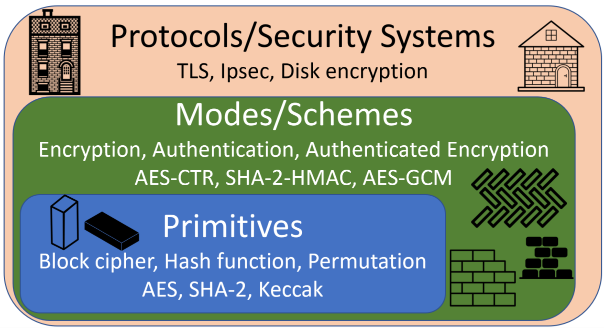
Expanding the Scope, Security and Efficiency of Classical Symmetric Primitives

Elena Andreeva, TU Wien, Austria

04/09/2025
GAPs Workshop, Singapore

Modular Approach to Building Cryptographic Systems

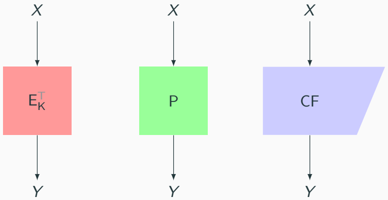
Modular aka Provable Approach



- \+ Bottom-up approach
- \+ **Primitives** over blocks $\Rightarrow$ **Schemes** on long data $\Rightarrow$ **Systems**
- \+ Reductionist security
- \+ Proofs allow to estimate min. adversarial attack resources (time, number/length of messages)

Symmetric-Key Primitives (Fixed-Size Block)

1. **Block cipher:** $\mathcal{K} \times \mathcal{X} \mapsto \mathcal{Y}$ where $|\mathcal{X}| = |\mathcal{Y}|$
   3-DES, AES
2. **Permutation:** $\mathcal{X} \mapsto \mathcal{Y}$ where $|\mathcal{X}| = |\mathcal{Y}|$
   Keccak in SHA3
3. **Compression function:** $\mathcal{X} \mapsto \mathcal{Y}$ where $|\mathcal{X}| > |\mathcal{Y}|$
   SHA2



Symmetric-Key (SK) Schemes for Secure Systems

SK primitives used for **classical**:

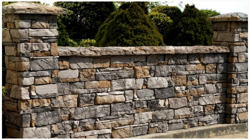
- **Encryption**
- **Message authentication**
- **Authenticated Encryption (AE)**

Modern security systems/protocols require SK primitives and modes with advanced and diverse security (NMR, BBB, RUP, MUS, CS, etc.), performance, cost requirements:

- TLS, IPSec, WiFi, DiskEnc, Key derivation, Multi-party Computation, Zero-Knowledge proofs, Blockchains, Messaging protocols (e.g, Signal, WhatsApp), etc.

Central Questions

1. Are we using the best-suited SK primitives in **modern** cryptographic systems?

 OR 

2. Are we not designing primitives/modes largely **in isolation**?

Why the Questions?

- Legacy: DES → 3-DES → AES (main application encryption)
  MD5 → SHA1 → SHA2 (main application integrity)

- More recent: SKINNY , DEOXYS TBCs, PRINCE, etc.
- Typical security in schemes:
  - AES 128-bit block → 64-bit security
  - PRINCE 64-bit block → 32-bit security
  + DEOXYS 128-bit block, 128-bit tweak → 128-bit security
  - Robust security harder with classical primitives
- Cost and suitability
  - DEOXYS vs AES → 14 vs 10 internal AES rounds
  - Typical AE overhead: 1BC per block to encrypt + 2 extra BC or 1 multiply per block for authentication
  - Hash function (HKDF) for randomness extraction ...

Goal: Primitives that are suited to modern system functionalities with adequate security at minimal cost

Towards Novel Primitives

1. Tweaks and innate **robustness** features – BBB, RUP, NMR, MUS, length-independence, block-wise atks or SC resistance, etc.
2. **Well-cryptanalyzed** structure – SPN, Tweakey, IEM, Sum of Ps
3. **Expansion** better-suited for $ Extraction/Enc/AE/etc.
4. **Improved performance**: internal/external parallelism, optimized for long/short M, high throughput, etc.
5. αn-bit expansion need be more **efficient** than α TBCs of n-bit

Forkcipher
Fixed-Output-Length (FOL)
Tweakable Expanding Function[a]

---

[a]E. Andreeva, V. Lallemand, A. Purnal, R. Reyhanitabar, A. Roy, D. Vizàr
ForkAE, second round in NIST LW Standardization, 2019
Forkcipher: A New Primitive for Authenticated Encryption of Very Short Messages, ASIACRYPT'19

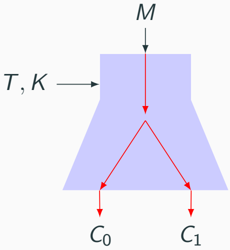
Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

K: secret and random

T: public tweak

Forward execution ($2n$-bit output)



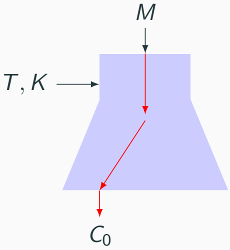
Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

K: secret and random

T: public tweak

Output selection C_0 (left n-bit output = TBC)



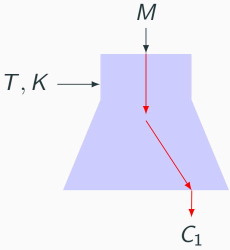
Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

K: secret and random

T: public tweak

Output selection C_1 (right n-bit output = TBC)



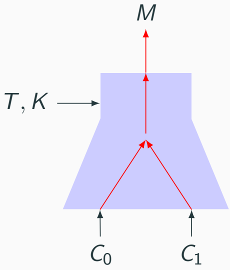
Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

K: secret and random

T: public tweak

Inversion from either one or both C_0 and C_1



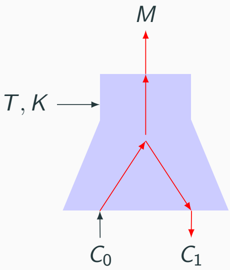
Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

K: secret and random

T: public tweak

Reconstruction from C_0 to C_1 through forking point



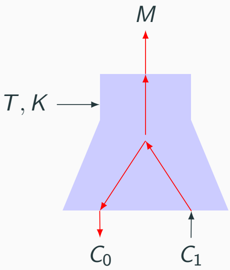
Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

K: secret and random

T: public tweak

Reconstruction from C_1 to C_0 through forking point

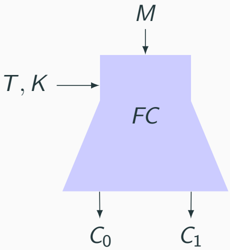


Forkcipher: FOL Tweakable Expanding Function

$FC : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_0 \times \mathcal{C}_1$ with $|M| = |C_0| = |C_1| = n$

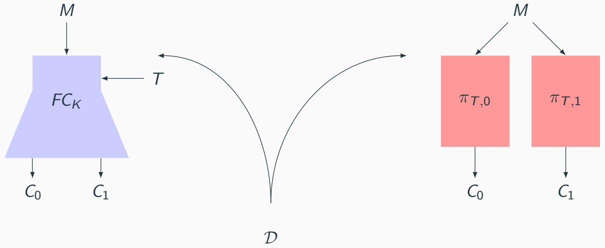
K: secret and random

T: public tweak



- ✓ Primitive generalizes to muti-forkcipher MFC
- ✓ Each branch is invertible $\Rightarrow$ output **invertible**
- ✓ Target *good* **definition** for secure randomness generation for Enc/AE/key derivation, etc.)?

Forkcipher Security Target



PseudoRandom Tweaked Forked Permutation PRTFP

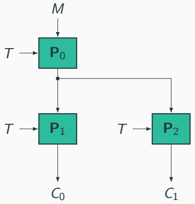
Indistinguishability from a pair of random, tweaked π_0, π_1 under chosen ciphertext attack

$$Adv_{FC}^{prtfp}(\mathcal{D}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{D}^{FC_K} \Rightarrow 1] - \Pr[\mathcal{D}^{\pi_0,\pi_1} \Rightarrow 1].$$

IFI: Iterate-Fork-Iterate Modular Forkcipher Framework

n-to-$2n$-bit **IFI** : $\mathcal{K} \times \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^{2n}$

- ✓ $\mathbf{P_0}, \mathbf{P_1}, \mathbf{P_2}$ independent, random $\Rightarrow$ $2n$-**bit expansion**
- ✓ $\mathbf{P_0} \rightleftarrows \mathbf{P_1}$ and $\mathbf{P_0} \rightleftarrows \mathbf{P_2}$ **invertibility**



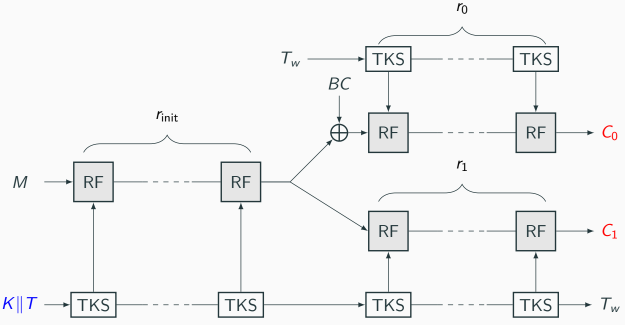
IFI is indistinguishable from independent, random π_1, π_2

$$Adv_{\mathbf{IFI}}^{\mathbf{PRTFP}}(\mathcal{D}) = 0.$$

Forkcipher Instance: ForkSkinny

Based on ISO/IEC 18033-7 standard SKINNY TBC components

$2n$-bit ciphertext of 1 FORKSKINNY $\approx$ 1,6 SKINNY (not 2)



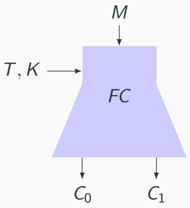
RF: round function; TKS: tweakey schedule; BC: branch constant; r_{init}, $r_0 = r_1$: nr rounds before and after fork.

Primitive	block	tweak	tweakey	r_{init}	r_0	r_1
FORKSKINNY-64-192	64	64	192	17	23	23
FORKSKINNY-128-192	128	64	192	21	27	27
FORKSKINNY-128-256	128	128	256	21	27	27
FORKSKINNY-128-288	128	128	288	25	31	31

Generic Designs of FOL Expanding Functions

- Forkcipher[n-to-$2n$] from IFI as FOL-PRF with $n/2$-bit security
- ForkSTH[n-to-$n+s$] [Dutta et al.'22] from STH as FOL-PRF with $(n - s/2)$-bit security
- FTEM[n-to-$2n$] with $r = 2$ [Kim et al.'20] is $2n/3$-bit PRTFP
- F1 and F2 [Mandal'25] full n-bit security
- FEM[n-to-$2n$] for any r is $rn/(r+1)$-bit secure PRFP and FTEM-ITS[n-to-$2n$] for any r is $rn/(r+1)$-bit secure PRTFP [Andreeva et al.'25]

Forkciphers Contd



– Reconstruction ⇒ Partial or full **inversion not always needed** for Enc and Dec, randomness generation, etc.

\+ VOL-PRF: MFC, ForkEDMD, ForkCENC, Hydra, Ciminion, Farfalle, etc.

\+ Benefits of tweak and >**2-block** expansion ⇒ Butterknife VOL-TPRF

ButterKnife VOL-Tweakable Expanding Function[a]

[a]E. Andreeva, B. Cogliati, V. Lallemand, M. Minier, A. Purnal, A. Roy
Masked Iterate-Fork-Iterate: A New Design Paradigm for Tweakable Expanding Pseudorandom Function, ACNS, 2024

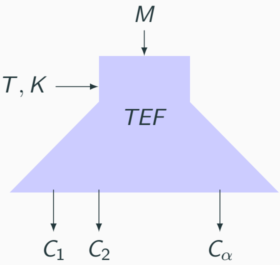
VOL-Tweakable Expanding Function

$TEF : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_1 \times \mathcal{C}_2 \times \ldots \times \mathcal{C}_\alpha$

$|K| = |T| = |M| = |C_1| = \ldots = |C_\alpha| = n$

K: secret and random T: public tweak

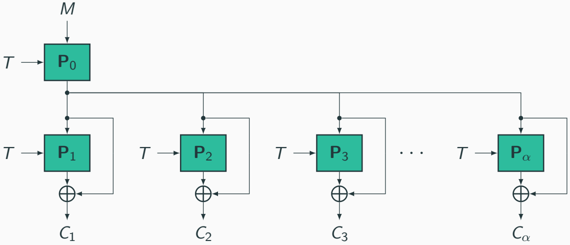
Forward-only evaluation



mIFI: Masked IFI Modular VOL-TPRF Framework

n-to-αn-bit **TEF** : $\mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^{\alpha n}$

- ✓ $\mathbf{P_0} \ldots \mathbf{P}_\alpha$ independent, random $\Rightarrow$ αn-**bit expansion**
- ✓ $\mathbf{P_0}$ state $\oplus$ masks all output branches $\Rightarrow$ **no invertibility**



A TEF following mIFI is n-bit secure VOL-TPRF

mIFI is indistinguishable from a uniformly random αn-bit string

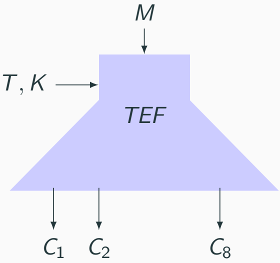
$$Adv^{\textbf{VOL-TPRF}}_{\textbf{mIFI}}(\mathcal{D}) \leq \frac{\sqrt{2\alpha} q}{2^n}.$$

ButterKnife mIFI Instance

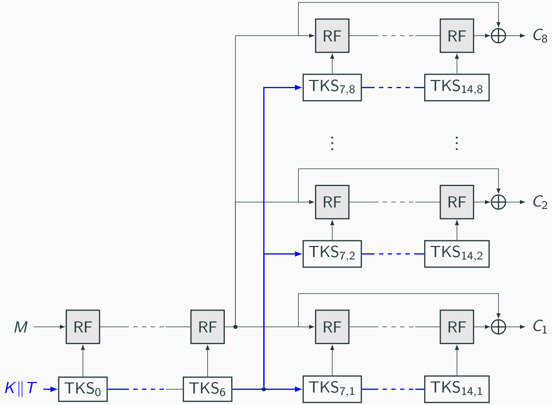
$$\text{BUTTERKNIFE} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \mapsto \mathcal{C}_1 \times \mathcal{C}_2 \times \ldots \times \mathcal{C}_8$$

$$|K| = |T| = |M| = |C_1| = \ldots = |C_8| = 128$$

- Based on AES-based DEOXYS-TBC-256 (ISO standard)
- 7 DEOXYS-TBC-256 rounds until state forks
- 8 more rounds after fork in each $\alpha = 8$ **parallel** branches
- **$8n$-bit ciphertext** for 1 BUTTERKNIFE $\approx$ 5 DEOXYS (not 8)



ButterKnife Structure



RF: round function; TKS: tweakey schedule; r_{init}, $r_1 = \ldots = r_8$: nr rounds before and after fork, resp.

Primitive RF&TKS	block	tweak	tweakey	r_{init}	$r_1, \ldots r_8$
Deoxys-256	128	128	256	7	8

Security and Performance Advantages of (FOL/VOL-)TEFs for

1. ForkAE: 2nd round in LW **AEAD** NIST standardization, 2019
2. Forkcipher: New Primitive for **AEAD** of Very Short Ms, ASIACRYPT'19
3. SAEF: **AEAD** Mode with OAE and RUP-OAE security, SAC'21, SCN'24
4. Let's Go Eevee! A Friendly and Suitable Family of **AEAD** Modes for IoT-to-Cloud Secure Computation, ACM CCS'23
5. Masked Iterate-Fork-Iterate: A New Design Paradigm for Tweakable Expanding Pseudorandom Function, Deterministic **AE**, ACNS'24
6. 1,2,3,Fork: CTR **Encryption** Variants Based on Generalized FC, ToSC'21
7. PAE: Towards More Efficient&BBB-secure **AE** From a Single Public Permutation, ICICS 2023 8. Forkcipher-Based **PRNG**, ACNS'23
9. Skye: Fast **KDF** from Expanding PRF& Application to Signal, AsiaCCS'24
10. Light**MAC**, Adv. in Mathematics of Communications'23
11. Sonikku: Gotta Speed, Keed! Family of Fast&Secure **MAC**s, CANS'25
12. FEDT: Forkcipher-based **Leakage-resilient** BBB-secure **AE**, CiC'24
13. Authenticity **AE**&**MAC** in Presence of **Leakage** using Forkcipher, CiC'25
14. **TEM** - in progress, 2025

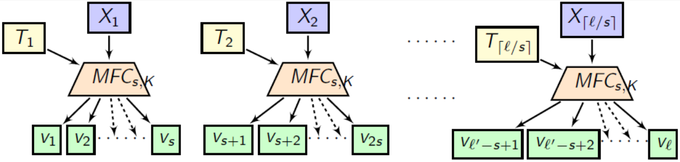
TEF $\underset{\text{Proof}}{\Longrightarrow}$ LW/Cloud/Det AEAD, ENC, PRNG, KDF, MAC, SC resistance, ...

Encryption (GCTR) [a]

[a] E. Andreeva, Bhati, D. Vizár
1,2,3,Fork: CTR Encryption Variants Based on Generalized Forkcipher, ToSC'21

Towards Encryption with TEF via TCTR

1. Apply TEF on inputs (X_j, T_j) where:
   - (X_j, T_j) are output by $f_X(N, R, j)$ and $f_T(N, R, j)$
   - f_X, f_T: simple f-ns $\{\|, \oplus, \mathrm{copy}\}$, N is nonce, R random value and j counter
2. TCTR generates random key-stream V

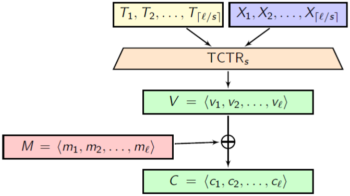


vs

Random key stream

General Tweakable Counter Mode (GCTR) Encryption

Apply the random key-stream V on M



GCTR$_s$ Encryption: $\mathcal{E}_K(N, R, M)$

```
for j ← 1 to ⌈ℓ/s⌉ do
    X_j = f_X(N, R, j)
    T_j = f_T(N, R, j)
T, X ← ⟨T_1, ..., T_⌈ℓ/s⌉⟩, ⟨X_1, ..., X_⌈ℓ/s⌉⟩
V ← TCTR_{s,K}(T, X)
C ← M ⊕ V
return C
```

- 22 variants with security from $n/2$ to **full** n**-bit** + **NMR**
- Recommend: GCTR-3 with $T_j = R \oplus j$, $X_j = N$, and GCTR-7 with $T_j = N \| j$, $X_j = R$

GCTR-ForkSkinny Efficiency

- The cost to compute 1 keystream n-bit block with GCTR-ForkSkinny is ≈ **0.8** of cost with GCTR-Skinny
- GCTR-ForkSkinny becomes **twice** faster than GCTR-Skinny using primitive level parallelism
- TEF evaluated *only* forward: possible instances with ButterKnife

AE(AD) [a]

[a] E. Andreeva, V. Lallemand, A. Purnal, R. Reyhanitabar, A. Roy, D. Vizàr
ForkAE, second round in NIST LW Standardization, 2019
Forkcipher: A New Primitive for Authenticated Encryption of Very Short Messages, ASIACRYPT'19

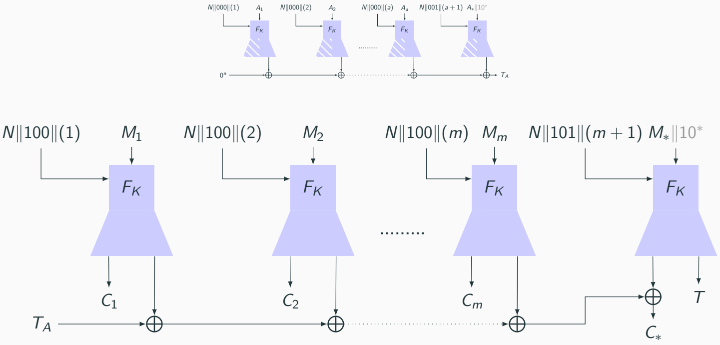
E. Andreeva, B. Cogliati, V. Lallemand, M. Minier, A. Purnal, A. Roy
Masked Iterate-Fork-Iterate: A New Design Paradigm for Tweakable Expanding Pseudorandom Function, ACNS, 2024

ForkAE

ForkAE family of AEAD schemes:

- Parallel mode: **PAEF** and **rPAEF**
- Sequential mode: **SAEF**

- **Rate-1**: 1 primitive call to Enc and Auth 1 block and no authentication overhead $\Rightarrow$ well-suited for short data

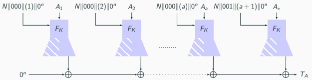


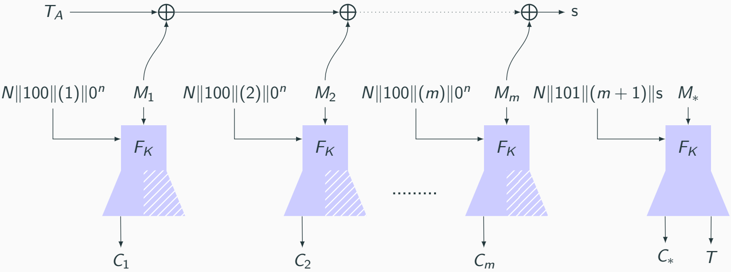
n-bit AE security

$$Adv_{PAEF}^{privacy}(\mathcal{A}) \leq Adv_{FC}^{PRTFP}(\mathcal{D})$$

$$Adv_{PAEF}^{auth}(\mathcal{A}) \leq Adv_{FC}^{PRTFP}(\mathcal{D}) + \frac{q_v \cdot 2^n}{(2^n - 1)^2}$$

Reduced Parallel AEAD from a Forkcipher: rPAEF

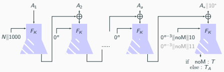


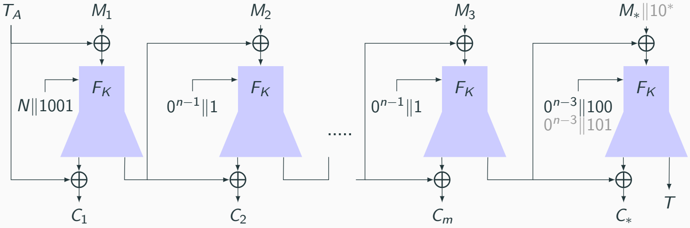


n-bit AE security

$$Adv^{privacy}_{rPAEF}(\mathcal{A}) \leq Adv^{PRTFP}_{FC}(\mathcal{D})$$

$$Adv^{auth}_{rPAEF}(\mathcal{A}) \leq Adv^{PRTFP}_{FC}(\mathcal{D}) + \frac{q_v \cdot 2^n}{(2^n - 1)^2}$$





$n/2$-bit AE security but also NMR and and OAE-RUP secure

$$Adv^{privacy}_{SAEF}(\mathcal{A}) \leq Adv^{PRTFP}_{FC}(\mathcal{D}) + 2\frac{(\sigma - q)^2}{2^n}$$

$$Adv^{auth}_{SAEF}(\mathcal{A}) \leq Adv^{PRTFP}_{FC}(\mathcal{D}) + \frac{2(\sigma - q + 1)^2}{2^n} + \frac{\sigma(\sigma - q)}{2^n} + \frac{q_v(q+2)}{2^n}$$

ForkAE Family Performance

- Rate-1: 1 ForkSkinny call per data block
- Excellent performance for **short messages**: ($\leq$ 4 blocks) (r)PAEF and SAEF 25% to 58% better in SW than Skinny-AEAD
- rPAEF better than Skinny-AEAD in SW for **all data sizes**
- Multiple **trade-offs in speed-resource design space** and excellent throughput per area in HW
- (r)PAEF: ***n*-bit** secure
- SAEF: **OAE-RUP** and **NMR** secure
- (r)PAEF further speed-up possible with ButterKnife

Deterministic Authenticated Encryption (DAE)

Authenticity, **Confidentiality** + **Same** $K, M \Rightarrow$ **Same** C

- Key wrap, search on encrypted data, LW etc.

  Challenges: Increase speed & security with classical primitives: 2 data passes (1 Enc and 1 Auth) GCM-SIV, SCT, ZAE, etc.

  SAFE = ButterKnife-CTR Enc + Polynomial-based hash

  ZAFE = ButterKnife-CTR Enc + ZMAC

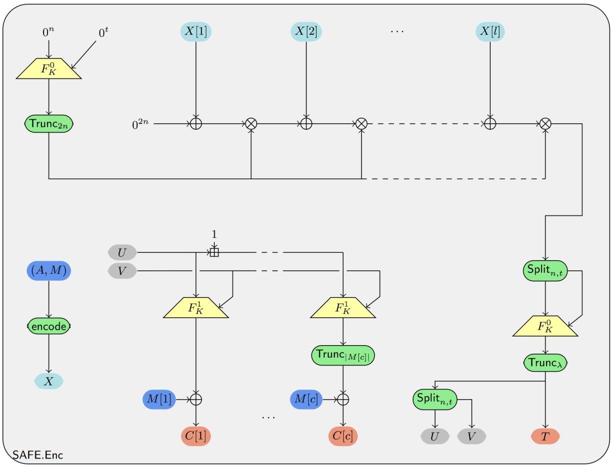
✓ Prallelizable and full 128-bit security

✓ **Best performance** among *equally* 128-bit secure existing DAE schemes ($>$ 25% ZAE and $>$ 50% Enc speed-up)

✓ **Comparable performance** with *less* 64-bit secure DAE

SAFE Deterministic Authenticated Encryption

SAFE: fast TPRF-based CTR encrypt. + polynomial-based hash



$\frac{n+\min(n,t)}{2}$**-bit DAE security**

DAE SW Implementations

Table 1: DAE implementations in cycles per byte (c/B) for long M (64KiB), decoupled **Auth** and **Enc**, and **Total**=**Enc**+**Auth**

		Skylake 3.2 GHz			Cascade Lake SP 2.7 GHz		
	Sec	**Auth**	**Enc**	**Total**	**Auth**	**Enc**	**Total**
AES-GCM-SIV	64	**0.30**	0.63	$\boxed{\mathbf{0.93}}$	**0.31**	0.63	$\boxed{\mathbf{0.94}}$
SCT	64	0.87	0.87	1.74	0.87	0.87	1.74
ZAE (256)	128	0.61	0.87	1.48	0.61	0.87	1.48
ZAE (384)	128	0.59	0.99	1.58	0.59	0.99	1.58
ZAE (256-384)	128	0.59	0.87	1.46	0.59	0.87	1.46
SAFE	128	0.63	$\boxed{\mathbf{0.55}}$	$\boxed{\mathbf{1.18}}$	0.62	$\boxed{\mathbf{0.55}}$	$\boxed{\mathbf{1.17}}$
ZAFE	128	0.60	$\boxed{\mathbf{0.55}}$	$\boxed{\mathbf{1.15}}$	0.60	$\boxed{\mathbf{0.55}}$	$\boxed{\mathbf{1.15}}$

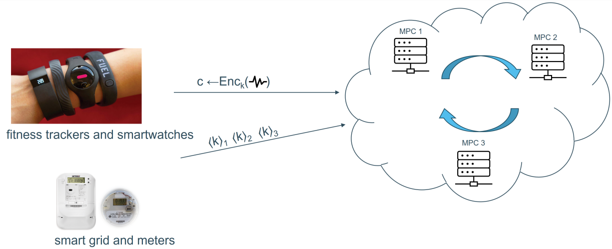
Throughput increase:

- ✓ $\approx$ 58% over ZAE **Enc** due to ButterKnife TPRF
- ✓ $\approx$ 24% over ZAE **Total**
- ✓ ButterKnife in CTR mode faster than AES (0.55 vs 0.63 c/B)

IoT-to-Cloud AEAD[a]

[a] A. Bhati, E. Pohle, A. Abidin, E. Andreeva, B. Preneel, Let's Go Eevee! A Friendly and Suitable Family of AEAD Modes for IoT-to-Cloud Secure Computation, ACM CCS'23

IoT-to-Cloud Setting



IoT: Client encrypts data (AEAD) and sends key shares

- AEAD: small area, small state, RAM, energy/power-efficient

- Large attack surface ⇒ NMR, block-wise, BBB security

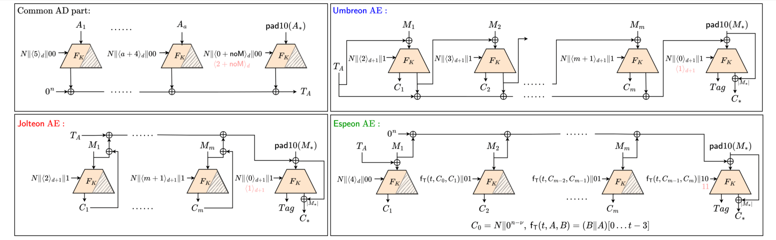
Cloud: MPC servers jointly decrypt and compute f-n F over data

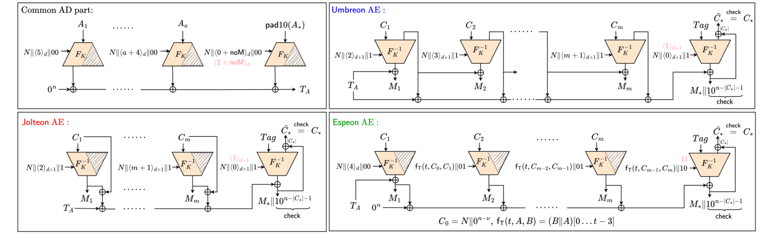
- Low-depth circuit & **parallelizable** Dec (to reduce communication cost)

- Independent of party shares

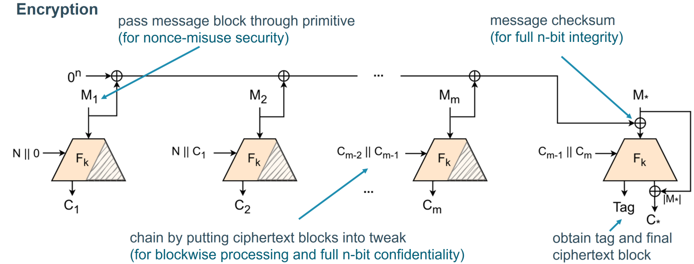
- CTR-PMAC, CTR-HtMAC for Cloud, (Rotaru et al., ToSC'18)

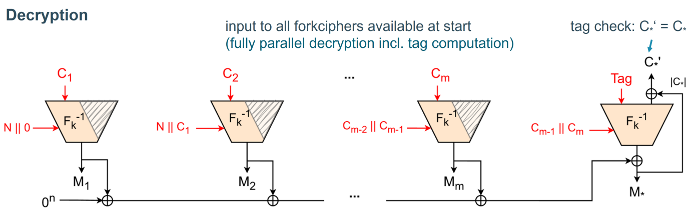
Eevee AEAD Family for IoT-to-Cloud Computation



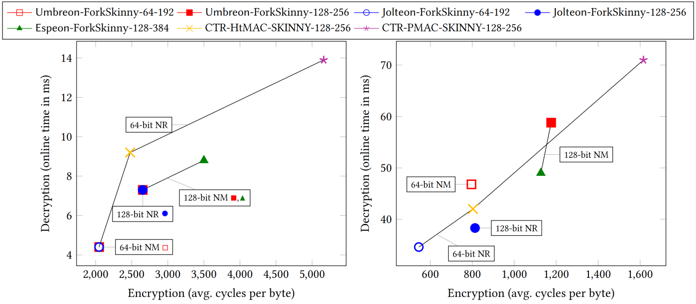


Espeon AEAD Family for IoT-to-Cloud





Eevee AEAD: Performance



(a) Message length = 8 byte. **(b) Message length ≥ 100 byte.**

Setting: 3 parties, < 1 ms latency, Cortex-M4

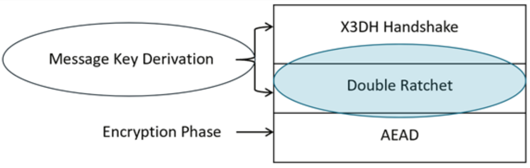
- Jolteon compared to CTR-HtMAC: 32% to 45% speed-up IoT-Enc 16% to 66% MPC-Dec throughput for 8B/100B data, no NMR
- Espeon: similar performance but with tweak-size-dependent NMR
- Umbreon: strongest security guarantees - full *n*-bit OAE security

Key Derivation [a]

[a] A.S. Bhati, A. Dufka, E. Andreeva, A. Roy, B. Preneel Skye: An Expanding PRF based Fast KDF and its Applications, AsiaCCS'24

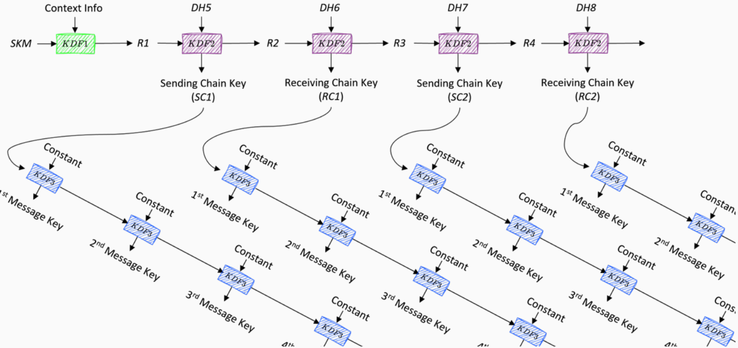
Key Derivation Function in Signal

- KDF generates uniform&random stream from weakly random K
- Singal uses HKDF



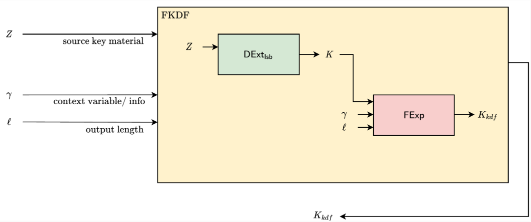
HKDF in Signal

- HKDF follows extract-then-expand paradigm
- HKDF is defined for general sources, not optimal for source-specific derivation, e.g. Diffie-Hellman in Signal
- HKDF is hghly sequential
- Proof relies on RO (SHA-2)



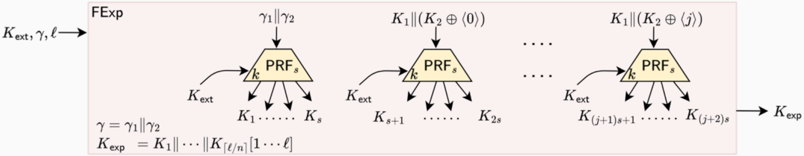
The Skye Key Derivation Function

1) Dedicated **source-specific randomness extractor**: DExt
2) **Randomness expansion** with TEF (ButterKnife): FExp



The Skye Key Derivation Function: Expansion

1) Dedicated **source-specific randomness extractor** : DExt
2) **Randomness expansion** with TEF (ButterKnife): FExp



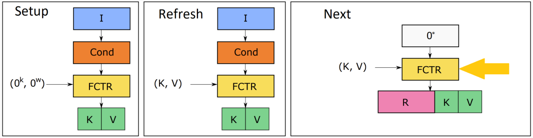
- parallelizable, 128-bit security in standard model

- Skye in Signal: 38% to 64% speed-up in unidirectional messaging, and 12-36% when 10 messages are sent and received at once

PRNG (FCRNG) [a]

[a] E. Andreeva and A. Weninger
A Forkcipher-Based Pseudo-Random Number Generator, ACNS'23

- NIST existing RNG standard CTR-DRBG, SP 800-90A
- Work with GCTR-style (FCTR) for randomness derivation
- Security proof *does not* degrade with the length of random inputs nor amount of requested pseudorandom bits
- 33% to 50% faster than CTR-DRBG standard with ForkSkinny and ButterKnife, resp.



Conclusions and Open Questions

Tweakable Expanding Primitives:

- More efficient and secure replacement for many classical symmetric schemes
- But also well-suited for modern and emerging applications
- Will benefit from new instances (low-latency, no tweak, algebraic, etc.), new design frameworks and cryptanalysis
- Can TEF support different design functionalities than PRF, PRTFP, or emerging privacy-friendly applications?

Thank you!

elena.andreeva@tuwien.ac.at

Bibliography 1

1. E. Andreeva, V. Lallemand, A. Purnal, R. Reyhanitabar, A. Roy, D. Vizàr
ForkAE, second round in NIST LW Standardization, 2019

2. E. Andreeva, V. Lallemand, A. Purnal, R. Reyhanitabar, A. Roy, D. Vizàr
Forkcipher: A New Primitive for Authenticated Encryption of Very Short Messages, ASIACRYPT'19

3. A. Dutta, J. Guo, E. List
Forking sums of permutations for optimally secure and highly efficient PRFs, DCC'25

4. H. Kim ,Y. Lee, J. Lee
Forking Tweakable Even-Mansour Ciphers, ToSC'20

5. S. Mandal
Tweakable ForkCipher from Ideal Block Cipher, CIC'25

6. E. Andreeva, A.S. Bhati, A. Weninger
Multiforked Iterated Even-Mansour and a Note on the Tightness of IEM Proofs, SAC'25

7. E. Andreeva, B. Cogliati, V. Lallemand, M. Minier, A. Purnal, A. Roy
Masked Iterate-Fork-Iterate: A New Design Paradigm for Tweakable Expanding Pseudorandom Function, ACNS, 2024

8. A.S. Bhati, E. Andreeva, D. Vizàr
OAE-RUP: A Strong Online AEAD Security Notion and its Application to SAEF, SCN'24

9. A.S. Bhati, E. Andreeva, D. Vizàr
Nonce-Misuse Security of the SAEF Authenticated Encryption mode, SAC'21

10. A.S. Bhati, E. Pohle, A. Abidin, E. Andreeva, B.Preneel
Let's Go Eevee! A Friendly and Suitable Family of AEAD Modes for IoT-to-Cloud Secure Computation, ACM CCS'23

Bibliography 2

11. A.S. Bhati, E. Andreeva, D. Vizár, B. Preneel 1, 2, 3, Fork: Counter Mode Variants based on a Generalized Forkcipher, ToSC'21

12. E. Andreeva, A. Weninger,
Forkcipher-Based Pseudo-Random Number Generator, ACNS'23

13. A.S. Bhati, A. Dufka, E. Andreeva, A. Roy, B. Preneel
Skye: An Expanding PRF based Fast KDF and its Applications, AsiaCCS'24

14. N. Datta, A. Dutta1, ,C.M.-Lopez
LightMAC : Fork it and make it faster, Adv. in Mathematics of Communications'23

15. A.S. Bhati, E. Andreeva, S. Müller, D. Vizŕ
Sonikku: Gotta Speed, Keed! Family of Fast and Secure MACs, CANS'25

16. N. Datta, A. Dutta, E. List, S. Mandal
FEDT: Forkcipher-based Leakage-resilient BBB-secure AE, CIC'24

17. F. Berti, F.X. Standaert I. Levi
Authenticity in the Presence of Leakage using a Forkcipher, CIC'25

18. A. Bhattacharjee1, R. Bhaumik, A. Dutta, E. List
PAE: Towards More Efficient and BBB-secure AE From a Single Public Permutation, ICICS 2023

19. D. Rotaru, N.P. Smart, and M. Stam
Modes of Operation Suitable for Computing on Encrypted Data, FSE'18

Figures credits to: A.S. Bhati, A. Roy, A. Weninger, B. Cogliati, E. Andreeva