UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

Marija Bogavac

Digitalni potpis u kriptovalutama: poređenje ECDSA i EdDSA

master rad

Beograd, 2026.

Mentor:

dr Dragan Đokić, docent
Univerzitet u Beogradu, Matematički fakultet

Članovi komisije:

dr Goran Đanković, redovni profesor
Univerzitet u Beogradu, Matematički fakultet

dr Sana Stojanović Đurđević, docent
Univerzitet u Beogradu, Matematički fakultet

Datum odbrane: 25. septembar 2026.

Sadržaj

1 Uvod

1.1 Motivacija

Digitalni potpis je kriptografski postupak za utvrđivanje autentičnosti, integriteta i neporecivosti digitalnih poruka, dokumenata ili softvera. Omogućava zaštitu od zlonamernog menjanja podataka, lažnog predstavljanja, prevara i neovlašćenog pristupa u digitalnom prostoru.

Pre pojave digitalnih tehnologija, potpis je dugo bio jedino sredstvo za utvrđivanje autentičnosti u pisanoj komunikaciji, iako sa određenim manama. Mogao je lako da se falsifikuje i usled standardne pozicije (na poslednjoj strani dokumenta), lako su mogle da se dodaju nove ili uklone prethodne strane, ili dodaju reči koje nisu bile u izvornom dokumentu. Kao odgovor na ovo uveden je voštani pečat, kojim bi se zatvorila koverta ili svitak sa pisanom prepiskom i koji bi bio garant integriteta. Ukoliko primalac dobije kovertu sa polomljenim pečatom, znao bi da je pismo otvarano.

U digitalnoj komunikaciji falsifikovanje potpisa i menjanje teksta u porukama i dokumentima postalo je znatno jednostavnije zbog mogućnosti jednostavnog kopiranja, izmene i umnožavanja digitalnog sadržaja bez vidljivih tragova promena, i javila se potreba za tehnologijom koja će rešiti ovaj problem.

Pomoću digitalnog potpisa svaki korisnik može da potpiše dokument i taj potpis kao i sadržaj dokumenta može biti verifikovan. Ovo se postiže korišćenjem asimetrične kriptografije, kod koje se, za razliku od simetrične kriptografije gde se koristi jedan zajednički tajni ključ, koristi par matematički povezanih ključeva:

- privatni ključ: tajni ključ poznat samo pošiljaocu, koristi se za kreiranje potpisa;
- javni ključ: ključ koji je poznat svima koji razmenjuju poruke i koristi se za verifikovanje potpisa pošiljaoca.

Tehnologije kao što su blokčejn, kriptovalute i pametni ugovori se u potpunosti oslanjaju na digitalni potpis, jer je svaka transakcija zapravo poruka, potpisana privatnim ključem korisnika i na taj način se bez posrednika (centralizovanog sistema) osigurava da je korisnik i vlasnik sredstava. Tema ovog master rada je primena digitalnog potpisa u blokčejn tehnologijama.

1.2 Uloga

Digitalni potpis je tehnika koja vezuje osobu za digitalne podatke. Ova veza može biti jednostavno proverena od strane bilo koga u toj mreži. Digitalni potpis omogućava dokazivanje posedovanja privatnog ključa bez njegovog otkrivanja.

Digitalni potpis ima tri glavne funkcije:

1. Autentifikacija - potpis dokazuje da je vlasnik privatnog ključa, koji je takođe i vlasnik sredstava, autorizovao trošenje tih sredstava

2. Neporecivost - pošto se pretpostavlja da jedino pošiljalac poseduje svoj privatni ključ i niko drugi ne zna njegov privatni ključ, samo on može proizvesti potpis za određenu transakciju. Na taj način, ukoliko dođe do spora u budućnosti, primalac ima dokaz da je to stvarno bio potpis pošiljaoca

3. Integritet - autorizovana transakcija ne može biti izmenjena od strane neautorizovanih trećih lica. Tehnika koja ovo osigurava je potpisivanje heširanih[1] transakcija. Ako se promeni iznos, adresa primaoca ili makar jedan bit transakcije, menja se heš i potpis postaje nevalidan.

[1]Heširanje je proces koji preslikava ulaz proizvoljne dužine u izlaz fiksne dužine. Ovakva funkcija se naziva heš funkcija. Bilo koja mala promena ulaza, daje potpuno drugačiju vrednost, ali se za isti ulaz dobija uvek isti izlaz. Osobina ove funkcije je da je gotovo nemoguće naći inverz, tj. ulaz na osnovu izlaza.

1.3 Osnove blokčejn tehnologije

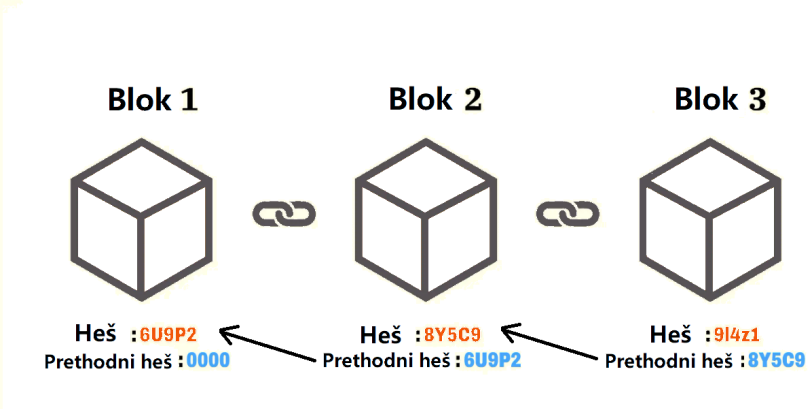
Blokčejn tehnologija predstavlja osnovu za funkcionisanje brojnih savremenih digitalnih sistema, među kojima posebno mesto zauzimaju kriptovalute. U okviru ovih sistema, blokčejn omogućava beleženje i verifikaciju transakcija bez potrebe za centralnim autoritetom, oslanjajući se na mrežu učesnika i mehanizme konsenzusa. Termin kriptovaluta ima dva značenja. Može se odnositi na samu digitalnu valutu (na primer bitkoin ili eter) ili može označavati ceo sistem koji obuhvata protokol, mrežu učesnika, pravila validacije transakcija i mehanizme konsenzusa pomoću kojih se održava validno stanje sistema.

Transakcija u kriptovalutama sadrži:

- ulaz - nepotrošeni izlazi iz prethodnih transakcija (UTXO - *Unspent Transaction Output*);
- izlaz - kome šaljemo i koji iznos;
- proviziju - iznos koji se plaća učesnicima mreže za obradu i potvrđivanje transakcije;
- digitalni potpis - kriptografski dokaz kojim pošiljalac potvrđuje da ima pravo da raspolaže sredstvima obuhvaćenim transakcijom.

Digitalni potpis je jedinstven za svaku transakciju. Čak iako dve transakcije sadrže istog pošiljaoca, primaoca i iznos, opet bi se digitalni potpis razlikovao zbog drugačijeg ulaza. Mreža održava stanje trenutnog skupa nepotrošenih izlaza UTXO i odbacuje svaku transakciju koja pokušava da potroši UTXO koji više nije dostupan.

Blokčejn predstavlja distribuiranu bazu podataka u kojoj su transakcije grupisane u blokove koji su povezani u lanac. Efekat lanca se ostvaruje time što svaki blok sadrži heš vrednost prethodnog bloka. Čuvanjem heš vrednosti prethodnog bloka, čuva se integritet i nepromenljivost podataka, jer svaka izmena u prethodnom bloku menja njegov heš i narušava lanac. Na slici 1.1, prikazano je povezivanje blokova u blokčejnu putem heš vrednosti prethodnog bloka.



Slika 1.1: Lanac blokova

Blok je struktura koja sadrži zaglavlje i telo.

- Zaglavlje sadrži:
  - verziju protokola za kreiranje blokova - određuje pravila i način na koji se blok formira i obrađuje u mreži;
  - heš prethodnog bloka;
  - merkle koren - heš vrednost korena stabla izgrađenog od svih transakcija u bloku. Ovim se obezbeđuje da se promenom jedne transakcije, menja i koren;
  - vremenski pečat - kada je blok nastao;
  - težinu T - 256-bitni broj koji u heksadecimalnom zapisu ima veliki broj početnih nula, *target*;
  - *nonce* (*number used once*) - 32-bitni (ili veći) broj.
- Telo sadrži:
  - broj transakcija;
  - listu transakcija.

Transakcije se šalju čvorovima u mreži i smeštaju u *mempool*, privremeni skup nepotvrđenih transakcija. Rudari su učesnici u mreži koji biraju iz *mempool*-a koje će transakcije uključiti u novi blok, najčešće prema visini provizije. Rudarenje je proces karakterističan za blokčejn mreže koje za postizanje konsenzusa koriste dokaz o radu (*Proof-of-Work*), u kojem rudari ulažu računarsku snagu u rešavanje kriptografskog

zadatka. Cilj je pronaći takav *nonce* da vrednost dobijena heširanjem celog bloka bude manja od zadate ciljne vrednosti T. Ovo se može raditi samo grubom silom, tj. menjanjem broja *nonce* redom. Težina se menja na nekoliko minuta da bi rešavanje bilo još teže. Rudar koji prvi pronađe validno rešenje stiče pravo da doda novi blok u blokčejn, a za to dobija odgovarajuću nagradu i preuzima provizije od transakcija.

Učesnici u sistemu (čvorovi) validiraju blokove koji su novonapravljeni. Kada čvor primi novi blok, proverava:

- da li je dobro izvršen dokaz o radu;
- da li heš pokazuje na pravi prethodni blok;
- da li su sve transakcije ispravno potpisane;
- da li pošiljaoci imaju dovoljno sredstava;
- da li ima dvostrukog trošenja, to jest provera da li su sredstva u ulazu već jednom potrošena;
- da li je nagrada rudaru ispravna;
- da li su ispoštovana sva pravila protokola.

Ako je validacija uspešna, čvor dodaje blok u svoju kopiju blokčejna i prosleđuje ga dalje. Prihvata se validni lanac sa najvećim akumuliranim dokazom o radu (najduži lanac). Zbog toga, iako svako čuva svoju kopiju, većina čvorova vremenom konvergira u isti lanac.

Dakle, transakcija je izvršena kada je uključena u validan blok koji je prihvaćen od strane mreže.

2 Digitalni potpis i osnovni pojmovi

Za razumevanje koncepta digitalnog potpisa potrebno je uvesti osnovne pojmove na kojima se zasniva. U ovom poglavlju biće predstavljeni javni i privatni ključ, kao i proces potpisivanja i verifikovanja.

2.1 Javni i privatni ključ

U nastavku ćemo formalno definisati pojmove javnog i privatnog ključa, opisati način na koji se oni generišu i međusobno povezuju, kao i jednosmerne funkcije na kojima počiva njihova bezbednost.

Neka je M skup poruka, a C skup šifrovanih poruka (šifrata). Asimetrična kriptografija koristi par ključeva: javni ključ k_{pub} za enkripciju $e_{k_{\text{pub}}} : M \to C$ i privatni ključ k_{priv} za dekripciju $d_{k_{\text{priv}}} : C \to M$ takav da za svako $m \in M$ važi jednakost:

$$d_{k_{\text{priv}}}\big(e_{k_{\text{pub}}}(m)\big) = m.$$

Poznati su javni ključ, način enkripcije i dekripcije, a jedina tajna je privatni ključ. Za neko $c \in C$, naći privatni ključ za koji važi

$$d_{k_{\text{priv}}}(c) = m,$$

gde je m originalna poruka, predstavlja težak problem i odatle proizilazi sigurnost asimetrične kriptografije.

Za generisanje para $(k_{\text{priv}}, k_{\text{pub}})$, odnosno određivanja javnog na osnovu privatnog ključa, koriste se jednosmerne funkcije

$$f : X \to Y,$$

kod kojih se vrednost

$$f(x) = y$$

može odrediti u polinomijalnom vremenu u odnosu na dužinu zapisa x u bitovima, dok je teško izračunati inverznu funkciju

$$f^{-1} : Y \to X,$$

tako da važi

$$x = f^{-1}(y).$$

Teško izračunljivo znači da ne postoji poznat algoritam koji f^{-1} računa u polinomijalnom vremenu. Iterativnim pristupom, proveravanjem svih mogućih vrednosti x, inverzna funkcija se može izračunati u eksponencijalnom vremenu $O(2^d)$, gde je d dužina zapisa y, tj. k_{pub} u bitovima. Ovo predstavlja gornju granicu složenosti dobijenu grubom silom. Postoje algoritmi manje složenosti, ali oni i dalje zahtevaju veliki broj operacija, pa za dovoljno veliku dužinu d izračunavanje f^{-1} u realnom vremenu nije izvodljivo.

2.2 Potpisivanje i verifikovanje

Proces potpisivanja i verifikovanja se sastoji iz nekoliko faza:

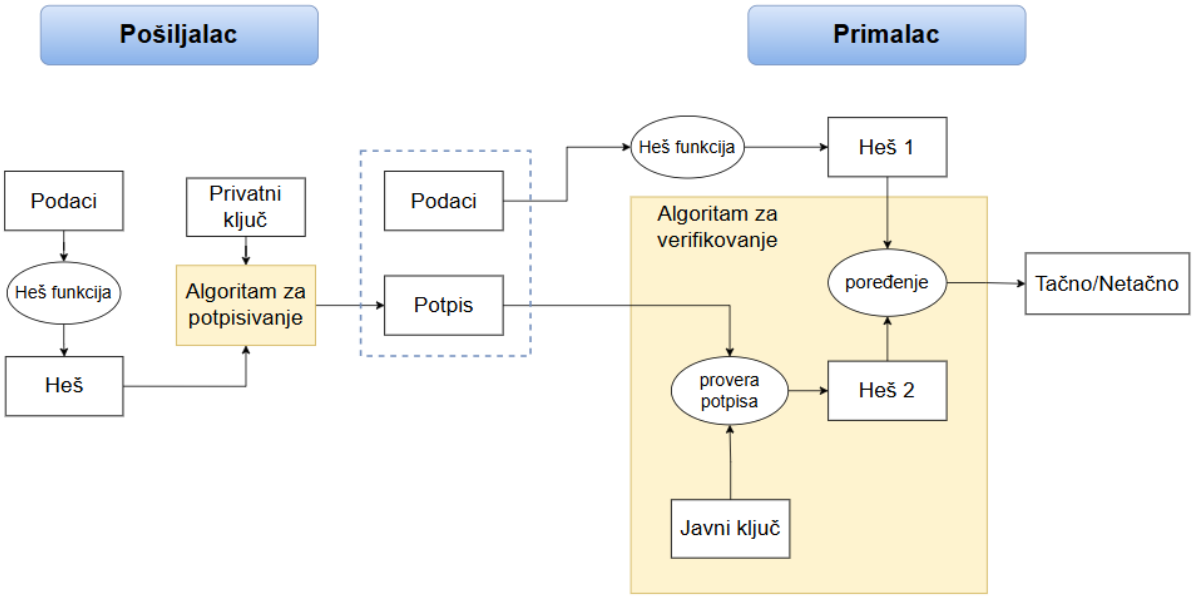
- pošiljalac generiše heš vrednost iz podataka;
- heš vrednost i privatni ključ se zatim unose u algoritam za potpisivanje, koji proizvodi digitalni potpis nad datim hešom. Potpisivanje heša je efikasnije jer se potpis računa nad kratkom vrednošću, a sami podaci mogu biti proizvoljno dugi. Funkcija potpisivanja se primenjuje nad heširanim podacima i privatnim ključem čime se dobija digitalni potpis:

  ```
  Sign(hashed_data, private_key) = Signature
  ```

- potpis se dodaje uz podatke i zajedno se šalju primaocu;
- primalac unosi digitalni potpis i javni ključ u algoritam za verifikaciju i dobija heš vrednost;
- primalac propušta primljene podatke kroz istu heš funkciju kako bi generisao heš vrednost;
- heš vrednosti se porede. U slučaju da su podaci izmenjeni od strane napadača, verifikacija potpisa neće uspeti jer se heševi neće poklapati. Funkcija verifikacije se primenjuje nad heširanim podacima, digitalnim potpisom i javnim ključem, čime se dobija potvrdan ili odrečan odgovor o validnosti digitalnog potpisa:

  ```
  Verify(hashed_data, Signature, public_key) = True/False
  ```

Na dijagramu 2.1 je prikazan proces potpisivanja i verifikovanja podataka.



Slika 2.1: Proces potpisivanja i verifikovanja

3 Eliptičke krive u kriptografiji

Eliptičke krive predstavljaju matematičku strukturu koja se koristi u različitim kriptografskim algoritmima, pre svega za digitalno potpisivanje i uspostavljanje zajedničkih tajni. Kada je potrebno šifrovati samu poruku, asimetrična kriptografija se obično kombinuje sa simetričnom kriptografijom. Na primer, Alisa i Bob žele da komuniciraju šifrovanim porukama. Najpre, oni uspostave zajedničku tajnu putem ECDH protokola (*Elliptic Curve Diffie Hellman*) tako što se dve strane dogovore koja će im biti zajednička tajna (deljeni ključ) preko nesigurnog kanala bez direktnog slanja privatnog ključa. Tada svaka strana generiše svoj privatni ključ, a zajednički ključ se dobija množenjem sopstvenog privatnog sa tuđim javnim ključem. Zatim Alisa i Bob nezavisno izvode drugi ključ iz zajedničke tajne, koji se koristi za šifrovanje poruke simetričnim algoritmom. Time se kombinuju prednosti oba pristupa: asimetrična kriptografija rešava problem bezbedne razmene ključa, dok je simetrična kriptografija pogodnija za efikasno šifrovanje velikih količina podataka.

Pre primene eliptičkih krivih, asimetrična kriptografija se već zasnivala na matematičkim problemima koje je lako rešiti u jednom smeru, dok je njihovo obrnutno rešavanje računski zahtevno. Klasičan primer predstavlja RSA, čija se sigurnost zasniva na težini faktorizacije velikih brojeva. Za razmenu ključeva koristi se i *Diffie–Hellman*-ov protokol, čija je sigurnost zasnovana na problemu diskretnog logaritma u konačnim grupama. Eliptičke krive nisu nastale zato što prethodni pristupi nisu funkcionisali, već kao rezultat traženja drugih matematičkih struktura koje bi pružile visok nivo sigurnosti uz manje ključeve i efikasnije proračune. Na primer, 256-bitni ključ zasnovan na odgovarajućoj eliptičkoj krivoj pruža približno nivo sigurnosti koji zahteva znatno veći RSA ključ veličine 3072-bita. Manji ključevi znače manje podataka koji se moraju čuvati i prenositi, kao i efikasnije izvršavanje određenih operacija. Zbog toga su eliptičke krive naročito pogodne za uređaje i sisteme u kojima su memorija, procesorska snaga, propusni opseg ili potrošnja energije ograničeni.

Ideja je bila da se pronađe matematička struktura u kojoj postoji sličan odnos između jednostavnosti direktnog i težine obrnutog problema, ali koja omogućava da se isti nivo bezbednosti postigne sa znatno manjim ključevima. Kod eliptičkih

krivih sigurnost se zasniva na težini problema diskretnog logaritma na eliptičkoj krivoj (ECDLP) koji će biti opisan u ovom poglavlju. Ako je poznata tačka G i tačka $P = lG$, relativno je lako izračunati P kada su G i l poznati, dok je izračunavanje l na osnovu G i P eksponencijalno teško u odnosu na veličinu ključa. Ova osobina se direktno koristi pri formiranju privatnog i javnog ključa.

Kod kriptografije zasnovane na eliptičkim krivama, privatni ključ je tajan broj koji bira sam korisnik, dok se javni ključ dobija primenom operacije nad javno poznatom tačkom sa krive G. Ako je privatni ključ broj l, javni ključ je tačka $P = lG$. Javni ključ može biti objavljen i dostupan svima, dok privatni ključ mora ostati tajan. Dakle, čak i uz poznavanje G i P, ne može se praktično izračunati privatni ključ l.

U ovom radu eliptičke krive su od posebnog značaja jer predstavljaju osnovu algoritama za digitalno potpisivanje zasnovanih na eliptičkim krivama, ECDSA i EdDSA.

3.1 Matematička osnova

U nastavku biće opisani osnovni pojmovi teorije grupa neophodni za razumevanje matematičke strukture eliptičkih krivih, dok se detaljniji pregled može pronaći u [7].

Definicija 1. *Grupa je algebarska struktura $(G, \circ)$, gde je G neprazan skup, $a \circ$ binarna operacija na skupu G, koja zadovoljava osobine:*

- *asocijativnost:* $(a \circ b) \circ c = a \circ (b \circ c)\ \forall a, b, c \in G$;
- *neutral:* $(\exists e \in G)(\forall a \in G)\ a \circ e = e \circ a = a$;
- *inverz:* $(\forall a \in G)(\exists a^{-1} \in G)\ a \circ a^{-1} = a^{-1} \circ a = e$.

Definicija 2. *Grupa je Abelova (komutativna) ukoliko pored navedenih osobina zadovoljava i osobinu komutativnosti $a \circ b = b \circ a, \forall a, b \in G$.*

Definicija 3. *Ako su $(G, \circ)$ i $(H, *)$ dve grupe, onda je grupa $(H, *)$ podgrupa grupe $(G, \circ)$ ukoliko je: $H \subseteq G$ i $x * y = x \circ y$, $\forall x, y \in H$.*

Definicija 4. *Grupa G je ciklična ukoliko postoji element $g \in G$ takav da se svaki element grupe može dobiti ponavljanjem grupne operacije $\circ$ nad elementom g:*

$$G = \langle g \rangle = \{e, g, g \circ g, g \circ g \circ g, ...\}.$$

Element g naziva se generator grupe.

Definicija 5. *Broj elemenata konačne grupe G je red grupe sa oznakom $|G|$.*

Definicija 6. *Red elementa $g \in G$, za konačnu grupu G, je najmanji pozitivan ceo broj n za koji važi:*

$$\underbrace{g \circ g \circ g \circ \cdots \circ g}_{n \ puta} = e.$$

Lema 1. *Za bilo koji element g konačne grupe, red elementa g jednak je redu ciklične podgrupe koju on generiše, $|\langle g \rangle|$.*

Teorema 1. *Lagranž: Red podgrupe deli red grupe, za konačnu grupu G.*

Definicija 7. *Indeks podgrupe H u konačnoj grupi G je broj h takav da važi $h \cdot |H| = |G|$.*

Definicija 8. *Konačno polje $\mathbb{F}_q$, $q = p^d$, za prost broj p, je algebarska struktura $(\mathbb{F}_q, +, \cdot, 0, 1)$ za koju važi:*

- *$\mathbb{F}_q$ ima konačan broj elemenata, $|\mathbb{F}_q| = q$;*
- *$\mathbb{F}_q$ čini Abelovu grupu sa neutralom 0 u odnosu na operaciju $+$;*
- *$\mathbb{F}_q^* = \mathbb{F}_q \setminus \{0\}$ čini Abelovu grupu sa neutralom 1 u odnosu na operaciju $\cdot$;*
- *distributivnost: $(\forall a, b, c \in \mathbb{F}_q)$ $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$.*

U polju $\mathbb{F}_p$, gde je p prost broj, multiplikativni inverz se određuje pomoću proširenog Euklidovog algoritma:

1. pronaći brojeve x, y za koje važi $ax + py = gcd(a, p) = 1$;
2. redukovati po modulu p tako da se dobije $ax = 1 \pmod{p}$ gde je x multiplikativni inverz od a.

3.2 Eliptičke krive nad konačnim poljima

Za primenu eliptičkih krivih u kriptografiji potrebno je razmatrati eliptičke krive nad konačnim poljima. U nastavku će biti definisana eliptička kriva nad konačnim poljem i objašnjene njene osnovne karakteristike.

Definicija 9. *Eliptička kriva nad konačnim poljem* $\mathbb{F}_q$, $q = p^d$, *definiše se kao:*

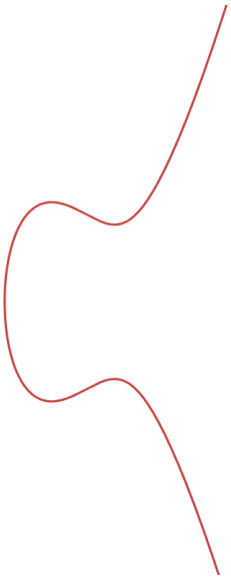
$$E(\mathbb{F}_q) = \{(x, y) \in \mathbb{F}_q \times \mathbb{F}_q | y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\},$$

gde su $a, b \in \mathbb{F}_q$ *tako da je* $\Delta = -16(4a^3 + 27b^2) \neq 0$, *a* $\mathcal{O}$ *je tačka u beskonačnosti.*

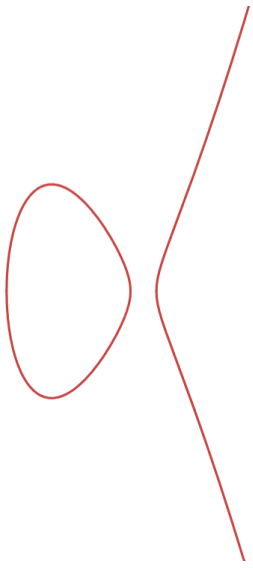
Napomena 1. U ovom radu razmatraće se krive nad konačnim poljima $\mathbb{F}_p$, $(q = p)$, gde je p veliki prost broj. U savremenim kriptografskim standardima se najčešće koriste konačna polja sa brojem elemenata približno 2^{256}.

Definicija 10. *Red eliptičke krive je broj svih tačaka koje pripadaju krivoj i označava se sa* $|E(\mathbb{F}_p)|$.

Na slikama 3.1 i 3.2 su dati primeri eliptičkih krivih nad $\mathbb{R}$. Za razliku od eliptičkih krivih nad $\mathbb{R}$, eliptičke krive nad konačnim poljem $\mathbb{F}_p$ nemaju ovakav kontinuirani geometrijski oblik, već ih čini konačan skup tačaka sa koordinatama iz $\mathbb{F}_p$.



Slika 3.1: Eliptička kriva nad $\mathbb{R}$
$y^2 = x^3 - 3x + 8$, $\Delta < 0$



Slika 3.2: Eliptička kriva nad $\mathbb{R}$
$y^2 = x^3 - 6x + 5$, $\Delta > 0$

3.3 Operacije na eliptičkoj krivoj

Tačka u beskonačnosti $\mathcal{O}$ predstavlja neutralni element.
Inverz tačke $P = (x, y)$ na eliptičkoj krivoj definisana je na sledeći način:

$$\ominus P = (x, -y)$$

dok je inverz neutralnog elementa:

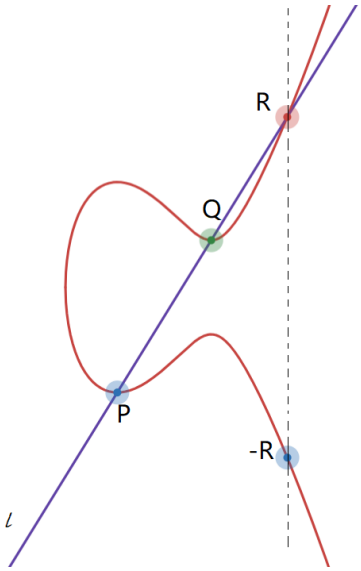
$$\ominus \mathcal{O} = \mathcal{O}.$$

Sabiranje dve tačke $P \oplus Q$ na eliptičkoj krivoj definiše se u zavisnosti od međusobnog odnosa tačaka:

1. ako je jedna od tačaka neutralni element, sabiranje je trivijalno:
   - $P = \mathcal{O}$: $\mathcal{O} \oplus Q = Q$;
   - $Q = \mathcal{O}$: $P \oplus \mathcal{O} = P$;
2. ako su P i Q različite od $\mathcal{O}$ i predstavljaju suprotne tačke, odnosno ako je $Q = \ominus P$, njihov zbir je neutralni element:
   $$P \oplus Q = \mathcal{O};$$
3. u slučaju kada su P i Q različite tačke, $P, Q \neq \mathcal{O}$, $Q \neq P, \ominus P$ povuče se prava l kroz P i Q i razmatraju se slučajevi:
   a) l seče eliptičku krivu u još tačno jednoj tački R ($\neq P, Q$);
   b) l je tangentna na eliptičku krivu u jednoj od tačaka P i Q, označimo je sa R.

   Tada je $P \oplus Q = \ominus R$.
4. ako su P i Q iste tačke, $P = Q \neq \mathcal{O}, \ominus P$ povuče se tangenta l u tački P, koja će preseći eliptičku krivu u još tačno jednoj tački R (različitoj od P).
   Tada je $2P = P \oplus P = \ominus R$.

Teorema 2. $(E(\mathbb{F}_p), \oplus, \ominus, \mathcal{O})$ *je Abelova grupa.*



Slika 3.3: Eliptička kriva $y^2 = x^3 - 3x + 3$ nad $\mathbb{R}$, slučaj 3.a)

Metod dupliranja tačke

Neka je G tačka sa eliptičke krive, a x veliki broj. Ako je potrebno izračunati xG tj. sabrati tačku G samu sa sobom x puta, biće potrebno dosta vremena za iterativno sabiranje. Koristi se efikasniji metod „dupliranje tačke" (*double and add algorithm*), kojim se višestruko sabiranje tačke G zamenjuje nizom dupliranja i sabiranja tačaka. U svakom koraku promenljiva R predstavlja trenutni rezultat, koji se duplira pri obradi svakog sledećeg bita, a ukoliko je obrađeni bit jednak 1, rezultatu se dodaje tačka G. Postupak se sprovodi kroz sledeće korake:

1. zapis broja x u binarnom obliku

$$x = x_0 + x_1 \cdot 2 + x_2 \cdot 4 + x_3 \cdot 8 + \ + x_r \cdot 2^r, x_0, x_1, ..., x_r \in \{0, 1\};$$

2. prvi bit je uvek 1, pa njega računamo kao inicijalizaciju: $R = G$;

3. za svaki sledeći bit sleva nadesno se radi:

   - dupliranje $R = 2R$;
   - ako je bit 1, dodaje se G.

Broj operacija + da bi se izračunalo xG je $O(logx)$.

Primer: Primena algoritma dupliranja tačke na izračunavanje izraza $100G$.
Za $x = 100$, binarni zapis broja je

$$100 = 2^6 + 2^5 + 2^2 = (1100100)_2.$$

Počevši od $R = G$ i obrađujući bitove sleva nadesno, dobijamo

$$100G = 2(2(G + 2(2(2(G + 2G))))).$$

Iz definicije 6 primenjene na grupu $E(\mathbb{F}_p)$ sledi da je red tačke $G \in E(\mathbb{F}_p)$ najmanji pozitivan ceo broj n za koji važi

$$nG = \mathcal{O}.$$

Jakobijeve koordinate

Jakobijeve koordinate predstavljaju projektivni prikaz tačaka eliptičke krive. Za razliku od afinih koordinata, u kojima se tačka predstavlja parom (x, y), u Jakobijevim koordinatama tačka se predstavlja trojkom $(X : Y : Z)$. Svaka afina tačka (x, y) može se predstaviti Jakobijevim koordinatama kao:

$$(X : Y : Z) = (x : y : 1),$$

a prelazak iz projektivnih u afine koordinate se izvodi transformacijom:

$$x = X/Z, y = Y/Z, \qquad Z \neq 0.$$

Važi jednakost

$$(X : Y : Z) = (\lambda X : \lambda Y : \lambda Z), \qquad \lambda \neq 0,$$

gde se λ poništava pri transformaciji u afine koordinate. Na ovaj način različite trojke mogu predstavljati istu tačku.

Za Vajerštrasovu krivu, afine koordinate (x, y) povezane su sa Jakobijevim koordinatama $(X : Y : Z)$ relacijama:

$$x = \frac{X}{Z^2}, \qquad y = \frac{Y}{Z^3}, \qquad Z \neq 0,$$

čime se dobija jednačina Vajerštrasove krive u Jakobijevim koordinatama:

$$Y^2 = X^3 + aZ^4X + Z^6b.$$

Posebno, tačka u beskonačnosti $\mathcal{O}$ u Jakobijevim koordinatama predstavlja se kao:

$$\mathcal{O} = (0 : 1 : 0).$$

Ove koordinate su pogodne za efikasnu implementaciju jer omogućavaju jednostavnije izvođenje operacija nad tačkama i u radu će biti korišćene u implementaciji ECDSA.

3.4 Problem diskretnog logaritma na eliptičkim krivama - ECDLP

Sigurnost kriptografskih algoritama zasnovanih na eliptičkim krivama zasniva se na težini rešavanja problema diskretnog logaritma na eliptičkim krivama. U nastavku će ovaj problem biti objašnjen.

Problem diskretnog logaritma na eliptičkim krivama (ECDLP) predstavlja problem pronalaženja celog broja l takvog da važi:

$$P = lG.$$

Problem se ne posmatra u celoj grupi $E(\mathbb{F}_p)$, nego u cikličnoj podgrupi generisanoj tačkom G:

$$\langle G \rangle = \{\mathcal{O}, G, 2G, 3G, \dots\}$$

i ta podgrupa ima red n, što je takođe i red tačke G po Lemi 1. Broj n se, zbog sigurnosti, standardno bira kao veliki prost broj $n \approx p$.

Zbog operacije sabiranja u grupi, l označava koliko puta treba sabrati tačku G samu sa sobom da bi se dobila tačka P koja je takođe na krivoj:

$$P = \underbrace{G + G + G + \cdots + G}_{l \text{ puta}}$$

Za izračunavanje tačke P, koristi se metod dupliranja tačke (3.3).

Tačke G i P se objavljuju, P je javni ključ, a l privatni.

Najbrži algoritam koji može rešiti ECDLP u $E(\mathbb{F}_p)$ se približno izvršava u $\sqrt{n}$ koraka. Naziva se „Pollard ρ" napad i o njemu će biti reči kasnije.
Čak i sa ovim algoritmom bilo bi neizvodljivo izračunati diskretni logaritam jer je

$$\sqrt{n} \approx \sqrt{p} = \sqrt{2^{256}} = 2^{128} \approx 10^{38}$$

i dalje ogroman broj, veći od starosti svemira u sekundama ($\approx 10^{17}$).

4 Forme eliptičkih krivih i bilinearna preslikavanja

Eliptička kriva se može zapisati u različitim oblicima jednačina, kao što su Vajerštrasov, Tvistovan Edvardsov i Montgomerijev oblik. Različiti oblici ne predstavljaju različite vrste matematičkih objekata, već različite načine predstavljanja eliptičke krive, pri čemu svaki od njih može biti pogodniji za određene operacije i primene. Zbog toga je korisno uspostaviti veze između ovih oblika i omogućiti prelazak iz jednog u drugi, uz očuvanje strukture krive. U ovom poglavlju biće predstavljene navedene forme, njihove karakteristike i operacije nad tačkama, a zatim i biracionalna preslikavanja koja omogućavaju povezivanje ovih oblika.

4.1 Vajerštrasova forma eliptičke krive

Vajerštrasova forma predstavlja jedan od osnovnih modela za zapisivanje jednačine eliptičke krive. U nastavku će najpre biti predstavljena opšta Vajerštrasova kriva, a zatim njena pojednostavljena forma koja se često koristi u kriptografiji.

Definicija 11. *Neka je $\mathbb{F}_p$ konačno polje, gde je p prost broj veći od 3. Opšta Vajerštrasova kriva nad poljem $\mathbb{F}_p$ predstavlja skup tačaka koje zadovoljavaju jednačinu:*

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

gde su $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}_p$, zajedno sa tačkom u beskonačnosti $\mathcal{O}$.

Odgovarajućom linearnom smenom eliminišu se članovi xy, y i y^2 i opšta Vajerštrasova jednačina se može svesti na (već uvedeni) jednostavniji oblik:

$$y^2 = x^3 + ax + b,$$

gde su $a, b \in \mathbb{F}_p$.

Vajerštrasova kriva je geometrijski simetrična x-osi, odnosno ako je (x, y) na krivoj onda je i $(x, -y)$.

Vajerštrasov model eliptičke krive je značajan u matematici i kriptografiji. Teorija eliptičkih krivih je razvijena nad ovim modelom koji u stvari predstavlja kriptografski standard.

4.2 Edvardsova forma eliptičke krive

Pored Vajerštrasove forme, eliptičke krive se mogu predstaviti i u drugim formama. Jedna od njih je Edvardsova forma, koja je posebno značajna zbog jednostavnije aritmetike i efikasnije implementacije operacija nad tačkama.

Definicija 12. *Neka je $\mathbb{F}_p$ konačno polje, gde je $p \neq 2$ prost broj. Edvardsova kriva nad poljem $\mathbb{F}_p$ predstavlja skup tačaka koje zadovoljavaju jednačinu:*

$$x^2 + y^2 = 1 + dx^2y^2,$$

pri čemu je parametar $d \in \mathbb{F}_p \setminus \{0, 1\}$.

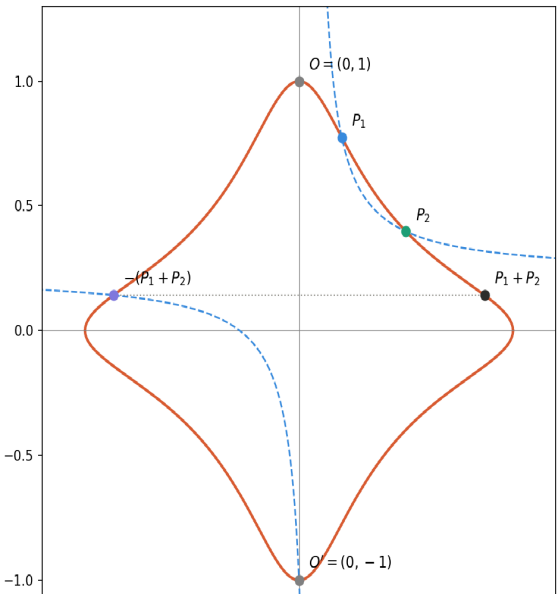
Definicija 13. *(Edvardsov zakon sabiranja). Neka su $(x_1, y_1), (x_2, y_2)$ tačke na Edvardsovoj krivoj $E : x^2 + y^2 = 1 + dx^2y^2$. Zbir ovih tačaka je:*

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - x_1x_2}{1 - dx_1x_2y_1y_2} \right).$$

Neutral je (0,1), a inverz od (x_1, y_1) je $(-x_1, y_1)$.

Sabiranje je jednostavnije u odnosu na sabiranje kod Vajerštrasovih krivih jer ove uniformne formule važe za sve tačke krive i ne razdvajaju se u posebne slučajeve.

Što se tiče geometrijske interpretacije, sabiranje tačaka na Edvardsovoj krivoj se može predstaviti konstrukcijom specifične hiperbole koja prolazi kroz tačke koje sabiramo P_1 i P_2, tačku $O = (0, -1)$ i čija jednačina zavisi od jednačine Edvardsove krive. U odgovarajućem dodatnom preseku hiperbole i krive nalazi se $-(P_1 + P_2)$.



Slika 4.1: Edvardsova kriva $x^2 + y^2 = 1 - 15x^2y^2$ na $\mathbb{R}$

Tvistovana Edvardsova kriva

Tvistovana Edvardsova kriva predstavlja proširenje Edvardsove forme koje omogućava obuhvatanje šire klase eliptičkih krivih.

Definicija 14. *Neka su $a, d \in \mathbb{F}_p$, $a, d \neq 0, a \neq d$. Tvistovana Edvardsova kriva (Twisted Edwards curve) nad poljem $\mathbb{F}_p$ predstavlja skup tačaka koje zadovoljavaju jednačinu:*

$$ax^2 + y^2 = 1 + dx^2y^2.$$

Edvardsova kriva je poseban slučaj Tvistovane Edvardsove krive gde važi $a = 1$. Uvođenjem dodatnog parametra a proširuje se skup eliptičkih krivih koje se mogu predstaviti u ovom obliku.

Definicija 15. *(Tvistovan Edvardsov zakon sabiranja). Neka su $(x_1, y_1), (x_2, y_2)$ tačke na Edvardsovoj krivoj $E : ax^2 + y^2 = 1 + dx^2y^2$. Zbir ovih tačaka je:*

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2} \right).$$

4.3 Montgomerijeva forma eliptičke krive

Pored Vajerštrasove i Edvardsove forme, biće predstavljena i Montgomerijeva forma krive kako bi se preko nje uspostavilo odgovarajuće preslikavanje između Edvardsove i Vajerštrasove forme eliptičke krive.

Definicija 16. *Neka su $A, B \in \mathbb{F}_p$, $B \neq 0$ i $A^2 \neq 4$. Montgomerijeva kriva nad poljem $\mathbb{F}_p$ predstavlja skup tačaka koje zadovoljavaju jednačinu:*

$$By^2 = x^3 + Ax^2 + x.$$

4.4 Biracionalna preslikavanja

Biracionalna preslikavanja omogućavaju prelazak između različitih oblika iste eliptičke krive. U zavisnosti od konkretne primene, mogu se koristiti različiti oblici jednačina. Na primer:

- kod Edvardsovih krivih sabiranje tačaka je jednostavno (isti način za sve slučajeve) za razliku od Vajerštrasovih krivih kod kojih ima grananja na više slučajeva u sabiranju tačaka;
- Montgomerijeve krive omogućavaju efikasno množenje tačke skalarom, što doprinosi njihovim dobrim performansama u praksi;
- Vajerštrasova kriva najviše služi u teorijske svrhe, za analizu i dokazivanje teorema kao najopštiji i najproučeniji oblik krive.

Definicija 17. *Neka su E_1 i E_2 dve eliptičke krive definisane nad istim poljem. Preslikavanje $\phi : E_1 \to E_2$ je biracionalno ako se može zapisati pomoću racionalnih funkcija[1] i ako postoji inverzno preslikavanje $\phi^{-1} : E_2 \to E_1$ koje se takođe može zapisati pomoću racionalnih funkcija, osim na konačno mnogo tačaka gde preslikavanja nisu definisana[2].*

Napomena 2. Biracionalno preslikavanje predstavlja izomorfizam grupa. Skoro svakoj tački jednog modela pridružuje se tačno jedna tačka drugog modela i obrnuto. Ono nije definisano u konačnom broju tačaka zbog racionalnih funkcija koje su

[1] Količnik dva polinoma, pri čemu imenilac nije jednak nuli.

[2] To su tačke za koje imenilac racionalne funkcije postaje 0, a ima ih konačno mnogo jer polinom ima konačan broj nula.

nedefinisane u tim tačkama. Za svaku od tih tačaka, preslikavanje se može algebarski proširiti ili opisati ekvivalentnim izrazom tako da se uspostavi korespondencija između modela. Na taj način će red prve krive biti jednak redu druge krive, odnosno broj tačaka nad konačnim poljem ostaje isti nakon preslikavanja. Očuvana je grupa sa svojim osobinama i operacijom. Ako je $P + Q = R$ u prvom modelu onda za biracionalno preslikavanje ϕ je i $\phi(P) + \phi(Q) = \phi(R)$ u drugom modelu. ECDLP ostaje jednako težak jer $Q = lP$ prelazi u $\phi(Q) = l\phi(P)$.

Definicija 18. *Eliptičke krive E_1 i E_2 su biracionalno ekvivalentne ako između njih postoji biracionalno preslikavanje.*

Preslikavanje: Montgomerijeva kriva $\leftrightarrow$ Tvistovana Edvardsova kriva

Montgomerijeva kriva i Tvistovana Edvardsova kriva su biracionalno ekvivalentne. U nastavku će prvo biti predstavljeno preslikavanje sa Tvistovane Edvardsove na Montgomerijevu krivu, a zatim inverzno preslikavanje.

Neka je Tvistovana Edvards kriva $ax^2 + y^2 = 1 + dx^2y^2$. Da bi postojala biracionalno ekvivalentna Montgomerijeva kriva, potrebno je da postoji racionalna transformacija (iz smene koja sledi, potrebno je da važe uslovi $v \neq 0$ i $u \neq -1$).
Uvodi se smena:

$$x = \frac{u}{v}, \quad y = \frac{u-1}{u+1}.$$

Sređivanjem se dobija jednačina:

$$\underbrace{\frac{4}{a-d}}_{B} v^2 = u^3 + \underbrace{\frac{2(a+d)}{a-d}}_{A} u^2 + u$$

koja predstavlja Montgomerijevu krivu.

Sada će biti predstavljeno inverzno preslikavanje, odnosno preslikavanje sa Montgomerijeve na Tvistovanu Edvardsovu krivu.

Neka je Mongomerijeva kriva $By^2 = x^3 + Ax^2 + x$. Da bi postojala biracionalno ekvivalentna Tvistovana Edvardsova kriva, potrebno je da važe uslovi: $B \neq 0$, $A \neq \pm 2$ (obezbeđuje nesingularnost krive) i da postoji racionalna transformacija (iz zamene koja sledi, to su uslovi $y \neq 1$ i $x \neq 0$).
Uvodi se smena:

$$u = \frac{1+y}{1-y}, \quad v = \frac{1+y}{(1-y)x}$$

Sređivanjem se dobija jednačina:

$$\underbrace{\frac{A+2}{B}}_{a} x^2 + y^2 = 1 + \underbrace{\frac{A-2}{B}}_{d} x^2 y^2$$

koja predstavlja Tvistovanu Edvardsovu krivu.

Preslikavanje: Vajerštrasova kriva $\leftrightarrow$ Montgomerijeva kriva

Vajerštrasova kriva i Montgomerijeva kriva su biracionalno ekvivalentne. U nastavku će prvo biti predstavljeno preslikavanje sa Vajerštrasove na Montgomerijevu krivu, a zatim inverzno preslikavanje.

Neka je Vajerštrasova kriva $y^2 = x^3 + ax + b$. Da bi postojala biracionalno ekvivalentna Montgomerijeva kriva, potrebno je da postoji tačka reda 2 na Vajerštrasovoj krivoj. Tačka koja pripada svakoj Montgomerijevoj krivoj je $(0, 0)$ i ona je reda 2. Treba naći takve tačke na Vajerštrasovoj krivoj i one su oblika:

$$P + P = \mathcal{O} \implies P = \ominus P \implies (x, y) = (x, -y) \implies P = (x, 0).$$

Kada se ova tačka ubaci u jednačinu krive dobija se:

$$0 = x^3 + ax + b.$$

Tačka reda 2 na Vajerštrasovoj krivoj postoji ako i samo ako polinom $0 = x^3 + ax + b$ ima koren u $\mathbb{F}_p$.

Neka je koren tačka $(\alpha, 0)$. Vrši se translacija za $-\alpha$ po $x-$osi da bi se došlo u tačku $(0, 0)$ što je Montgomerijeva posebna tačka.
Uvodi se smena:

$$x = u + \alpha, \;\; y = v.$$

Sređivanjem se dobija jednačina:

$$\underbrace{\frac{1}{3\alpha^2 + a}}_{B} v^2 = u^3 + \underbrace{\frac{3\alpha}{3\alpha^2 + a}}_{A} u^2 + u$$

koja predstavlja Montgomerijevu krivu.

Sada će biti predstavljeno inverzno preslikavanje, odnosno preslikavanje sa Montgomerijeve na Vajerštrasovu krivu.

Neka je Mongomerijeva kriva $Bv^2 = u^3 + Au^2 + u$. Da bi postojala biracionalno ekvivalentna Vajerštrasova kriva, potrebno je da važe uslovi: $B \neq 0$ i $A \neq \pm 2$.
Uvodi se smena:

$$u = Bx - \frac{A}{3}, \quad v = By.$$

Sređivanjem se dobija jednačina:

$$y^2 = x^3 + \underbrace{\frac{3 - A^2}{3B^2}}_{a} x + \underbrace{\frac{2A^3 - 9A}{27B^3}}_{b}$$

koja predstavlja Vajerštrasovu krivu.

Preslikavanje: Tvistovana Edvardsova kriva $\leftrightarrow$ Vajerštrasova kriva

U nastavku će biti pokazano kako se može uspostaviti preslikavanje između Tvistovane Edvardsove i Vajerštrasove krive, koristeći osobine relacije ekvivalentnosti.

Biracionalna ekvivalentnost zadovoljava sve tri osobine relacije ekvivalentnosti: refleksivnost, simetričnost i tranzitivnost. Zbog osobine tranzitivnosti, ako postoji biracionalno preslikavanje ϕ iz Tvistovane Edvardsove krive u Montgomerijevu krivu i biracionalno preslikavanje ψ iz Montgomerijeve krive u Vajerštrasovu krivu, onda postoji biracionalno preslikavanje $\psi \circ \phi$ iz Tvistovane Edvardsove krive u Vajerštrasovu. Zbog osobine simetričnosti važi i obrnuto (iz Vajerštrasove u Tvistovanu Edvardsovu krivu).

Iako Vajerštrasov i Edvardsov model imaju različite jednačine i geometrijski izgled, oni mogu predstavljati istu eliptičku krivu ako su biracionalni ekvivalentni.

5 ECDSA

Algoritam digitalnog potpisa ECDSA (*Elliptic Curve Digital Signature Algorithm*) je zasnovan na problemu diskretnog logaritma na Vajerštrasovim eliptičkim krivama. ECDSA je najšire standardizovana šema digitalnog potpisa zasnovana na eliptičkim krivama i nalazi se u standardima kao što su ANSI X9.62, FIPS 186-2, IEEE 1363-2000 i ISO/IEC 15946-2.

Najpoznatija kriptovaluta koja koristi ECDSA je Bitkoin i primenjuje se nad krivom *secp256k1* [2]. Pored Bitkoina, ECDSA koriste i Ethereum, Litecoin, Bitcoin Cash, Dogecoin.

Parametri krive *secp256k1* su:

- konačno polje $\mathbb{F}_p$ gde je $p = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF$ $FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEFFFFFC2F$ tj. $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$;
- eliptička kriva nad $\mathbb{F}_p$, $E : y^2 = x^3 + 7$;
- tačka $G = (x_G, y_G) = (0x79BE667EF9DCBBAC55A06295CE870B07$ $029BFCDB2DCE28D959F2815B16F81798,$ $0x483ADA7726A3C4655DA4FBFC0E1108A8FD17B448A68554199C$ $47D08FFB10D4B8)$, odnosno $2^{255} < x_G < 2^{256}$, $2^{254} < y_G < 2^{255}$;
- red tačke G, $n = 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE$ $BAAEDCE6AF48A03BBFD25E8CD0364141$, tj. $n \approx 2^{256}$;
- indeks $h = 1$ - prema Lagranžovoj teoremi, za indeks $h = 1$ red podgrupe generisane tačkom G jednak je redu cele grupe, pa važi $|E(\mathbb{F}_p)| = n$.

5.1 Algoritam za potpisivanje

Potrebni parametri za potpis su:

- n - red tačke generatora G, tj. broj koji ispunjava $nG = \mathcal{O}$;
- l - privatni ključ, broj iz intervala $(0, n)$ generisan nasumično;
- P - javni ključ je tačka lG na krivoj gde je G generator tačka sa krive
- $H(m)$ - heš poruke koja se potpisuje
- k - *nonce*, nasumično generisan broj koji je jedinstven u svakom potpisu. Taj broj se može koristiti samo jednom i tajan je.

Potpis je uređeni par (r, s), gde je r x-koordinata tačke $kG \bmod n$, a s se dobija izrazom $k^{-1}(H(m) + lr) \bmod n$.

Javno poznati parametri koji su poznati svim korisnicima: jednačina eliptičke krive, prost broj p koji određuje konačno polje $\mathbb{F}_p$, generator G, red n generatora G i indeks h. Javni ključ svakog korisnika je javno dostupan, dok se privatni ključ čuva u tajnosti, generiše samo jednom i koristi za svaki potpis. Heš vrednost poruke $H(m)$, *nonce* k i potpis (r, s) generišu se prilikom svakog potpisivanja.

5.2 Algoritam za verifikovanje potpisa

Verifikovanje vrši primalac tako što koristi vrednosti $H(m)$, tačku G, javni ključ P i potpis (r, s). Računaju se:

$$w = s^{-1} \bmod n$$
$$u_1 = H(m) \cdot w \bmod n$$
$$u_2 = r \cdot w \bmod n$$

$$X = u_1 G \oplus u_2 P$$

Potpis se smatra validnim ukoliko važi:

$$r \stackrel{?}{=} x(X) \bmod n,$$

gde je $x(X)$ x-koordinata tačke X. Sada ćemo objasniti zašto navedeni postupak verifikacije funkcioniše. Za formule:

$$s = k^{-1}(H(m) + lr) \pmod{n}$$

i $P = lG$, je

$$s^{-1} = k(H(m) + lr)^{-1} \pmod{n}.$$

Za tačku X se dobija

$$\begin{aligned} X &= s^{-1}H(m)G \oplus s^{-1}rP \\ &= s^{-1}(H(m) + lr)G \\ &= k(H(m) + lr)^{-1}(H(m) + lr)G \\ &= kG. \end{aligned}$$

Jednakost $X = kG$ pokazuje da se tokom verifikacije dobija ista tačka koja je korišćena pri generisanju vrednosti r, to jest x-koordinata tačke kG je jednaka $r \bmod n$.

5.3 Implementacija

U nastavku je prikazana implementacija algoritma ECDSA. Implementacija obuhvata definisanje parametara eliptičke krive i heš funkcije, generisanje privatnog i javnog ključa, generisanje digitalnog potpisa i njegovu verifikaciju. Radi jasnijeg razdvajanja navedenih funkcionalnosti, implementacija je organizovana u više *Python* fajlova.

Pošiljalac i primalac oboje znaju parametre i heš funkciju. U fajlu *public_data_and_methods.py* se definišu parametri eliptičke krive *secp256k1*, generator G, red n generatora i funkcija za izračunavanje heš vrednosti poruke.

```
1  from ecdsa.ellipticcurve import PointJacobi
2  from ecdsa.curves import SECP256k1
3
4  import hashlib
5
6  el_curve = SECP256k1
7  G = el_curve.generator
8  n = el_curve.order
9
10 def hash_message(message):
11     hash_bytes = hashlib.sha256(message).digest()
12     return int.from_bytes(hash_bytes, 'big')
```

U implementaciji se koristi modul ecdsa u kome se nalaze formula, generator i red eliptičke krive [10]. Tačke će biti reprezentovane u Jakobijevim koordinatama korišćenjem klase *PointJacobi*. U njoj su već implementirane operacije sabiranja tačaka i množenja skalarom. Koristi se heš funkcija SHA256 koja bilo koji ulaz hešira na izlaz veličine 256 bitova. Parametar *big* označava da prvi bajt niza bajtova ima najveću težinu.

U fajlu *private_and_public_key.py* je prikazana implementacija generisanja ključeva na strani pošiljaoca.

```
import secrets
from public_data_and_methods import *

private_key = secrets.randbelow(n - 1) + 1
public_key = private_key * G
```

Privatni ključ je slučajno izabran broj iz intervala $(0, n)$.

U fajlu *sign.py* je prikazana implementacija generisanja digitalnog potpisa na strani pošiljaoca.

```
from private_and_public_key import *
from public_data_and_methods import *

def sign(hashed_message, private_key):
    k = secrets.randbelow(n-1)+1
    R = k * G
    r = R.x()
    k_inv = pow(k, -1, n)
    s = (k_inv * (hashed_message + private_key * r)) % n
    return (r,s)

message = input('Enter a message: ')
hashed_message = hash_message(message.encode())
signature = sign(hashed_message, private_key)

with open('data.txt', 'w') as f:
    f.write(f'{public_key.x()} {public_key.y()} {message} {signature[0]}
          {signature[1]}')
```

Poruka koja se potpisuje se unosi sa standardnog ulaza. Slanje poruke i potpisa je ostvareno preko text fajla u koji pošiljalac upisuje, a iz koga primalac čita podatke.

U fajlu *verify.py* je prikazana implementacija verifikovanja potpisa na strani primaoca.

```
from public_data_and_methods import *

def verify(hashed_message, signature, public_key):
    r = signature[0]
```

```
5       s = signature[1]
6       s_inv = pow(s, -1, n)
7       w = s_inv
8       u1 = hashed_message * w % n
9       u2 = r * w % n
10      X = u1*G + u2 * public_key
11      return r == X.x() % n
12
13  with open('data.txt', 'r') as f:
14      public_x, public_y, message, signature_r, signature_s = f.read().split(
            ' ')
15
16  public_key = PointJacobi(SECP256k1.curve, int(public_x), int(public_y), 1)
17  signature = (int(signature_r), int(signature_s))
18  hashed_message = hash_message(message.encode())
19
20  if(verify(hashed_message, signature, public_key)):
21      print('Signature is valid!')
22  else:
23      print('Signature is not valid!')
```

Javni ključ je tačka sa krive i pretvorena je u Jakobijev zapis dodavanjem koordinate $Z = 1$.

Verifikacija digitalnog potpisa neće biti uspešna ukoliko poruka koja se proverava nije ista kao poruka koja je potpisana, odnosno ukoliko je poruka izmenjena. Takođe, verifikacija neće biti uspešna ako je potpis izmenjen.

6 EdDSA

EdDSA (*Edwards-curve Digital Signature Algorithm*) predstavlja algoritam digitalnog potpisa zasnovan na problemu diskretnog logaritma na Edvardsovim eliptičkim krivama. Jedna od specifičnosti ovog algoritma je što za svaku operaciju potpisivanja nije potrebno generisati novu slučajnu vrednost. EdDSA generiše potpise deterministički. Za dve iste poruke, potpis će biti isti.

EdDSA se koristi u kriptovalutama: Solana, Monero, Cardano itd. gde se primenjuje nad krivom *Ed25519*[1] [3].

Parametri krive *Ed25519* su:

- konačno polje $\mathbb{F}_p$ gde je $p = 0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFED$
  tj. $p = 2^{255} - 19$;
- eliptička kriva nad $\mathbb{F}_p$, $E : -x^2 + y^2 = 1 - \frac{121665}{121666}x^2y^2$; [2]
- $G = (0x216936D3CD6E53FEC0A4E231FDD6DC5C692CC7609525A7B2C9562D608F25D51A,$
  $0x6666666666666666666666666666666666666666666666666666666666666658)$
  odnosno $x_G \approx 4.6 \cdot 10^{76}, y_G \approx 4.6 \cdot 10^{76}$;
- red tačke G, $n = 0x1000000000000000000000000000000014DEF9DEA2F79CD65812631A5CF5D3ED$
  tj. $n \approx 2^{252}$;
- indeks $h = 8$ - kriva ima 8 puta više tačaka od podgrupe generisane tačkom G.

[1] Ova kriva je biracionalno ekvivalentna Montgomerijevoj krivoj Curve25519 čija je jednačina: $y^2 = x^3 + 486662x^2 + x$. Koristi se za efikasnu razmenu ključeva u mehanizmu Difi Helman na eliptičkim krivama (ECDH)

[2] $d = 121655 \cdot 121666^{-1} \pmod p$ je ceo broj u $\mathbb{F}_p$

6.1 Algoritam za potpisivanje

Potrebni parametri za potpis su:

- *seed* - tajni ključ u izvornoj formi. Njegovim heširanjem se dobijaju privatni ključ i *prefix*;
- l - privatni ključ dobijen iz prve polovine $H(seed)$ nakon postavljanja i brisanja određenih bitova;
- $prefix$ - druga polovina od $H(seed)$;
- P - javni ključ je tačka lG na krivoj gde je G generator;
- k - broj dobijen heširanjem konkateniranih vrednosti prefiksa i poruke[3] $k = H(prefix\|m) \bmod n$.

Potpis je uređeni par (R, s), gde je $R = kG$, a s se dobija izrazom $s = k + H(R\|P\|m)l \bmod n$.

Javno poznati parametri koji su poznati svim korisnicima: jednačina eliptičke krive, prost broj p koji određuje konačno polje $\mathbb{F}_p$, generator G, red n generatora G i indeks h. Javni ključ svakog korisnika je javno dostupan, dok se *seed*, $prefix$ i privatni ključ čuvaju u tajnosti, generišu samo jednom i koriste za svaki potpis. Heš vrednost poruke $H(m)$, vrednost k i potpis (R, s) generišu se prilikom svakog potpisivanja.

6.2 Algoritam za verifikovanje potpisa

Verifikovanje vrši primalac tako što koristi poruku m, tačku G, javni ključ P i potpis (R, s). Primalac nezavisno izračunava obe strane jednakosti i poredi dobijene vrednosti. Potpis se smatra validnim ukoliko važi:

$$sG \stackrel{?}{=} R \oplus H(R\|P\|m)P.$$

Sada ćemo objasniti zašto navedeni postupak verifikacije funkcioniše. Ova jednakost se dobija tako što se razvije s i iskoriste formule $P = lG$ i $R = kG$:

$$sG = (k + H(R\|P\|m)l)G = kG \oplus H(R\|P\|m)lG = R \oplus H(R\|P\|m)P.$$

[3]U ECDSA se k generiše kao nasumična tajna vrednost za svaku poruku, dok se u EdDSA k dobija deterministički iz privatnog ključa i poruke, čime se izbegava potreba za generisanjem novog slučajnog *nonce*-a pri svakom potpisivanju

Na osnovu dobijene jednakosti možemo zaključiti da je potpis matematički povezan sa javnim ključem P, koji je izveden iz tajnog ključa l.

6.3 Implementacija

U nastavku je prikazana implementacija algoritma EdDSA. Implementacija obuhvata definisanje parametara eliptičke krive i heš funkcije, generisanje privatnog i javnog ključa, generisanje digitalnog potpisa i njegovu verifikaciju. Radi jasnijeg razdvajanja navedenih funkcionalnosti, implementacija je organizovana u više *Python* fajlova.

Pošiljalac i primalac oboje znaju parametre i heš funkciju. U fajlu *public_data_and_methods.py* se definišu parametri eliptičke krive *Ed25519*, generator G, red n generatora i funkcija za izračunavanje heš vrednosti poruke.

```
from pure25519.basic import Base, L
from pure25519.basic import scalarmult_element, encodepoint,
    bytes_to_element

G = Base
n = L

import hashlib
def hash_message(message):
    hash_bytes = hashlib.sha512(message).digest()
    return int.from_bytes(hash_bytes, 'little')
```

U implementaciji se koristi modul *pure25519.basic* u kome se nalaze formula, generator i red eliptičke krive [9]. Koristi se heš funkcija SHA512 koja bilo koji ulaz hešira na izlaz veličine 64 bajtova.

U fajlu *private_and_public_key.py* je prikazana implementacija generisanja ključeva na strani pošiljaoca.

```
from public_data_and_methods import *
import secrets

def clamping(h):
    h = bytearray(h)
    h[0] &= 248
    h[31] &= 63
    h[31] |= 64
    return h

seed = secrets.token_bytes(32)
hashed_seed = hashlib.sha512(seed).digest()
```

```
13
14 private_key_bytes = clamping(hashed_seed[:32])
15 prefix = hashed_seed[32:]
16
17 private_key = int.from_bytes(private_key_bytes, 'little')
18 public_key = G.scalarmult(private_key)
```

Seed je nasumični broj veličine 32 bajta. Heširanjem se dobija vrednost dužine 64 bajta. Prva polovina se koristi za pravljenje privatnog ključa, a druga polovina je *prefix*. *Clamping* je postupak kojim se, promenom određenih bitova, sirova heš vrednost privatnog ključa prilagođava tako da bude odgovarajućeg oblika za krivu *Ed25519* tako što se:

- bitovskom konjunkcijom broja 248 i najnižeg bajta, privatni ključ postaje deljiv sa 8 (zbog brisanja tri najmanje značajna bita). Ovim se sprečavaju problemi povezani sa operacijama nad tačkama iz malih podgrupa
- bitovskom konjunkcijom najvišeg bajta sa brojem 63 dva najznačajnija bita se postavljaju na 0
- bitovskom disjunkcijom najvišeg bajta sa brojem 64 se postavlja bit 254 na 1. Time se obezbeđuje da privatni ključ nije premali ili blizu maksimalnog opsega

U fajlu *sign.py* je prikazana implementacija generisanja digitalnog potpisa na strani pošiljaoca.

```
1  from private_and_public_key import *
2
3  message = input('Enter a message: ')
4
5  k = hash_message(prefix + message.encode()) % n
6  R = G.scalarmult(k)
7
8  s = (k + hash_message(R.to_bytes() + public_key.to_bytes() +
       message.encode())*private_key) % n
9
10 with open('data.txt', 'w') as f:
11     f.write(
12         public_key.to_bytes().hex() + ' ' +
13         R.to_bytes().hex() + ' ' + str(s) + ' ' + message
14     )
```

Poruka koja se potpisuje se unosi sa standardnog ulaza. Slanje poruke i potpisa je ostvareno preko tekstualnog fajla u koji pošiljalac upisuje, a iz koga primalac čita podatke.

U fajlu *verify.py* je prikazana implementacija verifikovanja potpisa na strani primaoca.

```
from public_data_and_methods import *

def verify(message, signature, public_key):
    (R,s) = signature
    x1 = G.scalarmult(s)
    H = hash_message(R.to_bytes() + public_key.to_bytes() +
        message.encode()) % n
    x2 = R.add(public_key.scalarmult(H))
    return x1 == x2

with open('data.txt', 'r') as f:
    data = f.read().split(" ")

public_key_bytes = bytes.fromhex(data[0])
R_bytes = bytes.fromhex(data[1])
public_key = bytes_to_element(public_key_bytes)
R = bytes_to_element(R_bytes)
s = int(data[2])

signature = (R,s)

message = data[3]

if(verify(message, signature, public_key)):
    print('Signature is valid!')
else:
    print('Signature is not valid!')
```

Tokom potpisivanja, korišćena je ugrađena funkcija *to_bytes* koja pretvara tačku u bajtove. Tokom verifikovanja, koristi se funkcija *bytes_to_element* koja vraća nazad bajtove u tačku.

Kao i kod ECDSA, verifikacija digitalnog potpisa neće biti uspešna ukoliko su poruka ili potpis izmenjeni.

Link ka *github* repozitorijumu koji sadrži obe implementacije: `https://github.com/MarijaBB/ECDSA-and-EdDSA-implementation`.

7 Uporedna analiza

U nastavku će biti opisane glavne razlike između ECDSA i EdDSA, njihove prednosti i nedostaci.

7.1 Različiti indeksi krivih

Kriva *secp256k1* ima indeks 1, što znači da je ukupan broj tačaka krive jednak broju elemenata podgrupe koju generiše tačka G, što je prost broj.

Kriva *Ed25519* ima indeks 8, sto znači da postoji više tačaka na krivoj od broja elemenata podgrupe koju generiše tačka G. Pored velike podgrupe reda n, postoje i male podgrupe čiji red deli 8 i takve tačke se nazivaju tačke malog reda.

U razmeni ključeva (ECDH protokol), može se desiti da napadač pošalje kao javni ključ tačku P' sa malim redom r. Posmatrajmo privatni ključ primaoca kao:

$$l = q \cdot r + (l \bmod r).$$

Primalac pomnoži svoj privatni ključ l tačkom P' i zbog $rP' = \mathcal{O}$ dobija se:

$$S = lP' = (q \cdot r + (a \bmod r))P' = q \cdot rP' \oplus (a \bmod r)P' = (a \bmod r)P'.$$

Pošto je r malo, postoji samo nekoliko mogućnosti za vrednost S. Primalac koristi S za pravljenje ključa koji se koristi u simetričnom algoritmu šifrovanja poruke, šifruje poruku i šalje napadaču. Napadač pokušava da dešifruje, tako što isprobava sve moguće vrednosti $S = \{0 \cdot P', 1 \cdot P', ..., (r-1) \cdot P'\}$ i kada nađe smislenu dešifrovanu poruku, saznaje tačno S.

Ako napadač pošalje više različitih tačaka malog reda (reda 2, 4, 8) u više odvojenih razmena ključeva sa istim privatnim ključem primaoca l, on prikuplja $l \bmod 2, l \bmod 4, l \bmod 8$. Preko Kineske teoreme o ostacima, napadač može prikupiti dovoljno informacija za rekonstruisanje celog privatnog ključa.

Zbog toga je bitno da primalac prvo proveri da li je $8P = \mathcal{O}$. Kod javnog ključa reda n ovaj proizvod nikad nije $\mathcal{O}$, dok je kod zlonamerne tačke malog reda jednak $\mathcal{O}$, pa se takva tačka treba odbaciti.

7.2 Sigurnost

Kod ECDSA se javlja rizik zbog generisanja *nonce* vrednosti k. Taj broj se ne sme ponoviti za različite poruke. Takođe, ne sme biti moguće predviđanje njegove vrednosti. Ako napadač zna vrednost k, može izračunati privatni ključ l iz formule:

$$s = k^{-1}(H(m) + lr).$$

U obe implementacije koristi se kriptografski siguran generator koji je zasnovan na entropiji operativnog sistema (vremenski događaji, prekidi ulazno-izlaznih jedinica, hardverske i sistemske aktivnosti). Ovako je rizik od ponavljanja ili predvidljivosti brojeva sveden na minimum.

Dodatno, EdDSA koristi deterministično generisanje potpisa koje ne bira *nonce* slučajno pri svakom potpisivanju već se *nonce* generiše pomoću unapred izabranog i fiksiranog privatnog parametra (*prefix*-a) i poruke - dakle deterministički. Sledi da EdDSA generiše isti potpis za istu poruku što ne smanjuje sigurnost algoritma, jer napadač bez poznavanja *prefix*-a ne može da izvede vrednost *nonce*-a.

Oba algoritma koriste privatne ključeve veličine 256b što doprinosi sigurnosti.

7.3 Performanse

Najskuplji deo implementacije ECDSA algoritma je traženje inverza slučajno generisanog broja: $k^{-1} \bmod n$. Operacija nije direktna već se u pozadini koristi prošireni Euklidov algoritam što usporava rad programa. Modularna inverzija se ne pojavljuje u EdDSA algoritmu i time ga čini bržim.

Pored toga, sabiranje tačaka Vajerštrasove krive zahteva različite formule u zavisnosti od različitih slučajeva ($P \oplus Q$, $P \oplus P$, $\ominus P \oplus P$). Time se u implementaciji sabiranja tačaka javlja dodatna provera uslova i grananje što povećava složenost. Formule za sabiranje tačaka Edvardsove krive su iste u većini slučajeva. Tako se pojednostavljuje kod i olakšava izrada sigurne i brze implementacije.

7.4 Upotreba u blokčejn sistemima

Prednost ECDSA je dugogodišnja primena u blokčejn sistemima i široka podrška u već postojećoj infrastrukturi. Veliki broj biblioteka, novčanika i protokola podržava ovaj algoritam, što olakšava povezanost različitih sistema.

Razlog zašto Bitkoin koristi ECDSA je taj što EdDSA nije postojao u vreme dizajniranja Bitkoina. Bitkoin je nastao 2009. godine, a EdDSA 2011. godine tako da noviji sistemi (Cardano, Solana, Monero,...) koriste EdDSA.

8 Napadi na eliptičke krive

U ovom poglavlju prikazana su dva načina ugrožavanja bezbednosti sistema zasnovanih na eliptičkim krivama. Prvi je stvarni bezbednosni propust u implementaciji generatora koji je doveo do kompromitovanja privatnih ključeva, dok je drugi, Pollard ρ napad, opisan sa teorijskog aspekta kao metoda za rešavanje problema diskretnog logaritma na eliptičkim krivama.

8.1 Loš generator

Novčanik je softverska struktura koja lokalno čuva privatni ključ korisnika, podatke za upravljanje sredstvima, pomaže pri kreiranju transakcija i potpisivanju.

Određene Bitkoin novčanik aplikacije na *Android* platformi su do 2013. godine koristile *Android Java SecureRandom* generator [1] dok se nije desila greška. Generisane su predvidive ili ponovljene *nonce* vrednosti tokom potpisivanja, tako da su napadači mogli da izvedu privatne ključeve. Poznato je da je iz jednog novčanika ukradeno približno 55 Bitkoina, čija je tadašnja vrednost iznosila oko 5.500 američkih dolara.

8.2 Pollard ρ napad

Pollard ρ algoritam je metoda za rešavanje problema diskretnog logaritma na eliptičkim krivama (ECDLP), koja pokušava da pronađe l u formuli $P = lG$ na brži način u odnosu na jednostavno isprobavanje svih mogućnosti.

Neka je n red podgrupe generisane tačkom G u kojoj se rešava problem diskretnog logaritma. Osnovna ideja Pollard ρ algoritma jeste pronalaženje različitih parova celih brojeva (a', b') i (a'', b''), $a', b', a'', b'' \in \mathbb{Z}_n$ takvih da važi

$$a'G + b'P = a''G + b''P$$

Tada sledi

$$(a' - a'')G = (b'' - b')P$$

Pošto je

$$P = lG$$

dobijamo

$$(a' - a'')G = (b'' - b')lG,$$

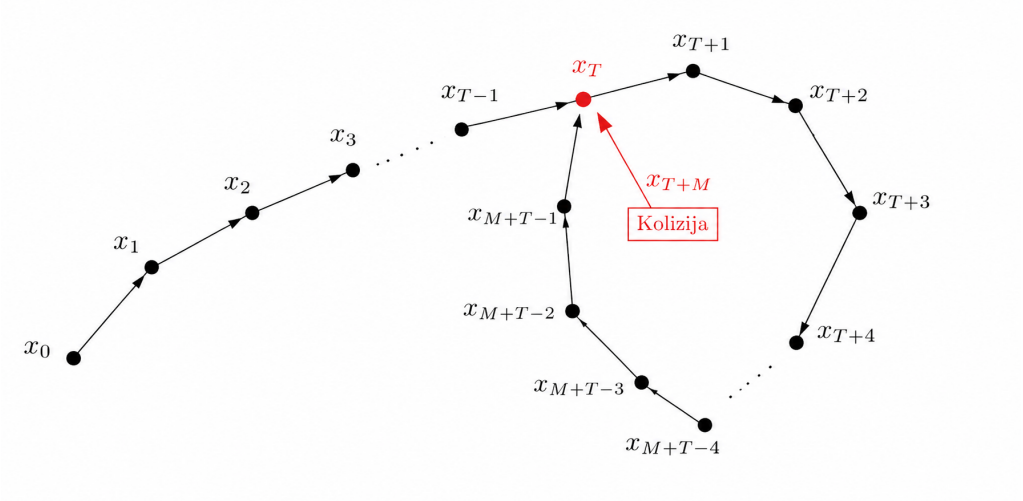
$$(a' - a'') = (b'' - b')l \pmod{n},$$

odnosno

$$l = \frac{a' - a''}{b'' - b'} \pmod{n}.$$

Naivan način za pronalaženje takvih parova (a', b') i (a'', b'') jeste da se nasumično biraju brojevi iz intervala $[0, n-1]$ i da se u tabelu smeštaju trojke $(a, b, aG + bP)$, sortirane prema trećoj komponenti [12]. Postupak se nastavlja sve dok se ista tačka $aG + bP$ ne pojavi drugi put. Takav događaj naziva se kolizija. Dakle, pronalaskom kolizije moguće je rešiti ECDLP i time otkriti vrednost privatnog ključa.
Prema paradoksu rođendana, očekivani broj iteracija do prve kolizije iznosi približno $\sqrt{n}$ pokušaja. Zbog veličine broja n ($\approx 2^{256}$ kod *secp256k1* krive ili $\approx 2^{252}$ kod *Ed25519*), teško je naći koliziju.

Slikoviti prikaz algoritma izgleda kao grčko slovo ρ i odatle naziv algoritma:



Slika 8.1: Pollard ρ metod, x_i predstavlja tačku $a_iG + b_iP$

9 Zaključak

Cilj ovog rada je upoznavanje sa konceptom digitalnog potpisa, zašto postoji, kada se koristi i kako. Proučeni su teorijski deo i implementacija algoritama za pravljenje digitalnih potpisa u modernim kriptovalutama kao što su Bitkoin i Solana. U teorijskom delu prikazani su osnovni koncepti asimetrične kriptografije, matematičke osnove eliptičkih krivih i opis oba algoritma, dok je u praktičnom delu realizovana njihova implementacija u programskom jeziku *Python*.

Analizom algoritama, može se zaključiti da ECDSA i EdDSA obezbeđuju visok nivo bezbednosti kada se pravilno koriste i kada su pravilno implementirani. Većina incidenata koji su se desili nisu bili usled "sloma krive" matematički, već zbog grešaka u implementaciji.

ECDSA je široko prihvaćeno rešenje koje se koristi u velikom broju postojećih sistema, a EdDSA predstavlja savremenije rešenje kojim se eliminišu problemi vezani za nepravilno generisanje *nonce* vrednosti i pojednostavljuje implementacija, uz zadržavanje visokog nivoa bezbednosti i efikasnosti.

Bibliografija

[1] blog.lxgr. Android's securerandom - not even nonce. https://blog.lxgr.net/posts/2013/08/15/android-securerandom-not-even-nonce/, 2013.

[2] Daniel R. L. Brown. Standards for efficient cryptography. https://www.secg.org/sec2-v2.pdf, 2010.

[3] Centre for research on Cryptography and Security. Standard curve database. https://std.neuromancer.sk/other/Ed25519, 2020.

[4] David Forney. Mit course 6.451, introduction to finite fields. https://web.stanford.edu/~marykw/classes/CS250_W19/readings/Forney_Introduction_to_Finite_Fields.pdf, 2005.

[5] Money. What is blockchain? https://money.com/what-is-blockchain/, 2022.

[6] Tanja; Bernstein Daniel J. Peters, Christiane; Lange. Curves, codes, and cryptography. https://backend.orbit.dtu.dk/ws/portalfiles/portal/6292900/prod21322797327838.20110510.diss%5B1%5D.pdf, 2011.

[7] Zoran Petrović. *Algebra 1.* 2015.

[8] Tutorials Point. Cryptography digital signatures. https://www.tutorialspoint.com/cryptography/cryptography_digital_signatures.htm, 2021.

[9] pypi.org. pure-python curve25519/ed25519 routines. https://pypi.org/project/pure25519, 2018.

[10] pypi.org. Ecdsa cryptographic signature library (pure python). https://pypi.org/project/ecdsa, 2026.

[11] Jeffrey Hoffstein; Jill Pipher; Joseph H. Silverman. *An Introduction to Mathematical Cryptography.* 2008.

[12] Darrel Hankerson; Alfred Menezes; Scott Vanstone. *Guide to Elliptic Curve Cryptography*. 2004.

[13] Dragan Đokić. kurs kriptografija - master. `https://poincare.matf.bg.ac.rs/~dragan.djokic/kriptografija_9.pdf`, 2024.

Biografija

Marija Bogavac je rođena 18. februara 2001. godine u Vranju. Završila je prirodno-matematički smer u Gimnaziji „Bora Stanković" u Vranju. Osnovne akademske studije završila je na Matematičkom fakultetu Univerziteta u Beogradu, na smeru Računarstvo i informatika, 2023. godine, sa prosečnom ocenom 8,70. Nakon završetka osnovnih studija upisala je master akademske studije na Matematičkom fakultetu, na programu Matematika i računarstvo.

Tokom potrage za zaposlenjem, bavila se razvojem različitih projekata, od igrica do bioinformatičke aplikacije za analizu proteina, kao i aplikacije za prikupljanje, obradu i vizuelizaciju *Ethereum* transakcija. Projekti su uglavnom razvijani u programskom jeziku *Python*.

Od 2024. godine zaposlena je u kompaniji *Euronet*, gde radi u oblasti finansijskih tehnologija. Tokom više od dve godine rada stekla je iskustvo sa *SQL* bazama podataka i programskim jezicima *C#* i *JavaScript*.