A 24 szó: mi az a kriptográfiai identitás

A kriptográfiai identitás nem jelszó: egyetlen szerver sem tárolja, és nem állítható vissza. A BIP39 mechanizmus didaktikus magyarázata, hogy miért pontosan huszonnégy szó, és milyen valós teher hárul arra, aki birtokolja őket.

Hogy értsük egymást: Ha elfelejted a Gmail-jelszavadat, a Google visszaállítja neked. Ha elveszíted a kriptográfiai identitást alkotó 24 szót, nincs kitől elkérned őket. Nem arról van szó, hogy az eljárás szigorú — hanem arról, hogy a másik oldalon nem létezik senki. Ez a különbség a lényeg.

A különbség a jelszó és az identitás között

A jelszó a klasszikus internetes modellben nem a felhasználó identitása. Hanem egy bizonylat. A felhasználónak van identitása — név, e-mail cím, ügyfélszám —, és ahhoz, hogy bizonyítsa egy szervernek, hogy ő az, akinek mondja magát, bemutat egy jelszót, amelyet a szerver összehasonlít egy tárolt lenyomattal. Ha a lenyomatok egyeznek, a szerver engedélyezi a munkamenetet. Ha a jelszó elveszik, a felhasználó ugyanaz a felhasználó marad; amit elveszít, az a bizonylat, és létezik egy helyreállítási folyamat — e-mail a regisztrált címre, biztonsági kérdés —, amellyel az pótolható.

A kriptográfiai identitás másképp működik. Ez nem egy hitelesítő adat, amelyet valaki összehasonlít egy tárolt lenyomattal; ez *önmagában* egy teljes matematikai titok. Mindegy, hol található — papíron, eszközön, vagy akár egy idegen szerveren —: az identitás a matematikája miatt létezik, nem pedig az alapján, aki érvényesíti. Itt egy olyan tulajdonság jelenik meg, mint amit a «Mi is valójában a SHA-256» című írásban láttunk: a birtoklást nem a titok bemutatásával bizonyítják, hanem azzal, hogy aláírásra használják. Az így létrejött aláírást bárki ellenőrizheti egy nyilvános értékkel, amely matematikailag magából a titokból származik, anélkül, hogy ismernie kellene a titkot, és anélkül, hogy harmadik fél közvetítene az ellenőrzésben. Akinek megvan a titok, az maga az identitás; aki elveszíti, az megszűnik annak lenni. Az ítélet kategorikus: **nincs senki, akitől kérhetnéd az identitás visszaadását. Ez a valaki nem létezik, mert eleve nem is birtokolta azt.**

Amit huszonnégy szó képvisel

A kriptográfiai identitást általában egy harminckét bájtos — kétszázötvenhat bites — matematikai titok képviseli. Egy szám, amelyet nehéz megjegyezni, és még nehezebb hiba nélkül leírni. A kriptoipar 2013-ban oldotta meg ezt a problémát egy kicsi és elegáns, BIP39 nevű szabvánnyal: egy módszerrel, amellyel ezt a kétszázötvenhat bitet egy huszonnégy szóból álló sorozatként lehet megjeleníteni, amelyeket egy kétezer-negyvennyolc szavas hivatalos listából választanak ki. A mögötte lévő aritmetika elegánsan illeszkedik; aki részletesen akarja látni, a margón megtalálja.

A számolás a végéről indul. A titok kétszázötvenhat bitjét akarjuk megjeleníteni nyolc bit ellenőrző összeg (checksum) hozzáadásával: összesen kétszázhatvannégy bit. Ha ezeket huszonnégy szóra osztjuk el — ami kezelhető szám a veszteségmentes feljegyzéshez és diktáláshoz —, minden szónak pontosan tizenegy bit információt kell hordoznia. Tizenegy bit pedig kettő a tizenegyediken lehetőséget jelent, azaz kétezer-negyvennyolcat. Ezért pontosan ekkora a hivatalos BIP39 szókincs: a lista a problémához mérten létezik, nem fordítva.

A számolás nem dekoratív. Ha valaki huszonhárom szót helyesen ír le, de a huszonnegyediknél hibázik, az ellenőrző összeg észlelni fogja: a szoftver azt mondja neki, hogy „ez a sorozat érvénytelen”. Ha valaki mind a huszonnégyet helyesen írja le, a szoftver egyértelműen ugyanazt az identitást fogja levezetni. A szólista kiválasztása is tudatos: a BIP39 szókincs szavai rövidek, jól megkülönböztethetőek, ékezet nélküliek, és úgy választották ki őket, hogy minimálisra csökkentsék a fonetikai és helyesírási tévesztéseket. Ez egy olyan szókincs, amelyet arra terveztek, hogy az emberek veszteségmentesen megjegyezhessék, leírhassák és diktálhassák.

A mondattól a kulcsig

A huszonnégy szó nem a kriptográfiai kulcs, amely aláírja az üzeneteket. Az eredeti entrópia visszaállítható reprezentációi, amelyek a PBKDF2 nevű determinisztikus folyamat révén egy hatvannégy bájtos magvúvá (seed) alakulnak át. Ebből a magból származnak, szintén determinisztikusan, azok a konkrét kriptográfiai kulcsok, amelyeket a felhasználó használ: egy privát kulcs az aláíráshoz és egy megfelelő nyilvános kulcs, amelyet az aláírások ellenőrzéséhez tesznek közzé. Ugyanaz a mechanizmus különböző rendszerekben: a kriptovaluták a secp256k1 görbét használják; a Signal protokoll és sok modern rendszer az Ed25519-et használja a Curve25519 görbén. Egy konkrét görbe, például az Ed25519 esetében a BIP32 és a SLIP-0010 szabványok ezt a hatvannégy bájtos magot veszik, és determinisztikusan levezetik azt a harminckét bájtot, amely a tényleges aláíró kulcsot alkotja — ugyanazt a harminckét bájtot, amellyel a következő szakasz kódpéldája kezdődik.

Ez a standard módja annak, ahogyan az egész iparág bemutatja a mechanizmust a felhasználónak —kriptovaluta-tárcák, decentralizált identitáskezelők, a Signal a tartós identitás részében, a Solo2 közöttük—: a felhasználó a gyakorlatban soha nem látja a magot vagy a származtatott kulcsokat. A huszonnégy szót látja az identitása létrehozásakor, és opcionálisan felírja egy papírra. A szavak ezután az eszközei között utaznak, amikor migrálni akarja az identitást: beírja őket az új alkalmazásba, az alkalmazás ugyanazt a magot, ugyanazokat a kulcsokat, ugyanazt az identitást vezeti le. Ez egy hordozható, kriptográfiailag szilárd és ésszerű keretek között megjegyezhető mechanizmus.

Hogyan írjunk alá a kulccsal (egy Zig-ecsetvonás)

Zig-ben, miután megvan a huszonnégy szóból származtatott harminckét bájtos mag, az üzenet aláírása az Ed25519-cel néhány sorban elfér:

```
const std = @import("std");
const Ed25519 = std.crypto.sign.Ed25519;

// 'semilla' son los 32 bytes derivados de las 24 palabras.
const par = Ed25519.KeyPair.create(semilla);

// Firmar un mensaje con la clave privada:
const mensaje = "Este mensaje lo escribí yo.";
const firma = try par.sign(mensaje, null);

// Cualquiera con la clave pública del par puede verificar:
try Ed25519.Signature.verify(firma, mensaje, par.public_key);
```

Az aláírási művelet hatvannégy bájtot eredményez —ezt hívják aláírásnak—, amelyet csak a megfelelő privát kulcsból lehetett létrehozni. Az ellenőrzés nyilvános: bárki, aki rendelkezik a nyilvános kulccsal, ellenőrizheti, hogy az aláírás megfelel-e az üzenetnek. A privát kulcs nélkül senki sem tud érvényes aláírást készíteni az adott üzenethez; a nyilvános kulccsal mindenki felismerheti, ha egy aláírás érvényes. Ez az aszimmetria teszi lehetővé, hogy az aláíró anélkül bizonyítsa a szerzőséget, hogy megosztaná a titkot.

Az előző példa a kézikönyv minimális változata. A Solo2 valódi kódjában a lánc két fájlon halad keresztül, az egyik JavaScript nyelven íródott, a felhasználó böngészőjében fut, és a huszonnégy szóból rekonstruálja az entrópiát, a másik Zig nyelven a *zcatcrypto* könyvtárban, amely átveszi ezt az entrópiát, és levezeti a konkrét kriptográfiai kulcsokat. A böngésző oldaláról indulva:

```
// solo2/web-app/js/lib/bip39.js
async function mnemonicToEntropy(mnemonic, lang) {
    const validation = await validateMnemonic(mnemonic, lang);
    if (!validation.valid) {
        return { entropy: null, valid: false, error: validation.error };
    }
    const wordlist = WORDLISTS[lang || 'en'];
    const words = mnemonic.trim().split(/\s+/);

    // Cada palabra aporta 11 bits (su índice en la lista de 2048).
    let bits = '';
    for (let i = 0; i < words.length; i++) {
        bits += wordlist.indexOf(words[i]).toString(2).padStart(11, '0');
    }

    // 24 palabras = 264 bits. Los primeros 256 son la entropía.
    const entropyBytes = new Uint8Array(32);
    for (let j = 0; j < 32; j++) {
        entropyBytes[j] = parseInt(bits.slice(j * 8, (j + 1) * 8), 2);
    }
    return { entropy: entropyBytes, valid: true };
}
```

Ez a harminckét bájtnyi entrópia, egy másik, ugyanabban a lépésben levezetett harminckét bájttal együtt, a Zig WebAssembly moduljába utazik, amely létrehozza a tényleges Ed25519 kulcsokat. A teljes függvény a végső memóriatörléssel együtt elfér egyetlen képernyőn:

```
// zcatcrypto/wasm/bindings/identity.zig
const Ed25519 = std.crypto.sign.Ed25519;
const X25519  = std.crypto.dh.X25519;

export fn identity_generate() ?*IdentityHandle {
    var seed: [64]u8 = undefined;
    if (!common.getRandomBytes(&seed)) return null;

    const handle = common.wasm_allocator.create(IdentityHandle) catch return null;

    // Bytes 0..31: semilla determinista del par Ed25519 (firma).
    const sign_kp = Ed25519.KeyPair.generateDeterministic(seed[0..32].*) catch {
        common.wasm_allocator.destroy(handle);
        return null;
    };
    handle.sign_secret = sign_kp.secret_key.toBytes();
    handle.sign_public = sign_kp.public_key.toBytes();

    // Bytes 32..63: secreto X25519 (para acordar claves de cifrado con el otro).
    handle.exchange_secret = seed[32..64].*;
    handle.exchange_public = X25519.recoverPublicKey(handle.exchange_secret) catch {
        common.wasm_allocator.destroy(handle);
```

```
        return null;
    };

    @memset(&seed, 0);  // Borra la semilla de la memoria.
    return handle;
}
```

Két részletet érdemes megjegyezni. Az első: ugyanaz a seed mindig ugyanazt a kulcspárt hozza létre – pontosan ez teszi lehetővé az identitás visszaállítását a huszonnégy szó beírásával egy új eszközön. A második: a seed az utolsó sorban kifejezetten törlődik a memóriából. Ezen a ponton túl már maga a függvény sem tudná rekonstruálni a kulcsokat; a felhasználó szavai lennének az egyetlen forrás.

Azoknak, akik kis számokkal szeretnék ellenőrizni. Az aláírási séma teljes egészében végigjárható olyan számokkal, amelyek elég kicsik ahhoz, hogy a számításokat kézzel végezzük el. Aki nem szeretne belemenni az aritmetikába, átugorhatja ezt a blokkot anélkül, hogy elveszítené a cikk fonalát; aki szeretné látni a mechanizmust lépésről lépésre működni, itt megtalálja. **A nyilvános szabályok**, amelyeket bárki elolvashat: egy $p = 23$ prímszám (a valódi Ed25519-ben ez körülbelül hetvenhét számjegyű; huszonhármat használunk, hogy a számítások elférjenek egy oldalon), egy $g = 2$ alap, amelynek rendje ebben a csoportban $q = 11$, és az a konvenció, hogy a g-vel végzett összes aritmetika *módulo p* történik, és minden kitevő *módulo q* redukálódik. **A privát választás**, egyetlen és soha meg nem osztott: az $x = 6$ titok. Ez az identitás.

1. lépés — Az identitás nyilvános része. Egyszer számítják ki, és nyíltan közzéteszik.

$$y = g^x \bmod p$$

$$y = 2^6 \bmod 23 = 64 \bmod 23 = 18$$

Az identitás nyilvános része a **18**. Bárki foghatja és használhatja az ezzel az identitással készített aláírások ellenőrzésére. Senki sem tudja a titkos 6-os számot visszaállítani pusztán a 18-as megfigyelésével: ez a diszkrét logaritmus probléma, amelyre a végén visszatérünk.

2. lépés — Üzenet aláírása. Az identitás birtokosa alá akarja írni az $m = 7$ üzenetet. Először választ egy új $k = 4$ véletlen értéket, amelyet csak egyszer használ fel, és soha nem oszt meg (a valódi Ed25519-ben a k-t determinisztikusan vezetik le az üzenetből és a titokból az újrafelhasználás veszélyének elkerülése érdekében, de a szerepe pontosan ez). Ezután három számot számít ki:

$$r = g^k \bmod p = 2^4 \bmod 23 = 16$$

$$e = H(r, m) \bmod q = (16 + 7) \bmod 11 = 1$$

$$s = (k + x \cdot e) \bmod q = (4 + 6 \cdot 1) \bmod 11 = 10$$

Az aláírás az **(r, s) = (16, 10)** pár. Nyíltan utazik az üzenettel együtt. Bárki elolvashatja. Didaktikai megjegyzés: a valódi Ed25519-ben a H függvény a SHA-512, amely kriptográfiailag robusztus; itt az $e = (r + m) \bmod q$ egyszerűsítést használjuk, hogy az olvasó végigkövethesse a lépéseket hash kiszámítása nélkül. Az algoritmus szerkezete ugyanaz.

3. lépés — Az aláírás ellenőrzése. Az ellenőrző rendelkezik az $y = 18$ nyilvános résszel, az $m = 7$ üzenettel és az $(r, s) = (16, 10)$ aláírással. Ugyanúgy rekonstruálja az e-t – $e = (16 + 7) \bmod 11 = 1$ –, és ellenőrzi, hogy teljesül-e ez az egyenlőség:

$$g^s \bmod p \stackrel{?}{=} r \cdot y^e \bmod p$$

Külön számítja ki a két oldalt:

Izquierda: $2^{10} \bmod 23 = 1024 \bmod 23 = 12$

Derecha: $16 \cdot 18^{1} \bmod 23 = 288 \bmod 23 = 12$

A két oldal eredménye **12**. Az aláírás érvényes. Bárki, akinek megvan a 18-as nyilvános rész, eljuthat erre a következtetésre anélkül, hogy valaha is tudta volna, hogy a titok a 6-os volt.

És mi van egy harmadik féllel, aki hamisítani próbál? Éva látta a csatornán áthaladó összes nyilvános adatot: $p = 23$, $g = 2$, $q = 11$, $y = 18$, $m = 7$, $r = 16$, $s = 10$. Ahhoz, hogy egy *másik* üzenetet írjon alá ezen identitás nevében, ismernie kellene az x-et. Az egyetlen útja az, hogy megkérdezi magától: „melyik x kitevőre teljesül a $2^x \bmod 23 = 18$?”. $p = 23$ esetén kipróbálhatja a 0, 1, 2, 3, … értékeket, és másodpercek alatt megtalálja. De ha a 23-at kicseréljük az Ed25519 valódi méreteinek megfelelő prímszámra, a lehetséges kitevők tere meghaladja a megfigyelhető univerzum atomjainak számát. **Ma az emberiség által nem ismert olyan algoritmus, amely ezt a teret kevesebb mint milliárd év alatt be tudná járni.** Ez ugyanaz a diszkrét logaritmus probléma, amely az előző cikk Diffie-Hellman eljárásának alapja, itt az aláírási sémára alkalmazva.

Amit most végigjártunk, az *pontosan* Schnorr, az az aláírási séma, amelynek az Ed25519 egy elliptikus görbére adaptált változata. A valódi Ed25519-ben minden művelet egy konkrét görbe (Curve25519) pontjain történik prímszám modulo egészek helyett, és a H függvény a SHA-512 a fent használt játékösszeg helyett. A két helyettesítés implementációs beállítás – kriptográfiai ellenállóképesség elérése a brute force ellen, további biztonsági tulajdonságok elérése a k számára. Az algoritmus szerkezete, a három művelet és az aszimmetria oka ugyanaz.

Itt érdemes egy rövid szünetet tartani, mert a teljes lánc egy gyors pillantásra összetéveszthető a trió egy másik primitívjével: a hash-sel. Ez nem az. A hash egy egyedi függvény, amely tömörít – sok bájt megy be, egy rövid ujjlenyomat jön ki, ott véget ér az út. A kriptográfiai identitás egy kiegészítő matematikai pár: a titok marad és aláír; a nyilvános párját közzéteszik és ellenőriznek. Ahol a hash az információt egyetlen irányba omlasztja össze, az identitás aszimmetriát hoz létre két fél között. A hash tanúsítja, mit mondtak; az identitás tanúsítja, ki mondta.

Amit a mondat nem

Három gyakori tévhitet érdemes eloszlatni. A mondat nem jelszó a szó szoros értelmében: nem hasonlítják össze egy szerveren tárolt ujjlenyomattal; a felhasználó eszközébe írják be az identitás matematikai rekonstruálásához. A mondat nem állítható vissza: ha elveszik, nincs kitől elkérni; ha duplikálják, az identitást is duplikálják. A mondat nem az identitástól elválasztható hitelesítő adat: a mondat *maga* az identitás. Aki birtokolja, felléphet identitásként, további engedély, engedélyezési folyamat vagy a helyreállítás lehetősége nélkül.

Ez a harmadik tulajdonság az, ami megváltoztatja az ügy súlyát. Az elveszett jelszó adminisztratív kényelmetlenség. Az elveszett kriptográfiai identitás maga az identitás. Egy harmadik fél által megtalált papír a mondattal nem csak a fióklopás kockázata: ez a teljes identitás átadása. A rendszer ígérete — hogy senki ne vonhassa vissza az identitásodat, és ne blokkolhasson önkényesen — elválaszthatatlanul együtt jár a felelősséggel — hogy te vagy az egyetlen őrzője valaminek, amit senki sem tud visszaállítani helyetted.

Az ígéret és a súly

A kriptográfiai identitásmodellt gyakran *önszuverénnek* —angolul self-sovereign— nevezik. A szóválasztás tudatos, és meglehetősen pontosan leírja az állapotot. A felhasználó szuverén az identitása felett egy szinte középkori értelemben: nem adja meg semmilyen király, semmilyen kibocsátó, semmilyen központi hatóság; és ezek egyike sem vonhatja vissza. De a középkori uralkodóhoz hasonlóan a felhasználó viseli hibái teljes következményét: nincs régens, aki döntene helyette, ha elveszíti a pecsétet.

A harmadik fél által kezelt identitás és az önszuverén identitás közötti választásra nincs egyetlen univerzális helyes válasz. Egy lényegtelen fórumfiók esetében a kezelt identitás valószínűleg arányos a kockázattal. Egy jogilag kötelező erejű dokumentumokat aláíró szakmai identitás, a saját megtakarításait őrző gazdasági identitás vagy az érzékeny információkat rábízó ügyfelekkel való szakmai kommunikáció identitása esetében a helyzet megváltozik. Ott a kérdés már nem az, hogy «kényelmes-e?», hanem az lesz: «rajtam kívül kinek van hatalma úgy eljárni, mint én, és milyen körülmények között?».

Hol jelenik meg ez a mechanizmus a valódi rendszerekben

A BIP39 a Bitcoin világában született 2013-ban, és gyorsan elterjedt a teljes kriptovaluta-ökoszisztémában: ma már minden komoly pénztárca elfogadja a tizenkettő vagy huszonnégy szóból álló BIP39-kifejezést a tulajdonosa gazdasági identitásának biztonsági mentéseként. A kriptovalutákon kívül ugyanaz az alapvető koncepció — egy kriptográfiai pár, amely közvetítő nélkül bizonyítja a szerzőséget — más, eltérő szintaxisú rendszerekben is megjelenik. Az SSH-kulcsok, amelyeket egy rendszergazda használ a szervereihez való hozzáféréshez, klasszikus példák: egy privát kulcs, amelyet a rendszergazda a saját gépén őriz, és egy nyilvános, amelyet minden szerverre átmásolnak; semmilyen központosított szolgáltatáshoz hasonlítható egység nem avatkozik be. A Signal protokoll Ed25519-et használ az eszközön lévő tartós kulcsanyaggal; az európai eIDAS a minősített aláírás tekintetében ugyanerre a kriptográfiai elvre épül, azzal a különbséggel, hogy a kulcsot a felhasználó helyett egy minősített bizalmi szolgáltató őrzi.

A Solo2, e kiadvány kiadói platformja, egy huszonnégy szóból álló BIP39-kifejezést használ minden felhasználó identitásaként. A felhasználó a fiókja létrehozásakor egyszer látja a szavakat. Ezeket nem tárolják sem a Solo2, sem más szerverein: ha a felhasználó feljegyzi és megőrzi őket, örökre megtartja identitását. Ha elveszíti őket, elveszítette őket. Ez a közvetítő nélküli architektúra következetes következménye: ha a Solo2 vissza tudná adni az identitást az azt elvesztő felhasználónak, akkor bárkinek odaadhatná, aki nyomást gyakorol a Solo2-re, hogy adja át neki.

A szakmai olvasónak

Négy megfontolás azok számára, akik szakmai környezetben fontolgatják a kriptográfiai önrendelkező (autosoberana) identitás bevezetését:

1. A kifejezés maga az identitás. A fizikai megőrzés — papír, több másolat különböző helyeken, esetleg gravírozott fém a hosszú távú használathoz — több garanciát nyújt, mint a digitális megőrzés, amely növeli a támadási felületet anélkül, hogy csökkentené a veszteség kockázatát.
2. Nincs helyreállítás. A folyamat megtervezése abból a feltételezésből kiindulva, hogy egy napon az elsődleges másolat elvész, sokkal célszerűbb, mint a veszteség napján szembesülni ezzel. Egy második, földrajzilag elkülönített másolat szinte minden forgatókönyvet megold.
3. Ez nem ugyanaz, mint egy eIDAS minősített tanúsítvány. Az Unión belüli minősített aláíráshoz — közjegyzői okiratok, bizonyos hivatali eljárások — a jogszabályok minősített szolgáltatót írnak elő, aki a kulcsot őrzi. A kriptográfiai önrendelkező identitás a szakmai kommunikációt és a bizonyító erejű okirat-aláírást szolgálja, de nem helyettesíti automatikusan a minősített tanúsítványt azokban az esetekben, amikor a jogszabály azt előírja.
4. Ha az identitást át kell ruházni — öröklés, szakmai utódlás, tevékenység beszüntetése —, célszerű az eljárást előre, nem pedig utólag előkészíteni. A pecsétviasszal (lacre) lezárt borítékokkal végzett formális eljárások, a végrendeleti végrehajtónak szóló utasítások, a közjegyzői letétbe helyezés olyan klasszikus megoldások, amelyek tökéletesen összeegyeztethetőek az eszköz kriptográfiai jellegével.

Ez a cikk lezárja azt a fogalmi triót, amely a ciklust indította — hash, titkosítás, identitás —. A három gondolat egymásra épül: a hash adja a megmásíthatatlan ujjlenyomatot, a titkosítás adja a bizalmasságot megbízható harmadik fél nélkül, az identitás adja a szerzőséget engedélyező harmadik fél nélkül. Mindháromnak van egy olyan tulajdonsága, amely szintén nem ideológiai: átruházzák a szolgáltatást kezelőtől a szolgáltatást

használóra azokat a technikai képességeket, amelyek hagyományosan az operátornál voltak. Velük együtt felelősségeket is átruháznak. Bármelyikről való őszinte beszéd megköveteli a másik kettőről való beszédet is.

Források és további olvasnivalók

- Palatinus, M.; Rusnak, P.; Voisine, A.; Bowe, S. — *BIP-0039: Mnemonic code for generating deterministic keys*, 2013-as Bitcoin fejlesztési javaslat. De facto szabvány a helyreállítási kifejezésekhez a kriptoiparban.
- RFC 8032 — Edwards-Curve Digital Signature Algorithm (EdDSA), beleértve az Ed25519-et. IETF, 2017. január. A kortárs ipar nagy részében használt aláírási séma normatív specifikációja.
- RFC 2898 — PKCS #5: Password-Based Cryptography Specification, 2.0-s verzió. IETF, 2000. szeptember. Meghatározza a BIP39 kifejezésből magig (seed) történő származtatáshoz használt PBKDF2 algoritmust.
- A 910/2014/EU rendelet (eIDAS) és annak a 2024/1183/EU rendelet (eIDAS 2) általi továbbfejlesztése — az elektronikus azonosításra és a minősített aláírásra vonatkozó európai keretrendszer. Az önrendelkezőtől eltérő rendszer, de fogalmilag ugyanazokra a kriptográfiai primitívekre épül.
- Allen, C. — *The Path to Self-Sovereign Identity* (2016). Kanonikus szöveg az önrendelkező modell elveiről és elkötelezettségeiről, korábbi, de releváns a kortárs megoldási család megértéséhez.

← ElőzőAz üzleti modell mint a bizalom jeleKövetkező → Self-hosting mint szakmai gyakorlat

Legutóbbi olvasmányok

- Reflexió · 2026. június 29. Nem vagy névtelen
- Reflexió · 2026. május 27. Amit egy aláírás nem tud megoldani
- Elemzés · 2026. május 26. Valódi vs. látszólagos adatvédelem: a kérdések, amelyeket érdemes feltenni

Vigye magával ezt a cikket, ahová csak szüksége van rá.

↓ Markdown ↓ Egyszerű szöveg ↓ PDF

A fájl letöltődik az Ön eszközére. Onnan elmentheti, importálhatja a Solo2-be, vagy megoszthatja bárhol. A Cuadernos nem dönt Ön helyett a fájl sorsáról.

Viaszpecsét · SHA-256 76265438bc76c3c1ab5aa4025221b3bbbda4fc72f8559e33a54604cf99bd0a86

Cuadernos Lacre · A Menzuri Gestión S.L. kiadványa ·
írta R.Eugenio · szerkesztette a Solo2 csapata.