Padding

varying message sizes

Enc: add padding

Dec: remove padding

↓ padding scheme works only for messages of size < plaintext bits

$m = 1010\underbrace{00}_{\text{pad}}$

remove padding

Using this, you can encrypt 0 with ElGamal

What if I want to encrypt a very long message? GB

Encrypt(PK, very long M):

generate $ sym key K (AES-CTR)

$\underbrace{Enc_{sym}(K, M)}_{C_1}$; $\underbrace{Enc_{pub}(PK, K)}_{C_2}$

Decrypt(SK, $(C_1; C_2)$):

$Dec_{pub}(SK, C_2) \rightarrow K$

$Dec_{sym}(K, C_1) \rightarrow M$

Cryptographic hash functions

$H: \{0,1\}^* \longrightarrow \{0,1\}^L$

SHA256 $L=256$

deterministic

$H(x)$ = hash of x

digest

fingerprint

$\{0,1\}^{1000} \rightarrow \{0,1\}^{256}$

$2^{1000} \rightarrow 2^{256}$

Correctness: deterministic

Efficiency: computing H should be easy

Security:

) One-way "preimage resistance"

$\forall$ Adv, $\Pr[x \leftarrow \$\,\text{rand}; y = H(x) : \text{Adv}(y) \to x] \doteq \text{negl}\left(\frac{1}{2^{|\text{size of output of } H|}}\right)$

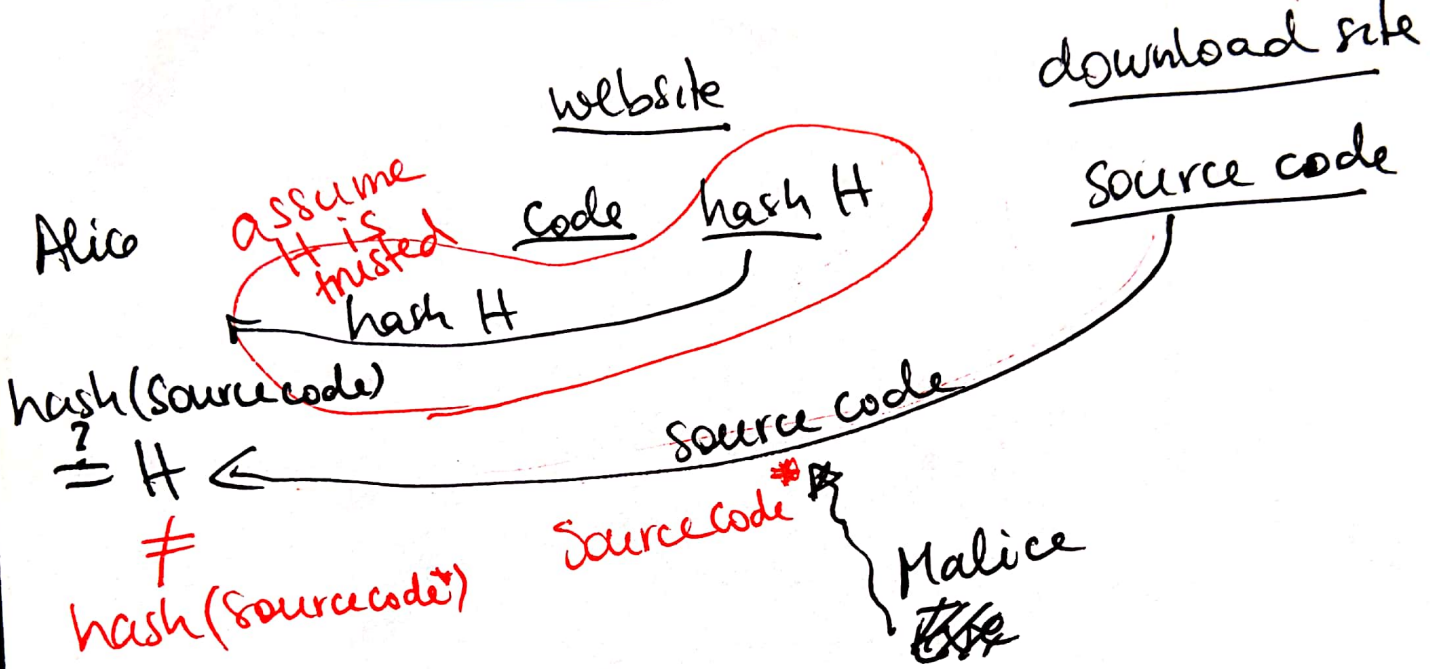
~~$\forall$ Adv, $\forall x$, $\Pr[\text{Adv}(H(x)) \to x] = \text{negl}$~~

~~"2" $\to$ H(2)~~

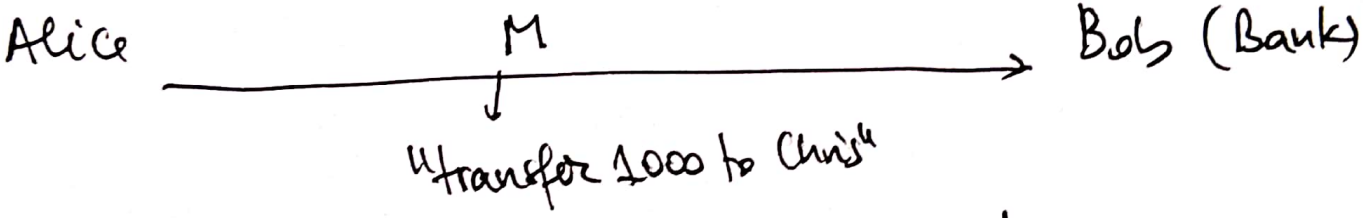
2) Collision-resistance (CR)

It is infeasible to find (x, x') s.t. $(x \neq x')$ and $H(x) = H(x')$

SHA256 is assumed currently to be CR


website
download site
Alice
assume H is trusted
code
hash H
source code
hash H
hash(Source code)
?
= H
source code
≠
hash(sourcecode*)
Sourcecode*
Malice

	Symmetric-Key	Asym-Key
Confidentiality	symmetric-key encryption (AES-CBC)	public-key enc (ElGamal)
Integrity & auth	MAC (AES-EMAC)	Digital Signatures (RSA)



authenticity: M comes from the expected user

integrity: M was not modified

↑ ElGamal $(g^r \bmod p ; \cancel{M \cdot g^k \bmod p})$

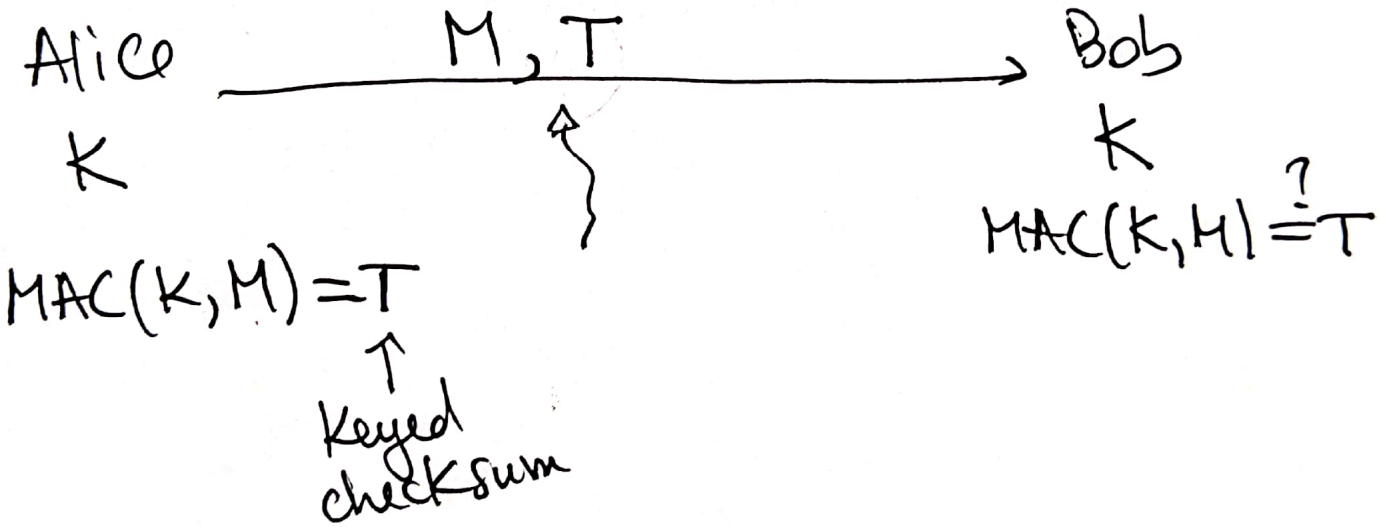
$\times 2$

$\equiv 2M \cdot g^k \bmod p$

↓ easy to modify

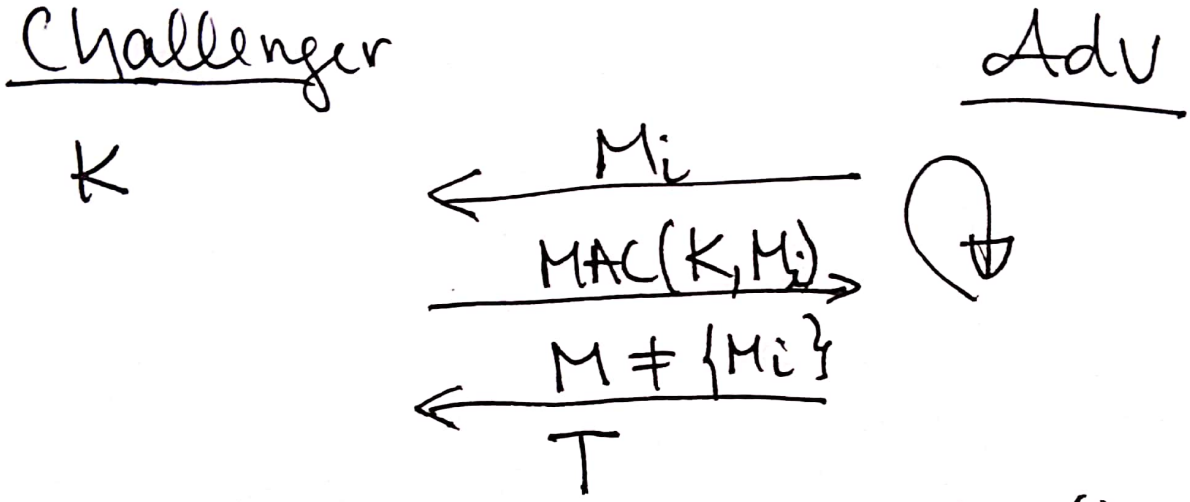
Encryption does not provide these properties

MAC (Message authentication code)



Correctness: determinism
Efficiency: compute MAC should be poly time
Security: EU-CPA existentially-unforgeable under chosen plaintext attack

Adv cannot forge a MAC for a new message

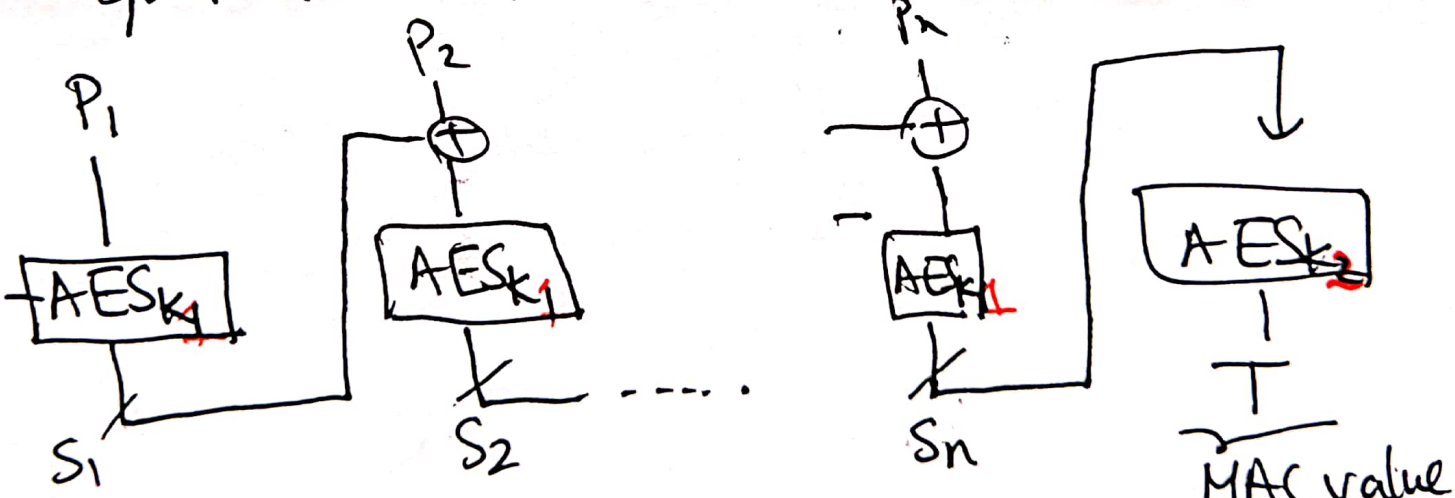


$\nexists$ Adv, $\Pr[K \leftarrow \text{Keygen}(); \text{Adv}() \rightarrow (M,T)$
s.t. $M \neq M_i$, $\text{MAC}(K,M) = T] = \text{negl}$

AES-EMAC

Kygen() :
Choose 2 $\$$ keys $\underbrace{(K_1, K_2)}_{256} = K$

MAC(K, M): $\overbrace{\quad}^{128}$
Split M into $P_1 \| P_2 -- \| P_n$



Propose $MAC'(K, M) = (S_1, \ldots, S_n)$ insecure
can forge MAC' ; Adv asks MAC $M = (P_1 \| P_2) \rightarrow (S_1, S_2)$
$M^* = P_1$; MAC is S_1

Propose $MAC''(K, M) = S_n$ (insecure MAC for exercise)

cAdv asks for $P_1 \rightarrow S_1$ receives.

$P_1' \rightarrow S_1'$

$P_1 \| P_2 \rightarrow T$

forgeable

⇒ points to needing K_2 at the end

cAdv forges MAC for $(P_1'; S_1 \oplus P_2 \oplus S_1')$ is T

Alice

K

M

K_E, K_M

$C = Enc(K_E, M)$;

$MAC(K_M, C)$

Bob

K =

$= (K_E, K_M)$

both confidentiality & integrity/auth

1. checks MAC for C using K_M

$MAC(K_M, C) \overset{?}{=} T$ ✓

2. Decrypt to get M

$Dec(K_E, C) = M$