Engineer

Kryptering

at gøre data ulæselige uden nøglen

Hvad det er

Kryptering omdanner læsbare data, klarteksten, til chiffertekst ved at køre dem gennem en offentlig algoritme sammen med en hemmelig værdi, en nøgle. Med den rigtige nøgle giver dekrypteringen klarteksten tilbage byte for byte. At algoritmen er offentlig, er med vilje: AES, som de fleste systemer bruger, vandt en åben konkurrence hos NIST i 2000 og er udgivet som en gratis standard. Krypteringen hviler alene på, at nøglen holdes hemmelig.

Algoritmerne falder i to familier. En symmetrisk algoritme bruger den samme nøgle begge veje. AES, specificeret i FIPS 197, krypterer data i blokke på 128 bit med en nøgle på 128, 192 eller 256 bit. ChaCha20 fra RFC 8439 krypterer en strøm af data og er omkring tre gange så hurtig som AES på hardware uden særlige AES-instruktioner. Symmetrisk kryptering er hurtig nok til en hel disk, men begge ender skal have den samme nøgle, og det svære er at få den sikkert frem.

Kryptografi med offentlige nøgler løser det med et nøglepar, hvor den ene nøgle offentliggøres og den anden holdes privat. To parter kan hver især kombinere deres private nøgle med den andens offentlige og nå den samme fælles hemmelighed uden at sende den. Den slags beregninger er langsomme, så rigtige systemer bruger begge dele: en udveksling med offentlige nøgler aftaler en frisk symmetrisk nøgle, og den symmetriske algoritme bærer dataene. TLS, protokollen bag hver `https://` -adresse, arbejder sådan.

Den vigtigste moderne idé er autentificering. En ren krypteringsalgoritme skjuler indholdet, men lader det stå åbent for ændringer: i tællertilstand (counter mode) vender en ændret bit i chifferteksten den samme bit i klarteksten, og modtageren dekrypterer den ændrede besked, som var den ægte. Autentificeret kryptering med tilknyttede data, AEAD, tilføjer et mærke (tag), som modtageren

tjekker, før den stoler på en eneste byte. AES-GCM og ChaCha20-Poly1305 er det almindelige par, og TLS 1.3, udgivet som RFC 8446 i 2018, beholdt kun AEAD-algoritmer og gav enhver nøgleudveksling med offentlige nøgler forward secrecy, så en servernøgle, der stjæles senere, åbner ingen optagne sessioner.

Brug begrebet i en beslutning

Brug »Hvad er kryptering?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Konti, adgangskoder og rettigheder sat op, så folk har præcis den adgang, jobbet kræver, og fratrådte mister den samme dag.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har skrevet 1.000 RBAC-regler til Linux-servere — 1,000 adgangsregler skrevet for at lukke et sikkerhedshul

Har skrevet 1.000 RBAC-regler til Linux-servere Kontakt os →

Hvorfor det betyder noget

Data bliver sjældent på hardware, ejeren selv styrer; de krydser andres netværk og ligger i andres datacentre og forlader bygningen på bærbare computere. Kryptering er det, der lader dem passere de steder sikkert: en stjålet disk, et aflyttet kabel eller en lækket bucket giver chiffertekst, og chiffertekst uden nøglen er støj.

Det flytter problemet i stedet for at fjerne det: når data er krypteret, er deres sikkerhed nøglens sikkerhed. NIST's anbefaling om nøglehåndtering, SP 800-57, sammenligner nøgler med kombinationen til et pengeskab: en modstander, der kender kombinationen, går lige forbi det stærkeste pengeskab, og dårlig nøglehåndtering kan let undergrave en stærk algoritme. Det meste af det egentlige arbejde er derfor nøglearbejde: at gemme nøgler adskilt fra dataene, begrænse, hvor meget hver enkelt beskytter, udskifte dem efter en plan og vogte reservekopien lige så nøje som originalen.

Lovgiverne trækker den samme linje. Databeskyttelsesforordningen nævner pseudonymisering og kryptering af personoplysninger blandt sikkerhedsforanstaltningerne i artikel 32. Artikel 34 fritager en dataansvarlig fra at underrette hver berørt person om et brud, når de lækkede oplysninger var uforståelige for enhver uden autoriseret adgang, og kryptering er det eksempel, artiklen giver.

Kryptering under transport, mest TLS, beskytter data på ledningen, og kryptering i hvile beskytter dem på lager. Selve brugen ligger uden for begge: en server, der dekrypterer en post for at arbejde med den, har klarteksten og nøglen på samme tid, så den, der styrer serveren, læser det samme som den.

Sådan ser det ud i praksis

Under transport er det daglige arbejde at holde certifikaterne gyldige. Da hundrede Docker-applikationer havde brug for SSL/TLS-certifikater, oprettede et ACME-workflow hvert certifikat, fornyede det, før det udløb, og udrullede det til Ubuntu-værter med Apache og Nginx. Målet var at tage mennesket ud af løkken, for mennesket er den del, der glemmer, og et udløbet certifikat tager en app ned lige så sikkert som en knækket algoritme.

For lagrede data er det nyttige spørgsmål, hvem der har nøglen. Virksomhedens implementering af krypteret sikkerhedskopiering uden for værten blev udformet til at udvide gendannelsesberedskabet ud over et udbydersnapshot. Den bruger restic, som krypterer det, der gemmes, med AES-256 i tællertilstand og autentificerer det med Poly1305-AES, og dens scripts understøtter afsendelse af snapshots over SFTP til lager hos en anden udbyder. I det design modtager udbyderen chiffertekst, og virksomheden har adgangssætningen i en beskyttet fil. Det beskriver de implementerede kontroller, ikke deres aktivering i drift: beskrivelsen dateret 2026-10-27 angiver, at timerne var deaktiveret uden et konfigureret backuplager, og dokumenterer ingen gennemført gendannelse i drift. Kryptering er en egenskab ved den foreslåede lagringsvej; at bevise, at sikkerhedskopien kan gendannes, kræver, at den separate gendannelsestest bliver kørt.

Selv adgangssætningens vej ind i programmet betød noget. Den læses fra en fil, for supervisoren behandler undvigetegn i miljøværdier: rummede adgangssætningen en omvendt skråstreg, ville den tavst være blevet ændret på vejen, og restic ville have fået en anden adgangssætning end den, backuplageret blev oprettet med. Den implementerede månedlige gendannelsestest er beregnet til at gendanne en fil med kendt indhold fra backuplageret og sammenligne den med det forventede resultat. Da testen kræver adgangssætningen, ville en vellykket kørsel også bekræfte, at nøglen stadig åbner dataene.

Da platformen var ved at få en vært mere, afgjorde det samme ræsonnement, hvordan hver driftshemmelighed blev delt per vært. En streng for et backuplager angiver en placering. Separate restic-lagre kan have forskellige hovednøgler, selv når de bruger samme adgangssætning. En person, der får fat i adgangssætningen og kan læse begge lagres krypterede data og nøglefiler, kan åbne dem begge, så deling af adgangssætningen kobler deres eksponering sammen. Hver hemmelighed blev til et opslag med værten som nøgle, og argumentet står ved siden af konfigurationen. SP 800-57 siger det generelt: at begrænse, hvad én nøgle beskytter, begrænser skaden, når den lækker.

Hvor det går galt

Nøglen findes ét sted. En stærk algoritme behandler en ejer, der har glemt adgangskoden, nøjagtig som en angriber. Dokumentationen til restic siger det ligeud: mister man adgangskoden, er dataene tabt for altid. Ethvert krypteret lager har brug for en ekstra kopi af sin nøgle, opbevaret uden for rækkevidde af det, der ødelægger den første.

Nøglen ligger ved siden af dataene. En database, der er krypteret med en nøgle i den samme servers konfiguration, eller en sikkerhedskopi, hvis adgangssætning ligger i den samme bucket, stopper kun en tyv, der tager halvdelen af filerne. Diskkryptering er den stille udgave: den beskytter et drev, der tages ud af en slukket maskine, mens en kørende server allerede har låst sin disk op, og en angriber på den server læser klartekst.

Hemmeligholdelse forveksles med integritet. Et system bygget på en ren krypteringstilstand skjuler indholdet og lader stadig en angriber ændre det. Brug en AEAD-konstruktion, og kassér enhver besked, hvis mærke ikke stemmer, før der handles på en eneste byte af den.

En nonce gentages. AES-GCM og ChaCha20-Poly1305 tager en nonce, en værdi, der skal være unik for hver besked under den samme nøgle. RFC 8439 kalder det den vigtigste sikkerhedsovervejelse: to beskeder med samme nonce deler nøglestrøm og engangsnøgle til autentificering, og det afslører XOR af de to klartekster. En tæller er en sikker måde at lave nonces på, så længe én tæller ejer nøglen: to processer, der deler en nøgle og hver tæller fra nul, giver netop den gentagelse, RFC'en advarer imod.

En adgangskode bruges som nøgle. Mennesker vælger adgangskoder; algoritmer kræver tilfældige nøgler af fast længde. En adgangssætning skal først gennem en nøgleafledningsfunktion, sådan som restic gør med scrypt, som RFC 7914 gør hukommelseskrævende, så massegætning på specialhardware forbliver dyr. Gemte login-adgangskoder er et særskilt tilfælde: de hører hjemme i sådan en funktion som hashværdier, for en server, der kan dekryptere dem, kan også lække dem.

Metadata forbliver synlige. Kryptering skjuler indhold, mens størrelse, timing og adresserne i begge ender stadig kan ses. TLS 1.3 lader en afsender tilføje fyld (padding) for at skjule, hvor store beskederne er, og fyldet er valgfrit, så længden siger ofte mere, end ejeren regner med.

Læs mere

- NIST Glossary: encryption — NIST's definitioner af begrebet, hver med sin kilde.
- FIPS 197: Advanced Encryption Standard (AES) — AES-specifikationen med blokstørrelse og nøglelængder.
- NIST SP 800-57 Part 1 Rev. 5: Recommendation for Key Management — hvordan nøgler laves, gemmes, bruges og udfases, og hvad et læk betyder.
- RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3 — protokollen bag `https://`, med AEAD-beskyttelse og forward secrecy.
- RFC 8439: ChaCha20 and Poly1305 for IETF Protocols — strømalgoritmen, autentificeringen og deres AEAD-kombination, med reglerne for nonces.
- restic: References — hvordan et backupværktøj afleder nøgler fra en adgangskode og krypterer og autentificerer sine data.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring udrulningsforløbet, driftsrisiciene og dem, der har vagten. Vi kan afgrænse en ændring med kontroller, en vej tilbage og tydeligt ansvar.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er kryptering?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

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

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/encryption/