



UNSW
AUSTRALIA

Module2, Part 1: Crypto Building Blocks

Securing Wireless Networks, COMP4337/9337

Never Stand Still

Professor Sanjay K. Jha

School of Computer Science and Engineering, UNSW

Today's Agenda – Part 1

- Learn Crypto building blocks for
- Confidentiality and Authentication :Secret Keys
- Integrity: Message Digests (Hash)
- Integrity and Authentication: HMAC, MAC
- Non-Repudiation: Digital Signature

Security Fundamentals (revision)

confidentiality:

only sender, intended receiver should “understand” message contents

- ☀ sender encrypts message
- ☀ receiver decrypts message

authentication:

sender, receiver want to confirm identity of each other

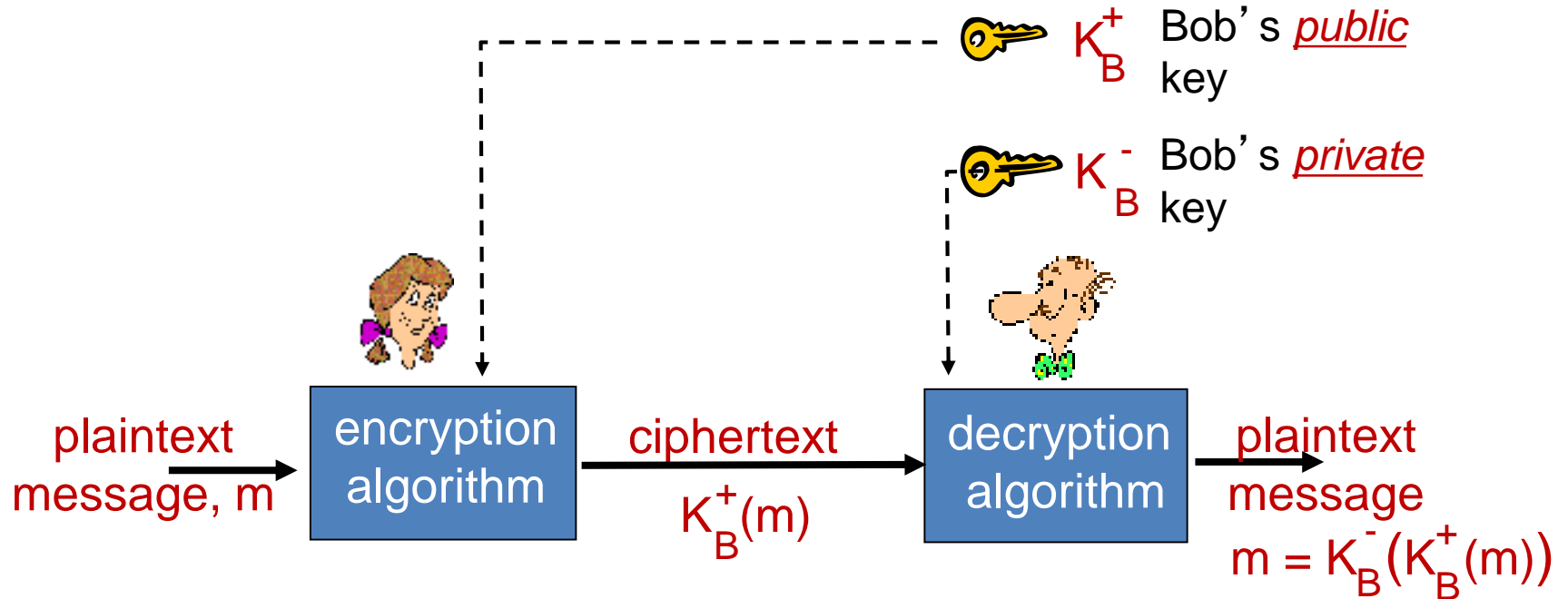
message integrity:

ensure message not altered (in transit, or afterwards)
without detection

Non-repudiation:

associating actions or changes to a unique individual

Public Key Cryptography



Public key encryption algorithms

- Need $K_B^+ (.)$ and $K_B^- (.)$ such that

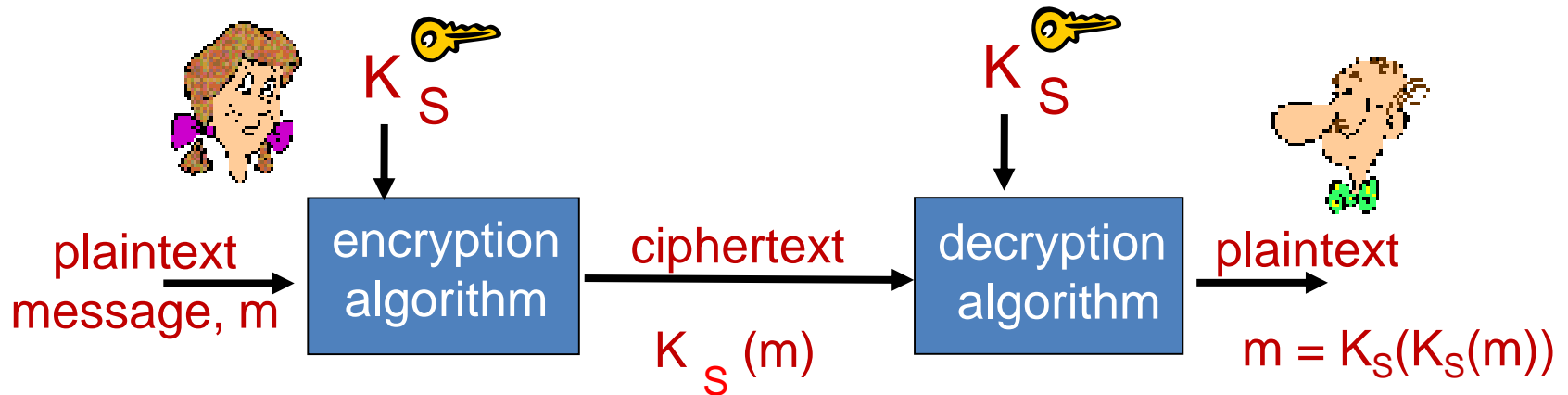
$$K_B^- (K_B^+ (m)) = m$$

- given public key, it should be impossible to compute private key

RSA: Rivest, Shamir, Adelson algorithm

More on Public Key in later week

Symmetric Key Cryptography



symmetric key crypto: Bob and Alice share same (symmetric)
key: K_S

Q: how do Bob and Alice agree on key value?

Session Keys

- Public-key cryptography is computationally intensive as compared to symmetric key crypto
- Exponentiation is computationally intensive
- Symmetric Algorithms much faster than RSA

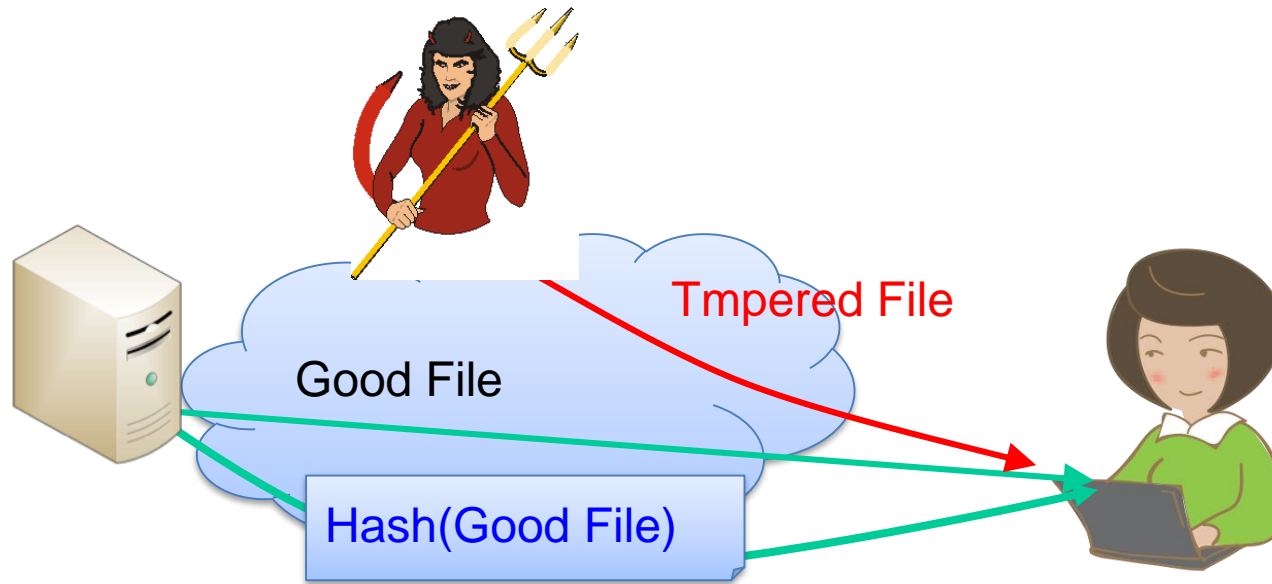
Session key, K_S

- Bob and Alice use RSA to exchange a symmetric key K_S
- Once both have K_S , they use symmetric key cryptography

Confidentiality vs Integrity

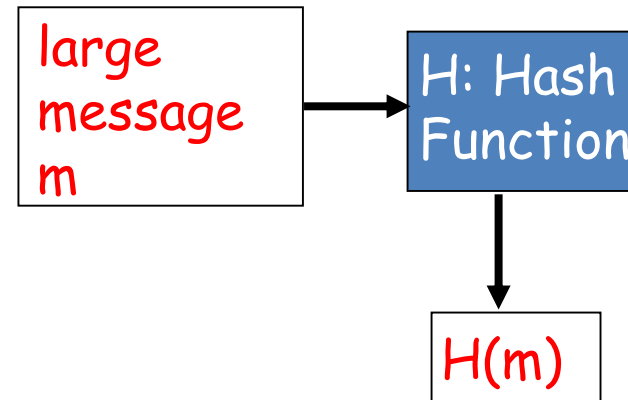
- Confidentiality: message private and secret
- Integrity: protection against message tempering
- Encryption alone may not guarantee integrity
 - Attacker can modify message under encryption without learning what it is
- Public Key Crypto Standards (PKCS)
 - “RSA encryption is intended primarily to provide confidentiality... It is not intended to provide integrity”
- Both Confidentiality and integrity are needed for security

Example on Integrity



- Software distribution protection:
 - For a Good File and Hash(Good File), it is infeasible to find a Tampered File (containing rootkit or Trojan) such that $\text{Hash}(\text{Good File}) = \text{Hash}(\text{Tampered File})$

Hash/Message Digests (MD)



- Function $H()$ that takes as input an arbitrary length message and outputs a fixed-length string:
“message signature”
- The values returned $H()$ are called **hash values**, **hash codes**, **digests**, or simply **hashes**.
- Note that $H()$ is a many-to-1 function
- $H()$ is often called a “hash function”
- Desirable properties:
 - Easy to calculate
 - Irreversibility: Can't determine m from $H(m)$
 - Collision resistance: Computationally difficult to produce m and m' such that $H(m) = H(m')$
 - Seemingly random output

MD Randomness

Message digest of a hashing should have a random pattern

- Randomness: any bit in digest is “1” half the time
- Change input only one bit, and the hash will change half of the digest bits
- Diffusion: if hash function does not exhibit the avalanche effect to a slight change of input, then it has poor randomization, and thus a cryptanalyst can make predictions about the input, being given only the output

Poor Crypto Hash Example

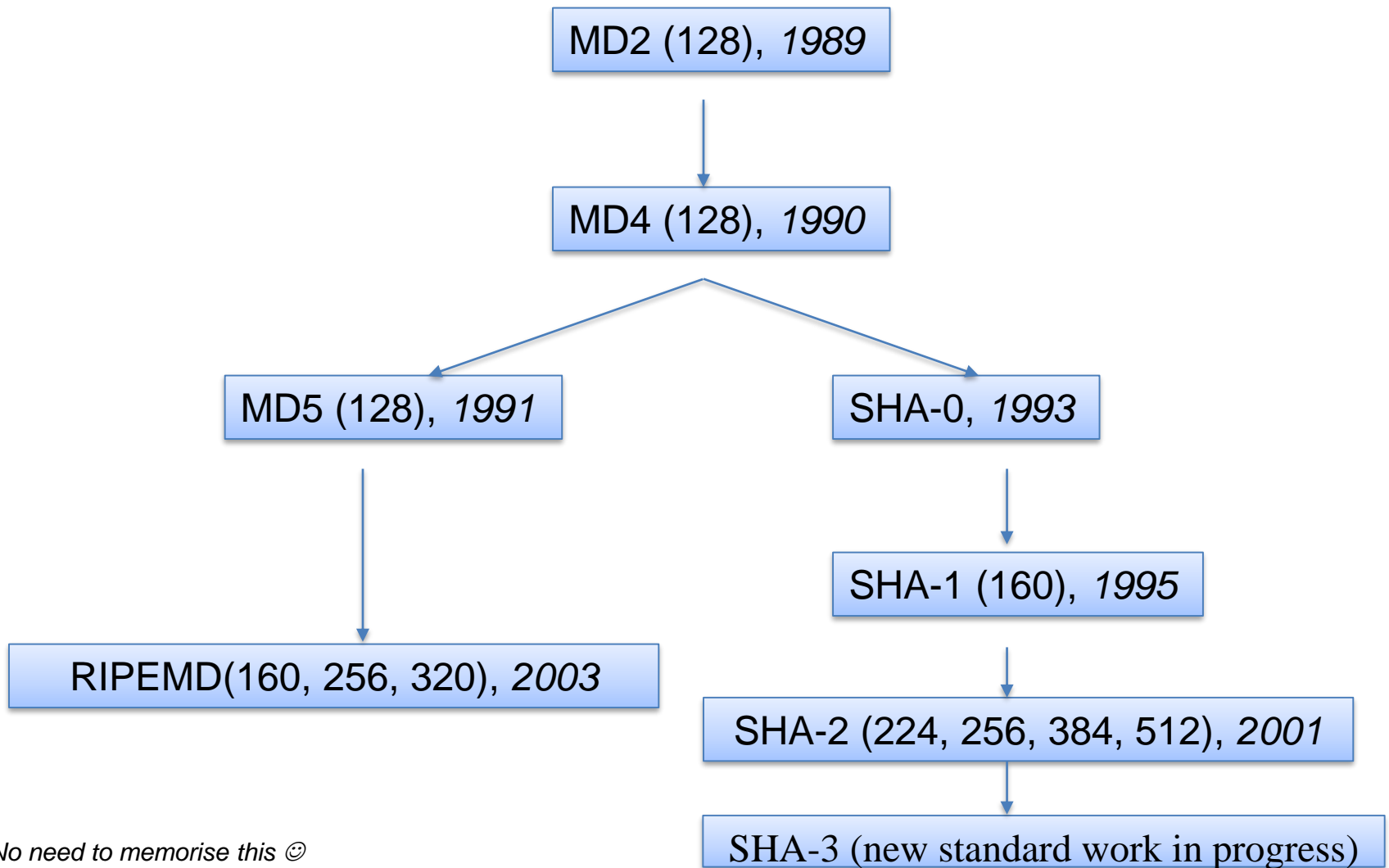
Internet checksum has some properties of hash function:

- ü produces fixed length digest (16-bit sum) of message
- ü is many-to-one

But given message with given hash value, it is easy to find another message with same hash value:

<u>message</u>	<u>ASCII format</u>		<u>message</u>	<u>ASCII format</u>
I O U 1	49 4F 55 31		I O U <u>9</u>	49 4F 55 <u>39</u>
0 0 . 9	30 30 2E 39		0 0 . <u>1</u>	30 30 2E <u>31</u>
9 B O B	39 42 D2 42		9 B O B	39 42 D2 42
	<u>B2 C1 D2 AC</u>	different messages but identical checksums!		<u>B2 C1 D2 AC</u>

Standard Hash Function History



No need to memorise this ☺

Example: SHA-256

- Plaintext

Phishing Explained

Phishing scams are typically fraudulent e-mail messages appearing to come from legitimate sources like your bank, your Internet Service Provider, eBay, or PayPal, for example. These messages usually direct you to a fake web site and ask you for private information (e.g., password, credit card, or other account updates). The perpetrators then use this private information to commit identity theft.

Warning Signs

.....

URLs Don't Match - Place your mouse over the link in the e-mail message. If the URL displayed in the window of your browser is not exactly the same as the text of the link provided in the message, run. It's probably a fake. Sometimes the URLs do match and the URL is still a fake.

- Digest in BASE64

Tx9S2IwqlrGI7hhNJ4s5K7qiYt3PjQSD6vWH4QB17
yg =

- The corresponding digest in binary format is listed as follows:

```
1010110001010110011110011101000000111010000111101101110100101001011011111011010001000010001001101100001
0110011010101101100100110111100010011001101100101010011010001110100110110111101011000000011011000101110
10011011111111110100110111100001100101001
```

Example: Diffusion in SHA-256

- Plaintext: only change the first letter from P to Q

Qhishing Explained

Phishing scams are typically fraudulent e-mail messages appearing to come from legitimate sources like your bank, your Internet Service Provider, eBay, or PayPal, for example. These messages usually direct you to a fake web site

URLs Don't Match - Place your mouse over the link in the e-mail message. If the URL displayed in the window of your browser is not exactly the same as the text of the link provided in the message, run. It's probably a fake. Sometimes the URLs do match and the URL is still a fake.

- Digest in BASE64 changes from

`Tx9S2IwqlrGI7hhNJ4s5K7qiYt3PjQSD6vWH4QBl7yg =`

to

`AAHew/sW6W4X39EQq7ctFEb3PyHxka+T3D1UQJhDkw =`

- The corresponding digest in binary format is listed as follows:

```
1010010011001011100010110101000100101100110011111010000001001011011011100101100010110000
010011001110011001011001110010000010000101000101010000110111110101001011011011111111111
1110111011111011010011000000100000111100101110110011001011010111111010000010000
```

Basic Structure of SHA-1

For illustration, not examinable, details in Wu/Irwin text

Against padding attacks

Message length
($K \bmod 2^{64}$)

$$L \times 512 \text{ bits} = n \times 32 \text{ bits}$$

K bits

Message

100...0

Padding
(1 to 512 bits)

64
bits

Split message into 512-bit blocks

← 512 bits

512 bits

512 bits

- 512 bits

$$Y_0$$
$$Y_1$$
$$Y_i$$
$$Y_{l-1}$$

IV

160

1

H

160



160

160

160-bit **buffer** (5 registers)
initialized with magic values

Hash function

Applied to each 512-bit block
and current 160-bit buffer.

160-bit
digest

SHA properties

Algorithm Name	Max. Message Size (bits)	Block Size (bits)	Word size (bits)	Digest size (bits)	Collision resistance (bits)
SHA-1	2^{64}	512	32	160	80
SHA-256	2^{64}	512	32	256	128
SHA-384	2^{128}	1024	64	384	192
SHA-512	2^{128}	1024	64	512	256

No need to memorise this 😊, appreciate and consult table as needed. Details on collision resistance is beyond scope of this course. We will see some related practical issues in WEP protocol later. Basically determines strength of scheme under brute force attack i.e. how quickly a collision can be found on a fast machine. 160 bit digest is considered unsafe with fast computers.

Attacks on Hash (Self study/no-exam)

- MD5 is one of the most widely used cryptographic hash functions
- Attack on MD5 to find collisions efficiently
 - About 15 minutes up to an hour computation time
 - Finding a collision for MD5 is easily feasible
- This attack is also able to break hash functions efficiently, including HAVAL-128, MD4, RIPEMD, SHA-0 and SHA-1
- SHA-1 near-collision attack needs $2^{57.5}$ hashes
 - Marc Stevens: HashClash on 11/8/2010
 - <http://code.google.com/p/hashclash/>
- SHA-1 Chosen-prefix collision attack: $2^{77.06}$
- Ref:
 - Xiaoyun Wang, Hongbo Yu: How to Break MD5 and Other Hash Functions. EUROCRYPT 2005: 19-35
 - Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, Xiuyuan Yu: Cryptanalysis of the Hash Functions MD4 and RIPEMD. EUROCRYPT 2005: 1-18
 - Xiaoyun Wang, Yiqun Lisa Yin, Hongbo Yu: Finding Collisions in the Full SHA-1. CRYPTO 2005: 17-36
 - Xiaoyun Wang, Hongbo Yu, Yiqun Lisa Yin: Efficient Collision Search Attacks on SHA-0. CRYPTO 2005: 1-16
 - Hongbo Yu, Gaoli Wang, Guoyan Zhang, Xiaoyun Wang: The Second-Preimage Attack on MD4. CANS 2005: 1-12

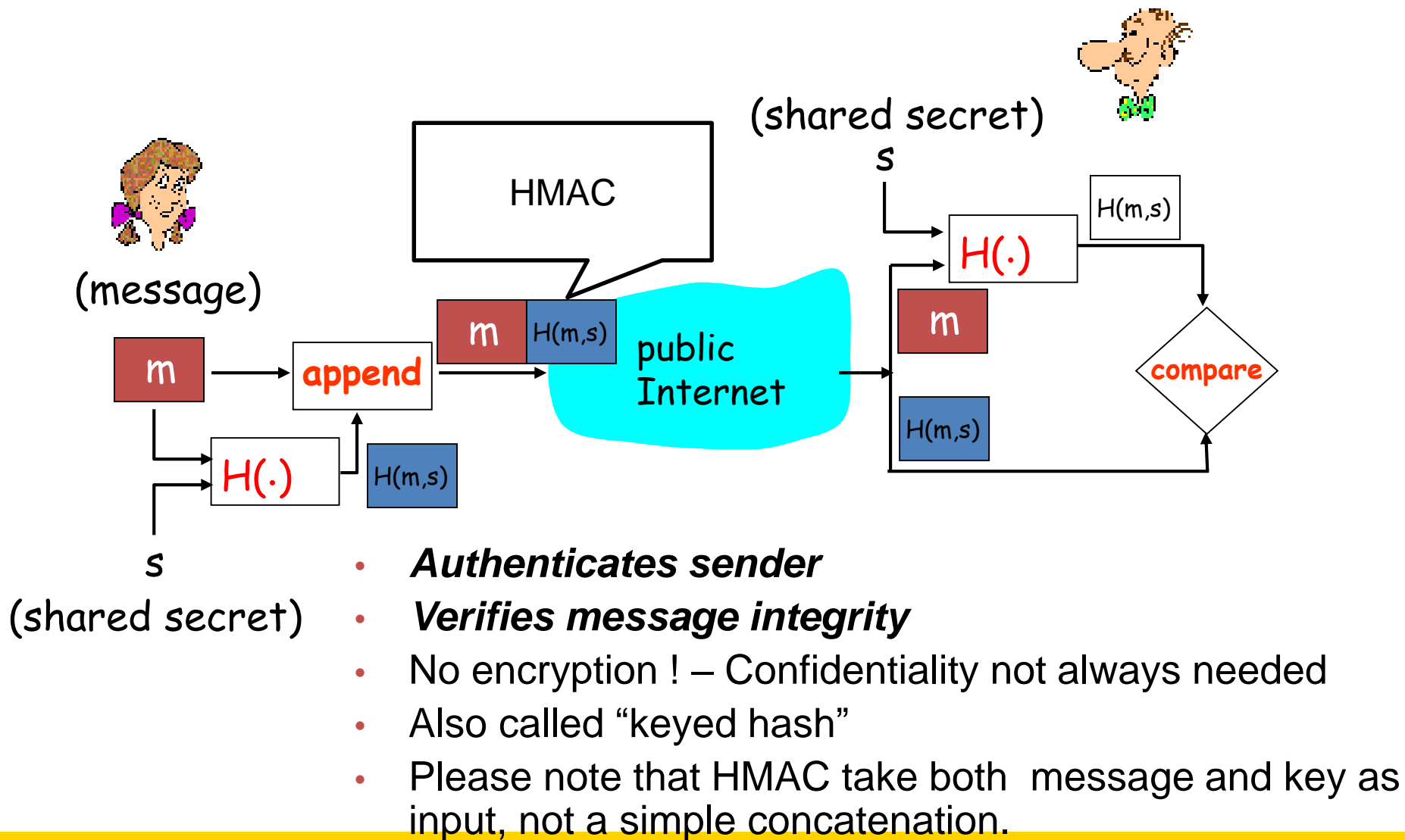
Integrity vs Authentication

- ❖ suppose Alice creates msg m , and calculates hash $H(m)$ using a hash function (e.g. SHA-1)
- ❖ Alice send the extended message $(m, H(m))$ to Bob
- ❖ Bob upon receipt of this message independently calculates $H(m')$ from the message
- ❖ If $H(m) = H(m')$, message integrity verified.
- ❖ *What if Trudy intercepts the original message, replaces it with a new message n and its hash $H(n)$?*

HMAC

- FIPS 198, RFC 2104
 - Keyed-hash message authentication code: a message authentication code that uses a cryptographic key in conjunction with a hash function
 - Runs data and authentication key through hash function twice
 - Hashing is faster than encryption in software
 - Allow the use of any hash function, e.g., SHA-256, or SHA-512
 - No US export restrictions on HMAC and hash
- Used in TLS and IPSec (later weeks)

HMAC: Integrity and Authentication



Integrity of files using HMAC

Example:

- Use the following key in ASCII format.

`TigersFootballWarEagleBCSChampion`

- The corresponding Hex format for the key is

`546967657273466F6F7462616C6C5761724561676C65424353436
8616D70696F`

Example: HMAC-SHA-256

- Plaintext

Phishing Explained

Phishing scams are typically fraudulent e-mail messages appearing to come from legitimate sources like your bank, your Internet Service

.....

- HMAC Digest in BASE64

HadCmwGX9EGKBon0C+6XEInXCI8 =

- Only change the first char from *Tiger* to *Siger* in key, HMAC Digest in BASE64

Ar89wBq+6rxq4Eenho53TMiHvSw =

Hash, MAC and HMAC

- One way Hash Function: Doesn't take any secret key or its operation
 - Easy to compute (both hardware and software solutions possible)
 - Concatenation of a Secret Key and Message can be passed through a hash function without any need for encryption/decryption for efficiency.
- Message Authentication Code: MAC
 - Can use functions like DES, last bits of the cipher-text can be used as code. However, encryption software is slow, hence may be avoided.
- HMAC: Key is XORed with ipad (part of pad) and then concatenated with the message m. This mix is run through a hash function. The result is then concatenated with XOR of Key and opad (part of pad). Now, this whole mix is run through hash function one more time.

Digital signatures

cryptographic technique analogous to hand-written signatures:

- sender (Bob) digitally signs document, establishing he is document owner/creator.
- *verifiable, nonforgeable*: recipient (Alice) can prove to someone that Bob, and no one else (including Alice), must have signed document


Digital signatures

simple digital signature for message m :

- Bob signs m by encrypting with his private key K_B^- , creating “signed” message, $K_B^-(m)$

Bob's message, m

Dear Alice
Oh, how I have missed
you. I think of you all the
time! ...(blah blah blah)
Bob

 K_B^- Bob's private
key

Public key
encryption
algorithm

$m, K_B^-(m)$

Bob's message,
 m , signed
(encrypted) with
his private key

Digital signatures

- ❖ suppose Alice receives msg m , with signature: m , to $K_B(m)$
- ❖ Alice verifies m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$.
- ❖ If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key.

Alice thus verifies that:

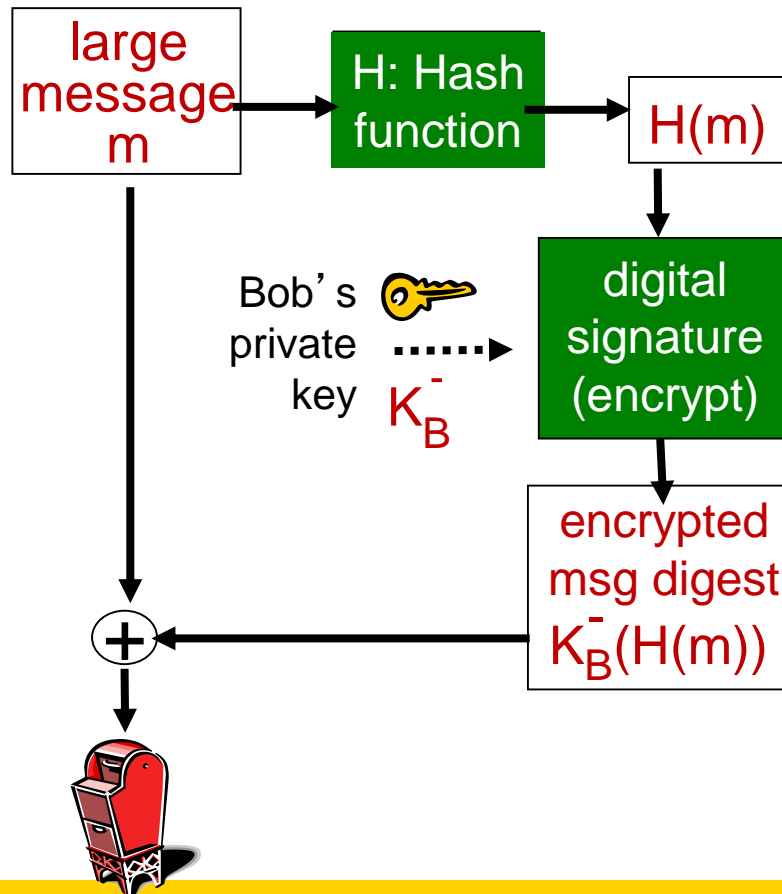
- ü Bob signed m
- ü no one else signed m
- ü Bob signed m and not m'

non-repudiation:

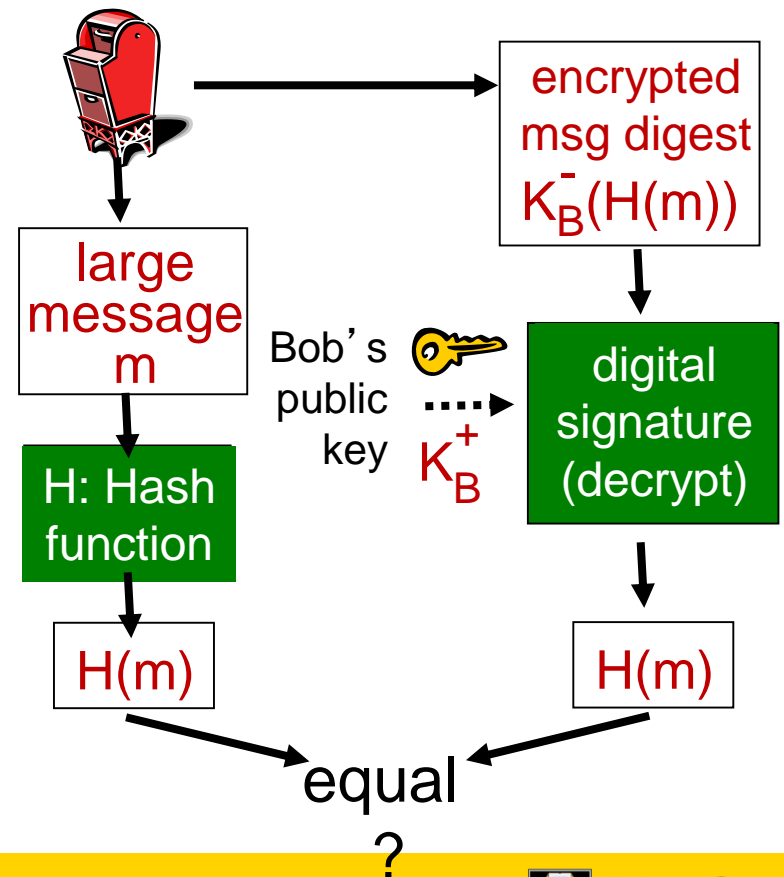
- ✓ Alice can take m , and signature $K_B^-(m)$ to court and prove that Bob signed m

Digital signature = signed message digest

Bob sends digitally signed message:



Alice verifies signature, integrity of digitally signed message:



Summary

- Lot of these crypto mechanism are building block for security protocols: both wired and wireless
- Acknowledgement:
 - Adaptation of foils from Kurose/Ross (revision from basic networking subject COMP3331)
 - Some material from Wu & Irwin book has lot more on various protocols and standards in chapter 20.
 - Details of algorithms beyond scope, you can consult many texts on crypto if interested.
 - Stallings: Network Security Essentials, Chapter 3 has good summary of Hash, HMAC, MAC etc.