Chapter 8 Security in Computer Networks

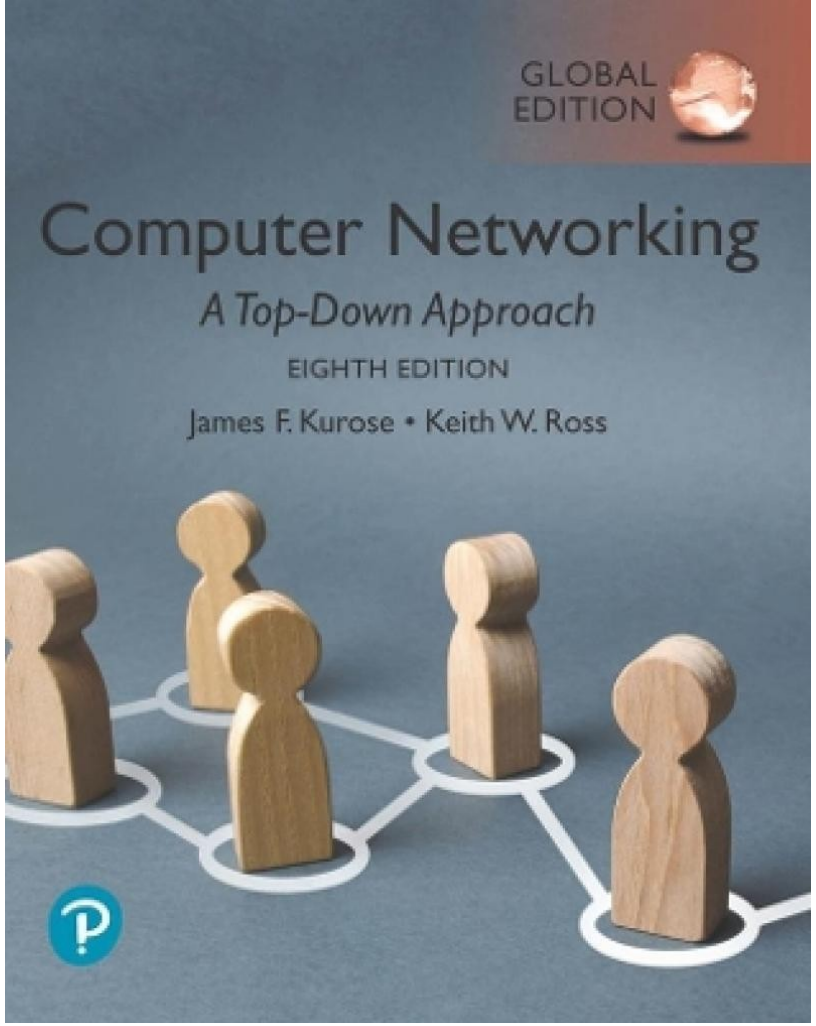
A note on the use of these PowerPoint slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

For a revision history, see the slide note for this page.

Thanks, and enjoy! JFK/KWR



Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Security: overview

Chapter goals:

- understand principles of network security:
  - cryptography and its *many* uses beyond "confidentiality"
  - authentication
  - message integrity
- security in practice:
  - firewalls and intrusion detection systems
  - security in application, transport, network, link layers

Chapter 8 outline

8.1 What is network security?

8.2 Principles of cryptography

8.3 Message integrity and digital signatures

~~8.4 Authentication~~

8.5 Securing e-mail

8.6 Securing TCP connections: TLS

~~8.7 Network layer security: IPsec~~

~~8.8 Security in wireless and mobile networks~~

8.9 Operational security: firewalls and IDS

Network security: Security properties

confidentiality: only the intended receiver should read messages

data integrity: the receiver wants to ensure that data have not been altered

message authenticity, data origin authentication:

1. confirm that the message came from the stated sender (sender authenticity)
2. confirm that the message has not been changed (integrity)

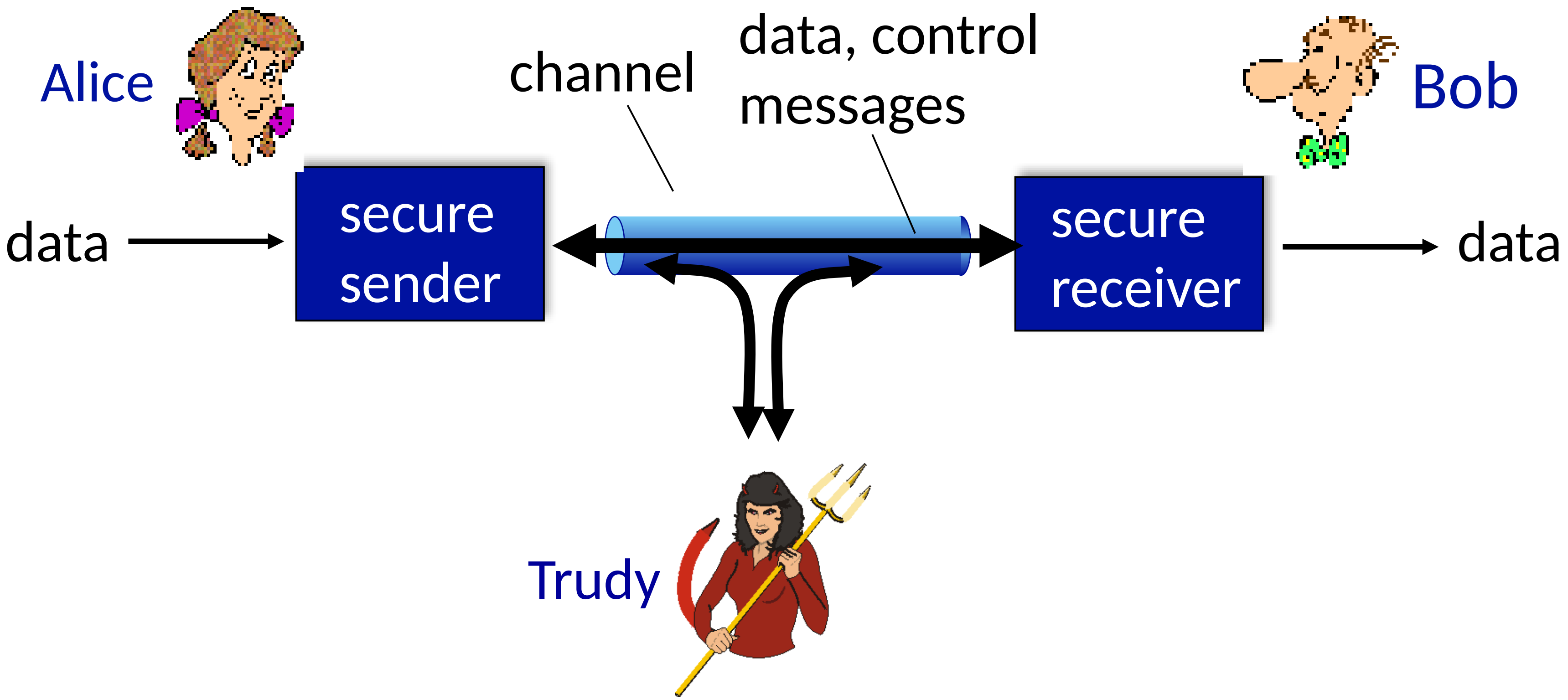
entity authentication:

- Alice and Bob want to confirm the identities of each other during a session

availability: services must be accessible and available to users

Friends and enemies: Alice, Bob, Trudy

- well-known in network security world
- Bob, Alice (lovers!) want to communicate "securely"
- Trudy (intruder) may intercept, delete, add messages



Types of attacks (1)

- eavesdropping: intercept messages
  - confidentiality breach
- data modification
  - integrity breach
  - authenticity breach
- impersonation, spoofing
  - authenticity breach
- replay attacks
  - “sort of” authenticity breach

Types of attacks (2)

System-oriented attacks:

- denial of service: prevent service from being used by others (e.g., by overloading resources)

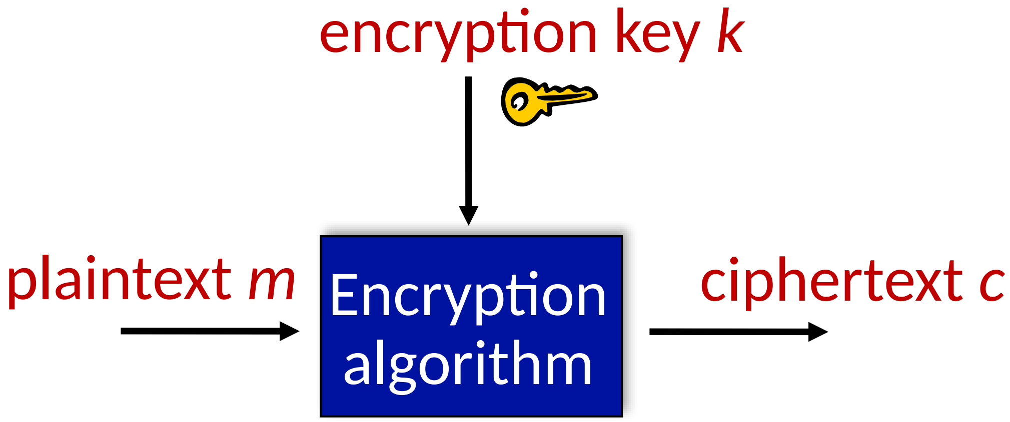
availability breach

- hijacking: “take over” ongoing connection by removing sender or receiver, inserting himself in place

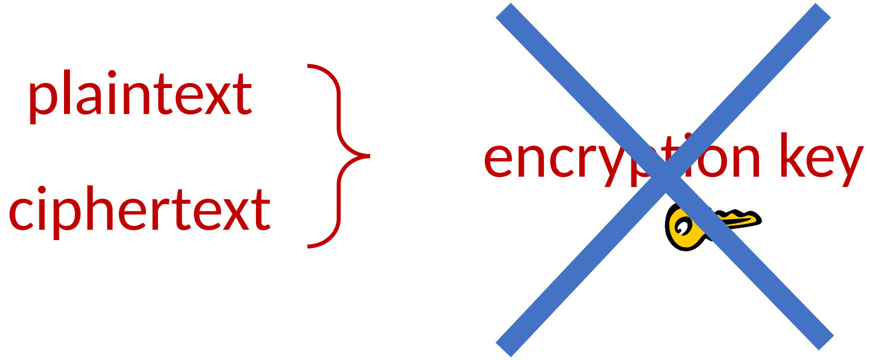
Chapter 8 outline

The language of cryptography

- Plaintext *m*
- Encryption key *k*
- Ciphertext *c*
- Cryptographic algorithms
  - Encryption algorithm E
  - Decryption algorithm D



$$c = E_k(m)$$



The language of cryptography

- Symmetric key cryptography
  - Aka. secret key cryptography

- Asymmetric key cryptography
  - Aka. public key cryptography

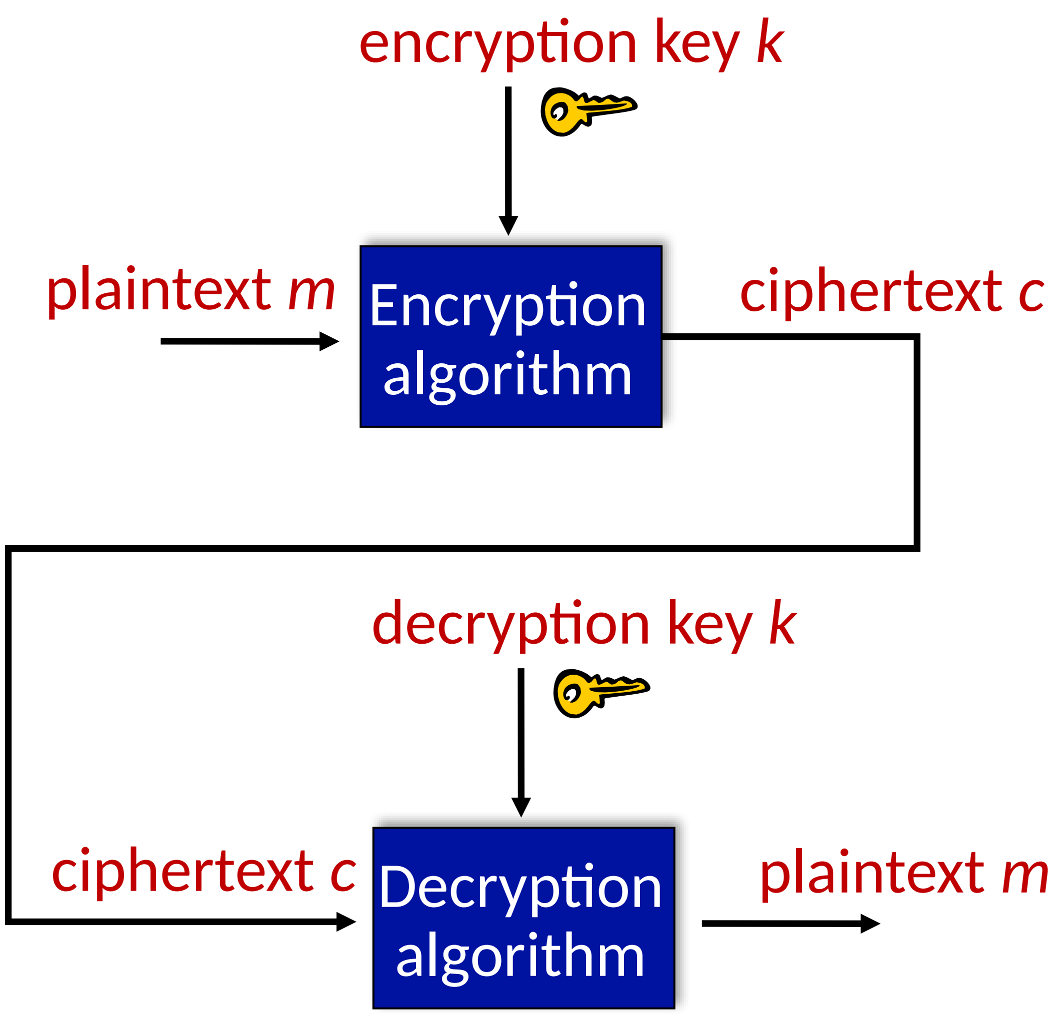
Symmetric key cryptography

Encryption key = decryption key

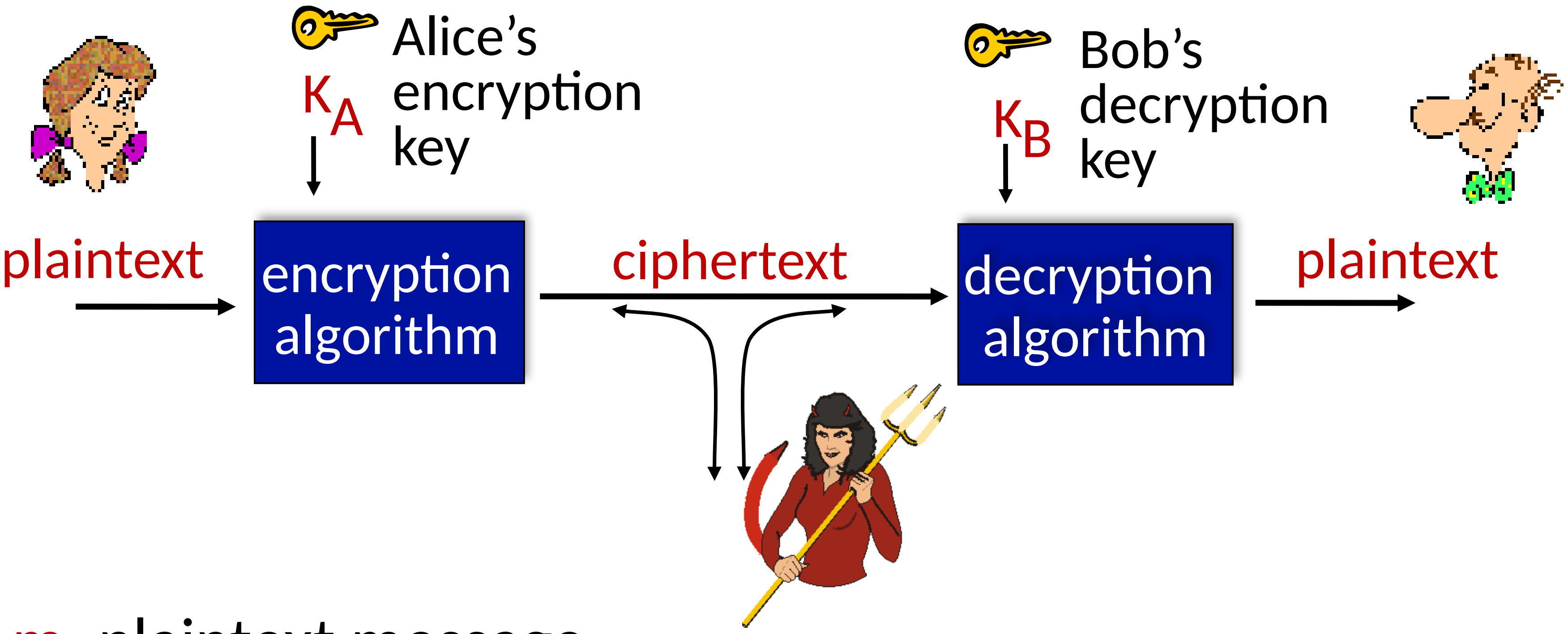
- Symmetric key, secret key

- Plaintext m
- Ciphertext c
- Cryptographic algorithms
  - Encryption algorithm E
  - Decryption algorithm D

$$c = E_k(m)$$
$$m = D_k(E_k(m))$$
$$m = D_k(c)$$



Adversary, attacker, cryptanalysis



m: plaintext message

$c=E_K(m)$: ciphertext, encrypted with key K_A

$m = D_K(E_K(m))$

Breaking an encryption scheme

- brute force: search through all keys
  - key space

- known-plaintext attack: Trudy has plaintext corresponding to ciphertext
  - *e.g.*, in monoalphabetic cipher, Trudy determines pairings for a,l,i,c,e,b,o,
- chosen-plaintext attack: Trudy can get ciphertext for chosen plaintext

Caesar cipher

- The rotation equivalent to the modulus operation:
  - Encryption: $C = P + K \bmod 26$
  - Decryption: $P = C - K \bmod 26$

```
Plaintext:  bob. i love you. al
Ciphertext: dqd. k nqxg aqw. cnk
```

- Trivial complexity
  - With only 26 possible keys to try out, an exhaustive key search is easy
  - Thus, no security

Monoalphabetic ciphers

- Each letter in the plaintext is mapped to another letter in the ciphertext
- The mapping is always the same

Encryption key:

```
   plaintext letter:  abcdefghijklmnopqrstuvwxyz

  ciphertext letter:  mnbvcxzasdfghjklpoiuytrewq
```

Encryption:

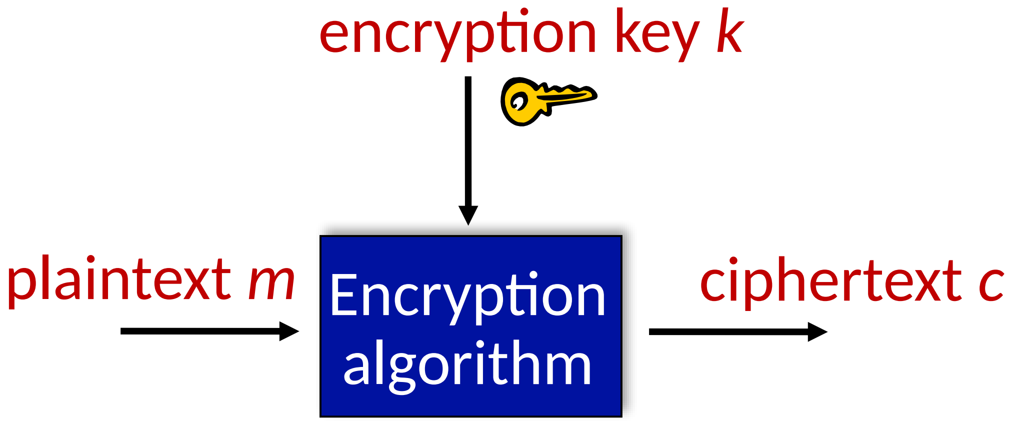
```
Plaintext:  bob. i love you. alice
Ciphertext: nkn. s gktc wky. mgsbc
```

Mapping 26 letters to 26 letters:

26! = 403 291 461 126 605 635 584 000 000 possible keys

Symmetric key crypto: Block ciphers

- Two paired algorithms, one for encryption and the other for decryption
- Deterministic
- Fixed block sizes
  - plaintext /ciphertext block of size *n* bits (e.g. 128 bits)
  - key size *k* bits (e.g. 128, 256 bits)
- Important properties:
  - Cannot tell the difference between ciphertext block and random bits
  - One-way-property
  - Avalance effect



The avalanche effect

- Randomization property
- A desirable property of cryptographic algorithms, typically block ciphers and cryptographic hash functions
- If flipping a single input bit, the output changes significantly (e.g., half the output bits flip)
  - Both encryption (plaintext, key) and decryption (ciphertext, key)
- Very important property for block ciphers and hash functions
- A poor avalanche property can make a cipher vulnerable to cryptanalysis

Symmetric key crypto

DES: Data Encryption Standard

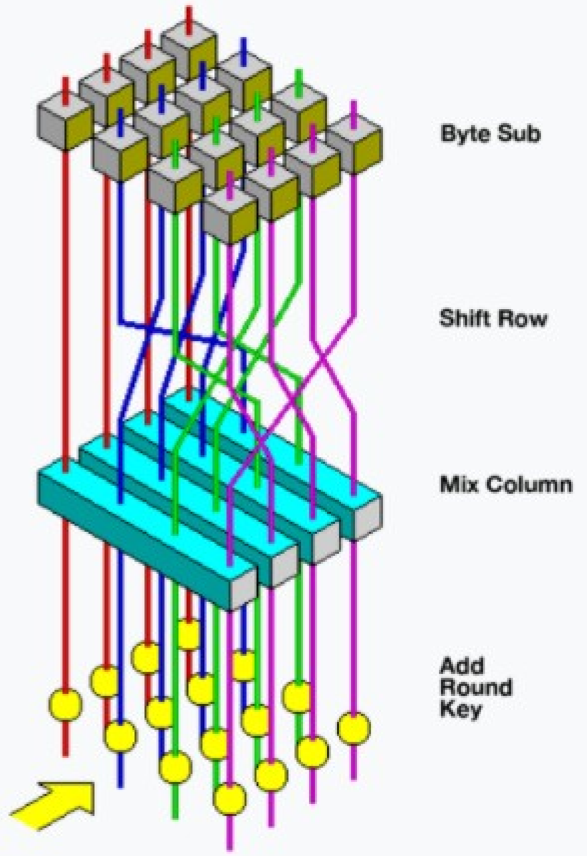
- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64-bit block size

$$2^{56} = 72{,}057{,}594{,}037{,}927{,}936$$

- how secure is DES?
  - DES Challenge: 56-bit-key-encrypted, phrase decrypted (brute force) in less than a day
  - no known good analytic attack
- making DES more secure:
  - 3DES: encrypt 3 times with 3 different keys

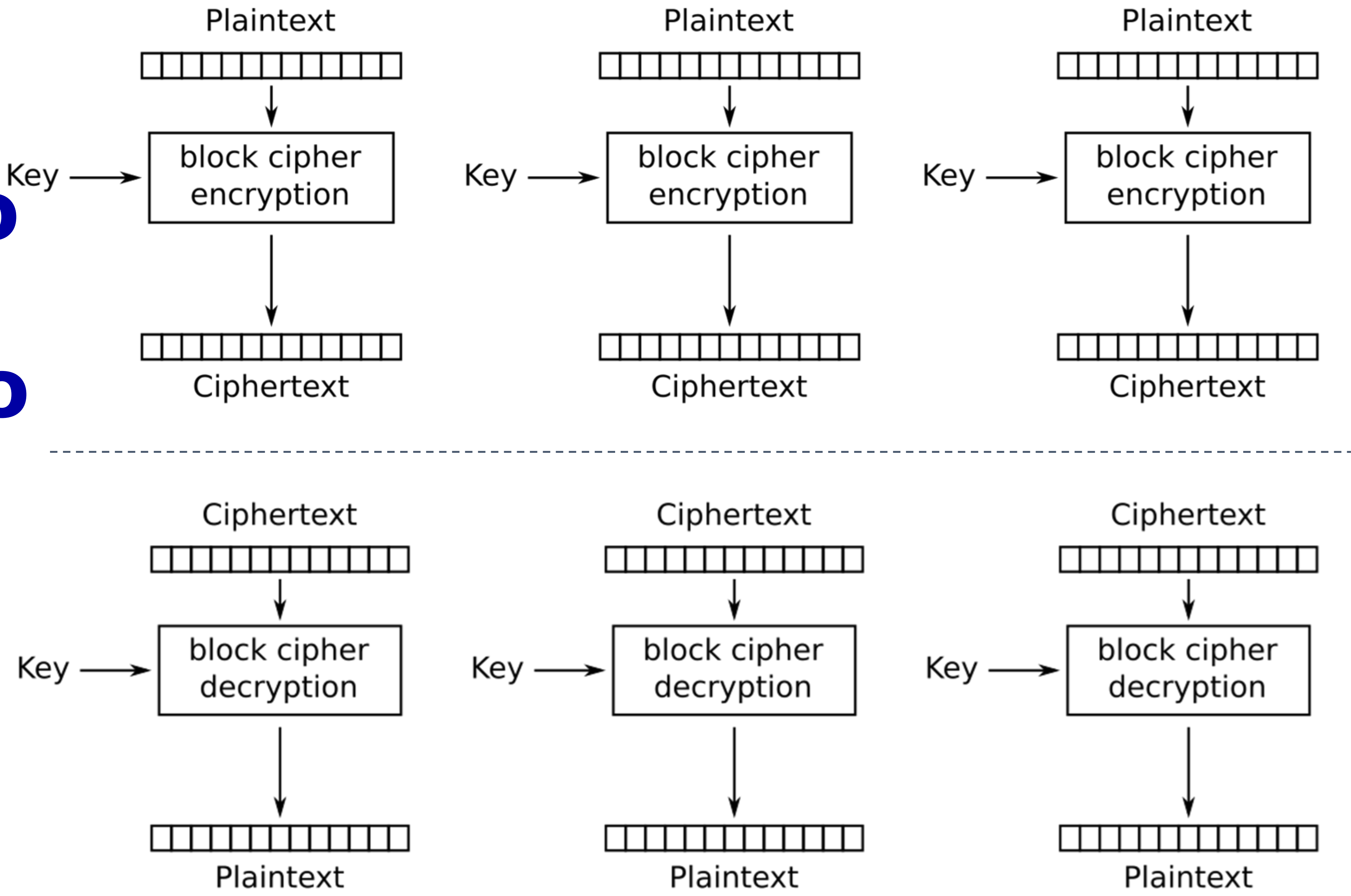
AES: Advanced Encryption Standard

- symmetric-key NIST standard, replaced DES (Nov 2001)
- also known by its original name **Rijndael**
- processes data in 128-bit blocks
- 128-, 192-, or 256-bit keys
- brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

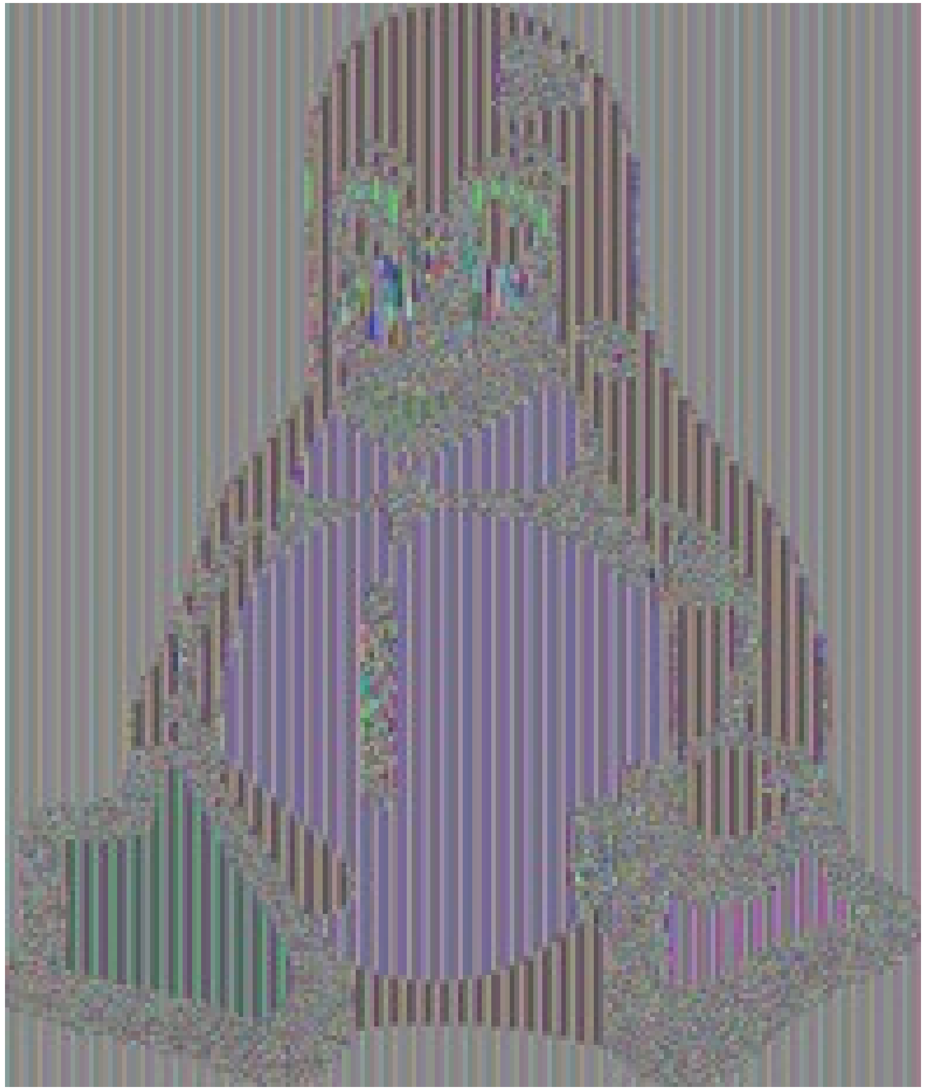


Visualization of the AES round function

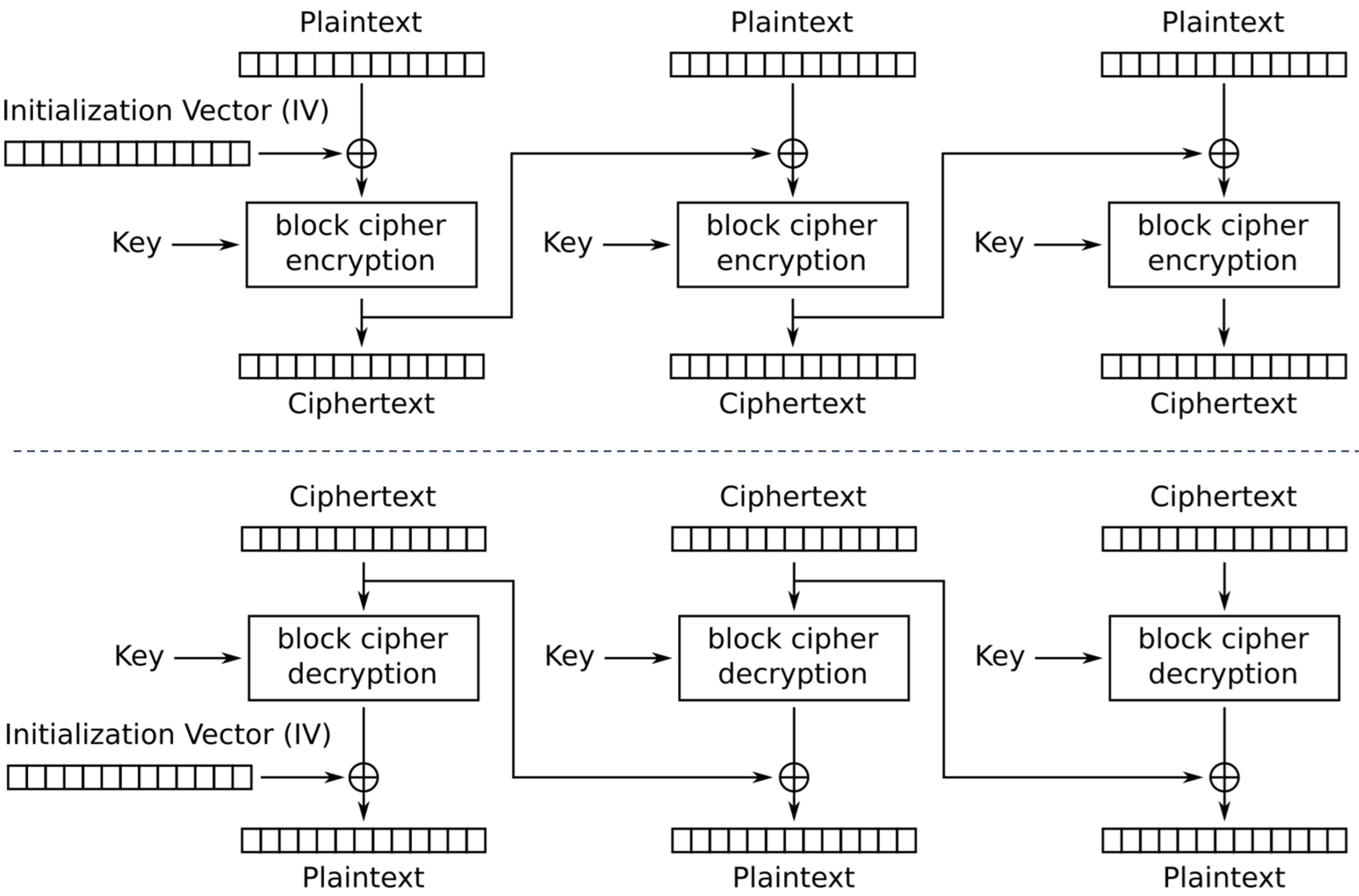
Electronic codebook (ECB)



ECB can reveal pattern information

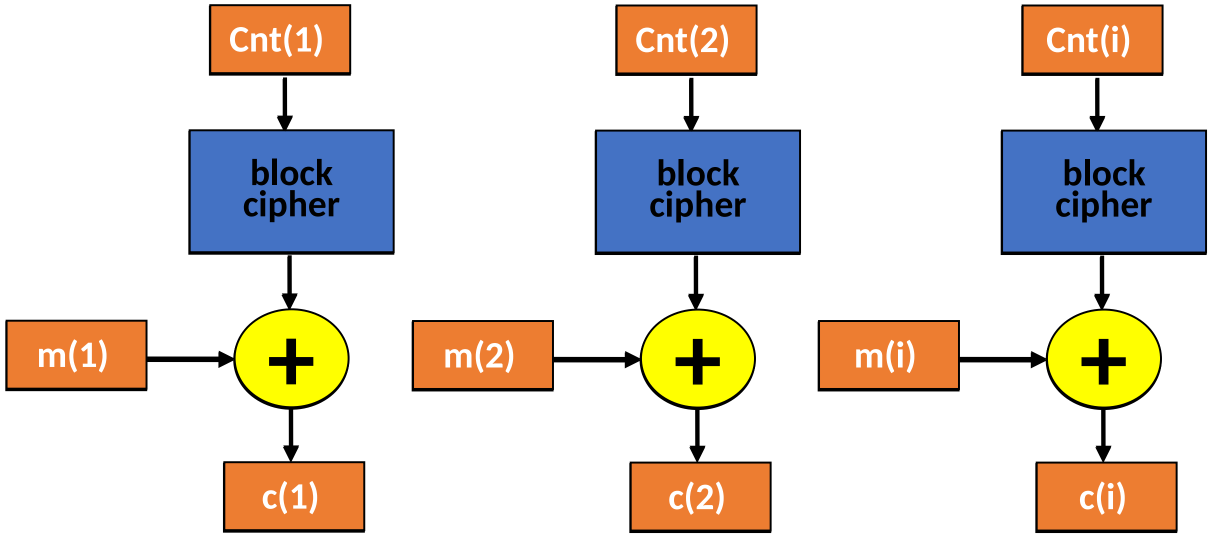


Cipher block chaining

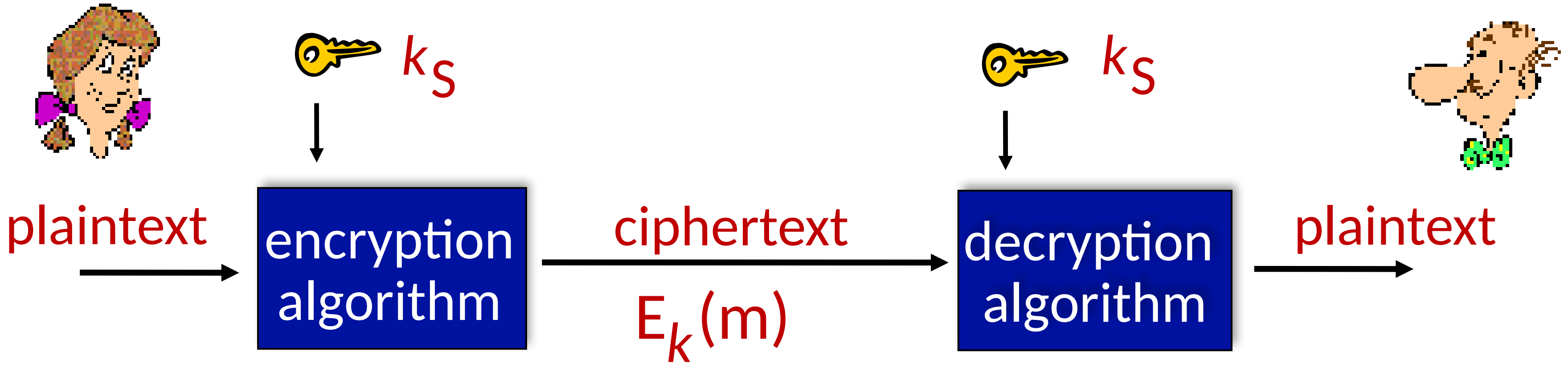


Counter mode (CTR or CM)

- counter mode: a counter is incremented by one for each block and run through the cipher and XOR with message to produce the cipher text
  - receiver has got the counter start value, an Initialization Vector (IV), and encrypts the counter the same way and XOR with cipher text to produce original message



Symmetric key cryptography



Symmetric-key cryptography: Bob and Alice share same secret (symmetric) key: *k*

- *e.g.*, key is knowing substitution pattern in mono alphabetic substitution cipher

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

Public Key Cryptography

symmetric key crypto:

- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

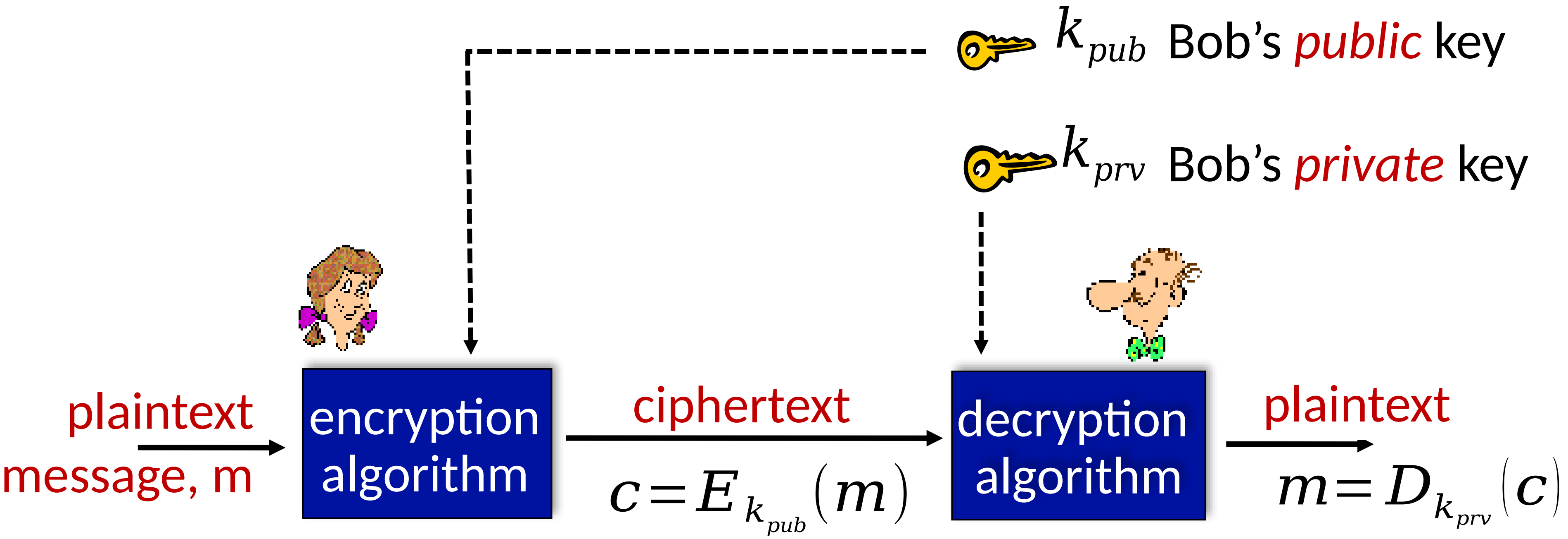
public key crypto

- *radically* different approach [Diffie-Hellman76, RSA78]
- sender, receiver do *not* share secret key
- *public* encryption key known to *all*
- *private* decryption key known only to receiver

Public Key Cryptography

Key pair :
- encryption: public key - known by anybody
- decryption: private key - known ***only*** by the owner

Security property: Confidentiality protection



Public key encryption algorithms

requirements:

1 need and such that

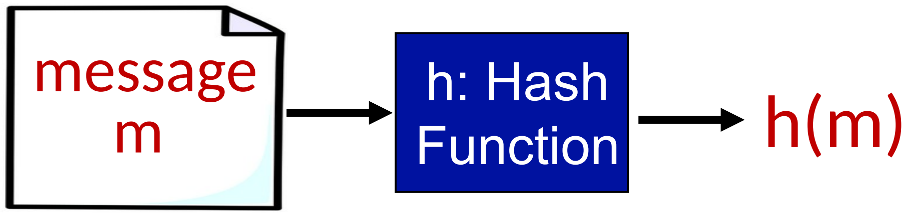
$$m = D_{k_{prv}}\left(E_{k_{pub}}(m)\right)$$

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

Chapter 8 outline

Hash functions

- A hash function ***h(m)*** converts any input ***m*** into a hash (or message digest) of fixed length ☾ "fingerprint" of ***m***
- Secure one-way function



Hash function properties:

- Onewayness: given message digest ***x***, computationally infeasible to find ***m*** such that ***x = h(m)***
- Difficult to find two messages, so that ***h(m1) = h(m2)***

Hash function algorithms

- Message Digest (MD5)
  - computes 128-bit message digest in 4-step process.
  - arbitrary 128-bit string x, appears difficult to construct msg m whose MD5 hash is equal to x
- Secure Hash Algorithm (SHA-1)
  - US standard [NIST, FIPS PUB 180-1]
  - 160-bit message digest
- Secure Hash Algorithm (SHA-2)
  - US standard [NIST, FIPS PUB 180-2]
  - SHA-256, SHA-384 and SHA-512 named after their digest length

Message authentication

If Bob receives a message from Alice, how can Bob be sure that:

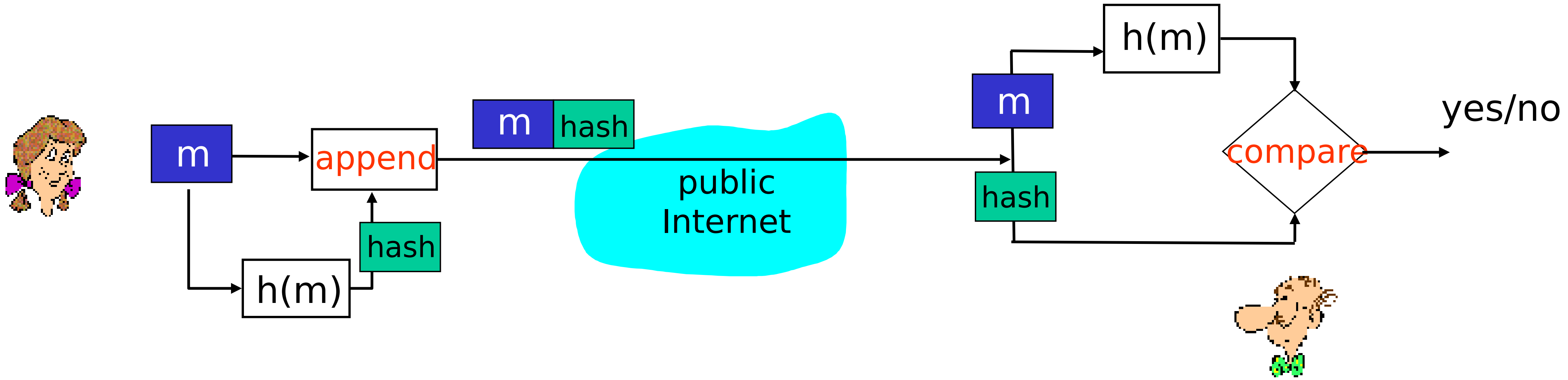
1. Alice sent it? impersonation, spoofing attacks
2. the message has not been intentionally modified by an adversary? integrity breech

Message authentication aka. data origin authentication:

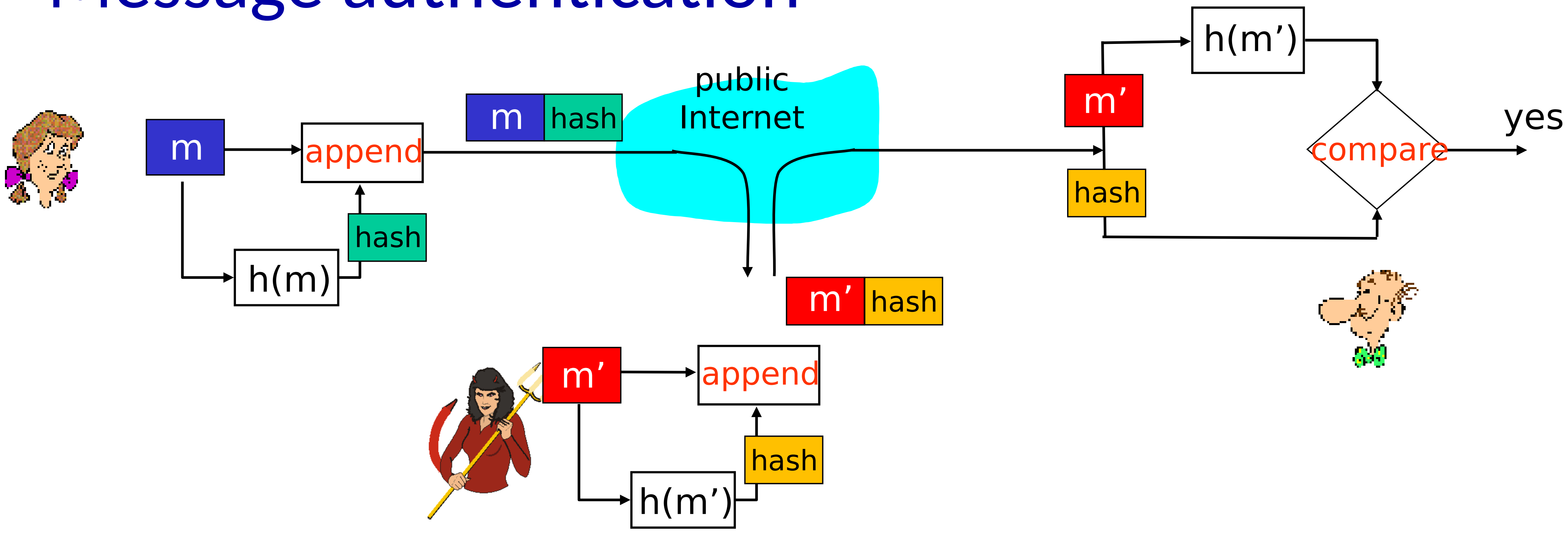
3. confirm that the message originated from the stated sender (sender authenticity)
4. confirm that the message has not been changed (integrity)

Message authentication

What if Alice appends a hash?



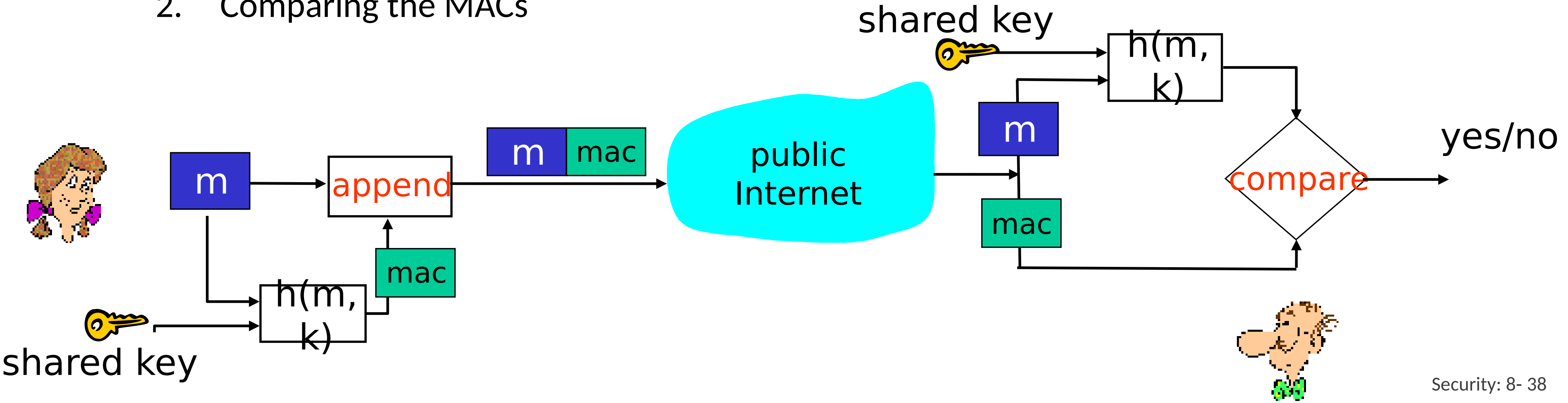
Message authentication



- Important: Hashes do not protect against an adversary that
  1. modifies message contents
  2. pretends to be a somebody else

Message authentication

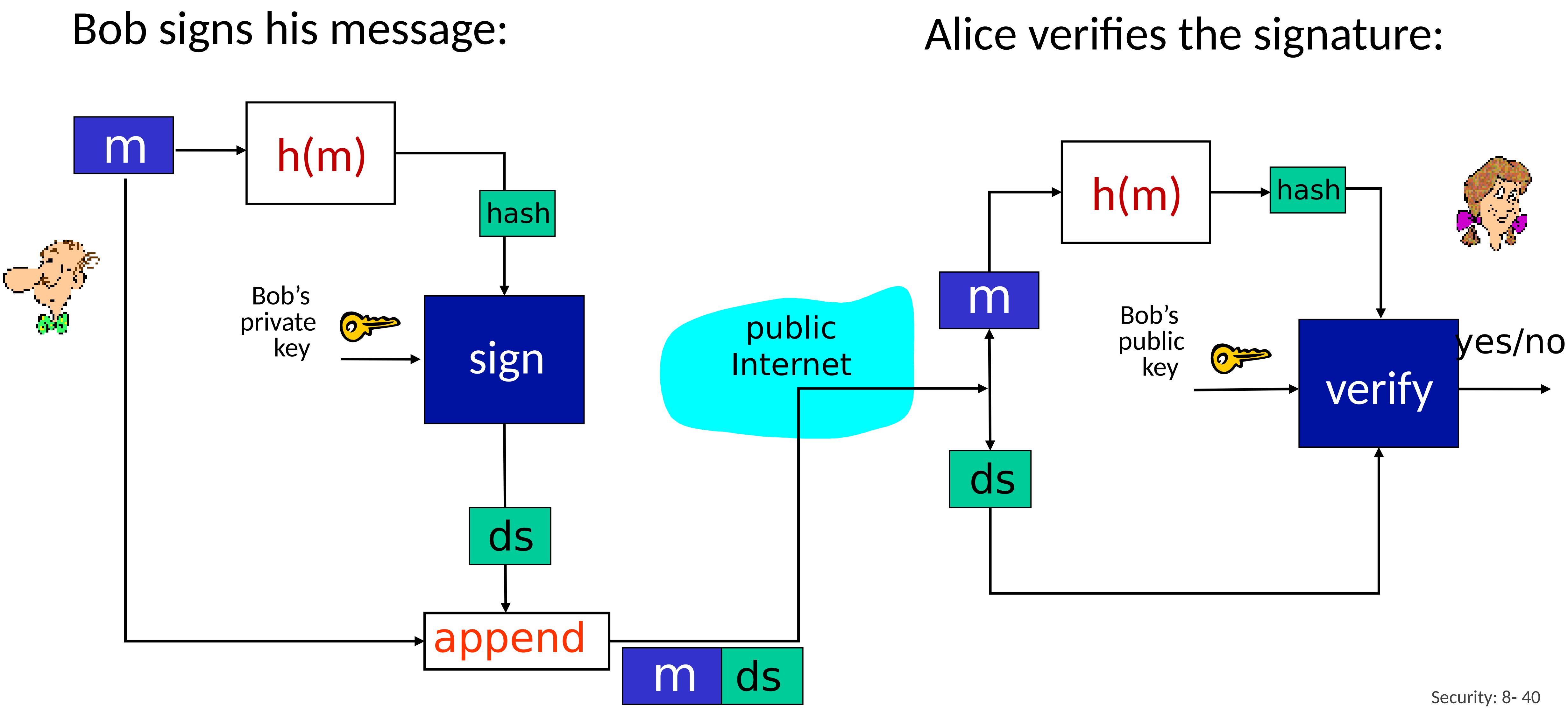
- Message authentication codes (MAC) are a short piece of information used for authenticating and integrity-checking a message
- Alice and Bob share a secret key
  1. Alice computes a hash of the key and the message
  2. Bob verifies the message by:
     1. Computing a hash of his key and the received message
     2. Comparing the MACs



Digital signatures

- Message authentication works for two parties sharing a secret key
- The receiver can prove the identity of the sender
- How can it be proven to anybody who
  1. is the sender of a message?
  2. is the originator of a file (e.g. document)?
- Digital signatures use asymmetric cryptography
- Security property: Non-repudiation
  - Not forgeable
    - In contrast, handwritten signatures are forgeable

Digital signatures

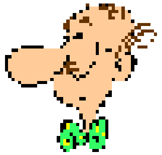


Symmetric key cryptography

- Confidentiality protection

$$c = E_k(m)$$

$$\xrightarrow{\quad c \quad}$$



$$m = D_k(c)$$

Symmetric key cryptography

- Message authentication

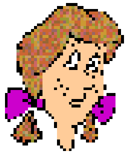
$$mac = f(m, k)$$

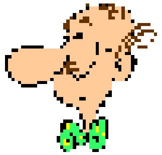
$$\xrightarrow{[m, mac]}$$

$$mac \overset{?}{=} f(m, k)$$

Symmetric key cryptography

- Authenticated encryption:
  Confidentiality protection + message authentication





$$h = f(m)$$

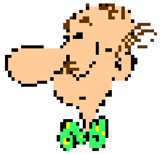
$$c = E_k(m, h) \xrightarrow{\quad c \quad}$$

$$[m, h] = D_k(c)$$

$$h \stackrel{?}{=} f(m)$$

Asymmetric key cryptography

- Digital signature



$$s = sign_{k_{prv}}\left(f(m)\right)$$

$$[m, s]$$

$$f(m) \stackrel{?}{=} verify_{k_{pub}}(s)$$

Security properties and cryptographic primitives

	Confidentiality	Data integrity	Message authentication	Non-repudiation
Hash functions	No	No	No	No
Symmetric key encryption	Yes	No*	No*	No
Public key encryption	Yes	No	No	No
Message authentication codes	No	Yes	Yes	No
Digital signatures	No	Yes	Yes	Yes

* The encryption contains an appended hash

Replay attacks

- A passive adversary can eavesdrop and record
  - encrypted messages (or plaintext messages)
  - plaintext messages with MACs
  - signed messages
- Then later replayed (sent) to the same receiver
  - The messages appear authentic
- Thus, the adversary can successfully deceive the receiver
- Encryption, MACs and/or digital signatures alone do not prevent replay attacks

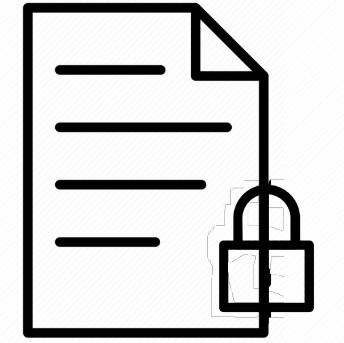
Digital certificates

Authenticity of public keys

public key problem:

- when Alice obtains Bob's public key (from web site, e-mail, diskette), how does she know it is Bob's public key, not Trudy's?

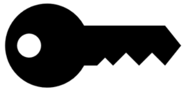
Threat scenario 1



Encrypt

Decrypted

Attacker's public key



Threat scenario 2

Sign

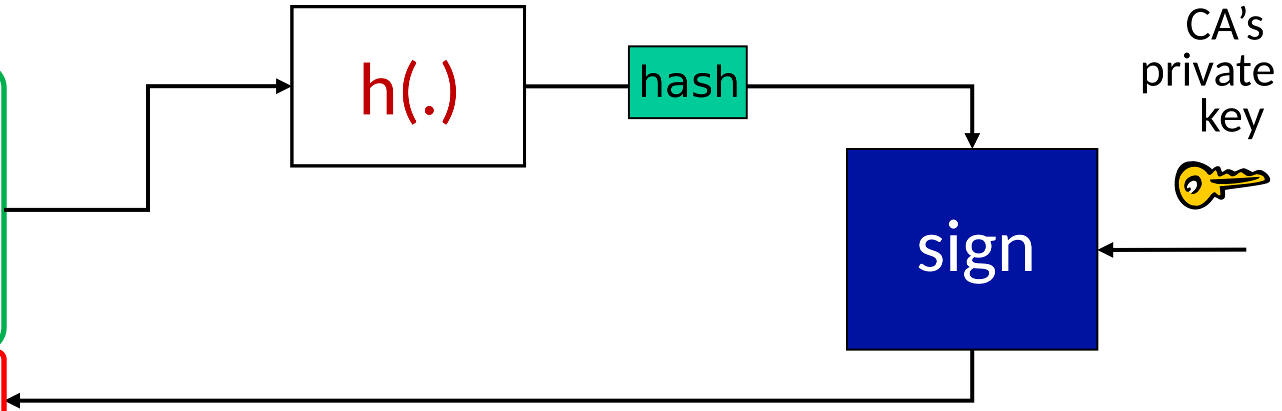
Attacker's signature key

Attacker's verification key

Signature verification

Digital certificates

- A document to prove the authenticity of a public key/verification key
- Digital certificate is a binds a public key to the owner's name
- Assumes a point of trust: issued by a certificate authority (CA)
- The certificate contains
  1. Owner's name
  2. Owner's public key
  3. CA's name
  4. CA's signature



- To prove the validity of a public key, the certificate must be validated

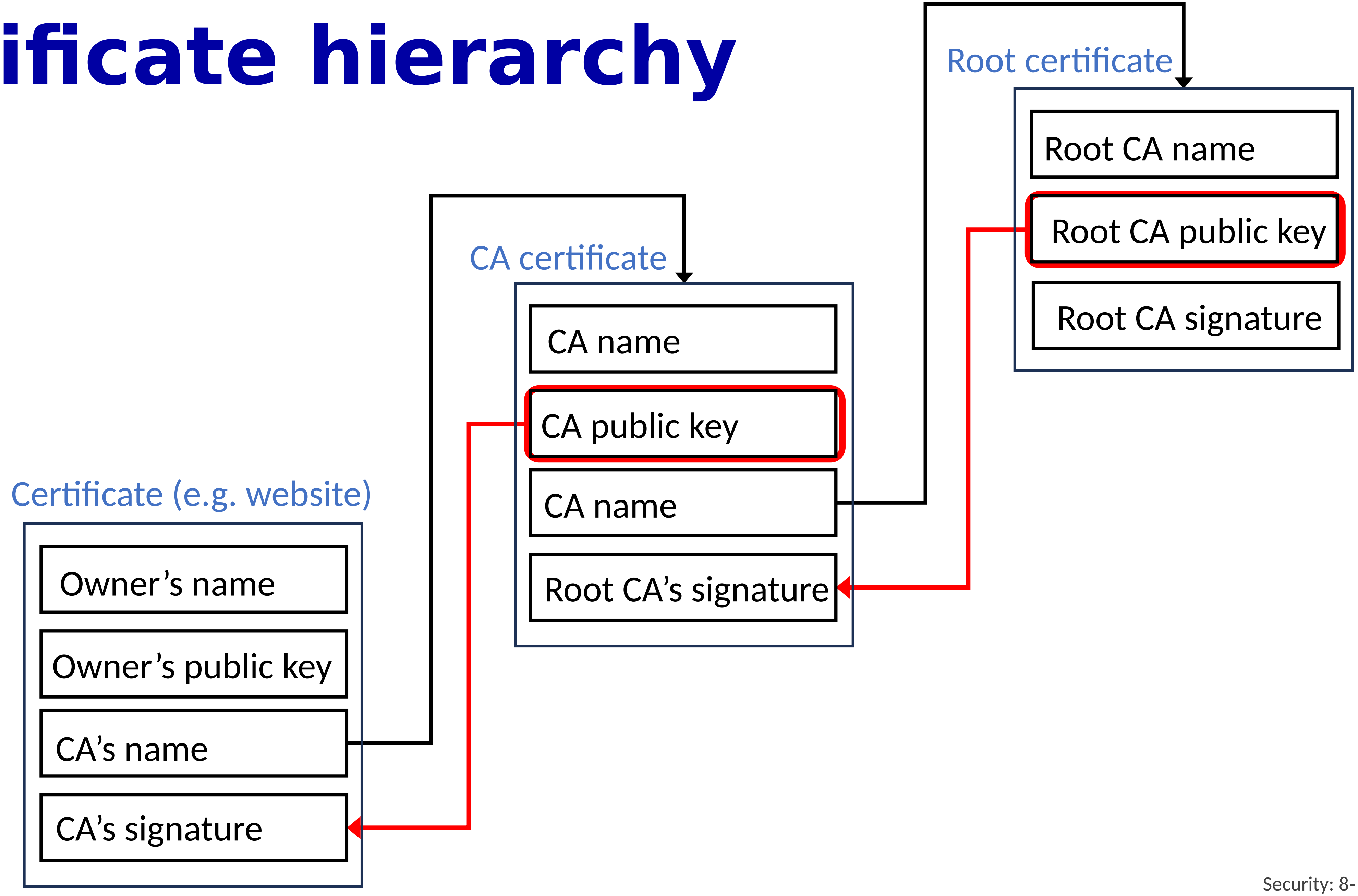
X.509 v.1 certificate format

Version	Version number of the certificate.
Serial Number	Unique number for each certificate issued by a certificate authority (CA).
Signature	The identifier for the cryptographic algorithm used by the CA to sign the certificate.
Issuer	The distinguished name (DN) of the certificate's issuing CA.
Validity	The time period for which the certificate is valid.
Subject	The distinguished name (DN) of the certificate owner.
Subject Public Key Info	The public key owned by the certificate subject.

What it is not

- A public key certificate is not a proof of identity
- A public-key certificate is a statement that the public key contained in it belongs to the named owner and has the properties specified in the certificate
- Once the certificate has been checked, the public key can be extracted from the certificate and then used for its specified purpose

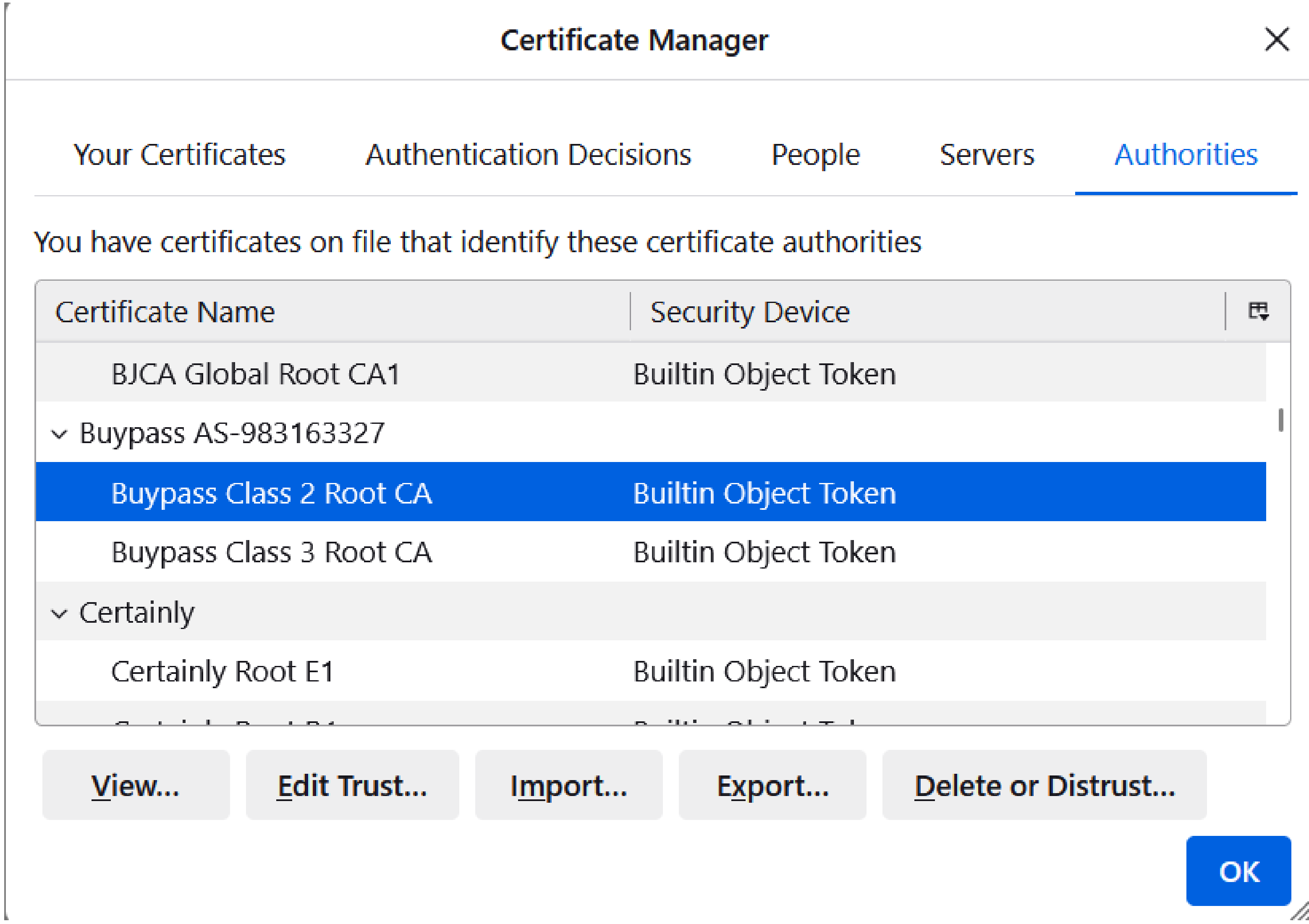
Certificate hierarchy



Browser certificates

Browser root certificates

- Web-browsers have prestored root certificates
- Mozilla Firefox
  - 47 root certificates, representing 52 organizations
  - Actual count gives 176 root certificates, representing 75 organizations



Browser root certificates

(The browser hides the signatures)

Buypass Class 2 Root CA

Subject Name

Country	NO
Organization	Buypass AS-983163327
Common Name	Buypass Class 2 Root CA

Issuer Name

Country	NO
Organization	Buypass AS-983163327
Common Name	Buypass Class 2 Root CA

Validity

Not Before	Tue, 26 Oct 2010 08:38:03 GMT
Not After	Fri, 26 Oct 2040 08:38:03 GMT

Public Key Info

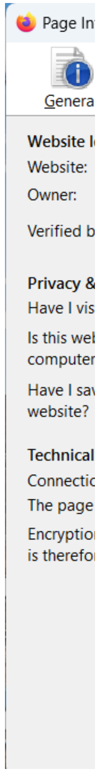
Algorithm	RSA
Key Size	4096
Exponent	65537
Modulus	D7:C7:5E:F7:C1:07:D4:77:FB:43:21:F4:F4:F5:69:E4:EE:32:01:DB:A3:86:1F:E4:59:0D:...

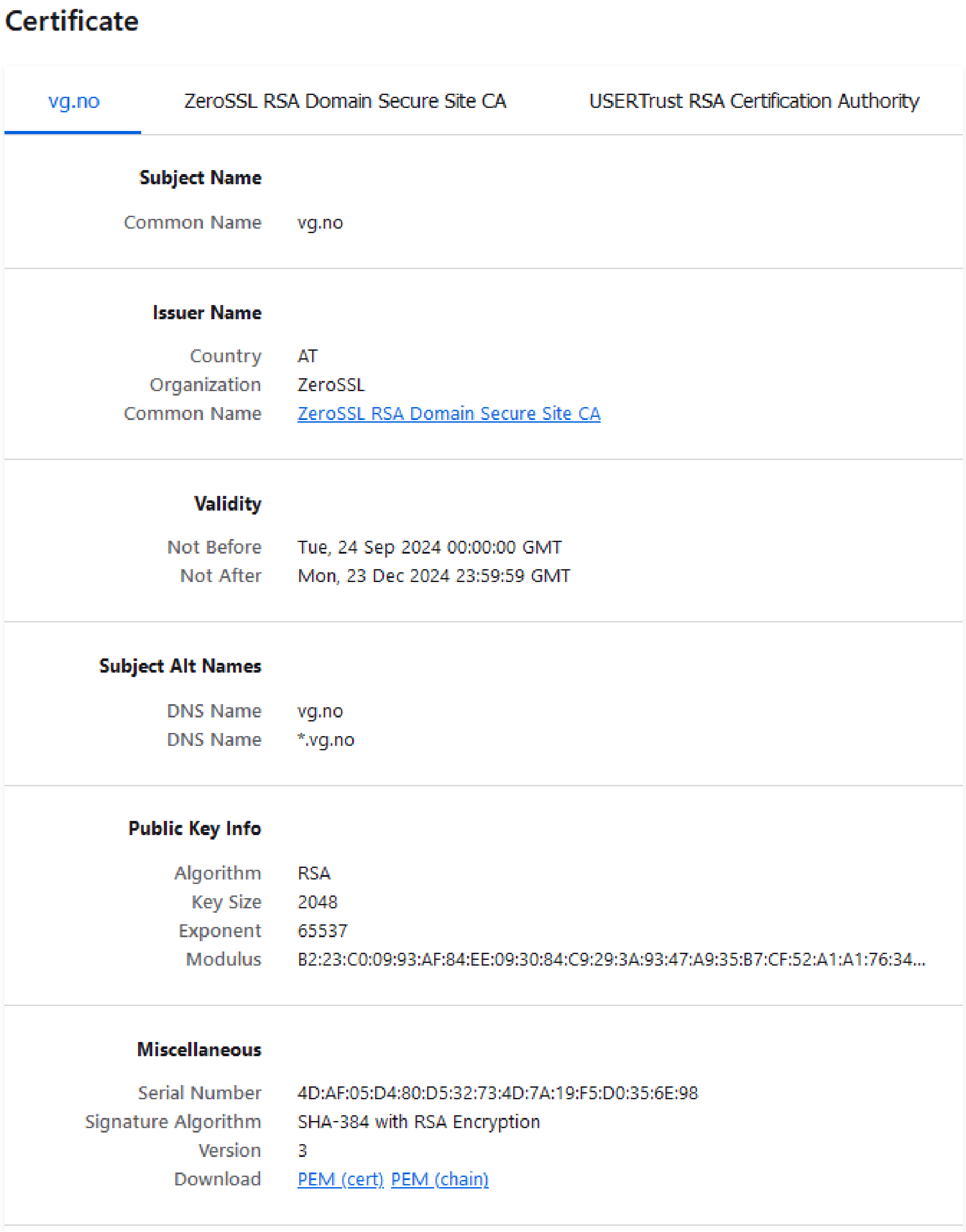
Miscellaneous

Serial Number	02
Signature Algorithm	SHA-256 with RSA Encryption
Version	3
Download	PEM (cert) PEM (chain)

HTTPS

- HTTPS is indicated by padlock symbol
  - downloads the cerificate chain of a visited website
  - Parts of certs. can be viewed

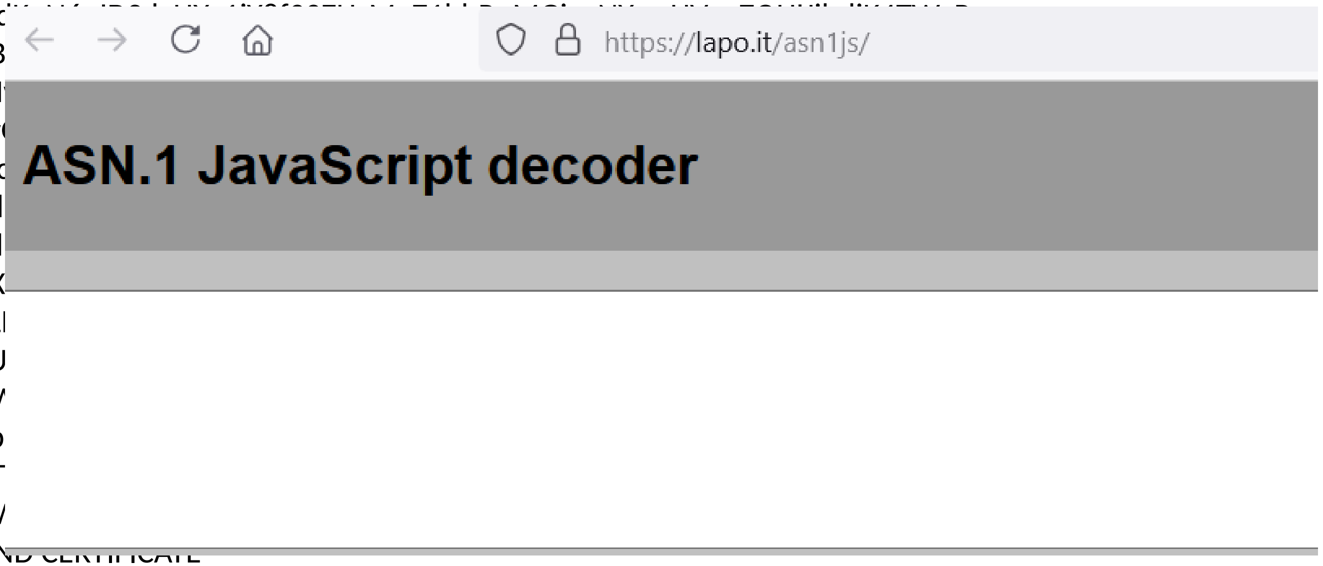




PEM-format

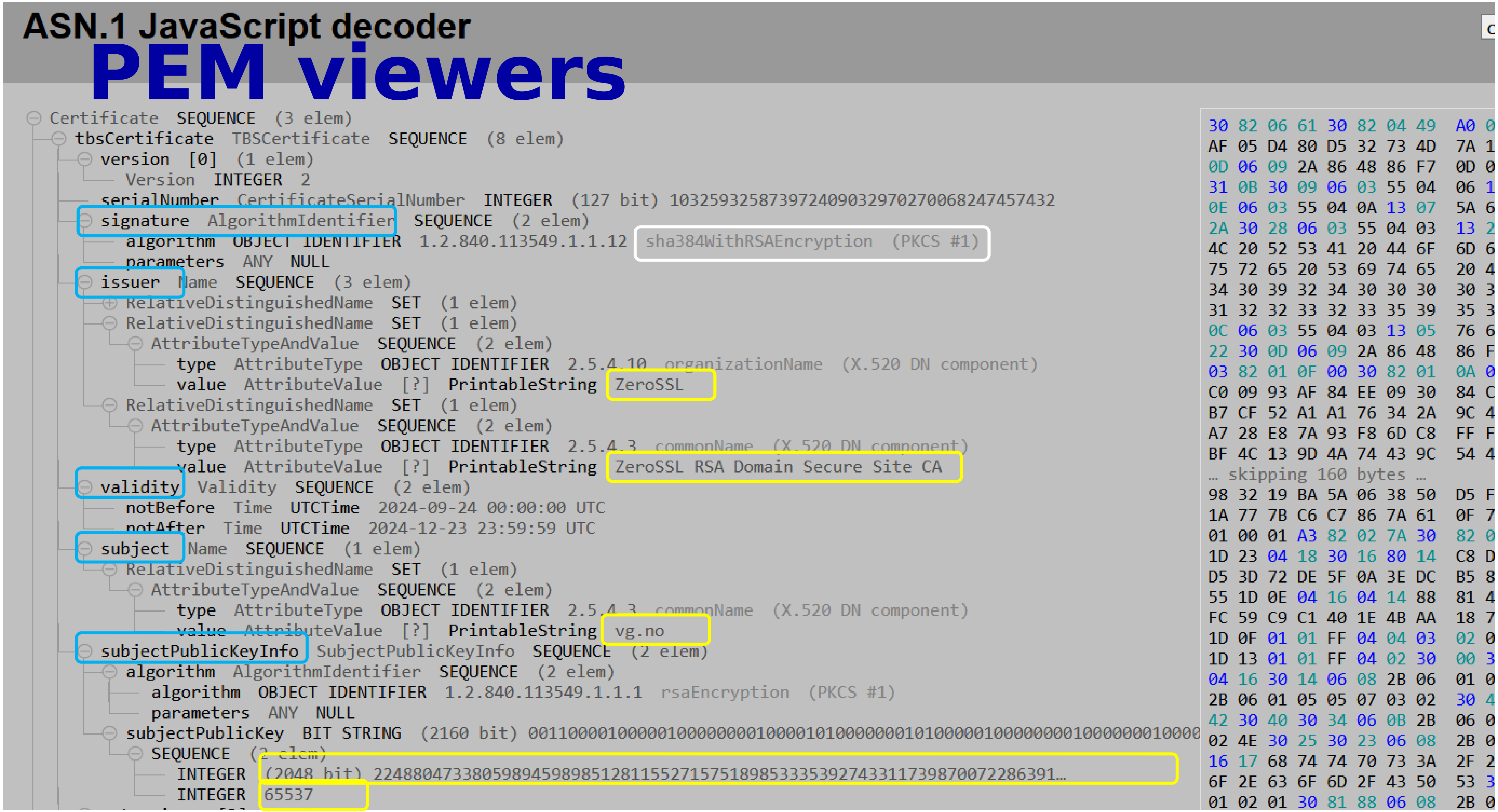
- The certs are stored in PEM-files
  - PEM: Privacy Enhanced Mail
- Can be viewed by a PEM-“decoders”
  - Some does not view signatures

-----BEGIN CERTIFICATE-----
MIIFWTCCA0GgAwIBAgIBAjANBgkqhkiG9w0BAQsFADBOMQswCQYDVQQGEwJOTzEd
MBsGA1UECgwUQnV5cGFzcyBBUy05ODMxNjMzMjcxIDAeBgNVBAMMF0J1eXBhc3Mg
Q2xhc3MgMiBSb290IENBMB4XDTEwMTAyNjA4MzgwM1oXDTQwMTAyNjA4MzgwM1ow
TjELMAkGA1UEBhMCTk8xHTAbBgNVBAoMFEJ1eXBhc3MgQVMtOTgzMTYzMzI3MSAw
HgYDVQQDDBdCdXlwYXNzIENsYXNzIDIgUm9vdCBDQTCCAiIwDQYJKoZIhvcNAQEB
BQADggIPADCCAgoCggIBANfHXvfBB9R3+0Mh9PT1aeTuMgHbo4Yf5FkNuud1g1Lr
6hxhFUi7HQfKjK6w3Jad6sNgkoaCKHOcVgb/S2TwDCo3SbXlzwx87vFKu3MwZfPV
L4O2fuPn9Z6rYPnT8Z2SdIrkHJasW4DptfQxh6NR/Md+oW+OU3fUl8FVM5I+GC91
1K2GScuVr1QGbNgGE41b/+EmGVnAJLqBcXmQRFBoJJRfuLMR8SlBYaNByyM21cHx
MlAQTn/0hpPshNOOvEu/XAFOBz3cFIqUCqTqc/sLUegTBxj6DvEr0VQVfTzh97QZ
QmdiXnfgolXsttlpF9U6r0TtSsWe5HonfOV116rLJeffawrbD02TTqigzXsu8lkB
arcNuAeBfos4GzjmCleZPe4h6KP1DBbdi+w0jpwqHAAVF41og9JwnxgIzRFo1clr
Us3ERo/ctfPYV3Me6ZQ5BL/T3jjetFPsaRyifsSP5BtwrfKi+fv3FmRmaZ9JUaLi
FRhnBkp/1Wy1TbMz4GHrXb7pmA8y1x1LPC5aAVKRCfLf6o3YBkBjqhHk/sM3nhRS
P/TizPJhk9H9Z2vXUq6/aKtAQ6BXNVN48FP4YUIHZMbXb5tMOA1jrGKvNouicwoN
9SG9c
AgMB
uvdM
9bqw
A20uc
Oluwl
+fsicd
KcZ7X
DISCLI
H8SIU
I+uUV
5t98b
3PFaT
Y11aV
-----EN

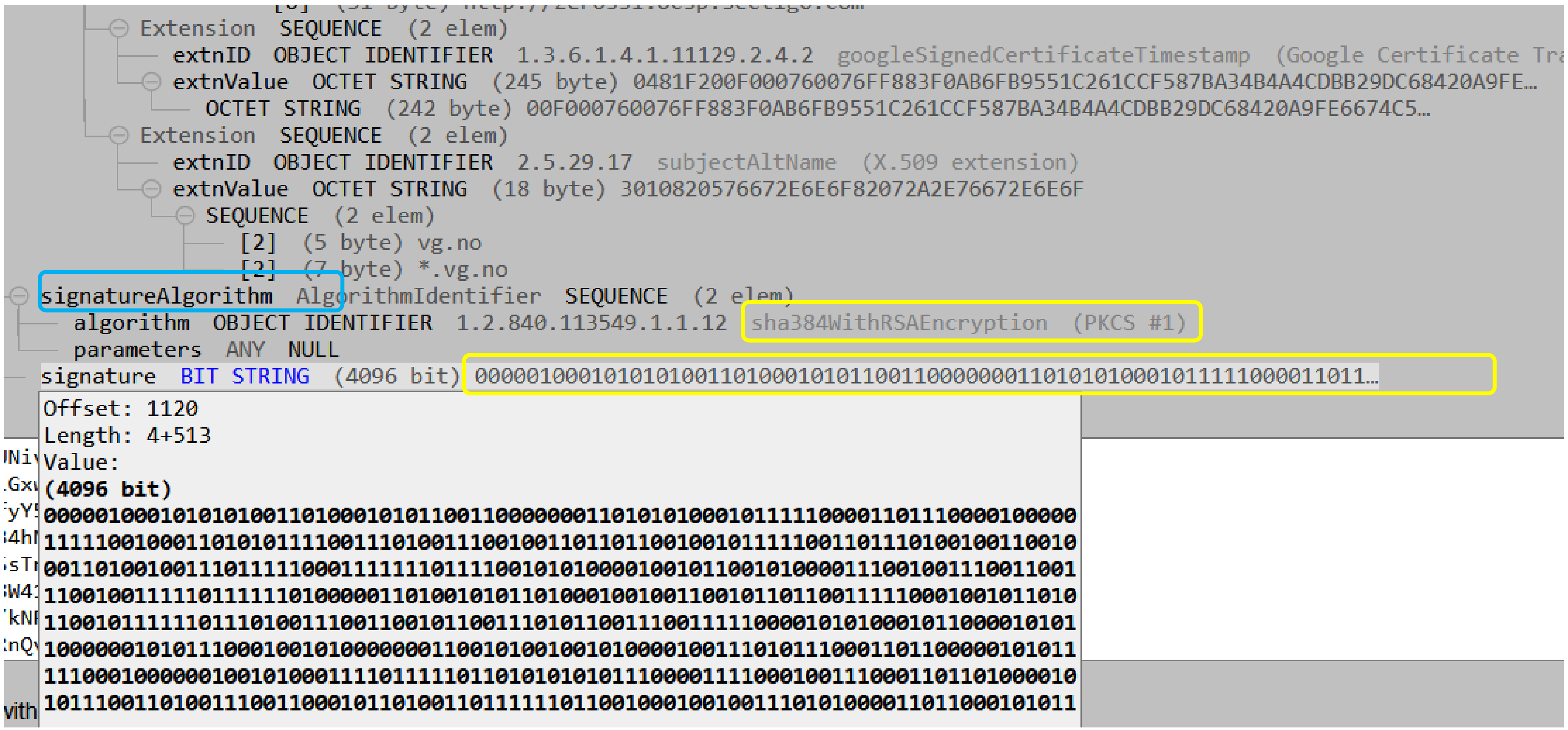


https://report-uri.com/home/pem_decoder

PEM viewers



ASN.1 JavaScript decoder
Certificate SEQUENCE (3 elem)
tbsCertificate TBSCertificate SEQUENCE (8 elem)
version [0] (1 elem)
Version INTEGER 2
serialNumber CertificateSerialNumber INTEGER (127 bit) 103259325873972409032970270068247457432
signature AlgorithmIdentifier SEQUENCE (2 elem)
algorithm OBJECT IDENTIFIER 1.2.840.113549.1.1.12 sha384WithRSAEncryption (PKCS #1)
parameters ANY NULL
issuer Name SEQUENCE (3 elem)
RelativeDistinguishedName SET (1 elem)
RelativeDistinguishedName SET (1 elem)
AttributeTypeAndValue SEQUENCE (2 elem)
type AttributeType OBJECT IDENTIFIER 2.5.4.10 organizationName (X.520 DN component)
value AttributeValue [?] PrintableString ZeroSSL
RelativeDistinguishedName SET (1 elem)
AttributeTypeAndValue SEQUENCE (2 elem)
type AttributeType OBJECT IDENTIFIER 2.5.4.3 commonName (X.520 DN component)
value AttributeValue [?] PrintableString ZeroSSL RSA Domain Secure Site CA
validity Validity SEQUENCE (2 elem)
notBefore Time UTCTime 2024-09-24 00:00:00 UTC
notAfter Time UTCTime 2024-12-23 23:59:59 UTC
subject Name SEQUENCE (1 elem)
RelativeDistinguishedName SET (1 elem)
AttributeTypeAndValue SEQUENCE (2 elem)
type AttributeType OBJECT IDENTIFIER 2.5.4.3 commonName (X.520 DN component)
value AttributeValue [?] PrintableString vg.no
subjectPublicKeyInfo SubjectPublicKeyInfo SEQUENCE (2 elem)
algorithm AlgorithmIdentifier SEQUENCE (2 elem)
algorithm OBJECT IDENTIFIER 1.2.840.113549.1.1.1 rsaEncryption (PKCS #1)
parameters ANY NULL
subjectPublicKey BIT STRING (2160 bit) 0011000010000010000000010000101000000010100000100000000100000000010000
SEQUENCE (2 elem)
INTEGER (2048 bit) 224880473380598945989851281155271575189853335392743311739870072286391…
INTEGER 65537
… skipping 160 bytes …



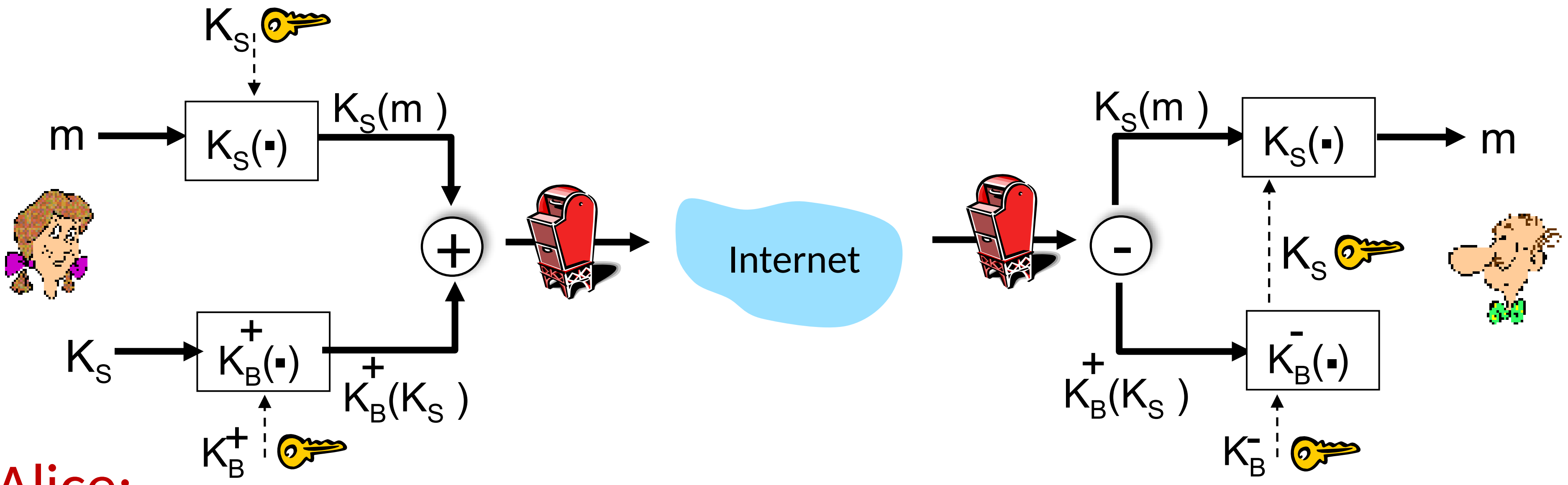
Extension SEQUENCE (2 elem)
extnID OBJECT IDENTIFIER 1.3.6.1.4.1.11129.2.4.2 googleSignedCertificateTimestamp (Google Certificate Tra
extnValue OCTET STRING (245 byte) 0481F200F000760076FF883F0AB6FB9551C261CCF587BA34B4A4CDBB29DC68420A9FE…
OCTET STRING (242 byte) 00F000760076FF883F0AB6FB9551C261CCF587BA34B4A4CDBB29DC68420A9FE6674C5…
Extension SEQUENCE (2 elem)
extnID OBJECT IDENTIFIER 2.5.29.17 subjectAltName (X.509 extension)
extnValue OCTET STRING (18 byte) 3010820576672E6E6F82072A2E76672E6E6F
SEQUENCE (2 elem)
[2] (5 byte) vg.no
[2] (7 byte) *.vg.no
signatureAlgorithm AlgorithmIdentifier SEQUENCE (2 elem)
algorithm OBJECT IDENTIFIER 1.2.840.113549.1.1.12 sha384WithRSAEncryption (PKCS #1)
parameters ANY NULL
signature BIT STRING (4096 bit) 0000010001010101001101000101011001100000001101010100010111110000110111…
Offset: 1120
Length: 4+513
Value:
(4096 bit)
000001000101010100110100010101100110000000110101010001011111000011011100001000000
111110010001101010111100111010011100100110110110010010111110011011101001001100010
001101001001110111110001111111011110010101000010010110010100001110010011100110001
110010011111011111101000001101001010110100010010011001011011001111100010010110100
110010111111011101001110011001011001110101100111001111100001010100010110000100101
100000010101110001001010000000110010100100101000010011101011100011011000001010111
111000100000010010100011110111110110101010101110000111100010011100011011010000100
101110011010011100110001011010011011111101100100010010011101010001101100010101111

Chapter 8 outline

Chapter 8 outline

Secure e-mail: confidentiality

Alice wants to send *confidential* e-mail, m, to Bob.

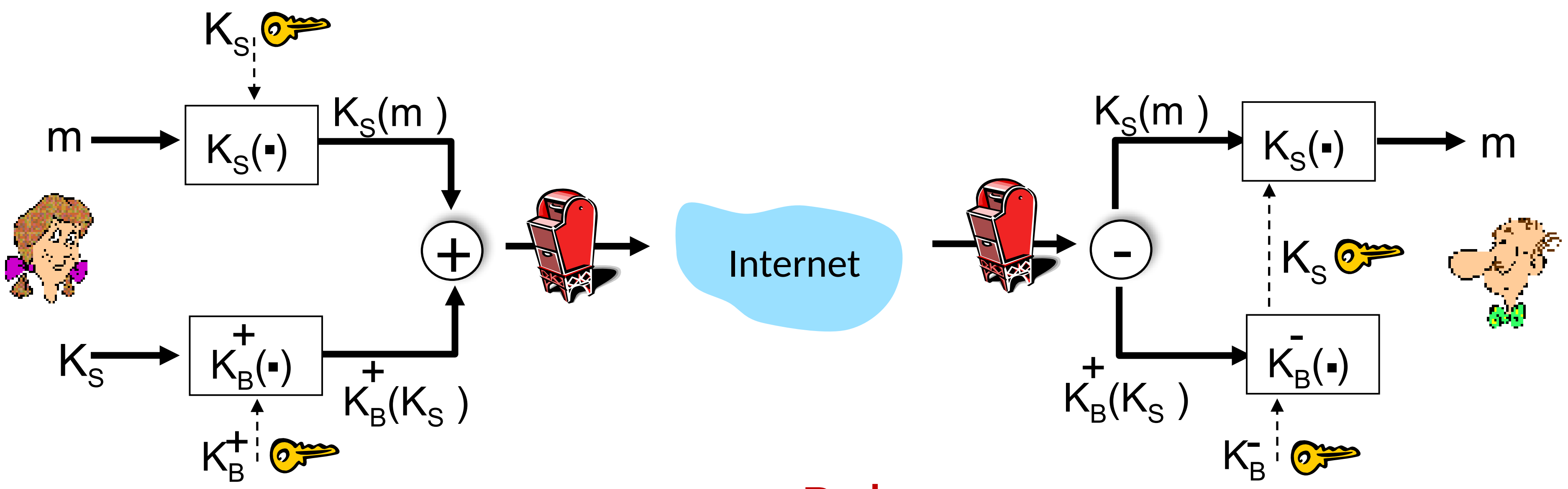


Alice:

- generates random *symmetric* private key, K_S
- encrypts message with K_S (for efficiency)
- also encrypts K_S with Bob's public key
- sends both $K_S(m)$ and $K^+_B(K_S)$ to Bob

Secure e-mail: confidentiality (more)

Alice wants to send *confidential* e-mail, m, to Bob.

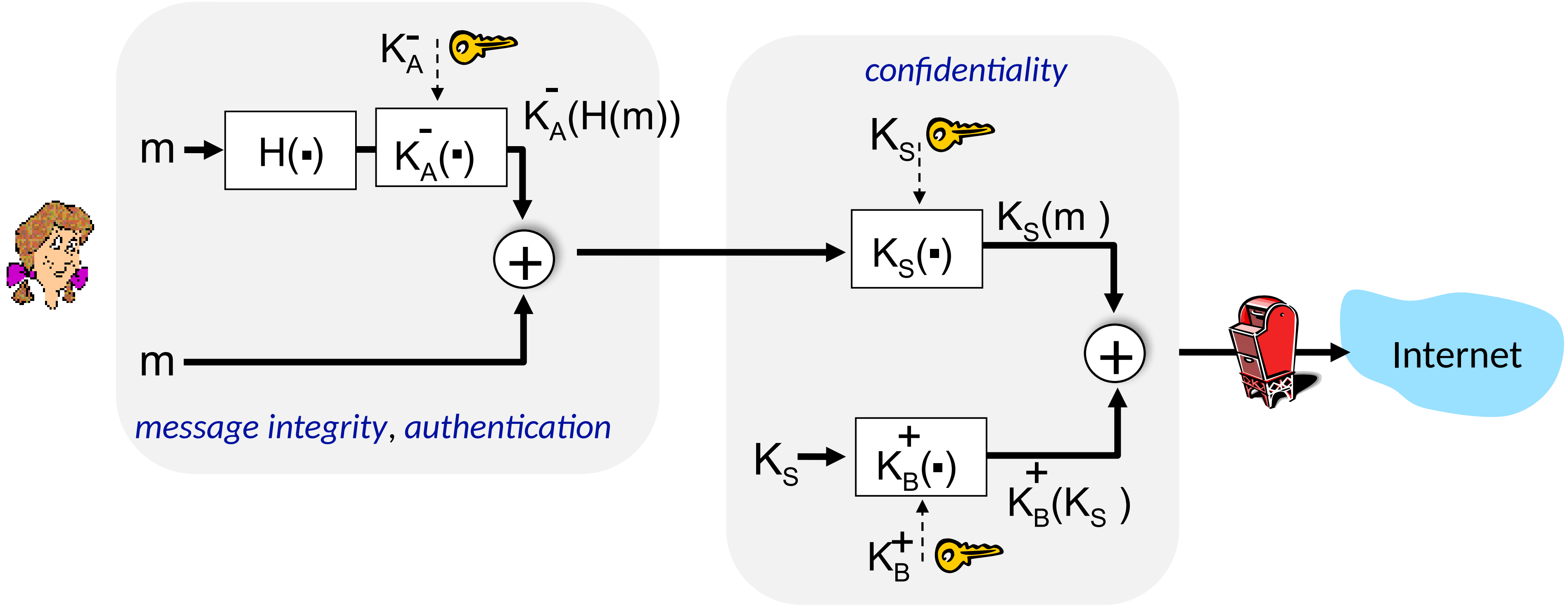


Bob:

- uses his private key to decrypt and recover K_S
- uses K_S to decrypt $K_S(m)$ to recover m

Secure e-mail: integrity, authentication

Alice sends m to Bob, with *confidentiality, message integrity*, *authentication*



Alice uses three keys: her private key, Bob's public key, new symmetric key

What are Bob's complementary actions?

Pretty Good Privacy (PGP)

- internet e-mail encryption scheme, de-facto standard
- uses symmetric key cryptography, public key cryptography, hash function, and digital signature as described
- provides secrecy, sender authentication, integrity
- inventor, Phil Zimmerman, was target of 3-year federal investigation

A PGP signed message:

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

Bob: Can I see you tonight?
Passionately yours, Alice

-----BEGIN PGP SIGNATURE-----
Version: PGP 5.0
Charset: noconv
yhHJRHhGJGhgg/
12EpJ+lo8gE4vB3mqhFEvZP9t6n7G6m5Gw
2
-----END PGP SIGNATURE-----
```

A secret PGP message:

```
-----BEGIN PGP MESSAGE-----
Version: PGP 5.0
u2R4d+/
jKmn8Bc5+hgDsqAewsDfrGdszX68liKm5F
6Gc4sDfcXytRfdS10juHgbcfDssWe7/
K=lKhnMikLo0+1/
BvcX4t==Ujk9PbcD4Thdf2awQfgHbnmKlo
k8iy6gThlp
-----END PGP MESSAGE-----
```

Chapter 8 outline

8.1 What is network security?

8.2 Principles of cryptography

8.3 Message integrity and digital signatures

8.4 Authentication

8.5 Securing e-mail

8.6 Securing TCP connections: TLS

~~8.7 Network layer security: IPsec~~

~~8.8 Security in wireless and mobile networks~~

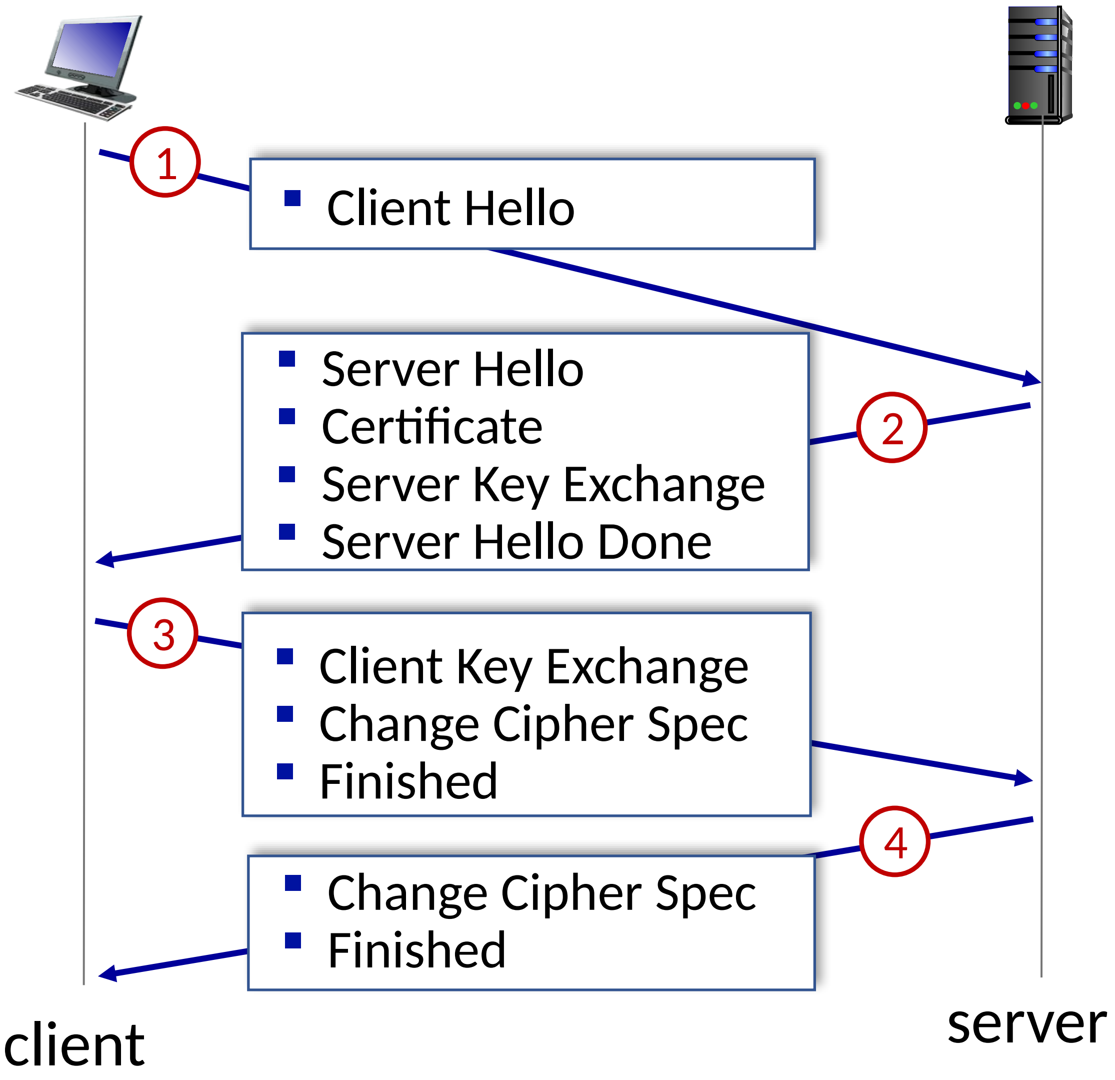
8.9 Operational security: firewalls and IDS

Transport-layer security (TLS)

- widely deployed security protocol above the transport layer
  - supported by almost all browsers, web servers: https (port 443)
- provides:
  - confidentiality: via *symmetric encryption*
  - integrity: via *cryptographic hashing*
  - authentication: via *public key cryptography*

  all techniques we have studied!
- history:
  - early research, implementation: secure network programming, secure sockets
  - secure socket layer (SSL) deprecated [2015]
  - TLS 1.3: RFC 8846 [2018]

TLS 1.2 handshake – DH key exchange: 2 RTT



1
- Client Hello: List of supported cipher suites and nonce

2
- Server Hello: Selected cipher suite and nonce
- Certificate: RSA or ECDSA
- Server Key Exchange: signed DHE or ECDHE parameters and key share
- Server Hello Done: Done

3
- Client Key Exchange: DHE or ECDHE key share
- Change Cipher Spec: Have generated session keys and switch to encrypted communication
- Finished: MAC of sent/received handshake messages

4
- Change Cipher Spec: Have generated session keys and switch to encrypted communication
- Finished: MAC of sent/received handshake messages

Diffie-Hellman key exchange Ephemeral (DHE)

1. Alice and Bob publicly agree to use a modulus $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).
2. Alice chooses a secret integer $a = 4$, then sends Bob $A = g^a \bmod p$
   - $A = 5^4 \bmod 23 = 4$
3. Bob chooses a secret integer $b = 3$, then sends Alice $B = g^b \bmod p$
   - $B = 5^3 \bmod 23 = 10$
4. Alice computes $s = B^a \bmod p$
   - $s = 10^4 \bmod 23 = 18$
5. Bob computes $s = A^b \bmod p$
   - $s = 4^3 \bmod 23 = 18$
6. Alice and Bob now share a secret (the number 18).

[4]

Both Alice and Bob have arrived at the same values because under mod p,

$$A^b \bmod p = g^{ab} \bmod p = g^{ba} \bmod p = B^a \bmod p$$

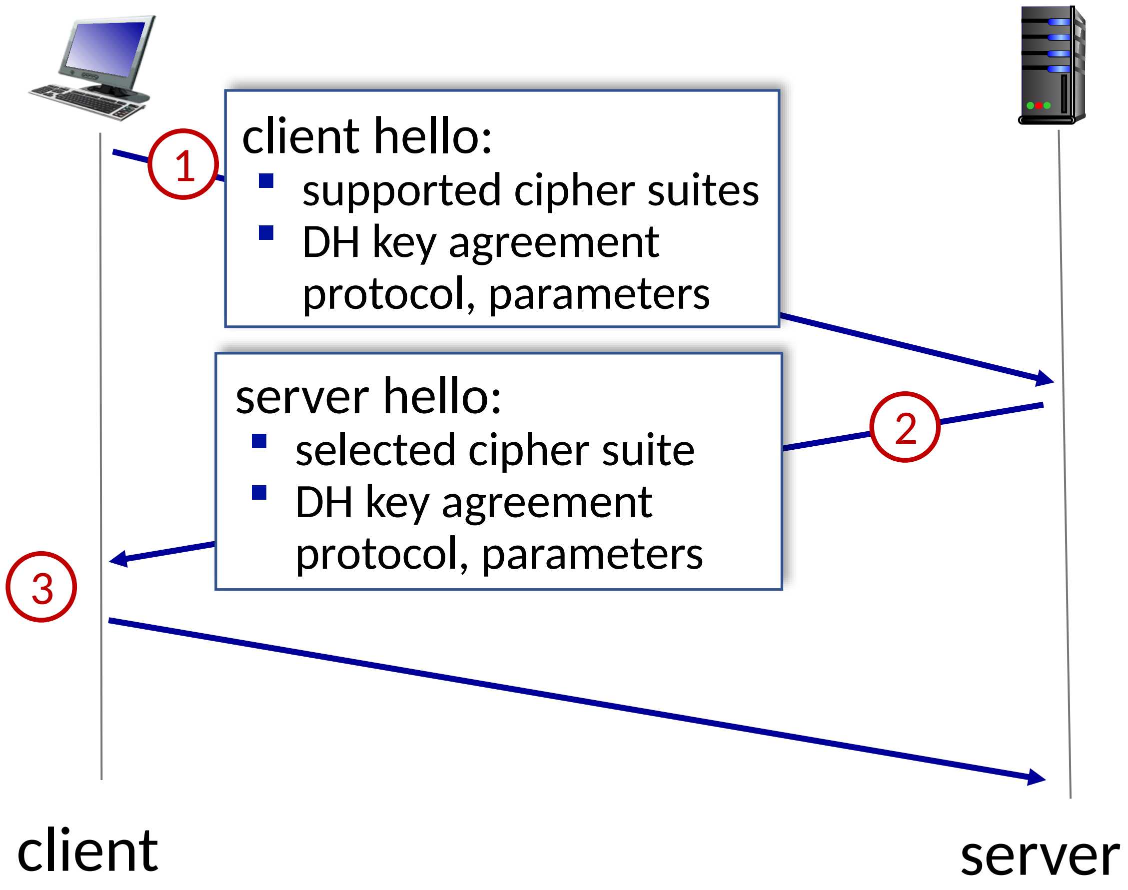
Diffie-Hellman key exchange Ephemeral (DHE)

- Prime **p** is huge (e.g., 6144-bit MODP Group (Group 17) from RFC 3526):
  337515218214385611845185231599674123300648978057418465481738904744294299013266724452032351019191654
  839641943594609948810620893878937628140442574382044325739410830148270060902589258751610180963277323
  358005958319159760142088223040073278481327349332978858032136752615649626033404572207768263225000580
  913109672539766199739880336636663851881552126562680795017262233696934279998041344678101207723564985
  969455323665274005175754719693358549052745041195095923660137119541482588848792245999152034563158810
  347765530836769957183355985863955911699995708245150350175435333526975252877533325005271765695768949
  267349504692935961340950866037168600863020515445396526890912990997845889190523834630577894405654606
  814419024423999564190605216296046973478790246543138001860783165269645292880627408790110351759200591
  921785614731990062058967194350147653455184908823666071109053034491525562211632321274264406919211346
  487666356958502392313045917442156109850296368954067188807663082492273159842675422662594896843722239
  164454110159005062394192679097163203312089889781808689874316237103476179923562014490238922032301330
  094214639142912013460631252196369642616835915410143442392753407356909977322220697587739633908763605
  465157552805170421605254873028981223116697996794475304536003993426970327144585495912859394539490349
  812481143223223672386450425159844478907889178235763300191516965686543141530585475920913660145501438
  196851700683437001046776090411663697600809334136054989623820777788455998349074759534307874462013845
  673285306752757929623548837708069008271836857183534695747316805206219445409477346190351771800579730
  226525710321965982292591948757099947097217931541586865157485072742241813169487971046010682120152329
  216914824963468544136987197501906011027052744810505432398151306860736010763045122845492184598460460
  822535967624338274190600890294170448712183160209231099889157071175 67
- Generator **g** is generally small (for OpenSSL always 2)
- The secret choices **s** are usually huge (256-bits and above)

TLS: 1.3 cipher suite

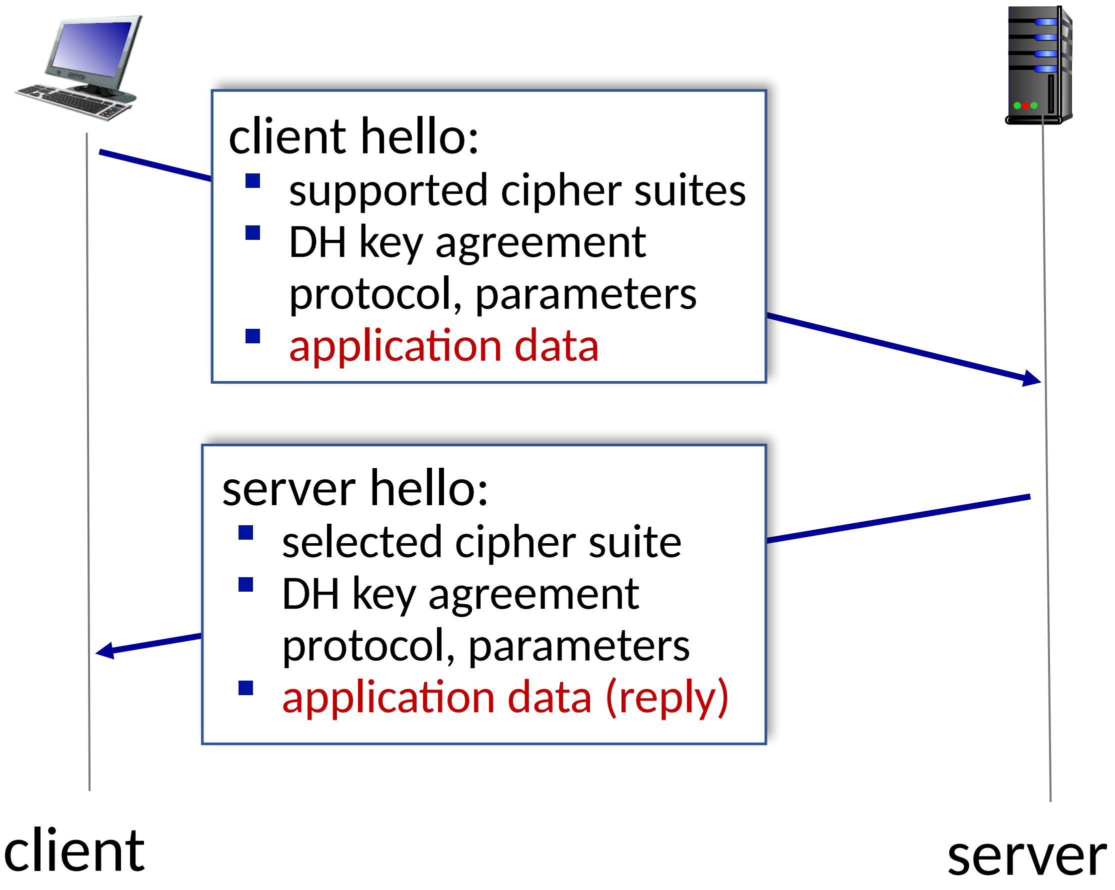
- "cipher suite": algorithms that can be used for key generation, encryption, MAC, digital signature
- TLS: 1.3 (2018): more limited cipher suite choice than TLS 1.2 (2008)
  - only 5 choices, rather than 37 choices
  - *requires* Diffie-Hellman (DHE and ECDHE) for key exchange, rather than DH and RSA
  - combined encryption and authentication algorithm ("authenticated encryption") for data rather than serial encryption, authentication
    - 4 based on AES
  - HMAC uses SHA (256 or 384) cryptographic hash function

TLS 1.3 handshake: 1 RTT



1. client TLS hello msg:
   - *guesses* key agreement protocol, parameters
   - indicates cipher suites it supports
2. server TLS hello msg chooses
   - key agreement protocol, parameters
   - cipher suite
   - server-signed certificate
3. client:
   - checks server certificate
   - generates key
   - can now make application request (e.g.., HTTPS GET)

TLS 1.3 handshake: 0 RTT



- initial hello message contains encrypted application data!
  - "resuming" earlier connection between client and server
  - application data encrypted using "resumption master secret" from earlier connection
- vulnerable to replay attacks!
  - maybe OK for get HTTP GET or client requests not modifying server state

Chapter 8 outline

8.1 What is network security?

8.2 Principles of cryptography

8.3 Message integrity and digital signatures

8.4 Authentication

8.5 Securing e-mail

8.6 Securing TCP connections: TLS

~~8.7 Network layer security: IPsec~~

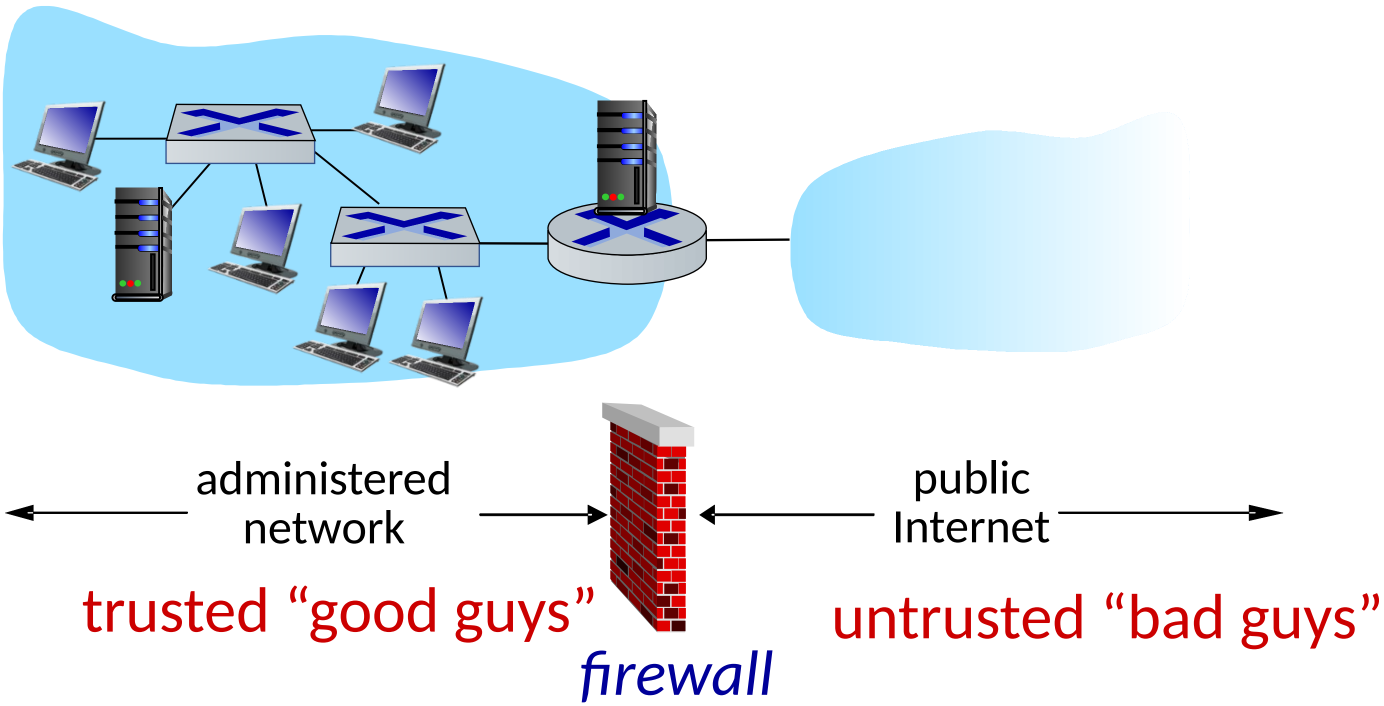
~~8.8 Security in wireless and mobile networks~~

8.9 Operational security: firewalls and IDS

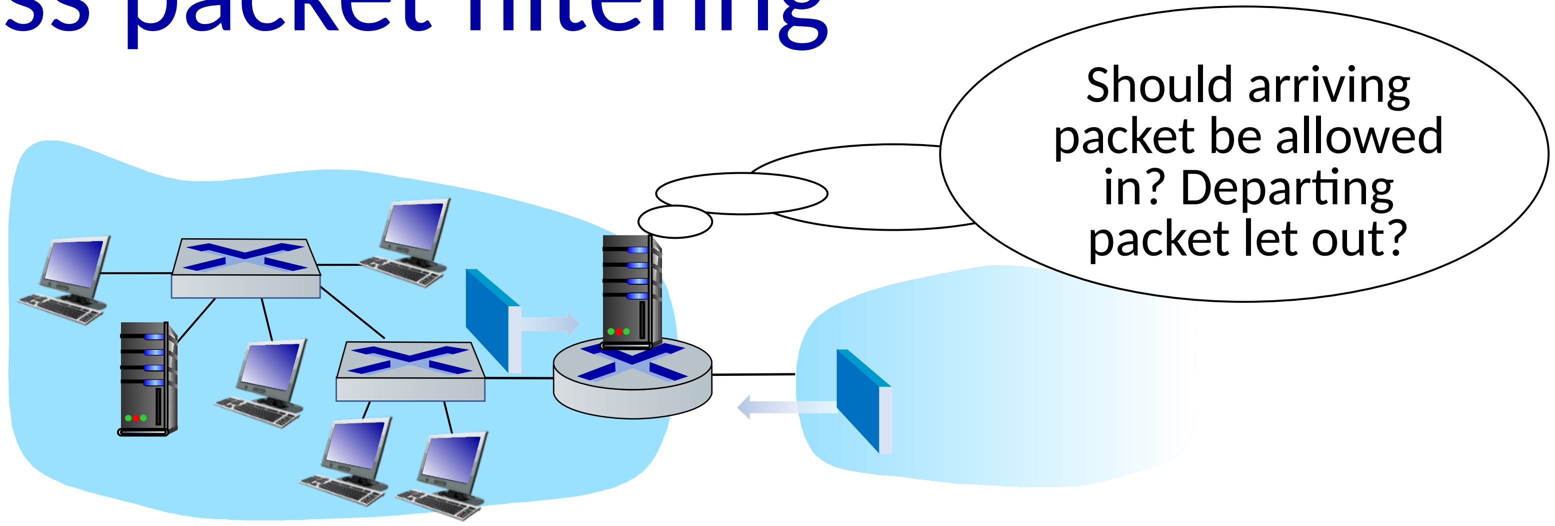
Firewalls

firewall

isolates organization's internal network from larger Internet, allowing some packets to pass, blocking others



Stateless packet filtering



- internal network connected to Internet via router firewall
- filters packet-by-packet, decision to forward/drop packet based on:
  - source IP address, destination IP address
  - TCP/UDP source, destination port numbers
  - ICMP message type
  - TCP SYN, ACK bits

Access Control Lists

ACL: table of rules, applied top to bottom to incoming packets: (action, condition) pairs looks like OpenFlow forwarding (Ch. 4)!

action	source address	dest address	protocol	source port	dest port	flag bit
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----
deny	all	all	all	all	all	all

Stateful packet filtering

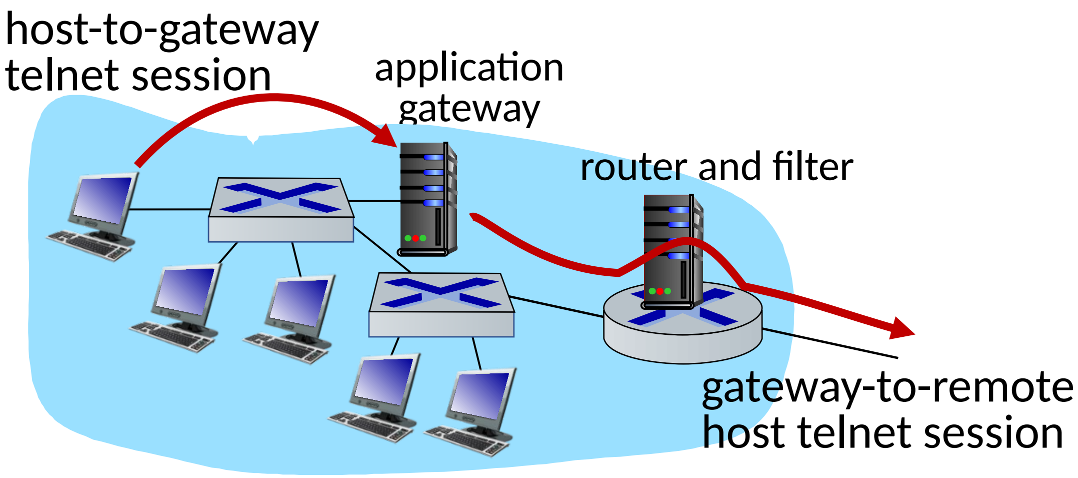
- *stateless packet filter:* heavy handed tool
  - admits packets that "make no sense," e.g., dest port = 80, ACK bit set, even though no TCP connection established:

action	source address	dest address	protocol	source port	dest port	flag bit
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK

- *stateful packet filter:* track status of every TCP connection
  - track connection setup (SYN), teardown (FIN): determine whether incoming, outgoing packets "makes sense"
  - timeout inactive connections at firewall: no longer admit packets

Application gateways

- filter packets on application data as well as IP/TCP/UDP fields
- *example:* allow select internal users to telnet outside



1. require all telnet users to telnet through gateway
2. for authorized users, gateway sets up telnet connection to dest host
   - gateway relays data between 2 connections
3. router filter blocks all telnet connections not originating from gateway

Intrusion detection systems

- packet filtering:
  - operates on TCP/IP headers only
  - no correlation check among sessions
- IDS: intrusion detection system
  - deep packet inspection: look at packet contents (e.g., check character strings in packet against database of known virus, attack strings)
  - examine correlation among multiple packets
    - port scanning
    - network mapping
    - DoS attack

Network Security (summary)

basic techniques……

- cryptography (symmetric and public key)
- message integrity
- end-point authentication

…. used in many different security scenarios

- secure email
- secure transport (TLS)
- ~~IP sec~~
- ~~802.11, 4G/5G~~

operational security: firewalls and IDS