Criptarea end-to-end, explicată cu adevărat

Ce spun furnizorii când spun E2EE și ce nu spun. O explicație didactică a mecanismului și a limitelor sale, fără ambalajul publicitar.

Să ne înțelegem: WhatsApp spune că mesajele tale sunt criptate end-to-end. Este adevărat — și nu este suficient. Dacă backup-ul merge în iCloud sau Google Drive fără criptare suplimentară, criptarea este compromisă pe propriul tău telefon. Întrebarea operațională nu este dacă e criptat, ci unde se află cheile.

Ce înseamnă criptarea, cu adevărat

Criptarea unui mesaj înseamnă transformarea acestuia în ceva ce seamănă cu zgomotul pentru oricine nu posedă o anumită informație numită cheie. Operațiunea se face pe dispozitivul celui care trimite și, cu cheia corectă, este anulată pe dispozitivul celui care primește. Între cele două, mesajul călătorește ca o succesiune de octeți fără nicio semnificație aparentă. Aceasta este ideea simplă. Restul articolului se ocupă de nuanțele care o transformă, după caz, într-o garanție reală sau într-o etichetă de marketing.

Adjectivul *end-to-end* — în română *de la un capăt la altul*, abreviat E2EE — adaugă o precizie. Criptarea nu se face pentru ca un server intermediar să o poată citi și livra. Se face astfel încât numai cele două capete — dispozitivul celui care trimite și dispozitivul celui care primește — să posede cheia. Orice server prin care trece mesajul vede zgomotul, nu mesajul. Aceasta este diferența tehnică față de criptarea *în tranzit*, unde conținutul călătorește criptat de la un server la altul, dar fiecare server prin care trece îl decriptează pentru a-l retrimite, recuperând temporar textul în clar.

Paradoxul secretului partajat

Există o problemă evidentă. Pentru ca două persoane să poată cripta și decripta mesaje între ele, ambele au nevoie de aceeași cheie. Dar cum se pun de acord asupra acestei chei dacă tot ceea ce își trimit, prin definiție, trece printr-un canal unde cineva ar putea asculta? Convenirea asupra cheii pe același canal unde o vor folosi ulterior pare imposibilă: dacă atacatorul o aude la convenire, va putea decripta tot ce urmează. Timp de decenii, criptografia clasică a rezolvat acest lucru pe calea grea: cheile erau livrate personal, înainte de a începe să fie folosite, în întâlniri fizice. Ambasadorii purtau serviete cu chei cusute în căptușeala hainei.

În poșta electronică contemporană, acea soluție nu este scalabilă. Dacă ar trebui să mergem fizic la casa fiecărei persoane cu care intenționăm să comunicăm criptat, nu am mai ajunge să vorbim cu nimeni. Întrebarea pusă acum cincizeci de ani de comunitatea criptografică a fost aceasta: este posibil ca două persoane care nu se cunosc și care împart doar un canal public să convină, pe acel același canal public, asupra unui secret pe care nimeni care ascultă canalul să nu îl poată cunoaște?

Eleganța Diffie-Hellman

În 1976, doi matematicieni numiți Whitfield Diffie și Martin Hellman au demonstrat ceva aparent imposibil: că două persoane, vorbind doar printr-un canal public — un canal unde oricine poate auzi tot ce spun —, pot conveni asupra unei parole secrete fără ca vreun ascultător să o poată descoperi. Sună a magie. Nu este: este matematică. Schimbul de chei Diffie-Hellman, așa cum este cunoscut de atunci, este baza pentru practic toată comunicarea criptată de pe internet, iar o jumătate de secol de utilizare intensivă și control academic mondial îi confirmă soliditatea. Cine dorește să vadă intuiția vizuală sau matematica poate continua lectura. Cine preferă să aibă încredere că funcționează poate, de asemenea, să continue fără a pierde firul articolului.

Pentru cine dorește să o intuiască într-o imagine, există o analogie cunoscută cu culori. Imaginează-ți că Alice și Bruno convin deschis asupra unei culori de bază — să zicem galben — sub ochii Evei, care îi ascultă. Fiecare alege în privat o a doua culoare secretă și își amestecă secretul cu galbenul. Alice obține un portocaliu particular; Bruno obține un verde particular. Schimbă rezultatele sub ochii Evei. Acum fiecare amestecă culoarea primită cu propriul secret și amândoi ajung la aceeași culoare finală, deoarece ordinea amestecurilor nu contează. Eva a văzut galbenul și cele două amestecuri intermediare, dar nu și secretele; fără unul dintre secrete nu poate ajunge la culoarea finală. Matematica reală schimbă culorile cu exponențieri în grupuri modulare sau curbe eliptice, dar ideea este aceeași: secretul partajat este construit în public fără ca cineva de pe canal să îl poată reconstrui.

În aritmetică, pentru cine preferă să vadă mecanismul: Alice alege un număr secret a, Bruno alege b. Schimbă g^a și g^b la vedere pe canal. Alice calculează $(g^b)^a$ și Bruno calculează $(g^a)^b$; amândoi ajung la același g^{ab}. Eva vede g, g^a și g^b trecând prin canal, dar recuperarea lui a din g^a — așa-numita problemă a logaritmului discret — necesită un timp de calcul astronomic superior vârstei universului atunci când g este ales într-un grup matematic adecvat.

Pentru cine vrea să verifice cu numere mici. Schimbul Diffie-Hellman poate fi parcurs integral cu cifre suficient de mici pentru a face calculele manual. Cine preferă să nu intre în aritmetică poate sări acest bloc fără a pierde firul articolului; cine vrea să vadă mecanismul funcționând pas cu pas îl va găsi aici. **Regulile publice**, pe care oricine le poate citi: un număr prim $p = 11$ (în Diffie-Hellman-ul real este de

vreo trei sute de cifre; folosim unsprezece pentru ca acele calcule să încapă pe o pagină), o bază $g = 2$, și convenția că toată aritmetica se face *modulo p* — se calculează, se împarte la p, și se păstrează restul, ca un ceas cu unsprezece poziții care revine la zero când trece de zece. **Alegerile private**, câte una pentru fiecare și niciodată împărtășite: Alice alege $a = 4$. Bruno alege $b = 7$.

Pasul 1. Alice calculează $2^4 = 16$, apoi 16 mod 11 = **5**. Trimite cinciul. Eva îl notează.

Pasul 2. Bruno calculează $2^7 = 128$, apoi 128 mod 11 = **7**. Trimite șaptele. Eva îl notează și ea. După cele două trimiteri, carnețelul Evei conține patru date: $p = 11$, $g = 2$, A = 5, B = 7. Îi lipsește numărul partajat pe care Alice și Bruno urmează să-l derive — și pe care Eva nu-l va putea reconstrui.

Pasul 3. Alice ia șaptele pe care Bruno i l-a trimis și îl ridică la exponentul său privat $a = 4$. Pentru a evita manevrarea lui $7^4 = 2401$, se calculează pe părți aplicând modulo la fiecare pas:

$7^2 = 49$

49 mod 11 = 5

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

25 mod 11 = 3

Alice obține numărul **3**.

Pasul 4. Bruno ia cinciul pe care Alice i l-a trimis și îl ridică la exponentul său privat $b = 7$. Din nou pe părți:

$5^2 = 25$ mod 11 = 3

$5^4 = (5^2)^2 = 3^2 = 9$ mod 11 = 9

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

În final $5^7 = 5^6 \times 5 = 5 \times 5 = 25$ mod 11 = **3**.

Bruno obține de asemenea **3**.

Amândoi au ajuns la același număr, 3, lucrând în paralel. Niciunul nu și-a trimis exponentul privat în niciun moment. Alice nu știe că $b = 7$; Bruno nu știe că $a = 4$. Fiecare a folosit valoarea publică pe care a trimis-o celălalt combinată cu propriul exponent privat, și s-au întâlnit la aceeași destinație. **De ce ajung la același număr?** Ceea ce a calculat fiecare: Alice, $(g^b)^a = 2^{7\times4} = 2^{28}$ mod 11. Bruno, $(g^a)^b = 2^{4\times7} = 2^{28}$ mod 11. Este aceeași cantitate pentru că ordinea de înmulțire a exponenților nu contează ($7 \times 4 = 4 \times 7$). Fiecare a ajuns pe un drum diferit la aceeași destinație.

Și Eva? Are în carnețelul ei $p = 11$, $g = 2$, A = 5, B = 7, și ar dori 3-ul. Pentru a-l calcula ar trebui să cunoască a sau b — dar niciunul nu a călătorit prin canal. Singura ei cale este să se întrebe: «pentru ce exponent a se îndeplinește 2^a mod 11 = 5?». Cu p atât de mic poate încerca 0, 1, 2, 3, 4... și să-l găsească în mai puțin de un minut. Dar la înlocuirea lui 11 cu un număr prim de trei sute de cifre, spațiul exponenților posibili are mai multe elemente decât atomi există în universul observabil. **Nu există în prezent niciun algoritm cunoscut de umanitate care să poată parcurge acel spațiu în mai puțin de miliarde de ani.** Este așa-numita *problemă a logaritmului discret*: ușor înainte, computațional imposibil înapoi. Și este motivul pentru care criptarea rezistă chiar dacă Eva a urmărit toată conversația literă cu literă.

Trei ingrediente simple —aritmetica pe un ceas, exponențierea, și comutativitatea înmulțirii ($a \cdot b = b \cdot a$)— combinate produc un protocol de care jumătate din umanitate depinde în fiecare zi pentru comunicările lor private. Niciuna dintre cele trei piese, luate separat, nu pare specială. Ceea ce este decisiv este asamblarea.

De la Diffie-Hellman la protocolul Signal

Criptarea end-to-end pe care o folosesc astăzi aplicațiile de mesagerie profesională se bazează, aproape fără excepție, pe o versiune elegantă și întărită a schimbului Diffie-Hellman. Protocolul Signal, proiectat de Trevor Perrin și Moxie Marlinspike între 2013 și 2016, este referința. Combină două idei cheie. Prima, schimbul de chei în curbe eliptice (X25519), care produce secretul partajat inițial între două dispozitive. A doua, așa-numitul Double Ratchet — clichet dublu —, care reînnoiește cheile automat cu fiecare mesaj, astfel încât compromiterea dispozitivului astăzi nu permite decriptarea mesajelor trecute, nici a mesajelor viitoare odată ce clichetul a fost rotit.

În Zig, schimbul X25519 care produce secretul partajat între două dispozitive începe în șase linii, folosind biblioteca standard:

```
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)
```

Ce se întâmplă în acele șase linii: Cheile publice călătoresc deschis. Cheile private nu părăsesc niciodată dispozitivul respectiv. Fiecare parte derivă, pornind de la cheia sa privată și cheia publică a celeilalte, un același secret de treizeci și doi de octeți pe care nimeni de pe canal nu îl poate recupera. Acel secret servește ulterior ca sămânță pentru a cripta mesajele schimbate. Double Ratchet din protocolul Signal adaugă o rotație constantă a acelui material pentru ca compromiterea unui instantaneu să nu compromită restul conversației.

Și ce se află mai exact în interiorul `std.crypto.dh.X25519`? Nicio magie ascunsă. Sunt două funcții scurte care pot fi citite în întregime în propria bibliotecă standard din Zig. Prima derivă cheia publică din cea privată — «g^a»-ul schimbului:

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

În limbajul articolului: cheia privată este «înmulțită» — în sensul eliptic, nu în cel aritmetic elementar — cu punctul de bază al curbei Curve25519, iar rezultatul este serializat în treizeci și doi de octeți. Operațiunea `clampedMul` este versiunea întărită a acelei înmulțiri scalare: încorporează măsurile de siguranță pe care comunitatea criptografică le-a adăugat de-a lungul anilor pentru a rezista la familiile cunoscute de atacuri. Două rânduri de corp de funcție.

A doua funcție combină cheia ta privată cu cheia publică pe care ți-o trimite cealaltă parte. Este «$(g^b)^a$»-ul schimbului, cel care produce secretul partajat de treizeci și doi de octeți pe care niciunul dintre voi nu a ajuns să-l transmită:

```
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();
}
```

Alte două rânduri. Cheia publică primită este interpretată ca un punct pe curbă și este «înmulțită» cu propria cheie privată. Prin comutativitatea operațiunii pe curbă — analogă cu comutativitatea înmulțirii exponenților pe care am văzut-o în exemplul numeric —, ambele părți ajung la același punct serializat: exact secretul partajat despre care vorbește articolul.

Asta e tot. Ceea ce într-o aplicație pare magie este, în realitate, două funcții de trei rânduri fiecare. Complexitatea tehnică se concentrează într-o singură operațiune, `clampedMul`, care este scrisă mai jos în aceeași bibliotecă standard, revizuită de decenii de comunitatea criptografică internațională, și disponibilă pentru oricine vrea să o citească literă cu literă. Nu există nicio cutie neagră nici în aplicația noastră, nici în biblioteca standard din Zig. Există cod open source pe care un om îl poate înțelege, alegându-și ritmul în care vrea să aprofundeze.

Ce protejează criptarea end-to-end

Ceea ce E2EE protejează bine, presupunând o implementare corectă, este conținutul mesajului în tranzit. Un server intermediar care primește și retrimite datele criptate va vedea o succesiune de octeți neinteligibili. Un atacator cu acces la cablu, la router, la punctul de acces wifi, va vedea același lucru. Un furnizor de servicii care păstrează copii ale traficului nu îl va putea citi ulterior. Un Guvern care ordonă operatorului serviciului să livreze conținutul va primi aceiași octeți neinteligibili pe care serverul îi avea în primul rând.

Aceasta, în termeni practici, este mult. Este diferența între a scrie o scrisoare în interiorul unui plic opac și a o scrie pe o carte poștală. Ambele ajung. Numai una păstrează conținutul în fața poștașului.

Ce nu protejează criptarea end-to-end

Merită știut la fel de bine. E2EE nu protejează metadatele: serverul știe în continuare că utilizatorul A trimite date utilizatorului B, la ce oră, cu ce frecvență și de unde, deși nu știe ce spune. Aceste metadate, așa cum am argumentat deja în *A cripta nu înseamnă a fi privat*, sunt adesea mai revelatoare decât conținutul. A ști că cineva a sunat la un birou de avocatură specializat în divorțuri într-o vineri la ora 22:00 timp de treizeci de minute spune o poveste pe care conținutul apelului nu a spus-o niciodată. Este aceeași situație cu a vedea o persoană intrând și ieșind de mai multe ori dintr-o clinică oncologică: nu este nevoie să auzi nimic din ceea ce se vorbește înăuntru pentru a-ți imagina ce se întâmplă. O singură metadată izolată poate să nu însemne nimic; mai multe intersectate între ele desenează ceva mult prea asemănător cu adevărul. E2EE nu protejează capetele: dacă dispozitivul receptorului este compromis de un program malițios, mesajul este decriptat normal pentru acel receptor și programul malițios îl citește. E2EE nu protejează împotriva identității interlocutorului în sine: dacă Alice crede că vorbește cu Bruno, dar un atacator s-a interpus la început (un *man in the middle*) și protocolul nu include o verificare independentă, cele două părți ajung să vorbească cu intrusul crezând că vorbesc între ele.

Există un al patrulea lucru care merită formulat fără ambiguitate. E2EE nu împiedică un furnizor care afirmă că îl oferă să păstreze, în plus, o copie a mesajului necriptat în propriile sisteme. Afirmația «mesajele mele sunt criptate end-to-end» și afirmația «furnizorul nu păstrează conținutul meu» nu sunt aceleași. O aplicație poate îndeplini prima afirmație în timp ce o încalcă pe a doua; am văzut acest lucru în titlurile presei în mod repetat din 2018. Utilizatorul, cu excepția cazului în care codul clientului este verificabil, nu are nicio modalitate tehnică de a distinge un caz de celălalt fără o investigație expertă. Cazul cel mai cunoscut în rândul publicului larg: WhatsApp criptează mesajele end-to-end în tranzit, ale dar dacă utilizatorul activează copia de rezervă în iCloud sau Google Drive fără criptare suplimentară, acea copie este stocată lizibil în infrastructura unui terț, iar criptarea este ruptă la capătul utilizatorului însuși.

Întrebarea pe care operatorul nu vrea să o audă

O aplicație care afirmă că criptează end-to-end poate, tehnic, să facă unul din trei lucruri în ceea ce privește cheile:

1. **Cheile rezidă numai pe dispozitive.** Se generează și rezidă exclusiv pe dispozitivele utilizatorilor; operatorul nu le cunoaște și nu le stochează. Este cazul optim.
2. **Operatorul poate accesa dacă dorește.** Operatorul deține cheile utilizatorilor (sau le poate genera după bunul plac) și le păstrează în bazele sale de date. Dacă dorește sau este obligat, poate citi conținutul. Acesta este cazul majorității serviciilor „cloud”.
3. **Operatorul nu poate accesa prin design, dar controlează accesul.** Operatorul nu are cheile, dar are controlul asupra aplicației care le generează. Dacă este obligat, poate trimite o actualizare malițioasă care să capteze cheile sau conținutul înainte de criptare. Acesta este cazul multor servicii E2EE comerciale.

Prin urmare, întrebarea operativă nu este dacă ceva este criptat, ci cine deține controlul asupra dispozitivului și asupra software-ului care gestionează cheile. În Solo2, cheile rezidă exclusiv în Seiful tău (IndexedDB criptată cu parola ta), iar software-ul este cod sursă deschis verificabil.

Pentru cititorul profesionist

Criptarea end-to-end este un instrument pentru suveranitatea digitală. Dar, ca orice instrument, eficacitatea sa depinde de mâna care îl mânuiește și de solul pe care se sprijină.

1. Unde sunt generate cheile criptografice și unde rezidă fizic? Dacă operatorul poate accesa cheile (chiar și temporar, chiar și sub pretextul recuperării), E2EE-ul este doar nominal.
2. Există o verificare independentă a interlocutorului (numere de securitate, coduri QR, comparație out-of-band) care să prevină un atac man-in-the-middle în timpul stabilirii conversației?
3. Codul clientului poate fi auditat — este deschis, publicat, reproductibil — sau presupune să te încrezi pe cuvânt în furnizor cu privire la ceea ce face clientul în realitate?
4. Ce metadate generează și păstrează serviciul și pentru cât timp? Chiar dacă conținutul este opac, metadatele pot reconstrui o mare parte din informațiile sensibile.

Aceste patru întrebări nu cer informații tehnice avansate; ele cer informații la care orice operator onest poate răspunde în documentația sa publică. Calitatea și precizia răspunsului spun la fel de mult despre produs ca și răspunsul însuși.

Criptarea end-to-end, bine făcută, este una dintre cele mai fine construcții pe care criptografia contemporană a oferit-o practicii cotidiene. Ideea originală — două persoane pot conveni asupra unui secret pe un canal public — aparține lui Whitfield Diffie și Martin Hellman, 1976; o jumătate de secol mai târziu continuăm să trăim în consecința ei. Dar, așa cum se întâmplă cu orice promisiune tehnică, valoarea sa depinde de îndeplinirea reală, nu de etichetă. Întrebarea profesionistului onest nu este «este criptat?», ci «cine are cheile?». Răspunsurile au consecințe diferite. Merită să le cunoașteți.

Surse și lecturi suplimentare

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, noiembrie 1976. Articol fundamental privind criptografia cu cheie publică.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, specificație publică de la Open Whisper Systems, revizuire 2016. Baza protocolului Signal și a derivatelor sale industriale.
- RFC 7748 — Elliptic Curves for Security (IETF, ianuarie 2016). Specificația normativă a curbelor X25519 și X448 utilizate în schimburile moderne de chei.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). Capitole privind schimbul de chei și protocoalele de criptare autentificată.
- Regulamentul (UE) 2024/1183 privind un cadru european pentru o identitate digitală (eIDAS 2) — stabilește cadre în care verificarea independentă a interlocutorului dobândește sprijin instituțional și în care distincția dintre criptarea nominală și cea reală are consecințe juridice diferite.

← PrecedentKill switch și capturarea instituționalăUrmător →Modelul de afaceri ca semnal de încredere

Lecturi recente

- Analiză · 18 mai 2026 Confidențialitate reală vs. aparentă: întrebările pe care e bine să ți le pui
- Analiză · 18 mai 2026 Self-hosting ca practică profesională
- Concept · 18 mai 2026 Cele 24 de cuvinte: ce este o identitate criptografică

Luați acest articol cu dumneavoastră oriunde aveți nevoie.

↓ Markdown ↓ Text simplu ↓ PDF

Fișierul se va descărca pe dispozitivul dumneavoastră. De acolo îl puteți salva, importa în Solo2 sau partaja oriunde doriți. Cuadernos nu decide destinația în locul dumneavoastră.

Sigiliu de ceară · SHA-256 c4e883b262b2b15f2379fe05f8bcf088b5624f06b1d1606fe399faef2f1796bb

Cuadernos Lacre · O publicație a Menzuri Gestión S.L. ·
scrisă de R.Eugenio · editată de echipa Solo2.

Acest site nu folosește cookie-uri și nu încarcă resurse de la terți. Folosește un contor anonim de vizite găzduit (Umami, pe serverul nostru european) și minimul de JavaScript necesar pentru cele două controale din antet: temă deschisă sau închisă și selector de limbă. Fără trackere, fără profilare, fără partajarea datelor. Dacă doriți să ne urmăriți: RSS.