



UNIVERZITET SINGIDUNUM

FAKULTET ZA INFORMATIKU I RAČUNARSTVO

**Razvoj sopstvenog rešenja za
kriptografsku zaštitu sa
implementiranim modulom za
generisanje simetričnog kriptološkog
ključa**

- Diplomski rad -

Mentor:

Doc. dr Aleksandar Jevremović

Student:

Marija Vujošević

Br. Indeksa: 2011/201751

Beograd, 2013.



FAKULTET ZA INFORMATIKU I RAČUNARSTVO

UNIVERZITET SINGIDUNUM

FAKULTET ZA INFORMATIKU I RAČUNARSTVO

Beograd, Danijelova 32

Kandidat: Marija Vujošević

Broj indeksa: 2011/201751

Smer: Informatika i računarstvo

Tema: Razvoj sopstvenog rešenja za kriptografsku zaštitu sa implementiranim modulom za generisanje simetričnog kriptološkog ključa

Zadatak: Razvijenim rešenjem generisati čisto slučajne ključeve (TRNG). Prikazati eksperimentalne rezultate testiranja i porediti ih sa uzorkom sa random.org

MENTOR

Datum odobrenja rada:
Beograd, __.__.____.

Doc. dr Aleksandar Jevremović

DEKAN

Prof. dr Miroslav Lutovac

Razvoj sopstvenog rešenja za kriptografsku zaštitu sa implementiranim modulom za generisanje simetričnog kriptološkog ključa

Sažetak: U ovom radu fokus je stavljen na razvoj sopstvene aplikacije za kriptografsku zaštitu fajlova upotrebom simetričnih šifarskih algoritama (DES, AES). U aplikaciji će biti implementiran sopstveni modul za generisanje kriptoloških ključeva, zasnovan na vrednostima dobijenim preko pokreta miša. Razvoj modula za generisanje i upravljanje ključevima biće podvrgnut teorijsko-informacionoj analizi kakva se očekuje za primene ključeva u ovakve svrhe. Deo algoritma koji se odnosi na upravljanje ključevima obezbediće siguran način skladištenja. Za mesta skladištenja ključeva predviđeni su hardverski *tokeni* sa kontrolom pristupa (USB *token*). Pored razvoja i implementacije navedenih krypto komponenti, stavljen je fokus i na programsku ergonomiju koja će obezbediti pravilnu upotrebu programa u cilju smanjenja grešaka koje mogu da izazovu određene bezbednosne probleme na strani korisnika.

Ključne reči: simetrični šifarski sistemi, generisanje simetričnih kriptologških ključeva, pokreti miša

Solution development with the implemented cryptographic protection module for a symmetric key generation

Abstract: In this paper, we focused on the development of our own application for file-encryption using symmetric encryption algorithms (DES, AES). The application will implement its own module of cipher keys based on the values obtained via the mouse movement. Development of modules for generating and managing keys will be subjected to theoretical-informational analysis which is expected for uses the keys for such purposes. Part of the algorithm that refers to key management will provide a safe means of storage. Hardware tokens to control access (USB token) are provided for key storage. In addition to the development and implementation of these cryptographic components, the focus is placed on the software ergonomics that will ensure the proper use of the solution in order to reduce errors that can cause some security problems in the industry.

Keywords: symmetric cipher systems, symmetric key generation, mouse movement

SADRŽAJ

Spisak slika	- 3 -
Spisak tabela	- 4 -
1 Uvod	- 5 -
1.1 Predmet istraživanja	- 5 -
1.2 Doprinosi rada	- 6 -
1.3 Struktura rada	- 6 -
2 Pregled u oblasti istraživanja	- 7 -
2.1 Prednosti i nedostaci šifrovanja fajlova u Windows operativnom sistemu	- 7 -
2.2 Razvoj aplikacija za šifrovanje fajlova simetričnim kriptološkim ključem	- 8 -
2.2.1 AES Crypt	- 8 -
2.2.2 AxCrypt	- 9 -
2.2.3 CryptoForge	- 10 -
2.2.4 Folder Lock	- 11 -
2.2.5 TrueCrypt	- 13 -
2.3 Primena tehnologije zasnovane na simetričnom kriptološkom ključu	- 14 -
3 Teorijske osnove istraživanja	- 15 -
3.1 Simetrični šifarski sistemi	- 15 -
3.1.1 DES	- 16 -
3.1.2 AES	- 18 -
3.2 Režimi rada blokovskih šifara	- 21 -
3.2.1 Elektronska kodna knjiga (ECB)	- 22 -
3.2.2 Ulančavanje blokova šifrata (CBC)	- 22 -
3.2.3 Inicijalni vektor	- 23 -
3.2.4 Ispunjavanje	- 24 -
3.3 Heš funkcija	- 24 -
3.3.1 MD5	- 24 -

3.3.2	SHA.....	- 24 -
3.3.3	Kriptografska so	- 25 -
3.4	Generatori slučajnih brojeva	- 26 -
3.4.1	Pravi generator slučajnih brojeva (TRNG)	- 26 -
3.4.2	Pseudoslučajni generator brojeva (PRNG)	- 27 -
3.4.3	Entropija	- 27 -
3.5	NIST testovi	- 28 -
3.5.1	Ispitivanje učestalosti u nizu.....	- 28 -
3.5.2	Ispitivanje učestalosti u bloku	- 29 -
3.5.3	Ispitivanje uzastopnih ponavljanja istih bitova u nizu	- 30 -
3.5.4	Ispitivanje najdužeg uzastopnog ponavljanja jedinica u bloku	- 30 -
3.5.5	Ispitivanje stanja binarne matrice	- 32 -
3.5.6	Ispitivanje diskretne Furijerove transformacije.....	- 32 -
3.5.7	Ispitivanje nepreklapajućih uzoraka	- 33 -
3.5.8	Ispitivanje preklapajućih uzoraka	- 34 -
3.5.9	Maurerov univerzalni statistički test	- 35 -
3.5.10	Ispitivanje linearne kompleksnosti	- 35 -
3.5.11	Serijski test.....	- 36 -
3.5.12	Ispitivanje približne entropije	- 37 -
3.5.13	Ispitivanje slučajnih zbirova.....	- 38 -
3.5.14	Ispitivanje slučajne digresije	- 39 -
3.5.15	Ispitivanje slučajne promenljive digresije.....	- 40 -
4	Pregled predloženog rešenja	- 41 -
4.1	Generička šema predloženog rešenja	- 41 -
4.2	Postavka i objašnjenje eksperimentalnog okruženja	- 42 -
4.3	Performanse predloženog okruženja sa prikazom eksperimentalnih rezultata .	- 48 -
4.4	Oblast primene predloženog rešenja	- 50 -
5	Zaključak.....	- 51 -

5.1	Sumarni ciljevi rada i ostvareni rezultati	51 -
5.2	Predlog za budući rad	51 -
6	Literatura.....	52 -

Spisak slika

Slika 2.1	– AES Crypt – Šifrovanje fajla	8 -
Slika 2.2	– AES Crypt – Dešifrovanje fajla	8 -
Slika 2.3	– AxCrypt – Šifrovanje fajla	9 -
Slika 2.4	– AxCrypt – Unos lozinke prilikom šifrovanja fajla.....	10 -
Slika 2.5	– CryptoForge - Šifarski algoritmi.....	11 -
Slika 2.6	– Folder Lock – Zaključavanje fajlova	12 -
Slika 2.7	– Folder Lock – Šifrovanje fajlova.....	12 -
Slika 2.8	– TrueCrypt – Generisanje ključa pokretima miša	14 -
Slika 2.9	– TrueCrypt – Generisan ključ	14 -
Slika 3.1	– Proces šifrovanja i dešifrovanja u simetričnim šifarskim sistemima.....	15 -
Slika 3.2	– DES blokovska šifra	16 -
Slika 3.3	– Jedna runda DES algoritma	17 -
Slika 3.4	– E-kutija	17 -
Slika 3.5	– S-kutija	18 -
Slika 3.6	– P-kutija.....	18 -
Slika 3.7	– ByteStub operacija	19 -
Slika 3.8	– ShiftRow operacija.....	20 -
Slika 3.9	– MixColumn operacija	20 -
Slika 3.10	– AddRowKey operacija.....	20 -
Slika 3.11	– AES funkcija runde za runde 1,2,..., $n-1$	21 -
Slika 3.12	– ECB režim rada	22 -
Slika 3.13	– CBC režim rada	22 -
Slika 3.14	– Šematski prikaz rada režima za ulančavanje blokova	23 -
Slika 3.15	– Vizuelna uporedna analiza rezultata TRNG (levo) i PRNG (desno)	27 -
Slika 4.1	– Šema aplikacije - Generisanje ključa	41 -
Slika 4.2	– Šema aplikacije - Šifrovanje.....	42 -
Slika 4.3	– Šema aplikacije - Dešifrovanje.....	42 -
Slika 4.4	– Početni panel razvijenog rešenja	43 -
Slika 4.5	– Odabir nivoa sigurnosti	44 -
Slika 4.6	– Obaveštenje pre početka generisanja ključa	44 -
Slika 4.7	– Proces generisanja ključa	45 -
Slika 4.8	– Generisanje ključa uspešno završeno	45 -
Slika 4.9	– Zaštita ključa lozinkom	46 -
Slika 4.10	– Proces šifrovanja željenog fajla upotrebom generisanog ključa.....	46 -

Slika 4.11 – Unos lozinke za verifikaciju ključa	46 -
Slika 4.12 – Odabir jezika aplikacije	47 -
Slika 4.13 – O programu	47 -
Slika 4.14 – Prikaz dobijenog ključa preko histograma	49 -
Slika 4.15 – Prikaz vrednosti dobijenih preko X ose	50 -
Slika 4.16 – Prikaz vrednosti dobijenih preko Y ose	50 -
Slika 4.17 – Vizuelna uporedna analiza rezultata sa random.org (levo) i generisanog ključa (desno).....	50 -

Spisak tabela

Tabela 2.1 – TrueCrypt – Šifarski algoritmi.....	13 -
Tabela 3.1 – Heš algoritmi	25 -
Tabela 3.2 – Karakteristike dva tipa generatora slučajnih brojeva	27 -
Tabela 4.1 – Hardverske karakteristike razvojnog i eksperimentalnog okruženja	47 -
Tabela 4.2 – Rezultati serijskog testa	48 -
Tabela 4.3 – Rezultati ispitivanja entropije sa preklapanjem	49 -
Tabela 4.4 – Rezultati ispitivanja entropije bez preklapanja.....	49 -

1 Uvod

1.1 Predmet istraživanja

Informacije predstavljaju važan resurs u savremenom poslovanju. Bez obzira u kom se obliku čuvaju, moraju biti adekvatno zaštićene. Iz tog razloga zaštita informacija, očuvanje njene poverljivosti, integriteta i celovitosti je od presudne važnosti.

Istraživanjem dosadašnjeg stanja u ovoj oblasti i predstavljenih rešenja, došlo se do zaključka da je potrebno razviti rešenje za kriptografsku zaštitu fajlova i generisanje ključeva koji će potom biti podvrgnuti teorijsko-informacionoj analizi.

Razvijena aplikacija biće korišćena za šifrovanje svih vrsta fajlova simetričnim šifarskim algoritmima. Korisnik će imati mogućnost odabira nivoa sigurnosti koji želi da koristi i u skladu sa tim će se primenjivati odgovarajući algoritam (DES, AES 128 ili 192), a samim tim i odgovarajuća dužina ključa.

U aplikaciji će biti implementiran sopstveni modul za generisanje kriptoloških ključeva, zasnovan na vrednostima dobijenim preko pokreta miša. Proces generisanja ključa vršiće se sakupljanjem koordinata, u okviru panela za generisanje koji će se prikazati korisniku, na kojima je kliknuo mišem. Vreme potrebno za njegovo generisanje zavisice od nivoa sigurnosti. Nakon što je ključ generisan, od korisnika će se zahtevati da unese lozinku kojom će se dodatno štititi taj ključ. Na ovaj način biće korišćena dvofaktorska autentifikacija kombinovanjem nečega što korisnik ima i nečega što zna.

Generisan ključ će se čuvati isključivo na hardverskom *tokenu* (USB *token*). Za šifrovanje će moći da se izabere isključivo fajl koji se nalazi na nekoj drugoj lokaciji u odnosu na onu na kojoj je smešten ključ.

Takođe, pažnja će biti stavljena i na programsku ergonomiju radi obezbeđivanja pravilne upotrebe programa kako bi se sprečile eventualne greške od strane korisnika i na taj način onemogućili potencijalni bezbednosni problemi.

Aplikacija će biti razvijena na Microsoft .NET platformi, u programskom jeziku C#. Za prikaz dobijenih rezultata biće korišćen alat Matlab R2012b.

1.2 Doprinosi rada

U osnovne doprinose ovog rada spadaju:

- Generisanje ključa na čisto slučajan način
- Upotreba miša za dobijanje ključa
- Ključ se dodatno štiti lozinkom koju definiše korisnik
- Rešenje koje će pružiti visok nivo sigurnosti podataka koji se šifruju
- Visok nivo ergonomije prilikom upotrebe aplikacije

U nastavku rada svi doprinosi će biti detaljno opisani.

1.3 Struktura rada

U nastavku rada biće predstavljene neke od aplikacija za šifrovanje fajlova simetričnim kriptološkim ključem, zatim teorijske osnove istraživanja u kojima će biti obrađeni simetrični šifarski sistemi, režimi rada blokovskih šifara, heš funkcije, generatori slučajnih brojeva i testovi definisani od strane NIST-a (eng. *National Institute of Standards and Technology* - američka federalna agencija za standarde i tehnologiju).

Nakon toga biće predstavljeno i razvijeno rešenje uključujući i rezultate testova kojima su podvrgnuti ključevi u teorijsko-informacionoj analizi. Na kraju će biti prikazani ostvareni rezultati kao i predlog za budući rad.

2 Pregled u oblasti istraživanja

2.1 Prednosti i nedostaci šifrovanja fajlova u Windows operativnom sistemu

Jedna od novina uvedena u *Microsoft Windows 2000* operativnom sistemu i NTFS 5.0 fajl sistemu predstavlja *Encrypting File System* (EFS) tehnologija, koja je dizajnirana za brzo šifrovanje fajlova na čvrstom disku računara. EFS sistem koristi i javni i privatni ključ za šifrovanje i CriptoAPI arhitekturu. Upotreba odgovarajućeg simetričnog algoritma za šifrovanje se razlikuje u različitim verzijama *Windows*-a. Tako je *Microsoft Windows 2000* koristio DESX, *Windows XP* 3DES, a *Windows XP SP1*, *Windows 2003*, *Windows Vista* i *Windows 7* koriste AES. Šifrovanje datoteka ne zahteva od korisnika da izvrši preliminarne operacije. Tokom prvog šifrovanja datoteke, korisniku se automatski izdaju sertifikat i privatni ključ.

Jedna od značajnih karakteristika EFS je ta da fajlovi ostaju zaštićeni i kada se prekopiraju u drugi folder ili na drugi NTFS disk. U slučaju prenosa na drugi fajl sistem, fajlovi se automatski dešifruju. Kada korisnik dodaje nove datoteke u šifrovani direktorijum, oni bivaju automatski šifrovani. Od korisnika se ne zahteva da eksplicitno dešifruje fajl pre upotrebe, jer je EFS ugrađen u operativni sistem i automatski će izvršiti ovu funkciju, uz poštovanje svih bezbednosnih mera.

EFS tehnologija omogućava da fajlovi šifrovani od strane jednog korisnika ne mogu biti otvoreni od strane drugog ukoliko on ne poseduje odgovarajuće dozvole. Nakon što je šifrovan, fajl ostaje šifrovan bez obzira na to da li se nalazi na originalnoj ili je premešten na neku drugu lokaciju na disku. Mogu se šifrovati sve vrste fajlova, uključujući i izvršne.

Korisnik koji ima dozvolu za dešifrovanje fajla je u stanju da radi sa njim kao sa bilo kojim drugim, bez ikakvih ograničenja. Nasuprot tome, drugim korisnicima je pristup zabranjen. Ovaj pristup je veoma pogodan. Korisnik ima mogućnost da pouzdano i brzo ograniči pristup poverljivim informacijama drugih članova domaćinstva ili kolegama sa kojima deli računar.

EFS izgleda kao idealan alat, međutim to nije slučaj. Podaci šifrovani korišćenjem ove tehnologije mogu biti u potpunosti izgubljeni, na primer, prilikom ponovne instalacije operativnog sistema.

Fajlovi na disku su šifrovani pomoću FEK (eng. *File Encryption Key*). FEK se šifruje pomoću glavnog (eng. *master*) ključa, koji je šifrovan pomoću odgovarajućih ključeva korisnika. Sami korisnički ključevi su šifrovani hešovanom vrednošću korisničke lozinke, a hešovana lozinka koristi SYSKEY bezbednosnu funkciju.

Ovaj lanac šifrovanja trebalo bi pouzdano da štiti podatke, ali u praksi, kao što će biti objašnjeno u nastavku, zaštita se na kraju može svesti na upotrebu lozinke korisnika za prijavljivanje na sistem. Zbog ovakvog lanca šifrovanja, ukoliko

se lozinka izgubi ili promeni ili ukoliko operativni sistem padne ili se ponovo instalira, postaje nemoguće da se dobije pristup EFS šifrovanim fajlovima. U stvari, pristup može biti trajno izgubljen. Standardni korisnici ne razumeju u potpunosti kako funkcioniše EFS i često plaćaju cenu toga time što gube svoje podatke.

2.2 Razvoj aplikacija za šifrovanje fajlova simetričnim kriptološkim ključem

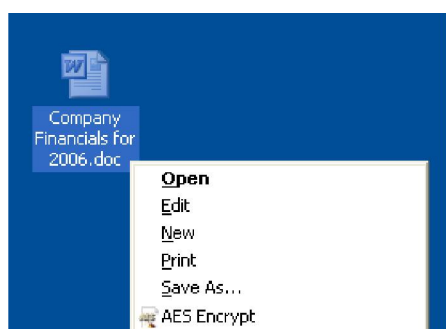
S obzirom na sistem za šifrovanje fajlova u *Windows* operativnom sistemu i njegove prednosti i mane, razvijeno je dosta aplikacija za šifrovanje simetričnim ključem. U nastavku će biti predstavljene neke od najboljih na tržištu.

2.2.1 AES Crypt

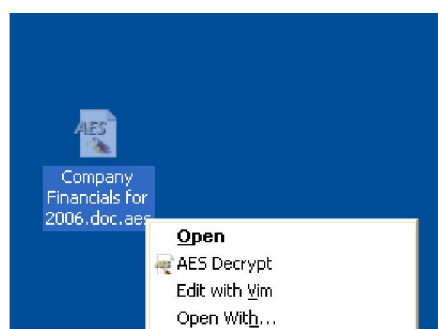
AES Crypt predstavlja *open-source* aplikaciju za šifrovanje fajlova AES blokovskom šifrom. Podržava rad na *Windows*, *Linux* i *MacOS* operativnim sistemima. Postoje 32-bitna i 64-bitna verzija aplikacije, kao i aplikacija sa grafičkim interfejsom i konzolna aplikacija. Aplikacija sa grafičkim interfejsom se zapravo integriše u *Windows Explorer*.

Prilikom šifrovanja korisnik unosi lozinku kojom se štite inicijalni vektor i ključ koji se generišu na slučajan način. Dužina lozinke može biti i samo jedan karakter. Nakon šifrovanja originalni fajl se zadržava, a šifrovani fajl u formatu *ime_fajla.originalna_ekstenzija.aes* se čuva na istoj lokaciji kao i originalni fajl.

Veličina bloka i IV su 128 bita, dok je dužina ključa 256 bita. Od funkcija za sažimanje koristi SHA-256.



Slika 2.1 – AES Crypt – Šifrovanje fajla



Slika 2.2 – AES Crypt – Dešifrovanje fajla

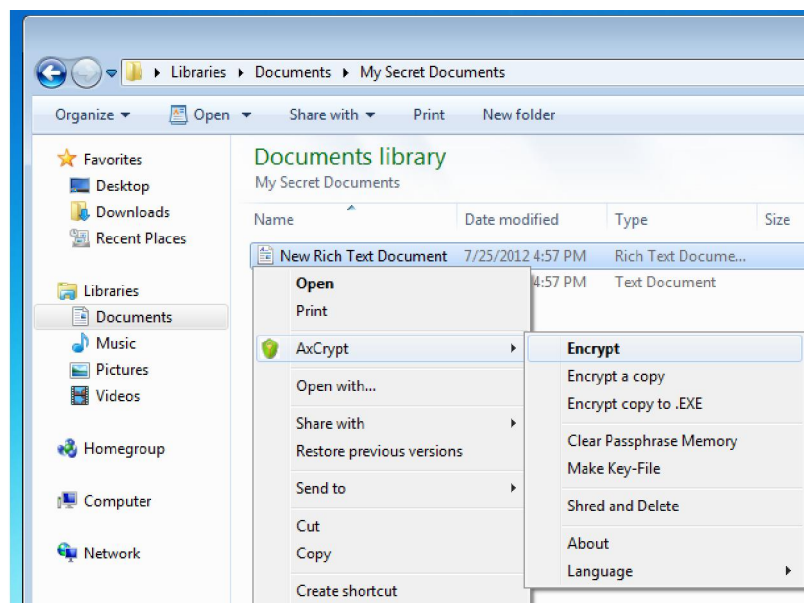
2.2.2 AxCrypt

AxCrypt predstavlja jednu od vodećih aplikacija za šifrovanje fajlova na *Windows* operativnom sistemu. Aplikacija je besplatna i nakon instalacije se integriše u *Windows Explorer*.

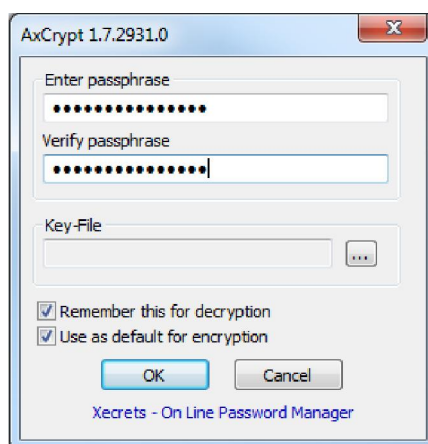
Prilikom šifrovanja unosi se lozinka i/ili se odabira unapred generisani fajl sa ključem kojim se štiti šifrat. Dužina lozinke nema ograničenja, iako se korisnicima savetuje da šifra ne sadrži manje od 5 karaktera, samog ograničenja u aplikaciji nema.

Postoji mogućnost generisanja random ključa od 128 bita. Moguće ga je sačuvati jedino na nekom od prenosnih medijuma (USB *token*). Ukoliko se zajedno koriste lozinka i generisani ključ, vrši se njihova konkatencija, zatim se hešuju da bi se takav ključ koristio prilikom šifrovanja odnosno dešifrovanja. AxCrypt koristi AES 128 blokovsku šifru i SHA-1 funkciju za sažimanje.

Šifrovani fajl dobija naziv u formatu *ime_fajla-originalna_ekstenzija.axx*. Originalni fajl se ne čuva već biva zamenjen šifrovanim. Aplikacija podržava šifrovanje više fajlova odjednom.



Slika 2.3 – AxCrypt – Šifrovanje fajla



Slika 2.4 – AxCrypt – Unos lozinke prilikom šifrovanja fajla

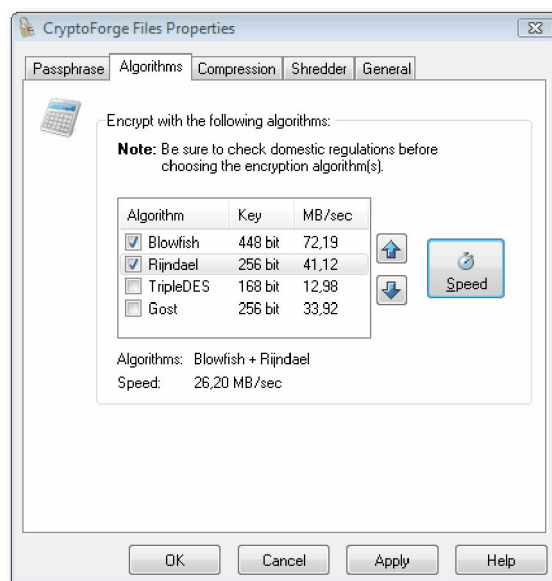
2.2.3 CryptoForge

CryptoForge predstavlja jednostavnu aplikaciju za upotrebu za šifrovanje fajlova kako za privatne tako i za poslovne korisnike. Omogućava zaštitu fajlova, foldera ili poruka upotrebom do četiri simetrična algoritma (i CBC režima rada):

- **Blowfish** (448-bitni ključ)
- **Rijndael** (256-bitni ključ)
- **Trostruki DES** (168-bitni ključ)
- **Gost** (256-bitni ključ)

Omogućava rad sa svim tipovima fajlova, bilo da se nalaze na lokalnom računaru ili na nekom od računara u mreži. Maksimalna veličina fajla koju podržava je 16TB.

Prilikom pokretanja aplikacije, korisnik unosi lozinku, koja se hešuje i koja predstavlja ključ za šifrovanje i dešifrovanje sadržaja. Sadrži ugrađenu funkciju koja pruža mogućnost kompresovanja sadržaja pre šifrovanja čime se dodatno povećava sigurnost. Šifrovani fajl dobija naziv u formatu *ime_fajla-originalna_ekstenzija.cfe*. Originalni fajl se ne čuva već biva zamenjen šifrovanim. Takođe, postoji mogućnost šifrovanja i samog naziva fajla. U tom slučaju fajlu će biti dodeljen nasumično generisan naziv sa ekstenzijom *.cfe*.



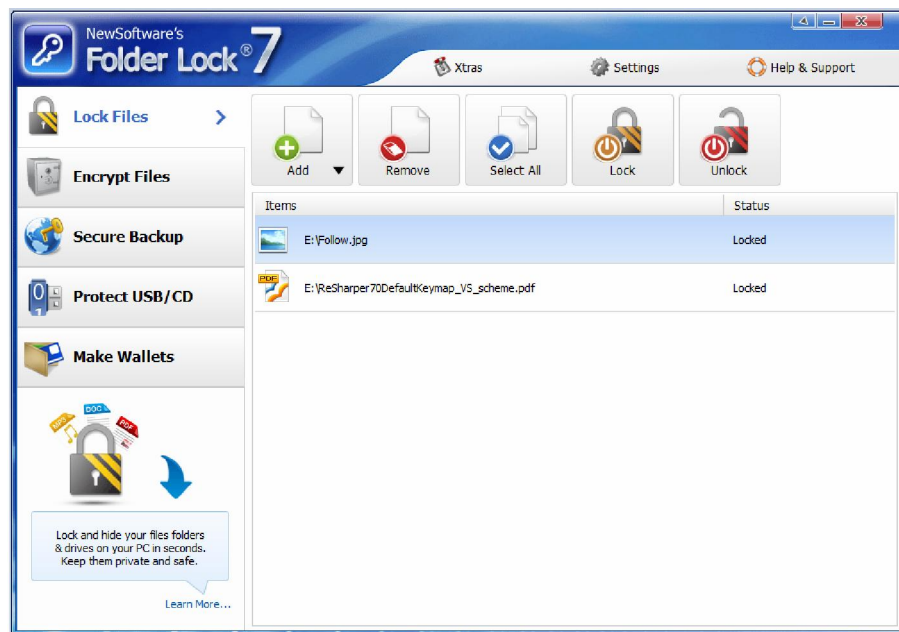
Slika 2.5 – CryptoForge - Šifarski algoritmi

2.2.4 Folder Lock

Folder Lock je koncept bezbednosti podataka koji omogućava zaključavanje, skrivanje i zaštitu fajlova, foldera i diskova lozinkom, šifrovanje fajlova i foldera, a opciono omogućava kreiranje rezervnih kopija šifrovanih fajlova na sigurnom online nalogu. Pored zaključavanja, šifrovanja i sigurne rezervne kopije, Folder Lock nudi dodatne funkcije za zaštitu podataka, kao što su zaštita USB / CD, zaključavanje fajlova, kreiranje novčanika, seckanje fajlova.

Radi na *Windows* operativnom sistemu, kako na 32-bitnom, tako i na 64-bitnom. Za šifrovanje fajlova koristi 256-bitnu AES enkripciju i 128-bitnu enkripciju za sve online komunikacije.

Nakon prvog pokretanja aplikacije od korisnika se zahteva da unese glavnu lozinku (eng. *Master password*) čija dužina mora biti u rasponu od 6-100 karaktera. Nakon toga, sva administrativna prava (uključujući šifrovanje i zaključavanje fajlova) se vezuju za ovu lozinku. Ona postaje glavna lozinka za celu aplikaciju, a može se podesiti ili menjati od strane korisnika koji je prvi instalirao aplikaciju. S obzirom da celo funkcionisanje, a samim tim i sigurnost aplikacije zavisi od njene jačine, neophodno je da bude dovoljno jaka da se odupre bilo kojoj vrsti potencijalnog napada (npr. *brute-force*, napad rečnikom, itd). Kao prevencija napada na lozinku, korisnik ima mogućnost podešavanja opcije za sprečavanje višestrukog unosa pogrešne lozinke.



Slika 2.6 – Folder Lock – Zaključavanje fajlova



Slika 2.7 – Folder Lock – Šifrovanje fajlova

2.2.5 TrueCrypt

TrueCrypt je softverski sistem za uspostavljanje i održavanje *on-the-fly* šifrovanog diska što znači da su podaci automatski šifrovani pre nego što su sačuvani i dešifrovani odmah nakon što su učitani, bez dodatne intervencije od strane korisnika. Svi podaci koji se šifruju odnosno dešifruju se, umesto na disku, čuvaju u RAM memoriji. Bez unosa ispravne lozinke i fajla sa ključem, nije moguće pristupiti nijednom podatku koji se nalazi na šifrovanom disku.

TrueCrypt particije šifruju se nekim od navedenih algoritama:

Algoritam	Dužina ključa	Veličina bloka
AES	256	128
Serpent	256	128
Twofish	256	128
AES-Twofish	256; 256	128
AES-Twofish-Serpent	256; 256; 256	128
Serpent-AES	256; 256	128
Serpent-Twofish-AES	256; 256; 256	128
Twofish-Serpent	256; 256	128

Tabela 2.1 – TrueCrypt – Šifarski algoritmi

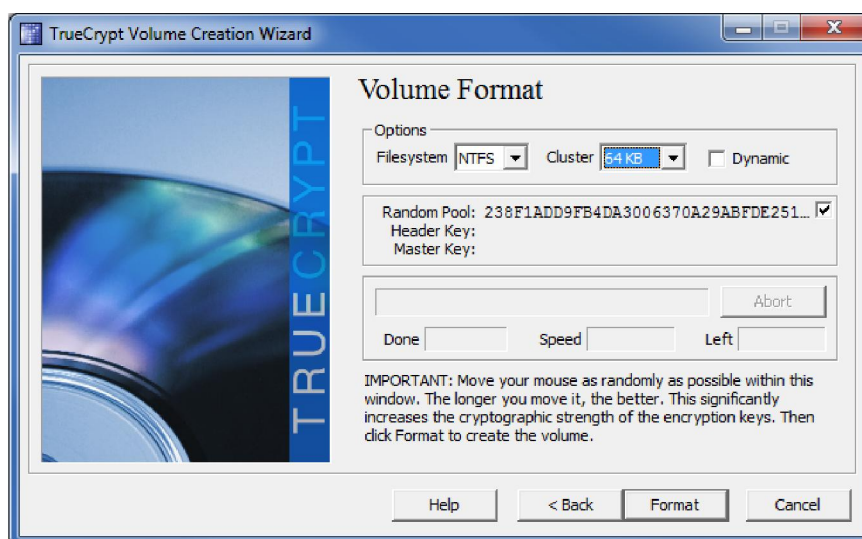
Aplikacija podržava sledeće algoritme za sažimanje:

- RIPEMD-160
- SHA-512
- Whirlpool

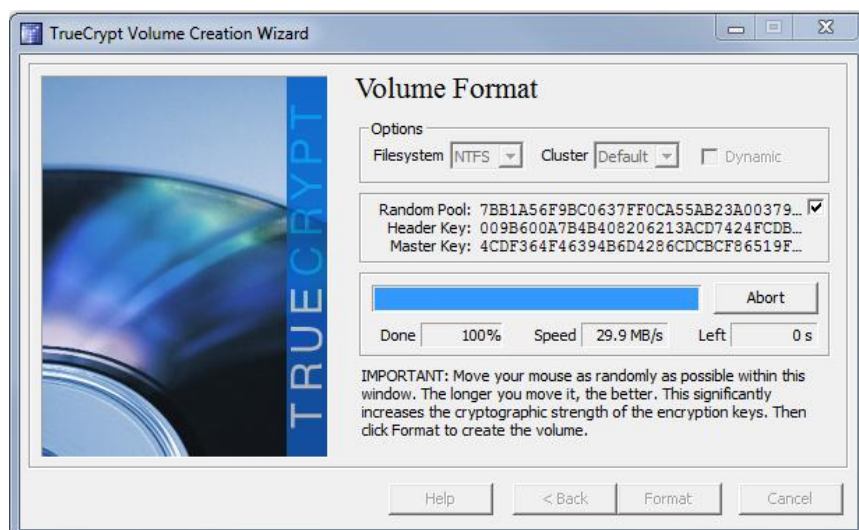
Generator slučajnih brojeva (RNG) koristi se za generisanje glavnog ključa za šifrovanje, pomoćnog ključa, za kriptografsku so kao i za ključ. U RAM memoriji se kreira prostor (eng. *pool*) od 320 bajtova u koji se smeštaju podaci iz sledećih izvora:

- Pokreti miša
- Pritisci tastera na tastaturi
- Na *Mac* i *Linux* operativnim sistemima: vrednosti generisane od strane ugrađenog generatora slučajnih vrednosti
- Na *Windows* operativnom sistemu: statistika mrežnog interfejsa, vremenske promenljive...

Prilikom kreiranja particije na kojoj će se nalaziti fajlovi koji se šifruju, potrebno je definisati lozinku za tu particiju. Maksimalan broj karaktera je 64, a prilikom unosa kratke lozinke korisnik se upozorava ukoliko je uneo manje od 20 karaktera.



Slika 2.8 – TrueCrypt – Generisanje ključa pokretima miša



Slika 2.9 – TrueCrypt – Generisan ključ

2.3 Primena tehnologije zasnovane na simetričnom kriptološkom ključu

Rešenja bazirana na upotrebi simetričnog kriptološkog ključa pogodna su za privatne i poslovne korisnike koji imaju potrebu za zaštitom određenih podataka koje čuvaju ili na svom računaru ili na nekom prenosnom medijumu. Poslovni korisnici imaju potrebu za zaštitom poverljivih kompanijskih podataka, dok privatni korisnici žele da neke svoje podatke zaštite ukoliko se oni nalaze na računaru koji dele sa npr. drugim članovima porodice.

Ovakva rešenja su pogodna ukoliko ne postoji potreba za razmenom ključa sa drugim korisnicima. U tom slučaju bi se zahtevao njegov transfer kroz neki siguran komunikacioni kanal. Za takve potrebe možda je dobro razmisliti o upotrebi asimetričnog kriptološkog ključa.

3 Teorijske osnove istraživanja

Svojstvo da se informacije mogu čitati samo od autorizovanih korisnika ili primaoca predstavlja tajnost podataka. Zaštita tajnosti informacija postiže se na taj način što se u procesu šifrovanja otvorena poruka, upotrebom odgovarajućeg ključa, transformiše u novi oblik informacije - šifrat. Dešifrovanje predstavlja reverzni proces u odnosu na šifrovanje - transformiše šifrat u otvoreni tekst. Šifrovanjem se postiže zaštita tajnosti i integritet informacija.

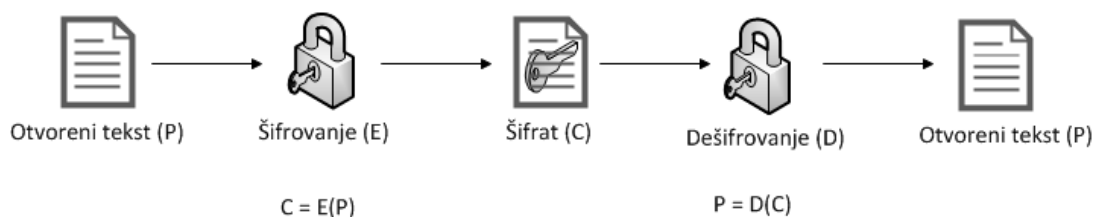
Tajnost poruke zasniva se isključivo na tajnosti ključa s obzirom da je rad svih šifarskih algoritama koji se danas koriste poznat. Ključ predstavlja početnu vrednost algoritma kojim se vrši šifrovanje, a tajnost se ogleda u tome što njega poseduje samo osoba koja vrši šifrovanje.

Tehnike šifrovanja imaju veliki značaj iz razloga što imaju glavnu ulogu u zaštiti od informatičkih prevara, povrede sigurnosti podataka, zaštiti pouzdanosti korespondencije, profesionalne tajne i u elektronskoj trgovini.

3.1 Simetrični šifarski sistemi

Simetrični šifarski sistemi predstavljaju sisteme šifrovanja tajnim ključem pri čemu je ključ za šifrovanje identičan ključu za dešifrovanje. Karakterišu se relativno velikom brzinom rada i jednostavnom implementacijom.

Kod šifrovanja poruke upotrebom simetričnih sistema, tajnost i autentičnost poruke zasnivaju se na autentičnosti ključa, a sistem je bezbedniji ukoliko se ključ generiše na što slučajniji način.



Slika 3.1 – Proces šifrovanja i dešifrovanja u simetričnim šifarskim sistemima

Simetrični šifarski sistemi se dele na:

- Klasične
- Sekvencijalne
- Blokove

U nastavku će biti opisan rad blokovskih šifara.

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 i šifrat se dele na blokove fiksne dužine (64, 128, 192, 256) u bitovima, zatim se svaki blok nezavisno šifrue. Šifrat se dobija na osnovu otvorenog teksta iterativnom primenom funkcije koja se naziva runda. Ulaz u funkciju runde je ključ i izlaz prethodne runde. Blokovske šifre kombinuju svojstva konfuzije i difuzije i kompletnost. Najčešće se realizuju softverski.

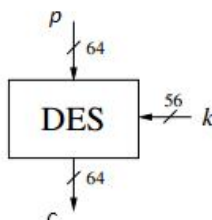
3.1.1 DES

Data Encryption Standard (DES) razvijen je 1970-tih godina. Predstavlja blokovsku šifru sa simetričnim ključem usvojenu od strane NIST-a (eng. *National Institute of Standards and Technology* - američka federalna agencija za standarde i tehnologiju). Pripada grupi modernih šifara.

Cilj dizajna je bio da se uvedu principi *konfuzije* i *difuzije*.

- **Konfuzija** – Šifrat treba da zavisi od otvorenog teksta i ključa na složen način.
- **Difuzija** – Svaki bit šifrata treba da je funkcija svih bitova otvorenog teksta i svih bitova ključa.

DES vrši šifrovanje blokova dužine 64 bita ključem dužine 56 bita.



Slika 3.2 – DES blokovska šifra

Svaki blok otvorenog teksta se transformiše u 16 rundi koje obavljaju identičnu operaciju. U svakoj rundi se koristi 48 bita ključa (podključ).

DES je Fejstel šifra. Fejstel šifra (nazvana po pioniru blokovskih šifara Horstu Fejstelu) ne predstavlja određenu šifru nego generalni princip dizajniranja šifri. Kod Fejstel šifri otvoreni tekst P se deli na blokove tako što se svaki blok deli na dva dela – levi L i desni R .

$$P = (L_0, R_0)$$

U svakoj rundi i ($i = 1, 2, \dots, n$), leva i desna strana se izračunavaju na sledeći način:

$$L_i = R_{i-1}$$

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

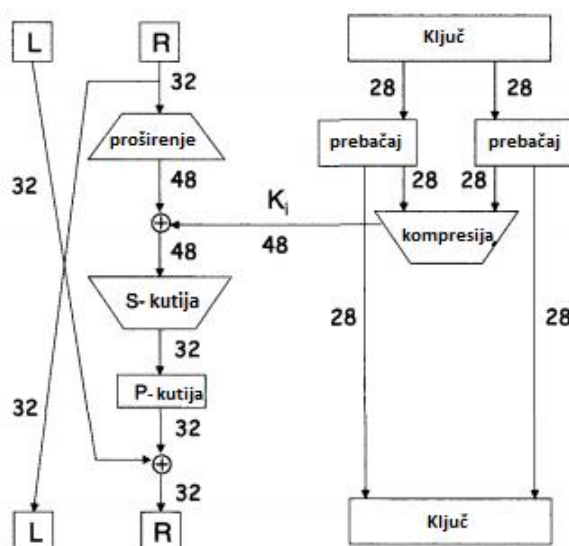
F je funkcija runde.

K_i je podključ za rundu i . Podključ je izveden iz ključa K prema specifičnom algoritmu. Izlaz iz runde je šifrat C .

$$C = (L_n, R_n)$$

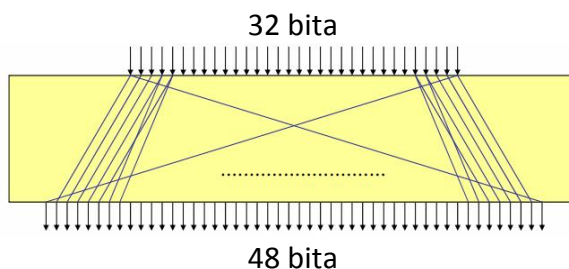
Pošto je DES Fejstel šifra, šifrovanje prati pravila prikazana u jednačini:

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

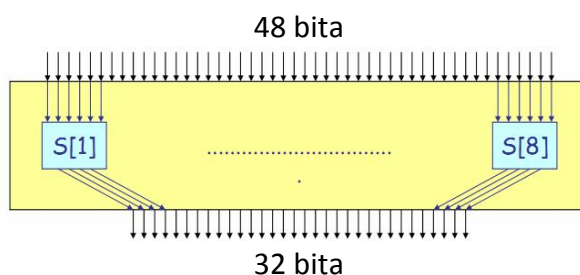


Slika 3.3 – Jedna runda DES algoritma

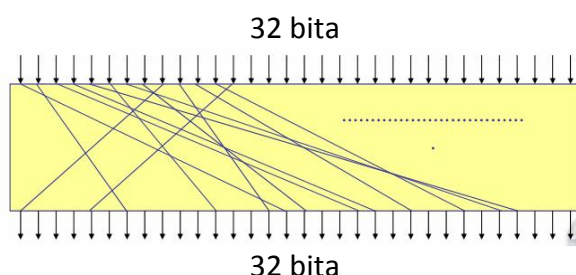
E-kutija (eng. *expand*) proširuje i permutuje 32 ulazna bita na 48 bita i podključ je XOR-ovan sa rezultatom. Potom S-kutije (eng. *substitution*) kompresuju ta 48 bita na 32 bita pre nego što se rezultat prosledi P-kutiji (eng. *permutation*). Izlaz iz P-kutije je XOR-ovan sa starom levom polovinom da bi se dobila nova desna polovina.



Slika 3.4 – E-kutija



Slika 3.5 – S-kutija



Slika 3.6 – P-kutija

Trideset godina intenzivne kriptanalize nije otkrilo nikakve propuste tipa “back door”. Sa aspekta sigurnosti, danas je DES ranjiv na *brute-force* napade 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 nađu na onu pravu, odnosno, primenjuju potpunu pretragu ključeva.

3.1.2 AES

Advanced Encryption Standard (AES) je najrasprostranjenija simetrična šifra. Iako se izraz „Standard“ odnosi samo na Vladu SAD-a, AES je obavezan za upotrebu i u raznim industrijskim standardima, a takođe se koristi i u komercijalne svrhe. U komercijalne standarde koji koriste AES spadaju standard za Internet sigurnost *IPsec*, *TLC*, standard za šifrovanje *Wi-Fi* komunikacije IEEE 802.11i, *SSH* (eng. *Secure Shell*) mrežni protokol, Internet komunikacija preko *Skype*-a i razni drugi sigurnosni proizvodi.

Razvijen je krajem 1990-tih godina od strane NIST-a i zasnovan na *Rijndael* (izgovara se slično kao eng. *Rain doll*) algoritmu. AES je kao i DES, blokovska šifra sa simetričnim ključem standardizovana od strane NIST-a. Za razliku od DES-a, 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.

Otporan na poznate napade, veoma je brz, moguć je paralelni dizajn, kao i implementacija na mnogim procesorima i pametnim karticama. Do danas nisu uočeni bolji napadi od *brute-force* napada na AES.

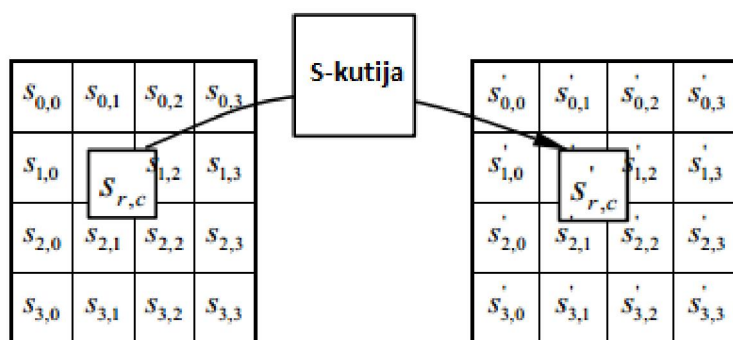
Funkcionalni parametri AES-a su:

- Dužina bloka otvorenog teksta je 128, 192 i 256 bita
- Dužina ključa je 128, 192 i 256 bita
- Ima ukupno od 10 do 14 rundi, zavisno od dužine ključa
- U svakoj rundi koriste se 4 funkcije
 - Nelinearni sloj (eng. *ByteSub*)
 - Sloj linearnog mešanja (eng. *ShiftRow*)
 - Nelinearni sloj (eng. *MixColumns*)
 - Dodatni sloj ključa (eng. *AddRoundKey*).

Bitna karakteristika AES-a je da je dužina bloka otvorenog teksta u potpunosti nezavisna od dužine ključa. To znači da je moguće šifrovati blok otvorenog teksta dužine 128 bita sa ključem dužine 256 bita.

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 sprovedu razne operacije. Završna vrednost matrice ili stanje, kopira se u izlazni šifarski blok. AES ulazne podatke tretira kao matrice dimenzije 4x4.

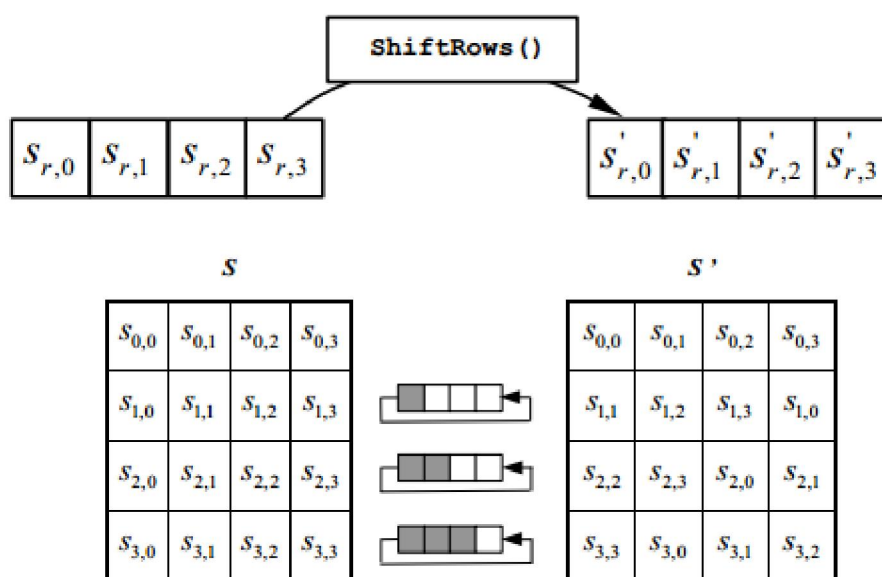
ByteSub operacija se primenjuje na svaki bajt a_{ij} tako da je $b_{ij} = \text{ByteSub}(a_{ij})$. Kao rezultat se dobija niz b_{ij} prikazan na slici 3.7.



Slika 3.7 – *ByteSub* operacija

ByteSub operacija je za AES isto što i S-kutija za DES. Može se posmatrati kao nelinearna, ali invertibilna, kompozicija dve matematičke operacije ili kao tabela.

ShiftRow operacija predstavlja ciklično pomeranje bajtova u svakom redu matrice 4x4. U prvom redu ne postoji pomeranje, u drugom se pomera jedan bajt, u trećem dva i u poslednjem redu tri bajta se pomeraju u levo.



Slika 3.8 – ShiftRow operacija

MixColumn operacija se primenjuje na svaku kolonu matrice 4x4 kao što je prikazano na slici 3.9. Ovu operaciju čine operacije pomeranja i XOR-ovanja.

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \quad \text{za } 0 \leq c < Nb.$$

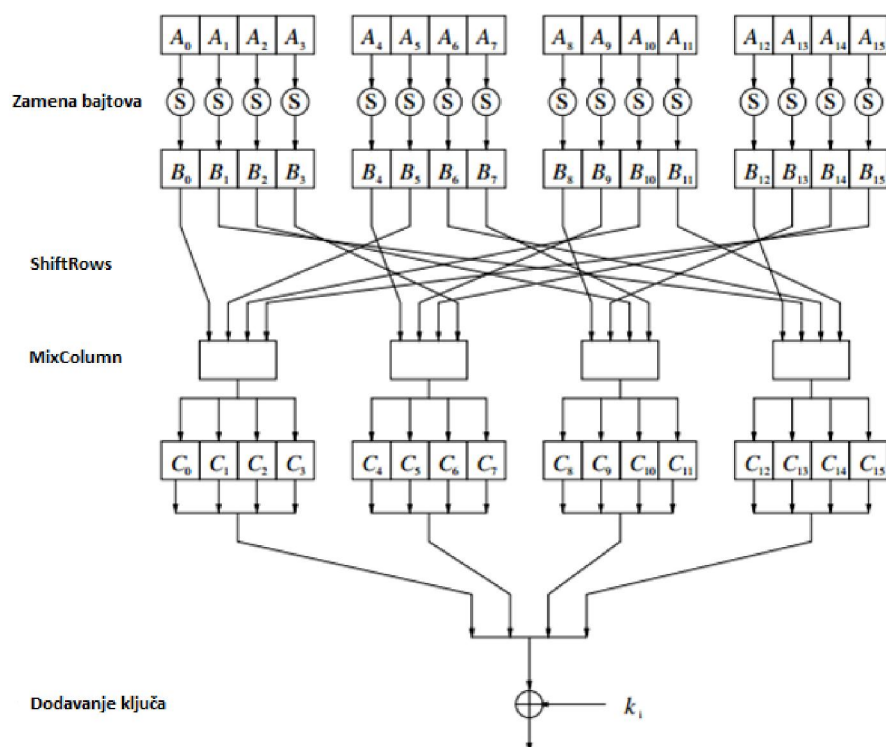
Slika 3.9 – MixColumn operacija

AddRoundKey je jednostavna operacija. Kao kod DES-a, algoritam rasporeda ključa je korišćen da bi se generisao podključ za svaku rundu.

Ako je K_{ij} matrica 4x4 podključa za određenu rundu, onda se podključ XOR-uje sa trenutnom matricom 4x4 a_{ij} koja predstavlja ulaz kao što je prikazano u nastavku:

$$\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \oplus \begin{bmatrix} k_{00} & k_{01} & k_{02} & k_{03} \\ k_{10} & k_{11} & k_{12} & k_{13} \\ k_{20} & k_{21} & k_{22} & k_{23} \\ k_{30} & k_{31} & k_{32} & k_{33} \end{bmatrix} = \begin{bmatrix} b_{00} & b_{01} & b_{02} & b_{03} \\ b_{10} & b_{11} & b_{12} & b_{13} \\ b_{20} & b_{21} & b_{22} & b_{23} \\ b_{30} & b_{31} & b_{32} & b_{33} \end{bmatrix}$$

Slika 3.10 – AddRoundKey operacija

Slika 3.11 – AES funkcija runde za runde 1,2,..., n_r-1

3.2 Režimi rada blokovskih šifara

Upotrebom blokovskih šifara (DES, AES) uvek postoji potreba da se šifruje više od jednog bloka otvorenog teksta. Prilikom šifrovanja odnosno dešifrovanja, svaki otvoreni tekst se deli na blokove veličine n bita. Otvoreni tekst koji prelazi n bita se deli na n -bitne blokove. Ukoliko dužina otvorenog teksta nije deljiva sa n bita, mora biti dopunjena do veličine bloka.

Postoji više načina da se šifruje dugačak otvoreni tekst sa blokovskim šiframa. Najpoznatiji režimi rada su:

- **ECB** – Elektronska kodna knjiga (eng. *Electronic Code Book*)
- **CBC** – Ulančavanje blokova šifrata (eng. *Cipher Block Chaining*)
- **CFB** – Sprega šifrata (eng. *Cipher Feedback*)
- **OFB** – Sprega izlazne vrednosti (eng. *Output Feedback*)
- **CTR** – Brojački (eng. *Counter*)

Svi režimi rada su nezavisni (odnosno primenljivi) na svim blokovskim algoritmima.

3.2.1 Elektronska kodna knjiga (ECB)

Elektronska kodna knjiga (eng. *Electronic codebook* - ECB) predstavlja najjednostavniji režim rada blokovskih šifara. Princip rada ovog režima je u tome da se za isti blok otvorenog teksta dobija isti blok šifrata. U ovom režimu svaki blok šifrata je potpuno nezavisan.

U ECB režimu, šifrovanje se vrši na sledeći način

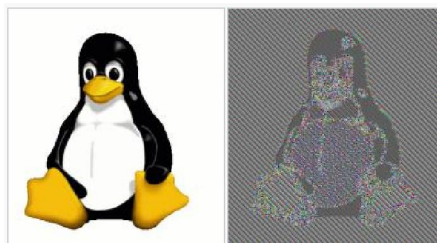
$$C_i = E(P_i, K) \quad \text{za } i = 0, 1, 2$$

a dešifrovanje

$$P_i = D(C_i, K) \quad \text{za } i = 0, 1, 2$$

Problem sa ECB režimom se ogleda u tome što se prilikom kriptanalize poređenjem više otvorenih tekstova i šifrata bez upotrebe ključa može primetiti da se delovi poruke ponavljaju. Napadač može zaključiti da je $C_i = C_j$ i iz toga zaključuje da je $P_i = P_j$. Različite poruke imaju zajedničke bitne sekvence. Redudansa poruka može biti velika.

Ilustracija ECB režima prikazana je na slici 3.12.

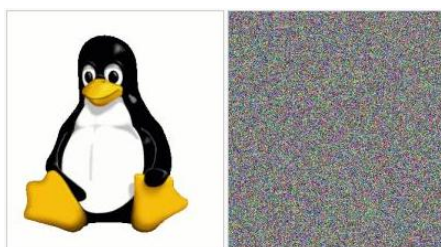


Slika 3.12 – ECB režim rada

3.2.2 Ulančavanje blokova šifrata (CBC)

Režim ulančavanja blokova šifrata (eng. *Cipher-block chaining* - CBC) predstavlja najčešće korišćen režim. Radi na principu ulančavanja blokova šifrata. Šifrat od prethodnog bloka otvorenog teksta koristi se za "zamagljivanje" otvorenog teksta sledećeg bloka, pre nego što se primeni algoritam za šifrovanje.

Ilustacija CBC režima prikazana je na slici 3.13.



Slika 3.13 – CBC režim rada

Formula za šifrovanje u CBC režimu je

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

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

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

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

a za dešifrovanje

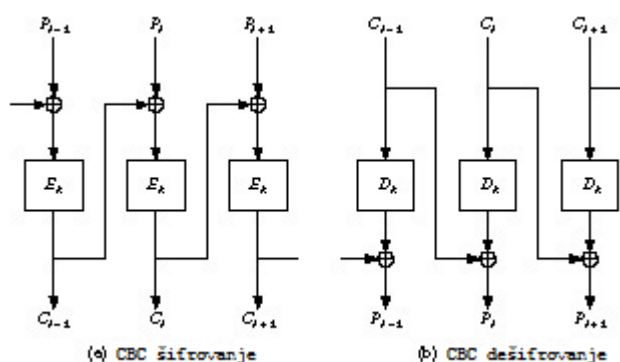
$$P_i = D(C_i, K) \oplus C_{i-1} \quad \text{za } i = 0, 1, 2$$

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

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

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

U režimu za ulančavanje blokova šifrata, otvoreni tekst se prvo XOR-uje predhodnim blokom šifrata pre nego što se šifruje. Nakon što se blok otvorenog teksta šifruje, njegov šifrat se smešta u registar povratnih vrednosti. Pre šifrovanja novog bloka otvorenog teksta, XOR-uje se sa povratnom vrednošću da bi se dobio novi ulaz za naredni blok. Prednost CBC režima je ta da identičan otvoreni tekst neće identično odgovarati šifratu.



Slika 3.14 – Šematski prikaz rada režima za ulančavanje blokova

3.2.3 Inicijalni vektor

Inicijalni vektor se koristi samo za generisanje prvog bloka šifrata. Poželjno je da bude generisan na čisto slučajan način i nije neophodno da bude tajan. Dužina inicijalnog vektora (IV) je uvek jednaka veličini bloka date šifre.

3.2.4 Ispunjavanje

Većina otvorenog teksta koji je potrebno šifrovati nije deljiv sa veličinom bloka i iz tog razloga poslednji blok nema dovoljan broj bita. Rešenje ovog problema ogleda se u ispunjavanju tog bloka. Ispunjavanje (eng. *padding*) se može vršiti nulama, jedinicama i nulama. PKCS7 predstavlja jedan od najpopularnijih režima ispunjavanja.

3.3 Heš funkcija

S obzirom da šifrovanje ne štiti integritet poruke, za zaštitu integriteta (autentičnosti sadržaja poruke) koristi se heš funkcija (funkcija za sažimanje poruke odnosno za generisanje skraćene verzije poruke). Za otvoreni tekst različite dužine, nakon primene heš funkcije dobija se tekst fiksne dužine. Glavna karakteristika heš funkcije u kriptografskim aplikacijama je ta da mora biti jednosmerna.

$$y = h(M)$$

h – heš funkcija

M – poruka

Y – rezultat heš funkcije

Najpoznatija grupa heš funkcija je MD4 grupa. MD5, SHA i RIPEMD pripadaju ovoj grupi.

3.3.1 MD5

MD5 (eng. *Message Digest*) predstavlja jednosmernu funkciju koja za ulaz proizvoljne dužine daje izlaz od 128 bita. Razvijena je 1991. godine od strane Rolanda Rivesta. Bila je široko rasprostranjena (za sigurnost Internet protokola, za poređenje sačuvanih lozinki). Poznato je da ima slabosti i da nije otporna na kriptografske napade. I pored ovoga, zbog svoje brzine rada se koristi u proveru integriteta većih fajlova.

3.3.2 SHA

Nakon što su primećene slabosti kod MD5 funkcije, 1993. godine NIST je uveo novi standard - SHA (eng. *Secure Hash Algorithm*). Prvi član ove porodice je nazvan SHA, iako je on danas poznat pod nazivom SHA-0. Modifikovan je 1995. god. i dobijen je SHA-1. Razlika između SHA-0 i SHA-1 ogleda se u rasporedu funkcija za kompresovanje koji obezbeđuje veću kriptografsku sigurnost.

NIST je 2002. god. uveo novu familiju algoritama (SHA-2) koje se razlikuju po dužini heš vrednosti: SHA-256, SHA-384 i SHA-512. SHA-224 je objavljen 2004. god. Ovi algoritmi su bezbedniji zbog dužine heš vrednosti i određenih promena u algoritmu. Takođe su kompatibilni sa AES blokvsom šifrom.

SHA-1 predstavlja industrijski standard u kriptografiji i pored toga što algoritmi iz SHA-2 porodice pružaju veću sigurnost. 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.

Algoritam		Izlaz (u bitovima)	Inicijalna veličina u bitovima	Broj rundi	Kolizije
SHA-0		160			
SHA-1		160	512	80	Da
SHA-2	SHA-224	224	512	64	Za sad ne
	SHA-256	256	512	64	Ne
	SHA-384	384	1024	80	Ne
	SHA-512	512	1024	80	Ne

Tabela 3.1 – Heš algoritmi

3.3.3 Kriptografska so

Prilikom logovanja korisnika na sistem, vrši se provera validnosti njegove lozinke. Ukoliko se u bazi čuvaju heš vrednosti lozinke, prilikom autentifikacije računa se heš vrednost unete lozinke od strane korisnika i poredi se sa vrednošću iz baze. Ukoliko se one poklapaju, korisnik je autentifikovan i omogućen mu je pristup sistemu.

S druge strane, ukoliko napadač želi da pristupi, može koristiti napad sa rečnikom (eng. *dictionary attack*). Da bi mu se otežao posao koristi se kriptografska so (eng. *salt*) koja predstavlja slučajne vrednosti koje se dodaju lozinci pre upotrebe heš funkcije. Kriptografska so i rezultat heš funkcije se smeštaju u bazu. Ukoliko je broj mogućih „posoljenih“ vrednosti dovoljno veliki, ovim se praktično eliminiše napad sa rečnikom, jer napadač za svaku kriptografsku so mora da generiše heš vrednost.

„Posoljena“ vrednost ne mora da bude tajna. Ona se može posmatrati kao inicijalni vektor (IV) kod blokovskih šifara čime se postiže da se isti blokovi otvorenog teksta šifruju na različit način. Po tom principu, primenom „posoljene“ vrednosti, identične lozinke dobijaju različite heš vrednosti.

$$y = h(p,s)$$

p – lozinka

s – slučajna vrednost

h – heš funkcija

y – rezultat heš funkcije lozinke i slučajne vrednosti

3.4 Generatori slučajnih brojeva

Slučajnost predstavlja suštinu za generisanje ključeva i jednu od glavnih sigurnosnih komponenti za mnoge kriptografske protokole. Ukoliko se za generisanje ključa koristi neodgovarajuća slučajnost, celokupna sigurnost protokola može biti ugrožena. Sa ovog stanovišta, generisanje ovakvih slučajnih podataka je od velike važnosti.

Generatori slučajnih brojeva (eng. *Random Number Generator* - RNG) se koriste u velikom broju aplikacija za simulacije i kriptografiju. Predstavljaju važan deo mnogih kriptografskih sistema, kao što su generisanje ključeva, inicijalizacija vektora, za ispunjavanje poruka i drugo.

Generatori slučajnih brojeva su uređaji koji generišu niz brojeva ili neku vrstu simbola koji se pojavljuju nasumice. Koriste se u razne svrhe, kao što su simulacija, modelovanje složenih fenomena, kriptografija i naravno zauvek popularne igre i kockanje. Postoje dva osnovna pristupa za generisanje slučajnih brojeva:

- Pravi generator slučajnih brojeva (eng. *True Random Number Generator* - TRNG)
- Pseudoslučajni generator brojeva (eng. *Pseudo Random Number Generator* - PRNG)

3.4.1 Pravi generator slučajnih brojeva (TRNG)

Pravi generator slučajnih brojeva (eng. *True Random Number Generator* - TRNG) izvlači slučajnosti iz fizičkih fenomena kao što su atmosferski šum, male varijacije u pokretima miša ili vreme između poteza mišem. Uglavnom se koristi u igrama i kockanju gde je potrebna istinska slučajnost, jer su previše spori za upotrebu u statističkim i kriptografskim aplikacijama.

TRNG karakteriše činjenica da njegov izlaz ne može da se reprodukuje. Na primer, ako se novčić baci 100 puta i snimi dobijena sekvenca od 100 bita, biće gotovo nemoguće da bilo ko generiše istu sekvencu od 100 bita. Šanse za uspeh su 1/2100, što predstavlja izuzetno malu verovatnoću. TRNG je zasnovan na fizičkim procesima. Neki od primera su bacanje novčića, bacanje kockice, šum poluprovodnika... U kriptografiji, TRNG je često potreban za generisanje sesijskih ključeva, koji se potom distribuiraju korisnicima.

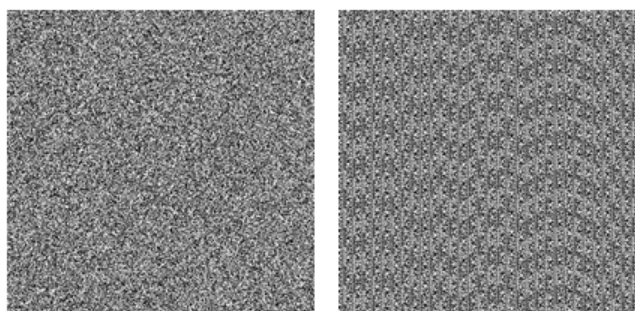
3.4.2 Pseudoslučajni generator brojeva (PRNG)

Pseudoslučajni generator brojeva (eng. *Pseudo Random Number Generator* - PRNG) predstavlja generator za generisanje pseudoslučajnih brojeva na osnovu inicijalnog stanja koje je određeno na slučajan način.

U odnosu na TRNG, PRNG je algoritam koji koristi neku matematičku formulu ili za generisanje niza brojeva koji se pojavljuju nasumice. Niz brojeva proizvedenih nije zaista slučajan. Potpuno je određen proizvoljnim početnim stanjem. PRNG je efikasan, determinističan i periodičan, što ga čini pogodnim za aplikacije gde se zahteva mnogo brojeva i gde je korisno da se isti redosled može lako ponoviti kao što su simulacije i modelovanja. PRNG treba da poseduje neke karakteristike da bi se smatrao dobrim PRNG-om, kao što su uniformnost, nezavisnost, pravilna inicijalizacija, nepredvidivost, efikasnost i prenosivost. Kvalitet slučajnosti PRNG-a se meri izlaganjem skupu statističkih testova koji se nazivaju konzervativni testovi. Svaki kriptografski siguran PRNG treba da zadovolji dva osnovna uslova: test sledećeg bita i stanje kompromisa ekstenzija.

Karakteristike	PRNG	TRNG
Efikasnost	da	ne
Determinizam	da	ne
Periodičnost	da	ne

Tabela 3.2 – Karakteristike dva tipa generatora slučajnih brojeva



Slika 3.15 – Vizuelna uporedna analiza rezultata TRNG (levo) i PRNG (desno)

3.4.3 Entropija

Teorija informacija se bavi merenjem i prenosom informacija kroz komunikacioni kanal. Šenonova teorija informacija predstavlja osnovni rad u ovoj oblasti, koja pruža mnogo korisnih alata koji se zasnivaju na merenju informacije u smislu bita ili u smislu složenosti struktura potrebnih za kodiranje datih podataka.

U teoriji informacija pod pojmom entropije misli se na Šenonovu entropiju objavljenu u njegovom čuvenom radu iz 1948. godine „Matematička teorija komunikacija“. Ona predstavlja meru za količinu neodređenosti odnosno prosečnu količinu informacije koju sadrže generisane poruke nekog informacionog izvora.

Količina informacija u poruci M se meri entropijom poruke, označenom sa $H(M)$. Entropija poruke merena u bitovima je $\log_2 n$, u kojima je n broj mogućih značenja. Ovo podrazumeva da je svako značenje podjednako verovatno.

Za poruku P koja predstavlja binarni niz, entropija zavisi od verovatnoće 0 i 1. Entropija binarnih nizova se definiše na sledeći način:

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

Kod binarnih sekvenci, entropija H predstavlja informaciju po jednom bitu. Poželjna vrednost entropije je $H \approx 1$. Ova vrednost važi za verovatnoće pojavljivanja nula i jedinica $P_{1/0} \{ \sim 0.5, \sim 0.5 \}$.

Entropija se često koristi u vezi sa generatorima slučajnih brojeva. Redosled slučajnih promenljivih se naziva slučajna ako ima maksimalnu entropiju po bitu, što je 1:0.

3.5 NIST testovi

NIST-ov paket testova predstavlja statistički paket koji se sastoji od 15 testova koji su razvijeni za testiranje slučajnosti binarne sekvence koje proizvode hardver ili softver na bazi kriptografski slučajnih ili pseudoslučajnih brojeva. Ovi testovi se fokusiraju na različite vrste neslučajnosti koje bi mogle da postoje u nizu. Neki testovi su podeljeni u više podtestova.

3.5.1 Ispitivanje učestalosti u nizu

U fokusu ovog testa je ispitivanje odnosa jedinica i nula u nizu bitova. Cilj je uočavanje jednakosti pojavljivanja ove dve vrednosti. Potreban je približan broj jedinica i nula u sekvenci. Svi podtestovi koji proizilaze iz ovog testa direktno zavise od njegove uspešnosti.

Test se poziva preko metode **Frequency(n)** gde je n dužina bitnog uzorka. Metoda koristi i dodatni parametar ϵ koji predstavlja niz bitova generisanog preko RNG ili PRNG koji se testira.

Primer:- Ulazni parametri

$\varepsilon = 11001001000011111101101010100010001000010110100011$
 $00001000110100110001001100011001100010100010111000$
 $n = 100$

- Obrada

$S_{100} = -16$
 $S_{obs} = 1.6$

- Izlazni parametar

$P = 0.109599$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.2 Ispitivanje učestalosti u bloku

Test posmatra odnos jedinica i nula u n -bitnim blokovima. Cilj predstavlja uočavanje jednakosti broja jedinica i nula u svakom n -bitnom bloku.

Poziv testa se vrši preko metode **BlockFrequency(M, n)** gde su

M – dužina svakog bloka

n – dužina bitnog uzorka

Kao i kod testa za ispitivanje učestalosti u nizu, koristi se i dodati parametar ε .

Primer:- Ulazni parametri

$\varepsilon = 11001001000011111101101010100010001000010110100011$
 $00001000110100110001001100011001100010100010111000$
 $n = 100$
 $M = 10$

- Obrada

$N = 10$
 $\chi^2 = 7.2$

- Izlazni parametar

$$P = 0.706438$$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.3 Ispitivanje uzastopnih ponavljanja istih bitova u nizu

Posmatrana karakteristika u ovom testu je ukupan broj uzastopnih ponavljanja jednog broja u nizu. Svrha ovog testa je da se odredi da li broj uzastopnih ponavljanja nula i jedinica odgovara očekivanoj slučajnoj sekvenci.

Poziv testa se vrši preko metode **Runs(n)** gde je n dužina bitnog uzorka. Takođe postoji i dodatni parametar ε .

Primer:

- Ulazni parametri

$$\begin{aligned}\varepsilon &= 11001001000011111101101010100010001000010110100011 \\ &00001000110100110001001100011001100010100010111000 \\ n &= 100 \\ \tau &= 0.02\end{aligned}$$

- Obrada

$$\begin{aligned}\pi &= 0.42 \\ Vn(obs) &= 52\end{aligned}$$

- Izlazni parametar

$$P = 0.500798$$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.4 Ispitivanje najdužeg uzastopnog ponavljanja jedinica u bloku

U ovom testu se posmatra 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.

Poziv testa vrši se preko metode **LongestRunOfOnes(n)** gde je n dužina bitnog uzorka. Takođe postoji i dodatni parametar ε .

M – dužina svakog bloka. Test je podešen tako da za M koristi tri vrednosti: $M = 8$, $M = 128$ i $M = 104$ u skladu sa dužinom parametra n :

Minimum n	M
128	8
6272	128
750,000	104

Primer:

U slučaju kada je $K = 3$ i $M = 8$:

- Ulazni parametri

$\varepsilon = 11001100000101010110110001001100111000000000001001$
 $00110101010001000100111101011010000000110101111100$
 $1100111001101101100010110010$
 $n = 128$

- Obrada

Maksimalno ponavljanje podbloka	Maksimalno ponavljanje podbloka
11001100 (2)	00010101 (1)
01101100 (2)	01001100 (2)
11100000 (3)	00000010 (1)
01001101 (2)	01010001 (1)
00010011 (2)	11010110 (2)
10000000 (1)	11010111 (3)
11001100 (2)	11100110 (3)
11011000 (2)	10110010 (2)

$v_0 = 4$; $v_1 = 9$; $v_2 = 3$; $v_4 = 0$; $x_2 = 4.882457$

- Izlazni parametar

P-value = 0.180609

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.5 Ispitvanje stanja binarne matrice

Svrha ovog testa je provera linearne zavisnosti između podnizova fiksne dužine iz originalnog niza.

Poziv testa vrši se preko metode **Rank(n)** gde je n – dužina bitnog uzorka. Koristi se i dodati parametar ϵ .

Primer:

- Ulazni parametri

$\epsilon = 11001001000011111101101010100010001000010110100011$
 $00001000110100110001001100011001100010100010111000$
 $n = 100$

- Obrada

$N_1 = 46$
 $N_0 = 47.5$
 $d = -1.376494$

- Izlazni parametar

$P = 0.168669$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.6 Ispitivanje diskretne Furijerove transformacije

Fokus ovog testa predstavlja najviše vrednosti diskretne Furijerove transformacije niza. Svrha ovog testa je da otkrije periodične funkcije u testiranom nizu koja bi ukazivala na odstupanje od pretpostavke o slučajnosti. Namera je da se otkrije da li je broj najviših vrednosti koje prelaze 95% značajno različit od preostalih 5%.

Poziv testa vrši se preko metode **DiscreteFourierTransform(n)** gde je n – dužina bitnog uzorka. Koriste se i dodatni parametri:

M – broj redova u matrici

Q – broj kolona u matrici

Primer:- Ulazni parametri

ε = prvih 1000000 binarnih vrednosti proširenje ε

$n = 100000$

$M = Q = 32$

- Obrada

$N = 97$

$F_M = 23, F_{M-1} = 60, N - F_M - F_{M-1} = 14$

$\chi^2 = 1.2619656$

- Izlazni parametar

$P = 0.532069$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.7 Ispitivanje nepreklapajućih uzoraka

Karakteristika koja se posmatra u ovom testu je učestalost pojave svih mogućih n -bitnih uzoraka gde ne dolazi do preklapanja u celom ispitivanom nizu. Test bi trebalo da otkrije da li je broj pojava uzoraka koji se ne preklapaju približno jednak broju koji se očekuje za niz slučajnih brojeva.

Poziv testa vrši se preko metode **NonOverlappingTemplateMatching(m,n)** gde su

m – dužina bita u šablonu

n – dužina bitnog uzorka

Primer:- Ulazni parametri

$\varepsilon = 2^{20}$ bitova dobijenih preko G-SHA-1 generatora

$n = 2^{20}$

$B = 000000001$

- Obrada

$\mu = 255.984375$

$\sigma^2 = 247.499999$

$W1 = 259; W2 = 229; W3 = 271; W4 = 245; W5 = 272; W6 = 262;$

$W7 = 259; i W8 = 246$

$\chi^2(\text{obs}) = 5.999377$

- Izlazni parametar

$P = 0.110434$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.8 Ispitivanje preklapajućih uzoraka

Karakteristika koja se posmatra u ovom testu je učestalost pojave svih mogućih n -bitnih uzoraka gde dolazi do preklapanja u celom ispitivanom nizu. Test bi trebalo da otkrije da li je broj pojava uzoraka koji se preklapaju približno jednak broju koji se očekuje za niz slučajnih brojeva.

Poziv testa vrši se preko metode **OverlappingTemplateMatching(m,n)** gde su

m – dužina bita u šablonu

n – dužina bitnog uzorka

Primer:

- Ulazni parametri

ϵ = binarno proširenje ϵ do 1000000 bita

$n = 1000000$

$B = 111111111$

- Obrada

$v0 = 329; v1 = 164; v2 = 150; v3 = 111; v4 = 78; \text{ and } v5 = 136$

$\chi^2(\text{obs}) = 8.965859$

- Izlazni parametar

$P = 0.110434$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.9 Maurerov univerzalni statistički test

Ovim testom se ispituju 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.

Poziv testa vrši se preko metode **Universal(L, Q, n)** gde su

L – dužina svakog bloka

Q – broj blokova u inicijalnoj sekvenci

n – dužina bitnog uzorka

Takođe postoji i dodatni parametar ϵ .

Primer:

- Ulazni parametri

ϵ = binarno proširenje ϵ do 1000000 bita

$n = 1000000$

$B = 111111111$

- Obrada

$v_0 = 329$; $v_1 = 164$; $v_2 = 150$; $v_3 = 111$; $v_4 = 78$; and $v_5 = 136$

χ^2 (obs) = 8.965859

- Izlazni parametar

$P = 0.110434$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.10 Ispitivanje linearne kompleksnosti

Ovaj test se koristi za dobijanje dužine linearnog pomeračkog registra (eng. *Linear feedback shift register* - LFSR). Svrha ovog testa je utvrđivanje da li je sekvenca dovoljno kompleksa da se može smatrati slučajnom.

Poziv testa vrši se preko metode **LinearComplexity(M, n)** gde su

M – dužina svakog bloka

n – dužina bitnog uzorka

Koristi se i dodati parametar ϵ kao i K – broj stepeni slobode

Primer:

- Ulazni parametri

ϵ = prvih 1000000 binarnih vrednosti proširenja ϵ

$n = 1000000 = 10^6$

$M = 1000$

- Obrada

$v_0 = 11; v_1 = 31; v_2 = 116; v_3 = 501; v_4 = 258; v_5 = 57; v_6 = 26$

$\chi^2(\text{obs}) = 2.700348$

- Izlazni parametar

$P = 0.845406$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.11 Serijski test

Test se koristi za utvrđivanje učestalosti svih mogućih preklapanja n -bitnog niza u čitavoj sekvenci.

Poziv testa vrši se preko metode **Serial(m, n)** gde su

m – dužina svakog bloka

n – dužina bitnog uzorka

Koristi se i dodati parametar ϵ .

Primer:

- Ulazni parametri

ϵ = 1000000 bita binarnog proširenja ϵ

$m = 2$

$n = 1000000 = 10^6$

- Obrada

#0s = 499971

#1s = 500029

$$\#00s = 250116$$

$$\#01s = \#10s = 249855$$

$$\#11s = 250174$$

$$\psi^2_2 = 0.343128; \psi^2_1 = 0.003364; \psi^2_0 = 0.000000$$

- Izlazni parametar

$$\nabla\psi^2_2 = 0.339764$$

$$\nabla^2\psi^2_2 = 0.336400$$

$$P_1 = 0.843764$$

$$P_2 = 0.561915$$

- Zaključak

S obzirom da su P_1 i $P_2 \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.12 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.

Poziv testa vrši se preko metode **ApproximateEntropy(m, n)** gde su

m – dužina svakog bloka

n – dužina bitnog uzorka

Koristi se i dodati parametar ϵ .

Primer:

- Ulazni parametri

$$\epsilon = 110010010000111111011010101000100010000101101000110000100011010011000100110001100110001001100010100010111000$$

$$m = 2$$

$$n = 100$$

- Obrada

$$\text{ApEn}(m) = 0.665393$$

$$\chi^2(\text{obs}) = 5.550792$$

- Izlazni parametar

$P = 0.235301$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.13 Ispitivanje slučajnih zbirova

Svrha ovog testa je potreba da se utvrdi da li je kumulativni zbir parcijalnih sekvenci koje se javljaju u nizu koji se testira suviše veliki ili suviše mali u odnosu na očekivano ponašanje tog kumulativnog zbira slučajnih sekvenci.

Poziv testa vrši se preko metode **CumulativeSums(mode,n)** gde su

mode – mode = 0 ili mode = 1

n – dužina bitnog uzorka

Koristi se i dodati parametar ε .

Primer:

- Ulazni parametri

$\varepsilon = 11001001000011111101101010100010001000010110100011$
 $00001000110100110001001100011001100010100010111000$

$n = 100$

$mode = 0 \mid \mid mode = 1$

- Obrada

$z = 1.6 \mid \mid z = 1.9$

- Izlazni parametar

$P = 0.219194 \mid \mid P = 0.114866$

- Zaključak

S obzirom da je $P \geq 0.01$, smatra se da je sekvenca slučajna.

3.5.14 Ispitivanje slučajne digresije

Svrha ovog testa je da se utvrdi da li broj određenih stanja u okviru ciklusa odstupa od onoga što bi se očekivalo od slučajne sekvence. Ovaj test je zapravo niz od osam testova (i zaključaka), jedan test i zaključak za svako od stanja: -4, -3, -2, -1 i +1, +2, +3, +4.

Poziv testa vrši se preko metode **RandomExcursions(n)** gde je n – dužina bitnog uzorka. Koristi se i dodati parametar ε .

Primer:

- Ulazni parametri

$\varepsilon = 1000000$ bita binarnog proširenja ε

$n = 1000000 = 10^6$

- Obrada

$J = 1490$

Stanje = x	x^2	P	Zaključak
-4	3.835698	0.573306	slučajan
-3	7.318707	0.197996	slučajan
-2	7.861927	0.164011	slučajan
-1	15.692617	0.007779	nije slučajan
+1	2.485906	0.778616	slučajan
+2	5.429381	0.365752	slučajan
+3	2.404171	0.790853	slučajan
+4	2.393928	0.792378	slučajan

- Zaključak

Za x , u sedam stanja važi da je $P \geq 0.01$, što dovodi do zaključka da je sekvenca slučajna.

3.5.15 Ispitivanje slučajne promenljive digresije

Svrha testa je da se utvrdi da li ukupan broj stanja u kojima se niz nalazi, između $[-9, -1]$ i $[1, 9]$ premašuje očekivane vrednosti čisto slučajne sekvence. Ovaj test zapravo predstavlja grupu od 18 testova (i zaključaka), svaki test i zaključak za svako stanje: $-9, -8, \dots, -1$ i $+1, +2, \dots, +9$.

Poziv testa vrši se preko metode **RandomExcursionsVariant(n)** gde je n – dužina bitnog uzorka. Koristi se i dodati parametar ε .

Primer:

- Ulazni parametri

$\varepsilon = 1000000$ bita binarnog proširenja ε

$n = 1000000 = 10^6$

- Obrada

$J = 1490$

Stanje = x	x^2	P	Zaključak
-4	3.835698	0.573306	slučajan
-3	7.318707	0.197996	slučajan
-2	7.861927	0.164011	slučajan
-1	15.692617	0.007779	nije slučajan
+1	2.485906	0.778616	slučajan
+2	5.429381	0.365752	slučajan
+3	2.404171	0.790853	slučajan
+4	2.393928	0.792378	slučajan

- Zaključak

S obzirom da je $P \geq 0.01$ za svako od 18 stanja za x , smatra se da je sekvenca slučajna.

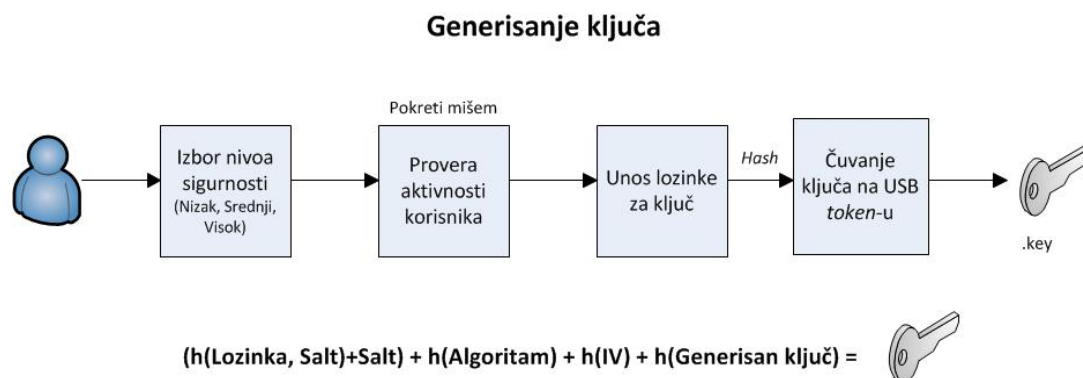
4 Pregled predloženog rešenja

U ovom poglavlju biće predstavljeno razvijeno rešenje. Prvo će biti objašnjeno preko generičkih šema, a zatim i njegova konkretna primena. Nakon toga, biće prikazane performanse i rezultati dobijeni preko nekoliko testova kojima je podvrgnuto.

4.1 Generička šema predloženog rešenja

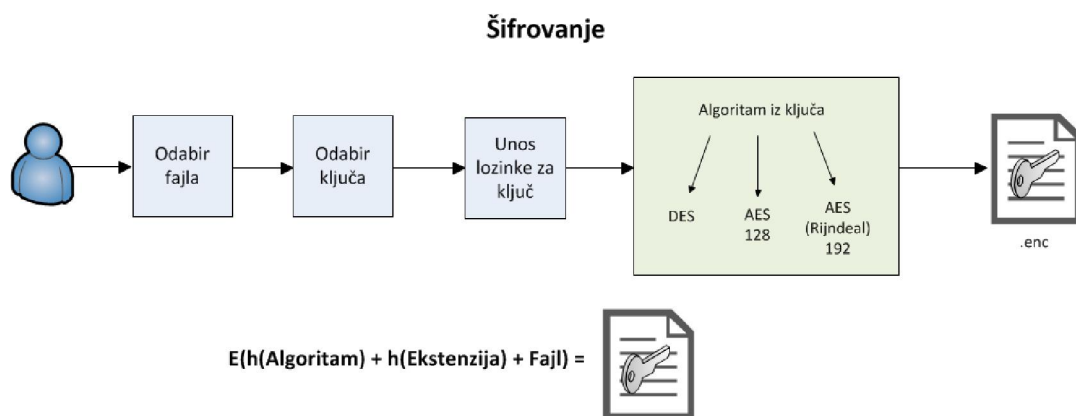
Razvijeno rešenje se sastoji iz tri faze: generisanje ključa, šifrovanje i dešifrovanje. Parametri za generisanje ključa zavise od nivoa sigurnosti odabranog od strane korisnika. Nakon procesa generisanja ključa, korisnik unosi lozinku kojom se on štiti. Materijal za ključ čine lozinka, algoritam za šifrovanje (izabran na osnovu nivoa sigurnosti) i parametri dobijeni preko miša. Jedan deo tih parametara se koristi za inicijalni vektor, a drugi za sam ključ.

Korisnikova lozinka se čuva kao „posoljena“, pa hešovana vrednost. Potom se uzima heš vrednost algoritma, inicijalnog vektora i ključa i zatim se vrši konkatencija te četiri vrednosti. Nakon toga se kreira fajl sa ekstenzijom **.key** (slika 4.1) i čuva na izabranoj lokaciji od strane korisnika (na USB *tokenu*).



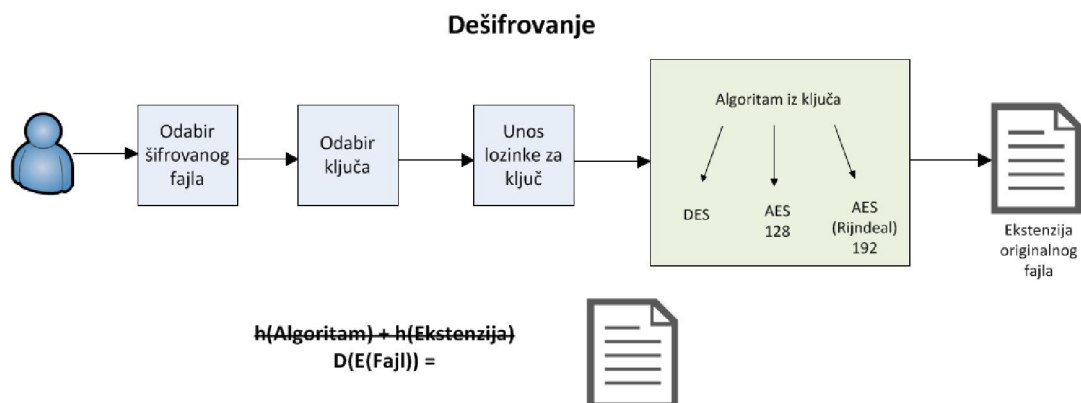
Slika 4.1 – Šema aplikacije - Generisanje ključa

U procesu šifrovanja nakon uspešne validacije unete lozinke od strane korisnika, iz ključa se uzima algoritam kojim se šifrjuje fajl. Šifrovani fajl sadrži heš vrednosti algoritma i njegove originalne ekstenzije i naravno, sam fajl. Nakon šifrovanja, originalni fajl se briše, a šifrovani dobija ekstenziju **.enc** (slika 4.2). Iz tog razloga se čuva originalna ekstenzija.



Slika 4.2 – Šema aplikacije - Šifrovanje

Prilikom dešifrovanja, kao i prilikom šifrovanja, od korisnika se zahteva da, nakon što izabere fajl koji želi da dešifruje i odgovarajući ključ, unese lozinku. Ukoliko je ispravna, proverava se da li algoritam iz ključa odgovara onom iz šifrovanog fajla. Potom se iz fajla uklanjaju podaci o algoritmu i ekstenziji i on se dešifruje. Podatak o ekstenziji se koristi za vraćanje originalne ekstenzije fajla (slika 4.3).



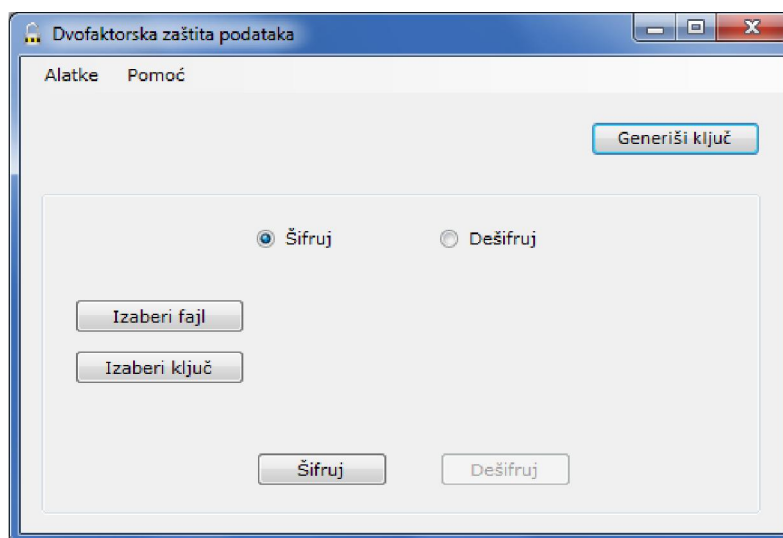
Slika 4.3 – Šema aplikacije - Dešifrovanje

4.2 Postavka i objašnjenje eksperimentalnog okruženja

Aplikacija je razvijena na Microsoft .NET platformi (.NET Framework 4.0), u programskom jeziku C#. Za testiranje korišćena je aplikacija namenjena za te svrhe, a kao okruženje za grafički prikaz dobijenih rezultata korišćen je alat Matlab R2012b.

Svrha razvijenog rešenja je šifrovanje svih vrsta fajlova simetričnim šifarskim algoritmima korišćenjem ključa koji je generisan na slučajan način pokretima miša. Dobijeni ključ predstavlja vrstu TRNG i podvrgnut je teorijsko-informacionoj analizi.

Aplikacija je osmišljena tako da korisniku omogućava lak i intuitivan rad. Na glavnom panelu, korisnik ima mogućnost da odabere da li želi da generiše ključ ili da šifruje ili dešifruje željeni fajl (slika 4.4).



Slika 4.4 – Početni panel razvijenog rešenja

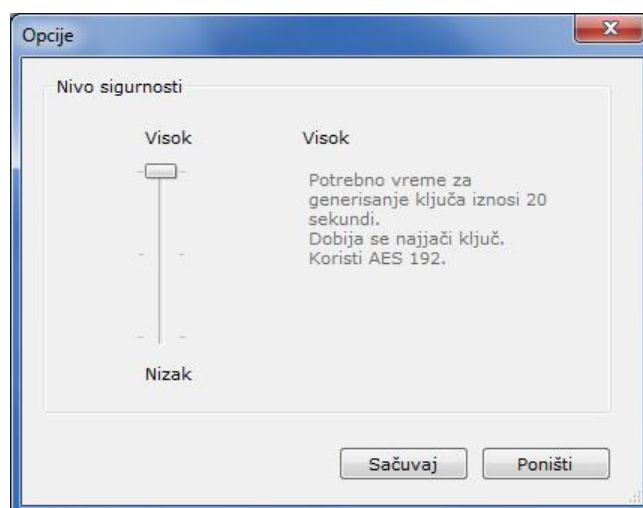
Nakon instalacije i inicijalnog pokretanja aplikacije, prvenstveno je neophodno generisanje barem jednog ključa kojim se može vršiti šifrovanje fajlova. Pre samog generisanja istog, potrebno je iz menija *Alatke* odabrati opciju *Opcije* koja omogućava odabir nivoa sigurnosti koji će se koristiti kako za generisanje ključeva (vreme potrebno za njegovo generisanje) tako i za samo šifrovanje fajlova.

Definisana su tri nivoa sigurnosti:

- Nizak
- Srednji
- Visok

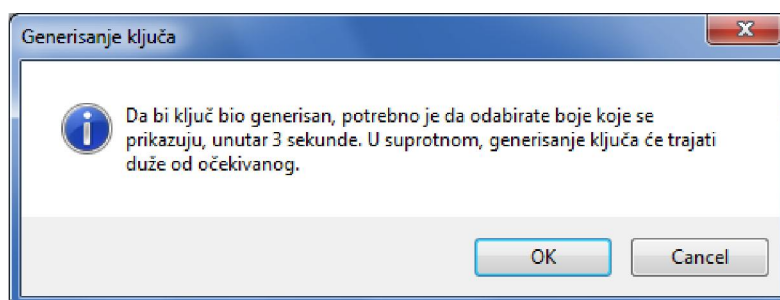
Odabirom niskog nivoa koristiće se DES šifarski algoritam. Srednji nivo podrazumeva AES 128 (dužina bloka 128 bita, dužina ključa 128 bita). Na visokom nivou sigurnosti koristiće se AES 192 (*Rijndael*) (dužina bloka 192 bita, dužina ključa 192 bita).

Prilikom odabira određenog nivoa, korisnik dobija i kraće objašnjenje šta koji od nivoa znači kako bi mu bio olakšan odabir (slika 4.5). Heš funkcija SHA-256 se koristi za sve nivoe sigurnosti.



Slika 4.5 – Odabir nivoa sigurnosti

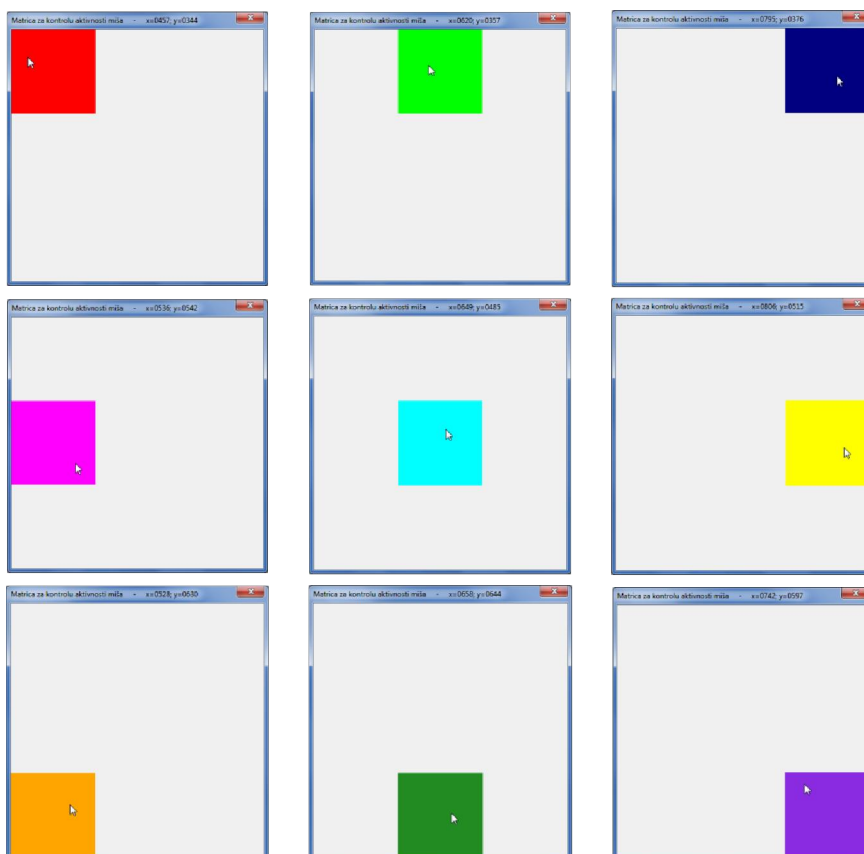
Sledeći korak za korisnika predstavlja generisanje ključa. Pre samog početka, obaveštava se o akcijama koje je potrebno da preduzme tokom trajanja generisanja (slika 4.6).



Slika 4.6 – Obaveštenje pre početka generisanja ključa

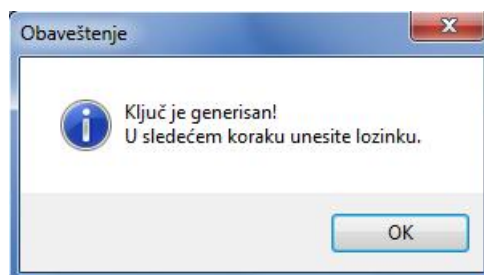
Zatim počinje proces generisanja. Od korisnika se zahteva da klikne mišem na kvadrat koji je označen bojom. Nakon svakog klika, prikazuje se sledeći slučajno izabran kvadrat. Tokom ovog perioda, skupljaju se koordinate miša prilikom klika (slika 4.7). Ukoliko korisnik nije bio aktivan barem tri sekunde, ukupno vreme potrebno za generisanje se za toliko uvećava.

Tokom generisanja, materijal za ključ se čuva u memoriji.



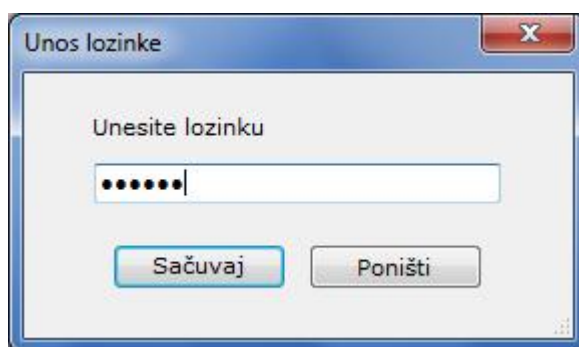
Slika 4.7 – Proces generisanja ključa

Nakon završetka, korisnik se obaveštava da je ključ generisan (slika 4.8).



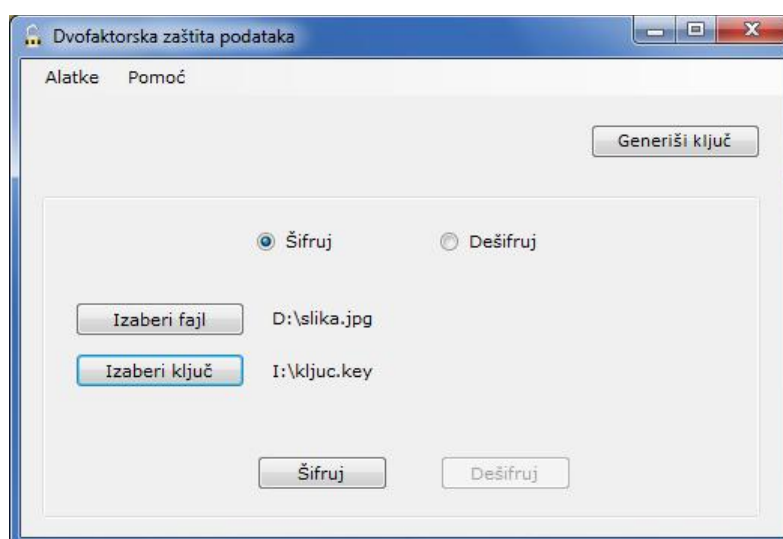
Slika 4.8 – Generisanje ključa uspešno završeno

U narednom koraku se od njega zahteva da unese lozinku kojom će se štititi taj ključ (slika 4.9). Minimalna dužina mora biti 6 karaktera. Na ovaj način u procesu šifrovanja ili dešifrovanja, korisnik će morati da koristi dvofaktorsku autentifikaciju koristeći nešto što ima (ključ) i nešto što zna (lozinka). Sledeći korak je čuvanje ključa na odabranom harverskom *tokenu* (USB *token*). Čuvanje na čvrstom disku nije moguće.

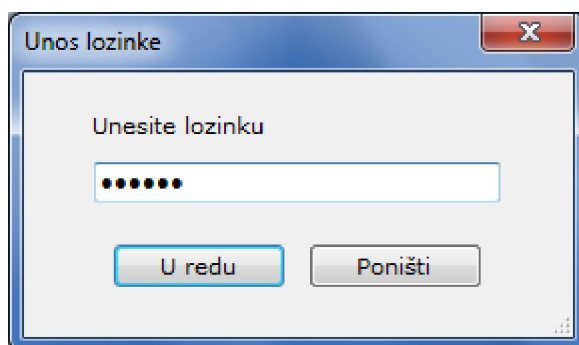


Slika 4.9 – Zaštita ključa lozinkom

Korisnik može generisati neograničeni broj ključeva za sva tri nivoa sigurnosti. Kada poseduje ključeve, može ih koristiti za šifrovanje ili dešifrovanje odabirom željenog fajla i ključa. Fajl koji se šifrjuje može se nalaziti na bilo kojoj lokaciji osim one na kojoj se nalazi i ključ (slika 4.10). Pre samog šifrovanja vrši se verifikacija ključa putem lozinke koja je unesena prilikom njegovog generisanja (slika 4.11).



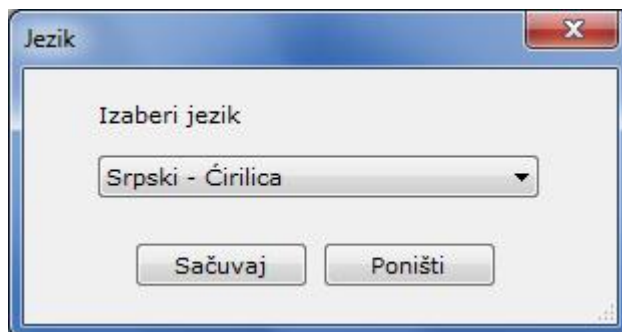
Slika 4.10 – Proces šifrovanja željenog fajla upotrebom generisanog ključa



Slika 4.11 – Unos lozinke za verifikaciju ključa

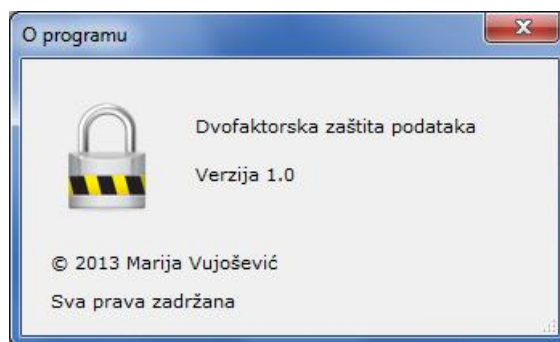
Nakon šifrovanja, originalni fajl se briše, a novi šifrovani fajl dobija ekstenziju **.enc** i čuva se na istoj lokaciji na kojoj je bio i originalni fajl.

Sa osnovnog panela aplikacije, iz menija *Alati – Jezik*, korisnik ima mogućnost izbora jezika aplikacije. Lokalizovana je na srpski (latinica i ćirilica) i engleski (slika 4.12).



Slika 4.12 – Odabir jezika aplikacije

Iz menija *O programu*, prikazuju se informacije o aplikaciji (njenoj verziji i autoru) (slika 4.13).



Slika 4.13 – O programu

❖ Specifikacija hardverske platforme

Ceo proces razvoja rešenja, kao i njegovo testiranje u eksperimentalnom okruženju sprovedeno je na hardverskoj platformi sa sledećim karakteristikama:

Komponente	Opis
Procesor	Intel i5 – 3570 3.4GHz
Memorija	8GB RAM
Hard disk	1TB
Rezolucija ekrana	1280x1024

Tabela 4.1 – Hardverske karakteristike razvojnog i eksperimentalnog okruženja

4.3 Performanse predloženog okruženja sa prikazom eksperimentalnih rezultata

Jaka informaciona analiza generisanog ključa je od velike važnosti iz razloga što postavlja teorijske okvire za utvrđivanje jačine dobijenog kriptološkog ključa. Korišćenjem Šenonove entropije dolazi se do prosečne količine informacija koje su sadržane u dobijenom ključu.

Nad dobijenim ključevima sprovedeni su testovi za procenu informacionog sadržaja (objašnjeni u poglavlju 3.5). Od opisanih testova, za ovo rešenje najvažniji su serijski test i ispitivanje entropije preklapajućih i nepreklapajućih uzoraka. Dobijeni rezultati predstavljeni su u uporednom prikazu sa uzorkom preuzetim sa www.random.org. Rezultati testova prikazani su u tabelama 4.2, 4.3 i 4.4.

Serijski test		
Bigrami		
	random.org	generisan ključ
00	7773	7471
01	7828	8299
10	7827	8300
11	7821	9294
Trigrami		
	random.org	generisan ključ
000	3818	3537
001	3955	3935
010	3863	4172
011	3965	4127
100	3955	3936
101	3873	4364
110	3965	4127
111	3856	5167

Tabela 4.2 – Rezultati serijskog testa

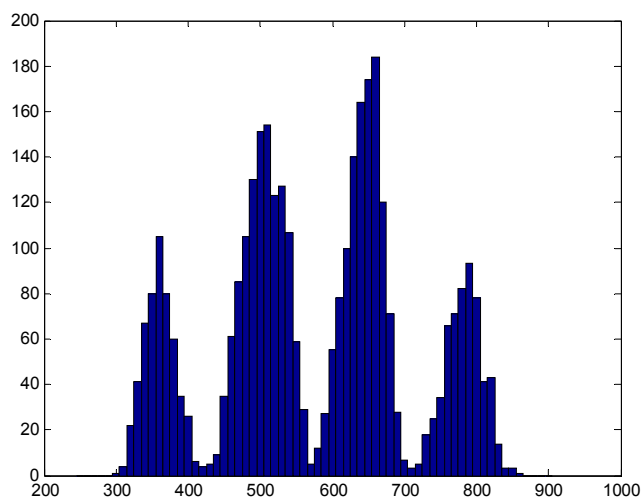
Entropija sa preklapanjem		
	random.org	generisan ključ
Monobit	0.9999982981270112	0.9978454731456681
Bigram	0.9999969195844207	0.9978439065269784
Trigram	0.9999512260990479	0.9974066948986483
Matrica 4x4	0.9999630196341835	0.997744280937179

Tabela 4.3 – Rezultati ispitivanja entropije sa preklapanjem

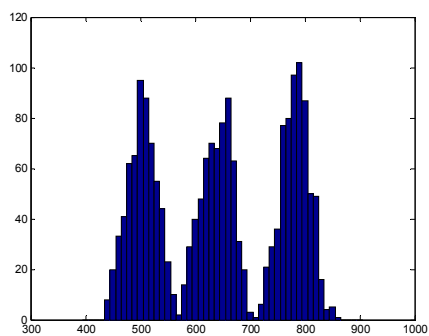
Entropija bez preklapanja		
	random.org	generisan ključ
Monobit	0.9999982981270112	0.9978454731456681
Bigram	0.9999443575006272	0.9978439065269784
Trigram	0.9999343024266536	0.9974066948986483
Matrica 4x4	0.9999630196341835	0.997744280937179

Tabela 4.4 – Rezultati ispitivanja entropije bez preklapanja

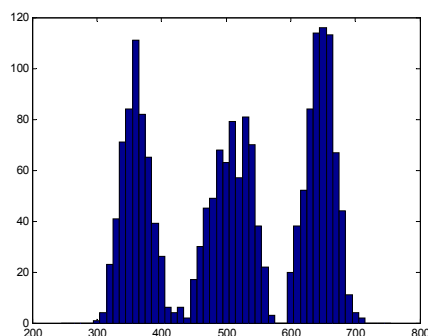
Na slikama 4.14, 4.15 i 4.16 predstavljeni su dobijeni rezultati preko histograma u Matlab-u.



Slika 4.14 – Prikaz dobijenog ključa preko histograma

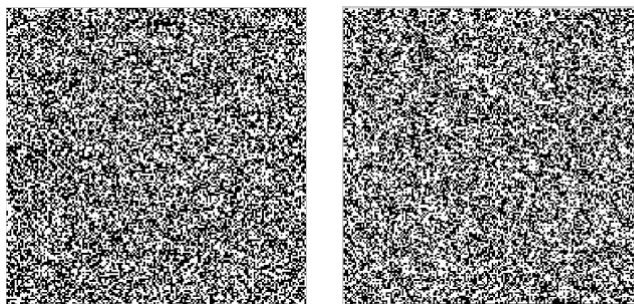


Slika 4.15 – Prikaz vrednosti dobijenih preko X ose



Slika 4.16 – Prikaz vrednosti dobijenih preko Y ose

Generisani ključ predstavljen je preko bitmape i upoređen je sa uzorkom sa random.org gde se jasno vidi da je dobijen TRNG (slika 4.17).



Slika 4.17 – Vizuelna uporedna analiza rezultata sa random.org (levo) i generisanog ključa (desno)

4.4 Oblast primene predloženog rešenja

Razvijeno rešenje podjednako se može koristiti kako u privatne tako i u poslovne svrhe. Zauzima vrlo malo prostora na disku. Namenjeno je svima kojima je potreban visok stepen zaštite podataka, a istovremeno omogućava lako i intuitivno korišćenje.

Poslovni korisnici aplikaciju mogu koristiti za npr. zaštitu strogo poverljive dokumentacije i drugih fajlova koji su od vitalne važnosti za kompaniju. Sa druge strane, privatni korisnici takođe imaju potrebu za zaštitom određenih podataka, naročito ako se oni nalaze na računarima koje dele sa drugim korisnicima (članovima porodice, prijateljima...).

5 Zaključak

5.1 Sumarni ciljevi rada i ostvareni rezultati

U ovom radu razvijeno je rešenje za kriptografsku zaštitu svih vrsta fajlova uz implementirani modul za generisanje kriptološkog ključa preko pokreta miša koji su uzeti kao materijal za ključ zbog potrebe da se omogući što veća slučajnost, a samim tim i entropija. Cilj je bio da se postigne pravi generator slučajnih brojeva (TRNG). Takođe je omogućeno generisanje neograničenog broja ključeva uz odabir određenog nivoa sigurnosti. Dobijeni ključevi se skladište isključivo na hardverskom *tokenu* (USB *token*). Dosta pažnje je posvećeno i ergonomiji aplikacije kako bi korisnici sa lakoćom mogli da je koriste.

Generisani ključevi su podvrgnuti teorijsko-informacionoj analizi u eksperimentalnom okruženju kojom je potvrđeno da je postignuta željena slučajnost. Kao reprezentativni uzorak sa kojim su upoređivane dobijene vrednosti korišćene su vrednosti preuzete sa sajta random.org.

Da bi se sprečila zloupotreba ključeva od strane drugih korisnika ili u slučaju krađe istih, uvedena je dvofaktorska autentifikacija prilikom šifrovanja odnosno dešifrovanja fajlova. Generisani ključevi se dodatno štite lozinkom.

5.2 Predlog za budući rad

Razvijena aplikacija predstavlja verziju 1.0. U planu je dalji razvoj u cilju dodatnog poboljšanja ergonomije. Takođe, biće razmatrano rešenje problema razmene i upravljanja ključevima koje nije obuhvaćeno ovim radom.

6 Literatura

- [1] M. Stamp, Information security: principles and practice, 2nd ur., New Jersey: John Wiley & Sons, 2011.
- [2] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd ur., New Jersey: John Wiley & Sons, 1996.
- [3] C. Paar / J. Pelzl, Understanding cryptography, Verlag, Berlin, Heidelberg: Springer, 2010.
- [4] M. Veinović / S. Adamović, Kriptologija 1, Beograd: Univerzitet Singidunum, 2013.
- [5] „Specification for the AES,“ NIST, 2001. <http://www.nist.gov/CryptoToolkit>
- [6] A. Jagannatham, *Mersenne Twister – A Pseudo Random Number Generator and its Variants*.
- [7] G. Srivastava, *Pseudorandom number generator using multiple sources of entropy*, University of Victoria, 2006.
- [8] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray / S. Vo, „A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications,“ NIST, Gaithersburg, MD, 2010.
- [9] ElcomSoft, „Advantages and disadvantages of EFS and effective recovery of encrypted data,“ ElcomSoft Co. Ltd., Moscow, 2007.
- [10] „AES Crypt,“ <http://www.aescrypt.com>
- [11] „AxCrypt,“ <http://www.axantum.com/axcrypt>
- [12] „CryptoForge,“ <http://www.cryptoforge.com>
- [13] „Folder lock,“ <http://www.newsoftwares.net/folderlock>
- [14] „TrueCrypt,“ <http://www.truecrypt.org>
- [15] „Random.org,“ <http://www.random.org>