Cryptography
Lecture 5

Dr. Panagiotis Rizomiliotis

Agenda

- Encryption Security
- Hash security
- MAC security

Confidentiality

SYMMETRIC ENCRYPTION SCHEMES

Security

- *perfect security,*
  - *an information-theoretic notion introduced by* Shannon and showed by him to be met by the one-time pad scheme.
  - *regardless of the computing power available to the adversary, the ciphertext provides it no information about the plaintext beyond the a priori information it had prior to seeing the ciphertext*
  - it requires a key as long as the total amount of data encrypted
- *computational security*
  - The security will only hold with respect to adversaries of limited computing power.

Shannon's perfect secrecy definition

Let (E,D) be a cipher over (K,M,C)

(E,D) has perfect secrecy if $\forall\ m_0, m_1 \in M$ ($|m_0| = |m_1|$)

$\{ E(k,m_0) \} = \{ E(k,m_1) \}$ where $k \longleftarrow K$

Perfect Secrecy

Theorem

One time pad has perfect secrecy

- Proof: easily…

Theorem

Perfect secrecy implies that the size of the key K greater or equal to plaintext M

Impractical!

Quiz

Can a stream cipher have perfect secrecy?

Yes, if the PRG is really “secure”

No, there are no ciphers with perfect secrecy

Yes, every cipher has perfect secrecy

No, since the key is shorter than the message

Quiz

Can a stream cipher have perfect secrecy?

Yes, if the PRG is really “secure”

No, there are no ciphers with perfect secrecy

Yes, every cipher has perfect secrecy

[✓ No, since the key is shorter than the message]

Two times padding/ re-using the IV (attack 1)

- Let
  - $c1 = m1 \oplus k$
  - $c2 = m2 \oplus k$
- Eve eavesdrops c1,c2

<u>1st attack: Know Plaintext Attack</u>

Let m1 be a known plaintext to Eve. Then trivially:

$$c1 \oplus c2 \oplus m1 = m1 \oplus k \oplus m2 \oplus k \oplus m1 = m2$$

Example

- Alice (two times the same keystream)

m1: 0 1 1 0 1 1 1

k: 1 0 1 1 0 1 0 $\oplus$

---

c1: 1 1 0 1 1 0 1

m2: 1 0 0 1 0 1 1

k: 1 0 1 1 0 1 0 $\oplus$

---

c2: 0 0 1 0 0 0 1

Example

- Eve

m1: 0 1 1 0 1 1 1

c2: 0 0 1 0 0 0 1

$\oplus$

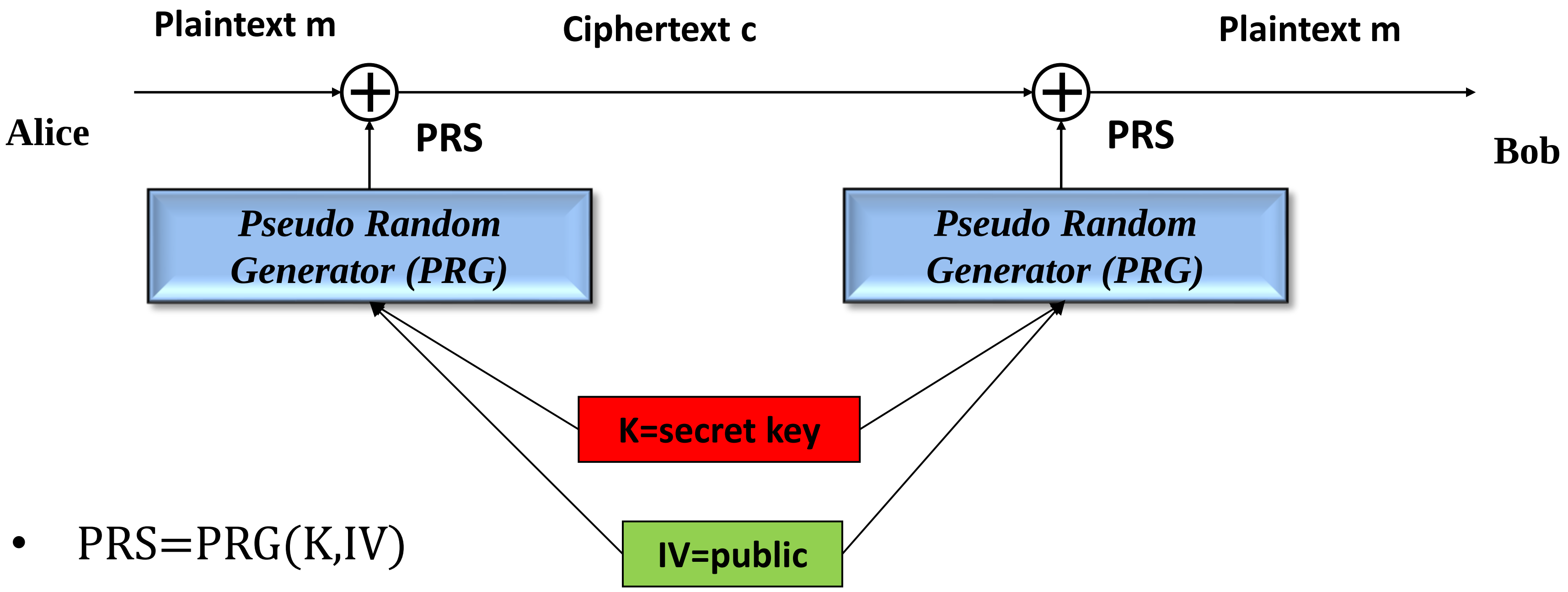
c1: 1 1 0 1 1 0 1

m2: 1 0 0 1 0 1 1

Stream Ciphers

- One time padding is also a stream cipher requirement
- Remember that:
  - The generator PRG produces a pseudorandom sequence PRS
    - $\mathrm{PRS}=\mathrm{PRG}(K,IV)$
    - c = m $\oplus$ PRS

Stream Ciphers (synchronous)



- PRS=PRG(K,IV)
- $c = m \oplus \text{PRS}$

Stream Ciphers

- When the same key/IV pair is used the generator produces the same PRS
- Thus, we have
  - c1 = m1 ⊕ PRS
  - c2 =m2 ⊕ PRS
- The same attach. The IV must never repeat for the same key.

Example

- Alice (two times the same IV)

m1: 0 1 1 0 1 1 1

$\oplus$

PRS: 1 0 1 1 0 1 0

c1: 1 1 0 1 1 0 1

m2: 1 0 0 1 0 1 1

$\oplus$

PRS: 1 0 1 1 0 1 0

c2: 0 0 1 0 0 0 1

Example

- Eve

m1:	0	1	1	0	1	1	1	
c2:	0	0	1	0	0	0	1	
c1:	1	1	0	1	1	0	1	⊕
m2:	1	0	0	1	0	1	1	

Two times padding/ re-using the IV (attack 2)

- Let
  - $c1 = m1 \oplus k$
  - $c2 = m2 \oplus k$
- Eve eavesdrops c1,c2

<u>2^{nd} attack: Known Plaintext Statistics</u>

- Eve computes:

  $c=c1 \oplus c2=m1 \oplus k \oplus m2 \oplus k =m1\oplus m2$

  – Eve combines the (most probable) values of m1 and m2 until she produces c

  – It is an efficient way to find candidate pairs (m1,m2)

Example

- Alice (two times the same keystream)

$$\begin{array}{l l} m1: & 0\ 1\ 1\ 0\ 1\ 1\ 1 \\ k: & 1\ 0\ 1\ 1\ 0\ 1\ 0 \\ \hline c1: & 1\ 1\ 0\ 1\ 1\ 0\ 1 \end{array} \oplus$$

$$\begin{array}{l l} m2: & 1\ 0\ 0\ 1\ 0\ 1\ 1 \\ k: & 1\ 0\ 1\ 1\ 0\ 1\ 0 \\ \hline c2: & 0\ 0\ 1\ 0\ 0\ 0\ 1 \end{array} \oplus$$

Example

- Eve

c1: 1 1 0 1 1 0 1

c2: 0 0 1 0 0 0 1 ⊕

c: 1 1 1 1 1 0 0

Trivial leakage:

When the bits of c are zero then the corresponding bits of m1 and m2 are the same

Example

- Eve

$$\begin{array}{ll} c1: & 1\ 1\ 0\ 1\ 1\ 0\ 1 \\ c2: & 0\ 0\ 1\ 0\ 0\ 0\ 1 \quad \oplus \\ \hline c: & 1\ 1\ 1\ 1\ 1\ 0\ 0 \end{array}$$

Scenario:

Let any m used by Alice be of the form

m=X||D

where X is one of {100, 000,011}.

Then, m1 = X1|D1 and m2= X2|D2

We have that for the different possible X:

$$100 \oplus 000 = 100$$
$$100 \oplus 011 = 111$$
$$000 \oplus 011 = 011$$

Since the first 3 bits of c are 111, then the first 3 of m1, m2 are {100,011}/ we don't know which is which.

We can improve the attack we more ciphertexts

WEP - SECURITY

Wired Equivalent Privacy (WEP)

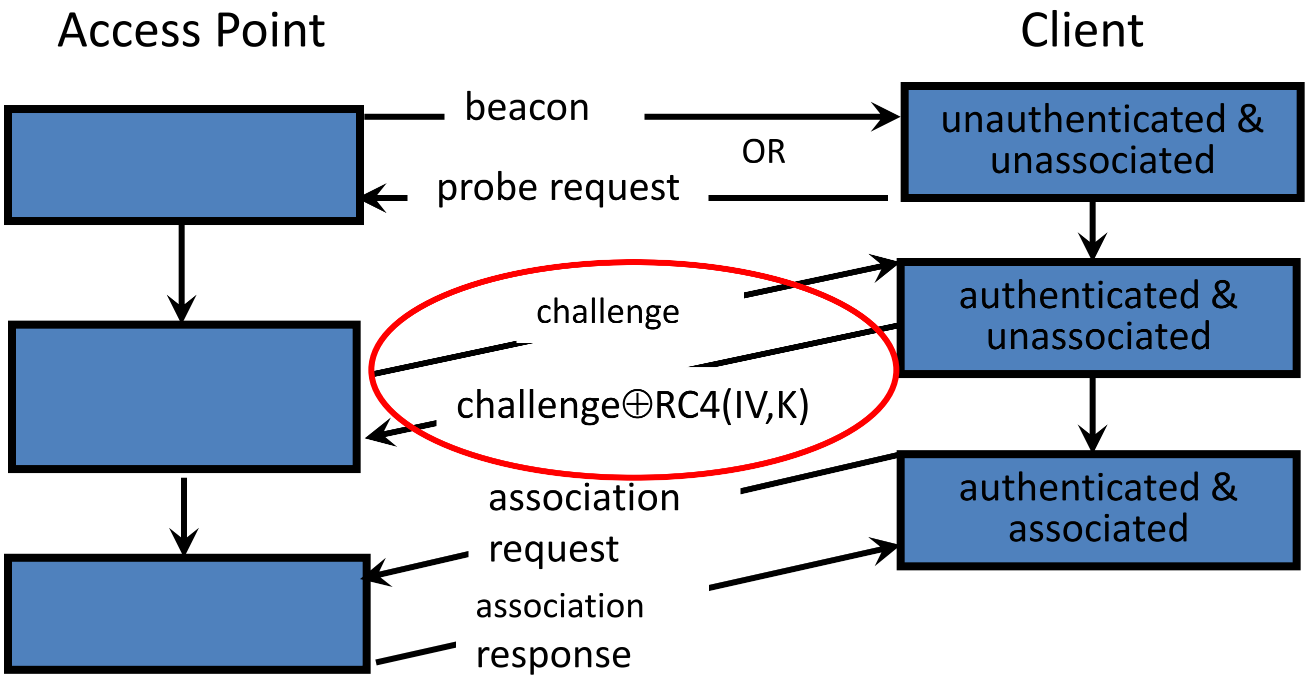
- WEP - Part of original 802.11 specification published in 1999.
- Confidentiality
  - Uses RC4 Stream cipher
  - Has static 40-bit base key (common for all the clients)
  - A 64-bit per-packet key
  - A 24-bit Initialization Vector (IV)
- Integrity
  - Uses Integrity Check Value (ICV) to verify integrity
  - No key!!

Characteristics - notes

- Stateless protocol
  - Mobile stations and access points are not required to keep past state
- Encrypted CRC-32 used as integrity check
  - Fine for random errors, but not deliberate ones
  - Linear
    - CRC(X+Y) = CRC(X)+CRC(Y)
- RC4 keystream should not be reused
  - One-time pad

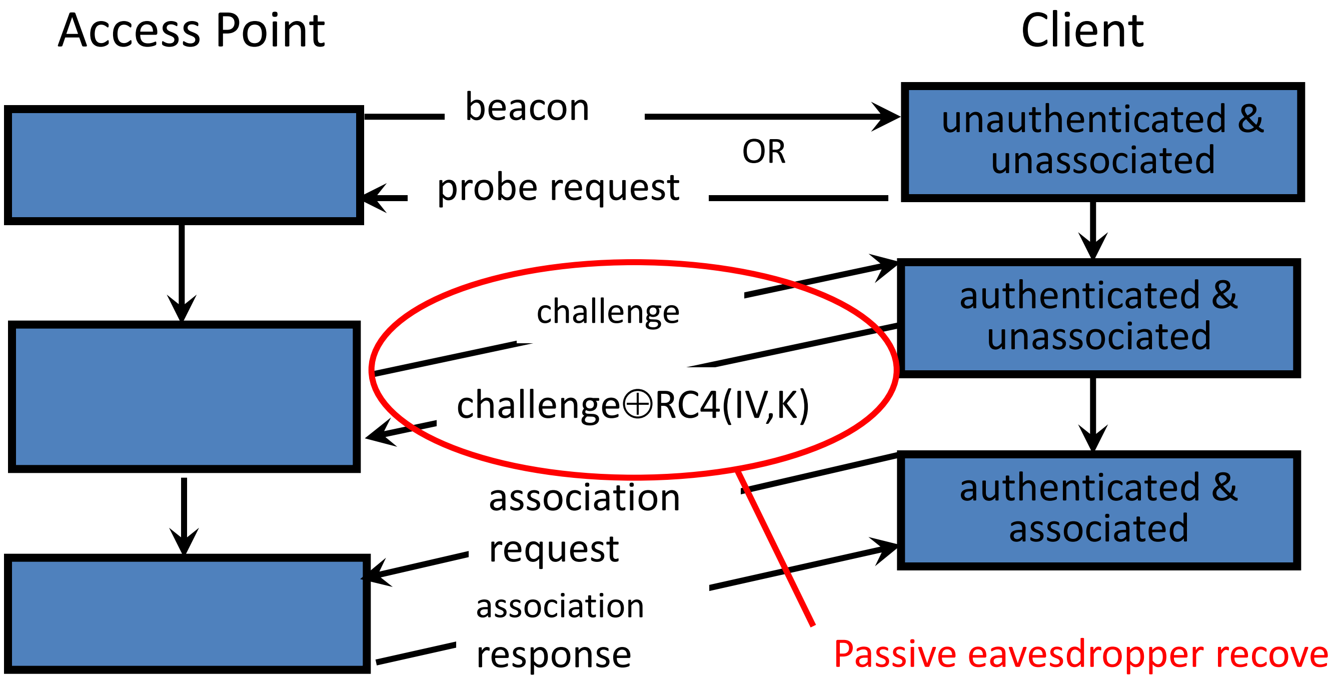
Shared-Key Authentication

Prior to communicating data, access point may require client to authenticate



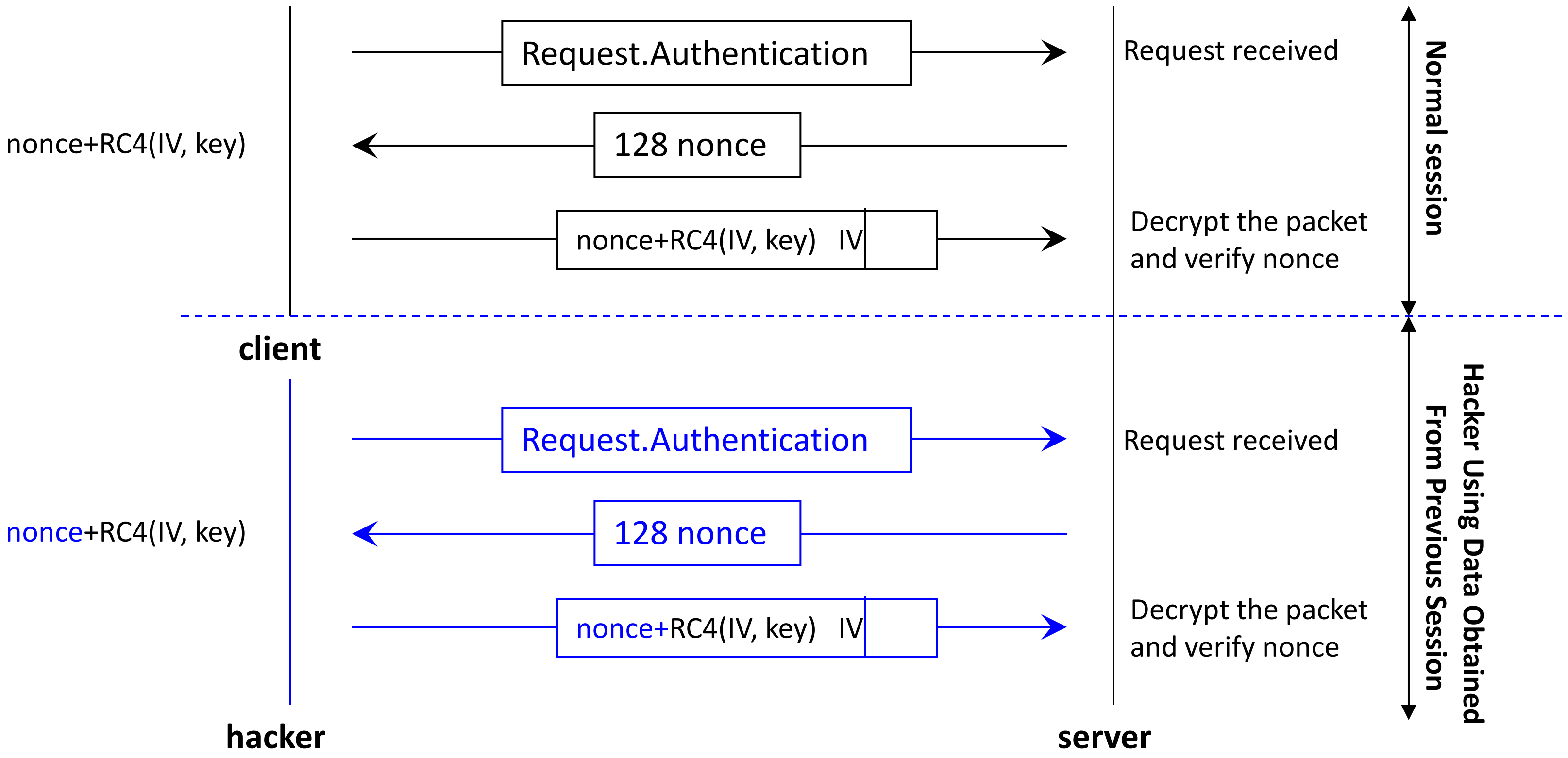
Shared-Key Authentication

Prior to communicating data, access point may require client to authenticate



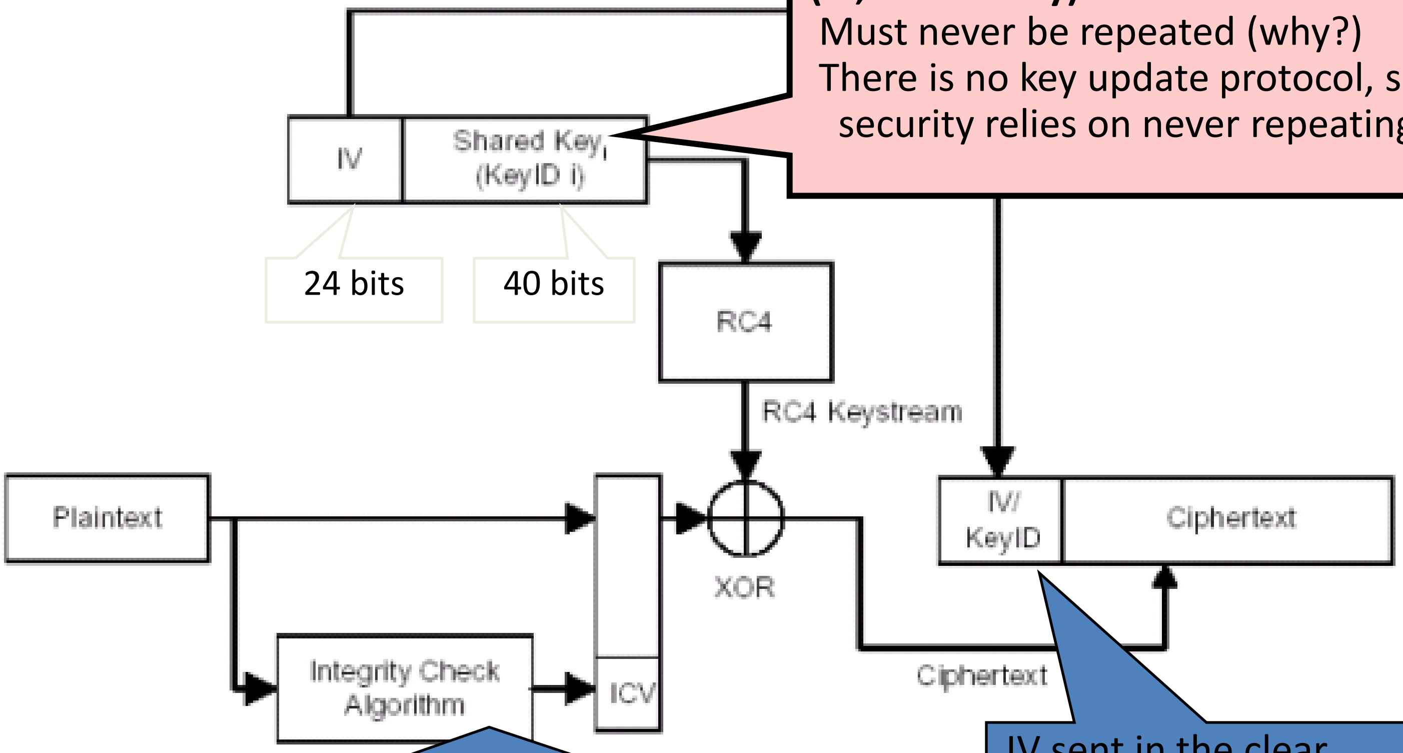
Passive eavesdropper recovers RC4(IV,K), can respond to any subsequent challenge <u>without</u> knowing K

Attack on Access Control



- It is possible to get authenticated without knowing the secret key! (shown in blue)
- We only need a plaintext, ciphertext pair of a legitimate authentication. (shown in black)

How WEP “Privacy” Works



(IV, shared key) used as RC4 seed

Must never be repeated (why?)

There is no key update protocol, so security relies on never repeating IV

IV sent in the clear

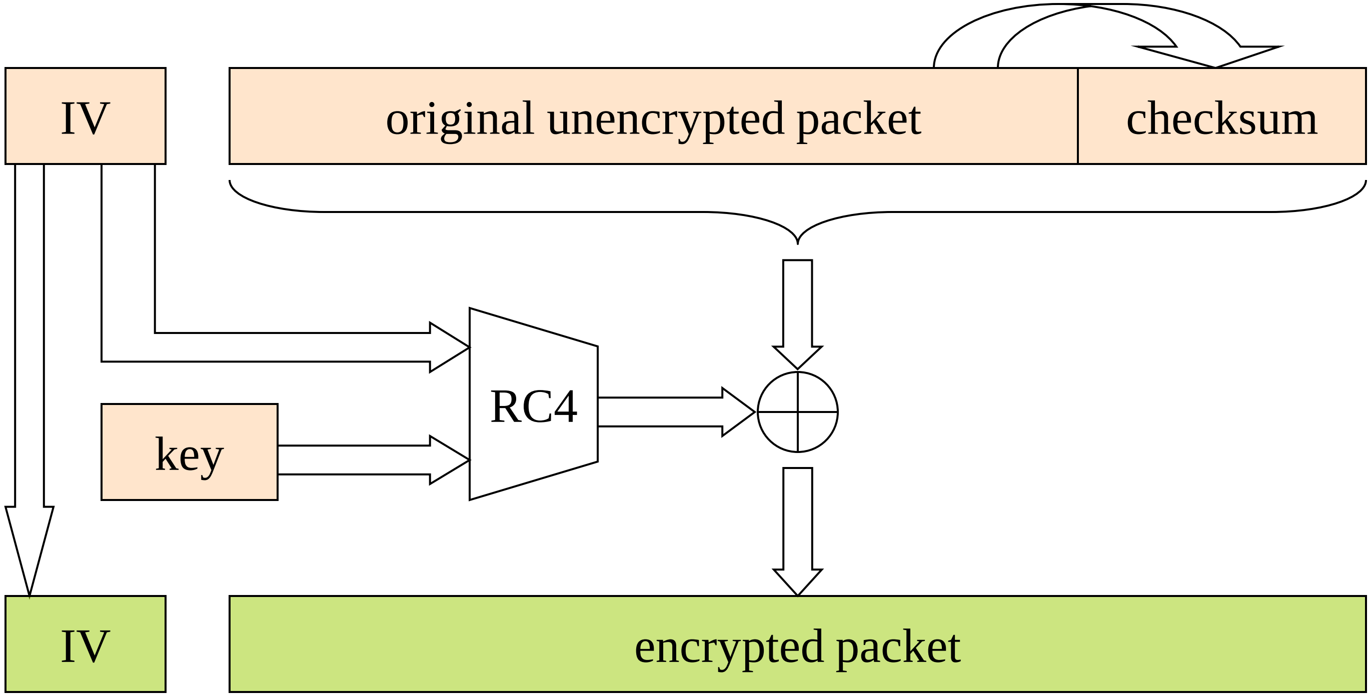
Worse: <u>changing IV with each packet is optional!</u>

CRC-32 checksum is linear in ⊕:

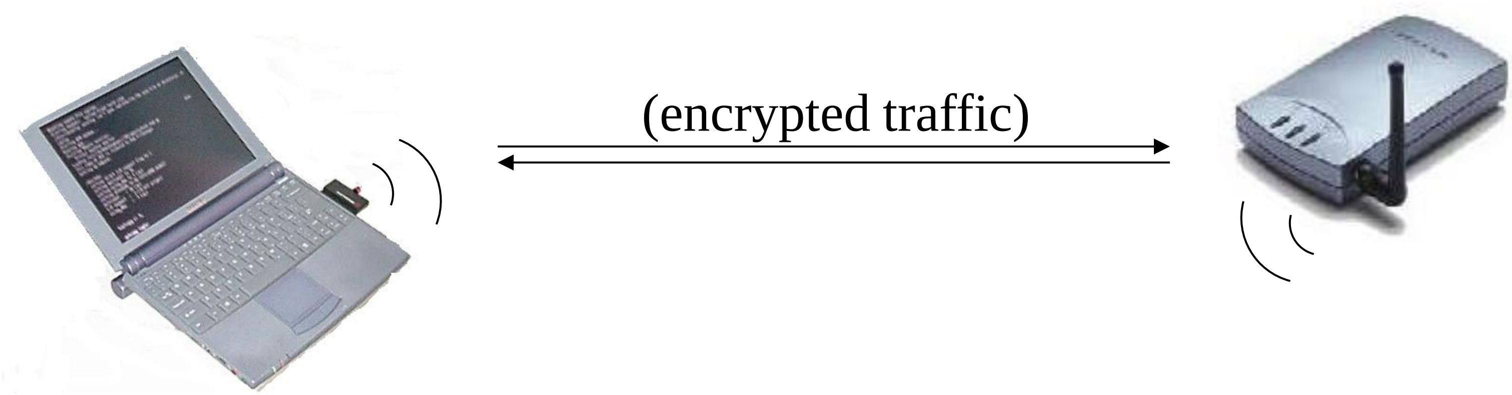
if attacker flips some plaintext bits, he knows which bits of CRC to flip to produce the <u>same checksum</u>

no integrity!

WEP

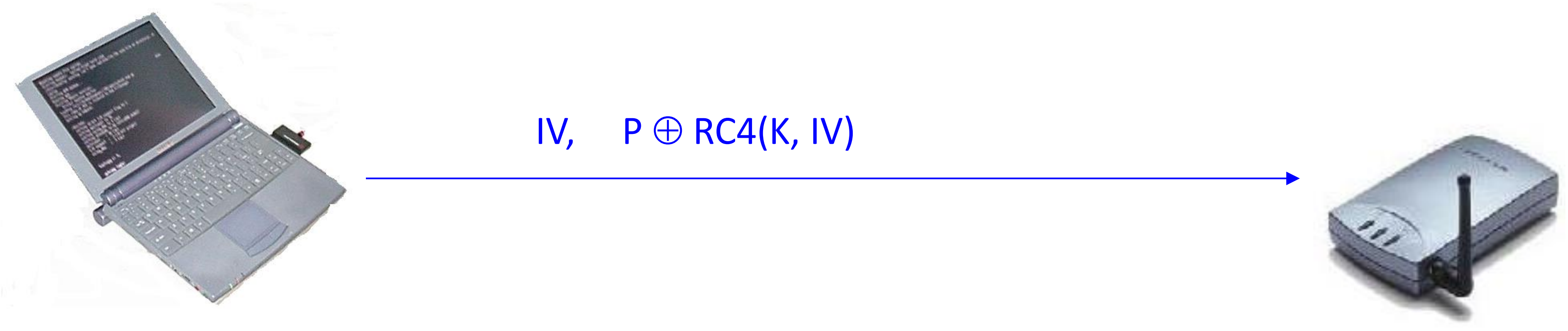

IV
original unencrypted packet
checksum
RC4
key
IV
encrypted packet

WEP



- Share a single cryptographic key among all devices
- Encrypt all packets sent over the air, using the shared key
- Use a checksum to prevent injection of spoofed packets

WEP - A Little More Detail



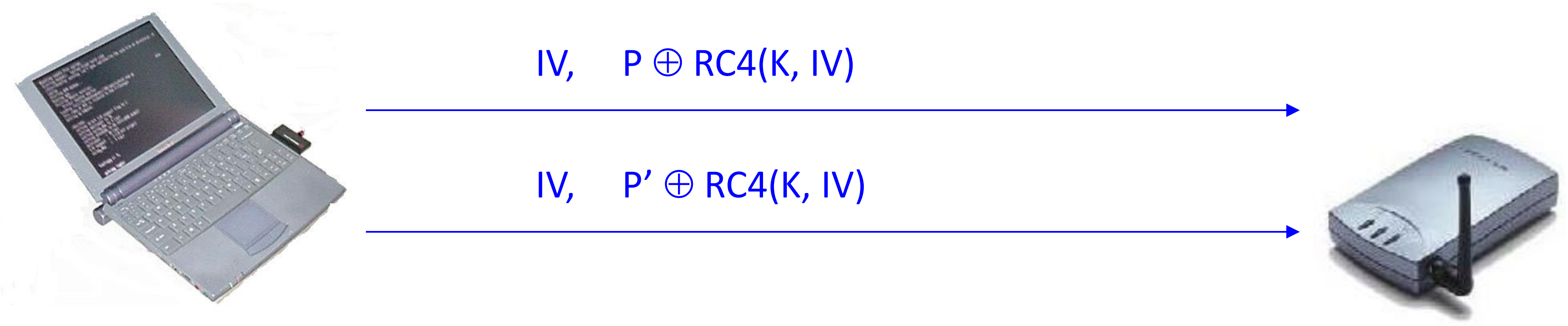
IV, $P \oplus RC4(K, IV)$

- WEP uses the RC4 stream cipher to encrypt a TCP/IP packet (P) by xor-ing it with keystream (RC4(K, IV))

A Property of RC4

- Keystream leaks, under known-plaintext attack
  - Suppose we intercept a ciphertext C, and suppose we can guess the corresponding plaintext P
  - Let $Z = RC4(K, IV)$ be the RC4 keystream
  - Since $C = P \oplus Z$, we can derive the RC4 keystream Z by $P \oplus C = P \oplus (P \oplus Z) = Z$
- This is not a problem ... unless keystream is reused!

A Risk of Keystream Reuse



IV, $P \oplus \mathrm{RC4}(K, IV)$

IV, $P' \oplus \mathrm{RC4}(K, IV)$

- If IV’s repeat, confidentiality is at risk
  - If we send two ciphertexts (C, C’) using the same IV, then the xor of plaintexts leaks ($P \oplus P' = C \oplus C'$), which might reveal both plaintexts

➤ Lesson: If RC4 isn’t used carefully, it becomes insecure

A note on IVs

- What if random IVs were used?
- IV space – 2^{24} possibilities
- Collision after 4000 packets
- Rough estimate: a busy AP sends 1000 packets/sec
- Collision every 4s!
- Even with counting IV (best case), rollover every few hours

So..

- If we have 2^{24} known plaintexts, can decrypt every packet!!!!

Attack #1: Keystream Reuse

- WEP didn’t use RC4 carefully
- The problem: IV’s frequently repeat
  - The IV is often a counter that starts at zero
  - Hence, rebooting causes IV reuse
  - Also, there are only 16 million possible IV’s, so after intercepting enough packets, there are sure to be repeats

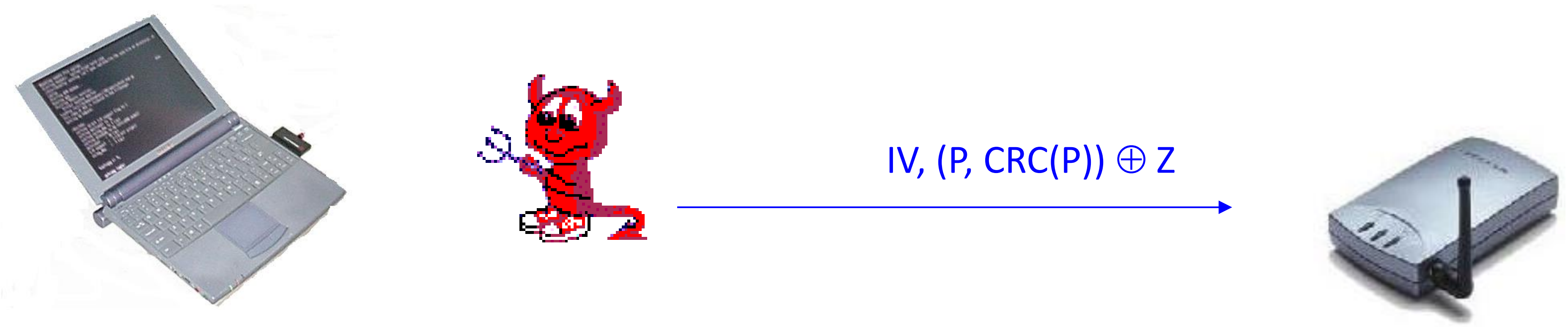
➤ Attackers can eavesdrop on 802.11 traffic

  - An eavesdropper can decrypt intercepted ciphertexts even without knowing the key

Attack #2: Spoofed Packets

- Attackers can inject forged 802.11 traffic
  - Learn RC4(K, IV) using previous attack
  - Since the checksum is unkeyed, you can then create valid ciphertexts that will be accepted by the receiver

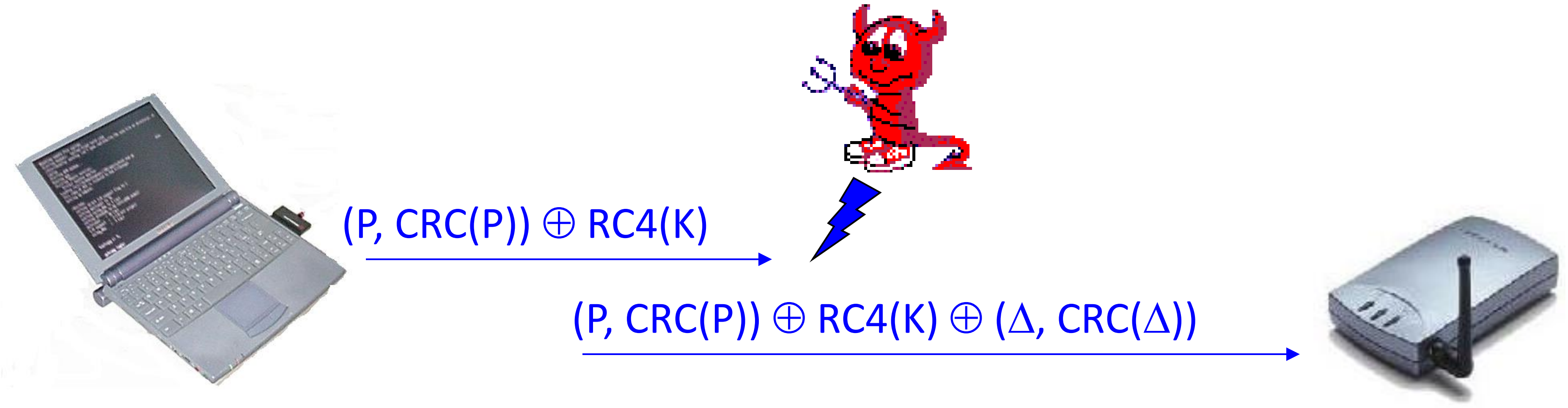
Attack #2: Spoofed Packets



- Attackers can inject forged 802.11 traffic
  - Learn Z = RC4(K, IV) using previous attack
  - Since the CRC checksum is unkeyed, you can then create valid ciphertexts that will be accepted by the receiver

- Attackers can bypass 802.11b access control
  - All computers attached to wireless net are exposed

Attack #3: Packet Modification



$(P, \mathrm{CRC}(P)) \oplus \mathrm{RC4}(K)$

$(P, \mathrm{CRC}(P)) \oplus \mathrm{RC4}(K) \oplus (\Delta, \mathrm{CRC}(\Delta))$

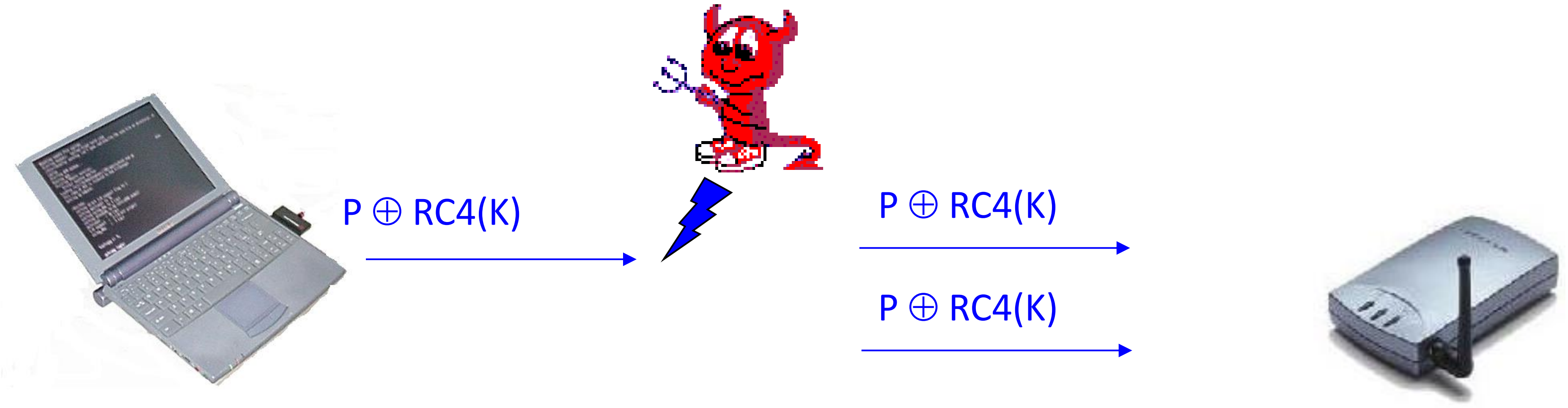
- CRC is linear

  $\Rightarrow \mathrm{CRC}(P \oplus \Delta) = \mathrm{CRC}(P) \oplus \mathrm{CRC}(\Delta)$

  $\Rightarrow$ the modified packet ($P \oplus \Delta$) has a valid checksum

➤ Attacker can tamper with packet (P) without breaking RC4

Attack #4: Replay Attacks



P ⊕ RC4(K)

P ⊕ RC4(K)

P ⊕ RC4(K)

- Attacker can replay plaintext (P) without breaking RC4
- Stateless!!

Attacks

- Andrea Bittau, Mark Handley, and Joshua Lackey. The final nail in WEP's cofin. In IEEE Symposium on Security and Privacy, pages 386-400. IEEE Computer Society, 2006.
- 2. Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting mobile communications: the insecurity of 802.11. In ACM MobiCom 2001, pages 180-189. ACM Press, 2001.
- 3. Rafik Chaabouni. Break WEP faster with statistical analysis. Technical report, EPFL, LASEC, June 2006. http://lasecwww.epfl.ch/pub/lasec/doc/cha06.pdf.
- Scott R. Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. In Serge Vaudenay and Amr M. Youssef, editors, Selected Areas in Cryptography 2001, volume 2259 of Lecture Notes in Computer Science, pages 1-24. Springer, 2001.
- Andreas Klein. Attacks on the RC4 stream cipher. submitted to Designs, Codes and Cryptography, 2007.
- KoreK. chopchop (experimental WEP attacks). http://www.netstumbler.org/showthread.php?t=12489, 2004.
- KoreK. Next generation of WEP attacks? http://www.netstumbler.org/showpost.php?p=93942&postcount=35, 2004.
- Subhamoy Maitra and Goutam Paul. Many keystream bytes of RC4 leak secret key information. Cryptology ePrint Archive, Report 2007/261, 2007. http://eprint.iacr.org/.
- Toshihiro Ohigashi, Hidenori Kuwakado, and Masakatu Morii. A key recovery attack on WEP with less packets. to be published, 2007.
- Yuko Ozasa, Yoshiaki Fujikawa, Toshihiro Ohigashi, Hidenori Kuwakado, and Masakatu Morii. A study on the Tews, Weinmann, Pyshkin attack against WEP. In IEICE Tech. Rep., volume 107 of ISEC2007-47, pages 17{21, Hokkaido, July 2007. Thu, Jul 19, 2007 - Fri, Jul 20 : Future University-Hakodate (ISEC, SITE, IPSJ-CSEC).
- Adam Stubblefield, John Ioannidis, and Aviel D. Rubin. A key recovery attack on the 802.11b wired equivalent privacy protocol (WEP). ACM Transactions on Information and System Security, 7(2):319{332, May 2004.
- Erik Tews, Ralf-Philipp Weinmann, and Andrei Pyshkin, ‘Breaking 104 bit WEP in less than 60 seconds’, Cryptology ePrint Archive, Report 2007/120, 2007. http://eprint.iacr.org/.

Recall Shannon's perfect secrecy

Let (E,D) be a cipher over (K,M,C)

(E,D) has perfect secrecy if $\forall\, m_0, m_1 \in M \quad (\ |m_0| = |m_1| \ )$

$$\{ E(k,m_0) \} \quad = \quad \{ E(k,m_1) \} \qquad \text{where} \quad k \longleftarrow K$$

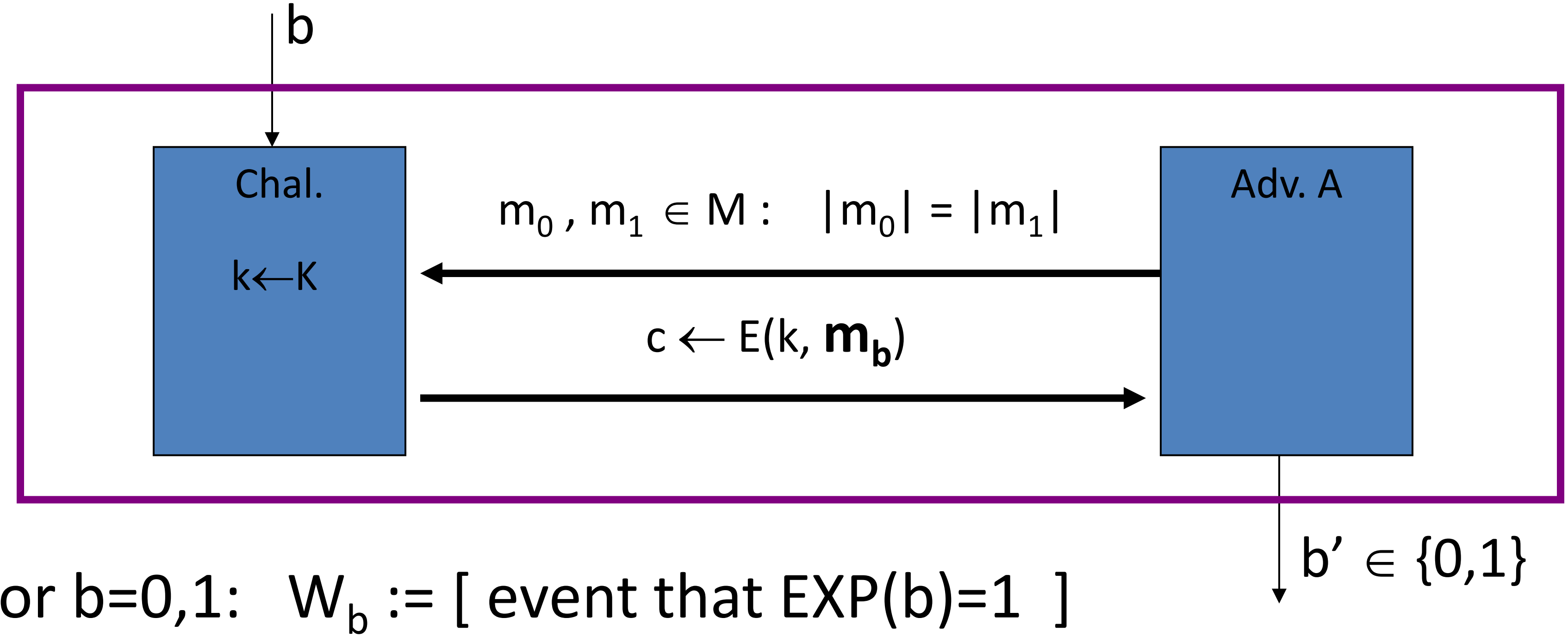
(E,D) has perfect secrecy if $\forall\, m_0, m_1 \in M \quad (\ |m_0| = |m_1| \ )$

$$\{ E(k,m_0) \} \ \approx_{\mathbf{p}} \ \{ E(k,m_1) \} \qquad \text{where} \quad k \longleftarrow K$$

… but also need adversary to exhibit $m_0, m_1 \in M$ explicitly

Semantic Security/ IND-CPA

For   b=0,1   define experiments EXP(0) and EXP(1) as:


b
Chal.
k←K
Adv. A
$m_0 , m_1 \in M : \quad |m_0| = |m_1|$
$c \leftarrow E(k, \mathbf{m_b})$
$b' \in \{0,1\}$

for b=0,1: W_b := [event that EXP(b)=1]

$$\text{Adv}_{SS}[A,E] := \big|\ \Pr[\ W_0\ ] - \Pr[\ W_1\ ]\ \big| \quad \in [0,1]$$

Semantic Security (one-time key)

Def: $\mathbb{E}$ is **semantically secure** if for all efficient A

$\text{Adv}_{\text{SS}}[A,\mathbb{E}]$ is negligible.

- Indistinguishability under chosen-plaintext attack (IND-CPA)

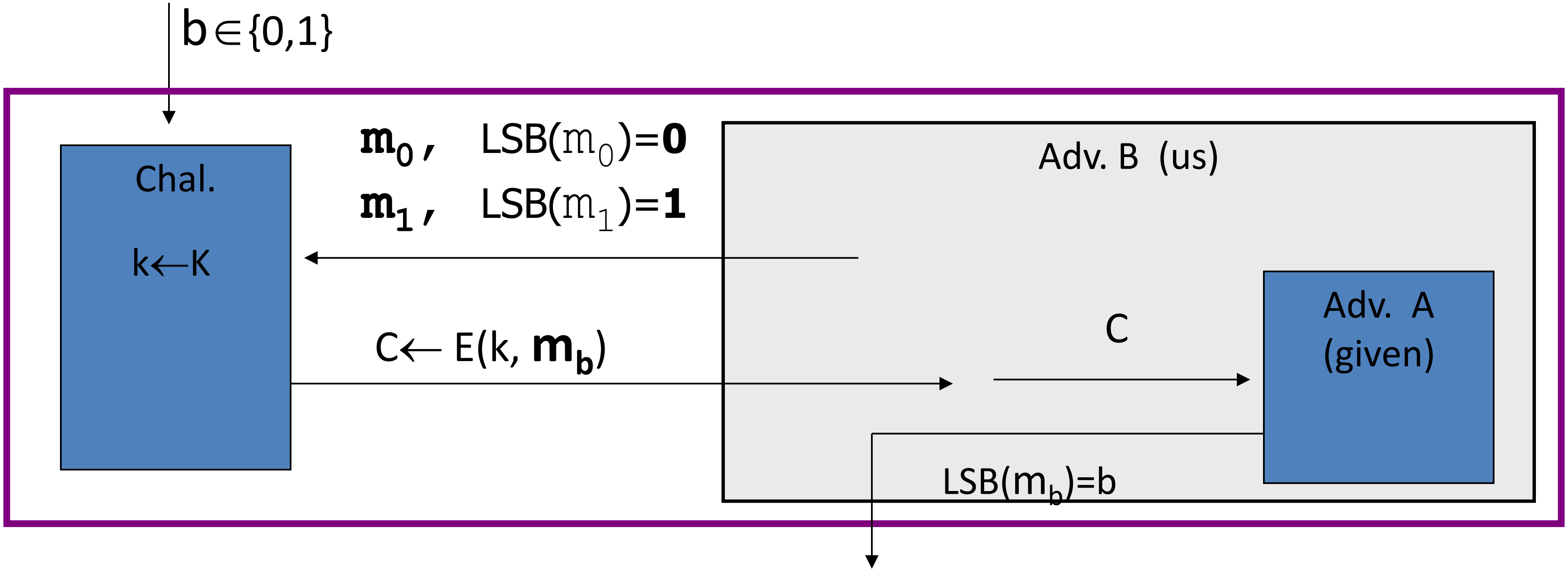
⇒ for all explicit m_0 , $m_1 \in M$:

$$\{ E(k,m_0) \} \approx_p \{ E(k,m_1) \}$$

Example 1

Suppose efficient A can always deduce LSB of PT from CT.

⇒ $\mathbb{E}$ = (E,D) is not semantically secure.



$b \in \{0,1\}$
Chal.
$k \leftarrow K$
$\mathbf{m_0}$, LSB($\mathtt{m}_0$)=**0**
$\mathbf{m_1}$, LSB($\mathtt{m}_1$)=**1**
$C \leftarrow E(k, \mathbf{m_b})$
Adv. B (us)
C
Adv. A (given)
LSB(m_b)=b

Then Adv$_{SS}$[B, $\mathbb{E}$] = $\Big|$ Pr[**EXP(0)**=1] − Pr[**EXP(1)**=1] $\Big|$ = |0 – 1| = 1

Example 1

- When algorithm A works with probability p (not certain) then the attack is the same, only the advantage changes.
- Example, p = 0.8

Then $\mathrm{Adv}_{SS}[B, \mathbb{E}] = \left| 0{,}8 - 0{,}2 \right| = |0{,}6| = 0{,}6$

Example 2

- Έστω ότι ο (E,D) είναι ένας semantically secure cipher όπου ο χώρος του μηνύματος και του ciphertext είναι $\{0,1\}^n$. Eϊναι το ακόλουθο σχήμα κρυπτογράφησης είναι semantically secure?

$$E'(k,m)=E(k,m)\|\|(LSB(m)\oplus MSB(m))$$

Proof (sketch):

1. Use the definition to evaluate the scheme
2. We are looking for two messages such that LSB(m)$\oplus$ MSB(m) gives different output
3. Two such messages, for any m, are
   - m0 = (0||m||0)
   - m1 = (1||m||0)

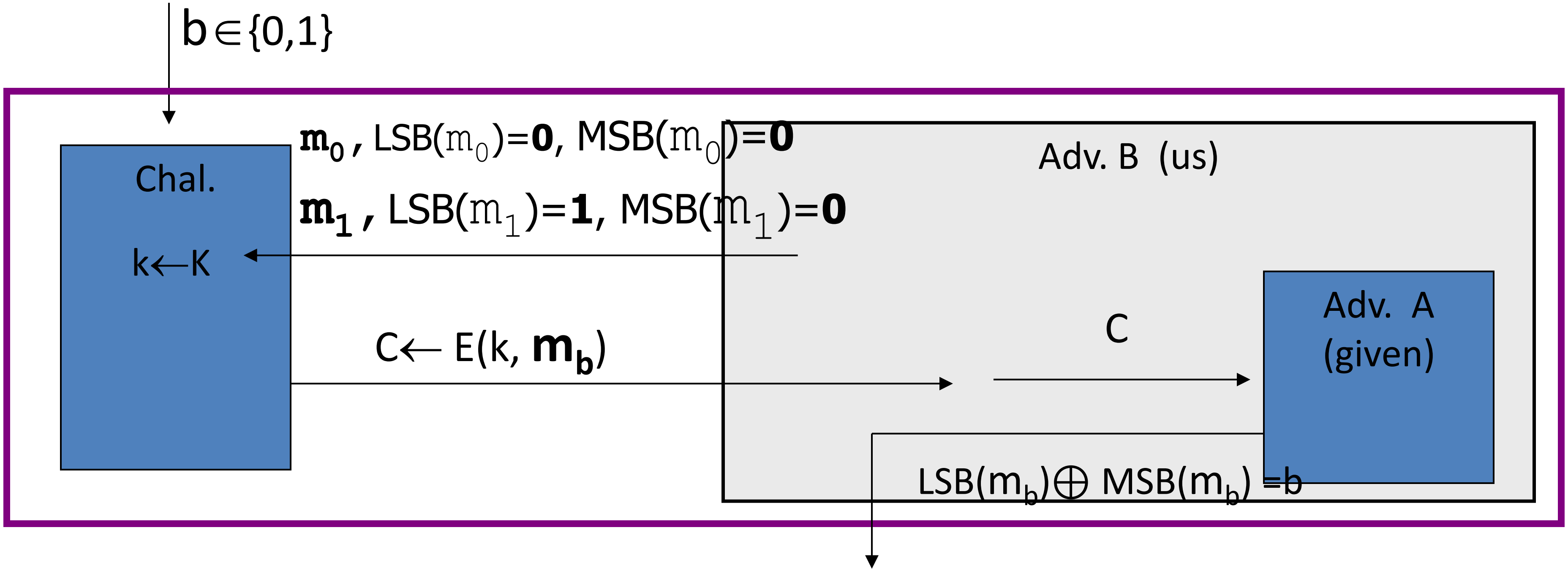
   (there are also other choices of course.)

- The advantage is 1.

Example 2

Suppose efficient A can always deduce LSB$\oplus$ LSB of PT from CT.

$\Rightarrow$ $\mathbb{E}$ = (E,D) is not semantically secure.



$b \in \{0,1\}$

Chal.

$k \leftarrow K$

$\mathbf{m_0}$, LSB(m_0)=**0**, MSB(m_0)=**0**

$\mathbf{m_1}$, LSB(m_1)=**1**, MSB(m_1)=**0**

$C \leftarrow E(k, \mathbf{m_b})$

Adv. B (us)

C

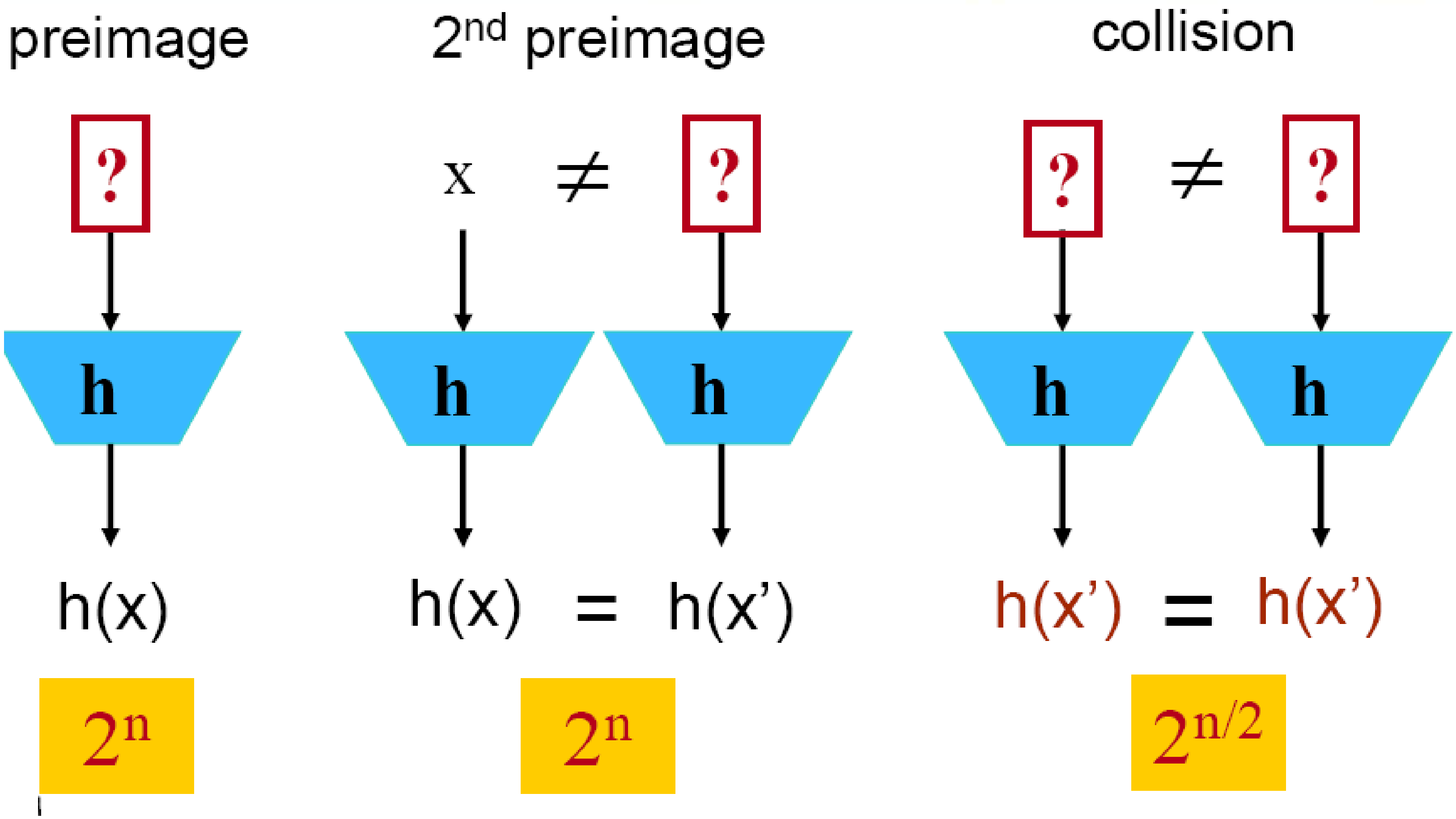
Adv. A (given)

LSB(m_b)$\oplus$ MSB(m_b) =b

Then Adv$_{SS}$[B, $\mathbb{E}$] = $\Big|$ Pr[**EXP(0)**=1] – Pr[**EXP(1)**=1] $\Big|$ = |0 – 1| = 1

HASH FUNCTIONS

Cryptographic properties



preimage

h(x)

2^n

2nd preimage

h(x) = h(x')

2^n

collision

h(x') = h(x')

$2^{n/2}$

Exercise 1

Let $H:\{0,1\}^* \rightarrow T$ be a collision resistant hash function. Is the following hash function collision resistant?

$$H'(m)=H(m)\oplus H(m\oplus 1^{|m|})$$

Where $|m|$ is length of m and 1^x is a string of x 1's.

For instance, let m=10101. Then, $|m|=5$ and $1^{|m|}=1^5=11111$

Proof

As a general rule of thumb, when you a hash function is not secure, you need to provide the attack. Otherwise, you must craft a proof.

Clearly, it is easy to show that any two message m, m' such that $m'= m\oplus 1^{|m|}$, they have that same hash value:

$H'(m')=H(m')\oplus H(m'\oplus 1^{|m'|})= H(m\oplus 1^{|m|})\oplus H(m\oplus 1^{|m|} \oplus 1^{|m|})=$

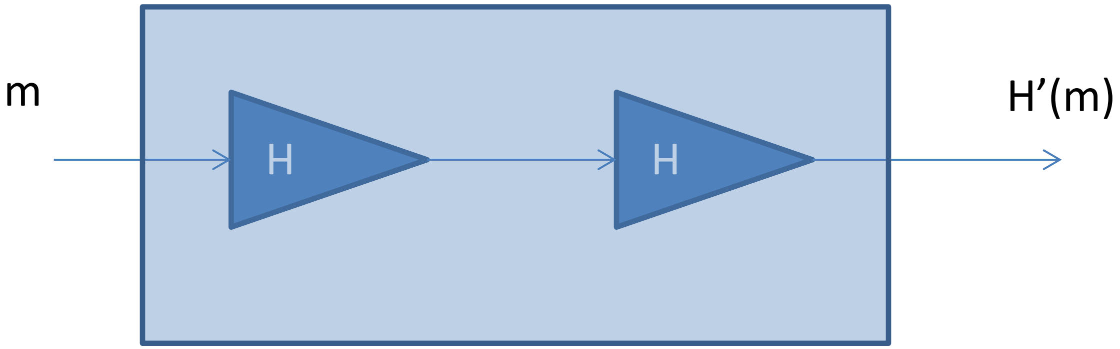
$= H(m\oplus 1^{|m|})\oplus H(m)=H'(m)$

Thus, it is not collision resistant.

Exercise 2

1. Let $H: \{0,1\}^* \rightarrow \{0,1\}^n$ be a collision resistant hash function. Is the following hash function collision resistant?

$$H'(m)=H(H(m))$$



Proof (sketch)

- Let H’(m)=H(H(m)) and let’s assume that H’(m) is not collision resistant. Thus, there is a polynomial algorithm A that can compute a pair of messages m1 and m2, more efficiently than O($2^{n/2}$), such that:

H’(m1)=H’(m2)

Thus, it holds H(H(m1))=H(H(m2)). We distinguish two cases:

1. H(m1)=H(m2). Then, the algorithm A can compute collisions for H(m), more efficiently than O($2^{n/2}$). This is a contradiction.
2. H(m1)≠H(m2). Then, the messages y1=H(m1) and y2=H(m2)

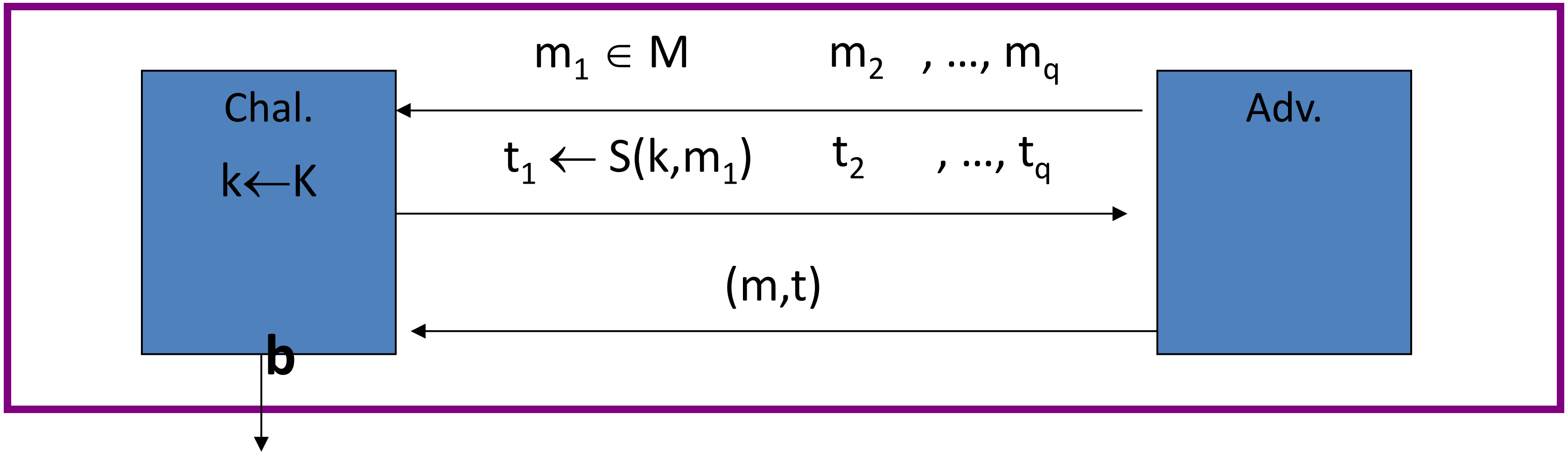
H(H(m1))=H(H(m2)) <=> H(y1)=H(y2)

are collisions for H(m). That is that, the algorithm A can compute collisions for H(m), more efficiently than O($2^{n/2}$). This is a contradiction.

MAC SECURITY

Strong Unforgeability under Chosen Message Attack (SUF-CMA)

- For a MAC I=(S,V) and adv. A define a MAC game as:

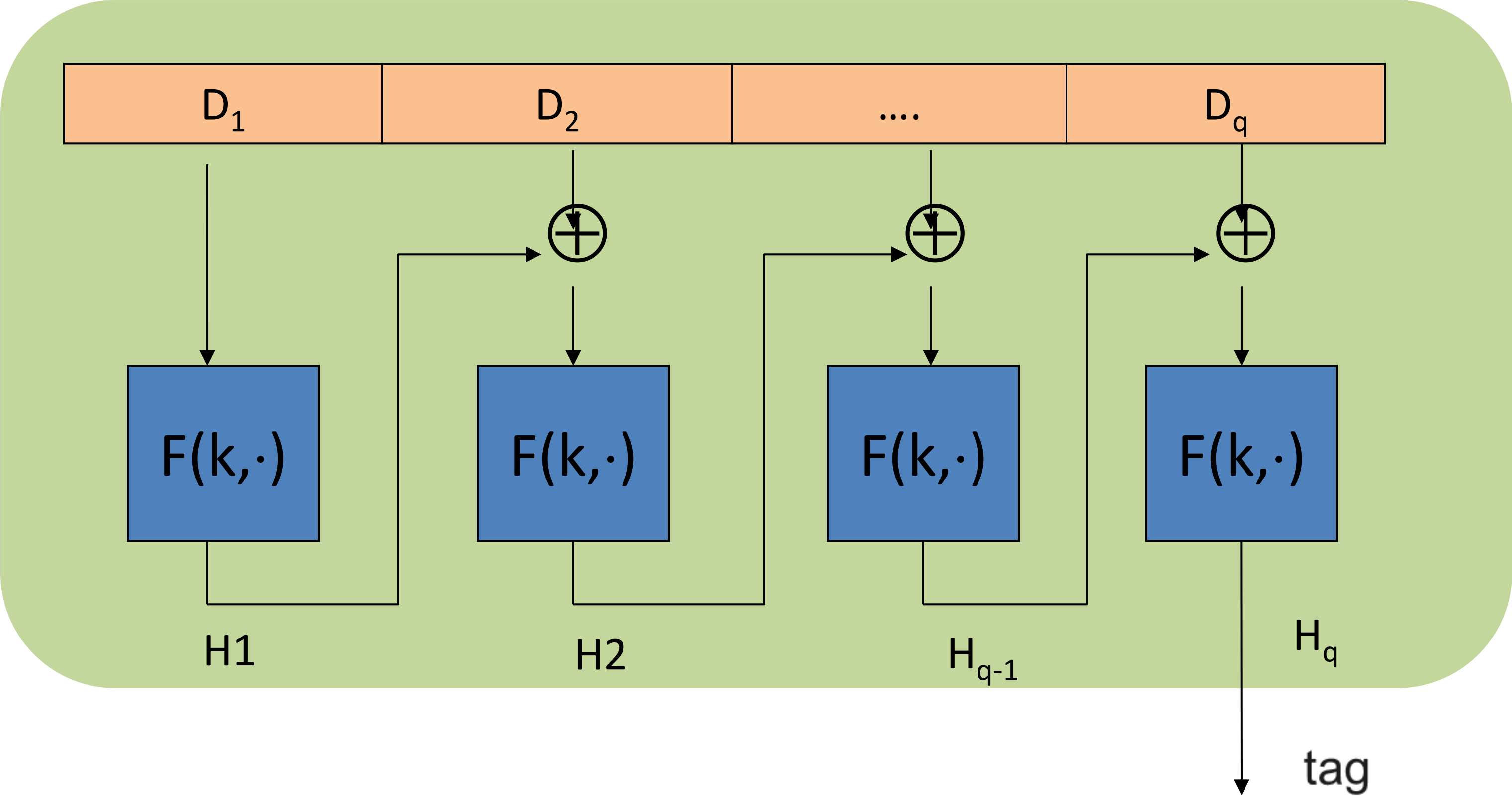

Chal.
$k \leftarrow K$
$m_1 \in M$ $\quad m_2$, …, m_q
$t_1 \leftarrow S(k,m_1)$ $\quad t_2$, …, t_q
(m,t)
Adv.
b

$$\begin{cases} \mathbf{b}=1 & \text{if } V(k,m,t) = \text{`yes'} \quad \text{and } (m,t) \notin \{ (m_1,t_1), \ldots, (m_q,t_q) \} \\ \mathbf{b}=0 & \text{otherwise} \end{cases}$$

Def: I=(S,V) is a **secure MAC** if for all "efficient" A:

$\mathrm{Adv}_{\mathrm{MAC}}[A,I]$ = Pr[Chal. outputs 1] is "negligible."

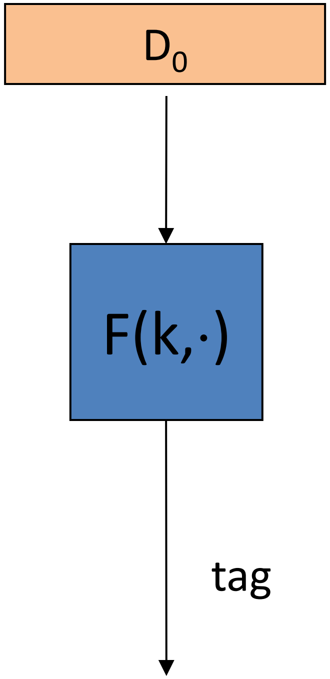
(RAW) CBC-MAC security



(RAW) CBC-MAC security

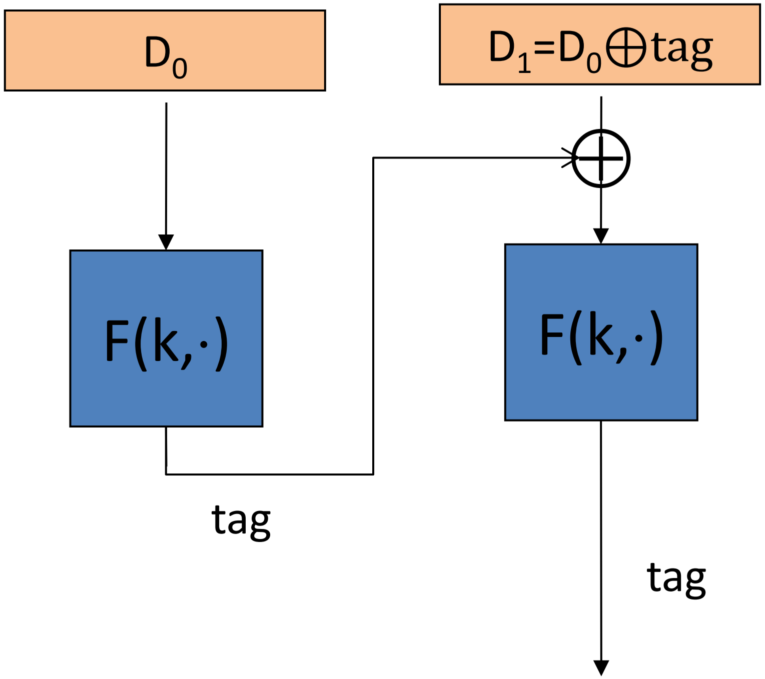
- Let m = D0

(size single block)

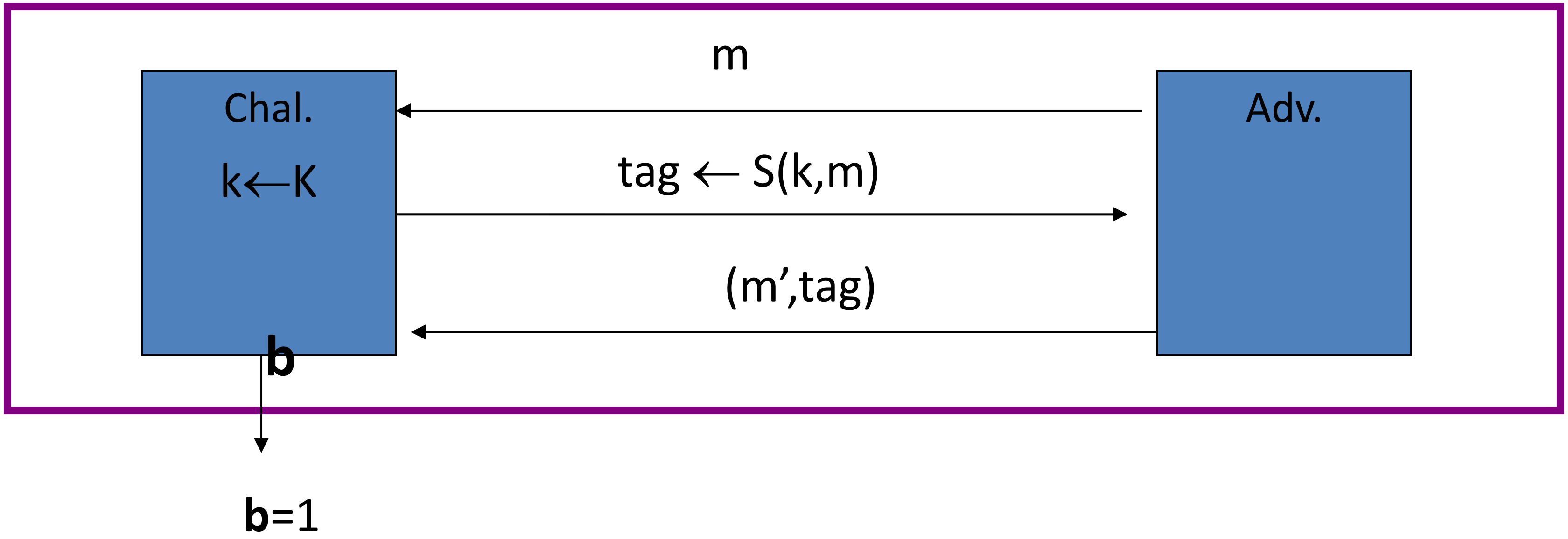


- Let m’ = D0||D0⊕tag

(size two blocks)



Strong Unforgeability under Chosen Message Attack (SUF-CMA)

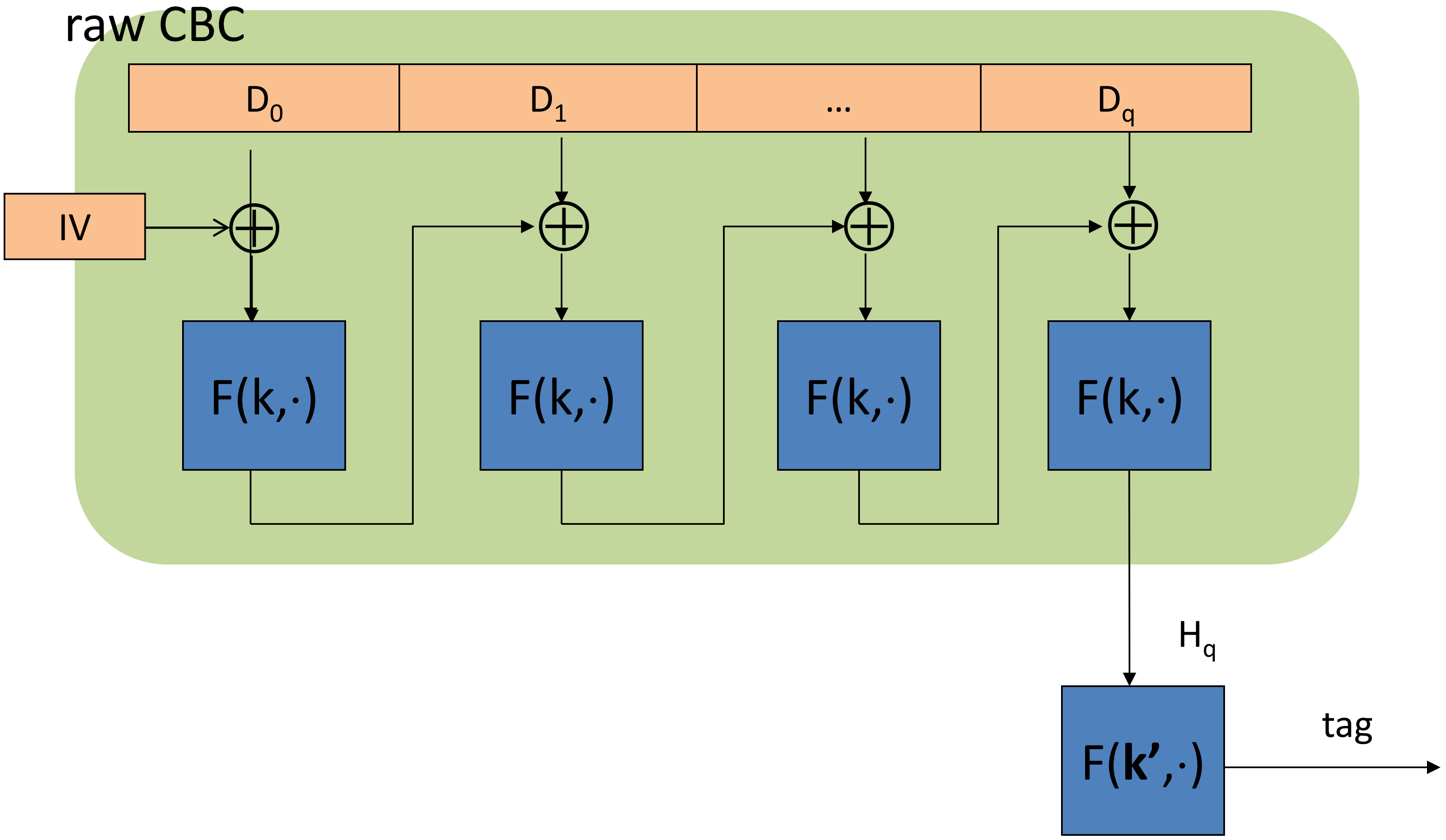


$\text{Adv}_{\text{MAC}}[A,I] = \Pr[\text{Chal. outputs } 1]=1$

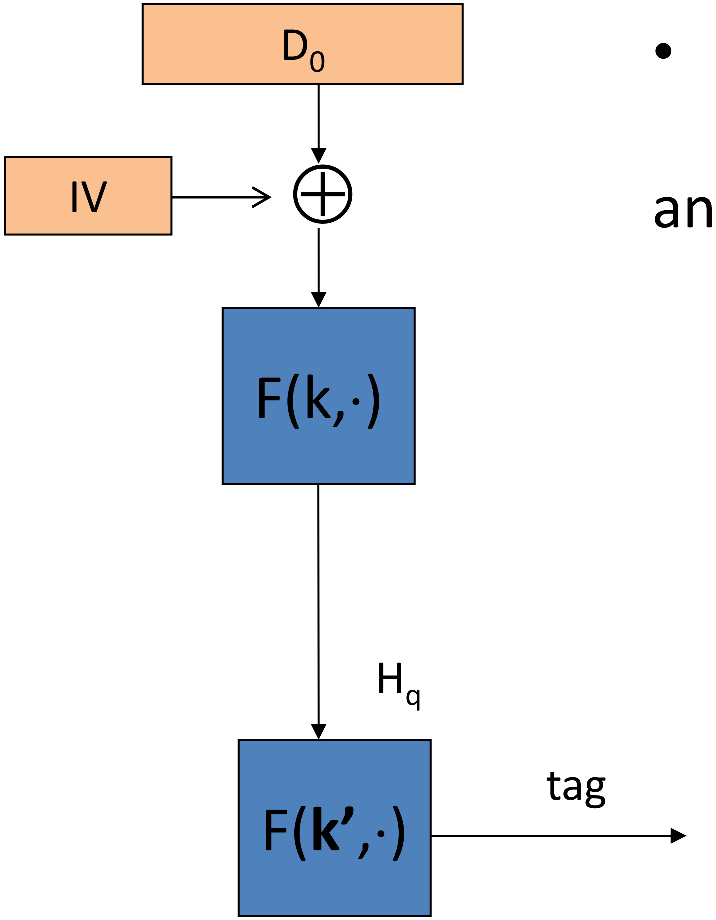
Ασκηση

- Έστω ότι το ECBC-MAC επιλέγει ένα τυχαίο IV για κάθε μήνυμα που προστατεύεται και περιλαμβάνει το IVστο tag. Δηλαδή, S(k,m):=(r, $ECBC_r$(k,m)) όπου το $ECBC_r$(k,m) αναφέρεται στο ECBC χρησιμοποιώντας το r ως IV. Ο αλγόριθμος επιβεβαίωσης V με το κλειδί k, το μήνυμα m, και το tag (r,t) επιστρέφει ``1'', όταν t=$ECBC_r$(k,m) και ``0'', διαφορετικά. Ο αλγόριθμος MAC δεν είναι ασφαλής. Γιατί;

Modified encrypted CBC-MAC (EMAC)



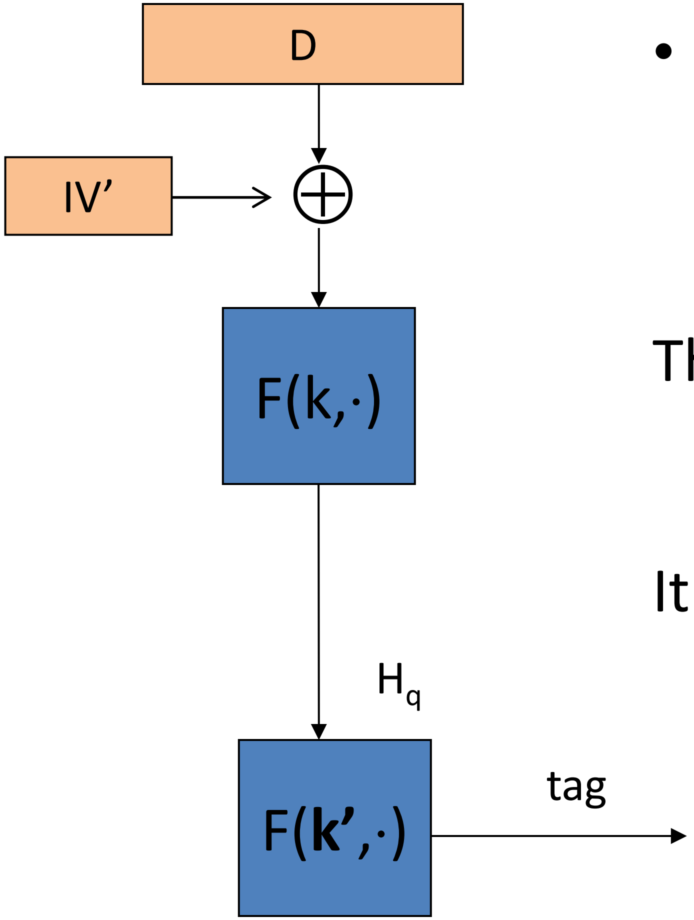
Modified encrypted CBC-MAC (EMAC)



- Let $m_1 = D_0$. Then,
  $tag=(IV,\ ECBC_{IV}(k,m_1))$

and $IV \oplus D_0$ is the input of F()

Modified encrypted CBC-MAC (EMAC)



- It is easy to verify that for any message (for any D)
  - m_2= D, and
  - IV’ = IV $\oplus$ D $\oplus$ D_0

The input of F() is again

IV’ $\oplus$ m2 =IV $\oplus$ D $\oplus$ D_0 $\oplus$D=IV $\oplus$ D_0

It holds that

tag=(IV’, $ECBC_{IV'}(k,m_2)$),

ECBC-MAC and HMAC analysis

<u>Theorem</u>: For any L>0,

For every eff. q-query PRF adv. A attacking F_{ECBC} or F_{NMAC} there exists an eff. adversary B s.t.:

$$\mathrm{Adv}_{PRF}[A, F_{ECBC}] \leq \mathrm{Adv}_{PRP}[B, F] + 2\,q^2 / |X|$$

$$\mathrm{Adv}_{PRF}[A, F_{HMAC}] \leq q \cdot L \cdot \mathrm{Adv}_{PRF}[B, F] + q^2 / 2|K|$$

CBC-MAC is secure as long as $q \ll |X|^{1/2}$

HMAC is secure as long as $q \ll |K|^{1/2}$ (2^{64} for AES-128)

An example

$$\text{Adv}_{\text{PRF}}[A, F_{\text{ECBC}}] \le \text{Adv}_{\text{PRP}}[B, F] + \mathbf{2\, q^2 / |X|}$$

q = # messages MAC-ed with k

Suppose we want $\text{Adv}_{\text{PRF}}[A, F_{\text{ECBC}}] \le 1/2^{32}$ $\Leftarrow$ $q^2 / |X| < 1/ 2^{32}$

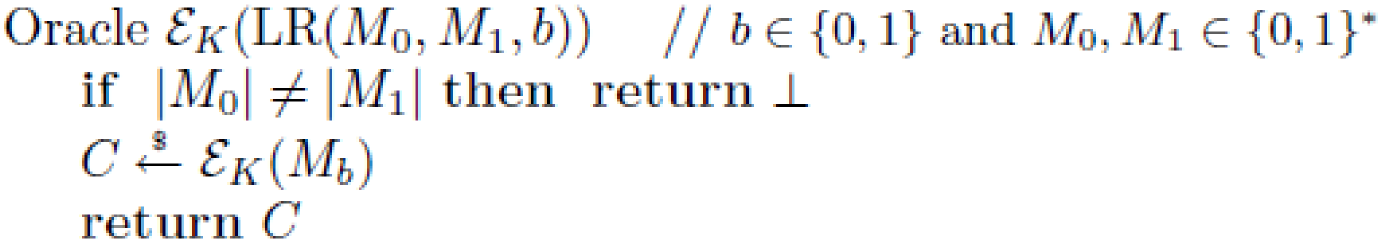
- AES: $|X| = 2^{128}$ $\Rightarrow$ $q < 2^{48}$

  So, after 2^{48} messages must, must change key

- 3DES: $|X| = 2^{64}$ $\Rightarrow$ $q < 2^{16}$

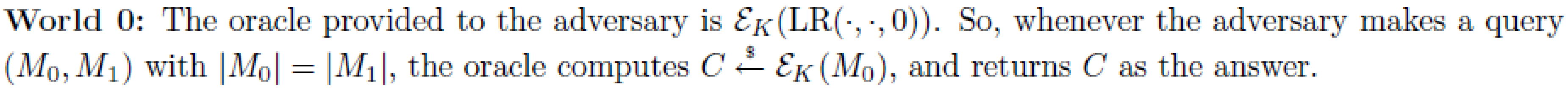
Questions?

IND-CPA

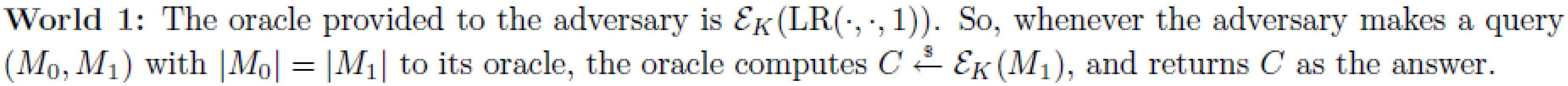


```
Oracle $\mathcal{E}_K(\mathrm{LR}(M_0, M_1, b))$    // $b \in \{0,1\}$ and $M_0, M_1 \in \{0,1\}^*$
    if $|M_0| \neq |M_1|$ then return $\perp$
    $C \xleftarrow{\$} \mathcal{E}_K(M_b)$
    return $C$
```

- We define a world



World 0: The oracle provided to the adversary is $\mathcal{E}_K(\mathrm{LR}(\cdot,\cdot,0))$. So, whenever the adversary makes a query (M_0, M_1) with $|M_0| = |M_1|$, the oracle computes $C \xleftarrow{\$} \mathcal{E}_K(M_0)$, and returns C as the answer.

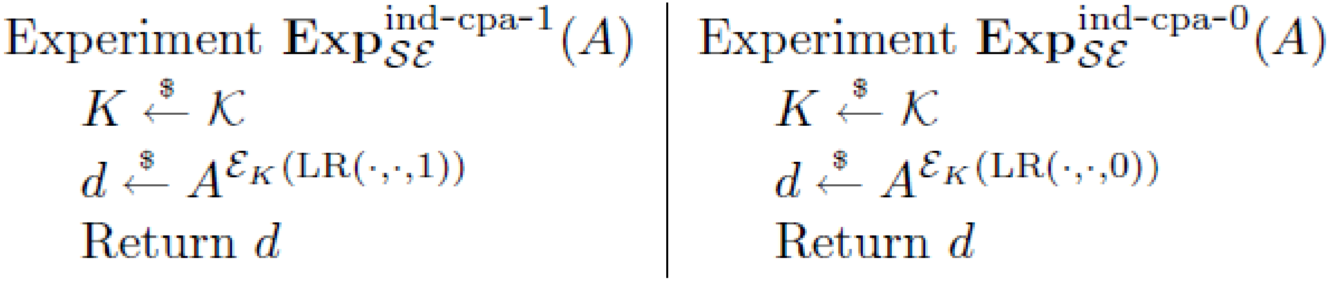


World 1: The oracle provided to the adversary is $\mathcal{E}_K(\mathrm{LR}(\cdot,\cdot,1))$. So, whenever the adversary makes a query (M_0, M_1) with $|M_0| = |M_1|$ to its oracle, the oracle computes $C \xleftarrow{\$} \mathcal{E}_K(M_1)$, and returns C as the answer.

- The problem for the adversary is, after talking to its oracle for some time, to tell which of the two oracles it was given.

Security definition

- Let *SE = (K, E,D) be a symmetric encryption scheme, and let A be an algorithm that has* acc ing exp



```
Experiment $\mathbf{Exp}_{\mathcal{SE}}^{\text{ind-cpa-1}}(A)$
    $K \xleftarrow{\$} \mathcal{K}$
    $d \xleftarrow{\$} A^{\mathcal{E}_K(\text{LR}(\cdot,\cdot,1))}$
    Return $d$
```

```
Experiment $\mathbf{Exp}_{\mathcal{SE}}^{\text{ind-cpa-0}}(A)$
    $K \xleftarrow{\$} \mathcal{K}$
    $d \xleftarrow{\$} A^{\mathcal{E}_K(\text{LR}(\cdot,\cdot,0))}$
    Return $d$
```

$$\mathbf{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(A) \quad = \quad \mathbf{Pr}\left[\mathbf{Exp}_{\mathcal{SE}}^{\text{ind-cpa-1}}(A) = 1\right] - \mathbf{Pr}\left[\mathbf{Exp}_{\mathcal{SE}}^{\text{ind-cpa-0}}(A) = 1\right] .$$

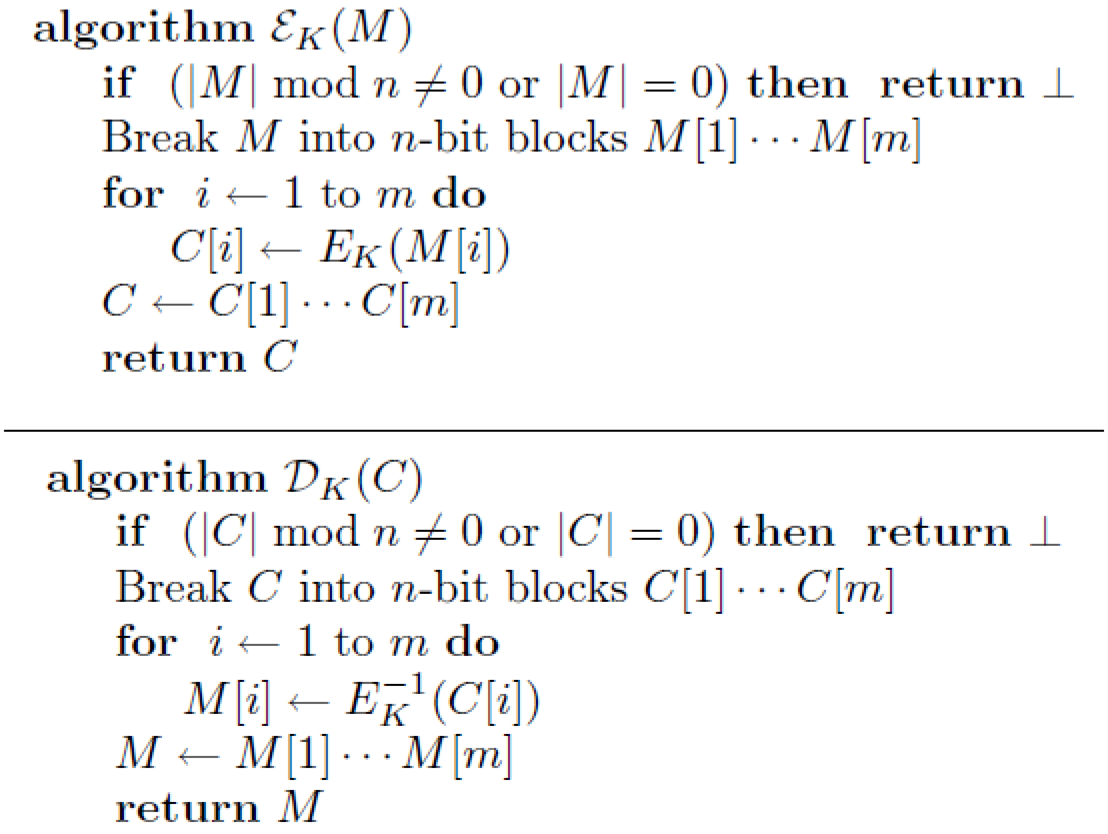
The *IND-CPA advantage of A is defined as*

Security definition

- IND-CPA is a very strong notion of security and covers all our security goals

- Easy to check…
  - Recover the secret key
  - Distinguish the ciphertext from a random message
  - Create a valid plaintext-ciphertext pair
  - Recover the plaintext from the ciphertext

ECB mode is NOT IND-CPA

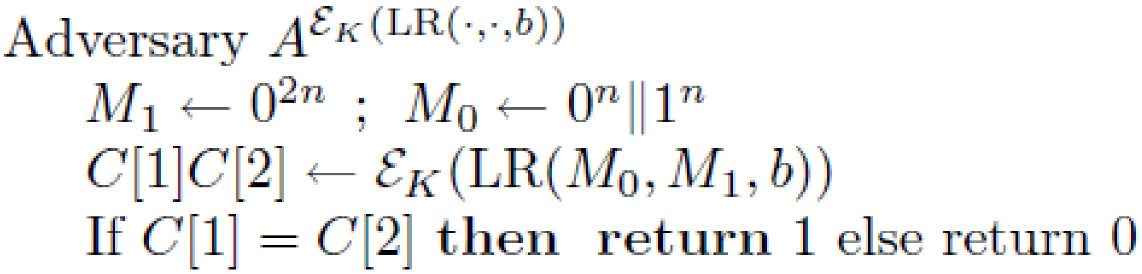


```
algorithm $\mathcal{E}_K(M)$
    if ($|M| \bmod n \neq 0$ or $|M| = 0$) then return $\perp$
    Break $M$ into $n$-bit blocks $M[1] \cdots M[m]$
    for $i \leftarrow 1$ to $m$ do
        $C[i] \leftarrow E_K(M[i])$
    $C \leftarrow C[1] \cdots C[m]$
    return $C$
```

```
algorithm $\mathcal{D}_K(C)$
    if ($|C| \bmod n \neq 0$ or $|C| = 0$) then return $\perp$
    Break $C$ into $n$-bit blocks $C[1] \cdots C[m]$
    for $i \leftarrow 1$ to $m$ do
        $M[i] \leftarrow E_K^{-1}(C[i])$
    $M \leftarrow M[1] \cdots M[m]$
    return $M$
```

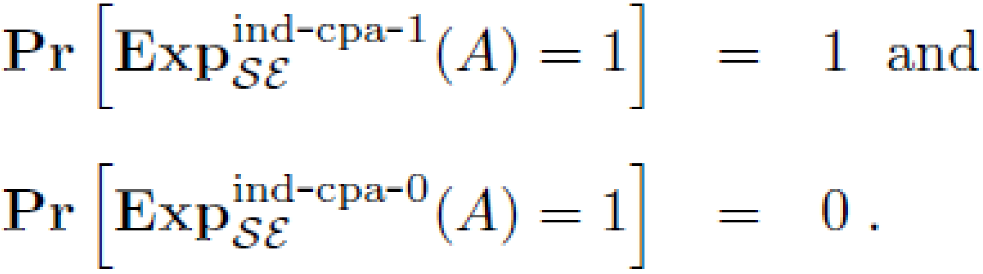
Attack ECB mode

- W ary *A.*



```
Adversary $A^{\mathcal{E}_K(\mathrm{LR}(\cdot,\cdot,b))}$
    $M_1 \leftarrow 0^{2n}$ ; $M_0 \leftarrow 0^n \| 1^n$
    $C[1]C[2] \leftarrow \mathcal{E}_K(\mathrm{LR}(M_0, M_1, b))$
    If $C[1] = C[2]$ then return 1 else return 0
```

- *We cla*



$$\Pr\left[\mathbf{Exp}_{\mathcal{SE}}^{\text{ind-cpa-1}}(A) = 1\right] = 1 \text{ and}$$

$$\Pr\left[\mathbf{Exp}_{\mathcal{SE}}^{\text{ind-cpa-0}}(A) = 1\right] = 0 \, .$$

Why?

- In world 1,
  - *b = 1,*
  - *the oracle returns* $C[1]C[2] = E_K(0^n)||E_K(0^n)$*,*
  - *so* $C[1] = C[2]$ *and A* returns 1.
- In world 0,
  - *b = 0,*
  - *the oracle returns* $C[1]C[2] = E_K(0^n)||E_K(1^n)$*.*
  - *Since* E_K *is a permutation, C[1],C[2] are different. So A returns 0 in this case*
- *Thus,* $\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(A) = 1 - 0 = 1$

That is…

- ECB is an insecure encryption scheme *even if the underlying block cipher E is highly secure!!!*
- The weakness is not in the tool being used (here the block cipher) but in the manner we are using it.!!!
- It is the ECB mechanism that is at fault!!!

- Maybe it is a more generic problem…

Any deterministic, stateless schemes is insecure

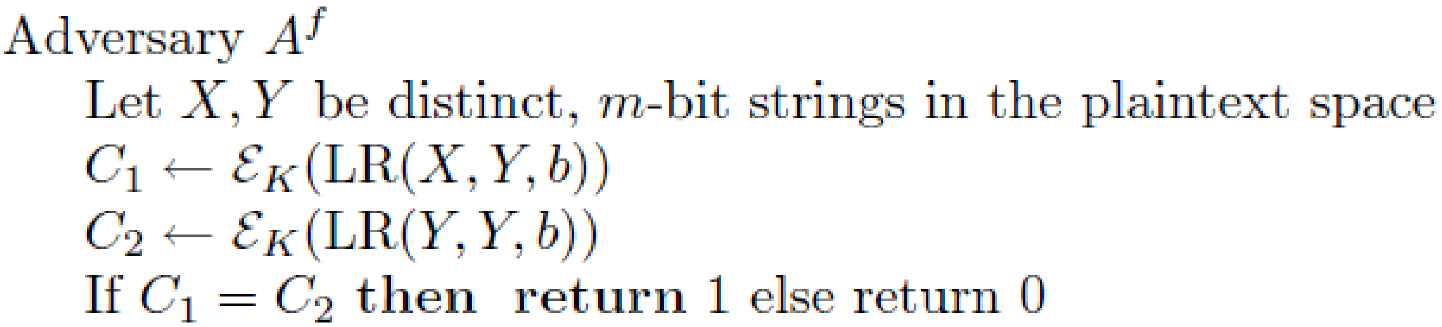
- Let *SE = (K,E,D) be a deterministic, stateless symmetric encryption scheme. Assume* there is an integer *m such that the plaintext space of the scheme c ct strings of length m.* Then there is an adversary *A such that:*

$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(A) = 1$$

Adversary *A runs in time O(m) and asks just two queries, each of length m.*

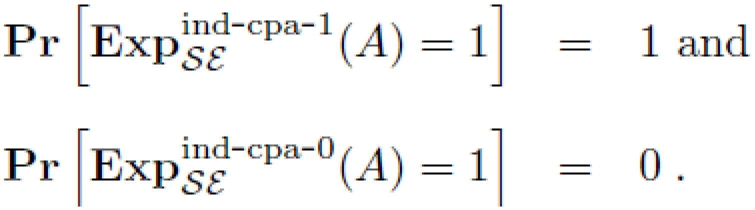
Proof

- We must describe the adversary *A*
- *A is given an lr-encryption* oracle *f* = *E_K(LR(, , b))*
- The goal of *A is to determine the value of b (the*



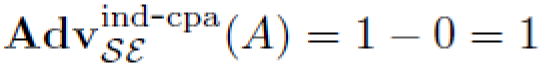
```
Adversary $A^f$
    Let $X, Y$ be distinct, $m$-bit strings in the plaintext space
    $C_1 \leftarrow \mathcal{E}_K(\mathrm{LR}(X, Y, b))$
    $C_2 \leftarrow \mathcal{E}_K(\mathrm{LR}(Y, Y, b))$
    If $C_1 = C_2$ then return 1 else return 0
```

- *It is easy to show that*



$$\begin{aligned}\Pr\left[\mathbf{Exp}^{\text{ind-cpa-1}}_{\mathcal{SE}}(A) = 1\right] &= 1 \text{ and} \\ \Pr\left[\mathbf{Exp}^{\text{ind-cpa-0}}_{\mathcal{SE}}(A) = 1\right] &= 0\,.\end{aligned}$$

- *That it*



$$\mathbf{Adv}^{\text{ind-cpa}}_{\mathcal{SE}}(A) = 1 - 0 = 1$$