Matematička pozadina MACsec protokola

Buklijaš, Josip

Master's thesis / Diplomski rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split, University of Split, Faculty of science / Sveučilište u Splitu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:166:380078

PRIRODOSLOVNO–MATEMATIČKI FAKULTET
SVEUČILIŠTA U SPLITU

JOSIP BUKLIJAŠ

MATEMATIČKA POZADINA MACSEC PROTOKOLA

DIPLOMSKI RAD

Split, rujan 2021.

PRIRODOSLOVNO–MATEMATIČKI FAKULTET
SVEUČILIŠTA U SPLITU

ODJEL ZA MATEMATIKU

MATEMATIČKA POZADINA MACSEC PROTOKOLA

DIPLOMSKI RAD

Student:
Josip Buklijaš

Mentorica:
doc. dr. sc. Marija Bliznac Trebješanin

Split, rujan 2021.

Uvod

Kriptografija kao grana matematike je prisutna među ljudima kroz komunikaciju, odnosno postizanje tajnosti komunikacije, stoljećima. Razni podaci, koji su među ljudima razmjenjivani nesigurnim kanalima, u krivim rukama su mogli predstavljati veliku opasnost. To je posebno dolazilo do izražaja u ratovima. Razvojem interneta, gdje se svakodnevno ogroman broj podataka razmjenjuje putem kanala koji nije siguran, kriptografija dobiva sve više na značaju te računalna sigurnost postaje jedna od najbitnijih primjena kriptografije. U ovom radu predstavit ćemo MACsec protokol koji brine o sigurnosti računalnih podataka te približiti matematičku pozadinu takvog protokola.
U prvom poglavlju navodimo definicije raznih algebarskih struktura te teorema pomoću kojih možemo razumjeti kako konstruirati konačna polja. U drugom poglavlju pozornost obraćamo na kriptosustave, s posebnim naglaskom na AES. Opisujemo blokovne i protočne šifre, kao i Galoisov način brojača koji primjenjujemo u MACsec protokolu. Opis samog MACsec protokola dan je u posljednjem poglavlju.

Sadržaj

Poglavlje 1

Matematičke predispozicije

Da bismo mogli definirati AES kriptosustav i opisati djelovanje tog kriptosustava koji se koristi kod MACsec protokola, ključno je definirati Galoisovo polje (eng. *Galois Field*), koje se često zove i konačno polje, čijom aritmetikom se koristimo u operacijama AES kriptosustava. A kako bismo definirali algebarsku strukturu polja, pa onda i konačnog polja, moramo najprije definirati i opisati jednostavnije algebarske strukture.

1.1 Algebarske strukture

U ovom potpoglavlju definirat ćemo osnovne algebarske strukture poput grupa, prstena te polja. Također, proći ćemo i kroz osnovne primjere tih struktura.

1.1.1 Grupe

Najprije dajemo definiciju strukture koju nazivamo grupa.

Definicija 1.1 *Za neprazan skup G s binarnom operacijom $\circ : G \times G \to G$ kažemo da je grupa, ako vrijedi:*

1. *Operacija* $\circ$ *ima svojstvo asocijativnosti, odnosno vrijedi* $(a \circ b) \circ c = a \circ (b \circ c),\ \forall a, b, c \in G$.
2. *Postoji element* $e \in G$ *takav da je* $e \circ a = a \circ e = e,\ \forall a \in G$. *Element* e *nazivamo neutralni element ili jedinica u* G.
3. *Za svaki* $a \in G$, *postoji* $b \in G$ *takav da je* $a \circ b = b \circ a = e$. *Element* b *zovemo inverzni element od* a, *te ga često označavamo s* a^{-1}.
4. *Ako vrijedi* $a \circ b = b \circ a$ *za svaki* $a, b \in G$, *kažemo da je grupa* G *komutativna ili Abelova.*

Grupa $(G, \circ)$ se često označava samo s G, ako nam je iz konteksta jasno o kojoj se binarnoj operaciji radi. Kardinalni broj skupa G naziva se red grupe G i označava s $|G|$. Ako je $|G|$ konačan, onda kažemo da je G konačna grupa, a u suprotnom da je beskonačna grupa. Ako je operacija $\circ$ zbrajanje, grupa se zove aditivna grupa i označavamo je $(G, +)$, a ako je operacija $\circ$ množenje, onda ju nazivamo multiplikativna grupa te ju obično označavamo $(G, \cdot)$. Najčešće se susrećemo sa sljedećim primjerima grupa.

Primjer 1.2 *Skupovi brojeva* $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ *i* $\mathbb{C}$ *zajedno s operacijom zbrajanja, pri čemu je* $e = 0$ *i* $a^{-1} = -a$, *te skupovi* $\mathbb{Q}^* = \mathbb{Q} \setminus \{0\}$, $\mathbb{R}^* = \mathbb{R} \setminus \{0\}$ *i* $\mathbb{C}^* = \mathbb{C} \setminus \{0\}$ *s operacijom množenja, pri čemu je* $e = 1$ *i* $a^{-1} = 1/a$ *su grupe.*

1.1.2 Prsteni

Ako u Abelovoj grupi $(G, +)$ možemo definirati jednu dodatnu operaciju koja će također morati zadovoljavati određena svojstva (koja ćemo definirati posebno), dolazimo do nove algebarske strukture koju nazivamo **prsten**.

Definicija 1.3 *Prsten je neprazan skup* R *s dvije binarne operacije* $+$ *i* $\cdot$, *u oznaci* $(R, +, \cdot)$, *koje nazivamo zbrajanje i množenje i koje zadovoljavaju*

sljedeća svojstva:

1. *(R,+) je Abelova grupa.*
2. *Operacija množenja $\cdot$ je asocijativna, odnosno vrijedi:*

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c, \ \forall a, b, c \in R.$$

3. *Operacija množenja je distributivna prema zbrajanju slijeva i zdesna:*

$$a \cdot (b + c) = a \cdot b + a \cdot c, \ (a + b) \cdot c = a \cdot c + b \cdot c, \ \forall a, b, c \in R.$$

Ako nam je jasno o kojim se operacijama radi, prsten $(R, +, \cdot)$ možemo kraće označavati s R.
Neutralni element u Abelovoj grupi $(R, +)$ nazivamo nulom te označavamo s 0. Ako je množenje u prstenu R komutativno, onda za taj prsten kažemo da je komutativni prsten. Ukoliko postoji element $1 \in R$ za kojeg vrijedi $1 \cdot a = a \cdot 1 = a, \ \forall a \in R$, za prsten R kažemo da je prsten s jedinicom. Navedimo neke primjere prstena:

Primjer 1.4 *Standardni primjeri prstena s jedinicom su $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ i $\mathbb{C}$ sa standardnim operacijama zbrajanja i množenja.*

Osim ovih standardnih primjera, navedimo još jedan primjer koji će nam biti zanimljiv i čija ćemo svojstva detaljnije opisati u ovom poglavlju.

Primjer 1.5 *S $\mathbb{R}[x]$ označimo skup polinoma s realnim koeficijentima*

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n, \ a_n \in \mathbb{R}, \ a_n \neq 0.$$

Taj skup, zajedno sa standardnim operacijama zbrajanja i množenja polinoma, čini prsten. Nula u prstenu $\mathbb{R}[x]$ je nul polinom $\mathbf{0}(x) = 0$, dok je jedinica konstantni polinom $\mathbf{1}(x) = 1$.

Definirajmo različite vrste prstena, s obzirom na svojstva koje množenje ima.

Definicija 1.6 *Neka je R prsten s jedinicom. Ukoliko svaki element $a \in R,\ a \neq 0$, ima inverz $a^{-1} \in R$ s obzirom na operaciju množenja, onda R nazivamo* ***prsten s dijeljenjem****. Ako je pritom i množenje u R komutativno, onda R nazivamo* ***polje****.*

O poljima te konačnim poljima, koja su nam od posebnog interesa, ćemo više u idućem potpoglavlju.

Definicija 1.7 *Prsten R nazivamo* ***integralna domena*** *ako $xy = 0,\ x, y \in R$, implicira $x = 0$ ili $y = 0$.*

Primjer integralne domene je prsten cijelih brojeva $\mathbb{Z}$. Svaki prsten s dijeljenjem je integralna domena, pa je posebno i svako polje integralna domena. Iz Definicije 1.7 slijedi da za $x, y \in R \setminus \{0\}$, gdje je R integralna domena, slijedi i $xy \neq 0$. Sljedeći teorem govori o odnosu integralne domene i polja, te o uvjetima koje integralna domena mora zadovoljavati da bi bila polje.

Teorem 1.8 *Neka je R komutativna integralna domena s jedinicom. Ako je R konačan skup, onda je R polje.*

Dokaz. Za fiksni $a \in R,\ a \neq 0$, definirajmo preslikavanje $\phi : R \to R,\ \phi(x) = ax$. Želimo pokazati da element $a \in R,\ a \neq 0$, ima multiplikativni inverz, odnosno surjektivnost funkcije ϕ, tj. da za $1 \in R$ postoji jedinstveni $b \in R$ takav da je $\phi(b) = 1$, što je zapravo $ba = ab = 1$. Time ćemo pokazati da je R polje. Da bismo dokazali surjektivnost funkcije ϕ, dovoljno će biti pokazati njenu injektivnost, jer je R konačan skup. Time će dokaz biti u potpunosti proveden. Dokažimo dakle injektivnost funkcije ϕ. Neka je $ax = ay$, odnosno $a(x - y) = 0$. Budući da je R integralna domena, a $a \neq 0$, onda slijedi $x - y = 0$. Dakle, $x = y$, odnosno ϕ je injekcija. ■

Prije nego prijeđemo na polja i posebno konačna polja, potrebne su nam

još neke definicije, od kojih je najvažnija definicija **kvocijentnog prstena**, budući da njega identificiramo s poljem, te nam služi za konstrukciju (konačnih) polja.

Definicija 1.9 *Neka je $(R, +, \cdot)$ prsten. Neka je $S \subseteq R$ neprazan podskup od R. Ako je $(S, +, \cdot)$ također prsten, onda S zovemo podprsten od R.*

Svaki prsten R ima barem 2 podprstena, koje zovemo trivijalni podprsteni, a to su $\{0\}$ i R.

Definicija 1.10 *Neprazan skup I koji je podskup prstena R nazivamo ideal ako je:*

(i) $a - b \in I,\ \forall a, b \in I,$

(ii) $ra \in I,\ ar \in I,\ \forall a \in I,\ \forall r \in R.$

Svaki ideal je podprsten, dok obratno ne vrijedi uvijek, jer drugo svojstvo iz Definicije 1.10 ne vrijedi općenito za podprsten.

Primjer 1.11 *Neka je $\mathbb{Q}$ uz standardne operacije zbrajanja i množenja prsten. Tada je $\mathbb{Z}$ podprsten od $\mathbb{Q}$, no nije ideal. Primjerice, $2 \in \mathbb{Z},\ \frac{1}{3} \in \mathbb{Q}$, no $2 \cdot \frac{1}{3} = \frac{1}{3} \cdot 2 = \frac{2}{3} \notin \mathbb{Z}$.*

Definicija 1.12 *Neka je R prsten i $S \subseteq R$. Najmanji ideal koji sadrži S nazivamo ideal generiran skupom S, te ga označavamo sa (S). Ako je S konačan skup s elementima $\{s_1, s_2, \ldots, s_n\}$, onda koristimo oznaku $(S) = (s_1, s_2, \ldots, s_n)$. Ideal koji je generiran samo s jednim elementom $a \in R$ nazivamo glavni ideal i označavamo s (a).*

Definicija 1.13 *Za ideal P u komutativnom prstenu R kažemo da je prost, ako su zadovoljena sljedeća dva svojstva:*

- *Za elemente $a, b \in R$, takve da je $a \cdot b \in P$, vrijedi $a \in P$ ili $b \in P$,*
- $P \neq R$.

Navedimo i jedan primjer prostog ideala.

Primjer 1.14 *Neka je prsten R jednak skupu cijelih brojeva $\mathbb{Z}$. Za skup $P = \{2n \mid n \in \mathbb{Z}\}$ svih parnih cijelih brojeva vrijedi da je prosti ideal prstena R. Doista, znamo da za parni broj $c = a \cdot b \in P$, $a, b \in R$, mora vrijediti da je barem jedan od elemenata a i b element P, odnosno da je paran broj.*

Definicija 1.15 *Za element p komutativnog prstena R kažemo da je prost, ako je glavni ideal (p) nenul prosti ideal.*

Kvocijentni prsten R modulo ideal I

Neka je I ideal u prstenu R. Definirajmo relaciju ekvivalencije $\sim$ na R na sljedeći način:

$$a \sim b \text{ ako i samo ako je } a - b \in I.$$

Klasa ekvivalencije elementa $a \in R$ je

$$\overline{a} = \{b \in R \mid a - b \in I\} = \{a + r \mid r \in I\} = a + I.$$

Skup svih klasa ekvivalencije $\overline{a}$ označimo s

$$R/I = \{a + I \mid a \in R\}.$$

Na skupu R/I definiramo zbrajanje i množenje na sljedeći način:

$$(a + I) + (b + I) = (a + b) + I,$$

$$(a + I) \cdot (b + I) = (a \cdot b) + I.$$

Lako se provjeri da ovako definirane operacije ne ovise o predstavniku klase ekvivalencije, te stoga tvrdimo da su dobro definirane. Nadalje, skup R/I s tako definiranim operacijama zbrajanja i množenja čini prsten. Nula u prstenu R/I je klasa ekvivalencije $\overline{0} = I$. Ukoliko prsten R ima i jedinicu $1 \in R$, onda je jedinica u prstenu R/I klasa ekvivalencije $\overline{1} = 1 + I$. Svojstva asocijativnosti i distributivnosti se nasljeđuju iz prstena R.

Definicija 1.16 *Neka je $(R, +, \cdot)$ prsten te I ideal u tom prstenu. Tada prsten $(R/I, +, \cdot)$ zovemo kvocijentni prsten R modulo I.*

Ukoliko je jasno o kojim se operacijama u kvocijentnom prstenu radi, umjesto $(R/I, +, \cdot)$, često pišemo samo R/I. Navedimo i jedan primjer kvocijentnog prstena.

Primjer 1.17 *Neka je (n) glavni ideal u prstenu $\mathbb{Z}$. Ako je $n = 0$, onda je kvocijentni prsten $\mathbb{Z}/(0) = \mathbb{Z}$. Ako je $n \neq 0$, onda je kvocijentni prsten $\mathbb{Z}/(n) = \{k + (n) \mid k \in \mathbb{Z}\}$ jednak prstenu $\mathbb{Z}_n = \{\overline{0}, \overline{1}, \ldots, \overline{n-1}\}$ sa zbrajanjem i množenjem modulo n.*

Definicija 1.18 *Neka je R prsten. Ideal I u R je maksimalni ideal ako vrijedi:*

- *$I \subsetneq R$,*
- *Ne postoji ideal J u R takav da vrijedi $I \subsetneq J \subsetneq R$.*

Teorem 1.19 *Neka je R komutativni prsten s jedinicom. Neka je I ideal u R. Sljedeće dvije tvrdnje su ekvivalentne:*

1. *I je maksimalni ideal.*
2. *Kvocijentni prsten R/I je polje.*

1.1.3 Polja

Definiciju algebarske strukture polja smo već naveli, a ovdje dajemo alternativnu definiciju u kojoj navodimo koja sve svojstva neprazni skup F treba imati da bi bio polje. Nakon definicije navodimo i standardne primjere polja.

Definicija 1.20 *Uređena trojka $(F, +, \cdot)$, gdje je F neprazan skup, $a + : F \to F$ i $\cdot : F \to F$ binarne operacije na F, naziva se polje ako vrijedi:*

1. *$(F, +)$ je aditivna komutativna grupa s neutralnim elementom 0,*
2. *$(F \setminus \{0\}, \cdot)$ je multiplikativna komutativna grupa s neutralnim elementom 1,*
3. *Vrijedi distributivnost operacije $\cdot$ prema operaciji $+$, odnosno*

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c), \ \forall a, b, c \in F.$$

Primjer 1.21 *Standardni primjeri polja su $\mathbb{C}, \mathbb{R}$ i $\mathbb{Q}$, uz operacije zbrajanja $+$ i množenja $\cdot$. Kod sva tri polja, neutralni element aditivne grupe je 0, a multiplikativne grupe 1. Svaki element a ima aditivni inverz $-a$, a svaki nenul element a ima multiplikativni inverz $1/a$.*

Uočimo da $\mathbb{Z}$ nije polje jer niti jedan element x skupa $\mathbb{Z}$ osim -1 i 1 nema multiplikativni inverz u $\mathbb{Z}$, već za njihove inverze x^{-1} vrijedi $x^{-1} = \frac{1}{x} \notin \mathbb{Z}$. Ako je skup F iz definicije konačan skup, tada kažemo da je $(F, +, \cdot)$ **konačno polje**. U kriptografiji nas gotovo uvijek samo takva polja i zanimaju, a takva polja imaju naziv i **Galoisova**[1] **polja** (eng. *Galois Field*). Broj elementa konačnog polja se naziva **red polja**. Podsjetimo se i definicije karakteristike polja, te definicije potpolja.

Definicija 1.22 *Neka je F polje. Pretpostavimo da postoji $n \in \mathbb{N}$ takav da je $n \cdot 1 = 0$, gdje su 0 i 1 neutralni elementi za zbrajanje i množenje,*

[1] nazvana po francuskom matematičaru Evaristeu Galoisu

redom. Najmanji takav prirodni broj n naziva se ***karakteristika polja*** *F. Ako takav broj ne postoji, onda kažemo da polje F ima karakteristiku nula. Karakteristiku polja F označavamo s $char(F)$.*

Definicija 1.23 *Neka je F polje. Za podskup $F' \subseteq F$ kažemo da je potpolje, ako je i samo polje s obzirom na iste operacije zbrajanja i množenja kao i u polju F, te ako se nula i jedinica iz F preslikavaju u nulu i jedinicu u F', redom.*

Lema 1.24 *Svako polje F, karakteristike p, gdje je p prost broj, sadrži $\mathbb{Z}_p$ kao potpolje.*

Dokaz. Pogledajmo podskup $\{0, 1, \ldots, p-1\}$ skupa F. Taj podskup čini p različitih elemenata skupa F, budući da je $char(F) = p$. Taj podskup zadovoljava pravila zbrajanja i množenja kao i $\mathbb{Z}_p$, te ga možemo označiti sa $\mathbb{Z}_p = \mathbb{Z}/(p)$. Stoga kad restringiramo zbrajanje i množenje s F na $\{0, 1, \ldots, p-1\}$, slijedi da je $\mathbb{Z}_p$ potpolje, koje je sadržano u F. ■

Iskažimo teorem koji govori o redu konačnog polja.

Teorem 1.25 *Ako je F konačno polje reda m, onda je m potencija prostog broja, odnosno vrijedi $m = p^n$, za pozitivni cijeli broj n i prosti broj p. Tada je p karakteristika od F.*

Dokaz. Po Lemi 1.24 F sadrži potpolje $\mathbb{Z}_p$, pa imamo inkluziju $\mathbb{Z}_p \hookrightarrow F$. U ovakvim slučajevima smatramo da je F vektorski prostor nad poljem $\mathbb{Z}_p$. Kako je F konačan onda je i konačnodimenzionalan pa ima neku bazu $\{b_1, b_2, \ldots, b_n\}$, to znači da se svaki vektor $x \in F$ može prikazati kao linearna kombinacija baze, odnosno

$$x = \alpha_1 b_1 + \alpha_2 b_2 + \cdots + \alpha_n b_n, \ \alpha_i \in \mathbb{Z}_p.$$

Što znači da postoji $|\mathbb{Z}_p|^n$ mogućih vektora u F (jer za svaki α_i imamo samo p mogućnosti), to jest $|F| = p^n$. ■

Teorem 1.26 *Neka je F polje. Tada je char(F) nula ili prost broj.*

Dokaz. Karakteristiku polja F označimo s c. Ako je $c = 0$, tvrdnja vrijedi. Pretpostavimo dakle da je $c \neq 0$. Tada je $c \cdot 1 = 0$, gdje je 1 jedinica u F. Tvrdimo da je c prost broj, čime će dokaz biti proveden. Pretpostavimo suprotno tvrdnji, dakle neka je c složen broj. Tada se c može faktorizirati kao $c = c_1 \cdot c_2$, $1 < c_1, c_2 < c$. Imamo $(c_1 \cdot c_2) \cdot 1 = 0$, a iz toga slijedi $(c_1 \cdot 1) \cdot (c_2 \cdot 1) = 0$. Budući da je F integralna domena, slijedi $c_1 \cdot 1 = 0$ ili $c_2 \cdot 1 = 0$. Sada za svaki element $x \in F$ slijedi $c_1 \cdot x = 0$ ili $c_2 \cdot x = 0$. Time smo došli do kontradikcije jer smo imali $c_1 < c$ i $c_2 < c$. Dakle, c je prost broj. ■

Iz prethodnih razmatranja slijedi da je polje beskonačnog reda ukoliko je karakteristike nula. Ako je konačnog reda, karakteristika mu je prost broj. Dakle, konačno polje sa $3, 17$ ili 29 elemenata može postojati. Također može postojati i konačno polje s 256 elemenata ($2^8 = 256$), no konačno polje s primjerice 14 elemenata ne može postojati. Razlog tomu je što 14 nije potencija prostog broja ($14 = 2 \cdot 7$).

U prethodnom potpoglavlju smo naglasili da kvocijentni prsten možemo identificirati s prstenom. U Primjeru 1.17 smo naveli kvocijentni prsten $\mathbb{Z}/(n)$ i spomenuli kako je jednak prstenu $\mathbb{Z}_n = \{\overline{0}, \overline{1}, \ldots, \overline{n-1}\}$ sa zbrajanjem i množenjem modulo n. Ako je n prost broj, taj kvocijentni prsten nam je primjer konačnog polja. Ipak pokažimo i na drugi način kako svaki element od $\mathbb{Z}_n$, ima inverz ako je n prost broj. Drugim riječima, za $a \in \mathbb{Z}_n$, postoji $b \in \mathbb{Z}_n$, tako da je $a \cdot b = 1$. Da bismo dokazali tu tvrdnju, potrebno je uvesti definiciju kongruencije, odnosno relacije *biti kongruentan modulo m.* Izreći ćemo i dokazati teorem iz kojeg izravno slijedi gore iskazana tvrdnja.

Definicija 1.27 *Neka su $a, b \in \mathbb{Z}$. Ako $m \in \mathbb{Z}, m \neq 0$, dijeli razliku $a - b$, onda kažemo da je a kongruentan b modulo m i pišemo $a \equiv b \pmod{m}$.*

Definicija 1.28 *Neka su $b, c \in \mathbb{Z}$. Cijeli broj a zovemo zajednički djelitelj od b i c ako a dijeli b i ako a dijeli c. Ako je barem jedan od brojeva b i c različit od nule, postoji samo konačan broj zajedničkih djelitelja od b i c. Najvećeg od njih zovemo najveći zajednički djelitelj od b i c, te ga označavamo s (b,c).*

Definicija 1.29 *Skup $\{x_1, \ldots, x_m\}$ nazivamo potpuni sustav ostataka modulo m, ako $(\forall y \in \mathbb{Z})$ $(\exists!\ x_i)$ za kojeg vrijedi $y \equiv x_i \pmod{m}$.*

Teorem 1.30 *Neka su $a, m \in \mathbb{N}, b \in \mathbb{Z}$. Kongruencija $ax \equiv b \pmod{m}$ ima rješenja ako i samo ako $d = (a,m)$ dijeli b. Tada kongruencija ima točno d rješenja modulo m.*

Dokaz. Označimo s d najveći zajednički djelitelj brojeva a i m. Neka kongruencija $ax \equiv b \pmod{m}$ ima rješenja. Tada postoji $y \in \mathbb{Z}$ tako da je $ax - my = b$. Vidimo da onda d dijeli b. Pretpostavimo sada suprotno, odnosno da d dijeli b. Stavimo $a' = \frac{a}{d},\ b' = \frac{b}{d},\ m' = \frac{m}{d}$. Riješimo kongruenciju $a'x \equiv b' \pmod{m'}$. Ona ima točno jedno rješenje modulo m', budući da vrijedi $(a', m') = 1$. To je zato što se svaki ostatak modulo m', pa tako i b', dobiva za točno jedan x iz potpunog sustava ostataka modulo m'. Sada, ako je x' neko rješenje kongruencije $a'x \equiv b' \pmod{m'}$, onda su sva rješenja od $ax \equiv b \pmod{m}$ dana s $x = x' + nm'$, za $n \in \mathbb{Z}$, dok su sva neekvivalentna rješenja dana s $x = x' + nm'$, za $n = 0, 1, \ldots, d - 1$. Dakle, u ovom slučaju, ako d dijeli b, kongruencija $ax \equiv b \pmod{m}$ ima točno d rješenja modulo m.

■

Iz ovog teorema slijedi da ako je p prost broj i ako a nije djeljiv s p, onda kongruencija $ax \equiv b \pmod{p}$ uvijek ima jedinstveno rješenje. To povlači

da skup ostataka $\{0, 1, \ldots, p-1\}$ pri dijeljenju s p, uz zbrajanje i množenje modulo p, čini polje koje označavamo s $\mathbb{Z}_p$ ili nekad s $\mathbb{F}_p$.

Definicija 1.31 *Neka je F polje. Podskup $K \subseteq F$ koji je i sam polje s obzirom na operacije definirane u polju F, nazivamo potpolje polja F. Tada je polje F proširenje polja K. Za $K \neq F$, kažemo da je K pravo potpolje polja F. Polje koje nema pravih potpolja, zovemo prosto polje.*

Definicija 1.32 *Neka je p prost broj. Prsten $\mathbb{Z}_p$ nazivamo* ***prosto polje*** *ili Galoisovo polje, koje označavamo s $GF(p)$. Aritmetika u $GF(p)$ je standardna aritmetika modulo p.*

Prije nego navedemo primjer jednog prostog polja reda p, naglasimo da osim što su zbrajanje i množenje u tom polju modulo p, aditivni inverz elementa a je dan s $a + (-a) \equiv 0 \pmod{p}$, a multiplikativni inverz elementa $a,\ a \neq 0$, dan s $a \cdot a^{-1} = 1$.

Primjer 1.33 *Promotrimo konačno polje $GF(5) = \{0, 1, 2, 3, 4\}$. Nula u tom polju je element 0, a jedinica je element 1. U tablicama prikažimo zbrajanje i množenje u tom 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

Tablica 1.1: Zbrajanje modulo 5 u $GF(5)$

$\cdot$	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

Tablica 1.2: Množenje modulo 5 u $GF(5)$

Navedimo aditivne, odnosno multiplikativne inverze elemenata iz $GF(5)$, koje možemo pročitati iz tablica 1.1 i 1.2:

$$-0 = 0,\ -1 = 4,\ -2 = 3,\ -3 = 2,\ -4 = 1.$$

$$0^{-1} \text{ ne postoji},\ 1^{-1} = 1,\ 2^{-1} = 3,\ 3^{-1} = 2,\ 4^{-1} = 4.$$

Navedimo još jedan vrlo bitan primjer prostog polja s dva elementa: $GF(2)$, koje je najmanje konačno polje. $GF(2) = \{0, 1\}$, a operacije zbrajanja i množenja su modulo 2. Ovo polje, $GF(2)$, nam je vrlo bitno u nastavku

$+$	0	1
0	0	1
1	1	0

Tablica 1.3: Zbrajanje modulo 2 u $GF(2)$

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

Tablica 1.4: Množenje modulo 2 u $GF(2)$

rada, jer operacije koje se koriste u algoritmu AES kriptosustava koje ćemo opisati u sljedećem poglavlju, koriste ovo zbrajanje i množenje modulo 2.

Pritom je zbrajanje realizirano logičkim operatorom XOR, odnosno operatorom *isključivo ILI*, dok je množenje modulo 2 realizirano logičkim operatorom AND, odnosno operatorom I.

1.1.4 Prsteni polinoma

Uvedimo pojam polinoma, odnosno prstena polinoma, na sljedeći način.

Definicija 1.34 *Neka je R komutativni prsten. Formalni red potencija nad R je niz*

$$\sigma = (s_0, s_1, \dots, s_i, \dots),$$

*pri čemu su $s_i \in R$ koeficijenti od σ. Skup svih formalnih redova potencija nad R označavamo s $R[[x]]$. (Formalni) **polinom** nad prstenom R je formalni red potencija $\sigma = (s_0, s_1, \dots, s_i, \dots)$ takav da postoji $n \in \mathbb{N}_0$ sa svojstvom $s_k = 0$, za $k > n$, odnosno*

$$\sigma = (s_0, s_1, \dots, s_n, 0, 0, \dots),$$

pri čemu su $s_i \in R$ koeficijenti polinoma σ. Skup svih polinoma nad R označavamo s $R[x]$.

Nulpolinom je niz $\sigma = (0, 0, \dots)$ i njega označavamo s 0.
Ako je $\sigma \neq 0$, onda $n \in \mathbb{N}_0$, takav da je $s_n \neq 0$ i $s_k = 0$, za $k > n$, nazivamo stupnjem polinoma σ i pišemo $deg\ \sigma = n$, a s_n zovemo vodećim koeficijentom polinoma σ. Ukoliko je $s_n = 1$, onda kažemo da je polinom σ normiran. Posebno se stupanj nulpolinoma definira kao $deg(0) = -\infty$. Naglasimo da niz $\sigma = (s_0, s_1, \dots, s_i, \dots)$ možemo shvatiti kao zapis sume

$$s_0 + s_1 x + s_2 x^2 + \cdots + s_i x^i + \dots.$$

Definirajmo operacije u $R[[x]]$, skupu svih formalnih redova potencija nad R. Zbrajanje je definirano na prirodan način, odnosno po koordinatama, dok je

množenje definirano tako da odgovara množenju "svaki sa svakim". Neka su $\sigma = (s_i)$ i $\tau = (t_i)$ formalni redovi. Njihova suma je definirana kao red

$$\sigma + \tau = (s_i + t_i).$$

Njihov umnožak je definiran kao red

$$\sigma \cdot \tau = (u_k),$$

gdje je $u_k = \sum_{i=0}^{k} s_i t_{k-i}$, odnosno kao

$$\begin{aligned}(s_0 + s_1 x + s_2 x^2 + \dots) \cdot (t_0 + t_1 x + t_2 x^2 + \dots) = \\ = s_0 t_0 + (s_0 t_1 + s_1 t_0)x + (s_0 t_2 + s_1 t_1 + s_2 t_0)x^2 + \dots .\end{aligned}$$

Propozicija 1.35 *$R[[x]]$ je uz gore definirane operacije komutativni prsten.*

Dokaz. Iz definicije samih operacija vidimo da su dobro definirane. Nula je nulpolinom, dok je jedinica niz $1 = (1, 0, 0, \dots)$. Aditivni inverz niza $\sigma = (s_i)$ je niz $(-s_i)$. Ostala svojstva slijede trivijalno. ■

Lema 1.36 *Neka su σ i $\tau \in R[x]$ nenul polinomi.*

1. *Vrijedi $\sigma + \tau = 0$ ili $deg(\sigma + \tau) \leq max\{deg\ \sigma, deg\ \tau\}$.*
2. *Vrijedi ili $\sigma\tau = 0$ ili $deg(\sigma\tau) \leq deg\ \sigma + deg\ \tau$.*
3. *Ako je R integralna domena, onda je $\sigma\tau \neq 0$ i*

$$deg(\sigma\tau) = deg\ \sigma + deg\ \tau.$$

Iako R nije formalno podskup od $R[x]$ ni od $R[[x]]$, prirodno ga možemo identificirati sa skupom konstantnih polinoma

$$R' = \{(r, 0, 0, \dots) \mid r \in R\}.$$

Uz takvu identifikaciju imamo sljedeće rezultate.

Korolar 1.37 *i) R i $R[x]$ su podprsteni od $R[[x]]$.*

ii) Ako je R integralna domena, onda je i $R[x]$ integralna domena.

Polinome iz $R = R'$ nazivamo konstantnim polinomima. $R[x]$ nazivamo **prsten polinoma nad** R. Za njegove elemente, odnosno polinome, ćemo koristiti oznaku $f(x)$, umjesto $f \in R[x]$, budući da je takva oznaka uobičajena.

Propozicija 1.38 *Ako je $\sigma = (s_i) \in R[x]$ polinom stupnja n, onda je*

$$\sigma = s_0 + s_1 x + s_2 x^2 + \cdots + s_n x^n,$$

gdje svaki $s \in R$ identificiramo s elementom $(s, 0, 0, \dots) \in R[x]$. Nadalje, ako je $\tau = t_0 + t_1 x + t_2 x^2 + \cdots + t_m x^m$, onda vrijedi $\sigma = \tau$ ako i samo ako je $n = m$ i $s_i = t_i,\ \forall i \in \{0, 1, \dots, n\}$.

Sljedeći teorem - Teorem o dijeljenju s ostatkom, kojeg zovemo i Euklidov algoritam, navodimo bez dokaza, a on će nam pomoći za lakše razumijevanje ireducibilnosti polinoma te faktorizacije polinoma.

Teorem 1.39 *Neka je R komutativni prsten s jedinicom i neka su $f(x), g(x) \in R[x],\ g(x) \neq 0$. Ako je vodeći koeficijent od $g(x)$ invertibilan u R, onda postoje jedinstveni polinomi $q(x), r(x) \in R[x]$ takvi da vrijedi*

$$f(x) = g(x)q(x) + r(x),\ deg(r(x)) < deg(g(x)).$$

Ireducibilnost polinoma

Definicija 1.40 *Neka je F polje. Kažemo da je polinom $f(x) \in F[x]$* ***ireducibilan nad poljem*** *F ako je:*

1. $deg(f(x)) \geq 1$,
2. $f(x) = g(x)h(x),\ g(x), h(x) \in F[x] \implies g(x) \in F$ *ili* $h(x) \in F$.

Drugi uvjet iz definicije nam zapravo govori, da jedan od polinoma $g(x), h(x)$, iz faktorizacije $f(x) = g(x)h(x)$, mora biti konstantan polinom. Na nekakav neformalni način, možemo reći da je polinom ireducibilan ako se njegovom faktorizacijom ne mogu dobiti dva nekonstantna polinoma, oba manjeg stupnja od početnog polinoma. Ako polinom nije ireducibilan nad poljem F, kažemo da je reducibilan nad tim poljem. Iz definicije imamo da je polinom prvog stupnja uvijek ireducibilan. Sama definicija "ireducibilnosti" ne može ići bez drugog dijela koji kaže "nad kojim poljem" je polinom ireducibilan. To je zato što neki polinomi mogu biti ireducibilni nad jednim poljem, a nad drugim poljem mogu biti reducibilni. Primjer takvog polinoma je $p(x) = x^2 + 1$, koji je ireducibilan nad $\mathbb{R}$, dok je nad $\mathbb{C}$ reducibilan, odnosno može se faktorizirati na ovaj način:

$$x^2 + 1 = (x - i)(x + i).$$

Navedimo kriterij za provjeru reducibilnosti te uvedimo novi pojam - korijen polinoma. Nakon toga, navest ćemo i kriterij za provjeru ireducibilnosti.
Neka je R prsten i neka je $f(x) = a_0+a_1x+a_2x^2+\cdots+a_{n-1}x^{n-1}+a_nx^n \in R[x]$. Tada za $\alpha \in R$ možemo definirati preslikavanje $f(x) \mapsto f(\alpha)$, s $R[x]$ na R, koje zovemo evaluacija.

Definicija 1.41 *Neka je R podprsten komutativnog prstena S i neka je $f(x) \in R[x]$. Za element $\alpha \in S$ ćemo reći da je korijen polinoma $f(x)$ u S ako je $f(\alpha) = 0$.*

Ovakvu smo definiciju morali izreći jer korijen polinoma ne mora nužno biti u istom prstenu u kojem su i njegovi koeficijenti. Primjer smo vidjeli kod polinoma $p(x) = x^2 + 1 \in \mathbb{Z}[x]$, kojem su korijeni $\pm i \in \mathbb{C}$.

Teorem 1.42 *Neka je F polje. Element $\alpha \in F$ je korijen polinoma $p(x) \in F[x]$ ako i samo ako polinom $x - \alpha$ dijeli $p(x)$.*

Iz prethodnog teorema izravno slijedi da je polinom $p(x) \in F[x],\ deg(p(x)) \geq 2$ reducibilan u F, ako ima korijen $\alpha \in F$. Obrat općenito ne vrijedi, odnosno ako je polinom reducibilan nad F, ne znači da ima i korijen iz F. Primjer jednog takvog polinoma nam je $p(x) = x^6 + 3x^4 + 3x^2 + 1$ koji je reducibilan nad $\mathbb{R}$, odnosno vrijedi $x^6 + 3x^4 + 3x^2 + 1 = (x^2 + 1)(x^2 + 1)(x^2 + 1)$, a $p(x)$ nema korijen iz $\mathbb{R}$. No, obrat vrijedi u slučaju nekih posebnih polinoma, o čemu govori sljedeći teorem.

Teorem 1.43 *Neka je $f(x) \in F[x]$ polinom nad poljem F, stupnja 2 ili 3. Ako je polinom $f(x)$ reducibilan nad F, onda $f(x)$ ima korijen u F.*

U sljedećem teoremu ćemo iskazati kriterij ireducibilnosti polinoma iz $\mathbb{Z}[x]$ nad poljem $\mathbb{Q}$.

Teorem 1.44 (Eisensteinov kriterij) *Neka je $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1} + a_nx^n \in \mathbb{Z}[x],\ n \geq 1$. Ako postoji prosti broj p takav da $p \mid a_k$ za $k = 0, 1, \ldots, n-1,\ p \nmid a_n$ i $p^2 \nmid a_0$, onda je $f(x)$ ireducibilan nad $\mathbb{Q}$.*

Pokažimo na jednom primjeru kako pomoću ovog teorema možemo provjeriti ireducibilnost polinoma nad $\mathbb{Q}$.

Primjer 1.45 *Neka je $f(x) = 4x^5 - 3x^4 + 12x^3 + 6 \in \mathbb{Z}[x]$. Za prosti broj p (ako postoji) mora vrijediti da dijeli $a_k,\ k = 0, 1, \ldots, 4$, a da ne dijeli a_n. Za $p = 3$, vidimo da to vrijedi, odnosno 3 dijeli koeficijente $-3, 12, 6$, a ne dijeli 4. Još je potrebno provjeriti da i $p^2 = 3^2 = 9$ ne dijeli 6, a budući da i to vrijedi, iz Eisensteinovog kriterija proizlazi da je $f(x)$ ireducibilan nad $\mathbb{Q}$.*

Konstrukcija konačnih polja

Ireducibilni polinomi su vrlo bitni u prstenu polinoma $R[x]$, te ih vrlo neformalno možemo poistovjetiti s prostim brojevima u nekom polju. Već smo pokazali da je kvocijentni prsten $\mathbb{Z}/(p)$, za prosti broj p, kojeg često označavamo

$\mathbb{Z}_p$, primjer konačnog polja. Razlog zbog kojeg vrijedi analogija između prostih brojeva i ireducibilnih polinoma je sljedeći teorem, iz kojeg slijedi da su ireducibilni polinomi nad poljem R prosti elementi prstena $R[x]$.

Teorem 1.46 *Neka je R polje. Svaki polinom $f(x) \in R[x]$ se može faktorizirati na sljedeći način:*

$$f(x) = c{f_1}^{n_1}(x){f_2}^{n_2}(x)\dots{f_k}^{n_k}(x),$$

gdje je $c \in R,\ c \neq 0,\ n_1, \dots, n_k \in \mathbb{N}$, a $f_1(x), \dots, f_k(x) \in R[x]$ ireducibilni normirani polinomi.

Konačno, sljedeći teorem nam gore opisanu "analogiju" između prostih brojeva i ireducibilnih polinoma potvrđuje, te opisuje način kako iz prstena polinoma $R[x]$ možemo dobiti polje.

Teorem 1.47 *Neka je $f(x) \in R[x]$. Kvocijentni prsten $R[x]/(f(x))$ je polje ako je polinom $f(x)$ ireducibilan nad poljem R.*

Dokaz. Neka je $f(x)$ ireducibilan polinom nad poljem R. Tvrdimo da je tada kvocijentni prsten $R[x]/(f(x))$ polje. Pokažimo da svi elementi tog kvocijentnog prstena, različiti od nule, imaju multiplikativni inverz. Neka je $r(x) + (f(x))$ proizvoljan element od $R[x]/(f(x))$, za $r(x) \in R[x]$, koji je različit od nule, odnosno za kojeg vrijedi $r(x) \notin (f(x))$. To znači da $f(x)$ ne dijeli polinom $r(x)$. Kako bismo odredili inverz, potrebno je odrediti najveći zajednički djelitelj polinoma $r(x)$ i $f(x)$. Kako $(r(x), f(x)) \mid f(x)$, a $f(x)$ je ireducibilan, slijedi $(r(x), f(x)) = k \cdot f(x)$ ili $(r(x), f(x)) = k$, za $k \in R,\ k \neq 0$. Pretpostavimo $(r(x), f(x)) = k \cdot f(x)$. Tada $k \cdot f(x) \mid r(x)$, odnosno $k \cdot f(x) \cdot b(x) = r(x)$ za neki $b(x)$. No onda imamo $f(x) \cdot (k \cdot b(x)) = r(x)$, odnosno $f(x)$ dijeli $r(x)$, što je kontradikcija. Dakle, mora

vrijediti $(r(x), f(x)) = k$, za $k \in R$. Iz toga dalje imamo da postoje polinomi $m(x), n(x)$, za koje vrijedi

$$r(x) \cdot m(x) + f(x) \cdot n(x) = k.$$

Dalje imamo,

$$\begin{aligned} r(x) \cdot \left(\frac{1}{k} \cdot m(x)\right) + f(x) \cdot \left(\frac{1}{k} \cdot n(x)\right) &= 1 \\ r(x) \cdot \left(\frac{1}{k} \cdot m(x)\right) + f(x) \cdot \left(\frac{1}{k} \cdot n(x)\right) + (f(x)) &= 1 + (f(x)) \\ r(x) \cdot \left(\frac{1}{k} \cdot m(x)\right) + (f(x)) &= 1 + (f(x)) \\ [r(x) + (f(x))] \cdot \left[\left(\frac{1}{k} \cdot m(x)\right) + (f(x))\right] &= 1 + (f(x)). \end{aligned}$$

Dakle, multiplikativni inverz proizvoljnog elementa $r(x)+(f(x)) \in R[x]/(f(x))$ je $\left(\frac{1}{k} \cdot m(x)\right) + (f(x))$. ■

Sljedećim primjerom ćemo prikazati primjenu prethodnog teorema. Nakon što dobijemo polje, provjerit ćemo dodatno i aksiome polja izravno iz tablice zbrajanja te množenja.

Primjer 1.48 *Neka je $\mathbb{Z}_2[x]$ prsten polinoma nad poljem $GF(2) = \mathbb{Z}_2$. Polinom*

$$p(x) = x^2 + x + 1 \in \mathbb{Z}_2[x]$$

je ireducibilan nad poljem $GF(2)$, jer nema korijena. Naime $f(0) = f(1) = 1 \neq 0$. Ideal $(x^2 + x + 1)$ je generiran polinomom $p(x)$. Kvocijentni prsten

$$\mathbb{Z}_2[x]/(x^2+x+1) = \{ax+b+(x^2+x+1), a, b \in GF(2)\} = \{ax+b, a, b \in GF(2)\}$$

ima 4 elementa:

$$\{0, 1, x, x + 1\}.$$

Elemente tog kvocijentnog prstena možemo shvatiti i kao ostatke dijeljenja polinoma iz $\mathbb{Z}_2[x]$ polinomom (idealom) $p(x)$. Stupanj takvih polinoma koji

su ostatak pri dijeljenju mora biti manji od stupnja polinoma $p(x)$, odnosno može biti najviše 1*. Prethodni teorem nam tvrdi da oni čine polje uz operacije zbrajanja i množenja modulo 2. Prikazat ćemo i tablično te dvije operacije, pa ćemo se i iz tablica uvjeriti da je navedeni skup uistinu polje. Između*

+	0	1	x	x+1
0	0	1	x	x+1
1	1	0	x+1	x
x	x	x+1	0	1
x+1	x+1	x	1	0

Tablica 1.5: Operacija zbrajanja u $\mathbb{Z}_2[x]/(x^2+x+1)$

·	0	1	x	x+1
0	0	0	0	0
1	0	1	x	x+1
x	0	x	x+1	1
x+1	0	x+1	1	x

Tablica 1.6: Operacija množenja u $\mathbb{Z}_2[x]/(x^2+x+1)$

ostalih očitih svojstava polja koja vrijede, vidimo da je jedinica u odnosu na zbrajanje element 0*, a u odnosu na množenje* 1*, a komutativnost nam je vidljiva iz simetričnosti tablica s obzirom na glavnu dijagonalu.*

1.2 $GF(2^n)$

Polja koja su nam od interesa u algoritmu AES kriptosustava su konačna polja s 256 elemenata. Njih označavamo s $GF(2^8)$. Polja s upravo toliko elemenata su izabrana jer se svaki element može prikazati pomoću jednog bajta

(odnosno 8 bitova). No $256 = 2^8$ nije prost broj, pa prsten $\mathbb{Z}_{256}$ nije polje. Dakle, zbrajanje i množenje u $GF(2^8) = GF(256)$ moramo definirati na neki drugi način. Općenito, takva polja $GF(2^n),\ n > 1$ zovemo **proširenja polja** (eng. *extension fields*). Elementi polja $GF(2^n)$ nisu predstavljeni cijelim brojevima, već polinomima s koeficijentima iz polja $GF(2)$. Takvi polinomi su najviše stupnja $n-1$, tako da je svaki element polja $GF(2^n)$ jedinstveno određen kombinacijom točno n koeficijenata iz polja $GF(2)$. Općeniti element $A(x)$, polja $GF(2^8)$, ima oblik:

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

Primijetimo da je u polju $GF(2^8)$ točno $2^8 = 256$ polinoma. Polinom $A(x)$ možemo prikazati u formi jednog bajta, odnosno 8 bitova na sljedeći način:

$$A = a_7a_6 \ldots a_1a_0.$$

1.2.1 Zbrajanje i oduzimanje u $GF(2^n)$

Zbrajanje i oduzimanje polinoma, odnosno elemenata polja $GF(2^n)$, je vrlo jednostavno, te se uz odgovarajuće potencije varijable x koeficijenti zbrajaju modulo 2, odnosno zbrajaju se u polju $GF(2)$.
Formalnije, neka su $A(x), B(x) \in GF(2^n)$. Njihov zbroj definiran je s:

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

Njihova razlika definirana je s:

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

Primijetimo da su zbrajanje i oduzimanje modulo 2 zapravo iste operacije, jer je 1 sam sebi aditivni inverz u $GF(2)$, pa je $a + 1 \cdot b = a - 1 \cdot b,\ a, b \in GF(2)$.

Pogledajmo jedan primjer zbrajanja dvaju polinoma iz $GF(2^8)$. Rezultat zbrajanja nam je ujedno i rezultat oduzimanja ta dva polinoma.

Primjer 1.49 $A(x) = x^7 + x^5 + x^2 + 1,\ B(x) = x^6 + x^5 + x^3 + x^2 + x + 1.$
$C(x) = A(x) + B(x) = x^7 + x^6 + x^3 + x.$

1.2.2 Množenje u $GF(2^n)$

Množenjem dvaju polinoma iz polja $GF(2^n)$ na standardni način, možemo dobiti polinom stupnja većeg od n, te takav polinom ne bi bio element polja $GF(2^n)$. Takav produkt bi trebao biti polinom stupnja $n-1$ ili manjeg. Stoga nakon množenja, rezultat moramo dodatno podijeliti ireducibilnim polinomom stupnja n, čiji su koeficijenti iz $GF(2)$, te ostatak pri takvom postupku uzimamo kao "konačni" rezultat. Takav postupak kraće zovemo množenje modulo ireducibilni polinom. To sebi možemo približiti tako da zamislimo da tražimo predstavnika klase u kvocijentnom prstenu modulo ideal generiran tim ireducibilnim polinomom. Formalno, množenje u $GF(2^n)$ definiramo na sljedeći način:

Definicija 1.50 *Neka su* $A(x), B(x) \in GF(2^n)$. *Neka je*

$$P(x) = \sum_{i=0}^{n} p_i x^i,\ p_i \in GF(2),$$

ireducibilni polinom. Umnožak $A(x) \cdot B(x)$ *je jednak:*

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

Dakle, svako polje $GF(2^n)$ je definirano kao kvocijentni prsten modulo ireducibilni polinom stupnja n (odnosno modulo ideal generiran tim polinomom). Pokažimo na jednom primjeru kako množimo dva polinoma iz polja koje ćemo u nastavku rada često spominjati, $GF(2^8)$. Kad to polje koristimo kod algoritma AES kriptosustava, množenje se obavlja modulo ireducibilni polinom

$f(x) = x^8 + x^4 + x^3 + x + 1$. Osim tog polinoma, mogli smo iskoristiti i druge polinome za definirati polje $GF(2^8)$, poput $a(x) = x^8 + x^4 + x^3 + x^2 + 1$ ili $b(x) = x^8 + x^5 + x^3 + x + 1$.

Primjer 1.51 *Neka su* $A(x) = x^7 + x^5 + x^4 + 1,\ B(x) = x^3 + x \in GF(2^8)$. *Neka je* $f(x) = x^8 + x^4 + x^3 + x + 1$ *ireducibilni polinom. Pomnožimo li samo* $A(x) \cdot B(x)$, *dobit ćemo polinom*

$$\begin{aligned}(x^7 + x^5 + x^4 + 1) \cdot (x^3 + x) &= x^{10} + x^8 + x^8 + x^7 + x^6 + x^5 + x^3 + x \\ &= x^{10} + (1+1)x^8 + x^7 + x^6 + x^5 + x^3 + x \\ &= x^{10} + x^7 + x^6 + x^5 + x^3 + x.\end{aligned}$$

Sada taj polinom podijelimo s $f(x)$ *te ostatak pri tom dijeljenju uzimamo kao konačan rezultat operacije* $A(x) \cdot B(x)$.

$$\begin{aligned}(x^{10} + x^7 + x^6 + x^5 + x^3 + x) : (x^8 + x^4 + x^3 + x + 1) \\ &= x^2 + \frac{x^7 - x^2 + x}{x^8 + x^4 + x^3 + x + 1} \\ &= x^2 + \frac{x^7 + x^2 + x}{x^8 + x^4 + x^3 + x + 1}.\end{aligned}$$

Konačno, $A(x) \cdot B(x) = x^7 + x^2 + x$.

1.2.3 Inverzi u $GF(2^n)$

Definicija 1.52 *Za konačno polje* $GF(2^n)$ *i njemu odgovarajući ireducibilni polinom* $P(x)$, *inverzni element od* $A \in GF(2^n)$, *u oznaci* A^{-1}, *je definiran s:*

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

U implementaciji manjih polja, kao što je primjerice $GF(2^8)$, često se koristi tablica s unaprijed izračunatim inverzima za sve elemente danog polja. Prikazat ćemo tablicu svih inverza elemenata polja $GF(2^8)$ (zapisanih dvjema

heksadecimalnim znamenkama), koje koristimo kod algoritma AES kriptosustava (naravno modulo $P(x) = x^8 + x^4 + x^3 + x + 1$). Iz Tablice 1.7 lako

	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	C3	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

Tablica 1.7: Multiplikativni inverzi u $GF(2^8)$ kod AES algoritma

čitamo inverze pojedinih elemenata, odnosno polinoma iz $GF(2^8)$. Prikažimo to sljedećim primjerom.

Primjer 1.53 *Neka je* $A(x) = x^7+x^5+x^4+1$. *Taj polinom možemo zapisati u obliku bajta:*

$$A(x) = (10110001)_2 = (B1)_{16.}$$

Sada za bajt $B1$, *u tablici pronađemo element na presjeku retka* B *te stupca* 1. *Vidimo da je to element* $(E0)_{16}$.

$$(E0)_{16} = (11100000)_2 = x^7 + x^6 + x^5 = A^{-1}(x).$$

Ovaj rezultat možemo dodatno provjeriti s:

$$A^{-1}(x) \cdot A(x) = (x^7 + x^6 + x^5) \cdot (x^7 + x^5 + x^4 + 1) \equiv 1 \pmod{P(x)}.$$

Poglavlje 2

AES i GCM-AES kriptosustav

2.1 Moderni simetrični blokovni kriptosustavi

Da bismo mogli opisati algoritam AES kriptosustava koji je jedan od ključnih pojmova u ovom radu, prvo ćemo se podsjetiti definicija kriptosustava i simetričnih blokovnih kriptosustava.

Definicija 2.1 *Kriptosustav je uređena petorka* $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *za koju vrijedi:*

- $\mathcal{P}$ *je konačan skup osnovnih elemenata otvorenog, početnog teksta.*
- $\mathcal{C}$ *je konačan skup osnovnih elemenata šifrata.*
- $\mathcal{K}$ *je konačan skup ključeva.*
- *za svaki* $k \in \mathcal{K}$ *postoji funkcija šifriranja* $e_k \in \mathcal{E}$ *i odgovarajuća funkcija dešifriranja* $d_k \in \mathcal{D}$*. Pritom su* $e_k : \mathcal{P} \to \mathcal{C}$ *i* $d_k : \mathcal{C} \to \mathcal{P}$ *funkcije sa svojstvom da je* $d_k(e_k(x)) = x$*, za svaki otvoreni tekst* $x \in \mathcal{P}$*.*

Iz ove definicije slijedi da funkcije šifriranja e_k moraju biti injekcije. Doista, ako bi za dva različita početna teksta x_1, x_2 dobili isti šifrat, primatelj ga ne bi mogao ispravno dešifrirati, te funkcije dešifriranja d_k ne bi bile dobro

definirane. Za neki kriptosustav kažemo da je simetrični blokovni kriptosustav, ako se obrađuje blok po blok elemenata otvorenog teksta x, tako da koristimo ključ K koji je isti i za šifriranje i za dešifriranje. To nas navodi na zaključak da je tajnost ključa najbitnija za sigurnost kod takvih kriptosustava. Prvo ćemo opisati svojevrsnog prethodnika AES kriptosustava – DES kriptosustav.

2.1.1 DES

Kriptografija kroz 1970-e postaje sve više zanimljiva ljudima u različitim industrijama, pogotovo zbog razvoja financijskih transakcija, koje postaju neizostavne u svakodnevnom poslovanju. Pojavila se potreba za šifrom koja će biti korištena među ljudima cijeloga svijeta, dakle trebalo je propisati nekakav standard koji bi ta "nova" šifra zadovoljavala. Američki NBS (*National Bureau of Standards*) je to upravo i napravio. Raspisali su javni natječaj za novi kriptosustav koji je trebao zadovoljavati ove uvjete:

- vrlo visok stupanj sigurnosti
- potpuna specifikacija te lako razumijevanje algoritma
- sigurnost nije u tajnosti algoritma, već u tajnosti ključa
- laka dostupnost svim korisnicima
- prilagodljivost kod uporabe pri različitim primjenama
- ekonomičnost implementacije u elektroničkim uređajima
- efikasnost
- mogućnost provjere.

Konačno je pristigao prijedlog algoritma razvijenog od IBM-a. Taj algoritam je temeljen na Feistelovoj šifri[1], koju i danas koriste gotovo svi simetrični

[1]Kod Feistelove šifre, šifriranje i dešifriranje su slične operacije te se obje sastoje od funkcija koje se iterativno ponavljaju određeni broj puta.

blokovni algoritmi. Ključno u toj šifri je naizmjenično korištenje supstitucija i transformacija kroz više iteracija, odnosno rundi. Na kraju je 1976. godine taj novi kriptosustav prihvaćen kao standard, te je dobio ime *Data Encryption Standard* (DES). DES šifrira početni tekst duljine 64 bita, koristeći ključ koji je duljine 56 bitova, da bi dobio šifrat duljine 64 bita. U ovom radu nećemo dati čitav opis algoritma, već samo spomenuti da su funkcije koje se koriste u samom algoritmu linearne uz iznimku "S-kutija" koje su ključne za samu sigurnost algoritma. Sama konstrukcija S-kutija nam je nepoznata te nam nisu poznati razlozi zbog kojih S-kutije tako izgledaju. Kod AES-a koji je nasljednik DES-a to neće biti tako, već će način konstrukcije tih S-kutija biti jasan, odnosno eksplicitno naveden.

2.1.2 Nedostatci DES-a

Za sami DES algoritam je ključno to što koristi ključ duljine 56 bitova, pa je već od same objave algoritma, glavna zamjerka bila kratka duljina ključa. No u konačnici su ostali pri duljini ključa od 56 bitova, ponajviše zbog toga da bi stao na tadašnje čipove. Ukupan broj mogućih ključeva iznosi 2^{56} te su već krajem 1970-ih postojale indicije i predviđanja da bi se ubrzo moglo konstruirati računalo koje bi u nekom razumnom vremenu moglo primjenom "grube sile" (eng. *Brute force*) razbiti poruke koje su šifrirane DES-om. Konačno je 1998. godine to i postignuto pomoću računala razvijenog od strane EFF-a (*Electronic Frontier Foundation*). Takav problem je riješen višestrukom upotrebom DES-a. Prihvaćenom je postala trostruka upotreba DES-a, nazvana Trostruki DES (često označavana i 3DES). Umjesto dosadašnjih 16 rundi koje su se koristile kod običnog DES algoritma, Trostruki DES koristi 48 rundi, odnosno tri puta nad istim tekstom vrši DES šifriranje. Također, koriste se 3 ključa (svaki duljine 56 bitova) K_1, K_2 i K_3 koji nužno ne moraju

biti različiti. Prvo šifriramo tekst (oznaka P) s K_1, pa dešifriramo s K_2, te opet šifriramo s K_3. S C označavamo šifrat.

$$C = e_{K_1}(d_{K_2}(e_{K_3}(P))),$$

$$P = d_{K_3}(e_{K_2}(d_{K_1}(C))).$$

Ovakav način koji pruža daleko više sigurnosti je bio prihvatljiviji nego da se kreira jedna potpuna nova, jača i sigurnija šifra. Ključan razlog zbog kojeg ovakav način šifriranja ima smisla i daje veću sigurnost od jednostrukog DES-a jest činjenica da skup od 2^{56} permutacija koje su dobivene DES-om (radi jednostavnosti taj skup ćemo nazvati ovdje DES) nije podgrupa grupe $\{0,1\}^{64}$. Štoviše, DES nije ni sam grupa jer nije ni zatvoren. Da je DES grupa, Trostruki DES ne bi bio nimalo snažniji od običnog DES-a jer bi kompozicija triju ključeva opet bila jedan ključ, te bi takvo korištenje više ključeva bio samo gubitak vremena. No pošto DES nije grupa, Trostruki DES se pokazao kao odlično rješenje novonastalog problema.

Pažljivog čitatelja bi moglo zanimati zbog čega se radije nije krenulo ka rješenju ovog problema primjenom dvostrukog DES-a, a ne trostrukog koji zahtijeva više vremena te više resursa. Ukoliko bi koristili dvostruki DES, svaki blok bi šifrirali s dva ključa K_1 i K_2 (od kojih su oba duljine 56 bitova), te budući da DES nije grupa, imamo osjećaj da time povećamo sigurnost, jer bismo očekivali da imamo $2^{56} \cdot 2^{56} = 2^{128}$ operacija potrebnih za dekriptiranje šifrata. No u stvarnosti sigurnost ostaje gotovo pa ista, odnosno grubom silom nam je potrebno tek dvostruko više operacija da bismo dekriptirali šifrat, 2^{57} u odnosu na 2^{56} operacija koje smo imali kod običnog DES-a. Razlog zbog kojeg nam sigurnost ne raste onoliko koliko smo očekivali je u jednom poznatom napadu opisanom još 1977. godine od strane Diffiea i Hellmana. Taj napad se zove "susret u sredini" (eng. *Meet in the middle*).

Napad "susret u sredini"

U kratkim crtama ćemo opisati taj napad koji nam pokazuje da dvostruki DES ne daje dovoljno poboljšanje u sigurnosti u odnosu na DES. Označimo s P naš tekst kojeg želimo šifrirati, a s C šifrat. Imamo

$$C = e_{K_2}(e_{K_1}(P)),$$

$$P = d_{K_1}(d_{K_2}(C)),$$

gdje je e_K funkcija šifriranja koja koristi ključ K, a d_K funkcija dešifriranja ključem K. Funkcija d_K je inverzna funkciji e_K. Naivni pristup za dobiti P, odnosno tekst, bio bi upotreba grube sile. Prvo bismo nad C koristili sve moguće ključeve K_2 za dešifriranje, pa onda još na svaki takav dobiven izlaz koristili i svaki mogući ključ K_1. Konačno bismo dobili $2^{k_1}2^{k_2} = 2^{(k_1+k_2)}$ mogućih tekstova P, gdje nam je k_i duljina u bitovima za ključ K_i, za $i = 1, 2$. U našem slučaju Dvostrukog DES-a to bi bilo 2^{128} mogućih kombinacija.
Napad "susret u sredini" koristi mnogo efikasniji pristup.

$$C = e_{K_2}(e_{K_1}(P)),$$

$$d_{K_2}(C) = d_{K_2}(e_{K_2}(e_{K_1}(P))) = e_{K_1}(P).$$

Ovdje šifrat C dešifriramo ključem K_2 te nam je takav tekst jednak šifriramo početnom tekstu P koji je šifriran ključem K_1. Dakle, napadaču je dovoljno nad šifratom C isprobati sve moguće ključeve K_2 za dešifriranje te nad početnim tekstom P sve moguće ključeve K_1 za šifriranje. Napadaču sada samo preostaje usporediti rezultate od $e_{K_1}(P)$ i $d_{K_2}(C)$ te kad pronađe poklapanje, ključeve K_1 i K_2 sprema kao potencijalne ključeve kojima će dešifrirati početni tekst P. Takve ključeve zovemo *ključevi - kandidati*. Napadač će na kraju odlučiti koji su mu *ključevi - kandidati* ispravni tako da će s tim parom testirati drugi početni tekst i šifrat.

U našem slučaju je i za jednu i za drugu operaciju ($e_{K_1}(P)$ i $d_{K_2}(C)$) potrebno 2^{56} operacija, odnosno ukupan broj operacija koji je potreban za razbijanje ove šifre je $2^{56} + 2^{56} = 2^{57}$, kako smo već i spomenuli ranije. Dakle, za poboljšanje DES-a uzet je Trostruki DES.

Sigurnost Trostrukog DES-a

Kod Trostrukog DES-a koristimo tri ključa K_1, K_2 i K_3 koji nužno ne moraju biti različiti. Jedna teoretska opcija nam odmah otpada, a to je ako su sva tri ključa jednaka, tj. $K_1 = K_2 = K_3$. Primijetimo da nam je tada Trostruki DES zapravo jednak običnom DES-u. Naime, nakon što smo šifrirali početni tekst, odmah smo ga i dešifrirali istim ključem, tako da zapravo u prva dva korištenja DES-a nismo ništa ni napravili. Tek trećim korištenjem šifriramo početni tekst, a to nam je zapravo korištenje običnog (jednostrukog) DES-a. Dakle dvije nam opcije preostaju kod odabira ključa, a to su dva različita ključa te sva tri različita ključa.
Kod verzije s dva ključa vrijedi $K_1 = K_3$. Imamo:

$$C = e_{K_1}(d_{K_2}(e_{K_1}(P))),$$

$$d_{K_1}(C) = d_{K_1}(e_{K_1}(d_{K_2}(e_{K_1}(P)))) = d_{K_2}(e_{K_1}(P)).$$

Za šifru šifriranu tom verzijom kriptosustava znamo potreban broj operacija za dekriptiranje. On iznosi $2^{56} + 2^{112} \approx 2^{112}$ operacija, jer kad koristimo napad "susret u sredini", za kombinaciju ključeva K_2, K_1 imamo $2^{56*2} = 2^{112}$ kombinacija, a za ključ $K_3 = K_1$ još 2^{56} različitih kombinacija. U ovoj verziji Trostrukog DES-a, napadom "susret u sredini" ne postižemo nikakvo poboljšanje u odnosu na predviđenu sigurnost kriptosustava.
Druga verzija, u kojoj imamo sva tri različita ključa, K_1, K_2, K_3,

$$C = e_{K_1}(d_{K_2}(e_{K_3}(P))),$$

$$d_{K_1}(C) = d_{K_1}(e_{K_1}(d_{K_2}(e_{K_3}(P)))) = d_{K_2}(e_{K_3}(P)).$$

nije imuna na napad "susret u sredini". Naime, iako se čini da trima različitim ključevima imamo sigurnost od $2^{56*3} = 2^{168}$, zapravo napadom "susret u sredini" (zbog jednakog razloga kao i kod verzije s dva ključa), dovoljno je ponovno 2^{112} operacija za dekriptiranje.

2.1.3 AES kriptosustav

Trostruki DES se je pokazao kao sjajna alternativa DES-u, no on je također imao svojih nedostataka, od kojih su ovo glavni:

- Trostruki DES koristi 48 rundi, a vjerojatno su dovoljne i 32 runde da bi se postigla jednaka sigurnost.
- 64 – bitni blokovi se nisu pokazali najefikasnijima kod nekih primjena.
- Programske implementacije Trostrukog DES-a su se pokazale presporima, pogotovo kod multimedijalnih podataka.

Iako je Trostruki DES poboljšanje DES-a, zbog ovih nedostataka ipak nije uzet kao standard, već je 1997. godine, *National Institute of Standards and Technology* (NIST) raspisao natječaj za kriptosustav koji će postati standard te kao takav zamijeniti DES. Do trenutka izbora novog kriptosustava za standard, tu je ulogu na neki način imao Trostruki DES. Tom novoizabranom kriptosustavu ime će biti **AES**, odnosno *Advanced Encryption Standard*. Za takav novi kriptosustav, NIST je postavio 3 glavna zahtjeva koja mora ispunjavati. AES mora biti:

1. simetričan
2. blokovni
3. takav da podržava rad sa 128 - bitnim blokovima i ključevima s tri duljine: 128, 192 i 256 bitova.

Godinu nakon objavljivanja natječaja, isti je i zaključen, te su kao konačni finalisti u obzir uzeti sljedeći kriptosustavi: RIJNDAEL, MARS, RC6, SERPENT i TWOFISH. Pobjednik tog natječaja bio je RIJNDAEL, razvijen od strane belgijskih kriptografa, kojeg je NIST konačno 26. studenog 2001. nakon nekoliko godina provedenih u procesu standardizacije, proglasio za AES. Premda su svi finalisti natječaja bili sigurni i otporni na sve poznate vrste napada, RIJNDAEL je izabran zbog manjih zahtjeva za memorijom, kao i zbog veće efikasnosti.

Opis rada AES kriptosustava

Kriptosustav AES šifrira blokove otvorenog teksta duljine 128 bitova. Njih prikazujemo kao 16 bajtova (bajt je duljine 8 bita) $b_0, b_1, \ldots, b_{15}$, koje možemo prikazati kao elemente matrice 4x4.

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

Svaka takva matrica se naziva AES blok. Svaki element AES bloka možemo prikazati kao polinom iz polja $GF(2^8)$. $GF(2^8)$ je konačno polje s 2^8 elemenata koje možemo shvatiti kao polinome oblika

$$a_7x^7 + a_6x^6 + \cdots + a_1x + a_0,\ a_i \in \{0, 1\}.$$

Operacije u polju su zbrajanje i množenje polinoma iz $\mathbb{Z}_2[x]$ modulo fiksni ireducibilni polinom $f(x) = x^8 + x^4 + x^3 + x + 1$, tako da u našem slučaju uzimamo da je

$$GF(2^8) = \mathbb{Z}_2[x]/(f(x)).$$

Za polinom $f(x)$ zahtijevamo da bude ireducibilan nad $\mathbb{Z}_2[x]$, jer inače ne možemo dobiti konačno polje $GF(2^8)$. Takve polinome nam je pogodno prikazivati nizovima 8 bitova (odnosno bajtom) pa onda još kraće i s dvije heksadecimalne znamenke.

Primjer 2.2 *Polinom $g(x) = x^5 + x^2 + x + 1$ prikažemo bajtom 00100111, ili heksadecimalnim zapisom s 27.*

Ovisno o duljini ključa, različit broj rundi se koristi pri AES šifriranju, za 128 bitni ključ (koji nam je od primarnog interesa) se koristi 10 rundi, za 192 bitni 12 rundi, dok se za 256 bitni ključ koristi 14 rundi. Svaka runda se sastoji od sljedeće četiri operacije:

- SubBytes
- ShiftRows
- MixColumns
- AddRoundKey.

Radi lakšeg zapisa i razumijevanja elemenata AES bloka, svaki njegov element (bajt) b_i označimo s $a_{i,j}$, gdje je (i, j) mjesto u matrici na kojem se nalazi element b_i. Inicijalno se prije prve runde odradi operacija *AddRoundKey*, odnosno dodavanje ključa, dok se u posljednjoj rundi ne obavlja operacija *MixColumns*. Prva operacija, ***SubBytes*** doprinosi nelinearnosti šifre. Najjednostavniji prikaz ove operacije je pomoću S-kutije koja je prikazana u Tablici 2.1. Svaki bajt $a_{i,j}$ iz AES bloka se zamijeni sa $S(a_{i,j})$ tako da bajt $a_{i,j}$ prvo zapišemo u heksadecimalnom zapisu, prvom broju u takvom zapisu pridružimo redak, a drugom broju stupac S-kutije. Zatim na presjeku tog retka i stupca iz S-kutije pročitamo vrijednost $S(a_{i,j})$. Formalniji opis operacije *SubBytes* bio bi da nad svakim elementom AES bloka vršimo dvije operacije, prvo mijenjanje tog bajta njegovim multiplikativnim inverzom u

	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

Tablica 2.1: S - kutija

$GF(2^8)$, uz napomenu da 00 jedini nema inverz te zato ostaje nepromijenjen u ovom koraku. Drugi korak je da nad tim elementom AES bloka napravimo i afinu transformaciju. Nad svakim elementom $a_{i,j} = a_7a_6 \dots a_1a_0$ AES bloka, djelujemo na sljedeći način:

$$a'_i = a_i \oplus a_{(i+4) \bmod 8} \oplus a_{(i+5) \bmod 8} \oplus a_{(i+6) \bmod 8} \oplus a_{(i+7) \bmod 8} \oplus c_i,$$

za $i = 0, 1, \dots, 7$, gdje je $c = c_7c_6 \dots c_1c_0 = 01100011$.
S-kutija, odnosno preciznije, fiksna afina transformacija je birana tako da se izbjegnu potencijalne fiksne točke. Dakle, za svaki element b_i AES bloka vrijedi $a_{i,j} \neq S(a_{i,j})$.

Druga operacija, ***ShiftRows***, djeluje nad recima AES bloka. Ona ciklički pomiče $i-$ti redak AES bloka za i mjesta ulijevo, gdje je $i \in \{0, 1, 2, 3\}$.

Dakle prvi redak bloka ostane nepromijenjen, drugi se ciklički pomakne lijevo za jedno mjesto, treći za dva mjesta, a četvrti za tri mjesta. Vidimo da nakon ove operacije, svaki blok sadrži jednake elemente kao što je sadržavao i prije ove operacije, no nisu svi na istom mjestu. Iako ova operacija djeluje nad recima AES-bloka, njezina se važnost uočava nad stupcima istog bloka. Postiže se da se oni ne šifriraju neovisno, što je vrlo bitno, jer bi u suprotnom umjesto jednog bloka imali četiri nezavisna bloka po stupcima, a time i mnogo slabiju šifru.

Kod treće operacije, ***MixColumns***, stupce svakog AES bloka promatramo kao polinome nad $GF(2^8)$, tako da stupac $A_i = (a_{0,i}, a_{1,i}, a_{2,i}, a_{3,i})$ promatramo kao polinom $a_{3,i}x^3 + a_{2,i}x^2 + a_{1,i}x + a_{0,i}$. Te stupce, odnosno polinome dalje množimo modulo $x^4 + 1$ s fiksnim polinomom $a(x) = 03x^3 + 01x^2 + 01x + 02$. Razlog zbog kojeg množimo modulo polinom $x^4 + 1$, je taj da bismo kao rezultat množenja dobili polinom stupnja manjeg od 4. Polinom $x^4 + 1$ nije ireducibilan nad $\mathbb{Z}_2[x]$, odnosno vrijedi $x^4 + 1 = (x + 1)^4$. On ne treba ni biti ireducibilan, budući da fiksni polinom $a(x)$ ima inverz $a^{-1}(x) = 0Bx^3 + 0Dx^2 + 09x + 0E$. Činjenica da polinom $x^4 + 1$ nije ireducibilan nad $\mathbb{Z}_2[x]$, znači da neće svi polinomi iz $\mathbb{Z}_2[x]$ imati inverz. No za algoritam AES kriptosustava je dovoljno da barem jedan polinom taj inverz ima, a polinom $a(x)$ ga ima. Dakle nakon ove operacije stupac A_i postaje $M(A_i)$, gdje s M označavamo množenje stupca A_i polinomom $a(x)$. Ova operacija, zajedno s *ShiftRows*, pridonosi difuziji cijelog kriptosustava. Difuzija nam osigurava da se (statistički) promjenom jednog bita nad početnim tekstom, polovica bitova mijenja u šifratu.

Posljednja operacija koja se obavlja u pojedinoj rundi,***AddRoundKey***, je jednostavno XOR operacija AES bloka i pripadnog međuključa za trenutnu rundu. Da bismo dobili međuključeve, vrši se proširenje ključa k (eng. *Key*

Expansion). Neka nam je duljina ključa k 128 bitova, te neka je zapisan matricom 4 x 4 (AES blok). Ta četiri stupca AES bloka k ćemo proširiti na ukupno 44 stupca, gdje su prva 4 stupca naš ključ k a iz ostalih stupaca čitamo međuključeve $k_1, \ldots, k_{10}$. To "proširenje" stupaca obuhvaća kombinaciju operacija poput rotiranja bajtova za jedno mjesto ulijevo, računanja vrijednosti S-kutije nad bajtovima dane riječi te operacije XOR.

Dešifriranje AES kriptosustava

Dešifriranje AES vrši se na sličan način poput šifriranja, gdje umjesto *SubBytes*, *ShiftRows*, *MixColumns* i *AddRoundKey* koristimo njima inverzne operacije. Bitno je naglasiti da je i poredak operacija obrnut kod dešifriranja. Operacija *AddRoundKey* je sama sebi inverz, dok se inverzne operacije kod ostalih operacija vrlo lako mogu objasniti.

$SubBytes^{-1}$ koristi prvo inverz afine transformacije koju smo koristili kod *SubBytes* te zatim svaki element zamijeni inverzom u $GF(2^8)$, jer je u svakom polju $(a^{-1})^{-1} = a$. Kao što smo spomenuli da se dva koraka u *SubBytes* operaciji mogu zamijeniti S-kutijom koja u sebi kombinira ta dva koraka, tako i ovdje imamo inverznu S-kutiju.

$ShiftRows^{-1}$ ciklički pomiče elemente $i - tog$ retka AES bloka šifrata udesno za i mjesta, gdje je $i \in \{0, 1, 2, 3\}$, za razliku od *ShiftRows* koji ih je pomicao ulijevo.

$MixColumns^{-1}$ stupce AES bloka šifrata (koje opet kao i operaciji *MixColumns* promatramo kao polinome), množi modulo $x^4 + 1$ s polinomom $b(x) = 0Bx^3 + 0Dx^2 + 09x + 0E$, koji je inverz polinoma $a(x)$ fiksno zadanog za operaciju *MixColumns*.

Budući da primatelj također zna ključ k, on može istim algoritmom kao i pošiljatelj generirati međuključeve $k_1, \ldots, k_{10}$, no on ih koristi u obrnutom

redoslijedu.

Kriptoanaliza AES kriptosustava

AES je siguran i otporan na *brute force* napade kao i na ostale poznate metode kriptoanalize, poput linearne i diferencijalne kriptoanalize. Nekoliko je napada na algoritam AES kriptosustava koji su efikasniji od *brute force* metode. Zasad najefikasniji poznati napad koji napada sve runde je *Biclique* napad. Ostali poznati napadi su napadali reducirani broj rundi (obično 7 ili 8 rundi). Metoda *Biclique* ili prevedeno metoda kompletnog bipartitnog grafa je jedna varijanta napada "susret u sredini" kojemu je potrebno $2^{126.1}$ operacija za dekriptirati algoritam AES-a s duljinom ključa od 128 bitova te postiže malu prednost u odnosu na *brute force* napad kojemu treba 2^{128} operacija.

2.2 Moderni simetrični protočni kriptosustavi

Simetrični kriptosustavi se dijele na blokovne i protočne. Nakon što smo opisali način djelovanja blokovnih šifri, u prvom redu blokovnog AES-a, sada ćemo se posvetiti protočnim šiframa (eng. *stream ciphers*).

2.2.1 Protočne šifre

Blokovne šifre, koje smo upoznali, šifriraju cijeli blok početnog teksta odjednom s istim ključem. Dakle kako će se šifrirati pojedini bajt (zapravo slovo, jer je u ASCII kodu slovo predstavljeno jednim bajtom, odnosno s 8 bitova) unutar početnog teksta, ne ovisi samo o njemu i ključu kojim ga šifriramo, već ovisi i o svim ostalim bajtovima početnog teksta koji se nalaze u istom bloku. Kod protočnih šifri, svaki se bajt zasebno šifrira. To se postiže do-

davanjem bajta iz niza ključeva (eng. *key stream*) u bajt početnog teksta. Niz ključeva nam je zapravo nasumični (zapravo lažno nasumični) niz bajtova. Generiranje nasumičnih (eng. *random*) vrijednosti će biti obrađeno u sljedećem potpoglavlju. Protočne šifre se dijele na sinkrone (eng. *synchronous*) i asinkrone (eng. *asynchronous*).

- Sinkrone protočne šifre su one kod kojih niz ključeva ovisi jedino o ključu.
- Kod asinkronih protočnih šifri, niz ključeva, osim o ključu, ovisi i o početnom tekstu te o šifratu.

Šifriranje i dešifriranje protočnim šiframa

Definicija 2.3 *Neka su* $P = p_1, p_2, \ldots, p_n$, $C = c_1, c_2, \ldots, c_n$ *i* $S = s_1, s_2, \ldots, s_n$ *redom početni tekst, šifrat te niz ključeva, pri čemu su* $p_i, c_i, s_i \in GF(2^8)$, *za* $i \in \{1, 2, \ldots, n\}$. *Funkcija šifriranja protočnom šifrom glasi:*

$$e_{s_i}(p_i) = p_i \oplus s_i, \ i \in \{1, 2, \ldots, n\}.$$

Funkcija dešifriranja protočne šifre je:

$$d_{s_i}(c_i) = c_i \oplus s_i, \ i \in \{1, 2, \ldots, n\}.$$

Uočimo da su funkcije šifriranja i dešifriranja definirane na isti način, te dokažimo da je d_{s_i} uistinu funkcija dešifriranja. Koristimo svojstvo asocijativnosti operacije $\oplus$ te činjenicu da je $a \oplus a = 0$, za $a \in \{0, 1\}$.

$$d_{s_i}(c_i) = d_{s_i}(e_{s_i}(p_i)) = d_{s_i}(p_i \oplus s_i) = (p_i \oplus s_i) \oplus s_i = p_i \oplus (s_i \oplus s_i) = p_i.$$

Operacija isključivo ili

Oznaka operacije $\oplus$ koju smo koristili za šifriranje, odnosno dešifriranje, nam predstavlja operaciju isključivo ili (eng. *exclusive or*) koju kraće označavamo

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

Tablica 2.2: Tablica istinitosti za XOR operaciju ($C = A \oplus B$)

s XOR. Operacija XOR igra vrlo važnu ulogu u modernoj kriptografiji te će se ova operacija koristiti često i u ostatku ovog rada, a primijetimo i da je njeno djelovanje na elemente $0, 1$ analogno zbrajanju modulo 2, koje smo opisali u prvom poglavlju ovog rada.

Razlog zbog kojeg je ona toliko često korištena je zapravo vrlo jednostavan i lako vidljiv iz tablice istinitosti. Naime, bez obzira koji bit (0 ili 1) šifrirali, postoji jednaka mogućnost da on bude šifriran s nulom ili jedinicom, a to nam ovisi i o bitu s_i iz niza ključeva. A ako nam se niz ključeva generira nasumično (što ćemo i zahtijevati), to nam garantira da nikako ne možemo predvidjeti s čime će nam pojedini bit biti šifriran, odnosno jednaka je vjerojatnost da je šifriran nulom, kao i da je šifriran jedinicom.

Vidimo da je šifriranje vrlo jednostavno, odnosno operacija XOR početnog teksta i niza ključeva. Stoga je ključno kako generirati takav niz ključeva, gdje će taj niz ključeva ovisiti i o ključu same šifre, jer u tom postupku zapravo leži čitava sigurnost ovakvih šifri. Vrlo je bitno da ne miješamo pojmove ključa (eng. *key*) i niza ključeva (eng. *key stream*).

Navedimo još i da će različiti ključevi odrediti i da nizovi ključeva budu različiti, dok će kod istih ključeva i nizovi ključeva biti jednaki. Bez toga dešifriranje ne bi bilo moguće, jer primatelj poruke (koji ima isti ključ kao i pošiljatelj) mora moći izgenerirati isti niz ključeva kao što je izgenerirao i

pošiljatelj, da bi mogao dešifrirati poslanu poruku.

2.2.2 Generatori nasumičnih brojeva

Već smo spomenuli kako je niz ključeva nasumično (eng. *random*), odnosno lažno nasumično (eng. *pseudorandom*) generiran. Opišimo tri vrste generatora nasumičnih brojeva.

Generatori pravih nasumičnih brojeva

Generatori pravih nasumičnih brojeva (eng. *True random number generators*) ili kraće *TRNG* su nam dani u prirodi, odnosno mjerenjem nekih nasumičnih pojava u prirodi imamo definirane takve generatore. Primjer takve nasumične pojave je radioaktivni raspad tijekom kratkog vremenskog perioda. Neki lakše shvatljivi primjer može biti bacanje kockice ili bacanje novčića. Dva osnovna uvjeta koja nasumični ili slučajni brojevi moraju zadovoljavati su ovdje zadovoljena. Ta dva uvjeta su:

1. **Nepredvidljivost** koja nam osigurava da sljedeći broj ne možemo predvidjeti bez obzira na to što znamo sve brojeve koji su se u povijesti tim generatorom pojavili. Primjerice, ako smo bacili novčić deset puta i svaki put nam je pao na istu stranu, to nipošto ne znači da je veća vjerojatnost da jedanaesti put padne na drugu stranu. I dalje je ishod nepredvidljiv.
2. **Uniformnost** koja nam osigurava da ako smo dovoljno puno puta bacili primjerice kockicu, da se svih šest brojeva pojavljuje u približno jednakim omjerima.

Generatori lažno nasumičnih brojeva

Generiranje nasumičnih brojeva bacanjem kockica ili novčića nikako nije praktičan način, te je bilo potrebno definirati generatore koji će računalnim algoritmima dati nizove naizgled nasumičnih brojeva. "Naizgled" je ovdje ključna riječ, jer se tu ne radi o doista nasumičnim brojevima. Njih zovemo generatori lažno nasumičnih brojeva (eng. *Pseudorandom number generator*) te ih kraće označavamo s $PRNG$. Njihovi algoritmi imaju inicijalne, odnosno početne vrijednosti, koje nazivamo sjeme (eng. *seed, seed value*). Poznavajući tu inicijalnu vrijednost, i funkciju iz koje se iduće vrijednosti računaju i određuju, možemo rekonstruirati čitav niz tih "nasumičnih" brojeva. To je upravo razlog zbog čega te brojeve zovemo lažno nasumični (eng. *pseudorandom*). Njih obično računamo rekurzivno na sljedeći način:

$$s_0 = sjeme$$

$$s_{i+1} = f(s_i),\ i \in \mathbb{N}_0.$$

Generalizacija ovog generatora je u obliku $s_{i+1} = f(s_i, s_{i-1}, ..., s_{i-t})$, za neki fiksni broj t. Jedan od najčešćih načina kojima dobivamo niz slučajnih brojeva je pomoću generatora linearne kongruencije:

$$s_0 = sjeme$$

$$s_{i+1} \equiv as_i + b \pmod{m},\ a, b, m \in \mathbb{N},\ i \in \mathbb{N}_0.$$

Takvi brojevi, jasno nisu nasumični i ponavljaju se u nekom određenom periodu, no budući da za period uzmemo veliki broj, to periodičko ponavljanje možemo zanemariti.

Primjer 2.4 *Jedan primjer takvog generatora je funkcija* rand() *korištena u programskom jeziku C*

$$s_0 = 12345$$

$$s_{i+1} \equiv 1103515245 s_i + 12345 \pmod{2^{31}},\ i \in \mathbb{N}_0.$$

Vrlo je bitno da ovakvi generatori imaju dobra statistička svojstva, odnosno da nizovi lažno nasumičnih brojeva dobro aproksimiraju nizove pravih nasumičnih brojeva. Raznim statističkim testovima se ovakva svojstva provjeravaju.

Kriptografski sigurni generatori lažno nasumičnih brojeva

Generatori linearne kongruencije posjeduju jedan veliki nedostatak koji bi potencijalno mogao ugroziti sigurnost kriptosustava. Naime pojavi li se prethodno generirani broj ponovno u nizu, ono što nakon njega slijedi, bit će ponovno isti brojevi koji su slijedili nakon što je on i prvi put generiran. Stoga su uvedeni i generatori s dodatnim svojstvom, kako bi tu manjkavost eliminirali.

Kriptografski sigurni generatori lažno nasumičnih brojeva (eng. *cryptographically secure pseudorandom number generators*) su posebni tip generatora lažno nasumičnih brojeva s dodatnim svojstvom koje se zove **nepredvidljivost** (eng. *unpredictability*). To nam svojstvo osigurava da za danih n bitova niza ključeva $s_i, s_{i+1}, ..., s_{i+n-1},\ n \in \mathbb{N}$ nije moguće računalno odrediti sljedeće bitove s_{i+n}, s_{i+n+1}. Formalnije, ne postoji algoritam koji u polinomnom vremenu može s vjerojatnošću većom od 50% odrediti sljedeći bit s_{i+n}. To svojstvo se često naziva "test sljedećeg bita" (eng. *next-bit test*), odnosno za generator nasumičnih brojeva koji posjeduje svojstvo nepredvidljivosti, kažemo da prolazi "test sljedećeg bita". Ovakvi generatori imaju svojstvo da za danih n bitova ne možemo odrediti ni njegove prethodnike $s_{i-1}, s_{i-2}, ...$ Spomenuto svojstvo nepredvidljivosti je vezano samo uz kriptografiju, dok u drugim poljima to svojstvo nije potrebno.

2.2.3 Jednokratna bilježnica

Definicija 2.5 *Kriptosustav je savršeno siguran ako šifrat ne daje nikakvu informaciju o otvorenom tekstu.*

Ovakvu definiciju savršeno sigurnog kriptosustava dao je Claude Shannon 1949. godine. Takav kriptosustav može jedino biti moguć kada je

$$|\mathcal{K}| \geq |\mathcal{C}|.$$

Zbog činjenice da su funkcije šifriranja injekcije, znamo da vrijedi i

$$|\mathcal{C}| \geq |\mathcal{P}|.$$

A u graničnom slučaju kada je $|\mathcal{C}| = |\mathcal{P}| = |\mathcal{K}|$, Shannon je pokazao da će kriptosustav biti savršeno siguran ako i samo ako za svaki $x \in \mathcal{P}$ i za svaki $y \in \mathcal{C}$ postoji jedinstveni ključ $K \in \mathcal{K}$ takav da vrijedi $e_K(x) = y$.
Najpoznatija realizacija savršeno sigurnog kriptosustava je tzv. **jednokratna bilježnica** (eng. *one-time pad*), često kraće označavana i *OTP*.

Definicija 2.6 *Neka je $n \in \mathbb{N}$ fiksan prirodni broj, te neka je*

$$\mathcal{P} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$$

gdje su $\mathcal{P}, \mathcal{C}, \mathcal{K}$ redom prostori početnog teksta, šifrata i prostor ključeva. Za ključ $K = (k_1, k_2, \ldots, k_n) \in \mathcal{K}$ definiramo:

$$e_K(x_1, x_2, \ldots, x_n) = (x_1 \oplus k_1, x_2 \oplus k_2, \ldots, x_n \oplus k_n),$$

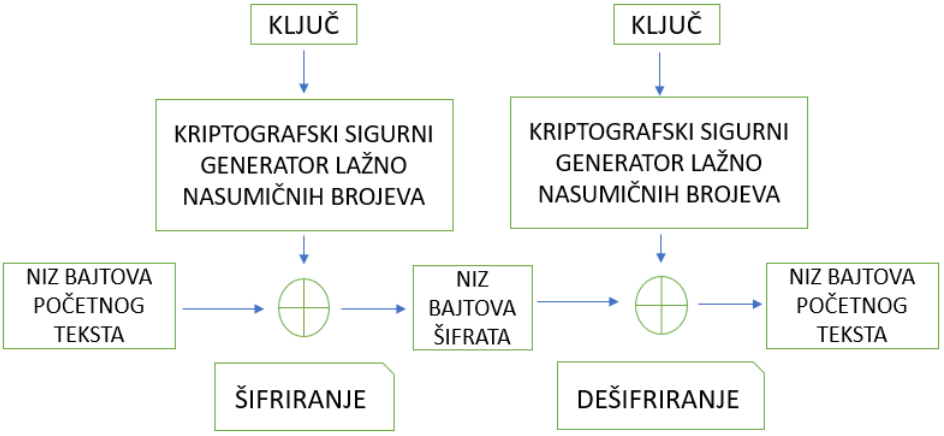
$$d_K(y_1, y_2, \ldots, y_n) = (y_1 \oplus k_1, y_2 \oplus k_2, \ldots, y_n \oplus k_n).$$

*Tako definiran kriptosustav zove se **jednokratna bilježnica**.*

Ovom kriptosustavu je lako odrediti ključ ako znamo, uz šifrat, i početni tekst. Naime, osim što vrijedi $y_i = x_i \oplus k_i$, također vrijedi i relacija $k_i = x_i \oplus y_i$. Zbog toga se sigurnost kod ovakvog kriptosustava postiže samo ako se svaki ključ koristi samo jednom. Zbog toga se i sami kriptosustav zove "jednokratna bilježnica". Praktični nedostatak ovakvom kriptosustavu je duljina ključa, koja je jednaka duljini same poruke. A budući da ključ također mora biti sigurno prenesen, ovakav kriptosustav nije pogodan za komercijalnu uporabu.

Zamjena jednokratne bilježnice protočnom šifrom

Vidjeli smo kako su funkcije šifriranja i dešifriranja kod protočne šifre i jednokratne bilježnice zapravo jednake funkcije. Tu dolazi do poboljšanja teorijski savršeno sigurne šifre - jednokratne bilježnice (koja nije praktična zbog duljine ključa), koja postaje protočna šifra na sljedeći način. Ključ nam više nije duljine kao početni tekst i šifrat, već iz njega generiramo niz ključeva koji će biti te duljine. Tako umjesto da vršimo operaciju XOR s ključem, vršimo ju s nizom ključeva. Generiranje niza ključeva je najvažniji korak kod protočnih šifri. Već je istaknuto kako u tom postupku leži sigurnost ovakvih šifri, te se iz tog razloga koristi kriptografski sigurni generator lažno nasumičnih brojeva. S takvim pravilno izabranim generatorom, može se postići da je sigurnost protočne šifre jednaka sigurnosti jednokratne bilježnice. Malo preciznije, teorijska analiza je pokazala da je za ključ duljine 128 bitova, period nakon kojeg će se početi ponavljati lažno nasumični brojevi, dulji od 10^{100}. Usporedbe radi, procjenjuje se da je ukupan broj atoma u poznatom svemiru između 10^{78} i 10^{82}.



Slika 2.1: Operacije šifriranja i dešifriranja kod protočnih šifri

2.3 Različiti načini djelovanja blokovnih šifri

AES kriptosustav ima pet standardiziranih načina djelovanja, a to su ECB (*Electronic Code Book*), CBC (*Cipher Block Chaining*), CFB (*Cipher Feedback*), OFB (*Output Feedback*) te onaj koji je nama najzanimljiviji CTR (*Counter*).

ECB

ECB način djelovanja AES šifre je najjednostavniji od navedenih. On početni tekst p podijeli na blokove $p_1, \ldots, p_n$ jednake duljine kao i ključ, s tim da je posljednji blok p_n po potrebi nadopunjen. Svi blokovi su onda šifrirani ključem k. Ovakvim načinom šifriranja dobivamo isti šifrat dvaju jednakih početnih tekstova. Zbog toga ovaj način nije preporučljiv.

CBC

Ulančavanje blokova šifrata (eng. *Cipher Block Chaining*) je jedan od načina da se ispravi taj nedostatak kod ECB načina djelovanja. Umjesto da se

svaki blok šifrira na isti način, ovdje svakom od n blokova dodamo neku dodatnu informaciju. Prvom bloku p_1 početnog teksta p dodamo inicijalni vektor (eng. *initialization vector*) kojeg kraće označavamo s IV, a ostalim blokovima $p_i, i > 1$ dodamo blok prethodnog šifrata c_{i-1}. Dakle, imamo

$$c_1 = e_k(p_1 \oplus IV)$$

$$c_i = e_k(p_i \oplus c_{i-1}),\ i \in \{2, 3, \ldots, n\}.$$

Jedan nedostatak ovakvog načina šifriranja jest to što nam se početni tekst p proširuje za jedan blok IV.

CFB

Cipher Feedback način šifriranja ili kraće CFB, baš kao i sljedeća dva načina (OFB i CTR), nam omogućuje da blokovni kriptosustav koristimo kao protočni. Protočni kriptosustav šifrira tako da na početni tekst primjeni operaciju $\oplus$ (XOR) s generatorom niza ključeva (eng. *keystream generator*). Neka nam je n veličina bloka. Tada u ovom načinu šifriramo dijelove bloka duljine s, gdje je $1 \leq s \leq n$. Početni tekst sada dijelimo na q blokova $p_1, p_2, \ldots, p_q$ duljine s bitova. Uz inicijalni vektor IV duljine n bitova koristimo i pomoćni registar R duljine n bitova. Na početku nam registar ima jednaku vrijednost kao i IV. U $i-$tom koraku šifriranja nad registrom R_i rabimo funkciju šifriranja e_k te taj rezultat spremamo u registar. Tada uzimamo prvih s bitova iz registra R_i te ih operacijom XOR zbrojimo s p_i. Rezultat ove operacije je c_i i njega spremamo na desnih s mjesta od registra. Mjesto za c_i stvorimo tako da pomaknemo bitove ulijevo za s pozicija. Šifriranje nastavljamo dok ga ne obavimo za sve blokove $p_i,\ i \in \{1, \ldots, q\}$. Budući da ovim načinom šifriramo manje blokove (općenito je $s \leq n$), time povećavamo broj koraka za obavljanje šifriranja te ga usporavamo. Još jedan nedostatak CFB-a je taj

što se generiranja niza ključeva ne može obaviti neovisno o samom šifriranju.

OFB

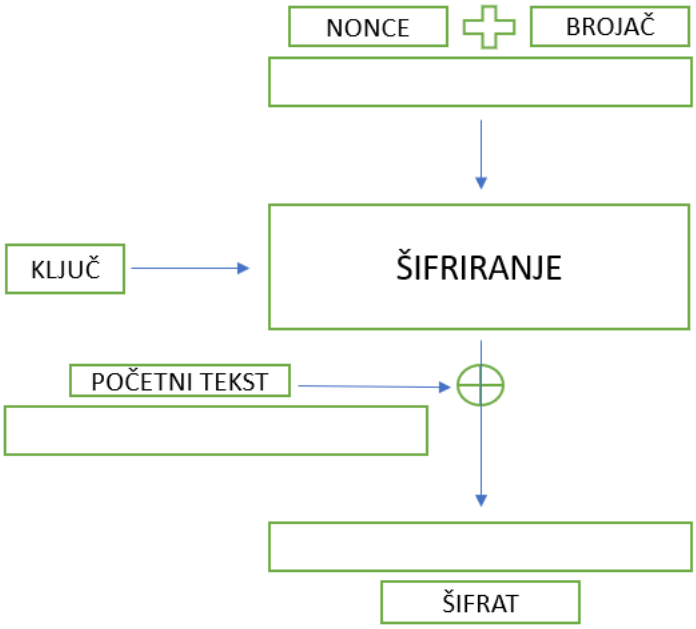
Output Feedback ili kraće OFB također omogućuje da blokovni kriptosustav koristimo kao protočni. Kod ovog načina djelovanja, niz ključeva se generira prije djelovanja operacije XOR nad šifratom. OFB kao i CFB koristi podatke o rezultatima (eng. *Feedback*) kao što im imena i sugeriraju. Razlika je u tome da CFB koristi prethodne šifrate, a OFB ne. Zbog toga u OFB načinu djelovanja ne može doći do propagacije grešaka, a u CFB je to bio slučaj.

CTR

Način brojača (eng. *Counter mode*) ili kraće CTR je najčešće korišten način djelovanja blokovnih šifri. Pri šifriranju, blokovi šifrata ne ovise o prethodnim blokovima šifrata, što nam odmah daje mogućnost paralelnog šifriranja. To uvelike ubrzava postupak šifriranja. Slično kao što smo u CBC, CFB i OFB načinu koristili IV, ovdje koristimo jednokratnu vrijednost $NONCE$ (eng. *Nonce - Number used only once*), koji je nasumično odabrana vrijednost veličine jednake bloku kojeg šifriramo i brojač i (eng. *Counter*). Brojač i nam zapravo predstavlja korak šifriranja.
Prvo djelujemo funkcijom šifriranja na $NONCE + i$ u $i-$tom koraku te taj rezultat zbrojimo operacijom XOR s p_i, koji je $i-$ti blok početnog teksta. Konačan rezultat u $i-$tom koraku je šifrat c_i

$$c_i = p_i \oplus e_k(NONCE + i).$$

U slučaju da zadnji blok p_q početnog teksta ima s bitova, $s < n$, odnosno manji broj bitova nego li je broj bitova u bloku, onda se samo prvih s bitova iz $e_k(NONCE+q)$ zbraja operacijom XOR s p_q, dok ostale bitove odbacujemo.



Slika 2.2: Grafički prikaz Counter načina šifriranja

Dakle nema potrebe za nadopunjavanjem tog posljednjeg bloka početnog teksta. Budući da se blokovi mogu nezavisno šifrirati, ne može doći niti do propagacije grešaka ako se neka pojavi u pojedinom bloku.

2.4 Galoisov način brojača - GCM

Galoisov način brojača (eng. *Galois/Counter Mode*) kojeg ćemo kraće označavati s GCM, je način djelovanja za simetrične blokovne šifre koji je široko prihvaćen i korišten (između ostalog i zbog velike brzine šifriranja, od preko 10 gigabita po sekundi). Primjenjuje se i u MACsec protokolu za Ethernet sigurnost (*IEEE 802.1AE*). GCM se obično definira za blokovne šifre s duljinom bloka od 128 bitova, no česte su izvedbe i za duljine blokova od 96 ili pak 256 bitova. Ovaj način, kao što mu ime kaže jest način brojača (CTR)

čije smo djelovanje već opisali. No GCM nam uz šifriranje osigurava i dodatna svojstva koja želimo da naša šifra posjeduje. GCM to postiže tako da računa autentikacijski kod poruke (eng. *Message Authentication Code*) kojeg kraće označavamo MAC. Često se u literaturi MAC označava i GMAC (eng. *Galois Message Authentication Code*) da bismo naglasili u kojem ga načinu rada koristimo. MAC možemo shvatiti kao kontrolni zbroj (eng. *Checksum*) koji se računa na strani pošiljatelja. Često se i naziva "kriptografski kontrolni zbroj". Primatelj će također izračunati MAC iz primljene poruke i usporediti ga s onim koji je pridodan poruci od strane pošiljatelja. Ovime su dvije postavke kriptografije zadovoljene.

- Prva je **autentikacija poruke** (eng. *message authentication*), odnosno činjenica da je primatelj siguran da je poruka stvorena i poslana od strane ispravnog pošiljatelja, od onog od kojeg i očekujemo poruku.
- Druga postavka koja nam je ovime osigurana zove se **integritet** (eng. *integrity*). Ona nam znači da nitko tijekom procesa slanja poruke, tu poruku nije neovlašteno izmijenio.

Ova dva svojstva koja nam kriptosustav sada posjeduje su vrlo bitna. Dosad smo razmišljali kako želimo šifrirati našu poruku tako da je samo primatelj kojem smo je namijenili zna dešifrirati i pročitati. Nitko u komunikacijskom kanalu tko presretne tu poruku nije u stanju tu poruku pročitati i razumjeti. No nitko nam ne može garantirati da ta poruka nije izmijenjena od strane neovlaštene osobe. Neka smo primjerice šifrirali iznos novca u nekoj bankovnoj transakciji. Ako je taj šifrat izmijenjen, a primatelj kod dešifriranja ne primijeti ništa neobično (u našem primjeru i dalje stoji iznos, no samo drukčiji od željenog), ta transakcija neće biti obavljena kako smo željeli i potencijalno nam se može nanijeti velika (materijalna) šteta. Ovakvo nešto

nam se ne može dogoditi kod blokovnih šifri, jer znamo da promjena jednog bita u bloku, utječe na promjenu cijelog bloka, no kod protočnih šifri, promjena jednog bita u šifratu nam taj iznos u zamišljenoj transakciji mijenja. To je zbog već opisanog načina šifriranja i dešifriranja protočnih šifri, gdje se koristi operacija *XOR* između početnog teksta (šifrata) i niza ključeva. Ovakav potencijalni rizik nije samo teorijski rizik, već je vrlo realan, jer doista nije teško pretpostaviti gdje se u poruci koju banka šalje nalazi iznos novca (ili čak i broj računa), budući da banke često koriste isti format poruke kod takvih transakcija. Autentikacijski kodovi poruke (MAC) nam taj rizik eliminiraju.

Prije nego objasnimo operacije šifriranja i dešifriranja, recimo i da ovaj način rada svoje ime duguje polju u kojem se vrše operacije. To polje je Galoisovo (konačno) polje $GF(2^{128})$. Operacije množenja u ovom polju obavljaju se modulo ireducibilni polinom $x^{128} + x^7 + x^2 + x + 1$.

Galoisov način brojača ima dvije operacije, **autenticirano šifriranje** (eng. *authenticated encryption*) i **autenticirano dešifriranje** (eng. *authenticated decryption*). Pridjev "autenticirano" je tu da dodatno naglasi najbitnije svojstvo koje ovakav način rada ima.

2.4.1 Autenticirano šifriranje

Funkcija autenticiranog šifriranja ima sljedeće ulazne parametre:

- Tajni ključ K duljine odgovarajuće duljini bloka (128 bitova).
- Inicijalni vektor IV kojeg shvaćamo kao $NONCE$ korišten pri opisu CTR načina djelovanja blok šifri. Preporučena duljina vrijednosti IV je 96 bitova jer se s tom duljinom IV može efikasnije procesuirati, iako se slobodno može izabrati bilo koja duljina između 1 i 2^{64}. IV možemo generirati generatorom lažno nasumičnih brojeva.

- Početni tekst P kojeg šifriramo.
- Dodatni autenticirani podatak (eng. *Additional authenticated data (AAD)*) kojeg kraće označavamo s A. To je podatak koji je autenticiran, ali nije šifriran. On može sadržavati razne vrijednosti, poput adresa, portova, verzija protokola i drugih polja koja govore na koji način obraditi ili proslijediti početni tekst.

Dva su izlazna parametra:

- Šifrat C duljine jednake početnom tekstu P.
- Autentikacijska oznaka (eng. *authentication tag*) koju označavamo s T. Njezina duljina može biti bilo koja vrijednost između 0 i 128. Duljinu od T označavamo s t.

Neka je početni tekst P podijeljen na blokove $p_1, p_2, \ldots, p_n$ od kojih su svi, osim eventualno p_n, duljine 128 bitova. Posljednji blok p_n ne mora uvijek biti kompletan, a njegovu duljinu ćemo označiti s u. Analogno ćemo i šifrat C podijeliti na blokove $c_1, c_2, \ldots, c_n$. Dodatni autenticirani podatak A prikazujemo nizom blokova $a_1, a_2, \ldots, a_m$, gdje posljednji blok u nizu, a_m, ne mora biti kompletan te njegovu duljinu označimo s v. Operacija autenticiranog šifriranja je prikazana sljedećim pravilima:

$$H = e_K(0^{128})$$

$$y_0 = \begin{cases} IV \| 0^{31}, & \text{ako je } \text{len}(IV) = 96. \\ GHASH(H, \{\}, IV), & \text{inače.} \end{cases}$$

$$y_i = incr(y_{i-1}),\ i = 1, 2, \ldots, n$$

$$c_i = p_i \oplus e_K(y_i),\ i = 1, \ldots, n-1$$

$$c_n = p_n \oplus MSB_u(e_K(y_n))$$

$$T = MSB_t(GHASH(H, A, C) \oplus e_K(y_0)).$$

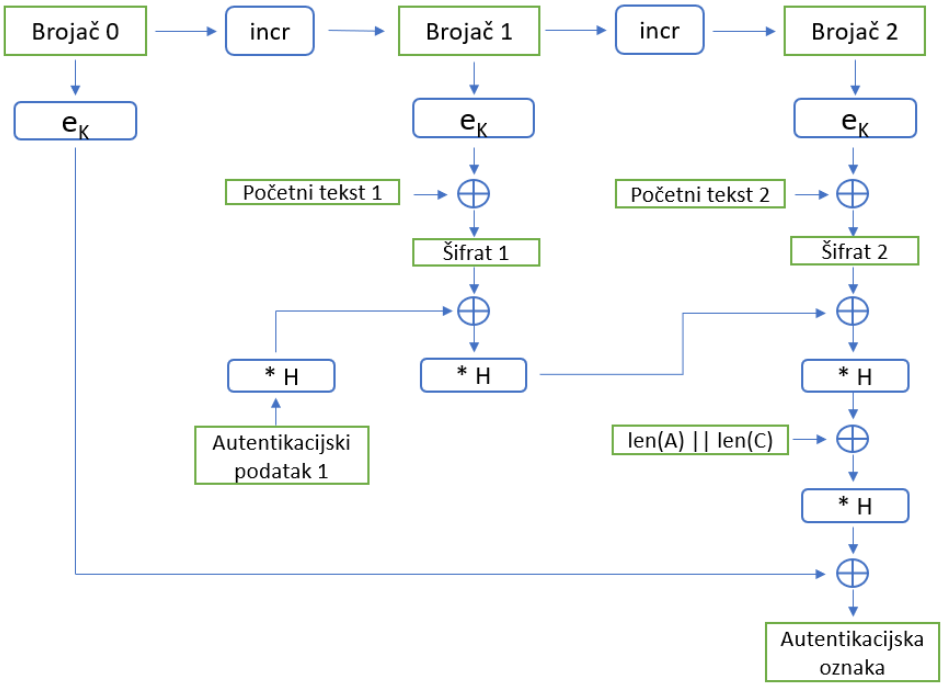
Kod ovih izraza smo koristili operator $\|$ koji nam predstavlja konkatenaciju dvaju stringova, funkciju *incr* koja predstavlja zbrajanje s brojem 1 (inkrement) modulo 2^{32}, funkciju MSB_a koja vraća a najznačajnijih bitova argumenta, funkciju *len* koja vraća duljinu argumenta u bitovima, simbol $\{\}$ koji predstavlja string duljine nula te funkciju $GHASH$. Funkcija $GHASH$ je definirana na sljedeći način:

$$GHASH(H, A, C) = x_{m+n+1}$$

gdje su varijable x_i, za $i = 0, 1, \ldots, m+n+1$ prikazane na sljedeći način, a pritom su ulazni parametri A i C podijeljeni na svoje blokove:

$$x_i = \begin{cases} 0, & \text{za } i = 0 \\ (x_{i-1} \oplus a_i) * H, & \text{za } i = 1, \ldots, m-1 \\ (x_{m-1} \oplus (a_m \| 0^{128-v})) * H, & \text{za } i = m \\ (x_{i-1} \oplus c_i) * H, & \text{za } i = m+1, \ldots, m+n-1 \\ (x_{m+n-1} \oplus (c_n \| 0^{128-u})) * H, & \text{za } i = m+n \\ (x_{m+n} \oplus (len(A) \| len(C))) * H & \text{za } i = m+n+1. \end{cases}$$

Ovdje smo opisali i postupak autentikacije, kod kojeg imamo niz autentikacijskih parametara x_i koji su dobiveni zbrajanjem ($\oplus$) parametra x_{i-1} i trenutnog šifrata c_i, odnosno dodatnog autenticiranog podatka a_i. Tu vrijednost pomnožimo ($*$) s *hash* međuključem H, kojeg smo dobili šifriranjem 128 - bitnog ulaza samih nula ključem K. Funkcije $\oplus$ i $*$ su standardno zbrajanje i množenje u polju $GF(2^{128})$.



Slika 2.3: Prikaz operacija kod autenticiranog šifriranja

2.4.2 Autenticirano dešifriranje

Funkcija autenticiranog dešifriranja ima pet ulaznih parametara: K, IV, C, A i T. Jedan je izlazni parametar, početni tekst P ili specijalni simbol $FAIL$ ako ulazni parametri nisu autentični. Ova funkcija će uvijek vratiti $FAIL$ kad ulazni parametri C, IV, A, T nisu kreirani funkcijom šifriranja s istim ključem K. Sljedećim pravilima opisujemo postupak autenticiranog dešifriranja, a sam postupak je sličan autenticiranom šifriranju:

$$H = e_K(0^{128})$$

$$y_0 = \begin{cases} IV \| 0^{31}, & \text{ako je len}(IV) = 96. \\ GHASH(H, \{\}, IV), & \text{inače.} \end{cases}$$

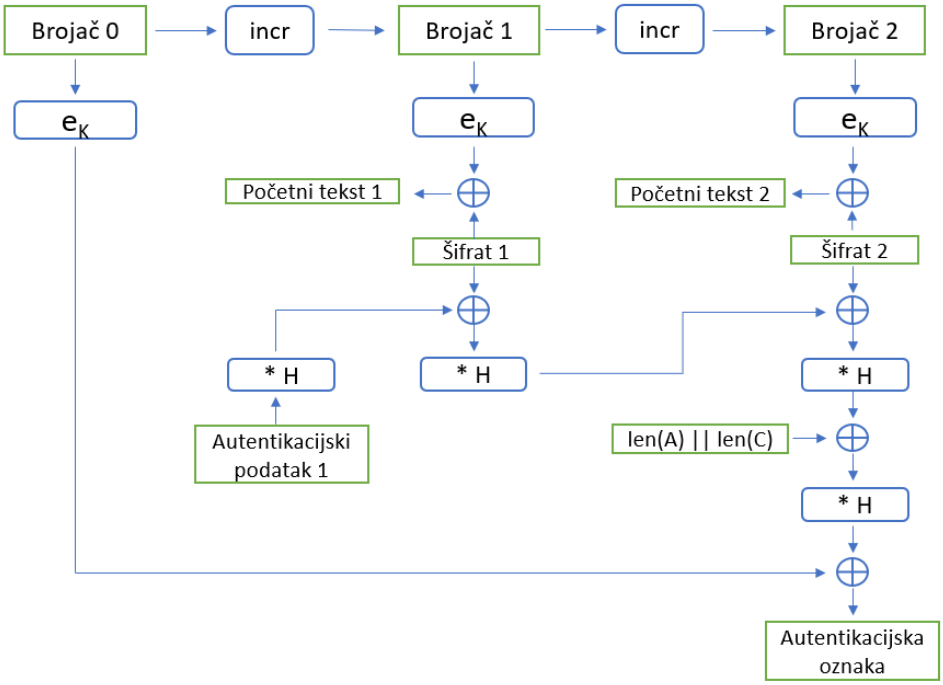
$$T' = MSB_t(GHASH(H, A, C) \oplus e_K(y_0))$$

$$y_i = incr(y_{i-1}),\ i = 1, 2, \ldots, n$$

$$p_i = c_i \oplus e_K(y_i),\ i = 1, \ldots, n-1$$

$$p_n = c_n \oplus MSB_u(e_K(y_n)).$$

Kod dešifriranja se računa vrijednost T' kako bi se mogla usporediti s vrijednosti T koja je pridružena šifratu C. Ako te dvije vrijednosti nisu jednake, funkcija će kao izlaz vratiti $FAIL$, a inače će se odrediti početni tekst P.



Slika 2.4: Prikaz operacija kod autenticiranog dešifriranja

Neka dodatna svojstva GCM načina rada AES kriptosustava

- Vidjeli smo da GCM omogućava šifriranje poruka zajedno s autentikacijom. Ponekad samo želimo autenticirati poruku, bez da je šifriramo. To je također podržano i onda taj način ne zovemo GCM, već GMAC.

- I za autentikaciju i za enkripciju koristimo samo jedan ključ. To nam zahtijeva manju količinu memorije koja nam je potrebna.
- I pošiljatelj i primatelj koriste jednaki inicijalni vektor IV.
- Paralelizam je prisutan jednako kao što je prisutan i kod CTR načina djelovanja, te je dodatno moguće paralelno vršiti šifriranje te autenticiranje što dodatno doprinosi brzini.
- Duljina šifrata jednaka je duljini početnog teksta.
- Ne dolazi do propagacije grešaka.

Poglavlje 3

MACsec protokol

3.1 OSI model

OSI (*Open Systems Interconnection*) model je model koji kroz sedam slojeva daje apstraktni opis arhitekture mreže. ISO (*International Organization of Standardization*), odnosno međunarodna organizacija za standardizaciju, ga je službeno prihvatila 1984. godine kao prvi standardni model za mrežnu komunikaciju. Od tada je prihvaćen u svim velikim računalnim i telekomunikacijskim kompanijama. Njegovih sedam slojeva je podijeljeno u dvije grupe:

- Donji sloj sa svoja četiri sloja: fizički sloj, podatkovni sloj, mrežni sloj i transportni sloj. Glavni zadatak ovih slojeva je prijenos podataka te se to ovim slojevima i definira.

- Gornji sloj koji obuhvaća sesijski sloj, prezentacijski sloj te aplikacijski sloj. Ova tri sloja opisuju komunikaciju između računala i korisnika, što uključuje i način rada korisnika s aplikacijama te međusobnu komunikaciju aplikacija.

Slika 3.1: Prikaz sedam slojeva OSI modela

Svaki sloj ima svoju funkcionalnost, pri obavljanju svojih funkcija koristi uslugu koju pruža niži sloj, a sam pruža uslugu višem sloju. To odvajanje slojeva nazivamo uslojavanje (eng. *layering*). Time su postignute mnoge pogodnosti, jer je mrežna komunikacija svedena na manje i jednostavnije dijelove. Ukratko ćemo dati opis svakog sloja, pri čemu ćemo posebnu pažnju obratiti na drugi sloj, odnosno podatkovni sloj, jer je upravo na tom sloju definiran MACsec protokol.

3.1.1 Fizički sloj OSI modela

Prvi odnosno najniži sloj OSI modela je fizički sloj (eng. *Physical Layer*). On definira te je odgovoran za fizička svojstva mrežnih uređaja te za same fizičke komponente mreže, kao što su konektori, mediji za prijenos, oblik signala, razina napona i brzina prijenosa podataka. Svi ti parametri su definirani protokolima ovog sloja.

3.1.2 Podatkovni sloj OSI modela

Sljedeći sloj u hijerarhiji je podatkovni sloj (eng. *Data Link Layer*). Taj sloj je odgovoran za razmjenu podataka između mrežnih uređaja (općenito čvorova na mreži) koji su fizički povezani na mreži. Pri razmjeni podataka, podaci su upakirani u okvire (eng. *Frames*). Ovaj sloj regulira protok okvira tako da ne bi došlo do zagušenja, koje se može pojaviti u slučaju kad je primatelj okvira spor, a pošiljatelj brz. Također u slučaju dupliciranih okvira, podatkovni sloj će ih identificirati te obrisati.

Pakiranje bitova u okvire

Na podatkovnom sloju, bitovi se, za razliku od fizičkog sloja, pakiraju u posebne oblike podataka, koje zovemo okviri (eng. *Frames*). Ako je veličina samih okvira prevelika, oni se dalje mogu podijeliti u više manjih okvira. Takvi se okviri kreiraju na kraju podatkovnog sloja koji prima podatke u obliku signala od fizičkog sloja. Svi okviri imaju sličnu strukturu, koja se sastoji od tri glavna dijela. Oni su zaglavlje, polje podatka koji se šalje i završetak. Zaglavlje i završetak se mogu zajednički nazvati polja s upravljačkom informacijom. Zaglavlje sadrži dijelove poput početka okvira, polja adrese, tipa i upravljačkog polja, dok završetak sadrži polje provjere (detekcija grešaka) te završetak okvira.

Sinkronizacija

Problem kojim se ovaj sloj bavi je sinkronizacija, odnosno dogovor o detekciji početka, kao i kraja okvira koji se prenosi, kojeg moraju postići dvije strane uključene u prijenos. Više je načina kojima se sinkronizacija može postići, a među najčešćim su omeđivanje početnim i završnim znakom ili zastavicom (eng. *flag*) te brojanjem znakova. Metoda omeđivanja početnim i završnim

znakom (točnije nizom znakova) se češće koristi. Njome je postignuto to da svaki okvir počinje s istim nizom znakova koji označavaju početak teksta, te svaki okvir završava s istim nizom znakova koji označavaju završetak teksta. Ovom metodom postižemo da ako ipak dođe do gubitka sinkronizacije u jednom dijelu prijenosa podataka, da strana primatelja samo potraži niz znakova koji označava početak odnosno kraj teksta.

Kontrola pogreški i kontrola toka

Sljedeći problem koji pri eventualnom pojavljivanju na fizičkom sloju biva ispravljen na podatkovnom sloju jest ispravljanje pogrešaka, odnosno kontrola pogreški. Da bi se osigurao pouzdani prijenos podataka koristi se potvrda prijema (eng. *Acknowledgement*). Potvrda prijema može biti ili pozitivna ili negativna. Pozitivna potvrda prijema će biti poslana u slučaju primitka ispravnog okvira. Negativna potvrda prijema će biti poslana u situaciji kad je primljen oštećen, odnosno dupliciran okvir. Nakon negativna potvrde prijema, okvir se mora ponovno poslati. Naravno može se dogoditi da se okvir ili potvrda prijema uopće ne pošalju, pa da se ne bi dogodilo predugo čekanje na nešto što možda nikada neće ni stići, uvodi se brojač. Taj brojač će čekati jedan određeni vremenski period (eng. *Timeout*) da bi okvir stigao do odredište te da prijemna strana natrag pošalje potvrdu prijema. Ako unutar tog vremenskog perioda stigne natrag potvrda prijema (pozitivna), pošiljatelj onda zna da je okvir poslan uspješno, te će nastaviti sa slanjem sljedećeg okvira iz reda za slanje. U slučaju da ne stigne natrag potvrda prijema, brojač će upozoriti pošiljatelja na problem, te će pošiljatelj znati da je ili okvir ili potvrda prijema negdje izgubljena, te će ponovno poslati okvir. U tim situacijama se može dogoditi da kod primatelja dođe i do dupliciranja okvira, pa da bi se to spriječilo, svakom se okviru zadaje redni broj te pri-

matelj onda nema problema pri njihovom razlikovanju. Osim ovako opisanog protokola za kontrolu pogrešaka, postoje i drugi koji na podatkovnom sloju rade isti zadatak.

Detekcija i korekcija pogrešaka

Spomenuli smo da će negativna potvrda prijema biti poslana i u slučaju kad je primljen oštećen okvir. Postavlja se pitanje kako detektirati te kako ispraviti takve oštećene okvire. Greške na okviru podataka su vrlo česte. Različite ih stvari na fizičkom sloju mogu prouzročiti. Postoje tri tipa na koje se ovakve greške mogu podijeliti. To su:

- Pogreške na jednom bitu (eng. *Single bit error*). Takve promjene jednog bita neće utjecati na susjedne bitove te je ovakve greške relativno lako naći i ispraviti.
- Pogreška na više bitova (eng. *Multiple bits error*). Bitovi koji su promijenjeni nisu uzastopni, te su omeđeni s ispravnim bitovima.
- Pogreške u nizu bitova (eng. *Burst error*) gdje je više od jednog uzastopnog bita promijenjeno.

Dva su pristupa kojima se štitimo od takvih pogrešaka. Prvi pristup, u kojem želimo samo otkriti (detektirati) grešku nazivamo detekcija greške (eng. *Error detection*), dok drugi pristup, kojim tu grešku želimo odmah na prijemnoj strani i ispraviti, zovemo korekcija greške (eng. *Error correction*). Kod oba pristupa, pošiljatelj uz bitove poruke šalje i dodatne bitove, odnosno redundanciju kojom se detektira, odnosno korigira greška. Često nam je isplativije ponovno poslati podatke nego li ispravljati greške na prijemnoj strani, pa tada koristimo detekciju greški. Tri osnovne tehnike kojima detektiramo greške su:

- Provjere pariteta (eng. *Parity checks*),
- Kontrolni zbroj (eng. *Checksum*),
- Cikličke provjere redundancije (eng. *Cyclic redundancy checks*).

Vrlo je bitno naglasiti da ovim tehnikama možemo otkriti različite vrste grešaka, no ako njima ne otkrijemo nijednu grešku, to nam ne daje zaključak da greške u stvarnosti nema.

Provjera pariteta se obavlja dodavanjem dodatnog bita na podatak, kojeg zovemo paritetni bit. Time broj jedinica činimo parnim (u slučaju parnog pariteta) ili neparnim (u slučaju neparnog pariteta). U slučaju parnog pariteta, pošiljatelj pri slanju podatka broji jedinice te ako je paran broj jedinica, kao paritetni bit dodaje nulu. Ako je broj jedinica neparan, kao paritetni bit dodaje jedinicu. U slučaju neparnog pariteta, situacija je obrnuta. Primatelj, pri primitku također broji jedinice. U slučaju parnog pariteta, ako je broj jedinica paran, prihvaća taj okvir, a ako je neparan, taj okvir se odbacuje. Pravilo je analogno pri slučaju neparnog pariteta. Bitno je naglasiti da je provjera pariteta, kao tehnika, prihvatljiva samo u slučaju pogrešaka na jednom bitu.

Kod kontrolnog zbroja se uz poruku primatelju šalje i zbroj svih jedinica u bloku. Primatelj onda sam izračuna broj jedinica te ih usporedi s dobivenom vrijednošću, pa zna hoće li dobiveni okvir prihvatiti ili ne.

Metoda cikličke provjere redundancije je metoda koja se danas najčešće koristi za detekciju grešaka. U ovoj metodi se niz bitova promatra kao niz koeficijenata polinoma. Dakle niz

$$a_n, a_{n-1}, a_{n-2}, \ldots, a_1, a_0$$

odgovara polinomu

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0.$$

Pošiljatelj i primatelj moraju imati dogovor oko toga koji polinom $G(x)$ se uzima kao polinom generator. Zatim izračunamo

$$x^k P(x) : G(x),$$

gdje je k stupanj polinoma G. Dobiveni ostatak pri dijeljenju označimo s $R(x)$ te koeficijente tog polinoma dodamo na kraj okvira. Tako dobiveni okvir primatelj dijeli s $G(x)$, te ako za ostatak pri tom dijeljenju dobije nulu, zaključuje da nema grešaka pri prijenosu tog okvira. U suprotnom, postoje greške pri prijenosu. Ova metoda je dobra za otkrivanje pogrešaka u nizu bitova (eng. *Burst errors*). Osim ovih nabrojenih uloga podatkovnog sloja, postoji i kontrola pristupa (eng. *Access control*). Tu ulogu ćemo spomenuti i posebno obraditi u poglavlju u kojem ćemo se baviti arhitekturom lokalnih mreža.

3.1.3 Mrežni sloj OSI modela

Sljedeći sloj OSI modela je mrežni sloj. U tom sloju segment podataka se zove paket (eng. *Packet*). Glavna uloga mrežnog sloja je prijenos paketa podataka između čvorova na općenito različitim mrežama te usmjeravanje (eng. *Routing*). Usmjeravanje bi bilo pronalaženje najboljeg puta kojim će paket stići na odredište. Tu funkciju obavljaju uređaji koje nazivamo usmjerivači (eng. *Router*). Algoritmi za usmjeravanje su odgovorni za donošenje odluke kojim putem se paketi prenose. Njih dijelimo na statičke (koji put biraju unaprijed te taj put šalju svim usmjerivačima kod podizanja mreže) te na dinamičke (koji se kod donošenja odluke prilagođavaju promjenama u prometu te u topologiji). Mrežni sloj koristi logičko adresiranje (eng. *Logical Addressing*) za adresiranje paketa, najčešće pomoću protokola IP.

3.1.4 Prijenosni sloj OSI modela

Prijenosni ili transportni sloj ima ulogu prijenosa podataka između uređaja. Na ovom sloju podaci se pakiraju u segmente (eng. *Segment*). To je zadnji sloj u kojem se podaci pakiraju. Prijenosni sloj preko svojih protokola definira način na koji će se veza uspostaviti između izvora i odredišta. Najčešći protokoli ovog sloja su TCP (*Transmission Control Protocol*) i UDP (*User Datagram Protocol*). TCP ima kontrolu nad svakim paketom, te za svaki "zagubljeni" paket, zatraži ponovno slanje. Time se daje prednost integritetu podatka koji se šalje u odnosu na brzinu slanja. Praktički obrnuti pristup ima UDP. Kod njega je bitna brzina prijenosa, dok protokol nema kontrolu nad tim je li se eventualno neki paket izgubio putem. Gubitak pojedinačnog paketa nam nije toliko bitan, već je bitnije da se cijela poruka što brže prenese, pa zbog toga ovaj protokol veliku primjenu ima kod komunikacije, odnosno prijenosa govora, kao i kod multimedijalnih aplikacija. Za prijenosni sloj se često kaže da je on srce OSI modela.

3.1.5 Gornja tri sloja OSI modela

Sesijski sloj OSI modela

Sesijski sloj je odgovoran za uspostavu veze, upravljanje sesijom i prekid veze između aplikacija. Ako dođe do prekida veze iz bilo kojeg razloga, ponovno je uspostavlja. Sinkronizacija veze, koja je vrlo bitna je također osigurana ovim slojem. Pri sinkronizaciji koristi kontrolne točke (eng. *Checkpoints*) koje pomažu pri detekciji grešaka, te će se tu veza ponovno sinkronizirati ako je potrebno.

Prezentacijski sloj OSI modela

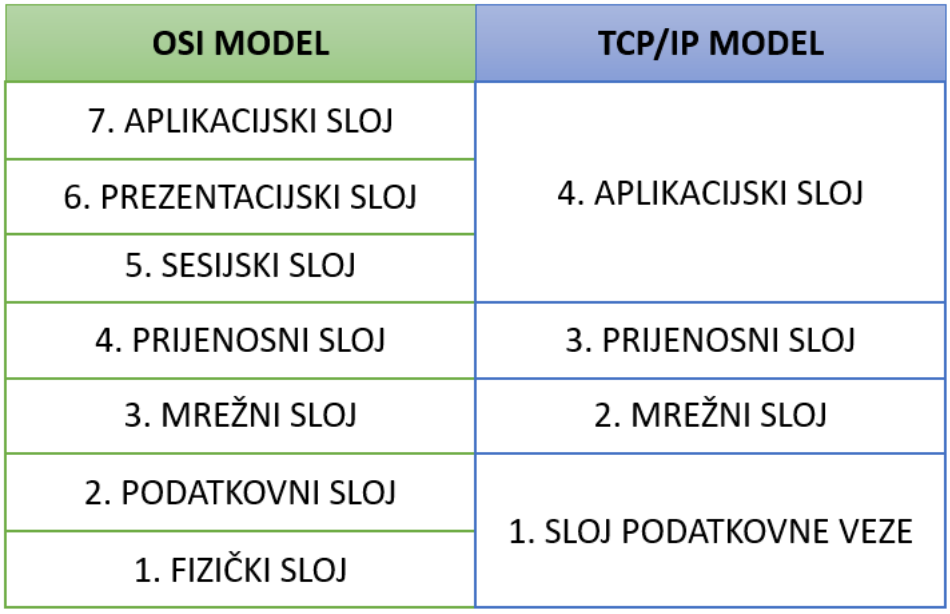
Prezentacijski sloj se ponekad naziva i sloj prevoditelj. Upravo nas taj drugi naziv upućuje na glavnu funkciju ovog sloja, a to je prevođenje kodova (primjerice ASCII koda) kako bi se na gornjem sloju podaci mogli prikazati na standardni način razumljiv korisniku. Ovaj sloj se bavi i sažimanjem podataka kako bi se lakše mogli prenijeti mrežom. To je posebno bitno kod prijenosa velikih multimedijalnih datoteka.

Aplikacijski sloj OSI modela

Na samom vrhu OSI referentnog modela je aplikacijski sloj. Na njemu se definiraju sučelja prema krajnjim korisnicima mrežnih usluga. On pruža usluge aplikacijama van OSI modela te upućuje zahtjev za uslugama prezentacijskog sloja. Primjer protokola na ovom sloju je FTP (*File Transfer Protocol*) koji služi za postavljanje i preuzimanje datoteka na poslužitelj, odnosno s poslužitelja.

3.2 TCP/IP model

OSI model služi kao referentni ili konceptualni model koji se praktički i ne koristi u praksi. U praksi je zato zastupljen TCP/IP model kojim ostvarujemo komunikaciju preko mreže. Ovaj model je ime dobio po dva najvažnija protokola koje koristi u mrežnom sloju (IP) i transportnom sloju (TCP). Ključna razlika između dva modela je u broju slojeva. TCP/IP za razliku od OSI modela koji koristi sedam slojeva, koristi četiri sloja, no na način da su gornja tri sloja OSI modela (aplikacijski, prezentacijski i sesijski sloj) objedinjena u jedan sloj kojeg jednim imenom zovemo aplikacijski sloj, te donja dva sloja objedinjena u jedinstveni sloj podatkovne veze. Na slici 3.2



Slika 3.2: Usporedba OSI modela s TCP/IP modelom

možemo vidjeti te razlike u broju slojeva. No ključna razlika jest ta da je TCP/IP model temeljen na protokolima, kao što mu ime i sugerira. Dakle, prvo su razvijeni protokoli, a zatim model, dok je kod OSI model prvotno on razvijen i opisan, a zatim su se kreirali protokoli.

3.3 Arhitektura lokalnih mreža

3.3.1 Lokalna mreža

Lokalna računalna mreža ili kraće LAN (eng. *Local Area Network*) je komunikacijska mreža koja je namijenjena za međusobno povezivanje različitih uređaja na malim udaljenostima, odnosno na malom prostoru. Najčešće mislimo na područje jedne zgrade kad govorimo o rasponu lokalne mreže. Osnovna joj je funkcija visoko pouzdani prijenos podataka velikom brzinom na

malim udaljenostima. Propusnost komunikacijskog kanala je toliko velika da se gotovo jednakom brzinom mogu dohvatiti podaci s udaljenog računala unutar iste lokalne mreže kao i podaci s diska vlastitog računala. Glavna prednost ovakvog umrežavanja računala je dijeljenje zajedničkih resursa između više različitih računala. Osnovni zahtjevi koje moramo zadovoljiti pri kreiranju lokalne mreže su:

- Omogućiti veliku brzinu prijenosa i veliku širinu propusnog pojasa
- Osigurati visoku pouzdanost komponenti lokalne mreže, a u slučaju kvara komponente, taj kvar se ne smije propagirati na ostatak mreže
- Kompatibilnost uređaja te jednostavnost konfiguracije mreže
- Omogućiti proširenje mreže novim računalima u svakom trenutku
- Držati se grupe standarda *IEEE 802* (Standardi za lokalne mreže).

Skup tehnologija za povezivanje uređaja unutar lokalne mreže, koji omogućuju njihovu međusobnu komunikaciju definiranu protokolima, naziva se **Ethernet**. Lokalna mreža radi na dva najniža sloja OSI modela. Dakle, njena arhitektura je definirana fizičkim slojem te dvama podslojevima podatkovnog sloja:

- Sloj upravljanja logičkom vezom ili LLC (eng. *Logical Link Control*)
- Sloj upravljanja pristupom ili MAC (eng. *Media Access Control*).

3.3.2 Upravljanje logičkom vezom (LLC)

LLC podsloj ima zadatak komunicirati s višim slojevima OSI modela te čini gornju polovicu podatkovnog sloja u ovakvoj podjeli. To bi značilo da preuzima pakete od mrežnog sloja te ih onda pretvara u okvire. Nadalje, on kontrolira sinkronizaciju, multipleksiranje odnosno demultipleksiranje paketa. Zadužen je i za kontrolu pogrešaka te kontrolu toka.

3.3.3 Upravljanje pristupom (MAC)

Dok je LLC implementiran softverski, MAC je implementiran hardverski na mrežnoj kartici. To je podsloj koji je u kontaktu s fizičkim slojem i koji definira protokole kojima se pristupa fizičkom mediju ili nekim drugim zajedničkim resursima. Budući da mnoge mreže koriste zajedničke resurse (primjerice jedan zajednički mrežni kabel), potrebno je imati pravila za upravljanje takvim resursima kako ne bi došlo do konflikata. MAC je odgovoran za enkapsulaciju okvira tako da budu pogodni za prijenos fizičkim medijem.

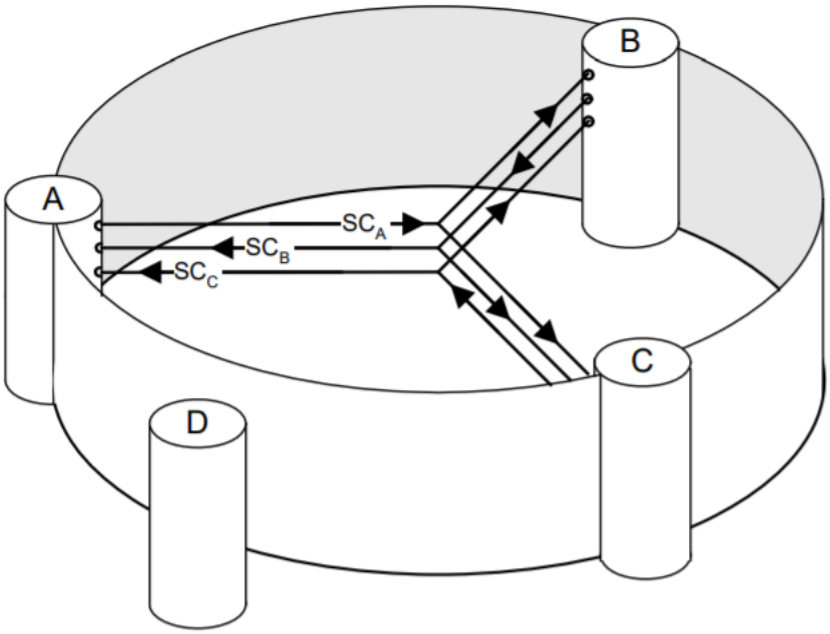
3.4 Opis MACsec protokola

MACsec (eng. *Media Access Control security*) protokol je definiran standardom *IEEE 802.1AE*. To je protokol podatkovnog sloja, koji se koristi unutar lokalne mreže tako da štiti vezu (eng. *link*) između mrežnih uređaja. Njime su izbjegnuti mnogi sigurnosni rizici na podatkovnom sloju, poput napada uskraćivanjem usluge (eng. *denial of service*), prisluškivanja (eng. *eavesdropping*) i upada (eng. *intrusion*). Kad je MACsec omogućen, osigurava se dvosmjerna sigurna veza između povezanih uređaja. Ta veza će biti uspostavljena nakon razmjene i verifikacije sigurnosnih ključeva. Prema standardu *IEEE 802.1AE*, samo autenticirani čvorovi se mogu pridružiti mreži, a osnovne usluge, odnosno sigurnosna svojstva koja MACsec protokol pruža su:

- **Integritet podataka bez povezivanja** (eng. *Connectionless data integrity*) osigurava da se okvir ne može neopaženo promijeniti.
- **Autentičnost podataka** (eng. *Data origin authenticity*) osigurava da se okvir prima od autenticiranog čvora u mreži.
- **Povjerljivost** (eng. *Confidentiality*) koja onemogućava prisluškivanje. To je osigurano time što se dio okvira s podacima koji se šalju šifrira.

- **Zaštita od ponovne reprodukcije** (eng. *Replay protection*) osigurava da se isti okvir ne može primiti više od jednog puta. Preciznije, napadač ne može neprimijećeno kopirani okvir ponovno poslati.
- **Ograničeno kašnjenje okvira** (eng. *Bounded receive delay*) koje osigurava da okvir ne može bit presretnut te da njegova isporuka ne može biti odgođena na više od nekoliko sekundi, bez da se to ne primijeti.

Kod ovog protokola, jedan od ključnih pojmova koji koristimo je **sigurno povezujuće udruženje** (eng. *Secure Connectivity Association*). Taj pojam kraće označavamo s CA, pa ćemo takav naziv koristiti dalje u radu. On je definiran protokolom o sporazumu ključeva (eng. *Key Agreement Protocol*). Njega kraće nazivamo MKA, što je skraćenica za *MACsec Key Agreement protocol*. CA čini potpuno (simetrično i tranzitivno) povezan podskup pristupnih točaka jedne lokalne mreže. Te pristupne točke su tada podržane MACsec protokolom. Njih nazivamo MACsec entiteti (eng. *MAC Security Entities*), i označavamo sa SecY, dok entitete sporazuma ključeva zovemo KaYs (eng. *Key Agreement Entities*). Svaki priključak koji može sudjelovati u CA, obuhvaća i SecY i KaYs entitete. Nakon što se uspostavi CA te obavi međusobna autentikacija, generiraju se i distribuiraju novi ključevi (poštujući pravila protokola o sporazumu ključeva - MKA), pomoću kojih entiteti uspostave **sigurnosno udruženje** (eng. *Secure Association*) ili kraće SA. SA shvaćamo kao sigurnosnu vezu koja omogućuje da se ispoštuju sva već navedena sigurnosna svojstva MACsec protokola pri prijenosu okvira podataka od jednog entiteta prema drugim entitetima. Svaki SA sadrži jedinstveni tajni ključ ili jedinstveni skup tajnih ključeva ukoliko kriptografske operacije koje koristimo pri zaštiti okvira podataka zahtijevaju više takvih ključeva. Takav ključ zovemo SA ključ, ili kraće SAK (eng. *Security Association Key*). Na Slici 3.3 vidimo primjer povezujućeg udruženja između tri entiteta (A,

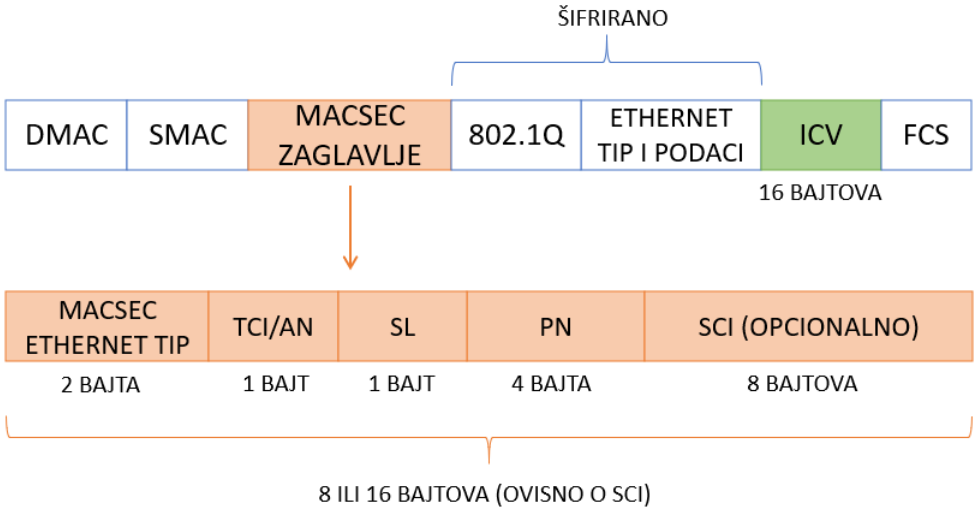


Slika 3.3: Sigurna komunikacija između tri MACsec entiteta

B i C), dok je jedan (D) izvan CA. Strelicama su prikazani i jednosmjerni sigurni kanali (eng. *Secure Channel*), koje kraće nazivamo SC. To je veza koja omogućava siguran prijenos okvira od jednog entiteta k ostalim unutar CA. SC je podržan nizom SA-ova, od kojih svaki ima svoj SAK, pa je dopušteno periodično korištenje novih ključeva bez prekida veze. Svaki SA je identificiran SA identifikatorom (eng. *Secure Association Identifier*), koji se sastoji od SC identifikatora (eng. *Secure Channel Identifier*) zajedno s brojem udruženja (eng. *Association Number*), kojeg označavamo s AN. SC identifikator se pritom sastoji od MAC adrese i identifikatora priključka. SA identifikator je potreban SecY entitetu koji je primatelj da prepozna SA te pripadni SA ključ, kako bi mogao dekriptirati i autenticirati poslani okvir podataka. Generiranje i podjela SA ključeva (SAK) SecY entitetima unutar CA, je povjerena MKA protokolu.

3.4.1 Izgled okvira kod MACsec protokola

Već smo spomenuli SA identifikator koji je opcionalni dio MACsec zaglavlja, a koji je dio okvira podataka pri MACsec protokolu. Na Slici 3.4 možemo vidjeti izgled okvira podataka kod MACsec protokola. Na standardni okvir



Slika 3.4: Format MACsec okvira

podataka u podatkovnom sloju OSI modela, dodajemo MACsec zaglavlje te polje koje zovemo **vrijednost provjere integriteta** (eng. *Integrity check value*), koje kraće označavamo s ICV. Vrijednost koja se sprema u polje ICV, je zapravo autentikacijska oznaka T, koju smo izračunali pomoću GCM-AES, na način prikazan na Slici 2.3. MACsec zaglavlje se sastoji od sljedećih polja:

- **MACsec Ethernet tip:** To je polje veličine 2 bajta koje govori da je ono što slijedi nakon njega, MACsec okvir. On ima vrijednost $(88E5)_{16}$.
- **TCI/AN:** Oznaka podatka o kontroli (eng. *Tag Control Information*), ili kraće TCI, zajedno s brojem udruženja (eng. *Association Number*), koji se kraće označava s AN, je polje duljine jednog bajta koje označava broj verzije MACsec-a.

- **SL:** Sljedeći bajt je polje u kojem je postavljena duljina šifriranih podataka. Označujemo ga sa SL (eng. *Short Length*).
- **PN:** Sljedeća četiri bajta nazivamo broj paketa (eng. *Packet Number*), ili kraće PN. Koristi se za zaštitu od ponovne reprodukcije i generiranje vektora inicijalizacije (zajedno s identifikatorom sigurnog kanala).
- **SCI:** Posljednjih osam bajtova su opcionalni i predstavljaju identifikator sigurnog kanala, odnosno kraće SC identifikator.

MACsec koristi GCM-AES za autenticirano šifriranje, te GMAC ako je potrebna samo autentikacija, bez šifriranja. Šifriraju se sva polja nakon MACsec zaglavlja, uključujući i VLAN oznaku (802.1Q) iz originalnog Ethernet okvira. Polja prije MACsec zaglavlja (DMAC i SMAC) koja predstavljaju MAC adresu primatelja i pošiljatelja (eng. *Destination MAC Address, Source MAC Address*), se ne šifriraju. U originalnom Ethernet okviru postoji i još jedno polje veličine 4 bajta, FCS (eng. *Frame Check Sequence*), kojim okvir završava, a koje je zaduženo za cikličku provjeru redundancije. To polje se ne šifrira MACsec protokolom, te se samo navede nakon ICV polja.

3.4.2 Paketi šifriranja

Paket šifriranja (eng. *Cipher suite*) je općenito specifikacija kriptografskih algoritama zajedno s vrijednostima parametara (poput duljine ključa) koje će koristiti upravo ti algoritmi. MACsec koristi GCM-AES za provjeru autentičnosti i za šifriranje, s GCM-AES-128 i GCM-AES-256 programskim paketima za šifriranje, odnosno tim skupovima šifri. Dva dodatna paketa šifriranja, GCM-AES-XPN-128 i GCM-AES-XPN-256, dodana su za podršku proširenim brojevima paketa (eng. *extended packet numbers*) za postizanje

većih Ethernet brzina. U drugom poglavlju ovog rada, GCM-AES-128 je opisan kao uobičajeno, odnosno standardno zadani paket šifriranja koji koristi AES-128 simetričnu blok šifru. Vidjeli smo kako duljina tajnog ključa K, kod tog paketa šifriranja, odgovara veličini odgovarajućeg AES bloka. Analogno zaključujemo kako kod GCM-AES-256 paketa šifriranja, duljina tajnog ključa K iznosi 256 bitova.

Literatura

[1] Christof Paar, Jan Pelzl. *Understanding Cryptography* Springer-Verlag, 2010

[2] Andrej Dujella, Marcel Maretić, *Kriptografija*, Element, 2007

[3] Andrej Dujella, *Uvod u teoriju brojeva, nastavni materijal*, PMF - Matematički odjel, Sveučilište u Zagrebu

[4] Saša Krešić Jurić, *Algebarske strukture, nastavni materijal*, Odjel za matematiku, Prirodoslovno-matematički fakultet Split, 2013

[5] Gordan Radobolja, *Osnovne algebarske strukture, nastavni materijal*, Prirodoslovno-matematički fakultet Split, 2021

[6] Aaron Landesman, *Notes on finite fields*, https://web.stanford.edu/ aaronlan/assets/finite-fields.pdf

[7] Avinash Kak, *Computer and Network Security*, https://engineering.purdue.edu/kak/compsec/NewLectures/

[8] Dobre Blazhevski, Adrijan Bozhinovski, Biljana Stojchevska, Veno Pachovski, *Modes of Operation of the AES Algorithm*, University American College Skopje, 2013

[9] David A. McGrew, John Viega, *The Galois/Counter Mode of Operation (GCM)*

[10] Morris Dworkin, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*

[11] Ante Burilović, *Računalne mreže, predavanja*, https://mapmf.pmfst.unist.hr/ aburilovic/nastava.html

[12] IEEE, *IEEE 802.1AE-2018, Media Access Control (MAC) Security*

TEMELJNA DOKUMENTACIJSKA KARTICA

PRIRODOSLOVNO–MATEMATIČKI FAKULTET
SVEUČILIŠTA U SPLITU
ODJEL ZA MATEMATIKU

DIPLOMSKI RAD

MATEMATIČKA POZADINA MACSEC PROTOKOLA

Josip Buklijaš

Sažetak:

U ovom radu je opisan algoritam MACsec protokola koji se koristi za sigurnu razmjenu podataka na lokalnoj računalnoj mreži. Taj algoritam koristi Galoisov način brojača AES kriptosustava za rad. Dokazani su osnovni teoremi koji su služili za konstrukciju konačnih polja u kakvim se obavljaju operacije algoritma AES kriptosustava. Operacije šifriranja i dešifriranja, koje su osnovne operacije kod Galoisovog načina brojača su detaljno opisane.

Ključne riječi:

Konačna polja, AES kriptosustav, Galoisov način brojača, lokalna mreža.

Podatci o radu:

broj stranica: 76, broj slika: 8, broj tablica: 9, jezik izvornika: hrvatski

Mentorica: *doc. dr. sc. Marija Bliznac Trebješanin*

Članovi povjerenstva:

doc. dr. sc. Gordan Radobolja

Jelena Pleština, mag. math

Povjerenstvo za diplomski rad je prihvatilo ovaj rad 22.09.2021.

TEMELJNA DOKUMENTACIJSKA KARTICA

FACULTY OF SCIENCE, UNIVERSITY OF SPLIT
DEPARTMENT OF MATHEMATICS

MASTER'S THESIS

THE MATHEMATICS BEHIND THE MACSEC PROTOCOL

Josip Buklijaš

Abstract:

This paper describes the MACsec protocol algorithm used for secure data exchange on a local computer network. This algorithm uses the Galois/Counter Mode of the AES cryptosystem to operate. The basic theorems that served for the construction of finite fields in which the operations of the AES cryptosystem algorithm are performed, are proved. Encryption and decryption operations, which are the basic operations of the Galois/Counter Mode, are described in detail.

Key words:

Finite Fields, AES cryptosystem, Galois/Counter Mode, Local Area Network.

Specifications:

number of pages: 76, number of pictures: 8, number of tables: 9, original language: Croatian)

Mentor: *assistant prof. Marija Bliznac Trebješanin*

Committee:

assistant prof. Gordan Radobolja

Jelena Pleština, mag. math

This thesis was approved by a Thesis commettee on 22.09.2021.