Penyulitan hujung-ke-hujung, dijelaskan dengan sebenar

Apa yang dikatakan oleh penyedia apabila mereka menyebut E2EE, dan apa yang tidak mereka katakan. Penjelasan didaktik tentang mekanisme dan batasannya, tanpa bungkus pemasaran.

Biar jelas: WhatsApp mengatakan mesej anda disulitkan hujung-ke-hujung. Ia benar — dan ia tidak mencukupi. Jika sandaran pergi ke iCloud atau Google Drive tanpa penyulitan tambahan, penyulitan itu rosak pada telefon anda sendiri. Soalan operasi bukanlah sama ada ia disulitkan, sebaliknya di mana kunci itu berada.

Maksud penyulitan yang sebenar

Mengenkripsi mesej bermaksud mengubahnya menjadi sesuatu yang kelihatan seperti bunyi bising bagi sesiapa yang tidak memiliki maklumat tertentu yang disebut kunci. Operasi dilakukan pada peranti pengirim dan, dengan kunci yang betul, dibatalkan pada peranti penerima. Di antaranya, mesej berpindah sebagai urutan bait tanpa makna yang jelas. Itulah idea ringkasnya. Selebihnya artikel ini membincangkan nuansa yang mengubahnya, bergantung pada kes, menjadi jaminan nyata atau sekadar label pemasaran.

Kata sifat *hujung-ke-hujung* — dalam bahasa Inggeris *end-to-end*, disingkat E2EE — menambahkan ketepatan. Enkripsi tidak dilakukan supaya pelayan perantara dapat membaca dan menyampaikannya. Ia dilakukan supaya hanya kedua-dua hujung — peranti pengirim dan peranti penerima — yang memiliki kunci tersebut. Mana-mana pelayan yang dilalui mesej hanya melihat bunyi bising, bukan mesej. Itulah perbezaan teknikal dengan enkripsi *dalam transit*, di mana kandungan berpindah terenkripsi dari satu pelayan ke pelayan berikutnya, tetapi setiap pelayan yang dilaluinya mendekripsinya untuk menghantarnya semula, memulihkan teks asli secara sementara.

Paradoks rahsia bersama

Terdapat masalah yang jelas. Supaya dua orang dapat mengenkripsi dan mendekripsi mesej di antara mereka sendiri, kedua-duanya memerlukan kunci yang sama. Namun, bagaimana mereka menyepakati kunci tersebut jika semua yang mereka kirimkan sesama sendiri, mengikut definisi, melalui saluran di mana seseorang mungkin mendengar? Menyepakati kunci di saluran yang sama yang kemudiannya akan mereka gunakan nampaknya mustahil: jika penyerang mendengarnya semasa menyepakatinya, mereka akan dapat mendekripsi semua perkara selepasnya. Selama beberapa dekad, kriptografi klasik menyelesaikan masalah ini dengan cara yang sukar: kunci dihantar secara peribadi, sebelum mula digunakan, dalam pertemuan fizikal. Duta-duta membawa beg kunci yang dijahit ke dalam lapisan kot mereka.

Dalam e-mel kontemporari, penyelesaian tersebut tidak berskala. Jika kita terpaksa pergi secara fizikal ke rumah setiap orang yang ingin kita ajak berkomunikasi secara terenkripsi, kita tidak akan sempat bercakap dengan sesiapa pun. Soalan yang dikemukakan lima puluh tahun lalu oleh komuniti kriptografi adalah ini: adakah mungkin bagi dua orang yang tidak mengenali satu sama lain dan hanya berkongsi saluran awam untuk menyepakati, di saluran awam yang sama, suatu rahsia yang tidak dapat diketahui oleh sesiapa pun yang mendengar saluran tersebut?

Keanggunan Diffie-Hellman

Pada tahun 1976, dua ahli matematik bernama Whitfield Diffie dan Martin Hellman menunjukkan sesuatu yang kelihatan mustahil: bahawa dua orang, yang hanya bercakap melalui saluran awam — saluran di mana sesiapa sahaja dapat mendengar semua yang mereka katakan — dapat menyepakati kata laluan rahsia tanpa ada pendengar yang dapat menemuinya. Ia kedengaran seperti sihir. Padahal bukan: ia matematik. Pertukaran kunci Diffie-Hellman, sebagaimana dikenali sejak itu, adalah asas kepada hampir semua komunikasi terenkripsi di internet, dan penggunaan intensif selama setengah abad serta pemeriksaan akademik global mengesahkan kekuatannya. Sesiapa yang ingin melihat gerak hati visual atau matematik boleh terus membaca. Sesiapa yang lebih suka percaya bahawa ia berfungsi juga boleh meneruskan tanpa kehilangan alur artikel.

Bagi sesiapa yang ingin merasainya dalam sebuah gambar, terdapat analogi yang dikenali dengan warna. Bayangkan Alicia dan Bruno menyepakati warna asas secara terbuka — katakanlah kuning — di depan Eva, yang mendengar mereka. Masing-masing memilih warna rahsia kedua secara peribadi dan mencampur rahsia mereka dengan warna kuning. Alicia mendapat oren tertentu; Bruno mendapat hijau tertentu. Mereka menukar hasilnya di depan Eva. Sekarang masing-masing mencampur warna yang diterima dengan rahasia mereka sendiri, dan kedua-duanya sampai pada warna akhir yang sama, kerana urutan pencampuran tidak penting. Eva telah melihat warna kuning dan dua campuran perantara, tetapi bukan rahsianya; tanpa salah satu rahsia tersebut dia tidak dapat mencapai warna akhir. Matematik yang sebenarnya menggantikan warna untuk eksponensiasi dalam kumpulan modular atau kurva elips, tetapi idenya sama: rahsia bersama dibina di depan umum tanpa ada sesiapa di saluran tersebut yang dapat membina semula.

Dalam aritmetik, bagi sesiapa yang lebih suka melihat mekanismenya: Alicia memilih nombor rahsia a, Bruno memilih b. Mereka menukar g^a dan g^b secara terbuka di saluran tersebut. Alicia menghitung $(g^b)^a$ dan Bruno menghitung $(g^a)^b$; kedua-duanya sampai pada g^{ab} yang sama. Eva

melihat g, g^a, dan g^b melalui saluran tersebut, tetapi memulihkan a dari g^a — yang disebut masalah logaritma diskret — memerlukan masa komputasi yang secara astronomi lebih tinggi daripada usia alam semesta apabila g dipilih dalam kumpulan matematik yang sesuai.

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.

Dari Diffie-Hellman ke protokol Signal

Penyulitan hujung-ke-hujung yang digunakan oleh aplikasi pemesejan profesional hari ini bertumpu, hampir tanpa kecuali, pada versi pertukaran Diffie-Hellman yang elegan dan diperkukuhkan. Protokol Signal, yang direka oleh Trevor Perrin dan Moxie Marlinspike antara tahun 2013 dan 2016, adalah rujukan. Ia menggabungkan dua idea kunci. Pertama, pertukaran kunci dalam kurva elips (X25519), yang menghasilkan rahsia bersama awal antara dua peranti. Kedua, yang disebut Double Ratchet — gear ganda — yang memperbaharui kunci secara automatik dengan setiap mesej, supaya mengompromikan peranti hari ini tidak membenarkan dekripsi mesej masa lalu, atau mesej masa depan setelah gear diputar.

Di Zig, pertukaran X25519 yang menghasilkan rahsia bersama antara dua peranti muat dalam enam baris, menggunakan pustaka 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)
```

Apa yang berlaku dalam enam baris tersebut: Kunci awam berpindah secara terbuka. Kunci peribadi tidak pernah meninggalkan peranti masing-masing. Setiap pihak menurunkan, dari kunci peribadinya dan kunci awam pihak lain, rahsia tiga puluh dua bait yang sama yang tidak dapat dipulihkan oleh sesiapa pun di saluran tersebut. Rahsia itu kemudiannya berfungsi sebagai benih untuk mengenkripsi mesej yang dipertukarkan. Double Ratchet daripada protokol Signal menambah rotasi berterusan materi tersebut supaya kompromi sesaat tidak mengompromikan baki perbualan.

Dan apakah sebenarnya di dalam `std.crypto.dh.X25519`? Tiada sihir tersembunyi. Ia adalah dua fungsi pendek yang boleh dibaca secara keseluruhan dalam pustaka standard Zig sendiri. Yang pertama menerbitkan kunci awam daripada kunci peribadi — «g^{a}» pertukaran tersebut:

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

Dalam bahasa artikel ini: kunci peribadi «didarabkan» — dalam pengertian elips, bukan aritmetik asas — oleh titik asas keluk Curve25519, dan hasilnya diserialkan kepada tiga puluh dua bait. Operasi `clampedMul` adalah versi diperkukuh bagi pendaraban skalar tersebut: ia menggabungkan perlindungan yang ditambah oleh komuniti kriptografi selama bertahun-tahun untuk menahan keluarga serangan yang diketahui. Dua baris badan fungsi.

Fungsi kedua menggabungkan kunci peribadi anda dengan kunci awam yang dihantar oleh pihak lain kepada anda. Ia adalah «$(g^{b})^{a}$» pertukaran itu, yang menghasilkan rahsia bersama tiga puluh dua bait yang tidak pernah dihantar oleh mana-mana daripada anda:

```
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();
}
```

Dua baris lagi. Kunci awam yang diterima ditafsirkan sebagai satu titik pada keluk, dan «didarabkan» dengan kunci peribadi sendiri. Melalui komutativiti operasi keluk — serupa dengan komutativiti pendaraban eksponen yang kita lihat dalam contoh angka — kedua-dua pihak berakhir dengan titik bersiri yang sama: tepat rahsia bersama yang dibicarakan oleh artikel tersebut.

Itu sahaja. Apa yang kelihatan seperti sihir dalam aplikasi adalah, pada hakikatnya, dua fungsi yang terdiri daripada tiga baris setiap satu. Kerumitan teknikal tertumpu pada satu operasi, `clampedMul`, yang ditulis lebih jauh ke bawah dalam pustaka standard yang sama, disemak selama beberapa dekad oleh komuniti kriptografi antarabangsa, dan tersedia kepada sesiapa sahaja yang mahu membacanya huruf demi huruf. Tidak ada kotak hitam sama ada dalam aplikasi kami atau dalam pustaka standard Zig. Terdapat kod sumber terbuka yang dapat difahami oleh manusia, pada kadar yang mereka pilih untuk mendalaminya.

Apa yang dilindungi oleh penyulitan hujung-ke-hujung

Apa yang dilindungi E2EE dengan baik, dengan andaian pelaksanaan yang betul, adalah kandungan mesej dalam transit. Pelayan perantara yang menerima dan meneruskan data terenkripsi akan melihat urutan bait yang tidak dapat difahami. Penyerang dengan akses ke kabel, router, titik akses wifi, akan melihat hal yang sama. Penyedia perkhidmatan yang menyimpan salinan trafik tidak akan dapat membacanya di kemudian hari. Kerajaan yang memerintahkan pengendali perkhidmatan untuk menyerahkan kandungan akan menerima bait tidak dapat difahami yang sama dengan yang dimiliki pelayan pada asalnya.

Ini, dalam istilah praktikal, sangat banyak. Ini adalah perbezaan antara menulis surat di dalam sampul surat legap dan menulisnya di poskad. Kedua-duanya sampai. Hanya satu yang menjaga kandungan daripada posmen.

Apa yang tidak dilindungi oleh penyulitan hujung-ke-hujung

Penting untuk mengetahuinya dengan baik. E2EE tidak melindungi metadata: pelayan masih mengetahui bahawa pengguna A menghantar data kepada pengguna B, pukul berapa, dengan kekerapan berapa dan dari mana, walaupun tidak tahu apa yang dikatakannya. Metadata ini, seperti yang telah kami hujah dalam *Mengenkripsi bukan bermaksud peribadi*, seringkali lebih mendedahkan daripada kandungannya. Mengetahui bahawa seseorang menelefon firma guaman pakar perceraian pada hari Jumaat pukul 22:00 selama tiga puluh minit menceritakan sebuah kisah yang tidak pernah diceritakan oleh kandungan panggilan tersebut. Ini adalah situasi yang sama dengan melihat seseorang masuk dan keluar beberapa kali dari klinik onkologi: tidak perlu mendengar apa-apa yang dibicarakan di dalam untuk membayangkan apa yang sedang berlaku. Satu metadata yang terpencil mungkin tidak bermakna apa-apa; beberapa yang saling berkaitan melukis sesuatu yang terlalu serupa dengan kebenaran. E2EE tidak melindungi hujungnya: jika peranti penerima dikompromikan oleh program jahat, mesej didekripsi secara normal untuk penerima tersebut dan program jahat membacanya. E2EE tidak melindungi terhadap identiti lawan bicara itu sendiri: jika Alicia yakin dia bercakap dengan Bruno tetapi penyerang telah menyelitkan diri di awal (seorang *man in the middle*) dan protokol tidak menyertakan pengesahan bebas, kedua-dua pihak akhirnya bercakap dengan penceroboh sambil menyangka mereka sedang bercakap antara satu sama lain.

Ada hal keempat yang wajar dirumuskan tanpa kekaburan. E2EE tidak menghalang penyedia yang mendakwa menawarkannya daripada juga menyimpan salinan mesej yang tidak terenkripsi di sistemnya sendiri. Pernyataan «mesej saya terenkripsi hujung-ke-hujung» dan pernyataan «penyedia tidak menyimpan kandungan saya» tidaklah sama. Sebuah aplikasi dapat memenuhi pernyataan pertama sambil melanggar pernyataan kedua; kita telah melihatnya di tajuk berita berulang kali sejak 2018. Pengguna, melainkan kod pelanggan dapat disahkan, tidak mempunyai cara teknikal untuk membezakan satu kes daripada kes yang lain tanpa siasatan pakar. Kes yang paling dikenali orang ramai: WhatsApp mengenkripsi mesej hujung-ke-hujung dalam transit, tetapi jika pengguna mengaktifkan sandaran di iCloud atau Google Drive tanpa enkripsi tambahan, salinan itu disimpan secara terbaca di infrastruktur pihak ketiga, dan enkripsi rosak di hujung pengguna itu sendiri.

Soalan yang tidak mahu didengar oleh pengendali

Aplikasi yang mendakwa mengenkripsi hujung-ke-hujung dapat, secara teknikal, melakukan satu daripada tiga perkara berkaitan kunci:

1. **Kunci hanya berada di peranti.** Kunci dicipta dan berada secara eksklusif di peranti pengguna; pengendali tidak mengetahuinya atau menyimpannya. Ini adalah kes yang optimum.
2. **Pengendali boleh mengakses jika mereka mahu.** Pengendali mempunyai kunci pengguna (atau boleh menjananya sesuka hati) dan menyimpannya dalam pangkalan data mereka. Jika mereka mahu atau dipaksa, mereka boleh membaca kandungan tersebut. Ini adalah kes bagi kebanyakan perkhidmatan «awan».
3. **Pengendali tidak boleh mengakses mengikut reka bentuk, tetapi mengawal akses.** Pengendali tidak mempunyai kunci, tetapi mempunyai kawalan ke atas aplikasi yang menjananya. Jika dipaksa, mereka boleh menghantar kemas kini berniat jahat yang menangkap kunci atau kandungan sebelum penyulitan. Ini adalah kes bagi banyak perkhidmatan E2EE komersial.

Oleh itu, soalan operasi bukanlah sama ada sesuatu itu disulitkan, sebaliknya siapa yang mempunyai kawalan ke atas peranti dan perisian yang menguruskan kunci. Dalam Solo2, kunci hanya berada dalam «Bilik Kebal» anda (IndexedDB yang disulitkan dengan kata laluan anda) dan perisian tersebut adalah kod sumber terbuka yang boleh disahkan.

Untuk pembaca profesional

Penyulitan hujung-ke-hujung adalah alat untuk kedaulatan digital. Tetapi seperti setiap alat, keberkesanannya bergantung pada tangan yang memegangnya dan landasan tempat ia berpijak.

1. Di manakah kunci kriptografi dijana dan di manakah ia berada secara fizikal? Jika pengendali boleh mengaksesnya (walaupun sementara, walaupun dengan alasan pemulihan), E2EE tersebut hanyalah nominal.
2. Adakah terdapat pengesahan bebas lawan bicara (nombor keselamatan, kod QR, perbandingan luar jalur) yang menghalang serangan man-in-the-middle semasa penubuhan perbualan?
3. Adakah kod pelanggan boleh diaudit — terbuka, diterbitkan, boleh dihasilkan semula — atau adakah ia memerlukan kepercayaan kepada kata-kata penyedia tentang apa yang sebenarnya dilakukan oleh pelanggan?
4. Apakah metadata yang dihasilkan dan disimpan oleh perkhidmatan, dan untuk berapa lama? Walaupun kandungannya legap, metadata boleh membina semula sebahagian besar maklumat sensitif.

Keempat-empat soalan ini tidak meminta maklumat teknikal lanjutan; ia meminta maklumat yang boleh dijawab oleh mana-mana pengendali yang jujur dalam dokumentasi awam mereka. Kualiti dan ketepatan jawapan memberitahu tentang produk itu sama seperti jawapan itu sendiri.

Penyulitan hujung-ke-hujung, jika dilakukan dengan betul, adalah salah satu binaan terbaik yang diberikan oleh kriptografi kontemporari kepada amalan harian. Idea asal — dua orang boleh bersetuju dengan rahsia di saluran awam — milik Whitfield Diffie dan Martin Hellman, 1976; setengah abad kemudian kita terus hidup dalam konsekuensinya. Namun, seperti mana-mana janji teknikal, nilainya bergantung pada pemenuhan sebenar, bukan pada label. Soalan profesional yang jujur bukanlah «adakah ia disulitkan?», sebaliknya «siapa yang memegang kuncinya?». Jawapannya mempunyai akibat yang berbeza. Ada baiknya mengetahuinya.

Sumber dan bacaan lanjut

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, November 1976. Artikel asas mengenai kriptografi kunci awam.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, spesifikasi awam oleh Open Whisper Systems, semakan 2016. Asas kepada protokol Signal dan terbitan industrinya.
- RFC 7748 — Elliptic Curves for Security (IETF, Januari 2016). Spesifikasi normatif bagi keluk X25519 dan X448 yang digunakan dalam pertukaran kunci moden.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). Bab-bab tentang pertukaran kunci dan protokol penyulitan disahkan.
- Peraturan (EU) 2024/1183 mengenai rangka kerja identiti digital Eropah (eIDAS 2) — mewujudkan rangka kerja di mana pengesahan bebas lawan bicara memperoleh sokongan institusi, dan di mana perbezaan antara penyulitan nominal dan sebenar mempunyai akibat undang-undang yang berbeza.

← SebelumnyaKill switch dan penangkapan institusiSeterusnya →Model perniagaan sebagai isyarat kepercayaan

Bacaan terkini

- Analisis · 18 Mei 2026 Privasi nyata vs semu: soalan yang perlu anda tanya diri sendiri

- Analisis · 18 Mei 2026 Self-hosting sebagai amalan profesional
- Konsep · 18 Mei 2026 24 perkataan: apakah itu identiti kriptografi

Bawa artikel ini bersama anda ke mana sahaja anda memerlukannya.

↓ Markdown ↓ Teks biasa ↓ PDF

Fail akan dimuat turun ke peranti anda. Dari sana anda boleh menyimpannya, mengimportnya ke Solo2, atau berkongsinya di mana sahaja anda mahu. Cuadernos tidak memutuskan destinasi untuk anda.

Mohor lilin · SHA-256 995ad6ad87dc457db5187af0654323678af2b039b307ad55cdae6a616d368261

Cuadernos Lacre · Penerbitan daripada Menzuri Gestión S.L. ·
ditulis oleh R.Eugenio · disunting oleh pasukan Solo2.

Laman web ini tidak menggunakan kuki dan tidak memuatkan sumber pihak ketiga. Ia menggunakan pembilang lawatan tanpa nama yang dihoskan sendiri (Umami, pada pelayan Eropah kami) dan JavaScript minimum yang diperlukan untuk dua kawalan pengepala: tema terang atau gelap, dan pemilih bahasa. Tiada penjejak, tiada pemprofilan, tiada perkongsian data. Jika anda ingin mengikuti kami: RSS.