SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Ivan Nađ

DES I AES

Diplomski rad

Voditelj rada:
doc.dr.sc. Zrinka Franušić

Zagreb, srpanj, 2014.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Uvod

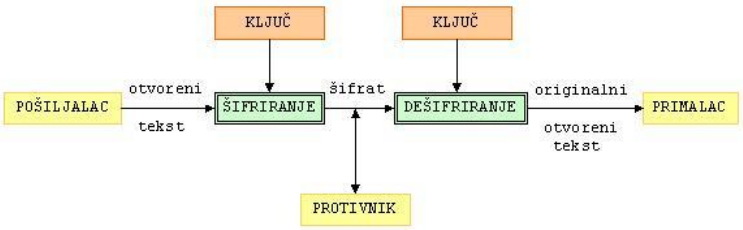
Kriptografija je znanstvena disciplina koja se bavi proučavanjem metoda za slanje poruka u takvom obliku da je sadržaj poruke namijenjen samo onima koji ih moraju, odnosno mogu pročitati i, naravno, samom pošiljaocu poruke.

Elementi kriptografije bili su prisutni već i kod starih Grka. Spartanci su upotrebljavali napravu za šifriranje zvanu *skital*, što je zapravo bio drveni štap oko kojeg se namotavala vrpca od pergamenta (na nju se okomito pisala poruka). Nakon upisivanja poruke, vrpca bi se odmotala i izgledala nesuvislo naivnom promatraču. Sadržaj poruke mogao je pročitati samo onaj koji je imao štap jednake debljine kao onaj na kojem je poruka napisana.

Slika 0.1: Skital

Osnovni zadatak kriptografije je omogućiti komunikaciju preko nesigurnog komunikacijskog kanala tako da netko tko ima mogućnost nadzora tog komunikacijskog kanala ne razumije sadržaj poruke.

Poruku koji pošiljatelj želi poslati primatelju zvat ćemo *otvoreni tekst* (eng. plaintext). Pošiljatelj transformira otvoreni tekst koristeći neki unaprijed dogovoreni *ključ* (eng. key). Taj postupak naziva se *šifriranje*, a rezultat šifriranja *šifrat* (eng. ciphertext). Obrnuti postupak transformiranja šifrata nazad u isti otvoreni tekst nazivamo *dešifriranje*.



Slika 0.2: Shema klasične kriptografije

Kroz povijest se otvoreni tekst šifrirao na raznorazne načine, s time da sve te načine šifriranja najčešće klasificiramo s obzirom na sljedeća tri kriterija:

1. **Tip operacija koje se koriste pri šifriranju.** Tako imamo podjelu na *supstitucijske šifre*, u kojima se svaki element otvorenog teksta zamjenjuje nekim drugim elementom, te *transpozicijske šifre* u kojima se elementi otvorenog teksta premještaju (permutiraju). Primjerice, ako riječ MATKA šifriramo u SYZWY, načinili smo supstituciju, a ako je šifriramo u KAMTA, načinili smo transpoziciju.

2. **Način obrade otvorenog teksta.** Ovdje imamo podjelu na *blokovne šifre* koje obrađuju jedan po jedan blok otvorenog teksta, koristeći jedan te isti ključ, te *protočne šifre* kod kojih se elementi otvorenog teksta obrađuju jedan po jedan, koristeći paralaleno generirani niz ključeva.

3. **Tajnost ključa.** Ovdje je osnovna podjela na sustave s *tajnim* i sustave s *javnim* ključem. Kod sustava s tajnim ključem, ključ za dešifriranje se može izračunati poznavanjem ključa za šifriranje i obratno. Sigurnost ovakvih sustava leži u tajnosti ključa. Kod sustava s javnim ključem, ključ za dešifriranje se ne može izračunati iz ključa za šifriranje. Sigurnost leži u tome da bilo koja osoba može šifrirati poruku poznatim ključem, ali samo osoba s odgovarajućim ključem za dešifriranje može saznati sadržaj te poruke.

S obzirom da su se kroz povijest vrlo važne informacije često šifrirale, npr. političke i vojne, uvijek su postojali ljudi kojima je sadržaj šifrirane poruke bio od interesa pa su šifrati često ”napadani” raznim metodama, od kojih razlikujemo 4 osnovne:

1. **Samo šifrat.** Protivnik posjeduje samo šifrat od nekoliko poruka šifriranih istim algoritmom i njegov cilj je otkriti otvoreni tekst od što više poruka ili ključ.

2. **Poznati otvoreni tekst** Protivnik posjeduje šifrat neke poruke i njemu odgovarajući otvoreni tekst. Njegov je cilj otkriti ključ ili algoritam dešifriranja.

3. **Odabrani otvoreni tekst.** Protivnik ima mogućnost odabira teksta koji će biti šifriran i može dobiti njegov šifrat. Ovakve metode su se događale kroz povijest, primjerice kada je neka vojna sila zaplijenila napravu za šifriranju suprotne strane. Cilj protivnika je otkriti ključ.

4. **Odabrani šifrat.** Protivnik ima pristup alatu za šifriranje pa može odabrati šifrat i dobiti njemu odgovarajući otvoreni tekst. Ovakve metode se često koriste u sustavima s javnim ključem. Cilj protivnika je otkriti ključ.

Kroz godine su se razvile mnoge metode šifriranja podataka, a ujedinjenjem vojnog i poslovnog svijeta je došlo do potrebe razvijanja univerzalne metode šifriranja podataka radi suradnje i komunikacije sve većeg broja korisnika.

Tema ovog rada je opis dvaju simeričnih kriptosustava koji predtavljaju standard za šifriranje elektronskih podataka DES (Data Encryption Standard) i AES (Advanced Encryption Standard). DES je bio standard za šifriranje od 1977. godine do kraja 20. stoljeća. Naslijedio ga je i danas aktualan kriptosustav AES.

Rad je podijeljen u dva poglavlja - jedno posvećeno DES-u, a jedno AES-u. Detaljno je opisan algoritam za šifriranje oba kriptosustava. DES se sastoji od niza transpozicija i supstitucija. Njegov najzanimljivi dio leži u korištenju tzv. S-kutija, matrica tipa 4×16 za koje ni danas sa sigurnošću ne znamo kako su nastale. Današnji standard za šifriranje AES je specifičan po tome što koristi aritmetiku konačnih polja, konkretno polja $GF(2^8)$, a čiji se elementi praktično mogu reprezentirati kao 1 bajt (8 bitova). Iz tog razloga se u radu navode najvažnije činjenice iz područja algebarskih struktura. Ukratko su opisane i mogućnosti napada na ove sustave, to jest njihova kriptoanaliza.

Poglavlje 1

DES

1.1 Povijest DES-a

Sredinom 20. stoljeća, bržim razvojem tržišta finacijskih usluga, kriptografija je postala potrebna i zanimljiva povećem broju korisnika. Dotad se najviše koristila u vojne i diplomatske svrhe, bez nekog standarda (npr. Playfairova i Hillova šifra) koji su sve više tražile sve brojnije multinacionalne kompanije koje su međusobno surađivale.

Tu potrebu je uočio i američki NBS (*National Bureau of Standards*), danas imena NIST (*National Institute of Standards and Technology*), te je 1972. započeo s iniciranjem programa za zaštitu podataka, da bi već sljedeće 1973. godine, 15. svibnja, raspisao natječaj za kriptosustav koji bi postao svjetski standard, a morao bi zadovoljiti sljedeće uvjete:

- visoki stupanj sigurnosti,
- potpun opis i lako razumijevanje algoritma,
- sigurnost algoritma osigurava ključ, a ne tajnost algoritma,
- dostupnost velikom broju korisnika,
- široka primjenjivost,
- ekonomičnost ugradnje u elektroničke uređaje,
- efikasnost,
- provjerljivost,
- mogućnost izvoza.

Nijedan od kriptosustava pristiglih na natječaj nije zadovoljavao sve gore navedene uvjete pa je raspisan još jedan natječaj, 27. kolovoza 1974., na kojem je "najozbiljnija" prijava bila ona koju je poslao IBM-ov tim i čiji algoritam se zasniva na tzv. *Feistelovoj šifri*, ideji koju je uveo sam voditelj ovog tima Horst Feistel 1973. godine. Glavna ideja je alternirana uporaba supstitucija i transpozicija kroz više rundi.

U razvoj algoritma se uključila i američka sigurnosna agencija, NSA, čiji je najveći utjecaj bio na skraćivanje ključa i na misteriozne S-kutije (o njima će još biti riječi kasnije). Predloženi kriptosustav objavljen je 17. ožujka 1975. godine u časopisu *Federal Register* i tražili su se javni komentari, a održane su i dvije radionice o predloženom standardu. Glavne zamjerke velikog dijela kriptografa su bile upravo gore navedene preinake na koje je utjecala NSA. Jedna od zamjerki je bila i da je algoritam vjerojatno "oslabljen" namjerno i to toliko da ga NSA, ali nitko drugi, svaki put može lako i brzo pročitati i presresti svaku poruku.

Na kraju rasprave, odbor američkog senata zadužen za program je odbacio ovakve komentare i zaključio da algoritam nema nikakve statističke ni matematičke slabosti u tom trenutku, s obzirom na tadašnji stupanj razvoja računala, te da NSA ni na koji način nije direktno, već samo indirektno, utjecala na razvoj algoritma. Do dana današnjeg nije u potpunosti razjašnjen utjecaj NSA, a razjašnjenju cijele situacije nisu pomogle ni međusobno proturječne izjave upravo članova IBM-ova tima koji je radio na razvoju algoritma.

Konačno, u studenom 1976. predloženi kriptosustav je prihvaćen kao standard i nadjenuto mu je ime ***Data Encryption Standard*** ili kraće **DES**, a 15. siječnja 1977. godine je i službeno objavljen.

1.2 Opis algoritma

DES svoj posao šifriranja započinje grupiranjem otvorenog teksta u blokove veličine 64 bita, a nastavlja koristeći ključ K veličine 56 bitova. Ključ je zapravo veličine 64 bita, ali se 56 bitova direktno koristi u algoritmu, a preostalih 8 se ne koristi direktno, nego služi za provjeru pariteta o čemu će biti riječi kasnije. Šifrat se također ispisuje u blokovima veličine 64 bita.

Glavne etape

Algoritam ima 3 etape :

1. etapa: Dani otvoreni tekst, nazovimo ga x, najprije permutiramo inicijalnom permutacijom IP, i dobivamo x_0. Pišemo

$$IP(x) = x_0.$$

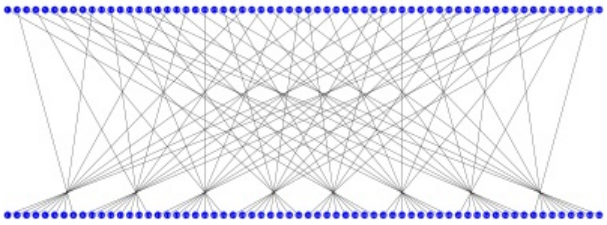
Također, rezultat dobiven permutiranjem dijelimo na dva dijela, lijevi dio L_0 i desni dio R_0, od kojih svaki od njih sadrži 32 bita. Dakle, lijevi dio L_0 se sastoji od prva 32 bita, a desni dio R_0 od zadnja 32 bita. Naravno, onda x_0 možemo zapisati kao

$$x_0 = L_0 R_0.$$

Zbog preglednosti, permutaciju IP zapisujemo kao matricu, odnosno tablicu tipa 8×8.

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Fiksna inicijalna permutacija IP



Slika 1.1: Shema permutacije IP

2. etapa: U ovoj etapi lijevi i desni dio dobiveni u prethodnoj etapi prolaze kroz 16 iteracija na sljedeći način

$$L_i = R_{i-1},\ R_i = L_{i-1} \oplus f(R_{i-1}, K_i),$$

$i = 1, 2, 3, ..., 16$. Oznaka $\oplus$ predstavlja operaciju *ekskluzivno ili*, odnosno XILI (XOR). Operaciju $\oplus$ možemo shvatiti i kao zbrajanje/oduzimanje u brojevnom sustavu s bazom 2, odnosno kao zbrajanje u polju $\mathbb{Z}_2$. Možemo ju shvatiti i kao operaciju koja daje rezultat *istina* ako se oba unosa razlikuju i *laž* ako su unosi isti. To jest, $0 + 1 = 1 + 0 = 1$ i $0 + 0 = 1 + 1 = 0$.

K_1, K_2,..., K_{16} je niz tzv. *međuključeva* koji se dobivaju permutiranjem nekih bitova iz K i duljine su 48 bitova. Ključ za svaku pojedinu iteraciju se generira u samoj iteraciji.

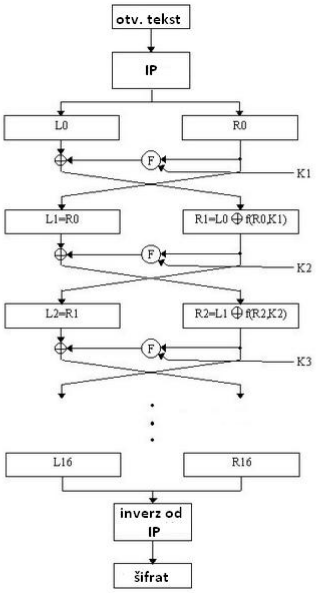
Funkcija f se još naziva i *Feistelova funkcija* i predstavlja srž ovog algoritma za šifriranje. Njezino računanje će biti opisano nešto kasnije, nakon kratkog opisa treće etape algoritma.

3. etapa: U ovoj posljednjoj etapi se primjenjuje inverzna permutacija IP^{-1} na $R_{16}L_{16}$ čime dobivamo šifrat y. Odnosno,

$$y = IP^{-1}(R_{16}L_{16}).$$

Uočimo da smo lijevi i desni dio, L_{16} i R_{16}, koji smo dobili kao rezultat 2. etape u ovoj etapi zapisali u obratnom poretku, prvo smo napisali desni dio a zatim lijevi.

Cijeli gore opisani algoritam je zorno prikazan sljedećom shemom:



Slika 1.2: Shema *DES* algoritma

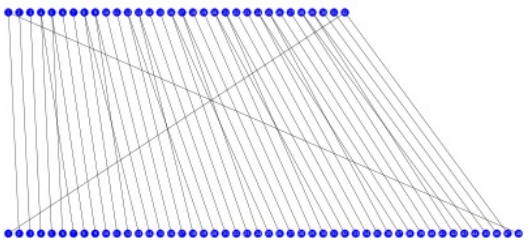
Feistelova funkcija

Sada ćemo opisati djelovanje Feistelove funkcije f. To je funkcija dvije varijable, odnosno dva argumenta. Prvi argument u i-toj iteraciji je desni dio R_{i-1} koji ćemo za sada označiti s A i koji je duljine 32 bita. Drugi argument je međuključ K_i duljine 48 bita koji ćemo označiti s J. Funkcija $f(A, J)$ se računa u 4 koraka.

1. korak: Budući da ulazni argumenti A i R nisu iste duljine, prvi argument A proširujemo s niza bitova duljine 32 na niz bitova duljine 48. To se čini pomoću fiksne funkcije proširenja E. Tako dobivamo niz $E(A)$ duljine 48 bitova. Fiksna funkcija proširenja E dana je 8×6 tablicom:

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Fiksna funkcija proširenja E



Slika 1.3: Shema funkcije proširenja E

2. korak: Sada računamo $E(A) \oplus J$ i rezultat pišemo kao spoj (konkatenaciju) osam 6-bitnih nizova B_i

$$B = B_1B_2B_3B_4B_5B_6B_7B_8.$$

3. korak: U ovom koraku koristi se 8 ranije spomenutih tzv. *supstitucijskih kutija* ili *S-kutija*: $S_1, S_2, ..., S_8$. Svaka S-kutija je zapravo fiksna 4×16 matrica čiji su elementi brojevi između 0 i 15. Na svaki od nizova B_j iz prethodnog koraka djelovat će njegova S-kutija S_j. Napomenimo, retke matrica S_j numeriramo od 0 do 3 (umjesto 1 do 4), a stupce od 0 do 15 (umjesto 1 do 16).

Dani blok bitova

$$B_j = b_1b_2b_3b_4b_5b_6,$$

koji je duljine 6 bitova, "dolazi" na kutiju S_j nakon čega dobivamo niz bitova $S_j(B_j)$ duljine 4. Sada opišimo kako se efektivno računa $S_j(B_j)$. Prvi i zadnji bit od B_j

određuju binarni zapis retka $r = (b_1 b_6)_2 \in \{0, 1, 2, 3\}$ matrice S_j. Srednja četiri bita od B_j određuju binarni zapis stupca $c = (b_2 b_3 b_4 b_5)_2 \in \{0, 1, \ldots, 15\}$ matrice S_j. Sada je $S_j(B_j)$ po definiciji jednako $S_j(r, c)$ zapisano kao binarni broj duljine 4.

Primjerice, gledamo $S_1(B)$ za neki unos $B_j = 011011$. Redak određuju prvi i zadnji bit koji su ovdje 01 (binarni zapis), odnosno broj retka je 1 (dekadski zapis). Stupac određuju srednja 4 bita koji su ovdje 1101 (binarni zapis), odnosno broj stupca je 13 (dekadski zapis). Na presjeku 1. retka i 13. stupca u matrici S_1 nalazi se broj 5 (dekadski zapis), odnosno 0101 (binarni zapis). Na taj način je

$$S_1(011011) = 0101.$$

Dakle, kao rezultat ovog koraka dobivamo 8 nizova, duljine 4 bita, koje označavamo kao

$$C_j = S_j(B_j), \quad j = 1, 2, \ldots, 8.$$

S_1

14	4	3	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

S-kutije *DES* algoritma

4. korak: Iz prethodnog koraka dobivamo 8 nizova C_j, $j = 1, \ldots, 8$ duljine 4 bita. Njihovom konkatenacijom imamo niz bitova duljine 48:

$$C = C_1C_2C_3C_4C_5C_6C_7C_8.$$

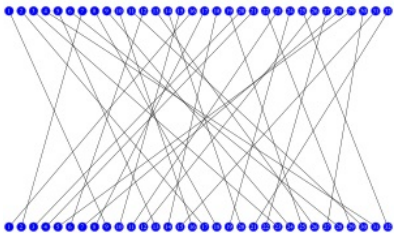
Na C još djeluje fiksna završna permutacija P koja rezultira nizom $P(C)$. Konačno, to je upravo vrijednost funkcije f, odnosno

$$f(A, J) = P(C).$$

Fiksna završna permutacija je dana sljedećom tablicom (matricom) 8×4.

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Fiksna završna permutacija P



Slika 1.4: Shema permutacije P

Međuključevi

Na kraju nam je preostalo opisati računanje tablice međuključeva K_1, K_2, ...,K_{16} koji su se koristili u drugoj etapi DES-a. Svi oni dobivaju se iz ključa K. Kao što smo već prije spomenuli, ključ K se sastoji od 64 bita od kojih 56 bitova predstavlja ključ, a preostalih 8 služe za testiranje pariteta. Tih 8 paritetnih bitova nalaze se na pozicijama 8, 16, ..., 64 i definirani su tako da svaki bajt (8 bitova) sadrži neparan broj jedinica. Oni se zanemaruju kod računanja tablice ključeva. Postupak računanja tablice međuključeva je opisan u nastavku.

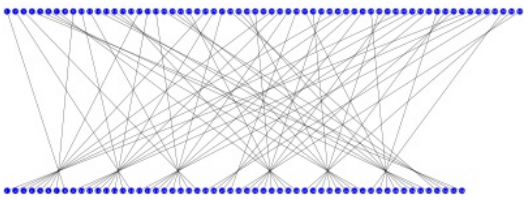
Za dani početni 64-bitni ključ K zanemarimo paritetne bitove (gore smo spomenuli na kojim mjestima se oni nalaze), a preostale permutiramo fiksnom permutacijom $PC1$. Stoga dobivamo

$$PC1(K) = C_0D_0,$$

gdje C_0 sadrži prvih 28, a D_0 drugih 28 bitova od $PC1(K)$.

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Fiksna permutacija $PC1$



Slika 1.5: Shema permutacije $PC1$

Sada kada smo definirali C_0 i D_0, sljedeće C_i-ove i D_i-ove, za $i = 1, 2, ..., 16$, računamo tako da svaki prethodnik, C_{i-1} i D_{i-1}, ciklički pomaknemo ulijevo za jednu ili dvije pozicije, ovisno o tome koliki je i. Broj pomaka ulijevo je dan sljedećom tablicom:

i	Broj pomaka ulijevo
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Slika 1.6: Tablica pomaka

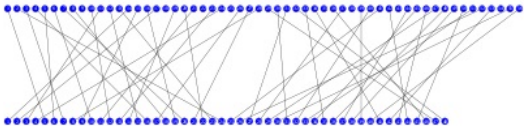
Međuključevi su onda dani s

$$K_i = PC2(C_iD_i),$$

gdje je $PC2$ fiksna *kompresijska* permutacija koja ulazni niz od 56 bitova permutira i smanjuje na niz duljine 48 bitova. Svaki od međuključeva može se računati u odgovarajućoj iteraciji druge etape šifriranja.

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	28
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Fiksna kompresijska permutacija $PC2$



Slika 1.7: Shema kompresijske permutacije $PC2$

Dešifriranje

Vrlo je korisno što za postupak dešifriranja DES koristi isti algoritam kao i za šifiranje s jednom razlikom da se u drugoj etapi koristi niz međuključeva u obrnutom poretku: K_{16}, K_{15}, ...,K_1. Dakle, dešifriranjem nekog šifrata y bi opet trebali dobiti otvoreni tekst x od kojeg smo i krenuli.

Šifrat y smo dobili primjenom inverzne permutacije IP^{-1} na $R_{16}L_{16}$. Stoga, primjenom inicijalne permutacije IP na y dobivamo y_0, odnosno

$$y_0 = IP(y) = IP(IP^{-1}(R_{16}L_{16})) = R_{16}L_{16}.$$

Nadalje, nakon jedne runde dešifriranja dobivamo $R_{15}L_{15}$, nakon dvije $R_{14}L_{14}$, itd. sve do zadnje runde nakon koje dobivamo R_0L_0. Zamjenom poretka dobivamo L_0R_0, što smo zapravo u prvoj etapi prethodno zapisanog algoritma zapisali kao x_0. Sjetimo se da smo x_0 dobili primjenom fiksne inicijalne permutacije IP na x, odnosno $x_0 = IP(x)$. Primjenom inverzne permutacije IP^{-1} dobivamo traženi otvoreni tekst x, odnosno $x = IP^{-1}(IP(x)) = x$ što smo i željeli dokazati.

1.3 Kriptoanaliza DES-a

Uvođenje DES-a u ovakvom obliku su od početka pratile kritike. Izumitelji kriptografije javnog ključa, Whitfield Diffie i Martin Hellman, tvrdili su da bi računalo od 20 milijuna dolara i milijun posebno dizajniranih VLSI čipova, od kojih je svaki sposoban testirati jedan ključ po mikrosekundi, moglo za otprilike 1 dan razbiti poruku šifriranu DES-om. David Chaum i Jan-Hendrik Evertse su uspješno napali verziju DES-a od 4 runde i to 2^{19} brže od običnog običnog testiranja ključeva. Ipak, njihov napad nije mogao proći sljedećih sedam rundi.

Nijedna od ovih metoda, odnosno napada, nije predstavljala ozbiljnu prijetnu za sustav sve dok dva Izraelca i Japanac nisu u DES-u pronašli anomaliju koja je vodila do prvih metoda koje su teoretski bile osjetno bolje nego testiranje ključeva redom. Eli Biham i Adi Shamir su 1990. otkrili metodu imena *diferencijalna kriptoanaliza* pomoću koje se može razbiti *DES* uz poznavanje šifrata za 2^{47} otvorenih tekstova što je ipak značajno manje od 2^{56} što je broj ključeva koje je potrebno testirati uz jedan poznati par otvoreni tekst-šifrat. Prema kasnijim tvrdnjama jednog od tvoraca, već su i sami tvorci DES-a znali za ovu metodu.

Japanski kriptolog Mitsuru Matsui je, nekoliko godina kasnije, otkrio metodu *linearne kriptoanalize* kojom je opisao napad "poznati otvoreni tekst" i koji za razbijanje DES-a treba u prosjeku 2^{43} otvorenih tekstova.

Ironično, ni diferencijalna ni linearna kriptoanaliza nisu dovele do pada DES-a. Brži i jeftiniji čipovi su to napravili. U srpnju 1998. godine, *Electronic Frontier Foundation* je

napravio tzv. "DES Cracker". Cijena mu je bila 250000 dolara, a poruku šifriranu *DES*-om razbio je za 56 sati. Ipak, metode Bilhama, Shamira i Matsuija su izuzetno važne jer su kasnije značajno pomogle u izradi novog standarda AES-a, o kojemu će kasnije biti riječi.

Diferencijalna kriptoanaliza

Kako se DES sve više istraživao, to su sve više na površinu počele isplivavati "tajanstvene" stvari u vezi S-kutija. Iako imaju uravnotežen *izlaz*(svaki se pojavljuje 4 puta, po jednom u svakom retku), postoje određene neuravnoteženosti. Naime, *izlaz* razlika *ulaz* ima nejednaku distribuciju. Upravo tu činjenicu su iskoristili Biham i Shamir.

Osnovna ideja diferencijalne kriptonalize jest usporedba XOR-a dvaju otvorenih tekstova sa XOR-om dvaju odgovarajućih šifrata. Dakle, možemo reći da je to u biti napad odabranim otvorenim tekstom koji vrši analizu između dva otvorena teksta šifrirana istim ključem. Te se razlike - diferencija koriste da bi se odredio najvjerojatniji ključ. XOR vrijednost dvaju otvorenih tekstova je zapravo njihova razlika. Zaista, već smo ustanovili da je oduzimanje u $\mathbb{Z}_2$ isto što i zbrajanje, odnosno isto što i operacija XOR.

Sve se temelji na pretpostavci da postoji veza između ulaznih i izlaznih XOR-ova za specifičnu S-kutiju. Konkretno, mnogi ulazni XOR-ovi na izlazu iz S-kutije daju isti izlazni XOR. Kreiraju se 64 ulazna XOR-a koji su nakon prolaska kroz S-kutiju distribuirani na 16 mogućih vrijednosti izlaznih XOR-ova. Budući da je ustanovljeno da ta distribucija nije uniformna, shvatilo se da bi to moglo predstavljati slabu točku samog sustava, odnosno dobru osnovu za kripto-napad.

Biham i Shamir su, na temelju analize razlika, pronašli algoritam koji pronalazi 48 bitova ključa uz poznavanje šifrata za 2^{47} otvorenih tekstova. Ispada da bi u tom slučaju bilo potrebno napraviti oko 2^{55} DES operacija što je vrlo mala ušteda s obzirom na broj napada "grubom silom" uz jedan poznati par otvoreni tekst- šifrat i koji iznosi 2^{56}. Zbog svega ovog, pretpostavlja se da je samim tvorcima DES-a ovaj napad bio u nekom obliku poznat. I to je upravo razlog zbog kojeg su se odlučili da je broj rundi u DES-u upravo 16. Naime, ovaj bi napad bio puno učinkovitiji ukoliko bi broj rundi bio manji.

Linearna kriptoanaliza

Kao što je prethodno spomenuto, linearnu kriptoanalizu je uveo japanski kriptolog Mitsuru Matsui 1993. godine. On je zapazio da se neki bitovi ključa mogu prilično dobro aproksimirati linearnom funkcijom. Čini se da ova metoda nije bila poznata ni samim

tvorcima DES-a.

Označimo s $B[i]$ i-ti bit otvorenog teksta (bilo koje duljine) i definirajmo

$$B[i_1, i_2, ..., i_k] = B[i_1] \oplus B[i_2] \oplus ... \oplus B[i_k].$$

Označimo bitove otvorenog teksta s $P[1], P[2]$,..., šifrata s $C[1], C[2]$,..., te ključa s $K[1]$, $K[2]$,... Želimo naći linearne jednadžbe oblika:

$$P[i_1, i_2, ..., i_a] \oplus C[j_1, j_2, ..., j_b] = K[k_1, k_2, ..., k_c],$$

koje za slučajni otvoreni tekst i šifrat vrijede s vjerojatnošću $p \neq 0.5$ (što je p dalji od 0.5, to bolje). Tada računamo lijevu stranu za mnogo parova otvoreni tekst - šifrat kako bi dobili očekivanu vrijednost s desne strane. Ako isto to napravimo za više od $|p-0.5|^{-2}$ parova, vjerojatnost pogreške postaje vrlo mala. Preciznije, Matsui je pokazao da ta vjerojatnost iznosi 97.7%. Ako isto to napravimo za dvostruko više parova, vjerojatnost uspjeha se penje na 99.8%.

Matsui je promatrao vezu između bitova prije i poslije "prolaska" kroz različite S-kutije i primijetio, primjerice, da se prvi, drugi, treći, četvrti i šesti bit ulaza treće S-kutije poklapa s paritetom izlaza u točno 38 od 64 slučaja. Nadalje, drugi bit ulaza pete S-kutije poklapa se s XOR-om od sva 4 bita izlaza iste kutije s vjerojatnošću 0.19.

Linearna kriptoanaliza radi upravo tako da povezuje takve relacije. Sljedeće je pitanje koliko otvorenih tekstova mora biti proučeno. Matsui je pokazao da je u prosjeku potrebno 2^{43} takvih otvorenih tekstova. S mrežom od 12 radnih stanica, linearna kriptoanaliza je 1994. uspjela razbiti poruku šifriranu DES-om za 50 dana.

Poglavlje 2

AES

2.1 Povijest AES-a

U rujnu 1997. je, već ranije spomenuti, američki institut NIST objavio natječaj za novi standard u kriptografiji, *Advanced Encryption Standard*, odnosno AES. Natječaj je zaključen 15.6.1998. Od 21 pristigle prijave, 15 ih je zadovoljilo NIST-ove kriterije. U tri kruga procjene prijedloga, NIST i međunarodna znanstvena zajednica raspravljali su o prednostima i nedostacima svakog od prijedloga i tako polako smanjivali broj potencijalnih kandidata. U kolovozu 1999., NIST je objavio 5 finalista: MARS (IBM), RC6 (RSA Security Inc), Twofish (Counterpane Systems), Serpent (Anderson, Biham i Knudsen) te Rijndael (Rijmen i Daemen). U listpadu 2000., objavljeno je da je RIJNDAEL pobjednik natječaja za AES, a u studenom 2001. je i službeno prihvaćen kao američki federalni standard.

Što se samog natječaja tiče, NIST je postavio sljedeće zahtjeve na kriptosustav:

- morao je biti simetričan blokovni kriptosustav, s blokovima veličine 128 bitova,
- trebao je raditi s ključevima duljine 128, 192 i 256 bitova,
- trebao je biti siguran u odnosu na druge prijavljene kriptosustave,
- efikasnost i primjenjivost na software i hardware.

Očekuje se da će AES biti dominantan simetrični kriptosustav još barem nekoliko desetljeća. Zanimljivo je da je 2003. godine američka sigurnosna agencija (NSA) objavila da koristi AES u šifriranju svojih povjerljivih dokumenata. Do onda su se, za takve povjerljive stvari, koristili nepoznati tajni algoritmi.

2.2 Matematička strana AES-a: Konačna polja

Algoritam AES-a baziran je na aritmetici konačnih polja. Pogotovo se koriste za konstrukciju S-kutija. Zato ćemo najprije ponoviti bitne pojmove i tvrdnje iz algebarskih stuktura vezane uz konačna polja, s naglaskom na konačno polje $GL(2^8)$.

Grupa, prsten, polje

Konačno polje je, pojednostavljeno, konačan skup elemenata u kojem možemo zbrajati, oduzimati, množiti i dijeliti. Prije nego razjasnimo što je to polje, treba nam koncept jednostavnijih algebarskih struktura - grupe i prstena.

Definicija 2.2.1. *Neprazan skup G je* grupa *s obzirom na binarnu operaciju*

$$\cdot : G \times G \to G$$

ako vrijede sljedeća svojstva:

1. $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ *za sve* $x, y, z \in G$ *(*asocijativnost*),*
2. *postoji* $e \in G$ *takav da je*
$$e \cdot x = x \cdot e = x,$$
za sve $x \in G$ *(postojanje* neutralnog elementa*),*
3. *za sve* $x \in G$ *postoji* $y \in G$ *takav da je*
$$x \cdot y = y \cdot x = e$$
(postojanje inverznog elementa*,* $y = x^{-1}$*).*

Ako vrijedi i svojstvo komutativnosti*, to jest ako je* $x \cdot y = y \cdot x$ *za sve* $x, y \in G$ *onda kažemo da je grupa* Abelova *ili* komutativna*. Kraće, kažemo da je uređeni par* $(G, \cdot)$ *grupa, odnosno Abelova grupa.*

Grubo govoreći, grupa je skup s jednom operacijom i odgovarajućom inverznom operacijom. Ako se ta operacija naziva zbrajanje, odgovarajuća inverzna operacija je oduzimanje.

Prirodno je promatrati i skupove na kojima postoje dvije binarne operacije, koje najčešće tradicionalno nazivamo zbrajanje i množenje.

Definicija 2.2.2. *Neprazan skup R je* prsten *s obzirom na binarne operacije*

$$+ : R \times R \to R \quad i \quad \cdot : R \times R \to R$$

ako je ispunjeno sljedeće:

1. $(R, +)$ *je komutativna grupa s neutralnim elementom* 0 *(*nula*),*
2. $(R, \cdot)$ *je polugrupa, tj. množenje je asocijativno,*
3. *vrijedi distributivnost množenja prema zbrajanju, tj.*

$$x \cdot (y + z) = x \cdot y + x \cdot z$$

$$(x + y) \cdot z = x \cdot z + y \cdot z,$$

za sve $x, y, z \in F$.

Ako postoji jedinični element*, ili kraće* jedinica*,* $1 = 1_R \in R$*, takav da je*

$$1 \cdot x = x \cdot 1 = x,$$

$\forall x \in R$*, onda kažemo da je R prsten s jedinicom.*

Kako bismo u jednoj strukturi imali "pokrivene" sve četiri osnovne računske operacije, trebamo skup koji sadrži i aditivnu i multiplikativnu grupu. Takav skup nazivamo polje.

Definicija 2.2.3. *Neprazan skup F naziva se* polje *s obzirom na binarne operacije*

$$+ : F \times F \to F \quad i \quad \cdot : F \times F \to F$$

ako je ispunjeno sljedeće:

1. $(F, +)$ *je komutativna grupa s neutralnim elementom* 0 *(*nula*),*
2. $(F \backslash \{0\}, \cdot)$ *je komutativna grupa s neutralnim elementom* 1 *(*jedinica*)*
3. *vrijedi distributivnost množenja prema zbrajanju*

$$(x + y) \cdot z = x \cdot z + y \cdot z,$$

za sve $x, y, z \in F$.

Kraće, kažemo da je $(F, +, \cdot)$ *polje.*

Primjerice, skup realnih brojeva $\mathbb{R}$ je polje s 0 kao neutralnim elementom za aditivnu grupu (zbrajanje) i 1 kao neutralnim elementom za multiplikativnu grupu (množenje). Svaki realni broj x ima svoj inverz za zbrajanje, $-a$, a svaki realni broj a, različit od nule, ima svoj inverz za množenje, $1/a$.

U kriptografiji su često od interesa polja s konačnim brojem elemenata koja nazivamo konačna polja. *Red* konačnog polja je broj elemenata tog polja. Dakle, ako je $|F| = m$, onda kažemo da je F polje reda m. Prirodno je pitati se postoji li za svaki prirodan broj m neko konačno polje reda m. Odgovor na to pitanje daje nam sljedeći teorem

Teorem 2.2.4. *Polje reda m postoji ako i samo ako je m potencija prostog broja, $m = p^n$, gdje je n prirodan broj, a p neki prosti broj.*

Broj p iz prethodnog teorema naziva se *karakteristika* konačnog polja.

Prema Teoremu 2.2.4, primjerice, postoje konačna polja s 11 elemenata, ili s 81 elementom (jer je $3^4 = 81$), ili s 256 elemenata (jer je $2^8 = 256$), ali ne postoji konačno polje s npr. 12 elemenata, jer 12 nije potencija prostog broja.

Polje $\mathbb{Z}_p$

Najjednostavniji primjer konačnih polja su polja prostog reda, tj. ona u kojima je $n = 1$ (za $m = p$). Elementi polja $GF(p)$ mogu biti zapisani kao niz cijelih brojeva $0, 1, \ldots, p-1$. $GF(p)$ je polje uz operacije zbrajanja i množenja brojeva modulo p. Umjesto $GF(p)$, često se koristi oznaka $\mathbb{Z}_p$.

Teorem 2.2.5. *Neka je m neki prirodan broj, za $\mathbb{Z}_m = \{0, 1, ..., m-1\}$. Pokazuje se da je, uz zbrajanje i množenje modulo m, skup $\mathbb{Z}_m$ komutativni prsten s jedinicom.*
$\mathbb{Z}_m$ je polje ako i samo ako je $m = p$.

Kako bismo mogli računati u prostom polju, moramo se držati pravila prstena cijelih brojeva: zbrajanje i množenje se rade modulo p, aditivni inverz svakog elementa a je b iz $\mathbb{Z}_p$ takav da je $a + b \equiv 0 \pmod{p}$, a multiplikativni inverz svakog ne-nul elementa a je $x \in \mathbb{Z}_p$ koji je rješenje kongruencije $ax \equiv 1 \pmod{p}$. Evo i jednog primjera prostog polja.

Primjer 2.2.6. *Pogledajmo konačno polje $GF(5) = \{0, 1, 2, 3, 4\}$. Tablice ispod prikazuju kako zbrajati i množiti bilo koja dva elementa, kao i aditivne i multiplikativne inverze. Koristeći ove tablice, lako možemo obaviti sav potreban račun u ovom polju.*

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

·	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

$$-0 = 0, -1 = 4, -2 = 3, -3 = 2, -4 = 1,$$
$$1^{-1} = 1, 2^{-1} = 3, 3^{-1} = 2, 4^{-1} = 4.$$

Primjer 2.2.7. *Veoma važno prosto polje je* $GF(2) = \{0, 1\}$*, koje je ujedno i najmanje konačno polje. Računanje se u njemu odvija modulo 2 i daje sljedeće tablice:*

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Konačna polja $GF(2^m)$

U AES-u se koristi aritmetika konačnog polja koje sadrži 256 elemenata i označava se $GF(2^8)$. Ovo polje je odabrano jer svaki element polja može biti predstavljen jednim bajtom (8 bitova). Primjerice, koristi se za konstrukcije S-kutija i operaciju `MixColumns` (koju ćemo opisati kasnije), gdje AES svaki bajt podataka tretira kao element polja $GF(2^8)$ i manipulira podacima izvodeći računske operacije u tom konačnom polju.

Ipak, ako konačno polje nije prostog reda, a 2^8 to sigurno nije, operacije zbrajanja i množenja cijelih brojeva ne možemo izvoditi modulo 2^8. Kao prvi problem nameće se kako reprezentirati elemente takvog polja. U nastavku ćemo vidjeti da elemente konačnog polja $GF(2^m)$ dobro reprezentiraju polinomi čiji su koeficijenti elementi pripadnog prostog polja $\mathbb{Z}_2$. Nadalje, trebat ćemo opisati i operacije uz koje će taj skup polinoma biti polje.

Dakle, u polju $GF(2^m)$ elementi nisu cijeli brojevi nego polinomi s koeficijentima iz $GF(2)$. Najveći stupanj takvih polinoma je $m - 1$, stoga imamo ukupno m koeficijenata za svaki element konačnog polja. U polju $GF(2^8)$ je svaki element $A \in GF(2^8)$ reprezentiran s

$$A(x) = a_7x^7 + \cdots + a_1x + a_0, a_i \in GF(2) = \{0, 1\}.$$

Primijetimo da je upravo $256 = 2^8$ takvih polinoma jer to konačno polje ima 256 elemenata. Skup ovakvih 256 polinoma je konačno polje $GF(2^8)$ uz operacije zbrajanja i množenja koje ćemo opisati u dijelu koji slijedi. Važno je primijetiti i da se svaki polinom može vrlo jednostavno zapisati i spremiti u digitalnom obliku kao 8-bitni vektor

$$A = (a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0).$$

Nije potrebno zapisiviti i spremati faktore kao što su x^7 i x^6 jer je već iz same pozicije bita jasno kojoj potenciji x^i svaki koeficijent pripada.

Zbrajanje u $GF(2^m)$

Pogledajmo kako se zbraja i oduzima u konačnim poljima. Ključni dio AES-a koristi upravo takvo zbrajanje. Zbrajanje i oduzimanje u konačnim poljima se zapravo svodi na standardno zbrajanje i oduzimanje polinoma: zbrajamo i oduzimamo koeficijente uz iste potencije. Zbrajanje i oduzimanje koeficijenata radimo u polju $GF(2)$.

Definicija 2.2.8. *Neka su $A(x)$ i $B(x)$ polinomi iz $GF(2^m)$. Suma neka dva elementa se onda računa kao:*

$$C(x) = A(x) + B(x) = \sum_{i=0}^{m-1} c_i x^i,\ c_i \equiv a_i + b_i \pmod 2,$$

a razlika kao

$$C(x) = A(x) - B(x) = \sum_{i=0}^{m-1} c_i x^i,\ c_i \equiv a_i - b_i \equiv a_i + b_i \pmod 2.$$

Imajmo na umu da koeficijente zbrajamo (oduzimamo) modulo 2, što su zapravo iste operacije.

Evo i jednog primjera računa sume dvaju elemenata iz $GF(2^m)$.

Primjer 2.2.9. *Neka je $A(x) = x^7 + x^5 + x^4 + x + 1$ i $B(x) = x^4 + x^2 + x + 1$. Tada je:*

$$C(x) = A(x) + B(x) = (x^7) + (x^5) + (x^4 + x^4) + (x^2) + (x + x) + (1 + 1) = x^7 + x^5 + x^2.$$

Množenje u $GF(2^m)$

Množenje u $GF(2^m)$, odnosno u $GF(2^8)$, je jedna od ključnih operacija AES-a. U prvom koraku se dva elementa konačnog polja $GF(2^m)$, reprezentirana polinomima, množe koristeći standardno pravilo množenja polinoma. Neka su $A(x), B(x) \in GF(2^m)$,

$$A(x) = a_{m-1}x^{m-1} + ... + a_0,$$

$$B(x) = b_{m-1}x^{m-1} + ... + b_0.$$

Definiramo $C'(x) = A(x) \cdot B(x)$

$$C'(x) = A(x) \cdot B(x) = (a_{m-1}x^{m-1} + ... + a_0) \cdot (b_{m-1}x^{m-1} + ... + b_0),$$

i

$$C'(x) = c'_{2m-2}x^{2m-2} + ... + c'_0,$$

gdje su

$$\begin{aligned} c'_0 &= a_0 b_0 \mod 2, \\ c'_1 &= a_0 b_1 + a_1 b_0 \mod 2, \\ &\vdots \\ c'_{2m-2} &= a_{m-1} b_{m-1} \mod 2. \end{aligned}$$

Imajmo na umu da su svi koeficijenti a_i, b_i i c_i elementi $GF(2)$ i da s njima računamo u $GF(2)$. Produkt polinoma, u oznaci $C(x)$, će biti većeg stupnja od $m-1$ pa mora biti reduciran. Osnovna ideja je slična množenju u prostim poljima: pomnožimo dva cijela broja, podijelimo rezultat prostim brojem i na kraju promatramo samo ostatak. U konačnim poljima onda radimo slično: umnožak polinoma dijelimo određenim polinomom te nakon dijeljenja polinoma promatramo samo ostatak. Tu nam trebaju ireducibilni polinomi, to jest polinomi koje ugrubo možemo usporediti s prostim brojevima, a njihovi jedini faktori su broj 1 i oni sami.

Definicija 2.2.10. *Neka su $A(x)$ i $B(x)$ polinomi iz $GF(2^m)$ te neka je $P(x)$ ireducibilni polinom stupnja m, nad poljem $GF(2)$ ireducibilni polinom. Umnožak neka dva elementa $A(x), B(x)$ se onda računa kao*

$$C(x) \equiv A(x) \cdot B(x) \pmod{P(x)},$$

to jest $C(x)$ je ostatak pri dijeljenju umnoška polinoma $C'(x)$, $C'(x) = A(x) \cdot B(x)$, polinomom $P(x)$.

Iz definicije slijedi da svako polje $GF(2^m)$ zahtijeva ireducibilni polinom $P(x)$ stupnja m s koeficijentima iz $GF(2)$. Imajmo na umu da nije svaki polinom ireducibilan. Primjer reducibilnog polinoma je polinoma $x^4 + x^3 + x + 1$ jer je

$$x^4 + x^3 + x + 1 = (x^2 + x + 1)(x^2 + 1)$$

pa ga ne možemo iskoristiti pri konstrukciji polja proširenja $GF(2^4)$. U AES-u se koristi ireducibilni polinom

$$P(x) = x^8 + x^4 + x^3 + x + 1.$$

Primjer 2.2.11. *Želimo pomnožiti dva polinoma $A(x) = x^3 + x^2 + 1$ i $B(x) = x^2 + x$ u polju $GF(2^4)$. Ireducibilni polinom ovog konačnog polja je $P(x) = x^4 + x + 1$. Umnožak polinoma je*

$$C'(x) = A(x) \cdot B(x) = x^5 + x^3 + x^2 + x.$$

Sada možemo reducirati C' standardnim dijeljenjem polinoma. Nekad je lakše reducirati svaki od vodećih članova x^4 i x^5:

$$x^4 = 1 \cdot P(x) + (x + 1)$$

$$x^4 \equiv x + 1 \pmod{P(x)}$$
$$x^5 \equiv x^2 + x \pmod{P(x)}.$$

Sada nam još samo preostaje unijeti reducirani izraz za x^5 u rezultat $C'(x)$:

$$C(x) \equiv x^5 + x^3 + x^2 + x \pmod{P(x)}$$
$$C(x) = (x^2 + x) + (x^3 + x^2 + x) = x^3$$

Primjer 2.2.12. *Želimo pomnožiti dva polinoma $A(x) = x^6+x^4+x^2+x+1$ i $B(x) = x^7+x+1$ u polju $GF(2^8)$. Ireducibilni polinom ovog polja je $P(x) = x^8 + x^4 + x^3 + x + 1$. Umnožak polinoma je*

$$C'(x) = A(x) \cdot B(x) = x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1.$$

Standardnim dijeljenjem polinoma reduciramo C' i dobivamo ostatak $C(x) = x^7 + x^6 + 1$.

Važno je ne pomiješati množenje u $GF(2^m)$ sa množenjem cijelih brojeva. Prisjetimo se da se polinomi, odnosno elementi polja, obično u računalima pohranjuju kao vektori bitova. Ako se vratimo na prethodni primjer množenja i polinom $A(x)$ zapišemo kao vektor (1 1 0 1) i polinom $B(x)$ kao vektor (0 1 1 0), uočavamo da smo dobili umnožak x^3, odnosno (1 0 0 0). Taj umnožak **nije** identičan umnošku cijelih brojeva gdje bi rezultat množenja bio drugačiji. Ako polinome shvatimo kao cijele brojeve, tada je $(1101)_2 = 13_{10}$ i $(0110)_2 = 6_{10}$, a rezultat jednak $(1001110)_2 = 78_{10}$ što je očito različito od rezultata dobivenog množenjem u konačnom polju. Zaključak je da iako možemo elemente polja zapisati kao cijele brojeve, ipak se ne možemo olako služiti računom s cijelim brojevima.

Inverz u $GF(2^m)$

Invertiranje u $GF(2^8)$ je također jedna od ključnih operacija AES-a, posebice dijela koji sadrži S-kutije. Za konačno polje $GF(2^m)$ i odgovarajući ireducibilni polinom $P(x)$, inverz A^{-1} ne-nul elementa $A \in GF(2^m)$ je definiran kao:

$$A^{-1}(x) \cdot A(x) \equiv 1 \pmod{P(x)}.$$

U manjim poljima, koja sadrže 2^{16} ili manje elemenata, često se koriste tablice koje sadrže unaprijed izračunate inverze svih elemenata polja. Sljedeća tablica sadrži vrijednosti koje se koriste unutar S-kutije AES-a. Tablica sadrži sve inverze u $GF(2^8)$ modulo $P(x) = x^8 + x^4 + x^3 + x + 1$ u heksadekadskom zapisu. Poseban slučaj je element polja 0 za koji inverz ne postoji. Ipak, za S-kutiju AES-a, potrebna je i tablica supstitucija za svaku moguću vrijednost unosa. Zato je S-kutija dizajnirana tako da vrijednosti unosa 0 odgovara

vrijednost izlaza 0.

x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	8D	F6	CB	52	7B	D1	E8	4F	29	C0	B0	E1	E5	C7
1	74	B4	AA	4B	99	2B	60	5F	58	3F	FD	CC	FF	40	EE	B2
2	3A	6E	5A	F1	55	4D	A8	C9	C1	0A	98	15	30	44	A2	C2
3	2C	45	92	6C	F3	39	66	42	F2	35	20	6F	77	BB	59	19
4	1D	FE	37	67	2D	31	F5	69	A7	64	AB	13	54	25	E9	09
5	ED	5C	05	CA	4C	24	87	BF	18	3E	22	F0	51	EC	61	17
6	16	5E	AF	D3	49	A6	36	43	F4	47	91	DF	33	93	21	3B
7	79	B7	97	85	10	B5	BA	3C	B6	70	D0	06	A1	FA	81	82
8	83	7E	7F	80	96	73	BE	56	9B	9E	95	D9	F7	02	B9	A4
9	DE	6A	32	6D	D8	8A	84	72	2A	14	9F	88	F9	DC	89	9A
A	FB	7C	2E	3C	8F	B8	65	48	26	C8	12	4A	CE	E7	D2	62
B	0C	E0	1F	EF	11	75	78	71	A5	8E	76	3D	BD	BC	86	57
C	0B	28	2F	A3	DA	D4	E4	0F	A9	27	53	04	1B	FC	AC	E6
D	7A	07	AE	63	C5	DB	E2	EA	94	8B	C4	D5	9D	F8	90	6B
E	B1	0D	D6	EB	C6	0E	CF	AD	08	4E	D7	E3	5D	50	1E	B3
F	5B	23	38	34	68	46	03	8C	DD	9C	7D	A0	CD	1A	41	1C

Primjer 2.2.13. *Iz prethodne tablice čitamo da je inverz od*

$$x^7 + x^6 + x = (1100010)_2 = (C2)_{hex} = (xy)$$

dan elementom iz retka C i stupca 2:

$$(2F)_{hex} = (00101111)_2 = x^5 + x^3 + x^2 + x + 1.$$

To lako možemo provjeriti množenjem:

$$(x^7 + x^6 + x) \cdot (x^5 + x^3 + x^2 + x + 1) \equiv 1 \ (\text{mod } P(x)).$$

Spomenimo i da gornja tablica nije sama S-kutija koja je malo složenija i biti će kasnije opisana. Alternativa ovakvim tablicama je eksplicitno računanje inverza.

Polinomi s koeficijentima iz $GL(2^8)$

U AES-u se koriste i polinomi stupnja manjeg ili jednakog 3 s koefcijentima iz polja $GL(2^8)$. Dakle, to su *polinomi čiji su koeficijenti ponovo polinomi*. Svaki takav polinom zapravo reprezentira jedan vektor duljine 4 bajta. Primjer takvog polinoma je

$$57 \cdot x^3 + A0 \cdot x^2 + 1B \cdot x + 0F,$$

pri čemu koeficijente 57, $A0$, $1B$ i $0F$ zapisane heksadecimalno shvaćamo kao polinome iz $GL(2^8)$. Primjerice, $57 = x^6 + x^4 + x^2 + x + 1$.

Zbrajanje ovakvih polinoma nije problematično, jer zbrajamo koeficijente (polinome) uz iste potencije. Ako ovakva dva polinoma pomnožimo, dobit ćemo polinom stupnja

manjeg ili jednakog od 6 i zato ćemo ponovo morati gledati ostatak pri dijeljenju s nekim polinomom stupnja jednakog 4. Za potrebe AES-a, odabran je polinom $x^4 + 1$. Dakle, ako su

$$\begin{aligned} a(x) &= a_3x^3 + a_2x^2 + a_1x + a_0, \\ b(x) &= b_3x^3 + b_2x^2 + b_1x + b_0, \end{aligned}$$

gdje su $a_0, \ldots, a_3$ i $b_0, \ldots, b_3$ elementi konačnog polja $GL(2^8)$, tada definiramo

$$d(x) = a(x) \otimes b(x) = a(x) \cdot b(x) \mod (x^4 + 1).$$

Za

$$d(x) = d_3x^3 + d_2x^2 + d_1x + d_0,$$

pokazuje se da se koeficijenti $d_0, \ldots, d_3$ mogu dobiti kao umnožak sljedećih matrica

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix}.$$

Problem u ovom pristupu jest u tome da polinom $x^4 + 1$ nije ireducibilan nad poljem $GL(2)$. Zaista, $x^4 + 1 = (x + 1)^4$. To implicira da opisana struktura (skup polinoma stupnja manjeg od 4 čiji su koeficijenti iz $GL(2^8)$) nije polje! To jest, većina elemenata nema svoj multiplikativni inverz. No, za implementaciju u algoritam AES-a bit će bitno da samo jedan takav polinom ima inverz i to je polinom

$$a(x) = 03 \cdot x^3 + 01 \cdot x^2 + 01 \cdot x + 02.$$

Primjer 2.2.14. *Pokazujemo da je produkt polinoma* $a(x) = 03 \cdot x^3 + 01 \cdot x^2 + 01 \cdot x + 02$ *i* $b(x) = 0B \cdot x^3 + 0D \cdot x^2 + 09 \cdot x + 0E$ *jednak* $d(x) = 01$. *Najprije imamo:*

$$d_3 = 03 \cdot 0E + 01 \cdot 09 + 01 \cdot 0D + 02 \cdot 0B.$$

To računamo:

$$\begin{gathered} 03 \cdot 0E = (x + 1) \cdot (x^3 + x^2 + x) = x^4 + x = 12, \\ 01 \cdot 09 = 09, \\ 01 \cdot 0D = 0D, \\ 02 \cdot 0B = x \cdot (x^3 + x + 1) = x^4 + x^2 + x = 16. \end{gathered}$$

Tada je

$$d_3 = 12 + 09 + 0D + 16 = (x^4 + x) + (x^3 + 1) + (x^3 + x^2 + 1) + (x^4 + x^2 + x) = 00.$$

Na isti način računamo i d_2 :

$$d_2 = 01 \cdot 0E + 01 \cdot 09 + 02 \cdot 0D + 03 \cdot 0B$$

$$01 \cdot 0E = 0E,$$

$$01 \cdot 09 = 09,$$

$$02 \cdot 0D = x \cdot (x^3 + x^2 + 1) = x^4 + x^3 + x = 1A,$$

$$03 \cdot 0B = (x + 1) \cdot (x^3 + x + 1) = x^4 + x^3 + x^2 + 1 = 1D,$$

$d_2 = 0E + 09 + 1A + 1D = (x^3 + x^2 + x) + (x^3 + 1) + (x^4 + x^3 + x) + (x^4 + x^3 + x^2 + 1) = 00.$

Analogno računamo i d_1 :

$$d_1 = 01 \cdot 0E + 02 \cdot 09 + 03 \cdot 0D + 01 \cdot 0B$$

$$01 \cdot 0E = 0E,$$

$$02 \cdot 09 = x \cdot (x^3 + 1) = x^4 + x = 12,$$

$$03 \cdot 0D = (x + 1) \cdot (x^3 + x^2 + 1) = x^4 + x^2 + x + 1 = 16,$$

$$01 \cdot 0B = 0B,$$

$d_1 = 0E + 12 + 16 + 0B = (x^3 + x^2 + x) + (x^4 + x) + (x^4 + x^2 + x + 1) + (x^3 + x + 1) = 00.$

Jedino se d_0 *razlikuje od prethodnih:*

$$d_0 = 02 \cdot 0E + 03 \cdot 09 + 01 \cdot 0D + 01 \cdot 0B$$

$$02 \cdot 0E = x \cdot (x^3x^2 + x) = x^4 + x^3 + x^2 = 1C,$$

$$03 \cdot 09 = (x + 1) \cdot (x^3 + 1) = x^4 + x^3 + x + 1 = 1B,$$

$$01 \cdot 0D = 0D,$$

$$01 \cdot 0B = 0B,$$

$d_0 = 1C + 1B + 0D + 0B = (x^4 + x^3 + x^2) + (x^4 + x^3 + x + 1) + (x^3 + x^2 + 1) + (x^3 + x + 1) = 01.$

Dakle, $d(x) = 01$, *odnosno* $b(x)$ *je inverz polinoma* $a(x)$ *s obzirom na operaciju* $\otimes$.

2.3 Opis algoritma

Algoritam AES-a je gotovo identičan predloženom Rijndaelu. Jedina razlika je što veličine blokova i ključa Rijndaela variraju između 128, 192 i 256 bitova, a AES standard radi s blokovima veličine 128 bitova.

Kao što je i prije spomenuto, NIST je od kandidata zahtijevao 3 različite duljine ključa. Broj rundi u AES-u ovisi upravo o dužini ključa, i to prema sljedećoj tablici:

duljine ključa	broj rundi
128 bitova	10
192 bita	12
256 bitova	14

Za razliku od DES-a, AES ne koristi Feistelovu funkciju. Prisjetimo se, DES ne šifrira cijeli blok odjednom nego ga dijeli u dva bloka po 32 bita dok, s druge strane, AES šifrira svih 128 bitova odjednom, bez ikakvih dijeljenja u manje blokove. Upravo zato AES ima manji broj rundi.

Ulaz (input) je blok od 128 bitova (16 bajtova) s elementima iz konačnog polja $GF(2^8)$. Tih 16 bajtova možemo prikazati i 4x4 matricom koju nazivamo AES-blok:

A_0	A_4	A_8	A_{12}
A_1	A_5	A_9	A_{13}
A_2	A_6	A_{10}	A_{14}
A_3	A_7	A_{11}	A_{15}

Kako je i prije spomenuto, AES se sastoji od 10 rundi. Svaka runda ima 4 operacije:

- `Byte Substitution` (skraćeno `SubBytes`),
- `ShiftRows`,
- `MixColumns` i
- `Key Addition` (skraćeno `AddRoundKey`).

`ShiftRows` i `MixColumns` se često jednim imenom nazivaju i difuzijskim operacijama. Svaka od operacija radi s čitavim blokom od 128 bitova. Svaka se runda, osim prve, sastoji od sve 4 operacije. Također, u zadnjoj rundi algoritma se ne obavlja `MixColumns` operacija što cijeli algoritam onda čini simetričnim. Prije spomenutih 10 rundi, vrši se inicijalna runda dodavanja ključa, odnosno operacija `AddRoundKey`.

Operacija SubBytes

Prva operacija svake runde AES-a je `SubBytes`. Ona je jedini nelinearan dio algoritma i sastoji se od dva koraka:

1. svaki element AES-bloka zamijeni se svojim inverzom u $GF(2^8)$ (osim elementa 00 koji jedini nema inverz pa zato ostaje nepromijenjen),
2. primijeni se fiksna afina transformacija

$$b'_i = b_i \oplus b_{(i+4)mod8} \oplus b_{(i+5)mod8} \oplus b_{(i+6)mod8} \oplus b_{(i+7)mod8} \oplus c_i,$$

gdje je c =01100011.

Ovu transformaciju često zapisujemo kao S-kutiju. Svaki element AES-bloka (matrice) zapišemo heksadecimalno, npr $b_i = (n_1, n_2)$, gdje je n_1 broj retka, a n_2 broj stupca S-kutije.

x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Primjer 2.3.1. *Pretpostavimo da je bajt ulaza u S-kutiju $b_i = (C2)_{hex}$. Tada je vrijednost koja ga mijenja:*

$$S((C2)_{hex}) = (25)_{hex}$$

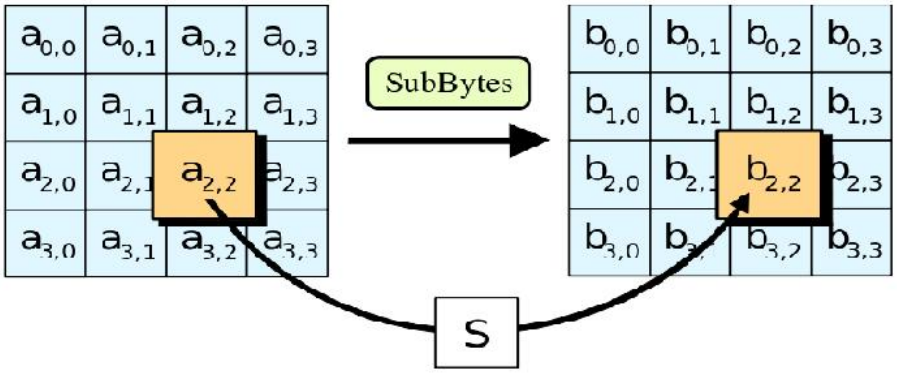
Na bitovnoj razini, ovu supstituciju možemo zapisati kao:

$$S(11000010) = (00100101).$$

Kako je `SubBytes` jedini nelinearni dio AES-a, očito je da onda ne vrijedi

$$\texttt{SubBytes}(A) + \texttt{SubBytes}(B) \neq \texttt{SubBytes}(A + B).$$

Supstitucija S-kutijom je bijekcija, što znači da svaki od $2^8 = 256$ mogućih elemenata ulaza odgovara točno jednom elementu izlaza. To nam omogućava vrlo jednostavno provođenje obrnutog postupka, što nam treba kod dešifriranja. Najčešće se u raznim softverskim implementacijama koristi gore prikazana S-kutija, s fiksiranim elementima (zapisanima u heksadekadsom obliku).



Slika 2.1: Shema `SubBytes` operacije

Operacija `ShiftRows`

Difuzijske operacije su, za razliku od `SubBytes` dijela, linearne, odnosno vrijedi

$$DIFF(A) + DIFF(B) = DIFF(A + B).$$

`ShiftRows` operacija ciklički pomiče elemente $n-$tog retka AES-bloka za i mjesta ulijevo ($i = 0, 1, 2, 3$). Konkretno, drugi redak trenutne matrice ciklički jedan bajt ulijevo, treći redak dva bajta ulijevo i četvrti redak tri bajta ulijevo. Prvi redak ostaje nepromijenjen. Primjerice, ako je ulaz `ShiftRows` operacije matrica $B = (B_0, B_1, ..., B_{15})$, odnosno:

B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

onda je izlaz nova matrica:

B_0	B_4	B_8	B_{12}
B_5	B_9	B_{13}	B_1
B_{10}	B_{14}	B_2	B_6
B_{15}	B_3	B_7	B_{11}

Operacija `MixColumns`

Ova operacija zapravo miješa svaki stupac matrice. Budući da svaki bajt ulaza utječe na četiri bajta izlaza, `MixColumn` je jedna od najvažnijih operacija u AES-u.

Stupci matrice AES-bloka se shvate kao polinomi s koeficijentima iz $GF(2^8)$. Dakle, stupac $A_i = a_{0i}a_{1i}a_{2i}a_{3i}$ shvatimo kao polinom $a_{3i}x^3 + a_{2i}x^2 + a_{1i} + a_{0i}$. Te polinome množimo polinomom $03x^3 + 01x^2 + 01x + 02$ (jedinim polinomom koji ima inverz), odnosno fiksnom 4x4 matricom

$$\begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix}.$$

Primjer računa:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}.$$

$$b_0 = 02a_0 + 03a_1 + 01a_2 + 01a_3,$$

$$b_1 = 01a_0 + 02a_1 + 03a_2 + 01a_3,$$

$$b_2 = 01a_0 + 01a_1 + 02a_2 + 03a_3,$$

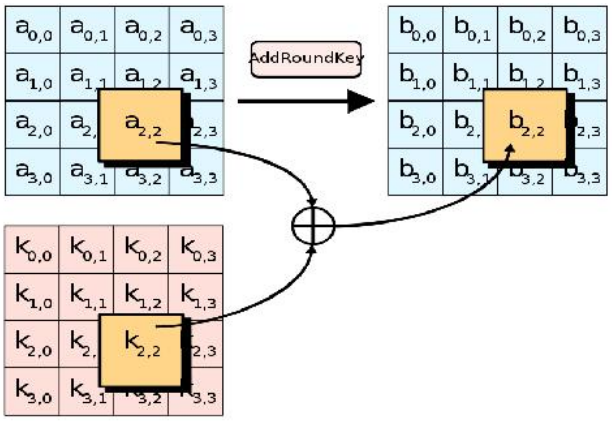
$$b_3 = 03a_0 + 01a_1 + 01a_2 + 02a_3.$$

Prisjetimo se, svaki b_i i a_i bajt je zapravo 8-bitna vrijednost reprezentirana elementom iz $GF(2^8)$, polju u kojem se obavlja i sav račun s koeficijentima. Konstante u gornjoj matrici su zapravo zapisane u heksadekadskom zapisu: 01 se odnosi na polinom iz konačnog polja $GF(2^8)$, s koeficijentima (00000001), odnosno to je element 1; 02 se odnosi na polinom iz istog konačnog polja s koeficijentima (00000010), odnosno to je element x; 03 se odnosi na polinom s vektorskim zapisom (00000011), odnosno $x + 1$.

Konstante 01, 02 i 03 su prvotno izabrane jer je njihova softverska implementacija vrlo jednostavna. Množenje s 01 je množenje identitetom i ne zahtijeva neku posebnu operaciju. Množenje brojevima 02 i 03 je opet vrlo jednostavno pomoću tablica s 256 elemenata (16×16 tablice). Kao alternativa za množenje brojem 02, implementirano može biti i množenje s x što je zapravo pomak ulijevo za jedan bit i redukcija sa $P(x) = x^8 + x^4 + x^3 + x + 1$. Slično, alternativa za množenje brojem 03, koji je reprezentiran polinomom $x + 1$, je množenje tim istim polinomom, nakon kojeg opet slijedi redukcija polinomom $P(x)$. Važno je napomenuti da je redukcija polinomom $P(x)$ potrebna ako i samo ako dobijemo polinom stupnja većeg od 7.

Operacija `AddRoundKey`

Ovdje zapravo XOR operacijom kombiniramo međuključeve, koji odgovaraju trenutnoj rundi, s AES-blokom (matricom). Svaki bajt AES-bloka kombiniramo s odgovarajućim bajtom međuključa (bajtovi na istoj poziciji u matrici):



Slika 2.2: Shema kombiniranja AES-bloka i međuključa

Podsjetimo, XOR operacija je zapravo zbrajanje u $GF(2)$. Međuključevi su iste veličine kao i AES-blok, a nastaju iz glavnog ključa koristeći Rijndael redoslijed ključeva.

Međuključevi

Prije same konstrukcije međuključeva, odvija se tzv. predračun u kojem se ključ proširi korištenjem operacije XOR i cikličkog pomaka. Takav prošireni ključ se sastoji od 44 "riječi", odnosno vektora veličine 4 bajta. Međuključevi se biraju iz tog proširenog ključa tako da se prvi međuključ sastoji od prve 4 riječi, drugi od sljedeće četiri itd.

Što se samog predračuna, odnosno proširivanja ključa tiče, prve 4 riječi predstavljaju zadani ključ, a svaka sljedeća riječ r_i se rekurzivno dobiva iz prethodne r_{i-1} primjenom operacije XOR s riječi r_{i-4}. Ako je broj i višekratnik broja 4, onda se prije same XOR operacije riječ rotira jedno mjesto ulijevo (operacija `RotWord`). Primjerice, riječ $r_0r_1r_2r_3$ postaje riječ $r_1r_2r_3r_0$. Nakon toga, operacija `SubWord` uzima jedan po jedan bajt riječi i na njega primjenjuje S-kutiju. Na kraju, obavlja se operacija XOR s riječi $[02^{(i-4)/4}, 00, 00, 00]$.

Šifriranje teksta se odvija tek kasnije. Ovakav pristup često koriste računala kada treba jednim ključem šifrirati velike količine podataka.

Ako želimo dešifrirati neki šifrat, najprije je potrebno rekurzivno izračunati sve međuključeve i tek onda započeti s dešifriranjem šifrata. Zato je dešifriranje uvijek malo sporije od šifriranja.

Primjer 2.3.2. *Pokažimo kako se 128-bitni blok otvorenog teksta šifrira AES ključem iste duljine. Zapišimo najprije otvoreni tekst i ključ kao AES blokove. Elemente AES blokova zapisat ćemo kao heksadecimalne brojeve, najprije otvoreni tekst*

32	*88*	*31*	*e0*
43	*5a*	*31*	*37*
f6	*30*	*98*	*07*
a8	*8d*	*a2*	*34*

,

a onda i ključ

2b	*28*	*ab*	*09*
7e	*ae*	*f7*	*cf*
15	*d2*	*15*	*4f*
16	*a6*	*88*	*3c*

.

Prije samog algoritma, odvija se generiranje tablice međuključeva. Kako je i prije spomenuto, prve 4 riječi predstavlja zadani ključ. Kako indeksiranje počinje od 0, zapravo 5. riječ ima indeks 4 i na nju primjenjujemo `RotWord` *i* `SubWord` *transformacije, kao i na sve sljedeće s indeksima koji su višekratnici broja 4.*

Prije prve runde, obavljamo operaciju `AddRoundKey`*, odnosno XOR ulaznog AES bloka i ključa. Time dobivamo novu tablicu*

19	*a0*	*9a*	*e9*
3d	*f4*	*c6*	*f8*
e3	*e2*	*8d*	*48*
be	*2b*	*2a*	*08*

.

Na svaki od elemenata u tablici primijenimo `SubBytes` *operaciju, čime dobivamo*

d4	*e0*	*b8*	*1e*
27	*bf*	*b4*	*41*
11	*98*	*5d*	*52*
ae	*f1*	*e5*	*30*

.

Nakon toga, slijedi primjena `ShiftRows` *operacije, čime dobivamo*

d4	*e0*	*b8*	*1e*
bf	*b4*	*41*	*27*
5d	*52*	*11*	*98*
30	*ae*	*f1*	*e5*

.

Prisjetimo se, `MixColumns` *operacijom svaki se od stupaca množi ranije spomenutom fiksnom matricom iz 2.3. Time dobivamo*

04	*e0*	*48*	*28*
66	*cb*	*f8*	*06*
81	*19*	*d3*	*26*
e5	*9a*	*7a*	*4c*

.

Na kraju svake runde, obavlja se XOR trenutnog AES-bloka s ranijem generiranim međuključem za tu rundu. Ovdje time dobivamo

a4	*68*	*6b*	*02*
9c	*9f*	*5b*	*6a*
7f	*35*	*ea*	*50*
f2	*2b*	*43*	*49*

.

Isto izvršavamo još 9 rundi, s time da u zadnjoj rundi izostavljamo `MixColumns` *operacije. Dobiveni šifrat, zapisan u AES-bloku je onda*

39	*02*	*dc*	*19*
25	*dc*	*11*	*6a*
84	*09*	*85*	*0b*
1d	*fb*	*97*	*32*

.

2.4 Dešifriranje

Kako AES, za razliku od DES-a, ne sadrži Feistelovu funkciju, moraju se koristiti inverzi svih operacija. `SubBytes` postaje `InvSubBytes`, `MixColumn` postaje `InvMixColumn` itd. Također, koristimo obrnuti redoslijed međuključeva.

Kako se u zadnjoj rundi šifriranja ne provodi `MixColumn` operacija, prva runda invertiranja onda ne koristi odgovarajuću inverznu operaciju. Što se ostalih rundi AES-a tiče, svaka od njih sadrži svaku od operacija, stoga u njima koristimo svaku od invertiranih operacija. Kako je XOR operacija sama sebi inverzna, `AddRoundKey` operacija je ista kod šifriranja i dešifriranja, odnosno jednaka je `InvAddRoundKey` operaciji.

2.5 *Brute force* napadi na AES

Kako je i sam DES ”pao” pod *brute force* napadima, odnosno napadima grubom silom, zanima nas može li se isto dogoditi u skorijoj budućnosti i s trenutnim svjetskim standardom kriptiranja podataka, AES kriptosustavom.

Poznato je da je DES pao ponajprije zbog prekratkog ključa od samo 56 bitova. Kako napadi grubom silom podrazumijevaju provjeru svih mogućih kombinacija ključeva, duži ključevi znače i teže probijanje sustava grubom silom.

Naravno, ako im damo beskonačno mnogo vremena, logično je da će napadi grubom silom u nekom trenutku dati rezultata. Kako se ovdje radi o izuzetno osjetljivom području kojim se dnevno vrti milijarde i milijarde dolara, korisnike sustava zanima može li sustav pasti i brže.

Sljedeća tablica prikazuje ukupan broj mogućih ključeva u ovisnosti o nihovoj duljini:

Duljina ključa	Broj mogućih kombinacija
1 bit	2
2 bita	4
4 bita	16
8 bitova	256
16 bitova	65536
32 bita	$4.2 \cdot 10^9$
56 bitova (DES)	$7.2 \cdot 10^{16}$
64 bita	$1.8 \cdot 10^{19}$
128 bitova (AES)	$3.4 \cdot 10^{38}$
192 bita (AES)	$6.2 \cdot 10^{57}$
256 bitova (AES)	$1.1 \cdot 10^{77}$

U tablici vidimo da je broj mogućih kombinacija ključeva za standardnu verziju AES-a, s ključem duljine 128 bitova, poprilično velik. Prema navodima u Wikipediji, trenutno postoje superračunala koja bi po sekundi mogla provjeriti $10.51 \cdot 10^{12}$ kombinacija ključeva. Kako jedna godina ima točno 31536000 sekundi, za probijanje AES-a, koji koristi ključ duljine 128 bitova, trebalo bi

$$(3.4 \cdot 10^{38}) : ((10.51 \cdot 10^{12}) \cdot 31536000) = 1.02 \cdot 10^{18}$$

godina, što je otprilike milijarda milijardi godina. Kao što je pokazano gornjim računom, čak ni superračunalo, za koje se samo pretpostavlja da postoji, ne bi moglo u dovoljno kratkom vremenskom roku probiti AES. Sljedeća tablica pokazuje maksimalna vremenska razdoblja probijanja sustava i za ostale duljine ključeva (s istim superračunalom):

Duljina ključa	Najduže moguće vrijeme probijanja
56 bitova (DES)	399 sekundi
128 bitova (AES)	$1.02 \cdot 10^{18}$ godina
192 bita (AES)	$1.872 \cdot 10^{37}$ godina
256 bitova (AES)	$3.31 \cdot 10^{56}$ godina

Iz gornje tablice se vidi da bi za probijanje AES-a, s ključem duljine 128 bitova, bilo potrebno oko milijardu milijardi godina, što je puno više čak i od starosti svemira, čija se starost procjenjuje na 13.75 milijardi godina. Iz navedenog je očito kako je napad grubom silom na AES prilično neučinkovit i nepraktičan.

Postoji još zanimljivih primjera. Ako bi svaka osoba na svijetu imala 10 računala, od kojih svako može provjeravati milijardu kombinacija ključeva po sekundi i ako uzmemo prosjek da se, što je praksa pokazala, ključevi pronalaze nakon provjere 50% ukupnog broja kombinacija, te pretpostavku da cijela populacija zemlje od 7 milijardi stanovnika sudjeluje u procesu, za pronalazak jednog ključa trebalo bi čak $7.7 \cdot 10^{25}$ godina.

Iako je očito da je napad grubom silom na AES nepraktičan, to ipak ukazuje na njegovu ranjivost, pogotovo s ubrzanim razvojem brzina procesora u posljednjih nekoliko godina. Pogotovo s pojavljivanjem raznih metoda kriptoanalize koje polako, ali sigurno, smanjuju ukupan broj potrebnih operacija i/ili mogućih kombinacija ključeva.

Bibliografija

[1] J. Daemen, V. Rijmen: *The Design of Rijndael - The Advanced Encryption Standard*, Springer, Berlin, 2002.,

[2] A. Dujella, M. Maretić: *Kriptografija*, Element, Zagreb, 2007.,

[3] A. Dujella: *Diskretna matematika - matematičke osnove kriptografije javnog ključa*, http://web.math.pmf.unizg.hr/~duje/diskretna/diskretna.pdf,(travanj, 2014.)

[4] S. Landau: *Communications Security for the Twenty-First Century: The Advanced Encryption Standard*, Notices of the AMS, vol. 47 (2000), 450.-459.,

[5] S. Landau: *Standing the Test of Time: The Data Encryption Standard*, Notices of the AMS, vol. 47 (2000), 341.-349.,

[6] C. Paar, J. Pelzl: *Understanding Criptography*, Springer, Bochum, 2009.,

[7] Advanced Encryption Standard - Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/Advanced_Encryption_Standard,(lipanj, 2014.)

[8] Data Encryption Standard - Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/Data_Encryption_Standard,(svibanj, 2014.)

[9] U.S. Department Of Commerce: Data Encryption standard (DES), http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf,(svibanj, 2014.)

Sažetak

Data Encryption Standard (DES) je naziv za simetrični kriptosustav šifriranja elektronskih podataka, objavljen od strane američkog instituta za standarde (NBS) 15. siječnja 1977. godine, nakon 4 godine dugog natječaja i procesa standardiziranja. Razvio ga je IBM-ov tim, na čelu s Horstom Feistelom, 1974. godine. DES primjenjuje ključ duljine 56 bitova na 64-bitne blokove podataka. Proces se odvija u nekoliko faza i uključuje 16 rundi operacija. Mnogi kriptoanalitičari su pokušali pronaći razne brze metode razbijanja DES-a. Početkom 1997. godine, zajedničkim snagama internetske populacije od 14000 korisnika računala koji su isprobavali razne šifre, jedna poruka je konačno dešifrirana otkrivanjem ključa nakon isprobavanja samo 18 od mogućih 72 kvadrilijuna mogućih ključeva. NIST je kasnije objavio da DES neće ponovno dobiti certifikat kao standard i da će prihvaćati prijedloge za njegove zamjene. Sljedeći prihvaćeni standard će biti poznat pod imenom *Advanced Encryption Standard*.

The Advanced Encryption Standard (AES) je specifikacija za način šifriranja elektronskih podataka, objavljen od strane američkog instituta za standarde i tehnologiju (NIST) 26. studenog 2001. godine, nakon petogodišnjeg procesa natječaja i standardizacije u kojem se natjecalo 15 kandidata, od kojih je Rijndael procijenjen najboljim i odabran kao budući standard. Svaki AES šifrat se sastoji od blokova veličine 128-bitova i ključeva veličine 128, 192 i 256 bitova. AES je mnogo puta analiziran što je dovelo do upotrebe na svjetskoj razini, kao što je to bila situacija i s njegovim prethodnikom, *Data Encryption Standard*-om. AES je prvi javno objavljeni i u potpunosti dostupni način šifriranja podatak, odobren i od američke sigurnosne agencije NSA.

Summary

The Data Encryption Standard (DES) is a specification for a symmetric-key algorithm for the encryption of electronic data adopted by the U.S. National Bureau of Standards (NBS) on January 15, 1977 after a 4 year-long contest and standardization process. It was developed by an IBM team, with Horst Feistel's lead, around 1974. DES applies a 56-bit key to each 64-bit block of data. The process can run in several modes and involves 16 rounds or operations. Many cryptanalysts have attempted to find shortcuts for breaking the system. Early in 1997, a cooperative effort on the Internet of over 14,000 computer users trying out various keys finally deciphered the message, discovering the key after running through only 18 quadrillion of the 72 quadrillion possible keys. NIST has later indicated DES that will not be recertified as a standard and submissions for its replacement are being accepted. The next standard will be known as the Advanced Encryption Standard (AES).

The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data adopted by the U.S. National Institute of Standards and Technology (NIST) on November 26, 2001 after a 5-year standardization process in which fifteen competing designs were presented and evaluated before Rijndael was selected as the most suitable. Each AES cipher has a 128-bit block size, with key sizes of 128, 192 and 256 bits. The AES ciphers have been analyzed extensively and are now used worldwide, as was the case with its predecessor, the Data Encryption Standard (DES). AES is the first publicly accessible and open cipher approved by the NSA.

Životopis

Rođen sam 24.8.1990. godine u Koprivnici. Od šeste godine živim u Ferdinandovcu, gdje sam pohađao i osnovnu školu. Tijekom tog razdoblja, sudjelovao sam na natjecanjima različitih razina, od školske do državne, iz različitih predmeta: matematike, geografije i ekologije te istaknuti rad u dramsko-recitatorskoj sekciji. Najveći uspjeh na natjecanjima mi je četvrto mjesto iz geografije na državnom natjecanju 2005. godine. Iste godine sam nagrađen za najsvestranijeg učenika i učenika s odličnim uspjehom kroz cijelo školovanje.

Pohađao sam opću gimnaziju dr. Ivana Kranjčeva u Đurđevcu. Tijekom tog razdoblja, sudjelovao sam na natjecanjima različitih razina iz matematike i geografije. Maturirao sam 2009. godine, uz oslobođenje polaganja mature zbog odličnog uspjeha kroz cijelo školovanje, s maturalnim radom imena iz geografije imena "Norveška".

Godine 2009. sam upisao preddiplomski studij matematike, nastavnički smjer, na matematičkom odsjeku Prirodoslovno-matematičkog fakulteta u Zagrebu. Isti preddiplomski studij sam završio 2012. godine. Godine 2012. sam upisao diplomski studij matematike, nastavnički smjer, na matematičkom odsjeku Prirodoslovno-matematičkog fakulteta u Zagrebu.