Spook: Sponge-based Leakage-resistant Authenticated Encryption

Davide Bellizia   Francesco Berti   Olivier Bronchain   Gaëtan Cassiers
Sébastien Duval   Chun Guo   Gregor Leander   Gaëtan Leurent   Itamar Levi
Charles Momin   Olivier Pereira   Thomas Peters   François-Xavier Standaert
Balazs Udvarhelyi   Friedrich Wiemer

December 17, 2020

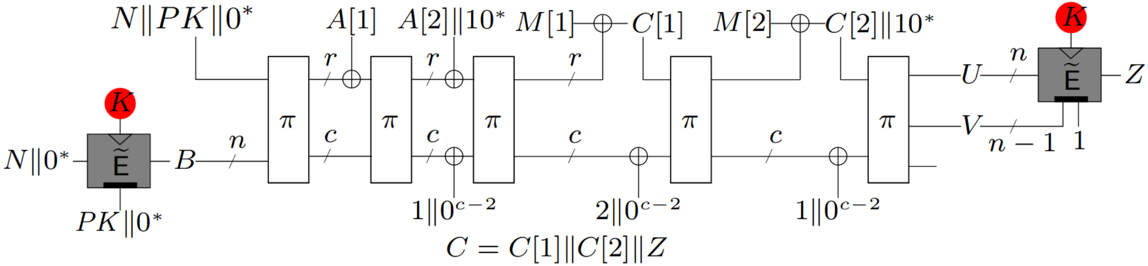
UCLouvain
TECHNISCHE UNIVERSITÄT DARMSTADT
LMV Laboratoire de mathématiques de Versailles - CNRS UMR 8100
UVSQ université PARIS-SACLAY
cnrs
SHANDONG UNIVERSITY 1901
RUHR UNIVERSITÄT BOCHUM RUB
Inria informatics mathematics
Tradition of Excellence Bar-Ilan University אוניברסיטת בר-אילן

Table of Contents

Goals of Spook

- authenticated encryption
- side-channel security
  - strong confidentiality in encryption with leakage
  - strong integrity in encryption/decryption with leakage
- low-energy
- nonce misuse-resilient
- multi-user security

Mode of Operation

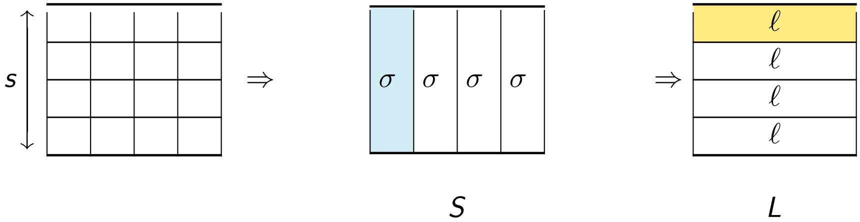


S1P mode of operation. K is the secret key, $A[i]$ are AD blocks, $M[i]$ are message blocks, $C[i]$ are ciphertext blocks, Z is the tag, $\tilde{E}$ is a TBC, π is a 384- or 512-bit permutation, N is a nonce, PK is a public key.

Only $\tilde{E}$ needs to be DPA-secure (masked).

Clyde ($\tilde{E}$)

Clyde is a Tweakable Block-Cipher based on an LS-design with S-box σ and L-box ℓ:



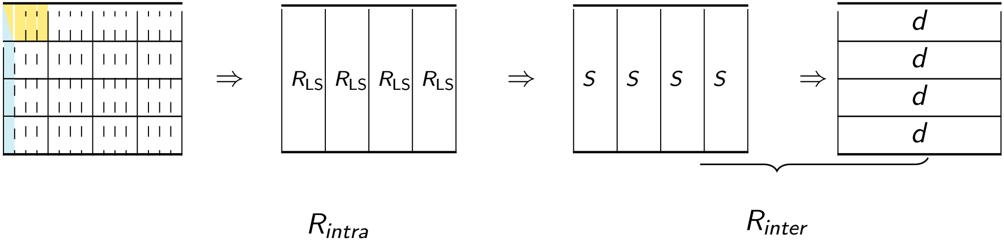
$R_{\text{LS}} = L \circ S \circ W$
with W a constant addition.
$\text{LS}(\sigma, \ell, r) = (R_{\text{LS}})^r$

Clyde is a tweakable-LS-design: tweakey addition is used every 2 rounds.

Clyde: 4-bit σ, 2x32-bit ℓ, $r = 12$

Shadow v2 (π)

Shadow is a multiple-LS-design (mLS-design) with S-box σ, L-box ℓ, D-box d:



$$R_{mLS} = R_{inter} \circ R_{intra}$$
$$\text{mLS}(\sigma, \ell, d, r) = (R_{mLS})^r$$

Shadow v2:

- 4-bit σ
- 2x32-bit ℓ
- 3x3 or 4x4 d, MDS
- $r = 6$

Table of Contents

Security Claims

4 settings: with or without leakage, single- or multi-user.

Single-user with leakage:

Integrity:

- setting: Ciphertext Integrity with nonce Misuse-resistance and Leakages in encryption and decryption (CIML2)
- security bits: $n - log(n)$
- model: leak-free Clyde, unbounded leakage on Shadow (Shadow state is public)

Confidentiality:

- setting: Chosen-Ciphertext Attack with nonce misuse-resilience and Leakage in encryption (CCAmL1)
- security bits: $n/2$
- model: leak-free Clyde, oracle-free and hard-to-invert leakage on Shadow

Security Claims

4 settings: with or without leakage, single- or multi-user.
Multi-user with leakage (max 2^{n-2} users):

Integrity:

- setting: Ciphertext Integrity with nonce Misuse-resistance and Leakages in encryption and decryption (CIML2)
- security bits: $n - 2log(n)$
- model: leak-free Clyde, unbounded leakage on Shadow (Shadow state is public)

Confidentiality:

- setting: Chosen-Ciphertext Attack with nonce misuse-resilience and Leakage in encryption (CCAmL1)
- security bits: $n/2$
- model: leak-free Clyde, oracle-free and hard-to-invert leakage on Shadow

Security Claims

4 settings: with or without leakage, single- or multi-user.

Single-user without leakage:

Integrity:
- setting: Ciphertext Integrity with nonce Misuse-resistance (CIM)
- security bits: $n - log(n)$

Confidentiality:
- setting: Chosen-Ciphertext Attack real-or-random with nonce-misuse resilience (CCAm\$)
- security bits: $n - log(n)$

Security Claims

4 settings: with or without leakage, single- or multi-user.
Multi-user without leakage (max 2^{n-2} users):

Integrity:
- setting: Ciphertext Integrity with nonce Misuse-resistance (CIM)
- security bits: $n - 2log(n)$

Confidentiality:
- setting: Chosen-Ciphertext Attack real-or-random with nonce-misuse resilience (CCAm$)
- security bits: $n - 2log(n)$

Mathematical Cryptanalysis Challenge 2

Attacks against:

- Clyde (6/8/10/12 rounds)
- Shadow v2 (6/8/10/12 rounds)
- Spook v2 mode (between 2/2 and 12/12 rounds)
- integrity with unbounded leakage (12-round Clyde, 2- to 12-round Shadow)

Deadline: February 28, 2021

Also Side-channel Challenges

Table of Contents

Conclusion

Spook is designed to be low-energy (software and hardware), to handle nonce-misuse and leakages.

A lot of information at `https://www.spook.dev`

- Specifications
- Implementations
- Benchmark articles
- Security articles
- Challenge specifications and results

Cryptanalysis is welcome! Belgian chocolate, beer + PacMan-bases prizes to earn!

Thank you for your attention!

Tag Verification

If SCA-secure receive $Z_{challenge}$, compute U, compute $\tilde{E}^{-1}(Z_{challenge})$, compare with U
that way the correct tag is never computed during tag verification
Note: implementing $\tilde{E}^{-1}$ has a small overhead

Otherwise compute Z, test equality with $Z_{challenge}$