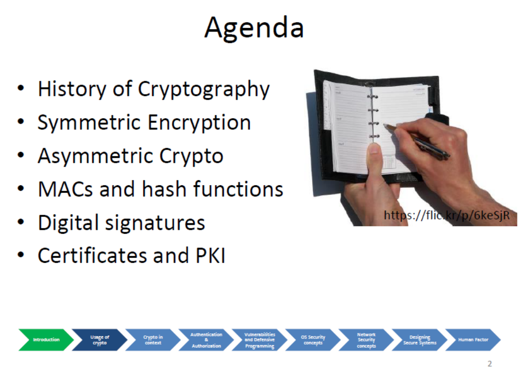
Use the homework template

Cryptography

General ideas without math fo the field

CIA – last class

Confidentiality



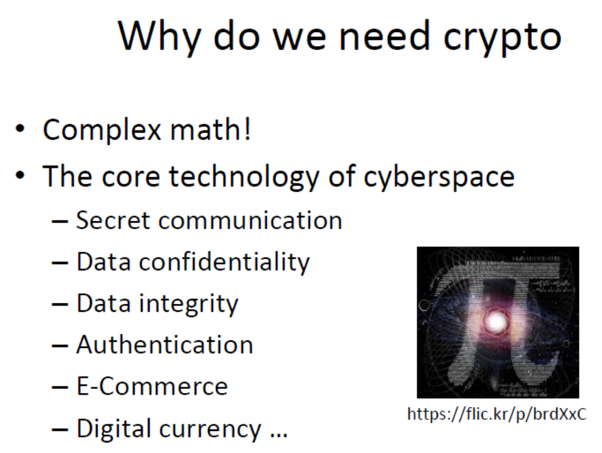
Agenda

- History of Cryptography
- Symmetric Encryption
- Asymmetric Crypto
- MACs and hash functions
- Digital signatures
- Certificates and PKI

https://flic.kr/p/6keSjR

Introduction | Usage of crypto | Crypto in context | Authentication & Authorization | Vulnerabilities and Defensive Programming | OS Security concepts | Network Security concepts | Designing Secure Systems | Human Factor

2



Why do we need crypto

- Complex math!
- The core technology of cyberspace
  - Secret communication
  - Data confidentiality
  - Data integrity
  - Authentication
  - E-Commerce
  - Digital currency …

https://flic.kr/p/brdXxC

Avoid complex math

- Other courses for that

Why do we need it?

- Keep secrets
  - Communication etc

Relevant to big and small companies

Everywhere but don't notice it

History of Cryptography

- Egypt, China
- אתב"ש בספר ירמיהו
- A cipher or cryptosystem is used to encrypt the plaintext
- The result of encryption is ciphertext
- We decrypt ciphertext to recover plaintext

Heirographs

And china

- Writing language was secret
- Only kings and high level personnel knew how to read and write that was the secret

Hebrew language

- Yirmiyahu
- Switch between letters
- First letter is last
- Babel is shashach
- A = Z
- Good for hiding things

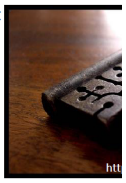
Idea:

Plain text transform to cipher text

- Hide real message

Key

- Need strong cipher
  - Need many different strong ciphers!
  - How to make someone to forget the cipher?
- A key is used to configure a cryptosystem



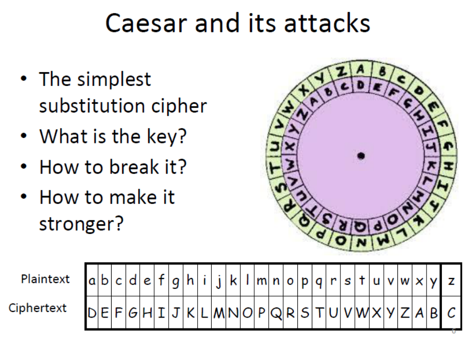
Strong cipher

- Atbash
- Caeser

But idea is that if we want to change things its not flexible

- We need to enable it to be flexible

- If we change it to exchange messages secretly but then we decide to work with someone else
- We don't want him to tell the code
- Our cipher needs something to change easily without breaking system / cipher / code



Caesar and its attacks

- The simplest substitution cipher
- What is the key?
- How to break it?
- How to make it stronger?

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

-
- We need a key, which we can change
- Use different key for cipher to look different
- Used to configure system

Famous cipher is caeser cipher

Idea is each letter gets different value, key is 5 then a becomes f

Caeser sed # 3 and always used the same number , not good … not really a key as it was permanent but he knew how

A+3= D

How to break the cipher

Find the key….

Bruteforcing – guess 1 out of 26 till get to it

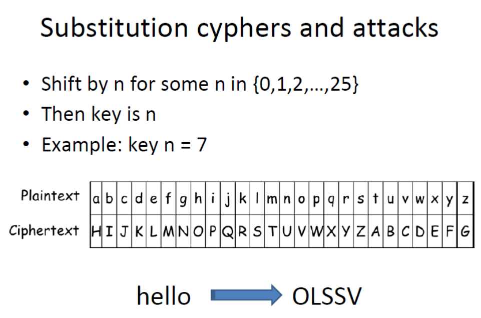
How to strengthen it?

- 3+4 = 7 and in group of keys
- 26 keys
    - Quite easy to break

Only 26 letters… assign letters differently…not in order

Chose a random distribution of letters

Scramble it all

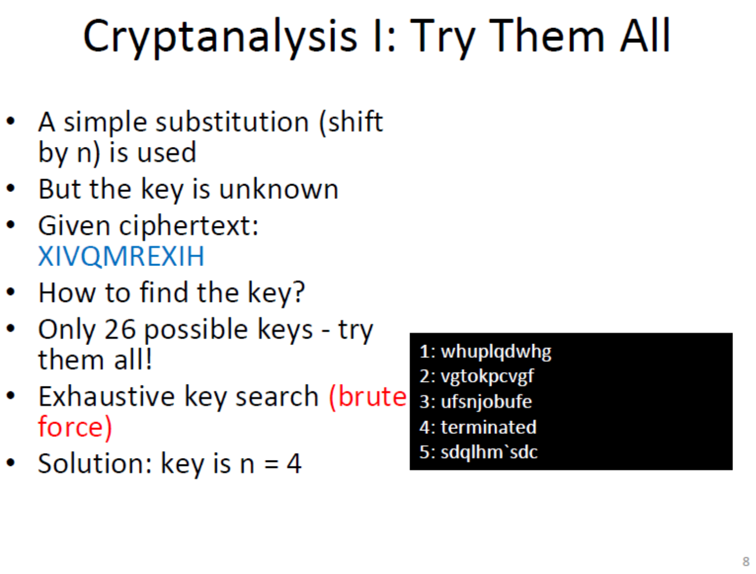


Substitution cyphers and attacks

- Shift by n for some n in {0,1,2,…,25}
- Then key is n
- Example: key n = 7

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	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G

hello $\longrightarrow$ OLSSV



Cryptanalysis I: Try Them All

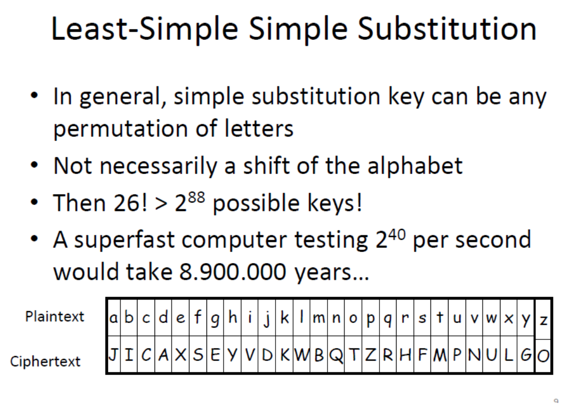
- A simple substitution (shift by n) is used
- But the key is unknown
- Given ciphertext: XIVQMREXIH
- How to find the key?
- Only 26 possible keys - try them all!
- Exhaustive key search (brute force)
- Solution: key is n = 4

```
1: whuplqdwhg
2: vgtokpcvgf
3: ufsnjobufe
4: terminated
5: sdqlhm`sdc
```

8

Keys are known but can bruteforce over all 26 choices then to break it

Make things more complex



Least-Simple Simple Substitution

- In general, simple substitution key can be any permutation of letters
- Not necessarily a shift of the alphabet
- Then 26! > 2^{88} possible keys!
- A superfast computer testing 2^{40} per second would take 8.900.000 years…

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	J	I	C	A	X	S	E	Y	V	D	K	W	B	Q	T	Z	R	H	F	M	P	N	U	L	G	O

9

Then don’t have to shift the alphabet but scramble the letters and then factorial of 26

To make it more difficult

Random scramble

26 letters = 26 possibilities

1st letter 26 options, second 25, etc….

Many possibilities

Caeser but made it more powerful

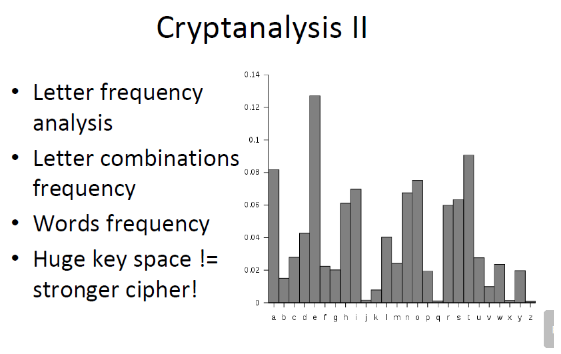
How to get plaintext back

- Bruteforce with every possibility
    - Tool is a computer
    - But a superfast computer still would take 9 million years

Stuck here

Options:

- Language is limited , use linguistics to solve it
- Frequency of letters isn't the same



Cryptanalysis II

- Letter frequency analysis
- Letter combinations frequency
- Words frequency
- Huge key space != stronger cipher!

-
- Vowels and groups such as I,G etc
- Next exercise we will see how it works
- Z is rare compared to E and T

Key space = 2^88 doesn't mean that its stronger per say

Easy to break

Types of crypto attacks

- Cipher text only (Caesar)
- Known plaintext (Enigma, RC4)
- Chosen plaintext
- Chosen ciphertext
- Kerckhoffs (1883) Algorithm can't stay secret (Enigma, A5, DVD, US Navy)
  - Security through obscurity

Radiation at B-52 Crash Site
Los Angeles Times
FINAL
U.S. SHIPS HEADING FOR NORTH KOREA

Enigma – encryption machine for ww2 with germans

British army intercepted and tried to break the enigma cipher

Used known plaintext

- Germans were precise and every day at same time broadcasted the weather forecast
  - British army understood it and it helped them to decipher
- chosen plaintext
  - British army trick, they bombed specific places in Germany with unique names
  - Germans broadcasted that with enigma cipher and then the British used those names to get the cipher
- Chosen ciphertext
  - Digital converters for tv
    - Gets some scrambled signal
    - It then translates to plaintext
  - If you have the cipher text and translation you can find the key

Kerckhoff

- Dutch
- Suggested something strong and important, later in course
- Algo cannot stay secret if it is secret it is more insecure
  - Eventually it will be leaked or discovered
  - Security through obscurity
  - If not secured by itself then find the secret etc.
  - Find strong algo then publish, and will stay strong still

Enigma was secret machine but British army broke it

Another case which will refer later with Korean war with Russians

Cryptography wasn't defined till Shannon

- Does entropy

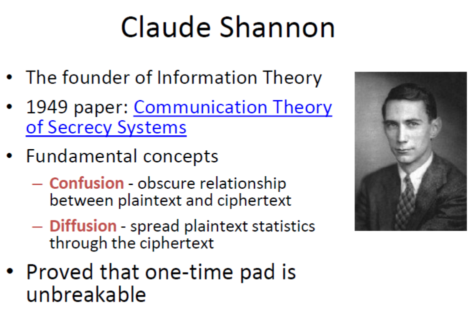
Confusion

Obscure relationship between plaintext and cipher text

Don't want to deduce from cipher text what is plain text

Diffusion

- No statistic relationship in the ciphertext
- If we make a change in the plaintext, get a completely different ciphertext
- Switch 1 letter in plaintext but get a 100% different cipher text otherwise it will be easier to reverse/crack
-

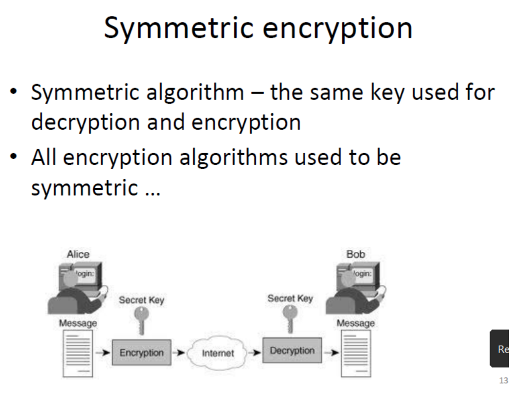


Claude Shannon

- The founder of Information Theory
- 1949 paper: Communication Theory of Secrecy Systems
- Fundamental concepts
  - **Confusion** - obscure relationship between plaintext and ciphertext
  - **Diffusion** - spread plaintext statistics through the ciphertext
- Proved that one-time pad is unbreakable

Another contribution, will see later

- One time pad
  - Now we have something unbreakable we can rest
  - https://en.wikipedia.org/wiki/One-time_pad
  - But also an issue



Symmetric encryption

- Symmetric algorithm – the same key used for decryption and encryption
- All encryption algorithms used to be symmetric …

See it everywhere

Symmetric algo – same key for encryption and decryption

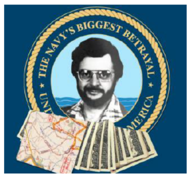
- All algos in the past used to be symmetric
- Someone
- Non repudation
  - Anyone can write a message
  - Bank message – say one employee wrote a message
  - Can deny the process
- Another technical problem
  - All need to have a key… one person wants to connect with 50 people.
    - Will need a key for each person
      - 1 key for writer and 1 for reader

Lots of keys is a problem

My friend is in the US and im in Israel – how to send the key secretly

Modern symmetric encryption

- Translates the bytes of plaintext into ciphertext
- DES, 3-DES and now AES are standard symmetric encryption algorithms
- Problem – key distribution (n*n keys for n pairs)
- Problem – key management
- Problem – non-repudiation



Will see in next meeting

We use bytes not letters

That way we can represent characters too

DES, 3DES – broken not safe

AES is safer to use

Key distribution – need many keys

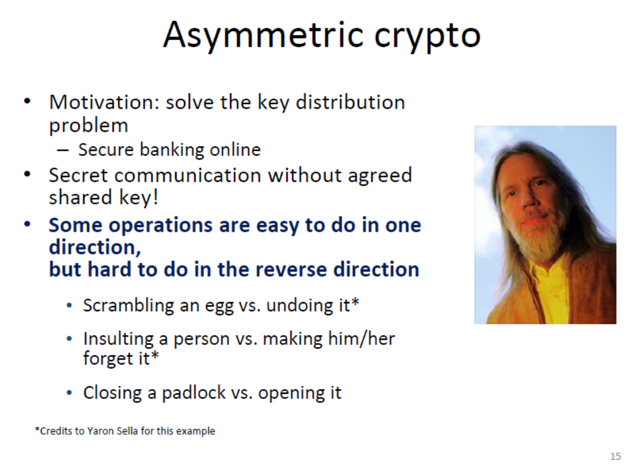
Key management

Non repudiation

Johnny walker – spy for Russians – saw old keys and could break us navy ciphers with those keys

- Lots of damage to us army
- The keys were old but Russians could still decrypt old ciphers

Solution:



Asymmetric crypto

- Motivation: solve the key distribution problem
  - Secure banking online
- Secret communication without agreed shared key!
- **Some operations are easy to do in one direction,**
  but hard to do in the reverse direction
  - Scrambling an egg vs. undoing it*
  - Insulting a person vs. making him/her forget it*
  - Closing a padlock vs. opening it

*Credits to Yaron Sella for this example

15

Algos that include few keys

And solve the other issues

Solve key distr problem

Secure banking online – saves money

RSA group member mentioned 3 billion dollar savings each year to bank savings for online using crypto

Secret comms without shared key

1 direction is easy

Other direction is hard

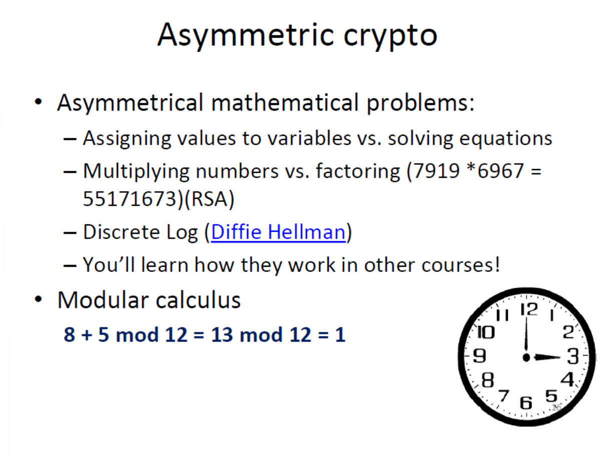
Easy to close padlock, anyone can close it but need a key to unlock it

- Idea is that we want to get mechanism which is easy to do for one side but not easy to get from the other side

Communicate with someone via sending message and closing in box then send to someone else who has the key to open it

Only 1 person with the key and the person seding the message just needs to lock it and send it back

Easy to close but hard to open



Asymmetric crypto

- Asymmetrical mathematical problems:
  - Assigning values to variables vs. solving equations
  - Multiplying numbers vs. factoring (7919 *6967 = 55171673)(RSA)
  - Discrete Log (Diffie Hellman)
  - You'll learn how they work in other courses!
- Modular calculus

  8 + 5 mod 12 = 13 mod 12 = 1

Easy to get from one side not the other

Easy to multiply 2 huge numbers and with calculator get a quick solution

We use prime numbers but if we get a huge number and ask what are the multipliers

Even for computers it's a hard task

RSA – S is one of the members and that's what they did

And also Diffie helkman (guy in pic) – another method

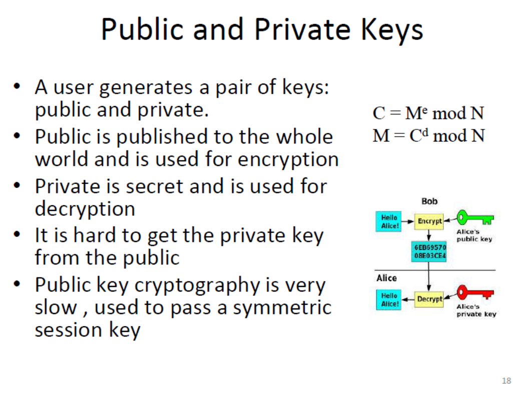
We got something hard to break and takes a long time – cryptography course gets details

We can use modular calc we can represent the sum to be equal to 1

8+5 / 12 used In clocks

12 hours and 13 is 1 pm

That's the idea



Public and Private Keys

- A user generates a pair of keys: public and private.
- Public is published to the whole world and is used for encryption
- Private is secret and is used for decryption
- It is hard to get the private key from the public
- Public key cryptography is very slow , used to pass a symmetric session key

$C = M^e \bmod N$
$M = C^d \bmod N$

Bob
Hello Alice!
Encrypt
Alice's public key
6EB69570 08E03CE4
Alice
Hello Alice!
Decrypt
Alice's private key

18

Alice and bob

- Bob sends alice a message

- Lock is another key – public one
    - Send me a secret message and use public key

Different than symettric crypto with same keys

Now we have 1 key and the other is given to the class

Public key is like the box

APIS usually use this system

Symmetric is still used and everywhere

But the idea is different

User generates pair of keys, public and private

Public goes to everyone and private he only has it

The thing is that its slow

- Remember this
- Solution use the asymmetric to give you the symmetric key and then communicate with the symmetric key
    - Symmetric is faster than asymmetric decryption

Only 1 person has matched private key for that public key

M stands for the message

E stands for the public key

N is number (modular calc)

C is cypher text

D is the private key

$$C = M^e \bmod N$$
$$M = C^d \bmod N$$

How can I send someone messages back

Each side has own set of keys

1 person communicates with the class

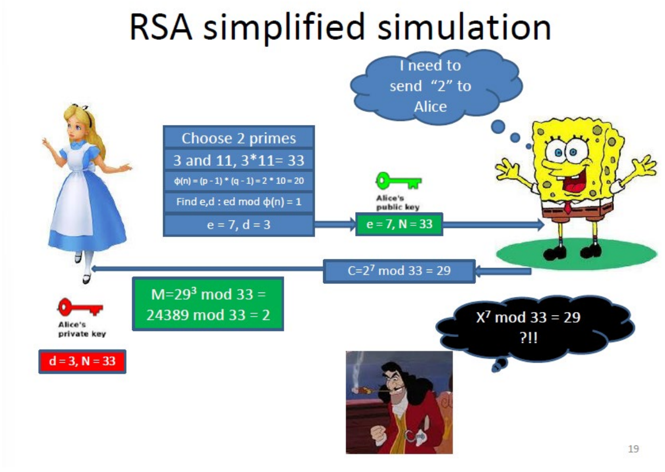
- Just need 2
- For me to send to them need their public keys

For them to send to you – 1 public key which matches with your private key

Send message to someone else requires their public keys

Prime # process is very easy 1 way not the other way

But still have to do it anyways – harder to calculate (compared to symmetric)


RSA simplified simulation
I need to send "2" to Alice
Choose 2 primes
3 and 11, 3*11= 33
ϕ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
Find e,d : ed mod ϕ(n) = 1
e = 7, d = 3
Alice's public key
e = 7, N = 33
C=2^7 mod 33 = 29
M=29^3 mod 33 = 24389 mod 33 = 2
Alice's private key
d = 3, N = 33
X^7 mod 33 = 29 ?!!
19

Bob send to alice uses alice public key

Alice choses two prime #s usually large #s

Eulers function – group theory stuff

She gets a group # then she picks 2 numbers that euler's fn is equal to 1

E = 7 public key

D = 3 private key

Alice sends 2 things , sends the public key and #

Bob uses equation we showed, takes the messages and takes it to power of 7 (public key)

Mods the # and sends to alice and she uses the private key to decrypt message and gets back #2

Don't need to know Euler's stuff for this class

Bob sends 29 not 2… encrypted message and not message itself

Eve can listen (on net and can intercept the message that bob sends to alice and gets 29)

- Know alice uses that public key, can put in equation and can deduce message itself but its not easy
- Brute force takes more than his lifetime for big messages

Krechov relates to this

- Security through obscurity

- Everything is known
- Algo, public keys and even encrpypted #'s and cipher itself but its hard to deduce the message from cipher text even though we know the algorithm

Caser Cipher

- Constant shift = caser cipher
- More complex = Scramble alphabet

Symettric cryptography

- Each person has same private key (shared) in order to communicate secretly
- Requires lots of keys
- Key management
- Non repudiation

Solution is asymmetric cryptography

- Public, private shared key
- Public goes to everyone and private only I have it
- Only 1 person can read messages
- Disadvantage
  - Slow
  - Solution: use symmetric communication always but asymmetric coms only once
  - Send a secret key via asymmetric – but it doesn't relate to the asymmetric pair
  - Then communicate easily via symmetric

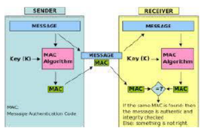
Till now: We discussed C (confidentiality) and got the solution to that

Next thing is integrity

- Message authentication code MAC – not computer mac address

Message Authentication Code (MAC)

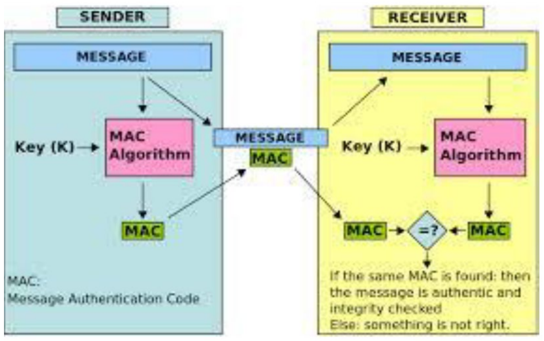
- Why do we need it?
- A number added to a message
- Uses a shared secret key
- Authentication and Integrity (no confidentiality)


SENDER
MESSAGE
Key (K)
MAC Algorithm
MAC
MAC: Message Authentication Code
MESSAGE
MAC
RECEIVER
MESSAGE
Key (K)
MAC Algorithm
MAC
=?
MAC
If the same MAC is found: then the message is authentic and integrity checked
Else: something is not right

To get integrity

- Idea , add # to message and that number is very unique

- In MAC – will also be symmetric
- I have the key and the person to communicate with also has the same secret key
- Bob (sender) takes message and the key, use mac algo and generate #
- Attach # to msg itself
    - Which means its not confidentiality but just integrity


SENDER
MESSAGE
Key (K)
MAC Algorithm
MAC
MAC:
Message Authentication Code
MESSAGE
MAC
RECEIVER
MESSAGE
Key (K)
MAC Algorithm
MAC
=?
MAC
If the same MAC is found: then the message is authentic and integrity checked
Else: something is not right.

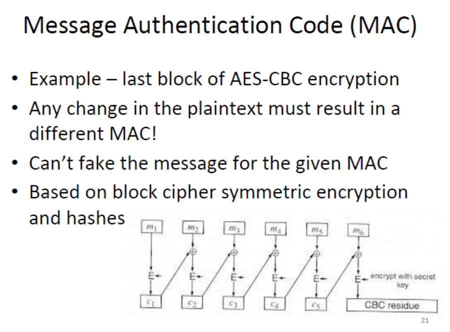
Message and unique mac #

Alice takes that message and she also has the secret key (same) and same MAC algo

- She will go over the same process that bob (sneder) did and generate her MAC and it will be the same
- Note that If someone changed the message (check of 50 usd vs 5000 usd)
    - She should expect to get a different MAC
        - It will be influenced by that and will be different and that's how we say its not reliable
- Non repudiation relates to asymmetric cryptography not this
- Not the same (it had 2 identicalkeys that are the same)
    - Can say someone else sent it …
    - Me at bank, I have the key as well as the bank
    - Can say someone at the bank changed something

Symmetric system will give us integrity – will know the correct message

AES – CDC and many other algorithms do it



Message Authentication Code (MAC)

- Example – last block of AES-CBC encryption
- Any change in the plaintext must result in a different MAC!
- Can't fake the message for the given MAC
- Based on block cipher symmetric encryption and hashes

m_1 m_2 m_3 m_4 m_5 m_6
E
encrypt with secret key
c_1 c_2 c_3 c_4 c_5
CBC residue
21

General idea

Take message and cut into pieces (smaller) until size of key

Take each one of the pieces, encrypt with key, get cipher and solve with next part of msg

Encrypt it again and get another cipher see the chain and goes till end of message

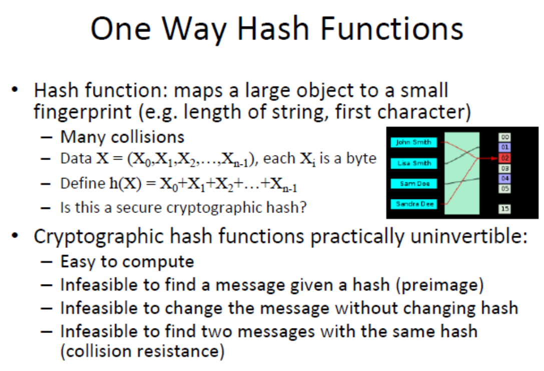
And end message gets the result, mac # and attach this CBC residue and attach to message and send it to receiver

Not necessary in http

Sha 256 will speak about it soon

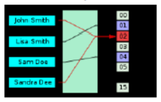
Based on chain of blocks

Another method is to use one way hash functions such as SHA256



One Way Hash Functions

- Hash function: maps a large object to a small fingerprint (e.g. length of string, first character)
  - Many collisions
  - Data $X = (X_0, X_1, X_2, \dots, X_{n-1})$, each X_i is a byte
  - Define $h(X) = X_0 + X_1 + X_2 + \dots + X_{n-1}$
  - Is this a secure cryptographic hash?
- Cryptographic hash functions practically uninvertible:
  - Easy to compute
  - Infeasible to find a message given a hash (preimage)
  - Infeasible to change the message without changing hash
  - Infeasible to find two messages with the same hash (collision resistance)



Similar to MAC but not entirely the same

Hash gives just integrity but MAC also gives authentication (secret key too)

Hash is small figure which represents larger object and can relate to the text itself such as first character or length of string … but that's a problem

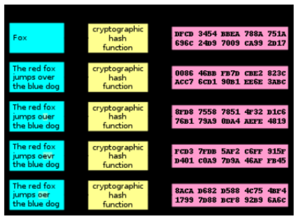
Lots of things with same link

- mAny collisions possible
- take the sum of everything
- hash – we want to be uninvertible
  - given hash number not deduce message
  - easy to compute – make one
  - change message we want has to change too
    - relate to shannon's principles

also should be hard to find 2 messages with same hash – avoid collisions

One Way Hash Functions

- Provide integrity (no key!)
- Hashed MAC (HMAC) – uses a key to provide also authentication
- Widely used: software packages, firmware updates, digital signatures…
- SHA (1 - deprecated, 256), MD5 – also deprecated, see here why


Fox
cryptographic hash function
DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17
The red fox jumps over the blue dog
cryptographic hash function
0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC
The red fox jumps ouer the blue dog
cryptographic hash function
8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819
The red fox jumps oevr the blue dog
cryptographic hash function
FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45
The red fox jumps oer the blue dog
cryptographic hash function
8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

1 word fox goes through hash function and we get this #

Something like 20 bytes

And longer message has the same hash size, hard to relate hash to message itself

If there is a typo we can see an example of what happens, hash is totally different result

- remember the distribution law we had

hash gives integrity , uses hash to know that the program is authentic – otherwise hash would be different

Microsoft also publishes program with the hash it should be

Take file and make hash from the file

Combine hash and mac , hash the mac gives stronger authentication too, widely used

- software packs
- updates
- digital signatures

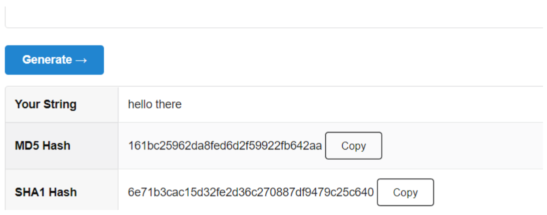
SHA as well – 256 is widely used

MD5 too

Example:

MD5 hash generator – online

https://www.md5hashgenerator.com/



Your String	hello there
MD5 Hash	161bc25962da8fed6d2f59922fb642aa
SHA1 Hash	6e71b3cac15d32fe2d36c270887df9479c25c640

https://github.com/trailofbits/publications/blob/bd3f318151829c7763facc6239cbbf0c0ac76ddf/presentations/Analyzing%20the%20MD5%20Collision%20in%20Flame/flame-md5.pdf

https://github.com/trailofbits/publications/blob/bd3f318151829c7763facc6239cbbf0c0ac76ddf/README.md

can also take a file and hash it

can be different amount of chars in hash

pigpen cipher

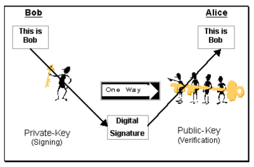
Digital signatures

Using the private key for encryption uniquely authenticates the sender

Everyone can verify that Alice sent this message- signature

Can't be reused for another message

Can prove it was Alice who sent the message (unlike in shared secret schemes)



Replates to asymmetric keys

Bob can sign his messages with own private key

- we know it came form bob, only one with this private key
    - how do we know that?
        - We have he public key that relates to his private key
        - He can sign everything with private key and we open it with his matched public key and know its from bob

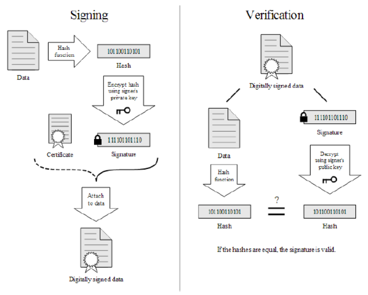
Digital signature – helps with non repudiation

- Prove message was from bob
- They don't share the same keys (not like symmetric case)

Digital signing

Digital signing

- Authentication + integrity
- RSA
- DSA
- Used in: secure mail, official documents, software images, certificates, ...



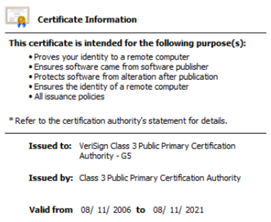
Get data ,hash it then encrypt with private key to get signature, add certificate attached to data , send to receiver to get digitally signed data, data, signature, and public key to encrpy, hash data and compare the two to know its authentic and from right person and covers authentic, integrity, etc.

Uses RSA – private and public key

Pgp, communication with mails

Certificates

- Where all public keys are kept?
- What guarantees that this is indeed Alice's public key?
- Certificate:
  - Credentials (name, organization, email)
  - Public key
  - Signed by someone you trust
    - Chain Of Trust
- X.509 is the standard format



Certificate Information

This certificate is intended for the following purpose(s):

- Proves your identity to a remote computer
- Ensures software came from software publisher
- Protects software from alteration after publication
- Ensures the identity of a remote computer
- All issuance policies

* Refer to the certification authority's statement for details.

Issued to: VeriSign Class 3 Public Primary Certification Authority - G5

Issued by: Class 3 Public Primary Certification Authority

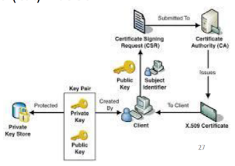
Valid from 08/ 11/ 2006 **to** 08/ 11/ 2021

Certificates

- Public keys kept
  - Organize process
  - Practical

Public Key Infrastructure

- In real life we need lots of certificates, who will sign all of them?
- A small amount of trusted Certificates Authorities (CA) – root CAs
- Root CA's public keys are preinstalled



Alice generates a public and private key

She sends public key with request to the CA (certificate authorities)

- DB of people's public keys
- Organization

Trust the CA – that's the chain of trust

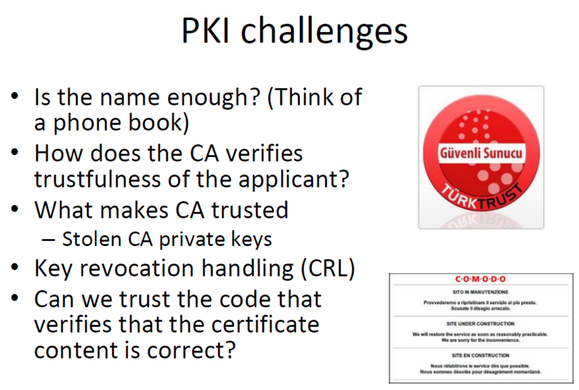
They give authorities to other smaller groups who are responsible for cert but eventually they will be on the top of chain of trust

They provide the x509 cert format to alice – this format includes the official cert she gets and includes alice's public key, published to rest of world

Includes digital signature of CA

- Uses same tequnique as last time bob = CA and their public key to confirm they are the CA

Problem is public key infrastructure – PKI


Güvenli Sunucu
TÜRKTRUST

PKI challenges

- Is the name enough? (Think of a phone book)
- How does the CA verifies trustfulness of the applicant?
- What makes CA trusted
  - Stolen CA private keys
- Key revocation handling (CRL)
- Can we trust the code that verifies that the certificate content is correct?

C·O·M·O·D·O

SITO IN MANUTENZIONE

Provvederemo a ripristinare il servizio al più presto.
Scusate il disagio arrecato.

SITE UNDER CONSTRUCTION

We will restore the service as soon as reasonably practicable.
We are sorry for the inconvenience.

SITE EN CONSTRUCTION

Nous rétablirons le service dès que possible.
Nous sommes désolés pour désagrément momentané.

-
- Two examples for breaking into the CA databases and use stolen private keys (stole CA's private keys)
  - Someone broke into their bank

So they needed to revoke it

Not unbreakable

Solution is revoking keys, cancel the old private keys and generate new ones

Can always doubt if the process is safe and proper

- There is life and we need to proceed

https://www.boxentriq.com/code-breaking/pigpen-cipher

next time continue with cryptography

see shannon's solution and how it looks like