Engineer

RBAC

個人に直接ではなく、ロールへ権限を付与するアクセス制御

どのようなものか

RBACは、誰であるかではなく、持っているロールを基に、その人が何をしてよいかを決めます。略称は、ロールベースのアクセス制御を意味し、モデルを形式化したNISTは、「ロールに基づくセキュリティー」とも呼びます。ロールは、権限をまとめて名前を付けたもので、「経理事務担当」や「本番環境の読み取り専用」などがあります。利用者にロールを割り当て、利用者が行使するすべての権限は、そのどれかを通して得ます。

NISTのDavid FerraioloとRick Kuhnは、当時の強制アクセス制御や任意アクセス制御に代わるものとして、1992年に一般的なモデルを提案しました。1996年には、Sandhu、Coyne、Feinstein、Youmanが、層に分けました。利用者、ロール、権限を持つRBAC0、ロール階層を加えたRBAC1、制約を加えたRBAC2、両方を持つRBAC3です。二つの流れは統合され、ANSI/INCITS 359-2004という標準になり、2012年に改訂されました。参照モデルには、四つの構成要素があります。すべてのシステムに必須なのは、コアRBACだけで、利用者とロールの対応付け、権限とロールの対応付けに加え、利用者が割り当てられたロールの一部を有効にするセッションを持ちます。階層型RBACでは、上位のロールが下位のロールを継承できます。静的な職務分離は、一人に同時に割り当てられるロールを制約し、動的な職務分離は、一つのセッションで同時に有効にできるロールを制約します。そのため、支払いを起票するロールと承認するロールを一人で持っていても、一つのセッションでは併用できないように制約できます。

通常のグループとの差は、見た目ほど大きくありません。NISTのFAQは、次のように説明します。グループは、普通は利用者の集合で、ほとんどのグループの仕組みは、利用者へ直接権限を付けることも許します。FAQは、これを「抜け穴」と呼び、利用者と権限の関係を制御しにくくするとしています。RBACは、権限をロールだけに付けることで、この穴を閉じます。理由は、同じFAQの別の一文にあります。組織のロールは比較的安定していますが、利用者と権限は数が多く、速く変わるため、すべてのアクセスをロール経由にすると、管理とレビューが容易になります。

データベースは、モデルを直接実装していることがよくあります。PostgreSQLには、バージョン8.1以来、ロールという一種類の主体があり、そのロールが、利用者、グループ、または両方として機能します。グループとして使うロールは、通常LOGINを持たず、ほかの権限と同じように、所属を付与します。メンバーがグループの権限をどう使うかは、二つのメンバーシップオプションで制御します。INHERITは、グループの権限を自動的に与え、SETは、SET ROLEで、そのロールへ切り替えられるようにします。必要なセッションでだけロールを有効にする、RBACの考え方に近い動作です。

```
-- ロールが権限を持ち、このロールではログインしません
CREATE ROLE reporting NOLOGIN;
GRANT USAGE ON SCHEMA sales TO reporting;
GRANT SELECT ON ALL TABLES IN SCHEMA sales TO reporting;

-- 利用者にはテーブルの権限を直接与えず、ロールを付与します
CREATE ROLE alex LOGIN;
GRANT reporting TO alex;

-- 意図して有効にする必要がある強力なロール
CREATE ROLE schema_admin NOLOGIN;
GRANT schema_admin TO alex WITH INHERIT FALSE;
-- 後で必要なセッション内で実行：SET ROLE schema_admin;
```

概念を判断に生かす

「RBAC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

アクセス制御は、異なる方向を向いた二つの問いに答える必要があります。このリクエストを今許可するか、という問いと、システム全体で誰が何をできるか、という問いです。多くの方式は、最初の問いにはうまく答えます。RBACが目指すのは、二つ目です。答えが、ロールと所属する人の短い一覧になり、監査担当者や管理者が、実際に読めるためです。別の方式である属性ベースのアクセス制御は、利用者、リソース、状況の属性に対する規則を、判断のたびに評価します。NISTは、これを「導入は容易な場合もあるが、管理は複雑になり得る」と、一文で評しています。

実際に判断すべきなのは、ロールだけでは足りなくなる境界です。1992年のモデルは、アクセスを許すのは、ロールの条件を満たす「場合に限る」としています。アクセス時の追加検査の余地を残すために、選んだ表現です。医師のロールを持つ人は、診療記録を読めますが、すべての患者の記録ではありません。この医師とその患者の関係は、役職名ではなく、データに関する事実です。そこで、データベースの行レベルセキュリティーや、アプリケーションの属性の規則など、二つ目の仕組みを置きます。ロールを置き換えるのではなく、その上に重ねます。

当社は、まず、実際の仕事に基づく名前のロールでアクセスをモデル化し、権限はロールだけに付け、ログインアカウントへ直接付けないことを勧めます。ロールの一覧は、一人がすべて読める長さに保ちます。ロールでは境界を引けないと実際に分かった箇所にだけ、行単位や属性の規則を加え、データベースで強制できるものは、データベースに任せます。データベースの検査は、誰かが忘れたコードの経路を含め、すべてのクライアントに適用されるためです。

実際の活用例

2022年から2024年のデータベースエンジニアの業務では、すべての開発者環境、アプリケーションのインスタンス、PostgreSQLとMS SQLのデータベース、Linuxサーバーへのアクセスを、1,000件のRBACルールで管理しました。モデルは単純でしたが、量が課題でした。権限を「本番データベースへの読み取り専用アクセス」など、再利用できる区分に分け、開発者、アプリケーションのインスタンス、データベース管理者などのロールへ割り当てました。開発者は、指定されたサーバーにだけ到達でき、アプリケーションのインスタンスには、横方向への侵入拡大を防ぐため、限定的な権限を与えました。Ansibleで、環境間で一貫してルールを適用し、ルールは、チームが監査や変更をできるよう、文書化しました。提供した成果は、ロールベースのルール一式と、再現可能な適用・レビューの方法です。

ロールだけでアクセスを制御することは、あまりありません。地図アプリケーションをMSSQLから、PostGISを加えたPostgreSQLへ移したときは、複数の利用者グループにデータ保護方針を適用するため、ロールベースのアクセス制御と、行レベルセキュリティーを一緒に導入しました。この組み合わせが、一般的な分担です。ロールは、グループが触ってよいテーブルを決め、行のポリシーは、その中のどの行を扱ってよいかを決めます。

最も厳密な形では、アプリケーションからテーブルへ直接アクセスできなくします。2025年からの担当業務における関数を中心にしたPostgreSQLのデータ層では、アプリケーションの全操作が、1,275個の関数のどれかで、その関数を34個

のスキーマに分けています。APIがログインするロールmarinerは、アプリケーションの関数へのEXECUTEと、スキーマへのUSAGEだけを持ち、それ以外は持ちません。一括の権限付与の代わりに、約4,000件を明示的に付与します。関数はSECURITY DEFINERとして動き、search_pathを固定した、別のログイン不能なロールが所有し、スーパーユーザーのアカウントは、マイグレーションとcron専用に保ちます。APIが侵害された場合でも、関数が許す操作に制限されます。代償も明記されています。新しいフィールドには、マイグレーションと関数の変更が必要です。

問題が起きるところ

- **ロールが増え、誰も読み切れなくなります。** 例外ごとに、「請求処理以外の開発者」や「一つのレポートを追加で読める開発者」などのロールを作り、やがて人より多くなります。ロールの爆発的な増加は、レビューできる短い一覧という、RBACの目的を損ないます。例外ごとに作るより、一人に二つか三つを与え、権限の集合が同じになったロールを統合します。
- **ログインアカウントへ直接、権限を付けます。** 緊急のチケットで、一度利用者へ直接GRANTすると、ロールのレビューでは見えないアクセスが、その利用者に残ります。NISTがグループの仕組みについて説明する抜け穴を、手作業で作り直した形です。ロールにだけ付与し、レビュー用のクエリーで、ログインできるロールが持つ権限を探します。
- **ロールで付与する前に、既定の設定がアクセスを与えます。** PostgreSQLのPUBLICは、すべてのロールを意味し、既定では、新しい関数へのEXECUTEと、データベースへのCONNECT、TEMPORARYを受け取ります。PostgreSQL 14以前から更新したデータベースでは、PUBLICが、publicというスキーマにオブジェクトを作れる場合もあります。これらの既定の権限は、意図的に取り消し、できれば、オブジェクトを作るトランザクション内で行います。
- **割り当てるロールを、増やすだけになります。** 人が仕事を変えても、以前のロールを新しいものと一緒に保ち、一時的なアクセスも、削除が誰の担当でもないため、恒久化します。割り当てには、それぞれ理由と見直し日が必要で、その仕事を離れたら、付随するロールも取り除くべきです。
- **規則を迂回するアカウントがあります。** スーパーユーザーとBYPASSRLSを持つロールは、行レベルセキュリティーを無視し、通常、テーブルの所有者も、FORCE ROW LEVEL SECURITYを設定しない限り同様です。所有者やスーパーユーザーで接続するアプリケーションは、すべての行のポリシーを飾りに変えてしまいます。何も所有しないロールで接続します。

参考資料

- NISTロールベースアクセス制御プロジェクト — アーカイブされたプロジェクトのページで、モデルの歴史、2004年の標準、2012年の改訂を説明します。

- NIST RBAC FAQ — ロールとグループの違い、RBAC0からRBAC3、職務分離、標準の四つの構成要素です。
- PostgreSQL：ロールの所属 — グループロール、所属の付与と取り消し、INHERITとSETのオプションです。
- PostgreSQL：行セキュリティーポリシー — 既定での拒否、行のポリシーを迂回するロール、FORCE ROW LEVEL SECURITYです。
- PostgreSQL：権限 — PUBLICが既定で受け取るものと、取り消し方です。
- SQL Server：データベースレベルのロール — SQL Serverデータベース内の、固定ロールと利用者定義のロールです。

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

「RBAC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/rbac/