Cifrado de extremo a extremo, explicado de verdad

Lo que dicen los proveedores cuando dicen E2EE, y lo que no dicen. Una explicación didáctica del mecanismo y sus límites, sin el envoltorio publicitario.

Para entendernos: WhatsApp dice que tus mensajes están cifrados de extremo a extremo. Es verdad — y no es suficiente. Si la copia de seguridad va a iCloud o Google Drive sin cifrado adicional, el cifrado se rompe en tu propio teléfono. La pregunta operativa no es si está cifrado, sino dónde residen las claves.

Lo que cifrar significa, de verdad

Cifrar un mensaje es transformarlo en algo que parece ruido para cualquiera que no posea cierta información llamada clave. La operación se hace en el dispositivo del que envía y, con la clave correcta, se deshace en el dispositivo del que recibe. En medio, el mensaje viaja como una sucesión de bytes sin significado aparente. Esa es la idea sencilla. El resto del artículo se ocupa de los matices que la convierten, según el caso, en una garantía real o en una etiqueta de mercado.

El adjetivo *de extremo a extremo* —en inglés *end-to-end*, abreviado E2EE— añade una precisión. El cifrado no se hace para que un servidor intermedio pueda leerlo y entregarlo. Se hace para que solo los dos extremos —el dispositivo del que envía y el dispositivo del que recibe— posean la clave. Cualquier servidor por el que el mensaje pase ve el ruido, no el mensaje. Esa es la diferencia técnica con el cifrado *en tránsito*, donde el contenido va cifrado de un servidor al siguiente, pero cada servidor por el que pasa lo descifra para reenviarlo, recuperando temporalmente el texto en claro.

La paradoja del secreto compartido

Hay un problema obvio. Para que dos personas puedan cifrar y descifrar mensajes entre sí, ambas necesitan la misma clave. Pero, ¿cómo se ponen de acuerdo en esa clave si todo lo que se mandan, por definición, pasa por un canal donde alguien podría estar escuchando? Acordar la clave en el mismo canal donde después la usarán parece imposible: si el atacante la oye al acordarla, podrá descifrar todo lo posterior. Durante decenios, la criptografía clásica resolvió esto por la vía dura: las claves se entregaban en persona, antes de empezar a usarse, en encuentros físicos. Los embajadores cargaban con maletines de claves cosidos al forro del abrigo.

En el correo electrónico contemporáneo, esa solución no escala. Si tuviéramos que ir físicamente a casa de cada persona con la que pretendiéramos comunicarnos de forma cifrada, no llegaríamos a hablar con nadie. La pregunta planteada hace cincuenta años por la comunidad criptográfica era esta: ¿es posible que dos personas que no se conocen y que solo comparten un canal público acuerden, en ese mismo canal público, un secreto que nadie que escuche el canal pueda conocer?

La elegancia de Diffie-Hellman

En 1976, dos matemáticos llamados Whitfield Diffie y Martin Hellman demostraron algo aparentemente imposible: que dos personas, hablando solo por un canal público —un canal donde cualquiera puede escuchar todo lo que dicen—, pueden ponerse de acuerdo en una contraseña secreta sin que ningún oyente pueda descubrirla. Suena a magia. No lo es: es matemática. El intercambio de claves Diffie-Hellman, como se conoce desde entonces, es la base de prácticamente toda la comunicación cifrada de internet, y medio siglo de uso intensivo y escrutinio académico mundial avalan su solidez. Quien quiera ver la intuición visual o la matemática puede seguir leyendo. Quien prefiera confiar en que funciona también puede continuar sin perder el hilo del artículo.

Para quien quiera intuirlo en una imagen, hay una analogía conocida con colores. Imagina que Alicia y Bruno acuerdan en abierto un color base —digamos amarillo— a la vista de Eva, que les escucha. Cada uno elige en privado un segundo color secreto y mezcla su secreto con el amarillo. Alicia obtiene un naranja particular; Bruno obtiene un verde particular. Intercambian los resultados a la vista de Eva. Ahora cada uno mezcla el color recibido con su propio secreto, y ambos llegan al mismo color final, porque el orden de las mezclas no importa. Eva ha visto el amarillo y las dos mezclas intermedias, pero no los secretos; sin alguno de los secretos no puede llegar al color final. La matemática real cambia los colores por exponenciaciones en grupos modulares o curvas elípticas, pero la idea es la misma: el secreto compartido se construye en público sin que nadie en el canal pueda reconstruirlo.

En aritmética, para quien prefiera ver el mecanismo: Alicia elige un número secreto a, Bruno elige b. Intercambian g^a y g^b en abierto sobre el canal. Alicia calcula $(g^b)^a$ y Bruno calcula $(g^a)^b$; ambos llegan al mismo g^{ab}. Eva ve g, g^a y g^b pasar por el canal, pero recuperar a desde g^a —el llamado problema del logaritmo discreto— requiere un tiempo de cómputo astronómicamente superior a la edad del universo cuando g se elige en un grupo matemático adecuado.

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.

De Diffie-Hellman al protocolo Signal

El cifrado de extremo a extremo que usan hoy las aplicaciones de mensajería profesional descansa, casi sin excepción, sobre una versión elegante y endurecida del intercambio Diffie-Hellman. El protocolo Signal, diseñado por Trevor Perrin y Moxie Marlinspike entre 2013 y 2016, es la referencia. Combina dos ideas clave. La primera, el intercambio de claves en curvas elípticas (X25519), que produce el secreto compartido inicial entre dos dispositivos. La segunda, el llamado Double Ratchet —doble engranaje—, que renueva las claves automáticamente con cada mensaje, de modo que comprometer el dispositivo hoy no permite descifrar mensajes pasados, ni mensajes futuros una vez se ha rotado el engranaje.

En Zig, el intercambio X25519 que produce el secreto compartido entre dos dispositivos cabe en seis líneas, usando la biblioteca estándar:

```
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)
```

Lo que pasa en esas seis líneas: Las claves públicas viajan abiertamente. Las claves privadas no salen nunca del dispositivo respectivo. Cada parte deriva, a partir de su privada y la pública de la otra, un mismo secreto de treinta y dos bytes que nadie en el canal puede recuperar. Ese secreto sirve después como semilla para cifrar los mensajes intercambiados. El Double Ratchet del protocolo Signal añade una rotación constante de ese material para que el compromiso de un instante no comprometa el resto de la conversación.

Y dentro de `std.crypto.dh.X25519`, ¿qué hay exactamente? No magia oculta. Son dos funciones cortas que se pueden leer enteras en la propia biblioteca estándar de Zig. La primera deriva la clave pública desde la privada — el «g^a» del intercambio:

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

En el lenguaje del artículo: la clave privada se «multiplica» —en el sentido elíptico, no en el aritmético elemental— por el punto base de la curva Curve25519, y el resultado se serializa en treinta y dos bytes. La operación `clampedMul` es la versión endurecida de esa multiplicación escalar: incorpora las salvaguardas que la comunidad criptográfica fue añadiendo a lo largo de años para resistir familias conocidas de ataques. Dos líneas de cuerpo de función.

La segunda función combina tu clave privada con la clave pública que la otra parte te envía. Es el «$(g^b)^a$» del intercambio, el que produce el secreto compartido de treinta y dos bytes que ninguno de los dos llegó a transmitir:

```
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();
}
```

Otras dos líneas. La clave pública recibida se interpreta como un punto sobre la curva, y se «multiplica» por la clave privada propia. Por la conmutatividad de la operación de curva —análoga a la conmutatividad de la multiplicación de exponentes que vimos en el ejemplo numérico—, ambas partes acaban con el mismo punto serializado: exactamente el secreto compartido del que habla el artículo.

Eso es todo. Lo que en una aplicación parece magia es, en la realidad, dos funciones de tres líneas cada una. La complejidad técnica se concentra en una sola operación, `clampedMul`, que está escrita más adelante en la misma biblioteca estándar, revisada durante décadas por la comunidad criptográfica internacional, y disponible para cualquiera que quiera leerla letra por letra. No hay caja negra ni en nuestra aplicación ni en la biblioteca estándar de Zig. Hay código abierto que un humano puede entender, eligiendo el ritmo al que quiere entrar.

Qué protege el cifrado de extremo a extremo

Lo que el E2EE protege bien, asumiendo una implementación correcta, es el contenido del mensaje en tránsito. Un servidor intermedio que reciba y reenvíe los datos cifrados verá una sucesión de bytes ininteligibles. Un atacante con acceso al cable, al router, al punto de acceso wifi, verá lo mismo. Un proveedor del servicio que conserve copias del tráfico no podrá leerlo a posteriori. Un Gobierno que ordene al operador del servicio entregar el contenido recibirá los mismos bytes ininteligibles que tenía el servidor en primer lugar.

Esto, en términos prácticos, es mucho. Es la diferencia entre escribir una carta dentro de un sobre opaco y escribirla en una postal. Las dos llegan. Solo una preserva el contenido frente al cartero.

Qué no protege el cifrado de extremo a extremo

Conviene saberlo igual de bien. El E2EE no protege los metadatos: el servidor sigue sabiendo que el usuario A envía datos al usuario B, a qué hora, con qué frecuencia y desde dónde, aunque no sepa qué dice. Estos metadatos, ya lo hemos argumentado en *Cifrar no es ser privado*, son a menudo más reveladores que el contenido. Saber que alguien llamó a un despacho de abogados especializado en divorcios un viernes a las 22:00 durante treinta minutos cuenta una historia que el contenido de la llamada nunca contó. Es la misma situación que ver a una persona entrar y salir varias veces de una clínica oncológica: no hace falta oír nada de lo que se habla dentro para imaginar lo que está pasando. Un solo metadato suelto puede no significar nada; varios cruzados entre sí dibujan algo demasiado parecido a la verdad. El E2EE no protege los extremos: si el dispositivo del receptor está comprometido por un programa malicioso, el mensaje se descifra normalmente para ese receptor y el programa malicioso lo lee. El E2EE no protege contra la identidad del interlocutor en sí: si Alicia cree estar hablando con Bruno pero un atacante se ha interpuesto al inicio (un *man in the middle*) y el protocolo no incluye verificación independiente, las dos partes acaban hablando con el intruso pensando que hablan entre sí.

Hay una cuarta cosa que conviene formular sin ambigüedad. El E2EE no impide que un proveedor que afirma ofrecerlo guarde, además, una copia del mensaje sin cifrar en sus propios sistemas. La afirmación «mis mensajes están cifrados de extremo a extremo» y la afirmación «el proveedor no conserva mi contenido» no son la misma. Una aplicación puede cumplir la primera mientras incumple la segunda; lo hemos visto en titulares de prensa repetidamente desde 2018. El usuario, salvo que el código del cliente sea verificable, no tiene forma técnica de distinguir un caso del otro sin investigación experta. El caso más conocido en el público general: WhatsApp cifra los mensajes de extremo a extremo en tránsito, pero si el usuario activa la copia de seguridad en iCloud o Google Drive sin cifrado adicional, esa copia se almacena legible en infraestructura de un tercero, y el cifrado se rompe en el extremo del propio usuario.

La pregunta que el operador no quiere oír

Una aplicación que afirma cifrar de extremo a extremo puede, técnicamente, hacer una de tres cosas con respecto a las claves:

1. **Las claves residen solo en los dispositivos.** Se generan y residen exclusivamente en los dispositivos de los usuarios; el operador no las conoce ni las almacena. Es el caso óptimo.
2. **Las claves residen en los dispositivos, pero el operador conserva una copia.** Bajo formulaciones como «recuperación de cuenta», «sincronización entre dispositivos» o «funcionalidad opcional». En este caso el cifrado es de extremo a extremo solo nominalmente; el operador, si quiere o si se lo ordenan, puede descifrar.
3. **Las claves las gestiona el operador entero.** Con un envoltorio criptográfico que conserva la apariencia del E2EE sin la sustancia. Este último caso es menos frecuente, pero existe.

La pregunta operativa para el profesional, por tanto, no es si el proveedor afirma cifrar de extremo a extremo. La pregunta es: ¿dónde residen las claves, quién las controla, y qué pasaría si una orden judicial llegara mañana al operador pidiendo descifrar mis conversaciones? Las tres respuestas a esa pregunta son completamente distintas, aunque las tres aplicaciones se presenten en su publicidad con la misma etiqueta.

Para el lector profesional

Cuatro preguntas operativas a formular al evaluar un servicio de comunicación profesional que afirma ofrecer cifrado de extremo a extremo:

1. ¿Dónde se generan las claves criptográficas y dónde residen físicamente? Si el operador puede acceder a ellas (incluso temporalmente, incluso bajo formulación de recuperación), el E2EE es nominal.
2. ¿Existe verificación independiente del interlocutor (números de seguridad, códigos QR, comparación fuera de banda) que impida un ataque de hombre en el medio durante el establecimiento de la conversación?
3. ¿El código del cliente es auditable —abierto, publicado, reproducible— o exige confiar en la palabra del proveedor sobre lo que el cliente hace en realidad?
4. ¿Qué metadatos genera y conserva el servicio, y por cuánto tiempo? Aunque el contenido sea opaco, los metadatos pueden reconstruir buena parte de la información sensible.

Estas cuatro preguntas no piden información técnica avanzada; piden información que cualquier operador honesto puede responder en su documentación pública. La calidad y precisión de la respuesta dice tanto del producto como la respuesta misma.

El cifrado de extremo a extremo, hecho bien, es una de las construcciones más finas que la criptografía contemporánea ha entregado a la práctica cotidiana. La idea original —dos personas pueden acordar un secreto en un canal público— pertenece a Whitfield Diffie y Martin Hellman, 1976; medio siglo después seguimos viviendo en su consecuencia. Pero, como sucede con cualquier promesa técnica, su valor depende del cumplimiento real, no de la etiqueta. La pregunta del profesional honesto no es «¿está cifrado?», sino «¿quién tiene las claves?». Las respuestas tienen consecuencias distintas. Conviene saberlas.

Fuentes y lectura adicional

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, noviembre de 1976. Artículo fundacional de la criptografía de clave pública.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, especificación pública de Open Whisper Systems, revisión de 2016. Base del protocolo Signal y sus derivados industriales.
- RFC 7748 — Elliptic Curves for Security (IETF, enero de 2016). Especificación normativa de las curvas X25519 y X448 usadas en intercambios de clave modernos.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). Capítulos sobre intercambio de claves y protocolos de cifrado autenticado.
- Reglamento (UE) 2024/1183 de espacio europeo de identidad digital (eIDAS 2) — establece marcos donde la verificación independiente del interlocutor adquiere apoyo institucional, y donde la distinción entre cifrado nominal y cifrado real tiene consecuencias jurídicas distintas.

← AnteriorKill switch y la captura institucionalSiguiente →El modelo de negocio como señal de confianza

Lecturas recientes

- Análisis · 18 de mayo de 2026 Privacidad real vs aparente: las preguntas que conviene hacerse
- Análisis · 18 de mayo de 2026 Self-hosting como práctica profesional
- Concepto · 18 de mayo de 2026 Las 24 palabras: qué es una identidad criptográfica

Llévate este artículo donde lo necesites.

↓ Markdown ↓ Texto plano ↓ PDF

El archivo se descarga a tu dispositivo. Desde ahí puedes guardarlo, importarlo a Solo2, o compartirlo donde quieras. Cuadernos no decide el destino por ti.

Sello de lacre · SHA-256 e8d609f93c3b97970df4a7dee9d349453c8d4e7699d4e6d31e81df49c28f6f9a

Cuadernos Lacre · Una publicación de Menzuri Gestión S.L. ·
escrita por R.Eugenio · editada por el equipo de Solo2.