Communication in Electronic Systems

Lecture 9: Introduction to Security in Communication Systems

Lecturer: Petar Popovski

TA: Junya Shiraishi, João H. Inacio de Souza

email: petarp@.es.aau.dk

AALBORG UNIVERSITY
DENMARK

Connectivity

Course Overview: Part 2. Communication and Networking

- MM5: Introduction to Communication Systems
- MM6: Simple Multiuser Systems and Layered System Design
- MM7: Network Topology and Architecture
- MM8: Networking and Transport Layers
- **MM9: Introduction to Security**
- Guest lecture
- MM10: Packets and Digital Modulation
- MM11: Communication Waveforms
- MM12: Workshop on Modulation and Link Operation

outline

1. security requirements, threat models and security risk analysis
2. basic cryptographic methods
    - encryption
    - authentication
    - integrity protection
3. cryptographic protocols
    - key agreement: Diffie-Hellman
    - no-key protocol, mental poker, simple digital cash

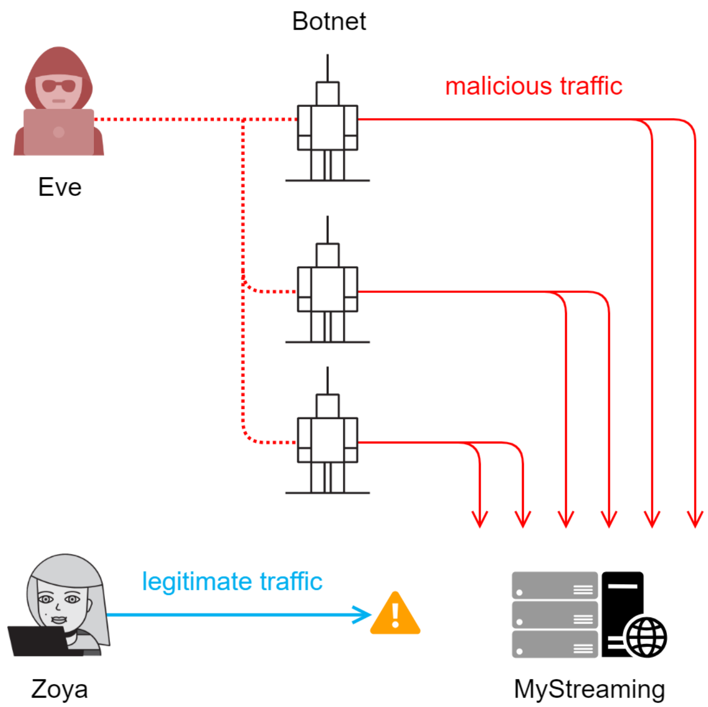
security: main requirements

- authentication: the communication parties want to be sure that the other party is indeed the one claimed.
- confidentiality: only sender and receiver shall be able to read the transferred data.
- privacy/anonymity: personal information (including own identity) should not be revealed.
- integrity: assurance that data has not been changed on the way from the sender to the receiver.
- availability: network and services shall be available whenever needed.
- non-repudiation: a user cannot deny having used a certain service.
- legal requirements: country specific legal security requirements (e.g. Legal interception, etc.)
- double spending: ensure that digital money is spent only once (the main invention in Bitcoin).

your experience on security attack?

the largest DDoS attack (until now) (1)

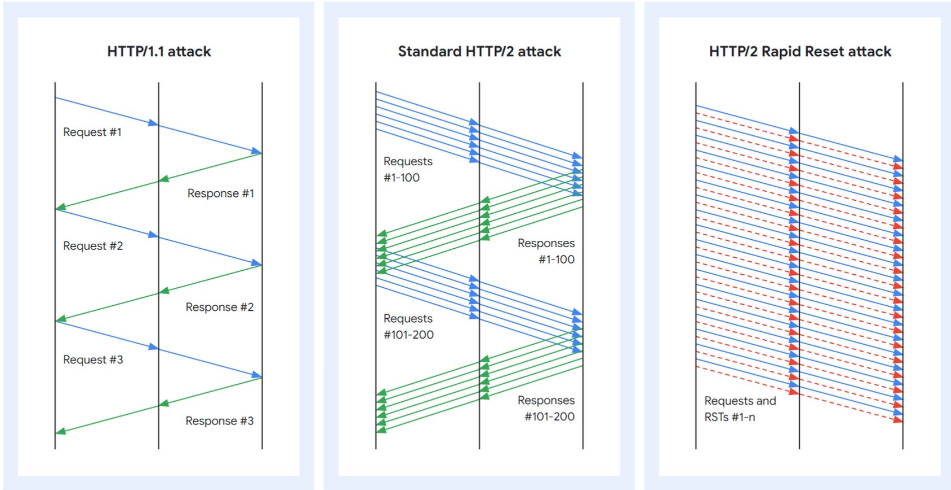
- distributed denial-of-service (DDoS)
- disrupt a server, service, or network by producing huge amounts of illegitimate requests
- botnet
  - machines infected by malware
  - controlled by the attacker
- solutions
  - blackhole routing
  - rate limiting
  - application firewall



the largest DDoS attack (until now) (2)

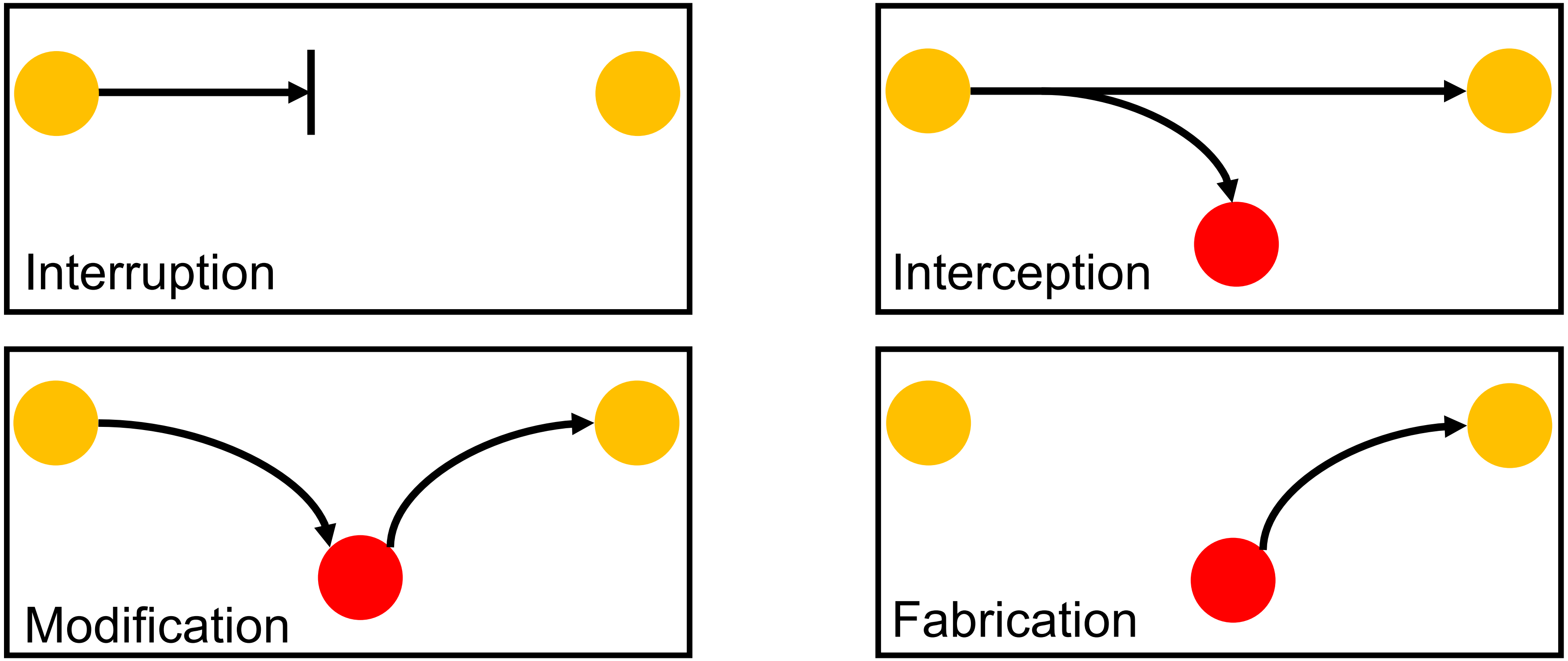
- Google, August 2023
  - 398 million requests per minute
  - 2-minute attack
  - more requests than 1 month of Wikipedia

- HTTP/2 rapid request technique
  - attack at the application layer
  - up to 100 parallel requests/connection
  - RST_REQUEST frame
  - a client can cancel a request unilaterally



- mitigation
  - we cannot simply ban clients that use the RST_REQUEST frame
  - track connection statistics
  - ban clients that cancel a high fraction of requests

security attacks



passive vs active attacks

- passive attacks
  - difficult to detect -> prevention, not detection
  - release of message contents
  - traffic analysis

- active attacks
  - masquerade
  - replay
  - modification of messages
  - denial of service

how to make a system secure?

- why increased focus on security when
  a system contains a communication network part?
  - system becomes open
  - information is transmitted over an open network (internet, large user community, flexibility)
  - Off-the-Shelf (OTS) technology
  - wireless networks (easy access to medium, mobility)

from threat analysis to security solutions

- threat analysis
  - **what attacks** can occur for the system?
  - what is the **impact** of the attack?
  - how **difficult** is it for different user groups (internal/external/OAM staff) to perform the attack (~how likely is the attack)?
- security solutions
  - which **methods/technologies** can prevent a certain attack?
  - how **costly** are the solution approaches?
  - what **limitations** are created by the security solution?
- not all attacks can be prevented
  - adequate logging for post-analysis
  - intrusion detection and alarming
  - contingency plans
- security is only partially addressed by technology;
  adequate processes are equally important

threat/risk analysis

- early discovering potential vulnerabilities that could be fixed
- estimating the associated risk to the business
  and the resources available to fix the vulnerabilities
- OWASP Risk Rating Methodology: https://owasp.org
  but remember modifications to address the specific needs
- two factors are taking into account in this risk model: ***likelihood*** and ***impact***
  - Risk = likelihood x impact
  - Step 1: Identifying a risk
  - Step 3: Factors for estimating likelihood
  - Step 3: Factors for estimating impact
  - Step 4: Determining the Severity of the Risk

estimating likelihood

- threat agent factors
    - skill level
    - motive
    - opportunity - resources
    - size
- vulnerability factors
    - ease of discovery
    - ease of exploit
    - awareness
    - intrusion detection

estimating impact

- technical impact factors
    - loss of privacy, integrity, availability, accountability
- business impact factors
    - financial damage
    - reputation damage
    - privacy violation

determining severity of the risk

Likelihood and Impact Levels	
0 to <3	LOW
3 to <6	MEDIUM
6 to 9	HIGH

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
		Likelihood		

basic cryptographic methods

terminology

- plaintext - original message
- ciphertext - coded message
- cipher - algorithm for transforming plaintext to ciphertext
- key – secret information used in cipher known only to sender/receiver
- encipher (encrypt) - converting plaintext to ciphertext
- decipher (decrypt) - recovering ciphertext from plaintext
- cryptography - study of encryption principles/methods
- cryptanalysis (code breaking) –
  study of principles/methods of deciphering ciphertext without knowing key

encryption

- **the encryption security depends on the secrecy of the key, not the secrecy of the algorithm**

- sender and the receiver have obtained the copy of the secret key in a secure fashion and must keep the key secure

- practical reasons – makes it feasible for widespread use.
- manufacturers can and have developed low-cost chip implementations of data encryption algorithms.
  - widely available and incorporated into a number of products.

cryptographic systems classified along 3 independent dimensions

- the type of operations used for transforming plaintext to ciphertext
- the types of keys used
- the way in which the plaintext is processed

the type of operations used for transforming plaintext to ciphertext

- **substitution**
  - each element in the plaintext is mapped into another element
  - example of a key for shift-by-3 “Caesar’s cipher”

plaintext	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
ciphertext	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

  - fourscoreandsevenyearsago -> IRXUVFRUHDQGVHYHQBHDUVDJR
  - how many keys of this type exist? how would you go on to break them?

the type of operations used for transforming plaintext to ciphertext

- **transposition -** elements in the plaintext are rearranged; fundamental requirement is that no information be lost.
- **product systems -** multiple stages of substitutions and transpositions

plaintext: attackxatxdawn   ciphertext: xtawxnattxadakc

	col 1	col 2	col 3
row 1	a	t	t
row 2	a	c	k
row 3	x	a	t
row 4	x	d	a
row 5	w	n	x

permute rows and columns

$\Rightarrow$

	col 1	col 3	col 2
row 3	x	t	a
row 5	w	x	n
row 1	a	t	t
row 4	x	a	d
row 2	a	k	c

key: matrix size and permutations

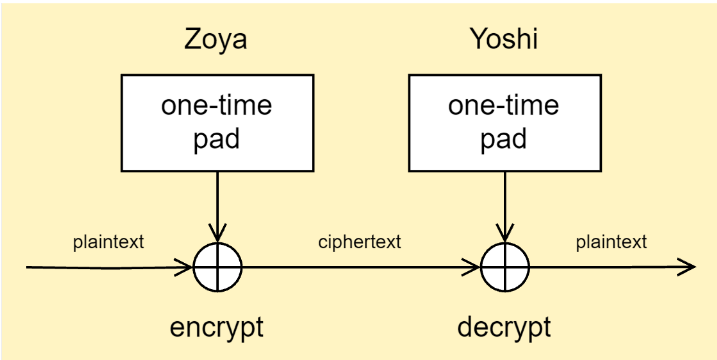
cryptographic systems classified along 3 independent dimensions

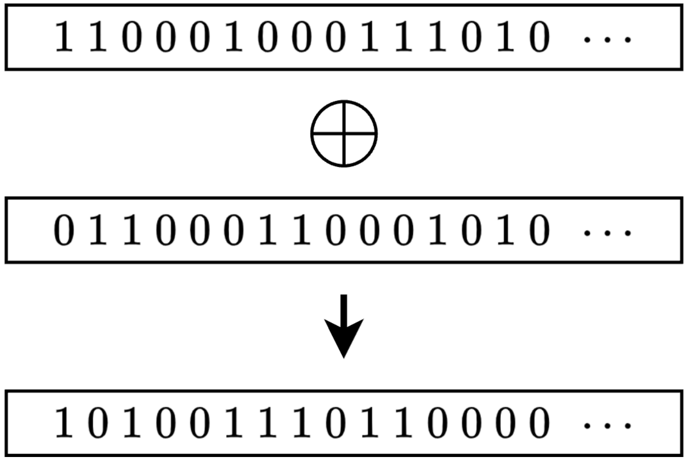
- the type of operations used for transforming plaintext to ciphertext

- the types of keys used
  - **symmetric,** if both sender and receiver use the same key
  - **asymmetric,** or public-key encryption if the sender and receiver each use a different key

- the way in which the plaintext is processed
  - **block cipher** processes the input one block of elements at a time, producing an output block for each input block
  - **stream cipher** processes the input elements continuously, producing output one element at a time, as it goes along

one-time pad





- A stream cipher with infinite long random sequence as a key is the only provable secure scheme

one-time pad: practical key exchange

- physical layer security
- Quantum Key Distribution

attack type	what is known by the cryptanalyst
Ciphertext only	•Encryption algorithm •Ciphertext to be decoded
Known plaintext	•Encryption algorithm •Ciphertext to be decoded •One or more plaintext-ciphertext pairs formed with the secret key
Chosen plaintext	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen ciphertext	•Encryption algorithm •Ciphertext to be decoded •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen text	•Encryption algorithm •Ciphertext to be decoded •Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key •Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

- computationally secure scheme
  - the cost of breaking the cipher exceeds the value of the encrypted information
  - the time required to break the cipher exceeds the useful lifetime of the information

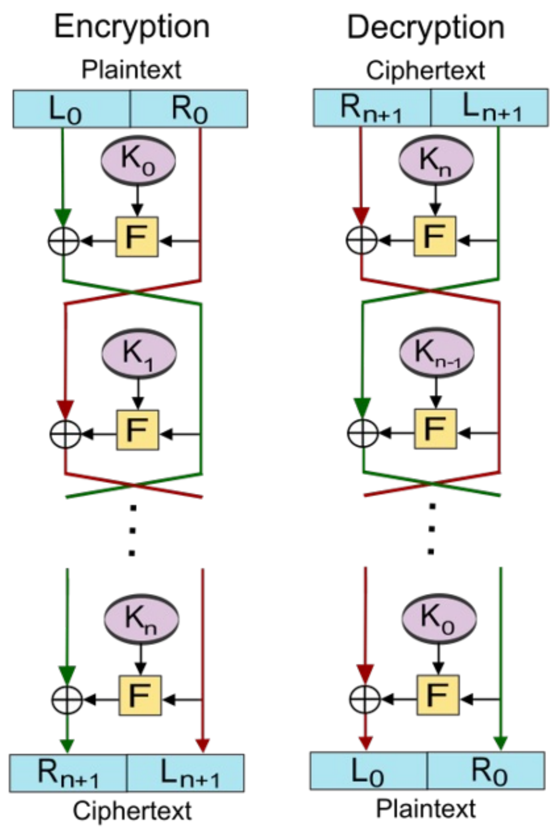
block cipher

- plaintext and ciphertext consist of fixed-sized blocks
- ciphertext obtained from plaintext by iterating a **round function**
- input to round function consists of key and output of previous round
- usually implemented in software

Feistel cipher structure

- all conventional block encryption algorithms have this structure
  - not a specific block cipher, but rather a blueprint

- parameters for a concrete realization:
  - block size, e.g. 64 bits
  - key size, e.g. 128 bits
  - number of rounds, e.g. 16
  - subkey generation algorithm
  - round (inner) function F



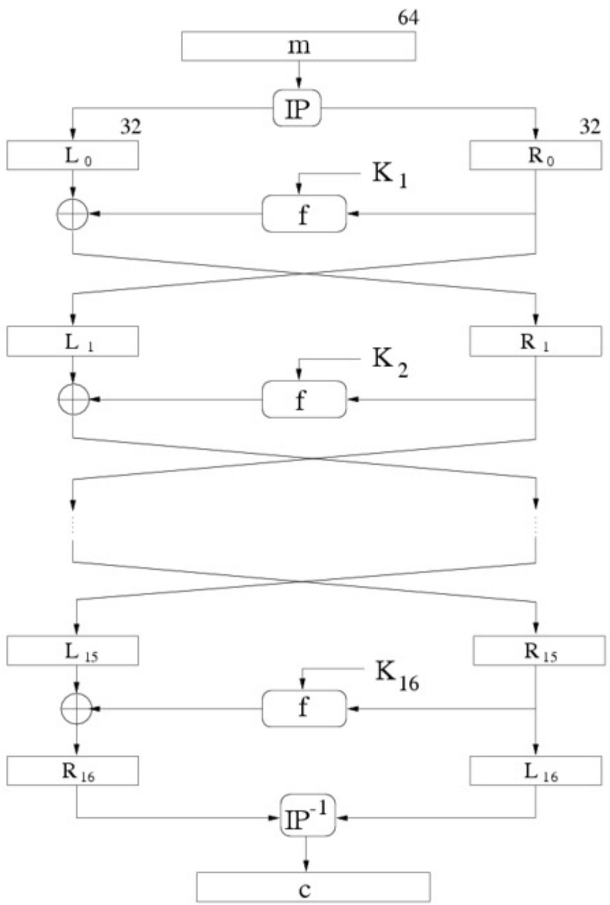
symmetric block encryption algorithms

- Data Encryption Standard (DES)
- Triple DES (3DES)
- Advanced Encryption Standard (AES)
- Blowfish
- RC5

data encryption standard (DES)

- DES is a Feistel cipher with…
  - 64 bit block length
  - 56 bit key length
  - 16 rounds
  - 48 bits of key used each round (subkey)

- Advanced Encryption Standard (AES)
  - block size: 128 bits
  - key length: 128, 192 or 256 bits
  - 10 to 14 rounds (depends on key length)



block cipher modes of operation

- a symmetric block cipher processes one block of data at a time
  - In the case of DES and 3DES, the block length is b=64 bits
  - For AES, the block length is b=128
- for longer amounts of plaintext,
  it is necessary to break the plaintext into b bit blocks, padding the last block if necessary

- questions
  - How to encrypt multiple blocks?
  - Do we need a new key for each block?
  - Encrypt each block independently?
  - How to handle partial blocks?

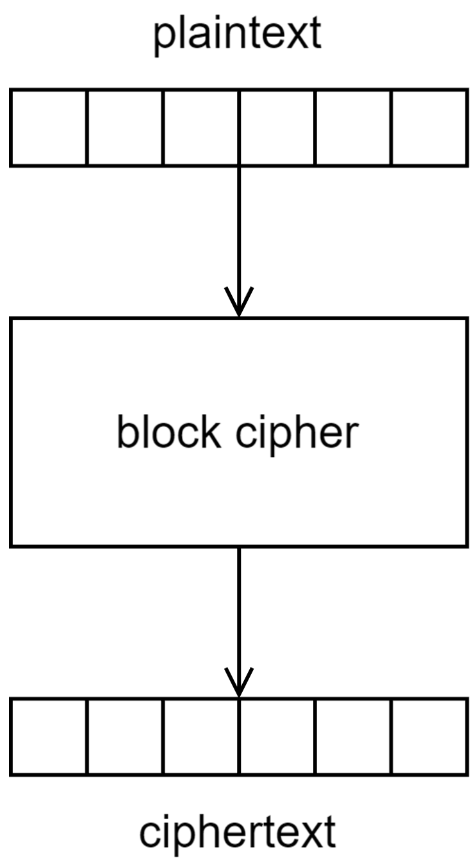
Electronic Code Book (ECB)

- notation: C = E(P, K)
- given plaintext P0, P1, …, Pm, …
- most obvious way to use a block cipher:

Encrypt	Decrypt
C0 = E(P0, K)	P0 = D(C0, K)
C1 = E(P1, K)	P1 = D(C1, K)
C2 = E(P2, K) …	P2 = D(C2, K) …

the trouble with ECB

- if the same b-bit block of plaintext appears more than once in the message, it always produces the same ciphertext
- due to this, for lengthy messages, the ECB mode may not be secure
- if the message is highly structured, it may be possible for a cryptanalyst to exploit these regularities



cipher block chaining (CBC)

- blocks are “chained” together
- a random initialization vector (IV), is required to initialize CBC mode
- IV is random, but not secret
- trouble: if a block/packet is lost, then all subsequent cannot be decrypted
  - recall our flow control and transport protocols from the last time

Encryption	Decryption
C0 = E(IV $\oplus$ P0, K),	P0 = IV $\oplus$ D(C0, K),
C1 = E(C0 $\oplus$ P1, K),	P1 = C0 $\oplus$ D(C1, K),
C2 = E(C1 $\oplus$ P2, K),…	P2 = C1 $\oplus$ D(C2, K),…

public-key cryptography

- two keys, one to encrypt, another to decrypt
  - Alice uses Bob’s public key to encrypt
  - Only Bob’s private key decrypts the message

- based on “trap door, one way function”
  - “One way” means easy to compute in one direction, but hard to compute in other direction
  - Example: Given p and q, product N = pq easy to compute, but hard to find p and q from N
  - “Trap door” is used when creating key pairs

- encryption
  - Suppose we encrypt M with Bob’s public key
  - Bob’s private key can decrypt C to recover M

- digital signature

public-key cryptography

- each party has a pair of keys: K1 is the public key and K2 is the secret key, such that $D_{K2}(E_{K1}(M))=M$
- knowing the public-key and the cipher, it is computationally infeasible to compute the private key
  - thereby **asymmetric** crypto system
- the public-key K1 may be made publicly available
  - many can encrypt, only one can decrypt

- two parties who do not share any private information through communications arrive at some secret not known to any eavesdroppers
  - use it to share a secret key

RSA (Rivest, Shamir and Adleman 1978)

- based on difficulty of determining prime factors of large numbers
- approach
  - select secret primes p,q (>100 decimal digits)
  - communicate N=pq, the modulus
  - choose e relatively prime to $(p-1)(q-1)$
  - find d such that $ed = 1 \bmod (p-1)(q-1)$
  - public key is (N,e)
  - private key is d
- encryption and decryption
  - Encryption: $c = m^e \bmod N$
  - Decryption: $m = c^d \bmod N$

a simple RSA example

- generate RSA key par
  - select “large” primes $p = 11$, $q = 3$
  - then $N = pq = 33$ and $(p - 1)(q - 1) = 20$
  - choose $e = 3$ (relatively prime to 20)
  - Find d such that $ed = 1 \bmod 20$. we find that $d = 7$ works
  - Public key: $(N, e) = (33, 3)$, Private key: $d = 7$

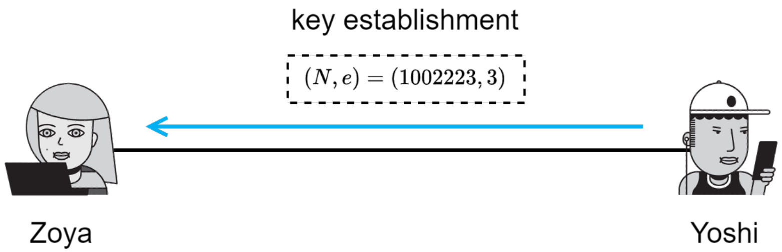
- suppose the message to encrypt is $M = 8$
- ciphertext C is computed as
  $C = M^e \bmod N = 8^3 = 512 = 17 \bmod 33$
- decrypt C to recover the message M by

  $M = C^d \bmod N = 17^7 = 410{,}338{,}673 = 12{,}434{,}50 * 33 + 8 = 8 \bmod 33$

RSA cube root attack (1)

- efficient RSA cryptosystems
  - encryption exponent $e = 3$
  - fast encryption with just 2 multiplications
  - condition for the cube root attack: $M < N^{1/3}$

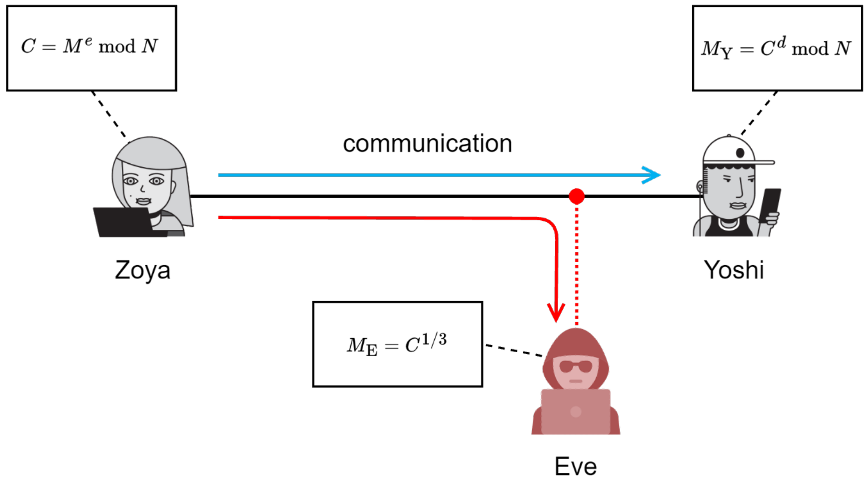
- public and private keys
  - $(N, e) = (1002223{,}3)$
  - $d = 661467$



- checking if the keys are valid
  - $N = pq$, $p = 101$, $\mathrm{q} = 9923$, p and q are primes $\Rightarrow N$ is a valid modulus
  - $(p-1)(q-1) = 992200$ and $e = 3$ are coprimes $\Rightarrow e$ is a valid encryption exponent
  - $ed \bmod (p-1)(q-1) = 1 \ \Rightarrow d$ is a valid decryption exponent

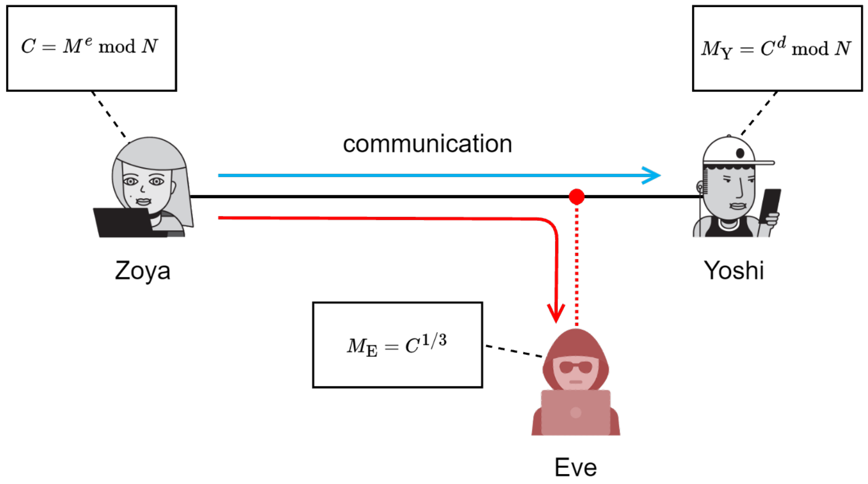
RSA cube root attack (2)

- transmitting ASCII characters
  - $\mathrm{A} \equiv 01000001_2 \equiv 65_{10}$
  - $\mathrm{U} \equiv 01010101_2 \equiv 85_{10}$
- Zoya transmits 3 messages: A, A, U
- Eve can obtain the messages without the decryption exponent
  - $M < N^{1/3}$, that is $85 < 1002223^{1/3} \approx 100$



Character	Bits	Plaintext (M)	Cipher (C)	Yoshi (M_Y)	Eve (M_E)
A	01000001	65	274625	65	65
A	01000001	65	274625	65	65
U	01010101	85	614125	85	85

RSA cube root attack (3)



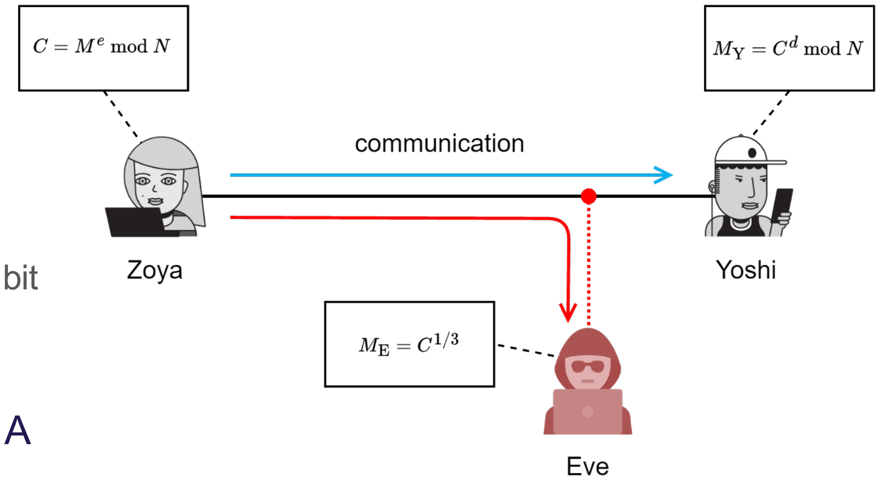
- padding
  - adding extra bits to the message
  - beginning, middle, or end of the message
  - padding bits are discarded by the receiver

- Zoya adds a bit 0 at the end of each message

- Eve cannot obtain the messages without the decryption exponent
  - $M > N^{1/3}$, that is $130 > 1002223^{1/3} \approx 100$

Character	Bits	Plaintext (M)	Cipher (C)	Yoshi (M_Y)	Eve (M_E)
A	010000010	130	192554	130	57.74
A	010000010	130	192554	130	57.74
U	010101010	170	904108	170	96.69

RSA cube root attack (4)



- obfuscation
  - the last bit is always discarded by Yoshi
  - so Zoya can randomize the value of the padding bit

- Zoya adds a bit 0 at the end of the first A
- then Zoya adds a bit 1 at the end of the second A

- the padding bit changed the cipher, so Eve cannot detect a message repetition

Character	Bits	Plaintext (M)	Cipher (C)	Yoshi (M_Y)	Eve (M_E)
A	010000010	130	192554	130	57.74
A	010000011	131	243645	131	62.45
U	010101010	170	904108	170	96.69

public key infrastructure (1)

- digital certificate contains name of user and user's public key (possibly other info too)
- signed by the issuer, a Certificate Authority (CA), such as VeriSign
  - M = (Alice, Alice's public key), S = $[M]_{CA}$
  - Alice's Certificate = (M, S)
- signature on certificate is verified using CA's public key
  - must verify that M = $\{S\}_{CA}$
- certificate authority (CA) is a trusted 3rd party (TTP):
  - creates and signs certificates

public key infrastructure (2)

- the collection of elements needed to securely use public key crypto
  - Key generation and management
  - Certificate authority (CA) or authorities
  - certificate revocation lists (CRLs), etc.

- no general standard for PKI, but three generic trust models
  - monopoly model: universally trusted CA
  - oligarchy of multiple trusted CAs, used in browsers
  - anarchy model: everyone is a CA, users decide whom to trust

integrity protection

- **integrity**: detect unauthorized writing (i.e., detect unauthorized mod of data)
  - example: Inter-bank fund transfers

- Message Authentication Code (MAC)
  - used for data integrity. not the same as confidentiality!

- MAC can be computed as CBC (Cipher Block Chaining) residue
  - $C_0 = E(IV \oplus P_0, K)$,
  - $C_1 = E(C_0 \oplus P_1, K)$,
  - $C_2 = E(C_1 \oplus P_2, K)$,…
  - $C_{N-1} = E(C_{N-2} \oplus P_{N-1}, K)$ = MAC

  - send IV, P_0, P_1, …, P_{N-1} and MAC
  - both sender and receiver need to know K (symmetric key!)

integrity protection with hash functions

- Cryptographic Hash Functions, *‚Fingerprint‘*: h=H(m)
  - For given m, h=H(m) efficiently computable
  - $M=H^{-1}(h)$ not efficiently computable
  - For given m and h=H(m): difficult to find m_2 with $H(m_2)=h=H(m)$
- a hash uniquely represents a given text
  and it is difficult to produce another text that hashes to the same value
- any change to the message after the digest has been perform is detectable
- examples: Message Digest 5 (MD5), Secure Hash Function (SHA)
- ideal hash function: random oracle
  - gives a fully random answer to every query
  - provides the same answer for the same query
  - useful in analyzing hash functions

cryptographic hash function: properties

- it is deterministic so the same message always results in the same hash
- it is quick to compute the hash value for any given message
- it is infeasible to generate a message from its hash value
  except by trying all possible messages
- a small change to a message should change the hash value so extensively that the new hash value appears uncorrelated with the old hash value
- it is infeasible to find two different messages with the same hash value

authentication

- symmetric authentication
- based on shared secret: key K
- problem: Key K must not be transmitted over channel in the authentication protocol
- Solution: Challenge-response methods
  - Scenario: B authenticates at A
    - A picks random number ‚rand' (the challenge) and sends to B
    - B computes cryptographic one-way function y=f(rand,K) and sends y to A, e.g. exponentiation $y = \text{rand}^K \bmod n$
    - A verifies whether the received y equals f(rand,K)

- Asymmetric Authentication
    - Based on pairs of private and public key
    - Public Key Infrastructures (PKI), certificates

replay attacks

- where a valid signed message is copied and later resent
- countermeasures include
  - use of sequence numbers (generally impractical)
  - timestamps (needs synchronized clocks)
  - challenge/response (using unique **nonce**)

- some questions
  - assume that two parties know each other's public keys
  - how to prevent a replay attack?
  - how to ensure that the messages are not delayed?

questions on replay attacks

- assume that two parties know each other’s public keys
  - If one message is sent from A to B, what can be verified?
  - if two messages are exchanged, what can be verified?

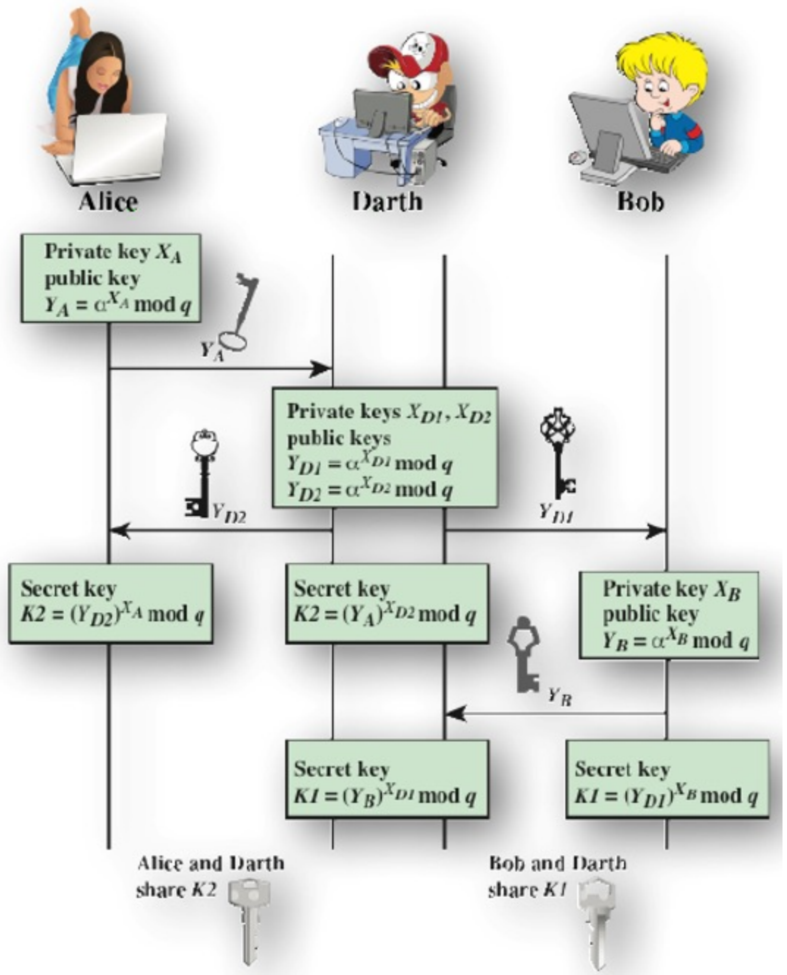
- how to prevent a replay attack?
- how to ensure that the messages are not delayed?

cryptographic protocols

cryptographic protocols: key agreement

- Task: Agreement on joint, secret session key k
- Diffie-Hellman key exchange
  - invented by Williamson (GCHQ) and, independently, by D and H (Stanford)
  - a "key exchange" algorithm, used to establish a shared symmetric key
  - based on discrete log problem
    - **given:** g, p, and g^k mod p**, find:** exponent k
    - this problem is hard
  - Steps
    1. A and B agree on prime number p and integer g (can be publicly known)
    2. A selects secret a, B selects secret b
    3. A computes $\alpha = g^a$ mod p and sends α to B,
       B computes $\beta = g^b$ mod p and sends β to A
    4. A computes k= β^a mod p , B computes k= α^b mod p
    5. A and B can communicate with secret session key k

D-H: man-in-the middle attack



no-key protocol: Shamir‘s three-pass protocol

- confidential message transmission without shared or public/private keys
- principle: A wants to send m to B
  1. A encrypts m with its ‚local key‘ $c_A = E_A(m, k_A)$
  2. A transmits c_A to B
  3. B encrypts received message with its own ‚local key‘: $c_{AB} = E_B(c_A, k_B)$
  4. B sends c_{AB} to A
  5. A decrypts the message with its key and sends result back to B
  6. B decrypts received value with its own key

– B has obtained the message m when the encryption operations commute i.e., $E_B(E_A(m,k_A), k_B) = E_A(E_B(m,k_B), k_A)$ for all m

– Example: exponentiation in finite fields

conclusions and outlook

- we have defined the context for security/attacks in digital systems
  - only a sample considered, vast area
- basic cryptographic models
  - symmetric/secret key
  - asymmetric/public key
- elements of cryptographic protocols

BACKUP

examples of security threats

- eavesdropping messages
- modifying messages on their path from sender to receiver
- using somebody else's identity
- manipulate charging
  - use services without payment or with payment from third person's account
  - 'overcharge' third persons account (without use of services)
- block certain functionality (Denial of Service Attacks)

- possible origin/point of attack
  - via external Interfaces: e.g., connection to Internet
  - while passing through un-trusted intermediate networks (e.g. backbone connecting site networks)
  - air interface/wireless links
  - malicious processes/users within the distributed system
    - viruses, worms, etc.
  - distributed attacks, e.g. via botnets
  - network management/administration