# Tehnologije raspodijeljenih registara

**Sosa, Ante**

**SVEUČILIŠTE U ZAGREBU**

**PRIRODOSLOVNO–MATEMATIČKI FAKULTET**

**MATEMATIČKI ODSJEK**

Ante Sosa

# TEHNOLOGIJE RASPODIJELJENIH REGISTARA

Diplomski rad

Voditelj rada:
Izv.prof.dr.sc. Zoran Škoda

Zagreb, rujan 2020.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

*Zahvaljujem se mentoru izv.prof.dr.sc. Zoranu Škodi na strpljenju i pomoći prilikom pisanja ovoga rada. Također se zahvaljujem svojoj obitelji koja mi je pružala neizmjernu podršku tijekom studiranja.*

# Sadržaj

# Uvod

Raspodijeljeni registri (engl. distributed ledgers) uključujući lance klijetki (engl. blockchains) pojavili su se 2009. godine kao javno ovjereni zapisnici podataka u bizantinskom raspodijeljenom okruženju. Raspodijeljeno okruženje je mreža samostalnih sudionika, zvanih čvorovi, gdje jedno računalo predstavlja jedan čvor. Bizantinsko okruženje je okruženje gdje nisu nužno svi čvorovi dobronamjerni u suradnji, nego svatko gleda svoj interes. Svaki čvor u mreži zapisuje podatke u klijetku, a cilj raspodijeljene mreže je postići suglasje (konsenzus) o valjanosti stanja javnog zapisa, međusobnim izmjenjivanjem informacija preko nepouzdane i potencijalno ugrožene mreže. Uz pomoć kriptografije i izvjesnih poticaja, suglasja u bizantinskim okruženjima su moguća. Kolektivno ovjereni zapisi u ovoj tehnologiji su trajni i neizmjenjivi. Zapisi mogu imati raznu semantiku, od zapisa vlasništva, prijenosa vlasništva, do ugovora i procedura poslovanja i komunikacije.
Lanci klijetki se mogu podijeliti u generacije po svojstvima njihovih protokola i algoritama: prvu generaciju predstavljaju projekti s jednim lancem, te koriste algoritam dokaza o radu (Bitcoin), drugu generaciju predstavljaju jednolančani projekti koji podržavaju pametne ugovore (Ethereum), treća generacija uvodi više lanaca i prelazi na algoritam dokaza o zalogu (EOS), te naposljetku četvrta generacija koja je još u razvoju uvodi brzu komunikaciju među lancima te ugrađenu funkcionalnost dijeljenja lanaca na lance krhotina (TON). Većina projekata spada u točno jednu od navedenih kategorija. Cilj svakog lanca klijetki je postići tri svojstva: sigurnost, skalabilnost i decentraliziranost.

# Poglavlje 1

# Bitcoin

## 1.1 Uvod

Bitcoin je decentralizirani elektronički novac koji omogućuje transakcije od korisnika do korisnika (engl. peer-to-peer) bez potrebe za trećom stranom. Bitcoin nije prvi elektronički novac, ali prvi rješava problem dvostruke potrošnje (engl. double spending). Rad [8] skrivenog autora poznatog pod pseudonimom Satoshi Nakamoto, objavljen je 2008. godine, a mreža Bitcoina zaživjela je 2009. godine. Glavna motivacija za razvoj Bitcoina bila je nepovjerenje u financijske institucije, koje služe kao pouzdana treća strana. Internetske transakcije koje se oslanjaju isključivo na financijske institucije isključuju mogućnost slanja manjih količina novca, zbog velikih transakcijskih naknada. U usporedbi s fizičkim novcem koji počiva na povjerenju financijskim institucijama, protokol Bitcoina temelji se na kriptografiji. Protokol je siguran sve dok pošteni čvorovi kontroliraju većinu procesorske snage.
Bitcoin ima karakteristike valute[5]:

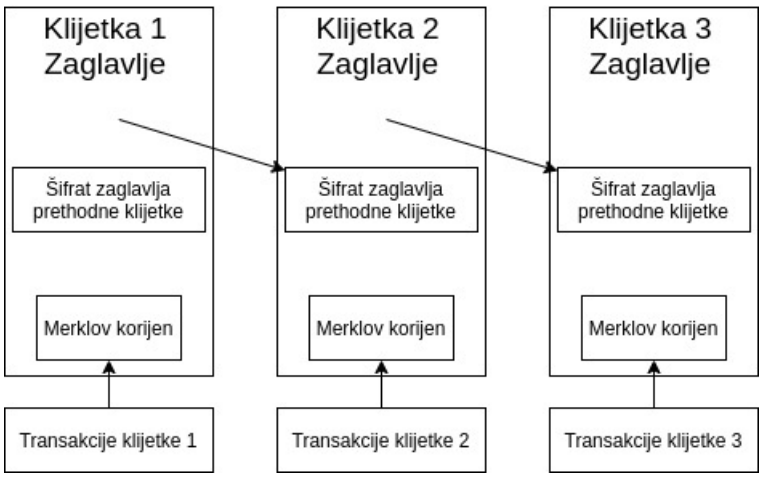
- Rijetkost - protokol Bitcoina ima određenu ukupnu količinu novčića (engl. coins) koji će ikada biti u opticaju (21 milijun novčića Bitcoina).
- Zamjenjivost - svaki novčić Bitcoina ima istu vrijednost kao bilo koji drugi i lako je zamjenjiv za proizvoljno dobro ili uslugu.
- Djeljivost - svaki novčić Bitcoina djeljiv je na 100,000,000 jedinica, nazvanih Satoshi, po autoru originalnog kôda Bitcoina.
- Trajnost - svaki novčić Bitcoina je trajan i jednako vrijedan bez obzira o vremenu nastanka.
- Prenosivost - sredstva su lako prenosiva koristeći privatne ključeve.

## 1.2 Općenito

Mrežu Bitcoina čine čvorovi koji su međusobno povezani. Svako računalo koje je spojeno na mrežu Bitcoina zove se čvor. Puni čvor (engl. Full node) je čvor koji potpuno verificira sva pravila protokola Bitcoina, te je jedini način za korištenje mreže Bitcoina potpuno pouzdano. Svaki puni čvor drži javno raspodijeljeni digitalni zapisnik. Digitalni zapisnik Bitcoina je lanac klijetki (engl. blockchian). Klijetka (engl. block) sastoji se od zaglavlja i tijela. U tijelu klijetke spremljeni su podaci o transakcijama. Transakcije su zapravo promjene stanja zapisnika. U zaglavlju klijetke spremljeni su sljedeći podaci:

- Verzija (4 bajta) - koristi se za provjeru je li klijetka izrudarena koristeći najnoviju verziju konsenzusa.
- Šifrat (engl. hash) prethodne klijetke (32 bajta) - referenca prethodne klijetke.
- Šum (engl. nonce) (4 bajta) - slučajna brojčana vrijednost koju tvorac klijetke koristi za manipulaciju vrijednosti šifrata u procesu rudarenja.
- Korijen Merkleovog stabla (engl. Merkle root) (32 bajta) - šifrat korijena Merkleovog stabla od transakcija klijetke.
- Vremenska oznaka (4 bajta) - vremenska oznaka klijetke u UNIX formi.
- Težinska meta (4 bajta) - trenutni težinski cilj u kompaktnoj formi.

Prva klijetka je klijetka postanka (engl. genesis block). Sve klijetke u lancu osim klijetke postanka koriste šifrat prethodne klijetke za računanje svoga šifrata. Time je moguće pratiti promjene stanja zapisnika od bilo kojeg trenutka do klijetke postanka.




Slika 1.1: Lanac klijetki

Dodatno, to čini lanac klijetki Bitcoina nepromjenjivim, jer ako se neka transakcija u klijetki $n$ promjeni, promijenit će se i šifrat klijetke $n$, zajedno sa šifratima svih klijetki nakon klijetke $n$. U slučaju da zlonamjerni rudar pokuša proširiti izmijenjene klijetke, pošteni rudari će brzo shvatiti da klijetka $n$ sadrži nevaljanu transakciju te će je odbaciti.
Kako protokol Bitcoina nema centralni server, komunikacija među čvorovima se odvija direktno od čvora do čvora. Stoga, ako se pojedini čvor isključi iz mreže, mreža ostaje i dalje aktivna ako postoji barem jedan aktivan puni čvor. Prema Metcalfeovom zakonu [9] korisnost mreže Bitcoina je proporcionalna kvadratu njenih korisnika, tj. punih čvorova. Drugim riječima, svaki novi puni čvor koji se spoji na mrežu Bitcoina, dodaje korisnost sustavu nelinearno. Korisnost je jednaka zbroju svih mogućih kanala među punim čvorovima i iznosi: $\frac{n(n-1)}{2}$, gdje je $n$ broj punih čvorova. Nije realno očekivati da će svaki čvor biti spojen sa svakim drugim, što se fenomenološki može izraziti uračunavanjem faktora $A$ koji s vremenom opada. Korisnost se tada izražava preciznije: $A \times \frac{n(n-1)}{2}$.

## 1.3 Kriptografija Bitcoina

Funkcija raspršivanja (engl. hash function) je funkcija koja se koristi za preslikavanje podataka proizvoljne veličine u vrijednost fiksne veličine. Vrijednost koju funkcija raspršivanja vrati zove se šifrat (engl. hash). Idealna funkcija raspršivanja ima sljedeća svojstva:

- Deterističnost - šifriranje iste vrijednosti uvijek rezultira identičnim šifratom.
- Brzina - svaku vrijednost moguće je brzo šifrirati.
- Jednosmjernost - za dani šifrat nemoguće je generirati vrijednost koja ga generira.
- Nije moguće pronaći dvije različite vrijednosti koje generiraju isti šifrat.
- Mala promjena vrijednosti rezultira promjenu šifrata tako da nova vrijednost šifrata nije korelirana sa starom vrijednošću šifrata.

Pod pojmom nemoguće misli se računalno neisplativo. Bitcoin u svojem protokolu koristi simetričnu kriptografiju koja je pogodna za raspodijeljeno okruženje jer koristi isti ključ za šifriranje i dešifriranje. Koriste se sljedeće funkcije šifriranja: SHA-256, RIPEMD-160. Za kreiranje privatnih i javnih ključeva Bitcoina koristi se algoritam digitalnog potpisa eliptičke krivulje (engl. Elliptic Curve Digital Signature Algorithm). Eliptičke krivulje su krivulje trećeg reda koje nad poljem kompleksnih brojeva opisuju plohe genusa 1, a imaju prirodnu strukturu Abelove grupe. Za kriptografiju su relevantna racionalna rješenja, odnosno rješenja nad konačnim poljima. Eliptička krivulja koja se koristi u Bitcoinu definirana je jednadžbom: $secp256k1 : Y^2 = (X^3 + 7)$ nad poljem $(F_p)$.

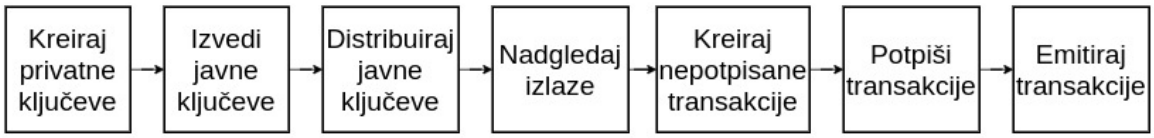
## 1.4 Adresa na Bitcoinu

Adresa na Bitcoinu je identifikator od 25-35 alfanumeričkih znakova, isključujući veliko slovo ”O”, veliko slovo ”I”, malo slovo ”l” i znamenku ”0” kako bi se izbjegla vizualna dvosmislenost. Adresa je anonimna u smislu da nitko ne može znati kome adresa pripada. Može se generirati izvanmrežno, tj. bez povezanosti na mrežu. Standardizirani zapis adrese je u obliku QR kôda, koji je moguće jednostavno skenirati mobilnim uređajem. Privatni ključ Bitcoina je slučajno generiran broj, koji služi za potpisivanje transakcija, tj. slanje i primanje novčića Bitcoina. Koristeći kriptografske funkcije, javni ključ se uvijek može generirati iz privatnog ključa, ali privatni ključ se ne može generirati iz javnog. Adresa predstavlja moguće odredište za plaćanje.

**Algoritam 1:** Kreiranje adrese iz javnog ključa

1. Primijeni funkciju SHA-256 na javni ključ.
2. Primijeni funkciju RIPEMD-160 na rezultat prethodnog koraka.
3. Slijepi verziju bajta kao prefiks rezultatu prethodnog koraka (00 je verzija bajta, a 0x00 za glavnu mrežu).
4. Primijeni dva puta funkciju SHA-256 na rezultat prethodnog koraka.
5. Prva 4 bajta rezultata iz prethodnog koraka slijepi na kraj rezultata koraka 3.
6. Primijeni funkciju Base58 na binarnu adresu Bitcoina dobivenu u 5. koraku.

Base58 je klasa shema za prevođenje binarno zapisanog teksta u tekstualni zapis. Izmislio ga je osnivač Bitcoina, Satoshi Nakamoto. Slično je Base64 kodiranju, samo je promijenjeno da se izbjegne vizualna dvosmislenost između određenih alfanumeričkih znakova. Kada se računa šifrat u protokolu Bitcoina uobičajeno se računa dva puta. Većinom se koristi funkcija SHA-256, ali kada je poželjan kraći šifrat koristi se funkcija RIPEMD-160. Novčanik za Bitcoin je program koji generira privatne ključeve, izvodi pripadajuće javne ključeve, ako je potrebno distribuira javne ključeve, nadgleda nove nepotrošene izlaze (ishode) javnih ključeva, stvara i potpisuje transakcije trošeći izlaze (ishode) i emitira potpisane transakcije.



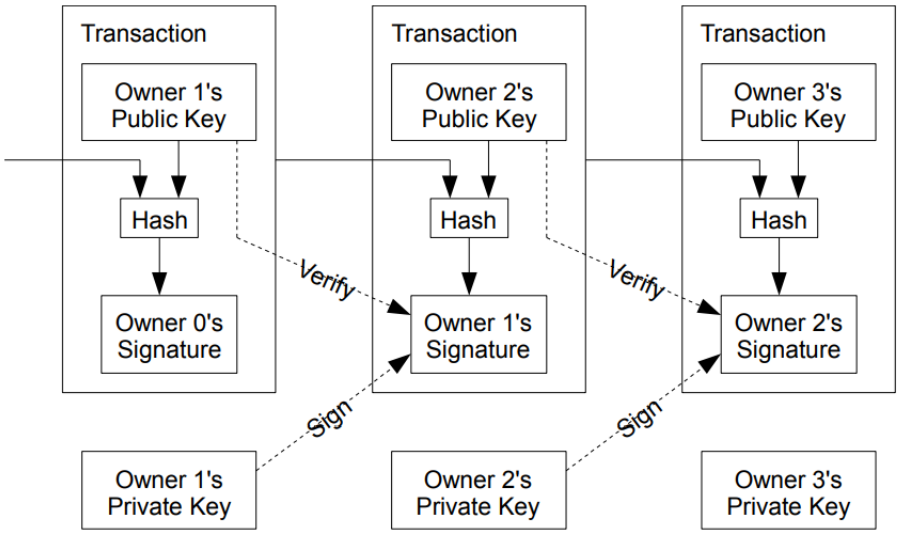

Slika 1.2: Novčanik

Novčanik može biti izvanmrežan zapis ili bilo koje sredstvo koje sadrži informaciju o

ključevima. Dodatno, novčanik može biti zapis sjemena koji omogućuje manipulaciju s ključevima. Adrese se mogu generirati na deterministčan način izvođenjem ključeva iz sjemena (engl. seed), koristeći hijerarhijski deterministički novčanik (engl. hierarchical deterministic wallet). Hijerarhijski deterministički novčanik ili skraćeno HD novčanik je uveden s BIP32 i kasnije unaprijeđen s BIP44. BIP je skraćeno od prijedlog za poboljšanje Bitcoina (engl. Bitcoin improvement proposal). HD novčanik generira niz privatnih ključeva, gdje je svaki ključ određen prethodnim ili roditeljskim privatnim ključem u nizu, a prvi ključ u nizu je određen sjemenom. Stoga, poznavanje sjemena dozvoljava korisniku obnavljanje svih ostalih privatnih ključeva u nizu.
Trenutno postoje tri formata adresa: P2PKH (skraćeno od engl. Pay to public key hash), P2SH (skraćeno od engl. Pay to script hash) i Bech32 adresa. P2PKH adresa počinje brojkom 1 i osjetljiva je na mala i velika slova, P2SH počinje brojkom 3 i osjetljiva je na mala i velika slova, a Bech32 počinje s ”bc1” i nije osjetljiva na mala i velika slova.
Izvorno se koristio P2PK (skraćeno od engl. Pay to public key) format adrese, koji je zapravo predstavljao plaćanje na javni ključ. Satoshi se kasnije odlučio na korištenje P2PKH formata, koji predstavlja plaćanje na šifrat javnog ključa. Dva su osnovna razloga za tu promjenu. Prvo, kriptografija eliptičkih krivulja koja se koristi za generiranje privatnih i javnih ključeva ranjiva je modificiranim Shorovim algoritmom za rješavanje problema diskretnog logaritma na eliptičkim krivuljama. To znači da se, koristeći kvantna računala, privatni ključ može izvesti iz javnog ključa. Kvantna računala dovoljnog kapaciteta za ugrožavanje trenutnog nivoa standardne kriptografije tipa SHA-3 još nisu sagrađena. Povećanjem malih eksperimentalnih uređaja, očekuje se dostizanje granice za ugrožavanje kriptografske sigurnosti prototipnih uređaja za nekoliko godina, a komercijalnih nekoliko godina kasnije. Koristeći šifrat javnog ključa za primanje, otkrivanje javnog ključa samo kada se novčići Bitcoina potroše i jednokratno koristeći adrese, takav napad nije moguć. Drugo, koristeći manji šifrat, lakše ga je isprintati i ugraditi u manje medije za pohranu, npr. QR kôd.

## 1.5 Transakcije

Elektronički novčić definira se kao lanac digitalnih potpisa. Vlasnik elektroničkog novčića, svojim privatnim ključem može dokazati vlasništvo elektroničkog novčića. Vlasnik može poslati isti drugom korisniku, potpisivanjem transakcije i javnog ključa novog vlasnika. Transakcija koja je potpisana dokazuje da vlasnik posjeduje taj novčić, tj. da ga ima pravo poslati. Primalac ne može potvrditi da se primljeni novčić već prije nije potrošio ako nema sve ostale transakcije. Primalac treba dokaz da je svaku transakciju, potvrdila većina čvorova. To se može postići bez treće strane ako su sve transakcije objavljene javno. Primalac može biti siguran da pošiljalac već prije nije potrošio transakciju ako ne postoji nijedna druga transakcija koja konzumira prvu i dogodila prije prve. Kako bi se postigao konsenzus oko vremena transakcija potreban je dokaz o radu (engl. proof-of-work).




Slika 1.3: Transakcija

Transakcija se sastoji od jednog ili više ulaza i jednog ili više izlaza (ishoda). Transakcijski ulaz je referenca izlaza prethodne transakcije. Izlaz je potrošen ako je iskorišten u nekoj transakciji kao ulaz. Osnovna sustavna jedinica transakcije Bitcoina je transakcijski izlaz. Transakcijski izlazi su nedjeljivi komadi valute bitcoin, zabilježeni na lancu klijetki i cijela mreža ih je prepoznala kao valjane. Puni čvorovi Bitcoina prate sve dostupne i nepotrošene transakcijske izlaze (engl. unspent transaction outputs) ili kraće UTXO. Skup UTXO transakcija raste stvaranjem novih izlaza i smanjuje se konzumacijom (trošenjem) postojećih. Svaka transakcija predstavlja promjenu stanja UTXO skupa.
Bitcoin Core je softver krajnjeg korisnika koji služi za slanje transakcija, te općenito interakciju s mrežom Bitcoina. Koristeći Bitcoin Core moguće je dohvatiti izvornu transakciju, koja dekodirana ima sljedeći oblik:

```
{
  "version": 1,
  "locktime": 0,
  "vin": [
    {
      "txid": "<transakcijski_šifrat>",
      "vout": 0,
      "scriptSig" : "<prvi_dio_sigscript>[ALL]<drugi_dio_sigscript>",
      "sequence": 4294967295
    }
  ],
```

```
  "vout": [
    {
      "value": 0.01500000,
      "scriptPubKey": "OP_DUP OP_HASH160 <adresa_drugog_izlaza>
                       OP_EQUALVERIFY OP_CHECKSIG"
    },
    {
      "value": 0.08450000,
      "scriptPubKey": "OP_DUP OP_HASH160 <adresa_prvog_izlaza>
                       OP_EQUALVERIFY OP_CHECKSIG",
    }
  ]
}
```

Transakcija sadrži jedan ulaz, koji se sastoji od 4 elementa:

- Transakcijski identifikator (txid) - referenca na transakciju koja sadrži izlaz koji se troši.
- Indeks (vout) - određuje koji se izlaz troši iz navedene transakcije.
- Otključavajuća skripta (scriptSig)
- Redni broj (sequence)

Transakcija sadrži dva izlaza, svaki od njih je definiran vrijednošću i kriptografskom skriptom koja određuje uvjete za trošenje. Izvorna transakcija ne sadrži adresu pošiljatelja, adresu primatelja čak ni vrijednost koja je poslana nije eksplicitno navedena. Protokol Bitcoina ne sadrži novčiće, pošiljatelje, primatelje, račune, stanja računa ni adrese. Sve to je napravljeno na većoj razini kako bi se korisnicima olakšala upotreba i razumijevanje protokola Bitcoin. Većina transakcija sadrži transakcijsku naknadu, koja služi kao kompenzacija rudaru za uključivanje transakcije u klijetku. Transakcijska naknada je razlika zbroja svih transakcijskih ulaza i zbroja transakcijskih izlaza. Visina naknade ne ovisi o veličini same transakcije u bitcoinima, već o veličini transakcije u kilobajtima. Općenito, transakcijske naknade su određene tržištem, tako što algoritmi, koje koriste rudari za formiranje novih klijetki uključuju transakcije ovisno o raznim kriterijima koji su stvoreni prema vlastitim interesima rudara. U nekim situacijama, moguće je da ju rudar uključi u svoju klijetku besplatno, tj. bez naknade.

## 1.6 Rudarenje

Rudarenje (engl. mining) je proces stvaranja novih klijetki i samim time produživanje lanca klijetki. Za stvaranje novih klijetki protokol Bitcoina koristi koncept dokaza o radu. Do-

kaz o radu prvi puta se spominje u radu iz 1997. godine pod nazivom Hashcash [1] autora Adam Back-a. Hashcash je izumljen kao metoda odvraćanja neželjene pošte i sprječavanje distribuiranog napada uskraćivanja usluge (engl. DDoS attack). Hashcash je osmišljen kao štit sustava koji uz pomoć dokaza o radu, služi kao usko grlo kompjutorske aktivnosti. Problem neželjene pošte rješava se obvezom pošiljatelja da obavi računarski složen proces, prije svake poslane poruke. Kao rezultat, server može pretpostaviti da je pošiljalac dobronamjeran, jer u suprotnom mora potrošiti ogromnu količinu procesorske snage za napad. Podešavanjem težine, tj. količine procesorske snage potrebne za slanje pošte, usko grlo se može ostvariti u bilo kojoj situaciji.
Bitcoin koristi sličnu primjenu dokaza o radu za postizanje sigurnosti mreže, te za stvaranje novih bitcoina. Puni čvorovi koji sudjeluju u procesu rudarenja zovu se rudari. Svaki puni čvor održava memorijski bazen (engl. Mempool) za sebe. Nakon što čvor verificira transakciju, ona je u stanju čekanja da ju rudar uključi u svoju klijetku. Rudari odabiru novopredložene transakcije iz memorijskog bazena i grupiraju ih u klijetku.

**Algoritam 2:** Dokaz o radu

1 Ako (SHA256(redni_broj_klijetke + šifrat_prethodne_klijetke + vremenska_oznaka + transakcijski_podaci + šum) < težinska_meta)
2 Onda proširi dokaz o radu mreži
3 Inače povećaj šum i ponovi korak 1

Proces rudarenja uključuje šifriranje potencijalne klijetke algoritmom SHA256 i provjeravanje zadovoljava li dobiveni šifrat trenutni težinski cilj i ako ne zadovoljava mijenjanje šuma u zaglavlju klijetke i ponovno šifriranje sve dok se ne pronađe valjana klijetka sa šumom manjim od težinskog cilja. Nakon što rudar pronađe valjani šifrat proširi ga po mreži koristeći protokol ogovaranja. Protokol ogovaranja funkcionira tako što čvor koji ima informaciju koju želi proširiti slučajno odabere još jedan čvor i pošalje mu informaciju, te svaki čvor koji ima informaciju šalje ju drugom čvoru. Nakon što rudar primi novu klijetku, trivijalno je lagano primijeniti funkciju šifriranja nad primljenom konfiguracijom klijetke i verificirati da je šifrat valjan. Svaki rudar se može ponašati kako god poželi u sustavu lanca klijetki, ali pravila konsenzusa određuju da samo valjane promjene lanca klijetki budu prihvaćene od strane svih ostalih čvorova. Da bi drugi čvorovi prihvatili novu klijetku ona mora imati sljedeća svojstva:

- Šifrat klijetke mora biti manji ili jednak od trenutnog težinskog cilja.
- Vremenska oznaka mora biti valjana.
- Vremenska oznaka mora biti veća od medijana vremenskih oznaka prethodnih 11 klijetki i ne smije biti kasnije od 2 sata u budućnosti.

- Transakcije moraju biti valjane.
- Klijetka ne smije sadržavati već potrošene izlaze (Dvostruka potrošnja nije dozvoljena).
- Sve potrošnje su autorizirane, tj. scriptSigs s dodatnim podacima koji čine svjedočanstvo zadovoljavaju scriptPubKeys.
- Prva transakcija je valjana transakcija kovanice (engl. coinbase transaction).

Provjera navedenih svojstava rezultira sustavom koji ekonomski jamči da se rad nastavlja na valjanim klijetkama, proširenih na mreži i prihvaćenih od većeg dijela mreže. Detaljnije, svakom rudaru je u interesu provjeravati valjanost klijetke, jer ako uključi nevaljanu klijetku i počne raditi na tom lancu vjerojatno će izgubiti nagradu i time vrijeme utrošeno za pronalazak valjanog šifrata klijetke.
U kontekstu lanca klijetki postoje tri vrste klijetki: klijetke glavne grane (engl. main branch blocks), klijetke bočne grane (engl. side branch blocks), te klijetke siročići (engl. orphan blocks).
Klijetke glavne grane su klijetke koje produžuju trenutnu glavnu granu lanca klijetki. Glavna grana lanca klijetki je grana koja ima najveću duljinu, tj. na kojoj je obavljen najveći rad. Klijetke bočne grane su klijetke koje referenciraju na roditeljsku klijetku koja nije trenutni vrh lanca klijetki. Klijetke siročići su klijetke koje referenciraju na roditeljsku klijetku koja nije poznata čvoru koji procesira klijetku.
Klijetke bočne grane se ne nalaze u glavnoj grani, ali ako se više rada obavi nad njima, tj. nove klijetke su izrudarene i referenciraju ih kao roditeljske klijetke, moguća je reorganizacija glavne grane. Reorganizacija lanca klijetki je situacija kada klijent otkrije novi težinski najduži dobro formiran lanac klijetki koji isključuje jednu ili više klijetki koje su dio lanca za kojeg je klijent mislio da je težinski najduži dobro formiran lanac klijetki. Ako se prilikom reorganizacije lanca klijetki, izgubi klijetka određenog rudara, on gubi nagradu za izrudarenu klijetku. Može se desiti da dva rudara izrudare dvije različite klijetke u malom vremenskom razmaku. Tada dio mreže prije sazna za jednu klijetku i prihvati jednu granu lanca, a ostatak mreže drugu. Pretpostavka je da će eventualno rudari jedne grane lanca pronaći novu klijetku prije druge skupine pa će i ostatak mreže prihvatiti duži lanac. Stoga se većina rudara vraća radu na pronalasku klijetki glavne grane lanca. Rudari imaju motivaciju podržati najduži lanac pošto je u njega uloženo najviše procesorske snage. Tako rudarenje rješava problem dvostruke potrošnje. Rudarenje također rješava problem distribucije novca u mreži bez centralnog tijela tako što rudar u klijetku uključuje transakciju kovanice kojom na svoju adresu šalje nagradu klijetke zajedno s transakcijskim naknadama, koja služi kao poticaj za uloženu procesorsku snagu. Nagrada klijetke izvorno je iznosila 50 bitcoina te se prepolavlja svakih 210,000 klijetki ili otprilike svakih 4 godine. Ovisno o ukupnoj količini procesorske snage koja sudjeluje u procesu rudarenja, za svakih

2016 klijetki dodanih na lanac klijetki protokol ažurira težinski cilj. Time se održava prosječno vrijeme potrebno za pronalazak valjane klijetke oko 10 minuta. Konkretno, ako je prosječno vrijeme rudarenja klijetke manje od 10 minuta, težinski cilj se poveća. Obratno, ako je vrijeme rudarenja klijetke veće od 10 minuta težinski cilj se smanji. Time mreža ima svojstvo skaliranja ovisno o procesorskoj snazi.

## 1.7 Nedostaci Bitcoinovog protokola

### Napad pedeset i jednog postotka rudara

Napad pedeset i jednog postotka rudara predstavlja situaciju gdje rudar ili skupina rudara posjeduje više od pedeset posto ukupne procesorske snage mreže i koristi ju za zlonamjerne radnje. U slučaju uspješnog napada pedeset i jednog postotka rudara, napadač nema mogućnost stvaranja novih bitcoina ili pripisivanja vlasništva nad bitcoinima koji mu nikada nisu pripadali, zato što čvorovi neće prihvatiti nevaljanu transakciju, a pošteni čvorovi nikada neće prihvatiti klijetku koja ju sadrži. Napadač zapravo ima mogućnost dvostruke potrošnje svojih bitcoina i sprečavanja potvrđivanja pojedinih transakcija.
Najefikasnija obrana od napada pedeset i jednog postotka rudara je decentralizacija rudara, tako da nitko nema velik dio procesorske snage. Pod pretpostavkom da je procesorska snaga dovoljno decentralizirana i poštena, jedini način za izvršavanje napada je proizvodnja novog hardvera, što je jako neisplativo s obzirom na dobivenu moć nad mrežom. Ukoliko bi određena skupina pridobila više od pedeset posto procesorske snage puno im je isplativija opcija da budu pošteni i rudarenjem novih klijetki dobivaju nagrade, nego da pokušaju kompromitirati sigurnost mreže.

### Centralizacija

Pošto mali rudari imaju jako slabe šanse da izrudare novu klijetku, oni se udružuju u rudarske bazene (engl. mining pools) kako bi zajedničkim procesorskim snagama povećali svoje šanse. Naposljetku, nagrade rudarskog bazena se dijele rudarima u omjeru uložene procesorske snage. Udruživanjem u rudarski bazen, rudar više nema potpunu kontrolu nad svojom procesorskom snagom, te rudarski bazen može iskoristiti njegovu procesorsku snagu za napad pedeset i jednog postotka rudara.
U trenutku pisanja ovoga rada, tri najveća rudarska bazena Bitcoina imaju oko 45% ukupne procesorske snage, dok 80% procesorske snage mreže Bitcoina pripada rudarima iz Kine, što je čini prilično centraliziranom.

## Skalabilnost

Kapacitet transakcija zabilježenih na lancu klijetki Bitcoina ograničen je veličinom klijetke od jednog megabajta i stvaranjem novih klijetki svakih 10 minuta u prosjeku. Konkretno, protokol Bitcoina u prosjeku može procesirati između 3 i 7 transakcija u sekundi. U slučaju da se velika količina transakcija treba procesirati, odabiru se one s najvećim transakcijskim naknadama, što može dovesti do ogromnih naknada.
Postoje dvije klase proširenja problema uskog transakcijskog grla na Bitcoinu. Prva su lančana proširenja koja uključuju povećanje veličine klijetke, smanjenje vremena potrebnog za rudarenje nove klijetke i druga tehnološka poboljšanja. U drugu klasu spadaju vanlančana proširenja kao što su vanlančani prijenosi i vanlančani kanali stanja. Vanlančana proširenja zovu se i rješenja drugog sloja (engl. second-layer solutions), a najpoznatiji mehanizam drugog sloja koji se koristi je Lightning Network.

## Nestajanje nagrada klijetki

Protokol Bitcoina određuje maksimalni broj bitcoina koji će ikada biti u opticaju na 21 milijun bitcoina. Nagrada za pronalazak nove klijetke je izvorno bila 50 bitcoina, te se prepolavlja svakih 210,000 klijetki. Stoga, nakon izrudarene 6,930,000. klijetke, nagrada za pronalazak nove klijetke će se smanjiti s 1 Satoshija ($10^{-8}$ bitcoina) na 0 Satoshija. Pošto se prepolavljanje nagrada događa u prosjeku svake 4 godine, nestajanje nagrada za rudarenje nove klijetke možemo očekivati oko 2140. godine. Tada će jedini poticaj rudarima za održavanje sigurnosti mreže Bitcoina biti transakcijske naknade.
Što će se dogoditi kada poticaji za rudarenje nestanu? Da bismo mogli odgovoriti na to pitanje potrebno je objasniti poslovanje rudarenja. Rudarenje bitcoina je jako složena i dinamična ekonomija s velikim brojem varijabli, koje međusobno utječu jedna na drugu. Profitabilnost svakog pojedinog rudara naviše određuje cijena struje koju plaća, trenutna i predviđena vrijednost bitcoina i udio procesorske snage koju posjeduje. Rudari računaju profitabilnost pojedinog hardvera i odlučuju na dnevnoj bazi hoće li s njim nastaviti rudariti. Ako određeni broj rudara isključi svoj hardver zbog neprofitabilnosti i time se npr. ukupna procesorska snaga prepolovi, rudari koji i dalje rudare postaju dvostruko profitabilniji nakon ažuriranja težinskog cilja.
Konačno, možemo zaključiti da se neće ništa posebno desiti nakon nestanka nagrada za rudarenje klijetki što se već prije nije desilo, zato što se rudari svakodnevno susreću s problemom profitabilnosti.

## 1.8 Skripte Bitcoina

Skriptni jezik Bitcoina je jezik napravljen naročito za Bitcoin, baziran je na stogu, nije potpun po Turingu i procesira se s lijeva na desno (obrnuta poljska notacija). Skripta Bitcoina je niz naredbi zabilježenih u svakoj transakciji koje opisuju koji korisnik i na koji način, može dobiti pristup nad sredstvima. Bitcoin transakcije koje na određenu adresu šalju sredstva koriste skriptu Pay-to-Public-Key-Hash, ali transakcije nisu ograničene samo slanjem na adresu. Štoviše, skripte zaključavanja mogu se napisati kako bi se izrazili složeniji uvjeti.
Opkodovi (engl. opcodes) predstavljaju listu svih skriptnih riječi. Neki od najkorištenijih optkodova su:

- OP_PUSHDATA1 - sljedeći bajt sadrži broj bajtova koji će se dodati na stog.
- OP_VERIFY - Ako gornja točka na stogu nije istina, označava transakciju kao nevažeću.
- OP_DUP - Duplicira gornju točku stoga.
- OP_EQUAL - Ako su ulazi jednaki vraća 1, inače 0.
- OP_EQUALVERIFY - Isto kao i OP_EQUAL, samo dodatno primijeni OP_VERIFY
- OP_RIPEMD160 - Ulaz je šifriran funkcijom RIPEMD160.
- OP_HASH256 - Ulaz je šifriran dva puta koristeći funkciju SHA256.
- OP_HASH160 - Ulaz je šifriran dva puta: prvo koristeći funkciju SHA256, a potom funkciju RIPEMD160.

### Primjer P2PKH skripte

Skripta standardne transakcije na adresu Bitcoina (P2PKH) ima sljedeći oblik:

```
scriptPubKey: OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
scriptSig: <sig> <pubKey>
```

Prvi dio skripte, scriptPubKey se nalazi u izlazu prethodno nepotrošene transakcije, a scriptSig se nalazi u ulazu transakcije koja troši. Skripta na sljedeći način vrši računanje na stogu:

1. Kombiniranje scriptSig i scriptPubKey.
   Stog: prazan.
   ```
   Skripta: <sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash>
            OP_EQUALVERIFY OP_CHECKSIG
   ```

2. Dodavanje konstanti na stog.
   Stog: $< sig >< pubKey >$
   `Skripta: OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`

3. Kopiranje gornje točke stoga.
   Stog: $< sig >< pubKey >< pubKey >$
   `Skripta: OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`

4. Šifriranje gornje točke stoga. $< pubHashA >$ je šifrat od $< pubKey >$.
   Stog: $< sig >< pubKey >< pubHashA >$
   `Skripta: <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`

5. Dodavanje konstante na stog.
   Stog: $< sig >< pubKey >< pubHashA >< pubKeyHash >$
   `Skripta: OP_EQUALVERIFY OP_CHECKSIG`

6. Provjera jednakosti gornje dvije točke stoga.
   Stog: $< sig >< pubKey >$
   `Skripta: OP_CHECKSIG`

7. Provjeren je potpis gornje dvije točke stoga.
   Stog: Istina
   Skripta: Prazna

## Primjer skripte zaključavanja bitcoina do određenog datuma

Skripta zaključavanja bitcoina do određenog datuma ima sljedeći oblik:

```
scriptPubKey: <expiry time> OP_CHECKLOCKTIMEVERIFY OP_DROP OP_DUP
              OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
scriptSig: <sig> <pubKey>
```

Skripta na sljedeći način vrši računanje na stogu:

1. Kombiniranje scriptSig i scriptPubKey.
   Stog: prazan.
   `Skripta: <sig> <pubKey> <expirytime> OP_CHECKLOCKTIMEVERIFY OP_DROP OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG`

2. Dodavanje konstanti na stog.
   Stog: < *sig* >< *pubKey* >< *expirytime* >
   Skripta: OP_CHECKLOCKTIMEVERIFY OP_DROP OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

3. Gornja točka stoga je provjerena s trenutnim vremenom ili visinom klijetke.
   Stog: < *sig* >< *pubKey* >< *expirytime* >
   Skripta: OP_DROP OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

4. Gornja točka stoga je izbrisana.
   Stog: < *sig* >< *pubKey* >
   Skripta: OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

5. Gornja točka stoga je kopirana.
   Stog: < *sig* >< *pubKey* >< *pubKey* >
   Skripta: OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

6. Gornja točka stoga je šifrirana.
   Stog: < *sig* >< *pubKey* >< *pubHashA* >
   Skripta: <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

7. Dodavanje konstante.
   Stog: < *sig* >< *pubKey* >< *pubHashA* >< *pubKeyHash* >
   Skripta: OP_EQUALVERIFY OP_CHECKSIG

8. Provjerena je jednakost gornje dvije točke stoga.
   Stog: < *sig* >< *pubKey* >
   Skripta: OP_CHECKSIG

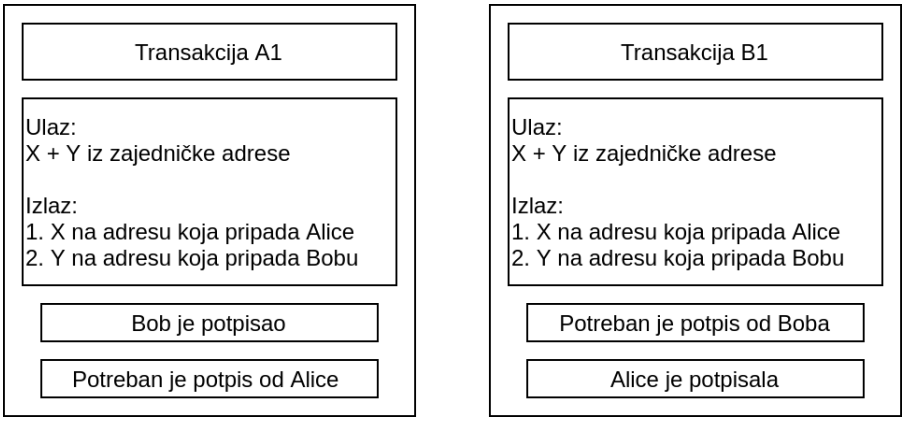
9. Provjeren je potpis gornje dvije točke stoga.
   Stog: Istina
   Skripta: Prazna

## 1.9 Lightning network

Lightning Network [11] stvara usmjerenu mrežu kanala plaćanja gdje je komunikacija između čvorova dvosmjerna. Dva korisnika koja žele koristiti Lightning Network moraju otvoriti višepotpisni kanal tako što zabilježavaju transakcije financiranja kanala na glavni lanac Bitcoina. Pošto plaćanja unutar kanala nisu zabilježena na lancu Bitcoina, naknade su drastično manje nego na samom lancu, te je propusnost veća. Stanje unutar kanala određeno je zadnjom transakcijom koja je potpisana od oba korisnika kanala. Ako korisnici žele finalizirati transakciju, oba potpisuju transakciju izlaza i time sve prijašnje transakcije unutar kanala postaju nevaljane. Lightning Network ne zahtijeva da se obje strane slože oko zatvaranja kanala. U svakom trenutku, obje strane imaju opciju zatvaranja kanala tako što objave zadnju transakciju potpisanu od obje strane na glavni lanac. Ako dva korisnika žele slati sredstva jedan drugome, a nemaju otvoren kanal koji ih direktno povezuje, slanje je moguće ako postoji put platnih kanala između njih u mreži.

### Platni kanal

Neka su Alice i Bob korisnici koji žele otvoriti platni kanal. Prvo moraju napraviti transakcije koje prenose njihova sredstva na zajedničku adresu za depozit. Sredstva sa zajedničke adrese mogu biti potrošena samo kad je transakcija potpisana privatnim ključevima oba korisnika. Prije nego pošalju sredstva na zajedničku adresu, moraju napraviti transakcije povrata sredstava (A1 i B1), jer jedna strana može odbiti surađivati, te druga strana gubi pristup poslanim sredstvima. Ulaz transakcija povrata je izlaz transakcije zajedničke adrese. Potom međusobno potpisuju transakcije povrata i zamjene ih. Tek sada mogu sigurno objaviti početnu transakciju depozita na glavni lanac. Sada, ako jedna strana odbije surađivati, druga strana treba potpisati svoju transakciju i zabilježiti ju na glavni lanac.
Da bi korisnici napravili plaćanje u kanalu, obje strane trebaju prepraviti transakciju A1 i B1. Recimo da Bob želi poslati Z bitcoina, stoga Alice prihvaća primiti $X + Z$ a Bob $Y - Z$ bitcoina. Kako su transakcije A1 i B1 promijenjene, sada su zadnje transakcije A2 i B2. Ovakva zamjena nije sigurna, pošto jedna strana može objaviti neku staru transakciju na glavni lanac. Taj problem riješen je uvođenjem jednokratnih privatnih ključeva svaki puta kada korisnici naprave novu transakciju. Korisnici međusobno otkrivaju svoje stare ključeve, te generiraju nove. Nijedna strana neće prihvatiti transakciju ako druga strana nije otkrila svoj prethodni ključ. Ako jedna strana ima prethodni privatni ključ druge strane, ona može potrošiti sva sredstva druge strane samo u slučaju da druga strana pokuša zabilježiti staru transakciju na glavni lanac. Također, potrebna je još jedna modifikacija koja oslobađa sredstva korisnika koji zatvara kanal, tek nakon određenog broja klijetki (T).
Ako Alice potpiše i objavi svoju transakciju, koja je prethodno bila potpisana od strane Boba, dva su moguća scenarija:




Slika 1.4: Transakcije povrata A1 i B1

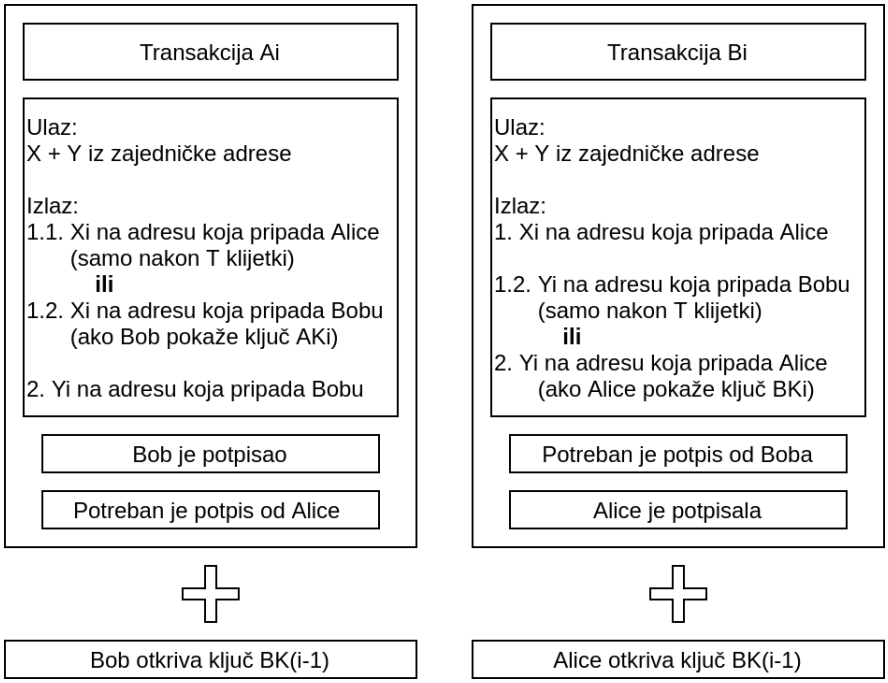
1. Alice pokušava zabilježiti staru transakciju umjesto posljednje. Bob primjećuje da ga Alice pokušava prevariti, te pošto Bob poznaje privatni ključ kojeg je objavila, on može sva sredstva s njene adrese. Očito, ovaj scenarij nije pogodan za Alice.

2. Alice pokušava zabilježiti posljednju transakciju. U ovom slučaju Bob nema zadnji privatni ključ koji pripada Alice, pa je zatvaranje kanala valjano. Konačno, Bob bi trebao dobiti svoj dio sredstava u idućoj klijetki, a Alice tek nakon T klijetki.

Broj klijetki T potreban za otključavanje sredstava korisnika koji je inicirao zatvaranje kanala unaprijed je dogovoren među čvorovima. To vrijeme dovoljno je da drugi korisnik kanala provjeri jeli zatvaranje kanala pošteno. Da bi korisnik otkrio prevaru nužno je da ne bude van mreže duže od T klijetki.

## Platni kanal s posrednikom

U slučaju da Alice i Bob nemaju otvoren direktan platni kanal, ali oboje imaju otvoren kanal s Charliejem, oni mogu međusobno slati sredstva preko posrednika. Stoga, Alice može poslati sredstva Charlieju, a on ih prosljeđuje Bobu i obratno. Problem nastaje ako Charlie ne želi poslati sredstva Bobu, koja je primio od Alice. Taj problem je riješen korištenjem tajnog broja $e$. Bob prvo kreira tajni broj $e$, izračuna njegov šifrat $H_e$, te šalje šifrat Alice. Alice kreira transakciju AC kojom šalje sredstva Charlieju samo u slučaju da Charlie zna tajni broj $e$. Charlie kreira sličnu transakciju CB koja šalje sredstva Bobu samo ako mu Bob otkrije tajni broj $e$. Tek sada Bob vidi da će moći primiti sredstva te mu otkriva tajni broj $e$. Sada Charlie može dokazati Alice da je on poslao sredstva Bobu, tako što otkrije tajni broj $e$, te prima sredstva transakcije AC.




Slika 1.5: Transakcije povrata Ai i Bi

Postoji scenarij kada Bob odluči varati tako što ne otkrije tajni broj Charlieju. U tom slučaju, ako Bob želi povući svoja sredstva mora zatvoriti kanal s Charliejem i zabilježiti transakciju CB na glavnom lancu, ali kako bi mogao trošiti sredstva mora otkriti tajni broj *e* kojeg će i Charlie vidjeti.

## Usmjeravanje

Kako bi plaćanje unutar Lightning mreže bilo moguće, nužno je postojanje barem jednoga puta povezanih platnih kanala između platitelja i primatelja. Problem usmjeravanja (engl. routing) je problem pronalaska optimalnog puta između dva čvora u mreži platnih kanala. Cijena puta određena je raznim dinamičkim faktorima kao što su: topologija mreže, propusnost kanala, dostupnost čvorova i naknade posrednika.
Problem usmjeravanja još uvijek nije riješen. Problem usmjeravanja ne ovisi o sržnoj specifikaciji Lightning Networka ni o konstrukciji platnih kanalna. Trenutno je mreža jako mala, pa pronalazak rješenja problema usmjeravanja nije nužan za normalno funkcioniranje mreže.

## Nedostaci Lightning Networka

Najpopularnija kritika Lightning Networka je centralizacija koja krši jedan od glavnih principa mreže Bitcoina. Naime, prilikom stvaranja platnih kanala, novi čvorovi se prirodno žele povezati s velikim čvorištima, koji imaju dosta sredstava i dobro su povezani s ostatkom mreže. Problem centralizacije mreže može se riješiti automatskim stvaranjem platnog kanala, koristeći algoritam koji isključuje opciju odabiranja čvora za spajanje.
Lightning Network se ne može koristiti s računima koji nisu uvijek spojeni na mrežu, tzv. engl. cold storage. Sredstva su zaključana u platnim kanalima i kontrolirana su od strane računa koji su uvijek spojeni na mreži. Nije točno da svi čvorovi u mreži moraju biti spojeni u svakom trenutku, jer ako pojedini čvor nije spojen na mrežu, drugi čvorovi mogu nadgledati njegove kanale i reagirati ako se dogodi zlonamjerna radnja.
Transakcije na Lightning Networku nisu sigurne kao transakcije na glavnom lancu Bitcoina, pa se preporučuje da se veći prijenosi ne obavljaju na Lightning Networku.

# Poglavlje 2

# Ethereum

## 2.1 Uvod

Ethereum [4] je globalna platforma otvorenog kôda, bazirana na tehnologiji lanaca klijetki koja podržava funkcionalnost pametnih ugovora. Projekt je osmislio Vitalik Buterin, 2013. godine, a mreža je zaživjela u srpnju 2015. godine. Namjera Ethereuma je proširiti mogućnosti dotada skromnog skriptnog jezika Bitcoina do programskog jezika potpunog po Turingu. Ethereum omogućuje programerima da naprave proizvoljne aplikacije na bazi konsenzusa. Ether (ETH) je matična valuta Ethereuma, koja predstavlja digitalni novac i koristi se za plaćanje transakcijskih naknada. Pametni ugovori su programi na višoj razini koji se kompajliraju u Ethereumovoj virtualnoj mašini (EVM) i uključuju u Ethereumov lanac klijetki. Najpoznatiji jezik za Ethereumove pametne ugovore je Solidity (sintakse slične jezicima C i JavaScript), a koriste se još i Serpent (sličan Pythonu), LLL (jezik niže razine).

## 2.2 Računi na Ethereumu

Računi na Ethereumu su objekti koji čine stanje mreže Ethereuma. Svaki račun je predstavljen adresom dugačkom 20 bajtova. Općenito, postoje dvije vrste računa: računi u vanjskom vlasništvu i ugovorski računi. Računi u vanjskom vlasništvu su kontrolirani privatnim ključevima, dok su ugovorski računi kontrolirani njihovim kôdom ugovora. Računi u vanjskom vlasništvu nemaju kôd, ali mogu poslati poruku s računa tako da stvore i potpišu transakciju. Svaki put kad ugovorski račun primi poruku njegov kôd se aktivira, te dozvoljava čitanje, pisanje u lokalni spremnik i slanje drugih poruka ili stvaranje ugovora.

Račun na Ethereumu se sastoji od 4 polja:

- Jednokratan broj (engl. nonce) - osigurava da se svaka transakcija može procesirati samo jednom.
- Stanje ethera - predstavlja trenutno stanje ethera na računu.
- Programski kôd ugovora računa - postoji samo kod ugovorskih računa.
- Spremnik - svaki račun sadrži spremnik koji je zadano prazan.

Jednokratan broj računa osigurava procesiranje svake transakcije točno jednom, tako što se njegova vrijednost poveća za 1 svaki puta kada račun pošalje transakciju. Jednokratni broj računa mora biti strogo rastući, tj. ne može transakcija s jednokratnim brojem 1 biti zabilježena prije transakcije s jednokratnim brojem 0. Dodatno, nije moguće preskakanje vrijednosti ili povećanje jednokratnog broja za više od 1.

## 2.3 Poruke i transakcije

Poruke u mreži Ethereuma su slične transakcijama u mreži Bitcoina. Ključna razlika je mogućnost kreiranja poruke od strane ugovora u mreži Ethereuma, dok transakcija Bitcoina može biti kreirana samo izvana, tj. posjedovanjem privatnog ključa. Ethereumove poruke imaju određenu opciju sadržavanja podataka. Također, ako je primatelj Ethereumove poruke ugovorski račun, on ima mogućnost vratiti odgovor. Time poruke Ethereuma obuhvaćaju koncept funkcija.
Ethereumova transakcija predstavlja potpisani paket podataka koji sadržava poruku poslanu od računa u vanjskom vlasništvu. Transakcije sadrže: adresu primatelja poruke, potpis pošiljatelja, količinu Ethera, podatke koje šalje, STARTGAS i GASPRICE.
Kako bi se spriječile beskonačne petlje, svaka transakcija mora odrediti ograničenje broja računarskih koraka koji su potrebni za izvršenje kôda, uključujući početnu poruku i sve poruke koje se pokreću tijekom izvršenja kôda. To ograničenje je određeno STARTGAS varijablom, a GASPRICE je naknada koju rudar dobiva po odrađenom računarskom koraku.
Ako izvršenje transakcije potroši resurse određene STARTGAS varijablom (”ostane bez goriva”), sve promjene stanja se vraćaju, osim plaćanja naknada. S druge strane ako se izvršenje zaustavi s određenom količinom resursa, ostatak se vraća pošiljatelju. Ako primatelj transakcije nije ugovor, tada je ukupna transakcijska naknada jednaka danom varijablom GASPRICE pomnoženom s veličinom transakcije u bajtovima, a podaci poslani s transakcijom se ignoriraju.

Proces Ethereumove funkcije promjene stanja može se opisati sljedećim algoritmom.

**Algoritam 3:** Proces Ethereumove funkcije promjene stanja

1 Provjeri je li transakcija dobro formirana, tj. ima li točan broj vrijednosti, je li potpis valjan, poklapa li se jednokratni broj jednokratnom broju računa pošiljatelja. Ako nije vrati grešku.
2 Odredi adresu pošiljatelja iz potpisa i izračunaj transakcijsku naknadu STARTGAS * GASPRICE. Oduzmi naknadu s računa pošiljatelja i povećaj pošiljatelju jednokratni broj. Ako pošiljatelj nema dovoljno sredstava vrati grešku.
3 Inicijaliziraj GAS=STARTGAS i skini određenu količinu goriva po bajtu transakcije.
4 Pošalji vrijednost transakcije s računa pošiljatelja na račun primatelja. Ako račun primatelja ne postoji stvori ga. U slučaju da je račun primatelja ugovor, izvrši kôd ugovora do kraja ili dok ne ponestane goriva.
5 Ako je prijenos vrijednosti neuspješan jer pošiljalac nije imao dovoljno sredstava ili je izvršavanje kôda potrošilo svo gorivo, vrati sve promjene stanja osim naplate naknada i dodaj naknade na račun rudara.
6 Inače, pošalji ostatak goriva pošiljatelju i naknade za iskorišteno gorivo rudaru.

## 2.4 Izvršenje kôd

Kôd Ethereumovih ugovora napisan je u nisko-razinskom bajtkodnom jeziku baziranom na stogu, a referira se kao Ethereumov kôd virtualne mašine (engl. Ethereum virtual machine code) ili kraće EVM kôd. Kôd se sastoji od niza bajtova, a svaki bajt predstavlja jednu operaciju. Izvršavanje kôda je zapravo beskonačna petlja koja se sastoji od ponavljajućeg izvršavanja operacija na trenutnom programskom brojaču, koji počinje od nule i povećava se za jedan sve do kraja kôda ili dok se ne vrati greška.
Spomenute operacije imaju pristup trima vrstama prostora za spremanje podataka:

- Stog - u stog se mogu spremati vrijednosti veličine 32 bajta.
- Memorija - beskonačno proširiv niz bajtova.
- Dugotrajni spremnik ugovora - spremnik koji funkcionira na principu ključ/vrijednost gdje su ključ i vrijednost veličine 32 bajta. Za razliku od stoga i memorije, koji se brišu kad računanje završi, dugotrajni spremnik zadržava spremljene podatke i nakon završetka računanja.

Kôd ugovora ima pristup vrijednosti, adresi pošiljatelja, podacima dolazeće transakcije i podacima zaglavlja klijetke. Stanje Ethereumove virtualne mašine je uređena osmorka: (stanje_klijetke, transakcija, poruka, kôd, memorija, stog, programski_brojač, gorivo).

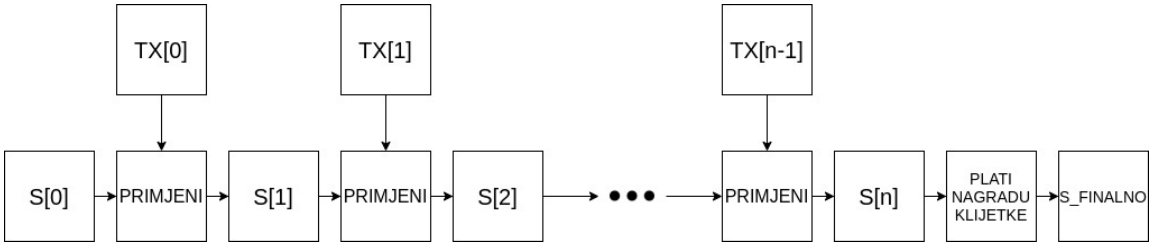
Prva četiri parametra osmorke su invarijantna s ugovorom, dok druga četiri, zajedno s dugotrajnim spremnikom opisuju stanje ugovora. U svakom krugu izvršavanja, trenutna naredba se pronalazi na bajtu pozicije programski_brojač. Svaka naredba ima poseban utjecaj na uređenu osmorku stanja EVMa.
Na primjer, operacija SUB uzima dvije točke s vrha stoga, oduzima gornju od donje i stavlja dobiveni rezultat na stog. Nakon toga smanjuje gorivo za 3 i povećava programski_brojač za 1.

## 2.5 Rudarenje i lanac klijetki

Iako je lanac klijetki Ethereuma u mnogim pogledima sličan Bitcoinovom, razlikuje se u sadržaju klijetke. Ethereumove klijetke, za razliku od Bitcoinovih, sadrže kopiju liste transakcija, zadnje stanje mreže, redni broj klijetke i težinu. Vrijeme potrebno za pronalazak nove klijetke u Ethereumovom lancu varira između 10 i 30 sekundi. Značajno manje vrijeme za pronalazak nove klijetke znači da je procesiranje transakcija gotovo trenutno.
Za razliku od algoritma Bitcoina za podešavanje težinskog cilja, algoritam Ethereuma po potrebi mijenja težinski cilj u svakoj klijetki te uzima u obzir i računarsku snagu mreže.

```
vrijeme_klijetke = vremenska_oznaka_trenutne_klijetke -
                   vremenska_oznaka_roditeljske_klijetke
težinski_cilj_trenutne_klijetke = težinski_cilj_roditeljske_klijetke +
                   (težinski_cilj_roditeljske_klijetke // 2048) *
                   max(1 - (vrijeme_klijetke // 10), -99) +
                   int(2**((trenutni_redni_broj_klijetke // 100000) - 2))
```

Operator // predstavlja cjelobrojno dijeljenje, a operator ** potenciranje. Prvi dio jednadžbe za ažuriranje težinskog cilja povećava težinski cilj ako je vrijeme pronalaska klijetke strogo manje od 10 sekundi, a smanjuje težinski cilj ako je za pronalazak klijetke bilo potrebno barem 20 sekundi. Drugi dio jednadžbe uvijek povećava težinski cilj.




Slika 2.1: Promjene stanja

Ethereumov algoritam valjanosti klijetke sadrži sljedeće korake:

**Algoritam 4:** Ispitivanje valjanosti klijetke

1. Provjeri postoji li referencirana prethodna klijetka i je li valjana.
2. Provjeri je li vremenska oznaka klijetke veća od prethodne referencirane klijetke i manja od 15 minuta u budućnosti.
3. Provjeri valjanost broja klijetke, težine, transakcijskog korijena, korijena ujakove klijetke i limit goriva.
4. Provjeri valjanost dokaza o radu trenutne klijetke.
5. Postavi početno stanje S[0] na korijensko stanje prethodne klijetke STANJE_KORIJENA.
6. Neka je TX lista transakcija klijetke, koja sadrži n transakcija. Za sve $i \in [0, n-1]$, na stanje $S[i]$ primijeni transakciju $TX[i]$. Ako bilo koja primjena vrati grešku ili ukupno gorivo iskorišteno u klijetki prijeđe GASLIMIT, vrati grešku.
7. Neka je S_FINALNO jednako stanju $S[n]$ na koje su primijenjene nagrade rudaru za pronađenu klijetku.
8. Provjeri je li S_FINALNO jednako stanju STANJE_KORIJENA, ako je klijetka je valjana.

Iako ovaj pristup zahtjeva spremanje čitavog stanja u svakoj klijetki, efikasnost je usporediva s Bitcoinovom. Stanje je spremljeno u strukturi stabla i svaka nova klijetka mijenja samo mali dio stabla. Dvije susjedne klijetke imaju većinu stabla jednaku, te podaci jednom spremljeni se mogu referencirati više puta koristeći šifrate podstabala. Za razliku od Bitcoina, za spremanje transakcija Ethereumov protokol koristi modifikaciju Merkleovog stabla, poznatog kao Patricia stablo, koje uz mijenjanje čvorova dopušta dodavanje i brisanje čvorova.

## Ethash

Ethash [12] je algoritam dokaza o radu implementiran za Ethereum 1.0. Ethash predstavlja najnoviju verziju Dagger-Hashimoto algoritma, koji je predstavio Vitalik Buterin i Ethereumov tim. Pošto se izvorna verzija oba algoritma (Dagger i Hashimoto) za dokaz o radu dosta promijenila, više nije bilo prikladno izvorno ime algoritma.
Motivacija za razvoj novog algoritma za dokaz o radu je bila sljedeća:

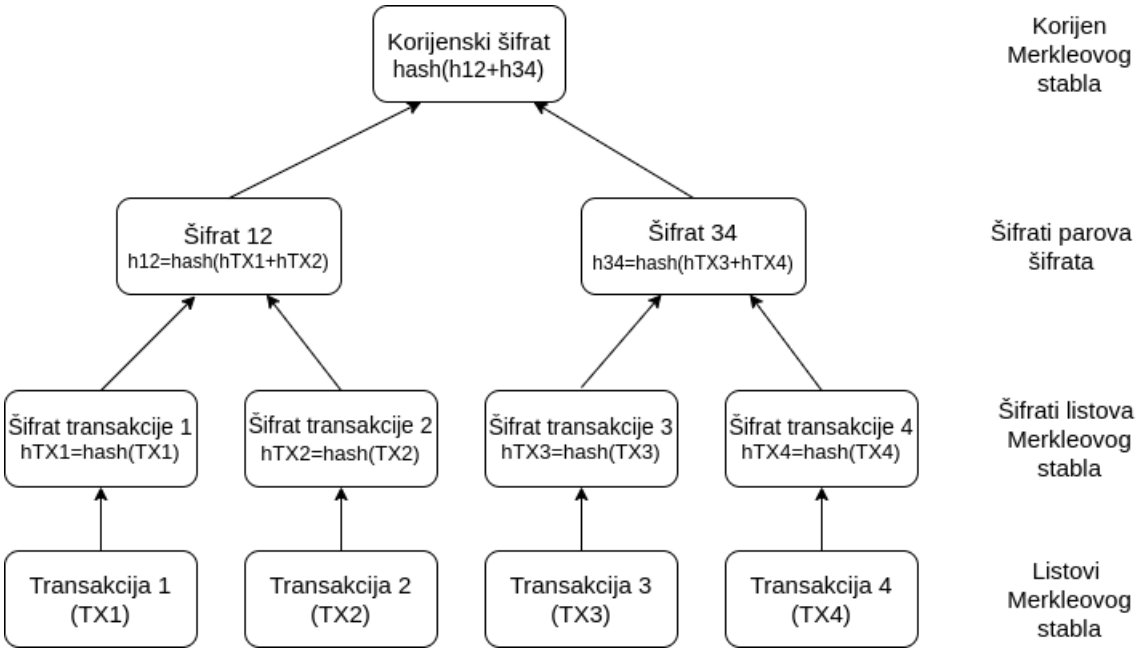
- ASIC-otpornost
- Lagani klijent (engl. light client) dobiva mogućnost verifikacije.
- Skladištenje potpunog lanca klijetki

Razvojem integriranih krugova za specifičnu primjenu (engl. application-specific integrated circuit) ili skraćeno ASIC, manji rudari su postali neprofitabilni, te je rudarenje na mreži Bitcoina postalo dosta centralizirano. Ethereumov protokol rješava taj problem razvijanjem algoritma za dokaz o radu koji je otporan na ASIC uređaje, tako da se pored ponavljajućih operacija povećavanja šuma i šifriranja, u algoritam dokaza o radu dodaju memorijski teške operacije. Memorijski teške operacije su operacije za čije izvršavanje, potrebno vrijeme je ovisno o količini memorije potrebne za čuvanje podataka. Stoga, velika većina napora rudara će biti potrošena na čitanje DAGa, umjesto na računanje podataka preuzetih iz njega. Ethash koristi skup podataka izvorne veličine 1 GB poznat kao Ethash DAG i predmemoriju za lagane klijente veličine 16 MB. Ti podaci se iznova obnavljaju svake epohe, tj. svakih 30,000 klijetki, te linearno rastu kako se lanac klijetki povećava. Proces rudarenja podrazumijeva uzimanje slučajnog djela podataka iz DAGa za generiranje miješanih šifrata koristeći transakcije i podatke o primanju, zajedno s kriptografskim šumom kako bi dobili šifrat ispod dinamične granice težinskog cilja.
Ethash algoritam je definiran kao: $(m, n) = PoW(H_{\not n}, H_n, d)$ gdje je $m$ miješani šifrat, koji zajedno s šumom $n$ dokazuje da je klijetka izvršila dovoljno računanja. $H_{\not n}$ je zaglavlje nove klijetke bez miješanog šifrata i šuma, $H_n$ je šum zaglavlja klijetke, a $d$ je DAG.
Verifikacija se može izvršiti korištenjem male količine memorije, tako da se iz predmemorije generiraju samo dijelovi DAGa potrebni za verifikaciju.

## 2.6 Merkleovo i Patricia stablo

Merkleovo stablo je vrsta binarnog stabla, kojemu su listovi šifrati a čvorovi koji nisu listovi su šifrati svoje djece. Takva stabla osiguravaju brzu verifikaciju uključivosti transakcije u određenu klijetku, efikasno spremanje velike količine podataka, te cjelokupnost podataka (jedan šifrat predstavlja sve transakcije). U slučaju promjene bilo koje od transakcija sadržanih u stablu, promjenit će se i korijen stabla. Bitcoin koristi Merkleova stabla za spremanje svih transakcija pojedine klijetke. Merkleova stabla zapravo omogućuju verifikaciju svih transakcija bez da je tijelo transakcije uključeno u zaglavlju klijetke. U klijetki Bitcoina nalazi se korijen Merklovog stabla svih transakcija. Vrijednost korijena Merkleovog stabla je šifrat, koji se dobije šifriranjem parova šifrata transakcija, te nadalje šifriranje dobivenih šifrata sve dok se ne dobije samo jedan šifrat, tj. korijen stabla. Kada je korijen Merkleovog stabla pojedine klijetke poznat, svaka promjena transakcije sadržane u stablu mijenja šifrat njezinog roditeljskog čvora, te stoga mijenja i korijen stabla.
Koristeći Merkleova stabla moguće je brzo i sigurno provjeriti jeli proizvoljna transakcija zabilježena u $n$-toj klijetki. Potrebno je dohvatiti sve transakcije $n$-te klijetke, rekonstruirati šifrate cijelog Merkleovog stabla i te usporediti dobiveni šifrat korijena stabla s korijenom koji je zabilježen u zaglavlju klijetke. Ako su šifrati korijena Merkleovog stabla jednaki, možemo biti sigurni da je transakcija sadržana u $n$-toj klijetki. Dodatno, ako je poznato

gdje je točno transakcija sadržana u stablu i ako su poznati šifrati grana, za provjeru valjanosti transakcije dovoljno je provjeriti šifrate grane koja sadrži danu transakciju. Drugim riječima, provjera počinje s računanjem šifrata transakcije i njene susjede transakcije u stablu (ako postoji), te provjere dobivenog šifrata s očekivanom vrijednošću dohvaćenom iz stabla. Potom se dohvati susjedni čvor dobivenog šifrata, te se izračuna šifrat njega i susjednog čvora i usporedi s očekivanom vrijednošću. Proces se nastavlja sve dok se ne izračuna i verificira korijen ili dok se dobije šifrat koji je različit od očekivanog tijekom procesa šifriranja. U slučaju dobivanja šifrata koji nije jednak očekivanom, možemo biti sigurni da transakcija nije sadržana u stablu na danoj poziciji u klijetki. Time se broj šifriranja i uspoređivanja smanjuje na $\lceil log_2 m \rceil + 1$, gdje je $m$ broj transakcija sadržanih u klijetki, tj. Merkleovom stablu.




Slika 2.2: Merkleovo stablo

## Patricia stablo

Za razliku od protokol Bitcoina, protokol Ethereuma koristi Patricia stabla za spremanje transakcija, potvrda (engl. receipts) i stanja. Merkleova stabla su pogodna za spremanje podataka koji se nikad mijenjaju. S druge strane, stablo stanja mreže Ethereuma je mapiranje, gdje su ključevi adrese, a vrijednosti su deklaracije računa, koje sadrže stanje računa, šum i kôd ako je račun ugovor. Stanje se mijenja često, jer se računi dodaju, stanja se

mijenjaju i šumovi računa se povećavaju. Patricia stablo (engl. Patricia trie) je vrsta stabla koja sprema podatke u svakom čvoru za razliku od Merkleovog stabla koje sprema podatke samo u listovima. Pošto su podaci spremljeni u svim čvorovima, omogućene su efikasne operacije izmjene, ubacivanja i brisanja, te brzo računanje šifrata korijena stabla nakon operacija. Još jedna prednost Patricia stabla je činjenica da se točke čiji ključevi počinju s istim vrijednostima nalaze blizu jedan drugome u stablu.

## 2.7 Primjene pametnih ugovora

Najosnovnija podjela pametnih ugovora po principu primjene je sljedeća: financijske, polufinancijske i nefinancijske aplikacije.
Financijske aplikacije uključuju financijske derivate, financiranja, sporedne valute, depozitne račune, oporuke, ugovore zaštite. Kod polufinancijskih aplikacija novac je uključen, ali postoji i nemonetarna strana ugovora, npr. automatska isplata nagrade za rješenje računarskih problema. Nefinancijske aplikacije isključuju novčani segment, npr. decentralizirano glasanje i vladavina.

### Sustavi tokena

Sustavi tokena su jedna od najzastupljenijih primjena pametnih ugovora na Ethereumu. Sustav tokena predstavlja bazu podataka s najmanje jednom operacijom: računu A oduzmi X jedinica i dodaj ih na račun B, pod uvjetom da račun A odobrava transakciju i prije transakcije račun A ima najmanje X jedinica. Tokeni imaju razne upotrebe, od predstavljanja američkog dolara ili zlata do poticajnih sustava, nedjeljivih kupona i pametnog vlasništva. Jednostavna implementacija sustava tokena:

```
od = poruka.pošiljatelj
do = poruka.podaci[0]
vrijednost = poruka.podaci[1]

ako ugovor.spremnik[od] >= vrijednost:
   ugovor.spremnik[od] = ugovor.spremnik[od] - vrijednost
   ugovor.spremnik[do] = ugovor.spremnik[do] + vrijednost
```

Navedenoj implementaciji nedostaje početna raspodjela tokena, mogućnost provjere stanja adrese od strane drugih ugovora i par rubnih slučajeva. Jedan od najvećih problema vezanih za financijske pametne ugovore je volatilnost bazne valute Ethera. Najprihvaćenije rješenje navedenog problema je imovina podržana kod izdavatelja imovine. Drugim riječima, izdavatelj imovine napravi pomoćnu valutu u kojoj ima pravo izdati, uništiti jedince imovine i svakom korisniku koji izvanmrežno preda jedinicu osnovne valute (američki dolar, zlato

ili dr.) pripiše vlasništvo nad digitalnom jedinicom valute. Izdavatelj obeća predati jednu baznu jedincu svakom korisniku koji pošalje nazad digitalnu jedinicu valute. Navedeni mehanizam omogućuje svakoj nekriptografskoj jedinici da bude uvedena u sustav lanca klijetki. Važno je napomenuti da ako je izdavatelj nepošten cijeli sustav se raspada.

## Sustav identiteta

Jedna od prvih alternativnih kriptovaluta nakon Bitcoina je bila Namecoin [4], koji je koristio tehnologiju sličnu Bitcoinovoj za osiguranje sustava registracije imena. Namecoin je omogućio da korisnici zabilježe svoje ime i druge podatke u javnom lancu klijetki. Glavna primjena je bila preslikavanje domena i IP adresa. Osnovni pametni ugovor koji pruža usluge slične Namecoinovim izgleda:

```
domena = poruka.podaci[0]
adresa = poruka.podaci[1]

ako je prazan ugovor.spremnik[domena]:
   ugovor.spremnik[domena] = adresa
```

Navedeni ugovor provjerava je li domena već pohranjena u memoriji ugovora i ako nije registrira ju i njenu pripadajuću IP adresu. Jednom kada je domena registrirana, ona ostaje zauvijek u memoriji. Ugovoru se može dodati funkcija koja će omogućiti drugim ugovorima da dohvate podatke domene. Dodatno moguće je dodati mehanizam koji omogućuje vlasniku da promjeni podatke ili pripiše vlasništvo nad njima drugom korisniku.

## Decentralizirano pohranjivanje datoteka

Ethereumovi ugovori omogućuju razvoj decentraliziranog sustav za pohranu datoteka, gdje korisnici koji iznajmljuju svoju neiskorištenu lokalnu memoriju mogu zaraditi naknadu. Korisnik koji želi koristiti uslugu pohranjivanja datoteka podijeli podatke u blokove, te šifrira svaki blok iz sigurnosnih razloga i stvori Merkleovo stablo od njega. Pametni ugovor svakih N klijetki, koristeći šifrat prethodne klijetke, izabere na slučajan način indeks Merkleovog stabla i daje određenu količinu ethera prvom korisniku koji pošalje transakciju s pojednostavljenim verifikacijskim dokazom o vlasništvu bloka na određenom indeksu u stablu. Kada korisnik poželi skinuti svoju datoteku, koristi kanal mikro plaćanja gdje plaća malu količinu ethera za svaka 32 kilobajta datoteke koja skine. Korisnik može biti siguran da je njegova datoteka pohranjena barem na jednom čvoru, ako ugovor i dalje isplaćuje naknadu. U slučaju da ugovor prestane isplaćivati naknadu, to znači da nitko ne pohranjuje određenu datoteku.

## Decentralizirane autonomne organizacije (DAO)

Decentralizirana organizacija je virtualni entitet koji ima određeni broj članova, koji sa suglasjem od 67% članova mogu trošiti sredstva organizacije i mijenjati njen kôd. Organizacija može trošiti sredstva na nagrade, plaće i internu valutu. Decentralizirane autonomne organizacije imaju obilježja tradicionalne tvrtke ili neprofitne organizacije. DAO se može implementirati kao pametni ugovor koji sadrži samomijenjajući kôd koji se mijenja ako su dvije trećine članova suglasne s promjenom. Iako je kôd ugovora nepromjenjiv, promjenjivi dijelovi se mogu implementirati zasebnim ugovorima. Takav ugovor bi imao tri tipa transakcije, koji se razlikuju u podacima koji su priloženi transakciji:

- $[0, i, K, V]$ - registracija prijedloga na indeksu $i$ za promjenu adrese u indeksu pohrane $K$ u vrijednost $V$
- $[0, i]$ - registraciju glasa za prijedlog na indeksu $i$.
- $[2, i]$ - dovršetak prijedloga na indeksu $i$ ako je skupljeno dovoljno glasova.

Ugovor održava popis svih otvorenih prijedloga o promjeni pohrane, zajedno s listom adresa koje su glasale za njih. Dodatno, ugovor sadrži i listu svih adresa članova organizacije. Kada za bilo koji prijedlog promjene pohrane glasa najmanje dvije trećine članova, transakcija za dovršetak prijedloga može izvršiti promjenu. Alternativni model je decentralizirana firma, gdje svaki član može imati određeni broj dionica. U tom slučaju, suglasnost dvije trećine dionica je nužna za promjenu pohrane.

## Ostale primjene pametnih ugovora

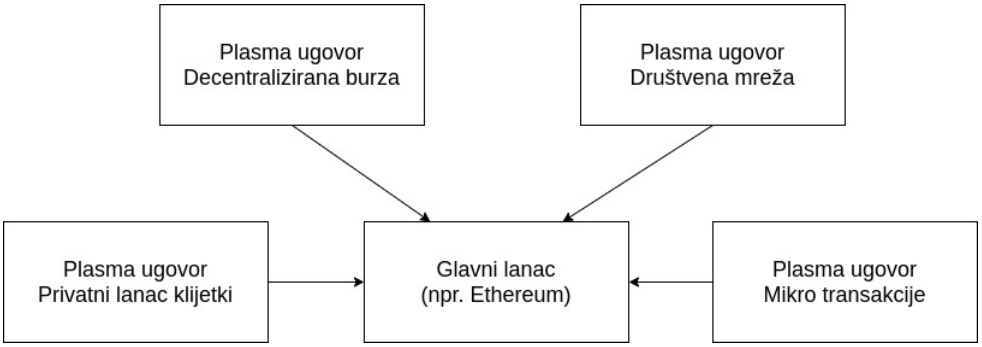
Pametna višepotpisna udruga predstavlja ugovor koji omogućuje definiranje pravila i uvjeta o trošenju zajedničkih sredstava. Slično Bitcoinovim višepotpisnim transakcijama, samo puno fleksibilnije rješenje koje omogućuje postavljanje detaljnijih pravila. Također, Ethereumov višestruki potpis je asinkron, tj. dva korisnika mogu registrirati svoje potpise na lanac klijetki u različitim vremenima, te zadnji potpis će automatski poslati transakciju.
Decentralizirano ažuriranje podatka je protokol koji omogućuje pružanje proizvoljnog broja vrijednosti, npr. cijene Ethera u odnosu na američki dolar, temperature u Zagrebu i dr. Protokol nagrađuje sve korisnike koji su poslali vrijednosti između 25. i 75. percentila. Interes svih sudionika je da pošalju što točniju vrijednost kako bi dobili nagradu, a protokol može imati točnu vrijednosti na lancu klijetki.
Koristeći pametne ugovore moguća su osiguranja u poljoprivredi. Nužna je implementacija financijskog derivata koji koristi podatke o vremenskim oborinama. Poljoprivrednik može kupiti derivat koji vrši isplatu obrnuto u odnosu na količinu oborina na određenom području. Ako dođe do suše poljoprivredniku se automatski isplaćuje osiguranje, u suprotnom ako je padalo dovoljno kiše, poljoprivrednik ne dobiva ništa.

Decentralizirano kockanje moguće je implementirati na mreži Ethereuma, a najjednostavniji protokol na kojem je kockanje bazirano je ugovor o razlici šifrata zadnje i sljedeće klijetke. Povrh tog protokola moguće je izgraditi složenije protokole za kockanje s malim naknadama i nemogućnošću varanja.
Računarstvo u oblaku se danas koristi više nego ikada prije. Ethereumova virtualna mašina se može iskoristiti za stvaranje verifikacijskog sustava za računanje problema koji ne uključuju komunikaciju među procesima. Sustav je pouzdan zahvaljujući sigurnosnim depozitima.

## 2.8 Plasma

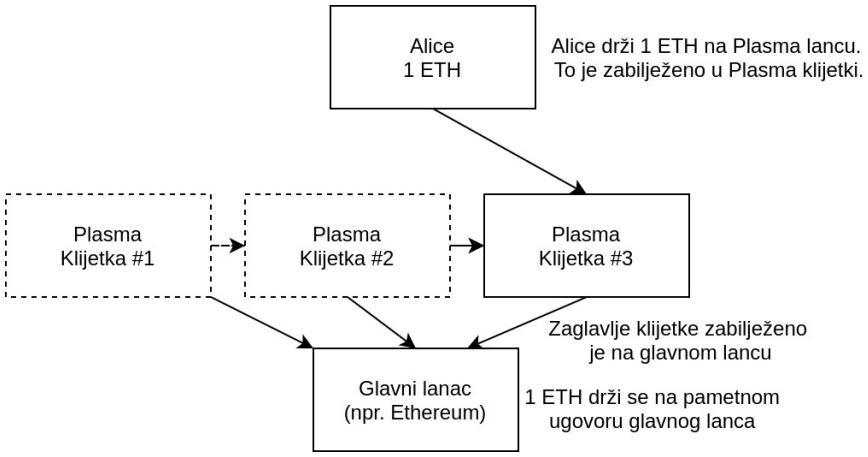
Ethereum se suočava s problemom uskog grla po broju transakcija koje može procesirati. Naime, mreža Ethereuma može procesirati u prosjeku 20 transakcija u sekundi, što je dosta manje u usporedbi s kartičarskom kućom Visa koja podržava preko 20,000 transakcija u sekundi. Nužan uvjet za usvajanje kriptovaluta kao sredstva za svakodnevno plaćanje je veći broj procesiranih transakcija u sekundi tj. bolja skalabilnost. Trenutni lanci klijetki su jako spori i skupi kako bi osigurali sigurnost protokola. Većina rješenja koja povećavaju skalabilnost lanaca klijetki žrtvuju bar dio sigurnosti ili decentralizacije. Plasma [10] je programski okvir (engl. framework) za implementaciju skalabilnih aplikacija. Njena struktura je izgrađena koristeći pametne ugovore i Merkleova stabla, te omogućuje stvaranje neograničenog broja lanaca djece (engl. child chains). Drugim riječima Plasma je rješenje drugog sloja, što znači da se Ethereumov glavni lanac ne mijenja, nego se ugrađuju nove funkcionalnosti povrh njega. Time se Ethereumov glavni lanac klijetki rasterećuje, a pomoćni lanac omogućuje brze i jeftine transakcije. Plasmine aplikacije većinu svog rada obavljaju izvan glavnog lanca.
Kako se većina rada obavlja izvan korijenskog lanca, potreban je način finaliziranja promjena. Zato se koriste predaje stanja (engl. state commitments), koje predstavljaju kriptografski način spremanja zbijene verzije stanja aplikacije. Svaki put kada korisnik poželi izaći s lanca Plasme koristi predaju stanja. Osnovno pravilo Plasme je sigurnost, tj. sredstva uvijek može preuzeti vlasnik i nitko drugi.
Implementacija Plasme može se podijeliti na dva dijela dizajna: preoblikovanje svih oblika računanja u skup funkcija preslikavanja i smanjivanja (engl. MapReduce), te neobavezna opcija sudjelovanja u dokazu o zalogu. MapReduce je skup funkcija koje organiziraju i računaju podatke koji se nalaze na više lanaca. Komunikacija između pomoćnih lanaca i glavnog lanca osigurana je dokazima prijevare (engl. fraud proofs), tako da je glavni lanac odgovoran za sigurnost mreže i kažnjavanje zlonamjernih korisnika. Svaki pomoćni lanac ima poseban mehanizam validacije klijetki i implementaciju dokaza prijevare, koji mogu biti implementirani povrh raznih algoritama za postizanje konsenzusa, od kojih su najpoznatiji dokaz o radu, dokaz o zalogu i dokaz o autoritetu. Pomoćni lanci klijetki se

organiziraju u hijerarhiju stabla, gdje je svaki lanac posebna grana glavnog lanca klijetki koja sadrži povijest pomoćnog lanca klijetki, te izračun preslikavanja i smanjivanja spremljenih u Merkleovim dokazima (engl. Merkle proofs). Pošto se samo Merkleovi dokazi emitiraju periodično na glavni lanac klijetki tijekom ispravnih prijelaza stanja, omogućena je skalabilnost, niske cijene transakcija i računanja.



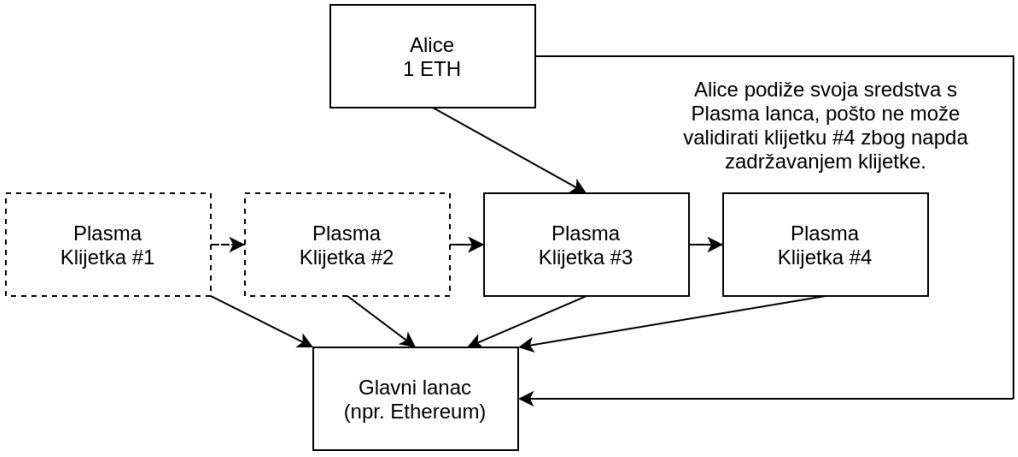

Slika 2.3: Plasmini ugovori

Svatko može kreirati proizvoljni lanac Plasme za skalabilnost pametnog ugovora raznih primjena. Plasmin lanac klijetki se sastoji od vanjskih višestranačkih kanala, koji mogu držati stanje jedni u ime drugih. Sadržaj Plasminog lanca klijetki se ne objavljuje na glavnom lancu, nego se zabilježi šifrat zaglavlja klijetke i dokaz prevare ako postoji. To je efikasno rješenje jer je dosta promjena stanja zabilježeno jednim šifratom.




Slika 2.4: Lanac Plasme

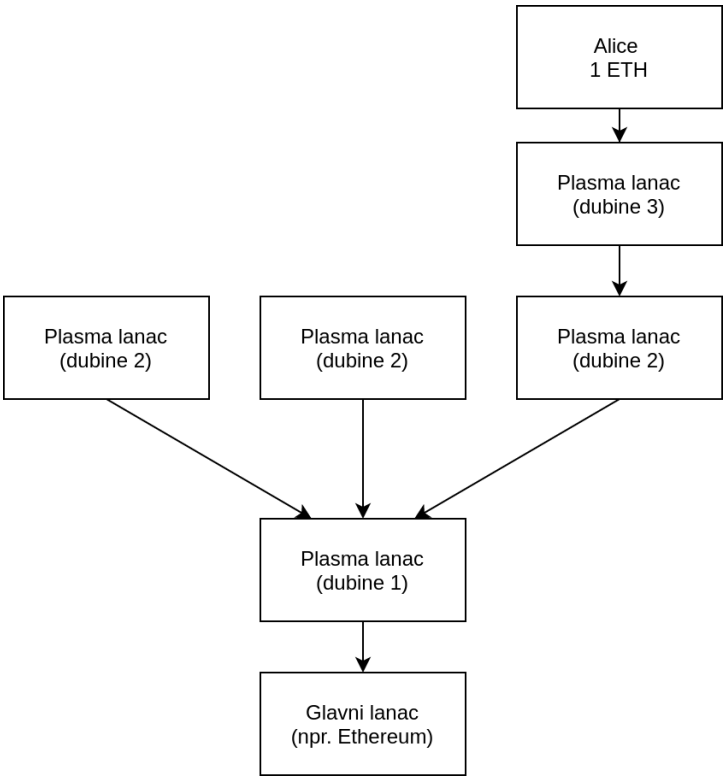
Isprekidani objekti predstavljaju stare klijetke na lancu Plasme, a pune linije predstavljaju zadnju klijetku koja je emitirana i zabilježena na glavnom lancu. U slučaju nevažećih klijetki, konsenzus se provodi dokazima o prijevari korijenskog lanca. U slučaju da je dokaz o prevari emitiran na glavni lanac, klijetka se poništava, a tvorac klijetke se kažnjava. Alice ne mora imati stanje svog računa zabilježeno na glavnom lancu, zato što stanje računa glavnog lanca određuje pametni ugovor koji implementira Plasmin lanac. Dokazi prevare su implementirani kao skup pametnih ugovora na glavnom lancu koji provode stanje pomoćnog lanca tako da smanjuju pokušaje prevare. Dokazi prevare provode interaktivni protokol povlačenja sredstava, za koje je potrebno određeno vrijeme. Ako korisnik odluči podići sredstva iz lanca Plasme, mora posvjedočiti trenutno stanje glavne knjige svih sudionika. Potom, svaki korisnik mreže ima određeni broj dana za dokaz koji svjedoči jesu li neka sredstva već bila potrošena. Ako neki korisnik dokaže da su sredstva bila potrošena, podizanje sredstava se otkazuje. U suprotnom, nakon isteka perioda za dokaze, ako nema podizanja većeg prioriteta, korisnik može podići svoja sredstva na glavni lanac.
U slučaju napada zadržavanjem klijetke (engl. block withholding attack), korisnici mogu brzo i jeftino povući svoja sredstva.




Slika 2.5: Napad zadržavanjem klijetke

Četvrta klijetka na lancu Plasme je klijetka koju je napadač zadržao i potvrdio na glavnom lancu, ali ju Alice nije mogla dohvatiti. Stoga povlači svoja sredstva i emitira dokaz o sredstvima na glavni lanac, te se njezino povlačenje procesira nakon perioda određenog za sporove.
Plasmini lanci klijetki imaju oblik stabla. Podizanje sredstava s određenog lanca podnosi se na bilo koji roditeljski lanac, te se naposljetku zabilježava u glavni lanac. Moguće je stvoriti

prijelaze stanja koji su periodično emitirani na roditeljski lanac, koji se potom emitiraju sve do glavnog lanca. Ta funkcionalnost omogućuje jako dobru skalabilnost računanja i promjena stanja računa, pošto se podnosi samo mali dio podataka na roditeljski lanac. U slučaju uključivanja nevaljane klijetke u lancu Plasme, oporavak se vrši primjenom stanja roditeljskog lanca Plasme.




Slika 2.6: Struktura stabla Plasme

Kako nisu svi podaci raširenи svim korisnicima, nego samo validatorima stanja, svim korisnicima je u interesu pratiti lanac na kojemu imaju sredstva kako bi mogli periodički kažnjavati prevare i kako bi mogli pravovremeno povući sredstva u slučaju napada zadržavanjem klijetke.
Glavni nedostatak Plasme je problem masovnog izlaska, koji predstavlja scenarij u kojem velika količina korisnika pokuša izaći iz lanca Plasme u kratkom vremenskom periodu, te se time preoptereti Ethereumov glavni lanac i stvori usko grlo. Takva aktivnost može biti sredstvo zlonamjernog napadača za narušavanja funkcionalnosti glavnog lanca.

# Poglavlje 3

# Telegram Open Network

## 3.1 Uvod

Telegram Open Network (TON) [6] je brz, siguran i skalabilan lanac klijetki, koji može procesirati milijune transakcija u sekundi. Iako mreža još nije zaživjela, TON ima cilj pružiti funkcionalnost distribuiranog superservera, pružajući usluge smještaja raznih aplikacija. TON se sastoji od više vrsta lanaca klijetki:

- Glavni lanac (engl. master blockchain)
- Radni lanac klijetki (engl. working blockchain)
- Lanac klijetki krhotina (engl. shard blockchain)
- Svaka klijetka u lancu krhotina

Glavni lanac je jedinstveni lanac klijetki koji sadrži osnovne informacije o protokolu i pripadajućim parametrima, skupu validatora i njihovim ulozima, skupu trenutno aktivnih radnih lanaca, njihovih krhotina i naposljetku šifrate nedavnih klijetki svih radnih lanaca i lanaca krhotina. Radni lanci su lanci koji sadržavaju transakcije pametnih ugovora i prijenose vrijednosti. Svaki radni lanac ima svoja pravila, tj. posebni format adresa i transakcija, te različite virtualne mašine i bazne valute. Svaki radni lanca mora zadovoljavati osnovne kriterije kako bi interoperabilnost među različitim radnim lancima bila moguća i jednostavna. Nadalje, svaki radni lanac podijeljen je na lance klijetki krhotina, kojih može biti do $2^{60}$ po radnom lancu. Lanci klijetki krhotina sadržavaju podatke samo za podskup adresa ovisno o nekoliko prvih najznačajnijih bitova. Svaka klijetka u lancu klijetki krhotina je zapravo mali lanac klijetki, tzv. vertikalni lanac.

## 3.2 Inovacije koje uvodi TON

Telegram Open Network uvodi dvije značajne inovacije:

- Mehanizam vertikalnih popravki (engl. ”self-healing” vertical blockchain mechanism)
- Trenutno usmjeravanje po hiperkocki (engl. Instant Hypercube Routing)

Navedene inovacije omogućuju TONovom lancu klijetki da bude istovremeno brz, pouzdan i skalabilan.
Pošto TON uvodi funkcionalnost vertikalnih lanaca, TONov lanac klijetki se još naziva 2-lanac klijetki. Obično se vertikalni lanac sastoji od samo jedne klijetke. Normalno će se samo valjane klijetke zabilježiti, ali protokol dopušta otkrivanje prethodno zabilježenih nevaljanih klijetki i njihov popravak. U slučaju da postane nužno popraviti nevaljanu klijetku u lancu krhotina, nova klijetka koja sadrži zamjenu ili razliku se dodaje u vertikalni lanac. Da bi riješili problem zabilježene nevaljane klijetke, većina drugih sustava lanaca klijetki mora odbaciti sve promjene nakon zadnje valjane promjene. Ovakav pristup je loš je se odbacuje velik broj ispravnih i izvršenih transakcija.
Za razliku od drugih projekata koji koriste pristup odozgo prema dolje za dijeljenje lanca klijetki (engl. blockchain sharding), TON koristi pristup odozdo prema gore. Drugim riječima lanac se maksimalno podijeli, tako da svaki račun ili pametni ugovor ostane u svome lancu klijetki krhotina. Pošto je neefikasno čuvati ogromnu količinu lanaca klijetki koji uglavnom imaju rijetke promjene na sebi, lanci više računa se grupiraju u jedan lanac klijetki krhotina. Stoga, lanci računa postoje samo virtualno unutar lanaca klijetki krhotina. Navedeni pristup se naziva paradigma beskonačnog dijeljenja (engl. infinite sharding paradigm). Pošto su računi zapravo lanci krhotina, jedini način da jedan račun utječe na drugi je putem poruka.
Druga inovacija koju uvodi TON je trenutno usmjeravanje po hiperkocki (engl. Instant Hypercube Routing) koja omogućuje procesiranje poruke napravljene u jednoj klijetki lanca krhotina već u drugoj klijetki odredišnog lanca krhotina, bez obzira na broj lanaca krhotina u sustavu.
Sustav može podržati do $2^{32}$ radnih lanaca. Postojanje radnog lanca je virtualno, zato što je svaki radni lanac sačinjen od lanaca krhotina. U slučaju da postoji samo jedan lanac krhotina onda on predstavlja stvarni lanac klijetki. Radni lanac se stvara posebnom transakcijom koja sadrži specifikaciju radnog lanca u glavnom lancu. Nakon što je transakcija stvaranja zabilježena na glavnom lancu, dvije trećine validatora treba prihvatiti novi radni lanac, zato što će oni biti obavezni nadograditi svoj softver za procesiranje klijetki.
Svaki radni lanac identificiran je nenegativnim 32-bitnim cijelim brojem: (identifikator_radnog_lanca: unit32), a svaki lanac krhotina identificiran je parom:

(identifikator_radnog_lanca, prefiks_krhotine), gdje je prefiks_krhotine bitovni niz duljine najviše 60, koja određuje podskup računa koji su uključeni u taj lanac krhotina.
TONov lanac klijetki omogućuje dinamičko dijeljenje (engl. dynamic sharding) gdje se krhotina (w,s) može podijeliti na (w,s.0) i (w,s.1) ako su ispunjeni neki uvjeti, npr. transakcijsko opterećenje na krhotinu (w,s) je veće duži vremenski period. Obratno, ako je tijekom određenog perioda opterećenje malo, krhotine (w,s.0) i (w,s.1) se mogu automatski spojiti u (w,s).
Inicijalno je definiran samo jedan radni lanac, tzv. nulti radni lanac (engl. workchain zero). On se koristi za rad s TONovim pametnim ugovorima i transakcijama Grama, TONove bazne valute. Gram tokeni se koriste za stjecanje uloge validatora, plaćanje transakcijskih naknada, plaćanje goriva za izvršavanje kôda pametnih ugovora i plaćanje skladištenja.
Klijetke glavnog lanca i lanca krhotina generiraju se svakih 5 sekundi. Klijetka glavnog lanca se generira otprilike jednu sekundu nakon što su generirane klijetke lanaca krhotina, zato što klijetka glavnog lanca mora sadržavati šifrate najnovijih klijetki svih lanaca krhotina. Jednom kad glavni lanac sadrži šifrat klijetke lanca krhotine, stanje lanca krhotine je nepromjenjivo i svaka iduća klijetka ga može referencirati. To omogućuje korištenje transakcija i poruka u drugim lancima krhotina pet sekundi nakon što su bile zabilježene. Svaka nova klijetka lanca krhotina sadrži šifrat najnovije klijetke glavnog lanca. Iz vanjske perspektive šifrat zadnje klijetke iz glavnog lanca određuje stanje sustava.

## 3.3 Dokaz o zalogu

TONov lanac klijetki za postizanje konsenzusa koristi dokaz o zalogu (engl. Proof-of-Stake), tj. za generiranje novih klijetki u lancu krhotina i glavnom lancu. Dokaz o zalogu uključuje skup posebnih čvorova zvanih validatori, koji su uložili Gram tokene posebnom transakcijom na glavnom lancu kako bi dobili pravo generiranja i verifikacije novih klijetki. Podskup validatora je pripisan svakom lancu krhotina na deterministič an pseudoslučajan način i mijenja se svakih 1024 klijetke. Validatori pripisani određenom lancu krhotina uzimaju predložene transakcije od klijenata, predlažu novu klijetku i konačno postižu konsenzus.
Validatori i ostali čvorovi provjeravaju valjanost predložene klijetke, te ako je validator predložio nevaljanu klijetku može automatski biti kažnjen oduzimanjem dijela ili čak cijelog zaloga ili može dobiti suspenziju na određeno vrijeme. Prilikom postizanja konsenzusa, validatori među sobom dijele transakcijske naknade i nagradu za pronalazak nove klijetke. Jedan validator može biti odabran za sudjelovanje u nekoliko podskupova validatora, te se od njega očekuje da će izvršiti validaciju i algoritam za postizanje konsenzusa paralelno.

Postoje dvije vrste dokaza o zalogu:

- Delegirani dokaz o zalogu (engl. Delegated Proof-of-Stake)
- Bizantski otporan (engl. Byzantine Fault Tolerant)

Delegirani dokaz o zalogu podrazumijeva određenog opće poznatog tvorca svake klijetke, koji sam potpisuje klijetku i nitko drugi ne može stvoriti određenu klijetku umjesto njega. S druge strane, bizantski otporan dokaz o zalogu definira poznati podskup validatora, te svaki od njih ima pravo predložiti novu klijetku, a protokol odabire koja će klijetka biti zabilježena. Prije objavljivanja klijetke ostalim čvorovima, većina validatora mora validirati i potpisati novopredloženu klijetku.
Nakon što su sve klijetke lanca krhotina generirane ili je vrijeme isteklo, nova klijetka, koja sadrži šifrate najnovijih klijetki svih lanaca krhotina kreira se na glavnom lancu. Za kreiranje nove klijetke na glavnom lancu koristi se bizantski otporan dokaz o zalogu svih validatora.

# Poglavlje 4

# Interoperabilnost

## 4.1 Uvod

Prvih par godina razvoja tehnologija raspodijeljenih registara, smatrala se potreba za samo jednim lancem klijetki. Kako je vrijeme odmicalo, pojavljivali su se novi projekti otvorenog tipa koji su imali različite prednosti u odnosu na Bitcoin: sigurnost, privatnost, efikasnost, skalabilnost, složenost platforme, jednostavnost upotrebe za programere i drugi. Time se javila se potreba za interoperabilnošću između različitih lanaca klijetki. Iz tehničke perspektive razlikujemo tri tipa interoperabilnosti među lancima [2]:

- Centralizirana ili višepotpisna javno-bilježnička shema - stranka ili grupa stranaka se dogovori izvesti akciju na lancu B kada se dogodi događaj na lancu A.
- Sporedni lanci (engl. sidechains) - sustav unutar lanca klijetki koji može čitati i validirati događaje i stanja na drugim lancima klijetki.
- Zaključavanje šifrata (engl. hash-locking) - postavljanje operacija na lancu A i lancu B koje imaju isti okidač, obično otkrivanje praslike određenog šifrata.

Postoji nekolicina slučajeva potencijalne uporabe interoperabilnosti [3]. Prijenosna imovina omogućuje prijenos imovine s matičnog lanca koji je autoritativan za njegovo vlasništvo na drugi lanac, trgovina istih, korištenje imovine kao kolateral i mogućnost povrata imovine na matični lanac.
Nerijetko lanci klijetki i pametni ugovori trebaju dohvatiti informacije van njihove mreže, pa im je potreban svjedok (engl. oracle) koji će im pružiti te informacije. Takva svjedočanstva su usluge koje šalju i provjeravaju pojave u stvarnom svijetu i dostavljaju te podatke pametnim ugovorima, pokrećući promjene stanja na lancu klijetki. Jedna primjena međulančanih svjedoka (engl. cross-chain oracles) uključuje postojanje pametnog

ugovora na jednom lancu koji izvodi neku akciju s nekom adresom samo ako primi dokaz od identitetskog svjedoka na drugom lancu koji potvrđuje da je adresa posebnog jedinstvenog identiteta.
Opterećivanje imovine predstavlja zaključavanje imovine na lancu A i postavljanje uvjeta zaključavanja ovisno o aktivnosti na lancu B. Primjene su založno pravo, kolateral u financijskim derivatima, stečajne naknade, sudski nalozi i razni slučajevi upotreba koji uključuju sigurnosne depozite.
Još jedan primjer je zamjena imovine, u tehničkim krugovima poznata kao atomna zamjena (engl. Atomic swap). Cilj je omogućiti korisniku X zamijeniti digitalnu imovinu A s digitalnom imovinom B čiji je vlasnik korisnik Y, pri čemu su imovine A i B na različitim lancima, a korisnici A i B imaju račune na oba lanca. Zamjena mora biti sigurna, tj. oba prijenosa moraju biti uspješno izvršena ili nijedan ne smije biti izvršen.
Naposljetku, opći međulančani ugovori uključuju ugovore koji omogućuju isplaćivanje dividende na lancu A vlasnicima imovine čije vlasništvo je registrirano na lancu B.

## 4.2 Atomna međulančana zamjena

Atomna međulančana zamjena (engl. atomic cross-chain swap) je zamjena između dva čvora koristeći pametne transakcije. One omogućuju zamjenu dvaju različitih kriptovaluta od korisnika do korisnika bez potrebe za centralnim tijelom, kao npr. burzom. Izraz atomna u nazivu označuje da je zamjena ograničena na skup binarnih izlaza, tj. zamjena će biti izvedena u potpunosti ili neće biti izvršena nikako. Sve dok se zamjena ne dogodi oba korisnika imaju potpunu kontrolu nad svojim kriptovalutama.
Koncept atomnih zamjena osmislio je Tier Nolan u svibnju 2013. godine, a prva atomna zamjena je uspješno izvršena 20. rujna 2017. godine između lanaca klijetki Decred (DCR) i Litecoin (LTC) [7]. Od tada nekoliko projekata i decentraliziranih burzi, kao što su 0x i Lightning Labs donekle su uspjeli ugraditi atomne zamjene.
Razlikujemo dvije vrste atomnih zamjena: lančane (engl. on-chain) i vanlančane (engl. off-chain). Lančana atomna zamjena je zamjena kojom se mijenjaju dvije različite kriptovalute, ali homogenih lanaca klijetki, npr. Litecoin i Bitcoin ili Litecoin i Decred. Vanlančana atomna zamjena je zamjena koja se ne odvija na glavnom lancu nego na posebnom lancu, npr. Lightning Network na mreži Bitcoina. Pored zamjena na homogenim lancima, vanlančane atomne zamjene podržavaju zamjene među heterogenim lancima klijetki, npr. zamjene između Bitcoin i Ethereum ERC-20 tokena.
Sljedeći uvjeti su nužni za uspješnu atomnu zamjenu: oba lanca klijetki moraju podržavati funkciju šifriranja istog tipa i ugovore o vremenskom zaključavanju, te posebne funkcije za programiranje algoritma zamjene.

## Algoritam

Pretpostavimo da dva korisnika, Alice i Bob, žele zamijeniti svoje kriptovalute po dogovorenom tečaju. U slučaju da ne žele koristiti atomne zamjene potrebne su najmanje dvije transakcije: 1. Alice šalje BTC Bobu na mreži Bitcoina, 2. Bob šalje XRP Alice na mreži Ripplea. Ova zamjena nije sigurna, jer nakon što je prvi prijenos uspješno obavljen, te ne može biti vraćen Bob nema obavezu poslati XRP tokene Alice.
Algoritam atomne zamjene je siguran, tj. niti jedna strana ne može pridobiti vlasništvo nad sredstvima druge strane, bez da druga strana ne zadobije vlasništvo nad sredstvima prve strane.

**Algoritam 5:** Atomna zamjena

1 Alice kreira adresu ugovora.
2 Alice generira tajnu vrijednost i izračuna njen šifrat.
3 Alice šalje svoj BTC na adresu ugovora. Vlasništvo nad tim BTCom može biti stečeno ispunjenjem bilo kojeg od dva uvjeta:
(I) Pružanjem vrijednosti koja je generirala šifrat iz koraka 2 potpisanom Bobovim privatnim ključem.
(II) Pružanjem vrijednosti koja je generirala šifrat iz koraka 2 potpisanom privatnim ključem koji pripada Alice nakon određenog vremena.
4 Alice šalje šifrat iz koraka 2 i adresu Bobu.
5 Bob generira adresu ugovora koristeći šifrat koji je primio od Alice.
6 Bob šalje svoj XRP na adresu ugovora generiranog u koraku 5. Vlasništvo nad tim XRPom može biti stečeno ispunjenjem bilo kojeg od dva uvjeta:
(I) Pružanjem vrijednosti koja je generirala šifrat iz koraka 2 potpisanom privatnim ključem koji pripada Alice.
(II) Pružanjem vrijednosti koja je generirala šifrat iz koraka 2 potpisanom Bobovim privatnim ključem nakon određenog vremena.
U prvom je slučaju tajna vrijednost automatski objavljena na mreži.
7 Alice dobiva XRP iz generirane adrese koristeći tajnu vrijednost koja je generirala šifrat zajedno sa svojim potpisom, te otkrivanjem vrijednosti Bobu.
8 Bob dobiva BTC koristeći svoj privatni ključ i vrijednost dobivenu od Alice.

U drugim slučajevima 3. i 6. koraka algoritma koristi se šifrirani ugovor s vremenskim zaključavanjem (engl. hashed timelock contract) ili skraćeno HTLC, koji onemogućuje podizanje sredstava bez otkrivanja tajne vrijednosti u određenom vremenskom periodu. Da bi se atomna zamjena uz pomoć HTLC uspješno izvršila nužno je da obje strane obave primanje sredstava u određenom vremenskom roku. U slučaju da bar jedna strana ne obavi primanje uspješno, poništava se kompletna zamjena, te se izvrše povrati sredstava. Cijeli je proces automatiziran, te su obje strane osigurane od prijevare.

Iako su atomne zamjene poštene i sigurne za oba korisnika, iracionalno ili nepravovremeno ponašanje može rezultirati gubitkom sredstava. Ako Alice, nakon što je kreirala ugovor, generirala tajnu vrijednost, te poslala sredstva na adresu ugovora, otkrije svoju tajnu vrijednost Bobu, on može dohvatiti njena sredstva iz ugovora, te također zadržava svoja sredstva. Da bi se Alice osigurala od tog scenarija, nužno je da prvi puta otkrije tajnu vrijednost prilikom otključavanja sredstava iz Bobovog ugovora. To podrazumijeva da je Bob već kreirao ugovor i poslao svoja sredstva na adresu ugovora. Nadalje, nakon što je Alice preuzela sredstva iz Bobovog ugovora, ako u određenom vremenskom periodu Bob ne dohvati sredstva iz ugovora kojeg je Alice kreirala, ona može dohvatiti svoja sredstva. Pošto su za dohvaćanje sredstava iz oba ugovora potrebni potpisi tajne vrijednosti privatnim ključem, niti jedan korisnik, osim Alice i Boba, ne može dohvatiti sredstva zaključana u ugovorima, čak ni ako mu je poznata tajna vrijednost.

## Prednosti i nedostaci

Atomne zamjene imaju mnoge prednosti, a neke od njih su: decentralizacija, sigurnost, naknade, kompletnost i brzina. Jedna od glavnih prednosti atomnih zamjena je eliminacija posrednika u procesu zamjene različitih kriptovaluta, pri čemu korisnici ne moraju vjerovati čak niti jedan drugome. Burze kriptovaluta su nerijetko mete hakerskih napada, što rezultira gubitkom vlasništva nad kriptovalutama. Atomne zamjene eliminiraju potrebu za njima. Sve naknade vezane uz zamjenu putem burzi su eliminirane. Konačno, kriptovalute se mogu bolje natjecati s fizičkim novcem, zbog bolje interoperabilnosti. Vanlančane atomne zamjene su gotovo trenutne.
S druge strane, atomne zamjene imaju brojne nedostatke. Kriptovalute čiji lanci klijetki ne podržavaju pametne ugovore ne mogu implementirati atomne zamjene. Burze i poznati novčanici nemaju ugrađenu funkcionalnost atomnih zamjena. Lančane atomne zamjene su poprilično spore. Ako je zamjena neuspješna, nije reverzibilna ali povrat je zaključan određeni vremenski period. Atomne zamjene ne eliminiraju u potpunosti rizik od jedinstvene točke kvara.

# Bibliografija


[1] A. Back et al., *Hashcash-a denial of service counter-measure*, (2002).

[2] V. Buterin, *Chain interoperability*, R3 Research Paper (2016).

[3] V. Buterin, *Chain interoperability*, R3 reports (2016).

[4] V. Buterin et al., *A next-generation smart contract and decentralized application platform*, white paper **3** (2014), br. 37.

[5] L. Davidson i W. E Block, *Bitcoin, the Regression Theorem, and the emergence of a new medium of exchange*, Quarterly Journal of Austrian Economics **18** (2015), br. 3, 311.

[6] N. Durov, *Telegram Open Network*, 2017, dostupno na `https://www.kriptovaluta.hr/wp-content/uploads/2018/03/TON-Technology.pdf` (rujan 2020.).

[7] M. H. Miraz i D. C. Donald, *Atomic cross-chain swaps: development, trajectory and potential of non-monetary digital token swap facilities*, Annals of Emerging Technologies in Computing (AETiC) Vol **3** (2019).

[8] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008, dostupno na `https://bitcoin.org/bitcoin.pdf` (rujan 2020.).

[9] T. Peterson, *Metcalfe's Law as a Model for Bitcoin's Value*, Alternative Investment Analyst Review Q **2** (2018).

[10] J. Poon i V. Buterin, *Plasma: Scalable autonomous smart contracts*, White paper (2017), 1–47.

[11] J. Poon i T. Dryja, *The bitcoin lightning network: Scalable off-chain instant payments*, 2016.

[12] G. Wood et al., *Ethereum: A secure decentralised generalised transaction ledger*, Ethereum project yellow paper **151** (2014), br. 2014, 1–32.

# Sažetak

Tehnologije raspodijeljenih registara, od kojih su najpoznatiji lanci klijetki, pojavljuju se 2009. godine kao rješenja za postizanje konsenzusa u bizantinskom okruženju. U ovom diplomskom radu objašnjena je funkcionalnost najpopularnijih javnih lanaca klijetki, čiji se razvoj može podijeliti prema njihovim funkcionalnostima. Bitcoin, kao prvi javni lanac klijetki koji rješava problem dvostruke potrošnje, ima glavnu funkcionalnost digitalnog novca, tj. bilježenje i prijenos vrijednosti. Ethereum, kao inovacija, uvodi pametne ugovore koji znatno proširuju funkcionalnost dotada prilično primitivnog Bitcoinovog skriptnog jezika. Trenutno se Ethereum suočava s problemom skalabilnosti, pa se prirodno javljaju rješenja za taj nedostatak bez narušavanja decentralizacije i sigurnosti mreže. Telegram Open Network je mreža koja uz sve funkcionalnosti Ethereuma uvodi trenutno usmjeravanje po hiperkocki te mehanizam vertikalnih popravki, čime poboljšava skalabilnost za više redova veličine. Pošto je područje lanaca klijetki te općenito raspodijeljenih registara popularno, i iz dana u dan se razvijaju nove decentralizirane platforme, mogućnost interoperabilnosti među njima dobiva na važnosti.

# Summary

Distributed ledger technologies, the most famous of which are blockchains, emerged in 2009. as consensus solutions in the byzantine environment. In this thesis, the functionality of the most popular public blockchains is explained. The development of blockchains can be divided according to their functionalities. Bitcoin as the first public blockchain to solve the problem of double spending has the main functionality of digital money, ie. recording and transferring value. Ethereum as an innovation introduces smart contracts which significantly expand the functionality of the hitherto rather primitive Bitcoin's scripting language. Currently, Ethereum is facing a problem of scalability, however, remedies arise naturally, without decreases of decentralization and network security. One of them is Telegram Open Network, a network that, with all the functionality of Ethereum, introduces instant hypercube routing and "self-healing" vertical blockchain mechanism, thus improving scalability by several orders of magnitude. Due to the popularity of the field of blockchains and generally distributed ledgers, new decentralized platforms develop on a daily basis, hence, the possibility of interoperability among them is gaining in importance.

# Životopis

Ante Sosa rođen je 1995. godine u Zagrebu, gdje je završio osnovnu školu nakon koje upisuje Prirodoslovno-matematičku Gimnaziju Lucijana Vranjanina. Tijekom srednje škole sudjelovao je na natjecanjima iz matematike i fizike.
Nakon završenog srednjoškolskog obrazovanja upisuje preddiplomski sveučilišni studij Matematika na Prirodoslovno-matematičkom fakultetu u Zagrebu. Prvostupničku diplomu stječe 2017. godine, te iste upisuje sveučilišni diplomski studij Računarstvo i matematika na istom fakultetu. Tijekom diplomskog studija sudjeluje na Erasmus+ programu na Sveučilištu u Ljubljani.
Nakon treće godine studija, zapošljava se kao programer u tvrtki Ericsson, a potom radi kao analitičar podataka u tvrtki Reversing Labs u Zagrebu. U slobodno vrijeme rekreativno se bavi nogometom, te uživa u istraživanju novih tehnologija na području lanaca klijetki.