ZMAC: Specification, Security Proof, and Instantiation Updates*

Tetsu Iwata†

Nagoya University, Japan

Joint work with Kazuhiko Minematsu, Thomas Peyrin, and Yannick Seurin

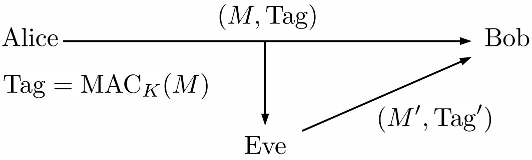
ASK 2017
Fenglin Hotel, Changsha, China
December 10, 2017

* Based on: Iwata, Minematsu, Peyrin, and Seurin. ZMAC: A Fast Tweakable Block Cipher Mode for Highly Secure Message Authentication. CRYPTO 2017

† Supported by JSPS KAKENHI, Grant-in-Aid for Scientific Research (B), Grant Number 26280045

Introduction: Message Authentication Code (MAC)

- Symmetric-key Crypto for tampering detection
- MAC : $\mathcal{K} \times \{0,1\}^* \to \mathcal{T}$
- Alice computes Tag $=$ MAC$(K, M) =$ MAC$_K(M)$ and sends $(M,$ Tag$)$ to Bob
- Bob checks if $(M,$ Tag$)$ is authentic by computing tag locally
- If MAC$_K(*)$ is a variable-input-length PRF, it is secure

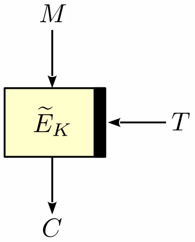


Tweakable Block Cipher (TBC)

Extension of ordinal Block Cipher (BC), formalized by Liskov et al. [LRW02]

- $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$, tweak $T \in \mathcal{T}$ is a public input
- $(K, T) \in \mathcal{K} \times \mathcal{T}$ specifies a permutation over $\mathcal{M}$
- Let $\mathcal{M} = \{0,1\}^n$ and $\mathcal{T} = \{0,1\}^t$

We implicitly assume additional small tweak $i = 1, 2, \dots,$ used for *domain separation*, and write as $\widetilde{E}^i_K(T, X)$ when necessary



Building TBC

Block cipher modes for TBC: LRW [LRW02] and XEX [Rog04]

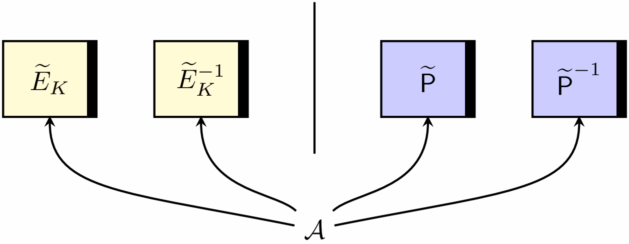
- Efficient but security is up to the birthday bound ($O(2^{64})$ attack when AES is used)
- Beyond-the-birthday-bound (BBB) security is possible (e.g. [Min09][LST12][LS15]) but not really efficient

Dedicated designs:

- HPC [Sch98]
- Threefish in Skein hash function [FLS+10]
- Deoxys-BC, Joltik-BC, KIASU-BC [JNP14a], SCREAM [GLS+14],
  - in the CAESAR submissions
- SKINNY [BJK+16], QARMA [Ava17], . . .

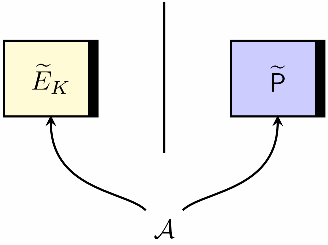
Security notions of TBC [LRW02]

- Indistinguishable from the set of independent uniform random permutations indexed by tweak
  - Tweakable uniform random permutation (TURP) denoted by $\widetilde{\mathsf{P}}$
  - Tweak is chosen by the adversary
- CCA-secure TBC = TSPRP



Security notions of TBC [LRW02]

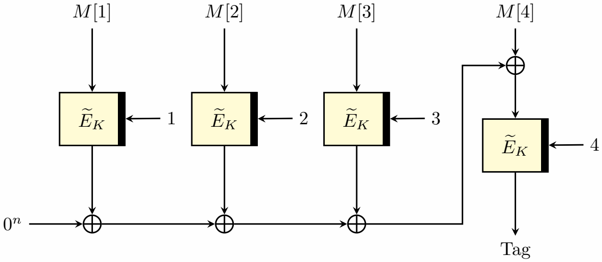
- Indistinguishable from the set of independent uniform random permutations indexed by tweak
  - Tweakable uniform random permutation (TURP) denoted by $\widetilde{\mathsf{P}}$
  - Tweak is chosen by the adversary
- CCA-secure TBC = TSPRP
- CPA-secure TBC = TPRP



Building MAC with TBC : PMAC1

PMAC1 by Rogaway [Rog04], introduced in the proof of PMAC

- Parallel
- Security is up to the birthday bound wrt the block size (n)
  - $\mathtt{Adv}^{\mathtt{tprp}}_{\mathsf{PMAC1}}(\sigma) = O(\sigma^2/2^n)$ for σ queried blocks
  - Thus $n/2$-bit security

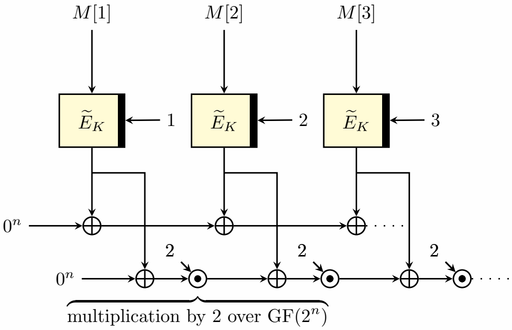


PMAC1

Building MAC with TBC: PMAC_TBC1k

PMAC_TBC1k by Naito [Nai15]

- $2n$-bit chaining similar to PMAC_Plus [Yas11]
  - Finalization by $2n$-bit PRF built from TBC
- BBB-secure: improve security of PMAC1 to n **bits**
- Same computation cost as PMAC1 (except for the finalization)

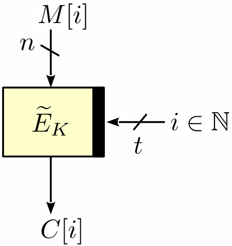


PMAC_TBC1k (message hashing part)

Efficiency of MAC

These TBC-based MACs are **not** optimally efficient

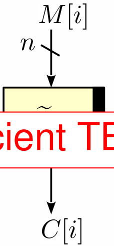
- They process n**-bit input per 1 TBC call**
- t-bit tweak does not process message – reserved for block index



Efficiency of MAC

These TBC-based MACs are **not** optimally efficient

- They process n**-bit input per 1 TBC call**
- t-bit tweak does not process message – reserved for block index



Optimally-efficient TBC-based MAC?

Our proposal: ZMAC ("The MAC") [IMPS17]

ZMAC is

- The first **optimally efficient** TBC-based MAC
  - $(n+t)$-bit input per 1 TBC call
- Parellel, and **BBB-secure**
  - $\min\{n, (n+t)/2\}$-bit security, e.g. n-bit-secure when $t \geq n$

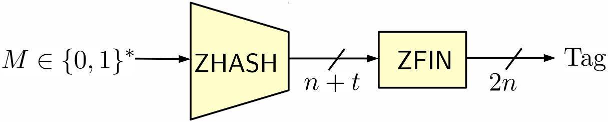
It uses TBC as a sole primitive, and secure if TBC is a TPRP

Structure of ZMAC

A simple composition of message hashing and finalization (Carter-Wegman MAC):

- $\mathsf{ZMAC} = \mathsf{ZFIN} \circ \mathsf{ZHASH}$
- $\mathsf{ZHASH} : \mathcal{M} \to \{0,1\}^{n+t}$ is a computational universal hash function
- $\mathsf{ZFIN} : \{0,1\}^{n+t} \to \{0,1\}^{2n}$ is a PRF
  - Output truncation if needed

Unified specs for any t ($t = n$ or $t < n$ or $t > n$)

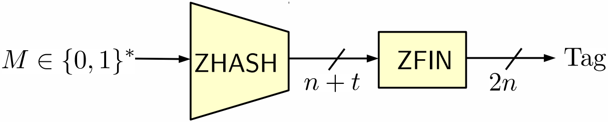


Structure of ZMAC

A simple composition of message hashing and finalization (Carter-Wegman MAC):

- ZMAC $=$ ZFIN $\circ$ ZHASH
- ZHASH : $\mathcal{M} \to \{0,1\}^{n+t}$ is a computational universal hash function
- ZFIN : $\{0,1\}^{n+t} \to \{0,1\}^{2n}$ is a PRF
  - Output truncation if needed

Unified specs for any t ($t = n$ or $t < n$ or $t > n$)

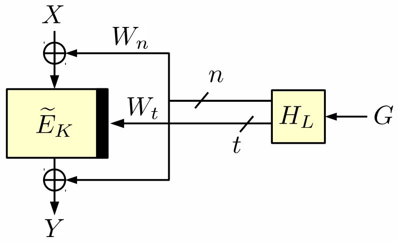


We focus on ZHASH

How ZHASH works: tweak extension

Optimal efficiency implies t-bit tweak of $\widetilde{E}$ must be extended to incorporate block index
This can be done by XTX [MI15], an extension of LRW and XEX:

- Global tweak $G \in \mathcal{G}$, $|\mathcal{G}| > 2^t$
- Keyed function $H : \mathcal{L} \times \mathcal{G} \to (\{0,1\}^n \times \{0,1\}^t)$
- XTX$[\widetilde{E}, H]_{K,L}(G, X) = \widetilde{E}_K(W_t, W_n \oplus X) \oplus W_n$ with $(W_n, W_t) = H_L(G)$

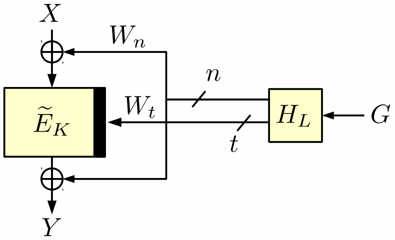


How ZHASH works: security of XTX/XT

XTX is secure if H is ϵ-partial AXU (pAXU) [MI15] :

$$\max_{G \neq G', \delta \in \{0,1\}^n} \Pr[L \xleftarrow{\$} \mathcal{L} \,:\, H_L(G) \oplus H_L(G') = (\delta, 0^t)] \leq \epsilon$$

that is, n-bit part is close to differentially uniform and t-bit part has a small collision probability

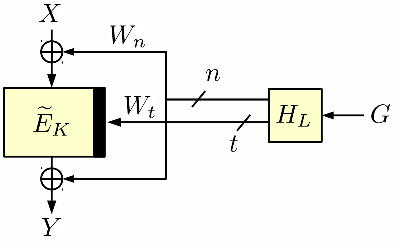


How ZHASH works: security of XTX/XT

In our case, $G \in \underbrace{\{0,1\}^t}_{\text{message part}} \times \underbrace{\mathbb{N}}_{\text{block index}}$ †, and block index is **a counter**

Then XTX can be instantiated and optimized by

- Using the "doubling" trick as XEX
- Omitting the outer mask to Y (as decryption is not needed)



† Omitting domain separation variable

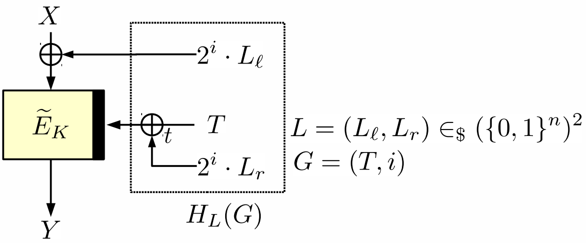
How ZHASH works: security of XTX/XT

The resulting scheme is **XT** , using $H_L(G)$ defined as

$$H_{(L_\ell,L_r)}(T,i) = (2^{i-1}L_\ell, 2^{i-1}L_r \oplus_t T), \text{ using two } n\text{-bit keys } (L_\ell, L_r)$$

Details:

- 2^iX is X multiplied by 2 over $\mathrm{GF}(2^n)$ for i times
  - Computation is easy by caching $2^{i-1}X$ as done in XEX
- $X \oplus_t Y = \mathtt{msb}_t(X) \oplus Y$ if $t \le n$, $(X \| 0^{t-n}) \oplus Y$ if $t > n$
  - Chop-or-pad before sum



How ZHASH works: security of XTX/XT

Lemma

Let $\widetilde{\mathsf{P}} : \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a TURP and H is ϵ-pAXU. Then,

$$\mathtt{Adv}^{\mathtt{tprp}}_{\mathsf{XT}[\widetilde{\mathsf{P}},H]}(q) \leq \frac{q^2\epsilon}{2}.$$

and our H is $1/2^{n+\min\{n,t\}}$-pAXU. Thus,

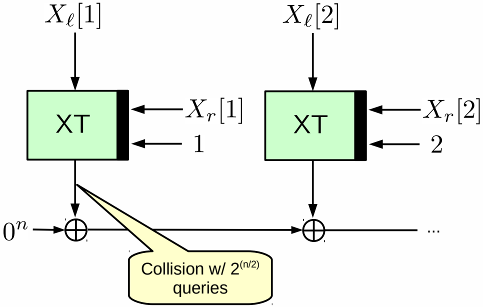
$$\mathtt{Adv}^{\mathtt{tprp}}_{\mathsf{XT}[\widetilde{\mathsf{P}},H]}(q) \leq \frac{q^2}{2^{n+\min\{n,t\}+1}}.$$

Therefore, **XT has $\min\{n,(n+t)/2\}$-bit, BBB-security**

How ZHASH works: chaining scheme

Given XT, it's easy to apply it in the PMAC-like single-chaining hashing scheme

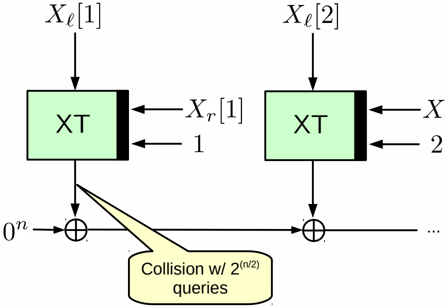
- Message is divided into $(n + t)$-bit blocks, $(X_\ell[i], X_r[i])$ for $i = 1, 2, \dots$
- This is optimally efficient, but security is up to the birthday bound



How ZHASH works: chaining scheme

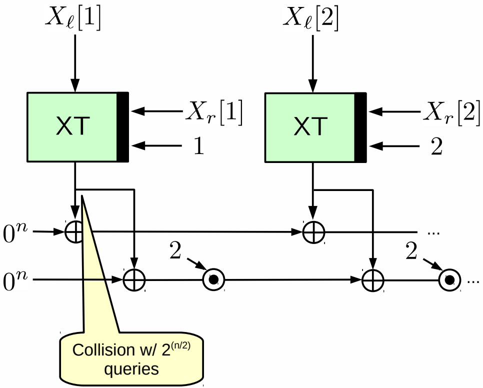
Given XT, it's easy to apply it in the PMAC-like single-chaining hashing scheme

- Message is divided into $(n+t)$-bit blocks, $(X_\ell[i], X_r[i])$ for $i = 1, 2, \ldots$
- This is optimally efficient, but security is up to the birthday bound
- Need a larger chaining value



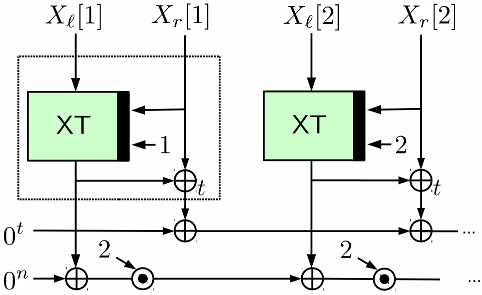
How ZHASH works: chaining scheme

- Naive use of $2n$-bit chaining scheme [Nai15][Yas11] doesn't work
  - XT output collision still breaks the scheme



How ZHASH works: chaining scheme

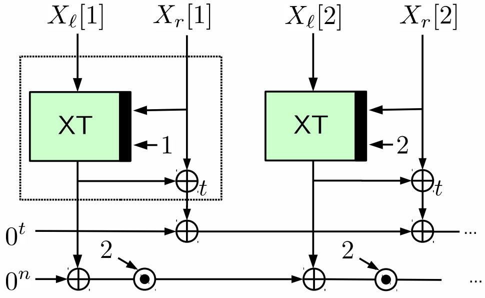
- Key observation: to avoid these collision attacks, the process of (X_ℓ, X_r) (the dotted box) **must be a permutation**
- A Feistel-like **1-round** permutation works ($\mathbb{ZHASH}$)



$\mathbb{ZHASH}$

How ZHASH works: chaining scheme

- Key observation: to avoid these collision attacks, the process of (X_ℓ, X_r) (the dotted box) **must be a permutation**
- A Feistel-like **1-round** permutation works ($\mathbb{ZHASH}$)



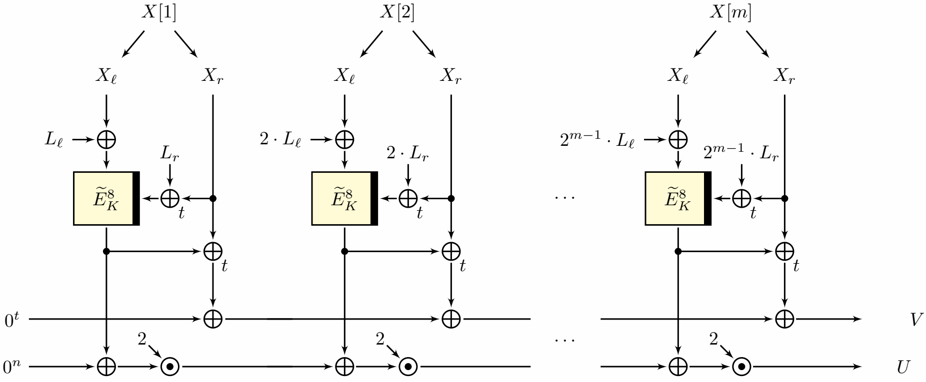
$\mathbb{ZHASH}$

Lemma

$\mathbb{ZHASH}$ (w/ XT using TURP) is ϵ-almost universal for $\epsilon = 4/2^{n+\min\{n,t\}}$

Full ZHASH

Input: $X = (X[1], \ldots, X[m])$, $|X[i]| = n + t$
Output (U, V), $|U| = n$, $|V| = t$

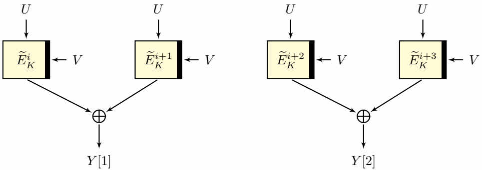


Details:

- $X \oplus_t Y = \mathtt{msb}_t(X) \oplus Y$ if $t \leq n$, $(X \,\|\, 0^{t-n}) \oplus Y$ if $t > n$
- $2 \cdot X$: multiplication by 2
- L_ℓ and L_r : two n-bit masks from $\widetilde{E}_K$ w/ domain separation

ZFIN

ZFIN simply encrypts U with tweak V twice (for each n-bit output) and takes a sum (with domain separation)



PRF security of ZFIN

- ZFIN is essentially "Sum of Permutations" [Luc00, BI99, Pat08a, Pat13, CLP14, MN17]
- From a recent result by Dai et al. [DHT17], ZFIN is **n-bit secure**

Lemma

$$\mathtt{Adv}^{\mathtt{prf}}_{\mathsf{ZFIN}[\widetilde{\mathsf{P}}]}(q) \leq 2\left(\frac{q}{2^n}\right)^{3/2}$$

Security of ZMAC

Combining all lemmas,

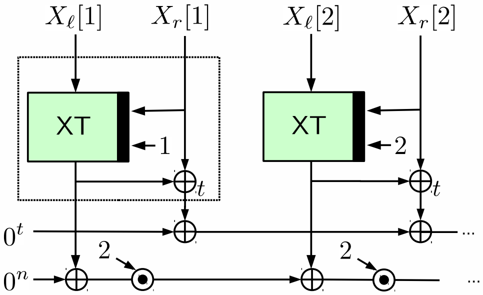
Theorem

For $q \le 2^{n-4}$ queries of total σ $(n+t)$-bit blocks,

$$\mathtt{Adv}^{\mathtt{prf}}_{\mathsf{ZMAC}[\widetilde{\mathsf{P}}]}(q,\sigma) \le \frac{2.5\sigma^2}{2^{n+\min\{n,t\}}} + 4\left(\frac{q}{2^n}\right)^{3/2}.$$

Thus ZMAC is $\min\{n,(n+t)/2\}$-bit secure

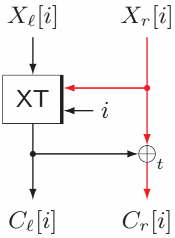
Security Proof



$\mathbb{ZHASH}$

- $\mathbb{ZHASH}$ is ϵ-almost universal for $\epsilon = 4/2^{n+\min\{n,t\}}$
- $\max_{\substack{X \in (\{0,1\}^{n+t})^m \\ X' \in (\{0,1\}^{n+1})^{m'} \\ X \neq X'}} \Pr_{\mathsf{XT}}[\mathbb{ZHASH}_{\mathsf{XT}}(X) = \mathbb{ZHASH}_{\mathsf{XT}}(X')] \leq \epsilon$

A Feistel-like Network Is a Permutation



- red lines are t bits
- $X \oplus_t Y = \texttt{msb}_t(X) \oplus Y$ if $t \leq n$, $(X \,\|\, 0^{t-n}) \oplus Y$ if $t > n$

Breaking into Cases

- $\mathbb{ZHASH}$ is ϵ-almost universal for $\epsilon = 4/2^{n+\min\{n,t\}}$
- For any distinct $X \in (\{0,1\}^{n+t})^m$ and $X' \in (\{0,1\}^{n+1})^{m'}$,

$$\Pr_{\mathsf{XT}}[\mathbb{ZHASH}_{\mathsf{XT}}(X) = \mathbb{ZHASH}_{\mathsf{XT}}(X')] \leq \epsilon$$

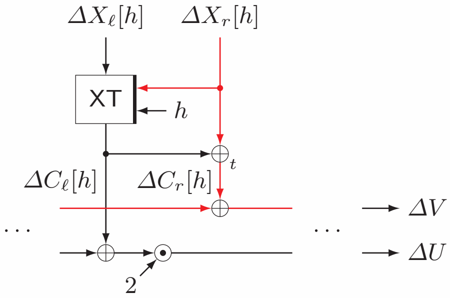
Cases:

1. $m = m'$, $\exists h, X[h] \neq X'[h]$, and $\forall i \neq h, X[i] = X'[i]$
(same number of blocks, difference in exactly one block)
2. $m = m'$, $\exists h, s, X[h] \neq X'[h]$ and $X[s] \neq X'[s]$
(same number of blocks, difference in two (or more) blocks)
3. $m' = m + 1$
4. $m' \geq m + 2$

- focus on the case $t \leq n$

Case 1

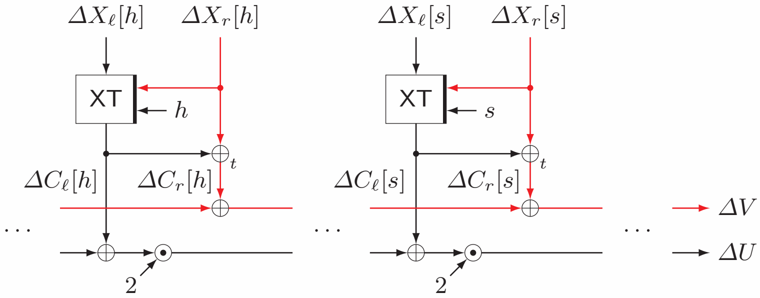
- $m = m'$, $\exists h, X[h] \neq X'[h]$, and $\forall i \neq h, X[i] = X'[i]$
- same number of blocks, difference in exactly one block



- $(\Delta C_\ell[h], \Delta C_r[h]) \neq (0^n, 0^t)$, so $(\Delta U, \Delta V) \neq (0^n, 0^t)$
- $\Pr_{\mathsf{XT}}[\mathbb{ZHASH}_{\mathsf{XT}}(X) = \mathbb{ZHASH}_{\mathsf{XT}}(X')] = 0$

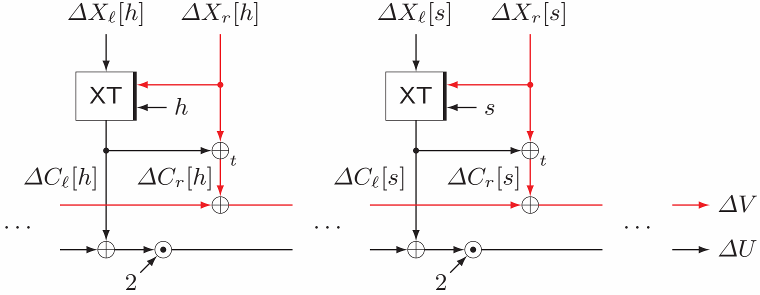
Case 2

- $m = m'$, $\exists h, s, X[h] \neq X'[h]$ and $X[s] \neq X'[s]$
- same number of blocks, difference in two (or more) blocks



- $(\Delta C_\ell[h], \Delta C_r[h]) \neq (0^n, 0^t)$ and $(\Delta C_\ell[s], \Delta C_r[s]) \neq (0^n, 0^t)$
- approach: use $\Delta C_\ell[h]$ and $\Delta C_\ell[s]$ as randomness

Case 2



- $\Delta U = 0^t \;\Leftrightarrow\; 2^{m-h-1}\Delta C_\ell[h] \oplus 2^{m-s-1}\Delta C_\ell[s] = \Delta_1$
- $\Delta V = 0^n \;\Leftrightarrow\; \Delta C_r[h] \oplus \Delta C_r[s] = \Delta_2$

  $\Leftrightarrow\; \mathtt{msb}_t(\Delta C_\ell[h] \oplus \Delta C_\ell[s]) = \Delta_2'$

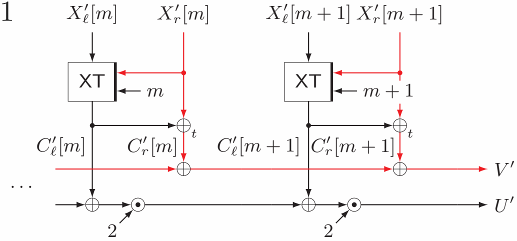
  $\Leftrightarrow\; \Delta C_\ell[h] \oplus \Delta C_\ell[s] = \Delta_2' \,\|\, *$

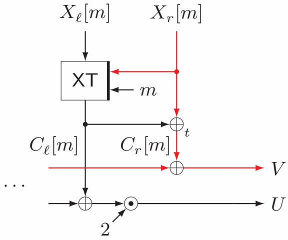
Case 2

- $\begin{cases} \Delta U = 0^t \\ \Delta V = 0^n \end{cases} \Leftrightarrow \begin{cases} 2^{m-h-1}\Delta C_\ell[h] \oplus 2^{m-s-1}\Delta C_\ell[s] = \Delta_1 \\ \Delta C_\ell[h] \oplus \Delta C_\ell[s] = \Delta_2' \parallel * \end{cases}$
- For each $(\Delta_2, \Delta_2' \parallel *)$, one possibility for $(\Delta C_r[h], \Delta C_r[s])$
  - at most 2^{n-t} possible values of $(\Delta C_r[h], \Delta C_r[s])$ s.t. $(\Delta U, \Delta V) = (0^n, 0^t)$
- at least $(2^n - 1)^2$ possible choices for $(\Delta C_r[h], \Delta C_r[s])$
- $\Pr[(\Delta U, \Delta V) = (0^n, 0^t)] \leq \dfrac{2^{n-t}}{(2^n-1)^2} \leq \dfrac{4}{2^{n+t}}$

Case 3

- $m' = m + 1$





- use $C_\ell[m], C'_\ell[m], C'_\ell[m+1]$ as randomness

- $\Delta U = 2(C_\ell[m] \oplus 2C'_\ell[m] \oplus C'_\ell[m+1] \oplus \Delta_1)$
- $\Delta V = \mathtt{msb}_t(C_\ell[m] \oplus C'_\ell[m] \oplus C'_\ell[m+1]) \oplus \Delta_2$

Case 3

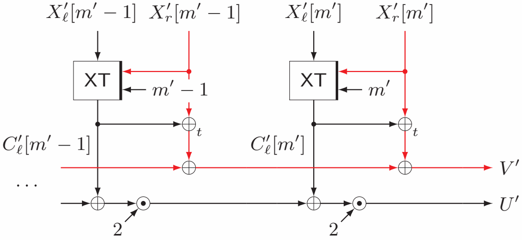
- $\varDelta U = 2(C_\ell[m] \oplus 2C'_\ell[m] \oplus C'_\ell[m+1] \oplus \varDelta_1)$
- $\varDelta V = \mathtt{msb}_t(C_\ell[m] \oplus C'_\ell[m] \oplus C'_\ell[m+1]) \oplus \varDelta_2$
- $\begin{cases} \varDelta U = 0^t \\ \varDelta V = 0^n \end{cases} \Leftrightarrow \begin{cases} C_\ell[m] \oplus 2C'_\ell[m] \oplus C'_\ell[m+1] = \varDelta'_1 \\ C_\ell[m] \oplus C'_\ell[m] \oplus C'_\ell[m+1] = \varDelta_2 \parallel * \end{cases}$
- Letting $Y = C_\ell[m] \oplus C'_\ell[m+1]$ and $Z = C'_\ell[m]$ yields

$$\begin{cases} Y \oplus 2Z = \varDelta'_1 \\ Y \oplus Z = \varDelta_2 \parallel * \end{cases}$$

  which has a unique solution
- they are uniform over $\{0,1\}^n$
- $\Pr[(\varDelta U, \varDelta V) = (0^n, 0^t)] \leq \frac{2^{n-t}}{2^{2n}} \leq \frac{1}{2^{n+t}}$

Case 4

- $m' \geq m + 2$



- use $C'_\ell[m'-1]$ and $C'_\ell[m']$ as randomness
- $\Delta U = 2(2C'_\ell[m'-1] \oplus C'_\ell[m'] \oplus \Delta_1)$
- $\Delta V = \mathtt{msb}_t(C'_\ell[m'-1] \oplus C'_\ell[m']) \oplus \Delta_2$
- the same analysis as Case 3 can be used
- $\Pr[(\Delta U, \Delta V) = (0^n, 0^t)] \leq \dfrac{1}{2^{n+t}}$
- $\Pr[(\Delta U, \Delta V) = (0^n, 0^t)] \leq \dfrac{4}{2^{n+t}}$ for all cases

Instantiation Updates*

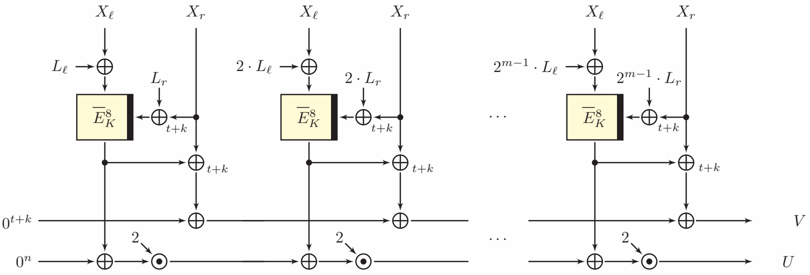
- In [IMPS17], we used Deoxys-BC and SKINNY to instantiate ZMAC
  - standard TPRP security assumption
- "XOR some extra tweak material to the key input of the TBC"
  - originally proposed by [LRW02] for BCs
- Given $\widetilde{E}^i : \{0,1\}^k \times \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$, regard it as

$$\overline{E}^i : \{0,1\}^k \times \{0,1\}^{t+k} \times \{0,1\}^n \to \{0,1\}^n$$

∗ Thanks to Christof Beierle for the suggestion.

Instantiation Updates

- Input: $X = (X[1], \ldots, X[m])$,
  $|X[i]| = n + (t + k), X[i] = (X_\ell[i], X_r[i])$: $X_r[i]$ is $t + k$ bits
- Output (U, V), $|U| = n$, $|V| = t + k$



- can process $(n + t + k)$ bits per 1 TBC call

Remarks

- related-key security of $\widetilde{E}$ is needed (strong assumption)
- limited to the birthday security w.r.t. k
  - due to a generic birthday attack against $E_{K \oplus T}(\cdot)$ by [BK03]
  - $E_{K_i}(X)$ for $1 \leq i \leq 2^{k/2}$ and $E_{K \oplus T_j}(X)$ for $1 \leq j \leq 2^{k/2}$
- with Deoxys-BC-256, $k = 128, t = 124, n = 128$ (4 bits for domain separation)
  - 64-bit security, expected to be 50% faster
  - related-key security will not be an issue (also for SKINNY)

Instantiation with AES-128

- Can use ZMAC with AES-128
  - 64-bit security
  - estimated speed: 0.45 cpb (taking into account the 1.4 slowdown for recomputation of the key schedule at every block
  - AES-256 is not suitable because of the related-key attack [BKN09] schedule)

Concluding remarks

- Reviewed ZMAC, a highly secure and fast MAC based on TBC
- Security Proof
- Instantiation updates

The power of XEX-like masking:

- We already see it in many blockcipher modes (e.g. PMAC, OCB)
- ZMAC shows it is also powerful for TBC modes
- As dedicated TBCs are becoming popular, this direction looks worth to be further explored

Concluding remarks

- Reviewed ZMAC, a highly secure and fast MAC based on TBC
- Security Proof
- Instantiation updates

The power of XEX-like masking:

- We already see it in many blockcipher modes (e.g. PMAC, OCB)
- ZMAC shows it is also powerful for TBC modes
- As dedicated TBCs are becoming popular, this direction looks worth to be further explored

Thank you!