



Mladen Veinović
Saša Adamović

KRIPTOLOGIJA I

OSNOVE ZA ANALIZU I SINTEZU ŠIFARSKIH SISTEMA

UNIVERZITET SINGIDUNUM

Fakultet za informatiku i računarstvo

Mladen Veinović
Saša Adamović

KRIPTOLOGIJA I

**Osnove za analizu i sintezu
šifarskih sistema**

Treće izdanje

Beograd, 2018.

KRIPTOLOGIJA I - Osnove za analizu i sintezu šifarskih sistema

Autori:

dr Mladen Veinović
dr Saša Adamović

Recenzenti:

dr Milan Milosavlejić
dr Branko Kovačević
dr Aleksandar Jevremović

Izdavač:

UNIVERZITET SINGIDUNUM
Beograd, Danijelova 32
www.singidunum.ac.rs

Za izdavača:

dr Milovan Stanišić

Lektura:

Lidija Mandić

Tehnička obrada:

Saša Adamović

Dizajn korica:

Aleksandar Mihajlović

Godina izdanja:

2018.

Tiraž:

500 primeraka

Štampa:

Mobid, Loznica

ISBN: 978-86-7912-576-7

Copyright:

© 2018. Univerzitet Singidunum

Izdavač zadržava sva prava.

Reprodukcija pojedinih delova ili celine ove publikacije nije dozvoljeno.

Fysis kriptesthai filei - priroda voli da se skriva

Heraklit, Fragment 123.*

(Razumevanje prirode nikad nije lako, ali razumevanje je ipak moguće. Pomoću razumevanja se na osnovu postojećih znanja može stvarati novo, odnosno ljudi koji razumeju mogu preduzimati korisne aktivnosti zato što mogu stvoriti novo znanje ili barem nove informacije na osnovu već postojećeg znanja.)

¹ Preuzeto iz predavanja Prof. dr Milana Milosavljevića, Univerzitet Singidunum, Beograd.

1.	UVOD.....	11
1.1.	Ciljevi i pravila	11
1.1.1.	Simetrični modeli	14
1.1.2.	Asimetrični modeli	17
1.1.3.	Rezime.....	19
1.2.	Ostali kriptološki ciljevi.....	19
1.2.1.	Generatori slučajnih i pseudoslučajnih nizova.....	20
1.2.2.	Autorizacija	21
1.2.3.	Protokoli autentifikacije	21
1.3.	Praktični kriptološki problemi.....	22
1.3.1.	Protokoli, strane koje komuniciraju i protivnici.....	22
1.3.2.	Kriptologija i bezbednost računara	23
1.3.3.	Pravila igre.....	24
1.4.	Pristupi izučavanju kriptologije	25
1.4.1.	Faze u razvoju kriptologije	25
1.4.2.	Razvoj kriptologije kroz kriptanalizu	25
1.4.3.	Šenonova sigurnost simetričnog šifrovanja	26
1.4.4.	Teorija računarske kompleksnosti	27
1.4.5.	Jednosmerne funkcije	28
2.	KLASIČNA KRIPTOGRAFIJA.....	30
2.1.	Šifre transpozicije	30
2.1.1.	Transpozicija kolona.....	31
2.1.2.	Kriptoanaliza transpozicija kolona	33
2.1.3.	Dvostruka transpozicija kolona	33
2.1.4.	Kriptoanaliza dvostruke transpozicije	34

2.2. Šifre zamene - supstitucije	35
2.2.1. Šifre proste zamene (monoalfabetske).....	35
2.2.2. Kriptoanaliza Cezarove šifre.....	37
2.2.3. Homofone šifre	39
2.2.4. Kriptoanaliza homofone šifre.....	40
2.3. Poligramske šifre.....	41
2.3.1. Plejfer šifra	41
2.3.2. Kriptoanaliza Plejfer šifre	42
2.3.3. Hill-ova šifra	44
2.3.4. Kriptoanaliza Hilove šifre	47
2.4. Polialfabetske zamene	47
2.4.1. Kriptoanaliza Vižnerove šifre	49
2.5. Perfektne šifre.....	52
2.5.1. One-time pad	56
2.6. Kodne knjige	59
3. SIMETRIČNI ŠIFARSKI SISTEMI.....	61
3.1. Sekvencijalne šifre.....	63
3.1.1. A5/1.....	69
3.1.2. RC4	73
3.2. Blokove šifre	77
3.2.1. Fejstel šifra	77
3.2.2. DES	80
3.2.3. Trostruki DES.....	94
3.2.4. AES	98
3.2.5. Još tri blokove šifre	105
3.2.6. TEA	106
3.3. Režimi rada blokovekih šifara	108

3.4. Integritet.....	114
3.5. Kratak pregled.....	117
4. ASIMETRIČNI ŠIFARSKI SISTEMI	118
4.1. Uloga javnog ključa u razmeni simetričnih ključeva	120
4.2. Difi-Helman (<i>Diffie-Hellman</i>)	122
4.3. RSA	127
4.4. Primena asimetričnih šifarskih sistema.....	135
4.4.1. RSA šifrovanje i dešifrovanje.....	136
4.4.2. RSA digitalno potpisivanje.....	137
4.4.3. Tajnost i neporecivost.....	140
4.4.4. Infrastruktura javnih ključeva	142
4.4.5. Prednosti i nedostaci kriptografije sa javnim ključevima.....	146
5. HEŠ FUNKCIJE.....	151
5.1. Svojstva sigurnih heš funkcija.....	151
5.2. Rođendanski problem	154
5.3. Klasične sume – nekriptografske heš funkcije	157
5.4. MD5	158
5.5. SHA	161
5.6. Heš funkcija i integritet	164
6. AUTENTIFIKACIJA	168

6.1. Lozinke	169
6.1.1. Ključevi ili lozinke	170
6.1.2. Izbor lozinke	171
6.1.3. Napadi preko lozinke	172
6.1.4. Verifikacija lozinke	173
6.1.5. Drugi problemi sa lozinkama.....	174
6.2. Biometrija	175
6.2.1. Iris kao biometrijski podatak.....	177
6.2.2. Biometrijska autentifikacija	182
6.2.3. Nedostaci biometrijskih sistema	184
6.2.4. Zaštita privatnost biometrijskih podataka	186
7. VIZUELNA KRIPTOGRAFIJA I SKRIVANJE INFORMACIJA	192
7.1. Vizuelna kriptografija	192
7.2. Steganografija	197
7.2.1. Istorijski osvrt na nastanak steganografije	197
7.2.2. Moderna steganografija.....	198
7.2.3. Digitalni vodeni pečat	199
7.2.4. LSB supstitucija	201
7.3. Generisanje slučajnih brojeva.....	207
7.3.1. Vrste generatora	207
7.3.2. Razlika PRNG / TRNG.....	209
7.3.3. Generatori slučajnih brojeva.....	210
7.3.4. Naknadno procesiranje ekstrakta entropije izvora.....	212
8. BIBLIOGRAFIJA	214
DODATAK A – MODULARNA ARITMETIKA	216

DODATAK B – EDUKATIVNO RAZVOJNO OKRUŽENJE CRYPTOOL.....	232
REČNIK POJMOVA.....	237



Predgovor

Kriptologija I – je udžbenik koji je namenjen studentima Fakulteta za informatiku i računarstvo Univerziteta Singidunum za pripremu ispita iz predmeta „Kriptologija 1“, a može se koristiti i za savladavanje gradiva iz predmeta u kojima se koriste kriptološki mehanizmi za zaštitu podataka. Udžbenik sadrži uvodni materijal u oblast informacione bezbednosti.

Kriptologija je nauka o zaštiti podataka koja je dugo vremena bila orijentisana samo na profesionalne sisteme za zaštitu podataka (vojska, policija, bezbednosne agencije i Ministarstvo spoljnih poslova). Od kada postoji komunikacija, postoji i potreba da se podaci koji se prenose zaštite od napadača, tj. od onih kojima podaci nisu namenjeni. Od vremena komercijalizacije Interneta, kada je on počeo da se koristi kao infrastruktura za poslovanje, kriptologija je našla svoju primenu u svim aplikacijama koje se odnose na elektronsko poslovanje: e-trgovina, e-bankarstvo, *e-government* i sl. Isto se može reći i za aplikacije za mobilne uređaje. Poverljivost (tj. tajnost) podataka koji se prenose je samo jedan od servisa koji se zahteva. Slobodno se može reći da su danas jednako važni i servisi za autentifikaciju, integritet podataka, dostupnost i neporecivost.

U udžbeniku se razmatraju perfektni i primenjeni šifarski sistemi. Perfektni ili apsolutno tajni šifarski sistemi se zasnivaju na šifrovanju primenom pravih slučajnih nizova. Mana ovakvih sistema je u tome što se zahteva da dužina slučajnog niza mora da odgovara dužini otvorene poruke koja se šifruje. U računarskim aplikacijama, naročito u onim koje su namenjene za rad u realnom vremenu, navedeni koncept nije primenljiv u praksi. Zbog toga postoje računarski sigurni sistemi koji se zasnivaju na šifarskim algoritmima i šifarskim ključevima. U njihovoj osnovi su generatori pseudoslučajnih nizova, koji obezbeđuju dobra statistička svojstva niza za šifrovanje, a koji se inicijalizuje na bazi ključeva relativno male dužine.

Prvi deo udžbenika se odnosi na istorijske šifre, tj. one koje danas imaju samo istorijski značaj, ali predstavljaju odličan uvod u moderne šifre. To su šifre zamene i

šifre premeštanja koje su se razvijale i koristile do Drugog svetskog rata. Razmatranja počinju sa Cezarovom šifrom, preko Vižnerovih tablica, kodnih knjiga, pa sve do elektromehaničkih uređaja za šifrovanje. Najpoznatija mašina za šifrovanje iz tog vremena je dobro poznata "Enigma", o kojoj su se potpunije informacije mogle saznati sa distance od 30 do 40 godina.

Zatim se izučavaju simetrični i asimetrični šifarski sistemi sa stanovišta moderne kriptografije. Navedena podela se odnosi na kriptološke ključeve. U simetričnim sistemima se na obe strane komunikacije, pored algoritma šifrovanja, koristi i tajni simetrični ključ kojim se inicijalizuje algoritam. Tajnost simetričnih sistema se zasniva na tajnosti ključa. Bez poznavanja tajnog ključa, najbolji mogući napad na ovakve sisteme bi bila pretraga po svim mogućim ključevima. Najpoznatiji algoritmi iz ove klase su DES, 3DES, AES. Asimetrični šifarski sistemi su mlađi od simetričnih. Pojavili su se sredinom sedamdesetih godina prošlog veka. Nastali su kao odgovor na problem distribucije tajnih simetričnih ključeva. U asimetričnim sistemima jedan ključ se koristi za šifrovanje (javni ključ), a drugi za dešifrovanje (privatni ključ). Ova dva ključa su u jedinstvenoj vezi, ali se iz jednog ne može odrediti drugi. Cela teorija počiva na jednosmernim funkcijama koje se lako mogu izračunati u jednom smeru. Za inverznu transformaciju su potrebni dodatni podaci. Tajnost ovakvih sistema počiva na složenosti izračunavanja nekih teških matematičkih problema. Najpoznatiji algoritam iz ove klase je RSA. Asimetrični sistemi su pogodni za realizaciju dodatnih funkcionalnosti kod savremenih aplikacija, a odnose se na servis za autentifikaciju, integritet podataka i neporecivost. Za ove servise primenjuju se tehnike digitalnog potpisivanja nad heš vrednostima otvorenih poruka.

Teorijski materijali su podržani praktičnim primerima koji su realizovani u namenskom softverskom paketu CRYPTOTOOL. To je softver otvorenog koda preko koga se mogu na efikasan način proveriti svi koncepti iz kriptologije. U njemu postoje gotove biblioteke šifarskih algoritama i dodatnih kriptološki funkcija. Zapravo, već posle savladavanja radnog okruženja, moguće je realizovati jedan šifarski sistem.

Cilj ovog predmeta jeste da studenti savladaju koncepte modernih šifarskih sistema. Ovi koncepti nalaze svoju primenu u savremenim IT tehnologijama: u softveru, operativnim sistemima, računarskim mrežama, bazama podataka i Web aplikacijama,

posebno u poslovnim sistemima. U praksi se standardno koriste višeslojna rešenja zaštite. Na tržištu postoji velika potreba za kadrovima sa znanjima iz kriptologije. To nisu klasični informatičari, programeri, administratori, već oni koji mogu da na ispravan i potpun način razumeju sve bezbednosne aspekte. Svaka organizacija ima potrebu da brine o bezbednosti svojih podataka. U današnjim uslovima, upravo su sistematska znanja iz zaštite komparativna prednost i osnova za kvalitetnije zapošljavanje.

Jedan od osnovnih problema pri učenju kriptologije predstavlja njena kompleksnost i zasnovanost na složenim matematičkim principima i formulama. Najveći broj bezbednosnih rešenja danas ne zahteva detaljno poznavanje pomenutih matematičkih principa već samo njihovo osnovno razumevanje i povezivanje sa ulogom u praktičnim rešenjima. Prebacivanjem fokusa iz domena matematičkih osnova bezbednosnih rešenja u domen praktične primene kod savremenih informacionih i telekomunikacionih tehnologija značajno se menja i profil studenata u ovoj oblasti. Iz tog razloga je ovo preusmeravanje neophodno podržati odgovarajućim edukacionim pristupima, materijalima i uslovima za učenje.

Naša iskustva sa korišćenjem tradicionalnog pristupa kod podučavanja kriptologije ukazala su na potrebu za novim, interaktivnijim i kolaborativnijim načinima za sticanje znanja iz ove oblasti. Umesto korišćenja *bottom-up* pristupa - kretanja od potrebnih matematičkih aparata ka njihovoj kriptografskoj primeni, odlučili smo se za *top-down* pristup: predstavljanje kriptoloških principa u kontekstu njihove praktične primene u cilju izazivanja interesovanja za savladavanje njihovih pratećih matematičkih principa.

Na kraju, zahvaljujemo se recenzentima na korisnim sugestijama koje su unete u udžbenik. Posebnu zahvalnost dugujemo redovnom profesoru dr Milanu Milosavljeviću, osnivaču moderne kriptologije na našim prostorima i šire, koji nam je svojim iskustvom, savetima i prisustvom pomagao i bio inspiracija u toku pisanja ovog udžbenika. Ovo je prvo izdanje, te su sve kritike i sugestije čitaoca dobro došle.

1. UVOD

Kroz istoriju je postojala potreba da se omogući komunikacija između dve ili više strana, a da se zadrži privatnost informacija koje se prenose, naročito u slučaju kada postoji prisluškivanje komunikacionog kanala. Sa tom potrebom je nastala kriptografija (eng. *Cryptography*²). Dok je obezbeđivanje privatnosti ostao glavni cilj, polje kriptografije je prošireno na niz drugih oblasti, ne samo u okviru bezbednosti komunikacija, kao što su integritet i autentičnost komunikacija, već i na mnoge druge sofisticiranije ciljeve.

Danas je primena kriptografije široko rasprostranjena iako je nekada bila primenljivana uglavnom u vojnim okvirima. Na primer, pri kupovini preko Interneta upravo se kriptografija koristi da bi se osigurala privatnost broja platne kartice dok se prenosi od korisnika do servera prodavnice. U elektronskom bankarstvu se kriptografija koristi da bi se obezbedio pristup ličnim podacima i transakcijama.

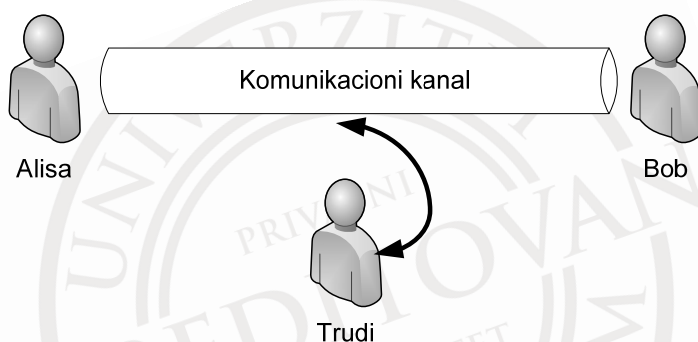
Kriptografija se koristi skoro od kada je pisanje izmišljeno. Većim delom svoje istorije, kriptografija je bila umetnost, igra maštovitih rešenja i napada. Iako je zadržala neke od svojih starih čari, poslednjih tridesetak godina je donelo nešto sasvim novo. Umetnost kriptografije je spojena sa naukom, a danas govorimo o modernoj kriptologiji. Danas je kriptologija temelj računarske i komunikacione tehnologije. Zasniva se na strogim matematičkim principima i spaja oblasti kao što su teorija brojeva, teorija računarske kompleksnosti i teorija verovatnoće.

1.1. CILJEVI I PRAVILA

Moderna kriptologija rešava veliki broj problema. Najvažniji od svih problema i dalje je ostvarivanje bezbedne komunikacije preko nesigurnih komunikacionih kanala. Da bismo lakše predstavili ovaj problem, definisaćemo određene pojmove: **pošiljaoc** (eng. *Sender* - *S*) i **primaoc** (eng. *Receiver* - *R*) koji žele da uspostave komunikaciju

² Kriptografija (ili kriptologija); nastala od grčke reči κρυπτός - "skriveno, tajno"; i γράφειν - "pisati", ili λογία - "nauka", respektivno)

među sobom. Često ćemo naći u radovima iz kriptologije da se za likove u primerima koriste imena Alisa i Bob. U kriptologiji se uvek razmatra i treća strana, a to je protivnik (eng. *Adversary* - A). Protivnik predstavlja izvor pretnji. Zamišljamo ga kao nekog ko ima pristup komunikacionom kanalu i želi da kompromituje bezbednost strana koje komuniciraju. Zbog toga je u upotrebi i treći lik, Trudi. Ljudski likovi se uvode zbog lakšeg razumevanje problema, a i suvoparna razmatranja postaju zabavnija. Međutim, to ne znači da su pošiljalac i primalac samo ljudska bića, već oni mogu biti i nešto drugo, kao na primer: mobilni telefon, računar ili neki proces na računaru.



Slika 1.1 Kriptologija teži dostizanju idealnog komunikacionog kanala između pošiljaoca primaoca

Idealni komunikacioni kanal. Zamislimo naše dve strane kako komuniciraju kroz neprobojnu i zvučno nepropustivu cev, kroz koju pošiljalac može da šapne poruku i primalac da je čuje. Niko drugi ne može da čuje njihov razgovor niti da izmeni bilo šta u njemu. Ova cev predstavlja savršeni komunikacioni kanal, dostupan samo pošiljaocu i primaocu, kao da su sami na svetu. Sa stanovišta bezbednosti ovaj komunikacioni kanal je savršen. Na žalost, u stvarnom životu ne postoje idealni komunikacioni kanali između strana koje bi želele da komuniciraju. Najčešće se takve komunikacije obavljaju preko javnih mreža, kao što je Internet. Osnovni cilj kriptologije jeste da omogući komunikaciju koja ima slična svojstva sa idealnim komunikacionim kanalima.

Svi aspekti idealnog kanala se ne mogu dostići, tako da kriptolozi imaju još neke ciljeve kojima konstantno žele da se približe. Prvi cilj je **privatnost**. Obezbediti

privatnost znači sakriti sadržaj komunikacije od protivnika. Drugi cilj je **autentičnost i integritet**. Želimo da primalac, nakon što primi poruku od pošiljaoca, ima mogućnost da proveri da li je poruka zaista stigla od pošiljaoca, a ne od protivnika i da niko nije promenio poruku na njenom putu od pošiljaoca do primaoca.

Protokoli. U cilju obezbeđivanja servisa privatnosti i autentičnosti, definisani su odgovarajući kriptološki protokoli. Protokol je skup pravila koja razumeju i poštuju strane u komunikaciji. Kriptološkim protokolima se vrši izbor algoritama po kojima se vrši šifrovanje/dešifrovanje, dogovaraju se ključevi i usaglašavaju drugi kriptološki parametri. Nije teško zaključiti da, u želji da osiguraju svoju komunikaciju, strane moraju znati nešto ili biti u stanju da urade nešto što treća strana (napadač) ne zna ili nije u stanju da uradi. Zapravo, mora postojati određena asimetrija između situacija u kojima se nalaze strane koje komuniciraju i situacije u kojoj se nalazi protivnik.

U tekstu će se koristiti izrazi koji su formalno definisani.

Pismo ili alfabet (eng. *alphabet*) je konačan skup. Obično koristimo grčko slovo Σ za označavanje pisma. Na primer $\Sigma = \{0,1,2,3,4,5,6,7,8,9\}$ je pismo koje se sastoji od deset karaktera, a $\Sigma = \{0,1\}$ je pismo koje se sastoji od dva karaktera i ono se zove binarno pismo. Niz (eng. *string*) je konačan niz karaktera. Broj karaktera u nizu se zove dužina (eng. *length*) niza. Dužina niza X se označava sa $|X|$. Znači da niz $X = 1011$ ima dužinu četiri i da je u binarnom pismu. Niz $Y = \text{kriptografija}$ je dužine 14 i napisan je srpskim pismom. Niz dužine nula se naziva "prazan niz" i označava se sa φ . Ako su X i Y nizovi, onda je njihov spoj koji se obeležava sa $X||Y$ jednak karakterima niza X na koje se nadovezuju karakteri niza Y . Na primer, $1111||0010 = 111110010$. Skoro da se sve može predstaviti nekim nizom. Ovo je posebno bitno, zbog toga što se u računarskim tehnologijama sve predstavlja binarnim nizovima. Ako je a broj a X niz onda je $Y = (a, X)$ neki niz koji kodira broj a i X . Nije teško da ako imamo a i X , izračunamo Y . Obrnuto, ako imamo samo Y teško se može izračunati a i X . Jezik (eng. *language*) je grupa nizova koji svi nastaju iz jednog pisma. Ako je Σ pismo onda Σ^* predstavlja sve nizove čiji karakteri potiču iz Σ . Na primer $\{0,1\}^* = \{\varphi, 01,000,110,1,0,0111,\dots\}$.

Slika 1.2 Definicije nekih osnovnih pojmova

U kriptologiji postoje dva osnovna bezbednosna modela, a to su: simetrični (eng. *symmetric*) ili model sa tajnim ključevima i asimetrični (eng. *asymmetric*) ili model sa javnim i privatnim ključevima. Moderni kriptološki sistemi su najčešće kombinovani (hibridni) u cilju postizanja kompletne zaštite, najčešće u više nivoa.

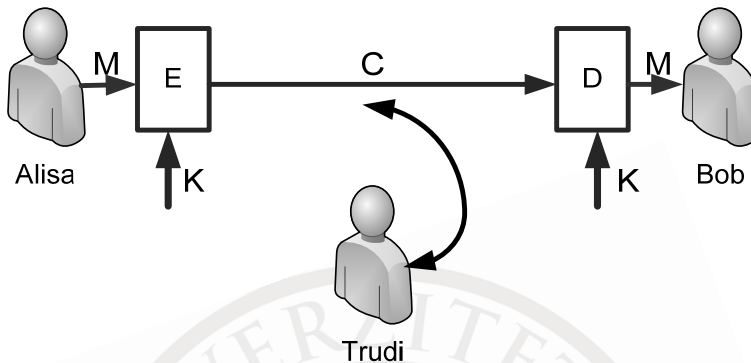
1.1.1. SIMETRIČNI MODELI

U praksi, najjednostavniji i najčešći slučaj je kada pošiljalac i primalac imaju jedan ključ koji protivnik nema. Ovaj model se naziva simetrični model. Cela teorija ovih modela se zasniva na jednom ključu koji protivnik ne sme da ima. Ključ je uglavnom slučajni niz dužine K bitova. Pošiljalac i primalac moraju nekako da iskoriste ključ da bi zaštitili svoju komunikaciju od protivnika. Problem razmene kriptoloških ključeva za simetrične šifarske sisteme, kao i postupci za ugovaranje tajnog simetričnog ključa, jesu predmet razmatranja u ovom udžbeniku. U svim šifarskim sistemima, posebna pažnja se posvećuje generisanju i distribuciji kriptoloških ključeva, što zahteva organizacione mere, tj. sprovođenje specifičnih "politika" zaštite. Uspešnost šifrovanja i dešifrovanja zahteva sprovođenje postupaka za inicijalizovanje na predajnoj i prijemnoj strani, koji se jednom rečju zovu kriptološka sinhronizacija.

U kriptologiji se pretpostavlja da je ključ na sigurnom i da ga znaju samo strane koje žele da komuniciraju. Ako je na primer ključ čuvan na nekom računaru, pretpostavlja se da je taj računar zaštićen od domašaja napadača, a zaštita takvog sistema je posao sistema za zaštitu računara. Zbog visokotehnoloških rešenja hardvera, operativnih sistema i samog Interneta, često se izvlači zaključak da su jedino sigurni izolovani računari, tj. računari koji nisu povezani na javne mreže.

Simetrični šifarski sistemi. Protokol koji se koristi u simetričnim sistemima se naziva šema simetričnog šifrovanja. Ovakva šema Π se može označiti sa $\Pi = (K, E, D)$. Sa E je označen algoritam šifrovanja (eng. *encryption algorithm*). Poruka M koju pošiljalac želi da pošalje se najčešće naziva otvoreni tekst (eng. *plaintext*). Pošiljalac šifrue (eng. *encrypts*) otvoreni tekst kombinovanjem ključa K , algoritma šifrovanja E i otvorenog teksta M . Tako nastali tekst naziva se šifrat (eng. *ciphertext*) C . Algoritam D predstavlja algoritam dešifrovanja (eng. *decryption algorithm*). Primalac primenjuje algoritam D sa ključem K na šifrat C . Nakon uspešnog dešifrovanja dobija se prvobitno šifrovana poruka M , tj. otvoreni tekst. Najčešće, ključ je slučajni niz

odgovarajuće dužine. Algoritmi šifrovanja su predmet standardizacije (DES, 3DES, AES i sl.) i generišu se prema strogim matematičkim principima.



Slika 1.3 Simetrično šifrovanje. Pošiljalac i primalac imaju isti ključ, K . Protivnik nema ključ. M je otvoreni tekst, a C je šifrat.

Šema simetričnog šifrovanja je dobro poznata. Napadač može da poznaje model šifarskog sistema, da zna algoritam po kojem je vršeno šifrovanje, ima na raspolaganju šifrat, ali bez poznavanja ključa K nije u stanju da otkrije poruku M koja se prenosi. Razlikovaćemo apsolutno tajne simetrične šifarske sisteme u kojima ne postoji mogućnost dešifrovanja šifrata C i otkrivanja poruke M . U praktičnoj upotrebi su računarski sigurni simetrični šifarski sistemi kod kojih nije moguće u razumnom vremenu doći do otvorene poruke M , odnosno najbolji mogući put za dešifrovanje je isprobavanje svih mogućih vrednosti ključa K , što se danas smatra teškim problemom za realizaciju u realnom vremenu, ukoliko je ključ dovoljne dužine.

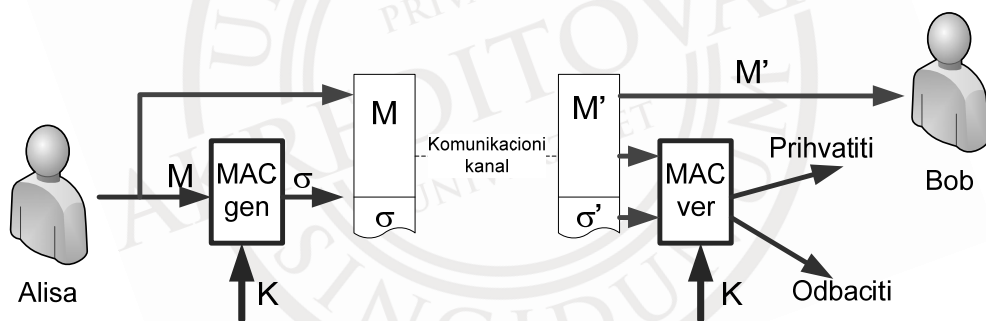
Šta je privatnost? Cilj šeme simetričnog šifrovanja je da spreči protivnika, koji poznaje šifrat da sazna otvoreni tekst. Dobijanje otvorenog teksta bi moglo da bude ostvareno u 2^n pokušaja, što nije loše ako je n neki mali broj. Ovo nam objašnjava jednu novu teoriju, da šema šifrovanja nije sigurna niti nesigurna, već da postoji određena verovatnoća da ona bude razbijena.

Postoji još faktora koji bi mogli uticati na sigurnost komunikacije. Na primer, da protivnik već zna nešto o poruci i pre nego što je poslata. Recimo da Alisa i Bob žele da kupe neke akcije, i da poruka koju žele da razmene može biti "kupi" ili "nemoj da

kupiš“, 1 ili 0. To znači da protivnik ima 50% šansi da pogodi poruku. Treba imati na umu i ovakav slučaj.

Takođe, treba da budemo toga svesni da protivnik najčešće nije neko ko nema dovoljno znanja i ne ulaže dovoljno energije u napade. Uglavnom je protivnik neko ko je veoma moćan. Pored toga, u kriptologiji, preciznije u kriptanalizi poznat je jedan princip koji kaže da će napadač uložiti onoliko sredstava u napad, koliko vredi informacija do koje želi da dođe.

Autentičnost i integritet poruke. Poruka poslata primaocu od strane pošiljaoca često je poslata kroz nesigurni komunikacioni kanal (npr. kao što je Internet). Želimo da obezbedimo primaocu mogućnost da sa sigurnošću utvrdi da je poruka poslata od strane pošiljaoca, a ne od protivnika i da nije izmenjena prilikom prenosa. Najčešće korišćeni način za rešavanje problema autentičnosti poruke je primena koda za autentičnost poruke (eng. *message authentication code* - MAC).



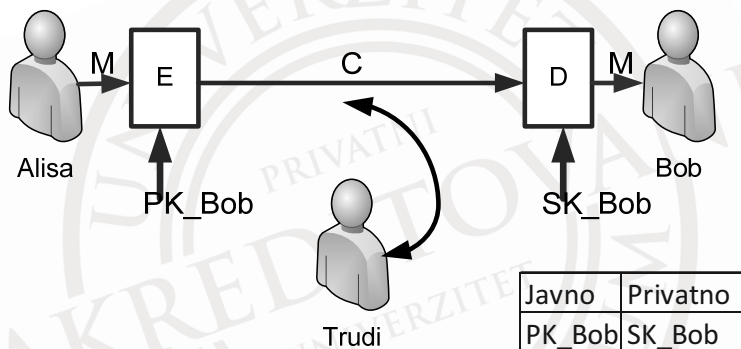
Slika 1.4 Kod autentičnosti poruke. Dodatak σ se kombinuje sa otvorenim tekstom M i primalac na osnovu njega odlučuje da li je poruka potekla od pošiljaoca sa kojim deli ključ K

Kada pošiljalac želi da pošalje poruku M , kreira se dodatak (eng. *hash*) koji se šifruje odgovarajućim algoritmom i ključem K . Dobijeni šifrovani dodatak dodaje se otvorenom tekstu M i šalje se poruka (M, σ) . Primalac dobija poruku koja može biti izmenjena zbog uticaja šuma u komunikacionom kanalu (M', σ') . Verifikacija se sprovodi nad dobijenom porukom primenom istog algoritma za dešifrovanje i ključa K . Ako je izlaz verifikacionog algoritma 1, on prihvata poruku kao autentičnu, a ako nije, on poruku smatra kao prevaru. Njegova reakcija se može ogledati u ignorisanju

poruke, preko obaveštavanja pošiljaoca o prevari, do prekidanja komunikacije. Verifikaciona šema počiva na deljenom tajnom ključu K koga poseduju samo pošiljalac i primalac.

1.1.2. ASIMETRIČNI MODELI

Ključ za koji znaju samo strane koje komuniciraju i njegova tajnost, nije jedini način čuvanja privatnosti komunikacije. Asimetrični šifarski sistem, poznat i kao model sa javnim ključem, zasnovan je na postojanju dva ključa: javnog (eng. *public key* – PK) i tajnog ključa (eng. *secret key* - SK). Javni ključ je poznat javnosti i može biti objavljen.



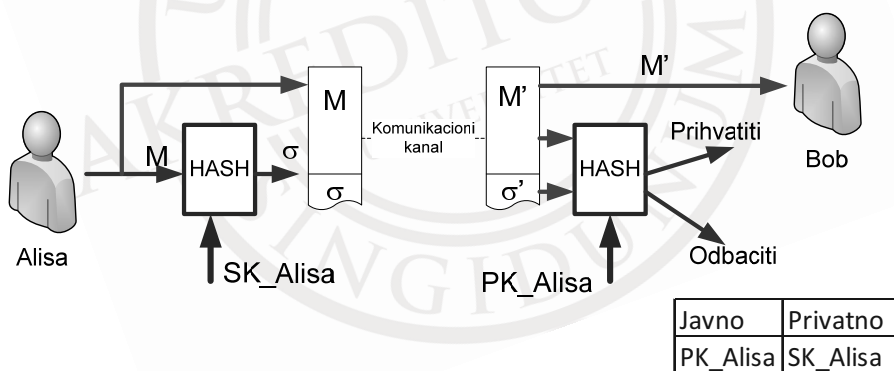
Slika 1.5 Primer asimetričnog modela.

Asimetrično šifrovanje. Asimetrično šifrovanje se zasniva na primeni specifičnih algoritama za šifrovanje i dešifrovanje i primeni dva ključa koja su međusobno povezana (javni i privatni ključ). Pretpostavlja se da pošiljalac ima kopiju PK_{Bob} javnog ključa primaoca. Takođe, zbog njegove javnosti smatramo da i napadač ima taj isti ključ. Da bi poslao tajnu poruku, pošiljalac šifrue otvoreni tekst M i pravi šifrat C na osnovu algoritma šifrovanja, $C = E(M, PK_{Bob})$. Takav šifrat se šalje primaocu koji ga dešifrue uz pomoć tajnog ključa, $M = D(C, SK_{Bob})$. Suština asimetričnih šifarskih sistema je u tome da se šifrovana poruka na bazi javnog ključa ne može dešifrovati na bazi istog ključa. Potreban je drugi par ovog ključa – privatni ključ.

Ideja kriptografije sa javnim ključem i sama mogućnost ostvarivanja ovakvog cilja je fantastična. Recimo da nikada nismo upoznali primaoca! Možemo mu poslati tajnu

poruku tako što ćemo potražiti neke informacije u javno dostupnoj bazi i šifrovati poruku na način tako da samo on može da je dešifruje. Tvorcima ideje o kriptografiji sa javnim ključem smatraju se Whitfield Diffie i Martin Hellman³.

Digitalni potpisi. Način za potvrđivanje autentičnosti i integriteta poruke u asimetričnom modelu jeste digitalni potpis (eng. *Digital signature*). U ovom slučaju, pošiljalac ima javni ključ PK_{Alisa} i odgovarajući tajni privatni ključ SK_{Alisa} . Pretpostavićemo da primalac zna javni ključ PK_{Alisa} koji pripada pošiljaocu, strani koja potpisuje. Takođe, smatramo i da protivnik ima javni ključ. Kada pošiljalac želi da pošalje poruku M , on njoj isto dodaje dodatne bitove, σ , koji se nazivaju potpis (eng. *signature*) te poruke. Tačnije, nastaje tako što se najpre izračuna sažetak poruke M (*hash*), a zatim se sažetak šifruje primenom asimetričnog algoritma na bazi privatnog ključa. Na prijemu, primalac određuje da li je poruka autentična na osnovu dobijenih M' i σ' i javnog ključa pošiljaoca PK_{Alisa} primenjujući ih na verifikacioni (eng. *verification*) algoritam. Ako je poruka prihvaćena, primalac je smatra kao autentičnu, ako nije, smatra je kao pokušaj prevare.



Slika 1.6 Šema digitalnog potpisivanja je određena na osnovu algoritma za generisanje ključa, algoritma potpisa i algoritma potvrde.

³ Whitfield Diffie i Martin Hellman, američki kriptolozi, jesu pioniri kriptografije sa javnim ključem. U naučnom radu objavljenom 1976. godine "*New Directions in Cryptography*" uveli su potpuno novu metodu za distribuciju kriptoloških ključeva.

Jedna od razlika između MAC i digitalnog potpisa je neporecivost (eng. *non-repudiation*). Sa MAC, svako ko može da potvrdi autentičnost poruke, kojoj je MAC dodeljen, može i da napravi takvu poruku. Na primer, tako zaštićena poruka ne bi mogla biti iskorišćena na sudu. Međutim, sa digitalnim potpisom jedino strana koja potpisuje može da napravi poruku koja se može verifikovati javnim ključem PK . I to se može iskoristiti kao dokaz na sudu.

1.1.3. REZIME

U nastojanju da što bolje oponašamo osobine savršenog kanala, postavljamo dva osnovna cilja: obezbeđivanje privatnosti poruke i obezbeđivanje autentičnosti i integriteta poruke. Dva osnovna načina za postizanje tih ciljeva jesu upravo simetrični i asimetrični šifarski sistemi.

Tabela 1: Rezime osnovnih osobina simetričnog i asimetričnog modela.

	Simetrični model	Asimetrični model
Privatnost poruke	Simetrično šifrovanje (tajni ključ)	Asimetrično šifrovanje (javni ključ strane primaoca)
Autentičnost i integritet poruke	Kod autentičnosti poruke (MAC)	Šema digitalnog potpisivanja (privatni ključ pošiljaoca)

1.2. OSTALI KRIPTOLOŠKI CILJEVI

U kriptologiji postoji mnogo ciljeva. Neki se odnose na one koje smo već opisali. Osnova za sve šifarske sisteme jesu generatori slučajnih nizova. Za praktične sisteme zaštite, cilj kriptologa je sinteza kvalitetnih generatora pseudoslučajnih nizova. Pri realizaciji teži se ka jednoznačnoj autentifikaciji strana u komunikaciji.

1.2.1. GENERATORI SLUČAJNIH I PSEUDOSLUČAJNIH NIZOVA

Slučajni impulsi se u stvarnosti javljaju samo u prirodnim pojavama i situacijama koje je nemoguće predvideti (na primer bacanje potpuno simetričnog novčića). U primeni se međutim upotrebljavaju hardverska i/ili programska rešenja koja dovode do toga da su dobijeni impulsi manje ili više slučajni. U praksi se pri proučavanju slučajnosti nekog niza impulsa koristi skup odabranih testova kojima se proverava posmatrani niz impulsa (za koji želimo ustanoviti da li je slučajan). Zaključak da je niz slučajan zapravo znači da niz pokazuje neke karakteristike koje se očekuju od niza slučajnih impulsa.⁴

Niz slučajnih impulsa generiše se od strane generatora slučajnih impulsa (eng. *random number generators*-RNG). Generatore slučajnih impulsa možemo generalno podeliti u dve kategorije: generatori istinski slučajnih impulsa (eng. *true random number generators*-TRNG) i generatori pseudoslučajnih impulsa (eng. *pseudorandom number generators*-PRNG). TRNG meri neku prirodnu pojavu koja je slučajna, pa su i rezultati njegovog rada stvarno slučajni impulsi. PRNG koristi računarske algoritme koji stvaraju niz prividno slučajnih impulsa, a koji su zapravo unapred određeni i zato se zovu pseudo-slučajni⁵. Jedna od najvažnijih karakteristika pseudoslučajnih generatora je periodičnost izlaznog niza. Zavisno od strukture algoritma i početnog stanja, perioda izlaznog niza je kraća ili duža, ali uvek konačna.

Generalno, gde je prioritet slučajnost, hardverski generatori istinski slučajnih impulsa su poželjniji od generatora pseudoslučajnih impulsa.

⁴ Na Internetu postoji skup javno dostupnih testova za ocenu slučajnosti generatora slučajnih i pseudoslučajnih nizova. Najkompletniji testovi su se mogu naći kod NIST-a (*National Institute of Standards and Technology*) - Američki nacionalni institut za standarde i tehnologiju.

⁵ Generatori slučajnih impulsa koji se mogu pozvati iz nekog programskog jezika najčešće su pseudoslučajni generatori. Za istu ulaznu inicijalnu sekvencu (*seed*) dobija se isti izlaz iz generatora.

1.2.2. AUTORIZACIJA

Uobičajeno je da dve strane žele da uspostave bezbednu komunikaciju. Pod bezbednom komunikacijom smatramo onu u kojoj su strane sigurne da komuniciraju jedna sa drugom i da je njihova komunikacija sakrivena od bilo koje treće strane. Jedan takav slučaj je, na primer, kada Alisa želi da se sa određene udaljenosti prijavi na svoj računar. Ili komunikacija između klijenta i servera u elektronskom bankarstvu, gde klijent npr. ima mogućnost da izvrši elektronsko plaćanje, tj. prenos sredstava sa svog računa na neki drugi račun. Po definiciji autorizacija se odnosi na dozvole pristupa pojedinim resursima sistema i dozvole za određene akcije nad resursima kojima se pristupa. U principu, pun pristup svim resursima sistema se ne daje svim korisnicima. Samo privilegovani korisnik, poput administratora, mogao bi dobiti dozvolu da instalira softver. Autorizacija je skup postupaka i tehnika za ograničavanje rada autentifikovanih korisnika.

1.2.3. PROTOKOLI AUTENTIFIKACIJE

Da bi se pristupilo većini Internet servisa, kao što su elektronska pošta, Internet bankarstvo, Internet kupovina i sl., potrebno je prvo dokazati da ste onaj za koga se predstavljate. Ovaj proces, prilikom koga se utvrđuje identitet, poznat je kao autentifikacija. Autentifikacija je proces utvrđivanja da li korisniku (osoba, računar, mašina, jednom rečju entitet) treba dozvoliti pristup sistemu. Razlikuju se problemi autentifikacije na lokalni ili udaljeni računar. Kod lokalne autentifikacije podrazumeva se da se u cilju potvrđivanja identiteta koristi nešto što znate (na primer lozinka), nešto što imate (na primer pametni telefon) ili neka jedinstvena osobina (na primer skener mrežnjače oka ili otisak prsta). Kod autentifikacije na udaljeni računar, podaci potrebni za autentifikaciju se prenose računarskom mrežom gde se primenjuju bezbednosni protokoli u cilju otklanjanja mogućnosti manipulacije.

Problem korišćenja samo lozinke prilikom autentifikacije je očigledan. U slučaju kompromitacije lozinke, napadač može da ostvari neovlašćeni pristup Internet računima i informacijama. Ako se koristi isto korisničko ime i lozinka za više Internet računova, šteta može biti još veća. U cilju bezbednije autentifikacije, danas se sve više uvode „snažnije“ metode koje zahtevaju korišćenje više od jednog faktora

autentifikacije. Očigledan primer dvostruke autentifikacije predstavlja korišćenje kreditne kartice prilikom podizanja novca sa bankomata. Da biste pristupili bankomatu potrebno je da posedujete nešto (svoju kreditnu karticu) i da nešto znate (svoj PIN). Ako neko ukrade vašu karticu, ne može je zloupotребiti sem ukoliko ne zna vaš PIN (upravo zbog toga je važno da ga nikad ne zapisujete na kartici).

Računarski protokoli su namenjeni za ostvarivanje bezbedne autentifikacije, kada se ona radi preko računarske mreže. U tu svrhu se koriste simetrični i asimetrični šifarski sistemi i heš funkcije. Dobro su poznati različiti bezbednosni protokoli koji imaju svoju namenu u slojevitoj arhitekturi računarske komunikacije, SSL, IPSec, WEP, WPA i sl. Suština ovih protokola jeste da se spreče napadi tipa: ponavljanje poruka, modifikacija poruka na prenosnom putu ili potpuno preuzimanje komunikacije, npr. u slučaju napada tipa čovek u sredini (eng. *Man-in-the-middle-attack*).

1.3. PRAKTIČNI KRIPTOLOŠKI PROBLEMI

U nastavku se razmatra nekoliko tipičnih kriptoloških problema iz prakse.

1.3.1. PROTOKOLI, STRANE KOJE KOMUNICIRAJU I PROTIVNICI

Kriptografija se bavi konstruisanjem i analizom protokola koji prevazilaze probleme nastale pod uticajem protivnika. Do sada su navedeni primeri nekoliko problema sa protokolima i nekoliko različitih protokola.

Po pravilu mi se prema stranama kojima pružamo zaštitu odnosimo kao prema "dobrim momcima" i želimo da im obezbedimo postizanje njihovih ciljeva. To se radi tako što se konstruišu protokoli koje će oni moći da koriste. Protokol postavlja pravila ponašanja svake strane prilikom obavljanja komunikacije. Protokol je u suštini program.

Protokol može biti takav da kada strana završi komunikaciju, može sačuvati neke informacije za sledeći put (eng. *stateful*). Na primer, strana može znati "ovo je prvi put da pokrećem komunikaciju", "ovo je drugi put da pokrećem komunikaciju", itd.

Protokoli se razlikuju prema problemima koje rešavaju. Nije isto rešenje za protokole za simetrično šifrovanje i za protokole kojima se vrši autentifikacija strana u komunikaciji.

Protivnik predstavlja izvor problema u kriptologiji. On nastoji da iskoristi potencijalne slabosti protokola i da bez kriptanalize dođe do otvorene poruke M , najčešće varanjem druge strane u komunikaciji. Nasuprot tome, protokol nastoji da odbije bilo kakve napade protivnika. U suštini to je igra u kojoj pobeđuje onaj ko je "pametniji" poznavalac protokola.

Kriptografija je najviše fokusirana na protivnika. Proučavajući o tome šta on može da uradi, a šta ne. Ako se ne pronađu adekvatni odgovori na ova pitanja dalji rad ne bi dao neke značajnije rezultate. Kao što smo rekli, protivnik bi mogla biti stvarna osoba, ali takođe može biti i samostalni program za napad, konkurentska kompanija, kriminalna organizacija, vladina institucija, grupa hakera ili nekolicina nesrećnih okolnosti zajedno.

Ako zamislimo moćnog protivnika, mi zauzimamo pesimističan stav prema događajima koji mogu krenuti loše. Trudimo se da uspemo iako neko želi da osujeti naše planove. Možda niko to ne želi. U tom slučaju treba ostvariti bar visoku pouzdanost u zaštiti.

1.3.2. KRIPTOLOGIJA I BEZBEDNOST RAČUNARA

Dobri protokoli su ključni alati za konstruisanje bezbednih računarskih sistema. Loši protokoli predstavljaju slabu kariku preko koje se može lako provaliti u računarske sisteme, izvršiti napad na račun u banci, prisluškivati telefonski razgovor itd. Projektovanje dobrog protokola je veoma teško. Lako je potceniti problem i smisliti protokol koji će se kasnije pokazati kao nefunkcionalan. Vreme i trud potrebni za kvalitetno projektovanje protokola često su potcenjeni. Da bi se posao obavio kako treba, potrebni su znanje, trud i vreme.

Bezbednost ima puno aspekata. Da bi sistem bio bezbedan, potrebno je kombinovati veliki broj faktora. Na primer, ne bi trebalo da postoji mogućnost da hakeri probiju u

sistem i koriste naš nalog, da uzmu naše podatke ili upropaste neki rad. Ovo su zadaci bezbednosti sistema.

Kriptološki protokol je samo deo slagalice. Ako je loše projektovan, protivnik će to iskoristiti. Na primer, ako protokol šalje vašu šifru nezaštićenu, tako da je svako može razumeti, onda je to problem protokola, a ne sistema.

Bezbednost sistema je dobra onoliko koliko je jaka i njegova najslabija karika. Ovo je i razlog zašto je teško projektovati siguran sistem. Da bismo postigli bezbednost koja nam je potrebna, neophodno je da se posvetimo svim problemima: kako da zaštitimo naše mašine od napadača, kako da projektujemo dobar protokol itd.

Obezbeđivanje bezbednosti je težak zadatak. Takođe, različiti sistemi zahtevaju različite vrste zaštite, a time čineći ovu oblast još obimnijom. Jedini način rešavanja ovih problema je deljenje problema sigurnosti na više manjih, lakše rešivih problema.

1.3.3. PRAVILA IGRE

Kriptologija ima određena pravila. Prvo pravilo je da protivnika možemo savladati samo putem dobrih protokola. Danas se zaštita podataka ne zasniva na posedovanju naoružanja i sile, već na primeni strogih matematičkih principa. Kriptologija je danas podržana matematički preciznim garancijama, vremenskom dimenzijom i dostupnom računarskom tehnologijom za rešavanje složenih problema. Metode sile su možda efikasne, ali to nije kriptologija.

Još jedno pravilo na kome većina kriptologa insistira jeste da protokol bude javan. A da tajni deo može biti jedino ključ. Ključevi su tajni podaci, a ne algoritmi. Zašto insistiramo da naši protokoli budu javni? Postoji nekoliko razloga. Odlučni napadač će ionako saznati kakav je to algoritam, pošto po pravilu on mora biti ugrađen u različite programe i računare. Pokušaj da učinimo protokol tajnim je skoro nemoguć. A i skrivanje algoritma dovodi u pitanje da li mi želimo postići sigurnost ili zavaravanje. Pored toga, naučni rad ne može dovoljno napredovati ako je tajan, tako da skrivanje kriptoloških metoda može značiti da metod ostane nedovoljno razvijen.

Vladine organizacije koje imaju veze sa kriptologijom često svoje mehanizme čine tajnim. Za njih je saznavanje kriptološki mehanizama još jedna prepreka koju

protivnik mora preskočiti. Zašto išta otkrivati? Neke organizacije imaju druge razloge zašto ne žele da otkriju svoje mehanizme u koje se ubraja širenje kriptografskog znanja, ili strah od razotkrivanja sposobnosti (ili nesposobnosti) same organizacije.

1.4. PRISTUPI IZUČAVANJU KRIPTOLOGIJE

Razmotrimo ukratko istoriju kriptologije i njena dva razvojna pravca. Razvoj kroz kriptooanalizu i razvoj kroz dokaze.

1.4.1. FAZE U RAZVOJU KRIPTOLOGIJE

Istorijski razvoj kriptologije se može grubo podeliti u tri faze. U prvoj, ranoj fazi, algoritmi su morali da se prave uz pomoć papira i mastila. Cezar je koristio kriptograme. Njegov i ostali načini simetričnog šifrovanja su uključivali šifrovanje zamenom. To znači šifrovanje načinom permutacije $\pi: \Sigma \rightarrow \Sigma$ (zmena, slovo za slovo u okviru jednog pisma). Simbol $\sigma \in \Sigma$ je šifrovan kao $\pi(\sigma)$, a neki tekst je šifrovan tako što se šifrue svaki simbol zasebno. Dešifrovanje se sastoji od primenjivanja π^{-1} . Ovakav način šifrovanja danas ima samo istorijsku vrednost.

Druga faza razvoja je faza kriptografskih mašina. Ova faza se vezuje za period Drugog svetskog rata. Najpoznatija kriptografska mašina tog vremena je bila nemačka "Enigma".

Poslednja faza razvoja kriptografije je moderna kriptologija. Centralnu ulogu imaju matematika i računari. Računari omogućavaju korišćenje kompleksnih algoritama, a matematika omogućava njihovo projektovanje. Naučni period kriptografije datira od 1949. godine i radova Kloda Šenona.

1.4.2. RAZVOJ KRIPTOLOGIJE KROZ KRIPTOANALIZU

Tradicionalno, mehanizmi kriptologije su se razvijali fokusirani na konkretne napade i načine kako da ih odbiju. To bi izgledalo, na primer, ovako:

1. Kriptografski cilj je određen;

2. Ponuđeno je rešenje;
3. Traganje za načinom napada na ponuđeno rešenje;
4. Kada se pronađe potencijalna slabost rešenja, vraćamo se na korak 2 i ponavljamo proces.

Treći korak se naziva kriptanaliza.

Po nekima, kriptografija se odnosi na kreiranje kriptografskih mehanizama, kriptanaliza se odnosi na napade na te mehanizme, a kriptologija obuhvata obe pomenute. Mi smo često koristili reč kriptografija tamo gde bi reč kriptologija bila adekvatnija. Mnogi kriptolozi ne pridaju značaja ovoj razlici, pa nećemo ni mi.

Postoje neki problemi u razvoju kriptografije kroz kriptanalizu. Očigledan problem je to što se nikada ne zna da li je posao do kraja obavljen kako treba i kada je posao završen. Pomenuti proces bi trebalo da se ponavlja sve dok projektant ne 'oseti' da je proizvod adekvatan. Ali projektant mora da prihvati da greške mogu biti otkrivene u bilo kom trenutku. Uz malo sreće, napadi neće biti fatalni, mada se može se desiti i da budu. Čak se može desiti da projektant shvati da je popravka sistema teška ili čak nemoguća.

Kriptanaliza zahteva visoku inteligenciju. Smatra se da kriptanaliza ne može biti naučena, već da dobar kriptanalitičar mora imati jak "osećaj" za matematiku i veoma razvijen intelekt. Iz tog razloga su kriptanalitičari veoma cenjeni.

1.4.3. ŠENONOVA SIGURNOST SIMETRIČNOG ŠIFROVANJA

Kriptografija sa simetričnim šifrovanjem, u kojoj dokazi i definicije igraju značajnu ulogu, počinje sa radom Kloda Šenona (*Claude Shannon*). Šenon ne samo da je bio "otac" teorije informacija, već se smatra da je i "otac" moderne kriptologije.

Sigurnost koju pominjemo znači poraziti protivnika, što znači da moramo da definišemo šta protivnik hoće i šta zapravo tajnost predstavlja. Prema Šenonu, savršena (perfektna, apsolutna) tajnost je ona po kojoj prilikom razbijanja šifrata C možemo sa istom verovatnoćom dobiti bilo koje dve poruke M_1 i M_2 .

Da bismo imali apsolutnu tajnost simetričnog šifarskog sistema, dužina ključa treba da odgovara dužini poruke koja se šifrue. Međutim, u praktičnoj primeni kriptografije, jedan ključ često se koristi za šifrovanje velikih količina podataka koji se prenose ogromnim brzinama. To znači da se u najvećem broju današnjih praktičnih rešenja za prenos ne može postići Šenonova apsolutna tajnost. Zbog toga moramo da prihvatimo sisteme koji nisu savršeni, ali su ipak dovoljno dobri i odgovaraju za računarsku primenu.

1.4.4. TEORIJA RAČUNARSKE KOMPLEKSNOSTI

Moderna kriptologija je povezana sa problemima koji se odnose na intenzitet računarske moći koja je protivniku na raspolaganju. Tajnost u računarskim sigurnim šifarskim sistemima zasnivamo na pretpostavci da protivnik nema dovoljno "računarskog" vremena. Teorijski posmatrano, praktične šifarske sisteme je moguće probiti, ali se u realnom vremenu to ne može desiti. Najčešće su ovakvi napadi nedostižni.

Ovo je radikalna promena posmatrana sa mnogih aspekata. Ono što prenosi kriptografiju iz domena teorije informacija u domen računarske tehnologije. I to zapravo menja ono što možemo postići.

Mi želimo da dostignemo ovakve tvrdnje:

Pretpostavljamo da protivnik koristi ne više od t računarskih ciklusa i verovatnoća da on probije naš metod je $t/2^{200}$. Kao što smo primetili, ova tvrdnja je zasnovana na verovatnoći. Skoro sve tvrdnje sa kojima ćemo se susreti će biti zasnovane na verovatnoći.

Takođe, nismo ni govorili o tome kako protivnik radi. Koji algoritam ili tehniku koristi, o tome ništa ne znamo. A naša tvrdnja je čvrsta i pored svega toga. To je čini veoma snažnom tvrdnjom.

Do sada nam je postalo jasno da tvrdnje kao ova pomenuta, u praksi predstavljaju dovoljno dobru zaštitu. Međutim, protivnik usavršava i unapređuje svoje dekripterske mogućnosti te je i mogućnost probijanja date šifre sve veća. Kada bi

protivnik imao računarsku snagu od 2^{200} ciklusa mi ne bismo imali nikakvu sigurnost. Međutim, niko nema toliku računarsku snagu.

1.4.5. JEDNOSMERNE FUNKCIJE

Protokole na najnižem nivou nazivamo osnovni protokoli. Protokoli višeg nivoa su izgrađeni nad ovim protokolima. U kriptologiji, protokoli se konstruišu na osnovu kriptografskog problema sa kojim se susrećemo. Kriptološki protokoli objašnjavaju kako šifrovati, kako potvrditi identitet, kako distribuirati ključ. Oni predstavljaju nadogradnju osnovnih protokola. Osnovni protokoli su zasebni protokoli i oni su jednostavniji od protokola višeg nivoa. Oni imaju određenu složenost i bezbednosna svojstva, ali sami ne rešavaju čitav kriptografski problem. Oni se moraju pravilno koristiti da bi se postigao željeni rezultat. Ranije se nije pravila neka veća razlika između osnovnih i viših protokola. U nekim slučajevima razlika i ne postoji.

Računarska priroda moderne kriptologije nalaže da se rešenja moraju zasnivati na teškim problemima sa računarskog aspekta. Teško je pronaći odgovarajući metod koji bi bio teško rešiv čak i primenom moćnih računara.

Jedno od odgovarajućih rešenja bi mogle biti jednosmerne funkcije. To su funkcije, $f: D \rightarrow R$, koje za zadato D daju R , za koje važe dva pravila:

- (1) f je lako izračunati, odnosno postoji efikasn timer koji za zadato $x \in D$, daje $y = f(x) \in R$.
- (2) Teško je izračunati inverznu vrednost funkcije. Protivniku koji ima $y \in R$ je veoma teško da izračuna x tako da je $f^{-1}(y) = x$. Ovo važi u slučaju ograničene računarske moći.

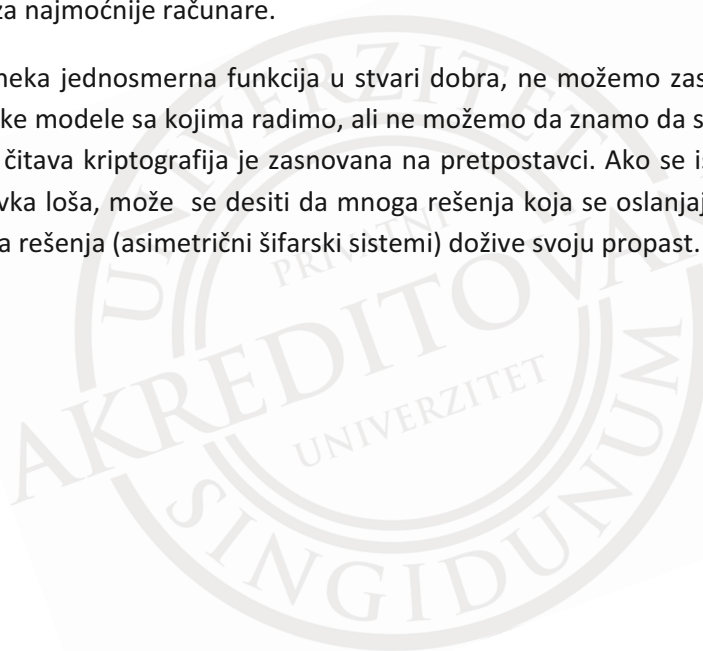
Da li je moguće napraviti funkcije sa takvim osobinama? Kaže se da su primeri jednosmernih funkcija veoma česti u stvarnom životu. Lakše je razbiti stakleni tanjir, nego sastaviti ga ponovo. Naravno, mi želimo da projektujemo matematički model koji možemo implementirati u sistem.

Jedan izvor primera proističe iz teorije brojeva i on pokazuje veliku povezanost između kriptografije i teorije brojeva. Veliki deo kriptografije je zasnovan na teoriji

brojeva. Jedna jednostavna jednosmerna funkcija zasnovana na teoriji brojeva je množenje. Na primer funkcija f uzima dva broja a i b i množi ih da bi dobila $N = ab$. Ne postoji ni jedan poznati algoritam koji može brzo i uvek da reši problem tako što će iz proizvoljno dodeljenog $N = ab$ da odredi a i b .

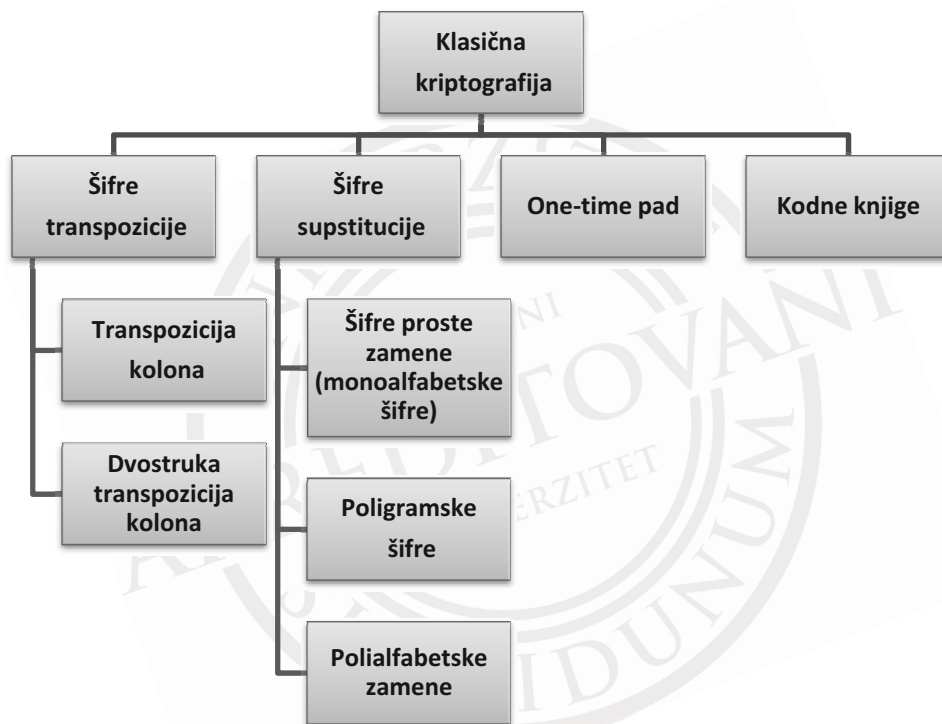
Isto tako jedna od jednosmernih funkcija je i diskretna eksponencijalna funkcija. Inverzna funkcija eksponencijalnoj je diskretna logaritamska funkcija. Ispostavilo se da ne postoji algoritam koji dovoljno brzo rešava logaritamske funkcije. Recimo da je za dovoljno veliku vrednost, računanje diskretne logaritamske funkcije preveliki zadatak i za najmoćnije računare.

Koliko je neka jednosmerna funkcija u stvari dobra, ne možemo zasigurno znati. Mi imamo neke modele sa kojima radimo, ali ne možemo da znamo da su zaista efikasni. Odnosno, čitava kriptografija je zasnovana na pretpostavci. Ako se ispostavi da je ta pretpostavka loša, može se desiti da mnoga rešenja koja se oslanjaju na savremena kriptološka rešenja (asimetrični šifarski sistemi) dožive svoju propast.



2. KLASIČNA KRIPTOGRAFIJA

U ovom poglavlju ukratko će biti razmotrene klasične šifre koje su dominirale u dalekoj istoriji. One su izabrane zbog svog istorijskog značenja i da bi se prikazali osnovni principi koji postoje i danas kod modernih šifara. Slika 2.1, napravljena je podela prema tipovima klasičnih šifara.



Slika 2.1 Podela klasične kriptografije na tipove šifri

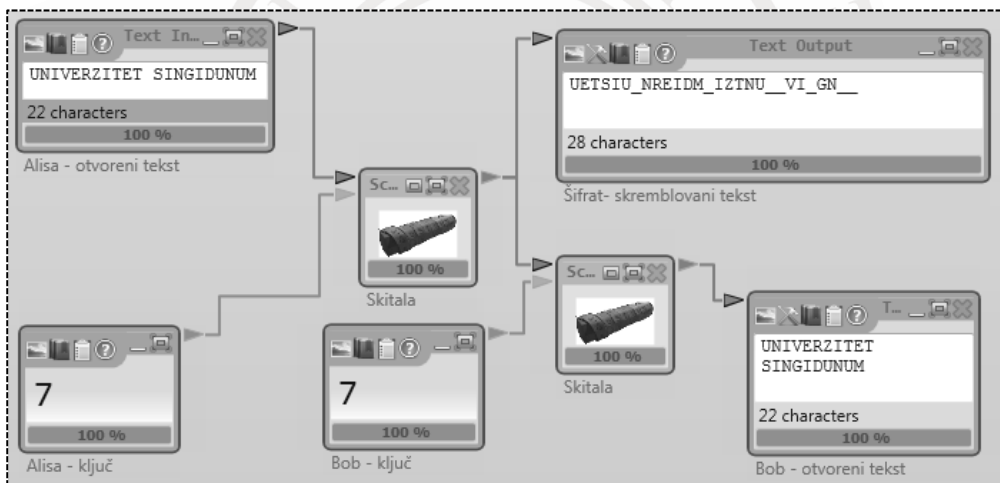
2.1. ŠIFRE TRANSPOZICIJE

Šifre transpozicije rade na principu skremblovanja otvorenog teksta, tj. slova otvorenog teksta se međusobno premeštaju. Rezultat ovog tipa šifre je skremblovani tekst koji predstavlja šifrat. Ključ predstavlja primenjenu transpoziciju. Način rada ovog tipa klasične šifre odgovara Šenonovom (*Claude Shannon*) principu difuzije.

Difuzija je definisana kao način širenja statistike otvorenog teksta i šifrata. Ideja o primeni principa difuzije našla je primenu i kod modernih šifara.

Prva šifra ovog tipa je Skitala. Koristili su je Spartanci (500 god.). Princip rada šifre je jednostavan. Obavija se kožna traka oko štapa, zatim se poruka piše na tako dobijenom omotaču duž štapa. Kada je poruka napisana, kožna traka se razmota i dobije se ispremeštan ili skremblovan tekst. Znakove je mogao pročitati samo onaj ko je imao štap jednake debljine. Sada možemo da zaključimo da je kompletna tajnost ove šifre određena debljinom štapa, tj. ključ je zapravo debljina štapa.

Slika 2.2, Alisa i Bob koriste Skitalu za šifrovanje poruke. Alisa generiše otvoreni tekst i određuje ključ (debljina štapa). Na drugoj strani Bob dobija šifrat, poseduje isti ključ i uspešno dešifruje poruku.



Slika 2.2 Šifrovanje i dešifrovanje sa Skitalom

2.1.1. TRANSPOZICIJA KOLONA

Ukoliko bismo želeli da u današnje vreme realizujemo programski šifru Skitalu, uradili bismo to na sledeći način. Otvoreni tekst poruke se postavi u redove jedne dvodimenzionalne matrice. Broj redova zavisi od dužine otvorenog teksta. Šifrat se dobija iščitavanjem kolona iste matrice. Efekat je isti kao kada bi se koristila Skitala, tj. ključ je broj kolona.

1. PRIMER:

Pretpostavimo da je matrica dimenzije [3 x 7]:

U	N	I	V	E	R	Z
I	T	E	T	S	I	N
G	I	D	U	N	U	M

Otvoreni tekst: „UNIVERZITET SINGIDUNUM“

Šifrat: [UIG NTI IED VTU ESN RIU ZNM]

Šifrat je dobijen iščitavanjem kolona matrice, leva na desno.

Moguće je unapređenje transpozicije kolona pomoću ključne reči. U ovom slučaju ključna reč određuje redosled transpozicija. Izabrana ključna reč se tumači po abecednom redosledu. Dužina ključne reči može biti jednaka ili manja od broja kolona. Ukoliko je manja, ključna reč se ponavlja.

2. PRIMER:

Šifrovanje pomoću ključne reči „STUDENT“.

S ₂₄	T ₂₆	U ₂₇	D ₆	E ₉	N ₁₉	T ₂₆
U	N	I	V	E	R	Z
I	T	E	T	S	I	N
G	I	D	U	N	U	M

Otvoreni tekst: „UNIVERZITET SINGIDUNUM“

Šifrat: [VTU ESN RIU UIG NTI ZNM IED]

Šifrat je dobijen iščitavanjem kolona prema redosledu pozicija od slova po alfabetu iz ključne reči.

Ključ: STUDENT

2.1.2. KRIPTOANALIZA TRANSPOZICIJA KOLONA

Posmatrajmo prethodni primer sa aspekta kriptanalitičara. Neka Trudi ima na raspolaganju šifrat: [VTU ESN RIU UIG NTI ZNM IED]. Matrica je dimenzija $[n \times m]$, za neko n i m . Šifrat ima ukupno 21 slovo = $[n \times m]$. Trudi treba da pronađe broj mogućih dimenzija za matricu $[n \times m]$. U ovom primeru postoje dve dimenzije matrice, jedna matrica dimenzije $[3 \times 7]$ i druga $[7 \times 3]$. U sledećem koraku Trudi će da formira matrice sa šifratom koji poseduje. Zatim će da započne potpunu pretragu svih mogućnosti, tako što će da menja redosled kolona sve dok ne dobije smislen tekst.

3. PRIMER:

0	1	2	3	4	5	6	3	2	6	0	4	1	5	3	4	6	0	1	2	5		
V	E	R	U	N	Z	I	U	R	I	V	N	E	Z	U	N	I	V	E	R	Z		
T	S	I	I	T	N	E	,	I	I	E	T	T	S	N	...	I	T	E	T	S	I	N
U	N	U	G	I	M	D	G	U	D	U	I	N	M	G	I	D	U	N	U	M		

Iz ovog primera kriptanalize vidimo da je jedino moguće rešenje za Trudi potpuna pretraga prostora ključeva. Ako Trudi izabere tačnu matricu $[3 \times 7]$ i započne potpunu pretragu ključa, u nekom momentu će dobiti smisleni tekst ili poruku. Ako je prostor ključa dovoljno veliki, takav napad ne može da se završi za prihvatljivo vreme. Tačnije, verovatnoća uspeha je veoma mala. Za sigurnost bilo koje šifre (kriptografskog mehanizma) neophodan je veliki prostor ključeva, to i nije dovoljan uslov.

2.1.3. DVOSTRUKA TRANSPOZICIJA KOLONA

Dvostruka transpozicija predstavlja dalje unapređenje transpozicija kolona sa ključnom reči. Izvodi se tako što se menja redosled kolona i redosled redova u izabranoj dimenziji matrice $[n \times m]$. U ovom slučaju ključ je dimenzija matrice i permutacije po kolonama i redovima. U sledećem primeru pokazaćemo realizaciju dvostruke transpozicije na prethodnom primeru.

4. PRIMER:

Šifrovanje dvostrukom transpozicijom.

Otvoreni tekst: „UNIVERZITET SINGIDUNUM“

kolone	0	1	2	3	4	5	6		kolone	1	0	6	2	4	3	5
red 0	U	N	I	V	E	R	Z		red 1	T	I	N	E	S	T	I
red 1	I	T	E	T	S	I	N	>>>	red 0	N	U	Z	I	E	V	R
red 2	G	I	D	U	N	U	M		red 2	I	G	M	D	N	U	U

Šifrat: [TINESTI NUZIEVR IGMDNUU]

Šifrat je dobijen skremblovanjem rednih brojeva (od 0 do $n - 1$, n -broj kolona ili redova) kolona i redova. Zatim, su iščitane vrednosti po redovima.

Ključ: dimenzija matrice $[3 \times 5]$ i permutacije $(1,0,6,2,4,3,5), (1,0,2)$

2.1.4. KRIPTOANALIZA DVOSTRUKE TRANSPOZICIJE

Posmatrajmo prethodni primer sa aspekta kriptanalitičara. Neka Trudi ima na raspolaganju šifrat: [TINESTI NUZIEVR IGMDNUU]. Matrica je dimenzija $[n \times m]$, za neko n i m . Šifrat ima ukupno 21 slovo = $[n \times m]$. Trudi treba da pronađe broj mogućih dimenzija za matricu $[n \times m]$. U ovom primeru postoje dve dimenzije matrice, jedna matrica dimenzije $[3 \times 7]$ i druga $[7 \times 3]$, ali postoji veliki broj mogućih permutacija, koji definišemo kao:

$$3! \cdot 7! \approx 2^{15}$$

Postavlja se pitanje da li postoji skraćeni napad ili lakši put, a to bi značilo da Trudi ne mora da primeni kriptanalitički metod za potpunu pretragu celog prostora ključeva. U prvom korak Trudi treba da pogodi odgovarajuću dimenziju matrice, a u drugom koraku pokušava prvo da pogodi permutacije kolona. Na ovaj način Trudi traži smislenu reč samo u koloni 0. Ova metoda skraćenog napada na dvostruku transpoziciju poznata je pod nazivom "podeli (zavadi) pa vladaj". To znači da Trudi napada samo deo prostora ključeva, a ovo predstavlja važnu strategiju napada. Iz tog

razloga, sam dizajn algoritma za šifrovanje zahteva pažljivu analizu od strane kriptanalitičara. Osnovni cilj kriptografa je da dizajniraju šifru koja je imuna na ovakve tipove napada. Ova anomalija prisutna je i danas kod nekih šifara.

2.2. ŠIFRE ZAMENE - SUPSTITUCIJE

Šifre zamene ili supstitucije mogu se porediti sa šifrom transpozicija. Kod šifre sa transpozicijama, slovima otvorenog teksta se menja raspored, dok vrednosti originalnih slova ostaju iste. Nasuprot tome, kod šifri sa zamenom, raspored slova otvorenog teksta ostaje nepromenjen, ali slova teksta menjaju svoju vrednost, tj. preslikavaju se u druga slova. Sa aspekta kriptanalize pretpostavimo da napadač poznaje algoritam, ali ne zna ključ. Koliko ključeva mora da isproba? Odgovor je 26!. Ovo bi značilo da je potrebno 5 milijardi godina, ako svaka osoba na Zemlji troši jednu sekundu za testiranje jednog ključa.

Postoji više tipova šifara zamene. Monoalfabetska šifra koristi fiksnu zamenu tokom cele poruke. Homofone šifre predstavljaju unapređenje šifre prostih zamena, uvodeći element slučajnosti. Poligramske šifre rade na principu zamena nad većim grupama slova u otvorenom tekstu. Polialfabetske šifre koriste veliki broj zamena na različitim pozicijama u poruci, u kojoj se slovo iz otvorenog teksta preslikava u jednu od nekoliko mogućnosti u šifratu.

2.2.1. ŠIFRE PROSTE ZAMENE (MONOALFABETSKE)

Cezarova šifra je primer šifre proste zamene. Julije Cezar je koristio ovu šifru 50-30. god. u komunikaciji sa svojim prijateljima. Šifrat se dobija tako što se svako slovo otvorenog teksta od A do W zameni sa slovom koje je za 3 mesta dalje po abecednom redu, a slova X, Y i Z se zamenjuju slovima A, B i C.

Cezarovu šifru možemo pregledno prikazati na sledeći način:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Prvi red u tabeli posmatramo kao otvoreni tekst, a drugi red kao šifrat za pomeranje alfabeta otvorenog teksta za 3 pomeraja. Cezarova šifra funkcioniše tako što se svakom od 26 slova u alfabetu otvorenog teksta dodeli broj od 0 do 25. A je 0, B je 1, C je 2, ..., Z je 25.

Cezarov algoritam za šifrovanje/dešifrovanje:

- $c_i = p_i + K_E \pmod{26}$
- $p_i = c_i + (26 - K_E) \pmod{26}$
- p_i i-to slovo otvorenog teksta
- c_i i-to slovo šifrata
- K_E je ključ (pomeraj), u ovom primeru je 3

5. PRIMER:

Kriptografski sistem baziran na Cezarovoj šifri.

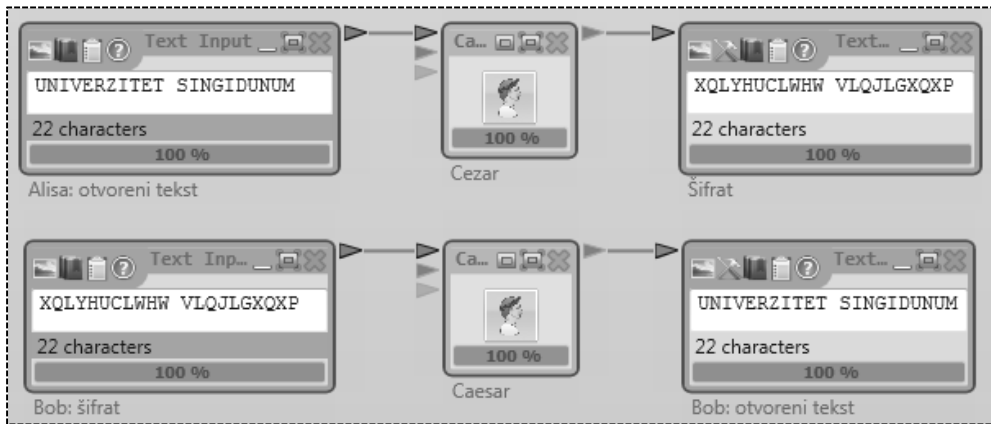
Alisa i Bob koriste Cezarovu šifru za šifrovanje poruke. Alisa generiše otvoreni tekst i određuje ključ 3. Na drugoj strani Bob dobija šifrat, poseduje isti ključ i uspešno dešifruje poruku.

Šifrovanje:

$$c_i(U) = p_i + K_E \pmod{26} = X \quad (\text{primer šifrovanja za prvo slovo otvorenog teksta})$$

Dešifrovanje:

$$p_i(X) = c_i + (26 - K_E) \pmod{26} = U \quad (\text{primer dešifrovanja za prvo slovo šifrata})$$



Slika 2.3 Šifrovanje i dešifrovanje sa Cezarovom šifrom

U **Slika 2.3**, možemo vidjeti da su ključevi za šifrovanje i dešifrovanje isti, ali su algoritmi različiti. Kod šifrovanja imamo pomeraj u levo za 3 pozicije, a kod dešifrovanja pomeraj u desno za 3 pozicije.

2.2.2. KRIPTOANALIZA CEZAROVE ŠIFRE

Cezarova šifra proste zamene sa pomerajem je dosta slaba i jednostavna za kriptanalitičara. Šifra ima samo 26 mogućih ključeva. Najjednostavnije je probati sve ključeve, tj. primeniti metod potpune pretrage ključa. Iz ovog razloga potrebno je napraviti neku izmenu u ovoj šifri zamene u cilju povećanja prostora ključeva. Ključ ne mora biti pomeraj za određeni broj (3). Ključ može da bude bilo koja permutacija slova. Misli se na broj permutacija od 26 slova u šifratu. U drugom redu tabele ispod, prikazana je jedna izabrana permutacija koja predstavlja ključ.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	M	C	X	Y	Z	H	I	J	N	O	P	T	U	V	Q	K	L	D	E	F	G	R	S	W

Nakon napravljene izmene, prostor mogućih ključeva možemo da definišemo na sledeći način:

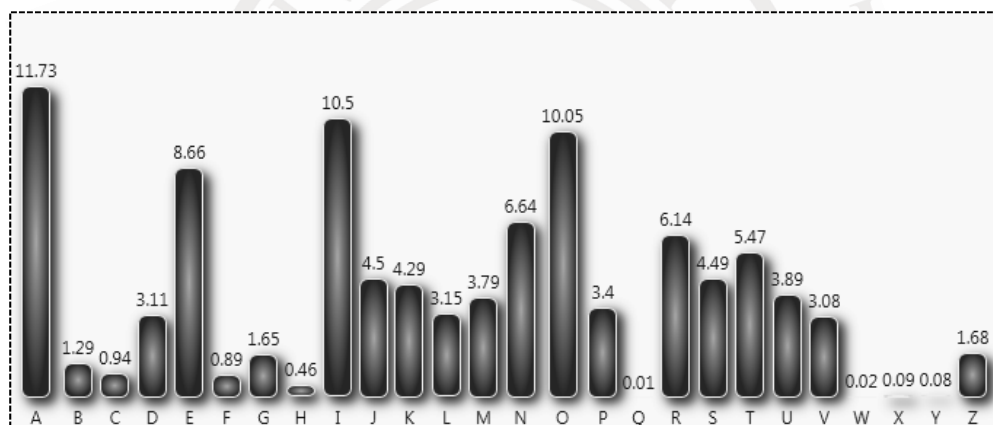
$$26! > 2^{88} \approx 4 \cdot 10^{26}$$

Trudi zna da je primenjena prosta zamena, ali nema mogućnost da isproba svih 2^{88} ključeva. U tom slučaju koristi statistiku ili statističku strukturu srpskog jezika. Meri se frekvencija pojavljivanja pojedinih slova u šifratu. Zatim poređenjem dobijenih frekvencija iz šifrata, određuje se koje slovo je sa kojim zamenjeno. Svaki svetski jezik poseduje svoju jedinstvenu distribuciju slova u reči.

6. PRIMER:

Frekvencije slova u Srpskom jeziku.

Za sada nam je poznata samo činjenica da su najfrekventnija slova u srpskom jeziku A, I, O, E, N, R i T. Na osnovu izabranog uzorka teksta iz srpskog jezika, izračunaćemo frekvencije slova koje važe za srpski jezik. **Slika 2.4.** frekvencije slova u srpskom jeziku.

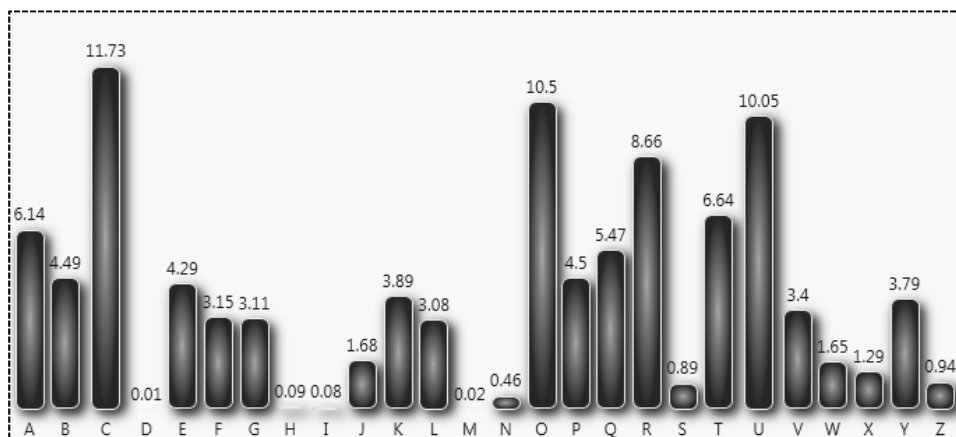


Slika 2.4 Frekvencije slova u srpskom jeziku

7. PRIMER:

Kriptoanaliza Cezarove šifre na osnovu frekvencije slova.

Alisa i Bob koriste Cezarovu šifru za šifrovanje poruke. Ključ je jedna od permutacija 26 slova. Trudi je uspela da presretne šifrat i pokušava pomoću statistike da dešifruje poruku, **Slika 2.5** Frekvencije slova u šifratu na srpskom jeziku. Ne postoji drugi povoljan način, ukoliko Trudi zna na kom je jeziku napisan otvoreni tekst. Pretpostavimo da je u ovom slučaju otvoreni tekst napisan na srpskom jeziku.



Slika 2.5 Frekvencije slova u šifratu na srpskom jeziku

Na osnovu dobijenih frekvencija, Trudi je uspeła da dešifruje poruku. Pronađeni ključ je permutacija od 26 slova iz skupa ključeva 2^{88} , [A, B, M, C, X, Y, Z, H, I, J, N, O, P, T, U, V, Q, K, L, D, E, F, G, R, S, W].

2.2.3. HOMOFOENE ŠIFRE

Homofona šifra predstavlja unapređenje šifre proste zamene. Šifrat kod ove šifre može da sadrži više od 26 znakova. Pretpostavlja se da je otvoreni tekst definisan iz skupa 26 slova. Većina slova iz otvorenog teksta se u šifratu zamenjuje jednoznačno, kao i prethodnim primerima. Neka slova otvorenog teksta mogu da u šifratu budu predstavljena na dva ili više načina. Razlog uvođenja dodatnih znakova jeste širenje statistike šifrata.

Svako slovo otvorenog teksta šifruje sa brojevima od 00, 01, ..., 32. Uvedeno je 6 dodatnih brojeva. U ovom slučaju neko slovo će biti predstavljeno sa dva broja. Slova su izabrana u odnosu na njihove frekvencije u srpskom jeziku. Obično su to slova sa najvećim frekvencijama. Na osnovu frekventnog teksta koji je urađen u prethodnim primerima za srpski jezik, izabraćemo slova: A, I, O, E, N i R, koja ćemo predstaviti na dva načina (dva broja) u šifratu.

8. PRIMER:

Šifrovanje homofonom šifrom.

A	A	B	C	D	E	E	F	G	H	I	I	J	K	L	M
04	05	08	01	02	06	11	12	13	14	15	07	03	09	10	16
N	N	O	O	P	Q	R	R	S	T	U	V	W	X	Y	Z
20	17	28	29	30	21	22	23	24	18	19	25	26	27	31	32

U datoj matrici slova predstavljaju otvoreni tekst. Šifrovanjem otvorenog teksta svako slovo otvorenog teksta se preslikava u izabrani broj. Dobijeni brojevi predstavljaju šifrat homofone šifre.

Otvoreni tekst: "UNIVERZITET SINGIDUNUM" (slova N, E i I se ponavljaju)

Šifrat je dobijen pojedinačnom zamenom slova iz otvorenog teksta na osnovu brojeva koji su određeni za svako slovo alfabeta, prikazano u tabeli iznad.

Šifrat: [19,20,15,25,06,22,32,07,18,11,18,24,15,17,13,07,02,19,20,19,16]

2.2.4. KRIPTOANALIZA HOMOFONE ŠIFRE

Za pravilan rad homofone šifre bitan je izbor slova sa najvećim frekvencijama. Zatim, bitno je ustanoviti pravilo kada dodeliti jednu od dve brojne vrednosti za isto slovo otvorenog teksta. Nakon izbora najfrekventnijih slova i dodelom duplih brojnih vrednosti za izabrana slova, utičemo značajno na sigurnost šifre. Šifra postaje otpornija na statističke napade, ali ipak ostaju ranjive na napade poznatog dela otvorenog teksta.

Ipak, statističkom analizom moguće je doći do informacija o ključu. Šifrat generisan na ovaj način, trebao bi da prikrije statističke osobine otvorenog teksta. U tom slučaju šifrat bi morao da ima osobine slučajnog niza. Postizanje slučajnosti je veoma složen proces, teško je precizno definisati slučajnost ili entropiju. Entropija u ovom slučaju predstavlja maksimalnu neodređenost šifrata. Ukoliko bi šifrat posedovao maksimalnu neodređenost (visoku entropiju), kriptanalitičar bi imao minimalne

šanse da dođe do ključa ili otvorenog teksta na osnovu šifrata. Kriptografi ulažu velike napore da se uspešno bore sa napadima zasnovanim na statističkim metodama.

2.3. POLIGRAMSKE ŠIFRE

Poligramske šifre predstavljaju još jedno unapređenje supstitucionih šifara. Grupe slova su šifrovane zajedno, a ne kao pojedinačna slova. Primer: "ABA" se šifruje sa "RTQ", "ABB" se šifruje sa "SLL", itd. Predstavnici ovog tipa šifara su plejfer šifra (eng. *Playfeir*) i Hilova (eng. *Hill*) šifra. Neki ih nazivaju i pretečama blokovskih šifara.

2.3.1. PLEJFER ŠIFRA

Postupak šifrovanja sa plejfer šifrom zasniva se na zameni blokova od dva slova ili simbola (bigrami). Ovaj postupak šifrovanja predložio je Čarls Vitston (Charles Wheatstone) 1854. godine. Plejfer šifru su koristili Britanci u vreme Prvog svetskog rata. Ključ Plejfer šifre bila je matrica dimenzija $[5 \times 5]$ sa 25 slova, karakter "J" se nije koristio ili je bio prisvojen nekom drugom slovu. Matrica se konstruiše na osnovu izabrane ključne reči. Par slova (bigram) otvorenog teksta m_1 i m_2 šifruje se u skladu sa sledećim pravilima:

- Ako su m_1 i m_2 u istom redu, tada su c_1 i c_2 dva slova desno od m_1 i m_2 , usvojeno je da je prva kolona susedna desna kolona poslednjoj koloni;
- Ako su m_1 i m_2 u istoj koloni, tada su c_1 i c_2 dva slova ispod m_1 i m_2 , usvojeno je da je prvi red susedni donji poslednjem redu;
- Ako su m_1 i m_2 u različitim kolonama i redovima, tada su c_1 i c_2 ostale dve ivice (temena) pravougaonika koji sadrži ivice m_1 i m_2 , gde je c_1 u istom redu kao i m_1 dok je c_2 u istom redu kao i m_2 ;
- Ako je $m_1 = m_2$, tada se vrši umetanjem neutralnog (*null*) karaktera (na primer, X) između m_1 i m_2 ;
- Ukoliko poruka otvorenog teksta ima neparan broj slova tada se na kraj poruke dodaje neutralni (*null*) karakter.

9. PRIMER:

Postupak šifrovanja Plejfer šifrom.

Ključ: Generisana matrica [5 x 5] za izabranu ključnu reč "SING";

S	I	N	G	A
B	C	D	E	F
H	K	L	M	O
P	Q	R	T	U
V	W	X	Y	Z

Otvoreni tekst:	SI	NG	ID	UN	UM
Šifrat:	IN	GA	NC	RA	TO

U ovom primeru šifrovanja vidimo kako je dizajniran algoritam plejfer šifre za postupak šifrovanja. Da bi postupak dešifrovanja bio inverzan, primenjuje se identičan postupak kao i kod šifrovanja nad šifratom za inicijalnu matricu, generisanu na osnovu ključne reči.

Plejfer šifra ima nekoliko značajnih prednosti. Imajući u vidu da je šifra bigramska, u šifratu se gube pojedinačna slova koja dosta utiču na frekvenciju jednog svetskog jezika. Šifrovanjem bigrama, smanjuje se značajan broj elemenata koji je dostupan statistici, primenom metoda za analizu frekvencija. Međutim, broj bigrama je mnogo veći od broja individualnih slova. Broj slova u srpskom jeziku je 30, a broj bigrama više od 600. Vršanjem analize frekvencija nad bigramima, dobili bismo potpuno ujednačene frekvencije za razliku od frekvencija visoko frekventnih slova. Analiza bigrama u srpskom jeziku za 30 visoko frekventnih bigrama prikazana je u tabeli sledećeg poglavlja.

2.3.2. KRIPTOANALIZA PLEJFER ŠIFRE

Sigurnost ove šifre zavisi od veličine otvorenog teksta koji se šifruje. Ukoliko je šifrat dovoljno velik, moguće je primeniti analizu frekvencija nad bigramima. Tačnije,

ukoliko je uzorak dovoljan, kriptanalitički metod na osnovu statistike će biti uspešniji. Poznato je da i kod ove šifre jedan deo strukture jezika ostaje sačuvan. Slova u šifratu nisu uniformno raspoređena, tako da njihova raspodela odgovara raspodeli nekog slučajnog niza. Zapravo, razlika koja postoji između ovih raspodela (raspodela slova u šifratu i u slučajnom nizu) predstavlja taj deo strukture jezika koji će kasnije biti iskorišćen za kriptanalizu plejfer šifre. **Tabela 2.** sadrži analize frekvencija za 30 najfrekventnijih bigrama u srpskom jeziku, kao i 30 najfrekventnijih bigrama u šifratu koji je dobijen sa plejferovom šifrom na osnovu istog otvorenog teksta.

Tabela 2 - Analiza frekvencije bigrama u srpskom jeziku i šifratu.

Uzorak: Tekst sadrži 15000 reči na srpskom jeziku						
Analiza:		Visoko frekventni bigrami srpskog jezika (otvoreni tekst)		Visoko frekventni bigrami u šifratu (plejfer šifrat)		
R.B.	Bigrami	Frekvencija (%)	Broj	Bigrami	Frekvencija (%)	Broj
1	ri	2.0836	1989	GC	2.3561	1449
2	je	1.9966	1906	NW	1.7138	1054
3	st	1.5933	1521	QN	1.7138	1054
4	na	1.5640	1493	NI	1.6699	1027
5	ko	1.5493	1479	GN	1.6569	1019
6	en	1.5190	1450	DG	1.6553	1018
7	an	1.5179	1449	SG	1.5789	971
8	ra	1.5085	1440	GS	1.3968	859
9	ti	1.3744	1312	NS	1.3350	821
10	ij	1.3283	1268	UN	1.3187	811
11	no	1.3053	1246	LH	1.3090	805
12	pr	1.2225	1167	GP	1.2764	785
13	om	1.2099	1155	QG	1.1903	732
14	me	1.2089	1154	KN	1.1138	685
15	ni	1.1953	1141	QT	1.0911	671
16	re	1.1314	1080	NG	1.0716	659
17	va	1.1282	1077	AL	1.0439	642
18	po	1.0559	1008	TM	0.9984	614
19	tr	1.0318	985	SN	0.9951	612

20	to	1.0308	984	TD	0.9659	594
21	ta	1.0277	981	LF	0.9285	571
22	ov	1.0235	977	HZ	0.9008	554
23	or	1.0088	963	AQ	0.8976	552
24	is	0.9983	953	HA	0.8829	543
25	te	0.9805	936	GU	0.8797	541
26	ja	0.9638	920	HO	0.8764	539
27	da	0.9491	906	GK	0.8748	538
28	od	0.9239	882	UT	0.8732	537
29	ma	0.9187	877	AK	0.8716	536
30	ne	0.9156	874	UM	0.8667	533

Ukoliko je dobijeni šifrat isuviše mali, analizom frekvencija nije moguće uraditi kriptanalizu datog šifrata kao što je to prikazano u prethodnom primeru. U ovakvim slučajevima moguće je primeniti metod "poznata reč ili verovatna reč", koja je sadržana u šifratu. Metod se sastoji u tome da se pretpostave reči čija je verovatnoća visoka da će se pojaviti u šifratu. Primer: "Univerzitet", "Singidunum". Zatim se u šifratu traži struktura koja odgovara pretpostavljenoj reči i na taj način se postupno rekonstruiše matrica na osnovu koje je generisan šifrat.

2.3.3. HILL-OVA ŠIFRA

Šifru je kreirao Lester S. Hill 1929. Hilova šifra je još jedan predstavnik poligramskih šifara. Predstavlja šifru zamene koja radi na osnovu linearne algebre. Hilova šifra je prva šifra koja praktično radi na grupi (bloku) od više od tri slova otvorenog teksta. Ključ ove šifre je kvadratna matrica. Hill uvodi inverzne matrice i množenje matrica za proces mešanja otvorenog teksta. Veličina bloka otvorenog teksta određuje dimenzije matrice koja se koristi kao ključ. Na primer, blok od 3 slova koristi matricu $[5 \times 5]$, blok od 4 slova koristi matricu $[4 \times 4]$ itd... Sada možemo da zaključimo na osnovu dizajna algoritma da Hilova šifra sigurno predstavlja preteču savremenih blok šifri. Dizajn algoritma koristi jednostavne linearne jednačine.

Hilova šifra sa matricom $[n \times m]$, ne sakriva samo frekvencije o pojedinačnim slovima, već i frekvencije o bigramima. Iz tog razloga, Hilovu šifru možemo smatrati potpuno sigurnu na napad na osnovu šifrata. Tačnije, Hilovom šifrom u potpunosti je

prikrivena struktura jezika na kome je naš otvoreni tekst. Međutim i ovo je još jedna šifra koja omogućava kriptanalizu na osnovu napada "*poznati otvoreni tekst*", a još lakše pomoću napada "*izabrani otvoreni tekst*". Ovo je jedan od razloga zašto ovaj kriptografski algoritam nije bio praktično upotrebljavan.

Da bi se suprotstavio svakodnevnom optužbama da je njegov sistem isuviše komplikovan za svakodnevnu upotrebu, Hill je svojevremeno konstruisao mašinu za šifrovanje prema njegovom algoritmu. Mašina je koristila serije lančanih zupčanika. Međutim mašinu nikada nije uspeo da proda.

Hillov veliki doprinos predstavlja upotreba matematike u cilju osmišljavanja i analize kriptografskih mehanizama. Bitno je napomenuti da analiza ovog algoritma zahteva primenu jedne grane matematike pod nazivom "*Teorija brojeva*".

10. PRIMER:

Šifrovanje i dešifrovanje sa Hilovom šifrom.

Primer će se oslanjati na linearnu algebru i teoriju brojeva. Prikazan je proces šifrovanja i dešifrovanja.

Postupak šifrovanja sa Hilovom šifrom:

Ključ: matrica K [4 X 4];

$$K = \begin{bmatrix} 10 & 13 & 23 & 13 \\ 03 & 03 & 03 & 01 \\ 20 & 16 & 21 & 21 \\ 17 & 02 & 02 & 01 \end{bmatrix}$$

Otvoreni tekst: „STUDENTI”

$$\text{„STUD”}_{\text{blok1}} = \begin{bmatrix} 18 \\ 19 \\ 20 \\ 3 \end{bmatrix}, \text{„ENTI”}_{\text{blok2}} = \begin{bmatrix} 4 \\ 13 \\ 19 \\ 8 \end{bmatrix};$$

Šifrat blok₁: $(K * OT_blok_1) \bmod(26) = [AVXO]$

$$\begin{bmatrix} 10 & 13 & 23 & 13 \\ 03 & 03 & 03 & 01 \\ 20 & 16 & 21 & 21 \\ 17 & 02 & 02 & 01 \end{bmatrix} \times \begin{bmatrix} 18 \\ 19 \\ 20 \\ 3 \end{bmatrix} = \begin{bmatrix} 826 \\ 174 \\ 1147 \\ 387 \end{bmatrix} = (\bmod 26) \begin{bmatrix} 16 \\ 18 \\ 3 \\ 23 \end{bmatrix} = \begin{bmatrix} Q \\ S \\ D \\ X \end{bmatrix}$$

Šifrat blok₂: $(K * OT_blok_2) \bmod(26) = [WMXK]$

$$\begin{bmatrix} 10 & 13 & 23 & 13 \\ 03 & 03 & 03 & 01 \\ 20 & 16 & 21 & 21 \\ 17 & 02 & 02 & 01 \end{bmatrix} \times \begin{bmatrix} 4 \\ 13 \\ 19 \\ 8 \end{bmatrix} = \begin{bmatrix} 750 \\ 116 \\ 855 \\ 140 \end{bmatrix} = (\bmod 26) \begin{bmatrix} 22 \\ 12 \\ 23 \\ 10 \end{bmatrix} = \begin{bmatrix} W \\ M \\ X \\ K \end{bmatrix}$$

Šifrat: [AVXOWMXK]

Proces dešifrovanja sa Hilovom šifrom:

$$"AVXO"_{\text{blok1}} = \begin{bmatrix} 16 \\ 18 \\ 3 \\ 23 \end{bmatrix}, "WMXK"_{\text{blok2}} = \begin{bmatrix} 22 \\ 12 \\ 23 \\ 10 \end{bmatrix};$$

Ključ: inverzna matrica [4 X 4];

$$K^{-1} = \begin{bmatrix} 5 & 13 & 16 & 2 \\ 19 & 14 & 19 & 16 \\ 25 & 13 & 23 & 11 \\ 9 & 11 & 8 & 17 \end{bmatrix}$$

Otvoreni tekst blok₁: $(K^{-1} * \text{Šifrat_blok}_1) \bmod(26) = [\text{STUD}]$

$$\begin{bmatrix} 5 & 13 & 16 & 2 \\ 19 & 14 & 19 & 16 \\ 25 & 13 & 23 & 11 \\ 9 & 11 & 8 & 17 \end{bmatrix} \times \begin{bmatrix} 16 \\ 18 \\ 3 \\ 23 \end{bmatrix} = \begin{bmatrix} 408 \\ 981 \\ 956 \\ 757 \end{bmatrix} = (\bmod 26) \begin{bmatrix} 18 \\ 19 \\ 20 \\ 3 \end{bmatrix} = \begin{bmatrix} S \\ T \\ U \\ D \end{bmatrix}$$

Otvoreni tekst blok₂: $(K^{-1} * \text{Šifrat_blok}_2) \bmod(26) = [\text{ENTI}]$

$$\begin{bmatrix} 5 & 13 & 16 & 2 \\ 19 & 14 & 19 & 16 \\ 25 & 13 & 23 & 11 \\ 9 & 11 & 8 & 17 \end{bmatrix} \times \begin{bmatrix} 22 \\ 12 \\ 23 \\ 10 \end{bmatrix} = \begin{bmatrix} 654 \\ 1183 \\ 1345 \\ 654 \end{bmatrix} = (\bmod 26) \begin{bmatrix} 4 \\ 13 \\ 19 \\ 8 \end{bmatrix} = \begin{bmatrix} E \\ N \\ T \\ I \end{bmatrix}$$

Otvoreni tekst: "STUDENTI"

2.3.4. KRIPTOANALIZA HILOVE ŠIFRE

Linerane šifre su dosta slabe. Linearnost sama po sebi uvodi određene pravilnosti na osnovu kojih je moguće konstruisati kriptanalitički metod, budući da se linearne jednačine mogu lako rešiti. Jake šifre moraju imati nelinearnost, ali linearne komponente su korisne. Šifra ne sme biti u potpunosti linearna. Kriptanalitičari pokušavaju da aproksimiraju nelinearne delove algoritma sa linearnim jednačinama.

2.4. POLIALFABETSKE ZAMENE

Polialfabetaska zamena predstavlja sistem za šifrovanje koji eliminiše preslikavanje originalnih frekvencija pojavljivanja slova u otvorenom tekstu u njihove supstitucione zamene u šifratu, korišćenjem višestrukih supstitucija.

Vižnerova šifra predstavlja polialfabetSKU šifru zamene. Metod je prvobitno opisao Đovan Batista (Giovan Battista Bellaso) 1553 godine. Međutim šema algoritma je kasnije pripisana Vižneru (Blasie de Vigenere) u 19. veku i od tada je šifra poznata kao

"Vižnerova šifra". Šifra je imala veliku primenu kod klasičnih šifarskih sistema. Korišćena je u Drugom svetskom ratu. Vižnerova šifra predstavlja "Multi-Cezarovu" šifru. U Vižnerovoj šifri svako slovo otvorenog teksta se preslikava u jedno od n mogućih slova (k -dužina ključne reči) u zavisnosti od pozicije u tekstu.

Ključ Vižnerove šifre je u formi $K = (k_0, k_1, \dots, k_{n-1})$ gde je svako slovo iz skupa vrednosti $k_i \in \{0, 1, 2, \dots, 25\}$. Proces šifrovanja definisan je na sledeći način: $c_i = p_i + k_{i(\text{mod } n)} \pmod{26}$, a proces dešifrovanja: $p_i = c_i + k_{i(\text{mod } n)} \pmod{26}$.

Vižnerov algoritam šifrovanja možemo interpretirati kao ponavljajuću sekvencu (sa pomerajem k) prostih zamena.

11. PRIMER:

Šifrovanje i dešifrovanje sa Vižnerovom šifrom.

Ključ: SINGI, ključ je, $K = (18, 8, 13, 6, 8)$

Otvoreni tekst: [SINGIDUNUM]

Šifrovanje:

	S	I	N	G	I	D	U	N	U	M	
	18	8	13	6	8	3	20	13	20	12	
+	S	I	N	G	I	S	I	N	G	I	
	18	8	13	6	8	18	8	13	6	8	mod(26)
	10	16	0	12	16	21	2	0	0	20	
	K	Q	A	M	Q	V	C	A	A	U	

Šifrat: [KQAMQVCAAU]

Dešifrovanje:

	K	Q	A	M	Q	V	C	A	A	U	
	10	16	0	12	16	21	2	0	0	20	
-	S	I	N	G	I	S	I	N	G	I	
	18	8	13	6	8	18	8	13	6	8	mod(26)
	18	8	13	6	8	3	20	13	20	12	
	S	I	N	G	I	D	U	N	U	M	

Šifrovanje i dešifrovanje sa Vižnerovom šifrom moguće je obaviti i preko Vižnerove kvadratne matrice za isti otvoreni tekst i ključnu reč, primer je dat na slici.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Slika 2.6 Vižnerova kvadratna matrica za šifrovanje

2.4.1. KRIPTOANALIZA VIŽNEROVE ŠIFRE

Iz ovog prethodnog primera, Slika 2.6. za Vižnerovu šifru, zaključujemo da šifra predstavlja samo niz od dve šifre prostih zamena. Kriptoanalitičar bi trebalo da izvede samo k napada na prostu zamenu, preduslov je da poseduje dovoljno šifrata i da odredi k (dužinu ključa).

Za određivanje dužine ključne reči uveden je metod Indeks koincidencije. Ukoliko je šifrat sačinjen od engleskog alfabeta, imamo: n_0 broj **A**-ova, n_1 broj **B**-ova,..., n_{25} broj **Z**-etova u šifratu, a n je jednako: $n = n_0 + n_1 + \dots + n_{25}$. Indeks koincidencije onda definišemo na sledeći način:

$$I = \frac{\frac{n_0}{2} + \frac{n_1}{2} + \dots + \frac{n_{25}}{2}}{\frac{n}{2}} = \frac{1}{n(n-1)} \sum_{i=0}^{25} n_i(n_i-1)$$

Šta ova veličina meri? Indeks koincidencije daje verovatnoću da su dva slučajno izabrana slova identična. Za otvoreni engleski tekst, verovatnoća da su dva slova ista je jednaka:

$$p_0^2 + p_1^2 + \dots + p_{25}^2 \approx 0.065$$

gde je p_i verovatnoća i^{log} slova. Stoga je za šifrat koji je nastao prostom zamenom, $I \approx 0.065$. Ako bi slova bila sa ravnomernim verovatnoćama, $p_i = \frac{1}{26}$, tada bi $p_0^2 + p_1^2 + \dots + p_{25}^2 \approx 0.03846$. Ujedno $I \approx 0.03846$ važi za polialfabetske šifre vrlo dugačkih ključnih reči.

Sad kad imamo ove rezultate možemo da odredimo dužinu ključne reči u Vižnerovoj šifri. Ako je dužina ključne reči k , a dužina poruke n , u tom slučaju upisujemo šifrat u matricu sa k kolona i n/k redova. Selektujemo dva slova iz istih kolona, ekvivalentno odabiru u slučaju proste zamene, zatim selektujemo dva slova iz različitih kolona, ekvivalentno odabiru slučajnih slova.

Kada pretpostavimo k kolona i n/k redova, određujemo približan broj poklopljenih parova iz iste kolone i dva različita reda:

$$0.065 \binom{\frac{n}{k}}{2} k = 0.065 \frac{1}{2} \binom{\frac{n}{k}}{k} \left(\frac{n}{k} - 1 \right) k = 0.065 \left(\frac{n(n-k)}{2k} \right)$$

Približan broj poklopljenih parova iz dve različite kolone i bilo koja dva reda, dat je izrazom:

$$0.03846 \binom{k}{2} \left(\frac{n}{k} \right)^2 = 0.03846 \frac{n^2 (k-1)}{2k}$$

Približna vrednost indeksa koincidencije, data je izrazom:

$$I \approx \frac{0.03846 \frac{n^2(k-1)}{2k} + 0.065 \left(\frac{n(n-k)}{2k} \right)}{\left(\frac{n}{2} \right)}$$

$$= \frac{0.03846n(k-1) + (0.065)(n-k)}{k(n-1)}$$

Rešenje po koloni k daje:

$$k \approx \frac{0.2654n}{(0.065 - I) + n(I - 0.03846)}$$

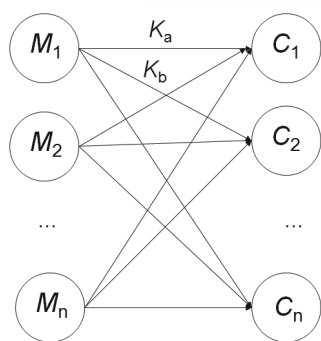
Uzima se n i I (poznato iz šifrata) u cilju aproksimacije dužine ključne reči Vižnerove šifre. Kada je pronađen Indeks koincidencije (William F. Friedman, 1920), predstavljao je revoluciju u kriptanalizi. Vrlo koristan za kriptanalizu klasičnih šifara iz doba Drugog svetskog rata. U statistici je test incidencija i koincidencija dobro poznat, postoje i mnogi drugi korisni statistički testovi.

Drugi metod za određivanje dužine ključne reči naziva se Kasiskijev test i uveo ga je Fridrih Kasiski (Friedrich Kasiski) 1863. godine. Metod se zasniva na činjenici da će dva identična dela otvorenog teksta biti šifrovana na isti način (istim delom ključne reči) ukoliko se njihove početne pozicije razlikuju za neki umnožak od k , gde je k dužina ključa. Obrnuto, ako uočimo dva identična dela u šifratu, dužine minimum 3 slova, tada je vrlo verovatno da oni odgovaraju identičnim delovima otvorenog teksta.

Kasiskijevim testom u šifratu pronalazimo parove identičnih delova dužine od minimum 3 slova, ako postoje takvi parovi, određujemo rastojanje između početnih pozicija. Izračunaćemo njihova rastojanja $k_1, k_2, k_3, \dots, k_i$, a potom pretpostaviti da k deli bar većinu k_i -ova. Kada smo odredili dužinu ključne reči, dobili smo sličnu situaciju kao kod kriptanalize Cezarove šifre. Ako posmatramo samo ona slova koja su šifrovana pomakom za k_1 slova, onda su ona šifrovana Cezarovom šifrom.

2.5. PERFEKTNE ŠIFRE

Pojam „perfektna šifara“ uveo je Klod Šenon (Claude Shannon) 1949. godine. Perfektna šifra predstavlja kriptografski mehanizam kod koga šifrat ne daje nikakvu informaciju o otvorenom tekstu, niti daje informaciju o ključu koji je korišćen. Tačnije, pretpostavimo da se u prostoru otvorenih tekstova P , otvoreni tekst x pojavljuje sa verovatnoćom $p(x)$, u tom slučaju postoji ključ koji preslikava svaku poruku u svaki šifrat sa jednakom verovatnoćom.



Slika 2.7 Ilustrovana definicija perfektnih šifara

Šenon koristi entropiju kao veličinu koja meri prosečnu količinu informacije sadržane u porukama. Za optimalno šifrovanje poruka koje se šalju preko komunikacionog kanala važno je opisati i količinu informacije koja je sadržana u pojedinačnim porukama. Količina informacije koja je sadržana u pojedinačnim porukama direktno zavisi od verovatnoće. Što je verovatnoća poruke veća, količina informacije u poruci je manja i što je verovatnoća poruke manja, količina informacije u poruci je veća. Funkcija koja zadovoljava oba zahteva je logaritam. U tom slučaju informacioni sadržaj neke poruke x se definiše na sledeći način:

$$I_x = \log_2 \frac{1}{p_x} = -\log_2 p_x, \quad p_x \text{ predstavlja verovatnoću poruke } x.$$

Primićemo da se koristi binarni logaritam, logaritam sa osnovom dva, što znači da količina informacije koja je sadržana u poruci predstavlja broj bita koji su potrebni za opis poruke. Jedinica mere za količinu informacije u ovom slučaju je 1 bit.

Ukoliko imamo poruku P koja sadrži x_i nezavisnih simbola sa verovatnoćama p_i , $i=0, \dots, L-1$, entropiju onda možemo da definišemo kao prosečnu količinu informacije koja je sadržana u pojedinačnim porukama i data je izrazom:

$$H = -\sum_{i=0}^{L-1} p_i \log p_i$$

12. PRIMER:

Poruka P sadrži simbole verovatnosnog skupa simbola $S = \{x_1, x_2, x_3\}$, čije su verovatnoće $p_{x_i} = \{0.25, 0.25, 0.50\}$. Odrediti (1) pojedinačnu količinu informacije sadržanu u simbolima poruke P , (2) ukupnu količinu informacije u poruci P i (3) entropiju poruke P .

(1) Pojedinačne količine informacije

$$I_{x_1} = \log_2(1/p_{x_1}) = \log_2(1/0.25) = 2 \text{ bita}$$

$$I_{x_2} = \log_2(1/p_{x_2}) = \log_2(1/0.25) = 2 \text{ bita}$$

$$I_{x_3} = \log_2(1/p_{x_3}) = \log_2(1/0.50) = 1 \text{ bit}$$

(2) Ukupna količina informacije u poruci P

$$I_P = I_{x_1} + I_{x_2} + I_{x_3} = 2 + 2 + 1 = 5 \text{ bitova}$$

(3) Entropija poruke P

$$\begin{aligned} H &= -p_{x_1} \log_2(p_{x_1}) - p_{x_2} \log_2(p_{x_2}) - p_{x_3} \log_2(p_{x_3}) = \\ &= -0.25 \log_2(0.25) - 0.25 \log_2(0.25) - 0.5 \log_2(0.5) = \\ &= 0.5 + 0.5 + 0.5 = 1.5 \text{ bitova} \end{aligned}$$

Ukoliko imamo poruku P koja je u formi binarnog niz, tada entropija zavisi od verovatnoće poruka 0 i 1. Ako je verovatnoća prve poruke jednaka p , onda je

verovatnoća druge poruke $(1-p)$. Entropija binarnih nizova je definisana sledećim izrazom:

$$H = -p \log_2 p - (1-p) \log_2 (1-p)$$

13. PRIMER:

(1) Odrediti entropije H za dva binarna niza čije su verovatnoće poruka $BN_1 = \{0.25, 0.75\}$ i (2) $BN_2 = \{0.50, 0.50\}$, dužina nizova je 1000 bitova. Zatim odrediti (3) ukupnu količinu informacije sadržane u svakom nizu.

(1) Entropija prvog niza:

$$\begin{aligned} H_{BN1} &= 0.25 \log_2 (0.25) - (1-0.25) \log_2 (1-0.25) = \\ &= 0.8113 \text{ bit} \end{aligned}$$

(2) Entropija drugog niza:

$$\begin{aligned} H_{BN2} &= 0.5 \log_2 (0.5) - (1-0.5) \log_2 (1-0.5) = \\ &= 1 \text{ bit} \end{aligned}$$

(3) Ukupna količina informacije za BN_1 i BN_2 :

$$I_{BN1} = H_{BN1} * 1000 = 0.8113 * 1000 = 811,3$$

$$I_{BN2} = H_{BN2} * 1000 = 1 * 1000 = 1000$$

Iz ovog primera možemo da zaključimo da binarni niz BN_1 od ukupno 1000 bitova, sadrži 811 bitova informacije i 189 bitova redundantse, dok niz BN_2 identične dužine, sadrži 1000 bitova informacije i 0 bitova redundantse. Binarni niz BN_2 je informacijski jači i njega ćemo verovatno koristiti za ključ kod OTP šifre.

Ovaj primer predstavlja samo prvi korak analize slučajnosti kod binarnih nizova. Pretpostavimo slučaj u kome niz BN_2 sadrži redom 500 bitova jedinica, a potom 500 bitova nula. Verovatnoće nula i jedinica će davati maksimalnu vrednost entropije $H \approx 1$. Da li je ovo slučajan niz? Naravno da nije, neophodno je proveriti uniformnost ovog binarnog niza. Pređimo na sledeći primer.

14. PRIMER:

(1) Odrediti entropiju binarnog niza BN_1 dužine 50 bitova u odnosu na njegovu uniformnu distribuciju. Niz je generisan na slučajan način, a kao izvor korišćen je atmosferski šum.

$$BN_1 = [1000011010001100011101100010101011110001010100010]$$

U prvom koraku, prebrojaćemo sve parove sačinjene od nula i jedinica. Ukupan broj parova je 4. Parove ćemo tretirati kao poruke ovog verovatnosnog skupa $S = \{00, 01, 10, 11\}$.

- (1) Vrednosti koje smo predstavili sa p_{s_i} su verovatnoće poruka S u binarnom nizu.

$$p_{s_{00}} = 0.24, p_{s_{01}} = 0.12, p_{s_{10}} = 0.48, p_{s_{11}} = 0.16;$$

- (1) Entropiju sada određujemo na način kao u primeru 13, zatim dobijenu vrednost delimo sa dva da bismo dobili vrednost entropije za jedan bit.

$$\begin{aligned} H &= -p_{s_{00}} \log_2(p_{s_{00}}) - p_{s_{01}} \log_2(p_{s_{01}}) - p_{s_{10}} \log_2(p_{s_{10}}) - p_{s_{11}} \log_2(p_{s_{11}}) = \\ &= -0.24 \log_2(0.24) - 0.12 \log_2(0.12) - 0.48 \log_2(0.48) - 0.16 \log_2(0.16) = \\ &= 1.7925 / 2 = 0.8962 \text{ bit} \end{aligned}$$

Rezultat H od 0,8962 bita predstavlja količinu informacije koja je sadržana po jednom bit-u u datom binarnom nizu. Ako pomnožimo ukupan broj bita iz niza sa dobijenom vrednosti entropije $H=0,8962$, dobićemo ukupno 44.81 bita informacije. Možemo da zaključimo da ovaj niz nije apsolutno slučajan, ako je jedinica mere za količinu informacije 1 bit.

Sada nam je možda malo jasnije zašto je toliko teško pronaći informacioni izvor koji će generisati čisto slučajne binarne nizove. U praksi je to uglavnom slučaj da se koriste pseudo slučajni generatori. Pseudo generatori otklanjaju sva prirodna nesavršenstva pravih slučajnih nizova. U tom slučaju dobijamo i pseudo sigurnost. Za potvrdu ispravnosti pseudo generatora, neophodno je sprovesti niz testova koje je predložio NIST, za proveru postojanja osbina slučajnih nizova.

Perfektni šifarski sistemi podrazumevaju bezuslovnu ili teorijsku sigurnost u odnosu na protivnika koji raspolaže neograničenim vremenom i računarskim resursima. Danas se ova sigurnost naziva informaciono-teorijska sigurnost.

Dokaziva bezuslovna sigurnost znači da se za zadati kriptografski sistem može dokazati nominovana sigurnost u odnosu na protivnika koji raspolaže neograničenim vremenskim i računarskim resursima.

Nominovana sigurnost jeste pažljiv i kompletan opis operacionog scenarija u kome se egzaktno definiše:

- Od kakvih napada kriptografski sistem treba da obezbedi zaštitu;
- Šta je poznato napadaču;
- Šta je poznato korisnicima sistema;
- Sve fizičke pretpostavke, koje bliže određuju šta napadač i korisnici sistema mogu da urade.

2.5.1. ONE-TIME PAD

One-time pad (OTP) je jedina dokazano sigurna šifra. Niti jedna druga šifra koju ćemo razmatrati nije dokazano sigurna. OTP je nepraktična za većinu situacija, ali ima svoju primenu. OTP su uveli Gilber Vernam (Gilbert Vernam) i Džozef Mauborgn (Joseph Mauborgne) 1917. godine. Upotrebom Šenonofih definicija koje važe za perfektne šifarske sisteme, bilo je moguće potvrditi da se radi o savršenom kriptografskom mehanizmu. Za razliku od dosadašnjih šifara o kojima smo govorili, OTP radi direktno nad binarnim podacima.

Šifra radi na sledećem principu. Šifrat se iz otvorenog teksta dobije tako što se primeni operacija "ekskluzivno ili" (XOR) za otvoreni tekst i ključ $C = P \oplus K$, svi podaci su u binarnom formatu. Postupak dešifrovanja je identičan postupku šifrovanja: $P = C \oplus C$.

15. PRIMER:

Šifrovanje i dešifrovanje sa OTP šifrom i kriptanaliza.

Šifrovanje: Otvoreni tekst ("SINGIDUNUM") $\oplus$ ključ = šifrat (NGDUUMSIGD)

Otvoreni tekst	S	I	N	G	I	D	U	N	U	M
	000	001	010	011	001	100	101	010	101	110
Ključ	010	010	110	110	100	010	101	011	110	010
Šifrat	010	011	100	101	101	110	000	001	011	100
	N	G	D	U	U	M	S	I	G	D

Dešifrovanje: Šifrat (NGDUUMSIGD) $\oplus$ ključ = otvoreni tekst (SINGIDUNUM)

Šifrat	N	G	D	U	U	M	S	I	G	D
	010	011	100	101	101	110	000	001	011	100
Ključ (pravi)	010	010	110	110	100	010	101	011	110	010
Otvoreni tekst	000	001	010	011	001	100	101	010	101	110
	S	I	N	G	I	D	U	N	U	M

Dešifrovanje: Šifrat (NGDUUMSIGD) $\oplus$ ključ_X = otvoreni tekst (GINISUNISU).
Kriptanaliza, isprobavanje slučajnih ključeva na osnovu rečnika.

Šifrat	N	G	D	U	U	M	S	I	G	D
	010	011	100	101	101	110	000	001	011	100
Ključ (lažni)	001	010	110	100	101	011	010	000	011	001
Otvoreni tekst	011	001	010	001	000	101	010	001	000	101
	G	I	N	I	S	U	N	I	S	U

Sigurnost OTP se zasniva na slučajnosti ključa, kao i na jednokratnoj upotrebi ključa, U praksi je jako teško generisati čisto slučajne nizove koji će biti kvalitativno verifikovani sa aspekta teorijsko informacione analize. Sa kriptografskog stanovišta, zahtevano osnovno svojstvo binarnog slučajnog ključa određuje se entropijom H . Kao što smo već naveli, entropija kod binarnih sekvenci predstavlja informaciju po

jednom bitu. Poželjna vrednost entropije je $H \approx 1$. Ova vrednost entropije važi za verovatnoće pojavljivanja nula i jedinica $p_{1/0} \{ \sim 0.5, \sim 0.5 \}$. Naravno, sve ovo važi pod uslovom da postoji uniformna distribucija unutar binarne sekvence.

OTP je dokazano bezbedna kada se koristi po definisanim pravilima:

- Šifrat ne daje nikakvu informaciju o otvorenom tekstu;
- Svaki otvoreni tekst iste dužine je podjednako verovatan;
- Ključ mora biti "slučajan", može se koristiti samo jedanput;
- Ključ je poznat samo pošiljaocu i primaocu;
- Ključ je iste dužine kao i poruka;
- Mehanizam integriteta ne postoji.

16. PRIMER:

Predstavljen je slučaj sa ponovljenim ključem kod OTP-a. Kriptoanalitičar saznaje otvoreni tekst druge poslate poruke na osnovu snimljenih šifrata za obe poruke i poznatog otvorenog teksta prve poruke.

Operacija 1		Operacija 3			
Otvoreni tekst A	11010011	Šifrat A	01110101		
	XOR		XOR		
Ključ:	10100110	Šifrat B	10001011		
Šifrat A	01110101		11111110	Operacija 5	
				Otvoreni tekst A	11010011
					XOR
				Šifrat A + Šifrat B	11111110
				Otvoreni tekst B	00101101
Operacija 2		Operacija 4			
Otvoreni tekst B	00101101	Otvoreni tekst A	11010011		
	XOR		XOR		
Ključ:	10100110	Otvoreni tekst B	00101101		
Šifrat B	10001011		11111110		

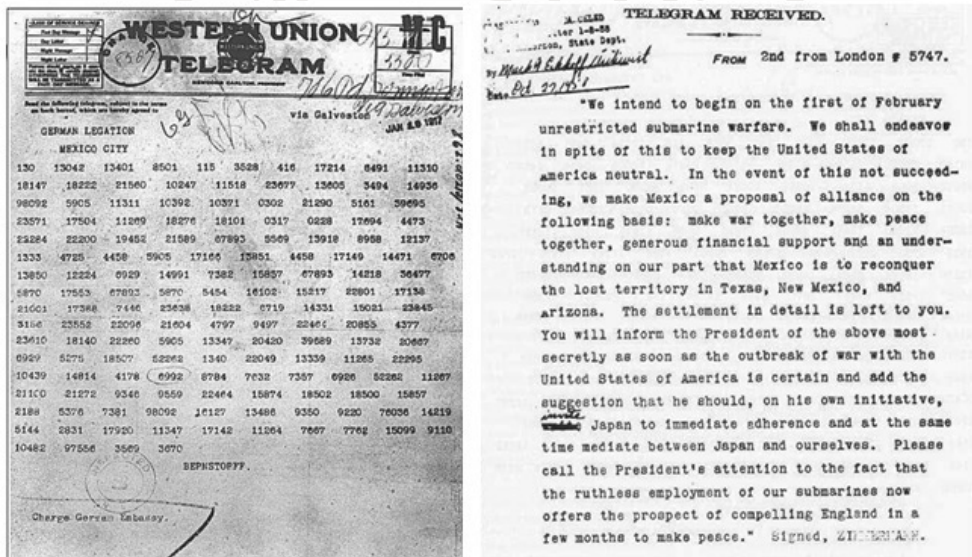
Projekat "VENONA" predstavlja One-time pad u realnom životu. Sovjetska špijunska mreža formirana je na teritoriji SAD četrdesetih godina sredinom prošlog veka. U pitanju je bila nuklearna špijunaža. Za jednu ovakvu akciju neophodno je bilo razmeniti na hiljade šifrovanih poruka. Međutim, sovjetski špijuni su uneli ključeve

prilikom ulaska u SAD. Šifru su koristili na pravilan način iz sigurnosnih razloga. Ponavljanje ključa daje realnu mogućnost za probijanje poruka.

2.6. KODNE KNJIGE

Kodna knjiga je rečnik koji uparuje reči (frazе) i kodne oznake. Preciznije rečeno, postoje dve kodne knjige, jedna se koristi za šifrovanje a druga za dešifrovanje. Ključ ove šifre je kodna knjiga. Sigurnost ovog šifarskog sistema se zasniva na fizičkoj bezbednosti kodne knjige. Kodne knjige su imale veliku primenu u Drugom svetskom ratu. Mnoge savremene šifre su zasnovane na ovom principu.

Cimermanov telegram je šifrovan kodnom knjigom. Cimermanov telegram je jedna od najpoznatijih kodnih knjiga u istoriji. Dešifrovanje Cimermanovog telegrama bio je povod za ulazak SAD u Prvi svetski rat. Slika 2.8, prikazan je originalni šifrat i dešifrovan poruka.



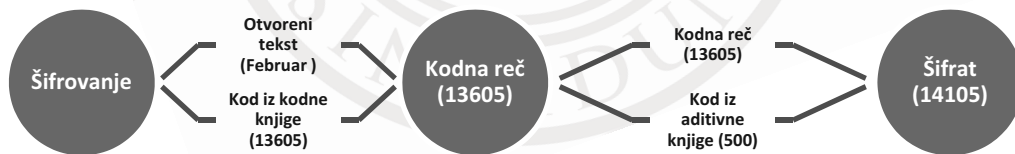
Slika 2.8 Cimermanov telegram (šifrat i otvoreni tekst)

17. PRIMER:

Šifarnik Cimermanovog telegrama. Za svaku reč je bila određena šifra.

Reči/Šifre
Februar/13605
Fest/13732
Finanzielle/13850
Folgender/13918
Frieden/17142
Friedenschluus/17149

Britanci su uz pomoć Rusa delom rekonstruisali kodnu knjigu, a zatim dešifrovali delove telegrama. Na slici ispod nalazi se jedan deo telegrama. Kodne knjige su podložne statističkoj analizi. Slično kao i kod šifara zamene, ali je za napad potrebno mnogo više šifrata. U toku Drugog svetskog rata, kodne knjige su bile veoma popularne. Problem je bila skupa zamena, a sa druge strane i teško izvodljiva. Da bi ostale u upotrebi, uvedene su dodatne aditivne kodne knjige. Aditivna kodna knjiga je dodatna knjiga koja sadrži mnogo slučajnih brojeva. Sekvence slučajnih brojeva se sabiraju sa kodnim rečima i formiraju konačni šifrat. Početna pozicija iz aditivne knjige treba da se dogovori između pošiljaoca i primaoca.



Slika 2.9 Primer šifrovanja sa aditivnom kodnom knjigom

Kod upotrebe aditivnih kodnih knjiga početno mesto u kodnoj knjizi određuje pošiljalac. Određenom mestu šalje se nezaštićeno "otvoreno" uz šifrat. Drugi naziv za početno mesto je indikator poruke. Za dešifrovanje, pored ključa neophodan je i indikator poruke. Savremeni izraz koji se koristi za indikator poruke kod modernih šifara je "inicijalni vektor". Na ovaj način se povećava sigurnost kodne knjige. Ako se indikator poruke koristi samo jednom za jednu poruku, šifra postaje One-time pad.

3. SIMETRIČNI ŠIFARSKI SISTEMI

U klasičnoj kriptografiji koriste se takozvani "papir olovka algoritmi". Koristio se isti ključ za šifrovanje i dešifrovanje (simetrični ključ). Otvoreni tekst i šifrat se uglavnom predstavljao sa slovima. Krajem Drugog svetskog rata, uvedeni su računari u kriptanalizu, što je dalje indukovalo i kod kriptografa upotrebu računara za šifrovanje i dešifrovanje.

Moderni algoritmi podrazumevaju fazu kodovanja, otvoreni tekst se pre šifrovanja prvo predstavi preko binarnih nizova ili bitova. Način kodovanja nije tajna, koristi se ASCII tabela. ASCII tabela je skraćenica od „Američki standardni kod za razmenu podataka“. Algoritmi za šifrovanje i dešifrovanje sada rade na nizovima bita, a tako smo dobili digitalno šifrovanje.

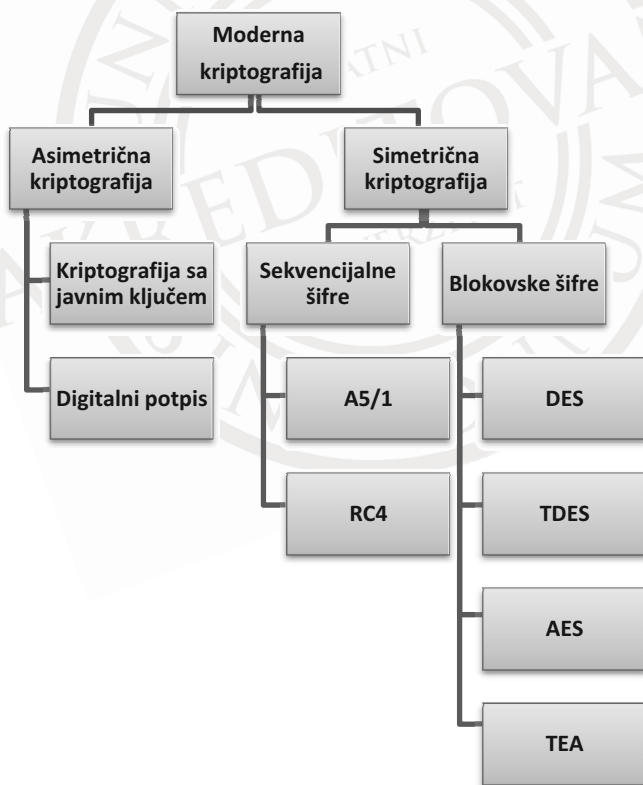
Šifrovanje pomoću računara je u osnovi slično klasičnim oblicima šifrovanja. I dalje se koriste kombinacije transpozicije "premeštanja" i supstitucije "zamene". Postoje brojne prednosti moderne kriptografije u odnosu na klasičnu kriptografiju od kojih izdvajamo dve najbitnije. Prva prednost je u tome da je moguće realizovati daleko složenije šifre primenom računara za razliku od onih koje se realizuju mehaničkim putem. Druga, elektronika je neuporedivo brža od mehanike što utiče na brzinske performanse algoritama za šifrovanje i dešifrovanje.

U ovom poglavlju razmatraju se dve vrste simetrične kriptografije, sekvencijalni i blokovski šifarski sistemi. Sekvencijalne šifre su slične One-time pad-u. Zasnivaju se na generatorima pseudoslučajnih nizova koji se inicijalizuju na bazi tajnog ključa relativno male dužine. Generisani pseudoslučajni niz ima statističke karakteristike koje su slične karakteristikama pravih slučajnih nizova. Jedina je razlika u tome što se pseudoslučajni nizovi ponavljaju posle odgovarajuće periode dugog trajanja.

Blokovske šifre se mogu posmatrati kao moderni naslednici klasičnih kodnih knjiga gde ključ određuje kodnu knjigu. Unutrašnji rad algoritama savremenih blokovskih šifara je kompleksan. Kod blokovskih šifara, otvoreni tekst se deli na blokove fiksne i zadate dužine (64, 128, 192, 256) u bitovima, zatim se svaki blok nezavisno šifrjuje. Interno, blokovske šifre primenjuju i konfuziju i difuziju.

Razmotrimo principe rada dva poznata algoritma sekvencijalnih šifara: A5/1 i RC4. Algoritam A5/1 korišćen je u mobilnoj telefoniji i predstavlja klasu sekvencijalnih šifara zasnovanih na hardveru i takozvanim neuniformno taktovanim linearnim pomeračkim registrima. RC4 se koristi na mnogim mestima, uključujući SSL i WEP protokole. RC4 je jedinstven među sekvencijalnim šiframa pošto je dizajniran tako da se efikasno implementira u softveru.

Kod blokovskih šifara razmotrićemo DES jer je relativno jednostavan (po standardu blokovskih šifri) i najstariji od svih, blokovska šifra sa kojom se sve ostale porede. Takođe, razmotriće se nekoliko drugih popularnih blokovskih šifara. Na kraju ćemo diskutovati o primeni blokovskih šifara za ostvarivanje poverljivosti i integriteta podataka. **Slika 3.1**, data je osnovna podela moderne kriptografije, sa šiframa koje predstavljaju istaknute predstavnike određenog tipa šifre.



Slika 3.1 Podela modernih šifara

3.1. SEKVENCIJALNE ŠIFRE

Sekvencijalne šifre uzimaju ključ K , dužine n bita i koriste ga za inicijalizaciju generatora pseudoslučajnih nizova. Dobijena pseudoslučajna sekvenca je po svojim statističkim osobinama slična pravom slučajnom nizu. U odnosu na njega razlikuje se po tome što je periodična. Periodičnost nekog generatora pseudoslučajnih nizova možemo najlakše razumeti ako zamislimo da se krećemo kružnom putanjom i u jednom momentu ćemo se naći na početku. Isto tako pseudoslučajni generator će početi da generiše vrednosti u nekom momentu ponovo iz početka. Broj pseudoslučajnih vrednosti će zavisiti upravo od dužine te kružne putanje.

Neki autori sekvencijalne šifre nazivaju i pseudo-*One-time pad* šifre. Da bi smo bolje razumeli počecemo od Šenona (*Claude Shannon*) koji je definisao dva tipa sigurnosti. Prva je bezuslovna sigurnost, što znači da se za zadati kriptografski sistem može rigorozno dokazati nominovana sigurnost u odnosu na protivnika koji raspolaže neograničenim vremenskim i računarskim resursima. Šifra koja odgovara datom opisu bezuslovne sigurnosti je *One-time pad* i predstavlja Šenonov ideal apsolutne sigurnosti.

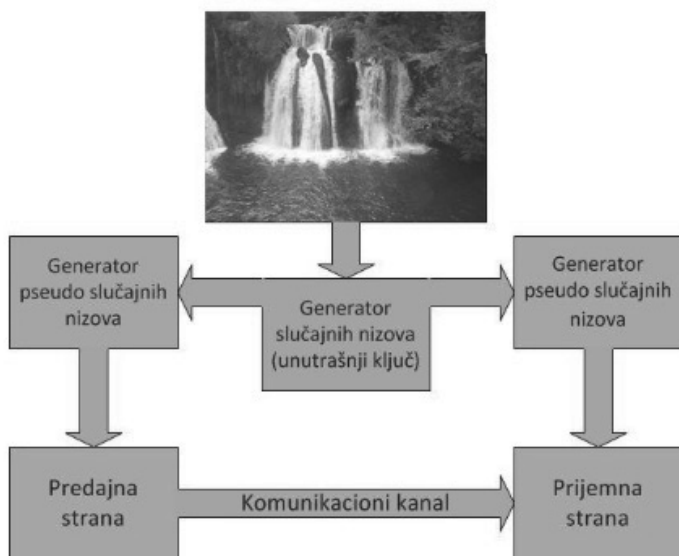


Slika 3.2 Šenonov ideal sigurnosti

Slika 3.2, prikazan je prirodni informacioni izvor koji se koristi kao generator slučajnih nizova. Prave generatore slučajnih nizova moguće je pronaći u prirodi. Ujedno, dobijeni slučajni niz iz datog informacionog izvora predstavlja ključ K koji se koristiti za šifrovanje na način koji je objašnjen kod *One-time pad* šifre.

Druga sigurnost koju je definisao Šenon je dokaziva računarska sigurnost, što znači da se za zadati kriptografski sistem može rigorozno dokazati nominovana sigurnost u odnosu na protivnika koji poseduje specifične vremenske i računarske resurse. Upravo šifra koja najviše odgovara definiciji računarske sigurnosti je simetrična sekvencijalna šifra i po Šenonu predstavlja pseudoideal računarske sigurnosti.

Slika 3.3, prikazan je prirodni informacioni izvor koji ima ulogu generatora slučajnih nizova koji se koriste kao inicijalna stanja pseudoslučajnih generatora za generisanje pseudoslučajnih nizova. Generisani pseudoslučajni nizovi koriste se za ključ K koji se primenjuje za šifrovanje na način koji je objašnjen kod *One-time pad* šifre.



Slika 3.3 Šenonov pseudo ideal

Proces šifrovanja se svodi na XOR operaciju nad bitovima otvorenog teksta i bitovima pseudoslučajne sekvence (radnog ključa). Dešifrovanje se vrši identičnim pseudoslučajnim nizom na prijemu koji se XOR-uje sa dobijenim šifratom C .

Korišćenje pseudoslučajne sekvence (eng. *keystream*) je identično korišćenju ključa kod *One-time pad*-a i naziva se radni ključ koji je generisan na osnovu ključa K .

Funkcija sekvencijalne šifre SK može se posmatrati kao:

$$SK(K) = S$$

gde je K ključ, a S predstavlja rezultujući pseudoslučajni niz.

Za dati pseudoslučajni niz $S = s_0 + s_1 + \dots + s_i$ i otvoreni tekst $P = p_0 + p_1 + \dots + p_i$ generiše se šifrat $C = c_0 + c_1 + \dots + c_i$. Primenom XOR operacije nad odgovarajućim bitovima dobija se šifrat C :

$$C = \{c_0 = p_0 \oplus s_0; c_1 = p_1 \oplus s_1; c_i = p_i \oplus s_i\};$$

U procesu dešifrovanja šifrata C , pseudoslučajna sekvencija S se ponovo koristi, odnosno XOR-uje sa bitovima šifrata:

$$P = \{p_0 = c_0 \oplus s_0; p_1 = c_1 \oplus s_1; p_i = c_i \oplus s_i\};$$

Poznato nam je da šifrat kod *One-time pad*-a nastaje tako što se primeni XOR operacija nad otvorenim tekstom i ključem. Iz ovoga vidimo da dužina ključa K mora biti ista dužini otvorenog teksta. Osnovna ideja je bila da se zameniti slučajni niz ili ključ K sa pseudoslučajnim nizom ili radnim ključem (ključ koji se koristi za šifrovanje). Ovo zahteva projektovanje generatora pseudoslučajnih brojeva (PRNG). Dobijeno je više rešenja za projektovanje PRNG-a. Jedno od rešenja je primena linearnih pomeračkih registara sa povratnom spregom (LPR).

Linearni pomerački registri obuhvataju dve funkcije. Prva se odnosi na funkciju pomeračkog registra dužine n bita koji se u svakom taktu pomera za jedno mesto u desno. Druga funkcija je funkcija povratne sprege koja određuje (na osnovu određenih bitova u registru) koji bit će se upisati na prvo mesto. Funkcija može biti linearna. Izlaz iz pomeračkog registra je jedan bit, najčešće krajnji desno. Izlazni bit je određen na osnovu početnog stanja i povratne sprege. Perioda pomeračkog registra je broj bita izlazne sekvence, pre nego što počne da se ponavlja.

Pomerački registar dužine n , može da ima $2^n - 1$ različitih početnih stanja, jedino ne mogu biti sve nule. Pravilnim odabirom povratne sprege može se postići perioda izlaznog niza od $2^n - 1$ bita. Generisani izlazni niz ima osobine pseudoslučajnog niza što znači da sadrži osobinu periodičnosti.

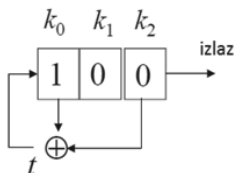
Nažalost, sadrži i osobine koje nisu prihvatljive, prvih n bitova izlaza su jednaki početnom stanju. Na osnovu poznavanja svih 2^n bita izlaza jednog generatora može se rekonstruisati generator. Sa aspekta kriptanalize, uspeh kriptanalitičara bio bi zagarantovan.

18. PRIMER:

Primer jednog LPR za generisanje radnog ključa.

Ključ $K : n = 3 \rightarrow k_0 k_1 k_2 = 100$

Takt registra je određen sa: $t = k_0 \oplus k_2, k_i = k_{i-1}, k_0 = t$, Slika 3.4. Dati takt je binarna reprezentacija polinoma $x^3 + x^2 + 1$.



Slika 3.4 Definicija jednog takta registra

Tabela 3 Vrednosti po taktu za stanje registra 100.

takt	stanje	izlaz	takt	stanje	izlaz	takt	stanje	izlaz
0	1 0 0	1	4	1 0 1	0	8	1 1 0	1
1	1 1 0	1	5	0 1 0	0	9	1 1 1	0
2	1 1 1	0	6	0 0 1	1	10	0 1 1	1
3	0 1 1	1	7	1 0 0	1	11	: : :	:

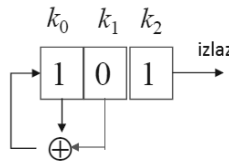
Tabela 3, sadrži stanja datog registra. Za svako stanje kao ulaz u naredni takt prikazan je izlaz veličine 1 bit kao izlaz iz takta. Izlazi zajedno čine radni ključ maksimalne dužine $2^n - 1$. Na osnovu vrednosti takta 7 za stanje 100 označeno crvenom bojom, vidimo da je registar periodičan. Njegova perioda je 7.

19. PRIMER:

Primer jednog LPR za generisanje radnog ključa sa lošom povratnom spregom koja prouzrokuje malu periodu.

Ključ $K: n=3 \rightarrow k_0 k_1 k_2 = 101$

Takt registra je određen sa: $t = k_0 \oplus k_1, k_i = k_{i-1}, k_0 = t$, Slika 3.5. Dati takt je binarna reprezentacija polinoma $x^2 + x + 1$.



Slika 3.5 Definicija jednog takta registra

Tabela 4 Vrednosti po taktu za stanje registra 101.

takt	stanje	izlaz	takt	stanje	izlaz	takt	stanje	izlaz
0	1 0 1	1	4	1 1 0	0	8	0 1 1	1
1	1 1 0	0	5	0 1 1	1	9	1 0 1	1
2	0 1 1	1	6	1 0 1	1	10	1 1 0	0
3	1 0 1	1	7	1 1 0	0	11	: : :	:

Tabela 4, prikazana su stanja datog registra. Slično kao i u prethodnom primeru, ali je druga definicija povratne sprege. Na osnovu vrednosti takta 3, 6 i 9 za stanje 101 označeno crvenom bojom, vidimo da je registar periodičan i da povratna sprega određuje njegovu periodu. Njegova perioda je 3.

Iz prethodnih primera videli smo da je izlaz LPR registra periodičan i da zavisi od povratne sprege. Radni ključ se ponavlja, možemo ga prihvatiti ako je perioda dovoljno velika i koristi se samo unutar jedne periode. U praksi bi značilo da perioda određuje momenat promene inicijalnog stanja ili ključa K . Primenjeni statistički testovi dali su elemente pseudoslučajnosti za ovako generisan radni ključ.

Upotrebljivost LPR registra za generisanje radnog ključa zavisi od dizajna koji kombinuje više LPR registara. Uz uvođenje elemenata nelinearnosti mogu se prevazići navedeni problemi. Prevazilaženje problema značilo bi možda računarsku, ali ne i bezuslovnu sigurnost.

Sa pretpostavkom da pošiljalac i primalac imaju isti algoritam za generisanje pseudoslučajne sekvence i da oba koriste isti ključ K koji se dostavlja posebnim sigurnim kanalima (kurirskim kanalima), ovaj sistem omogućava praktičnu generalizaciju *One-time pad*-a. Međutim, dobijeni šifrat nema osobinu apsolutne tajnosti i ne odgovara Šenonovom idealu sigurnosti, već govorimo o računarskoj tajnosti, odnosno sistem nije dokazivo siguran.

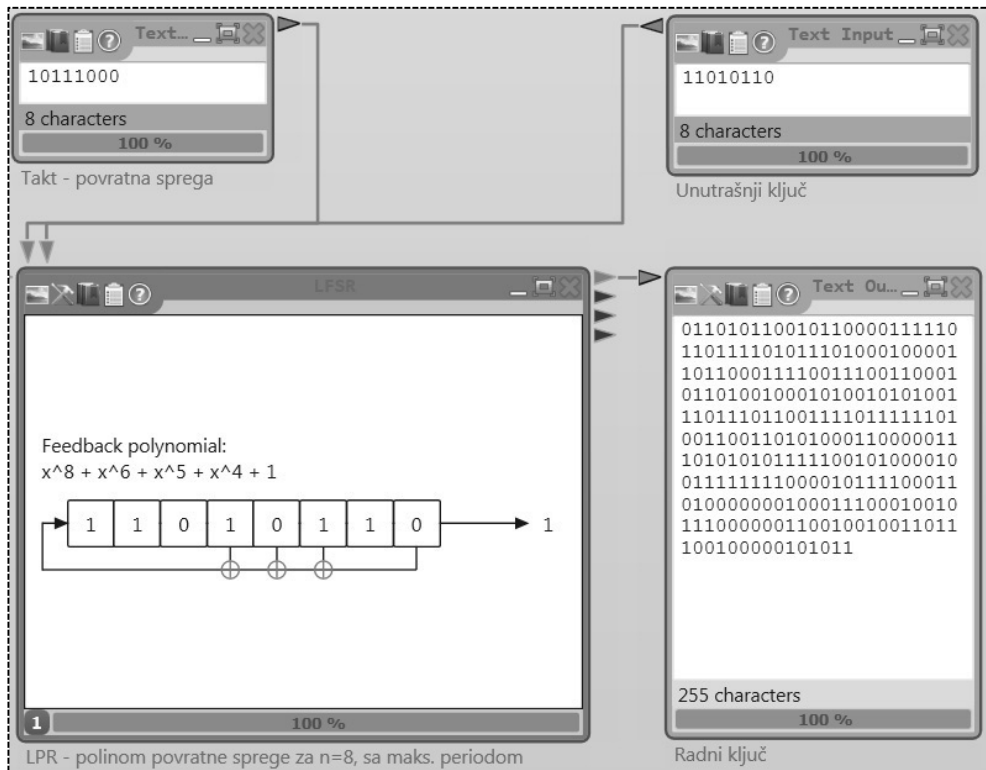
Sledeća tabela sadrži povratnu spregu definisanu polinomima i njihovu binarnu reprezentaciju za vrednost $n = 11$ bitova.

Tabela 5 Polinom povratne sprege maksimalne periode za $n=11$.

Bitovi n	Polinom povratne sprege (takt)	Binaran reprezentacija polinoma povratne sprege (takt)	Period $n - 1$
3	$x^3 + x^2 + 1$	110	7
4	$x^4 + x^3 + 1$	1100	15
5	$x^5 + x^3 + 1$	10100	31
6	$x^6 + x^5 + 1$	110000	63
7	$x^7 + x^6 + 1$	1100000	127
8	$x^8 + x^6 + x^5 + x^4 + 1$	10111000	255
9	$x^9 + x^5 + 1$	100010000	511
10	$x^{10} + x^7 + 1$	1001000000	1023
11	$x^{11} + x^9 + 1$	10100000000	2047

20. PRIMER:

Konstruisati jedan linearni pomerački registar (LPR) za $n = 8$ bitova, sa maksimalnom periodom. Povratna sprega za najveću periodu, određena je sa polinomom $x^8 + x^6 + x^5 + x^4 + 1$, tabela 5.



Slika 3.6 – rešenje problema u Cryptool edukativnom softveru

3.1.1. A5/1

Prva sekvencijalna šifra koja će se razmatrati je A5/1. Šifra se koristi za kriptografsku zaštitu podataka kod GSM zaštitnog protokola. Korišćen je za obezbeđivanje sigurne komunikacije između telefona i bazne stanice. Algoritam je držan u tajnosti, ali je spletom okolnosti dospao u javnost. Kasnije se pokazalo da dizajn algoritma ima

ozbiljnih slabosti. Ovaj algoritam ima algebarski opis, ali takođe može biti prikazan kroz relativno jednostavan dijagram.

A5/1 koristi tri linearna pomeračka registra sa povratnom spregom (LPR), koje ćemo mi nazvati X , Y i Z . Registar X se sastoji od 19 bitova, koje ćemo označiti sa $X = x_0 + x_1 \dots x_{18}$. Registar Y se sastoji od 22 bita $Y = y_0 + y_1 \dots y_{21}$ i registar Z se sastoji od 23 bita $Z = z_0 + z_1 \dots z_{22}$. Primetićemo da su uglavnom kriptografske vrednosti vezane za stepen broja 2, tako da nije slučajno da sva tri registra zajedno imaju 64 bita.

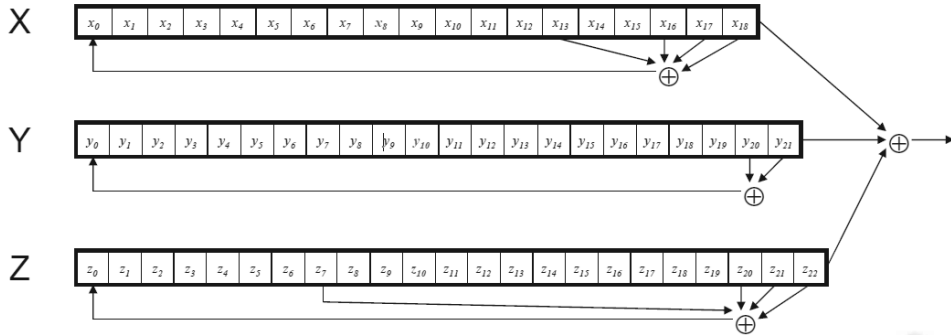
Ključ K je takođe dužine 64 bita. Ključ se koristi kao inicijalna vrednost koja se smešta u ova tri registra. Odnosno, ključ se koristi za postavljanje početnog stanja u sva tri registra. Ključ K zbog svoje uloge u praksi naziva se unutrašnji ključ. Kada je ključ postavljen u registre, linearni registri kreću u produkciju pseudoslučajnih nizova. Generisani pseudoslučajni nizovi u praksi se nazivaju radni ključevi.

Imajući u vidu da se radni ključ generiše na osnovu K upotrebom generatora radnog ključa, generator radnog ključa mora zadovoljiti stroge kriptografske zahteve. Zasnovanost generatora na matematičkom aparatu zahteva detaljnu statističku analizu da bi se potvrdilo da su osobine dobijenog radnog ključa što sličnije osobinama nekog slučajnog niza K .

Projektovani pseudoslučajni generator - PRNG (eng. *pseudo random number generator*), na osnovu slučajnog početnog stanja K dužine 64 bita, generiše mnogo duži niz pseudoslučajnih bitova. Očekivana količina slučajnih bitova je približno $\approx 2^{19}$ za dužinu unutrašnjeg ključa K od 64 bita ili inicijalnog stanja pseudoslučajnog generatora. Od dizajna linearnog pomeračkog registra sa povratnom spregom zavisice dužina radnog ključa ili veličina njegove periode.

Pre nego što opišemo kako se radni ključ generiše, reći ćemo nešto više o ovim linearnim registrima X , Y i Z . Broj bita ili perioda radnog ključa koji se generiše na osnovu 64-bitnog ključa je veoma veliki. Kriptografski sistemi koji koriste pomeračke registre se najčešće realizuju hardverski. Softverska realizacija je moguća, ali često je manje efikasna. Savremeni procesori omogućavaju prihvatljivo rešenje u nekim slučajevima. Pomerački registri su i danas osnova za realizaciju šifarskih

sistema, a sa njima se postiže da generisana pseudoslučajna sekvenca ima veliku periodu.



Slika 3.7 Generisanje radnog ključa sa A5/1 algoritmom

U svakom koraku: $m = maj(x_8, y_{10}, z_{10})$, primer: $maj(0, 1, 0) = 0$, $maj(1, 1, 0) = 1$

Ako je $x_8 = m$ tada se računa t i pomera X :

$$t = x_{13} \oplus x_{16} \oplus x_{17} \oplus x_{18}$$

$$x_i = x_{i-1} \text{ za } i = 18, 17, \dots, 1 \text{ i } x_0 = t$$

Ako je $y_{10} = m$ tada se računa t i pomera Y :

$$t = y_{20} \oplus y_{21}$$

$$y_i = y_{i-1} \text{ za } i = 21, 20, \dots, 1 \text{ i } y_0 = t$$

Ako je $z_{10} = m$, tad se računa t i pomera Z :

$$t = z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$$

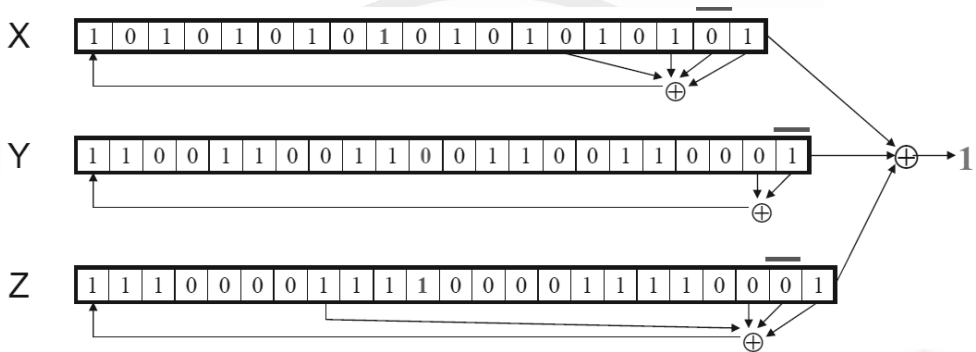
$$z_i = z_{i-1} \text{ za } i = 22, 21, \dots, 1 \text{ i } z_0 = t$$

Bit radnog ključa se računa kao: $x_{18} \oplus y_{21} \oplus z_{22}$.

Vidimo da je svaka vrednost (x_8, y_{10}, z_{10}) jedan bit radnog ključa. Unutrašnji ključ određuje početno stanje registra. Pomeranje ili šifrovanje registra zavisi od vrednosti (x_8, y_{10}, z_{10}) . Bit radnog ključa se u svakom trenutku dobija XOR-om poslednjih bitova registra $x_{18} \oplus y_{21} \oplus z_{22}$.

21. PRIMER:

Generisnaje jednog bit-a radnog ključa za $m = maj(x_8, y_{10}, z_{10}) = maj(1, 0, 1) = 1$



Slika 3.8 Jedan takt registra A5/1 algoritma

U ovom primeru registar X se pomera, registar Y se ne pomera i registar Z se pomera. Bit radnog ključa se računa kao: $s = 0 \oplus 1 \oplus 0 = 1$. Dobijeni bit radnog ključa se XOR-uje sa otvorenim tekstom (šifrovanje) ili XOR-uje sa šifratom (dešifrovanje). I tako ponavljamo ceo proces kako bi smo generisali onoliko bitova ključa koliko nam je potrebno za proces šifrovanja ili dešifrovanja.

Možemo da zapazimo da se stanje u registru menja tokom pomeranja bita. Posledica toga je da se posle generisanja jedne sekvence bita, stanje najmanje dva registra X, Y, Z promenilo, što rezultuje da su novi bitovi na pozicijama (x_8, y_{10}, z_{10}) . Zbog toga možemo ponoviti ovaj proces i generisati novi radni ključ.

Iako deluje komplikovano način za generisanje jednog bita radnog ključa, A5/1 se lako može implementirati u hardver i generisati bitove brzinom koja je proporcionalna brzini takta. Zapravo broj taktova određuje i broj bita radnog ključa.

Takođe, broj bita radnog ključa koji može biti generisan iz jednog unutrašnjeg ključa od 64 bita je praktično neograničen, ali ono što je sigurno jeste da će se u jednom momentu ponoviti.

A5/1 algoritam je predstavnik velike klase šifara koje su bazirane na pomeračkim registrima i imlementirane u hardver. Ovi sistemi su nekada bili kraljevi simetričnih kriptografskih sistema, ali poslednjih godina su im blokovske šifre preuzele tu titulu. Danas kad se sekvencijalne šifre koriste, koristi se RC4 algoritam o kome ćemo govoriti u sledećem poglavlju.

Zašto se tako masovno prešlo sa sekvencijalnih šifara na blokovske šifre? U prošlosti sekvencijalne šifre bazirane na pomeračkim registrima, bile su potrebne da bi se držao korak sa bitskim brzinama podataka u različitim komunikacionim kanalima. Kada se sa zaštite tekstualnih podataka prešlo na zaštitu kompleksnijih podataka (npr. kao što su audio fajlovi) problem zaštite podataka u realnom vremenu je postao još izraženiji. U telekomunikacijama su sve više počele da se koriste rešenja za prenos podataka na visokim bitskim brzinama. U prošlosti, softver baziran na kriptografskim algoritmima nije mogao da generiše bitove ključa dovoljno brzo za takve aplikacije. Danas postoji samo nekoliko aplikacija za koje kriptografski baziran softver nije odgovarajući. Kao dodatak, blokovske šifre je lako napraviti i one mogu da rade isto što i sekvencijalne, čak i mnogo više. Ovo su neki od glavnih razloga zašto su blokovske šifre sada dominantne.

3.1.2. RC4

RC4 je sekvencijalna šifra koja se u svojoj osnovi razlikuje od A5/1. Algoritam RC4 je napravljen za softversku implementaciju, dok je A5/1 dizajniran za hardver. RC4 generiše bajt radnog ključa u jednom taktu, dok A5/1 proizvodi u jednom taktu samo jedan bit radnog ključa. Sa aspekta vremenskih performansi, generisanje bajta u jednom taktu je mnogo bolje od generisanja jednog bita.

RC4 algoritam je veoma jednostavan zato što je njegova osnova tabela koja sadrži permutaciju svih mogućih vrednosti od 0-255 bajtova. Suština je u sledećem: svaki put kada se generiše bajt radnog ključa, tabela se modifikuje na takav način da uvek

sadrži permutaciju od $\{0,1,\dots,255\}$. Zbog ovog stalnog menjanja ili samo-modifikovanja, tabela i sama šifra za kriptanalitičara predstavlja pokretnu metu.

Kompletna dizajn RC4 algoritma radi nad bajtovima podataka. Prva faza algoritma inicijalizuje tabelu koristeći ključ. Označićemo ključ kao $ključ[i]$ za $i = 0,1,\dots,N-1$, gde je svaki $ključ[i]$ jedan bajt. Označićemo tabelu $S[i]$, gde je $S[i]$ takođe bajt. Pseudokod za inicijalizovanje permutacije S dat je u tabeli 1.

Jedna interesantna odlika RC4 algoritma jeste da ključ može biti bilo koje dužine od 1 do 256 bajtova. Najčešća dužina ključa je između 5 i 40 bajta. Ključ se koristi samo da bi se inicijalizovala permutacija S . Možemo da zapazimo da se 256 bajt-ni niz K dobija prosto ponavljajući ključ dok se niz ne napuni.

Nakon faze inicijalizovanja, svaki bajt radnog ključa se generiše na osnovu pseudokoda, **Tabela 6**. Izlaz koji smo označili kao jedan bajt radnog ključa je bajt nad kojim će se primeniti XOR sa bajtom otvorenog teksta ako je faza šifrovanja ili XOR sa bajtom šifrata ako je faza dešifrovanja.

Tabela 6 Faza šifrovanja

```

za  $i = 0$  do 255
     $S[i] = i$ 
     $K[i] = ključ[i \bmod N]$ 
sledeći  $i$ 
     $j = 0$ 
zar  $i = 0$  do 255
     $j = (j + S[i] + K[i]) \bmod 256$ 
    zamena_mesta( $S[i], S[j]$ )
sledeći  $i$ 
 $i = j = 0$ 

```

RC4 algoritam se može posmatrati kao samo-modifikujuća tabela. Vrlo je sofisticirano rešen, jednostavan i softverski efikasan. Međutim, postoje napadi koji su izvodljivi, ali samo pod nekim uslovima. Ukoliko je algoritam implementiran na adekvatan način, napad neće biti lako izvodljiv, a to znači da je prvih 256 bajtova radnog ključa odbačeno.

Za pravilno inicijalizovanje RC4 algoritma, neophodno je odbaciti prvih 256 bajtova radnog ključa. U protivnom, napadač bi mogao da rekonstruiše radni ključ na osnovu prvih 256 bajtova šifrata. Ovo može biti postignuto dodavanjem 256 koraka u fazu inicijalizovanja gde svaki dodatni korak generiše i odbacuje bajt radnog ključa, prateći algoritam, **Tabela 7**. Za pravilno funkcionisanje algoritma, Alisa i Bob zajedno implementiraju ove dodatne korake, da bi mogli koristiti RC4 šifru za sigurnu komunikaciju.

Tabela 7. Faza inicijalizovanja

$$i = (i + 1) \bmod 256$$

$$j = (j + S[i]) \bmod 256$$

$$\text{zamena_mesta}(S[i], S[j])$$

$$t = (S[i] + S[j]) \bmod 256$$

$$\text{radni_ključ} = S[t]$$

RC4 se danas koristi u mnogim aplikacijama, uključujući zaštitne protokole SSL i WEP. Google danas koristi RC4 algoritam u TLS zaštitnom protokolu za obezbeđivanje sigurne komunikacije između klijenta i servera u servisu za elektronsku poštu (gmail.com). U današnje vreme, u pružanju ovakvih usluga je neophodno koristiti kriptografsku zaštitu. Ne zaboravimo da je i RC4 šifra pseudo-One-time pad šifra i da su njene visoke performanse stvorile konkurentsku prednost nad ostalim modernim šiframa. Zbog ogromnog broja Internet korisnika i zahtevane brzine odziva, Google je izabrao RC4 algoritam zbog svih svojih dobrih karakteristika.

Međutim, algoritam je veoma star i nije optimizovan za 32-bitnu arhitekturu procesora. Možemo zaključiti na osnovu izgleda njegovog dizajna da je prilagođen starim 8-bitnim procesorima. Kako je ugrađen u velikom broju savremenih računarskih protokola i aplikacija, još uvek ima značaj sa kriptografskog aspekta.

Sekvencijalne šifre su nekada bile primarne, ali su sada relativno retke u poređenju sa blokovskim šiframa. Neki su otišli toliko daleko da su proglasili i njihovo iščeznuće i kao dokaz naveli su da u poslednjih par godina nije bilo napora da se razvije ni jedna nova sekvencijalna šifra. Ipak, danas se javlja povećan broj značajnih aplikacija za koje su sekvencijalne šifre mnogo pogodnije nego blokovske. Primer za takve aplikacije su bežični uređaji (bežične mreže), uređaji sa ograničenom procesorskom snagom i

uređaji sa velikim protokom podataka (zaštita videa i govora). Nesumnjivo, priče o iščeznuću blokovskih šifri su bile značajno preuveličane.

Slika 3.9, prikazana je jedna jednostavna implementacija RC4 algoritma u Java programskom jeziku. Korišćen je kriptografski paket klasa "javax.crypto.*" Ključ ili lozinka koju algoritam koristi za inicijalizovanje je niz karaktera "SINGIDUNUM", a otvoreni tekst ili poruka "UNIVERZITET_SINGIDUNUM". Prikazane su obe faze, faza šifrovanja i dešifrovanja.

```
package rc4;

import java.security.Key;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class RC4 {
    //----- RC4 -----
    public static byte[] rc4(byte[] data, byte[] bkey) throws Exception {
        Key key = new SecretKeySpec(bkey, "ARCFOUR");
        Cipher c = Cipher.getInstance("ARCFOUR");
        c.init(Cipher.ENCRYPT_MODE, key);
        byte[] encVal = c.doFinal(data);
        return encVal;
    }

    public static void main(String[] args) throws Exception {
        byte[] key = "SINGIDUNUM".getBytes();
        byte[] text = "UNIVERZITET_SINGIDUNUM".getBytes();
        byte[] encrypted = rc4(text, key);
        byte[] decrypted = rc4(encrypted, key);

        System.out.println("Šifrat: " + new String(encrypted));
        System.out.println("Otvoreni tekst: " + new String(decrypted));
    }
}
```

Šifrat: 0J0 ǂ<0>0000 rZ0
Otvoreni tekst: UNIVERZITET_SINGIDUNUM

Slika 3.9 Implementacija RC4 algoritma u Javi

3.2. BLOKOVSKJE ŠIFRE

Ponavljajuća blokovska šifra deli otvoreni tekst na blokove fiksne veličine i generiše blokove šifrata fiksne veličine. U mnogim slučajevima, šifrat se dobija iz otvorenog teksta ponavljanjem funkcije F za određeni broj rundi. Funkcija F , koja zavisi od izlaza prethodne runde i ključa K , je poznata kao funkcija runde ne zbog svog oblika, nego zbog toga što je primenjena preko više rundi. Ulazni parametri za svaku rundu su ključ i izlaz iz prethodne runde.

Blokovske šifre kombinuju zajedno svojstvo difuzije, svojstvo konfuzije i kompletnosti. Blok otvorenog teksta odredićemo sa P_i , a blok šifrata C_i . Difuzija podrazumeva da poznavanje para P_i i C_i ne sme da omogući da se na osnovu C_j odredi odgovarajuće P_j . Kod blokovskih šifara zahtevano je da male promene u bloku otvorenog teksta (1 bit) treba da izazovu nepredvidive promene u datom bloku šifrata. Svojstvo konfuzije podrazumeva da će svi ključevi biti jednako verovatni u slučaju napada potpunom pretragom ključeva. Na kraju, celovitost podrazumeva da će svaki bit šifrata postati funkcija svakog bita ključa.

Ciljevi dizajna blokovskih šifri su sigurnost i efikasnost. Nije toliko teško razviti razumno sigurnu blokovsku šifru ili efikasnu blokovsku šifru, ali dizajnirati nešto što je i sigurno i efikasno, zahteva visok stepen kriptografskog umeća. Blokovske šifre se najčešće realizuju softverski.

3.2.1. FEJSTEL ŠIFRA

Feistel šifra je nazvana po Nemačkom kriptografu Horstu Feistelu (*Horst Feistel*), koji je bio pionir u razvoju dizajna blokovskih šifara, radio je u IBM-u. Ovo su bila inicijalna istraživanja koja su kulminirala razvojem DES (*Data Encryption Standard*) algoritma 1970. godine. Feistel šifra predstavlja dizajn blokovske šifre, a ne posebnu šifru.

Principi rada Feistel šifre. Kod Feistel šifre, otvoreni tekst se deli na levu i desnu polovinu:

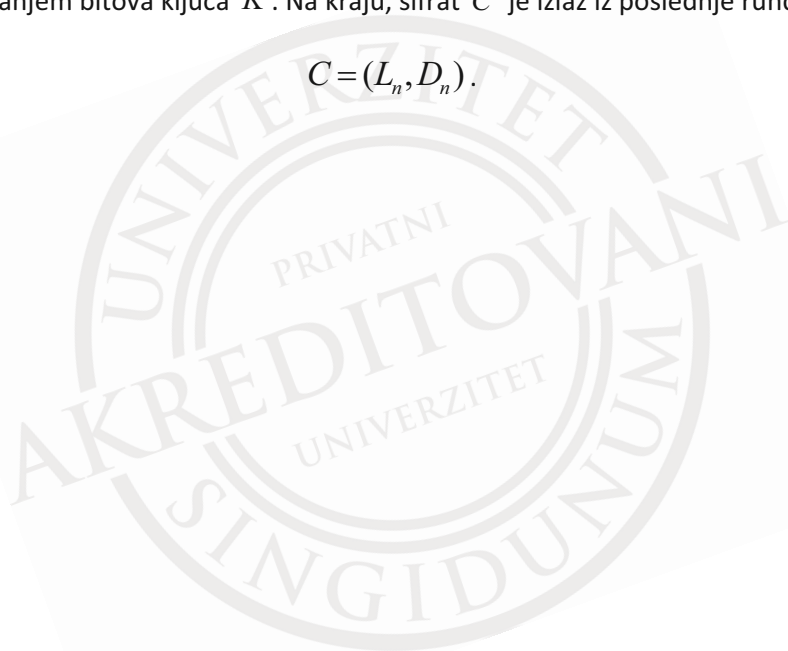
$$P = (L_{OT}, D_{OT}),$$

i za svaku novu rundu $i = \{1, 2, \dots, n\}$, generiše se nova leva i desna polovina, određena na osnovu pravila:

$$L_i = D_{i-1}, D_i = L_{i-1} \oplus F(D_{i-1}, K_i),$$

gde je F funkcija runde, a K_i je podključ za rundu i . Podključ se dobija kombinovanjem bitova ključa K . Na kraju, šifrat C je izlaz iz poslednje runde:

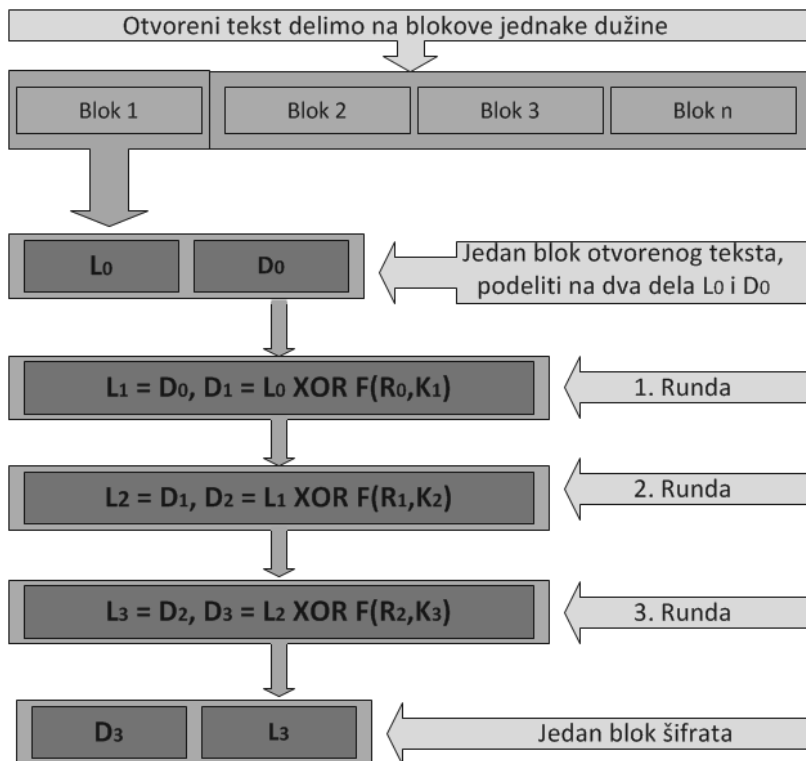
$$C = (L_n, D_n).$$



22. PRIMER:

Fejstel, primer šifrovanja u tri runde.

U prvom koraku delimo neki otvoreni tekst u blokove fiksne dužine od 64 bita. Zatim uzimamo jedan blok otvorenog teksta i delimo ga na dva dela $P_0 = (L_0, D_0)$. Nakon toga, računamo blok šifrata za tri runde **Slika 3.10**.



Slika 3.10 Fejstel šifra, primer šifrovanja u 3 runde

Na osnovu funkcije F za generisanje nove runde i podključa K_i , primetićemo da je nova leva polovina L_i stara desna polovina D_i , a nova desna polovina D_i je stara leva polovina L_i .

Svakako, potrebno je sada dešifrovati dobijeni blok šifrata. Lepota Fejstel šifre jeste da mi možemo dešifrovati šifrat, bez obzira na pojedinačnu funkciju runde F_i . Da bi to mogli uraditi, rešava se jednačina:

$$L_i = D_{i-1}, D_i = L_{i-1} \oplus F(D_{i-1}, K_i)$$

za $D_i - 1$ i $L_i - 1$, što nam dozvoljava da pokrenemo proces unazad. Za $i = \{1, 2, \dots, n\}$, pravilo za dešifrovanje je:

$$D_{i-1} = L_i, L_{i-1} = D_i \oplus F(D_{i-1}, K_i),$$

gde je F funkcija runde, a K_i je podključ. Krajnji rezultat je originalni otvoreni tekst:

$$P = (L_{OT}, D_{OT}).$$

Svaka funkcija runde F će raditi u Fejstel šifri ako izlaz funkcije F obezbeđuje ispravan broj bitova. Posebno je dobro što ne postoji potreba da funkcija F bude inverzna. Međutim, Fejstel šifra neće biti kriptografski sigurna za neke funkcije F . Primetićemo da se sva pitanja vezana za sigurnost Fejstel šifre svode na pitanja o funkciji runde F i rasporedu podključa K_i . Raspored ključa obično nije bitna stvar tako da se najveći fokus daje analizi funkcije F .

3.2.2. DES

Data Encryption Standard, poznat kao DES je razvijen u "mračnom dobu kompjuterske ere" 1970-ih godina. Dizajn je baziran na takozvanoj Lucifer šifri, Fejstel šifri razvijenoj od strane tima u IBM-u. DES je iznenađujuće jednostavna blokovska šifra, ali priča o tome kako je od Lucifer šifre nastao DES je sve, samo ne i jednostavna.

Do sredine 1970-ih bilo je jasno čak i birokratama u američkoj vladi da postoji legitimna komercijalna potreba za sigurnom kriptografijom. U to vreme,

kompjuterska revolucija je bila u toku, a potreba za poverljivošću digitalnih podataka je ubrzano rasla.

Sredinom 1970-ih, kriptografija je bila veoma slabo poznata izvan vojnih i vladinih krugova, a oni nisu o tome mnogo govorili i to je još uvek tako. Prve kriptografske konferencije pojavile su se tek posle 1980. godine. Posledica ove izolovanosti je bila da preduzeća nisu mogla da procene sigurnost koju nudi neki kriptografski proizvod. Sigurnost koju je nudila većina takvih proizvoda, bila je veoma loša.

U ovom okruženju Nacionalni biro za standarde (eng. *National Bureau of Standards*) ili NBS, a sada poznat kao NIST, izdao je zahtev za predlog novih šifara. Pobjednik bi postao standard američke vlade i skoro sigurno *de facto* industrijski standard. Veoma malo razumnih rešenja je podneto i veoma brzo je postalo jasno da je IBM-ova Lucifer šifra jedini ozbiljan kandidat na tom konkursu.

U tom trenutku, NBS je imao problem. Bilo je veoma malo kriptografske stručnosti u NBS-u, tako da su se okrenuli vladinim stručnjacima, super-tajnoj Nacionalnoj sigurnosnoj agenciji (eng. *National Security Agency*) ili NSA. NSA dizajnira i pravi kriptografske mehanizme koji se koriste u američkoj vojsci i vladi za zaštitu visoko poverljivih informacija.

NSA se nerado umešala u priču sa DES-om, ali pod pritiskom su se na kraju složili da prouče dizajn Lucifer šifre i iznesu neka mišljenja, uz uslov da uloga NSA ostane u tajnosti. Kada je ova informacija objavljena, (što je svakako bilo neizbežno u Americi) mnogi su bili sumnjičavi da je NSA ostavila "zadnja vrata" (eng. *backdoor*) u DES-u, tako da samo ona može razbiti šifru. Sigurno, SIGNIT misija NSA i generalna klima nepoverenja prema vladi, podsticala je takve strahove. U odbranu NSA, vredi primetiti da se za 30 godina intenzivne kryptoanalize nije otkrio nijedan "*backdoor*" ili propust u dizajnu DES-a. Ipak, ovo je zadržalo sumnju na DES od samog početka.

Lucifer je najzad postao DES, ali ne pre nego što je učinjeno nekoliko ne baš suptilnih promena. Najočiglednija promena je smanjenje dužine ključa sa 128 bitova na 64 bita. Međutim, nakon rigorozne informacione analize, otkriveno je da se 8 od 64 bita ključa odbacuje, pa je stvarna dužina ključa samo 56 bitova. Kao rezultat ovih promena, očekivano vreme za napad potpunom pretragom ključeva je smanjeno sa 2^{127} na 2^{55} . Prema tome, DES je za 2^{72} lakši za razbijanje od Lucifera.

Razumljivo, sumnja se da je NSA imala ulogu u namernom slabljenju DES-a. Međutim, naknadne kriptanalize DES algoritma su otkrile napade koji zahtevaju neznatno manje vremena od isprobavanja 2^{55} ključeva. Kao rezultat, DES je verovatno jak sa ključem od 56 bitova kao što bi bio i sa dužim Lucifer ključem.

Suptilne promene Lucifera uključile su i kutije zamene ili S-kutije koje su objašnjene u donjem delu teksta. Posebno su ove promene podsticale sumnju o ostavljenim "zadnjim vratima". Međutim, vremenom je postalo jasno da su promene S-kutija ustvari ojačala algoritam nudeći zaštitu protiv kriptanalitičkih metoda koje nisu bile poznate (barem ne izvan NSA i o tome se nije govorilo) sve do nekoliko godina unazad. Nepobitna činjenica je da, ko god da je modifikovao Lucifer algoritam (a to je NSA), znao je šta radi i značajno je ojačao algoritam.

Cilj dizajnera je bio da uvede princip konfuzije, što znači da šifrat treba da zavisi od otvorenog teksta i ključa na složen način i princip difuzije, što znači da svaki bit šifrata treba da je funkcija svih bitova otvorenog teksta i svih bitova ključa. Pored ova dva principa zahtevan je i "lavinski efekat", a to bi značilo da male promene ulaza treba da izazovu velike promene izlaza. Kod DES-a promena jednog bita ključa ili jednog bita od ukupno 64 bita otvorenog teksta menja 50% bita bloka šifrata.

U nastavku ćemo navesti osnovnu numerologiju ili specifikaciju DES algoritma.

DES je Fejstel šifra sa sledećim funkcionalnim parametrima:

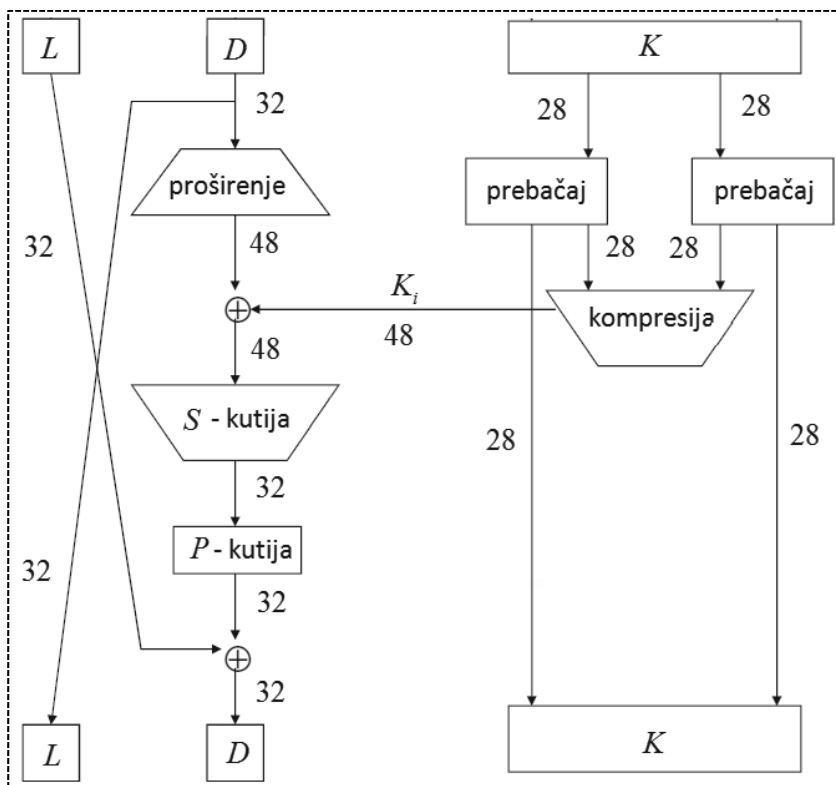
- Dužina bloka otvorenog teksta je 64 bita;
- Dužine ključa je 56 bitova sa ukupno 2^{56} mogućih ključeva;
- Ima ukupno 16 rundi pre generisanja jednog bloka šifrata;
- Dužina podključa koji se koristi u svakoj rundi je 48 bitova;

Svaka runda je jednostavna prema standardu o kreiranju blokovskih šifri.

Slika 3.11, potvrđuje da je DES Fejstel šifra kao što je definisano u poglavlju Fejstel šifra. Funkcija F runde je kompozicija proširene permutacije, zbira podključa, S-kutije i P-kutije, kao što je dato u navedenom poglavlju Fejstel šifre.

Pošto je DES Fejstel šifra, šifrovanje prati pravila prikazana u jednačini:

$$F(D_{i-1}, K_i) = P - kutija(S - kutija(proširenje(D_{i-1} \oplus K_i)))$$



Slika 3.11 Kompletna šema DES algoritma

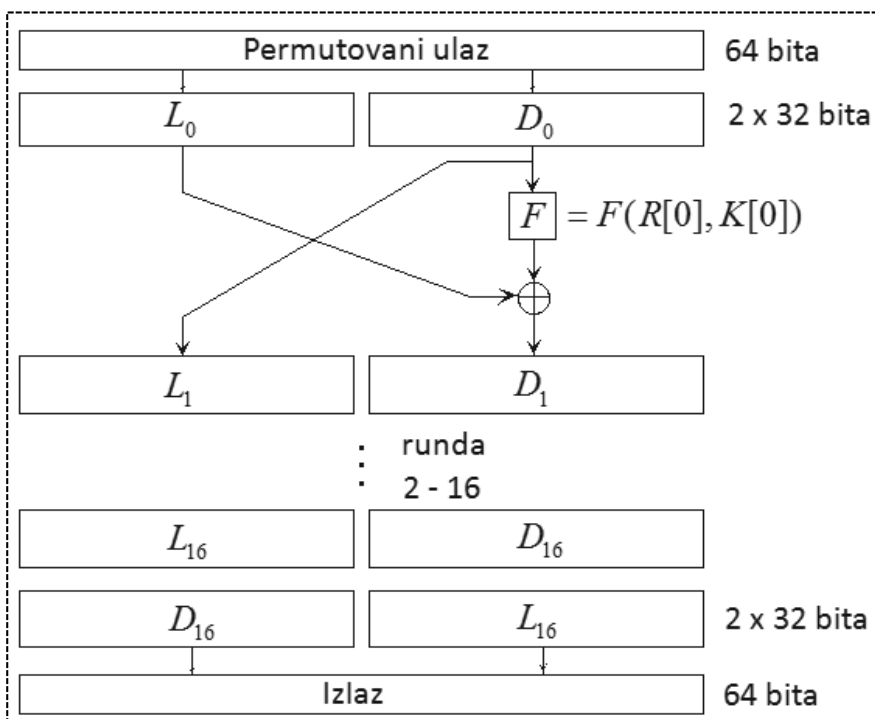
Šifrovanje:

- Svaki blok se transformiše u 16 rundi zamena i permutacija;
- Permutacije unose difuziju u podatke, zamene konfuziju (Šenonovi principi);
- U svakoj rundi se koristi 48 bitova podključa;
- Inicijalna i inverzna permutacija ne doprinose jačini šifre.

Dešifrovanje:

- Proces je identičan procesu šifrovanja, samo se izvršava sa podključem u obrnutom redosledu.

Na **Slika 3.12.** prikazana je šema izvršenja 16 rundi DES algoritma. Kao što je prikazano u šemi, izlaz iz svake runde XOR-uje se sa odgovarajućim podključem.

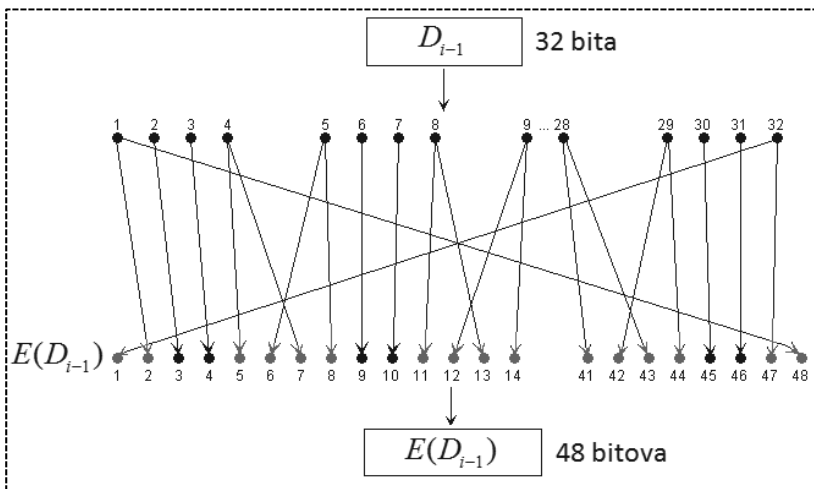


Slika 3.12 DES algoritam 16 rundi

E-kutija (ekspanziona permutacija)

E-kutija predstavlja ekspanzionu permutaciju koja proširuje svoj ulaz od 32 bita na 48 bita. Funkcija menja redosled bita, a neke ponavlja zbog potrebe za lavinskim efektom. **Slika 3.13**, ilustruje način na koji je to urađeno. Tačno mapiranje određuje se ekspanzionom tabelom E, koja nije prikazana ovde. $E(D_{i-1})$

Funkcija $F(D_{i-1}, L_i)$, svaki blok od 32 bita D_{i-1} i L_{i-1} proširuje pre XOR-a sa odgovarajućim 48-bitnim podključem K_i . To znači da će se neke vrednosti iz D_{i-1} morati udvostručiti.

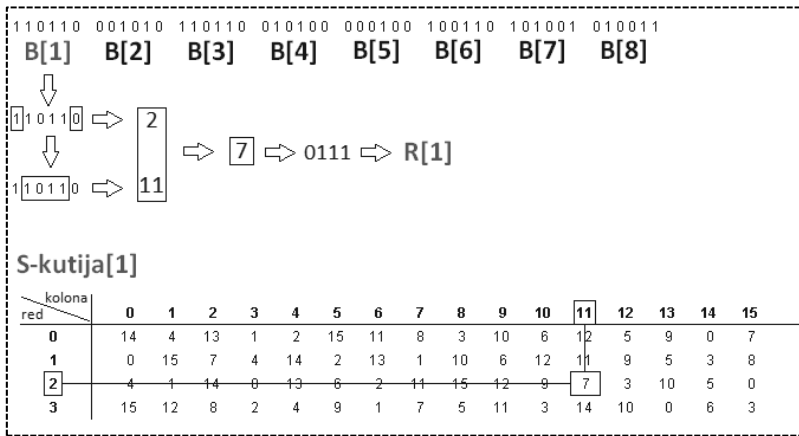


Slika 3.13 Funkcija proširenja – ekspanziona permutacija

S-kutije (funkcija supstitucije i kompresije)

S-kutije u DES-u su jedna od najvažnijih sigurnosnih funkcija u kompletnom algoritmu. Za razliku od drugih linearnih kutija, S-kutije su nelinearne i teške za kriptanalizu. Videćemo da su S-kutije zajednička funkcija većini modernih blokovskih šifri. U DES-u, svaka S-kutija preslikava 6 bitova u 4 bita. DES koristi osam S-kutija koje zajedno preslikavaju 48 bitova u 32 bita. Zbog toga se svaka može posmatrati kao niz od 4 reda i 16 vrsta, sa jednom 4-bitnom vrednosti koja je smeštena u bilo kojoj od 64 pozicije. Iste S-kutije se koriste za svaku rundu u DES-u.

Slika 3.14 prikazuje rad prve S-kutije. Ulaz u funkciju F koja predstavlja funkciju S-kutija je veličine 48 bitova. Zatim, ulaz od 48 bitova je podeljen na 8 blokova jednakih dužina. Svaki blok je označen sa indeksnim vrednostima $B[i]$, iz skupa vrednosti $i = \{1, 2, \dots, 8\}$. Dužina svakog $B[i]$ bloka je 6 bitova. Indeksne vrednosti i , govore nam koja je od S-kutija primenjena na dati blok $B[i]$. S-kutije (supstitucione zamene) su set od 8 matrica dimenzije $[4 \times 16]$ sa konstantnim vrednostima.



Slika 3.14 Jedna S-kutija DES-a

U sledećem koraku je potrebno odrediti osam S_n vrednosti za $n = \{1, 2, \dots, 8\}$, na osnovu pojedinačne $B[i]$ vrednosti za odgovarajuću S-kutiju. Za svaki S_n izdvaja se prvi i poslednji bit. Izdvojeni bitovi konvertuju se u decimalnu vrednost koja predstavlja indeks reda S-kutije. Unutrašnja četiri bita koja su preostala konvertuju se takođe u decimalnu vrednost koja predstavlja indeks kolone. Sada kada imamo vrednosti indeksa kolone i reda, uzimamo decimalnu vrednost iz prve S-kutije. Uzeta vrednost se deli sa brojem 2, a dobijeni ostatak predstavlja novi $B[i]$ blok dužine od 4 bita, gde zapravo konvertujemo decimalnu vrednost u binarnu.

23. PRIMER:

Slika 3.14, primer za rad prve S-kutije:

$$B[1] = \begin{matrix} r & & r \\ 1 & 1 & 0 & 1 & 1 & 0 \\ k & k & k & k & k & k \end{matrix}$$

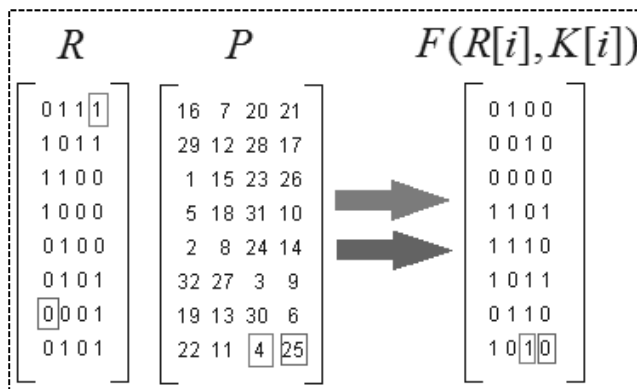
$$10 = 1 * 2^1 + 0 * 2^0 = 2;$$

$$1011 = 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 11;$$

$$S-kutija[red][kolona] = [2][11] = 7;$$

$$R_1 = 0111$$

Sve preostale $B[i]$ vrednosti su zamenjene na isti način. Vrednost R_i je rezultat i transformacija funkcije F prve S-kutije. Krajnji rezultat je concatenacija bitova svih R_i vrednosti za $i = \{1, 2, \dots, 8\}$. Sada sledi još jedna transformacija od R , a to je permutacija na osnovu matrice zamena P .



Slika 3.15 Tabela zamena (permutacija 32 bita)

Slika 3.15, prikazana je permutaciona matrica P i matrica R koja predstavlja izlaz S-kutija. Matrica P sadrži konstantne vrednosti, koje predstavljaju nove pozicije bitova iz matrice R .

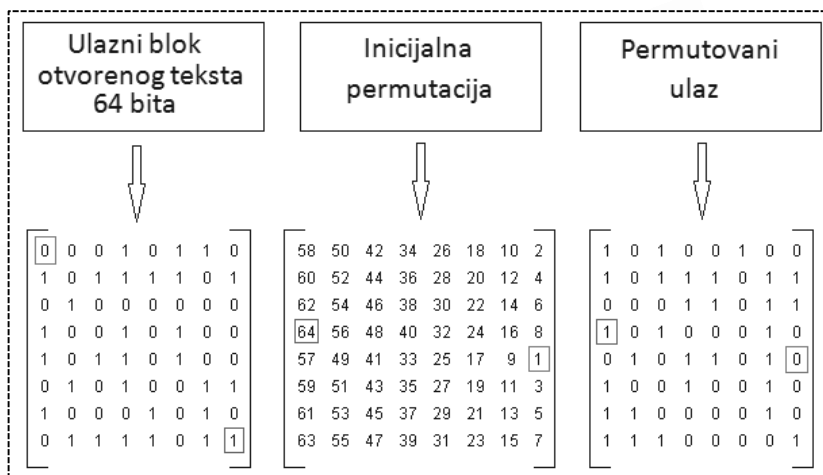
Iz ovog primera vidimo da funkcija F ima sopstvenu tabelu zamena. Na kraju, funkcija F (funkcija S-kutije) na izlazu daje $F(D_{i-1}, K_i)$ veličine 32 bita koja se XOR-uju sa levom polovinom L_i .

S-kutije su originalno bile zaštićene od objavljivanja iz čega proizilazi da je nesumnjiv cilj bio da one ostanu tajna. Međutim, kao što se i moglo predvideti, S-kutije u DES-u bile su reversno analizirane i postale su javno poznate skoro odmah.

P-kutije (Inicijalna permutacija)

DES P-kutija predstavlja niz permutacija nad 64-bitnim blokom otvorenog teksta. P-kutija doprinosi maloj sigurnosti šifre i njena prava svrha se izgubila u magli istorije. Jedno od mogućih objašnjenja jeste da su dizajneri želeli da učine DES mnogo težim

za softversku implementaciju budući da je originalni predlog predviđao hardversku implementaciju. To je očigledno bila nada da će DES ostati realizovan samo na hardverskim platformama ili možda zbog verovanja da će hardverska realizacija omogućiti da algoritam ostane u tajnosti. Permutacija P-kutije je jedini značajan deo DES-a koji nam je ostao na strani otvorenog teksta, a kasnije i algoritam rasporeda ključa koji se koristi za generisanje podključa.

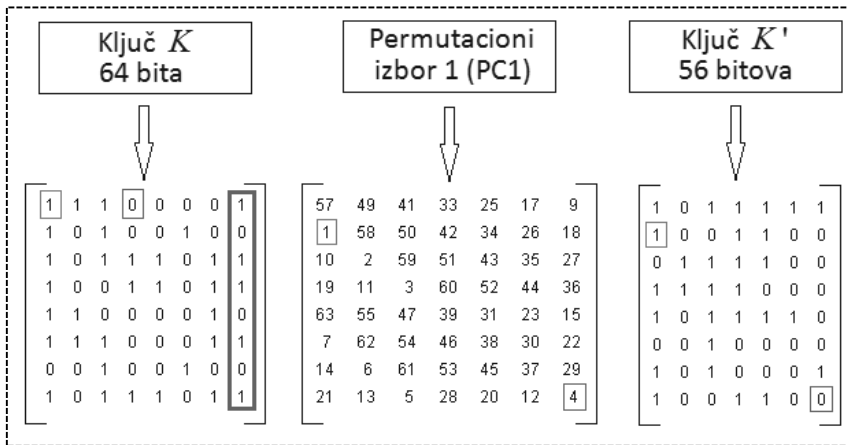


Slika 3.16 P-kutija

Slika 3.16. prikazan je jedan blok otvorenog teksta veličine 64 bita i permutaciona matrica iste veličine. Ulazni bitovi otvorenog teksta se preuređuju prema inicijalnoj permutaciji. Zapravo, brojevi u permutacionoj matrici određuju položaj ulaznih bita, tako da svaki bit bude na svom mestu. Krajnji rezultat P-kutije je permutovani ulaz.

K-kutija (kutija za generisanje podključa)

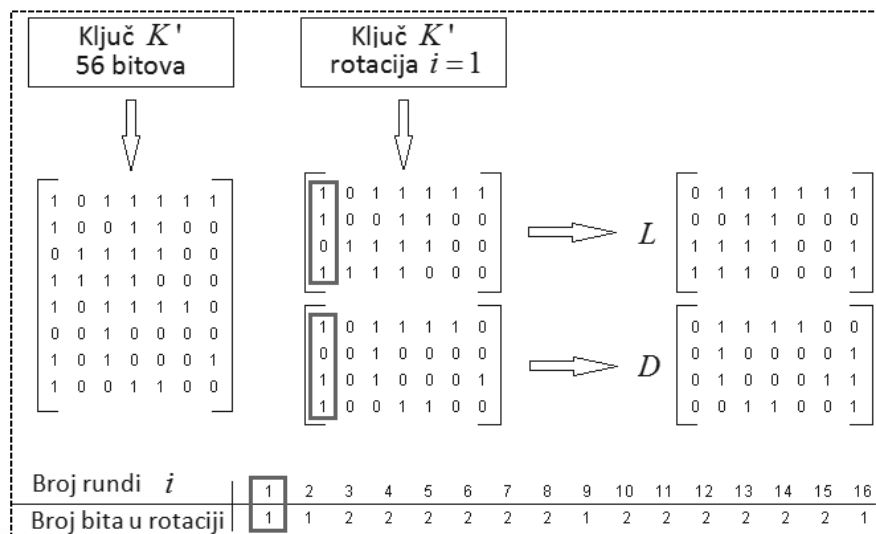
DES K-kutije donekle predstavljaju kompleksan proces, ali konačni rezultat je generisan podključ K_i , dužine 48 bitova, na osnovu ključa K' , dužine 56 bitova. Za svaku rundu i od ukupno 16 rundi, bira se novi K_i za svako $i = \{1, \dots, 16\}$. Detaljno ćemo objasniti šemu generisanja podključeva jer su detalji najbitniji za sigurnost blokovskih algoritama. Uglavnom su uspešni napadi na blokove šifre bili plod grešaka u algoritmima za generisanje podključeva.



Slika 3.17 K-kutija (prva faza)

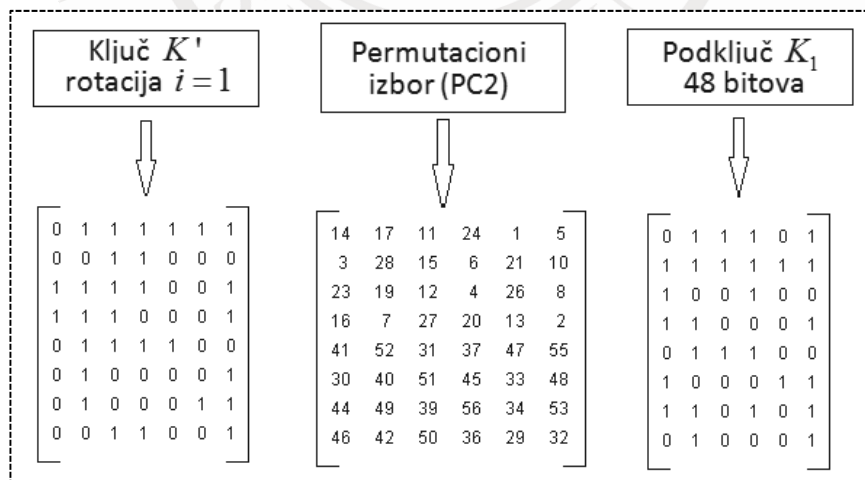
Osnovna uloga K-kutije je generisanje podključa K_i dužine 48 bitova. Slika 3.17. prikazana je prva faza K-kutije, od ukupno tri faze. U prvoj fazi, pre prve izabrane permutacije, odbacićemo krajnje desne bitove ključa K (uokvireno crvenom bojom). Ključ K je sada skraćen na 56 bitova. U sledećem koraku, dobijeni ključ K' dužine 56 bitova, postavimo u permutacionu matricu koju nazivamo permutacioni izbor 1 (PC1). Bitovi ključa K se preuređuju onako kako je zadato u matrici inicijalne permutacije (PC1). Dobijeni rezultat je novi ključ K' , koji predstavlja skraćeni i permutovani ključ K .

U drugoj fazi K-kutije, Slika 3.18. predstavljen je proces generisanja 16 podključeva K_i za svako $i = \{1, \dots, 16\}$, čija je dužina 48 bitova. To je proces koji od K' generiše podključeve K_i . Ključ K' je podeljen na dve polovine, na levu L i desnu D polovinu. Tabela koja je navedena na slici definiše broj bitova koji će se rotirati za svaki ključ K_i . Za prvi podključ K_1 (uokvireno), u matrici K' se rotiraju samo prvi bitovi, kao i u drugoj polovini. Dobijeno novo K' za određeno $i = \{1, \dots, 16\}$, predstavlja ulaz za poslednju fazu generisanja prvog podključa K_1 .



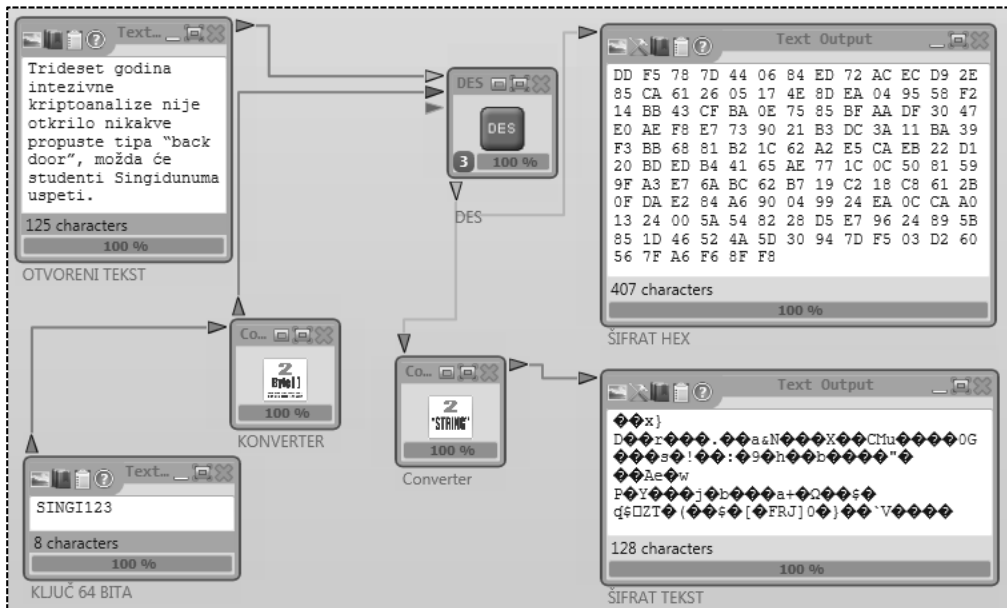
Slika 3.18 K-kutija (druga faza)

Slika 3.19, prikazan je poslednji korak ili treća faza za generisanje prvog podključa. U ovoj fazi koristi se izabrana permutacija (PC2). Matrica permutacije (PC2) sadrži 48 pozicija. Za date pozicije izdvojićemo bitove iz ključa K' (određen u drugoj fazi). Krajni rezultat je prvi podključ K_1 . Druga i treća faza se ponavljaju za sve vrednosti i definisane u tabeli Slika 3.18, a zavise od reda runde.



Slika 3.19 K-kutija (treća faza, generisanje prvog podključa)

Slika 3.20, prikazana je jedna jednostavna realizacija DES algoritma u Cryptool-u za šifrovanja otvorenog teksta.



Slika 3.20 Primer sa DES algoritmom u Cryptool-u

Sigurnost DES algoritma

Nekoliko reči o sigurnosti DES-a koje mogu biti korisne. Prvo, matematičari su veoma dobri u rešavanju linearnih jednačina, a jedini deo kod DES-a koji nije linearan su S-kutije. Zbog matematičkih zakonitosti, linearne šifre su inherentno slabe pa su S-kutije presudne za sigurnost DES-a. Zapravo, proširena permutacija ima važnu bezbednosnu ulogu, a u nešto manjoj meri i raspored ključeva.

Uprkos zabrinutosti u vezi dizajna DES-a, posebno uloge NSA-a u celom procesu DES je očigledno izdržao test vremena. Trideset godina intenzivne kriptanalize nije otkrilo nikakve propuste tipa "back door". Danas, DES je ranjiv samo zbog male dužine ključa, a ne zato što postoji neki ozbiljniji skraćeni napad. Iako su razvijeni neki napadi koji teorijski zahtevaju neznatno manje posla u odnosu na potpunu pretragu ključeva, svi dosad napravljeni programi za razbijanje DES-a jednostavno isprobavaju

sve kombinacije ključeva dok ne naiđu na onu pravu, odnosno, primenjuju potpunu pretragu ključeva.

Napad potpune pretrage ključeva podrazumevao bi da napadač mora da isproba 2^{56} ključeva, ali danas je napad potpunom pretragom ključeva izvodljiv. Godine 1993. Majkl Viner (*Michael Wiener*) je pokazao da je moguće napraviti hardver kojim se može razbiti DES napadom tipa poznati otvoreni tekst:

- za 35 sati sa budžetom od 100.000 dolara;
- za 3.5 sata sa budžetom od 1 milion dolara;
- za 21 minut sa budžetom od 10 miliona dolara;

2006. je napravljena nova mašina COPACABANA od strane tima sa Univerziteta Bohumu i Kil u Nemačkoj. Za razliku od drugih mašina ova mašina je komercijalno dostupna. U 2008. godini razvijena je nova verzija, COPACABANA RIVYERA koja je smanjila vreme potrebno za razbijanje DES-a na manje od jednog dana. Ona koristi 128 Spartan-3 5000 FPGA čipove. Trenutno *SciEngines* kompanija u Nemačkoj RIVYERA drži rekord u razbijanju DES-a metodom potpune pretrage ključeva. Jedan od zanimljivijih aspekata projekta COPACABANA je količina potrebnog novca za izgradnju takvog hardvera. Jedna mašina može se napraviti za 10.000 dolara.



Slika 3.21 COPACABANA RIVYERA 2008

Neke institucije su tada verovatno već posedovale takav hardver. Neizbežan zaključak jeste da su DES dizajneri znali šta rade.

Istorijski značaj DES-a je veliki. DES se može posmatrati kao pokretač za razvoj savremene kriptografije sa simetričnim ključevima, tako da postoji doza ironije budući da je NSA, doduše nevoljno, doprinela njegovom stvaranju. U nastavku, objasnićemo trostruki DES koji je korišćen za efikasno proširenje dužine ključa kod DES-a. Zatim će uslediti kratak pregled još nekoliko blokovskih šifara. Na kraju će se

dati prikaz današnjeg standarda za blokovske šifre pod nazivom AES (eng. *Advanced Encryption Standard*) koji se koristi u većini realizacija.

24. PRIMER:

Slika 3.22, Još jedna realizacija DES algoritma u Java programskom jeziku.

```
package DES;
import java.security.Key;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class DES {
    //----- DES metod koji se koristi za šifrovanje -----
    public static byte[] DES_sifruj(byte[] data, byte[] bkey) throws Exception {
        Key key = new SecretKeySpec(bkey, "DES");
        Cipher c = Cipher.getInstance("DES");
        c.init(Cipher.ENCRYPT_MODE, key);
        byte[] encVal = c.doFinal(data);
        return encVal;
    }
    //----- DES metod koji se koristi za dešifrovanje -----
    public static byte[] DES_desifruj(byte[] data, byte[] bkey) throws Exception {
        Key key = new SecretKeySpec(bkey, "DES");
        Cipher c = Cipher.getInstance("DES");
        c.init(Cipher.DECRYPT_MODE, key);
        byte[] encVal = c.doFinal(data);
        return encVal;
    }
    //----- Izvršni metod, koji sadrži proces šifrovanja i dešifrovanja
    public static void main(String[] args) throws Exception{
        byte[] kljuc = "SINGI123".getBytes(); //8 bajtova ili 64 bita
        byte[] otvoreni_tekst1;
        String otvoreni_tekst2 = "Trideset godina intezivne kriptanalize nije "
            + "otkrilo nikakve propuste tipa back door!"
            + " Možda će studenti Singidunuma uspeti.";
        otvoreni_tekst1 = otvoreni_tekst2.getBytes();
        byte[] sifrat = DES_sifruj(otvoreni_tekst1, kljuc);
        byte[] otovoreni_tekst = DES_desifruj(sifrat, kljuc);
        System.out.println(new String(sifrat));
        System.out.println(new String(otovoreni_tekst));
    }
}
```

Slika 3.22 Implementacija DES-a u Java programskom jeziku

3.2.3. TROSTRUKI DES

Pre nego što nastavimo da govorimo o drugim šiframa, neophodno je da se prikaže popularna varijanta DES-a poznata kao trostruki DES ili 3DES. Ali pre toga, potrebno nam je označavanje. Neka P bude blok otvorenog teksta, K je ključ, C je odgovarajući blok šifrata. Za DES, C i P su po 64 bita, dok je K 56 bitova, ali naša notacija je uopštena. Obeležavanje koje ćemo prihvatiti za šifrovanje otvorenog teksta P sa ključem K je:

$$C = E(P, K),$$

a odgovarajuće obeležavanje dešifrovanja:

$$P = D(C, K).$$

Primitićemo da su za isti ključ, šifrovanje i dešifrovanje inverzne operacije, odnosno:

$$P = D(E(P, K), K)$$

$$C = E(D(C, K), K)$$

Međutim, uopšteno je da je:

$$P \neq D(E(P, K_1), K_2)$$

$$C \neq E(D(C, K_1), K_2)$$

kada je:

$$K_1 \neq K_2$$

U jednom trenutku, DES je bio sveprisutan, ali njegova dužina ključa je danas nedovoljna. Međutim, postoji način da se DES koristi sa većom dužinom ključa. Intuitivno, u prvom koraku se pretpostavlja da bi dvostruki DES bio dovoljan, pa sledi:

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

Ovo bi nudilo pogodnosti ključa od 112 bitova (dva DES ključa po 56 bitova), ali jedina mana bi bila gubitak efikasnosti zbog dve DES operacije.

Međutim, postoji napad tipa „eng. *meet-in-the-middle*” na dvostruki DES koji ga čini manje-više sličnim sa običnim DES-om. Iako je ovaj napad nepraktičan, veoma je blizu da bismo se opustili. Ovo je napad izabranog otvorenog teksta gde se pretpostavlja da napadač uvek bira određeni otvoreni tekst P i dobija odgovarajući šifrat C .

Pretpostavimo da će Trudi da izabere određenu otvorenu poruku P i dobije odgovarajući šifrat C , što je za dupli DES:

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

Trudi ima cilj da pronađe ključ ključ K_1 i K_2 . Ona prvo preračunava tabelu veličine 2^{56} koja sadrži parove $E(P, K)$ i K za sve moguće vrednosti ključa K . Trudi sortira ovu tabelu na vrednosti $E(P, K)$. Sada na osnovu ove tabele i šifarskog teksta C , Trudi dešifruje C sa ključem $\tilde{K}$ dok ne nađemo vrednosti $X = D(C, \tilde{K})$ koji je u tabeli. Onda zbog načina na koji je napravljena tabela, imamo da je $X = E(P, K)$ za neke K , a Trudi sada ima:

$$D(C, \tilde{K}) = E(P, K)$$

gde su $\tilde{K}$ i K poznati. Da je trudi pronašla 112-bitni ključ možemo videti šifrovanjem obe strane sa $\tilde{K}$, što daje:

$$C = E(E(P, K), \tilde{K}).$$

Tada u datoj jednačini imamo, imamo:

$$K_1 = K \text{ i } K_2 = \tilde{K}.$$

Ovaj napad na dvostruki DES zahteva da Trudi preračuna, sortira i smesti ogromnu tabelu od 2^{56} elemenata. Ali računanje tabele se radi samo jednom, a ako koristimo tu tabelu više puta (napadajući dvostruki DES), posao za računanje tabele može biti amortizovan preko broja napada. Zanimajući posao potreban za preračun tabele, posao se sastoji od računanja $D(C, K)$ dok ne pronađemo odgovarajuće poklapanje u tabeli. Ovo zahteva posao od 2^{56} kao napad potpunom pretragom ključa na običan DES. Tako da, uopšteno govoreći, dvostruki DES nije ništa sigurniji od običnog DES-a.

Pošto dvostruki DES nije siguran, da li će se trostruki DES pokazati bolje? Pre nego što krenemo da razmatramo napade, moramo definisati trostruki DES. Čini se da bi logičan pristup trostrukom DES-u bio:

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

ali to nije način na koji se to radi. Umesto toga, trostruki DES se definiše kao:

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

Primetićete da trostruki DES koristi samo dva ključa za šifrovanje-dešifrovanje-šifrovanje ili EDE se koristi umesto šifru-šifruj-šifruj EEE. Razlog zašto se koriste samo dva ključa je taj što je 112 bitova dovoljno, ali zašto EDE? Iznenadujuće, odgovor je povratna kompatibilnost.

Ako se 3DES koristi sa $K_1 = K_2 = K$ onda se to pretvara u obični DES

$$C = E(D(E(P, K), K), K) = E(P, K)$$

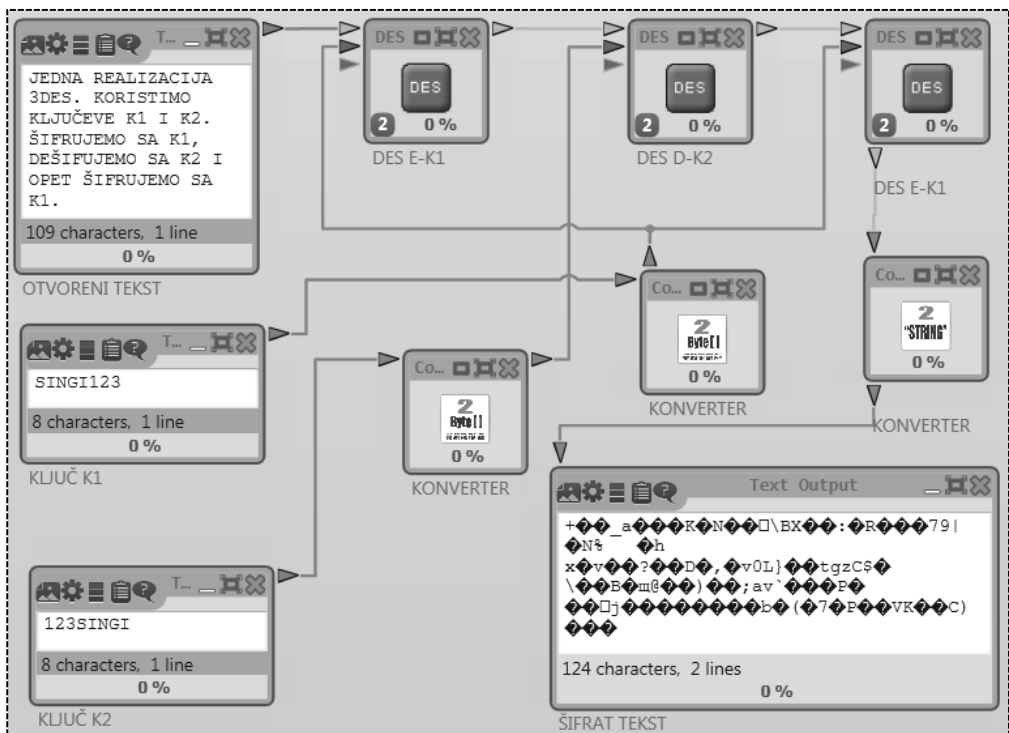
Šta onda da kažemo o napadu na trostruki DES? Možemo reći da je napad "Čovek u sredini" (eng. *meet-in-the-middle*) kao kod dvostrukog DES-a nepraktičan jer je predračun tabele neizvodljiv ili je rad po napadu neizvodljiv.

Trostruki DES ostaje popularan i danas, ali sa razvojem AES-a (*Advanced Encryption Standard*) i još nekih modernih alternativa trostruki DES će postepeno izlaziti iz upotrebe.

Slika 3.23, prikazana je jedna jednostavna realizacija 3DES algoritma u Cryptool-u za šifrovanja izabranog otvorenog teksta. Definisana su dva ključa K_1 i K_2 . Ova implementacija 3DES koristi *EDE* realizaciju, K_1 za šifrovanje i K_2 za dešifrovanje.

25. PRIMER:

Slika 3.23, realizacija 3DES algoritma sa DES algoritmom u Cryptool-u.

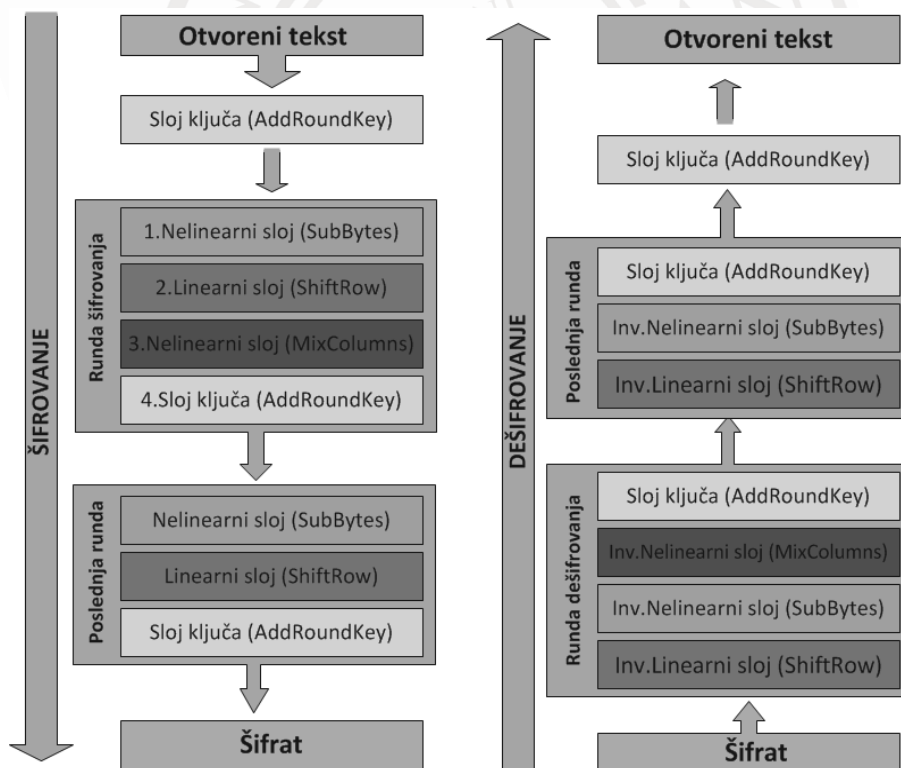


Slika 3.23 Primer sa 3DES algoritmom u Cryptool-u

3.2.4. AES

Tokom 1990-tih shvatili su svi, pa i američka vlada da je DES nadživeo svoju korisnost. Presudan problem sa DES-om jeste da je dužina ključa od 56 bitova osetljiva na napad potpunom pretragom ključa. Posebno razvijeni programi za razbijanje DES-a mogu da otkriju DES ključeve za nekoliko sati, a i distribuirani napadi koristeći računare dobrovoljnih učesnika na Internetu, uspeali su da pronađu DES ključeve.

Početkom 1990-ih, Nacionalni institut za standarde i tehnologiju (NIST) koji je danas inkarnacija NBS-a, izdao je zahtev za kriptografski predlog algoritma (*Advanced Encryption Standard*) ili AES. Za razliku od zahteva za DES pre dvadeset godina, NIST je bio preplavljen kvalitetnim predlozima. Broj kandidata se sveo na pregršt finalista i algoritam poznat pod nazivom "Rijndael" (izgovara se nešto slično kao eng. *Rain doll*) je bio odmah izabran.



Slika 3.24 AES algoritam – šifrovanje/dešifrovanje

AES nadmetanje je vođeno na fer način i za razliku od DES nadmetanja, NSA je bila javno prisutna kao jedan od sudija. Zbog toga, ne postoje verodostojne tvrdnje da su neka "zadnja vrata" ubačena u AES. AES je visoko rangiran u kriptografskoj zajednici. Šamir je izjavio da će AES sa ključem od 256 bitova biti siguran zauvek bez obzira na predstojeće napretke u računarskoj tehnologiji.

Kao i DES, AES je iterativna blokovska šifra. Ali za razliku od DES-a, AES algoritam nije Fejstel šifra. Glavna posledica toga je da AES operacije moraju biti povratne da bi se nešto moglo dešifrovati. Takođe, za razliku od DES-a, AES algoritam ima komplikovanu matematičku strukturu. Mi ćemo dati kraći opis algoritma. Velika je količina informacija o svim aspektima AES-a dostupna za čitanje, ali mi ćemo zaobići elegantnu matematičku strukturu. U svakom slučaju, nijedan algoritam u istoriji nije bio više ispitivan kao AES za relativno kratko vreme.

AES je otporan na poznate napade, veoma je brz, moguć je paralelni dizajn, kao i implementacija na mnogim procesorima i pametnim karticama. Navešćemo ukratko osnovnu numerologiju ili specifikaciju AES algoritma.

AES je šifra sa sledećim funkcionalnim parametrima:

- Dužina bloka otvorenog teksta je 128, 192 i 256 bitova;
- Dužine ključa je 128, 192 i 256 bitova;
- Ima ukupno od 10 do 16 rundi, zavisno od dužine ključa;
- U svakoj rundi koriste se 4 funkcije;
 - Nelinearni sloj (*ByteSub*),
 - Sloj lineranog mešanja (*ShiftRow*),
 - Nelinearni sloj (*MixColumns*),
 - Dodatni sloj ključa (*AddRoundKey*).

Bitno je napomenuti za AES da je dužina bloka otvorenog teksta u potpunosti nezavisna od dužine ključa. To znači da je moguće šifrovati blok otvorenog teksta dužine 128 bitova sa ključem dužine 256 bitova.

Sve operacije AES algoritma obavljaju se na dvodimenzionalnom nizu bitova, odnosno matrici stanja. Šifrovanje, odnosno dešifrovanje se vrši tako što se ulazni blok podatka kopira u matricu stanja nad kojom se sprovode razne operacije. Završna

vrednost matrice ili stanje, kopira se u izlazni šifarski blok. AES ulazne podatke tretira kao matrice dimenzije $[4 \times 4]$.

Nelinearni sloj (*SubBytes*)

Ulaz					Izlaz			
19	a0	9a	e9		d4	e0	b8	1e
3d	f4	c6	f8	→	27	bf	b4	41
e3	e2	8d	48		11	98	5d	52
be	2b	2a	08		ae	f1	e5	30

U tabeli je prikazan rezultat operacije *SubBytes* za dati ulaz. Ulaz je matrica dimenzije $[4 \times 4]$ sa ukupno 16 bajtova. U tabeli iznad, ilustrovan je rezultat funkcije *SubBytes* za vrednost prvog bajta. Bajt **19**, transformisan je u bajt **d4**.

hex	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb

Slika 3.25 S-kutija AES algoritma

SubBytes predstavlja AES-ov ekvivalent DES S-kutije i može se posmatrati kao nelinearna, ali invertibilna kompozicija dve matematičke funkcije ili se može jednostavno posmatrati kao *lookup* tabela, **Slika 3.25**.

Linearni sloj – pomeranje redova (*ShiftRow*)

Operacija pomeranja redova ili jednostavno ciklično pomeranje bajtova u poslednja tri reda matrice dimenzije [4 x 4] je predstavljena ispod.

Ulaz						Izlaz			
d4	e0	b8	1e			d4	e0	b8	1e
27	bf	b4	41	→	1	bf	b4	41	27
11	98	5d	52	→→	2	5d	52	11	98
ae	f1	e5	30	→→→	3	30	ae	f1	e5

Prvi red se ne pomera, drugi red se pomera za jedan bajt ulevo, treći red za 2 bajta ulevo i poslednji red za 3 bajta u levo. Primetićemo da je *ShiftRow* invertovan prostim pomeranjem u suprotnu stranu, a iz tog razloga spada u linearne funkcije.

Nelinearni sloj mešanja kolona (*MixColumns*)

Operacija mešanja kolona je primenjena za svaku kolonu matrice dimenzije [4 x 4], kao što je prikazano u tabeli ispod.

Ulaz u funkciju za mešanje kolona:

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

Izračunaćemo izlaz za prvu kolonu ulaza:

Ulaz				K1	K1'
02	03	01	01	d4	04
01	02	03	01	X bf	→ 66
01	01	02	03	5d	81
03	01	01	02	30	e5

Rezultat funkcije mešanja je:

Izlaz			
04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	43

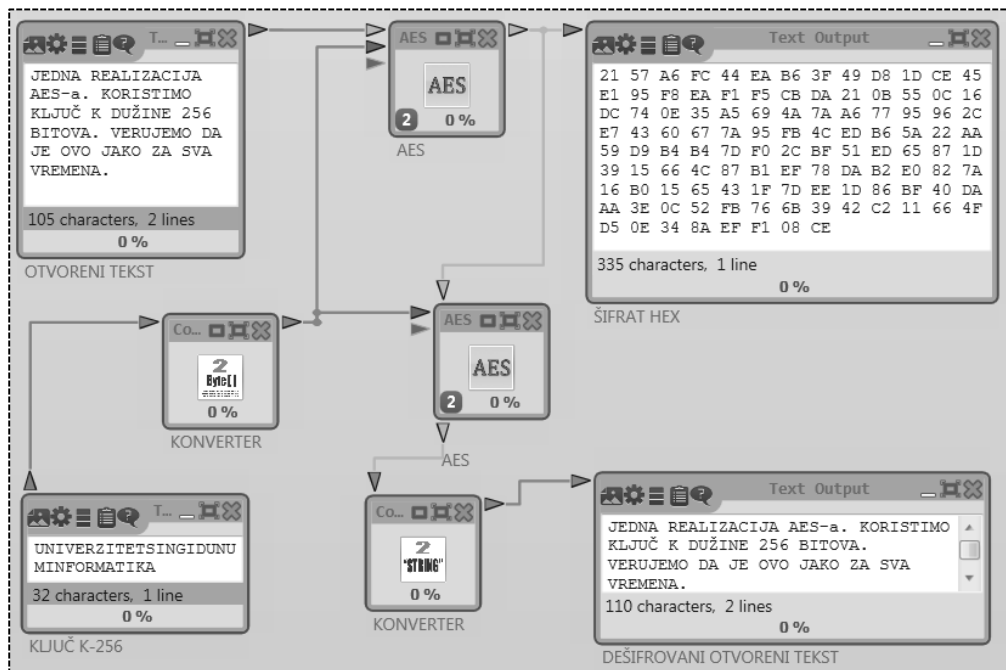
Mešanje kolona sastoji se od pomeranja i XOR operacije i najbolje je implementirana kao lookup tabela. Sveobuhvatna operacija je nelinearna, ali invertibilna, kao i *SubBytes* ima ulogu istu kao S-kutije kod DES-a. Miksovanje kolona (*MixColumns*) sa pomeranjem redova (*ShiftRow*) u AES-u predstavlja osnovni izvor difuzije.

Dodavanje sloja ključa (*AddRoundKey*)

Dodavanje generisanog sloja ključa je jednostavna operacija. Kao kod DES-a, algoritam rasporeda ključa je korišćen da bi se generisao podključ za svaku rundu. Neka K_{ij} bude $[4 \times 4]$ matrica podključa za određenu rundu. Onda se podključ XOR-uje sa trenutnom $[4 \times 4]$ matricom a_{ij} koja predstavlja ulaz kao što je prikazano dole:

Ulaz				K-runda 1					Izlaz				
04	e0	48	28	XOR	a0	88	23	2a	=	a4	68	6b	02
66	cb	f8	06		fa	54	a3	6c		9c	9f	5b	6a
81	19	d3	26		fe	2c	39	76		7f	35	ea	50
e5	9a	7a	43		17	b1	39	05		f2	2b	43	49

Zanemarićemo AES raspored ključa, ali kao i kod svake blokovske šifre to je značajan deo sigurnosti algoritma. Na kraju, kao što smo napomenuli gore, svaka od četiri funkcije *SubBytes*, *ShiftRow*, *MixColumns* i *AddRoundKey* su invertibilne. Rezultat, je da je čitav algoritam invertibilan i na osnovu toga AES se može dešifrovati kao što se može i šifrovati.



Slika 3.26 Šifrovanje i dešifrovanje sa AES algoritmom

Slika 3.26, prikazana je jedna jednostavna realizacija AES algoritma u Cryptool-u za šifrovanja izabranog otvorenog teksta. Definisan je ključa K . Ova implementacija AES-a sadrži proces šifrovanja i dešifrovanja.

26. PRIMER:

Slika 3.27, još jedna realizacija AES algoritma u Java programskom jeziku.

```
package aes;

import java.security.Key;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import sun.misc.BASE64Encoder;

public class AES {
    //----- AES metod za šifrovanje -----
    public static byte[] sifrovanje(byte[] tekst, byte[] bKljuc) throws Exception {
        Key kljuc = new SecretKeySpec(bKljuc, "AES");
        Cipher c = Cipher.getInstance("AES");
        c.init(Cipher.ENCRYPT_MODE, kljuc);
        byte[] encVal = c.doFinal(tekst);
        return encVal;
    }
    //----- AES metod za sešifrovanje -----
    public static byte[] desifrovanje(byte[] tekst, byte[] bKljuc) throws Exception {
        Key kljuc = new SecretKeySpec(bKljuc, "AES");
        Cipher c = Cipher.getInstance("AES");
        c.init(Cipher.DECRYPT_MODE, kljuc);
        byte[] encVal = c.doFinal(tekst);
        return encVal;
    }
    //----- Izvršni metod koji se koristi za šifrovanje/dešifrovanje
    public static void main(String[] args) throws Exception{
        byte[] kljuc = "1234567891234567".getBytes(); //minimum 16 bajtova -(128bit)
        byte[] text = "Singidunum Univerzitet, Kriptologija 1".getBytes();
        byte[] sifrovano = sifrovanje(text, kljuc);
        byte[] desifrovano = desifrovanje(sifrovano, kljuc);
        System.out.println(new BASE64Encoder().encode(sifrovano));
        System.out.println(new String(desifrovano));
    }
}
```

Slika 3.27 Implementacija AES-a u Java programskom jeziku

3.2.5. JOŠ TRI BLOKOVSKE ŠIFRE

U ovom delu ćemo kratko obraditi tri dobro poznata blokovska algoritma, IDEA (*International Data Encryption Algorithm*), *Blowfish*, i RC6. Svaki od njih ima neke specifičnosti u svom dizajnu. U sledećem delu razmatra se TEA (*Tiny Encryption Algorithm*).

IDEA je delo Džejsma L. Masija (*James L. Massey*) jednog od najvećih kriptografa modernog vremena. IDEA ima blok dužine 64 bita i ključ dužine 128 bitova. Najsavremenija odlika IDEA je to što koristi izmešanu modularnu aritmetiku. Algoritam kombinuje sabiranje po modulu 2 ili XOR sa sabiranjem po modulu 2^{16} i Masejovo množenje koje je skoro množenje po modulu 2^{16} . Ove operacije zajedno proizvode potrebnu nelinearnost, a rezultat je da nije potrebna eksplicitna S-kutija. Kao što vidimo, IDEA kombinuje različite matematičke operacije. Masi je očigledno prvi koji je koristio ovaj pristup, koji je uobičajen danas.

Blowfish je jedan od Brus Šnajerovih (*Bruce Schneier*) omiljenih kriptografskih algoritama jer ga je on i dizajnirao. Šnajer je poznati kriptograf i dobar pisac o svim stvarima koje se tiču bezbednosti. Interesantna stvar kod *Blowfisha* jeste ta što koristi S-kutije zavisne od veličine ključa, umesto fiksnih S-kutija. *Blowfish* generiše S-kutije zasnovane na ključu. Može se dokazati da su tipične *Blowfish* S-kutije dovoljno jake. *Blowfish* šifrjuje 64-bitne blokove. Ključ je primenjive dužine, maksimalne veličine od 32 do 448 bitova. Po strukturi se skoro poklapa sa Fejstelovim dizajnom šifre.

$$\begin{aligned} D_i &= L_{i-1} \oplus K_i \\ L_i &= D_{i-1} \oplus F(L_{i-1} \oplus K_i) \end{aligned}$$

Funkcija F runde koristi 4 S-kutije. Svaka S-kutija preslikava 8 bitova u 32 bita.

RC6 je delo Ron Rivest (*Ron Rivest*) čija su kriptografska dostignuća stvarno zadivljujuća, uključujući i sistem javnih ključeva RSA, prethodno spomenutu RC4 sekvencijalnu šifru, kao i jednu od najpoznatijih heš funkcija MD5. Neobičan aspekt RC6 je to što koristi rotacije zavisne od podataka. RC6 šifrjuje blok veličine 128 bitova podatka. Dužine ključeva su 128, 192, 256 i više bitova. Broj rundi je od 0 do 255. Veoma je neobično da se oslanja na podatke kao na jedan od suštinskih delova

operacije kriptografskog algoritma. RC6 je bio jedan od AES finalista, iako je na kraju izgubio od Rijindael. Ove tri šifre ilustruju mali uzorak mnogih varijacija korišćenih u potrazi idealnog balansa između sigurnosti i performansi u dizajnu blokovskih šifara.

3.2.6. TEA

Poslednja blokovska šifra o kojoj ćemo govoriti je *Tiny Encryption Algorithm* ili TEA. Dijagrami koje smo prikazali do sada mogu uputiti na to da su blokovske šifre potrebno kompleksne. TEA veoma lepo pokazuje da to nije slučaj.

TEA koristi blok dužine 64 bita i ključ dužine 128 bitova. Algoritam podrazumeva sabiračku aritmetiku sa 32 bita reči, tako da sve matematičke operacije koriste moduo 2^{32} i svi bitovi preko 32 bita dužine se automatski odsecaju. Broj rundi varira, ali mora biti relativno veliki. Opšte prihvaćeno mišljenje je da su 32 runde sigurne. Ipak, svaka runda TEA algoritma je otprilike kao dve runde Fejstelove šifre (slično DES-u), pa je ovo grubo jednako kao 64 runde kod DES. To je relativno dosta rundi.

U dizajnu blokovskih šifara mora uvek da postoji kompromis između kompleksnosti svake runde i potrebnog broja rundi. Šifre kao što je DES pokušavaju da pogode balans između ove dve stvari, dok AES redukuje broj rundi koliko god je to moguće, ali o trošku veće kompleksnosti funkcije runde. Na neki način, TEA predstavlja potpunu suprotnost AES-u pošto TEA koristi veoma proste funkcije runde. Kao posledica ovih prostih rundi, broj rundi mora biti veliki kako bi se postigao visok nivo sigurnosti. Pseudokod za šifrovanje sa TEA algoritmom sa pretpostavkom da su korišćene 32 runde, **Tabela 8**, gde je simbol "<<" levo pomeranje (ne kružno, šiftovanje) i simbol ">>" desno pomeranje (ne kružno).

Tabela 8 Šifrovanje sa TEA algoritmom

```

(K[0],K[1],K[2],K[3])=128 bitova ključ
(L,D) = otvoreni_tekst
delta = 0x9e3779b9
suma = 0
za i = 1 do 32
    suma = suma + delta
    L = L + (((D<<4)+K[0])XOR(D+suma)XOR((D>>5)+K[1]))
    D = D + (((L<<4)+K[2])XOR(L+suma)XOR((L>>5)+K[3]))
sledeći i
Šifrat = (L,D)

```

Jedna zanimljiva stvar koju treba primetiti kod TEA jeste da ona nije Fejstel šifra tako da treba razdvojiti rutine šifrovanja i dešifrovanja. Ipak, TEA je toliko blizu da bude Fejstel šifra, ali opet dovoljno daleko da to ne bude TEA. TEA koristi sabiranje i oduzimanje umesto XOR-ovanja. Međutim, potreba za razdvajanje rutina šifrovanja i dešifrovanja je najmanji problem kod TEA jer se koristi veoma malo koda, a i algoritam je takođe razumno efikasan iako sadrži veliki broj rundi. TEA algoritam za dešifrovanje sa 32 runde, **Tabela 9**.

Tabela 9 Dešifrovanje sa TEA algoritmom

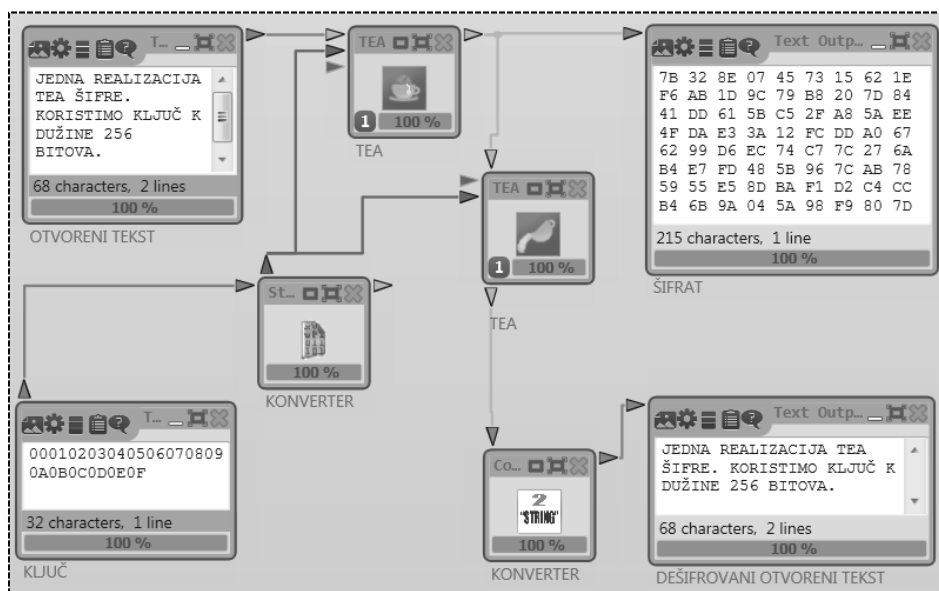
```

(K[0],K[1],K[2],K[3])=128 bitova ključ
(L,D)= šifrat
delta = 0x9e3779b9
suma = delta<<5
za i = 1 do 32
    D = D - (((L<<4)+K[2])XOR(L+suma)XOR((L>>5)+K[3]))
    L = L - (((D<<4)+K[0])XOR(D+suma)XOR((D>>5)+K[1]))
    suma = suma - delta
sledeći i
otvoreni_tekst = (L,D)

```

Slika 3.28 TEA šifrovanje i dešifrovanje, prikazana je jedna jednostavna realizacija TEA algoritma u Cryptool-u. Definisan je ključa K . Ključ je prosleđen algoritmu u

heksadecimalnom format. Ova implementacija TEA šifre sadrži proces šifrovanja i dešifrovanja.



Slika 3.28 TEA šifrovanje i dešifrovanje

Moguć je napad na ključ ili napad "povezanim ključem" na TEA. Ako kriptanalitičar zna da su dve poruke šifrovane ključevima koji su povezani na poseban način onda otvoreni tekst može biti otkriven. Ovo je napad veoma male verovatnoće koji se u većini slučajeva može sigurno ignorisati. Ali ako ste kojim slučajem malo zabrinuti za ovaj napad, postoji malo kompleksnija varijanta TEA, poznata kao proširena TEA ili XTEA koja prevazilazi ovaj problem. Postoji i pojednostavljena verzija TEA poznata kao STEA, koja je veoma slaba i omogućava da se ilustruju određeni tipovi napada.

3.3. REŽIMI RADA BLOKOVSKIH ŠIFARA

Korišćenje sekvencijalnih šifara je jednostavno. Generiše se ključ koji je iste dužine kao otvoreni tekst (ili šifrat) i XOR-uje se. Korišćenje blokovskih šifara je takođe jednostavno, sve dok je na raspolaganju samo jedan blok za šifrovanje. Kako se može šifrovati više blokova? Izgleda da ipak nije tako jednostavno kao što se čini.

Pretpostavimo da imamo niz blokova nekog otvorenog teksta, na primer:

$$P_0, P_1, P_2, \dots$$

Za izabrani fiksni ključ K , blokovska šifra je kodna knjiga iz razloga što kreira jednoznačno preslikavanje između blokova otvorenog teksta i blokova šifrata. Sledeći ideju kodne knjige, očigledna stvar koju treba uraditi je da se koristi blokovska šifra u takozvanom režimu "elektronske kodne knjige" ili *ECB* režimu. U *ECB* režimu, šifrujemo tako što koristimo formulu:

$$C_i = E(P_i, K) \text{ za } i = 0, 1, 2, \dots, \text{ na primer:}$$

$$C_0 = E(P_0, K)$$

$$C_1 = E(P_1, K)$$

$$C_2 = E(P_2, K), \dots$$

Takođe, dešifrujemo na osnovu sledeće formule:

$$P_i = D(C_i, K) \text{ za } i = 0, 1, 2, \dots, \text{ na primer:}$$

$$P_0 = D(C_0, K)$$

$$P_1 = D(C_1, K)$$

$$P_2 = D(C_2, K), \dots$$

Ovaj pristup funkcioniše, ali postoji nekoliko ozbiljnih sigurnosnih problema sa *ECB* režimom. Kao rezultat istraživanja na temu sigurnosti, *ECB* režim ne bi trebalo da se koristi u praksi.

Pretpostavimo da je *ECB* režim korišćen i da napadač primeti da je $C_i = C_j$. Posle toga napadač sa sigurnošću zna da je $P_i = P_j$. Iako se ovo može smatrati bezazlenim, postoje slučajevi gde napadač zna deo otvorenog teksta i svako podudaranje sa poznatim blokom otkriva novi blok. Čak i kada napadač ne zna P_i ili P_j da su neke informacije otkrivene, tačnije da su ova dva bloka otvorenog teksta ista. Mi ne želimo

da kryptoanalitičaru odamo nešto tako lako, pogotovo ako postoje jednostavni načini da se takva anomalija izbegne.

Masi daje dramatičnu ilustraciju posledica ove takozvane male slabosti. Dat je sličan primer (**Slika 3.29.**), koji prikazuje nešifrovanu sliku logoa Univerziteta Singidunum, pored slike koja je šifrovana u *ECB* režimu. Svaki blok na desnoj slici je šifrovan, ali blokovi koji su isti na originalnoj slici, isti su i u *ECB* šifrovanoj poruci. Primetićemo da nije bitno koja blokovska šifra je korišćena, pomalo neočekivan rezultat **Slika 3.29.**, jedino zavisi od činjenice da je korišćen *ECB* režim, a ne od detalja algoritma. U ovom slučaju nije teško za napadača da pogodi otvoreni tekst iz dobijenog šifrata.

27. PRIMER:



Slika 3.29 *ECB* režim

Na sreću, postoji i bolji način korišćenja blokovskih šifara, kojim se izbegava slabost *ECB* režima. Sada ćemo govoriti o najčešće korišćenom metodu, a to je metod ulančavanja blokova šifrata ili *CBC* režimu. U *CBC* režimu, šifrat od prethodnog bloka otvorenog teksta koristi se za "zamagljivanje" otvorenog teksta sledećeg bloka, pre nego što se primeni algoritam za šifrovanje.

Formula za šifrovanje u *CBC* režimu je:

$$C_i = E(P_i \oplus C_{i-1}, K) \text{ za } i = 0, 1, 2, \dots, \text{ na primer:}$$

$$C_0 = E(IV \oplus P_0, K)$$

$$C_1 = E(C_0 \oplus P_1, K)$$

$$C_2 = E(C_1 \oplus P_2, K), \dots$$

Zatim, sledi dešifrovanje na osnovu formule:

$$P_i = D(C_i, K) \oplus C_{i-1} \text{ za } i = 0, 1, 2, \dots, \text{ na primer:}$$

$$P_0 = IV \oplus D(C_0, K)$$

$$P_1 = C_0 \oplus D(C_1, K)$$

$$P_2 = C_1 \oplus D(C_2, K), \dots$$

Prvi blok zahteva poseban način vođenja pošto nema šifarskog bloka C_{i-1} . IV predstavlja inicijalni vector i poželjno je da bude generisan na čisto slučajan način. Nije neophodno da bude tajan. Koristi se samo za generisanje prvog bloka šifrata. Dužina inicijalnog vektora IV je uvek jednaka veličini bloka date šifre.

Prednost *CBC* režima je u tome da identičan otvoreni tekst neće odgovarati identično šifratu. Ovo je ilustrovano poređenjem nešifrovane slike logoa Univerziteta Singidunum sa istom slikom koja je dobijena od šifrata, **Slika 3.30**.

28. PRIMER:



Slika 3.30 *CBC* režim

Zbog ulančavanja, moguća briga sa *CBC* režimom jeste nastanak grešaka. Kada je šifrat poslat kroz komunikacioni kanal, u toku prenosa mogu nastati greške. Nula može postati jedinica ili obrnuto. U tom slučaju jedna nastala greška na bloku šifrata će učiniti otvoreni tekst nepopravljivim. Ako jedna prenesena greška učini otvoreni tekst nepopravljivim, onda je *CBC* režim beskoristan u mnogim aplikacijama. To bi značilo da je lanac pokidan na određenim mestima. Svakako, u cilju umanjenja rizika

od nastanka grešaka koriste se zaštitni kodovi (eng. *Error Corecting Codes*) koji su dobro poznati u teoriji komunikacija.

Pretpostavimo da je blok šifrata C_i oštećen recimo u $G \neq C_i$. Onda je:

$$P_i \neq D(G, K) \oplus C_{i-1} \text{ i } P_{i+1} \neq D(C_{i+1}, K) \oplus G, \text{ ali}$$

$$P_{i+2} \neq D(C_{i+2}, K) \oplus C_{i+1},$$

i svi sledeći blokovi su dešifrovani pravilno. Tačnije, svaki blok otvorenog teksta jedino zavisi od dva uzastopna bloka šifrata tako da se greška ne nagomilava dalje od dva bloka. Ipak, činjenica da greška od jednog bita može da prouzrokuje da dva čitava bloka budu neispravna, ozbiljna je briga u okruženju sa visokom verovatnoćom greške kao što je bežični prenos podataka (WiFi). Sekvencijalne šifre nemaju ovaj problem jer jedan pokvaren bit šifrata rezultuje jednim pokvarenim bit-om u otvorenom tekstu i to je razlog zašto se sekvencijalne šifre koriste kod bežičnih aplikacija (GSM).

Još jedna briga kod blokovskih šifara je "iseći nalepi" (eng. *cut-paste*) napad. Pretpostavimo da imamo otvoreni tekst, napisan je u drugom redu, **Tabela 10**.

Tabela 10 Napad „iseći nalepi“ A

	Blok P_0								Blok P_1								Blok P_2							
1	N	o	v	a	c	_	z	a	_	A	l	i	s	u	_	j	e	_	1	0	0	R	S	D
2	N	o	v	a	c	_	z	a	_	T	r	u	d	i	_	j	e	_	0	5	0	R	S	D
	Blok P_3								Blok P_4								Blok P_5							

Otvoreni tekst je podeljen na tri bloka. Karakter "_" je prazno mesto. Zatim poruku šifrujemo šifrom koja radi nad ulaznim blokovima otvorenog teksta podatka veličine 64 bita, pretpostavlja da svaki karakter koristi 8 bitova (npr. 8-bitni ASCII).

Pretpostavimo da je šifrat nastao u *ECB* režimu. Onda su blokovi šifrata izračunati kao:

$$C_i = E(P_i, K) \text{ za } i = 0, 1, \dots, 3;$$

Sada pretpostavimo da Trudi zna da je korišćen *ECB* režim, poznaje generalnu strukturu otvorenog teksta i zna da će ona primiti 50 RSD, ali ne zna koliko će Alisa primiti. Ona pretpostavlja da je u pitanju duplo veći iznos od njenih 50 RSD. Trudi sada možda može reorganizovati blokove šifrata u:

$$C_0, C_1, C_5, C_3, C_4, C_2,$$

zatim će Bob ovo dešifrovati kao, **Tabela 11**:

Tabela 11 Napad „iseći nalepi“ B

	Blok P_0										Blok P_1										Blok 3 P_2									
1	N	o	v	a	c	_	z	a	_	A	I	i	s	u	_	j	e	_	0	5	0	R	S	D						
2	N	o	v	a	c	_	z	a	_	T	r	u	d	i	_	j	e	_	1	0	0	R	S	D						
	Blok P_3										Blok P_4										Blok P_5									

Međutim, pogrešno je misliti da *CBC* režim može eliminisati “iseći nalepi” napad. Napad je moguć iako je malo teži za realizaciju i neki podaci mogu biti oštećeni.

Takođe je moguće koristiti blokovske šifre za generisanje kriptoloških ključeva koji mogu bit korišćeni kao ključevi za sekvencijalne šifre. Postoji nekoliko prihvatljivih načina da se ovo uradi, ali mi ćemo spomenuti samo jedan najpopularniji, a to je režim brojača ili *CTR*. Kao i *CBC*, *CTR* režim koristi inicijalni vektor *IV*. Što se tiče rada *CTR* režima, *IV* se šifruje, a zatim XOR-uje sa blokom otvorenog teksta. Za svaki naredni blok otvorenog teksta, vrednost *IV* se uvećava za 1. Formula za šifrovanje u *CTR* režimu je:

$$C_i = P_i \oplus E(IV + i, K) \text{ za } i = 0, 1, 2, \dots, \text{ na primer:}$$

$$C_0 = P_0 \oplus E(IV, K)$$

$$C_1 = P_1 \oplus E(IV + 1, K)$$

$$C_2 = P_2 \oplus E(IV + 2, K)$$

a za dešifrovanje:

$$P_i = C_i \oplus E(IV + i, K) \text{ za } i = 0, 1, 2, \dots, \text{ na primer:}$$

$$P_0 = C_0 \oplus E(IV, K)$$

$$P_1 = C_1 \oplus E(IV + 1, K)$$

$$P_2 = C_2 \oplus E(IV + 2, K), \dots$$

CTR režim se često koristi kada je potreban nasumičan pristup. Dok se nasumični pristup sasvim jednostavno koristi u *CBC* režimu, u nekim slučajevima u *CBC* režimu neće biti poželjan nasumičan pristup.

Pored *ECB*, *CBC* i *CTR* režima, postoje još mnogi drugi režimi koji koriste blokovske šifre. Dali smo opis onih koji se najviše koriste. Ipak, tri režima o kojima smo najviše govorili sigurno su najzastupljeniji u odnosu na ostale.

Najzad, treba napomenuti da se bezbednost podataka najčešće odnosi na dva blago različita slučaja. Sa jedne strane, šifrujemo podatke da bi mogli da ih pošaljemo preko nesigurnog kanala. Sa druge strane, šifrujemo podatke koji se čuvaju na nezaštićenim medijima kao što je računarski hard disk. Simetrične šifre mogu da se koriste da bi se rešio bilo koji od ova dva veoma slična problema. Simetrični ključ može da štiti integritet podatka kao što ćemo videti u narednom poglavlju.

3.4. INTEGRITET

Dok se tajnost bavi prevencijom neautorizovanog čitanja, integritet se bavi prevencijom neautorizovanog pisanja. Na primer, pretpostavimo da se vrši elektronski transfer sredstava sa jednog računa na drugi. Vi ne želite da drugi znaju za ovu transakciju i u tom slučaju će šifrovanje obezbediti željenu poverljivost. Svejedno da li postoji zabrinutost za poverljivost ili ne, mnogo je važniji zahtev da se transakcija obavi onako kako treba. U takvim slučajevima govorimo o integritetu podataka, tj. o nemogućnosti njihove neovlašćene promene.

U prethodnom delu proučavali smo blokovske šifre i njihovo korišćenje vezano za poverljivost. Ovde ćemo pokazati kako se blokovske šifre koriste kako bismo obezbedili integritet podataka.

Važno je shvatiti da su tajnost i integritet dva različita pojma. Šifrovanje sa nekom šifrom, od *One-time pad*-a do blokovskih šifri, ne štiti podatke od malicioznih ili nepažljivih promena. Ako Trudi promeni šifrat ili ako se dogodi greška tokom prenosa, integritet podataka je narušen i mi želimo da znamo da je došlo do promena. Pokazali smo nekoliko primera, a biće ih i još koji pokazuju da šifrovanje ne osigurava integritet.

Autentifikacioni kod poruke ili MAC koristi blokovsku šifru da osigura integritet podatka. Procedura je jednostavno šifrovanje podataka u *CBC* režimu, odbacujući sve šifarske blokove, osim poslednjeg. Ovaj poslednji blok šifrata, poznat kao *CBC* ostatak služi kao MAC. Onda je formula za MAC, pretpostavljajući n blokova podataka, $P_0, P_1, P_2, \dots, P_{n-1}$ data kao:

$$C_0 = E(P_0 \oplus IV, K),$$

$$C_1 = E(P_1 \oplus C_0, K),$$

.....

$$C_{n-1} = E(P_{n-1} \oplus C_{n-2}, K) = MAC$$

Primetićemo da se koristi IV i da je potrebno razmeniti zajednički simetrični ključ. Da bismo uprostiti, pretpostavimo da Alisa i Bob zahtevaju integritet, ali da nisu zabrinuti za poverljivost. Zatim koristeći ključ K koji Alisa i Bob dele, Alisa računa MAC i šalje otvoreni tekst, inicijalni vektor IV i MAC Bobu. Po prijemu poruke, Bob računa MAC koristeći ključ K i primljeni IV i otvoreni tekst. Ako je izračunati MAC jednak primljenom MAC-u, onda je integritet podatka potvrđen. Ako se izračunati MAC ne slaže sa primljenim MAC-om, Bob zna da je nešto nije u redu. Još jednom, kao i u *CBC* režimu, pošiljalac i primalac moraju unapred razmeniti isti simetrični ključ K .

Zašto se računanje MAC-a radi? Pretpostavimo da Alisa pošalje Bobu:

$$IV, P_0, P_1, P_2, P_3, MAC'$$

Sada, ako Trudi promeni blok otvorenog teksta P_1 u recimo X tokom prenosa, onda kada Bob bude pokušao da verifikuje MAC on računa:

$$C_0 = E(P_0 \oplus IV, K), C_1 = E(X \oplus C_0, K), C_2 = E(P_2 \oplus C_1, K),$$

$$C_3 = E(P_3 \oplus C_2, K) = MAC \neq MAC'.$$

Ovo radi zbog toga što se svaka promena u bloku otvorenog teksta prenosi kroz sledeće blokove u procesu izračunavanja MAC-a.

Prisetimo se da kod *CBC* režima dešifrovanja, promena u bloku šifrata se manifestuje samo na dva bloka otvorenog teksta. Nasuprot tome, MAC koristi činjenicu da se kod *CBC* šifrovanja bilo kakva promena na otvorenom tekstu skoro sigurno umnožava do poslednjeg bloka. Ovo je ključna osobina koja omogućava MAC-u da obezbedi integritet.

Često su potrebni i poverljivost i integritet da bismo ovo postigli, mogli bismo da računamo MAC sa jednim ključem, a onda da šifrujemo podatak sa drugim ključem. Međutim, to ima dvostruko više posla nego što je potrebno za postizanje samo integriteta ili pouzdanosti. Zbog efikasnosti bi bilo korisno da se postigne i tajnost i integritet sa jednim *CBC* šifrovanjem podataka. Pretpostavimo da *CBC* režimom šifrujemo podatak jednom, pošaljemo dobijeni šifrat i izračunamo MAC. Onda bismo slali ceo šifrat zajedno sa poslednjim blokom (ponovo). Poslednji blok bi bio dupliran i poslat dva puta. Očigledno je da slanje iste stvari dva puta ne doprinosi dodatnoj sigurnosti. Na nesreću, ne postoji primenljiv način da se postigne i tajnost i integritet sa jednim šifrovanjem podatka.

Računanje MAC-a zasnovanog na *CBC* režimu šifrovanja nije jedini način koji obezbeđuje integritet podatka. Hešovan MAC ili HMAC je još jedan standard pristupa integritetu, a i digitalni potpis se nudi kao još jedna opcija. Detaljnije karakteristike HMAC-a i digitalno potpisivanje, razmatra se u kasnijim poglavljima.

3.5. KRATAK PREGLED

U ovom prethodnom poglavlju se nalazi dobar deo materijala o kriptografiji sa simetričnim ključevima. Postoje dva različita tipa simetričnih šifri: sekvencijalne šifre i blokovske šifre. Ukratko smo govorili o dve sekvencijalne šifre, A5/1 i RC4. Setite se da su sekvencijalne šifre pseudo *One-time pad* šifre gde smo menjali dokazivu sigurnost za praktičnost.

Blokovske šifre, sa druge strane, mogu se smatrati kao "elektronski" ekvivalent klasične kodne knjige. Govorili smo detaljno o DES blokovskoj šifri i spomenuli smo ukratko nekoliko drugih blokovskih šifara. Takođe smo uzeli u obzir različite režime korišćenja blokovskih šifara, a ističemo *ECB*, *CBC* i *CTR* režime. Takođe smo pokazali da blokovske šifre mogu biti korišćene i za obezbeđenje integriteta podataka.

U kasnijim poglavljima ćemo videti da su simetrične šifre takođe korisne u protokolima za autentifikaciju. Interesantno je primetiti da su sekvencijalne šifre, blokovske šifre i heš funkcije (pokrivene u narednom poglavlju) ekvivalentne jedne drugima na neki način jer sve što možemo uraditi sa jednom, možemo postići i sa ostale dve, mada u nekim slučajevima ne bi bilo prirodno da se stvarno uradi tako. Iz tog razloga, sva tri se često mogu proglasiti kriptografskim "primitivcima".

Kriptografija simetričnih ključeva je jedna obimna oblast i ovde smo samo "zagrevali po površini". Međutim, opremljeni sa znanjem iz ovog poglavlja bićemo spremni da se diskutujemo o bilo kojoj temi koja se tiče asimetričnih šifara koje se javljaju u kasnijim poglavljima.

4. ASIMETRIČNI ŠIFARSKI SISTEMI

 simetrični šifarski sistemi predstavljaju jedno od najvećih dostignuća kriptologije sedamdesetih godina dvadesetog veka. Nastali su na osnovu rešavanja ključnih problema simetričnih šifarskih sistema koji se odnose na distribuciju tajnih simetričnih ključeva. U literaturi se koriste i termini: kriptografija sa javnim ključevima (eng. *Public Key Cryptography* – PKI) ili kriptografija sa dva ključa (eng. *two key cryptography*).

U simetričnim šifarskim sistemima isti ključ se koristi i za šifrovanje i za dešifrovanje. U asimetričnoj kriptografiji jedan ključ se koristi za šifrovanje, a drugi za dešifrovanje. Suština je u tome da ključ za šifrovanje može biti javan, a samo ključ za dešifrovanje mora da bude tajan (privatan). Naravno, ova dva ključa su u jedinstvenoj vezi. Šifarski algoritmi u asimetričnim sistemima su takvi da se jedino primenom para ključeva može realizovati uspešno šifrovanje i dešifrovanje. Dešifrovanje nije moguće istim ključem kojim se vrši šifrovanje. Navedenim principima eliminiše se jedan od najvećih problema u simetričnim šifarskim sistemima koji se odnosi na distribuciju tajnih simetričnih ključeva.

Asimetrični šifarski sistemi imaju takve osobine da na jednostavan način obezbeđuju širi skup funkcija bezbednosti u odnosu na simetrične šifarske sisteme. Oni se karakteristično koriste za efikasnu autentifikaciju, integritet poruka, neporocivost i sl., a retko kada se koriste za funkciju poverljivosti. Najčešće se kombinuju sa simetričnim šifarskim sistemima. Na primer, jedna od popularnih primena PKI sistema je uspostavljanje simetričnog ključa za šifrovanje i dešifrovanje u simetričnom šifarskom sistemu.

Javna kriptografija je relativno nova nauka pošto je nastala kao rezultat rada kriptologa kasnih 1960-ih i početkom 1970-tih godina. Primena javne kriptografije je na neki način dovela do revolucije u kriptologiji koja se do tada hiljadama godina oslanjala na simetrične šifarske sisteme. Visoka računarska kompleksnost algoritama u asimetričnoj kriptografiji utiče na performanse u praktičnoj primeni, te se stoga ne preporučuje za šifrovanje velike količine podataka. Međutim, u odnosu na ostale bezbednosne zahteve ova tehnika ima evidentne prednosti u odnosu na simetrične sisteme.

PKI sistemi se zasnivaju na matematičkim jednosmernim funkcijama sa zamkom (eng. *trap door one-way function*). To su funkcije koje se lako računaju u jednom smeru, a teško u drugom (računarski, tj. praktično neizvodljivo). Zamka osigurava da napadač ne može lako (direktno) da izračuna iz javnog ključa privatni ključ. Faktorizacija je relevantan primer za jednosmernu funkciju. Na osnovu dva prosta broja p i q lako je izračunati njihov proizvod $N = pq$, ali je za dovoljno veliki broj N , teško izračunati faktore p i q .

Podsetimo da u simetričnoj kriptografiji, otvoreni tekst se obeležava sa P , a šifrat sa C . U PKI sistemima tradicija je da se otvorena poruka koja se šifruje obeležava sa M . Kada se koriste likovi iz kriptologije, tada i Alisa i Bob treba da imaju svoj par ključeva (javni ključ i odgovarajući privatni). Svako može da šifruje poruku za Boba tako što će koristiti njegov javni ključ. Samo Bob može da dešifruje poruku, s obzirom da samo on ima odgovarajući privatni ključ. Pored navedenog, Bob može da primeni *digitalni potpis* na poruku M tako što će je "šifrovati" svojim privatnim ključem. Svako ko ima pristup Bobovom javnom ključu može verifikovati digitalni potpis njegovim "dešifrovanjem". Navedena tehnika digitalnog potpisivanja je jedna od najkorisnijih funkcionalnosti javne kriptografije. Digitalno potpisivanje elektronskih dokumenata odgovara svojeručnom potpisu na papiru, ali ima i niz drugih prednosti. Digitalni potpis je sastavni deo dokumenta, dok se potpis na papiru može dobiti njegovim kopiranjem sa drugog dokumenta. Verifikaciju potpisa na papiru može da potvrdi samo obučeni stručnjak, dok je verifikacija digitalnog potpisa jednoznačna i lako izvodljiva za svakog ko ima pristup Bobovom javnom ključu.

U nastavku će se razmotriti najpoznatiji algoritam koji se koristi u asimetričnoj kriptografiji, a to je RSA. Dalje, razmatraće se Diffie-Helman šema za uspostavljanje tajnog ključa koja ima značajnu praktičnu primenu. Poseban aspect predstavlja kriptografija na bazi eliptičnih krivi (eng. *Elliptic Curve Cryptography* - ECC) koja na poseban način primenjuje matematiku u javnoj kriptografiji. Osnovna prednost eliptične kriptografije je efikasnost realizacije koja je opredeljuje za primenu u sistemima sa ograničenim resursima (kao što su bežični senzorski sistemi). Za razumevanje navedenih koncepata, neophodno je praktično znanje iz osnova modularne aritmetike.

4.1. ULOGA JAVNOG KLJUČA U RAZMENI SIMETRIČNIH KLJUČEVA

Nakon Drugog svetskog rata, primena računara u kriptologiji bila je privilegija državnih službi. Razvoj nauke i tehnike, doprineo je da računari i njihova primena postanu dostupni širim društvenim slojevima.

- 1947.-tranzistor (*Bell Laboratories*);
- 1951. *Ferranti*, 1953. *IBM* –komercijalni računari;
- 1955.-programski jezik *Fortran*;
- 1959. prva integrisana kola;
- 1969. ARPAnet (preteča interneta) .

1960-tih godina, poslovni svet počinje sve više da koristi računare za šifrovanje (transfer novca, pregovori). U tom momentu, cilj svake poslovne organizacije bio je zaštićena komunikacija izvan kompanije. Jedan od problema je problem standardizacije, a kao rešenje nameće se DES, AES i drugi algoritmi.

Sledeći problem je razmena ključa u simetričnim šifarskim sistemima i pronalaženje sigurnog načina za razmenu simetričnih ključeva između dve udaljene lokacije. Na primer, banka treba da obavi zaštićenu transakciju sa klijentom, ali kako dostaviti ključ? Najbezbednije je da se dostava ključa obavi lično. Kod ovog načina dostavljanja veliki problem predstavlja potrebno vreme za dostavu ključa. Manje bezbedno je angažovanje kurirskih službi u te svrhe. Postavlja se pitanje da li je to nezavisna organizacija i da li je to slaba karika u sistemu?

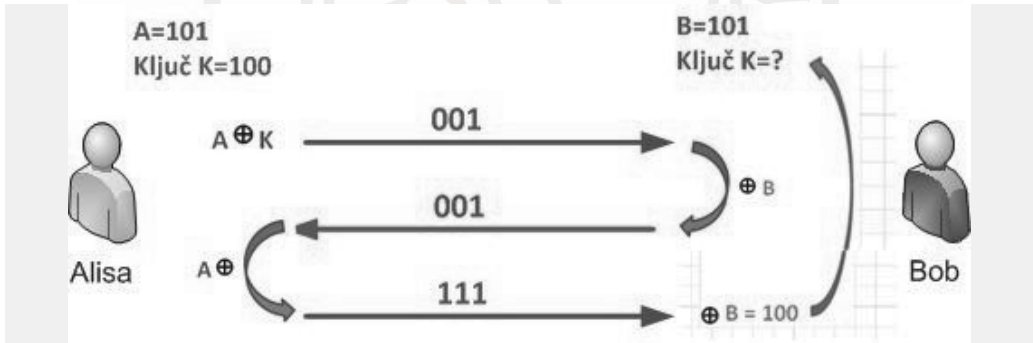
Sličan scenario jeste dostava ključa vojnim jedinicama u ratnim uslovima, dostava ključa nuklearnim podmornicama koje se nalaze na 1000 kilometara od vojne baze. Međutim, država raspolaže novcem, resursima i može da se izbori sa ovakvim problemima, ali za civilni sektor je ovo bio gotovo nerešiv problem.

Razmotrićemo sada jedan klasičan problem. Alisa i Bob žele da imaju sigurnu komunikaciju. Zna se ili pretpostavlja da Trudi prisluškuje njihov komunikacioni kanal. Alisa i Bob mogu povremeno da se lično sastaju i razmenjuju ključeve za naredni period komuniciranja. Ovo rešenje je uglavnom teško izvodljivo u praksi, a može da bude i nemoguće.

Na osnovu ovakvih činjenica, postavlja se pitanje da li je moguće ostvariti sigurnu komunikaciju bez uobičajene razmene ključa? Ili, da li je moguće tajno komunicirati bez razmene simetričnog ključa? Odgovor na ovo pitanje indukovao bi rešenje i podigao sigurnost na viši nivo. Napomenućemo da kriptografski sistem nije bezbedan ukoliko se ključ prenosi preko komunikacionog kanala, makar bio i u formi šifrata.

Novo nastala ideja je šifrovanje bez razmene ključa. To bi značilo da sada Alisa i Bob mogu da koriste isti komunikacioni kanal za dogovor o simetričnom ključu, razmenjujući parametre na osnovu kojih Trudi ne može da izračuna tajni simetrični ključ, dok Alisa i Bob na osnovu istih parametara mogu da izračunaju ključ.

Na primer, Alisa ostavlja poruku u metalni sandučić koji zaključava katanacem A i šalje ga Bobu. Nakon prijema, Bob stavlja svoj katanac B i vraća pošiljku sa dva katanca Alisi. Alisa skida svoj katanac A i vraća pošiljku Bobu na kojoj je još uvek katanac B . Bob sada može da otvori pošiljku, dok napadač nije mogao to da uradi ni u jednom trenutku njihove komunikacije što ne znači da ne postoje i bezbednosni problemi druge vrste u ovoj komunikaciji između Alise i Boba.



Slika 4.1 Razmena ključa javnim komunikacionim kanalom

Slika 4.1, prikazana je razmena simetričnog ključa između Alise i Boba. U prvoj fazi scenarija, Alisa i Bob treba da generišu po jedan tajni (privatni) parametar A i B čije vrednosti samo oni znaju. Alisa pored tajnog parametra generiše i simetrični ključ koji će koristiti za šifrovanje u nekoj budućoj komunikaciji sa Bobom. Kao što je prikazano u scenariju, nakon tri runde razgovora Bob dobija identičan ključ K koji je generisala

Alisa. Ako Trudi bude prisluškivala komunikacioni kanal, neće biti u prilici da sazna neku informaciju o ključu K .

Uprkos opšte prihvaćenom mišljenju da je ovaj problem nerešiv, jedna grupa entuzijasta je krajem 70-tih godina prošlog veka ponudila rešenje. Istraživanja u ovom pravcu su ubrzo dovela do razvoja šifarskih sistema sa javnim ključem. To je bila revolucija u kriptografiji 20-tog veka i najveće otkriće od pojave monoalfabetske šifre.

4.2. DIFI-HELMAN (*DIFFIE-HELLMAN*)

Difi-Helman protokol predložili su Vitfild Difi (*Whitfield Diffie*) i Martin Helman (*Martin Hellman*)⁶. Difi i Helman su tražili matematičke funkcije za koje redosled šifrovanja i dešifrovanja nije bitan, na primer:

$$f(g(x)) = g(f(x))$$

Ovakve funkcije postoje, većina ih je dvosmerna, odnosno mogu se lako izračunati ali je lako naći i njihovu inverznu vrednost. Primer dvosmernih funkcija:

$$f(x) = 2x; f(x) = x^2$$

Primer za ovakvu funkciju je uključivanje ili isključivanje prekidača. Međutim, ovakva vrsta funkcija nije poželjna u kriptografiji. Od značaja su jednosmerne funkcije (*one way*), tačnije neki oblici ovih funkcija.

Jednosmerne funkcije relativno lako mogu da se izračunaju, ali njihova inverzna vrednost može da se odredi samo izuzetno složenim postupkom. Za dato x lako se računa $f(x)$, ali je za dato $f(x)$ teško izračunati x ili je potrebno ogromno vreme

⁶ Vitfild Difi rođen je 1944. godine u Njujorku. 1965. diplomirao na Stanford Univerzitetu. Dostupno više na Veb sajtu: <http://en.wikipedia.org/wiki/Whitfield>.

Martin Helman rođen je 1945. godine, u Bronksu. Doktorirao na Stanford Univerzitetu. Dostupno više na sajtu: <http://en.wikipedia.org/wiki/MartinHellman>.

uz neograničene resurse. Dobri primeri za jednosmernost su lomljenje čaše ili mešanje boje. Njihov značaj je u tome što poruka šifrovana jednosmernom funkcijom ne može da se dešifruje. Posebno su od značaja za kriptografiju sa javnim ključem jednosmerne funkcije sa zamkom (eng. *trapdoor one way function*).

Jednosmerne funkcije sa zamkom su poseban oblik jednosmernih funkcija. Lako ih je izračunati u jednom (direktnom) smeru, ali teško im je izračunati inverznu vrednost. Ako je poznata tajna vrednost (zamka), onda se lako može izračunati i direktna i inverzna vrednost.

Na primer, za dato x lako je izračunati $f(x)$, teško je izračunati x iz $f(x)$, ali ako je poznata tajna vrednost, iz $f(x)$ lako se ponovo računa x . Modularna aritmetika podrazumeva prisustvo velikog broja takvih jednosmernih funkcija.

Pod pojmom „teško“ podrazumevaju se problemi koji se ne mogu rešiti u prihvatljivom vremenskom periodu iako su na raspolaganju najbolji poznati algoritmi i najbolja tehnologija.

Osnovni problem ovakvih funkcija je što nije dokazano da postoje jednosmerne funkcije i jednosmerne funkcije sa zamkom, strogo matematički gledano. Uprkos ovoj tvrdnji, postoje dve funkcije koje se smatraju kandidatima za funkcije sa pomenutim osobinama:

- Diskretni eksponent čija je inverzna funkcija diskretni logaritam,
- Proizvod celih brojeva čija je inverzna funkcija faktorizacija dobijenog broja.

Navedene dve funkcije su lake za izračunavanje, dok se veruje da to nije slučaj sa njihovim inverznim funkcijama.

Skraćeniciu „DH“ uglavnom ćemo koristiti za Difi-Helman algoritam ili protokol za razmenu ključa. Protokol je razvijen nezavisno na dva mesta⁷. Predstavlja algoritam za razmenu zajedničkih simetričnih ključeva. Nije namenjen za šifrovanje ili digitalno

⁷ *Government Communications Headquarters – GCHQ*: Džems Elis, Kliford Koks, Malkom Vilijamson. Stanford Univerzitet: Difi i Helman

potpisivanje. Sigurnost DH algoritma zasniva se na računski složenom računanju (jednosmerne funkcije) diskretnog algoritma za zadate g, p i $g^n \bmod p$ naći n . Za poznato g i x , gde je $x = g^n$, može da se odredi n :

$$n = \log_g(x)$$

Ako je $x = g^n \bmod p$, n se takođe određuje preko logaritma ali diskretnog (diskretni algoritam). U nastavku ćemo proći kroz matematičke osnove DH algoritma. Neka je p veliki prost broj i g takvo da se za svako $x \in \{1, 2, \dots, p-1\}$ može naći n tako da je:

$$x = g^n \bmod p$$

Vrednosti p i g su javne. U komunikaciji između Alise i Boba, ove parametre može da odredi bilo ko od njih, a zatim se parametri razmenjuju javnim kanalom. Alisa bira svoju tajnu vrednost a . Bob bira svoju tajnu vrednost b . Ove vrednosti predstavljaju velike slučajne proste brojeve.

U prvoj rundi, Alisa javno šalje vrednost $g^a \bmod p$ Bobu. Bob javno šalje vrednost $g^b \bmod p$ Alisi. U drugoj rundi, oboje na osnovu primljenih vrednosti računaju zajedničku tajnu vrednost $g^{ab} \bmod p$. Dobijena zajednička tajna vrednost može da se koristi kao simetrični ključ.

Pretpostavimo da Alisa i Bob koriste vrednost $g^{ab} \bmod p$ kao simetrični ključ. Trudi može da sazna vrednosti $g^a \bmod p$ i $g^b \bmod p$ jer su poslate javno, kanalom veze. Međutim, Trudi ne može na osnovu ovih vrednosti da sazna vrednost a ili b , a ako bi to mogla da uradi, sistem bi bio razbijen, a to bi značilo da je rešila problem diskretnog algoritma. Standardom PKCS #3⁸, definisane su i preporučene vrednosti

⁸ PKCS #3: Diffie-Hellman Key Agreement Standard.

Dostupno na Veb sajtu: <http://www.rsa.com/rsalabs/node.asp?id=2126>

parametara a, b, g i p za generisanje i razmenu zajedničkih simetričnih ključeva. U praksi se za p vrednost koristi veliki broj koji je veći od 1024 bita.

U primeru ispod, prikazaćemo jednu realizaciju DH protokola.

29. PRIMER:

Izbor javnih parametara p i g :

$$p = 33, g = 3$$

Izbor tajnih parametara a i b :

$$a = 11, b = 9$$

Alisa šalje vrednost A_v Bobu:

$$A_v = g^a \bmod p = 3^{11} \bmod 33 = 3$$

Bob šalje vrednost B_v Alisi:

$$B_v = g^b \bmod p = 3^9 \bmod 33 = 15$$

Alisa računa ključ K_{Alisa} :

$$K_{Alisa} = B_v^a \bmod p = 15^{11} \bmod 33 = 15$$

Bob računa ključ K_{Bob} :

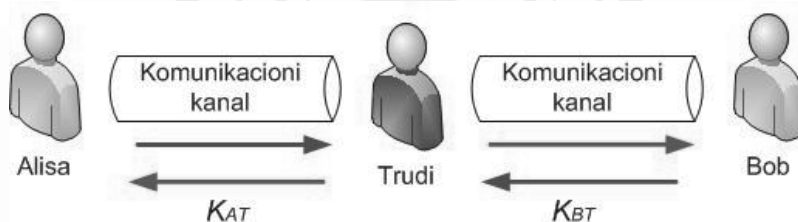
$$K_{Bob} = A_v^b \bmod p = 3^9 \bmod 33 = 15$$

Rezultat K_{AB} :

$$K_{Alisa} == K_{Bob} = 15$$

U ovom primeru radi lakšeg razumevanja DH protokola, koristili smo male vrednosti za parametre algoritma.

Međutim, postoje ozbiljni problem i kod implementacije DH algoritma. DH algoritam je osetljiv na napad tipa „Čovek u sredini“ (eng. *Man-in-the-middle*). Ovaj napad može da bude aktivan ili pasivan. Ukoliko se radi o aktivnom napadu, on bi mogao da presretne poruku, izmeni je i prosledi dalje, dok je kod pasivnog napadača moguće jedino prisluškivanje komunikacionog kanala. Pretpostavlja se da Trudi može da ima ulogu aktivnog napadača u scenariju „Čovek u sredini“. U ovom slučaju DH protokol bi bio nebezbedan jer bi Trudi delila jedan simetrični ključ K_{AT} sa Alisom i jedan simetrični ključ K_{TB} sa Bobom i tako postala posrednik u komunikaciji.



Slika 4.2 DH i „Čovek u sredini“ – aktívni napad

Da bi se izbegli ovakvi problemi ili sprečio dobro poznati napad „Čovek u sredini“, **Slika 4.2**, koji je uvek prisutan u bilo kom scenariju komunikacije, potrebno je obezbediti mehanizam autentifikacije. Ovaj mehanizam bi podrazumevao obostranu autentifikaciju što znači da će Alisa i Bob biti sigurni u integritet razmenjenih poruka. Zaštita integriteta poruka podrazumeva da poruke u komunikacionim kanalima nisu izmenjene od neautentifikovane strane. Bitno je imati svest o postojanju mogućnosti ovakvog napada, ne samo u DH protokolu već i u drugim protokolima.

Asimetrični šifarski sistemi

Difi i Helman su prvi predložili primenu asimetričnog šifarskog sistema. Za šifrovanje i dešifrovanje koristio bi se različit ključ.

Na primer, Alisa ima jedan javni ključ koji je svima dostupan i koristi se za šifrovanje poruke. Bob je šifrovao poruku sa Alisinim javnim ključem i poslao Alisi, a samo ona

ima tajni ključ koji je neophodan za dešifrovanje poruke. Ideja za ovakav šifarski sistem je njihova, ali oni nisu predložili funkciju koja bi radila na ovaj način.

Odlike kriptografije sa javnim ključevima:

- zasnovana je na matematičkim funkcijama umesto na postupcima zamene ili transpozicije;
- koristi različite ključeve za šifrovanje i dešifrovanje;
- jedan ključ je privatni (tajan), drugi može da bude potpuno javan;
- privatni i javni ključ su povezani odgovarajućim matematičkim relacijama;
- ima lošije osobine od kriptografije sa simetričnim ključevima;
- do 1000 puta sporija od simetričnih sistema;
- može da se koristi za šifrovanje, autentifikaciju, digitalni potpis i razmenu simetričnih ključeva;
- nisu pogodni za šifrovanje veće količine podataka.

Na osnovu navedenih odlika, predložena su tri načina upotrebe kriptografije sa javnim ključem:

- Šifrovanje i dešifrovanje (poverljivost);
- Digitalni potpis (autentifikacija, ..);
- Razmena simetričnog ključa.

4.3. RSA

Najpoznatiji algoritam iz asimetrične kriptografije, RSA, dobio je naziv po početnim slovima njegovih autora (Rivest, Šamir i Adleman)⁹. Zasniva se na matematički složenom postupku faktorizacije (rastavljanja) proizvoda dva velika prosta broja na njegove faktore (činioce). Problem kod ovog pristupa je u tome što ne postoji čvrsti matematički dokaz da ne postoji kraći put za faktorizaciju.

⁹RSA je skraćenica od početnih slova naučnika Ron Rivest, Shamir Adi i Leonard Adleman, koji su prvi javno opisali novi algoritam 1977. godine. Clifford Cocks, engleski matematičar, razvio je ekvivalentan sistem 1973. godine, ali je on bio klasifikovan najvećim stepenom tajnosti sve do 1997. godine.

Za šifrovanje kod asimetričnih sistema se koristi jednosmerna funkcija sa zamkom. Suština je u pronalaženju funkcije E koja menja otvorenu poruku M u šifrat C , tj. $C = E(M, K_e)$. Parameter K_e se naziva javni ključ i služi isključivo za šifrovanje. Ne može se upotrebiti za dešifrovanje, tj. $M \neq D(C, K_e)$ zbog toga što je E jednosmerna funkcija. Ukoliko je poruka namenjena za Boba, svako može šifrovati poruku za njega korišćenjem njegovog javnog ključa K_e . Za dešifrovanje Bob mora da upotrebi tajnu vrednost K_d (privatni ključ) i da primeni inverznu funkciju $M = D(C, K_d)$ kako bi se iz šifrata C dobila otvorena poruka M . Uslov za uspešno šifrovanje i dešifrovanje je $M = D(E(M, K_e), K_d)$. Jasno je da je neophodna pogodna matematička funkcija koja ima navedena svojstva, a pripada klasi jednosmernih funkcija sa zamkom. U navedenom primeru privatni ključ K_d je zamka. Poznavanjem zamke proces dešifrovanja je jednostavan. Bez zamke, dešifrovanje je praktično nemoguće. Funkcija koja zadovoljava navedene pretpostavke je oblika:

$$f(x) = x^e \bmod N.$$

Uz odgovarajući izbor vrednosti e i N ova funkcija je jednosmerna. Uz poznavanje tajne vrednosti (zamke) može da se nađe njena inverzna funkcija. Bez poznavanja zamke, ona je praktično nerešiva.

Da bi se generisao RSA javni i privatni par ključeva, izaberu se dva velika prosta broja p i q i formira se njihov proizvod $N = p \cdot q$. Zatim se izabere broj e , uzajamno prost sa proizvodom $\varphi(N) = (p - 1)(q - 1)$ i uz uslov $e < \varphi(N)$. Konačno, potrebno je odrediti d na osnovu sledećeg $d = e^{-1}(\bmod \varphi(N))$, tj. zadovoljeno je sledeće $ed = 1(\bmod \varphi(N))$. Nadalje, faktori p i q se ne razmatraju.

Broj N je modul, broj e je eksponent za šifrovanje, a d je eksponent za dešifrovanje.

RSA par ključeva se sastoji iz delova:

Javni ključ: (N, e) ,

Privatni ključ: d .

Za operacije šifrovanja i dešifrovanja primenjuje se funkcija modularnog eksponenta na sledeći način:

$$\text{Šifrovanje: } C = M^e \bmod N,$$

$$\text{Dešifrovanje: } M = C^d \bmod N.$$

Tajnost asimetričnog šifarskog sistema sa RSA algoritmom zasniva se na složenosti faktORIZACIJE broja N . Nakon što bi se izvršila faktORIZACIJA broja N (koji je javan) na činioce p i q , bilo bi lako, korićenjem javno dostupne vrednosti e odrediti privatnu vrednost d iz uslova $ed = 1 \pmod{\varphi(N)}$. Za dovoljno velike brojeve p i q , a samim tim i N , proces faktORIZACIJE je dugotrajan posao. Uz dostupne računarske resurse može se uporediti sa procesom totalne pretrage po prostoru svih ključeva kod simetričnih šifarskih sistema (totalna pretraga). Ono što je problem kod asimetričnih šifarskih sistema je nepostojanje čvrstog matematičkog dokaza da li je rastavljanje na faktore jedini način za razbijanje RSA.

Da li RSA zaista radi? S obzirom da je $C = M^e \bmod N$, potrebno je pokazati da je:

$$M = C^d \bmod N = M^{ed} \bmod N$$

30. PRIMER:

RSA generisanja tajnog ključa. Izabrati velike proste brojeve p i q :

$$p = 11, q = 3$$

Odrediti vrednost N i $\varphi(N)$:

$$N = pq = 11 * 3 = 33$$

$$\varphi(N) = (p-1)(q-1) = (11-1)(3-1) = 20$$

Izabrati broj e (uzajamno prost):

$$e = 3$$

Pronaći d takvo da je $ed = 1 \pmod{20}$:

$$d = e^{-1} \pmod{\varphi(N)} = 3^{-1} \bmod 20 = 7$$

Rezultat je javni (N, e) i privatni ključ d :

$$(N, e) = (33, 3)$$

$$d = 7$$

Napomena:

Nakon izračunatog ključa d , parametri p i q se uništavaju na siguran način, jer sigurnost zavisi baš od ovih parametara.

31. PRIMER:

RSA šifrovanje i dešifrovanje

Javni ključ (N, e) i privatni ključ d :

$$(N, e) = (33, 3)$$

$$d = 7$$

Izabrana poruka M :

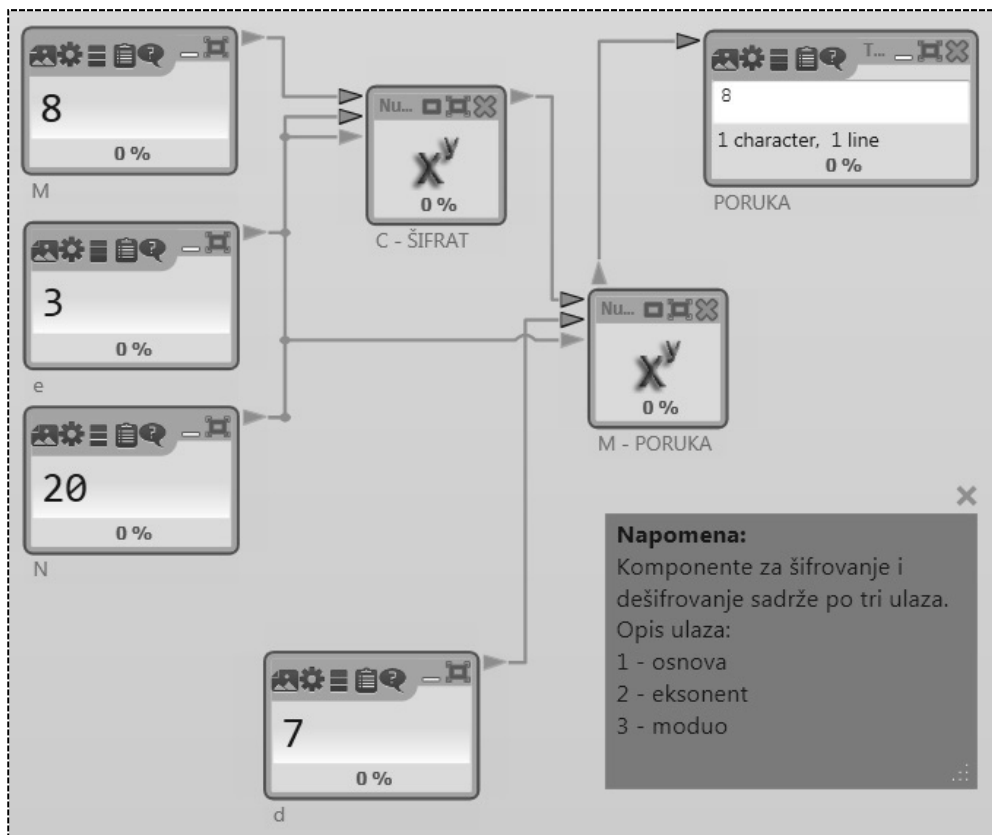
$$M = 8$$

Šifrovanje, generisanje šifrata C :

$$C = M^e \pmod{N} = 8^3 \pmod{20} = 12$$

Dešifrovanje šifrata C :

$$M = C^d \pmod{N} = 12^7 \pmod{20} = 8$$

32. PRIMER:**Slika 4.3.** Jedna realizacija šifrovanja i dešifrovanja sa RSA algoritmom u Cryptool-u**Slika 4.3** Jedna realizacija šifrovanja i dešifrovanja sa RSA algoritmom u Cryptool-u

Postupak šifrovanja i dešifrovanja obuhvata celobrojne operacije sa porukom M . Pre šifrovanja, karaktere poruke M prvo treba pretvoriti u broj. U praksi se za pretvaranje karaktera u decimalnu vrednost koristi ASCII¹⁰ tabela.

¹⁰ ASCII je skraćenica za Američki standardni kod za razmenu informacija. Računari mogu razumeti saobrojeve, tako da ASCII kod je brojčana reprezentacija karaktera, kao što su 'A' ili '@'. Dostupno na Veb sajtu: <http://www.asciitable.com/>

Na primer, poruku „UNIVERZITET SINGIDUNUM“ kada pretvorimo preko ASCII tabele, dobijamo niz decimalnih vrednosti {85, 78, 73, 86, 69, 82, 90, 73, 84, 69, 84 83, 78, 73, 78, 71, 68, 85, 78, 85, 77}. **Slika 4.4**, prikazan je jedan fragment ASCII tabele.

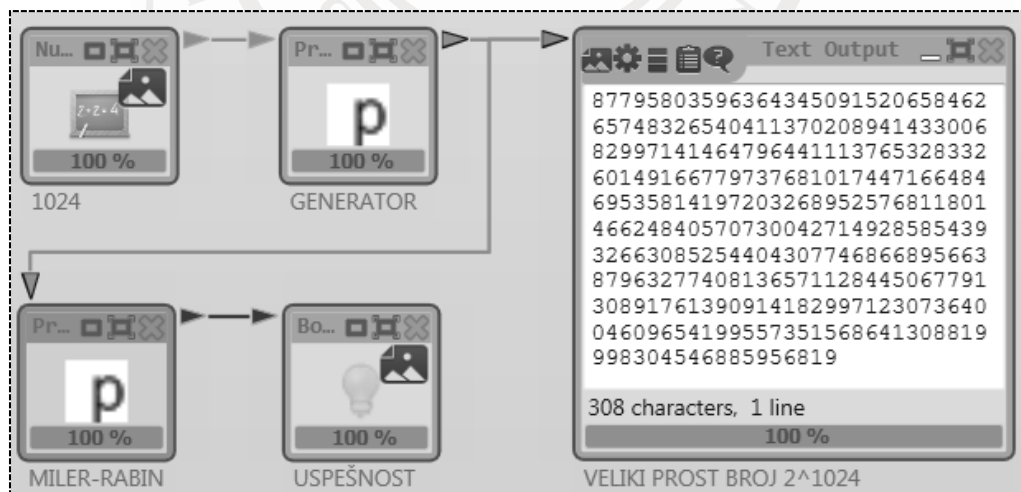
Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
64	40	100	@	@	96	60	140	`	`
65	41	101	A	A	97	61	141	a	a
66	42	102	B	B	98	62	142	b	b
67	43	103	C	C	99	63	143	c	c
68	44	104	D	D	100	64	144	d	d
69	45	105	E	E	101	65	145	e	e
70	46	106	F	F	102	66	146	f	f
71	47	107	G	G	103	67	147	g	g
72	48	110	H	H	104	68	150	h	h
73	49	111	I	I	105	69	151	i	i

Slika 4.4 Jedan deo ASCII tabele

Računa se po modulu N . Potrebno je, da je $N > M$, da bi proces šifrovanja bio jednoznačan. Ukoliko se desi i pored izbora velikog N , $M > N$, onda poruka M treba da se rastavi na manje celine ili blokove.

33. PRIMER:

Slika 4.5, Generisanje velikog prostog broja (2^{1024}) i primena Miller-Rabinovog testom.



Slika 4.5 Generisanje velikog prostog broja i test na prostost

Postoji problem pronalaženja velikih prostih brojeva. Za $N = pq$, p i q su veliki prosti brojevi, N treba da je dovoljno veliko. Prostih brojeva ima beskonačno, što znači da postoji dovoljno prostih brojeva za ovakve potrebe. Koriste se različiti algoritmi za pronalaženje prostih brojeva od kojih su najefikasniji: Miler-Rabinov test¹¹, APR test i dr.

Pored bitnosti načina za izbor vrednosti N , postoji i problem izbora javnog ključa e . Javni ključ e mora da bude uzajamno prost sa $(p-1)(q-1)$, pa se za e često bira prost broj. Zbog brzinskih performansi, postoji zahtev da e bude što manje, ali ta odluka može da naruši snagu algoritma. Mnogi korisnici koriste isti eksponent e unutar javnih ključeva. Obično je to vrednost $e = 65537$. Na ovaj način se ne kompromituje šifarski sistem, a omogućava se da proces šifrovanja bude značajno brži od dešifrovanja.

Još jedan problem je računanje tajnog ključa d . Kao što smo već videli, računanje privatnog ključa d , svodi se na rešavanje jednačine:

$$ed = 1(\text{mod } \phi(N))$$

Obzirom da je e izabrano takvo da je:

$$\text{gcd} = (e, \phi(N)) = 1,$$

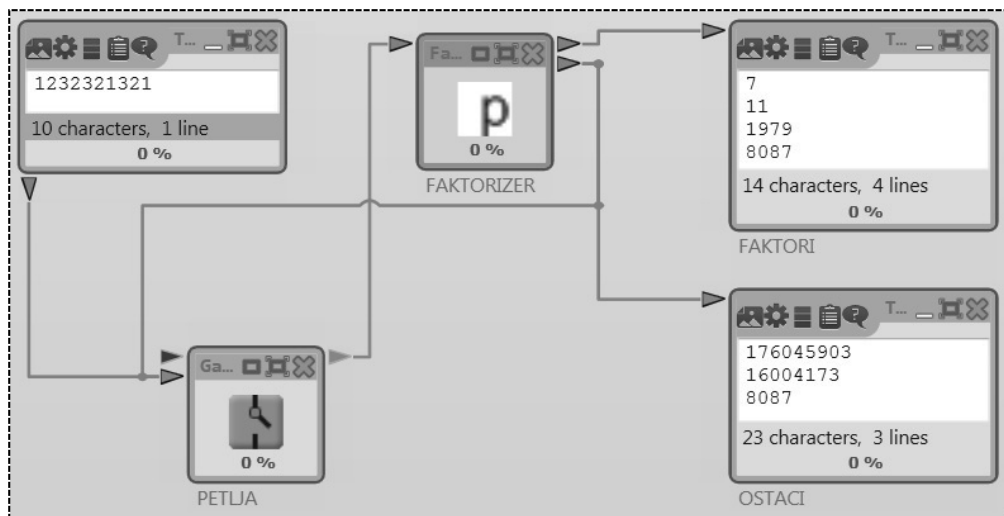
Ova jednačina uvek može da se reši. Jedna od mogućnosti za rešavanje ove jednačine je pomoću proširenog Euklidskog algoritma.

¹¹ Miller–Rabin primality test.

Dostupno na Veb sajtu: http://en.wikipedia.org/wiki/Miller–Rabin_primality_test

34. PRIMER:

Slika 4.6, upotreba metoda Kvadratno Sito (eng. Quadratic Sieve)



Slika 4.6 Primer faktorizacije manjih prostih brojeva

Što se tiče bezbednosti RSA šifarskog algoritma, vrednosti C , e i N su za svakoga javne. Postavlja se pitanje, može li Trudi da rekonstruiše poruku M ? Trenutno ne postoji dokaz da Trudi može efikasno da rekonstruiše poruku M na osnovu poznavanja šifrata C , javnih ključeva e i N , a da pri tome ne zna $\phi(N)$.

Dokazano je da problem pronalaženja $\phi(N)$ podjednako složen kao i faktorizacija broja N . Veruje se, mada nije dokazano, da je problem faktorizacije velikih brojeva praktično nerešiv.

RSA Laboratories¹² je objavila takmičenje u kategoriji faktorizacije velikih prostih brojeva koji se koriste u javnim kriptosistemima. Takmičenje je završeno 2007. godine i tada su postignuti najbolji rezultati u ovoj kategoriji. Poslednji problem je bio

¹² Dostupno na Veb sajtu: <http://www.rsa.com/rsalabs/node.asp?id=2093>

faktorizacija N dužine 768 dekadskih cifara. Nakon toga nije bilo novih rezultata i smatra se da je to trenutno najbolji rezultat koji je moguće danas postići. Potrebno vreme je bilo nešto malo manje od 3 kalendarske godine.

Povećanje granice bezbednosti zahteva povećanje dužine ključa. Osnovni razlog je konstantno unapređenje algoritama za faktorizaciju na polju kriptanalize. Sa druge strane, vreme potrebno za šifrovanje i dešifrovanje je proporcionalno trećem stepenu dužine ključa. Odluka da se poveća granica bezbednosti povećanjem ključa loše utiče na brzinske performanse RSA algoritma. Osnovna primena RSA je za šifrovanje kriptografskih ključeva i digitalni potpis. Simetrični šifarski sistemi se koriste za zaštitu podataka.

Merenjem performansi RSA algoritma, utvrđeno je da je RSA algoritam 1500 puta sporiji od DES algoritma. Deo algoritma koji služi za računanje eksponenta i modula u najvećoj meri doprinosi ovom lošem rezultatu. Zatim, generisanje brojeva koji se koriste u RSA algoritmu takođe zahtevaju neko vreme, kao i testiranje izabrane vrednosti N u odnosu na poznate metode za faktorizaciju.

Na osnovu određenih saznanja i takmičenja u kojima su najbolji bili nagrađeni pozamašnim sumama novca, preporučene dužine za N su $N > 1000$ bitova (1024, 2048, ...).

4.4. PRIMENA ASIMETRIČNIH ŠIFARSKIH SISTEMA

Primenom kriptografije sa javnim ključem može da se postigne poverljivost. Poverljivost podrazumeva siguran prenos podataka preko nebezbednih veza, kao i skladištenje podataka na nebezbednim medijima. Zatim, autentifikaciju i digitalni potpis koji obezbeđuju integritet i neporecivost. Bitno je napomenuti da nema sistema neporecivosti u sistemima sa simetričnim ključevima.

4.4.1. RSA ŠIFROVANJE I DEŠIFROVANJE

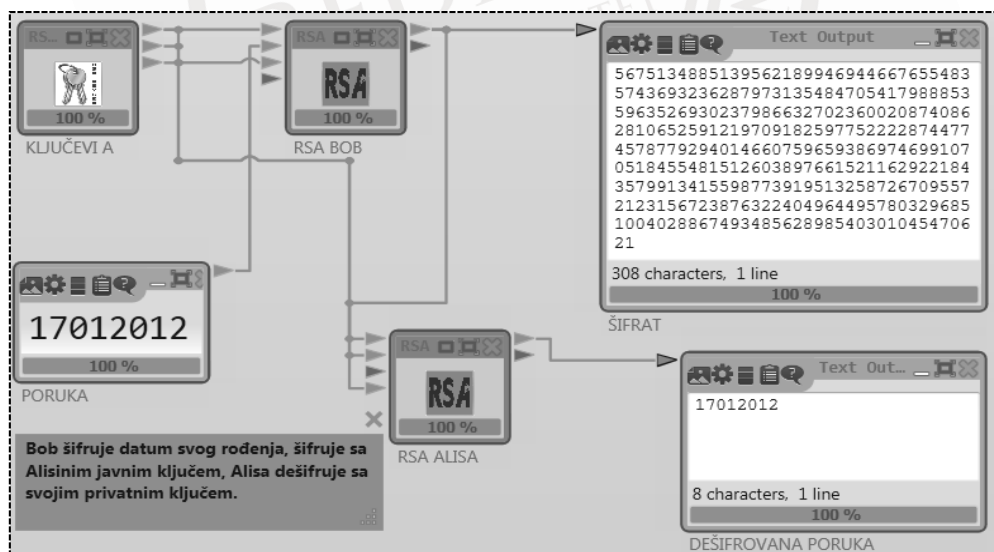
Javni ključ označićemo sa (N, e) , a privatni sa d . Ukoliko poruku označimo sa M , šifrat je C , iz toga sledi da je:

$$C = M^e \bmod N,$$

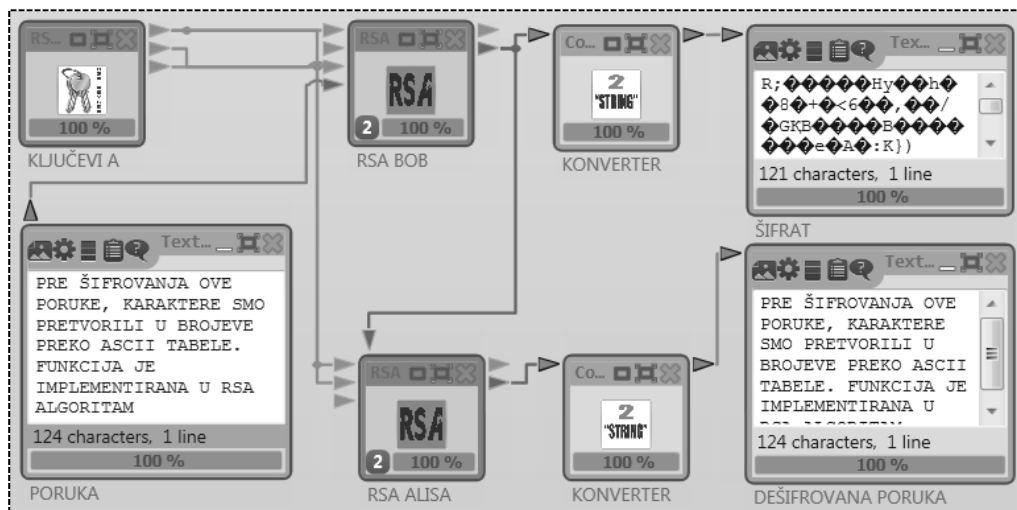
$$M = C^d \bmod N$$

Prethodno u RSA poglavlju detaljno smo objasnili operaciju šifrovanja sa RSA algoritmom. Bitno je napomenuti da sa RSA algoritmom možemo da šifrujemo dva tipa ulaznih podataka. Prvi tip je tip podataka kao celobrojna vrednost za uslovom da je $N > M$, a drugi tip podatka predstavlja neki niz karaktera ili tekstualne poruke. Ukoliko se radi o tekstualnim porukama, neophodno je karaktere poruke pretvoriti u decimalne vrednosti preko ASCII tabele. Obe realizacije pokazaćemo u sledećim Cryptool projektima.

35. PRIMER:



Slika 4.7 Šifrovanje celobrojnih vrednosti sa RSA

36. PRIMER:

Slika 4.8 Šifrovanje i dešifrovanje tekstualnih poruka sa RSA

4.4.2. RSA DIGITALNO POTPISIVANJE

Kod asimetričnih šifarskih sistema javni ključ je (N, e) , a privatni d . Ukoliko poruku označimo sa M , digitalni potpis sa S , onda je:

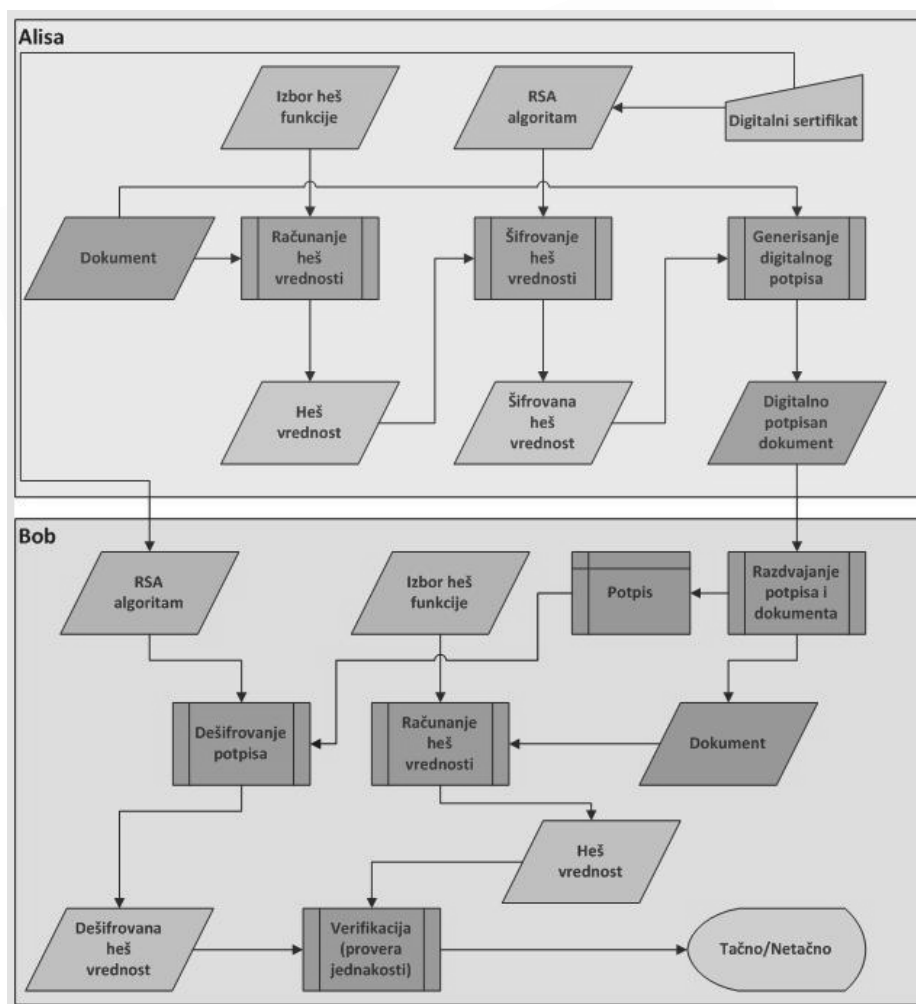
$$S = M^d \pmod{N}$$

Kod RSA algoritma, dešifrovanje i potpisivanje su iste operacije. Za računanje digitalnog potpisa S neophodno je poznavanje privatnog ključa d . Potvrda ispravnosti ili verifikacija digitalnog potpisa na poruci M može da se uradi na sledeći način:

$$S^e \pmod{N} = (M^d)^e \pmod{N} = M$$

Možemo primetiti da je operacija ista kao i kod šifrovanja. Svako ko zna (N, e) može da potvrdi ispravnost digitalnog potpisa.

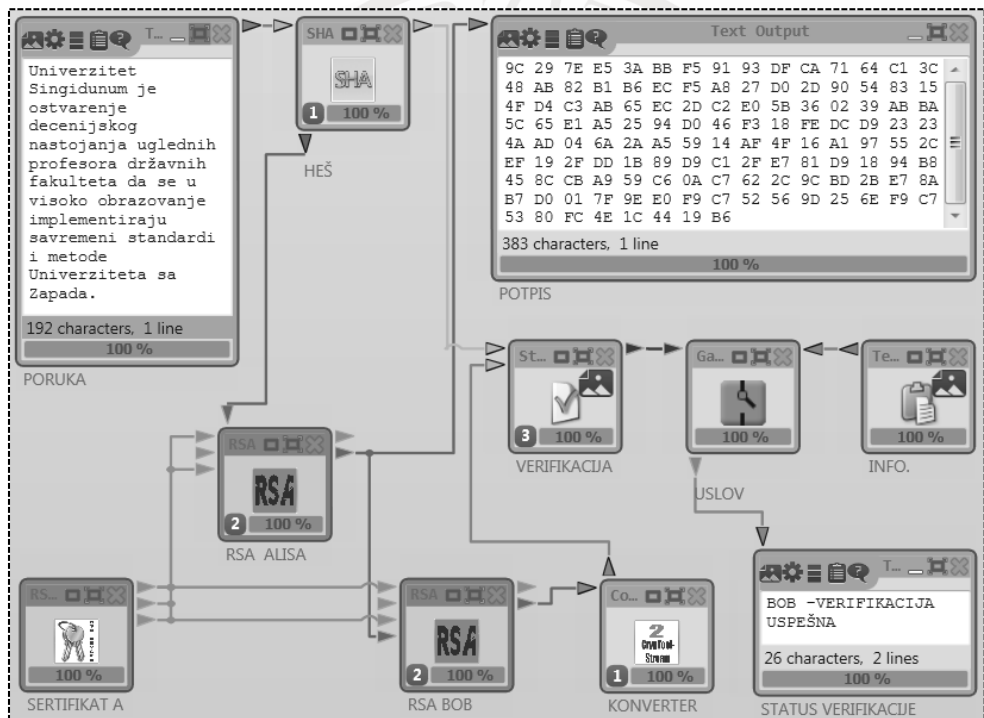
Na primer, Alisa želi da pošalje digitalno potpisanu poruku M Bobu. U ovom momentu Alisi i Bobu nije bitna poverljivost. To znači da Trudi može da pročita poruku M , ali ne može da je promeni zbog postojanja digitalnog potpisa S poruke M . Trudi je samo pasivan napadač (može da prisluškuje). Vlasnik digitalnog potpisa u ovom scenariju je Alisa. Na drugoj strani, Bob prima poruku M i digitalni potpis S , a zatim koristi javni ključ (N, e) da bi uspešno verifikovao digitalni potpis i bio siguran da je primljeni sadržaj poruke M načinila Alisa.



Slika 4.9 Digitalni potpis i verifikacija

Slika 4.9, prikazana je jedna standardna primene digitalnog potpisa. U scenariju, Alisa potpisuje poruku i šalje je Bobu. Bob prima digitalno potpisanu poruku i započinje proceduru verifikacije digitalnog potpisa da bi se uverio u identitet pošiljaoca Alise. Prikazane su sve komponente koje su neophodne za digitalno potpisivanje i verifikaciju digitalnog potpisa za izabrani dokument, kao i sam redosled izvršavanja svih nezaobilaznih procedura.

37. PRIMER:



Slika 4.10 Digitalno potpisivanje i verifikacija potpisa

Slika 4.10, jedna realna realizacija digitalnog potpisa sa pravim algoritmima i vrednostima.

4.4.3. TAJNOST I NEPORECIVOST

Neporecivost je svojstvo koje sprečava entitet da porekne postojaće obaveze ili izvršene akcije. Neporecivost osigurava da ugovori, transakcije i druge izvršene akcije preko Interneta ne mogu biti kasnije opovrgnute od strane bilo koje uključene stranke. Neporecivost znači da se može potvrditi da su pošiljalac i primaoc zaista entiteti koji su poslali, odnosno primili poruku.

Neporecivost nije moguće obezbediti u sistemu sa simetričnim ključem. Pokazaćemo to na sledećem primeru.

Alisa izdaje nalog za kupovinu 100 akcija svom brokeru Bobu. Alisa izračuna MAC primenom simetričnog ključa K_{AB} . Na ovaj način obezbeđen je servis integriteta. Međutim, vrednost akcija smanjila se za 80%, Alisa tvrdi da nije izdala nalog za kupovinu akcija. Postavlja se pitanje, da li može Bob da dokaže da je Alisa izdala nalog za kupovinu akcija? Odgovor je NE iz razloga što Bob zna simetrični ključ K_{AB} i on je mogao sam da napiše poruku. Sada možemo da zaključimo da Bob ne može da dokaže da je Alisa izdala nalog za kupovinu.

Sada ćemo u sledećem primeru da vidimo kako izgleda ovaj isti scenario samo sa primenom sistema sa javnim ključevima.

Alisa izdaje nalog za kupovinu 100 akcija svom brokeru Bobu. Alisa digitalno potpisuje nalog svojim privatnim ključem. U ovom momentu obezbeđen je servis integriteta. Vrednost akcija se smanjuje za 80%, Alisa tvrdi da nije izdala nalog za kupovinu. Postavlja se pitanje, da li može Bob da dokaže da je Alisa izdala nalog za kupovinu akcija? Odgovor je DA, iz razloga što samo neko ko poseduje Alisin privatni ključ je mogao da digitalno potpiše nalog. Svakako, podrazumeva se da Alisin privatni ključ nije ukraden.

Pre nego što napravimo vezu između tajnosti i neporecivosti, upoznaćemo se sa notacijom operacija u sistemu javnih ključeva, radi lakšeg razumevanja.

Notacija za tajnost:

Alisa šalje Bobu šifrovane poruke sa Bobovim javnim ključem (N, e) :

$$C = \{M\}_{Bob}$$

Bob dešifruje poruku koju je primio od Alise sa svojim privatnim ključem d :

$$M = \{C\}_{Bob}$$

Notacija za neporecivost:

Alisa šalje Bobu digitalno potpisanu poruku sa privatnim ključem d :

$$S = [M]_{Alisa}$$

Bob prima digitalnu poruku i verifikuje potpis sa Alisinim javnim ključem (N, e) :

$$M = [S]_{Alisa}$$

U poslednjem slučaju S je digitalno potpisana poruka i formalno se poklapa sa dešifrovanjem. Iz toga sledi:

$$\{[M]_{Alisa}\}_{Alisa} = M = [\{M\}_{Alisa}]_{Alisa}$$

Sada pretpostavimo da je potrebno da se istovremeno ostvare servisi tajnosti i neporecivosti. Postavlja se pitanje da li sistemi sa javnim ključevima obezbeđuju oba zahteva? Pre nego što damo odgovor, imajmo na umu da postoje dva moguća scenarija. U prvom Alisa prvo potpiše poruku, a zatim šifruje što odgovara slučaju $\{[M]_{Alisa}\}_{Bob}$, i drugi scenario u kom Alisa prvo šifruje poruku, a zatim potpisuje što odgovara slučaju $[\{M\}_{Bob}]_{Alisa}$.

Ne treba zaboraviti da je javni ključ svima dostupan. Svako može da izračuna $\{M\}_{Alisa}$. Privatni ključ je tajan. Samo Alisa može da izračuna $[C]_{Alisa}$ ili $[M]_{Alisa}$ i da digitalno potpiše poruku svojim privatnim ključem, ali svi mogu da provere ispravnost potpisa ukoliko znaju javni ključ. Sada možemo da kažemo da redosled ovih operacija nije toliko bitan za bezbednost sistema.

4.4.4. INFRASTRUKTURA JAVNIH KLJUČEVA

U ovom delu poglavlja, diskutovaćemo o sertifikatima javnih ključeva i sertifikacionim telima. Digitalni sertifikat (eng. *certificate*) je potpisan dokument koji omogućava korisnicima i organizacijama proveru identiteta učesnika u komunikaciji. Sertifikat se digitalno potpisuje da bi se utvrdila veza između korisnika i javnog ključa. Javni ključ, korisnički identitet, informacije o vlasniku sertifikata i digitalni potpis su sastavni delovi sertifikata. Upotrebom digitalnih sertifikata obezbeđuju se sledeći sigurnosni elementi:

- Identitet korisnika
- Integritet poruke
- Autorizacija
- Neporicanje

Upotreba digitalnih sertifikata rešava neke od ključnih problema bezbednosti informacija. Ukoliko nam poruka stiže od korisnika čiji identitet utvrđujemo potpisom treće strane kojoj se veruje, nema razloga da sumnjamo da je takva poruka kompromitovana, odnosno da je stigla od osobe koje je neovlašćeno presrela komunikaciju.

CA je skraćenica od Sertifikaciono telo (eng. *Certification Authority*). Sertifikaciono telo CA je treća strana od poverenja TTP (eng. *Trusted third party*) koja izdaje i potpisuje sertifikate. Time se obezbeđuje integritet podataka, u ovom slučaju integritet izdatog sertifikata. Potpis na sertifikatu može se proveriti uz pomoć javnog ključa izdavaoca sertifikata – CA.

Sertifikaciono telo predstavlja centralni deo infrastrukture javnih ključeva. CA generiše, izdaje i poništava sertifikate. CA ima i ulogu potpisivanja sertifikata svojim privatnim ključem. Korišćenjem javnog ključa sertifikacionog centra, svako može proveriti integritet sertifikata. Zbog velike važnosti ove komponente, u slučaju napada CA se brani metodom samouništenja. To podrazumeva uništenje svih ključeva.

Na Windows serverskim operativnim sistemima moguće je instalirati poseban servis koji bi predstavljao sertifikaciono telo. Pitanje koje se postavlja je ko bi verovao

takvom sertifikacionom telu. U realnom okruženju postoje kompanije koje izdaju sertifikate, kojima se može verovati i koji su podržani od strane Internet pretraživača. Neke od najpoznatijih kompanija iz ove oblasti su: VeriSign, Entrust Authority i Comodo inc.

Proverom potpisa na sertifikatu istovremeno se utvrđuje i identitet vlasnika odgovarajućeg javnog ili privatnog ključa. Međutim, na ovaj način ne može se utvrditi identitet izdavača sertifikata. Sertifikati su javni. Zajednički format svih sertifikata, definisan je sa standardom X.509.¹³

Format X.509 određuje format zapisa i definiše polja sertifikata. Ovaj standard takođe predviđa načine povlačenje sertifikata kao i algoritme za šifrovanje. Prvi PKI projekat je podrazumevao upotrebu upravo X.509 standarda. Mnoge kompanije u svetu svoje poslovanje temelje na ovom standardu. VISA i MasterCard kao u svojim elektronskim transakcijama primenjuju X.509 standard. U nastavku poglavlja dajemo osnovne strukture X.509 sertifikata verzije 3.

X.509 ver3: Standardni sertifikat, odobren od strane ITU¹⁴	
Verzija	Verzija standarda X.509
Serijski broj	CA utvrđuje jedinstvenost sertifikata
Algoritam potpisivanja	Koji je algoritam
Davalac sertifikata	Ime CA po X.500
Period važenja	Početak i kraj
Ime korisnika sertifikata	Korisnik za čiji ključ se garantuje
Javni ključ	
Identifikator davaoca sertifikata	Neobavezno
Identifikator davaoca sertifikata	Neobavezno
Proširenja	
Potpis	Urađen sa privatnim ključem CA

CA predstavlja ovlašćenu organizaciju za izdavanje sertifikata koja ne mora da bude stalno prisutana na mreži. Osnovni zadatak je izdavanje sertifikata korisnicima. To može biti USB token, CD, pametna kartica i dr. CA za svaki generisani sertifikat računa

¹³ Dostupno na Veb sajtu: <http://en.wikipedia.org/wiki/X.509>

¹⁴ Dostupno na Veb sajtu: <http://www.itu.int/rec/T-REC-X.509>

heš vrednost, a zatim digitalno potpisuje sa svojim privatnim ključem. Na svakom sertifikatu dostupan je javni ključ ovlašćenog CA. Svaki CA¹⁵ naplaćuje svoje usluge.



Slika 4.11 Struktura digitalnog sertifikata izdatog od strane CA, verzije 3

Osnovni zadatak sertifikata je da poveže javni ključ sa podacima kao što su pojedinac, preduzeće, univerzitet itd (**Slika 4.11**). Sertifikati nisu tajni. Alisa i Bob u komunikaciji mogu da naprave hiper vezu na matičnim Veb stranama ka svojim sertifikatima.

Na primer, ako Trudi pokuša da napadne komunikaciju između Alise i Boba, jedino može da zameni Alisin i Bobov sertifikat svojim sertifikatom i da generiše svoj javni ključ. Međutim, Alisi ili Bobu već nakon prve poruke biće jasno da nije dobar potpis.

PKI je skraćenica koju koristimo za infrastrukturu sistema sa javnim ključevima (eng. *Public Key Infrastructure*). *PKI* infrastruktura sastoji se od:

- Korisnika;
- Ovlašćenih CA organizacija;

¹⁵ Dostupno na Veb sajtu: <http://www.comodo.com/>

- Sertifikata;
- Kataloga;
- Listi povučenih sertifikata.

Infrastruktura javnog ključa sa sastoji od svih podsistema koji nisu neophodni za bezbednu upotrebu kriptografije sa javnim ključevima:

- Generisanje i upravljanje ključevima;
- Sertifikaciona tela CA;
- Povlačenje sertifikata CRL;

Ne postoji opšti standard za PKI. U nastavku ćemo razmotriti nekoliko modela poverenja.

Monopolski model čini jedinstvena organizacija od poverenja sa jedno CA za sve korisnike. Ovaj model je predložila VeriSign¹⁶ iz razumljivih razloga jer je najveće komercijalno CA. Rizik je ako se takvo CA telo bilo kada kompromituje. Veliki problem može nastati ako se takvom CA ne veruje.

Oligarhijski model čini više sertifikacionih tela. Kod ovog modela, korisnik sam bira i odlučuje kom sertifikacionom telu će ukazati poverenje, a kom ne. Ovaj pristup je zastupljen kod savremenih Internet pretraživača (Veb brauzera). Oni mogu da imaju 80 i više različitih sertifikata samo da bi se verifikovao jedan digitalni potpis.

Anarhijski model je dosta različit od prethodna dva. U ovom modelu svako može da bude CA. Korisnik sam odlučuje kome će verovati. Ovaj pristup se koristi u PGP¹⁷ (eng. *Pretty Good Privacy*). Da bismo razumeli zašto se ovaj model naziva anarhijski, navešćemo sledeći primer. Pretpostavimo da je sertifikat potpisao Frank i da ga ne poznajemo, ali verujemo Bobu koji kaže da je Alisa od poverenja i da ona garantuje za Franka. Međutim, pitamo se dalje se pitamo da li treba da verujemo Franku?

Postoje mnogi drugi modeli od poverenja na bazi arhitekture modula PKI sistema:

¹⁶ Dostupno na Veb sajtu: <http://www.verisigninc.com/?cmp=SEMG02:01I>

¹⁷ Dostupno na Veb sajtu: http://en.wikipedia.org/wiki/Pretty_Good_Privacy

- Hijerarhijski model;
- Rešetkasti;
- Tranzicioni (eng. *Bridge*) CA;

Hijerarhijski model arhitekture podrazumeva *PKI* sa jednim CA i dva ili tri nivo CA po dubini kojima rut (jedan glavni za sve CA) CA izdaje sertifikate za rad. U ovom modelu svi CA i korisnici mu veruju. Možemo da ga posmatramo kao jednosmerni put poverenja.

Rešetkasti model uključuje više ravnopravnih CA koji veruju jedan drugome. Možemo da ga posmatramo kao dvosmerni put poverenja.

Tranzicioni model CA predstavlja više administrativno *PKI* sistem ili „Državu“. U njemu, organizacija CA *PKI* sistema ispod može biti rešetkasta.

4.4.5. PREDNOSTI I NEDOSTACI KRIPTOGRAFIJE SA JAVNIM KLJUČEVIMA

Na samom početku ovog poglavlja, navešćemo nekoliko prednosti kriptografije sa javnim ključem. Prvo, poverljivost koja je obezbeđena bez deljenja tajni. Ovo je veoma bitno i korisno u današnjem komercijalnom svetu. Ne postoji problem sa razmenom ključeva kao kod simetričnih šifarskih sistema.

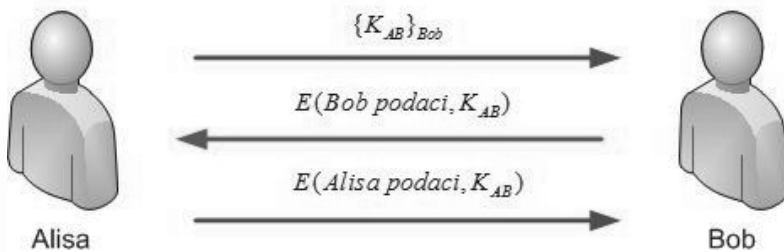
Drugo, autentifikacija može da se obavi bez deljenja tajni. Koristi se digitalni potpis kao dokaz o poreklu poruke. Ne postoji potreba za sigurnom zaštitom javnog ključa. Jedino postoji potreba da se sa sigurnošću zna da je Alisin javni ključ zaista njen ključ.

Sa druge strane, mane asimetrične, a prednosti simetrične kriptografije su brzina i ne postoji potreba za *PKI* infrastrukturuom. Algoritmi kod javne kriptografije su 2 do 3 puta sporiji iz razloga što je modularna (eksponencijalna) aritmetika računarski zahtevna.

Brzina kod sistema sa javnim ključevima nije jedini problem, postoji problem i kod dužine ključa. Dužina ključa kod *RSA* je 1024 ili 2048 bitova, a kod *AES* algoritma 128,192 ili 256 bitova. Drugi nedostaci su što se bezbednost zasniva na pretpostavkama koje nisu dokazane. I dalje ostaje mogućnost za rešavanje problema faktORIZACIJE u budućnosti.

Problem ove vrste rešava se izgradnjom hibridnih sistema. Pod hibridnim sistemom podrazumeva se kriptografski sistem u kome se sistem sa javnim ključem koristi za razmenu simetričnog ili sesijskog ključa, a za šifrovanje podataka koristi se sistemi sa simetričnim ključem. Tipična upotreba je IPsec i SSL.

Na sledećoj šemi prikazan je model jednog hibridnog šifarskog sistema u praksi. Uspostavljen je simetrični ključ pomoću sistema javnih ključeva, a šifrovanje podataka obavlja se sa simetričnim ključem. Za datu šemu pokušaćemo da utvrdimo da li je Bob siguran da komunicira sa Alisom?



Slika 4.12 Hibridni sistem sa nedostacima

Slika 4.12, ako bolje pogledamo šemu, primetićemo da je u prvoj rundi Alisa šifrovala izabrani simetrični ključ $\{K\}_{Bob}$ sa Bobovim javnim ključem, a zatim poslala Bobu. Sada možemo da zaključimo da je sistem sa javnim ključem korišćen samo za šifrovanje, ali ne i za potpisivanje. Samim tim, Bob nije autentifikovao Alisu što znači da na strani Alise može da bude bilo ko pa čak i Trudi.

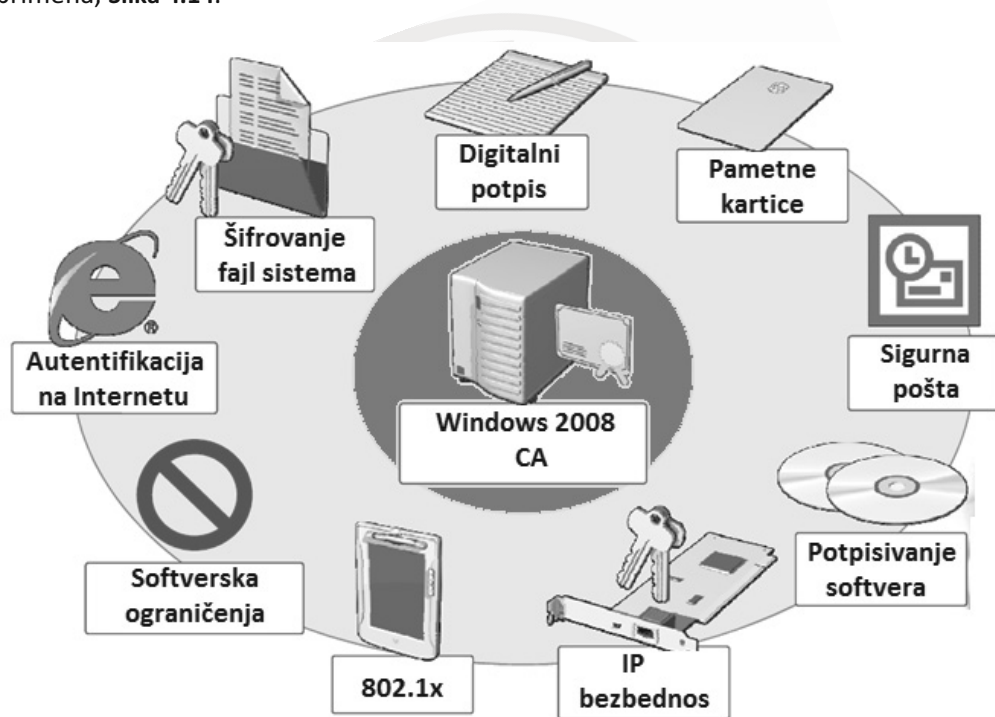


Slika 4.13 Prepravljeni hibridni sistem

Slika 4.13, predstavlja prepravljen hibridni šifarski sistem u odnosu na nedostatke koje smo uočili na prethodnom protokolu.

Tek sada u ovom drugom modelu hibridnog šifarskog sistema postoji obostrana autentifikacija i sigurna razmena simetričnog ključa. Alisa je sigurna da se na strani Boba stvarno nalazi on i Bob je siguran da se na strani Alise stvarno nalazi Alisa. Simetrični ključ koji se razmenjuje nalazi se u šifrovanoj formi.

PKI infrastruktura danas ima široku primenu u savremenim aplikacijama. Neke od primena, **Slika 4.14**.



Slika 4.14 Primena PKI infrastrukture

Ukratko ćemo razmotriti zahteve za bezbednost elektronske pošte (eng. *e-Mail*). Osnovni zahtevi kod elektronske pošte su:

- *Poverljivost*, koja mora da obezbedi tajnost sadržaja poruke za treću stranu;
- *Autentifikaciju*, utvrđivanje identiteta pošiljaoca poruke;

- *Integritet*, zaštita od izmene sadržaja od treće strane;
- *Neporecivost*, nemogućnost poricanja slanja, pošiljalac ne može naknadno da tvrdi da poruku nije poslao;

Sistem koji se koristi za ispunjenje svih navedenih zahteva je *PGP* (eng. *Pretty Good Privacy*). Razvio ga je Fil Cimerman (*Phil Zimmerman*). *PGP* je najbolji, integrisani, dostupni kriptografski algoritam u jedan algoritam. Može da se koristi na različitim platformama (*Unix, Windows, Macintosh*). Originalno je bio besplatan, ali sada postoje i komercijalne verzije.

PGP objedinjuje dva servisa:

- Servis za poruke;
- Servis za distribuciju ključa;

Servis poruke integriše servise za:

- Autentifikaciju;
- Poverljivost;
- Kompresiju;
- kompatibilnost sa programima za e-poštu;

Servis za distribuciju ključa integriše servise za:

- Generisanje, distribuciju i povlačenje javnih i tajnih ključeva;
- Generisanje i prenos simetričnih (sesijskih) ključeva ili *IV* (inicijalnih vektora);

Autentifikacija poruke je zasnovana na digitalnom potpisu, tačnije na sistemima sa javnim ključem. U procesu autentifikacije Alisa kreira heš vrednost od poruke. Zatim heš vrednost šifruje sa *RSA* algoritmom u kom koristi svoj privatni ključ po sistemu digitalnog potpisivanja. Na drugoj strani, Bob za tako primljenu poruku koristi Alisin javni ključ da bi dešifrovao poruku po sistemu za digitalno potpisivanje ili verifikaciju poslate heš vrednosti. Zapravo, poredi dve heš vrednosti (heš vrednost koju je dobio i heš vrednost koju je izračunao). Ako se slažu heš vrednosti, Bob zaključuje da jedino onaj koji ima privatni ključ može da generiše poruku.

Poverljivost poruke je zasnovana na šifrovanju sa simetričnim ključem u *CFB* šifarskom modu sa upotrebom slučajnog simetričnog ključa i *IV* (Inicijalnog vektora). Simetrični ključ i *IV*, generišu se sa javnim ključem primaoca. Podržani simetrični algoritmi su: *CAST*, *IDEA* i *3DES*. Podržani asimetrični su: *RSA* i *ElGamal*¹⁸.

Poverljivost uključuje više operacija. Na predajnoj strani generiše slučajan broj koji se koristi kao simetrični ključ samo za slanje prve poruke. Prva poruka se šifrjuje sa nekim od navedenih simetričnih blokovskih šifri. Korišćeni simetrični ključ se šifrjuje sa *RSA* algoritmom i javnim ključem primaoca. Dobijeni šifrat koji sadrži simetrični ključ, spaja se sa šifrovanom porukom. Prijemna strana koristi *RSA* algoritam i privatni ključ da bi došla do odgovarajućeg simetričnog ključa. Dobijeni simetrični ključ se koristi za dešifrovanje poruke.

Na osnovu ovog objašnjenja ili sleda operacija, možemo da zaključimo da su servis poverljivosti i autentifikacije primenjeni nad jednom porukom.

U poslednjem koraku se koristi kompresija. Poruka se kompresuje posle potpisivanja, ali svakako pre šifrovanja. Osnovni razlog je potreba da se omogući čuvanje nekompresovane poruke sa potpisom radi kasnije verifikacije. Obično se kao algoritam kompresije koristi *ZIP*¹⁹.

¹⁸ Dostupno na Veb sajtu: http://en.wikipedia.org/wiki/ElGamal_encryption

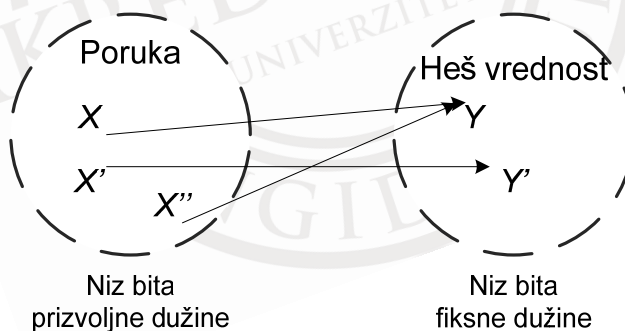
¹⁹ Dostupno na Veb sajtu: [http://en.wikipedia.org/wiki/Zip_\(file_format\)](http://en.wikipedia.org/wiki/Zip_(file_format))

5. HEŠ FUNKCIJE

Heš (eng. *hash*) funkcija u najosnovnijem obliku predstavlja sumu poruke. Na primer, to može biti suma heksadecimalnih vrednosti. Ako je data poruka 0x270 x280 x29 suma je 84. Problem u ovom slučaju je što se lako može naći druga kombinacija sadržaja koja daje isti heš (na primer, poruka: 0x250 x280 x31). Drugim rečima, isti heš se može dobiti za više različitih poruka. Kriptografski cilj je da jedan heš odgovara samo jednoj poruci. Heš funkcije se standardno primenjuju kod provere integriteta dobijene poruke.

5.1. SVOJSTVA SIGURNIH HEŠ FUNKCIJA

Proizvoljnu poruku $\{0,1\}^n$ gde je n broj bitova tj. dužina poruke, heš funkcija kompresuje u oblik $\{0,1\}^k$, gde je k određen broj bitova. Heš funkcije su funkcije kod kojih se vrši kompresija sa gubicima. Osnovna ideja je data na sledećoj slici gde su sa leve strane prikazani nizovi proizvoljne dužine, a sa desne strane je skup nizova fiksne dužine, **Slika 5.1**.

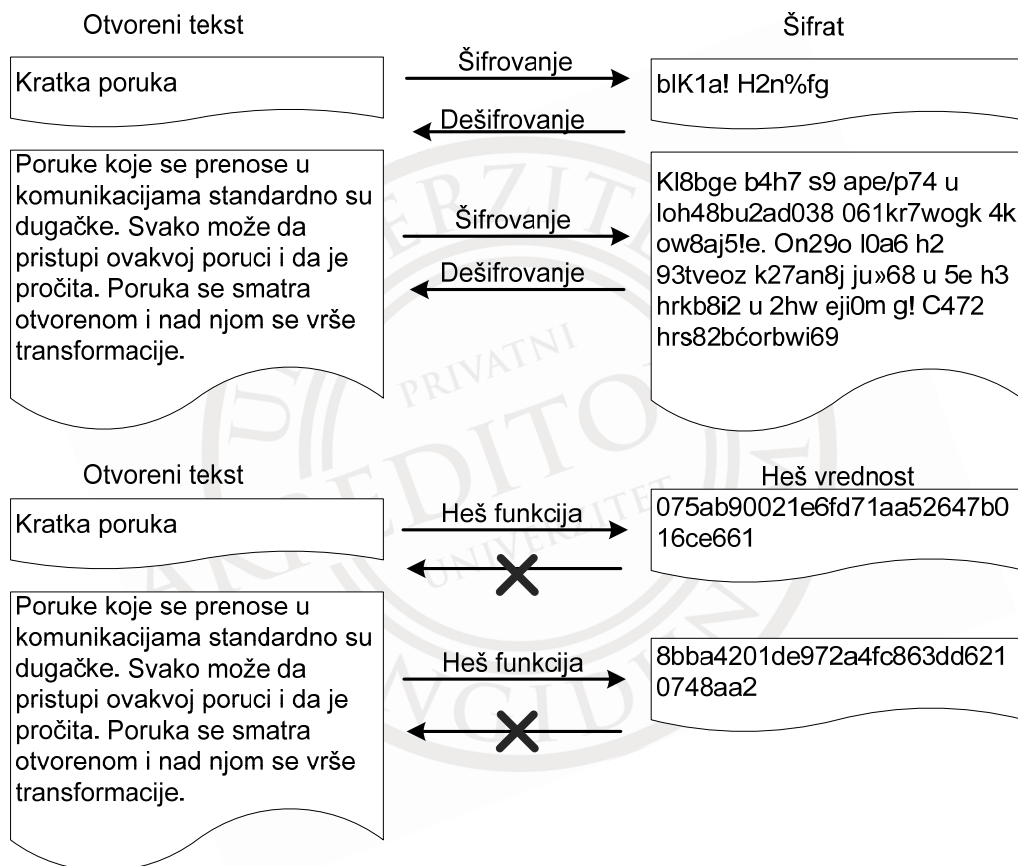


Slika 5.1 Heš funkcija vrše kompresiju sa gubicima

Dobra heš funkcija raspoređuje vrednosti uniformno. Dobijena heš vrednost ima osobinu slučajnosti. Verovatnoća da neka slučajna poruka da određenu sumu je $\frac{1}{2^k}$.

Heš funkcije se ne koriste za šifrovanje. One imaju pogodna svojstva koja se koriste za ostvarivanje kriptografskih funkcionalnosti koje se odnose na integritet prenesenih podataka, autentifikaciju učesnika u komunikaciji, neporicivost transakcija i sl. Suština

je u njihovoj jednosmernosti koja je posledica kompresije sa gubicima. Kod šifrovanja, najčešće veličina šifrata odgovara veličini otvorene poruke. Kod heš vrednosti, njena dužina je konstantna za izabranu heš funkciju i ne zavisi od dužine otvorenog teksta. Kod šifrata moguće je dešifrovanje ukoliko se poznaje ključ za dešifrovanje. Kod heš vrednosti ne postoji inverzna funkcija pošto je heš jednosmerna funkcija.



Slika 5.2 Razlika između šifrovanja i heš funkcija

Kriptografska heš funkcija $h(x)$ mora da poseduje sledeća svojstva:

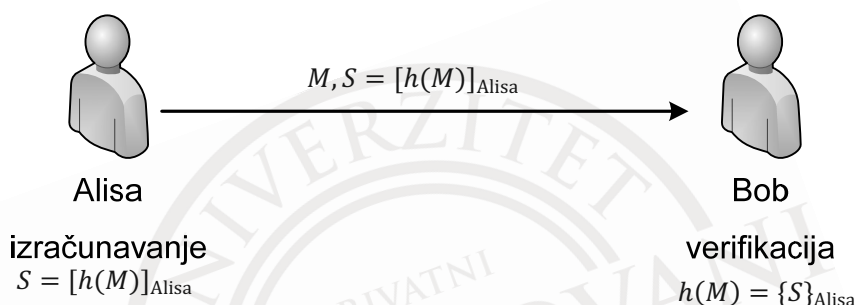
- Kompresija – Za proizvoljan broj bitova ulazne poruke x , dužina dobijene heš vrednosti $y = f(x)$ je konačna i najčešće manja od same ulazne poruke (na primer 128, 160 bitova i sl.).
- Efikasnost - heš vrednost $h(x)$ treba da se jednostavno izračuna za bilo koji ulaz x . Kompleksnost izračunavanja, naravno, zavisi od dužine ulazne poruke, ali ne značajno.
- Jednosmernost - heš funkcije su jednosmerne funkcije. To znači da je jednostavno izračunati heš vrednost za datu poruku, ali je veoma teško rekonstruisati poruku iz date heš vrednosti, tj. praktično je nemoguće.
- Otpornost na kolizije - kod dobre heš funkcije teško je naći sudar (koliziju). Ako je data poruka x_1 čija je heš vrednost $h(x_1)$ teško je naći drugu poruku x_2 tako da njihove sume budu jednake, tj. da je $h(x_1) = h(x_2)$.
- Lavinski efekat - promena samo jednog bita na ulazu treba da rezultuje u promenu bar polovine bita dobijene heš vrednosti.

Pošto je ulaz proizvoljne dužine koja znatno nadmašuje izlaz (heš vrednost), sigurno je da postoji veliki broj kolizija. Na primer, pretpostavimo da heš funkcija generiše 128 bitni izlaz. Ako se za ulaz uzme poruka dužine 160 bita, to je 32 bita duže od izlaza. Sledi da postoji još $2^{32} \approx 4.000.000.000$ mogućih ponavljanja. Otpornost na kolizije podrazumeva da je računarski veoma kompleksno naći jednu od njih.

Jedna od najvažnijih primena heš funkcija postoji kod digitalnog potpisivanja. Kao što je razmatrano u prethodnim poglavljima, Alisa potpisuje otvorenu poruku M šifrovanjem na bazi privatnog ključa, tj. $S = [M]_{\text{Alisa}}$. Dalje, Alisa šalje otvorenu poruku M i potpis S , a Bob verifikuje digitalni potpis šifrovanjem dobijenog potpisa Alisinim javnim ključem, tj. $M = \{S\}_{\text{Alisa}}$. Ukoliko je poruka M velike dužine javljaju se dva problema. Prvi problem se odnosi na kompleksnost digitalnog potpisivanja i verifikacije potpisa, tj. duža poruka se duže obrađuje, a poznato je da su algoritmi za potpisivanje spori. Drugi problem se odnosi na dodatno opterećivanje komunikacionog kanala, tj. za određeni propusni opseg potrebno je preneti dva puta više bita (i poruka i potpis).

Navedene slabosti se otklanjaju ako se primenjuje MAC (eng. *message authentication code*). Pretpostavimo da Alisa poseduje heš funkciju h . Tada se $h(M)$ može posmatrati kao "otisak prsta" (eng. *fingerprint*) poruke M , $h(M)$ je značajno manje

od M , ali identifikuje M . U ovom slučaju Alisa prvo nalazi heš vrednost poruke M , a zatim se računa $S = [h(M)]_{\text{Alisa}}$. Algoritam za izračunavanje heš vrednosti je efikasan u poređenju sa bilo kojim algoritmom za šifrovanje. Sa druge strane, digitalno se potpisuje mali broj bitova (heš je kratke dužine) što dalje doprinosi efikasnosti izračunavanja. Na prijemu Bob vrši verifikaciju digitalnog potpisa, tj. potvrđuje se $h(M) = \{S\}_{\text{Alisa}}$. Treba zapaziti da je Alisa Bobu slala samo poruku M i digitalni potpis S kratke dužine, **Slika 5.3**.



Slika 5.3 Korektno digitalno potpisivanje

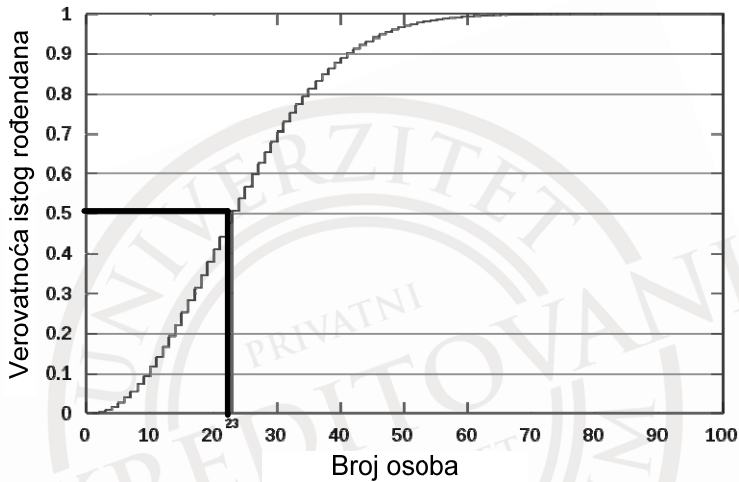
5.2. ROĐENDANSKI PROBLEM

Rođendanski problem²⁰ (eng. *the birthday problem*) je značaja za različit oblasti kriptografije. Jednim delom se odnosi i na heš funkcije.

Navedeni problem se odnosi na sledeću situaciju. Pretpostavimo da se u jednoj sobi nalazi N osoba. Postavlja se pitanje koliko treba da je N pa da se u sobi nađu osobe koje imaju rođendan istog dana. Sa stanovišta verovatnoće značajno je i sledeće pitanje. Koliko treba da bude N pa da se sa verovatnoćom $1/2$ (50%) u sobi nađe osoba koja ima rođendan istog dana. Ekvivalentno ovom pitanju je da se u zavisnosti od broja N nađe verovatnoća da u sobi ne postoje dve osobe koje su rođene istog

²⁰ U teoriji verovatnoće rođendanski problem ili paradoks odnosi se na verovatnoću da u skupu od N slučajnih izabranih osoba neko ima rođendan istog dana.

dana. Prethodna dva pitanja su komplementarna zbog toga što, ako se zna odgovor na jedno pitanje zna se i na drugo (dobijena verovatnoća na jedno pitanje se oduzme od 1). Sa verovatnoćom od 100% dve osobe će imati rođendan istog dana, ako je $N=367$ (postoji 366 mogućih rođendana uključujući 29. februar). Međutim, ako se zahteva 99% verovatnoće dovoljno je 57 osoba. Za verovatnoću od 50% broj osoba je 23, Slika 5.4.



Slika 5.4 Rođendanski problem

Gruba aproksimacija rešenja rođendanskog problema pokazuje da se do rešenja dolazi ako se nađe $\sqrt{(\text{broj mogućih rođendana} = 366)}$. Dobijeni rezultat se može iskoristiti za analizu heš funkcija. Pretpostavimo da heš funkcija $h(x)$ proizvodi vrednost dužine N . To znači da postoji 2^N različitih jednako verovatnih heš vrednosti. Na osnovu rešenja rođendanskog problema može se tvrditi je za dobijanje kolizije potrebno ispitati $\sqrt{2^N} = 2^{N/2}$ različitih ulaza. Dobijeni rezultat ukazuje na sledeće. Ukoliko se primeni napad grubom silom (eng. *brute force attack*) potrebno je da se ispita $2^{N/2}$ različitih ulaza i pronaći će se dva koja daju isti izlaz. Ovakav napad je porediv sa napadom na simetrične kriptografske sisteme. Ukoliko je kod simetričnih sistema koristi tajni ključ dužine N za napad grubom silom potrebno je ispitati $2^N/2 = 2^{N-1}$ mogućih ključeva. Pod pretpostavkom da ne postoje prečice (tajna

vrata, slabosti i sl.) u kompletnim sistemima zaštite gde se najčešće koriste više tehnika, potrebno je obezbediti sledeće:

Za isti nivo bezbednosti, dužina heš vrednosti mora da bude barem dva puta veća od dužine tajnog ključa kod simetričnog šifarskog sistema.

Rezultat iz rođendanskog problema može da se koristi i za napad na šifarski sistem. U engleskoj literaturi napad je poznat pod nazivom *Birthday Attack*. Podsetimo se, kada Alisa želi da digitalno potpiše poruku M , ona vrši šifrovanje i izračunava $S = [h(M)]_{\text{Alisa}}$ i šalje M i S Bobu. Pretpostavimo da heš funkcija daje vrednost dužine N bitova. Napadač Trudi sprovodi sledeće akcije zbog mogućeg napada koji se odnosi na rođendanski problem:

- Trudi generiše poruku X koja sadrži “nezgodne” podatke za Alisu. Naravno, sama Alisa nikada neće potpisati takvu poruku;
- Trudi takođe generiše jednu naivnu poruku Y , koja sadrži rutinske podatke i koju Alisa u nekom protoklu regularno potpisuje i sigurno joj ne može naneti štetu;
- U nastavku, Trudi generiše $2^{N/2}$ varijacija naivne poruke Y . Varijacije nastaju nebitnim (minimalnim) izmenama polazne poruke Y . Označimo ih sa Y_i gde je $i = 0, 1, 2, \dots, 2^{N/2} - 1$. Sve dobijene poruke imaju isti naivni smisao kao i polazna, ali im se heš vrednosti razlikuju;
- Slično prethodnom, Trudi generiše $2^{N/2}$ varijacija “nezgodne” poruke X . Varijacije nastaju nebitnim (minimalnim) izmenama polazne poruke X . Označimo ih sa X_i gde je $i = 0, 1, 2, \dots, 2^{N/2} - 1$. Sve dobijene poruke imaju isti “nezgodan” smisao kao i polazna, ali im se heš vrednosti razlikuju;
- Trudi sprovodi heš funkciju nad svim porukama X_i i Y_i . Pošto joj je poznat rođendanski problem ona očekuje da pronađe koliziju, tj. $h(X_j) = h(Y_k)$. Ukoliko pronađe koliziju, Trudi šalje naivnu poruku Y_k ka Alisi i traži da je ona potpiše. Pošto se radi o standardnoj i naivnoj poruci, Alisa je potpisuje i vraća ka Trudi Y_k i $[h(Y_k)]_{\text{Alisa}}$. Pošto je $h(X_j) = h(Y_k)$ sledi da je $[h(X_j)]_{\text{Alisa}} = [h(Y_k)]_{\text{Alisa}}$. Trudi je uspela je da dobije Alisin digitalni potpis na svoju “nezgodnu” poruku X . Poruka bi mogla čak da bude npr. prebaciti sve pare sa Alisinog bankovnog računa na Trudin račun.

Ovde treba zapaziti da je napadač Trudi uspeła da dobije Alisin digitalni potpis na svoju poruku, bez napada na šifarski sistem koji se koristi za digitalno potpisivanje niti na odgovarajući privatni ključ. Razmatrani napad je napad tipa grube sile na korišćenu heš vrednost koja se digitalno potpisuje. Za prevenciju od ovog napada, u šifarskim sistemima treba koristiti heš funkcije koje daju duže heš vrednosti kako bi se onemogućilo da potencijalni napadač izračunava $2^{N/2}$ različitih heš vrednosti u razumnom vremenu. Razmatrani problem dodatno ukazuje na kriptološke sisteme sa praktičnom tj. računarski sigurnom tajnošću.

5.3. KLASIČNE SUME – NEKRIPTOGRAFSKE HEŠ FUNKCIJE

Postoji veliki broj heš funkcija koje su u primeni, ali nisu pogodne za kriptografsku primenu. Neka je X skup polaznih podataka dužine n

$$X = (X_0, X_1, X_2, \dots, X_{n-1}),$$

gde je X_i bajt. Jedna moguća heš funkcija $h(X)$ definiše se sa

$$h(X) = (X_0 + X_1 + X_2 + \dots + X_{n-1}) \bmod 256.$$

Ovakva funkcija ima osobinu kompresije zato što za ulaz bilo koje dužine daje 8-bitni izlaz. Za ovakvu heš funkciju lako je pronaći koliziju. Na osnovu rođendanskog problema jasno je da je posle samo $2^4 = 16$ pokušaja moguće naći ulaz koji daje istu heš vrednost. Pored toga, još jednostavnije je direktno konstruisati kolizije. Na primer, zamenom dva bajta u ulaznim podacima uvek se dobija kolizija.

$$h(10101010, 00001111) = h(00001111, 10101010) = 10111001.$$

Treba zapaziti da u razmatranoj funkciji ne samo da je izlaz kratke dužine, već je algebarska struktura koja je u osnovi ove funkcije isuviše jednostavna.

Neka je definisana druga heš funkcija $h(X)$ na sledeći način

$$h(X) = (nX_0 + (n-1)X_1 + (n-2)X_2 + \dots + 2X_{n-2} + X_{n-1}) \bmod 256.$$

Postavlja se pitanje da li je ovo dobra heš funkcija. Ona je bolja od prethodne, ali još uvek ne zadovoljava sve zahteve. Bolja je od prethodne zbog toga što kada se izmeni redosled bajtova na ulazu dobija se različit rezultat, tj.

$$h(10101010, 00001111) \neq h(00001111, 10101010).$$

Međutim, i dalje se na osnovu rođendanskog problema brzo mogu naći kolizije. Pored toga, i dalje je lako konstruisati kolizije. Na primer:

$$h(00000001, 00001111) = h(00000000, 00010001) = 00010001.$$

Jedna od najpoznatijih funkcija koja je u upotrebi je ciklična redundantna provera – CRC (eng. *cyclic redundancy check*). Najčešće se koristi kao kod za detekciju greške kod prenosa podataka u računarskim mrežama ili na uređajima za memorisanje i čuvanje podataka. I kod ove funkcije lako je pronaći kolizije ili konstruisati kolizije za svaki CRC. Međutim, postoje primeri gde je CRC korišćen kao kriptografska heš funkcija, što je apsolutno pogrešno. Na primer, kod WEP²¹ protokola u bežičnim računarskim mrežama CRC se koristi kao kriptografska funkcija za očuvanje integriteta podataka koji se prenose. Inteligentni napadač će bez problema promeniti same podatke, a da se CRC ne promeni. Navedeni problem je tako postao osnova za različite vrste napada na WEP protokol.

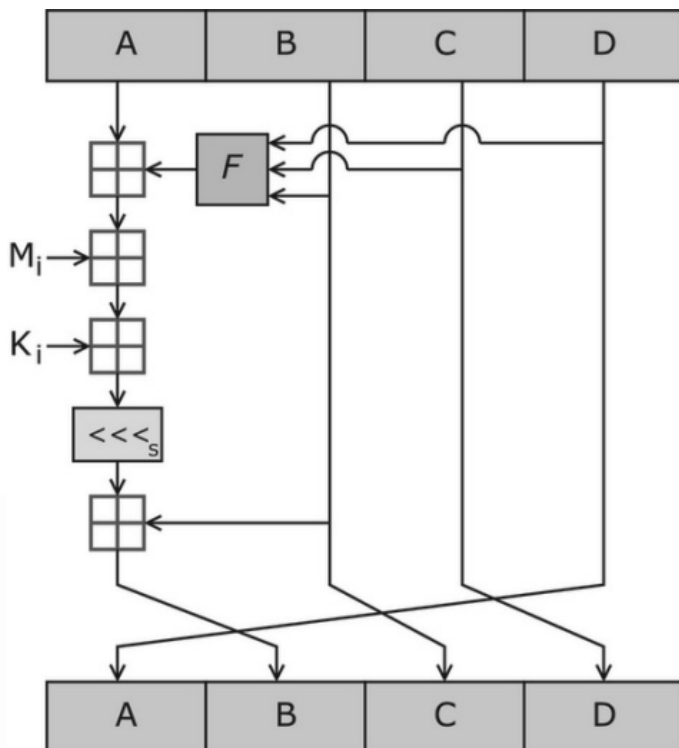
5.4. MD5

MD5 (eng. *Message-Digest algorithm 5*) heš funkcija predstavlja primer prave kriptografske heš funkcije koja se najčešće koristila. Danas se zna da ova heš funkcija ima slabosti i da je neotporna na kriptografske napade, tako da se ređe primenjuje u kriptografiji. Još uvek ima svoju namenu u proveru integriteta većih fajlova (pre svega zbog svoje brzine rada). Dužina dobijene heš vrednosti (sažetka, suma) je 128 bitova. Tipično se predstavlja sa 32 heksadecimalne cifre.

²¹ Wired Equivalent Privacy (WEP) je šifarski protokol u IEEE 802.11 bežičnim računarskim mrežama. Definisan je sa ciljem da se obezbedi servis poverljivosti kod bežičnih mreža koji je uporediv sa tradicionalnim žičanim računarskim mrežama.

MD5 algoritam je razvio Ronald Rivest 1991. godine. Baziran je na prethodnom MD4 algoritmu i nešto je sporiji od njega. Danas se zna da je podložan na „*brute force birthday attack*“. Jedan takav projekat pod imenom MD5CRK je pokrenut 1. marta 2004. godine sa namerom da dokaže slabost MD5 algoritma. Nedugo zatim, 17. avgusta 2004. godine, objavljeno je da su Ksiaoun Vang, Denguo Feng, Ksuejia Lai i Ksongbo Ju uspešno razbili algoritam odnosno da su pronašli koliziju na algoritmu. Za razbijanje ovog algoritma bio im je potreban samo jedan sat na IBM p690 klasteru. 2005. godine Arjen Lenstra, Ksiaoun Vang, i Bene de Veger su demonstrirali su kreiranje dva X.509 sertifikata sa različitim javnim ključevima i istom MD5 heš vrednosti. Nekoliko dana potom Vlastimil Klima je kreirao unapređeni algoritam koji je u stanju da na običnom PC računaru za nekoliko sati kreira koliziju MD5 algoritma. Danas je to moguće uraditi za jedan minut.

MD5 algoritam (**Slika 5.5**) kao ulaz koristi n -bitnu poruku (niz bitova) $m_0, m_1, m_2, \dots, m_{n-2}, m_{n-1}$. Poruka se mora dopuniti bitovima kako bi njena ukupna dužina odgovarala broju 448. Poruka se proširuje tako da joj nedostaju 64 bita, da njena ukupna dužina u bitovima bude deljiva sa 512. Najčešći način proširivanja poruke je da se poruci prvo doda jedan bit sa vrednosti 1, a zatim slede bitovi sa vrednošću 0. Tako će se poruci dodati ili samo jedan bit ili u najgorem slučaju 512 bitova. Nakon proširenja poruke, poruci je potrebno dodati 64-bitnu reprezentaciju broja b (b je dužina izvorne poruke pre njenog proširenja). U slučaju da se dužina poruke ne može prikazati pomoću 64 bita, poruci se dodaju samo nižih 64 bita. Bitovi reprezentacije broja b se dodaju poruci kao dve 32-bitne reči, pri čemu je reč manje težine prva pridodata.



Slika 5.5 Jedna MD5 operacija

Nakon što je poruka pripremljena za MD5 algoritam, potrebno je inicijalizovati 128-bitni MD bafer. MD bafer se sastoji od četiri 32-bitnih reči A, B, C i D. Kao inicijalne vrednosti reči u heksadecimalnom sistemu se koriste: A=67452301, B=EFCDA89, C=98BADCFE i D=10325467. Posle inicijalizovanja pokreće se MD5 algoritam koji se ponovo izvodi za svakih sledećih 512 bitova poruke. Samo jezgro algoritma predstavlja funkcija za kompresiju koja se sastoji od četiri ciklusa. Svaki od četiri ciklusa ima sličnu strukturu, ali svaki koristi drugačiju primitivnu logičku funkciju F, G, H ili I. Konačan rezultat predstavljaju vrednosti u registrima A, B, C i D koje se sabiraju sa njihovim inicijalnim vrednostima. Svaki od tih četiri registara predstavlja jednu četvrtinu dobijene heš vrednosti ulazne poruke.

38. PRIMER:**Slika 5.6,** Upotreba MD5 funkcije u Cryptool-u.**Slika 5.6** Primer MD5 heš funkcije

Primetićemo, **Slika 5.6**, da poruka 2 sadrži na kraju reči dva slova „M“ i to je jedina razlika u odnosu na poruku 1, dok su heš vrednosti potpuno različite.

5.5. SHA

SHA (eng. *Secure Hash Algorithm*) je klasa kriptografskih funkcija za sažimanje. U ovoj grupi kriptografskih funkcija najčešće korišćena funkcija je SHA-1 koja je našla primenu u velikom broju kriptografskih protokola kao što su TLS, SSL, PGP, SSH, S/MIME i IPsec. SHA-1 se može smatrati naslednikom MD5 algoritma. SHA algoritmi su kreirani od strane američke agencije za bezbednost (NSA) i publikovani kao zvanični standard vlade SAD. Heš vrednost je dužine 160 bitova.

Prvi član grupe SHA algoritama, publikovan je 1993. godine, i zvanično nazvan SHA, ali se on često u literaturi naziva SHA-0, da bi se sprečila zabuna sa njegovim kasnijim naslednicima. Dve godine kasnije nastaje SHA-1, prvi naslednik SHA. Još četiri varijante ovog algoritma su publikovane i to SHA-224, SHA-256, SHA-384 i SHA-512 i oni se jednim imenom nazivaju SHA-2, a razlikuje ih samo dužina sažetka koji proizilazi iz njih. Algoritmi SHA-256, SHA-384 i SHA-512 objavljeni su 2002. godine dok je varijanta SHA-224 objavljena 2004. godine. Ove duže varijante su znatno sigurnije u odnosu na SHA-0 i SHA-1, prvenstveno zbog dužine sažetka i određenih promena u algoritmu. Nove verzije SHA algoritma koriste druge vrednosti pomeranja kao i dodatne konstante, ali u suštini nema značajnih promena u odnosu na prethodnike. I pored veće sigurnosti oni se i danas manje koriste u odnosu na SHA-1 koji predstavlja industrijski standard u kriptografiji.

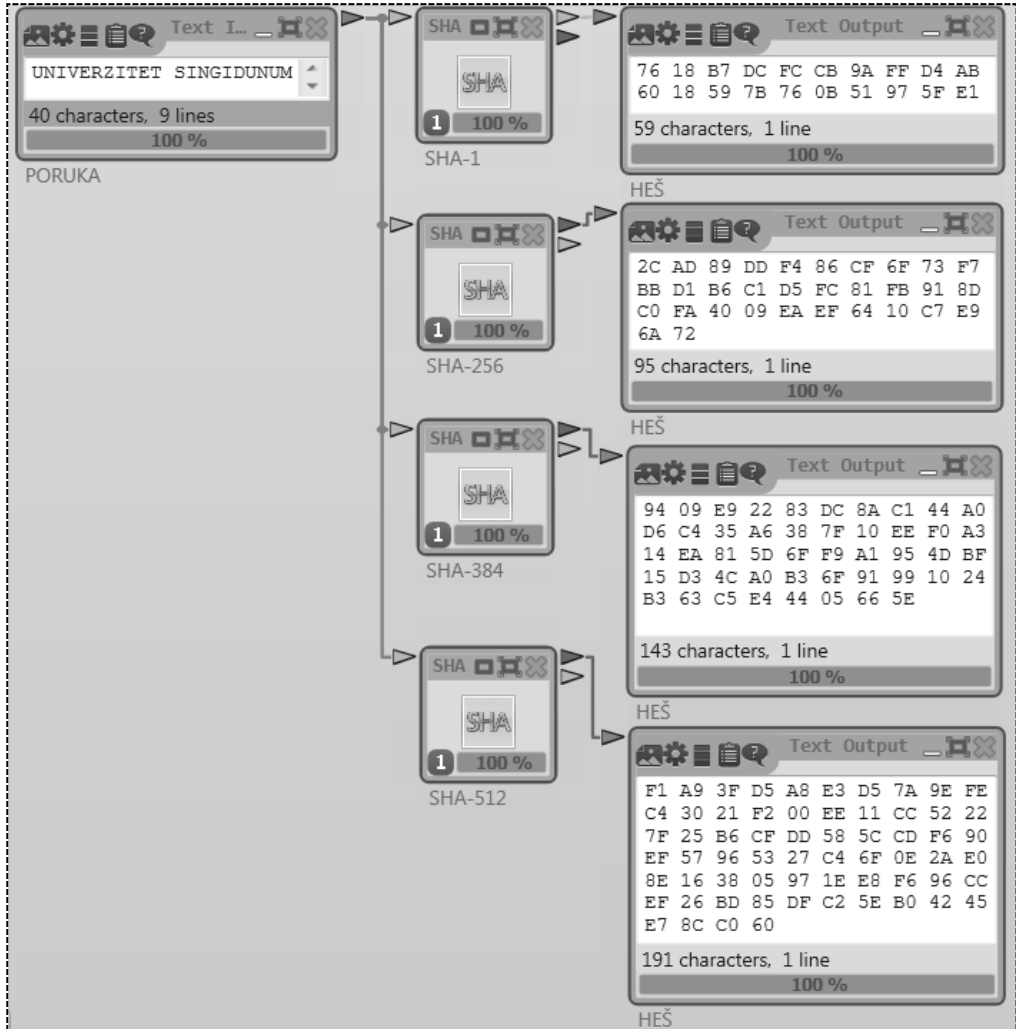
SHA-0 i SHA-1 algoritmi se danas ne mogu smatrati bezbednim zato što postoje kriptografski napadi koji su sposobni da pronađu koliziju za relativno kratko vreme.

Tabela 12 Poređenje heš algoritama

Algoritam	Veličina sažetka u bitovima	Inicijalna veličina u bitovima	Veličina bloka u bajtovima
SHA-0	160		
SHA-a	160	160	64
SHA-224	224	256	64
SHA-256	256	256	64
SHA-384	384	512	128
SHA-512	512	512	128

39. PRIMER:

Slika 5.7, potreba različitih verzija SHA funkcije u Cryptool-u.



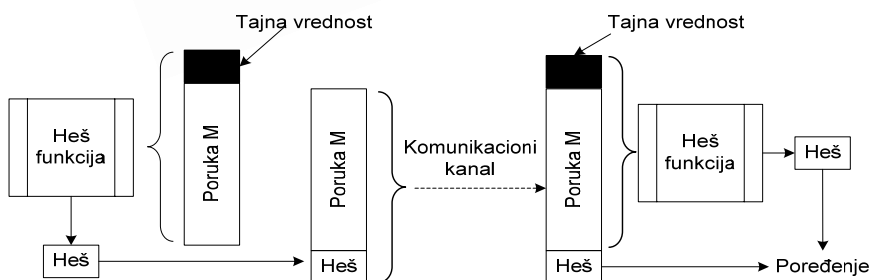
Slika 5.7 Primeri SHA heš funkcije

5.6. HEŠ FUNKCIJA I INTEGRITET

Za obezbeđivanje funkcije integriteta poruke koja se prenosi primenjuje se kod za autentifikaciju poruke – MAC (eng. *message authentication code*). U poglavlju o blokovskim šiframa razmatrana je jedna tehnika za generisanje MAC-a gde se blok šifra koristi u CBC režimu rada. Kako heš funkcija predstavlja sažetak poruke (eng. *message digest*) moguće je da se ova vrednost iskoristi za verifikaciju integriteta poruke. Ovakav pristup se označava sa HMAC (eng. *hashed MAC*).

Za obezbeđivanje funkcionalnosti integriteta poruke M , Alisa izračunava heš vrednost $h(M)$ i šalje ka Bobu i M i $h(M)$. Ukoliko se na prenosnom putu desi promena poruke iz M u M' Bob će jednostavno konstatovati da se to ne slaže sa dobijenom heš vrednosti $h(M)$. Međutim, razmatrani koncept nije dovoljan. Napadač Trudi može da zameni originalnu poruku M sa svojom porukom M' , a da zatim izračuna i da zameni $h(M)$ sa $h(M')$. Na ovaj način Bob ne bi mogao da detektuje promenu podataka. Da bi se ovo izbeglo neophodno je da se heš vrednost šifrjuje. Moguće je da se koristi simetričan šifarski sistem sa tajnim deljenim ključem između Alise i Boba, $E(h(M), K)$. Drugi način je da se koristi asimetričan šifarski sistem, tj. u ovom slučaju $[h(M)]_{Alisa}$.

Umesto šifrovanja koje je standardno kompleksan proces za realizaciju, moguće je koristiti HMAC bez šifrovanja. Kod ovog pristupa tajni deljeni ključ K se direktno dodaje poruci M pre izračunavanja heš vrednosti. Moguća su dva pristupa. Ključ se može dodati na početak ili na kraj poruke. Nakon toga se sprovodi $h(K, M)$ ili $h(M, K)$, respektivno. Princip korišćenja deljene tajne vrednosti, izračunavanja i korišćenja HMAC, prikazano Slika 5.8.



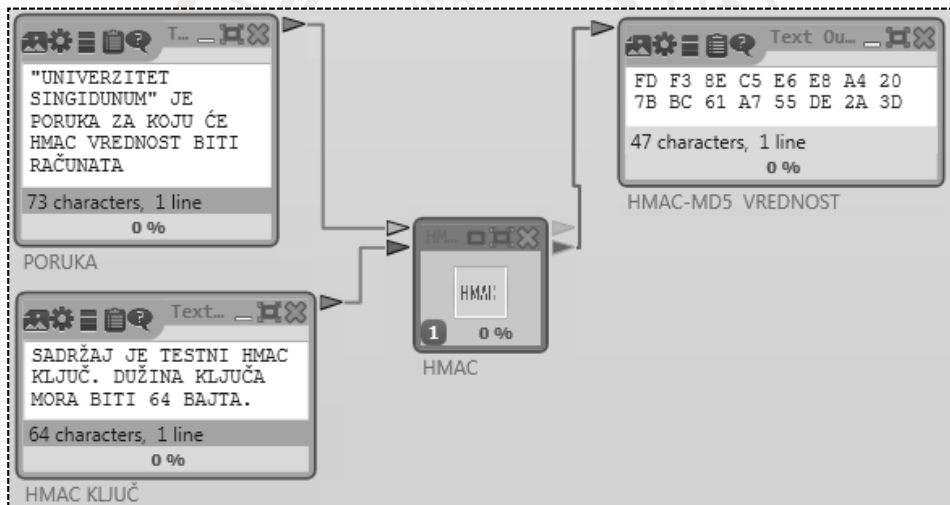
Slika 5.8 HMAC bez šifrovanja

Oba predložena postupka za HMAC imaju svoje slabosti i omogućavaju napad na dobijeni HMAC, tj. napadač može promeniti poruku M u M' bez poznavanja tajnog deljenog ključa K . Problem je u tome što većina kriptografskih heš funkcija rade blokovski. Na primer, MD5 ili SHA1 se izračunavaju nad blokovima dužine 512 bitova. Dalje, izračunavanje se vrši u više rundi gde je izlaz jedne runde ulaz u drugu rundu, a za transformaciju nad blokovima se koriste poznate funkcije. Pretpostavimo da se izračunava HMAC po principu $h(K, M)$. Pošto heš algoritam radi nad blokovima tada se poruka M može predstaviti kao $M = (B_1 B_2)$. Neka je početna inicijalna vrednost za prvu rundu A , a koristi se poznata funkcija F . Tada se može pisati:

$$h(M) = F(F(A, B_1), B_2) = F(h(B_1), B_2).$$

40. PRIMER:

Slika 5.9, računanje HMAC vrednosti za izabranu poruku u Cryptool-u.



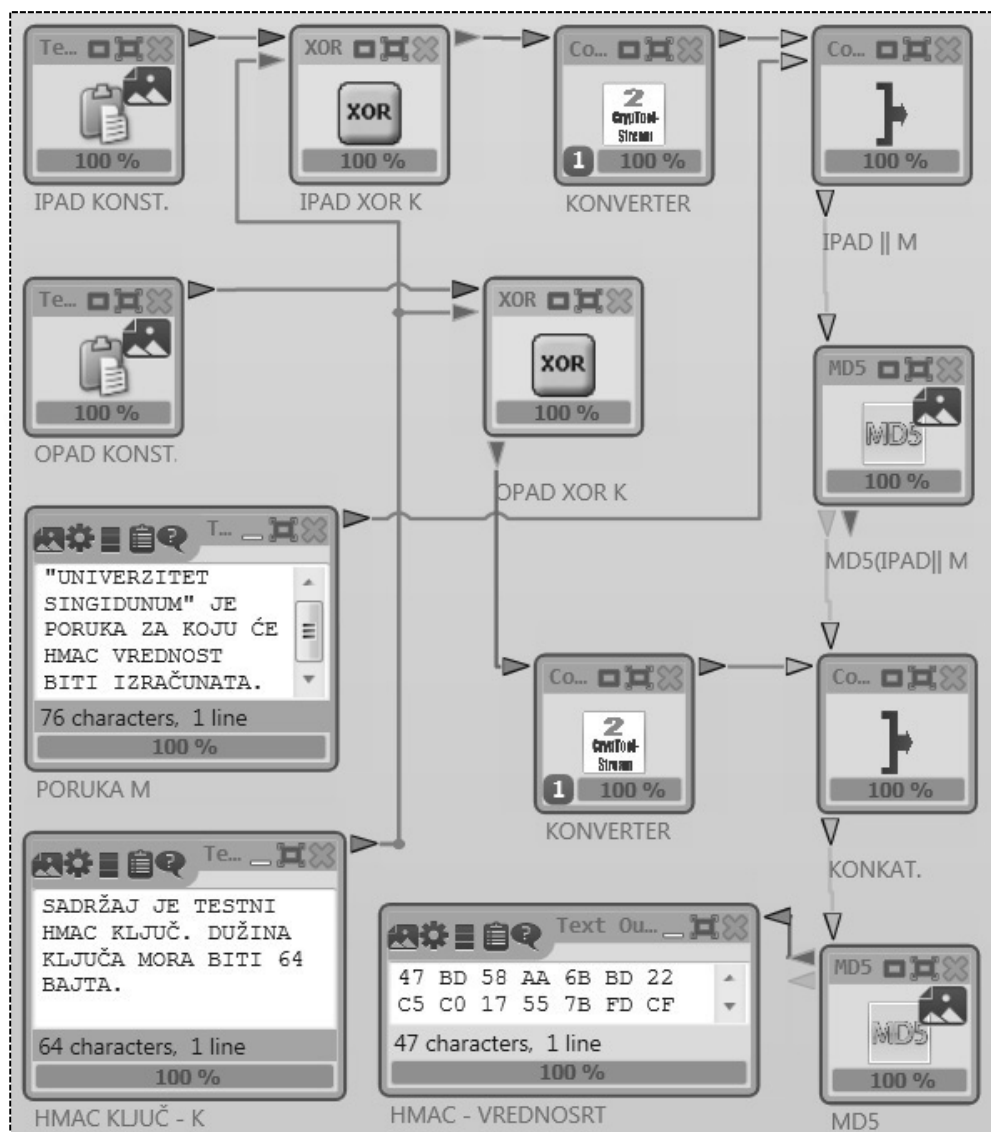
Slika 5.9 Računanje HMAC vrednosti

Trudi može na osnovnu poruku da dopiše deo X i da generiše svoju poruku M' na način da je $M' = (M, X)$. Nadalje Trudi može bez poznavanja ključa K da pronade $h(K, M')$ na osnovu $h(K, M)$ na osnovu sledećeg:

$$h(K, M') = h(K, M, X) = F(h(K, M), X).$$

41. PRIMER:

Slika 5.10, realizacija HMAC algoritma u Cryptool-u za RFC 2104.



Slika 5.10 Realizacija HMAC algoritma u Cryptool-u

Slična situacija nastaje i kada se radi sa $h(M, K)$. Navedeni problemi su posledica jednostavnog dodavanja ključa K na početak ili na kraj poruke. Rešenje koje se koristi u praksi na sofisticiraniji način promeša tajni deljeni ključ K sa porukom M , a zatim se izračunava heš vrednost.

HMAC algoritam u RFC dokumentu 2104²², definisan je izrazom:

$$HMAC(K, m) = H((K \oplus opad) \parallel H((K \oplus ipad) \parallel M))$$

H - kriptografska heš funkcija,

K - tajni ključ,

M - poruka koja će biti autentifikovana,

$\parallel$ - simbol za konkatenaciju (nadovezivanje, spajanje),

$\oplus$ - XOR ili ekskluzivno ili,

$opad$ - spoljna dopuna (jedan heksadecimalni blok konstantne dužine),

$ipad$ - unutrašnja dopuna (heksadecimalni blok konstantne dužine)

Snaga HMAC vrednosti zavisi od veličine tajnog ključa koji je korišćen za računanje vrednost. Najpoznatiji napad je napad sa potpunom pretragom tajnog ključa ili isprobavanjem svih mogućnosti. HMAC je znatno manje izložen napadu „sudaranja“ za razliku od osnovnih heš funkcija. HMAC-MD5 ne pati od istih nedostataka koji su pronađeni u MD5.

Vrednosti $ipad$ i $opad$ nisu kritične za bezbednost algoritma, ali poželjno je da budu generisane na takav način da njihova Hemingova²³ distanca bude velika. Ovim se postiže manja zajednička informacija između unutrašnjih i spoljašnjih delova.

²² RFC-2104. Dostupno na Veb sajtu: <http://tools.ietf.org/html/rfc2104>

²³ Hamingovom distancom određuje se stepen različitosti dve binarne sekvence. Očekivana Hamingova distanca za dva slučajna niza iznosi 0,5.

6. AUTENTIFIKACIJA

Termin kontrola pristupa (eng. *access control*) se koristi za sve bezbednosne probleme koji se odnose na pristup resursima sistema. U ovoj široj definiciji postoje dva područja koja su od primarnog značaja, autentifikacija i autorizacija (odobravanje). Autentifikacija (eng. *authentication*) je proces utvrđivanja da li korisniku (osoba, računar, mašina, jednom rečju entitet) treba dozvoliti pristup sistemu. Na ovo mestu razmotrićemo metode koje se koriste da bi se izvršila autentifikacija osobe na lokalni računar. Posebna vrsta problema postoji kada informacija o autentifikaciji mora da prođe kroz mrežu. Iako deluje kao da su ove dve vrste problema blisko povezane one su, zapravo, potpuno različite. Kada su računarske mreže u pitanju, autentifikacija je skoro u potpunosti problem vezan za bezbednosne protokole.



Slika 6.1 Autentifikacija je postupak za pouzdano predstavljanje

Po definiciji, autentifikovanim korisnicima je dozvoljen pristup resursima sistema. U principu, pun pristup svim resursima sistema se ne daje svim korisnicima. Na primer, samo privilegovani korisnik, poput administratora, mogao bi dobiti dozvolu da instalira softver. Autorizacija je skup postupaka i tehnika za ograničavanje rada autentifikovanih korisnika. Treba zapaziti da je autentifikacija binarna odluka – pristup je ili odobren ili nije, dok se autorizacija odnosi na više pravila o pristupanju različitim sistemskim resursima. U polju bezbednosti, terminologija je daleko od toga da bude standardizovana. U svakodnevnoj upotrebi, kontrola pristupa ima širu

definiciju u koju spadaju i oblasti autentifikacije i autorizacije. Ove dve oblasti kontrole pristupe se mogu sažeto predstaviti na sledeći način:

- Autentifikacija: Da li ste onaj za koga se predstavljate?
- Autorizacija: Da li imate dozvolu da uradite određenu akciju?

U nastavku će se razmotriti različite metode koje se uglavnom koriste da bi se autentifikovala osoba od strane lokalnog računara. U suštini, treba “ubediti” računar da je korisnik zaista onaj za koga se predstavlja. Naravno, poželjno je da se ovo obavi na najsigurniji mogući način. Računar može autentifikovati osobu na osnovu sledećeg:

- nešto što znate;
- nešto što imate;
- nešto što jeste.

Lozinka je primer “*nečega što znate*”. One su često najslabija karika u mnogim savremenim bezbednosnim sistemima. Primer “*nečega što imate*” jeste npr. platna kartica, mobilni telefon i sl. Kategorija *nešto što jeste* se odnosi na biometriju. Danas se, na primer, najavljivanje za rad na laptop računaru (logovanje) može uraditi preko skenera prsta. Pouzdaniji mehanizmi mogu da kombinuju više navedenih tehnika.

6.1. LOZINKE

Idealna lozinka je nešto što znate, nešto što računar može da proveri da znate i nešto što niko drugi ne može da pogodi – čak ni sa pristupom neograničenim računarskim resursima. U praksi je veoma teško stvoriti takvu idealnu lozinku.

Danas je skoro nemoguće koristiti računar bez korišćenja velikog broja lozinki. Mnogi korisnici se loguju na svoj računar tako što ukucaju korisničko ime i lozinku. Pored toga, koriste se i drugi podaci koji se ne smatraju lozinkama, ali funkcionišu na isti način. Na primer, PIN broj koji se koristi kod platnih kartica je, zapravo, lozinka. Ukoliko neko zaboravi lozinku, postoje opcije autentifikovanja preko broja lične karte, matičnog broja, datuma rođenja, ili nekog drugog ličnog podatka. U tom slučaju, te informacije služe kao lozinka. Problem sa ovim vrstama lozinke je što najčešće nisu

tajne. Korisnici često biraju loše lozinke što potencijalnim napadačima pojednostavljuje otkivanje (razbijanje) lozinke (eng. *password cracking*). Postoje matematički dokazi koliko je teško bezbednost nekog sistema zasnivati na lozinkama.

Sa bezbednosnog stanovišta, rešenje za probleme sa lozinkama bi bilo da se umesto njih koriste slučajno generisani kriptografski ključevi. Tada bi pokušaj razbijanja lozinke bio ekvivalentan sa problemom pretrage po svim mogućim realizacijama datog kriptografskog ključa. Problem sa takvim pristupom je što ljudi moraju da pamte svoje lozinke, a nije lako pamtiti slučajne binarne cifre. Treba spomenuti i popularnost lozinke. Mnogo je ekonomičnije i jednostavnije bazirati autentifikaciju na nečemu što nam je poznato, pre svega na nečemu što znamo. Poseban problem predstavlja potreba za korišćenjem različitih lozinki za različite sisteme koje koristimo. Takođe, lozinke su besplatne, za razliku od kartica i biometrijskih uređaja.

6.1.1. KLJUČEVI ILI LOZINKE

Pretpostavimo da je umesto lozinke u upotrebi kriptografski ključ dužine 64 bita. Broj mogućih ključeva u ovom slučaju je 2^{64} . Pretpostavimo da su ključevi potpuno slučajni i da ne postoji napad skraćenim postupkom (ne postoji slabost u realizaciji – prečica). Za razbijanje ovakve lozinke bilo bi potrebno u proseku da se ispita 2^{63} ključeva kako bi napad bio uspešan.

Sa druge strane, kada bi neko pokušao da razbije lozinku od 8 ASCII karaktera, gde je svaki karakter kodiran sa 8 bitova ($2^8 = 256$ mogućih vrednosti za jedan karakter), imao bi $256^8 = 2^{64}$ mogućih lozinki. Na prvi pogled, razbijanje lozinki deluje podjednako kompleksno kao razbijanje ključeva. Međutim, korisnici uglavnom ne biraju lozinke potpuno slučajno zato što moraju da ih zapamte. Na taj način, bilo ko ko pokuša da razbije lozinku rešavao bi lakši problem. Broj kombinacija koje treba ispitati kod lozinke je sigurno manji od 2^{63} .

Za razbijanje lozinki u upotrebi su rečnici dobro poznatih lozinki. Na primer, ako je lozinka dužine 8 karaktera za njeno razbijanje se može koristiti rečnik unapred spremljenih lozinki veličine $2^{20} \approx 1.000.000$ reči. Sa ovakvim rečnikom postojala bi velika verovatnoća da se otkrije lozinka. Nasuprot ovome, ako bi napadač pokušavao da razbije ključ dužine 64 bita sa rečnikom od 2^{20} unapred spremljenih ključeva,

šansa da to uradi bi iznosila $2^{20}/2^{64} = 1/2^{44}$ što približno iznosi 1 od 17 triliona. Suština problema sa lozinkama je u tome što nisu slučajne. Poseban problem su korisnici koji veoma često biraju jednostavne lozinke koje se lako pamte npr: password, lozinka, admin, 123456, bojan i sl.

6.1.2. IZBOR LOZINKE

Ne stvaraju se sve lozinke na isti način. Na primer, lozinke sa ličnim imenima i datumima rođenja su slabe lozinke koje se lako mogu razbiti. S obzirom na to da bezbednost informacija i sistema često zavisi od jačine lozinke, potrebno je da korisnik odabere lozinku koju nije lako razbiti, ali koja može lako da se pamti. Moguće je argumentovati jačinu sledećih lozinki:

- Kd8\$6g)0!25mV6%
- 52987160028731
- POkemON
- P10Di7god

Prva lozinka, "Kd8\$6g)0!25mV6%" je preteška za napadača, ali je takođe preteška i za Alisu pošto mora da je pamti. Druga lozinka na listi ima previše cifara da bi bilo ko od korisnika mogao da je zapamti. Lozinku POkemON je možda teško pogoditi s obzirom na to da nije standardna reč iz rečnika, a koristi se kombinacija velikih i malih slova. Problem nastaje ako neko poznaje korisnika i njegova interesovanja te bi ova lozinka, u tom slučaju, bila relativno laka za razbijanje. Poslednja lozinka na spisku P10Di7god, možda na prvi pogled deluje kao da spada u kategoriju lozinki koje je teško i pogoditi i zapamtiti. S obzirom na to da je ova lozinka zapravo skraćena fraza (pre 10 dana i 7 godina), korisniku bi bilo lako da je zapamti, a bilo bi teško nekome ko nije korisnik da je pogodi.

Prema mnogim istraživanjima najbolja opcija su lozinke koje se zasnivaju na skraćenim frazama zato što ih je lako zapamtiti, a teško ih je razbijati. Pored toga, dobro je poznato da je teško postići poslušnost korisnika. U nekim situacijama je lakše korisniku obezbediti lozinku, mada korisnik teže pamti lozinku koja mu je data

od one koju sam smisli. Da bi se testirala jačina lozinke, najbolje je da administratori koriste program za razbijanje lozinki kao test jačine. Često se predlaže da se lozinka menja na određeno vreme, a u nekim slučajevima je to i obaveza korisnika. Korisnik bi ovo mogao da izbegne ukucavanjem iste lozinke ili korišćenjem par istih lozinki više puta. Problemi sa korisnicima i lozinkama je što ih nije lako naterati da izaberu relativno jaču i komplikovaniju lozinku (eng. strong password).

6.1.3. NAPADI PREKO LOZINKE

Posmatrajmo napadača Trudi koja nema pristup (nalog za rad) određenom računarskom sistemu. Tipična putanja za napad bi bila:

napadač → korisnik → administrator

Standardno napadač pokušava da dobije pristup nalogu bilo kog korisnika, a zatim pokušava da poveća nivo svojih privilegija na sistemu. U ovom slučaju, jedna slaba lozinka u sistemu (ili jedna slaba lozinka na celoj mreži) bi bila sasvim dovoljna da se uspešno ostvari prvi korak napadača.

Još jedan problem se javlja kada su u pitanju pokušaji razbijanja lozinke. Na primer, često je pravilo da sistem zaključa nalog korisnika posle tri neuspešna pokušaja. Postavlja se pitanje koliko dugo sistem tada treba da nalog drži zaključanim. Vreme zaključavanja može da bude vremenski ograničeno ili administrator može ručno da ponovo pokrene ovaj servis. Ukoliko je vreme zaključavanja predugačko, napadač može isti napad da sprovede sa nalogima svih ostalih korisnika. Nakon ponovnog pokretanja ovog servisa napadač takođe može da ponovi sličan napad sa drugim lozinkama. Navedena razmatranja ukazuju na to da postoji mogućnost da napadač sprovede napad tipa odbijanja servisa - DOS (eng. *denial of service*). Rešenje za ovaj problem je u kompromisu koji administrator sprovodi podešavajući vreme zaključavanja naloga od minimalnog do maksimalnog. Kratko vreme zaključavanja omogućava ciklične napade, a dugo vreme zaključavanja otvara mogućnost DOS napada.

6.1.4. VERIFIKACIJA LOZINKE

Verifikacija lozinke se odnosi na proveru tačnosti unete lozinke. Da bi računar utvrdio validnost lozinke, mora postojati nešto sa čim je može uporediti. Odnosno, računar mora imati pristup tačnoj lozinci u nekom obliku. Najlošije rešenje je smeštanje sirove lozinke u neku datoteku na disku računara pošto bi ona tada postala laka meta za napadača. Kao i u mnogim drugim problemima koji se javljaju na polju zaštite podataka, rešenje se sastoji u primeni kriptografije. Šifrovanje lozinke npr. simetričnim šifarskim sistemom nije od velike pomoći u ovom slučaju, zato što bi dešifrovanje lozinke zahtevalo prvo pristup simetričnom ključu za dešifrovanje. Na ovaj način se dolazi do identičnog problema. Najsigurniji način je da se umesto samih lozinki na hard disku računara čuvaju njene heš vrednosti. Prednost ovog postupka je što napadač direktnim putem može doći samo do heš vrednosti, ali ne i do same lozinke. Kao što smo razmatrali u poglavlju o heš vrednostima, sve ovo je posledica toga što je heš funkcija jednosmerna. Na primer, ako bi neka lozinka bila $J7@6gh801mY$, na hard disku bi se nalazila njena heš vrednost

$$y = h(J7@6gh801mY)$$

gde je h kriptografska heš funkcija. Kada korisnik želi da se uloguje na sistem, unosi lozinku x , ona se preračunava u heš vrednost i upoređuje sa y koje je smešteno na hard disku računara. Ukoliko je $y = h(x)$ pretpostavlja se da je lozinka tačna i korisniku bi bio odobren pristup.

Naravno, ukoliko bi napadač znao vrednost y , mogao bi da pogađa sve moguće lozinke x dok ne pronađe x za koje je $y = h(x)$ (eng. *forward search attack*) i tada bi lozinka bila razbijena. Pretpostavimo da napadač ima rečnik koji sadrži N najverovatnijih lozinki, na primer

$$d_0, d_1, d_2, \dots, d_{N-1}.$$

U tom slučaju bi mogao da unapred izračuna heš za svaku lozinku u rečniku, tj.

$$y_0 = h(d_0), y_1 = h(d_1), y_2 = h(d_2), \dots, y_{N-1} = h(d_{N-1}).$$

Ukoliko napadač uspe da na računaru dobije pristup datoteci koja sadrži heš vrednosti lozinki, on samo treba da izvrši poređenje sa svojim unapred izračunatim

heš vrednostima. Takođe, napadač može da objavi svoj rečnik uobičajenih lozinki i odgovarajuće heš vrednosti na Internetu, što znatno olakšava posao svakom narednom napadaču. Dakle, i u ovom slučaju je moguć napad sa rečnikom (eng. *dictionary attack*).

Razmatrani napad preko unapred izračunatih heš vrednosti za uobičajene lozinke se može otežati. U tu svrhu se, pre izračunavanja heš vrednosti, lozinki dodaju slučajni brojevi – kriptografska so (eng. *salt*). Dodatak ne mora da bude tajna. Praktično ovaj dodatak se može posmatrati kao inicijalizacioni vektor (IV) kod blokovskih šifarskih sistema u CBC režimu rada. IV vektorom se postiže da se isti blokovi otvorenog teksta šifruju na različit način. Analogno tome, primenom *salt* vrednosti identične lozinke se transformišu u različite heš vrednosti.

Neka je p nova, tek uneta lozinka. Generiše se slučajna vrednost s i zatim se izračunava $y = h(p, s)$. U fajlu sa lozinkama se čuva par (s, y) . Vrednost dodatka s nije ništa tajnija od same heš vrednosti. Nakon toga, da bi se verifikovala uneta lozinka x , izdvaja se par (s, y) iz datoteke sa lozinkama, izračuna se $h(x, s)$ i zatim se rezultati porede sa sačuvanom vrednošću y . Verifikacija lozinke sa dodatkom *salt* vrednošću je podjednako laka kao u slučaju gde nema upotrebe *salt*-a. Međutim, posao napadača je u ovom slučaju znatno otežan. Pretpostavimo da je, na primer, lozinka jednog korisnika kombinovana sa vrednošću s_a , a lozinka drugog sa vrednošću s_b . Da bi pogodio lozinku prvog korisnika uz pomoć rečnika, napadač bi morao da izračuna heš svake reči u rečniku sa vrednošću s_a . Međutim, da bi napao lozinku drugog korisnika, morao bi da ponovo izračuna sve heš vrednosti sa dodatkom s_b . Za fajl sa lozinkama u kojima je broj korisnika N , posao napadača se povećan N puta. Na taj način, datoteka sa unapred izračunatim heš vrednostima postaje beskorisna.

6.1.5. DRUGI PROBLEMI SA LOZINKAMA

Mogućnost razbijanja lozinke jeste jedan od najvećih problema u radu sa lozinkama. Međutim, ima i drugih problema. Danas mnogi korisnici imaju potrebu da koriste veći broj lozinki što dovodi do problema pamćenja tih lozinki. Zbog ovog često se dešava da se korisnici vraćaju na neku prethodnu lozinku ili da imaju istu lozinku koju koriste za više namena. To dovodi do toga da je lozinka onoliko bezbedna koliko i

najnebezbednije mesto na kojem se koristi. Ukoliko bi se pronašla lozinka nekog korisnika, napadač bi mogao da je isproba i na drugim mestima gde dati korisnik ima pristup.

Socijalni inženjering je, takođe, jedna od glavnih briga koje dolaze sa korišćenjem lozinki. Na primer, putem mejlova se često dobijaju poruke kojima se neko predstavlja kao administrator i kome je navodno potrebna vaša lozinka da bi se ispravio problem sa računarom ili softverom na računaru. Po statistici, preko 30% korisnika bi dalo svoju lozinku na ovaj način. Razni hardverski i softverski dodaci (eng. *keyboard logger*), špijunski softver (eng. *spyware*) i druge vrste štetnog softvera su takođe ozbiljna pretnja sigurnosnim sistemima koji se zasnivaju na lozinkama.

Još jedan od glavnih bezbednosnih problema jeste što mnogi korisnici ne menjaju privremene lozinke koje su dobijene od administratora (eng. *default password*). Organizacije sa dobrom bezbednosnom politikom korisnicima nameću pravila upotrebe jakih lozinki. Postoje mnogi popularni softverski alati za razbijanje lozinki koji su dostupni hakerima. S obzirom na to da je jako malo veštine potrebno za korišćenje ovih alata, gotovo svako može danas da razbija lozinke. Bezbednosni problem sa lozinkama su među vodećim bezbednosnim problemima u svetu. U nastavku se razmatra primena biometrije u cilju prevazilaženja problema u radu sa lozinkama.

6.2. BIOMETRIJA

Savremene informacione tehnologije i globalizacija dramatično su promenili današnji svet računarskih mreža, koje se odlikuju visokim stepenom integracije različitih elektronskih servisa. Obzirom da je broj Internet servisa i novih korisnika usluga na Internetu svakodnevno u porastu, povećava se količina i vrednost razmenjenih informacija preko Interneta. Informacije koje se razmenjuju u mreži i memorišu mogu biti kompromitovane ili ukradene ukoliko nisu adekvatno obezbeđene.

U svim tipovima mrežne komunikacije poseban izazov predstavljaju servisi za kontrolu pristupa. Njihova uloga je da zaštite podatke i informacije od neovlašćenog pristupa preko dovoljno sigurnih procedura za proveru identiteta. Tradicionalni sistemi za proveru identiteta baziraju se na lozinkama koje pamtimo ili nečemu što nosimo sa sobom. Obično su to pametne kartice, USB tokeni, mobilni telefoni i td. Ovi

uređaji ne mogu garantovati da se radi o legitimnim korisnicima, jer ne postoji jaka veza između autentifikatora i servisa za autentifikaciju. Problem nastaje usled njihovog kompromitovanja (krađe, gubljenja, kopiranja).

Problem je rešen uvođenjem još jednog atributa, a to je "nešto što jesmo". Biometrija ili bio-informacija se nameće kao moguće rešenje za obezbeđenje čvrste veze između autentifikatora i servisa za autentifikaciju. Biometrijski sistemi su bazirani na fizičkim ili bihevioralnim karakteristikama ljudskih bića kao što je lice, glas, otisak prsta i iris.

Biometrijski podaci imaju potencijalnu prednost da budu jedinstveni identifikatori jedne osobe. U zavisnosti od vrste biometrijskog izvora, mogu da sadrže dovoljnu ili nedovoljnu količinu informacije za određenu primenu u kriptografiji. Za izdvajanje maksimalne količine informacija neophodno je dobro poznavati osobine biometrijskih podataka, okarakterisane prema osobinama i vrsti biometrijskog izvora, kao i tehnologije za precizno očitavanje i izdvajanje konzistentne informacije.

Obzirom na osobine koje poseduju biometrijski podaci, biometrija je postala ozbiljan kandidat za zamenu tradicionalnih metoda na polju autentifikacije i autorizacije pojedinaca. Danas mnoge zemlje razmatraju ili već koriste biometrijske podatke za podizanja sigurnosti u cilju zaštite pojedinca od krađe identiteta ili lažnog predstavljanja.

Ove sisteme je moguće videti u razvijenim zemljama na aerodromskim terminalima. U vreme kada terorizam predstavlja ozbiljnu pretnju u avionskom saobraćaju, mnogi aerodromi su opremljeni sa biometrijskim uređajem za očitavanje biometrije irisa. U bazi tog sistema nalaze se biometrijske informacije o osobama koje se nalaze na crnim listama i kojima je zabranjeno letenje zbog sumnji da mogu izazvati teroristički napad. Kako god, ovo je realnost u današnjem informacionom društvu, ali dobro je što ova tehnologija pronalazi primenu u spašavanju ljudskih života.

Upravljanje lozinkama je najslabija tačka svakog kriptografskog sistema, iz razloga što lozinka može biti ukradena, izgubljena i pogođena, primenom nekih metoda za potpunu pretragu (eng. *Brute force*) lozinki. Imajući u vidu da biometrijski sistem uvek proizvodi odgovor „DA“ ili „NE“, a to je samo jedan bit informacije. Dakle, očigledna je uloga biometrije u konvencionalnom kriptografskom sistemu, a to je samo

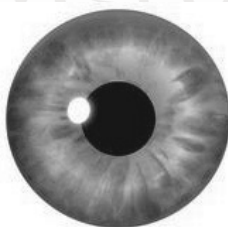
menadžment za upravljanje lozinkama. Nakon faze verifikacije, ukoliko je odgovor DA, sistem otključava ili oslobađa lozinku ili ključ. Ključ mora de se čuva na sigurnom mestu ili uređaju od poverenja.

Interakcija između biometrije i kriptografije predstavlja dve potencijalne komplementarne tehnologije. Postoje različiti biometrijski izvori koji sadrže jedinstvene informacije u vidu ličnih karakteristika jedne osobe. Na ovoj pretpostavci može biti izgrađen napredni sistem za autentifikaciju za prepoznavanje pojedinca sa visokim stepenom sigurnosti.

Primera radi, čvrstom kombinacijom biometrije i kriptografije moguće je imati potencijalno jaku vezu između digitalnog potpisa i osobe koja ga je sačinila sa visokim stepenom sigurnosti. Postoje značajni tehnološki izazovi, kao što su tačnost, pouzdanost, sigurnost podataka, prihvatljivost, cena, interoperabilnost, kao i izazovi vezani za obezbeđenje efikasne zaštite privatnosti. Sve ovo predstavlja uobičajene ranjivosti biometrijskih sistema sa aspekta bezbednosti.

6.2.1. IRIS KAO BIOMETRIJSKI PODATAK

Šara irisa (**Slika 6.2**) ili obojeni deo oka je prilično haotična (neodređena) - slučajna. Prisutan je mali ili gotovo nikakav uticaj genetike. Irisi kod identičnih blizanaca su potpuno različiti. Stanje irisa je stabilno kroz celokupan životni vek, dok to nije slučaj i sa drugim biometrijskim izvorima (biometrijama).



Slika 6.2 Iris oka

U ovom delu analiziramo sve procese na putu od slike oka do iris koda (biometrijskog templejta) u momentu autentifikacije. Kompletan sistem sadrži sledeće faze:

- Segmentaciju;
- Normalizaciju;
- Kodovanje;
- Prepoznavanje.

Faza segmentacije predstavlja izolaciju iris regiona na slici oka. Iris region posmatramo kroz dva kruga, spoljni krug do kapaka i unutrašnji krug do zenice. Spoljašnji krug uglavnom je jednim delom prekriven kopcima i trepavicama. Potrebna je takva tehnika koja će da izoluje ili izuzme ovakve objekte i precizno prepozna kružni deo irisa.

Uspešno prepoznavanje zavisi od kvaliteta slika. Slike u CASIA ²⁴ bazi podataka su visokog kvaliteta, jer nema refleksije zbog upotrebe infra-crvene svetlosti za osvetljavanje. Problem mogu da prave osobe sa tamnim pigmentom, pošto tada imamo smanjenje kontrasta, što prepoznavanje čini težim i nepreciznijim. Faza segmentacije je kritična za uspeh jednog sistema za prepoznavanje, jer loš rezultat se dalje prenosi na ostale faze sistema.

Tehnika koju koristimo za pravilno izdvajanje irisa je Hafov algoritam ili Hafova transformacija. Hafova transformacija predstavlja algoritam koji je dosta primenjiv u kompjuterskoj grafici za prepoznavanje jednostavnih geometrijskih oblika kao što su linije i krugovi. Za ovu potrebu korišćena je kružna Hafova transformacija za automatsku segmentaciju irisa, **Slika 6.3**.

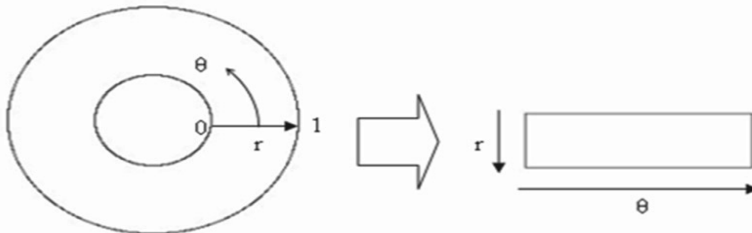


Slika 6.3 Faza segmentacije

²⁴ Standardizovana baza podataka koja sadrži veliki broj slika očiju, baza je namenjena za istraživanja kod biometrijskih sistema. Dostupno na Veb sajtu: <http://biometrics.idealtest.org/dbDetailForUser.do?id=4>

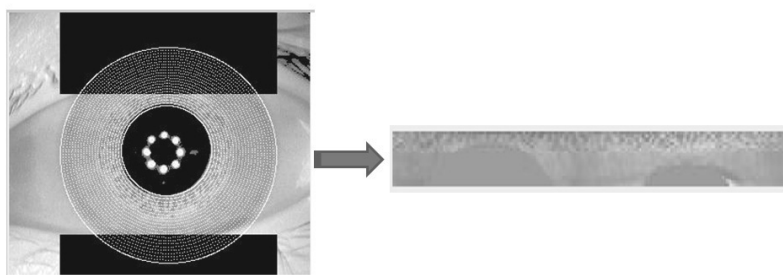
Faza normalizacije ima ulogu transformacije iris regiona u zadate fiksne dimenzije koje su određene unapred za sledeću fazu. Varijacija u dimenziji obično nastaje zbog istezanja zenice prouzrokovano promenom nivoa osvetljenja. Postoje i drugi razlozi, a to su: udaljenost između kamere i oka, rotacija kamere, rotacija oka u očnoj duplji. Procesom normalizacije dobija se iris region koji ima fiksne dimenzije u cilju postizanja karakterističnih funkcija za isto oko na više slika koje su kreirane u različitim uslovima. Pored navedenih problema koje rešava proces normalizacije bitno je i napomenuti da iris nije uvek koncentričan ili prestenastog oblika a jako je bitno da ima stalan radijus.

Model koji se koristi je „Daugmanov Rubber sheet model“, koji ima za cilj da ponovo mapira sve tačke unutar iris regiona na par polarnih koordinata (r, θ) , gde je r na intervalu od $[0,1]$ i θ je ugao $[0, 2\pi]$.



Slika 6.4 Daugmanov Rubber sheet model

Centar oka predstavlja referentnu tačku i radijalni vektori koji prolaze kroz region irisa, **Slika 6.4**. Broj tačaka koji je izabran radijalnim linijama definisan je kao radijalna rezolucija. Broj radijalnih linija definisan je kao ugaona rezolucija. Imajući u vidu da krugovi irisa mogu biti nekoncentrični, potrebno je primeniti metod konfiguracije nad ponovnim mapiranjem tačaka u iris regionu.



Slika 6.5 Faza normalizacije

Normalizacija obezbeđuje kvalitetnije rezultate, ali nismo potpuno u mogućnosti da oporavimo iris regione zbog većih varijacije zenice, **Slika 6.5**.

Faza kodovanja irisa ima za cilj da obezbedi najkonzistentnije informacije prisutne u regionu irisa za precizno prepoznavanje pojedinca. Samo važne karakteristike irisa moraju biti kodovane da bi bilo moguće uspešno prepoznati određenu osobu u fazi prepoznavanja. Većina sistema za ove potrebe koristi vejvlet transformacije za ekstraktovanje ili izdvajanje jedinstvenih karakteristika za generisanje digitalnog biometrijskog templejta.

Biometrijski templejt irisa koji se generiše u fazi kodovanja moraće da poseduje mernu jedinice podudaranja koja daje meru sličnosti između dva irisa. Ovakva vrsta metrike treba da daje jedan opseg vrednosti kada se porede isti irisi i drugi opseg vrednosti kada se porede različiti irisi. Poželjno je da ne postoji poklapanje između ova dva opsega vrednosti tako da možemo sa sigurnošću da kažemo da li su dva iris templejta potekli od istog ili različitog irisa.

Vejvlet transformacija se koristi za razlaganje podataka u komponentama koje se pojavljuju na različitim rezolucijama unutar iris regiona. Vejvleti imaju značajnu prednost u odnosu na Furijeovu transformaciju kod koje je frekvencija podataka lokalizovana.

Broj vejvlet filtera ili kako se naziva drugačije banka filtera je primenjena nad 2D iris regionom. Izlaz primenjene vejvlet funkcije za kodovanje je jedinstvena informacija koja predstavlja stepen neodređenosti nekog pojedinca u odnosu na druge osobe.

Sliku je programski gledano, lakše posmatrati i obrađivati u frekventnom domeu gde lako možemo detektovati nebitne informacije i odstraniti ih iz signala. Za ove potrebe koristi se Gaborov filter koji zbog svojih svojstava precizno razlikuje konzistentne informacije u slici irisa koja sadrži druge šumne podatke. Za odbacivanje šuma iz podataka Gaborov filter je idealan.

Gabor filteri su u stanju da obezbede optimalne zastupljenosti informacije u podatku unutar jednog signala u prostoru i prostorne frekvencije. Jezgro Gaborovog filtera je dvodimenzionalna funkcija koju čini kompleksni sinusoidni talas modelovan Gausovom funkcijom.

Realni deo može se opisati sa funkcijom:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x^2 + \gamma^2 y^2}{2\sigma^2}\right) \cos\left(2\pi \frac{x}{\lambda} + \psi\right)$$

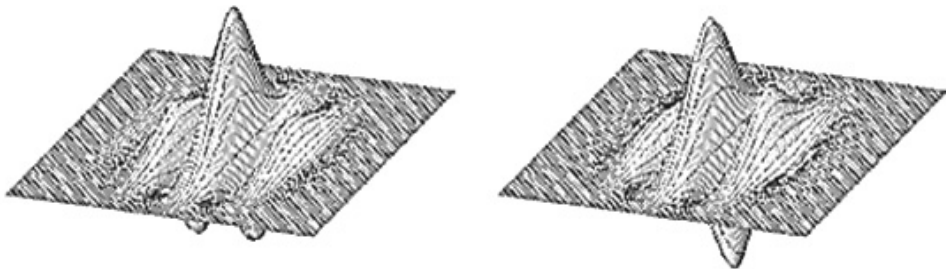
a imaginarni deo:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x^2 + \gamma^2 y^2}{2\sigma^2}\right) \sin\left(2\pi \frac{x}{\lambda} + \psi\right)$$

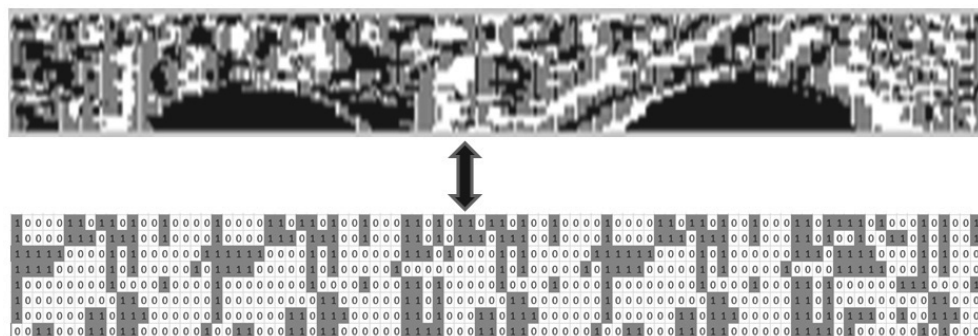
gde su:

$$x = x \cos \theta + y \sin \theta$$

$$y = x \cos \theta + y \sin \theta$$



Slika 6.6 Realni i imaginarni deo impulsivnog odziva



Slika 6.7 Biometrijski templejt irisa

Kao rezultat faze kodovanja (informacioni ekstrakt iz šare irisa), dobija se 256 bajtova iris koda (2048 bitova), Slika 6.7.

Faza prepoznavanja je merenje sličnosti dva irisa koje se zasniva na merenju Hamingovog rastojanja. Računa se $d(x, y)$ na 2048-bitovskom iris kodu. Perfektno poklapanje, što nije realna situacija u praksi, daje rastojanje $d(x, y) = 0$. Za identičan iris, očekivano rastojanje je 0.08. Poklapanje se prihvata ako je rastojanje manje od 0.32.

6.2.2. BIOMETRIJSKA AUTENTIFIKACIJA

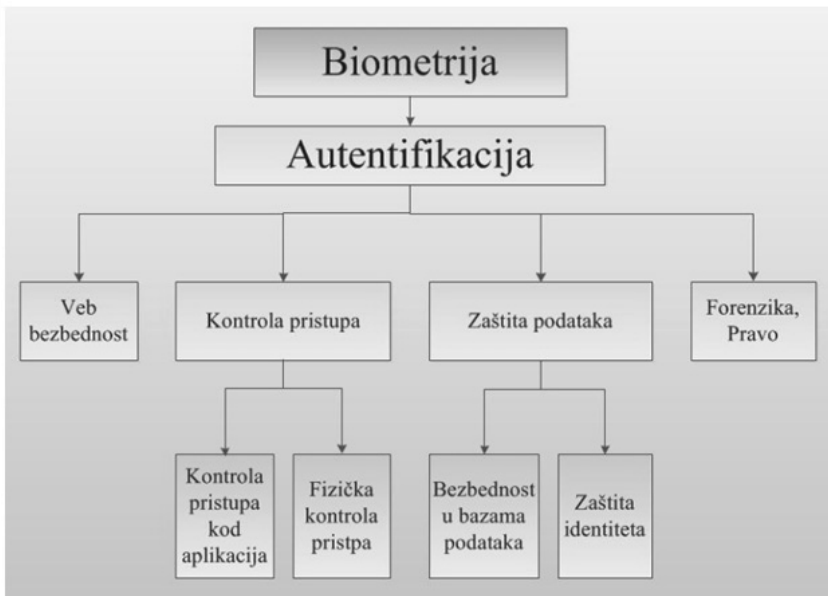
Autentifikacija ili provera korisnika je sastavni deo opšte bezbednosti svih informacionih sistema. Tradicionalno, autorizacija korisnika znači uvođenje korisničkog imena i lozinke, tehnika koja je u upotrebi već decenijama. Napravljene su mnoge inkrementalne promene, izdvajamo neke osnovne: lozinke se više ne šalju u otvorenom tekstu preko mreže i zahtevaju se jače lozinke, ali i dalje, osnovni pristup se nije promenio. Slabosti ovog pristupa su dobro poznate i zbog njih su prisutni problemi na dnevnom nivou.

Za većinu poslovnih okruženja, potencijalne pretnje uključuju:

- napadi na privatnost, u kojima napadači neovlašćeno pristupaju tuđim privatnim informacijama;

- napadi ili krađa poslovne tajne, u kojima pojedinci kompromituju neku organizaciju, u ovakvim slučajevima organizacije mogu da budu izložene ogromnim finansijskim gubicima;
- napadi u cilju zloupotrebe, u kojima pojedinac neovlašćeno manipuliše sa tuđim podacima (čest problem koji se javlja u bankarskim sistemima, preusmeravanje transakcija na druge račune);

Sa druge strane, kriptografski mehanizmi namenjeni za zaštitu poverljivosti, integriteta podataka, zbog kontinualnog tehnološkog razvoja zahtevaju kriptološke ključeve većih dužina koje je potrebno čuvati na sigurnim mestima. Ovakve informacije nije moguće zapamtiti i neophodno je koristiti različite hardverske uređaje za skladištenje kriptoloških ključeva. Takođe, slabosti i ovog pristupa su poznate, ukoliko hardverski medijumi bude kompromitovan jednim od mnogih načina. Biometrija nudi nove, bolje pristupe kod autentifikacije korisnika. Obzirom da su tradicionalne lozinke najslabija karika u mreži i sistemu bezbednosti, biometrija predstavlja novu potencijalnu tehnologiju koja treba da bude brzo usvojena u cilju rešavanja problema na ovim poljima.



Slika 6.8 Primena biometrije

Autentifikacija na bazi biometrije obezbeđuje pogodnu i sigurnu proveru identiteta korisnika kako u lokalnom mrežnom okruženju tako i preko Interneta. U koliko je zahtevan viši nivo bezbednosti na mreži, moguće je koristiti biometriju samostalno ili u kombinaciji sa drugim akreditivima, **Slika 6.8.**

U ovoj eri informacionih tehnologija, popularnost raste za sisteme koji koriste biometrijsku tehnologiju zbog njihovog napretka, praktičnosti i preciznosti u pouzdanom određivanju identiteta korisnika u odnosu na tradicionalne metode. Biometrijski sistemi mogu de se koriste u svrhe provere verodostojnosti, a to uključuje sigurnosne sisteme za kontrolu pristupa kod finansijskih servisa (ATM i banke), vladine institucije, digitalna forenzika i sprovođenje zakona, granična kontrola pristupa i Internet bezbednost koja dalje uključuje sve segmente kriptografske bezbednosti.

Biometrijske sisteme je značajno teže prevariti, nego dosadašnje tradicionalne sisteme na bazi lozinki. Imajući u vidu raznolikost biometrijski izvora, moguće je izvršiti njihovu podelu prema kvalitetu i količini informacije koju mogu da sadrže, na osnovu koje je moguće kasnije generisati biometrijske ključeve za kriptografske svrhe, kao i za bezbednu autentifikaciju.

6.2.3. NEDOSTACI BIOMETRIJSKIH SISTEMA

Razvojem biometrijskih sistema i njihovom prvom tradicionalnom primenom, prepoznati su i prvi nedostaci. Nedostaci su različitog tipa i možemo da ih podelimo u dve osnovne kategorije. Prvu kategoriju čine svi nedostaci koji se odnose na zaštitu privatnosti biometrijskih podataka koji su u direktnoj vezi sa nepromenljivim osobinama biometrijskih podataka (jedinstvenost, nezamenjivost, sadrže šum), dok drugu kategoriju čine nedostaci koji su vezani za sigurnosti sistema kod kojih se biometrijski podaci koriste za autentifikaciju.

Tradicionalni model primene iris biometrije je veoma slab u odnosu na zaštitu privatnosti i nije u skladu sa drugim aspektima bezbednosti. Iako biometrijski templejti mogu biti šifrovani, to nije dovoljno dobro rešenje za bezbednost biometrijskih podataka. Jer, ukoliko su kriptološki ključevi iz nekog razloga kompromitovani, biometrijski podaci su izgubljeni za uvek. Sa druge strane

biometrijski templejti se ne nalaze uvek u šifrovanoj formi zbog faze prepoznavanja u kojoj se vrši provera poklapanja dva biometrijska templejta. U momentu provere poklapanja biometrijski podaci su prisutni u nešifrovanoj formi.

Ukoliko je napad u ovoj fazi uspešno realizovan, napadač će posedovati odgovarajuću originalnu biometriju. Ponovljenim napadom, pristup sistemu je zagarantovan. Nastala situacija može da postane još teža ukoliko je vlasnik izgubljene biometrije koristio biometriju za pristup u više različitih i nepovezanih servisa. Ovo predstavlja ozbiljan bezbednosni problem, a takođe može biti problem zbog pozivanja na odgovornost odgovornih lica.

Važno je imati na umu da postoji velika varijabilnost kod svih tipova biometrijskih izvora. Varijabilnost biometrijskih podataka nastaje u svim fazama obrade biometrijskih podataka. Od momenta očitavanja do generisanja biometrijskih templejta za fazu prepoznavanja. Biometrijski podaci poseduju takve osobine zbog kojih nije moguće dva puta očitati iste biometrijske podatke sa zajedničkog biometrijskog izvora. Tehnika za prepoznavanje identiteta na osnovu lica, sklona je visokom stepenu varijabilnosti zbog različitih uticaja osvetljenja i ugla pomeranja subjekta. To je jedan od razloga zbog koga smo zamoljeni da ne koristimo sliku sa našim osmehom u dokumentima. Slično tome, brojni faktori utiču na izdvajanje konzistentnih i pouzdanih uzoraka otiska prsta. Na osnovu analize osobina različitih biometrija, utvrđeno je da iris oka sadrži najveću količinu informacije i najmanju varijabilnost, sa druge strane odlikuje ga i visok stepen pouzdanosti i konzistentnosti.

Kao posledica varijabilnosti, živi biometrijski uzorci mogu biti u potpunoj suprotnosti sa uskladištenim referentnim podacima. Kod biometrijskih sistema nije moguće postići tačnost od 100 %. Kada biometrijski sistem nije u stanju da potvrdi visok stepen sličnosti u fazi prepoznavanja, prinuđen je da odbaci legitimnog korisnika. Ovakav događaj nazvan je lažno odbacivanje od strane sistema (eng. *False Rejected Rate-FRR*). U ovom slučaju korisnik je prinuđen da ponovi postupak iz početka.

Biometrijski sistemi za prepoznavanje mogu biti dizajnirani tako da se smanji stopa lažnog odbacivanja i da prepoznavanje bude uspešno, što će omogućiti pravilno funkcionisanje sistema. Standardne metode za poboljšanje su kontrolisanja uslova u kojima se očitavaju izvori biometrijskih uzoraka i poboljšavaju matematički algoritmi. Jedan od načina da se to uradi, jeste da se smanji prag ili granica za prepoznavanje.

Postoje problemi sa ovim pristupom jer direktno utičemo na stopu lažnog prihvatanja (eng. *False Acceptance Rate- FAR*), a taj događaj bi značio da je sistem napravio grešku kod identifikacije i prihvatio pogrešnu osobu na bazi tuđeg referentnog biometrijskog podatka.

Imajući u vidu da je moguće smanjenje jednog parametra na račun drugog i obrnuto, neophodno je napraviti kompromis između njih tako da sistem bude prilagođen okruženju. Neka okruženja dozvoljavaju visok FRR, a druga mogu tolerisati visok FAR.

Vrednosti parametara FAR i FRR koje su navedene od strane prodavca sistema često su nepouzdana. Iz tog razloga neophodno je konsultovati nezavisne izvore informacija, kao što su, biometrijska takmičenja u organizaciji američkog Nacionalnog instituta za standarde (NIST). Za većinu biometrijskih sistema vrednost FRR kreće od 0.1 % do 20 %, što znači da će legitimni korisnik biti odbijen najmanje jednom od 1000 puta. FAR kreće od jednog prihvaćenog u 100 (aplikacije niskog stepena sigurnosti), i jednog prihvaćenog u 10.000.000 (aplikacije visokog stepena sigurnosti). Ostali izazovi za biometrijske sisteme su brzina. Sistem mora da bude sposoban da napravi preciznu odluku u realnom vremenu. Sa druge strane kao jedna najbitnija sposobnost biometrijskih sistema je da budu otporni na napade.

6.2.4. ZAŠTITA PRIVATNOST BIOMETRIJSKIH PODATAKA

Zaštita privatnosti podrazumeva da se prava biometrija "nikako" ne čuva originalno u bazama podataka. Da bi se izbeglo čuvanje originalne biometrije, iris digitalni kod ili biometrijski templejt jedne osobe bi bio memorisan u formi rezultata neke neinvertibilne funkcije. Na ovaj način originalna biometrija u vidu slike ili digitalnog biometrijskog templejta nebi bila memorisana u bazama podataka ili na drugim medijima za čuvanje podataka.

Opisana metodologija o kojoj ćemo govoriti u nastavku poglavlja obezbeđuje mogućnost generisanja više biometrijskih templejta na osnovu originalnog iris koda, a ti kodovi međusobno nisu korelisani. Generisani templejti mogu biti sačuvani na istim ili različitim mestima za skladištenje, iako pripadaju jednoj osobi.

Primenom šeme za zaštitu privatnosti biometrije sa mogućim opozivom biometrije "*eng. Cancelable iris biometric*", originalna biometrija neće biti sačuvana na klijentskim računarima, u bazama podataka ili drugim mestima slične namene. Posle neophodnog preprocsiranja biometrije, kao što je proces segmentacije, normalizacije i kodovanja, originalni iris kod biće uništen ili transformisan izabranom neinvertibilnom funkcijom. Dobar primer za neinvertibilnu funkciju je lomljenje staklene čase na stotinu delića, koju nije moguće ponovo sastaviti.

Rezultat ovakve transformacije predstavlja novu lošu kopiju iris koda koji je potpuno siguran jer ne postoji mogućnost za otkrivanje originalne iris biometrije. Naziv ove šeme za primenu biometrije sada je potpuno opravdan jer vrlo lako je moguće generisati novi potpuno drugačiji iris templejt promenom inicijalnih parametara date transformacije. Na ovaj način stekla se mogućnost za generisanje hiljade različitih biometrijskih templejta na osnovu jednog ili dva irisa od jedne osobe. U nastavku ovog poglavlja dat je pregled od nekoliko radova u kojima su predložena slična rešenja za obezbeđenje privatnosti biometrijskih podataka.

U nastavku ovog poglavlja govorićemo o predlozima i rešenjima za zaštitu privatnosti koji su navedeni u naučnim radovima širom sveta.

U radu, (1998), analizirani su metodi za zaštitu privatnosti iris biometrije i metodi za povećanje stepena tačnosti algoritama za prepoznavanje ili poređenje dva iris koda. Koriste se jednosmerne funkcije (*eng. One-way function*) za zaštitu originalnih biometrijskih podataka i drugih autorizacionih informacija. Ove jednosmerne funkcije nisu kao heš funkcije o kojima je bilo reči u prethodnim poglavljima. Dobra stvar kod ovih funkcija je što podaci i dalje čuvaju određene karakteristike koje mogu odgovarati samo jednoj osobi, ukoliko je reč o biometrijskim podacima. Autori ovog rada polaze od pretpostavke da je za razvoj sigurnih aplikacija i sistema neophodno predvideti u dizajnu i implementirati sigurnu korisničku autentifikaciju. Prvo je razmatrana "*Of-line*", a kasnije "*On-line*" korisnička autentifikacija bazirana na biometrijskim sistemima gde se mera za tačnost prepoznavanja identiteta koristi Hamingovo rastojanje (*eng. Hamming distance*).

Prvi model koji je predstavljen u radu, koristi se za povećanje tačnosti algoritama za identifikaciju i autorizaciju u sigurnim aplikacijama tako što čvrsto vezuje (*eng. Binding data*) biometrijski templejt sa autorizacionim informacijama i smešta ih na

pametne kartice. U ovom slučaju dobijamo sistem sa visokim performansama, jer sistem u momentu verifikacije proverava originalnu biometriju sa podacima na pametnoj kartici.

Drugi model u istom radu, specijalno je dizajniran i koristi se za zaštitu privatnosti biometrijskih podataka. Biometrijski podaci su enkapsulirani u autorizacionim informacijama i ne zahtevaju sigurne hardverske tokene. Kompletно istraživanje je sprovedeno u cilju olakšavanja praktične primene biometrijskih sistema u sigurnim aplikacijama. Administrator sistema za autentifikaciju generiše javni i privatni ključ. Javni ključ je smešten u biometrijskom čitaču. Na pametnoj kartici nalaze se informacije: ime osobe, osnovne informacije o izdavaču ATM-a, kao i digitalni potpis uvezanih biometrijskih podataka sa slučajnom binarnom sekvencom. Za proces autentifikacije neophodno je da osoba priloži pametnu karticu, nakon toga biometrijski templejt se generiše nezavisno od korisnika. Zatim se konstrukcijom kodova za ispravljanje grešaka (*eng. Error-Correcting-Code*) dobija ispravna biometrija i u poslednjoj fazi vrši se verifikacija. Digitalni potpis je verifikovan javnim ključem sistem administratora. Autori u radu na ovaj način rešavaju pitanje privatnosti. U predloženom rešenju neophodno je koristiti jednosmerne funkcije (*eng. Hash*) ako nismo sigurni da dolazi do curenja informacija i potpisane autentifikacione podatke koji su inkorporirani sa slučajnom bitnom sekvencom koja je kasnije kodovana i uvezana sa biometrijom.

U radu (1999) autori predlažu novu šemu poznatu pod nazivom "*eng. Fuzzy commitment scheme*". Ova šema podrazumeva dva metoda, metod za prikrivanje (*eng. Concealing*) i metod za uvezivanje podataka (*eng. Binding*). U odnosu na konvencionalnu kriptografiju, za otključavanje ovakvog tipa nastalog podatka neophodan je jedinstven identifikator, slično kao kriptološki ključ za dešifrovanje. Karakteristike predložene šeme su vrlo primenjive kod aplikacija kao što su biometrijski sistemi za autentifikaciju, čiji su podaci izloženi dejstvu različitih šumova. Predložena šema od autora (Ari Juels, Martin Wattenberg, 1999 str. 4, 5) je tolerantna na nastale greške. Sposobna da zaštiti biometrijske podatke na sličan način kao i konvencionalna kriptografija, ili kao jednosmerne funkcije koje se koriste za zaštitu alfanumeričkih lozinki.

U radu (2001), analizirane su prednosti biometrije sa aspekta novo predloženog rešenja kod kontrole pristupa sa naprednim sistemima za autentifikaciju. Identifikovane su slabosti kod tradicionalnih biometrijskih sistema i predstavljeno je novo rešenje kojim bi se eliminisale neke od ovih slabosti. Za ilustraciju autori (N. K. Ratha, J. H. Connell, R. M. Bolle, 2001) koriste otisak prsta kao biometrijski podatak. Analiza je obuhvatila i druge biometrijske metode. Osnovni problem tradicionalnih biometrijskih sistema je nemogućnost zamene originalne biometrije sa novom, ukoliko dođe do kompromitovanja biometrijskih podataka.

U cilju ublažavanja ovog problema autori (N. K. Ratha, J. H. Connell, R. M. Bolle, 2001 str. 629) predlažu koncept pod nazivom (*eng. Cancelable biometrics*). On se sastoji od namerne, ali ponovljive distorzije (izobličenja) biometrijskog signala na osnovu izabrane transformacije. Na ovaj način biometrijski signal je izobličen u svim sferama primene, za upis u sistem i svaku proveru identiteta. Sa ovim pristupom svaka nova instanca može koristiti različitu transformaciju, čineći kros-podudaranje nemogućim. Osim toga ako jedna instanca biometrijskog templejta bude kompromitovana, vrlo lako se može kreirati nova sa transformacionom funkcijom, slično kao kada imamo potpuno novu osobu. Za proces distorzije izabrana je jedna neinvertibilna funkcija. Na ovaj način, čak i ako je napadaču poznat dizajn algoritma transformacione funkcije i transformisani biometrijski podatak, napadač nije u mogućnosti da rekonstruiše originalnu biometriju. Metod distorzije, kao transformacione funkcije je moguće primeniti u domenu bilo kog signala. Biometrijski signal se može direktno transformisati posle akvizicije ili kao i obično posle procesa ekstrakcije iris koda. U idealnom slučaju neinvertibilna funkcija treba da bude takva da čak i sa poznavanjem originalne biometrije nije moguće povratiti jednu ili više biometrijskih instanci. Primeri su dati od strane autora (N. K. Ratha, J. H. Connell, R. M. Bolle str. 629-630). U prvom primeru za biometriju lica (*eng. Face biometric*) primenjena je distorzija nad originalnom slikom nakon čega se distorzovana slika nije podudarala sa originalnom. U drugom primeru za biometriju prsta (*eng. Fingerprint biometrics*) korišćen je jedan od metoda za skremblovanje signala čime je postignut identičan rezultat kao i u prethodnom primeru. Metod koji je korišćen za distorziju signala opisan je u radu (1992).

Pored neinvertibilnih funkcija u transformacionom domenu, mogu biti primenjene i šifarske funkcije (*eng. Encryption*) za zaštitu privatnosti biometrijskih podataka.

Tehnika koju predlažu autori u ovom radu, drugačija je od šifarskih funkcija, jer ukoliko budu kompromitovani neki parametri sistema, kao što je kriptološki ključ, biometrija je izgubljena zauvek. Dok ova mogućnost ne postoji kod neinvertibilnih funkcija. Razlika je što transformisani signal zadržava iste osobine koje su bitne za sisteme autentifikacije, dok te osobine nisu sadržane u šifrovanim biometrijskim podacima. Parametar distorzije u transformacionom domenu moguće je kontrolisati od strane servisa ili još bolje od strane osobe koja se identifikuje. U cilju podizanja najvišeg stepena privatnosti, predloženo je čuvanje parametra transformacije (*eng. State*) na pametnim karticama koje su dodatno zaštićene sa PIN kodom. Ukoliko osoba izgubi pametnu karticu ili kartica bude ukradena, to neće bitno uticati na privatnost korisnika i bezbednost sistema. Na ovaj način je ublažena zabrinutost za kršenje privatnosti upotrebom sistema za autentifikaciju na bazi biometrijskih podataka.

U radu (2008), autori predlažu novi metod za zaštitu privatnosti kod iris biometrije. Za ekstrakciju iris koda ili generisanje biometrijskog templejta koriste 1D log-Gabor filter, slično implementaciji (Libor Masek, 2003). U prvom koraku koriste razmotanu sliku irisa kojoj dodaju eksternu slučajno generisanu informaciju za proces distorzije signala. Transformaciju koju koriste je neinvertibilna i opoziva.

U radu (2009), autori u cilju prevazilaženja problema neopozivosti biometrijskih podataka tj. zaštite privatnosti biometrijskih podataka, predložili su šemu sa dva faktora zaštite koristeći iris biometriju i lozinku. U ovoj šemi, svakom korisniku se određuje zaseban ključ za mešanje iris koda koji predstavlja jedinstvenu permutaciju. Drugi faktor ove šeme predstavlja uvođenje kodova za detekciju i ispravljanje grešaka (*eng. Error-Correcting-Code*) u cilju smanjenja varijabilnosti u biometrijskim podacima. Šema mešanja kod različitih irisa u fazi poređenja povećava Hamingovo rastojanje ostavljajući originalno rastojanje nepromenjeno, dok ECC smanjuje Hamingovo rastojanje u fazi poređenja istih irisa u većoj meri nego u fazi poređenja različitih irisa. Primenom šeme mešanja postignuti su bolji rezultati koji se manifestuju kroz Hamingovo rastojanje u poređenju istih i različitih irisa. Ključ koji se koristi za mešanje, zaštićen je lozinkom koja ovaj sistem čini biometrijski opozivovim. Na ovaj način biometrijski podaci su u zaštićenoj formi koja štiti privatnost biometrijskih podataka.

U radu iz 2010. godine., autori su predstavili novu šemu za zaštitu iris biometrije. Slično kao i u drugim postojećim radovima, predložena transformacija za iris kod je neinvertibilna, čak i ako je poznata napadaču. Predloženi metod sadrži tri faze. U prvom koraku generisan je iris kod, u drugom koraku ekstraktovani su konzistentni bitovi irisa i u trećem koraku primenjen je algoritam koji pruža zaštitu privatnosti i opoziv iris biometrije.

Na osnovu navoda povezanih radova sa ovom oblasti, koji se bave pronalaženjem rešenja za probleme na polju zaštite privatnosti biometrijskih podataka ili (*eng. Cancelable Biometric*), možemo da izvedemo sledeće zaključke:

Većina autora u radu predlaže jednosmerne funkcije ili transformacije. Poseduju osobinu neinvertibilnosti preko koje je obezbeđena mogućnost opoziva biometrijskih podataka kao i ponovno generisanje biometrijskih podataka u svrhu autentifikacije. Na ovaj način su zamenjeni tradicionalni biometrijski sistemi.

Pojedine transformacione funkcije zadržavaju u podacima osnovne karakteristike biometrijskih podataka, koje pozitivno utiču na Hamingovo rastojanje, metod koji koristimo za prepoznavanja ili poređenje biometrijskih podataka.

Prepoznate su određene slabosti kod transformacija sa lozinkama, koje predstavljaju jedinstvenu "*permutaciju*". Problem je poistovećen sa kompromitovanjem kriptološkog ključa u kriptografiji.

Pored primene transformacija, određeni autori poistovećuju biometrijske podatke sa komunikacionim kanalima sa šumom. Ovakva ideja uvodi primenu kodova za ispravljanje grešaka (*eng. Error-Correcting-Code*), metod preuzet iz oblasti telekomunikacija.

7. VIZUELNA KRIPTOGRAFIJA I SKRIVANJE INFORMACIJA

Još jedna od srodnih kriptografskih tema koje se ne uklapaju direktno u do sada pomenuta poglavlja. Prvo ćemo razmotriti jedan od sistema za deljenje tajni. To je konceptualno jednostavna procedura koja može da bude od koristi kada je potrebno podeliti tajnu između korisnika. Na kraju poglavlja, ukratko ćemo izložiti temu informatičkog skrivanja, gde je osnovni cilj sakrivanje informacija u drugim podacima, kao što je ugrađivanje neke informacije u sliku. Ako samo pošiljalac i primalac znaju da su informacije skrivene u podacima, drugi učesnici mogu samo da sumnjaju da je došlo do komunikacije.

Skrivanje informacija je velika oblast i savremeni, način komunikacije za uspešno izbegavanje napada čovek u sredini. Postoje priče da terorističke organizacije primenjuju ovaj način komunikacije za pripremu terorističkih napada u svetu.

7.1. VIZUELNA KRIPTOGRAFIJA

Vizuelna kriptografija je specijalna tehnika šifrovanja koja omogućava skrivanje informacija (tajnih poruka) u slici na takav način da je čovek može dešifrovati bez upotrebe računara ili bilo kakvih drugih izračunavanja, ako se koristi ispravan ključ. Ovu tehniku su predložili Noni Noar i Adi Shamir (RSA) 1994. godine na EUROCRYPT²⁵ konferenciji posvećenoj kriptografskim istraživanjima. Ova tehnika ima takve karakteristike koje je čine poželjnom za upotrebu u finansijskim transakcijama preko Interneta.

Za predajnu i prijemnu stranu koja treba da šifruje i dešifruje skrivenu poruku, vizuelna kriptografija predstavlja jednostavnu tehniku i siguran sistem, nalik na One-time pad. Vizuelna kriptografija našla je svoju primenu u domenu Vodenih pečata, čija je osnovna uloga zaštita autorskih prava. U nekim državama ova tehnika se

²⁵ Godišnja međunarodna konferencija o teoriji i primeni kriptografskih tehnika. Dostupno na Veb sajtu: <http://www.iacr.org/conferences/eurocrypt2013/>

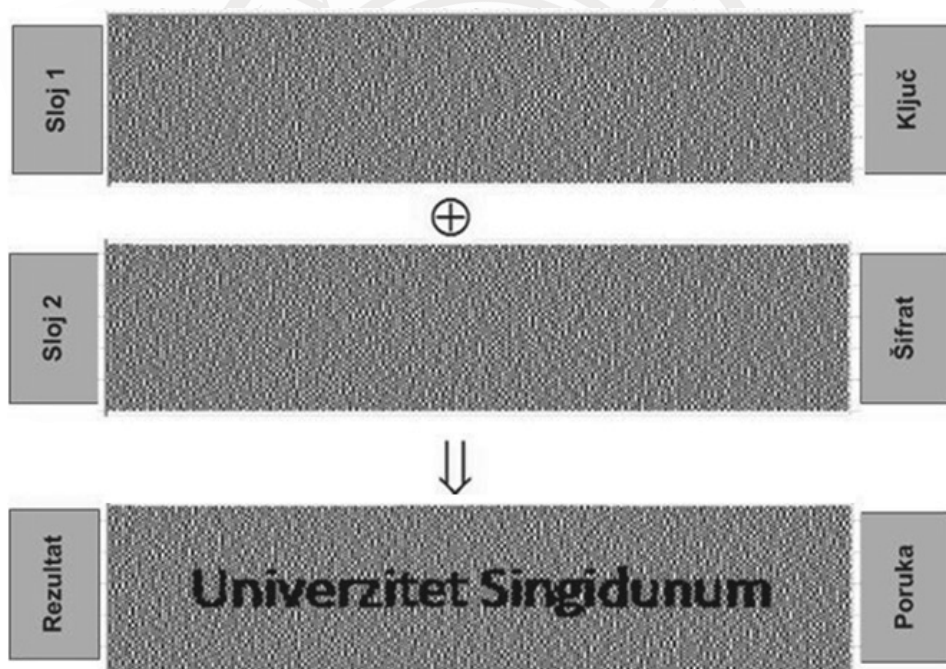
koristila za proveru glasačkih listića u cilju zaštite od falsifikovanih glasačkih listića. U drugim situacijama koristili su je za proveru obveznica.

U primeru **Slika 7.1**, možete posmatrati dva sloja, koja kad se pravilno poklope ili poravnaju, generišu skrivenu tajnu informaciju (poruku).

Odšampajte ovaj primer na transparentnoj foliji ili ukoliko ste vešti da pokušate u nekom od programa za obradu fotografije. Ukoliko ste pravilno postupali sa ovom tehnikom dobićete skrivenu poruku "Univerzitet Singidunum".

42. PRIMER:

Slika 7.1, vizuelna kriptografija sa dva sloja.



Slika 7.1 Primer sa vizuelnom kriptografijom

Slika 7.1, oba sloja su iste rezolucije i poravnanjem jedne preko druge, jasno se može pročitati skrivena informacija. Vizuelna kriptografija radi sa slojevima identičnih ili dodatnih komplementarnih piksela i predstavlja vizuelnu varijantu XOR-ovanja.

Obzirom da se radi o slučajnom generisanju vrednosti sa kojima su predstavljeni pikseli na slici, ova tehnika se može posmatrati kao One-time-pad šifra. Vizuelna kriptografija nasleđuje sigurnosne osobine One-time-pad šifre. Osnovno je da ne postoji mogućnost otkrivanja skrivene informacije iz slike, čime je obezbeđena perfektna sigurnost.

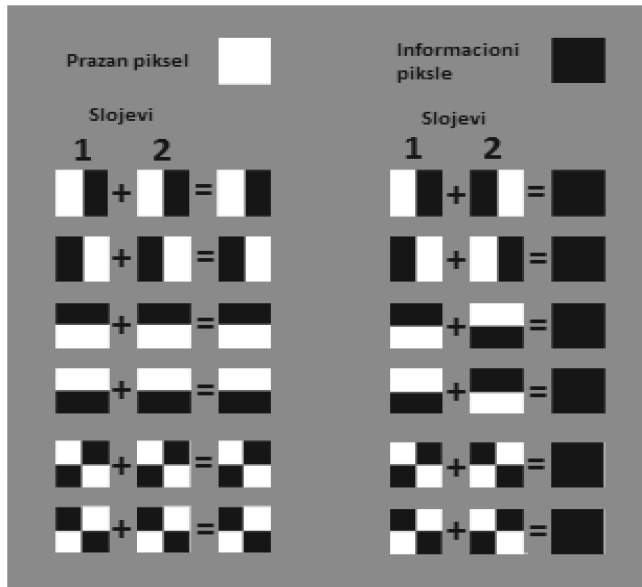
Vizuelna kriptografija koristi dve transparentne slike, gde svaka slika predstavlja zaseban sloj. Jedna slika sadrži slučajno generisane vrednosti piksela (Gausov beli šum) i ta slika predstavlja ključ, dok druga slika sadrži skrivenu informaciju. Obe slike su iste rezolucije i poravnanjem jedne preko druge jasno se može pročitati skrivena informacija.

Tajna slika	Deo 1	Deo 2	Rezultat
			
			
			
			

Slika 7.2 Princip rada algoritma za vizuelnu kriptografiju

Svaki piksel se deli na manje blokove, **Slika 7.2**. Uvek postoji isti broj belih (transparentnih) i crnih blokova. Deljenjem piksela na dva dela dobijamo jedan beli i jedan crni blok, a deljenjem na četiri jednaka dela dobijamo dva bela i dva crna bloka.

Slika 7.3, pikseli koji su podeljeni na četiri dela mogu se nalaziti u šest različitih stanja. Piksel na sloju 1 ima određeno stanje, a piksel u sloju dva može da ima isto ili suprotno stanje. Ako je piksel na sloju 2 isti kao na sloju 1, preklapanjem će se dobiti pola belih i pola crnih piksela. Ali, ako je on suprotan od piksela na sloju 1, preklapanjem će se dobiti crni piksel koji predstavlja informaciju (informaciju o tajnoj poruci).



Slika 7.3 Podela piksela

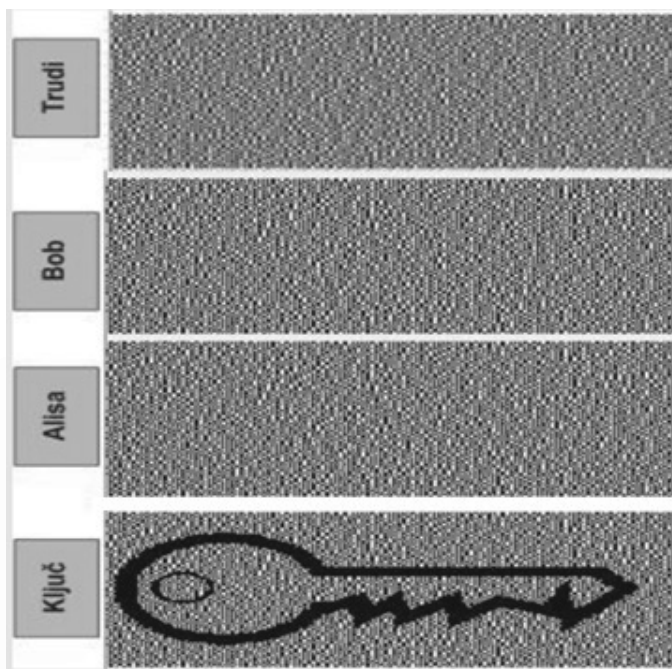
Sloj 1 poseduje piksele čije su vrednosti određene na slučajan način, slično načinu za generisanje ključa kod *One-time-pad* šifre i može da ima jedno od šest mogućih stanja. Sloj 2 je isti kao i sloj 1, isključivo sadrži fiksne crne piksele koji su nosioci informacije u fazi preklapanja. Oni imaju uvek stanje različito od stanja na sloju 1.

Primena ove tehnike skrivanja informacija može da se koristi u svrhu deljenja tajni, konkretno u bankarstvu (kad je zahtevano deljenje informacije o ključu na više korisnika ili više manjih ključeva koji sami ne mogu da otključaju sef) i u protokolima za autentifikaciju u cilju zaštite biometrijskih podataka.

Na primer, Alisa, Bob i Trudi su zaposleni u istoj banci. Nije poželjno da Trudi sama pristupa sefu. Svi troje poseduju delove informacije o ključu sa kojim mogu da regenerišu ključ sa kojim otključavaju sef. Sistem je konfigurisan tako da zahteva sva tri dela informacije o ključu. U momentu otključavanja sefa Alisa, Bob i Trudi prilažu delove ključa, nakon toga sistem generiše jedan glavni ključ sa kojim može da se otključa sef. Prednost ove tehnike je što omogućava podelu ključa na više zaposlenih u banci, što podiže bezbednost celokupnog sistema.

43. PRIMER:

Slika 7.4, Primer za vizuelnu kriptografiju sa tri sloja.



Slika 7.4 Primena vizuelne kriptografije u bankarstvu

Slika 7.4, dat je primer deljenja glavnog ključa na tri informativna dela koja poseduju tri korisnika. (Alisa, Bob i Trudi).

Tehnika za vizuelnu kriptografiju našla je svoju primenu u sigurnim komunikacijama. Na primer, Alisa će Bobu unapred poslati 1 ili 2 sloja generisana na slučajan način. Kada Alisa bude imala poruku za Boba, generisaće sloj 2 na osnovu sloja 1 i poslati Bobu. Bob ima sloj 1 koji je Alisa unapred poslala, preklopiće ili poravnati ova dva sloja bez upotrebe računara i otkriti tajnu poruku od Alise. Ukoliko je Trudi presrela sloj koji sadrži informaciju, za nju će biti nemoguće da otkrije poruku.

7.2. STEGANOGRAFIJA

Steganografija je tehnika skrivanja tajnih poruka na takav način da niko osim predajne i prijemne strane nije svestan postojanja komunikacije. Skrivanje poruka se temelji na prerušavanju poruke unutar slike, filmova i teksta. Osnovna prednost steganografije u odnosu na kriptografiju je činjenica da poruke ne privlače pažnju na sebe. Možemo da kažemo da je steganografijom moguće izbeći napad „čovjek u sredini“, obzirom da napadač nije svestan postojanja komunikacije u nekom komunikacionom kanalu.

Steganografija ima prednost u zemljama u kojima su kriptografske tehnike za šifrovanje poruka zabranjene. Kriptografskom zaštitom zaštićene su samo poruke, dok se za steganografiju može reći da štiti i poruke i strane koje učestvuju u komunikaciji. Steganografija podrazumeva i skrivanje poruka unutar računarskih fajlova i tokova podataka. Ukoliko se radi o digitalnoj steganografiji, poruka može biti skrivena unutar dokumenata, slike ili filma. Film je multimedijalna datoteka i kao takva zbog svoje veličine je idealna za steganografsku komunikaciju. Njihova veličina im daje prednost kada je u pitanju osetljivost na suptilne promene.

Primena steganografije danas u svetu je raznolika. Spajanjem slike sa tekstualnim datotekama možemo obezbediti poverljivost važnih informacija, čuvajući ih od sabotaze, krađe ili neovlašćenog gledanja. Nažalost, steganografija ima i nelegalnu svrhu u špijuniranju i u terorizmu (BBC je u više navrata navodio da su teroristi za sigurnu i tajnu komunikaciju koristili neke od steganografskih tehnika). Najznačajnija i najčešća upotreba je u digitalnim vodenim pečatima koji se koriste za zaštitu autorskih prava.

7.2.1. ISTORIJSKI OSVRT NA NASTANAK STEGANOGRAFIJE

Upotreba steganografije potiče od davnina (440 g. PNE), Herodot je naveo dva primera steganografije u svom delu „Herodotova istorija“. U delu je navedeno kako je Demaratus poslao upozorenje o predstojećem napadu na Grčku zapisivanjem poruke na drveni kalup voštane ploče za pisanje, pre izlivanja voska. Voštane ploče su se dosta koristile kao papir jer je bila omogućena ponovna upotreba površine za pisanje.

Drugi drevni primer upotrebe steganografije je priča o Histaeusu i njegovom vernom robu na čiju glavu je Histiaeus ispisao poruku i sačekao da mu kosa izraste. Kasnije je rob tu poruku neprimećeno preneo kako bi pokrenuo ustanak protiv Persijanaca.

7.2.2. MODERNA STEGANOGRAFIJA

Moderna steganografija razvija se od 1985. godine nastankom personalnih računara i sa primenom računara na do tada poznate klasične steganografske tehnike. Početni razvoj je bio spor, ali sa vremenom razvile su se mnoge aplikacije za steganografiju.

Digitalna (moderna) kriptografija primenjuje sledeće tehnike:

- Skrivanje poruka u najmanje značajnim bitima slike;
- Skrivanje poruke u postojećem šifratu. Ideja je da prvo šifrujemo poruku, a zatim njen sadržaj rasporedimo i zamenimo sa blokovima nekog drugog šifrata;
- Preslikavanje statističkih osobina jedne datoteke na drugu datoteku, kako bi prva ličila na drugu datoteku;
- Namerno, neprimetno usporavanje računarske mreže, u kojoj vreme može predstavljati simboliku za prenos neke poruke;
- Namerna promena redosleda paketa unutar računarske mreže;
- Blog steganografija koristi izdvojene poruke, gde delove poruke ostavlja kao komentare na raznim blog stranicama.

Danas postoje različiti programi koji implementiraju mnoge steganografske tehnike za skrivanje informacija, poput: SteganPEG, StegFS, OpenPuff, MP3Stego, Hcovert, VoVoIP i drugi. Sa druge strane razvijeno je i mnogo programa za stegoanalizu, čiji je zadatak da prepoznaju steganografiju u komunikaciji i da unište poruku.

SteganPEG omogućava skrivanje bilo kog digitalnog sadržaja unutar JPEG formata. Na primer, moguće je sakriti jednu ili više slike unutar druge slike. Prednost ovog programa je što koristi vrlo popularan i rasprostranjen JPEG format za čuvanje slika.

StegFS je visoko-sofisticirani alat razvijen za Linux operativni sistem koji uz šifrovanje poruka koristi i steganografske metode. Skrivanje vrši na takav način na nije moguće dokazati postojanje drugih poruka.

OpenPuff je steganografski alat koji omogućava skrivanje poruka nad više fajlova unutar jednog direktorijuma (Foldera), čime je praktično omogućeno skrivanje velike količine poruka. Služi se kriptografskim metodama, oponaša razne šumove, što otežava posao stegoanalizi.

mp3stego program koji koristi steganografsku tehniku za skrivanje poruke u mp3 formatu u toku kompresije. Skrivenne poruke su šifrovane i u toj formi skrivene.

Hcovert je možda najinteresantniji steganografski program koji koristi GET zahteve HTTP protokola za slanje skrivenih poruka određenom serveru. Tehniku je moguće primeniti u fazi slanja i fazi primanja poruke.

VoVoIP je eksperimentalni alat koji demonstrira novu pretnju u VoIP komunikaciji i to u obliku tajnog komunikacionog kanala. Dodatni kanal je postignut kompresijom tajnih podataka u klasičnu govornu komunikaciju, baziran na PCM-u. Prednost ove tehnike je nemogućnost prisluškivanja i otkrivanja.

7.2.3. DIGITALNI VODENI PEČAT

Prelaskom iz analognog u digitalni domen, kao i obrada podataka na digitalnom domenu dovela je do nemogućnosti razlikovanja originalnih sadržaja od kopija. Sve ovo je otežalo zaštitu autorskih prava i vlasništva. Nastala je potreba za mogućnošću razlikovanja originalnog ili izvornog podatka od lažnih kopija istog. U ovom momentu na scenu je stupila tehnika za digitalno označavanje podataka. Digitalno označavanje podataka je pojam koji predstavlja tehniku ubacivanja digitalnog pečata u digitalni dokument sa namerom kasnijeg detektovanja ili ponovnog uzimanja žiga.

Područje digitalnih vodenih pečata je još uvek slabo istraženo, iako danas postoje algoritmi za zaštitu bilo koje vrste digitalnog sadržaja (dokumenti, slike, audio i video materijali, mape i računarski programi).

Možda interesantan primer za digitalni vodeni pečat jeste događaj iz 2012. godine iako ne postoje čvrsti dokazi, da je na ovaj način Google zaštitio svoje mape od kopiranja. Prema tvrdnjama australijskih naučnika, ostrvo na Južnom Pacifiku, koje se može naći na mapama Google Earth-a ne postoji. Na tom mestu dubina vode je prema brodskim sondama 1.400 metara. U pitanju je veliki komad zemlje koji se na Google-ovim mapama naziva Peskovitim ostrvom, a nalazi se između Australije i Nove Kaledonije. Naučnici sa Univerziteta u Sidneju koji su otišli na koordinate gde bi trebalo da se nalazi ostrvo nisu pronašli ništa osim okeana. Ovu grešku prijavili su Google-u, ali za sada nije izvršena prepravka na mapama, **Slika 7.5**.



Slika 7.5 Nepostojeće ostrvo na Google mapi

Digitalni vodeni pečat može se ubaciti uz pomoć algoritma za kodovanje u neki digitalni sadržaj ili sa istim algoritmom izvaditi iz označenog sadržaja i na taj naći se jednoznačno određuje vlasnik i integritet digitalnog sadržaja.

Prema vizuelnoj percepciji razlikuju se četiri tipa vodenih pečata. Prvi vrsta je vidljivi vodeni pečat, obično je to jasno vidljiv logo neke oznake. Ovo možemo često videti na digitalnim PDF formularima neke ustanove. Druga vrsta je robusni i vizuelno nevidljiv, ali ga dekodeer vrlo lako detektuje i uklanja, sa druge strane otporan je na napade. Ova vrsta pečata se koristi za zaštitu autorskih prava multimedijalnih sadržaja. Treća vrsta je lomljivi vodeni pečat koji je vizuelno nevidljiv, ali može se detektovati i nije otporan na napade. Četvrta vrsta je dvostruki vodeni pečat koji predstavlja kombinaciju vidljivog i nevidljivog vodenog pečata.

Digitalni vodeni pečati imaju primenu u dokazivanju autentičnosti podataka. Za ovu svrhu se najčešće koriste lomljivi vodeni pečati. Vrlo značajna uloga digitalnih

vodenih pečata je pri praćenju emitovanog programa na televiziji, radiju i sličnim uređajima.

Na primer, agencije za marketing koriste algoritme za merenje dužine trajanja reklamnih programa na televiziji. Imajmo na umu da je cena jedne sekunde reklame malo bogatstvo u vreme nekog velikog sportskog događaja i da je jako bitno imati informaciju o dužini trajanja reklamnog programa. U ovom slučaju vodeni pečat je utisnut u sadržaj reklamnog materijala, dekodirer treba da detektuje vodeni pečat i tako odredi vreme trajanja reklame na televiziji ili radiju. Na ovaj način oglašivači su sigurni da su platili svaku sekundu emitovane reklame.

Postoje određene primene u kojima dodatna informacija o digitalnom sadržaju treba da sadrži informacije o krajnjem korisniku, a ne o vlasniku sadržaja. Problem identifikacije curenja informacija može se rešiti sa distribucijom loših kopija na osnovu originala, gde između svih kopija postoje neznatne razlike.

Na primer, ovakva upotreba digitalnog vodenog pečata našla je primenu u filmskoj industriji kod distribucije filmova različitim bioskopima. Svaki bioskop dobija svoju kopiju filma i ako se na tržištu pojavi piratska verzija filma, lako se može otkriti iz kog je bioskopa potekla.

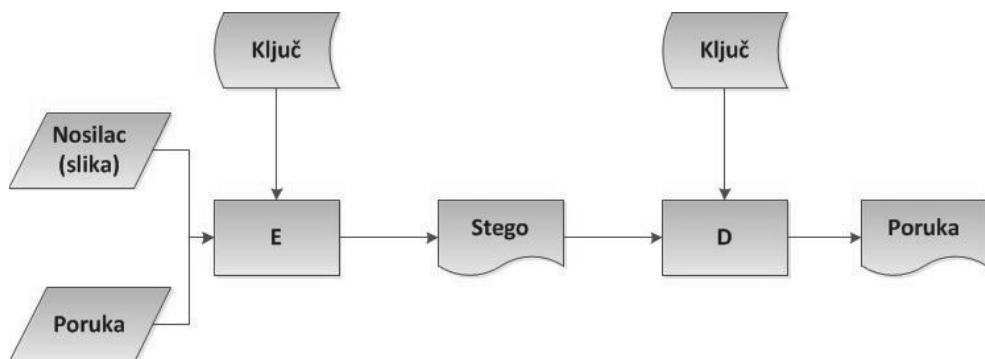
Sigurnost vodenih pečata govori koliko su oni otporni na različite spoljne napade. Napadi mogu biti različite prirode, a neki od njih su JPEG kompresija, razne geometrijske transformacije (menjanje veličine, odrezivanje, rotacije) za podizanje kvaliteta slike.

7.2.4. LSB SUPSTITUCIJA

LSB (eng. *least significant bit*) je supstitucija jedne od najčešće korišćenih steganografskih tehnika. Ova tehnika ima dobre performanse kad se primenjuje na slikama. Veliku količinu informacija moguće je sakriti zanemarljivim uticajem na datoteku nosioca skrivene informacije.

LSB supstitucija (**Slika 7.6**) uzima najmanje značajne bitove iz izabrane datoteke i menja ih sa bitovima tajne poruke. Dakle, ovi bitovi se menjaju bitovima tajne

poruke. LSB supstitucija se lako primjenjuje na slike velike rezolucije sa mnogo detalja. Za metodu zamene bitova koristi se sekvencijalni algoritam (RC4).



Slika 7.6 LSB steganografska šema

- E - steganografska funkcija za ugrađivanje poruke
- D - steganografska funkcija za izdvajanje poruke
- Nosilac - medijum unutar koga se skriva tajna poruka
- Poruka - tajna poruka koja treba biti skrivena
- Ključ - steganografski ključ, parametra funkcije E i D
- Stego – steganografska datoteka

Da bi prijemna strana uspešno rekonstruisala poruku, primalac mora da zna poskup najmanje značajnih bitova u koje je poruka skrivena. Tehnikom obrnutom od ugrađivanja tajne poruke, izdvajaju se najmanje značajni bitovi, slažu se odgovarajućim redosledom i tako se dobija poruka. Kada bismo tajnu poruku ovom metodom skrivali u 2 ili 3 najmanje značajna bita elemenata prikrivene datoteke, još uvek bi ljudsko oko teško moglo primetiti razliku u odnosu na original.

Ukoliko koristimo 24-bitni BMP format slike dobijamo dovoljno prostora za skrivanje podataka. Svaki piksel ili boja piksela je predstavljena kao kombinacija crvene, zelene i plave boje. Svaka od ovih boja predstavljena je jednim bajtom. Kombinacija jednog bajta crvene, zelene i plave daje 3 bajta (24 bita) koji predstavljaju 1 piksel. Svaki piksel može uzeti jednu od 256 nijansi.

LSB je najjednostavnija metoda iako ne pruža mnogo sigurnosti. LSB metoda se odlikuje sa velikim kapacitetom koji se može iskoristiti za skrivanje poruke ili ugrađivanje digitalnog vodenog pečata.

Negativna strana steganografije leži u činjenici da je za skrivanje poruke potrebno koristiti objekat nosilac koji uglavnom nosi veću količinu podataka nego sama tajna poruka. Tehnike stegoanalize nastoje da otkriju postojanje skrivene poruke unutar nekorisne informacije. Posebno su od značaja tehnike stegoanalize koje kao podlogu koriste statističke testove. Takve tehnike, osim uloge detektovanja skrivene poruke, često mogu da odrede približnu veličinu poruke.

44. PRIMER:

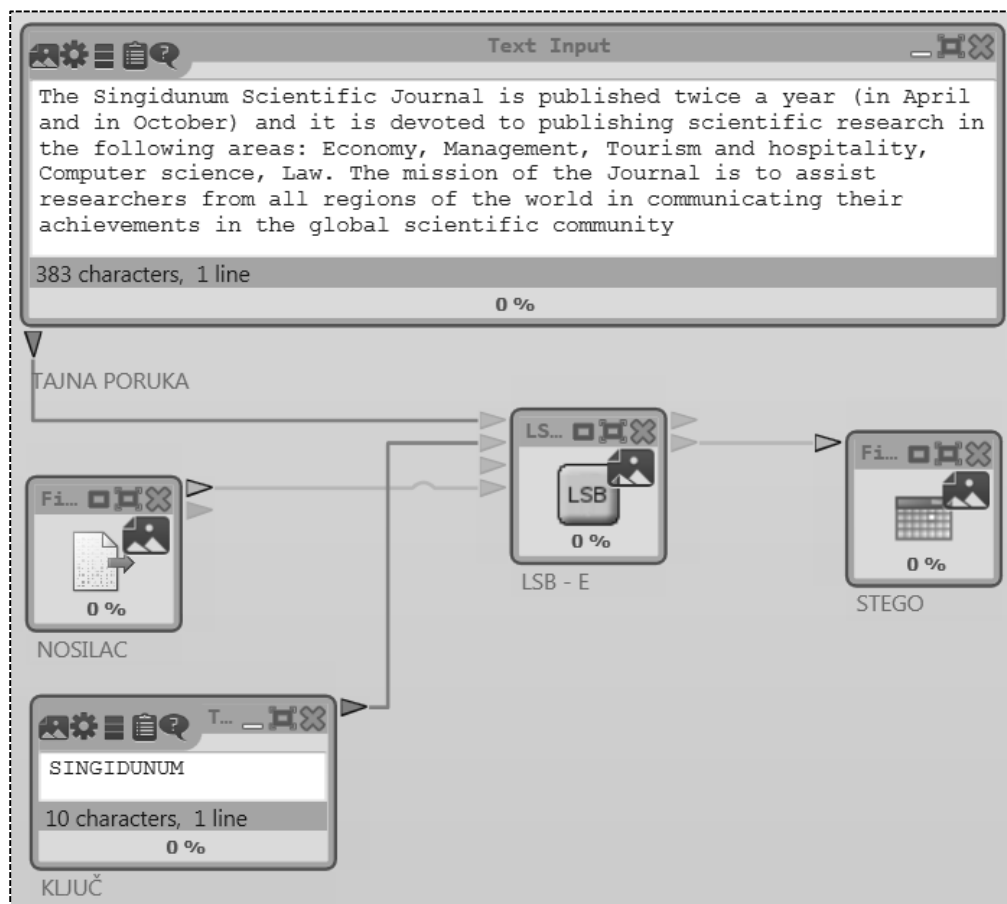
U sledećem primeru korišćemo LSB steganografsku tehniku u Cryptool projektu. Za podatak nosilac izabrana je, **Slika 7.7**.



Slika 7.7 Podatak nosilac

U objekat podatak nosilac (slika u ovom slučaju) ugrađićemo sledeću tajnu poruku:

„The Singidunum Scientific Journal is published twice a year (in April and in October) and it is devoted to publishing scientific research in the following areas: Economy, Management, Tourism and hospitality, Computer science, Law. The mission of the Journal is to assist researchers from all regions of the world in communicating their achievements in the global scientific community.“, Slika 7.8.



Slika 7.8 LSB supstitucija (kreiranje stego objekta)

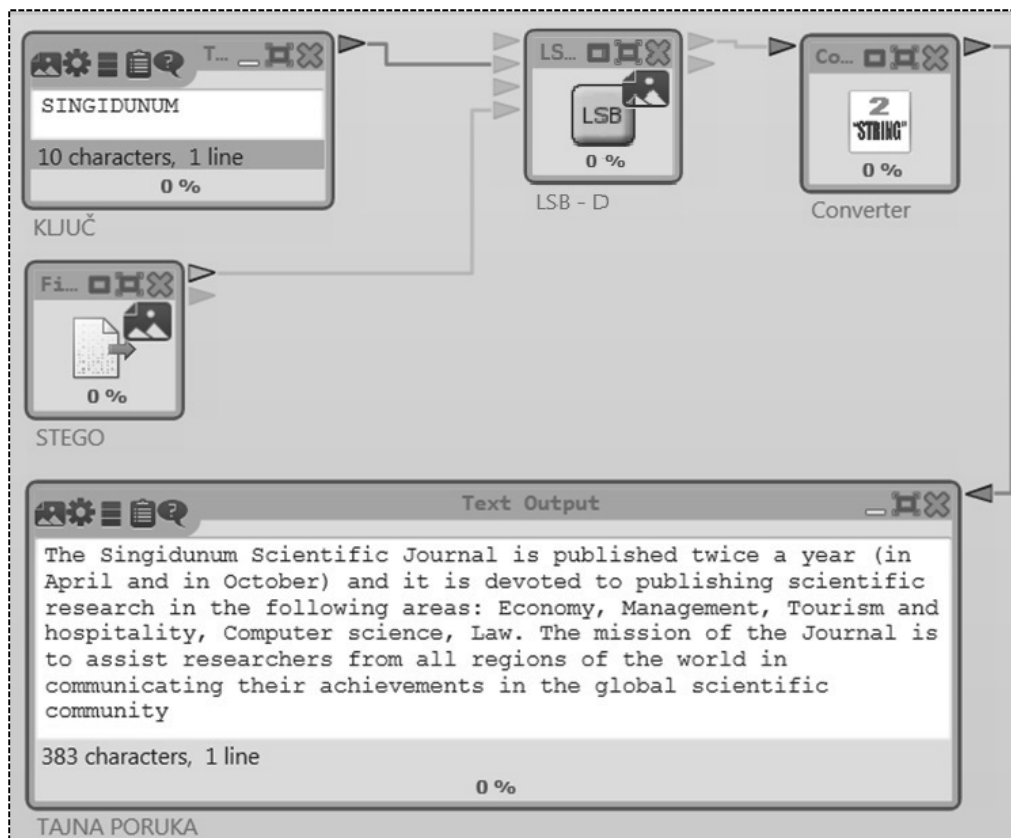
Rezultat steganografski objekat (Slika 7.9):



Slika 7.9 Stego podatak koji sadrži tajnu poruku

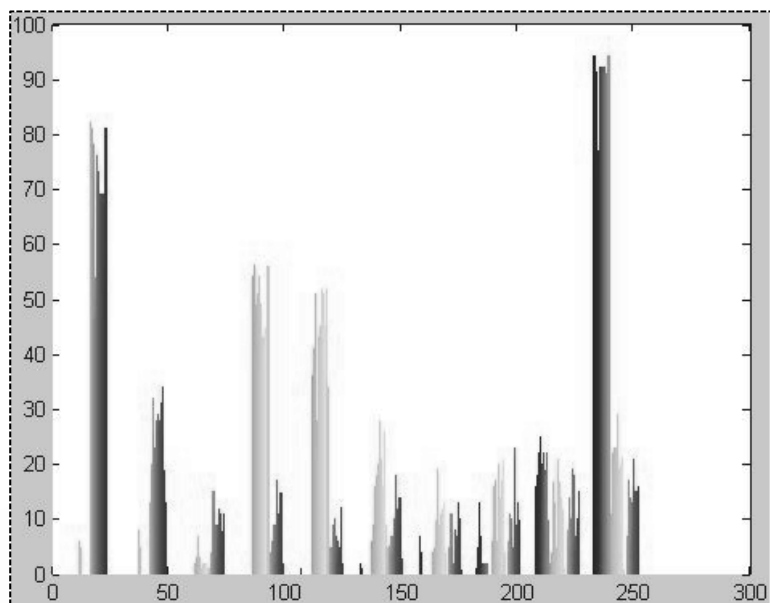
45. PRIMER:

Slika 7.10, u sledećem primeru dekovali smo steganografski podatak i uspehi da vratimo tajnu poruku.

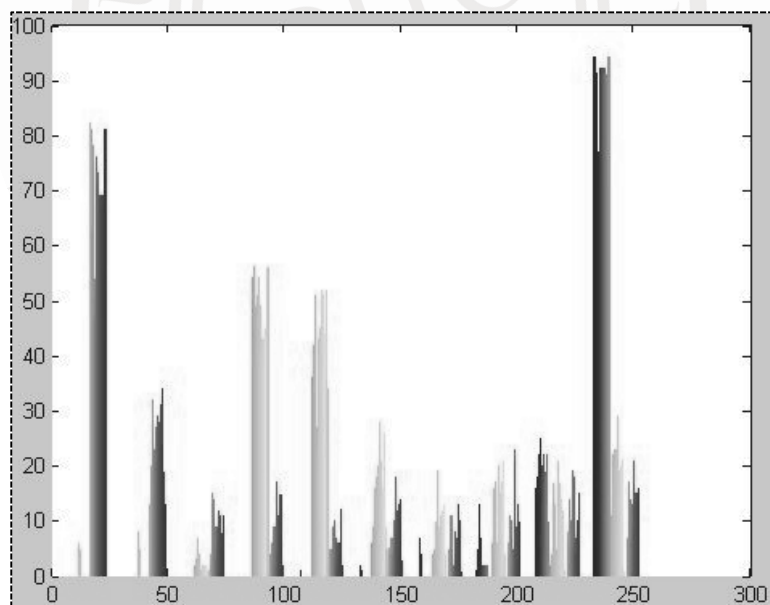


Slika 7.10 LSB supstitucija (izdvajanje tajne poruke iz stego objekta)

Primera radi u sledećem primeru smo odredili histogram originalne slike (**Slika 7.11**) i stego slike (**Slika 7.12**). Dobijeni rezultati su prikazani ispod. Između ova dva histograma primetna je neznatna razlika koju je moguće odrediti tek ukoliko posedujemo originalnu sliku ili podatak nosilac.



Slika 7.11 Histogram originalne slike



Slika 7.12 Histogram stego slike

7.3. GENERISANJE SLUČAJNIH BROJEVA

Potreba za slučajnim i pseudoslučajnim brojevima javila se zbog velike primene u kriptografskim sistemima. Zajedničko za sve kriptografske sisteme je potreba za kriptološkim ključem koji je generisan na slučajan način. Mnogi kriptografski protokoli, protokoli za autentifikaciju, RSA algoritmi za generisanje digitalnog potpisa koriste na ulazu slučajne ili pseudoslučajne vrednosti.

Za dizajn generatora potrebno je razmotriti sve parametre izvora slučajnosti koji kvalitativno utiču na generatore potpuno slučajnih vrednosti koje direktno imaju uticaj na sigurnost kriptografskih sistema, kao i sisteme za modelovanje i simulacije različitih aplikacija.

Slučajna binarna sekvenca može se tumačiti kao rezultat bacanja novčića koji je označen sa „pismo“ ili „glava“ gde je verovatnoća ishoda (0 i 1) približno 0,5. Bacanje novčića predstavlja savršen generator slučajnih vrednosti jer „1“ i „0“ će biti nasumično raspodeljene (ujednačena raspodela). Svi elementi sekvenci generišu se nezavisno jedna od druge (statistička nezavisnost), a vrednosti sledećih sekvenci se ne mogu predvideti, bez obzira koliko je elemenata prethodno generisano.

7.3.1. VRSTE GENERATORA

Postoje dva osnovna pristupa za generisanje slučajnih brojeva korišćenjem računara. Pseudoslučajni generatori brojeva PRNG (eng. *Pseudorandom Number Generators*) i pravi slučajni generatori brojeva TRNG (eng. *True Random Number Generator*). Svaki od navedenih pristupa za generisanje slučajnih brojeva ima svoje prednosti i mane.

Karakteristike TRNG su poprilično drugačije od PRNG, TRNG su neefikasni u odnosu na PRNG. Potrebno im je znatno više vremena da proizvedu potrebnu količinu slučajnih brojeva. Slučajni brojevi dobijeni preko TRNG su potpuno nedeterministički, što znači da dati niz slučajnih brojeva nije moguće reprodukovati. Bitna karakteristika ovih generatora slučajnosti je izbegnuta perioda datih slučajnih nizova.

Pseudoslučajni generatori su algoritmi koji na osnovu početnih (inicijalnih) stanja daju izlazni niz pseudoslučajnih brojeva. Početni uslov je zadovoljen ukoliko je ulaz u gen-

erator „istinski“ slučajan. Početni „istinski“ slučajan niz se dobija primenom generatora „istinski“ slučajnih brojeva TRNG i predstavlja ključ k odnosno inicijalno stanje. Generisane pseudoslučajne sekvence su znatno duže od ključa i smatraju se pouzdanim ukoliko zadovoljavaju određene statističke testove. Determinističke karakteristike generatora se ogledaju u mogućnosti predviđanja brojeva u nizu, odakle i sami generatori nose naziv pseudoslučajnih. Reprodukovanje izlaznih nizova je moguće samo uz poznavanje početnih vrednosti, odnosno početnih uslova. Zanimljivo je da se pseudoslučajnih brojevi ponekad pokazuju uspešnijim od slučajnih brojeva dobijenih iz fizičkih izvora. Pesudorandom generatori PRNG pokazuju dobre karakteristike uz veoma oprezan odabir početnih stanja što dovodi do prikrivanja determinističkih osobina koje se mogu upotrebiti za kriptanalizu.

Generatori pseudoslučajnih brojeva su brži i lako ostvarivi. Kada je reč o kriptografiji, velika pažnja se poklanja odabiru samog generatora, njegovim osobinama i potencijalnim statističkim slabostima koje se mogu upotrebiti prilikom kriptanalitičkih napada. Kada je reč o statističkim testovima pomoću kojih se utvrđuju osobine generisanog niza brojeva, nijedan generator ne poseduje stoprocentno zadovoljavajuće parametre. Loši generatori padaju na prostim testovima, kao što dobri generatori padaju na komplikovanim testovima. Pošto ne postoji univerzalna kolekcija testova čiji prolaz garantuje da je dati generator potpuno pouzdan, generatori pseudoslučajnih brojeva se konstruišu na osnovu ozbiljne matematičke analize njihovih strukturalnih osobina.

U nastavku poglavlja daje se opis poželjnih statističkih svojstava koja se očekuju od generisanog niza brojeva:

- verovatnoća pojave jednog broja treba biti jednaka za sve brojeve u prostoru u kome se generišu slučajni brojevi. Ova osobina se odnosi na ceo generisani niz, ali i za njegov podniz;
- korelacija između generisanih brojeva pod niza ne sme da postoji, odnosno ne sme da postoji zavisnost između podnizova;
- generator je dobro konstruisan ukoliko je broj stanja kroz koja on prolazi prilično velik. Ukoliko se generator u jednom trenutku nađe u stanju u kojem je već bio, dolazi do ponavljanja nekog već generisanog podniza. Perioda dobro konfigurisanog generatora mora biti što veća.

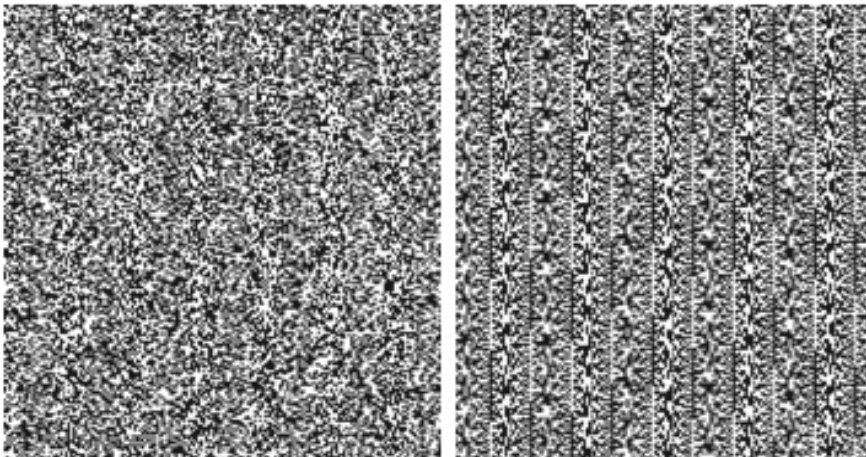
U nastavku rada biće opisani neki od generatora pseudoslučajnih brojeva i metoda za testiranje koji se najčešće koriste.

7.3.2. RAZLIKA PRNG / TRNG

Navedene karakteristike čine TRNG podobnijim za veći skup aplikacija u odnosu na PRNG koji je nepodoban za aplikacije kao što su kriptografski sistemi za šifrovanje podataka i igre na sreću. Nasuprot tome, loša efikasnost i nedeterministička priroda TRNG čini ih manje pogodnim za simulacije i modelovanje aplikacija koje često zahtevaju više podataka nego što je moguće proizvesti sa TRNG.

Tabela 13 Karakteristike dva tipa proizvoljnog broja generatora.

Karakteristike	PING	TRNG
Efikasnost	da	ne
Determinizam	da	ne
Periodičnost	da	ne



Slika 7.13 Vizuelna uporedna analiza rezultata TRNG (slika levo) i PRNG (slika desno)

7.3.3. GENERATORI SLUČAJNIH BROJEVA

Entropija izvora obično se sastoji od neke fizičke veličine, kao što su šum, vreme izvršavanja korisničkih procesa (npr. taster miša ili tastature) ili kvantni efekti u poluprovodniku. Pri generisanju slučajnih sekvenci razne kombinacije ovih ulaza se mogu koristiti. Rezultat TRNG generatora može se koristiti kao slučajan broj ili može predstavljati stanje pseudoslučajnih generatora PRNG.

Da bi se rezultat koristio direktno (bez dalje obrade), izlaz iz TRNG mora da zadovolji stroge kriterijume slučajnosti merene statističkim testovima kako bi se utvrdilo da se fizički izvori TRNG ulaza pojavljuju nasumično. Na primer, fizički izvor kao što je elektronski šum može da sadrži superpozicije redovne strukture, kao što su talasi ili druge periodične pojave, koji mogu biti slučajni, ali primenom statističkih testova oni nisu posledica slučajnosti.

Za kriptografske svrhe, izlaz TRNG-a treba da bude nepredvidiv. Međutim, neki fizički izvori kao što je vremenski vektor mogu biti prilično predvidivi. Ovakvi problemi rešavaju se kombinovanjem različitih izvora i dobijeni rezultat se daje na ulaz TRNG-a. Međutim postoji mogućnost da dobijeni rezultati TRNG-a ne mogu proći neke od statističkih testova. Za ispitivanje kvaliteta generisanog niza koristi se mnoštvo testova. Međutim ne postoji jedan univerzalni test na osnovu koga se može tvrditi da je određen niz generisanih brojeva potpuno slučajan i da poseduje sve osobine slučajnosti. Na osnovu rezultata testova je moguće zaključiti da li niz pokazuje neke od osobina koje bi pokazivao niz generisan od strane idealnog generatora (prirodne pojave). Nasuprot tome, na osnovu rezultata se može sigurno zaključiti da neki niz brojeva nije slučajan, odnosno da generator radi loše.

Prilikom ispitivanja slučajnih nizova potrebno je odrediti statističku vrednost pomoću koje se određeni rezultat prihvata ili odbacuje. Takođe je potrebno odrediti kritičnu vrednost za koju se ne očekuje da će biti promašena. Kritična vrednost bi predstavljala veoma malu verovatnoću realizacije određenog događaja. Na osnovu pomenutih vrednosti se određuju granice preko kojih se ne sme ići ili ispod kojih se dobijena statistička vrednost ne sme spuštati. U skladu sa tim generisani nizovi se smatraju slučajnim ili suprotno generatori se smatraju neispravnim. Mera sigurnosti sa kojom se generator prihvata ili ne, naziva se nivo značaja. Prevelika odnosno,

premala vrednost ovog parametra može dovesti do odbacivanja niza. Svaki ispit za određivanje kvaliteta generatora meri neku od poželjnih karakteristika slučajnih brojeva.

Nakon serije ispitivanja moguće je zaključiti da ispitivani generator poseduje željena svojstva i da bi kao takav mogao biti dobar generator. Sa druge strane, moguće je zaključiti da ispitivani generator nije odgovarajući. Ovakve zaključne tvrdnje moguće je donositi sa 99% sigurnošću. Obično se u ispitima upotrebljavaju dve osnovne raspodele:

- Očekivana raspodela posmatranih vrednosti u nekom prostoru
- Očekivana raspodela učestalosti pojave nekih događaja

U nastavku poglavlja navešćemo nekoliko testova za generatore pseudoslučajnih brojeva koje preporučuje NIST (*National Institute of Standards and Technology*):

- Maurerov univerzalni statistički test – Ispituje mogućnosti kompresije niza bez gubitka informacije. Za niz koji se može značajno kompresovati smatra se da nije niz slučajnih bitova zbog činjenice da je kompresija niza efikasnija ukoliko niz pokazuje periodična svojstva.
- Ispitivanje preklapajućih uzoraka – posmatrana karakteristika u ovom testu je učestalost pojave svih mogućih n -bitnih uzoraka uz preklapanje u celom ispitivanom nizu. Test bi trebalo da otkrije da li je broj pojava preklapajućih uzoraka približno jednak broju koji se očekuje za niz slučajnih brojeva.
- Ispitivanje približne entropije – u ovom testu posmatra se takođe učestalost pojave svih mogućih preklapajućih n -bitnih uzoraka u nizu. Cilj je poređenje učestalosti preklapajućih blokova sa očekivanim rezultatima.
- Ispitivanje učestalosti u nizu – u ovom testu ispituje se odnos jedinica i nula u nizu bitova. Cilj je uočavanje jednakosti pojavljivanja ove dve vrednosti.
- Ispitivanje učestalosti u bloku – posmatra se odnos jedinica i nula u n -bitnim blokovima. Cilj predstavlja uočavanje jednakosti broja jedinica i nula u svakom n -bitnom bloku.
- Ispitivanje uzastopnih ponavljanja istih bitova u nizu – posmatrana karakteristika u ovom testu je ukupan broj uzastopnih ponavljanja jednog broja u nizu.

- Ispitivanje najdužeg uzastopnog ponavljanja jedinica u bloku – u ovom testu posmatra se najduže uzastopno ponavljanje jedinica u n-bitnim blokovima. Svrha je određivanje da li se dužina najdužeg uzastopnog ponavljanja poklapa sa dužinom koja bi se očekivala u nizu slučajnih brojeva.

7.3.4. NAKNADNO PROCESIRANJE EKSTRAKTA ENTROPIJE IZVORA

Metoda kombinovanja različitih izvora (entropija izvora) i post digitalne obrade, rezultuje sa binarnim sekvencama koje poseduju osobine nepredvidivosti, čak i ako izvor bitova poseduje male verovatnoće nepredvidivosti.

Postdigitalna obrada je proces stvaranja pouzdano nepredvidivih sekvenci iz nepouzdan nepredvidivog izvora sekvenci ili predstavlja poboljšanje entropije izvora po bitu. Sam proces je neophodan zbog nemogućnosti kontinualne analize izvora i dobijenih sekvenci u realnom vremenu. Na ovaj način ispravljamo loše osobine TRNG-a.

Jedan od glavnih metoda u procesu destilacije je statističko uklanjanje grešaka koje se bazira na algoritmu mapiranja prelaza (eng. *transition mapping*). Analiziraju se po dva bita zajedno i u slučaju da postoji prelaz između dva bita (01 - 10) samo jedan od njih prihvata se kao slučajni. Ako nema prelaza (00 - 11) bitovi se odbacuju kao neslučajni. Von Neuman (Von Neumann) je osmislio kompletan algoritam i on u potpunosti uklanja pojavu dugih nizova 0 ili 1.

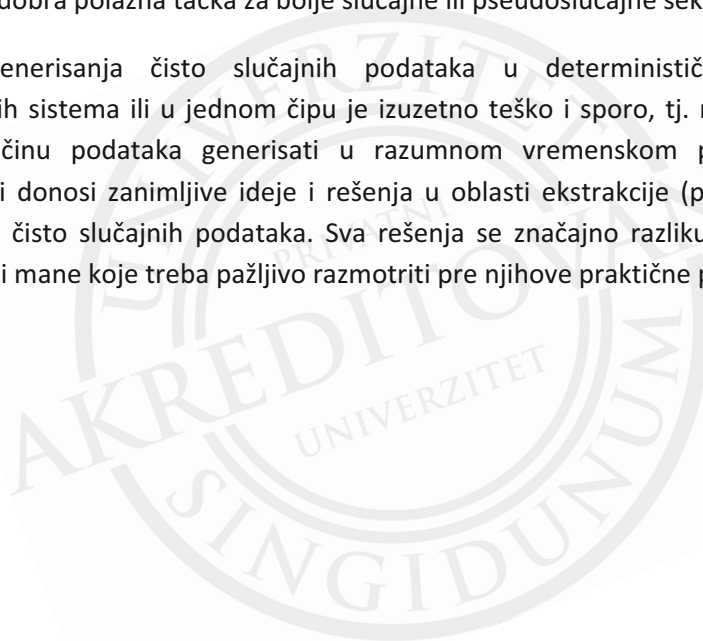
Drugi način je paritet niza (eng. *stream parity*) koji se realizuje na sledeći način. Nizovi se dele na parove zatim se određuje paritet svakog para. Parovi kod kojih je paritet različit odbacuje se, a kod kojih je isti zadržava se samo prvi bit. Bitovi koji nisu odbačeni čine niz koji ulazi u sledeći korak. U zavisnosti od kvaliteta dobijene kompresije entropije izvora određuje se stopa eksponencijalnog skraćanja polaznog niza.

Upotrebom procesa postdigitalne obrade, poboljšava se entropija izvora što može biti dobra polazna tačka za bolje slučajne ili pseudoslučajne sekvence. Ovakvom vrstom digitalne post obrade podataka obezbeđujemo uniformnost, izlaz prevazilazi određene korelacije ili statističku zavisnost izazvanu od hardverskog izvora čisto

slučajnih podataka. Na ovaj način obezbeđujemo i analiziramo bezbedne, efikasne i jedinstvene primere slučajnih brojeva.

U današnjem svetu savremeni računari i razna bezbednosna rešenja oslanjaju se na kriptografske mehanizme. Generisanje visokog kvaliteta slučajnosti je vitalni (a možda i najteži) korak mnogih kriptografskih operacija koji imaju veliki uticaj na sigurnost i zbog toga je naglašena potreba za pažljivim dizajnom generatora slučajnosti. Navedeni su svi osnovni bezbednosni aspekti distribuiranih slučajnih sekvenci. Upotrebom procesa postdigitalne obrade, povećavamo entropiju izvora što može biti dobra polazna tačka za bolje slučajne ili pseudoslučajne sekvence.

Proces generisanja čisto slučajnih podataka u determinističkom okruženju računarskih sistema ili u jednom čipu je izuzetno teško i sporo, tj. moguće je samo malu količinu podataka generisati u razumnom vremenskom periodu. Teorija slučajnosti donosi zanimljive ideje i rešenja u oblasti ekstrakcije (post procesiranja podataka) čisto slučajnih podataka. Sva rešenja se značajno razlikuju i imaju svoje prednosti i mane koje treba pažljivo razmotriti pre njihove praktične primene.



8. BIBLIOGRAFIJA

Aiden A. Bruen Cryptography, information theory, and error-correction [Book]. - San Diego : WILEY-INTERSCIENCE , 2005.

Alexander W. Dent, Chris J. Mitchell User's Guide to Cryptography and Standards, Computer Security Series [Book]. - Boston : Artech House, 2005.

Alfred J. Menezes, Paul C. van Oorschot, Scot A. Vanstone Handbook of Applied Cryptography [Book]. - New Jersey : CRC Press, 2001.

Ari Juels, Martin Wattenberg A fuzzy commitment scheme [Conference] // CCS '99 Proceedings of the 6th ACM conference on Computer and communications security. - New York : ACM, 1999. - pp. 28-36.

CrypTool [Online] // <http://www.cryptool.org/en/>. - 2013.

Davida, G.I., Yair Frankel, Brian J. Matt On Enabling Secure Applications Through Off-line Biometric Identification [Conference]. - Oakland, CA : Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on, 1998. - p. 148.

Dragan Pleskonjić Sigurnost računarskih sistema i mreža [Book]. - 2007 : Mikro Knjiga, Beograd.

Fred Piper and Sean Murphy Cryptography: A Very Short Introduction [Book]. - London : Oxford University Press, 2002.

Jinyu Zuo, Nalini K. Ratha, and Jonathan H. Connell Cancelable Iris Biometric [Conference] // In proceeding of: Pattern Recognition.. - [s.l.] : ICPR 2008. 19th International Conference, 2008.

Libor Masek Peter Kovesi MATLAB Source Code for a Biometric Identification System Based on Iris Patterns [Conference]. - 2003.

Milan Milosavljević, Saša Adamović, Irina Branović, Dejan Živković Teaching interactive cryptography: the case for CrypTool [Conference] // Communication and Energy Systems and Technologies - ICEST. - [s.l.] : ICEST, 2011. - p. 46.

Mladen Veinović, Aleksandar Jevremović Računarske mreže [Book]. - Beograd : Univerzitet Singidunum, 2010.

Moni Naor and Adi Shamir Visual Cryptography [Conference]. - [s.l.] : EUROCRYPT , 1994.

N. K. Ratha, J. H. Connell, R. M. Bolle Enhancing security and privacy in biometrics-based authentication systems [Journal]. - [s.l.] : IBM SYSTEMS JOURNAL, 2001. - 3 : Vol. 40. - 0018-8670.

Niels Ferguson, Bruce Schneier, Tadayoshi Kohno Cryptography Engineering, Design, Principles and Practical Application [Book]. - Canada : John Wiley & Sons, 2010.

Osama Ouda, Norimichi Tsumura, Toshiya Nakaguchi Tokenless Cancelable Biometrics Scheme for Protecting IrisCodes [Conference] // Pattern Recognition (ICPR), 2010 20th International Conference on. - Istanbul : 2010 International Conference on Pattern Recognition, 2010. - pp. 882- 885.

Sanjay Kanade, Dijana Petrovska-Delacretaz, Bernadette Dorizzi Cancelable Iris Biometrics and Using Error Correcting Codes to Reduce [Conference]. - Miami, FL : Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 2009. - pp. 120-127. - 978-1-4244-3991-1.

Schneier Bruce Applied Cryptography: Protocols, Algorithms, and Source Code in C [Book]. - Canada : John Wiley & Sons, 1996. - 2nd Edition.

Stamp Mark Information Security [Book]. - New Jersey : John Wiley & Sons, 2011. - 2nd Edition.

T. Beier, S. Neely Feature-Based Image Metamorphosis [Conference]. - New York : [s.n.], 1992. - pp. 35–42.

DODATAK A – MODULARNA ARITMETIKA

Savremena kriptografija je zasnovana prvenstveno na upotrebi matematičkih sistema (alata) koji pripadaju teoriji brojeva, tj. delu matematike čiji je primarni cilj proučavanje svojstava prirodnih i celih brojeva. Pod prirodnim brojevima mislimo na skup pozitivnih celih brojeva:

$$N = \{1, 2, 3, 4, \dots\},$$

a pod celim brojevima skup elemenata:

$$N = \{0, +1, +2, +3, \dots\}.$$

Imajući u vidu predmet ispitivanja, tj skup prirodnih i celih brojeva sa kojima se čovek susreo još u drevnim vremenima, teorija brojeva predstavlja jednu od najstarijih matematičkih disciplina. Još u antičkom dobu su bili poznati pojmovi i znanje koje se i dan danas koristi u osnovnim i srednjim školama. Neki algoritmi otkriveni u tom periodu se i danas smatraju efikasnim računarskim algoritmima koji su takođe prisutni u modernoj kriptografiji i kriptografskim algoritmima. Dela Leonarda Ojlera (1707-1783) a posebno uvođenje pojma kongruencije Karl Fridrih Gaus (1777-1855) postavili su temelje moderne teorije brojeva i srodne matematičke discipline - algebre.

U sledećem odeljku ćemo uključiti osnovne informacije iz oblasti teorije brojeva koje se koriste u oblasti primenjene kriptografije. Iako spada u manje popularne discipline matematike, poželjno je napomenuti da uspešno upravljanje ovim sistemom omogućava razumevanje i korišćenje praktičnih algoritama. Tipičan, ali ne i jedini primer jeste generisanje prostih brojeva koji su osnovni gradivni elementi savremenih kriptografskih algoritama i protokola, kao sto su digitalni potpis, šifrovanje javnim ključem itd.

Modularna aritmetika

Modularna aritmetika je aritmetički sistem za cele brojeve gde se brojevi vraćaju u krug nakon što dostignu određenu vrednost – *modulo*. Modularnost za brojeve uveo

je Karl Fridrih Gaus u čuvenom delu "Rasprava o Aritmetici" (*Disquisitiones Arithmeticae*), objavljenom 1801 godine.

Opšte poznata primena modularne aritmetike sastoji se u 24-časovnom merenju vremena: dan traje od ponoći do sledeće ponoći i podeljen je na 24 časa, od 0 do 23h. Ako je u određenom trenutku 21:00h (devet uveče), četiri sata kasnije vreme ne iznosi 25:00h (kao kod uobičajenog sabiranja: $21 + 4 = 25$), već je tada 01:00h (narednog dana).

Ako kažemo da ćemo se sa nekim sresti u 11 časova, to može značiti i 23 sata. Ako koristimo samo 12-časovni interval, onda su 11 i 23 ekvivalentni po modulu 12 i to zapisujemo u obliku:

$$23 \equiv 11 \pmod{12}.$$

Uopšteno, ako je $a \equiv b \pmod{n}$, onda je $a = b + kn$ pri čemu je k ceo broj, tj razlika $a - b$ je deljiva brojem n . Ako je a pozitivan broj i b broj između 0 i $n - 1$, tada se broj b može posmatrati kao ostatak deljenja broja a brojem n . Kaže se da je a kongurentan (identičan) sa brojem b po modulu n (simbol $\equiv$ označava kongruenciju, a broj n modul kongruencije)

Primer za inverzije:

$$6 \pmod{5} = 1 \text{ ili } 6 \equiv 1 \pmod{5}$$

$$45 \pmod{6} = 3 \text{ ili } 45 \equiv 3 \pmod{6}$$

$$40 \pmod{10} = 0 \text{ ili } 40 \equiv 0 \pmod{10}$$

Operacija $a \bmod n$ daje ostatak a koji se nalazi u intervalu od 0 do $n - 1$. Takva operacija se naziva *modularna redukcija*. Modularna aritmetika je u mnogim aspektima slična kao i normalna aritmetika. Mogu se koristiti operacije komutativnosti, atributivnosti i distributivnosti.

- $a \equiv a \pmod{n}$ za svako $a \in \mathbb{Z}$

- ako $a \equiv b \pmod{n}$ tada je $b \equiv a \pmod{n}$
- ako je $a \equiv b \pmod{n}$ i $b \equiv c \pmod{n}$, tada je $a \equiv c \pmod{n}$
- ako1 je $a \equiv b \pmod{n}$ i istovremeno $c \equiv d \pmod{n}$, tada je $(a + c) \equiv (b + d) \pmod{n}$
- ako je $a \equiv b \pmod{n}$ i istovremeno $c \equiv d \pmod{n}$, tada je $(a \cdot c) \equiv (b \cdot d) \pmod{n}$
- ako je $(a \cdot c) \equiv (b \cdot c) \pmod{n}$ a brojevi c, n su nedeljivi, tada je $a \equiv b \pmod{n}$

Operacije sa modulima

Operacija sabiranja:

$$(a \pmod{N} + b \pmod{N}) \pmod{N} = (a + b) \pmod{N}$$

Primeri:

$$(4 + 8) \pmod{5} = 2$$

$$(4 + 4) \pmod{8} = 8 \pmod{8} = 0$$

$$(8 + 13) \pmod{5} = 21 \pmod{5} = 1$$

Operacija oduzimanja:

$$(a \pmod{N} - b \pmod{N}) \pmod{N} = (a - b) \pmod{N}$$

Primeri:

$$(10 - 5) \pmod{3} = 2$$

$$(10 - 5) \pmod{5} = 0$$

$$(12 - 6) \pmod{6} = 6 \pmod{6} = 0$$

$$(22 - 8) \pmod{6} = 14 \pmod{6} = 2$$

Operacija množenja:

$$(a \pmod{N} \cdot b \pmod{N}) \pmod{N} = (a \cdot b) \pmod{N}$$

Primeri:

$$(2 \cdot 5) \pmod{5} = 0$$

$$(3 \cdot 4) \pmod{5} = 2$$

$$(5 \cdot 5) \pmod{5} = 25 \pmod{5} = 0$$

$$(6 \cdot 5) \pmod{4} = 30 \pmod{4} = 2$$

Operacija stepenovanja:

$$a \pmod{N} = b \Rightarrow a^k \pmod{N} = b^k \pmod{N}$$

Primeri:

$$6 \pmod{4} = 2$$

$$6^2 \pmod{4} = 36 \pmod{4} = 0$$

$$6^3 \pmod{4} = 216 \pmod{4} = 0$$

Prosti brojevi

Prosti brojevi su pozitivni celi brojevi koji su deljivi samo sa samim sobom i jedinicom. Ostali brojevi, koji pored sebe i jedinice imaju i druge delioce su složeni. Iako se broj 1

smatra prostim brojem, mnoge definicije podrazumevaju da su prosti brojevi veći ili jednaki od 2. Možemo primetiti da je broj 2 jedini paran prost broj.

Osobine prostih brojeva - Da bismo nastavili sa radom, razmotrićemo neke od važnijih osobina prostih brojeva:

- Funkcija koja daje broj svih prostih brojeva koji su manji od broja n označava se sa $\varphi_{(n)}$ i naziva se "Ojlerova fi funkcija". Prema teoremi o prostim brojevima, za prost broj n važi $\varphi_{(n)} = n - 1$.
- Svaki pozitivan celi broj se na jedinstven način može predstaviti kao proizvod prostih brojeva. Svaki celi broj $n \geq 2$ ima faktorizaciju koja se sastoji od proizvoda potencija prostih brojeva u obliku: $n = p_1 e_1 p_2 e_2 \dots p_k e_k$ gde su p_i uzajamno prosti ($e_i \geq 1$)
- Broj prostih brojeva je beskonačan.
- Marsenovi prosti brojevi imaju oblik $2p-1$. Značajni su zbog toga što se, u odnosu na brojeve koji nisu ovog oblika, na računaru, može lako proveriti da li su prosti. Najveći poznati prosti brojevi često imaju upravo ovaj oblik.
- Prosti brojevi blizanci su prosti brojevi p i $p+2$ tj. susedni prosti brojevi čija je razlika 2.
- Prost broj p je jak prost broj ako: $p-1$ ima veliki prost faktor r , $p+1$ ima veliki prost faktor i ako $r-1$ ima veliki prost faktor.

Područje upotrebe prostih brojeva u kriptografiji - Proste brojeve koristimo u gotovo svim kriptološkim sistemima za generisanje privatnih i javnih ključeva. Efikasno generisanje parametara kojima se generišu ključevi je osnovni preduslov za kriptološke sisteme sa javnim ključem. Kao jedan od primera upotrebe prostih brojeva, može se navesti generisanje para ključeva za RSA algoritam pomoću prostih brojeva p i q . Oni moraju biti dovoljno veliki i slučajni, u smislu da je verovatnoća da bilo koji od njih bude pogođen dovoljno mala, tako da napadač ne može steći prednost kroz optimizaciju strategije pretrage na osnovu tih verovatnoća. Prost brojevi se mogu upotrebiti kako bi se neke njihove dodatne osobine iskoristile protiv nekih specijalizovanih napada.

Značaj jakih prostih brojeva za asimetričnu kriptografiju - U literaturi koja se odnosi na *RSA* algoritme, često se predlaže da se u izboru para ključeva koriste jaki prosti brojevi p i q za generisanje n . Jaki prosti brojevi imaju određene osobine koje čine proizvod n teško faktorizovanim. Razlog ovakvog izbora je to što su neke od metoda faktorizacije kao što su *Polardove* $p-1$ i $p+1$ metode, naročito pogodne za proste brojeve p za koje $p-1$ ili $p+1$ imaju samo male proste faktore. Jaki prosti brojevi su otporni na ovakve napade. Napredak u faktorizaciji tokom poslednjih deset godina pokazuje opravdanost upotrebe jakih prostih brojeva. Za generisanje ključeva u *RSA* algoritmu, preporučuje se upotreba velikih prostih brojeva reda veličine 100 decimalnih cifara i više. Poznato nam je da se generisanje ključeva za *RSA* algoritam odvija tako da se izaberu dva velika broja p i q iz kojih računamo $n = p \cdot q$, a zatim odaberemo e i d uz poštovanje određenih uslova. Postoji dobar razlog zbog čega n ima tačno dva prosta faktora. Prvi razlog je jednostavnost formule za računanje $\varphi_{(n)} = (p-1)(q-1)$, a drugi je teža faktorizacija n , jer je poznato da je faktorizaciju teže izvršiti ukoliko broj ima manje prostih faktora.

Brzi algoritmi za modularnu eksponenciju

Osnovni problem je pronaći ostatak pri deljenju celog broja a^k sa n , tj. pronaći $a^k \pmod{n}$. U obzir treba uzeti da i a i k mogu biti veoma veliki brojevi. Očigledna, ali u većini slučajeva neizvodljiva metoda, sastojala bi se u potenciranju a eksponentom k , a zatim deljenjem sa n i pronalaženjem ostatka. Kako u pojedinim kriptografskim operacijama broj a može imati stotinu i više decimalnih cifara, ovakav metod nije moguće izvesti ni na modernim računarima.

Testovi za ispitivanje da li je broj prost

Generalni model izbora prostog broja - Za kriptografe je generisanje velikih prostih brojeva uvek predstavljalo problem. Generalan metod za generisanje prostih brojeva jeste da se generiše broj n odgovarajuće dužine, a da se zatim proverava da li je on prost. Trivijalnija provera da li je broj prost, bila bi da se proverí da li je on deljiv sa bilo kojim prostim brojem manjim od n . Uopšteni algoritam bi se sastojao u sledećem:

- Generisati neparan broj n koji je kandidat za prost broj;
- Proveriti da li je n prost;
- Ako n nije prost broj vratiti se na prvi korak.

Neznatna unapređenja se mogu postići načinom izbora sledećeg broja koji je kandidat za prost broj tokom povratka na prvi korak. Najprostiji način je provera upotrebom sekvence $n+2$, $n+4$, $n+6$, ...

Testovi za ispitivanje da li je broj prost - Prethodan metod je neefikasan jer zahteva dosta vremena za proveru i znatne računarske resurse. Umesto provere da li su svi brojevi manji od posmatranog broja n , njegovi delioci se koriste kao testovi za ispitivanje da li je broj prost. Postoje dve grupe testova za ispitivanje da li je broj prost:

- "stvarni testovi" čiji je rezultat podatak da li je broj-kandidat prost ili nije prost i
- "probabilistički testovi" koji nam govore da je broj-kandidat ili "verovatno prost" ili je složen.

Probabilistički testovi deklariraju celi broj n kao kandidata za prost broj sa nekom verovatnoćom, dok stvarni testovi obezbeđuju matematički dokaz da je broj prost ili složen. Probabilistički testovi generalno zahtevaju manje računskih resursa i izvršavaju se za kraće vreme od stvarnih testova.

Probabilistički testovi - Probabilistički testovi se najčešće koriste za ispitivanje da li je broj prost. Pored testiranog broja n , koriste i slučajno odabrane brojeve a . Ovakvim testovima se za prost broj nikada ne može dobiti rezultat da je složen, ali je moguće da složen broj testom bude prepoznat kao prost. Redukovanje ove greške može se postići ukoliko se test ponavlja sa nekoliko slučajno odabranih vrednosti a . Za dva najčešće korištena testa, *Fermaov* i *Miler-Rabinov*, za bilo koji složeni broj n najmanje polovina brojeva koje je moguće birati za a detektuju da je n složen. Ovo znači da k ponavljanja smanjuje grešku verovatnoće za najviše 2^{-k} i može se učiniti proizvoljno malom povećavanjem broja pokušaja k .

Osnovni algoritam svakog probabilističkog testa bio bi:

- slučajno biramo broj a ;
- proveravamo određene jednakosti (u zavisnosti od odabranog testa) koje uključuju brojeve a i n ;
- ako jednakosti ne važe test se prekida i broj a je "svedok složenosti" broja n
- ako jednakosti važe vraćamo se na prvi korak i ponavljamo postupak dok ne postignemo potrebnu sigurnost.

Ako nakon određenog broja ciklusa ustanovimo da n nije složen broj, onda ga možemo proglasiti "verovatno prostim".

Fermaov test - Fermaov test bazira se na jednoj od osobina prostih brojeva iskazanoj kroz Fermaovu teoremu. Prema Fermanovoj teoremi, ako je n prost broj onda za svaki celi broj, a takav da je $1 \leq a \leq n-1$ i uzajamno prost sa n važi:

$$a^{n-1} \equiv 1 \pmod{n} \text{ ili } a^{n-1} - 1 \equiv 0 \pmod{n}.$$

Pronalaženje barem jednog broja a iz intervala $1 \leq a \leq n-1$ koji je uzajamno prost sa n za koje ne vrede $a^{n-1} - 1 \equiv 0 \pmod{n}$ garantuje da je n složen broj. Takav broj a se naziva "*Fermanov svedok složenosti*" broja n . Čak i ako za svako a vredi prethodna jednakost, to nije dovoljno da bi se tvrdilo da je n prost. Ako je n složen broj i postoji broj a , $1 \leq a \leq n-1$, takvo da je $a^{n-1} \equiv 1 \pmod{n}$, onda je broj a "*Fermanov lažov*" da je broj n prost, a broj n je pseudoprost za bazu a . Fermanovi pseudoprosti brojevi mogu zadovoljiti Fermanov test. Primer za ovu tvrdnju je složen broj $341 = 11 \cdot 31$ koji je pseudoprost za bazu 2 tj. $2^{340} \equiv 1 \pmod{341}$. "*Karmajkov broj*" je celi složen broj, takav da za sve cele brojeve a iz opsega $[1, n-1]$ koji su uzajamno prosti sa n , tj. $\text{NZD}(a, n) = 1$ važi $a^{n-1} \equiv 1 \pmod{n}$. Najmanji Karmajkov broj je $561 = 3 \cdot 11 \cdot 17$. Ovih brojeva u opsegu $1-10^{15}$ ima 105.212, a u opsegu $1-10^{18}$ oko 1.400.000, odnosno približno jedan u 700 milijardi brojeva. Svaki Karmajkov broj ima najmanje tri prosta faktora. Nedostatak Fermanovog testa se ogleda u sledećem: ako je n Karmajkov broj, tada su jedini svedoci oni brojevi a ($1 \leq a \leq n-1$) koji nisu uzajamno prosti sa n , a ovo znači da ako su svi prosti faktori broja n veliki, čak i Fermanov test sa velikim brojem

iteracija t može dati rezultat da je n prost broj. Neefikasnost Fermanovog testa je otklonjena u *Solovej-Štrasenovom* i *Miler-Rabinovom* testu koji počivaju na strožim kriterijumima.

Solovej-Štrasenov test - *Solovej-Štrasenov* test je postao popularan pojavom asimetričnih šifarskih sistema, a posebno pojavom *RSA*. On koristi *Jakobijev simbol* $J(a, n)$ za ispitivanje da li je n prost broj, a zasnovan je na Ojlerovom kriterijumu. Prema Ojlerovom kriterijumu, ako je n prost broj onda za svaki celi broj a takav da je $1 \leq a \leq n-1$ i koji je uzajamno prost sa n važi:

$$a^{(n-1)/2} \equiv J(a, n) \pmod{n}.$$

Složen broj n koji je uzajamno prost sa a i zadovoljava $a^{(n-1)/2} \equiv J(a, n) \pmod{n}$ kažemo da je "Ojlerov pseudoprost broj za bazu a ", a broj a je "Ojlerov lažov". Ako a nije uzajamno prost sa n ili ako je $a^{(n-1)/2} \not\equiv J(a, n) \pmod{n}$, kažemo da je a "svedok" složenosti broja n .

Primer:

$n = 91 = 7 \cdot 13$ je Ojlerov pseudoprost broj za bazu 9 jer je $9^{45} \equiv 1 \pmod{91}$ i $J(9, 91) = 1$.

Ponavljanjem testa t puta sa različitim vrednostima a , verovatnoća da složeni broj n prođe svih t testova nije veća od $1/2^t$. Broj a koji dokazuje da je n prost broj je svedok. Verovatnoća da slučajan broj a bude svedok nije manja od 50%.

Lemanov test - Lemanov test je nešto jednostavniji od Solovej-Štrasenovog testa jer se ne računa Jakobijan. Kao i kod Solovej-Štrasenovog testa verovatnoća da je broj a svedok složenosti broja n nije manja od 50%. Test treba ponoviti t puta za različite vrednosti broja a . Ako je u svih t koraka vrednost $a^{(n-1)/2} \equiv 1$ ili -1 , ali nije uvek 1, p je verovatno prost broj sa mogućnošću greške od $1/2^t$.

Miller-Rabinov test - *Miller-Rabinov* test je efikasniji, a podjednako je korektan kao i Solovej-Štrasenov, pa ga je u potpunosti i zamenio. Zbog svoje jednostavnosti ovo je najčešće korišćen probabilistički test za ispitivanje da li je broj prost. Naziva se još i "jaki pseudoprosti test" jer se zasniva na nekim osobinama jakih pseudoprostih brojeva.

Predstavimo broj n u obliku $n-1 = 2^s \cdot r$ gde je s najveći stepen broja 2 koji deli $n-1$. Jasno je da je r neparan broj. Jak pseudoprost broj za bazu a je neparan složen broj n , ako postoji broj j , $0 \leq j \leq s-1$, takav da važi ili $a^r \equiv 1 \pmod{n}$ ili $a^{2^{j^*r}} \equiv n-1 \pmod{n}$. Broj a se naziva "jakim lažovom" da je broj n prost. Ako je ipak $a^r \not\equiv 1 \pmod{n}$ i $a^{2^{j^*r}} \not\equiv n-1 \pmod{n}$ za sve j , $0 \leq j \leq s-1$, onda je a "jak svedok" da je n složen broj.

Verovatnoća da neki složen broj prođe test, brže opada u ovom testu nego u ostalim i iznosi $1/4^t$, gde je t broj ciklusa. Tri četvrtine mogućih vrednosti a sigurno su svedoci. Za većinu slučajnih brojeva, oko 99,9% mogućih vrednost a su svedoci.

Faktorizacija brojeva

Faktorizacija brojeva jedan je od značajnijih problema u teoriji brojeva, ali i u kriptografiji jer se sigurnost popularnog RSA kriptografskog algoritama zasniva na nemogućnosti napadača da faktorizira velike brojeve u prihvatljivom vremenu. Sledeća tvrdnja je takodje poznata i kao osnovna teorema aritmetike. Svaki pozitivan ceo broj $N > 1$ može da se predstavi kao proizvod jednog ili više prostih brojeva u sledećem obliku:

$$N = p_1^{e_1} \cdot p_2^{e_2} \cdot \dots \cdot p_n^{e_n}$$

gde su $p_1 \dots p_n$ različiti prosti brojevi, dok su $e_1 \dots e_n$ prirodni brojevi. Ovaj postupak naziva se faktorizacija broja n . Faktorizacija broja je jedinstvena.

Najveći zajednički delilac

U matematici, najveći zajednički delilac (NZD) dva cela broja različita od nule je najveći pozitivan ceo broj koji deli oba broja bez ostatka.

Najveći zajednički delilac brojeva a i b se označava kao $NZD(a, b)$ ili ponekad jednostavnije kao (a, b) .

Na primer, $NZD(12, 18)=6$, $NZD(-4, 14)=2$ i $NZD(5, 0)=5$. Dva broja su *uzajamno prosta* ako im je najveći zajednički delilac jednak 1. Na primer, 9 i 28 su uzajamno prosti.

Najveći zajednički delilac se načelno može izračunati razlaganjem dva broja na proste činioce i upoređivanjem činilaca, kao u sledećem primeru: da bismo našli $NZD(18, 84)$, nalazimo proste činioce od $18=2 \cdot 3^2$ i $84=2^2 \cdot 3 \cdot 7$, primećujemo da je *preklapanje* dva izraza $2 \cdot 3$; pa je $NZD(18, 84)=6$. U praksi, ovaj metod je izvodljiv samo za jako male brojeve; razlaganje na proste činioce načelno može da bude vrlo komplikovano. Mnogo efikasniji metod je *Euklidov algoritam*.

Euklidov algoritam - Euklidov algoritam je zasnovan na principu da se najveći zajednički delilac dva broja ne menja ukoliko se manji broj oduzme od većeg, pa se zatim odredi NZD novodobijenog broja i manjeg od prethodna dva. Na primer, 21 je NZD za 252 i 105 ($252=21 \cdot 12$; $105=21 \cdot 5$); pošto je $252-105=147$, NZD za 147 i 105 je takođe 21. Kako je veći od dva polazna broja na ovaj način smanjen, ponavljanjem postupka dobijaće se sve manji brojevi, dok se jedan od njih ne svede na nulu.

Euklidov algoritam je iterativne prirode, što znači da se krajnji rezultat dobija u nizu koraka, dok se međurezultat proizvoljnog koraka koristi u prvom narednom. Ukoliko je k ceo broj kojim su označeni koraci algoritma počevši od nule, prvom koraku odgovara jednakost $k=0$, drugom $k=1$, i tako dalje.

Svaki korak počinje sa dva pozitivna ostatka r_{k-1} i r_{k-2} . Kako algoritam osigurava da se ostaci svakim korakom neprekidno smanjuju, r_{k-1} je manje od svog

prethodnika r_{k-2} . Cilj k -tog koraka je da se odrede količnik q_k i ostatak r_k takvi da važi jednakost

$$r_{k-2} = q_k r_{k-1} + r_k$$

gde je $r_k < r_{k-1}$. Drugim rečima, umnošci manjeg broja r_{k-1} se oduzimaju od većeg broja r_{k-2} sve dok je dobijeni ostatak manji od r_{k-1} .

U prvom koraku ($k=0$), ostaci r_{-2} i r_{-1} su jednaki a i b respektivno, a to su upravo brojevi za koje se traži NZD. U sledećem koraku ($k=1$), ostaci postaju jednaki b i ostatku početnog koraka r_0 ... Na osnovu toga, algoritam se može predstaviti nizom jednakosti:

$$a = q_0 b + r_0$$

$$b = q_1 r_0 + r_1$$

$$r_0 = q_2 r_1 + r_2$$

$$r_1 = q_3 r_2 + r_3$$

...

Ukoliko je a manje od b , u prvom koraku algoritma treba zameniti brojeve, odnosno, za $a < b$, početni količnik q_0 je jednak nuli, a ostatak r_0 je jednak a . Na taj način, r_k je manji od svog prethodnika r_{k-1} za sve $k \geq 0$.

Kako se ostaci smanjuju u svakom koraku i kako ne mogu biti negativni, ostatak r_N u nekom trenutku mora postati jednak nuli pa se tada algoritam zaustavlja. Poslednji ostatak r_{N-1} koji je različit od nule je najveći zajednički delilac brojeva a i b . Broj N ne može biti beskonačan pošto između nule i prvog ostatka r_0 postoji konačan broj pozitivnih celih brojeva.

Promenljive a i b se smenjuju u čuvanju prethodnih ostataka r_{k-1} i r_{k-2} . Ukoliko je na početku iteracije a veće od b , onda a dobija vrednost r_{k-2} , pošto je $r_{k-2} > r_{k-1}$. U toku izvršavanja petlje a se smanjuje oduzimanjem prethodnog ostatka b sve dok a ne postane manje od b . Tada je a sledeći ostatak r_k . Zatim se b smanjuje oduzimanjem a dok ne postane manje od njega i tada postaje jednako r_{k+1} . Proces se nastavlja dok b ne postane jednako nuli.

Modularna inverzija

Postoji aditivna inverzija i multiplikativna.

Aditivna inverzija x po modu N je broj koji treba sabrati sa x da bi mod tog zbira bio jednak 0. Označava se sa $-x$. To nije negativan broj nego samo oznaka.

Primer:

Kolika je aditivna inverzija $4 \pmod{5}$

Odgovor: 1

$$-1 \pmod{5} = 4 \text{ jer je } (1+4) \pmod{5} = 0$$

Multiplikativna inverzija x po modu n je broj koji treba pomnožiti sa x da bi mod tog proizvoda bio jednak 1, a označava se sa x^{-1} . To nije broj manji od 1 već samo oznaka.

Primer:

Kolika je multiplikativna inverzija $4 \pmod{5}$?

Odgovor:

$$x \cdot 4 \pmod{5} = 1; 4 \cdot 4 \pmod{5} = 1; x = 4 \text{ ili}$$

$$4^{-1}(\bmod 5) = 4,$$

jer je:

$$(4 \cdot 4)(\bmod 5) = 1$$

Ojlerova funkcija

Ojlerova funkcija je dobila ime po Švajcarskom matematičaru *Leonardu Ojleru* (1707-1783). Ojlerova φ funkcija je važna uglavnom zbog toga što daje veličinu multiplikativnih grupa celih brojeva po modulu n .

$\varphi(n)$ je broj iz skupa pozitivnih celih brojeva manjih od n koji su uzajamno prosti u odnosu na n .

Primeri:

Ako je p prost broj onda je:

$$\varphi_{(p)} = p - 1$$

Ako je p prost broj onda je:

$$\varphi_{(p^n)} = p^n - p^{n-1}$$

Ako su m i n uzajamno prosti brojevi onda je:

$$\varphi_{(mn)} = \varphi_{(m)}\varphi_{(n)}$$

Ako su p i q prosti onda je:

$$\varphi_{(pq)} = \varphi_{(p)}\varphi_{(q)} = (p-1)(q-1)$$

Za svaki pozitivan broj n i svako x koje je uzajamno prosto sa n vazi:

$$x^{\varphi(n)} \equiv 1 \pmod{n} \quad x^{\varphi(n)} \pmod{n} = 1$$

Kvadratni ostatak

Neka je $(a, m) = 1$. Ako kongruencija $x^2 \equiv a \pmod{m}$ ima rešenja, onda kažemo da je a kvadratni ostatak modula m .

Primer:

Kvadratni ostaci modula 6 su 1, 3, 4, a nedostaci su 2 i 5.

Teorema:

Neka je $p \in P$ neparan prosti broj. Redukovani sistem ostataka modula p sastoji se od $\frac{p-1}{2}$ kvadratnih ostataka i $\frac{p-1}{2}$ kvadratnih neostataka.

Kineska teorema o kvadratnom ostatku - Sledeće tvrđenje poznato je po kineskom matematičaru Sun-Tse-u iz prvog veke nove ere, pa je poznato pod nazivom "kineska teorema o ostacima". Neka su $m_1, m_2, \dots, m_k$ po parovima uzajamno prosti pozitivni celi brojevi. Tada sistem linearnih kongruencija:

$$x \equiv c_1 \pmod{m_1}; x \equiv c_2 \pmod{m_2}; \dots; x \equiv c_k \pmod{m_k}$$

ima rešenje za sve $c_1, c_2, \dots, c_k \in \mathbb{Z}$ i pri tome skup svih rešenja čini jednu klasu ostatka po modulu $m_1 m_2 \cdots m_k$.

Dokaz izvodimo indukcijom po k , broju jednačina koje čine sistem. Pretpostavimo da je $k > 2$ i da svaki sistem tipa $x \equiv c_1 \pmod{m_1}, x \equiv c_k \pmod{m_k}$ koji se sastoji od $k - 1$ jednačine ima rešenje, pri čemu skup svih rešenja čini jednu klasu ostatka po modulu proizvoda datih modula. Zbog toga je sistem:

$$x \equiv c_1 \pmod{m_1}; x \equiv c_2 \pmod{m_2}; \dots; x \equiv c_{k-1} \pmod{m_{k-1}}$$

ekvivalentan uslovu $x \equiv c \pmod{m_1 m_2 \cdots m_{k-1}}$, za pravilno odabrano $c \in \mathbb{Z}$.

Diskretni logaritmi

Modularno stepenovanje u obliku:

$$y \equiv a^x \pmod{n}$$

pripada takozvanim jednosmernim funkcijama koje se u modernoj kriptologiji često koriste.

Kvantifikacija prethodnog izraza je jednostavna. Inverzan (suprotan) problem za modularno stepenovanje je pronaći diskretni logaritam broja, tj. naći brojeve x tako da važi:

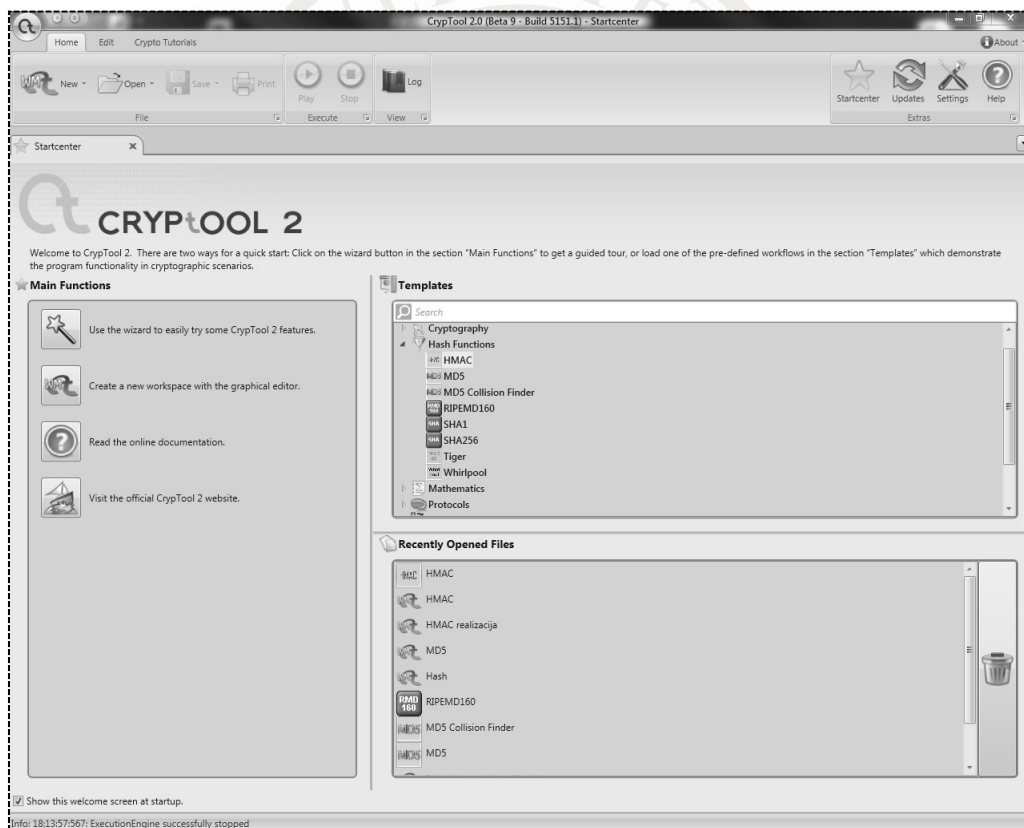
$$a^x \equiv y \pmod{n} \text{ i } a^x \pmod{n} \equiv c$$

Ovaj problem je vremenski zahtevan i smatra se teškim. Nije poznato da li postoji efikasan algoritam za rešavanje problema diskretnog logaritma.

Modularna aritmetika ima puno korisnih primena u današnjem svetu. Mnogi misle da je matematika bespotrebna nauka, ali to nije tačno. Osim teorije brojeva i kriptografije čiju smo primenu videli u ovom radu, modularna aritmetika ima primenu i u računarstvu, vizuelnoj umetnosti, pa čak i muzici.

DODATAK B – EDUKATIVNO RAZVOJNO OKRUŽENJE CRYPTOOOL

CrypTool je interaktivna kriptografska laboratorija razvijena kao „open source“ projekat. Laboratorija nudi vizualizaciju pojmova, podataka i fundamentalnih kriptografskih objekata i metoda. Uključuje popularne algoritme i protokole za šifrovanje kao i više metoda za kriptanalizu. Moguće je modelovati metode za kriptanalizu (Frekventni test, Fridmenov test, Kaziskijev test), kao i kriptanalizu klasičnih i modernih šifara na teorijskoj i praktičnoj osnovi modelovanjem eksperimentalnog okruženja sa simulacijom u realnom vremenu. Ovaj knjiga opisuje naš nastavni model i iskustvo, kao i pozitivne povratne informacije dobijene od strane studenata su potvrdile i bolji kvalitet usvojenih pristupa (Slika 0.1).



Slika 0.1 Izgled razvojnog okruženja CrypTool 2

Instalacija razvojnog okruženja - Na zvaničnoj stranici CrypTool-a dostupne su 32-bitne i 64-bitne verzije programa za Windows platformu.

<http://www.cryptool.org/en/ct2-download-en>

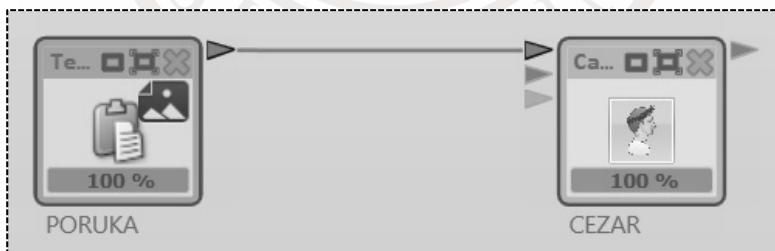
Pre pokretanja instalacije CrypTool-a, neophodno je instalirati *Microsoft .NET framework*. Instalaciju je moguće preuzeti sa linka:

<http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=17718>

Na našem YouTube kanalu možete da pratite video materijale koji prate sadržaj ove knjige. Pored materijala za predmet kriptologija 1, dostupni su i materijali za kriptologiju 2 i osnove teorije informacija i kodovanja.

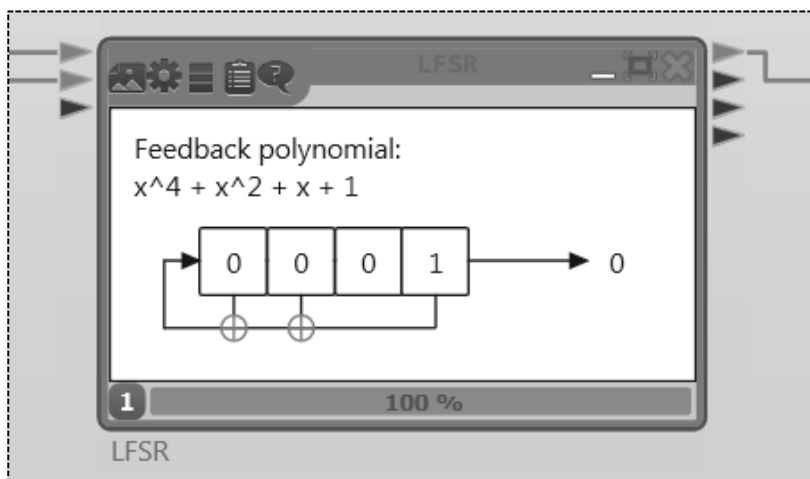
<http://www.youtube.com/user/salerc>

Moderni „Plug-in-Play“ interfejs za vizuelno programiranje – CrypTool 2 pruža grafički korisnički interfejs za vizuelno programiranje. Dakle, sa okruženjem je lako upravljati, podaci mogu lako da se kontrolišu, vizuelizuju i na taj način je ostvarena jednostavna interakcija sa kriptografskim funkcijama. Vektorski orjentisano grafičko okruženje u Windows-ovom stilu daje korisnicima mogućnost dobrog trenutnog pregleda. CrypTool 2 sledi *Microsoft Office 2007* korisnički interfejs smernica koji obezbeđuje dosledan i bogat korisnički doživljaj, **Slika 0.2**.



Slika 0.2 Cryptool 2 „Plug-in-Play“ interfejs

Vizuelizacija algoritama – Mnoge komponente mogu da se koriste za vizualizaciju toka programa (algoritma) i na taj način mogu demonstrirati svoj unutrašnji rad. Ovo je vrlo značajno za korisnika jer mu omogućava da shvati sve detalje kriptografskog algoritma, onako kako se dati algoritam koristi u realnom scenariju, **Slika 0.3**.



Slika 0.3 Vizuelizacija algoritama

Sveobuhvatne funkcije za kriptanalizu – Cryptool 2 pruža različite kriptanalitičke alate (metode) za analizu, dok sa nekim mogu čak neke klasične i moderne šifre da se razbiju. Mnogo drugih funkcija za kriptanalizu je na raspolaganju, kao što su testovi frekvencija za analizu strukture prirodnih jezika.

Open source - Cryptool je uvek bio *open source* projekat. Cryptool je licenciran od strane *Apache Open Source Licence 2.0*.

Sa Cryptool-om, moderni algoritmi su implementirani u skladu sa internacionalnim standardima predloženim od strane NIST-a. Autori Cryptool-a nisu implementirali funkcionalnosti tamo gde su bili dostupni dobri besplatni programi na Internetu: Tako, npr. rečnički napadi na datoteke sa lozinkama (pogledajte *John-the-Ripper*) još uvek nisu deo Cryptool-a. Ipak, Cryptool poseduje metod za proveru kvaliteta vaših lozinki.

RSA kriptosistem je sveobuhvatno pokriven i implementiran za različita kodovanja. RSA može da se generiše iz prostih brojeva odabranih od strane korisnika. Generisanje ključa, šifrovanje i dešifrovanje mogu biti ispraćeni korak-po-korak.

Faktorizacija brojeva ima takođe značajnu primenu u kriptografiji. Jednostavni RSA kriptosistemi mogu biti sa lakoćom razbijeni upotrebom algoritama za razlaganje predstavljenih u Cryptool-u.

Ove interaktivne demonstracije (vizualizacije) su od pomoći za bolje razumevanje sledećih tema:

- kreiranje elektronskog potpisa;
- procesiranje hibridnog šifrovanja kako promene na dokumentu mogu da promene heš vrednost;
- osetljivost heš algoritama;
- generisanje kolizija heš vrednosti (paradoks rođendana);
- Diffie-Hellman razmena ključeva;
- napad na sporedne kanale;
- protokoli za autentikaciju;
- steganografske tehnike (LSB);
- deljenje tajne korišćenjem Kineske Teoreme Ostatka;
- vizualizacije nekih algoritama (npr. Vižner, AES, Enigma);
- drevne teoreme teorije brojeva (kineska teorema ostatka) mogu da se primene (npr. kada su sve planete našeg Sunčevog sistema poređane u jednu liniju);
- 3D vizualizacija velikih skupova nasumičnih brojeva;
- računarski zasnovano učenje teorije brojeva;
- dodavanje tačaka na eliptičnoj krivoj.

Kriptologija je zahtevna i teška naučna disciplina koja zahteva jaku matematičku pozadinu zbog bezbednosti šifarskih sistema koja se zasniva na nesposobnosti rešavanja raznih algebarskih problema, problema iz teorije brojeva i kombinatorike.

Zbog navedene kompleksnosti CrypTool open-source laboratorija nam je omogućila da na praktičan način koristimo sve dostupne algoritme za modelovanje našeg šifarskog sistema ili za modelovanje raznih kriptanalitičkih metoda. Laboratorija koristi originalne algoritme izrađene po NIST-ovom (eng. *National Institute of Standards and Technology*) standardu. Prikazani kriptanalitički modeli su uspešno realizovani što je potvrđeno dobijenim rezultatima. U ovom udžbeniku, Cryptool je predstavljen kao moćan alat za kombinovanje teorije i prakse u kriptologiji.

Jedan od osnovnih problema pri učenju kriptologije predstavlja njena kompleksnost i zasnovanost na složenim matematičkim principima i formulama. Najveći broj bezbednosnih rešenja danas ne zahteva detaljno poznavanje pomenutih matematičkih principa već samo njihovo osnovno razumevanje i povezivanje sa ulogom u praktičnim rešenjima. Prebacivanjem fokusa iz domena matematičkih osnova bezbednosnih rešenja u domen praktične primene kod savremenih informacionih i telekomunikacionih tehnologija značajno se menja i profil studenata u ovoj oblasti. Iz tog razloga je ovo preusmeravanje neophodno podržati odgovarajućim edukacionim pristupima, materijalima i uslovima za učenje.

Naša iskustva sa korišćenjem tradicionalnog pristupa kod podučavanja kriptologije ukazala su na potrebu za novim, interaktivnijim i kolaborativnijim načinima za sticanje znanja iz ove oblasti. Umesto korišćenja bottom-up pristupa - kretanja od potrebnih matematičkih aparata ka njihovoj kriptografskoj primeni - odlučili smo se za top-down pristup: predstavljanje kriptoloških principa u kontekstu njihove praktične primene u cilju izazivanja interesovanja za savladavanje njihovih nosećih matematičkih principa. Za realizaciju navedenog pristupa javila se potreba za softverskim edukacionim okruženjima koja nude odgovarajući nivo apstrakcije pri rešavanju problema.

Ovom prilikom se zahvaljujemo svim učesnicima projekta „CrypTool“, kao i sponzorima projekta koji su nam omogućili da na jedan interesantan i interaktivan način savladamo gradivo iz kriptologije.

REČNIK POJMOVA



AES - Advanced Encryption Standard. Blokovska šifra sa simetričnim ključem, standardizovana od strane NIST-a. Pripada grupi modernih šifara. Dužina ključa je 128, 192, i 256 bitova.

ASCII - američki kodni standard za razmenu informacija. Ove kodove koristi veliki broj računarskih sistema za prevođenje karaktera u binarne brojeve.

Algoritam - opis za rešavanje nekog problema.



Biometrija - tehnika za autentifikaciju koja koristi jedinstvene fizičke karakteristike nekog pojedinca. Postoje različiti tipovi biometrije.

Biometrijski templejt - digitalizovana fizička karakteristika nekog pojedinca za određeni biometrijski izvor koji se koristi za autentifikaciju date osobe.

Binarna sekvenca - niz karaktera (jedinica i nula) ili binarni niz.



VOIP - protokol koji se koristi za digitalizaciju govora, pakovanje i slanje preko TCP/IP veze.

Vizuelna kriptografija - specijalna tehnika šifrovanja koja omogućava skrivanje informacija na slici, na takav način da se može dešifrovati ljudskim okom bez upotrebe računara.

Venona - predstavlja One-time pad u realnosti. Koristila ga je sovjetska špijunska mreža koja je formirana na teritoriji SAD 1940-tih.

G

GSM - globalni sistem za telekomunikacije. Internacionalni standard za satelitske telefone.

D

DES - Data Encryption Standard. Blokovska šifra sa simetričnim ključem usvojena od strane NIST-a. Pripada grupi modernih šifara. Dužina ključa je 56 bitova, prevaziđen je od strane AES-a.

Digitalni potpis - protokol koji koristi primalac poruke da bi verifikovao pošiljaoca poruke. Za verifikaciju se koristi javni ključ pošiljaoca. Obezbeđuje servis neporecivosti.

Dešifrovanje - proces transformacije šifrata u otvoreni tekst.

Digitalni vodeni pečat - tehnika za zaštitu autorskih prava digitalnih sadržaja.

Difi-Helman - protokol namenjen za razmenu simetričnih ključeva preko javnih komunikacionih kanala.

E

Enigma - mašina za šifrovanje iz familije mehaničkih rotor mašina. Koristile je nemačke snage za vreme Drugog svetskog rata.

Entropija - mera za količinu neodređenosti ili prosečna količina informacije koju sadrže generisane poruke nekog informacionog izvora.

ECB - režim za šifrovanje kod blokovskih šifara u kom je svaki blok šifrata potpuno nezavisan. Ovaj režim može da bude nebezbedan.

I

Iris - šara irisa ili obojeni deo oka koji je prilično haotičan (neodređen) ili slučajan. Predstavlja biometrijski izvor sa visokim performansama.

Integritet - prevencija od neautorizovane izmene sadržaja (poruke).

J

Javni ključ - element sistema sa javnim ključem ili javni ključ nekog korisnika koji je dostupan svim korisnicima u mreži. Koristi se u fazi šifrovanja.

Jednosmerna funkcija - sam naziv kaže da je funkcija neinvertibilna, ako je $y = f(x)$, tada je $x \neq f(y)$. Heš funkcije su vrsta jednosmernih funkcija.

K

Kriptoanaliza - umetnost i nauka o razbijanju šifrovanih poruka.

Kriptološki ključ - binarna sekvenca veće dužine, poseduje osobine slučajnih nizova. Koristi se kao početno stanje algoritma u fazi šifrovanja i dešifrovanja.

Kriptograf - lice koje dizajnira šifre.

Kriptoanalitičar - lice koje dizajnira metode za razbijanje šifara.

Kriptografija - nauka o pravljenju algoritama za šifrovanje.

Kriptoanaliza - nauka o razbijanju algoritama za šifrovanje.

L

LAN - lokalna mreža.

LFSR - linearni pomerački registar sa povratnom spregom. Njegova uloga se sastoji u generisanju pseudoslučajnih sekvenci. Ima periodu (ponavlja se nakon izvesnog vremena).

LSB - steganografska tehnika za ugrađivanje tajnih poruka u podatak nosilac po principu zamene bitova sa najmanjim značajem.

Lozinka - tajni akreditiv (string od prosečno 8 karaktera) koji se koristi za autentifikaciju kod tradicionalnih sistema za autentifikaciju. Korisnik je u stanju da zapamti više lozinki, dok kriptološke ključeve nije moguće zapamtiti.

M

MAC - kod za proveru autentičnosti poruke.

N

NIST - američka federalna agencija za standarde i tehnologiju.

NSA - Nacionalna agencija za bezbednost. Američka agencija koja je odgovorna za bezbednost i kriptoevalu elektronskih komunikacija.

O

One-time pad - perfektna šifra koja pripada grupi šifara sa simetričnim ključem. Ključ je generisan na slučajan način i ima jednaku dužinu kao i poruka. Ključ ne sme nikad da se ponavlja.

Otvoreni tekst - poruka u fazi šifrovanja ili rezultat funkcije za dešifrovanje.

P

PGP - program za sigurnu elektronsku poštu koji objedinjuje servise za kriptografsku zaštitu podataka i digitalno potpisivanje. Razvijen od strane Fila Cimermana.

PKI - infrastruktura sa javnim ključevima. Sadrži bazu javnih ključeva preko kojih obezbeđuje autentifikaciju u srednjim i velikim mrežama.

Privatni ključ - element sistema sa javnim ključem ili tajna koju čuva svaki korisnik. Koristi se u fazi dešifrovanja.

PRNG - generator za generisanje pseudoslučajnih brojeva na osnovu inicijalnog stanja koje je određeno na slučajan način.

Poruka - podatak koji se razmenjuje preko mreže između predajne i prijemne strane. Može da bude u šifrovanoj formi.

Protokol - protokol je skup pravila koje razumeju i poštuju strane u komunikaciji.

R

RSA - kriptografski sistem sa javnim ključem, patentiran od strane Rivesta, Šamira i Adlemana 1976. Sigurnost RSA zasniva se na složenosti faktORIZACIJE velikih brojeva. Ima široku primenu, implementiran je u PGP i SSL-u.

RC4 - sekvencijalna šifra sa simetričnim ključem. Pripada grupi modernih šifara. Dužina ključa jednaka je poruci. Neki je nazivaju pseudoperfektna šifra. Za generisanje ključa koristi se generator za pseudoslučajne brojeve. Pravila implementacije su skoro ista kao i kod perfektnih šifara.

S

Sertifikat - elektronski fajl, obično sadrži javni ključ koji je digitalno potpisan od Sertifikacionog tela. Sertifikati se koriste za autentifikaciju na Internetu.

Sertifikaciono telo - treća strana od poverenja koja potpisuje i distribuira sertifikate.

SSL - široko primenjeni zaštitni protokol za autentifikaciju na Internetu (Internet transakcije)

Simetrični ključ - kriptološki ključ koji je identičan na predajnoj i prijemnoj strani.

Steganografija - nauka o skrivanju informacija u drugim bezazlenim podacima. Čest primer je skrivanje tekstualnog sadržaja u neku sliku.

T

TCP/IP - protokol koji se koristi za prenos informacija u računarskim mrežama. TCP/IP je postao standard za sve mreže povezane sa Internetom.

TDES - blokovska šifra sa simetričnim ključem, usvojena od strane NIST-a. Bazirana na sukcesivnoj aplikaciji tri DES algoritma sa različitim ključevima. Predstavlja ojačanje klasičnog DES-a.

TRNG - generator slučajnih brojeva; ovakve generatore je moguće konstruisati u prirodnim informacionim izvorima.

F

Faktorizacija - rastavljanje velikih prostih brojeva na činioce iz skupa prostih brojeva.

Fejstel šifra - nazvana po nemačkom kriptografu Horstu Fejstelu (Horst Feistel) koji je pionir u razvoju dizajna blokovskih šifara, radio je u IBM-u. Ovo su bila inicijalna istraživanja koja su kulminirala razvoju DES (Data Encryption Standard) algoritma 1970. godine. Fejstel šifra predstavlja dizajn blokovske šifre, a ne posebnu šifru.

H

Haker – originalno se termin koristi za kreativnog programera koji razvija programe za ilegalan pristup računarskim mrežama u cilju krađe ili uništenja informacija. Dobar haker treba da poseduje dobre kriptanalitičke veštine.

Hamingovo rastojanje - koristi se za određivanje razlike između dva niza jednake dužine.

Heš funkcije - funkcija koja generiše skraćenu verziju poruke. U kriptografskim aplikacijama heš funkcije moraju da budu jednosmerne sa malom verovatnoćom da dve različite poruke mogu da daju isti rezultat.

C

CRC - Cyclic Redudancy Check. Kod široke primene, koristi se za detektovanje grešaka. Obično pre dešifrovanja, CRC se proverava.

CBC - režim za šifrovanje kod blokovskih šifara. Radi na principu ulančavanja blokova šifrata.

Š

Šifra - algoritam za šifrovanje. Mogu da postoje različiti tipovi šifara. Osnovne podele su na klasične i moderne, simetrične i asimetrične, apsolutno i praktično tajne.

Šifrat - rezultat funkcije za šifrovanje ili transformacija otvorene poruke koja je inverzna uz posedovanje pravog ključa.

Šifrovanje - proces transformacije otvorenog teksta u šifrat.

CLASS OF SERVICE DESIRED	
Full Day Message	☒
Day Letter	☐
Night Message	☐
Night Letter	☐

WESTERN UNION

TELEGRAM

RECEIVED CARLTON FREEMANT

M-C	
Date	
Time Filed	

Send the following telegram, subject to the terms on back hereof, which are hereby agreed to

via Galveston

JAN 19 1917

GERMAN LEGATION
MEXICO CITY

130	13042	13401	8501	115	3528	416	17214	6491	11310
18147	18222	21580	10247	11518	23677	13605	3494	14938	
98092	5905	11311	10392	10371	0302	21290	5181	39695	
23571	17504	11209	18276	18101	0317	0228	17694	4473	
23284	22200	19452	21589	07893	5569	13918	8958	12137	
1333	4725	4458	5905	17166	13851	4458	17149	14471	0708
13850	12224	6929	14991	7382	15857	67893	14218	36477	
5870	17553	07893	5870	5454	18102	15217	22801	17138	
21001	17388	7410	23638	18222	0719	14331	15021	23845	
3152	23552	22006	21604	4797	9497	22405	20855	4377	
23410	18140	22290	5905	13347	20420	39689	13732	80607	
8928	5275	18507	52202	1340	22049	13339	11265	22295	
10439	14814	4178	8992	8784	7632	7357	6926	52268	11267
21100	31272	9346	9559	22494	15874	18502	18500	15857	
2182	5376	7381	98092	16127	13486	9350	9220	76036	14219
5144	2831	17920	11347	17142	11264	7667	7762	15099	9110
5870	17553	07893	5870	5454	18102	15217	22801	17138	
21001	17388	7410	23638	18222	0719	14331	15021	23845	
3152	23552	22006	21604	4797	9497	22405	20855	4377	
23410	18140	22290	5905	13347	20420	39689	13732	80607	
8928	5275	18507	52202	1340	22049	13339	11265	22295	
10439	14814	4178	8992	8784	7632	7357	6926	52268	11267
21100	31272	9346	9559	22494	15874	18502	18500	15857	
2182	5376	7381	98092	16127	13486	9350	9220	76036	14219
5144	2831	17920	11347	17142	11264	7667	7762	15099	9110
10482	97558	3569	3070						

BEPNSTOPFF.

CIP - Каталогизација у публикацији - Народна библиотека Србије, Београд

003.26(075.8)

004.056.55(075.8)

ВЕИНОВИЋ, Младен, 1962-

Kriptologija I : osnove za analizu i sintezu šifarskih sistema / Mladen
Veinović, Saša Adamović. - 3. izd. - Beograd : Univerzitet Singidunum, 2018
(Loznica : Mobid). - 244 str. : ilustr. ; 24 cm

Na vrhu nasl. str.: Fakultet za informatiku i računarstvo. - Tiraž 500. -
Rečnik pojmova: str. 237-243. - Napomene i bibliografske reference uz
tekst. - Bibliografija: str. 214-215.

ISBN 978-86-7912-576-7

1. Адамовић, Саша, 1985- [аутор]
а) Криптологија б) Криптографија

© 2018.

Sva prava zadržana. Nijedan deo ove publikacije ne može biti reprodukovан u bilo kom vidu i putem bilo kog medija, u delovima ili celini bez prethodne pismene saglasnosti izdavača.



Mladen Veinović
Saša Adamović

KRIPTOLOGIJA I

OSNOVE ZA ANALIZU I SINTEZU ŠIFARSKIH SISTEMA

„Kriptologija I“ je udžbenik koji sadrži neophodna znanja iz osnova kriptologije uz primenu matematičkog aparata koji je potreban za analizu i sintezu savremenih šifarskih sistema. Pomoću ovog udžbenika savladavaju se osnovni servisi zaštite informacija: tajnost, integritet, autentifikacija i neporecivost. Detaljno se razmatraju svojstva simetričnih i asimetričnih šifarskih sistema. Studenti će biti osposobljeni da samostalno procenjuju kvalitet zadatog šifarskog sistema i razumeju njegovo mesto i ulogu u savremenom i integrisanom računarskom okruženju.

