Engineer

Secret

Zugangsdaten, die Software braucht: Schlüssel oder Passwort

Was es ist

Ein Secret für den Systembetrieb ist ein Wert, mit dem Software oder ihr Betreiber in einem System handeln kann und der jedem anderen, der ihn liest, dieselben Befugnisse gibt. Dazu zählen ein Datenbankpasswort, ein API-Token, der private Schlüssel eines SSH-Schlüsselpaars und die Passphrase, die eine Sicherung verschlüsselt. Im Alltag spricht man meist einfach von Secrets. Das Erstellen, Speichern, Rotieren und Vernichten dieser Werte wird Secrets-Management genannt.

Zwei Dinge unterscheiden sie vom umgebenden Code. Das erste ist der Leser. Ein Benutzerpasswort sitzt im Gedächtnis eines Menschen und wird bei einer Aufforderung eingegeben. Ein Secret für den Betrieb wird dagegen von einem Programm, einem Bereitstellungsskript oder einem geplanten Job gelesen; niemand muss es sich merken. Das OWASP Secrets Management Cheat Sheet weist ihm deshalb einen eigenen Lebenszyklus zu: mit den geringsten nötigen Rechten erzeugt, nach Zeitplan rotiert, bei möglichem Abfluss widerrufen und, wo die Plattform es erlaubt, mit Ablaufdatum versehen. Die Lebensdauer reicht von Minuten bis zu Jahren. Benutzerpasswörter sind von dieser regelmäßigen Rotation ausgenommen. Das zweite ist die Grenze zwischen einem Secret und der darauf verweisenden Konfiguration. Ein Hostname oder ein Pfad zu einem Sicherungsrepository sagt, wo etwas ist. Das Secret verschafft dem Inhaber Zugang und ist der Teil, dessen Kopien gezählt werden müssen.

Auch Secrets sind verschachtelt. Ihr Speicher braucht ein eigenes Secret. Das Cheat Sheet weist darauf hin, dass dieses primäre Secret, etwa die Root-Zugangsdaten eines Cloudanbieters, meist anderswo aufbewahrt werden muss.

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist ein Secret?“ für eine konkrete Entscheidung. Wir können helfen, das Konzept mit dem System in Verbindung zu bringen, das Sie entwickeln oder einkaufen. Konten, Passwörter und Berechtigungen werden so geordnet, dass alle genau den beruflich erforderlichen Zugriff haben und ihn am Tag ihres Ausscheidens verlieren.

Unsere Leistungen entdecken Kontakt aufnehmen →

Die Annahmen hinter dem Begriff prüfen

Nutzen Sie die Erklärung und die verlinkten Arbeiten, um Fragen für Ihr Projekt zu formulieren. Wir können die Annahmen gemeinsam mit Ihnen prüfen, bevor ein Werkzeug ausgewählt wird. 1.000 RBAC-Regeln für sichere Linux-Server — 1,000 Zugriffsregeln zum Schließen einer Sicherheitslücke

1.000 RBAC-Regeln für sichere Linux-Server Kontakt aufnehmen →

Warum es wichtig ist

Ein Secret funktioniert für jeden, der es vorlegt. Ein Token hält nicht fest, wer es eigentlich besitzen sollte. Jede Kopie erweitert deshalb den Kreis der Menschen und Programme, die es nutzen können. Ein Quelltextrepository ist dafür der schlechteste Ort: Jeder Klon enthält die vollständige Historie. Ein einmal eingecheckter Wert wandert auf jeden Laptop, jeden Build-Runner und jeden Spiegel, der die Änderungen abruft. Er bleibt in dieser Historie, nachdem die betreffende Zeile gelöscht wurde.

Standards machen daraus Anforderungen. Der OWASP Application Security Verification Standard verlangt eine Lösung wie einen Schlüsseltresor, um Backend-Secrets zu erstellen, zu speichern, den Zugriff darauf zu kontrollieren und sie zu vernichten. Secrets dürfen weder im Anwendungsquelltext noch in Build-Artefakten enthalten sein. Für den Zugriff verlangt der Standard das Prinzip der geringsten Rechte; auf der höchsten Stufe müssen sie ablaufen und rotiert werden. NIST SP 800-53 ergänzt die Kontrollerweiterung IA-5(7): Unverschlüsselte statische Authentikatoren dürfen nicht in Anwendungen oder andere statische Speicher eingebettet werden. Die Erläuterung schließt auch ein Schlupfloch: Wird ein Wert genau in der gespeicherten Form verwendet, gilt er als unverschlüsselt, unabhängig von der Datei, in der er liegt.

Dabei gibt es echte Zielkonflikte. Ein zentraler Speicher ist ein weiteres System, das gerade dann laufen muss, wenn es am dringendsten gebraucht wird. Laut Cheat Sheet erwarten Menschen bei einem Vorfall rasch neue Zugangsdaten. Ein gemeinsames Secret für mehrere Dienste spart Arbeit, kostet aber Zuordenbarkeit: Nach einem Abfluss lässt sich nicht sagen, welcher Inhaber es verloren hat. Der manuelle Aufwand wächst mit jedem Secret. Deshalb bevorzugt das Cheat Sheet automatische Rotation. Es warnt außerdem: Gestohlene Zugangsdaten funktionieren auch nach einem Neustart der Anwendung weiter. Erst ihr Widerruf beendet das.

Wie es in der Praxis aussieht

Diese Website nennt ihre Regel auf der Seite zur sicheren Arbeitsweise des Unternehmens: Secrets bleiben aus dem Repository heraus. Zugangsdaten liegen in einem Secret-Speicher oder in der Bereitstellungsschicht, weder im Quelltext noch in irgendeinem Build-Erzeugnis. Bei jedem Commit läuft ein automatischer Scan. Ein Fund stoppt den Commit.

Die technischen Hinweise zum Repository erklären das Vorgehen. gitleaks prüft bei jedem Commit den Arbeitsbaum und erneut in der vollständigen Qualitätsprüfung. Ein Treffer ist ein harter Stopp, den niemand mit einer Ausnahmeliste umgehen soll. Der Abhängigkeitsordner und die erzeugten Ausgabebäume sind ausgeschlossen, weil sie von Grund auf neu erzeugte Kopien sind. Die Hinweise dokumentieren auch einen weniger auffälligen Fehler: Verschwindet während des Scans eine Datei, gibt gitleaks eine Fehlerzeile aus und beendet sich trotzdem erfolgreich. Deshalb scheitert die ausführende Task bei jeder Fehlerzeile. Die Website selbst ist statisch, hält keine Anwendungs-Secrets und wird über SSH mit Schlüsseln bereitgestellt, die die Eigentümerin verwahrt.

Secrets aus dem Repository herauszuhalten verlagert sie an einen Ort außerhalb seiner Prüfungen. Ein Preflight-Play für Secrets schließt diese Lücke. Die Secrets waren gerade pro Host aufgeteilt worden, und der Code war richtig. Der verschlüsselte Tresor auf dem eigenen Rechner des Betreibers, für die verbindliche Qualitätsprüfung unsichtbar, enthielt aber noch den alten Wert für einen einzelnen Host. Der erste Konvergenzlauf erreichte den Live-Host, erledigte acht Tasks und brach bei der neunten ab, nachdem die Swap-Einstellungen schon in Produktion geschrieben waren. Die Antwort war ein Play, das keinen Host kontaktiert, keine Systemdaten erfasst und zur Prüfung des Tresors dieselben zwei Taskdateien wie der echte Konvergenzlauf ausführt, in etwa einer Sekunde, bevor die Bereitstellung irgendetwas anderes tut. Derselbe Code statt einer erneuten Formulierung derselben Regel verhindert, dass Prüfung und Konvergenz auseinanderlaufen. Ein falsches Secret führt jetzt nach zwei Sekunden zu einer Ablehnung auf einem Laptop.

Die Aufteilung selbst ist der Wechsel zu einem eigenen Wert pro Host für jedes Secret des Betriebs. Die Plattform hatte einen Host und sollte bald zwei haben. Mehrere Secrets waren als Einzelwerte geschrieben, darunter das Sicherungsrepository, seine Passphrase und ein Alarmschalter. Jeder Wert wurde zu einer Map mit dem Inventory-Hostnamen als Schlüssel. Zwei Gründe stehen neben der Konfiguration. Ein von zwei Hosts geteilter Totmannschalter bleibt

grün, solange einer noch ein Lebenszeichen sendet. Der zweite verdeckt so den Ausfall des ersten. Getrennte restic-Repositories können verschiedene Masterschlüssel haben, auch wenn sie dieselbe Passphrase verwenden. Wer diese Passphrase erlangt und die verschlüsselten Daten und Schlüsseldateien beider Repositories lesen kann, kann aber beide entsperren. Die gemeinsame Passphrase gefährdet dadurch beide zugleich, statt die Hosts voneinander zu isolieren. Der vorhandene Host steht in allen vier Maps mit leeren Werten für sein Repository und beide Schalter. Das sagt: Die Einbindung besteht, der Betreiber muss den Wert aber noch liefern.

Wo es schiefgeht

Ein Secret landet in einem Commit. Ein Scanner findet einen Wert in einem Diff, oder jemand entdeckt ein Token in einer bereits hochgeladenen Datei. Beim Commit stand es in einer Konfigurationsdatei oder einer Testfixture. Das OWASP Cheat Sheet gibt die Reihenfolge vor: erst den offengelegten Wert widerrufen, dann rotieren und erst danach aus Code oder Protokollen entfernen. Die Historie zum Entfernen des Commits umzuschreiben verändert alle späteren Commits und bricht ihre Links. Das muss deshalb geplant werden. Erkennung gehört vor den Commit, in einen Pre-Commit-Hook oder den Editor, wo ein Fund nichts kostet.

Der Scan bleibt grün, obwohl er Dateien nie gelesen hat. Ein Lauf endet grün, obwohl ein Teil des Baums ungeprüft blieb: Eine Datei verschwand während des Durchlaufs, ein Ausschluss war zu weit gefasst, oder das Secret hat kein erkennbares Muster. Musterregeln erkennen Tokens mit festem Format und übersehen ein von einem Menschen erfundenes Passwort. Deshalb empfiehlt das Cheat Sheet mehr als ein Werkzeug. Behandeln Sie jeden Scannerfehler als Fehlschlag und beschränken Sie Ausschlüsse auf Kopien, die ein Build neu erzeugt.

Ein Secret dient vielen Inhabern. Zwei Hosts, zwei Dienste oder ein ganzes Team verwenden denselben Wert. Nach einem Abfluss können Protokolle nicht sagen, welche Kopie betroffen war. Für einen Austausch muss der Wert bei allen Inhabern gleichzeitig ersetzt werden. SP 800-53 verlangt, Authentikatoren von Gruppen- oder Rollenkonten bei jeder Änderung der Mitgliedschaft zu ändern. Mit einem eigenen Secret je Inhaber wird daraus ein einzelner Austausch statt einer koordinierten Aktion.

Das Secret wird mit dem Build ausgeliefert oder gelangt in ein Protokoll. Ein Containerimage, ein gebündeltes Skript oder eine Debugzeile trägt den Wert aus seinem vorgesehenen Ort heraus. ASVS nennt Build-Artefakte neben dem Quelltext. Das Cheat Sheet verlangt, Secrets aus Protokollen herauszuhalten und zu maskieren, wo eine Zeile auf eines verweisen muss. Scannen Sie neben dem Repository auch die Build-Erzeugnisse.

Weiterführende Lektüre

- OWASP Secrets Management Cheat Sheet — Lebenszyklus eines Secrets, Erkennung vor dem Commit und Reihenfolge der Behebung nach einem Abfluss.
- OWASP Application Security Verification Standard — Die Anforderungen zur Secret-Verwaltung in Gruppe 13.3: Tresor, geringste Rechte und Rotation.
- NIST SP 800-53, Revision 5 — Authentikatorenverwaltung in IA-5 mit IA-5(7) gegen eingebettete statische Authentikatoren.
- Gitleaks — Der Scanner, den diese Website bei jedem Commit über ihren Arbeitsbaum laufen lässt.

Glossar

Für Ihre tatsächlichen Rahmenbedingungen auswählen

Beschreiben Sie den Bereitstellungsweg, die Betriebsrisiken und die Bereitschaftsverantwortlichen. Wir können eine Änderung mit Prüfungen, einem Wiederherstellungsweg und klaren Zuständigkeiten abgrenzen.

Preise ansehen Kontakt aufnehmen →

Die Begriffe praktisch nutzbar machen

Fragen Sie, wie „Was ist ein Secret?“ auf Ihre Organisation zutrifft. Ein konkretes Beispiel aus Ihrem bestehenden System reicht aus, um das Gespräch zu beginnen.

Unsere Leistungen entdecken

welcome@engineer.company +45 50 10 61 84 Kopenhagen, Dänemark

Nehmen Sie diese Seite mit

Kontaktkarte

Mit Sorgfalt und Liebe gestaltet von D.

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

Copyright © 2025 – heute · Engineer ApS · Alle Rechte vorbehalten.

Gedruckt von https://engineer.company/de/glossary/secret/