

CRIPTANALIZA. REZULTATE ȘI TEHNICI MATEMATICE

Ediția I apărută la: Ed. Univ. Buc, 2004, ISBN 973575975-6.

Vasile PREDA, Emil SIMION și Adrian POPESCU

Ediția a doua 2011

Cuprins

1	INTRODUCERE	15
2	NOȚIUNI GENERALE	19
2.1.	Obiectul criptanalizei	19
2.2.	Criterii și standarde	20
2.2.1.	Beneficii ale standardelor	20
2.2.2.	Organisme de standardizare	21
2.2.3.	Standardele ISO 15408 și FIPS 140-2	22
2.3.	Modelul OSI (Open System Interconectation)	22
2.3.1.	Definirea nivelurilor rețelei	22
2.3.2.	Nivelul fizic	23
2.3.3.	Nivelul legătură date	23
2.3.4.	Nivelul rețea	24
2.3.5.	Nivelul transport	24
2.3.6.	Nivelul sesiune	24
2.3.7.	Nivelul prezentare	24
2.3.8.	Nivelul aplicație	25
2.3.9.	Protocolul TCP/IP	25
2.4.	Testarea sistemelor criptografice	25
2.4.1.	Introducere	25
2.4.2.	Programul de validare a modulelor criptografice	26
2.4.3.	Proceduri de certificare și autorizare	28
2.4.4.	Autoritate de certificare	28
2.5.	Procesul de selectare a modulelor criptografice	29
2.5.1.	Faza de planificare	29
2.5.2.	Faza de definire a cerințelor și specificațiilor de securitate	31
2.5.3.	Faza de achiziție	31
2.5.4.	Faza de operare	32
2.6.	Operații în criptanaliză	32

2.6.1.	Principii criptanalitice	32
2.6.2.	Criterii de evaluare	33
2.6.3.	Patru operații de bază ale criptanalizei	33
2.6.4.	Evaluare și spargere	34
2.7.	Clasificări ale atacurilor criptanalitice	38
2.7.1.	Tipuri de atac asupra algoritmilor de cifrare	38
2.7.2.	Tipuri de atac asupra cheilor	40
2.7.3.	Tipuri de atac asupra protocoalelor de autentificare	41
2.7.4.	Tipuri de atac asupra sistemului	42
2.7.5.	Atacuri hardware asupra modulelor criptografice	42
2.8.	Aplicații	43
3	TEORIA COMPLEXITĂȚII ALGORITMILOR	45
3.1.	Introducere	45
3.2.	Algoritmi și mașini Turing	45
3.3.	Teoria problemelor NP - complete	46
3.4.	Exemple de probleme NP - complete	48
3.5.	Limite actuale ale calculatoarelor	51
3.6.	Aplicații	52
4	ANALIZA STATISTICO-INFORMAȚIONALĂ	53
4.1.	Noțiuni teoretice	53
4.2.	Generatoare și teste statistice	54
4.2.1.	Generatoare uniforme	54
4.2.2.	Conceptul de test statistic	54
4.2.3.	Modele statistice pentru generatoare	56
4.2.4.	Teste elementare de aleatorism statistic	57
4.2.5.	Interpretarea rezultatelor testelor statistice	58
4.3.	Entropia variabilelor aleatoare discrete	59
4.4.	Surse de aleatorism de numere întregi	62
4.4.1.	Formula analitică a probabilității ideale a unei surse de aleatorism de numere întregi	62
4.4.2.	Metoda de calcul efectiv al lui p respectiv q	63
4.5.	Metode de decorelare	64
4.5.1.	Metode deterministe	64
4.5.2.	Metode nedeterministe	64
4.6.	Teste statistice de aleatorism	65
4.6.1.	Algoritmul de implementare al testului frecvenței	65
4.6.2.	Algoritmul de implementare al testului serial	66
4.6.3.	Algoritmul de implementare al testului succesiunilor	67

4.6.4.	Algoritmul de implementare al testului autocorelației temporale	68
4.6.5.	Algoritmul de implementare al testului autocorelațiilor temporale	68
4.6.6.	Algoritmul de implementare al testului autocorelației circulare	69
4.6.7.	Algoritmul de implementare al testului autocorelațiilor circulare	70
4.6.8.	Algoritmul de implementare al testului poker	70
4.6.9.	Algoritmul de implementare al testului <i>CUSUM</i> (sumelor cumulate)	72
4.6.10.	Algoritmul de implementare al testului de aproximare a entropiei	75
4.6.11.	Algoritmul de implementare al testului lui Maurer (1992) și testul entropiei	76
4.6.12.	Algoritmul de implementare al testului χ^2	78
4.6.13.	Algoritmul de implementare al testului Kolmogorov-Smirnov	80
4.6.14.	Testul spectral (transformarea Fourier discretă)	81
4.6.15.	Teste de corelație	83
4.6.16.	Algoritmul de implementare al testului corelațiilor temporale și circulare	83
4.6.17.	Creșterea sensibilității algoritmilor de testare statistică	83
4.7.	Teste de aleatorism algoritmic	85
4.7.1.	Scurt istoric	85
4.7.2.	Măsurarea complexității	86
4.7.3.	Complexitatea segmentului	89
4.7.4.	Complexitatea segmentului ca măsură a aleatorismului	90
4.8.	Teste de necorelare algoritmică	92
4.8.1.	Formularea problemei	92
4.8.2.	Principii de test	92
4.9.	Teste de verificare a jocurilor de tip Casino	93
4.9.1.	Metoda $3-\sigma$ pentru ruletă	94
4.9.2.	Metoda $3-\sigma$ pentru diferențe la ruletă	94
4.9.3.	Metoda X^2 pentru ruletă	94
4.9.4.	Metoda X^2 aplicată diferențelor pentru ruletă	95
4.9.5.	Metoda X^2 pentru jocurile de tip loto	95
4.10.	Aplicații	95
5	CODIFICAREA IN ABSENȚA PERTURBAȚIEI	99
5.1.	Introducere	99
5.2.	Codificarea în absența perturbației	99
5.3.	Codurile Fano și Huffman	102
5.3.1.	Algoritmul de implementare a codului Fano	102
5.3.2.	Algoritmul de implementare a codului Huffman	102

5.4.	Coduri optime	103
5.5.	Aplicații	104
6	CRIPTANALIZA CIFRURILOR CLASICE	109
6.1.	Substituția simplă și multiplă	109
6.1.1.	Substituția simplă	109
6.1.2.	Substituția multiplă	111
6.2.	Substituția polialfabetică	113
6.2.1.	Caracteristicile și identificarea sistemelor de substituție polialfabetică	113
6.2.2.	Atacul sistemelor polialfabetice	114
6.3.	Soluția unui cifru de substituție	114
6.4.	Transpoziția	115
6.5.	Sisteme mixte	115
6.6.	Proceduri de identificare a sistemului	115
6.6.1.	Funcția Kappa	116
6.6.2.	Funcția Chi	117
6.6.3.	Funcția Psi	118
6.6.4.	Funcția Phi	120
6.7.	Funcții simetrice de frecvență a caracterelor	121
6.8.	Atac stereotip asupra cifrurilor de substituție	122
6.9.	Atac de tip frecvență maximă a cifrurilor de substituție	122
6.10.	Concluzii	123
6.11.	Aplicații	124
7	CRIPTANALIZA CIFRURILOR FLUX	127
7.1.	Atacul generatoarelor pseudoaleatoare	127
7.2.	Criptanaliza liniară	128
7.2.1.	Complexitatea liniară	128
7.2.2.	Algoritmul Berlekamp-Massey. Rezultate teoretice	134
7.2.3.	Implementarea algoritmului Berlekamp-Massey	134
7.2.4.	Testul Berlekamp ca test statistico-informațional	135
7.3.	Metoda corelației	137
7.4.	Metoda corelației rapide	137
7.4.1.	Transformata Walsh-Hadamard	137
7.4.2.	Testul statistic Walsh-Hadamard	141
7.4.3.	Caracterizarea proprietăților criptografice	144
7.5.	Atacul Siegenthaler	148
7.6.	Atacul consistenței liniare	148
7.7.	Metoda sindromului linear	149

7.7.1.	Formularea problemei	149
7.7.2.	Preliminarii teoretice	149
7.7.3.	Algoritmul Sindromului Linear	150
7.7.4.	Numere critice și numere de rafinare	150
7.8.	Corecția iterativă a erorii	154
7.8.1.	Prezentare generală	154
7.8.2.	Prezentarea algoritmilor de corecție iterativă	155
7.8.3.	Rezultate experimentale	157
7.8.4.	Concluzii	157
7.9.	Algoritm de criptanaliză diferențială	159
7.10.	Câteva tehnici de proiectare	160
7.10.1.	Transformarea neliniară feed-forward	160
7.10.2.	Generatorul Geffe	160
7.10.3.	Generatorul Jennings	161
7.10.4.	Generatorare cu tact controlat	162
7.10.5.	Generatoare cu ceasuri multiple	165
7.10.6.	Generatoare autodecimate	166
7.11.	Exemplu de atac criptanalitic	166
7.12.	Aplicații	168
8	CRIPTANALIZA CIFRURILOR BLOC	173
8.1.	Introducere și concepte generale	173
8.2.	Securitatea și complexitatea atacurilor	174
8.3.	Criterii de evaluare a cifrurilor bloc	174
8.4.	Moduri de operare	174
8.4.1.	Modul ECB (electronic code-book)	175
8.4.2.	Modul CBC (cipher-block chaining)	176
8.4.3.	Modul CFB (cipher feedback)	179
8.4.4.	Modul OFB (output feedback)	180
8.4.5.	Modul BC (block chaining)	182
8.4.6.	Modul BC cu sumă de control (BC-checksum)	182
8.4.7.	Modul OFBNLF (output feedback block with a nonlinear function)	182
8.4.8.	Cascade de cifruri și cifrări multiple	183
8.5.	Generarea tabelor de substituție	186
8.6.	Criptanaliza diferențială	187
8.7.	Criptanaliza liniară	187
8.8.	Alte metode	187
8.9.	Implementări și rezultate experimentale	188
8.9.1.	Implementarea standardului de cifrare A.E.S.	188

8.9.2.	Testarea algoritmului AES	189
8.9.3.	Rezultate experimentale	189
8.9.4.	Interpretarea rezultatelor	192
8.10.	Concluzii	192
8.11.	Aplicații	193
9	CRIPTANALIZA CIFRURILOR CU CHEI PUBLICE	197
9.1.	Principii de bază	197
9.1.1.	Introducere	197
9.1.2.	Securitatea algoritmilor cu cheie publică	198
9.1.3.	Comparații ale algoritmilor asimetrice și a algoritmilor simetrici	198
9.2.	Algoritmi de tip rucsac	199
9.2.1.	Algoritmi rucsac supercrescător	199
9.2.2.	Crearea cheii publice din cheia privată	199
9.2.3.	Cifrarea	200
9.2.4.	Descifrarea	200
9.2.5.	Implementarea efectivă	200
9.3.	Algoritmul RSA	200
9.3.1.	Descrierea principiilor de cifrare și descifrare	200
9.3.2.	Viteza algoritmilor tip RSA	201
9.3.3.	Securitatea RSA-ului	203
9.3.4.	Tipuri de atacuri asupra algoritmilor RSA	204
9.3.5.	Trape în generarea cheilor RSA	209
9.4.	Algoritmul Pohlig-Hellman	209
9.5.	Algoritmul Rabin	210
9.6.	Algoritmul ElGamal	211
9.7.	Curbe eliptice	211
9.8.	Aplicații practice	213
9.9.	Teste de primalitate și metode de factorizare	214
9.9.1.	Teste de primalitate	214
9.9.2.	Metode de factorizare	217
9.9.3.	Metode de generare a numerelor prime	217
9.10.	Infrastructura Cheilor Publice (PKI)	218
9.10.1.	Elementele PKI	218
9.10.2.	Ghid de folosire a tehnologiei PKI în rețele deschise	219
9.10.3.	Riscuri ale utilizării tehnologiei PKI	220
9.10.4.	Standarde ale PKI	221
9.11.	Aplicații	221

10	CRIPTANALIZA SEMNĂTURILOR DIGITALE	225
10.1.	Prezentare generală	225
10.2.	Noțiuni preliminare	226
10.3.	Funcții hash	228
10.3.1.	Generalități	228
10.3.2.	Algoritmi hash	229
10.3.3.	Funcții hash bazate pe cifruri bloc	230
10.3.4.	Funcții hash nebazate pe cifruri bloc	230
10.4.	Modalități de realizare a semnăturilor digitale	231
10.4.1.	Aplicarea criptosistemelor simetrice	231
10.4.2.	Aplicarea criptosistemelor asimetrice	232
10.4.3.	Apelarea la funcții hash unidirecționale	232
10.4.4.	Semnături digitale convenționale (normale)	233
10.5.	Alte tipuri de semnături digitale	235
10.5.1.	Semnătura invizibilă	235
10.5.2.	Semnături fail-stop	235
10.6.	Legislația în domeniu	235
10.7.	Aplicații	236
11	CRIPTANALIZA PROTOCOALELOR CRIPTOGRAFICE	237
11.1.	Protocole elementare	237
11.1.1.	Protocole de schimb de chei	237
11.1.2.	Protocole de autentificare	241
11.1.3.	Autentificarea și schimbul de chei	244
11.1.4.	Protocole de transfer orb	248
11.1.5.	Analiza formală a protocoalelor de autentificare și a protocoalelor de schimb de chei	248
11.1.6.	Divizarea secretului	249
11.1.7.	Partajarea secretului	249
11.2.	Protocole avansate	249
11.2.1.	Protocol de tip demonstrație convingătoare fără detalii	249
11.2.2.	Protocol de tip dezvăluire minimă	250
11.2.3.	Protocol de tip dezvăluire zero	250
11.2.4.	Protocole de tip transfer bit și aplicații	250
11.2.5.	Alte protocoale avansate	254
11.3.	Divizarea și partajarea secretelor	254
11.3.1.	Protocol de divizare a secretului	255
11.3.2.	Protocolul de partajare LaGrange	255
11.3.3.	Protocolul de partajare vectorial	256
11.3.4.	Protocolul de partajare Asmuth-Bloom	256

11.3.5. Protocolul de partajare Karnin-Greene-Hellman	256
11.3.6. Atacuri asupra protocoalelor de partajare (divizare) a secretului	257
11.4. Exemple de implementare	257
11.4.1. Scheme de autentificare	257
11.4.2. Algoritmi de schimb al cheilor	260
11.5. Aplicații	264
12 CRIPTANALIZA SISTEMELOR DE CIFRARE ANALOGICE	265
12.1. Formularea problemei	265
12.2. Calcul Operațional	265
12.2.1. Transformata Laplace	266
12.2.2. Transformata Fourier	267
12.2.3. Transformata Fourier Discretă	269
12.2.4. Transformata Cosinus Discretă	271
12.2.5. Transformata Walsh	271
12.2.6. Transformata z	272
12.3. Caracterizarea variabilelor aleatoare	273
12.4. Conversia Analogic/Digital	274
12.4.1. Modulația în puls	274
12.5. Cifrarea Analogică	275
12.5.1. Inversarea spectrului	275
12.5.2. Rotirea spectrului inversat	276
12.5.3. Amestecarea spectrului	277
12.5.4. Multiplexarea în timp	279
12.6. Aplicații	279
13 MANAGEMENTUL CHEILOR CRIPTOGRAFICE	281
13.1. Managementul cheilor	281
13.2. Generarea cheilor	283
13.3. Protecția cheilor criptografice	284
13.3.1. Cheie utilizator	284
13.3.2. Arhivarea cheilor	285
13.3.3. Distrugerea cheilor	285
13.4. Lungimea cheilor criptografice	286
13.5. Aplicații	287
A METODELE ȘI TEHNICILE DE PROGRAMARE	289
A.1. Structuri de date	289
A.2. Alocarea memoriei	290

A.3. Recursivitate	290
A.4. Metoda backtracking	290
A.5. Tehnica Divide et Impera	291
A.6. Tehnica branch and bound	291
A.7. Programarea dinamică	292
A.8. Tehnica greedy	292
A.9. Aplicații	293
B ELEMENTE DE TEORIA PROBABILITĂȚILOR	295
B.1. Caracteristici ale variabilelor aleatoare	295
C STATISTICĂ DESCRIPTIVĂ	297
C.1. Momentele unei variabile aleatoare	297
C.2. Teoria selecției	298
C.3. Aplicații	299
D TEORIA ESTIMAȚIEI	301
D.1. Tipuri de estimatori	301
D.2. Margini inferioare ale estimatorilor	302
D.3. Estimația de verosimilitate maximă	304
D.4. Estimația Bayesiană	304
E REPATIȚII STATISTICE	307
E.1. Repartiții continue	307
E.1.1. Repartiția normală	307
E.1.2. Repartiția lognormală	308
E.1.3. Repartiția uniformă	308
E.1.4. Repartiția exponențială	309
E.1.5. Repartiția gama	310
E.1.6. Repartiția beta	312
E.1.7. Repartiția Cauchy	313
E.2. Distribuții discrete	313
E.2.1. Distribuția Bernoulli	313
E.2.2. Distribuția binomială	313
E.2.3. Distribuția Poisson	314
E.2.4. Distribuția hipergeometrică	314
E.2.5. Distribuția geometrică	314
E.3. Calculul numeric al cuantilelor	314
E.3.1. Cuantila repartiției normale	315
E.3.2. Cuantilele repartiției chi-pătrat	315

F	SERII DINAMICE STAȚIONARE	317
F.1.	Exemple de serii dinamice	317
F.2.	Procese stochastice	317
F.3.	Staționaritate și strict staționaritate	319
F.3.1.	Relația dintre Staționaritate și Strict Staționaritate	320
F.4.	Estimarea și eliminarea componentelor tendință și sezoniere	322
F.4.1.	Eliminarea tendinței în absența componenetei sezoniere	323
F.4.2.	Eliminarea simultană a componentelor tendință și sezoniere	325
F.5.	Funcția de autocovarianță a unui proces staționar	327
F.5.1.	Funcția de autocovarianță de selecție	329
G	MODELUL AUTOREGRESIV-MEDIE MOBILĂ	331
G.1.	Modelul autoregresiv AR(p)	331
G.2.	Modelul medie mobilă MA(q)	332
G.3.	Modelul ARMA(p,q)	332
G.4.	Modelul ARIMA(p,d,q)	333
G.5.	Probleme puse proceselor ARIMA(p,d,q)	333
H	SIMULAREA VARIABILELOR ALEATOARE	335
H.1.	Tehnici de simulare	335
H.2.	Legea tare a numerelor mari	336
I	ELEMENTE DE TEORIA NUMERELOR	339
I.1.	Teste de primalitate	339
I.2.	Lema chinezescă a resturilor	340
I.3.	Numărul de numere prime	341
I.4.	Simbolurile lui Legendre și Jacobi	341
	BIBLIOGRAFIE	343

Capitolul 8

CRIPTANALIZA CIFRURILOR BLOC

*Nearly every inventor of a cipher system was been convinced of the unsolvability of his brain child.
David Kahn*

8.1. Introducere și concepte generale

Prezentul capitol face o prezentare a noțiunii de cifru bloc, a modurilor de operare precum și a principalelor caracteristici ale acestora. În finalul capitolului se prezintă o serie de tehnici și metode de criptanaliză a cifrurilor bloc care vor fi exemplificate pe noul standard de cifrare bloc AES (*Advanced Encryption Standard*).

Cifrurile bloc procesează informația pe blocuri de o lungime stabilită apriori. În cele ce urmează vom nota prin n lungimea blocului procesat (exprimată în biți), V_n spațiul vectorilor n dimensionali și prin \mathcal{K} spațiul cheilor. Un bloc de text clar se va nota prin M iar un bloc de text cifrat se va nota prin C .

Definiția 8.1.1. Un cifru bloc pe n biți este o funcție $E : V_n \times K \rightarrow V_n$ astfel încât pentru orice cheie $k \in K$ funcția $E(\cdot, k)$ este o funcție inversabilă (funcția de cifrare cu ajutorul cheii k) din V_n în V_n . Funcția inversă este funcția de decifrare și va fi notată prin $D_K(\cdot) = E_K^{-1}(\cdot)$.

Reamintim că a *sparge un cifru* nu înseamnă în mod obligatoriu de a găsi o cale practică astfel încât un interceptor să recupereze textul clar numai din criptograme. În cadrul criptografiei academice, regulile sunt relaxate considerabil. A sparge un

cifru înseamnă a găsi o slăbiciune care poate fi exploatată pentru recuperare cheii și/sau a textului cu o complexitate mai mică decât atacul brut.

8.2. Securitatea și complexitatea atacurilor

Obiectivul unui cifru bloc este să asigure confidențialitatea. Obiectivul core-spunzător al adversarului este să descopere textul original din textul cifrat.

Un cifru bloc este *total spart* dacă a fost descoperită cheia și *spart parțial* dacă adversarul poate reconstitui o parte din textul clar, dar nu și cheia din textul cifrat.

Pentru a evalua securitatea unui cifru bloc se obișnuiește ca întotdeauna să se presupună că adversarul:

- i) are acces la toate datele transmise prin canalul de text cifrat;
- ii) (presupunerea lui Kerckhoff): știe toate detaliile despre funcția de cifrare, mai puțin cheia secretă.

Cele mai importante clase de atacuri pentru cifrurile cu chei simetrice sunt:

- atac pe baza textului cifrat;
- atac pe baza textului clar/cifrat;
- atac pe baza textului clar/cifrat ales de criptanalist;
- atac pe baza cheilor.

8.3. Criterii de evaluare a cifrurilor bloc

Următoarele criterii pot fi folosite pentru a evalua cifrurile bloc:

- nivelul de securitate estimat (pe baza rezistenței la anumite tipuri de atacuri);
- mărimea cheii;
- mărimea blocului de date care se cifrează;
- complexitatea funcției de cifrare;
- expansiunea datelor;
- propagarea erorilor;
- viteza de cifrare a datelor.

8.4. Moduri de operare

Există două moduri principale de utilizare în practică a algoritmilor simetrici: cifrarea bloc și cifrarea secvențială. Cifrarea bloc operează cu blocuri de text clar și cifrat-de regulă de 64 de biți, uneori chiar mai mari. Cifrarea secvențială operează cu secvențe de text clar și cifrat de un bit sau octet. În cazul cifrării bloc, același bloc de text clar va fi cifrat de fiecare dată în același bloc de text cifrat, folosind

aceeași cheie. În cazul cifrării secvențiale, secvențele similare de text clar vor fi cifrate diferit în cazul unor cifrări repetate.

Modurile de cifrare constituie combinații ale celor două tipuri de bază, unele folosind metode feedback, altele realizând simple operații. Aceste operații sunt simple, deoarece securitatea este atributul cifrării și nu al modului în care se realizează schema de cifrare. Mai mult, modul de realizare a cifrării nu duce la compromiterea securității date de algoritmul de bază.

Un cifru bloc cifrează textul în blocuri de n biți de mărimi fixe. În general este folosită valoarea $n = 64$ biți. Cele mai cunoscute moduri de operare sunt ECB (*electronic code book*), CBC (*cipher block chaining*), CFB (*cipher feedback block*) și OFB (*output feedback block*). În funcție de modul de operare al cifrului bloc se vor aplica atacuri specifice. Vom nota în cele ce urmează, cu E_k se notează funcția de cifrare a blocului în timp ce cu D_k notăm funcția de descifrare.

8.4.1. Modul ECB (electronic code-book)

Modul ECB este cea mai obișnuită formă de cifrare bloc: un bloc de text clar este transformat într-un bloc de text cifrat, fiecare bloc fiind cifrat independent și fiecare cheie fiind diferită de celelalte. Dacă același bloc de text clar se cifrează întotdeauna în același bloc de text cifrat, teoretic este posibilă o carte de coduri în care să se facă asocierea text clar-text cifrat. Pentru blocuri de 64 de biți rezultă un număr de 2^{64} intrări în cartea de coduri- mărime prea mare pentru a permite memorarea și manipularea.

Modul ECB este cel mai simplu mod de lucru, fiecare bloc de text clar fiind cifrat independent. Cifrarea se poate face luând aleator blocuri din cadrul fișierului. Acest mod de cifrare este indicat pentru cifrarea documentelor care sunt accesate aleator, ca de exemplu baze de date, unde fiecare înregistrare poate fi adăugată, ștersă, cifrată sau descifrată independent de celelalte.

Problema ridicată de ECB este aceea că, dacă un criptanalist are pentru câteva mesaje și textul clar și pe cel cifrat, poate afla codul, fără a deține și cheia de cifrare.

Padding-ul

Completarea blocurilor (*padding*) este folosită atunci când mesajele nu se împart exact în blocuri de 64 de biți. Se completează ultimul bloc cu un model zero-unu, alternând cifrele de 0 și 1, astfel încât blocul să devină complet.

Algoritmul ECB

Intrare: Cheia K de k biți, mesajul clar $M = M_1, \dots, M_t$ pe blocuri de n biți.

Ieșire: Textul cifrat $C = C_1, \dots, C_t$ care ulterior se descifrează pentru a descoperi textul original.

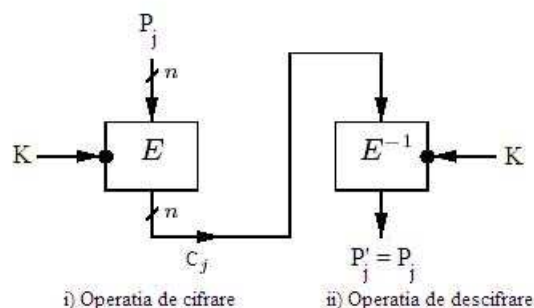


Figura 8.1: Modul de lucru ECB.

1. *Cifrarea*: pentru orice $i = 1, \dots, t$ avem: $C_i = E_k(M_i)$.
2. *Descifrarea*: pentru orice $i = 1, \dots, t$ avem: $M_i = D_k(C_i)$.

Proprietăți ale modului de operare ECB

Următoarele proprietăți rezultă din modul de construcție al cifrului bloc tip ECB.

1. Din texte identice rezultă texte cifrate identice.
2. Dependența în lanț: blocurile sunt cifrate independent de alte blocuri.
3. Propagarea erorilor: una sau mai multe erori într-un singur bloc de texte cifrate afectează descifrarea numai a acelui bloc.

8.4.2. Modul CBC (cipher-block chaining)

Acest mod folosește un mecanism *feedback*, deoarece rezultatul cifrării unui bloc anterior revine prin buclă și intervine în cifrarea blocului curent. Cu alte cuvinte, blocul anterior este folosit pentru a modifica cifrarea următorului bloc. În acest fel, textul cifrat nu mai depinde doar de textul clar, ci și de modul de cifrare al blocului anterior.

În modul CBC, textul clar, înainte de a intra în modul de cifrare propriu-zis, este însumat mod 2 cu blocul de text cifrat anterior. Figura 8.2 prezintă operația de cifrare/descifrare în modul CBC.

După ce blocul de text clar este cifrat, textul cifrat rezultat este stocat într-un registru al buclei de reacție. Înainte ca următorul text clar să fie cifrat, el este sumat mod 2 cu blocul din registrul de reacție și devine următoarea intrare în rutina de cifrare. După cifrare, conținutul registrului este înlocuit cu blocul cifrat. În acest fel, cifrarea blocului i depinde de toate cele $i - 1$ blocuri anterioare.

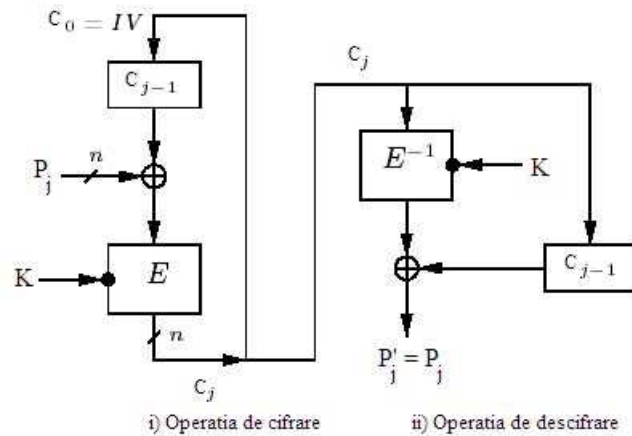


Figura 8.2: Modul de lucru CBC.

În procesul de descifrare (care este exact procesul invers cifrării), textul cifrat este descifrat normal și depozitat în registrul de reacție. După ce următorul bloc este descifrat, el este făcut sumă mod 2 cu conținutul registrului de reacție.

Din punct de vedere matematic, procesul de cifrare arată astfel:

$$C_i = E_k(P_i \oplus C_{i-1}).$$

Ecuțiile corespunzătoare operației de descifrare sunt:

$$P_i = C_{i-1} \oplus D_k(C_i).$$

Vectorul de inițializare

Modul de lucru CBC face ca același text să se transforme în blocuri diferite de text cifrat. Două mesaje identice se vor transforma în același mesaj cifrat. Mai mult, două mesaje care încep la fel vor fi cifrate identic până la prima diferență. Prevenirea acestui lucru se poate face cifrând primul bloc cu un vector de date aleator. Acest bloc de date aleatoare se numește vector de inițializare notat cu IV , numit și variabilă de inițializare sau valoare inițială pentru înlănțuire. Vectorul IV nu are nici un înțeles de sine stătător, el doar face cifrarea oricărui mesaj, unică. Când receptorul descifrează acest bloc, el îl utilizează pentru inițializarea registrului buclei de reacție.

Vectorul de inițializare nu trebuie să fie secret acesta poate fi transmis în clar împreună cu mesajul cifrat. Dacă acest lucru pare greșit, să considerăm că avem

un mesaj din câteva blocuri, B_1, B_2, \dots, B_i , astfel încât B_1 este cifrat cu IV , B_2 este cifrat utilizând ca vector de inițializare textul cifrat de la B_1 , etc. Dacă avem n blocuri, sunt $n - 1$ vectori de inițializare expuși, chiar dacă vectorul original este ținut secret.

Padding-ul

Completarea blocurilor este analoagă celei de la modul ECB. Există însă aplicații în care textul cifrat trebuie să aibă aceeași mărime ca textul clar. Este foarte probabil ca fișierul ce urmează să fie cifrat să fie pus în aceeași locație de memorie. În acest caz rămâne un ultim bloc, mai mic, a cărui cifrare se va face diferit. Se presupune că ultimul bloc are j biți. După cifrare, ultimul bloc întreg, se mai cifrează încă o dată, se selectează ultimii j biți din rezultat și se face suma mod 2 cu blocul incomplet.

Algoritmul CBC

Intrare: Cheia K pe k biți, vectorul inițial IV de n biți, mesajul clar $M = M_1, \dots, M_t$ pe blocuri de n biți.

Ieșire: Textul cifrat $C = C_1, \dots, C_t$ care ulterior se descifrează pentru a descoperi textul original.

1. *Cifrarea:* $C_0 = IV$, și recursiv avem

$$C_j = E_k(C_{j-1} \oplus M_j).$$

2. *Descifrarea:* $C_0 = IV$, pentru orice $j = 1, \dots, t$ avem

$$M_j = C_{j-1} \oplus D_k(C_j).$$

Proprietăți ale modului de operare CBC

Au loc următoarele proprietăți ale modului de lucru CBC.

1. Texte identice: blocuri de texte cifrate, identice, rezultă când același text este cifrat sub aceeași cheie și cu același vector IV . Schimbând IV , cheia sau primul bloc de text, rezultă un text cifrat diferit.
2. Dependența în lanț: mecanismul în lanț face ca textul cifrat c_j să depindă de x_j și de blocurile de text precedente.
3. Propagarea erorilor: o singură eroare de un bit în blocul textului cifrat c_j afectează descifrarea blocurilor c_j și c_{j+1} .
4. Dacă o eroare (inclusiv pierderea unuia sau mai multor blocuri) apare în blocul c_j , dar nu în c_{j+1} , c_{j+2} este corect descifrat din x_{j+2} .

8.4.3. Modul CFB (cipher feedback)

În timp ce modul CBC procesează textul de n biți odată, unele aplicații cer ca unități de r biți ale textului să fie cifrate și transmise fără întârziere pentru un număr fixat $r \leq n$. În acest caz se folosește modul CFB.

Modul CFB poate fi privit ca un mod secvențial cu autosincronizare, însă la nivel de blocuri de text cifrat. Algoritmul CFB la nivel de bloc operează cu o coadă de mărime egală cu cea a blocului. În primă fază aceasta este inițializată cu VI , analog modului CBC.

Dacă mărimea blocului de cifrat este n , atunci modul CFB pe n biți are forma pentru cifrare

$$C_i = P_i \oplus E_k(C_{i-1}),$$

respectiv pentru descifrare

$$P_i = C_i \oplus E_k(C_{i-1}).$$

Figura 8.3 ilustrează această formă.

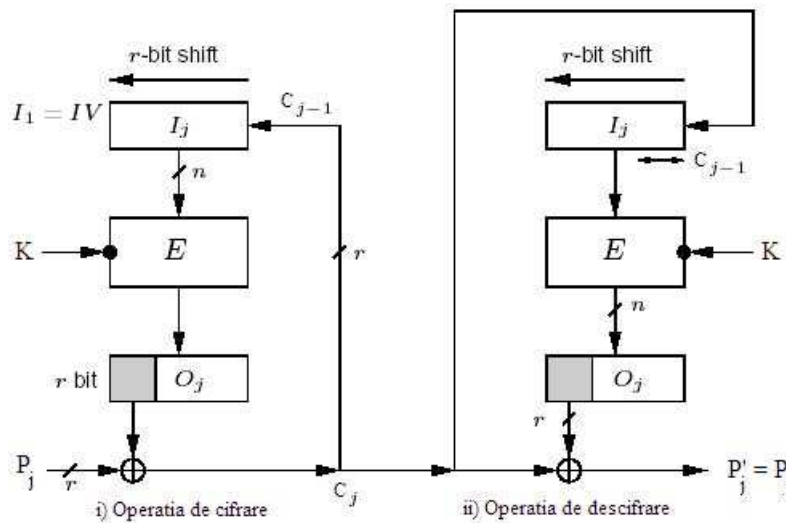


Figura 8.3: Modul de lucru CFB.

Algoritmul CFB

Intrare: Cheia K pe k biți, vectorul inițial IV de n biți, textul clar pe blocuri de r biți $M = M_1, \dots, M_t$ cu $r \leq n$.

Ieșire: Blocuri de text cifrate pe r biți care se descifrează pentru a descoperi textul clar.

1. *Cifrarea:* $I_1 = IV$ (I_j este valoarea de intrare a unui registru de deplasare). Fie blocul de r biți: x_1, \dots, x_r . Pentru $1 \leq j \leq r$ se execută:

a) $O_j = E_k(I_j)$ (calculează blocul cifrat de ieșire);

b) t_j = "cei mai la stânga" r biți ai lui O_j ;

c) $c_j = x_j \oplus t_j$;

d) $I_{j+1} = 2^r I_j + c_j \pmod{2^n}$.

2. *Descifrarea:* $I_1 = IV$, pentru $1 \leq j \leq r$ calculează: $x_j = c_j \oplus t_j$, unde t_j, O_j, I_j sunt calculați ca mai sus.

Observația 8.4.1. Dacă $r = n$ atunci recurența corespunzătoare cifrării este:

$$C_j = M_j \oplus E_k(M_{j-1}),$$

cu

$$M_0 = IV.$$

8.4.4. Modul OFB (output feedback)

Modul de operare OFB poate fi folosit în aplicațiile în care toate erorile de propagare trebuie să fie evitate. Este asemănător cu modul CFB și permite cifrarea blocurilor cu diverse mărimi, dar diferă în sensul că rezultatul funcției de cifrare E servește ca feedback. Acest mod se numește, uneori, cu reacție internă, deoarece bucla de reacție este independentă atât de textul clar, cât și de cel cifrat.

Dacă mărimea blocului este n , modul OFB pe n biți se scrie pentru cifrare:

$$C_i = P_i \oplus S_i;$$

iar pentru descifrare:

$$P_i = C_i \oplus S_i;$$

unde

$$S_i = E_k(S_{i-1}),$$

reprezintă starea care este independentă de textul clar și de cel cifrat.

Modul OFB pe n biți este prezentat în figura 8.4.

Există două versiuni:

-ISO (necesită un feedback de n biți și este mai sigură);

-FIPS (permite $r < n$ biți de feedback).

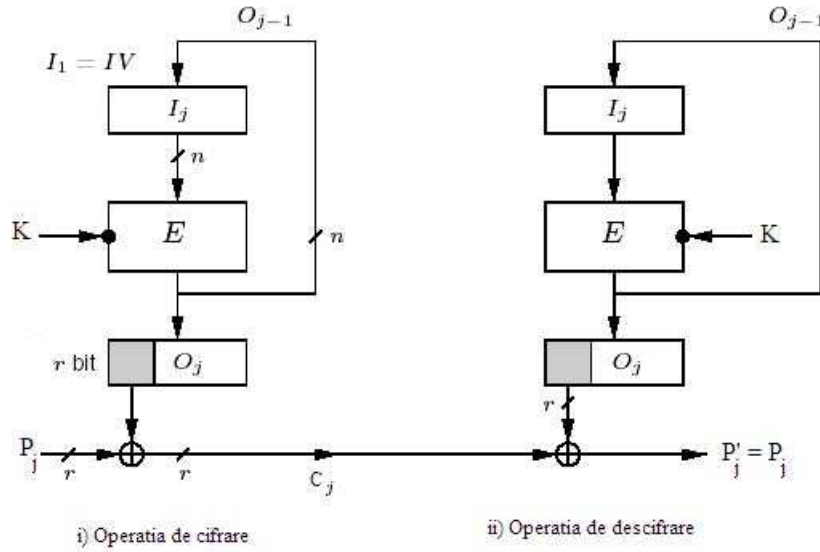


Figura 8.4: Modul de lucru OFB pe n biți.

Algoritm OFB (pentru ISO):

Intrare: Cheia k , vectorul IV pe n biți, blocuri de text clar pe r biți x_1, \dots, x_r ($1 \leq r \leq n$)

Ieșire: Blocuri de text cifrate c_1, \dots, c_r pe r biți.

1. *Cifrarea:* $I_1 = IV$. Pentru $1 \leq j \leq r$, dându-se x_j

a) $O_j = E_k(I_j)$.

b) t_j sunt cei mai de la stânga r biți ai lui O_j .

c) $c_j = x_j \oplus t_j$.

d) $I_{j+1} = O_j$.

2. *Descifrarea:* $I_1 \leftarrow IV$. Pentru $1 \leq j \leq r$ calculăm: $x_j = c_j \oplus t_j$, unde t_j, O_j, I_j sunt calculați ca mai sus.

Algoritm OFB (cu feedback de r biți) (pentru FIPS 81) :

Intrare: Cheia k pe k biți, IV pe n biți, blocurile de text pe r biți x_1, \dots, x_r cu $1 \leq r \leq n$:

Ieșire: Blocurile de text cifrate c_1, \dots, c_r .

Ca și în algoritmul OFB, dar înlocuind relația

$$I_{j+1} = O_j$$

cu

$$I_{j+1} = (2^r I_j + t_j) \bmod 2^n.$$

Observația 8.4.2. O variație a modului de lucru OFB este *modul de lucru contor* în care trecerea de la o stare la alta este dată de funcția de incrementare: $C_i = P_i \oplus S_i$; unde $S_i = E_k(\text{contor}[i])$, $\text{contor}[i + 1] = \text{contor}[i] + 1$.

8.4.5. Modul BC (block chaining)

A folosi un algoritm bloc în modul BC înseamnă a face suma mod 2 între intrarea în blocul de cifrare și suma mod 2 a tuturor blocurilor cifrate anterior. În același fel ca în CBC, procesul va fi declanșat de un *IV*.

Ecuatiile care descriu comportarea modului de operare BC sunt, pentru cifrare

$$C_i = E_k(M_i \oplus F_i),$$

unde

$$F_{i+1} = F_i \oplus C_i,$$

cu

$$F_1 = IV.$$

Operația de descifrare are forma:

$$P_i = F_i \oplus D_k(C_i).$$

8.4.6. Modul BC cu sumă de control (BC-checksum)

Modul BC cu sumă de control este similar modului BC simplu, deosebirea făcându-se la ultimul bloc în care se face suma mod 2 a blocurilor clare anterioare. Modul BC cu sumă de control ne asigură de faptul că orice modificare în structura cifrată a blocurilor se va evidenția cu ocazia descifrării ultimului bloc. Dacă acest ultim bloc are în componență și elemente de integritate atunci testul de integritate se va face foarte simplu.

8.4.7. Modul OFBNLF (output feedback block with a nonlinear function)

OFBNLF este o variantă atât a modului OFB, cât și a celui ECB, în care cheia se schimbă la fiecare bloc.

Ecuatiile care descriu comportarea acestui mod de operare sunt, pentru cifrare:

$$C_i = E_{k_i}(P_i),$$

pentru descifrare:

$$P_i = D_{k_i}(C_i),$$

unde:

$$K_i = E_k(K_{i-1}).$$

8.4.8. Cascade de cifruri și cifrări multiple

Există mai multe modalități de a combina algoritmi bloc pentru a obține noi algoritmi de cifrare. Scopul urmărit este de a încerca să se îmbunătățească securitatea și prin alte mijloace decât prin scrierea unui nou algoritm. Cifrarea multiplă este una din tehnicile combinării: se utilizează un algoritm pentru a cifra același text clar de mai multe ori, cu mai multe chei. Cascadarea are același principiu, dar utilizează mai multe chei.

Definiția 8.4.1. O cascadă de cifruri este concatenarea a $L \geq 2$ blocuri de cifruri (numite etape), fiecare cu chei independente.

Textul este intrarea primei etape iar ieșirea (rezultatul) etapei i este intrarea etapei $i + 1$ și ieșirea etapei L este ieșirea cascadei textului cifrat.

Definiția 8.4.2. Cifrarea multiplă este similară unei cascade de L cifruri identice, dar cheile etapei nu trebuie să fie independente și cifrurile etapei pot fi, fie un cifru bloc E , fie funcție de descifrare corespunzătoare $D = E^{-1}$.

Cifrarea dublă

Un mod simplu de îmbunătățire a securității unui algoritm bloc este de a cifra un bloc cu două chei diferite. Prima dată se cifrează blocul cu prima cheie, apoi întregul bloc de text cifrat rezultat se cifrează cu cea de a doua cheie. Descifrarea este procesul invers.

Definiția 8.4.3. Cifrarea dublă este definită ca $E(x) = E_{K_2}(E_{K_1}(x))$.

Blocul de text cifrat rezultat din dubla cifrare ar trebui să fie mult mai greu de spart utilizându-se o căutare exhaustivă. Pentru o lungime a cheii de n biți, față de 2^n variante posibile de chei la o singură cifrare, avem 2^{2^n} variante posibile, ceea ce pentru un algoritm de 64 de biți înseamnă 2^{128} de chei posibile.

Metoda Davies-Price. Este o variantă de CBC, a cărei reprezentare matematică este următoarea:

-pentru cifrare:

$$C_i = E_{k_1}(P_i \oplus E_{k_2}(C_{i-1})),$$

-respectiv pentru descifrare:

$$P_i = D_{k_1}(C_i) \oplus D_{k_2}(C_{i-1}).$$

Double OFB/Counter. Această metodă utilizează un algoritm bloc pentru a genera două șiruri de chei ce se utilizează la cifrarea unui text clar.

Astfel putem scrie pentru variabila internă S_i :

$$S_i = E_{k_1}(S_{i-1} \oplus I_1),$$

unde (I_1 este variabilă tip contor)

$$I_1 = I_1 + 1,$$

și respectiv pentru variabila internă T_i :

$$T_i = E_{k_2}(T_{i-1} \oplus I_2),$$

unde (I_2 este variabilă tip contor)

$$I_2 = I_2 + 1.$$

Funcția de cifrare este:

$$C_i = P_i \oplus S_i \oplus T_i,$$

Cifrarea triplă

Definiția 8.4.4. Cifrarea triplă este definită ca:

$$E(x) = E_{K_3}^{(3)}(E_{K_2}^{(2)}(E_{K_1}^{(1)}(x)))$$

unde $E_K^{(j)}$ este E_k sau $D_K = E_K^{-1}$.

Cifrarea triplă cu două chei. O idee mai bună, propusă de Tuchman, prelucrează un bloc de trei ori cu două chei: mai întâi cu prima cheie, apoi cu a doua, apoi din nou cu prima. El sugerează ca expeditorul să cifreze de trei ori, așa cum specifică algoritmul, iar destinatarul, la descifrare, va urma algoritmul invers: descifrare cu prima cheie, cifrare cu a doua, descifrare cu prima.

$$C = E_{k_1}(D_{k_2}(E_{k_1}(P))),$$

$$P = D_{k_1}(E_{k_2}(D_{k_1}(C))).$$

Această metodă se mai numește și EDF (*encrypt-decrypt-encrypt*). Dacă algoritmul are o cheie de n biți, atunci schema va avea o cheie de $2n$ biți. Modelul cifrare-descifrare a fost proiectat de IBM pentru a proteja implementările convenționale ale algoritmului și a fost adăugat la DES și în standardele ISO.

Cifrarea triplă cu trei chei. În cazul cifrării triple, numărul de trei chei este cel mai potrivit. Lungimea cheii este mare, iar memorarea sa nu poate fi o problemă.

$$C = E_{k_3}(D_{k_2}(E_{k_1}(P))),$$

$$P = D_{k_1}(E_{k_2}(D_{k_3}(C))).$$

Alte variante de cifrare triplă.

Inner CBC

Se cifrează întregul mesaj în modul CBC, de trei ori diferit. Sunt necesari trei vectori de inițializare diferiți.

Operația de cifrare are următoarea formă:

$$C_i = E_{k_3}(S_i \oplus C_{i-1}),$$

unde

$$S_i = D_{k_2}(T_i \oplus S_{i-1}),$$

și

$$T_i = E_{k_1}(P_i \oplus T_{i-1}).$$

Pentru descifrare se folosește următoarea formulă:

$$P_i = T_{i-1} \oplus D_{k_1}(T_i),$$

unde

$$T_i = S_{i-1} \oplus E_{k_2}(S_i),$$

și

$$S_i = C_{i-1} \oplus D_{k_3}(C_i).$$

iar C_0 , S_0 și T_0 sunt vectori de inițializare.

Outer CBC

Se cifrează întregul fișier în modul CBC, de trei ori diferit. Este necesar un singur vector de inițializare.

Pentru cifrare:

$$C_i = E_{k_3}(D_{k_2}(E_{k_1}(P_i \oplus C_{i-1}))),$$

respectiv pentru descifrare:

$$P_i = C_{i-1} \oplus D_{k_1}(E_{k_2}(D_{k_3}(C_i))).$$

Ambele moduri necesită mai multe resurse decât în cazul unei singure cifrări: mai mult *hardware* sau mai mult timp.

8.5. Generarea tabelelor de substituție

Tabele de substituție, cunoscute sub acronimul de *S-box*, joacă un rol important în proiectarea unui cifru bloc. O tabelă de substituție S este o funcție bijectivă de la $GF(2^n)$ la $GF(2^n)$ (uzual $n = 8$) cu următoarele proprietăți:

- grad mare de neliniaritate;
- rezistență la criptanaliza diferențială;
- construcție și computabilitate eficientă.

Următorul algoritm generează tabele de substituție din $GF(2^n)$ cu proprietățile menționate mai sus. Algoritmul își are originea în Nyberg [52].

PAS 0. Se alege un polinom ireductibil $m(x)$ generatorul lui $GF(2^n)$, \mathbf{A} matrice inversabilă din $M_{n \times n}(\mathbf{Z}_2)$, \mathbf{b} vector din \mathbf{Z}_2^n cu ponderea Hamming $\frac{n}{2}$.

PAS 1. Se alege $F(x)$ din $GF(2^n)$ de forma x^{-1} sau x^{2^k+1} , $k \in \mathbf{N}$.

PAS 2. Tabela S este definită prin formula:

$$S(x) = \mathbf{A} \cdot F(x) + \mathbf{b}.$$

Observația 8.5.1. Algoritmul AES utilizează funcția $F(x) = x^{-1}$.

8.6. Criptanaliza diferențială

Criptanaliza diferențială a fost introdusă inițial de *E. Biham* și *A. Shamir* la începutul anilor 90. Cifrul pe care s-a aplicat, de către aceștia, a fost standardul american de cifrare a datelor *DES* (Data Encryption Standard). În esență tehnica criptanalizei diferențiale constă într-o formă particulară a tehnicii *atacului cu text clar cunoscut*: se analizează perechi de text clar cunoscut (C_i, C_j) , care sunt într-o anumită relație (de exemplu ponderea *Hamming* a sumei mod 2 este o constantă dată sau un vector dat: $C_i \oplus C_j = D_{ij}$) și efectul asupra textelor cifrate cu aceeași cheie (de exemplu vectorul sumei mod 2 a textelor cifrate: $M_i \oplus M_j$). Scopul urmărit este acela de a recupera (cel puțin probabilist) cheia de cifrare K .

8.7. Criptanaliza liniară

Criptanaliza liniară este o tehnică prin care se urmărește construcția unui sistem de ecuații liniare între biții textului clar, ai textului cifrat și ai cheii. Rezolvarea acestui sistem de ecuații duce la aflarea cheii de cifrare. Sistemul de ecuații ce se construiește poate fi chiar un sistem probabilist, în sensul că o ecuație este verificată cu o anumită probabilitate.

8.8. Alte metode

În studiul cifrurilor bloc se mai pot aplica următoarele metode și tehnici criptografice (vezi *Schneier* [70]):

- Metoda caracteristicilor diferențiale condiționate,
- Criptanaliza cu ajutorul cheilor deplasate,
- Criptanaliza diferențial-lineară,
- Metoda combinată diferențial-lineară,
- Metoda criptanalizei diferențiale de ordin superior,
- Metoda criptanalizei diferențiale de ordin superior a cifrului KN,
- Metoda aproximării multi-lineare,
- Metoda diferențialelor trunchiate,
- Metoda generalizată a criptanalizei lineare,
- Metoda de atac prin interpolare,
- Metoda de atac prin funcții non-surjective,
- Tehnica transformatei Walsh-Hadamard.

8.9. Implementări și rezultate experimentale

8.9.1. Implementarea standardului de cifrare A.E.S.

Cifrul bloc AES (Advanced Encryption Standard) este noul standard de cifrare a datelor care înlocuiește standardul DES. Standardul AES este un caz particular al algoritmului de cifrare RIJNDAEL (proiectat de Joan Daemen și Vincent Rijmen) în sensul că are setate lungimea cheii de cifrare la 128 biți iar dimensiunea blocului de date care se cifrează la 128, 192 sau 256 biți. În figura 8.5 se prezintă ultimii 10 finaliști (algoritmii) care au participat la selecția standardului. Implementarea algoritmului RIJNDAEL s-a realizat în limbajul de programare C.

Cifru	Sursă documentare
CAST-256	Entrust Technologies, Inc. (represented by Carlisle Adams)
CRYPTON	Future Systems, Inc. (represented by Chae Hoon Lim)
DEAL	Richard Outerbridge, Lars Knudsen
DFC	CNRS - Centre National pour la Recherche Scientifique - Ecole Normale Supérieure (represented by Serge Vaudenay)
E2	NTT - Nippon Telegraph and Telephone Corporation
FROG	TecApro Internacional S.A.
HPC	Rich Schroepel
LOKI97	Lawrie Brown, Josef Pieprzyk, Jennifer Seberry
MAGENTA	Deutsche Telekom AG (represented by Dr. Klaus Huber)
MARS	IBM (represented by Nevenko Zunic)
RC6	RSA Laboratories (represented by Matthew Robshaw)
RIJNDAEL	Joan Daemen, Vincent Rijmen
SAFER+	Cylink Corporation (represented by Dr. Lily Chen)
SERPENT	Ross Anderson, Eli Biham, Lars Knudsen
TWOFISH	Bruce Schneier, John Kelsey, Doug Whiting, David

Figura 8.5: Principalii algoritmi implicați în procesul de selectare a standardului A.E.S.

8.9.2. Testarea algoritmului AES

Pentru fixarea ideilor vom nota, în cele ce urmează, prin n dimensiunea blocului de date al cifrului bloc iar prin k lungimea cheii acestuia. Algoritmii candidați pentru standardul de cifrare bloc *AES* au fost supuși testelor statistice (vezi *NIST 800-22* [107]), construcția eșantioanelor (vezi *Soto* [87]) fiind realizată după cum urmează:

-*avalanșa cheii*: se fizează în mod aleatoriu un set de chei, se setează la 0—peste tot textul (text de referință) și se urmărește efectul cifrării textului de referință cu ajutorul setului de chei perturbate într-un singur bit;

-*avalanșa textului de intrare*: se fizează în mod aleatoriu o mulțime de texte, se setează la 0—peste tot cheia (cheia de referință) și se urmărește efectul produs prin cifrarea, cu cheia de referință, a mulțimii textelor perturbate într-un singur bit;

-*corelație intrare/ieșire*: se urmărește detectarea corelațiilor dintre intrare și ieșire prin utilizarea a r chei de cifrare pseudoaleatoare (de referință) și a s texte de intrare pseudoaleatoare (de referință), cifrându-se, în modul ECB, toate cele s texte cu toate cele r chei.

-*corelație în modul de lucru CBC*: se fixează, în mod pseudoaleatoriu un set de chei de cifrare (de referință), se setează la 0—peste tot vectorul de inițializare precum și textul clar (suficient de lung) și se urmărește detecția corelației din concatenarea tuturor textelor cifrate rezultate.

-*text clar cu densitate redusă*: se testează aleatorismul statistic al cifrării tuturor textelor clare de pondere Hamming 0, 1 și 2 cu ajutorul unui set de chei generate pseudoaleator.

-*chei cu densitate redusă*: se testează aleatorismul statistic al cifrării textelor clare generate pseudoaleator cu toate cheile de pondere Hamming 0, 1 și 2.

-*text clar cu densitate mare*: se testează aleatorismul statistic al cifrării tuturor textelor clare de pondere Hamming n , $n - 1$ și $n - 2$ cu ajutorul unui set de chei generate pseudoaleator.

-*chei cu densitate mare*: se testează aleatorismul statistic al cifrării textelor clare generate pseudoaleator cu toate cheile de pondere Hamming n , $n - 1$ și $n - 2$.

8.9.3. Rezultate experimentale

Vom prezenta în cele ce urmează o serie de rezultate experimentale obținute în cazul aplicării testelor de avalanșă strictă, imunitate la corelație, balans, nedegenerare precum și a unor variante de criptanaliză diferențială asupra algoritmului de cifrare bloc RIJNDAEL (s-au folosit pentru testare numai cheile de pondere Hamming 0 și 1 iar ca text de referință textul nul).

1. Rezultatele testului de avalanșă strictă asupra algoritmului RIJNDAEL

Cheia de referință: cheia 0-peste tot
 Textul de referință: 0-peste tot
 Acțiune efectuată: perturbarea cu un bit a cheii de referință.
 Dimensiunea cheii: 128 biți
 Volumul eșantionului: 128 biți
 Riscul de efectuare al testării statistice: 0,05
 Deplasarea inițială (cuvinte ignorate): 0
 Parametrul de codificare al datelor 8 biți
 Sunt afișate procentele de schimbări pe pozițiile slabe ale cheii de bază:
 60,156% poziția 31,
 60,938% poziția 37,
 60,156% poziția 51,
 40,602% poziția 125.
 Numărul de poziții slabe ale cheii 4
 Decizie: algoritmul implementat indeplinește criteriul de avalanșă strictă.
 Statistica testului (procentul de respingeri) este: $-0,973329$
 Indicații asupra atacului optimal cu chei de pondere 1: probabilitățile extreme sunt 40,625 poziția 125 respectiv 60,938 poziția 37.

2. Rezultatele testului de imunitate la corelație asupra algoritmului RIJNDAEL

Cheile de referință sunt cheile de pondere 0 și 1
 Textul de referință: 0-peste tot
 Acțiune efectuată: perturbarea cu un bit a cheii de referință.
 Dimensiunea cheii: 128 biți
 Deplasarea inițială (cuvinte ignorate) 0
 Parametrul de codificare al datelor 8 biți
 Volumul eșantionului: 128 biți
 Riscul de efectuare al testării statistice 0,05
 Sunt afișate procentele de schimbări de la cheile slabe:
 Decizie: cheia de pondere 0 este imună la corelație.
 Rezultatele pentru cheile de pondere 1 sunt următoarele:
 60,156% poziția 43
 59,375% poziția 74
 Numărul de corelații ale cheilor de pondere 1 este 2
 Decizie: Algoritmul implementat indeplinește criteriul de imunitate la corelație pentru cheile de pondere Hamming 1.
 Statistica testului (procentul de respingeri) este: $-1,784436$
 Indicații asupra atacului optimal cu chei de pondere 1: Probabilitățile extreme sunt 41,406% poziția 82 respectiv 60,156% poziția 43.

3. Rezultatele testului de balans asupra algoritmului RIJNDAEL

Cheile de referință sunt cheile de pondere 1

Textul de referință: 0-peste tot

Dimensiunea spațiului de intrare este de 128 biți

Dimensiunea spațiului de ieșire este de 8 biți

Depalarea inițială (ture in gol) 0

Parametrul de codificare al datelor: 8 biți

Riscul de efectuare al testarii statistice 0,05

Probabilitatea de balans este 0,5

Sunt afisate numărul de intrări care realizează fiecare byte:

Se testează funcția 0 : $N[0] = 62$, $N[1] = 66$

Statistica testului este 0,5156

Se testează funcția 1 : $N[0] = 60$, $N[1] = 68$

Statistica testului este 0,5312

Se testează funcția 2 : $N[0] = 64$, $N[1] = 64$

Statistica testului este 0,50

Se testează funcția 3 : $N[0] = 61$, $N[1] = 67$

Statistica testului este 0,5234

Se testează funcția 4 : $N[0] = 64$, $N[1] = 64$

Statistica testului este 0,5

Se testează funcția 5 : $N[0] = 59$, $N[1] = 69$

Statistica testului este 0,539

Se testează funcția 6 : $N[0] = 62$, $N[1] = 66$

Statistica testului este 0,5156

Se testează funcția 7 : $N[0] = 54$, $N[1] = 74$

Statistica testului este 0,5781

Decizie: algoritmul implementat indeplinește criteriul de balans.

4. Rezultatele testului de nedegenerare asupra algoritmului RIJN-DAEL

Cheia de referință: cheia 0—peste tot

Textul de referință: 0-peste tot

Dimensiunea cheii: 128 biți

Volumul eșantionului: 128 biți

Deplasare inițială (cuvinte ignorate): 0

Parametrul de codificare al datelor 8 biți

Sunt afișate punctele de degenerare ale algoritmului: nu este cazul

Numărul de puncte degenerate ale algoritmului 0

Decizie: algoritmul implementat îndeplinește criteriul de nedegenerare.

5. Rezultatele testului de criptanaliză diferențială asupra algoritmului RIJNDAEL

Dimensiunea cheii 128 biți

Dimensiunea bloc de date inițial: 128 biți

Bloc de date inițial 16 de 0– peste tot

Riscul de efectuare al testării statistice: 0,05

Probabilitatea de corelare: 0,5;

Iterații efectuate: 1000;

Sunt afișate cele mai semnificative cheii de bază (indice de coincidență extram global) pentru cheile de pondere Hamming 0 și 1:

Cheia nulă are patternul:

Indice de coincidență maxim între două cifrări succesive este 0.640625 obținut la iterația 914;

Indice de coincidență minim între două cifrări succesive este 0.343750 obținut la iterația 311;

Indice de coincidență maxim cu textul inițial este 0.640625 obținut la iterația 890;

Indice de coincidență minim cu textul inițial este 0.335938 obținut la iterația 65;

Cheia cea mai slabă (corelație directă) este 17 cu indicele maxim de coincidență între două cifrări succesive 0.695313 obținut la iterația 667;

Cheia cea mai slabă (corelație indirectă) este 13 cu indice minim de coincidență între două cifrări succesive 0.328125 obținut la iterația 419;

Cheia cea mai slabă (autocorelație directă) este 88 cu indice maxim de coincidență cu textul inițial 0.679688 obținut la iterația 281;

Cheia cea mai slabă (autocorelație indirectă) este 43 cu indice minim de coincidență cu textul inițial 0.320313 obținut la iterația 168.

8.9.4. Interpretarea rezultatelor

După cum se observă algoritmul de cifrare RIJNDAEL îndeplinește cerințele formulate în cadrul capitolului 3. Acest lucru nu înseamnă faptul că algoritmul este absolut sigur: *în criptografie nici o regulă nu este absolută* (Étienne Bazeris, 1901). O descriere completă a interpretării rezultatelor testelor statistico-informaționale poate fi găsită în *Simion* [84].

8.10. Concluzii

Cifrurile bloc pot fi programate să funcționeze ca un cifru flux și reciproc (a se vedea modul de lucru OFB). Cea mai bună definiție, care pune în evidență diferențele dintre aceste două structuri de cifrare, este următoarea:

Definiția 8.10.1. Cifrurile bloc operează asupra datelor cu transformări (fixe) asupra blocurilor de date clare; cifrurile flux operează asupra datelor cu transformări (ce variază în timp) asupra cuvintelor (biți, octeți, etc.) din textul clar.

În cazul aplicațiilor practice, cifrurile bloc par mai generale (pot funcționa în unul dintre cele patru moduri principale) iar cifrurile flux sunt mai simplu de analizat din punct de vedere matematic. O altă diferență este aceea că cifrurile flux cifrează sau descifrează un singur cuvânt de date la un tact, deci nu sunt optime pentru implementările software. Cifruile bloc sunt mai simplu de implementat din punct de vedere soft deoarece acestea operează cu blocuri de cuvinte de procesor deci sunt mai rapide. Pe de altă parte cifrurile flux sunt mai ușor de implementat în aplicațiile software.

8.11. Aplicații

Deoarece nu există o formulă matematică universală care să poată fi aplicată în operația de criptanaliză, am propus ca exerciții la acest capitol modificări ale unor algoritmi de cifruri bloc consacrate. Sunt date o serie de indicații precedate de o scurtă descriere a algoritmilor propriu-ziși.

Exercițiul 8.11.1. Studiați următoarele simplificări ale algoritmului RC5:

-RC5 cu 8 iterații dar fără rotații;

-RC5 cu 8 iterații iar numărul de rotații egal cu numărul de iterații.

Răspuns. În cele ce urmează facem o scurtă descriere a cifrului RC5 cu r iterații. Acesta are lungimea blocului de date variabilă dar vom considera în cele ce urmează că aceasta a fost setată la 64 biți. Operația de cifrare folosește $2r+2$ chei dependente de cuvintele pe 32 biți $S_0, S_1, S_2, \dots, S_{2r+2}$ unde r este numărul de iterații. Pentru cifrare blocul de date se împarte în două părți de 32 biți notate cu L respectiv R (RC5 face apel la codificarea *little-endian* pentru împachetarea octeților în cuvinte: primul octet se transformă în cele mai puțin semnificative poziții ale lui L , etc.). Apoi avem:

$$\begin{cases} L = L + S_0, \\ R = R + S_1. \end{cases}$$

Pentru $i = 1, \dots, r$ se execută:

$$\begin{cases} L = ((L \oplus R) \ll R) + S_{2i}, \\ R = ((R \oplus L) \ll L) + S_{2i+1}. \end{cases}$$

Ieșirea constă în registrele L și R . Simbolul \oplus are semnificația sumei mod 2, simbolul \ll semnifică rotire circulară și în fine simbolul $+$ are semnificația sumei mod 2^{32} . Operația de descifrare este similară (interven operatorii \oplus, \gg și $-$). Modul de construcție al secvenței S (care derivă din cheie) nu este esențial în cadrul acestui exercițiu.

Dacă setăm numărul de iterații $r = 8$ și nu facem nici un fel de rotații atunci pentru $i = 1, \dots, 8$ se execută:

$$\begin{cases} L = (L \oplus R) + S_{2i}, \\ R = (R \oplus L) + S_{2i+1}. \end{cases}$$

Algoritmul astfel setat nu îndeplinește criteriul de avalanșă strictă (schimbarea unui bit în blocul de text clar produce, în medie, schimbări de 50% la ieșire). Schema de mai sus permite atacul cu ajutorul tehnicii criptanalizei liniare pentru aflarea lui S , deci a cheii efective.

Dacă setăm numărul de iterații $r = 8$ și numărul de rotații egal cu r atunci pentru $i = 1, \dots, 8$ se execută:

$$\begin{cases} L = ((L \oplus R) \ll 8) + S_{2i}, \\ R = ((R \oplus L) \ll 8) + S_{2i+1}. \end{cases}$$

Algoritmul astfel setat nu îndeplinește criteriul de avalanșă strictă. Schema de mai sus permite atacul cu ajutorul tehnicii criptanalizei diferențial/liniare pentru aflarea lui S .

Exercițiul 8.11.2. Studiați următoarele simplificări ale algoritmului DES:

- DES cu 12 iterații dar fără aplicațiile S ;
- DES cu 4 iterații;
- DES cu 6 iterații.

Răspuns. Cifrul bloc DES (proiectat în 1977) este sub controlul unei chei efective de 56 biți (cheia de bază este de 64 biți, 8 biți fiind pentru detecția erorilor) iar mărimea blocului de date este de 64 biți. Textul clar este permutat iar apoi este împărțit în două blocuri L și R de lungime 32 biți. Se execută apoi iterativ operațiile (pentru $i = 1, \dots, \text{numărul de iterații}$):

$$\begin{cases} L_i = R_i, \\ R_i = L_i \oplus f(R_{i-1}, K_i). \end{cases}$$

În final textul este supus permutării inverse. Ne concentrăm asupra descrierii funcției $f : \mathbf{Z}_2^{32} \times \mathbf{Z}_2^{48} \rightarrow \mathbf{Z}_2^{32}$. Inițial blocul R (32 biți) este extins cu ajutorul

funcției E la un bloc pe 48 biți care este sumat mod 2 cu cheia K (extinsă la 48 biți cu ajutorul algoritmului de producere a subcheilor). Opt aplicații $S : \mathbf{Z}_2^6 \rightarrow \mathbf{Z}_2^4$ produc o ieșire pe 32 biți care este permutată pentru a produce ieșirea finală dintr-o iterație. Dacă aplicațiile S sunt fixe (se selectează 4 biți din 6 în mod fix) atunci se poate aplica tehnica criptanalizei diferențiale (biții de la ieșire sunt biții de la intrare (sumați mod 2 cu cheia K) dar într-o altă ordine).

Algoritmul DES cu 4 cât și cu 6 iterații poate fi spart cu ajutorul tehnicii atacului cu text clar cunoscut.

Exercițiul 8.11.3. Studiați regula B a algoritmului *Skipjack* cu 8 iterații.

Exercițiul 8.11.4. Ce defect are un algoritm de cifrare care este închis (un algoritm de cifrare se numește *închis* dacă pentru orice chei k_1 și k_2 există o cheie k_3 astfel încât pentru orice text clar M avem $E_{k_1}E_{k_2}(M) = E_{k_3}(M)$)?

Răspuns. Ca metodă de atac generică se poate opta pentru cifrarea repetitivă.

Exercițiul 8.11.5. Aplicați tehnica criptanalizei diferențiale și criptanalizei liniare asupra algoritmului FEAL.

Exercițiul 8.11.6. Studiați tehnica criptanalizei diferențiale în cazul algoritmului DES cu 16 iterații.

Exercițiul 8.11.7. Aplicați tehnica criptanalizei liniare în cazul algoritmului DES cu 16 iterații.

Exercițiul 8.11.8. Având la dispoziție un cifru bloc $E_k(\cdot)$ proiectați un cifru flux și viceversa.

Exercițiul 8.11.9. Scrieți funcția analitică a celor opt funcții de substituție S ale cifrului DES.

Exercițiul 8.11.10. Fie $E_M(\cdot)$ și $D_K(\cdot)$ funcțiile de cifrare respectiv descifrare ale unui cifru. Care este valoarea lui $D_K(E_K(M))$?

Notă. Descrierea algoritmilor RC5, DES, Skipjack și FEAL poate fi găsită în *Schneier* [69] sau *Menezes* [50].

Exercițiul 8.11.11. Implementați modalități de testare a cifrurilor bloc.

Exercițiul 8.11.12. Implementați modalități de generare a tabelelor de substituție.

Exercițiul 8.11.13. Fie $E(\cdot, \cdot)$ o funcție de cifrare pe m biți de cheie și n biți de date. Care este valoarea maximă a lui m astfel încât cheia efectivă a cifrului să fie m ?