24 Kelime: Kriptografik Kimlik Nedir?

Kriptografik bir kimlik şifre değildir: hiçbir sunucu onu saklamaz ve geri kurtarılamaz. BIP39 mekanizmasının didaktik bir açıklaması, neden tam olarak yirmi dört kelime ve onlara sahip olanın üzerine binen gerçek yük nedir?

Birbirimizi anlamak için: Gmail şifrenizi unutursanız, Google sizin için onu sıfırlar. Kriptografik bir kimliği oluşturan 24 kelimeyi kaybederseniz, onları isteyecek kimse yoktur. Prosedür katı olduğu için değil — diğer uçta kimse olmadığı için. Bu fark, her şeydir.

Şifre ve Kimlik Arasındaki Fark

Klasik internet modelinde şifre, kullanıcının kimliği değildir. Bu bir belgedir. Kullanıcının bir kimliği vardır — bir isim, bir e-posta, bir müşteri numarası — ve bir sunucuya iddia ettiği kişi olduğunu kanıtlamak için, sunucunun sakladığı bir izle karşılaştırdığı bir şifre sunar. İzler eşleşirse, sunucu oturuma izin verir. Şifre kaybolursa, kullanıcı aynı kullanıcı olarak kalır; kaybettiği şey belgedir ve onu eski haline getirmek için bir kurtarma prosedürü — kayıtlı adrese bir e-posta, bir güvenlik sorusu — mevcuttur.

Kriptografik kimlik başka bir şekilde çalışır. Birinin kayıtlı bir izle karşılaştırdığı bir kimlik belgesi değildir; *kendisi* başına eksiksiz bir matematiksel sırdır. Nerede olduğu fark etmez — bir kağıtta, bir cihazda, hatta yabancı bir sunucuda —: kimlik, onu doğrulayan kişi sayesinde değil, matematiği sayesinde vardır. Burada «SHA-256 Gerçekte Nedir?» bölümünde gördüğümüze benzer bir özellik ortaya çıkıyor: sahiplik, sırrı sergileyerek değil, onu imzalamak için kullanarak kanıtlanır. Bu şekilde üretilen imza, sırrın kendisini bilmeye gerek kalmadan ve kontrolde üçüncü bir tarafın aracılığına ihtiyaç duymadan, sırrın kendisinden matematiksel olarak türetilen genel bir değerle herkes tarafından kontrol edilebilir. Sırra sahip olan kimliktir; onu kaybeden artık kimlik olmaktan çıkar. Karar kesindir: **kimliği size iade etmesini isteyeceğiniz kimse yoktur. O kişi mevcut değildir, çünkü zaten en başta ona sahip değildi.**

Yirmi Dört Kelime Neyi Temsil Eder?

Kriptografik kimlik genellikle otuz iki baytlık — iki yüz elli altı bitlik — matematiksel bir sırla temsil edilir. Hatırlanması zor ve hatasız bir şekilde yazıya dökülmesi daha da zor olan bir sayı. Kripto endüstrisi bu sorunu 2013 yılında BIP39 adlı küçük ve zarif bir standartla çözdü: bu iki yüz elli altı biti, iki bin kırk sekiz kelimelik resmi bir listeden alınan yirmi dört kelimelik bir dizi olarak temsil etme yolu. Arkasındaki aritmetik zarafetle uyuşuyor; bunu ayrıntılı olarak görmek isteyenler yan notta bulabilirler.

Hesaplama sondan başlar. Sırrın iki yüz elli altı bitini sekiz bitlik sağlama toplamı (checksum) ekleyerek temsil etmek istiyoruz: toplamda iki yüz altmış dört bit. Bunları yirmi dört kelimeye bölersek — kayıp olmadan not etmek ve dikte etmek için yönetilebilir bir sayı — her kelime tam olarak on bir bitlik bilgi sağlamalıdır. Ve on bir bit, iki üzeri on bir olasılıktır, yani iki bin kırk sekiz. Bu nedenle resmi BIP39 kelime listesi tam olarak bu boyuttadır: liste soruna göre mevcuttur, tersi değil.

Hesaplama dekoratif değildir. Birisi yirmi üç kelimeyi doğru bir şekilde yazıya döker ve yirmi dördüncüde hata yaparsa, sağlama toplamı bunu tespit edecektir: yazılım ona "bu dizi geçerli değil" diyecektir. Birisi yirmi

dördün de doğru bir şekilde yazıya dökerse, yazılım belirsizliğe yer bırakmadan aynı kimliği türetecektir. Kelime listesinin seçimi de bilinçlidir: BIP39 kelime listesindeki kelimeler kısadır, birbirinden farklıdır, aksan işareti içermezler, fonetik ve yazım karışıklıklarını en aza indirmek için seçilmişlerdir. İnsanlar tarafından kayıp olmadan hatırlanmak, yazılmak ve dikte edilmek üzere tasarlanmış bir kelime listesidir.

İfadeden anahtara

O yirmi dört kelime, mesajları imzalayan kriptografik anahtar değildir. Bunlar, PBKDF2 adı verilen deterministik bir süreç aracılığıyla altmış dört baytlık bir tohuma (seed) dönüştürülen orijinal entropinin geri kazanılabilir bir temsilidir. Bu tohumdan, yine deterministik bir şekilde, kullanıcının kullandığı somut kriptografik anahtarlar türetilir: imzalamak için bir özel anahtar ve imzaları doğrulamak için yayınlanan karşılık gelen bir genel anahtar. Farklı sistemlerde aynı mekanizma: kripto para birimleri secp256k1 eğrisini kullanır; Signal protokolü ve birçok modern sistem Curve25519 eğrisi üzerinde Ed25519 kullanır. Ed25519 gibi somut bir eğri için BIP32 ve SLIP-0010 standartları bu altmış dört baytlık tohumu alır ve deterministik olarak etkin imza anahtarını oluşturan otuz iki baytı türetir — bir sonraki bölümdeki kod örneğinin başladığı otuz iki baytın aynısı.

Bu, tüm endüstrinin mekanizmayı kullanıcıya sunma standart yoludur —kripto para cüzdanları, merkezi olmayan kimlik yöneticileri, kalıcı kimlik kısmında Signal, aralarında Solo2—: kullanıcı pratikte tohumu veya türetilmiş anahtarları asla görmez. Kimliğini oluştururken yirmi dört kelimeyi görür ve isteğe bağlı olarak bunları bir kağıda not eder. Kelimeler daha sonra kimliği taşımak istediğinde cihazları arasında seyahat eder: bunları yeni uygulamaya girer, uygulama aynı tohumu, aynı anahtarları, aynı kimliği türetir. Taşınabilir, kriptografik olarak sağlam ve makul sınırlar içinde hatırlanabilir bir mekanizmadır.

Anahtar ile nasıl imzalanır (Zig dokunuşu)

Zig'de, yirmi dört kelimeden türetilen otuz iki baytlık tohuma sahip olduğunuzda, Ed25519 ile bir mesajı imzalamak birkaç satıra sığar:

```
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);
```

İmzalama işlemi, yalnızca karşılık gelen özel anahtardan oluşturulabilen altmış dört bayt —imza olarak adlandırılır— üretir. Doğrulama halka açıktır: genel anahtara sahip olan herkes imzanın mesaja karşılık gelip gelmediğini kontrol edebilir. Özel anahtar olmadan hiç kimse o mesaj için geçerli bir imza oluşturamaz; genel anahtar ile herkes bir imzanın geçerli olup olmadığını tespit edebilir. Bu asimetri, imzalayanın sırrı paylaşmadan yazarlığını kanıtlamasına olanak tanıyan şeydir.

Önceki örnek kılavuzun minimum versiyonudur. Solo2'nin gerçek kodunda zincir iki dosyadan geçer: Biri kullanıcının tarayıcısında yaşayan ve yirmi dört kelimeden entropiyi yeniden oluşturan JavaScript dosyası, diğeri ise bu entropiyi alıp somut kriptografik anahtarları türeten *zcatcrypto* kütüphanesindeki Zig dosyasıdır. Tarayıcı tarafından başlayarak:

```
// 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 };
}
```

Bu otuz iki baytlık entropi, aynı adımda türetilen diğer otuz iki bayt ile birlikte, asıl Ed25519 anahtarlarını üreten Zig'in WebAssembly modülüne gider. Nihai bellek temizliği ile birlikte tüm fonksiyon tek bir ekrana sığar:

```
// 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;
}
```

İki ayrıntıya dikkat çekmekte fayda var. Birincisi: aynı tohum (seed) her zaman aynı anahtar çiftini üretir — işte tam da bu, yirmi dört kelimeyi yeni bir cihaza girerek kimliğin geri kazanılmasını sağlar. İkincisi: tohum, son satırda bellekten açıkça silinir. Bu noktadan sonra, fonksiyonun kendisi bile anahtarları yeniden oluşturamaz; kullanıcının kelimeleri tek kaynak olacaktır.

Küçük sayılarla kontrol etmek isteyenler için. İmza şeması, hesaplamaları elle yapmaya yetecek kadar küçük rakamlarla baştan sona takip edilebilir. Aritmetiğe girmemeyi tercih edenler makalenin akışını bozmadan bu bloğu atlayabilirler; mekanizmanın adım adım nasıl çalıştığını görmek isteyenler burada bulabilirler. **Herkesin okuyabileceği genel kurallar**: bir asal sayı $p = 23$ (gerçek Ed25519'da bu yaklaşık yetmiş yedi basamaklıdır; hesaplamaların bir sayfaya sığması için yirmi üç kullanıyoruz), bu gruptaki derecesi $q = 11$ olan bir taban $g = 2$ ve g ile yapılan tüm aritmetik işlemlerin *módulo p* yapıldığı ve tüm üslerin *módulo q* indirgendiği kuralıdır. **Özel seçim**, tek bir tane ve asla paylaşılmayan: sır $x = 6$. Kimlik budur.

Adım 1 — Kimliğin genel kısmı. Bir kez hesaplanır ve açıkça yayınlanır.

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

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

Kimliğin genel kısmı **18**'dir. Herkes bunu alabilir ve bu kimlikle atılan imzaları doğrulamak için kullanabilir. Sadece 18'i gözlemleyerek hiç kimse sır 6'yı geri kazanamaz: İşte sonunda tekrar döneceğimiz ayrık logaritma problemi budur.

Adım 2 — Bir mesajı imzalamak. Kimlik sahibi $m = 7$ mesajını imzalamak istiyor. Yalnızca bir kez kullanılacak ve asla paylaşılmayacak olan yeni bir rastgele değer $k = 4$ seçerek başlar (gerçek Ed25519'da k, yeniden kullanım tehlikesini önlemek için mesajdan ve sırdan deterministik olarak türetilir, ancak oynadığı rol tam olarak budur). Daha sonra üç sayı hesaplar:

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

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

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

İmza, **(r, s) = (16, 10)** çiftidir. Mesajla birlikte açıkça taşınır. Herkes okuyabilir. Didaktik not: gerçek Ed25519'da H fonksiyonu kriptografik olarak sağlam olan SHA-512'dir; burada okuyucunun bir özet (hash) hesaplamasına gerek kalmadan adımları takip edebilmesi için $e = (r + m) \bmod q$ basitleştirmesini kullanıyoruz. Algoritma yapısı aynıdır.

Adım 3 — İmzayı doğrulamak. Doğrulayıcıda genel kısım $y = 18$, mesaj $m = 7$ ve imza $(r, s) = (16, 10)$ bulunur. e'yi aynı şekilde yeniden oluşturur — $e = (16 + 7) \bmod 11 = 1$ — ve bu eşitliğin sağlanıp sağlanmadığını kontrol eder:

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

İki tarafı ayrı ayrı hesaplar:

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

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

Her iki taraf da **12** sonucunu verir. İmza geçerlidir. Genel kısım 18'e sahip olan herkes, sırrın 6 olduğunu hiç bilmeden bu sonuca varabilir.

Ya sahtecilik yapmaya çalışan üçüncü bir taraf? Eva, kanaldan geçen genel olan her şeyi gördü: $p = 23$, $g = 2$, $q = 11$, $y = 18$, $m = 7$, $r = 16$, $s = 10$. Bu kimlik adına *farklı* bir mesaj imzalamak için x'i bilmesi gerekirdi. Tek

yolu kendine şu soruyu sormaktır: "Hangi x üssü için $2^x \bmod 23 = 18$ sağlanır?". $p = 23$ ile 0, 1, 2, 3, … deneyerek bunu saniyeler içinde bulabilir. Ancak 23 yerine Ed25519'un gerçek boyutlarındaki bir asal sayıyı koyduğunuzda, olası üslerin alanı gözlemlenebilir evrendeki atom sayısını aşar. **Bugün insanlık tarafından bilinen ve bu alanı milyarlarca yıldan daha kısa sürede tarayabilecek hiçbir algoritma yoktur.** Bu, bir önceki makaledeki Diffie-Hellman'ın temelini oluşturan ve burada imza şemasına uygulanan aynı ayrık logaritma problemidir.

Şu an üzerinden geçtiğimiz şey *tam olarak* Schnorr'dur; Ed25519, bunun eliptik bir eğriye uyarlanmış bir varyantıdır. Gerçek Ed25519'da tüm işlemler, bir asal sayı modülündeki tam sayılar yerine belirli bir eğrinin (Curve25519) noktaları üzerinde yapılır ve H fonksiyonu yukarıda kullandığımız oyuncak toplama yerine SHA-512'dir. İki ikame, uygulama ayarlamalarıdır — kaba kuvvete karşı kriptografik direnç kazanmak, k için ek güvenlik özellikleri kazanmak —. Algoritmik yapı, üç işlem ve asimetrinin nedeni aynıdır.

Burada kısa bir ara vermekte fayda var, çünkü tüm zincir hızlı bir bakışla üçlünün diğer ilkel birimi olan özet (hash) ile karıştırılabilir. Değildir. Özet, sıkıştıran benzersiz bir fonksiyondur — birçok bayt girer, kısa bir ayak izi çıkar, yol orada biter. Kriptografik kimlik, birbirini tamamlayan matematiksel bir çifttir: Sır kalır ve imzalar; genel karşılığı ise yayınlanır ve doğrular. Özetin bilgiyi tek bir yönde çökerttiği yerde, kimlik iki yarı arasında bir asimetri kurar. Özet neyin söylendiğine tanıklık eder; kimlik ise kimin söylediğine tanıklık eder.

İfadenin ne olmadığı

Sıkça yapılan üç yanlış anlaşılmayı gidermek yerinde olacaktır. İfade, tam anlamıyla bir şifre değildir: bir sunucuda saklanan parmak iziyle karşılaştırılmaz; kimliği matematiksel olarak yeniden oluşturmak için kullanıcının cihazına girilir. İfade geri kazanılamaz: kaybolursa, isteyebileceğiniz kimse yoktur; kopyalanırsa, kimlik de kopyalanmış olur. İfade, kimlikten ayrılabilir bir kimlik bilgisi değildir: ifade kimliğin *kendisidir*. Ona sahip olan, ek izin olmadan, yetkilendirme süreci olmadan, kurtarma olasılığı olmadan o kimlik olarak hareket edebilir.

Konunun ağırlığını değiştiren bu üçüncü özelliktir. Kaybedilen bir şifre idari bir sıkıntıdır. Kaybedilen bir kriptografik kimlik, kimliğin kendisidir. Üçüncü şahıslar tarafından bulunan ifadeli bir kağıt, hesap çalınma riski değildir: tüm kimliğin teslim edilmesidir. Sistemin vaadi —kimsenin kimliğinizi iptal edememesi veya sizi keyfi olarak engelleyememesi—, sorumlulukla ayrılmaz bir şekilde gelir —kimsenin sizin yerinize geri getiremeyeceği bir şeyin tek koruyucusu olduğunuz sorumluluğu.

Vaat ve ağırlık

Kriptografik kimlik modeli genellikle *öz-egemen* (İngilizce literatürde self-sovereign) sıfatıyla anılır. Kelime seçimi bilinçlidir ve durumu oldukça doğru bir şekilde tanımlar. Kullanıcı, neredeyse ortaçağa ait bir anlamda kimliği üzerinde egemendir: onu hiçbir kral, hiçbir ihraççı, hiçbir merkezi otorite vermez; yukarıdakilerin hiçbiri onu geri de alamaz. Ancak ortaçağ hükümdarı gibi kullanıcı da hatalarının tüm sonuçlarına katlanır: mührünü kaybederse onun yerine karar verecek bir naip yoktur.

Üçüncü bir tarafça yönetilen kimlik ile öz-egemen kimlik arasındaki seçimde evrensel olarak tek bir doğru cevap yoktur. Önemsiz bir forum hesabı için yönetilen kimlik muhtemelen riskle orantılıdır. Ancak yasal olarak bağlayıcı belgeleri imzalayan profesyonel bir kimlik için, kişisel tasarrufları koruyan ekonomik bir kimlik için, hassas bilgiler emanet etmiş müşterilerle kurulan profesyonel iletişim kimliği için konu değişir. Orada soru «konforlu mu?» olmaktan çıkar ve «benim dışımda kimin benim gibi hareket etme gücü var ve hangi koşullar altında?» haline gelir.

Bu mekanizma gerçek sistemlerde nerede karşımıza çıkıyor?

BIP39, 2013 yılında Bitcoin dünyasında doğdu ve hızla tüm kripto para ekosistemine yayıldı: bugün her ciddi cüzdan, sahibinin ekonomik kimliğinin yedeği olarak on iki veya yirmi dört kelimelik bir BIP39 ifadesini kabul etmektedir. Kripto paraların dışında, aynı temel kavram — bir aracı olmaksızın yazarlığı kanıtlayan kriptografik çift — farklı sözdizimine sahip diğer sistemlerde de karşımıza çıkmaktadır. Bir sistem yöneticisinin sunucularına erişmek için kullandığı SSH anahtarları klasik bir durumdur: yöneticinin kendi makinesinde tuttuğu bir özel anahtar ve her sunucuya kopyalanan bir genel anahtar; merkezi bir hizmetle karşılaştırılabilecek hiçbir yapı müdahale etmez. Signal protokolü, cihaz üzerinde kalıcı anahtar materyali ile Ed25519 kullanır; Avrupa eIDAS standartları, nitelikli imza kısmında, anahtarın kullanıcı yerine nitelikli bir güven hizmeti sağlayıcısı tarafından saklanması farkıyla, aynı kriptografik ilkeye dayanır.

Bu yayının yayıncı platformu olan Solo2, her kullanıcının kimliği olarak yirmi dört kelimelik bir BIP39 ifadesi kullanır. Kullanıcı, hesabını oluştururken kelimeleri bir kez görür. Bunlar Solo2'nin veya başka birinin sunucusunda saklanmaz: kullanıcı bunları not eder ve saklarsa kimliğini sonsuza dek korur. Kaybederse, kaybeder. Bu, arada bir operatörün bulunmadığı mimarinin tutarlı bir sonucudur: Solo2 kimliğini kaybeden kullanıcıya kimliğini geri verebilseydi, Solo2'ye baskı yapan herhangi birine de verebilirdi.

Profesyonel okuyucu için

Profesyonel bir bağlamda kriptografik öz-egemen (autosoberana) kimliği benimsemeyi değerlendirenler için dört mülahaza:

1. İfade kimliğin kendisidir. Fiziksel muhafaza — kağıt, farklı yerlerde birkaç kopya, nihayetinde uzun süreli kullanım için kazınmış metal — kayıp riskini azaltmadan saldırı yüzeyini artıran dijital muhafazadan daha fazla garanti sunar.
2. Kurtarma yoktur. Süreci, bir gün birincil kopyanın kaybolacağı varsayımıyla tasarlamak, bunu kaybolduğu gün keşfetmekten çok daha mantıklıdır. Coğrafi olarak ayrılmış ikinci bir kopya neredeyse tüm senaryoları çözer.
3. Bu, eIDAS nitelikli sertifikası ile aynı şey değildir. Birlik'teki nitelikli imza için — noter senetleri, idare ile yapılan belirli işlemler — mevzuat, anahtarı saklayan nitelikli bir sağlayıcı gerektirir. Kriptografik öz-egemen kimlik, profesyonel iletişim ve kanıt değeri olan belge imzalama için hizmet eder, ancak normun gerektirdiği durumlarda nitelikli sertifikanın yerini otomatik olarak almaz.
4. Kimlik devredilecekse — miras, mesleki halefiyet, faaliyetin durdurulması — prosedürü sonradan değil, önceden hazırlamakta fayda vardır. Mühür mumu (lacre) ile mühürlenmiş zarflar içeren resmi prosedürler, vasiyeti yerine getirene verilen talimatlar, noter ofisine emanet etme, varlığın kriptografik doğasıyla mükemmel şekilde uyumlu klasik düzenlemelerdir.

Bu makale, döngüyü açan kavramsal üçlüyü kapatıyor — hash, şifreleme, kimlik —. Bu üç fikir birbirinin üzerine inşa edilmiştir: hash değiştirilemez parmak izini verir, şifreleme güvenilir bir üçüncü taraf olmaksızın gizliliği verir, kimlik ise veren bir üçüncü taraf olmaksızın yazarlığı verir. Üçü de ideolojik olmayan bir özelliği paylaşır: Teknik yetenekleri, geleneksel olarak operatörde bulunan hizmeti yönetenden hizmeti kullanana aktarırlar. Sorumlulukları da onlarla birlikte aktarırlar. Bu üçünden herhangi biri hakkında dürüstçe konuşmak, diğer ikisi hakkında da konuşmayı gerektirir.

Kaynaklar ve ek okumalar

- Palatinus, M.; Rusnak, P.; Voisine, A.; Bowe, S. — *BIP-0039: Mnemonic code for generating deterministic keys*, 2013 tarihli Bitcoin iyileştirme önerisi. Kripto endüstrisinde kurtarma ifadeleri için fiili standart.
- RFC 8032 — Edwards-Curve Digital Signature Algorithm (EdDSA), Ed25519 dahil. IETF, Ocak 2017. Günümüz endüstrisinin büyük bir kısmında kullanılan imza şemasının normatif spesifikasyonu.
- RFC 2898 — PKCS #5: Password-Based Cryptography Specification, sürüm 2.0. IETF, Eylül 2000. İfadeden tohum (seed) türetmede kullanılan PBKDF2 algoritmasını tanımlar.

- 910/2014 sayılı Tüzük (AB) (eIDAS) ve 2024/1183 sayılı Tüzük (AB) (eIDAS 2) ile gelişimi — elektronik kimlik ve nitelikli imza için Avrupa çerçevesi. Öz-egemen olandan farklı bir rejim, ancak kavramsal olarak aynı kriptografik ilkelere dayanmaktadır.
- Allen, C. — *The Path to Self-Sovereign Identity* (2016). Öz-egemen modelin ilkeleri ve taahhütleri üzerine temel metin, daha eski olmasına rağmen günümüz çözüm ailesini anlamak için hala geçerlidir.

← ÖncekiBir güven sinyali olarak iş modeliSonraki →Profesyonel bir uygulama olarak self-hosting

Son okumalar

- Düşünce · 29 Haziran 2026 Anonim değilsin
- Düşünce · 27 Mayıs 2026 Bir imzanın düzeltemeyeceği şeyler
- Analiz · 26 Mayıs 2026 Gerçek vs. görünür gizlilik: kendinize sormanız gereken sorular

Bu makaleyi ihtiyacınız olan her yere yanınızda götürün.

↓ Markdown ↓ Düz metin ↓ PDF

Dosya cihazınıza indirilecektir. Oradan kaydedebilir, Solo2'ye aktarabilir veya istediğiniz yerde paylaşabilirsiniz. Cuadernos hedef noktaya sizin yerinize karar vermez.

Mühür mumu · SHA-256 ad62255cc5f1b3c8a42d61dab8b7499d5d08b500cdb98e268ec227cf9500487b

Cuadernos Lacre · Menzuri Gestión S.L. yayını ·
R.Eugenio tarafından yazıldı · Solo2 ekibi tarafından düzenlendi.