

## Introducere în criptografia simetrică

- Criptografia = instr. fundam. pt protejarea comunicatiilor intr-un sistem.

→ nu poate asigura securitate fără alte instrumente ajutătoare (controlul accesului)

- Criptografia → protocole de securitate  
1) confidențialitate → canonicul

privat al inf între 2 sau > entități

2) integritate → integritatea inform. (inf care este alterată, modificată)

3) autenticitate → stabilirea identit. entităților intr-un sistem

4) nerepudiere - pt neștiința inf.

Criptografie simetrică - (cu chei private)

- asimetrică (cu chei publice)

• Parametri de securitate

→ P. de sec. = lungimea unei chei de cădere

sau: dimensiunea minimă a unei informații

pt a atinge securitatea la nivelul general

→ În criptografie parametru de sec. se notează

cu  $l^n$  (unul = param. de sec.  $n$ )

→  $n$  biti de 1 pt a reprezenta un număr  $n$

- Algoritmul utilizat = probabilitate  $\Rightarrow$

→ În ceea ce privește algoritmul poate face o decizie cu o probabilitate - chiar o număr probabil - într-oice stare să fi avut. este 1.

Ao.

$$\begin{array}{c} \xrightarrow{\Delta_i} \Delta_{i+1} \\ \xrightarrow{\Delta_i^K} \Delta_{i+1}^K \end{array} \quad \left. \begin{array}{l} P_{i+1}^1 \\ \vdots \\ P_{i+1}^K \end{array} \right\} \quad \sum_{j=1}^k P_{i+1}^j = 1$$

$A =$  algoritm

$A(x) = y$  dacă  $A$  este determinist și produce  $y$ , astfel căd are  $x$  la intrare.

$y \leftarrow A(x) \Leftrightarrow A$  pe intrarea  $x$  produce  $y$ , și  $A$  = probabilist.

Pe o intrare  $x$ ,  $A$  poate produce mai multe istorii fiecare cu o anumită probabilitate.

$x \leftarrow A$  ( $\Rightarrow x$  este selectat uniform random din mulțimea  $A$  (este cale echivalentă))

$A$  = finit:  $\Rightarrow$  probabilități cu care este realizat  $x$

$$\text{este } \frac{1}{|A|}$$

Comunicatia între 2 entități = părțile unui canal, care poate fi  $\{stare\}$  sau  $\{nec stare\}$ .

Dacă toate canalele au  $H$  diște  $\Rightarrow$  nu am mai avea nevoie de criptografie; anoj. canalelor = mesajele.

Canalele nesigure  $\rightarrow$  sunt supuse lo  $\begin{cases} \text{atac pasiv} \\ \text{atac activ} \end{cases}$

- atac pasiv: monitorizarea echipamentului, obț inform. care circulă în canal

atac activ = monitorizare + alterarea informației  
(inclusiv injectarea de noi informații)

Cauzaile restante sunt reprezentate de un adversar (atacator) = intrus

în fct de ceea ce, adversarul  $\begin{cases} \text{activ} \\ \text{pasiv} \end{cases}$

- adversarul poate fi  $\begin{cases} \text{intern} \\ \text{extern} \end{cases}$  și  $\begin{cases} \text{noi} \\ \text{călători} \end{cases}$

- în criptografie și - adversarul  $\alpha$  este modelat printr-un algoritm probabilist de complexitate polynomială (Time)  $\Rightarrow$  PPT.

Se poate avea acces la un oracol

oracol = un formalist general ce dezvoltă un algoritm, furnizând inform. Algoritmul conține acesta îl se cere.

Ex de întrebare: n este prim? (a consulta tot un oracol).

$O$  = oracol;  $\mathcal{A}^O$  = algoritmul  $\alpha$  care are posibilitatea de a consulta un oracol  $O$

Schemă de criptare simetrică

$\rightarrow$  3- uplu:  $\mathcal{G}(G, E, D)$  unde

$G$  = algoritm probabilist de chei de criptare  $K$

$K \leftarrow \mathcal{G}(1^n)$

generând  $t_0$  și  $t_1$  f. dependență  $\rightarrow t_0$  și  $t_1$  să fie că mai random.

2)  $E$  = algoritm probabilist de criptare care primește de la o cheie  $k$  și un mesaj  $m$  generația criptată

$c \leftarrow E(k, m)$  se mai notează  $c \leftarrow E_k(m)$

Criptare probabilistă ( $\rightarrow$  mesajul nu este același pentru că poate fi criptat diferit în fel de timpul în care este criptat -

3)  $D$  = algoritm determinist de decriptare, care primește  $k$ ,  $t_m$ ,  $t_c$ , cu  $c \leftarrow E_k(m)$  are loc  $D(k, c) = m$  (nu mai scris  $D_k(c) = m$ )

$$E_k(m) \xrightarrow{c} c' \geq D_k(c') = m$$

În criptografie, principiul fundamental stabilit de către Kerchoffs spune că :

- criptosistemul trebuie să fie public, și doar cheia de criptare să fie secretă - (chiar dacă e public, nu ar face securitate)
- ex: criptosistem A5/1  $\rightarrow$  fol. de GSM  $\rightarrow$  a fost decriptat de atacatori - (e foarte slab).

A5/2, A5/3 (mai bune).

- Păstreaza secretă  $\rightarrow$  incrustați fraude.

Modele de securitate -

Cum alegem în funcție un sistem de sec.

$\rightarrow$  Modele practice :

1) modelul COA (cipher text only attack)

⇒ atacatorul vede criptotextul și nu se

deducrește mesajul sau cheia -

2) KPA (known plain text attack)

⇒ adversarul de-a lungul timpului cunoaște

tot că  $m_1$  a fost criptat prin  $C_1$

$m_2$  a fost criptat prin  $C_2$

⇒ cercetă și găsește mesajul sau cheia.

Modele active

1) CPA - (chosen plaintext attack)

⇒ adversarul are probabilitatea să obțină criptotextul asociat unui plaintext ales de el  
(lunchtime attack)

2) CCA - chosen ciphertext attack -

adversarul are probabilitatea de a alege mesajele originale asociate unor criptotexte

(poate fi făcut și adoptiv  $\Rightarrow$  adu. Dacă poate alege el un criptotext și să întrebe care e meajul original corespunzător)

- CCA este cel mai puternic (primitivul nu poate fi rezistent la CCA)

⇒ fără nici un cas scheme de criptare la atacuri CPA pot fi transformate în scheme CCA sigure.

Criptosistemul simetric  $\rightarrow$  2 clase fundamentale

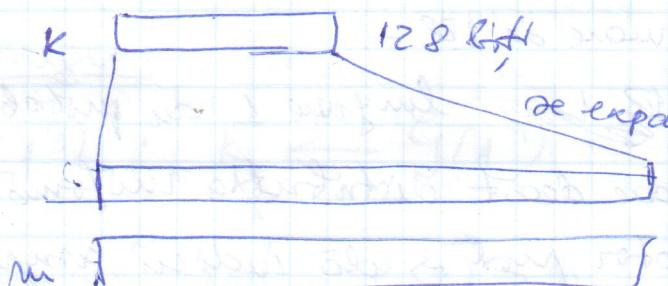
- sir (stream)

- bloc

- Criptost. stream (sir) criptografă un mesaj

$m = m_1 \dots m_l$  folosind o cheie - sir de aceeași lungime ca și mesajul.

$K = K_1 \dots K_l$ ; dacă cheia  $K$  este de obținut generată de un generator de chei primar de la o cheie și mică.



de expandarea cheia poate fi lungimea mesajului

Exemplu: RC4 (se utilizează în TLS)  $\rightarrow$  o-a an-

că este slab.

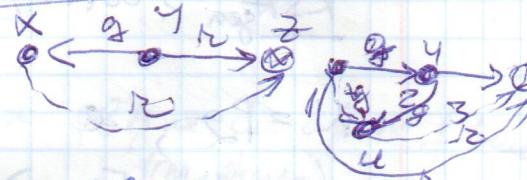
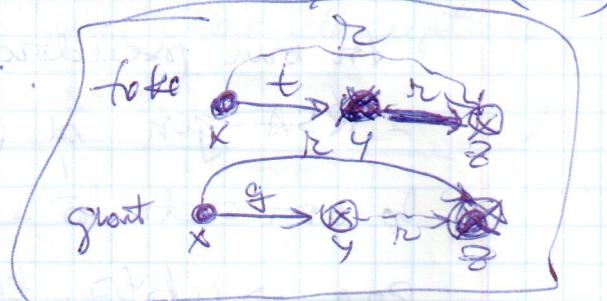
- Cu probabilitate mare pot fi prese decrifrante din cheie str., ex: al doilea octet din cheia să fie 00Hexa ( $0^8$ ) cu probabilitate  $1/128$ .

$$m = m_1 \dots m_l$$

$$K = K_1 \dots K_l$$

$$\text{XOR: } (m_i \oplus K_i) \dots (m_l \oplus K_l)$$

$$K_2 \rightarrow \text{cunoște } m_2$$

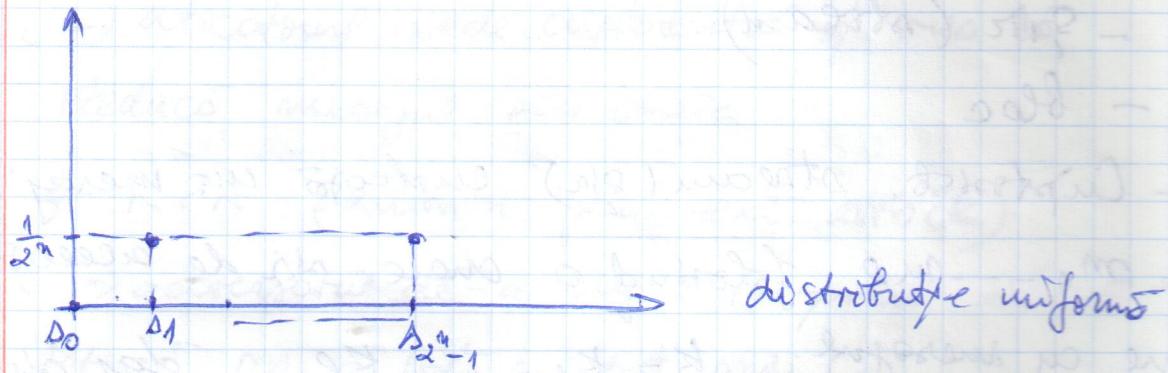


- Criptostemul A5/1. - acasă problema

îl își folosește E0 (pentru Bluetooth)

generatorul G îl face modul pseudorandom (PRNG)

Pseudorandom:  $\{0, 1\}^n$ ,  $2^n$  secuții



$$|K| = 128 \text{ bits}$$

$\rightarrow$  Vrei să generezi chei de lungime  $n$ ;  $n \geq 128$

$n$  e mult mai mare de 128 -

Dacă durele de ~~de la~~ lungime  $n$  au probabilitatea

mai mare decât distribuția uniformă

$\Rightarrow$  e un generator prost  $\rightarrow$  doar unul din aranjamente

cheii  $\Rightarrow$  dist. cheilor să fie de căt mai

aproape de ~~pe~~ distribuția uniformă (aceste  
pseudorandom)

$\rightarrow$  Minunii m-a demonstrat că și un generător  
nu poate fi pseudorandom.

- RSA-gen și BBS-gen sunt cele mai bune  
la ora actuală -

RSAgen  $\rightarrow$  utilizare exponentiere modulară (e lent)

$$N = p_2, e$$

$$x_0 \in [1, N]$$

$$x_1 = x_0^e \bmod N$$

$$x_2 = x_1^e \bmod N \cdot$$

dim fiecare  $x_i$ . Dacă se alege

$\ell(n)$  biti cei mai puțini semnificați  $\Rightarrow$  cei  $\ell(n)$  de generoși cheia și.

(plec de la  $x_0, e$ , generă  $x_1 \rightarrow$  iau cei mai puțin semnif. biti dim  $x_1$ , generă  $x_2$  și iau cei mai puțini semnif. biti dim  $x_2$   $\dots \rightarrow$  rezultă cheia și.)



$(x_1)_{\ell(n)}$   $(x_2)_{\ell(n)}$ .

Complexitatea RSA gen:  $(\log N)^3$

Schreier.

$$g(1^n) : k \leftarrow g(1^n)$$

$$\mathcal{E}(k, m) |m| = \ell(n) > n$$

$$G(k), \text{ săt } c = G(k) \oplus m$$

XOR

decriptare:  $D(k, c)$  se proced identic: se calculează  
 $m = G(k) \oplus c$ .

→ în cadrul criptoistemelor și, cheia de criptare este achimbată de la mesaj la mesaj -

$$m_1 \longrightarrow c_1 = m_1 \oplus G(k)$$

$$m_2 \longrightarrow c_2 = m_2 \oplus G(k)$$

$$c_1 \oplus c_2 = m_1 \oplus m_2$$

- ele sunt foarte rapide în practică
- A 2-a etapă de cript - cu cheie sau:

Criptoistem de bloc = mesajul m e împărțit în blocuri de mesaj și criptarea se face bloc cu bloc.

- Cine din schemele generale:

NP. Că avem o familie de funcții

$$(F_K)_{K \in \mathbb{K}} \quad F_K : \{0,1\}^m \rightarrow \{0,1\}^n$$

spațiu de chei:  $n = 64, 128, 192, 256$ .

Criptarea unui bloc se face astfel:

$$\rightarrow G(1^n) : k \xleftarrow{u} \mathbb{K} \text{ extrage o cheie}$$

$$\rightarrow E(k, m) : \text{se generează random}$$

$r \xleftarrow{u} \{0,1\}^n$  sau criptex-

$$\text{ful e de forma } c = (r, F_K(r) \oplus m)$$

Decriptare:  $D(k, c) : c = (r, s)$

$$m = s \oplus F_K(r)$$

Dacă familia de funcții  $F_K$  e pseudorandom

(PRF)  $\rightarrow$  se generează output către mai

cărăpare de random pur, atunci schema e

CPA sigură - (A) probabilistic.

$$m \rightarrow r_1 \quad (r_1, F_K(r_1) \oplus m)$$

$$r_2 \quad (r_2, F_K(r_2) \oplus m)$$

Pt familia de funcții  $F_K$ , putem considera

$$(DES_K)_{K \in \{0,1\}^{64}} \quad - \text{cheie de lungime 64.}$$

$$(AES_K)_{K \in \{0,1\}^{128 \text{ sau } 192 \dots}}$$

$$\text{sau } (3DES_K)_{K \in \{0,1\}^{128}} \text{ (cu 3 chei)}$$

$$3DES_{(K_1, K_2)}(x) = DES_{K_2}(DES_{K_1}^{-1}(DES_{K_1}(x)))$$

(e în temă) → tema 2.

- Moduri de criptare:

- ECB :  $m = m_1 \dots m_e$

(electronic code  
block)

$K \downarrow$

$$c_1 = F_K(m_1)$$

$$c_e = F_K(m_e)$$

$r \in \{0,1\}^w$  rotunjire,  $c_1 = F_K(m_1 \oplus r)$

$$c_e = F_K(m_e \oplus r)$$

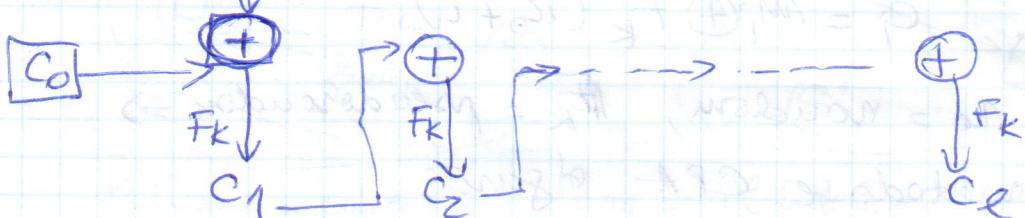
apoi de mai multe ori e criptat la fel  $\Rightarrow$  vulnerabilitate majoră.

În practică, acest mod nu trebuie utilizat.

### CBC (cipher block chaining).

$$m = m_1 \dots m_e$$

$c_0$  = random uniform



$$c_1 = F_K(m_1 \oplus c_0)$$

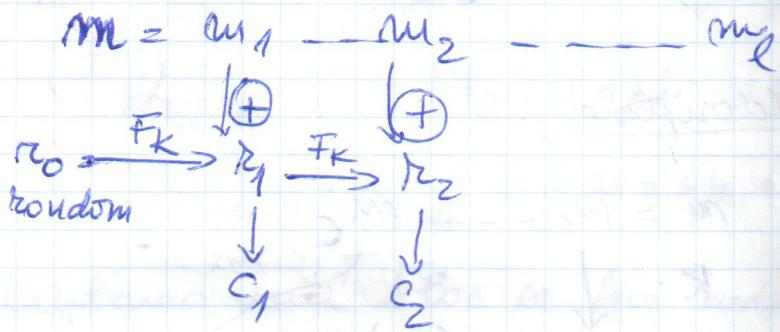
$$c_2 = F_K(m_2 \oplus c_1)$$

$$c = (c_0, c_1, \dots, c_e)$$

$c_0$  = vector de inițializare, ( $IV$ )

Dacă  $c_0$  = random și metoda de inițializare randomă, atunci  $\rightarrow CPA = \text{sigur}$

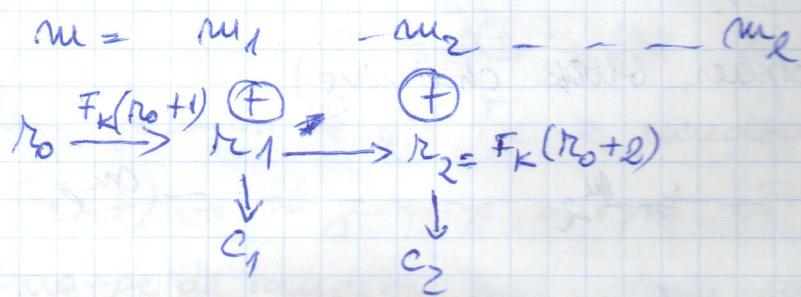
OFB → (output feedback)



$c_i = m_i \oplus F_k^i(r_0)$  funcție cripto-sistem OFB.

CPT sigur dacă  $r_0 = \text{random} \cdot \text{rf}$  metoda de initializare = random

CTR (counter).



$$c_i = m_i \oplus F_k(r_0 + i)$$

$r_0 = \text{random}$ ,  $F_k = \text{pseudorandom} \Rightarrow$  metoda e CPT sigur.

→ avem acces direct la blocul de criptext pt că nu depinde un bloc de blocurile anterioare.

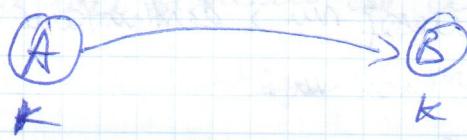
- CTR = teoretic mai eficient decât CBC și OFB.

$$C_i = F_K(M_i \oplus C_{i-1}) \quad C_{i-1} \sim M_{i-1}$$

Decrip  
tare

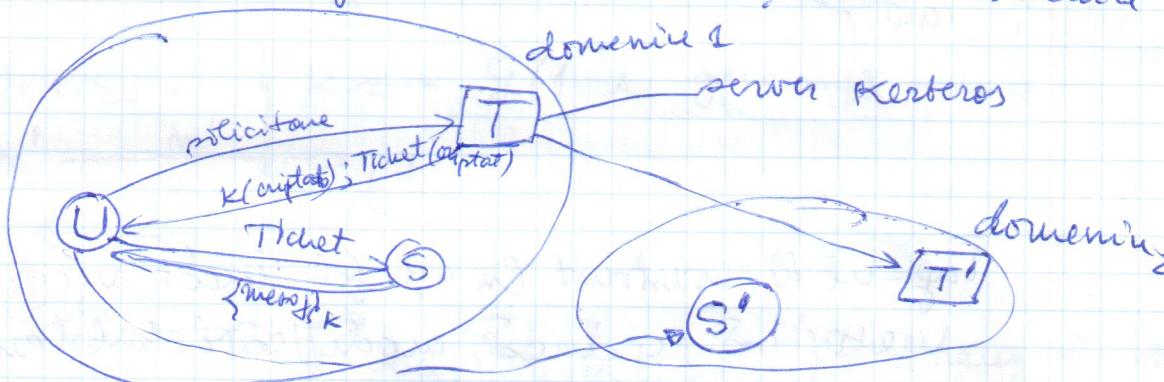
$$M_i = F_K^{-1}(C_i) \oplus C_{i-1}$$

Metodele CBC, OFB, CTR - nu sunt în formă  
actuală CCA sigură, pot fi modificate pt a  
fi forțată sigură.



problema distribuției cheii,  
cum aleg A și B să aibă acces  
la cheia de criptare?

- a luat noastră criptografie cu chei publice
- protocole de distribuție a cheii → KDC = protocole de distribuție a cheilor
- Kerberos = protocolul deserveste un domeniu



Ticketul = criptat ai doar S îl înțelege -

- Ticketul conține acestași cheie K pe care a primit-o pt u-

dacă există mai multe domenii → fiec dom. fiind  
deservit de către un server Kerberos, atunci accesul  
lui U spre S' se face prin ~~deservirea~~ Kerberos din dom.  
proprietatea utiliz. serviciile pentru serverul Kerberos  
din propriul domeniu.

Sâmbătă de la 10 - recuperarea curs - C112

- SHA-2 (256)
  - SHA-3 (512)
  - Whirlpool (512)
- Recomandare -
- MDS și SHA1 sunt slabă de tot, nu se recomandă  
a fi folosite.

Trebuie să fie totdeauna datele și autențificarea mesajelor

Ant. mes: un destinatord pot căuta dacă un este  
autentic expeditorului sau nu -

(entitatea A transmite un mesaj și destinatarul poate să  
stabilească dacă mesajul e autentic sau nu -

MAC = message authentication code -

-fie hash -

-semestru de gitale.

Obs: Autentic mesajelor nu înseamnă că cea ce este  
transmis - Nu putem să știm că mesajul este pe cale  
de a fi autentic.

Tag-ul trebuie construit în același fel ca să nu se modifică nici un  
mesaj și să producă modificări substantiale în TAG

← O schemă de autenticare a mesajelor e un triplet

$\mathcal{P} = (\mathcal{G}, \text{Mac}, \mathcal{V})$  unde  $\mathcal{G}$  = generator de cheie, algoritmul probabilist polytime.  $K \leftarrow \mathcal{G}(1^n)$

$\text{Mac}$  = un algoritm probabilist de complexitate polinomială care primește de la o cheie  $K$  și un mesaj și  
generează un mesaj de autenticare.

$$t \leftarrow \text{Mac}_K(K, m); \text{ mac}_K \leftarrow \text{Mac}_K(K)$$

## Integritatea datelor și autenticitatea mesajelor

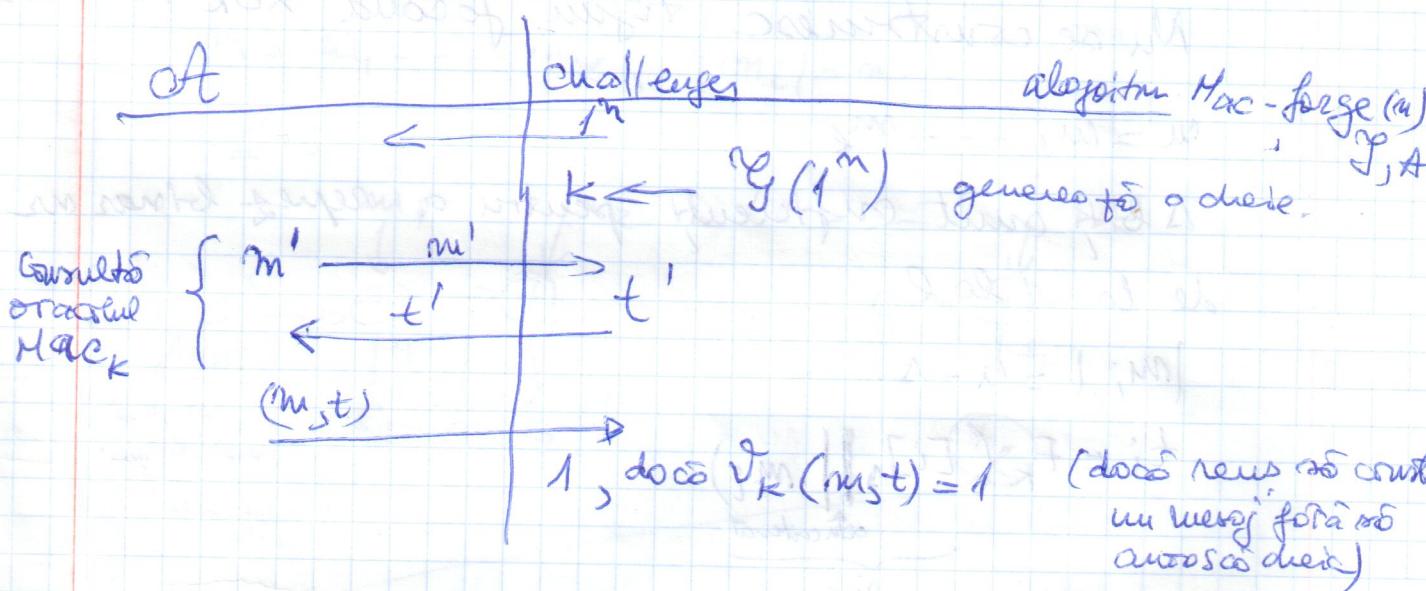
D = algoritm determinist polytime cu proprietatea  $f_K$ ,  $m$  = mesaj,  $\pi \leftarrow \text{Mac}_K(m)$  are loc  $\mathcal{V}_K(K, m, \pi) = 1$ .

$$(\mathcal{V}_K(m, \pi))$$

Mac - determinist;  $\text{Mac}: K \times \{0,1\}^* \rightarrow \{0,1\}^*$   
 atunci schema de autenticitate a mesajului = cod de autenticitate a mesajelor (notat MAC)

- un MAC trebuie să fie rezistent la falsificarea tag-ului.

At = adversary, Ch = challenger.



Rezistență la falsificare:

$P(\text{Mac-forge}_{f_K}(n) = 1)$  neigrijabilă (trebuie să fie)

Tehnici de construcție:

- Considerăm  $(f_K)_{K \in K}$  funcție pseudorandom (PRF)

(adică  $f_K = \text{DES}_K, 3\text{DES}_K, \text{AES}_K$ ).

$|m_i| = n$  (1 bloc)

$$t = F_K(m)$$

Metoda este sigură  $(m, t)$

$$m = m_1 \dashv\dashv m_e, |m_i| = n$$

$$t_i = F_K(m_i)$$

$$t = t_1 \oplus \dots \oplus t_e$$

metoda nu e sigură:

$$m = m_1 \text{ } m_2 \Rightarrow t = t_1 \oplus t_2 = 0^n$$

$$m' = m_1 \text{ } m_2 \text{ } m_1 \text{ } m_2, t' = 0^n \text{ tot valid.}$$

$$m_1, m_2 \rightarrow t \text{ (afac)} \Rightarrow$$

$m_2, m_1$  are același tag (XOR e comutativ)

Nu se construiesc taguri folosind XOR pe blocuri

$$m = m_1 \dashv\dashv m_e$$

3 biti sunt suficiente pentru a reprez binar nr de lo 1 lo 2,

$$|m_i| = n - 1.$$

$$t_i = F_K([i]_S \parallel \underbrace{m_i}_{\text{concatenat}})$$

$t = t_1 \oplus \dots \oplus t_e$  ca putem fi siguri, dar nu e sigur

$$m_1, m_2, \bar{m}_1, \bar{m}_2$$

$$t_1 = \text{tag pt } m_1, m_2$$

$$t_2 = \text{tag pt } m_1, \bar{m}_2$$

$$t_3 = \text{tag pt } \bar{m}_1, m_2$$

$$t_1 \oplus t_2 \oplus t_3 = \text{tag pt } \bar{m}_1, \bar{m}_2 \text{ deci nu e sigur}$$

$$M = M_1 \dots M_\ell$$

$1, \dots, \ell \quad \frac{m}{4}$  bits ( $m = \text{multiplo de } 4$ )

$$M_i \in \{0, 1\}^{\frac{m}{4}}$$

$$r \leftarrow \begin{cases} \text{random} \\ \text{uniform} \end{cases} \{0, 1\}^{\frac{m}{4}} \text{ (msg bit)}$$

$$\text{se construieste } t_i = F_K(r || l || i || m_i)$$

random  $\xrightarrow{\text{eg in locuri alegi}} \text{index message bloc}$

$t = (r, t_1, \dots, t_\ell)$  mesajul este astfel -

toatele sunt lung decat mesajul.

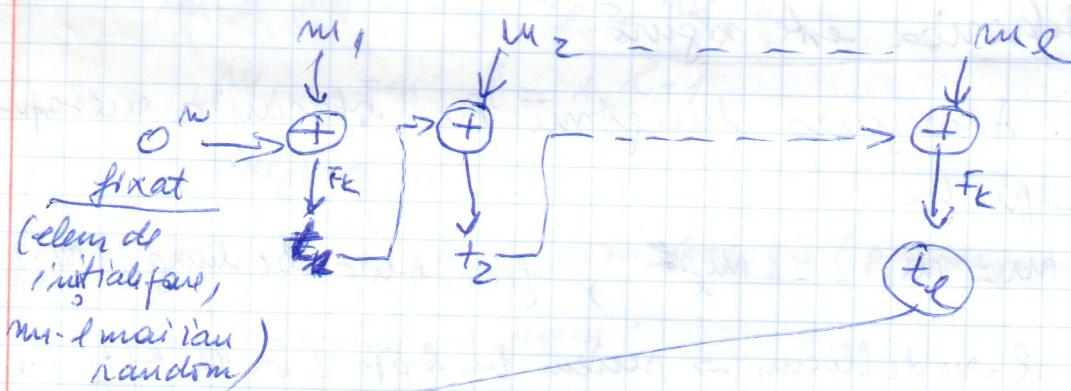
daca  $m = \text{mare} \Rightarrow$  toatele  $t$  sunt mari.

Reducerea dimensiunii toatalui  $\rightarrow$  tehnica standard răsărită

$\rightarrow$  CBC-MAC

$$M = M_1 \dots M_\ell, |M_i| = m$$

$$K \in \mathbb{F}$$



$$\underline{\text{Mac}_K(m) = t_\ell}$$

Aceasta este o construcție doar dacă  $F_K$  este PRF.

- care să utilizeze numai pt mesaje de aceeași lungime!

CBC-MAC pt mesaje de lg variabilă

$$K \in \mathbb{F}$$

$$\text{Mac}_K(m) = F_K(0 \oplus m) = t.$$

$$m' = \underbrace{m}_{\text{concatenat}} (\oplus t)$$

$$t_1 = F_k(m \oplus 0) = t$$

$$t_2 = F_k(\underbrace{t_1 \oplus m}_{t} \oplus t) = F_k(m) = t$$

$\text{Mac}_k(m') = t = \text{Mac}_k(m)$  și nu se poate fi tagul.

m se utilizează CBC standard pt msg de lungimi diferențiate

Soluții pt CBC MAC cu msg de lungime variabilă

① Utilizare 2 chei,  $K, K_1$

mesajul  $m = \text{dot}$ ; se construiește CBC MAC pt  $m$  cu cheia  $k$ ,

fie același  $t$ ,

- aplic  $F_{K_1}(t) = t'$  → mac-ul.

- Atacul anterior nu mai funcționează.

→ Tehnica este sigură

② Adaugarea lungimii în blocuri a mesajului ca prefix.

$m = m_1 \dots m_l$ , fără blocuri lung. ( $m_i \neq m$ )

$l = m$  de blocuri → scris în baza  $\geq m$  de biti → se

concatenează la mijloc →

$[l]_m m_1 \dots m_l = m'$  → construiește CBC MAC

pt  $m'$ .

l poate fi scris pe n biti?

$n > 80$  bits  $\Rightarrow l < 2^{80}$

⇒ în modul că nu utilizăm blocuri de mesaj de  $2^{80}$ .

Metoda e signură, deci

Nu puteti  $[E]_2$  (l imposibil) sa coada mesajul  
lui (exemplu de exercitiu pt examen)

$m_1, m_2, m_3$  - se sau nu este anotă că se poate  
 $m_1, \underline{m_1 m_3}$  falsifică toate)

③ Soluția băsează pe cheia în față de lungimi neegale:

$$m = m_1 \dots m_l$$

$\Rightarrow$  - calculă  $k_l = F_K(l)$ , l este numărul blocurilor

$k_l$  = construiesc CBC-MAC.

- și ac metoda e sigură.

• ④ C-MAC (standard NIST)

utilizează 3 chei:  $k_1, k_2, k_3$  și ca în calcul punctul de  
padare a textului pt ce toate blocați să fie de același  
lungime.

$$m = m_1 \dots m_{l-1} m_l$$

$$(m_i) = m_i \quad i = 1, l-1$$

$$|m_l| \leq m;$$

Se ia  $m_l$  și se completează (padat) cu zerouri

atât la început, cât și la sfârșit;

$$\underbrace{m_l 0 \dots 0}_n \oplus k_1 = m'_l \rightarrow \text{daca } |m'_l| < m$$

$$\text{nu fml } \oplus k_2 = m'_l \text{ adică } fud = m$$

construiesc  $m' = m_1 \dots m_{l-1} m'_l$  și aplic CBC-MAC standard

Metoda e sigură

- pt fermecile 1, 2, 3 - padarea e recomandată a se face  
numai 1 următor de ori cu zero-uri e nevoie.

$m$ , mesaj;  $\bar{m}$  = un padot

$$\underline{m' \neq m'' \Rightarrow \bar{m}' \neq \bar{m}''}$$

- MAC -

- proprietatea de securitate

- construcție pt MAC pe blocuri, construcția CBC MAC și CBC MAC cu msg de lungime variabilă -

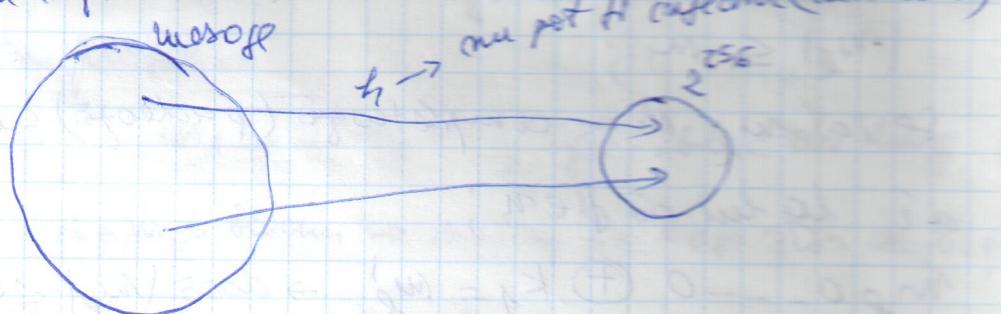
Integritatea datelor. - se asigură că principalul nume funcții hash. - pleacând de la un mesaj  $m$ ,

$$m \xrightarrow{h} h(m)$$

rezultat  
refunat

(message digest).  
care înseamnă  
de dimensiuni mici

în caz are prop. că modif. funcții în mesaj  
conduc în mod probabil la modificarea mai  
în refunat.



$$\exists m_1 \neq m_2 \text{ cu } h(m_1) = h(m_2)$$

fct hash - construită dintr-un generator de chei  $G$  și o  
fct  $H$ , unde  $G$  = algoritm probabilistic forte  
- fct  $H$  :  $K \times D \longrightarrow R$

dom de mesaje  
spațiu de  
refunat,  
f0, f1, f2\*

cu proprietatea: dacă  $K = \text{cheie}$   
 $m = \text{mesaj} ; |H(K, m)| < |m|$

fătă  $H$  trebuie să fie calculabilă în timp polinomial determinist.

$H(K, m) \rightarrow$  dimensiune ref. < dimensiune mesajului de intrare  
 dacă la  $\text{MAC}_K$  - se aplică un criptosistem  $F_K : \{0,1\}^n \rightarrow$   
 $\rightarrow \{0,1\}^n$ , căfătă Hash arătam

$$H_K : \{0,1\}^n \rightarrow \{0,1\}^s, \text{ cu } s \ll n$$

Propri. de securitate pt făt hash

① Resistență la coliziuni.

Ch:  $K \leftarrow \mathcal{G}(1^n)$  challengerul gener. o cheie pe care  
 o dă adversarului

$\text{ct}(K) \rightarrow$  constr. 2 mesaje  $m_0, m_1$ ,

adversarul către făt dacă poate construi o colizie.

$$m_0 \neq m_1, \text{ și } H_K(m_0) = H_K(m_1) \Leftrightarrow \text{colizie}$$

• Făt = rezistent la coliziuni dacă probabilitatea aduers.

de a det. coliziuni e neglijabilă. (nu e posibil ca  $m_0$  și  $m_1$  să fie aceeași).

$(R_2)$  = collision Resist (2 urme de la cele 2 mesaje)

② Resistență la cel de al doilea algoritm  
 second pre-image attack. (rezistență slabă  
 la coliziuni).

challenger = ch.

$$\text{ch} : K \leftarrow \mathcal{G}(1^n)$$

$$m_0 \leftarrow D$$

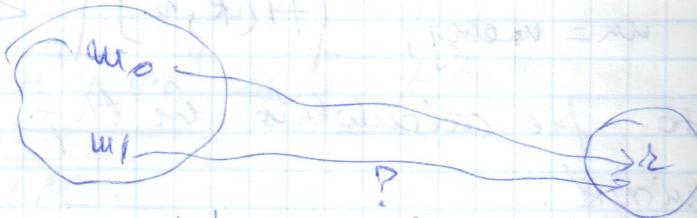
extrage un  
 mesaj  $m_0$

$$\text{ct}(K, m_0) \quad m_1$$

lă dobătăresc.

$$m_1 \neq m_0 \text{ și } H_K(m_0) = H_K(m_1) \text{ una din }\text{ componentele e fixată de challenger} \rightarrow \text{to be so's. Colizie}$$

fiare -



(Impostură) = o secună cu structura difuză  
dar cu același rezultat hash.

H - este rezistentă la 2-năt prelucrările doar  
probabilitatea adversarului de a găsi un e neglijabilă

③ Rezistență la prelucrare (OW = one way)  
(sau CR)

Challenge: generoș k  $\leftarrow \mathcal{G}(1^n)$

$$m_0 \leftarrow D$$

$$y = H_k(m_0)$$

dacă adversarul încearcă  $A(k, y)$ , adică trece prin

găsește un  $m_1$  cu  $H_k(m_1) = y$ .

H - rezistență la atac de prelucrare doar  
probabilitatea de a calcula  $m_1$  e neglijabilă

CR - schema e one-way  $\rightarrow$  pe calculă nu se  
pot calcula invocațiile, iar greu să se pot -

$\boxed{\text{CR2}} \Rightarrow \text{CR1} \Rightarrow \text{CR}$  și următoarele lucru - început  
(ceea mai față) - și cea următoare pt fct hash.

- Tehnici de construcție pt fct hash -

$\rightarrow$  Se construiește fct hash care reduc un surse mic de biti,  
după care acesta se iterează -

$$H_K : \{0,1\}^n \rightarrow \{0,1\}^m, m > n$$

$h = \text{fct de iteratie} \rightarrow H$

$h_k = \text{fct de compresie}$   
(fct harsă de compresie)

Iteratia să să aibă urm. proprietate;

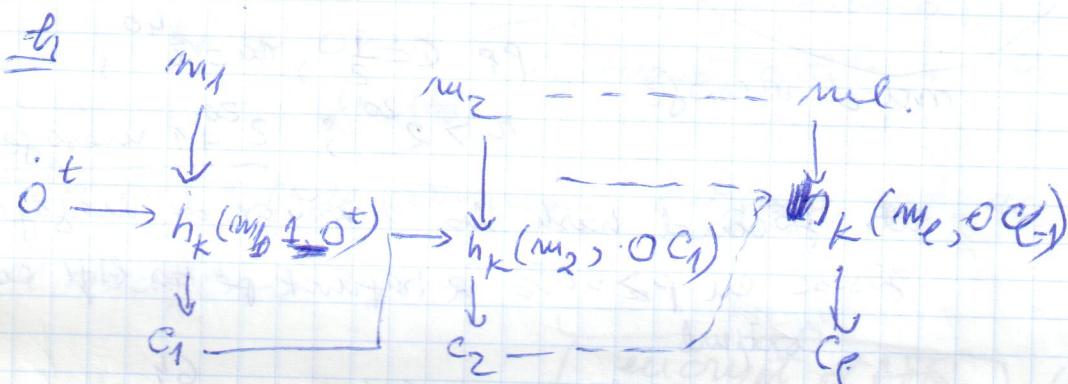
- dacă  $h$  e CR2 algor, atunci și  $H$  are să fie CR2 algor (e fundamental).
- Una din cele mai cunosc. tehnici ale iteratiei.

MD (Merkle Damgård)

(nu are leg. cu MD5 = message digest ~~is safe~~)

~~MD6~~ → Rivest

Standartul actual e SHA3 - care are valență a fct (Keccak) -



$$H_K(m) = c_r, \quad h_k : \{0,1\}^n \rightarrow \{0,1\}^s.$$

$$t \rightarrow n-s-1 \quad (n-s \geq 2)$$

$$\text{diferit} \quad (n-s=1)$$

MD poștează proprietatea CR2.

• ac. iteratie e folosită în MD5 și SHA1

alte tehnici: sponge; Paragon -

- Atac fundamental = atacul jocului de măstere.

M = proprietăți

R obiecte, - fiecare având 1 proprietate,

→ dacă ~~M <= R~~, C = constanță arbitrală astfel

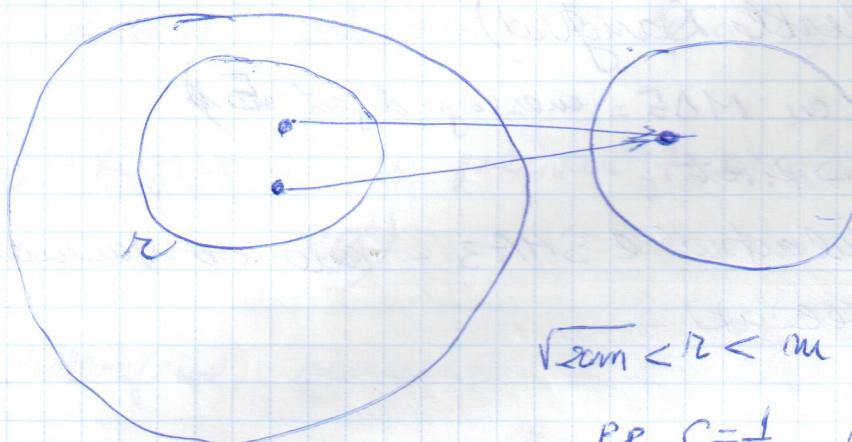
$$\sqrt{2cm} < R < m, \quad \text{atunci cu probabilitate}$$

mai mare de  $\frac{1}{2}$ .  $\exists r_1 \neq r_2$  cu acelasi proprietate.

ex:  $m = 365$  și,

proprietate: nu scut din flux și a anumitor doar  $r = 23$  (nr de elevi dintr-o clasă)

atunci cu  $p > \frac{1}{2}$ , și copiii au același probabilitate de a fi scuti.



$$\text{nr refuzate, } m = 2^{256}$$

$$\sqrt{2m} < r < m$$

nr de mesaje

$$PP C = \frac{1}{2}, m = 2^{40}$$

$$r > 2^{20}; 2^{20} \text{ mesaje}$$

at. folosind hash la transmiterea de mesaje, găsește cu  $p \geq 50\%$  și refuze pe toți bitii cu același conținut.

→ de aici se aleg 128 biti →  $2^{64}$  mesaje din MD5.

(MD5 - o din cele mai bune funcții de generație de criptografie și este slabă)

la SHA1 → 160 biti → teoretic nu îl poate atinge niciun alt hash de mesaje; dar și SHA1 nu e rezistent la coliziuni.

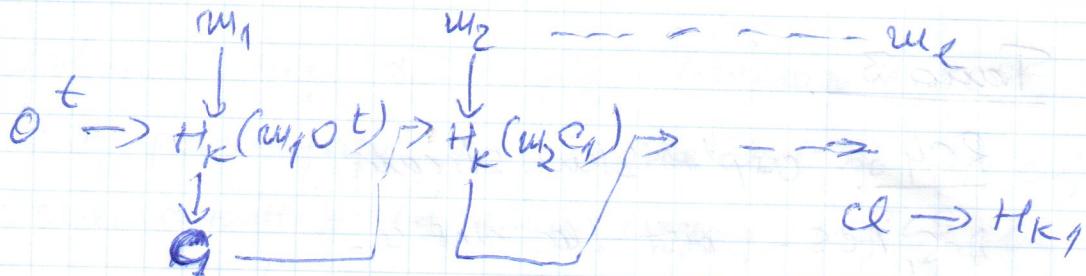
SHA2 și SHA3 pot fi utilizate.

### Construcția de MAC cu funcții HASH

NMAC - pt HMAC

↓  
Nested MAC

de la  $H_K$  - fct hash cu cheie; un vector de lungime fixă de la  $O^t$



$$NMAC = \underbrace{H_{K_1}(H_K(m))}_{\text{iterata MerkleDamgard}}$$

HMAC = NMAC în care cheile  $K \neq K_1$  sunt XOR cu ipad (innerpad)

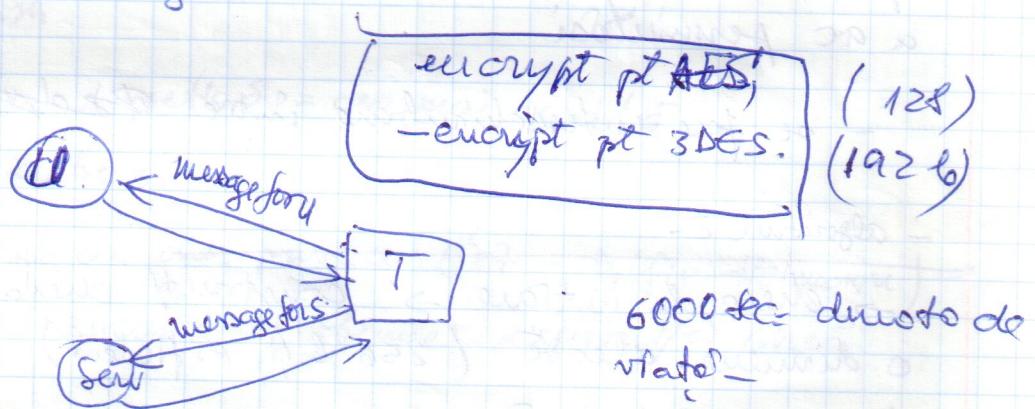
$\downarrow$  pe urmă de părante fixate, un octet cauz.  
acele 4 octeți 0, 4 octeți 1.

$$K \oplus \underline{\text{ipad}}$$

fixat

$K_1 \oplus \underline{\text{opad}}$ . — părante extinse  
fixat (inversează celelalte 4 octeți din  $K_1$ )

- Dacă folosește compresie și sigură, atunci ac. metoda de iterare este sigură.



~~Serv~~ Serv: proces request. verif. doceș  
cheile și userii coresp.)  $\rightarrow U_1 = U_2 \wedge L_1 = L_2$   
User: înregistrează user, înregistrează serviciu

afis, mouse, timestamp,

### Teme 3.

RC4. criptosistem stream

des, aes - o săt bloc clasic -

intrare  $\rightarrow$  cheie K

ieșire  $\rightarrow$  cheie stream KS

criptotext = plaintext  $\oplus$  KS

plaintext = criptotext  $\oplus$  KS

pot negați KS apoi ca să nu am delay.

$\rightarrow$  negătm cheia byte cu byte, pe măsură ce apar textul -

RC4 are 2 pasuri

- KSA
- PRGA.

- în I pas - se generează permutarea identică

$D = i$

apoi în făc de cheia de intrare se face o reordinare a ac permutării

$\rightarrow$  se face interzchimbare între ~~PSR~~ -

- algoritmul 2 -

$\rightarrow$  cheia de intrare  $\rightarrow$  construiește cheia stream pe o dimensiune, alesă (256 biti suficient)

bonus: statistică -

- anumiti bytes au o destul de frecvență  
 $\rightarrow$  să băiem în sistem, pt că XOR une schimbă criptotextul pe bitii săi, în mod n. fără

$\rightarrow$  se folosește RC4  $\rightarrow$

$\rightarrow$  cel mai dur: avem KS  $\rightarrow$  găsim cheia

alte variante;

- parte din bonus se referă la pct b din semestr

→ folosirea bătăii de o pt a compromite RC4 -

- se poate folosi ca un plăințător să fie criptat cu cheie diferență - (f aminte împrej; ex TCS în care un atacator poate obține criptarea unui același text cu KS diferență)

de născut material Google pt atac RC4 →

→ găsim algoritmul pe net - cum se atacă RC4 -

- altă variantă de atac: - găsește cheie,  
- atac: FMS - de ex - găsește criptotext initial.

Hmp: 3 săpt → de astăzi înainte -

teme 2 și mai puține peste 2 săpt pt atât.

\* ~~Proiect~~

Rec:

- MAC -

CBC-MAC  
N-MAC  
HMAC

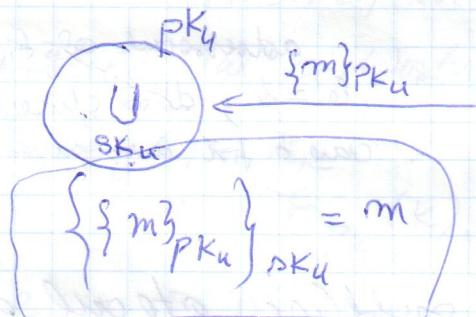
fct hash.

Criptografia simetrică = cea mai rapidă -

KDC - kerberos

Criptografie cu chei publice (77 - Diffie-Hellman)

→ rezolvă problema problemelor distribuției cheii



Def. Un criptosistem cu chei publice PKE e

un triplet  $\mathcal{G} = (G, E, D)$

$G$  = algoritm probabilist de tip polinomial,  $G(t^n) \rightarrow$  ge

generatoare de cheie (PK, SK)

cheie publică cheie privată  
publică (secretă)

$E(PK, m)$  generatoare probabilist un criptotext c

$$C \leftarrow E_{PK}(m)$$

Criptarea se face probabilist.

$D(SK, C)$  = algoritm determinist cu proprietate de deschiere

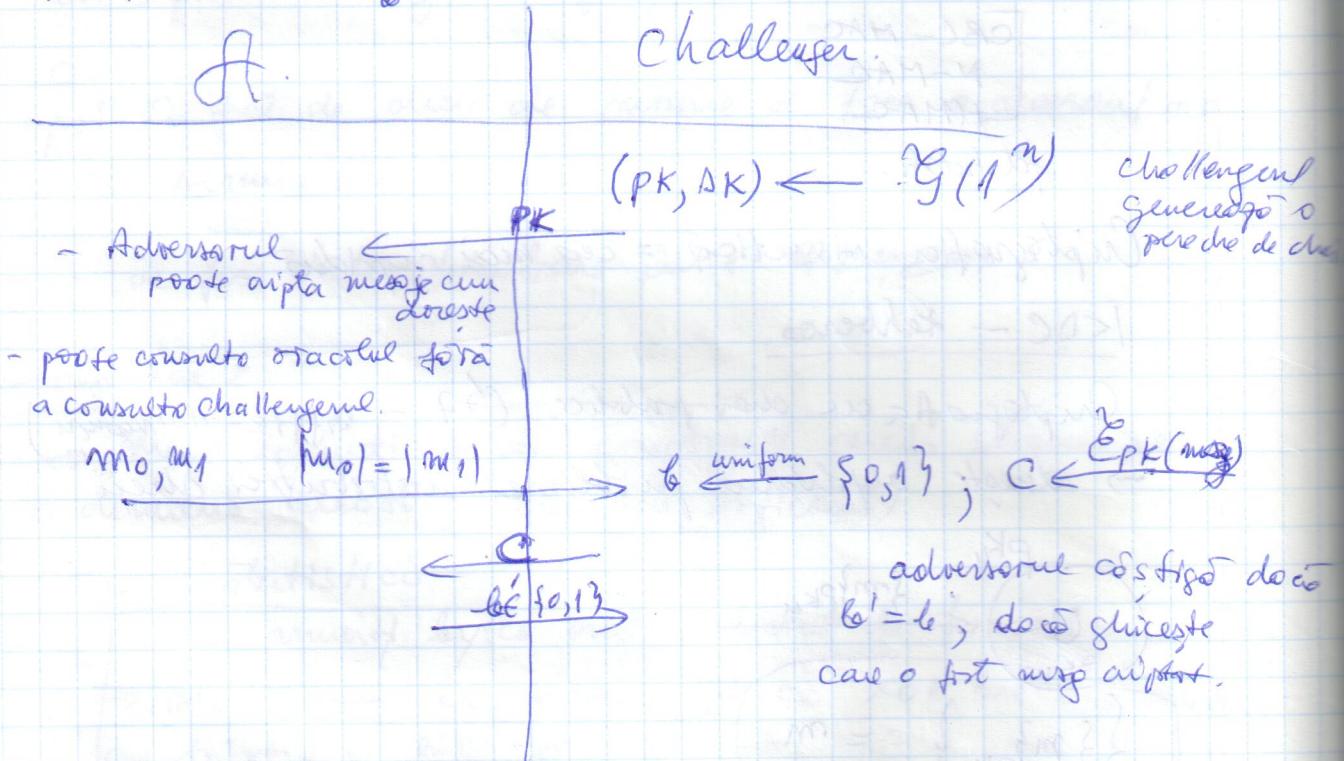
$$c \leftarrow D_{SK}(m) \text{ astfel că } m = D(SK, C)$$

Notiuni de securitate

IND-CPA - reziste atac de pluriștext aleator

IND-CCA - reziste atac de criptotext aleator

pt IND-CPA -



În criptografie cu cheie publică, atacul de pluriștext aleator nu poate fi evitat.

IND CPA este sigur doar probabilitate adversarului de a ghici ~~parola~~ din cele 2 mesaje și cel criptat este neglijabilă.

- un același mesaj criptat de mai multe ori produce rezultate diferite ( $\Rightarrow$  schema este probabilistică)

Schemele deterministe nu sunt sigure la atac de plaintext ales, (IND-CPA)

- RSA e deterministic  $\Rightarrow$  nu e sigur la atac de plaintext ales.
- Schema se face sigur printr-o construcție hibridă - (frecvent utilizată)

Se consideră o schema  $\mathcal{G} = (\mathcal{G}, \mathcal{E}, \mathcal{D})$  cu cheie publică

și o schema  $\mathcal{G}' = (\mathcal{G}', \mathcal{E}', \mathcal{D}')$  cu cheie secreta

Construcția hibridă:  $\mathcal{G}'' = (\mathcal{G}'', \mathcal{E}'', \mathcal{D}'')$

$\mathcal{G}'' = \mathcal{G}$  (se extrage o cheie  $P_K, S_K$  din  $\mathcal{G}(1^n)$ )

- se extrage o cheie  $K$  pt criptosist simetric,  
 $K \leftarrow \{0,1\}^m$

pt a se cripta mesajul  $m$ ,  $\mathcal{E}''(P_K, m)$  se procedă astfel: se criptează msg cu  $K$  (privat) pt criptosist simetric, iar cheia privată se criptează cu cheia publică, iar criptotextul este  $C = (C_1, C_2)$

$$\mathcal{E}'' C_1 = \mathcal{E}_K(m)$$

$$C_2 = \mathcal{E}_{P_K}(K).$$

RSA + 3DES

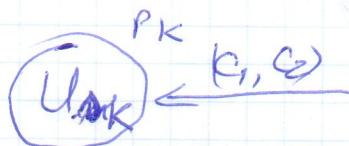
$$\xrightarrow{K}$$

3DES<sub>K</sub>(m)

$$\xrightarrow{C_1}$$

$\rightarrow$  RSA<sub>P\_K</sub>(K)

$$\xrightarrow{C_2}$$



pt parțea mare de mesaj se criptă cu cheie simetrică, pt cō e f. rapidă  
 (cript. cu cheie publică e de 1000 de ori mai lentă) → criptat doar punctul mai - pt viteză  
 dacă schema ~~G~~ este IND-CPA sigură,  
 iar schema ~~G'~~ este IND-CPA sigură,  
 schema hibridă e tot IND-CPA sigură ( $G^H$ )

### Construcții de criptare cu cheie publică

- Merkle-Hellman → criptosistem baza pe problema mulțimii -  
     -  $\exists$  un atac se folos. algoritmul UU → rezolvă prob. mulțimii
- Rivest-Shamir-Adleman → RSA -  
     → cel mai utilizat actual -  
     → problema factorizării

$P, q$  - nr prime mari, distincte.

$$N = Pq$$

$$e \in \mathbb{Z}_{\phi(N)}^* - ((e, \phi(N)) = \text{coprime} = 1)$$

$$\Rightarrow \exists e^{-1} \bmod \phi(N) = d.$$

$$\phi(N) = (P-1)(Q-1) \text{ fct Euler}$$

$$PK = (N, e)$$

$$OK = (P, Q, d) \quad P, Q \text{ nu pot fi publice.}$$

$$m \in \mathbb{N}, \quad c = m^e \bmod N$$

$$d = c^d \bmod N$$

$$c^d \bmod N = m^{ed} \bmod N = m \bmod N$$

$$ed \equiv 1 \pmod{\phi(n)}$$

RSA este probabilist  $\rightarrow$  nu este IND CPA sigur

- Se anunta ca tot. pe factorizare:

- adversarul cunoaste doar  $N \neq e$ , nu poate calcula  $m^e \bmod d$  pt că nu poate calcula  $\phi(N)$  ușor  $\rightarrow$  (nu poate factoriza lui  $N$ )

dacă  $m = m^e$

$$m < \sqrt[3]{N}$$

$$e=3 \rightarrow C = m^3 \bmod N = m^3$$

$$C = m^3 \Rightarrow m = \sqrt[3]{C}$$

pp că pot creați  $m$  cu 3 acii distințe;

$$(N_1, e_1), (N_2, e_2), (N_3, e_3) \text{ cu } e_1 = e_2 = e_3 = 3$$

$$C_1 = m^3 \bmod N_1$$

$$C_2 = m^3 \bmod N_2$$

$$C_3 = m^3 \bmod N_3$$

Conform TCR  $\rightarrow$  există un unic  $\bar{C} \bmod N_1 N_2 N_3$  ce verifică ecuația anterioră -

$$\bar{C} = m^3 \bmod N_1 N_2 N_3$$

$$\text{două } m < \sqrt[3]{N_1 N_2 N_3} \Rightarrow \bar{C} = m^3 \Rightarrow m = \sqrt[3]{\bar{C}}$$

Cum folosim totul RSA - ul. pt a fi sigur?

- Padded RSA -

$\xrightarrow{1} (N, e) \quad (N, d)$  se generează cheia publică și  
cheia privată

$\|N\| = \text{lung. reprez binară a lui } N$

$\ell(n) = \text{lungimea maximă a mesajelor a pt fi criptate}$   
 $m \in \{0,1\}^{\ell(n)}$

se generează random  $r \leftarrow \underbrace{\{0,1\}}_{\geq 8 \text{ octetii} \neq 0}$   $\|N\| - \ell(n) - 1$

Criptextul:  $c = \underbrace{(r \| m)}_N^{\ell} \text{ mod } N$ .

se pun în foto niște biti random

d. i. reprezentarea astfel pe  $\frac{1024-1}{\text{biti}} = 1023$

(pot cripta mesaje pe 1023 de biti)

$\rightarrow$  pot cripta pe 800 biti  $\rightarrow 223$  d. i. los pt random

$\Rightarrow$  criptarea devine probabilistică -

lungimea  $r \| m$  este  $\|N\| - 1$

$\rightarrow r \| m = c^d \text{ mod } N \rightarrow$  extragem ultimii

$\ell(n)$  biti mai puțini semnificativi biti  $\rightarrow$  extragem  $m$

### Standardul PKCS #1

• Public Key Cript Standard RSA

- plecăm de la  $N$ ,

- avem mesajul  $m$ , a cărui lungime  $|m| \leq \ell - 10$ ,  
unde  $\ell = \text{lungimea max pt criptare}$  - ( $\ell = \|N\| - 1$ )  
de ex.

decō  $N$  e pe 1024 biti,

$m$  poate reprezenta pe 1023, dar e costisitor să  
lucră la nivel de byte.  $\Rightarrow$  se lucrează pe octeți

Se părăsojă  $m$  cu 02 Hexa

$02 \| r \| 00 \| m$

$\geq 8 \text{ octeti} \neq 0$ , a și tot mesajul nu ar trebui să aibă lungime

~~•~~ r este generat random

→ mesajul de criptatofă e pe l biti,

mesaj original e pe cel mult l-10 (în octeți)

- Primul octet e 02 = m de ordin pt criptare RSA

001 = pt demnătura RSA -

- decifrare → vede 02 → mesaj criptat.

RSA → e la vers 2.2 acum.

Vers 1.5 mai punea un octet ovindecă lui 02 ⇒

⇒ multe atacuri -

PKCS#1 cu V2.2 = implementarea eea mai bună (sigură)

V2.2 se presupune că e IND-CPA (dor nu o-a demonstra)

---

### Criptosistemul El-Gamal

• folosește o const de tipul:

$G$  = grup finit, ciclic, de ordin  $q$ .

$\exists$  un generator  $g \in G$ ;  $G = \{g^0, g^1, \dots, g^{q-1}\}$   
(căs generață toate elem) (coful al mai utilizat e  $\mathbb{Z}_p^*$ ,  $p$  prim)

$PK = (G, q, g, g^x)$ , unde  $x \in \{0, 1, \dots, q-1\}$

~~•~~ SK =  $(G, q, g, x)$  - cheia secretă.

$g^x \xrightarrow{\quad} x = ?$  e o problemă întractabilă

$x = \log_g g^x \rightarrow$  problema logaritmului discret

Am ce face criptarea

$m$  - mesaj;

$E_{PK}(m) = ?$

generăm random  $y \leftarrow \{0, \dots, q-1\}$

$$c = (g^y, m \cdot g^{xy})$$

Dim  $m \cdot g^{xy}$  și  $g^y$  m potem calcula usor m

$$g^x, g^y \xrightarrow{?} g^{xy} \rightarrow \text{Problema DH Diffe-Hellman -}$$

→ Criptarea este probabilistica.

Pt decifrare -

$$\text{Dec}_K(c) : c = (g^y, m \cdot g^{xy})$$

$$\circ K = (G, g, y, x)$$

$$g^y \xrightarrow{x} (g^y)^x = g^{xy}$$

$$m \cdot g^{xy} /: g^{xy} \rightarrow \boxed{m}$$

Teorema. Dacă PLD (nuțe log discrete) este intractabilă, atunci criptosist El Gamal este IND CPA sigur.

In practică, El Gamal nu se utilizează ca;

- In practică;

$$G = \mathbb{Z}_p^*, p = \text{prim}, p = 2q + 1, q = \text{prim}$$

Mesajele se iau din multa  $\{1, \dots, 2\}$ ,  $2 = \frac{p-1}{2}$ .

$$m \in \{1, \dots, 2\}$$

$$m' = \underline{m^2 \bmod p} \rightarrow \text{reducere la patrate}$$

$$\mathbb{Z}_p^* = \{1, 2, \dots, \frac{p-1}{2}\} \cup \mathbb{Z}_{\frac{p+1}{2}}, \quad p-1 = 2q.$$

Se aplică El Gamal pe an!

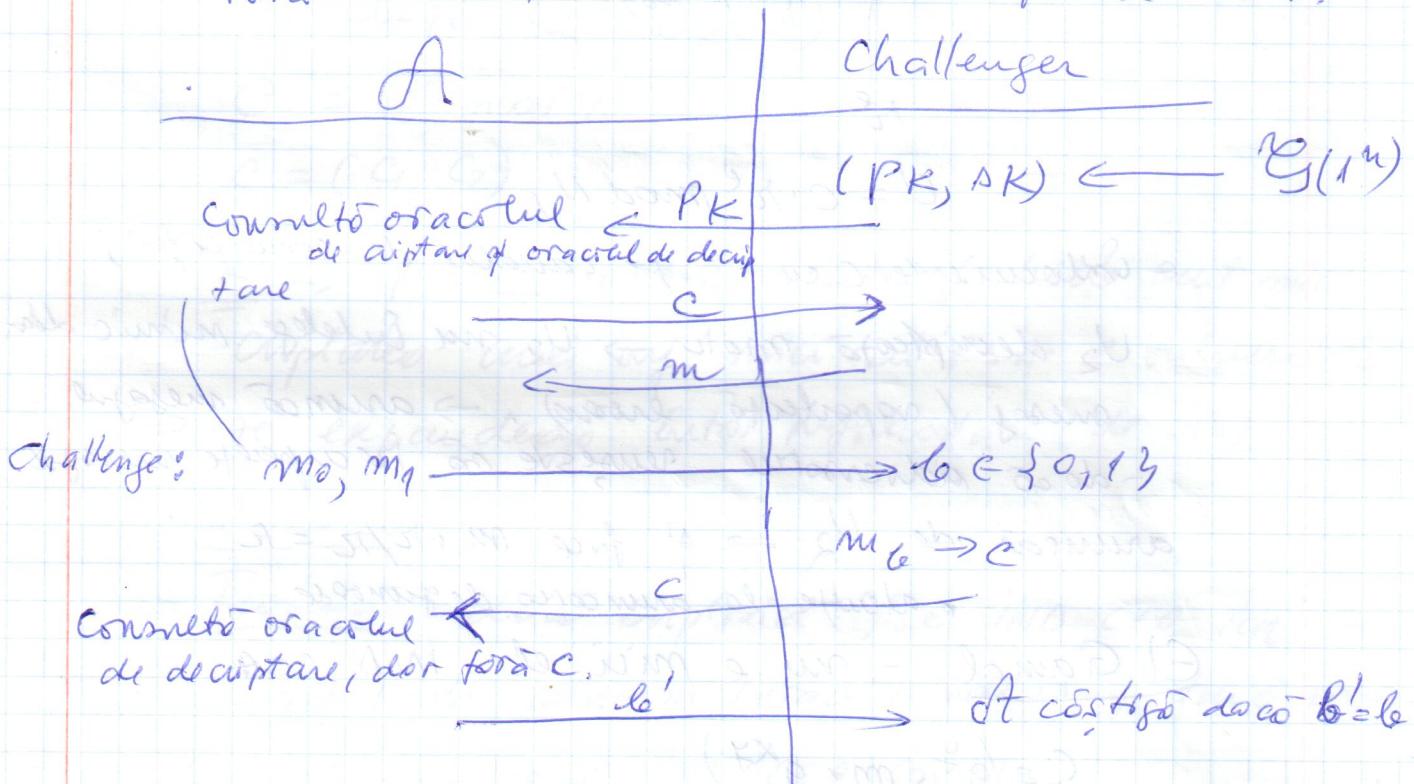
Pt decriptare, se găsește  $m'$ , după care  $\rightarrow$

$m^2 \equiv m' \pmod p$  are exat 2 soluții pt că alege  
stupra din intervalul  $\{1, 2\}$

dacă an conține doar 0, nu mai e în  $\{1, 2\}$  și se face  $\frac{m+1}{2} \rightarrow$  pt că l aduce în  $\{1, 2\}$ .

### IND-CCA

Nici ElGamal, nici RSA nu sunt rezistente la CCA.



Schemă e IND-CCA sigură dacă probabilitatea ca adversarul să cojugeze este neglijabilă

RSA nu e IND-CCA sigură.

- Pe că avem  $(N, e)$  cheia publică PK

$SK = (P, Q, d)$  cheia secretă

$m$  = mesaj pt criptare;

$c$  = criptotextul asociat,  $c = m^e \pmod N$

Adv generează random  $r$ , calculează  $r^e \pmod N$ ,  
și apoi calculează  $c' = c \cdot r^e \pmod N$

→ transmite challenge-ul  $c'$

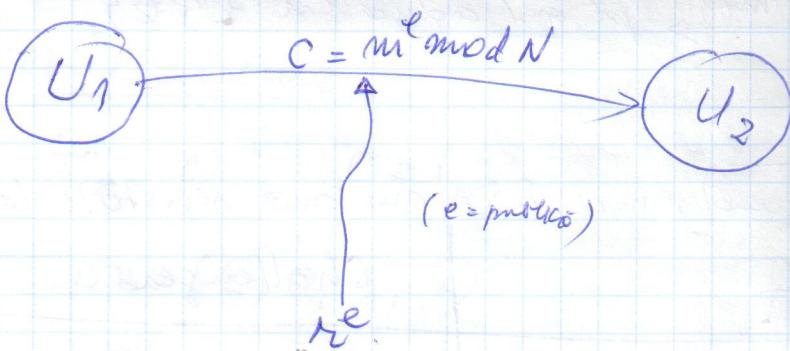
$$c' = (m \cdot r)^e \bmod N$$



$$\boxed{m \cdot r}$$

→ transmite  $m \cdot r$  adversarului

care calculeaza  $m \cdot r / r \rightarrow$  gaseste  $\boxed{m}$



$$c' = c \cdot r^e \bmod N,$$

- înlocuiește  $c$  cu  $c'$  pe canalul de comunicatie;

$U_2$  decifreaza  $m \cdot r$  →  $U_2$  nu înțelege nimic din mesaj (raportarea eroare). → anunță mesajul

- doar adversarul reușește să recuperă mesajul.

anunțat de  $U_2$  → și face  $m \cdot r / r = r$

• atenție la anunțarea de găsimie

El Gamal - nu e nici el IND CCA

$$C = (g^x, \underbrace{m \cdot g^{xz}}_{c_2}, \underbrace{r}_{c_1})$$

generăm  $r$  random,  $C_2' = C_2 \cdot r \rightarrow$  se dezinlocuiește  $C_2$  cu  $C_2'$

→ Nu decriptare se obține  $m \cdot r$  - fără întrebări,

⇒ adversarul face  $m \cdot r / r \neq$  obține  $m$ .

Cum se securizează ale 2 criptosistemuri?

## RSA - sigură do IND CCA

- constructie hibridă -

- se utilizează fct hash,

$$H: \{0,1\}^{2n} \rightarrow \{0,1\}^n$$

$\mathcal{Y}$  = criptosistem simetric,

$$r \xleftarrow{u} \{0,1\}^{2n}$$

$K = H(r)$  cheie pt  $\mathcal{Y}$

se calculează  $C_2 = \mathcal{E}_K(m)$

$$C_1 = r^e \bmod N$$

$$C = (C_1, C_2)$$

$K$  → pt un criptosistem simetric, cheile sunt mici -

criptarea unei inf. mici cu RSA e vulnerabilă

⇒ se expandează către informația - (de aia).

Se demonstrează:

Teoremu - dacă criptosistem  $\mathcal{Y}$  e IND CCA și  $H$  =

= fct hash random (oracle random), atunci schema discutată anterior e IND CCA sigură.

## Semnături digitale (RSA)

căメtădă de autentificare a mesajelor;

- ideea: o criptă cu  $K_S$  și de a verifica cu  $K_P$ .

$$\begin{array}{c} \textcircled{U} \\ \text{SK} \end{array} \xrightarrow{\text{PK}_m, \{m\}_{\text{SK}}}$$

$$\{m\}_{\text{SK}} = m^d \bmod N$$

verificarea semnătării:  $(m^d)^e \bmod N \stackrel{?}{=} m^e \Rightarrow \text{OK}$ .

El Gamal → conduce la DSS (digital signature standard).

Pt implementarea DSS nu RSA se utilizează neapărat PKCS. (standardele pt cript cu chei fruse)

se pot

dag → Hlu, descriere

- link to pdf curs -
- link .sme pdf-urile urcate -
- link pt file modul ( ) -

modul 1 → echipa -

## Managementul cheilor

TLS → pt creațea unui canal securizat între 2 entități A-B,

handshake; se schimbă între A și B material de cheie<sup>4</sup> din care A și B extrag cheia de comunicare.

### • Generarea cheilor

după algoritm {  
- matrice }  
- publice

după modul de filozofie {  
- chei pt confidențialitate  
- pt autentificare

după durată →  
> long term keys  
> short term keys

în PGP → un nivel de cheie -

- securitatea în sistem depinde de enunțementul cheilor

→ 6 cuprind key management;

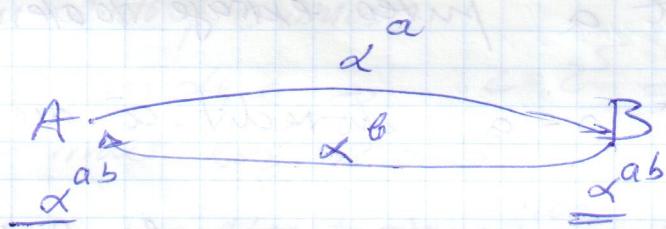
- Generarea materialului de cheie
- distribuția cheilor.
- stocarea materialelor de cheie
- folosirea materialelor de cheie
- update, revizuire și distrugerea materialului de cheie

### Tehnici de generare a cheilor:

1. Servirarea de chei - folosind funcții de extirpare  
de cheie.

2. Utilizarea generatorilor de secrete binare.

#### 1. Servirarea de chei



$\alpha^a$  mod ar pim p

$\alpha^b$  = generator din  $Z_p^*$

a = parametrul secret al lui A

b = parametrul secret al lui B,

$\alpha^{ab}$  = material de cheie - din care urmează să  
se desluze cheile pt confidențialitate, autenticitate  
și integritate (chei de MAC) care se vor utiliza pt o  
gestură de lucru.

din  $\alpha^{ab}$  sunt deriveate chei pe termen scurt

cheile derivate sunt sigure? Materialul de cheie  
e suficient de random pt a fi sigur securitatea  
derilor derivate?

NIST SP 800-56A - spune că aceste chei și  
derivate astfel:

$h$  = funcție hash;

materialul de cheie = SKM (source key material)  
→ se generează o secu binary astfel

se aplică  $h$  pe reprez binary a lui  $t$ , concatenat  
cu SKM, concatenat cu info, concatenat ---

$h(t^u || SKM || \text{info}) || \dots || h(t^u || SKM || \text{info})$

~~este~~ este aleas să i securizeze astăzi o lungime de  
convenabilă pt a putea extrage toate cheile.

acestă tehnică păstrează securitatea

intr-o abordare mai veche, primele 2 arg din  
fct hash erau inversate - (avea și astăzi slăbituri)

- ultima versiune de generare de chei - se bazează  
pe tehnica "extract then expand":

i) - Din SKM se extrage o cheie pt o funcție  
pseudorandom (fct indexată după cheie care are  
o distribuție similară funcției pseudorandom).

$(AES_K, AES_K)$

• familii de fct  
pseudorandom,  
indexate după o cheie

2)  $\rightarrow$  expand; se expandează materialul de cheie  
cu funcția pseudorandom a cărei cheie a fost  
determinată de procedura extract.

$$\underbrace{\text{PRK}}_{\substack{\text{cheie} \\ \text{pt făt} \\ \text{pseudorandom}}} = \underbrace{\text{XTR}}_{\substack{(\text{extract})}} (\underbrace{\text{SKM}, \text{XTS}}_{\substack{\text{solt} \\ \text{"random"} \\ \text{generat, dar fixat}}})$$

$\rightarrow$  Se generează secretul KM

$$\underbrace{\text{KM}}_{\substack{\text{"key material"} \\ (\text{key material})}} = \underbrace{\text{PRF}}_{\substack{\text{făt} \\ \text{pseudo} \\ \text{random}}} \left( \underbrace{\text{CT} \times \text{info}}_{\substack{\text{informații} \\ \text{de context}}} \cup \underbrace{\text{L}}_{\substack{\text{lungime key material} \\ \text{de generat.}}} \right)$$

Dacă SKM = omic (ex: o parola) atunci procedura  
de derivare de key e difuză. deci mesajele mici  
au entropie mică.

32 caractere  $\rightarrow 32 = 2^5 \Rightarrow$  entropie (mugă) cel  
mult  $\log_2 32 = 5$ .

dacă în alfabet avem  $2^{100}$  caractere, entropia = 100  
(produsul de incertitudine, cu cît entropia e mai  
mare, produsul de random e mai mare)

- Metoda anterioră nu mai e potrivită către  
PRK dintr-un mesaj mic nu mai e astăzi de ~~recomandat~~  
random  $\Rightarrow$  se procedează altfel:

P = parola

F = funcție de 3 argumente,  $F(P, S, C)$

F - se obține prin iterarea unei funcții H cu un  
fotă cheie  $\Rightarrow$  materialul de cheie derivat =

$\Rightarrow \cancel{F} \quad Y = F(P, S, C)$

→ Sună 2 standarde:

PKCS#5 (public key cryptography standard)

1) - H este considerat MD2, MD5 sau SHA-1

F = funcția hash înțeleasă C

$F = H^C$  (H iterată de 3 ori)

$$\boxed{Y = H^C(P || S)} \quad S = \text{salt, fixat.}$$

2) H este funcție hash cu cheie, cheie este chiar parola; acesta se aplică. Atunci

$$Y = H_P^1(S) \oplus H_P^2(S) \oplus \dots \oplus H_P^C(S)$$

$\downarrow$  OR

Cele 2 variante sunt state sigure, dar nu sunt "foarte sigure".

- în cadrul acestor state sigure o diversitate are posibilitatea de a utiliza faptul că este oracol.
- ia o valoare x, primește H(x) și poate repeta asta de n ori.

În cadrul acestor state sigure, adăugați posibilitatea de a utiliza H ca oracol, deci că F este coracol.

- pt către o parte altă de el, are posibilitatea de a reda cum arăta reținutul.
- pt că este foarte rezistent la securitate foarte.
- este necesar ca controlul C să fie pusă nu în argumentul funcției.

$$\bar{Y} = H^C(P || S || C)$$

$$T = H_p^A(\Delta || 1) \oplus \dots \oplus H_p^C(\Delta || c)$$

→ derivare de chei = de la un material sau  
suficient de mare;

- sau derivare de chei de la o parolă, cu  
interoare de funcții hash.

In Unix, derivarea de chei de la o parolă =

$$Y = DES_p^{25}(0 \dots 0)$$

sunt, secretul de 64 de biti zero

ECC = criptografie cu arbe eliptice

grupul  $\mathbb{F}_q^*$  de mult. cu o serie de puncte pe o  
arbă eliptică).

### Distrinții de secrete random

- o secret este random dacă entropia sa este mare
- o secret e random dacă e format din secrete produse independent.
- O secret e random dacă are proprietăți statistice bune.

RX : un bit dintr-o secret nu poate fi generat din  
2 secreturi autonome sau successive

- calculul medianselor sau succesorilor trebuie să fie cu  
o probabilitate foarte mică, neglijabilă

### Clasificarea RNG

- Generatoare deterministe (pseudorandom)
  - pot fi pure sau interioare
  - pure : generatorul nu folosește inform. externe împreună
  - interioare : (folos. de ex closul procesorului, temperatură)

- Generatoare deterministe (true random):
  - (TRNG)  $\rightarrow$  proces fizic sau parcurse si datorită unui calculator.
  - Acestea pot fi pure sau hibride.

### PRNG pur (determinist)

Definire: este un 4-uple  $G = (S, O, f, g)$

$S =$  mulțime finită de stări

$f =$  funcție de transiție.

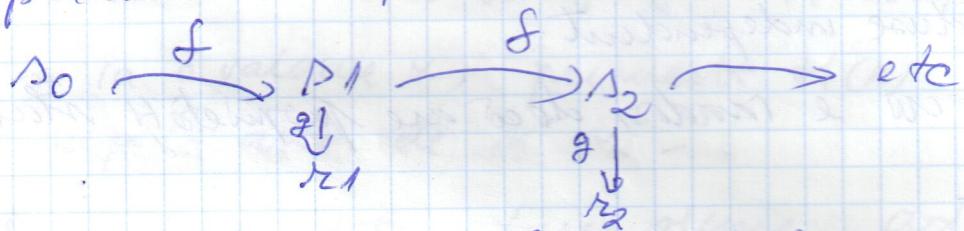
$\theta =$  multime finită de expresii

$g: \theta \rightarrow O$  funcție de output

$f: S \rightarrow S$

$g: S \rightarrow \theta$

$s_0$  = se generează random și de la ea se pleacă mai departe.



RCY e construit în acest fel -

(se stabilește starea initială, apoi se modifică starea și se generează output  $\Rightarrow$  (în al 2-lea algoritm))

- Se numără se generează separat ( $s_0$ )

Generatul e determinist (folosind același  $s_0$  ar da același rezultat !!)

Exemplu de PRNG pur

$$f(x) = ax + c \bmod m$$

nu însă  
căci

Generatul mu e suficient de bun (nu satisfacă R<sub>2</sub>)

→ pot calcula predecesori și succesiuni

dacă cunoști x, pot calcula  $ax + c \bmod m$

-  $x \in \mathbb{Z}_m$

$$\begin{matrix} 0, 1, \dots, m-1 \\ i_1, i_2, \dots, i_m \end{matrix}$$

atât el, dar în același ordin.

alt exemplu: LFSR. (A 5/1 din GSM)

stare e un vector  $(x_1, \dots, x_k)$

plecând de la ac vector se construie un nou element, aplicând funcția → se constr. un nou element  $(x_2, \dots, x_k, x_{k+1})$

$$\underbrace{x_1, x_2, x_3, \dots, x_{k+1}}_{\text{etc.}}$$

Un alt tip de PRNG -

- ex: plecând de la un criptosistem bloc,

$$S = \{0, 1\}^n \times \{0, 1\}^m$$

$$(x, k) \rightarrow (\{x_{j_K}\}, k) \rightarrow (\{\{x_{j_K}\}_K\}, k) \rightarrow \text{etc.}$$

$\downarrow$   
 $\{x\}_K$                                      $\{\{x\}_K\}_K$

~~doar~~

Criptosistemul satisfacă R<sub>1</sub>, dacă nu are probleme statistice grave. Dar R<sub>2</sub> nu e sat. → (dacă atea forme cunoștește starea internă a PNNG)

- Amoșterea sănii interne nu ar trebui să permită generația de noi elemente "random" →  
→ fel de output ar trebui să fie complet cotaț

→ R3: funcția de output trebuie să fie one way (să fie greu să calculezi statele anterioare)

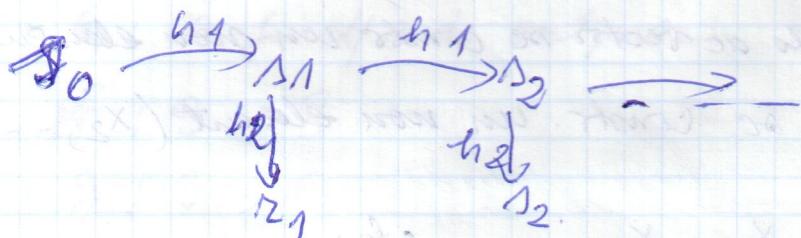
Fct. one way apar sub formă de fct. hash în criptografie -

ex 5: PRNG cu fct. hash.

$$S = \{0,1\}^n$$

$$\text{fct. hash } h_1 : \{0,1\}^n \rightarrow \{0,1\}^n$$

$$h_2 : \{0,1\}^m \rightarrow \{0,1\}^m, m < n$$



Dacă  $h_1 = h_2$  nu e satisf. R2  $\Leftrightarrow \Delta_1 = \Delta_2$

E: Generarea BBS (Blum Blum Shub) PRNG

→ bazați pe nefiduciozări potențiale

$$m = p_1 p_2, \quad p_1, p_2 \equiv 3 \pmod{4}$$

$$S_0 \in \mathbb{Z}_n$$

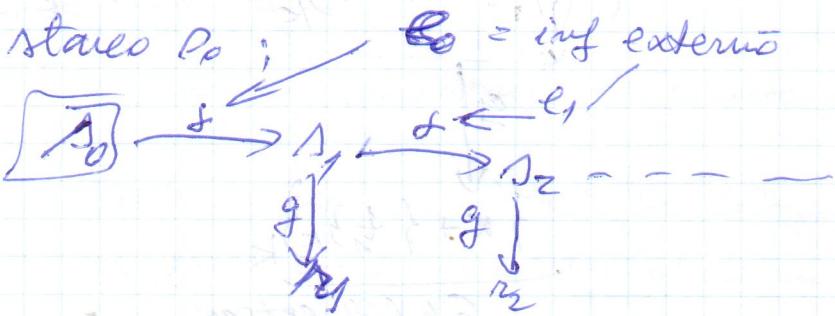
$S_1 = S_0^2 \pmod{n}$  și se extrage bitul cel mai putin semnificativ -

$$S_2 = S_1^2 \pmod{n} \rightarrow \text{etc.}$$

⇒ are complexitate ridicată (calculul modular

$$\text{se poate } \Rightarrow O(\log^2 n)$$

## Generatoare mixte folosesc și inf externe



acest PRNG mixt are și proprietatea de nedeterminism  
dacă  $e_1 = \text{temp proasembină}$  - dacă plac de la moment  
diferite, am rezultat diferență.

- Gradul de nedeterminism este însă mai mare.  
Pentru un astfel de generator, este probabil ca  
selecția de la oarecare etapă să rămână, gen. să  
genereze aceeași secvență random la moment de timp  
diferit.

Dacă  $e_0, e_1$  și  $r_1$  sunt obinute în același timp, mico  
putem începe să avem un prob.  $\rightarrow$  curățenie  
stare internă, și parcursând toate variantele  
de probabilitate  $\rightarrow$  se obțin stări uniforme.

$\Rightarrow$  Ris. Generarea ebină random curățenilor  
stări interne să aibă o probabilitate mică, neglijabilă.  
mult, căci să aibă și să fie frica predictibilității -  
- - -

## Generatoare PRNG care folosesc 3DES

Criptografă elem de  $8 \times 8$  bit, cu cheie de 128 bit.

Modulul EDE = criptare-decriptare, criptare.

Se pleacă cu  $A_0 = (X, K)$  și se aplică f cu  $t_1$

$$D_0 = (x, k) \xrightarrow{f} \left( \left\{ x + \left\{ t_{ij}^3 \right\}_K \right\}_K \oplus \left\{ t_{ik}^3 \right\}_K, k \right)$$

eg

$$\overbrace{\left\{ x + \left\{ t_{ij}^3 \right\}_K \right\}_K}^{64 \text{ biti extrai}}$$

$$t_{ij}^3 \\ \downarrow \\ f \longrightarrow$$

delay trebuieste = de lungime valoare.

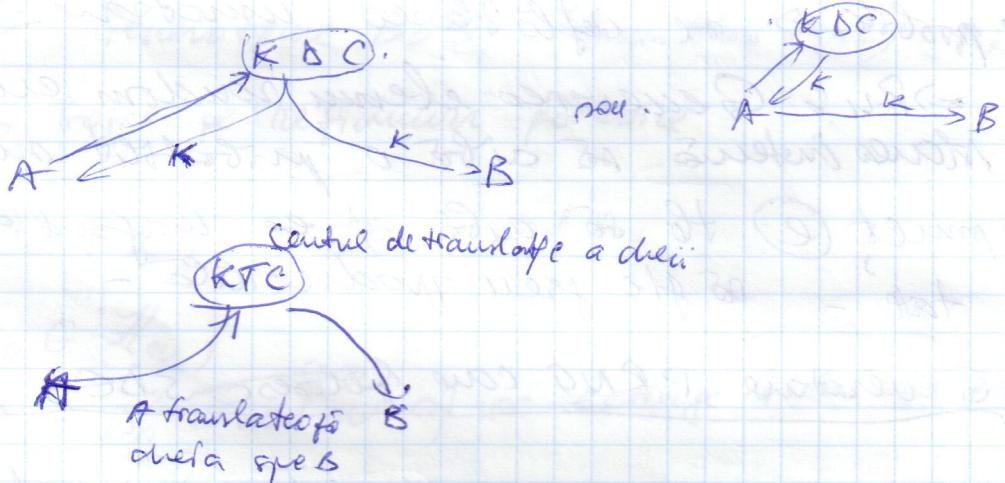
### Stabilirea cheilor

- Protocoale de distribuire a cheii

→ folosesc material de cheie pre-distribuit

TPP - trusted third party  $\rightarrow$  serve de intermediar  
centru de distribuire a cheilor

KDC  $\rightarrow$  key distribution center.

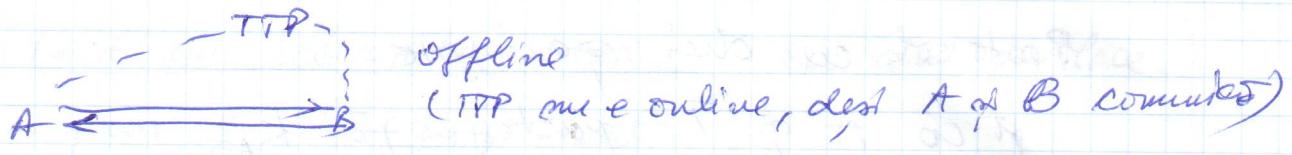


Authority de certificare -

TPP - poate fi inline, online sau offline

$A \longrightarrow \text{TPP} \longrightarrow B$  (inline)

$A \xleftarrow{\quad} \text{TPP} \xrightarrow{\quad} B$  online



(la comentul electronic, TTP este banca, de obicei)

- Protopoalele - urmăresc pe lungă confidențialitate, menținerea cheii și controlul comun al cheii pt cele 2 părți implicate; mai sunt și urm. cerințe necesare: autentificarea partilor (A foto de B și B foto de A sau în ambele sensuri, mutuală)
- autentif - originii dozelor
- autentificarea implicită a cheii
- confirmarea cheii -  
(Cele 2 entități și fiecare dintre ei lucrau cu același cheie)
- autentificarea explicită a cheii (autentif implicită + confirmare = autentif explicită)

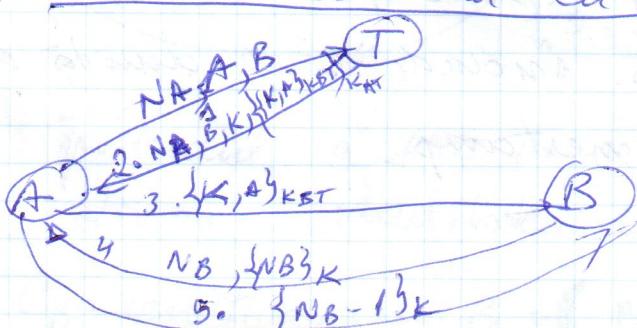
#### \* Protocoale de distribuție -

- point to point (lucrătoare doar cele 2 părți care să distribue cheia)
- centralizate (cu TTP)

- Key agreement (2 părți care să acorde acord comun unui material de cheie pe care îl vor utiliza mai departe)

#### Exemple de protocoale:

##### 1. Needham-Schroeder cu chei partajate



A = solicită cheia  
B = secolică în rețea

Protocol cu cheie partajată  $\rightarrow$

A și B au parte la cheie partajată  $K_{AB}$

B și T au parte la cheie partajată  $K_{BT}$

$\{m\}_K$  = criptare cu cheia  $K$ .

A dorește să comunique cu B și are nevoie de o cheie.

$N_A$  = number once used - (nonce)

Este număr utilizat de la serviciul

$N_A$  identifică unic sesiune.

Cheia este autentificată implicit prin intermediul serverului - cheia e confirmată către A și B.

$$A \quad m^r \\ m^r \neq N_B ; \quad m^r = \{m\}_K - 1$$

$$m^r - 1 \rightarrow \{m^r - 1\}_K$$

dacă serverul acceptă reacția  $\rightarrow$  cheia e confirmată.

A se autentifică către B.

Hicketul  $\equiv \{K, A\}$ ; cind B primește mesajul de la A și verifică dacă identifică entitatea și cea din Hicket.

sesiune 1: A comunică cu B folosind  $K_1$ ,

sesiune 2: A solicită o cheie de comunicare cu B;

- serverul să o refuze; și poate folosi tehnica din ultima sesiune  $\rightarrow$  (un hicket mai vechi); serverul calculează un nounce, și îl poate decripta cheia -

Protocol Kerberos include și o durată de viață, și datele of an timestamp.

Needham-Schroeder Lowe este un protocoal public.

- protocol de agreement.

A și B schimbă contacții  $N_A$  și  $N_B$  între ei, din cauza vor generă drept de comunicare.

(material de cheie)

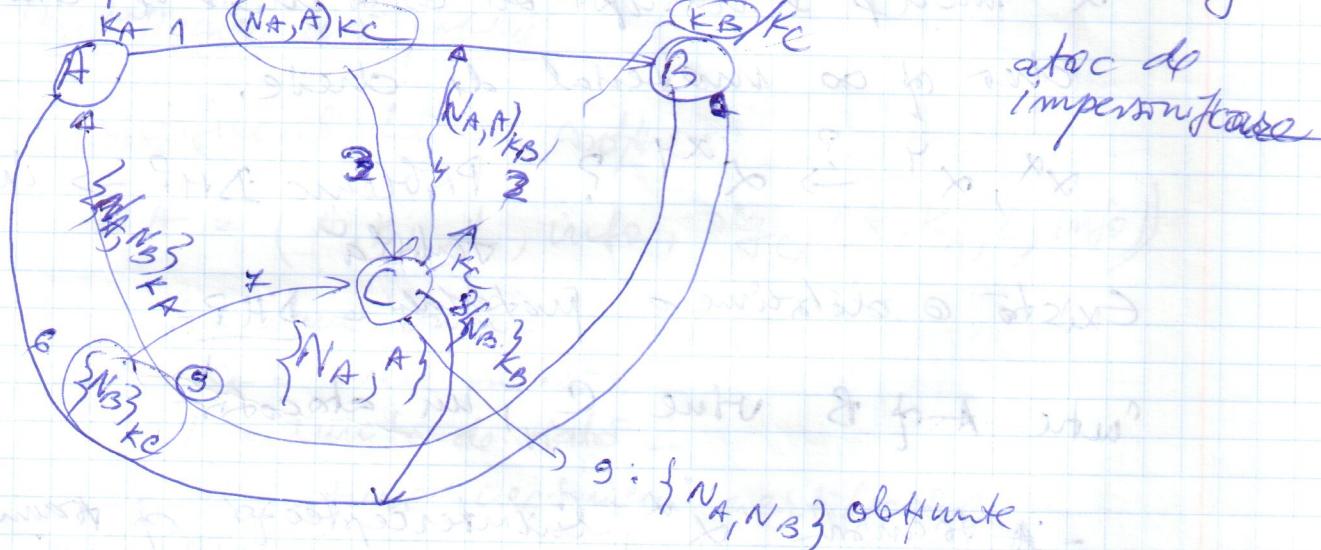
A generează un nume  $N_A$  random; A obține  $\{N_A, A\}_{KB}$

2 B generează un nume  $N_B$  și obține  $\{N_A, N_B, B\}_{KA}$ ;  $B \rightarrow A$ ; - confirmă  ~~$N_A$~~   $N_A$  există.

-  $A \rightarrow B \{N_B\}_{KB}$

$N_A, N_B$  - din acasă vor decide dacă ce vor face.  
ulterior.

Modificare:  
to poate să scoată identitatea lui B din mesaj



Protocolul a fost implementat pe Unix folosind introducerea identității lui B  $\rightarrow \{N_A, N_B, B\}_{KA}$ .

C\_B - certificarea lui B  $\rightarrow (B, K_B, \text{certif de securitate})$

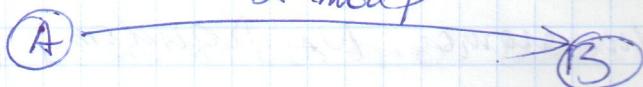
## Diffie Hellman Protocol

- schimb de cheie
- protocol de acordare (material de cheie)

Se utilizeaza un nr prim  $p$

$$\alpha = \text{root primitive mod } p$$

$$\alpha^x \pmod{p}$$



$$x = 1, p-2$$

$$1 < x \leq p-2$$

$x = \text{random}$ .

$$1 \leq y \leq p-2$$

$y = \text{random}$ .

$x, y = \text{generate random co elem secrete ale lui } A \text{ si } B$ .

A trimite lui B  $\alpha^x \pmod{p}$

$$B \text{ calculeaza } (\alpha^x)^y = \alpha^{xy} \pmod{p}$$

B trimite lui A  $\alpha^y \pmod{p}$

A calculeaza  $(\alpha^y)^x \pmod{p} = \alpha^{xy} \pmod{p} = \text{santitate comună}$

$\alpha^{xy} \pmod{p}$  e utilizat ca co de cheie de comunicare  
dintre co material de cheie.

$$\alpha^x, \alpha^y \rightarrow \alpha^{xy} ? \quad \text{Problema DHP} \rightarrow \text{intractabila}$$

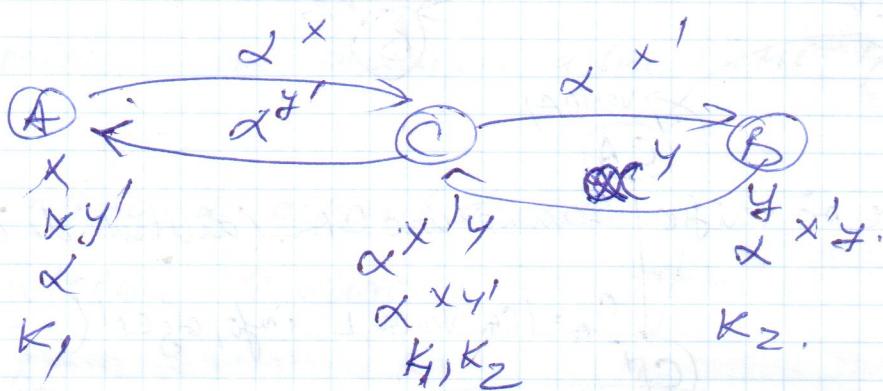
Există o abordare puternică - DHP -

Cineva A și B văd C, un atacator

- A trimite  $\alpha^x$ , B interceptează și trimite lui B  $\alpha^{x'}$
- B trimite lui C  $\alpha^{y'}$ ; care nedecryptează la  $\alpha^{x'y'}$

dorul C poate calcula  $x^y$

- A utilizează K<sub>1</sub> dor de la font o informație C.  $\rightarrow$  impersonare



$$\{m\}_{K_1} \rightarrow m$$

$$\{m\}_{K_2} \rightarrow m$$

$$\{m'\}_{K_1} \leftarrow \{m'\}_{K_2}$$

### Station to station protocol

Protocolul are autentificarea mutuală a celor două parteneri și autentificarea mutuală explicită.

(verifica)

### Infrastructuri de chei publice (PKI)

#### Certificarea cheii publice.

Certificatul lui A arată:

$$CA = (A, K_A, L, info, sig_{CA}(A, K_A, L, info))$$

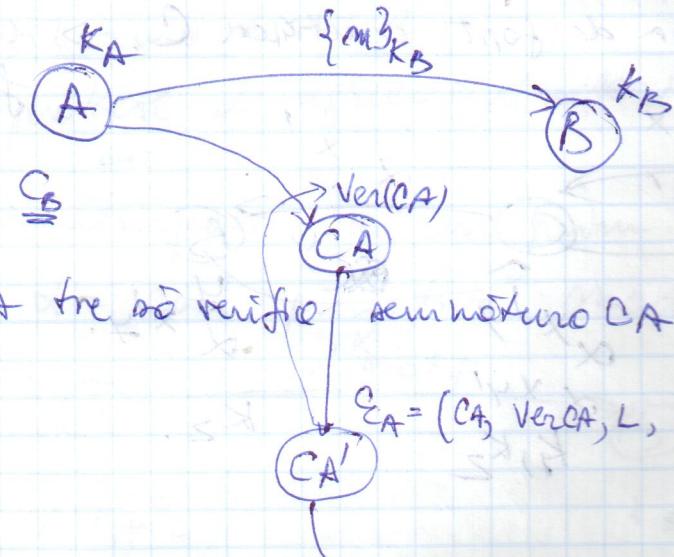
A = nume owner

K<sub>A</sub> = cheie publică

L = domeniu de validitate -

semnatura certificatului certificatului

#### Autorități de certificare ⇒ Visa, VeriSign



A trebuie să verifice semnatura CA (autoritate de certificare)

$$C_A = (CA, \text{VerCA}, L, \text{info}, \text{sigCA}'(\dots))$$

Lant de încredere

Potrivit se aranjează că o autoritate de certificare rodocimă: (Root CA)



2 autorități se pot certifica reciproc

### Revocarea certificatelor

- Când se cheie publică e compromisă, sau durata de vîrstă a expirat, fișe revocate -
- Revocarea unei chei publice = revocarea certificatului → liste de revocare a certificatelor;
- și informații despre certificatelor valabile -
- protocolul care dă statutul acestor certificate valabile (OCSP)

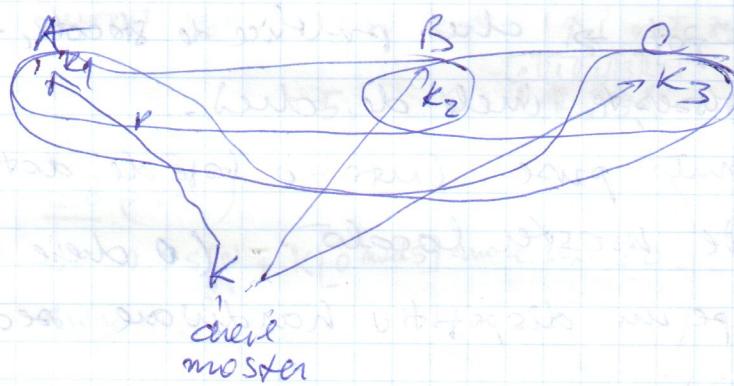
PGP 5.0 → se poate instala pe Thunderbird.

70% dintre utilizatori nu → capabili să utilizeze astăzi de certificate.

S/MIME = vari. secundară a lui MIME (pentru postele electronice)

### Alte tehnici de ochinut de cheie

- 1) • Partojirea de secrete



partojore  $K_1, K_2, K_3 \rightarrow$  oricare  $\geq$  dim ei pot forma ce cheie  $K$ .

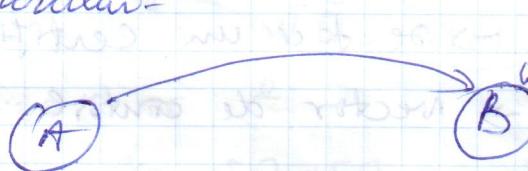
- 2) → Distribuirea prim călal cuantic.

→ două căi cuantice  
deja perturbate comunicarea

- 3) Criptare boala pe identitate

→ Cheia publică se calculează din identitatea destinatarului

cheia secretă



$ID(B) =$  nume / adr. email / etc.

$H(ID(B)) =$  cheie publică a lui B.

Transmis mesajul cu această cheie publică

Bare nevoie de o cheie secretă ← PKG (generator de chei private) → care să furnizeze cheia privată sau

VOLTAGE (USA)

INDENTUM (UK)

→ cost soft bogat pe id base  
encryption produce!

În 2001 apare însistem bogat pe id based encryption -

### Stocarea cheilor

Ement securizat → chei publice de stocat, etc.

PGP - filosofie inel de chei.

- cheile sunt posse într-o formă de date criptată cu o cheie master locală - (o cheie mare →

→ stocată pe un dispozitiv hardware securizat inclusiv HHC)

### Atacuri side-channel

Blocajul cheilor → ANSI X9.17 -

### Modurile de utilizare a cheilor -

1) K → mai multă utilizator, dar cu diverse restricții

(+ politici de utilizare a cheilor)

- tag-uri - titluri - care recomandă un utilizator al cheii

- sau: utilizator intr-un mod specific utilizatorului

- sau: notarizare → se face un certificat pt cheia resp.  
notarizare + tag → vector de control.

! - Utilizarea particulară de control al accesului în meniu generalul cheilor -



EB 9R

०८८

~~ANAK~~

Protoceratops

→ Vulnerable MIM. (man in the middle)

- the identification - intercalation (the drug is incorporated)

station-to-station = Janato Subnets 154/15

Hellman -> 305 + 200(2000) + 1024 = 6

Setup: p - min - max (512 bit)

$$275 \leftarrow \text{Next} \quad \times \in \mathbb{Z}_p^* \quad \text{where } (275)_{10} + [3]_2 + 5 = 1$$

$\alpha$  = generator (odd primitive) pt  $\mathbb{F}_{p^2}$

pt  $P = 2q + 1$ ,  $P, q$  prime; impairs

generato altrò  $2 \leq \alpha \leq p-2$

calculo de los modos

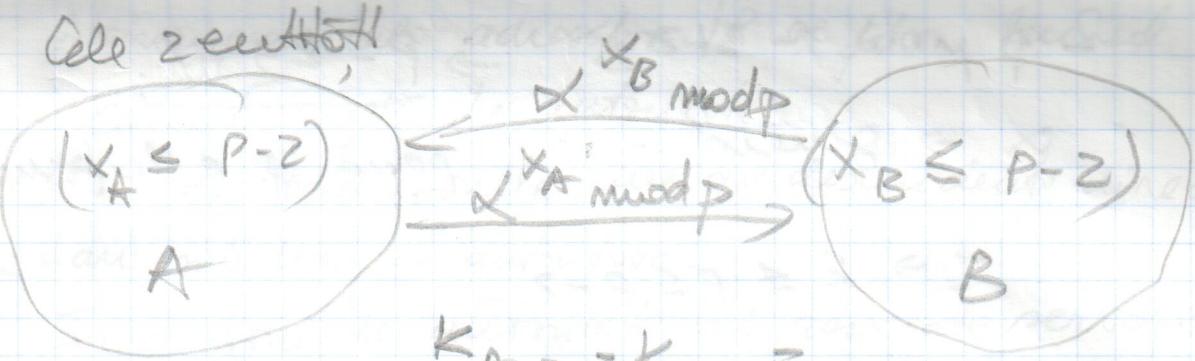
do  $\alpha \leftarrow \alpha^2 \bmod p$  if they return ( $\alpha$ )  
else return ( $p - \alpha$ )

alg merge unuai pt mure pîme de

$$\text{from } P = 2g + 1$$

- In Java liegen die **Byte** tegger

-glucometabol 2) apr. P-UL-IL calc.



$$\begin{aligned}
 K_{A,B} &= K_{B,A} = \\
 &= \alpha^{x_A x_B \bmod p} = \\
 &= y_B^{x_A \bmod p} = y_A^{x_B \bmod p}
 \end{aligned}$$

B reține înf. propriu și astfel poate semnala "nu îl informați trimit";  
în fel, A - ar putea să nu îl trimit.

O semnătură generată cheia privată,  
(se folosește RSA - gata făcut sau DSS  
gata făcut) - E nevoie de o cheie publică  
pentru verificare;

Păcătoarea certificatului =  $\alpha^{x_A}$

He un PKI care să fie complet;

⇒ la RSA cheia publică este numărul  $n, e$ .  
Partea 2 - aplicabile la station to station -  
- deriv. o cheie comună folosind SHA-256 -  
(gata implementată)

(Handbook of Applied Cryptography, cap. 12)

- \$ prime - 3123  $\rightarrow P = \underline{29+1}$   
 $\alpha \rightarrow$  generat - până p e de formă.

$$\alpha \rightarrow 2 \leq p \leq P-2$$

$\rightarrow \alpha = \begin{cases} \text{pașum public} & (\text{una din cele } 3) \\ P-\alpha \end{cases}$

DNS SEC, extensie de securitate pt DNS.

DNS - probleme:

- lipso autentificare
- lipso mecanism de verificare a integrității

DNS sec - adaugă un mecanism de autentificare + integritate

- Nu adaugă confidențialitate de mai mult fel
- Autentificare și integritate = nume scrisoare digitală și lanturi de încredere

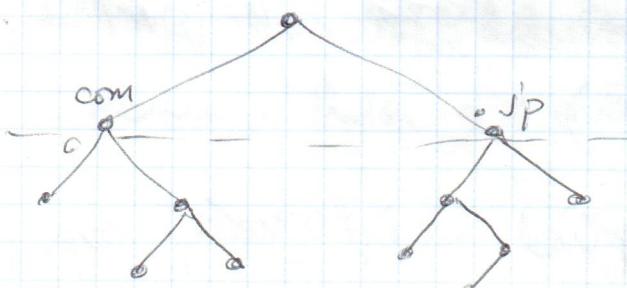
Fiecare host (găzdu) are asociat un IP

- găzdu (host) primesc un nume; nu neapărat nume  $\leftrightarrow$  IP (fileer hosts.txt)  
 (în trecut)  $\rightarrow$  domeniu pe care de acasă

$\rightarrow$  mai târziu; DNS, dynamic DNS; NIS, LDAP...etc

Adăugă IP = structură arborescentă  $\Rightarrow$  arboarele DNS

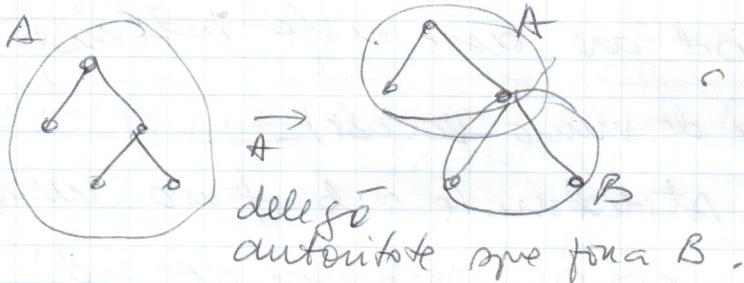
- Arboarele DNS  $\rightarrow$  împărțire în domenii și subdome



Managementul adreselor IP se face lucru în calcul zonele de autoritate.

- de obicei zonele care au descendenti directi au suveranitate de autoritate -

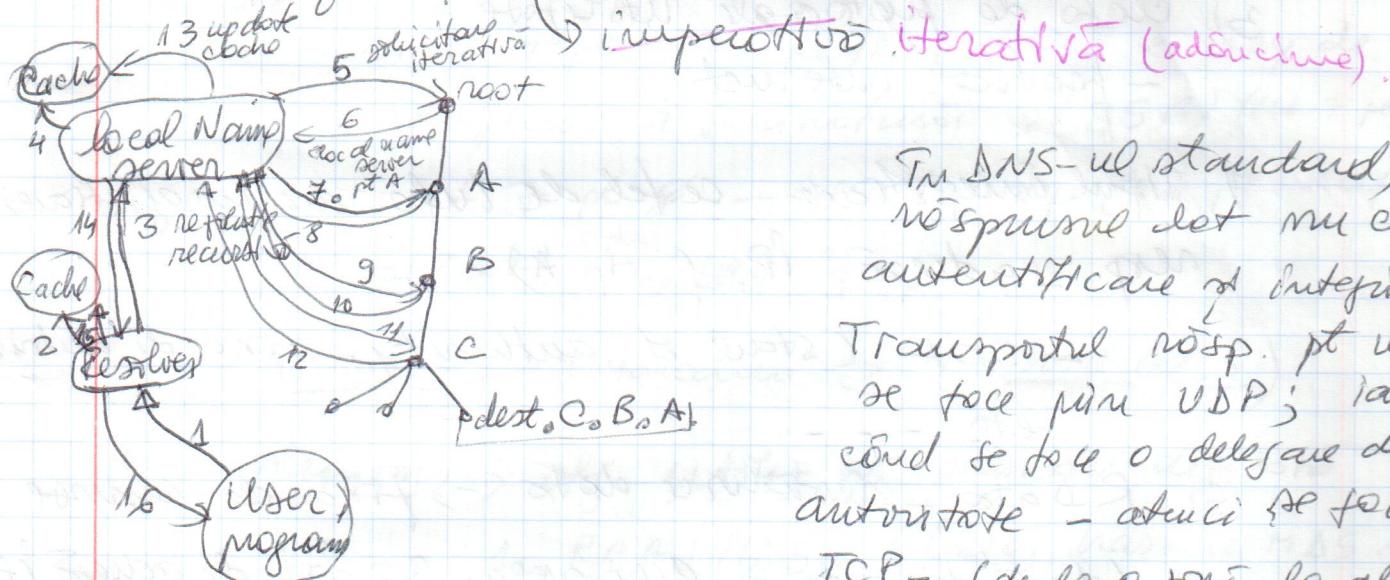
Tieci zone de autoritate care asociate servore de nume (D. primare și servire secundare)



- la primele de delegare se adaugă înf. speciale de securitate

legătura între zonele gozdăi și înf. asociate gozdăi = reprezentarea numelui de domeniu

Reprezentare → recursivă (BFS).



În DNS-ul standard, nu există seturi de configurare autentificare și integrare.

Transportul răsp. pt utilizare se face prin UDP; însă când se face o delegare de autoritate - atunci se folos.

TCP - (de la o fază la alta)

Serviciul de număr local pt A poate fi în zona de autoritate a lui B, de ex.

Cum adăugăm securitate la DNS?

- Semnătura digitală combinată cu un lant de încredere (⇒ metodă de verificare)

o autenticitate semnătură.

- Când se cere ID-ul pt destinatar, înf tb semnată - (găseșc elem de autentic. ale semnăturii săi C în B  $\Rightarrow$  etapele următoare până ating la root  $\Rightarrow$  lăsat de încredere)

$\rightarrow$  În fiecare nod are asociată informații

(în servirele de nume locale).

- înf sunt structuri sub formă înregistrării de resurse (RR - resource record)

Structura generală a unei înreg. de resurse;

1) Nume domeniu.

2) Timpul de valoare înregistrării (TTL  $\rightarrow$  live)

3) clasa de protocoale utilizate  
- protocol internet

4) Tipul înregistrare - ce fel de info are înregistrare  
resp.: adresa IP (A) -

- SOA (start of authority - zona de autoriz.  
etc - - -

5) RData - resource data  $\rightarrow$  formă de resurse

(dacă A =  $\star \rightarrow$  aici avem 32 biti de adresa IP)

DNS are vine cu 4 tipuri noi de RR -

1) HPSK DNS KEY - cheie de semnare

(pt semnătura digitală)

2) RRSIG (se găs. semnătura digitală pt  
o mulțime de bune de același HPSK)

3) NSEC / NSEC3 (în forma de delegare de autoritate pt autentificarea responsabilului)

4) DS - necesară verificarea cheii publice de ~~verificare~~ autentif a semnăturii -

#### 1. DNS KEY

nume dom	TTL	clasa	tip	RDATA
		TN	A	
			SOA	
			MX	

(severul)

pt DNS KEY:

RDATA:

Flags	Protocol	Algorithm
Public Key		

forma de Flags: flag - binic cu diverse utiliz.

- bitul 7 pe 1  $\Rightarrow$  Keya publică e key de verificare a semnăturii  $= (256)$  (bit 7 pe 1)

bit 7 pe 0 - are alt scop (nu pt verificare)  
 $= (257)$   $\Rightarrow$  (bit 7 pe 0)

Protocol  $\rightarrow$  3 valori  $\exists$  (must be 3)

Algorithm  $\rightarrow$  alg. utiliz pt semnare digitata

1. RSA / MD5. (intotdeauna hash cu MD5 pt apoi semnare digit cu RSA)

3. DSA / SHA1. (elgamal Standardizat)

5. RSA / SHA1.

8. RSA / SHA-256 (sha2)

9) RSA / SHA-512. (sha3)

cel mai  $\Rightarrow$  13) ECDSA / SHA-256. (recm. pt utilizare)  
bună  $\Rightarrow$  alg de semnare pe curbe eliptice !

Sext → 14. ECDSA / SHA -384 - recomandat pt utilizare

Public Key e specificat în format Base 64 -

- Cum se lucrează în Base 64:

→ tabelă cu 64 elem; tabeloarele sunt ordonate

astăzi: 0-255 → caracterele A-Z

urm 26 elem  $\Rightarrow$  a-Z

urm 10 caractere: 0-9 -

· 1 caracter  $\Rightarrow$  + (pe poz 62)

· 1 caracter  $\Rightarrow$  / slash

= pt padare -

→ total: 64 caractere -

Secv. binară grupată în octetii și secv. contine un număr de octetii; se împart în grupe de 6 biti de la stânga dreapta; fiec. grup de 6 biti este un nr. între 0 și 63  $\Rightarrow$  un codificat cu tabela Base 64.

Apare o problemă; dacă ultimul grup nu are 6 biti, se pune 4 de zero de la stăpână și se poate face cu 2 de egal.

rezolvare:  

6 biti	2biti	0000
--------	-------	------

 == (câte un egal pt 2 de zero)

6 biti	6 biti	4biti	00
1	6	4	

 = (un singur egal pt 2 de 0)

## Hipul RRSIG :

Contine semnătura digitală pe un set de înregistrările de același tip → RR Set

RR Sig e semnătura digitală pe un RRSet

- aceloră înreg sunt ordonate canonice (lexicografic)

(ex iau toate înreg de același tip, se ordenează; apoi se iau urm. înreg de același tip → se ordenează etc)

Componente pt RRSIG.

TIP acoperit (ex: SOA, etc - -)	TIP algoritmul utilizat pt semnat	(e Hacheț) Labels pt a distinge între zone de adorit și diferențiat
Original TTL (același TTL pt toate)		
Signature expiration		
Signature Inception		
Key Tag - tag-ul cheii pt contur <b>ropidă = un hash.</b> <b>signature</b>		
Signer's owner (proprietatea semnatului)		
semnătură (în baza 64)		

$$\text{Signature} := \text{sig}_K(RRSIG = \text{Rdata} \parallel \text{RR}(1) \dots \parallel \text{RR}(n))$$

unde RR sig - Rdata se referă la toate componente din tabelul de mai sus.

altă,  
înreg., NSEC → are 2 scopuri →

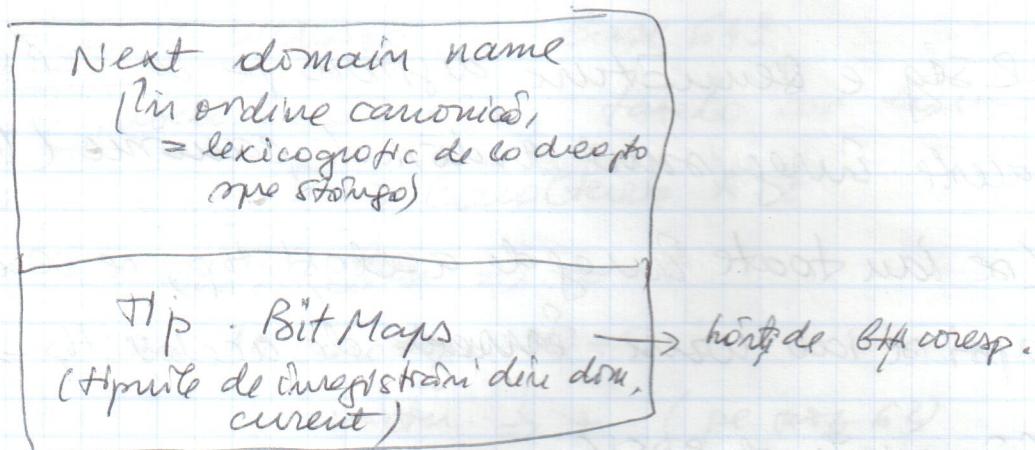
1) furnizează urm. nume de domeniu în ordine canonico, sau urm. punct de delegare de autoritate.

2) furnizează puncte de înregistrare ale dom.

En conj.

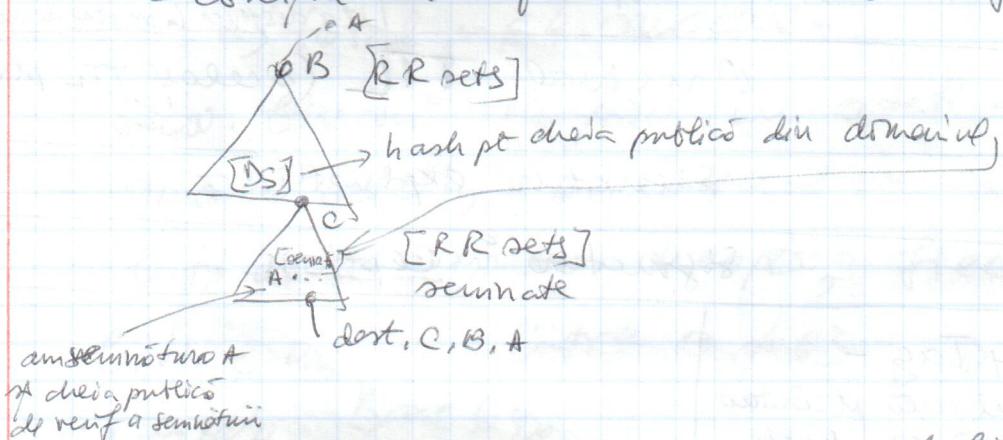
- acesta tipuri de înreg. autentifică și respunsele în special negativul unei reprezentări

Formatul compunerii RData:



Tipul DS.

- conține un reperuri hashi de verificare a semnăturii



→ pretează autenticitate

RData pt DS

2 octetii Key TAG	Algoritm 1 octet	Tipuri de reperuri Hash
Reperurile hashi (mesaj digest)		

Digest  
Type.

Tipuri de → 1. pt SHA 1  
reperuri Hash. } 2 → 255 - neasignate

Deploymentul DNS sec la nivel de root → iulie 2012

DNSSEC → dezvoltarea structură arborelui DNS -

→ fac verificări de nume de reprezentări pt domenii care nu există -

→ dezvoltă un interog pînă la structura și unei forme de autoritate → Tehnică se numește

### Zone enumeration -

- pt a contracara acest lucru → S-a introdus

NSEC 3. În care numele unui domeniu sau punct de delegare e inclus în varianta Hash (nu se mai include în cîr) - în ordinea de afisare nu mai e ordinea canonico pe domeniul; ci ordinea canonico pe hash-ul numelui de domeniu.

Struct. compului RData:

algoritm de hash 1 octet	Flogs	iteratii (de câte ori se aplică hash-ul)
Salt length (cînd random number length)	Salt	(random care se adaugă cînd se face hash)
Hash length. 160 biti (20 octeti)	Next Hashed	Owner Name (ordinea se face pe hash, nu pe nume de dom)
Type BTH Maps.		

În Nsec 3 nu se include lungimea Salt-ului și nici lungimea hash-ului -

ANSEC 3 →

DNSEC → adaugă autenticitate și integritate

- nu aruncă confidențialitate
- nu aruncă protecție pt denial of service.
- Există și alte soluții de securitate a DNS-ului.

→ Baza se pe criptografie cu chei publice și protocol Kerberos -

(dor există probleme monogenanțului cheilor public  
(în diverse certificatelor, doar verific hash-urile existente)

## TLS

- Obținut din standardizarea SSL și Embunckit

- Nu se referă neraportare.

se obține doar : confidențialitate

: integritate

: autenticitate

- SSL 3.0 - compus din 2 layeri.

① subprotocol care se ocupă cu prelucrarea datelor - core conferă confid. și integr. (SSL Record)

② protocoalele - celelalte:

- handshake, schimbul de date (autenticitatea și securitatea handshake), mesaj de alertă, SSL application data (conține date de la aplicație și spune ce trebuie făcut cu ele)

Conecțiune = transport în modelul OSI

→ sunt parametri de codare -

→ mediul în care oferă parametrii criptografici într-o secvență potrivit careva mai multe combinații

- fiec. conex. are parametri specifici și parametrii implementației, obținuți din parametrii secvenții

Secvențele sunt cauzate prin parametrii -

→ SSL e un protocol bogat pe stare

- o stare a unei conexiuni furnizează parametrii conexiunii

Starea de read = parametru necesar pt a citi mesajul primul

Starea de write = am parametrii mesajelor trimise de utilizator

→ dacă vreau să schimb parametrii criptografici →

se realizează alte 2 stări { pending read

pending write.

Protocolul SSL CCS nu face decât să aminte parțial de comunicării că o-a schimbat starea în pending sau invers.

### - Parametrii SSL :

- suport criptografic

- secret master

- flag - care spune că în sesiunea curentă se pot crea alte conexiuni.

### - Starea noii conexiuni :

- 2 nounce - unul la client și unul la server

- 2 key pt integritate.

- chei de criptare client-server.

- vector de initializare.

- numere de secvențe pt a stabili ordinea mesajelor vehiculate

Master secret : - se taie din el segmentele din care se fac cheile

- cum se alcătuiește secretul master? :

- se utilizează un secret pre-master, nounce-ul client, server și multe constante predefinite; se utilizează ~~SHA-1~~

~~și MD5-(old!)~~ → SHA-256, 512.

Secretul pre-master - sunt mai multe metode :

- RSA → clientul alege un secret pre-master, îl cripteză cu cheia publică a serverului și îl trimite serverului

Metoda DH → în 3 variante: 1) fixă - secretul master se stabilește între client și server și se stabilește  $\alpha^x$ ;  $\alpha^y$

~~0019BB4A88E7 moe secret~~

din cele 2 se calculeaza  $\alpha^x \beta^y$

→ parametrii sunt de la prim certificat, sunt semnati.

2) metoda efemera - - schimbă  $\alpha^x \beta^y$ , semnează  
acești parametrii ca metoda curentă (RSA sau DSS)

(în metoda fixă,  $\alpha^x \beta^y$  sunt bătute în casă,  
sunt specifici și nu certificata)

3) metoda DH dor fără autenticare,  
calculând  $\alpha^x \beta^y$  dor său - i mai securiză).

Metoda Fortezzo (similar RSA, dor se folosește  
alt criptosistem în loc de RSA → Skipjack)

### Generarea secretului Master

sha(A, me-master-secret) =

= sha('0x41' || me-master-secret || Nc || ns)

generarea parametrilor criptografici -

→ Key block - se face MD5 (me-master-secret || sha(A,  
master-secret)) || MD5 - - - - (de unde că am nevoie)

- din blocul asta fac de la dreapta 32 octeti cheie  
în ordinea specificată pe slide ⇒ client server write MAC

EXAMEN PE 30  
IANUARIE

} client/server write key  
} client server init vector  
} și ce mai am nevoie

### Protocolul SSL handshake

→ cel mai complex - vezi pe slide 18~

Sunt 2 forme de boga pt hand shake

- stabilirea unei noi sesiuni
- refacerea sesiunii existente

Ce conține fiec. mesaj:

- sunt  
cripto  
fiec
- criptosist pt criptare
  - metoda de hash sau MAC
  - și alte metode asociate lor

- mesajele conțin și pswm. frag!

Sender: - e o constantă -

SSL Record Protocol

- se aplică și alți de compresie, dacă e ceeașt, iar dacă nu, compresionează cu NULL.
- se adaugă MAC în cadrul
- absolut totul se cripteză
- se adaugă header SSL.

El recepte și face procesarea în sens invers - (se refac jocurile transmise)

- Cum se calculează MAC-ul:

→ Se ia cheia de MAC, se padurește, (un fel de HMAC)  
și se concatenează cu restul: se mai face hash pe tot ceeașt  
de pașmorti (vezi slide 28)

Criptare: dacă se folosește criptosistem stream, nu se mai  
face padure (generă cheia către mesaj)

- dacă folosește criptosistem bloc, trebuie să padure și  
vector de inițializare către criptorul următor.

SSL CCS - protocol

- mesajul e un sg octet, care spune cum se schimbă  
starea din currență în pending

## SSL Alert Protocol

→ transmite mesaj de alertă

## SSL Application Data Protocol

- în datele care să transmită su SSL și să transmită celuilalt protocol care să ia măsură.

## TLS

SSL standardizat.

- cu căt mai mulți intermediali, cu atât mai slabă securitate - → recomandat socket to socket.
- struct identică cu SSL.
- Modificări noastre:
  - generarea sec. master
  - utilizare HMAC în loc de MAC
  - utilizare SHA 2
  - utilizare un alt mod de inițializare
  - noi rute criptografice - bazate pe AES sau pe ECC (criptogr. puin ambele tipuri)

→ SSL 3.0 → trecere la TLS 1.0 și apoi 2.0.

3DES: criptare cu o cheie, decriptare cu o altă cheie și criptare din nou cu prima cheie.

## Securitatea poștei electronice

- Formulă securător S/MIME

MIME permite header de tipuri difcile -

Pentru transmiterea datelor - trebuie folosită de codificare: 7bit, 8bit/binary, quoted-printable, b64.

→ 7bit + puncte -  
7bit pt. text

→ 8bit text

cauză: special  
sunt codif cu ajut  
subtilitate " = "

ce  
mai  
folo  
site

base64 → se împarte textual în grupe de 6 biti.  
→ fiec. caracter de 6 biti se mapează într-un caracter printabil pe 8 biti, pe baza unei tabele (se face o extindere de la 6 la 8 biti)

### Pretty good privacy (PGP)

soft specializat - dezvoltat de Mummerson

- oferă autenticitate
- confidențialitate.

-  $A \rightarrow B$  -  $x = \text{mesaj}$ ,

$A$  semnează  $x$  (nu se generează mesajul complet pt că ar nevoie o semnătură de lungimea mesajului; se semnează doar un segment hash al mesajului); mesajul + semnatura se fac zip (archiv) -> compactare

- Generează o cheie pt un criptosistem simetric și de exemplu; pt criptat mesajul cu cheia k.

se ţine cheia k și se criptă mesajul cu cheia publică a destinatarului - Nu se criptă direct cu cheia publică pt că procesul ar fi mult mai lent  
Nu există protecție împotriva impersonificării.

### Gestionează cheile

- am nevoie de chei private pt semnat  
- și de cheie publică pt verificarea semnatului în de cheile publice ale destinatarilor.

PGP → are inele de chei în care face gestionarea acestor chei ( $\Rightarrow$  sunt 2 tabele (inele de chei))  
- un tabel pt cheile mele  
- un tabel pt cheile destinatarului

Reyid → ultimii 64 biti cei mai semnificativi ai cheii - (cheile sunt mai multe, de ex 1024 biti)

$K_p$  - cheie publică utilizator

$K_d$  - cheie privată

Semnificație: cu cheia privată, se criptază cu cheia publică - (criptare cu cheia privată, decriptare cu cheia publică).

→ ~~Cheia privată se criptă~~ - cu ajutorul parolei de utilizator - > se face hash pe  $P_k$  și se face o cheie -

- inelul de chei publice al destinatarului = în fel ca inelul de chei private ale userului, dor că nu să fie compromisă implementarea -

→ gradul de legitimitate a cheii

- gradul de încredere al semnatorului

- gradul de încredere a destinatarului.

- măsură semnatorului -

## S/MIME

- similar cu PGP - , e propus de RSA Data Security

Mesajul, se generează o  $K$ , se generează o semnătură, se semnează mesajul -

se generează un ID al alg de criptare

- ID al cheii publice al destinat. (certificatul public).

Nu se utilizează compresie ZIP.

~ se codifică în base64

## Semnătura digitală -

- se ia mesaj  $x$ , se transformă  $x$  în base 64  
 $\Rightarrow \underline{x'}$

$x'$  este semnătura și se adaugă parametrii necesari și totul se face în base 64 (se transformă în b64)

Părțile de anonimizare - parte a comunicării private  
 scheme de vot electronic  
 comert electronic

Anonimizare  $\rightarrow$  | al achitării  
 identitatea

- 'unlinkability' = presupune că nu există legătură între utilizator și achitare.

TOR - the onion Routing - rutează în "foi de ceapă"

- se pot trimite mesaje cu păsăriile anonimătății (nu stă că cine a trimis mesajul)
- Multe o mărire de noduri (mix-uri) se extrage un nr de mix-uri și se realizează un drum de la sender la destinat. Drumul e encapsulat în mesajul trimis.

In momentul în care mesajul ajunge la 1 mix, se despoilează primul mesaj și se vede că cu ce folosește mesaj.

- Astfel mai multe mesaje, care se pun să circule împreună, sunt permisă între ele, rândom; Astfel, se protejează calea doar prin acolo, și astăzi sunt retransmitute în următorul ordin, până când mesajul ajunge la destinat. A proteja scrierile de la B în mesaj. B poate răspunde la mesaj, dar fără să fie cind răspunde cum se face: A introduce în mesaj o adresa de return - o secvență de mixuri difeite

de cale initială pînă cînd mesajul este agățat  
înapoi. Cînd tráfcul e redus, se introduc mesaje  
dummy și se rădonăgește.

### Comert electronic.

#### • plată electronică

- 2 clase de tehnici

- 1) bolete pe baza hash -> valoare electrică a unui curent de cecuri -

- fiecare curent de cecuri are accesul său

2) bolete pe baza fizice rezistente la foudă

- dispozitivul care citează cardul. El face  
rezistență la foudă - (tamper-resistant)

- Protocolul SET (secure electronic transaction)

fizică do Visa și MasterCard (care l-au făcut)

~~tit~~

SUP (Small Value Payment).

### Schemă de identificare

- se utilizează în special la smartcard

- o entitate (smartcard) își demonstrează identitatea folosind de către entitate (utilizator).

- protocoale: GQ

Schnorr.

Okamoto.

### Zero-knowledge proofs -

- cardul are o inf. secretă memorată morscată  
de ex un pașum  $X$ , dorită să fie  
formă  $\alpha^X$  (pb. legătură discripție).

- cardul demonstrează că are aceea inf secretă, dorită  
fără a o dezvăluia în clor.

IPsec IP datagram

Vers 4: → payload (lucrătorul)  
    ) ip header (cu flags)

În modul transport - nu se mărește header,  
se buceafă pe payload.

~~IPv4~~ → ESP header = padare payload a î.

Impună ne blocarea payload de preluat și fie  
de 128 biti - (toate). → se impună payloadul  
în bucăți de 128 biti pt a se cripta  
→ după asta se apă autentică -.

la IPv6

- se adaugă payload, dor la criptare se criptează inclusiv headerul destinatie (dest) .
- se autentică -
- și headerul se pune în coadă. (ESP auth)

Conținut mutabil din payload nu se criptează,  
(mutabil = cele care se modifică în traiul rutorii - care contin info despre rutare)

- Tunnelare - (Tunnel Mode)

- Totul e încopiat (payload), pochetul primește un nou header.  
→ E recomandat caud unu din găzde e primit  
de securitate. (Transport Mode e în special folos  
la comunicarea între 2 găzde)

Protecția e totală, dor costurile de procesare  
ascese

AH + ESP în mod tunnel:

- Toate datagramme sunt pe autenticatoare,
- și se construiește un nou header
- Autentificarea se face pe toate datagramme (indiferent cămpurile nemutabile din header)
- La IPv6 → similar.

AH = rezultatul autentificării (auth. header)

ESP în mod tunnel

→ similar

AH pe ESP

$$HMAC(m) = H(K \oplus \text{epad}, H(K \oplus \text{ipad}, m))$$

↓  
1 singură aplicație hash

rezultat (aplicat de mai multe ori)

HMAC-SHA1-96

→ HMAC cu SHA1 restructurat la 96 bit (se iau doar primii 96 biti)

- O anotă de securitate e bidirectională - (dacă A stabilește AH cu B, el va comunica cu B prin AH, dar nu e obligatoriu și invers).
- ⚠️ AS ocărante de securitate în preluare (câte una pt fiecare entitate).

## Asociu de securitate Bundle

- asociu de securitate se alerga prin asocuri multistapă selectori de securitate.
- internet key Exchange (IKE protocol)
- protocolul stabilește asociu de securitate
- autentifică partile care comunică
- stabilește algoritmi de comunicare.  
→ moment: versiunea 3 IKE.

IKE: formă din perechi de comunicare  
pe care → o entitate trimite o cerere, celalalt răspunde.

3 perechi fundamentale: IKE-ST-init → stabilește cheie cu care se protejează toată comunicarea  
mai departe → apoi se stabilesc pași pt IKE SA.  
~~⇒ IKE SA init~~

→ se stabilește între primul SA și după  
aceea următorul se stabilește cu Child SA.

⇒ poate face re-key pe toată sesiunea (IKE SA)

sau re-key pe asociere (Child SA)

IKE SA = asociu de securitate pt IKE, nu pt IPsec.

(⇒ funcție pseudorandom e o funcție de  
funcții indexate după cheile criptografice -

Keys = SKd || SKaf || SKar || SKei || SKer || SKpi || SKpr  
↓ ↓  
autenticare comunicare  
pt payload

## IKE-AUTH

- al doilea schimb de mesaj -
- autentificare + stab. împreună asoc. de securitate care poate fi folosită la IPsec

## Create Child SA

→ pt creație sau modificare de asociere de securitate.

Re-Key = cea veche se sterge și se folosește una nouă creată.

Când se face child SA nu mai conține anumite informații comunicarea (rolurile se pot invinge).

TS; TSR - valori DH (Diffie-Hellman)

Rekey pt trage securitate IKE SA.

SKd - cheia pt securitate.

SA - oprire ce reprezintă dintre asoc. de securitate (SRA, MD5, etc. --)

N - Nounce.

KEi → valori DH.

dacă fac Rekey la IKE, nu trebuie să mai fac încă un ~~child~~ child SA pt IKE AUTH - (nu e necesar decât la autentificare).

## Informational

N - bude fac anumite

D - sterg asoc. de securitate

CP - modifică asoc. de securitate

### - IND - CPA =

nu pot distinge intre două mesaje care arună un singur criptotext.  
(de la care din cele 2 mesajuri acel criptotext)

→ cript. poate provine de la oricare din cele 2, cu probabilit. egale.

$\overset{?}{\text{IND-CCA}} \Rightarrow \text{IND-CPA}$  doar nu și invers

Văd mai bine, de preferat, deșor să e ușoară.

→ Concepte de securitate pt sisteme - { IND-CPA  
IND-CCA }.

### - Conc. de securitate pt fct HASH.

→ În practică e bine de către fct. resist la colțuri, nu one-way.

### - SA bundle -

La IPSEC o arce de securitate standardează protocolul care va fi utilizat AH sau ESP (doar unul) doar intra-o amintire directă (nu ambele).

- doar ureau să folosesc ESP de la A la B, nu nu de 2 arce de securitate.

O colecție de arce de securitate, cîntă o direcție

-(un packet) = SA bundle

### - cum lucrajă DNS sec - (PP de securitate)

→ propune și tipuri de înreg moi (+1)  
+ de bază)

ce urm. scop -

- folos. să stim formata înregistrare

### - La IPSEC - la fel

- Encapsulare datagram (fără formata lor, să cunoascem principale

EQ → challenger-response-  
zero knowledge.

### Smart card - cui

ex. Visa - portocal.

→ au un microcip capabil de operări matematice.

- autentificare biometrică : pe măre cerebrale → se creezează starea fizică a persoanei -

- trei să răspund ambele întrebări; ai ambele emprese, și în funcție de răspuns - se identifică starea emoțională -

- felinici de atac și putere :

la calculul exponentului → diferență de consum de energie pt calcul la Exponentul cu 1 sau cu 2<sup>0</sup> → se nășoară exact și se spune cu probabilitate foarte mare ce bit a fost în exponent, și sau 1, ⇒ se reface exponentul integral.

- Examen : 30 :

• lista de subiecte - mărime de ambiție.

28 - consultanță la cabinet.

CPA / IND CPA - atacă : (de două întrebări doar  
- atac de plaintext aleș).

- adăv. alese ambele plaintexte pt care vrea să obțină  
ciphertext (vrea să vadă cum se criptă un text aleș);

- dor, pe baza ac. experim. alese de el, va fi. să  
monteze un atac; Acum CPA îl însoțit de o metodă  
prin care să spunem că schema e rezist la CPA,

Stă lo SSL & TLS

stă lo S/MIME + PGP

→ Ce proprietăți de securitate are acest;

de ex. } fără integritate  
} - (nu are nicio confidențialitate)

- lo modul general să trătește.

întrebare : 1, 2, 3 -- cu mai multe pct. din

aspekte gen. cripto ; hash, proprietăți, mod  
de iterare frecvent, etc și HMAC(?)

- MAC, HMAC ...

- înlăturări în criptare : ePC, --- etc.

! (+) - Controlul accesului -

(subiect algen)

- ori o probă din motrica de control -

- ori sistemul take-giveout

→ să arătăm că un subiect nu obține un  
drept. ( să nu rezolvă jocul resp.).

alt ex: se dă un protocol de acc cu refulare:

- A → B

B → C

- se stă că protoc. poate fi alcătuit prin impresa-  
nificare : găsiti un atac.

metoda radix 64 → base64

se dă ascii și tab de  
conversie și nu facem conversie.

- se împart în grupe de 6b, să treacă în tab  
și se calc. cu grupuri de 8 b

Controlul acc : tehnici de memorare a motri  
cii de control ; tehnici de securitate (no read  
up, no write down).

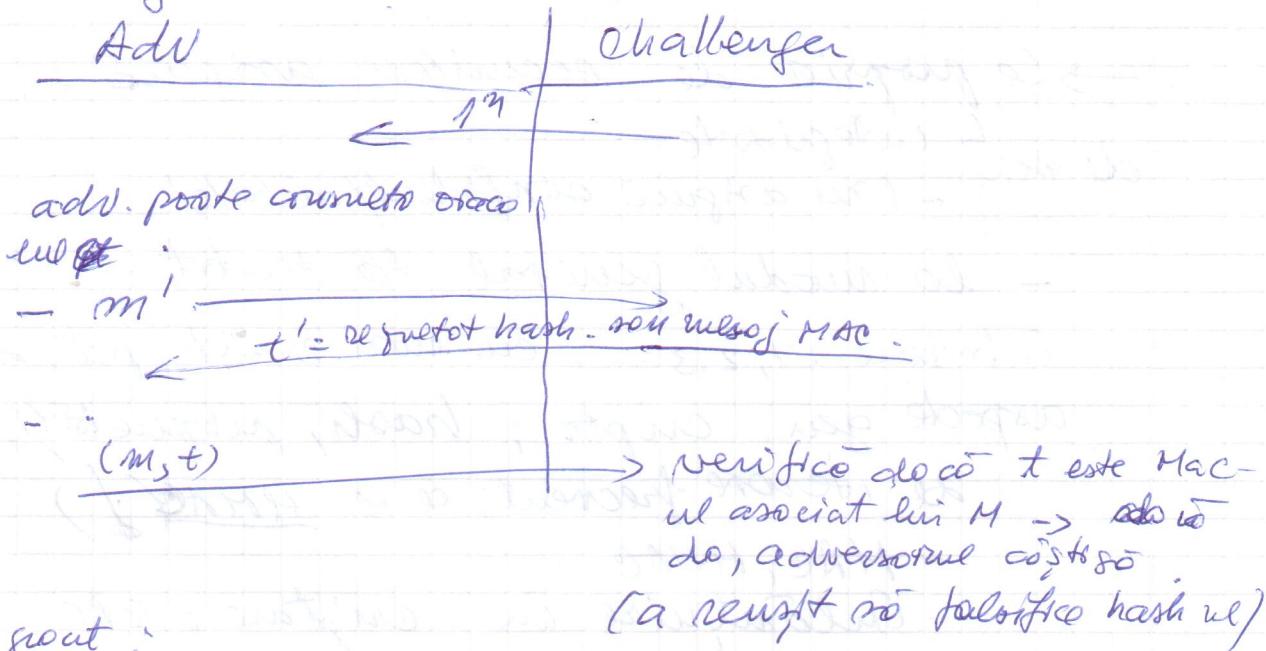
Modelul schematic nu se dă.

tehnici de monogenental taylor, except  
Diffie Helmann. -

NU

Reffents: for Entre adv of challenger.

challenge-ul transmite paron de sec 1<sup>n</sup>



doce sunt e conditi; et pot stimula massim Turja  
- de potestate care e stans cureretur si care e confirmare  
alului.

→ surgiu de debate e legado de opiniões  
magnus turing, que é proibido decidir sobre -

- compresia datelor în PGP - un dom.  
(un dom algoritmul de compresie)