Engineer

Kryptering

å gjøre data uleselige uten nøkkelen

Hva det er

I kryptering kjøres lesbare data, klarteksten, gjennom en publisert algoritme sammen med en hemmelig verdi som kalles en nøkkel, og ut kommer chiffertekst. På engelsk heter det encryption, og enkelte NIST-publikasjoner bruker enciphering, altså chiffrering. Dekryptering med riktig nøkkel gir tilbake klarteksten byte for byte. At algoritmen er offentlig, er med vilje: AES, det chifferet de fleste systemer griper til, vant en åpen konkurranse i regi av NIST i 2000 og er publisert som en fri standard. Bare nøkkelen er hemmelig.

Chiffer finnes i to familier. Et symmetrisk chiffer bruker den samme nøkkelen begge veier. AES, spesifisert i FIPS 197, krypterer data i blokker på 128 bit med en nøkkel på 128, 192 eller 256 bit. ChaCha20, fra RFC 8439, er et strømchiffer som er omtrent tre ganger så raskt som AES på maskinvare uten egne AES-instruksjoner. Symmetriske chiffer er raske nok til en hel disk, men begge ender trenger den samme nøkkelen, og å få den trygt fram er den vanskelige delen.

Kryptografi med offentlig nøkkel svarer på det med et nøkkelpar, der den ene nøkkelen publiseres og den andre holdes privat. To parter kan hver kombinere sin egen private nøkkel med den andres offentlige og komme fram til den samme felles hemmeligheten uten noen gang å sende den. Operasjoner med offentlig nøkkel er trege, så virkelige systemer bruker begge deler: partene blir enige om en fersk symmetrisk nøkkel gjennom en utveksling med offentlig nøkkel, og det symmetriske chifferet frakter dataene. TLS, protokollen bak hver `https://` -adresse, virker på denne måten.

Den moderne ideen som betyr mest, er autentisering. Et rent chiffer skjuler innholdet, men lar det ligge åpent for endring: i tellermodus (CTR) snur én endret bit i chifferteksten den samme biten i klarteksten, og mottakeren dekrypterer den endrede meldingen som om den var ekte. Autentisert

kryptering med tilknyttede data, AEAD, legger til en tagg som mottakeren sjekker før den stoler på en eneste byte. AES-GCM og ChaCha20-Poly1305 er det vanlige paret, og TLS 1.3, publisert som RFC 8446 i 2018, beholdt bare AEAD-chiffer og krevde forward secrecy for hver nøkkelutveksling med offentlig nøkkel, slik at en servernøkkel som stjeles senere, ikke åpner noen sesjon som er tatt opp.

Bruk begrepet i en beslutning

Bruk «Hva er kryptering?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Kontoer, passord og tilganger ordnet slik at folk har nøyaktig den tilgangen jobben krever, og de som slutter, mister den samme dag.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Skrev 1 000 RBAC-regler for å sikre Linux-servere — 1,000 tilgangsregler skrevet for å tette et sikkerhetshull

Skrev 1 000 RBAC-regler for å sikre Linux-servere Ta kontakt →

Hvorfor det er viktig

Data blir sjelden værende på maskinvare som eieren selv styrer. De krysser andres nettverk, ligger i noen andres datasenter og forlater bygningen på bærbare maskiner. Kryptering er det som lar dem passere gjennom slike steder uten fare: en stjålet disk, en avlyttet kabel eller en lagringsbøtte som lekker, gir chiffertekst, og chiffertekst uten nøkkelen er støy.

Det flytter problemet i stedet for å fjerne det: når dataene først er kryptert, er sikkerheten deres den samme som sikkerheten til nøkkelen. NISTs anbefaling for nøkkelhåndtering, SP 800-57, sammenligner nøkler med kombinasjonen til en safe: en motstander som kjenner kombinasjonen, går rett forbi den sterkeste safen, og dårlig nøkkelhåndtering kan lett undergrave en sterk algoritme. Det meste av det virkelige arbeidet er derfor arbeid med nøkler: å lagre dem atskilt fra dataene, begrense hvor mye hver av dem beskytter, bytte dem etter en fast plan og vokte reservekopien like nøye som originalen.

Regelverket trekker den samme grensen. Personvernforordningen (GDPR), som gjelder som norsk lov gjennom personopplysningsloven, nevner pseudonymisering og kryptering av personopplysninger blant sikkerhetstiltakene i artikkel 32. Artikkel 34 lar den behandlingsansvarlige slippe å underrette hver berørt person om et brudd når opplysningene som lakk, var uleselige for alle uten autorisert tilgang til dem, og kryptering er eksempelet artikkelen gir.

Kryptering brukes to steder. Kryptering under overføring (in transit), for det meste TLS, beskytter data på linjen. Kryptering av lagrede data (at rest) beskytter dem der de ligger. Øyeblikket der dataene brukes, faller utenfor begge: en server som dekrypterer en oppføring for å arbeide med den, holder klarteksten og nøkkelen samtidig, så den som styrer den serveren, leser det den leser.

Slik ser det ut i praksis

Under overføring er det daglige arbeidet å holde sertifikatene gyldige. Da hundre Docker-applikasjoner trengte SSL/TLS-sertifikater, opprettet en ACME-basert arbeidsflyt hvert sertifikat, fornyet hvert av dem før det gikk ut, og distribuerte dem til verter med Ubuntu Linux som kjørte Apache og nginx. Poenget var å ta mennesket ut av ligningen, fordi mennesket er den delen som glemmer, og et utløpt sertifikat tar ned en app like sikkert som et ødelagt chiffer.

For lagrede data er det nyttige spørsmålet hvem som har nøkkelen. Selskapets implementering av krypterte sikkerhetskopier utenfor verten ble utformet for å utvide gjenopprettingsberedskapen utover leverandørens øyeblikksbilde. Den bruker restic, som krypterer det som lagres, med AES-256 i tellermodus og autentiserer det med Poly1305-AES. Skriptene har støtte for å sende øyeblikksbilder over SFTP til lagring hos en annen leverandør. I denne utformingen mottar leverandøren chiffertekst, mens selskapet har passfrasen i en beskyttet fil. Det beskriver de implementerte kontrolltiltakene, ikke aktiveringen av dem i drift: dokumentasjonen datert 2026-10-27 sier at timerne var deaktivert uten et konfigurert sikkerhetskopirepo, og dokumenterer ingen fullført gjenoppretting i drift. Kryptering er en egenskap ved den foreslåtte lagringsveien; å bevise at sikkerhetskopien kan gjenopprettes, krever at den separate gjenopprettingstesten kjøres.

Selv veien passfrasen tar, hadde betydning. Den leses fra en fil, fordi tjenestestyreren tolker escape-sekvenser i miljøverdier, og en passfrase med en omvendt skråstrek ville i det stille ha blitt en annen enn den som opprettet sikkerhetskopirepoet. Den implementerte månedlige gjenopprettingstesten er laget for å gjenopprette en fil med kjent innhold fra sikkerhetskopirepoet og sammenligne den med det forventede resultatet. Fordi testen krever passfrasen, ville en vellykket kjøring også bekrefte at nøkkelen fortsatt åpner dataene.

Da plattformen skulle få en vert nummer to, avgjorde det samme resonnementet hvordan de fire driftshemmelighetene ble delt opp per vert. En streng for et sikkerhetskopirepo angir en plassering. Separate restic-repoer kan

ha ulike hovednøkler selv om de bruker samme passfrase. Noen som får tak i passfrasen og kan lese de krypterte dataene og nøkkelfilene i begge repoene, kan åpne begge, så deling av passfrasen kobler eksponeringen deres sammen. Hver hemmelighet ble en tabell med én verdi per vert, med begrunnelsen skrevet ved siden av konfigurasjonen. SP 800-57 beskriver det generelle tilfellet: å begrense hva én nøkkel beskytter, begrenser skaden når den lekker.

Hvor det går galt

Nøkkelen finnes bare ett sted. Et sterkt chiffer behandler en eier som har glemt passordet, nøyaktig som en angriper. Dokumentasjonen til restic sier det rett ut: mister du passordet, er dataene dine tapt for godt. Hvert kryptert lager trenger en ekstra kopi av nøkkelen, oppbevart der det som ødelegger den første, ikke kan nå.

Nøkkelen ligger ved siden av dataene. En database som er kryptert med en nøkkel i den samme serverens konfigurasjon, eller en sikkerhetskopi der passfrasen ligger i den samme lagringsbøtten, stopper bare en tyv som tar halvparten av filene. Diskkryptering er den stillere varianten: den beskytter en disk som er tatt ut av en maskin som er slått av, mens en server i drift allerede har låst opp disken sin, og en angriper på den serveren leser klartekst.

Hemmelighold blir tatt for integritet. Et system som bygger på en ren chiffermodus, skjuler innholdet og lar likevel en angriper endre det. Bruk en AEAD-konstruksjon, og forkast enhver melding der taggen ikke stemmer, før du handler på en eneste byte av den.

En nonce blir gjentatt. AES-GCM og ChaCha20-Poly1305 tar en nonce, en verdi som må være unik for hver melding under én nøkkel. RFC 8439 kaller dette den viktigste sikkerhetsbetraktningen: to meldinger med samme nonce deler nøkkelstrøm og engangsnøkkel for autentisering, og det avslører XOR av de to klartekstene. En teller er en trygg måte å lage nonces på, så lenge én teller eier nøkkelen: to prosesser som deler en nøkkel og hver for seg begynner å telle fra null, skaper nøyaktig den gjentakelsen RFC-en advarer mot.

Et passord står i stedet for en nøkkel. Folk velger passord; chiffer trenger tilfeldige nøkler med fast lengde. En passfrase går derfor først gjennom en nøkkelavledningsfunksjon. Hos restic er det scrypt, som RFC 7914 har utformet som minnekrevende, så det forblir dyrt å gjette i stort omfang på spesialbygd

maskinvare. Lagrede innloggingspassord er et eget tilfelle: de skal lagres som hashverdier fra en slik funksjon, fordi en server som kan dekryptere dem, også kan lekke dem.

Metadataene forblir synlige. Kryptering skjuler innhold, mens størrelse, tidspunkt og adressene i hver ende fortsatt kan ses. TLS 1.3 lar en avsender fylle ut postene sine (padding) for å skjule hvor store de er, og utfyllingen er valgfri, så lengden sier ofte mer enn eieren regner med.

Les mer

- NIST Glossary: encryption — hver definisjon NIST-publikasjonene bruker, hver med kildedokumentet sitt.
- FIPS 197: Advanced Encryption Standard (AES) — spesifikasjonen av AES, med blokkstørrelsen og de tre nøkkellengdene.
- NIST SP 800-57 Part 1 Rev. 5: Recommendation for Key Management — hvordan nøkler lages, lagres, brukes og tas ut av bruk, og hva som følger når en av dem lekker.
- RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3 — protokollen bak `https://`, med poster som bare bruker AEAD, og nøkkelutveksling med forward secrecy.
- RFC 8439: ChaCha20 and Poly1305 for IETF Protocols — et strømchiffer, en autentisator og AEAD-paret de danner, med reglene for nonces.
- restic: References — hvordan et virkelig verktøy for sikkerhetskopiering avleder nøkler fra et passord og deretter krypterer og autentiserer dataene sine.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er kryptering?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/encryption/