



- MASTER STUDIJSKI PROGRAM -
SAVREMENE INFORMACIONE TEHNOLOGIJE

Kriptografija eliptičnih krivih

- Master rad -

Kandidat:

Kristina Kunjadić-Ćulibrk
Broj indeksa: 2015/410102

Mentor:

Prof. dr Mladen Veinović

Beograd, septembar 2016.



- MASTER STUDIJSKI PROGRAM -
SAVREMENE INFORMACIONE TEHNOLOGIJE

UNIVERZITET SINGIDUNUM

Departman za postdiplomske studije

Beograd, Danijelova 32

Kandidat: Kristina Kunjadić-Ćulibrk

Broj indeksa: 2015/410102

Smer: Savremene informacione tehnologije

Tema: Kriptografija eliptičnih krivih

MENTOR

Mesto i datum odobrenja rada:
Beograd, 15. septembar 2016.

Prof. dr Mladen Veinović

Kriptografija eliptičnih krivih

Sažetak: U ovom radu biće objašnjena primena eliptičnih krivih u oblasti kriptografije. Eliptične krive su glatke algebarske krive prvog reda. Algebarske krive su krive koje se mogu predstaviti algebarskim polinomom sa dve promenljive. Grafici algebarskih krivih su ravanske krive i mogu se predstaviti u dvodimenzionom dekartovom pravouglom koordinatnom sistemu. Glatke krive su krive koje su diferencijabilne u svakoj tački krive.

Kriptografija eliptičnih krivih predstavlja posebnu vrstu asimetričnih šifarskih sistema odnosno sistema sa javnim ključevima - PKI (Public Key Infrastructure). Pored toga, eliptične krive se mogu koristiti i u algoritmima za razmenu kriptografskih ključeva, kao i u algoritmima za digitalno potpisivanje.

U radu će biti opisani nekih od najpoznatijih napada na algoritme bazirane na eliptičnim krivama. Završni deo rada predstavlja rešenje u kome su implementirani prethodno pomenuti algoritmi za šifrovanje. Rešenje je pokaznog karaktera i nije namenjeno za profesionalnu upotrebu.

Primena eliptičnih krivih u kriptografiji, koja će biti opisana u radu, značajno podiže nivo bezbednosti šifrovanih podataka.

Ključne reči: kriptografija, eliptične krive, zaštita podataka, kriptografski ključevi, šifrovanje, kriptoanaliza, izvorni kod

Elliptic Curve Cryptography

Abstract: In this paper will be explained the use of elliptic curves in cryptography. Elliptic curves are smooth algebraic curve of the first order. Algebraic curves are curves that can be represented by algebraic polynomial with two variables. Graphics of algebraic curves are planar curves, and can be represented in two-dimensional rectangular Cartesian coordinate system. Smooth curves are curves that are differentiable at each point of the curve.

Elliptic curves cryptography is a special type of asymmetric encryption system or a system with public keys - PKI (Public Key Infrastructure). In addition, the elliptic curve can be used in the algorithms for exchanging cryptographic keys and algorithms for the digital signature.

The review of some of the most popular attacks on algorithms based on elliptic curves will be given. The final part of the paper represents the source code in which encryption algorithm, previously described, was implemented. The program is only made for the purposes of demonstration and is not intended for professional use.

The application of elliptic curves in cryptography, which will be described in the paper, significantly raises the level of security of encrypted data.

Keywords: cryptography, elliptic curves, data protection, cryptographic keys, encryption, cryptanalysis, source, code

Sadržaj

1. Uvod	1
2. Kriptografski sistemi	5
2.1 Simetrični kriptografski sistemi	6
2.1.1 Sekvencijalni šifarski sistemi	7
2.1.2 Blok šifarski sistemi	8
2.2 Asimetrični kriptografski sistemi	9
2.3 Digitalni potpis	12
3. Eliptične krive	13
3.1 Algebarske krive	13
3.2 Definisane eliptičnih krivih	14
3.2.1 Geometrija eliptičnih krivih	15
3.2.2 Sabiranje dve različite tačke eliptične krive	15
3.2.3 Sabiranje tačke eliptične krive sa samom sobom	16
3.2.4 Određivanje suprotnog elementa	17
3.2.5 Algebra eliptičnih krivih	18
4. Primena eliptičnih krivih u kriptografiji	21
4.1 Generisanje para kriptografskih ključeva	21
4.1.1 Validacija javnog ključa	23
4.2 Šema digitalnog potpisa	23
4.2.1 Algoritam digitalnog potpisa primenom eliptične krive	24
4.2.2 Verifikacija digitalnog potpisa ECDSA	25
4.2.3 Dokaz verifikacije digitalnog potpisa	25
4.3 Šifrovanje i dešifrovanje	26
4.3.1 Algoritam šifrovanja javnim ključem primenom eliptičnih krivih	26
4.3.2 Algoritam dešifrovanja privatnim ključem primenom eliptičnih krivih	27
4.4 Uspostavljanje sesijskog ključa	28
4.4.1 Protokol za uspostavljanje sesijskog ključa primenom eliptičnih krivih	28
5. Analiza napada na kriptografske sisteme eliptičnih krivih	31
5.1 Problem diskretnog logaritma u sistemu eliptičnih krivih	31
5.1.1 Polig-Helmanov algoritam napada	32
5.1.2 Polar ro algoritam napada	33
5.2 Napadi na kriptografske ključeve	34

5.2.1 Napadi malih podgrupa	35
5.2.2 Napad primenom lažnih eliptičnih krivih.....	35
5.2.3 Napad prilikom ustanovljavanja ključa	36
6. Oblasti primene kriptografije eliptičnih krivih	39
7. Zaključak	43
Literatura.....	47
Dodatak 1	49
Dodatak 2.....	59

Spisak slika

<i>Slika 1. Sistem za šifrovanje Skitale [2].....</i>	<i>1</i>
<i>Slika 2. Šematski prikaz simetričnog šifarskog sistema</i>	<i>5</i>
<i>Slika 3. Šematski prikaz sekvencijalnog šifarskog sistema</i>	<i>7</i>
<i>Slika 4. Konusni (Aplonijevi) preseki [12]</i>	<i>13</i>
<i>Slika 5. Neki karakteristični oblici grafika eliptičnih krivih [13].....</i>	<i>15</i>
<i>Slika 6. Grafička interpretacija sabiranja dva različita elementa u skupu E.....</i>	<i>16</i>
<i>Slika 7. Grafička interpretacija sabiranja elementa sa samim sobom u skupu E</i>	<i>17</i>
<i>Slika 8. Grafička interpretacija određivanja suprotnog elementa u skupu E.....</i>	<i>17</i>

1. Uvod

Moderno doba ili doba u kojem se sada nalazimo moglo bi se nazvati i doba elektronskih komunikacija. Mnogobrojni digitalni uređaji su pronašli svoju široku primenu, kako u vidu desktop ili laptop računara, tako i u vidu mobilnih uređaja kao što su tablet računari i mobilni telefoni. Elektronski uređaji su sa sobom su doneli gotovo beskrajne mogućnosti. Pogodnosti tih uređaja koristimo svakodnevno, a one podrazumevaju pre svega komunikaciju sa bilo kojom osobom, ma koliko ona bila udaljena od nas, kao i čuvanje mnogobrojnih podataka koji mogu biti poverljivi, bilo da se radi o podacima poslovne ili privatne prirode. Samo postojanje poverljivih podataka, bilo da želimo da ih podelimo sa nekim putem komunikacionog kanala ili jednostavno želimo da ih sačuvamo na svom disku kao i da ih zaštitimo od neovlašćenog pristupa, nameće nam pitanje kako da zaštitimo te podatke.

Pogledom u istoriju možemo zaključiti da problem zaštite podataka nije isključivo pitanje savremenog doba. O tome svedoče pokušaji ljudi kroz istoriju da što uspešnije sakriju dragocene informacije. Još od 1900 godina p.n.e. kada su Egipćani koristili nestandardne hijeroglif u natpisima, što se smatra prvim primerom pisane kriptografije [1] preko stare Grčke u kojoj je razvijen sisteme za šifrovanje „Skitale“ koji je prikazan na Slici 1, pa sve do savremene ere simetričnih i asimetričnih šifarskih sistema. Nesumnjivo je da su tehnike zaštite podataka tokom svog istorijskog razvoja doživele izuzetan razvoj i napredak.



Slika 1. Sistem za šifrovanje Skitale [2]

Naučna oblast koja obrađuje načine zaštite podataka naziva se kriptografija. Može se prevesti kao nauka o tajnom ili skrivenom pisanju. Uporedo sa kriptografijom razvijala se i kriptanaliza. Kriptanaliza je oduvek težila da otkrije tajni sadržaj poruke koji je dobijen primenom nekih od metoda kriptografije. Kriptografi su nastojali da osmisle što bolji sistem za šifrovanje podataka dok su kriptanalitičari pokušavali da dešifruju proizvod šifrovanja bez znanja ulaznih parametara sistema. Takmičenje između kriptografije i kriptanalize se nastavlja i u sadašnjosti.

Kriptografija se zasniva na matematičkim modelima algoritama. Vremenom su razvijani sve bolji matematički modeli koji su zatim pretočeni u algoritme. Algoritam predstavlja skup postupaka koji se sprovodi nad početnom informacijom koja se nalazi u otvorenom obliku, a često se naziva i otvorenim tekstom. Nakon sprovedenog pomenutog skupa operacija koje su definisane algoritmom i nazivaju se procesom šifrovanja, kao rezultat se dobija nerazumljiv podatak koji se naziva šifratom. Proces dešifrovanja predstavlja inverzan proces procesu šifrovanja. Primenom istog algoritma kojim je šifrat dobijen, ali inverznim postupkom, od šifrata se dobija početni tekst ili otvoreni tekst.

Kriptografski ključ predstavlja binarni niz koji se sastoji od nula i jedinica, a čija dužina zavisi od zahteva samog kriptografskog algoritma koji se koristi. Kriptografski ključ, pored otvorenog teksta predstavlja ulazni parametar na osnovu koga se u izabranom algoritmu vrši proces šifrovanja ili dešifrovanja. Kao i svaki drugi ključ, kriptografski ključ zahteva naročito pažljivo rukovanje i čuvanje. Posebna oblast kriptografije se bavi generisanjem, distribucijom i upravljanjem kriptografskim ključevima.

Prilikom generisanja kriptografskog ključa veoma je bitno da je generisani niz zaista slučajan (truly random). Najbolje je osloniti se na fizičke procese u specijalizovanim uređajima kao što je HSM (Hardware Security Module) ili nekim drugim uređajima namenjenim za ove svrhe. Slučajni niz se može formirati i primenom nekog od softverskih alata, međutim, najčešće se dobija niz koji se nakon određenog broja binarnih elemenata ponavlja u sekvencama (pseudo random). Očigledno je da je kriptanalitičarima znatno jednostavnije da zaključie kako će izgledati kriptografski ključ ukoliko se radi o pseudo random nizu. Iz tih razloga softverski način kreiranja kriptografskih ključeva treba izbegavati. U matematici su razvijeni matematički postupci kojima se može pokazati da se neki niz ponavlja nakon određenog broja članova u pravilnim sekvencama.

Kriptografski algoritmi se mogu podeliti u dve osnovne kategorije, a to su simetrični i asimetrični kriptografski algoritmi. Simetrični kriptografski algoritmi za proces šifrovanja i proces dešifrovanja koriste isti kriptografski ključ. Ukoliko neko poseduje kriptografski ključ koji se koristi u simetričnim šifarskim sistemima, tada sa lakoćom može dešifrovati bilo koji šifrat i pristupiti otvorenom tekstu ili kreirati novi šifrat na osnovu nekog otvorenog teksta. Asimetrični šifarski sistemi koriste jedan ključ za šifrovanje, a drugi za dešifrovanje teksta. Jedan od ključeva se naziva tajnim kriptografskim ključem i poseduje ga samo vlasnik, dok se drugi kriptografski ključ naziva javnim i može mu pristupiti svako. To znači da poruku koja je namenjena određenom primaocu, a koja je šifrovana javnim kriptografskim ključem, može dešifrovati samo vlasnik tajnog kriptografskog ključa.

Prilikom šifrovanja simetričnim algoritmom dva učesnika koji komuniciraju razmenjuju kriptografski ključ. Ukoliko u zaštićenoj komunikaciji učestvuje više učesnika potrebno je da svaki učesnik pošalje svoj ključ svakom drugom učesniku, kao i da preuzme ključeve od svih drugih učesnika. Može se lako zaključiti da broj razmenjenih ključeva raste sa brojem učesnika. Broj ukupnih razmena ključeva se može izračunati kao broj kombinacija od n elemenata, odnosno učesnika druge klase, pri čemu se razmene ne ponavljaju. Odnosno:

$$C_2^n = \binom{n}{2} = \frac{n \cdot (n-1)}{2 \cdot 1} = \frac{n \cdot (n-1)}{2} \quad (1)$$

Ukoliko se koristi asimetrični kriptografski algoritam, tada svaki od učesnika treba da preuzme tačno $n-1$ ključeva odnosno svaki od učesnika treba da preuzme javne ključeve ostalih učesnika u komunikaciji.

Prednost korišćenja asimetričnog algoritma je što se javni ključevi preuzimaju u otvorenom obliku jer ne predstavljaju nikakvu tajnu. Kod sistema sa otvorenim šifarskim sistemom za razmenu ključeva, moraju se koristiti posebni algoritmi koji zadržavaju tajnost ključeva. Kao primer algoritma za razmenu kriptografskih ključeva u simetričnom šifarskom sistemu može se navesti Difi-Helmanov algoritam (Bailey Whitfield Diffie, 1944; Martin Edward Hellman, 1945).

Kriptografski sistem koji koristi asimetrične algoritme naziva se sistemom sa javnim ključevima (Public Key Infrastructure - PKI).

Jedan od najbitnijih elemenata PKI sistema je Sertifikaciono telo (Certification Authority). Sertifikaciono telo, pored svojih ostalih funkcija, generiše par javnog i tajnog ključa za svakog korisnika i predstavlja element od poverenja u komunikaciji. Možemo zaključiti da je veoma bitno da Sertifikaciono telo bude pod nadzorom, kako bi se sprečilo njegovo kompromitovanje. Iz tog razloga, postoji čitav niz propisa koji regulišu rad Sertifikacionog tela. Ukoliko bi neko neovlašćeno pristupio Sertifikacionom telu, jasno je da bi kompletna komunikacija u posmatranom PKI sistemu bila ugrožena.

Eliptične krive predstavljaju jedan od oblika glatkih algebarskih krivih. Sve algebarske krive se mogu predstaviti algebarskim izrazom, pa tako i eliptične krive. Eliptične krive predstavljaju ravanske krive, odnosno njihov grafik se nalazi u jednoj ravni. Eliptične krive su takođe i glatke krive, što znači da su neprekidne, odnosno nemaju tačke prekida i diferencijabilne su u svakoj tački u oblasti definisanosti. Detaljnije će biti obrađene u daljem tekstu.

Eliptične krive su pronašle svoju primenu u kriptografiji. Primenu eliptičnih krivih u kriptografiju su nezavisno jedan od drugog uvela dva istraživača i to Nil Koblitz (Neal Koblitz, 1948) i Viktor Miler (Victor Saul Miller, 1947) godine 1985. Tako je nastala Kriptografija eliptičnih krivih (Elliptic curve cryptography ECC), dok je u široku upotrebu uvedena 2004. i 2005. godine.

Posebna pažnja u ovom radu biće posvećena mogućim napadima na kriptografski sistem eliptičnih krivih. Kriptoanaliza je posebna naučna disciplina koja je usko povezana sa kriptografijom. U kriptografiji se dešifrovanje posmatranog šifrata izvršava tako što su poznati svi neophodni kriptografski elementi za dešifrovanje šifrata, kao što su algoritam i kriptografski ključ. Kriptoanaliza predstavlja ustanovljavanje načina dešifrovanja šifrata, ali bez poznatih svih neophodnih kriptografskih elemenata. Kriptografija i kriptoanaliza se zajedničkim imenom nazivaju kriptologija. Kriptografija i kriptoanaliza imaju međusobno suprotstavljene ciljeve. Kriptografija ima za cilj da osigura bezbednost podataka tako što će razviti kriptografski sistem koji pruža visoku bezbednost. Sa druge strane, kriptoanaliza ima za cilj da pronađe slabe tačke, odnosno nedostatke i mane u kriptografskom sistemu i otkrije otvorenu poruku iz šifrata bez poznavanja svih neophodnih kriptografskih elemenata. Postupak koji izводи kriptoanalitičar se naziva još i napad na kriptografski sistem i cilj napada je ugrožavanje bezbednosti šifrovanih podataka i njihovo otkrivanje. Na taj način ne dolazi samo do ugrožavanja napadnutih podataka već i do kompromitovanja celokupnog kriptografskog sistema [3].

Svaki kriptografski sistem mora biti podvrgnut kriptoanalitičkom testiranju. Na taj način se potvrđuje ili opovrgava kvalitet kriptografskog sistema, kao i bezbednost šifrovanih podataka. Kriptografski sistem se ne može smatrati pouzdanim dokle god ne prođe rigorozna kriptoanalitička testiranja [4].

Kriptoanalitičari najčešće koriste značajne računarske resurse kao i posebne postupke, tehnike i procese za napad na kriptografski sistem. Uspešni kriptoanalitičari mogu razotkriti otvoreni tekst iz dela šifrata ne poznavajući kriptografski ključ kojim je šifrovanje izvršeno.

Kriptoanaliza može biti korišćena i u nezakonite svrhe u cilju otkrivanja tajni i sticanja nezakonite koristi, bilo da se radi o materijalnoj koristi ili sticanju određenih saznanja radi sticanja poslovne, političke ili obaveštajne prednosti. Ako se kriptografska zaštita pravilno primeni, moguće je obezbediti dovoljno visok nivo zaštite podataka. Dovoljno visokim nivoom zaštite, smatra se onaj nivo zaštite za koji znamo da napadač ne

može razbiti kriptografski sistem u prihvatljivom vremenskom periodu koristeći sve svoje raspoložive računarske resurse.

Potrebno je imati na umu da najslabija karika u sistemu kriptografske zaštite jeste čovek. Metode dolaženja do kriptografskih podataka korišćenjem osoba koje imaju saznanja o kriptografskim elementima sistema naziva se socijalni inženjering [5].

Osnovni doprinos Kriptografije eliptičnih krivih u PKI sistemu je korišćenje kraćih ključeva uz očuvanje istog stepena bezbednosti šifrovanih podataka. Pored toga, Kriptografija eliptičnih krivih, našla je svoju primenu i u digitalnom potpisivanju.

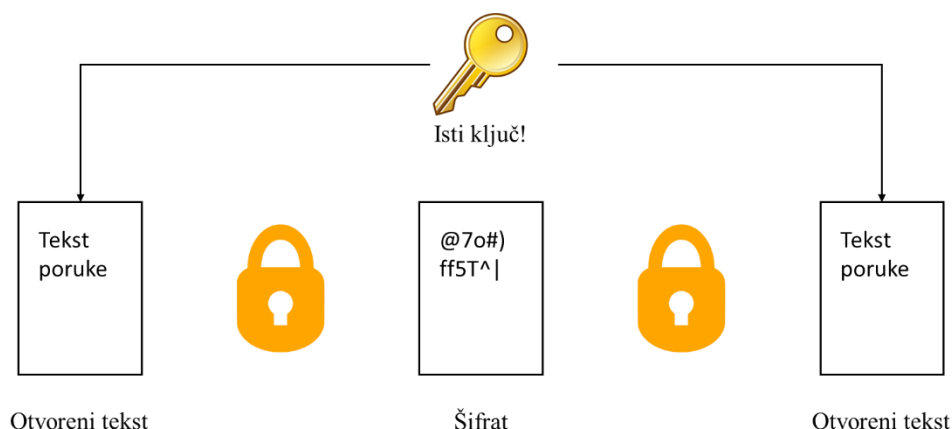
U radu će biti obrađeni osnovni elementi kriptografije eliptičnih krivih. Pored toga biće dat i pregled tipičnih napada na kriptografski sistem eliptičnih krivih.

2. Kriptografski sistemi

Sama reč *kriptografija* potiče od grčkih reči κρυπτος, „skriveno“, i γραφειν „pisanje“ i mogla bi se prevesti kao skriveno pisanje. Kriptografija se može definisati kao nauka koja se bavi načinima i metodima očuvanja tajnosti informacija. Razmena podataka u savremenom svetu se svakim danom sve više umnožava, pri čemu treba imati na umu da se među razmenjenim podacima nalaze i poverljivi podaci. Prilikom razmene podataka, poverljivi podaci mogu biti presretnuti od strane napadača i na taj način doći u posed osoba ili organizacija kojima nisu namenjeni. Takav ishod se naziva kompromitovanjem podataka. Opisani problemi se mogu prevazići šifrovanjem podataka koji na taj način postaju nedostupni napadačima.

Kao što je već pomenuto u uvodu ovog rada, osnovni elementi kriptografskog sistema su kriptografski algoritam i kriptografski ključ, dok su osnovne operacije šifrovanje i dešifrovanje. Pri tome, treba imati na umu da je dužina kriptografskog ključa definisana algoritmom koji se koristi i različita je za različite algoritme. Takođe je pomenuto da se kriptanaliza bavi pokušajem otkrivanja otvorenog teksta bez poznatog kriptografskog ključa, i to se naziva kriptografskim napadom.

Simetrični kriptografski sistem se može prikazati šemom prikazanoj na slici 2:



Slika 2. Šematski prikaz simetričnog šifarskog sistema

Osnovni postulati bezbednosti razmene informacija se mogu iskazati u sledeća četiri pojma [6]:

1. Poverljivost
 2. Autentifikacija
 3. Integritet
 4. Neporecivost
-
1. Poverljivost podataka podrazumeva da će podatke koje je pošiljac poslao moći da vidi jedino primalac kome je poruka namenjena. Pri tome se misli da ukoliko treća osoba dođe u posed šifrovanih podataka neće moći da otkrije sadržaj originalne poruke. Pretpostavimo da se radi o poruci u kojoj se opisuju planovi novog proizvoda neke kompanije. Ukoliko bi neovlašćena osoba bila u mogućnosti da dešifruje podatke moglo bi doći do otkrivanja bitnih planova za razvoj pomenute kompanije i do gubitka komparativne prednosti na tržištu.

2. Pod pojmom autentifikacija se podrazumeva utvrđivanje identita sagovornika sa kojim se podaci razmenjuju. U svakodnevnom razgovoru identitet osobe sa kojom razgovaramo utvrđujemo na osnovu više različitih parametara. Podrazumeva se da osobu poznajemo pa znamo kako izgleda, prepoznavamo njen glas ili ukoliko razgovaramo telefonom prepoznavamo i broj telefona sa koga nas osoba poziva. Pored toga, postoje i lični dokumenti na kojima se nalaze podaci o osobi, pa se identitet osobe može utvrditi i na osnovu ličnih dokumenata. U svetu elektronske razmene utvrđivanje identiteta nije tako jednostavno i lako se može dogoditi da se neka treća osoba predstavi kao validni sagovornik jer ne postoji ni vizuelni, ni fizički kontakt.
3. Integritet podataka koji se razmenjuju podrazumeva da su podaci koje je primalac poruke primio identični podacima koje je pošiljalac poruke poslao. Ukoliko kupac u prodavnici napiše ček na određeni iznos, pri čemu prodavac posmatra kako kupac popunjava ček, sigurno je da će prodavac biti uveren kako su podaci na čeku koje je kupac popunio originalni. Prilikom elektronskih transakcija potrebno je uvesti posebne mehanizme kako bi se utvrdilo da je poslata poruka identična onoj koju je primalac primio. Potrebno je da poruka bude osigurana od moguće izmene podataka.
4. Neporecivost podrazumeva da se poslata poruka ne može negirati po bilo kom osnovu. Kako po pitanju samog sadržaja, tako i po pitanju identiteta osobe koja je poruku poslala. Vratimo se još jednom na primer plaćanja čekom u prodavnici. Kada kupac popuni ček pred prodavcem i preda ga prodavcu, nakon toga nije moguće da u banci tvrdi da je iznos na čeku pogrešan ili da je neko drugi popunio ček umesto njega. Banka ima tačnu evidenciju čekova koje njihov klijent poseduje, dok prodavac utvrđuje identitet kupca kako uvidom u lične dokumente, tako i upoređivanjem potpisa kupca na kartici i potpisa na čeku. U svetu elektronske razmene podataka postoje odgovarajući mehanizmi kojima se sprovodi opisano, iako ne postoji ni vizuelni ni fizički kontakt.

Kriptografske tehnike koje se koriste da bi se implementirali pomenuti bezbednosni postulati su šifra i digitalni potpis. Možemo zaključiti da se za zaštitu podataka kao osnovni element zaštite koristi kompleksan šifarski sistem. Potrebno je napomenuti da se tajnost podataka realizuje primenom simetričnih šifarskih sistema, dok se primena tehnologije digitalnog potpisa zasniva na asimetričnim šifarskim sistemima odnosno na PKI sistemima.

Kao što je već pomenuto, postoje dva osnovna tipa šifarskih sistema i to su simetrični i asimetrični šifarski sistemi o kojima će biti reči u narednom tekstu [7].

2.1 Simetrični kriptografski sistemi

Osnovna odlika simetričnih šifarskih sistema je da se za šifrovanje i dešifrovanje koristi isti ključ. Pomenuta odlika je praktična sa stanovišta korisnika, ali je istovremeno i bitan nedostatak. Kriptografski ključ predstavlja tajnu vrednost koju poznaju samo pošiljalac i primalac podataka. Kriptografski ključ predstavlja deljenu tajnu i zato joj treba posvetiti posebnu pažnju. U tom cilju, razvijeni su posebni algoritmi za razmenu simetričnog kriptografskog ključa. Ukoliko napadač otkrije kriptografski ključ, moguće je da sam šifrue i

dešifruje podatke, bez znanja primaoca i pošiljaoca podataka. Na taj način bi se narušili postulati bezbednosti koji su prethodno opisani.

Simetrični šifarski sistemi se mogu podeliti u dve osnovne grupe:

1. Sekvencijalni šifarski sistemi
2. Blok šifarski sistemi.

U narednom tekstu će biti posebno obrađen svaki od pomenutih sistema.

2.1.1 Sekvencijalni šifarski sistemi

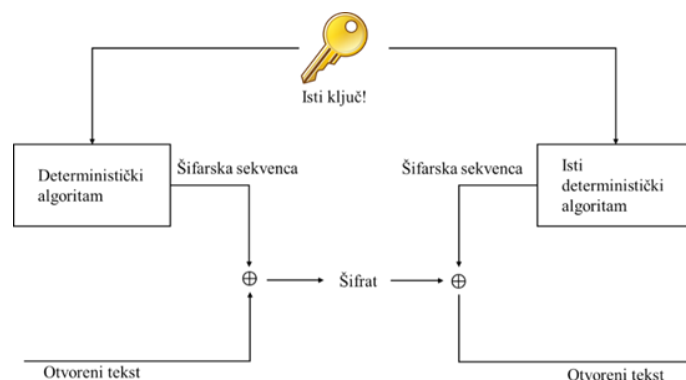
Pre nego što budu obrađeni sekvencijalni simetrični šifarski sistemi, biće dato kratko podsećanje iz matematike koje je neophodno za razumevanje funkcionisanja sekvencijalnih šifarskih sistema.

Prilikom realizacije sekvencijalnih šifarskih sistema koristi se sabiranje po modulu 2, odnosno sabiranje u binarnom numeričkom sistemu. Sabiranje po modulu 2 se označava matematičkim simbolom $\oplus$ ili u različitim programskim jezicima naredbom XOR (Exclusive OR), a naziva se još i logičkim sabiranjem. Binarni sistem sa sastoji od dva osnovna elementa, a to su 0 (nula) i 1 (jedan). Pravilo koje važi prilikom binarnog sabiranja dato je u Tabeli 1:

$\oplus$	0	1
0	0	1
1	1	0

Tabela 1. Šematski prikaz simetričnog šifarskog sistema

Osnovna ideja sekvencijalnog šifarskog sistema je da se binarni niz koji se šifrjuje, sabere po modulu 2 sa kriptografskim ključem, koji je takođe binarni niz i na taj način se dobija niz koji predstavlja šifrat [8]. Kriptografski ključ u vidu binarnog niza se generiše korišćenjem algoritma. Kako je već ranije napomenuto svaki slučajni niz koji se definiše algoritmom predstavlja pseudoslučajni niz. U slučaju sekvencijalnog šifarskog sistema neophodno je korišćenje algoritma kako bi se isti niz generisao, kako na strani pošiljaoca tako i na strani primaoca poruke. Takav algoritam koji generiše pseudoslučajne brojeve naziva se determinističkim algoritmom. Binarni nizovi koje generišu deterministički algoritmi imaju slične osobine kao i slučajni nizovi. Sekvencijalni šifarski sistem se može prikazati šemom prikazanoj na slici 3:



Slika 3. Šematski prikaz sekvencijalnog šifarskog sistema

Pseudoslučajni nizovi su periodični u širem smislu, što znači da mogu imati aperiodični početak. Period šifarskog niza označava broj binarnih elemenata posle koga se elementi ponavljaju i mora da bude barem jednake dužine kao i dužina niza koji se šifruje. U praksi, generišu se nizovi čiji je period mnogo puta veći od dužine niza koji se šifruje.

Kao primer sekvencijalnog šifarskog algoritma možemo navesti RC4 algoritam - Rivest Cipher 4 (Ronald Rivest 1947).

2.1.2 Blok šifarski sistemi

Prilikom korišćenja blok šifarskih sistema, originalna poruka se deli na blokove koji se šifruju korišćenjem kriptografskih ključeva i to po blokovima od dva ili više elemenata. Svaki element se šifruje drugačije i to u zavisnosti od načina šifrovanja susednih elemenata koji se nalaze u istom bloku. Svaki od blokova elemenata se šifruje uvek na isti način, bez obzira na mesto na kome se nalazi u otvorenom tekstu. Imajući u vidu ovu činjenicu, očigledno je da isti otvoreni tekst, koji se šifruje identičnim ključem, daje uvek šifrate koji su istovetni. Dešifrovanje poruke se najčešće radi od početka šifrata do njegovog kraja, pa se na taj način dobija otvoreni tekst. Može se primetiti da ukoliko želimo da dešifrujemo samo deo poruke koji nas interesuje, nije obavezno da se vrši dešifrovanje cele poruke od samog početka. Moguće je izvršiti dešifrovanje samo tog bloka koji nas zanima. Osnovni elementi blok šifarskog sistema su:

- a) Inicijalna ili početna transformacija
 - b) Kriptografski slaba funkcija koja se ponavlja r puta
 - c) Finalna ili konačna transformacija
 - d) Algoritam za ekspanziju ili razvoj ključa
-
- a) Inicijalna ili početna transformacija može da sadrži jednu ili dve funkcije. Prva funkcija randomizuje ili maskira ulazne podatke tj. vrši izmenu blokova koji sadrže samo nule ili samo jedinice. Pomenuta funkcija, najčešće, ne zavisi od kriptografskog ključa. Druga funkcija ima za cilj da što je moguće više oteža pojedine vrste kriptografskih napada na blok šifarske sisteme. Ona zavisi od kriptografskog ključa.
 - b) Svaka runda ili svako ponavljanje slabe kriptografske funkcije čini jedna nelinearna funkcija čiji su parametri delovi kriptografskog ključa kao i delovi ulaznih podataka. Nelinearna funkcija može biti jednosmerna, ali i ne mora. Nelinearna funkcija može sadržati samo jednu operaciju koja je veoma složena ili može sadržati niz uzastopnih, a međusobno različitih, jednostavnijih transformacija. Runde se međusobno povezuju sabiranjem po modulu 2, svakog pojedinačnog bita sa podacima koji dolaze iz prethodne runde. U prvom koraku se uzimaju podaci koji se dobijaju nakon inicijalne transformacije. Dešifrovanje se vrši na isti način, ali obrnutim redom, korišćenjem istog kriptografskog ključa. Tako se iz šifrata rekonstruiše otvoreni tekst.
 - c) Finalna ili konačna transformacija obezbeđuje da operacije šifrovanja i dešifrovanja budu simetrične.

- d) Algoritam za ekspanziju ili razvoj ključa ima ulogu da pretvori kriptografski ključ, koji je najčešće ograničene dužine, u više podključeva koji se sastoje od više bitova.

Blok šifarski sistemi se najčešće koriste za šifrovanje kratkih poruka kao što su lozinke, kriptografski ključevi, digitalni potpis, identifikacioni podaci i slično. Blok šifarski sistemi nisu zgodni za šifrovanje velikih količina podataka, kao što su tabele, tekst koji je formatiran i pogotovo grafički fajlovi. Strukturu navedenih dokumenata bi bilo lako odrediti.

Kao neki od primera blok šifarskih sistema navode se: DES, AES, BLOWFISH, TWOFISH itd.

Jedan od osnovnih problema u kriptografiji sa tajnim ključevima je distribucija ili razmena kriptografskih ključeva. Dva korisnika koji u međusobnoj komunikaciji razmenjuju šifrovane podatke moraju pre početka komunikacije da izaberu tajni ključ. Za razmenu tajnog ključa učesnici u komunikaciji moraju koristiti bezbedan kanal. Ako se desi da siguran kanal za razmenu ključa ne postoji, tada razmena ključa predstavlja problem koji može značajno ugroziti bezbednost komunikacije.

Još jedan važan nedostatak blok šifarskih sistema je nemogućnost sprovođenja procedure digitalnog potpisivanja. Naime, u šifarskim sistemima sa tajnim ključevima ne postoji mogućnosti, u opštem slučaju, da se poruke digitalno potpišu. U tom slučaju primalac poruke nije siguran da je pošiljalac poruke ujedno i njen autor.

2.2 Asimetrični kriptografski sistemi

Za razliku od simetričnih šifarskih sistema kod kojih se šifrovanje i dešifrovanje vrši istim kriptografskim ključem, kod asimetričnih šifarskih sistema se koriste dva ključa. Ključevi su u stvari dva broja koji moraju biti uzajamno prosti. Jedan od tih ključeva koji je javno dostupan naziva se javni ključ, dok se drugi ključ naziva tajni ključ i dostupan je samo njegovom vlasniku. U asimetričnom šifarskom sistemu postoje dve familije parova funkcija. Prva funkcija služi za šifrovanje i označava se sa E što je uobičajena oznaka od engleske reči za šifrovanje – *encryption*. Druga funkcija služi za dešifrovanje i označava se sa D što je uobičajena oznaka od engleske reči za dešifrovanje – *decryption*. Za šifrovanje se koristi prvi ključ ili k_1 dok se za dešifrovanje koristi drugi ključ k_2 . Funkcije za šifrovanje i dešifrovanje se mogu označiti i kao E_{k_1} - šifrovanje ključem k_1 i D_{k_2} dešifrovanje ključem k_2 . Sada se sa M označi skup otvorenih poruka. Njegov naziv dolazi od engleske reči *message*. Dalje se sa X označi skup šifrata pošto se u matematici kao najčešća oznaka za nepoznatu veličinu uzima slovo x . Šifrovanjem se M prevodi u X , dok je dešifrovanje obrnut proces koji X prevodi u M . To se može napisati i na sledeći način:

$$E_{k_1} : M \rightarrow X \quad (2)$$

$$D_{k_2} : X \rightarrow M \quad (3)$$

Dalje, za svaku otvorenu poruku m koja se nalazi u skupu M važi sledeće:

$$D_{k_2}(E_{k_1}(m)) = m \quad (4)$$

Pre nego što se započne sa korišćenjem opisanog sistema, učesnici u komunikaciji A i B se moraju dogovoriti koji par ključeva k_1, k_2 će koristiti. Prethodno se moraju utvrditi parovi uzajamno prostih brojeva koji se mogu koristiti kao kriptografski ključevi. Parovi uzajamno prostih brojeva se mogu prikazati kao elementi skupa koji se označava sa K i iz koga učesnici biraju par ključeva koji žele da koriste. Pretpostavimo da učesnik u komunikaciji A šalje poruku m iz skupa M učesniku u komunikaciji B . Učesnik A šifrue i šalje poruku m koristeći funkciju E_{k_1} što se može prikazati sledećim izrazom:

$$E_{k_1}(m) = c \quad (5)$$

Nakon izračunavanja, učesnik A šalje izračunatu vrednost c korisniku B .

Učesnik u komunikaciji B , kome je poruka i namenjena, želi da dođe do sadržaja otvorene poruke. Kako bi rekonstruisao izvornu poruku koju mu je A poslao, B mora da dešifruje šifrat c koji je dobio od A . U tu svrhu koristi funkciju za dešifrovanje D_{k_2} (4)

Kada se u kriptografiji kaže da je neki proračun lako izračunljiv tada se misli na činjenicu da se izračunavanje može izvršiti u prihvatljivo kratkom vremenskom roku. Na primer ako se neki proračun može izvršiti u roku od nekoliko sati, pa u određenim slučajevima čak i u roku od nekoliko dana, taj proračun se smatra lako izračunljivim. Ukoliko bi za izračunavanje određenog problema bile potrebne stotine ili hiljade godina, uz korišćenje svih raspoloživih resursa, za takav problem se kaže da je teško izračunljiv ili intraktabilan.

U realnim primenama asimetričnog šifarskog sistema parovi funkcija E_{k_1}, D_{k_2} moraju biti izabrani tako da su lako izračunljivi za učesnike u komunikaciji, a da su pri tome teško izračunljivi za napadača. Napadač ili kriptanalitičar neovlašćeno pokušava da dođe do otvorenog teksta m pri čemu poseduje samo šifrat c što nije ni malo jednostavan zadatak.

Infrastruktura sa javnim ključevima – Public Key Infrastructure – PKI

Kako bi se u praksi koristio šifarski sistem sa javnim ključevima, neophodno je da se definiše funkcija koja se naziva jednosmerna funkcija (One-Way Function – OWF [9]):

$$f: M \rightarrow X \quad (6)$$

Definisana jednosmerna funkcija mora biti lako izračunljiva na način na koji je to opisano u prethodnom tekstu:

$$f(m) = c \quad (7)$$

Istovremeno je neophodno da se funkcija koja je inverzna jednosmernoj funkciji teško izračunava, na način na koji je to opisano u prethodnom tekstu. Inverzna funkcija se predstavlja na sledeći način:

$$f^{-1}(c) = m \quad (8)$$

Postoji mogućnost da se za jednosmernu funkciju može izračunati njoj inverzna funkcija ukoliko se poznaje neka dodatna informacija. Neophodna informacija se naziva zamka, a za

takvu jednosmernu funkciju se kaže da je funkcija sa zamkom (*Trapdoor One-Way Function* - *TOF*).

Prilikom primene šifarskog sistema sa javnim ključem mora biti ispunjeno sledeće:

Posmatra se familija jednosmernih funkcija koje u sebi sadrže zamku. Korisnik se označi slovom U od engleske reči *user* i izabere na slučajan način ključ k_u iz skupa K . Posle toga korisnik objavljuje E_u uz pomoć koga može da se izračuna funkcija f_u koja predstavlja šifrovanje poruke m . Vrednost E_u predstavlja javni ključ korisnika U . Funkcija $t(k_u)$ se naziva zamka pomoću koje se može izračunati inverzna funkcija f_u^{-1} i tako doći do tajnog ključa kojim se može izvršiti dešifrovanje šifrata c .

Ukoliko učesnik u komunikaciji A hoće da pošalje poruku m drugom učesniku u komunikaciji B , pre svega je potrebno da u katalogu javnih ključeva pronađe javni ključ učesnika B , što je E_b . Nakon toga, A šalje poruku $f_b(m) = c$ korisniku B . Pošto je poruka namenjena učesniku B isključivo on može da izračuna inverznu funkciju f_b^{-1} i na taj način jedino učesnik B može da rekonstruiše originalnu poruku m .

$$f_b^{-1}(c) = f_b^{-1}(f_b(m)) = m \quad (9)$$

Treba imati na umu da postoji matematički problem u šifarskim sistemima sa javnim ključem. Problem se sastoji od toga što ne postoji matematički dokaz da postoje jednosmerne funkcije. Takođe ne postoji ni matematički dokaz da postoje jednosmerne funkcije sa zamkom. Iako ne postoje precizni matematički dokazi, postoje dve funkcije za koje se može smatrati da poseduju pomenuta svojstva:

- Funkcija koju čini proizvod dva cela broja, čija je inverzna funkcija rastavljanje dobijenog rezultata na proste činioce.
- Diskretno stepenovanje, čija je inverzna funkcija diskretni logaritam

Dve pomenute funkcije se mogu smatrati lakim za izračunavanje, pri čemu postoji uverenje da se teško mogu izračunati njihove inverzne funkcije. Ne postoji precizan matematički dokaz i to može izazvati sumnju u osnovne postavke šifarskog sistema sa javnim ključem.

Pretpostavka je da postoji ceo broj n i postoji uverenje da je teško rastaviti broj n na proste činioce. Što se tiče druge funkcije, ako postoje celi brojevi a i b , postoji uverenje da je teško izračunati x tako da je $a^x = b$

Bezbednost šifarskih sistema sa javnim ključevima, koji se danas primenjuju, meri se pre svega brojem operacija koje je neophodno izvršiti da bi se izračunala inverzna funkcija, mada ne postoji matematički dokaz da ne postoji algoritam za lako izračunavanje inverznih funkcija.

Asimetrični šifarski algoritam - RSA (Rivest Shamir Adleman) algoritam je jedan od najviše korišćenih algoritama. Značajna primena RSA algoritma je u sistemima elektronskog plaćanja.

2.3 Digitalni potpis

Digitalni potpis je definisan od strane američkog NIST-a (National Institute of Standards and Technology) kao standard digitalnog potpisa DSS (Digital Signature Standard). Standard NIST FIPS PUB 186-2 DSS se bazira na kriptografiji sa javnim ključem koja koristi dva ključa: javni i privatni ključ. DSS uključuje tri algoritma za digitalni potpis i to:

- RSA (Rivest Shamir Adleman)
- DSA (Digital Signature Algorithm)
- ECDSA (Elliptic Curve DSA)

RSA je dobro poznat algoritam, dok će primena eliptičnih krivih biti posebno razmatrana u ovom radu. U narednom delu teksta će biti opisan DSA algoritam za generisanje i proveru digitalnog potpisa [10].

DSA generisanje ključeva

Strana A u komunikaciji izvršava sledeće:

1. Izbor prostog broja q takvog da je $2^{2159} < q < 2^{2160}$
2. Izbor 1024 bitnog prostog broja p sa osobinom $q|(p-1)$
3. Izbor elementa $h \in Z_p^*$ i izračunavanje $g = h^{\frac{p-1}{q}} \bmod p$. Izračunavanje se ponavlja sve dok se ne dobije $g \neq 1$
4. Izbor slučajnog celog broja $x \in [1, q-1]$
5. Izračunavanje $y = g^x \bmod p$
6. Javni ključ A je (p, q, g, y)
7. Privatni ključ A je (x)

DSA generisanje potpisa

U cilju potpisivanja poruke m , A izvršava sledeće:

1. Izbor slučajnog celog broja $k \in [1, q-1]$
2. Izračunavanje $r = (g^k \bmod p) \bmod q$
3. Izračunavanje $k^{-1} \bmod q$
4. Izračunavanje $s = (k^{-1}(h(m) + x \cdot r) \bmod q$, gde $h(m)$ označava SHA-1 heš vrednost otvorene poruke m .
5. Ako je $s = 0$, vratiti se na početni korak.
6. Digitalni potpis poruke m je uređeni par celih brojeva (r, s)

DSA verifikacija potpisa

U cilju verifikacije potpisa poruke m , B izvršava sledeće:

1. Uzima javni ključ A (p, q, g, y)
2. Izračunavanje $w = s^{-1} \bmod q$ i $h(m)$
3. Izračunavanje $u_1 = h(m) \cdot w \bmod q$ i $u_2 = r \cdot w \bmod q$
4. Izračunavanje $v = (g^{u_1} \cdot y^{u_2} \bmod p) \bmod q$
5. Verifikacija digitalnog potpisa je uspela ako i samo ako je $v = r$

Na ovaj način je digitalni potpis verifikovan.

3. Eliptične krive

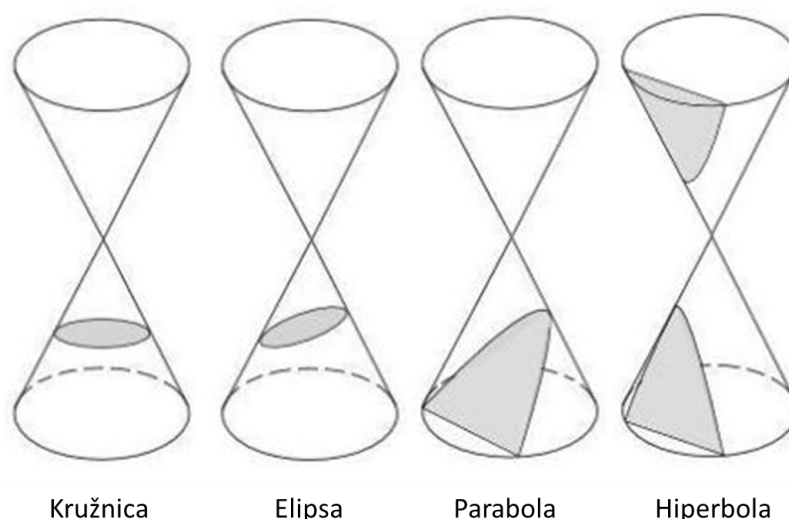
Eliptične krive predstavljaju familiju glatkih algebarskih krivih, što znači da se mogu opisati algebarskim izrazom i da im je prvi izvod definisan u svakoj tački domena na kome je kriva definisana. Potrebno je naglasiti da eliptične krive nemaju nikakve veze sa elipsama ili drugim konusnim presecima [11]. Eliptične krive se koriste u različitim oblastima matematike, i to od teorije brojeva do kompleksne analize, kao i u matematičkoj fizici. Posebnu primenu eliptične krive imaju u kriptografiji, što je i centralna tema ovog rada. Pre definisanja eliptičnih krivih biće dati osnovni elementi algebarskih krivih.

3.1 Algebarske krive

Algebarske krive predstavljaju skup tačaka u ravni koje se mogu definisati algebarskim izrazom:

$$f(x, y) = 0 \quad (10)$$

Sama ravan u kojoj su krive definisane predstavljena je dvodimezionim Euklidovim prostorom. Najviši stepen u polinomu određuje red algebarske krive. Familiju algebarskih krivih drugog reda, koje su i najčešće proučavane krive, čine konusni preseki ili Apolonijevi preseki. Prvi ih je definisao Apolonije iz Pergama u staroj Grčkoj (262. p.n.e. – 190. p.n.e.). Pretpostavimo da postoje dva identična kružna saosna konusa spojena vrhovima. Ukoliko se konusi preseku nekom ravni dobijaju se krive koje se definišu kao konusni preseki. Konusni preseki su prikazani na Slici 4:



Slika 4. Konusni (Aplonijevi) preseki [12]

Algebarski izrazi koji definišu konusne preseke su:

Kružnica:

$$(x - p)^2 + (y - q)^2 = r^2 \quad (11)$$

Gde su p i q koordinate tačke centra kružnice, dok je r poluprečnik kružnice.

Elipsa:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (12)$$

Gde su a i b poluose elipse.

Parabola:

$$y^2 = 2px \quad (13)$$

Gde je p rastojanje između tačke žiže i prave direktrise.

Hiperbola:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \quad (14)$$

Gde je a realna poluosa i b imaginarna poluosa.

Algebarski izrazi koji definišu konusne preseke su drugog stepena, pa sledi da su konusni preseki algebarske krive drugog reda.

Algebarske krive trećeg stepena se mogu javiti u više različitih oblika. Najjednostavniji oblik algebarske krive trećeg reda je:

$$y = ax^3 + bx^2 + cx + d \quad (15)$$

Pri čemu mora biti zadovoljen uslov da je $a \neq 0$. Prikazani algebarski izraz opisuje kubnu parabolu.

Jedna od familija algebarskih krivih trećeg reda su i eliptične krive.

3.2 Definisanje eliptičnih krivih

Eliptične krive predstavljaju skup tačaka u ravni čiji je položaj definisan sledećim algebarskim izrazom:

$$y^2 = x^3 + ax + b \quad (16)$$

Navedeni izraz naziva se još i Vajerštrasovom normalnom formom eliptičnih krivih (Karl Theodor Wilhelm Weierstrass, 1815 – 1897).

Pri čemu se izraz:

$$\Delta = 4a^3 + 27b^2 \quad (17)$$

naziva Diskriminantom polinoma i mora biti različit od nule ($\Delta \neq 0$). Na ovaj način se postiže da kriva nije singularna, odnosno da je regularna, a to sledi iz uslova:

$$f(x, y) = \frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = 0 \quad (18)$$

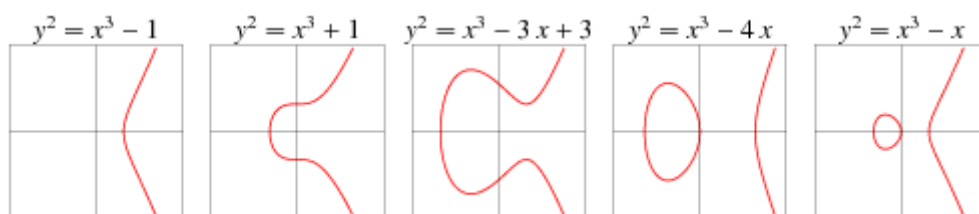
Ispunjavanjem navednog uslova se postiže da su koreni polinoma $x^3 + ax + b$ različiti.

Eliptičnoj krivoj se može dodati tačka O koja se nalazi u beskonačnosti, odnosno čije su koordinate $O(\infty, \infty)$ i tada se može formirati skup E , koji pored tačaka eliptične krive sadrži i tačku O i to na sledeći način:

$$E = \{(x, y): y^2 = x^3 + ax + b\} \cup \{O\} \quad (19)$$

3.2.1 Geometrija eliptičnih krivih

Grafik eliptične krive može imati različite oblike, u zavisnosti od parametara a i b . Neki od karakterističnih oblika grafika eliptičnih krivih su dati na Slici 5:



Slika 5. Neki karakteristični oblici grafika eliptičnih krivih [13]

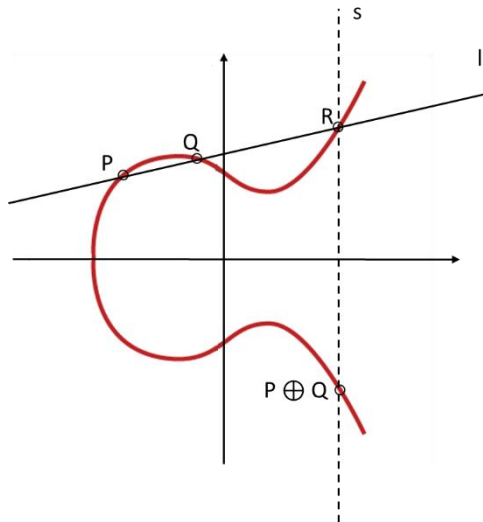
3.2.2 Sabiranje dve različite tačke eliptične krive

Uočimo dve različite tačke P i Q iz skupa E , tako da tačke P i Q pripadaju grafiku eliptične krive. Poznato je da se kroz dve tačke u ravni može postaviti jedna i samo jedna prava. Postavimo pravu l tako da sadrži tačke P i Q . Očigledno je da je prava l sečica grafika eliptične krive. Ukoliko su tačke P i Q sa koordinatama $P(x_1, y_1)$ i $Q(x_2, y_2)$ tada se koeficijent pravca prave l može izračunati na sledeći način:

$$k = \operatorname{tg} \alpha = \frac{y_2 - y_1}{x_2 - x_1} \quad (20)$$

U opštem slučaju, prava l će presecati grafik eliptične krive u još jednoj, trećoj tački koja takođe pripada grafiku eliptične krive. Označimo sa R treću presečnu tačku prave l sa grafikom eliptične krive. Postavimo sada pravu s koja sadrži tačku R tako da je prava s paralelna y osi pravouglog koordinatnog sistema, pri čemu je istovremeno normalna na x osu pravouglog koordinatnog sistema. Presek prave s i grafika eliptične krive označimo sa $-R$. Očigledno je da je tačka $-R$ osno simetrična sa tačkom R , u odnosu na x osu pravouglog koordinatnog sistema. Na ovaj način je definisano sabiranje tačaka P i Q koji se nalaze na grafiku eliptične krive, odnosno elemenata skupa E ili u oznaci $E, P \in E$ i $Q \in E$. Tačka $-R$,

koja predstavlja rezultat sabiranja tačaka P i Q takođe se nalazi na grafiku eliptične krive, odnosno element je skupa E , ili u oznaci $-R \in E$. Iz navednog se može zaključiti da je skup E zatvoren u odnosu na sabiranje. Odnosno, zbir bilo koja dva elementa skupa E je takođe element skupa E . Zbir dva elementa P i Q iz skupa E se označava kao $P \oplus Q$ ili $P + Q$. Grafička interpretacija opisanog je prikazana na Slici 6:



Slika 6. Grafička interpretacija sabiranja dva različita elementa u skupu E

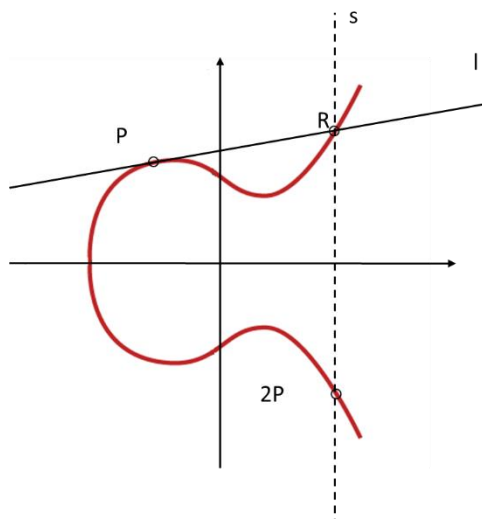
3.2.3 Sabiranje tačke eliptične krive sa samom sobom

Postavlja se pitanje, kako je moguće sabrati tačku eliptične krive samu sa sobom, kada kroz uočenu tačku prolazi beskonačno mnogo pravih u posmatranoj ravni. Ukoliko pođemo od razmišljanja kao u prethodnom primeru sabiranja dve različite tačke, doći ćemo do zaključka da je potrebno postaviti tangentu u tački P na eliptičnu krivu.

Imajući u vidu izraz (20) i ako sada pustimo da koordinata x_2 teži x_1 , tačka Q će težiti P duž grafika eliptične krive. Zato će koeficijent pravca sečice postepeno težiti koeficijentu pravca tangente krive kako se x_2 približava x_1 . Zato iz (20) dobijamo:

$$k_t = \lim_{x_2 \rightarrow x_1} \frac{y_2 - y_1}{x_2 - x_1} \quad (21)$$

Gde je k_t koeficijent pravca tangente. Označimo sada sa l tangentu na eliptičnu krivu u tački $P(x_1, y_1)$. Uočimo sada tačku preseka tangente l sa grafikom eliptične krive i označimo je sa R . Analogno, kao i u slučaju sabiranja dve različite tačke, postavimo sada pravu s koja sadrži tačku R tako da je prava s paralelna y osi pravouglog koordinatnog sistema, pri čemu je istovremeno normalna na x osu pravouglog koordinatnog sistema. Presek prave s i grafika eliptične krive označimo sa $-R$. Očigledno je da je tačka $-R$ osno simetrična sa tačkom R u odnosu na x osu pravouglog koordinatnog sistema. Tačka $-R$, koja predstavlja rezultat sabiranja tačaka P sa samom sobom i takođe se nalazi na grafiku eliptične krive, odnosno element je skupa E , ili u oznaci $-R \in E$. Ili u oznaci $P \oplus P$ ili $2P$. Grafička interpretacija prethodno opisanog je data na Slici 7.

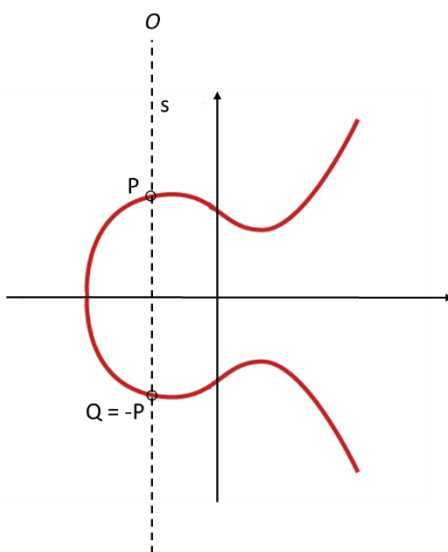


Slika 7. Grafička interpretacija sabiranja elementa sa samim sobom u skupu E

3.2.4 Određivanje suprotnog elementa

Ako se tačka P nalazi na grafiku eliptične krive ili u oznaci $P \in E$, možemo uočiti njoj osno simetričnu tačku u odnosu na osu x u pravouglom koordinatnom sistemu. Osno simetričnu tačku dobijamo tako što postavimo pravu s kroz tačku P , tako da je paralelna sa y osom pravouglog koordinatnog sistema i normalna na x osu pravouglog koordinatnog sistema. Označimo sa $-P$ tačku preseka prave s sa grafikom eliptične krive. Kako bismo izvršili sabiranje $P \oplus -P$ potrebna nam je i treća tačka preseka sa grafikom eliptične krive koja u ovom slučaju ne postoji. Postavlja se pitanje kako izvršiti opisano sabiranje?

Do rešenja dolazimo imajući u vidu obrazac (19) u kome kao treću tačku uzimamo tačku O koja je takođe član skupa E . Pošto se tačka O nalazi u beskonačnosti definiše se pravilo da se tačka O može smatrati tačkom bilo koje prave paralelne sa y osom koordinatnog sistema odnosno bilo koje vertikale. Grafička interpretacija je prikazana na Slici 8:



Slika 8. Grafička interpretacija određivanja suprotnog elementa u skupu E

3.2.5 Algebra eliptičnih krivih

Nakon grafičke interpretacije potrebno je definisati i algebarsku interpretaciju operacija nad skupom E . Osobine sabiranja elemenata u skupu E se mogu definisati sledećim svojstvima:

1. $P + O = O + P = P \quad \forall P \in E$, neutralnost elementa O u odnosu na sabiranje
2. $P + (-P) = O \quad \forall P \in E$, postojanje suprotnog elementa u odnosu na sabiranje
3. $P + (Q + R) = (P + Q) + R \quad \forall P, Q, R \in E$, asocijativnost u odnosu na sabiranje
4. $P + Q = Q + P \quad \forall P, Q \in E$, komutativnost u odnosu na sabiranje

Pretpostavimo da želimo da saberemo dve tačke iz skupa E i to:

$$P(x_1, y_1) \wedge Q(x_2, y_2) \quad (22)$$

Pri čemu je skup E skup tačaka eliptične krive takav da je:

$$E: y^2 = x^3 + ax + b \quad (23)$$

Povezaćemo tačke P i Q pravom l takva da je:

$$l: y = \lambda x + v \quad (24)$$

Gde je λ koeficijent pravca prave l provučen kroz tačke eliptične krive P i Q , dok je v odsečak te iste prave na y osi. U opštem slučaju λ predstavlja nagib prave l prema x osi, odnosno tangens ugla α koji prava l zaklapa sa pozitivnim delom x ose, a predstavlja se formulom:

$$\lambda = \operatorname{tg} \alpha \quad (25)$$

U slučaju eliptičnih krivih važi sledeće:

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & \text{ako je } P \neq Q \\ \frac{3x_1^2 + a}{2y_1} & \text{ako je } P = Q \end{cases} \quad (26)$$

Dok je:

$$v = y_1 - \lambda x_1 \quad (27)$$

Koordinate presečne tačke prave l (24) i eliptične krive E (23) koju smo prethodno označili sa R dobijamo rešavanjem sledeće jednačine:

$$(\lambda x + v)^2 = x^3 + ax + b \quad (28)$$

Pošto je u pitanju jednačina trećeg stepena jasno je da je, u opštem slučaju, moguće dobiti tri realna i različita rešenja. Imajući u vidu da su nam već poznata dva rešenja, odnosno x koordinate tačaka P i Q (x_1, x_2) potrebno je da pronađemo x koordinatu tačke R , odnosno x_3 .

Ukoliko izvršimo transformaciju polinoma na sledeći način:

$$\begin{aligned} x^3 + ax + b - (\lambda x + \nu)^2 &= \\ (x - x_1)(x - x_2)(x - x_3) &= \\ x^3 - (x_1 + x_2 + x_3)x^2 + (x_1x_2 + x_1x_3 + x_2x_3)x - x_1x_2x_3 \end{aligned} \quad (29)$$

Izjednačavanjem koeficijenata po x^2 dobijamo:

$$-\lambda^2 = -x_1 - x_2 - x_3 \quad (30)$$

Odakle sledi da je:

$$x_3 = \lambda^2 - x_1 - x_2 \quad (31)$$

Nakon toga izračunavamo y_3 na sledeći način:

$$y_3 = \lambda x_3 + \nu \quad (32)$$

Na taj način smo izračunali koordinate tačke R, odnosno, s obzirom na prethodno:

$$P + Q = (x_3, -y_3) \quad (33)$$

Imajući u vidu prethodno izneto mogu se formulisati sledeća pravila prilikom sabiranja tačaka eliptičnih krivih, odnosno sabiranja tačaka u skupu E:

1. Ako je $P \neq Q$ i pri tome je $x_1 = x_2$ tada je $P + Q = O$
2. Ako je $P = Q$ i pri tome je $y_1 = 0$ tada je $P + Q = 2P = O$
3. Ako je $P \neq Q$ i pri tome je $x_1 \neq x_2$ tada je:

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} \text{ i } \nu = \frac{y_1x_2 - y_2x_1}{x_2 - x_1}$$
4. Ako je $P = Q$ i pri tome je $y_1 \neq 0$ tada je:

$$\lambda = \frac{3x_1^2 + a}{2y_1} \text{ i } \nu = \frac{-x_1^3 + ax_1 + 2b}{2y_1}$$

I konačno:

$$P + Q = (\lambda^2 - x_1 - x_2, \lambda^3 + \lambda(x_1 + x_2) - \nu) \quad (34)$$

Na ovaj način su obrađeni osnovni elementi eliptičnih krivih kao i osnovna pravila sabiranja tačaka eliptičnih krivih, odnosno sabiranja u skupu E. Još neka dodatna svojstva će biti obrađena u narednom delu teksta koji je posvećen primeni eliptičnih krivih u kriptografiji. Dodatne osobine eliptičnih krivih će biti obrađivane onim redom kako se bude ukazivala potreba za pojašnjenjima kriptografije eliptičnih krivih.

4. Primena eliptičnih krivih u kriptografiji

Kako je opisano u prethodnom tekstu, eliptične krive su predstavljene nesingularnim polinomima trećeg stepena sa dve promenljive. Takođe je pokazano da se tačke eliptične krive mogu kombinovati putem posebno definisanih načina sabiranja kojim se prevodi grafik eliptične krive u Abelovu grupu. Abelova grupa je grupa elemenata za koje važe komutativni zakoni. Prethodno je pokazano da za sabiranje u skupu E , koji čine tačke eliptične krive, važe zakoni komutacije i asocijacije. Imajući u vidu navedeno, može se zaključiti da u odnosu na operaciju sabiranja, tačke eliptične krive iz skupa E čine Abelovu grupu.

Eliptične krive su našle svoju primenu u rešavanju problema u različitim oblastima, kako u problemima u teoriji brojeva, tako i u problemima algoritama u kriptografiji. Eliptične krive u kriptografiji su pronašle svoju primenu u asimetričnim šifarskim sistemima odnosno sistemima sa javnim ključem ili PKI sistemima [14].

4.1 Generisanje para kriptografskih ključeva

Korišćenjem eliptičnih krivih može se formirati par ključeva, pri čemu će jedan od njih biti tajni ključ, dok će drugi biti javni ključ. Dva generisana ključa se koriste na sličan način, kao i ključevi koji nisu generisani korišćenjem eliptičnih krivih, o čemu će biti reči kasnije u tekstu.

Par ključeva koji se dobija primenom eliptičnih krivih se nalazi u posebnoj relaciji sa skupom domenskih parametara: $D(q, FR, S, a, b, P, n, h)$. Pretpostavimo da postoji konačno polje sa q elemenata nad kojim je definisana eliptična kriva E . To su tačke x ose pravouglog koordinatnog sistema. Označimo to polje sa F_q . Očigledno je da je red tog polja q pošto to polje sadrži q elemenata. Uočimo sada tačku P na eliptičnoj krivoj E . Može se napisati izraz da je $P \in E(F_q)$. Svaka tačka P eliptične krive E se definiše sa dve koordinate x i y . Na taj način smo dobili novi skup, skup svih tačaka eliptične krive E . Ako taj skup označimo sa P možemo uočiti da taj skup čini n tačaka eliptične krive E . Skup P se može smatrati i poljem reda n ili u oznaci P_n . Tako da je red polja P_n u stvari n pošto polje P_n ima n elemenata. Elementi polja P_n nisu brojevi već parovi brojeva koji predstavljaju koordinate tačaka eliptične krive.

Konačana polja su polja sa konačnim brojem elemenata i nazivaju se još i Galoisova polja (Evariste Galois, 1811 – 1832). Red polja je broj elemenata koje polje sadrži i to je prost broj ili stepen prostog broja. Prost broj je ceo broj koji je deljiv isključivo sa samim sobom i brojem jedan.

Prilikom određivanja domenskih parametara potrebno je voditi računa o bezbednosnim ograničenjima, kao i o ograničenjima koja je neophodno poštovati prilikom implementacije rešenja. Obično se domenski parametri razmenjuju između učesnika u komunikaciji, iako u nekim primenama mogu biti karakteristični za pojedinog korisnika.

Domenski parametri $D(q, FR, S, a, b, P, n, h)$ čine sledeći elementi:

1. Red q polja F_q
2. Indikator FR (field representation) se koristi za prikazivanje elemenata polja F_q

3. Slučajni element S (seed) ukoliko je eliptična kriva slučajno definisana korišćenjem određenih algoritama
4. Koeficijenti $a, b \in F_q$ definišu jednačinu eliptične krive E nad poljem F_q
5. Dva elementa $x_p, y_p \in F_q$ koji definišu konačnu tačku $P = (x_p, y_p) \in E(F_q)$. Tačka P je primarnog reda i naziva se baznom tačkom
6. Red n polja P_n
7. Kofaktor $h = \frac{\#E(F_q)}{n}$

Javni ključ predstavlja slučajno izabrana tačka Q iz grupe tačaka P . Odgovarajući privatni ključ se izračunava na sledeći način:

$$d = \log_P Q \quad (35)$$

Učesnik u komunikaciji koji kreira par ključeva mora biti siguran da su domenski parametri validni. Svaki učesnik u komunikaciji koji će koristiti generisani javni ključ mora imati mogućnost verifikacije zavisnosti između javnog ključa i domenskih parametara. U praksi se to može postići na više različitih načina. Moguće je da parametre i ključeve generiše sertifikaciono telo (Certification Authority – CA) koje poštovanjem pravila garantuje pomenutu međuzavisnost između javnog ključa i domenskih parametara. Takođe je moguće da svi učesnici u komunikaciji koriste iste domenske parametre, pri čemu svaki učesnik u komunikaciji može utvrditi međuzavisnost između domenskih parametara i javnog ključa bilo kog od učesnika u komunikaciji.

Algoritam generisanja para ključeva, privatnog i javnog, korišćenjem eliptičnih krivih se može definisati i na sledeći način:

Ulazni parametri su parametri domena: $D(q, FR, S, a, b, P, n, h)$

Izlazni parametri su: javni ključ Q i privatni ili tajni ključ d

1. Izabere se: $d \in_R [1, n - 1]$
2. Izračuna se: $Q = dP$
3. Rezultat je par ključeva (Q, d)

Radi pojašnjenja privatni ključ d se bira iz skupa R tačaka eliptičnih krivih koje zadovoljavaju neophodne uslove slično kao u klasičnoj kriptografiji. Može se primetiti da je problem izračunavanja privatnog ključa d na osnovu javnog ključa Q zapravo problem izračunavanja diskretnog logaritma eliptičnih krivih.

Problem diskretnog logaritma se može posmatrati na sledeći pojednostavljen način [15]:

Za date elemente r i q kao i prost broj p potrebno je pronaći broj k takav da je:

$$r = qk \bmod p \quad (36)$$

Ili na drugi način, ukoliko se uoče dve tačke eliptične krive P i Q potrebno je pronaći broj k takav da je $Pk = Q$. Broj k se naziva diskretnim algoritmom od Q za osnovu P . Veoma je važno da broj d bude zaista slučajan (truly random), odnosno da je verovatnoća izbora nekog broja podjednako mala za svaki od brojeva. Na taj način se otežava potencijalnom napadaču da pogodi broj d na osnovu povećane verovatnoće za izbor posmatranog broja.

4.1.1 Validacija javnog ključa

Validacija javnog ključa ima za cilj da se izvrši provera da li javni ključ poseduje određena aritmetička svojstva. Ukoliko je validacija uspešna time se dokazuje da javni ključ zadovoljava određena svojstva, ali to ne znači da ga neko koristi ili da pripada nekome od učesnika u komunikaciji. Validacija javnog ključa je posebno važna prilikom razmene odnosno kreiranja kriptografskog ključa korišćenjem Diffie-Hellman-ovog algoritma. Ukoliko bi jedan od učesnika u razmeni ključa namerno poslao pogrešan podatak, na taj način bi mogao da sazna privatni ključ drugog učesnika.

Algoritam validacije javnog ključa se može definisati i na sledeći način:

Ulazni parametri su parametri domena: $D(q, FR, S, a, b, P, n, h)$ i javni ključ Q

Izlazni parametri su: prihvatanje ili odbijanje javnog ključa Q

1. Provera da li je $Q \neq \infty$
2. Provera da li x_Q i y_Q zaista predstavljaju element iz polja F_q . Na primer ako je polje F_q reda 2^m tada su x_m i y_m nizovi bitova dužine m .
3. Provera da li tačka Q leži na eliptičnoj krivoj čiji su parametri a i b .
4. Provera da li je $nQ = \infty$
5. Ako je bilo koja provera neuspešna, javni ključ nije verifikovan. Ako su sve provere uspešne tada je javni ključ verifikovan.

Može se primetiti da postoji mnogo brži način provere da li je $nQ = \infty$ nego što je množenje tačke Q sa n . Za eliptične krive koje se najčešće koriste u praksi, dovoljno je potvrditi korake 1, 2 i 3 iz čega se može zaključiti da je $nQ = \infty$.

Na ovaj način je pokazano kako se vrši generisanje ključeva, kao i verifikacija javnog ključa primenom eliptičnih krivih.

4.2 Šema digitalnog potpisa

Digitalni potpis u digitalnom svetu je jednak svojeručnom potpisu u realnom svetu. Za verifikaciju svojeručnog potpisa u realnom svetu zaduženi su grafolozi. Grafolozi su sudski veštaci koji poseduju posebna znanja i veštine pomoću kojih mogu utvrditi da li dva rukopisa potiču od iste osobe ili su ih pisale različite osobe. Potrebno je naglasiti da je verovatnoća tvrdnje veoma visoka, ali nije apsolutna.

U digitalnom svetu se koriste digitalni potpisi koji imaju identičnu pravnu snagu kao i svojeručni potpisi u realnom svetu. Digitalni potpisi se koriste za proveru identiteta pošiljaoca, za proveru integriteta podataka, kao i za neporecivost poslatih podataka. Šeme digitalnog potpisivanja se koriste u sertifikacionim telima za potpisivanje digitalnih sertifikata, pa se na taj način povezuje vlasnik digitalnog sertifikata sa svojim javnim ključem.

Šema digitalnog potpisivanja se sastoji od četiri osnovna algoritma:

1. Algoritam za generisanje domenskih parametara $D(q, FR, S, a, b, P, n, h)$

2. Algoritam za generisanje para ključeva koji kao ulazne parametre uzima parametre domena $D(q, FR, S, a, b, P, n, h)$, dok kao izlazne parametre daje par ključeva Q i d , odnosno javni i tajni ključ.
3. Algoritam za generisanje digitalnog potpisa koji kao ulazne parametre uzima parametre domena D , privatni ključ d i poruku m . Kao izlazni parametar algoritam dobija se digitalni potpis Σ .
4. Algoritam za verifikaciju digitalnog potpisa koji kao ulazne parametre uzima parametre domena D , javni ključ Q , poruku m , kao i pretpostavljeni digitalni potpis Σ . Nakon provere, pretpostavljeni digitalni potpis Σ se potvrđuje ili odbacuje.

Pretpostavićemo da su parametri domena $D(q, FR, S, a, b, P, n, h)$ validni. Takođe ćemo pretpostaviti da je javni ključ Q validan i da je generisan u skladu sa domenskim parametrima D . Verifikacija digitalnog potpisa će uvek prihvatiti ulazne parametre (D, Q, m, Σ) ako je Σ generisan korišćenjem algoritma za generisanje digitalnog potpisa korišćenjem ulaznih parametara (D, d, m) .

4.2.1 Algoritam digitalnog potpisa primenom eliptične krive

U literaturi se označava kao ECDSA (Elliptic Curve Digital Signature Algorithm) što je slično algoritmu DSA (Digital Signature Algorithm) u kriptografiji koja nije bazirana na primeni eliptičnih krivih. ECDSA je najrasprostranjeniji algoritam digitalnog potpisa koji je standardizovan, a koristi eliptične krive. Definisan je u sledećim standardima: ANSI X9.62, FIPS 186-2, IEEE 1363-2000 kao i ISO/IEC 15946-2, dok se nalazi u pripremi za standardizaciju i u još nekim standardima.

U daljim razmatranjima sa H će biti označena kriptografska heš funkcija čija izlazna veličina, heš vrednost, mereno u bitima, nije duža od n . Ukoliko se desi da je dužina heš vrednosti veća od n , tada se uzima u razmatranje samo onoliko bitova heš vrednosti kolika je dužina n .

Algoritam digitalnog potpisa korišćenjem eliptičnih krivih (ECDSA) može se definisati i na sledeći način:

Ulazni parametri su parametri domena: $D(q, FR, S, a, b, P, n, h)$, privatni ključ d i poruka m .

Izlazni parametri su: digitalni potpis (r, s) .

1. Izabere se $k \in_R [1, n - 1]$
2. Izračuna se $kP = (x_1, y_1)$ zatim se x_1 konvertuje u ceo broj $\overline{x_1}$
3. Izračuna se $r = \overline{x_1} \bmod n$. Ako je $r = 0$ ide se ponovo na korak 1
4. Izračuna se $e = H(m)$
5. Izračuna se $s = k^{-1} (e + dr) \bmod n$. Ako je $s = 0$ ide se ponovo na korak 1
6. Konstatuje se konačan rezultat – digitalni potpis (r, s)

4.2.2 Verifikacija digitalnog potpisa ECDSA

Neophodno je da primalac bude siguran da je pošiljalac zaista potpisao poruku odnosno da je digitalni potpis verodostojan. U tom cilju je potrebno izvršiti verifikaciju digitalnog potpisa.

Algoritam verifikacije digitalnog potpisa korišćenjem eliptičnih krivih (ECDSA) može se definisati i na sledeći način:

Ulazni parametri su parametri domena: $D(q, FR, S, a, b, P, n, h)$, javni ključ Q , poruka m i digitalni potpis (r, s) .

Izlazni parametri su: prihvatanje ili odbijanje digitalnog potpisa.

1. Provera da li su r i s celi brojevi u intervalu $[1, N - 1]$. Potrebno je napomenuti da se digitalni potpis proglašava nevažećim i odbacuje u slučaju da bilo koji od uslova koji se proveravaju u algoritmu nije ispunjen.
2. Izračuna se $e = H(m)$
3. Izračuna se $\omega = s^{-1} \bmod n$
4. Izračunavaju se $u_1 = e\omega \bmod n$ i $u_2 = r\omega \bmod n$
5. Izračuna se $X = u_1P + u_2Q$
6. Ako je $X = \infty$ digitalni potpis se odbacuje
7. Pretvaranje x koordinate x_1 od X u ceo broj $\bar{x}_1$ i izračuna se $v = \bar{x}_1 \bmod n$
8. Ako je $v = r$ digitalni potpis se prihvata i smatra se validnim, u suprotnom, digitalni potpis se odbacuje.

4.2.3 Dokaz verifikacije digitalnog potpisa

Ako je potpis (r, s) na poruci m zaista generisao legitimni potpisnik tada je:

$$s \equiv k^{-1}(e + dr) \pmod{n} \quad (37)$$

Dalje se izraz može razviti:

$$s \equiv k^{-1}(e + dr) \equiv s^{-1}e + s^{-1}rd \equiv \omega e + \omega rd \equiv u_1 + u_2d \pmod{n} \quad (38)$$

Tako da se dobija:

$$X = u_1P + u_2Q = (u_1 + u_2d)P = kP \quad (39)$$

Odakle sledi da je $v = r$ što je i zahtevano kao potvrda digitalnog potpisa dobijenog primenom eliptičnih krivih.

4.3 Šifrovanje i dešifrovanje

Šifrovanje javnim ključem se koristi kako bi se obezbedila tajnost poruke. Algoritmi za šifrovanje i dešifrovanje u asimetričnom šifarskom sistemu su znatno sporiji od algoritama u simetričnom šifarskom sistemu, čak i do 1000 puta. Iz navednog razloga, najčešće se koriste za šifrovanje malih količina podataka kao što su brojevi kreditnih kartica, PIN vrednosti (Personal Identification Number), kao i za bezbedno prenošenje kriptografskih ključeva za simetrične šifarske algoritme. Tako preneti ključevi za simetrične šifarske algoritme se koriste za šifrovanje veće količine podataka ili za autentifikaciju poruke.

Šema šifrovanja javnim ključem i dešifrovanje tajnim ili privatnim ključem se sastoji od četiri osnovna algoritma:

1. Algoritam za generisanje domenskih parametara $D(q, FR, S, a, b, P, n, h)$
2. Algoritam za generisanje para ključeva koji kao ulazne parametre uzima parametre domena $D(q, FR, S, a, b, P, n, h)$, dok kao izlazne parametre daje par ključeva Q i d , odnosno javni i tajni ključ.
3. Algoritam za šifrovanje koji kao ulazne parametre uzima domenske parametre D , javni ključ Q i otvoreni tekst poruke m . Kao izlaz, algoritam daje šifrovani tekst odnosno šifrat c .
4. Algoritam za dešifrovanje uzima kao ulazne parametre domenske parametre D , privatni ključ d i šifrat c . Kao rezultat dešifrovanja može se javiti odbacivanje šifrata c kao nekorektnog ili dobijanje otvorenog teksta poruke m .

4.3.1 Algoritam šifrovanja javnim ključem primenom eliptičnih krivih

Šema integrisanog šifrovanja primenom eliptičnih krivih se u literaturi označava kao Elliptic Curve Integrated Encryption Scheme (ECIES) i predložena je od strane Balara (Mihir Bellare, 1992) i Rogaveja (Phillip Rogaway, 1992). Predložena šema predstavlja varijantu el-Gamal-ove šeme šifrovanja javnim ključem. Šema je definisana sledećim standardima: ANSI X9.63 i ISO/IEC 15946-3 dok se nalazi i u nacrtu standarda IEEE P1363a.

U primeni ECIES-a se poruka šifrjuje javnim ključem pošiljaoca, a dešifrjuje tajnim ili privatnim ključem primaoca. Na početku procesa je potrebno definisati dva simetrična ključa k_1 i k_2 . Za tu svrhu se koristi Diffie-Hellman-ov algoritam za generisanje dva simetrična ključa k_1 i k_2 . Procedura se odvija na sledeći način: Ključ k_1 se koristi za šifrovanje otvorenog teksta koristeći simetrični šifarski sistem, dok se k_2 koristi za autentifikaciju šifrovanog teksta. U definisanju algoritma ECIES šifrovanja koristiće se sledeće veličine:

1. KDF predstavlja funkciju za generisanje ključeva koja je izvedena od heš funkcije H .
2. ENC je funkcija za šifrovanje simetričnim algoritmom, dok je DEC funkcija za dešifrovanje, takođe simetričnim algoritmom.
3. MAC je algoritam za autentifikaciju poruke ili Message Authentication Code algorithm.

Algoritam za šifrovanje javnim ključem korišćenjem eliptičnih krivih (ECIES) može se definisati i na sledeći način:

Ulazni parametri su parametri domena: $D(q, FR, S, a, b, P, n, h)$, javni ključ Q i poruka m .

Izlazni parametri su: šifrat (R, C, t) .

1. Izabere se $k \in {}_R[1, n - 1]$
2. Izračuna se $R = kP$ i $Z = hkQ$. Ako je $Z = \infty$ ide se ponovo na korak 1.
3. $(k_1, k_2) \leftarrow KDF(x_Z, R)$ gde je x_Z x koordinata tačke Z
4. Izračuna se $C = ENC_{k_1}(m)$ i $t = MAC_{k_2}(C)$
5. Izlazni parametri (R, C, t)

4.3.2 Algoritam dešifrovanja privatnim ključem primenom eliptičnih krivih

Algoritam za dešifrovanje tajnim ključem korišćenjem eliptičnih krivih (ECIES) može se definisati i na sledeći način:

Ulazni parametri su parametri domena: $D(q, FR, S, a, b, P, n, h)$, privatni ključ d i šifrat (R, C, t)

Izlazni parametri su: otvoreni tekst m ili odbacivanje šifrata.

1. Izvrši se validacija javnog ključa. Ukoliko validacija pokaže da javni ključ nije ispravan šifrat se odbacuje.
2. Izračuna se $Z = hdR$. Ako je $Z = \infty$ tada se šifrat odbacuje
3. $(k_1, k_2) \leftarrow KDF(x_Z, R)$ gde je x_Z x koordinata tačke Z
4. Izračuna se $t' = MAC_{k_2}(C)$. Ako je $t' \neq t$ tada se šifrat odbacuje
5. Izračuna se $m = DEC_{k_1}(C)$
6. Izlazni parametar je otvoreni tekst m .

Dokaz za algoritam dešifrovanja može se pokazati na sledeći način:

Ako je šifrat (R, C, t) zaista generisan od strane legitimnog učesnika u komunikaciji šifrovanjem otvorenog teksta m , tada je:

$$hdR = hd(kP) = hk(dP) = hkQ \quad (40)$$

Na taj način učesnik u komunikaciji koji dešifruje poruku računa iste ključeve (k_1, k_2) kao i učesnik u komunikaciji koji je šifrovao poruku. Tako se prihvata šifrat c i dobija se otvoreni tekst m .

4.4 Uspostavljanje sesijskog ključa

Protokol razmene ključeva ima za cilj da omogući učesnicima u komunikaciji razmenu tajnih ključeva korišćenjem nezaštićene mreže. Tako razmenjen ključ se može koristiti za simetrično šifrovanje ili za autentifikaciju. Osnovna ideja je da se tajni ključ tj. sesijski ključ generiše kod jednog učesnika u komunikaciji, a zatim na bezbedan način prenosi drugim učesnicima u komunikaciji. Na bezbedan način znači da niko neovlašćeno ne može doći u posed tajnog ključa. Algoritam ECIES koji je opisan u prethodnom tekstu se može smatrati transportnim protokolom u kome učestvuju dva učesnika u komunikaciji. Protokol razmene ključeva je u stvari protokol uspostavljanja sesijskog ključa, pri čemu učesnici u komunikaciji međusobno razmenjuju tajne podatke. U narednom delu teksta će biti obrađen protokol za razmenu ključeva koji je izveden iz Diffie-Hellman-ovog protokola za razmenu ključeva.

Protokol za uspostavljanje sesijskog ključa ima dva osnovna cilja:

1. *Posredna autentifikacija ključa.* Pošiljalac poruke je siguran da niko osim primaoca, kome je poruka namenjena, ne može doći u posed kriptografskog ključa. Pri tome, pošiljalac poruke nije siguran da primalac kome je poruka namenjena zaista poseduje kriptografski ključ.
2. *Neposredna autentifikacija ključa.* Ukoliko primalac poruke potvrdi da poseduje ključ pošiljaoca i ukoliko su ispunjeni uslovi iz posredne autentifikacije tada se smatra da je izvršena neposredna autentifikacija kriptografskog ključa.

U narednom delu teksta biće prikazan protokol za uspostavljanje sesijskog ključa primenom eliptičnih krivih.

4.4.1 Protokol za uspostavljanje sesijskog ključa primenom eliptičnih krivih

Ukoliko se vrši uspostavljanje sistema ključeva tj. protokol uspostavljanja sesijskog ključa između dva entiteta, on se u literaturi naziva još i Station-To-Station ili STS protokol. STS protokol se zasniva na problemu diskretnog logaritma i dat je prema standardu ANSI X9.63. U daljem opisu algoritma će se koristiti oznake kao i u opisima prethodno datih algoritama.

U narednom protokolu je cilj da učesnici u komunikaciji A i B uspostave zajednički tajni ključ. Smatra se da je protokol neuspešno izvršen ako barem jedna verifikacija bude neuspešna.

Protokol STS se može prikazati na sledeći način:

Poruke koje se razmenjuju protokolom:

$A \rightarrow B: A, R_A$

$A \leftarrow B: B, R_B, s_B = \text{SIGN}_B(R_B, R_A, A), t_B = \text{MAC}_{k_1}(R_B, R_A, A)$

$A \rightarrow B: s_A = \text{SIGN}_A(R_A, R_B, B), t_A = \text{MAC}_{k_1}(R_A, R_B, B)$

1. A izabere $k_A =_R [1, n - 1]$ zatim izračunava $R_A = k_A P$ i pošalje A, R_A prema B

2. B vrši sledeće operacije:

2.1 Vršiti validaciju ključa R_A

2.2 Bira $k_B = r[1, n - 1]$ a zatim računa $R_B = k_B P$

2.3 Računa $Z = hk_B R_A$ i verifikuje da je $Z \neq \infty$

2.4 $(k_1, k_2) \leftarrow KDF(x_Z)$ gde je x_Z x koordinata tačke Z

2.5 Računa $s_B = SIGN_B(R_B, R_A, A)$, $t_B = MAC_{k_1}(R_B, R_A, A)$

2.6 Šalje: B, R_B, s_B, t_B prema A

3. A izvršava sledeće operacije:

3.1 Vršiti validaciju ključa R_B

3.2 Računa $Z = hk_A R_B$ i verifikuje da je $Z \neq \infty$

3.3 $(k_1, k_2) \leftarrow KDF(x_Z)$ gde je x_Z x koordinata tačke Z

3.4 Verifikuje da je s_B digitalni potpis od B za poruku (R_B, R_A, A)

3.5 Računa $t = MAC_{k_1}(R_B, R_A, A)$ i verifikuje da je $t = t_B$

3.6 Računa $s_A = SIGN_A(R_A, R_B, B)$, $t_A = MAC_{k_1}(R_A, R_B, B)$

3.7 Šalje: s_A, t_A prema B

4. B izvršava sledeće operacije:

4.1 Verifikuje da je s_A potpis od A na poruci (R_A, R_B, B)

4.2 Izračunava $t = MAC_{k_1}(R_A, R_B, B)$ i verifikuje da je $t = t_A$

5. Ključ za sesiju je k_2

Ključ k_2 služi isključivo za šifrovanje dalje komunikacije i naziva se Traffic Encryption Key (TEK). Iz prethodno opisanog algoritma može se zaključiti da je $Z = hk_A k_B P$ što je izvedeno iz javnih ključeva R_A, R_B , predstavlja deljenu tajnu između učesnika u komunikaciji A i B. Množenjem sa h i proverom da li je ispunjen uslov $Z \neq \infty$ uveravamo se da je Z reda n . Uspešna verifikacija digitalnih potpisa $s_A = SIGN_A(R_A, R_B, B)$ i $s_B = SIGN_B(R_B, R_A, A)$ uveravaju učesnike u komunikaciji u identitet sagovornika. Uspešna verifikacija t_A i t_B uverava učesnike u komunikaciji da je svaki od učesnika korektno izračunao deljenu tajnu Z .

U ovom poglavlju su dati osnovni elementi kriptografije eliptičnih krivih. Nije se ulazilo u detaljne opise svakog od parametara, kao ni u precizne dokaze pojedinih izračunavanja, pre svega imajući u vidu potrebni obim i svrhu rada.

5. Analiza napada na kriptografske sisteme eliptičnih krivih

Otkrivanje zaštićenih podataka oduvek je predstavljalo izazov koji je bio motivisan interesom da se zaštićeni podaci otkriju. Otkrivanje zaštićenih podataka je donosilo materijalnu, stratešku ili neku drugu korist organizacijama ili pojedincima. Kriptoanaliza predstavlja oblast koja proučava kriptografski sistem i ima za cilj pronalaženje slabosti samog sistema. Na osnovu slabosti kriptografskog sistema kriptoanalitičari su u manjoj ili većoj meri u stanju da otkriju elemente sistema. Krajnji cilj može predstavljati otkrivanje otvorenog teksta poruke ili otkrivanje elemenata koji omogućavaju napadaču da preuzme identitet učesnika u komunikaciji. Na taj način, napadač može slati i primati poruke u ime nekog drugog i tako ostvariti određenu korist ili prednost. Ovakve aktivnosti se nazivaju još i razbijanjem kriptosistema.

Pre izvođenja samog napada na kriptografski sistem potrebno je izvršiti analizu kritičnih tačaka sistema. Napadi na sisteme se zasnivaju na analizi slabih tačaka i na iskorišćavanju uočenih slabosti. Prilikom napada na kriptografski sistem važi pravilo da se napad izvodi u tačkama u kojima je sistem najranjiviji. Na taj način se uz upotrebu najmanje moguće angažovanih resursa postižu najbolji rezultati napada [18]. Potrebno je imati u vidu da se kriptografija eliptičnih krivih zasniva na problemu diskretnog logaritma eliptičnih krivih. Iz tih razloga se i napadi na kriptografski sistem takođe zasnivaju na problemu diskretnog logaritma eliptičnih krivih. U daljem tekstu će biti obrađen problem diskretnog logaritma eliptičnih krivih dok će nakon toga biti obrađeni tipovi napada koji se na njemu zasnivaju [19].

5.1 Problem diskretnog logaritma u sistemu eliptičnih krivih

Problem rešavanja diskretnog logaritma eliptičnih krivih predstavlja centralno mesto na kome se definiše bezbednost celokupnog sistema. Iz tog razloga, rešavanje problema diskretnog logaritma eliptičnih krivih predstavlja i centralnu tačku napada na sam sistem [20].

Problem diskretnog logaritma eliptičnih krivih (Elliptic Curve Discrete Logarithm Problem - ECDLP) se može definisati na sledeći način [21]:

Ako je data eliptična kriva E koja je definisana nad konačnim poljem F_q gde je tačka $P \in E(F_q)$ reda n i tačka $Q \in \langle P \rangle$, znači da je tačka Q izabrana tačka kao jedna od tačaka eliptične krive P . Potrebno je pronaći ceo broj l takav da je $l \in [0, n - 1]$ takav da je zadovoljeno $Q = lP$. Ceo broj l se zove diskretni logaritam od Q za osnovu P . Ili u oznaci: $l = \log_P Q$.

Parametri eliptične krive moraju biti pažljivo izabrani kako bi bili otporni na poznate napade na ECDLP.

Najjednostavniji tip napada na ECDLP je primena rednog pretraživanja odnosno isprobavanja vrednosti za l i to tako što l uzima vrednosti: 1,2,3,4 ... itd. pri čemu se izračunava: $P, 2P, 3P, 4P \dots$ itd. sve dok se ne dobije vrednost Q . Napad rednim pretraživanjem se naziva još i *brute force attack*.

Najnepovoljniji slučaj je ukoliko je l najveće moguće odnosno jednako redu n . Tada će biti potrebno n koraka da se dođe do rešenja. U proseku će biti potrebno dvostruko manje koraka, odnosno $n/2$ koraka. Kako bi se smanjila uspešnost *brute force* napada potrebno je izabrati n toliko veliko da nije izvodljivo, odnosno da je teško izvodljivo i izračunati l sa postojećim računarskim resursima i u prihvatljivom vremenu. Na primer, dobro je izabrati n tako da je $n \geq 2^{80}$.

Za napad opšte namene na ECDLP najefikasnija je kombinacija Polig-Helmanovog (Stephen Pohlig, 1963, Martin Hellman, 1945) algoritma i Polar ro algoritma (John Pollard, 1941).

5.1.1 Polig-Helmanov algoritam napada

Polig-Helmanov algoritam (Stephen Pohlig) [22] čini izračunavanje logaritma $l = \log_p Q$ znatno efikasnijim. Pretpostavimo da se broj n može rastaviti na r prostih činioca p_i koji su stepenovani na neki stepen e_i . Opisana operacija se naziva još i faktorizacija broja koja se može predstaviti sledećim matematičkim izrazom:

$$n = p_1^{e_1} p_2^{e_2} \dots p_r^{e_r} \quad (41)$$

Polig-Helmanov algoritam izračunava:

$$l_i = l \bmod p_i^{e_i} \quad (42)$$

Pri čemu je $1 \leq i \leq r$. Na taj način se dobija sistem kongruencija ili sistem kongruentnih jednačina:

$$\begin{aligned} l &= l_1 \bmod p_1^{e_1} \\ l &= l_2 \bmod p_2^{e_2} \\ &\dots \\ l &= l_r \bmod p_r^{e_r} \end{aligned} \quad (43)$$

za $l \in [0, n - 1]$. Dalje se rešava prikazani sistem pri čemu se kao krajnji rezultat dobija traženo l_i . Radi jednostavnijeg pisanja u sledećim izrazima će umesto p_i biti pisano p , a umesto e_i će biti pisano e . Traženo l_i će biti izračunato na sledeći način:

$$l_i = z_0 + z_1 p + z_2 p^2 + \dots + z_{e-1} p^{e-1} \quad (44)$$

Pri čemu $z \in [0, p - 1]$. Vrednosti $z_0, z_1, \dots, z_{e-1}$ se izračunavaju redom, jedna po jedna vrednost. Najpre se izračunavaju:

$$P_0 = \left(\frac{n}{p}\right) P \quad (45)$$

i

$$Q_0 = \left(\frac{n}{p}\right) Q \quad (46)$$

Dalje se vrši izračunavanje:

$$Q_0 = \frac{n}{p} Q = \frac{n}{p} lP = l \left(\frac{n}{p} P \right) = lP_0 = z_0 P_0 \quad (47)$$

Odakle se izračunava:

$$z_0 = \log_{P_0} Q_0 \quad (48)$$

Na isti način se izračunavaju i preostale vrednosti $z_1, z_2, \dots, z_{e-1}$

5.1.2 Polar ro algoritam napada

Osnovna ideja kod Polar ro (John Pollard, 1941) napada [23] je da postoje različiti parovi celih brojeva po modulu n (c', d') i (c'', d'') takvih da zadovoljavaju sledeći uslov:

$$c'P + d'Q = c''P + d''Q \quad (49)$$

Dalje je:

$$(c' - c'')P = (d'' - d')Q = (d'' - d')lP \quad (50)$$

a zatim:

$$(c' - c'') = (d'' - d')l \pmod{n} \quad (51)$$

Pošto je $l = \log_P Q$, l se može izračunati na sledeći način:

$$l = (c' - c'')(d'' - d')^{-1} \pmod{n} \quad (52)$$

Pojednostavljeni ili kako se u literaturi naziva i „naivni“ metod Polar ro algoritma napada zasniva se na sledećem: za pronalaženje parova (c', d') i (c'', d'') biraju se dva slučajna broja $c, d \in [0, n-1]$, a zatim se izračunava $cP + dQ$. Ta tri elementa $c, d, cP + dQ$ se smeštaju u posebnu tabelu. Izračunavanje se ponavlja sve dok se ponovo ne dobije ista vrednost za $cP + dQ$. Takav događaj se naziva kolizijom.

Polar ro algoritam napada koji se primenjuje zahteva gotovo jednako vreme za izvršavanje kao i „naivni“ metod, ali zato zahteva mnogo manje memorijskih kapaciteta. Ova činjenica je veoma korisna kada se radi o specifičnim hardverskim uređajima sa malim memorijskim kapacitetima, kao što su različite vrste mobilnih uređaja, senzora i drugih specifičnih uređaja.

Ako sa X označimo $X = cP + dQ$ i ako označimo funkciju $H(X) = j$ gde je j index odgovarajućeg para celih brojeva (c, d), Polar ro algoritam napada na ECDLP se može prikazati na sledeći način:

Ulazni parametri su: $P \in E(F_q)$ reda n .

Izlazni parametri su: diskretni algoritam $l = \log_P Q$.

1. Izabere se broj L kao stepen broja 2

2. Izabere se funkcija H u intervalu $\{1, 2, \dots, L\}$
3. Za j koje se menja od 1 do L vrše se sledeća izračunavanja:
 - 3.1 Izabere se $a_j, b_j \in_R [0, n - 1]$
 - 3.2 Izračuna se $R_j = a_j P + b_j Q$
4. Izabere se $c', d' \in_R [0, n - 1]$ i izračuna se $X' = c'P + d'Q$
5. Dodele se vrednosti: $X'' \leftarrow X', c'' \leftarrow c', d'' \leftarrow d'$
6. Ponavlja se sledeće izračunavanje sve dok ne bude ispunjen uslov da je $X' = X''$
 - 6.1 Izračunava se $j = H(X')$
Dodeljuju se vrednosti: $X' \leftarrow X' + R_j, c' = c' + a_j \bmod n, d' \leftarrow d' + b_j \bmod n$
 - 6.2 Za vrednost i od 1 do 2 vrše se sledeća izračunavanja:
Izračunava se $j = H(X'')$
Dodeljuju se vrednosti $X'' \leftarrow X'' + R_j, c'' \leftarrow c'' + a_j \bmod n, d'' \leftarrow d'' + b_j \bmod n$
- Kada je zadovoljen uslov da je $X' = X''$ izračunavanja se završavaju
7. Ako je $d' = d''$ algoritam vraća neuspeh kao rezultat
U suprotnom se izračunava $l = (c' - c'')(d'' - d')^{-1} \bmod n$ i kao izlaz iz algoritma se vraća veličina l .

Na ovaj način su prikazani neki od mogućih napada na rešavanje problema diskretnog logaritma koji čini osnovni postulat bezbednosti kriptografije eliptičnih krivih.

5.2 Napadi na kriptografske ključeve

U kriptanalizi se napad na kriptografske ključeve smatra najefikasnijim. Kriptografski algoritmi koji se koriste su opšte poznati, matematički dokazani i višestruko testirani. Validna testiranja kriptografskih algoritama je propisao američki institut NIST (National Institute of Standards and Technology) u vidu FIPS (Federal Information Processing Standard) standarda koji definišu upravljanje informacijama [24]. Takođe u Sjedinjenim Američkim Državama postoji specijalizovana agencija za bezbednost NSA (National Security Agency) [25] koja je u izvesnom smislu nadređeni organ NIST-u. NSA kategoriše i odobrava za upotrebu svaki od kriptografskih algoritama. Takođe NSA eventualno odobrava davanje na upotrebu kriptografskih algoritama drugim državama, ukoliko proceni da na taj način ne bi bili ugroženi nacionalni interesi Sjedinjenih Američkih Država.

Iz prethodno navedenog se može zaključiti da su kriptografski algoritmi standardizovani, provereni i veoma bezbedni. Prilikom napada na sistem treba se voditi logikom da se napad vrši na najslabije tačke sistema. Ukoliko se radi o kriptografiji, element koji je svakako slabiji od kriptografskog algoritma je kriptografski ključ. Iz tog razloga su napadi na kriptografske ključeve, odnosno njihovo kompromitovanje, daleko efikasniji nego napadi na kriptografske algoritme.

Posebna oblast kriptografije se bavi generisanjem, upravljanjem i distribucijom kriptografskih ključeva i ima za cilj da obezbedi najvišu moguću bezbednost kriptografskih ključeva.

U narednom delu teksta biće opisani neki od poznatih napada na kriptografske ključeve u kriptografiji eliptičnih krivih.

5.2.1 Napadi malih podgrupa

U prethodnom delu teksta je dat detaljan opis načina na koji se vrši validacija javnog ključa u kriptografiji eliptičnih krivih. Napad na malu podgrupu se zasniva na pretpostavci da će učesnik u komunikaciji napraviti propust prilikom validacije javnog ključa sagovornika i da neće izvršiti svaki korak u algoritmu validacije javnog ključa [26]. Pretpostavimo da je A učesnik u komunikaciji čiji je par ključeva (Q, d) , a koji su u vezi sa domenskim parametrima $D(q, FR, S, a, b, P, n, h)$. Prilikom izvršavanja Difi-Helmanovog (ECDH) protokola, drugi učesnik u komunikaciji, učesnik B, vrši autentifikaciju ključa Q . Učesnik B bira $r \in_R [1, n-1]$ i šalje učesniku u komunikaciji A izračunatu vrednost $R = rP$. Nakon toga, učesnik u komunikaciji A računa tačku $K = dR$, dok učesnik u komunikaciji B istovremeno izračunava istu tačku $K = rQ$. Na taj način oba učesnika, A i B izračunavaju deljeni tajni ključ $k = KDF(K)$, gde je KDF (Key Derivation Function) posebna funkcija koja pomoću vrednosti tačke K izračunava ključ k . Može se primetiti da opisani postupak obezbeđuje jednosmernu autentifikaciju učesnika A učesniku B, što je sasvim prihvatljivo i dovoljno u slučaju primene SSL (Secure Sockets Layer) protokola. Radi podsećanja, prilikom primene SSL protokola, server autentifikuje korisnika, ali korisnik ne autentifikuje server. Pretpostavka je da učesnici u komunikaciji A i B koriste ključ k kako bi autentifikovali međusobne poruke korišćenjem MAC (Message Authentication Code) algoritma.

Pretpostavimo da učesnik A, prilikom validacije javnog ključa, propusti da izvrši proveru da je $nQ = \infty$. Neka je sada broj l prost delilac kofaktora h . Prilikom izvođenja napada malih podgrupa učesnik B šalje učesniku A tačku R reda l , umesto tačke P reda n . Tada učesnik A izračunava $K = dR$ kao i $k = KDF(K)$. Pošto je R reda l , zato će i K biti reda l osim ako nije ispunjen uslov da je $d = 0 \bmod l$. Ako je ispunjen uslov $d = 0 \bmod l$ tada će biti $K = \infty$. Prema tome je $K = d_l R$ gde je $d_l = d \bmod l$. Sada učesnik A šalje poruku m kao i svoj autentifikacioni kod $t = MAC_k(m)$ učesniku B. Sada učesnik B redom bira $l' \in [0, l-1]$ sve dok ne dobije da je $t = MAC_{k'}(m)$ gde je $k' = KDF(K')$ i gde je $K' = l'R$. Tada je sa visokom verovatnoćom $d_l = l'$. Očekivani broj pokušaja koje treba da preduzme učesnik B kako bi došao do rezultata je $l/2$. Učesnik B može da ponavlja napad sa različitim tačkama R u paru sa prostim brojevima $l_1, l_2, \dots, l_s$. Ako je kofaktor h relativno velik tada učesnik B dobija značajne informacije o privatnom ključu d učesnika A. Ukoliko koristi redno pretraživanje, učesnik B može dobiti i kompletan tajni ključ d učesnika A.

U praktičnim primenama se za vrednost kofaktora h obično uzimaju male vrednosti, kao na primer: $h = 1, 2$ ili 4 . Takav izbor čini napad malih podgrupa manje efikasnim i njime se može otkriti samo manji broj bitova privatnog ključa d korisnika A.

Veoma jednostavan način odbrane od opisanog napada je da se validacija javnog ključa sagovornika u komunikaciji izvrši po svim tačkama algoritma.

5.2.2 Napad primenom lažnih eliptičnih krivih

Pre razmatranja napada primenom lažnih eliptičnih krivih treba se podsetiti da je eliptična kriva definisana izrazom (16) u poglavlju 3.2 i da je način sabiranja dve tačke eliptične krive, definisan kao u poglavlju 3.2.5. Osnovno zapažanje, na kome počiva napad primenom lažnih eliptičnih krivih, je to da se prilikom sabiranja tačaka eliptične krive E koja je definisana nad poljem F_q , ne uključuje koeficijent b u izračunavanju zbira dve tačke.

Pretpostavimo da postoji druga eliptična kriva E' koja je takođe definisan nad poljem F_q [27]. Eliptična kriva E' se u svojoj Vajerštrasovoj normalnoj formi (16) u poglavlju 3.2 razlikuje jedino u koeficijentu b . Tada su zakoni sabiranja tačaka eliptičnih krivih E i E' potpuno identični jer, kao što je pomenuto, koeficijent b ne učestvuje u izračunavanjima zbira dve tačke, definisano izrazom (26) u poglavlju 3.2.5. Sada se eliptična kriva E' naziva lažnom eliptičnom krivom u odnosu na eliptičnu krivu E .

Pretpostavimo sada da učesnik u komunikaciji A ne sprovodi validaciju tačke koju dobija primenom ECDH protokola. Učesnik u komunikaciji B, koji je u ovom slučaju napadač, bira lažnu krivu E' takvu da eliptična kriva $E'(F_q)$ sadrži tačku R reda l . Sada napadač B šalje tačku R učesniku u komunikaciji A. Učesnik u komunikaciji A izračunava $K = dR$ kao i $k = KDF(R)$. Kao i prilikom napada malih podgrupa učesnik u komunikaciji A šalje učesniku u komunikaciji B, u ovom slučaju napadaču, poruku m kao i svoj autentifikacioni kod $t = MAC_k(m)$. Napadač B izračunava $d_l = d \bmod l$. Napad se ponavlja sa drugim tačkama R ili sa drugim lažnim eliptičnim krivima. Na ovaj način postoji mogućnost da napadač B otkrije privatni ključ d učesnika u komunikaciji A.

Postoji veoma jednostavan način odbrane od opisanog napada, a to je provera da tačka koja je primljena od sagovornika u komunikaciji zaista pripada legitimnoj eliptičnoj krivoj.

Na osnovu dva do sada opisana napada, očigledno je da se svaki od napada oslanja na neopreznost učesnika u komunikaciji kao i na njihovu sklonost da ne sprovedu određene korake u algoritmima, bilo iz razloga što žele što pre da pošalju ili prime poruku, bilo iz razloga što se osećaju zaštićenima više nego što je to zaista slučaj ili su jednostavno lenji da sprovedu sve predviđene korake u proceduri.

Iz tih razloga svaki napad kriptanalitičara započinje pažljivim posmatranjem učesnika u komunikaciji i otkrivanjem njihovih loših navika i slabosti.

5.2.3 Napad prilikom ustanovljavanja ključa

Pretpostavimo da postoje dva učesnika u komunikaciji A i B koji komuniciraju korišćenjem otvorene mreže. Za zaštitu same komunikacije se koristi kriptografski ključ koji se ustanovljava na takav način da učesnici u komunikaciji mogu bezbedno ustanoviti kriptografski ključ kojim će štititi komunikaciju, a pri čemu napadač koji ima pristup otvorenom komunikacijskom kanalu ne može doći u posed tog ključa. U tu svrhu je razvijen poseban protokol za uspostavljanje zajedničkog ključa za zaštitu komunikacije putem otvorene mreže [28]. Zajednički ustanovljeni ključ se može koristiti kao kriptografski ključ u simetričnim kriptografskim sistemima kako bi se ostvarili ciljevi poverljivosti, kao i zaštite integriteta podataka.

Protokol ustanovljavanja transportnog ključa je protokol u kome jedan učesnik u komunikaciji kreira tajni ključ i prenosi ga drugom učesniku u komunikaciji na bezbedan način. Protokol ECIES koji je opisan u poglavlju 4.3.1 se može smatrati transportnim protokolom za razmenu ključa između dva učesnika u komunikaciji pri čemu otvorena poruka predstavlja sam kriptografski ključ. Protokol za ustanovljavanje ključa podrazumeva da svaki od učesnika u komunikaciji generiše određene informacije na osnovu kojih se izračunava zajednički tajni ključ. Za razmenu i generisanje kriptografskog ključa koristi se Difi-Helmanov algoritam kao i različite varijacije ovog algoritma.

U idealnom slučaju, protokol razmene kriptografskog ključa kao rezultat ima uspostavljenje tajnog ključa. Proces se može uporediti sa situacijom kada bi se dva učesnika u komunikaciji međusobno poznavala i kada bi se susreli na tajnom mestu, pri čemu bi odredili zajednički ključ na potpuno slučajan način. Ključ koji će se čuvati i koristiti nakon generisanja ni u kom pogledu ne bi smeo biti kompromitovan, jer bi se na taj način dovela u pitanje bezbednost generisanja samog ključa, odnosno ne bi imala nikakvog značaja. Veoma je teško formalno definisati šta podrazumeva bezbedna razmena kriptografskog ključa. Zato će u daljem tekstu biti navedeni poželjni bezbednosni zahtevi koje bi protokol za razmenu i generisanje kriptografskog ključa trebalo da zadovolji kako bi se mogućnost napada na sam protokol svela na minimum.

Bezbedan protokol bi trebalo da bude u stanju da izdrži kako pasivne tako i aktivne napade putem kojih napadač pokušava da onemogući izvršavanje protokola ili pokušava da sazna kriptografski ključ. Prilikom pasivnih napada napadač pokušava da sazna kriptografski ključ samo praćenjem komunikacije i razmenjenih podataka. Prilikom aktivnih napada, napadač prati komunikaciju i izvršava algoritam. Tom prilikom napadač vrši i brisanje, menjanje ili ubacivanje svojih poruka u komunikaciju.

Kako bi se ograničila količina podataka koje kriptanalitičar može da vidi, korisno je da prilikom svake razmene podataka, podaci budu šifrovani drugim ključem koji je unapred definisan. Ključevi koji služe za ovu namenu nazivaju se sesijski ključevi. Sesijski ključevi su drugačiji za svaku novu sesiju [29].

Kako bi se umanjila mogućnost kriptanalitičkog napada prilikom uspostavljanja i razmene kriptografskog ključa, potrebno je obezbediti sledeće:

1. Implicitna autentifikacija ključa

Podrazumeva mehanizam koji omogućava učesniku u komunikaciji A da bude uveren da niko drugi osim učesnika u komunikaciji B ne može doći do informacija o razmenjenom ključu. Na ovaj način, učesnik A ne zna da li je učesnik B zaista i došao u posed kriptografskog ključa.

2. Eksplicitna autentifikacija ključa

Prilikom eksplicitne autentifikacije ključa, učesnik u komunikaciji A je siguran da je učesnik B izračunao kriptografski ključ i da ga zaista poseduje. Podrazumeva se da je uslov naveden u implicitnoj autentifikaciji ispunjen.

Kako bi se otpornost na kriptanalitičke napade prilikom razmene ključa dodatno unapredila, bilo bi korisno da su ispunjeni i sledeći uslovi:

1. Kompromitovanje jednog od ključeva ne znači da su i ključevi ostalih učesnika kompromitovani, što znači da nije naneta šteta drugim komponentama algoritma.
2. Ukoliko je kompromitovan ključ učesnika A, to znači da se napadač može predstavljati drugim učesnicima u komunikaciji kao A. U tom slučaju bi bilo korisno da se napadač ne može predstavljati učesniku A kao neko od drugih legalnih učesnika u komunikaciji.
3. Potreban je mehanizam koji uverava učesnika A da razmenjuje ključ sa učesnikom B. Ukoliko napadač pokuša da se predstavi kao učesnik B, potrebno je da učesnik A dobije informaciju o tome i prekine komunikaciju.

U prethodnom tekstu su obrađeni neki od mogućih napada na kriptografski sistem eliptičnih krivih. Detaljno obrađeni napadi se mogu pronaći u literaturi. Treba imati u vidu da postoje i vrste napada koje nisu javno objavljene, ali se mogu primeniti u realnim napadima.

6. Oblasti primene kriptografije eliptičnih krivih

Vizuelizaciju o tome koliko je teško razbiti algoritme koji su bazirani na kriptografiji eliptičnih krivih u odnosu na standardne RSA i DH algoritme, dao je Arjen K. Lenstra (Arjen Klaas Lenstra, 1956) [30]. Količinu energije potrebne za razbijanje kriptografskog algoritma, uporedio je sa time koju količinu vode ta ista energija može da dovede do tačke ključanja. Ovim merenjima, došao je do poređenja da razbijanje RSA ključa od 228 bita, zahteva manje energije nego što je to potrebno da proključa jedna kafena kašičica vode. Dok razbijanje ključa algoritma baziranog na eliptičnim krivima od 228 bita zahteva dovoljno energije da proključa sva voda koja se nalazi na našoj planeti. Da bi RSA algoritam dostigao ovaj nivo sigurnosti, morao bi da koristi ključ od 2380 bita.

Algoritmi bazirani na eliptičnim krivima omogućavaju da se uspostavi isti nivo bezbednosti sa značajno manjim kriptografskim ključem, ali pored toga i uz korišćenje znatno manje procesorske snage i potrošnje energije. Iz navednog razloga algoritmi bazirani na eliptičnim krivima počinju da se koriste sve češće u razvoju mobilnih aplikacija. Koristeći RSA kriptografski algoritam može se održati isti nivo zaštite uz upotrebu značajno većih ključeva, međutim to istovremeno rezultuje znatno sporijim performansama.

Danas, sve veći broj sajtova primenjuje kriptografiju eliptičnih krivih kako bi zaštitili privatnost na svim poljima, od klijentove HTTPS konekcije do načina kako se razmenjuju podaci između centara za obradu podataka.

Dokaz da kriptografija eliptičnih krivih dobija sve više na popularnosti i postaje sve rasprostranjenija, daje njena upotreba u mnogim aplikacijama: vlada Sjedinjenih Američkih Država koristi ovu vrstu zaštite kada je u pitanju komunikacija unutar njihovog sistema, Tor projekt je koristi u osiguravanju anonimnosti putem Interneta. Takođe, to je mehanizam uz pomoć koga se dokazuje vlasništvo nad najpoznatijom elektronskom valutom – *bitcoin*. Servis kompanije Apple pod nazivom iMessage koristi ECC u svrhu generisanja digitalnih potpisa. Eliptične krive se koriste i u svrhu šifrovanja DNS informacija sa DNSCurve protokola, a predstavljaju i preporučenu metodu za autentifikaciju prilikom sigurnog pretraživanje interneta putem SSL/TLS protokola.

Ukoliko pristupate HTTPS verziji Cloudflare bloga iz neke od novijih verzija pretraživača Chrome-a ili Firefox-a, vaš pretraživač će koristiti ECC. U Chrome pretraživaču možete videti koji kriptografski algoritmi se koriste za uspostavljanje sigurne konekcije klikom na ikonicu katanca, koji se nalazi odmah pored adrese sajta koji se posećuje, a zatim klikom na connection tab. U slučaju Cloudflare bloga, klikom na pomenuto dugme, dobiće se informacija da pretraživač koristi ECDHE_RSA algoritam (Elliptic Curve Diffie Hellman Ephemeral). ECDHE predstavlja mehanizam za razmenu ključeva koji se zasniva na eliptičnim krivama. Navedeni algoritam se koristi za osiguravanje opcije koja se naziva „perfect forward secrecy“ u okviru SSL protokola [31]. Pomenuti termin označava da se u okviru svake konekcije kreira novi javni ključ, što nije slučaj ukoliko se za šifrovanje i dešifrovanje podataka između klijenta (web pretraživača) i servera koristi RSA algoritam. Prilikom korišćenja RSA algoritma jedan javni ključ se koristi mesecima ili godinama i na taj način se može ugroziti čitava prošla i buduća konekcija. U slučaju da se privatni ključ SSL servera ukrade ili na bilo koji drugi način dođe u posed neovlašćenog lica, čitava komunikacija koju je neko mogao snimiti se može i dešifrovati. U slučaju korišćenja ECDHE

algorima za svaku novu konekciju generiše se novi par ključeva, što znači da bi napadač morao da razbije ključ za svaku pojedinačnu konekciju ukoliko želi da sazna koji su se podaci u njoj razmenjivali. Takođe, generisani ključ je kratkotrajan (eng. *ephemeral*) što znači da ga server briše nakon svake završene konekcije i ne postoji mogućnost da bilo ko dođe u njegov posed i dešifruje prethodnu konekciju [32].

ECDHE_RSA označava da se u trenutnoj konekciji koristi RSA algoritam za utvrđivanje identiteta servera. Web sajtovi koji koriste RSA, koriste ga iz razloga što je njihov SSL sertifikat vezan za par RSA ključeva. Moderniji pretraživači takođe podržavaju sertifikate bazirane ne eliptičnim krivima. Ukoliko sajt koristi SSL sertifikat koji je zapravo sertifikat baziran na eliptičnim krivama, umesto ECDHE_RSA dobili bismo ECDHE_ECDSA. Ovo označava da bi se za utvrđivanje identiteta servera koristio ESDSA (Elliptic Curve Digital Signature Algorithm).

Razlika u performansama ukoliko se koristi ECDSA algoritam u odnosu na RSA algoritma je drastična. Čak i sa starijim verzijama OpenSSL protokola kod kojih kod nije optimizovan za eliptične krive, ECDSA digitalni potpis sa ključem od 256 bita je preko 20 puta brži nego RSA digitalni potpis od 2048 bita.

Neki nedostaci eliptičnih krivih

Svaki od algoritama za šifrovanje, pa tako i algoritmi bazirani na eliptičnim krivama, pored svojih dobrih imaju i loše strane. Određeni nedostaci sprečavaju da algoritam bude široko prihvaćen u praksi.

Jedna od osetljivih stavki je generator brojeva koje smatramo slučajnim, a koji koristi matematiku eliptičnih krivih. Generator pod nazivom Dual Elliptic Curve Deterministic Random Bit Generator (Dual_EC_DRBG) je standardizovan od strane NIST-a. Međutim, nakon početka njegove upotrebe, prijavljeno je da postoji mogućnost postojanja *backdoor*-a u samom generatoru. To bi značilo da bi neko sa određenim tajnim brojem bio u mogućnosti da potpuno predvidi brojeve koji se korisniku predstavljaju kao slučajni. Nedavno je kompanija pod nazivom RSA povukla iz prodaje više svojih proizvoda koji su kao automatski generator slučajnih brojeva imali podešen Dual_EC_DRBG generator. Bilo da unutar samog generatora postoji *backdoor* ili ne, to ne menja snagu algoritama kriptografije eliptičnih krivih, već postavlja pitanja vezana za sam proces standardizacije eliptičnih krivih. To je takođe jedan od razloga zbog koga je bitno osigurati da sistem koristi prave slučajne brojeve.

Neki od kriptografa imaju generalno nepoverenje kada se radi o samom NIST institutu i standardima koji su podržani od strane NSA (National Security Agency). Skoro sve eliptične krive koje su trenutno u širokoj upotrebi spadaju u ovu kategoriju. Za ove specifične krive, ne postoje poznati napadi, ali je činjenica da postoje i loše eliptične krive, pa se iz tog razloga, mnogi ne usuđuju da primene eliptične krive u svojim rešenjima. Ostvaren je napredak u razvoju eliptičnih krivih sa efikasnom aritmetikom koja nije povezana sa NIST institutom. Međutim, do usvajanja novih standarda eliptičnih krivih i njihove konkretne primene ostalo je još nekoliko godina.

Postoji još jedan problem vezan za ECC a podrazumeva patente. Postoji više od 130 patenata koji pokrivaju specifične upotrebe eliptičnih krivih i u vlasništvu su BlackBerry kompanije. Većina ovih patenata je licencirana za upotrebu od strane privatnih organizacija,

čak i u odnosu na NSA, barem formalno. Ukoliko se programeri odluče za implementaciju algoritama baziranih na eliptičnim krivama, moraju prvo dobro da prouče da li njihova implementacija narušava neki od postojećih patenata, što predstavlja dodatan posao. Godine 2007. Certicom kompanija je tužila Sony kompaniju zbog nekih upotreba eliptičnih krivih, međutim ta tužba je odbačena 2009. godine.

Algoritam za digitalni potpis ECDSA opisan u poglavlju 4.2.1 ima slabu stranu u odnosu na tradicionalni RSA algoritam, a to je da zahteva kvalitetan izvor entropije. Bez adekvatnih slučajnih brojeva, postoji mogućnost za otkrivanje privatnog ključa. Mana u random generatoru na Androidu omogućila je hakerima da otkriju ECDSA privatni ključ koji je korišćen za zaštitu Bitcoin online novčanika više ljudi 2013. godine. Sony PlayStation implementacija algoritma ECDSA je imala slične ranjivosti.

Zaključak je da je kvalitetan izvor slučajnih brojeva neophodan za pravilno i sigurno funkcionisanje algoritama. Iz tog razloga, Dual_EC_DRBG generator, nije preporučen.

Prva generacija kriptografskih asimetričnih algoritama, kao što su RSA i Diffie-Hellman-ov način razmene ključeva, još uvek predstavljaju standard u mnogim slučajevima. Međutim, ECC sa svim svojim prednostima veoma brzo postaje sve prikladnije rešenje za privatnost i bezbednost na mreži.

7. Zaključak

Eliptične krive predstavljaju jednu od familija algebarskih krivih. Primenu eliptičnih krivih u kriptografiji prvi su predložili Nil Koblitz (Neal Koblitz, 1948) i Viktor Miler (Victor Saul Miller, 1947) 1985. godine. Kriptografija eliptičnih krivih (Elliptic curve cryptography - ECC) se primenjuje u kriptografskim sistemima koji koriste asimetrične algoritme, odnosno u PKI sistemima kao što su RSA, El Gamal i drugi slični algoritmi. U asimetričnim sistemima svaki korisnik poseduje dva ključa, javni i tajni, pri čemu se javni ključ koristi za šifrovanje, dok se privatni ključ koristi za dešifrovanje i digitalno potpisivanje. Veoma je važno da tajni ključ ostane poznat samo vlasniku. Ukoliko bi došlo do kompromitovanja tajnog ključa istovremeno bi došlo i do kompromitovanja celokupnog šifarskog sistema. Iz tog razloga se posebna pažnja posvećuje čuvanju i zaštiti privatnih kriptografskih ključeva. Privatni kriptografski ključevi se najčešće čuvaju na različitim tokenima, kao što su kartice sa mikrokontrolerom, USB tokeni, kartice mobilnih uređaja i drugo. Imajući u vidu da se računarski kapaciteti neprekidno usavršavaju i da se njihove mogućnosti značajno povećavaju, za kompromitovanje određenog šifarskog sistema potrebno je sve manje i manje vremena. Kao primer se može uzeti DES šifarski sistem za čije razbijanje su nekada bile potrebne godine, dok se danas korišćenjem savremenih računarskih resursa kao i primenom paralelnog procesuiranja vreme za razbijanje DES algoritma meri satima.

Istraživači u oblasti kriptografije, koja se prvenstveno zasniva na matematici i matematičkim modelima, čine napore radi unapređenja šifarskih sistema. Cilj je da se očuva bezbednost šifarskih sistema i obezbedi pouzdana zaštita podataka, uprkos napretku računarske tehnologije i uvećanju računarskih resursa. Značajni prodori su napravljeni u šifarskim sistemima sa javnim ključevima i to pre svega uvođenjem kriptografije eliptičnih krivih, a posebno uvođenjem kvantne kriptografije kao posebne oblasti. Kriptografija eliptičnih krivih nalazi sve veću primenu u različitim oblastima gde se zahteva pouzdana kriptografska zaštita, a pri čemu se raspolaze ograničenim memorijskim i procesorskim resursim. Eliptične krive, kao posebna familija algebarskih krivih su posebno obrađene u matematici. Pojedine definicije i dokazi iz oblasti eliptičnih krivih precizno definišu sve osobine eliptičnih krivih, ali zbog obimnosti samih definicija i dokaza, oni nisu detaljno prikazani u ovom radu. Detaljni dokazi se mogu naći u matematičkoj literaturi koja obrađuje eliptične krive. Osobine eliptičnih krivih se koriste kao dodatni mehanizmi za poboljšanje postojećih asimetričnih šifarskih sistema. Svaki od sistema sa javnim ključem ima neku osnovnu matematičku operaciju na kojoj zasniva izračunavanje kriptografskih parametara. Na primer, kod RSA algoritma osnovna matematička operacija je stepenovanje. U ECC sistemima je osnovna operacija sabiranje tačaka eliptične krive, odnosno množenje koje se može prikazati kao višestruko sabiranje. U radu su detaljno dati opisi i definicije sabiranja tačaka eliptičnih krivih, odnosno elementa skupa E kome tačke eliptične krive pripadaju.

ECC sistemi su prvenstveno našli svoju primenu u sistemima i uređajima koji imaju ograničene memorijske i procesorse resurse. Korišćenje ECC sistema je zastupljeno u bežičnim komunikacionim uređajima, karticama sa čipom, kao i na Web serverima koji istovremeno izvršavaju mnogo sesija od kojih svaka od sesija zahteva određenu količinu memorije i određeno procesorsko vreme. Posebnu primenu su našli u mobilnim uređajima kao što su pametni mobilni telefoni koji su široko rasprostranjeni i čije se funkcionalnosti neprekidno proširuju. [16] Šifarski sistemi eliptičnih krivih se koriste i u aplikacijama kojima je neophodna bezbednost za njihovo funkcionisanje, ali nemaju mogućnost čuvanja tajnih podataka kao ni mogućnost korišćenja dovoljno procesorskih kapaciteta koliko bi to zahtevali drugi šifarski sistemi. ECC se koriste u slučajevima gde se manipuliše kraćim setovima

podataka kao što su slučajevi razmene ključeva, digitalno potpisivanje, autentifikacija i slično.

Napadi na šifarske sisteme kao svoj osnovni cilj postavljaju kompromitovanje šifarskog sistema. Napadač pokušava da, bez poznavanja osnovnih parametara kriptografskog sistema, otkrije sadržinu šifrovane poruke odnosno da dođe u posed otvorenog teksta. Oblast kriptologije koja se bavi razbijanjem šifarskih sistema naziva se kriptanaliza.

Postoji uverenje da svaki kreator kvalitetnog algoritma mora biti dobar kriptanalitičar. Analizom tuđih algoritama kriptanalitičari stiču iskustva koja su neophodna za razvoj sopstvenih algoritama. Uočavanjem nedostataka u kriptografskim sistemima stiče se saznanje o tome šta bi trebalo promeniti u novom šifarskom sistemu koji bi bio bolji od prethodnih. Pre posmatranja i ocene samog kriptografskog sistema potrebno je imati uvid u prethodne aktivnosti kreatora sistema. Ukoliko se radi o osobi koja je u svojoj praksi pokazala da je u stanju da razbije neki kriptografski sistem, može se opravdano smatrati da je sistem koji je ta osoba kreirala pouzdan. Naravno, novi sistem mora biti podvrgnut rigoroznom kriptanalitičkom testiranju, kao i bilo koji drugi sistem. Ukoliko je autor sistema neko ko se bavio samo površnim proučavanjem kriptografije, kriptografskom sistemu toga autora treba pristupiti sa rezervom.

U radu su razmotreni primeri mogućih kriptanalitičkih napada na kriptografski sistem eliptičnih krivih. Razmotreni su mogući napadi rešavanjem problema diskretnog logaritma u sistemu eliptičnih krivih. Pomenuti problem čini jedan od osnovnih postulata bezbednosti kriptografskog sistema eliptičnih krivih. Uspešan napad na problem diskretnog logaritma u sistemu eliptičnih krivih bi značio kompromitovanje čitavog sistema. Pored toga razmotreni su i Polig-Helmanov algoritam kao i Polar ro algoritam napada. Na kraju su obrađeni i mogući napadi na kriptografske ključeve.

Osnovni zaključak koji se nameće prilikom analize napada na kriptografske sisteme uopšte, pa i na kriptografske sisteme eliptičnih krivih, je taj da uspeh napada zavisi pre svega od slabosti samog sistema koji je izložen napadu. Kao očigledan primer je navedeno preskakanje određenih koraka u verifikaciji kriptografskog ključa što otvara vrata napadaču.

Najvažnija preporuka za obezbeđivanje pune bezbednosti sistema je poštovanje svih bezbednosnih standarda, procedura i postupaka koji su propisani za izvršavanje u kriptografskom sistemu eliptičnih krivih.

Prednosti i nedostaci kriptografskih sistema eliptičnih krivih su sledeći:

Prednosti kriptografskih sistema eliptičnih krivih

- Kraće dužine ključa za postizanje istog stepena bezbednosti. Minimalna dužina ključa u ECC sistemima je 132 bita dok je za RSA algoritam 952 bita [17].
- Kraći šifrat i digitalni potpisi
- Brže šifrovanje, dešifrovanje, digitalno potpisivanje kao i verifikacija digitalnog potpisa
- Odlični protokoli za razmenu ključeva
- Zahtevanje manje računarskih resursa u odnosu na druge šifarske sisteme
- Veća fleksibilnost sistema i mogućnost prilagođavanja različitim uređajima

Nedostaci kriptografskih sistema eliptičnih krivih

- Komplikovana i složena implementacija
- Postoje hipereliptični sistemi koji nude još kraće ključeve od ECC
- Kraći kriptografski ključevi su podložniji kriptanalitičkim napadima
- Novi algoritmi mogu imati nepoznate slabosti koje će se manifestovati tek u toku rasprostranjenije primene.

Kriptografija eliptičnih krivih predstavlja značajan napredak u kriptografiji. U periodu koji sledi očekuje se da ECC sistem pokaže sve svoje kvalitete.

Literatura

- [1] Zvanični sajt instituta SANS iz Berlina,
<https://www.sans.org/reading-room/whitepapers/vpns/history-encryption-730>
- [2] Istorijski sajt „Quattro passi nella storia“ („Četiri koraka istorije“)
<http://www.quattropassinellastoria.it/>
- [3] Department of the army, Field manual No 34-40-2 Headquarters, *Basic Cryptanalysis*, University of Michigan,
<http://www.umich.edu/~umich/fm-34-40-2/#ps>Postscript files.<HR>
- [4] C. C. Palmer, *Ethical hacking*, IBM Systems Journal, vol 40, No 3, Armonk, USA, 2001.
- [5] Christopher Hadnagy, *Social Engineering: The Art of HumanHacking*, WileyPublishing, Inc., Indianapolis, USA, 2011.
- [6] M.Veinović, S.Adamović, *Kriptologija I*, ISBN: 978-86-7912-469-2, Univerzitet Singidunum, Beograd, 2013.
- [7] The University of Texas at Austin Department of Computer Science (UTCS), Foundations of Computer Security
<https://www.cs.utexas.edu/users/byoung/cs361/lecture44.pdf>
- [8] C.E. Shannon, *Communication Theory of Secrecy Systems*, University California Los Angeles, USA, 1949
- [9] Linkopings University, Department of Electrical Engineering, Jan-Ake Larsson
<http://www.icg.isy.liu.se/courses/tsit03/forelasningar/cryptolecture06.pdf>
- [10] Johannes Buchmann, *The Digital Signature Algorithm (DSA)*, Technische Universitat Darmstadt, Fachbereich Informatik, Deutschland, 2001
- [11] J. Silverman, *An Introduction to the Theory of Elliptic Curves*, Brown University and NTRU Cryptosystems, Inc., Summer School on Computational Number Theory and Applications to Cryptography University of Wyoming, USA, 2006
- [12] <http://www.lesbillgates.com/Apollonius.html>
- [13] Wolfram MathWorld, the web's most extensive mathematics resource
<http://mathworld.wolfram.com/EllipticCurve.html>
- [14] D.Hankerson, A.Menezes, S.Vanstone, *Guide to Elliptic Curve Cryptography*, Springer-Verlag, New York, Inc. ISBN 0-387-95273-X, New York, USA, 2004.
- [15] Sajt fime Certicom: <https://www.certicom.com/index.php/52-the-elliptic-curve-discrete-logarithm-problem>
- [16] S.Tillich, J.Grossch, *A Survey of Public-Key Cryptography on J2ME-Enabled Mobile Devices*, Graz University of Technology, Institute for Applied Information Processing and Communications, Springer-Verlag Berlin Heidelberg, Gratz, Austria, 2004

-
- [17] Ernst Kani, *The State of the Art of Elliptic Curve Cryptography*, Department of Mathematics and Statistics, Queen's University, Kingston, Ontario, Canada, 2003
- [18] Jon Erickson, *Hacking: The art of exploitation*, No starch press, San Francisco, USA, 2008.
- [19] Tim Guneyasu, Christof Paar, Jan Pelzl, *On the security of elliptic curve cryptosystems against attacks with special-purpose hardware*, Horst Gortz Institute for IT Security, Ruhr University Bochum, Germany. 2006.
- [20] Gretel S. Sia, *A brief survey of the discrete logarithm problem*, University of Hawaii at Manoa, USA, 2011.
- [21] Kristin E. Lauter, Katherine E. Stange, *The elliptic curve discrete logarithm problem and equivalent hard problems for elliptic divisibility sequences*, Microsoft Research, Redmond, Department of Mathematics, Harvard University, Cambridge, USA, 2008.
- [22] Martin Lysoe Sommerseth, Haakon Hoeiland, *Defending against Pohlig-Hellman attacks*, Norges teknisk-naturvitenskapelige universitet, Alesund, Gjøvik, Norway, 2015.
- [23] Nagaratna Hegde, P.Deepthi, *Pollard rho algorithm for integer factorization and discrete logarithm problem*, International Journal of Computer Applications (0975 – 8887) Volume 121 – No.18, Hyderabad, India, 2015.
- [24] <http://www.nist.gov/>
- [25] <https://www.nsa.gov/>
- [26] Junfeng Fan, Benedikt Gierlichs, Frederik Vercauteren, *To Infinity and Beyond: Combined Attack on ECC using Points of Low Order*, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium, 2011.
- [27] Marc Joye, *Fault Attacks on Elliptic Curve Cryptosystems*, Thomson Security Labs, Atlanta, USA, 2009.
- [28] Dave Singelee, Bart Preneel, *Key Establishment Using Secure Distance Bounding Protocols*, Katholieke Universiteit Leuven, Leuven-Heverlee, Belgium, 2008.
- [29] Jean-Sebastien Coron, *Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems*, Ecole Normale Supérieure, France, 2007.
- [30] <https://people.epfl.ch/arjen.lenstra>
- [31] <http://arstechnica.com/security/2013/10/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/>
- [32] <https://blog.cloudflare.com/staying-on-top-of-tls-attacks/>

Dodatak 1

```
// KriptografijaElipticnihKrivih_byKristina.cpp
/* ----- */
* Namena fajla: Primer primene elipticnih krivih
* nad elementima konacnog polja
* Program je radjen u C++ programskom jeziku
* U okruzenju MS Visual Studio 2010
* Program ima za cilj da pokaze primenu postupaka i procedura
* koje su teorijski i matematički definisane
* u Master radu
* Programsko rešenje je dato u dva odvojena fajla
* jedan od njih je ovaj: Eliptične_krive.cpp
* dok je drugi: Konacna_polja.h u kome su definisane
* matematicke funkcije koje se više puta pozivaju
* Programski kod predstavlja praktičnu dopunu Master rada
* kao i njegov sastavni deo
* u programskom kodu su korisceni delovi koda
* koji se mogu pronaci u slobodnoj formi na Internetu,
* a koje su razvijali razliciti autori,
* na razlicitim programerskim forumima kao sto su:
* http://www.codeproject.com/
* http://stackoverflow.com/
* http://cboard.cprogramming.com/
* http://www.bloodshed.net/
* http://forums.devshed.com/
* http://forums.codeguru.com/
* i drugi.
* ----- */

/* ----- */
* Prekompajlirani heder, dodaje se automatski prilikom
* kreiranja C++ projekta u okruzenju MS Visual Studio 2010 IDE
* (Integrated Development Environment)
* ----- */

#include "stdafx.h"

// Standardne biblioteke koje se ukljucuju u program
#include <cstdlib>
#include <iostream>
#include <vector>
#include <math.h>

// Standardni namespace koji se ukljucuje u program
using namespace std;

#include "ElementiKonacnihPolja_byKristina.hpp"

namespace KriptografijaEK
{
/* ----- */
* Definise se elipticna kriva nad konacnim poljem reda P
* u Vajerstrasovom obliku:  $E = x^3 + ax + b$ 
* prilikom izracunavanja se koriste obicni celi brojevi
* tipa int
* u realnim uslovima bi trebalo koristiti cele brojeve
* tipa long int ili cak long long int
* Parametar P predstavlja red konacnog polja Fp
* nad kojim je elipticna kriva definisana
* ----- */
}
```

```

template < int P >
class ElipticnaKriva
{
    public:
/* ----- */
* Elipticna kriva je definisana nad konacnim poljem          *
* (Galoisovo polje) Fp                                     *
* u daljem kodu se definisu elementi elipticne krive       *
* ----- */

    typedef ElementKonacnogPolja < P > ekp_t;

/* ----- */
* Tacka elipticne krive kao element                        *
* sastoji se od dva elementa konacnog polja                *
* u daljem kodu se definisu elementi elipticne krive      *
* koordinate tacke moraju da zadovolje jednačinu          *
* elipticne krive                                           *
* ----- */

    // Klasa tacka elipticne krive
    class Tacka
    {
        friend class ElipticnaKriva < P >;
        typedef ElementKonacnogPolja < P > ekp_t;
        ekp_t x_;
        ekp_t y_;
        ElipticnaKriva *ek_;
    };

/* ----- */
* Algoritam za mnozenje dvostruke vrednosti elemenata     *
* elementom m                                              *
* ----- */

    void dodajDvostruko ( int m, Tacka& adresa_tacke )
    {
        if ( m > 0 )
        {
            Tacka r = adresa_tacke;
            for ( int n = 0; n < m; ++n )
            {
                r += r; // dvostruka vrednost
            }
            adresa_tacke = r;
        }
    }

/* ----- */
* Algoritam za mnozenje dvostruke vrednosti elemenata     *
* skalarom k sa opcijom cuvanja medjurezultata            *
* izmedju binarnog i skalarnog oblika                     *
* ----- */

    Tacka skalarnoMnozenje ( int k, const Tacka& a )
    {
        Tacka adresa_tacke = a;
        Tacka rezultat = Tacka ( 0, 0, *ek_ );
        int i = 0, j = 0;
        int b = k;

        while( b )
        {
            if ( b & 1 )

```

```

        {
            // prolazi se kroz skup bitova
            dodajDvostruko( i-j, adresa_tacke );
            rezultat += adresa_tacke;
            j = i; // poslednji bit iz skupa
        }
        b >>= 1;
        ++i;
    }

    return rezultat;
}

// Sabiranje dve tacke elipticne krive
void saberi_dve_tacke ( ekp_t x1, ekp_t y1, ekp_t x2, ekp_t y2,
                        ekp_t & xR, ekp_t & yR ) const
{
    // specijalni slucajevi sabiranja dve tacke
    if ( x1 == 0 && y1 == 0 )
    {
        xR = x2;
        yR = y2;
        return;
    }

    if ( x2 == 0 && y2 == 0 )
    {
        xR = x1;
        yR = y1;
        return;
    }

    if ( y1 == -y2 )
    {
        xR = yR = 0;
        return;
    }

    // Opsti slucaj
    ekp_t s;

    if ( x1 == x2 && y1 == y2 )
    {
        // 2 * P
        s = ( 3 * (x1.i() * x1.i() ) + ek_ -> a() ) / ( 2 * y1 );
        xR = ( (s * s) - 2 * x1 );
    }
    else
    {
        // P + Q
        s = ( y1 - y2 ) / ( x1 - x2 );
        xR = ( (s * s) - x1 - x2 );
    }

    if ( s != 0 )
    {
        yR = ( -y1 + s*(x1 - xR) );
    }
    else
    {
        xR = yR = 0;
    }
}

```

```

    }
}

// Konstruktori klase Tacka
Tacka ( int x, int y )
    : x_(x),
      y_(y),
      ek_(0)
{}

Tacka ( int x, int y, ElipticnaKriva < P > & ElipticnaKriva )
    : x_(x),
      y_(y),
      ek_( &ElipticnaKriva )
{}

Tacka(const ekp_t& x, const ekp_t& y, ElipticnaKriva < P > & ElipticnaKriva)
    : x_(x),
      y_(y),
      ek_( &ElipticnaKriva )
{}

public:
static Tacka JEDAN;

// Copy constructor
Tacka( const Tacka& tck )
{
    x_ = tck.x_;
    y_ = tck.y_;
    ek_ = tck.ek_;
}

// Preklapanje operatora
Tacka& operator = ( const Tacka& tck )
{
    x_ = tck.x_;
    y_ = tck.y_;
    ek_ = tck.ek_;
    return *this;
}

// Vraca vrednost komponente x tacke kao elementa konacnog polja Fp
ekp_t x() const { return x_; }

// Vraca vrednost komponente y tacke kao elementa konacnog polja Fp
ekp_t y() const { return y_; }

/* ----- */
* Naredni deo koda vrsi izracunavanje
* reda elementa klase Tacka
* Izvršavanje narednog dela koda može biti veoma sporo
* u zavisnosti od parametara elipticne krive koji se izaberu
* u ovom primeru su izabrani parametri koji ne zahtevaju
* mnogo vremena i koji služe samo u svrhe testiranja.
* U realnim uslovima se može desiti da se
* izvršavanje koda i zaglavi
* u zavisnosti od procesora i RAM memorije
* računara na kome se program izvršava
* U tom slučaju bi trebalo drugacije podesiti promenljivu
* maxPeriod
* ----- */

```

```

    unsigned int Order( unsigned int maxPeriod = ~0 ) const
    {
        Tacka r = *this;
        unsigned int n = 0;
        while( r.x_ != 0 && r.y_ != 0 )
        {
            ++n;
            r += *this;
            if ( n > maxPeriod ) break;
        }
        return n;
    }

/* ----- */
* Preklapanje operatora za klasu Tacka
* ----- */

// negacija
Tacka operator - ( )
{
    return Tacka ( x_, -y_ );
}

// ==
friend bool operator == ( const Tacka& tck_izr, const Tacka& tck )
{
    return ( tck_izr.ek_ == tck.ek_ ) && ( tck_izr.x_ == tck.x_ ) &&
           ( tck_izr.y_ == tck.y_ );
}

// !=
friend bool operator != ( const Tacka& tck_izr, const Tacka& tck )
{
    return (tck_izr.ek_ != tck.ek_ ) || ( tck_izr.x_ != tck.x_ ) ||
           ( tck_izr.y_ != tck.y_ );
}

// a + b
friend Tacka operator + ( const Tacka& tck_izr, const Tacka& tck )
{
    ekp_t xR, yR;
    tck_izr.saberi_dve_tacke ( tck_izr.x_, tck_izr.y_, tck.x_, tck.y_,
    xR, yR );
    return Tacka (xR, yR, *tck_izr.ek_ );
}

// a * int
friend Tacka operator * (int k, const Tacka& tck )
{
    return Tacka ( tck ).operator *= ( k );
}

// +=
Tacka& operator += ( const Tacka& tck )
{
    saberi_dve_tacke (x_, y_, tck.x_, tck.y_, x_, y_ );
    return *this;
}

// a *= int
Tacka& operator *= ( int k )
{

```

```

        return (*this = skalarnoMnozenje ( k, *this ) );
    }

    // ostream handler: prikazuje koordinate elementa klase Tacka
    friend ostream& operator << ( ostream& os, const Tacka& p )
    {
        return ( os << "(" << p.x_ << ", " << p.y_ << ")" );
    }
}; // Kraj klase Tacka

/* ----- */
/* Primena elipticnih krivih u kriptografiji */
/* ----- */
typedef ElipticnaKriva < P > this_t;
typedef class ElipticnaKriva < P > :: Tacka Tacka_t;

/* ----- */
/* Konstruktor */
/* Inicijalizacije elipticne krive u Vajerstrasovom obliku */
/*  $y^2 = x^3 + ax + b$  */
/* ----- */
ElipticnaKriva( int a, int b )
: a_(a),
  b_(b),
  m_tabela_(),
  tabela_je_popunjena_ (false)
{}

/* ----- */
/* Izracunavanje tacaka elipticne krive */
/* Inicijalizacije u Vajerstrasovom obliku */
/*  $y^2 = x^3 + ax + b$  */
/* Ako je red polja veliki */
/* izvršavanje programa može trajati malo duže */
/* ----- */
void IzracunavanjeTacaka ()
{
    int x_vrednost [P];
    int y_vrednost [P];
    for ( int n = 0; n < P; ++n )
    {
        int n_na_kvadrat = n*n;
        x_vrednost [n] = ( (n * n_na_kvadrat ) + a_.i() * n +
                           b_.i() ) % P;
        y_vrednost [n] = n_na_kvadrat % P;
    }

    for ( int n = 0; n < P; ++n )
    {
        for ( int m = 0; m < P; ++m )
        {
            if ( x_vrednost[n] == y_vrednost[m] )
            {
                m_tabela_.push_back ( Tacka ( n, m, *this ) );
            }
        }
    }

    tabela_je_popunjena_ = true;
}

```

```

// Izdvajanje elementa klase Tacka na elipticnoj krivoj
Tacka operator [] (int n)
{
    if ( !tabela_je_popunjena_ )
    {
        IzracunavanjeTacka ();
    }

    return m_tabela_ [n];
}

// Broj elemenata u posmatranoj grupi
size_t Size() const { return m_tabela_.size(); }

// Red P posmatrane elipticne krive
int Degree() const { return P; }

// Parametar a kao element polja Fp
ElementKonacnogPolja < P > a() const { return a_; }

// Parametar b kao element polja Fp
ElementKonacnogPolja < P > b() const { return b_; }

// ostream handler: Prikazuje krivu u citljivom obliku
template < int T >
friend ostream& operator << ( ostream& os, const ElipticnaKriva< T >&
                             ElipticnaKriva )

// Stampa sve elemente iz grupe elipticne krive
ostream& StampajTabelu ( ostream &os, int columns = 4 );

private:
typedef std :: vector < Tacka > m_tabela_t;
m_tabela_t m_tabela_; // Tabela sa tackama
ElementKonacnogPolja < P > a_; // Paramtar a elipticne krive
ElementKonacnogPolja < P > b_; // Paramtar b elipticne krive
bool tabela_je_popunjena_ ; // true ako je tabela popunjena
}; // Kraj klase ElipticnaKriva

// Template kojim definisemo tacku elipticne krive
template < int T >
typename ElipticnaKriva < T > :: Tacka ElipticnaKriva < T > :: Tacka ::
    JEDAN ( 0, 0 );

// Definisanje stampanja Vajestrasove forme eliticne krive
template < int T >
ostream& operator << ( ostream& os, const ElipticnaKriva < T >& ElipticnaKriva )
{
    os << "y^2 mod " << T << " = ( x^3 " << showpos;
    if ( ElipticnaKriva.a_ != 0 )
    {
        os << ElipticnaKriva.a_ << "x ";
    }

    if ( ElipticnaKriva.b_.i() != 0 )
    {
        os << ElipticnaKriva.b_;
    }

    os << noshowpos << " ) mod " << T;
    return os;
}

```

```

    }

    // Definisanje stampanja tacaka elipticne krive
    template < int P >
    ostream& ElipticnaKriva < P > :: StampajTabelu ( ostream &os, int columns )
    {
        if ( tabela_je_popunjena_ )
        {
            int col = 0;
            typename ElipticnaKriva < P > :: m_tabela_t ::
                iterator iter = m_tabela_.begin();
            for ( ; iter != m_tabela_.end(); ++iter )
            {
                os << "( " << (*iter).x_.i() << ", " << (*iter).y_.i() << " ) ";
                if ( ++col > columns )
                {
                    os << "\n";
                    col = 0;
                }
            }
        }
        else
        {
            os << "ElipticnaKriva, F_" << P;
        }
        return os;
    }
} // Kraj namespace KriptografijaEK

// Deo koda koji generise slucajne brojeve
namespace SlucajniBrojevi
{
    float frand ()
    {
        static float norm = 1.0f / (float) RAND_MAX;
        return (float)rand() * norm;
    }

    int irand ( int min, int max )
    {
        return min + (int)( frand() * (float)( max - min ) );
    }
} // Kraj namespace SlucajniBrojevi

// Program koji izvrsava primer i poziva napisani kod
using namespace KriptografijaEK;
using namespace SlucajniBrojevi;

int main( int argc, char *argv[] )
{
    typedef ElipticnaKriva < 263 > ek_t; // Red polja je 263
    ek_t mojaElipticnaKriva(1,1); // Definisanje promenljive za tacku elipticne krive

    cout << "Primer koriscenja\nKriptografije elipticnih krivih\n\n";

    // Stampanje podataka o elipticnoj krivoj
    cout << "Elipticna kriva: " << mojaElipticnaKriva << "\n";
}

```

```

/* ----- */
* Izracunavanje tacaka elipticne krive
* Ova izracunavanja mogu trajati veoma dugo
* pogotovo ako se uzimaju vrednosti koje se koriste
* u realnim uslovima.
* U ovom primeru se izracunava relativno mali broj tacaka
* ----- */
mojaElipticnaKriva.IzracunavanjeTacaka ();

cout << "\nTacke krive su:\n";
mojaElipticnaKriva.StampajTabelu ( cout, 5 );
cout << "\n\n";

ek_t :: Tacka P = mojaElipticnaKriva [2];
cout << "Za neku tacku elipticne krive P = " << P << ", 2 * P = " << ( P + P )
    << "\n\n";

ek_t :: Tacka Q = mojaElipticnaKriva [3];
cout << "Za neku tacku elipticne krive Q = " << Q << ", P + Q = " << ( P + Q )
    << "\n\n";

ek_t::Tacka R = P;
R += Q;
cout << "P += Q = " << R << "\n\n";
R = P;
R += R;
cout << "P += P = 2 * P = " << R << "\n\n\n";

cout << "Primer sifrovanja poruke primenom elipticnih krivih\n";
cout << "=====\n\n";

// Izbor slucajne tacke elipticne krive
ek_t :: Tacka G = mojaElipticnaKriva [0];
while( ( G.y() == 0 || G.x() == 0 ) || ( G.Order() < 2 ) )
{
    int n = (int)(frand() * mojaElipticnaKriva.Size() );
    G = mojaElipticnaKriva [n];
}

cout << "Izabrana je tacka G = " << G << ", reda " << G.Order() << "\n\n";

// Ucesnik u komunikaciji A
int a = irand ( 1, mojaElipticnaKriva.Degree () - 1 );
ek_t :: Tacka Pa = a * G; // Javni kljuc od A
cout << "Javni kljuc ucesnika u komunikaciji A je Pa = " << a << " * " << G
    << " = " << Pa << "\n\n";

// Ucesnik u komunikaciji B
int b = irand ( 1, mojaElipticnaKriva.Degree () - 1 );
ek_t :: Tacka Pb = b * G; // Javni kljuc od B
cout << "Javni kljuc ucesnika u komunikaciji B je Pb = " << b << " * " << G
    << " = " << Pb << "\n\n";

// Napadac koji prisluskuje komunikaciju
int n = irand (1, mojaElipticnaKriva.Degree () - 1 );
ek_t :: Tacka Pn = n * G;
cout << "Javni kljuc napadaca je Pn = " << n << "*" << G << " = " << Pn << "\n\n";

cout << "\n\n";

```

```

/* ----- */
* Ucesnik u komunikaciji A sifruje svoju poruku
* i salje je ucesniku u komunikaciji B
* Poruku je neophodno prethodno podeliti na delove
* koji ce biti elementi Galoisovog polja Fp
* koje definise domen elipticne krive
* U primeru koji je dat P je malo tako da nije pogodno
* za koriscenje u realnoj kriptografiji
* U realnim uslovima trebalo bi izabrati znatno vece P
* ----- */

int m1 = 19;
int m2 = 72;

cout << "Poruka koju ucesnik A salje ucesniku B: (" << m1 << ", " << m2 << ")\n\n";

// Sifrovanje koriscenjem kljuc a ucesnika B
ek_t :: Tacka Pk = a * Pb;
ek_t :: ekp_t c1 ( m1 * Pk.x() );
ek_t :: ekp_t c2 ( m2 * Pk.y() );

// Sifrovana poruka je: Pa, c1, c2
cout << "Sifrovana poruka koju A salje B je {Pa,c1,c2} = {" << Pa << ", "
    << c1 << ", " << c2 << "}\n\n";

/* ----- */
* Ucesnik u komunikaciji B desifruje poruku
* koju je dobio od A koristeći javni kljuc A
* ----- */

Pk = b * Pa;
ek_t :: ekp_t m1d = c1 / Pk.x();
ek_t :: ekp_t m2d = c2 / Pk.y();

cout << "Ucesnik B desifruje poruku dobijenu od A a to je (" << m1d << ", "
    << m2d << ")" << "\n\n";

/* ----- */
* Napadac presrece poruku i pokusava da je desifruje
* koristeći javni kljuc ucesnika A
* ----- */

Pk = n * Pa;
m1d = c1 / Pk.x();
m2d = c2 / Pk.y();

cout << "Napadac pokusava da desifruje poruku\n";
cout << "koju je A poslao B i dobija (" << m1d << ", " << m2d << ")" << "\n";
cout << "Sto ocigledno nije prava poruka\n\n";

system("PAUSE");

return 0;
} // Kraj main funkcije

```

Dodatak 2

```
// ElementiKonacnihPolja_byKristina.hpp

// Standardna biblioteka koja se ukljucuju u program
#include <math.h>

// Standardni namespace koji se ukljucuje u program
using namespace std;

namespace KriptografijaEK
{
    /* ----- *
    * Pomocne funkcije *
    * ----- */

    namespace Pomocne
    {
        /* ----- *
        * Prosireni Euklidov algoritam za izracunavanje *
        * najveceg zajednickog delioca (NZD) *
        * koji rezultat daje u obliku  $g = a * u + b * v$  *
        * ----- */
        int NZD ( int a, int b, int& u, int &v )
        {
            u = 1;
            v = 0;
            int g = a;
            int u1 = 0;
            int v1 = 1;
            int g1 = b;
            while (g1 != 0)
            {
                int q = g / g1; // Deljenje celih brojeva
                int t1 = u - q * u1;
                int t2 = v - q * v1;
                int t3 = g - q * g1;
                u = u1;
                v = v1;
                g = g1;
                u1 = t1;
                v1 = t2;
                g1 = t3;
            }

            return g;
        }
    }

    /* ----- *
    * Resavanje linearne kongruentne jednacine oblika *
    *  $x * z == 1 \pmod{n}$  *
    * za dato x *
    * ----- */
    int LKJ(int x, int n)
    {
        n = std::abs(n);
        x = x % n; // Ako je  $x \Rightarrow 0$  tada je  $x \% n < |n|$ 
        int u, v, g, z;
        g = NZD ( x, n, u, v );
        if ( g != 1 )
        {

```

```

/* ----- */
* x i n moraju biti uzajamno prosti brojevi
* kako bi postojalo jedinstveno resenje
*  $x^{-1} \bmod n$ 
* ----- */
    z = 0;
}
else
{
    z = u % n;
}

return z;
}
} // Kraj namespace Pomocne

/* ----- */
* Definisane klase elemenata konacnog polja
* i definisanje izracunavanja koja se vrse
* nad objektima klase
* u slucaju da  $(-n) \bmod m$  daje negativnu vrednost
* Promenljiva P je red polja
* ----- */
template < int P >

class ElementKonacnogPolja
{
    int i_;

    void assign ( int i )
    {
        i_ = i;
        if ( i < 0 )
        {
/* ----- */
* Obrada slucaja kada je  $i < 0$  (i predstavlja indeks tacke)
* zelimo da budemo sigurni da ce biti u datom domenu
* elipticne krive
* ----- */
            i_ = ( i % P ) + 2 * P;
        }

        i_ %= P;
    }

public:

    // Konstruktor
    ElementKonacnogPolja ()
    : i_(0)
    {}

    // Konstruktor
    explicit ElementKonacnogPolja ( int i )
    {
        assign ( i );
    }

    // Copy constructor
    ElementKonacnogPolja (const ElementKonacnogPolja < P >& tck)
    : i_(tck.i_)
    {}

```

```

/* ----- *
 * Preklapanje operatora *
 * ----- */

// Adresa objekta
int i() const { return i_; }

// Negacija
ElementKonacnogPolja operator - () const
{
    return ElementKonacnogPolja ( -i_ );
}

// Dodeljivanje celobrojne vrednosti
ElementKonacnogPolja& operator = ( int i )
{
    assign ( i );
    return *this;
}

// Dodeljivanje elementa polja
ElementKonacnogPolja<P>& operator = ( const ElementKonacnogPolja<P>& tck )
{
    i_ = tck.i_;
    return *this;
}

// *=
ElementKonacnogPolja<P>& operator *= ( const ElementKonacnogPolja<P>& tck )
{
    i_ = ( i_*tck.i_ ) % P;
    return *this;
}

// ==
friend bool operator == ( const ElementKonacnogPolja<P>& tck_izr,
                           const ElementKonacnogPolja<P>& tck )
{
    return ( tck_izr.i_ == tck.i_ );
}

// == int
friend bool operator == ( const ElementKonacnogPolja<P>& tck_izr, int tck )
{
    return (tck_izr.i_ == tck);
}

// !=
friend bool operator != ( const ElementKonacnogPolja<P>& tck_izr, int tck )
{
    return ( tck_izr.i_ != tck );
}

// a / b
friend ElementKonacnogPolja<P> operator / (const ElementKonacnogPolja<P>&
      tck_izr, const ElementKonacnogPolja<P>& tck )
{
    return ElementKonacnogPolja<P>( tck_izr.i_ * Pomocne ::
      LKJ( tck.i_, P ) );
}

// a + b
friend ElementKonacnogPolja<P> operator + ( const ElementKonacnogPolja<P>&
      tck_izr, const ElementKonacnogPolja<P>& tck )
{
    return ElementKonacnogPolja<P>( tck_izr.i_ + tck.i_);
}

```

```

    }

    // a - b
    friend ElementKonacnogPolja<P> operator - ( const ElementKonacnogPolja<P>&
        tck_izr, const ElementKonacnogPolja<P>& tck )
    {
        return ElementKonacnogPolja<P>( tck_izr.i_ - tck.i_);
    }

    // a + int
    friend ElementKonacnogPolja<P> operator + ( const ElementKonacnogPolja<P>&
        tck_izr, int i )
    {
        return ElementKonacnogPolja<P>( tck_izr.i_ + i );
    }

    // int + a
    friend ElementKonacnogPolja<P> operator + ( int i,
        const ElementKonacnogPolja<P>& tck )
    {
        return ElementKonacnogPolja<P>( tck.i_ + i );
    }

    // int * a
    friend ElementKonacnogPolja<P> operator * ( int n,
        const ElementKonacnogPolja<P>& tck )
    {
        return ElementKonacnogPolja<P>( n*tck.i_);
    }

    // a * b
    friend ElementKonacnogPolja<P> operator * ( const ElementKonacnogPolja<P>&
        tck_izr, const ElementKonacnogPolja<P>& tck )
    {
        return ElementKonacnogPolja<P>( tck_izr.i_ * tck.i_);
    }

    // ostream handler
    template <int T>
    friend ostream& operator << ( ostream& os,
        const ElementKonacnogPolja<T>& g )
    {
        return os << g.i_;
    }
}; // Kraj klase ElementKonacnogPolja
} // Kraj namespace KriptografijaEK

```