INFORMATION SECURITY

Cryptographic Keys

Symmetric key for AES-128 cipher (16 bytes):

```
iA3_ jNHOIo DZSz [ASCII] =
= 69:41:33:5f:20:6a:4e:48:4f:49:6f:20:44:5a:53:7a [hexadecimal]
```

RSA 1024b public key of www.fe.up.pt (FEUP, 2009):

```
e: 65537 [decimal] = 10001 [hexadecimal]
n: [hexadecimal]
00:be:50:2a:81:7c:75:5c:c0:38:2c:f4:a8:0d:3d:e2:95:53:30:be:af:94:c5
:9f:fe:1d:06:62:67:13:8d:71:be:d8:66:91:79:74:fb:7c:3f:6a:a9:74:c4:9
3:87:7a:bc:47:df:07:dc:f7:65:4c:56:81:43:b3:e8:67:ad:6c:2d:37:b3:34:
14:e7:47:8b:ed:1a:b3:cb:04:93:4f:12:22:8e:d6:47:80:3c:a6:da:d6:f8:e2
:6b:ad:de:73:3b:ee:33:3b:31:b5:ef:b8:ed:52:f4:52:60:59:5e:c2:ed:b7:f
b:8d:4a:8a:52:ed:9f:25:d2:ee:00:ed:f9:15:ef:41
```

Passwords for accessing SiFEUP:

```
IamJohnDoe007 [average]
Iam00John7Doe [better]
```

Definition

- cryptographic key – piece of data needed for cryptographic operations
  - usually: number or string hard to memorize
  - many times: fit to a mathematical procedure (algorithm)
    - so, user cannot "choose" it: a “cryptographic key generator” is needed
  - is secret: known just by 1 to very few people
    - (depending on the algorithm or on the application)

Clarification

- Key is not, **usually**, password!
- Because, **usually**,
  - *password* is memorable, *key* is not
  - *password* is system-independent, *key* depends on cryptographic system
  - *password* is personal, *key* might be or not
  - *password* is used at the beginning of a computer session work, *key* is not
  - *password* is not directly used in cryptographic operations, *key* is
- **But** password can
  - act as a key (e.g in symmetric cryptography)
  - be used to generate a key (e.g. Password-Based Key Derivation Functions)
    - then, *password* strength limits strength of *key*!
      - (even as *passphrase*)

Examples of keys and passwords

- Public key, RSA-1024b of FEUP, `www.fe.up.pt` (2009...):
  - *e*: `65537` (decimal) = `10001` (`hexadecimal`)
  - *n* (byte by byte, in hexadecimal):
    ```
    00:be:50:2a:81:7c:75:5c:c0:38:2c:f4:a8:0d:3d:e2:95:53:30:be:af:94:c5:9f
    :fe:1d:06:62:67:13:8d:71:be:d8:66:91:79:74:fb:7c:3f:6a:a9:74:c4:93:87:7
    a:bc:47:df:07:dc:f7:65:4c:56:81:43:b3:e8:67:ad:6c:2d:37:b3:34:14:e7:47:
    8b:ed:1a:b3:cb:04:93:4f:12:22:8e:d6:47:80:3c:a6:da:d6:f8:e2:6b:ad:de:73
    :3b:ee:33:3b:31:b5:ef:b8:ed:52:f4:52:60:59:5e:c2:ed:b7:fb:8d:4a:8a:52:e
    d:9f:25:d2:ee:00:ed:f9:15:ef:41
    ```
  - or, in PEM[1] format:
    ```
    -----BEGIN PUBLIC KEY-----
    MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC+UCqBfHVcwDgs9KgNPeKVUzC+
    r5TFn/4dBmJnE41xvthmkXl0+3w/aql0xJOHerxH3wfc92VMVoFDs+hnrWwtN7M0
    FOdHi+0as8sEk08SIo7WR4A8ptrW+OJrrd5zO+4zOzG177jtUvRSYFlewu23+41K
    ilLtnyXS7gDt+RXvQQIDAQAB
    -----END PUBLIC KEY-----
    ```
- Symmetric key for an AES-128 cipher:
  - `iA3_ jNHOIo DZSz` [*16 chars*]
    - `69:41:33:5f:20:6a:4e:48:4f:49:6f:20:44:5a:53:7a [hexadecimal]`

1 Privacy-Enhanced Mail (https://en.wikipedia.org/wiki/Privacy-Enhanced_Mail)

...Examples of keys and passwords (cont.)

- Passphrase for accessing some sites:
  - `I am JohnDoe 007` [*16 chars*][2]
- Password for entering an authentication-protected sector of SiFEUP:
  - `IamJohnDoe007`
- One Time Password, OTP (S/Key), RFC 2289:
  - (*hash* = MD5; *n* = 99; *init passwd*: `A_Valid_Pass_Phrase`; *seed*: `AValidSeed`)
  - password 64b: `0x85c43ee03857765b`
    - alternative form: `FOWL KID MASH DEAD DUAL OAF`

2 Could also be used as a AES-128b key!

Key types **of cryptographic keys**

Designation	***"Owner" entity***	***Main application***	***Cryptographic type***	***Longevity***	***Efficiency***
personal	human	authentication	public-key	extended	low
session	communication channel	confidentiality	shared-key	short[3]	high[4]

3 to be use-resistant (prevent brute-force search and repetition attacks)
4 to accommodate heavy traffic

Key Management

- generation
  - problem solved: just take care with choosing of numbers (randomness...)
- storage
  - many "solutions", but still a problem swept under the rug!
- distribution
  - popular "research" topic with plenty of solutions

Generation

Problem

- randomness is essential
  - generation of random numbers: (special) physical source...
  - e.g.: *one-time pad…*
- other issues
  - choosing of numbers depends on algorithm
  - some known situations have to be avoided

Solution

- practical: cryptographically secure pseudo-random number generators
  - e.g. Ron Rivest's RC4 (with much care)
  - AES, SHA, etc. can also be used for generation!

Storage

Problem

- human memory has limitations: of space and of operation (faults)
  - so, keys have to be kept in physical (secure) places
    - normal solution:
      - cipher the keys with a (symmetric) key derived from a… password!!
- acute in asymmetrical systems:
  - having a (private) key is **being** an entity!

Recovery

- can be very important in certain situations (personal and political)
- can be made by special systems (*key escrow systems*)

Distribution

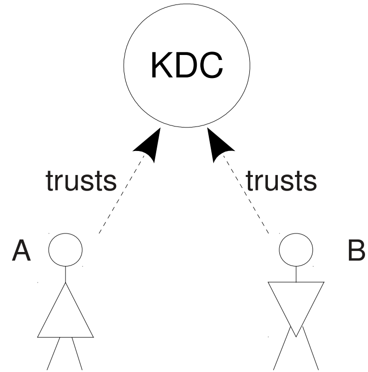
Problem

- physically separated entities must exchange/agree on cryptographic keys
  - acute in symmetrical systems
    - repeated problem for every key substitution (that should be frequent!)
  - conundrum: agreement needs authenticated entities
    - but authentication needs cryptographic keys!...

Solutions

- meeting in person (at least at first time)
- use <u>previously secured channels</u> (or alternative ones - *out-of-band*)
- use <u>insecure channels</u> with special protocols (e.g. Diffie-Hellman's)
- → in practice, a combination of these “solutions” might be used
- use a *trusted* Key Distribution Center

Key Distribution Center, KDC



- entity trusted by all other (user) entities
  - (at first) each (user) entity do not trust other (user) entities
  - **definition of X trusts T:**
    - X believes T operates in an honest way!
    - X have exchanged with T cryptographic info[5]
- symmetric systems:
  - generate, store and distribute secret keys (e.g. Kerberos)
  - mostly used for session keys (and even so...)
- asymmetric systems:
  - store and distribute public user keys (Public Key Server)
  - should **not** handle private keys!

5 $K_{X,T}$, or K_X^+ & K_T^+

Public Key Servers, PKS

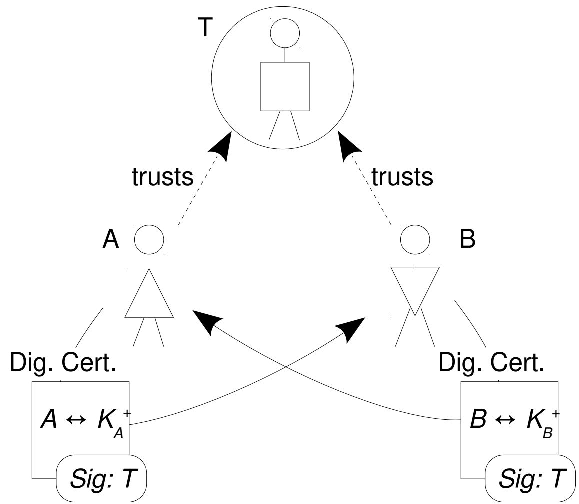
- Key Distribution Center for public key distribution!
- key generation not implied[6]
- key storage may be not implied[7]
- “entity - public key” mapping should be assured. So:
  - either the PKS scrutinizes and authenticates the keys it keeps and distributes
    - and then the PKS should be authenticated by the clients that rely on it
  - or the key information the PKS keeps is self-authenticated
    - and then the PKS does not control the mapping
    - clients need not authenticate the PKS, but should validate the keys themselves
      - e.g. PGP[8] public key server: `keys.openpgp.org`

6 Of course! Why not?!...
7 see below the role of a Certification Authority (CA)
8 Pretty Good Privacy (to be seen later)

Digital Certificates

Basics

- document that maps an entity to a cryptographic *public* key
  - the mapping is guaranteed by a “trusted” entity T[9] that “signs” it[10]
- with a reliable public key one can:
  - authenticate its owner, being it
    - a person, company, *website*…
  - and so validate owner's
    - documents, software…
  - send confidential information to owner



9 usually, but not necessarily, T is connoted with a Certification Authority (CA) (see below)
10 so, assuring the accuracy of the certificate's content (to be seen later)

Typical content

```
I hereby certify that the public key
    19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A
belongs to
    Robert John Smith
    12345 University Avenue
    Berkeley, CA 94702
    Birthday: July 4, 1958
    Email: bob@superdupernet.com
```

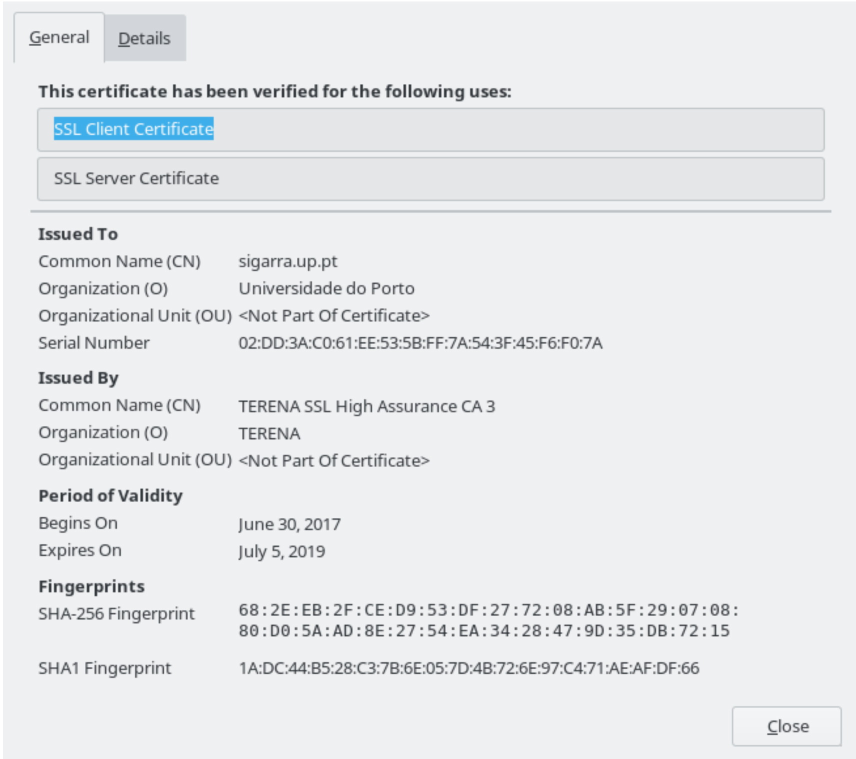
SHA-1 hash of the above certificate signed with the CA's private key

- name of the subject (entity to whom the certificate applies)
- subject's public key
- name of the emitter (e.g. *Certificate Authority*)
- digital signature of emitter
- expiration time of certificate
- serial number
- specific purpose
- etc.

...Digital Certificates...

Physical "face" of digital certificate

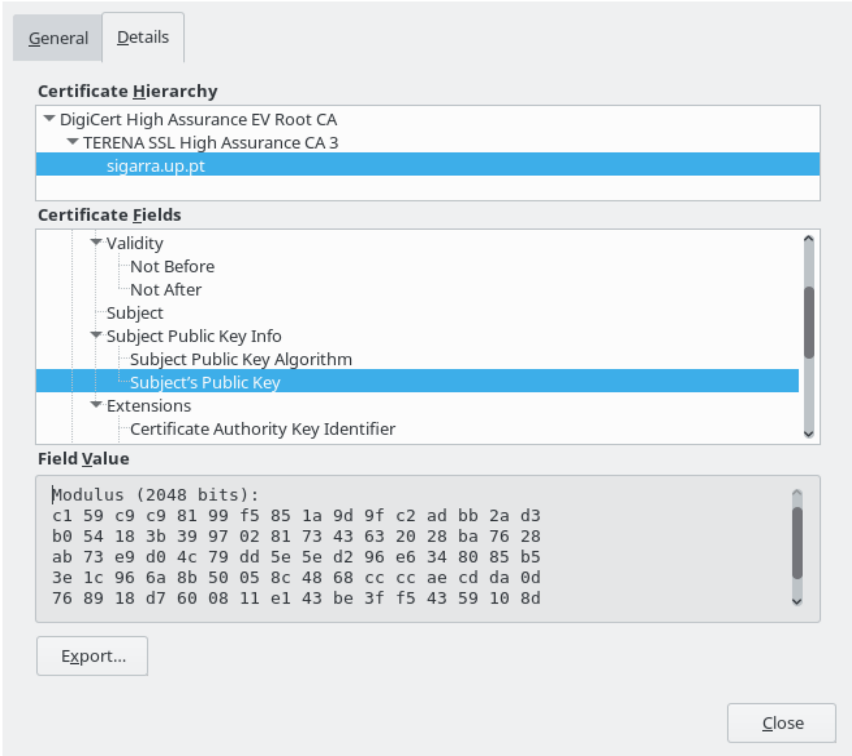
Fig. Old certificate of website sigarra.up.pt.



...Digital Certificates...

Physical “face” of digital certificate (cont.)

Fig. Details of certificate of sigarra.up.pt.



Why digital certificates?

- for getting reliable public keys[11]
  - reliability depends on the trust on the certificate issuer/signer
- but, usually:
  - certificates are *Internet X.509 Public Key Infrastructure*'s
  - emitters are "Certificate Authorities" (CA)
    - private, commercial, worldwide companies, operating in isolation or aggregation (more further down)
      - e.g. `DigiCert`, `Sectigo` (previously, `Comodo`)
    - some exceptions:
      - `CAcert`[12], `Let’s Encrypt`[13]

11 to be used for whatever purpose (authenticate entities, validate documents, cipher communications...)
12 emits all types of certificates (see below)
13 emits only Domain Validation certificates (see below)

...Utilization of digital certificates (cont.)

Revocation of certificates

- needed if one wants to change keys (private-public pair), for whatever reason
  - use of revocation lists, *Certificate Revocation Lists*
  - use of *Online Certificate Status Protocol* (RFC 2560)
  - use of expiration times (eventually, also with revocation lists)

Types of Digital Certificates

- what? types? isn't it just a mapping: entity ↔ public key ?
  - Yes, but… $$$ and, perhaps, increased security...
- Rough classification:

Certificate type	***Entity "type"***	***Checking "type" (by CA)***	***Application type***
Address Validation (AV)	Individual	simple (e.g. email address is ok)	S/MIME email
Individual Validation (IV)	Individual	"more" precise verification (e.g. is employee of company)	S/MIME email SSL/TLS client authentication
Domain Validation (DV)	Organization	simple (e.g. `postmaster@domain` answers)	SSL/TLS server authentication
Organization Validation (OV)	Organization	organization is legal and "owns" domain	SSL/TLS-enabled sites Code signing...
Extended Validation (EV)	Organization	conformance to specific CA/Browser Forum guidelines[14]	SSL/TLS-enabled sites Code signing…

14 "EV SSL Certificate Guidelines", which includes thorough, human verifiable checking of organization.

X.509 v.3 Digital Certificates

- currently, worldwide prevalent (in PKI dominated by Certification Authorities)
- based on the (old) OSI X.500 Directory Service, with some updating
  - e.g. X.500 name:
    - `/C=PT/O=Universidade Porto/OU=Dept. Informatica/CN=J.M. Cruz`
  - e.g. DNS name (X.509 v.3):
    - `jmcruz@fe.up.pt`
- many information fields[15]
  - some mandatory
    - Subject Public Key Algorithm, Validity, ...
  - lots of optionals (extensions)
    - Certificate Key Usage, Subject Alternative Name, ...

15 beyond subject's name and subject's public key!

Public Key Infrastructure, PKI

- formally (& commercially):
  - «*set of roles, policies, and procedures needed to create, manage, distribute, use, store & revoke* ***digital certificates*** *and manage public-key encryption*»[16]
  - «[aims] *to facilitate the secure electronic transfer of information for a range of network activities such as e-commerce, Internet banking and confidential email*»
- really, simply:
  - general scheme for binding public keys with entities
    - (certificate authorities?... digital certificates?…)
  - applications vary and security properties should be assured by communicating parties

Examples of PKIs

- *Internet X.509 Public Key Infrastructure* (IETF RFC 5280...)
- *OpenPGP model* (IETF RFC 4880...)

16 en.wikipedia.org/wiki/Public_key_infrastructure

...PKI (cont.)

Digital certificate's chain of certification – the centralized hierarchical model

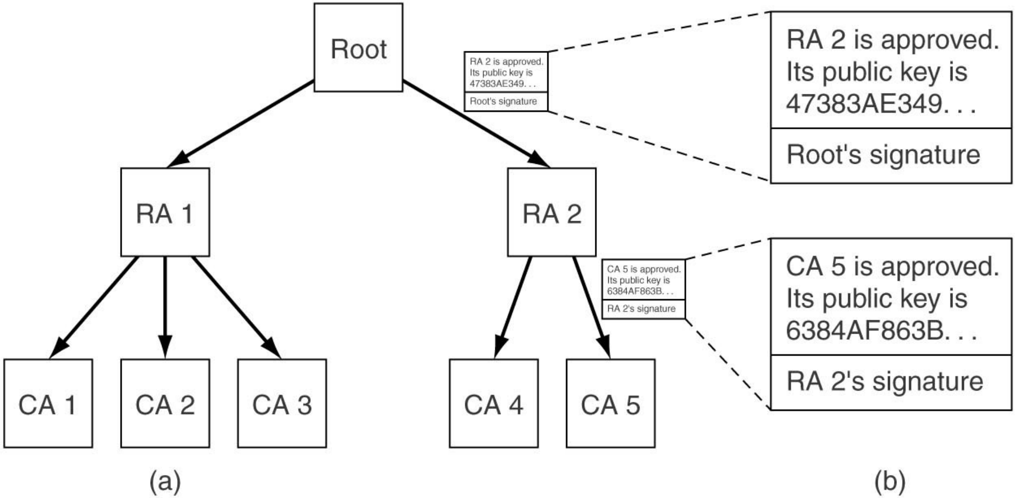


Fig. Certification of certificates in a centralized PKI. a) hierarchy; b) chain of certificates.

Digital certificate's chain of certification – the oligarchic model

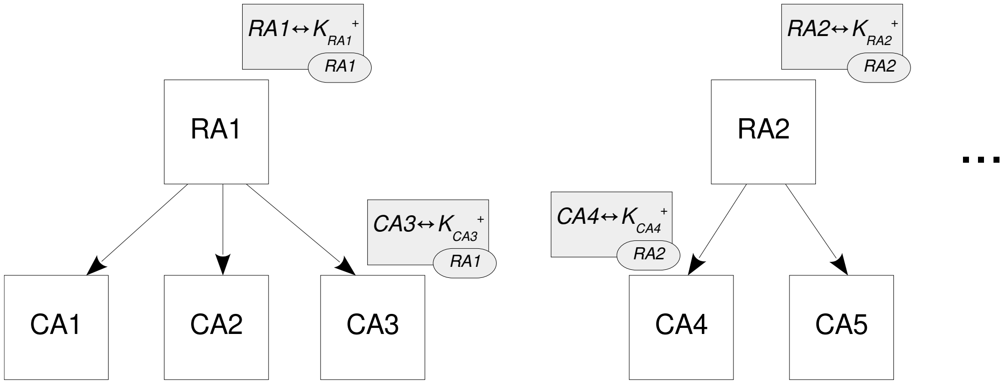


Fig. Certification of certificates in a oligarchic PKI. (Notice the self-signed certificates.)

Exercises:

- For each of the models, what minimum secure information must an user possess in order to use the corresponding Public Key Infrastructure?
- What can a user possibly gain with the oligarchic model?

Problems with Certificate Authorities

- they should be honest and reliable CAs, but:
  - they keep not acting as such[17]
- there are many of them[18], so
  - there are many points of failure!
- they should issue certificates in a controlled manner, but:
  - probably, not for <u>any</u> domain!
  - certainly, not without the knowledge of domain owner!

17 see for instance, sslmate.com/blog/post/history_of_ca_sanctions
18 more than 100 are installed in some renowned web browsers

Solutions (?)

- do not use Certificate Authorities services!
  - get public keys first hand (e.g. FEUP!…) or from trusted parties (e.g. Web of Trust)
  - Prob: SSL/TLS **needs** certificates[19];
    - so, create "local" CA and add them to software's *Certificate data store*!
- restrain CA's issuing to certain domains
  - use new DNS record type "Certification Authority Authorization" in own domain
- Certificate/Public Key Pinning
  - associate a host/person with its/his certificates or public keys collected from first time contacts[20] or from trustworthy sources
    - e.g. Secure Shell (SSH)'s solution
- Perspectives (Carnegie-Mellon), Transparency (Google)...

19 true in practice, but not in principle (RFC 7250)
20 technique sometimes called Trust on First Use (TOFU)