# CS 578 – Cryptography

Prof. Michael Backes

## Message Integrity, Collision-resistant Hash Functions

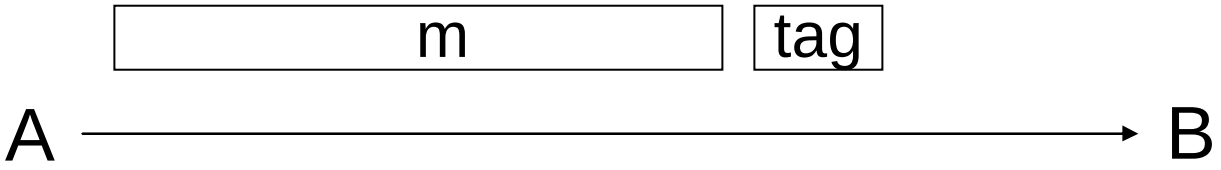
May 12, 2006

# Administrative Announcements

- Unsubscribe from the course:
  - In case you want to, last chance today
- Important: Changes concerning backup exam
  - Old solution: If you pass the final exam, you may write the backup exam for improving your grade → grade of backup exam counts, but no solid legal ground
  - Likely to be new solution: If you pass the final exam, you cannot write the backup exam

# Message Integrity

- Goal of message integrity:




- Alice generates tag, Bob verifies tag
- Goal: Attacker cannot change message, i.e., attacker cannot generate any valid pair (m,tag)

# Definition of MAC

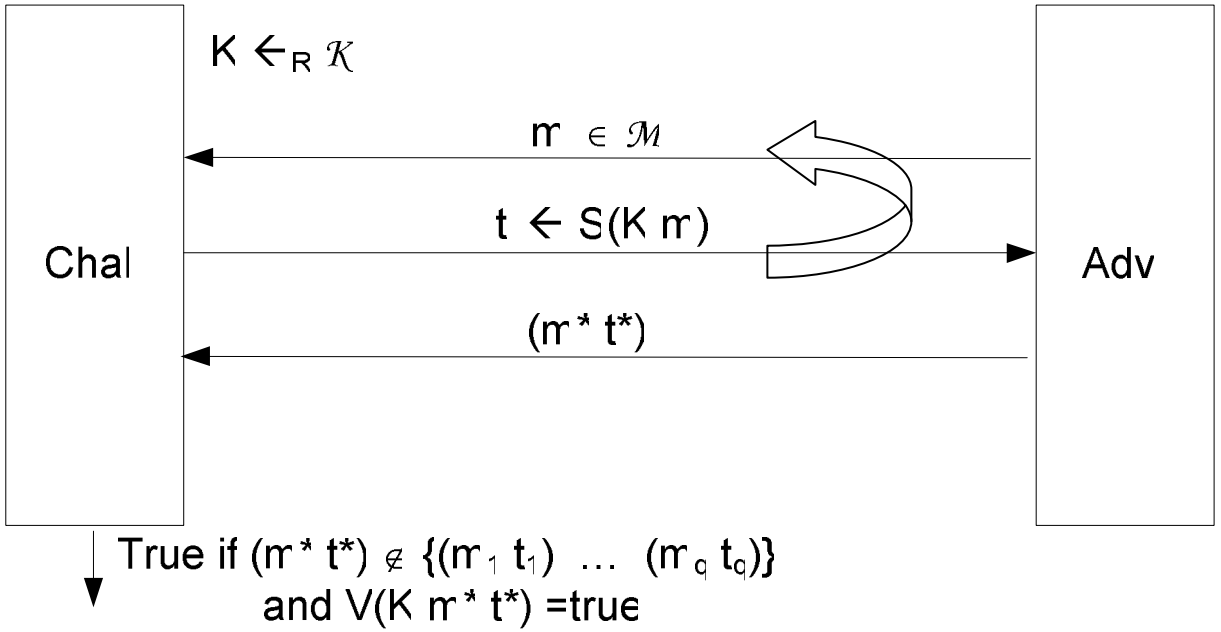
- Definition (MAC): A message authentication code (MAC) defined over $(\mathcal{K},\mathcal{M},\mathcal{T})$ is a pair (S,V) of efficient algorithms (S,V) where

  S: $\mathcal{K}\times\mathcal{M}\rightarrow\mathcal{T}$ and V: $\mathcal{K}\times\mathcal{M}\times\mathcal{T}\rightarrow \mathcal{B}ool$

  s.t. for all $m\in\mathcal{M}$, $K\in\mathcal{K}$: $(t \leftarrow S(K,m)) \rightarrow (V(K,m,t) = true)$
- Definition (Secure MACs, intuitively):
  - Attacker's power: chosen-message attack
    Attacker outputs $m_1 \ldots m_q$, and gets $t_i \leftarrow S(K,m_i)$
  - Attacker's goal: existential forgery
    Produce valid pair (m*,t*), i.e., V(K,m*,t*) = true, where $(m^*,t^*) \notin \{(m_1,t_1), \ldots, (m_q,t_q)\}$
- Attacker cannot even forge MACs on non-sensitive messages

# MACs

- For MAC I=(S,V), define the following MAC game:



# Definition of Secure MACs

- The advantage of adversary A attacking I is $Adv^{MAC}[A,I]$ = Pr[Challenger outputs true]

- Definition (Secure MACs): A MAC I=(S,V) is a secure against existential forgery under chosen-message attack (CMA) if for all efficient algorithms A: $Adv^{MAC}[A,I]$ is negligible.

# PRFs and MACs

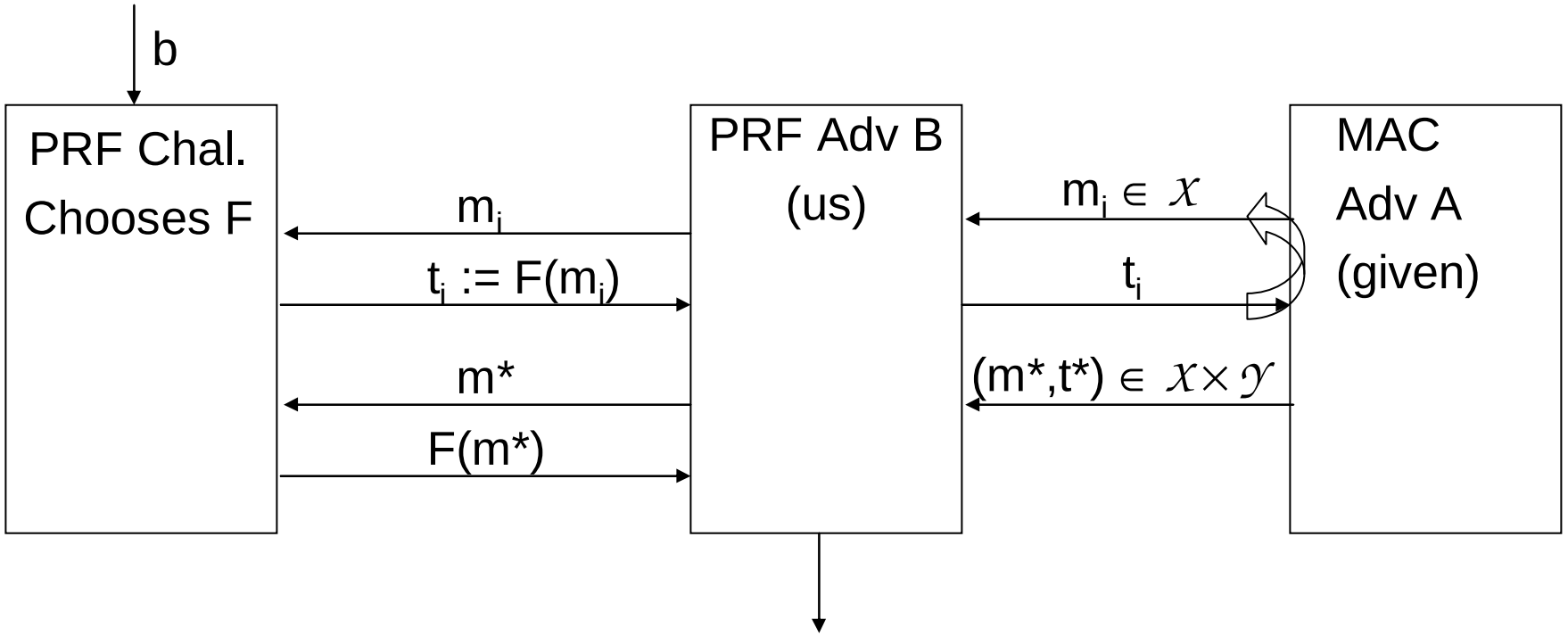
- Claim: Any PRF with sufficiently large range is a secure MAC
- Construction: Let E be a PRF over $(\mathcal{K},\mathcal{X},\mathcal{Y})$. Define $I_E$ := (S,V) as follows:
  - S(K,m) = E(K,m)
  - V(K,m,t) = true if t = E(K,m), false otherwise
- Example: AES is a MAC for 16-byte messages

# PRFs and MACs (cont'd)

- Lemma: If E is a PRF over $(\mathcal{K},\mathcal{X},\mathcal{Y})$ where $1/|\mathcal{Y}|$ is negligible, then $I_E$ is a secure MAC.
- In particular, for adversary A attacking MAC, there is an adversary B attacking E such that
  $Adv^{MAC}[A,I_E] \leq Adv^{PRF}[B,E] + 1/|\mathcal{Y}|$.

  [proof on the board]

# Proof Overview




0 if $F(m^*) = t^*$ and $(m^*,t^*) \notin \{(m_1,t_1),\ldots,(m_q,t_q)\}$

1 otherwise

# From small MACs to big MACs

- Question: Given a small-MAC, how to build a big-MAC?
- Method: given a small PRF (e.g., AES) build a big-PRF (which is then a MAC again)

# Standard Constructions

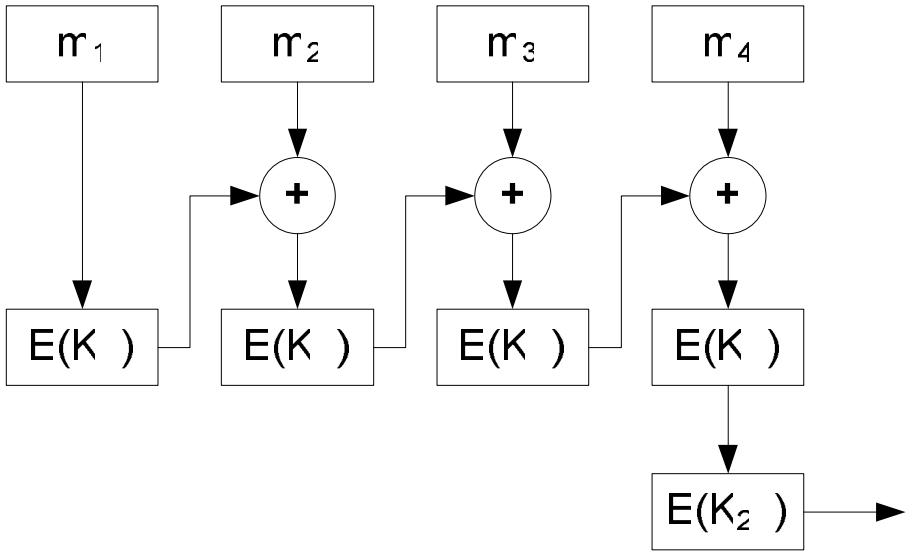
- CBC-MAC:
  - Banking
  - ANSI X9.9, X9.19
  - ISO
  - FIPS 186-3
- (PMAC: Not standardized but cute research material)
- HMAC:
  - Internet: SSL, IPSec, SSHv2

# Truncating MACs

- Suppose that a MAC I is a PRF over $(\mathcal{K},\mathcal{X},\mathcal{Y})$, outputting n bit tags
- Claim: OK to truncate MACs to w<n bits as long as $(½)^w$ is negligible.
- ANSI X9.19: Truncate to w=64,48,32 bits
  - Banking industry
  - ACH (Automated Clearing House) network

# (Encrypted) CBC-MAC

- Let E be a PRF over $(\mathcal{K},\mathcal{X},\mathcal{X})$, e.g., AES
- Define a PRF $E^{CBC}$ (and thus also a MAC) as follows:



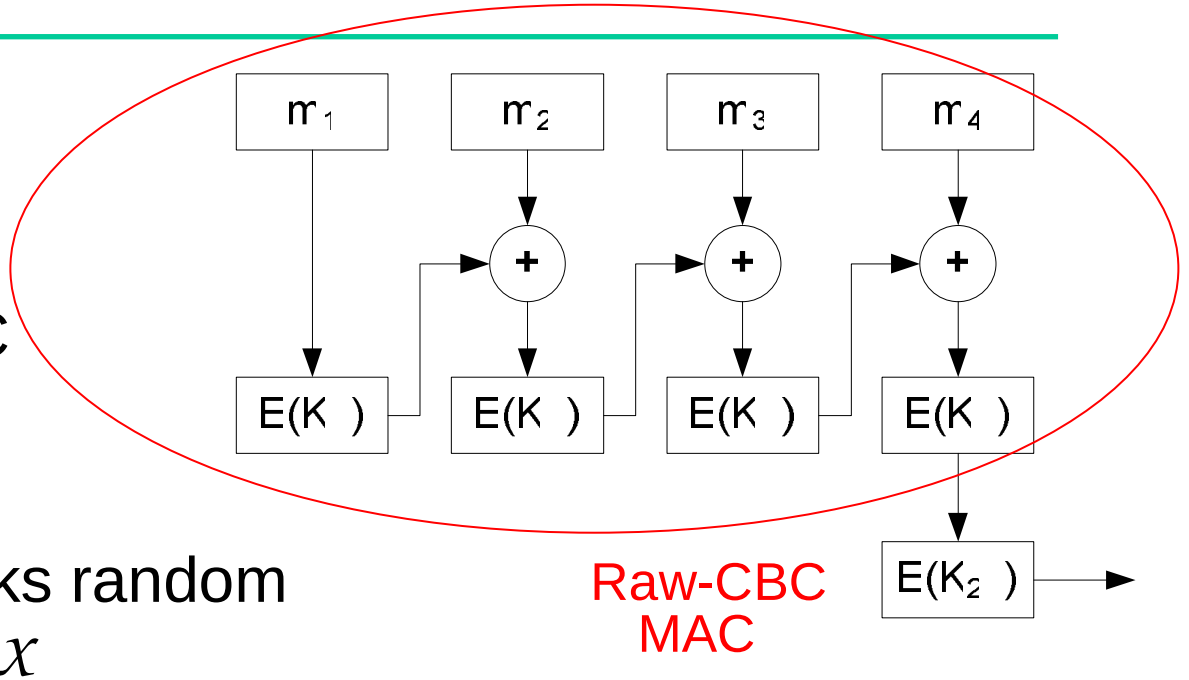

- $E^{CBC}$ is a function from $\mathcal{K}^2 \times \mathcal{X}^L \rightarrow \mathcal{X}$

# (Encrypted) CBC-MAC theorem

- CBC-MAC theorem: For any L > 0, if E is a PRF over $(\mathcal{K},\mathcal{X},\mathcal{X})$, then $E^{CBC}$ is a PRF over $(\mathcal{K}^2,\mathcal{X}^L,\mathcal{X})$
- In particular, for any q-query A attacking $E^{CBC}$ there exists an adversary B such that $Adv^{PRF}[A,E^{CBC}] \leq Adv^{PRF}[B,E] + q^2/|\mathcal{X}|$.
- CBC-MAC is secure as long as $q << x^{1/2}$

# Raw CBC-MAC

- Why the last encryption?
- Raw CBC-MAC is an insecure MAC!
- Attack: Adv picks random one-block m $\in \mathcal{X}$
- Chosen-message attack:
  Adv. request tag (MAC) for message m and gets t := E(K,m)
- Output t as MAC forgery on message (m, t $\oplus$ m).



# On Raw CBC-MAC and PRFs

- Claim: The tag t is really an existential forgery against Raw CBC-MAC.

  [proof on the board]

- Note: Raw CBC-MAC is a secure PRF for fixed message size
- Prepending message length also works, but not elegant

# Padding of CBC-MAC

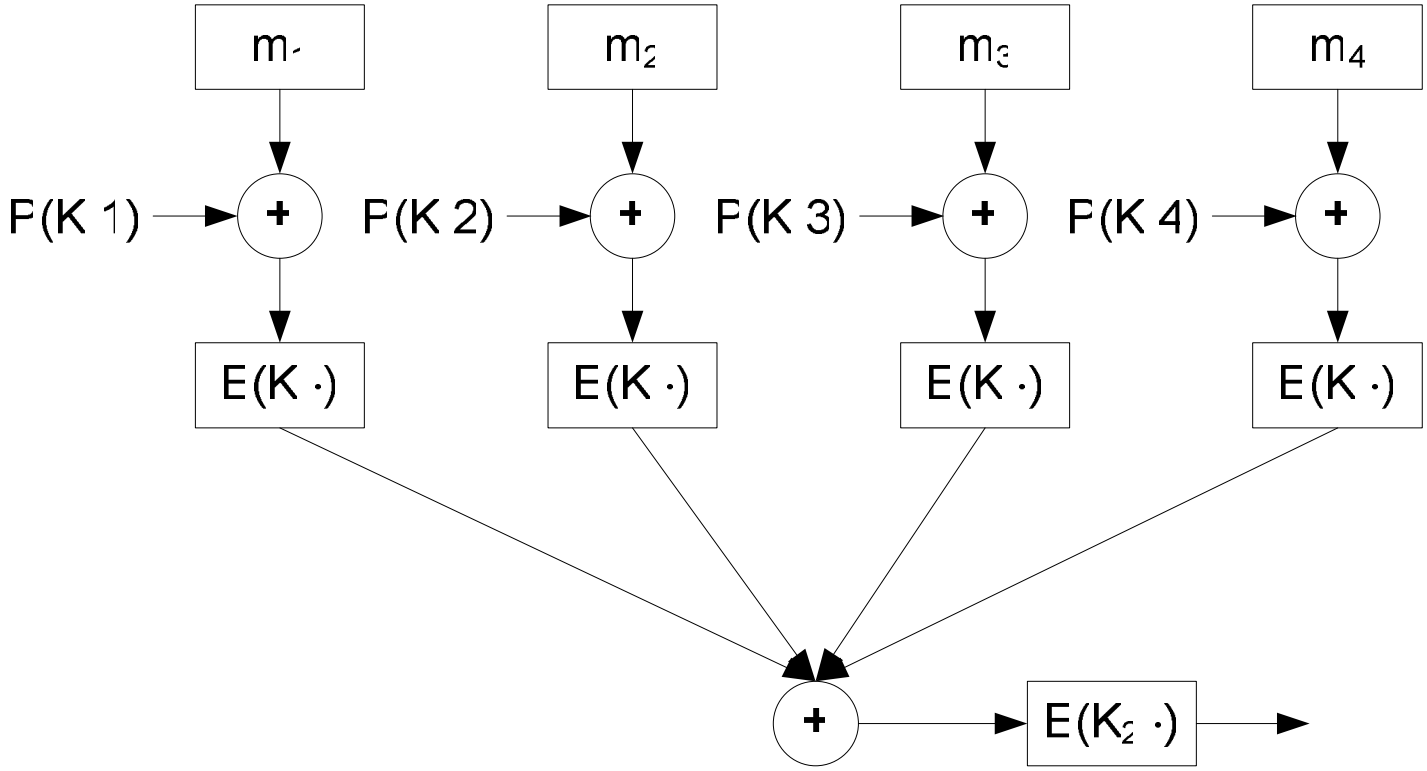
- CBC-MAC padding
- What if |m| is not a multiple of block size?
- Bad idea: Append 0's until |m| is multiple of block size
- Simple chosen-message attack:
  - ask for MAC on m
  - Obtain MAC on m||0, on m||00, on m||000, etc.

# Padding in Standards

- ISO: append “100…0”
  (if |m| is a multiple of the block size, append a new block “100…0”)
- NIST (CMAC): pad CBC-MAC without ever adding extra blocks

# PMAC – Parallel MAC

- Usual problem with CBC: sequential
- (One) remedy: PMAC – Parallel MAC



# PMAC Theorem

- PMAC Theorem: For any L > 0, if E is a PRF over $(\mathcal{K},\mathcal{X},\mathcal{X})$, then $E^{PMAC}$ is a PRF over $(\mathcal{K}^2,\mathcal{X}^L,\mathcal{X})$.
- In particular, for any q-query A attacking $E^{PMAC}$ there exists an adversary B such that $Adv^{PRF}[A,E^{PMAC}] < Adv^{PRF}[B,E] + 2q^2L^2/|\mathcal{X}|$.
- PMAC is secure as long as $q << |\mathcal{X}|^{1/2}/L$
- PMAC is incremental:
  Say we computed $t \leftarrow E^{PMAC}(K,m)$ for long m
  If m changes one block ($\rightarrow$ m* ) then $t^* \leftarrow E^{PMAC}(K,m^*)$ can be computed really fast

# Examples of MACS

- So far:
  - CBC-MAC, used by banks, etc., sequential
  - PMAC = Parallel MAC, not used in practice, incremental
- Now: HMAC, used in lots of Internet protocols, incremental
- First: Hash functions and collision resistance

# Hash Functions

- Let H : M → T be a hash function (non-keyed) (often H : {0,1}* → {0,1}$^n$)
- A collision for H is a tuple ($m_1$,$m_2$) with H($m_1$)=H($m_2$) $\wedge$ $m_1 \neq m_2$.
- "Definition" (Collision Resistant Hash Function, CRHF): A hash function is collision resistant if no algorithm is known that finds a collision for H in suitable time.
- Remark: Defining that "no efficient adversary exists that finds a collision" cannot be fulfilled

# Examples of CRHFs

- Used to have lots of CRHFs examples
- Broken:
  - MD5 (broken): 128-bits digest, 216 MB/s
  - SHA-1 (broken) 160-bits digest, 68 MB/s
  - (Not only non-sensical collisions for MD5 but selective ones.)
- Currently still available
  - SHA-256: 256-bits digest, 44.5 MB/s
  - Whirlpool (AES):  512-bits digest, 12.1 MB/s

# CRHFs and MACs

- Construction of big-MACs from small-MACs and CRHFs:
  - Let I=(S,V) be a secure MAC for short messages (e.g., AES) over $(\mathcal{K},\mathcal{M},\mathcal{T})$
  - Let H be collision-resistance hash function: H: $\mathcal{M}^{big} \rightarrow \mathcal{M}$
- Definition (big-MACs from small-MACs): Let I*=(S*,V*) be a MAC over $(\mathcal{K},\mathcal{M}^{big},\mathcal{T})$ where
  - S*(K,m) := S(K,H(m))
  - V*(K,m,t) = true iff V(K,H(m),t) = true
- Theorem: I* is a secure MAC.
- → AES(K,H(m)) is a secure MAC if H is a CRHF.
- How to build collision-resistant hash functions?