Capitolul 2

Protocoale de autentificare

Procesul de autentificare este un mecanism frecvent utilizat în toate mediile. Orice comunicare începe prin stabilirea identificării partenerilor: fiecare trebuie să prezinte dovezi care să îl autentifice în faţa celuilalt (sau în faţa unui sistem care îi acordă acces la anumite resurse).

Exemplul 2.1. *Istoria cunoaşte numeroase exemple de mecanisme de autentificare.*
Astfel, unele documente arată că în China de acum 2000 ani, amprentele formau un mijloc de identificare.
Mesagerii regali din Evul Mediu foloseau ca mjloc de identificare peceţile inelelor.
Sistemul Bertillon (datând din 1870) este utilizat în SUA pentru identificarea criminalilor. Acest sistem identifică o persoană printr-un ansamblu de măsurători cum ar fi înălţimea, lungimea braţelor, grosimea toracelui, lungimea degetului inelar, mărimea piciorului, circumferinţa capului. După ce în 1903 – într-un caz celebru – sunt identificaţi doi criminali cu aceleaşi măsuri Bertilion, sistemul este înlocuit cu identificarea prin amprente.

Mecanismele de autentificare se referă atât la partenerii de dialog cât şi la mesajele transmise de aceştia.

Principala deosebire între cele două tipuri constă în faptul că autentificarea unui mesaj nu dă nici o garanţie asupra momentului creerii mesajului, pe când autentificarea unui utilizator este implicit legată de momentul solicitării acestei autentificări. Analog, autentificarea unui utilizator nu oferă nici o informaţie despre conţinutul mesajelor pe care le va gestiona acesta în calitate de entitate autorizată, pe când autentificarea unui mesaj oferă anumite date despre conţinutul mesajului.

Aproape toate protocoalele prezentate în capitolele următoare se vor referi – implicit sau explicit – la autentificări de mesaje sau de utilizatori care solicită acces la anumite drepturi.

2.1 Autentificarea utilizatorului

Sunt mai multe motive pentru care este necesară o autentificare a utilizatorului:

Controlul accesului. Tendinţa actuală este de a întări complexitatea drepturilor de acces la resursele accesibile prin reţele de calculatoare (inclusiv Internet). Controlul accesului acordă sau restricţionează dreptul unui utilizator de a accesa anumite resurse; de asemenea, el protejează resursele, limitând accesul doar pentru utilizatorii autorizaţi. Cu ajutorul calculatoarelor se poate asigura un control al accesului pentru diverse tipuri de conexiuni, pentru unele baze de date sau chiar pentru intrarea în anumite clădiri.

Autorizarea. Autorizarea este procesul prin care unui utilizator i se alocă diverse drepturi de acces. Aceste drepturi includ anumite specificaţii, cum ar fi dreptul de a citi, de a scrie sau de a actualiza un anumit fişier. De exemplu, protocoalele de plăţi electronice solicită autorizarea celor care le utilizează, pentru a preveni fraudarea tranzacţiilor. O solicitare similară apare şi în cazul protocoalelor de vot electronic.

Auditarea. Chiar dacă nu sunt prevăzute protocoale de control, este adesea recomandabil să se reţină acţiunile tuturor utilizatorilor. Astfel de arhive sunt utile în cazul unor posibile apariţii de activităţi maliţioase. Experienţa arată multe atacuri asupra sistemelor de calcul provin de la persoane autorizate chiar de sistemele respective.

Este interesantă şi situaţia reciprocă: componentele electronice trebuie de asemenea autentificate. Un exemplu simplu: în unele locaţii există ATM-uri (cititoare de carduri) false; utilizatorii merg la ele, dau cardul şi PIN-ul, iar maşina le spune că momentan nu îi poate servi. La sfârşitul zilei, baza de date construită cu informaţiile de pe carduri şi PIN-urile respective este extrasă din maşină şi utilizată pentru furtul banilor din conturi.

Observaţia 2.1. *În general există o distincţie netă între noţiunile de "autentificare" a utilizatorului şi "identificarea" sa. Protocoalele de identificare se referă la situaţia următoare: utilizatorul respectiv a fost autentificat dar el trebuie să demonstreze că are dreptul să solicite accesul la anumite resurse. Vom întâlni astfel de instrumente de identificare în cadrul capitolelor legate de protocoale electronice de plată sau de vot electronic.*
De asemenea, noţiunile numite generic "zero - knowledge" se referă la protocoale teoretice de identificare.

În general, sistemele de autentificare a utilizatorilor se construiesc cu scopul de a determina

1. Ceva ce utilizatorul **ştie**. De exemplu, o *parolă*.

   Multe sisteme solicită CNP-ul sau numele de fată al mamei pentru a autentifica un utilizator.

2. Ceva ce utilizatorul **posedă**. De obicei este vorba de un lucru – fizic sau electronic – care aparţine utilizatorului. Cheile obişnuite de la uşă sunt exemple tipice. Dar aici poate fi inclus şi un anumit soft de firmă.

3. Ceva ce utilizatorul **este** (sau îl caracterizează). Acesta este principiul *biometric* de măsurare a anumitor proprietăţi biologice ale utilizatorului. Aceste proprietăţi biometrice pot fi statice (măsurători de amprente, retină etc) sau dinamice (analiza vocii, recunoaşterea scrisului de mână etc)

2.1.1 Parole

Cele mai răspândite mijloace de autentificare sunt *parolele (passwords)*. O parolă este o secvenţă alfanumerică cunoscută doar de utilizator (care se autentifică) şi de sistemul care asigură autentificarea. Uneori este posibil ca sistemul să nu cunoască parola, dar să o poată deduce din anumite informaţii pe care le deţine despre utilizator.

În mod uzual, scenariul este următorul: *Alice* are nevoie să acceseze resursele unui sistem (de exemplu un cont, o imprimantă, o aplicaţie soft). Pentru aceasta, ea trimite sistemului o pereche *(Alice, parolă)* şi – explicit sau implicit – specifică o resursă. Parola este folosită pentru a întări identitatea lui *Alice*; similar cu cererea ca la intrarea într-o instituţie, să prezinţi un act de identitate care să ateste cine declari că eşti !). La recepţionarea perechii, sistemul verifică parola şi – dacă ea corespunde celei asociate lui *Alice* – autorizează accesul la resursă[1].

Evident, din motive de securitate, parolele trebuie să fie secvenţe pe care *Alice* le poate memora sau – eventual – deduce uşor. În acelaşi timp, ele trebuie să fie greu de ”ghicit” de către orice altă entitate exterioară. Aceste proprietăţi sunt oarecum contradictorii: ceva ce este uşor de memorat are entropie mică, deci este şi uşor de ghicit.

De asemenea, sistemul păstrează parolele sub o formă criptată sau ca amprente (folosind un standard de dispersie), pentru a evita dezvăluirea lor în cazul unui atac reuşit asupra sistemului. Un exemplu sugestiv de gestiune a parolelor a fost prezentat în [2], pag. 71-72 referitor la parolele UNIX.

Vulnerabilităţi ale parolelor

Într-un survey publicat în 1979 în Communication of the ACM (pag 594-597), Morris şi Thompson au anallizat 3289 parole colectate aleator de la angajaţii care utilizau tehnică de calcul (în special IBM) şi au găsit că 2831 (86%) din acesta erau vulnerabile deoarece:

[1]Pentru unele resurse este asignat şi un ”tichet”, care acordă un acces limitat în timp.

15	erau formate dintr-un singur caracter ASCII
72	erau formate din 2 caractere ASCII
464	erau formate din 3 caractere ASCII
477	erau formate din 4 caractere alfanumerice
706	erau formate din 5 litere, de acelaşi tip (mari sau mici)
605	erau formate din 6 litere, toate mici.
	(de remarcat că $26^6 = 309M$, o dimensiune relativ mică).

Alte vulnerabilităţi ale parolelor includ în general:
- *Cuvinte uzuale.* Engleză, română sau alte limbi.
- *Nume cunoscute.* Nume de vedete, animale, prieteni, membri ai familiei, porecle.
- *Informaţii uşor de obţinut.* Date de naştere, numere de telefon, numărul maşinii etc.
- *Secvenţe de tastatură.* Ceva de genul "qwerty".
- *Permutări ale celor de sus.* De obicei scrieri inverse (în oglindă).

Sunt diverse recomandări privind construcţia şi păstrarea parolelor. De asemenea, securitatea unui sistem este mai bună dacă parolele se schimbă frecvent (cel puţin odată pe an).

Parole one-time

Parolele *one - time* sunt parole care se folosesc o singură dată. Acest procedeu asigură o mai bună securitate în faţa atacurilor prin forţă brută. Există diverse strategii de a asigura astfel de parole. Astfel:

- Unele sisteme oferă utilizatorului o listă de parole. Atunci când utilizatorul foloseşte o parolă, sistemul verifică dacă este în această listă. În caz afirmativ, autentifică intrarea şi – în paralel – elimină parola din listă.
  O variantă foloseşte o tabelă în care elementele sunt perechi *întrebare/răspuns.* Aici utilizatorul solicită autentificarea, sistemul alege aleator o întrebare şi aşteaptă răspunsul. Dacă acest răspuns corespunde întrebării respective, utilizatorul este autentificat, iar sistemul elimină perechea respectivă din tabelă (deci ulterior acea întrebare nu mai este folosită).

- Parole actualizate secvenţial: iniţial există o singură parolă (secretă). În timpul autentificării folosind parola α, utilizatorul generează şi transmite sistemului o nouă parolă $\alpha' = e_K(\alpha)$, unde K este o cheie derivată din α. Pentru următoarea comunicare, α este înlocuită cu α'. Metoda – deşi promiţătoare – devine dificilă atunci când apar întreruperi de comunicaţie.

- Parole bazate pe funcţii neinversabile. Par cele mai eficiente tipuri de parole *one - time*. Cea mai cunoscută strategie de construcţie de astfel de parole este schema Lamport.

În acest protocol este folosită o funcţie neinversabilă ϕ. În faza de iniţializare a schemei, *Alice* efectuează următoarele operaţii:

1. Alege un mesaj secret w şi un număr n de autentificări bazate pe w (uzual $n = 100$ sau $n = 1000$).
2. Transferă lui *Bob* printr-un canal sigur valoarea $w_0 = \phi^n(w)$.
3. *Bob* iniţializează un counter $i_{Alice} := 1$.

În timpul autentificării pentru deschiderea sesiunii cu numărul i, se procedează astfel:

1. *Alice* calculează $w_i = \phi^{n-i}(w)$ şi trimite lui *Bob* tripletul
$$(Alice, i, w_i)$$
(calculul lui w_i se poate face la fiecare nouă sesiune, sau se poate deduce dintr-o tabelă construită iniţial, odată cu operaţiile efectuate la determinarea lui w_0).
2. *Bob* verifică dacă $i = i_{Alice}$ şi dacă $\phi(w_i) = w_{i-1}$. Dacă ambele relaţii sunt verificate, *Bob* acceptă autentificarea, salvează w_i pentru verificarea următoare şi incrementează contorul: $i_{Alice} := i_{Alice} + 1$.

Ca o variantă a protocolului Lamport, *Alice* poate deţine o parolă α dată de *Bob*. Atunci când doreşte autentificarea, *Alice* trimite o pereche $(r, h(r\|\alpha))$, unde r este o secvenţă binară, iar h este o funcţie de dispersie. *Bob* face verificarea calculând $h(r\|\alpha)$ şi comparând rezultatul cu al doilea element al perechii primite. Pentru a elimina atacurile unui adversar activ, r trebuie să fie un element de tip nonce (utilizabil o singură dată).

2.1.2 Măsurători biometrice

Măsurătorile biometrice folosesc pentru autentificare caracteristicile fizice ale unei persoane. În general ele nu sunt folosite drept parole, deoarece – odată compromise, nu pot fi înlocuite. Cele mai utilizate măsurători biometrice utilizate pentru autentificarea unei persoane sunt: Recunoaşterea vocii, Dinamica semnăturii, Amprente, Geometria mâinii, Scanarea retinei, Scanarea irisului, Modelul facial sau alte caracteristici specifice.

La construirea unui sistem de autentificare bazat pe măsurători biometrice trebuie ţinut cont de timpul necesar acestor măsurători, de preţul aparaturii, de gradul de acceptare al utilizatorului de a se expune măsurătorilor biometrice, de ratele de eroare privind falsa - acceptare (a unui alt utilizator decât cel legal) şi falsa - respingere (rejectarea utilizatorului legal); de obicei, aceste două rate se consideră egale, deşi în realitate ele depind în primul rând de gradul de performanţă al aparaturii folosite.

Un studiu realizat în 2000 de Sandia National Labs (SUA) compară aceste modalităţi de autentificare a utilizatorului. Pe scurt, ele pot fi sumarizate în tabelele următoare:

Tehnica	**Rata de eroare**
Voce (analiză Alpha)	3%
Voce (analiză ECCO)	2%
Semnătură	2%
Scanarea retinei	0.4%
Geometria mâinii	0.1%
Amprente	9% falsa - respingere, 0% falsa - acceptare

Caracteristici	**Cea mai bună**	**Cea mai slabă**
Acceptarea utilizatorului	Mâna	Voce
Falsa - respingere	Mâna	Amprente
Falsa - acceptare	Mâna, retina, amprente	Voce
Mulţime detalii	Mâna, retina, amprente	Voce, semnătura
Dificultatea imitării	Retina	Voce, semnătura
Cost	Voce	Retina

Datorită ratei mari de eroare, măsurătorile biometrice nu constituie un avantaj faţă de parole. De obicei aceste două tipuri de autentificare se folosesc combinat (deci sistemele verifică atât ceea ce utilizatorul "*este*" cât şi ceea ce utilizatorul "*ştie*").

În secţiunile următoare ale acestui capitol ne vom ocupa de protocoale de autentificare a mesajelor. Vom face acest lucru separat de autentificarea utilizatorului care trimite mesajul (autentificarea ambelor entităţi se realizează folosind semnătura electronică, modalitate care va fi prezentată în capitolul următor).

De asemenea, autentificarea unui mesaj nu înseamnă totdeauna şi integritatea lui (deşi în majoritatea protocoalelor, acest lucru se subînţelege).

Sunt domenii de securitate a informaţiei – de care ne vom ocupa în alte volume – dedicate special problemelor de autentificare şi integritate a mesajelor. Amintim în acest sens Watermarking şi Fingerprint. Acum, în această etapă, vom discuta doar autentificarea prin Coduri de Autentificare a Mesajelor (MAC) şi prin canale subliminale.

2.2 MAC

Un cod de autentificare a mesajului (*MAC – Message Authentication Code*) este o combinaţie între o funcţie de compresie şi o cheie. El este folosit pentru a autentifica mesajul, asigurând în acelaşi timp şi certificarea integrităţii sale.
Codurile de autentificare a mesajelor sunt utilizate pe scară largă în protocoale legate de comunicarea pe Internet, cum sunt IPSec sau SSL/TLS.

Atunci când *Alice* trimite lui *Bob* un mesaj α (criptat sau nu), ea va construi un cod $MAC(\alpha)$ folosind o cheie şi funcţie de compresie cunoscute de ambii parteneri, după care va trimite perechea

$$(\alpha, MAC(\alpha))$$

La recepţia unei perechi (α, x), *Bob* calculează $MAC(\alpha)$ şi vede dacă acesta coincide cu x. În caz afirmativ, el va considera mesajul α ca fiind autentic.

Definiţia 2.1. *O pereche (α, x) cu $x = MAC(\alpha)$ se numeşte "pereche validă".*

De remarcat că atât *Alice* cât şi *Bob* pot calcula un MAC valid pentru un mesaj α.

2.2.1 $HMAC$

Dacă drept funcţii de compresie sunt folosite funcţii de dispersie, codurile de autentificare a mesajelor se numesc coduri $HMAC$ (Hash MAC).

Exemplul 2.2. *Unul din primele HMAC-uri a fost propus de IBM pentru mesajele trimise pe Internet. El este construit în felul următor:*
Fie K cheia secretă folosită; ea este partajată în două subchei $K = (k_1, k_2)$.
Pentru fiecare bloc de text clar α, codul de autentificare va fi

$$MAC(\alpha) = MD5(k_1 \| MD5(k_2 \| \alpha))$$

unde $MD5$ este funcţia de dispersie prezentată în Capitolul 1.

Codul $HMAC$ prezentat în continuare a fost construit de M. Bellare, R. Canetti şi H. Krawczyk, fiind cunoscut drept standardul RFC 2104 ([40]). El este de fapt o extensie a codului din Exemplul 2.2

Fie h o funcţie de dispersie criptografică care procesează mesaje de n octeţi şi produce rezumate de p octeţi (dacă h este $SHA-1$ atunci $n = 64$ şi $p = 20$).

Se mai defineşte un parametru t $(4 < t < p)$ care reprezintă numărul de octeţi din $HMAC$.
Dacă x este blocul de intrare şi K este cheia folosită, calculul lui $HMAC(x)$ este realizat de algoritmul următor:

1. Dacă $|K| > 8n$, se înlocuieşte K cu $h(K)$.
2. Se completează K la dreapta cu biţi $'0'$ până ajunge la n octeţi.
3. Se calculează
$$A = h(K \oplus opad \| h(K \oplus ipad \| x))$$
unde $ipad, opad \in \{0,1\}^{8n}$, $ipad = (36\ldots36)_{16}$, $opad = (5C\ldots5C)_{16}$.
4. $HMAC_K(x)$ este format din primii t octeţi din A.

Pentru a analiza securitatea acestui cod de autentificare, vom defini noţiunea de *MAC sigur* (pentru mesaje de o anumită lungime).

Este evident că obiectivul unui atac (produs de *Oscar*) este de a produce o pereche (α, x) validă pentru o cheie K (necunoscută dar fixată).
Printr-un atac cu text clar ales, *Oscar* va solicita MAC-uri pentru mesajele $\alpha_1, \alpha_2, \ldots, \alpha_n$ alese de el. Teoretic, putem considera un oracol ("black box") care răspunde la cererile lui *Oscar*, dând MAC-urile respective.

Deci, acesta va dispune de o listă de perechi valide

$$(\alpha_1, x_1), (\alpha_2, x_2), \ldots (\alpha_n, x_n)$$

generate cu o cheie necunoscută K.

Ulterior, când *Oscar* va emite o pereche (α, x), se presupune că $x \notin \{x_1, x_2, \ldots, x_n\}$. Dacă perechea emisă este validă, atunci spunem că este un *fals*.

Un MAC pentru care probabilitatea de a obţine un fals este neglijabilă se numeşte *MAC sigur*.

Pe baza acestor noţiuni, securitatea codului HMAC construit anterior este demonstrată cu ajutorul rezultatului următor:

Teorema 2.1. *Fie $h : \{0,1\}^* \longrightarrow \{0,1\}^p$ o funcţie de dispersie criptografică. Fiind date două chei $K_1, K_2 \in \{0,1\}^p$, considerăm algoritmul MAC definit prin*

$$MAC_{K_1,K_2}(x) = h(K_2 \| h(K_1 \| x))$$

Dacă aplicaţia MAC_{K_2} definită $MAC_{K_2}(x) = h(K_2 \| x)$ este un algoritm MAC sigur pentru mesaje x, $|x| = p$, atunci MAC_{K_1,K_2} este un algoritm MAC sigur pentru mesaje de lungime arbitrară.

Demonstraţie. ([69]). Să presupunem că avem un oracol $\mathcal{O}$ care dă valoarea $MAC_{K_2}(x)$ pentru orice x cu $|x| = p$, şi un adversar *Oscar* pentru MAC_{K_1,K_2}; vom construi un adversar pentru MAC_{K_2} în felul următor:

1. Generăm aleator K_1 şi simulăm un oracol pentru *Oscar*.

2. De câte ori *Oscar* trimite spre oracol un α_i, calculăm $h(K_1 \| \alpha_i)$ şi-l trimitem spre $\mathcal{O}$; acesta dă un răspuns x_i, pe care îl returnăm lui *Oscar* ca răspuns la cererea sa. Aceste procedeu se repetă pentru $i = 1, 2, \ldots, n$ (n o valoare arbitrară).

3. Când *Oscar* termină anunţând că a găsit o pereche falsă (α, x), calculăm $h(K_1 \| \alpha)$ şi-l trimitem spre $\mathcal{O}$. Dacă acesta răspunde cu una din valorile $x_1, \ldots, x_n$ înseamnă că am găsit o coliziune pentru funcţia de dispersie h – lucru imposibil, deoarece aceasta este o funcţie criptografică (deci cu coliziuni tari). Rezultă că $h(K_1 \| \alpha)$ este diferit de toate valorile $h(K_1 \| \alpha_i)$ calculate anterior. Notând $\beta = h(K_1 \| \alpha)$, tragem concluzia că am găsit o pereche falsă (β, x) cu $|\beta| = p$; lucru imposibil deoarece MAC_{K_2} este – prin ipoteză – un MAC sigur pentru mesaje de lungime p.

$\square$

2.2.2 $CBC - MAC$

Cea mai simplă modalitate de a construi un cod de autentificare al unui mesaj α este de a folosi un sistem de criptare simetric implementat în modul CBC şi de a utiliza ultimul bloc obţinut prin criptarea lui α drept $MAC(\alpha)$.
Metoda este numită $CBC - MAC$ şi a fost prezentată în [2], pag. 71 (folosind sistemul DES). Să o reamintim:

1. Textul clar x se împarte în blocuri de lungime fixată: $x = \alpha_1\alpha_2 \ldots \alpha_n$.

2. Se construieşte secvenţa $\beta_1, \beta_2, \ldots, \beta_n$ după formula

$$\beta_i = e_K(\beta_{i-1} \oplus \alpha_i) \qquad (i \geq 1)$$

   unde $\beta_0 = 00\ldots 0$, iar e_K este funcţia de criptare cu cheia K.

3. $MAC(x) = \beta_n$.

Metoda este foarte rapidă şi uşor de realizat. Ea prezintă însă unele slăbiciuni.
Astfel, să presupunem că ştim trei perechi valide (x_1, c_1), (x_2, c_2), (x_3, c_3). Mai mult, x_1 şi x_3 sunt mesaje de aceeaşi lungime, iar x_1 este un prefix propriu al lui x_2; deci putem scrie $x_2 = x_1\|\alpha\|x_2{}'$, unde α este un bloc.

Blocul criptat obţinut – în calculul lui c_2 – după criptarea lui α este $e_K(\alpha \oplus c_1)$.

Să notăm $\alpha' = \alpha \oplus c_1 \oplus c_3$ şi $x_4 = x_3\|\alpha'\|x_2{}'$.

În calculul MAC-ului c_4 (pentru x_4), după criptarea lui α' se obţine

$$e_K(c_3 \oplus \alpha') = e_K(\alpha \oplus c_1),$$

după care criptările pentru c_2 şi c_4 merg similar (urmând după blocul α'). Deci, în final $c_4 = MAC(x_4) = c_2$ şi se poate construi o pereche validă (x_4, c_2).

O variantă este codul de autentificare $EMAC$ (*Encrypted MAC*), obţinut prin criptarea cu altă cheie K' a ultimului bloc criptat β_n.
În acest caz, securitatea este egală cu rezistenţa lui CBC la un atac bazat pe paradoxul naşterilor.
Într-adevăr, să presupunem că *Oscar* obţine o coliziune: două perechi valide (x_1, c), (x_2, c) cu acelaşi MAC. El ia un mesaj arbitrar $x_3{}'$ şi solicită un cod de autentificare pentru mesajul $x_3 = x_1\|x_3{}'$. Să presupunem că primeşte $MAC(x_3) = c'$. Atunci $(x_2\|x_3{}', c')$ este o pereche validă.

În [69] se arată că probabilitatea de a obţine o coliziune din $t \cdot \sqrt{N}$ perechi valide (unde N este numărul de MAC-uri posibile) este $1 - e^{-t^2/2}$.

Deci – pentru a rezista la un astfel de atac – modul CBC de implementare utilizat pentru construirea unui $EMAC$ trebuie să fie similar unei funcţii de dispersie criptografică.

O modalitate simplă de a realiza acest lucru constă în eliminarea unor biţi din MAC; de exemplu trunchierea rezultatului la prima jumătate; acesta este de fapt standardul ISO/IEC 9797 (stabilit în 1989) de obţinere a codului de autentificare bazat pe algoritmi simetrici de criptare.

OMAC

OMAC (*One-key CBC MAC*) este un standard care operează cu mesaje a căror lungime nu este obligatoriu un multiplu al lungimii blocului de criptare. El lucrează cu un sistem simetric $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, având lungimea blocurilor criptate egală cu p, o cheie $K \in \mathcal{K}$, o familie $\mathcal{A}$ de constante şi două constante $C_1, C_2 \in \mathcal{A}$. Se mai defineşte o valoare t ($t < p$) ca fiind lungimea codului de autentificare.
Fie familia de funcţii

$$\mathcal{H} = \{H_L \mid L \in \mathcal{C},\ \ H_L : \mathcal{A} \longrightarrow \mathcal{C}\}$$

Să considerăm textul clar $x = \alpha_1\|\alpha_2\|\ldots\|\alpha_n$ cu $|\alpha_i| = p$ pentru $i < n, \quad |\alpha_n| \leq p$. $OMAC(x)$ se defineşte după următorul algoritm:

1. Fie $L = e_K(00\ldots0)$. Se determină $H_L(C_1)$ şi $H_L(C_2)$.
   (Acest pas poate fi preprocesat pentru o cheie dată K, el nedepinzând de mesajul x).
2. Dacă $|\alpha_n| < p$, α_n se completează cu bitul $'1'$ urmat – eventual – de un şir arbitrar de biţi, până ajunge la lungimea p. Spunem că α_n ”a fost completat”.
3. $\alpha_n := \begin{cases} \alpha_n \oplus H_L(C_1) & \text{dacă } \alpha_n \text{ nu a fost completat} \\ \alpha_n \oplus H_L(C_2) & \text{dacă } \alpha_n \text{ a fost completat} \end{cases}$
4. Se determină CBC MAC-ul pentru $\alpha_1\|\alpha_2\|\ldots\|\alpha_n$.
5. $OMAC(x)$ constă din primii t biţi din acest MAC.

Versiunea (curentă) $OMAC1$ este implementată în felul următor:
Se consideră $\mathcal{P} = \mathcal{C} = \{0,1\}^p$, unde $p = 64$ sau $p = 128$. Toate toate operaţiile se efectuează în $GF(2^p)$, inel structurat astfel: fiecare $\alpha = a_1a_2\ldots a_p \in Z_2^p$ este asociat cu polinomul $\alpha(X) = a_1X^{p-1} + \ldots + a_{p-1}X + a_p \in GF(2^p)$. Operaţiile pe $GF(2^p)$ sunt definite:
- adunarea $\alpha + \beta = \gamma$ este un XOR bit-cu-bit: dacă $\alpha = a_1\ldots a_p,\ \beta = b_1\ldots b_p,\ \gamma = c_1\ldots c_p$, atunci $c_i = a_i \oplus b_i$ $(1 \leq i \leq p)$.
- înmulţirea $\alpha' = \alpha \cdot X$ se face după algoritmul

1. Se ia vectorul α.
2. Se elimină primul bit (din stânga) şi se inserează la sfârşit bitul $'0'$; fie β vectorul obţinut.
3. Dacă bitul eliminat a fost $'1'$, atunci $\alpha' = \begin{cases} \beta \oplus 00\ldots1B & \text{dacă } p = 64 \\ \beta \oplus 00\ldots87 & \text{dacă } p = 128 \end{cases}$

   altfel $\alpha' = \beta$.

Funcţia $H_L(x)$ se defineşte $H_L(x) = L \cdot x$, iar constantele $C_1 = 00\ldots 2$, $C_2 = 00\ldots 4$ (în $GF(2^p)$, C_1 corespunde polinomului X, iar C_2 – polinomului X^2). Deci

$$H_L(C_1) = L \cdot X, \quad H_L(C_2) = H_L(C_1) \cdot X.$$

2.2.3 Modul autentificat de operare

După cum am văzut, un MAC asigură autentificarea şi integritatea unui mesaj. Nu s-a pus problema confidenţialităţii, mesajul în discuţie fiind un text clar.
Uneori însă este nevoie ca mesajele criptate să fie protejate prin algoritmi de autentificare şi integritate. Acest lucru este realizat de obicei combinând operaţia de criptare cu cea de construire a unui MAC; procedeul este numit *Mod autentificat de operare* (Authenticated Mode of Operation).

Sunt cunoscute diverse astfel de protocoale. Unul dintre cele mai frecvent folosite în acest moment este numit CCM (Counter with CBC-MAC) şi este o combinaţie cu AES (sau alt sistem simetric de criptare pe blocuri de 128 biţi).

Mai exact, fiind dat un mesaj α, se determină $T = MAC(\alpha)$ şi apoi se criptează $T\|\alpha$.

Pentru construcţia unui CCM folosesc:

- O cheie K a sistemului de criptare bloc pe 128 biţi.
- Doi parametri:
  M – mărimea (în octeţi) a lui T; M este un număr par în intervalul $[4, 16]$;
  L – mărimea (în octeţi) a zonei care codifică lungimea mesajului α; L este un număr în intervalul $[2, 8]$.
  M şi L sunt codificate pe câte 3 biţi fiecare, anume reprezentarea în binar a lui $(M-2)/2$ respectiv $L-1$. Valoarea $L = 1$ este rezervată (pentru aplicaţii viitoare, când lungimea mesajelor va depăşi 256^8 octeţi – cam 2^{34} GB).
- Un nonce[2] N de $15 - L$ octeţi.
- O valoare a de autentificare suplimentară (de exemplu un număr într-o secvenţă dintr-o sesiune de lucru, sau headerul unui pachet de date).
- $flag_1$: un octet definit
  $$flag_1 = 0\|adata\|M\|L$$
  unde $adata$ este un bit setat pe $'0'$ dacă şi numai dacă $|a| = 0$.

În prima etapă se calculează un CBC MAC pentru α în felul următor:

[2] Număr generat aleator şi folosit o singură dată.

1. Se determină blocul (de 128 biţi)
$$B_0 = flag_1\|N\||\alpha|$$
2. Se descompune α în blocuri de 128 biţi: $\alpha = B_1\|B_2\|\ldots\|B_n$
(eventual ultimul bloc se completează cu zero la sfârşit).
3. Dacă $adata = 1$, se construiesc câteva blocuri $B_0^1, \ldots B_0^r$ formate din $|a|$ urmat de a, completate apoi cu 0 până la un multiplu de 128 biţi.
4. Se calculează CBC MAC-ul mesajului $B_0\|B_0^1\|\ldots\|B_0^r\|B_1\|\ldots\|B_n$.
5. Se reţin în T primii M octeţi ai rezultatului.

În etapa a doua (de criptare) se procedează după algoritmul următor:

1. Se construiesc blocurile (de numărare) $A_0, A_1, \ldots$ definite
$$A_i = flag_2\|N\|i$$
unde numărul i este codificat pe L octeţi, iar $flag_2$ este un octet cu L pe primii 3 biţi şi 0 în rest.
Numărul acestor blocuri depinde de $|\alpha|$.
2. T se criptează în maniera sistemelor fluide de criptare (sae face un XOR între T şi primii M octeţi din $e_K(A_0)$).
3. α se criptează prin XOR-are cu primii $|\alpha|$ octeţi din $e_K(A_1)\|e_K(A_2)\|\ldots$
4. Mesajul final este concatenarea celor două texte criptate obţinute anterior.

2.3 Canale subliminale

Noţiunea de *canal subliminal* a fost introdusă de Simmons în 1984 ([63]); acesta a plecat în consideraţiile sale de la **problema prizonierilor**:

Doi complici la o crimă au fost arestaţi şi închişi în celule separate. Singura lor modalitate de comunicare este folosirea de mesaje scrise şi transmise prin curieri, despre care toată lumea ştie că sunt agenţi ai gardienilor.

Gardienii permit schimbul de mesaje atât pentru a avea controlul comunicaţiilor, cât şi în speranţa că pot păcăli pe cel puţin unul din deţinuţi să accepte un mesaj conceput (sau cel puţin unul modificat) de ei. În plus, deoarece există suspiciunea că deţinuţii *plănuiesc o evadare, toate mesajele sunt citite şi se transmit numai dacă conţinutul lor este considerat inocent.*

Cei doi deţinuţi sunt conştienţi de aceste riscuri. Ei trebuie însă să accepte comunicarea în orice condiţii, pentru a-şi putea coordona planul de evadare. Singura lor posibilitate este de a înşela gardienii, stabilind un mod secret de comunicare în cadrul mesajelor aflate sub control.
Deoarece ştiu că se pot introduce mesaje false sau se pot modifica mesaje, arestaţii vor lua în considerare numai mesaje pe care le pot autentifica.

Canalul subliminal este deci un canal ascuns (*cover channel*) care permite transmiterea de mesaje destinate numai celor autorizaţi.

Exemplul 2.3. *Să presupunem că deţinuţii cad de acord ca textele transmise să ascundă mesaje în modul următor: Din fiecare frază se păstrează doar cuvintele aflate pe poziţii impare. Din acestea, fiecare cuvânt de lungime pară reprezintă bitul* $'0';$*, iar fiecare cuvânt de lungime impară – bitul* $'1'$*. În acest mod, propoziţia*

"De treci codrii de aramă, de departe vezi albind, Şi auzi mândra glăsuire a pădurii de argint"

reprezintă mesajul 001100010.

Exemplul 2.4. *Să considerăm un canal ascuns elementar, care asigură transmiterea mesajelor de un bit. Atât deţinuţii cât şi gardienii sunt de acord cu transmiterea de criptograme de trei caractere binare. Mulţimea textelor criptate posibile poate fi partiţionată în două submulţimi:*

$$C_0 = \{000, 011, 101, 110\}, \qquad C_1 = \{111, 010, 100, 001\}$$

Deci în C_0 *se află secvenţele de pondere pară, iar în* C_1 *– cele de pondere impară. Orice secvenţă din* C_i *reprezintă criptarea bitului* i $(i = 0, 1)$*. Acest lucru îl cunosc atât deţinuţii cât şi gardienii.*

Deţinuţii cad de acord în prealabil asupra a două secvenţe particulare (câte una din fiecare mulţime) care să reprezinte caracterul ales de ei. Această convenţie constituie cheia secretă.

De exemplu, dacă sunt alese 011 *pentru mesajul* 0 *respectiv* 010 *pentru* 1*, în orice mesaj trimis de deţinuţi, doar aceste două secvenţe vor avea valoare pentru ei; celelalte sunt fără semnificaţie.*

Astfel, textul criptat

$$100, \ 010, \ 000, \ 011, \ 101$$

va reprezenta pentru gardieni 11000*, iar pentru deţinuţi – care păstrează doar al doilea şi al patrulea triplet –* 10.

O încercare a gardienilor de a trimite un mesaj fals are probabilitate de reuşită de 25%*. La fel, încercarea de modificare a mesajului (adică alegerea unui triplet din mulţimea complementară) are aceeaşi probabilitate de succes.*

De remarcat că un astfel de tip de comunicare asigură autentificarea mesajelor, nu şi confidenţialitatea: gardienii cunosc mulţimile C_0, C_1 precum şi semnificaţiile lor.

2.3.1 Canalul subliminal Simmons

Construcţia acestui canal de comunicaţie se bazează pe problema factorizării.

Fie p, q, r numere prime mari şi $n = p \cdot q \cdot r$. Pentru un număr $a \in Z_n^*$, conform teoremei chineze a resturilor, ecuaţia

$$x^2 \equiv a \quad (mod\ n)$$

are 0 sau 8 soluţii.

Canalul ascuns Simmons – pentru transmiterea unui bit m – este:

1. *Alice* şi *Bob* cad de acord în prealabil asupra a unei perechi de numere $(i, j),\quad 1 \le i < j \le 8$. Valoarea i este asociată lui $'0'$, iar j este asociată lui $'1'$.
2. *Alice* vrea să transmită mesajul criptat $a \in Z_n^*$, ales astfel ca congruenţa $x^2 \equiv a\ (mod\ n)$ să aibă 8 rădăcini.
   Ordonează crescător aceste rădăcini; fie acestea $a_1, a_2, \ldots, a_8$.
3. Dacă *Alice* intenţionează să transmită mesajul ascuns $'0'$, va trimite lui *Bob* perechea (a, a_i); dacă vrea să trimită mesajul ascuns $'1'$, va expedia (a, a_j).
4. La primirea perechii (a, b), *Bob* rezolvă sistemul
   $$x^2 \equiv a\ \ (mod\ p), \quad x^2 \equiv a\ (mod\ q), \quad x^2 \equiv a\ (mod\ r)$$
   şi află cele 8 soluţii ale congruenţei $x^2 \equiv a\ (mod\ n)$.
5. Ordonează aceste soluţii şi vede pe ce poziţie este b; dacă b este pe poziţia i, va obţine mesajul clar $m = 0$; dacă este pe poziţia j, va obţine $m = 1$. Oricare din celelalte valori nu vor duce la un text clar autentificat.

De remarcat că încercarea de a rezolva direct ecuaţia $x^2 \equiv a\ (mod\ n)$, fără a factoriza pe n, conduce la problema resturilor pătratice (problemă $\mathcal{NP}$ - completă).

Exemplul 2.5. *Fie* $p = 3,\ q = 5,\ r = 7$*; deci* $n = 105$. *Alice alege* $a = 64$. *Ecuaţia* $x^2 \equiv 64\ (mod\ 105)$ *este echivalentă cu sistemul de congruenţe:*

$x^2 \equiv 64\ (mod\ 3)$ *are soluţiile* 1 *şi* 2*;*
$x^2 \equiv 64\ (mod\ 5)$ *are soluţiile* 2 *şi* 3*;*
$x^2 \equiv 64\ (mod\ 7)$ *are soluţiile* 1 *şi* 6*.*

Combinând aceste soluţii – cu teorema chineză a resturilor – se obţin cele 8 *soluţii ale ecuaţiei iniţiale (ordonate crescător):*

$$8, \quad 13, \quad 22, \quad 43, \quad 62, \quad 83, \quad 92, \quad 97$$

Dacă valorile secrete alese de Alice şi Bob sunt $i = 1, \ j = 4$, *atunci mesajul* $(64, 8)$ *va însemna pentru Bob autentificarea lui* 64 *îmnpreună cu textul clar ascuns* $m = 0$, *iar* $(64, 43)$ *– autentificarea lui* 64 *împreună cu textul clar ascuns* $m = 1$.

Celelalte perechi $(64, 13), \ (64, 62), \ (64, 83), \ (64, 92), \ (64, 97)$ *sunt respinse de Bob ca neautentice.*

2.3.2 Canalul subliminal Ong - Schnorr - Shamir

În această schemă, *Alice* şi *Bob* dispun de un număr p mare (eventual prim) şi o cheie secretă $k, \ (k, p) = 1$ – cunoscută doar de cei doi parteneri.

Dacă *Alice* doreşte să trimită lui *Bob* un mesaj subliminal $y \in Z_p$ folosind textul criptat $x \in Z_p, \ (x, p) = 1, \ (y, p) = 1$, se va folosi schema următoare:

1. *Alice* calculează "autentificatorii"

$$\alpha = \frac{\frac{x}{y} + y}{2} \ (mod\ p), \qquad \beta = k \cdot \frac{\frac{x}{y} - y}{2} \ (mod\ p)$$

2. Trimite lui *Bob* tripletul (x, α, β).

3. *Bob* calculează (pentru autentificare)

$$x' = \alpha^2 - \frac{\beta^2}{k^2} \ (mod\ p)$$

Dacă $x' = x$, mesajul este autentic.

4. *Bob* află mesajul subliminal folosind formula

$$y = \frac{x}{\alpha + \frac{\beta}{k}} \ (mod\ p)$$

Exemplul 2.6. *Fie* $p = 15$ *şi* $k = 2$*; deci* $k^{-1} = 8 \ (mod\ 15)$.

Să presupunem că Alice doreşte să trimită lui Bob mesajul subliminal $y = 4$, *folosind textul criptat* $x = 13$.

La primul pas, ea calculează $\alpha = 13, \ \beta = 3$ *şi trimite apoi lui Bob tripletul* $(13, 13, 3)$.

La recepţie, Bob determină $x' = 13$*; deoarece* $x' = x$, *mesajul este autentic, aşa că se trece la calculul mesajului subliminal, pe baza formulei de la pasul* 4.

Dacă Bob ar fi primit tripletul $(7, 2, 4)$*, la pasul* 3 *ar fi obţinut valoarea* $x' = 8$*, care este diferită de* $x = 7$*. Deci mesajul nu este autentic.*

2.3.3 Canalul subliminal *El Gamal*

În sistemul de autentificare *El Gamal*, *Alice* alege un număr prim mare q şi un element primitiv $\alpha \in Z_q$. Valorile q şi α sunt publice.

Printr-un canal ascuns, *Alice* şi *Bob* stabilesc un număr $p \in Z_q^*$.

Să presupunem acum că *Alice* vrea să trimită lui *Bob* mesajul subliminal $y \in Z_q$, folosind textul criptat $x \in Z_q$. Protocolul dintre cele două părţi este următorul:

1. *Alice* calculează $\beta = \alpha^y \pmod q$.
2. Determină γ ca soluţie a ecuaţiei
$$x = p \cdot \beta + y \cdot \gamma \quad (mod\ (q-1))$$
3. Trimite lui *Bob* tripletul (x, β, γ).
4. *Bob* calculează
$$a = (\alpha^p)^\beta \cdot \beta^\gamma \quad (mod\ q)$$
5. Dacă $a \equiv \alpha^x \pmod q$, atunci mesajul este autentic.
6. *Bob* află mesajul subliminal
$$y = \frac{x - p \cdot \beta}{\gamma} \quad (mod\ (q-1))$$

Consistenţa congruenţei de la pasul 5 se obţine prin calculul:
$$a = (\alpha^p)^\beta \cdot \beta^\gamma \quad (mod\ q) = \alpha^{x-y\cdot\gamma} \cdot \alpha^{y\cdot\gamma} = \alpha^x \pmod q$$

Exemplul 2.7. *Să considerăm* $q = 11$ *şi* $\alpha = 2$ *un generator al lui* Z_{11}*.*
Presupunem că Alice şi Bob stabilesc drept cheie secretă $k = 5$*.*

Dacă Alice vrea să trimită lui Bob mesajul subliminal $y = 9$ *folosind textul criptat* $x = 5$*, ea va determina întâi*

$$\beta = \alpha^y = 2^9 = 6 \pmod{11}$$

apoi va rezolva ecuaţia

$$5 = 8 \cdot 6 + 9 \cdot \gamma \quad (mod\ 10)$$

cu soluţia $\gamma = 3$*.*

Deci tripletul trimis este $(5, 6, 3)$.
La recepţie, Bob calculează $a = (\alpha^p)^\beta \cdot \beta^\gamma \pmod{q} = (2^8)^6 \cdot 6^3 = 10 \pmod{11}$
şi $\alpha^x = 2^6 = 10 \pmod{11}$.
Cum cele două valori sunt egale, Bob decide că mesajul este autentic, şi trece la aflarea mesajului subliminal:
- determină întâi $3^{-1} = 7 \pmod{10}$, *apoi*
- $y = \dfrac{x - p \cdot \beta}{\gamma} = \dfrac{5 - 8 \cdot 6}{3} = 9 \pmod{10}$.

2.3.4 Canalul subliminal Seberry - Jones

Sistemul are ca bază un algoritm mai vechi de autentificare, numit *schema de autentificare rapidă a lui Shamir*; pentru comparaţie, vom începe prin a prezenta această schemă.

Schema de autentificare Shamir

Protocolul, bazat pe problema rucsacului, este propus de Shamir în 1978. În 1984 Odlyzco construieşte un atac reuşit asupra sa ([54]).

Fie n un număr natural şi p $(q \geq 2^n)$ un număr prim. În faza de construcţie, *Alice* efectuează următoarele operaţii:

1. Generează aleator o matrice binară $K_{n \times 2n} = (k_{ij})$ ale cărei elemente sunt secrete.
2. Determină un vector $A = (a_1, a_2, \ldots, a_{2n}),\ a_i \in Z_q$ care verifică relaţia
$$K \cdot A^T = \begin{pmatrix} 1 \\ 2 \\ \vdots \\ 2^{n-1} \end{pmatrix} \pmod{q}$$
n elemente sunt generate aleator, iar celelalte n se determină din rezolvarea acestei ecuaţii matriciale.
3. Perechea (A, q) este publică.

Să presupunem acum că *Alice* vrea să trimită lui *Bob* un mesaj $m \in Z_q$. Pentru autentificarea lui, ea construieşte vectorul $C = (c_1, c_2, \ldots, c_{2n})$ în felul următor:

1. Reprezintă m în binar: $m = a_1a_2 \dots a_n, \quad a_i \in \{0, 1\}$.
2. Construieşte simetricul $\tilde{m} = a_n \dots a_2a_1$.
3. Calculează $C = \tilde{m} \cdot K \pmod{q}$.
4. Trimite perechea (m, C).

La recepţie, *Bob* verifică autenticitatea lui m verificând dacă $C \cdot A^T = m$. Într-adevăr,

$$C \cdot A^T = (\tilde{m} \cdot K) \cdot A^T = \tilde{m}\left(K \cdot A^T\right) = \tilde{m} \cdot (1\ 2\ \dots\ 2^{n-1})^T = m \pmod{q}$$

Un factor de insecuritate îl constituie faptul că fiecare mesaj trimis conduce la construirea unui sistem de $2n$ ecuaţii liniare, din care se pot obţine unele informaţii referitoare la structura matricii de autentificare. Odlyzko a arătat că, în general, după interceptarea a n mesaje, *Oscar* poate deduce matricea K.

Ulterior, Shamir aduce unele îmbunătăţiri sistemului, adăugând la fiecare mesaj un vector binar aleator $R = (r_1, \dots, r_{2n})$.
Autentificatorul C asociat mesajului $m \in Z_q$ este construit astfel:

1. Se calculează $\alpha = m - R \cdot A^T \pmod{q}$.
2. Se determină $C' = \tilde{\alpha} \cdot K \pmod{q}$ şi apoi $C = C' + R$.
3. Se trimite prin canal perechea (C, m).

Relaţia de verificare a autenticităţii este aceeaşi cu cea din schema iniţială:

$$C \cdot A^T = (C' + R) \cdot A^T = C' \cdot A^T + R \cdot A^T = \tilde{\alpha} \cdot K \cdot A^T + R \cdot A^T = \alpha + R \cdot A^T = m \pmod{q}$$

Exemplul 2.8. *Fie $n = 3,\ q = 7$ şi matricea secretă*

$$K = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Pentru vectorul A, alegem aleator (din Z_7) primele trei componente; să presupunem că $a_1 = 1,\ a_2 = 3,\ a_3 = 4$. Restul componentelor se determină rezolvând ecuaţia matricială

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 3 \\ 4 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 4 \end{pmatrix} \pmod{7}$$

Se obţine în final $A = (1\ 3\ 4\ 4\ 1\ 2)$.

Să tratăm cele două variante ale schemei rapide Shamir.

1. *Nu se foloseşte vectorul aleator R.*
   Atunci un text clar – să spunem $m = 3$ *– este autentificat prin*

$$C = \tilde{m} \cdot K = (1\ 1\ 0) \cdot \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} = (1\ 1\ 2\ 1\ 1\ 0)$$

   La primirea perechii $(m, C) = (3, (1\ 1\ 2\ 1\ 1\ 0))$ *şi folosind cheia publică* (A, q)*, Bob autentifică* m *calculând*

$$C \cdot A^T = (1\ 1\ 2\ 1\ 1\ 0) \cdot (1\ 3\ 4\ 4\ 1\ 2)^T = 17 \equiv 3 \pmod 7$$

2. *Se foloseşte un vector binar aleator R; fie acesta* $R = (0\ 1\ 1\ 1\ 0\ 1)$*. Autentificatorul pentru mesajul* $m = 3$ *se calculează treptat astfel:*
   $\alpha = m - R \cdot A^T = 3 - (0\ 1\ 1\ 1\ 0\ 1) \cdot (1\ 3\ 4\ 4\ 1\ 2)^T = 3 - 13 \equiv 4 \pmod 7$.
   $\alpha = 4$ *se converteşte în secvenţa binară* $(1\ 0\ 0)$ *şi se inversează:* $\tilde{\alpha} = (0\ 0\ 1)$.
   $C' = \tilde{\alpha} \cdot K = (0\ 0\ 1) \cdot \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} = (1\ 0\ 0\ 0\ 1\ 1)$.
   $C = C' + R = (1\ 0\ 0\ 0\ 1\ 1) + (0\ 1\ 1\ 1\ 0\ 1) = (1\ 1\ 1\ 1\ 1\ 2)$.

   La primirea perechii $(3, (1\ 1\ 1\ 1\ 1\ 2))$*, Bob verifică autenticitatea mesajului m:*

$$C \cdot A^T = (1\ 1\ 1\ 1\ 1\ 2) \cdot (1\ 3\ 4\ 4\ 1\ 2)^T = 17 \equiv 3 \pmod 7$$

De remarcat că generarea autentificatorului C se realizează folosind reprezentarea binară a mesajului m, în timp ce procesul de autentficare operează în Z_q.

Canalul Seberry Jones

Fie q un număr prim mare şi $n = \lceil log_2 q \rceil$. Toate operaţiile vor avea loc în corpul Z_q.
În faza de pregărire a canalului subliminal, *Alice* construieşte următoarele elemente:

1. O matrice $K \in \mathcal{M}_{n\times 2n}(Z_q)$;
2. Un vector public $A = (a_1, \ldots, a_{2n})$, $a_i \in Z_q$. n componente ale lui A sunt generate aleator, iar celelalte sunt determinate de ecuaţia matricială

$$K \cdot A^T = \begin{pmatrix} 1 \\ 2 \\ \vdots \\ 2^{n-1} \end{pmatrix}$$

3. Un vector secret $B = (b_1, \ldots, b_{2n})$, $b_i \in Z_q$ cu $K \cdot B^T = 0$.

   Vectorul B este trimis lui *Bob* printr-un canal ascuns.

Să presupunem acum că *Alice* vrea să trimită lui *Bob* mesajul subliminal $y \in Z_q$, folosind mesajul $x \in Z_q$. Pentru aceasta:

1. Determină un vector binar $R = (r_1, r_2, \ldots, r_{2n})$ din ecuaţia
   $$Y = R \cdot B^T \ \ (mod\ q).$$
2. Criptează mesajul x în modul următor:
   (a) $x' = x - R \cdot A^T \ \ (mod\ q)$.
   (b) $\alpha' = x'_{(2)} \cdot K$, unde $x'_{(2)}$ este reprezentarea în binar a lui x'.
   (c) $\alpha = \alpha' + R$ (de remarcat că α şi α' sunt vectori cu $2n$ componente).
3. Trimite lui *Bob* perechea (x, α).

La recepţie, *Bob*:

1. Verifică egalitatea $\alpha \cdot A^T = x$; $(mod\ q)$. Dacă este îndeplinită, atunci mesajul x este autentic.
2. Determină mesajul subliminal y prin

$$\alpha \cdot B^T = (\alpha' + R) \cdot B^T = \left(x'_{(2)} \cdot K + R\right) \cdot B^T = x'_{(2)} \cdot K \cdot B^T + R \cdot B^T = R \cdot B^T = y \quad (mod\ q)$$

Exemplul 2.9. *Să presupunem că Alice şi Bob vor să trimită mesaje subliminale din Z_5. Deci $q = 5$ şi $n = \lceil log_2 5 \rceil = 3$.*

Alice alege aleator matricea K fie aceasta

$$K = \begin{pmatrix} 2 & 2 & 4 & 1 & 3 & 1 \\ 4 & 4 & 1 & 1 & 0 & 2 \\ 4 & 3 & 4 & 2 & 2 & 0 \end{pmatrix}$$

Similar schemei rapide Shamir (Exemplul 2.8), se determină un vector A din

$$K \cdot A^T = \begin{pmatrix} 2 & 2 & 4 & 1 & 3 & 1 \\ 4 & 4 & 1 & 1 & 0 & 2 \\ 4 & 3 & 4 & 2 & 2 & 0 \end{pmatrix} \cdot (2\ 2\ 2\ a_4\ a_5\ a_6)^T = \begin{pmatrix} 1 \\ 2 \\ 4 \end{pmatrix} \quad (mod\ 5)$$

Se găseşte $A = (2\ 2\ 2\ 2\ 4\ 1)$. *Vectorul B se determină din*

$$K \cdot B^T = \begin{pmatrix} 2 & 2 & 4 & 1 & 3 & 1 \\ 4 & 4 & 1 & 1 & 0 & 2 \\ 4 & 3 & 4 & 2 & 2 & 0 \end{pmatrix} \cdot (b_1\ b_2; b_3\ b_4\ b_5\ b_6)^T = 0 \quad (mod\ 5)$$

Deoarece Alcie are la dispoziţie 3 *ecuaţii şi* 6 *necunoscute, va fixa trei din ele (cu valori din* Z_5*); fie acestea* $b_1 = 1,\ b_2 = 2,\ b_3 = 4$.
Atunci elementele rămase vor fi $b_4 = 4,\ b_5 = 3,\ b_6 = 0$.
Deci cheia secretă este $B = (1\ 2\ 4\ 4\ 3\ 5)$.

Dacă acum Alice doreşte să trimită mesajul subliminal $y = 4$ *odată cu textul clar* $x = 3$, *ea va calcula întâi vectorul binar R din ecuaţia*

$$(r_1\ r_2\ r_3\ r_4\ r_5\ r_6) \cdot (1\ 2\ 4\ 4\ 3\ 0)^T = 4 \quad (mod\ 5)$$

Sunt multe secvenţe binare care verifică această ecuaţie. Să presupunem că a fost aleasă soluţia $R = (1\ 0\ 1\ 1\ 0\ 0)$. *Mai departe, Alice calculează secvenţa de autentificare:*
$x' = x - R \cdot A^T = 3 - (1\ 0\ 1\ 1\ 0\ 0) \cdot (2\ 2\ 2\ 2\ 4\ 1)^T = 2\ \ (mod\ 5)$.

$$\alpha' = x'_{(2)} \cdot K = (0\ 1\ 0) \cdot \begin{pmatrix} 2 & 2 & 4 & 1 & 3 & 1 \\ 4 & 4 & 1 & 1 & 0 & 2 \\ 4 & 3 & 4 & 2 & 2 & 0 \end{pmatrix} = (4\ 4\ 1\ 1\ 0\ 2).$$

$\alpha = \alpha' + R = (4\ 4\ 1\ 1\ 0\ 2) + (1\ 0\ 1\ 1\ 0\ 0) = (0\ 4\ 2\ 2\ 0\ 2)$.

Perechea $(3, (0\ 4\ 2\ 2\ 0\ 2))$ *este trimisă lui Bob.*

Acesta efectuează înmulţirea

$$\alpha \cdot A^T = (0\ 4\ 2\ 2\ 0\ 2) \cdot (1\ 2\ 4\ 4\ 3\ 0)^T = 3 \quad (mod\ 5)$$

Deoarece $x = 3$ *este egal cu* $\alpha \cdot A^T$, *autentificarea este verificată. Se recrează acum mesajul subliminal*

$$y = \alpha \cdot B^T = (0\ 4\ 2\ 2\ 0\ 2) \cdot (1\ 2\ 4\ 4\ 3\ 0)^T = 4 \quad (mod\ 5)$$

2.4 Exerciţii

2.1. *Să construim un MAC folosind modul CFB de implementare, în loc de modul CBC ([2]): fiind date blocurile de text clar* $\alpha_1, \ldots, \alpha_n$*, definim vectorul de inţializare* $\beta_0 = \alpha_1$*. Apoi criptăm secvenţa de blocuri* $\alpha_2, \ldots, \alpha_n$ *după formulele*

$$\beta_i = \alpha_{i+1} \oplus e_K(\beta_{i-1}) \qquad 1 \le i \le n-1$$

În final, $MAC(\alpha_1 \| \ldots \| \alpha_n) = e_K(\beta_{n-1})$*. Arătaţi că acesta este identic cu CBC MAC.*

2.2. *Fie* $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *un sistem de criptare simetric cu* $\mathcal{P} = \mathcal{C} = \{0,1\}^m$ *şi o funcţie de dispersie* $h : (\{0,1\}^m)^n \longrightarrow \{0,1\}^m$ *definită (pentru cheia* $K \in \mathcal{K}$*):*

$$h_K(y_1, \ldots, y_n) = e_K(y_1) \oplus \ldots \oplus e_K(y_n)$$

Arătaţi că codul HMAC construit pe baza acestei funcţii nu este sigur.

2.3. *Alice şi Bob intenţionează să comunice folosind canalul subliminal Simmons. Ei convin asupra numerelor prime* $p = 5,\ q = 7,\ r = 11$ *şi asupra unei strategii comune de primire: anume prima rădăcină (în ordonarea crescătoare) va reprezenta mesajul elementar* $'0'$*, iar a cincea – mesajul elementar* $'1'$*.*

Să se determine autentificatorii ambelor mesaje binare, la primirea mesajului criptat $c = 256$*.*

2.4. *Fie canalul subliminal Ong - Schnorr - Shamir dat de* $n = 21$ *şi cheia secretă* $k = 5$*. Să se determine dacă tripletul*

$$(c_1, c_2, c_3) \in \{(14, 12, 11), (11, 1, 5), (8, 18, 5)\}$$

transmite mesaje ascunse.
În caz afirmativ, să se determine aceste mesaje.

2.5. *Se consideră canalul subliminal El Gamal dat de* $q = 13,\ \alpha = 6$*. Să presupunem că Alice trimite mesajul* $y = 9$ *folosind cheia secretă* $r = 10$ *şi textul criptat* $x = 11$*.*
Să se determine parametrii (u, v)*.*

2.6. *Folosind aceeaşi parametri din problema anterioară, să se determine dacă tripletul* $(c, x, y) = (11, 5, 1)$ *asigură un canal subliminal. În caz afirmativ, să se determine textul clar corespunzător.*

2.7. *Schema El Gamal este definită peste corpul* Z_q *generat de elementul primitiv* α*. Ce se întâmplă cu procesele de criptare/decriptare dacă* $\alpha \in Z_q$ *nu este primitiv ?*

2.8. *Să se construiască un canal Sebery - Jones pentru transmiterea mesajelor ascunse din* Z_7*. Este posibilă folosirea matricii K şi a vectorului A din Exemplul 2.9 ? În caz afirmativ, urmăriţi procedeele de criptare şi decriptare pentru transmiterea mesajului ascuns* $m = 3$*.*

Bibliografie

[1] C. A. Asmuth, J. Bloom – *A modular approach to key safeguarding*, IEEE Trans on IT 29 (1983), pp. 208-210.

[2] Atanasiu, A. – *Securitatea Informaţiei, vol. 1, Criptografie*, ed. InfoData, Cluj, 2008.

[3] J. Benaloh, J. Leichter – *Generalized secret sharing and monotone functions*, în "Advances in Cryptology – CRYPTO 8", S. Goldwasser, ed., LNCS 403 (1989), pp. 27-35.

[4] S. Blake - Wilson, A. Menezes – *Authenticated Diffie - Hellman Key Agreement Protocols*, Proc. of the 5-th Intern. Workshop on Security Protocols, LNCS 1361, 1997, pp. 137-158.

[5] G. R. Blakley – *Safeguarding cryptographic keys*, în "Proc. of the National Computer Conference, 1979", American Federation of Information Processing Societies Proc. 48 (1979), pp. 313-317.

[6] Boneh, D., Franklin, M. – *Identity Based Encryption from the Weil Pairing*, SIAM Journal of Computing, vol. 32, no. 3, pp. 586-615.

[7] Boneh, D., Boyen, X. – *Efficient Selective - ID Secure Identity Based Encryption Without Random Oracles*, Proc. of EUROCRYPT 2004, Interlaken, Elveţia, 2-6 May 2004, pp. 223-238.

[8] J. N. Bos, D. Chaum - *Provably unforgable signatures*, LNCS, 740 (1993), pp. 1-14.

[9] S. Brands – *An Efficient Off-Line Electronic Cash System Based On The Representation Problem*, Technical Report CS-R9323, 1993, CWI, Amsterdam, Netherlands.

[10] D. Chaum, H. van Antwerpen – *Undeniable signatures*, LNCS, 435 (1990), pp. 212-216.

[11] D. Chaum, A. Fiat, M. Naor – *Untraceable Electronic Cash*, Advances in Cryptology, CRYPTO 8, S. Goldwasser (Ed.), Springer-Verlag, pp. 319-327.

[12] D. Chaum, E. van Heyst – *Group signatures*, Advances in Cryptology, EUROCRYPT 91, LNCS, vol. 547, Springer-Verlag (1991), pp. 257-265.

[13] D. Chaum, E. van Heijst, B. Pfitzmann – *Cryptographically strong undeniable signatures, unconditionally secure for the signer*, LNCS, 576 (1992), pp. 470-484.

[14] Chen, L. ş.a – *An Efficient ID-KEM Based on the Sakai-Kasahara Key Construction*, IEEE Proc. Information Theory, vol. 153, no. 1, (2006), pp. 19-26.

[15] T-S Chen, K-H Huang, Y-F Chung – *Digital Multi - Signature Scheme based on the Elliptic Curve Cryptosystem*, J. Comp. Sci & Technol. vol. 19 (2004), no. 4, pp. 570-573.

[16] X. Chen, F. Zang, K. Kim – *A new ID - based group signature scheme from bilinear pairings*, *http : //eprint.iacr.org/*2003/100*.pdf*, 2003.

[17] B. Chor, S. Goldwasser, S. Micali, B. Awerbuch – *Verifiable secret sharing and achieving simultaneity in the presence of faults*, Proc. of the 26th IEEE Symposium on the Foundations of Computer Science, (1985), pp. 383-395

[18] Cocks, C. – *an Identity Based Encryption Scheme Based on Quadratic Residues*, Proc. of the Eighth IMA Intern. Conf. on Cryptography and Coding, Cirencester, 17-19 Dec. 2001, pp. 360-363.

[19] J.S. Coron, D. Naccache, J. Stern – *On the security of RSA Padding*, In Advances of Cryptology CRYPTO 99, LNCS 1666, Springer - Verlag, 1999, pp. 1-18.

[20] I.B. Damgard –*A design principle for hash functions*, LNCS, 435 (1990), pp. 516-427.

[21] H. Delfs, H. Knebl – *Introduction to Cryptography, Second edition*, Springer Verlag, 2007.

[22] W. Diffie, M.E. Hellman – *Multiuser cryptographic techniques*, AFIPS Conference Proceedings, $45(1976), 109 - 112$

[23] W. Diffie, M.E. Hellman – *New Directions in Cryptography*, IEEE Trans. on Information Theory, vol. IT-22 (1976), pp 644-654

[24] H. Dobbertin – *Cryptanalysis of MD4*, Journal of Cryptology, 11 (1998), pp. 253-271.

[25] T. ElGamal – *A public key cryptosystem and a signature scheme based on discrete algorithms*, IEEE Trans. on Information Theory, 31 (1985), pp. 469-472.

[26] M. J. Farsi – *Digital Cash*, Masters Thesis in Computer Science, Dpt of Mathematics and Computing Science, Goteborg University 1997.

[27] P. Feldman – *A practical scheme for non-interactive verifiable secret sharing*, Proc. of the 28th IEEE Symposium on the Foundations of Computer Science, (1987), pp. 427-437.

[28] N. Ferguson – *Single Term Off-Line Coins*, Advances in Cryptology - EUROCRYPT 93, Springer-Verlag, pp. 318-328.

[29] Fujisaki, E, Okamoto, T. – *Secure Integration of Assymetric and Simmetric Encryption Schemes*, Proc. of Crypto 99, Santa Barbara, CA, August 20-24, 1999, pp. 537-554.

[30] T. El Gamal – *A public key cryptosystem and a signature scheme based on discrete algorithms*, IEEE Trans on Inf. Theory, 31 (1985), pp. 469-472.

[31] J. Gibson – *Discrete logarithm hash function that is collision free and one way*, IEEE Proceedings-E, 138 (1991), 407-410.

[32] H. Ghodosi, J. Pieprzyk, Safavi-Naini – *Remarks on the multiple assignment secret sharing scheme*, LNCS 1334 (1997), 72-82.

[33] E. van Heyst, T.P.Petersen – *How to make efficient fail-stop signatures*, LNCS, 658 (1993), 366-377.

[34] S. Iftene – *A generalisation of Mignottes secret sharing scheme*, în Proc. of the 6th Intern. Symposium on Symbolic and Numeric Algorithms for scientific computing, Timişoara, T. Jebelean, V. Negru, D. Petcu, D. Zaharia (eds), Sept. 2004, pp. 196-201, Mirton Publ. House (2004).

[35] I. Ingermarsson, G. D. Simmons – *A protocol to set up shared secret schemes without assistance of mutually trusted party*, EUROCRYPT 90, LNCS vol. 473, Springer - verlag 1991, pp. 266-282

[36] W. Jackson, K.M. Martin, C. M. O′ Keefe – *Muttualy trusted authority - free secret sharing schemes*, Journal of Cryptology, 10 (4), 1997, pp. 261-289

[37] E. D. Karnin, J. W. Greene, M. E. Hellman – *On secret sharing systems*, IEEE Trans. on Information Theory 29 (1983), pp. 35-41.

[38] V. Klima – *Tunnels in Hash Functions: $MD5$ Collisions within a Minute*, Cryptology ePrint Archive, http://eprint.iacr.org, Report 105 (2006).

[39] Klitz, E. – *On the Limitations of the Spread of an IBE-to-PKE Transformation*, Proc of PKC 2006, LNCS 3958, Springer, 2006, pp. 274-289.

[40] H. Krawczyk, M. Bellare, R. Canetti – *HMAC: Keyed - Hashing for Message Authentication*, RFC 2104, 1997.

[41] E. Kranakis – *Primality and Cryptography*, Wiley-Teubner Series in Computer Science (1986).

[42] H. Krawczyk – *Secret sharing made short*, în Advances in Cryptology – CRYPTO 93, D. R. Stinson, ed., LNCS 773 (1994), pp. 136-146.

[43] L. Law, S. Sabett, J. Solinas – *How to make a mint: The Cryptography of Anonymous Electronic Cash*, http://jya.com/nsamint.htm

[44] Mambo, Masahiro, Keisyke, Usuda, Okamoto – *Proxy signatures: Delegation of the power to sign mesages*, IEICE Trans. Fundamentals, ET9-A (1996), pp. 1338 - 1354.

[45] Martin, L – *Introduction to Identity – Based Encryption*, Artech House, Information Security and Privacy Series, 2008.

[46] T. Matsumoto, Y. Takashima, H. Imai – *On seeking smart public-key distribution systems*, The Trans. of the IECE of Japan, E69 (1986), pp. 99-106.

[47] C. Meadows – *Some threshold schemes without central key distributors*, Congressus Numerantium, 46 (1985), pp. 187-199.

[48] R. J. McEliece, D. Sarwate – *On sharing secrets and Reed-Solomon codes*, Comm. of the ACM 24 (1981), pp. 583-584.

[49] A. Menezes, P. van Oorschot, S. Vanstome – *Handbook of Applied Cryptography*, CRC Press Inc (1997)

[50] R.C. Merkle – *A fast software one-way functions and DES*, LNCS, 435 (1990), pp. 428-446.

[51] M. Mignotte – *How to share a secret*, în Cryptography Proc., Burg Feuerstein 1982, T. Beth, ed., LNCS 149 (1983), pp. 371-375.

[52] C. J. Mitchell, F. Piper, P. Wild – *Digital signatures*, Contemporary Cryptology, The Science of Information Integrity, IEEE Press, (1992), pp. 325-378.

[53] R. Needham, M. Schroeder – *Using encryption for authentication in large networks of computers.*, Comm. of the ACM 21, 12 (1978), pp. 993 999.

[54] A. M. Odlyzco – *Cryptanalytic attacks on the multiplicative knapsack cryptosystems and on Shamirs fast signature scheme*, IEEE Trans. on Information Theory, IT30 (1984), pp. 594-601.

[55] T. Okamoto, K. Ohta – *Universal electronic cash*, J. Feigenbaum (Ed.), Advances in cryptology CRYPTO91, LNCS 576, Springer-Verlag (1992).

[56] B. Preneel, R. Govaerts, J. Vandewalle – *Hash functions based on block ciphers: a syntetic approach*, LNCS, 773 (1994), pp. 368-378.

[57] M.O. Rabin – *Efficient dispersal of information for security, load balancing, and fault tolerance*, Journal of ACM, 36(2), 1989, pp. 335-348.

[58] R.L. Rivest – *The $MD4$ message digest algorithm*, LNCS, 537, (1991), pp. 303-311.

[59] R. L. Rivest – *The $MD5$ Message Digest Algorithm*, RFC 1321 (1992).

[60] A. Salomaa – *Criptografie cu chei publice*, Ed. Militară, 1994.

[61] A. Shamir – *Identity-Based Cryptosystems and Signature Schemes*, Proc. of CRYPTO 84, Santa Barbara, CA. August 19-22, 1984, pp. 47-53.

[62] A. Shamir – *How to share a secret*, Comm of the ACM 22 (1979), 612-613.

[63] G.J. Simmons – *The prisoners problem and the subliminal channel*, Advances in Cryptology - Crypto, 83 (1984), pp. 51-67.

[64] M. E. Smid, D. K. Branstad – *Response to comments on the $NIST$ proposed digital signature standard*, LNCS, 740 (1993), pp. 76-88.

[65] M. Stadler – *Publicly verifiable secret sharing*, Advances in Cryptology - EUROCRYPT 96, LNCS vol. 1070, Springer verlag (1996), pp. 190-199.

[66] D. R. Stinson – *Decomposition constructions for secret sharing schemes*, IEEE Trans. on Information Theory 40 (1994), pp. 118-125.

[67] D. Stinton – *Cryptographie, theorie et pratique*, Intern. Thompson Publ. France, 1995.

[68] Z. Tan, Z. Liu, C. Tan – *Digital proxy Blind Signature Schemes based on DLP and ECDLP*, MM Research Preprints, 212-217, Academia Sinica, Beijing, no. 21, Dec. 2002.

[69] S. Vaudenay – *A Classical Introduction to Cryptography*, Springer Verlag 2006.

[70] H.C.Williams – *Some public-key criptofunctions as intractable as factorisation*, Cryptologia, 9 (1985), pp. 224-237.

[71] ISO/IEC 9796; Information technology – Security Techniques – Digital Signature Scheme Giving message recovery, Intern. Organisation for Standardisation, Geneva, 1991.

[72] *Secure Hash Standard*, National Bureau of Standards, FIPS Publications 180, 1993.

[73] *SKIPJACK and KEA Algorithm Specifications*, versiunea 2.0, http://csrc.nist.gov/groups/ST/toolkit/documents/skipjack/skipjack.pdf

[74] *Digital signature standard*, National Bureau of Standards, FIPS Publications 186, 1994