הצפנה מקצה לקצה, ההסבר האמיתי

מה שספקים אומרים כשהם אומרים E2EE, ומה שהם לא אומרים. הסבר דידקטי על המנגנון ומגבלותיו, ללא המעטפת השיווקית.

כדי שנהיה ברורים: WhatsApp אומרת שההודעות שלך מוצפנות מקצה לקצה. זה נכון — וזה לא מספיק. אם הגיבוי עובר ל-iCloud או ל-Google Drive ללא הצפנה נוספת, ההצפנה נשברת בטלפון שלך עצמו. השאלה המבצעית אינה האם זה מוצפן, אלא היכן המפתחות שוכנים.

מה המשמעות האמיתית של הצפנה

הצפנת הודעה פירושה להפוך אותה למשהו שנראה כמו רעש לכל מי שאין לו מידע מסוים הנקרא מפתח. הפעולה מתבצעת במכשיר של השולח, ועם המפתח הנכון, היא מתבטלת במכשיר של המקבל. באמצע, ההודעה נוסעת כרצף של בייטים ללא משמעות נראית לעין. זהו הרעיון הפשוט. שאר המאמר עוסק בניואנסים שהופכים אותה, לפי העניין, לערובה אמיתית או לתווית שיווקית.

שם התואר *מקצה לקצה* — באנגלית *end-to-end*, ובקיצור E2EE — מוסיף דיוק. ההצפנה לא נעשית כדי ששרת ביניים יוכל לקרוא אותה ולמסור אותה. היא נעשית כדי שרק שני הקצוות — המכשיר של השולח והמכשיר של המקבל — יחזיקו במפתח. כל שרת שההודעה עוברת דרכו רואה את הרעש, לא את ההודעה. זהו ההבדל הטכני לעומת הצפנה במעבר, שבה התוכן נוסע מוצפן משרת אחד למשנהו, אך כל שרת שהוא עובר דרכו מפענח אותו כדי להעבירו הלאה, ומשחזר זמנית את הטקסט הגלוי.

הפרדוקס של הסוד המשותף

יש בעיה ברורה. כדי ששני אנשים יוכלו להצפין ולפענח הודעות ביניהם, שניהם זקוקים לאותו מפתח. אבל, איך הם מסכימים על המפתח הזה אם כל מה שהם שולחים אחד לשני, מעצם הגדרתו, עובר דרך ערוץ שבו מישהו יכול להאזין? הסכמה על המפתח באותו ערוץ שבו ישתמשו בו אחר כך נראית בלתי אפשרית: אם התוקף ישמע אותו בעת ההסכמה עליו, הוא יוכל לפענח את כל מה שיבוא אחר כך. במשך עשרות שנים, הקריפטוגרפיה הקלאסית פתרה זאת בדרך הקשה: המפתחות נמסרו באופן אישי, לפני תחילת השימוש בהם, במפגשים פיזיים. שגרירים נשאו תיקי מפתחות תפורים לבטנת המעיל שלהם.

באימייל המודרני, הפתרון הזה לא ניתן להרחבה. אם היינו צריכים ללכת פיזית לבית של כל אדם שאיתו אנו מתכוונים לתקשר באופן מוצפן, לא היינו מגיעים לדבר עם אף אחד. השאלה שהוצגה לפני חמישים שנה על ידי הקהילה הקריפטוגרפית הייתה זו: האם ייתכן ששני אנשים שאינם מכירים זה את זה ושחולקים רק ערוץ ציבורי יסכימו, באותו ערוץ ציבורי עצמו, על סוד שאף אחד שמאזין לערוץ לא יכול לדעת?

האלגנטיות של Diffie-Hellman

בשנת 1976, שני מתמטיקאים בשם Whitfield Diffie ו-Martin Hellman הוכיחו משהו שנראה בלתי אפשרי: ששני אנשים, המדברים רק דרך ערוץ ציבורי — ערוץ שבו כל אחד יכול לשמוע כל מה שהם אומרים — יכולים להסכים על סיסמה סודית מבלי שאף מאזין יוכל לגלות אותה. זה נשמע כמו קסם. זה לא: זה מתמטיקה. החלפת מפתחות Diffie-Hellman, כפי שהיא ידועה מאז, היא הבסיס לכל תקשורת האינטרנט המוצפנת כמעט, וחצי מאה של שימוש אינטנסיבי

ובחינה אקדמית עולמית מאשרים את חוסנה. מי שרוצה לראות את האינטואיציה הוויזואלית או את המתמטיקה יכול להמשיך לקרוא. מי שמעדיף לסמוך על כך שזה עובד יכול גם להמשיך מבלי לאבד את חוט המאמר.

למי שרוצה לדמיין זאת בתמונה, ישנה אנלוגיה ידועה עם צבעים. דמיינו שאליס ובוב מסכימים בגלוי על צבע בסיס — נניח צהוב — לעיניה של אווה, שמאזינה להם. כל אחד בוחר בסתר צבע סודי שני ומערבב את הסוד שלו עם הצהוב. אליס מקבלת כתום מסוים; בוב מקבל ירוק מסוים. הם מחליפים ביניהם את התוצאות לעיניה של אווה. עכשיו כל אחד מערבב את הצבע שהתקבל עם הסוד שלו עצמו, ושניהם מגיעים לאותו צבע סופי, כי סדר הערבובים אינו משנה. אווה ראתה את הצהוב ואת שני ערבובי הביניים, אך לא את הסודות; ללא אחד הסודות היא לא יכולה להגיע לצבע הסופי. המתמטיקה האמיתית מחליפה את הצבעים בחזקות בקבוצות מודולריות או עקומים אליפטיים, אך הרעיון הוא אותו רעיון: הסוד המשותף נבנה בפומבי מבלי שאף אחד בערוץ יוכל לשחזר אותו.

בארית‌מטיקה, למי שמעדיף לראות את המנגנון: אליס בוחרת מספר סודי a, ברונו בוחר b. הם מחליפים g^a ו-g^b בגלוי מעל הערוץ. אליס מחשבת $(g^b)^a$ וברונו מחשב $(g^a)^b$; שניהם מגיעים לאותו g^{ab}. אווה רואה את g, g^a ו-g^b עוברים בערוץ, אך שחזור a מתוך g^a — מה שנקרא בעיית הלוגריתם הדיסקרטי — דורש זמן מחשוב אסטרונומי הגבוה מגיל היקום כאשר g נבחר בקבוצה מתמטית מתאימה.

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:

$$49 = 7^2$$

$$49 \bmod 11 = 5$$

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

$$25 \bmod 11 = 3$$

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:

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

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

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

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, $(g^b)^a = 2^{7x4} = 2^{28}$ mod 11. Bruno, $(g^a)^b = 2^{4x7} = 2^{28}$ mod 11. Es la misma cantidad porque el orden de multiplicación de exponentes no importa (7 × 4 = 4 × 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.

מ-Diffie-Hellman לפרוטוקול Signal

הצפנה מקצה לקצה שבה משתמשות כיום אפליקציות מסרים מקצועיות נשענת, כמעט ללא יוצא מן הכלל, על גרסה אלגנטית ומחוזקת של החלפת מפתחות Diffie-Hellman. פרוטוקול Signal, שתוכנן על ידי Trevor Perrin ו-Moxie Marlinspike בין 2013 ל-2016, הוא נקודת הייחוס. הוא משלב שני רעיונות מרכזיים. הראשון, החלפת מפתחות בעקומים אליפטיים (X25519), המייצרת את הסוד המשותף הראשוני בין שני מכשירים. השני, ה-Double Ratchet — גלגל שיניים כפול — המחדש את המפתחות באופן אוטומטי עם כל הודעה, כך שפריצה למכשיר היום אינה מאפשרת פענוח הודעות עבר, ולא הודעות עתידיות ברגע שגלגל השיניים הסתובב.

ב-Zig, החלפת X25519 המייצרת את הסוד המשותף בין שני מכשירים נכנסת בשישה שורות, תוך שימוש בספריה הסטנדרטית:

```
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)
```

מה קורה באותן שש שורות: המפתחות הציבוריים נוסעים בגלוי. המפתחות הפרטיים אינם יוצאים לעולם מהמכשיר המתאים. כל צד גוזר, מתוך המפתח הפרטי שלו והציבורי של השני, אותו סוד של שלושים ושניים בייטים שאף אחד בערוץ אינו יכול לשחזר. הסוד הזה משמש אחר כך כזרע להצפנת ההודעות המוחלפות. ה-Double Ratchet של פרוטוקול Signal מוסיף סיבוב מתמיד של החומר הזה כדי שפריצה של רגע אחד לא תסכן את שאר השיחה.

ומה בדיוק יש בתוך `std.crypto.dh.X25519`? אין קסם נסתר. אלו שתי פונקציות קצרות שניתן לקרוא אותן במלואן בספרייה הסטנדרטית של Zig עצמה. הראשונה גוזרת את המפתח הציבורי מתוך הפרטי — ה-«g^a» של ההחלפה:

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

בשפת המאמר: המפתח הפרטי "מוכפל" — במובן האליפטי, לא במובן האריתמטי הבסיסי — בנקודת הבסיס של עקום Curve25519, והתוצאה עוברת סריאליזציה (serialization) לשלושים ושניים בייטים. הפעולה `clampedMul` היא הגרסה המוקשחת של כפל סקלרי זה: היא משלבת את אמצעי ההגנה שהקהילה הקריפטוגרפית הוסיפה לאורך השנים כדי לעמוד בפני משפחות ידועות של התקפות. שתי שורות של גוף פונקציה.

הפונקציה השנייה משלבת את המפתח הפרטי שלך עם המפתח הציבורי שהצד השני שולח לך. זהו ה-«$(g^b)^a$» של ההחלפה, אשר מייצר את הסוד המשותף של שלושים ושניים בייטים שאף אחד מכם לא שידר מעולם:

```
_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
                                                                                    {
```

עוד שתי שורות. המפתח הציבורי שהתקבל מפורש כנקודה על העקום, ו"מוכפל" במפתח הפרטי של עצמו. בזכות תכונת החילופיות (commutativity) של פעולת העקום — בדומה לחילופיות של כפל החזקות שראינו בדוגמה המספרית — שני הצדדים מסיימים עם אותה נקודה מסוראלת (serialized): בדיוק הסוד המשותף שעליו מדבר המאמר.

זה הכל. מה שנראה באפליקציה כקסם הוא, במציאות, שתי פונקציות בנות שלוש שורות כל אחת. המורכבות הטכנית מרוכזת בפעולה אחת, `clampedMul`, שכתובה בהמשך באותה ספרייה סטנדרטית, נבדקת מזה עשרות שנים על ידי הקהילה הקריפטוגרפית הבינלאומית, וזמינה לכל מי שרוצה לקרוא אותה אות אחר אות. אין שום קופסה שחורה לא באפליקציה שלנו ולא בספרייה הסטנדרטית של Zig. ישנו קוד מקור פתוח שבן אנוש יכול להבין, כשהוא בוחר את הקצב שבו הוא רוצה לצלול לתוכו.

על מה מגנה הצפנה מקצה לקצה

מה ש-E2EE מגן עליו היטב, בהנחת מימוש נכון, הוא תוכן ההודעה במעבר. שרת ביניים שיקבל ויעביר את הנתונים המוצפנים יראה רצף של בייטים בלתי מובנים. תוקף עם גישה לכבל, לנתב, לנקודת הגישה האלחוטית, יראה את אותו הדבר. ספק שירות שישמור עותקים של התעבורה לא יוכל לקרוא אותם בדיעבד. ממשלה שתורה למפעיל השירות למסור את התוכן תקבל את אותם בייטים בלתי מובנים שהיו לשרת מלכתחילה.

זה, במונחים מעשיים, הרבה מאוד. זה ההבדל בין כתיבת מכתב בתוך מעטפה אטומה לבין כתיבתו על גלויה. שניהם מגיעים. רק אחד שומר על התוכן מפני הדוור.

על מה לא מגנה הצפנה מקצה לקצה

כדאי לדעת זאת באותה מידה. ה-E2EE אינו מגן על מטא-נתונים: השרת עדיין יודע שמשתמש א' שולח נתונים למשתמש ב', באיזו שעה, באיזו תדירות ומאיפה, גם אם הוא לא יודע מה הוא אומר. המטא-נתונים האלה, כפי שכבר טענו ב-*להצפין זה לא להיות פרטי*, הם לעתים קרובות חושפניים יותר מהתוכן. הידיעה שמישהו התקשר למשרד עורכי דין המתמחה בגירושין ביום שישי בשעה 22:00 למשך שלושים דקות מספרת סיפור שתוכן השיחה מעולם לא סיפר. זהו אותו מצב כמו לראות אדם נכנס ויוצא מספר פעמים ממרפאה אונקולוגית: אין צורך לשמוע דבר ממה שנאמר בפנים כדי לדמיין מה קורה. מטא-נתון בודד עשוי לא לסמל דבר; כמה כאלה מוצלבים ביניהם מציירים משהו שדומה מדי לאמת. ה-E2EE אינו מגן על הקצוות: אם המכשיר של המקבל נפרץ על ידי תוכנה זדונית, ההודעה מפוענחת כרגיל עבור אותו מקבל והתוכנה הזדונית קוראת אותה. ה-E2EE אינו מגן מפני זהות בן השיח כשלעצמה: אם אליס מאמינה שהיא מדברת עם ברונו אך תוקף התערב בהתחלה (*man in the middle*) והפרוטוקול אינו כולל אימות עצמאי, שני הצדדים מסיימים לדבר עם הפולש במחשבה שהם מדברים זה עם זה.

יש דבר רביעי שכדאי לנסח ללא עמימות. ה-E2EE אינו מונע מספק שטוען שהוא מציע אותו לשמור, בנוסף, עותק של ההודעה ללא הצפנה במערכות שלו. הטענה «ההודעות שלי מוצפנות מקצה לקצה» והטענה «הספק אינו שומר את התוכן שלי» אינן זהות. אפליקציה יכולה לקיים את הראשונה תוך שהיא מפרה את השנייה; ראינו זאת בכותרות העיתונים שוב ושוב מאז 2018. למשתמש, אלא אם כן קוד הלקוח ניתן לאימות, אין דרך טכנית להבחין בין מקרה אחד למשנהו ללא חקירת מומחה. המקרה המוכר ביותר בציבור הרחב: WhatsApp מצפינה את ההודעות מקצה לקצה במעבר, אך אם המשתמש מפעיל את הגיבוי ב-iCloud או ב-Google Drive ללא הצפנה נוספת, העותק הזה נשמר קריא בתשתית של צד שלישי, וההצפנה נשברת בקצה של המשתמש עצמו.

השאלה שהמפעיל לא רוצה לשמוע

אפליקציה הטוענת שהיא מצפינה מקצה לקצה יכולה, מבחינה טכנית, לעשות אחד משלושה דברים לגבי המפתחות:

1. **המפתחות נמצאים רק במכשירים.** הם נוצרים ונמצאים אך ורק במכשירי המשתמשים; המפעיל אינו מכיר אותם ואינו מאחסן אותם. זהו המקרה האופטימלי.
2. **המפעיל יכול לגשת אם הוא רוצה.** למפעיל יש את המפתחות של המשתמשים (או שהוא יכול לייצר אותם כרצונו) והוא שומר אותם במסדי הנתונים שלו. אם הוא רוצה או נאלץ, הוא יכול לקרוא את התוכן. זהו המצב ברוב שירותי ה"ענן".
3. **המפעיל אינו יכול לגשת לפי התכנון, אך הוא שולט בגישה.** למפעיל אין את המפתחות, אך יש לו שליטה על האפליקציה שמייצרת אותם. אם הוא ייאלץ, הוא יכול לשלוח עדכון זדוני שילכוד את המפתחות או את התוכן לפני ההצפנה. זהו המצב בשירותי E2EE מסחריים רבים.

לכן, השאלה המבצעית אינה האם משהו מוצפן, אלא למי יש שליטה על המכשיר ועל התוכנה שמנהלת את המפתחות. ב-Solo2, המפתחות שוכנים אך ורק ב"כספת" שלך (IndexedDB מוצפנת בסיסמה שלך) והתוכנה היא קוד פתוח שניתן לאימות.

לקורא המקצועי

הצפנה מקצה לקצה היא כלי לריבונות דיגיטלית. אך כמו כל כלי, יעילותו תלויה ביד האוחזת בו ובקרקע עליה הוא נשען.

1. היכן מיוצרים המפתחות הקריפטוגרפיים והיכן הם שוכנים פיזית? אם למפעיל יש גישה אליהם (אפילו באופן זמני, אפילו תחת מסווה של שחזור), ה-E2EE הוא נומינלי בלבד.
2. האם קיים אימות עצמאי של בן השיח (מספרי אבטחה, קודי QR, השוואה מחוץ לערוץ) שמונע התקפת אדם באמצע (man-in-the-middle) במהלך ביסוס השיחה?
3. האם קוד הלקוח ניתן לביקורת — פתוח, מפורסם, ניתן לשחזור — או שהוא דורש לסמוך על מילת הספק לגבי מה שהלקוח עושה בפועל?
4. אילו מטא-נתונים השירות מייצר ושומר, ולכמה זמן? גם אם התוכן אטום, מטא-נתונים יכולים לשחזר חלק ניכר מהמידע הרגיש.

ארבע השאלות הללו אינן דורשות מידע טכני מתקדם; הן דורשות מידע שכל מפעיל ישר יכול לענות עליו בתיעוד הציבורי שלו. איכות הדיוק של התשובה אומרת על המוצר באותה מידה כמו התשובה עצמה.

הצפנה מקצה לקצה, כשהיא נעשית נכון, היא אחת המבנים העדינים ביותר שהקריפטוגרפיה העכשווית העניקה לפרקטיקה היומיומית. הרעיון המקורי — ששני אנשים יכולים להסכים על סוד בערוץ ציבורי — שייך ל-Whitfield Diffie ו-Martin Hellman, 1976; חצי מאה לאחר מכן אנו ממשיכים לחיות בתוצאותיו. אך כפי שקורה עם כל הבטחה טכנית, ערכה תלוי במימוש בפועל, ולא בתווית. שאלתו של איש המקצוע הישר אינה «האם זה מוצפן?», אלא «למי יש את המפתחות?». לתשובות יש השלכות שונות. כדאי לדעת אותן.

מקורות וקריאה נוספת

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, נובמבר 1976. המאמר המכונן של קריפטוגרפיית מפתח ציבורי.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, מפרט פומבי של Open Whisper Systems, עדכון 2016. הבסיס של פרוטוקול Signal ונגזרותיו התעשייתיות.
- RFC 7748 — Elliptic Curves for Security (IETF, ינואר 2016). המפרט התקני של עקומי X25519 ו-X448 המשמשים בהחלפות מפתחות מודרניות.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). פרקים על החלפת מפתחות ופרוטוקולי הצפנה מאומתת.
- תקנה (האיחוד האירופי) 2024/1183 בנושא מסגרת לזהות דיגיטלית אירופאית (eIDAS 2) — מייסדת מסגרות שבהן אימות עצמאי של בן השיח מקבל תמיכה מוסדית, ושבהן להבחנה בין הצפנה נומינלית להצפנה אמיתית יש השלכות משפטיות שונות.

← הקודם Kill switch ותפיסה מוסדית הבא → המודל העסקי כאות לאמון

קריאות אחרונות

- ניתוח · 18 במאי 2026 פרטיות אמיתית מול מדומה: השאלות שכדאי לשאול את עצמך
- ניתוח · 18 במאי 2026 Self-hosting כפרקטיקה מקצועית
- מושג · 18 במאי 2026 24 המילים: מהי זהות קריפטוגרפית

קחו את המאמר הזה אתכם לכל מקום שתצטרכו.

↓ Markdown ↓ טקסט פשוט ↓ PDF

הקובץ יורד למכשיר שלכם. משם תוכלו לשמור אותו, לייבא אותו ל-Solo2 או לשתף אותו היכן שתרצו. Cuadernos לא מחליטה על היעד עבורכם.

חותם שעווה · SHA-256 ac02fed63706c1c5bd5530939af6634d38458159010dd9b7b69c9eeba52a9ae2

Cuadernos Lacre · פרסום של Menzuri Gestión S.L. ·
נכתב על ידי R.Eugenio · נערך על ידי צוות Solo2.

אתר זה אינו משתמש בעוגיות (cookies) ואינו טוען משאבים מצד שלישי. הוא משתמש במונה ביקורים אנונימי באירוח עצמי (Umami, בשרת האירופי שלנו) ובמינימום ה-JavaScript הנדרש להעדפת ערכת הנושא הבהירה/כהה שלכם. ללא מעקבים, ללא פרופילינג, ללא שיתוף נתונים. אם תרצו לעקוב אחרינו: RSS.