UDC: 004.056.55 INFO M: str. 21-28

SOFTVERSKI SISTEM ZA VIZUELNU REPREZENTACIJU KLASIČNIH KRIPTOGRAFSKIH ALGORITAMA
SOFTWARE SYSTEM FOR VISUAL REPRESENTATION OF CLASSICAL CRYPTOGRAPHY ALGORITHMS

Žarko Stanisavljević, Jelena Stanisavljević

REZIME: Kriptografija više nije nauka rezervisana samo za matematičare. Bezbednost podataka, pa samim tim i kriptografija, sa razvojem i popularizacijom interneta postali su važni svima koji koriste moderne tehnologije. Iz tog razloga skup oblasti koje u nekoj meri izučavaju kriptografske algoritme se povećava. Softverski sistem koji je opisan u ovom radu ima za cilj da pomogne svima koji se upoznaju sa oblašću kriptografskih algoritama da to urade na što jednostavniji način. Sistem omogućava vizuelnu reprezentaciju klasičnih kriptografskih algoritama: Cezar, monoalfabetski, Playfair, Vigenere, Rail Fence, Row Transposition i Rotor Machine algoritama. Sistem omogućava da se brzo generišu raznovrsni primeri za ove algoritme na osnovu kojih se može lako upoznati sa načinom rada algoritama. Izbor algoritama je napravljen tako da svaki algoritam pruži neku novu informaciju korisniku, kako bi korisnik po završetku korišćenja sistema imao celovitu sliku o načinu funkcionisanja klasičnih kriptografskih algoritama.
KLJUČNE REČI: kriptografija, vizuelizacija algoritama, klasični kriptografski algoritmi, edukacija

ABSTRACT: Cryptography is no longer science reserved only for mathematicians. With the development of the Internet and its popularization and integration into all aspects of everyday life, data security and therefore cryptography become important for everyone who uses modern technologies. For this reason, a set of areas which study cryptographic algorithms to a certain extent increases. Software system that is described in this paper aims to help those who are trying to familiarize themself with the cryptographic algorithms to do it as simply as possible. The system provides a visual representation of classical cryptographic algorithms: Caesar, monoalphabetic, Playfair, Vigenere, Rail Fence, Row Transposition and Rotor Machine algorithms. The system makes it possible to quickly generate a variety of examples for these algorithms, based on which it can be easy to learn how algorithms work. Selection of algorithms is done so that every algorithm provides a new information to the user, so that the user at the end of the use of the system has a complete picture of how a classical cryptographic algorithms work.
KEY WORDS: cryptography, algorithms visualization, classical cryptographic algorithms, education

I. UVOD

Istorijski posmatrano kriptografija je uvek bila usko povezana sa vojnim primenama. Dugo je razvoj kriptografije bio skrivan od šire javnosti, koja nije imala ni potrebe za korišćenjem kriptografskih algoritama. U današnje vreme sigurnost i zaštita podataka imaju veću važnost nego ikada. Glavni razlog koji je doprineo tome je razvoj interneta i njegova popularizacija među širokom populacijom. To je dovelo do dizanja svesti ljudi o potrebi da svoj identitet i svoje podatke pravilno zaštite kako ne bi bili žrtve raznoraznih malverzacija. Danas ljudi praktično mogu da vode čitav svoj život virtuelno koristeći internet, pa se potreba za sigurnošću korišćenja interneta sa korporacija pomerila na pojedinca.

Tehnički gledano kriptografija je oblast koja pripada matematici, s obzirom da se sve sigurnosne aplikacije i sigurnosni protokoli na najnižem nivou oslanjaju na kriptografske algoritme, a kriptografski algoritmi su zasnovani na matematici koja stoji iza njih. Kako su primene kriptografije porasle, tako je i skup ljudi zainteresovanih za kriptografiju porastao. Danas postoji veliki broj oblasti različitih nauka koje u nekoj meri izučavaju kriptografske algoritme. Samim tim, javlja se veliki skup ljudi različitog predznanja i različitog pogleda na oblast sigurnosti i zaštite podataka, koji imaju potrebu da se u određenoj meri upoznaju sa kriptografskim algoritmima.

Kada govorimo o nastavi iz zaštite podataka kriptografski algoritmi spadaju u manje atraktivnu temu u odnosu na druge kojima se takvi predmeti obično bave [1]. Sa druge strane, kako se većina protokola i aplikacija na najnižem nivou upravo zasniva na kriptografskim algoritmima, važnost ove teme prevazilazi njenu popularnost kod studenata. Zbog toga se često smišljaju neki načini na koje bi se kriptografski algoritmi na zanimljiviji način približili studentima, kako bi oni na najjednostavniji način stekli znanje koje je potrebno da bi mogli da se bave složenijim temama.

Kao pomoć u izvođenju nastave odavno se koriste različiti softverski sistemi u zavisnosti od potreba oblasti u kojoj se upotrebljavaju. Ukoliko posmatramo oblasti gde se izučava neki hardver, tu se često zbog cene takvog hardvera prave softverski simulatori na kojima studenti mogu da se obučavaju [2], [3]. Ukoliko su u pitanju oblasti gde se izučavaju algoritmi, tada se koriste softverski sistemi za vizuelnu reprezentaciju algoritama, koji omogućavaju slikovit prikaz načina rada algoritma, brzo isprobavanje različitih primera rada algoritma i upoznavanje sa detaljima algoritma, koje bi bilo teško demonstrirati ručno [4], [5].

U ovom radu je opisan softverski sistem za vizuelnu reprezentaciju klasičnih kriptografskih algoritama. Svrha ovog sistema je da pomogne studentima raznih struka koji počinju da se bave oblašću zaštite podataka, kao i pojedincima zainteresovanim za ovu oblast, da na jednostavan način savladaju

osnovne koncepte klasičnih kriptografskih algoritama, kao i da saznaju detalje rada pojedinih algoritama. Sistem omogućava korisnicima da isprobavaju rad podržanih algoritama na različitim primerima koje sami smišljaju, da prate izvršavanje algoritma slovo po slovo, uz vizuelno naglašavanje putanje jednog slova od originalne do šifrovane poruke i da prolaskom kroz čitav sistem steknu celovitu sliku o klasičnim kriptografskim algoritmima.

Algoritmi koji su podržani u sistemu su birani tako da svaki novi algoritam doda neku novu informaciju, tako da kada se prođu svi podržani algoritmi korisnik ima kompletnu sliku. Algoritmi se mogu podeliti u tri grupe: substitucioni algoritmi (podrazumevaju da se svako slovo originalne poruke menja u šifrovanoj poruci), transpozicioni algoritmi (podrazumevaju da se izmeša redosled slova originalne poruke) i produkcioni algoritmi (kombinuju prva dva pristupa). Iz prve grupe podržani su: Cezar algoritam, monoalfabetski algoritam, *Playfair* algoritam i *Vigenere* algoritam. Iz druge grupe podržani su: *Rail Fence* algoritam i *Row Transposition* algoritam. Iz treće grupe podržan je *Rotor Machine* algoritam.

Ostatak rada organizovan je na sledeći način. U drugom poglavlju dat je opis kriptografskih algoritama podržanih u softverskom sistemu, kao i razlozi za izbor ovih algoritama, uz pregled postojećih sistema za vizuelnu reprezentaciju ovih algoritama. U trećem poglavlju dat je pregled mogućnosti realizovanog softverskog sistema uz prikaz rada sistema na primeru za svaki podržani kriptografski algoritam. U četvrtom poglavlju objašnjeni su najinteresantniji detalji implementacije realizovanog sistema, uz navođenje najbitnijih mogućnosti sistema u pomoći korisnicima za razumevanje podržanih algoritama. U petom poglavlju dat je zaključak rada.

II. ALGORITMI

U ovom poglavlju biće opisani algoritmi koji su odabrani za vizuelnu reprezentaciju u softverskom sistemu, biće dati razlozi za njihov odabir, kao i navedeni drugi sistemi koji ih takođe prikazuju. Detaljniji opis algoritama može se naći u [1].

A. Cezar algoritam

Cezar algoritam je prvi kriptografski algoritam za koji se zna. Smislio ga je Julije Cezar i sastojao se u tome da se svako slovo originalne poruke zameni slovom koje je za tri mesta udaljeno od njega u abecedi. Modifikacija ovog algoritma dozvoljava da pomeraj bude proizvoljan. Osim istorijskih razloga za implementaciju ovog algoritma, ovaj algoritam je odabran za vizuelnu reprezentaciju i zbog toga što je moguće na jednostavan način upoznati korisnika sa principom simetričnih kriptografskih algoritama (šifrovanja naspram dešifrovanja), ali takođe i zbog toga što je moguće prikazati princip kriptoanalize isprobavanjem svih mogućih vrednosti ključa (jer ih ima svega 25).

B. Monoalfabetski algoritam

Sledeći algoritam koji je odabran za vizuelnu reprezentaciju je monoalfabetski algoritam. Ovaj algoritam predstavlja dodatnu modifikaciju Cezar algoritma, tako što je sada korisniku dozvoljeno da sam odabere tabelu preslikavanja koja će se koristiti za zamenu slova originalne poruke. Na taj način je otklonjen najveći nedostatak Cezar algoritma (mali broj ključeva). Ovaj algoritam je zgodan i zbog toga što uvodi novu problematiku u razmatranje substitucionih algoritama, a to je koliko je frekvencija pojavljivanja slova u originalnom tekstu zastupljena u šifrovanom tekstu.

C. Playfair algoritam

Treći algoritam iz grupe substitucionih algoritama koji je odabran za vizuelnu reprezentaciju je *Playfair* algoritam. Problem koji je uveden kod monoalfabetskog algoritma, a koji se odnosi na statistiku originalne poruke u šifrovanoj poruci je rešavan na dva načina. Prvi način rešavanja je šifrovanje većeg broja slova istovremeno. *Playfair* algoritam je najbolji predstavnik grupe algoritama koja pokušava na ovaj način da reši opisani problem. Kod ovog algoritma šifrovanje nije na nivou slova, već na nivou slogova. Najpre se na osnovu ključne reči koju bira korisnik popunjava matrica preslikavanja koja je dimenzija 5x5. Ova matrica se popunjava red po red, tako što se prvo unose različita slova iz odabrane ključne reči, a zatim se ostala polja popunjavaju redom iz abecede onim slovima koja nedostaju. Rezultat je da imamo matricu sa 25 različitih slova (slova i i j se tretiraju kao jedno polje matrice). Nakon toga originalna poruka se deli na slogove, pri čemu se vodi računa da u jednom slogu ne smeju da se nađu dva ista slova. Ako do takve situacije dođe, tada se u poruku umeće neko slovo koje će razdvojiti ta dva ista slova (npr. X). Ukoliko poruka ima neparan broj slova poslednji slog se dopunjava nekim slovom koje je za to odabrano (može biti isto kao i za prethodnu situaciju). Sada se šifrovanje slogova vrši na osnovu matrice preslikavanja uz poštovanje sledećih pravila:

- ukoliko se slova iz sloga nalaze u istom redu u matrici preslikavanja, tada se svako slovo menja slovom koje se nalazi sa njegove desne strane iz istog reda u matrici preslikavanja,
- ukoliko se slova iz sloga nalaze u istoj koloni u matrici preslikavanja, tada se svako slovo menja slovom koje se nalazi ispod njega iz iste kolone u matrici preslikavanja,
- u suprotnom svako slovo se menja slovom koje se nalazi u istom redu kao to slovo, a u koloni u kojoj se nalazi njegov parnjak u matrici preslikavanja.

Ovakva vrsta algoritama napravila je pomak u odnosu na monoalfabetski algoritam, ali nije uspela da eliminiše opisani problem.

D. Vigenere algoritam

Poslednji algoritam iz grupe substitucionih algoritama koji je odabran za vizuelnu reprezentaciju je *Vigenere* algoritam. Problem koji je uveden kod monoalfabetskog algoritma, a koji se odnosi na statistiku originalne poruke u šifrovanoj poruci ovaj algoritam pokušava da reši na drugi način, a to je upotreba više različitih tabela preslikavanja za jednu poruku. Kod ovog algoritma imamo matricu sa 26 različitih tabela

preslikavanja gde svaka predstavlja po jedan modifikovani Cezar algoritam pri čemu je kod prve pomeraj 0, kod druge pomeraj 1, itd. Redovi i kolone ove matrice označeni su slovima abecede. Ključ za šifrovanje formira se na osnovu ključne reči koju bira korisnik. Potrebno je da ključ bude iste dužine kao i poruka, pa se u slučaju da je uneta ključna reč kraća od poruke za šifrovanje vrši dopunjavanje do odgovarajuće dužine. U originalnoj varijanti dopunjavanja ključa ključna reč se nadovezuje sama na sebe koliko god puta je potrebno kako bi ključ bio iste dužine kao i poruka. Naknadno je uvedeno autokey poboljšanje koje na ključnu reč nadovezuje deo originalne poruke, tako da dužina ključa bude jednaka dužini originalne poruke. Kada se formira ključ odgovarajuće dužine šifrovanje se vrši tako što se za svako slovo originalne poruke bira tabela preslikavanja koja će biti korišćena za šifrovanje na osnovu slova iz ključa. Drugim rečima slovo ključa određuje red u matrici, a slovo originalne poruke određuje kolonu u matrici iz koje se čita slovo šifrovane poruke. Ni ovaj algoritam ne uspeva da u potpunosti eliminiše problem definisan kod monoalfabetskog algoritma.

E. Rail Fence algoritam

Druga grupa algoritama koji su odabrani za vizuelnu reprezentaciju su transpozicioni algoritmi. Ovi algoritmi su zasnovani na drugačijem principu i umesto da menjaju sadržaj originalne poruke, pokušavaju da poruku zaštite permutacijom slova u originalnoj poruci. Ova grupa algoritama je važna, jer zajedno sa prethodnom formira osnovu za moderne kriptografske algoritme. Prvi predstavnik ove grupe algoritama koji je odabran za vizuelnu reprezentaciju je *Rail Fence* algoritam. Kod ovog algoritma originalna poruka se ispisuje dijagonalno po redovima i to cik-cak, najpre na dole, pa na gore i tako redom. Oblik koji se dobija ovakvim ispisivanjem poruke podseća na ogradu na pruzi, pa je tako algoritam i dobio ime. Šifrovana poruka se dobija tako što se poruka koja je ispisana na prethodno opisani način očitava i zapisuje red po red. Broj redova koji se koristi za ispis poruke predstavlja ključ.

F. Row Transposition algoritam

Drugi algoritam koji je odabran za vizuelnu reprezentaciju iz grupe transpozicionih algoritama je *Row Transposition* algoritam. Kod ovog algoritma poruka se ispisuje red po red u određenom broju kolona koji bira korisnik. Zatim se formira ključ kao proizvoljni redosled kolona. Na kraju se šifrovana poruka dobija tako što se poruka koja je zapisana na prethodno opisani način očitava i zapisuje kolonu po kolonu redosledom koji je definisan u ključu. Transpozicioni algoritmi imaju identičnu frekvenciju pojavljivanja slova u originalnoj poruci, kao i u šifrovanoj poruci, pa se zbog toga lako prepoznaju i nisu preterano teški za kriptoanalizu.

G. Rotor Machine algoritam

Razmatrajući prethodne dve grupe algoritama došlo se do zaključka da nijedna od njih ne može samostalno obezbediti dovoljnu pouzdanost. Zbog toga je osmišljena nova grupa algoritama, koja kombinuje prethodne dve i koja je nazvana produkcioni algoritmi. Prva primena produkcionih algoritama zabeležena je u tzv. rotor mašinama koje su korišćene za šifrovanje poruka u vojne svrhe u Drugom svetskom ratu. Rotor mašine su bile mehanički uređaji za šifrovanje poruka. Sastojale su se od određenog broja rotora od kojih je svaki imao 26 ulaznih i 26 izlaznih konektora, koji su interno bili međusobno fizički povezani tako da je svaki ulaz bio povezan sa samo jednim nasumično odabranim izlazom i obrnuto. Rotori su bili postavljani u mašinu tako da je tastatura na kojoj se unosila poruka bila povezana na ulazne konektore prvog rotora, izlazni konektori prvog rotora bili su povezani na ulazne konektore sledećeg, itd. da bi na kraju izlazni konektori poslednjeg rotora davali slovo šifrovane poruke. Šifrovanje se odvijalo tako što se pritiskom na slovo na tastaturi električni impuls prenosio kroz rotore da bi se na izlazu poslednjeg dobilo slovo šifrovanog teksta. Nakon svakog šifrovanog slova prvi rotor bi se zarotirao za jednu poziciju, svaki put kada prvi rotor okrene ceo krug, sledeći rotor se pomerao za jednu poziciju itd. U zavisnosti od broja rotora broj različitih mogućnosti za šifrovanje jednog slova se kretao od 17576 za 3 rotora do npr. 11881376 za 5 rotora. Ovakav algoritam predstavlja preteču savremenih simetričnih kriptografskih algoritama.

H. Pregled postojećih sistema

Na internetu je moguće pronaći različite softverske sisteme koji imaju mogućnost vizuelne reprezentacije nekog od opisanih algoritama. Svi ovi sistemi su ispravno realizovani i korisni određenim grupama korisnika za koje su dizajnirani. Pre realizacije novog softverskog sistema analizirani su postojeći, uočene su njihove prednosti i nedostaci i ova analiza je bila korisna prilikom dizajniranja novog sistema. Neki od najinteresantnijih sistema koji su dostupni biće opisani u nastavku.

Cryptool [6] je *open-source* program za kriptografiju i kriptoanalizu. Predviđen je za korišćenje u edukaciji, kako formalnoj u školama i na fakultetima, tako i neformalnoj u sklopu obuka zaposlenih u firmama. Fokus ovog programa nije na vizuelnoj reprezentaciji kriptografskih algoritama, već je u pitanju program mnogo većih dimenzija sa mogućnošću korišćenja velikog broja različitih kriptografskih algoritama za šifrovanje realnih podataka. Od nabrojanih klasičnih kriptografskih algoritama vizuelizacija je omogućena za Cezar, *Vigenere* i *Rotor Machine* algoritme. Vizuelizacija Cezar i *Vigenere* algoritama urađena je korišćenjem *Animal* alata [7] koji omogućava vizuelizaciju algoritama. Iz tog razloga vizuelizacija ovih algoritama je statička, odnosno predstavlja animaciju u kojoj se kroz predodređeni primer prikazuje način rada algoritma korak po korak uz mogućnost da korisnik kontroliše kretanje kroz animaciju. Za vizuelizaciju *Rotor Machine* algoritma prilagođen je *Enigma Machine* alat [8] koji predstavlja animaciju algoritma koji se koristio kod *Enigma* mašine. U ovoj animaciji rotori su prikazani kao dve koncentrične kružnice sa ispisanim slovima engleske abecede, pri čemu su interne i eksterne veze rotora označene linijama. Za svako uneto slovo originalne poruke bojama su označene linije veza koje su iskorišćene za šifrovanje tog slova. Zbog velikog broja linija preglednost u animaciji je nedovoljna, pa je samim tim i animacija zbunjujuća, posebno za početnike u ovoj oblasti.

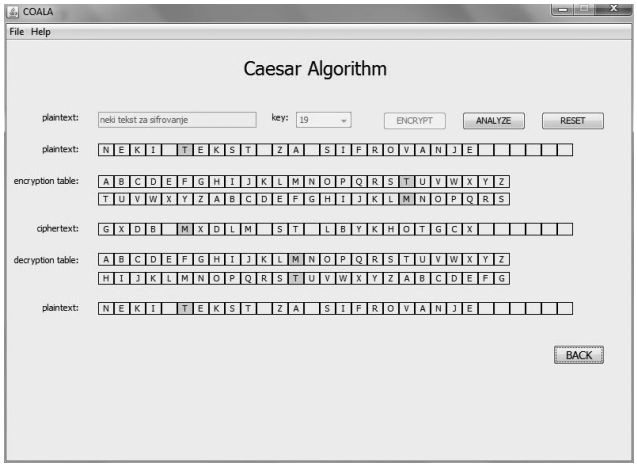
Crypto Club Cipher Tools [9] je interaktivni *online* alat za animaciju kriptografskih algoritama. Od nabrojanih klasičnih kriptografskih algoritama u ovom alatu podržana je vizuelizacija Cezar, monoalfabetskog i *Vigenere* algoritma. Vizuelizacija je dizajnirana tako da liči na igru i zahteva od korisnika da sam započne postupak šifrovanja, a zatim alat dovrši postupak. Nema posebnog označavanja svakog koraka u algoritmu, pa ovaj alat služi za uvežbavanje i proveru znanja koje je prethodno stečeno na neki drugi način. Za početnike koji bi želeli da korišćenjem alata nauče princip rada algoritama ovaj alat nije pogodan.

III. VIZUELNA REPREZENTACIJA

U ovom poglavlju biće prikazan način vizuelne reprezentacije svakog od podržanih kriptografskih algoritama. Svaki algoritam ima zadatak da prikaže nešto karakteristično za taj algoritam, a u sklopu celovite priče o klasičnim kriptografskim algoritmima.

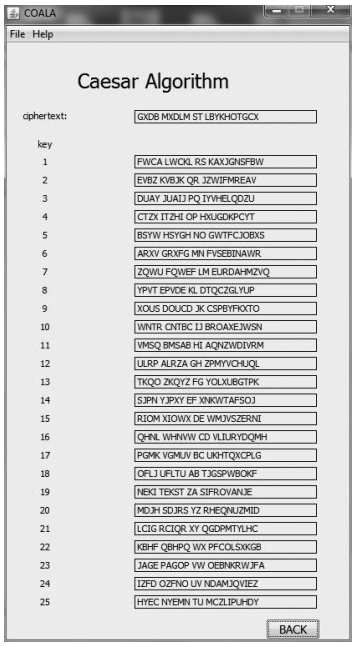
A. Cezar algoritam

Prvi podržani algoritam u sistemu je Cezar algoritam. Ovaj algoritam ima pre svega istorijsku važnost, jer demonstrira koliko dugo je potreba za zaštitom podataka postojala u ljudskoj istoriji. Na slici 1 prikazan je ekran sa vizuelnom reprezentacijom Cezar algoritma.



Slika 1. – *Vizuelna reprezentacija Cezar algoritma*

Korisnik ima mogućnost da unese tekst koji želi da šifruje i da izabere ključ koji želi da koristi. Pritiskom na dugme *ENCRYPT* dobija se prikaz postupka šifrovanja originalne poruke na osnovu tabele preslikavanja, kao i postupka dešifrovanja poruke na osnovu inverzne tabele preslikavanja. S obzirom da se radi o jednostavnom algoritmu ovde je pogodno prikazati i postupak dešifrovanja, koji se uvek svodi na inverzne operacije u odnosu na šifrovanje. Radi boljeg razumevanja, omogućeno je da korisnik može pozicioniranjem pokazivača miša na ekranu na neko od slova originalne ili šifrovane poruke da dobije dodatno bojom označen postupak transformacije tog slova, kao što je i prikazano na slici 1. Cezar algoritam je karakterističan i po tome što postoji svega 25 različitih ključeva koji se mogu primeniti za šifrovanje neke poruke. Zbog toga je ovo dobar trenutak da se demonstrira i kako se može uraditi kriptoanaliza ukoliko algoritam nije dovoljno bezbedan. Prikaz kriptoanalize dobija se pritiskom na dugme *ANALYZE*. Za primer sa slike 1 ekran sa prikazom kriptoanalize šifrovane poruke prikazan je na slici 2.

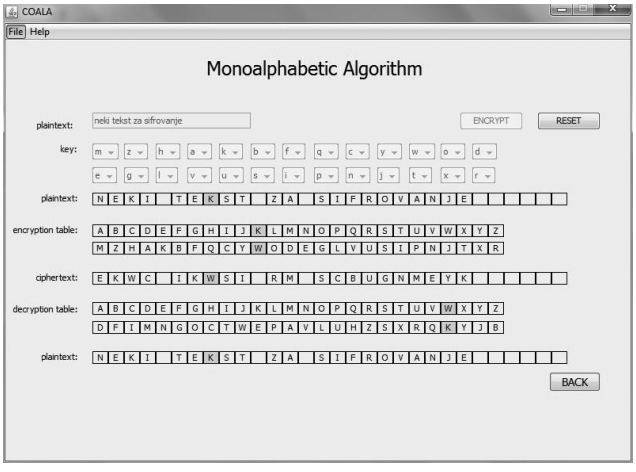


Slika 2. – *Ekran sa prikazom kriptoanalize Cezar algoritma*

Kod Cezar algoritma, s obzirom da je mali skup mogućih ključeva, kriptoanaliza se svodi na isprobavanje svih mogućih ključeva, što je i prikazano na slici 2. Kao što vidimo jedan od isprobanih ključeva daće originalnu poruku (u ovom slučaju to je ključ 19, koji je prethodno bio izabran u primeru sa slike 1) i samo je potrebno prepoznati koja od 25 mogućnosti bi mogla da bude originalna poruka.

B. Monoalfabetski algoritam

Naredni algoritam koji je podržan je monoalfabetski algoritam. Ovaj algoritam pokušava da prevaziđe nedostatak koji smo ustanovili kod Cezar algoritma, a to je mali broj mogućih ključeva. Na slici 3 prikazan je ekran sa vizuelnom reprezentacijom monoalfabetskog algoritma.



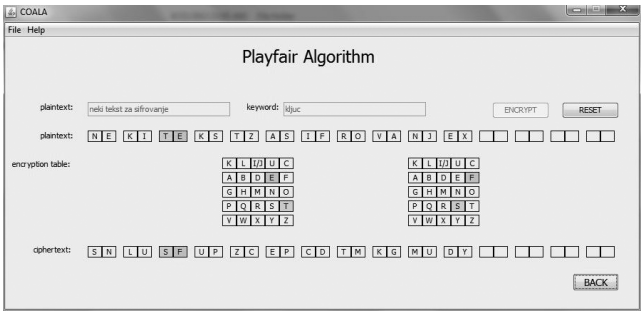
Slika 3. – *Vizuelna reprezentacija monoalfabetskog algoritma*

Kod monoalfabetskog algoritma korisnik može proizvoljno da izabere tabelu preslikavanja koju će koristiti za šifrovanje, što je u sistemu omogućeno odabirom 26 različitih slova ključa. Korisnik zatim unosi tekst za šifrovanje i pritiskom na dugme *ENCRYPT* dobija prikaz postupka šifrovanja, kao i postupka dešifrovanja, uz odgovarajuće tabele preslikavanja, slično kao i za Cezar algoritam. I ovde je omogućeno korisniku da pozicioniranjem pokazivača miša na ekranu na određeno slovo originalne ili šifrovane poruke dobije dodatno obeleženo kako je izvršena transformacija odabranog slova.

Proizvoljnim odabirom tabele preslikavanja korisnik sada ima na raspolaganju 26! različitih ključeva, čime je nedostatak Cezar algoritma svakako otklonjen. Postavlja se pitanje zbog čega ovaj algoritam nije zadržan u upotrebi. Odgovor je zbog toga što je analizom teksta na osnovu relativnih frekvencija slova u nekom jeziku moguće izvršiti kriptoanalizu šifrovane poruke, bez isprobavanja svih mogućih ključeva. Dakle, od ovog trenutka problem nije samo veličina ključa, već i koliko je statistika originalne poruke preneta na šifrovanu poruku. Kod monoalfabetskog algoritma ovo drugo je veoma izraženo, pa je zbog toga ovaj algoritam veoma nesiguran.

C. Playfair algoritam

Naredni algoritam koji je podržan je *Playfair* algoritam. Ovaj algoritam pokušava da prevaziđe nedostatak koji smo ustanovili kod monoalfabetskog algoritma, a to je veliko prenošenje statistitke iz originalne poruke u šifrovanoj poruci. Ovaj algoritam je predstavnik jednog od dva načina kojima je pokušano rešavanje opisanog problema. U ovom algoritmu pokušano je da se problem reši tako što će se šifrovanje vršiti na nivou slogova, umesto na nivou slova. Na slici 4 prikazan je ekran sa vizuelnom reprezentacijom *Playfair* algoritma.

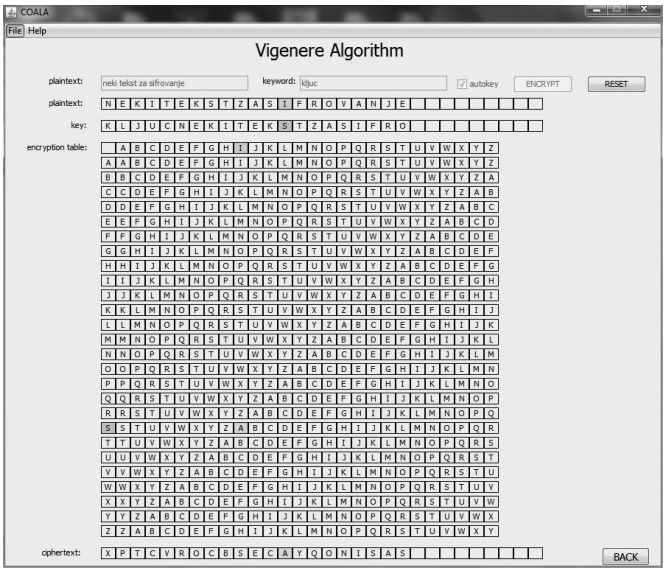


Slika 4. – *Vizuelna reprezentacija Playfair algoritma*

Kod *Playfair* algoritma originalna poruka se deli na slogove i zatim se slogovi prema definisanim pravilima šifruju pomoću matrice za šifrovanje koja je formirana na osnovu odabrane ključne reči, kao što je prikazano na slici 4. Korisniku je omogućeno da unese tekst za šifrovanje, kao i ključnu reč koja će se koristiti za šifrovanje. Zatim se pritiskom na dugme *ENCRYPT* dobija prikaz postupka šifrovanja. Matrica za šifrovanje, koja je popunjena na osnovu ključne reči, prikazana je dva puta kako bi moglo vizuelno da se vidi primenjeno pravilo za šifrovanje jednog sloga. Korisniku je omogućeno da pozicioniranjem pokazivača miša na ekranu na određeni slog originalne ili šifrovane poruke, može da dobije grafički prikaz primenjenog pravila za šifrovanje označenog sloga, pri čemu je svako slovo označeno različitom bojom radi bolje preglednosti. Na primer, na slici 4 vidimo primenu pravila za slog kod koga se slova ne nalaze ni u istom redu ni u istoj koloni u matrici za šifrovanje.

D. Vigenere algoritam

Naredni algoritam koji je podržan je *Vigenere* algoritam. Ovaj algoritam pokušava da prevaziđe nedostatak koji smo ustanovili kod monoalfabetskog algoritma, a to je veliko prenošenje statistitke iz originalne poruke u šifrovanoj poruci. Ovaj algoritam je predstavnik drugog od dva načina kojima je pokušano rešavanje opisanog problema. U ovom algoritmu pokušano je da se problem reši tako što će se za šifrovanje umesto jedne tabele preslikavanja koristiti više različitih tabela preslikavanja. Na slici 5 prikazan je ekran sa vizuelnom reprezentacijom *Vigenere* algoritma.

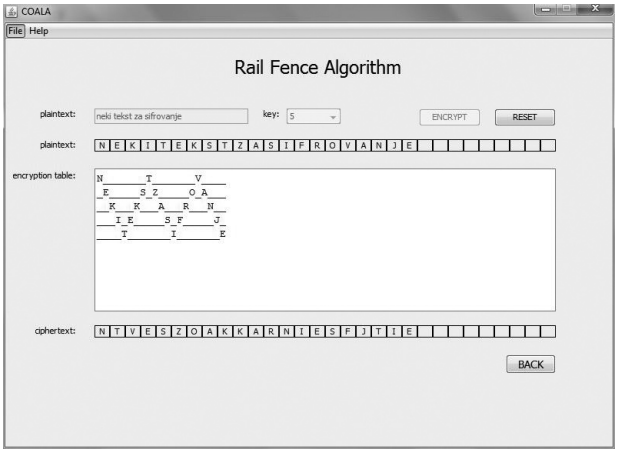


Slika 5. – *Vizuelna reprezentacija Vigenere algoritma*

Korisniku je omogućeno da unese tekst koji želi da šifruje, da unese ključ koji želi da koristi za šifrovanje i da odabere da li želi da koristi autokey poboljšanje ili ne, kao što je prikazano na slici 5. Pritiskom na dugme *ENCRYPT* dobija se prikaz postupka šifrovanja. Kod *Vigenere* algoritma za svako slovo originalne poruke koristi se druga tabela preslikavanja, pa je zbog toga prikazana matrica koja prikazuje svih 26 mogućih tabela preslikavanja. Slovo originalne poruke određuje kolonu u matrici, a slovo ključa određuje red u matrici i iz matrice se dobija slovo šifrovane poruke. Prethodno se ključ formira tako da bude iste dužine kao i poruka i to tako što se u originalnoj varijanti algoritma odabrana ključna reč nadovezuje potreban broj puta, odnosno u varijanti sa autokey poboljšanjem se na odabranu ključnu reč dodaje originalna poruka do potrebne dužine. Kako bi vizuelna preglednost bila bolja, korisniku je omogućeno da pozicioniranjem pokazivača miša na ekranu na određeno slovo originalne ili šifrovane poruke dobije bojom označenu transformaciju odabranog slova, kao što je prikazano na slici 5.

E. Rail Fence algoritam

Prvi predstavnik grupe transpozicionih algoritama koji je podržan je *Rail Fence* algoritam. Na slici 6 prikazan je ekran sa vizuelnom reprezentacijom *Rail Fence* algoritma.

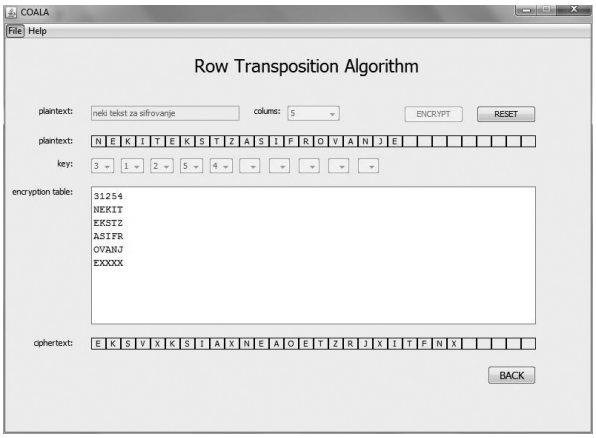


Slika 6. – *Vizuelna reprezentacija Rail Fence algoritma*

Korisnik ima mogućnost da unese tekst koji želi da šifruje i da odabere ključ koji želi da upotrebi za šifrovanje. Kod ovog algoritma ključ je zapravo broj redova u kojima će biti ispisana originalna poruka. Na slici 6 prikazan je primer rada ovog algoritma. Originalna poruka ispisana je dijagonalno u cik cak na dole, pa na gore i vidi se oblik po kome je algoritam i dobio naziv. Šifrovana poruka dobijena je iščitavanjem originalne poruke, koja je ispisana na prethodno opisani način, red po red.

F. Row Transposition algoritam

Drugi nešto složeniji predstavnik grupe transpozicionih algoritama koji je podržan je *Row Transposition* algoritam. Ova grupa algoritama ima identičnu frekvenciju pojavljivanja slova u originalnoj i šifrovanoj poruci i zasniva se na potpuno drugačijem konceptu u odnosu na substitucione algoritme. Na slici 7 prikazan je ekran sa vizuelnom reprezentacijom *Row Transposition* algoritma.

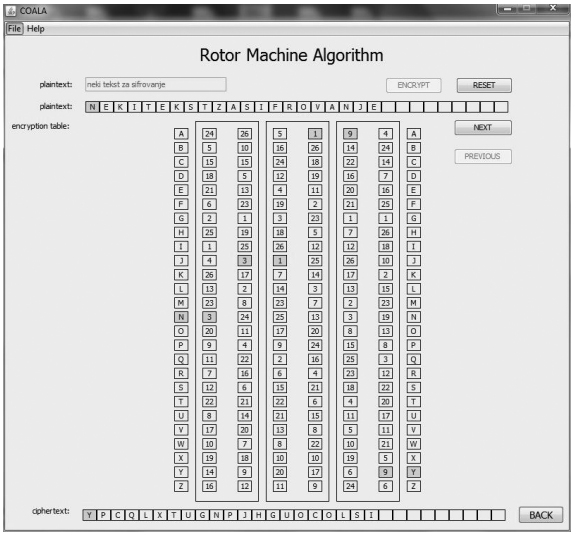


Slika 7. – *Vizuelna reprezentacija Row Transposition algoritma*

Korisnik ima mogućnost da unese tekst koji želi da šifruje, da odabere broj kolona koji želi da koristi u ključu i zatim da zada redosled kolona što praktično predstavlja ključ u ovom algoritmu, kao što je prikazano na slici 7. Kod ovog algoritma originalna poruka se ispisuje red po red u onoliko kolona koliko je korisnik odabrao. Šifrovana poruka dobija se tako što se poruka koja je ispisana na prethodno opisani način očita kolonu po kolonu i to onim redosledom koji je zadat ključem.

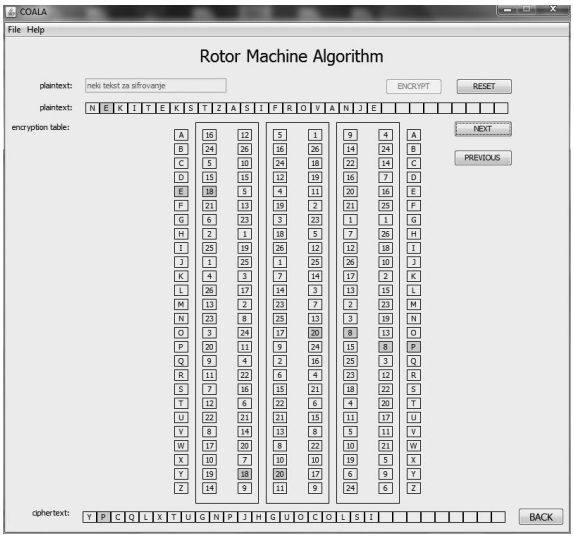
G. Rotor Machine algoritam

Jedini predstavnik grupe produkcionih algoritama koji je podržan je *Rotor Machine* algoritam. Kako ni substitucioni ni transpozicioni algoritmi samostalno nisu mogli da dostignu potreban nivo sigurnosti, osmišljeni su produkcioni algoritmi, koji kombinuju prethodna dva pristupa i čine veoma jake kriptografske algoritme. Prva njihova primena bila je u kriptografskim rotor mašinama u Drugom svetskom ratu. Ovi algoritmi predstavljaju preteču savremenih simetričnih kriptografskih algoritama. Na slici 8 prikazan je ekran sa vizuelnom reprezentacijom *Rotor Machine* algoritma.



Slika 8. – *Vizuelna reprezentacija Rotor Machine algoritma*

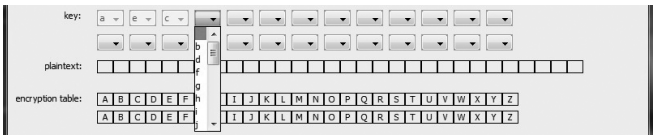
Korisnik ima mogućnost da unese tekst koji želi da šifruje i pritiskom na dugme *ENCRYPT* dobija prikaz postupka šifrovanja, kao što je prikazano na slici 8. Prikazana je rotor mašina sa tri rotora: brzo, srednje i sporo rotirajući. Brzo rotirajući rotor rotira za jedno mesto nakon šifrovanja svakog slova, srednje rotirajući rotor rotira za jedno mesto svaki put kada brzo rotirajući rotor napravi ceo ciklus i sporo rotirajući rotor rotira za jedno mesto svaki put kada srednje rotirajući rotor napravi ceo ciklus. Ovde se korisnik kroz postupak šifrovanja kreće koristeći dugme *NEXT* za prelazak na naredno slovo originalne poruke i pritiskom na dugme *PREVIOUS* za povratak na prethodno slovo originalne poruke. Ovaj put dodatno označavanje bojom je konstantno prikazano za slovo na kome je trenutno zaustavljen prikaz postupka šifrovanja. Vidi se na koji način slovo originalne poruke prolazi kroz svaki od tri rotora i šta se dobija kao slovo šifrovane poruke. U primeru sa slike 8, kada se pritisne dugme *NEXT* dobija se prikaz sa slike 9.



Slika 9. – *Vizuelna reprezentacija narednog slova kod Rotor Machine algoritma*

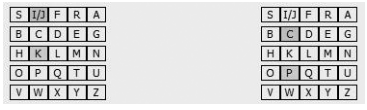
IV. IMPLEMENTACIJA

U ovom poglavlju objašnjeni su najinteresantniji detalji implementacije realizovanog sistema, uz navođenje najbitnijih mogućnosti sistema u pomoći korisnicima za razumevanje podržanih algoritama. Sistem je realizovan u programskom jeziku *Java* [10], uz korišćenje *Netbeans* [11] razvojnog okruženja.



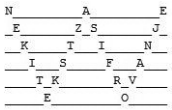
Slika 10. – *Izbor ključa kod monoalfabetskog algoritma*

Na slici 10 prikazan je način izbora ključa kod monoalfabetskog algoritma. Kod ovog algoritma korisnik na proizvoljan način bira tabelu preslikavanja i taj izbor predstavlja ključ. Izbor ključa u softverskom sistemu realizovan je uz pomoć 26 padajućih lista od kojih svaka na početku sadrži svih 26 slova engleske abecede. Kako je potrebno da ključ sadrži 26 različitih slova, svaki put kada korisnik odabere neko slovo u jednoj od padajućih lista, to slovo se briše iz svih ostalih lista. Na taj način je eliminisana mogućnost greške korisnika i korisniku se sugeriše kako treba da izgleda ključ u ovom algoritmu, čak i ako on nije prethodno bio upoznat sa tim.



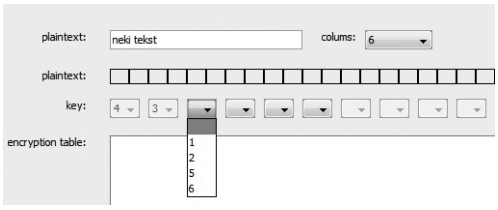
Slika 11. – *Prikaz zamene sloga kod Playfair algoritma*

Kod *Playfair* algoritma bilo je potrebno na jednoznačan način prikazati koje pravilo transformacije je primenjeno za određeni slog originalnog teksta. U softverskom sistemu je to urađeno na način prikazan na slici 11. Matrica koja se koristi za šifrovanje prikazana je dva puta. U levoj matrici označena su slova iz sloga originalne poruke i to svako slovo različitom bojom. U desnoj matrici označena su slova sloga koji menja slog originalne poruke u šifrovanoj poruci, ponovo različitim bojama za svako slovo, ali tako da boja odgovara boji slova iz leve matrice koje to slovo menja. Da je korišćena samo jedna matrica bilo bi konfuzno ustanoviti kako izgleda pravilo koje je primenjeno bez prethodnog predznanja. Ovakvom realizacijom je intuitivno prepoznati koja pravila se koriste za transformaciju kod *Playfair* algoritma.



Slika 12. – *Ispis šifre kod Rail Fence algoritma*

Kod *Rail Fence* algoritma, suština algoritma je u načinu ispisa originalnog teksta, kao što je prikazano na slici 12. U softverskom sistemu realizovano je da se originalni tekst ispisuje na odgovarajući način korišćenjem tekstualne komponente koja dozvoljava ispis u više redova. Ispis je realizovan tako što se formira niz *string*-ova onolike dužine koliko je korisnik odabrao da bude ključ. Zatim se pravi dvostruka petlja koja će presložiti uneti originalni tekst u ovaj niz *string*-ova tako da svaki *string* obuhvata jedan red teksta za ispis. Jasno je da će dužina svakog *string*-a u nizu biti jednaka dužini unetog originalnog teksta. U svakoj iteraciji petlje jedan od *string*-ova u nizu će dobiti slovo originalnog teksta, a svi ostali karakter koji je odabran za ispunjavanje praznina. Popunjavanje slova originalne poruke kreće od prvog reda, ide do poslednjeg, a kada dođe do poslednjeg vraća se unazad do prvog i tako dok se ne potroše sva slova originalnog teksta. Na kraju se formirani niz *string*-ova samo prepiše u tekstualnu komponentu red po red.



Slika 13. – *Izbor ključa kod Row Transposition algoritma*

Kod *Row Transposition* algoritma korisnik unosi ključ koji predstavlja jednu permutaciju redosleda kolona. Da bi se eliminisala mogućnost greške kod unosa ključa i da bi se korisniku sugerisalo kako treba da izgleda ključ koji bira, iskorišćeno je slično rešenje kao kod monoalfabetskog algoritma, kao što je prikazano na slici 13. Korisnik unosi broj kolona koji će koristiti za ispis originalne poruke. Na osnovu unetog broja kolona odgovarajući broj padajućih lista postaje aktivan i pri tom svaka od njih ima vrednosti od 1 do broja kolona koji je unet, kao mogućnost izbora. Sada korisnik treba da odabere redosled kolona i svaki put kada odabere neku vrednost u jednoj od padajućih lista, ta vrednost se briše u svim ostalim aktivnim padajućim listama, kako bi korisnik mogao da izabere samo različite vrednosti u ključu.

```
@Action
public void next() {
    if(!btnPrevious.isEnabled()) btnPrevious.setEnabled(true);
    if((highlighted+1)==count)
        if(btnNext.isEnabled()) btnNext.setEnabled(false);
    rotateRight(encryptionTableViewMatrix[1]);
    rotateRight(encryptionTableViewMatrix[2]);
    if(highlighted%26==0){
        rotateRight(encryptionTableViewMatrix[3]);
        rotateRight(encryptionTableViewMatrix[4]);
    }
    if(highlighted%676==0){
        rotateRight(encryptionTableViewMatrix[5]);
        rotateRight(encryptionTableViewMatrix[6]);
    }
    clearHighlight();
    setCiphertextView();
}
```

Slika 14. – *Programski kod metode next kod Rotor Machine algoritma*

Na slici 14 dat je programski kod metode *next* koja se poziva svaki put kada se pritisne dugme *NEXT* kod *Rotor Machine* algoritma (slika 8). Zadatak ove metode je da ispravno prikaže šifrovanje jednog slova u ovom algoritmu. Na početku se ažurira stanje dugmića *NEXT* i *PREVIOUS* u skladu sa trenutnom pozicijom slova originalne poruke za koju je prikazano šifrovanje. Zatim se vrši potrebno rotiranje svakog od tri rotora. Rotori su zajedno za ulaznom i izlaznom abecedom smešteni u jednu matricu koja ima 8 kolona i 26 redova. Nakon svakog šifrovanog slova prvi rotor se bezuslovno rotira za jedno mesto na dole (u kodu je označeno da je u pitanju rotiranje u desno). Ukoliko je prvi rotor završio kompletan ciklus rotacije (vratio se u početni položaj), tada drugi rotor rotira za jedno mesto na dole. Ukoliko je drugi rotor završio kompletan ciklus rotacije, tada treći rotor rotira za jedno mesto na dole. Nakon rotiranja pozivom metode *clearHighlight* brišu se označavanja bojom koja su prethodno bila postavljena za prikaz šifrovanja prethodnog slova originalne poruke. Na kraju se pozivom metode *setCiphertextView* bojom označava putanja slova originalne poruke kroz sva tri rotora do dobijenog slova šifrovane poruke.

V. ZAKLJUČAK

U ovom radu prikazan je softverski sistem za vizuelnu reprezentaciju klasičnih kriptografskih algoritama. Objašnjeni su algoritmi koji su odabrani za realizaciju, kao i razlozi za njihov izbor i dat je pregled drugih sistema koji ih prikazuju. Zatim je prikazan način rada sistema na primeru za svaki od podržanih kriptografskih algoritama. Na kraju su objašnjeni najinteresantniji detalji implementacije sistema sa posebnim osvrtom na metode korišćene za pomoć korisnicima u učenju podržanih algoritama.

Realizovani softverski sistem ima za cilj da omogući korisnicima koji se upoznaju sa oblašću zaštite podataka da na jednostavan i interesantan način savladaju osnovne koncepte klasičnih kriptografskih algoritama. U tu svrhu napravljen je interaktivan i intuitivan korisnički interfejs koji minimizuje mogućnost greške korisnika i maksimizuje efikasnost korisnika u učenju uz korišćenje ovog sistema.

VI. LITERATURA

[1] W. Stallings, “Cryptography and Network Security Principles and Practices,” 4th ed., Prentice Hall, 2005.

[2] B. Nikolic, Z. Radivojevic, J. Djordjevic, V. Milutinovic, “A Survey and Evaluation of Simulators Suitable for Teaching Courses in Computer Architecture and Organization,” IEEE Transactions on Education, vol. 52, no. 4, pp. 449-459, 2009.

[3] Z. Stanisavljevic, V. Pavlovic, B. Nikolic, J. Djordjevic, “SDLDS—System for Digital Logic Design and Simulation,” IEEE Transactions on Education, vol.56, no.2, pp.235-245, 2013.

[4] J. Urquiza-Fuentes, J. Á. Velázquez-Iturbide, “A survey of successful evaluations of program visualization and algorithm animation systems,” ACM Transactions on Computing Education (TOCE), vol. 9, no. 2, article no. 9, 2009.

[5] Z. Stanisavljevic, J. Stanisavljevic, “Softverski sistem za vizuelnu reprezentaciju Advanced Encryption Standard algoritma,” TELFOR, pp.1364-1367, 2011.

[6] Cryptool. Available: http://www.cryptool.org/en/.

[7] G. Rößling, M. Schüer, B. Freisleben, “The ANIMAL algorithm animation tool,” ACM SIGCSE Bulletin, vol. 32, no. 3, pp. 37-40, 2000.

[8] Enigma Machine Online. Available: http://enigmaco.de/enigma/enigma.swf.

[9] Crypto Club Cipher Tools. Available: http://cryptoclub.org/tools/ciphers.php.

[10] Java. Available: http://www.java.com/en/.

[11] Netbeans. Available: https://netbeans.org/.

master inž. Žarko Stanisavljević, asistent, Elektrotehnički fakultet Univerziteta u Beogradu, Kontakt: zarko.stanisavljevic@etf.bg.ac.rs Oblasti interesovanja: zaštita podataka i računarskih sistema, razvoj softverskih alata za elektronsko učenje, programiranje internet aplikacija, arhitektura i organizacija računara

master inž. Jelena Stanisavljević, stariji programer, Asseco SEE Srbija, Kontakt: jelena.stanisavljevic@asseco-see.rs Oblasti interesovanja: dizajniranje korisničkog interfejsa, programiranje, masovna obrada podataka, arhitektura i organizacija računara