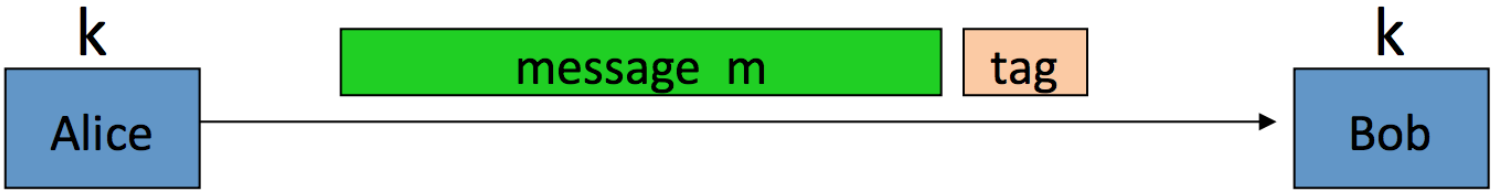
Tutorial 3 Crypto: Message Integrity

Teodora Baluta

March 16, 2018

Message integrity: MACs



Generate tag:
tag ← S(k, m)

Verify tag:
V(k, m, tag) $\stackrel{?}{=}$ `yes'

Def: **MAC** I = (S,V) defined over (K,M,T) is a pair of algs:

- S(k,m) outputs t in T
- V(k,m,t) outputs `yes' or `no'

MACs

Message **A**uthentication **C**ode ensures

- authenticity: the message comes from the person with the shared key
- integrity: no tampering attacks, attacker cannot modify message
- Is CRC a MAC?

MACs

Message **A**uthentication **C**ode ensures

- authenticity: the message comes from the person with the shared key
- integrity: no tampering attacks, attacker cannot modify message

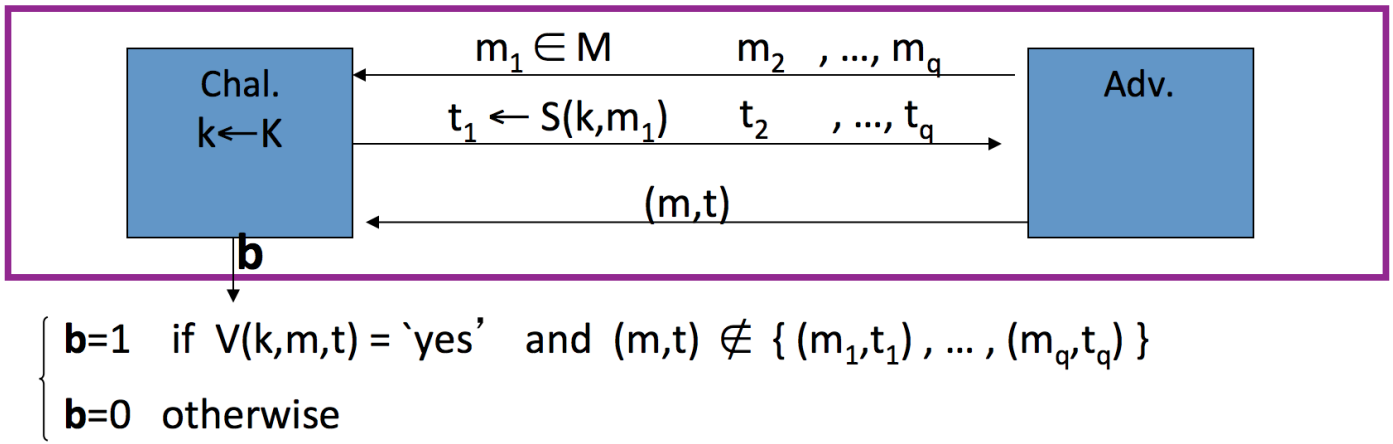
- Is CRC a MAC?
- No! MAC requires a **shared key**!

Existential Forgery

Chosen message attack: for m_1, m_2,...,m_q attacker get $t1$, $t2$, ...,t_q
Existential Forgery: when an attacker can create a tag for another message m

What is a Secure MAC?

- Adversary model: chosen message attack (m_1, m_2,...,m_q and $t1$, $t2$, ...,t_q)
- The attacker has to be able to produce $(m, t) \neq (m_i, t_i)$, $i \in \{1, ..., q\}$ that "tricks" the challenger
- The MAC is secure if for all "efficient" adversaries the probability that the challenger outputs 1 is negligible



Pseudo Random Function (PRF)

- F is a PRF, $F : K \times X \rightarrow Y$, defined over (K, X, Y) if there exists an efficient algorithm to evaluate $F(k, x)$
- Secure PRF
  - Let Funs[X, Y] be the set of all functions from X to Y
  - $S_F = \{F(k, x)$ such that $k \in K$ and $x \in X\} \subseteq Funs[X, Y]$
  - F is a secure PRF if it is indistinguishable from a random function

Example PRFs

- AES-128: $\mathsf{K} \times \mathsf{X} \rightarrow \mathsf{X}$ where $K = X = \{0,1\}^{128}$
- DES: $\mathsf{K} \times \mathsf{X} \rightarrow \mathsf{X}$ where $X = \{0,1\}^{64}$, $K = \{0,1\}^{56}$
- 3DES: $\mathsf{K} \times \mathsf{X} \rightarrow \mathsf{X}$ where $X = \{0,1\}^{64}, K = \{0,1\}^{168}$

MAC from Secure PRF

- We define MAC $I_F = (S_F, V_F)$ where $F : K \times X \to Y$ is a secure PRF
  - $S_F(k, m) = F(k, m)$
  - $V_F(k, m, t) =$ yes if $t = F(k, m)$, else no
- Example: MAC constructed with $F : K \times X \to Y$ with $Y = \{0, 1\}^{16}$

MAC from Secure PRF

- We define MAC as a pair $I_F = (S_F, V_F)$ where $F : K \times X \to Y$ is a secure PRF
  - $S_F(k, m) = F(k, m)$
  - $V_F(k, m, t) =$ yes if $t = F(k, m)$, else no
- **Bad Example**: MAC constructed with $F : K \times X \to Y$ with $Y = \{0, 1\}^{16}$
  - The adversary can guess the tag with probability $1/2^{16}$

MAC from Secure PRF

(Theorem) If $F : K \times X \to Y$ is a secure PRF **and** $|Y|$ **is large**, then $I_F = (S_F, V_F)$ is a secure MAC.

- $|Y|$ is large, say $|Y| = 2^{80}$

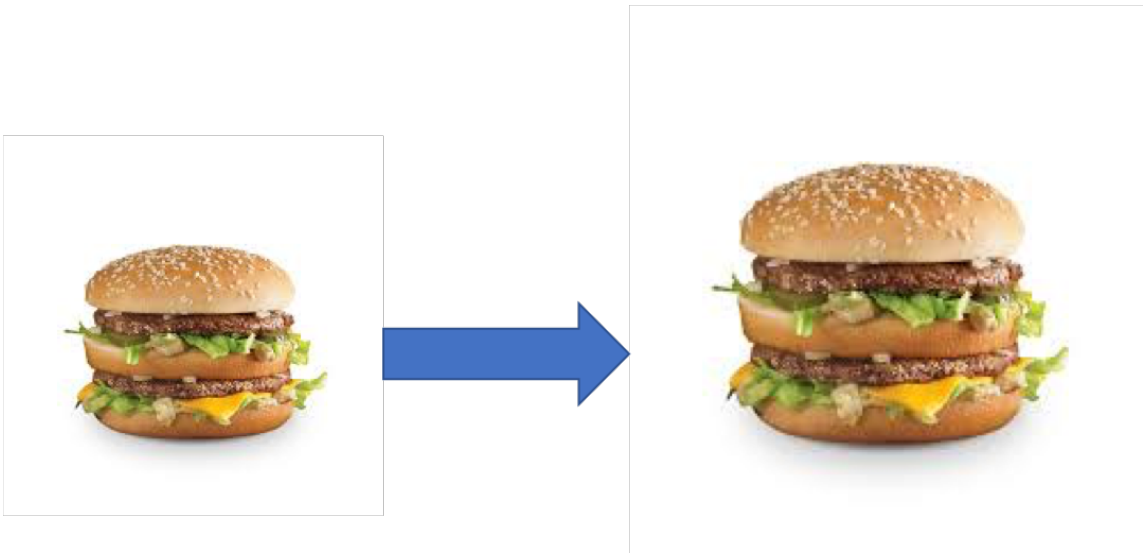
MAC from Secure PRF

(Theorem) If $F : K \times X \to Y$ is a secure PRF **and** $|Y|$ **is large**, then $I_F = (S_F, V_F)$ is a secure MAC.

- $|Y|$ is large, say $|Y| = 2^{80}$
- Then we can use AES-128!

Big MAC from Small MAC

- AES-128 takes as input 16-byte messages
- In practice, we want to compute tags for files (large amount of data). How do we go from small-PRF to big-PRF?



Naive Construction

- Break the message into blocks of 16 bytes
- $S(k, m) = F(k, m_1) \parallel F(k, m_2) \parallel F(k, m_3) \parallel .... \parallel F(k, m_q)$
- $V(k, m, t) = V(k, m_1, t_1) \wedge V(k, m_2, t_2) \wedge ... \wedge V(k, m_q, t_q)$

Naive Construction

Security Issues?

- swap the internal blocks of a message, to obtain a valid tag of a different message (same length)

Naive Construction

Security Issues?

- swap the internal blocks of a message, to obtain a valid tag of a different message (same length)
  - m = (m_1, m_2), I get S(k,m) = F(k, m_1) $\|$ F(k, m_2)

Naive Construction

Security Issues?

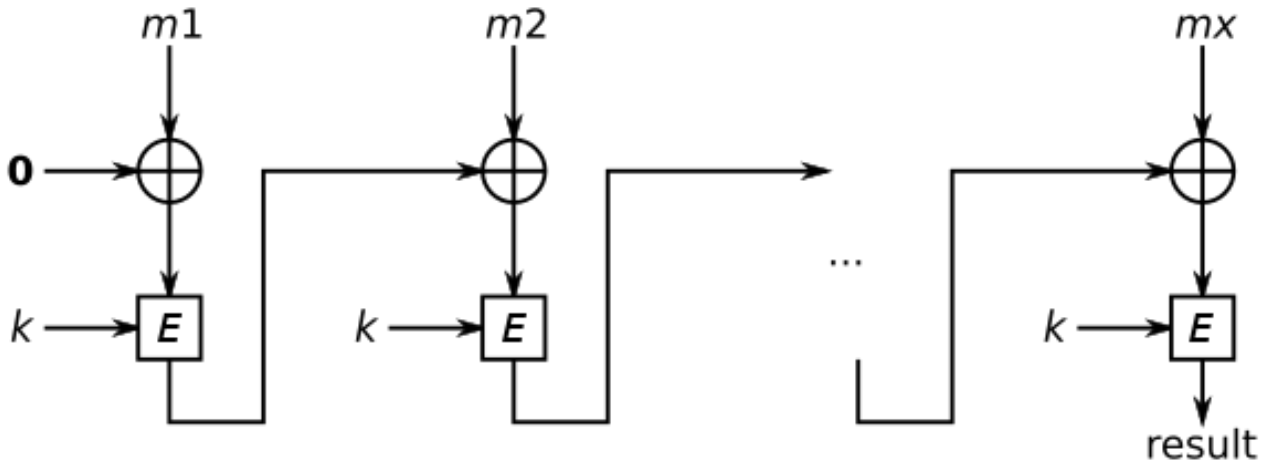
- swap the internal blocks of a message, to obtain a valid tag of a different message (same length)
  - m $= (m_1, m_2)$, I get S(k,m) = F(k, m_1) $\|$ F(k, m_2)
  - m' $= (m_2, m_1)$ I can construct S(k, m') = F(k, m_2) $\|$ F(k, m_1)

Naive Construction

Security Issues?

- swap the internal blocks of a message, to obtain a valid tag of a different message (same length)
  - m $= (m_1, m_2)$, I get S(k,m) = F(k, m_1) $\|$ F(k, m_2)
  - m' $= (m_2, m_1)$ I can construct S(k, m') = F(k, m_2) $\|$ F(k, m_1)
- **length extension attack**: can construct S(k, $m_1 \| m_2$) from tags for $m1$ and m_2

Raw CBC



- Split the message in blocks $m_1, m_2, ..., m_q$
- $c_i = E(k, m_i \oplus c_{i-1})$
- Raw CBC is secure for fixed-length messages

Raw CBC (variable-length messages)

Length Extension Attack

- Adversary gets pairs (m, t) and (m', t')

Raw CBC (variable-length messages)

Length Extension Attack

- Adversary gets pairs (m, t) and (m', t')
- Existential forgery attack: let $m'' = m \parallel m'$

Raw CBC (variable-length messages)

Length Extension Attack

- Adversary gets pairs (m, t) and (m', t')
- Existential forgery attack: let $m'' = m \parallel m'$
- Replace first block of m' (m'_1) with $m'_1 \oplus t \implies$ $m'' = m \parallel (m'_1 \oplus t) \parallel m'_2 \parallel ...m'_q$

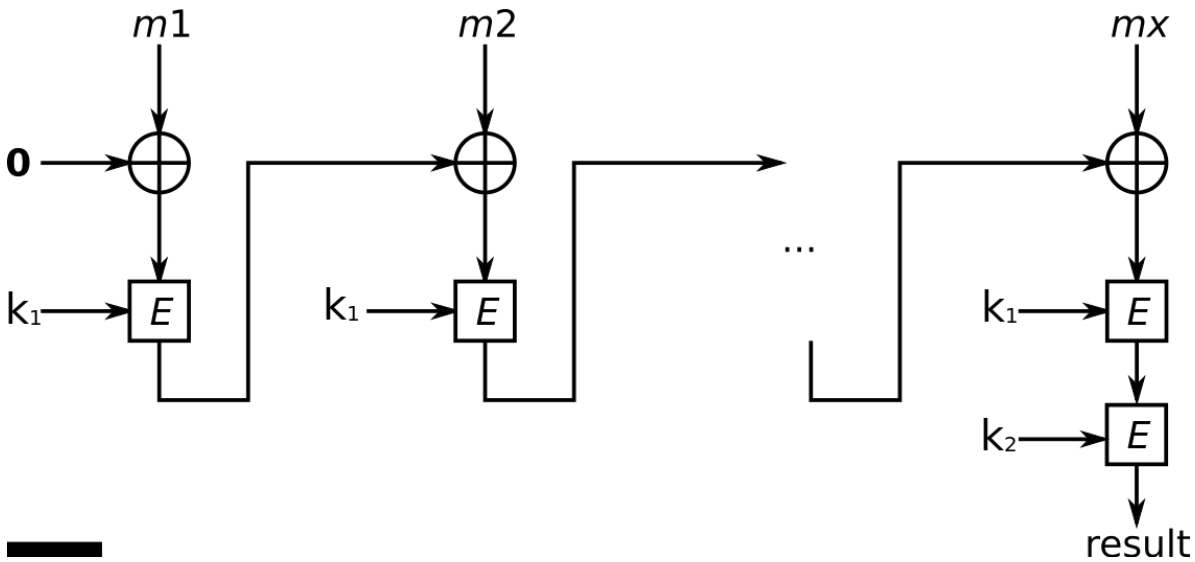
Raw CBC (variable-length messages)

Length Extension Attack

- Adversary gets pairs (m, t) and (m', t')
- Existential forgery attack: let $m'' = m \parallel m'$
- Replace first block of m' (m'_1) with $m'_1 \oplus t \implies$
  $m'' = m \parallel (m'_1 \oplus t) \parallel m'_2 \parallel ...m'_q$
- rawCBC$=E(k, m'') = E(k, m \parallel (m'_1 \oplus t) \parallel m'_2 \parallel ...m'_q) =$
  $E(k, t \oplus (m'_1 \oplus t) \parallel m'_2 \parallel ...m'_q) = E(k, m'_1 \parallel m'_2 \parallel ...m'_q) = t'$

CBC-MAC

- To fix this, encrypt the last step with $k_2 \neq k_1$



CBC-MAC vs CBC for encryption

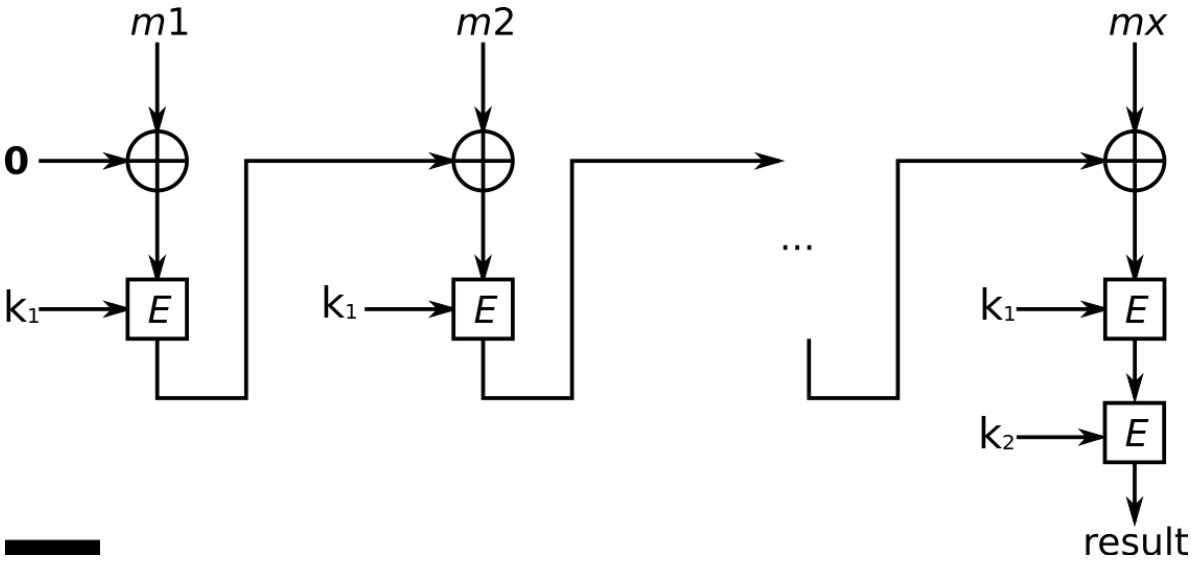
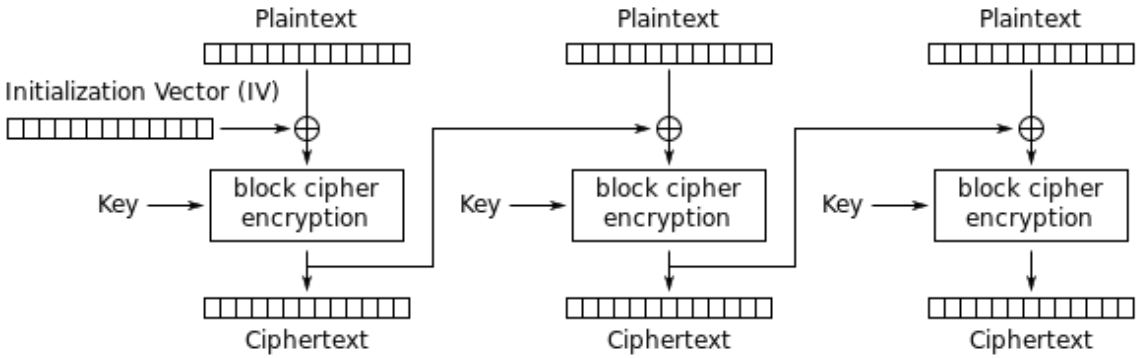


Figure: CBC-MAC Encrypt Last Block (ECBC)



Cipher Block Chaining (CBC) mode encryption

Why is IV set to 0?

- Say we use a randomly chosen IV for CBC-MAC
- As in the encryption cipher block chaining mode we send it in cleartext $\implies$ attacker knows IV
- Let $M_1 = P_1|P_2|...$ with IV_1 chosen randomly $\implies$ produces (M_1, T_1)
- First block of MAC is $E_k(IV_1 \oplus P_1)$
- Attacker produces $M_2 = P_1'|P_2|...$ and IV_1' such that $E_k(P_1' \oplus IV_1') = E_k(P_1 \oplus IV_1) \implies$ attacker gets tag T_1 for M_2
  - for every bit in $P_1' \neq$ bit in P_1 flip that bit in IV_1'
  - $P_1 \oplus IV_1 = P_1' \oplus IV_1'$

Hash Functions

A function H, $H : \{0,1\}^m \rightarrow \{0,1\}^n$ is a hash function if:

1. Efficient to compute H(x)
2. Computationally infeasible to find x from given H(x) = y
3. It is **collision resistant**

First 2 define a one-way function

Constructions from Hash Functions (just an idea)

- "small MAC to big MAC"
- I = (S, V) a MAC for short messages (e.g. AES)
- Use $H : M^{big} \rightarrow M$ to define $I^{big} = (S^{big}, V^{big})$
  - $S^{big}(k, m) = S(k, H(m))$
  - $V^{big}(k, m, t) = V(k, H(m), t)$
- If I is a secure MAC and H is collision resistant then I^{big} is a secure MAC.
- Example: S(k,m) = $AES_{2-block-cbc}$(k, SHA-256(m))
- CR is useful because if I have $m_0 \neq m_1$ and $H(m_0) = H(m_1)$
- Adversary asks for tag of m_0 and uses that as forgery for m_1

Digital Signatures

Alice and Bob do not share a key but each have a pair of (K_{pub}, K_{priv})

- Signing algorithm: $S(K_{priv}, m)$ =signature
- Verification algorithm: $V(K_{pub}, m$,signature$)$ = yes/no
- Example: Textbook RSA

Asymmetric Encryption vs Digital Signatures

- Both rely on a pair of public and private keys
- Alice wants to send to Bob an *encrypted* message
  - Alice encrypts the message with Bob's public key (anyone can send Bob messages)
  - Bob decrypts with his private key
- Alice wants to testify the message has not been tampered with and she is the source
  - Alice signs the message with Alice's private key = signature, attaches the signature to the message
  - Bob or anyone else can verify the signature using Alice's public key

Example GnuPGP

PGP stands for Pretty Good Privacy

- symmetric ciphers
- digital signatures
- DEMO...

Backup Slides

Pseudo Random Permutation (PRP)

- E is a PRP, $E : K \times X \rightarrow X$, defined over (K, X) such that
  - There exists "efficient" algorithm to evaluate $E(k, x)$
  - E is a one-to-one function
  - There exists "efficient" inversion algorithm $D(k, x)$
- Secure PRP
  - Let Perms[X] be the set of all **one-to-one** functions from X to X
  - $S_E = \{E(k, x)$ such that $k \in K$ and $x \in X\} \subseteq Perms[X]$
  - E is a secure PRP if it is indistinguishable from a random function

PRF and PRG

Any PRP is also a PRF

- A PRP is a PRF where X=Y and is efficiently invertible
- A PRP is sometimes called a block cipher