Cybersecurity for IoT –
Public Key Cryptography

Department of Electrical, Computer and Biomedical Engineering of University of Pavia

Master of Science Program in Computer Engineering

Instructor: Paris Kitsos

http://diceslab.cied.teiwest.gr

E-mail: pkitsos@teimes.gr

Pavia 2018

This lecture is based on "Cryptography and Network Security", 4/e, book by William Stallings

Keyed Hash Functions as MACs

- want a MAC based on a hash function
  - because hash functions are widely available
- hash includes a key along with message
- original proposal:
  - `KeyedHash = Hash(Key|Message)`

HMAC Design Objectives

- allow for easy replaceability of embedded hash function
- use and handle keys in a simple way
- have well understood cryptographic analysis of authentication mechanism strength

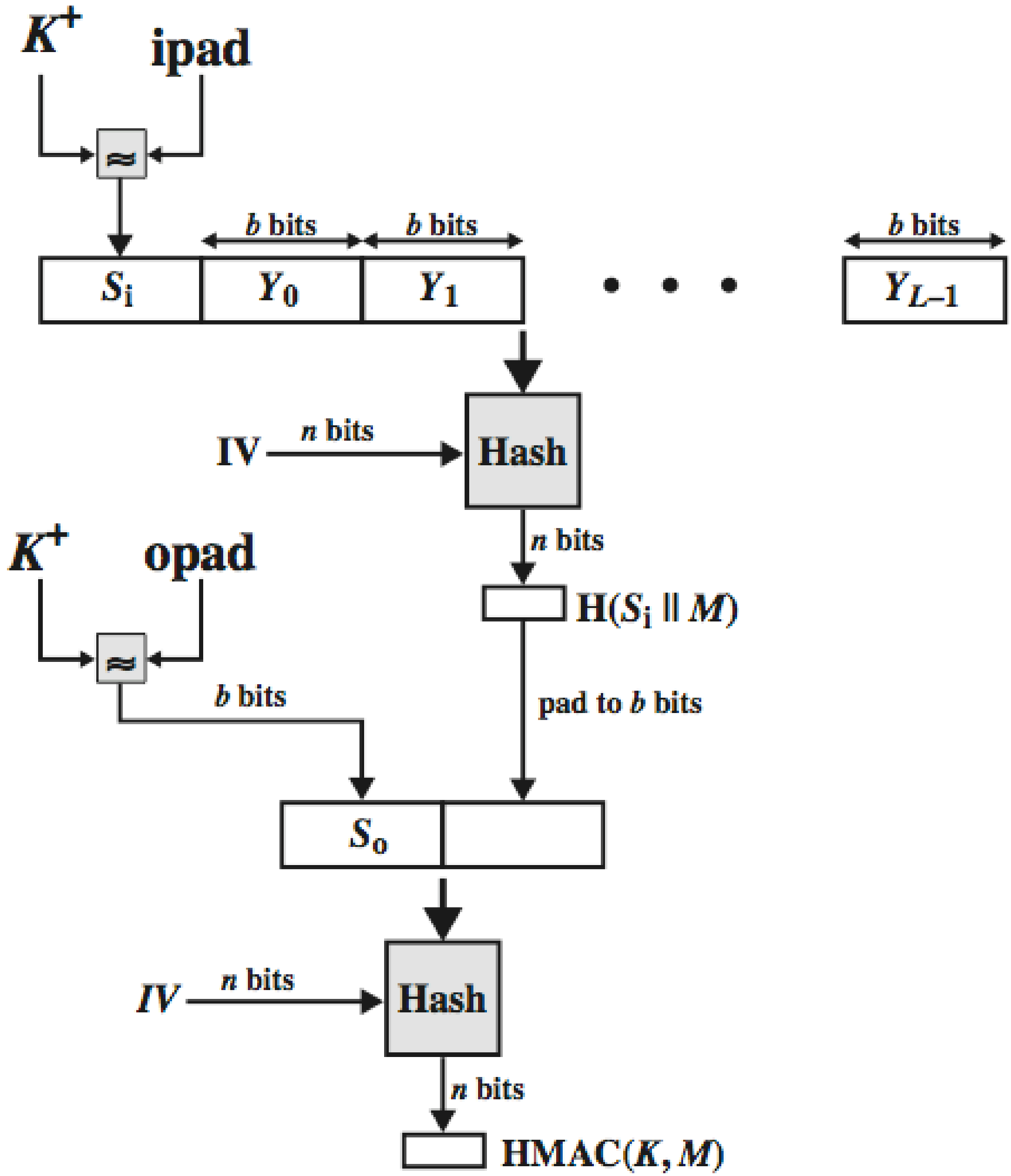
HMAC

- specified as Internet standard RFC2104
- uses hash function on the message:

$$HMAC_K(M) = Hash[(K^+ \text{ XOR } opad) \,||\, Hash[(K^+ \text{ XOR } ipad) \,||\, M)]\,]$$

  – where K^+ is the key padded out to size
  – `opad`, `ipad` are specified padding constants
- any hash function can be used
  – eg. SHA-1, SHA-2, SHA-3, Whirlpool

HMAC Overview



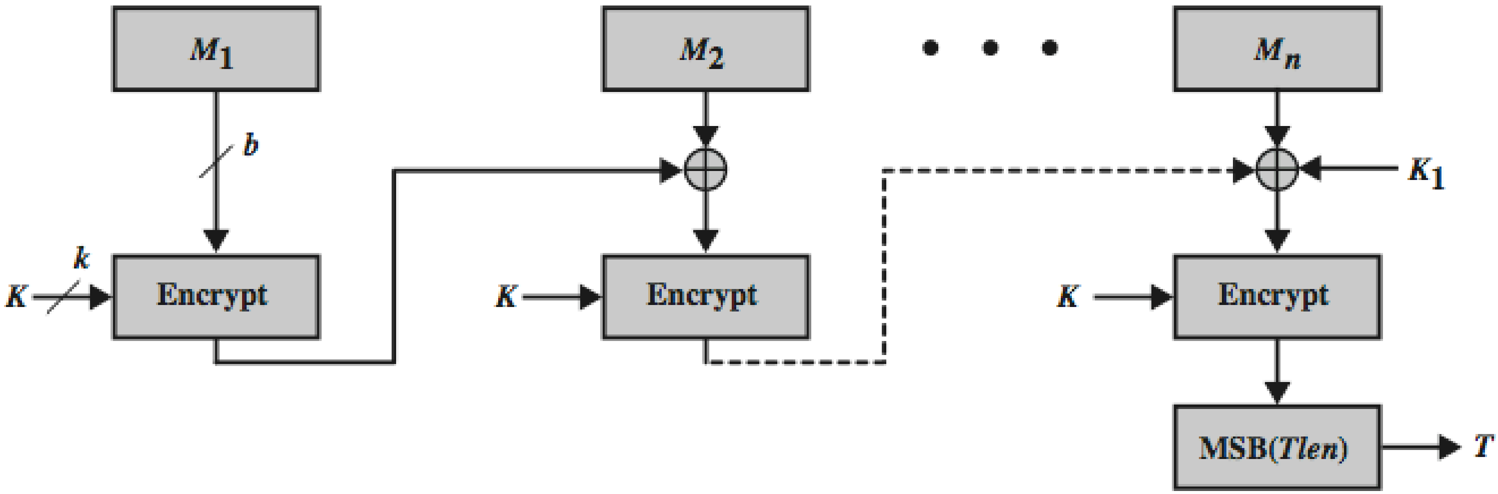
HMAC Security

- proved security of HMAC relates to that of the underlying hash algorithm
- attacking HMAC requires:
  – brute force attack on key used
- choose hash function used based on speed verses security constraints

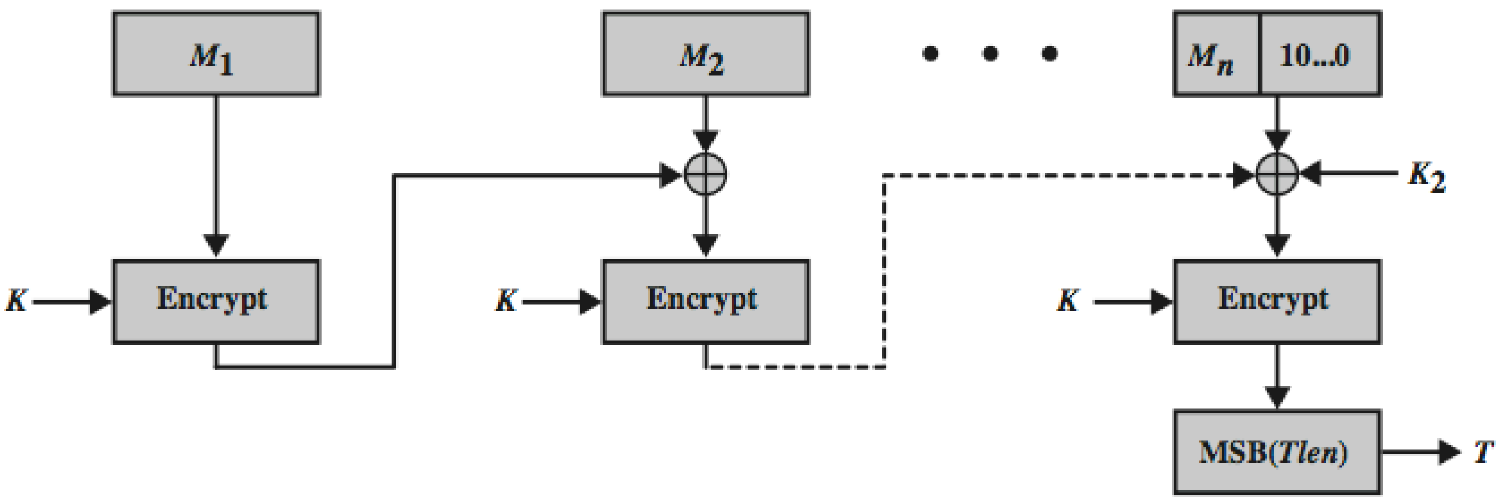
CMAC

- widely used in govt & industry
- but has message size limitation
- can overcome using 2 keys & padding
- Cipher-based Message Authentication Code (CMAC)
- adopted by NIST SP800-38B

CMAC Overview



(a) Message length is integer multiple of block size



(b) Message length is not integer multiple of block size

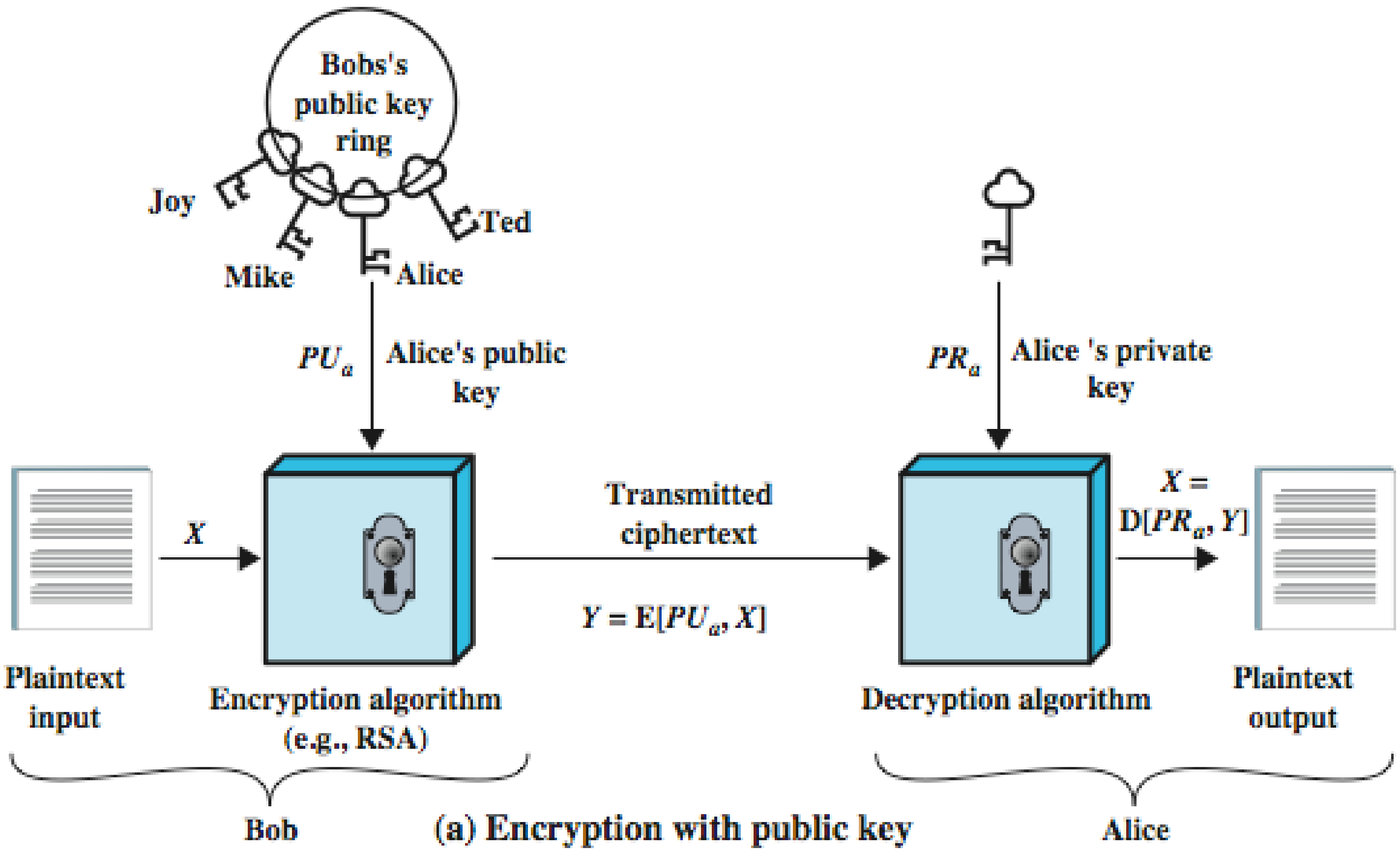
Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key crypto

Why Public-Key Cryptography?

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **digital signatures** – how to verify a message comes intact from the claimed sender
- public invention due to Whitfield Diffie & Martin Hellman at Stanford Uni in 1976
  - known earlier in classified community

Public-Key Cryptography



(a) Encryption with public key

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
- uses large integers (eg. 1024 bits and bigger)
- security due to cost of factoring large numbers

RSA En/decryption

- to encrypt a message M the sender:
  - obtains **public key** of recipient PU={e,n}
  - computes: $C = M^e \bmod n$, where $0 \le M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key PR={d,n}
  - computes: $M = C^d \bmod n$
- note that the message M must be smaller than the modulus n (block if needed)

Modulo Operation [Wikipedia]

- The modulo operation finds the remainder after division of one number by another (sometimes called modulus).
- Given two positive numbers, a (the dividend) and n (the divisor), a modulo n (abbreviated as a mod n) is the remainder of the division of a by n. For example, the expression "5 mod 2" would evaluate to 1 because 5 divided by 2 leaves a quotient of 2 and a remainder of 1

RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random: p, q
- computing their system modulus $n=p.q$
  - note $ø(n)=(p-1)(q-1)$
- selecting at random the encryption key e
  - where $1<e<ø(n)$, $\gcd(e,ø(n))=1$
- solve following equation to find decryption key d
  - $e.d \equiv 1 \bmod ø(n)$ and $0 \leq d \leq n$
- publish their public encryption key: PU={e,n}
- keep secret private decryption key: PR={d,n}

* The symbol $\equiv$ means equivalent

RSA Example - Key Setup

1. Select primes: p=17 & q=11
2. Calculate $n = pq$ =17 x 11=187
3. Calculate ø(n)=(p–1)(q-1)=16x10=160
4. Select e: gcd(e,160)=1; choose e=7
5. Determine d: $de \equiv 1 \bmod 160$ and $d < 160$
   Value is d=23. An example of a simple solution in the next slide!!!!
6. Publish public key PU={7,187}
7. Keep secret private key PR={23,187}

Solution of $de \equiv 1 \bmod 160$

We have $de \equiv 1 \bmod 160$ in which means

$de \bmod 160 = 1 \bmod 160$

$de \bmod 160 = 1$. If $e=7$ then

$7d \bmod 160 = 1$.

Then we are trying all the possibilities of d.

For d=1 then is equation is not true. For d=2 is also not true.

...

For d=23 the equation is true!!

RSA Example - En/Decryption

- sample RSA encryption/decryption is:
- given message M = 88 (nb. 88<187)
- encryption:

  $C = 88^7 \bmod 187 = 11$
- decryption:

  $M = 11^{23} \bmod 187 = 88$

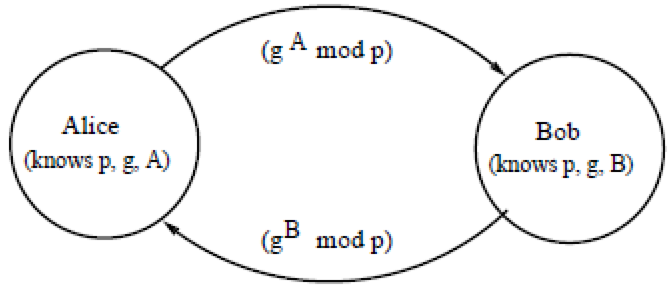
Diffie-Hellman Key Exchange

- first public-key type scheme proposed
- by Diffie & Hellman in 1976 along with the exposition of public key concepts
- is a practical method for public exchange of a secret key
- used in a number of commercial products

Diffie-Hellman Key Exchange

- a public-key distribution scheme
  - cannot be used to exchange an arbitrary message
  - rather it can establish a common key
  - known only to the two participants
- value of key depends on the participants (and their private and public key information)
- based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) - easy
- security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard

Diffie-Hellman...



Steps in the algorithm:

- The two users (e.g Alice and Bob) agree on a prime number p and a base g.
  - g must be a primitive root of p

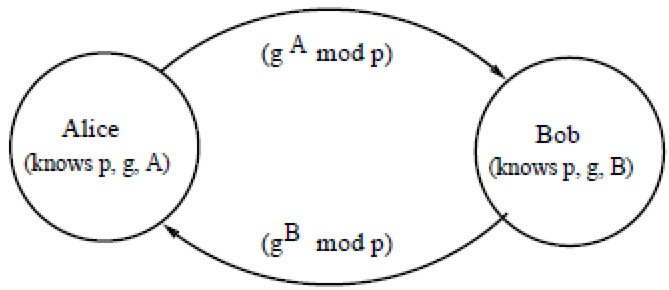
“g must be a primitive root of p” meaning

- Primitive root is an integer g, in which the powers mod p produce the numbers from 1 to p-1
- So, if g is a primitive root of the prime number p, then the numbers produced by g mod p, g^2 mod p, …, g^{p-1} mod p are 1stly) different and 2ndly) are equals to the numbers from 1 to p-1
  - For example, p = 14.
  - The number 14 is coprime with 1, 3, 4, 9, 11 and 13.
  - The number 3 is a primitive root of 14 because of:
    - 3 mod 14 = 3, 3^2 mod 14 = 9, 3^3 mod 14 = 13, 3^4 mod 14 = 11, 3^5 mod 14 = 5 …

Coprime integers

- Two integers a and b are said to be relatively prime, mutually prime, or coprime if the only positive integer (factor) that divides both of them is 1.
- This is equivalent to their greatest common divisor (gcd) being 1, gcd(a, b) =1.

…Diffie-Hellman



Alice chooses a secret number A, and sends Bob the (g^A mod p)

- Bob chooses a secret number B, and sends Alice the g^B mod p
- Alice computes ((g^B mod p)^A mod p)
- Bob computes ((g^A mod p)^B mod p)
- Both parties share the secret key $K_{AB} = g^{AB}$ mod p

Diffie-Hellman Key Exchange

- K_{AB} is used as session key in private-key encryption scheme between Alice and Bob
- if Alice and Bob subsequently communicate, they will have the **same** key as before, unless they choose new public-keys
- attacker needs an x, must solve discrete log

Diffie-Hellman Key Exchange Sceme

Bob		Alice
p, g	Public keys	p, g
A	Private keys	B
g^A mod p	Transmission	g^B mod p
$(g^B \bmod p)^A \bmod p$	Computation	$(g^A \bmod p)^B \bmod p$

Diffie-Hellman Example1

- users Alice & Bob who wish to swap keys:
- agree on prime $p=353$ and $g=3$
- select random secret keys:
  - Alice chooses $A=97$, Bob chooses $B=233$
- compute respective public keys:
  - $y_A=3^{97} \bmod 353 = 40$ (Alice)
  - $y_B=3^{233} \bmod 353 = 248$ (Bob)
- compute shared session key as:
  - $K_{AB}= y_B^{A} \bmod 353=248^{97} \bmod 353= 160$ (Alice)
  - $K_{AB}= y_A^{B} \bmod 353 = 40^{233} \bmod 353 = 160$ (Bob)

Diffie-Hellman Example2

- Alice and Bob agree on p = 23 and g = 5.
- Alice chooses a = 6 and sends 5^6 mod 23 = 8.
- Bob chooses b = 15 and sends 5^{15} mod 23 = 19.
- Alice computes 19^6 mod 23 = 2.
- 5 Bob computes 8^{15} mod 23 = 2.
- Then 2 is the shared secret.

29

Key Exchange Protocols

- users could create random private/public D-H keys each time they communicate
- users could create a known private/public D-H key and publish in a directory, then consulted and used to securely communicate with them
- authentication of the keys is needed

Questions??