Päästä päähän -salaus, todellinen selitys

Mitä palveluntarjoajat sanovat sanoessaan E2EE, ja mitä he jättävät sanomatta. Opetuksellinen selitys mekanismista ja sen rajoista ilman mainoskuorta.

Tehdään tämä selväksi: WhatsApp sanoo, että viestisi ovat päästä päähän -salattuja. Se on totta — eikä se riitä. Jos varmuuskopio menee iCloudiin tai Google Driveen ilman lisäsalausta, salaus murtuu omassa puhelimessasi. Toiminnallinen kysymys ei ole, onko se salattu, vaan missä avaimet sijaitsevat.

Mitä salaus todella tarkoittaa

Viestin salaaminen tarkoittaa sen muuttamista joksikin, joka näyttää kohinalta jokaiselle, jolla ei ole tiettyä avaimeksi kutsuttua tietoa. Toimenpide tehdään lähettäjän laitteessa ja oikealla avaimella se kumotaan vastaanottajan laitteessa. Välillä viesti kulkee tavujonona, jolla ei ole näkyvää merkitystä. Tämä on se yksinkertainen idea. Artikkelin loppuosa käsittelee niitä vivahteita, jotka tekevät siitä tapauksesta riippuen todellisen takuun tai pelkän markkinointietiketin.

Adjektiivi *päästä päähän* — englanniksi *end-to-end*, lyhennettynä E2EE — lisää täsmällisyyttä. Salausta ei tehdä siksi, että välissä oleva palvelin voisi lukea ja toimittaa sen. Se tehdään niin, että vain molemmilla päillä — lähettäjän laitteella ja vastaanottajan laitteella — on avain. Mikä tahansa palvelin, jonka kautta viesti kulkee, näkee kohinan, ei viestiä. Tämä on tekninen ero *siirron aikana* tapahtuvaan salaukseen, jossa sisältö kulkee salattuna palvelimelta toiselle, mutta jokainen palvelin, jonka kautta se kulkee, purkaa sen salauksen välittääkseen sen eteenpäin, palauttaen tekstin tilapäisesti selkokieliseksi.

Jaetun salaisuuden paradoksi

Tässä on ilmeinen ongelma. Jotta kaksi ihmistä voi salata ja purkaa viestejä keskenään, molemmat tarvitsevat saman avaimen. Mutta miten he sopivat tästä avaimesta, jos kaikki, mitä he lähettävät toisilleen, kulkee määritelmän mukaan kanavan kautta, jossa joku voisi olla kuuntelemassa? Avaimesta sopiminen samassa kanavassa, jossa he myöhemmin käyttävät sitä, tuntuu mahdottomalta: jos hyökkääjä kuulee sen sopimisen aikana, hän voi purkaa kaiken myöhemmän. Vuosikymmenten ajan klassinen kryptografia ratkaisi tämän vaikean kautta: avaimet toimitettiin henkilökohtaisesti ennen käytön aloittamista fyysisissä tapaamisissa. Suurlähettiläät kantoivat mukanaan avainsalkkuja, jotka oli ommeltu takin vuoriin.

Nykyaikaisessa sähköpostissa tuo ratkaisu ei skaalaudu. Jos meidän pitäisi mennä fyysisesti jokaisen sellaisen henkilön kotiin, jonka kanssa aiomme viestiä salatusti, emme pääsisi puhumaan kenenkään kanssa. Kryptografisen yhteisön viisikymmentä vuotta sitten esittämä kysymys oli tämä: onko mahdollista, että kaksi ihmistä, jotka eivät tunne toisiaan ja joilla on käytössään vain julkinen kanava, sopivat samassa julkisessa kanavassa salaisuudesta, jota kukaan kanavaa kuunteleva ei voi tietää?

Diffie-Hellmanin tyylikkyys

Vuonna 1976 kaksi matemaatikkoa nimeltään Whitfield Diffie ja Martin Hellman osoittivat jotain näennäisesti mahdotonta: että kaksi ihmistä, jotka puhuvat vain julkisen kanavan kautta — kanavan, jossa kuka tahansa voi kuulla kaiken heidän sanomansa — voivat sopia salaisesta salasanasta ilman, että kukaan kuuntelija voi saada sitä selville. Se kuulostaa taialta. Se ei ole sitä: se on matematiikkaa. Diffie-Hellman-avaimenvaihto, jollaisena se on siitä lähtien tunnettu, on käytännössä kaiken salatun internet-viestinnän perusta, ja puoli vuosisataa intensiivistä käyttöä ja maailmanlaajuista akateemista tarkastelua vahvistavat sen vankkuuden. Se, joka haluaa nähdä visuaalisen intuition tai matematiikan, voi jatkaa lukemista. Se, joka mieluummin luottaa siihen, että se toimii, voi myös jatkaa menettämättä artikkelin juonta.

Niille, jotka haluavat hahmottaa sen kuvana, on olemassa tunnettu analogia väreillä. Kuvittele, että Alice ja Bruno sopivat julkisesti perusvärin — sanotaan vaikka keltaisen — heitä kuuntelevan Evan silmien edessä. Kumpikin valitsee yksityisesti toisen salaisen värin ja sekoittaa salaisuutensa keltaiseen. Alice saa tietynlaisen oranssin; Bruno saa tietynlaisen vihreän. He vaihtavat tulokset Evan silmien edessä. Nyt kumpikin sekoittaa saamansa värin omaan salaisuuteensa, ja molemmat päätyvät samaan lopulliseen väriin, koska sekoitusjärjestyksellä ei ole väliä. Eva on nähnyt keltaisen ja molemmat välisekoitukset, mutta ei salaisuuksia; ilman jotakin salaisuuksista hän ei voi päästä lopulliseen väriin. Todellinen matematiikka korvaa värit potenssiinkoroituksilla modulaarisissa ryhmissä tai elliptisillä käyrillä, mutta idea on sama: jaettu salaisuus rakennetaan julkisesti ilman, että kukaan kanavalla voi rekonstruoida sitä.

Aritmetiikassa niille, jotka haluavat nähdä mekanismin: Alice valitsee salaisen luvun a, Bruno valitsee b. He vaihtavat g^a ja g^b avoimesti kanavalla. Alice laskee $(g^b)^a$ ja Bruno laskee $(g^a)^b$; molemmat päätyvät samaan tulokseen g^{ab}. Eva näkee g, g^a ja g^b kulkevan kanavalla, mutta a:n palauttaminen g^a:sta — niin sanottu diskreetin logaritmin ongelma — vaatii tähtitieteellisen laskenta-ajan, joka ylittää maailmankaikkeuden iän, kun g valitaan sopivassa matemaattisessa ryhmässä.

Niille, jotka haluavat tarkistaa sen pienillä luvuilla. Diffie-Hellman-vaihto voidaan käydä kokonaan läpi luvuilla, jotka ovat riittävän pieniä, jotta laskut voi tehdä käsin. Ne, jotka eivät halua syventyä aritmetiikkaan, voivat ohittaa tämän osion menettämättä artikkelin punaista lankaa; ne, jotka haluavat nähdä mekanismin toimivan vaihe vaiheelta, löytävät sen tästä. **Julkiset säännöt**, jotka kuka tahansa voi lukea: alkuluku $p = 11$ (aidossa Diffie-Hellmanissa se on noin kolmesataanumeroinen luku; käytämme lukuu yhdentoista, jotta laskut mahtuvat yhdelle sivulle), kantaluku $g = 2$ ja käytäntö, jonka mukaan kaikki aritmetiikka suoritetaan *modulo p* — lasketaan, jaetaan *p*:llä ja säilytetään jakojäännös, kuin yhdentoista tunnin kello, joka palaa nollaan ylitettyään kymmenen. **Yksityiset valinnat**, yksi kummallekin ja joita ei koskaan jaeta: Alice valitsee $a = 4$. Bruno valitsee $b = 7$.

Vaihe 1. Alice laskee $2^4 = 16$, sitten $16 \bmod 11 = \mathbf{5}$. Hän lähettää viitosen. Eva kirjoittaa sen muistiin.

Vaihe 2. Bruno laskee $2^7 = 128$, sitten $128 \bmod 11 = \mathbf{7}$. Hän lähettää seitsemän. Eva kirjoittaa sen myös muistiin. Kahden lähetyksen jälkeen Evan muistikirja sisältää neljä tietoa: $p = 11$, $g = 2$, A = 5, B = 7. Häneltä puuttuu jaettu luku, jonka Alice ja Bruno ovat juuri johtamassa — ja jota Eva ei pysty rekonstruoimaan.

Vaihe 3. Alice ottaa seitsemän, jonka Bruno hänelle lähetti, ja korottaa sen omaan yksityiseen eksponenttiinsa $a = 4$. Jotta ei tarvitsisi käsitellä lukua $7^4 = 2401$, lasku suoritetaan osissa soveltaen moduloa jokaisessa vaiheessa:

$7^2 = 49$

$49 \bmod 11 = 5$

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

$25 \bmod 11 = 3$

Alice saa luvun **3**.

Vaihe 4. Bruno ottaa viitosen, jonka Alice hänelle lähetti, ja korottaa sen omaan yksityiseen eksponenttiinsa $b = 7$. Jälleen osissa:

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

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

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

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

Myös Bruno saa luvun **3**.

Molemmat ovat saapuneet samaan lukuun, 3, työskentelemällä rinnakkain. Kumpikaan ei lähettänyt yksityistä eksponenttiaan missään vaiheessa. Alice ei tiedä, että $b = 7$; Bruno ei tiedä, että $a = 4$. Kumpikin käytti toisen lähettämää julkista arvoa yhdistettynä omaan yksityiseen eksponenttiinsa, ja he kohtasivat samassa määränpäässä. **Miksi he päätyvät samaan lukuun?** Mitä kumpikin laski: Alice, $(g^b)^a = 2^{7\times4} = 2^{28} \bmod 11$. Bruno, $(g^a)^b = 2^{4\times7} = 2^{28} \bmod 11$. Määrä on sama, koska eksponenttien kertolaskun järjestyksellä ei ole väliä ($7 \times 4 = 4 \times 7$). Kumpikin saapui eri reittiä samaan määränpäähän.

Entä Eva? Hänellä on muistikirjassaan $p = 11$, $g = 2$, A = 5, B = 7, ja hän haluaisi luvun 3. Laskeakseen sen hänen tulisi tietää a tai b — mutta kumpikaan ei ole kulkenut kanavan läpi. Hänen ainoa keinonsa on kysyä itseltään: «mille eksponentille a pätee $2^a \bmod 11 = 5$?». Näin pienellä *p*:llä hän voi kokeilla lukuja 0, 1, 2, 3, 4... ja löytää sen alle minuutissa. Mutta kun luku 11 korvataan kolmesataanumeroisella alkuluvulla, mahdollisten eksponenttien avaruudessa on enemmän alkioita kuin havaittavassa maailmankaikkeudessa on atomeja. **Ihmiskunta ei tällä hetkellä tunne yhtään algoritmia, joka pystyisi käymään tuon avaruuden läpi alle miljardeissa vuosissa.** Tämä on niin kutsuttu *diskreetti logaritmiongelma*: helppo eteenpäin, laskennallisesti mahdoton taaksepäin. Ja se on syy siihen, miksi salaus kestää, vaikka Eva olisi seurannut koko keskustelua kirjain kirjaimelta.

Kolme yksinkertaista ainesosaa — kelloaritmetiikka, potenssiinkorotus ja kertolaskun vaihdannaisuus ($a \cdot b = b \cdot a$) — yhdistettynä tuottavat protokollan, josta puolet ihmiskunnasta on riippuvainen joka päivä yksityisessä viestinnässään. Mikään näistä kolmesta osasta ei erikseen tarkasteltuna vaikuta erityiseltä. Ratkaisevaa on niiden yhdistäminen.

Diffie-Hellmanista Signal-protokollaan

Päästä päähän -salaus, jota nykyiset ammatilliset viestintäsovellukset käyttävät, perustuu lähes poikkeuksetta Diffie-Hellman-vaihdon tyylikkääseen ja vahvistettuun versioon. Signal-protokolla, jonka Trevor Perrin ja Moxie Marlinspike suunnittelivat vuosina 2013–2016, on esikuva. Se yhdistää kaksi avainideaa. Ensimmäinen on avaimenvaihto elliptisillä käyrillä (X25519), joka luo alkuperäisen jaetun salaisuuden kahden laitteen välille. Toinen on niin kutsuttu Double Ratchet — kaksoisräikkä —, joka uudistaa avaimet automaattisesti jokaisen viestin myötä, niin että laitteen kompromettoituminen tänään ei salli menneiden viestien purkamista, eikä tulevien viestien purkamista sen jälkeen kun räikkää on pyöräytetty.

Zigissä kahden laitteen välisen jaetun salaisuuden luova X25519-vaihto mahtuu kuudelle riville vakiokirjastoa käyttäen:

```
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)
```

Mitä noilla kuudella rivillä tapahtuu: Julkiset avaimet kulkevat avoimesti. Yksityiset avaimet eivät koskaan poistu kyseisestä laitteesta. Kumpikin osapuoli johtaa omasta yksityisestä ja toisen julkisesta avaimesta saman kolmenkymmenenkahden tavun salaisuuden, jota kukaan kanavalla ei voi palauttaa. Tuo salaisuus toimii myöhemmin siemenenä vaihdettavien viestien salaamiseen. Signal-protokollan Double Ratchet lisää tähän materiaaliin jatkuvan rotaation, jotta yhden hetken kompromettoituminen ei vaaranna loppua keskustelusta.

Ja mitä tarkalleen ottaen on `std.crypto.dh.X25519`:n sisällä? Ei piilotettua taikaa. Ne ovat kaksi lyhyttä funktiota, jotka voidaan lukea kokonaisuudessaan Zigin omasta standardikirjastosta. Ensimmäinen johtaa julkisen avaimen yksityisestä — vaihdon «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();
}
```

Artikkelin kielellä: yksityinen avain «kerrotaan» — elliptisessä, ei alkeis-aritmeettisessa mielessä — Curve25519-käyrän kantapisteellä, ja tulos sarjallistetaan kolmeksikymmeneksikahdeksi tavuksi. `clampedMul` -operaatio on vahvistettu versio tuosta skalaarikertolaskusta: se sisältää suojamekanismit, joita kryptografiayhteisö on lisännyt vuosien varrella vastustaakseen tunnettuja hyökkäysperheitä. Kaksi riviä funktion runkoa.

Toinen funktio yhdistää sinun yksityisen avaimen toisen osapuolen sinulle lähettämään julkiseen avaimeen. Se on vaihdon «$(g^b)^a$», joka tuottaa kolmenkymmenenkahden tavun jaetun salaisuuden, jota kumpikaan teistä ei koskaan siirtänyt:

```
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();
}
```

Kaksi riviä lisää. Vastaanotettu julkinen avain tulkitaan pisteenä käyrällä ja se «kerrotaan» omalla yksityisellä avaimella. Käyräoperaation vaihdannaisuuden vuoksi — analogisesti numeerisessa esimerkissä näkemämme eksponenttien kertolaskun vaihdannaisuuden kanssa — molemmat osapuolet päätyvät samaan sarjallistettuun pisteeseen: tarkalleen siihen jaettuun salaisuuteen, josta artikkelissa puhutaan.

Siinä kaikki. Mikä sovelluksessa näyttää taialta, on todellisuudessa kaksi kolmen rivin funktiota. Tekninen monimutkaisuus keskittyy yhteen ainoaan operaatioon, `clampedMul`, joka on kirjoitettu edempänä samassa standardikirjastossa, jota kansainvälinen kryptografiayhteisö on tarkastellut vuosikymmenten ajan, ja joka on kenen tahansa saatavilla, joka haluaa lukea sen kirjain kirjaimelta. Ei ole olemassa mustaa laatikkoa sen enempää sovelluksessamme kuin Zigin standardikirjastossakaan. On avointa lähdekoodia, jota ihminen voi ymmärtää, valiten tahdin, jolla hän haluaa siihen perehtyä.

Mitä päästä päähän -salaus suojaa

Se, mitä E2EE suojaa hyvin, olettaen oikean toteutuksen, on viestin sisältö siirron aikana. Välissä oleva palvelin, joka vastaanottaa ja välittää salatun datan, näkee sarjan käsittämättömiä tavuja. Hyökkääjä, jolla on pääsy kaapeliin, reitittimeen tai wifi-tukiasemaan, näkee saman. Palveluntarjoaja, joka säilyttää kopioita liikenteestä, ei voi lukea sitä jälkikäteen. Hallitus, joka määrää palvelun operaattorin luovuttamaan sisällön, saa samat käsittämättömät tavut, jotka palvelimella oli alun perin.

Tämä on käytännön termein paljon. Se on ero kirjeen kirjoittamisella läpinäkymättömän kuoren sisään ja sen kirjoittamisella postikorttiin. Molemmat saapuvat perille. Vain toinen säilyttää sisällön postinkantajalta.

Mitä päästä päähän -salaus ei suojaa

Tämä on syytä tietää yhtä hyvin. E2EE ei suojaa metatietoja: palvelin tietää edelleen, että käyttäjä A lähettää tietoja käyttäjälle B, mihin aikaan, millä taajuudella ja mistä, vaikka se ei tiedä, mitä sanotaan. Nämä metatiedot, kuten olemme jo argumentoineet artikkelissa *Salaaminen ei tarkoita yksityisyyttä*, ovat usein paljastavampia kuin sisältö. Tieto siitä, että joku soitti avioeroihin erikoistuneeseen asianajotoimistoon perjantaina klo 22.00 kolmenkymmenen minuutin ajan, kertoo tarinan, jota puhelun sisältö ei koskaan kertonut. Se on sama tilanne kuin nähdä ihmisen menevän ja tulevan useita kertoja onkologiaklinikalle: ei tarvitse kuulla mitään sisällä puhutusta kuvitellakseen, mitä on tapahtumassa. Yksittäinen erillinen metatieto ei välttämättä tarkoita mitään; useat ristiinviitatut piirtävät jotain, joka on aivan liian samanlaista kuin totuus. E2EE ei suojaa päätepisteitä: jos vastaanottajan laite on haittaohjelman saastuttama, viesti puretaan normaalisti kyseiselle vastaanottajalle ja haittaohjelma lukee sen. E2EE ei suojaa keskustelukumppanin identiteetiltä itsessään: jos Alice uskoo puhuvansa Brunon kanssa, mutta hyökkääjä on asettunut väliin alussa (*man in the middle*) eikä protokolla sisällä riippumatonta varmennusta, molemmat osapuolet päätyvät puhumaan tunkeilijan kanssa luullen puhuvansa toisilleen.

On neljäs asia, joka on syytä muotoilla ilman epäselvyyttä. E2EE ei estä palveluntarjoajaa, joka väittää tarjoavansa sitä, säilyttämästä lisäksi kopiota salaamattomasta viestistä omissa järjestelmissään. Väite ”viestini on päästä päähän -salattu” ja väite ”palveluntarjoaja ei säilytä sisältöäni” eivät ole sama asia. Sovellus voi täyttää ensimmäisen samalla kun se rikkoo toista; olemme nähneet tämän lehdistön otsikoissa toistuvasti vuodesta 2018 lähtien. Käyttäjällä ei ole teknistä tapaa erottaa tapausta toisesta ilman asiantuntijatutkimusta, ellei asiakkaan koodi ole todennettavissa. Suuren yleisön tunnetuin tapaus: WhatsApp salaa viestit siirron aikana päästä päähän, mutta jos käyttäjä aktivoi

varmuuskopioinnin iCloudiin tai Google Driveen ilman lisäsalausta, tuo kopio tallennetaan luettavana kolmannen osapuolen infrastruktuuriin, ja salaus murtuu käyttäjän omassa päässä.

Kysymys, jota operaattori ei halua kuulla

Sovellus, joka väittää salaavansa päästä päähän, voi teknisesti tehdä avainten osalta yhden kolmesta asiasta:

1. **Avaimet sijaitsevat vain laitteissa.** Ne luodaan ja ne sijaitsevat yksinomaan käyttäjien laitteissa; operaattori ei tunne niitä eikä tallenna niitä. Tämä on optimaalinen tapaus.
2. **Ylläpitäjä voi päästä tietoihin käsiksi, jos hän haluaa.** Ylläpitäjällä on käyttäjien avaimet (tai hän voi luoda ne haluamallaan tavalla) ja hän tallentaa ne tietokantoihinsa. Jos hän haluaa tai hänet pakotetaan, hän voi lukea sisällön. Tämä on tilanne useimmissa pilvipalveluissa.
3. **Ylläpitäjä ei suunnittelun perusteella pääse tietoihin käsiksi, mutta hän hallitsee pääsyä.** Ylläpitäjällä ei ole avaimia, mutta hän hallitsee sovellusta, joka ne luo. Jos hänet pakotetaan, hän voi lähettää haitallisen päivityksen, joka kaappaa avaimet tai sisällön ennen salausta. Tämä on tilanne monissa kaupallisissa E2EE-palveluissa.

Toiminnallinen kysymys ei siis kuulu, onko jokin salattu, vaan kenellä on laitteen ja avaimia hallinnoivan ohjelmiston hallinta. Solo2:ssa avaimet sijaitsevat ainoastaan holvissasi (salasanallasi salattu IndexedDB) ja ohjelmisto on todennettavaa avointa lähdekoodia.

Ammatilliselle lukijalle

Päästä päähän -salaus on digitaalisen suvereniteetin työkalu. Mutta kuten kaikkien työkalujen kohdalla, sen tehokkuus riippuu sitä pitelevästä kädestä ja maaperästä, johon se tukeutuu.

1. Missä kryptografiset avaimet luodaan ja missä ne fyysisesti sijaitsevat? Jos operaattori pääsee niihin käsiksi (edes tilapäisesti, edes palautuksen varjolla), E2EE on vain nimellinen.
2. Onko keskustelukumppanin henkilöllisyydestä olemassa riippumaton todennus (turvanumerot, QR-koodit, kanavan ulkopuolinen vertailu), joka estää väliintulohyökkäyksen keskustelun muodostamisen aikana?
3. Onko asiakaskoodi auditoitavissa — avoin, julkaistu, toistettavissa — vai vaatiiko se luottamista palveluntarjoajan sanaan siitä, mitä asiakasohjelma todellisuudessa tekee?
4. Mitä metatietoja palvelu luo ja säilyttää, ja kuinka kauan? Vaikka sisältö olisikin läpinäkymätöntä, metatiedoista voidaan rakentaa uudelleen suuri osa arkaluontoisista tiedoista.

Nämä neljä kysymystä eivät kysy pitkälle vietyä teknistä tietoa; ne kysyvät tietoa, johon jokainen rehellinen operaattori voi vastata julkisessa dokumentaatiossaan. Vastauksen laatu ja tarkkuus kertovat tuotteesta yhtä paljon kuin itse vastaus.

Päästä päähän -salaus on oikein toteutettuna yksi hienoimmista rakenteista, joita nykyaikainen kryptografia on tuonut jokapäiväiseen käytäntöön. Alkuperäinen idea — kaksi ihmistä voi sopia salaisuudesta julkisessa kanavassa — kuuluu Whitfield Diffielle ja Martin Hellmanille (1976); puoli vuosisataa myöhemmin elämme edelleen sen seurauksissa. Mutta kuten minkä tahansa teknisen lupauksen kohdalla, sen arvo riippuu todellisesta toteutumisesta, ei etiketistä. Rehellisen ammattilaisen kysymys ei ole "onko se salattu?", vaan "kenellä on avaimet?". Vastauksilla on erilaisia seurauksia. Ne on syytä tietää.

Lähteet ja lisälukemista

- Diffie, W.; Hellman, M. — *New Directions in Cryptography*, IEEE Transactions on Information Theory, marraskuu 1976. Julkisen avaimen salauksen perusartikkeli.
- Perrin, T.; Marlinspike, M. — *The Double Ratchet Algorithm*, Open Whisper Systemsin julkinen spesifikaatio, vuoden 2016 versio. Signal-protokollan ja sen teollisten johdannaisten perusta.
- RFC 7748 — Elliptic Curves for Security (IETF, tammikuu 2016). Nykyaikaisissa avaimenvaihdoissa käytettävien X25519- ja X448-käyrien normatiivinen spesifikaatio.
- Ferguson, N.; Schneier, B.; Kohno, T. — *Cryptography Engineering: Design Principles and Practical Applications* (Wiley, 2010). Lukuja avaimenvaihdosta ja todennetuista salausprotokollista.
- Eurooppalaista digitaalisen identiteetin kehystä (eIDAS 2) koskeva asetus (EU) 2024/1183 — luo puitteet, joissa keskustelukumppanin riippumaton todennus saa institutionaalista tukea ja joissa nimellisen ja todellisen salauksen välisellä erolla on erilaiset oikeudelliset seuraukset.

← EdellinenKill switch ja institutionaalinen valtausSeuraava →Liiketoimintamalli luottamussignaalina

Viimeaikaiset lukemiset

- Analyysi · 18. toukokuuta 2026 Todellinen vs. näennäinen yksityisyys: kysymykset, jotka kannattaa kysyä
- Analyysi · 18. toukokuuta 2026 Self-hosting ammatillisena käytäntönä
- Konsepti · 18. toukokuuta 2026 Ne 24 sanaa: mikä on kryptografinen identiteetti

Ota tämä artikkeli mukaasi minne tarvitset.

↓ Markdown ↓ Pelkkä teksti ↓ PDF

Tiedosto ladataan laitteellesi. Voit tallentaa sen, tuoda sen Solo2-sovellukseen tai jakaa sen haluamallasi tavalla. Cuadernos ei päätä tiedoston kohtaloa puolestasi.

Sinetti · SHA-256 1aa7e1fe046bb0daed13af60bfe1a6ce2101dcd924e21dbab35e79cf8d6e30ca

Cuadernos Lacre · Menzuri Gestión S.L. -julkaisu ·
kirjoittanut R.Eugenio · toimittanut Solo2-tiimi.