FUJITSU

FUJITSU
Enterprise Postgres 14 SP1

セキュリティ運用ガイド

Linux

J2UL-2749-02ZJZ0(00)
2022年9月

まえがき

本書の目的

本書は、“FUJITSU Software Enterprise Postgres”(以降、“FUJITSU Enterprise Postgres”と略します)を使ってシステムを構築/運用する際のセキュリティについて説明しています。

本書の読者

本書は、以下のような読者を対象に書かれています。

- FUJITSU Enterprise Postgresの導入を検討している方
- FUJITSU Enterprise Postgresでデータベースセキュリティ運用環境の設計、構築、運用を行う方
- FUJITSU Enterprise Postgresで構築されたデータベースシステムへアクセスする方

本書を読むためには、以下の知識が必要です。

- 業務に関する知識
- FUJITSU Enterprise Postgresに関する知識
- Linuxに関する一般的な知識

本書の構成

本書の構成と内容は以下のとおりです。

第1章 セキュリティの概要

セキュリティシステムの概要やFUJITSU Enterprise Postgresが提供するセキュリティ機能について説明します。

第2章 セキュリティ運用の概要

セキュリティ運用の概要について説明します。

第3章 責任者の作業

責任者が実施するセキュリティ対策の作業について説明します。

第4章 管理者の作業

管理者が実施するセキュリティ対策の作業について説明します。

第5章 利用者の作業

利用者が実施するセキュリティ対策の作業について説明します。

第6章 監査ログ機能

FUJITSU Enterprise Postgresが提供する監査ログ機能について説明します。

本書の出典

本ドキュメントは、以下のドキュメントの一部を引用しています。

- “データベースセキュリティガイドライン(2.0版)”
  (データベース・セキュリティ・コンソーシアム(DBSC))

輸出管理規制について

本ドキュメントを輸出または第三者へ提供する場合は、お客様が居住する国および米国輸出管理関連法規等の規制をご確認のうえ、必要な手続きをおとりください。

出版年月および版数

2022年 9月　第2版
2022年 2月　初版

著作権

Copyright 2021-2022 FUJITSU LIMITED

目　次

第1章 セキュリティの概要 1
1.1 セキュリティとは 1
1.2 セキュリティの要件 1
1.3 セキュリティの脅威 2
1.4 セキュリティの対象範囲 4
1.5 FUJITSU Enterprise Postgresが提供するセキュリティ 5
1.5.1 セキュリティの対象者 5
1.5.2 セキュリティ機能 6

第2章 セキュリティ運用の概要 8
2.1 セキュリティ運用の流れ 8

第3章 責任者の作業 9
3.1 重要情報の定義/リスク分析 9
3.2 アカウント管理ポリシーの策定 9
3.3 ログの取得ポリシーの策定 9
3.4 規約の策定 10
3.5 教育の実施 11
3.6 データベース管理業務のチェック 11
3.7 セキュリティ対策状況の定期診断 11

第4章 管理者の作業 12
4.1 教育の受講 12
4.2 初期設定 12
4.3 認証 13
4.3.1 アカウントの管理 13
4.3.2 パスワードの管理 14
4.3.3 接続と認証の設定 15
4.4 アクセスコントロール 15
4.5 暗号化 15
4.6 外部媒体の利用制御 16
4.7 サーバ/アプリケーションに対するセキュリティ対策 16
4.8 ログの管理 17
4.8.1 ログの取得 17
4.8.2 ログの保全 17
4.9 不正アクセス検知 17
4.10 ログの分析 18

第5章 利用者の作業 19
5.1 教育の受講 19
5.2 アカウント/パスワードの管理 19

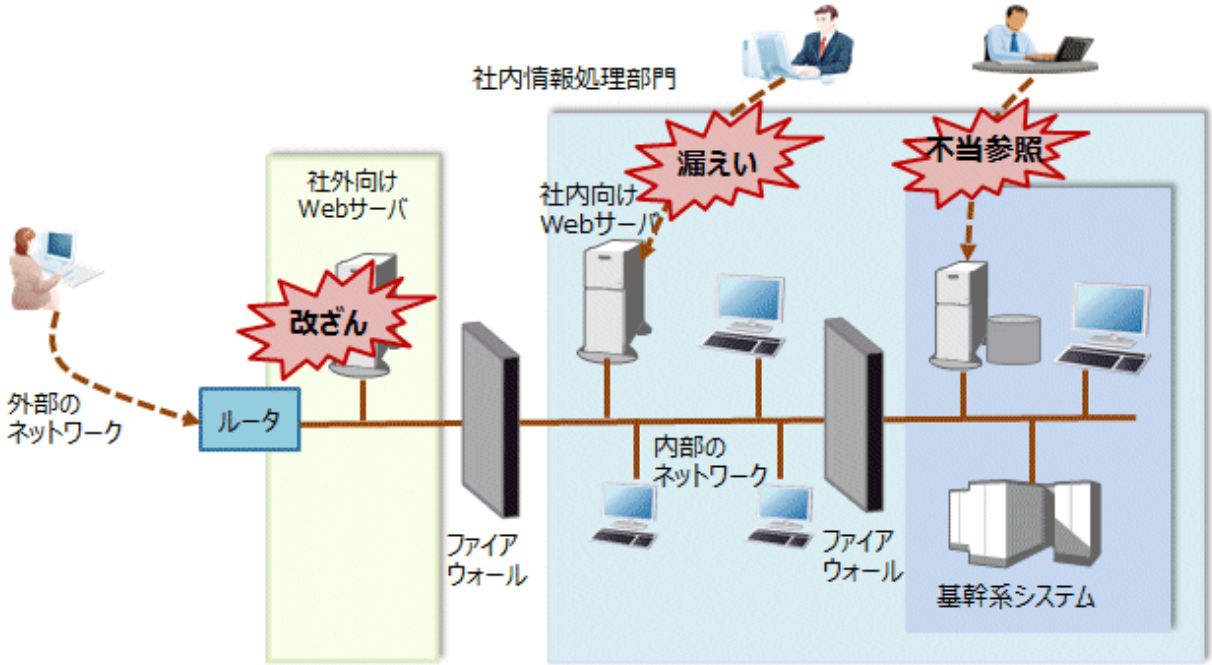
第6章 監査ログ機能 20
6.1 監査ログの出力モード 20
6.2 セットアップ 20
6.3 pgaudit設定ファイル 23
6.4 Session Audit Logging 25
6.5 Object Audit Logging 30
6.6 データベース多重化運用の場合 32
6.6.1 セットアップ 32
6.6.2 監査ログ取得の設定 33
6.7 SQLでの監査ログの参照 33
6.8 アンセットアップ 34

第1章 セキュリティの概要

1.1 セキュリティとは

セキュリティとは、情報の漏えいや改ざん、外部からの攻撃、進入、盗聴といった危険を排除すること、あるいは情報サービスへの不当な妨害を阻止することです。情報システムが社会基盤としての信頼を得るためには、漏えいや改ざんなどによるセキュリティの脅威を事前に防止するためのセキュリティ対策が必須です。

図1.1 セキュリティの脅威



情報システムにおけるセキュリティ対策は、以下に分類して考えることができます。

- ネットワーク
- Web
- アプリケーション
- データベース
- 端末

本書では、FUJITSU Enterprise Postgres利用時のデータベースセキュリティ対策を中心に説明します。

1.2 セキュリティの要件

情報システムに必要なセキュリティ要件を以下に示します。

セキュリティポリシーの維持

セキュリティポリシーとは、企業が情報資産に対してどのように取り組み、従業員がどのように行動すべきかといった方針を明文化した規範のことです。

セキュリティポリシーを維持しながら情報システムのセキュリティを遂行していく必要があります。

統合的なセキュリティ管理

セキュリティには、以下の側面があります。これらの側面から情報を統合的に管理する必要があります。

機密性

情報のアクセスを限定し、外部への情報漏えいを防止すること

対策例:情報漏えい防止、アクセス権の設定

完全性

情報が破壊されたり改ざんされたりせずに、完全であることを保証すること

対策例:改ざん防止や検出

可用性

障害を防ぎ、正常な運用を維持し、利用したいときに情報が利用できること

対策例:電源対策、システムの二重化

1.3 セキュリティの脅威

セキュリティの脅威とは、情報資産に対し、"1.2 セキュリティの要件"で示した「機密性」、「完全性」、「可用性」を脅かすものと定義します。データベースへのアクセスなど技術的な脅威を対象とし、物理的な破壊は含みません。

脅威は、脅威の発生元である登場人物、守るべき情報資産、手口、不正アクションの組合せで考えます。たとえば、「一般利用者が、データベース管理情報を、データベースの脆弱性を悪用して情報を入手し、情報を改ざんする」などが脅威となります。

セキュリティ対策を考える上で、まず、どのような脅威があるかを明確にする必要があります。ここでは、想定される脅威の一覧を以下に示します。登場人物、守るべき情報資産の定義については、それぞれ、"登場人物について"および"情報資産について"を参照してください。

想定される脅威

登場人物	情報資産	手口	不正アクション
一般利用者 社内利用者 システム責任者 システム開発者 システム管理者 システム運用者	データベース管理情報	パケットの盗聴	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
		パスワードの辞書攻撃	
		ソーシャルエンジニアリングによるID/パスワードの不正入手	
		設定を悪用した情報の不正入手	
		データベースの脆弱性を悪用した情報の不正入手	
		不正ルートで入手	
一般利用者 社内利用者	データベース一般情報	通常ルートで入手	入手できる情報の悪用(持ち出し)
		業務妨害を狙ったSQL発行	業務妨害(リソース枯渇)
一般利用者 社内利用者	データベース一般情報	パケットの盗聴	情報の不正改ざん、破壊(更新)
		パスワードの辞書攻撃	
		ソーシャルエンジニアリングによるID/パスワードの不正入手	
		設定ミスを悪用した情報の不正入手	
		データベースの脆弱性を悪用した情報の不正入手	
		不正ルートで入手	
システム責任者 システム開発者 システム管理者	データベース一般情報	パケットの盗聴	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
		パスワードの辞書攻撃	

登場人物	情報資産	手口	不正アクション
システム運用者		ソーシャルエンジニアリングによるID/パスワードの不正入手	
		設定ミスを悪用した情報の不正入手	
		データベースの脆弱性を悪用した情報の不正入手	
		不正ルートで入手	
システム開発者	データベース管理情報	バックドアの作成	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
	データベース一般情報		
システム責任者 システム管理者	データベース管理情報	不正なデータベース管理者アカウントの作成による情報の不正入手	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
	データベース一般情報		
システム責任者 システム運用者	データベース管理情報	データベース関連ファイル(定義ファイル、物理ファイルなど)改ざんによる情報の不正入手	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
	データベース一般情報		
データベース管理者	データベース管理情報	通常ルートで入手後、情報の悪用（持ち出し)	入手できる情報の悪用(持ち出し)
		管理情報からID/パスワードの不正利用	入手できる情報の改ざん、破壊
		管理情報の改ざんによる情報の不正入手	
		業務妨害を狙ったSQL発行	業務妨害(リソース枯渇)
	データベース一般情報	パケットの盗聴	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
		不正ルートで入手後、情報の悪用(持ち出し)	
データベース運用者	データベース管理情報	パケットの盗聴	情報の不正入手(参照) 情報の不正改ざん、破壊(更新)
		パスワードの辞書攻撃	
		ソーシャルエンジニアリングによるID/パスワードの不正入手	
		設定ミスを悪用した情報の不正入手	
		データベースの脆弱性を悪用した情報の不正入手	
		不正ルートで入手	
	データベース一般情報	通常ルートで入手	入手できる情報の悪用(持ち出し)
		業務妨害を狙ったSQL発行	業務妨害(リソース枯渇)

登場人物について

データベースセキュリティでは、データベースに関わる人物とその役割を以下のように定義します。

登場人物	役割
システム責任者	開発者、管理者、運用者の管理
システム開発者	データベース周辺のネットワーク構築 データベースサーバ構築
システム管理者	データベース周辺のネットワーク機器の運用 データベースサーバの運用
システム運用者	データベース周辺のネットワーク運用
データベース管理者	データベースシステムの構築 データベースシステムの運用
データベース運用者	業務の運用
社内利用者	社内のエンドユーザー
一般利用者	社外のエンドユーザー

情報資産について

データベースセキュリティでは、データベースサーバに格納する情報資産を保護する必要があります。

保護すべき情報資産を以下のように定義します。

データベース管理情報

- データベース構成情報(システムカタログ、ユーザーID/パスワードなど)
- データベースログ(アクセスログなど)

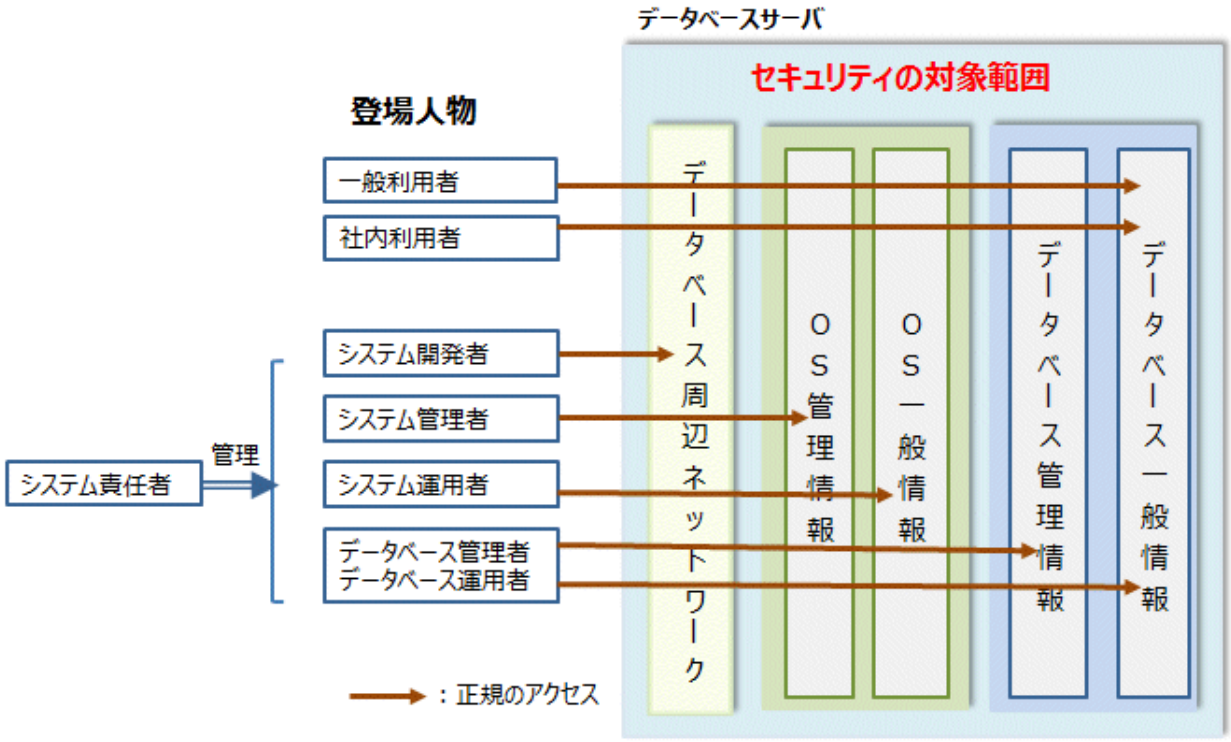
データベース一般情報

- 業務データ
- アプリケーション

1.4 セキュリティの対象範囲

データベースシステムでは、データベースサーバおよびデータベース周辺ネットワークをセキュリティの対象範囲とします。登場人物とそれぞれの登場人物がどの範囲で関わるかを明確にして、登場人物ごとにセキュリティ対策を考える必要があります。

以下に登場人物とセキュリティの対象範囲の関係を示します。



1.5 FUJITSU Enterprise Postgresが提供するセキュリティ

FUJITSU Enterprise Postgresでは、“1.2 セキュリティの要件”で示したセキュリティ要件を満たすセキュリティ機能が提供されています。

FUJITSU Enterprise Postgresが提供するセキュリティについて説明します。

1.5.1 セキュリティの対象者

FUJITSU Enterprise Postgresで構築するシステムでは、セキュリティに関する対象者を“責任者”、“管理者”、“利用者”とします。堅牢なセキュリティシステムを構築するために、それぞれの立場でセキュリティ対策を講じる必要があります。

セキュリティの対象者と“1.3 セキュリティの脅威”で示した登場人物についてのマッピングを下表に示します。

セキュリティの対象者	登場人物
責任者	システム責任者
管理者	システム開発者
	システム管理者
	システム運用者
	データベース管理者
	データベース運用者
利用者	一般利用者
	社内利用者

責任者

セキュリティポリシーを定め、組織全体の運用方針を決定します。

責任者の作業については、“第3章 責任者の作業”を参照してください。

管理者

システムを設計、構築および運用します。その際、管理者は、責任者が定めたセキュリティポリシーに従い、セキュリティ対策を実施する必要があります。

管理者の作業については、“第4章 管理者の作業”を参照してください。

利用者

データベースをアクセスする不特定多数の人であり、管理者以外の人です。データベースシステムに登録されている必要があり、データベースをアクセスする権限によりアクセスを制限されます。

利用者の作業については、“第5章 利用者の作業”を参照してください。

1.5.2 セキュリティ機能

FUJITSU Enterprise Postgresでは、以下のセキュリティ機能が提供されています。

- 認証
- アクセスコントロール
- 暗号化
- 監査ログ
- データ秘匿化

各機能について説明します。

認証

データベースにアクセスするデータベースユーザーの認証を行うことで、アクセス可能なデータベースを制限することができます。また、サーバを認証して、データベースサーバのなりすましを防止することができます。

認証の詳細については、“PostgreSQL Documentation”の“Server Administration”の“Client Authentication”を参照してください。

サーバの認証については、“PostgreSQL Documentation”の“Server Setup and Operation”の“Secure TCP/IP Connections with SSL”を参照してください。

アクセスコントロール

データベースのオブジェクトは、初期状態ではオブジェクトの作成者、またはオブジェクト作成時に所有者として指定されたデータベースユーザー(以降、両者を所有者と呼びます)、またはインスタンス管理者のみが使用できます。オブジェクトの所有者、またはインスタンス管理者は、データベースユーザーに対するアクセス権限を制御することで、データベースに接続したデータベースユーザーがどのような表にアクセスできるか、どのような操作を行うことができるかを制御することができます。

オブジェクトのアクセス制御の詳細については、“PostgreSQL Documentation”の“The SQL Language”の“Privileges”を参照してください。

暗号化

FUJITSU Enterprise Postgresでは、以下の要求事項を満たす透過的データ暗号化機能を提供します。

- 機密情報を判別不可能な状態にできること
- 暗号化キーとデータを分離して管理すること
- 定期的に暗号化キーの交換を行うこと

PostgreSQLでは、pgcryptoという暗号化機能が提供されており、FUJITSU Enterprise Postgresでも使用することができますが、アプリケーションに暗号化を意識した修正が必要となるため透過的データ暗号化機能の使用を推奨します。透過的データ暗号化については、“FUJITSU Enterprise Postgres 運用ガイド”の“透過的データ暗号化による格納データの保護”を参照してください。

また、クライアントとサーバ間の通信データに機密情報を含む場合、通信データを暗号化し、ネットワーク上の盗聴による脅威から通信データを保護する必要があります。

通信データの暗号化については、“FUJITSU Enterprise Postgres 運用ガイド”の“Secure Sockets Layerによる安全な通信の構成”を参照してください。

監査ログ

管理者の権限乱用、利用者のデータベースへの不正アクセスなどの脅威に対抗するための機能です。管理者や利用者の処理を追跡するための情報を監査ログとして取得、保持します。

管理者は、監査ログを定期的に参照・監視することによって、システムが何らかの影響を受けている、あるいは利用者が誤ったオペレーションを行っているためにシステムの資源が枯渇しかかっている、といった事象を検知し、適切な対応を行うことによって、情報漏えいやシステムダウンなどを未然に防ぐことができます。

監査ログについては、“第6章 監査ログ機能”を参照してください。

データ秘匿化

アプリケーションによって発行された問合せに対して、一部のデータを改訂して参照させることができます。

たとえば、従業員データの問合せに対して、実際のデータを表示せずに、8桁の従業員番号の最後の4桁以外を“*”で改訂して参照させる場合などに利用できます。

具体的には、データ秘匿化機能で改訂したデータをテスト用データベースに移行し、テストや開発を行うユーザーに参照させる利用方法があります。テスト時には本番環境データベースで利用するデータを用いてテストを行うことが求められます。しかし、機密データが流出するリスクが高まるため、本番環境データベースのデータをそのまま使用するべきではありません。本機能を利用することで、本番環境データベースにより近いデータをテストや開発環境でも安全に利用することができます。

データ秘匿化については、“FUJITSU Enterprise Postgres 運用ガイド”の“データ秘匿化”を参照してください。

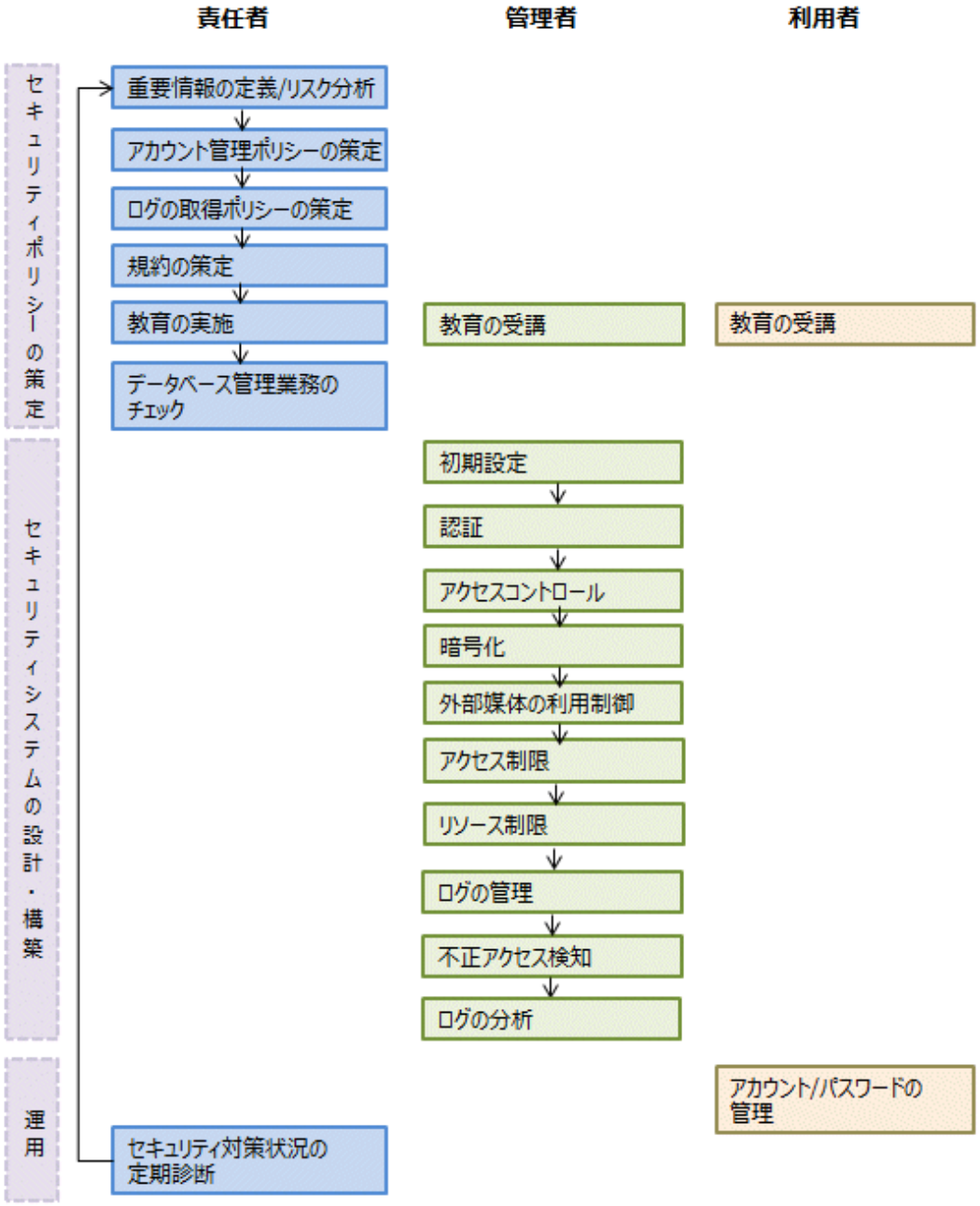
第2章 セキュリティ運用の概要

2.1 セキュリティ運用の流れ

FUJITSU Enterprise Postgresでセキュリティ環境を構築し、セキュリティ運用を行う場合の作業の流れを示します。

セキュリティ運用を行う場合、システムにセキュリティ機能を装備することによりセキュリティ脅威に対応する技術的な運用作業と、セキュリティの指針、教育制度および利用規則の制定などを実施する人的な作業があります。

図2.1 セキュリティ運用の流れ



第3章 責任者の作業

責任者は、セキュリティ対策の指針となるセキュリティポリシーを策定します。

3.1 重要情報の定義/リスク分析

セキュリティポリシーを策定する前に、重要情報の定義とリスク分析を行います。情報の重要度やリスク分析の結果に基づいて、どのようなセキュリティ対策を講じるのかを決定します。

重要情報の定義では、セキュリティ対策を有効に実施するため、守るべき情報を洗い出し、情報を重要度により分類します。守るべき情報としては、“情報資産について”に示したように、“データベース管理情報”、“データベース一般情報”があります。洗い出した情報は、“個人情報”、“機密情報”といった、情報の重要度により分類します。

リスク分析では、“想定される脅威”を参考に、起こりうる脅威を洗い出して、脅威に対するリスクを分析します。

また、年に一度を目安に、リスク分析を行うことで、業務に悪影響を及ぼす可能性がある脅威や関連する脆弱性を識別することができます。

3.2 アカウント管理ポリシーの策定

アカウント管理ポリシーの策定では、以下を実施し、策定したポリシーを文書化します。

システム利用者と役割の整理

“登場人物について”にもとづいて、対象とするシステムで必要な役割を洗い出します。また、役割ごとに要員を整理します。

アカウントの整理

役割ごとに適切な権限を持つようにアカウントを整理し、アカウントのポリシーを決定します。

- データベース管理者アカウント
  - データベース管理者とデータベース運用者でアカウントを分ける
  - データベース管理者アカウントは、特定の人しか利用できないようにする
  - データベース管理者権限を必要としない作業は、データベース管理者権限なしの別アカウントで実施する
- 一般アカウント

  アプリケーションの用途ごとに一般利用者のアカウントを作成する

アカウント管理ポリシーの見直し

セキュリティ対策を有効に実施するため、アカウントに関して、以下の見直しを実施します。

- 整理したアカウント、権限を定期的にチェックし、状況に適しているかを判断する
- システム変更や運用変更があった場合は、アカウント、権限を見直す
- 不適切なアカウント、権限を発見した場合は、再度、アカウントを整理する

3.3 ログの取得ポリシーの策定

ログの取得ポリシーの策定では、以下を実施し、策定したポリシーを文書化します。

ログ取得の目的を整理

何のためにログを取得するのかを明確にするため、取得するログの目的を整理します。

たとえば、目的の例として、「不正アクセス時の調査に利用する」、「何らかの問題が発生した場合に、捜査機関への証拠として提出する」などがあります。

取得するログの種類を決定

適切なログを取得するため、対象とするシステムにおいて取得できるログの種類を整理し、取得するログを決定します。

たとえば、ログの種類として、「OSログ」、「アプリケーションの実行ログ」、「データベースの監査ログ」などがあります。

ログ取得対象アクセスの整理

ログ取得の対象とするアクセスを決定するため、どのようなアクセスがあるかを整理します。

たとえば、以下のアクセスが考えられます。

- 重要情報に関するアクセス
  - 個人情報、機密情報、データベース管理情報へのアクセス
  - 業務時間外のアクセス
  - ログイン
  - 特定のSQL
- 不正が疑われるアクセス
  - 大量検索アクセス
  - 異なる場所からのアクセス
  - 業務時間外のアクセス

ログ取得内容の決定

取得したログを有効利用するため、ログとして必要な内容を整理し、取得内容を決定します。

たとえば、以下の出力内容が考えられます。

- いつ (時間)
- 誰が (データベースアカウント、アプリケーションユーザー)
- 何を (オブジェクトID、テーブル名)
- どこから (マシン名、IPアドレス)
- どうやって (SQL種別、SQL文)
- 実行結果 (成功/失敗)

ログの保全方針の策定

ログを正当な状態で利用するため、ログの保全指針を整理します。

たとえば、「保管場所」、「保管媒体」、「保管期間」、「アクセス制御」などがあります。

3.4 規約の策定

対象とするシステムのセキュリティ対策の基準となる規約を策定します。また、セキュリティ違反時の罰則を規定します。たとえば、以下のような規約や罰則を策定します。

- 規約の策定
  - セキュリティパッチや更新プログラムの適用
  - 無許可でのデータベースからの情報入手の禁止
  - 入手した情報について、許可された媒体以外への保存を禁止
- 不正利用による罰則の規定
  - 就業規則での罰則の規定
  - 罰則金の設定

3.5 教育の実施

情報セキュリティの重要性や必要性を認識させ、作業漏れや誤操作による不正アクセスを防ぐため、管理者および利用者に対して、セキュリティに関する教育を実施および推進します。

たとえば、「セキュリティポリシーの広報」、「教育スケジュールの策定」、「教育資料の策定」などを実施します。

3.6 データベース管理業務のチェック

管理者の作業ミスや不正行為を防止するため、以下の対策を実施します。

- データベースに関するセキュリティの事故や脆弱性の最新情報を常に収集する
- 管理業務は、事前申請を行ったうえで実施する
- 管理業務の記録を残す

3.7 セキュリティ対策状況の定期診断

セキュリティ対策が有効であるかを確認するため、セキュリティの脅威にもとづいたセキュリティ対策が適切に講じられているかを定期的に診断します。

また、脅威や脆弱性などに対し、現在のセキュリティ対策やセキュリティポリシーが有効であるかを評価し、問題がある場合は、セキュリティポリシーやセキュリティ対策を見直します。

第4章 管理者の作業

管理者は、責任者が策定したセキュリティポリシーに従い、システムの設計/構築/運用を行う際に、セキュリティ対策として以下を実施します。

事前準備

- 教育の実施

不正行為を防御するための対策

- 初期設定
- 認証
- アクセスコントロール
- 暗号化
- 外部媒体の利用制御
- アクセス制御
- リソース制限

不正行為を検知/追跡するための対策

- ログの管理
- 不正アクセス検知
- ログの分析

4.1 教育の受講

管理者は、責任者が策定した教育スケジュールに従って、セキュリティに関する教育を受講します。また、利用者に対し、教育の受講を指示します。

4.2 初期設定

データベースの脆弱性および不正アクセスの可能性が最小限となるように、システム構築の初期段階で以下のセキュリティ対策を実施します。また、データベースサーバでは、主として、データベースシステムのみを稼働させるような構成とします。

サーバの堅牢化

データベースサーバへの侵入や破壊などを防ぎ、セキュアなサーバでシステムが稼働されるようにOSやネットワークを設定します。

- 使用するOSにおいて、不要な機能やサービスを削除する
- 必要なプロトコルのみを有効にし、不要なプロトコルは無効にする
- ファイル共有、FTPなどセキュリティのレベルが比較的低いとされるサービス、プロトコル、デーモンにセキュリテイ機能を実装する

最新バージョンの導入

最新のセキュリティ対策を反映するため、常に最新のパッチを入手し、適用します。

必要最低限の機能の導入

システムの不正利用を防ぐため、インストール時に、必要な機能のみを導入します。

また、使用しない機能やサービスは、削除または無効にします。

ポートの変更

システムの不正利用を防ぐため、インストール時に作成されるデフォルトのポートを変更します。

ポイント

ポートは、FUJITSU Enterprise Postgresのセットアップで指定します。詳細は、“FUJITSU Enterprise Postgres 導入ガイド(サーバ編)”を参照してください。

通信機能のアクセス制限

通信機能を利用したシステムの不正利用を防ぐため、通信機能のアクセス制限を実施します。

データベース構成ファイルへのアクセス経路禁止の設定

データベース破壊を防ぐため、以下の対策を実施します。

- データベース構成ファイルへのアクセス許可者を制限し、定期的に権限見直しを実施する
- 表や定義スクリプトに対しては、管理者だけがアクセス可能とする

データベースへのアクセス経路の制限

データベースの不正利用や操作ミスを防止するため、データベースをアクセスするためのアプリケーションの配布範囲は、アクセス許可者が使用する機器のみに限定します。

不正プログラムの対処

アプリケーションのプログラムソースコードの改ざんなど、バックドアによるシステムへの不正侵入を防ぐため、稼働するプログラムは作成者を明文化し、改ざんなどがされないように、チェックおよびテストを実施します。また、安全なコーディング手法を採用し、一般的なコーディングの脆弱性の問題に対応できるようにしておくことも必要です。

システムのセキュリティ設定

システムのセキュリティ設定において、セキュリティに影響を与えることが明らかである場合、セキュリティパラメータを適切に設定するなどして、初期設定の段階で、確実にセキュリティ設定を実施します。

4.3 認証

悪意を持った利用者のなりすましによる情報漏えいや改ざんを防ぐため、データベースへのアクセス時には、必ず認証を行う必要があります。

パスワード認証は、データベースにログインする際に使用しますが、管理者は、認証で使用するアカウントやパスワードを厳重に管理します。

また、許可されたユーザーだけがデータベースにアクセスできるように、クライアントからデータベースへの接続に対する認証も確実に実施します。

4.3.1 アカウントの管理

アカウントの管理では、以下を実施します。

必要なアカウントの作成

なりすましなどのアカウントの不正利用を防ぐため、アカウントの作成時に、以下の対策を実施します。

- 必要なアカウントを選定する
- 利用者の権限を規定する
- データベース管理者アカウントと一般アカウントは、それぞれの権限に応じて、別々に作成する

ポイント

アカウントは、CREATE ROLE文で作成します。詳細は、“PostgreSQL Documentation”の“CREATE ROLE”を参照してください。

不要なアカウントの削除

「未使用のアカウント」や「製品インストール時にデフォルトで作成される業務で不要なアカウント」など、日常的に使用されないアカウントを削除します。

ポイント

アカウントは、DROP ROLE文で削除します。詳細は、“PostgreSQL Documentation”の“DROP ROLE”を参照してください。

アカウントのロックアウトの設定

定期的にアカウントの使用頻度をチェックし、長期間使用されないアカウントがある場合は、アカウントをロックします。また、ログインの失敗回数の許容回数を設定し、この回数を超えてログインに失敗した場合は、アカウントをロックします。また、アカウントがロックされた場合に、アカウントが再有効化されるまでの期間を設定します。

ポイント

LDAP認証を使用することで、アカウントのロックアウトを設定することができます。詳細は、“PostgreSQL Documentation”の“LDAP Authentication”を参照してください。

データベース管理者アカウントの管理

責任者が策定したアカウント管理ポリシーに従い、データベース管理者アカウントを管理します。

開発環境と運用環境のアカウント

開発環境で利用したアカウントの不正利用を防ぐため、運用環境での運用が開始される前に、開発環境で使用したアカウントを削除します。やむを得ず、開発環境で利用したアカウントを運用環境で利用する場合は、それぞれの環境において異なるパスワードを使用します。

一時利用アカウントの設定

一時的な利用者がシステムを利用する場合、共有アカウントに対し、利用ごとに一時的なパスワードを与えるか、または、一時的なアカウントを作成します。

4.3.2 パスワードの管理

パスワードの管理では、以下を実施します。

パスワードの複雑化

アカウントのパスワードは、他者に容易に推察されないように、「IDと同じパスワード」や「インストール時のデフォルトパスワード」の使用を禁止します。また、パスワードには、複雑性と強度を設定します。

パスワードの定期的な変更

パスワードを不正入手された場合に備え、パスワードの定期的な変更を設定します。また、初回利用時に使用したパスワードの変更を促すため、初回利用時のパスワードの強制的な変更を設定します。

パスワードの有効期限の設定

パスワードの定期的な変更を促進するため、パスワードに有効期限を設定します。

ポイント

パスワードの設定/変更は、CERATE ROLE文またはALTER ROLE文で指定します。詳細は、“PostgreSQL Documentation”の“CREATE ROLE”および“ALTER ROLE”を参照してください。

また、passwordcheckやLDAP認証を使用することで、以下を実施することができます。

- インストール時のデフォルトパスワードの変更
- パスワードの有効期間の設定
- パスワードの文字数や文字種のチェック

passwordcheckについては、“PostgreSQL Documentation”の“passwordcheck”を参照してください。また、LDAP認証については、“PostgreSQL Documentation”の“LDAP Authentication”を参照してください。

4.3.3 接続と認証の設定

利用者がデータベースを利用する場合に、許可された利用者だけがデータベースをアクセスできるように接続と認証を設定します。

ポイント

pg_hba.confでクライアント認証を設定します。詳細は、“PostgreSQL Documentation”の“Client Authentication”を参照してください。

4.4 アクセスコントロール

管理者および利用者に対して適切なアクセス権限が設定されていない場合、意図しない人物による情報アクセスが発生し、情報漏えいなどのセキュリティ事故が発生してしまう可能性があります。これらを防御するため、アクセス権限において、以下のセキュリティ対策を実施し、ルールに基づいたアクセスコントロールを行う必要があります。

ポイント

アクセス権限設定における留意点

- 全ユーザーに権限を付与できる特殊なアカウントの作成は禁止する
- 業務データなどの一般情報にアクセスできる一般アカウントの作成は禁止する

データベースへのアクセス要件の洗い出し

データベースの利用目的ごとに適切なアクセス権限を設定するため、以下の手順でアクセス要件を洗い出します。

1. 「データベース管理用」、「オブジェクト管理用」、「データアクセス用」など、アカウントの利用目的を分類します。
2. 「機能別」、「オブジェクト別」などアカウントの利用目的ごとに必要な権限を分類します。
3. それぞれの権限に基づいて、アカウントを分割します。
4. 分割したアカウントのそれぞれについて、アクセスする必要がある必要最小範囲のデータと必要最低限のアクセス内容(参照、更新、作成、削除)を洗い出し、データベースへのアクセス要件を決定します。

アクセス権限の設定

分割された各アカウントについて、データベースへのアクセス要件に基づいて、必要最低限の権限を付与します。また、管理者権限は、アカウントを限定して付与します。

アクセス権限の見直し

システムに対するアクセス要件の変化に対応するため、アクセス権限を定期的に見直し、不要なアクセス権限がないかを確認します。不要なアクセス権限が設定されている場合は、速やかにアクセス権限を修正します。

ポイント

アクセス権限は、GRANT文やREVOKE文で設定します。詳細は、“PostgreSQL Documentation”の“GRANT”および“REVOKE”を参照してください。

4.5 暗号化

データの盗難や通信の盗聴など、情報漏えい発生時のデータの不正利用を防ぐため、以下の暗号化対策を実施します。

通信の暗号化

データベースサーバとクライアント間におけるネットワーク上の盗聴に備え、暗号化機能を使用し、通信の暗号化を実施します。

通信の暗号化については、“FUJITSU Enterprise Postgres 運用ガイド”の“Secure Sockets Layerによる安全な通信の構成”を参照してください。

データの暗号化

データの盗難に備え、暗号化機能を使用し、データの暗号化を実施します。暗号化は、以下のデータを対象とします。

- データベースへの格納データ
- バックアップデータ
- データファイル

データの暗号化については、“FUJITSU Enterprise Postgres 運用ガイド”の“透過的データ暗号化による格納データの保護”を参照してください。

暗号鍵の管理

暗号化で使用する暗号鍵に対しては、アクセス可能な人物を最小限のデータベース管理者に限定します。

また、暗号化された情報が容易に復号されないように、ライフサイクル(生成、配布、保存、廃棄など)ごとに暗号鍵を適切に管理する仕組みを作り、暗号鍵を厳重に管理します。

暗号鍵の管理については、“FUJITSU Enterprise Postgres 運用ガイド”の“Secure Sockets Layerによる安全な通信の構成”および“透過的データ暗号化による格納データの保護”を参照してください。

4.6 外部媒体の利用制御

データベースに接続される外部媒体(CD/DVD、USBメモリ、外付けハードディスクなど)や端末の利用を制御し、データベースからのデータの持ち出しを制限することで、情報漏えいを未然に防ぐことができます。

外部媒体の接続制限

業務で使用しない外部媒体やプリンタを取り外し、情報の書き出し先となる外部媒体の接続を制限します。

外部媒体の利用制限

外部媒体やプリンタに対する接続制限を実施し、外部媒体への情報の書き出しを抑止します。

接続端末の利用制御

データベースに接続する端末に対して、以下の対策を実施することで、端末からの情報流出を防御します。

- 端末に対する外部媒体の接続制限を実施する
- 端末に対し、堅牢化のためのセキュリティ対策を実施する
- 端末アクセスの個人認証を実施する
- インストールされているソフトウェアを管理し、ソフトウェアの利用状況を監視する
- プリンタへの接続制限を実施する

4.7 サーバ/アプリケーションに対するセキュリティ対策

データベースに対するセキュリティ対策に加え、サーバやアプリケーションなどに対するセキュリティを強化することで、より強固なセキュリティシステムを実現できます。サーバやアプリケーションに対して、以下のセキュリティ対策を実施します。

アクセス制限

以下の対策を実施し、データベースサーバへのアクセスを制限します。

- データベースサーバは、ファイアウォールより内側に設置し、不特定多数の端末から直接データベースサーバにアクセスできないようにする。
- ローカルネットワークにおいては、ルータによりIPアドレスを制限するなどの対策を実施し、直接データベースサーバにアクセスできる端末やセグメントを限定する。

リソース制限

サービスの妨害や大量データの抽出を制限するため、一般利用者における、想定を超えるCPUリソースの使用を制限します。

4.8 ログの管理

ログは、管理者の権限乱用、利用者のデータベースへの不正アクセスなどの脅威に対抗するための機能です。情報漏えいや不正アクセスが発生した場合に、原因を特定するため、データベースに対する処理や操作を調査/追跡するための情報をログとして取得し管理します。

FUJITSU Enterprise Postgresでは、ログを取得、管理するための機能として、監査ログ機能が提供されています。監査ログ機能については、“第6章 監査ログ機能”を参照してください。

ここでは、情報漏えいや不正アクセスへの対策として、ログとして取得すべき情報とログの保全方法を説明します。

4.8.1 ログの取得

責任者が策定したログの取得ポリシーに従い、以下の監査ログを取得します。

ログイン情報

ログインおよびログアウト時のログを取得します。

データベースへのアクセス情報(参照/更新)

以下の情報に関するすべてのアクセスを取得します。

- データベースの一般情報（業務で使用する個人情報や機密情報など）
- データベースの管理情報（システムカタログ、ユーザーID/パスワードなど)

データベースオブジェクトの変更情報

データベースアカウント、テーブルなどのデータベースオブジェクトの作成、変更、削除に関するログを取得します。

監査ログに対する操作ログ

取得した監査ログの隠ぺいを防ぐため、監査ログの初期化や監査ログ機能の停止をログとして取得します。

4.8.2 ログの保全

責任者が策定したログの取得ポリシーに従い、ログの保全を実施します。

ログの保管

取得したログが他者に更新されないように、以下を実施し、ログを安全に保管します。

- ログを外部記憶媒体に保存し、外部記憶媒体は、施錠できる場所など、安全な場所に保管する
- ログの参照は、管理者のみに限定し、更新権を付与しないなど、ログにアクセス制限を設定する
- 問題発覚時期から遡った調査が必要な場合を考慮し、ログの保管期間を決定する

ログの改ざん防止

ログを多重化する、書き換え不可能なストレージを使用するなど、ログが改ざんされないような対策を実施します。

ログの暗号化

ログが容易に参照されないように、ログを暗号化します。

4.9 不正アクセス検知

不正アクセスに対処するため、データベースへの不正アクセスを検知する仕組みを設けて、アクセスを監視する必要があります。

不正アクセスの通知

ログイン失敗の許容回数を超えたアカウントロックが発生した場合、責任者および管理者に通知するなど検知した不正アクセスを通知する仕組みを作ります。

アクセス時間のチェック

以下の情報に対し、正規のアクセス時間帯以外の疑わしいアクセスがチェックできるような仕組みを作るとともに、それぞれの対策を実施します。

データベース管理情報へのアクセス検知

- ログを監視し、申請されていない時間帯のアクセスを検知する
- 正規のアクセス時間帯以外のアクセス許可を申請された場合、申請された内容と作業結果に相違のないことをログを見て確認する

データベース一般情報へのアクセス検知

- 一般アカウントごとに、データベースへのアクセスを許可する時間帯を決める
- セション情報のログから、正規のアクセス時間帯でないアクセスを検知する

アクセス不可の接続元のチェック

許可されていない接続元からのアクセスを検知するため、アクセス可能な接続元を定義し、許可されていない接続元からのアクセスを検知します。

データベース管理者アカウントおよび一般アカウントによるアクセスのパターン(接続元/OSユーザー/アカウント)を定義し、このパターン以外でのアクセスをチェックします。

4.10 ログの分析

情報漏えいや不正アクセスなどが疑われた場合、ログを分析する仕組みを作り、取得したログを分析することで、不正行為を検出します。以下の分析を実施します。

定期的なセション情報の分析

不正なログインを検出するため、以下のような観点でログのセション情報を分析します。

- ログイン失敗回数が多いセションの傾向
- 長時間ログインしているセションの傾向
- 大量のリソースを消費するセションの傾向

定期的なデータベースアクセス情報の分析

不正なデータベースへのアクセスを検出するため、以下のような観点でSQL文を分析します。

- 長時間に渡り実行されているSQLの傾向
- 大量のリソースを消費するSQLの傾向

第5章 利用者の作業

利用者は、システムを利用する際に、セキュリティ対策として以下を実施します。

5.1 教育の受講

利用者は、責任者または管理者の指示するセキュリティに関する教育を受講して、セキュリティについての知識を習得する必要があります。利用者間でセキュリティに関する共通の意識を持つことで、より強固なセキュリティシステムの確立が実現できます。

5.2 アカウント/パスワードの管理

利用者は、管理者から提示されたアカウントおよびパスワードを使用することで、データベースシステムを利用できます。その際、アカウントおよびパスワードを他人に悪用されないように、以下の対策を実施します。

- ログイン時にアカウントロックがかからないように、IDおよびパスワードは責任を持って管理する
- 定期的にパスワードを変更する
- パスワードに設定されている有効期限を遵守し、期限が近づいたら速やかにパスワードを変更する

第6章 監査ログ機能

PostgreSQLでは、ログ出力機能を利用することで、サーバログとして出力されたログを監査ログとして利用することができますが、SQL実行時のログとして、スキーマ名が出力されないなど、適切な分析ができないログが存在します。また、出力条件について、詳細な指定ができないため、ログ量が多く、性能劣化につながる可能性があります。

FUJITSU Enterprise Postgresの監査ログ機能は、pgauditに機能を拡張することで、データベースアクセスに関する詳細な情報を監査ログとして取得することができます。また、監査ログは、専用ログファイルまたはサーバログに出力できます。これにより効率的かつ正確なログ監視が可能になります。

注意

PostgreSQLがシングルユーザーモードで起動している場合には、監査ログ機能を利用することはできません。

6.1 監査ログの出力モード

pgauditでは、以下の2種類の監査ログを出力することができます。

Session Audit Logging

Session Audit Loggingは、バックエンドプロセス(クライアントから接続要求を受けたときに生成されるプロセス)で実行されたSQLに関する情報、データベースの起動や接続に関する情報、エラーに関する情報をログとして出力します。Session Audit Loggingでは、ログ出力のための条件を指定し、出力するログを絞り込むことで、大量のログ出力に対する性能劣化を防ぐことができます。

Session Audit Loggingについては、“6.4 Session Audit Logging”を参照してください。

Object Audit Logging

Object Audit Loggingは、特定のオブジェクト(テーブルや列)に対して、SELECT、INSERT、UPDATE、DELETEが実行されたときに、ログとして出力します。TRUNCATEには対応していません。Object Audit Loggingでは、指定されたロールに対して、権限が付与されているオブジェクト操作をログとして出力します。Session Audit Loggingよりもさらに細かい粒度でログ出力を制御することができます。

Object Audit Loggingについては、“6.5 Object Audit Logging”を参照してください。

参考

アプリケーションやコマンドによっては、FUJITSU Enterprise Postgresが内部でSQLを実行し、その監査ログが取得される場合があります。

また、同じステートメントIDを持つ複数のSQLの監査ログが取得される場合があります。これは、利用者がSQLを実行する場合に、FUJITSU Enterprise Postgresによって、内部で別のSQLが実行されるためです。

6.2 セットアップ

pgauditのセットアップ方法を説明します。

1. pgauditのファイルコピー

   スーパーユーザーで以下のコマンドを実行します。“<*x*>”は、製品のバージョンを示します。

   ```
   $ su -
   Password:******
   # cp -r /opt/fsepv<x>server64/OSS/pgaudit/* /opt/fsepv<x>server64
   ```

2. pgaudit設定ファイルの作成

   pgauditの動作に必要な情報を記述するpgaudit設定ファイルを任意のファイル名で作成します。pgaudit設定ファイルは、データベースの符号化方式と同じエンコードで作成してください。

   また、監査ログに関するポリシーを意図しないユーザーに参照されないようにするために、pgaudit設定ファイルにはデータベース管理者にのみ読み込み権限を設定してください。

   pgaudit設定ファイルの詳細については、“6.3 pgaudit設定ファイル”を参照してください。

注意

この時点では、pgaudit設定ファイルのruleセクションは定義しないでください。

pgaudit設定ファイルの例

```
[output]
logger = 'auditlog'
```

3. postgresql.confの設定

postgresql.confに、監査ログを利用するための以下のパラメータを設定します。

shared_preload_libraries

"pgaudit"を指定します。

pgaudit.config_file

pgaudit設定ファイルの配置先パス名を指定します。

相対パスで指定した場合は、データ格納先のディレクトリからの相対パスとなります。

log_replication_commands

"on"を指定します。

log_min_messages

ERROR以上が指定されていることを確認します。

監査ログをサーバログに出力する場合(pgaudit設定ファイルのloggerパラメータに"serverlog"を指定)は、サーバログに関する以下のパラメータを確認します。

logging_collector

"on"が指定されていることを確認します。

log_destination

"stderr"が指定されていることを確認します。

log_file_mode

許可された人物だけにサーバログへのアクセスを許可するために、サーバログのパーミッションが適切であることを確認します。

参考

log_file_modeパラメータのデフォルトは0600であり、データベース管理者のみアクセス可能です。

たとえば、データベース管理者のグループにも参照を許可する場合は、log_file_modeに0640を指定します。

例

```
log_file_mode = 0640
```

0000を指定することで、データベース管理者も参照不可にすることができます。ただし、ログ出力を行うため、書き込み権限は付与された状態となります。

監査ログを専用ログファイルに出力する場合(pgaudit設定ファイルのloggerパラメータに"auditlog"を指定)は、以下のパラメータを確認します。

max_worker_processes

max_worker_processesパラメータを設定している場合は、指定されている値に1を追加します。

参照

サーバログの詳細は“PostgreSQL Documentation”の“Error Reporting and Logging”を参照してください。

データベース多重化運用を使用する場合は、“6.6 データベース多重化運用の場合”を参照してください。

postgresql.confの例

下記の例では、監査ログ機能利用時に設定が必要なパラメータのみ記載しています。

```
shared_preload_libraries = 'pgaudit'
pgaudit.config_file = 'pgaudit.conf'
log_replication_commands = on
log_min_messages = WARNING
```

4. インスタンスの起動

   インスタンスを起動し、以下のメッセージが出力されることを確認してください。

```
LOG: pgaudit extension initialized
```

5. pgauditエクステンションの作成

   CREATE EXTENSIONを使用し、pgauditエクステンションを作成します。

```
$ psql
=# CREATE EXTENSION pgaudit;
=# ¥dx
                  List of installed extensions
Name    | Version |   Schema   |     Description
--------+---------+------------+---------------------------------
pgaudit | 1.0     | public     | provides auditing functionality
plpgsql | 1.0     | pg_catalog | PL/pgSQL procedural language
(2 rows)
```

6. pgaudit設定ファイルのパラメータ設定

   pgaudit設定ファイルのパラメータを追加/変更します。

   pgaudit設定ファイルの詳細は“6.3 pgaudit設定ファイル”を参照してください。

7. インスタンスの再起動

   pgaudit設定ファイルの変更を適用するため、インスタンスを再起動します。再起動後、変更内容が正しく反映されていることを確認してください。

```
LOG:  log_catalog = 1
LOG:  log_level_string =
LOG:  log_level = 15
LOG:  log_parameter = 0
LOG:  log_statement_once = 0
LOG:  role =
LOG:  logger = auditlog
LOG:  log_directory = pgaudit_log
LOG:  log_filename = pgaudit-%Y-%m-%d_%H%M%S.log
LOG:  log_file_mode = 0600
LOG:  log_rotation_age = 1440
LOG:  log_rotation_size = 10240
LOG:  log_truncate_on_rotation = 0
LOG:  fifo_directory = /tmp
LOG:  Rule 0
LOG:  pgaudit extension initialized
```

6.3 pgaudit設定ファイル

pgaudit設定ファイルには、pgauditの動作に必要な情報を指定します。pgaudit設定ファイルは、“outputセクション”、“optionセクション”、“ruleセクション”の3つのセクションから構成されています。

outputセクション

outputセクションは、以下の形式で指定します。

- パラメータ名 = '値'

outputセクションで指定可能なパラメータを下表に示します。

パラメータ名	説明	備考
logger	監査ログの出力先について、専用ログファイル(auditlog)/サーバログ(serverlog)を指定します。 デフォルトは、“auditlog”(専用ログファイル)です。	専用ログファイルは、データベースの符号化方式と同じエンコードで出力されます。
log_directory	監査ログを作成するディレクトリを指定します。ディレクトリは、絶対パスまたはデータ格納先のディレクトリからの相対パスで指定します。 デフォルトは、“pgaudit_log”です。	loggerパラメータに“auditlog”を指定した場合のみ有効
log_filename	監査ログのファイル名を指定します。postgresql.confファイルのlog_filenameと同様に、時刻によって変動するファイル名を指定します。 デフォルトは、“pgaudit-%Y-%m-%d_%H%M%S.log”です。	loggerパラメータに“auditlog”を指定した場合のみ有効
log_file_mode	許可された人物だけに監査ログファイルへのアクセスを許可するために、監査ログファイルのパーミッションを指定します。パラメータの値はchmod、およびumaskシステムコールで許容されるフォーマットで指定される数値モードです。デフォルトは“0600”です。 監査ログファイルのパーミッションについては、“6.2 セットアップ”の“log_file_mode”を参照してください。	loggerパラメータに“auditlog”を指定した場合のみ有効
log_rotation_age	監査ログファイルの最大寿命を指定します。ここで指定した時間(分単位)が経過すると、新しい監査ログファイルが生成されます。“0”を設定することで、時間に基づいた新しいログファイルの生成は無効になります。 指定可能な単位はmin(分)、h(時間)、d(日数)です。単位を省略した場合はmin(分)となります。 デフォルトは、“1d”(1日)です。	loggerパラメータに“auditlog”を指定した場合のみ有効
log_rotation_size	監査ログファイルの最大容量を指定します。ここで指定したキロバイト分のログがログファイルに出力された後、新しいログファイルが生成されます。“0”を設定することで、サイズに基づいた新しいログファイルの生成は無効になります。 指定可能な単位はkB(キロバイト)、MB(メガバイト)、GB(ギガバイト)です。単位を省略した場合はkB(キロバイト)となります。 デフォルトは、“10MB”です。	loggerパラメータに“auditlog”を指定した場合のみ有効
log_truncate_on_rotation	監査ログファイルを時間を基にしてローテーションする場合に、監査ログを既存の同名のファイルに上書きする(on)/上書きしない(off)を指定します。例えば、“on”を指定し、log_filenameに“pgaudit-%H.log”を指定すると、24個の時別なログファイルが生成され、それらは周期的に上書きされることになります。	loggerパラメータに“auditlog”を指定した場合のみ有効

パラメータ名	説明	備考
	デフォルトは、“off”です。“off”を指定すると既存の監査ログファイルに追記します。	
fifo_directory	監査ログファイルを出力するデーモンプロセスとバックエンドプロセスの間で使用するFIFO(名前付きパイプ)のディレクトリを指定します。p.PGAUDIT.nnnnという名前のFIFO(nnnnはpostmasterのPID)がfifo_directoriesディレクトリの中に作成されます。いずれのファイルも手作業で削除することはできません。 デフォルトは、“/tmp”です。	loggerパラメータに“auditlog”を指定した場合のみ有効

参考

loggerパラメータに“serverlog”を指定した場合、監査ログはサーバログにログメッセージとして出力されるため、監査ログの先頭にpostgresql.confファイルのlog_line_prefixパラメータによるステータス情報とメッセージ深刻度レベルが出力されます。

loggerパラメータを省略、または“auditlog”を指定した場合、監査ログは専用ログとして専用ログファイルに出力されるため、postgresql.confファイルのlog_line_prefixパラメータによるステータス情報やメッセージ深刻度レベルは出力されません。

監査ログの出力形式については、“6.4 Session Audit Logging”の“出力形式”または“6.5 Object Audit Logging”の“出力形式”を参照してください。

ポイント

log_file_modeパラメータは、postgresql.confのlog_file_modeパラメータの設定およびinitdbコマンドの--allow-group-accessオプションによる影響は受けません。また、監査ログを専用ログファイルに出力する場合、log_directoryパラメータで指定するディレクトリのデフォルト値はデータ格納先のディレクトリ配下になります。そのため、log_file_modeパラメータで指定したパーミッションが、initdbコマンドの--allow-group-accessオプションの対象となるデータ格納先のパーミッションと競合する場合には、log_directoryパラメータで指定するディレクトリはデータ格納先以外を指定してください。

optionセクション

optionセクションは、以下の形式で指定します。

- パラメータ名 = '値'

optionセクションで指定可能なパラメータを下表に示します。

パラメータ名	説明	備考
role	Object Audit Loggingで使用するroleの名前を指定します。 大文字、キーワード、マルチバイト文字およびカンマを指定する場合は、二重引用符で囲んでください。	Object Audit Loggingのみで使用するパラメータ
log_catalog	pg_catalogに対するログ出力の有効(on)/無効(off)を指定します。 pgAdminなど、pg_catalogにアクセスする監査ログを取得しない場合に“off”を指定します。 デフォルトは、“on”(有効)です。	
log_parameter	SQL実行でパラメータで渡した値の出力について、有効(on)/無効(off)を指定します。 デフォルトは、“off”(無効)です。	

パラメータ名	説明	備考
log_statement_once	同じSQLがログの出力対象となる場合、2回目以降のSQLの出力について、抑止する(on)/抑止しない(off)を指定します。 デフォルトは、“off”(抑止しない)です。	
log_level	監査ログのログレベルを指定します。 有効な値は、“DEBUG5”、“DEBUG4”、“DEBUG3”、“DEBUG2”、“DEBUG1”、“INFO”、“NOTICE”、“WARNING”、および“LOG”です。 デフォルトは、“LOG"です。	loggerパラメータに“serverlog”を指定した場合のみ有効

ruleセクション

ruleセクションは、Session Audit Loggingで使用されるセクションです。“6.4 Session Audit Logging”を参照してください。

注意

optionセクションにroleパラメータを指定した場合は、ruleセクションを指定しないでください。ruleセクションを指定すると、Object Audit LoggingとSession Audit Loggingの監査ログが混在して出力されるため、CSV形式で参照できなくなるためです。

6.4 Session Audit Logging

Session Audit Loggingでは、pgaudit設定ファイル内のruleセクションで、出力するログを絞り込むためのルールを指定します。

ルールは、以下の形式で指定します。値はカンマ区切りで複数指定できます。

- パラメータ名 = '値'
- パラメータ名 != '値'

ruleセクションに[rule]のみ記載し、パラメータを指定しない場合、すべての監査ログが出力されます。

例

```
[output]
logger = 'auditlog'
[rule]
```

また、ruleセクションを記載しない([rule]を記載しない)場合、監査ログは出力されません。

例

```
[output]
logger = 'auditlog'
```

ruleセクションで指定可能なパラメータを下表に示します。

パラメータ名	説明	指定例
timestamp	タイムスタンプの範囲。 指定形式については、“timestamp”を参照してください。	timestamp = '09:00:00 - 10:00:00, 18:00:00 - 18:30:00'
database	データベース名。 空の値を指定する場合は、二重引用符を2つ続けて指定("")してください。大文字、キーワード、	database = 'product_db'

パラメータ名	説明	指定例
	マルチバイト文字およびカンマを指定する場合は、二重引用符で囲んでください。	
audit_role	ロール名。 空の値を指定する場合は、二重引用符を2つ続けて指定("")してください。大文字、キーワード、マルチバイト文字およびカンマを指定する場合は、二重引用符で囲んでください。	audit_role = 'appuser1'
class	操作のクラス。 以下から選択します。複数指定が可能です。各クラスの意味については、“class”を参照。 ・ BACKUP ・ CONNECT ・ DDL ・ ERROR ・ FUNCTION ・ MISC ・ READ ・ ROLE ・ WRITE ・ SYSTEM	class = 'READ, WRITE'
object_type	オブジェクトの種類。 classパラメータが“READ”および“WRITE”の場合に有効になります。 以下から選択します。複数指定が可能です。 ・ TABLE ・ INDEX ・ SEQUENCE ・ TOAST_VALUE ・ VIEW ・ MATERIALIZED_VIEW ・ COMPOSITE_TYPE ・ FOREIGN_TABLE ・ FUNCTION	object_type = 'TABLE, INDEX'
object_name	オブジェクト名。 classパラメータが“READ”および“WRITE”の場合に有効になります。 テーブルなどスキーマ修飾できるオブジェクトは、スキーマ名で修飾してください。 大文字、キーワード、マルチバイト文字およびカンマを指定する場合は、二重引用符で囲んでください。“スキーマ名.テーブル名”といったオ	object_name = 'myschema.tbl1' object_name = '"myschema.テーブル1"' object_name = 'myschema.tbl1, "マイスキーマ.TABLE"'

パラメータ名	説明	指定例
	ブジェクト名を指定する場合は、スキーマ名修飾したオブジェクト名全体を二重引用符で囲んでください。 空の値を指定する場合は、二重引用符を2つ続けて指定("")してください。	
application_name	アプリケーション名。 空の値を指定する場合は、二重引用符を2つ続けて指定("")してください。	application_name = 'myapp'
remote_host	接続先のホスト名またはIPアドレス。 postgresql.confファイルのlog_hostnameパラメータに“on”を指定している場合は、ホスト名を指定してください。それ以外の場合は、IPアドレスを指定してください。なお、ローカルホストの場合は“[local]”を指定してください。 空の値を指定する場合は、二重引用符を2つ続けて指定("")してください。	remote_host = 'ap_server'

timestamp

ログ出力の対象を、「開始時刻」から「終了時刻」までのタイムスタンプの範囲で指定します。指定形式は、'hh:mm:dd-hh:mm:dd'(hhは24時間表記、また、hh、mm、ddは2桁で表記)です。

開始のタイムスタンプは、終了のタイムスタンプよりも小さい値を指定する必要があります。複数の範囲を指定する場合はタイムスタンプの開始と終了をコンマで区切ります。

終了時刻は、終了時刻のタイムスタンプのミリ秒までが対象となります。たとえば、'11:00:00 - 11:59:59'と指定した場合は、「11:00:00:000」から「11:59:59:999」までを対象とします。

pgauditのruleセクション内の評価で使用されたタイムスタンプは、ログエントリで発行されたタイムスタンプとは異なります。ログエントリは、pgauditの評価後に出力され、その時点でタイムスタンプが生成されるためです。

class

classパラメータで指定するクラスの意味は、下記のとおりです。

- READ: SELECT、COPY FROM
- WRITE: INSERT、UPDATE、DELETE、TRUNCATE、COPY TO
- FUNCTION: 関数呼び出し、DO
- ROLE: GRANT、REVOKE、CREATE ROLE、ALTER ROLE、DROP ROLE
- DDL: ROLEクラスのDDL以外のすべてのDDL (CREATE...、ALTER...など)
- CONNECT: 接続に関するイベント(要求、認証、切断)
- SYSTEM: インスタンスの起動、プライマリサーバへの昇格
- BACKUP: pg_basebackup
- ERROR: エラーで終了したイベント(PostgreSQLのエラーコードが00以外)。postgresql.confのlog_min_messagesパラメータがERROR以下に設定されている場合に使用可能。
- MISC: その他のコマンド (DISCARD、FETCH、CHECKPOINT、VACUUMなど)

ruleセクションの評価

- ログイベントが発生すると、ruleセクション内のすべての式が一度に評価されます。ログエントリは、ruleセクション内のすべてのパラメータがtrue(真)と評価された場合にのみ出力されます。

  たとえば、以下のルールが設定されていた場合、'apserver'が'myschema.tbl1'に対して行った操作のうち、午前10時から午前11時の間のクラス'WRITE'以外に対応する操作が監査ログとして出力されます。

  ```
  [rule]
  timestamp = '10:00:00-11:00:00'
  remote_host = 'apserver'
  object_name = 'myschema.tbl1'
  class != 'WRITE'
  ```

- pgaudit設定ファイルでは、ruleセクションを複数定義することができます。ログイベントは、それぞれのruleセクションによって評価され、ruleセクションが一致するごとに監査ログが出力されます。

  たとえば、以下のルールが設定されていた場合、重複した監査ログが出力されます。

  ```
  [rule]
  object_name = 'myschema.tbl1'
  [rule]
  object_name = 'myschema.tbl1'
  ```

- 1つのruleセクション内に同一のパラメータが複数指定されていた場合、最後に指定されているパラメータが有効になります。

  たとえば、以下のルールが設定されていた場合、“object_name = 'myschema.tbl3'”が有効になります。

  ```
  [rule]
  object_name = 'myschema.tbl1'
  object_name = 'myschema.tbl2'
  object_name = 'myschema.tbl3'
  ```

出力形式

Session Audit Loggingでは、監査ログが以下の形式で出力されます。

```
AUDIT: SESSION,READ,2022-03-12 19:00:58 PDT,
   (1)          (2)      (3)
[local],19944,psql,appuser,postgres,2/8, 2, 1,SELECT,,TABLE,myschema.account, ,
  (4)    (5)   (6)   (7)      (8)    (9)(10)(11)(12)(13)(14)     (15)         (16)
SELECT * FROM myschema.account;,<not logged>
       (17)                        (18)
```

No	出力内容
(1)	ログのヘッダー。 “AUDIT: SESSION”で固定です。
(2)	クラス
(3)	SQLの開始時間
(4)	リモートホスト名。 ローカルホストの場合は、[local]が出力されます。
(5)	バックエンドプロセスID
(6)	アプリケーション名。 アプリケーション名が指定されていない場合は、[unknown]が出力されます。
(7)	ユーザー名

No	出力内容
(8)	データベース名
(9)	仮想トランザクションID
(10)	ステートメントID
(11)	サブステートメントID
(12)	コマンドタグ
(13)	SQLSTATE
(14)	オブジェクトタイプ
(15)	オブジェクト名
(16)	エラーメッセージ
(17)	SQL。 CREATE ROLEなどSQLにパスワードが含まれている場合は、<REDACTED>に置換えられます。 また、pgaudit設定ファイルのoptionセクションのlog_statement_onceパラメータに“on”を指定している場合、2回目以降は<previously logged>が出力されます。
(18)	pgaudit設定ファイルのoptionセクションのlog_parameterパラメータの値により、以下の出力内容となります。 ・ log_parameter=onを指定 SQLにパラメータを指定している場合は、パラメータの値が空白区切りで連結されて出力されます。 SQLにパラメータを指定していない場合は、<none>が出力されます。 ・ log_parameter=off(デフォルト)を指定 <not logged>が出力されます。 また、pgaudit設定ファイルのoptionセクションのlog_statement_onceパラメータに“on”を指定している場合、2回目以降は<previously logged>が出力されます。

参考

以下の機能を利用した資源にアクセスする場合、(12)のコマンドタグが???として出力される場合があります。

- INSTEAD OFトリガ
- RULE
- VIEW
- 行単位のセキュリティポリシー
- テーブルの継承

使用例

Session Audit Loggingで、監査ログを取得する例を示します。

1.設定

pgaudit設定ファイルに、以下のruleセクションを指定します。

```
[rule]
class = 'READ, WRITE'
object_name = 'myschema.account'
```

2.ログを取得

以下のSQLをクライアントから実行します。

```
CREATE TABLE myschema.account
(
    id int,
    name text,
    password text,
    description text
);
INSERT INTO myschema.account (id, name, password, description) VALUES (1, 'user1', 'HASH1', 'blah, blah');
SELECT * FROM myschema.account;
```

以下の監査ログが取得できます。

classパラメータに'DDL'が定義されていないため、CREATE TABLEは監査ログとして出力されません。

```
AUDIT: SESSION,WRITE,2022-03-12 19:00:49 PDT,[local],19944,psql,appuser,postgres,
2/7,1,1,INSERT,,TABLE,myschema.account,,"INSERT INTO myschema.account (id, name, password, description) VALUES (1,
'user1', 'HASH1', 'blah, blah');",<not logged>
AUDIT: SESSION,READ,2022-03-12 19:00:58 PDT,[local],19944,psql,appuser,postgres,
2/8,2,1,SELECT,,TABLE,myschema.account,,SELECT * FROM myschema.account;,<not logged>
```

6.5 Object Audit Logging

Object Audit Loggingでは、ロールを使用することで、監査ログの取得を実現しています。

optionセクションのroleパラメータで、監査ログ取得のためのロールを指定します。ロールに実行されたコマンドの権限がある場合、また、別のロールから権限が継承されている場合、そのコマンドの操作を監査ログとして出力します。

たとえば、optionセクションのroleパラメータに“auditor”を設定後、accountテーブル対するSELECT権限とDELETE権限を“auditor”に付与します。この場合、accountテーブルに対し、SELECTまたはDELETEが実行されると、監査ログが出力されます。

出力形式

Object Audit Loggingでは、監査ログが以下の形式で出力されます。

```
AUDIT: OBJECT,1,1,READ,SELECT,TABLE,public.account,SELECT password FROM account;,<not logged>
     (1)     (2)(3)(4)  (5)     (6)         (7)                (8)                    (9)
```

No	出力内容
(1)	ログのヘッダー。 “AUDIT: OBJECT”で固定です。
(2)	ステートメントID
(3)	サブステートメントID
(4)	クラス名
(5)	コマンドタグ
(6)	オブジェクトタイプ
(7)	オブジェクト名
(8)	SQL。 pgaudit設定ファイルのoptionセクションのlog_statement_onceパラメータに“on”を指定している場合、2回目以降は<previously logged>が出力されます。

No	出力内容
(9)	pgaudit設定ファイルのoptionセクションのlog_parameterパラメータの値により、以下の出力内容となります。 ・ log_parameter=onを指定 SQLにパラメータを指定している場合は、パラメータの値がカンマ区切りで連結されて出力されます。 SQLにパラメータを指定していない場合は、<none>が出力されます。 ・ log_parameter=off(デフォルト)を指定 <not logged>が出力されます。 また、pgaudit設定ファイルのoptionセクションのlog_statement_onceパラメータに“on”を指定している場合、2回目以降は<previously logged>が出力されます。

参考

以下の機能を利用した資源にアクセスする場合、(5)のコマンドタグが???として出力される場合があります。

- INSTEAD OFトリガ
- RULE
- VIEW
- 行単位のセキュリティポリシー
- テーブルの継承

例

Object Audit Loggingでログを取得する例を示します。

ロールに対し、権限を付与する対象を細かく設定することで、ログの出力を制御することができます。

以下の例では、accountテーブルのログ取得は、列に付与された権限で制御されますが、account_role_mapテーブルのログ取得は、テーブルに付与された権限に対して制御されます。

1.設定

pgaudit設定ファイルのoptionセクションに、以下のroleパラメータを指定します。

```
[option]
role = 'auditor'
```

2.ロールの定義

Object Audit Loggingのためのロールを定義します。

```
CREATE USER auditor NOSUPERUSER LOGIN;
```

3.ログを取得

以下のSQLをクライアントから実行します。

```
CREATE TABLE account
(
    id int,
    name text,
```

```
    password text,
    description text
);
GRANT SELECT (password) ON public.account TO auditor;
SELECT id, name FROM account;
SELECT password FROM account;
GRANT UPDATE (name, password) ON public.account TO auditor;
UPDATE account SET description = 'yada, yada';
UPDATE account SET password = 'HASH2';
CREATE TABLE account_role_map
(
    account_id int,
    role_id int
);
GRANT SELECT ON public.account_role_map TO auditor;
SELECT account.password, account_role_map.role_id
  FROM account
  INNER JOIN account_role_map ON account.id = account_role_map.account_id;
```

以下の監査ログが取得できます。

acoountテーブルでは、権限が付与されている列に対する操作のみがログとして出力されます。

account_role_mapテーブルでは、テーブルに対して権限が付与されているため、テーブルに対する操作がログとして出力されます。

```
AUDIT: OBJECT,4,1,READ,SELECT,TABLE,public.account,SELECT password FROM account;,<not logged>
AUDIT: OBJECT,7,1,WRITE,UPDATE,TABLE,public.account,UPDATE account SET password = 'HASH2';,<not logged>
AUDIT: OBJECT,10,1,READ,SELECT,TABLE,public.account,"SELECT account.password, account_role_map.role_id
        FROM account
        INNER JOIN account_role_map ON account.id = account_role_map.account_id;",<not logged>
AUDIT: OBJECT,10,1,READ,SELECT,TABLE,public.account_role_map,"SELECT account.password, account_role_map.role_id
        FROM account
        INNER JOIN account_role_map ON account.id = account_role_map.account_id;",<not logged>
```

6.6 データベース多重化運用の場合

データベース多重化運用時の監査ログ取得について説明します。

6.6.1 セットアップ

構築済みのデータベース多重化環境に監査ログ機能をセットアップする場合、以下の手順でセットアップしてください。

1. pgauditのファイルコピー

   プライマリサーバおよびスタンバイサーバでpgauditのファイルをコピーします。

   pgauditのファイルのコピーについては、“6.2 セットアップ”の手順1を参照してください。

2. pgaudit設定ファイルの作成

   プライマリサーバでpgaudit設定ファイルを作成します。また、作成したpgaudit設定ファイルをスタンバイサーバにコピーします。

   pgaudit設定ファイルの作成については、“6.2 セットアップ”の手順2を参照してください。

3. postgresql.confの設定

   プライマリサーバおよびスタンバイサーバのpostgresql.confに、監査ログを利用するためのパラメータを設定します。パラメータには、同じ値を設定してください。

   設定するパラメータについては、“6.2 セットアップ”の手順3および“6.6.2 監査ログ取得の設定”を参照してください。

4. Mirroring Controllerのサーバ識別子.confファイルの設定

   プライマリサーバおよびスタンバイサーバのサーバ識別子.confファイルに、監査ログを利用するためのパラメータを設定します。

設定するパラメータについては、“6.6.2 監査ログ取得の設定”を参照してください。

5. インスタンスの起動

   プライマリサーバおよびスタンバイサーバのインスタンスを起動します。

6. pgauditエクステンションの作成

   プライマリサーバでCREATE EXTENSIONを使用し、pgauditエクステンションを作成します。

   pgauditエクステンションの作成については、“6.2 セットアップ”の手順5を参照してください。

7. pgaudit設定ファイルのパラメータ設定

   プライマリサーバでpgaudit設定ファイルのパラメータを追加/変更します。追加/変更したpgaudit設定ファイルは、スタンバイサーバにコピーします。

   設定するパラメータについては、“6.3 pgaudit設定ファイル”および“6.6.2 監査ログ取得の設定”を参照してください。

8. インスタンスの再起動

   プライマリサーバおよびスタンバイサーバのインスタンスを再起動します。

6.6.2 監査ログ取得の設定

データベース多重化運用では、多重化の状態確認や障害検知のため、Mirroring Controllerが定期的にデータベースにアクセスします。これにより、監査ログも定期的に取得され、ログファイルが枯渇してしまいます。このため、以下のパラメータを設定して、Mirroring Controllerによる監査ログを取得しないようにします。

postgresql.conf

log_connections

省略または“off”を指定します。

log_disconnections

省略または“off”を指定します。

Mirroring Controllerのサーバ識別子.confファイル

target_db

template1を指定します。

注意

新規にデータベースを作成する場合は、Mirroring Controllerを停止してから作成するか、テンプレートデータベースにtemplate1以外を指定して作成してください。

pgaudit設定ファイル

ruleセクションのdatabase

database != 'template1'を指定します。

6.7 SQLでの監査ログの参照

監査ログは、追加モジュールのfile_fdwを使用することで、SQLでアクセスすることができます。ここでは、専用ログファイルに出力したSession Audit Loggingを例に監査ログを参照する方法を説明します。

1. file_fdwのインストール

   CREATE EXTENSIONを使用し、file_fdwをエクステンションとしてインストールします。

```
$ psql
=# CREATE EXTENSION file_fdw;
=# ¥dx
                  List of installed extensions
```

```
Name     | Version |   Schema   |     Description
---------+---------+------------+-------------------------------------------
file_fdw | 1.0     | public     | foreign-data wrapper for flat file access
pgaudit  | 1.0     | public     | provides auditing functionality
plpgsql  | 1.0     | pg_catalog | PL/pgSQL procedural language
(3 rows)
```

2. 外部サーバの作成

   CREATE SERVERを使用し、file_fdwで管理する外部サーバを作成します。

```
$ psql
=# CREATE SERVER auditlog FOREIGN DATA WRAPPER file_fdw;
```

3. 監査ログのテーブルの作成

   CREATE FOREIGN TABLEを使用し、監査ログのテーブルのカラム、CSVファイル名とフォーマットを定義します。

```
$ psql
=# CREATE FOREIGN TABLE auditlog (
header text,
class text,
sql_start_time timestamp with time zone,
remote_host_name text,
backend_process_id text,
application_name text,
session_user_name text,
database_name text,
virtual_transaction_id text,
statement_id text,
substatement_id text,
command_tag text,
sqlstate text,
object_type text,
object_name text,
error_message text,
sql text,
parameter text
) SERVER auditlog
OPTIONS ( filename '/database/inst1/pgaudit_log/pgaudit-2022-03-12.log', format 'csv' );
```

注意

監査ログファイルをローテーションさせて複数の監査ログファイルが存在する場合は、それぞれの監査ログファイルに対してテーブルを作成する必要があります。

4. 監査ログの参照

   SELECTを使用し、監査ログを参照します。

```
$ psql
=# SELECT * FROM auditlog;
     header      |  class  |     sql_start_time     | remote_host_name | backend_process_id ...
-----------------+---------+------------------------+------------------+-------------------- ...
AUDIT: SESSION   | WRITE   | 2022-03-12 19:00:49+09 | [local]          | 19944              ...
AUDIT: SESSION   | READ    | 2022-03-12 19:00:58+09 | [local]          | 19944              ...
```

6.8 アンセットアップ

pgauditのアンセットアップ方法を説明します。

1. インスタンスの起動

   インスタンスを起動します。

2. pgauditエクステンションの削除

   DROP EXTENSION文を使用して、データベースからpgauditエクステンションを削除します。

```
$ psql -d <database name>
=# DROP EXTENSION pgaudit;
=# ¥q
```

3. postgresql.confの変更

   pgauditに関する以下のパラメータを削除します。

   - shared_preload_libraries
   - pgaudit.config_file

4. インスタンスの再起動

   インスタンスを再起動します。

5. pgauditのファイルの削除

   スーパーユーザーで以下のコマンドを実行します。“<*x*>”は、製品のバージョンを示します。

```
$ su -
Password:******
# rm -rf /opt/fsepv<x>server64/セットアップ時にコピーしたファイル
```

参考

セットアップ時にコピーしたファイルは以下で確認することができます。

```
# find /opt/fsepv<x>server64/OSS/pgaudit
```