Securitatea Sistemelor Informatice

- Curs 11 -
Semnături digitale și PKI

Adela Georgescu

Facultatea de Matematică și Informatică
Universitatea din București
Anul universitar 2022-2023, semestrul I

Scheme de semnătură digitală

- Schemele de semnătură digitală reprezintă echivalentul MAC-urilor în criptografia cu cheie publică, deși există câteva diferențe importante între ele;

Scheme de semnătură digitală

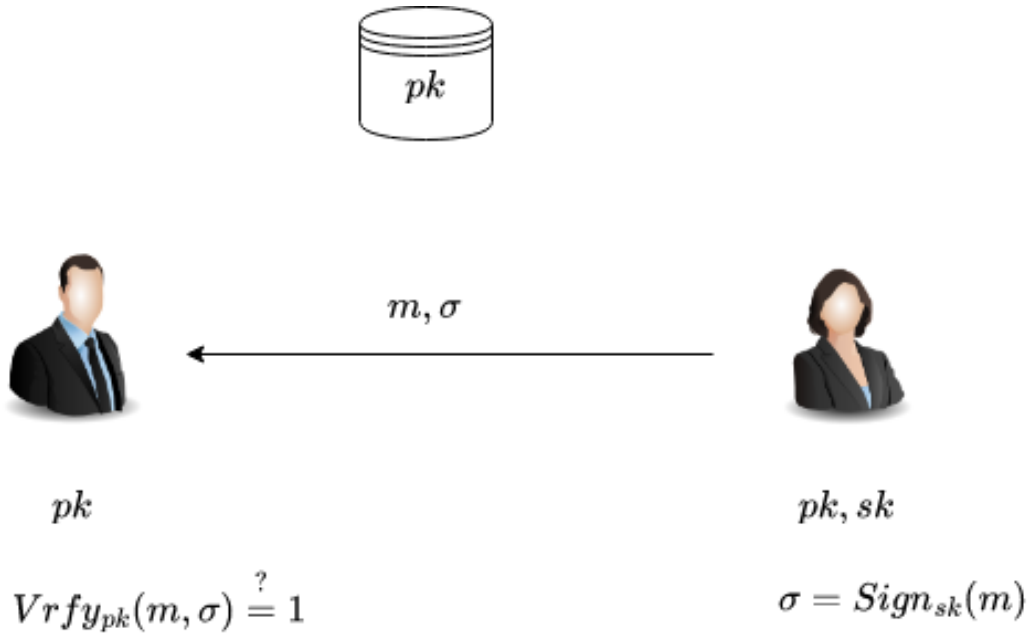
- Schemele de semnătură digitală reprezintă echivalentul MAC-urilor în criptografia cu cheie publică, deși există câteva diferențe importante între ele;

- O schemă de semnătură digitală îi permite unui semnatar S care a stabilit o cheie publică *pk* să semneze un mesaj în așa fel încât oricine care cunoaște cheia *pk* poate verifica originea mesajului (ca fiind S) și integritatea lui;

Scheme de semnătură digitală

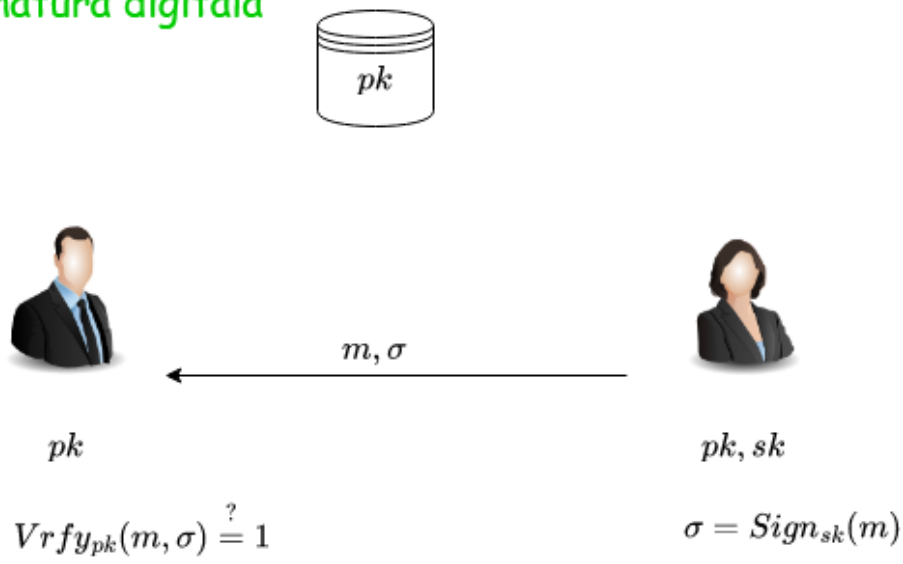
- Schemele de semnătură digitală reprezintă echivalentul MAC-urilor în criptografia cu cheie publică, deși există câteva diferențe importante între ele;

- O schemă de semnătură digitală îi permite unui semnatar S care a stabilit o cheie publică *pk* să semneze un mesaj în așa fel încât oricine care cunoaște cheia *pk* poate verifica originea mesajului (ca fiind S) și integritatea lui;

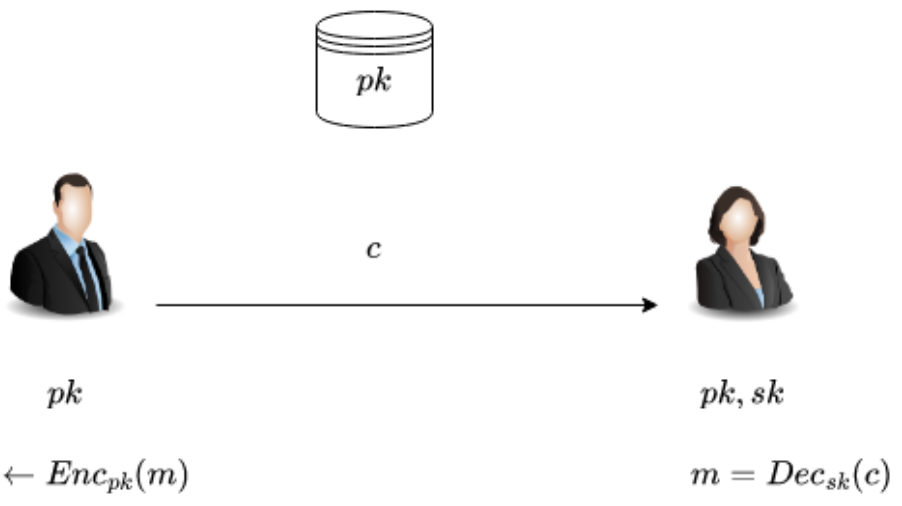


Semnătură digitală vs. criptare

Semnatura digitala



Criptare

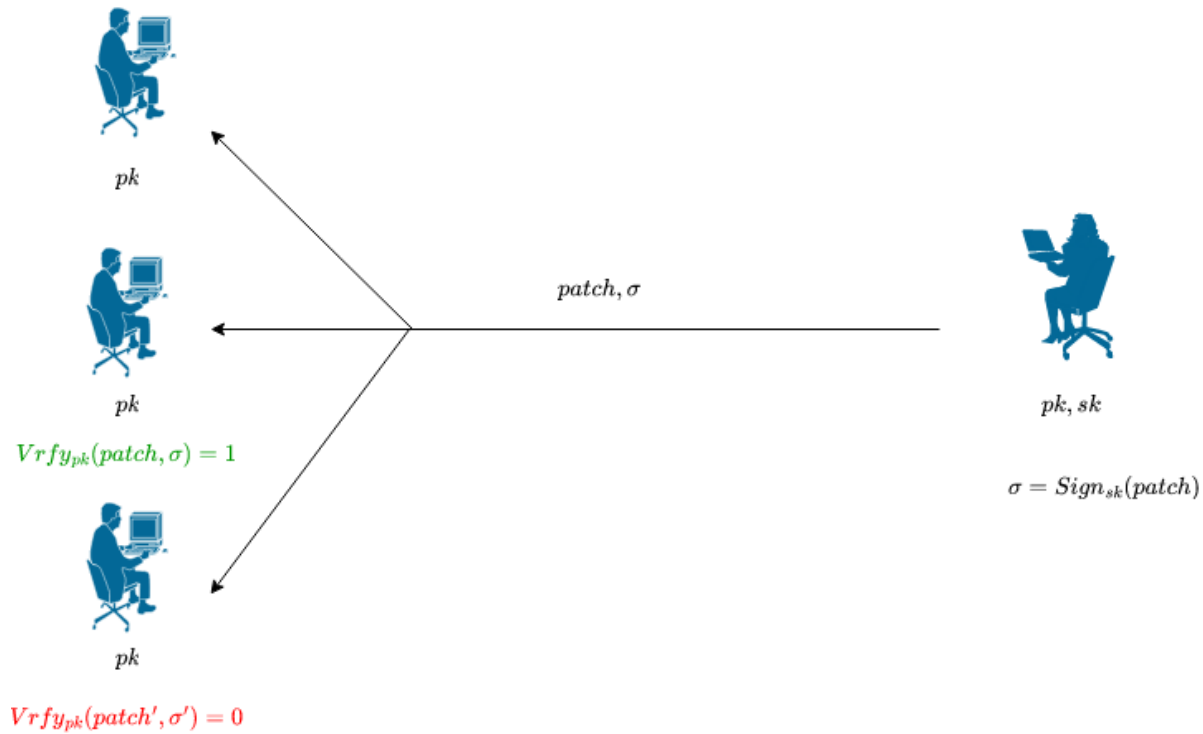


Aplicații ale semnăturilor digitale

- De pildă, o companie de software vrea să transmită patch-uri de software într-o manieră autentificată, așa încât orice client să poată recunoaște dacă un patch e autentic;

Aplicații ale semnăturilor digitale

- De pildă, o companie de software vrea să transmită patch-uri de software într-o manieră autentificată, așa încât orice client să poată recunoaște dacă un patch e autentic;
- În schimb, o persoană malițioasă nu poate păcăli un client să accepte un patch care a nu a fost realizat de compania respectivă.



Aplicații ale semnăturilor digitale

- Pentru aceasta, compania generează o cheie publică *pk* împreună cu o cheie secretă *sk* și distribuie cheia *pk* clienților săi, păstrând cheia secretă;

Aplicații ale semnăturilor digitale

- Pentru aceasta, compania generează o cheie publică *pk* împreună cu o cheie secretă *sk* și distribuie cheia *pk* clienților săi, păstrând cheia secretă;
- Atunci când lansează un patch de software *patch*, compania calculează o semnătură digitală σ pentru *patch* folosind cheia *sk* și trimite fiecărui client perechea $(patch, \sigma)$;

Aplicații ale semnăturilor digitale

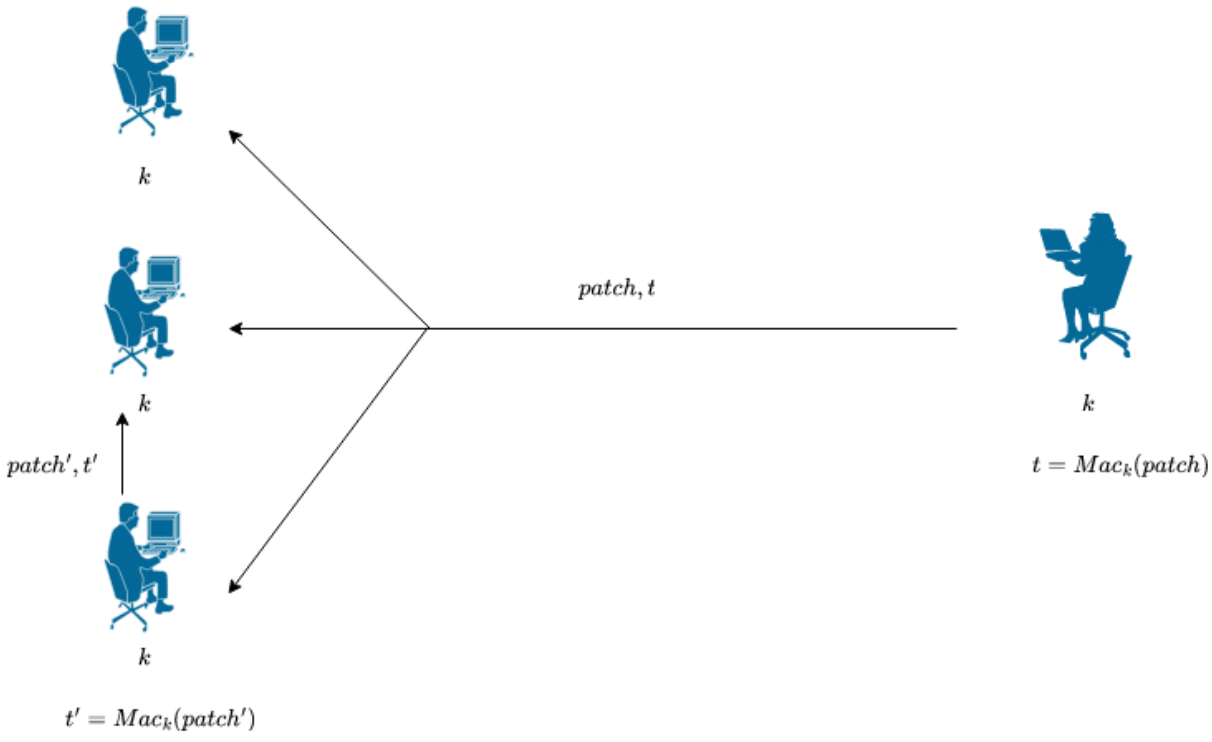
- Pentru aceasta, compania generează o cheie publică *pk* împreună cu o cheie secretă *sk* și distribuie cheia *pk* clienților săi, păstrând cheia secretă;
- Atunci când lansează un patch de software *patch*, compania calculează o semnătură digitală σ pentru *patch* folosind cheia *sk* și trimite fiecărui client perechea $(patch, \sigma)$;
- Fiecare client stabilește autenticitatea lui *patch* verificând dacă σ este o semnătură legitimă pentru *patch* cu privire la cheia publică *pk*;

Aplicații ale semnăturilor digitale

- Pentru aceasta, compania generează o cheie publică *pk* împreună cu o cheie secretă *sk* și distribuie cheia *pk* clienților săi, păstrând cheia secretă;
- Atunci când lansează un patch de software *patch*, compania calculează o semnătură digitală σ pentru *patch* folosind cheia *sk* și trimite fiecărui client perechea $(patch, \sigma)$;
- Fiecare client stabilește autenticitatea lui *patch* verificând dacă σ este o semnătură legitimă pentru *patch* cu privire la cheia publică *pk*;
- Deci compania folosește aceeași cheie publică pentru toți clienții și calculează o singură semnătură pe care o trimite tuturor.

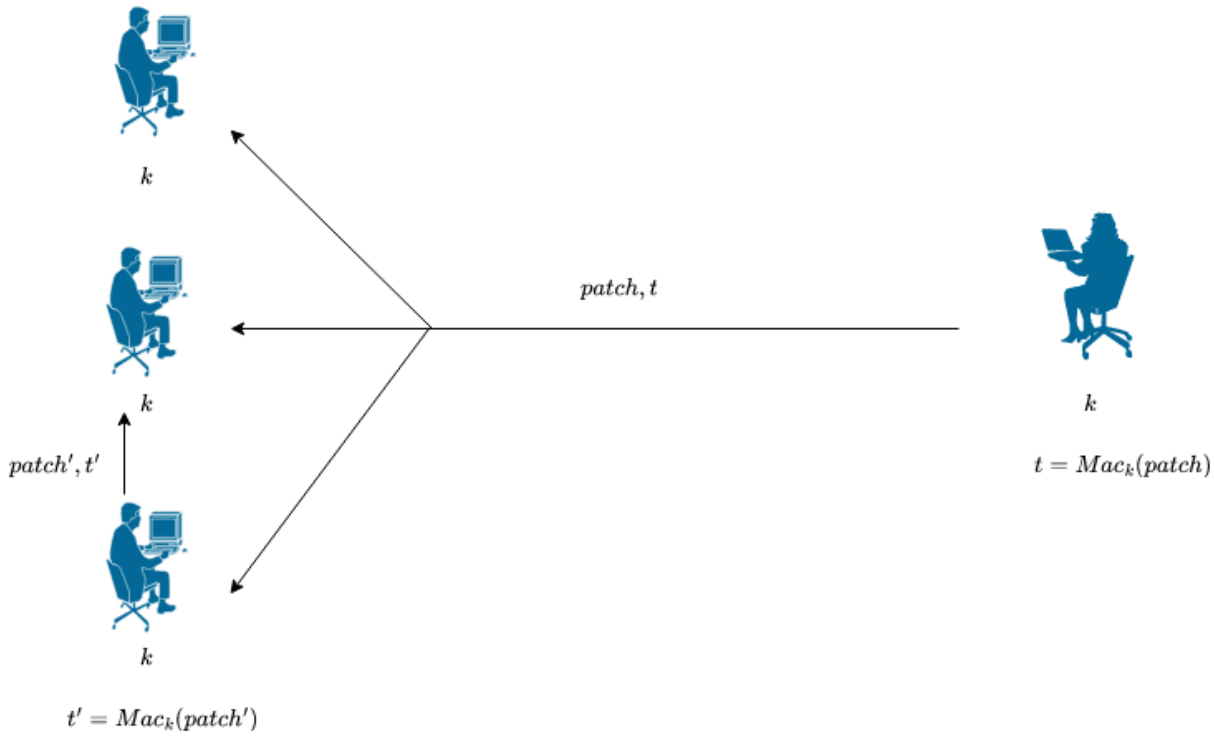
Aplicații ale semnăturilor digitale

- Putem înlocui semnătura digitală cu un MAC?



Aplicații ale semnăturilor digitale

- Putem înlocui semnătura digitală cu un MAC?



- In această situație, oricare dintre clienți poate emite un tag valid pentru un alt patch decât cel original și chiar îl poate trimite celorlalți clienți

Avantaje semnături digitale față de MAC-uri

- MAC-urile și schemele de semnătură digitală sunt folosite pentru asigurarea integrității (autenticității) mesajelor cu următoarele **diferențe**:

Avantaje semnături digitale față de MAC-uri

- MAC-urile și schemele de semnătură digitală sunt folosite pentru asigurarea integrității (autenticității) mesajelor cu următoarele **diferențe**:

- Schemele de semnătură digitală sunt *public verificabile*...

Avantaje semnături digitale față de MAC-uri

- MAC-urile și schemele de semnătură digitală sunt folosite pentru asigurarea integrității (autenticității) mesajelor cu următoarele **diferențe**:
- Schemele de semnătură digitală sunt *public verificabile*...
- ...ceea ce înseamnă că semnăturile digitale sunt *transferabile* - o terță parte poate verifica legitimitatea unei semnături și poate face o copie pentru a convinge pe altcineva că aceea este o semnătură validă pentru m;

Avantaje semnături digitale față de MAC-uri

- MAC-urile și schemele de semnătură digitală sunt folosite pentru asigurarea integrității (autenticității) mesajelor cu următoarele **diferențe**:
- Schemele de semnătură digitală sunt *public verificabile*...
- ...ceea ce înseamnă că semnăturile digitale sunt *transferabile* - o terță parte poate verifica legitimitatea unei semnături și poate face o copie pentru a convinge pe altcineva că aceea este o semnătură validă pentru m;
- Schemele de semnătură digitală au proprietatea de *non-repudiere* - un semnatar nu poate nega faptul că a semnat un mesaj;

Avantaje semnături digitale față de MAC-uri

- MAC-urile și schemele de semnătură digitală sunt folosite pentru asigurarea integrității (autenticității) mesajelor cu următoarele **diferențe**:
- Schemele de semnătură digitală sunt *public verificabile*...
- ...ceea ce înseamnă că semnăturile digitale sunt *transferabile* - o terță parte poate verifica legitimitatea unei semnături și poate face o copie pentru a convinge pe altcineva că aceea este o semnătură validă pentru m;
- Schemele de semnătură digitală au proprietatea de *non-repudiere* - un semnatar nu poate nega faptul că a semnat un mesaj;
- MAC-urile au avantajul că sunt cam de 2-3 ori mai eficiente (mai rapide) decât schemele de semnătură digitală.

Semnături digitale - Definiție

Definiție

O semnătură digitală definita peste $(\mathcal{K}, \mathcal{M}, \mathcal{S})$ *este formată din trei algoritmi polinomiali* $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ *unde:*

1. Gen*: este algoritmul de generare a perechii de cheie publică și cheie privată* (pk, sk)
2. $\mathrm{Sign} : \mathcal{K} \times \mathcal{M} \to \mathcal{S}$ *este algoritmul de generare a semnăturilor* $\sigma \leftarrow \mathrm{Sign}_{sk}(m)$*;*
3. $\mathrm{Vrfy} : \mathcal{K} \times \mathcal{M} \times \mathcal{S} \to \{0, 1\}$
   este algoritmul de verificare ce întoarce un bit $b = \mathrm{Vrfy}_{pk}(m, \sigma)$ *cu semnificația că:*
   - $b = 1$ *înseamnă* valid
   - $b = 0$ *înseamnă* invalid

a.î : $\forall m \in \mathcal{M},\ k \in \mathcal{K},\ \mathrm{Vrfy}_{pk}(m, \mathrm{Sign}_{sk}(m)) = 1.$

Securitate semnatură digitală - formalizare

- Formal, îi dăm adversarului acces la un *oracol* $\mathrm{Sign}_{sk}(\cdot)$;

Securitate semnatură digitală - formalizare

- Formal, îi dăm adversarului acces la un *oracol* $\mathrm{Sign}_{sk}(\cdot)$;
- Adversarul poate trimite orice mesaj m dorit către oracol și primește înapoi o semnătură corespunzătoare $\sigma \leftarrow \mathrm{Sign}_{sk}(m)$;

Securitate semnatură digitală - formalizare

- Formal, îi dăm adversarului acces la un *oracol* $\mathrm{Sign}_{sk}(\cdot)$;
- Adversarul poate trimite orice mesaj m dorit către oracol și primește înapoi o semnătură corespunzătoare $\sigma \leftarrow \mathrm{Sign}_{sk}(m)$;
- Considerăm că securitatea este impactată dacă adversarul este capabil să producă un mesaj m împreună cu o semnătură σ așa încât:

Securitate semnatură digitală - formalizare

- Formal, îi dăm adversarului acces la un *oracol* $\mathrm{Sign}_{sk}(\cdot)$;
- Adversarul poate trimite orice mesaj m dorit către oracol și primește înapoi o semnătură corespunzătoare $\sigma \leftarrow \mathrm{Sign}_{sk}(m)$;
- Considerăm că securitatea este impactată dacă adversarul este capabil să producă un mesaj m împreună cu o semnătură σ așa încât:
  1. σ este o semnătură validă pentru mesajul m: $\mathrm{Vrfy}_{pk}(m, \sigma) = 1$;

Securitate semnatură digitală - formalizare

- Formal, îi dăm adversarului acces la un *oracol* $\mathrm{Sign}_{sk}(\cdot)$;
- Adversarul poate trimite orice mesaj m dorit către oracol și primește înapoi o semnătură corespunzătoare $\sigma \leftarrow \mathrm{Sign}_{sk}(m)$;
- Considerăm că securitatea este impactată dacă adversarul este capabil să producă un mesaj m împreună cu o semnătură σ așa încât:
  1. σ este o semnătură validă pentru mesajul m: $\mathrm{Vrfy}_{pk}(m, \sigma) = 1$;
  2. Adversarul nu a solicitat anterior (de la oracol) o semnătură pentru mesajul m.

Securitate semnatură digitală - formalizare

- Despre o semnătură care satisface nivelul de securitate de mai sus spunem că *nu poate fi falsificată printr-un atac cu mesaj ales*;

Securitate semnatură digitală - formalizare

- Despre o semnătură care satisface nivelul de securitate de mai sus spunem că *nu poate fi falsificată printr-un atac cu mesaj ales*;
- Aceasta înseamnă că un adversar nu este capabil să falsifice o semnătură validă pentru nici un mesaj ...

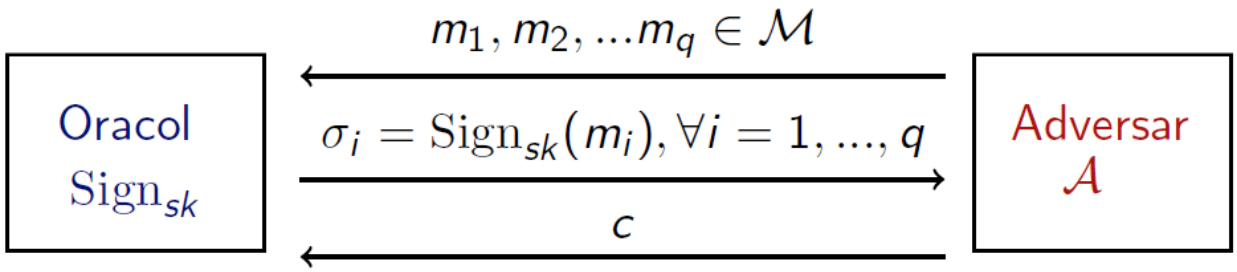
Securitate semnatură digitală - formalizare

- Despre o semnătură care satisface nivelul de securitate de mai sus spunem că *nu poate fi falsificată printr-un atac cu mesaj ales*;
- Aceasta înseamnă că un adversar nu este capabil să falsifice o semnătură validă pentru nici un mesaj ...
- ... deși poate obține semnături pentru orice mesaj ales de el, chiar *adaptiv* în timpul atacului.

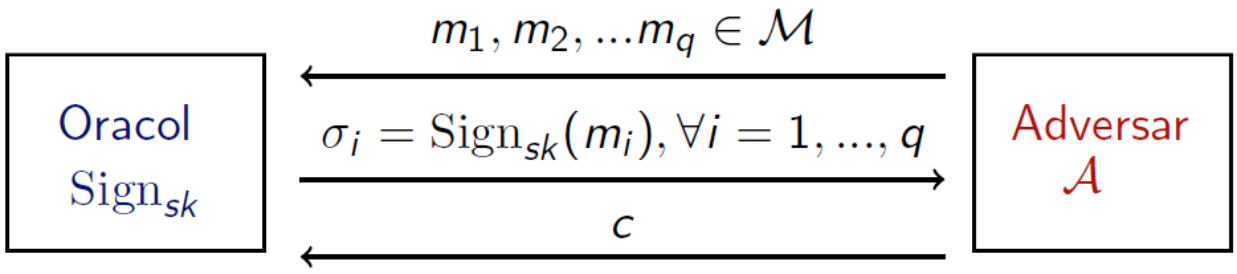
Securitate semnatură digitală - formalizare

- Despre o semnătură care satisface nivelul de securitate de mai sus spunem că *nu poate fi falsificată printr-un atac cu mesaj ales*;
- Aceasta înseamnă că un adversar nu este capabil să falsifice o semnătură validă pentru nici un mesaj ...
- ... deși poate obține semnături pentru orice mesaj ales de el, chiar *adaptiv* în timpul atacului.
- Pentru a da definiția formală, definim mai întâi un experiment pentru o semnătura $\pi = (\mathrm{Sign}, \mathrm{Vrfy})$, în care considerăm un adversar $\mathcal{A}$ și parametrul de securitate n;

Experimentul $\mathrm{Sign}_{\mathcal{A},\pi}^{forge}(n)$

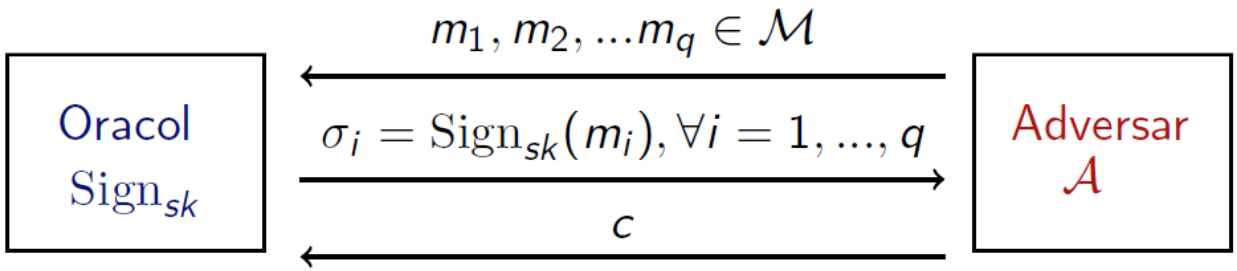


Experimentul $\mathrm{Sign}_{\mathcal{A},\pi}^{forge}(n)$



- Output-ul experimentului este 1 dacă și numai dacă: (1) $\mathrm{Vrfy}_k(m, t) = 1$ si (2) $m \notin \{m_1, ..., m_q\}$;

Experimentul $\mathrm{Sign}^{forge}_{\mathcal{A},\pi}(n)$



- Output-ul experimentului este 1 dacă și numai dacă:
  (1) $\mathrm{Vrfy}_k(m, t) = 1$ si (2) $m \notin \{m_1, ..., m_q\}$;
- Dacă $\mathrm{Sign}^{forge}_{\mathcal{A},\pi}(n) = 1$, spunem că $\mathcal{A}$ a efectuat experimentul cu succes.

Securitate semnaturi digitale

Definiție

O semnătură $\pi = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ *este sigură (nu poate fi falsificată printr-un atac cu mesaj ales) dacă pentru orice adversar polinomial* $\mathcal{A}$ *există o funcție neglijabilă* $negl$ *așa încât*

$$Pr[\mathrm{Sign}^{forge}_{\mathcal{A},\pi}(n) = 1] \leq negl(n).$$

Atacuri prin replicare

- Un atacator poate prelua o pereche de mesaj și semnătură digitală și o retrimite unui destinatar

Atacuri prin replicare

- Un atacator poate prelua o pereche de mesaj și semnătură digitală și o retrimite unui destinatar
- la fel ca în criptografia simetrică, definiția semnăturilor digitale nu protejează împotriva acestui tip de atac

Atacuri prin replicare

- Un atacator poate prelua o pereche de mesaj și semnătură digitală și o retrimite unui destinatar
- la fel ca în criptografia simetrică, definiția semnăturilor digitale nu protejează împotriva acestui tip de atac
- in contextul exemplului cu patch-ul de software, acest atac este problematic

Construcție scheme de semnătură digitală

Construcție scheme de semnătură digitală

- Paradigma "hash-and-sign" este sigură: înainte de semnare, mesajul trece printr-o funcție hash; varianta aceasta se folosește pe larg în practică;
  - $\sigma \leftarrow \mathrm{Sign}_{sk}(H(m))$; $\mathrm{Vrfy}_{\mathrm{pk}}(\mathrm{H(m)}) \stackrel{?}{=} \sigma$

Construcție scheme de semnătură digitală

- Paradigma "hash-and-sign" este sigură: înainte de semnare, mesajul trece printr-o funcție hash; varianta aceasta se folosește pe larg în practică;
  - $\sigma \leftarrow \mathrm{Sign}_{sk}(H(m))$; $\mathrm{Vrfy}_{\mathrm{pk}}(\mathrm{H}(\mathrm{m})) \stackrel{?}{=} \sigma$
- construcția este sigură atâta timp cât H este rezistentă la coliziuni

Construcție scheme de semnătură digitală

- Paradigma "hash-and-sign" este sigură: înainte de semnare, mesajul trece printr-o funcție hash; varianta aceasta se folosește pe larg în practică;
  - $\sigma \leftarrow \mathrm{Sign}_{sk}(H(m))$; $\mathrm{Vrfy}_{\mathrm{pk}}(\mathrm{H}(\mathrm{m})) \stackrel{?}{=} \sigma$
- construcția este sigură atâta timp cât H este rezistentă la coliziuni
- este avantajoasă pentru că oferă funcționalitatea unei semnături digitale (criptografie cu cheie publică) la costul unei operații din criptografia cu cheie secretă

Construcție scheme de semnătură digitală

- Paradigma "hash-and-sign" este sigură: înainte de semnare, mesajul trece printr-o funcție hash; varianta aceasta se folosește pe larg în practică;
  - $\sigma \leftarrow \mathrm{Sign}_{sk}(H(m))$; $\mathrm{Vrfy}_{\mathrm{pk}}(\mathrm{H}(\mathrm{m})) \stackrel{?}{=} \sigma$
- construcția este sigură atâta timp cât H este rezistentă la coliziuni
- este avantajoasă pentru că oferă funcționalitatea unei semnături digitale (criptografie cu cheie publică) la costul unei operații din criptografia cu cheie secretă
- folosită pe larg în practică

Prezumpția RSA - reamintire

- Se aleg două numere mari prime p și q

Prezumpția RSA - reamintire

- Se aleg două numere mari prime p și q
- Se calculează modulul $N = p \cdot q$

Prezumpția RSA - reamintire

- Se aleg două numere mari prime p și q
- Se calculează modulul $N = p \cdot q$
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;

Prezumpția RSA - reamintire

- Se aleg două numere mari prime p și q
- Se calculează modulul $N = p \cdot q$
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = (x^d)^e$

Prezumpția RSA - reamintire

- Se aleg două numere mari prime p și q
- Se calculează modulul $N = p \cdot q$
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = (x^d)^e$
- x^d se numeste rădăcina de ordin e a lui x modulo N

Prezumpția RSA - reamintire

- Se aleg două numere mari prime p și q
- Se calculează modulul $N = p \cdot q$
- Fie $\mathbb{Z}_N^*$ un grup de ordin $\phi(N) = (p-1)(q-1)$;
- Fixăm e cu $gcd(e, \phi(N)) = 1$. Atunci $(x^e)^d = x^{ed \bmod \phi(N)} = x \bmod N = (x^d)^e$
- x^d se numeste rădăcina de ordin e a lui x modulo N
- Prezumpția RSA: cunoscându-se doar N și e, este dificil să calculăm rădăcina de ordin e a unui mesaj $m \in \mathbb{Z}_N^*$

Semnatura digitală bazată pe RSA

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$

Semnatura digitală bazată pe RSA

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = m^d \bmod N$$

Semnatura digitală bazată pe RSA

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = m^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

Semnatura digitală bazată pe RSA

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = m^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

$$m \stackrel{?}{=} \sigma^e \bmod N$$

Această variantă de semnătură nu este sigură, se cunosc mai multe atacuri pentru ea

Un atac asupra semnăturii bazate pe RSA

- scop adversar: falsificarea semnăturii mesajului $m \in \mathbb{Z}_N^*$ pentru $pk = (N, e)$

Un atac asupra semnăturii bazate pe RSA

- scop adversar: falsificarea semnăturii mesajului $m \in \mathbb{Z}_N^*$ pentru $pk = (N, e)$
- acțiune adversar: alege $m_1, m_2 \in \mathbb{Z}_N^*$ a.i. $m = m_1 \cdot m_2 \bmod N$

Un atac asupra semnăturii bazate pe RSA

- scop adversar: falsificarea semnăturii mesajului $m \in \mathbb{Z}_N^*$ pentru $pk = (N, e)$
- acțiune adversar: alege $m_1, m_2 \in \mathbb{Z}_N^*$ a.i. $m = m_1 \cdot m_2 \bmod N$
- obține semnăturile σ_1 și σ_2 pentru mesajele m_1, m_2

Un atac asupra semnăturii bazate pe RSA

- scop adversar: falsificarea semnăturii mesajului $m \in \mathbb{Z}_N^*$ pentru $pk = (N, e)$
- acțiune adversar: alege $m_1, m_2 \in \mathbb{Z}_N^*$ a.i. $m = m_1 \cdot m_2 \bmod N$
- obține semnăturile σ_1 și σ_2 pentru mesajele m_1, m_2
- întoarce $\sigma = \sigma_1 \cdot \sigma_2 \bmod N$ ca semnătură validă pentru m

Un atac asupra semnăturii bazate pe RSA

- scop adversar: falsificarea semnăturii mesajului $m \in \mathbb{Z}_N^*$ pentru $pk = (N, e)$
- acțiune adversar: alege $m_1, m_2 \in \mathbb{Z}_N^*$ a.i. $m = m_1 \cdot m_2 \bmod N$
- obține semnăturile σ_1 și σ_2 pentru mesajele m_1, m_2
- întoarce $\sigma = \sigma_1 \cdot \sigma_2 \bmod N$ ca semnătură validă pentru m
- aceasta este o semnătură validă pentru că

$$\sigma^e = (\sigma_1 \cdot \sigma_2)^e = (m_1^d \cdot m_2^d)^e = m_1^{ed} \cdot m_2^{ed} = m_1 \cdot m_2 = m \bmod N$$

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

$$H(m) \stackrel{?}{=} \sigma^e \bmod N$$

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

$$H(m) \stackrel{?}{=} \sigma^e \bmod N$$

- Se poate verifica ușor că atacul precedent nu funcționează: $H(m_1) \cdot H(m_1) = \sigma_1^e \cdot \sigma_2^e = (\sigma_1 \cdot \sigma_2)^e \neq H(m_1 \cdot m_2)$

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

$$H(m) \stackrel{?}{=} \sigma^e \bmod N$$

- Se poate verifica ușor că atacul precedent nu funcționează: $H(m_1) \cdot H(m_1) = \sigma_1^e \cdot \sigma_2^e = (\sigma_1 \cdot \sigma_2)^e \neq H(m_1 \cdot m_2)$
  Această variantă de semnătură este sigură daca H îndeplinește două condiții:

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

$$H(m) \stackrel{?}{=} \sigma^e \bmod N$$

- Se poate verifica ușor că atacul precedent nu funcționează: $H(m_1) \cdot H(m_1) = \sigma_1^e \cdot \sigma_2^e = (\sigma_1 \cdot \sigma_2)^e \neq H(m_1 \cdot m_2)$
  Această variantă de semnătură este sigură daca H îndeplinește două condiții:
  - H este rezistentă la coliziuni

Varianta RSA-FDH (full-domain hash)

- Gen: generează N, e, d și $pk = (N, e)$ iar $sk = d$
- $\mathrm{Sign}(sk, m)$: semnează mesajul m folosind cheia $sk = d$ astfel

$$\sigma = H(m)^d \bmod N$$

- $\mathrm{Vrfy}(pk, m, \sigma)$: semnătura este validă dacă și numai dacă

$$H(m) \stackrel{?}{=} \sigma^e \bmod N$$

- Se poate verifica ușor că atacul precedent nu funcționează: $H(m_1) \cdot H(m_1) = \sigma_1^e \cdot \sigma_2^e = (\sigma_1 \cdot \sigma_2)^e \neq H(m_1 \cdot m_2)$
  Această variantă de semnătură este sigură daca H îndeplinește două condiții:
  - H este rezistentă la coliziuni
  - $H : \{0,1\}^* \to \mathbb{Z}_N^*$

Scheme de identificare - identification schemes

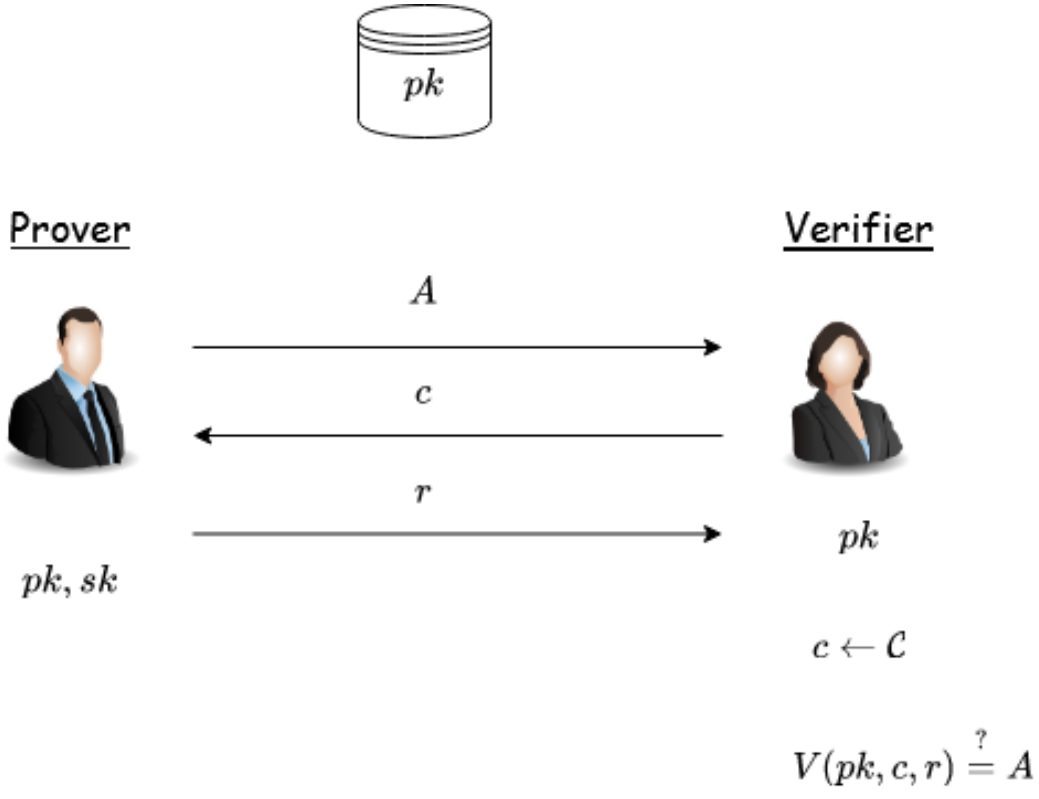
- sunt protocoale interactive care permit unei părti (Prover) să își demonstreze identitatea în fața unei alte părti (Verifier)

Scheme de identificare - identification schemes

- sunt protocoale interactive care permit unei părti (Prover) să își demonstreze identitatea în fața unei alte părti (Verifier)
- sunt foarte importante ca building block pentru semnături digitale (dar, în sine, au aplicabilitate limitată)

Scheme de identificare - identification schemes

- sunt protocoale interactive care permit unei părti (Prover) să își demonstreze identitatea în fața unei alte părti (Verifier)
- sunt foarte importante ca building block pentru semnături digitale (dar, în sine, au aplicabilitate limitată)
- în continuare, abordare informală



Scheme de identificare

- Securitate
  - față de adversarii *pasivi* - chiar dacă are acces la mesajele trimise în mai multe execuții ale protocolului, un adversar nu îl poate convinge pe Verifier să accepte

Scheme de identificare

- Securitate
  - față de adversarii *pasivi* - chiar dacă are acces la mesajele trimise în mai multe execuții ale protocolului, un adversar nu îl poate convinge pe Verifier să accepte
- Principala aplicație

Scheme de identificare

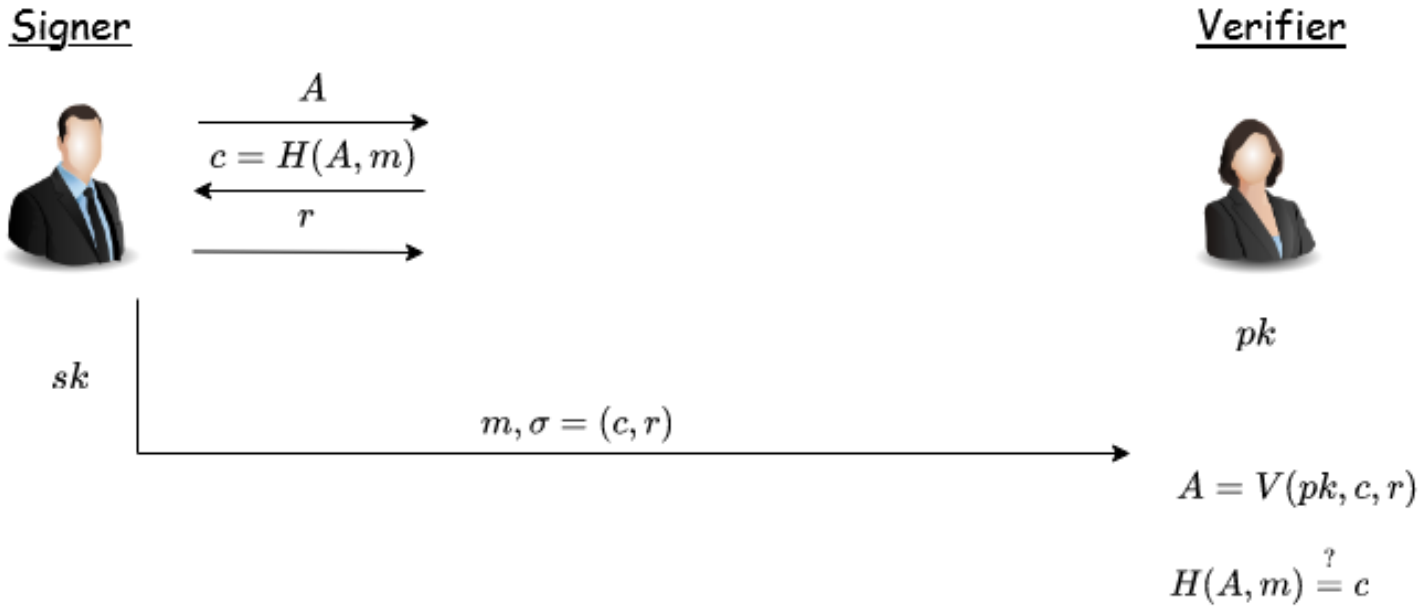
- Securitate
  - față de adversarii *pasivi* - chiar dacă are acces la mesajele trimise în mai multe execuții ale protocolului, un adversar nu îl poate convinge pe Verifier să accepte
- Principala aplicație
  - identificarea persoanelor prezente fizic; de pildă, deschiderea unei uși securizate pe baza unei cartele de acces

Scheme de identificare

- Securitate
  - față de adversarii *pasivi* - chiar dacă are acces la mesajele trimise în mai multe execuții ale protocolului, un adversar nu îl poate convinge pe Verifier să accepte
- Principala aplicație
  - identificarea persoanelor prezente fizic; de pildă, deschiderea unei uși securizate pe baza unei cartele de acces
  - nu este potrivită pentru autentificarea la distanță (pe internet)

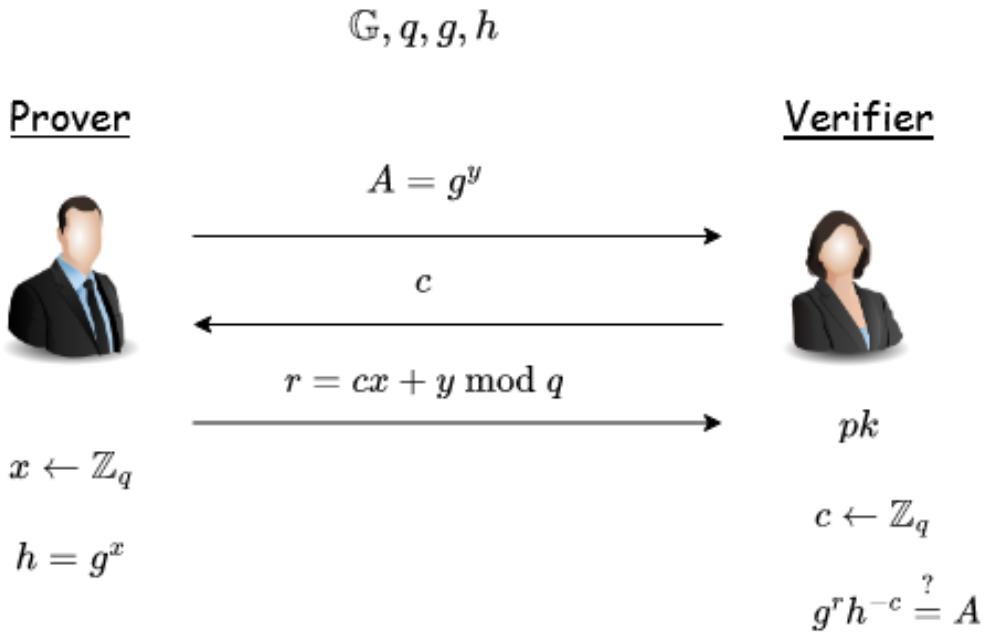
Transformarea schemelor de identificare in semnături digitale

- Pentru a semna, Prover-ul execută singur protocolul generând challenge-ul pe baza unei funcții hash - folosește *transformarea Fiat-Shamir*



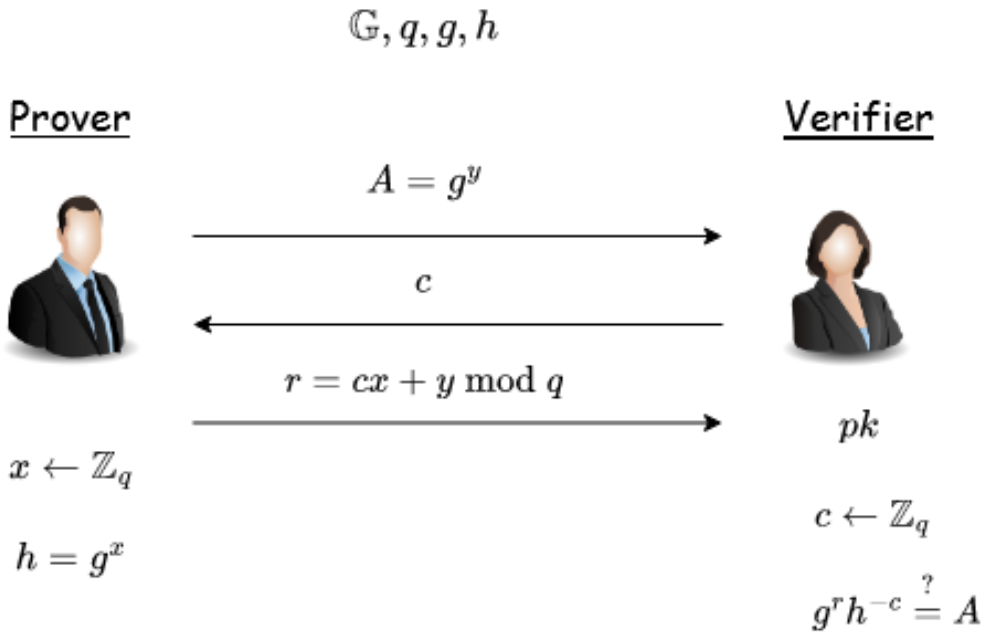
Schema de identificare Schnorr

- In schema de identificare de mai jos $\mathbb{G}$ este un grup ciclic de ordin q și generator g, $sk = x$ și $pk = (\mathbb{G}, q, g, y)$ unde $h = g^x$.



Schema de identificare Schnorr

- In schema de identificare de mai jos $\mathbb{G}$ este un grup ciclic de ordin q și generator g, $sk = x$ și $pk = (\mathbb{G}, q, g, y)$ unde $h = g^x$.



- Se poate verifica ușor ca $g^r h^{-c} = g^{cx+y} h^{-c} = (g^x)^c g^y h^{-c} = g^y = A$

Schema de identificare Schnorr - securitate

- Dacă problema logaritmului discret este dificilă, atunci schema de identificare Schnorr este sigura împotriva atacurilor pasive

Schema de identificare Schnorr - securitate

- Dacă problema logaritmului discret este dificilă, atunci schema de identificare Schnorr este sigura împotriva atacurilor pasive
- Dacă problema logaritmului discret este dificilă si H este modelata ca o funcție aleatoare, atunci semnătura Schnorr este sigură

Alte scheme de semnatură digitală

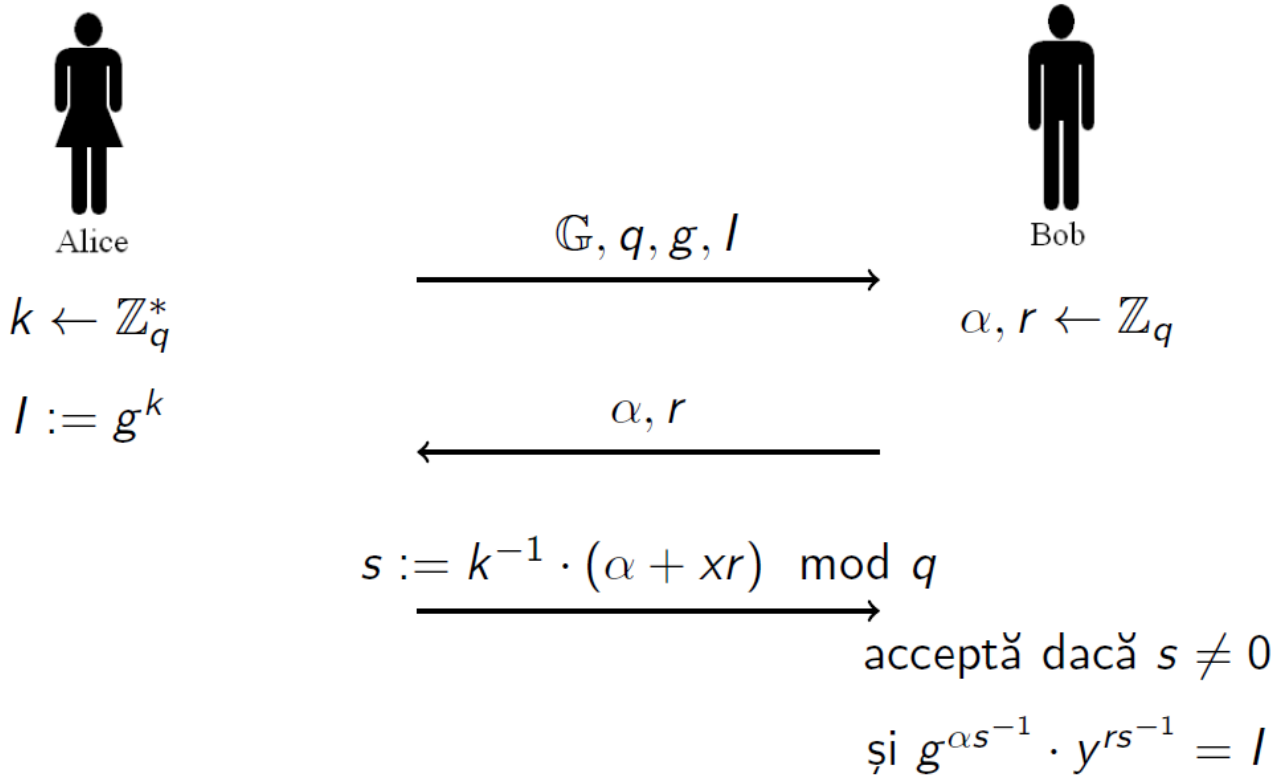
- Un alt exemplu folosit în practică este Digital Signature Algorithm (DSA) bazat pe problema logaritmului discret (a devenit standard US în 1994) dar și ECDSA (varianta DSA bazată pe curbe eliptice devenită standard în 1998), ambele fiind incluse în DSS (Digital Signature Standard).

Alte scheme de semnatură digitală

- Un alt exemplu folosit în practică este Digital Signature Algorithm (DSA) bazat pe problema logaritmului discret (a devenit standard US în 1994) dar și ECDSA (varianta DSA bazată pe curbe eliptice devenită standard în 1998), ambele fiind incluse în DSS (Digital Signature Standard).
- Atât DSA cât și ECDSA se bazează pe PLD în diferite clase de grupuri.

DSA/ECDSA

Sunt construite pe baza schemei de identificare de mai jos unde $\mathbb{G}$ este un grup ciclic de ordin q și generator g, $sk = x$ și $pk = (\mathbb{G}, q, g, y)$ unde $y = g^x$.



DSA/ECDSA

- Se poate verifica ușor că schema este corectă:
  $s = 0$ doar dacă $\alpha = -xr \mod q$ ceea ce se întâmplă cu probabilitate neglijabilă.
- Considerând $s \neq 0$, $s^{-1} \mod q$ există și

$$g^{\alpha s^{-1}} \cdot y^{rs^{-1}} = g^{\alpha s^{-1}} \cdot g^{xrs^{-1}} = g^{(\alpha + xr)s^{-1}} = g^{(\alpha + xr) \cdot k \cdot (\alpha + xr)^{-1}} = I$$

DSA/ECDSA

- Se poate verifica ușor că schema este corectă:
  $s = 0$ doar dacă $\alpha = -xr \mod q$ ceea ce se întâmplă cu probabilitate neglijabilă.
- Considerând $s \neq 0$, $s^{-1} \mod q$ există și

$$g^{\alpha s^{-1}} \cdot y^{rs^{-1}} = g^{\alpha s^{-1}} \cdot g^{xrs^{-1}} = g^{(\alpha + xr)s^{-1}} = g^{(\alpha + xr) \cdot k \cdot (\alpha + xr)^{-1}} = I$$

- Schema anterioară este sigură dacă PLD este dificilă în grupul $\mathbb{G}$.

DSA/ECDSA

- Se poate verifica ușor că schema este corectă: $s = 0$ doar dacă $\alpha = -xr \mod q$ ceea ce se întâmplă cu probabilitate neglijabilă.
- Considerând $s \neq 0$, $s^{-1} \mod q$ există și

$$g^{\alpha s^{-1}} \cdot y^{rs^{-1}} = g^{\alpha s^{-1}} \cdot g^{xrs^{-1}} = g^{(\alpha + xr)s^{-1}} = g^{(\alpha + xr) \cdot k \cdot (\alpha + xr)^{-1}} = I$$

- Schema anterioară este sigură dacă PLD este dificilă în grupul $\mathbb{G}$.
- Schemele de semnătură DSA/ECDSA se obțin prin transformarea schemei de sus într-una non-interactivă.

DSA/ECDSA

Modificările pentru a face schema non-interactivă sunt următoarele:

- notăm $\alpha = H(m)$ unde H este o funcție hash criptografică
- $r = F(I)$ pentru o funcție specifică $F : \mathbb{G} \to \mathbb{Z}_q$
- Varianta non-interactivă a schemei este mai jos
  - Gen: generează $\mathbb{G}$ un grup ciclic de ordin q și un generator g, alege uniform $x \in \mathbb{Z}_q$ și $y = g^x$. Cheia publică este $pk = (\mathbb{G}, q, g, y)$ iar cheia secretă este $sk = x$.
    Se aleg și functiile $H : \{0,1\}^* \to \mathbb{Z}_q$ și $F : \mathbb{G} \to \mathbb{Z}_q$.

DSA/ECDSA

Modificările pentru a face schema non-interactivă sunt următoarele:

- notăm $\alpha = H(m)$ unde H este o funcție hash criptografică
- $r = F(I)$ pentru o funcție specifică $F : \mathbb{G} \to \mathbb{Z}_q$
- Varianta non-interactivă a schemei este mai jos
  - Gen: generează $\mathbb{G}$ un grup ciclic de ordin q și un generator g, alege uniform $x \in \mathbb{Z}_q$ și $y = g^x$. Cheia publică este $pk = (\mathbb{G}, q, g, y)$ iar cheia secretă este $sk = x$.
    Se aleg și functiile $H : \{0,1\}^* \to \mathbb{Z}_q$ și $F : \mathbb{G} \to \mathbb{Z}_q$.

  - $\mathrm{Sign}(sk, m)$: alege uniform $k \in \mathbb{Z}_q^*$ și $r = F(g^k)$. Calculează $s := k^{-1} \cdot (H(m) + xr) \mod q$. Dacă $s = 0$ sau $r = 0$ se re-începe cu o nouă alegere a lui k.
    Semnătura rezultată este (r, s).

DSA/ECDSA

Modificările pentru a face schema non-interactivă sunt următoarele:

- notăm $\alpha = H(m)$ unde H este o funcție hash criptografică
- $r = F(I)$ pentru o funcție specifică $F : \mathbb{G} \to \mathbb{Z}_q$
- Varianta non-interactivă a schemei este mai jos
  - Gen: generează $\mathbb{G}$ un grup ciclic de ordin q și un generator g, alege uniform $x \in \mathbb{Z}_q$ și $y = g^x$. Cheia publică este $pk = (\mathbb{G}, q, g, y)$ iar cheia secretă este $sk = x$.
    Se aleg și functiile $H : \{0,1\}^* \to \mathbb{Z}_q$ și $F : \mathbb{G} \to \mathbb{Z}_q$.
  - $\mathrm{Sign}(sk, m)$: alege uniform $k \in \mathbb{Z}_q^*$ și $r = F(g^k)$. Calculează $s := k^{-1} \cdot (H(m) + xr) \mod q$. Dacă $s = 0$ sau $r = 0$ se re-începe cu o nouă alegere a lui k.
    Semnătura rezultată este (r, s).
  - $\mathrm{Vrfy}(pk, m, (r, s))$: semnătura este validă dacă și numai dacă

DSA/ECDSA

Modificările pentru a face schema non-interactivă sunt următoarele:

- notăm $\alpha = H(m)$ unde H este o funcție hash criptografică
- $r = F(I)$ pentru o funcție specifică $F : \mathbb{G} \to \mathbb{Z}_q$
- Varianta non-interactivă a schemei este mai jos
  - Gen: generează $\mathbb{G}$ un grup ciclic de ordin q și un generator g, alege uniform $x \in \mathbb{Z}_q$ și $y = g^x$. Cheia publică este $pk = (\mathbb{G}, q, g, y)$ iar cheia secretă este $sk = x$.
    Se aleg și functiile $H : \{0,1\}^* \to \mathbb{Z}_q$ și $F : \mathbb{G} \to \mathbb{Z}_q$.
  - $\mathrm{Sign}(sk, m)$: alege uniform $k \in \mathbb{Z}_q^*$ și $r = F(g^k)$. Calculează $s := k^{-1} \cdot (H(m) + xr) \mod q$. Dacă $s = 0$ sau $r = 0$ se re-începe cu o nouă alegere a lui k.
    Semnătura rezultată este (r, s).
  - $\mathrm{Vrfy}(pk, m, (r, s))$: semnătura este validă dacă și numai dacă

$$r \stackrel{?}{=} F(g^{H(m)\cdot s^{-1}} y^{r\cdot s^{-1}})$$

DSA/ECDSA - Securitate

- Securitatea semnăturii DSA/ECDSA se bazează pe problema logaritmului discret și pe faptul că F și G sunt alese corespunzător.

DSA/ECDSA - Securitate

- Securitatea semnăturii DSA/ECDSA se bazează pe problema logaritmului discret și pe faptul că F și G sunt alese corespunzător.
- Este foarte important ca la generarea semnăturii k să fie ales aleator, și deci sa nu fie predictibil. In caz contrar, cheia secretă x se poate afla (în ecuația $s = k^{-1} \cdot (H(m) + xr)$ mod q, singura necunoscută este x).

DSA/ECDSA - Securitate

- Securitatea semnăturii DSA/ECDSA se bazează pe problema logaritmului discret și pe faptul că F și G sunt alese corespunzător.
- Este foarte important ca la generarea semnăturii k să fie ales aleator, și deci sa nu fie predictibil. In caz contrar, cheia secretă x se poate afla (în ecuația $s = k^{-1} \cdot (H(m) + xr)$ mod q, singura necunoscută este x).
- De asemenea, folosirea aceluiași k pentru generarea a două semnături diferite duce la găsirea cheii secrete.

Certificate și PKI

- O problemă a criptografiei cu cheie publică o reprezintă distribuirea cheilor publice;

Certificate și PKI

- O problemă a criptografiei cu cheie publică o reprezintă distribuirea cheilor publice;
- Se rezolvă tot cu criptografia cu cheie publică: e suficient să distribuim o singură cheie publică în mod sigur...

Certificate și PKI

- O problemă a criptografiei cu cheie publică o reprezintă distribuirea cheilor publice;
- Se rezolvă tot cu criptografia cu cheie publică: e suficient să distribuim o singură cheie publică în mod sigur...
- Ulterior ea poate fi folosită pentru a distribui sigur oricât de multe chei publice;

Certificate și PKI

- O problemă a criptografiei cu cheie publică o reprezintă distribuirea cheilor publice;
- Se rezolvă tot cu criptografia cu cheie publică: e suficient să distribuim o singură cheie publică în mod sigur...
- Ulterior ea poate fi folosită pentru a distribui sigur oricât de multe chei publice;
- Ideea constă în folosirea unui *certificat digital* care este o semnătură care atașează unei entități o anume cheie publică;

Certificate și PKI

- De exemplu, dacă Charlie are cheia generată (pk_C, sk_C) iar Bob are cheia (pk_B, sk_B), iar Charlie cunoaște pk_B atunci el poate calcula semnătura de mai jos pe care i-o dă lui Bob:

$$\mathrm{cert}_{C \to B} = \mathrm{Sign}_{sk_C}(\text{"}\textit{Cheia lui Bob este } pk_B\text{"})$$

Certificate și PKI

- De exemplu, dacă Charlie are cheia generată (pk_C, sk_C) iar Bob are cheia (pk_B, sk_B), iar Charlie cunoaște pk_B atunci el poate calcula semnătura de mai jos pe care i-o dă lui Bob:

$$\mathrm{cert}_{C \to B} = \mathrm{Sign}_{sk_C}(\text{"}\textit{Cheia lui Bob este } pk_B\text{"})$$

- Această semnătură este un *certificat* emis de Charlie pentru Bob;

Certificate și PKI

- De exemplu, dacă Charlie are cheia generată (pk_C, sk_C) iar Bob are cheia (pk_B, sk_B), iar Charlie cunoaște pk_B atunci el poate calcula semnătura de mai jos pe care i-o dă lui Bob:

$$\mathrm{cert}_{C \to B} = \mathrm{Sign}_{sk_C}(\text{"}\textit{Cheia lui Bob este } pk_B\text{"})$$

- Această semnătură este un *certificat* emis de Charlie pentru Bob;

- Atunci când Bob vrea să comunice cu Alice, îi trimite întâi cheia publică pk_B împreună cu certificatul $\mathrm{cert}_{C \to B}$ a cărui validitate în raport cu pk_C Alice o verifică;

Certificate și PKI

- Rămân câteva probleme: cum află Alice pk_C, cum poate fi Charlie sigur că pk_B este cheia publică a lui Bob, cum decide Alice dacă să aibă încredere în Charlie;

Certificate și PKI

- Rămân câteva probleme: cum află Alice pk_C, cum poate fi Charlie sigur că pk_B este cheia publică a lui Bob, cum decide Alice dacă să aibă încredere în Charlie;
- Toate aceastea sunt specificate într-o *infrastructură cu chei publice* (PKI-public key infrastructure) care permite distribuirea la scară largă a cheilor publice;

Certificate și PKI

- Rămân câteva probleme: cum află Alice pk_C, cum poate fi Charlie sigur că pk_B este cheia publică a lui Bob, cum decide Alice dacă să aibă încredere în Charlie;
- Toate aceastea sunt specificate într-o *infrastructură cu chei publice* (PKI-public key infrastructure) care permite distribuirea la scară largă a cheilor publice;
- Există mai multe modele diferite de PKI, după cum vom vedea în continuare;

PKI cu o singură autoritate de certificare

- Aici există o singură autoritate de certificare (CA) în care toată lumea are încredere și care emite certificate pentru toate cheile publice;

PKI cu o singură autoritate de certificare

- Aici există o singură autoritate de certificare (CA) în care toată lumea are încredere și care emite certificate pentru toate cheile publice;
- CA este o companie, sau agenție guvernamentală sau un departament dintr-o organizație;

PKI cu o singură autoritate de certificare

- Aici există o singură autoritate de certificare (CA) în care toată lumea are încredere și care emite certificate pentru toate cheile publice;
- CA este o companie, sau agenție guvernamentală sau un departament dintr-o organizație;
- Oricine apelează la serviciile CA trebuie să obțină o copie legitimă a cheii ei publice pk_{CA};

PKI cu o singură autoritate de certificare

- Aici există o singură autoritate de certificare (CA) în care toată lumea are încredere și care emite certificate pentru toate cheile publice;
- CA este o companie, sau agenție guvernamentală sau un departament dintr-o organizație;
- Oricine apelează la serviciile CA trebuie să obțină o copie legitimă a cheii ei publice pk_{CA};
- Cheia pk_{CA} se obține chiar prin mijloace fizice; deși inconvenient, acest pas este efectuat o singură dată;

PKI cu mai multe autorități de certificare

- Modelul cu o singură CA nu este practic;

PKI cu mai multe autorități de certificare

- Modelul cu o singură CA nu este practic;
- In modelul cu multiple CA, dacă Bob dorește să obțină un certificat pentru cheia lui publică, poate apela la oricare CA dorește, iar Alice, care primește un certificat sau mai multe, poate alege în care CA să aibă încredere;

PKI cu mai multe autorități de certificare

- Modelul cu o singură CA nu este practic;
- In modelul cu multiple CA, dacă Bob dorește să obțină un certificat pentru cheia lui publică, poate apela la oricare CA dorește, iar Alice, care primește un certificat sau mai multe, poate alege în care CA să aibă încredere;
- De exemplu, browser-ele web vin preconfigurate cu un număr de chei publice ale unor CA stabilite ca toate fiind de încredere în mod egal (în configurația default a browser-ului);

PKI cu mai multe autorități de certificare

- Modelul cu o singură CA nu este practic;
- In modelul cu multiple CA, dacă Bob dorește să obțină un certificat pentru cheia lui publică, poate apela la oricare CA dorește, iar Alice, care primește un certificat sau mai multe, poate alege în care CA să aibă încredere;
- De exemplu, browser-ele web vin preconfigurate cu un număr de chei publice ale unor CA stabilite ca toate fiind de încredere în mod egal (în configurația default a browser-ului);
- Utilizatorul poate modifica această configurație așa încât să accepte doar certificate de la CA-uri în care el are încredere;

Delegare și lanțuri de certificate

- Charlie este un CA care emite certificate, inclusiv pentru Bob;

Delegare și lanțuri de certificate

- Charlie este un CA care emite certificate, inclusiv pentru Bob;
- Dacă pk_B este o cheie publică pentru semnătură, atunci Bob poate emite certificate pentru alte persoane; un certificat pentru Alice are forma

$$\mathrm{cert}_{B \to A} = \mathrm{Sign}_{sk_B}(\text{"}\textit{Cheia lui Alice este } pk_A\text{"})$$

Delegare și lanțuri de certificate

- Charlie este un CA care emite certificate, inclusiv pentru Bob;
- Dacă pk_B este o cheie publică pentru semnătură, atunci Bob poate emite certificate pentru alte persoane; un certificat pentru Alice are forma

$$\mathrm{cert}_{B\to A} = \mathrm{Sign}_{sk_B}(\text{"}\textit{Cheia lui Alice este } pk_A\text{"})$$

- Atunci când comunică cu Dan, Alice îi trimite

$$pk_A, \mathrm{cert}_{B\to A}, pk_B, \mathrm{cert}_{C\to B}$$

Delegare și lanțuri de certificate

- Charlie este un CA care emite certificate, inclusiv pentru Bob;
- Dacă pk_B este o cheie publică pentru semnătură, atunci Bob poate emite certificate pentru alte persoane; un certificat pentru Alice are forma

$$\mathrm{cert}_{B\to A} = \mathrm{Sign}_{sk_B}(\text{"}\textit{Cheia lui Alice este } pk_A\text{"})$$

- Atunci când comunică cu Dan, Alice îi trimite

$$pk_A, \mathrm{cert}_{B\to A}, pk_B, \mathrm{cert}_{C\to B}$$

- De fapt, $\mathrm{cert}_{C\to B}$ conține, în afară de pk_B și afirmația "Bob este de încredere pentru a emite certificate"; astfel, Charlie îl deleagă pe Bob să emită certificate;

Delegare și lanțuri de certificate

- Charlie este un CA care emite certificate, inclusiv pentru Bob;
- Dacă pk_B este o cheie publică pentru semnătură, atunci Bob poate emite certificate pentru alte persoane; un certificat pentru Alice are forma

$$\mathrm{cert}_{B\to A} = \mathrm{Sign}_{sk_B}(\text{"}\textit{Cheia lui Alice este } pk_A\text{"})$$

- Atunci când comunică cu Dan, Alice îi trimite

$$pk_A, \mathrm{cert}_{B\to A}, pk_B, \mathrm{cert}_{C\to B}$$

- De fapt, $\mathrm{cert}_{C\to B}$ conține, în afară de pk_B și afirmația "Bob este de încredere pentru a emite certificate"; astfel, Charlie îl deleagă pe Bob să emită certificate;
- Totul se poate organiza ca o ierarhie unde există un CA "rădăcină" pe primul nivel și n CA-uri pe al doilea nivel.

Modelul "web of trust"

- Aici oricine poate emite certificate pentru orice altcineva și fiecare utilizator decide cât de multă încredere poate acorda certificatelor emise de alți utilizatori;

Modelul "web of trust"

- Aici oricine poate emite certificate pentru orice altcineva și fiecare utilizator decide cât de multă încredere poate acorda certificatelor emise de alți utilizatori;
- De exemplu, dacă Alice are cheile publice pk_1, pk_2, pk_3 corespunzătoare lui C_1, C_2, C_3...

Modelul "web of trust"

- Aici oricine poate emite certificate pentru orice altcineva și fiecare utilizator decide cât de multă încredere poate acorda certificatelor emise de alți utilizatori;
- De exemplu, dacă Alice are cheile publice pk_1, pk_2, pk_3 corespunzătoare lui C_1, C_2, C_3...
- ...iar Bob, care vrea să comunice cu Alice, are certificatele $\mathrm{cert}_{C_1 \to B}$, $\mathrm{cert}_{C_3 \to B}$ și $\mathrm{cert}_{C_4 \to B}$ pe care i le trimite lui Alice;

Modelul "web of trust"

- Aici oricine poate emite certificate pentru orice altcineva și fiecare utilizator decide cât de multă încredere poate acorda certificatelor emise de alți utilizatori;
- De exemplu, dacă Alice are cheile publice pk_1, pk_2, pk_3 corespunzătoare lui C_1, C_2, C_3...
- ...iar Bob, care vrea să comunice cu Alice, are certificatele $\mathrm{cert}_{C_1 \to B}$, $\mathrm{cert}_{C_3 \to B}$ și $\mathrm{cert}_{C_4 \to B}$ pe care i le trimite lui Alice;
- Alice nu are pk_4 și nu poate verifica $\mathrm{cert}_{C_4 \to B}$; deci ca să accepte pk_B, Alice trebuie să decidă cât de multă încredere are în C_1 și C_3;

Modelul "web of trust"

- Aici oricine poate emite certificate pentru orice altcineva și fiecare utilizator decide cât de multă încredere poate acorda certificatelor emise de alți utilizatori;
- De exemplu, dacă Alice are cheile publice pk_1, pk_2, pk_3 corespunzătoare lui C_1, C_2, C_3...
- ...iar Bob, care vrea să comunice cu Alice, are certificatele $\mathrm{cert}_{C_1 \to B}$, $\mathrm{cert}_{C_3 \to B}$ și $\mathrm{cert}_{C_4 \to B}$ pe care i le trimite lui Alice;
- Alice nu are pk_4 și nu poate verifica $\mathrm{cert}_{C_4 \to B}$; deci ca să accepte pk_B, Alice trebuie să decidă cât de multă încredere are în C_1 și C_3;
- Modelul e atractiv pentru că nu necesită încredere într-o autoritate centrală;

Invalidarea certificatelor

- Atunci când un angajat părăsește o companie sau își pierde cheia secretă, certificatul lui trebuie invalidat;

Invalidarea certificatelor

- Atunci când un angajat părăsește o companie sau își pierde cheia secretă, certificatul lui trebuie invalidat;
- Există mai multe metode de invalidare între care:

Invalidarea certificatelor

- Atunci când un angajat părăsește o companie sau își pierde cheia secretă, certificatul lui trebuie invalidat;
- Există mai multe metode de invalidare între care:
- **Expirarea.** Se poate include data de expirare ca parte a unui certificat, care trebuie verificată împreună cu validitatea semnăturii;

Invalidarea certificatelor

- Atunci când un angajat părăsește o companie sau își pierde cheia secretă, certificatul lui trebuie invalidat;
- Există mai multe metode de invalidare între care:
- **Expirarea.** Se poate include data de expirare ca parte a unui certificat, care trebuie verificată împreună cu validitatea semnăturii;
- **Revocarea.** CA-ul poate, în mod explicit, revoca un certificat de îndată ce acesta nu mai poate fi folosit;

Invalidarea certificatelor

- Atunci când un angajat părăsește o companie sau își pierde cheia secretă, certificatul lui trebuie invalidat;
- Există mai multe metode de invalidare între care:
- **Expirarea.** Se poate include data de expirare ca parte a unui certificat, care trebuie verificată împreună cu validitatea semnăturii;
- **Revocarea.** CA-ul poate, în mod explicit, revoca un certificat de îndată ce acesta nu mai poate fi folosit;
- Aceasta se poate realiza prin includerea unui număr serial în certificat; la sfârșitul unei zile CA generează o listă de certificate revocate (care conține numerele seriale) pe care o distribuie sau publică.

Important de reținut!

- Semnături electronice
- Certificate digitale