KRIPTOGRAFIJA

1. Navesti razlike izmedju simetricnih i asimetricnih kriptosistema

► Kriptosistem je par: Algoritam za kriptovanje i algoritam za dekriptovanje
  ► Algoritmi su najcesce svima poznati
  ► Uvek zavise od parametra koji se zove kljuc, i koji se cuva u tajnosti (potpuno ili delimicno)
► Obicno:
  ► Alisa salje poruku Bobanu
  ► Cica krade sifrat i pokusava da ga dekriptuje ne znajuci kljuc (kriptoanaliza)
► Vrste kriptosistema:
  ► Simetricni kriptosistem: Alisa i Boban koriste isti kljuc za (de)kriptovanje. Unapred dogovore kljuc i cuvaju ga u tajnosti. Kljuc se periodicno menja. Neprakticno kada imamo veliki broj korisnika i svako komunicira sa svakim.
  ► Asimetricni kriptosistem (kriptosistem sa javnim kljucem): Alisa i Boban prave sopstvene kljuceve, kljuc za kriptovanje objavljuju, dok kljuc za dekriptovanje cuvaju u tajnosti.

2. Cezarova i afina sifra

Primer: Cezarova sifra
Slova A-Z kodiramo sa $\{0, \ldots, 25\} = Z_{26}$, protocna sifra: $$f(P) = P + b \bmod 26$$ Gde je $\in Z_{26}$ kodirani simbol, b = 16 tajni kljuc.   Alisa hoce da posalje poruku 'PAYMENOW', ona kodira   'PAYMENOW' → 15 0 24 12 4 13 14 22 $f\rightarrow$ 5 16 14 2 20 3 4 12 → 'FQOCUDEM'   Boban moze da procita poruku pomocu algoritma $$f^{-1}(C) = C - b \bmod 26$$

Uopstenje: afina sifra

$$f(P) = aP + b \bmod 26 \quad \text{i} \quad f^{-1}(C) = a'C + b' \bmod 26,$$
$$\text{gde je } a' = a^{-1} \text{ u } Z_{26}^{*} \text{ i } b' = -a^{-1}b$$

3. Kriptoanaliza Cezarove i afine sifre

$$f(P) = aP + b \bmod 26 \quad \text{i} \quad f^{-1}(C) = a'C + b' \bmod 26,$$

► Poznato: Najfrekventnije slovo u tekstu na engleskom jeziku je 'E'
► Cica pronalazi najfrekventnije slovo u sifratu, npr. neka je 'K', i pretpostavlja da je $f('E') = 'K'$, tj. $f^{-1}('K') = 'E'$

► Slicno, uporedjuje drugo najfrekventnije slovo i zakljucuje da je $f^{-1}('D') = 'T'$
► Cica resava sistem i pronalazi ključ.
► Ako sistem nema (jedinstveno) resenje: umesto 2. najfrekventnijeg slova moze koristiti 3, 4, ► Zakljucak: bolje raditi sa vecim blokovima slova (digrafovi)

Primer: Digrafovi
Par slova se kodira necim iz $\{0, 1, \ldots, 26^2 - 1 = 675\}$ Digraf 'NO' se kodira sa $26 \cdot N + 0 = 26 \cdot 13 + 14 = 352$, zatim se kriptuje $159_{\hookrightarrow kljuc} \cdot 352 + 580_{\hookrightarrow kljuc} = 440 \; mod \; 676$ sto je ekvivalent 'QY'

4. Matricno kriptovanje digrafa

► Neka je $A \in M_2(Z_n)$ invertibilna matrica, tj. takva da je $\det A$ invertibilno u Z_n
Npr. $n = 26$ i $A = (2\ 3\ 7\ 8)$

5. Jednosmerne funkcije. Navesti primer jednosmerne funkcije

► Kriptosistemi sa javnim kljucem koriste tzv. jednosmerne funkcije $f: X \to Y$
  ► ako je poznato $x \in X$ lako (brzo) se moze izracunati $y = f(x)$
  ► ali ako je poznato $y \in Y$ tesko je (neizvodljivo u realnom vremenu) izracunati x td. $y = f(x)$
► Dovoljno je da je f injekcija, a najcesce bijekcija
► Mogu se koristiti za:
  ► razmenu poruka (ali su sporiji od simetricnih sistema)
  ► razmenu kljuca koji ce koristiti za simetricne sisteme (bez neposrednog susreta)
  ► digitalni potpis
► Primer jednosmerne funkcije: SHA-256 funkcija za hesiranje

6. Difi-Helmanov algoritam za usaglasavanje kljuceva

► Ako je p prost broj, tada je $(Z_p, +_p, \cdot_p)$ je polje, gde je $Z_p = Z/(pZ) = \{0, 1, 2, \ldots, p-1\}$
► Multiplikativna grupa $Z_p^* = (Z_p \backslash \{0\}, \cdot_p)$ je ciklicna. tj. postoji generator (pritivni koren) $g \in Z_p \backslash \{0\}$ td. se svi elementi $Z_p \backslash \{0\}$ mogu videti kao stepeni g
► Difi-Helmanova razmena kljuca se zasniva na sledecem:
  ♣ ako znamo $g \in Z_p^*$ i $n \in N$ lako je odrediti g^n (tj. $g^n \; mod \; p$)
  ♠ ali ako znamo g i g^n tesko je odrediti n
► Algoritam:
  ► Alisa i Boban biraju prost broj p (priblizno 200-cifren) i generator $g \in Z_p^*$ i objavljuju p i q
  ► Alisa bira svoj tajni kljuc $a_A \in N$, racuna i objavljuje samo $g^{a_A} \; mod \; p$ (javni kljuc)
  ► Slicno, Boban bira tajni kljuc $a_B \in N$, racuna i objavljuje samo $g^{a_B} \; mod \; p$ (javni kljuc)
  ► I Alisa i Boban mogu izracunati
  $K = \left(g^{a_A}\right)^{a_B} mod \; p = \left(g^{a_B}\right)^{a_A} mod \; p$, i to ce biti njihov usaglasen kljuc
  ► Cica zna samo $q, \; g, \; g^{a_A}$ i g^{a_B}, i pomocu toga ne moze (brzo) da odredi K

7. Algoritam za stepenovanje ponovljenim kvadriranjem

► Brzi algoritam za ♣ tj. racunanje g^n u Z_p^* (sve operacije su po modulu p):

1. mozemo redukovati na $n < p - 1$ zbog Male Fermaove teoreme $g^{p-1} = 1$ u Z_p
2. Zapisati n binarno

$$n = \overline{n_r n_{r-1} \dots n_1 n_0} = \sum_{i=0}^{r} n_i \cdot 2^i,\ n_i \in \{0, 1\}$$

3. Izracunati

$$1,\ g,\ g^2,\ \left(g^2\right)^2 = g^{2^2},\ \left(g^{2^2}\right)^2 = g^{2^3},\ \left(g^{2^3}\right)^2 = g^{2^4},\ \dots,\ \left(g^{2^{r-1}}\right)^2 = g^{2^r}$$

4. g^n je proizvod onih g^{2^i} za koje je $n_i = 1$

   Dokaz:

$$g^n = g^{\sum_{i=0}^{r} n_i \cdot 2^i} = \prod_{i=0}^{r} g^{n_i \cdot 2^i} = \prod_{0 \le i \le r\ n_i = 1} g^{2^i}$$

► Vremenska slozenost:

1. $O(log^2 p)$ operacija jer je $\log log\ p \sim$ broj bitova broja p
2. $O(r \log log\ n) = O(log^2 p)$ operacija - jer r puta ponavljamo deljenje sa 2 i ostatak

   po modulu 2 (a $r \sim \log log\ n \ \le \log log\ p$)
3. r kvadriranja, a za svako kvadriranje treba po $O\left(log^2\left(g^{2^i}\right)\right) = O(log^2 p)$ operacija
4. najvise r mnozenja koja zahtevaju po najvise $O(log^2 p)$ operacija

► ukupno $O\left(r\ log^2 p\right) = O(log^3 p)$

► za mnozenje $g^n = gg \dots g$ bi trebalo $O\left(n\ log^2 p\right)$ operacija

► Napomena: algoritam radi i za modul m koji nije prost, ali moze da radi sporije jer umesto Male Fermaove koristi Ojlerovu teoremu.

8. Definisati diskretni logaritam. Navesti 3 kriptosistema koji se zasnivaju na problem diskretnog logaritma

Definicija
Neka je G grupa npr. F_q^*, i neka su $a,\ g \in G$. Najmanji prirodan broj n (ako postoji) takav da je $a = g^n$ zovemo diskretni logaritam od a u osnovi g i oznacavamo sa a

► 3 kriptosistema koji se zasnivaju na problemu diskretnog logaritma: Mesi-Omura, Difi-Helman, ElGamal

9. Mesi-Omura kriptosistem

► Koristi se za razmenu kljuceva ili poruka. Ako je duza poruka deli se na blokove

► Algoritam:
  ► Fiksira se konacno polje F_q i to je svima poznato (q je javni kljuc)
  ► Alisa i Boban biraju svoje tajne kljuceve e_A i e_B td.

$NZD(e_A,\ q-1) = NZD(e_{B,}\ q-1) = 1$

  ► Svako od njih racuna svoj tajni kljuc $d_i \equiv e_i^{-1}\ (mod\ q-1),\ i \in \{A,\ B\}$
  ► Ako je M blok poruke (kodiran elementom polja F_q) koju treba poslati Alisa racuna M^{e_A} i salje Bobanu
  ► Boban ne moze da procita M^{e_A}, ali moze da izracuna $\left(M^{e_A}\right)^{e_B} = M^{e_A e_B}$ i posalje nazad Alisi
  ► Sada Alisa racuna $\left(M^{e_A e_B}\right)^{d_A} = M^{e_B}$ i salje opet Bobanu. Ovde se koristi $e_A d_A \equiv 1\ (mod\ q-1)$ sto povlaci $M^{e_A d_A} = M$ u F_q
  ► Boban racuna $\left(M^{e_B}\right)^{d_B} = M$ i dolazi do pocetne poruke

10. Alisa salje Bobanu poruku pomocu Mesi-Omura kriptosistema, i pretpostavimo da je Cica videla celokupnu komunikaciju. Objasniti zasto Cica ipak ne moze da dekriptuje poruku

► Ako Cica presretne komunikaciju najvise sto moze da zna je M^{e_A}, M^{e_B} i $M^{e_A e_B}$ i javni kljuc q
► Da bi dosla do informacije M mora da izracuna:
  ► $e_A = \left(M^{e_A e_B}\right)$ diskretni logaritam
  ► $d_A \equiv e_A^{-1} (mod\ q-1)$
  ► $M = \left(M^{e_A}\right)^{d_A}$

11. ElGamalov kriptosistem

► Javni kljuc q(stepen prostog broja) i $g \in F_q^*$ generator
► Boban bira svoj tajni kljuc e_B i pomocu njega pravi javni kljuc g^{e_B} koji salje Alisi tj. objavljuje (kao kod Difi-Helmana)
► $M \in F_q$ kodirani blok (deo poruke) koju Alisa zeli da posalje, ona generise slucajan prirodan broj $k < q$ koji ce koristiti samo jednom za blok M (za naredno M bira novo k)
► Alisa salje Bobanu par informacija g^k i $Mg^{e_B k} = M\left(g^{e_B}\right)^k$ (ovo lako racuna jer zna $g,\ g^{e_B},\ k$ i M)
► Boban racuna $g^{e_B k} = \left(g^k\right)^{e_B}$, zatim njegov inverz i dobija $M = Mg^{e_B k}\left(g^{e_B k}\right)^{-1}$
► Cica mora da resi problem diskretnog logaritma da bi uradila prethodni korak
► Napomena: Boban samo jednom salje e_B Alisi (na pocetku), kod svakog bloka imamo jednu

razmenu (Alisa salje par $\left(g^k,\ M\,g^{e_B k}\right)$ Bobanu)

12. Kako se generise slucajan veliki prost broj

► U svim prethodnim algoritmima: izabrati kljuc = koristiti generator slucajnih brojeva
► Znamo kako radi generator (pseudo) slucajnih prirodnih brojeva
► Ali kako se generise slucajan prost broj?
► Nasumicno izabran broj verovatno nije prost
► Ne postoji formula za n-ti prost broj $p_n = \ldots$
► Princip rada generatora:
  ► izabere se neparan (veliki) pseudoslucajni broj n
  ► zatim se prost broj trazi u nizu $n,\ n+2,\ n+4,\ n+6,\ \ldots$
  ► treba nam efikasan nacin da proverimo da li je neki broj prost. Elementarno resenje je sporo, velika slozenost $O(\sqrt{n})$

13. Testovi primalnosti. Sta su ulazni i izlazni podaci kod testa primalnosti

► Testovi primalnosti su napravljeni tako da
  ► ako broj n padne na testu onda je n slozen
  ► ako broj n prodje test on moze (ali ne mora) da bude prost
  ► $v = \frac{card\{n\in[1,N] \mid n\ je\ prost\}}{card\{n\in[1,N] \mid n\ je\ prosao\ test\}}$ je verovatnoca da je broj koji je prosao test prost
  ► vece v – bolji test
  ► Test obicno zavisi od nekih parametara. Ponavljanje testa za razne (nezavisne) parametre povecava verovatnocu da je broj koji je preziveo sva testiranja zaista prost
► Ulazni parametar je broj n za kog proveravamo da li je prost, a izlazni parametar je v koji predstavlja verovatnocu da je broj prost.

14. Definisati pseudoproste i Karmajklove brojeve. Sta je glavni nedostatak Karmajklovog testa primalnosti

► Mala Fermaova teorema: (★) $a^{n-1} \equiv 1\ (mod\ n)$, za n prost i $NZD(a,\ n) = 1$
► Ali nije nemoguce da (★) vazi i ukoliko n nije prost

Definicija
Ako za prirodan broj a i slozen broj n td. $NZD(a,\ n) = 1$ vazi (★) kazemo da je n pseudoprost broj u bazi a

► Primer: Broj $n = 91 = 7\cdot 13$ je pseudoprost u bazi 3 jer je $3^{90} \equiv 1\ (mod\ 91)$, ali nije pseudoprost u bazi 2 jer je $2^{90} \equiv 64\ (mod\ 91)$
► Ojlerova teorema: $a^{\varphi(n)} \equiv 1\ (mod\ n)$
► Ako je n pseudoprost u bazi a mora biti $a^{n-\varphi(n)-1} \equiv 1\ (mod\ n)$
► $n - \varphi(n) - 1$ je obicno mnogo manji od $n - 1$
► U nasem primeru $91 - \varphi(91) - 1 = 18$, pa baze a u kojima je 91 pseudoprost treba traziti medju $a^{18} \equiv 1\ (mod\ 91)$

Teorema
1. Ako je n pseudoprost i u bazi a i u b onda je pseudoprost i u bazi ab 2. Ako je n pseudoprost u bazi a, ali nije pseudoprost u b onda nije pseudoprost ni u bazi ab 3. Ako n nije pseudoprost u bazi a onda nije pseudoprost ni u bazi b, za bar pola b-ova iz

$$Z_n^* = \{b \in Z_n \mid NZD(b, n) = 1\}$$

► Dokaz:

1. $(ab)^{n-1} = a^{n-1}b^{n-1} \equiv 1 \cdot 1 \ (mod\ n)$
2. $(ab)^{n-1} = a^{n-1}b^{n-1} \equiv b^{n-1} \not\equiv 1 \ (mod\ n)$
3. Za svaku bazu c u kojoj je n pseudoprost postoji baza $b = ca$ u kojoj n nije pseudoprost (prema 2.)

Definicija
Karmajklov broj n je slozen broj koji je pseudoprost u svakoj bazi $a \in Z_n^*$

► Vrlo efikasan test, ali ne odvaja proste od Karamajklovih brojeva

15. Miler-Rabinov test primalnosti

Miler-Rabinov test primalnosti
Neka je n neparan i $a \in Z$ td. $NZD(a, n) = 1$. Zapisemo $n - 1 = 2^r d$, gde je d neparan. $$a_j = a^{2^j d} \ mod\ p, \quad \text{za } j = 0, 1, \ldots, r - 1$$ Broj n prolazi Miler-Rabinov test u bazi a ako je ispunjen jedan od uslova 1. $a_0 = 1$ 2. Postoji $0 \leq s \leq r - 1$ td. $a_s = -1$

► Primetimo da je $a_{j+1} \equiv a_j^2 (mod\ p)$ i

1. $\Rightarrow$ svi $a_j = 1$
2. $\Rightarrow a_j = 1$ za $j > s$

► Za slozen n broj koji prolazi ovaj test u bazi a kazemo da je jako pseudoprost u bazi a

16. Veza izmedju pseudoprostih i jako pseudoprostih brojeva. Efikasnost Karmajklovog i Miler-Rabinov testa

Jako pseudoprost u bazi $a \Rightarrow$ pseudoprost u bazi a
Dokaz $a^{n-1} \equiv a_{r-1}^2 (mod\ n)$

► Posledica: Efikasnost Miler-Rabinovog testa $\geq$ efikasnost Karmajklovog testa

Teorema
1. Ne postoji slozen broj n koji je jako pseudoprost u svakoj bazi $a \in Z$ td. $NZD(a, n) = 1$ 2. Ako je n neparan slozen, onda on moze biti jako pseudoprost za najvise cetvrtinu baza $a \in Z_n^*$

► Efikasnost Miler-Rabinovog testa je najmanje $1 - \frac{1}{4^k}$, gde je k broj testiranja

17. Rivest-Samir-Ejdelman kriptosistem

► Alisa zeli da posalje poruku ili kljuc Bobanu
  ► Boban tajno bira dva velika prosta broja p i q, mnozi ih $n = pq$ i racuna $\varphi(n) = (p-1)(q-1)$
  ► Zatim Boban bira broj $1 \leq e \leq \varphi(n)$ td. $NZD(e, \varphi(n))$ i racuna $d = e^{-1} \bmod \varphi(n)$
  ► Boban salje Alisi javni kljuc (n, e), za d cuva kao svoj tajni kljuc. U ovom trenutku Boban moze da zaboravi p i q, ali je bitno da ih ne objavljuje
  ► Ako je $M < n$ kodirana poruka, Alisa racuna $N = M^e \bmod n$ i salje Bobanu
  ► Boban ima tajni kljuc d pomocu koga lako racuna $N^d \equiv M^{ed} \equiv M \pmod n$. Ovde se koristi $ed \equiv 1 \pmod{\varphi(n)}$ i Ojlerova teorema
  ► Cica vidi n, e i N, ali ne moze da dodje do poruke M sve dok ne odredi d tj. $\varphi(n)$

18. Prividno jednosmerne funkcije. Navesti primer prividno jednosmerne funkcije

► $f(M) = M^e \bmod n$ je primer prividno jednosmerne funkcije, to znaci da
  ► f je jednosmerna za Alisu i Cicu: one ne mogu da odrede f^{-1} u realnom vremenu
  ► f nije jednosmerna za Bobana: on je mogao da odredi f^{-1} jer je imao podataka vise (faktorizaciju n)
► Osnovna pretpostavka RSA kriptosistema: Ne moze se efikasno izracunati Ojlerova funkcija!
  ► Sustinski, Ojlerova funkcija je prividno jednosmerna, a onda se to prenosi i na $f(M) = M^e \bmod n$

Neka je $n = pq$ i neka je n poznato. Tada je $\varphi(n)$ poznato akko su poznati p i q
Dokaz: $(\Leftarrow)$ $\varphi(n) = (p-1)(q-1)$ $(\Rightarrow)$ Tada je poznati i $pq = n$ i $p + q = n - \varphi(n) - 1$ Vijetova pravila: p i q mogu odrediti kao koreni kvadratne jednacine $x^2 - (n - \varphi(n) - 1)x + n$

19. Digitalni potpis pomocu RSA kriptosistema

► Kako Alisa moze da ubedi Bobana u svoj indentitet? Tj. kako da Boban bude siguran da ne dobija poruke od Cice?
  ► Alisa generise javni kljuc (n, e) i svoj tajni kljuc d kao u RSA
  ► Boban generise (neku nasumicnu) poruku M i salje Alisi $M_1 = M^e \bmod n$
  ► Alisa treba da dekriptuje M_1 pomocu svoj kljuca d i vrati Bobanu $M_2 = M_1^d \bmod n$
  ► Boban poredi M_2 sa originalnom porukom M
  ► Cica ne moze (pre Alise) da odgovori Bobanu sta je M
  ► Cica ce na kraju videte poruku M_2, ali ne moze da je upotrebi jer Boban u sledecoj proveri generise novo M
► Drugacije uloge (u odnosu na preth.): Alisa ↔ Boban

Slanje poruke + digitalni potpis:

► Alisa treba da posalje poruku Bobanu, a da Boban bude siguran u njen identitet
  ► Alisa generise javni kljuc (n_A, e_A) i tajni kljuc d_A kao u RSA
  ► Boban generise javni kljuc (n_B, e_B) i tajni kljuc d_B kao u RSA
  ► $M < n_A, n_B$ kodirana Alisina poruka

- ► Alisa racuna $M_1 = M^{d_A} \, mod \, n_A$ i $M_2 = M_1^{e_B} \, mod \, n_B$ i salje Bobanu M_2
- ► Boban racuna $M_3 = M_2^{d_B} \, mod \, n_B$ i $M_4 = M_3^{e_A} \, mod \, n_A$, i upravo M_4 ce biti Alisina poruka

► Bitan redosled operacija:

- ► $M_3 \equiv_{n_B} M_2^{d_B} \equiv_{n_B} M_1^{e_B d_B} \equiv_{n_B} M_1 \Rightarrow M_3 = M_1$
- ► I slicno ce biti $M_4 = M$

20. Fermaov metod faktorizacije

► Pretpostavlja da je broj n proizvod dva prosta broja slicne velicine

► Napada slucaj koji deluje najoptimalnije za izbor kljuca n u RSA – tada bi elementarnim resetom tragali do $\sqrt{n}$

► $n = pq = s^2 - t^2$, gde su $s = \frac{p+q}{2}$ i $t = \frac{p-q}{2}$ prirodni brojevi

► Problem se svoji da nalazenje s i t, pri cemu je $s > \sqrt{n}$ i t malo

► U nizu $s_1 = \left[\sqrt{n}\right] + 1, \; s_2 = \left[\sqrt{n}\right] + 2, \; \ldots, \; s_i = \left[\sqrt{n}\right] + i, \; \ldots$ trazimo najmanji s_i td. je $s_i^2 - n$ potpun kvadrat tj. $t_i = \sqrt{s_i^2 - n}$ ceo broj (gde je $[\cdot]$ ceo broj)

► Tada je $p = s_i + t_i$ i $q = s_i - t_i$

21. Kriptoanaliza RSA Fermaovim metodom

► Naci inverz za mnozenje $\cdot_n \Leftrightarrow \varphi(n) = ? \Leftrightarrow$ rastaviti $n \Leftrightarrow$ naci pravi delilac od n

► Elementarno reseto (provera da li $p \mid n$ za $p \leq \sqrt{n}$) je presporo

► Osnovna pretpostavka RSA je da ne postoji efikasan nacin da se resi jedan od 4 prethodna problema (a samim tim i sva 4)

► Fermaov metod nam omogucava da lako nadjemo faktore p i q, pa samim tim i $\varphi(n)$ koje je onda $(p-1)(q-1)$

22. Navesti korake u algoritmu za Polardov $(p-1)$- metod

► Pretpostavke: $B \ll n$, $m = NZS(1, \; 2, \; \ldots, \; B)$ i n ima prost cinilac p td. $p - 1$ je B-gladak

► Polardov algoritam:

- ► Odabrati $2 \leq a \leq p - 1$ npr. $a = 2$
- ► Izracunati $x = a^m - 1 \, mod \, n$
- ► Ako je $x = 0$ promeniti a (npr. $a + 1$) i vratiti se na prvi korak
- ► Ako je $x \neq 0$ izracunati $g = NZD(n, \; x)$

► Ako je pp o B-glatkosti ispunjena, znamo da ce algoritam dati $g \in \{2, \; 3, \; \ldots, \; n-1\}$, $g \mid n$

► Ukoliko se pojavi $x = 1$ to znaci da pretpostavka o B-glatkosti nije ispunjena. Tada eventualno moze da se pokusa sa vecim B

23. Zasto se javni kljuc $n = pq$ u RSA ne moze izabrati tako da ne bude osetljiv na napad Polardovim metodom

► RSA je ranjiv ako je $n = pq$ td. jedan od $p - 1$ ili $q - 1$ je B-gladak

► Ne mozemo da proverimo B-glatkost $p - 1$ jer bismo se opet vratili na faktorizaciju

24. Kakvo poboljsanje donose elipticke krive u a) sigurnosti kriptosistema b) kriptoanalizi

► Mnogi kriptosistemi sa javnim kljucem (Difi-Helman, Mesi-Omura, ElGamal, …) imaju unapredjenu verziju koja se zasniva na eliptickim krivama

► Prednost: moguce je postici istu zastitu sa manjim kljucem koji koristi EK

► Primer: 256-bitni kljuc zasnovan na eliptickim krivama menja 3072-bitni kljuc

► EK se koriste u kriptoanalizi, posebno za napad na RSA. Cak i ako RSA ne koristi EK, moze biti napadnut algoritmom koji koristi EK

► Videli smo: Polardov $(p-1)$-metod faktorizacije je spor ako n nema prost cinilac p td. $p-1$ je B-gladak. Sa EK bice dovoljno da neki od $p+s$ (za malo s) bude B-gladak

25. Definisati i nacrtati elipticku krivu nad poljem realnih brojeva

► Elipticka kriva nad R je kriva definisana jednacina

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

gde su $a,\ b \in R$ td. $\Delta = -16\left(4a^3 + 27b^2\right) \neq 0$

► Na skup resenja se dodaje jos jedna "beskonacno daleka" tacka O, pa je

Definicija
$E(R) = \left\{(x,y) \in R \times R \mid y^2 = x^3 + ax + b\right\} \cup \{O\}$

► $E(R)$ se moze videti kao kriva u projektivnoj ravni $RP^2 = R^3 \text{—} \sim$

26. Definisati elipticku krivu nad konacnim poljem

► Nadalje: gde je q stepen prostog broja $p \neq 2,\ 3$

Definicija
$E\left(F_q\right) = \left\{(x,y) \in F_q \times F_q \mid y^2 = x^3 + ax + b\right\} \cup \{O\},$ gde su $a,\ b \in F_q$ td. $\Delta = -16(4a^3 + 27b^2) \neq 0$

► Sad je teze videti geometrijski, ali $\oplus$ i $\ominus$ se mogu racunati algebarski (pomocu formula u narednom pitanju "U 4. I 5. slucaju definicije…")

► $\left(E\left(F_q\right), \oplus, \ominus, O\right)$ je Abelova grupa

► Uobicajeno je da se pise $E\left(F_q\right)$, ali je mozda ispravnije $E\left(F_q; a, b\right)$ jer zavisi 3 parametra q, a i b

27. Definisati operacije na eliptickoj krivoj. Grupni zakon na eliptickoj krivoj

► Za $P = (x,y) \in E(R)$ definisemo $\ominus P = (x, -y)$ i $\ominus O = O$

► Za $P,\ Q \in E(R)$ definisemo sabiranje $P \oplus Q$:

1. ako je $P = O$: $O \oplus Q = Q$
2. ako je $Q = O$: $P \oplus O = P$
3. ako je $Q = \ominus P \neq O$: $P \oplus Q = O$
4. ako je $P,\ Q \neq O,\ Q \neq P,\ \ominus P$: povucemo pravu l kroz P i Q
   - 4.1 ili l sece E u jos tacno jednoj tacki $R(\neq P,\ Q)$
   - 4.2 ili je l tangentna na E u jednoj od tacaka P i Q, oznacimo je sa R
   - tada je $P \oplus Q = \ominus R$
5. ako je $P = Q \neq O,\ \ominus P$: povucemo tangentu l na $E(Q)$ u tacki P, on ce preseci $E(R)$

u jos jednoj tacki R (razlicitoj od P). Tada je $2P = P \oplus P = \ominus R$

► Slusaj 4.1 je najvazniji, sve ostalo su neki granicni slucajevi toga

Teorema (Grupni zakon na EK)
$(E(R), \oplus, \ominus, O)$ je Abelova grupa

► Gledano u projektivnom prostoru RP^2:
  ► EK $E(R)$ je dopunjena tackom $O = (0, 1, 0)$
  ► prava l je isto dopunjena beskonacno dalekom tackom $(x, y, 0)$ koja ne mora biti O
  ► Tada se $E(R)$ i l seku u tacno 3 (ne obavezno razlicite) tacke $P, Q, R \in RP^2$ td. $P \oplus Q \oplus R = O$

► U 4. i 5. slucaju definicije $\oplus$ mozemo da izvedemo jednacinu prave l i zatim nadjemo njen presek (zajednicko resenje) sa eliptickom krivom. Ako su $P = (x_1, y_1)$ i $Q = (x_2, y_2)$ dobijamo $P \oplus Q = (x_3, y_3)$ gde je

$$\textbf{Za } P \neq Q [x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2;\ y_3 = -y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3).$$

$$\textbf{Za } P = Q [x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1;\ y_3 = -y_1 + \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3).$$

($x_1 \neq x_2$ u 4. slucaju i $y_1 \neq 0$ u 5.)

28. Haseova teorema za broj tacaka na eliptickoj krivoj. Zasto rad sa grupom $(E(F_q), \oplus)$ nudi vise mogucnosti od grupe $(F_q \setminus \{0\}, \cdot)$

Haseova teorema
Kardinalnost grupe $E(F_q)$ je $q + 1 + s$, gde je $s \leq 2\sqrt{q}$.
Dodatno, za svaku celobrojnu vrednost $s \in [-2\sqrt{q}, 2\sqrt{q}]$ postoje $a, b \in F_q$ td. je $\lvert E(F_q) \rvert = q + 1 + s$

► Kljucna ideja:
  ► Umesto grupe $(F_q^*, \cdot)$ koristiti grupu $(E(F_q), \oplus)$
  ► Umesto fiksne kardinalnosti $q - 1$ imamo slobodu $[q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}]$
  ► Stepenovanje g^n se menja sa $nP = P \oplus P \oplus \ldots \oplus P$ (n puta)
  ► Primetimo da nP moze biti i O
  ► nP se racuna ponovljenim dupliranjem tacke (kao ponovljeno kvadriranje)
    ► Primer: Za $100P$ zapisemo binarno

$$100 = 2^6 + 2^5 + 2^2 = \overline{1100100}_2, \text{ tada je}$$

$$100P = 2(2(P \oplus 2(2(2(P \oplus 2P)))))$$

  ► nP se racuna sa $O(\log n \log\log n)$ operacija $\oplus$, a svaki $\oplus$ se realizuje sa nekoliko sabiranja, oduzimanja, mnozenja …

29. Problem diskretnog logaritma nad eliptickim krivama

► Stepenovanje g^n se menja sa $nP = P \oplus P \oplus \ldots \oplus P$ (n puta)

► Primetimo da nP moze biti i O

► nP se racuna ponovljenim dupliranjem tacke (kao ponovljeno kvadriranje)

Problem diskretnog logaritma
Ako je poznato P i nP odrediti n

► U praksi, ovo se resava jos sporije od diskretnog logaritma u F_q^*

30. Kodiranje i dekodiranje podataka pomocu eliptcke krive

► Kodiranje podataka pomocu EK:

► Ako je $q = p$ prost broj:

► Ovaj metod ce raditi uspesno sa verovatnocom $1 - \frac{1}{2^k}$, pri cemu k sami biramo, u praksi $k \in [30,\ 50]$

► Poruka koja se kodira se po potrebi deli na blokove m duzine N td. $Nk < q$

► Prvo se m prevede u numericki ekvivalent M (kao ranije)

► M treba kodirati tackom (x_0, q_0) sa krive $E: y^2 = x^3 + ax + b$ nad (Z_p)

► Pokusa se sa $x_0 = Mk + 1$, ukoliko $y^2 = x_0^3 + ax_0 + b$ ima resenja, izaberemo jedno takvo y_0, a ako nema nastavljamo sa $M + 2,\ M + 3,\ \ldots,\ M + k - 1$ sve dok ne pronadjemo (x_0, y_0)

► Dekodiranje podataka pomocu EK:

► Imamo tacku $(x_0,\ y_0)$, treba rekonstruisati poruku m:

► Za $q = p$ prost: m dobijamo kao $\left[\frac{x_0}{k}\right]$

31. Difi-Helmanovo usaglasavanje kljuca nad eliptickim krivama

► Javni kljuc: konacno polje F_q, elipticka kriva E: $y^2 = x^3 + ax + b$ nad F_q i tacka $P = (x_0, y_0) \in E(F_q)$, odnosno parametri $(q,\ a,\ b,\ x_0)$

► Pozeljno je da P bude generator grupe $(E(F_q),\ \oplus)$

► Alisa i Boban biraju svoje tajne kljuceve $a_A,\ a_B < |E(F_q)|$

► Zatim racunaju javne kljuceve $a_A P,\ a_B P \in E(F_q)$ i razmenjuju ih (objavljuju)

► Usaglaseni kljuc ce biti $K = (a_A a_B)\, P \in E(F_q)$

► Alisa moze da izracuna K kao $K = a_A(a_B P)$. I slicno Boban dolazi do K

► Cica vidi samo $a_A P$ i $a_B P$, ne i K

► Zadatak: Elipticka kriva koja se koristi za problem usaglasavanja kljuceva Diffie-Helman protokola je $E: y^2 = x^3 + 3x + 8$ nad poljem F_{13}. Ako se koristi generator $P = (2,\ 3)$, tajni kljucevi $e_A = 4,\ e_B = 5$, odrediti tacku koja se dobija kao rezultat usaglasavanja

32. ElGamalov kriptosistem nad eliptickim krivama

► Javni kljuc: konacno polje F_q, elipticka kriva $E: y^2 = x^3 + ax + b$ i tacka

$P = (x_0, y_0) \in E(F_q)$, odnosno parametri $(q,\ a,\ b,\ x_0)$
  ► Pozeljno je da P bude generator grupe $(E(F_q),\ \oplus)$

► Boban bira svoj tajni kljuc $e < |E(F_q)|$ i pomocu njega racuna javni kljuc $eP \in E(F_q)$

► Za svaki kodirani blok poruke $M \in E(F_q)$ Alisa generise slucajan broj $k < |E(F_q)|$ i salje Bobanu tacke kP i $M \oplus keP$, gde keP dobija mnozeci tacku eP (koju je dobila od Bobana) sa k

► Boban tacku keP moze dobiti tako sto kP pomnozi sa e

► Boban sabira tacke $M \oplus keP$ i $\ominus keP$ i dolazi do M

► Cica vidi samo eP i kP, $M \oplus keP$ i mora da resi problem diskretnog logaritma da bi dosla do poruke M

33. Lenstrin metod faktorizacije

► Faktorizacija pomocu EK

► Hocemo da faktorisemo broj n za koji verujemo da je slozen

► Lenstrin metod:
  ► pretpostavka: n ima prost cinilac p td. kardinalnost $|E(Z_p)|$ je B-gladak broj za neko malo B (inspirisano Polardovim $(p-1)$-metodom)
  ► ne pogadjati koje m radi vec pokusati sa $m = NZS(1,\ 2,\ \ldots, B)$ ili sa $m = B$! (m ne zavisi ni od n ni od p)
  ► znamo da $|E(Z_p)|$ deli ove m, pa ce biti $mP = O$ na $E(Z_p)$ tj. pojavice se imenilac g koji je deljiv sa p
  ► ali necemo racunati $mod\ p$, vec $mod\ n$

► Dakle, racunamo mP na $E(Z_n)$. Moze da se desi:
  ► ako uspemo da izracunamo mP bez problema – pretpostavka o B-glatkosti nije ispunjena, pokusati sa vecim B ili drugom eliptickom krivom
  ► ako se kao imenilac pojavi g deljiv sa n – promeniti tacku P
  ► ako se kao imenilac pojavi g koje nije deljivo sa n, ali nije ni invertibilno po modulu n – onda je $NZD(g,\ n)$ pravi delilac n

ZERO KNOWLEDGE PROOFS

1. ZKP i ilustrativni primeri

ZKP je kriptografska tehnika pomoću koje dokazivač dokazuje verifikatoru da zna neku informaciju bez otkrivanja same informacije.
Nisu dokazi u klasičnom matematičkom smislu, već su probabilistički dokazi tj. postoji minimalna verovatnoća da dokazivač ne zna informaciju, a da će dokaz biti prihvaćen od strane verifikatora.
Postoje dva glavna tipa dokaza sa nula znanja: interaktivni i neinteraktivni

Primeri:

- *Gde je Valdo?* – dokazivanje poynavanja lokacije nečega/nekoga unutar složenog skupa podataka, bez otkrivanja stvarne pozicije
- *Ali Babina pećina* – izazov je da dokazivač ubedi verifikatora da zna tajnz reč bez otkrivanja iste. Dokazivač ulazi u pećinu i bira jedan put, dok verifikator čeka napolju. Zatim verifikator nasumično traži od dokazivača da izađe kroz jedan od puteva. Dokazivač može otvoriti vrata koristeći tajnu reč, ako je to potrebno da bi ispunio zahtev verifikatora, čime dokazuje poznavanje reči bez njihovog otkrivanja.
- *Prijatelj daltonista* – kako dokazati daltonisti da sud ve lopte različitih boja bez otkrivanja samih boja. Jedan pristup je korišćenje serija zamene izmedju lopti, koje kontroliše daltonista, gde on može testirati da li dokazivač i dalje može identifikovati lopte kao različite.

2. Primene ZKP-a

Blockchain – za povećanje privatnosti transakcija
Finansije – za bezbednu razmenu podataka i verifikaciju bez otkrivanja podataka, npr. aplikant za kredit može dokazati da mu je plata unutar određenog opsega, bez otkrivanja tačnog iznosa. Slično tome, može dokazati da je iznos plaćanja unutar granice, ali ne prikazuje tačan iznos
Online glasanje – poboljšanje integriteta i privatnosti u sistemima za online glasanje; primer MACI (Minimum Anti-Collusion Infrastructure)
DIDs – za što bolju verifikaciju identiteta bez otkrivanja više informacija nego što je potrebno. Decentralizovana identifikacija daje pojedincu mogućnost da kontroliše pristup ličnim identifikatorima(npr. dokazivanje svog državljanstva bez otkrivanja poreskog ID-a ili detalja pasoša)
Mašinsko učenje – verifikovanje rezultata ML modela bez otkrivanja podataka ili modela; primena mašinskog učenja na neke osetljive podatke gde bi korisnik mogao znati rezultat inferencije modela na svojim podacima, a da pritom ne otkriva svoj ulaz trećoj strani (npr. u medicinskoj industriji).
Autentifikacija – dokazivanje da znamo neke informacije bez otkrivanja istih. Jednom kada je ZK- dokaz generisan korišćenjem javnih i privatnih ulaza, korisnik ga jednostavno može prezentovati radi autentifikacije svog identiteta kada mu je potrebno pristupiti usluzi.

3. Merkle Tree i ZK dokaz pripadnosti skupu

Merkle Tree je binarno stablo kod koga listovi sadrže heš transakcija, a unutrašnji čvorovi sadrže heš kombinaciju dece. Ovo stablo omogućava efikasnu pretragu sadržaja velike strukture podataka. Ispitivanje pripadnosti list-čvora stablu se odvija u logaritamskom vremenu.
Blok se sastoji od zaglavlja i tela. Zaglavlje, između ostalog, sadrži koren Merkle stabla, a telo sadrži sve potvrđene informacije o transakcijama u bloku.
Dokaz pripadnosti skupu se odvija preko protokola dokazivača i verifikatora, gde dokazivač ne sme otkriti informacije o samom dokazu. Ovo osigurava da se članstvo u skupu može dokazati bez otkrivanja detalja koji bi mogli otkriti sam sadržaj skupa ili informacije koje nisu namenjene javnosti.

4. Completeness, soundness i ZK

ZK dokaz mora zadovoljiti sva tri koraka:
Completeness – verifikator mora prihvatiti ispravan dokaz
Soundness – verifikator ne bi trebalo da prihvati netačan dokaz
Zero Knowledge – verifikator kroz javne parametre neće ništa naučiti o dokazu; ovo osigurava da čak i nakon što je dokaz potvrđen kao ispravan, verifikator neće dobiti nikakve dodatne informacije o sadržaju ili detaljima samog dokaza, osim onih koji su već javno poznati.

5. Ciklična grupa (Zp*,·)

$Z_p^* = \{1, \dots, p-1\}$ gde je p prost broj sa operacijom · definisanom kao $a \cdot b = a * b \bmod p$
Svojstva grupe:
Zatvorenost - ako su elementi a i b iz grupe, tada je i a · b takođe u grupi;
Asocijativnost - ako su elementi a, b, c iz grupe, tada je a · (b · c) = (a · b) · c takođe u grupi;
Neutral - ako je element a iz grupe, tada je a · 1 = a takođe u grupi;
Inverz - ako je element a iz grupe, tada postoji element b iz grupe tako da je a · b = 1;

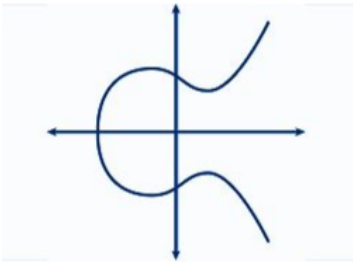
Neki element *a* je generator ciklične grupe Z_p^* ukoliko se svi ostali elementi te grupe mogu dobiti preko njega, tako da je $a_i = a^i \ mod \ p$
Ciklična grupa mora sadržati makar jedan generator.
$Z_p^* = \{1, \dots, p-1\}$ gde je p prost broj će uvek biti ciklična grupa.

Teorema pomoću koje brzo nalazimo generatore ciklične grupe:
Neka je g element ciklične grupe Z_p^*. On će biti generator te grupe akko $g^{((p-1)/q)} \neq 1 \bmod p$, za **svaki** prost broj q tako da q | (p-1)

6. Problem diskretnog logaritma

Neka je g generator ciklične grupe $Z_p^* = \{g, \dots, g^{p-1}\}$ gde je p prost broj.
Ako su g i p uzajamno prosti brojevi, tada na osnovu MFT važi $g^{p-1} = 1 \ mod \ p$ i $Z_p^* = \{1, g, \dots, g^{p-2}\}$
Broj x će biti diskretni logaritam od b u bazi g ako važi $\log_g b = x \Leftrightarrow g^x = b$ u grupi Z_p^*
gde je b element grupe Z_p^*, a g njen generator.

7. Eliptičke krive nad konačnim poljem



EK su grupe definisane nad konačnim poljem. EK:
$y^2 = x^3 + ax + b$, a, b iz F_q^*
EK su Abelove grupe ($E(F_q^*)$, +), pretpostavljamo da su EK nad konačnim poljem cikličke.
Operacije -, +, nP nad tačkama EK.
Tačke EK nad konačnim poljem su bukvalno samo tačke na grafiku

(lin. reg.), nema krive.
Dokazano je da skup tačaka u ECC-u ("Elliptic Curve Cryptography") uvek formira Abelovu grupu sa sledećim svojstvima:
Zatvorenost: Ako tačke P i Q pripadaju E(K), onda i P + Q pripada E(K);
Asocijativnost: (P + Q) + R = P + (Q + R); Identitet: Postoji identični element 0 takav da je P + 0 = P;
Inverz: Svaki element P ima inverz Q takav da je P + Q = 0;
Komutativnost: P + Q = Q + P;
Pretpostavićemo da je eliptična kriva nad konačnim poljem ciklična.

8. Add and Double algoritam

Računanje nP je jako sporo za veliko n, jer sabiramo P sa samim sobom n puta
nP računamo tako što n zapišemo binarno i time svodimo na logaritamski broj sabiranja.
Npr. $79*P = 2^6*P + 2^3* P + 2^2*P + 2^1*P + 2^0*P$

9. Multi-Scalar-Multiplication (bucket metod)

Usko grlo u algoritmima za dokazivanje kod većine EK zasnovanih na SNARK sistemima je Multi-Scalar-Multiplication algoritam.
Naivni algoritam koristi Add-And-Double algoritam, ali najbrži pristup je varijanta Pippengerovog algoritma koji nazivamo bucket metod.
Koraci algoritma:
- *Pozicioniranje skalara:* Svaki skalar se particioniše na *m* delova, tako da svaki deo sadrži *w* bitova
- *Akumulacija:* Paralelno obrađivanje skalara i tačaka i akumulacija rezultata u bakete
- *Optimizacija:* Tabele rezultata, paralelizacija, izbor eliptičke krive

10. Problem diskretnog logaritma nad eliptičkim krivama

- poznato je *P* i *mP*, a treba pronaći *m*
- analogno kod ciklične grupe Z_n* za g^n
- glavna operacija nad eliptičkim krivama
- jednosmerna funkcija
- 29. pitanje u kriptografiji

11. Uparivanje na eliptičkim krivama

Neka je dato $E(F_q)$ i G1 i G2 su podgrupe reda p, gde je p prost broj od EK.
Neka je g1 generator od G1, a g2 od G2.
Funkcija e: G1 x G2 → G_T, gde je G_T multiplikativna podgrupa od pola F_q reda p je uparivanje nad EK ako je zadovoljeno:
1) e (g1, g2) ≠ 1
2) e (P+Q, R) = e (P,R) * e (Q, R)
3) e (P, Q+R) = e (P,Q) * e (P, R)

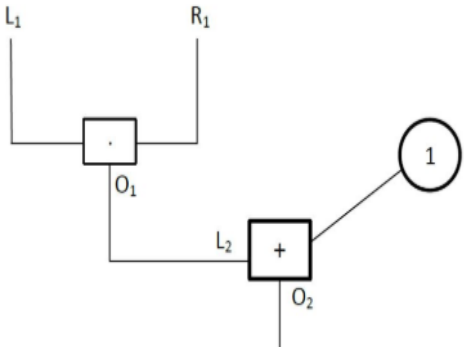
Tipovi bilinearnog uparivanja:
1. Ako je G1 = G2 i *e* je simetrično bilinearno uparivanje.
2. Ako G1 ≠ G2 i postoji efikasan homomorfizam ϕ: G2 → G1, ali ne i u suprotnom smeru
3. Ako G1 ≠ G2 i ne postoji efikasan homomorfizam između G2 i G1

12. STARKs & SNARKs

ZK SNARK – sažeti dokaz, brza verifikacija, koriste eliptičke krive (trusted setup - "povereno postavljanje" (trusted setup), što znači da se oslanja na početne parametre koji se veruju), veličina dokaza i vreme za verifikaciju zavise od aritmetičkog kola
ZK STARK – ne koriste eliptičke krive i ne zahtevaju povereno postavljanje, koristeći umesto toga heš funkcije, ovo ih čini transparentnijim i manje podložnim potencijalnim sigurnosnim rizicima vezanim za povereno postavljanje.

13. Aritmetizacija i sistem ograničenja (system constraints) kod ZKP-a

Aritmetizacija predstavlja prevođenje problema u aritmetičko kolo, koje prevodimo u ograničenja, koje prevodimo u polinome.
Aritmetizacija koristi operacije + i * gde ulazni podaci i izlaz moraju biti iz konačnog polja.
Verifikator proverava da li se izlaz aritmetičkog kola podudara sa javnom heširanom vrednošću dokazivača.



Gate constraints:
(1) $L_1 * R_1 - O_1 = 0$
(2) $L_2 + 1 - O_2 = 0$

Copy constraints:
(1) $L_1 = R_1$
(2) $O_1 = L_2$

Konstrukcija polinoma:
$L(1) = L_1$, $L(2) = L_2$, $R(1) = R_1$, $R(2) = R_2$, $O(1) = O_1$, $O(2) = O_2$
PLONK: $Lq_l + Rq_r + Oq_o + q_c + LRq_M = 0$

Poenta je da zapišemo deo kola preko polinoma koji pokriva celo kolo tako što podešavamo koeficijente uz delove koje želimo prikazati na 1, a ostale koeficijente na 0.

14. Komitmenti pomoću polinoma (Polynomial Commitments) kod SNARK-ova

Komitovanje predstavlja opredeljenje za neku vrednost tako da je ne smemo otkriti.
Komitovanje preko polinoma igra ključnu ulogu pri izgradnji efikasnih ZNP.
Omogućava dokazivanje ispravnosti polinoma, bez otkrivanja polinoma.
Najčešći tip polinomskog komitovanja je KZG, a koriste se još i Dory20, Dark20 i FRI.

15. Trusted setups kod Groth16 i PLONK-a

Trusted setup je procedura koja se obavlja jednaput radi generisanja podataka koji se koriste svaki put kada se neki kriptografski protokol pokrene.
Groth16 zahteva specifičan trusted setup za svako aritmetičko kolo, gde se koristi nasumično odabrana pomoćna tačka sa EK kako bi se sprečilo lažiranje dokaza.
PLONK nudi univerzalni i ažurirajući trusted setup, gde se koristi nasumično odabrana pomoćna tačka sa EK koja je nezavisna od kola.
Međutim, PLONK ima veće veličine dokaza što utiče na troškove gasa u Eterijum mreži.
Transparentni setup ne koristi tajne podatke (pomoćne tačke sa EK).

16. Non-Interactive Preprocessing argument system

Najpre se obavlja faza preprocesiranja u kojoj se generišu informacije koje dokazivač koristi da konstruiše dokaz koji će poslati verifikatoru.
Verifikator koristi samo taj dokaz i preprocesirane informacije da proveri ispravnost tvrdnje.

17. KZG

KZG je kriptografska šema koja se koristi pri komitovanju preko polinoma.
Omogućava dokazivanje ispravnosti polinoma, bez otkrivanja polinoma.
Generiše se komit za polinom i šalje se verifikatoru koji proverava da li je vrednost polinoma u određenoj tački zaista 0.
Faze KZG:

- setup – odaberemo nasumičnu tačku i parametre $H_0 = G$, $H_1 = Gs$, $H_2 = Gs^2$ …
- commit - com(f) = f(s) * G, gde s pripada konačnom polju F_p, a G je generator
- evaluate – $f(x_0) = y$, gde je x_0 nula $f(x) – y$ i $x – x_0 \mid f(x) – y$
  $f(s) – f(z) = (s – z) * h(s)$

18. PLONK

PLONK je ZNP sistem koji pripada SNARK grupi sistema.
To je univerzalni sistem, što znači da je potrebno inicirati trusted setup samo jednom koji će da važi za svako aritmetičko kolo. Trusted setup se sastoji od inicijalnih parametara koji se koriste tokom verifikacije.
PLONK se oslanja na komitovanje preko polinoma, gde dokazujemo ispravnost polinoma bez njegovog otkrivanja. Komitovanje preko polinoma generiše komit za neki polinom.
PLONK koristi permutacije baziranje na Langranžovim osnovama za definisanje ograničenja u kolu Ograničenja se sastoje od ulaznih podataka svake kapije i od vektora koeficijenata za svaku kapiju.
PLONK se može koristiti u Non Interactive Preprocessing Argument Systems.

19. Protokol Semafor

Semafor je ZKP protokol koji omogućava korisnicima da šalju poruke kao dokazani članovi grupe, bez otkrivanja identiteta. Takođe, onemogućuje ponovljeno slanje poruka.
Koristi se u privatnom glasanju, otkrivanju nepravilnosti, anonimni DAO i mikseri.
Semafori omogućavaju korisnicima da kreiraju objekat semafora i da ga dodaju u grupu kako bi poslali proverljiv anonimni signal.
Kolo semafora predstavlja jezgro ovog protokola i sastoji se od:

1. dokaza pripadnosti
2. anulirajućeg heša
3. signala

Kolo hešira anulirajući heš kako bi generisao identitet komita preko koga proverava dokaz pripadnosti u korenu Merkle drveta.

BLOCKCHAIN

1. Osobine blockchain-a

Danas, zbog jedinstvenih karakteristika blokčejna, on se koristi u različitim aplikacijama širom različitih industrija i slučajeva upotrebe. Neke od važnih osobina blokčejna su opisane u nastavku:

- *Poboljšana sigurnost*: Tehnologija blokčejna podržava poboljšanu sigurnost decentralizacijom svih informacija smeštenih u blokčejn mreži. Ovde, informacije ne mogu biti manipulisane, a heš jednog čvora je povezan sa prethodnim čvorom; promene u hešu jednog čvora dovode do promena u hešovima svih čvorova.

- *Decentralizovana mreža*: To znači da nema upravljačkog ili centralnog autoriteta u blokčejn mreži. Ovde, grupa čvornih čvorova kontroliše celu mrežu čineći je decentralizovanom.

- *Nepromenljivost*: Podaci u blokčejn mreži se smatraju trajnim zapisima transakcija. Jednom kada se blok doda u mrežu, ne može se izmeniti ili izbrisati. Ovde, svaki čvor blokčejn mreže ima repliku digitalne knjige, i svaki čvor proverava validnost transakcije kada se doda u mrežu. U blokčejn mreži, ako većina čvorova potvrdi transakciju, ona se zatim dodaje u javnu knjigu. Ovo osigurava poverenje u blokčejn mrežu.

- *Transparentnost*: Tehnologija blokčejna je dizajnirana da kontroliše korisničke podatke na transparentan način. Ovde, blokovi podataka su dostupni svakom čvornom čvoru i čvorni čvorovi mogu koristiti podatke prema svojoj potrebi.

2. Prednosti i mane blockchain-a

Prednosti:

- *Integritet podataka*: U blokčejn mreži, detalji transakcija koji su dodati u mrežu ne mogu se urediti, tj. informacije su nepromenljive. Na ovaj način, blokčejn mreža podržava integritet podataka, kao i visoku sigurnost.

- *Verifikacija*: Ovde, podaci se skladište na decentralizovan način, pa svi korisnici blokčejn mreže mogu lako proveriti ispravnost podataka koristeći ZKP.

- *Decentralizacija*: Budući da se hiljade uređaja koristi za skladištenje podataka u blokčejn mreži, ceo sistem, kao i podaci, visoko su otporni na bilo kakav tehnički kvar i zlonamerne napade.

- *Pratljivost*: Tehnologija blokčejna je dizajnirana na takav način da može stvoriti neopozivu revizorsku stazu, čineći je dostupnom i lako dostupnom za praćenje bilo koje informacije te lanca.

- *Sigurnost*: Blokčejn mreža je osigurana jer svaki entitet te mreže dobija jedinstveni heš koji je povezan sa prethodnim čvorom. Takođe, enkripcijska tehnika blokčejna otežava hakovanje cele blokčejn mreže.

- *Brže procesiranje*: Tradicionalni bankarski sistemi zahtevaju mnogo vremena za procesiranje i završetak transakcija. Međutim, nakon uvođenja tehnologije blokčejna, brzina transakcija se značajno povećava. To smanjuje vreme gotovo na minutu ili čak sekundu.

- *Bez mešanja treće strane:* Danas nijedna finansijska institucija ili vlada nema kontrolu nad bilo kojom kriptovalutom, koja se uglavnom koristi korišćenjem tehnologije blokčejna. To znači da nema mešanja treće strane u blokčejn mrežu.

- *Automatizacija*: Pametni ugovori i dostupnost podataka u svakom čvoru smanjuju složenost procesa validacije. To zauzvrat pomaže u automatizaciji procesa i poboljšava brzinu transakcija.

Mane:

- *Potrošnja energije*: Potrošnja energije u blokčejn mreži tokom rudarskog procesa je relativno visoka. To je uglavnom zato što kada se novi čvor dodaje u mrežu, mora biti komunikacije sa svim drugim čvorovima istovremeno. Održavanje realnog vremenskog glavne knjige je takođe jedan od glavnih razloga za potrošnju energije.

- *Nedozrelost:* Ipak, tehnologija blokčejna je u ranom stadijumu, pa obični ljudi nemaju mnogo poverenja u nju zbog nedostatka znanja. Ljudi nisu spremni da ulažu u nju. Međutim, nekoliko aplikacija baziranih na blokčejnu efikasno funkcioniše u različitim industrijama.

- *Vremenski zahtevno*: Da bi se novi blok dodao u blokčejn mrežu, rudari moraju izračunati vrednost nonce-a. U mnogim slučajevima, ovaj proces računanja je vrlo vremenski zahtevan, kao i troši mnogo resursa. Želi se ubrzati proces korišćenja tehnologije blokčejna u različitim industrijskim sektorima.

- *Pravne formalnosti i standardi*: Danas, mnoge zemlje su počele da rade na tehnologiji blokčejna. Međutim, to postaje prepreka za prihvatanje Bitkoina u mnogim zemljama od njihovih finansijskih institucija. Osim toga, pošto je blokčejn još uvek u svojoj ranoj fazi, ne postoje specifični standardi.

- *Napadi od 51%*: Normalno, blokčejn mreža se smatra sigurnom. Međutim, mogu se pojaviti neki sigurnosni napadi u blokčejn mreži, a napad od 51% je jedan od najčešćih napada. Ovaj napad se dešava kada entitet može upravljati sa više od 50% čvorova blokčejn mreže, što mu omogućava da poremeti blokčejn mrežu namerno modifikujući ili isključujući redosled transakcija.

- *Robusnost mreže:* Sve aplikacije bazirane na blokčejnu imaju osnovnu poslovnu logiku. Ova logika opisuje kako aplikacije funkcionišu u smislu poslovnih zahteva, i to je fiksna logika koja se ne može preoblikovati nakon razvoja aplikacije.

- *Težina razvoja:* Primena kompleksnih protokola za postizanje konsenzusa je veoma važna. Ovde, brza implementacija procesa nije moguća i potrebno je solidno oznavanje različitih programskih jezika za razvoj aplikacije bazirane na blokčejnu.

- *Skladištenje:* U blokčejn mreži, svaki blok ili transakcija se dodaju u lanac ili mrežu, što povećava veličinu baze podataka, a veličina knjiga se vremenom povećava. Trenutno, Bitkoin zahteva oko 200 GB skladištenja. Dakle, u mnogim slučajevima, to predstavlja problem za korisnike.

- *Razmera*: Razmera se smatra jednim od glavnih nedostataka tehnologije blokčejna jer je veličina bloka blokčejn mreže fiksna. Ako je veličina bloka bilo koje blokčejn mreže 1 MB, može sačuvati manje detalja transakcija na određenom bloku.

3. Primene blockchain-a

Bankarstvo i finansije:

- Međunarodna plaćanja: Blockchain tehnologija stvara neizmenjiv zapis o osetljivim informacijama ili podacima, što podržava upotrebu ove tehnologije u međunarodnim novčanim i platnim transferima.

- Kapitalna tržišta: Sistem zasnovan na blockchain tehnologiji takođe može poboljšati kapitalna tržišta. U kapitalnim tržištima, blockchain tehnologija može uticati na konsolidovani revizorski trag, brže poravnanje i namirenje, i operativna poboljšanja.
- Trgovinsko finansiranje: Blockchain tehnologija ima potencijal da racionalizuje poslove trgovinskog finansiranja. Podržava preduzeća da lako vrše transakcije jedni s drugima izvan geografskih granica.

- Regulatorna usaglašenost i revizija: Kako je blockchain tehnologija sigurna, može se koristiti za korisne revizije i računovodstvo. To je zato što blockchain tehnologija značajno smanjuje ljudsku grešku, kao i osigurava integritet podataka. Pored toga, korisnici ili vlasnici podataka mogu menjati bilo koje podatke ili zapise, kada se skladište pomoću blockchain tehnologije.

- Zaštita od pranja novca: Zbog poboljšane sigurnosne funkcije blockchain tehnologije, pogodna je za borbu protiv pranja novca.

- Osiguranje: U industriji osiguranja, blockchain tehnologija može se koristiti za smanjenje prevare u osiguranju, pojednostavljenje zahteva, poboljšanje sigurnosti i ubrzanje autentifikacije i zahteva.

Biznis:

- Upravljanje lancom snabdevanja: Primene blokčejna u proizvodnji, distribuciji i logistici su raznovrsne i raznolike. Blokčejn tehnologija može omogućiti praćenje robe u realnom vremenu, dok se kreću, kao i promenu vlasništva, tokom celog procesa lanca snabdevanja. Takođe može podržati sigurno, transparentno, bezbedno deljenje podataka i vidljivost ponude i potražnje.

- Zdravstvo: U pametnom ekosistemu zdravstva, podaci o zdravlju pacijenata, kao što su godine, pol, izveštaji laboratorijskih testova i medicinska istorija, uvek moraju biti

sigurno sačuvani. S obzirom na veliki broj pacijenata u pametnom zdravstvu, ovi podaci ne bi trebalo da se mešaju sa drugim nesenzitivnim podacima i trebalo bi ih čuvati odvojeno.

- Nekretnine: Korišćenjem blokčejn tehnologije, proces prodaje doma može biti ubrzan proverom svih relevantnih detalja. Najvažnije, može smanjiti šanse za prevaru, kao i održava transparentnost u celom procesu kupovine i prodaje.

- Mediji: U industriji komunikacija i medija, blokčejn tehnologija može biti korišćena za održavanje usaglašenosti sa revizijom, upravljanje digitalnim pravima, upravljanje ugovorima, digitalni identitet i praćenje autorskih prava. Ovde, vlasništvo nad video zapisima, muzikom i drugim povezanim sadržajem može biti glatko upravljano korišćenjem blokčejn mreže.

- Energija: Blokčejn tehnologija se istražuje u različitim oblastima upravljanja energijom, poput distribucije električne energije, trgovanja energijom, istraživanja gasa, skladištenja i transformacije. Ovde se može koristiti za izvršavanje transakcija snabdevanja energijom.

Vlada:

- Upravljanje evidencijama: Blokčejn ima veliki potencijal u situacijama gde se podaci pojedinaca mogu zabeležiti u blokčejnu i sve vlade i privatne institucije mogu biti čvorovi koji se odnose na te podatke. Ovo može obezbediti transparentan i siguran sistem.

- Glasanje: Blokčejn tehnologija može olakšati, transparentno i sigurno završavanje procesa glasanja. Ovde, ako napadač ili zlonamerni korisnik želi pristupiti glavnom sistemu ili serveru, ne može manipulisati drugim čvorovima zbog svojstva blokčejna koji onemogućava promene.

- Porezi: Obično, proces podnošenja poreske prijave je podložan mnogim ljudskim greškama. Blokčejn tehnologija može učiniti proces podnošenja poreske prijave efikasnijim, gde se informacije ili podaci čuvaju na privatnoj blokčejn mreži.

- Nevladine organizacije (NGO): Trenutno, broj nevladinih organizacija rapidno raste, gde ljudi širom sveta doniraju dobrotvornim organizacijama. Međutim, u mnogim slučajevima, oni nemaju pojma kako se njihov novac koristi od strane nevladinih organizacija. Ovde, blokčejn tehnologija može biti korišćena za efikasno upravljanje transparentnim sistemom, tako da svi donatori mogu videti kako se njihov novac koristi.

Ostale primene:

- Praćenje: U ovom digitalnom dobu, u mnogim slučajevima je važno znati o proizvodnim detaljima proizvoda. Blokčejn tehnologija može biti korišćena za praćenje originalne proizvodne zemlje proizvoda koristeći njene osobine transfera vlasništva. Danas ne postoji pouzdan sistem koji može pružiti potvrdu u realnom vremenu ili informacije o zemlji porekla proizvoda osim blokčejna.

- Veliki podaci: Veliki podaci se odnose na veliku količinu strukturisanih i nestrukturisanih podataka. Pošto blokčejn tehnologija podržava neizmenjivu

osobinu i svaki računar ili čvor blokčejn mreže proverava podatke koji su na njemu sačuvani, blokčejn tehnologija je izvršno rešenje za skladištenje velikih podataka.

- Industrija poljoprivrede: Mnogi slučajevi upotrebe blokčejna primenjuju se u poljoprivrednoj industriji. To je pretežno za praćenje proizvoda i transfer vlasništva poljoprivrednih proizvoda, poput kafe, palminog ulja, soje i mnogih drugih. Ovo može pomoći u održavanju kvaliteta proizvoda, kao i u održavanju sigurnosti. Istraživači takođe pokušavaju da prate zemljište poljoprivrednika putem blokčejn tehnologije.

4. Izazovi za usvajanje blokchain tehnologije

- Napad na sajber bezbednost: Mnogi istraživači su pokazali da blokčejn tehnologija nije sigurna protiv različitih sajber-napada. Međutim, ovi napadi su uglavnom mogući u dozvoljenim blokčejnima, koji su otvoreni za sve. Dozvoljeni ili konzorcijumski blokčejn je sigurniji jer nema pristupa ovoj mreži izvan odobrenih čvorova ili korisnika.

- Dvostruko trošenje: Dvostruko trošenje se odnosi na korišćenje iste kriptovalute u nekoliko transakcija. Ovde se iskorišćava vremenski interval između početka i validacije dve različite transakcije. U ovom slučaju, napadač ili zlonamerni korisnik dobija rezultat početka ili prve transakcije pre maksimalnog obaveštenja da je druga transakcija nevažeća. Time se stvara veliki problem u blokčejn mreži.

- Izazov modifikacije podataka: Postoje mnogi scenariji u kojima korisnici treba da promene informacije na blokčejn mreži. U blokčejn mreži je nemoguće izvršiti promene što dovodi do račvanja (forking). Postoje dva tipa račvanja, softversko i tvrdo. Softversko račvanje znači kada postoji promena u kodu ili protokolu i podaci prethodnih blokova su kompatibilni s njim. Međutim, većinom, promena koda ili protokola dovodi do tvrdog račvanja. Tvrdo račvanje je promena u protokolu koja nije kompatibilna s prethodnim verzijama, što dovodi do mnogih izazova. Smatra se jednim od ključnih nedostataka blokčejn tehnologije.

- Regulations of general data protection regulation (GDPR): GDPR se zasniva na principu da je lični podatak osobe dat kompaniji ili organizaciji koja je kontrolor podataka. Kontrolor podataka je pravno odgovoran za zaštitu podataka i podleže zakonu o zaštiti podataka Evropske unije. Ovi kontrolori podataka moraju da ispune obaveze GDPR-a. GDPR, koji je usvojila EU, ima određene smernice, koje su teško pratiti ako se koristi blokčejn tehnologija. Kako je već ranije diskutovano, blokčejn je distribuirana i decentralizovana mreža bez centralne vlasti sistema. Međutim, ovo je u sukobu sa GDPR-om, kao i glavni nedostatak korišćenja blokčejn tehnologije u EU.

- Visoka potrošnja energije: U dozvoljenim blokčejn mrežama, rudari rešavaju zagonetku kako bi dodali novi blok u blokčejn mrežu. U većini blokčejn mreža, postoji veliki broj rudara koji su uključeni u računanje, i samo jedan rudar pobeđuje u igri ili zagonetki. Napor drugih rudara postaje beskoristan. Ovo je veoma neefikasno jer se mnogo energije troši tokom računanja. Iz tog razloga, mnogi korisnici ili organizacije ne pokazuju interes za pridruživanje blokčejn mreži. Ako postoji mehanizam koji ne troši mnogo energije, mogao bi biti mnogo efikasniji.

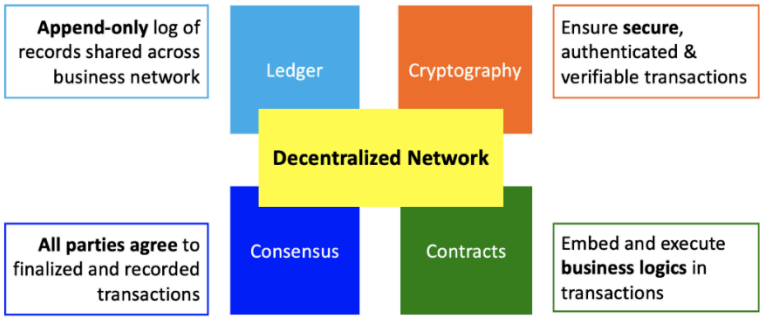
- Kompresija podataka: Čvorovi blokčejn mreže moraju imati mehanizam za kompresiju podataka pre njihovog skladištenja na blokčejn mreži. To je uglavnom zbog brzog povećanja veličine podataka.

- Stopa transakcija: Transakcije po sekundi (TPS) je brzina kojom se podaci u blokovima upisuju u blokčejn mrežu. Bitcoin mreža ima 3-7 TPS, a Ethereum ima 10-20 TPS. Ovi brojevi nisu dovoljni jer se milioni transakcija izvršavaju svakodnevno, što se takođe eksponencijalno povećava. Iz perspektive korisnika ili kompanija, TPS bi uvek trebalo da bude visok, kako bi bilo koja kompanija ili organizacija mogla da podrži ogroman broj transakcija svakodnevno.

- Troškovi: Potrošnja energije, tehnološka složenost i nedostatak blokčejn programera su neki od glavnih faktora koji čine implementiranje blokčejn tehnologije skupljim. Trošak je glavni faktor u usvajanju blokčejn tehnologije u različitim sektorima.

- Usaglašenost i pravna jasnoća: U većini slučajeva, blokčejn tehnologija nije direktno podržana bilo kojim zakonima i naredbama vlade. Moraju postojati specifična pravila i standardi koje definiše vlada za različite entitete poslovanja, tako da vlada može pratiti sve aktivnosti poslovanja koje koriste blokčejn tehnologiju. To sprečava usvajanje od strane poslovanja, gde su transakcije vezane za pravne procese i pokrivene putem pametnih ugovora. Slično tome, podaci blokčejn mreže nedostaju regulacije, posebno ako su povezani sa bilo kakvim ličnim i finansijskim podacima korisnika ili kupaca. Ovo obeshrabruje usvajanje blokčejn tehnologije u različitim vrstama poslovanja ili organizacija.

5. Osnovni elementi blockchain-a



- Decentralized networking: Blockchain se oslanja na decentralizovanu mrežu računara poznatih kao blockchain čvorovi, koji doprinose računarskim resursima za skladištenje i obradu transakcija. Ovi računari funkcionišu autonomno i komuniciraju međusobno na način peer-to-peer (P2P).

- Matematička kriptografija: Kriptografske metode koje se koriste u blockchain-u pružaju matematičke dokaze koji garantuju da blockchain funkcioniše kako je predviđeno. Kriptografski hešovi povezuju blokove podataka u lancu kako bi se sprečila promena podataka nakon upisa. Svaka transakcija je enkriptovana korišćenjem kriptografije javnog ključa kako bi se verifikovao pošiljalac putem digitalnog potpisa i osiguralo da samo namenjeni primalac može primiti transakciju. Poverljivost transakcije se održava pomoću metode Zero Knowledge Proof.
- Transaction ledger: Blockchain služi kao digitalna knjiga (ledger) koja hronološki skladišti transakcije u blokovima koji se dodaju samo na kraju.

- Distribuirani konsenzus: Odluke, poput validnosti transakcije, donose se na osnovu konsenzusa među učestvujućim čvorovima u blockchain mreži, jer nema centralne vlasti. Blockchain mreže koriste protokole konsenzusa kako bi osigurale saglasnost o svakoj transakciji ili bloku koji se dodaje. Proof-of-Work i Proof-of-Stake su popularni mehanizmi konsenzusa, pri čemu prvi daje moć odlučivanja čvorovima sa više računarske snage, a drugi čvorovima sa većim finansijskim udelom.

- Pametni ugovori: Aplikacije na blockchain-u se implementiraju kao "pametni ugovori", računarski programi koji funkcionišu autonomno bez ljudske intervencije. Pametni ugovori izvršavaju uslove poput onih u pravnim ugovorima, osiguravajući automatsko i tačno izvršenje.

6. **<u>Kriptografski elementi blockchain tehnologije</u>**

Jasno je da se neizmenjivost podataka u blockchain-u postiže zahvaljujući informacijama o prethodnom hešu koje povezuju uzastopne blokove u blockchain-u. Međutim, u teoriji, funkcija heša može imati različite ulazne vrednosti koje rezultiraju istim heš izlazom, što znači da se blok b_{j-1} može promeniti tako da njegova originalna vrednost ostane ista kao i pre, odnosno heš vrednost, $H(b_{i-1})$, koja se poklapa sa prethodnim hešom $b_j.prev$, koja je sačuvana u bloku b_j. U ovom slučaju, postupak provere bloka ne može otkriti promenu. Izbor funkcije heša je stoga ključan. Trebalo bi odabrati takvu funkciju koja, iako je teoretski moguće izmeniti blok bez otkrivanja, u praksi je to nemoguće ostvariti. Iz tog razloga, funkcija heša H koja se koristi u blockchain-u mora biti kriptografska funkcija heša, a ne bilo koja proizvoljna funkcija heša.

Podsetimo se da je funkcija heša jednosmerna funkcija koja za ulazni podatak proizvoljne dužine izlazni podatak generiše nisku konstantne dužine, pri čemu se vrednosti predstavljaju kao binarne niske. Na primer, SHA256 je funkcija heša koja generiše binarnu nisku od 256 bitova. Kriptografska funkcija heša H je funkcija heša sa tri osobine:

- Otporna na sudar: Nemoguće je naći različite ulazne poruke x i y tako da $H(x) = H(y)$.
- Sakrivanje: Imajući izlaz $c = H(x)$, nemoguće je pronaći ulaz x.
- Prijateljstvo sa slagalicom: Ako znamo heš vrednost $c = H(r \| x)$ ulazne poruke napravljene od spajanja r i x, i čak i ako znamo deo ulaza, x, ne možemo rekonstruisati preostali ulaz r u vremenskoj složenosti brže od 2^n gde je n binarna dužina izlaza c.

Zbog ovih svojstava, znajući $b_j.prev = H(b_{j-1})$, nemoguće je naći $b'_{j-1} \neq b_{j-1}$ tkd. $H(b_{j-1}) = H(b_{j-1})$. Sa H kao kriptografskom funkcijom heša, niko ne može promeniti postojeci blok bez detektovanja. Podaci blockchain-a su neotklonjivi.
Kriptografske metode takođe imaju mnoge druge primene u radu blockchain-a. Podsetimo se opklade između Alise i Boba na početku ovog poglavlja, u kojoj se postavlja pitanje šta ako Bob vara. Kriptografska funkcija heša H može rešiti ovaj problem varanja na sledeći način:

1) Alisa: pretpostavi da je njen predikat x ("glava" ili "rep")
    - Generiši tajni slučajni broj r (velike binarne dužine n).
    - Izračunaj $c = H(r \| x)$ (nazvan "predikcioni angažman").
    - Pošalji c Bobu, umesto slanja njenog predikata kao sirovih podataka.
2) Boban: po prijemu predikcionog angažmana c, poslaće Alisi iskreni ishod $x*$ bacanja novčića. Budući da je funkcija heša H kriptografska, on ne zna pravo predviđanje Alice, i kao takav, nema razloga za prevaru.
3) Alisa: po prijemu $x*$, ako je njena pretpostavka tačna, odnosno $x = x*$ ona će reći Bobanu da je pobedila tako što će mu poslati tajni broj r.
4) Boban: po prijemu broja r, proveriće da li se angažman c koji je ranije dobio od Alice podudara sa $H(r \| x*)$ i uverljivo prihvatiti gubitak.

Ovo rešenje naziva se šema angažmana u kriptografiji. Kritično je da tajni r generisan od strane Alise mora poticati iz velikog broja prostora. Ako je binarna dužina n mala, Bobanu će trebati kratko vreme da iscrpno pokuša sve moguće vrednosti r i kombinuje sa x="glava" ili x="rep" da bi video koja kombinacija zadovoljava $H(r \| x) = c$. Kada se ta kombinacija pronađe, može prevariti Alisu govoreći joj da je ishod suprotna vrednost x pronađena u toj kombinaciji. Kada je n velik, iako x može imati samo dve moguće vrednosti, "glava" ili "rep", Boban ne može rekonstruisati tajni r zahvaljujući "prijateljskom sa slagalicom" svojstvu H kao kriptografske funkcije heša.

7. Problem distribuiranog konsensenzusa?

Problemi distribuiranog konsenzusa mogu se uočiti u različitim vrstama algoritama koji se koriste za postizanje konsenzusa u blockchain mrežama. Evo nekoliko problema koji su prisutni u nekim od popularnih algoritama:

- Proof of Work (PoW): Ovaj algoritam zahteva veliku količinu računarske snage i vremena kako bi se rešio kriptografski problem. Napadi poput sebičnog rudarenja mogu ometati mrežu, ali ne utiču direktno na neizmenjivost blockchain-a.

- Proof of Stake (PoS): Iako PoS zahteva manje računarske snage od PoW, postoji problem "ništa na kocki" koji podrazumeva da čvorovi s najvećim ulogom uvek dobijaju šansu da dodaju novi blok, što može dovesti do centralizacije.

- Delegated Proof of Stake (DPoS): Iako ovaj algoritam omogućava korisnicima da glasaju za čvorove, to može dovesti do centralizacije ako samo mali broj čvorova dobija većinu glasova.

- Practical Byzantine Fault Tolerance (PBFT): PBFT je dobar za privatne blockchain mreže, ali ima ograničenu skalabilnost i nije pogodan za javne blockchain mreže.

- Stellar Consensus Protocol (SCP): Iako je decentralizovan, SCP ima nisku latenciju, ali zahteva kompleksan proces glasanja i konsenzusa.

- Raft: Iako je jednostavan za razumevanje i implementaciju, Raft je zavistan od lidera čvora, pa ako lider postane zlonameran, može ugroziti celokupni sistem.

8. Koncenzus zasnovan na dokazu rada (proof of work).

U PoW-u, svi čvorovi u blockchain mreži pokušavaju da reše kriptografsku heš funkciju. Ovde se koristi
SHA-256, koja generiše heš vrednost fiksne veličine od 256 bita. Da bi dodali čvor ili blok korišćenjem PoW-a, rudari (miners) pronalaze određeni broj kako bi rešili kriptografski problem. Ovaj proces je vremenski zahtevan i matematički težak, jer rudari koriste metodu grubih sila kako bi pronašli broj koji rešava problem. Ciljni broj podržava težinu mreže. U Bitcoinu, ciljni broj je obeležen na takav način da proces rudarenja može da se odvija svakih 10 minuta. Napadači ili zlonamerni korisnici mogu da utiču na PoW baziranu blockchain mrežu ako preuzmu kontrolu nad 25% računarske snage pomoću sebičnih rudarskih napada. Međutim, ovaj napad ne utiče na neizmenjivost blockchain mreže. PoW je bio veoma popularan u svetu kriptovaluta tokom mnogo godina.

Proof-of-work mining:

$$SHA256(SHA256(block\_header)) < (65535 << 208)/difficulty. \quad (1)$$
ID TARGET

$block\_header$: Ovo su podaci bloka koje treba da heširamo kako bismo stvorili validan heš
$difficulty$: Što je veća težina, to je teže pronaći validan heš

Nalazak zadovoljavajućeg identifikatora bloka nije lak zbog dvostrukog SHA256 heširanja. Nijedan algoritam nije bolji od brute force-a, koji zahteva O(2^{32}). U nejednačini (1), ako povećamo difficulty, vrednost TARGET će se smanjiti, čineći teže zadovoljenje nejednačine ID < TARGET
Ako se blokovi kreiraju suviše lako, pošto se svaki novo kreirani blok emituje u mrežu, trošak komunikacije bi bio veoma visok. Štaviše, kako rudari autonomno dodaju blokove, mnogi blokovi istovremeno kreirani od strane različitih čvorova će biti dodati na isti poslednji blok postojećeg blockchain-a (njihova lokalna kopija). To ne samo što izaziva ozbiljnu nesaglasnost i ranjivost na dvostruko trošenje, već i gubi napore većine rudara zbog činjenice da samo jedan blok može biti dodat sledeći globalnom blockchain-u.
Izazov pronalaženja dobrog ID-a koji zadovoljava nejednačinu (1) naziva se PoW problemom. PoW protokol takođe pomaže da se Bitcoin, kao valuta, zaštiti od inflacije. Njegovo sporo kovanje i ograničena ponuda stvara ograničenu cirkulaciju, čime se cena čini vrednom.

9. Koncenzus zasnovan na dokazu ulaganja (proof of stake).

Ovaj algoritam konsenzusa je generalizovana forma PoW-a. U PoS-u, čvorovi se nazivaju validatorima. Validatori validiraju izvršenje kako bi mogli zaraditi naknadu za transakciju. U ovom algoritmu konsenzusa, nema konkurencije između validatora za rešavanje računarskog problema. Čvor se bira korišćenjem lutrije za rudarenje novog bloka na osnovu količine učesnika. Izabrani čvor koristi tehniku digitalnog potpisa za dokazivanje vlasništva nad ulogom. Stoga, nije potrebna velika računarska snaga u PoS-u. Međutim, postoji novi problem jer čvor sa najvećom količinom uloga uvek dobija priliku da doda novi blok. Na taj način, indirektno se ponovo postaje centralizovana mreža. Štaviše, u PoS-u, ako izabrani čvor deluje kritično, nema ništa za izgubiti. Ovaj problem je poznat i kao "ništa za ulog".

10. Stanje blockchain-a: model zasnovan na transkacijama i na računima

Zavisno od toga kako je blokčejn dizajniran, struktura podataka stanja može biti različita. Može biti zasnovana na transakcijama (informacije o stanju sastoje se od liste transakcija) ili na računima (informacije o stanju sastoje se od stanja računa)
U *modelu zasnovanom na transakcijama*, poznatom kao neočekivani izlaz transakcije (UTXO) koji je predložio Bitcoin, svaka transakcija može poslati vrednost jednom ili više primalaca. Ovo se sastoji od sledećeg:

- Polje Izlaza: Lista primaoca adresa i iznos sredstava koji se šalje svakom od njih. Svaki izlazni transfer naziva se transakcijom UTXO.
- Polje Ulaza: Lista UTXO transakcija koje pružaju sredstva za ovu transakciju. Ove UTXO transakcije su prethodno poslale sredstva pošiljaocu i trenutno nisu potrošene.

Model zasnovan na računima je intuitivniji. Sličan je modelu računa u banci. Stanje se sastoji od informacija o stanju za svaku adresu. Kada se desi transakcija, stanje se odmah ažurira i čuva informacije o saldu na računima pošiljaoca i primaoca. Stoga, saldo na računu adrese

može se odmah dobiti bez ikakvog računanja.
Transakcija u modelu zasnovanom na računima je mnogo jednostavnija od UTXO transakcije. Prva uključuje samo jednu adresu primaoca i iznos sredstava koji se šalje. Mnogo je brže provjeriti da li pošiljalac ima dovoljno sredstava, što se postiže jednostavnim uporedjivanjem dva broja: da li saldo pošiljaoca premašuje iznos koji se šalje. Model zasnovan na računima nudi jasnu prednost kada je reč o omogućavanju "pametnih ugovora" (računarskih programa za implementaciju aplikacija na blokčejnu). Za pametne ugovore, transakcija može biti ne samo transfer vrednosti, već i poziv funkciji proizvoljne logike; sadrži kod podataka za izvršenje te funkcije. Procesiranje transakcije stoga uključuje izvršenje koda u transakciji. Kako su pametni ugovori računarski skupi, važnost jednostavnosti računanja je bitna. UTXO stvara računarski overhead jer sve transakcije trošenja moraju biti eksplicitno zabeležene.

11. Struktura blockchain lanca

Knjiga transakcija blokčejna prati strukturu lanca. Podaci su organizovani u lanac podataka: $b1$, $b2$, $b3$, …
Kada je potrebno sačuvati nove transakcije, one se stavljaju u novi blok koji će biti dodat na poslednji blok postojećeg lanca. U blokčejnu zasnovanom na računima, na primer, Ethereumu, blok takođe sadrži informacije o stanju blokčejna (saldi svih računa u trenutnom trenutku).
Osim čuvanja podataka o transakcijama, stanju blokčejna ako je primenljivo, i potrebnih informacija o zaglavlju, blok ima dva važna atributa:

- ID bloka $b_i.id$: Postavljen je na heš vrednost sadržaja bloka koristeći kriptografsku heš funkciju H; tj. $b_i.id = H(b_i)$. Ova heš funkcija je unapred definisana i javno poznata.
- Prethodni heš $b_i.prev$: Postavljen je na ID prethodnog bloka b_{i-1} na koji se dodaje blok bi; tj. $b_i.prev = b_{i-1}.id$.

Napomenuto je da ID bloka ne mora nužno biti smešten u bloku jer se može izračunati iz sadržaja bloka.
Informacija o prethodnom hešu je ključna za održavanje integriteta podataka lanca. Ako se bilo koji deo bilo kog bloka promeni nakon što je zabeležen u blokčejnu, to će biti otkriveno. To je zato što da bi se novi blok dodao u blokčejn, mora proći postupak koji se zove validacija bloka. Novi blok b_{i+1} je validan samo ako:

1) Prethodni heš je konsistentan: $b_{i+1}.prev = H(b_i)$
2) Sve transakcije u b_{i+1} su validne
3) Prethodni blok b_i je validan

Kao rezultat toga, postupak validacije za blok b_{i+1} zahteva proveru da li vrednost prethodnog heša smeštenog u bloku b_j odgovara heš vrednosti njenog prethodnog bloka b_{j-1} za sve $j \leq i+1$. Ako je raniji blok, recimo b_{j-1}, promenjen od svoje originalne vrednosti, kada izračunamo njegovu heš vrednost, $H(b_{j-1})$, naći ćemo da nije identičan sa prethodnim hešom $b_j.prev$ smeštenim u bloku b_j. To je kršenje i kao rezultat novi blok b_{i+1} se zaključuje kao nevažeći i ne dodaje se blokčejnu.
Posledica promene bloka b_j je da blokčejn nikada neće rasti izvan vremena te promene

12. Procesiranje transakcije

Kada neko pokrene transakciju sa blokčejnom, to se obično radi putem korisnički prijateljske aplikacije koja može da interaguje sa mrežom blokčejna putem API poziva. Ova transakcija

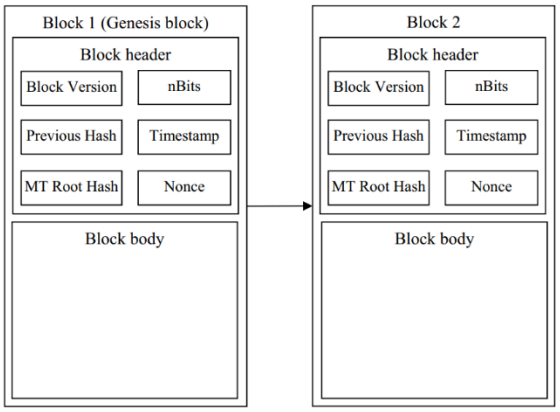
treba da bude poslata na čvor blokčejna (u praksi, na više čvorova u slučaju da jedan čvor može da zakaže ili da se ponaša neispravno) i biće obrađena na sledeći način:

- Svaki čvor X prilikom prvog prijema transakcije Tx:
  - Prosleđivanje transakcije: prosledi transakciju Tx susednim čvorovima X.
  - Verifikacija transakcije: proveri da li pošiljalac transakcije Tx ima dovoljno sredstava za slanje. Ako je tako, Tx se stavlja u mempool, koji je red čekanja validnih transakcija koje treba da budu stavljene u novi blok.
  - Kreiranje blokčejna: izvući transakcije koje čekaju iz mempoola da bi se uključile u novi blok b i dodati ovaj blok u postojeću knjigu transakcija blokčejna na čvoru X. Napomena: blok b mora uključivati informacije o prethodnom hešu.
  - Ažuriranje bloka: poslati novi blok b susednim čvorovima X.

- Svaki čvor Y prilikom prvog prijema bloka b:
  - Prosleđivanje bloka: prosledi blok b susednim čvorovima Y.
  - Validacija bloka: proveri validnost bloka b u postojećoj knjizi transakcija blokčejna čvora Y. Ova validacija zahteva proveru konsistentnosti informacija o prethodnom hešu i validnosti svake transakcije u bloku b.
  - Dodavanje bloka: dodaj blok b u knjigu transakcija blokčejna ako je validan. U suprotnom, ignorisati b.

Postupak obrade transakcije u blokčejnu je jednostavan i omogućava autonomnu obradu na čvorovima blokčejna. Međutim, ova jednostavnost dovodi do nekoliko problema sa konsistentnošću. Prvo, svaka transakcija je emitovana svim čvorovima i ista transakcija može biti dodata u različite blokove kreirane na različitim čvorovima. Moramo da osiguramo da svaka transakcija može biti dodata u blokčejn samo jednom. Drugo, različiti čvorovi paralelno kreiraju različite nove blokove da bi pokušali da ih dodaju na (isti) postojeći blokčejn. Moramo da osiguramo da samo jedan od njih bude dodat kao sledeći blok. Treće, različiti čvorovi mogu imati različite kopije blokčejna. Moramo da osiguramo da se slože oko jedne kopije kao globalno ispravne verzije. Da bismo rešili ove nekonzistentnosti, čvorovi se moraju redovno složiti oko trenutnog stanja blokčejna, i to je ono što nazivamo postizanje konsenzusa. Potreban nam je protokol konsenzusa.

13. Sadržaj transakcije i bloka



Blok se sastoji iz dva dela: (i) blok zaglavlja i (ii) blok tela. Kao što je već pomenuto, svaki blok je povezan sa prethodnim blokom ili roditeljskim blokom i formira lanac. Prvi blok, tj. genski blok, nema prethodni blok. Ovde se heš svakog bloka beleži u zaglavlju odgovarajućeg bloka. Blok zaglavlja sadrži sve informacije o bloku. Postoji šest atributa blok zaglavlja: (i) verzija bloka, (ii) heš korena Merkle stabla (MT), (iii) Nonce, (iv) nBits, (v) heš prethodnog bloka i (vi) vremenska oznaka.

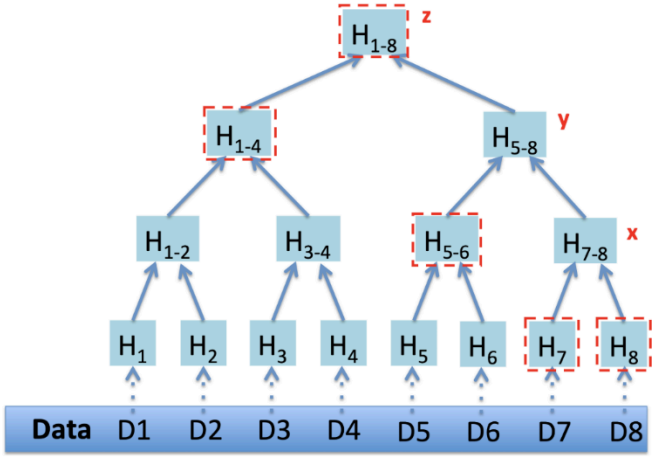
Verzija bloka sastoji se od nekoliko pravila autentifikacije mreže blokčejna koja moraju biti poštovana od strane svih čvorova.
Heš prethodnog bloka: Odnosi se na heš vrednost od 256 bita roditeljskog ili prethodnog bloka.
Heš korena Merkle stabla (MT): MT je struktura podataka koja se koristi za čuvanje jedne heš vrednosti za sve transakcije umesto čuvanja različitih heš vrednosti za različite transakcije. Takođe je poznat kao binarno heš stablo. MT kombinuje heš vrednosti transakcija u parovima, a heš vrednosti se kombinuju za sve transakcije i dobija se koren nazvan MT koren heš.
nBits: Označava trenutnu težinu, koja se koristi za kreiranje bloka.
Vremenska oznaka: Vremenska oznaka se odnosi na Unix vreme koje se koristi za beleženje vremena kada rudar započinje proces rudarenja. Struktura bloka prikazana je na slici 2.
Nonce: Nonce je zapravo slučajan broj od 32 bita. Može se koristiti samo jednom i prilagođava ga rudarima tokom procesa rudarenja. U Bitcoinu, rudari pokušavaju da pogode validan nonce kako bi izračunali heš bloka, koji mora zadovoljiti određene zahteve. Prvi rudar koji pronađe validan nonce koji rezultira heš bloka ima pravo ili moć da doda novi blok u mrežu blokčejna. Rudar takođe biva nagrađen za dodavanje bloka.

Telo bloka se uglavnom sastoji iz dva dela: (i) brojača transakcija i (ii) transakcija.
Brojač transakcija: Brojač transakcija čuva broj transakcija u bloku.
Transakcija: Transakcija se može nazvati komunikacijom između pošiljaoca i primaoca radi razmene imovine. U mreži blokčejna može biti više od jedne transakcije u bloku. Veličina bloka i transakcije određuju broj transakcija koje mogu biti prikazane u određenom bloku.

Postoje neki atributi transakcije:

- Iznos: Iznos je digitalna vrednost koju pošiljalac treba da prenese.
- Ulaz: Ulaz je osnovno detaljanje digitalne imovine koja želi da se prenese. Ovde, digitalna imovina treba da bude apsolutno prepoznatljiva i da uključuje vrednosti koje se razlikuju od drugih imovina.
- Izlaz: Izlaz čuva sve detalje računa primaoca. To uključuje vrednost digitalne imovine i identitet (ID) primaoca. Pored toga, izlaz sadrži neka pravila koja primaoc ne sme da prekrši kako bi primio povezanu vrednost.
- Heš transakcije ili ID: Svaka transakcija ima poseban heš ili ID transakcije koji podržava digitalni potpis na osnovu kriptografije sa javnim ključem.

14. Merkle tree



Transakcije su organizovane u bloku kao Merkle Stablo, binarno stablo gde svaki unutrašnji čvor čuva heš vrednost vrednosti dece. Na slici 8 je prikazano takvo stablo za Bitcoin, gde postoje osam transakcija $\{D1, D2, ..., D8\}$, svaka smeštena u listni čvor, unutrašnji čvor $H_{1-4} = H(H_{1-2} \parallel H_{3-4})$, unutrašnji čvor $H_{1-2} = H(H_1 \parallel H_2)$, unutrašnji čvor $H_1 = H(D1)$, $H_2 = H(D2)$, itd. Bitcoin koristi SHA2 za heš funkciju H.

Vrednost na korenu stabla, na primer, H_{1-8}, je heš korena Merkle stabla koji je smešten u zaglavlju Bitcoin bloka. Postoje dva ključna svojstva. Prvo, svaka promena u podacima transakcije uzrokuje promenu heša korena Merkle stabla. Kao takvo, ako se blok promeni, bilo da je reč o podacima transakcije ili ne-transakcionom delu, heš bloka će se promeniti i biti otkriven. Drugo, brzo je proveriti postojanje transakcije u bloku. Na primer, da bi dokazali da je transakcija $D7$ u bloku, proverilac treba da pruži četiri vrednosti kao dokaz: H_7, H_{1-4}, H_{5-6}, H_8.
Verifikator će izračunati sledeće:
$x=H(H_7 \| H_8)$
$y=H(x \| H_{5-6})=H(H(H_7 \| H_8) \| H_{5-6})$
$z=H(y \| H_{1-4})=H(H(H(H_7 \| H_8) \| H_{5-6}) \| H_{1-4})$
i uporediti z sa H_{1-8}. Njihova jednakost znači da je transakcija $D7$ u bloku.

Za Merkle stablo sa n transakcija, za ovu verifikaciju potrebno je O (log n) vreme. Trebalo bi O (n) vremena ako bismo naivno čuvali transakcije u strukturi sličnoj listi.
Čvor u Bitcoinu može biti ne-rudarski čvor. Tu je samo da vrši transakcije sa mrežom, ne zainteresovan za stvaranje blokova kako bi primio nagradu bloka. Budući da blok zaglavlje pruža dovoljno informacija za verifikaciju i vršenje transakcija, čvoru je potrebno samo blok zaglavlje, ne i pun sadržaj bloka. Pošto se stvarne transakcije ne čuvaju, zahtev za skladištenjem za takav čvor je skroman (samo 80 bajtova, koji se lako mogu smestiti u memoriju). Takođe je nizak i trošak komunikacije za povlačenje kopija lanca blokova od suseda (samo povlačenje blok zaglavlja).

15. Pametni ugovori

Mnoge aplikacije u stvarnom svetu mogu imati koristi od tehnologije blockchain-a, a imati posebnu blockchain mrežu za svaku pojedinačnu aplikaciju nije realno. Ovo je motivacija za Ethereum, da postoji univerzalni blockchain računar koji može pokretati aplikacije proizvoljnih namena. Da bi razvili takve aplikacije, programeri pišu računarske programe nazvane "pametni ugovori". Ethereum, stoga, važi za blockchain pametnih ugovora. Druge javne mreže blockchain-a pametnih ugovora uključuju Algorand, Tezos i Solanu.
Pametni ugovori se pišu koristeći programski jezik visokog nivoa (npr. Solidity, Viper, Flint, Bamboo). Solidity je najpopularniji jezik za mreže smart ugovora. On je Turing-kompletan, što znači da može simulirati bilo koju računsku operaciju. Za razliku od toga, Skript, programski jezik Bitcoina, nije Turing-kompletan. Zato je Skript vrlo lagan i pogodan za Bitcoin. Bitcoinu nije potreban univerzalni jezik jer je digitalna valuta jedini cilj Bitcoina.
Izvorni kodovi razvijenih pametnih ugovora su vidljivi javnosti. Stoga nema ničega što treba sakriti u radu smart ugovora i ljudi mogu biti sigurni da će raditi onako kako je programirano. S druge strane, u određenim slučajevima, složen smart ugovor može sadržati greške i druge sigurnosne rupe koje nisu lako uočljive; popravka je glavobolja nakon što je aplikacija već razvijena sa mnogo korisnika; imajte na umu da je blockchain nepromenjiv. Stoga, profesionalni projekti treba da pažljivo razmotre bezbednosne aspekte pre puštanja smart ugovora u rad.

16. Skalabilnost blockchain-a, rolapovi, L1/L2

Blockchain ima tri cilja: decentralizaciju, sigurnost i skalabilnost. Međutim, prema osnivaču Ethereuma, ovi ciljevi ne mogu biti savršeno ostvareni, što je nazvao "Trilemom skalabilnosti" za blockchain:

- *Decentralizacija:* Blockchain pruža računarstvo bez poverenja i ne oslanja se na centralnu tačku kontrole. Potrebno je da bude decentralizovan tako da čvorovi autonomno i jednako učestvuju.
- *Sigurnost:* Blockchain mora da funkcioniše kako se očekuje, otporan je na kvarove i napade. Kao što smo diskutovali u prethodnom odeljku, blockchain je sistem tolerantan na bizantske greške. Što više grešaka velikog praga čvorova može podneti, to bolju sigurnost pruža.
- *Skalabilnost:* Namerno je dizajniran kao "svetski" računar za sve ljude da pokreću sve aplikacije. Blockchain bi trebao da se skalira sa sve većim brojem transakcija, kako u smislu skladišta, tako i u smislu zahteva za računanje.

Cilj decentralizacije zahteva što više čvorova za validaciju blokova. Međutim, više validatora dovodi do veće težine održavanja konsenzusa, a time i sigurnosti. Kako se decentralizacija i sigurnost postižu samo sa malim blockchain mrežama (veličina mreže ili obim transakcija), cilj skalabilnosti nije ispunjen.

Blockchainovi često moraju da naprave kompromise u ovom trilemu. Na primer, Bitcoin nudi odličnu decentralizaciju i sigurnost, ali ne privlačnu skalabilnost. Solana, s druge strane, žrtvuje decentralizaciju za skalabilnost. Dok Ethereum nudi odličnu decentralizaciju, brzinu i sigurnost, njegova skalabilnost ima ograničenja.

Promena osnovnog dizajna, bilo da je reč o mehanizmu konsenzusa, strukturi bloka, ili kriptografskim metodama, na sloju 1 blockchain mreže ima svoje kompromise zbog trostruke skalarne dileme. Godine 2016, je predstavljen pristup skalabilnosti na sloju 2 koji se primenjuje na Ethereum, i, teorijski, na bilo kojem blockchainu na sloju 1. Predloženo rešenje, nazvano Plasma, gradi mrežu blockchaina visokog protoka sa sidrom na vrhu sloja 1 blockchain-a na sledeći način:
1) obrada transakcija na sloju 2: korisnici vrše transakcije na sloju 2 blockchain-a, stoga veoma brzo;
2) konačnost transakcija na sloju 1: informacije o stanju završenih transakcija se čuvaju u sloju 1 blockchain-a, stoga garantujući sigurnost protiv nepoštenih transakcija.
Na primer, Polygon je sloj 2 blockchain pametnih ugovora na vrhu Ethereum mreže kao sloj 1.

17. ERC20 tokeni, ERC721 tokeni

Bitcoin (BTC) je jedina osnovna token (digitalna valuta) Bitcoin blockchain mreže. Njeni korisnici međusobno vrše transakcije (plaćajući jedni drugima) koristeći BTC. Pametna ugovorna mreža takođe ima osnovni token (npr. ETH za Ethereum), čija je osnovna svrha omogućavanje korisnicima da implementiraju pametne ugovore i interaguju s njima. Da bi se implementirao pametni ugovor na Ethereumu, mora se platiti određeni iznos u ETH. Ova suma zavisi od računarske složenosti ugovora. Pored ETH-a, može se kreirati mnogo sekundarnih tokena koji služe različitim aplikacijama. Na primer, može se izgraditi aplikacija lojalnosti na vrhu Ethereuma i implementirati bodove lojalnosti kao token, ili vlada neke zemlje može izdati digitalnu valutu centralne banke (CBDC) kao token na vrhu Ethereuma.

Token je implementiran u obliku jednostavnog pametnog ugovora. Ako je token namenjen kao vrsta digitalne valute, ovaj ugovor čuva informacije o stanju tokena za svaki račun (adresu blockchaina) i uključuje osnovne funkcije koje omogućavaju pošiljaocu da prenese tokene primaocu (potrebno za transakciju stvarnog sveta), da potrošač prenese tokene u ime svog vlasnika primaocu (korisno za trgovinsku razmenu ili banku da prebaci novac sa računa

nekoga na primaoca, naravno samo uz dozvolu), ili u mnogim slučajevima, da izda nove i uništi postojeće tokene (korisno za vladu da se nosi s krizama inflacije).

Radi lakše kreacije tokena, definisani su neki standardi tokena i kreirani su šabloni pametnih ugovora. Prvi standardi su definisani za Ethereum mrežu, a njihovi pandani kasnije su pratili za druge mreže pametnih ugovora. Za Ethereum, ERC-20 je standard za zamjenjive tokene, ERC-721 za nezamjenjive tokene (NFT), a ERC-1155 za generičke višetokene (onaj koji može predstavljati zamjenjive tokene ili nezamjenjive tokene ili njihovu višestruku količinu). Na primer, ERC-20 se koristi za implementaciju kriptovalute, ERC-721 za digitalno predstavljanje fizičke imovine jedinstveno i neponovljivo kao NFT, a ERC-1155 za digitalno predstavljanje akcija jednog preduzeća. Osnovna implementacija Solidity pametnog ugovora ERC-20 prikazana je na slici 12.