Cele 24 de cuvinte: ce este o identitate criptografică

O identitate criptografică nu este o parolă: niciun server nu o stochează și nu poate fi recuperată. O explicație didactică a mecanismului BIP39, de ce exact douăzeci și patru de cuvinte și ce responsabilitate reală revine celui care le posedă.

Pentru a ne înțelege: Dacă îți uiți parola de Gmail, Google o resetează pentru tine. Dacă pierzi cele 24 de cuvinte care compun o identitate criptografică, nu ai cui să le ceri. Nu e vorba că procedura ar fi strictă — ci că nu există nimeni la celălalt capăt. Această diferență este esențială.

Diferența dintre o parolă și o identitate

O parolă, în modelul clasic de internet, nu este identitatea utilizatorului. Este un document justificativ. Utilizatorul are o identitate — un nume, un e-mail, un număr de client — și, pentru a demonstra unui server că este cine pretinde a fi, prezintă o parolă pe care serverul o compară cu o amprentă pe care o avea stocată. Dacă amprentele coincid, serverul acordă sesiunea. Dacă parola este pierdută, utilizatorul rămâne același utilizator; ceea ce pierde este documentul justificativ, și există o procedură de recuperare — un e-mail la adresa înregistrată, o întrebare de securitate — pentru a-l restitui.

O identitate criptografică funcționează altfel. Nu este o acreditare pe care cineva o compară cu o amprentă stocată; *este* un secret matematic complet în sine. Nu contează unde se află — pe o hârtie, într-un dispozitiv, chiar și pe un server străin —: identitatea există prin matematica sa, nu prin cine o validează. Aici apare o proprietate similară cu cea pe care am văzut-o în «Ce este de fapt SHA-256»: posesia nu se demonstrează prin expunerea secretului, ci prin folosirea acestuia pentru a semna. Semnătura astfel produsă poate fi verificată de oricine cu o valoare publică derivată matematic din secretul în sine, fără a fi nevoie de cunoașterea secretului și fără ca un terț să intervină în verificare. Cine are secretul, este identitatea; cine îl pierde, încetează să mai fie. Sentința este categorică: **nu există nimeni cui să îi ceri să îți dea identitatea înapoi. Acea persoană nu există, pentru că nu o deținea în primul rând.**

Ce reprezintă douăzeci și patru de cuvinte

Identitatea criptografică este reprezentată de obicei printr-un secret matematic de treizeci și doi de octeți — două sute cincizeci și șase de biți. Un număr greu de reținut și mai ales greu de transcris fără erori. Industria criptografică a rezolvat această problemă în 2013 cu un standard mic și elegant numit BIP39: o modalitate de a reprezenta acei două sute cincizeci și șase de biți ca o secvență de douăzeci și patru de cuvinte luate dintr-o listă oficială de două mii patruzeci și opt. Aritmetica din spate se potrivește cu eleganță; cine dorește să o vadă în detaliu o găsește în notă.

Numărătoarea începe de la sfârșit. Vrem să reprezentăm cei două sute cincizeci și șase de biți ai secretului adăugând opt biți de checksum: două sute șaizeci și patru de biți în total. Dacă îi împărțim în douăzeci și patru de cuvinte — un număr ușor de gestionat pentru notare și dictare fără pierderi — fiecare cuvânt trebuie să furnizeze exact unsprezece biți de informație. Iar unsprezece biți înseamnă doi la puterea a unsprezecea posibilități, adică două mii patruzeci și opt. De aceea, vocabularul oficial BIP39 are exact această dimensiune: lista există pe măsura problemei, nu invers.

Numărătoarea nu este decorativă. Dacă cineva transcrie corect douăzeci şi trei de cuvinte şi greşeşte la al douăzeci şi patrulea, checksum-ul îl va detecta: software-ul îi va spune „această secvenţă nu este validă”. Dacă cineva le transcrie corect pe toate cele douăzeci şi patru, software-ul va deriva aceeaşi identitate fără ambiguitate. Alegerea listei de cuvinte este, de asemenea, deliberată: cuvintele din vocabularul BIP39 sunt scurte, diferite între ele, fără diacritice, alese pentru a minimiza confuziile fonetice şi ortografice. Este un vocabular conceput pentru a fi reţinut, scris şi dictat de fiinţe umane fără pierderi.

De la frază la cheie

Cele douăzeci şi patru de cuvinte nu sunt cheia criptografică ce semnează mesaje. Ele sunt o reprezentare recuperabilă a entropiei originale care, printr-un proces determinist numit PBKDF2, se transformă într-o sămânţă (seed) de şaizeci şi patru de octeţi. Din acea sămânţă derivă, tot în mod determinist, cheile criptografice concrete pe care le foloseşte utilizatorul: o cheie privată pentru a semna şi o cheie publică corespunzătoare care este publicată pentru a verifica semnăturile. Acelaşi mecanism în sisteme diferite: criptomonedele folosesc curba secp256k1; protocolul Signal şi multe sisteme moderne folosesc Ed25519 pe curba Curve25519. Pentru o curbă concretă precum Ed25519, standardele BIP32 şi SLIP-0010 iau acea sămânţă de şaizeci şi patru de octeţi şi derivă, în mod determinist, cei treizeci şi doi de octeţi care constituie cheia de semnare efectivă — aceiaşi treizeci şi doi de octeţi cu care începe exemplul de cod din secţiunea următoare.

Aceasta este modalitatea standard prin care întreaga industrie prezintă mecanismul utilizatorului —portofele de criptomonede, manageri de identitate descentralizată, Signal în partea sa de identitate persistentă, Solo2 printre ele—: utilizatorul, în practică, nu vede niciodată sămânţa sau cheile derivate. El vede cele douăzeci şi patru de cuvinte la crearea identităţii sale şi, opţional, le notează pe o hârtie. Cuvintele călătoresc apoi între dispozitivele sale atunci când doreşte să migreze identitatea: le introduce în noua aplicaţie, aplicaţia derivă aceeaşi sămânţă, aceleaşi chei, aceeaşi identitate. Este un mecanism portabil, criptografic solid şi, în limitele rezonabilului, memorabil.

Cum se semnează cu cheia (o pensulă de Zig)

În Zig, odată ce ai sămânţa de treizeci şi doi de octeţi derivată din cele douăzeci şi patru de cuvinte, semnarea unui mesaj cu Ed25519 începe în câteva linii:

```
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);
```

Operaţiunea de semnare produce şaizeci şi patru de octeţi —numiţi semnătură— care nu au putut fi generaţi decât din cheia privată corespunzătoare. Verificarea este publică: oricine are cheia publică poate verifica dacă semnătura corespunde mesajului. Fără cheia privată, nimeni nu poate produce o semnătură validă pentru acel mesaj; cu cheia publică, toţi pot detecta dacă o semnătură este validă. Această asimetrie este cea care permite semnatarului să demonstreze paternitatea fără a partaja secretul.

Exemplul anterior este versiunea minimă din manual. În codul real al Solo2, lanţul traversează două fişiere: unul în JavaScript care rulează în browserul utilizatorului şi reconstruieşte entropia din cele douăzeci şi patru de

cuvinte, celălalt în Zig în cadrul bibliotecii *zcatcrypto* care preia acea entropie și derivă cheile criptografice concrete. Începând cu partea de browser:

```
// 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 };
}
```

Cei treizeci și doi de octeți de entropie, împreună cu alți treizeci și doi derivați în același pas, călătoresc către modulul WebAssembly al Zig care generează cheile Ed25519 propriu-zise. Funcția completă, cu curățarea finală a memoriei, încape pe un singur ecran:

```
// 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;
}
```

Două detalii merită menționate. Primul: aceeași sămânță (seed) produce întotdeauna aceeași pereche de chei — tocmai acest lucru permite recuperarea identității prin introducerea celor douăzeci și patru de cuvinte pe un dispozitiv nou. Al doilea: sămânța este ștearsă explicit din memorie în ultima linie. După acel punct, nici măcar funcția însăși nu ar putea reconstrui cheile; cuvintele utilizatorului ar fi singura sursă.

Pentru cine vrea să verifice cu numere mici. Schema de semnătură poate fi parcursă în întregime cu cifre suficient de mici pentru a face calculele manual. Cine preferă să nu intre în aritmetică poate sări peste acest bloc fără a pierde firul articolului; cine vrea să vadă mecanismul funcționând pas cu pas îl va găsi aici. **Regulile publice**, pe care oricine le poate citi: un număr prim $p = 23$ (în Ed25519 real are aproximativ șaptezeci și șapte de cifre; folosim douăzeci și trei pentru ca calculele să încapă pe o pagină), o bază $g = 2$ al cărei ordin în acest grup este $q = 11$, și convenția că toată aritmetica cu g se face *módulo p* și toți exponenții sunt reduși *módulo q*. **Alegerea privată**, una singură și niciodată partajată: secretul $x = 6$. Aceasta este identitatea.

Pasul 1 — Partea publică a identității. Se calculează o singură dată și se publică deschis.

$y = g^x \bmod p$

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

Partea publică a identității este **18**. Oricine o poate lua și o poate folosi pentru a verifica semnăturile făcute cu această identitate. Nimeni, observând doar cifra 18, nu poate recupera secretul 6: aceasta este problema logaritmului discret la care ne vom întoarce la final.

Pasul 2 — Semnarea unui mesaj. Posesorul identității vrea să semneze mesajul $m = 7$. Începe prin a alege o nouă valoare aleatorie $k = 4$, care va fi folosită o singură dată și nu va fi partajată niciodată (în Ed25519 real, k este derivat deterministic din mesaj și din secret pentru a evita pericolul reutilizării, dar rolul pe care îl joacă este exact acesta). Apoi calculează trei numere:

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

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

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

Semnătura este perechea **(r, s) = (16, 10)**. Călătorește la vedere împreună cu mesajul. Oricine o poate citi. Notă didactică: în Ed25519 real funcția H este SHA-512, criptografic robustă; aici folosim simplificarea $e = (r + m) \bmod q$ pentru ca cititorul să poată parcurge pașii fără a fi nevoie să calculeze un hash. Structura algoritmului este aceeași.

Pasul 3 — Verificarea semnăturii. Verificatorul are partea publică $y = 18$, mesajul $m = 7$ și semnătura $(r, s) = (16, 10)$. Reconstruiește e în același mod — $e = (16 + 7) \bmod 11 = 1$ — și verifică dacă această egalitate se respectă:

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

Calculează cele două părți separat:

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

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

Cele două părți dau **12**. Semnătura este validă. Oricine are partea publică 18 poate ajunge la această concluzie fără a fi știut vreodată că secretul a fost 6.

Și un terț care ar încerca să falsifice? Eva a văzut trecând prin canal tot ce este public: $p = 23$, $g = 2$, $q = 11$, $y = 18$, $m = 7$, $r = 16$, $s = 10$. Pentru a semna un mesaj *diferit* în numele acestei identități, ea ar trebui să cunoască x. Singura ei cale este să se întrebe: „pentru ce exponent x se îndeplinește $2^x \bmod 23 = 18$?”. Cu $p = 23$ ea poate încerca 0, 1, 2, 3, … și îl poate găsi în câteva secunde. Dar prin înlocuirea lui 23 cu un prim de dimensiunile reale ale lui Ed25519, spațiul exponenților posibili depășește numărul de atomi din universul observabil. **Nu există astăzi niciun algoritm cunoscut de umanitate care să poată parcurge acel spațiu în mai puțin de miliarde de ani.** Este aceeași problemă a logaritmului discret care stă la baza Diffie-Hellman din articolul anterior, aplicată aici schemei de semnătură.

Ceea ce tocmai am parcurs este *exact* Schnorr, schema de semnătură din care Ed25519 este o variantă adaptată la o curbă eliptică. În Ed25519 real, toate operațiunile se fac pe punctele unei curbe concrete (Curve25519) în loc de pe numere întregi modulo un prim, iar funcția H este SHA-512 în loc de suma de jucărie pe care am folosit-o mai sus. Cele două înlocuiri sunt ajustări de implementare — obținerea rezistenței criptografice la forța brută, obținerea unor proprietăți de securitate suplimentare pentru k —. Structura algoritmică, cele trei operațiuni, motivul asimetriei, sunt aceleași.

Se cuvine aici o scurtă pauză, deoarece întregul lanț poate fi confundat la o privire rapidă cu o altă primitivă din trio: hash-ul. Nu este. Un hash este o funcție unică ce comprimă — intră mulți octeți, iese o amprentă scurtă, acolo se termină drumul. O identitate criptografică este o pereche matematică complementară: secretul rămâne și semnează; contrapartea sa publică se publică și verifică. Acolo unde hash-ul colapsează informația într-un singur sens, identitatea stabilește o asimetrie între două jumătăți. Hash-ul atestă ce s-a spus; identitatea atestă cine a spus.

Ce nu este fraza

Trei idei greșite frecvente merită clarificate. Fraza nu este o parolă în sens propriu: nu este comparată cu o amprentă stocată pe un server; este introdusă pe dispozitivul utilizatorului pentru a reconstrui matematic identitatea. Fraza nu se recuperează: dacă este pierdută, nu există nimeni la cine să o ceri; dacă este duplicată, este duplicată și identitatea. Fraza nu este o acreditare separabilă de identitate: fraza *este* identitatea. Cine o deține poate acționa ca acea identitate, fără permisiune suplimentară, fără proces de autorizare, fără recuperare posibilă.

Această a treia proprietate este cea care schimbă ponderea problemei. O parolă pierdută este o neplăcere administrativă. O identitate criptografică pierdută este identitatea în sine. O hârtie cu fraza găsită de terți nu este un risc de furt al contului: este predarea întregii identități. Promisiunea sistemului — ca nimeni să nu îți poată revoca identitatea sau să te blocheze arbitrar — este însoțită inseparabil de responsabilitatea — că ești singurul custode a ceva ce nimeni nu poate restitui pentru tine.

Promisiunea și ponderea

Modelul de identitate criptografică primește adesea calificativul de *auto-suverană* —self-sovereign în literatura anglo-saxonă—. Alegerea cuvântului este deliberată și descrie condiția cu destul de multă precizie. Utilizatorul este suveran peste identitatea sa într-un sens aproape medieval: nu o acordă niciun rege, niciun emitent, nicio autoritate centrală; și niciuna dintre cele de mai sus nu o poate retrage. Dar, de asemenea, ca monarhul medieval, utilizatorul poartă întreaga consecință a greșelilor sale: nu există niciun regent care să ia decizii în locul său dacă pierde sigiliul.

Alegerea între o identitate gestionată de un terț și o identitate auto-suverană nu are un singur răspuns universal corect. Pentru contul unui forum irelevant, identitatea gestionată este probabil proporțională cu riscul. Pentru o identitate profesională care semnează documente obligatorii din punct de vedere juridic, pentru o identitate

economică ce păzește economii proprii, pentru o identitate de comunicare profesională cu clienți care au încredințat informații sensibile, problema se schimbă. Acolo întrebarea încetează să mai fie „este comod?” și devine „cine, în afară de mine, are puterea de a acționa ca mine și în ce circumstanțe?”.

Unde apare acest mecanism în sisteme reale

BIP39 s-a născut în lumea Bitcoin în 2013 și s-a extins rapid la întregul ecosistem al criptomonedelor: orice portofel serios acceptă astăzi o frază BIP39 de douăsprezece sau douăzeci și patru de cuvinte ca rezervă a identității economice a posesorului său. În afara criptomonedelor, același concept de bază — o pereche criptografică ce dovedește calitatea de autor fără intermediar — apare în alte sisteme cu sintaxă diferită. Cheile SSH pe care un administrator de sisteme le folosește pentru a-și accesa serverele sunt un caz clasic: o cheie privată pe care administratorul o păstrează pe mașina sa și una publică ce este copiată pe fiecare server; nicio entitate comparabilă cu un serviciu centralizat nu intervine. Protocolul Signal folosește Ed25519 cu material de cheie persistent pe dispozitiv; standardele europene eIDAS, în partea lor de semnătură calificată, se bazează pe același principiu criptografic, cu diferența că cheia este păstrată de un furnizor de servicii de încredere calificat în locul utilizatorului.

Solo2, platforma editorială a acestei publicații, folosește o frază BIP39 de douăzeci și patru de cuvinte ca identitate a fiecărui utilizator. Utilizatorul, la crearea contului său, vede cuvintele o singură dată. Acestea nu sunt stocate pe niciun server Solo2 sau al altcuiva: dacă utilizatorul le notează și le păstrează, își menține identitatea pentru totdeauna. Dacă le pierde, le pierde. Este consecința coerentă a unei arhitecturi fără operator la mijloc: dacă Solo2 ar putea returna identitatea utilizatorului care a pierdut-o, ar putea-o oferi și oricui ar face presiuni asupra Solo2 pentru a o obține.

Pentru cititorul profesional

Patru considerații pentru cine evaluează adoptarea identității criptografice auto-suverane (autosoberana) într-un context profesional:

1. Fraza este identitatea. Păstrarea fizică — hârtie, mai multe copii în locuri diferite, eventual metal gravat pentru utilizare pe termen lung — oferă mai multe garanții decât păstrarea digitală, care adaugă suprafață de atac fără a reduce riscul de pierdere.
2. Nu există recuperare. Proiectarea procesului presupunând că într-o zi copia primară se va pierde este mult mai indicată decât descoperirea acestui fapt în ziua în care se pierde. O a doua copie separată geografic rezolvă aproape toate scenariile.
3. Nu este același lucru cu un certificat calificat eIDAS. Pentru semnătura calificată în Uniune — acte notariale, anumite proceduri cu Administrația — legislația impune un furnizor calificat care păstrează cheia. Identitatea criptografică auto-suverană servește pentru comunicarea profesională și semnarea documentelor cu valoare probatorie, dar nu înlocuiește automat certificatul calificat în cazurile în care norma o impune.
4. Dacă identitatea urmează să fie transferată — moștenire, succesiune profesională, încetarea activității — este indicat să se pregătească procedura înainte, nu după. Procedurile formale cu plicuri sigilate cu ceară (lacre), instrucțiuni către un executor testamentar, depunerea la un birou notarial, sunt aranjamente clasice perfect compatibile cu natura criptografică a activului.

Acest articol încheie trioul conceptual care a deschis ciclul — hash, criptare, identitate —. Cele trei idei se construiesc una pe cealaltă: hash-ul oferă amprenta inalterabilă, criptarea oferă confidențialitatea fără o terță parte de încredere, identitatea oferă calitatea de autor fără o terță parte care să o acorde. Toate cele trei împărtășesc o proprietate care nu este nici ea ideologică: ele transferă, de la cel care gestionează un serviciu către cel care îl folosește, capacități tehnice care în mod tradițional aparțineau operatorului. Ele transferă odată cu ele și responsabilități. A vorbi cu onestitate despre oricare dintre cele trei necesită a vorbi și despre celelalte două.

Surse şi lecturi suplimentare

- Palatinus, M.; Rusnak, P.; Voisine, A.; Bowe, S. — *BIP-0039: Mnemonic code for generating deterministic keys*, propunere de îmbunătățire a Bitcoin din 2013. Standard de facto pentru fraze de recuperare în industria cripto.
- RFC 8032 — Edwards-Curve Digital Signature Algorithm (EdDSA), inclusiv Ed25519. IETF, ianuarie 2017. Specificație normativă a schemei de semnătură utilizată în mare parte din industria contemporană.
- RFC 2898 — PKCS #5: Password-Based Cryptography Specification, versiunea 2.0. IETF, septembrie 2000. Definește algoritmul PBKDF2 utilizat în derivarea BIP39 de la frază la sămânță (seed).
- Regulamentul (UE) 910/2014 (eIDAS) și evoluția sa prin Regulamentul (UE) 2024/1183 (eIDAS 2) — cadrul european pentru identitatea electronică și semnătura calificată. Regim diferit de cel auto-suveran, dar susținut conceptual de aceleași primitive criptografice.
- Allen, C. — *The Path to Self-Sovereign Identity* (2016). Text canonic despre principiile și angajamentele modelului auto-suveran, anterior, dar relevant pentru înțelegerea familiei de soluții contemporane.

← PrecedentModelul de afaceri ca semnal de încredereUrmător →Self-hosting ca practică profesională

Lecturi recente

- Reflecție · 29 iunie 2026 Nu ești anonim
- Reflecție · 27 mai 2026 Ceea ce o semnătură nu poate repara
- Analiză · 26 mai 2026 Confidențialitate reală vs. aparentă: întrebările pe care e bine să ți le pui

Luați acest articol cu dumneavoastră oriunde aveți nevoie.

↓ Markdown ↓ Text simplu ↓ PDF

Fișierul se va descărca pe dispozitivul dumneavoastră. De acolo îl puteți salva, importa în Solo2 sau partaja oriunde doriți. Cuadernos nu decide destinația în locul dumneavoastră.

Sigiliu de ceară · SHA-256 763f049da460570ac421f106dabadd531a800332266bdff4ac095ffeeda2ca68

Cuadernos Lacre · O publicație a Menzuri Gestión S.L. ·
scrisă de R.Eugenio · editată de echipa Solo2.