Végpontok közötti titkosítás, igazán elmagyarázva

Amit a szolgáltatók mondanak, amikor azt mondják: E2EE, és amit elhallgatnak. A mechanizmus és határainak didaktikus magyarázata, reklámcsomagolás nélkül.

Hogy tisztázzuk: A WhatsApp azt állítja, hogy az üzenetei végpontok között titkosítottak. Ez igaz — és nem elég. Ha a biztonsági mentés további titkosítás nélkül az iCloudba vagy a Google Drive-ra kerül, a titkosítás a saját telefonján törik meg. Az operatív kérdés nem az, hogy titkosítva van-e, hanem az, hogy hol találhatók a kulcsok.

Amit a titkosítás valójában jelent

Egy üzenet titkosítása azt jelenti, hogy olyan dologgá alakítjuk, ami zajnak tűnik bárki számára, aki nem rendelkezik egy bizonyos, kulcsnak nevezett információval. A művelet a küldő eszközén történik, és a helyes kulccsal a fogadó eszközén fordítódik vissza. Közben az üzenet látható értelem nélküli bájtok sorozataként utazik. Ez az egyszerű alapötlet. A cikk többi része azokkal az árnyalatokkal foglalkozik, amelyek esettől függően valódi garanciává vagy piaci címkévé teszik azt.

A *végpontok közötti* jelző — angolul *end-to-end*, rövidítve E2EE — pontosítást ad hozzá. A titkosítás nem azért történik, hogy egy köztes szerver elolvashassa és kézbesíthesse. Azért történik, hogy csak a két végpont — a küldő és a fogadó eszköze — rendelkezzen a kulccsal. Bármely szerver, amelyen az üzenet áthalad, a zajt látja, nem az üzenetet. Ez a technikai különbség a *tranzit közbeni* titkosításhoz képest, ahol a tartalom titkosítva utazik egyik szervertől a másikig, de minden szerver, amelyen áthalad, dekódolja a továbbításhoz, ideiglenesen visszaállítva a nyílt szöveget.

A megosztott titok paradoxona

Van egy nyilvánvaló probléma. Ahhoz, hogy két ember titkosítani és dekódolni tudja az egymásnak küldött üzeneteket, mindkettőjüknek ugyanarra a kulcsra van szüksége. De hogyan egyeznek meg ebben a kulcsban, ha minden, amit egymásnak küldenek, definíció szerint egy olyan csatornán halad át, ahol valaki hallgatózhat? Ugyanazon a csatornán megegyezni a kulcsban, ahol később használni fogják, lehetetlennek tűnik: ha a támadó hallja a megegyezéskor, képes lesz minden későbbit dekódolni. Évtizedekig a klasszikus kriptográfia ezt a nehezebb úton oldotta meg: a kulcsokat személyesen, a használat megkezdése előtt, fizikai találkozókon adták át. A nagykövetek a kabátjuk bélésébe varrt kulcsos táskákkal közlekedtek.

A kortárs e-mailben ez a megoldás nem skálázható. Ha mindenkihez személyesen el kellene mennünk, akivel titkosítva szeretnénk kommunikálni, soha nem jutnánk el odáig, hogy bárkivel is beszéljünk. A kriptográfiai közösség által ötven évvel ezelőtt feltett kérdés ez volt: lehetséges-e, hogy két ember, aki nem ismeri egymást és csak egy nyilvános csatornán osztozik, ugyanezen a nyilvános csatornán megegyezzen egy olyan titokban, amelyet a csatornát hallgató senki sem tudhat meg?

A Diffie-Hellman eleganciája

1976-ban két matematikus, Whitfield Diffie és Martin Hellman bebizonyított valami látszólag lehetetlent: két ember, akik csak egy nyilvános csatornán keresztül beszélnek — egy olyan csatornán, ahol bárki hallhat mindent, amit mondanak —, meg tud egyezni egy titkos jelszóban anélkül, hogy bármely hallgató fel tudná fedezni azt. Varázslatnak tűnik. Nem az: matematika. A Diffie-Hellman kulcscsere, ahogy azóta ismerik, alapja gyakorlatilag az összes titkosított internetes kommunikációnak, és fél évszázadnyi intenzív használat és globális tudományos vizsgálat igazolja szilárdságát. Aki látni szeretné a vizuális intuíciót vagy a matematikát, olvashat tovább. Aki inkább bízik abban, hogy működik, szintén folytathatja anélkül, hogy elveszítené a cikk fonalát.

Aki képben szeretné látni, annak van egy ismert analógia színekkel. Képzeld el, hogy Alice és Bruno nyilvánosan megegyeznek egy alapszínben — mondjuk a sárgában — az őket hallgató Éva szeme láttára. Mindketten választanak maguknak titokban egy második titkos színt, és összekeverik a titkukat a sárgával. Alice egy sajátos narancssárgát kap; Bruno egy sajátos zöldet. Kicserélik az eredményeket Éva szeme láttára. Most mindketten összekeverik a kapott színt a saját titkukkal, és mindketten ugyanahhoz a végső színhez jutnak, mert a keverés sorrendje nem számít. Éva látta a sárgát és a két köztes keveréket, de a titkokat nem; a titkok nélkül nem tud eljutni a végső színhez. A valódi matematika a színeket moduláris csoportokban vagy elliptikus görbéken végzett hatványozásra cseréli, de az ötlet ugyanaz: a közös titok nyilvánosan épül fel anélkül, hogy a csatornán bárki képes lenne rekonstruálni azt.

Az aritmetikában, aki jobban szereti látni a mechanizmust: Alice választ egy a titkos számot, Bruno választ egy b-t. Nyilvánosan kicserélik a g^a és g^b értékeket a csatornán. Alice kiszámítja a $(g^b)^a$ értéket, Bruno pedig a $(g^a)^b$ értéket; mindketten ugyanahhoz a g^{ab}-hez jutnak. Éva látja a g, g^a és g^b értékeket áthaladni a csatornán, de az a visszanyerése a g^a-ból — az úgynevezett diszkrét logaritmus probléma — csillagászati számítási időt igényel, amely meghaladja az univerzum életkorát, ha a g-t egy megfelelő matematikai csoportból választják.

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.

A Diffie-Hellman-tól a Signal protokollig

A végpontok közötti titkosítás, amelyet ma a professzionális üzenetküldő alkalmazások használnak, szinte kivétel nélkül a Diffie-Hellman csere egy elegáns és megerősített változatán alapul. A Trevor Perrin és Moxie Marlinspike által 2013 és 2016 között tervezett Signal protokoll a referencia. Két kulcsötletet ötvöz. Az első az elliptikus görbéken alapuló kulcscsere (X25519), amely létrehozza a kezdeti közös titkot két eszköz között. A második az úgynevezett Double Ratchet — kettős racsni —, amely minden üzenettel automatikusan megújítja a kulcsokat, így az eszköz mai feltörése nem teszi lehetővé a múltbeli üzenetek dekódolását, sem a jövőbeliekét, amint a racsni elfordult.

Zig-ben a két eszköz közötti közös titkot létrehozó X25519 csere hat sorban elfér, a standard könyvtárat használva:

```
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)
```

Mi történik abban a hat sorban: A nyilvános kulcsok nyíltan utaznak. A privát kulcsok soha nem hagyják el az adott eszközt. Mindkét fél a saját privát és a másik fél nyilvános kulcsából ugyanazt a harminckét bájtos titkot származtatja le, amelyet a csatornán senki sem tud visszanyerni. Ez a titok szolgál később magként a kicserélt üzenetek titkosításához. A Signal protokoll Double Ratchet megoldása folyamatos rotációt ad ehhez az anyaghoz, hogy egy pillanatnyi feltörés ne veszélyeztesse a beszélgetés többi részét.

És pontosan mi is van az `std.crypto.dh.X25519` belsejében? Semmi rejtett varázslat. Két rövid függvényről van szó, amelyek teljes egészében elolvashatók a Zig saját standard könyvtárában. Az első levezeti a nyilvános kulcsot a privátból — a csere «g^a»-ja:

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

A cikk nyelvezetével élve: a privát kulcsot „megszorozzák” — elliptikus értelemben, nem elemi aritmetikai értelemben — a Curve25519 görbe bázispontjával, és az eredményt harminckét bájttá szerializálják. A `clampedMul` művelet ennek a skaláris szorzásnak a megerősített változata: beépíti azokat a védelmeket, amelyeket a kriptográfiai közösség az évek során hozzáadott az ismert támadási családok kivédésére. Két sornyi függvénytörzs.

A második függvény kombinálja az Ön privát kulcsát a másik fél által elküldött nyilvános kulccsal. Ez a csere «$(g^b)^a$»-ja, amely létrehozza azt a harminckét bájtos közös titkot, amelyet soha egyikük sem továbbított:

```
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();
}
```

Még két sor. A kapott nyilvános kulcsot a görbe egy pontjaként értelmezik, és „megszorozzák” a saját privát kulccsal. A görbe művelet kommutativitása révén — ami analóg a numerikus példában látott kitevők szorzásának kommutativitásával — mindkét fél ugyanazzal a szerializált ponttal a végén: pontosan a közös titokkal, amiről a cikk beszél.

Ez minden. Ami egy alkalmazásban varázslatnak tűnik, az a valóságban két, egyenként háromsoros függvény. A technikai bonyolultság egyetlen műveletbe, a `clampedMul`-be koncentrálódik, amely lejjebb található ugyanabban a standard könyvtárban, amelyet a nemzetközi kriptográfiai közösség évtizedek óta felülvizsgál, és bárki számára elérhető, aki betűről betűre el akarja olvasni. Nincs fekete doboz sem a mi alkalmazásunkban, sem a Zig standard könyvtárában. Van nyílt forráskód, amelyet egy ember is megérthet, megválasztva, milyen tempóban akar elmélyedni benne.

Amit a végpontok közötti titkosítás véd

Amit az E2EE jól véd, feltételezve a helyes megvalósítást, az az üzenet tartalma a tranzit során. Egy köztes szerver, amely fogadja és továbbítja a titkosított adatokat, érthetetlen bájtok sorozatát fogja látni. Egy támadó, aki hozzáfér a kábelhez, a routerhez vagy a wifi hozzáférési ponthoz, ugyanezt fogja látni. Egy szolgáltató, aki megőrzi a forgalom másolatait, később nem tudja elolvasni azokat. Egy kormány, amely utasítja a szolgáltatót a tartalom átadására, ugyanazokat az érthetetlen bájtokat fogja kapni, amelyek a szervernek eredetileg is megvoltak.

Ez gyakorlati szempontból nagyon sok. Ez a különbség aközött, hogy egy levelet átlátszatlan borítékba írunk, vagy képeslapra. Mindkettő megérkezik. Csak az egyik őrzi meg a tartalmat a postással szemben.

Amit a végpontok közötti titkosítás nem véd

Ugyanilyen jól érdemes tudni. Az E2EE nem védi a metaadatokat: a szerver továbbra is tudja, hogy az A felhasználó adatokat küld a B felhasználónak, hány órakor, milyen gyakorisággal és honnan, még ha azt nem is tudja, mit mondanak. Ezek a metaadatok, ahogy azt már a *Titkosítani nem jelent privátnak lenni* cikkben érveltük, gyakran beszédesebbek, mint a tartalom. Annak ismerete, hogy valaki felhívott egy válásokra szakosodott ügyvédi irodát egy pénteki napon 22:00-kor harminc percen keresztül, olyan történetet mesél el, amelyet a hívás tartalma soha nem mondott el. Ez ugyanaz a helyzet, mint látni egy embert többször bemenni és kijönni egy onkológiai klinikáról: nem kell hallani semmit abból, amiről odabent beszélnek, hogy elképzeljük, mi történik. Egyetlen elszigetelt metaadat nem biztos, hogy jelent valamit; több egymással keresztezett adat azonban valami olyasmit rajzol ki, ami túlságosan hasonlít az igazsághoz. Az E2EE nem védi a végpontokat: ha a fogadó eszközét feltöri egy kártékony program, az üzenet normálisan dekódolódik a fogadó számára, és a kártékony program elolvassa azt. Az E2EE nem véd a beszélgetőpartner identitása ellen önmagában: ha Alice azt hiszi, hogy Brunóval beszél, de egy támadó beékelődött az elején (egy *man in the middle*), és a protokoll nem tartalmaz független ellenőrzést, a két fél végül a behatolóval beszél, miközben azt hiszik, egymással beszélnek.

Van egy negyedik dolog, amit érdemes kétértelműség nélkül megfogalmazni. Az E2EE nem akadályozza meg, hogy egy szolgáltató, aki azt állítja, hogy kínálja, ezen felül megőrizze az üzenet titkosítatlan másolatát a saját rendszereiben. Az „üzeneteim végpontok között titkosítottak” állítás és a „szolgáltató nem őrzi meg a tartalmamat” állítás nem ugyanaz. Egy alkalmazás teljesítheti az elsőt, miközben megsérti a másodikat; 2018 óta többször láttuk ezt a sajtóhírekben. A felhasználónak, hacsak a kliens kódja nem ellenőrizhető, nincs technikai módja arra, hogy szakértői vizsgálat nélkül megkülönböztesse az egyik esetet a másiktól. A nagyközönség körében legismertebb eset: a WhatsApp a tranzit során végpontok között titkosítja az üzeneteket, de ha a felhasználó aktiválja a mentést az iCloud-ba vagy a Google Drive-ba további titkosítás nélkül, az a másolat olvashatóan tárolódik egy harmadik fél infrastruktúráján, és a titkosítás magánál a felhasználónál törik meg.

A kérdés, amit az üzemeltető nem akar hallani

Egy alkalmazás, amely azt állítja, hogy végpontok között titkosít, technikailag három dolog egyikét teheti a kulcsokkal kapcsolatban:

1. **A kulcsok csak az eszközökön laknak.** Kizárólag a felhasználók eszközein jönnek létre és ott is laknak; az üzemeltető nem ismeri és nem tárolja őket. Ez az optimális eset.
2. **Az üzemeltető hozzáférhet, ha akar.** Az üzemeltető rendelkezik a felhasználók kulcsaival (vagy tetszés szerint generálhatja azokat), és azokat az adatbázisaiban tárolja. Ha akarja, vagy kényszerítik rá, elolvashatja a tartalmat. Ez a helyzet a legtöbb „felhőalapú” szolgáltatásnál.
3. **Az üzemeltető tervezés szerint nem férhet hozzá, de ő ellenőrzi a hozzáférést.** Az üzemeltető nem rendelkezik a kulcsokkal, de nála van az ellenőrzés az azokat generáló alkalmazás felett. Ha kényszerítik rá, küldhet egy kártékony frissítést, amely a titkosítás előtt rögzíti a kulcsokat vagy a tartalmat. Ez a helyzet számos kereskedelmi E2EE szolgáltatásnál.

A gyakorlati kérdés tehát nem az, hogy valami titkosítva van-e, hanem az, hogy ki tartja ellenőrzése alatt az eszközt és a kulcsokat kezelő szoftvert. A Solo2-nél a kulcsok kizárólag az Ön Széfjében (az Ön jelszavával titkosított IndexedDB) találhatók, és a szoftver ellenőrizhető nyílt forráskódú.

Szakmai olvasóknak

A végpontok közötti titkosítás a digitális szuverenitás eszköze. De mint minden eszköznél, a hatékonysága attól a kéztől függ, amely forgatja, és attól a talajtól, amelyen áll.

1. Hol generálódnak a kriptográfiai kulcsok és hol találhatók fizikailag? Ha az operátor hozzáférhet (még ha ideiglenesen is, akár a helyreállítás örve alatt), az E2EE csak névleges.
2. Létezik-e a beszélgetőpartner független ellenőrzése (biztonsági számok, QR-kódok, csatornán kívüli összehasonlítás), amely megakadályozza a man-in-the-middle támadást a beszélgetés létrehozása során?
3. Auditálható-e a kliens kódja — nyílt, publikált, reprodukálható —, vagy a szolgáltató szavára kell hagyatkozni arról, hogy a kliens valójában mit csinál?
4. Milyen metaadatokat generál és őriz meg a szolgáltatás, és mennyi ideig? Még ha a tartalom átlátszatlan is, a metaadatok képesek rekonstruálni az érzékeny információk jó részét.

Ez a négy kérdés nem igényel fejlett technikai ismereteket; olyan információkat kérnek, amelyekre minden őszinte operátor képes válaszolni a nyilvános dokumentációjában. A válasz minősége és pontossága legalább annyit elárul a termékről, mint maga a válasz.

A végpontok közötti titkosítás, ha jól csinálják, a kortárs kriptográfia egyik legfinomabb konstrukciója a mindennapi gyakorlat számára. Az eredeti ötlet — miszerint két ember megegyezhet egy titokban egy nyilvános csatornán keresztül — Whitfield Diffie és Martin Hellman nevéhez fűződik, 1976-ból; fél évszázaddal később még mindig ennek következményeiben élünk. De mint minden technikai ígéretnél, az értéke a valódi teljesítésen múlik, nem a címkén. A becsületes szakember kérdése nem az, hogy «titkosítva van-e?», hanem az, hogy «kinél vannak a kulcsok?». A válaszoknak eltérő következményei vannak. Érdemes ismerni őket.

Források és további olvasnivalók

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, 1976. november. A nyilvános kulcsú kriptográfia alapozó cikk.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, az Open Whisper Systems nyilvános specifikációja, 2016-os felülvizsgálat. A Signal protokoll és ipari származékainak alapja.
- RFC 7748 — Elliptic Curves for Security (IETF, 2016. január). A modern kulcscserékben használt X25519 és X448 görbék normatív specifikációja.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). Fejezetek a kulcscseréről és a hitelesített titkosítási protokollokról.
- (EU) 2024/1183 rendelet az európai digitális identitás keretrendszeréről (eIDAS 2) — olyan kereteket hoz létre, ahol a beszélgetőpartner független ellenőrzése intézményi támogatást kap, és ahol a névleges és a valós titkosítás megkülönböztetése eltérő jogi következményekkel jár.

← ElőzőKill switch és az intézményi kisajátításKövetkező →Az üzleti modell mint a bizalom jele

Legutóbbi olvasmányok

- Elemzés · 2026. május 18. Valódi vs. látszólagos adatvédelem: a kérdések, amelyeket érdemes feltenni
- Elemzés · 2026. május 18. Self-hosting mint szakmai gyakorlat
- Koncepció · 2026. május 18. A 24 szó: mi az a kriptográfiai identitás

Vigye magával ezt a cikket, ahová csak szüksége van rá.

↓ Markdown ↓ Egyszerű szöveg ↓ PDF

A fájl letöltődik az Ön eszközére. Onnan elmentheti, importálhatja a Solo2-be, vagy megoszthatja bárhol. A Cuadernos nem dönt Ön helyett a fájl sorsáról.

Viaszpecsét · SHA-256 425fce883b0f12b95b6d90870be8ad40fc32a08cc8dca57bcb1f597547f22b28

Cuadernos Lacre · A Menzuri Gestión S.L. kiadványa ·
írta R.Eugenio · szerkesztette a Solo2 csapata.