Simetrična kriptografija
3DES, AES,
Modovi rada blok šifratora

dr Slavica Tomović

Univerzitet Crne Gore

3DES

- Baziran na trostrukom izvršavanju DES algoritma i korišćenju 3 ključa
- Zašto ne 2DES?
  - $C = E(K_2, E(K_1, P))$
  - Dekripcija zahtjeva primjenu ključeva u obrnutom redosledu
  - $P = D(K_1, D(K_2, C))$
  - Ekvivalentna dužina ključa je dva puta veća ali to ne rešava problem
  - Moguće je pronaći ključ K_3 takav da je zadovoljena jednakost:

$$E(K_2, E(K_1, P)) = E(K_3, P)$$

  - Ponavljanje enkripcije 2 ili čak više puta nema smisla jer je postupak ekvivalentan jednostrukoj enkripciji
  - S obzirom da je broj mogućih reverzibilnih mapiranja za blok ulaznih vrijednosti $2^{64}!$, a da DES definiše mapiranje preko ključa sa $2^{56} (\ll 2^{64}!)$ mogućih vrijednosti, realno je pretpostaviti da dvostruka primjena DES algoritma vrlo vjerovatno neće rezultovati mapiranjem koje odgovara jednostrukoj DES enkripciji.

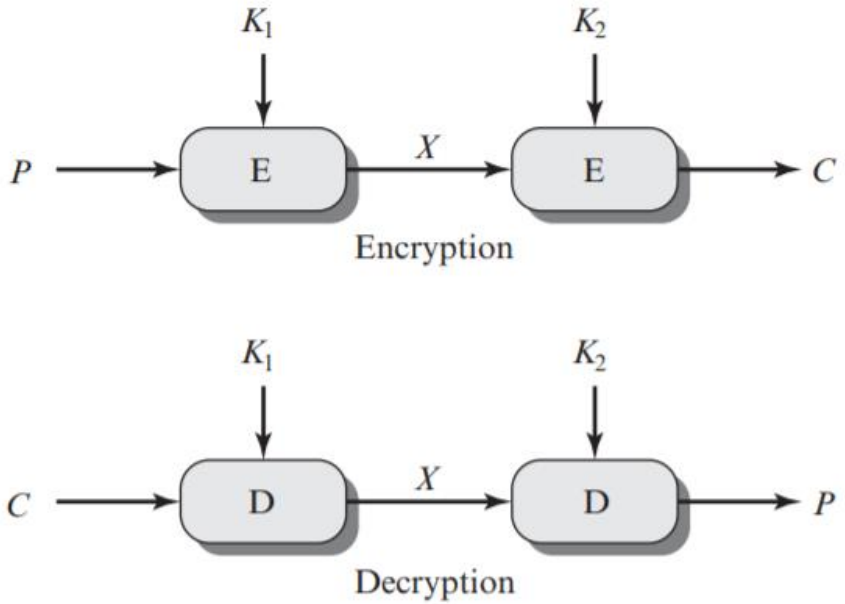
Zašto ne 2DES?

- Najveća prijetnja za 2DES su *Meet-in-the-middle* napadi.
- *Meet-in-the-middle* napadi su bazirani na observaciji da ukoliko imamo:

$$C = E(K_2, E(K_1, P))$$

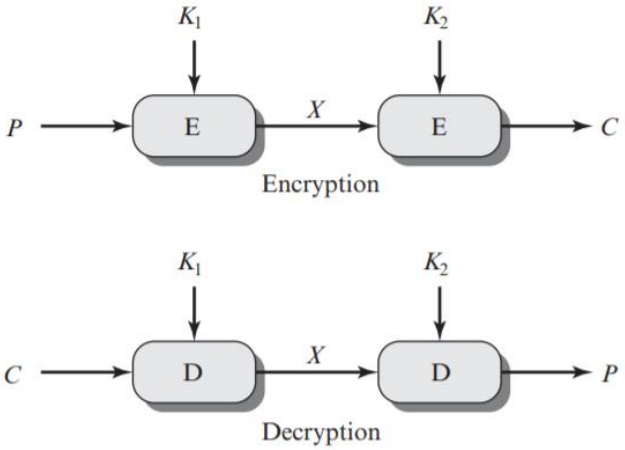
onda je:

$$X = E(K_1, P) = D(K_2, C)$$



Zašto ne 2DES?

- Za poznati par P i C napad se izvodi na sledeći način:
  - Šifrovati P sa svih 2^{56} mogućih vrijednosti ključa K_1;
  - Napraviti tabelu svih rezultata i sortirati po X;
  - Dekriptovati C koristeći svih 2^{56} mogućih vrijednosti ključa K_2, upoređujući svaki rezultat sa X iz tabele generisane u prethodnom koraku;
  - Ukoliko se ustanovi poklapanje, testirati dva rezultujuća ključa na novom poznatom paru (P,C); Ukoliko ključevi generišu očekivani *ciphertext,* napad je uspješno realizovan.



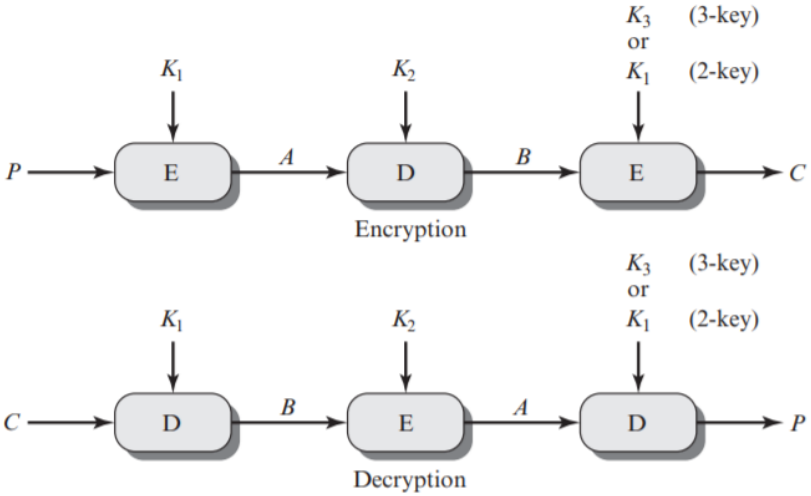
3DES sa dva ključa

- 3DES koristi tri enkripcije
- Za tri enktipcije se u opštem slučaju koriste 3 ključa, ali mogu se koristiti i 2 ključa sa E-D-E procedurom

$$C = \mathrm{E}(K_1, \mathrm{D}(K_2, \mathrm{E}(K_1, P)))$$
$$P = \mathrm{D}(K_1, \mathrm{E}(K_2, \mathrm{D}(K_1, C)))$$

- Dekripcija u drugom koraku nema kriptografski značaj.
  - Njena jedina prednost je što omogućava korisnicima 3DES algoritma da dekriptuju poruke korinika jednostrukog DES algoritma

$$C = \mathrm{E}(K_1, \mathrm{D}(K_1, \mathrm{E}(K_1, P))) = \mathrm{E}(K_1, P)$$
$$P = \mathrm{D}(K_1, \mathrm{E}(K_1, \mathrm{D}(K_1, C))) = \mathrm{D}(K_1, C)$$



3DES algoritam (2 ili 3 ključa)

3DES sa tri ključa

- Iako se ne zna za napade na 3DES sa dva ključa, postoje neke naznake da ih ima.
- Može se koristiti 3 DES sa 3 ključa da bi se otklonile i ove sumnje.
- Postupak enkripcije:

$$C = \mathrm{E}(K_3, \mathrm{D}(K_2, \mathrm{E}(K_1, P)))$$

- Kompatibilnost sa starijim DES verzijama ostvaruje se sa: $K_3 = K_2$ ili $K_1 = K_2$.
- Neke internet aplikacije koriste ovu tehnologiju, npr. PGP, S/MIME.

Advanced Encryption Standard (AES)

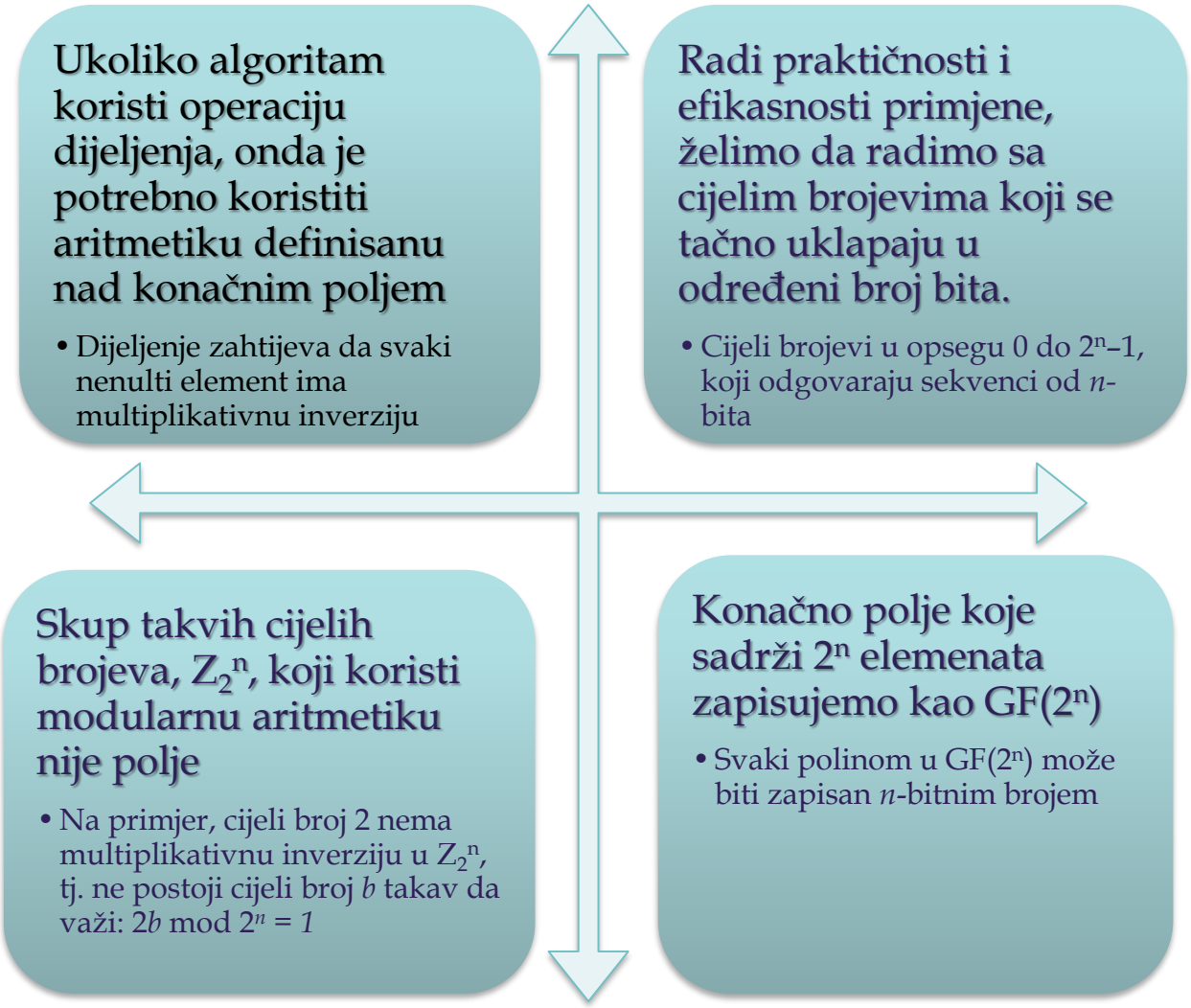
- AES je blok algoritam koji je zamijenio DES u komercijalnim aplikacijama.
- Enkripcija se vrši na nivou bloka od 128 bita.
  - Blokovi od 4 kolone po 4 bajta.
- Veličina ključa je 128, 192 ili 256 bita.
- AES **ne koristi** Feistel strukturu.
- Svaka itercija sastoji se od četiri zasebne funkcije:
  - Supstitucija bajtova
  - Permutacija
  - Supstitucija bazirana na aritmetičkim operacijam u GF(2^8)
  - XOR sa ključem

Aritmetika konačnih polja

- Kod AES (*Advanced Encryption Standard*) enkripcije sve operacije se obavljaju nad bajtima (8 bita).
- Aritmetičke operacije sabiranja, množenja i dijeljenja se obavljaju u konačnom polju GF(2^8).
  - Koristi se nesvodljivi polinom $m(x) = x^8 + x^4 + x^3 + x + 1$.
- Polje je skup nad kojim možemo obavljati operacije sabiranja, oduzimanja, množenja i dijeljenja, a da ne napustimo taj skup.
- Dijeljeljenje je definisano sledećim pravilom
  - $a / b = a\ (b^{-1})$

- Primjer konačnog polja (sa konačnim brojem elemenata) je skup Z_p kojeg čine cijeli brojevi {0, 1, ... , p - 1}, gdje je p prost broj i gdje se operacije obavljaju po modulu p.

Aritmetika konačnih polja



Porijeklo AES-a

- Postoji potreba za zamjenom DES-a.
  - Teoretski postoje analitički napadi koji ga mogu razbiti.
  - Napadi isprobavanjem svih mogućih vrijednosti ključa (*brute force*) postaju sve opasniji.
- Može se koristiti 3DES.
  - Sa većim ključem, a istom osnovnom strukturom algoritma, pokazano je da ne postoji kriptoanaliza koja ga je mogla oboritim osim *brute-force.*
  - Sporiji (tri puta se izvršava DES), a blokovi ostaju mali (64 bita).
  - 3DES koristi 48 iteracija da bi postigao sigurnost za koju su vjerovatno dovoljne 32 iteracije.
  - Softverske implementacije 3DES-a su prespore za neke primjene, posebno za digitalne video podatke.

Porijeko AES-a

- US NIST (*National Institute of Standards and Technology*) 1997. godine raspisuje konkurs za predlog novog AES algoritma enkripcije.
  - AES bi trebalo da ima sigurnost jednaku ili bolju od 3DES-a i znatno bolju efikasnost
  - Pored toga NIST specificira da bi AES morao da bude simetrični blok algoritam sa veličinom bloka od 128 bita i podrškom za veličine ključa od 128, 192, 256 bita.
- 15 kandidata je prihvaćeno u prvom krugu izbora 1998. godine.
- 5 kandidata je ušlo u uži izbor nakon drugog kruga izbora 1999.
- Izabran je Rijndael algoritam (dr. Joan Daemen i Dr. Vincent Rijmen) kao finalni AES 2000. godine.
- NIST je izdao novi standard (FIPS PUB 197) u novembru 2001. godine.

AES kriterijumi izbora

- Prvobitni kriterijumi (prvi i drugi krug):
  1. Sigurnost - napor da bi se nekom kriptoanalizom razbila šifra
     - *brute force* napadi su isključeni kao mogućnost jer je najmanja veličina ključa 128 bita.
  2. Cijena – efikasnost pri izračunavanju;
  3. Karakteristike algoritama i implementacije:
     - Fleksibilnost
     - Mogućnost različitih hardverskih i softverskih implementacija
     - Jednostavnost dizajna

AES kriterijumi izbora

- Konačni kriterijumi:
  1. Opšta sigurnost – algoritmi testirani od strane kriptografske zajednice u roku od tri godine (diferencijalna i linearna kriptoanaliza);
  2. Softverska implementacija – brzina, brzina u odnosu na platformu i brzina u odnosu na veličinu ključa;
  3. Okruženja sa ograničenim prostorom – memorijske kartice i sl:
  4. Hardverska implementacija – mogućnost optimizacije performansi;
  5. Napadi na implementaciju – dužina trajanja operacija;
  6. Enkripcija naspram dekripcije – stepen sličnosti;

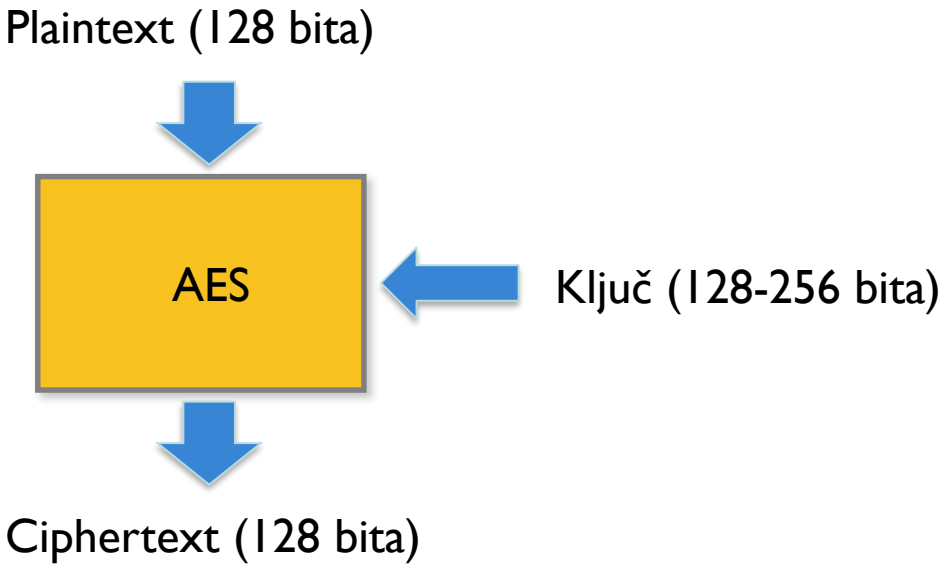
AES kriterijumi izbora

- …Konačni kriterijumi:
  7. Promjenjivost ključa – mogućnost brze zamjene ključa uz minimalne resurse;
  8. Fleksibilnost – mogućnost unapeđivanja algoritma (nove veličine blokova i/ili ključeva);
  9. Mogućnost maksimalnog iskorišćenja procesora.

AES ocjena Rijndael algoritma

1. Opšta sigurnost – nema poznatih napada.
2. Softverska implementacija - dobre performanse na različitim platformama.
3. Okruženja sa ograničenim prostorom – pogodan, mala potrošnja energije.
4. Hardverska implementacija – najbolji od svih finalista.
5. Napadi na implementaciju – najlakše se brani od ovakvih napada.
6. Enkripcija naspram dekripcije – razlikuju se.
7. Promjenjivost ključa – zahtjeva izvršavanje algoritma proširivanja ključa.
8. Fleksibilnost – može se prilagoditi veličina bloka/ključa na bilo koji umnožak od 32.
9. Mogućnost maksimalnog iskorišćenja procesora – odličan potencijal za paralelizam.

AES – Konceptualna šema



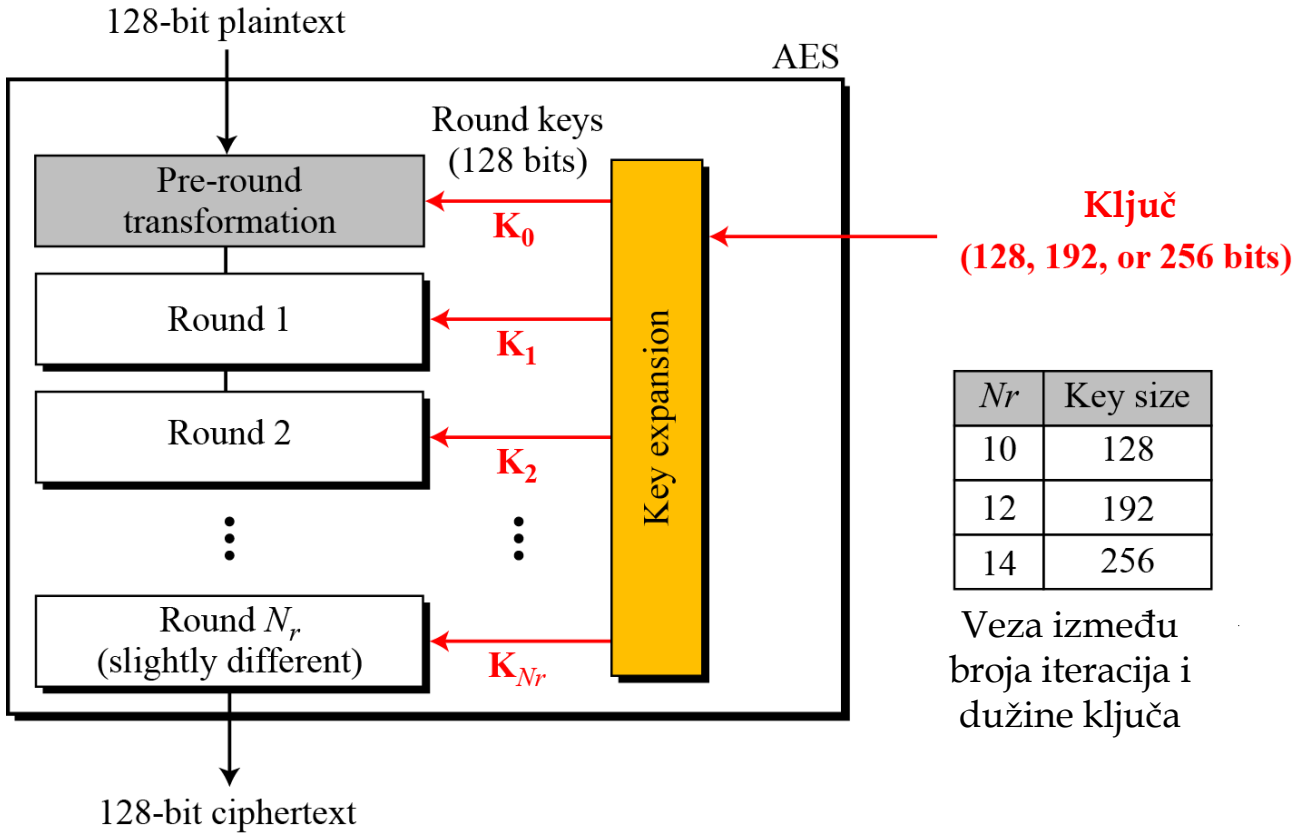
Veličina ključa (riječi/bajti/biti)	4/16/128	6/24/192	8/32/256
Veličina *plaintext*-a (riječi/bajti/biti)	4/16/128	4/16/128	4/16/128
Broj iteracija	10	12	14
Veličina ključa iteracije (riječi/bajti/biti)	4/16/128	4/16/128	4/16/128
Veličina proširenog ključa (riječii/bajti/biti)	44/176	52/208	60/240

Rijndael AES

- Autori Rijmen-Deamen iz Belgije
- Iterativni algoritam koji ne koristi Feistel strukturu
- Obrađuje podatke kao 4 grupe od po 4 bajta (128 bita)
  - Ove grupe u literature su poznate kao stanja (*states*)
- Ključ veličine 128/192/256 bita
- 128-bitni ključ se predstavlja kao matrica sa 4 kolone od po 4 bajta
  - Ključ se zatim proširuje u niz riječi od kojih svaka ima po 4 bajta
  - Ukupno ima 44 riječi za za ključ od 128 bita ($w[i],\ 1 \leq i \leq 44$)
  - Četiri različite riječi (128 bita) se koriste kao iterativni ključ u svakoj iteraciji
- U svakoj iteraciji izvršava operacije nad cijelim blokom podataka.

Rijndael AES

- Iteracije su (skoro) identične
  - Prva i zadnja iteracija se razlikuju



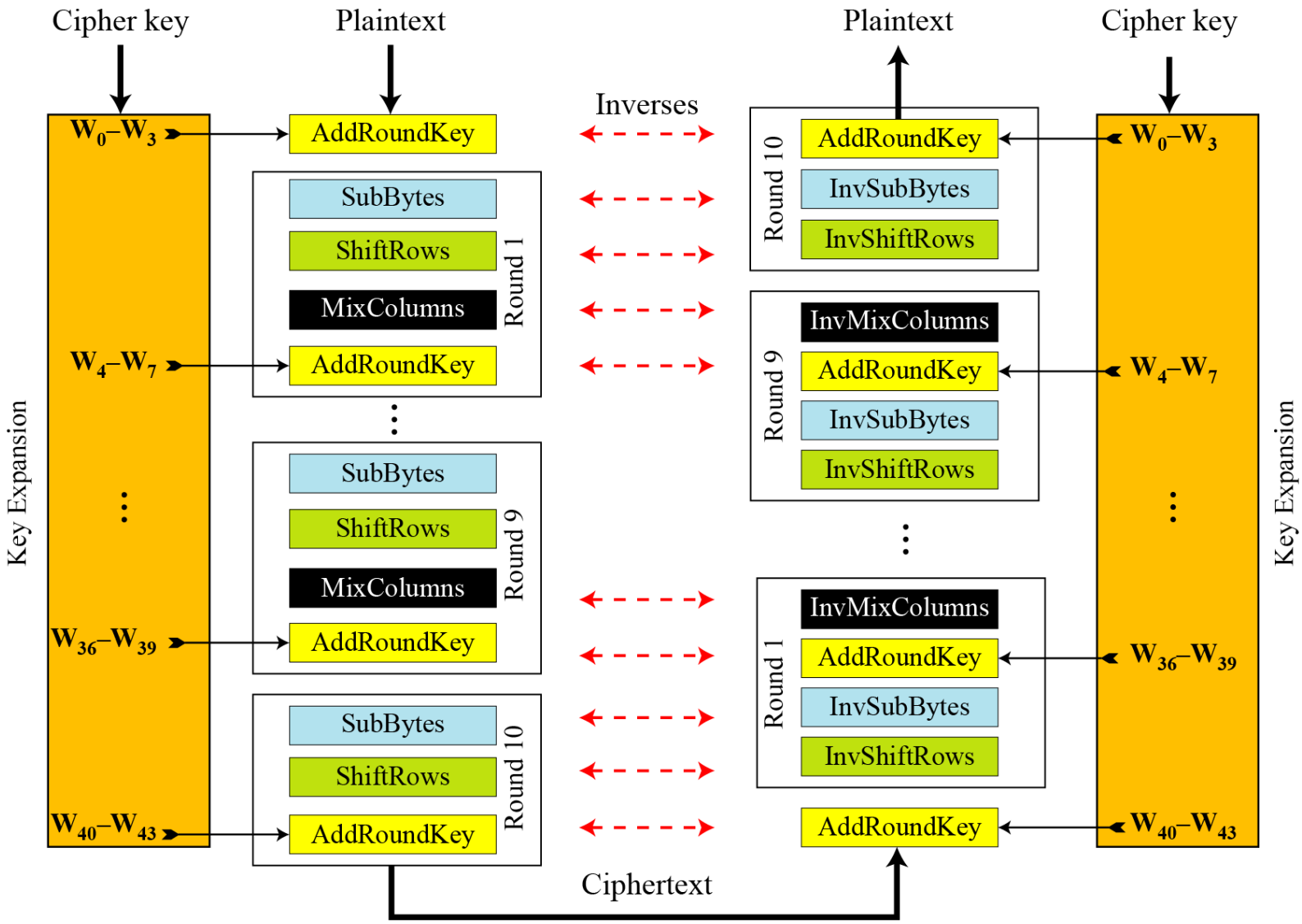
Nr	Key size
10	128
12	192
14	256

Veza između broja iteracija i dužine ključa

AES – sažeti opis

Proširenje ključa	• Iteracioni ključevi se izvode iz originalnog ključa koristeći Rijndael algoritam raspoređivanja.
Incijalna iteracija	• AddRoundKey: Svaki bajt stanja se kombinuje sa iteracionim ključem koristeći XOR operaciju nad bitima.
Iteracije	• SubBytes – nelinerni supstitucioni korak (1 S-box za svaki bajt) • ShiftRows – Permutuju se bajtovi između kolona • MixColumns – Supstitucija koja koristi aritmetiku u GF(2^8) • AddRoundKey – XOR stanja sa dijelom proširenog ključa
Finalna iteracija	• SubBytes • ShiftRows • AddRoundKey

AES cjelokupna struktura

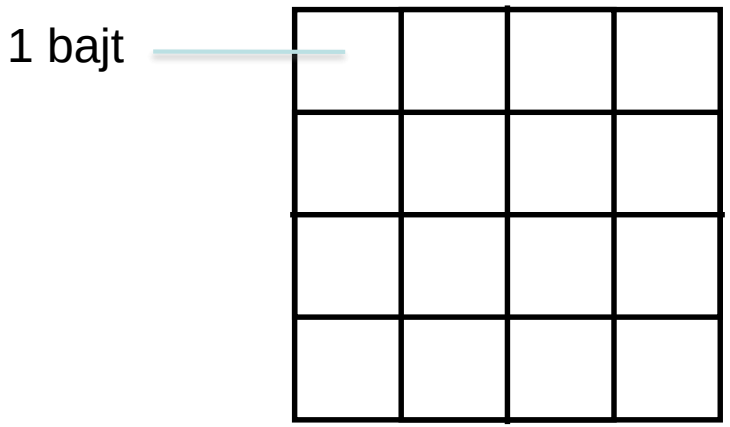


Karakteristike AES algoritma

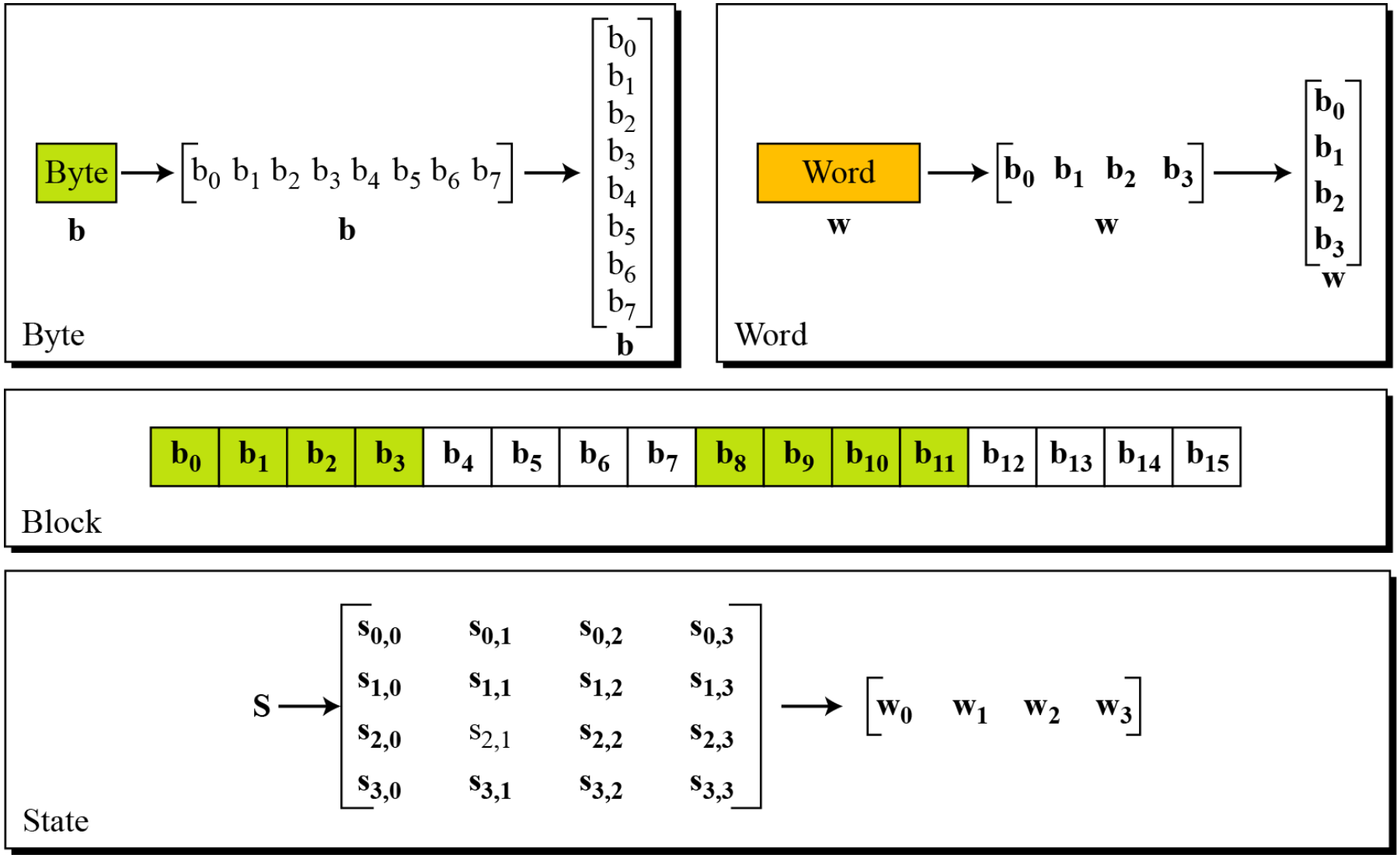
- Ne koristi Feistel strukturu.
- Počinje i završava se sa addRoundKey fazom, jer jedino ova faza koristi ključ.
- Ostale tri faze obezbjeđuju konfuziju, difuziju i nelinearnost, ali samostalno ne doprinose bezbjednosti jer ne koriste ključ.
- Svaka faza je invertibilna.
- Za AddRoundKey ivertibilnost se postiže XOR-om istog ključa iteracije i bloka.
- Za ostale faze koriste se inverzne funkcije pri dešifrovanju.
- Algoritam nije identičan kod enkripcije i dekripcije (mada se koristi prošireni ključ u inverznom redosledu)
- Finalna iteracija i kod enkripcije i kod dekripcije se sastoji samo od tri faze.

AES – struktura bloka podataka

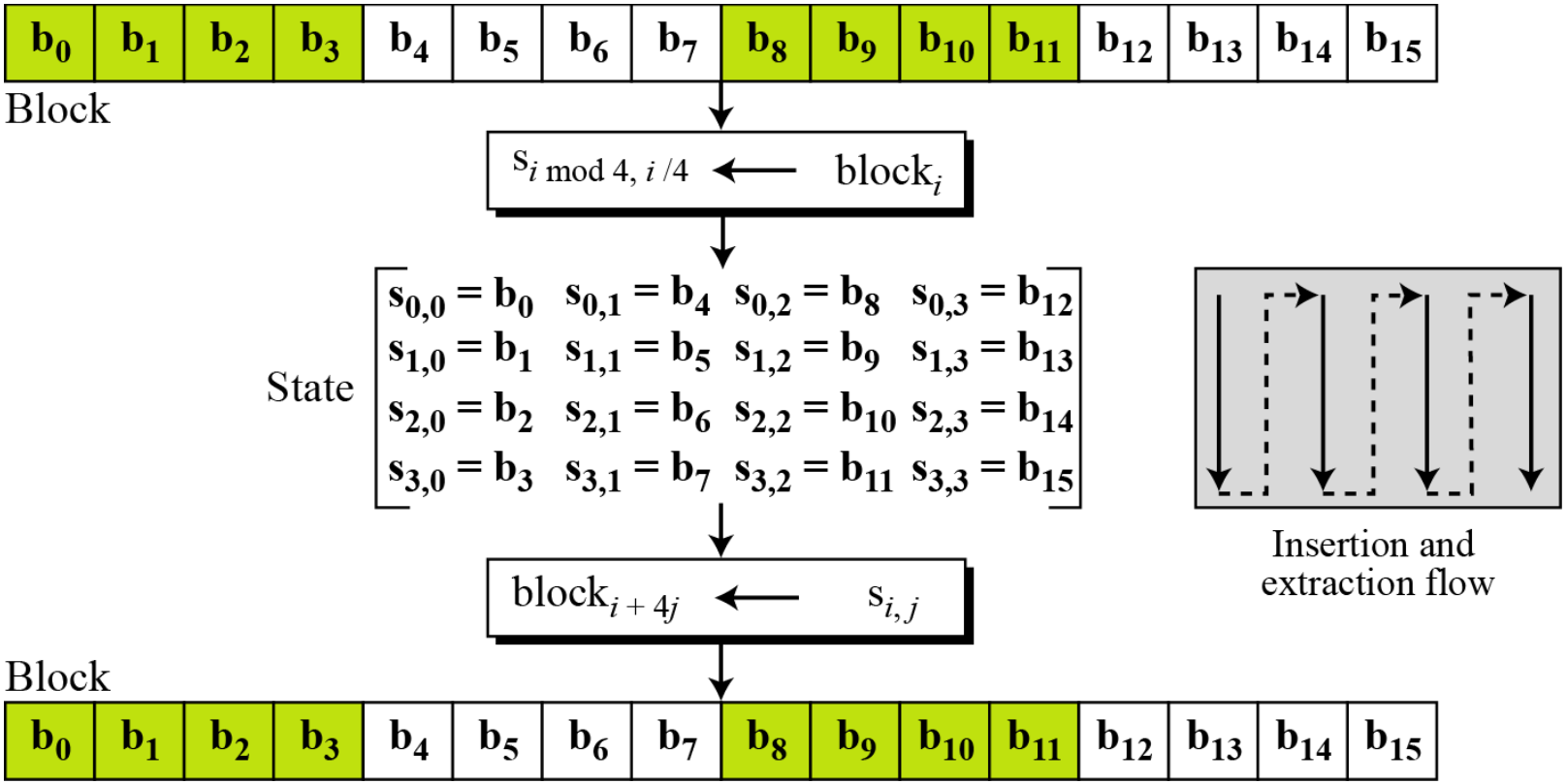
- Blok podataka se posmatra kako 4x4 tabela podataka.
- Predstavlja se 4x4 matricom bajtova.
- Ključ se proširuje u niz 32-bitnih riječi.



AES – struktura bloka podataka



AES – struktura bloka podataka

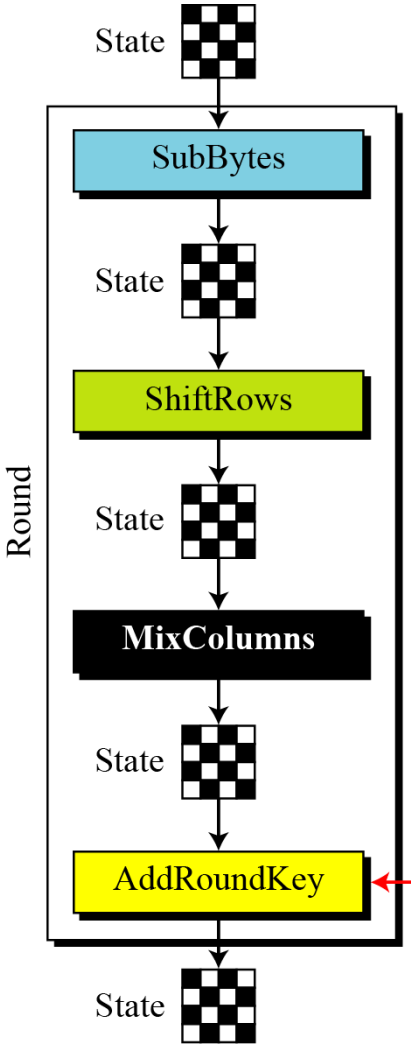


AES struktura bloka podataka

Text	A	E	S	U	S	E	S	A	M	A	T	R	I	X	**Z**	**Z**
Hexadecimal	00	04	12	14	12	04	12	00	0C	00	13	11	08	23	19	19

$$\begin{bmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 23 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{bmatrix}$$ State

AES – Struktura iteracije



Napomene:

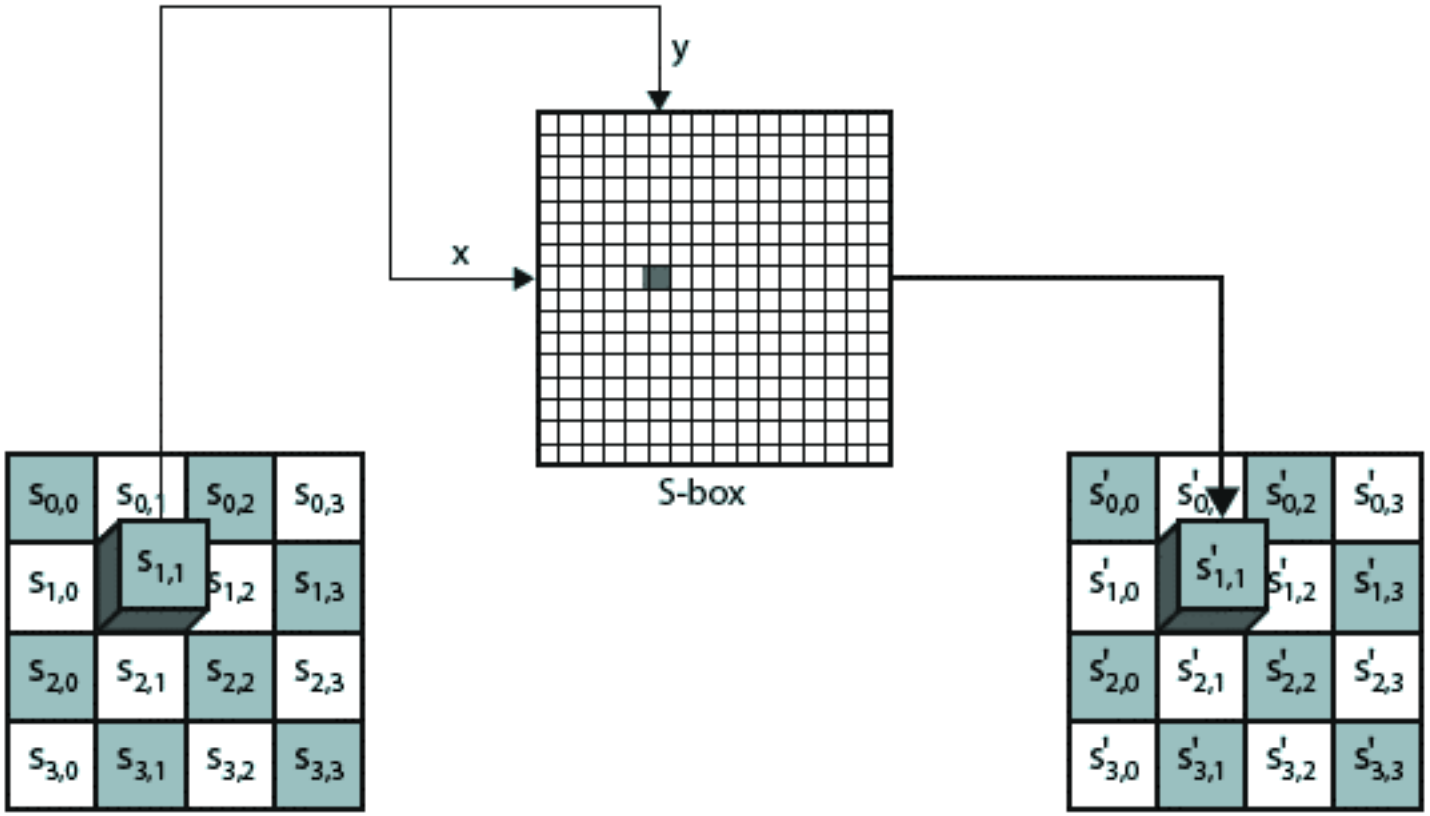
1. AddRoundKey operacija se koristi prije prve iteracije
2. Ne koristi se treća transformacija u poslednjoj iteraciji

SubBytes: Supstitucija bajtova

- Jednostavna zamjena svakog bajta.
- Koristi se jedna tabela veličine 16x16 bajtova (*S-box*) koja sadrži permutaciju svih mogućih 256-bitnih vrijednosti.
- Svaki bajt stanja (*state*) se mijenja sa bajtom iz tabele, iz reda koji je određen sa lijeva 4 bita, i kolone koja je određena sa desna 4 bita.
  - npr. bajt {95} se mijenja sa bajtom u 9. vrsti i 5. koloni.
- *S-box* se konstruiše korišćenjem definisane transformacije vrijednosti u GF(2^8).
- *S-box* je dizajniran da bude otporan na sve poznate napade.

SubBytes: Supstitucija bajtova

- SubBytes uključuje 16 nezavisnih bajt-u-bajt transformacija.



SubBytes tabela

		y														
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	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

(a) S-box

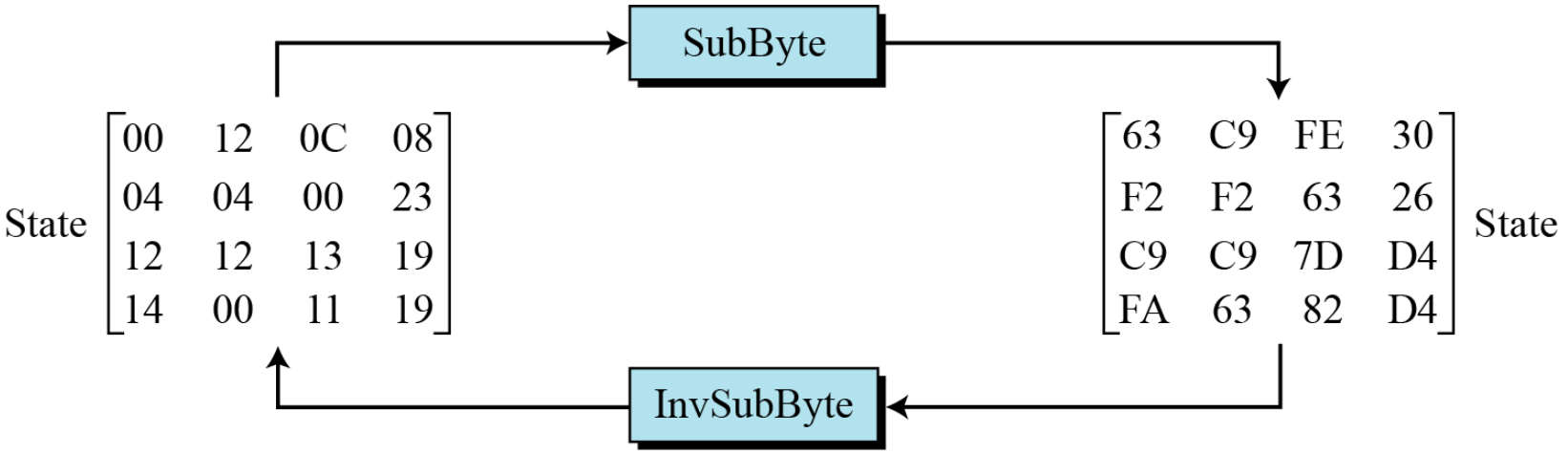
InvSubBytes tabela

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

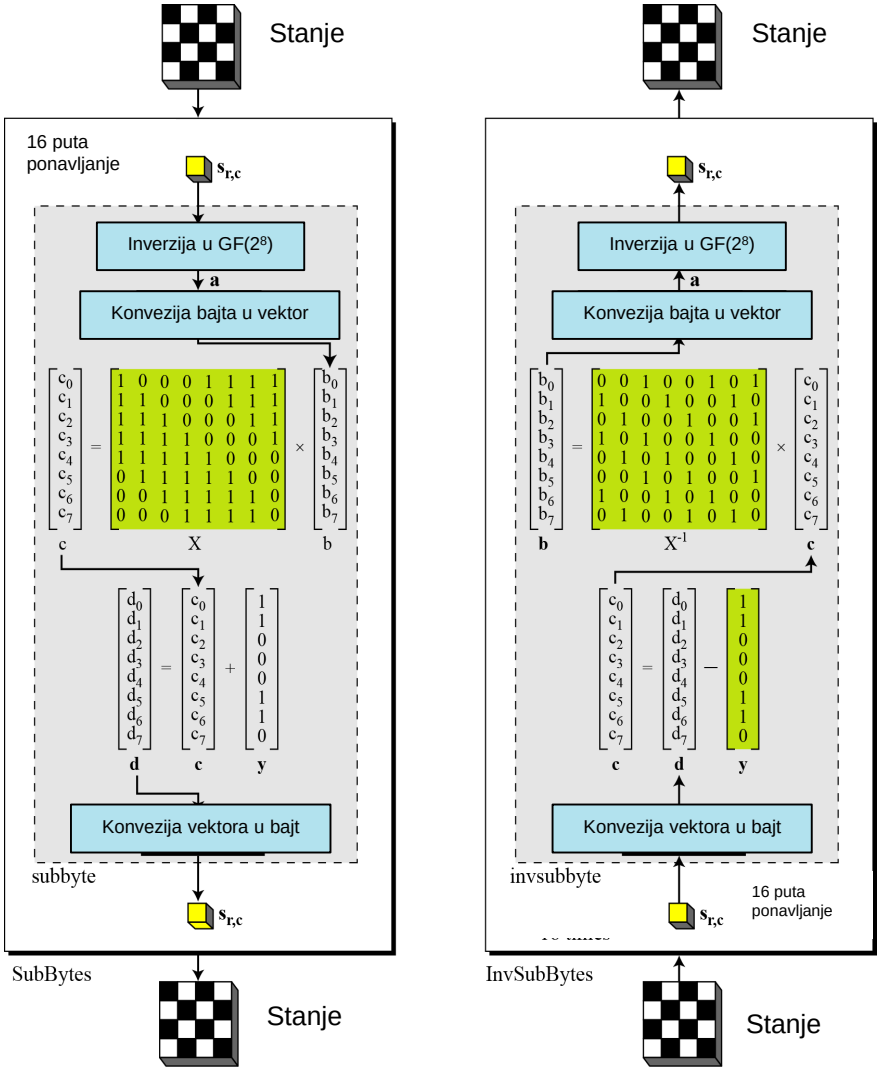
(b) Inverse S-box

Primjer SubBytes transformacije

- SubBytes i InvSubBytes su međusobno inverzne.

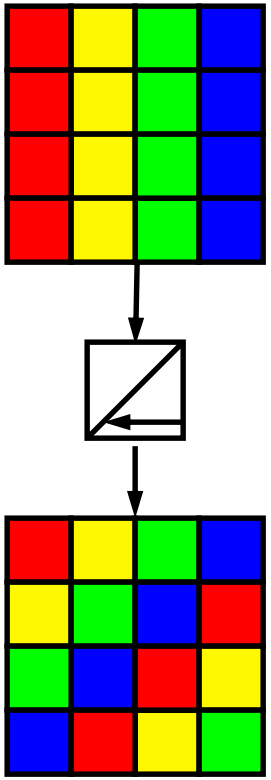


Implementacija SubBytes S-box-a

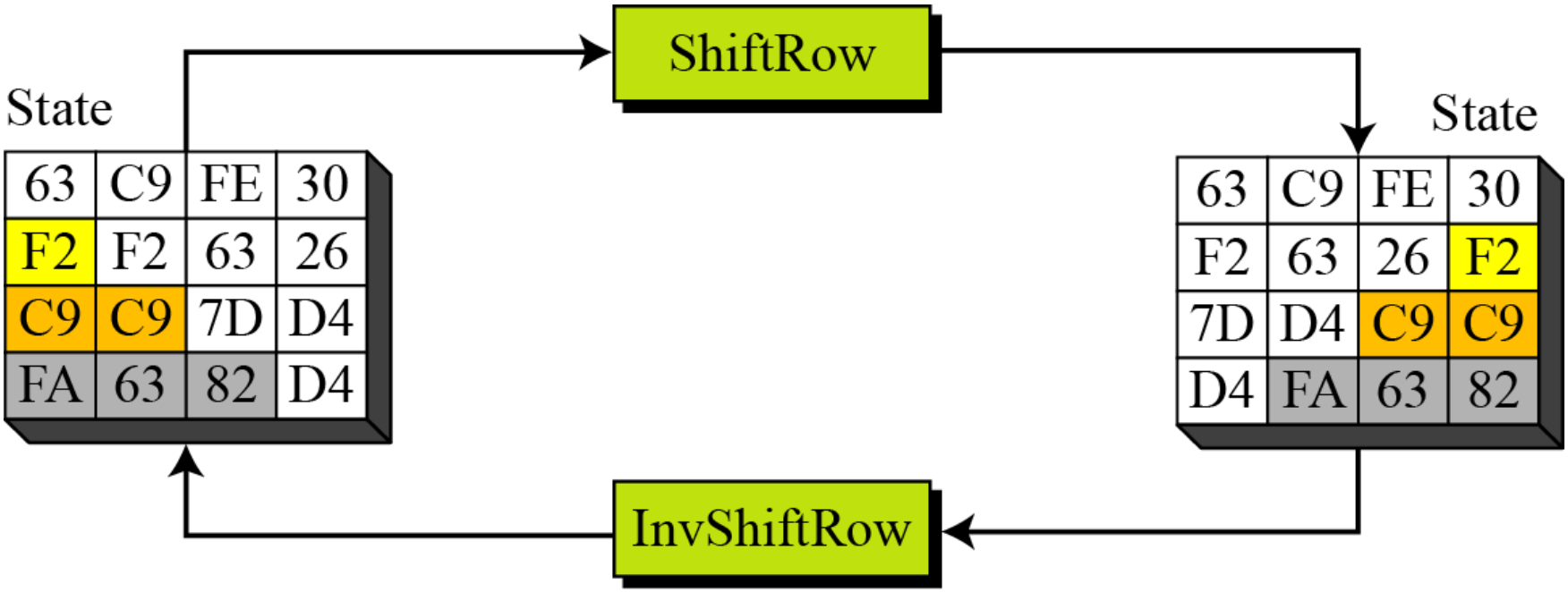


ShiftRows: Pomjeranje vrsta

- Za svaki red se radi kružno pomjeranje u lijevo
  - Prvi red se mijenja;
  - Drugi red se kružno pomjera u lijevo za 1 bajt;
  - Treći red se kružno pomjera u lijevo za 2 bajta;
  - Četvrti red se kružno pomjera ulijevo za 3 bajta.
  - Prilikom dekripcije vrši se pomjeranje u desno (InvShiftRows operacija).
- Kako se stanje obrađuje po kolonama (grupama), ovaj korak permutuje bajtove iz jedne kolone u 4 različite kolone.

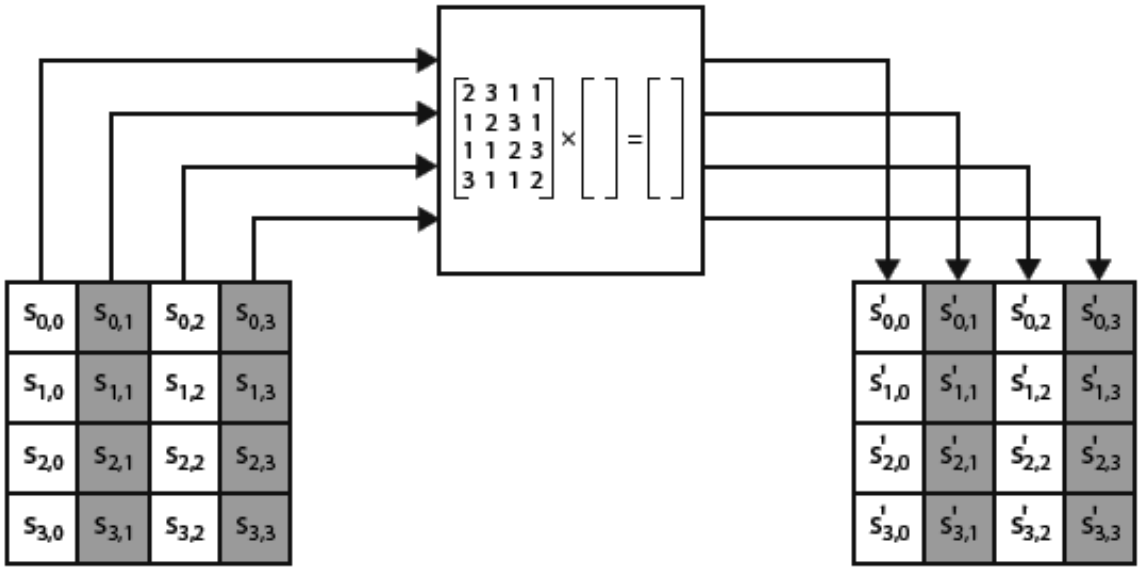


ShiftRows i InvShiftRows



MixColumns: Zamjena kolona

- Svaka kolona se obrađuje zasebno.
- Svaki bajt se zamjenjuje vrijednošću koja zavisi od sva 4 bajta kolone.
- Množenje matrica u GF(2^8), korišćenjem nesvodljivog polinoma $m(x) = x^8 + x^4 + x^3 + x + 1$.



MixColums: Primjer

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$$
\begin{aligned}
(\{02\}\cdot\{87\}) \oplus (\{03\}\cdot\{6E\}) \oplus \{46\} \oplus \{A6\} &= \{47\} \\
\{87\} \oplus (\{02\}\cdot\{6E\}) \oplus (\{03\}\cdot\{46\}) \oplus \{A6\} &= \{37\} \\
\{87\} \oplus \{6E\} \oplus (\{02\}\cdot\{46\}) \oplus (\{03\}\cdot\{A6\}) &= \{94\} \\
(\{03\}\cdot\{87\}) \oplus \{6E\} \oplus \{46\} \oplus (\{02\}\cdot\{A6\}) &= \{ED\}
\end{aligned}
$$

$$
\begin{aligned}
\{02\}\cdot\{87\} &= 0001\ 0101 \\
\{03\}\cdot\{6E\} &= 1011\ 0010 \\
\{46\} &= 0100\ 0110 \\
\{A6\} &= \underline{1010\ 0110} \\
& \quad 0100\ 0111 = \{47\}
\end{aligned}
$$

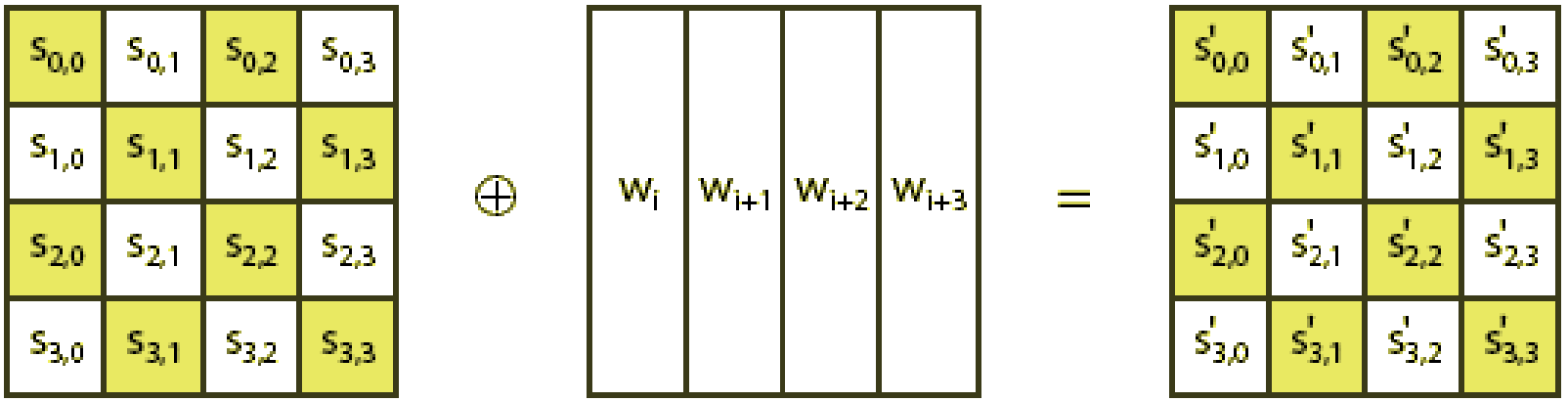
MixColumns

- Koeficijenti matrice su bazirani na linearnom maksimalnom rastojanju između kodnih riječi, što obezbjeđuje dobro miješanje bajtova svake kolone.
- U kombinaciji sa ShiftRows, MixColumns obezbjeđuje da nakon nekoliko iteracija svi biti izlaza zavise od svih bita na ulazu.
- Izbor koeficijenata ({01},{02},{03}) zasnovan je na razmatranjima koja se odnose na implementaciju.
  - Množenje sa ovim koeficijentima uključuje najviše operaciju pomjeranja (*shift*) i XOR.
- Koeficijenti kod InvMixColumns operacije su mnogo robusniji i teži za implementaciju.

AddRoundKey

- Sabiranje (XOR) matrice stanja sa 128-bitnim ključem iteracije.
- Obrada se vrši po kolonama (iako, tehnički predstavlja niz bajtovskih operacija).
- Inverzna operacija kod dekripcije je identična jer je XOR sam sebi inverzan, samo je potreban odgovarajući ključ iteracije.
- AddRoundKey operacija je dizajnirana da bude što je moguće jednostavnija.
- Kompleknost algoritma proširenja ključa i ostalih faza AES-a obezbjeđuju sigurnost.

AddRoundKey



- Primjer:

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

Stanje

⊕

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

Ključ iteracije

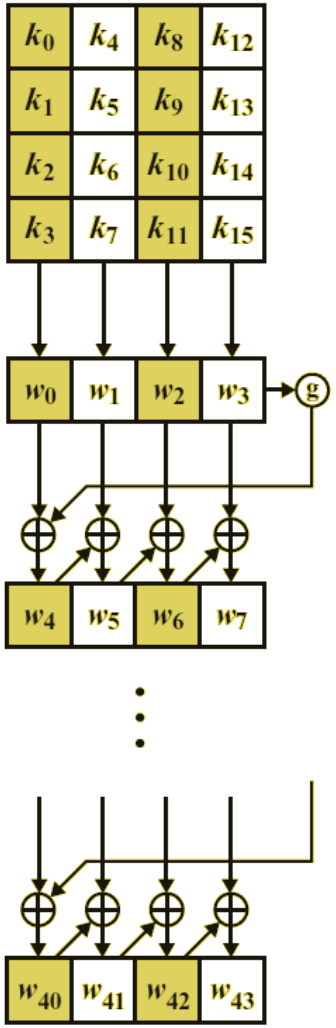
=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D6

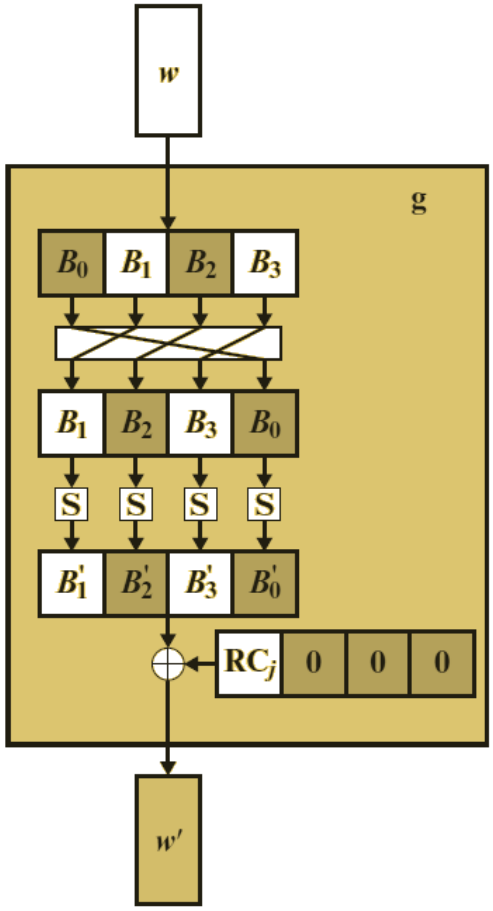
AES proširenje ključa

- Uzima 128/192/256-bitni (16/24/32-bajtni) ključ i razvija ga u niz od 44/52/60 32-bitnih riječi.
- Počinje sa kopiranjem ključa u prve četiri riječi.
- Zatim se u petlji prave riječi koje zavise od prethodne riječi ($w[i-1]$) i riječi koja je pozicionirana za 4 mjesta unazad ($w[i-4]$).
  - U tri od četiri slučaja primjenjuje se samo XOR.
  - Za riječi čija je pozicija multipl od 4, koristi se složenija funkcija (g).
- Dizajniran je da bude otporan na sve poznate napade.

AES proširenje ključa



Algoritam



Funkcija *g*

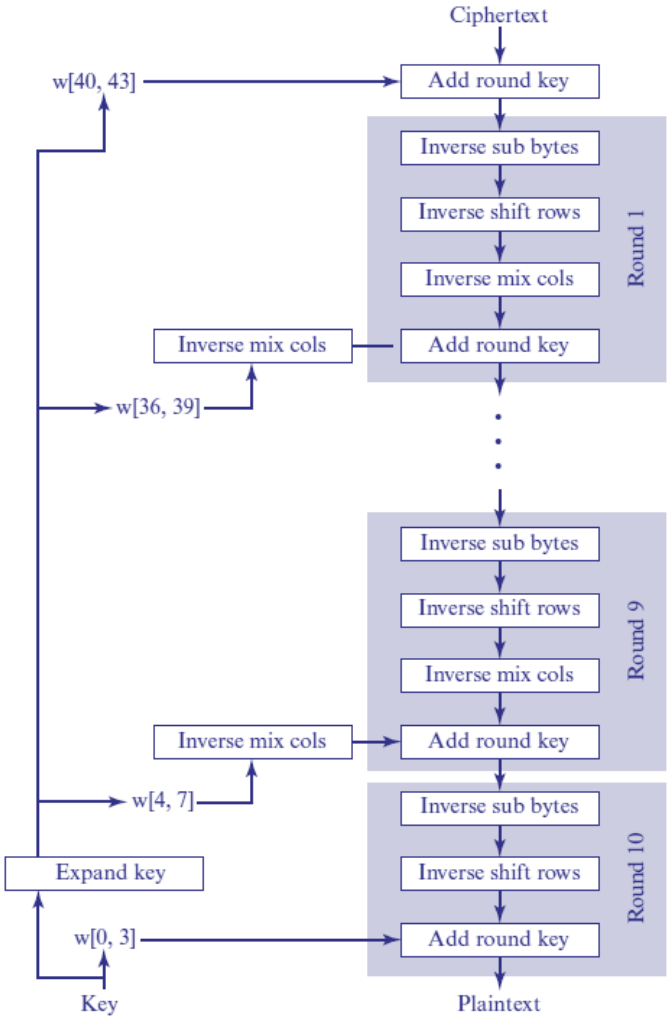
AES proširenje ključa

- Funkcija *g* se sastoji od sledećih podfunkcija:
  - Kružnog pomjeranja riječi ulijevo za jedan bajt.
  - Ulazna riječ [b0, b1, b2, b3] transformiše se u [b1, b2, b3, b0].
  - S-box funkcije koja vrši zamjenu za svaki bajt ulazne riječi.
  - XOR sa konstantom iteracije, RC[j].
- Tri prva bajta konstante iteracije su nula.
- Konstanta iteracije je različita u svakoj iteraciji, pri čemu je RC[1] = 1, RC[j] = {02} · RC[j − 1].
  - Operacija množenja definisana je u $GF(2^8)$.

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

AES dekripcija

- AES dekripcija nije identična enkripciji jer se koraci rade u inverznom redosledu.
- Struktura iteracije dekripcije: InvShiftRows, InvSubBytes, AddRoundKey, InvMixColumns.
- Može se definisati ekvivalentna inverzna šifra sa koracima kao kod enkripcije, ali uz korišćenje inveznih funkcija svakog koraka i sa različitim rasporedeom ključa.



Modovi rada blok šifratora

- Blok širfratori vrše enkripciju blokova fiksne veličine *b* bita i generišu *ciphertext* iste veličine.
- Ukoliko je poruka veća od *b* bita, onda se razbija u više blokova.
- Kada se više blokova enkriptuje istim ključem javljaju se brojni sigurnosni problemi.
- U cilju podsticanja primjene blok šifratora u raznim aplikacijama NIST je definisao pet modova rada ovih šifratora.
- Definisani modovi su predviđeni za upotrebu kod svih simetričnih blok šifratora, uključujući DES i AES.

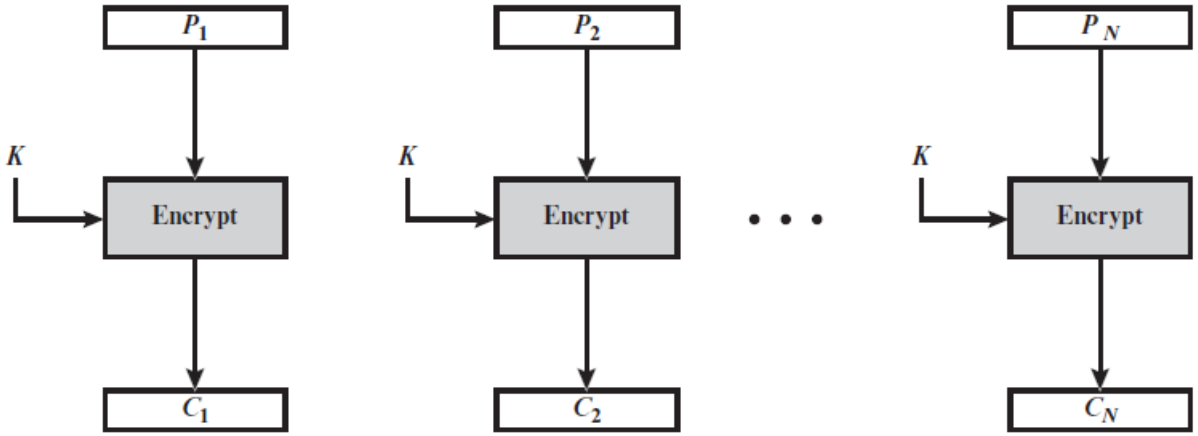
Modovi rada blok šifratora

Mod	**Tipična primjena**
Electronic Codebook (ECB)	Siguran prenos pojedinačnih vrijednosti (npr. ključeva)
Cipher Block Chaining (CBC)	Mod opšte namjene za blokovski prenos podataka; Autentifikacija
Cipher Feedback (CFB)	Mod opšte namjene za prenos podataka na nivou toka; Autentifikacija
Output Feedback (OFB)	Mod za prenos podataka na nivou toka kod kanala sa šumovima (npr. satelitska komunikacija)
Counter (CTR)	Mod opšte namjene za blokovski prenos podataka; Pogodan kada se zahtjeva velika brzina prenosa

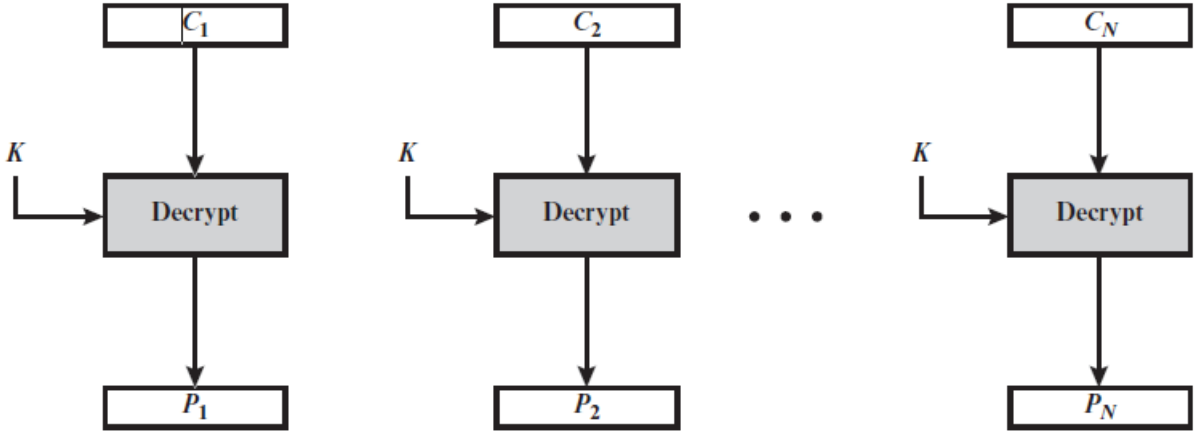
ECB mod

- *Electronic CodeBook* (ECB)
- Poruka se dijeli u blokove koji se nezavisno enkriptuju istim ključem (uz dopunu poslednjeg bloka po potrebi)
- Svaki blok *plaintext*-a se mapira u odgovarajući blok *ciphertext-a*.
  - Kao da imamo ogromnu knjigu kodova… Otuda i ime.
- Primjena: Siguran prenos poruka ne dužih od jednog bloka.
  - Na primjer, enkripcija tajnog ključa.

ECB mod



(a) Encryption



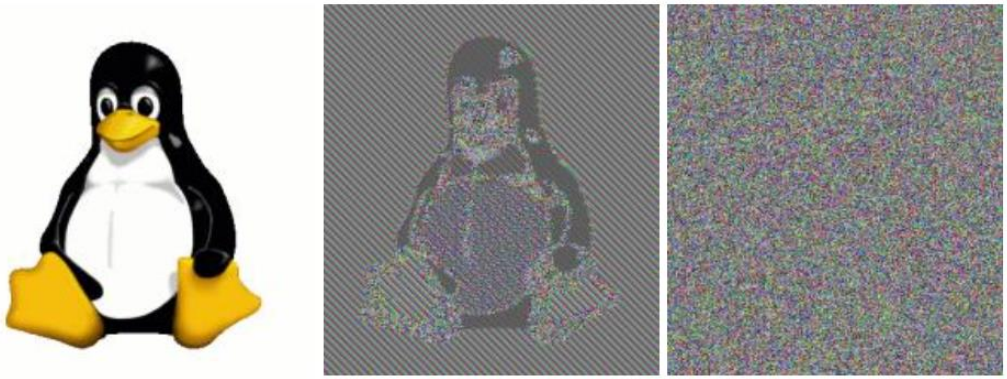
(b) Decryption

Prednosti i ograničenja ECB moda

- Ponavljanja u poruci se mogu vidjeti u enkriptovanom tekstu.
  - Isti blok nakon enkripcije uvijek daje isti rezultat.
  - Problematično kod prenosa grafičkih elemenata ili drugih poruka koje se jako malo mijenjaju.
- Ukoliko poruke imaju izreženu strukturu, napadač lako može uočiti regularnosti.
  - Na primjer, ukoliko poruka počinje sa nekim setom polja, napadač može imati na raspolaganju veći broj *plaintext-ciphertext* parova za kriptoanalizu.
  - Ukoliko se neki segmenti poruke ponavljaju sa periodom od b bita, napadač ih može lako identifikovati.

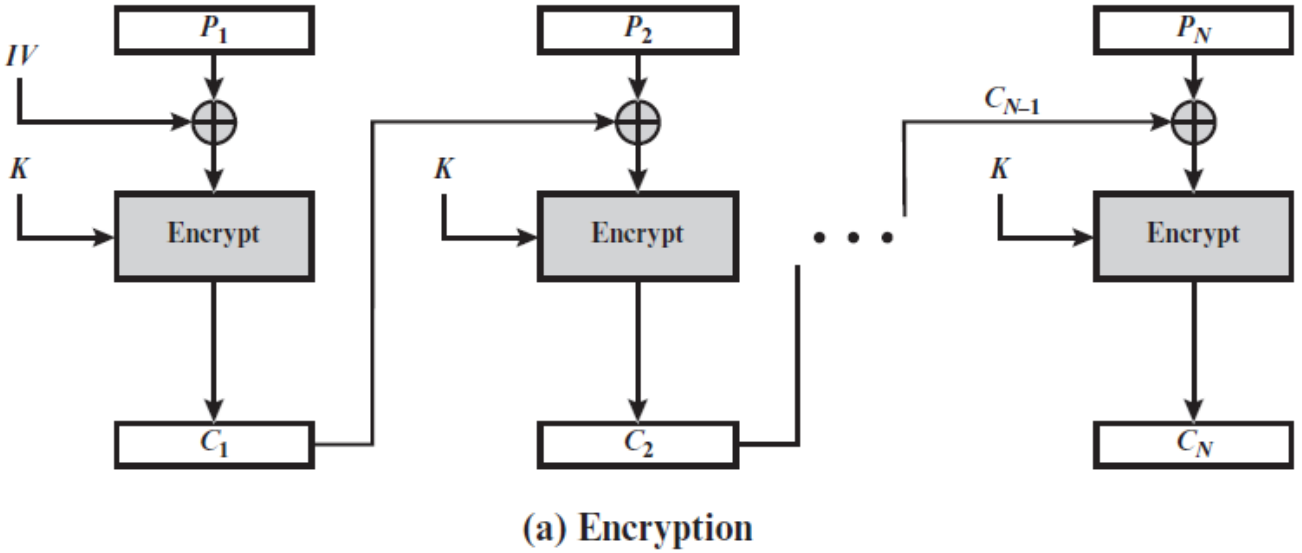
CBC mod

- *Cipher Block Chaining* (CBC)
- Ulaz algoritma enkripcije je XOR trenutnog *plaintext* bloka i prethodnog *ciphertext* bloka
- Isti ključ se koristi za svaki blok
- Lančana obrada sekvence *plaintext* blokova
- Ponovljeni *plaintext* blokovi mapiraju se u različite *ciphertext* blokove
- Ponavljajuće sekvence od *b* bita ne mogu se identifikovati

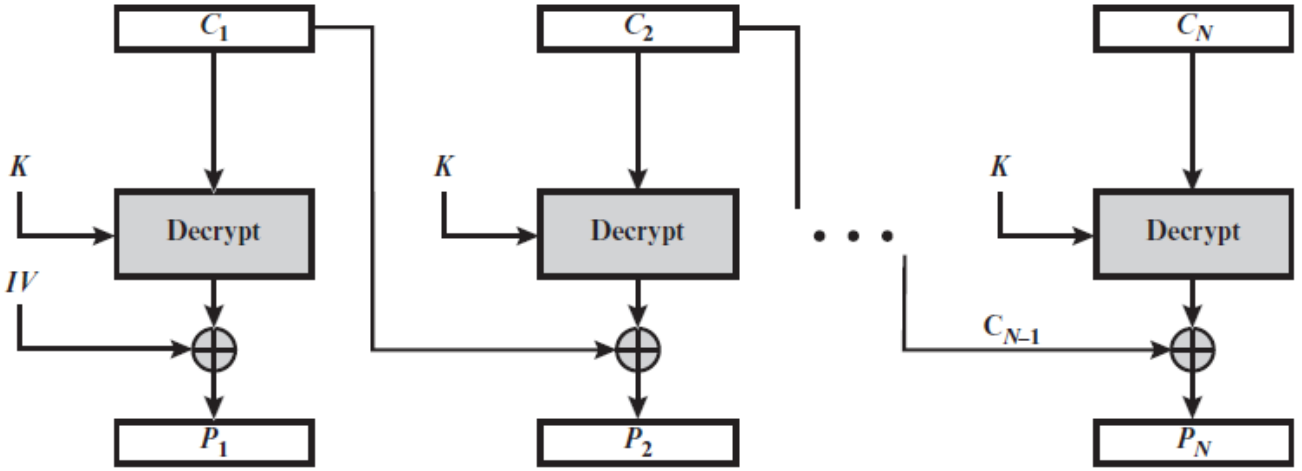


Orig. slika **ECB** **CBC**

CBC mod rada



(a) Encryption



(b) Decryption

CBC mod

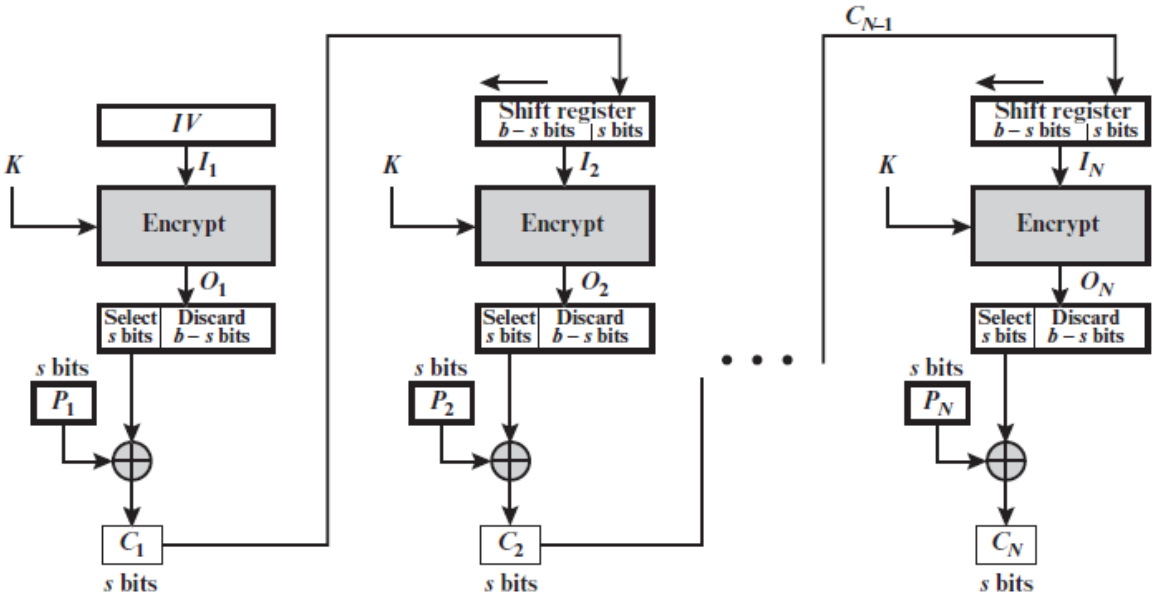
$C_1 = \mathrm{E}(K, [P_1 \oplus \mathrm{IV}])$ $C_j = \mathrm{E}(K, [P_j \oplus C_{j-1}]) j = 2, \ldots, N$	$P_1 = \mathrm{D}(K, C_1) \oplus \mathrm{IV}$ $P_j = \mathrm{D}(K, C_j) \oplus C_{j-1}\, j = 2, \ldots, N$

- Prilikom kreiranja prvog bloka ciphertext-a koristi se unaprijed definisana početna vrijednost (IV – *Initial vector*)
- Prilikom dekripcije prvog bloka vrši se XOR početne vrijednosti i izlaza algortima dekipcije.
- Početna vrijednost je iste veličine kao i šifrat.
  - Mora biti poznata pošiljaocu i primaocu, ali nepredvidiva za napadača.
- Ograničenja: Svaka promjena (greška) u poruci utiče na promjenu u svim blokovima; Nije moguća paralelizacija prilikom enkripcije.
- Primjena:
  - Šifrovanje velike količine podataka (*bulk* data)
  - Autentifikacija

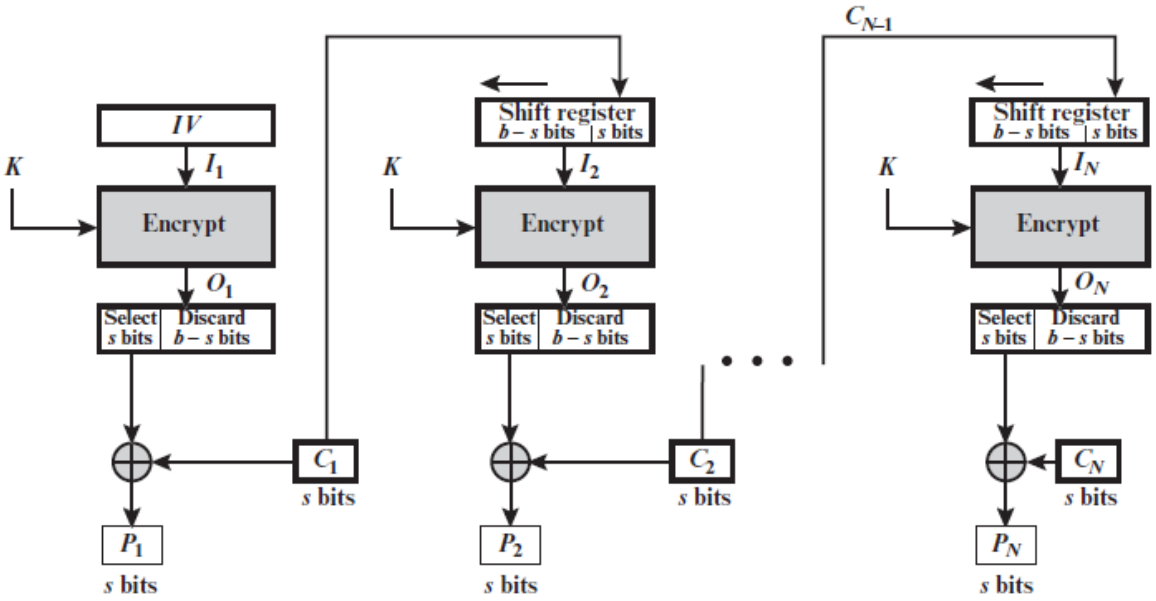
CFB mod

- *Cipher FeedBack (*CFB)
- Moguće je konvertovati blok šifrator u šifrator toka koristeći CFB, OFB ili CTR mod.
- Šifrator toka eliminiše potrebu za nadogradnjom (*padding*) poruke kako bi ona sadržala cijeli broj blokova
  - Može se izvršavati u realnom vremenu
  - Poruka se tretira kao tok bita
  - Jedinice *plaintext*-a (*s*, obično 8 bita) se uvezuju tako da svaka jedina *ciphertext*-a zavisi od svih prethodnih jedinica *plainext*-a
- Primjena: Šifrovanje toka podataka, autentifikacija.
- Ograničenja:
  - Greške se propagiraju kroz nekoliko blokova dok se ne shvati da se radi o grešci;
  - Nije moguće paralelizovati postupak enkripcije.

CFB mod



(a) Encryption



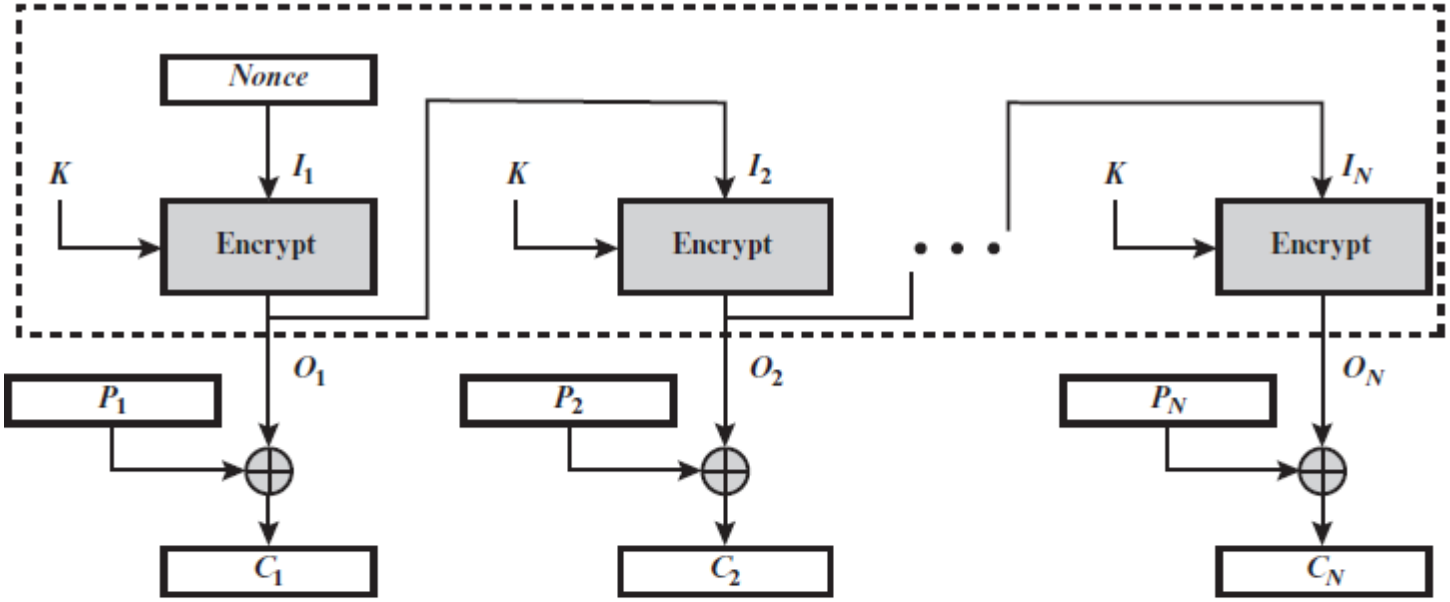
(b) Decryption

OFB mod

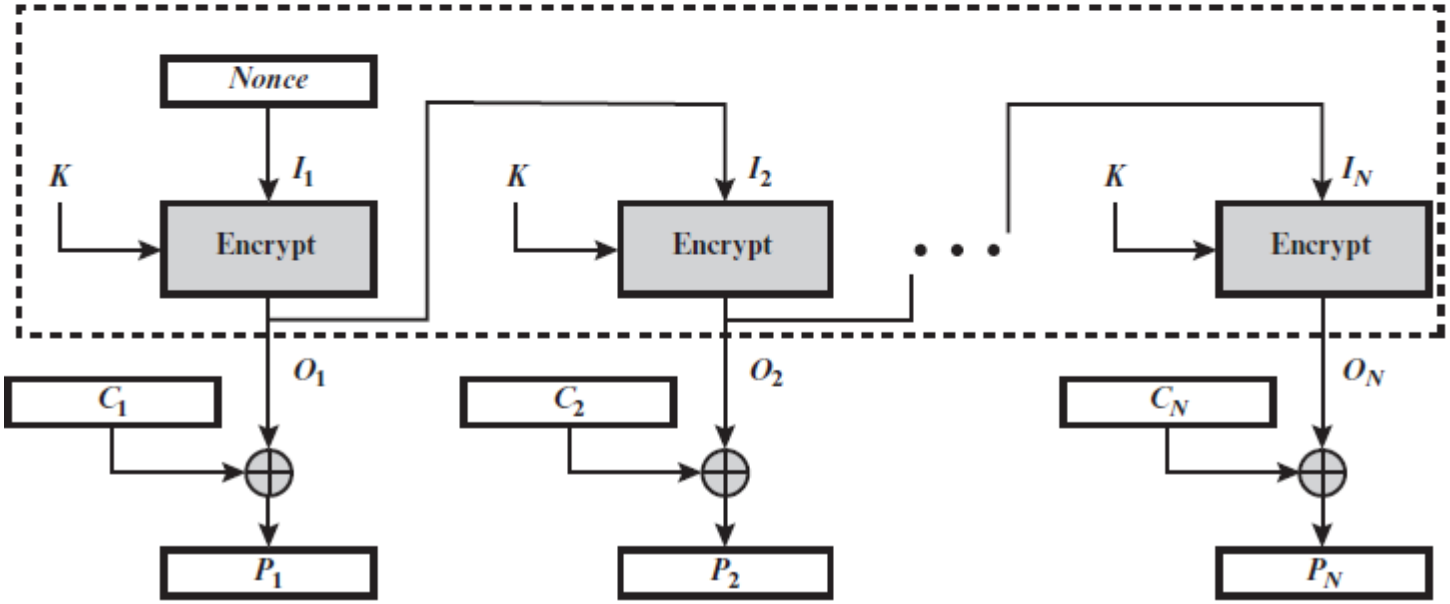
- *Output FeedBack* (OFB)
- Izlaz funkcije enkripcije postaje ulaz za enkripciju sledećeg *plaintext* bloka
- Izvršava se nad čitavim blokom *plaintext*-a i *ciphertext*-a
- Ukoliko je poslednji blok dužine $u<b$, najznačajanijih u bita poslednjeg izlaznog bloka se dovode na ulaz XOR funkcije.
- Kao i CBC i CFB, OFB zahtijeva definisanje početne vrijednosti (IV)
  - Kod OFB-a IV mora biti *nonce*, što znači da mora biti jedinstvena za svaku operaciju enkripcije.

$I_1 = Nonce$	$I_1 = Nonce$
$I_j = O_{j-1} \quad j = 2, \ldots, N$	$I_j = O_{j-1} \quad j = 2, \ldots, N$
$O_j = \mathrm{E}(K, I_j) \quad j = 1, \ldots, N$	$O_j = \mathrm{E}(K, I_j) \quad j = 1, \ldots, N$
$C_j = P_j \oplus O_j \quad j = 1, \ldots, N-1$	$P_j = C_j \oplus O_j \quad j = 1, \ldots, N-1$
$C_N^* = P_N^* \oplus \mathrm{MSB}_u(O_N)$	$P_N^* = C_N^* \oplus \mathrm{MSB}_u(O_N)$

OFB mod



(a) Encryption



(b) Decryption

Prednosti i nedostaci OFB moda

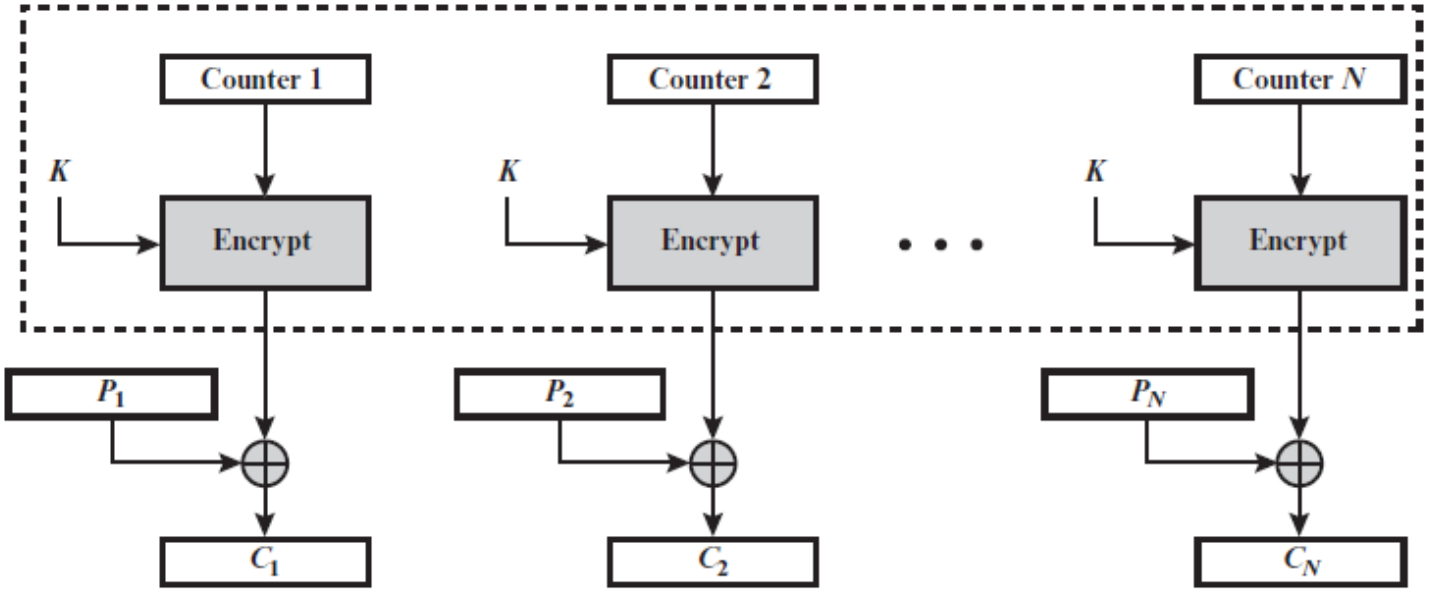
- Koristi se kada je potrebno započeti dekripciju poruke i prije nego što je cijela poruka stigla.
- Prednosti:
  - Povratna infromacija je nezavisna od poruke
  - Nema lančane propagacije grešaka
    - Na primjer, ukoliko se greška desila u prenosu *ciphertext*-a C_1, uticaće samo na *plaintext* P_1.
- Nedostaci:
  - Skloniji je napadima modifikacije toka nego CFB
    - Na primjer, inverzijom bita u *ciphertext*-u invertuje se odgovarajući bit u *plaintext*-u.
    - Na ovaj način je moguće kontrolisati dekriptovani *plaintext.*
    - Moguće je umetnuti novu poruku sa izmijenjenom *checksum*-om, a da to prijemna strana ne može da detektuje.
  - Neophodno je sinhronizovati pošiljaca i primaoca

CTR mod

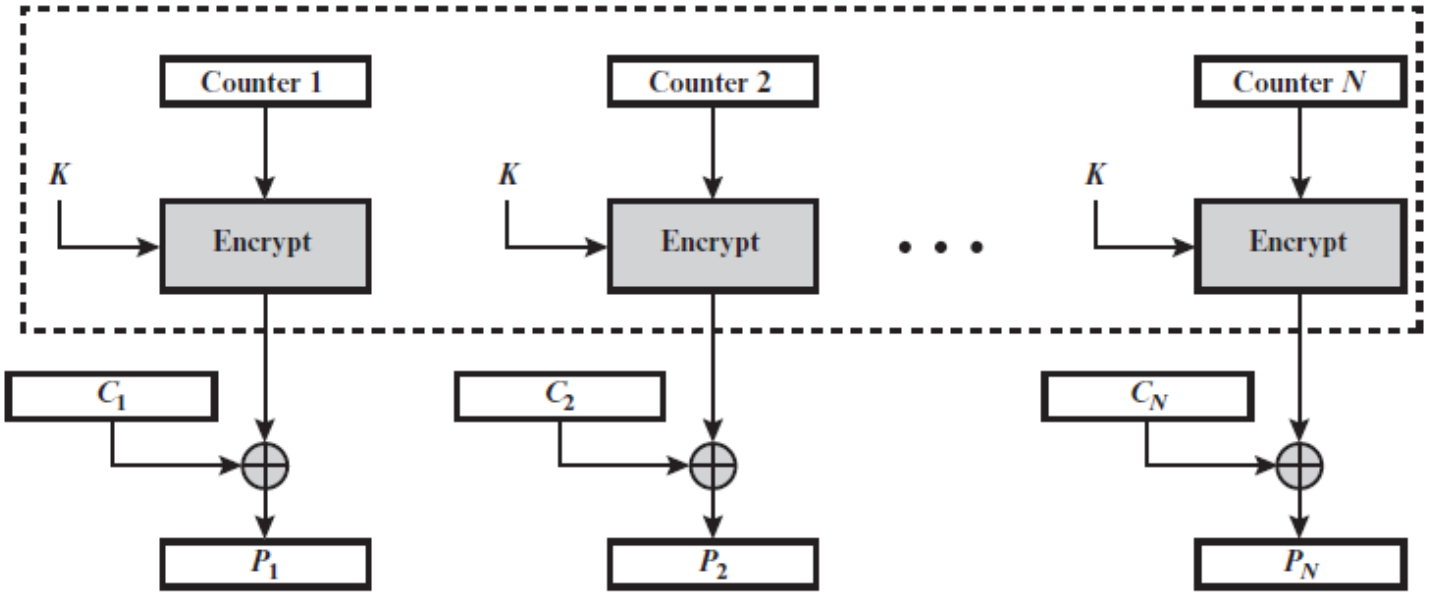
- Counter (CTR)
- Povećano interesovanje za ovom tehnikom kod sigurnosnih mehanizama za ATM i IP mreže.
- Primjena: enkripcija u mrežama velike brzine.
- Sličan OFB-u ali se enkriptuje vrijednost brojača (T), a ne povratne informacije
- Brojač je jednake dužine kao *plaintext* blok koji se enkriptuje
- Početna vrijednost (IV) mora biti *nonce*
  - Brojač mora biti različit za svaki *plaintext* blok koji se enkriptuje
- Brojač se incijalizuje na neku vrijednost a zatim uvećava za jedan (po modulu 2^b) za svaki naredni blok

$C_j = P_j \oplus \mathrm{E}(K, T_j) \qquad j = 1, \ldots, N-1$	$P_j = C_j \oplus \mathrm{E}(K, T_j) \qquad j = 1, \ldots, N-1$
$C_N^* = P_N^* \oplus \mathrm{MSB}_u[\mathrm{E}(K, T_N)]$	$P_N^* = C_N^* \oplus \mathrm{MSB}_u[\mathrm{E}(K, T_N)]$

CTR mod



(a) Encryption



(b) Decryption

CTR mod

- Ukoliko je poslednji *plaintext* blok dužine $u<b$ bita, najznačajnijih u bita izlaznog bloka se dovode na ulaz XOR funkcije dok se ostalih $b-u$ bita odbacuje
- Za razliku od ECB, CBC i CFB modova, CTR ne zahtijeva dopunjavanje poruka
- Prednosti:
  - Moguća je paralelna enkripcija (ili dekripcija) više blokova odjednom
    - Pogodan za linkove velikog kapaciteta
  - Moguće pretprocesiranje (priprema vrijednosti za XOR unaprijed)
  - Može se dekriptovati proizvoljan blok *ciphertext*-a bez prethodne dekripcije njegovih prethodnika
  - Dakazana bezbjednost – jednako siguran kao i ostali modovi
  - Jednostavnost – zahtjeva implementaciju samo algoritma enkripcije (ne i dekripcije)
- Ograničenja: Neophodno je osigurati da se ista kombinacija ključ/brojač ne ponovi prilikom enkripcije poruke

Rezime

- Aritmetika konačnih polja
- AES struktura
  - Generalna struktura
  - Detaljna struktura
- AES proširenje ključa
  - Algoritam proširenja ključa

- AES transformacione funkcije
  - SubBytes
  - ShiftRows
  - MixColumns
  - AddRoundKey
- AES dekripcija