Engineer

Härtung

Entfernen dessen, was ein Angreifer nutzen könnte

Was es ist

Härtung bedeutet, ein System so zu konfigurieren, dass es nur das anbietet und erlaubt, was seine Aufgabe erfordert: weniger laufende Dienste, weniger lauschende Ports, weniger Rechte für jeden Prozess und strengere Vorgaben für alles, was bleibt. Die Software ist vorher und nachher dieselbe. Es ändert sich, wie viel davon ein Eindringling erreichen kann und wie weit ihn ein einzelner Zugang trägt. Je nach eingeschränkter Schicht heißt das auch Systemhärtung, Serverhärtung, Betriebssystemhärtung oder SSH-Härtung.

Ein typischer Härtungsdurchlauf auf einem Linux-Server erfasst mehrere Schichten zugleich. Fernanmeldungen akzeptieren Schlüssel und lehnen Passwörter ab. Das root-Konto darf sich nicht direkt anmelden. Nicht verwendete Pakete werden entfernt, statt nur angehalten zu werden. Jeder dauerhaft laufende Dienst erhält einen eigenen unprivilegierten Benutzer, eine nur lesbare Sicht auf das Dateisystem und eine Liste erlaubter Systemaufrufe. Die Firewall lässt die wenigen Ports zu, die der Host für seine Aufgabe bereitstellen muss. Keiner dieser Schritte ist ungewöhnlich. Die Disziplin besteht darin, sie alle auf jedem Host umzusetzen und dauerhaft wirksam zu halten.

Es hilft, Härtung den Praktiken gegenüberzustellen, mit denen sie oft verwechselt wird. Patches beheben einen bestimmten Codefehler, sobald eine Korrektur vorliegt. Monitoring und Einbruchserkennung bemerken einen laufenden Angriff. Eine Firewall entscheidet, wer einen Port erreichen darf. Härtung entscheidet, was das Programm hinter diesem Port nach einem Zugriff tun darf und ob der Port überhaupt offen sein sollte. Veröffentlichte Baselines wie die CIS Benchmarks machen daraus Checklisten je Betriebssystem und Produkt. Als Mindestmaß sind sie nützlich, als Ziellinie irreführend.

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist Härtung?“ 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

Die eigentliche Entscheidung lautet, wofür ein begrenztes Sicherheitsbudget ausgegeben wird. Härtung ist insofern ungewöhnlich, als sie auch gegen Schwachstellen hilft, die erst nächstes Jahr entdeckt werden. Ein Patch schützt vor einer bekannten Schwachstelle. Ein gehärteter Dienst schränkt ein, was ein kompromittierter Prozess tun kann. Beispielsweise sollen Verbote, ausführbaren Speicher zu beschreiben, Raw Sockets zu öffnen oder auf Benutzerverzeichnisse zuzugreifen, die Auswirkungen einer Schwachstelle zur Ausführung von Code aus der Ferne begrenzen. Patches schließen Löcher; Härtung verringert den Nutzen jedes Lochs. Ein professioneller Betrieb braucht beides. Härtung verschafft Zeit zwischen der Veröffentlichung einer Schwachstelle und dem Eintreffen der Korrektur.

Die Kosten entstehen vor allem durch technische Arbeitszeit und das Risiko von Funktionsstörungen, weniger durch Lizenzen. Jede Einschränkung ist eine Wette darauf, dass der Dienst nie benötigt, was ihm entzogen wird. Einige dieser Wetten gehen verloren: ein Backupjob, der einmal pro Nacht läuft, eine Logrotation, die ein Signal sendet, ein Plugin, das ein externes Programm über die Shell aufruft. Deshalb gerät manuelle Härtung auf einem laufenden Server oft ins Stocken. Eine Einschränkung legt um drei Uhr morgens etwas lahm; wer damit einmal schlechte Erfahrungen gemacht hat, verschärft die Regeln nicht weiter. Als Code, mit geprüften und versionierten Einschränkungen, die bei jedem Neuaufbau identisch angewandt werden, lässt sich dieselbe Arbeit dagegen günstig wiederholen und sicher zurückrollen.

Für Auftraggeber lauten die praktischen Fragen: Wann soll die Härtung erfolgen und woran erkennt man, dass sie erfolgt ist? Die Antwort auf die erste Frage lautet: beim Aufbau, als Teil der Automatisierung, die den Host erstellt. Eine einmal auf einen laufenden Rechner angewandte Checkliste weicht innerhalb von Monaten vom tatsächlichen Zustand ab. Die zweite Frage beantworten

Nachweise aus dem laufenden System, nicht allein aus dem Konfigurationsrepository. Auditoren verlangen zunehmend beides: eine dokumentierte Baseline und den Beleg, dass die Produktion ihr entspricht.

Wie es in der Praxis aussieht

Auf dem eigenen Produktionshost dieses Unternehmens ist SSH der einzige interaktive Zugang, und seine Konfiguration wird von der Automatisierung geschrieben. Im September 2026 las ein Zugriffsbericht die aufgelösten Einstellungen des Daemons zurück und stellte fest, dass zwei beabsichtigte Direktiven überschrieben waren: Die Betriebssystemvorgaben wurden vor der Drop-in-Datei des Unternehmens geladen, und für diese Einstellungen gewinnt der erste Wert. Ein früher einsortiertes Dateinamenpräfix korrigierte die Reihenfolge. Die dauerhafte Änderung war eine dreistufige Validierung von 24 SSH-Direktiven bei jedem Abgleich: Jede Kandidatendatei muss sich parsen lassen, bevor sie die aktive Datei ersetzt. Nach der Installation muss die zusammengesetzte Konfiguration eine Prüfung bestehen. Anschließend werden die vom Daemon tatsächlich aufgelösten Einstellungen zurückgelesen und gegen alle 24 Vorgaben geprüft. Zwei weitere Direktiven blieben absichtlich außen vor, weil der Daemon sie nicht mehr implementiert.

Derselbe Host betreibt 13 durch die eigene Automatisierung installierte Unit-Dateien. Sie liefen mit den Standardrechten der Dienstverwaltung. Jede Unit-Vorlage erhielt eine Obergrenze für Privilegien, einen Systemaufruffilter, eingeschränkte Adressfamilien und Speicher, der entweder beschreibbar oder ausführbar sein darf, aber nie beides zugleich. Nach der eigenen Bewertung der Prozessaufsicht brachte die Sandbox-Arbeit einen Totmannschalter-Dienst von 9,6 auf 1,5, die öffentlich erreichbare Git-Forge von 8,3 auf 1,5 und die beiden Backup-Units von 9,6 auf 2,3 und 2,5. Beim Schreiben fiel auf, dass eine Direktive in drei Vorlagen falsch geschrieben war. Die Prozessaufsicht protokolliert einen unbekannten Schlüssel und startet die Unit trotzdem. Nun durchläuft jede Unit beim Abgleich das eigene Prüfwerkzeug der Prozessaufsicht.

Ein Härtungsreview im August 2026 fragte, ob die SSH-Schutzmaßnahmen gegen Brute Force wirkten, und las dazu das laufende System aus. Alle drei Schutzmaßnahmen waren seit dem Aufbau des Hosts wirkungslos geblieben: Die Sperr-Jail überwachte Port 22, während der Daemon auf 1986 lauschte. Eine Ratenbegrenzung der Firewall stand unter einer allgemeineren Regel, die zuerst griff. Die Sperraktion rief ein Paketfilterprogramm auf, das die Paketverwaltung nur empfiehlt und das nicht installiert war. Aus diesem Review entstanden zwei

Regeln, die das Repository weiterhin bestimmen: Eine Sicherheitskontrolle erhält eine Assertion statt eines Kommentars. Eine Firewall wird anhand der Position ihrer Regeln geprüft, nicht anhand ihres bloßen Vorhandenseins.

Wo es schiefgeht

Die Einstellung steht in der Datei, und der Dienst ignoriert sie. Viele Daemons lesen mehrere Dateien und behalten für jeden Schlüssel den ersten oder letzten gefundenen Wert. Eine Härtungsdirektive in einem Drop-in kann gegen einen früher geladenen Herstellerstandard verlieren. Fragen Sie das Programm, was es tatsächlich tun wird, bei SSH mit `sshd -T`, und vergleichen Sie diese Ausgabe bei jedem Durchlauf mit den vorgesehenen Werten.

Ein Tippfehler macht eine Kontrolle zum Kommentar. Dienstverwaltungen und viele Daemons protokollieren einen unbekannten Schlüssel und arbeiten weiter. Die Unit startet, das Dashboard ist grün, und die Einschränkung fehlt. Führen Sie beim Deployment das eigene Prüfwerkzeug des Parsers aus, damit eine unbekannte Direktive die Auslieferung stoppt.

```
# What the SSH daemon will do once every Include is resolved
sshd -T | grep -E '^(passwordauthentication|permitrootlogin) '
# How much a unit's sandbox still exposes, scored 0.0 to 10.0
systemd-analyze security --no-pager example.service
# Unknown directives and other unit-file errors
systemd-analyze verify /etc/systemd/system/example.service
```

Eine Schutzmaßnahme bewacht die falsche Tür. Ein Port wird verlegt, und die Sperrregel überwacht weiter den alten. Eine Ratenbegrenzung wird unter einer breiten Freigaberegel ergänzt, die bereits gegriffen hat. Die Kontrolle meldet sich als gesund, weil sie läuft. Das sagt nichts darüber aus, ob sie je auslöst. Prüfen Sie jede Kontrolle anhand ihrer Wirkung: Lösen Sie sie absichtlich von außen aus und bestätigen Sie, dass das Paket verworfen wird.

Die Sandbox legt nachts den Dienst lahm. Ein Systemaufruffilter oder ein schreibgeschützter Pfad blockiert den selten ausgeführten Codepfad, etwa den wöchentlichen Export oder den Backuphook. Der Fehler zeigt sich erst Stunden nach der Änderung, zur Laufzeit statt beim Parsen. Verschärfen Sie die Einschränkungen stufenweise, führen Sie nach jeder Stufe alle geplanten Jobs aus und achten Sie im Journal auf Prozesse, die durch den Filter beendet wurden.

Die Benchmark-Punktzahl wird zum Ziel. Ein Host kann jede Zeile einer allgemeinen Checkliste erfüllen, während seine einzige öffentlich erreichbare Anwendung als root läuft und Schreibzugriff auf ihren eigenen Code hat. Beginnen Sie damit, was aus dem Netzwerk erreichbar ist und worauf jeder erreichbare Prozess zugreifen könnte. Nutzen Sie den Benchmark anschließend, um zu finden, was diesem Review entgangen ist.

Weiterführende Lektüre

- Härtung im NIST-Glossar — die von NIST verwendete Definition mit der Veröffentlichung, aus der sie stammt.
- sshd_config(5) — jede SSH-Serverdirektive und die Regel, dass der erste für ein Schlüsselwort ermittelte Wert verwendet wird.
- systemd.exec(5) — Sandbox-Direktiven für Dienste: Systemaufruffilter, Adressfamilien und entweder beschreibbarer oder ausführbarer Speicher.
- systemd-analyze(1) — der Expositionswert des Unterbefehls `security` und der Unterbefehl `verify`, der unbekannte Direktiven erkennt.
- CIS Benchmarks — gemeinsam erarbeitete Konfigurationsbaselines für Betriebssysteme, Cloudplattformen und Serversoftware.



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 Härtung?“ 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/hardening/