METODE CRIPTOGRAFICE DE PROTECȚIE A INFORMAȚIEI

Tema 3: Cifruri fluide moderne

OBIECTIVELE TEMEI

A defini conceptul de sistem de criptare fluidă

A realiza o clasificare a cifrurilor fluide

A examina sistemul de criptare Vernam și schema one-time pad

A descrie modelul general utilizat în construcția cifrurilor fluide

A compara la nivel de proprietăți sistemele de criptare fluidă sincronizabile cu cele auto-sincronizabile

A examina cifruri fluide adaptate pentru implementare hardware sau implementare software

A examina cifrurile fluide finaliste la concursul e-STREAM, care satisfac rigorilor actuale de securitate

SUBIECTELE ABORDATE ÎN CADRUL TEMEI

Sisteme de criptare fluidă – partea I

Clasificarea cifrurilor fluide

Sistemul de criptare Vernam și schema one-time pad

Modelul general al cifrurilor fluide

Sisteme de criptare fluidă sincronizabile

Sisteme de criptare fluidă auto-sincronizabile

SUBIECTELE ABORDATE ÎN CADRUL TEMEI

Sisteme de criptare fluidă – partea II

Cifruri fluide adaptate pentru implementare hardware sau software

Sistemul de criptare SEAL

Sistemul de criptare RC4

Sistemul de criptare A5/1

Finalistele concursului e-STREAM

Cifrul Salsa20

Cifrul Trivium

INTRODUCERE

Cifrurile clasice cu cheie simetrică, care au fost examinate la secțiunea precedentă, operează la nivelul simbolurilor textului clar. Dezvoltarea rapidă a tehnicii de calcul, începând cu a doua jumătate a secolului XX, a condus la necesitatea elaborării unor sisteme de criptare ce operează la nivel de biți ai reprezentării binare pentru textul clar. Această necesitate se explică prin aceea că datele ce urmează a fi criptate nu întotdeauna conțin doar simboluri de text și cifre, ci și elemente de grafică, date audio și video ș.a. Astfel, este comod de transformat datele în șiruri de biți și de transmis aceste șiruri în formă criptată.

INTRODUCERE

De menționat că înainte de criptare, fiecare simbol este înlocuit cu un anumit număr de biți (de regulă, 8 biți), adică lungimea mesajului ce urmează a fi criptat devine una esențial mai mare (de 8 ori!). Reprezentarea binară a textului clar este realizată în două etape: mai întâi, folosind codificarea ASCII, fiecărui simbol al textului clar i se asociază un număr întreg din intervalul $[0,255]$, după care este realizată transformarea numerelor din zecimal în binar (fiecare număr este reprezentat pe 7 sau 8 biți) sau, uneori, pentru o reprezentare mai restrânsă, în hexazecimal.

INTRODUCERE

În ceea ce privește sistemele de criptare cu cheie simetrică moderne, există două clase de astfel de algoritmi: cifrurile fluide și cifrurile bloc.

Sistemele de criptare fluidă criptează biții din reprezentarea binară a textului clar, combinând prin XOR fiecare bit al textului clar cu bitul corespunzător din cheia fluidă. Spre deosebire de acestea, sistemele de criptare bloc operează la nivel de grup de biți de lungime fixată, cu aceeași cheie per grup.

INTRODUCERE

Definiția 2.2.1. Fie structura matematică $M=(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ definește un sistem de criptare. Un șir infinit de biți $k_1k_2k_3\ldots\left(k_i\in\{0,1\}\right)$ (sau de octeți), generat în mod pseudoaleator în baza biților cheii secrete $K\in\mathcal{K}$, este numit cheie fluidă.

Definiția 2.2.2. Structura matematică $M=(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ definește un sistem de criptare fluidă dacă textul clar m este divizat în grupuri de biți sau octeți, $m=m_1m_2m_3\ldots$, care sunt criptați, corespunzător, în $c=c_1c_2c_3\ldots=E_{k_1}(m_1)E_{k_2}(m_2)E_{k_3}(m_3)\ldots$, unde $k=k_1k_2k_3\ldots$ este o cheie fluidă.

INTRODUCERE

De regulă, într-un cifru fluid funcția de criptare este definită în baza operației sau exclusiv (numită și XOR sau suma modulo 2) între biții textului clar și biții corespunzători ai cheii fluide, $c_i = m_i \oplus k_i$. Textul criptat obținut reprezintă un șir de biți a cărui lungime este egală cu lungimea în biți a textului clar. Analog, funcția pentru decriptarea mesajului este definită astfel: $m_i = c_i \oplus k_i$. Ideea de a utiliza operația XOR la criptare și decriptare a fost propusă de către G. Vernam în 1917. Pentru a mări eficiența aplicațiilor software bazate pe utilizarea cifrurilor fluide, algoritmii moderni ai acestor cifruri operează pe cuvinte formate din biți, de lungime dependentă de arhitectura mașinii de calcul.

INTRODUCERE

Spre deosebire de acestea, *sistemele de criptare bloc* criptează grupuri de biți succesivi ai textului clar în baza unei transformări de criptare cu aceeași cheie fixată. Astfel, dacă $m = m_1 m_2 m_3 \ldots$ este textul clar (m_i sunt blocuri de biți din reprezentarea binară a textului clar), iar K - cheia secretă, atunci textul criptat este definit astfel: $c = c_1 c_2 c_3 \ldots = E_K(m_1) E_K(m_2) E_K(m_3) \ldots$.

De regulă, pentru cifrurile fluide viteza de criptare/decriptare este mai mare în raport cu cea corespunzătoare cifrurilor bloc, dar cifrurile fluide se confruntă cu anumite dificultăți de securitate dacă sunt implementate incorect, în particular, dacă cheia secretă inițială este utilizată de două sau mai multe ori.

Problema principală în construcția cifrurilor fluide ține de generarea unei chei fluide teoretic infinite. Aceasta se poate realiza fie aleator, fie în baza unui algoritm, care pornește de la o secvență mică de chei secrete. Un astfel de algoritm se numește generator de chei fluide.

INTRODUCERE

Cifrurile fluide sunt utilizate în aplicații din domeniul telecomunicațiilor, deoarece în cadrul acestora caracterele sunt procesate individual, imediat ce sunt recepționate, iar la dispoziție sunt puse resurse de calcul limitate. Cifrurile bloc sunt utilizate în special pentru a cripta comunicațiile în Internet.

În continuare vom introduce conceptele de bază ce caracterizează sistemele de criptare fluide. Sunt prezentate câteva sisteme de criptare fluide, precum SEAL, RC4, A5/1, care sunt depășite actualmente, ținând cont de atacurile criptanalitice întreprinse asupra lor, dar și cifruri precum Salsa20 și Trivium, utilizate pe larg în prezent.

CLASIFICAREA CIFRURILOR FLUIDE

2.2.2.1 Sistemul de criptare Vernam și schema one-time pad

Peste alfabetul binar $\{0,1\}$ definim transformarea de criptare Vernam (a se vedea *Figura 2.2.2*) prin relația $c_i = m_i \oplus k_i,\ i = 1,2,3,\ldots$, unde $m_1, m_2, m_3, \ldots$ sunt biții textului clar, $k_1, k_2, k_3, \ldots$ - biții cheii fluide, $c_1, c_2, c_3, \ldots$ - biții textului criptat, iar $\oplus$ este notația pentru operația XOR. Operația de decriptare a mesajului se definește astfel: $m_i = c_i \oplus k_i,\ i = 1,2,3,\ldots.$

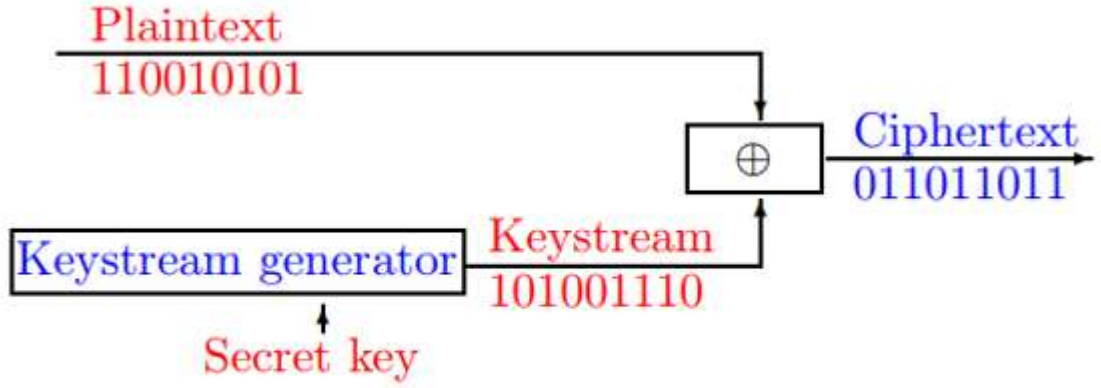

Figura 2.2.2. Ilustrarea cifrului Vernam

CLASIFICAREA CIFRURILOR FLUIDE

În cazul în care biții cheii fluide sunt generați aleator și aceasta nu mai este folosită ulterior, sistemul de criptare Vernam este numit schemă *one-time pad*. O schemă one-time pad este teoretic imposibil de spart și se zice că este necondiționat sigură împotriva atacurilor cu text criptat cunoscut [4]. Într-adevăr, pentru un text criptat cu o astfel de schemă, adversarul nu dispune de careva informație privind cheia fluidă sau textul clar. Pentru un sistem Vernam este dificil de generat o cheie de lungime egală cu cea a textului clar, care urmează să fie transmisă în mod securizat către destinatarul mesajului criptat.

CIFRURI FLUIDE COMPUTAȚIONAL SIGURE

Acest fapt conduce la ideea generării unor cifruri fluide în care cheia fluidă (de lungime considerabilă finită) să fie generată pseudoaleator, în baza unei chei secrete de dimensiune relativ mică (de exemplu, pe 128 biți), astfel ca rezultatul să pară aleator pentru un adversar (se simulează schema *one-time pad*). Astfel de cifruri fluide nu vor fi necondiționat sigure, dar, în schimb, pot fi computațional sigure.

C. Shannon a arătat că o condiție necesară ca o schemă de criptare cu cheie simetrică să fie necondiționat sigură este ca incertitudinea (entropia) cheii secrete să fie cel puțin egală cu incertitudinea textului clar. Schema *one-time pad* este necondiționat sigură, iar cheia acesteia are lungime minimal posibilă în raport cu alte scheme de criptare simetrică ce posedă această proprietate.

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

Definiția 2.2.3. Sistemul de criptare fluidă este numit cifru sincronizabil (sau sincron) dacă cheia fluidă a acestuia este generată independent de conținutul textului clar și al textului criptat.

Exemplul 2.2.1. Sistemul de criptare Vigenère poate fi definit ca un sistem de criptare fluidă sincronizabil. Fie m lungimea cuvântului cheie din sistemul Vigenère. Definim

$$\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_{26},\ \mathcal{K} = \left(\mathbb{Z}_{26}\right)^m,$$

$$E_z\left(x\right) = \mathrm{mod}\left(x+z, 26\right),\ D_z\left(y\right) = \mathrm{mod}\left(y-z, 26\right),$$

unde $\mathcal{L}$ este o mulțime finită nevidă, numită „alfabetul cheilor fluide".

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

Cheia fluidă $z_1 z_2 z_3 \ldots$ este definită astfel:

$$z_i = \begin{cases} k_i & dacă \ \ 1 \le i \le m \\ z_{i-m} & dacă \ \ i \ge m+1 \end{cases}.$$

Ea va genera din cheia fixă $K = (k_1, k_2, \ldots, k_m)$, cheia fluidă $k_1 k_2 \ldots k_m k_1 k_2 \ldots k_m \ldots$ □

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

Procedura de criptare cu un sistem de criptare fluidă sincronizabil poate fi descrisă ca un automat finit definit prin relațiile următoare:

$$\sigma_{i+1} = f(\sigma_i, k),\ z_i = g(\sigma_i, k),\ c_i = h(z_i, m_i),$$

unde σ_0 este starea inițială, care poate fi determinată din cheia k, f - funcția de tranziție a stărilor, g - funcția ce generează cheia fluidă z_i, iar h este funcția de ieșire ce combină cheia fluidă z_i și textul clar m_i pentru a produce textul criptat c_i. Procedeele de criptare și de decriptare sunt ilustrate în *Figura 2.2.4.*

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

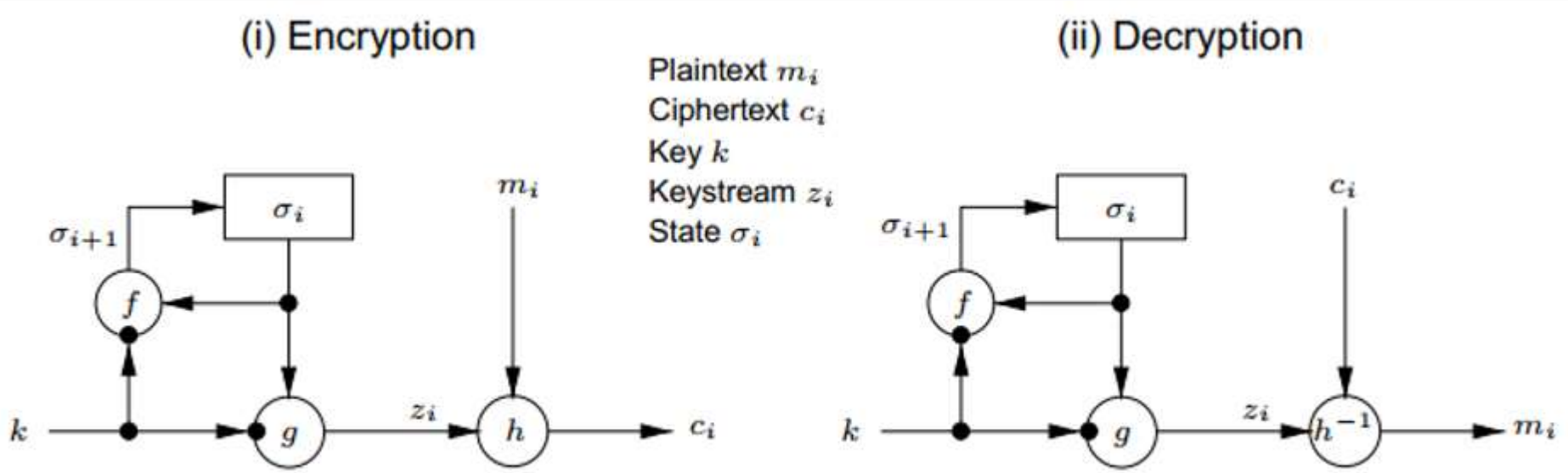

Figura 2.2.4. Modelul general al cifrului fluid sincronizabil

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

Proprietăți ale cifrurilor fluide sincronizabile:

- *Sincronizare.* Atunci când este folosit un cifru fluid sincronizabil, pentru a asigura o criptare/decriptare corectă, este necesar ca expeditorul și destinatarul să-și sincronizeze cheia fluidă, adică să folosească aceeași cheie fluidă și să opereze în aceeași locație (stare) a acesteia. Dacă în timpul transmisiei în textul criptat sunt inserați sau eliminați biți, atunci decriptarea eșuează și ea poate fi reluată numai în baza unor tehnici de resincronizare (cum ar fi, de exemplu, reinițializarea sau plasarea unor marcatori speciali la intervale regulate în textul transmis).

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

- *Nepropagarea erorii.* Modificarea unui bit din textul criptat (fără a se elimina sau adăuga biți) pe parcursul transmisiei datelor, nu va afecta decriptarea altor caractere.
- *Atacuri active.* Un adversar activ care elimină, inserează sau retrimite componente ale mesajului criptat, va provoca desincronizări și astfel, poate fi detectat la recepționarea mesajului. În schimb, adversarul poate efectua modificări de biți în textul criptat cu scopul de a urmări cum acestea afectează textul clar. Astfel un text criptat cu un cifru fluid sincronizabil necesită ca între utilizatorii implicați să fie implementate mecanisme de autentificare și de asigurare a integrității datelor.

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

Definiția 2.2.4. Vom spune că un sistem de criptare fluidă este aditiv binar dacă acesta este un cifru fluid sincronizabil în care cheia fluidă, precum și textul clar și cel criptat, sunt definite în cod binar ($\mathcal{L} = \mathcal{P} = \mathcal{C} = \mathbb{Z}_2$), iar funcția de ieșire h este operatorul XOR (notat prin $\oplus$).

SISTEME DE CRIPTARE FLUIDĂ SINCRONIZABILE

O ilustrare a cifrurilor fluide aditive binare este dată în *Figura 2.2.5*.

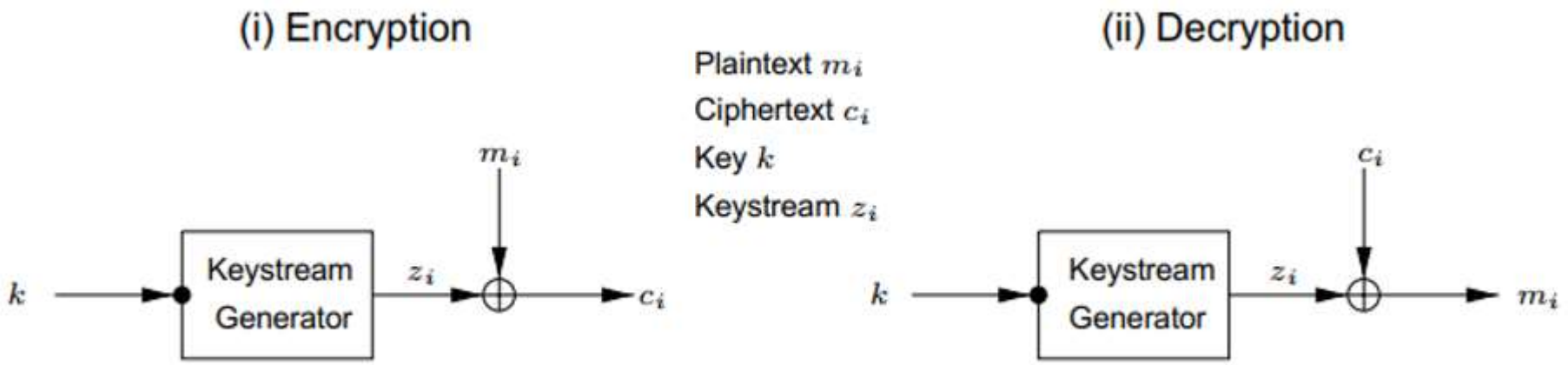

Figura 2.2.5. Modelul general al cifrului fluid aditiv

Generatorul de chei fluide este compus din funcția f, care generează starea următoare, și din funcția g (a se vedea *Figura 2.2.4*).

SISTEME DE CRIPTARE FLUIDĂ AUTO-SINCRONIZABILE

Definiția 2.2.5. Vom spune că un sistem de criptare fluidă este auto-sincronizabil (sau asincron) dacă funcția de generare a cheii fluide a acestuia depinde de un număr fixat de caractere criptate anterior.

Funcția de criptare a cifrului fluid auto-sincronizabil poate fi descrisă prin relațiile următoare:

$$\sigma_i = (c_{i-r}, c_{i-r+1}, \ldots, c_{i-1}),\ z_i = g(\sigma_i, k),\ c_i = h(z_i, m_i),$$

unde $\sigma_0 = (c_{-r}, c_{-r+1}, \ldots, c_{-1})$ este starea inițială (este cunoscută), k - cheia secretă, g - funcția ce generează cheia fluidă z_i, iar h este funcția de ieșire ce combină cheia fluidă z_i și textul clar m_i pentru a genera textul criptat c_i. Procedeele de criptare și de decriptare sunt ilustrate în *Figura 2.2.6.*

SISTEME DE CRIPTARE FLUIDĂ AUTO-SINCRONIZABILE

Proprietăți ale cifrurilor fluide auto-sincronizabile:

- *Auto-sincronizare.* Deoarece funcția pentru decriptare h^{-1} depinde numai de un număr fixat de caractere criptate anterior, desincronizarea – rezultată eventual prin inserarea sau ștergerea de caractere criptate – se poate evita. Cifrurile fluide auto-sincronizabile pot restabili proprietatea de sincronizare, afectând doar un număr finit de caractere din textul clar.

SISTEME DE CRIPTARE FLUIDĂ AUTO-SINCRONIZABILE

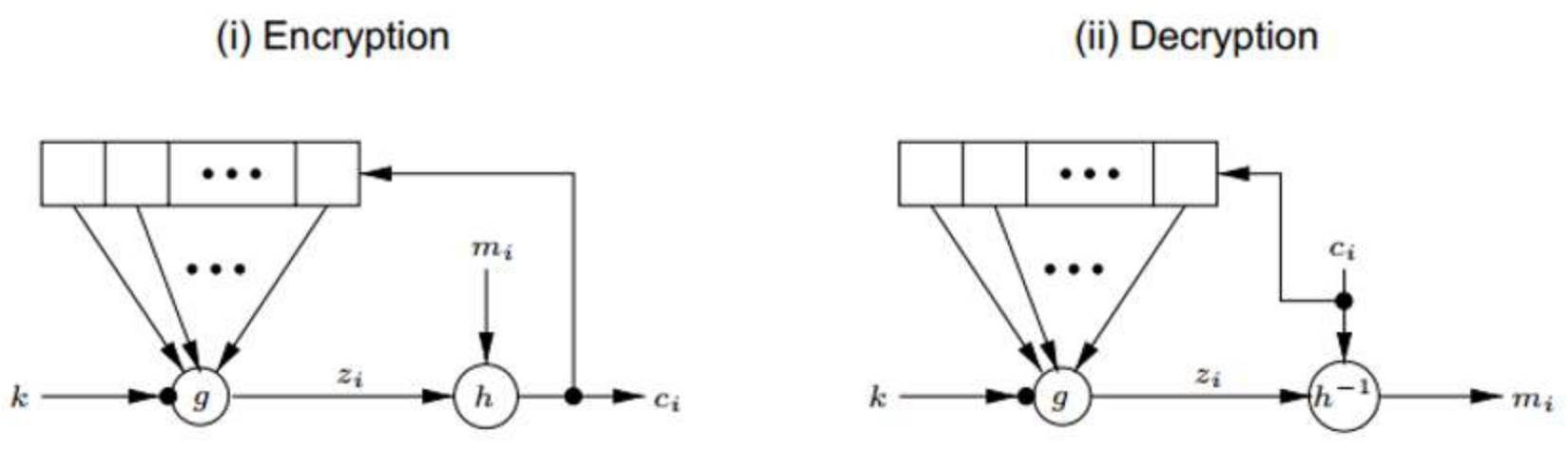

Figura 2.2.6. Modelul general al cifrului fluid auto-sincronizabil

SISTEME DE CRIPTARE FLUIDĂ AUTO-SINCRONIZABILE

- *Propagare limitată a erorii.* Dacă starea σ_i a unui sistem fluid auto-sincronizabil depinde de r caractere criptate anterior, atunci modificarea (eventual ștergerea sau inserarea) unui caracter din textul criptat poate duce la decriptarea incorectă a maxim r caractere, iar după aceea, decriptarea este efectuată corect.

SISTEME DE CRIPTARE FLUIDĂ AUTO-SINCRONIZABILE

- *Rezistență la atacuri active.* Din proprietățile menționate anterior rezultă că este relativ dificil de depistat atacul unui adversar activ (care poate modifica, insera sau șterge caractere), auto-sincronizarea readucând decriptarea la normalitate. De aceea, este necesar de implementat mecanisme suplimentare pentru a asigura autentificarea și integritatea datelor.

SISTEME DE CRIPTARE FLUIDĂ AUTO-SINCRONIZABILE

- *Difuzia textelor clare.* Deoarece fiecare caracter al textului clar influențează întregul text criptat ce urmează, eventualele proprietăți statistice ale textului clar sunt dispersate prin textul criptat. Prin urmare, din acest punct de vedere, sistemele de criptare fluidă auto-sincronizabile sunt mai rezistente decât cele sincronizabile la atacurile bazate pe redundanța (surplusul de informație) textului clar.

Un exemplu de cifru fluid auto-sincronizabil modern este Moustique, elaborat de către J. Daemen și P. Kitsos în anul 2006 în cadrul proiectului eSTREAM. În general, deoarece cifrurile fluide auto-sincronizabile sunt dificil de proiectat, acestea sunt mai puțin studiate și, respectiv, mai puțin utilizate în practică.

REGISTRELE DE DEPLASARE CU FEEDBACK LINIARE (LFSR – LINEAR FEEDBACK SHIFT REGISTER)

Un registru de deplasare cu feedback este format din:

- celule de stocare (locații de memorie), care la fiecare tact transmit celulei următoare bitul memorat la tactul anterior;
- componente ce realizează operația XOR;
- linie de feedback.

REGISTRELE DE DEPLASARE CU FEEDBACK LINIARE (LFSR – LINEAR FEEDBACK SHIFT REGISTER)

Stările celulelor și componentele XOR determină la fiecare tact (prin intermediul funcției de feedback) informația din prima celulă de stocare. Astfel, funcția de feedback exprimă orice element nou al șirului generat în funcție de elementele generate anterior. Dacă funcția de feedback este liniară (se poate implementa doar folosind operația XOR), spunem că generatorul este un registru de deplasare cu feedback liniar (LFSR).

REGISTRELE DE DEPLASARE CU FEEDBACK LINIARE (LFSR – LINEAR FEEDBACK SHIFT REGISTER)

Există mai multe argumente pentru utilizarea LFSR:

1. LFSR sunt bine adaptate pentru implementare hardware
2. LFSR pot produce șiruri de perioadă suficient de mare
3. LFSR pot produce șiruri cu proprietăți statistice bune
4. Datorită structurii, LFSR pot fi analizate în baza unor procedee algebrice.

REGISTRELE DE DEPLASARE CU FEEDBACK LINIARE (LFSR – LINEAR FEEDBACK SHIFT REGISTER)

Un registru de deplasare cu feedback liniar (LFSR) de lungime L constă din L etape numerotate $0,1,\ldots,L-1$, fiecare capabilă să memoreze un bit și care are o intrare și o ieșire, dar și un cronometru ce supraveghează deplasarea datelor. Pe parcursul fiecărui tact sunt realizate următoarele operații:

(i) Conținutul etapei inițiale (etapa zero) formează o componentă a șirului de ieșire;

(ii) Conținutul etapei i este mutat în starea $i-1$ pentru fiecare i, $i=\overline{1,L-1}$;

(iii) Noul conținut al etapei $L-1$ este bitul de feedback s_j, care este calculat prin XOR-ul conținuturilor anterioare ale unei mulțimi fixate de etape $0,1,\ldots,L-1$.

CIFRURI FLUIDE ADAPTATE PENTRU IMPLEMENTARE HARDWARE SAU SOFTWARE

Prin implementare hardware a unui algoritm înțelegem că sistemul de calcul conține circuite pentru a realiza toate operațiile algoritmului. Prin implementare software înțelegem că rezultatul final este obținut prin efectuarea de către procesor a unei mulțimi de instrucțiuni într-o ordine stabilită de către o entitate exterioară sistemului de calcul.

Pe când cifrurile fluide bazate pe LFSR sunt bine adaptate pentru implementare hardware, acestea nu sunt tocmai potrivite pentru implementare software. Acest fapt a condus la elaborarea de cifruri fluide special proiectate pentru implementare software rapidă. În continuare, vom prezenta câteva sisteme de criptare fluidă ce admit implementare eficientă cel puțin la unul din aceste profiluri.

SISTEMUL DE CRIPTARE RC4

Sistemul de criptare RC4 (Rivest Code #4, cunoscut și ca Alleged RC4 sau ARC4) a fost elaborat în 1987 de R. Rivest pentru societatea RSA Security. Mai mult timp a fost utilizat pe larg, datorită simplității și vitezei de execuție a acestuia. Destinat scopurilor comerciale, el a fost secret, fiind accesibil numai după semnarea unui protocol de confidențialitate. În septembrie 1994 însă, un anonim publică codul sursă al lui RC4 pe un forum web, după care acesta se răspândește rapid, stârnind o serie de polemici referitor la drepturile intelectuale. Se consideră că în prezent există mai multe implementări ilegale.

SISTEMUL DE CRIPTARE RC4

Printre sistemele care au utilizat RC4 sunt aplicații precum Microsoft Windows, Kerberos, Bit Torrent, Skype, AOCE (Apple Open Collaboration Environment), WEP (Wired Equivalent Privacy) și WPA (Wi-Fi Protected Access) - protocoale de criptare utilizate în routerele wireless. RC4 este utilizat pentru criptarea fișierelor în protocoale, cum ar fi, RSA SecurePC, sau în cadrul sistemelor pentru comunicații (WEP, WPA pentru carduri, criptarea traficului de date sau a site-urilor web bazate pe protocolul de rețea SSL/TLS).

SISTEMUL DE CRIPTARE RC4

Sistemul RC4 este un sistem de criptare fluidă binar aditiv, orientat pentru implementare software. Una dintre ideile utilizate în construcția cifrurilor fluide se bazează pe generarea unei permutări pseudoaleatoare a unui șir de biți, după care din această permutare se extrage un șir pseudoaleator de cuvinte (șir de biți de lungime fixată). Algoritmul RC4 operează pe octeți și folosește acest principiu de proiectare a cifrurilor fluide. Fiecare octet al textului clar este combinat prin XOR cu un octet al cheii fluide, astfel obținându-se un octet de text criptat.

SISTEMUL DE CRIPTARE RC4

La generarea cheii fluide este utilizat conceptul de stare internă (secretă), care se descrie prin două componente:

- Un tablou de stare S cu 256 de elemente octeți - $S_0, S_1, \ldots, S_{255}$, numit S-boxă. Elementele tabloului S reprezintă o permutare a numerelor de la 0 la 255 (numere reprezentabile pe 8 biți). O cheie k de lungime variabilă între 1 și 256 octeți este utilizată pentru a inițializa S-boxa;
- Două numere din intervalul $[0,255]$ (adică octeți), notate prin „i" și, respectiv, „j", care au semnificație de indicatori spre locație. Incrementul i este indicele deterministic, care este incrementat cu 1 (modulo 256) la fiecare pas, iar j servește ca indice pseudoaleator, care este reînnoit în dependență de cheia k și S-boxa S.

SISTEMUL DE CRIPTARE RC4

Criptarea cu sistemul RC4 este realizează în două etape: la etapa întâi S-boxa este inițializată, folosind un algoritm de expandare a cheii, numit KSA (Key-Scheduling Algorithm), iar la etapa a doua este aplicat algoritmul PRGA (Pseudo-Random Generation Algorithm), care selectează aleator din S-boxă câte un octet al cheii fluide. Octetul cheii fluide este combinat prin XOR cu octetul corespunzător al textului clar.

În Figura 2.2.9 este ilustrată ideea utilizată în RC4. Primele două blocuri descriu procedura de inițializare, iar restul – permutările aplicate în construcția octeților cheii fluide. Permutările sunt efectuate atât timp, cât în textul clar există octeți necriptați.

SISTEMUL DE CRIPTARE RC4

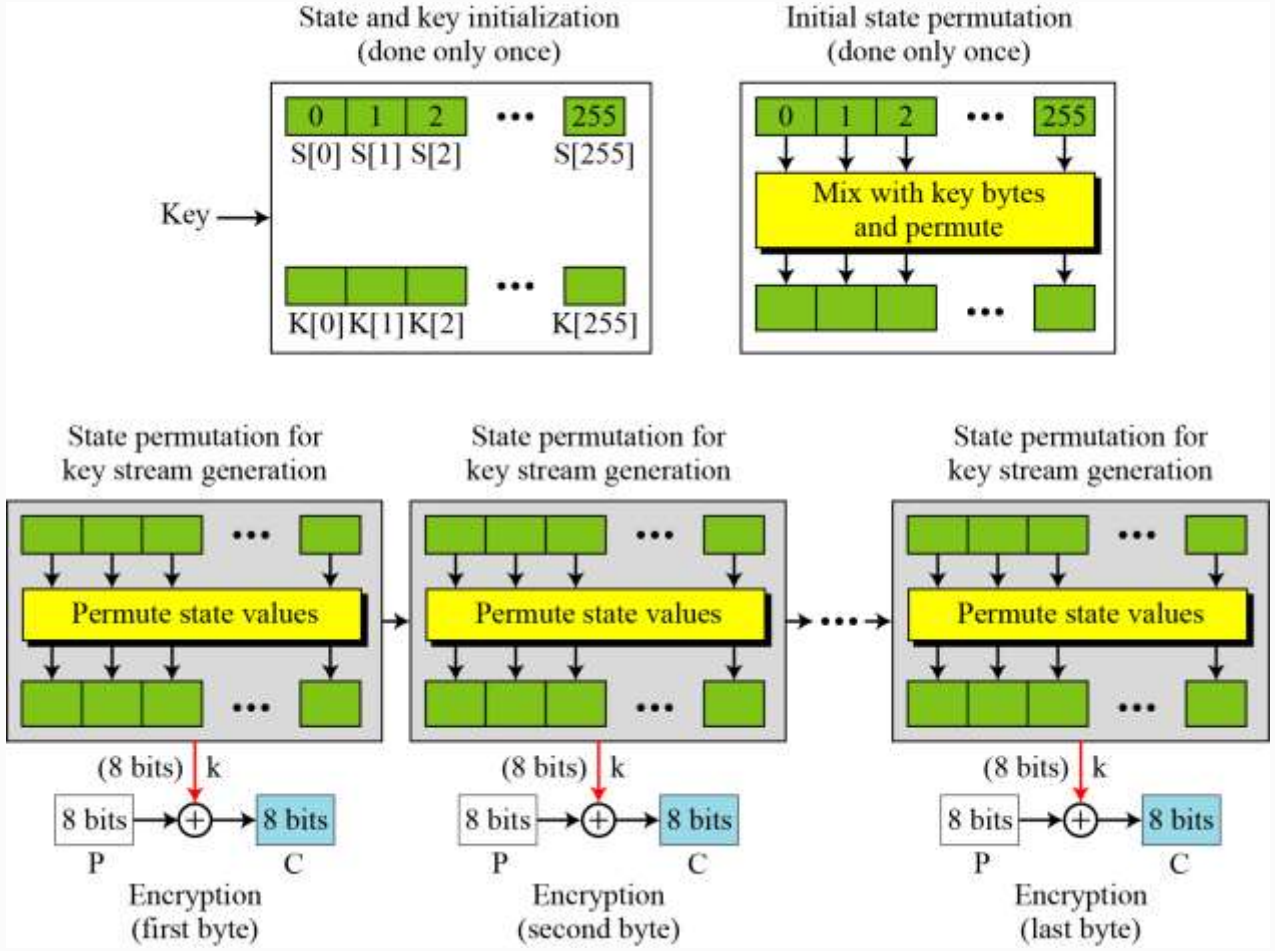

Figura 2.2.9. Ilustrarea etapelor algoritmului RC4

2.2.4.2.1 Algoritmul de inițializare KSA

Inițializarea procesului de criptare conform algoritmului RC4 necesită precizarea cheii secrete inițiale k în baza căreia sunt generați octeții cheii fluide. Dacă utilizatorul precizează cheia în cod Unicode, atunci aceasta este convertită în cod binar. Cheia secretă poate să aibă lungime între 1 și 256 de octeți, dar, de regulă, se introduc chei de lungime între 5 și 16 octeți. De exemplu, o cheie pe 40 de biți reprezintă un cod ASCII din 5 caractere. Cheii „pwd12" îi corespunde următorul echivalent binar:

01110000011101110110010000110001001100 10

SISTEMUL DE CRIPTARE RC4

Mai întâi, tabloul de stare S este inițializat cu valorile 0,1,...,255 (adică permutarea identică):

0	1	2	...	254	255

Fie $n \in \mathbb{N}, 1 \le n \le 256$, reprezintă numărul de octeți din cheia secretă inițială k (în majoritatea implementărilor n este în intervalul $[5,16]$). Dacă $n < 256$, prin repetarea octeților, cheia k este completată până lungimea acesteia va deveni egală cu 256 (am expandat cheia).

În continuare, folosind cheia expandată T, în tabloul S se efectuează 256 de permutări. Fiecare dintre cele 256 de elemente ale lui S este interschimbat cu elementul de indice j, unde j se determină în baza formulei $j := \mathrm{mod}\left(j + S_i + k_{\mathrm{mod}(i,n)}, 256\right)$ (valoarea inițială pentru j este $j = 0$), iar $k_{\mathrm{mod}(i,n)}$, $i = \overline{0,255}$ reprezintă octetul cu numărul i al cheii expandate de lungime 256 octeți.

Astfel, folosind cheia secretă k, algoritmul KSA generează permutarea pseudoaleatoare inițială a lui RC4 prin transpunerea unei permutări identice.

SISTEMUL DE CRIPTARE RC4

Algoritmul KSA de inițializare

Date de intrare:

k - cheia secretă de lungime n octeți

Date de ieșire:

S - S-boxa generată în baza lui k

/* Completarea inițială a tabloului de stare și a cheii extinse

Pentru $i := \overline{0,255}$ execută $\left\{ S_i := i,\ T_i := k_{\mathrm{mod}(i,n)} \right\}$

/* Permutarea octeților tabloului de stare S în baza octeților cheii expandate

$j := 0$

Pentru $i := \overline{0,255}$ execută

{

$j := \mathrm{mod}\left(j + S_i + T_i, 256 \right)$

$Interschimbă\left(S_i, S_j \right)$

}

2.2.4.2.2 Algoritmul PRGA de generare a cheii fluide

Permutarea inițială S obținută în urma aplicării algoritmului KSA, servește ca parametru de intrare pentru procedura PRGA, care generează octeții cheii fluide. Mai întâi, se inițializează cu zero indicatorii i, j, $i := 0,\ j := 0$. Pentru criptare și decriptare, algoritmul PRGA selectează din tabloul S câte un element, care constituie un octet al cheii fluide K.

La fiecare iterație se calculează valorile noi ale indicatorilor i, j, iar elementul S_i din tabloul S este interschimbat cu S_j. În continuare, valorile obținute pentru S_i și S_j se adună modulo 256, iar octetul $oct := S_{\mathrm{mod}(S_i+S_j,256)}$ este scris în tabloul K ce conține cheia fluidă (a se vedea *Figura 2.2.10*).

SISTEMUL DE CRIPTARE RC4

Iterațiile continuă până atunci când lungimea cheii fluide devine egală cu cea a textului clar. Fiecare element al lui S este permutat cel puțin o dată pe parcursul a 256 iterații.

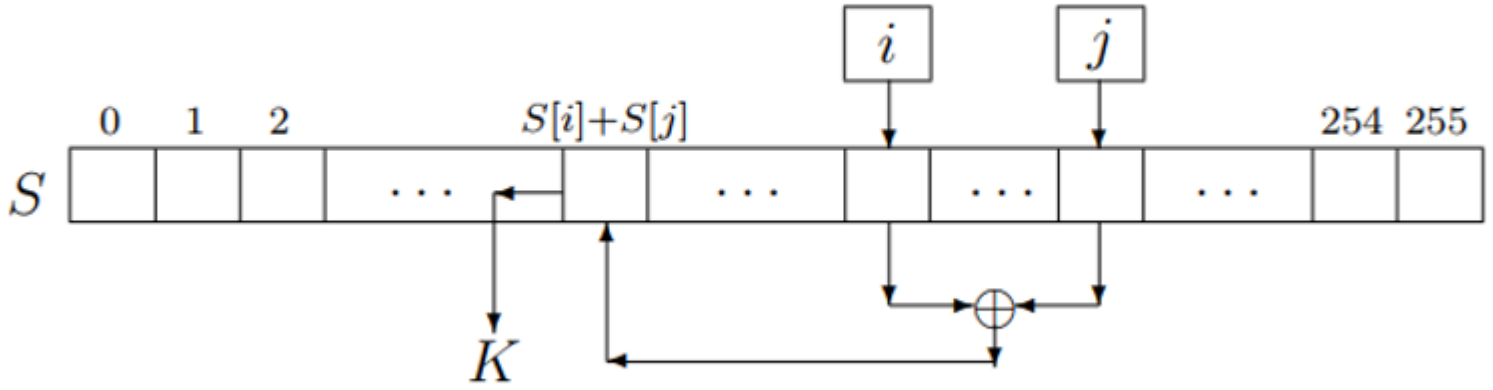

Figura 2.2.10. O etapă a algoritmului RC4

SISTEMUL DE CRIPTARE RC4

Algoritmul PRGA de generare a octeților cheii fluide și de criptare a octeților textului clar

Date de intrare:

S - S-boxa generată cu algoritmul KSA

lm - lungimea în octeți a textului clar

$\{m_r\}_{r=\overline{1,lm}}$ - octeții textului clar

Date de ieșire:

c - tablou ce conține octeții textului criptat

/* Permutarea în ciclu a octeților, generarea octeților cheii fluide și criptarea

$i := 0,\ j := 0,\ c := []$

Pentru $r := \overline{1,lm}$ execută

{

$i := \mathrm{mod}(i+1, 256),\ j := \mathrm{mod}(j+S_i, 256)$

$Interschimbă(S_i, S_j),\quad K := S_{\mathrm{mod}(S_i+S_j,256)},\quad c := [c, m_r \oplus K]$

}

SISTEMUL DE CRIPTARE RC4

La criptare, fiecare element al tabloului K (un octet) este combinat printr-un XOR cu octetul de text clar corespunzător, iar la decriptare – cu octetul de text criptat corespunzător.

2.2.4.2.3 Securitatea RC4

Sistemul de criptare RC4 este simplu și rapid în implementările software. El are circa $256! \times 256^2 = 2^{1700}$ stări posibile. Totuși, securitatea lui RC4 este considerată una redusă, din mai multe considerente, și nu se recomandă utilizarea sistemului în cadrul aplicațiilor moderne. Cheia fluidă generată are o ușoară preferință pentru anumite secvențe de octeți. În anul 2001, Fluhrer, Mantin și Shamir [9] au arătat că din toate cheile posibile RC4, primii octeți de ieșire nu sunt aleatori, oferind informații importante despre cheie.

SISTEMUL DE CRIPTARE RC4

Se recomandă ca în cadrul diferitor sesiuni să fie utilizate chei secrete diferite. În sistemele de criptare calitative, o măsură de securitate implementată constă în alegerea unui număr aleator nou (numit valoare nonce), care este folosit pentru criptarea fiecărui mesaj. Astfel, această valoare nonce face ca criptarea de două ori a aceluiași mesaj să genereze texte criptate diferite. O soluție sigură (care funcționează pentru orice sistem de criptare) pentru depășirea vulnerabilității lui RC4, constă în utilizarea unei chei pe termen lung din care, prin amestec cu o valoare nonce (după un algoritm prestabilit), se obține cheia necesară unei criptări. Însă mai multe aplicații care folosesc RC4, realizează o simplă concatenare a cheii cu valoarea nonce. Acest fapt a fost exploatat de Fluhrer, Mantin și Shamir pentru a sparge ulterior sistemul de criptare WEP (wired equivalent pruvacy) folosit în rețelele fără fir 802.11. Ulterior implementările sistemului RC4 s-au apărat, eliminând primii octeți (de regulă, 1024 octeți) din cheia fluidă, înainte de a o folosi.