تشفير الطرف إلى الطرف، شرح حقيقي

ما تقوله الشركات عندما تشير إلى E2EE، وما لا تقوله. شرح تعليمي للآلية وحدودها، بعيداً عن الغلاف الإعلاني.

للتوضيح: يقول WhatsApp إن رسائلك مشفرة من الطرف إلى الطرف. هذا صحيح — ولكنه ليس كافياً. إذا كانت النسخة الاحتياطية تذهب إلى iCloud أو Google Drive بدون تشفير إضافي، فإن التشفير ينكسر في هاتفك نفسه. السؤال العملي ليس ما إذا كان مشفراً، بل أين تقيم المفاتيح.

ما يعنيه التشفير، حقاً

تشفير رسالة ما يعني تحويلها إلى شيء يبدو كضجيج لأي شخص لا يملك معلومة معينة تسمى المفتاح. تتم العملية في جهاز المرسل، ومع المفتاح الصحيح، يتم فكها في جهاز المستلم. في المنتصف، تنتقل الرسالة كسلسلة من البايتات بدون معنى واضح. هذه هي الفكرة البسيطة. أما بقية المقال فيتناول التفاصيل الدقيقة التي تحولها، حسب الحالة، إلى ضمان حقيقي أو مجرد ملصق تسويقي.

تضيف صفة *من الطرف إلى الطرف* —بالإنجليزية *end-to-end*، واختصاراً E2EE— دقة إضافية. لا يتم التشفير لكي يتمكن خادم وسيط من قراءته وتسليمه، بل لكي يمتلك الطرفان فقط —جهاز المرسل وجهاز المستلم— المفتاح. أي خادم تمر عبره الرسالة يرى الضجيج، لا الرسالة. هذا هو الفرق التقني مع التشفير *أثناء النقل*، حيث ينتقل المحتوى مشفراً من خادم إلى آخر، ولكن كل خادم يمر عبره يقوم بفك تشفيره لإعادة إرساله، مما يستعيد النص الأصلي مؤقتاً.

مفارقة السر المشترك

هناك مشكلة واضحة. لكي يتمكن شخصان من تشفير وفك تشفير الرسائل بينهما، يحتاج كلاهما إلى نفس المفتاح. ولكن، كيف يتفقان على هذا المفتاح إذا كان كل ما يتبادلانه، بالتعريف، يمر عبر قناة قد يكون هناك من يتنصت عليها؟ يبدو الاتفاق على المفتاح في نفس القناة التي سيستخدمانه فيها لاحقاً أمراً مستحيلاً: إذا سمعه المهاجم عند الاتفاق عليه، فسيتمكن من فك تشفير كل ما يلي. لعقود من الزمن، حل علم التشفير الكلاسيكي هذا بالطريقة الصعبة: كانت المفاتيح تُسلم شخصياً، قبل البدء في استخدامها، في لقاءات فيزيائية. كان السفراء يحملون حقائب المفاتيح مخيطة في بطانة معاطفهم.

في البريد الإلكتروني المعاصر، لا يمكن لهذا الحل أن يتوسع. إذا كان علينا الذهاب فيزيائياً إلى منزل كل شخص نعتزم التواصل معه بشكل مشفر، فلن نتمكن من التحدث مع أحد. كان السؤال الذي طرحه مجتمع التشفير قبل خمسين عاماً هو: هل من الممكن لشخصين لا يعرفان بعضهما البعض ولا يتشاركان سوى قناة عامة أن يتفقا، في نفس تلك القناة العامة، على سر لا يمكن لأي شخص يتنصت على القناة معرفته؟

أناقة Diffie-Hellman

في عام ١٩٧٦، أثبت عالمان في الرياضيات هما Whitfield Diffie و Martin Hellman شيئاً يبدو مستحيلاً: أن شخصين، يتحدثان فقط عبر قناة عامة —قناة حيث يمكن لأي شخص سماع كل ما يقولانه— يمكنهما الاتفاق على كلمة سر دون أن يتمكن أي مستمع من اكتشافها. يبدو الأمر كالسحر، لكنه ليس كذلك: إنه الرياضيات. تبادل مفاتيح Diffie-Hellman، كما عُرف منذ ذلك الحين، هو الأساس لجميع اتصالات الإنترنت المشفرة تقريباً، ونصف قرن من

الاستخدام المكثف والتمحيص الأكاديمي العالمي يؤكد متانته. من يريد رؤية الحدس البصري أو الرياضيات يمكنه مواصلة القراءة. ومن يفضل الثقة في أنه يعمل يمكنه أيضاً المتابعة دون أن يفقد خيط المقال.

لمن يريد تخيله في صورة، هناك تشبيه معروف بالألوان. تخيل أن أليس وبرونو يتفقان علانية على لون أساسي — ليكن الأصفر— أمام إيفا التي تستمع إليهما. يختار كل منهما سراً لوناً ثانياً سرياً ويخلط سره مع الأصفر. تحصل أليس على برتقالي خاص؛ ويحصل برونو على أخضر خاص. يتبادلان النتائج أمام إيفا. الآن يخلط كل منهما اللون المستلم مع سره الخاص، ويصل كلاهما إلى نفس اللون النهائي، لأن ترتيب الخلط لا يهم. رأت إيفا الأصفر والخليطين الوسيطين، لكنها لم تَرَ الأسرار؛ وبدون أي من الأسرار لا يمكنها الوصول إلى اللون النهائي. تغير الرياضيات الحقيقية الألوان بعمليات أسية في مجموعات نمطية أو منحنيات إهليلجية، لكن الفكرة هي نفسها: السر المشترك يُبنى علانية دون أن يتمكن أي شخص في القناة من إعادة بنائه.

في الحساب، لمن يفضل رؤية الآلية: تختار أليس رقماً سرياً a، ويختار برونو b. يتبادلان g^a و g^b علانية عبر القناة. تحسب أليس $(g^b)^a$ ويحسب برونو $(g^a)^b$؛ وكلاهما يصلان إلى نفس الـ g^{ab}. ترى إيفا g و g^a و g^b تمر عبر القناة، لكن استعادة a من g^a —ما يسمى بمشكلة اللوغاريتم المنفصل— تتطلب وقتاً حسابياً يفوق عمر الكون بمراحل عندما يتم اختيار g في مجموعة رياضية مناسبة.

لمن يريد التحقق من ذلك بأرقام صغيرة. يمكن تتبع تبادل Diffie-Hellman بأكمله بأرقام صغيرة بما يكفي لإجراء الحسابات يدوياً. من يفضل عدم الخوض في الحساب يمكنه تخطي هذه الكتلة دون أن يفقد خيط المقال؛ ومن يريد رؤية الآلية تعمل خطوة بخطوة سيجدها هنا. **القواعد العامة**، التي يمكن لأي شخص قراءتها: عدد أولي $p = 11$ (في Diffie-Hellman الحقيقي يتكون من حوالي ثلاثمائة رقم؛ نستخدم أحد عشر لكي تتسع الحسابات في صفحة واحدة)، قاعدة $g = 2$، واتفاق على أن كل الحساب يتم *بالمقياس* p (modulo p) — تُجرى العملية، تُقسم على p، ويُحتفظ بالباقي، مثل ساعة بـ أحد عشر موضعاً تعود إلى الصفر عند تجاوز العشرة. **الاختيارات الخاصة**، واحدة لكل منهما ولا تتم مشاركتها أبداً: تختار أليس $a = 4$. يختار برونو $b = 7$.

الخطوة ١. تحسب أليس $2^4 = 16$، ثم $16 = 5 \bmod 11$. ترسل الخمسة. تدونها إيفا.

الخطوة ٢. يحسب برونو $2^7 = 128$، ثم $128 = 7 \bmod 11$. يرسل السبعة. تدونها إيفا أيضاً. بعد الإرسالين، يحتوي دفتر إيفا على أربعة معطيات: $A = 5$، $B = 7$، $g = 2$، $p = 11$. ينقصها الرقم المشترك الذي أوشكت أليس وبرونو على استنتاجه — والذي لن تتمكن إيفا من إعادة بنائه.

الخطوة ٣. تأخذ أليس السبعة التي أرسلها لها برونو وترفعه إلى أُسّها الخاص $a = 4$. لتجنب التعامل مع $7^4 = 2401$، يتم الحساب على أجزاء بتطبيق المقياس (mod) في كل خطوة:

$$7^2 = 49$$

$$49 \bmod 11 = 5$$

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

$$25 \bmod 11 = 3$$

تحصل أليس على الرقم **3**.

الخطوة ٤. يأخذ برونو الخمسة التي أرسلتها له أليس ويرفعه إلى أُسّه الخاص $b = 7$. مرة أخرى على أجزاء:

$$5^2 = 25 \bmod 11 = 3$$

$$5^4 = (5^2)^2 = 9 = 3^2 \bmod 11 = 9$$

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

أخيراً $5^7 = 5^6 \times 5 = 5 \times 5 = 25 \bmod 11 = \mathbf{3}$.

يحصل برونو أيضاً على **3**.

وصل كلاهما إلى نفس الرقم، 3، بالعمل بشكل متواز. لم يرسل أي منهما أسّه الخاص في أي وقت. لا تعرف أليس أن $b = 7$؛ لا يعرف برونو أن $a = 4$. استخدم كل منهما القيمة العامة التي أرسلها الآخر مدمجة مع أسّه الخاص، والتقيا في نفس الوجهة. **لماذا يصلان إلى نفس الرقم؟** ما حسبه كل منهما: أليس، $(g^b)^a = 2^{7\times4} = 2^{28} \bmod 11$. برونو، $(g^a)^b = 2^{4\times7} = 2^{28} \bmod 11$. إنها نفس الكمية لأن ترتيب ضرب الأسس لا يهم ($7 \times 4 = 4 \times 7$). وصل كل منهما عبر مسار مختلف إلى نفس الوجهة.

وماذا عن إيفا؟ لديها في دفترها $p = 11$، $g = 2$، $A = 5$، $B = 7$، وتود الحصول على 3. لحسابه ستحتاج إلى معرفة a أو b — لكن لم ينتقل أي منهما عبر القناة. طريقها الوحيد هو أن تسأل نفسها: «لأي أُس a يتحقق $2^a \bmod 11 = 5$؟». مع p صغير هكذا يمكنها تجربة 0، 1، 2، 3، 4... والعثور عليه في أقل من دقيقة. ولكن عند استبدال 11 بعدد أولي من ثلاثمائة رقم، يكون فضاء الأسس المحتملة يحتوي على عناصر أكثر من الذرات في الكون المرئي. **لا يوجد حتى يومنا هذا أي خوارزمية معروفة للبشرية يمكنها اجتياز هذا الفضاء في أقل من مليارات السنين.** هذه هي ما تُسمى بـ *مشكلة اللوغاريتم المنفصل*: سهلة إلى الأمام، مستحيلة حسابياً إلى الخلف. وهذا هو السبب في أن التشفير يقاوم حتى لو تابعت إيفا المحادثة بأكملها حرفاً بحرف.

ثلاثة مكونات بسيطة —حساب على ساعة، الرفع إلى أُس، وتبادلية الضرب ($a \cdot b = b \cdot a$)— مدمجة تنتج بروتوكولاً يعتمد عليه نصف البشرية كل يوم لاتصالاتهم الخاصة. لا تبدو أي من القطع الثلاث، على حدة، مميزة. الحاسم هو التجميع.

من Diffie-Hellman إلى بروتوكول Signal

تشفير الطرف إلى الطرف الذي تستخدمه تطبيقات المراسلة المهنية اليوم يعتمد، بلا استثناء تقريباً، على نسخة أنيقة ومعززة من تبادل Diffie-Hellman. بروتوكول Signal، الذي صممه Trevor Perrin و Moxie Marlinspike بين عامي ٢٠١٣ و ٢٠١٦، هو المرجع. يجمع بين فكرتين رئيسيتين. الأولى، تبادل المفاتيح في المنحنيات الإهليلجية (X25519)، التي تنتج السر المشترك الأولي بين جهازين. الثانية، ما يسمى Double Ratchet —الترس المزدوج— الذي يجدد المفاتيح تلقائياً مع كل رسالة، بحيث أن اختراق الجهاز اليوم لا يسمح بفك تشفير الرسائل الماضية، ولا الرسائل المستقبلية بمجرد تدوير الترس.

في Zig، تبادل X25519 الذي ينتج السر المشترك بين جهازين يتسع في ستة أسطر، باستخدام المكتبة القياسية:

```
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)
```

ما يحدث في تلك الأسطر الستة: تنتقل المفاتيح العامة علانية. المفاتيح الخاصة لا تخرج أبداً من الجهاز المعني. يشتق كل طرف، انطلاقاً من مفتاحه الخاص والمفتاح العام للطرف الآخر، نفساً السر المكون من اثنين وثلاثين بايتاً والذي لا يمكن لأي شخص في القناة استعادته. يعمل هذا السر لاحقاً كبذرة لتشفير الرسائل المتبادلة. يضيف Double Ratchet الخاص ببروتوكول Signal تدويراً مستمراً لهذه المادة لكي لا يؤدي اختراق لحظة معينة إلى اختراق بقية المحادثة.

وماذا يوجد بالضبط داخل `std.crypto.dh.X25519`؟ لا يوجد سحر خفي. هما دالتان قصيرتان يمكن قراءتهما بالكامل في المكتبة القياسية الخاصة بـ Zig. الأولى تشتق المفتاح العام من المفتاح الخاص — الـ «g^a» في التبادل:

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

بلغة المقال: يُضرب المفتاح الخاص —بالمعنى الإهليلجي، وليس الحسابي الأولي— بنقطة الأساس (base point) لمنحنى Curve25519، وتُسلسل النتيجة في اثنين وثلاثين بايتاً. عملية `clampedMul` هي النسخة المعززة من هذا الضرب السلمي: تدمج ضمانات الأمان التي أضافها مجتمع التشفير على مر السنين لمقاومة عائلات معروفة من الهجمات. سطران في جسم الدالة.

الدالة الثانية تدمج مفتاحك الخاص مع المفتاح العام الذي يرسله لك الطرف الآخر. إنها الـ «$(g^b)^a$» في التبادل، والتي تنتج السر المشترك المكون من اثنين وثلاثين بايتاً والذي لم ينقله أي منكما:

```
_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
                                                                                    {
```

سطران آخران. يُفسر المفتاح العام المُتلقى كنقطة على المنحنى، ويُضرب في المفتاح الخاص بك. بفضل التبادلية في عملية المنحنى —على غرار التبادلية في ضرب الأسس التي رأيناها في المثال الرقمي— ينتهي كلا الطرفين بنفس النقطة المُسلسلة: وهو بالضبط السر المشترك الذي يتحدث عنه المقال.

هذا كل شيء. ما يبدو في التطبيق كسحر، هو في الواقع دالتان من ثلاثة أسطر لكل منهما. يتركز التعقيد التقني في عملية واحدة، `clampedMul`، المكتوبة لاحقاً في نفس المكتبة القياسية، والتي تم تدقيقها لعقود من قبل مجتمع التشفير الدولي، ومتاحة لأي شخص يريد قراءتها حرفاً بحرف. لا يوجد صندوق أسود سواء في تطبيقنا أو في المكتبة القياسية لـ Zig. يوجد كود مفتوح المصدر يمكن لأي إنسان فهمه، باختيار السرعة التي يريد التعمق بها.

ما الذي يحميه تشفير الطرف إلى الطرف

ما يحميه E2EE جيداً، بافتراض تنفيذ صحيح، هو محتوى الرسالة أثناء النقل. الخادم الوسيط الذي يستلم ويعيد إرسال البيانات المشفرة سيرى سلسلة من البايتات غير المفهومة. المهاجم الذي لديه وصول إلى الكابل، الراوتر، أو نقطة وصول الواي فاي، سيرى الشيء نفسه. مزود الخدمة الذي يحتفظ بنسخ من حركة المرور لن يتمكن من قراءتها لاحقاً. الحكومة التي تأمر مشغل الخدمة بتسليم المحتوى ستتلقى نفس البايتات غير المفهومة التي كانت لدى الخادم في المقام الأول.

هذا، من الناحية العملية، يعد كثيراً. إنه الفرق بين كتابة رسالة داخل ظرف معتم وكتابتها على بطاقة بريدية. كلتاهما تصلان، لكن واحدة فقط تحفظ المحتوى أمام ساعي البريد.

ما لا يحميه تشفير الطرف إلى الطرف

يجدر معرفة ذلك جيداً أيضاً. E2EE لا يحمي الميتاداتا (البيانات الوصفية): لا يزال الخادم يعرف أن المستخدم أ يرسل بيانات للمستخدم ب، في أي ساعة، وبأي تكرار ومن أين، حتى لو كان لا يعرف ماذا يقول. هذه الميتاداتا، كما جادلنا بالفعل في *التشفير لا يعني الخصوصية*، غالباً ما تكون كاشفة أكثر من المحتوى. معرفة أن شخصاً ما اتصل بمكتب محاماة متخصص في الطلاق يوم الجمعة الساعة ٢٢:٠٠ لمدة ثلاثين دقيقة تحكي قصة لم يحكِها محتوى المكالمة أبداً. إنه نفس الموقف عند رؤية شخص يدخل ويخرج عدة مرات من عيادة أورام: لا حاجة لسماع أي شيء مما يقال في الداخل لتخيل ما يحدث. ميتاداتا واحدة معزولة قد لا تعني شيئاً؛ لكن عدة بيانات متقاطعة فيما بينها ترسم شيئاً قريباً جداً من الحقيقة. E2EE لا يحمي الأطراف: إذا كان جهاز المستلم مخترقاً ببرنامج ضار، فسيتم فك تشفير الرسالة بشكل طبيعي لذلك المستلم ويقوم البرنامج الضار بقراءتها. E2EE لا يحمي من هوية المحاور في حد ذاتها: إذا اعتقدت أليس أنها تتحدث مع برونو لكن مهاجماً تدخل في البداية (هجوم *man in the middle*) ولم يتضمن البروتوكول تحققاً مستقلاً، سينتهي الأمر بالطرفين بالتحدث مع الدخيل معتقدين أنهما يتحدثان مع بعضهما البعض.

هناك أمر رابع يجدر صياغته دون غموض. E2EE لا يمنع المزود الذي يدعي تقديمه من الاحتفاظ أيضاً بنسخة من الرسالة غير مشفرة في أنظمته الخاصة. الادعاء بأن «رسائلي مشفرة من الطرف إلى الطرف» والادعاء بأن «المزود لا يحتفظ بمحتواي» ليسا نفس الشيء. يمكن لتطبيق أن يلتزم بالأول بينما ينتهك الثاني؛ لقد رأينا هذا في عناوين الصحف بشكل متكرر منذ عام ٢٠١٨. لا يملك المستخدم، ما لم يكن كود العميل قابلاً للتحقق، طريقة تقنية لتمييز حالة عن الأخرى دون تحقيق خبير. الحالة الأكثر شهرة لدى الجمهور العام: يقوم WhatsApp بتشفير الرسائل

من الطرف إلى الطرف أثناء النقل، ولكن إذا قام المستخدم بتفعيل النسخ الاحتياطي في iCloud أو Google Drive بدون تشفير إضافي، فسيتم تخزين تلك النسخة قابلة للقراءة في بنية تحتية لطرف ثالث، وينكسر التشفير في طرف المستخدم نفسه.

السؤال الذي لا يريد المشغل سماعه

يمكن للتطبيق الذي يدعي التشفير من الطرف إلى الطرف، من الناحية التقنية، القيام بواحد من ثلاثة أشياء فيما يتعلق بالمفاتيح:

1. **المفاتيح تقيم فقط في الأجهزة.** يتم توليدها وتقيم حصرياً في أجهزة المستخدمين؛ لا يعرفها المشغل ولا يخزنها. هذه هي الحالة المثالية.
2. **يمكن للمشغل الوصول إذا أراد.** يمتلك المشغل مفاتيح المستخدمين (أو يمكنه توليدها كما يشاء) ويحفظها في قواعد بياناته. إذا أراد أو أُجبر، يمكنه قراءة المحتوى. هذا هو حال معظم الخدمات «السحابية».
3. **لا يمكن للمشغل الوصول حسب التصميم، لكنه يتحكم في الوصول.** لا يمتلك المشغل المفاتيح، لكنه يتحكم في التطبيق الذي يولدها. إذا أُجبر، يمكنه إرسال تحديث ضار يلتقط المفاتيح أو المحتوى قبل التشفير. هذا هو حال العديد من خدمات E2EE التجارية.

لذلك، فإن السؤال العملي ليس ما إذا كان الشيء مشفراً، بل من يملك السيطرة على الجهاز وعلى البرمجيات التي تدير المفاتيح. في Solo2، تقيم المفاتيح فقط في «بوابتك» (IndexedDB مشفرة بكلمة سرك) والبرمجيات هي كود مفتوح المصدر قابل للتحقق.

للقارئ المهني

تشفير الطرف إلى الطرف هو أداة للسيادة الرقمية. ولكن ككل أداة، تعتمد فعاليتها على اليد التي تمسك بها والأرض التي تستند إليها.

1. أين يتم توليد المفاتيح التشفيرية وأين تقيم فيزيائياً؟ إذا كان بإمكان المشغل الوصول إليها (حتى ولو مؤقتاً، أو تحت مسمى الاسترداد)، فإن E2EE هو مجرد اسم.
2. هل يوجد تحقق مستقل من هوية المحاور (أرقام أمان، رموز استجابة سريعة QR، مقارنة خارج القناة) يمنع هجوم رجل في المنتصف أثناء تأسيس المحادثة؟
3. هل كود العميل قابل للتدقيق —مفتوح، منشور، قابل لإعادة الإنتاج— أم يتطلب الثقة في كلمة المزود حول ما يفعله العميل في الواقع؟
4. ما هي الميتاداتا التي يولدها ويحتفظ بها الخدمة، ولكم من الوقت؟ حتى لو كان المحتوى معتماً، يمكن للميتاداتا إعادة بناء جزء كبير من المعلومات الحساسة.

هذه الأسئلة الأربعة لا تطلب معلومات تقنية متقدمة؛ بل تطلب معلومات يمكن لأي مشغل نزيه الإجابة عليها في توثيقه العام. جودة ودقة الإجابة تعبر عن المنتج بقدر الإجابة نفسها.

تشفير الطرف إلى الطرف، إذا نُفذ بشكل جيد، يعد واحداً من أروع البناءات التي قدمها علم التشفير المعاصر للممارسة اليومية. الفكرة الأصلية —إمكانية اتفاق شخصين على سر عبر قناة عامة— تعود إلى Whitfield Diffie و Martin Hellman، ١٩٧٦؛ وبعد نصف قرن ما زلنا نعيش في تبعات ذلك. ولكن، كما هو الحال مع أي وعد تقني، فإن قيمته تعتمد على الالتزام الحقيقي، وليس على التسمية. سؤال المهني النزيه ليس «هل هو مشفر؟»، بل «من يملك المفاتيح؟». للإجابات عواقب مختلفة. يجدر بك معرفتها.

المصادر ومزيد من القراءة

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*، IEEE Transactions on Information Theory، نوفمبر ١٩٧٦. المقال التأسيسي لتشفير المفتاح العام.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*، المواصفة العامة لـ Open Whisper Systems، مراجعة ٢٠١٦. أساس بروتوكول Signal ومشتقاته الصناعية.

- IETF) RFC 7748 — Elliptic Curves for Security، يناير ٢٠١٦). المواصفة المعيارية لمنحنيات X25519 و X448 المستخدمة في تبادلات المفاتيح الحديثة.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley، ٢٠١٠). فصول حول تبادل المفاتيح وبروتوكولات التشفير الموثق.
- لائحة (الاتحاد الأوروبي) 2024/1183 الخاصة بإطار الهوية الرقمية الأوروبية (eIDAS 2) — تؤسس أطراً حيث يكتسب التحقق المستقل من المحاور دعماً مؤسسياً، وحيث التمييز بين التشفير الاسمي والتشفير الحقيقي له عواقب قانونية مختلفة.

← السابقKill switch والاستحواذ المؤسسيالتالي → نموذج العمل كإشارة ثقة

قراءات حديثة

- تحليل · ١٨ مايو ٢٠٢٦ الخصوصية الحقيقية مقابل الظاهرية: الأسئلة التي ينبغي طرحها
- تحليل · ١٨ مايو ٢٠٢٦ الاستضافة الذاتية كممارسة مهنية
- مفهوم · ١٨ مايو ٢٠٢٦ الـ 24 كلمة: ما هي الهوية التشفيرية

خذ هذا المقال معك أينما احتجت إليه.

↓ ماركداون ↓ نص عادي ↓ PDF

سيتم تنزيل الملف على جهازك. من هناك يمكنك حفظه، أو استيراده إلى Solo2 أو مشاركته أينما تريد. Cuadernos لا تقرر الوجهة نيابة عنك.

ختم شمعي · SHA-256 be86b6c77b1f7e6740c6af6b9b2b6146b4a39af6167aac75ef495ff3c469069d

Cuadernos Lacre · منشور لشركة Menzuri Gestión S.L. ·
بقلم R.Eugenio · تحرير فريق Solo2.