Detection of Data Corruption via Combinatorial Group Testing and beyond

Kazuhiko Minematsu*

NEC

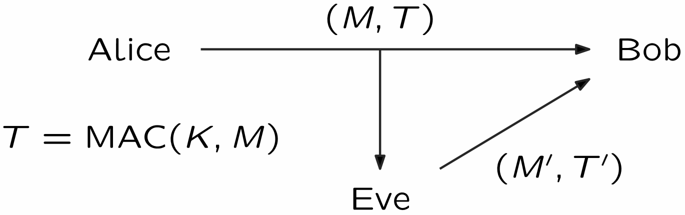
The 9th Asian-workshop on Symmetric Key Cryptography (ASK 2019)
December 14, 2019 Kobe, Japan

* Joint Work with Norifumi Kamiya

Introduction

Message Authentication Code (MAC)

- Symmetric-key Crypto for tampering detection
- Alice computes tag $T = \text{MAC}(K, M)$ for message M
- Bob verifies (M, T) by checking tag

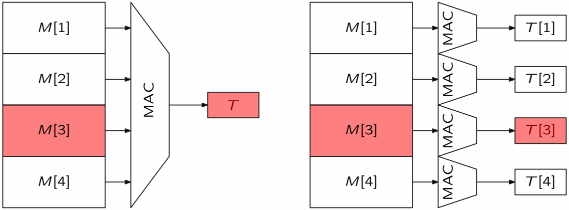


Limitation on Conventional MACs

When message M consists of m *items* (e.g. HDD sectors)

Say $d < m$ items were corrupted. How to detect them ?

- Important feature w/ many potential applications
  - Storage integrity, IoT, digital forensics etc.
- Trivial solutions have limitations :
  - One tag for all items : impossible
  - Tag for each item : possible but not scalable (m tags)

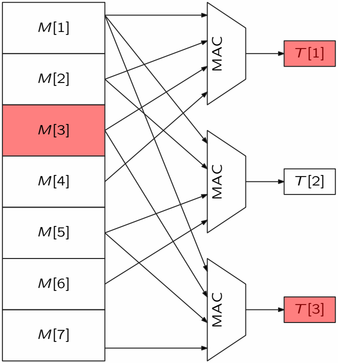


Can we reduce tags w/o losing the detection capability ?

Possible Direction : Overlapping MAC Inputs

Ex. $m = 7$ items, $t = 3$ tags

the scheme determined by 3×7 *test* matrix $\mathbf{H}$



$$\mathbf{H} = \begin{pmatrix} 1 & 1 & \color{red}{1} & 1 & 0 & 0 & 0 \\ 1 & 1 & \color{red}{0} & 0 & 1 & 1 & 0 \\ 1 & 0 & \color{red}{1} & 0 & 1 & 0 & 1 \end{pmatrix}$$

Possible Direction : Overlapping MAC Inputs

Suppose at most $d = 1$ item was corrupted.
The response (verification result) is 3 bits :

Response	000	001	010	011	100	101	110	111
Corrupted item	none	7	6	5	4	3	2	1

- One-to-one between the response and the pattern of corruption
- $\rightarrow$ the corrupted item can be identified

We call this **Corruption Detectable MAC**

Combinatorial Group Testing (CGT) and CDMAC

CDMAC is an application of combinatorial group testing (CGT)

- CGT : a method to find *defectives* using **group test** ("does group G contain any defective ?") [DH00]
  - invented during WWII by Durfman, as a method to find syphilis from blood samples
  - applications to biology and information science

For CDMAC :

- Group test = verification of a tag
- Defective = corrupted item

[DH00] Du and Hwang. Combinatorial Group Testing and Its Applications. World Scientific 2000

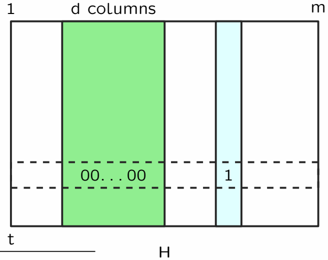
Disjunct Matrix

How to make the test matrix H?

- if $\mathbf{H}$ is ***d*-disjunct**, we can detect $\leq d$ corrupted items
- d-disjuct : “any union of $\leq d$ columns does not contain any other column”

Natural goal : use H of minimum rows (t) given (m, d)

- Lower bound : $t = \Theta(d^2 \log_2 m)$
- Most known constructions are sub-optimal
- Order-optimal construction exists [PR11]
- Constant-optimal : even the case $d = 2$ remains open for decades



[PR11] Porat and Rothchild. Explicit Nonadaptive Combinatorial Group Testing Schemes, IEEE IT 2011

Previous Work on CDMAC/CDHash

The view is not new :

- MAC for data forensics by Goodrich et al. [GAT05]
- Corruption-localizing MAC/hash function by Crescenzo et al. [CV06,CJS09]
- Use d-disjunct matrix to MAC/Hash function in a black-box way

Possible Applications

- (Cloud) Storage Integrity for (e.g.) forensics or proof-of-retrievablity
- Approximate/Robust authentication (e.g. biometrics or image)
- Low-bandwidth comminication such as IoT

[GAT05] Goodrich, Atallah and Tammasia. Indexing Information for Data Forensics. ACNS 2005

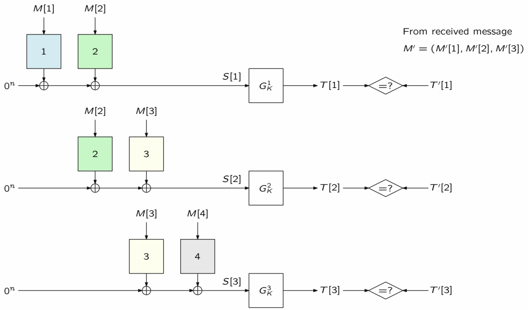
[CV06] Crescenzo and Vakil. Cryptographic hashing for virus localization. WORM 2006

[CJS09] Crescenzo, Jiang and Safavi-Naini. Corruption-Localizing Hashing. ESORICS 2009

Group-Test MAC [Min15]

First focus on the computational aspects of CD MAC:

- Naive tag computation : $O(w)$ time for $\mathbf{H}$ of weight w (worst case $O(mt)$)
- Showed that a XOR-MAC/PMAC-like structure allows $O(m+t)$ computation
- Provable security analysis for several relevant notions



[Min15] Minematsu. Efficient Message Authentication Codes with Combinatorial Group Testing. ESORICS 2015.

What [Min15] did and didn't

- The computation of CDMAC can be close to single (XOR-)MAC
- What about the communication ?
- The barrier of $O(d^2 \log m)$: no non-trivial CDMAC for $d = O(\sqrt{m/\log m})$ including [Min15]

New Approach to CDMAC [MK19]

XOR-GTM : a novel approach to CDMAC

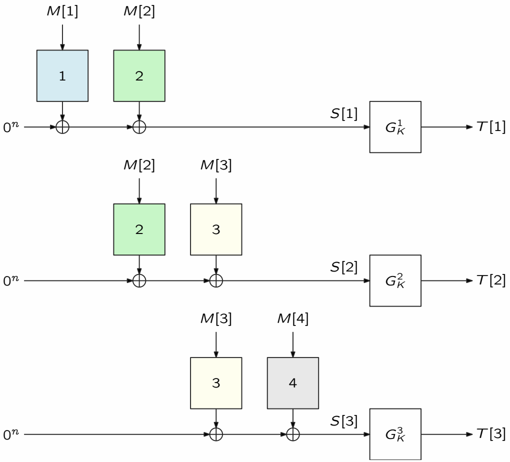
- Exploits the linearity of (intermediate) tags
- **Allows to break $O(d^2 \log m)$ communication barrier**
- Several concrete instantiations
  - Significantly smaller # of tags than **any** of known CDMAC
- Provable security based on standard primitives

[MK19] Minematsu and Kamiya. Symmetric-key Corruption Detection : When XOR-MACs meet Combinatorial Group Testing, ESORICS 2019

Baseline : GTM [Min15] for $(m = 4, t = 3)$

(caveat : this ex is not secure as a standard det MAC)

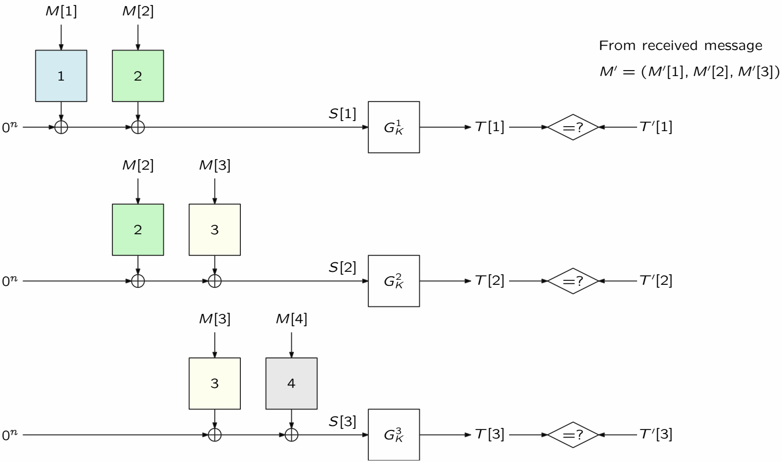
- Tagging : take 3 tags for $(M[1], M[2])$, $(M[2], M[3])$, $(M[3], M[4])$



Baseline : GTM [Min15] for $(m = 4, t = 3)$

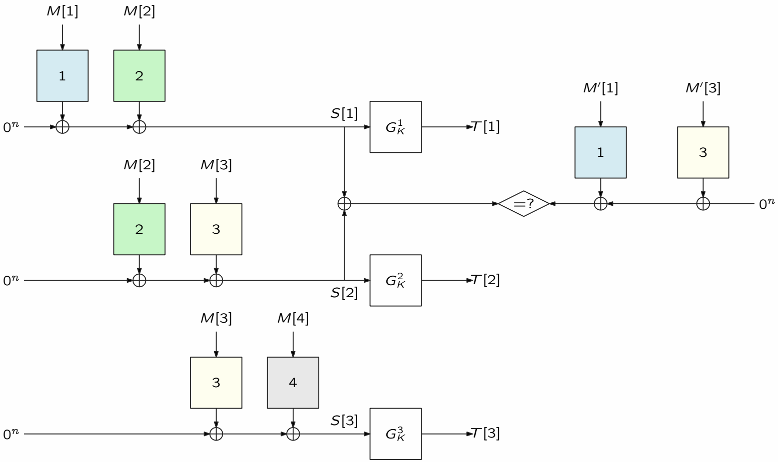
(caveat : this ex is not secure as a standard det MAC)

- Tagging : take 3 tags for $(M[1], M[2])$, $(M[2], M[3])$, $(M[3], M[4])$
- Verification : Check the matches of tags, and *decode*



Key Observation : Linearity of S

- Eg. $S[1] \oplus S[2]$ works for checking $(M[1], M[3])$
- New checkable subset **w/o increasing tags**
- $S[i]$ obtained by decrypting $T[i]$



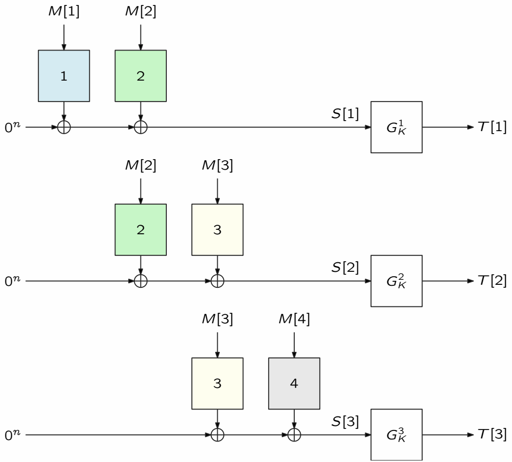
XOR-GTM : Parameters

- $(t \times m)$ test matrix $\mathbf{H}$
- Expansion rule R : a subset of $2^{\{1,\ldots,m\}}$ ($|R| = v$)
- Extended test matrix $\mathbf{H}^R$: $v \times m$ submatrix of span$(\mathbf{H})$ following R
  - This case : $(m = 7, t = 3, v = 6)$
  - $R = ((1), (2), (3), (1, 2), (2, 3), (1, 2, 3))$

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \quad \mathbf{H}^R = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

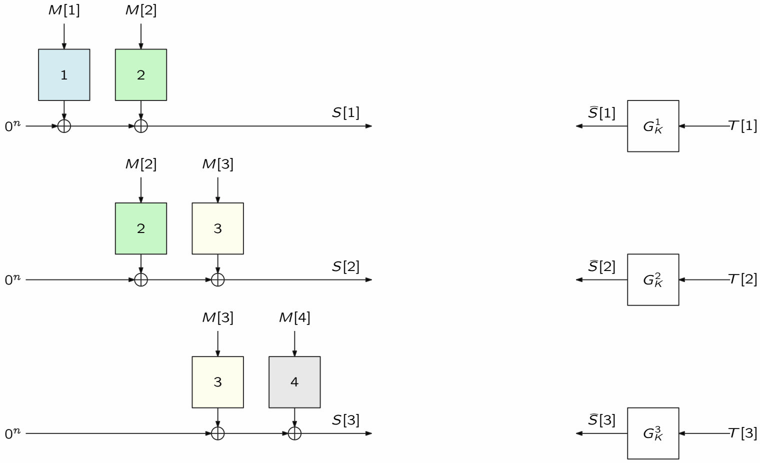
XOR-GTM : Tagging

The same as Min15 : compute $T = (T[1], T[2], T[3])$ **following** $\mathbf{H}$



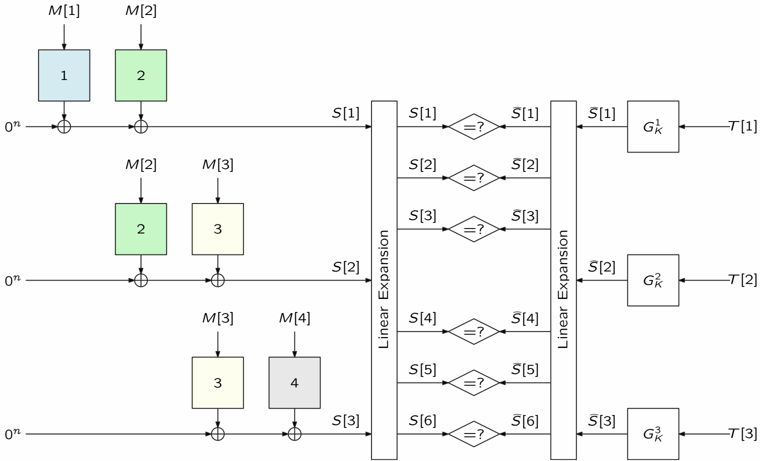
XOR-GTM : Verification Step 1

1. Decrypt T to recover intermediate tags $\widehat{S} = (\widehat{S}[1], \widehat{S}[2], \widehat{S}[3])$
2. Compute $S = (S[1], S[2], S[3])$ from the received message



XOR-GTM : Verification Step 2

1. Apply a linear expansion to $\widehat{S}$ and S by $\mathbf{H}^R$
2. Check the match $\widehat{S}[i] = S[i]$ for all i,
3. and remove all items those included in passed tests (**naive decoding**)
4. Remaining items are identified as corrupted



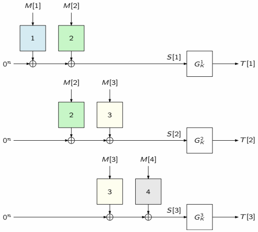
Properties of XOR-GTM

Security of Corruption Detection

- If $\mathbf{H}^R$ is d-disjunct, $\leq d$ corruptions can be found
- Security proved in a similar way as Min15 (eg decoder unforgeability)
  - Assuming PRF and TPRP
  - For standard MAC security $\mathbf{H}^R$ must include all-one row

Computational Efficiency : the same as Min15

- $m\ F_K$ calls + $t\ G_{K'}$ calls irrespective of $\mathbf{H}$
- Typically $m \gg t$, thus **almost efficient as single (XOR-)MAC**



Instantiations of XOR-GTM

To instantiate XOR-GTM

- $\mathbf{H}^R$ should be d-disjunct
- **Rank** (over GF(2^n)) for $\mathbf{H}^R$ determines the communication cost (i.e. the lows of $\mathbf{H}$)
  - $\mathbf{H}$ is a basis matrix of $\mathbf{H}^R$
- Thus what needed is **d-disjunct matrix of low rank**
- No easy :
  - Rank of test matrix was rarely studied in the field of CGT
  - Known small-row d-disjunct matrices tend to be high-rank (to our experiments)

Instantiations of XOR-GTM (Contd.)

What we found instead :

- (Near-)square matrices of large d, small rank
- ... almost useless in the context of CGT !
- studied in coding $\&$ design theory

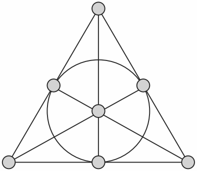
Three examples in the (full) paper of [MK19]:

- Macula
- Hadamard for large m and fixed $d = 2$
- **Finite Geometry-based** : large m and d

d-disjunct Matrices from Finite Geometry

- $\mathbf{P}^{(s)}$: $m \times m$ binary matrix, $m = 2^{2s} + 2^s + 1$ for integer $s > 0$
- Projective-plane incidence (PPI) matrix over GF(2^s)
  - (i, j) element $= 1$ iff i-th point is on j-th line

Example: $s = 1$ **(7 lines and 7 points)**



$$\mathbf{P}^{(1)} = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Properties of $\mathbf{P}^{(s)}$

From (classical) coding theory / design theory, $\mathbf{P}^{(s)}$ is

- 2^s-disjunct
- Rank $3^s + 1$

Significant advantage over any DirectGTM (conventional CDMAC)

- $t \approx 3^s$ tags to detect $d = 2^s$ corruptions (note $m = O(2^{2s})$)
- That is, $t = d^{\log 3} + 1 \approx d^{1.58}$
  - DirectGTM needs $O(d^2 \log m)$ tags
- Sparse parameter choice : mitigated by a class of Affine-plane matrices by Kamiya [Kam07] (designed for LDPC codes)

[Kam07] Kamiya. High-Rate Quasi-Cyclic Low-Density Parity-Check Codes Derived From Finite Affine Planes. IEEE IT 2007

Numerical Examples for Storage Applications

Ex. 128-bit tag for each 4K-byte sector of storage devices

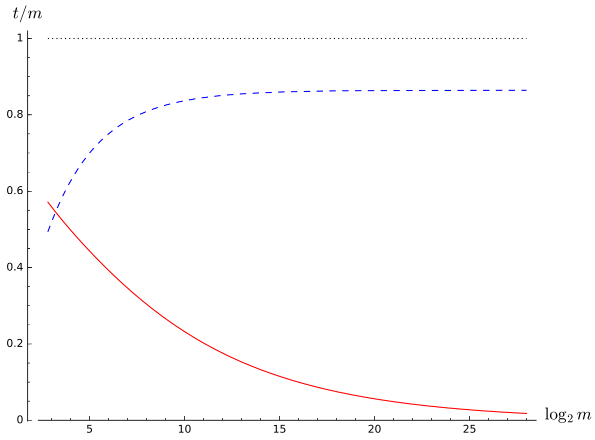
- XOR-GTM with PPI matrix reduces tags by a factor of **18$\sim$ 75**

Target: 4.4 TB HDD	Total tag size	Corrupted data	Imp. Factor
Trivial scheme	17.18 GB	No limit	1
(ideal) DirectGTM	14.85 GB	135 MB	1.15
XOR-GTM-PPI ($s = 15$)	229.58 MB	135 MB	74.82
Target: 1.1 TB HDD	Total tag size	Corrupted data	Imp. Factor
Trivial scheme	4.29 GB	No limit	1
(ideal) DirectGTM	3.71 GB	68 MB	1.15
XOR-GTM-PPI ($s = 14$)	76.52 MB	68 MB	56.06
Target: 4.3 GB Memory	Total tag size	Corrupted data	Imp. Factor
Trivial scheme	16.79 MB	No limit	1
(ideal) DirectGTM	14.50 MB	5 MB	1.15
XOR-GTM-PPI ($s = 10$)	0.94 MB	5 MB	17.86

Also performed experimental implementation up to $s = 5$ (see paper)

Communication Ratios (t/m)

- (Blue) : DirectGTM with a known lower bound of d-disjunct matrix [SG16]
- (Black) : DirectGTM with a conjectured lower bound [EFF85]
- (Red) : XOR-GTM-PPI



Concluding Remarks

- A new approach to corruption detection via MAC
- Significant improvement from the known schemes
  - Breaks the theoretical limit in communication
- Many future/ongoing directions
  - Implementation using PPI matrix of large s
  - Application to aggregate MAC [KL06], hash or digital signature, error-tolerant variant...

Concluding Remarks

- A new approach to corruption detection via MAC
- Significant improvement from the known schemes
  - Breaks the theoretical limit in communication
- Many future/ongoing directions
  - Implementation using PPI matrix of large s
  - Application to aggregate MAC [KL06], hash or digital signature, error-tolerant variant...

Thanks!

(Backup) Experimental Implementation

XOR-GTM-PPI on Linux (Ubuntu 16.04, Xeon E5 2.2 GHz):

- Using PMAC-AES for F^i_K and XEX-AES for $G^i_{K'}$ w/ AES-NI
- Utilized the matrix structure (**circulant**)
- As message items get long, the speed approaches the speed of PMAC itself (5.2 cpb for long inputs)

Size of each	$s = 1$		$s = 2$		$s = 3$		$s = 4$		$s = 5$	
message item	tag	verf	tag	verf	tag	verf	tag	verf	tag	verf
1 KB	14.6	20.8	16.6	20.7	14.8	22.5	20.67	23.5	15.4	15.5
2 KB	14.5	18.2	14.5	18.2	10.8	17.6	15.0	15.1	16.8	16.9
4 KB	13.5	16.9	10.1	16.9	12.9	14.0	6.3	10.5	12.6	12.7
1 MB	5.2	8.5	5.2	5.2	5.2	5.2	5.2	5.2	5.2	5.2

(cycles / input byte)

Now improved, the speed close to native PMAC (0.8 cpb) for 1MB