Ištisinis šifravimas, paaiškinta tikrai

Ką teikėjai sako, kai mini E2EE, ir ko jie nesako. Didaktinis mechanizmo ir jo ribų paaiškinimas be reklaminio apvalkalo.

Pabūkime aiškūs: „WhatsApp“ sako, kad jūsų pranešimai yra užšifruoti ištisiniu būdu. Tai tiesa — ir to nepakanka. Jei atsarginė kopija nukeliauja į „iCloud“ ar „Google Drive“ be papildomo šifravimo, šifravimas pažeidžiamas jūsų pačių telefone. Operatyvinis klausimas yra ne tai, ar jis užšifruotas, o kur yra raktai.

Ką šifravimas reiškia iš tikrųjų

Užšifruoti pranešimą reiškia paversti jį tuo, kas atrodo kaip triukšmas bet kam, kas neturi tam tikros informacijos, vadinamos raktu. Operacija atliekama siuntėjo įrenginyje, o su teisingu raktu ji atšaukiama gavėjo įrenginyje. Per vidurį pranešimas keliauja kaip baitų seka be akivaizdžios prasmės. Tai paprasta idėja. Likusi straipsnio dalis skirta niuansams, kurie, priklausomai nuo atvejo, paverčia tai tikra garantija arba rinkodaros etikete.

Būdvardis *ištisinis* — angliškai *end-to-end*, sutrumpintai E2EE — suteikia tikslumo. Šifravimas neatliekamas tam, kad tarpinis serveris galėtų jį perskaityti ir pristatyti. Jis atliekamas tam, kad tik du galai — siuntėjo įrenginys ir gavėjo įrenginys — turėtų raktą. Bet kuris serveris, per kurį pranešimas praeina, mato triukšmą, o ne pranešimą. Tai techninis skirtumas nuo šifravimo *tranzitu*, kai turinys keliauja užšifruotas iš vieno serverio į kitą, tačiau kiekvienas serveris, per kurį jis praeina, jį iššifruoja, kad galėtų persiųsti, laikinai atkurdamas atvirą tekstą.

Bendros paslapties paradoksas

Yra akivaizdi problema. Kad du žmonės galėtų tarpusavyje šifruoti ir iššifruoti pranešimus, abiems reikia to paties rakto. Tačiau kaip jie susitaria dėl šio rakto, jei viskas, ką jie siunčia vienas kitam, pagal apibrėžimą eina per kanalą, kuriame kažkas gali klausytis? Susitarti dėl rakto tame pačiame kanale, kuriame jie vėliau jį naudos, atrodo neįmanoma: jei užpuolikas jį išgirs susitarimo metu, jis galės iššifruoti viską, kas bus vėliau. Dešimtmečius klasikinė kriptografija tai sprendė sunkiuoju būdu: raktai buvo perduodami asmeniškai prieš pradedant naudoti fizinių susitikimų metu. Ambasadoriai nešiojosi lagaminėlius su raktais, įsiūtais į palto pamušalą.

Šiuolaikiniame el. pašte šis sprendimas nėra mastelis. Jei turėtume fiziškai eiti į namus kiekvienam asmeniui, su kuriuo ketiname bendrauti užšifruotai, nespėtume su niekuo pasikalbėti. Klausimas, kurį kriptografijos bendruomenė iškėlė prieš penkiasdešimt metų, buvo toks: ar įmanoma, kad du žmonės, kurie nepažįsta vienas kito ir kurie dalijasi tik viešu kanalu, tame pačiame viešame kanale susitartų dėl paslapties, kurios niekas, klausantis kanalo, negalėtų žinoti?

Diffie-Hellman elegancija

1976 m. du matematikai, vardu Whitfield Diffie ir Martin Hellman, įrodė kažką, kas atrodė neįmanoma: kad du žmonės, kalbėdami tik per viešą kanalą — kanalą, kuriame bet kas gali girdėti viską, ką jie sako — gali susitarti dėl slapto slaptažodžio be jokio klausytojo galimybės jį atrasti. Tai skamba kaip magija. Tai nėra magija: tai matematika. Diffie-Hellman apsikeitimas raktais, kaip jis žinomas nuo tada, yra praktiškai viso užšifruoto interneto ryšio pagrindas, o pusė amžiaus intensyvaus naudojimo ir pasaulinio akademinio tikrinimo patvirtina jo tvirtumą. Kas nori pamatyti vizualinę intuiciją ar matematiką, gali skaityti toliau. Kas nori pasitikėti, kad tai veikia, taip pat gali tęsti neprarasdamas straipsnio gijos.

Tiems, kurie nori tai įsivaizduoti paveikslėlyje, yra žinoma analogija su spalvomis. Įsivaizduokite, kad Alisa ir Brunas viešai susitaria dėl pagrindinės spalvos — sakykime, geltonos — matant Evai, kuri jų klausosi. Kiekvienas privačiai pasirenka antrą slaptą spalvą ir sumaišo savo paslaptį su geltona. Alisa gauna tam tikrą oranžinę spalvą; Brunas gauna tam tikrą žalią spalvą. Jie apsikeičia rezultatais matant Evai. Dabar kiekvienas sumaišo gautą spalvą su savo paslaptimi, ir abu gauna tą pačią galutinę spalvą, nes maišymo eiliškumas nėra svarbus. Eva matė geltoną spalvą ir du tarpinius mišinius, bet ne paslaptis; be kurios nors iš paslapčių ji negali gauti galutinės spalvos. Tikroji matematika spalvas pakeičia kėlimu laipsniu modulinėse grupėse arba elipsinėse kreivėse, tačiau idėja yra ta pati: bendra paslaptis sukuriama viešai be niekieno galimybės kanale jos rekonstruoti.

Aritmetikoje, tiems, kurie nori pamatyti mechanizmą: Alisa pasirenka slaptą skaičių a, Brunas pasirenka b. Jie apsikeičia g^a ir g^b atvirai kanale. Alisa apskaičiuoja $(g^b)^a$, o Brunas apskaičiuoja $(g^a)^b$; abu gauna tą patį g^{ab}. Eva mato g, g^a ir g^b einant kanalu, tačiau atkurti a iš g^a — vadinamoji diskretaus logaritmo problema — reikalauja astronominio skaičiavimo laiko, viršijančio visatos amžių, kai g pasirenkamas tinkamoje matematinėje grupėje.

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.

Nuo Diffie-Hellman iki Signal protokolo

Ištisinis šifravimas, kurį šiandien naudoja profesionalios pranešimų programėlės, beveik be išimčių remiasi elegantiška ir sutvirtinta Diffie-Hellman apsikeitimo versija. Signal protokolas, kurį 2013–2016 m. sukūrė Trevor Perrin ir Moxie Marlinspike, yra etalonas. Jis sujungia dvi pagrindines idėjas. Pirmoji — apsikeitimas raktais elipsinėse kreivėse (X25519), kuris sukuria pradinę bendrą paslaptį tarp dviejų įrenginių. Antroji — vadinamasis Double Ratchet — dvigubas krumpliastiebis — kuris automatiškai atnaujina raktus su kiekvienu pranešimu, todėl įrenginio kompromitavimas šiandien neleidžia iššifruoti praeities pranešimų, nei būsimų pranešimų, kai krumpliastiebis pasisuko.

Zig kalba X25519 apsikeitimas, sukuriantis bendrą paslaptį tarp dviejų įrenginių, telpa į šešias eilutes, naudojant standartinę biblioteką:

```
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)
```

Kas vyksta tose šešiose eilutėse: Viešieji raktai keliauja atvirai. Privatūs raktai niekada nepalieka atitinkamo įrenginio. Kiekviena šalis iš savo privačiojo ir kitos šalies viešojo rakto išveda tą pačią trisdešimt dviejų baitų paslaptį, kurios niekas kanale negali atkurti. Ši paslaptis vėliau tarnauja kaip pradinė reikšmė apsikeistiems pranešimams šifruoti. Signal protokolo Double Ratchet prideda nuolatinę šios medžiagos rotaciją, kad vienos akimirkos kompromitavimas nesukeltų pavojaus likusiam pokalbiui.

O kas tiksliai yra `std.crypto.dh.X25519` viduje? Jokios paslėptos magijos. Tai dvi trumpos funkcijos, kurias visas galima perskaityti pačioje „Zig“ standartinėje bibliotekoje. Pirmoji išveda viešąjį raktą iš privataus — mainų «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();
}
```

Straipsnio kalba: privatus raktas «padauginamas» — elipsine, ne elementaria aritmetine prasme — iš „Curve25519“ kreivės bazinio taško, o rezultatas serializuojamas į trisdešimt du baitus. `clampedMul` operacija yra sustiprinta tokios skaliarinės daugybos versija: ji apima apsaugos priemones, kurias kriptografijos bendruomenė pridėjo per daugelį metų, kad atsispirtų žinomoms atakų šeimoms. Dvi funkcijos kūno eilutės.

Antroji funkcija sujungia jūsų privatų raktą su viešuoju raktu, kurį jums siunčia kita šalis. Tai yra mainų «$(g^b)^a$», kuris sukuria trisdešimt dviejų baitų bendrą paslaptį, kurios nė vienas iš jūsų niekada neperdavė:

```
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();
}
```

Dar dvi eilutės. Gautas viešasis raktas interpretuojamas kaip taškas kreivėje ir «padauginamas» iš savo asmeninio privataus rakto. Dėl kreivės operacijos komutatyvumo — analogiško eksponentų daugybos komutatyvumui, kurį matėme skaitiniame pavyzdyje — abi šalys gauna tą patį serializuotą tašką: būtent bendrą paslaptį, apie kurią kalbama straipsnyje.

Štai ir viskas. Tai, kas programėlėje atrodo kaip magija, iš tikrųjų yra dvi funkcijos, kiekviena iš trijų eilučių. Techninis sudėtingumas sutelktas vienoje operacijoje, `clampedMul`, kuri parašyta toliau toje pačioje standartinėje bibliotekoje, dešimtmečius peržiūrima tarptautinės kriptografijos bendruomenės ir prieinama kiekvienam, kuris nori ją perskaityti raidė po raidės. Nei mūsų programėlėje, nei „Zig“ standartinėje bibliotekoje nėra juodosios dėžės. Yra atvirasis kodas, kurį žmogus gali suprasti, pasirinkdamas tempą, kuriuo nori į jį gilintis.

Ką saugo ištisinis šifravimas

Tai, ką E2EE gerai saugo, darant prielaidą, kad įgyvendinimas teisingas, yra pranešimo turinys tranzito metu. Tarpinis serveris, kuris gauna ir persiunčia užšifruotus duomenis, matys nesuprantamų baitų seką. Užpuolikas, turintis prieigą prie kabelio, maršrutizatoriaus ar wifi prieigos taško, matys tą patį. Paslaugų teikėjas, saugantis srauto kopijas, negalės jų perskaityti vėliau. Vyriausybė, nurodžiusi paslaugos operatoriui pateikti turinį, gaus tuos pačius nesuprantamus baitus, kuriuos serveris turėjo iš pradžių.

Tai praktine prasme yra labai daug. Tai skirtumas tarp laiško rašymo nepermatomame voke ir jo rašymo atviruke. Abu pasiekia tikslą. Tik vienas išsaugo turinį nuo paštininko.

Ko nesaugo ištisinis šifravimas

Verta tai žinoti taip pat gerai. E2EE nesaugo metaduomenų: serveris vis tiek žino, kad vartotojas A siunčia duomenis vartotojui B, kurią valandą, kokiu dažnumu ir iš kur, nors ir nežino, ką jis sako. Šie metaduomenys, kaip jau teigėme straipsnyje *Šifruoti nereiškia būti privačiam*, dažnai yra iškalbingesni už turinį. Žinojimas, kad kažkas penktadienį 22:00 valandą trisdešimt minučių skambino į skyrybų advokatų kontorą, pasakoja istoriją, kurios skambučio turinys niekada nepapasakojo. Tai ta pati situacija, kaip matyti asmenį kelis kartus užeinantį ir išeinantį iš onkologijos klinikos: nereikia girdėti nieko, apie ką kalbama viduje, kad įsivaizduotum, kas vyksta. Vienas atskiras metaduomuo gali nieko nereiškia; keli tarpusavyje susieti nupiešia kažką per daug panašaus į tiesą. E2EE nesaugo galinių taškų: jei gavėjo įrenginys yra kompromituotas kenkėjiškos programos, pranešimas tam gavėjui iššifruojamas įprastai ir kenkėjiška programa jį perskaito. E2EE nesaugo nuo paties pašnekovo tapatybės: jei Alisa tiki kalbanti su Brunu, bet užpuolikas įsiterpė pradžioje (*man in the middle*) ir protokolas neapima nepriklausomo patikrinimo, abi šalys baigia kalbėti su įsibrovėliu galvodamos, kad kalbasi tarpusavyje.

Yra ketvirtas dalykas, kurį verta suformuluoti be dviprasmiškumo. E2EE netrukdo teikėjui, teigiančiam, kad jį siūlo, papildomai saugoti neužšifruoto pranešimo kopiją savo sistemose. Teiginys „mano pranešimai yra užšifruoti ištisiniu būdu“ ir teiginys „teikėjas nesaugo mano turinio“ nėra tie patys. Programėlė gali vykdyti pirmąjį, pažeisdama antrąjį; tai matėme spaudos antraštėse ne kartą nuo 2018 m. Vartotojas, nebent kliento kodas būtų patikrinamas, neturi techninio būdo atskirti vieną atvejį nuo kito be ekspertinio tyrimo. Žinomiausias atvejis plačiajai visuomenei: WhatsApp užšifruoja pranešimus ištisiniu būdu tranzito metu, tačiau jei vartotojas aktyvuoja atsarginę kopiją iCloud arba Google Drive be papildomo šifravimo, ši kopija saugoma nuskaitoma trečiosios šalies infrastruktūroje, o šifravimas pažeidžiamas paties vartotojo gale.

Klausimas, kurio operatorius nenori girdėti

Programėlė, teigianti, kad šifruoja ištisiniu būdu, techniškai gali daryti vieną iš trijų dalykų, susijusių su raktais:

1. **Raktai yra tik įrenginiuose.** Jie generuojami ir yra išskirtinai vartotojų įrenginiuose; operatorius jų nežino ir nesaugo. Tai optimalus atvejis.

2. **Operatorius gali pasiekti, jei nori.** Operatorius turi vartotojų raktus (arba gali juos sugeneruoti savo nuožiūra) ir saugo juos savo duomenų bazėse. Jei jis nori arba yra priverstas, gali perskaityti turinį. Taip yra daugumos „debesų“ paslaugų atveju.
3. **Operatorius negali pasiekti pagal projektą, bet kontroliuoja prieigą.** Operatorius neturi raktų, bet kontroliuoja programėlę, kuri juos generuoja. Jei jis priverčiamas, gali išsiųsti kenkėjišką atnaujinimą, kuris užfiksuotų raktus arba turinį prieš šifravimą. Taip yra daugelio komercinių E2EE paslaugų atveju.

Todėl operatyvinis klausimas yra ne tai, ar kažkas užšifruota, o kas kontroliuoja įrenginį ir programinę įrangą, kuri valdo raktus. „Solo2“ raktai saugomi tik jūsų Saugykloje (jūsų slaptažodžiu užšifruota „IndexedDB“), o programinė įranga yra patikrinamas atvirasis kodas.

Profesionaliam skaitytojui

Ištisinis šifravimas yra skaitmeninio suvereniteto įrankis. Tačiau kaip ir kiekvienas įrankis, jo efektyvumas priklauso nuo rankos, kuri jį laiko, ir pagrindo, ant kurio jis remiasi.

1. Kur generuojami kriptografiniai raktai ir kur jie fiziškai yra? Jei operatorius gali prie jų prieiti (net laikinai, net po atkūrimo priedanga), E2EE yra tik nominalus.
2. Ar yra nepriklausomas pašnekovo patvirtinimas (saugumo numeriai, QR kodai, palyginimas už kanalo ribų), užkertantis kelią „man-in-the-middle“ atakai užmezgant pokalbį?
3. Ar kliento kodas yra audituojamas — atviras, paskelbtas, atkuriamas — ar reikia pasikliauti paslaugų teikėjo žodžiu apie tai, ką klientas iš tikrųjų daro?
4. Kuriuos metaduomenis paslauga generuoja ir saugo, ir kiek laiko? Net jei turinys nepermatomas, metaduomenys gali atkurti didelę dalį neskelbtinos informacijos.

Šie keturi klausimai neprašo sudėtingos techninės informacijos; jie prašo informacijos, į kurią bet kuris sąžiningas operatorius gali atsakyti savo viešojoje dokumentacijoje. Atsakymo kokybė ir tikslumas pasako apie produktą tiek pat, kiek ir pats atsakymas.

Ištisinis šifravimas, jei jis atliktas teisingai, yra viena iš subtiliausių konstrukcijų, kurias šiuolaikinė kriptografija suteikė kasdienei praktikai. Originali idėja — kad du asmenys gali susitarti dėl paslapties viešame kanale — priklauso Whitfield Diffie ir Martin Hellman, 1976 m.; po pusės amžiaus mes vis dar gyvename jos pasekmėmis. Tačiau, kaip ir bet kurio techninio pažado atveju, jo vertė priklauso nuo realaus įgyvendinimo, o ne nuo etiketės. Sąžiningo profesionalo klausimas yra ne „ar tai užšifruota?“, o „kas turi raktus?“. Atsakymai turi skirtingas pasekmes. Verta jas žinoti.

Šaltiniai ir papildomas skaitymas

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, 1976 m. lapkritis. Pagrindinis viešojo rakto kriptografijos straipsnis.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, vieša „Open Whisper Systems“ specifikacija, 2016 m. peržiūra. „Signal“ protokolo ir jo pramoninių darinių pagrindas.
- RFC 7748 — Elliptic Curves for Security (IETF, 2016 m. sausis). X25519 ir X448 kreivių, naudojamų šiuolaikiniuose raktų mainuose, normatyvinė specifikacija.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* („Wiley“, 2010). Skyriai apie raktų mainus ir autentifikuoto šifravimo protokolus.
- Reglamentas (ES) 2024/1183 dėl Europos skaitmeninės tapatybės sistemos (eIDAS 2) — sukuria pagrindus, kur nepriklausomas pašnekovo patvirtinimas įgauna institucinį palaikymą, ir kur skirtumas tarp nominalaus ir tikro šifravimo turi skirtingas teisines pasekmes.

← AnkstesnisKill switch ir institucinis užvaldymasKitas →Verslo modelis kaip pasitikėjimo signalas

Naujausi skaitiniai

- Analizė · 2026 m. gegužės 18 d. Tikras vs tariamas privatumas: klausimai, kuriuos verta užduoti sau
- Analizė · 2026 m. gegužės 18 d. Self-hosting kaip profesinė praktika
- Koncepcija · 2026 m. gegužės 18 d. 24 žodžiai: kas yra kriptografinė tapatybė

Pasiimkite šį straipsnį su savimi ten, kur jums reikia.

↓ Markdown ↓ Paprastas tekstas ↓ PDF

Failas bus atsisiųstas į jūsų įrenginį. Iš ten galite jį išsaugoti, importuoti į Solo2 arba bendrinti bet kur. Cuadernos nusprendžia ne jūsų naudai dėl paskirties vietos.

Vaško antspaudas · SHA-256 d9b7c47d470a836272eb73132750fc4b5e422895939b94de21537a5cf38b1c58

Cuadernos Lacre · Menzuri Gestión S.L. leidinys ·
parašė R.Eugenio · redagavo Solo2 komanda.