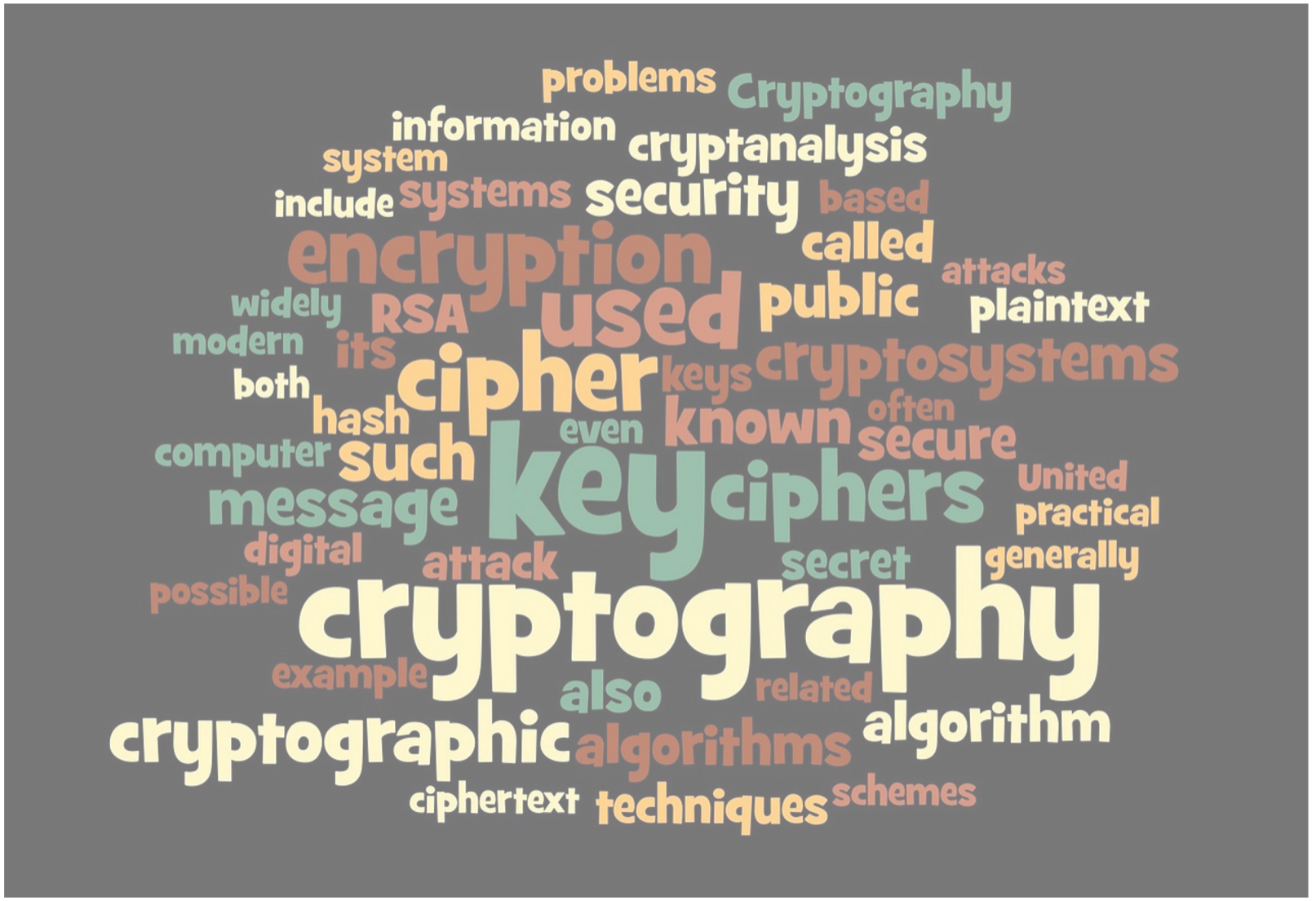
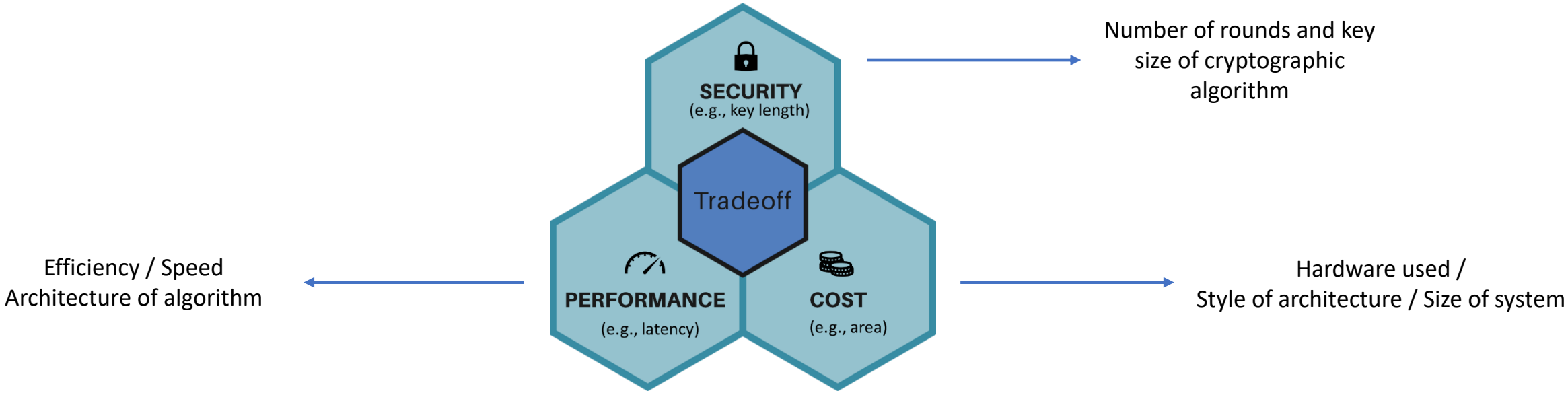


Secure Computing (CP8301)

Final Project - ESTATE

BY: Urmi Patel

Lightweight Cryptography



- An encryption method used to run on different devices with very low computing power
- Trade-off between lightweightness and security
- Extremely hard to achieve all three goals

ESTATE

- **E**nergy efficient and **S**ingle-state **T**weakable block cipher based M**A**C-**T**hen-**E**ncrypt
- Improved version of SUNDAE (2018)
- ESTATE used:

FCBC	->	authentication
OFB	->	encryption
TweAES	->	tweakable block cipher
TweGIFT	->	tweakable block cipher
AEAD	->	encryption mode

SUNDAE

- **S**mall **Un**iversal **D**eterministic **A**uthenticated **E**ncryption
- MAC-then-Encrypt type scheme
  - MAC -> OMAC
  - OFB -> encryption
- It is nonce-misuse resistant but does not provide RUP security
- Also requires an additional primitive calls
- SUNDAE designs have little overhead:

  Besides an n-bit state, it only requires two XORs block and one or two finite field multiplications with a constant per message.

Motivation For ESTATE Creation

Goal: Design a block cipher based lightweight AEAD

- Optimal state size like SUNDAE → Small as block size
- Nonce-misuse resistance like SUNDAE → Provides full security even when the nonce can be repeated

Additional :

- Optimal number of primitive calls → Almost zero additional calls
- Multiplication free construction → Doesn't require field multiplication
- INT-RUP secure → Adversary should not be able to forge, even when given access to unverified plaintext

Requirement For A Block Cipher Based Lightweight AEAD

Small state size

- Helpful in low area hardware implementation
- Memory constraint

Energy-efficient design

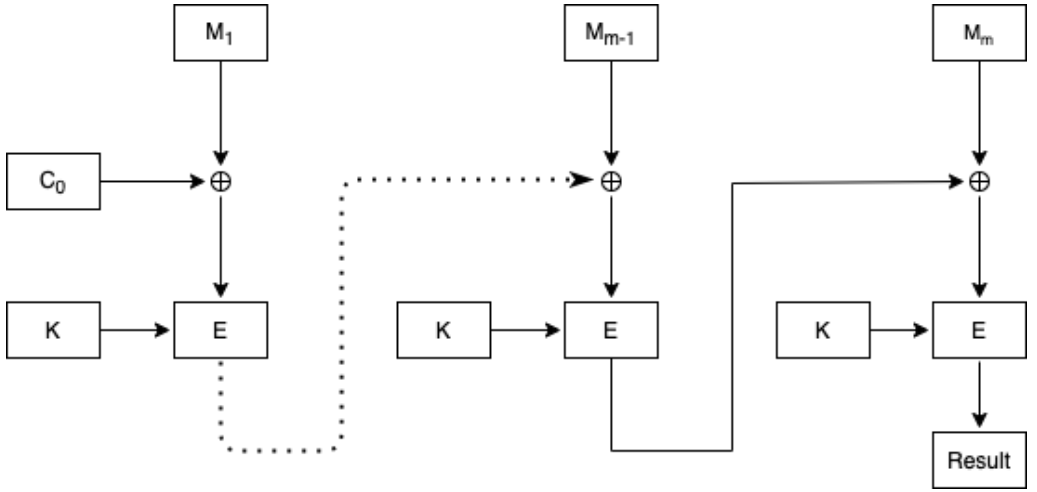
- Energy consumption by algorithm
- Efficient process of system

Defense in depth

- Construction must be robust
- Nonce-misuse resistance
- RUP security

CBC-MAC

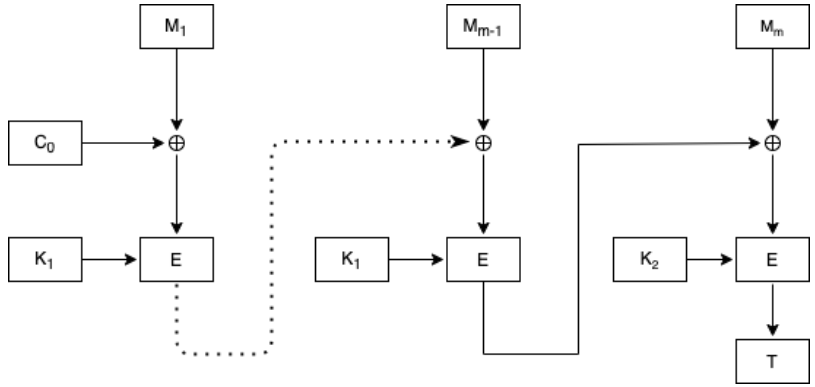
- Variant of CBC MAC (Cipher Block Chaining Message Authentication Code)
  - Construct a secure MAC for short and fixed length messages based on any block cipher.
  - Only the final output is the result, so verification is done by re-computing the result only.



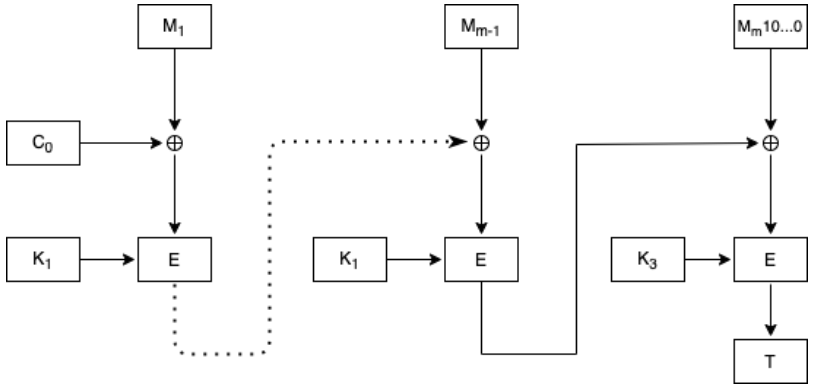
- When message lengths vary, the CBC MAC is not secure
- CBC MAC must be embellished using EMAC (Encrypted Message Authentication Code)

FCBC

- FCBC is the refinement of EMAC
- Difference is the number of keys used and how the last block of the message is treated.



Without padding , for last block use key K_2

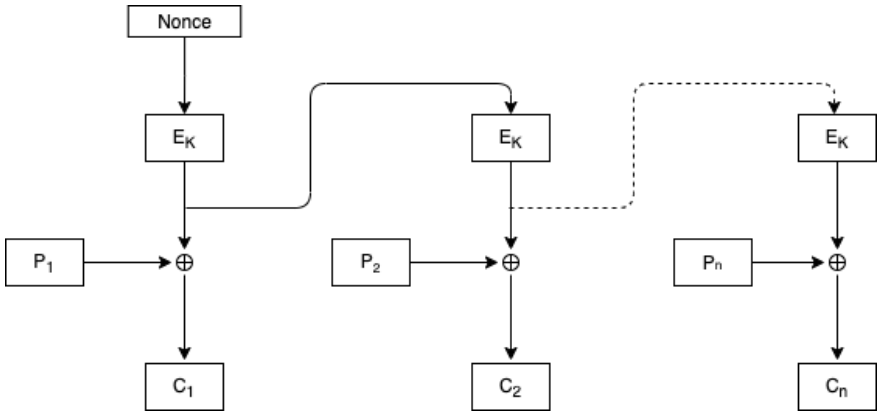


In case of padding , for a last block use key K_3

- Can not add '000' in padding because, message are same after padding zero.
- Idea is pad with '100..00' where 1 indicates beginning of pad.
- One can also add new dummy block, but for FCBC no dummy blocks are required, ambiguity resolved by use of K_2 and K_3.

OFB

- Output feedback mode
- Output of encryption process is directly placed in next stage of shift register without XOR operation.



- If there is any bit error out in the output of the encryption algorithm, it will not affect the C.T.
- This mode provides confidentiality by use of nonce as an initial vector (IV).
- Nonce -> value must be unique for each execution of the algorithm

  -> nonce helps to increase the complexity of the algorithm

AEAD

AE

Authenticated encryption

Provides privacy and confidentiality

AD

Associated data

Provides only integrity

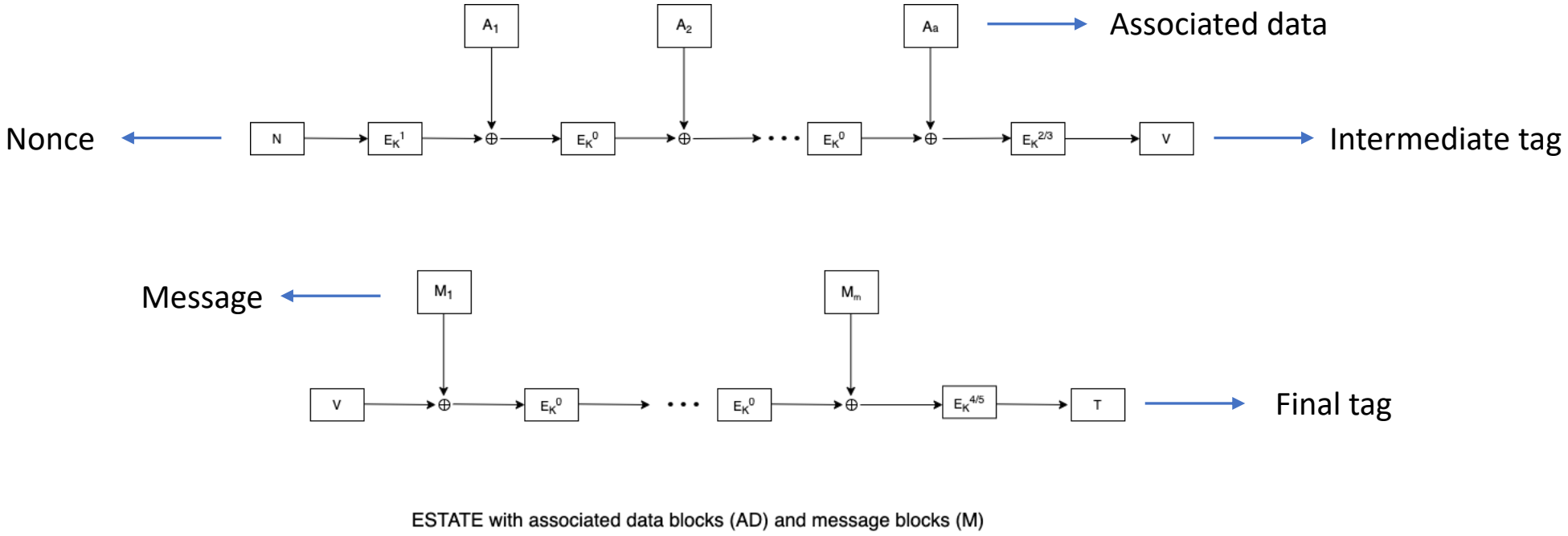
AE + AD = more security

AEAD Approach For ESTATE

- ESTATE uses AEAD approach and receives:
    - encryption key, K
    - a nonce, N
    - an associated data, A
    - a message, M
- Returns
    - a ciphertext, C
    - a tag, T

AEAD Approach For ESTATE

Case 1

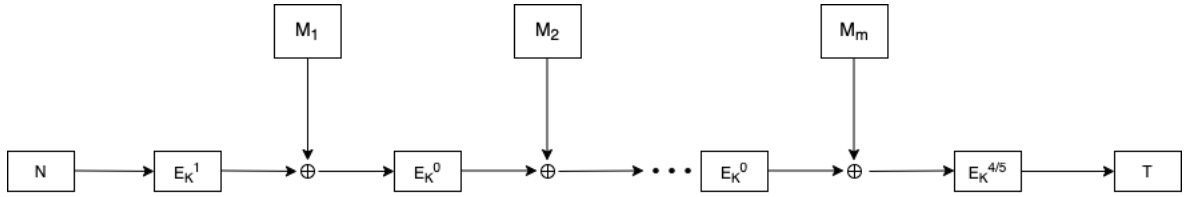


ESTATE with associated data blocks (AD) and message blocks (M)

- Encrypt the nonce using tweak value 1
- 0 has been used to process all the associated data and messages
- Last call is differentiated with tweaks 2,3 or 4,5 depending upon whether the data is full or partial

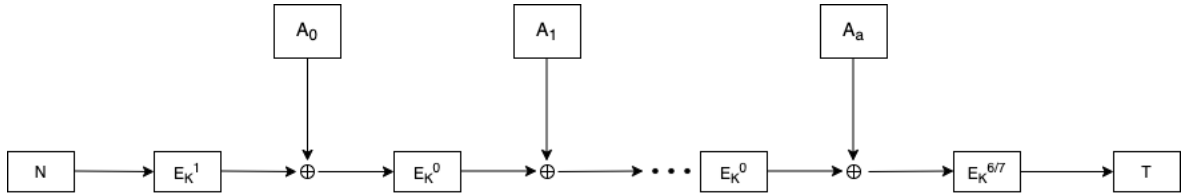
AEAD Approach For ESTATE

Case 2



ESTATE with message blocks (M) and empty AD blocks

Case 3



ESTATE with AD blocks and empty message blocks (M)

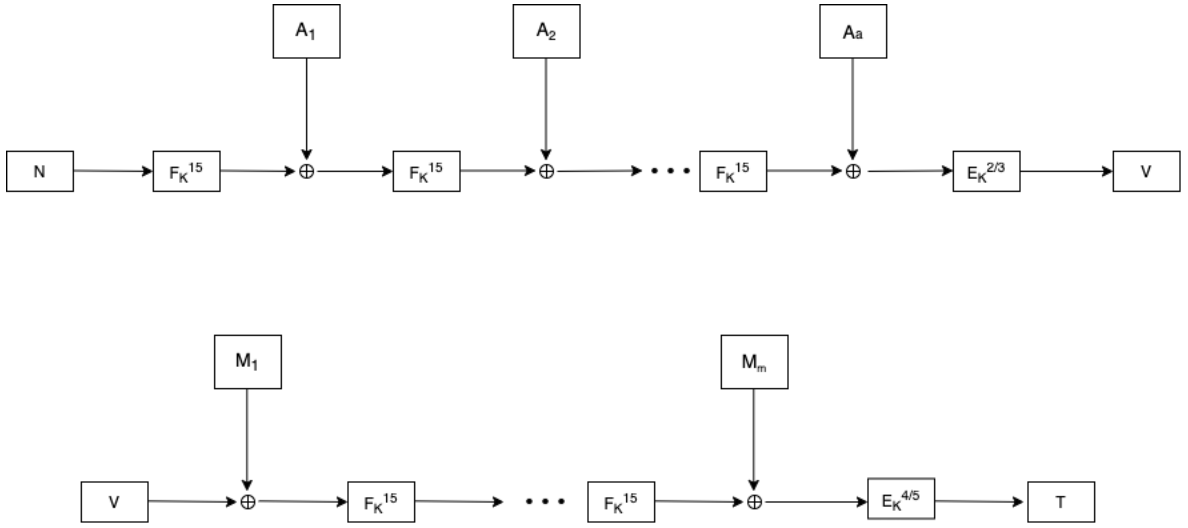
Case 4

Simply use unique tweak 8 to encrypt the nonce and that is the tag value $T := E_K{}^{8}(N)$

AEAD Approach For sESTATE

- sESTATE is a smaller version of ESTATE
- It replaces the full block cipher E with round reduce block cipher F, that always uses tweak 15
- But the last block uses full block cipher

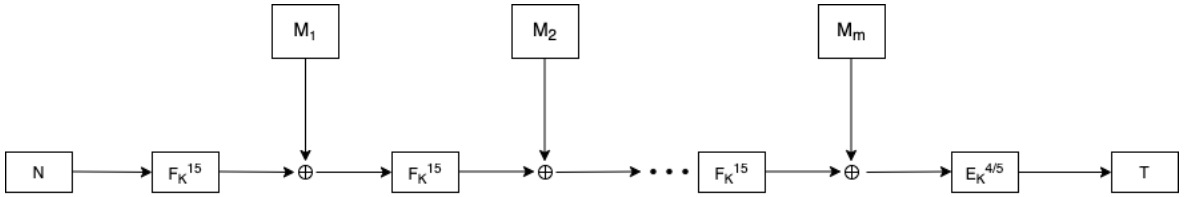
Case 1



sESTATE with Associated data blocks (AD) and message blocks (M)

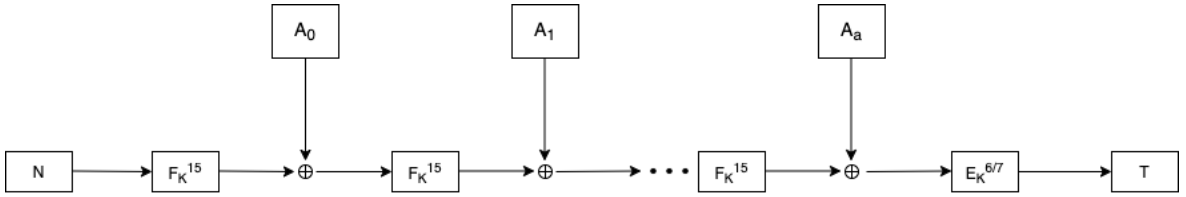
AEAD Approach For sESTATE

Case 2



sESTATE with message blocks (M) and empty AD blocks

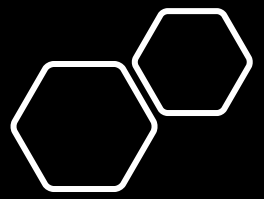
Case 3



sESTATE with AD blocks and empty message blocks (M)

Case 4

Simply use unique tweak 8 to encrypt the nonce and that is the tag value $T := E_K{}^{8}(N)$



Choice Of Tweaks

ESTATE -> 0 - 8 tweaks

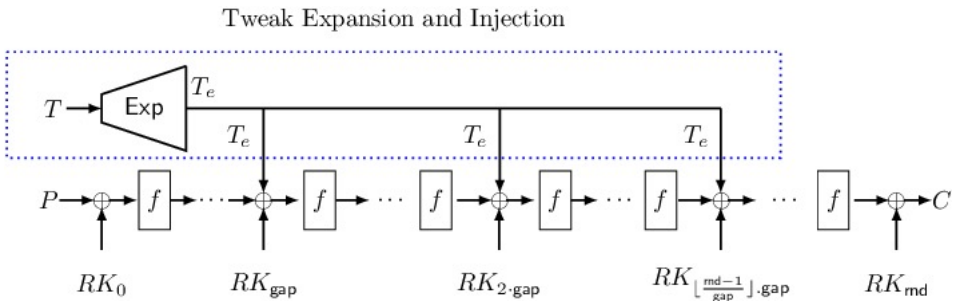
sESTATE -> 0 - 15 tweaks

SUNDAE uses only block cipher, why ESTATE requires tweakable block cipher !

Tweaks use for a domain separation

Tweaks #	Where	Why
0	To process the bulk of messages	Identical to block cipher
1	First block cipher call	Ensures RUP security
2 and 3	Full and partial AD block processing	For Domain separation
4 and 5	Full and partial final plaintext block processing	
6 and 7	Non-empty AD and empty message	
8	Empty AD and message	

The Elastic-Tweak Framework



- Use with any SPN based block ciphers that allows one to efficiently design tweakable block ciphers with short tweak.
- SPN – substitution permutation network (ex, AES)
- Convert block cipher (BC) to a short tweak tweakable block cipher (tBC)
- BC to tBC : BC [t, t_e, t_{ic}, gap]

  [tweak size, extended tweak size, positions where we insert the tweak bits, no of rounds between each tweak addition]
- How it works?
- Expand tweak with high distance error correcting code and then we enject it into state.

Tweaks In ESTATE

ESTATE has various constructions for a various application domain

TweAES-128: AES-128 [4, 8, 8, 2] → Energy Efficient

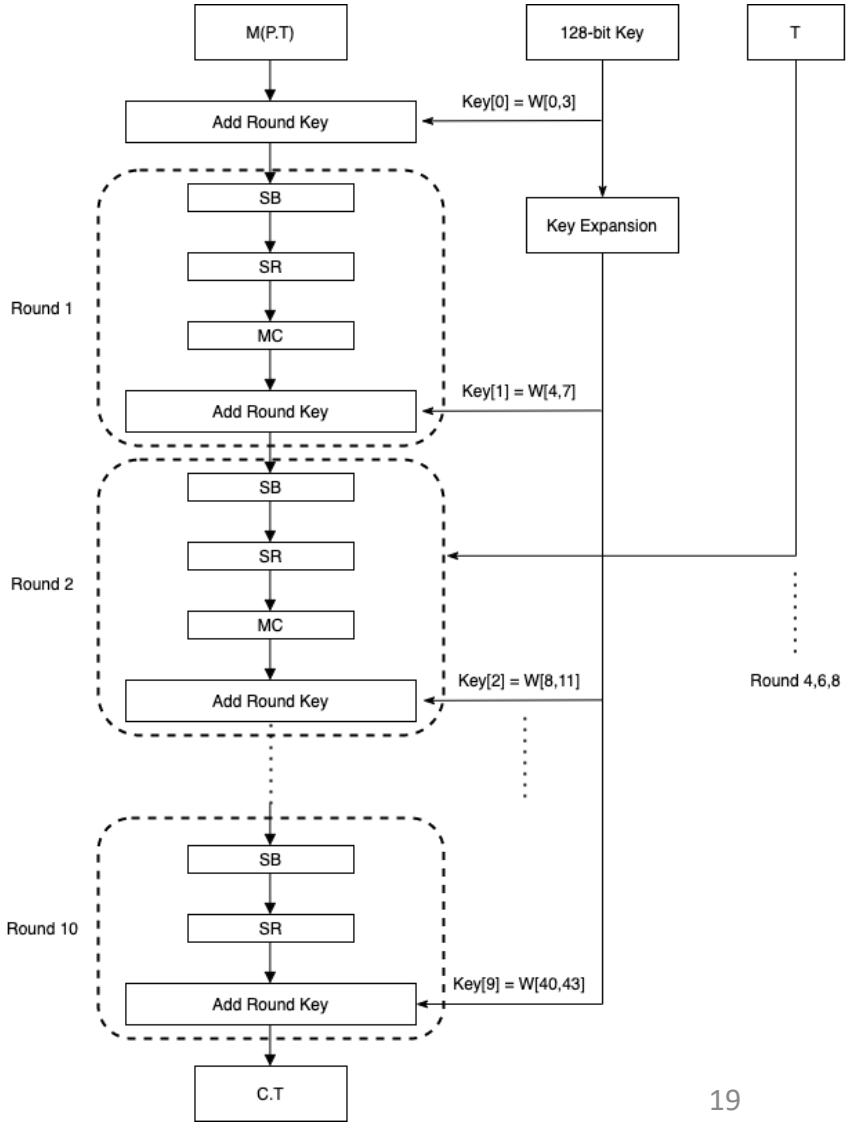
TweGIFT-128: GIFT-128 [4, 32, 32, 5] → Area Efficient

Tweakable Block Cipher (TweAES-128)

Key Size	128 bits
Tweak	4-bit
Block Size	128-bits
Number of Rounds	10

function TweAES(K, T , M)
(W43, . . . , W0) ← KeyGen(K)
X ← X ⊕ (W3, W2, W1, W0)
for i = 1 to 9 do
X ← SubBytes(X)
X ← ShiftRows(X)
X ← MixColumns(X)
X ← X ⊕ (W4i+3, W4i+2, W4i+1, W4i)
if i%2 = 0 then
X ← AddTweak(X, T)
X ← SubBytes(X)
X ← ShiftRows(X)
X ← X ⊕ (W43, W42, W41, W40)
return X

AddTweak: the 4-bit tweak first expanded to the 8-bit value with the help of linear code and in next step the 8-bit value is XORed to the state at an interval of 2 rounds (2,4,6,8).



Tweakable Block Cipher (TweAES-128-6)

Key Size	128 bits
Tweak	4-bit
Block Size	128-bits
Number of Rounds	6

Difference → Number of rounds
Last round (6th round) includes the mix column operation

```
Function TweAES-6(K,T,X)
    (W43,...,W0)←KeyGen(K,X)
    X←X⊕(W3,W2,W1,W0)
    for i=1 to 6 do
        X ← SubBytes(X)
        X ← ShiftRows(X)
        X ← MixColumns(X)
        X ← X ⊕ (W4i+3, W4i+2, W4i+1, W4i)
        if I % 2 = 0 and I < 6 then
            X ← AddTweak(X, T )
     return X
```

→ Here the AddTweak operation called in round 2 and 4 because the algorithm itself has an if condition for (i < 6) and that is why the 6th round is not performing AddTweak operation.

Tweakable Block Cipher (TweGIFT-128)

Key Size	128 bits
Tweak	4-bit
Block Size	128-bits
Number of Rounds	40

Each round of GIFT consists of 3 steps : which is conceptually similar to wrapping a gift

1. Put the content into a box (SubCells)
2. Wrap the ribbon around the box (PermBits)
3. Tie a knot to secure the content (AddRoundKey)

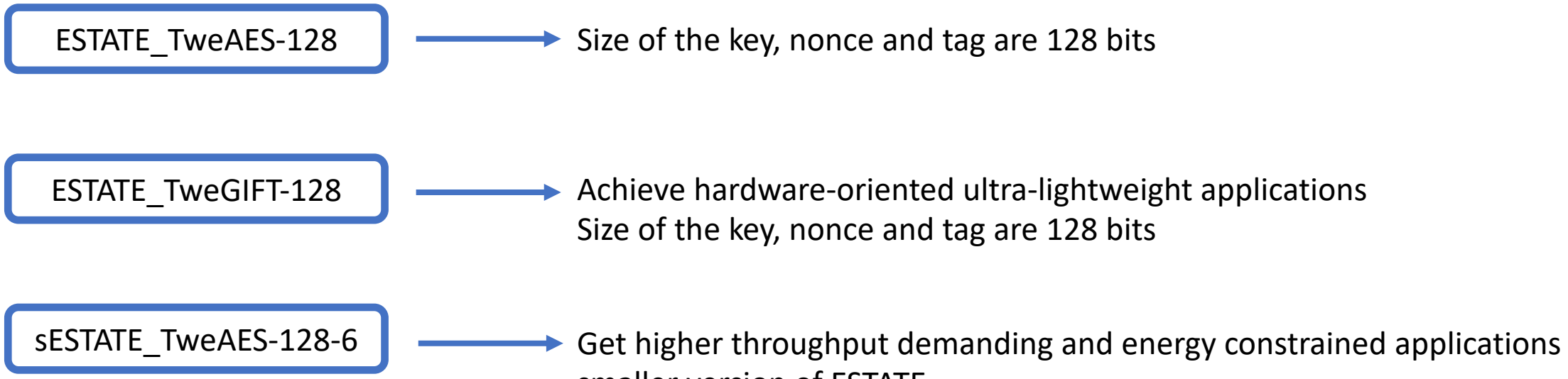
C -> The six bits are initialized to zero, and updated before being used in a given round
Each of the 6 bits is xored to a different nibble to break the symmetry

```
Function TweGIFT(K,T,X)
    C ← 000000
    for I = 0 to 39 do
            X ← SubCells(X)
            X ← PermBits(X)
            (K,X) ← AddRoundKey(K,X)
            (C,X) ← AddRoundConstant(C,X)
            if (i+1) % 5 = 0 and I < 39 then
                X ← AddTweak (X, T )
    return X
```

Here the 4-bit tweak is expanded to the 32-bit value with the help of linear code and the 32-bit value is XORed to the state at an interval of 5 rounds. Hence, the rounds 4,9,14,19,24,29 and 34 are affected.

ESTATE With Block Ciphers

AEAD scheme defined in ESTATE mode of operation with tweakable block cipher



Security

Approaches	Privacy		Integrity	
	Time	Data (in bytes)	Time	Data (in bytes)
ESTATE_TweAES-128	2^{128}	2^{64}	2^{128}	2^{64}
ESTATE_TweGIFT-128	2^{128}	2^{64}	2^{128}	2^{64}
sESTATE_TweAES-128	2^{112}	2^{60}	2^{112}	2^{60}

- All the algorithms are secure in nonce-misusing situation
- There are no hidden weaknesses in ESTATE and sESTATE modes of operations
- sESTATE_TweAES-128-6 allows a little bit less data and time as compared to ESTATE_TweAES-128 due to the round reduced function.
- ESTATE is very efficient for short messages

Obtained Features Of ESTATE And sESTATE

- Nonce-misuse Resistant
- Multiplication Free
- RUP Secure
- Robustness
- Optimal
- Single State

Hardware Security

Primitives	LUTs	FF	Slices	Frequency (MHz)	Clock cycles	Throughput (Mbps)
TweAES-128	1617	524	574	328.27	11	3819.87
ESTATE_TweAES-128	2235	679	1110	314.71	20	2014.14
TweGIFT-128	790	408	336	597.59	41	1865.65
ESTATE_TweGIFT-128	1413	698	616	580.11	80	928.27

- Used hardware: Virtex 7 FPGA
- LUTs stands for lookup tables that can be prefilled with a bunch of key value pairs that are utilized at some point in the future to quickly find values by key.
- With more LUTs, it is very hard to crack any system but at the same time it needs more hardware space to store the data structure.
- A greater number of cycles requires more time to perform and also gives a low throughput
- ESTATE_TweAES-128 is better in terms of performance

About Competition

Submission	Primitive	State size (bits)	Optimality	INT-RUP	Mult-free
ESTATE	tBC-128/128/4	260	✓	✓	✓
SUNDAE-GIFT	BC-128/128	256	×	×	×
Limdolen	BC-128/128	384	×	×	×
SIV-Rijndael256	tBC-256/128/4	388	✓	✓	✓
SIV-TEM-PHOTON	TBC-256/128/132	516	✓	✓	✓
TRIFLE	BC-128/128	384	×	×	×

SUNDAE and ESTATE continue in the second-round.

Comparison Between SUNDAE and ESTATE

	SUNDAE - AES	ESTATE -AES
Message length (bytes)	128	128
Cycles	181	171
Throughput (Mbps)	1713.13	1814.46

Throughput for short messages

Primitive calls:

ESTATE -> a + 2m

SUNDAE -> a + 2m + 1

- SUNDAE has one additional call to differentiate the emptiness of data
- ESTATE uses tweak to differentiate the data which is very efficient for short messages

Future Scope

- FCBC is not so desirable, one can use XCBC, where same key is used for all the block cipher.
- May be there is a chance of attack in MAC-then-encrypt method, alternative option is use of encrypt-then-MAC.
- Design of the algorithm: what if the user wants to verify the encrypted text before decrypting it? It is not possible with this scenario, firstly decrypting the cipher text and second, using FCBC to check the integrity of the message.

References

- N. D. A. J. C. M. L. M. N. Y. S. Avik Chakraborti, "ESTATE," p. 22, March 29, 2019.
- A. B. A. L. E. T. Subhadeep Banik, "SUNDAE: Small Universal Deterministic Authenticated Encryption for the Internet of Things," p. 35, 2018.
- P. W. H. H. Ping Zhang, "The INT-RUP Security of OCB with Intermediate (Parity) Checksum," p. 34.
- P. R. John Black, "CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions," p. 19, 2000.
- A. R. R. Mohan H.S, "REVISED AES AND ITS MODES OF OPERATION," International Journal of Information Technology and Knowledge Management, p. 8, June, 2012.
- M. Dworkin, "Recommendation for Block Cipher Modes of Operation," NIST, p. 65, 2001.
- "ADVANCED ENCRYPTION STANDARD (AES)," Federal Information Processing Standards Publication 197, p. 51, November, 2001.
- S. K. P. T. P. Y. S. S. M. S. Y. T. Subhadeep Banik, "GIFT: A Small Present Towards Reaching the Limit of Lightweight Encryption," p. 50, 2017.
- A. B. A. L. B. M. N. M. a. K. Y. Elena Andreeva, "How to securely release unverified plaintext in authenticated encryption," 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., p. 105–125, 2014.
- N. D. A. J. C. M. L. M. N. Y. S. Avik Chakraborti, "Elastic-Tweak: A Framework for Short Tweak Tweakable Block Cipher," p. 39.

Thank you