Ende-til-ende-kryptering, forklart på ekte

Hva tilbydere sier når de sier E2EE, og hva de ikke sier. En didaktisk forklaring av mekanismen og dens grenser, uten reklameinnpakning.

La oss være klare: WhatsApp sier at meldingene dine er ende-til-ende-krypterte. Det er sant — og det er ikke nok. Hvis sikkerhetskopien går til iCloud eller Google Drive uten ytterligere kryptering, brytes krypteringen på din egen telefon. Det operative spørsmålet er ikke om det er kryptert, men hvor nøklene befinner seg.

Hva kryptering egentlig betyr

Å kryptere en melding betyr å transformere den til noe som ser ut som støy for alle som ikke besitter en bestemt informasjon kalt en nøkkel. Operasjonen gjøres på enheten til den som sender, og med riktig nøkkel omgjøres den på enheten til den som mottar. Imellom reiser meldingen som en rekke bytes uten tilsynelatende mening. Dette er den enkle ideen. Resten av artikkelen tar for seg nyansene som gjør den, alt etter tilfelle, til en reell garanti eller en markedsføringsetikett.

Adjektivet *ende-til-ende* — på engelsk *end-to-end*, forkortet E2EE — legger til en presisjon. Krypteringen gjøres ikke for at en mellomliggende server skal kunne lese og levere den. Den gjøres for at kun de to endene — enheten til den som sender og enheten til den som mottar — skal besitte nøkkelen. Enhver server som meldingen passerer gjennom, ser støyen, ikke meldingen. Dette er den tekniske forskjellen fra kryptering *i transitt*, der innholdet reiser kryptert fra én server til den neste, men hver server den passerer dekrypterer den for å videresende den, og gjenoppretter midlertidig teksten i klartekst.

Paradokset om den delte hemmeligheten

Det er et åpenbart problem. For at to personer skal kunne kryptere og dekryptere meldinger seg imellom, trenger begge den samme nøkkelen. Men hvordan blir de enige om denne nøkkelen hvis alt de sender hverandre, per definisjon, går gjennom en kanal der noen kan lytte? Å avtale nøkkelen i den samme kanalen som de senere skal bruke den i, virker umulig: hvis angriperen hører den ved avtalen, vil vedkommende kunne dekryptere alt etterfølgende. I tiår løste klassisk kryptografi dette på den harde måten: nøklene ble levert personlig, før de ble tatt i bruk, i fysiske møter. Ambassadører bar kofferter med nøkler sydd inn i foret på frakkene sine.

I moderne e-post er ikke den løsningen skalerbar. Hvis vi måtte gå fysisk hjem til hver person vi hadde til hensikt å kommunisere kryptert med, ville vi aldri komme til å snakke med noen. Spørsmålet som ble stilt for femti år siden av det kryptografiske miljøet var dette: er det mulig for to personer som ikke kjenner hverandre og som bare deler en offentlig kanal, å avtale en hemmelighet i den samme offentlige kanalen, som ingen som lytter til kanalen kan vite om?

Elegansen i Diffie-Hellman

I 1976 demonstrerte to matematikere ved navn Whitfield Diffie og Martin Hellman noe tilsynelatende umulig: at to personer, som bare snakker gjennom en offentlig kanal — en kanal der hvem som helst kan høre alt de sier — kan bli enige om et hemmelig passord uten at noen lytter kan oppdage det. Det høres ut som magi. Det er det ikke: det er matematikk. Diffie-Hellman-nøkkelutveksling, som det har vært kjent som siden da, er grunnlaget for praktisk talt all kryptert kommunikasjon på internett, og et halvt århundre med intensiv bruk og global akademisk gransking bekrefter soliditeten. De som vil se den visuelle intuisjonen eller matematikken, kan lese videre. De som foretrekker å stole på at det fungerer, kan også fortsette uten å miste tråden i artikkelen.

For de som vil visualisere det, finnes det en kjent analogi med farger. Tenk deg at Alice og Bruno blir enige om en grunnfarge offentlig — la oss si gult — foran øynene på Eva, som lytter til dem. Hver velger en annen hemmelig farge privat og blander sin hemmelighet med den gule. Alice får en bestemt oransje; Bruno får en bestemt grønn. De utveksler resultatene foran øynene på Eva. Nå blander hver den mottatte fargen med sin egen hemmelighet, og begge når frem til den samme sluttfargen, fordi rekkefølgen på blandingene ikke betyr noe. Eva har sett det gule og de to mellomliggende blandingene, men ikke hemmelighetene; uten en av hemmelighetene kan hun ikke nå frem til sluttfargen. Den virkelige matematikken erstatter fargene med eksponentiering i modulære grupper eller elliptiske kurver, men ideen er den samme: den delte hemmeligheten bygges offentlig uten at noen i kanalen kan rekonstruere den.

I aritmetikk, for de som foretrekker å se mekanismen: Alice velger et hemmelig tall a, Bruno velger b. De utveksler g^a og g^b åpent over kanalen. Alice beregner $(g^b)^a$ og Bruno beregner $(g^a)^b$; begge når frem til samme g^{ab}. Eva ser g, g^a og g^b passere gjennom kanalen, men å gjenvinne a fra g^a — det såkalte diskrete logaritmeproblemet — krever en astronomisk beregningstid som overstiger universets alder når g velges i en egnet matematisk gruppe.

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 mod 11 = **5**. Envía el cinco. Eva lo anota.

Paso 2. Bruno calcula $2^7 = 128$, luego 128 mod 11 = **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 mod 11 = 5

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

25 mod 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:

$5^2 = 25$ 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

Finalmente $5^7 = 5^6 \times 5 = 5 \times 5 = 25$ mod 11 = **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, $(g^b)^a = 2^{7\times4} = 2^{28}$ mod 11. Bruno, $(g^a)^b = 2^{4\times7} = 2^{28}$ mod 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 mod 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.

Fra Diffie-Hellman til Signal-protokollen

Ende-til-ende-kryptering som brukes av dagens profesjonelle meldingsapper hviler, nesten uten unntak, på en elegant og herdet versjon av Diffie-Hellman-utvekslingen. Signal-protokollen, designet av Trevor Perrin og Moxie Marlinspike mellom 2013 og 2016, er referansen. Den kombinerer to nøkkelideer. Den første er nøkkelutveksling i elliptiske kurver (X25519), som produserer den opprinnelige delte hemmeligheten mellom to enheter. Den andre er den såkalte Double Ratchet — dobbelt tannhjul — som fornyer nøklene automatisk med hver melding, slik at kompromittering av enheten i dag ikke tillater dekryptering av tidligere meldinger, heller ikke fremtidige meldinger når tannhjulet er dreid.

I Zig tar X25519-utvekslingen som produserer den delte hemmeligheten mellom to enheter seks linjer, ved bruk av standardbiblioteket:

```
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)
```

Hva som skjer på de seks linjene: De offentlige nøklene reiser åpent. De private nøklene forlater aldri den respektive enheten. Hver part utleder, fra sin private og den andres offentlige, den samme hemmeligheten på trettito bytes som ingen i kanalen kan gjenvinne. Den hemmeligheten fungerer senere som frø for å kryptere de utvekslede meldingene. Signal-protokollens Double Ratchet legger til en konstant rotasjon av dette materialet slik at kompromittering av et øyeblikk ikke kompromitterer resten av samtalen.

Og hva nøyaktig er inne i `std.crypto.dh.X25519`? Ingen skjult magi. Det er to korte funksjoner som kan leses i sin helhet i Zigs eget standardbibliotek. Den første utleder den offentlige nøkkelen fra den private — utvekslingens «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();
}
```

I artikkelens språk: den private nøkkelen «multipliseres» — i elliptisk forstand, ikke i grunnleggende aritmetisk forstand — med basispunktet for Curve25519-kurven, og resultatet serialiseres til trettito bytes. Operasjonen `clampedMul` er den herdede versjonen av den skalare multiplikasjonen: den inkorporerer sikkerhetsmekanismene som det kryptografiske miljøet har lagt til gjennom årene for å motstå kjente familier av angrep. To linjer med funksjonskropp.

Den andre funksjonen kombinerer din private nøkkel med den offentlige nøkkelen som den andre parten sender deg. Det er utvekslingens «$(g^b)^a$», som produserer den trettito bytes store delte hemmeligheten som ingen av dere noen gang overførte:

```
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();
}
```

To linjer til. Den mottatte offentlige nøkkelen tolkes som et punkt på kurven, og «multipliseres» med ens egen private nøkkel. På grunn av kurveoperasjonens kommutative egenskap — analog til kommutativiteten i eksponentmultiplikasjonen som vi så i det numeriske eksemplet — ender begge parter opp med det samme serialiserte punktet: nøyaktig den delte hemmeligheten som artikkelen snakker om.

Det er alt. Det som i en applikasjon ser ut som magi er i virkeligheten to funksjoner på tre linjer hver. Den tekniske kompleksiteten er konsentrert i en enkelt operasjon, `clampedMul`, som er skrevet lenger ned i det samme standardbiblioteket, gjennomgått i flere tiår av det internasjonale kryptografiske samfunnet, og tilgjengelig for alle som ønsker å lese det bokstav for bokstav. Det er ingen svart boks hverken i applikasjonen vår eller i Zigs standardbibliotek. Det er åpen kildekode som et menneske kan forstå, og man kan selv velge tempoet man vil dykke ned i den med.

Hva ende-til-ende-kryptering beskytter

Det E2EE beskytter godt, forutsatt en korrekt implementering, er innholdet i meldingen under transitt. En mellomliggende server som mottar og videresender de krypterte dataene vil se en rekke uforståelige bytes. En angriper med tilgang til kabelen, ruteren, wifi-aksesspunktet vil se det samme. En tjenesteleverandør som oppbevarer kopier av trafikken vil ikke kunne lese den i ettertid. En regjering som pålegger tjenesteoperatøren å utlevere innholdet vil motta de samme uforståelige bytene som serveren hadde i utgangspunktet.

Dette er, i praktiske termer, mye. Det er forskjellen mellom å skrive et brev inni en ugjennomsiktig konvolutt og å skrive det på et postkort. Begge kommer frem. Bare én bevarer innholdet overfor postmannen.

Hva ende-til-ende-kryptering ikke beskytter

Det er verdt å vite det like godt. E2EE beskytter ikke metadata: serveren vet fortsatt at bruker A sender data til bruker B, på hvilket tidspunkt, med hvilken frekvens og hvorfra, selv om den ikke vet hva som blir sagt. Disse metadataene, som vi allerede har argumentert for i *Å kryptere er ikke å være privat*, er ofte mer avslørende enn innholdet. Å vite at noen ringte et advokatfirma spesialisert på skilsmisser en fredag kl. 22.00 i tretti minutter forteller en historie som innholdet i samtalen aldri fortalte. Det er samme situasjon som å se en person gå inn og ut av en onkologisk klinikk flere ganger: man trenger ikke høre noe av det som blir sagt inne for å forestille seg hva som skjer. Ett enkelt isolert metadatakunne betyr kanskje ingenting; flere kryssrefererte tegner noe som er altfor likt sannheten. E2EE beskytter ikke endepunktene: hvis mottakerens enhet er kompromittert av et ondsinnet program, dekrypteres meldingen normalt for den mottakeren og det ondsinnede programmet leser den. E2EE beskytter ikke mot identiteten til samtalepartneren i seg selv: hvis Alice tror hun snakker med Bruno, men en angriper har skutt seg inn i starten (en *man in the middle*) og protokollen ikke inkluderer uavhengig verifisering, ender de to partene med å snakke med inntrengeren i troen på at de snakker med hverandre.

Det er en fjerde ting som er verdt å formulere uten tvetydighet. E2EE hindrer ikke en leverandør som hevder å tilby det fra å også lagre en kopi av den ukrypterte meldingen i sine egne systemer. Påstanden «mine meldinger er ende-til-ende-kryptert» og påstanden «leverandøren lagrer ikke mitt innhold» er ikke de samme. En app kan oppfylle den første mens den bryter den andre; vi har sett det i avisoverskrifter gjentatte ganger siden 2018. Brukeren har, med mindre klientens kode er verifiserbar, ingen teknisk måte å skille ett tilfelle fra det andre uten ekspertundersøkelse. Det mest kjente tilfellet i den brede offentligheten: WhatsApp krypterer meldinger ende-til-ende under transitt, men hvis brukeren aktiverer sikkerhetskopi i iCloud eller Google Drive uten ytterligere kryptering, lagres den kopien lesbart i en tredjeparts infrastruktur, og krypteringen brytes i brukerens egen ende.

Spørsmålet operatøren ikke ønsker å høre

En app som hevder å kryptere ende-til-ende kan teknisk sett gjøre én av tre ting med hensyn til nøklene:

1. **Nøklene bor kun på enhetene.** De genereres og bor utelukkende på brukernes enheter; operatøren kjenner dem ikke og lagrer dem ikke. Dette er det optimale tilfellet.
2. **Operatøren kan få tilgang hvis de vil.** Operatøren har brukernes nøkler (eller kan generere dem etter eget ønske) og lagrer dem i sine databaser. Hvis de vil eller blir tvunget til det, kan de lese innholdet. Dette er tilfellet for de fleste «skybaserte» tjenester.
3. **Operatøren kan ikke få tilgang ved design, men de kontrollerer tilgangen.** Operatøren har ikke nøklene, men har kontroll over applikasjonen som genererer dem. Hvis de blir tvunget, kan de sende en ondsinnet oppdatering som fanger opp nøklene eller innholdet før kryptering. Dette er tilfellet for mange kommersielle E2EE-tjenester.

Det operative spørsmålet er derfor ikke om noe er kryptert, men hvem som har kontroll over enheten og programvaren som administrerer nøklene. I Solo2 befinner nøklene seg utelukkende i ditt Hvelv (IndexedDB kryptert med ditt passord), og programvaren er verifiserbar åpen kildekode.

For den profesjonelle leseren

Ende-til-ende-kryptering er et verktøy for digital suverenitet. Men som ethvert verktøy, avhenger effektiviteten av hånden som fører det og grunnen det står på.

1. Hvor genereres de kryptografiske nøklene, og hvor befinner de seg fysisk? Hvis operatøren kan få tilgang til dem (selv midlertidig, selv under påskudd av gjenoppretting), er E2EE kun nominell.
2. Finnes det uavhengig verifisering av samtalepartneren (sikkerhetsnumre, QR-koder, out-of-band sammenligning) som forhindrer et man-in-the-middle-angrep under etableringen av samtalen?
3. Kan klientkoden revideres — er den åpen, publisert, reproduserbar — eller krever det at vi stoler på leverandørens ord om hva klienten faktisk gjør?
4. Hvilke metadata genererer og lagrer tjenesten, og for hvor lenge? Selv om innholdet er ugjennomsiktig, kan metadata rekonstruere en god del av den sensitive informasjonen.

Disse fire spørsmålene ber ikke om avansert teknisk informasjon; de ber om informasjon som enhver ærlig operatør kan svare på i sin offentlige dokumentasjon. Kvaliteten og presisjonen av svaret sier like mye om produktet som selve svaret.

Ende-til-ende-kryptering, når det er gjort riktig, er en av de fineste konstruksjonene som moderne kryptografi har levert til daglig praksis. Den opprinnelige ideen — at to personer kan bli enige om en hemmelighet via en offentlig kanal — tilhører Whitfield Diffie og Martin Hellman, 1976; et halvt århundre senere lever vi fortsatt i konsekvensen av den. Men som med ethvert teknisk løfte, avhenger verdien av reell oppfyllelse, ikke av etiketten. Den ærlige fagpersonens spørsmål er ikke «er det kryptert?», men «hvem har nøklene?». Svarene har ulike konsekvenser. Det er verdt å kjenne dem.

Kilder og videre lesing

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, november 1976. Grunnleggende artikkel om offentlig nøkkel-kryptografi.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, offentlig spesifikasjon fra Open Whisper Systems, revisjon 2016. Grunnlaget for Signal-protokollen og dens industrielle derivater.
- RFC 7748 — Elliptic Curves for Security (IETF, januar 2016). Normativ spesifikasjon av X25519- og X448-kurvene som brukes i moderne nøkkelutvekslinger.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). Kapitler om nøkkelutveksling og autentiserte krypteringsprotokoller.
- Forordning (EU) 2024/1183 om rammeverk for europeisk digital identitet (eIDAS 2) — etablerer rammeverk der uavhengig verifisering av samtalepartneren får institusjonell støtte, og der skillet mellom nominell og reell kryptering har ulike juridiske konsekvenser.

← ForrigeKill switch og institusjonell fangstNeste →Forretningsmodellen som et signal om tillit

Siste lesninger

- Analyse · 18. mai 2026 Reelt vs. tilsynelatende personvern: Spørsmålene man bør stille seg selv
- Analyse · 18. mai 2026 Self-hosting som profesjonell praksis
- Konsept · 18. mai 2026 De 24 ordene: hva en kryptografisk identitet er

Ta med deg denne artikkelen dit du trenger den.

↓ Markdown ↓ Klartekst ↓ PDF

Filen lastes ned til enheten din. Derfra kan du lagre den, importere den til Solo2 eller dele den hvor du vil. Cuadernos bestemmer ikke destinasjonen for deg.

Lakksegl · SHA-256 cfa41deb26ef81dfddeeb8fbc2f906be30da15d01dbdc1a73ba0e3addfa1d7cc

Cuadernos Lacre · En utgivelse fra Menzuri Gestión S.L. ·
skrevet av R.Eugenio · redigert av teamet bak Solo2.

Dette nettstedet bruker ikke informasjonskapsler og laster ikke inn tredjepartsressurser. Det bruker en selvhustet anonym besøksteller (Umami, på vår europeiske server) og minimum nødvendig JavaScript for de to kontrollene i headeren: lyst eller mørkt tema, og språkvelger. Ingen trackere, ingen profilering, ingen deling av data. Hvis du vil følge oss: RSS.