IZBORNI PROJEKT

Usporedba hash algoritama u kriptovalutama

Petra Avsec

Algoritmi i strukture podataka
Zavod za računarstvo
prof. Kristijan Lenac
Tehnički fakultet Sveučilišta u Rijeci
Kolovoz 2020.

Sadržaj

1 Kriptovaluta

Kriptovaluta je vrsta digitalnog novca, elektronski način razmjene novca i zapisivanja transakcija na računalu. To je sredstvo razmjene koje koristi kriptografiju kako bi stvorilo i osiguralo prijenos novca.[1] Za razliku od banaka, kriptovalute su decentralizirani sustav što znači da nema središnje organizacije koja kontrolira vrijednost, provjerava transakcije ili na bilo koji drugi način utječe na razmjenu novca. Kriptovaluta je peer-to-peer sustav razmjene što znači da za svoje funkcioniranje koristi rad svojih korisnika, oni su zaslužni za stvaranje novih novčanih jedinica i prijenos postojećih.
Sigurnost i integritet transakcija osigurava zajednica rudara (*miners*) koji svojim računalima ovjeravaju transakcije dodavanjem vremenske oznake i ubacivanjem u tzv. glavnu knjigu svih transakcija (*ledger*). Transakcije su računalno zahtjevne i nepraktične za poništiti ili promijeniti što kriptovalutu čini vrlo sigurnim načinom prijenosa novca.
Sustav je siguran dok god većinu čvorova kontroliraju "pošteni" (*honest*) korisnici, a ne napadači (*attackers*).
Ispravnost svake jedinice kriptovalute osigurava blockchain.[2]

1.1 Pregled kriptovaluta

[3]

1.2 Blockchain

Sustav koji kriptovalute koriste za spremanje podataka o transakcijama je blockchain, decentralizirani mehanizam pohranjvanja informacija o transakcijama. Blockchain je distribuirana glavna knjiga u kojoj se spremaju i čuvaju podaci o transakcijama tako da svi korisnici imaju svoju kopiju, nema osobe ili organizacije koja ima popis transakcija te ih može mijenjati ili brisati. Sve informacije su vidljive svima koji koriste tu valutu te nije moguća manipulacija podataka.
Blockchain je realiziran kao rastuća lista podataka raspoređenih u blokove. Svaki

blok je povezan s prethodnim tako što sadrži hash vrijednost tog bloka. Osim hash vrijednosti, blokovi sadrže i: index bloka, hash prethodnog bloka, vremenske oznaku, podatake, koji su u slučaju kriptovaluta, transakcije.
Transakcije su strukturirane u obliku Merkle stabla koje se stvara hashiranjem podataka u više navrata. Prvo se hashiraju same transakcije, zatim upare dva hasha koji se ponovno hashiraju. Postupak se ponavlja dok ne dobijemo samo jedan hash, hash korijen (*root hash*) ili merkle korijen. Svaki list je hash transakcije, a svaki čvor hash prethodnih hasheva. Kada su podaci spremljeni u ovakvoj strukturi podataka, lako se može provjeriti da li je transakcija spremljena u tom setu podataka.
Potvrda da su transakcije određenog bloka prihvaćeni od strane ostalih čvorova i time ispravni, dobije se kada novi čvorovi počnu koristiti hash tog bloka u potrazi za novim.

1.2.1 Proof of Work

Proof of work je mehanizam kontrole pristupa koji koristimo kada želimo ograničiti, ali ne i zabraniti pristup resursu. Ovaj mehanizam od korisnika zahtjeva neku vrstu rada, najčešće procesorko vrijeme, i tako odvraća denial-of-service napade i druge vrste iskorištavanja usluga kao što je spam. Proof of work traži od korisnika da izračuna neku funkciju, tzv. *pricing function*.
Kriptovalute primjenjuju proof of work tako što miner mora pronaći nonce vrijednost koja hashirana zajedno sa ostalim parametrima koji ulaze u blok, zadovoljava neke uvjete. Kao primjer se može uzeti kriptovaluta Bitcoin, kod koje hashirana vrijednost mora imati određen broj bitova na početku hasha nula. Što je više nula u tom zahtjevu to je teže naći pripradajući hash.
Proof of Work se koristi u blockchainu kako bi se stvorio zapis transakcija koji se ne može lako promijeniti, tj. trebalo bi ponoviti potreban rad za sve transakcije koje su bile nakon te koju bi htjeli promijeniti. Najdulji lanac služi kao dokaz svih događaja u lancu i dokaz najveće potrošene procesorske snage.[4]

1.2.2 Hash funkcija

Hash funkcije su funkcije koje ulazne podatke proizvoljne dužine sažimaju u izlaz određenog formata i veličine. Idealni algoritmi neće imati kolizija, tj. za svaki ulaz, izlaz algoritma je različit, neovisno o veličini promjena ulaznih podataka.
Hash funkcije koje se koriste u blockchainu, kriptovalutama i općenito kriptografiji moraju biti jednostrane što znači da ne možemo lako iz izlaza dobiti odgovarajući ulaz.
Svojstva optimalnih hash funkcija:

- determinističke su - ista poruka (ulazna vrijednost) uvijek rezultira istim hashom
- velikom brzinom računaju hash vrijednost bilo koje poruke
- teško je i nepraktično generirati poruku koja daje određeni hash
- teško je pronaći dvije različite poruke sa istom hash vrijednosti
- mala promjena poruke treba imati veliki utjecat na izalznu hash vrijednost kako se te dvije poruke ne bi mogle povezati na temelju sličnih izlaznih vrijednosti (učinak lavine)[5]

Sigurne hash funkcije su otporne na sve vrste kriptoanalitičkih napada:

- preimage attack - napad u kojem se nastoji pronaći ulaz određenog hasha ukoliko znamo duljinu ulaza. Brute forceom (napad na koristeći grubu silu, u ovom kontekstu slanje svih mogućih ulaza u hash funkciju dok se ne dobije željeni hash) se ulaz može pronaći u 2^N evaluacija, ako je N duljina ulaza
- birthday attack - pokušaj pronalaska dva različita ulaza hash funkcije koji rezultiraju istim izlazom (kolizija) - $2^{L/2}$ evaluacija
- collision attack - pronađena su dva različita ulaza čiji su izlazi identični

2 Pregled hash algoritama

Puno kriptovaluta, različita svojstva blabla.

2.1 SHA-256

Secure Hash Algorithm 2 je set kriptografskih funkcija koje je 2001. objavila NSA. Broj 256 u nazivu označava veličinu izlazne vrijednosti funkcije koja iznosi 256 bitova. Postoji nekoliko različitih verzija ovog algoritma sa različitim veličinama izlaza od 224, 384, 512, 512/224 i 512/256 bitova.
SHA je jedan od najkorištenijih algoritama u svijetu, a koriste ga među ostalim kriptovalute Bitcoin, Namecoin, Peercoin, Nxt, MazaCoin, te se koristi i u autentikacijskom procesu Debian softver paketa.
SHA-256 je napravljen koristeći Merkle-Damgard strukturu koja je pak napravljena pomoću Davies–Meyer jednostrane kompresijske funkcije. Temeljna ideja iza Davies–Meyer konstrukcije je kompresija bloka teksta u n bitova koristeći enkripcijski algoritam. To se postiže stavljanjem nasumične početne vrijednosti od n bitova kao poruke i korištenjem teksta kao ključa. Time se kao produkt enkripcije dobije blok od n bitova. Na rezultat se zatim još primjeni operacija XOR kako bi se smanjila vjerojatnost stvaranja kolizije.
Merkle–Damgård hash funkcija je način stvaranja kriptografske hash funkcije otporne na kolizije iz jednostranih kompresijskih funkcija. Koristi se u SHA algoritmu kod rastavljanja ulaza na blokove duljine 512 bitova, koji se zatim obrađuju jedan po jedan koristeći jednostrane kompresijske funkcije bazirane na Davis-Meyeru.

2.1.1 Opis rada

Poruka se dijeli u blokove duljine 512 bitova, a ukoliko je blok manji, proširimo ga do 512 bitova. Blokovi se zatim obrađuju jedan po jedan.
Poruka se proširuje tako da se duljini poruke nadoda jedan bit na kraj čime se dobije N bitova. Na kraj dobivenog dodaje se $512 - 64 - N$ nula, a u zadnja 64 bita se zapiše duljinu originalne poruke. Blokovi od 512 bitova dijele se na 16 dijelova po 32

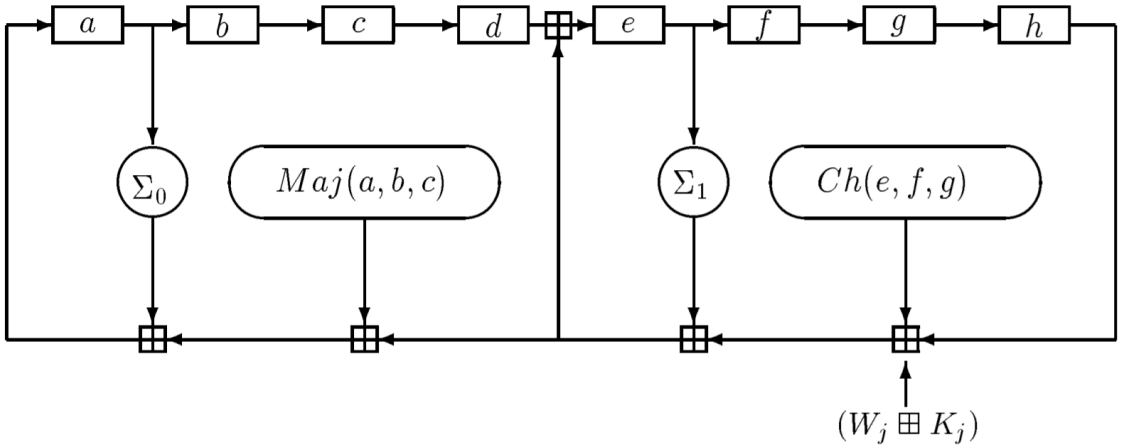
bita ($m0 - m15$).

Pseudokod:

- 8 početnih hash vrijednosti duljine 32 bita (ostaci korijena prvih osam prim brojeva):
  - $H(0)_1 = 6a09e667$
  - $H(0)_2 = bb67ae85$
  - $H(0)_3 = 3c6ef372$
  - $H(0)_4 = a54ff53a$
  - $H(0)_5 = 510e527f$
  - $H(0)_6 = 9b05688c$
  - $H(0)_7 = 1f83d9ab$
  - $H(0)_8 = 5be0cd19$
- Glavna petlja (N iteracija gdje je N broj blokova poruke):
  1. inicijalizacija a, b, c, d, e, f, g, h sa vrijednostima h_1 do h_8
  2. primjena SHA-256 kompresijske funkcije čime se dobiju nove vrijednosti varijabli $a - h$
  3. računanje novih $h_1 - h_8$, tako da $h_1 = a + h_1 ... h_8 = h + h_8$

  Rezulat izvršavanja petlje je hash poruke s početka.
- Kompresijska funkcija:
  - 64 iteracije, 6 logičkih funkcija koje se koriste za dobivanje novih vrijednosti $a - h$. Sve funkcije rade sa 32-bitnimm riječima(?) i izlaz im je velićine 32 bita. U računanju se koriste i 64 konstante, 32-bit riječi $K_0 - K_{63}$ (ostatci kubnih korijena prvih 64 prostih brojeva)

– u računanju se za kriptiranje koristi: bitwise XOR, bitwise AND, bitwise OR, bitwise complement, zbrajanje mod 2^{32}, posmak u desno za n bitova, rotacija u desno za n bitova[6]



Slika 1: Kompresijska funkcija SHA-256 algoritma. Slika prikazuje j-ti korak funkcije, a $\boxplus$ označava mod 2^{32} zbrajanje

Razina zaštite, sigurnosti, u kriptografiji je mjera snage(?) hash funkcije. Izražena je u bitovima, n - bit security, sto znaci da napadač treba izvesti 2^n operacija da bi razbio funkciju.

Broken hash funkcija - uspjeli smo izvesti barem jedan od napada: collision attack i preimage attack.

SHA-256 ima 128 bitnu razinu zaštite.

2.2 X11

X11 algoritam je, isto kao i SHA-256, proof of work algoritam. Za razliku od ostalih algoritama koristi drugačiji pristup hashiranju, ulančavanje algoritama. Razvio ga je Evan Duffield te je algoritam 2014. implementiran u protokol Darkcoin kriptovalute, kasnije preimenovane u DASH. Napravljen je kako bi se otežala izrada ASIC-a koji će efikasno rudariti X11 algoritam, iako danas više nije ASIC otporan.
ASIC (application-specific integrated circuit) IC čip spečificno izrađen za neki zadatak, svrhu. Kod kriptovaluta njihova uloga je računanje hash algoritama kriptovaluta (kao primjer možemo uzeti Bitcoin ASIC miner koji je dizajniran za računanje SHA-256).
ASIC mineri se oduvijek smatrju prjetnjom sustava rudarenja i kriptovaluta. Glavni su uzrok centralizacije hashing snage. GPU i CPU su u nepovoljnoj poziciji kraj ASIC-a ako se uzme u obzir rudarenje blokova i dobivanje nagrada za to, te tako ograničavaju većinu potencijalnih korisnika od rudarenja kriptovalute. ASICs su superiorniji CPU i GPU zato što mogu izračunati više hasheva po sekundi, tako da su rudari koji koriste ASIC u prednosti nad ostalima. S obzirom na navedene poteškoće s kojima se susreću kriptovalute te njihovi rudari, X11 algoritam je dizajniran kako bi bio efikasan i na CPU i GPU. Osmišljen je da bi bio čim duže ASIC resistant (ili duže nego ostale konkurentne kriptovalute) kako bi "hobbyists" mogli dulje sudjelovati u rudarenju, što je vrlo povoljno za Dash kriptovalutu, njeno širenje i dobru distribuciju.
-X11 algoritam je kombinacija 11 hash funkcija: BLAKE, BLUE MIDNIGHT WISH (BMW), Grøstl, JH, Keccak, Skein, Luffa, CubeHash, SHAvite-3, SIMD, ECHO. Poruka se predaje prvoj hash funkciji koja ga obradi i proslijedi svoj izlaz sljedećoj funkciji kao poruku.
X11 je sigurniji od SHA256. Sve hash funkcije koje se koriste kod X11 algoritma su bile kandidati kod traženja novog standarda, boljeg sigurnijeg algoritma SHA3 koji je temeljen na Keccak funkciji. Razvijeno je još algoritama koji se temelje na istoj ideji kao i X11: X13, X14, X15 i X17 koji koriste više hash funkcija.
Kriptovalute koje koriste X11 su Dash, Hatch, Pura, SmartCoin, CannabisCoin,

Influxcoin, StartCoin, Onix, i mnoge druge.[7]

2.3 SCRYPT

Još jedan u nizu hash algoritama koji su razvijeni s namjerom da se izbjegne stvaranje ASIC-ova te maksimalno oteža njihovo rudarenje kriptovaluta. Scrypt algoritam je vrlo memorijski zahtjevan - osim što zahtjeva od rudara da brzo stvaraju brojeve, ti brojevi se spremaju u RAM i treba im se pristupiti prije dobivanja konačnog hasha. Ovim pristupom se drastično smanjuje učinkovitost integriranih krugova specifične namjene.
Scrypt je osmislio Colin Percival za spremanje online sigurnosnih kopija UNIX operativnih sustava. Algoritam dodatno otežava rješavanje dodajući šum (noise) - nasumično generirane brojeve te tako povećava vrijeme potrebno za dobivanje hasha. Koriste ga kriptovalute Litecoin, Dogecoin, ProsperCoin, MonaCoin...
Algoritam je sačinjen nekoliko parametara:

- N - parametar koji označava koliko CPU/memorije traži algoritam
- p - paralelizacijski parametar, pozitivni cijeli broj
- r - veličina bloka
- S - "sol", nasumična vrijednost koja se često koristi kao dodatni ulaz u kriptografskim funkcijama. Služi kao zaštita od *Rainbow table* napada
- P - ulazna vrijednost, niz znakova koji želimo hashirati
- *dkLen* - željena duljina izlazne vrijednosti u oktetima

Navedene podatke prosljiedimo *key derivation* funkciji PBKDF2 (funkcija kojom izvlačimo tajne ključeve iz vrijednosti kao što su master key ili neke vrste lozinke). PBKDF2 je akronim za Password-Based Key Derivation Function 2, kojom smanjujemo ranjivost na napade grubom silom (brute force) tako da dobijemo izvedeni ključ.[8]

2.4 Dagger Hashimoto

Dagger Hashimoto je prethodnik, istraživački algoritam za kriptovalutu Ethereum 1.0, kasnije zamijenjen za algoritam Ethash. Razvijen je s namjerom da ispuni sljedeće zahtjeve:

- ASIC otpornost - mala mogućnost izrade ASIC-a koji će moći rudariti Ethereum ili kad se on uspije izraditi, minimalna isplativost ASIC-a naspram CPU
- širok spektar različitih jednostavnih uređaja, uređaja male snage, koji će moći potvrditi blok
- spremanje cijelog blockchain-a

Temelj algoritma Dagger Hashimoto činili su već poostojeći algoritmi:

- Hashimoto - autor algoritma je Thaddeus Dryja, a ASIC otpornost nastoji postići tako što je vezan za IO, ograničava pristupe memoriji tijekom rudarenja
- Dagger - razvio ga je Vitalik Buterin, ujedno i jedan od autora Ethereuma. Koristi acikličke grafove kako bi računanje bilo memorijski zahtjevno, ali i da bi validacija blokova istovremeno koristila relativno malo memorije. Alternativa isto tako memorijski zathjevnom algoritmu, Scryptu, odustalo se od njegova korištenja zbog otkrivenog manjka sigurnosti

[9]

2.4.1 Ethash

Ethash je ažurirana verzija Dagger Hashimota, iako se više ne može nazvati tako zato što je algoritam značajno promijenjen kako bi se ispravili svi nedostatci njegova prethodnika u zadnjim fazama razvoja.
The general route that the algorithm takes is as follows:

There exists a seed which can be computed for each block by scanning through the block headers up until that point. From the seed, one can compute a 16 MB

pseudorandom cache. Light clients store the cache. From the cache, we can generate a 1 GB dataset, with the property that each item in the dataset depends on only a small number of items from the cache. Full clients and miners store the dataset. The dataset grows linearly with time. Mining involves grabbing random slices of the dataset and hashing them together. Verification can be done with low memory by using the cache to regenerate the specific pieces of the dataset that you need, so you only need to store the cache. [10]

3 Zaključak

Literatura

[1] Kriptovaluta. https://en.wikipedia.org/wiki/Cryptocurrency. 09.08.2020.

[2] U. Mukhopadhyay, A. Skjellum, O. Hambolu, J. Oakley, L. Yu, and R. Brooks. A brief survey of cryptocurrency systems. In *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pages 745–752, 2016.

[3] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008. https://bitcoincore.org/bitcoin.pdf.

[4] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 139–147. Springer, 1992. doi: 10.1007/3-540-48071-4_10.

[5] Saif Al-Kuwari, James H. Davenport, and Russell J. Bradford. Cryptographic hash functions: Recent design trends and security notions. Cryptology ePrint Archive, Report 2011/565, 2011. https://eprint.iacr.org/2011/565.

[6] NIST. Descriptions of sha-256, sha-384, and sha-512. https://web.archive.org/web/20130526224224/http://csrc.nist.gov/groups/STM/cavp/documents/shs/sha256-384-512.pdf.

[7] Daniel Diaz Evan Duffield. Dash: A payments-focused cryptocurrency, 2014. https://github.com/dashpay/dash/wiki/Whitepaper.

[8] Colin Percival. Stronger key derivation via sequential memory-hard functions. 2009.

[9] Dagger hashimoto. https://github.com/ethereum/wiki/wiki/Dagger-Hashimoto, 15.08.2020.

[10] Ethash. https://eth.wiki/en/concepts/ethash/ethash, 15.08.2020.