



PostgreSQL安定運用のための 障害予防と検知

篠田典良 / 日本ヒューレット・パッカード株式会社 / 2014年12月12日

Open Source Conference 2014 .Enterprise / Room 5B

自己紹介

篠田典良(しのだのりよし)

- 所属
 - 日本ヒューレット・パッカード株式会社 テクノロジーコンサルティング事業統括
- 現在の業務
 - PostgreSQLをはじめ Oracle Database, Microsoft SQL Server, Vertica, Sybase ASE 等 RDBMS 全般に関するシステムの設計、チューニング、コンサルティング
 - オープンソース製品に関する調査、検証
 - Oracle Database 関連書籍の執筆
 - 弊社講習「Oracle Database エンジニアのための PostgreSQL 入門」講師



- 関連する URL

- HP Open Services

- PostgreSQL Internals
- PostgreSQL 9.4 新機能検証報告レポート

<http://h50146.www5.hp.com/services/ci/opensource/>

- Oracle ACE ってどんな人？

<http://www.oracle.com/technetwork/jp/database/articles/vivadeveloper/index-1838335-ja.html>



Agenda

PostgreSQL安定運用のための障害予防と検知

1. リソース不足の予防と推奨設定
2. ログファイルの設定と定期的にチェックするカタログ
3. 検知すべきログ
4. まとめ

スライド内で使用される PostgreSQL および OS のバージョンは以下の通り

- PostgreSQL 9.4 RC1 (標準オプションでビルド)
- Red Hat Enterprise Linux 6 Update 5 (x86-64)



1. リソース不足の予防と推奨設定



1.1 リソース不足の原因

OS か？ データベースか？

- 障害が発生するリソース不足は何等かのパラメータ値が不足している
 - オペレーティング・システムのカーネル・パラメータやユーザー・リソース制限
 - PostgreSQL のリソース制限パラメータの設定
 - インスタンス起動時に確保されるリソースと接続数や実行数に応じて確保されるリソースがある
- オペレーティング・システムのリソース
 - カーネル・パラメータの不足
 - プロセス数、オープンできるファイル数、CORE ファイルのサイズ制限
 - 仮想メモリーや共有メモリー容量の不足
- PostgreSQL のリソース制限パラメータ
 - 最大同時接続ユーザー数
 - 最大スレーブ・インスタンス数
 - 最大自動起動プロセス数



1.2 プロセスに関するリソース

postmasterを親プロセスとする複数のプロセスから構成

主なプロセス	起動数	用途	備考
postgres	1	親プロセス、接続待ち	postmasterとも呼ばれる
wal writer	0~1	一部のWAL書き込み	
writer	1	セグメント書き込み	テーブルやインデックス
logger	0~1	ログファイル書き込み	
checkpointer	1	チェックポイント実行	
autovacuum launcher	0~1	VACUUM 処理開始	
autovacuum worker	0~複数	VACUUM 処理実行	
postgres	接続数	クライアントSQL実行	
wal sender	接続数	スレーブからの接続	レプリケーション・マスター
wal receiver	0~1	マスターへの接続	レプリケーション・スレーブ
archiver	0~1	アーカイブログの出力	
stats collector	1	統計情報の更新	
bgworker	複数	カスタム・ワーカー	



1.2 プロセスに関するリソース

接続数とプロセス数

クライアントの接続数分のプロセスが起動される

- 起動数が変動するプロセス

プロセス	起動のタイミング	最大数を決めるパラメータ
postgres	クライアントの接続	max_connections
wal sender	レプリケーション・スレーブの接続 pg_basebackup コマンド実行時	max_wal_senders
autovacuum worker	自動 VACUUM 実行時	autovacuum_max_workers
bgworker	設定に依存	max_worker_processes

- ユーザーのプロセス制限を考慮(ファイル /etc/security/limits.conf)

postgres	soft	nproc	1024
postgres	hard	nproc	1024



1.3 メモリーに関するリソース

環境に応じて変更が必要なカーネル・パラメータ

- 以下の環境ではカーネル・パラメータの調整が必要になる場合がある

環境	調整が必要なカーネル・パラメータ
最大接続数が多い場合	kernel.sem
Huge Page 機能を使用する場合 (9.4～)	vm.nr_hugepages
搭載メモリーが大きい場合 (～9.2*)	kernel.shmmax

- セマフォ (kernel.sem) の計算式は以下の通り

- 最大バックエンド数 = $\text{max_connections} + \text{autovacuum_max_workers} + \text{max_worker_processes} + 1$
- 最大セマフォ数 = 最大バックエンド数 + 4
- セマフォ集合数 = $(\text{最大セマフォ数} + 15) / 16$

- 上記カーネル・パラメータ不足の場合は、インスタンス起動時にエラーになる
- パラメータ huge_pages = on (デフォルト値) の場合、Huge Pages 領域が確保できない場合は通常のページ・サイズでメモリーを確保する

* PostgreSQL 9.3 以降、メモリー構造が大きく変更されたため shmmax の考慮は不要



1.3 メモリーに関するリソース

共有メモリー領域

- 共有バッファは読み込んだデータのキャッシュとして使用される領域
 - パラメータ `shared_buffers` で指定
 - デフォルト値は 128 MB であり、小さすぎる(旧バージョンのデフォルト値は更に小さい)
 - 一般的にはチューニングにより物理メモリーの 20~40% を指定
- WAL バッファはトランザクション情報を一時的に保存する領域
 - パラメータ `wal_buffers` で変更
 - 大きくても数 MB~ 32 MB程度
 - デフォルトは -1 で、自動設定
 - 自動設定時はパラメータ `shared_buffers / 32` が指定
 - 自動設定時の下限は 64 KB、上限は 16 MB



1.3 メモリーに関するリソース

ヒープ・メモリー

プロセス毎に確保される仮想メモリー

- 稼働されるプロセス数と物理メモリー容量を考慮して設定
旧バージョンのデフォルト値はより小さい値
- 以下のパラメータを指定可能、最適値はチューニングによって決定

パラメータ	説明	デフォルト
temp_buffers	一時テーブル用バッファ	8 MB
work_mem	ソート処理、ハッシュ処理用バッファ	4 MB
maintenance_work_mem	VACUUM, インデックス作成等の処理用	64 MB
autovacuum_work_mem	自動VACUUM 専用(9.4~)	-1

- autovacuum_work_mem のデフォルト値(-1) は maintenance_work_mem を使う設定

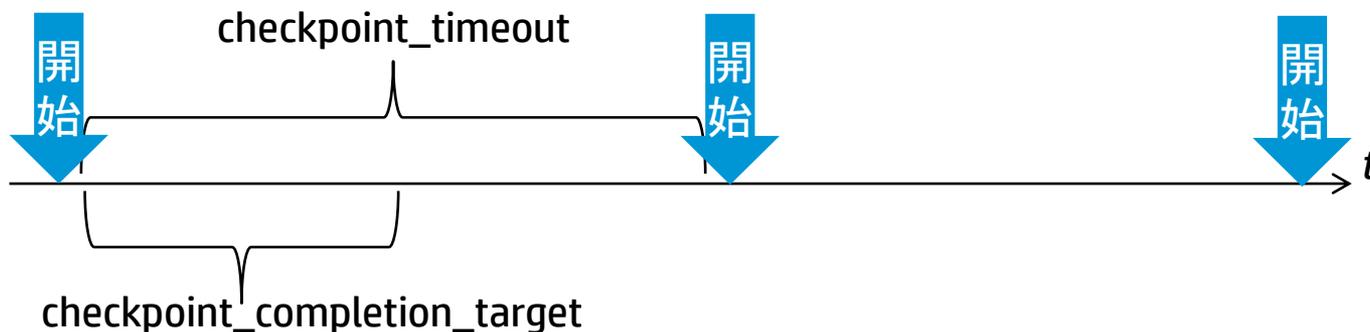
1.4 I/O に関するリソース

チェックポイント

メモリーとストレージの同期

- checkpointer プロセスが実行
- 変更されたメモリー上の情報とテーブルやインデックスを構成するファイルを同期
- 時間ベースのチェックポイント (パラメータ `checkpoint_timeout`)
- WAL ファイル書き込み量によるチェックポイント (パラメータ `checkpoint_segments`)
- チェックポイントのチューニング

チェックポイント間隔	I/O 量	リカバリ時間
間隔を短く	増加	短時間
間隔を長く	減少	長時間



1.4 I/O に関するリソース

チェックポイント

パラメータの推奨値

パラメータ	用途	推奨値(デフォルト)
checkpoint_segments	チェックポイントを開始するWALファイル書き込み数	128 (3)
checkpoint_timeout	チェックポイント時間間隔	30min (5min)
checkpoint_warning	頻繁なチェックポイントを警告する間隔	30min (30s)
checkpoint_completion_target	チェックポイント完了までの時間割合	0.9 (0.5)
log_checkpoints	チェックポイント情報のログ出力	on (off)

checkpoint_warning 期間内に WAL ファイルベースのチェックポイントに到達すると以下のログ出力

LOG: **checkpoints are occurring too frequently** (2 seconds apart)

HINT: Consider increasing the configuration parameter "checkpoint_segments".

1.4 I/O に関するリソース

VACUUMとは

不要レコードの削除処理

- PostgreSQLは追記型アーキテクチャ
 - UPDATE 文では更新済レコードを追記
 - DELETE文では物理削除を行わない
- 不要レコードを削除し、ページに空き領域を作る処理
- 不要レコード数は、pg_stat_{all | user | sys}_tables カタログの n_dead_tup 列から確認可能
- 実行契機
 - 自動VACUUM
 - VACUUM 文または VACUUM FULL文実行時(テーブル単位、データベース単位)
- VACUUM CONCURRENT
 - ページ内の不要レコードを切り詰めて空き領域を作成
 - 標準設定では自動的に実行
 - ファイル末尾の空きブロックを切り詰め
- VACUUM FULL
 - ブロックをまたがって切り詰めを行うため、セグメント・ファイルを縮小
 - 通常は実行不要



1.4 I/O に関するリソース

VACUUM の自動実行

デフォルトで自動的に行われる

- 統計情報の再計算

- 以下の計算式以上のレコードが更新 (UPDATE/DELETE) されたテーブルは自動的に 実行

$\text{autovacuum_vacuum_threshold} + \text{autovacuum_vacuum_scale_factor} * \text{レコード数}$

- パラメータはテーブル単位でも決定可能

- 大規模テーブルではテーブル単位に `autovacuum_vacuum_scale_factor` を 0 に、`autovacuum_vacuum_threshold` を適切な値に変更し、頻繁に VACUUM を実行することを推奨

1.4 I/O に関するリソース

統計情報の取得

SQL 文の実行計画を決定するためのデータ

- SQL文の実行計画作成には統計情報を使用
- 統計情報とはテーブルやインデックスのデータ情報
 - レコード数、ブロック数
 - ヒストグラム(データの最頻値、ばらつき)
- 取得契機
 - 自動 VACUUM
 - VACUUM ANALYZE文の実行
 - ANALYZE 文の実行
- 統計情報の確認
 - pg_statistic カタログ
 - pg_stats カタログ (pg_statistic カタログを見やすく整形している)



1.5 SQL 実行のためのリソース

統計情報の取得

SQL 文の実行計画を決定するためのデータ

- 統計情報はサンプリングによって行われる
 - サンプリング量はデフォルト 30,000 レコード (MAX(列の STATISTICS 値) × 300 レコード)
 - 列の STATISTICS 値とは？
 - ヒストグラムを収集するバケット数(データの範囲を入れる箱の数)
 - デフォルト値はパラメータ default_statistics_target で決まる(デフォルト 100)
 - 「ALTER TABLE table_name ALTER column_name **SET STATISTICS** value」文で変更
- 大規模なテーブルではサンプリング・レコード数を拡大するため、STATISTICS 値の拡大を推奨
 - デフォルトの統計情報計算式はテーブルのレコード数を 1,000,000 レコードと想定。
10,000,000 レコードのテーブルでは STATISTICS = 114 程度にすると必要なサンプリングが行われる

1.6 ネットワークに関するリソース

パラメータと pg_hba.conf ファイル

- postmaster プロセスがクライアントからの接続を待つ
- 接続待ちのポート番号は、パラメータ port で決定される(デフォルト 5432)
 - デフォルトを使用する場合でもパラメータ・ファイルに記述することを推奨
 - パラメータ・ファイルに記述が無い場合、環境変数 PGPORT が使用される
- クライアントの切断を検知するために Keep Alive 関連パラメータを設定(デフォルトは OS 設定を継承)

パラメータ	説明	デフォルト
tcp_keepalives_idle	アイドル・セッションにKeep Alive パケットを送信する時間間隔(秒)	0
tcp_keepalives_interval	Keep Alive パケットの再送間隔(秒)	0
tcp_keepalives_count	Keep Alive パケットの再送上限(回数)	0



1.7 CORE ファイル設定

トラブル解析のためにCOREファイルは重要

- Red Hat Enterprise Linux 6 の CORE ファイル出力設定
 - Red Hat Enterprise Linux 6 では 標準で ABRT 機能が有効になっている
 - 標準ではサイン済のバイナリのみ CORE を出力

/etc/abrt/abrt-action-save-package-data.conf ファイル

```
OpenPGPCheck = no
```

- CORE ファイルサイズの制限解除

/etc/security/limits.conf ファイル

```
postgres - core unlimited
```

\${HOME}/.bashrc ファイル

```
ulimit -c unlimited
```



2. ログファイルの設定と定期的に チェックするカタログ

2.1 ログファイルの設定

ログの種類

- サーバーログ
 - パラメータ logging_collector を on に設定することで取得 (デフォルト off)
 - SYSLOG (Windows 環境ではイベントログ) またはテキストファイルを選択可能
 - 標準では pg_log ディレクトリに作成される。パラメータ log_directory で指定 (データベース・クラスタからの相対、または絶対パス)
 - OSクラッシュ時には保存されていない場合がある (書き込み時に flush していないため)
 - ログのエンコードはデータベースと同じ
- pg_ctl コマンドのログ
 - インスタンスの起動 / 停止等のエラー解析に使用
 - ログファイルを指定しない場合は標準エラー (stderr) に出力
 - -l パラメータでファイル名を指定
 - 既にファイルが存在する場合は追記
 - インスタンス起動時にログが書けない場合は起動エラー



2.1 ログファイルの設定

ログのレベル

下記表の上位ほど影響が大きい

- どのレベルからログに出力するかは、パラメータ `log_min_messages` で指定
- どのレベルからログにSQLを出力するかは、パラメータ `log_min_error_statement` で指定
- どのレベルからクライアントに送信するかは、パラメータ `client_min_messages` で指定
- デフォルト設定を推奨

レベル	用途	備考
PANIC	全セッションが切断される	
FATAL	カレントセッションが切断される	
LOG	管理者向けメッセージ	
ERROR	現在のコマンドが中断される	<code>log_min_error_statement</code> デフォルト
WARNING	ユーザーへの警告	<code>log_min_messages</code> デフォルト
NOTICE	ユーザーへの補助情報	<code>client_min_messages</code> デフォルト
INFO	ユーザーからの要求による情報	
DEBUG1...5	開発者向けのメッセージ	



2.1 ログファイルの設定

ログファイルのフォーマット

設定が必要なパラメータ

- ログには「レベル」と「説明文字列」が出力される
- ログの先頭にはパラメータ `log_line_prefix` で指定された文字列が出力される
 - デフォルトは空文字
 - 少なくとも以下のパラメータを指定することを推奨

パラメータ指定	説明	備考
<code>%m</code>	タイムスタンプ	タイムゾーンはパラメータ <code>log_timezone</code>
<code>%a</code>	アプリケーション名	
<code>%u</code>	データベース・ユーザー名	
<code>%d</code>	データベース名	
<code>%r</code>	リモートホスト名 : ポート番号	ローカル接続の場合は <code>[local]</code>

- パラメータ `log_line_prefix` の最後には空白を含める
 - 例 `log_line_prefix = '%m %a %u %d %r '` 

2.1 ログファイルの設定

ログファイルの出力先

設定が必要なパラメータ

- 出力フォーマットはパラメータ `log_destination` で指定
 - 設定可能な値は `csv`, `syslog`, `stderr` (デフォルト `stderr`)
 - `syslog` を選択した場合にはパラメータ `syslog_facility` (デフォルト `local0`)、`syslog_ident` (デフォルト `postgres`) も指定
- ファイル名はパラメータ `log_filename` で指定する
(デフォルト `postgresql-%Y-%m-%d_%H%M%S.log`)
 - ローテーションを考慮してファイル名を決定
 - 主な指定パラメータ

パラメータ	意味
%Y	4桁西暦
%y	2桁西暦
%m	月番号(01~12)
%d	月内の日付(01~31)
%A	曜日名(英語)

パラメータ	意味
%H	時間(00~23)
%M	分(00~59)
%S	秒(00~59)
%h	時間(00~11)
%p	AM / PM



2.1 ログファイルの設定

ローテーション

- サイズおよび時間でローテーション可能
- サイズでローテーションする場合には、原則としてファイル名に時刻が必要
- ローテーションしようとした場合に、新ファイルが同じ名前になるとローテーションしない
- ログファイルの監視ツールがサポートしていることを確認する
- 出力中のログファイルを消すと再作成されない
 - logger プロセスにSIGUSR1 シグナルを送信して強制ローテーションさせることで再作成
- 関連するパラメータ

パラメータ	説明	デフォルト
log_truncate_on_rotation	ローテーション時にファイル切り詰め	off
log_rotation_age	ローテーション時間を指定	1d
log_rotation_size	ローテーションサイズを指定	10MB



2.1 ログファイル設定

パフォーマンス低下に関係するログの出力

設定を推奨するパラメータ

パラメータ	説明	推奨値(デフォルト)
log_min_duration_statement	実行時間が長い SQL 文情報	システム依存 (-1)
log_checkpoints	チェックポイント情報	on (off)
log_autovacuum_min_duration	実行時間が長い自動 Vacuum 情報	システム依存 (-1)
log_temp_files	外部ソート情報、work_mem で指定されたメモリ不足の可能性	0 (-1)
log_lock_waits	長時間ロック待ちセッション情報	on (off)



2.3 パフォーマンス監視カタログ

パフォーマンス情報の定期チェックが必要

pg_stat* カタログのチェック

- 正常稼働時の状態を知っておくことが重要

システムカタログ	確認できる内容	備考
pg_stat_activity	セッション数、実行中のSQL文	COUNT(*) で取得
pg_stat_bgwriter	チェックポイント情報	
pg_stat_archiver	アーカイブ情報(失敗を検知)	
pg_stat_database	データベース全体の状況、 一時領域(ディスクソート)の使用状況	パラメータ work_mem を検討
pg_stat_replication	レプリケーション情報	
pg_stat_*_tables	VACUUM 情報、更新レコード情報	
pg_statio_*_tables	I/O 方法(インデックス検索、全権検索)	
pg_stat_*_indexes	インデックス使用状況、アクセス状況	インデックスが有効か
pg_statio_*_indexes	インデックス I/O 状況	
pg_stat_user_functions	ファンクション実行状況	



2.4 リモート・インスタンスの監視

pg_isready コマンド

- pg_isready コマンドを実行すると、リモート・インスタンスの稼働をチェックできる(9.3～)
- 以下のパラメータが指定できる。データベース名 -d, ユーザー名 -U は無視される

パラメータ	説明	デフォルト
-h ホスト名	接続先ホスト名	localhost
-p ポート番号	接続先ポート番号	5432
-t タイムアウト	接続待ちタイムアウト(秒)	3
-q	画面表示を抑制	-

戻り値	説明	備考
0	インスタンス稼働中で接続可能	
1	インスタンス稼働中だが接続不可	
2	サーバーと通信不可	
3	パラメータ不正	

3. 検知すべきログ



3.1 ログの監視

PANIC エラーと FATAL エラー

- PANIC エラーが発生するとインスタンス停止
 - アプリケーションがエラーになるため検知は容易
 - 自動的な再起動や切り換えにはクラスタリング・ウェアや pgpool-II 等によるインスタンスの切り替えが必要
- FATAL エラーのログの多くは認証エラーだが、接続数制限超過の場合も発生する。
 - 一般ユーザーが利用できるのは `max_connections (100) - superuser_reserved_connections (3)` まで
 - 以下のメッセージが出力される
 - 管理者以外の接続数が超過した場合

FATAL: remaining connection slots are reserved for non-replication superuser connections

- 全て接続数を超過した場合

FATAL: sorry, too many clients already

3.1 ログの監視

どのようなエラーを通知すべきか？

- PANIC と FATAL の監視だけでは不十分
- レベルは LOG や WARNING でも対処が必要な障害が多数ある



3.2 プロセス障害のログ

バックエンド・プロセスの異常終了

- postmaster プロセス障害
 - インスタンス全体が停止
 - 実行中のトランザクションは次回起動時にロールバック
 - 再起動等はクラスタリング・ウェア、pgpool-II 等の役割
- postgres プロセス障害
 - プロセスの特定は pg_stat_activity カタログからプロセス ID を決定
 - SIGKILL シグナルで停止された場合、全セッションはリセットされ、障害発生直後のSQL文はエラー
 - 実行中の全トランザクションはロールバック
 - 以下のログが出力される

LOG: server process (PID 3571) was terminated by signal 9: Killed

LOG: terminating any other active server processes

WARNING: terminating connection because of crash of another server process

- セッションを終了させる場合は SIGTERM シグナルで行うことを推奨

3.2 プロセス障害のログ

バックエンド・プロセスの異常終了

- デフォルトでは自動的に再起動する(パラメータ restart_after_crash = on)
 - restart_after_crash = off の場合は、全プロセス停止
- writer, wal writer, checkpointner, autovacuum launcher プロセスは全体で再起動
 - 実行中の 全トランザクションはロールバック
 - クライアントとの接続は維持されるが、再起動直後の SQL 文はエラーになる
 - アプリケーションの対処が必要
 - その他のプロセス(logger, wal sender, wal sender, wal receiver 等)は単純に再起動
- レプリケーション環境ではスレーブ・インスタンスで起動している startup process が異常終了するとスレーブ・インスタンス全体が停止
- 以下のログが出力される(wal writer の場合)

LOG: WAL writer process (PID 3929) was terminated by signal 9: Killed

LOG: terminating any other active server processes

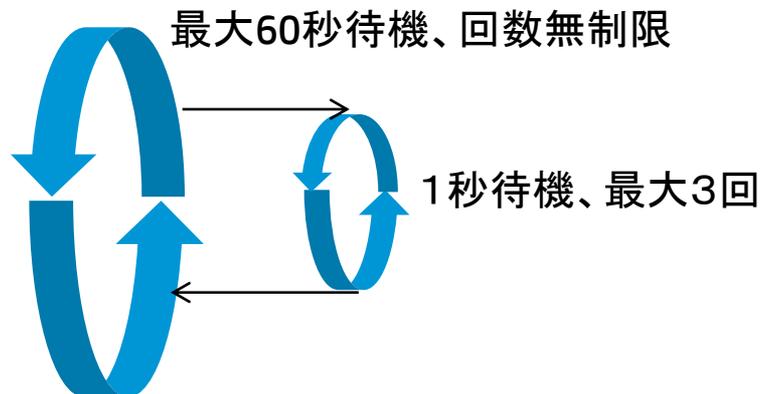
WARNING: terminating connection because of crash of another server process

3.2 プロセス障害のログ

アーカイブログの作成エラー

アーカイブログ保存先のデバイスフル等によるアーカイブログ作成エラー発生時

- パラメータ `archive_command` で指定されたコマンドが0以外の値を返すとアーカイブ失敗と見なされる
 - デバイスフル、プロセス起動エラー、パラメータ設定失敗、ディレクトリ削除等が原因
 - `pg_stat_archiver` カタログからも検知できる
 - アーカイブ化に失敗した場合、`pg_xlog` ディレクトリの WAL ファイルは再利用されない
 - このためアーカイブ化に失敗すると `pg_xlog` ディレクトリに大量の WAL ファイルが作成される
- ユーザーのトランザクションは停止しない
- 失敗したファイルについては自動的に再実行
 - 1秒間隔で、3回トライしてエラー→最大1分待つ→繰り返し



3.2 プロセス障害のログ

アーカイブログの作成エラー

ログの例

- エラー・レベルが LOG / WARNING になる場合がある

```
cp: cannot create regular file `/arch/00000001000000040000006A': Permission denied
2014-11-07 16:39:57.089 JST LOG: archive command failed with exit code 1
2014-11-07 16:39:57.089 JST DETAIL: The failed archive command was: cp
pg_xlog/00000001000000040000006A arch/00000001000000040000006A
cp: cannot create regular file `/arch/00000001000000040000006A': Permission denied
2014-11-07 16:39:58.106 JST LOG: archive command failed with exit code 1
2014-11-07 16:39:58.106 JST DETAIL: The failed archive command was: cp
pg_xlog/00000001000000040000006A arch/00000001000000040000006A
cp: cannot create regular file `arch/00000001000000040000006A': Permission denied
2014-11-07 16:39:59.122 JST LOG: archive command failed with exit code 1
2014-11-07 16:39:59.122 JST DETAIL: The failed archive command was: cp
pg_xlog/00000001000000040000006A arch/00000001000000040000006A
2014-11-07 16:39:59.123 JST WARNING: archiving transaction log file
"00000001000000040000006A" failed too many times, will try again later
```

3.3 ストレージ障害のログ

ブロック破損

- セグメントからの読み込み時にチェック
- チェックサム・エラーが検知されたテーブルは以降使用不可
 - パラメータ `ignore_checksum_failure` を on 指定することでチェックサムを無視できる(デフォルト off)
 - 対処
 - バックアップからリカバリ
 - チェックサムを無視したデータを使って再構築
- チェックサム・エラーが検知されると以下のログが出力される

```
WARNING: page verification failed, calculated checksum 1094 but expected 51919  
ERROR: invalid page in block 0 of relation base/24577/24578  
STATEMENT: select * from demo1 ;
```



3.3 ストレージ障害のログ

WAL 保存先ストレージ故障

- WAL 障害
 - トランザクション確定時にインスタンス異常終了
 - 前回のチェックポイントから最新トランザクションまでの更新情報は喪失
 - 対処
 - pg_resetxlog コマンドにより既存データを救済(-f オプションを指定)
 - ストリーミング・レプリケーションによるスレーブを昇格
- 以下のログが出力

```
2014-11-07 17:21:17.912 JST PANIC: could not open transaction log file
```

```
"pg_xlog/00000001000000040000006D": Permission denied
```

```
2014-11-07 17:21:17.918 JST LOG: startup process (PID 10750) was terminated by  
signal 6: Aborted
```

```
2014-11-07 17:21:17.918 JST LOG: aborting startup due to startup process failure
```

3.3 ストレージ障害のログ

OSクラッシュによるファイル削除

- インスタンスは正常起動
- SQL文はエラーになるが、セッションは維持
- 以下のログが出力

```
ERROR: could not open file "base/16385/24628": そのようなファイルやディレクトリはありません  
STATEMENT: select * from drop1 ;
```

- バックアップからデータファイルをリストアし、リカバリを実施することで復旧可能
- 更新トランザクションが発生しているテーブルについては、ファイルの削除が検知できない場合がある

3.3 ストレージ障害のログ

OSクラッシュによるファイル削除

削除されるファイルにより影響が異なる

主なファイル	影響	対処
pg_control	インスタンス起動不可	リストア + pg_resetxlog コマンドによるWAL再作成
WALファイル	インスタンス起動不可	pg_resetxlog コマンドによるWAL再作成
PG_VERSION	インスタンス起動不可	リストア
VM / FSM	影響なし	自動 VACUUM による生成
セグメントファイル	対象テーブルに依存	リストア + リカバリ
postgresql.conf	インスタンス起動不可	リストア
pg_hba.conf	インスタンス起動不可	リストア

まとめ



まとめ

まとめ

- 障害の検知は日々の監視の積み重ねから
- 正常なデータベースの状態を蓄積し、可視化する
- pg_stat_* カタログを定期的に検索し、チューニングの必要性を探る



Thank you

篠田 典良
テクノロジー事業統括
サービス統括本部 オープンソース部
シニアアーキテクト



Noriyoshi.Shinoda@hp.com

日本ヒューレット・パカード株式会社
本社
〒136-8711
東京都江東区大島2-2-1

