Engineer

ハードニング

攻撃者が悪用できるものを取り除く作業

どのようなものか

ハードニングは、システムが仕事に必要なものだけを公開、許可するよう設定する作業です。動くサービス、待ち受けポート、各プロセスの権限を減らし、残すものの既定値を厳しくします。前後でソフトウェア自体は同じです。変わるのは、侵入者が届く範囲と、一つの足掛かりから進める距離です。制限する層に応じて、システム、サーバー、OS、SSHのハードニングとも呼びます。

Linuxサーバーなら、一度にいくつかの層を扱うのが普通です。遠隔ログインは鍵を認め、パスワードを拒否します。rootは直接ログインできません。使わないパッケージは、止めるだけでなく取り除きます。常駐サービスごとに、専用の非特権利用者、ファイルシステムを読み取り専用で見せる制約、許すシステムコールの一覧を与えます。ファイアウォールは、ホストがサービスを提供するための少数のポートだけを通します。珍しい処理ではありません。すべてを全ホストで行い、その状態を保つことに規律が要ります。

混同されがちな実践と並べると、位置付けが分かります。パッチは、修正が出た後、コードの特定の欠陥を直します。監視と侵入検知は、進行中の攻撃を見つけます。ファイアウォールは、誰がポートへ届くかを決めます。ハードニングは、そのポートへアクセスされた後、その先のプログラムに何を許すか、そもそもそのポートを開くべきかを決めます。CIS Benchmarksなどの公開基準は、OSや製品ごとの確認項目へまとめます。最低限の基準には有用ですが、到達点と思えば誤解します。

概念を判断に生かす

「ハードニングとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 アカウント、パスワード、権限を整理し、業務に必要なアクセスだけを付与し、退職者の権限は当日に解除します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 Linuxサーバーを守る1,000件のRBACルールを作成 — 1,000 セキュリティ上の穴を塞ぐアクセスルール

Linuxサーバーを守る1,000件のRBACルールを作成 お問い合わせ →

なぜ重要なのか

決めるのは、限られたセキュリティ予算をどこへ使うかです。ハードニングは、来年初めて見つかる脆弱性にも効くという、珍しい性質があります。パッチは既知の一つの脆弱性から守りますが、堅牢にしたサービスは、攻撃を受けたプロセスにできることを制限します。例えば、実行可能なメモリーへの書き込み、rawソケットの利用、ホームディレクトリーへのアクセスを禁じることは、遠隔コード実行のバグによる影響を抑えるための対策です。パッチは穴を閉じ、ハードニングは、どの穴からでも得られるものを減らします。本格的な運用には両方が必要で、公表から修正が届くまでの間に時間を稼ぐのが、ハードニングです。

主な費用は、ライセンスより、エンジニアリングの時間と、壊すリスクです。制限は、そのサービスに、取り上げる機能が決して要らないという賭けで、外れることもあります。一晩に一度のバックアップ、シグナルを送るログローテーション、シェルを呼ぶプラグインなどです。そのため、本番サーバーを手で制限する仕事は止まりがちです。午前三時に何かを壊して痛い目を見ると、それ以上厳しくしなくなります。コードとして、制限をレビュー、版管理し、再構築のたびに同じように適用すれば、同じ仕事を低い負担で繰り返せ、安全に戻せます。

発注者にとっての問いは、いつ行い、実施済みだとどう確認するかです。前者は、ホストを作る自動化の一部として、構築時に行うことです。稼働中のマシンへ一度だけ確認項目を適用しても、数か月でずれます。後者は、設定リポジトリーではなく、稼働システムからの証拠です。監査者は、文書化した基準と、本番が一致する証拠の両方を、ますます求めるようになっています。

実際の活用例

自社の本番ホストでは、対話的に入る方法はSSHだけで、設定は自動化によって書き込まれます。2026年9月のアクセス報告が、デーモンの最終的な設定値を読み出すと、意図した二つのディレクティブが上書きされていました。OSの既定値が自社の差し込み設定より先に読み込まれ、その項目では最初の値が優先されるためです。並べたときに先に来るファイル名の接頭辞を付け、読み込み順を直しました。残った仕組みは、構成を収束させるたびに、SSHの24ディレクティブを三段階で検証する処理です。本番ファイルと交換する前に候補を解析できるこ

と、配置後に組み合わせた設定全体が検査を通ること、最後にデーモンが実際に決めた値を読み、24項目すべてに照らして検証することです。ほかの二つは、デーモンがもう実装しないため、意図して外しました。

同じホストでは、自社の自動化が導入した13のユニットファイルが、サービス管理機構の既定の権限で動いていました。各テンプレートへ、権限の上限、システムコールのフィルター、アドレスファミリーの制限、メモリーへの書き込みと実行を同時に許さない設定を加えました。管理機構自身の採点で、サンドボックス化はデッドマンズスイッチのサービスを9.6から1.5へ、インターネットに面するGitフォージを8.3から1.5へ、二つのバックアップユニットを9.6から2.3と2.5へ変えました。作業中、三つのテンプレートに、一つのディレクティブの誤記が見つかりました。未知のキーをログへ残しても、管理機構はユニットを起動します。今は、収束処理中にすべてのユニットを、管理機構自身の検証器へ通します。

2026年8月のレビューでは、SSHへの総当たりを防ぐ仕組みが働くかを問い、稼働システムを読みました。三つとも、ホスト構築以来、機能していませんでした。デーモンは1986で待つのに、遮断用のjailは22番ポートを見ていました。ファイアウォールの頻度制限は、先に一致する広い規則より下にありました。遮断処理は、パッケージ管理で推奨されるだけで、入っていないパケットフィルターの実行ファイルを呼びました。そこから二つの規則が生まれ、今もリポジトリーに適用します。セキュリティ制御には、コメントでなく検証用のアサーションを付けること。ファイアウォールは、規則の有無でなく、順番で確かめることです。

問題が起きるところ

ファイルにある設定を、サービスが無視します。 多くのデーモンは複数ファイルを読み、キーごとに最初か最後の値を採ります。差し込み設定の制限が、先に読むベンダーの既定値に負ける場合があります。実際に何をするかを、SSHなら `sshd -T` でプログラム自身に尋ね、毎回、意図した値と比べます。

誤記が、制御をコメントに変えます。 サービス管理機構や多くのデーモンは、未知のキーをログへ残し、続行します。ユニットは動き、画面は正常でも、制限はありません。デプロイの一部でパーサー自身の検証器を動かし、未知のディレクティブがあれば配備を止めます。

```
# すべてのIncludeを読み込んだ後のSSHデーモンの実効設定
sshd -T | grep -E '^(passwordauthentication|permitrootlogin) '
# ユニットのサンドボックスの露出度（0.0から10.0の評価）
```

```
systemd-analyze security --no-pager example.service
# 未知のディレクティブとその他のユニットファイルのエラー
systemd-analyze verify /etc/systemd/system/example.service
```

守る入口が違います。 ポートを移しても、遮断規則は古いものを見続けます。頻度制限を、すでに一致する広い許可規則の下へ入れます。動いているので正常と報告しますが、発動する証拠ではありません。効果で試します。外から意図して発動させ、パケットが落ちることを確かめます。

サンドボックスが、夜中にサービスを壊します。 システムコールのフィルターや読み取り専用のパスが、週次エクスポートやバックアップフックなど、まれなコード経路を壊します。変更の数時間後、解析時ではなく実行時に分かります。段階的に厳しくし、各段階の後ですべての定期ジョブを動かし、フィルターで終了させられたプロセスがないか、ジャーナルを見ます。

基準の点数が、目的になります。 汎用の確認項目をすべて通っても、外から届く唯一のアプリがrootで動き、自分のコードへ書き込める場合があります。まずネットワークから届くものと、各プロセスが触れる範囲を調べ、そのレビューが見落としたものを、基準で補います。

参考資料

- NIST用語集のハードニング — NISTの定義と、その出典の出版物です。
- sshd_config(5) — SSHサーバーの全ディレクティブと、キーごとに最初に得た値を使う規則です。
- systemd.exec(5) — システムコールのフィルター、アドレスファミリー、書き込みか実行の一方だけのメモリーなど、サービスのサンドボックス設定です。
- systemd-analyze(1) — `security` が示す攻撃への露出度の点数と、未知のディレクティブを検出する `verify` です。
- CIS Benchmarks — OS、クラウド基盤、サーバーソフトウェアの、合意に基づく設定基準です。

用語集

実際の制約に合わせて選ぶ

デプロイの流れ、運用上のリスク、緊急対応の担当者をお知らせください。検査、復旧手段、明確な責任の所在を伴う変更の範囲を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「ハードニングとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

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

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/hardening/