एंड-टू-एंड एन्क्रिप्शन, वास्तव में समझाया गया

जब प्रदाता E2EE कहते हैं तो वे क्या कहते हैं, और क्या नहीं कहते। विज्ञापन के बिना तंत्र और इसकी सीमाओं का एक शैक्षिक स्पष्टीकरण।

आइए स्पष्ट करें: WhatsApp कहता है कि आपके संदेश एंड-टू-एंड एन्क्रिप्टेड हैं। यह सच है — और यह पर्याप्त नहीं है। यदि बैकअप बिना अतिरिक्त एन्क्रिप्शन के iCloud या Google Drive पर जाता है, तो एन्क्रिप्शन आपके अपने फोन पर टूट जाता है। परिचालन संबंधी प्रश्न यह नहीं है कि यह एन्क्रिप्टेड है या नहीं, बल्कि यह है कि कुंजियाँ कहाँ रहती हैं।

एन्क्रिप्शन का वास्तव में क्या अर्थ है

किसी संदेश को एन्क्रिप्ट करने का अर्थ है उसे किसी ऐसे व्यक्ति के लिए शोर जैसा लगने वाली चीज़ में बदलना जिसके पास कुंजी नामक एक निश्चित जानकारी नहीं है। ऑपरेशन भेजने वाले के डिवाइस पर किया जाता है और सही कुंजी के साथ, प्राप्त करने वाले के डिवाइस पर इसे पूर्ववत किया जाता है। बीच में, संदेश बिना किसी स्पष्ट अर्थ के बाइट्स के उत्तराधिकार के रूप में यात्रा करता है। यह सरल विचार है। लेख का शेष भाग उन बारीकियों से संबंधित है जो इसे मामले के आधार पर एक वास्तविक गारंटी या मार्केटिंग लेबल में बदल देती हैं।

एंड-टू-एंड विशेषण — अंग्रेजी में *end-to-end*, संक्षिप्त रूप में E2EE — एक सटीकता जोड़ता है। एन्क्रिप्शन इसलिए नहीं किया जाता है कि एक मध्यवर्ती सर्वर इसे पढ़ सके और वितरित कर सके। यह इसलिए किया जाता है ताकि केवल दो सिरों — भेजने वाले के डिवाइस और प्राप्त करने वाले के डिवाइस — के पास कुंजी हो। कोई भी सर्वर जिसके माध्यम से संदेश गुजरता है, वह शोर देखता है, संदेश नहीं। यह *पारगमन में* (in transit) एन्क्रिप्शन के साथ तकनीकी अंतर है, जहां सामग्री एक सर्वर से दूसरे सर्वर पर एन्क्रिप्टेड यात्रा करती है, लेकिन प्रत्येक सर्वर जिससे वह गुजरती है, उसे आगे भेजने के लिए डिक्रिप्ट करता है, अस्थायी रूप से स्पष्ट पाठ को पुनर्प्राप्त करता है।

साझा रहस्य का विरोधाभास

एक स्पष्ट समस्या है। दो लोगों के बीच संदेशों को एन्क्रिप्ट और डिक्रिप्ट करने में सक्षम होने के लिए, दोनों को एक ही कुंजी की आवश्यकता होती है। लेकिन, वे इस कुंजी पर कैसे सहमत होते हैं यदि वे एक-दूसरे को जो कुछ भी भेजते हैं, परिभाषा के अनुसार, एक ऐसे चैनल से गुजरता है जहाँ कोई सुन रहा हो सकता है? उसी चैनल में कुंजी पर सहमत होना जहां वे बाद में इसका उपयोग करेंगे, असंभव लगता है: यदि हमलावर इस पर सहमत होते समय इसे सुन लेता है, तो वे बाद की हर चीज को डिक्रिप्ट करने में सक्षम होंगे। दशकों तक, शास्त्रीय क्रिप्टोग्राफी ने इसे कठिन तरीके से हल किया: चाबियाँ व्यक्तिगत रूप से, उनका उपयोग शुरू करने से पहले, भौतिक मुठभेड़ों में वितरित की गई थीं। राजदूत अपने कोट के अस्तर में सिली हुई चाबियों के ब्रीफकेस ले जाते थे।

समकालीन ईमेल में, वह समाधान स्केल नहीं करता है। यदि हमें प्रत्येक व्यक्ति के घर भौतिक रूप से जाना पड़ता जिससे हम एन्क्रिप्टेड संवाद करने का इरादा रखते थे, तो हम किसी से बात नहीं कर पाते। क्रिप्टोग्राफिक समुदाय द्वारा पचास साल पहले पूछा गया प्रश्न यह था: क्या यह संभव है कि दो लोग जो एक-दूसरे को नहीं जानते हैं और जो केवल एक सार्वजनिक चैनल साझा करते हैं, उसी सार्वजनिक चैनल में एक ऐसे रहस्य पर सहमत हों जिसे चैनल सुनने वाला कोई भी व्यक्ति नहीं जान सके?

Diffie-Hellman की सुंदरता

1976 में, Whitfield Diffie और Martin Hellman नामक दो गणितज्ञों ने कुछ ऐसा प्रदर्शित किया जो स्पष्ट रूप से असंभव था: कि दो लोग, केवल एक सार्वजनिक चैनल के माध्यम से बात करते हुए — एक ऐसा चैनल जहां कोई भी वह सब कुछ सुन सकता है जो वे कहते हैं — किसी भी श्रोता द्वारा खोजे जाने में सक्षम हुए बिना एक गुप्त पासवर्ड पर सहमत हो सकते हैं। यह जादू जैसा लगता है। यह नहीं है: यह गणित है। Diffie-Hellman कुंजी विनिमय, जैसा कि तब से जाना जाता है, इंटरनेट के लगभग सभी एन्क्रिप्टेड संचार का आधार है, और आधी सदी के गहन उपयोग और वैश्विक शैक्षणिक जांच इसकी दृढ़ता की पुष्टि करते हैं। जो कोई भी दृश्य अंतर्ज्ञान या गणित देखना चाहता है वह पढ़ना जारी रख सकता है। जो कोई यह भरोसा करना पसंद करता है कि यह काम करता है वह भी लेख के सूत्र को खोए बिना जारी रख सकता है।

जो कोई भी इसे एक छवि में समझना चाहता है, उसके लिए रंगों के साथ एक ज्ञात सादृश्य है। कल्पना कीजिए कि एलिस और ब्रूनो उन्हें सुनने वाली ईवा के सामने सार्वजनिक रूप से एक आधार रंग — मान लीजिए पीला — पर सहमत होते हैं। प्रत्येक निजी तौर पर एक दूसरा गुप्त रंग चुनता है और अपने रहस्य को पीले रंग के साथ मिलाता है। एलिस को एक विशेष नारंगी मिलता है; ब्रूनो को एक विशेष हरा मिलता है। वे ईवा के सामने परिणामों का आदान-प्रदान करते हैं। अब प्रत्येक प्राप्त रंग को अपने स्वयं के रहस्य के साथ मिलाता है, और दोनों एक ही अंतिम रंग पर पहुँचते हैं, क्योंकि मिश्रण के क्रम से कोई फर्क नहीं पड़ता। ईवा ने पीला और दो मध्यवर्ती मिश्रण देखे हैं, लेकिन रहस्य नहीं; बिना किसी रहस्य के वह अंतिम रंग तक नहीं पहुँच सकती। वास्तविक गणित रंगों को मॉड्यूलर समूहों या अण्डाकार वक्रों में घातांक के लिए बदल देता है, लेकिन विचार वही है: साझा रहस्य सार्वजनिक रूप से बनाया जाता है बिना चैनल में किसी के इसे पुनर्निर्माण करने में सक्षम हुए।

अंकगणित में, जो कोई तंत्र देखना पसंद करता है उसके लिए: एलिस एक गुप्त संख्या a चुनती है, ब्रूनो b चुनता है। वे चैनल पर खुले में g^a और g^b का आदान-प्रदान करते हैं। एलिस $(g^b)^a$ की गणना करती है और ब्रूनो $(g^a)^b$ की गणना करता है; दोनों एक ही g^{ab} पर पहुँचते हैं। ईवा g, g^a और g^b को चैनल से गुजरते हुए देखती है, लेकिन g^a से a

को पुनर्प्राप्त करना — जिसे असतत लघुगणक समस्या कहा जाता है — ब्रह्मांड की आयु से बेहतर खगोलीय कंप्यूटिंग समय की आवश्यकता होती है जब g को एक उपयुक्त गणितीय समूह में चुना जाता है।

Para quien quiera comprobarlo con números pequeños. El intercambio Diffie-Hellman se puede recorrer entero con cifras lo bastante reducidas como para hacer las cuentas a mano. Quien prefiera no entrar en aritmética puede saltarse este bloque sin perder el hilo del artículo; quien quiera ver el mecanismo funcionando paso a paso lo encontrará aquí. **Las reglas públicas**, que cualquiera puede leer: un primo $p = 11$ (en el Diffie-Hellman real es de unas trescientas cifras; usamos once para que las cuentas quepan en una página), una base $g = 2$, y la convención de que toda la aritmética se hace *módulo p* — se calcula, se divide entre p, y se conserva el resto, como un reloj de once posiciones que vuelve al cero al rebasar el diez. **Las elecciones privadas**, una cada uno y jamás compartidas: Alicia elige $a = 4$. Bruno elige $b = 7$.

Paso 1. Alicia calcula $2^4 = 16$, luego $16 \bmod 11 = \mathbf{5}$. Envía el cinco. Eva lo anota.

Paso 2. Bruno calcula $2^7 = 128$, luego $128 \bmod 11 = \mathbf{7}$. Envía el siete. Eva también lo anota. Tras los dos envíos, la libreta de Eva contiene cuatro datos: $p = 11$, $g = 2$, A = 5, B = 7. Le falta el número compartido que Alicia y Bruno están a punto de derivar — y que Eva no podrá reconstruir.

Paso 3. Alicia toma el siete que Bruno le envió y lo eleva a su exponente privado $a = 4$. Para evitar manejar $7^4 = 2401$, se calcula por partes aplicando el módulo en cada paso:

$7^2 = 49$

$49 \bmod 11 = 5$

$7^4 = (7^2)^2 = 5^2 = 25$

$25 \bmod 11 = 3$

Alicia obtiene el número **3**.

Paso 4. Bruno toma el cinco que Alicia le envió y lo eleva a su exponente privado $b = 7$. De nuevo por partes:

$5^2 = 25 \bmod 11 = 3$

$5^4 = (5^2)^2 = 3^2 = 9 \bmod 11 = 9$

$5^6 = 5^4 \times 5^2 = 9 \times 3 = 27 \bmod 11 = 5$

Finalmente $5^7 = 5^6 \times 5 = 5 \times 5 = 25 \bmod 11 = \mathbf{3}$.

Bruno obtiene también **3**.

Los dos han llegado al mismo número, 3, trabajando en paralelo. Ninguno envió su exponente privado en ningún momento. Alicia no sabe que $b = 7$; Bruno no sabe que $a = 4$. Cada cual usó el valor público que el otro envió combinado con su propio exponente privado, y se encontraron en el mismo destino. **¿Por qué llegan al mismo número?** Lo que calculó cada uno: Alicia, $(g^b)^a = 2^{7\times4} = 2^{28} \bmod 11$. Bruno, $(g^a)^b = 2^{4\times7} = 2^{28} \bmod 11$. Es la misma cantidad porque el orden de multiplicación de exponentes no importa ($7 \times 4 = 4 \times 7$). Cada cual llegó por un camino distinto al mismo destino.

¿Y Eva? Tiene en su libreta $p = 11$, $g = 2$, A = 5, B = 7, y quisiera el 3. Para calcularlo necesitaría conocer a o b — pero ninguno ha viajado por el canal. Su única vía es preguntarse: «¿para qué exponente a se cumple $2^a \bmod 11 = 5$?». Con p tan pequeño puede probar 0, 1, 2, 3, 4… y encontrarlo en menos de un minuto. Pero al sustituir 11 por un primo de trescientas cifras, el espacio de exponentes posibles tiene más elementos que átomos hay en el universo observable. **No existe a día de hoy ningún algoritmo conocido por la humanidad que pueda recorrer ese espacio en menos de miles de millones de años.** Es el llamado *problema del logaritmo discreto*: fácil hacia adelante, computacionalmente imposible hacia atrás. Y es la razón por la que el cifrado resiste aunque Eva haya seguido toda la conversación letra por letra.

Tres ingredientes simples —aritmética sobre un reloj, exponenciación, y conmutatividad de la multiplicación ($a \cdot b = b \cdot a$)— combinados producen un protocolo del que media humanidad depende cada día para sus comunicaciones privadas. Ninguna de las tres piezas, por separado, parece especial. Lo decisivo es el ensamblaje.

Diffie-Hellman से Signal प्रोटोकॉल तक

आज के पेशेवर मैसेजिंग एप्लिकेशन द्वारा उपयोग किया जाने वाला एंड-टू-एंड एन्क्रिप्शन, लगभग बिना किसी अपवाद के, Diffie-Hellman विनिमय के एक सुरुचिपूर्ण और कड़े संस्करण पर टिका है। Trevor Perrin और Moxie Marlinspike द्वारा 2013 और 2016 के बीच डिज़ाइन किया गया Signal प्रोटोकॉल संदर्भ है। यह दो प्रमुख विचारों को जोड़ता है। पहला, अण्डाकार वक्रों (X25519) में कुंजी विनिमय, जो दो उपकरणों के बीच प्रारंभिक साझा रहस्य उत्पन्न करता है। दूसरा, जिसे Double Ratchet — दोहरा गियर — कहा जाता है, जो प्रत्येक संदेश के साथ कुंजियों को स्वचालित रूप से नवीनीकृत करता है, ताकि आज डिवाइस से समझौता करने से पिछले संदेशों को डिक्रिप्ट करने की अनुमति न मिले, और न ही गियर घुमाए जाने के बाद भविष्य के संदेशों को।

Zig में, X25519 विनिमय जो दो उपकरणों के बीच साझा रहस्य उत्पन्न करता है, मानक पुस्तकालय का उपयोग करते हुए छह पंक्तियों में फिट बैठता है:

```
const std = @import("std");
const X25519 = std.crypto.dh.X25519;
```

```
// Alicia y Bruno generan cada uno un par (privada, pública).
const par_alicia = X25519.KeyPair.generate(io);
const par_bruno  = X25519.KeyPair.generate(io);

// Cada parte recibe la clave pública de la otra y deriva el mismo secreto.
const secreto_alicia = X25519.scalarmult(par_alicia.secret_key, par_bruno.public_key) catch unreachable;
const secreto_bruno  = X25519.scalarmult(par_bruno.secret_key,  par_alicia.public_key) catch unreachable;
// secreto_alicia == secreto_bruno  (32 bytes)
```

उन छह पंक्तियों में क्या होता है: सार्वजनिक कुंजियाँ खुले तौर पर यात्रा करती हैं। निजी कुंजियाँ कभी भी संबंधित डिवाइस को नहीं छोड़ती हैं। प्रत्येक पक्ष अपनी निजी और दूसरे की सार्वजनिक कुंजी से बत्तीस बाइट्स का वही रहस्य प्राप्त करता है जिसे चैनल में कोई भी पुनर्प्राप्त नहीं कर सकता है। वह रहस्य बाद में आदान-प्रदान किए गए संदेशों को एन्क्रिप्ट करने के लिए बीज के रूप में कार्य करता है। Signal प्रोटोकॉल का Double Ratchet उस सामग्री का निरंतर घुमाव जोड़ता है ताकि एक पल के समझौते से बाकी बातचीत से समझौता न हो।

और `std.crypto.dh.X25519` के अंदर वास्तव में क्या है? कोई छिपा हुआ जादू नहीं। ये दो छोटे फलन (functions) हैं जिन्हें पूरी तरह से Zig की अपनी मानक लाइब्रेरी में पढ़ा जा सकता है। पहला निजी कुंजी से सार्वजनिक कुंजी प्राप्त करता है — विनिमय का «g^a»:

```
pub fn recoverPublicKey(secret_key: [secret_length]u8) IdentityElementError![public_length]u8 {
    const q = try Curve.basePoint.clampedMul(secret_key);
    return q.toBytes();
}
```

लेख की भाषा में: निजी कुंजी को Curve25519 वक्र के आधार बिंदु से «गुणा» (अण्डाकार अर्थ में, प्रारंभिक अंकगणितीय अर्थ में नहीं) किया जाता है, और परिणाम को बत्तीस बाइट्स में क्रमांकित (serialized) किया जाता है। `clampedMul` ऑपरेशन उस स्केलर गुणन का कठोर संस्करण है: यह उन सुरक्षा उपायों को शामिल करता है जिन्हें क्रिप्टोग्राफ़िक समुदाय ने वर्षों से हमलों के ज्ञात परिवारों का विरोध करने के लिए जोड़ा है। फ़ंक्शन बॉडी की दो पंक्तियाँ।

दूसरा फ़ंक्शन आपकी निजी कुंजी को उस सार्वजनिक कुंजी के साथ जोड़ता है जो दूसरा पक्ष आपको भेजता है। यह विनिमय का «$(g^b)^a$» है, जो बत्तीस बाइट का साझा रहस्य उत्पन्न करता है जिसे आप दोनों ने कभी प्रेषित नहीं किया:

```
pub fn scalarmult(secret_key: [secret_length]u8, public_key: [public_length]u8) IdentityElementError![shared_length]u8 {
    const q = try Curve.fromBytes(public_key).clampedMul(secret_key);
    return q.toBytes();
}
```

दो और पंक्तियाँ। प्राप्त सार्वजनिक कुंजी की व्याख्या वक्र पर एक बिंदु के रूप में की जाती है, और स्वयं की निजी कुंजी से «गुणा» किया जाता है। वक्र संचालन की क्रमविनिमेयता (commutativity) द्वारा — जो कि संख्यात्मक उदाहरण में देखे गए घातांक के गुणन की क्रमविनिमेयता के अनुरूप है — दोनों पक्ष एक ही क्रमांकित (serialized) बिंदु के साथ समाप्त होते हैं: ठीक वही साझा रहस्य जिसके बारे में लेख बात करता है।

बस इतना ही। जो किसी एप्लिकेशन में जादू जैसा दिखता है, वह वास्तव में तीन-तीन पंक्तियों के दो फलन हैं। तकनीकी जटिलता एक ही ऑपरेशन, `clampedMul` में केंद्रित है, जो उसी मानक पुस्तकालय में आगे लिखा गया है, जिसकी दशकों से अंतर्राष्ट्रीय क्रिप्टोग्राफ़िक समुदाय द्वारा समीक्षा की गई है, और जो कोई भी इसे अक्षर दर अक्षर पढ़ना चाहता है उसके लिए उपलब्ध है। न तो हमारे एप्लिकेशन में और न ही Zig की मानक लाइब्रेरी में कोई ब्लैक बॉक्स है। एक ओपन सोर्स कोड है जिसे इंसान समझ सकता है, वह गति चुनकर जिस पर वह इसमें तल्लीन होना चाहता है।

एंड-टू-एंड एन्क्रिप्शन क्या सुरक्षित करता है

सही कार्यान्वयन को मानते हुए, E2EE पारगमन में संदेश की सामग्री की अच्छी तरह से रक्षा करता है। एक मध्यवर्ती सर्वर जो एन्क्रिप्टेड डेटा प्राप्त करता है और आगे भेजता है, वह समझ से बाहर बाइट्स का उत्तराधिकार देखेगा। केबल, राउटर, वाईफाई एक्सेस पॉइंट तक पहुंच रखने वाला हमलावर भी वही देखेगा। एक सेवा प्रदाता जो ट्रैफ़िक की प्रतियां रखता है, वह बाद में इसे पढ़ने में सक्षम नहीं होगा। एक सरकार जो सेवा ऑपरेटर को सामग्री वितरित करने का आदेश देती है, उसे वही समझ से बाहर बाइट्स प्राप्त होंगे जो सर्वर के पास पहले स्थान पर थे।

यह, व्यावहारिक शब्दों में, बहुत कुछ है। यह एक अपारदर्शी लिफाफे के अंदर एक पत्र लिखने और उसे पोस्टकार्ड पर लिखने के बीच का अंतर है। दोनों पहुँचते हैं। केवल एक ही डाकिया से सामग्री को सुरक्षित रखता है।

एंड-टू-एंड एन्क्रिप्शन क्या सुरक्षित नहीं करता है

इसे अच्छी तरह से जानना सार्थक है। E2EE मेटाडेटा की रक्षा नहीं करता है: सर्वर अभी भी जानता है कि उपयोगकर्ता A उपयोगकर्ता B को किस समय, किस आवृत्ति के साथ और कहाँ से डेटा भेजता है, हालाँकि वह यह नहीं जानता कि वे क्या कहते हैं। ये मेटाडेटा, जैसा कि हमने पहले ही *एन्क्रिप्ट करना निजी होना नहीं है* में तर्क दिया है, अक्सर सामग्री की तुलना में अधिक खुलासा करते हैं। यह जानना कि किसी ने शुक्रवार को रात 22:00 बजे तीस मिनट के लिए तलाक में विशेषज्ञता रखने वाली एक कानूनी फर्म को फोन किया, एक ऐसी कहानी बताता है जो कॉल की सामग्री ने कभी नहीं बताई। यह एक ऑन्कोलॉजी क्लिनिक में एक व्यक्ति को कई बार प्रवेश करते और छोड़ते हुए देखने जैसी ही स्थिति है: क्या हो रहा है इसकी कल्पना करने के लिए अंदर जो कुछ बोला जा रहा है उसे सुनने की जरूरत नहीं है। एक अकेला पृथक मेटाडेटा का कोई मतलब नहीं हो सकता है; कई क्रॉस-रेफ़रेंस सत्य के समान कुछ चित्रित करते हैं। E2EE सिरों (ends) की रक्षा नहीं करता है: यदि प्राप्तकर्ता का डिवाइस एक दुर्भावनापूर्ण प्रोग्राम द्वारा समझौता किया गया है, तो संदेश उस प्राप्तकर्ता के लिए सामान्य रूप से डिक्रिप्ट किया जाता है और दुर्भावनापूर्ण प्रोग्राम इसे पढ़ता है। E2EE अपने आप में वार्ताकार की पहचान के खिलाफ रक्षा नहीं करता है: यदि एलिस को लगता है कि वह ब्रूनो से बात कर रही है लेकिन एक हमलावर ने शुरुआत में खुद को बीच में रख लिया है (एक *man in the middle*) और प्रोटोकॉल में स्वतंत्र सत्यापन शामिल नहीं है, तो दोनों पक्ष घुसपैठिये से बात करते हुए समाप्त होते हैं और सोचते हैं कि वे एक-दूसरे से बात कर रहे हैं।

एक चौथी चीज़ है जिसे बिना किसी अस्पष्टता के तैयार करना सार्थक है। E2EE किसी प्रदाता को जो इसे पेश करने का दावा करता है, उसे अपने स्वयं के सिस्टम में अनएन्क्रिप्टेड संदेश की एक प्रति रखने से नहीं रोकता है। यह कथन कि «मेरे संदेश एंड-टू-एंड एन्क्रिप्टेड हैं» और यह कथन कि «प्रदाता मेरी सामग्री नहीं रखता है» समान नहीं हैं। एक एप्लिकेशन पहले को पूरा कर सकता है जबकि दूसरे का उल्लंघन कर सकता है; हमने इसे 2018 से बार-बार प्रेस की सुर्खियों में देखा है। उपयोगकर्ता के पास, जब तक कि क्लाइंट का कोड सत्यापन योग्य न हो, विशेषज्ञ जांच के बिना एक मामले को दूसरे से अलग करने का कोई तकनीकी तरीका नहीं है। आम जनता में सबसे ज्ञात मामला: WhatsApp पारगमन में संदेशों को एंड-टू-एंड एन्क्रिप्ट करता है, लेकिन यदि उपयोगकर्ता अतिरिक्त एन्क्रिप्शन के बिना iCloud या Google Drive में बैकअप सक्रिय करता है, तो वह प्रति किसी तीसरे पक्ष के बुनियादी ढांचे में पठनीय रूप से संग्रहीत की जाती है, और एन्क्रिप्शन स्वयं उपयोगकर्ता के अंत में टूट जाता है।

वह प्रश्न जो ऑपरेटर नहीं सुनना चाहता

एक एप्लिकेशन जो एंड-टू-एंड एन्क्रिप्ट करने का दावा करता है, तकनीकी रूप से कुंजियों के संबंध में तीन चीजों में से एक कर सकता है:

1. **कुंजियाँ केवल उपकरणों पर रहती हैं।** वे विशेष रूप से उपयोगकर्ताओं के उपकरणों पर उत्पन्न होती हैं और रहती हैं; ऑपरेटर उन्हें नहीं जानता है और न ही उन्हें संग्रहीत करता है। यह इष्टतम मामला है।
2. **ऑपरेटर यदि चाहे तो एक्सेस कर सकता है।** ऑपरेटर के पास उपयोगकर्ताओं की चाबियाँ होती हैं (या वह अपनी इच्छानुसार उन्हें उत्पन्न कर सकता है) और उन्हें अपने डेटाबेस में संग्रहीत करता है। यदि वह चाहता है या मजबूर किया जाता है, तो वह सामग्री पढ़ सकता है। अधिकांश 'क्लाउड' सेवाओं का यही हाल है।
3. **ऑपरेटर डिज़ाइन द्वारा एक्सेस नहीं कर सकता, लेकिन वह एक्सेस को नियंत्रित करता है।** ऑपरेटर के पास चाबियाँ नहीं होती हैं, लेकिन उसका उस एप्लिकेशन पर नियंत्रण होता है जो उन्हें उत्पन्न करता है। यदि मजबूर किया जाता है, तो वह एक दुर्भावनापूर्ण अपडेट भेज सकता है जो एन्क्रिप्शन से पहले चाबियों या सामग्री को कैप्चर कर लेता है। कई व्यावसायिक E2EE सेवाओं का यही हाल है।

इसलिए, परिचालन प्रश्न यह नहीं है कि क्या कुछ एन्क्रिप्टेड है, बल्कि यह है कि डिवाइस और सॉफ्टवेयर पर किसका नियंत्रण है जो चाबियों का प्रबंधन करता है। Solo2 में, चाबियाँ केवल आपकी 'तिजोरी' (आपकी पासवर्ड से एन्क्रिप्टेड IndexedDB) में रहती हैं और सॉफ्टवेयर सत्यापन योग्य ओपन सोर्स कोड है।

पेशेवर पाठक के लिए

एंड-टू-एंड एन्क्रिप्शन डिजिटल संप्रभुता के लिए एक उपकरण है। लेकिन हर उपकरण की तरह, इसकी प्रभावशीलता इसे चलाने वाले हाथ और उस जमीन पर निर्भर करती है जिस पर यह टिका है।

1. क्रिप्टोग्राफ़िक कुंजियाँ कहाँ उत्पन्न होती हैं और भौतिक रूप से कहाँ रहती हैं? यदि ऑपरेटर उन तक पहुँच सकता है (भले ही अस्थायी रूप से, भले ही रिकवरी के बहाने), तो E2EE केवल नाममात्र है।
2. क्या वार्ताकार का कोई स्वतंत्र सत्यापन (सुरक्षा नंबर, QR कोड, आउट-ऑफ-बैंड तुलना) है जो बातचीत की स्थापना के दौरान मैन-इन-द-मिडिल हमले को रोकता है?
3. क्या क्लाइंट कोड ऑडिट योग्य है — खुला, प्रकाशित, पुनरुत्पादनीय — या क्या इसके लिए यह विश्वास करने की आवश्यकता है कि क्लाइंट वास्तव में क्या करता है, इस पर प्रदाता के शब्द पर विश्वास किया जाए?
4. सेवा कौन सा मेटाडेटा उत्पन्न करती है और रखती है, और कितने समय के लिए? भले ही सामग्री अपारदर्शी हो, मेटाडेटा संवेदनशील जानकारी के एक बड़े हिस्से को फिर से बना सकता है।

ये चार प्रश्न उन्नत तकनीकी जानकारी नहीं मांगते हैं; वे ऐसी जानकारी मांगते हैं जिसका उत्तर कोई भी ईमानदार ऑपरेटर अपने सार्वजनिक दस्तावेज़ों में दे सकता है। उत्तर की गुणवत्ता और सटीकता उत्पाद के बारे में उतनी ही जानकारी देती है जितनी स्वयं उत्तर।

एंड-टू-एंड एन्क्रिप्शन, सही तरीके से किया गया, समकालीन क्रिप्टोग्राफी द्वारा दैनिक अभ्यास के लिए प्रदान किए गए बेहतरीन निर्माणों में से एक है। मूल विचार — दो लोग एक सार्वजनिक चैनल पर एक रहस्य पर सहमत हो सकते हैं — Whitfield Diffie और Martin Hellman, 1976 का है; आधी सदी बाद हम इसके परिणाम में जी रहे हैं। लेकिन, किसी भी तकनीकी वादे की तरह, इसका मूल्य वास्तविक पूर्ति पर निर्भर करता है, न कि लेबल पर। ईमानदार पेशेवर का सवाल यह नहीं है कि «क्या यह एन्क्रिप्टेड है?», बल्कि यह है कि «चाबियां किसके पास हैं?». उत्तरों के अलग-अलग परिणाम होते हैं। उन्हें जानना बेहतर है।

स्रोत और आगे पढ़ने के लिए

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, नवंबर 1976. सार्वजनिक-कुंजी क्रिप्टोग्राफी का मूलभूत लेख।
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, Open Whisper Systems द्वारा सार्वजनिक विनिर्देश, 2016 संशोधन। Signal प्रोटोकॉल और इसके औद्योगिक डेरिवेटिव का आधार।
- RFC 7748 — Elliptic Curves for Security (IETF, जनवरी 2016). आधुनिक कुंजी आदान-प्रदान में उपयोग किए जाने वाले X25519 और X448 वक्रों का मानक विनिर्देश।
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). कुंजी विनिमय और प्रमाणित एन्क्रिप्शन प्रोटोकॉल पर अध्याय।
- यूरोपीय डिजिटल पहचान ढांचे (eIDAS 2) पर विनियमन (यूरोपीय संघ) 2024/1183 — ऐसे ढांचे स्थापित करता है जहां वार्ताकार का स्वतंत्र सत्यापन संस्थागत समर्थन प्राप्त करता है, और जहां नाममात्र और वास्तविक एन्क्रिप्शन के बीच अंतर के अलग-अलग कानूनी परिणाम होते हैं।

← पिछलाKill switch और संस्थागत कब्जाअगला → भरोसे के संकेत के रूप में बिजनेस मॉडल

हाल की रीडिंग

- विश्लेषण · 18 मई, 2026 वास्तविक बनाम आभासी गोपनीयता: वे प्रश्न जिन्हें आपको खुद से पूछना चाहिए
- विश्लेषण · 18 मई, 2026 पेशेवर अभ्यास के रूप में सेल्फ-होस्टिंग
- अवधारणा · 18 मई, 2026 वे 24 शब्द: क्रिप्टोग्राफिक पहचान क्या है

इस लेख को अपने साथ ले जाएं जहाँ भी आपको इसकी आवश्यकता हो।

↓ मार्कडाउन ↓ सादा टेक्स्ट ↓ PDF

फ़ाइल आपके डिवाइस पर डाउनलोड हो जाएगी। वहां से आप इसे सहेज सकते हैं, Solo2 में आयात कर सकते हैं या जहां चाहें साझा कर सकते हैं। Cuadernos आपके लिए गंतव्य तय नहीं करता है।

मोहरबंद · SHA-256 b54b31acd1f77b3cf480e64df901e03e2a6d30cce0a47e44f929099c45153a3d

Cuadernos Lacre · Menzuri Gestión S.L. का एक प्रकाशन ·
R.Eugenio द्वारा लिखित · Solo2 की टीम द्वारा संपादित।