การเข้ารหัสแบบปลายทางถึงปลายทาง อธิบายอย่างแท้จริง

สิ่งที่ผู้ให้บริการพูดเมื่อพูดถึง E2EE และสิ่งที่พวกเขาไม่ได้พูด คำอธิบายเชิงวิชาการเกี่ยวกับกลไกและข้อจำกัด โดยไม่มีการห่อหุ้มด้วยการโฆษณา

เพื่อให้เข้าใจตรงกัน: WhatsApp บอกว่าข้อความของคุณถูกเข้ารหัสแบบปลายทางถึงปลายทาง มันคือเรื่องจริง — และมันไม่เพียงพอ หากการสำรองข้อมูลไปที่ iCloud หรือ Google Drive โดยไม่มีการเข้ารหัสเพิ่มเติม การเข้ารหัสก็จะถูกทำลายบนโทรศัพท์ของคุณเอง คำถามเชิงปฏิบัติการไม่ใช่ว่ามีการเข้ารหัสหรือไม่ แต่อยู่ที่ว่ารหัสอยู่ที่ไหน

การเข้ารหัสหมายถึงอะไรจริงๆ

การเข้ารหัสข้อความคือการเปลี่ยนให้เป็นสิ่งที่ดูเหมือนสัญญาณรบกวนสำหรับใครก็ตามที่ไม่มีข้อมูลเฉพาะที่เรียกว่ารหัส (Key) การดำเนินการนี้จะทำที่อุปกรณ์ของผู้ส่ง และด้วยรหัสที่ถูกต้อง มันจะถูกย้อนกลับที่อุปกรณ์ของผู้รับ ในระหว่างนั้น ข้อความจะเดินทางเป็นชุดของไบต์ที่ไม่มีความหมายชัดเจน นั่นคือแนวคิดง่ายๆ ส่วนที่เหลือของบทความจะเกี่ยวข้องกับรายละเอียดปลีกย่อยที่เปลี่ยนให้เป็น การรับประกันที่แท้จริง หรือเป็นเพียงป้ายทางการตลาดขึ้นอยู่กับกรณี

คำคุณศัพท์ *แบบปลายทางถึงปลายทาง* — ในภาษาอังกฤษคือ *end-to-end* ย่อว่า E2EE — เพิ่มความแม่นยำเข้าไป การเข้ารหัสไม่ได้ทำเพื่อให้เซิร์ฟเวอร์คนกลางสามารถอ่านและส่งมอบได้ แต่มันถูกทำเพื่อให้มีเพียงสองปลายทางเท่านั้น — อุปกรณ์ของผู้ส่งและอุปกรณ์ของผู้รับ — ที่ถือรหัสไว้ เซิร์ฟเวอร์ใดก็ตามที่ข้อความผ่านไปจะเห็นสัญญาณรบกวน ไม่ใช่ข้อความ นั่นคือความแตกต่างทางเทคนิคกับการเข้ารหัส *ระหว่างการส่งผ่าน* (In Transit) ซึ่งเนื้อหาจะเดินทางแบบเข้ารหัสจากเซิร์ฟเวอร์หนึ่งไปยังอีกเซิร์ฟเวอร์หนึ่ง แต่แต่ละเซิร์ฟเวอร์ที่ผ่านไปจะถอดรหัสเพื่อส่งต่อ ทำให้เนื้อหาที่เป็นข้อความปกติถูกกู้คืนชั่วคราว

ความย้อนแย้งของความลับที่ใช้ร่วมกัน

มีปัญหาที่ชัดเจนอยู่ เพื่อให้คนสองคนสามารถเข้ารหัสและถอดรหัสข้อความระหว่างกันได้ ทั้งคู่ต้องการรหัสเดียวกัน แต่พวกเขาจะตกลงรหัสนี้ได้อย่างไรหากทุกสิ่งที่พวกเขาส่งให้กัน โดยคำจำกัดความแล้ว ต้องผ่านช่องทางที่อาจมีใครบางคนกำลังแอบฟังอยู่? การตกลงรหัสในช่องทางเดียวกับที่จะใช้ในภายหลังดูเหมือนจะเป็นไปไม่ได้: หากผู้โจมตีได้ยินตอนที่ตกลงกัน เขาจะสามารถถอดรหัสทุกสิ่งที่ตามมาได้ เป็นเวลาหลายทศวรรษที่วิทยาการรหัสลับแบบดั้งเดิมแก้ปัญหานี้ด้วยวิธีที่ยากลำบาก: รหัสจะถูกส่งมอบด้วยตัวเอง ก่อนเริ่มใช้งาน ในการพบปะทางกายภาพ เอกอัครราชทูตจะถือกระเป๋าใส่รหัสที่เย็บติดกับซับในเสื้อโค้ทของพวกเขา

ในอีเมลร่วมสมัย วิธีการแก้ปัญหานั้นไม่สามารถขยายขนาดได้ หากเราต้องเดินทางไปยังบ้านของแต่ละคนที่เราตั้งใจจะสื่อสารด้วยแบบเข้ารหัสด้วยตนเอง เราคงไม่ได้คุยกับใครเลย คำถามที่ชุมชนวิทยาการรหัสลับตั้งขึ้นเมื่อห้าสิบปีก่อนคือ: เป็นไปได้ไหมที่คนสองคนที่ไม่รู้จักกันและแชร์แค่ช่องทางสาธารณะร่วมกัน จะสามารถตกลงความลับในช่องทางสาธารณะเดียวกันนั้นได้ โดยที่ไม่มีใครที่แอบฟังช่องทางนั้นจะล่วงรู้ได้?

ความสง่างามของ Diffie-Hellman

ในปี 1976 นักคณิตศาสตร์สองคนชื่อ Whitfield Diffie และ Martin Hellman ได้พิสูจน์ในสิ่งที่ดูเหมือนจะเป็นไปไม่ได้: ว่าคนสองคน ที่คุยกันผ่านช่องทางสาธารณะเท่านั้น — ช่องทางที่ใครๆ ก็ได้ยินทุกอย่างที่พวกเขาพูด — สามารถตกลงรหัสผ่านลับได้โดยที่ผู้ฟังคนใดก็ไม่สามารถค้นพบได้ มันฟังดูเหมือนเวทมนตร์ แต่มันไม่ใช่: มันคือคณิตศาสตร์ การแลกเปลี่ยนรหัส Diffie-Hellman ตามที่เป็นที่รู้จักตั้งแต่นั้นมา คือรากฐานของการสื่อสารแบบเข้ารหัสเกือบทั้งหมดบนอินเทอร์เน็ต และการใช้งานอย่างเข้มข้นเป็นเวลาครึ่งศตวรรษรวมถึงการตรวจสอบทางวิชาการทั่วโลกยืนยันความแข็งแกร่งของมัน ใครก็ตามที่ต้องการเห็นสัญชาตญาณทางสายตาหรือคณิตศาสตร์สามารถอ่านต่อได้ ใครก็ตามที่เลือกจะเชื่อมั่นว่ามันใช้งานได้ก็สามารถอ่านต่อได้โดยไม่เสียอรรถรสของบทความ

สำหรับใครที่ต้องการจินตนาการเป็นภาพ มีการเปรียบเทียบที่เป็นที่รู้จักด้วยสี ลองนึกภาพว่าอลิซและบรูโนตกลงกันอย่างเปิดเผยเรื่องสีพื้นฐาน — สมมติว่าเป็นสีเหลือง — ต่อหน้าอีวาที่กำลังแอบฟังพวกเขาอยู่ แต่ละคนเลือกสีลับสีที่สองเป็นการส่วนตัวและผสมความลับของตนเข้ากับสีเหลือง อลิซได้สีส้มเฉพาะตัว บรูโนได้สีเขียวเฉพาะตัว พวกเขาแลกเปลี่ยนผลลัพธ์กันต่อหน้าอีวา ตอนนี้แต่ละคนผสมสีที่ได้รับเข้ากับความลับของตัวเอง และทั้งคู่ก็ได้สีสุดท้าย

สีเดียวกัน เพราะลำดับของการผสมไม่มีผล อีวาเห็นสีเหลืองและส่วนผสมขั้นกลางสองอย่าง แต่ไม่เห็นความลับ หากไม่มีความลับอย่างใดอย่างหนึ่ง เธอไม่สามารถเข้าถึงสีสุดท้ายได้ คณิตศาสตร์จริงๆ จะเปลี่ยนสีเป็นการยกกำลังในกลุ่มมอดุลาร์หรือเส้นโค้งวงรี แต่แนวคิดยังเหมือนเดิม: ความลับที่ใช้ร่วมกันถูกสร้างขึ้นในที่สาธารณะโดยไม่มีใครในช่องทางนั้นสามารถสร้างมันขึ้นมาใหม่ได้

ในวิชาเลขคณิต สำหรับผู้ที่ชอบดูความลึกของกลไก: อลิซเลือกตัวเลขลับ a บรูโนเลือก b พวกเขาแลกเปลี่ยน g^a และ g^b อย่างเปิดเผยผ่านช่องทาง อลิซคำนวณ $(g^b)^a$ และบรูโนคำนวณ $(g^a)^b$ ทั้งคู่จะได้ g^{ab} ตัวเดียวกัน อีวาเห็น g, g^a และ g^b ผ่านช่องทาง แต่การกู้คืน a จาก g^a — หรือที่เรียกว่าปัญหาลอการิทึมแบบไม่ต่อเนื่อง (Discrete Logarithm Problem) — ต้องใช้เวลาในการคำนวณทางดาราศาสตร์ที่มากกว่าอายุของจักรวาล เมื่อเลือก g ในกลุ่มทางคณิตศาสตร์ที่เหมาะสม

สำหรับใครก็ตามที่ต้องการตรวจสอบด้วยตัวเลขเล็กๆ การแลกเปลี่ยน Diffie-Hellman สามารถทำความเข้าใจได้ทั้งหมดผ่านตัวเลขที่เล็กพอที่จะคำนวณด้วยมือได้ ใครที่ไม่อยากยุ่งเกี่ยวกับคณิตศาสตร์สามารถข้ามส่วนนี้ไปได้โดยไม่เสียอรรถรสของบทความ ส่วนใครที่อยากเห็นกลไกการทำงานทีละขั้นตอนจะพบมันที่นี่ **กฎสาธารณะ** ที่ทุกคนสามารถอ่านได้: จำนวนเฉพาะ $p = 11$ (ใน Diffie-Hellman ของจริงจะมีประมาณสามร้อยหลัก เราใช้สิบเอ็ดเพื่อให้การคำนวณอยู่ในหน้าเดียว), ฐาน $g = 2$, และข้อตกลงที่ว่าการคำนวณทางคณิตศาสตร์ทั้งหมดจะเป็นแบบ *โมดูโล p (modulo p)* — คุณคำนวณ หารด้วย p และเก็บเศษไว้ เหมือนนาฬิกาที่มีสิบเอ็ดตำแหน่งซึ่งจะกลับไปที่ศูนย์เมื่อผ่านเลขสิบ **การเลือกส่วนตัว** คนละหนึ่งค่าและไม่มีการเปิดเผย: อลิซเลือก $a = 4$ บรูโนเลือก $b = 7$

ขั้นตอนที่ 1 อลิซคำนวณ $2^4 = 16$ จากนั้น $16 \bmod 11 = \mathbf{5}$ เธอส่งเลขห้าไป อีวาจดมันไว้

ขั้นตอนที่ 2 บรูโนคำนวณ $2^7 = 128$ จากนั้น $128 \bmod 11 = \mathbf{7}$ เขาส่งเลขเจ็ดไป อีวาก็จดมันไว้เช่นกัน หลังจากการส่งสองครั้ง สมุดบันทึกของอีวามีข้อมูลสี่ชิ้น: $p = 11$, $g = 2$, A = 5, B = 7 เธอขาดตัวเลขที่ใช้ร่วมกันซึ่งอลิซและบรูโนกำลังจะดึงออกมา — และเป็นสิ่งที่อีวาจะไม่สามารถสร้างขึ้นใหม่ได้

ขั้นตอนที่ 3 อลิซเอาเลขเจ็ดที่บรูโนส่งมาให้เธอยกกำลังด้วยเลขชี้กำลังส่วนตัวของเธอ $a = 4$ เพื่อหลีกเลี่ยงการจัดการกับ $7^4 = 2401$ การคำนวณจะแบ่งเป็นส่วนๆ โดยใช้โมดูโลในแต่ละขั้นตอน:

$7^2 = 49$

$49 \bmod 11 = 5$

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

$25 \bmod 11 = 3$

อลิซได้ตัวเลข **3**

ขั้นตอนที่ 4 บรูโนเอาเลขห้าที่อลิซส่งมาให้เขายกกำลังด้วยเลขชี้กำลังส่วนตัวของเขา $b = 7$ แบ่งเป็นส่วนๆ อีกครั้ง:

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

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

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

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

บรูโนก็ได้ **3** เช่นกัน

ทั้งคู่ได้ตัวเลขเดียวกันคือ 3 โดยทำงานคู่ขนานกัน ไม่มีใครส่งเลขชี้กำลังส่วนตัวของตัวเองเลยในเวลาใดเวลาหนึ่ง อลิซไม่รู้ว่า $b = 7$ บรูโนไม่รู้ว่า $a = 4$ แต่ละคนใช้ค่าสาธารณะที่อีกฝ่ายส่งมารวมกับเลขชี้กำลังส่วนตัวของตนเอง และพวกเขาพบกันที่ปลายทางเดียวกัน **ทำไมพวกเขาถึงได้ตัวเลขเดียวกัน?** สิ่งที่แต่ละคนคำนวณ: อลิซ $(g^b)^a = 2^{7 \times 4} = 2^{28} \bmod 11$ บรูโน $(g^a)^b = 2^{4 \times 7} = 2^{28} \bmod 11$ มันเป็นจำนวนเดียวกันเพราะลำดับการคูณของเลขชี้กำลังไม่สำคัญ ($7 \times 4 = 4 \times 7$) แต่ละคนมาถึงปลายทางเดียวกันด้วยเส้นทางที่ต่างกัน

แล้วอีวาล่ะ? เธอมี $p = 11$, $g = 2$, A = 5, B = 7 ในสมุดบันทึก และเธออยากได้ 3 เพื่อคำนวณหามัน เธอจำเป็นต้องรู้ a หรือ b — แต่ไม่มีตัวใดส่งผ่านช่องทางนี้เลย วิธีเดียวของเธอคือถามตัวเองว่า: «สำหรับเลขชี้กำลัง a ใดที่ทำให้ $2^a \bmod 11 = 5$?» ด้วย p ที่เล็กขนาดนี้ เธอสามารถลอง 0, 1, 2, 3, 4... และหามันเจอในเวลาไม่ถึงนาที แต่เมื่อแทนที่ 11 ด้วยจำนวนเฉพาะขนาดสามร้อยหลัก พื้นที่ของเลขชี้กำลังที่เป็นไปได้จะมีองค์ประกอบมากกว่าจำนวนอะตอมในจักรวาลที่สังเกตได้ **ปัจจุบันยังไม่มีอัลกอริทึมใดที่มนุษยชาติรู้จักที่จะสามารถท่องไปในพื้นที่นั้นได้ในเวลาน้อยกว่าหลายพันล้านปี** นี่คือสิ่งที่เรียกว่า *ปัญหาลอการิทึมแบบไม่ต่อเนื่อง (Discrete Logarithm Problem)*: เดินหน้าไปได้ง่าย แต่เป็นไปไม่ได้ในทางคำนวณที่จะย้อนกลับ และนั่นคือเหตุผลที่การเข้ารหัสยังคงทนทานได้แม้ว่าอีวาจะติดตามบทสนทนาทั้งหมดแบบตัวอักษรต่อตัวอักษรก็ตาม

ส่วนผสมง่ายๆ สามอย่าง — คณิตศาสตร์แบบหน้าปัดนาฬิกา, การยกกำลัง, และคุณสมบัติการสลับที่ของการคูณ ($a \cdot b = b \cdot a$) — เมื่อรวมกันทำให้เกิดโปรโตคอลที่มนุษยชาติครึ่งหนึ่งต้องพึ่งพาทุกวันสำหรับการสื่อสารส่วนตัว ไม่มีส่วนใดในสามส่วนนี้ที่ดูพิเศษหากแยกกัน สิ่งชี้ขาดคือการประกอบเข้าด้วย

กัน

จาก Diffie-Hellman สู่โปรโตคอล Signal

การเข้ารหัสแบบปลายทางถึงปลายทางที่แอปพลิเคชันส่งข้อความระดับมืออาชีพใช้ในปัจจุบันนั้น แทบไม่มีข้อยกเว้น โดยอาศัยการแลกเปลี่ยนรหัส Diffie-Hellman เวอร์ชันที่สง่างามและแข็งแกร่ง โปรโตคอล Signal ที่ออกแบบโดย Trevor Perrin และ Moxie Marlinspike ระหว่างปี 2013 ถึง 2016 คือข้อมูลอ้างอิง มันรวมแนวคิดหลักสองประการเข้าด้วยกัน ประการแรก คือการแลกเปลี่ยนรหัสในเส้นโค้งวงรี (X25519) ซึ่งสร้างความลับที่ใช้ร่วมกันเริ่มต้นระหว่างอุปกรณ์สองเครื่อง ประการที่สอง คือสิ่งที่เรียกว่า Double Ratchet — เฟืองคู่ — ซึ่งจะต่ออายุรหัสโดยอัตโนมัติในทุกข้อความ เพื่อให้การที่อุปกรณ์ถูกเจาะข้อมูลในวันนี้ ไม่สามารถถอดรหัสข้อความในอดีตได้ รวมถึงข้อความในอนาคตเมื่อเฟืองถูกหมุนไปแล้ว

ในภาษา Zig การแลกเปลี่ยน X25519 ที่สร้างความลับที่ใช้ร่วมกันระหว่างอุปกรณ์สองเครื่องนั้นบรรจุอยู่ในหกบรรทัด โดยใช้ไลบรารีมาตรฐาน:

```
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)
```

สิ่งที่เกิดขึ้นในหกบรรทัดนั้น: รหัสสาธารณะเดินทางอย่างเปิดเผย รหัสลับไม่เคยออกจากอุปกรณ์ที่เกี่ยวข้อง แต่ละฝ่ายจะดึงความลับขนาดสามสิบสองไบต์แบบเดียวกันออกมาจากรหัสลับของตนและรหัสสาธารณะของอีกฝ่าย ซึ่งไม่มีใครในช่องทางสามารถกู้คืนได้ ความลับนั้นจะทำหน้าที่เป็นจุดเริ่มต้น (Seed) เพื่อเข้ารหัสข้อความที่แลกเปลี่ยนกันในภายหลัง Double Ratchet ของโปรโตคอล Signal เพิ่มการหมุนเวียนของข้อมูลนั้นอย่างต่อเนื่อง เพื่อให้การถูกเจาะข้อมูลเพียงช่วงขณะหนึ่งไม่ทำให้ส่วนที่เหลือของการสนทนาถูกเปิดเผยไปด้วย

แล้วข้างใน `std.crypto.dh.X25519` มีอะไรอยู่กันแน่? ไม่มีเวทมนตร์ใดซ่อนอยู่ มันคือฟังก์ชันสั้นๆ สองตัวที่สามารถอ่านได้ทั้งหมดในไลบรารีมาตรฐานของ Zig เอง ฟังก์ชันแรกจะดึงรหัสสาธารณะมาจากรหัสส่วนตัว — «g^a» ของการแลกเปลี่ยน:

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

ในภาษาของบทความนี้: รหัสส่วนตัวถูก «คูณ» — ในความหมายของเส้นโค้งวงรี ไม่ใช่คณิตศาสตร์พื้นฐาน — ด้วยจุดฐาน (Base Point) ของเส้นโค้ง Curve25519 และผลลัพธ์จะถูกแปลงให้อยู่ในรูปอนุกรม (Serialized) เป็นสามสิบสองไบต์ การทำงาน `clampedMul` คือเวอร์ชันที่แข็งแกร่งขึ้นของการคูณสเกลาร์นั้น: โดยรวมเอาการป้องกันที่ชุมชนวิทยาการรหัสลับเพิ่มเข้ามาตลอดหลายปีเพื่อต้านทานรูปแบบการโจมตีที่เป็นที่รู้จัก เนื้อหาฟังก์ชันเพียงสองบรรทัด

ฟังก์ชันที่สองรวมรหัสส่วนตัวของคุณเข้ากับรหัสสาธารณะที่อีกฝ่ายส่งมาให้คุณ มันคือ «$(g^b)^a$» ของการแลกเปลี่ยน ซึ่งสร้างความลับที่ใช้ร่วมกันขนาดสามสิบสองไบต์ที่คุณทั้งคู่ไม่เคยส่งออกไปเลย:

```
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();
}
```

อีกสองบรรทัด รหัสสาธารณะที่ได้รับจะถูกตีความเป็นจุดบนเส้นโค้ง และ «คูณ» ด้วยรหัสส่วนตัวของตนเอง ด้วยคุณสมบัติการสลับที่ของการคำนวณบนเส้นโค้ง — คล้ายกับการสลับที่ของการคูณเลขชี้กำลังที่เราเห็นในตัวอย่างตัวเลข — ทั้งสองฝ่ายจะลงเอยด้วยจุดในรูปอนุกรมเดียวกัน: ซึ่งก็คือความลับที่ใช้ร่วมกันที่บทความพูดถึงพอดี

นั่นคือทั้งหมด สิ่งที่ดูเหมือนเวทมนตร์ในแอปพลิเคชัน แท้จริงแล้วคือฟังก์ชันที่มีความยาวเพียงสามบรรทัดสองตัว ความซับซ้อนทางเทคนิคกระจุกตัวอยู่ในการดำเนินการเดียวคือ `clampedMul` ซึ่งเขียนไว้ด้านล่างในไลบรารีมาตรฐานเดียวกัน ได้รับการทบทวนมานานหลายทศวรรษโดยชุมชนวิทยาการรหัสลับระดับนานาชาติ และเปิดให้ทุกคนที่ต้องการอ่านแบบทีละตัวอักษร ไม่มีกล่องดำ (Black Box) ซ่อนอยู่ ไม่ว่าจะเป็นในแอปพลิเคชันของเราหรือในไลบรารีมาตรฐานของ Zig มีแต่รหัสโอเพนซอร์สที่มนุษย์สามารถเข้าใจได้ โดยเลือกจังหวะเวลาในการเจาะลึกได้เอง

สิ่งที่การเข้ารหัสแบบปลายทางถึงปลายทางปกป้อง

สิ่งที่ E2EE ปกป้องได้ดี โดยสมมติว่ามีการนำไปใช้งานอย่างถูกต้อง คือเนื้อหาของข้อความระหว่างการส่งผ่าน เซิร์ฟเวอร์คนกลางที่รับและส่งต่อข้อมูลที่เข้ารหัสจะเห็นชุดของไบต์ที่อ่านไม่รู้เรื่อง ผู้โจมตีที่เข้าถึงสายเคเบิล เราเตอร์ หรือจุดกระจายสัญญาณไวไฟจะเห็นสิ่งเดียวกัน ผู้ให้บริการที่เก็บสำเนาของทราฟฟิกไว้จะไม่สามารถอ่านมันได้ในภายหลัง รัฐบาลที่สั่งให้ผู้ให้บริการส่งมอบเนื้อหาจะได้รับไบต์ที่อ่านไม่รู้เรื่องแบบเดียวกับที่เซิร์ฟเวอร์มีตั้งแต่แรก

ในทางปฏิบัติแล้ว นี่เป็นเรื่องสำคัญมาก มันคือความแตกต่างระหว่างการเขียนจดหมายใส่ในซองทึบ กับการเขียนลงบนไปรษณียบัตร ทั้งสองอย่างส่งถึงที่หมายเหมือนกัน แต่มีเพียงอย่างเดียวที่รักษาเนื้อหาจากบุรุษไปรษณีย์ได้

สิ่งที่การเข้ารหัสแบบปลายทางถึงปลายทางไม่ได้ปกป้อง

เป็นเรื่องที่ควรค่าแก่การเรียนรู้ไว้อย่างดีเช่นกัน E2EE ไม่ได้ปกป้องข้อมูลเมตา (Metadata): เซิร์ฟเวอร์ยังคงรู้ว่าผู้ใช้ A ส่งข้อมูลให้ผู้ใช้ B ในเวลาใด ด้วยความถี่เท่าใด และจากที่ไหน แม้ว่าจะไม่รู้ว่าคุยเรื่องอะไรก็ตาม ข้อมูลเมตาเหล่านี้ตามที่เราได้โต้แย้งไปแล้วใน *การเข้ารหัสไม่ได้หมายถึงความเป็นส่วนตัว* มักจะเปิดเผยข้อมูลได้มากกว่าเนื้อหา การรู้ว่าใครบางคนโทรหาสำนักงานกฎหมายที่เชี่ยวชาญเรื่องการหย่าร้างในคืนวันศุกร์เวลา 22:00 น. เป็นเวลาสามสิบนาทีนั้นบอกเล่าเรื่องราวที่เนื้อหาการโทรไม่เคยบอก มันเป็นสถานการณ์เดียวกับการเห็นคนเดินเข้าออกคลินิกมะเร็งหลายๆ ครั้ง: ไม่จำเป็นต้องได้ยินสิ่งที่คุยกันข้างในก็จินตนาการได้ว่าเกิดอะไรขึ้น ข้อมูลเมตาเพียงอย่างเดียวที่แยกส่วนออกมาอาจไม่หมายถึงอะไร แต่เมื่อนำข้อมูลหลายส่วนมาไขว้กัน มันจะวาดภาพสิ่งที่คล้ายคลึงกับความจริงมากเกินไป E2EE ไม่ได้ปกป้องที่ปลายทาง: หากอุปกรณ์ของผู้รับถูกเจาะโดยโปรแกรมอันตราย ข้อความจะถูกถอดรหัสตามปกติสำหรับผู้รับคนนั้น และโปรแกรมอันตรายจะอ่านมันได้ E2EE ไม่ได้ปกป้องตัวตนของคู่สนทนาด้วยตัวเอง: หากอลิซเชื่อว่าเธอกำลังคุยกับบรูโน แต่ผู้โจมตีได้เข้ามาแทรกแซงตั้งแต่เริ่มต้น (การโจมตีแบบ *man in the middle*) และโปรโตคอลไม่มีการตรวจสอบที่เป็นอิสระ ทั้งสองฝ่ายจะจบลงด้วยการคุยกับผู้บุกรุกโดยคิดว่ากำลังคุยกันเอง

มีประการที่สี่ที่ควรกล่าวไว้อย่างชัดเจน E2EE ไม่ได้ขัดขวางผู้ให้บริการที่อ้างว่าเสนอการเข้ารหัสนี้จากการเก็บสำเนาข้อความที่ไม่ได้เข้ารหัสไว้ในระบบของตนเองด้วย คำกล่าวที่ว่า «ข้อความของฉันได้รับการเข้ารหัสแบบปลายทางถึงปลายทาง» และคำกล่าวที่ว่า «ผู้ให้บริการไม่ได้เก็บเนื้อหาของฉันไว้» นั้นไม่ใช่สิ่งเดียวกัน แอปพลิเคชันสามารถทำตามประการแรกได้ในขณะที่ละเมิดประการที่สอง ซึ่งเราได้เห็นในพาดหัวข่าวซ้ำแล้วซ้ำเล่าตั้งแต่ปี 2018 ผู้ใช้ไม่มีวิธีทางเทคนิคที่จะแยกแยะกรณีหนึ่งออกจากอีกกรณีหนึ่งได้โดยไม่ต้องมีการสืบสวนจากผู้เชี่ยวชาญ เว้นแต่โค้ดฝั่งไคลเอนต์จะสามารถตรวจสอบได้ กรณีที่เป็นที่รู้จักมากที่สุดในหมู่ประชาชนทั่วไป: WhatsApp เข้ารหัสข้อความแบบปลายทางถึงปลายทางระหว่างการส่งผ่าน แต่หากผู้ใช้เปิดใช้งานการสำรองข้อมูลใน iCloud หรือ Google Drive โดยไม่มีการเข้ารหัสเพิ่มเติม สำเนานั้นจะถูกจัดเก็บในรูปแบบที่อ่านได้ในโครงสร้างพื้นฐานของบุคคลที่สาม และการเข้ารหัสจะถูกทำลายลงที่ปลายทางของผู้ใช้เอง

คำถามที่ผู้ให้บริการไม่ต้องการได้ยิน

แอปพลิเคชันที่อ้างว่าเข้ารหัสแบบปลายทางถึงปลายทางสามารถทำสิ่งใดสิ่งหนึ่งในสามสิ่งนี้เกี่ยวกับรหัสได้ในทางเทคนิค:

1. **รหัสจะอยู่ที่อุปกรณ์เท่านั้น** รหัสจะถูกสร้างขึ้นและอยู่ที่อุปกรณ์ของผู้ใช้โดยเฉพาะ ผู้ให้บริการไม่รู้จักและไม่จัดเก็บรหัสเหล่านั้น นี่คือกรณีที่เหมาะสมที่สุด
2. **ผู้ให้บริการสามารถเข้าถึงได้หากต้องการ** ผู้ให้บริการถือรหัสของผู้ใช้ไว้ (หรือสามารถสร้างขึ้นใหม่ได้ตามต้องการ) และเก็บไว้ในฐานข้อมูลของตน หากต้องการหรือถูกบังคับ พวกเขาสามารถอ่านเนื้อหาได้ นี่คือกรณีของบริการ «คลาวด์» ส่วนใหญ่
3. **ผู้ให้บริการไม่สามารถเข้าถึงได้โดยการออกแบบ แต่ควบคุมการเข้าถึงได้** ผู้ให้บริการไม่ได้ถือรหัสไว้ แต่ควบคุมแอปพลิเคชันที่สร้างรหัสเหล่านั้น หากถูกบังคับ พวกเขาสามารถส่งการอัปเดตที่เป็นอันตรายเพื่อดักจับรหัสหรือเนื้อหาก่อนการเข้ารหัสได้ นี่คือกรณีของบริการ E2EE เชิงพาณิชย์จำนวนมาก

ดังนั้น คำถามเชิงปฏิบัติการไม่ใช่ว่าสิ่งนั้นถูกเข้ารหัสหรือไม่ แต่คือใครเป็นผู้ควบคุมอุปกรณ์และซอฟต์แวร์ที่จัดการรหัส ใน Solo2 รหัสจะอยู่ใน «ห้องนิรภัย» ของคุณเท่านั้น (IndexedDB ที่เข้ารหัสด้วยรหัสผ่านของคุณ) และซอฟต์แวร์เป็นรหัสโอเพนซอร์สที่ตรวจสอบได้

สำหรับผู้อ่านมืออาชีพ

การเข้ารหัสแบบปลายทางถึงปลายทางคือเครื่องมือสำหรับอธิปไตยทางดิจิทัล แต่เช่นเดียวกับเครื่องมือทุกชนิด ประสิทธิภาพของมันขึ้นอยู่กับมือที่ถือมันและพื้นดินที่มันเหยียบอยู่

1. รหัสคริปโตกราฟีถูกสร้างขึ้นที่ไหนและเก็บอยู่ที่ใดทางกายภาพ? หากผู้ให้บริการสามารถเข้าถึงรหัสเหล่านั้นได้ (แม้เพียงชั่วคราว แม้จะอยู่ภายใต้ข้ออ้างของการกู้คืนข้อมูล) E2EE ก็เป็นเพียงแค่ในนามเท่านั้น
2. มีการตรวจสอบผู้สนทนาอย่างอิสระ (หมายเลขความปลอดภัย, คิวอาร์โค้ด, การเปรียบเทียบภายนอกช่องทางสื่อสาร) ที่ป้องกันการโจมตีแบบคนกลางในระหว่างการตั้งค่าการสนทนาหรือไม่?
3. รหัสของไคลเอนต์สามารถตรวจสอบได้หรือไม่ — เปิดเผย เผยแพร่ ทำซ้ำได้ — หรือต้องเชื่อถือคำพูดของผู้ให้บริการเกี่ยวกับสิ่งที่ไคลเอนต์ทำจริง?

4. บริการสร้างและเก็บรักษาข้อมูลเมตาใดบ้าง และเก็บไว้นานเท่าใด? แม้ว่าเนื้อหาจะถูกปิดบัง แต่ข้อมูลเมตาก็สามารถปะติดปะต่อข้อมูลที่ละเอียดอ่อนส่วนใหญ่ขึ้นมาใหม่ได้

คำถามสี่ข้อนี้ไม่ได้ถามหาข้อมูลทางเทคนิคขั้นสูง แต่ถามหาข้อมูลที่ผู้ให้บริการที่ซื่อสัตย์รายใดก็สามารถตอบได้ในเอกสารสาธารณะของตน คุณภาพและความแม่นยำของคำตอบบอกอะไรเกี่ยวกับตัวผลิตภัณฑ์ได้มากพอๆ กับตัวคำตอบเอง

การเข้ารหัสแบบปลายทางถึงปลายทาง หากทำอย่างถูกต้อง ถือเป็นโครงสร้างที่ละเอียดอ่อนที่สุดชิ้นหนึ่งที่วิทยาการรหัสลับร่วมสมัยได้มอบให้กับการใช้งานในชีวิตประจำวัน แนวคิดดั้งเดิมที่ว่าคนสองคนสามารถตกลงความลับในช่องทางสาธารณะได้ เป็นของ Whitfield Diffie และ Martin Hellman ในปี 1976 ครึ่งศตวรรษต่อมาเรายังคงอยู่ในผลลัพธ์ของมัน แต่เช่นเดียวกับสัญญาทางเทคนิคใดๆ คุณค่าของมันขึ้นอยู่กับการปฏิบัติตามจริง ไม่ใช่ที่ป้ายกำกับ คำถามของมืออาชีพที่ซื่อสัตย์ไม่ใช่ «มีการเข้ารหัสหรือไม่» แต่คือ «ใครเป็นผู้ถือรหัส» คำตอบมีผลลัพธ์ที่แตกต่างกัน ควรค่าแก่การเรียนรู้ไว้

แหล่งข้อมูลและการอ่านเพิ่มเติม

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, พฤศจิกายน 1976 บทความรากฐานของวิทยาการรหัสลับแบบกุญแจสาธารณะ
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, ข้อกำหนดสาธารณะโดย Open Whisper Systems, ฉบับปรับปรุงปี 2016 พื้นฐานของโปรโตคอล Signal และอนุพันธ์ในอุตสาหกรรม
- RFC 7748 — Elliptic Curves for Security (IETF, มกราคม 2016) ข้อกำหนดเชิงบรรทัดฐานของเส้นโค้ง X25519 และ X448 ที่ใช้ในการแลกเปลี่ยนรหัสที่ทันสมัย
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010) บทเกี่ยวกับการแลกเปลี่ยนรหัสและโปรโตคอลการเข้ารหัสที่ได้รับการรับรอง
- กฎระเบียบ (สหภาพยุโรป) 2024/1183 ว่าด้วยกรอบอัตลักษณ์ดิจิทัลของยุโรป (eIDAS 2) — สร้างกรอบการทำงานซึ่งการตรวจสอบผู้สนทนาอย่างอิสระได้รับการสนับสนุนจากสถาบัน และความแตกต่างระหว่างการเข้ารหัสในนามกับการเข้ารหัสจริงมีผลทางกฎหมายที่แตกต่างกัน

← ก่อนหน้าKill switch และการยึดกุมโดยสถาบันถัดไป →โมเดลธุรกิจในฐานะสัญญาณแห่งความไว้วางใจ

บทความล่าสุด

- การวิเคราะห์ · 18 พฤษภาคม 2026 ความเป็นส่วนตัวที่แท้จริง vs ความเป็นส่วนตัวที่ฉาบฉวย: คำถามที่คุณควรตั้งกับตัวเอง
- การวิเคราะห์ · 18 พฤษภาคม 2026 Self-hosting ในฐานะการปฏิบัติทางวิชาชีพ
- แนวคิด · 18 พฤษภาคม 2026 คำ 24 คำ: อัตลักษณ์การเข้ารหัสคืออะไร

ดาวน์โหลดบทความนี้เก็บไว้เพื่อใช้งานได้ทุกที่ที่คุณต้องการ

↓ Markdown ↓ ข้อความธรรมดา ↓ PDF

ไฟล์จะถูกดาวน์โหลดลงในอุปกรณ์ของคุณ คุณสามารถบันทึก นำเข้าสู่ Solo2 หรือแชร์ได้ทุกที่ตามต้องการ Cuadernos จะไม่กำหนดปลายทางแทนคุณ

ตราประทับครั่ง · SHA-256 0edc2ace804420216f8954d0af46d7da1b78029e9df22c34ebd43021df3caf10

Cuadernos Lacre · สิ่งพิมพ์ของ Menzuri Gestión S.L. ·
เขียนโดย R.Eugenio · เรียบเรียงโดยทีมงาน Solo2