

Kriptovaluták elméleti hátttere

Ladóczki Bence

docens

Távközlési és Mesterséges Intelligencia Tanszék (TMIT)

Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar (VIK)

ladoczki.bence@vik.bme.hu

2026. október 15.

1 RANDAO-manipuláció: selfish mixing és forking támadás

A fent leírt LMD-GHOST fork-choice szabály nemcsak a konszenzus alapja, hanem egyúttal a támadási felület is: a javaslok kiválasztását vezérlő véletlenség-forrás, a *RANDAO* manipulálható azáltal, hogy a támadó befolyásolja, mely blokkok kerülnek a kanonikus láncba. Ez a proof-of-work selfish miningjének proof-of-stake megfelelője. Kutatócsoportunk ezzel a kérdéssel foglalkozott, és az eredményeinkből a 2025-ös ACM CCS konferencián jelent meg publikáció [1]; ez a szakasz annak a legfontosabb gondolatait foglalja össze.

Az Ethereum proof-of-stake protokolljában a blokk-javasló validátorokat egy elosztott véletlenség-forrás (distributed randomness beacon, DRB), a *RANDAO* jelöli ki. Minden javasló a blokkjába tesz egy `randao_reveal` értéket, amely az epoch-szám BLS-aláírása a saját titkos kulcsával; mivel a BLS-aláírás determinisztikus, ezt az értéket a javasló nem tudja grindinggel (próbálgatással) manipulálni. A beacon kimenete az összes eddigi hozzájárulás hash-ének bitenkénti XOR-ja, és ez a kimenet határozza meg a két epochhal későbbi javasloi listát. A legfontosabb sebezhetőség abból fakad, hogy a *kihagyott* (missed) blokkok hozzájárulása nullának számít, így aki kihagy egy blokkot, az megváltoztatja a *RANDAO* kimenetét. Ezt kihasználva egy nagy stake-kel rendelkező validátor (pl. egy staking pool) szándékosan kihagyhat blokkokat, ha a módosult *RANDAO*-kimenet a jövőben több slotot oszt neki – ezt nevezzük *selfish mixingnek*. A korábbi szakirodalom (Wahrstätter, illetve Alpturer–Weinberg) megmutatta, hogy egy 20% stake-kel rendelkező szereplő selfish mixinggel kb. 0,7% extra slotot szerezhet.

Mi egy ennél jóval erősebb új stratégiát vezettünk be, a *forking támadást*, ahol a támadó nem csak a saját blokkját hagyja ki, hanem egy becsületes javasló már közzétett blokkját *forkolja ki* a kanonikus láncból. A támadás akkor lehetséges, ha a támadó az epoch-határ körül megfelelő slot-sorrenddel rendelkezik (jelölésünkben **A****H****A**), és a stake-aránya legalább 20%. Az 1. ábra egy ilyen támadást illusztrál: a támadó titokban épít egy privát ágat, hagyja, hogy a becsületes javasló a 31. slotban publikáljon, majd a

következő epoch első slotjában felfedi a hosszabb privát ágát. Ez azért működik, mert az LMD-GHOST fork-choice szabály szerint a támadó ága $2\alpha + p_{\text{boost}}$ súlyt kap (ahol $p_{\text{boost}} = 0,4$ a proposer boost), míg a becsületes ág csak $1 - \alpha$ -t, így $\alpha \geq 0,2$ esetén a támadó ága nyer. A forking így kettős kárt okoz: egyrészt tovább torzítja a RANDAO-t (akár megduplázva a selfish mixinggel elérhető nyereséget), másrészt közvetlenül aláássa a láncba vetett bizalmat, mert már publikált, a felhasználók által véglegesnek hitt blokkok esnek ki. Ráadásul a támadó a kiforkolt blokk MEV-jét (maximal extractable value) is megszerezheti, és a kiesett slotjait a protokoll később teljesen kompenzálja, így a támadás szinte veszteségmentes.

A támadó optimális stratégiájának kiszámításához – a selfish mixing korábbi modelljét kiterjesztve – egy Markov-döntési folyamatot (MDP) építettünk, amely a kétepochnyi időre kiterjedő állapottér miatt rendkívül nagy irányított gráfot ad. Az MDP célfüggvénye a Γ_π hosszú távú várható nettó blokknyereség (a megszerzett blokkok mínusz a feláldozott blokkok), amelyet a támadó a π politikájával maximalizál, és minden döntési állapotban a legnagyobb várható hasznú cselekvést választja: vagy nem forkol, vagy valamelyik forking-cselekvést indítja. A 3. ábra a fő eredményünk: a forkolást és selfish mixinget együtt alkalmazó támadó szigorúan többet szerez, mint a csak selfish mixinget használó korábbi munka (AW24), és közben nő a kiforkolt becsületes blokkok száma (romló láncminőség). A 4. ábra a stacionárius eloszlásban mutatja, milyen gyakran választ egy racionális szereplő forkolást, megbánást (regret), selfish mixinget vagy becsületes viselkedést a stake-arány függvényében; jól látszik, hogy a nagy szereplők gyakran manipulálnak: egy $\alpha = 0,3$ stake-ű, gazdaságilag racionális validátor az epochok 38,24%-ában manipulálja a RANDAO-t.

Az Ethereum mainnet valós adatain is kerestük e támadások nyomait: egyértelmű, statisztikailag szignifikáns többepochos manipulációt egyelőre nem találtunk, de mivel a protokoll kriptográfiailag nem véd ellene, racionális szereplők mellett a jövőben várható a megjelenésük. Lehetséges ellenintézkedések a hosszabb epochok, illetve a RANDAO lecserélése egy nem-torzítható DRB-re (pl. VDF-alapú vagy súlyozott threshold VRF konstrukcióra).

A legegyszerűbb eset azt mutatja, hogyan dönt a támadó, ha az epoch két utolsó (tail) slotja az övé (a **A²** sorrend). Ekkor a két blokk közzétételéről vagy kihagyásáról négyféleképpen határozhat, mind a négy döntés más RANDAO-kimenetet (R^e) ad, és a támadó azt választja, amelyik hosszú távon a legtöbb blokkot hozza neki – ezt a döntési fát szemlélteti a 2. ábra.

A RANDAO beacon kimenetét az e epochban formálisan a következő rekurzió adja:

$$R^e := R^{e-1} \oplus \left(\bigoplus_{i=0}^{31} h(r_i^e) \right), \quad R^0 := 0^{256}, \quad (1)$$

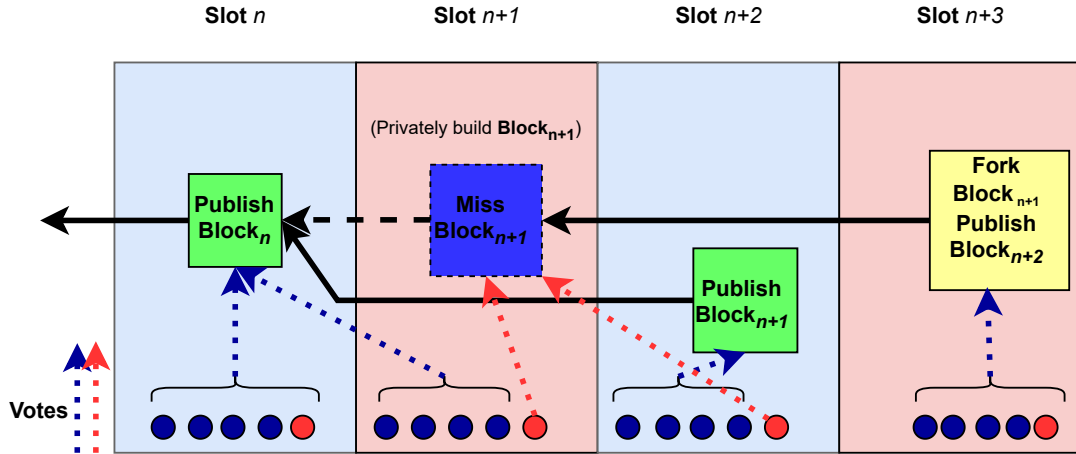
ahol $h(r_i^e)$ az i . slot `randao_reveal` értékének hash-e, és egy kihagyott blokk hozzájárulása $h(r_i^e) := 0^{256}$ – épp ez teszi a kimenetet a kihagyásokkal befolyásolhatóvá.

A forking támadás működésének kulcsa az LMD-GHOST súlyozás. A támadó privát ága és a becsületes ág súlya:

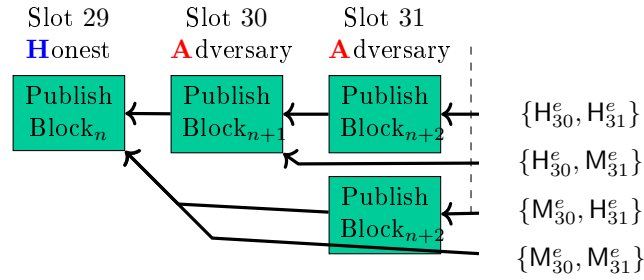
$$w_{\mathcal{A}} = 2\alpha + p_{\text{boost}}, \quad w_{\mathcal{H}} = 1 - \alpha, \quad (2)$$

és a támadó ága akkor nyer (azaz a becsületes blokk kiforkolható), ha $w_{\mathcal{A}} > w_{\mathcal{H}}$, ami $p_{\text{boost}} = 0,4$ mellett épp az $\alpha \geq 0,2$ feltételt adja.

A támadó a következő hosszú távú várható nettó blokknyereséget maximalizálja a π



1. ábra: A **AH.A** forking támadás. A támadó ($\mathcal{A}$) a 30. slotban titokban épít és privát ágán szavaz, hagyja a becsületes 31. slotot publikálni, majd a következő epoch 0. slotjában felfedi a kétblokkos privát ágát, kiforkolva a becsületes blokkot. A piros/kék pontok a támadó, illetve becsületes szavazatokat jelölik.



2. ábra: A selfish mixing döntési fája a **A²** támadási sorozatra. A támadó mind a négy esetre kiszámítja az R^e RANDAO-kimenetet, és a legkedvezőbbet (a legtöbb jövőbeli blokkot adót) választja. (H_i^e : az i . slot blokkját publikálja, M_i^e : kihagyja.)

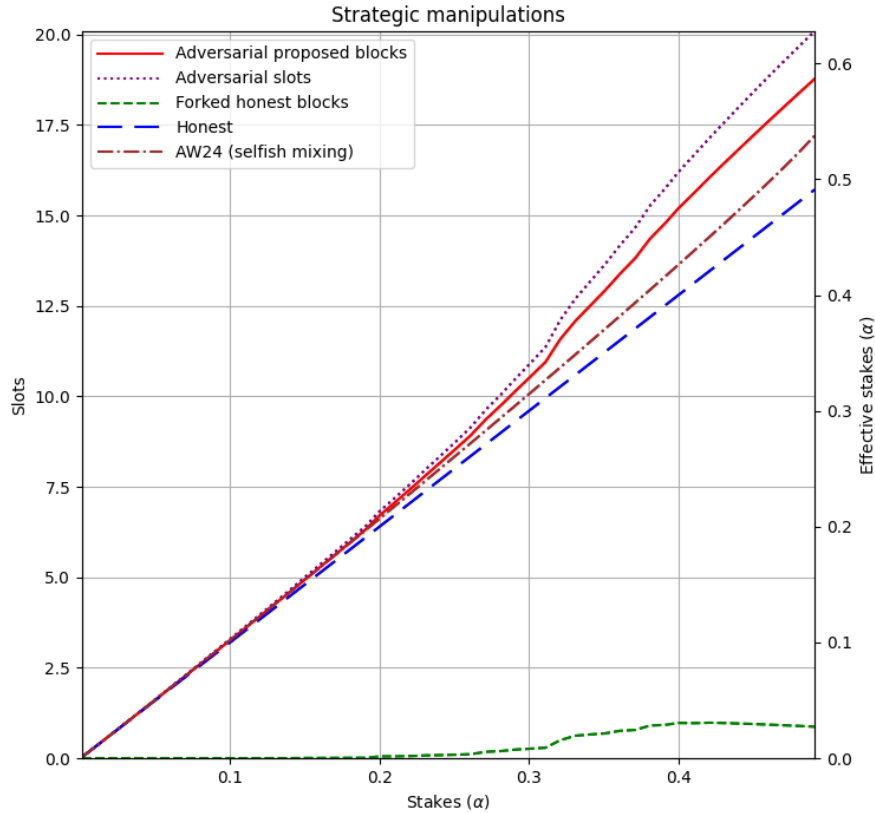
politikájával:

$$\Gamma_\pi := \lim_{N \rightarrow \infty} \mathbb{E} \left[\frac{1}{N} \sum_{e=1}^N \left(\mathcal{P}(e+2, R^e, \mathcal{A}) - \mathcal{S}(e, R^e, \mathcal{A}) \right) \right], \quad (3)$$

ahol $\mathcal{P}(e+2, R^e, \mathcal{A})$ a támadónak az $e+2$ epochban jutó blokkok száma a R^e kimenet mellett, $\mathcal{S}(e, R^e, \mathcal{A})$ pedig az e epochban feláldozott (kihagyott) blokkok száma.

1.0.1 Ellenintézkedés: időzáras titkosítás (timelock encryption)

A forking támadás mélyebb oka, hogy a támadó *előre látja*, mely RANDAO-kimenet hány jövőbeli slotot oszt neki, és így a kihagyásról vagy forkolásról tájékozott döntést hozhat. Ha ezt a jövőbeli információt el lehetne rejteni a döntés pillanatáig, a manipuláció elveszítené az alapját. Erre kínál megoldást az időzáras titkosítás (timelock encryption, azaz olyan titkosítás, amelynek titkos szövege csak egy előre megadott idő elteltével



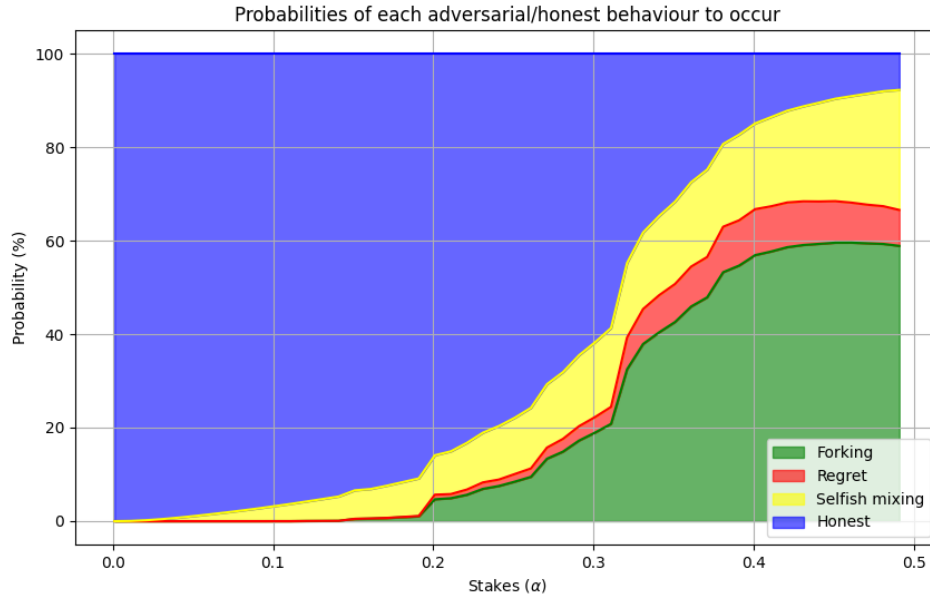
3. ábra: Különböző RANDAO-manipulációs stratégiák hatékonysága a stake-arány (α) függvényében. A forkolást és selfish mixinget együtt alkalmazó támadó (piros) több blokkot szerez, mint a csak selfish mixinget használó korábbi munka (AW24, barna). A zöld görbe a kiforkolt becsületes blokkok számát mutatja – ez a romló láncminőség ára.

fejthető vissza): a validátorok a hozzájárulásukat időzárral titkosítva adhatnák le, így a kimenet csak akkor válik ismertté, amikor már nem lehet rajta változtatni.

A *tlock* séma [2] egy gyakorlatban is működő időzáras titkosítás, amely egy küszöb- (threshold-) hálózatra épül. Az alapötlet, hogy a jövőbeli visszafejtő kulcsot egy küszöb-aláírás szolgáltatja: a hálózat minden időszelethez (round, azaz időintervallum) közzétesz egy BLS-aláírást, és ez az aláírás pontosan az adott időhöz tartozó privát kulcs. A konstrukció Boneh és Franklin azonosság-alapú titkosítását (identity-based encryption, azaz nyilvános kulcs helyett egy nyilvános azonosságra – itt az időszelet sorszámára – titkosító séma) [3] használja, ahol az azonosság szerepét az időszelet sorszáma tölti be.

A protokoll primitívei. A séma egy III-as típusú bilineáris csoporton (e, G_1, G_2, G_T, p) dolgozik, ahol $e : G_1 \times G_2 \rightarrow G_T$ a (nem-degenerált, hatékonyan számítható) párosítás, és négy kriptográfiai hash-függvényt (ütközés- és ősképpálló leképezést) használ:

$$\begin{aligned} H_1 : \{0, 1\}^* &\rightarrow G_2, & H_2 : G_T &\rightarrow \{0, 1\}^\ell, \\ H_3 : \{0, 1\}^\ell \times \{0, 1\}^\ell &\rightarrow \mathbb{Z}_p^*, & H_4 : \{0, 1\}^\ell &\rightarrow \{0, 1\}^\ell. \end{aligned}$$



4. ábra: Az egyes stratégiai viselkedések valószínűsége a stacionárius eloszlásban, a stake-arány függvényében. Nagy stake mellett a racionális szereplő egyre gyakrabban forkol, illetve selfish mixinget alkalmaz; $\alpha = 0,3$ -nál az epochok 38,24%-ában manipulál.

A hálózatot négy eljárás írja le; az első kettő a küszöb-hálózat elosztott (distributed, azaz több fél között megosztott) protokollja, az utóbbi kettő bárki által önállóan futtatható:

- **tSetUp** (elosztott kulcsgenerálás, DKG): a n résztvevő a Gennaro és társai által javasolt elosztott kulcsgenerálási protokollt [4] futtatja. Ennek végén minden i fél csak a saját s_i titkos részét (share, azaz megosztás-darab) ismeri a s közös titkos kulcsból, a nyilvános kimenet pedig a hálózati nyilvános kulcs $P = sG_1 \in G_1$. A s kulcs sosem áll elő egyben; bármely legalább t elemű I részhalmaz Lagrange-interpolációval rekonstruálhatja a hozzá tartozó aláírást.
- **tRoundKey** (időszelet-kulcs protokoll): ez maga a küszöb-BLS [5] aláírás a ρ időszelet sorszáma. Minden i fél kiszámítja a részaláírást $\pi_\rho^{(i)} = s_i H_1(\rho)$, majd t darab (ellenőrzött) részaláírásból a teljes időszelet-kulcs

$$\pi_\rho = \sum_{i \in I} \pi_\rho^{(i)} L_i(0) = s H_1(\rho) \in G_2 \quad (4)$$

áll elő, ahol L_i az I halmazon értelmezett i -edik Lagrange-polinom. Ezt a hálózat időszeletenként közlésezi.

- **Encrypt és Decrypt**: a titkosítás és visszafejtés a Boneh–Franklin séma választott-titkosszöveg-biztos (CCA-biztos) változata, lásd a 1. és 2. algoritmust.

A kulcsgondolat, hogy a titkosító a $PK_\rho = e(P, H_1(\rho))$ „időszeletre szóló nyilvános kulcsot” a nyilvános P -ből és az időszelet sorszámból önállóan ki tudja számítani – a visszafejtéshez viszont a π_ρ időszelet-kulcsra van szükség, amelyet a hálózat csak az adott idő elértekor tesz közzé. A $e(U, \pi_\rho) = e(G_1, H_1(\rho))^{rs} = e(P, H_1(\rho))^r$ egyenlőség (a

Algoritmus 1 tlock – titkosítás egy ρ időszetre

```

1: procedure ENC( $pp, \rho, M$ )                                 $\triangleright M \in \{0, 1\}^\ell$  az üzenet
2:    $PK_\rho \leftarrow e(P, H_1(\rho))$                          $\triangleright$  időszetre szóló nyilvános kulcs
3:    $\sigma \leftarrow_{\$} \{0, 1\}^\ell$                              $\triangleright$  véletlen egyszer használatos érték (nonce)
4:    $r \leftarrow H_3(\sigma, M)$ 
5:    $U \leftarrow rG_1$                                          $\triangleright$  efemer (egyszer használatos) nyilvános kulcs
6:    $V \leftarrow \sigma \oplus H_2(PK_\rho^r)$                      $\triangleright$  a  $\sigma$  elrejtése (commitment)
7:    $W \leftarrow M \oplus H_4(\sigma)$                          $\triangleright$  egyszeri kulcsú (one-time-pad) rejtjelezés
8:   return  $ct_\rho = (U, V, W)$ 
9: end procedure

```

Algoritmus 2 tlock – visszafejtés a π_ρ időszet-kulccsal

```

1: procedure DEC( $pp, \rho, \pi_\rho, ct_\rho$ )
2:    $(U, V, W) \leftarrow ct_\rho$ 
3:    $\sigma' \leftarrow V \oplus H_2(e(U, \pi_\rho))$ 
4:    $M' \leftarrow W \oplus H_4(\sigma')$ 
5:    $r \leftarrow H_3(\sigma', M')$ 
6:   if  $U = rG_1$  then                                        $\triangleright$  a commitment ellenőrzése
7:     return  $M'$ 
8:   else
9:     return  $\perp$ 
10:  end if
11: end procedure

```

párosítás bilinearitása miatt) garantálja, hogy a két oldalon ugyanaz a megosztott titok áll elő.

A séma egyik gyakorlati előnye, hogy a küszöb-hálózatnak nem kell tudnia a titkosításról: csak a megszokott módon közzéteszi az időszetenkénti aláírásait, a titkosítást és visszafejtést pedig bárki elvégezheti. A RANDAO-manipuláció elleni védelem így abból fakad, hogy ha a validátorok hozzájárulásai a leadás pillanatában még időzárral titkosítottak, a támadó a döntésekor nem ismeri a jövőbeli javasloi listát, ezért a kihagyás vagy a forkolás kimenete sem jósolható meg – a manipulációnak megszűnik a várható haszna. Ugyanez az elv alkalmazható a titkosított tranzakció-várólistákra (encrypted mempool, azaz a feldolgozásra váró, de titkosított tranzakciók halmaza) is a MEV és a front-running (versenyelőny-szerzés mások tranzakciói elé sorolással) ellen.

A gyakorlati megvalósítás egy létező küszöb-hálózatra, a League of Entropy nevű rendszerre épül: ez egy 2020 óta üzemelő, egymásban csak korlátozottan megbízó szervezetek (vállalatok, egyetemek, alapítványok) által működtetett, több kontinensre elosztott csomópontokból álló nyilvános véletlenség-forrás (randomness beacon), amely a drand szoftverrel néhány másodpercenként küszöb-BLS-aláírt véletlen értéket tesz közzé, és így a tlock sémához szükséges időszet-kulcsokat szolgáltatja.

Hivatkozások

- [1] Á. Nagy, J. Tapolcai, I. A. Seres, B. Ladóczki, *Forking the RANDAO: Manipulating Ethereum's Distributed Randomness Beacon*, Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security (CCS '25), Taipei, Taiwan, 2025. DOI: 10.1145/3719027.3744852.
- [2] N. Gailly, K. Melissaris, Y. Romailier, *tlock: Practical Timelock Encryption from Threshold BLS*, IACR Cryptology ePrint Archive, Report 2023/189, 2023.
- [3] D. Boneh, M. Franklin, *Identity-Based Encryption from the Weil Pairing*, Advances in Cryptology – CRYPTO 2001, Lecture Notes in Computer Science, vol. 2139, Springer, pp. 213–229, 2001.
- [4] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, *Secure Distributed Key Generation for Discrete-Log Based Cryptosystems*, Advances in Cryptology – EUROCRYPT '99, Lecture Notes in Computer Science, vol. 1592, Springer, pp. 295–310, 1999.
- [5] A. Boldyreva, *Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme*, Public Key Cryptography – PKC 2003, Lecture Notes in Computer Science, vol. 2567, Springer, pp. 31–46, 2003.