24 từ: danh tính mật mã là gì

Danh tính mật mã không phải là mật khẩu: không có máy chủ nào lưu trữ nó và nó không thể được khôi phục. Một lời giải thích mang tính giáo dục về cơ chế BIP39, tại sao chính xác là hai mươi bốn từ, và trọng trách thực sự đè nặng lên người sở hữu chúng.

Để hiểu nhau hơn: Nếu bạn quên mật khẩu Gmail, Google sẽ đặt lại cho bạn. Nếu bạn mất 24 từ tạo nên danh tính mật mã, không có ai để yêu cầu cấp lại. Không phải vì quy trình nghiêm ngặt — mà là vì không có ai ở đầu bên kia cả. Sự khác biệt đó chính là mấu chốt.

Sự khác biệt giữa mật khẩu và danh tính

Một mật khẩu, trong mô hình internet cổ điển, không phải là danh tính của người dùng. Nó là một chứng từ. Người dùng có một danh tính — một cái tên, một email, một mã số khách hàng — và để chứng minh với máy chủ rằng họ đúng là người họ tự xưng, họ xuất trình một mật khẩu mà máy chủ sẽ so sánh với một dấu vết đã lưu trữ. Nếu dấu vết khớp, máy chủ cấp phiên làm việc. Nếu mật khẩu bị mất, người dùng vẫn là người dùng đó; cái họ mất là chứng từ, và có một quy trình khôi phục — một email gửi đến địa chỉ đã đăng ký, một câu hỏi bảo mật — để cấp lại nó.

Danh tính mật mã hoạt động theo một cách khác. Nó không phải là một thông tin xác thực mà ai đó so sánh với dấu vết đã lưu trữ; nó *chính là* một bí mật toán học hoàn chỉnh. Không quan trọng nó nằm ở đâu — trên giấy, trong một thiết bị, hay thậm chí trên máy chủ của người khác — danh tính tồn tại nhờ toán học của nó, không phải nhờ người xác thực nó. Ở đây xuất hiện một đặc tính tương tự như những gì chúng ta đã thấy trong bài «SHA-256 thực sự là gì»: quyền sở hữu không được chứng minh bằng cách đưa bí mật ra, mà bằng cách dùng nó để ký. Chữ ký được tạo ra theo cách này có thể được bất kỳ ai kiểm tra bằng một giá trị công khai được dẫn xuất toán học từ chính bí mật đó, mà không cần biết bí mật và không cần bên thứ ba can thiệp vào việc kiểm tra. Ai có bí mật, người đó là danh tính; ai làm mất nó, không còn là danh tính nữa. Phán quyết là dứt khoát: **không có ai để yêu cầu trả lại danh tính cho bạn. Người đó không tồn tại, vì ngay từ đầu họ đã không giữ nó.**

Những gì hai mươi bốn từ đại diện

Danh tính mật mã thường được đại diện bởi một bí mật toán học có kích thước ba mươi hai byte — hai trăm năm mươi sáu bit. Một con số khó nhớ và thậm chí còn khó chép lại mà không sai sót. Ngành công nghiệp mật mã đã giải quyết vấn đề này vào năm 2013 với một tiêu chuẩn nhỏ và trang nhã mang tên BIP39: một cách để biểu diễn hai trăm năm mươi sáu bit đó dưới dạng một chuỗi gồm hai mươi bốn từ được lấy từ một danh sách chính thức gồm hai nghìn không trăm bốn mươi tám từ. Số học đằng sau nó khớp một cách trang nhã; ai muốn xem chi tiết có thể tìm thấy ở phần ghi chú.

Việc tính toán bắt đầu từ cuối. Chúng ta muốn biểu diễn hai trăm năm mươi sáu bit bí mật bằng cách thêm tám bit checksum: tổng cộng hai trăm sáu mươi tư bit. Nếu chúng ta chia chúng thành hai mươi bốn từ — một số lượng dễ quản lý để ghi chú và đọc mà không bị thất lạc — mỗi từ phải cung cấp chính xác mười một bit thông tin. Và mười một bit là hai lũy thừa mười một khả năng, tức là hai nghìn không trăm bốn mươi tám. Đó là lý do tại sao từ vựng chính thức của BIP39 có chính xác kích thước đó: danh sách tồn tại theo kích thước của vấn đề, không phải ngược lại.

Việc tính toán không phải để trang trí. Nếu ai đó chép đúng hai mươi ba từ và sai ở từ thứ hai mươi tư, checksum sẽ phát hiện ra: phần mềm sẽ báo cho họ "chuỗi này không hợp lệ". Nếu ai đó chép đúng cả hai mươi tư từ, phần mềm sẽ dẫn xuất ra cùng một danh tính mà không có sự mơ hồ nào. Việc lựa chọn danh sách từ cũng có chủ ý: các từ trong từ vựng BIP39 ngắn, khác biệt nhau, không có dấu phụ, được chọn để giảm thiểu sự nhầm lẫn về ngữ âm và chính tả. Đó là một bộ từ vựng được thiết kế để con người có thể nhớ, viết và đọc mà không bị thất lạc.

Từ cụm từ đến khóa

Hai mươi bốn từ đó không phải là khóa mã hóa dùng để ký tin nhắn. Chúng là một bản đại diện có thể khôi phục của entropy gốc, thông qua một quy trình xác định gọi là PBKDF2, được chuyển đổi thành một hạt giống (seed) sáu mươi bốn byte. Từ hạt giống đó, các khóa mã hóa cụ thể mà người dùng sử dụng được tạo ra một cách xác định: một khóa riêng để ký và một khóa công khai tương ứng được công bố để xác minh chữ ký. Cơ chế tương tự trong các hệ thống khác nhau: tiền điện tử sử dụng đường cong secp256k1; giao thức Signal và nhiều hệ thống hiện đại sử dụng Ed25519 trên đường cong Curve25519. Đối với một đường cong cụ thể như Ed25519, các tiêu chuẩn BIP32 và SLIP-0010 lấy hạt giống sáu mươi bốn byte đó và tạo ra một cách xác định ba mươi hai byte cấu thành nên khóa ký hiệu quả — cùng ba mươi hai byte mà ví dụ mã trong phần tiếp theo bắt đầu.

Đây là cách tiêu chuẩn mà toàn bộ ngành công nghiệp trình bày cơ chế này cho người dùng —ví tiền điện tử, trình quản lý định danh phi tập trung, Signal trong phần định danh bền vững của nó, Solo2 trong số đó—: trên thực tế, người dùng không bao giờ nhìn thấy hạt giống hoặc các khóa được tạo ra. Họ nhìn thấy hai mươi bốn từ khi tạo định danh của mình và tùy chọn ghi chúng ra giấy. Sau đó, các từ này sẽ di chuyển giữa các thiết bị của họ khi họ muốn chuyển đổi định danh: họ nhập chúng vào ứng dụng mới, ứng dụng sẽ tạo ra cùng một hạt giống, cùng các khóa, cùng một định danh. Đó là một cơ chế di động, vững chắc về mặt mã hóa và trong giới hạn hợp lý, có thể ghi nhớ được.

Cách ký bằng khóa (một nét vẽ Zig)

Trong Zig, khi bạn đã có hạt giống ba mươi hai byte được tạo ra từ hai mươi bốn từ, việc ký một tin nhắn bằng Ed25519 chỉ gói gọn trong vài dòng mã:

```
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);
```

Thao tác ký tạo ra sáu mươi tư byte —được gọi là chữ ký— vốn chỉ có thể được tạo ra từ khóa riêng tương ứng. Việc xác minh là công khai: bất kỳ ai có khóa công khai đều có thể kiểm tra xem chữ ký có khớp với tin nhắn hay không. Không có khóa riêng, không ai có thể tạo ra chữ ký hợp lệ cho tin nhắn đó; có khóa công khai, mọi người đều có thể phát hiện xem chữ ký có hợp lệ hay không. Sự bất đối xứng này là thứ cho phép người ký chứng minh quyền tác giả mà không cần chia sẻ bí mật.

Ví dụ trên là phiên bản hướng dẫn tối thiểu. Trong mã thực tế của Solo2, chuỗi đi qua hai tệp, một tệp bằng JavaScript nằm trong trình duyệt của người dùng và tái tạo entropy từ hai mươi bốn từ, một tệp khác bằng Zig trong thư viện *zcatcrypto* lấy entropy đó và tạo ra các khóa mật mã cụ thể. Bắt đầu từ phía trình duyệt:

```
// 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 };
}
```

Ba mươi hai byte entropy đó, cùng với ba mươi hai byte khác được tạo ra trong cùng một bước, truyền đến mô-đun WebAssembly của Zig để tạo ra các khóa Ed25519 thực sự. Hàm hoàn chỉnh, với việc dọn dẹp bộ nhớ cuối cùng, nằm gọn trong một màn hình:

```
// 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;
}
```

Hai chi tiết đáng chú ý. Thứ nhất: cùng một seed luôn tạo ra cùng một cặp khóa — chính điều này cho phép khôi phục danh tính bằng cách nhập hai mươi bốn từ vào một thiết bị mới. Thứ hai: seed được xóa rõ ràng khỏi bộ nhớ ở dòng cuối cùng. Sau thời điểm đó, ngay cả bản thân hàm cũng không thể tái tạo lại các khóa; lời nói của người dùng sẽ là nguồn duy nhất.

Dành cho những ai muốn kiểm tra bằng số nhỏ. Sơ đồ chữ ký có thể được theo dõi toàn bộ với các con số đủ nhỏ để thực hiện các phép tính bằng tay. Những ai không muốn đi sâu vào số học có thể bỏ qua khối này mà không làm mất mạch của bài viết; những ai muốn xem cơ chế hoạt động từng bước sẽ tìm thấy nó ở đây. **Các quy tắc công khai**, mà bất kỳ ai cũng có thể đọc: một số nguyên tố $p = 23$ (trong Ed25519 thực tế, nó có khoảng bảy mươi bảy chữ số; chúng tôi sử dụng hai mươi ba để các phép tính nằm gọn trong một trang), một cơ số $g = 2$ có bậc trong nhóm này là $q = 11$, và quy ước rằng tất cả các phép toán số học với g được thực hiện *módulo p* và tất cả các số mũ được rút gọn *módulo q*. **Lựa chọn riêng tư**, duy nhất và không bao giờ được chia sẻ: bí mật $x = 6$. Đó chính là danh tính.

Bước 1 — Phần công khai của danh tính. Nó được tính toán một lần và được công bố công khai.

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

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

Phần công khai của danh tính là **18**. Bất kỳ ai cũng có thể lấy nó và sử dụng nó để xác minh các chữ ký được tạo bằng danh tính này. Không ai, chỉ bằng cách quan sát số 18, có thể khôi phục được bí mật 6: đó là bài toán logarit rời rạc mà chúng ta sẽ quay lại ở cuối.

Bước 2 — Ký một tin nhắn. Chủ sở hữu danh tính muốn ký tin nhắn $m = 7$. Anh ta bắt đầu bằng cách chọn một giá trị ngẫu nhiên mới $k = 4$, giá trị này sẽ chỉ được sử dụng một lần và không bao giờ được chia sẻ (trong Ed25519 thực tế, k được tạo ra một cách xác định từ tin nhắn và bí mật để tránh nguy cơ sử dụng lại, nhưng vai trò của nó chính xác là thế này). Sau đó, anh ta tính toán ba con số:

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

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

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

Chữ ký là cặp **(r, s) = (16, 10)**. Nó truyền đi công khai cùng với tin nhắn. Bất kỳ ai cũng có thể đọc nó. Lưu ý về mặt sư phạm: trong Ed25519 thực tế, hàm H là SHA-512, mạnh mẽ về mặt mật mã; ở đây chúng tôi sử dụng phép đơn giản hóa $e = (r + m) \bmod q$ để người đọc có thể thực hiện các bước mà không cần tính toán mã băm (hash). Cấu trúc thuật toán là như nhau.

Bước 3 — Xác minh chữ ký. Người xác minh có phần công khai $y = 18$, tin nhắn $m = 7$ và chữ ký $(r, s) = (16, 10)$. Anh ta tái tạo e theo cùng một cách — $e = (16 + 7) \bmod 11 = 1$ — và kiểm tra xem đẳng thức này có đúng không:

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

Tính toán hai vế riêng biệt:

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

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

Cả hai vế đều cho kết quả **12**. Chữ ký hợp lệ. Bất kỳ ai có phần công khai 18 đều có thể đi đến kết luận này mà không cần biết bí mật là 6.

Còn một bên thứ ba cố gắng giả mạo thì sao? Eva đã thấy mọi thứ công khai đi qua kênh: $p = 23$, $g = 2$, $q = 11$, $y = 18$, $m = 7$, $r = 16$, $s = 10$. Để ký một tin nhắn *khác* dưới danh nghĩa danh tính này, cô ấy sẽ cần biết x. Cách duy nhất của cô ấy là tự hỏi: "với số mũ x nào thì $2^x \bmod 23 = 18$ đúng?". Với $p = 23$, cô ấy có thể thử 0, 1, 2, 3, … và tìm thấy nó trong vài giây. Nhưng khi thay thế 23 bằng một số nguyên tố có kích thước thực tế của Ed25519, không gian của các số mũ có thể vượt quá số lượng nguyên tử trong vũ trụ có thể quan sát được. **Ngày nay không có thuật toán nào được nhân loại biết đến có thể đi qua không gian đó trong ít hơn hàng tỷ năm.** Đó là cùng một bài toán logarit rời rạc làm cơ sở cho Diffie-Hellman trong bài viết trước, được áp dụng ở đây cho sơ đồ chữ ký.

Những gì chúng ta vừa trải qua *chính xác* là Schnorr, sơ đồ chữ ký mà Ed25519 là một biến thể được điều chỉnh cho đường cong elliptic. Trong Ed25519 thực tế, tất cả các phép toán được thực hiện trên các điểm của một đường cong cụ thể (Curve25519) thay vì trên các số nguyên modulo một số nguyên tố, và hàm H là SHA-512 thay vì phép cộng đồ chơi mà chúng tôi đã sử dụng ở trên. Hai sự thay thế là các điều chỉnh triển khai — đạt được khả năng chống mật mã đối với tấn công vét cạn (brute force), đạt được các thuộc tính bảo mật bổ sung cho k —. Cấu trúc thuật toán, ba phép toán, lý do của sự bất đối xứng, là như nhau.

Cần tạm dừng một chút ở đây, vì toàn bộ chuỗi có thể bị nhầm lẫn trong cái nhìn nhanh với một nguyên hàm khác của bộ ba: hash. Không phải vậy. Hash là một hàm duy nhất thực hiện nén — nhiều byte đi vào, một dấu vết ngắn đi ra, con đường kết thúc ở đó. Danh tính mật mã là một cặp toán học bổ sung: bí mật được giữ lại và ký; đối tác công khai của nó được xuất bản và xác minh. Trong khi hash thu gọn thông tin theo một hướng duy nhất, danh tính thiết lập sự bất đối xứng giữa hai nửa. Hash minh chứng cho những gì đã được nói; danh tính minh chứng cho người đã nói điều đó.

Những gì cụm từ không phải là

Cần làm rõ ba quan niệm sai lầm phổ biến. Cụm từ không phải là mật khẩu theo đúng nghĩa: nó không được so sánh với dấu vân tay được lưu trữ trên máy chủ; nó được nhập vào thiết bị của người dùng để tái tạo định danh bằng toán học. Cụm từ không thể khôi phục: nếu bị mất, không có ai để bạn yêu cầu cấp lại; nếu bị sao chép, định danh cũng bị sao chép. Cụm từ không phải là một thông tin xác thực có thể tách rời khỏi định danh: cụm từ *chính là* định danh. Bất kỳ ai có nó đều có thể hành động dưới danh nghĩa định danh đó mà không cần sự cho phép bổ sung, không cần quy trình ủy quyền, không có khả năng khôi phục.

Chính đặc tính thứ ba này làm thay đổi trọng lượng của vấn đề. Một mật khẩu bị mất là một sự phiền toái về mặt hành chính. Một định danh mã hóa bị mất chính là mất đi định danh. Một tờ giấy ghi cụm từ bị bên thứ ba tìm thấy không phải là rủi ro mất tài khoản: đó là việc bàn giao toàn bộ định danh. Lời hứa của hệ thống —rằng không ai có thể thu hồi định danh của bạn hoặc chặn bạn một cách tùy tiện— đi kèm không tách rời với trách nhiệm —rằng bạn là người duy nhất lưu giữ thứ mà không ai có thể khôi phục thay cho bạn.

Lời hứa và trọng lượng

Mô hình định danh mã hóa thường nhận được thuật ngữ *tự chủ* —self-sovereign trong các tài liệu tiếng Anh—. Việc lựa chọn từ ngữ là có chủ ý và mô tả tình trạng này khá chính xác. Người dùng có quyền tự chủ đối với định danh của mình theo nghĩa gần như thời trung cổ: không có vị vua, tổ chức phát hành hay cơ quan trung ương nào ban cấp; cũng không ai trong số đó có thể thu hồi nó. Nhưng cũng giống như vị vua thời trung cổ, người dùng phải chịu toàn bộ hậu quả cho những sai lầm của mình: không có nhiếp chính nào đưa ra quyết định thay cho bạn nếu bạn làm mất con dấu.

Sự lựa chọn giữa định danh do bên thứ ba quản lý và định danh tự chủ không có một câu trả lời đúng duy nhất cho tất cả mọi người. Đối với một tài khoản diễn đàn không quan trọng, định danh được quản lý có lẽ là phù hợp với rủi ro. Đối với một định danh chuyên nghiệp dùng để ký các tài liệu ràng buộc về mặt pháp lý, đối với một

định danh kinh tế bảo vệ tiền tiết kiệm cá nhân, đối với một định danh giao tiếp chuyên nghiệp với những khách hàng đã tin tưởng giao phó thông tin nhạy cảm, vấn đề sẽ khác đi. Khi đó, câu hỏi không còn là «nó có tiện lợi không?» mà trở thành «ai, ngoài tôi, có quyền hành động dưới danh nghĩa là tôi, và trong hoàn cảnh nào?».

Cơ chế này xuất hiện ở đâu trong các hệ thống thực tế

BIP39 ra đời trong thế giới Bitcoin vào năm 2013 và nhanh chóng lan rộng ra toàn bộ hệ sinh thái tiền mã hóa: bất kỳ ví nghiêm túc nào ngày nay đều chấp nhận một cụm từ BIP39 gồm 12 hoặc 24 từ làm bản sao lưu cho danh tính kinh tế của người sở hữu. Ngoài tiền mã hóa, cùng một khái niệm cơ bản — cặp mã hóa chứng minh quyền tác giả mà không cần trung gian — cũng xuất hiện trong các hệ thống khác với cú pháp khác. Các khóa SSH mà quản trị viên hệ thống sử dụng để truy cập máy chủ là một trường hợp kinh điển: một khóa riêng tư mà quản trị viên giữ trên máy của họ và một khóa công khai được sao chép vào mỗi máy chủ; không có thực thể nào tương đương với một dịch vụ tập trung can thiệp vào. Giao thức Signal sử dụng Ed25519 với tài liệu khóa cố định trên thiết bị; tiêu chuẩn eIDAS của Châu Âu, trong phần chữ ký đủ điều kiện, cũng dựa trên cùng một nguyên tắc mã hóa, với sự khác biệt là khóa được lưu giữ bởi một nhà cung cấp dịch vụ tin cậy đủ điều kiện thay vì người dùng.

Solo2, nền tảng xuất bản của ấn phẩm này, sử dụng cụm từ BIP39 gồm 24 từ làm danh tính cho mỗi người dùng. Người dùng khi tạo tài khoản sẽ thấy các từ này một lần duy nhất. Chúng không được lưu trữ trên bất kỳ máy chủ nào của Solo2 hay của bất kỳ ai khác: nếu người dùng ghi chú lại và bảo quản chúng, họ sẽ giữ được danh tính của mình mãi mãi. Nếu họ làm mất, họ sẽ mất tất cả. Đây là hệ quả nhất quán với kiến trúc không có nhà vận hành trung gian: nếu Solo2 có thể trả lại danh tính cho người dùng đã làm mất, thì Solo2 cũng có thể đưa nó cho bất kỳ ai gây áp lực lên Solo2 để lấy nó.

Dành cho độc giả chuyên nghiệp

Bốn cân nhắc cho những ai đang đánh giá việc áp dụng danh tính mã hóa tự chủ (autosoberana) trong bối cảnh chuyên nghiệp:

1. Cụm từ chính là danh tính. Việc bảo quản vật lý — giấy, nhiều bản sao ở những nơi khác nhau, hoặc cuối cùng là kim loại được khắc để sử dụng lâu dài — mang lại nhiều đảm bảo hơn so với bảo quản kỹ thuật số, vốn làm tăng bề mặt tấn công mà không giảm rủi ro mất mát.
2. Không có sự phục hồi. Việc thiết kế quy trình với giả định rằng một ngày nào đó bản sao chính sẽ bị mất sẽ khôn ngoan hơn nhiều so với việc phát hiện ra điều đó vào ngày nó bị mất. Một bản sao thứ hai được tách biệt về mặt địa lý sẽ giải quyết được hầu hết các tình huống.
3. It không giống với chứng thư đủ điều kiện eIDAS. Đối với chữ ký đủ điều kiện trong Liên minh — các văn bản công chứng, một số thủ tục với cơ quan hành chính — luật pháp yêu cầu một nhà cung cấp đủ điều kiện lưu giữ khóa. Danh tính mã hóa tự chủ (autosoberana) phục vụ cho giao tiếp chuyên nghiệp và ký kết tài liệu có giá trị chứng minh, nhưng không tự động thay thế chứng thư đủ điều kiện trong các trường hợp mà quy chuẩn yêu cầu.
4. Nếu danh tính chuẩn bị được chuyển giao — thừa kế, kế vị chuyên nghiệp, chấm dứt hoạt động — thì nên chuẩn bị quy trình trước, không phải sau. Các quy trình chính thức với phong bì được niêm phong bằng sáp (lacre), hướng dẫn cho người thi hành di chúc, ký gửi tại văn phòng công chứng, là những sắp xếp kinh điển hoàn toàn tương thích với bản chất mã hóa của tài sản.

Bài viết này khép lại bộ ba khái niệm đã mở đầu chu kỳ — hash, mã hóa, danh tính —. Ba ý tưởng này được xây dựng bồi đắp lên nhau: hash cung cấp dấu vân tay không thể thay đổi, mã hóa cung cấp tính bảo mật mà không cần bên thứ ba đáng tin cậy, danh tính cung cấp quyền tác giả mà không cần bên thứ ba cấp phép. Cả ba đều chia sẻ một đặc tính cũng không mang tính tư tưởng: chúng chuyển giao các khả năng kỹ thuật vốn truyền thống thuộc về nhà vận hành, từ người quản lý dịch vụ sang người sử dụng dịch vụ. Chúng cũng chuyển giao cả trách nhiệm theo cùng. Việc nói chuyện một cách trung thực về bất kỳ điều nào trong ba điều này đòi hỏi chúng ta cũng phải nói về hai điều còn lại.

Nguồn tham khảo và đọc thêm

- Palatinus, M.; Rusnak, P.; Voisine, A.; Bowe, S. — *BIP-0039: Mnemonic code for generating deterministic keys*, đề xuất cải tiến Bitcoin năm 2013. Tiêu chuẩn thực tế cho các cụm từ phục hồi trong ngành công nghiệp tiền mã hóa.
- RFC 8032 — Edwards-Curve Digital Signature Algorithm (EdDSA), bao gồm Ed25519. IETF, tháng 1 năm 2017. Đặc tả quy chuẩn của sơ đồ chữ ký được sử dụng trong phần lớn ngành công nghiệp đương đại.
- RFC 2898 — PKCS #5: Password-Based Cryptography Specification, phiên bản 2.0. IETF, tháng 9 năm 2000. Xác định thuật toán PBKDF2 được sử dụng trong dẫn xuất BIP39 từ cụm từ sang seed.
- Quy định (EU) 910/2014 (eIDAS) và sự phát triển của nó theo Quy định (EU) 2024/1183 (eIDAS 2) — khung pháp lý Châu Âu về danh tính điện tử và chữ ký đủ điều kiện. Một chế độ khác với danh tính tự chủ, nhưng về mặt khái niệm được hỗ trợ bởi cùng các nguyên hàm mã hóa.
- Allen, C. — *The Path to Self-Sovereign Identity* (2016). Văn bản kinh điển về các nguyên tắc và cam kết của mô hình tự chủ, tuy ra đời trước nhưng vẫn phù hợp để hiểu về nhóm các giải pháp đương đại.

← TrướcMô hình kinh doanh như một tín hiệu của sự tin cậyTiếp theo →Self-hosting như một thực hành chuyên nghiệp

Các bài đọc gần đây

- Suy ngẫm · 29 tháng 6, 2026 Bạn không ẩn danh
- Suy ngẫm · 27 tháng 5, 2026 Lo que una firma no puede arreglar
- Phân tích · 26 tháng 5, 2026 Quyền riêng tư thực sự vs biểu hiện: những câu hỏi bạn nên tự đặt ra

Tải bài viết này để sử dụng ở bất cứ đâu bạn cần.

↓ Markdown ↓ Văn bản thuần túy ↓ PDF

Tệp sẽ được tải xuống thiết bị của bạn. Từ đó, bạn có thể lưu trữ, nhập vào Solo2 hoặc chia sẻ ở bất cứ đâu. Cuadernos không quyết định điểm đến thay cho bạn.

Dấu sáp · SHA-256 c687a2b00646bbbc78950c040bc325a4dbe649efc43b8fe3e2029f5f89fd1d1e