Diffie-Hellman Key...
€2.00

Quick view
Man-In-The-Middle Attack
€4.00

The problems to be solved during the final exam: contact participation.

El Gamal E-Signature
€3.00

ElGamal Encryption
€3.00

Schnorr E-Signature
€3.00

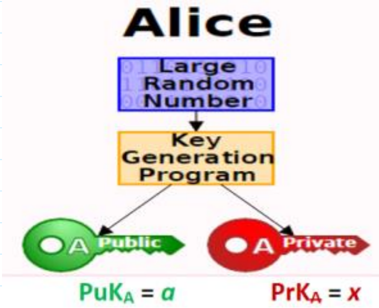
Schnorr Identification
€3.00

Quick view
MINI-HTTPS
€5.00

Public Parameters **PP** = (**p**, **g**):
\>> p=strongprime(28)
p = 268435019
g=2;
p - strong prime; **g** - generator.

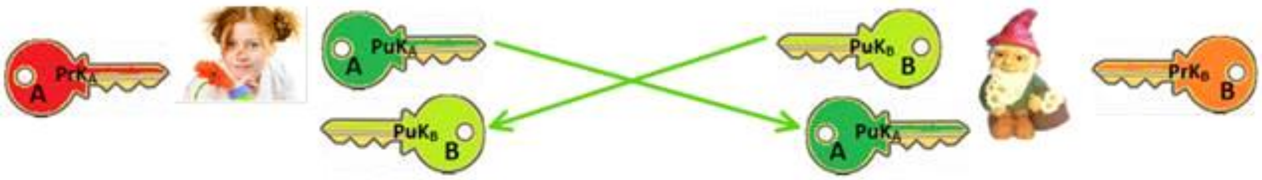
\>> p=int64(268435019)
p = 268435019
g=2;

Private key **PrK** and public key **PuK** generation for **Alice** and **Bob**.
PrK = *x* <-- randi ==> PuK = *α* = g^x mod p



\>> x=int64(randi(p-1))
x = 13426057
\>> a=mod_exp(g,x,p)
a = 2045067

\>> y=int64(randi(p-1))
y = 13426057
\>> b=mod_exp(g,y,p)
b = 2045067

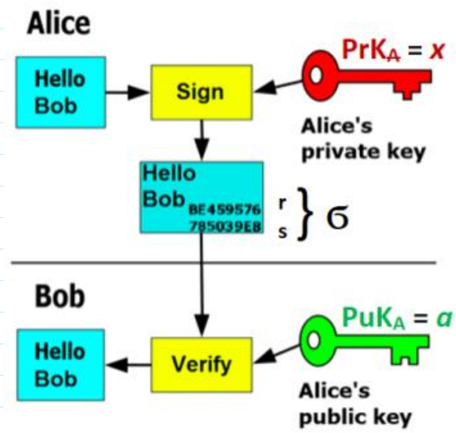


Public Key Infrastructure - PKI **Viešojo Rakto Infrastruktūra - VRI**

$A : (PrK_A, PuK_A)$ $B : (PrK_B, PuK_B)$

PuK_A

$PuK_A = \alpha = g^x \bmod p$



M message to be signed : Loan Contract.

$|M| \sim 10$ KB

Hash and sign paradigm:

$h = H(M)$; $|h| \sim 256$ bits ⟵ SHA256

$Sign(PrK_A, h) = \sigma = (r, s)$

M, σ, PuK_A ⟶ 1) $h' = H(M')$

2) $Ver(PuK_A, \sigma, h') = \begin{cases} True \\ False \end{cases}$

1) If Ver = True, then signature σ is formed using A's private key PrK_A which corresponds (is mathematically related) with A's public key PuK_A.

[ECDSA : $PrK_A = x$, $|x| \sim 256$ bits
$x \sim 2^{256}$ and $PuK_A = x \cdot G = A$ <--> $PuK_A = g^x \bmod p = \alpha$] nebūtina

$Zo : (PrK_Z, PuK_Z)$ $\xrightarrow{PuK_Z}$

Dear Bob I am A and
I am sending you my
public key

Public Key Infrastructure - PKI

$CA = (PrK_{CA}, PuK_{CA})$ It is as notarius office

Certification Authority - CA ⟹ Registration Authorities - RA - subsidiaries of CA

Verisign Trusted Third Party - TTP ⟹ all users recognizes CA

RA1 RA2 RA3 - - - -

PuK_{CA}
recognized by the users

Trusted Third Party - TTP ⇒ all users recognizes CA

RA1 RA2 RA3 ....

https://verysign.com

PuK_{CA} recognized by the users

browsers: Chrome, Opera...

A: PuK_A → RA —confirms A identity PuK_A $Data_A$→ CA:

(PrK_{CA}, PuK_{CA})

$M_A = PuK_A \| Data_A$

$h_A = H(PuK_A \| Data_A)$

$\sigma_A = Sign(PrK_{CA}, h_A)$

$Cert_A = \sigma_A \| PuK_A \| Data_A$

A: PuK_{CA}. ← $Cert_A$, PuK_{CA}

$h_A = H(PuK_A \| Data_A)$

$Ver(PuK_{CA}, \sigma_A, h_A) = \begin{cases} True \\ False \end{cases}$

$Sign(PrK_A, h) = \sigma$; —M, σ, PuK_A, $Cert_A$→ B: PuK_{CA}, PuK_A

1) $Cert_A \longrightarrow \sigma_A \| PuK_A \| Data_A$; M, σ

2) $h''_A = H(PuK_A \| Data_A)$

3) $Ver(PuK_{CA}, \sigma_A, h''_A) = \begin{cases} True \\ False \end{cases}$

4) $h' = H(M)$

5) $Ver(PuK_A, \sigma, h') = \begin{cases} True \\ False \end{cases}$

- **X509 v3 Standard**
-
- **SerialNumber**
- **Issuer** } Verisign
- **notBefore** } 2021.11.10; 18:10:07
- **notAfter** } 2022.11.10; 18:10:07
- **Subject** } A
- **Algorithm** } ECDSA
- **SubjectPublicKey** } PuK_A
- **extensions**

Zo ← $Cert_Z$ ← CA

2021.11.12; 19:10:11

2022.11.12; 19:10:11

↓

2022.11.12; 19:10:12

Zo: (PrK_Z, PuK_Z); $Cert_Z$.

L - loan contract → $h = H(L)$

$Sign(PrK_z, h) = \sigma_z$

L, σ_z, PuK_z → B: 1) +

$cert_z$ 2) +

3) +

Money ← 4) +

transfer 5) +

← to pay % for the loan

sorry my loan contract → is invalid since at the time you've singned it my certificate validity term expired

CA services: CRL – Certificates Revocation List

OCSP – On-line Certificates status Protocol

6) Verify if $Cert_z$ is not in certification revocation list (CRL).

7) If validity of $Cert_z$ is not expired.

- **Certificates Revocation List - CRL:**

Is a list of digital certificates that have been revoked by the issuing certificate authority (CA) before their scheduled expiration date and should no longer be trusted.

There are two different states of revocation defined in RFC 5280:

Revoked

A certificate is irreversibly revoked if, for example, it is discovered that the certificate authority (CA) had improperly issued a certificate, or if a private-key is thought to have been compromised. Certificates may also be revoked for failure of the identified entity to adhere to policy requirements, such as publication of false documents, misrepresentation of software behaviour, or violation of any other policy specified by the CA operator or its customer. The most common reason for revocation is the user no longer being in sole possession of the private key (e.g., the token containing the private key has been lost or stolen).

Hold

This reversible status can be used to note the temporary invalidity of the certificate (e.g., if the user is unsure if the private key has been lost). If, in this example, the private key was found and nobody had access to it, the status could be reinstated, and the certificate is valid again, thus removing the certificate from future CRLs.

A CRL is generated and published periodically, often at a defined interval. A CRL can also be published immediately after a certificate has been revoked. A CRL is issued by a CRL issuer, which is typically the CA which also issued the corresponding certificates, but could alternatively be some other trusted authority. All CRLs have a lifetime during which they are valid; this timeframe is often 24 hours or less. During a CRL's validity period, it may be consulted by a PKI-enabled application to verify a certificate prior to use.

To prevent spoofing or denial-of-service attacks, CRLs usually carry a digital signature associated with the CA by which they are published. To validate a specific CRL prior to relying on it, the certificate of its corresponding CA is needed.

The certificates for which a CRL should be maintained are often X.509/public key certificates, as this format is commonly used by PKI schemes.

From <https://en.wikipedia.org/wiki/Certificate_revocation_list>

- **On-line Certificates Status Protocol - OCSP:**

Is an Internet protocol used for obtaining the revocation status of an X.509 digital certificate.[1] It is described in RFC 6960 and is on the Internet standards track. It was created as an alternative to certificate revocation lists (CRL), specifically addressing certain problems associated with using CRLs in a public key infrastructure (PKI).[2] Messages communicated via OCSP are encoded in ASN.1 and are usually communicated over HTTP. The "request/response" nature of these messages leads to OCSP servers being termed *OCSP responders*.

Some web browsers use OCSP to validate HTTPS certificates.

- Since an OCSP response contains less data than a typical certificate revocation list (CRL), it puts less burden on network and client resources.[3]
- Since an OCSP response has less data to parse, the client-side libraries that handle it can be less complex than those that handle CRLs.[4]
- OCSP discloses to the responder that a particular network host used a particular certificate at a particular time. OCSP does not mandate encryption, so other parties may intercept this information.[

From <https://en.wikipedia.org/wiki/Online_Certificate_Status_Protocol>

Qualified and **Non-qualified** certificates

Qualified → mathes with e-signature law

Non-qualified → Is valid according to contract between parties

Eureka

EU e-document system

2008 m. – 2009 m.

Gemalto 800 000 €

Sagem 1 200 000 € ...

$\Sigma \sim 2\,400\,000$

Time Stamping Authority – TSA – Trusted Third Party (TTP)

A: L – loan contract → $h = H(L)$

$Sign(PrK_A, h) = \sigma$ —— $L, \sigma, PuK_A, Cert_A$ ——→ **TSA**: $(PrK_{TS}, PuK_{TS}), Cert_{TS}$.

PuK_{CA}, PuK_A $h = H(L)$

1. $Ver(PuK_{CA}, Cert_A) = True$
2. $Ver(PuK_A, \sigma, h) = True$
3. $DT = YYYY.MM.DD{:}hh{:}mm{:}ss{:}\ldots$
4. $h_{TS} = H(h, \sigma, DT, PuK_{TS}, Cert_{TS})$
5. $Sign(PrK_{TS}, h_{TS}) = \sigma_{TS}$

A: PuK_{CA} ←—— $DT, \sigma_{TS}, PuK_{TS}, Cert_{TS}$ ——

1. Verifies DT
2. Verifies validity of $Cert_{TS}$

3. $h'_{TS} = H(h, \sigma, DT, PuK_{TS}, Cert_{TS})$

4. $Ver(PuK_{TS}, \sigma_{TS}, h'_{TS}) = True \Rightarrow$ If: $\left\{ \begin{array}{l} h'_{TS} = h_{TS} \\ PuK_{TS} = g^{x_{TS}} \bmod p \end{array} \right\} \to True$

$L' = L \| DT \| \sigma_{TS}$

$h_L = H(L')$

$\sigma_L = Sign(PrK_A, h_L) = \sigma_L$

A: $\xrightarrow{L', \sigma_L, PuK_A, Cert_A}$ B: (PrK_B, PuK_B); PuK_{CA}

$DT, \sigma_{TS}, PuK_{TS}, Cert_{TS}$

1. $Ver(PuK_{CA}, Cert_{TS}) = True$
2. $Ver(PuK_{CA}, Cert_A) = True$
3. $h'_L = H(L')$; $h''_{TS} = H(h, \sigma, DT, PuK_{TS}, Cert_{TS})$
4. $Ver(PuK_{TS}, \sigma_{TS}, h''_{TS}) = True$
5. $Ver(PuK_A, \sigma_L, h'_L) = True$
6. OCSP: to verify that certificates are in the interval: [notBefore, notAfter] → Yes
7. CRL: do the $Cert_A$ and $Cert_{TS}$ not revoked → No

A: $\xleftarrow{\text{money transfer}}$ B

$M = \{a_{ij}\}, \quad i = 1, 2, \ldots, m; \quad j = 1, 2, \ldots, m$

$m = 3$

$$M = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$N = \{b_{jk}\}$

$M \cdot N = \{c_{ik}\}$

$$c_{11} = \sum_{j=1}^{m} a_{ij} b_{jk} = a_{11} \cdot b_{11} + a_{12} \cdot b_{21} + a_{13} \cdot b_{31}$$

$m = 3$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}$$