Security

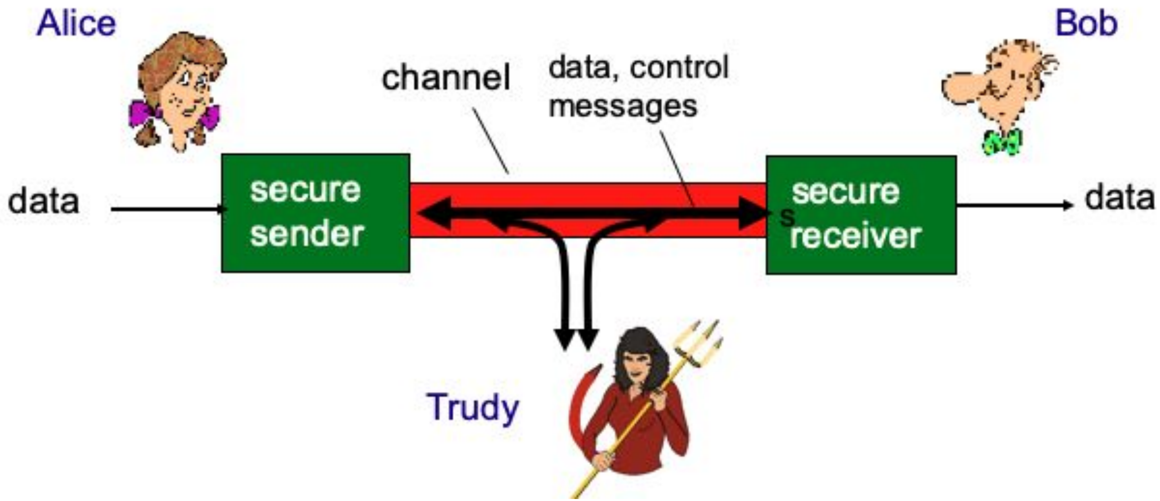
CS5700 Fall 2019

What is network security

- ***Confidentiality***: only sender and intended receiver should “understand” message content
- ***Authentication***: sender, receiver want to confirm identity of each other
- ***Message integrity***: sender, receiver want to ensure message not altered without detection
- ***Access and availability***: service must be accessible and available to users

Friends and enemies: Alice, Bob, Trudy

- Well-known in security world
- Bob, Alice (lovers!) want to communicate “securely”
- Trudy (intruder) may intercept, delete, add messages



Who might Bob, Alice be?

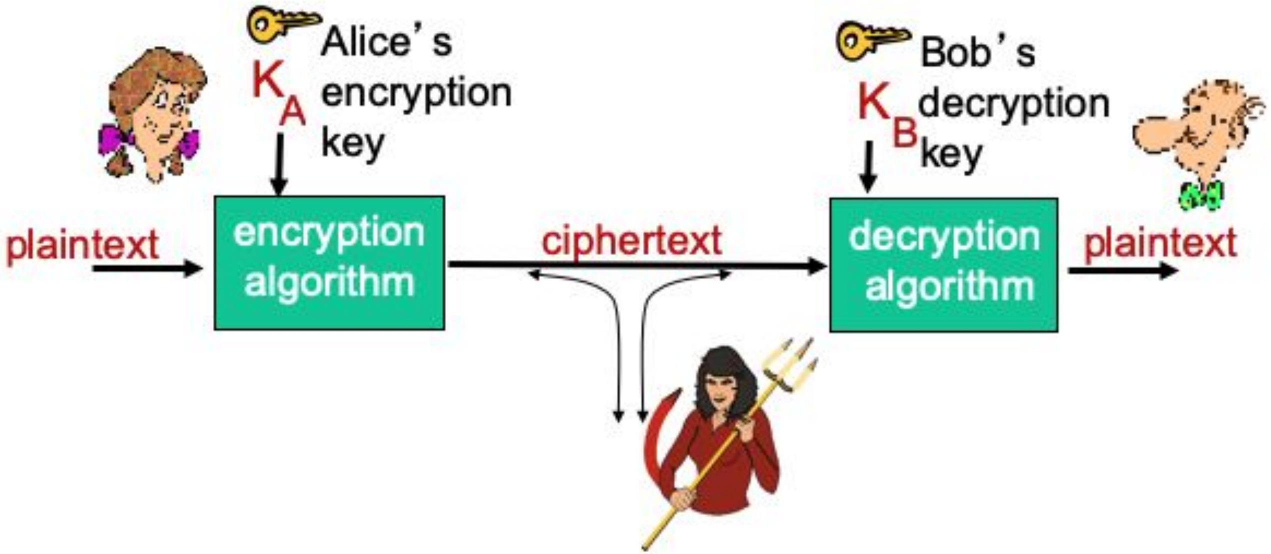
- Web browser and server for electronic transactions (e.g. online purchases)
- Online banking client/server
- DNS servers
- etc.

What can “bad guys” do?

- Eavesdrop: intercept messages
- Impersonation: can fake (spoof) source address in packet (or any field in packet)
- Hijacking: “take over” ongoing connection by removing sender or receiver, inserting himself in place
- Denial of service: prevent service from being used by others (e.g. by overloading resources)
- etc.

Principles of cryptography

The language of cryptography



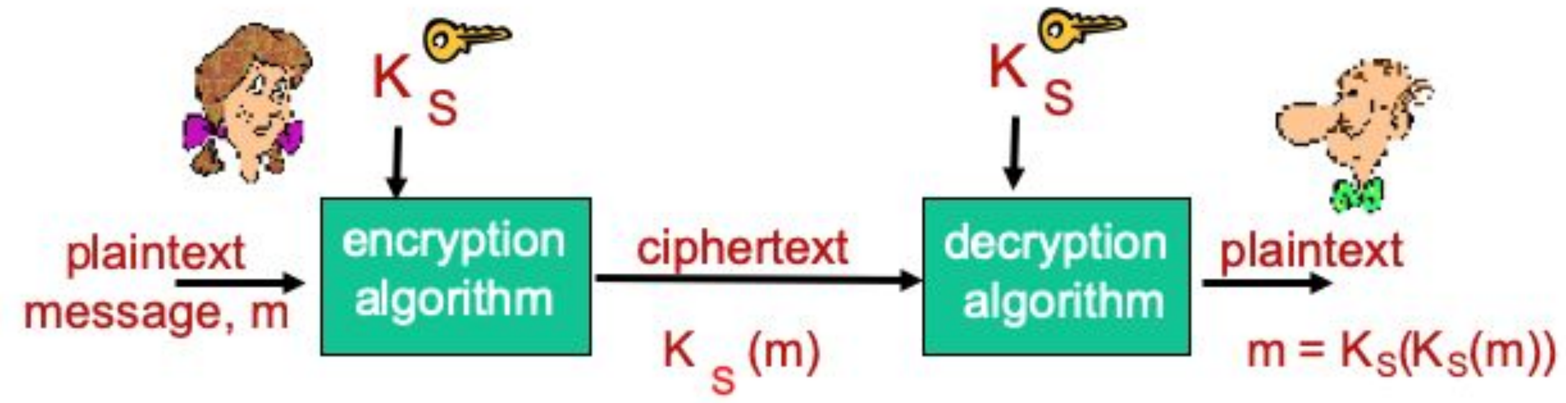
m plaintext message

$K_A(m)$ ciphertext, encrypted with key K_A

$m = K_B(K_A(m))$

Symmetric key cryptography

- Bob and Alice share same (symmetric) key K_S



Monoalphabetic cipher

- Substitute one letter for another
- How large is the key space?

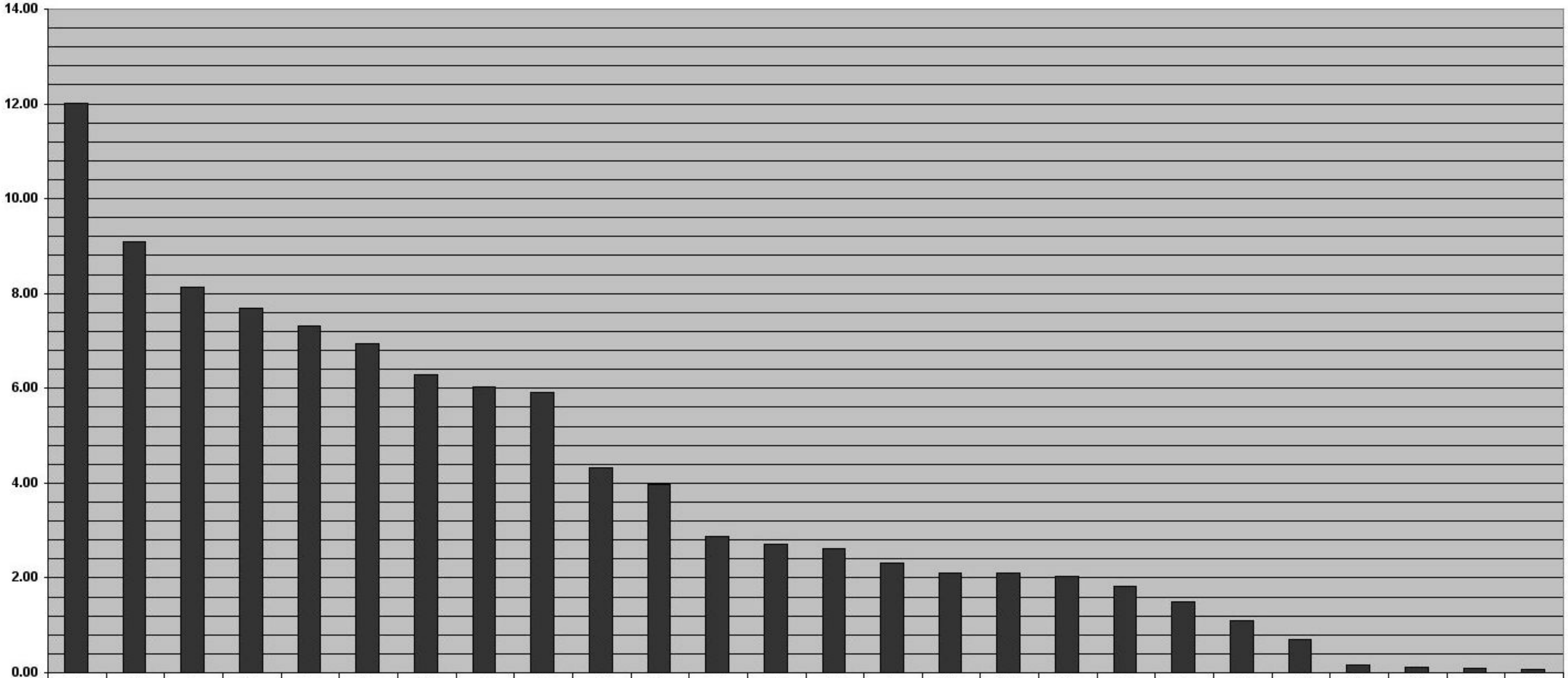
```
plaintext:   abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

e.g.:

```
 Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
```

English letter frequency



	E	T	A	O	I	N	S	R	H	D	L	U	C	M	F	Y	W	G	P	B	V	K	X	Q	J	Z
Frequency	12.02	9.10	8.12	7.68	7.31	6.95	6.28	6.02	5.92	4.32	3.98	2.88	2.71	2.61	2.30	2.11	2.09	2.03	1.82	1.49	1.11	0.69	0.17	0.11	0.10	0.07

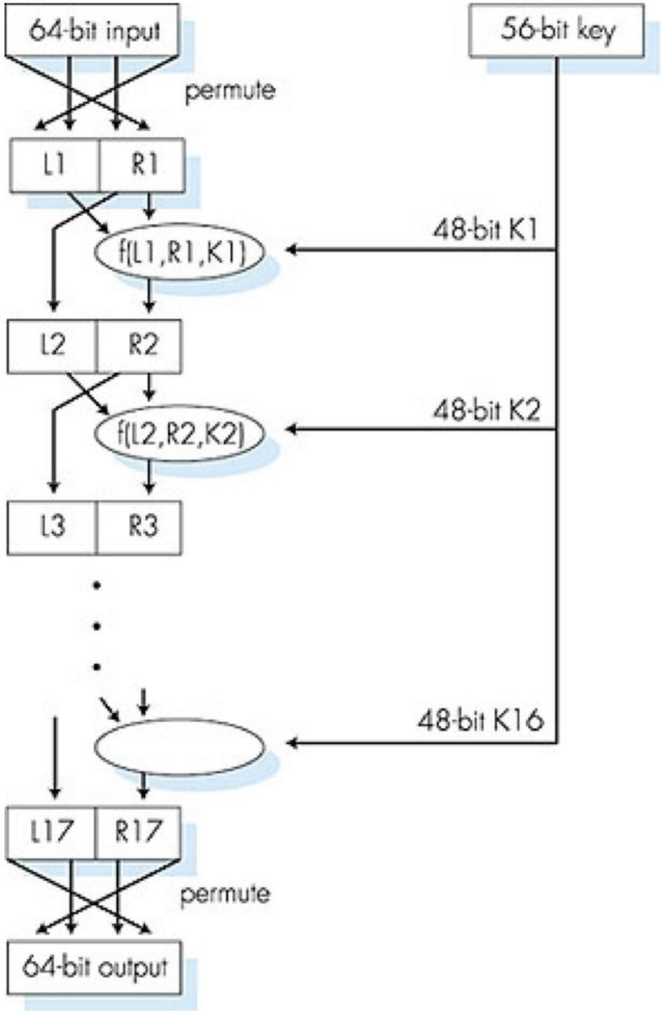
Letter

DES

- Data Encryption Standard
- 56-bit symmetric key, 64-bit plaintext input
- Block cypher
- How secure is DES?
    - brute force in less than a day
    - no known good analytic attack
- Make DES more secure
    - 3DES: encrypt 3 times with 3 different keys

DES

- 16 identical “rounds” of function application, each using different 48-bit key



AES

- Advanced Encryption Standard
  - Replaced DES
- Process data in 128-bit blocks
- 128, 192, or 256 bit keys
- Brute force decryption taking 1 sec on DES, takes 149 trillion years for AES

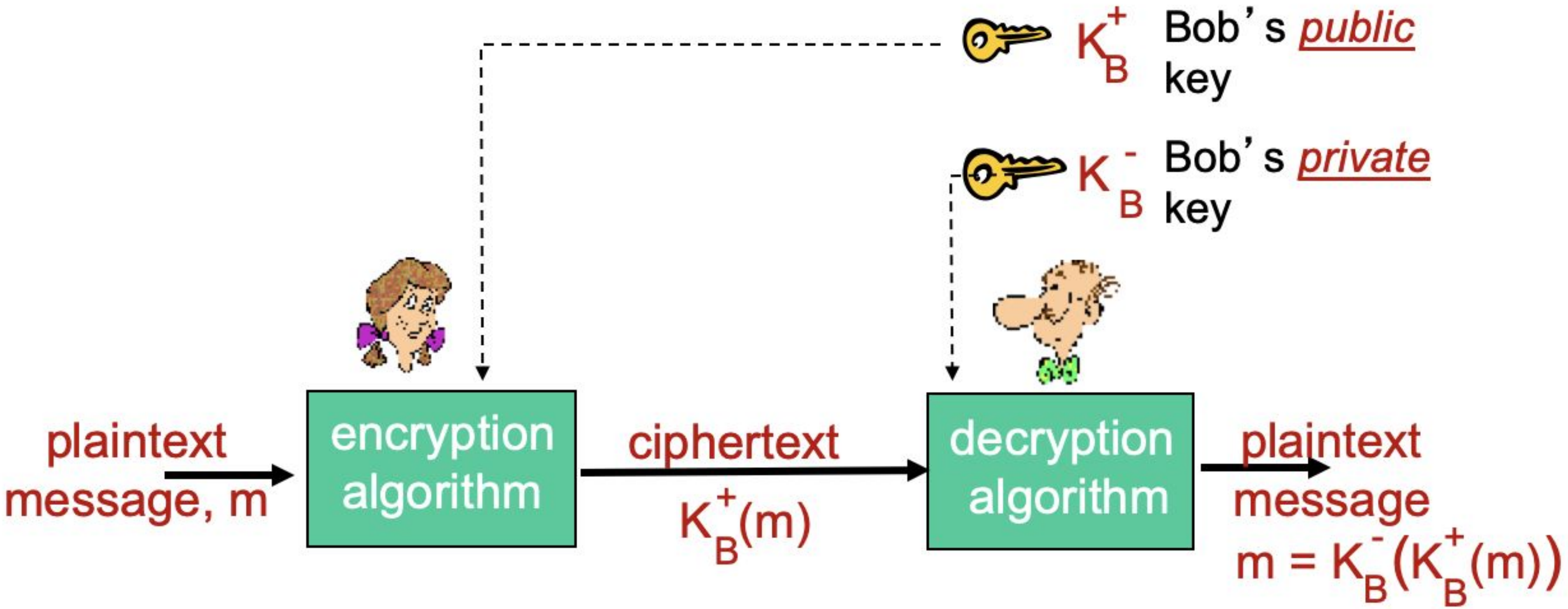
One issue with symmetric key cryptography

- Sender and receiver need to know shared secret key

Public key cryptography

- Sender and receiver don’t share the same key
- Public key is known to all
- Private key is only known to receiver
- E.g. Diffie-Hellman, RSA

Public key cryptography



Requirements

1. need $K_B^+(\cdot)$ and $K_B^-(\cdot)$ such that

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

2. given public key K_B^+, it should be impossible to compute private key K_B^-

Modular arithmetic

- x mod n = remainder of x when divided by n
- Facts:

$$[(a \bmod n) + (b \bmod n)] \bmod n = (a+b) \bmod n$$
$$[(a \bmod n) - (b \bmod n)] \bmod n = (a-b) \bmod n$$
$$[(a \bmod n) * (b \bmod n)] \bmod n = (a*b) \bmod n$$

$$(a \bmod n)^d \bmod n = a^d \bmod n$$

RSA

- Message: just a bit pattern (sequence of 0’s and 1’s)
- Bit pattern can be uniquely represented by an integer
- Thus, encrypting a message is equivalent to encrypting a integer number
- E.g m = 10010001. This message is uniquely represented by the decimal number 145
- To encrypt m, we encrypt the corresponding number, which gives a new number (the ciphertext)

RSA - public/private key pair

1. choose two large prime numbers p, q. (e.g., 1024 bits each)
2. compute $n = pq$, $z = (p-1)(q-1)$
3. choose e (with $e<n$) that has no common factors with z (e, z are "relatively prime").
4. choose d such that $ed-1$ is exactly divisible by z. (in other words: $ed \bmod z = 1$).
5. *public* key is $\underbrace{(n,e)}_{K_B^+}$. *private* key is $\underbrace{(n,d)}_{K_B^-}$.

RSA - encryption and decryption

0. given (n,e) and (n,d) as computed above

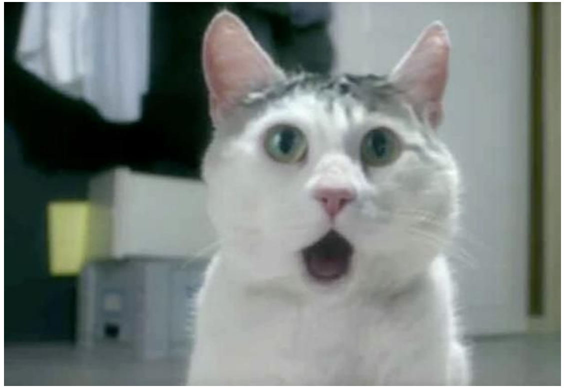
1. to encrypt message m $(<n)$, compute

   $c = m^{e} \bmod n$

2. to decrypt received bit pattern, c, compute

   $m = c^{d} \bmod n$

magic happens! $m = \underbrace{(m^{e} \bmod n)}_{c}{}^{d} \bmod n$



Bob chooses $p=5$, $q=7$. Then $n=35$, $z=24$.

$e=5$ (so e, z relatively prime).

$d=29$ (so $ed-1$ exactly divisible by z).

encrypting 8-bit messages.

	bit pattern	m	m^e	$c = m^e \bmod n$
encrypt:	00001100	12	24832	17

	c	c^d	$m = c^d \bmod n$
decrypt:	17	481968572106750915091411825223071697	12

RSA - another important property

$$\underbrace{K_B^-(K_B^+(m))}_{\substack{\text{use public key first,} \\ \text{followed by} \\ \text{private key}}} = m = \underbrace{K_B^+(K_B^-(m))}_{\substack{\text{use private key} \\ \text{first, followed by} \\ \text{public key}}}$$

result is the same!

RSA - why is it secure?

- Suppose you know Bob's public key (n,e). How hard is it to determine d?
- Essentially need to find factors of n without knowing the two factors p and q
- Factoring a big number is hard

RSA in practice

- Exponentiation in RSA is computationally intensive
- DES is at least 100 times faster than RSA
- Use public key crypto to
  - establish secure connection
  - establish symmetric session key for data encryption

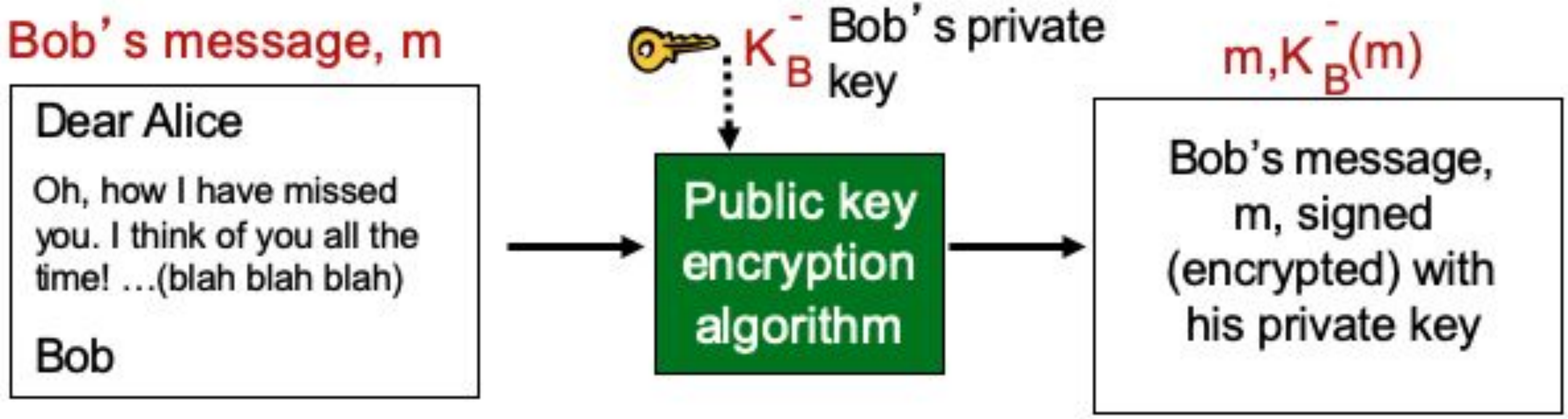
Message integrity

Digital signature

- Analogous to hand-written signatures
- Sender (Bob) digitally signs document, establishing he is the document owner/creator
- Verifiable and non-forgeable: recipient (Alice) can prove to someone that Bob, and no one else, must have signed document

Digital signature

- How about Bob signs m by encrypting with his private key?




ALICE, HOW TO VERIFY THIS MSG?
imgflip.com
memecrunch.com

Digital signature

- Alice receives msg m with its signature
- Apply Bob’s public key on the signature
- Compare if you have msg m as output. If so
  - Bob must have signed this msg
  - No one else signed m
  - Bob signed m and not other msg m’

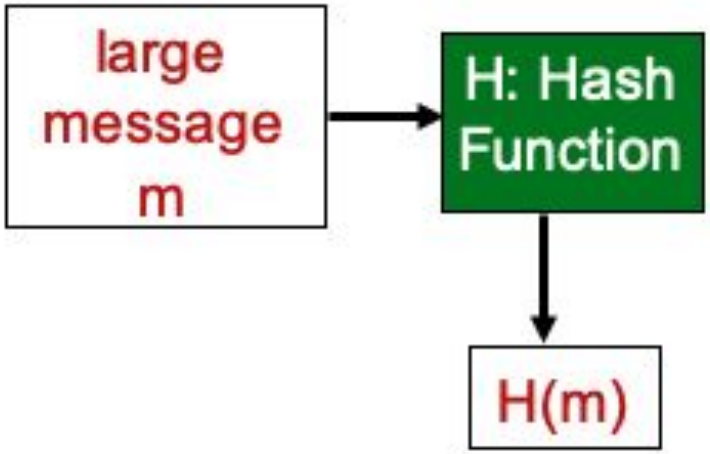
Bob cannot refute this

- Alice can take m and its signature to court and prove that Bob signed m

But, signing the entire m is...

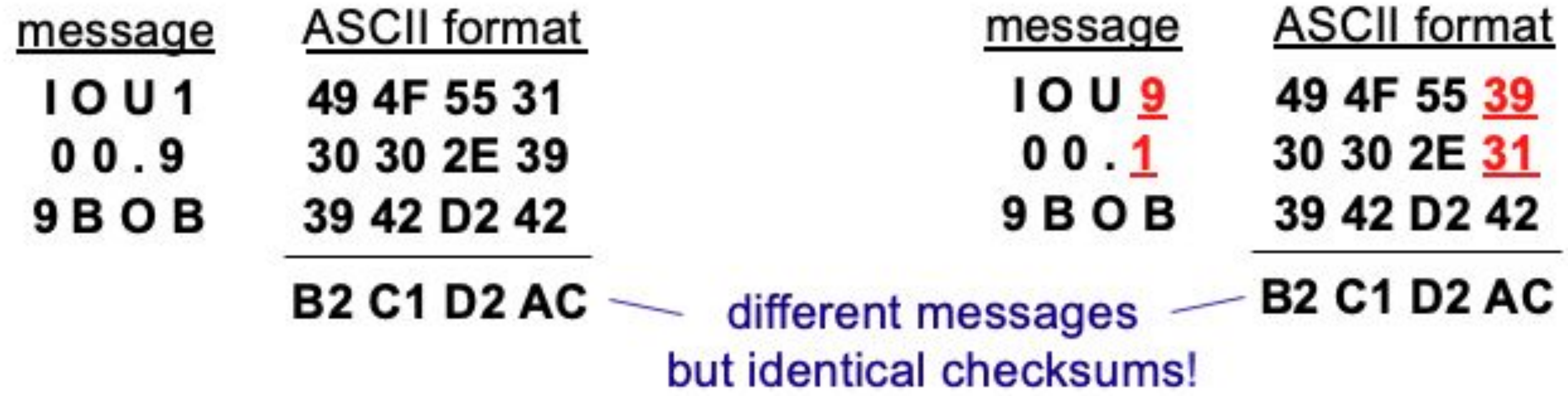
Message digest

- Use hash function!
- Produce fixed-size message digest
- Given message digest x, computationally infeasible to find m such that x = H(m)



Poor hash function - Internet checksum

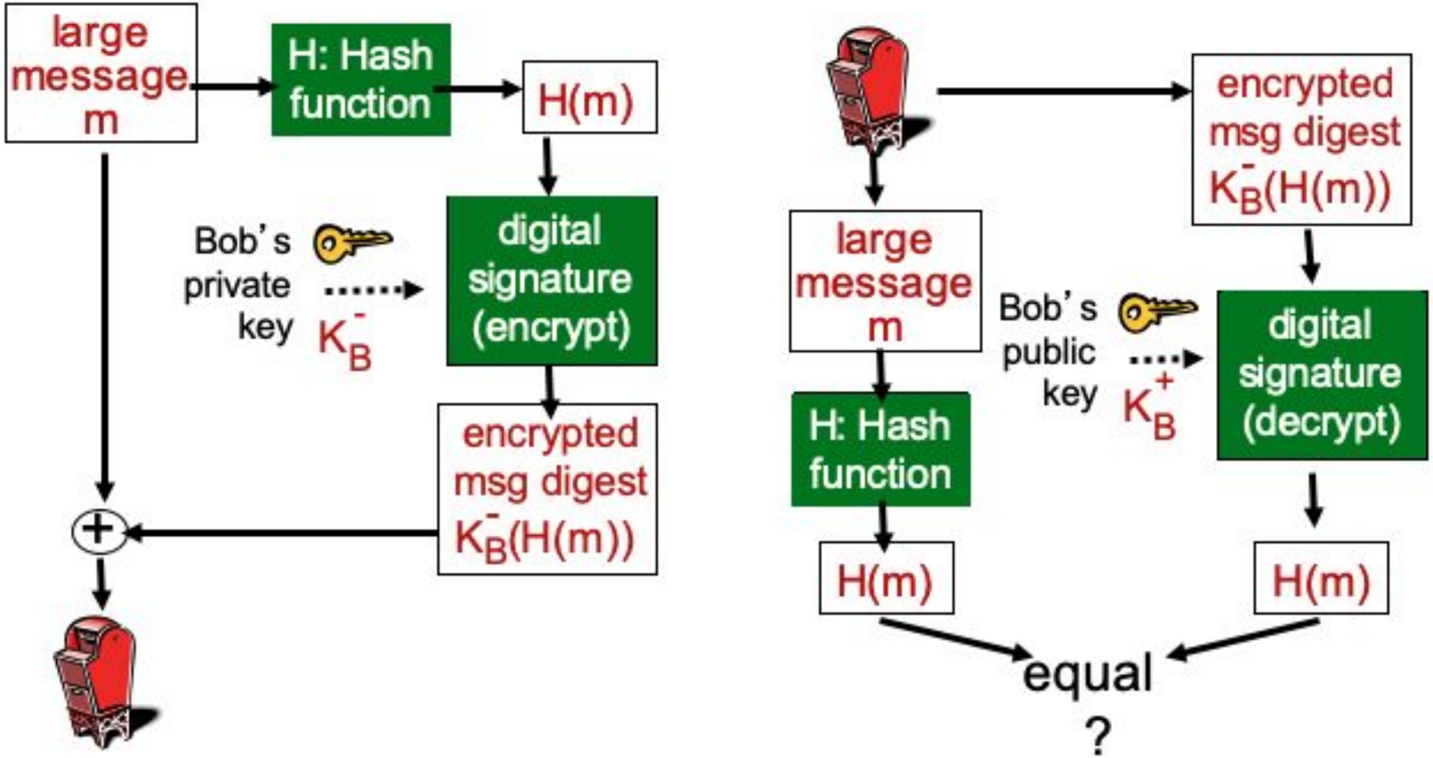
- Produce fixed length digest (16-bit)
- Very easy to find another message with the same hash



Good hash functions

- MD5
    - Compute 128-bit message digest
    - No longer considered secure
- SHA-1
    - 160-bit message digest
    - No longer considered secure
- SHA-2, SHA-3
    - Good to use

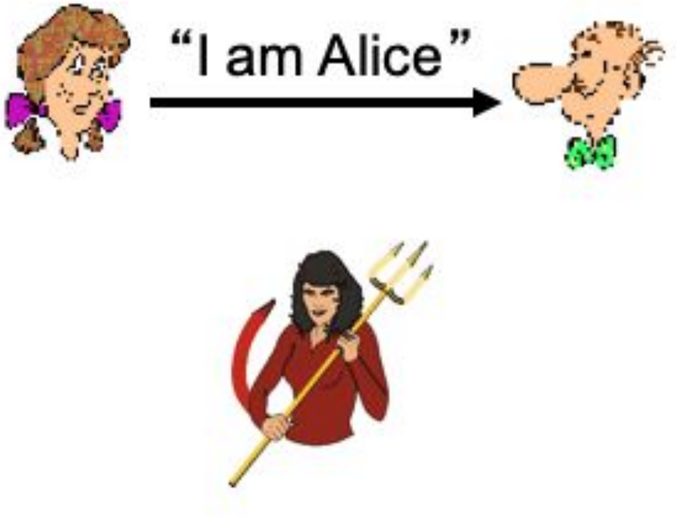
Digital signature = signed message digest

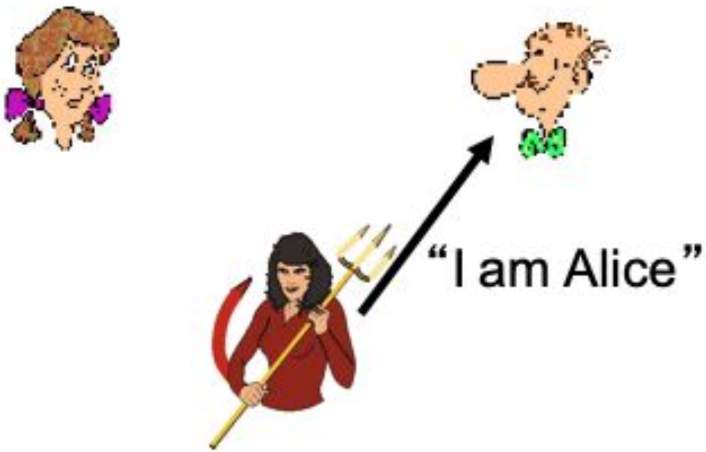


Authentication

Authentication - AP1.0

- Bob wants Alice to “prove” her identity to him
- Protocol AP1.0: Alice says “I am Alice”



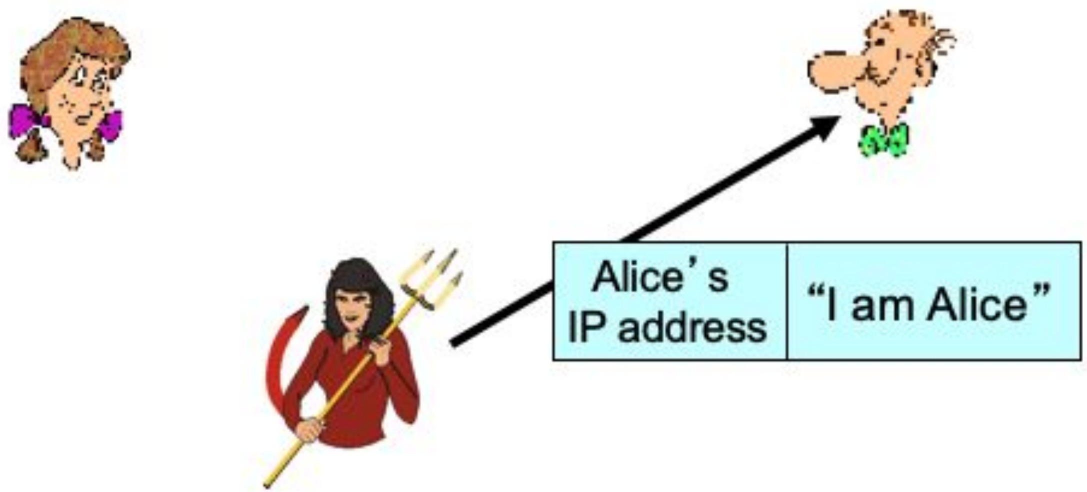


Authentication - AP2.0

- Protocol AP2.0: Alice says “I am Alice” in an IP packet containing her source IP address

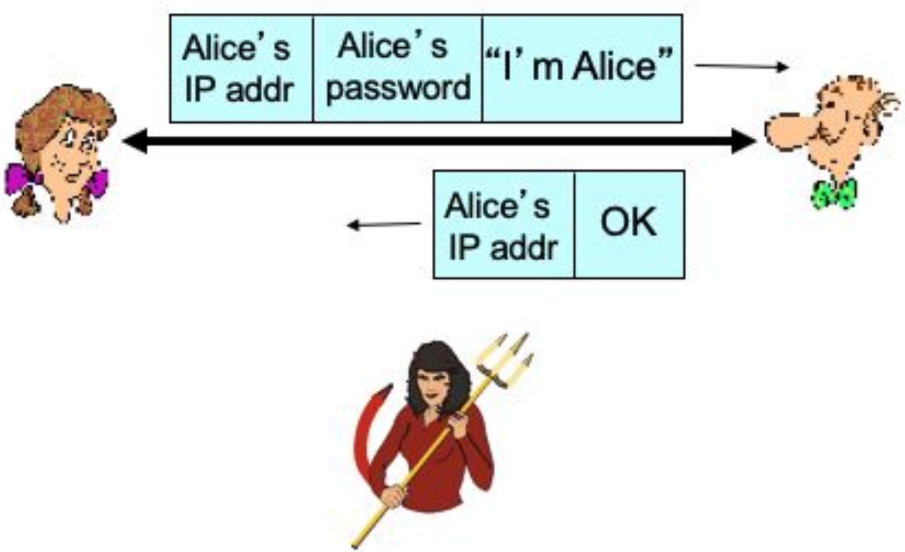
Authentication - AP2.0

- IP address is very easy to spoof



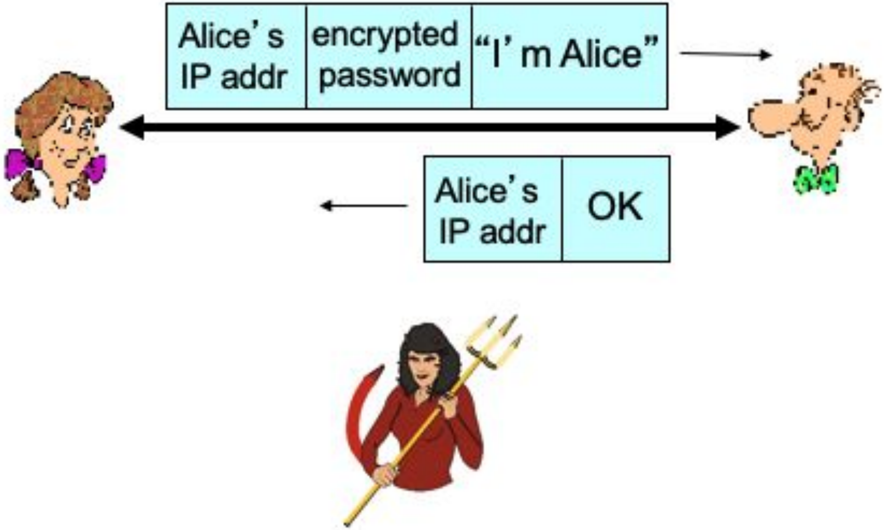
Authentication - AP3.0

- Protocol AP3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



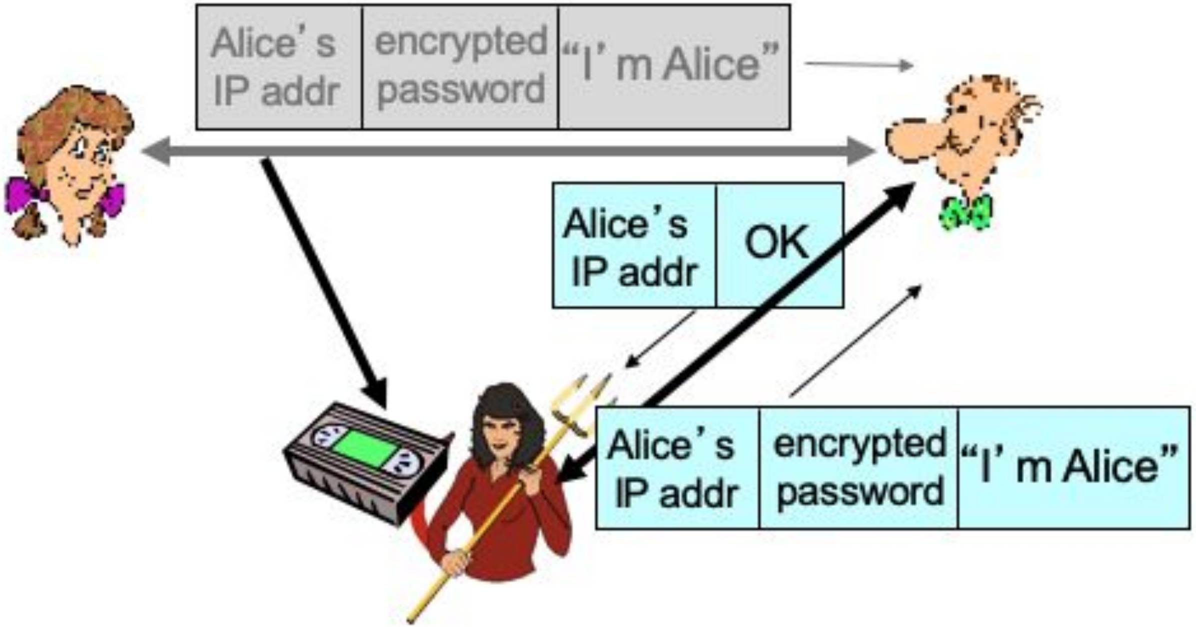
Authentication - AP3.1

- Protocol AP3.1: Alice says "I am Alice" and sends her ***encrypted*** secret password to "prove" it.



Authentication - AP3.1

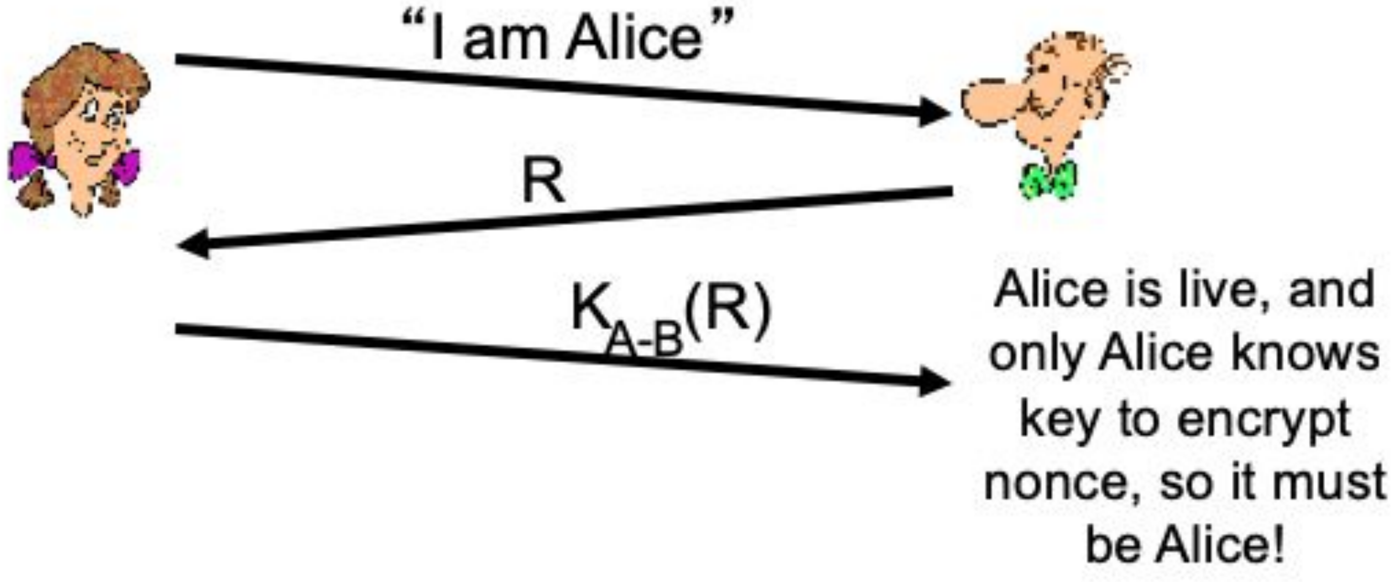
- Trudy records Alice’s packet and later plays it back



Authentication - AP4.0

- Need to avoid playback attack
- **nonce**: number ® used only *once-in-a-lifetime*
- Protocol AP4.0: to prove Alice “live”, Bob sends Alice nonce, R. Alice must return R encrypted with shared secret key

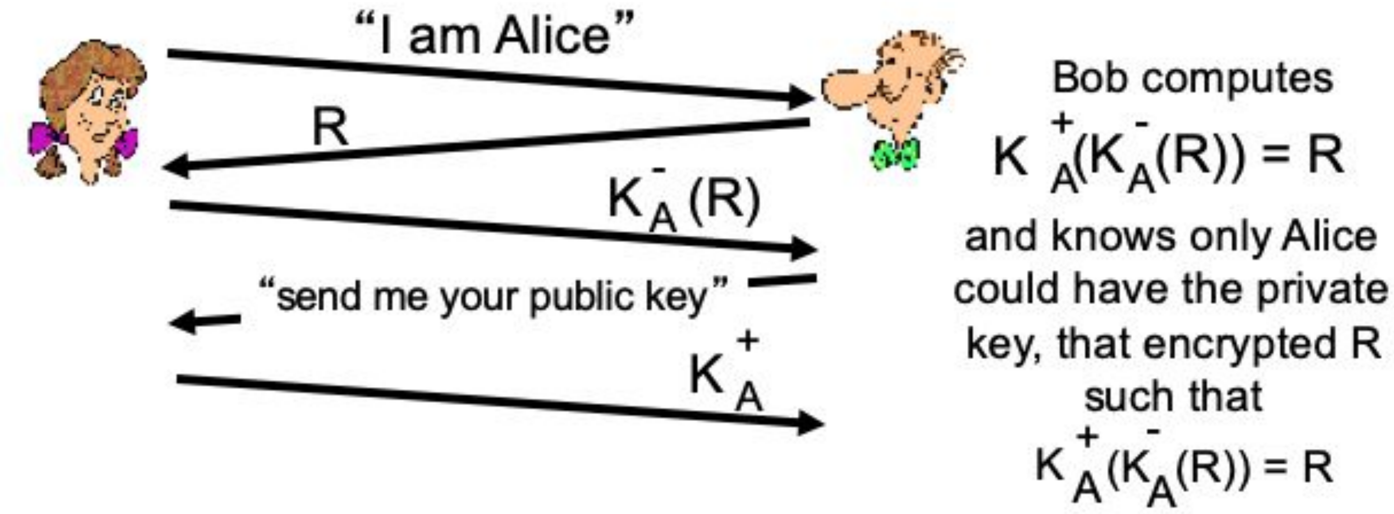
Authentication - AP4.0



AP4.0 requires shared symmetric key

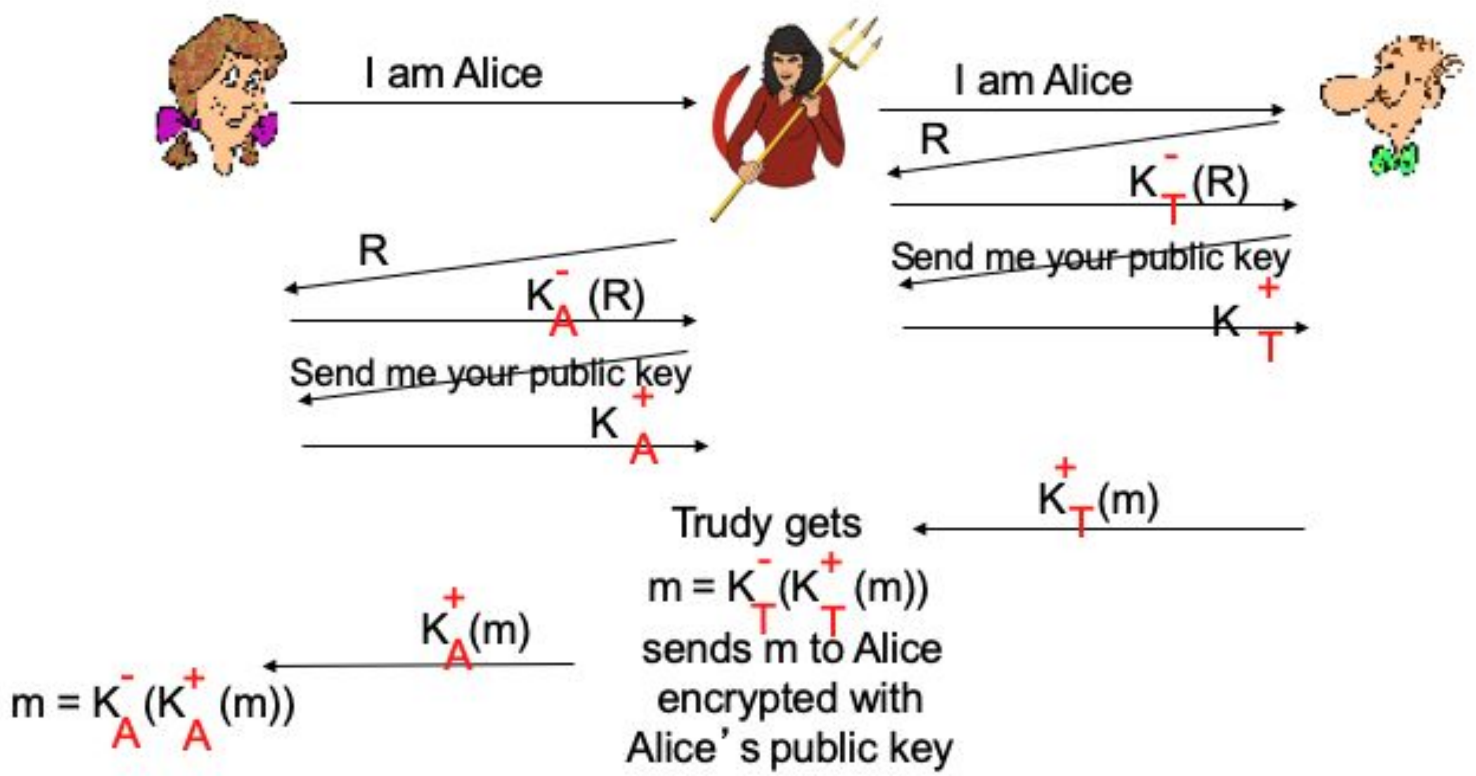
Authentication - AP5.0

- Use nonce + public key cryptography



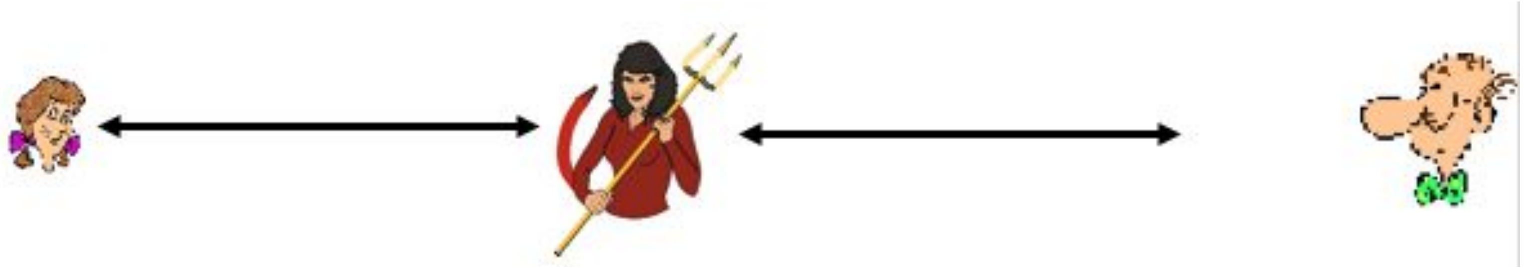
Do you like AP5.0?

Man (or woman) in the middle attack



Man (or woman) in the middle attack

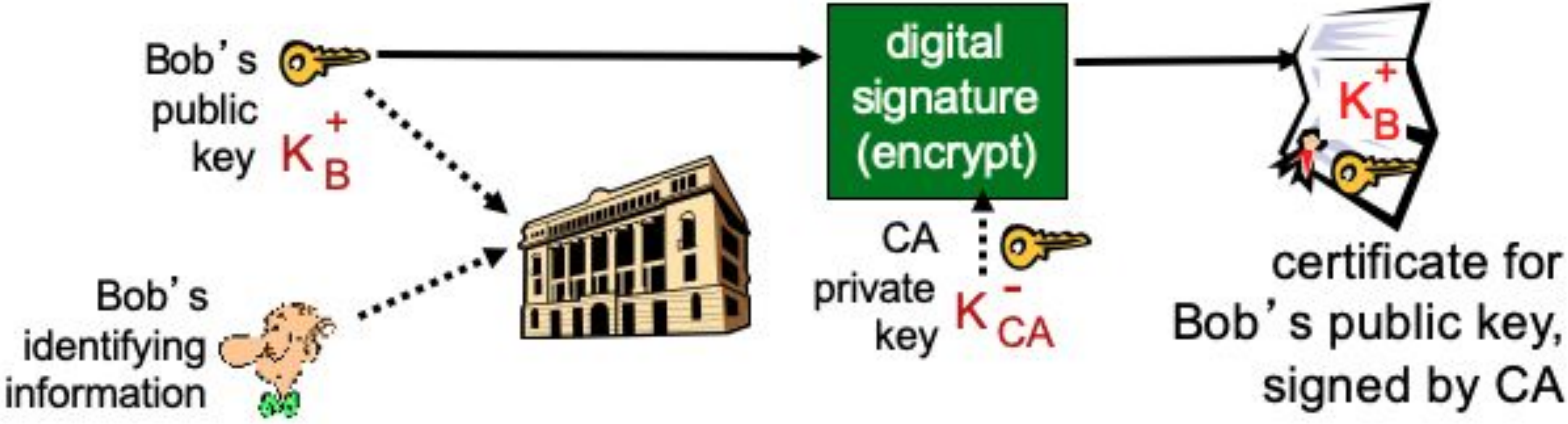
- Trudy poses as Alice (to Bob) and as Bob (to Alice)
- Difficult to detect
    - Bob receives everything that Alice sends, and vice versa.
- Problem is that Trudy receives all messages as well!



Certification authorities (CA)

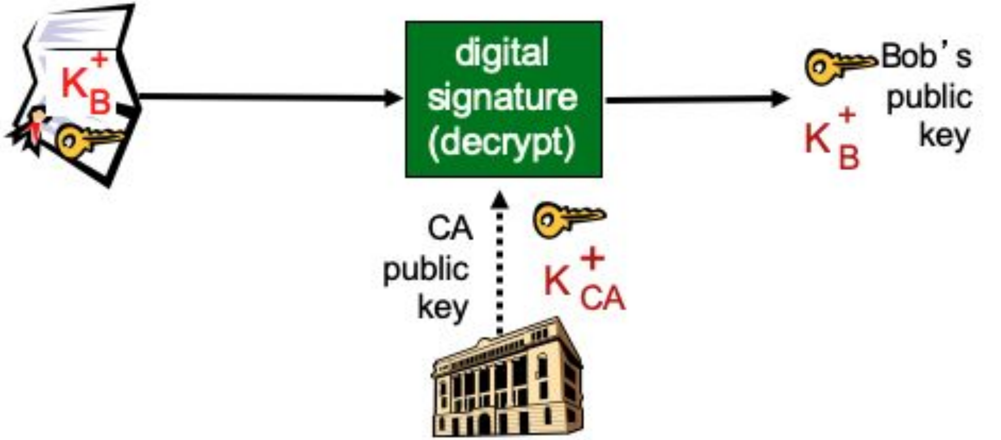
- CA binds public key to particular entity E
- E register its public key with CA
    - E provides “proof of identity” to CA
    - CA creates certificate binding E to its public key
    - Certificate containing E’s public key, digitally signed by CA (aka CA says “this is E’s public key”)

Certification authorities (CA)



Certification authorities (CA)

- When Alice wants Bob’s public key
  - gets Bob’s certificate
  - apply CA’s public key to Bob’s certificate



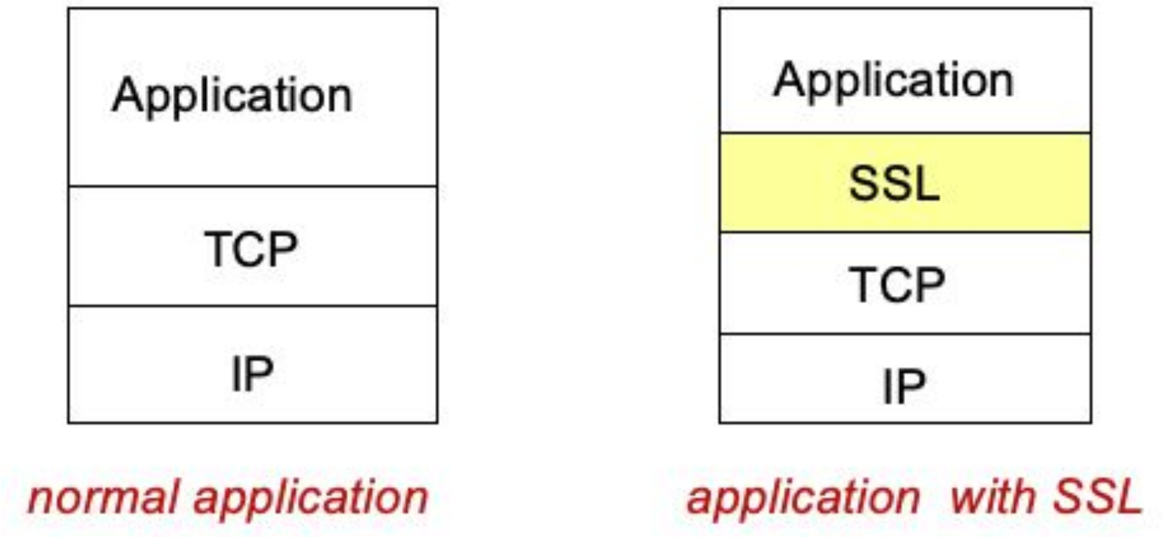
Securing TCP connection: SSL

SSL: secure socket layer

- Widely deployed security protocol
  - Supported by almost all browsers, web servers
  - https
  - Billions $/year over SSL
- Provides: confidentiality, integrity, authentication
- Variation - TLS: transport layer security
- Available to all TCP applications

SSL and TCP/IP

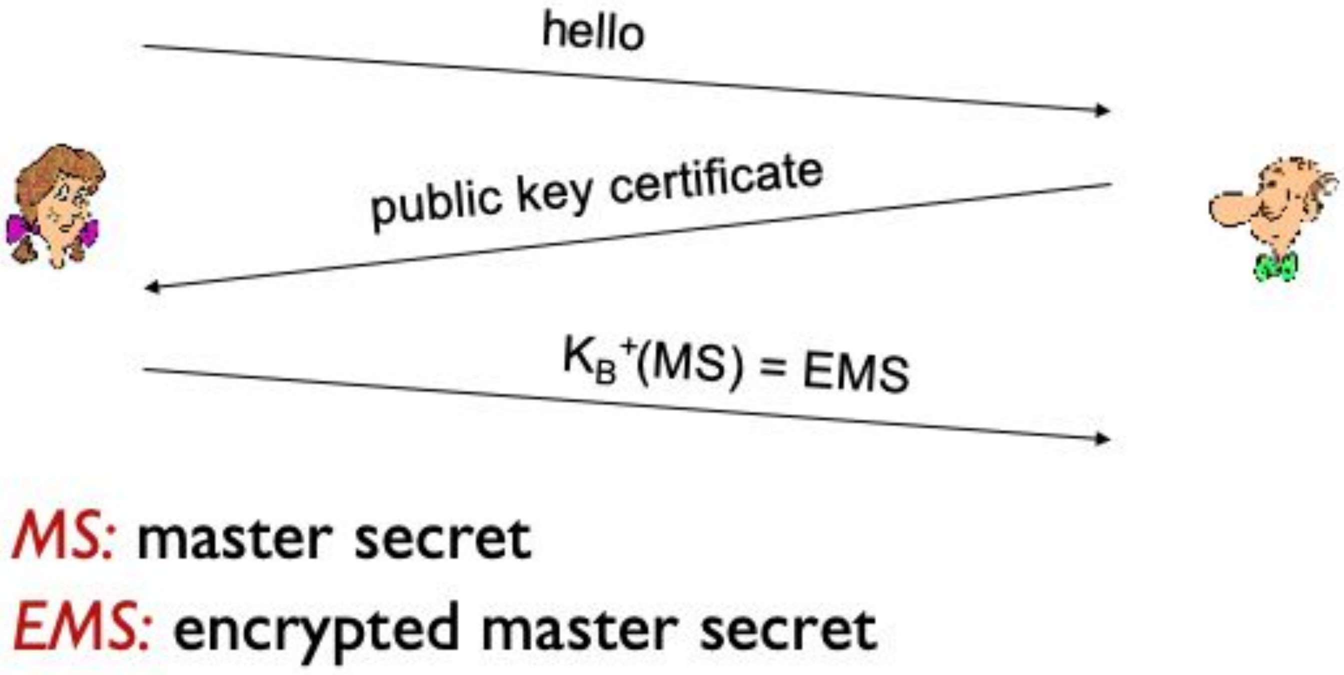
- SSL provides application programming interface (API)



Toy SSL

- ***Handshake***: Alice and Bob use their certificates, private keys to authenticate each other and exchange shared secret
- ***Key derivation***: Alice and Bob use shared secret to derive set of keys
- ***Data transfer***: data to be transferred is broken up into series of records
- ***Connection closure***: special msg to close connection

Toy SSL: handshake

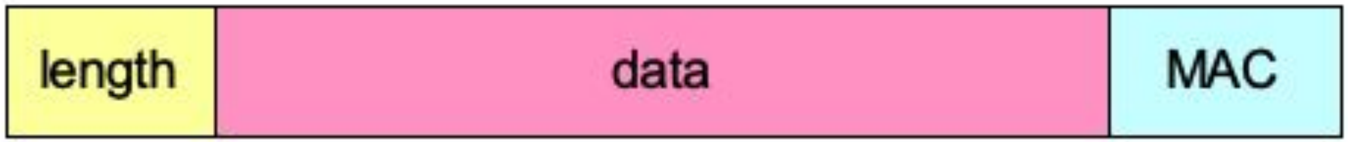


Toy SSL: key derivation

- Considered bad to use same key for more than one cryptographic operation
  - Use different keys for message authentication code (MAC) and encryption
- Four keys
  - Two client => server: encryption key and MAC key
  - Two server => client: encryption key and MAC key

Toy SSL: data records

- Why not encrypt data in constant stream?
  - Where would we put the MAC?
- Break stream in series of records
  - Each record carries a MAC
  - Receiver can act on each record as it arrives



Toy SSL isn’t complete

- How long are fields?
- Which encryption protocols?
- Want negotiation?
  - allow client and server to support different encryption algorithms
  - allow client and server to choose together specific algorithm before data transfer

SSL cipher suite

- Cipher suite
  - Public key algorithms
  - Symmetric encryption algorithms
  - MAC algorithm
- Negotiation: client and server agree on cipher suite

common SSL symmetric ciphers

- DES – Data Encryption Standard: block
- 3DES – Triple strength: block
- AES – Advanced Encryption Standard: block

SSL Public key encryption

- RSA

Questions?