Engineer

Operativ hemlighet

en åtkomstuppgift för program, som nyckel eller lösenord

Vad det är

En operativ hemlighet är ett värde som låter ett program, eller den som driftar det, agera på ett system, och som ger precis samma makt åt vem som helst annan som läser det. Lösenordet i en databasanslutning, en API-token, den privata halvan av ett SSH-nyckelpar och lösenfrasen som krypterar en säkerhetskopia är alla operativa hemligheter. I vardagligt tal krymper ordet till hemligheter, och arbetet med att skapa, förvara, rotera och förstöra dem kallas hantering av hemligheter, på engelska secrets management.

Två saker skiljer dem från koden runt omkring. Den första är läsaren. En användares lösenord finns i någons minne och skrivs in vid en prompt, medan en operativ hemlighet läses av ett program, ett skript för driftsättning eller ett schemalagt jobb, och ingen behöver komma ihåg den. Guiden Secrets Management Cheat Sheet från OWASP ger den därför en egen livscykel: den skapas med den minsta behörighet den behöver, roteras enligt ett schema, återkallas när den kan ha läckt och får ett utgångsdatum där plattformen tillåter det, med livslängder från minuter till år. Användares lösenord står utanför den regelbundna rotationen. Den andra saken är gränsen mellan en hemlighet och den konfiguration som pekar på den. Ett värdnamn eller sökvägen till ett repo för säkerhetskopior säger var något finns. Hemligheten är den del som släpper in den som har den, och den del vars kopior måste räknas.

Hemligheter ligger dessutom i lager. Valvet som förvarar dem behöver en egen hemlighet, och guiden påpekar att den primära hemligheten, t.ex. en molnleverantörs root-inloggning, oftast måste förvaras någon annanstans.

Använd begreppet i ett beslut

Använd ”Vad är en operativ hemlighet?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Konton, lösenord och behörigheter ordnade så att alla har exakt den åtkomst deras arbete kräver, och de som slutar förlorar den samma dag.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Skrev 1 000 RBAC-regler som säkrar Linux-servrar — 1,000 åtkomstregler, skrivna för att täppa till en säkerhetslucka

Skrev 1 000 RBAC-regler som säkrar Linux-servrar Hör av er →

Varför det spelar roll

En hemlighet fungerar för den som visar upp den. En token bär inga uppgifter om vem som var tänkt att inneha den, så varje kopia vidgar kretsen av människor och program som kan använda den. Repot är det sämsta stället för en hemlighet att hamna på: varje klon bär med sig hela historiken, så ett värde som har committats en enda gång följer med till varje bärbar dator, byggmaskin och spegling som hämtar det, och det finns kvar i historiken även sedan raden som innehöll det har tagits bort.

Standarderna gör detta till krav. OWASP Application Security Verification Standard, ASVS, kräver en lösning för hantering av hemligheter, t.ex. ett nyckelvalv, som skapar, förvarar, styr åtkomsten till och förstör hemligheterna i en backend, och slår fast att hemligheter inte får finnas i applikationens källkod eller i byggartefakter. Den kräver minsta möjliga behörighet för åtkomst till dem och, på sin högsta nivå, att de löper ut och roteras. NIST SP 800-53 lägger till kontrollförbättringen IA-5(7): okrypterade statiska autentiseringsuppgifter får inte bäddas in i applikationer eller i annan statisk lagring. Vägledningen till den täpper också till ett kryphål, eftersom ett värde som används precis så som det är lagrat räknas som okrypterat, oavsett vilken fil det ligger i.

Avvägningarna är verkliga. Ett centralt valv är ytterligare ett system som måste vara uppe just när det behövs som mest, och guiden påpekar att folk under en incident förväntar sig nya inloggningsuppgifter snabbt. Att låta flera tjänster dela på en hemlighet sparar arbete men kostar spårbarhet, eftersom ingen efter en läcka kan säga vilken innehavare som tappade den. Det manuella arbetet växer med varje ny hemlighet, och därför förordar guiden automatisk rotation. Den varnar också för att en omstart av applikationen låter en stulen inloggningsuppgift fortsätta fungera. Bara en återkallelse sätter stopp för den.

Så ser det ut i praktiken

Den här webbplatsen anger sin regel på sidan om hur företaget arbetar säkert: hemligheter ska hållas borta från repot. Inloggningsuppgifter finns i ett valv för hemligheter eller i driftsättningslagret, aldrig i källkoden och aldrig i något som ett bygge producerar, och en automatisk skanning körs vid varje commit och stoppar commiten när den hittar en.

Repots tekniska anteckningar beskriver hur. Skannern är gitleaks, som körs över arbetskatalogen vid varje commit och en gång till i den fullständiga kvalitetskontrollen, och en träff är ett hårt stopp som ingen är tänkt att vitlista bort. Katalogen med beroenden och de genererade utdataträden är undantagna, eftersom de är kopior som byggs om från grunden. Anteckningarna tar också upp ett tystare fel: när en fil försvinner mitt under en skanning skriver gitleaks ut en felrad och avslutas ändå som lyckad, så uppgiften som kör verktyget underkänner varje körning som har en felrad. Själva webbplatsen är statisk och har inga applikationshemligheter, och den driftsätts över SSH med nycklar som ägaren förvarar.

Att hålla hemligheter borta från repot flyttar dem till ett ställe som repots kontroller inte når, och en förhandskontroll för hemligheter täcker den luckan. Hemligheterna hade nyss delats upp per värd och koden var korrekt, men operatörens krypterade vault-fil, som ligger på operatörens egen maskin och utom synhåll för kvalitetsgrinden, innehöll fortfarande det gamla värdet för en enda värd. Den första konvergenskörningen nådde värddatorn i drift, körde åtta tasks och vägrade vid den nionde, efter att redan ha skrivit swap-inställningar till produktion. Svaret blev en play som inte kontaktar någon värd och inte samlar in några fakta, utan kör samma två task-filer som den riktiga konvergenskörningen mot vault-filen, på ungefär en sekund, innan provisioneringen gör något annat. Att återanvända koden i stället för att återge regeln en gång till hindrar kontrollen och konvergenskörningen från att glida isär. En felaktig hemlighet kostar nu en vägran på två sekunder på en bärbar dator.

Själva uppdelningen är övergången till ett eget värde per värd för varje operativ hemlighet. Plattformen hade en värd och skulle snart få två, och flera hemligheter hade skrivits som enskilda värden, bland dem repot för säkerhetskopior, dess lösenfras och en larmbrytare. Var och en blev en mappning med inventory-värdnamnet som nyckel, och två skäl står nedskrivna bredvid konfigurationen. Ett dödmansgrepp som delas av två värdar förblir grönt så länge någon av dem fortfarande hör av sig, så den andra värden döljer att den första har dött. Separata restic-repon kan ha olika huvudnycklar även om de

använder samma lösenfras. Den som får tag på lösenfrasen och kan läsa båda reponas krypterade data och nyckelfiler kan dock låsa upp båda. Delningen innebär alltså en gemensam risk vid en läcka i stället för att isolera värdarna. Den befintliga värden står med i alla fyra mappningarna med sitt repo och båda larmbrytarna tomma, vilket säger att kopplingen finns och att värdet fortfarande är något som operatören är skyldig.

Var det går fel

En hemlighet hamnar i en commit. En skanner flaggar ett värde i en diff, eller någon får syn på en token i en pushad fil, eftersom den låg i en konfigurationsfil eller en testfixtur när commiten gjordes. Guiden från OWASP anger ordningen för reparationen: återkalla först det exponerade värdet, rotera det sedan, och ta bort det ur koden eller loggarna först därefter. Att skriva om historiken för att bli av med commiten ändrar varje senare commit och bryter länkarna till dem, så det kräver planering. Upptäckten hör hemma före commiten, i en pre-commit-hook eller i editorn, där en träff inte kostar något.

Skanningen går förbi filer den aldrig läste. En körning slutar grön medan en del av trädet förblev ogranskad: en fil försvann mitt i genomgången, ett undantag drogs för brett, eller hemligheten har ingen igenkännbar form. Mönsterregler fångar tokens med ett fast format och missar ett lösenord som en människa har hittat på, och därför föreslår guiden mer än ett verktyg. Behandla varje fel från skannern som ett underkännande, och begränsa undantagen till kopior som ett bygge återskapar.

En hemlighet delas av många innehavare. Två värdar, två tjänster eller ett helt team använder samma värde. När det läcker kan loggarna inte säga vilken kopia som gick förlorad, och att byta ut det innebär att röra varje innehavare på en gång. SP 800-53 kräver att autentiseringsuppgifterna för grupp- och rollkonton byts så snart medlemskapet ändras, och med en hemlighet per innehavare blir det ett enda utbyte i stället för ett samordnat.

Hemligheten följer med bygget eller hamnar i en logg. En containeravbild, ett paketerat skript eller en felsökningsrad bär ut värdet från det ställe där det skulle stanna. ASVS nämner byggartefakter bredvid källkoden, och guiden ber att hemligheter hålls borta från loggar och maskeras där en rad måste hänvisa till en. Skanna det som bygget producerar lika väl som repot.

Läs mer

- OWASP Secrets Management Cheat Sheet — en hemlighets livscykel, upptäckt före commit och ordningen för reparation efter en läcka.
- OWASP Application Security Verification Standard — kraven på hantering av hemligheter i grupp 13.3: ett valv, minsta möjliga behörighet och rotation.
- NIST SP 800-53 Rev. 5 — hantering av autentiseringsuppgifter i IA-5, med IA-5(7) mot inbäddade statiska autentiseringsuppgifter.
- Gitleaks — skannern som den här webbplatsen kör över sin arbetskatalog vid varje commit.

Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är en operativ hemlighet?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/secret/