Engineer

権限

🎟 システムが人やプロセスに与える操作上の権利

どのようなものか

権限とは、本来システムが禁じる操作を、ある人や実行中のプログラムに許すものです。権限を与える立場の主体が、持たせるべきだと判断して付与します。他人のファイルを読む、1024未満のポートにバインドする、アカウントを作る、データベース関数を実行する、といった操作です。英語では、一つしか持たない主体はまれなのでprivilegesと複数形で呼び、アカウントやプロセスが境界のどちら側にいるかを、privileged、unprivileged、つまり特権・非特権という言葉で表します。NISTの用語集は、忘れがちな点を加えています。権限は、主体に認められた振る舞いを表し、権限を定める立場の主体が定義して、方針や規則に記すものです。

近い意味の言葉が二つあります。アクセス制御は、個々の要求を認めるかどうか判断することです。権限は、その判断が要求元について参照する事実の一つです。パーミッションは通常、同じ考えを資源の側から見たものです。ファイルには誰が読めるかが定められ、読み手は、そのファイルを読む権利を持ちます。

Linuxでは、正確な意味が定まっています。capabilities(7)マニュアルは、従来のUnixの区分を説明しています。実効ユーザーID 0で動くrootと呼ばれる特権プロセスは、カーネルのすべての権限検査を回避します。それ以外のプロセスは、自身の資格情報に基づいて検査されます。Linux 2.2以降、カーネルはrootの力をケーパビリティーという個別の単位へ分け、一つずつ有効・無効にできるようにしました。そのため、低い番号のポートへのバインドだけが必要なプログラムには、`CAP_NET_BIND_SERVICE`だけを与えられます。credentials(7)は、どの識別情報を検査するかを説明します。実ユーザーIDはプロセスの所有者を表し、実効ユーザーIDはカーネルが権限と照合する値です。保存されたset-user-IDは、昇格した識別情報をいったん使わずに置き、後から取り戻すためのものです。

データベースと組織も、それぞれが与える権利に同じ言葉を使います。データベースロールは、スキーマ、表、関数に対する権限を持ちます。管理者アカウントには、職員用アカウントにない権限があります。どの層でも、考えは同じです。通常は閉じている境界を越えるために、意図して与えた権利です。

概念を判断に生かす

「権限とは？」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

プログラムが持つ権限によって、問題が起きたときに及ぼせる被害の大きさが決まります。ケーパビリティーの目的も、まさにそこにあります。侵害されたプログラムが持つ権利を狭くすれば、同じプログラムをrootで動かす場合より被害を抑えられます。意図しないコードの実行を許す不具合とroot権限を持つウェブサーバーは、不具合を見つけた相手にマシン全体を渡してしまいます。同じ不具合でも、自身のディレクトリーを読み、一つのポートへバインドするだけのプロセスなら、届く範囲は限られます。

NISTは、これを最小権限として規定しています。SP 800-53の管理策AC-6は、利用者と、その代理で動くプロセスに対して、割り当てられた仕事に必要なアクセスだけを認めるよう、組織に求めます。強化策は、それを日常の作業へ落とし込みます。管理者はメールなどの日常業務に非特権アカウントを使い、特権アカウントは必要な人に限定し、特権機能の利用を記録し、権限を見直し、必要性を示せなくなったものを取り除きます。

既定の状態がそうなっていることは、あまりありません。systemdでは、`User=`行のないシステムサービスはrootで動きます。`NoNewPrivileges=`は、サービスと子プロセスが`execve()`を通じて権限を増やすことを禁じる設定ですが、誰かが有効にしなければ無効です。データベースのツールは、スキーマを作った際の設定のまま、すべての表の所有者として接続しがちです。新しい職員のアカウントも、既存の同僚のものから、権限ごとコピーしがちです。どれも初日は便利ですが、侵害された日には高く付きます。

権限を狭めるには時間がかかります。実際にプログラムが何をするかを調べる必要があり、狭すぎる権限は、インストール時ではなく実行時に不具合を起こします。チームが決めるのは、多くの場合、その調査時間を何に先に使うかです。インターネットに面するもの、失えば困るデータを持つもの、ほかのアカウントを変更できるアカウントが対象になります。購入を検討する人や監査者には、率直に聞くべき問いがあります。ここでは誰に、何に管理者権限があり、なぜ必要なのでしょうか。

実際の活用例

自社ホストでは、十三個のユニットファイルが、監督役の既定設定で与えられる、大半の権限を持って動いていました。すべてのユニットテンプレートに、各ユニット自身の権限を上限とする制限、システムコールフィルター、アドレスファミリーの制限、書き込みと実行を同時には許さないメモリー設定を加えました。デッドマンズスイッチの強化スコアは9.6から1.5へ、Gitフォージは8.3から1.5へ、二つのバックアップユニットは9.6から2.3と2.5へ変わりました。三つのテンプレートで、一つのディレクティブの綴りが間違っていたことも分かりました。監督役は、未知のキーをログに記しながらユニットを起動していたため、今は収束処理のたびに、すべてのユニットを監督役自身の検証器へ通します。ページは、スコアが示さない限界も記します。ユニットを壊すディレクティブの影響が表れるのは、実行時です。

あるアプリケーション基盤では、APIが最小権限のロールでデータベースにログインします。与えられるのは、パラメーター化クエリーを使ってSECURITY DEFINERとして動くアプリケーション関数に対する、EXECUTE権限だけです。プラン階層、ロール、組織や船ごとの範囲などの利用権限は、リクエストのたびにサーバーで再確認し、画面内の制限は利用体験のためだけのものとして扱います。

ある組織では、M365、1Password、Red Hat SSO、OKTA SSOにまたがる識別情報とアクセスを統合し、最小権限を徹底しました。着任時と離任時の手続きを標準化し、離れた後もアクセスが残ることなく、速やかに取り消すようにしました。

問題が起きるところ

ユニットに指定がないため、サービスがrootで動きます。 システムサービスに対する監督役の既定値はrootです。そのため、 `User=` 行がないユニットは、マシンのあらゆる権限を持ちます。サービスごとのユーザーを与えるか、

`DynamicUser=` でsystemdに割り当てさせます。これはユニットの開始時にユーザーを作り、停止時に解放します。

一つのケーパビリティーが、rootの力の大半を含みます。 `CAP_SYS_ADMIN` は非常に多くの操作を含み、そのマニュアルでも過剰に多くの役割を担うものとされ、「新しいroot」と呼ばれています。一つの機能を有効にするために与えれば、rootの力の大半を与えたことになります。必要なケーパビリティーを具体的に調べ、ケーパビリティーの上限集合を、その一覧に絞ります。

```
[Service]
User=web
NoNewPrivileges=yes
CapabilityBoundingSet=CAP_NET_BIND_SERVICE
AmbientCapabilities=CAP_NET_BIND_SERVICE
```

この設定なら、通常のユーザーで動くサービスがポート80へバインドでき、それ以外のケーパビリティーはすべて上限集合から取り除かれます。

プロセスが、開始時より強い権限を得ます。 通常のユーザーで動くプログラムでも、set-user-ID付きバイナリーや、ケーパビリティーが付いたファイルを実行でき、カーネルはその実行の権限を引き上げます。防御する側は、実効ユーザーIDが実ユーザーIDと違うプロセスとして把握します。 `NoNewPrivileges=yes` は、サービスと、そのサービスが起動するすべての子プロセスに対し、この経路を閉じます。

仕事が変わるたびに、権利が積み重なります。 チームを移った人が、新しいアクセスに加えて古いアクセスも持ち続けると、数年後には、そのアカウントに、経験したすべてのロールの権利が集まります。AC-6は定期的な見直しで、AC-2は異動・離任に連動したアカウント管理で対応し、仕事の変化をアクセスの変化にも結び付けます。

アプリケーション自身の接続用アカウントが、何でもできます。 APIがスキーマの所有者として接続すると、一つのインジェクションの欠陥から、どの表にも届いてしまいます。使う関数を呼ぶことしかできないログインロールなら、同じ欠陥による問題を、はるかに小さくできます。

参考資料

- NIST CSRC用語集の権限 — CNSSI 4009の定義で、権限を与える立場の主体が定義することも記されています。
- capabilities(7) — Linuxがrootを個々のケーパビリティーへ分け、スレッドがそれらを保持する仕組みです。
- credentials(7) — 実、実効、保存、ファイルシステムのユーザーIDと、それぞれが何を判断するかを扱います。
- systemd.exec(5) — `User=` 、 `DynamicUser=` 、 `NoNewPrivileges=` と、ケーパビリティーの設定です。
- NIST SP 800-53 Rev. 5 — 最小権限の管理策AC-6と、それを日常の実践にする強化策です。

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

「権限とは？」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/privilege/