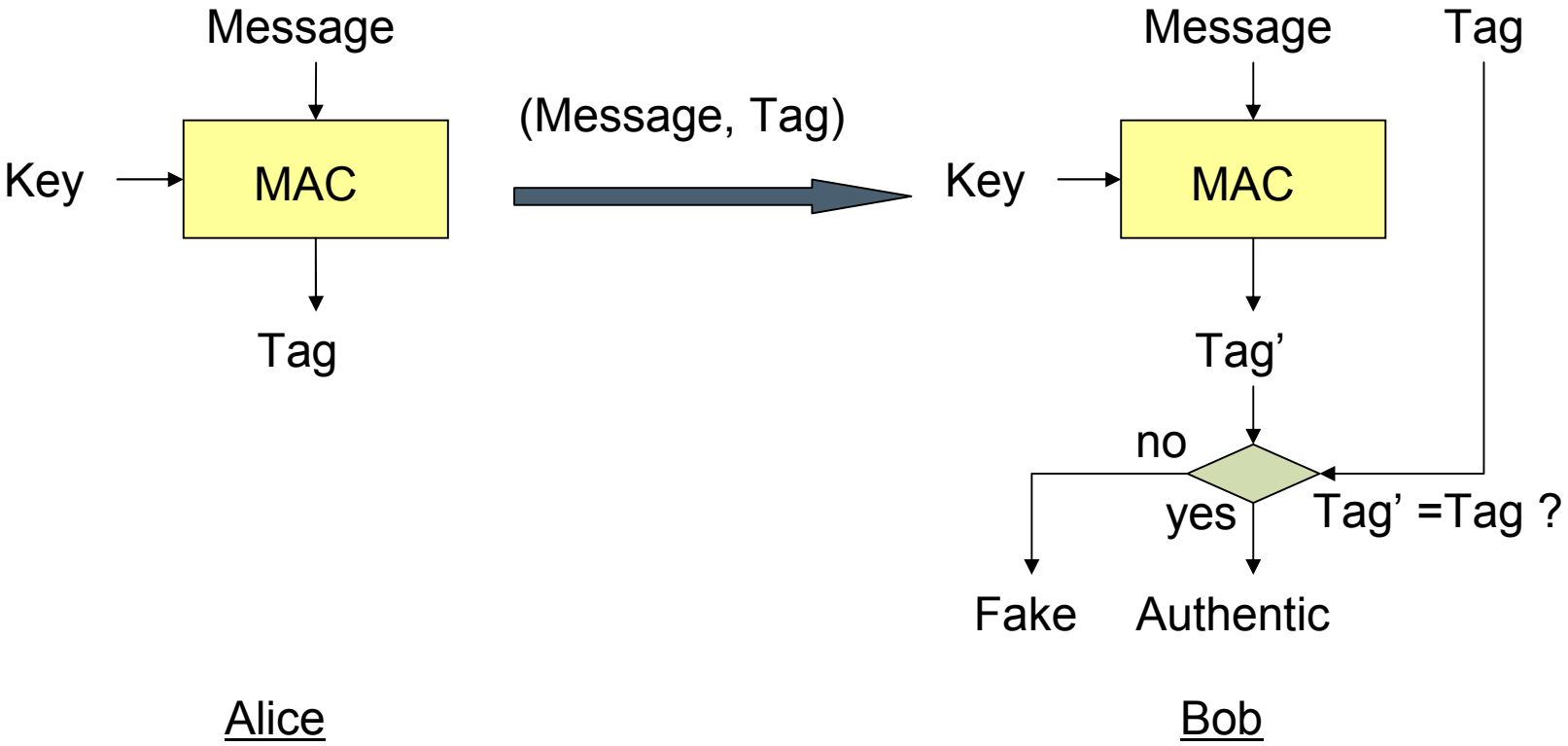
Provably Secure MACs from Differentially-uniform Permutations and AES-based Implementations

Kazuhiko Minematsu and Yukiyasu Tsunoo
NEC Corporation

Fast Software Encryption 2006, Graz, Austria

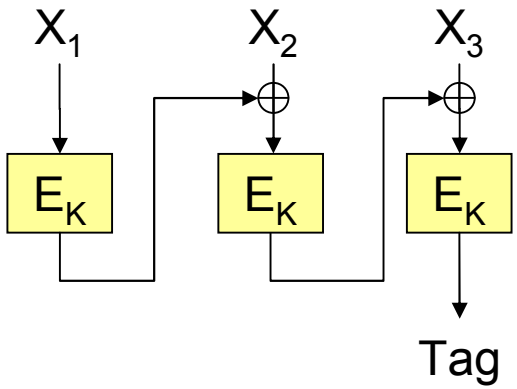
Message Authentication Code (MAC)

- Symmetric-key primitive to assure the authenticity of messages
- Tags must be unpredictable w/o the key



MAC modes of operations

- Block cipher (BC)-based MACs
  - CBC-MAC, EMAC [R95], XCBC [BR00], RMAC [JJV02], TMAC [KI03] , OMAC [IK03], etc.
- Goal: Provable security if BC = PRP (some exceptions exist)
  - Hard to fake a tag using any practical chosen-plaintext attack (CPA)



CBC-MAC

Motivation

- Speed of all MAC modes $\fallingdotseq$ BC's speed
- Can we have a BC-based MAC faster than the BC?
  - Typical solution is to introduce a well-optimized universal hash function (aka Carter-Wegman MAC), but we stick to BC-based MACs

Previous work: ALRED MAC [DR05a]

- MAC with (BC + reduced-round BC)
  - Chain of reduced-round BCs
- Many good properties
  - Faster-than-CBC-MAC
    - ✓ MAC speed ≒ reduced-round BC's speed
  - Very small amount of preprocessing
  - Provably secure against attacks w/o "internal collisions"
- Provably secure against all CPAs?
  - Unclear. Depending on the structure of the reduced-round BC

Our contributions

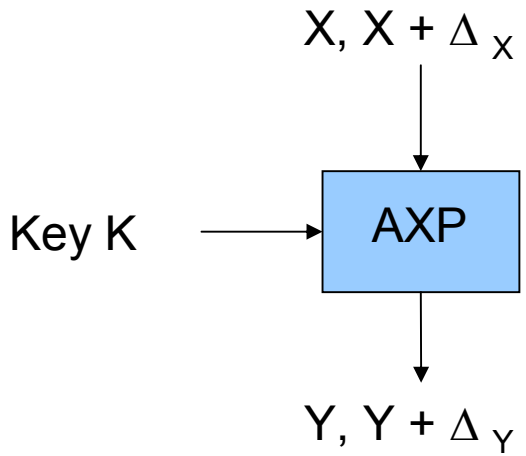
- MAC with (BC + Auxiliary Permutation (AXP))
  - AXP is an n-bit (possibly keyed) permutation and typical example is a reduced-round BC
  - Faster than CBC-MAC if AXP = a reduced-round BC
- Provably secure if AXP is differentially-uniform (= secure against DC) and BC = PRP
- (AES + 4-round AES) implementations
  - 1.4 ~ 2.5 times faster than the CBC-MAC-AES

Differentially-uniform permutation

- "AXP is ε-diff-unif" means:
  - Max. expected differential probability (MEDP) is ε

$$\max_{\Delta_X \neq 0, \Delta_Y} \Pr[\mathrm{AXP}_K(X) - \mathrm{AXP}_K(X + \Delta_X) = \Delta_Y] \leq \epsilon$$

  - A natural requirement of a (reduced-round) BC
- MEDP = MDP for a fixed permutation



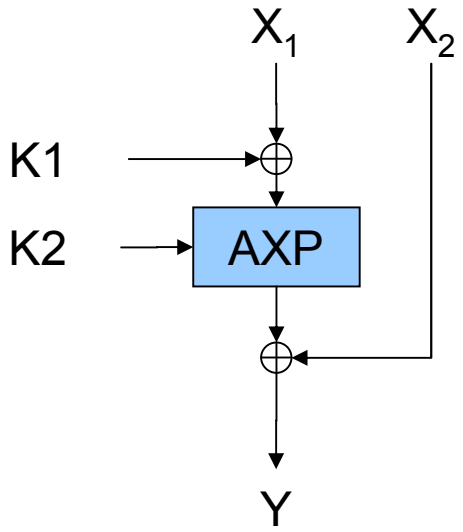
Road map

- First, we build two almost universal (AU) hash functions
  - Tree-structured AU hash
  - Iterative AU hash
    - ✓ Different characteristics in speed, memory, and preprocessing
- Then, we convert them to MACs
  - Tree-structured MAC
  - Iterative MAC (similar, but not identical to ALRED)
- Finally, we build MACs using AES

Tree-structured AU hash function

Basic observation

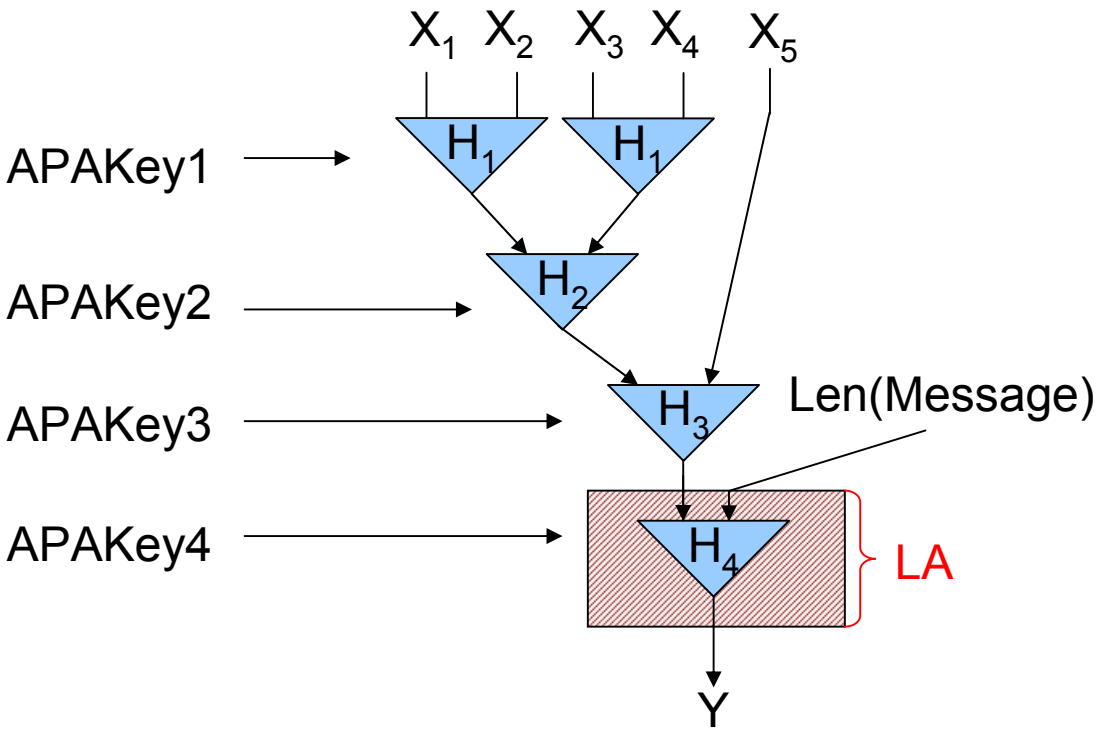
- Add-Permute-Add (APA) function : $\{0,1\}^{2n} \rightarrow \{0,1\}^{n}$ is ε-AU if $MEDP(AXP_{K2}) \leq \varepsilon$



APA function (APAKey = (K1, K2))

Known schemes to extend the input length

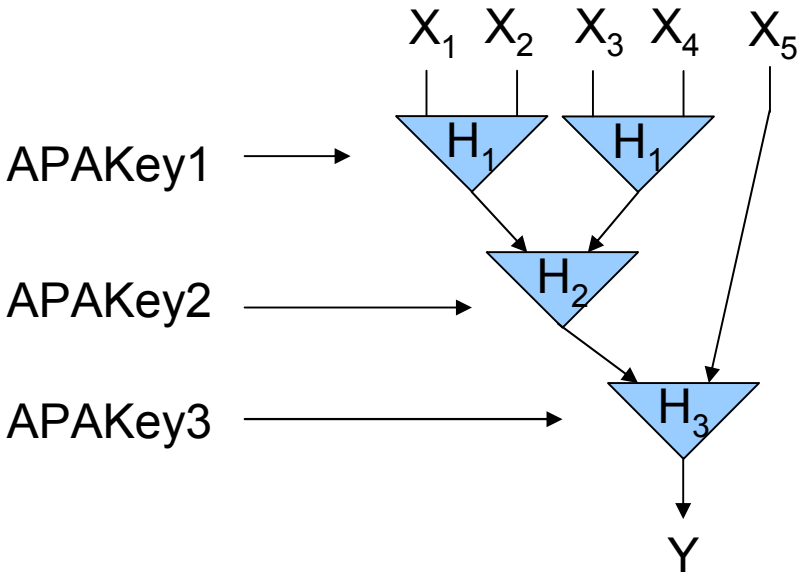
- Use Tree Hash [WC81] or Modified Tree Hash (MTH) [BCZ05] to extend the input length of APA
- Length Annotation (LA) was used to prevent collisions between unequal-length messages



Modified Tree Hash (w/ LA)

Our proposal: Modified Tree Hash w/o LA

- LA can be removed if MTH uses APA function
  - Due to the uniformity of APA's output for any input
- Improved efficiency, particularly for short messages
  - For m-block, (m-1) APA calls, $\lceil \log_2 m \rceil$ APA keys



Modified Tree Hash (w/o LA)

Collision probability of MTH w/o LA

- For any m and m'-block inputs, the maximum collision probability is

$$\mathrm{Coll}_{\mathrm{MTH}_{\mathbf{H}}}(m, m') \leq \max\{\lceil \log_2 m \rceil, \lceil \log_2 m' \rceil\} \cdot \epsilon$$

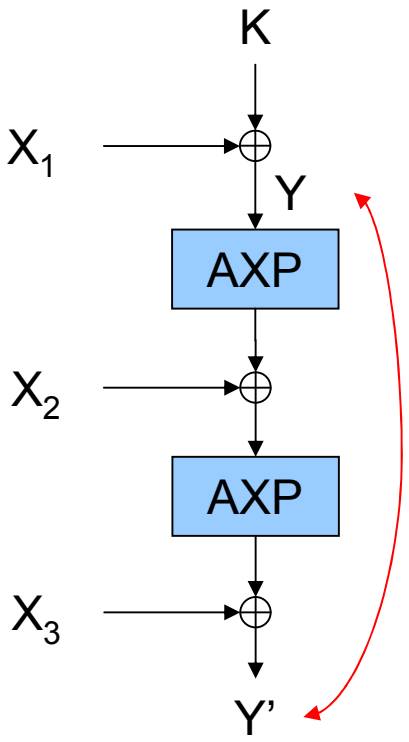
ϵ = MEDP(AXP_K)

Iterative AU hash function

Simple iteration is not always good

- a pair of *non-adaptively chosen* inputs yields a collision with prob. ≈1 if AXP = inversion on GF(2^n)
  - MDP(inv) = $4/2^n$ [N94]

$$\mathrm{inv}(x) = \begin{cases} x^{-1} & \text{if } x \neq \mathbf{0} \\ \mathbf{0} & \text{if } x = \mathbf{0} \end{cases}$$



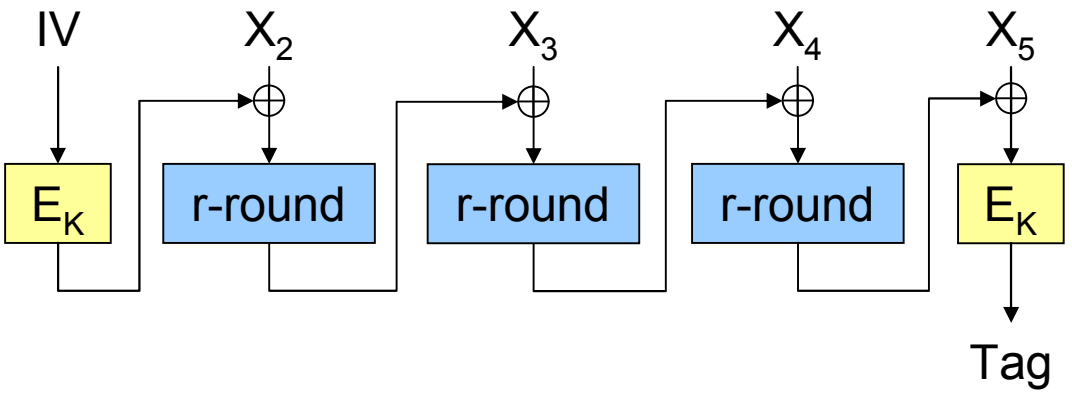
input	output
X_1	$\rightarrow Y = K \oplus X_1$

$(X'_1, \mathbf{0}, X_1 \oplus X'_1) \rightarrow Y' = (X_1 \oplus X'_1) \oplus (\mathbf{0} \oplus (K \oplus X'_1)^{-1})^{-1}$

$= K \oplus X_1$ w/ prob. ≈ 1

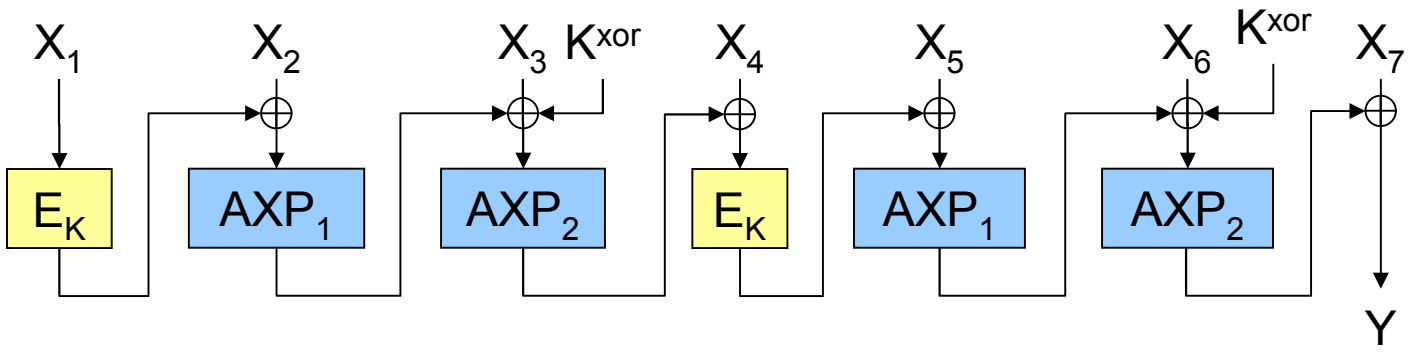
On ALRED

- A variant of ALRED uses one-key iteration with some padding scheme, and we can find a collision if AXP = inv
- Differential-uniformity is not sufficient to provide provable security with ALRED
- Caveat: this does not mean the total insecurity of ALRED, since reduced-round BCs are generally much more complex than inversion
  - A stronger assumption is needed...



Our proposal: Periodic CBC Hash (PCH)

- CBC-MAC-like chaining of BC and AXP
  - BC is called for every d+1 blocks (d is the interval)
  - For interval d, use d AXPs and (d-1) xor-keys
  - Interval defines the treading-off between speed and key length



PCH with interval = 2

Collision Probability of PCH

- If BC is truly random, then the max. collision prob. is:

$$\mathrm{Coll}_{\mathrm{PCH}_d[R,\mathbf{G}]}(m,m') \le d\epsilon_{\mathrm{dp}} + \epsilon_{\mathrm{sdp}} + \frac{(\lceil \frac{m}{d+1} \rceil + \lceil \frac{m'}{d+1} \rceil)^2 + 2}{2^{n+1}},$$

interval MEDP(AXP$_K$) MESDP(AXP$_K$)

- Additional (maybe weak) condition : Max. expected self-differential probability (MESDP) is small

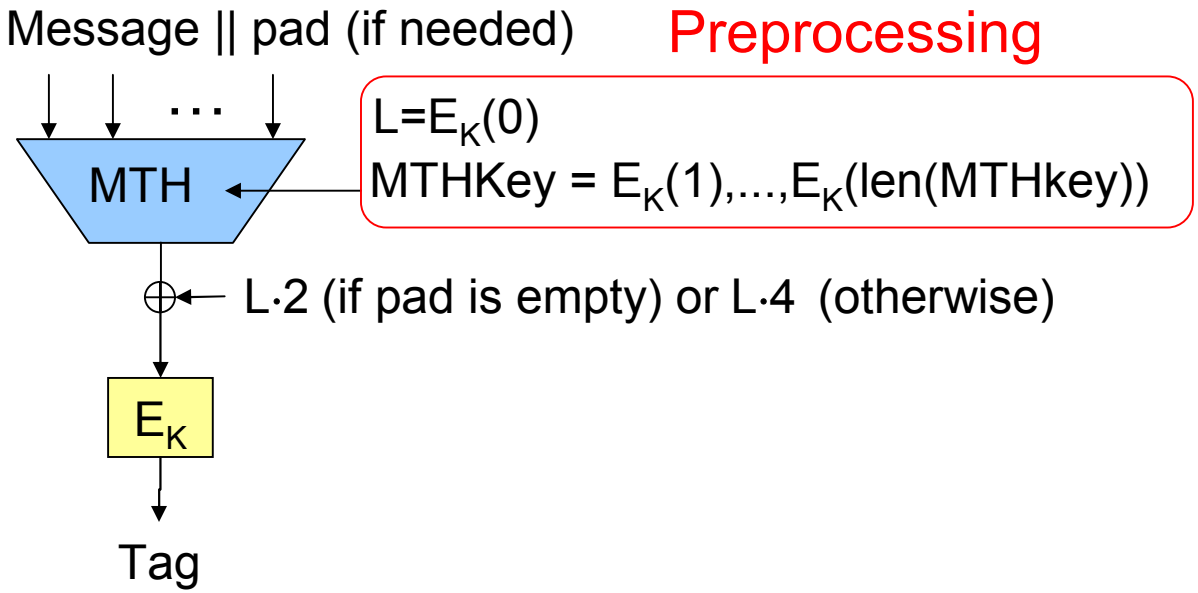
$$\max_{\Delta_Y} \Pr[X - \mathrm{AXP}_K(X) = \Delta_Y] \le \epsilon$$

- Computational AU hash if BC = PRP

From AU hash function to MAC

MT-MAC from MTH, PC-MAC from PCH

- TMAC and OMAC-like tweaking
  - (MTH or PCH) + one BC call + preprocessing
  - One-key for MT-MAC (K), two-key for PC-MAC (K and L)



MT-MAC (PC-MAC is similarly defined)

Security of our MACs

- For MT-MAC with max. message length = $n2^b$,

$$\mathtt{Adv}^{\mathtt{vilprf}}_{\text{MT-MAC}_b}(q,t,\sigma) \leq \mathtt{Adv}^{\mathtt{prp}}_{E_K}(\sigma+c,t') + \frac{(\sigma+c)^2}{2^n} + \epsilon_{\text{dp}}\sigma^2$$

- For PC-MAC with interval d,

c: a function of b (or d) and AXP's key length

$$\mathtt{Adv}^{\mathtt{vilprf}}_{\text{PC-MAC}_d}(q,t,\rho) \leq \mathtt{Adv}^{\mathtt{prp}}_{E_K}(\rho q+c,t') + \frac{2.5(\rho q+c)^2}{2^n} + (d\epsilon_{\text{dp}}+\epsilon_{\text{sdp}})\frac{q^2}{2}$$

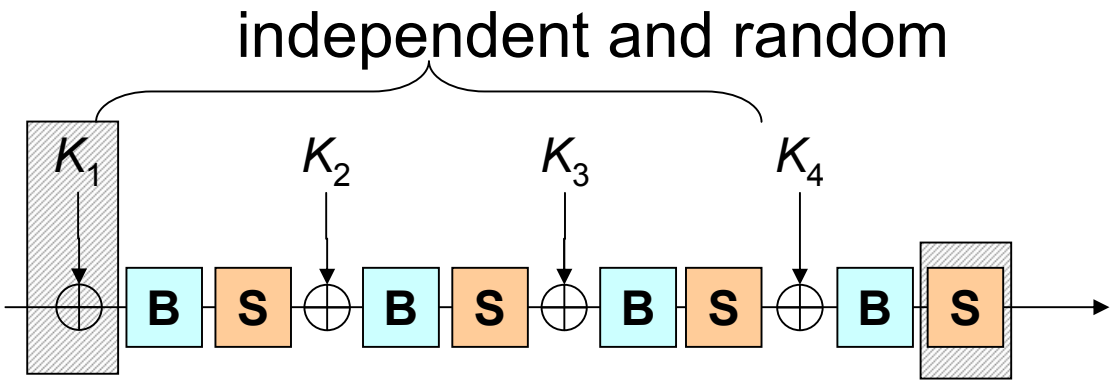
- Rough summary: they are secure if adversary's resource parameters are much smaller than $(\sqrt{\epsilon_{\text{dp}}})^{-1}$

  - q :number of queries
  - σ :number of total message blocks
  - ρ :max. length of a message (in block)

AES-based Implementations

MEDP and MESDP of 4-round AES

- MEDP $\leq 2^{-113}$ [K05] and MESDP = 2^{-128}
  - MESDP = 2^{-128} for all XORing-round-key structures
  - The first round key and the last diffusion layer can be removed
- We can use (AES + 4rAES)



4rAES

B: ByteSub

S: ShiftRow & MixColumn

Comparison

- (Good) Average 2.5 times faster (MT-MAC), 1.4 to 2.5 times faster (PC-MAC) than OMAC
- (Bad) Increased preprocessing and slightly-degraded security (due to the MEDP of 4rAES), and key length (for PC-MAC)

MAC	MsgLen	Rounds	Preproc	Key	Security (bit)
MT-MAC$_b$	$n2^b$	4	$4b+1$	128	56.5
PC-MAC$_d$	∞	$4+\frac{6}{d+1}$	$4d-1$	256	56.5
OMAC	∞	10	1	128	64
Pelican	∞	4	1	128	Unknown

Pelican MAC [DR05b]: ALRED using (AES + 4rAES)

Software Implementation

- Public-domain C source of AES by Rijmen et al.
- Gap between theory (2.5 times) and practice (2.0 times)
  - mainly due to the byte/word conversion

MAC	Tag computation (cycle/byte)	Preprocessing (cycles)
MT-MAC ($b = 32$)	12.5	53777 (estimate)
PC-MAC ($d = 1$)	18.5	1651
PC-MAC ($d = 5$)	14.4	8311
PC-MAC ($d = 17$)	13.1	28444
OMAC	25.1	821

Conclusion

- Provably secure MACs with (BC + reduced-round BC) that are faster than the CBC-MAC
- AES-based implementations
- Future work
  - Reduced preproc. of AES-based implementations
    - ✓ Use 4rAES w/ (partially) fixed keys?
  - Some improvements about PC-MAC (wrt key length and the security proof, etc.)

Thank you!

Questions? I’m so sleepy…