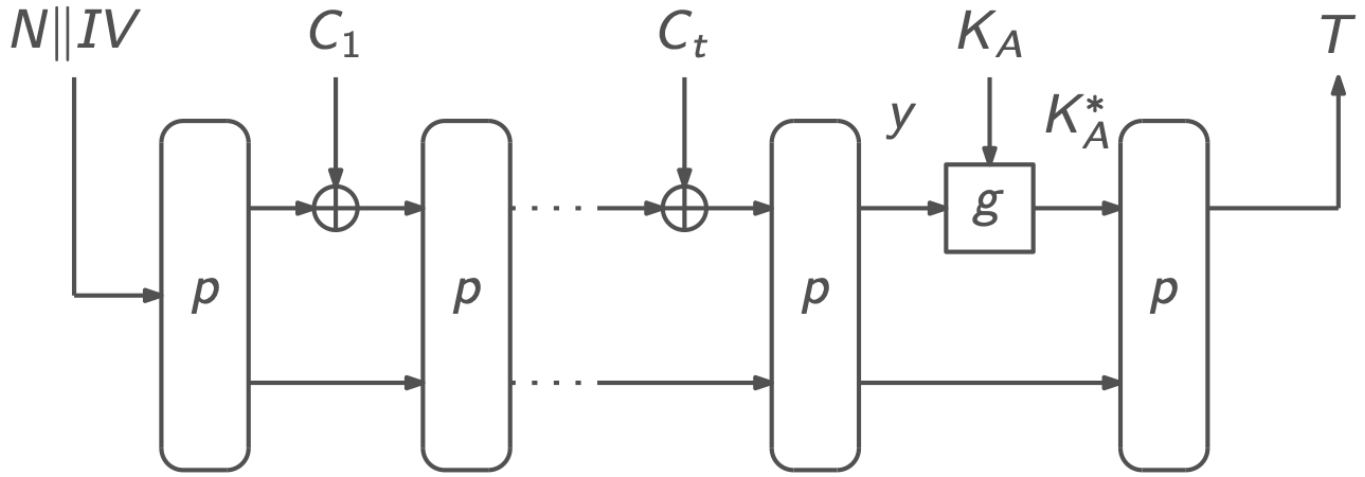


# Authentication / Verification

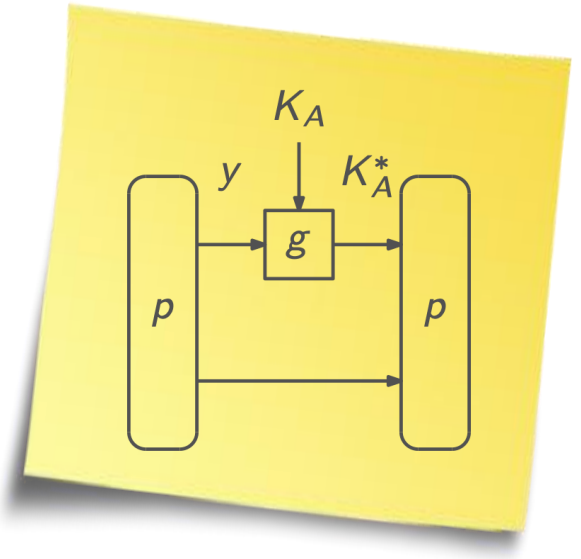
› Use suffix MAC instead of hash-then-MAC



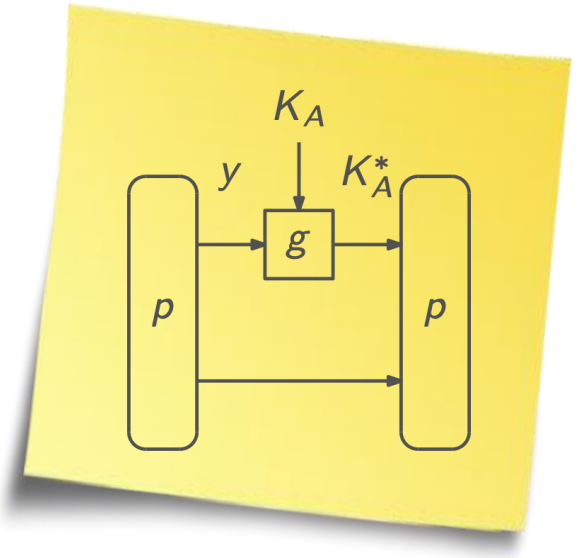
# Absorbing the key

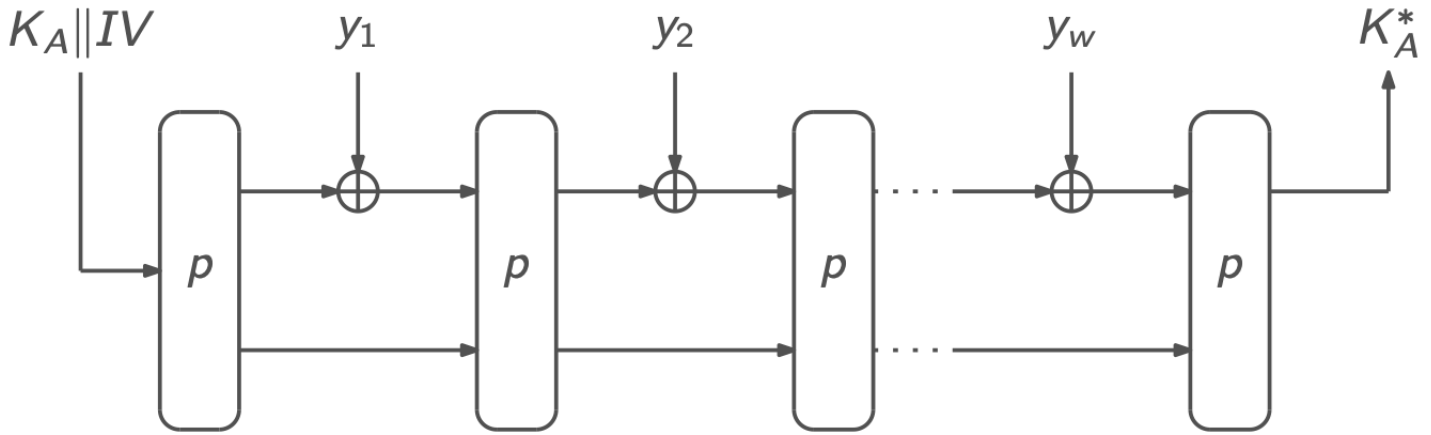
- Modular multiplication
- LPL and LWE
- Sponges



# Absorbing the key

- **Idea:** Reduce rate to a minimum
- Related to the classical GGM construction

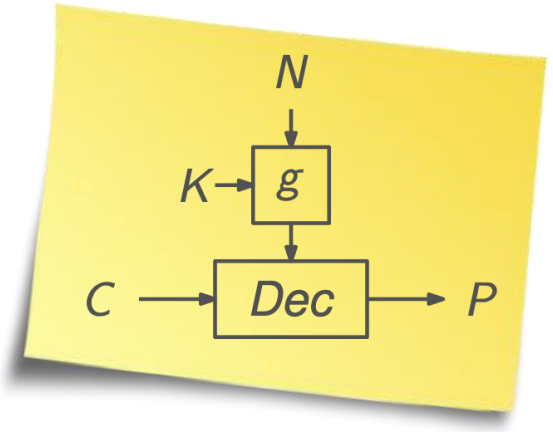


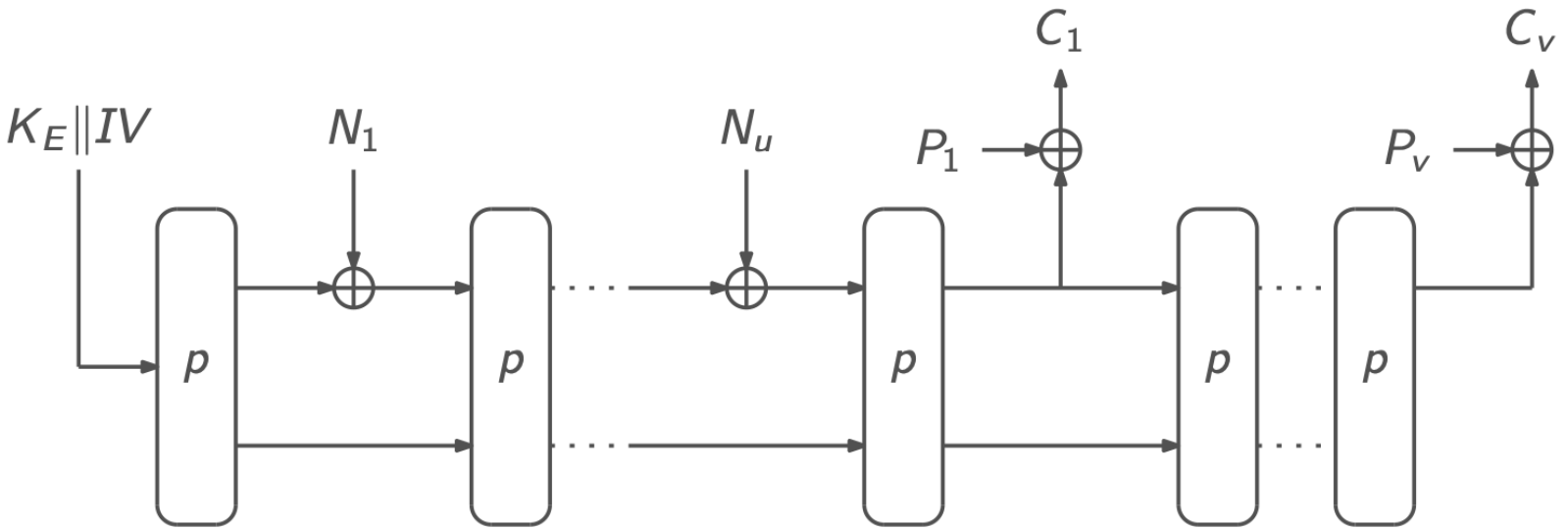




https://ieeexplore.ieee.org/document/6855576

# Encryption / Decryption

› Combine re-keying function and stream cipher



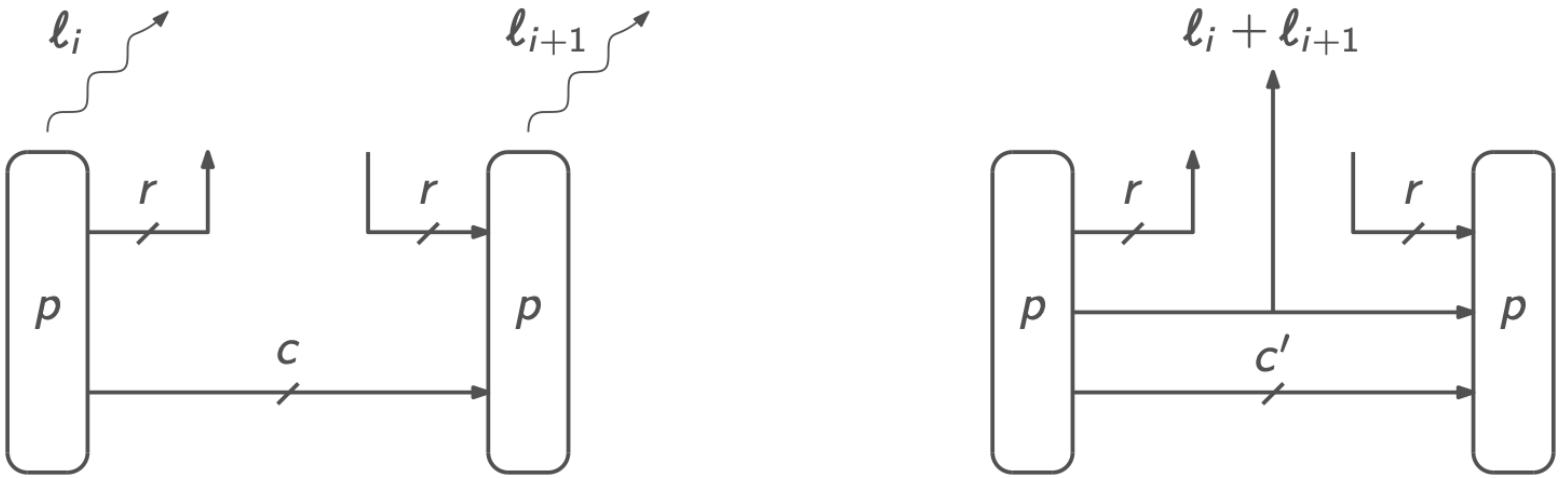

https://ieeexplore.ieee.org/document/6855576

# Benefits of Sponges

- Well-studied and analyzed
- Allows to implement a wide range of primitives
- No inverse building blocks (permutation) needed
- No key schedule, key is injected once
- Simple way to model side-channel-leakage

# Side-Channel Leakage

› Modelling side-channel leakage in sponges




https://eprint.iacr.org/2020/200

# Leakage Resilience

- C. Dobraunig and B. Mennink: Leakage Resilience of the Duplex Construction. ASIACRYPT 2019
- J.-P. Degabriele, C. Janson and P. Struck: Sponges Resist Leakage - The Case of Authenticated Encryption. ASIACRYPT 2019
- C. Guo, O. Pereira, T. Peters and F.-X. Standaert: Towards Low-Energy Leakage-Resistant Authenticated Encryption from the Duplex Sponge Construction. FSE 2020
- C. Dobraunig and B. Mennink: Security of the Suffix Keyed Sponge. FSE 2020
- C. Dobraunig and B. Mennink: Leakage Resilience of the ISAP Mode - A Vulgarized Summary. NIST Lightweight Cryptography Workshop 2019
- …

# Instances

- **Ascon**
  - ISAP-A-128A
  - ISAP-A-128
- **Keccak-p[400]**
  - ISAP-K-128A
  - ISAP-K-128

https://isap.iaik.tugraz.at/

# FPGA benchmarks for ISAP-A-128A and ISPA-K-128A

› FPGA benchmarks for AD+PT throughput in [Mbit/s] and area in [LUTs]

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ISAP-A-128A** | 609.5 | 2157 | 0.28 |
| **ISAP-K-128A** | 829.6 | 3491 | 0.23 |
| **AES-GCM** | 2700.8 | 3270 | 0.83 |

Xilinx Artix-7

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ISAP-A-128A** | 551.1 | 3026 | 0.18 |
| **ISAP-K-128A** | 567.0 | 3767 | 0.15 |
| **AES-GCM** | 1548.3 | 8754 | 0.18 |

Intel Cyclone 10 LP

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ISAP-A-128A** | 238.9 | 3623 | 0.07 |
| **ISAP-K-128A** | 282.2 | 5703 | 0.05 |
| **AES-GCM** | 1384.4 | 6740 | 0.21 |

Lattice ECP5

https://eprint.iacr.org/2020/1207

# ASIC benchmarks for ISAP-A-128A and ISPA-K-128A

› ASIC benchmarks for AD+PT and throughput in [bits/cycle] and scaled area

| | Throughput | Area | Throughput / Area |
|---|---|---|---|
| **ISAP-A-128A** | 2.46 | 1.08 | 2.27 |
| **ISAP-K-128A** | 3.42 | 1.19 | 2.87 |
| **AES-GCM** | 11.63 | 2.75 | 4.22 |
| **AES-GCM** | 3.88 | 2.37 | 1.64 |

https://eprint.iacr.org/2021/049

# Features of ISAP

- AEAD scheme following the NIST call
  - Provides increased robustness against DPA on algorithmic level

- Enables several use-cases
  - Multiple decryption of stored data
  - Unidirectional/Broadcast communication

- ISAP is best suited for applications where performance is not critical, but robustness against side-channel attacks is needed, and code size and area matters

https://isap.iaik.tugraz.at/

# Summary

- **Lightweight Authenticated Encryption**
  - Security of 128 bits
  - Efficient on constraint devices
- **Security**
  - Well analysed/understood
  - Large security margin
- **Efficiency**
  - Small and fast on constraint devices in HW and SW
  - Good performance on modern CPUs
  - Natural side-channel protection


Part of your life. Part of tomorrow.