TIS
IT Holdings Group
Go Beyond

ツールをちょい足しして PostgreSQLの運用を楽にする

～ pg_monz, fluentd 他 ～

TIS株式会社
中西 剛紀

自己紹介

TIS IT Holdings Group | Go Beyond

- **氏名：中西 剛紀 (なかにし よしのり)**
- **所属：TIS株式会社　戦略技術センター**
- **主な活動領域：PostgreSQL全般**
  - 日本PostgreSQLユーザ会(JPUG)
    勉強会でたまに講演しています
    http://www.slideshare.net/naka24nori/jpug25
  - PostgreSQLエンタープライズコンソーシアム (PGECons)
- **お仕事：社内でOSSを活用する支援**

[PR]TIS OSSサポートサービス

TIS IT Holdings Group | Go Beyond

- OSSのサポートはじめました。
  - 対象OSS

[PR]TIS OSSサポートサービス

TIS IT Holdings Group | Go Beyond

- お問合せ先

TIS株式会社
OSSサポートサービス担当窓口

oss-sales@ml.tis.co.jp

AGENDA

TIS IT Holdings Group | Go Beyond

- 運用で使うPostgreSQLの標準機能
- 稼働統計情報を使った監視の実態
- Zabbixを使ったPostgreSQLの監視
- ログを使った監視の実態
- FluentdでPostgreSQLのログを収集
- FluentdでPostgreSQLのログ監視を強化

はじめに

- 本日紹介するノウハウは機能的に実現可能なことを確認済ですが、本番運用に適用する際は各自検証されることをお勧めします。(特に性能面)

はじめに

TIS IT Holdings Group | Go Beyond

- **動作確認で利用したプロダクト**
  - CentOS 6.4(x86_64)にRPMで導入

プロダクト名	バージョン
PostgreSQL	9.3.4
Zabbix	2.2.4
Fluentd(td-agent)	1.1.20

  - Fluentdプラグインをgemで導入

プラグイン名	バージョン
fluent-plugin-tail-multiline	0.1.5
fluent-plugin-rewrite-tag-filter	1.4.1
fluent-plugin-parser	0.3.4
fluent-plugin-pgjson	0.0.4

データベースの運用管理

- **データベース運用管理の目的**
  - DBの状態を把握して健全な状態に保つ
- **データベース運用管理の種類**
  - 死活監視
  - リソース監視
  - 性能分析/チューニング
  - バックアップ/リストア
- **監視が運用管理の基本のき**
  - 正しく現状を把握しなければ何もできない

監視に使うPostgreSQLの機能

TIS IT Holdings Group | Go Beyond

- 稼働統計情報
- ログ

監視に使うPostgreSQLの機能

TIS IT Holdings Group | Go Beyond

- 稼働統計情報
- ログ

稼働統計情報とは

- **稼働統計情報って何？**
  - PostgreSQLのアクティビティ情報を蓄積
  - stats_collectorというプロセスが記録
  - ユーザはビューを通して参照可能
- **稼働統計情報の取得方法**
  - SELECT文でビューにアクセスして取得

稼働統計情報とは

- **稼働統計情報で何がわかる？**
  - 代表的な稼働統計情報用のビュー

ビュー名	取得できる情報
pg_stat_activity	接続中のクライアントの処理状況 ・ 実行しているSQL ・ 接続開始、トランザクション開始、SQL開始時間 ・ プロセス状態（SQL実行中/問合せ待ち/ロック待ち）
pg_stat_database	DB単位のアクティビティ状況 ・ コミット/ロールバック数 ・ 現在の接続数 ・ 更新&参照件数 ・ deadlock数
pg_stat_user_tables	テーブル単位のアクティビティ状況 ・ シーケンシャルスキャン回数、件数 ・ 更新件数 ・ 最後の autovacuum、analyze 時間

稼働統計情報の難点

TIS IT Holdings Group | Go Beyond

- **SQLを書くのがダルい**
  - キャッシュヒット率の計算

```
SELECT  datname, round(blks_hit*100/(blks_hit+blks_read), 2)  AS cache_hit_ratio
FROM pg_stat_database WHERE blks_read > 0

datname | cache_hit_ratio
------------+-----------------
postgres | 99.00
```

- **主に累積値かつ揮発情報を保持**
  - 定期的に情報取得し、過去値との差分で判断

稼働統計情報を扱うツール

TIS IT Holdings Group | Go Beyond

- **pgperf**
  - 稼働統計情報保存用スキーマとテーブル作成
  - 任意のタイミングで情報一括取得
  - 好きなツールで蓄積した情報を自由に分析・活用することができる
- **pg_statsinfo**
  - PostgreSQL専用の監視ツール
  - 稼働統計情報を一定間隔でリポジトリに保存
  - pg_stats_reporterでグラフィカルな形での解析、出力が可能
  - 高機能な反面、導入時の設定や運用は煩雑

pg_monz（ぴーじーもんず）

- **PostgreSQL monitoring template for Zabbix**
- **ZabbixにPostgreSQLの監視機能を追加するテンプレート**
  - TIS と SRA OSS, Inc. 日本支社で共同開発
  - Apache License Version 2.0で公開

pg_monzでできること

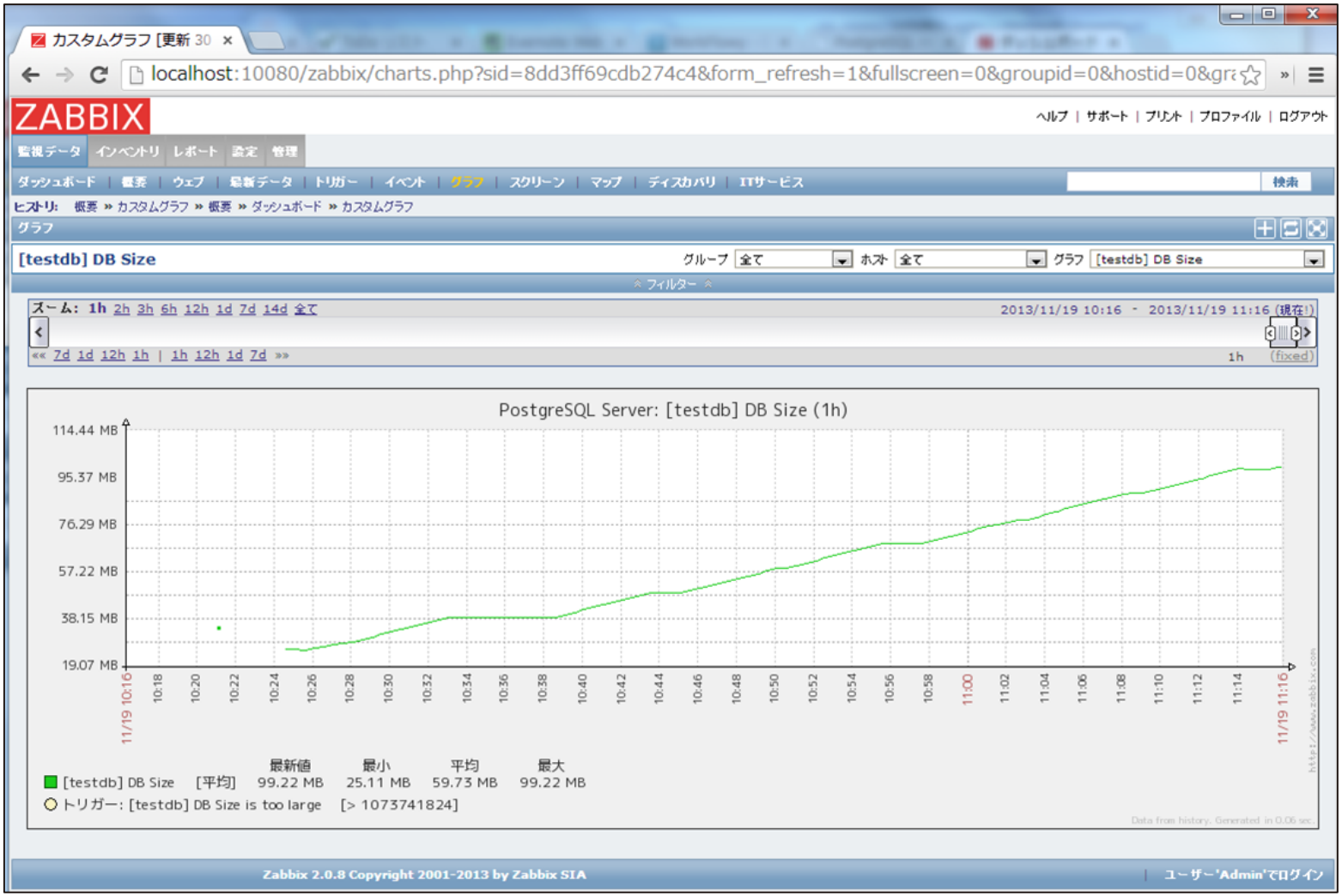
- 死活監視
- ログ監視
- データベース容量の監視
- データベース接続数の監視
- チェックポイント実行回数の監視
- データベースキャッシュヒット率の監視

pg_monzでできること

- COMMIT/ROLLBACK数の監視
- 一時ファイル発生状況の監視
- 滞留バックエンド処理の監視
- テーブル単位の情報収集（オプション）
  - VACUUM実行回数
  - キャッシュヒット率
  - シーケンシャルスキャン実行回数 etc

監視画面イメージ

- データベース容量の遷移



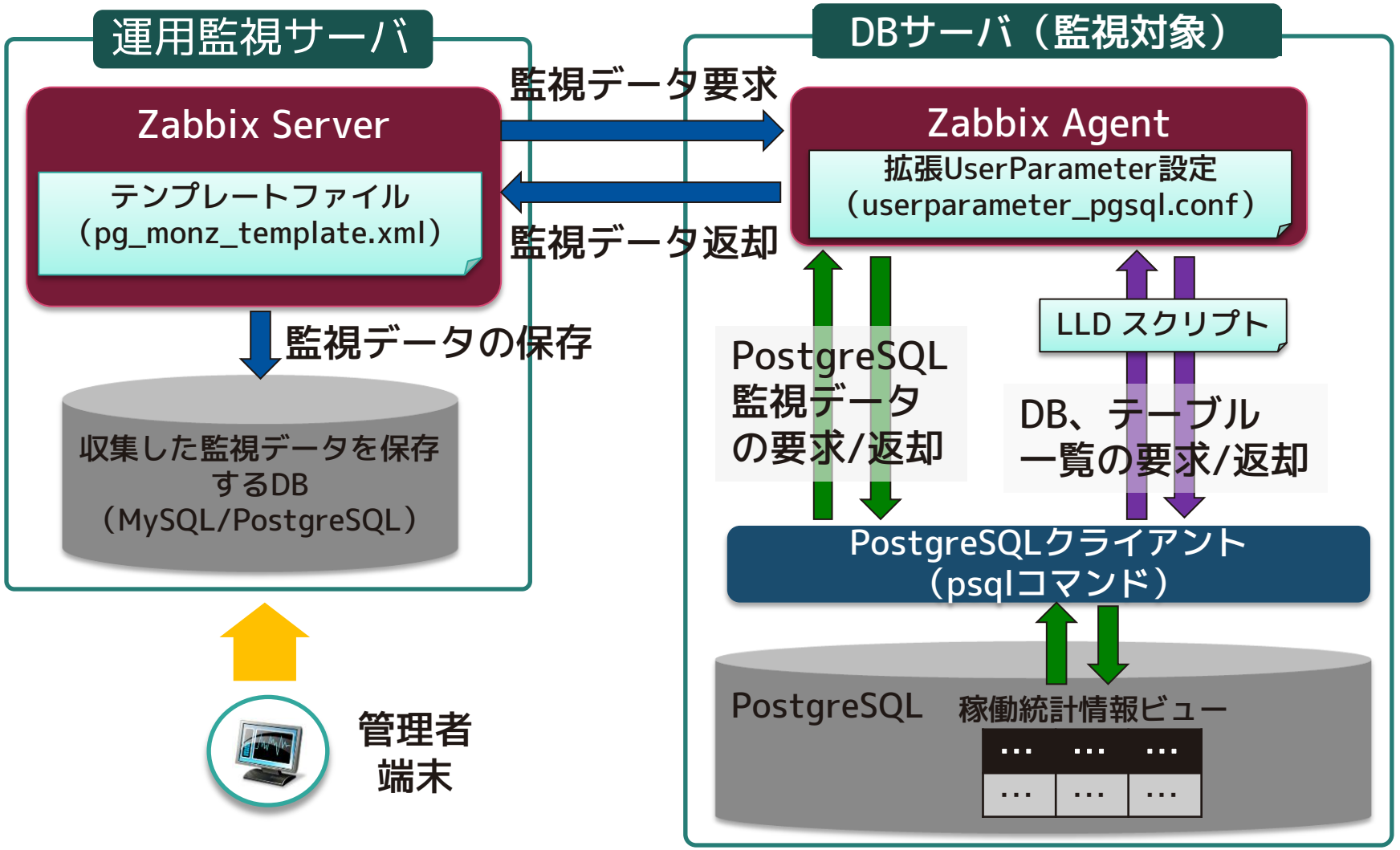
pg_monzの特徴

- **導入が容易**
  - Zabbix標準機能を利用
  - 特別なモジュール導入や設定変更は不要
- **導入手順**
  - 監視対象のサーバへ3つのファイルをコピー
  - Zabbixサーバへテンプレートをインポート
- **これだけでPostgreSQL監視がスタート**

pg_monzの特徴

- **データベース増減に監視設定が自動追随**
  - Zabbixのローレベルディスカバリ機能(LLD)でデータベースの増減を自動検出
- **活用例）**
  導入当初はデータベースDB1の監視アイテムのみ作成し、その後データベースDB2を追加
  - 通常は手動でDB2の監視アイテムを設定する。
  - pg_monzはDB2の監視アイテムを自動設定(手動での作成は不要)

pg_monzのしくみ



監視に使うPostgreSQLの機能

TIS IT Holdings Group | Go Beyond

- 稼働統計情報
- ログ

ログからわかること

- **何らかの異常が発生したこと**
  - 深刻なレベル(PANIC,FATAL)から特に影響ないレベル(INFO)まで
- **処理されたSQL**
  - 設定時間を超過したSQL(スロークエリ)
  - 監査目的で全てのSQLを出力することも可能
- **チェックポイント、自動VACUUMの実行**
- **デッドロックの発生 etc**

PostgreSQLのログの難点

TIS IT Holdings Group | Go Beyond

- **古いログのメンテ機能がない**
  - ログローテーションは可能
    - log_rotation_age, log_rotation_size
  - クリーンアップ機能はない
    - logrotate, cron 等を駆使してください
- **障害メッセージ出現時に警告できない**
  - 別途、監視ツールを使って監視する
  - Zabbix + pg_monz

PostgreSQLのログの難点

TIS IT Holdings Group | Go Beyond

- **ログをルーティングする機能がない**
  - なんでも1つのログファイルに吐き出す。
  - エラーレベルやカテゴリで分けたりできないので、ログから見たい情報を探すのが面倒
  - PostgreSQLサーバが起動できないような深刻なエラーメッセージとスロークエリログを一緒にしないで。。。

pg_monz(Zabbix)でログ監視

TIS IT Holdings Group | Go Beyond

- **深刻なエラーレベルのログを検知して、警告を出すことはできる**
  - PANIC, FATAL, ERRORの文字列を検知
- **Zabbixで細かなログ監視はやりづらい**
  - 特定の文字列が含まれるか、しかわからない
  - 細かな条件を指定した監視はできない

PostgreSQLのログを活用

TIS IT Holdings Group | Go Beyond

- **ログにしか出ない情報を性能分析/チューニングで有効に使えたら**
  - スロークエリのSQL文字列、処理時間
  - チェックポイントの実行時間
  - デッドロックの発生状況
  - ロック待ちで長時間かかったプロセス情報

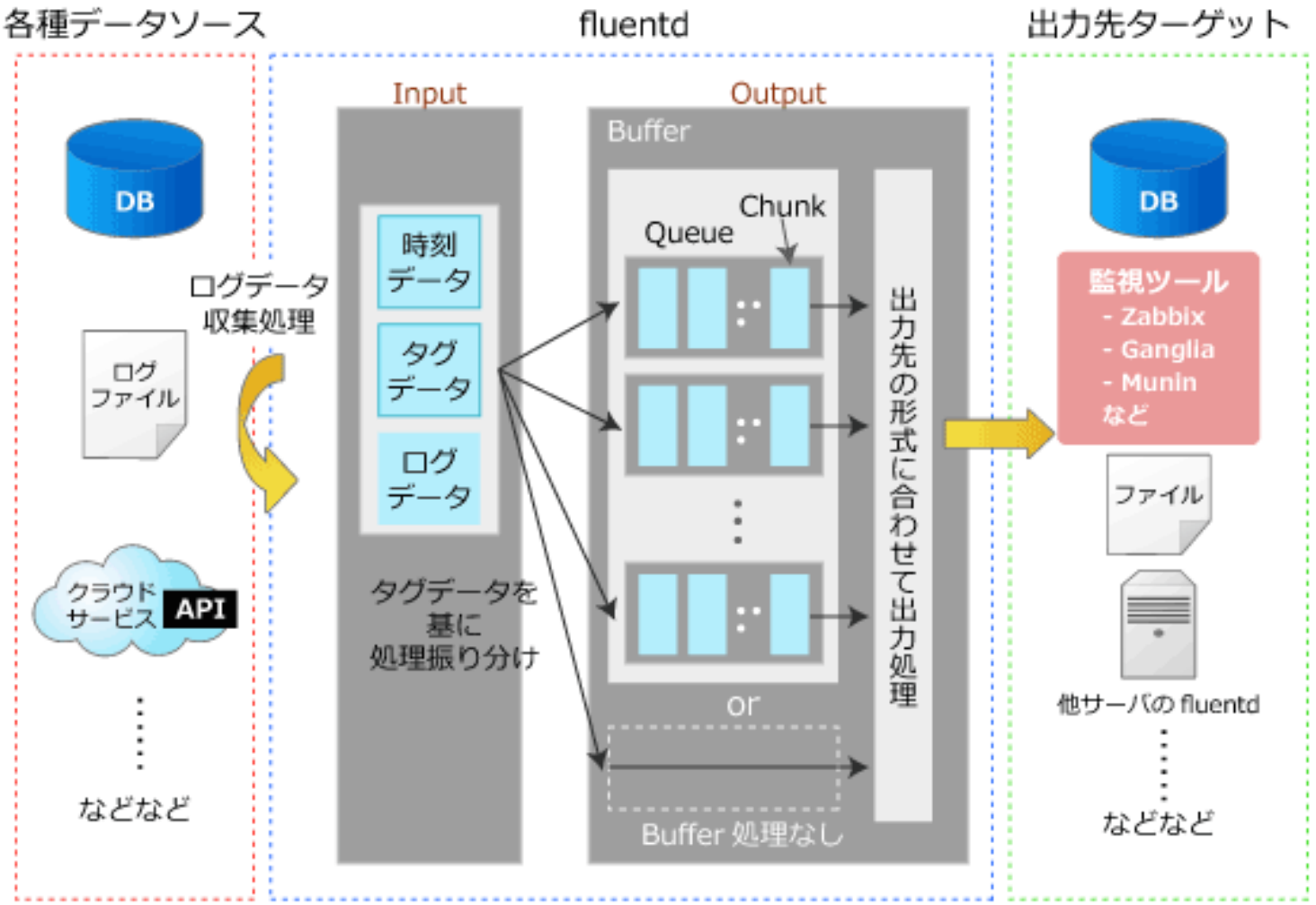
ログを分析しやすい形に加工したい

そこでFluentdですよ

TIS IT Holdings Group | Go Beyond

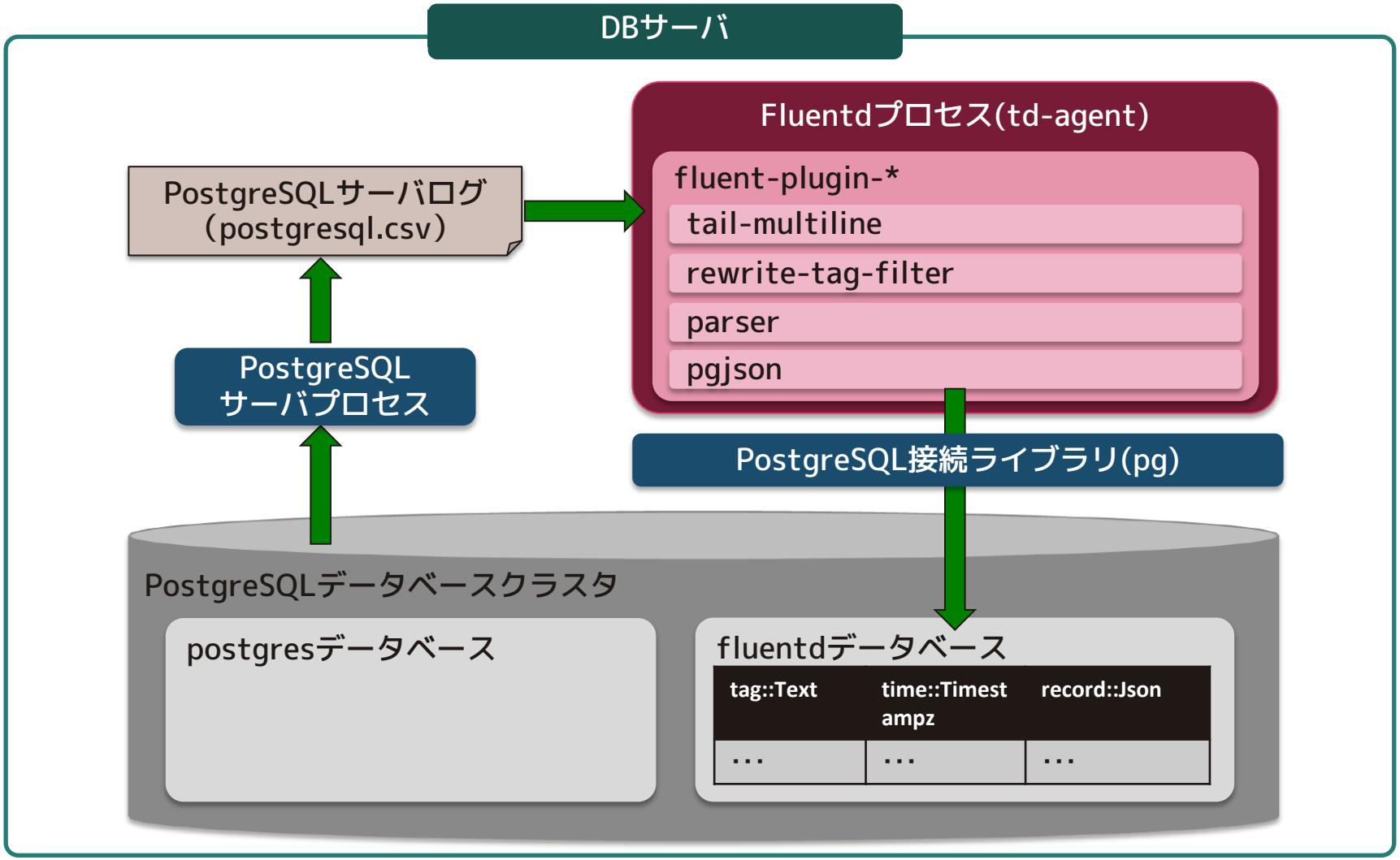
- **Fluentdとは**
  - 軽量でプラガブルなログ収集ツール
  - Apache License Version 2.0
- **Fluentdの特徴**
  - ログ管理を3つの層に分けて管理
    - インプット、バッファ、アウトプット
  - 各層がプラガブルなアーキテクチャ
    - 用途に応じたプラグインを追加するだけで使える

Fluentdの動作のしくみ

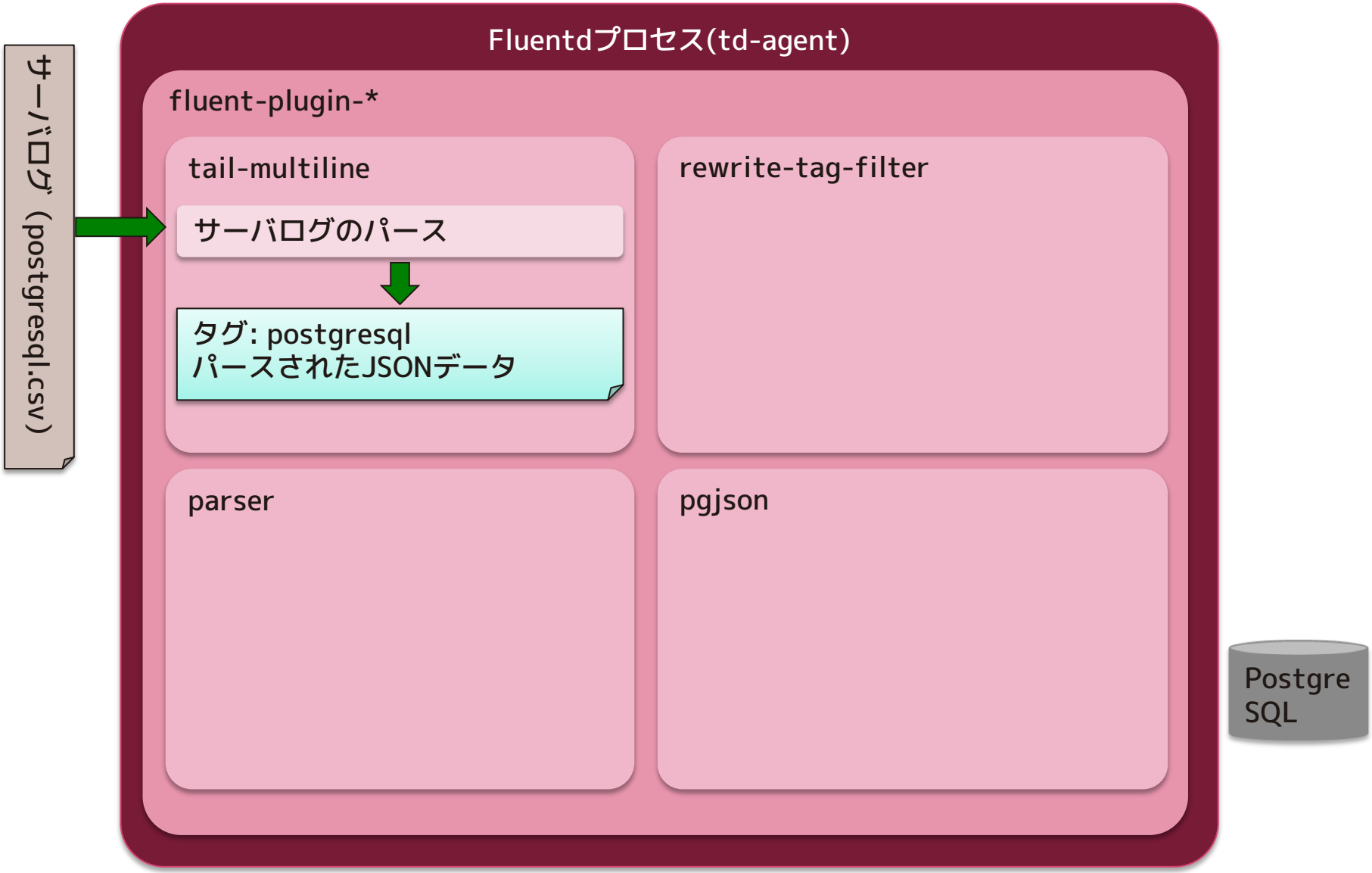


引用元: http://www.atmarkit.co.jp/ait/articles/1402/06/news007.html

Fluentdでログを収集する



Fluentdでログを収集する



PostgreSQLの設定

- postgresql.conf

```
log_destination      = 'stderr,csvlog'
logging_collector    = on
log_directory        = '/tmp'
log_filename         = 'postgresql.log'
log_file_mode        = '0644'
log_line_prefix      = '[%t][%p][%c-%l][%x][%e]%q (%u, %d, %r, %a)'
log_rotation_age     = 0
log_rotation_size    = 0

log_min_duration_statement      = '1s'
log_autovacuum_min_duration     = 0
log_checkpoints      = 'on'
log_lock_waits       = 'on'
log_temp_files       = 0
```

- CSVログの中身

```
2014-06-26 07:19:23.292 GMT,,,14376,,53abc97b.3828,1,,2014-06-26 07:19:23 GMT,,0,LOG,00000,"database
system is ready to accept connections",,,,,,,,,""
2014-06-26 07:19:23.293 GMT,,,14383,,53abc97b.382f,1,,2014-06-26 07:19:23
GMT,,0,LOG,00000,"autovacuum launcher started",,,,,,,,,""
2014-06-26 07:22:20.667 GMT,"postgres","fluentd",14782,"[local]",53abc9ff.39be,1,"SELECT",2014-06-26
07:21:35 GMT,3/0,0,LOG,00000,"duration: 30031.664 ms  statement: select pg_sleep(30);",,,,,,,,,"psql"
```

fluent-plugin-tail-multiline

TIS IT Holdings Group | Go Beyond

- **ログファイルに書き込まれた情報を常時取り込むインプットプラグイン**
  - 標準プラグイン(in_tail)の複数行対応版
  - PostgreSQLのCSVログには改行が含まれる
- **Fluentdの設定**

```
<source>
  type              tail_multiline
  time_format       %Y-%m-%d %H:%M:%S.%L %Z
  path              /tmp/postgresql.csv
  tag               postgresql
  pos_file          /var/log/td-agent/postgresql.log.pos
  format_firstline  /^¥d{4}-¥d{2}-¥d{2}/
  format            /^(?<time>[^",]*),"?(?<user_name>(?:[^",]|"")*)"?,"?(?<database_name>(?:[^",]|"")*)"?,
                    (?<process_id>[^",]*),"?(?<connection_from>(?:[^",]|"")*)"?,(?<session_id>[^",]*),
                    (?<session_line_num>[^",]*),"?(?<command_tag>(?:[^",]|"")*)"?,
                    (?<session_start_time>[^",]*),(?<virtual_transaction_id>[^",]*),(?<transaction_id>[^",]*),
                    (?<error_severity>[^",]*),(?<sql_state_code>[^",]*),"?(?<message>(?:[^"]|"")*)"?,
                    (?<detail>[^",]*),"?(?<hint>(?:[^",]|"")*)"?,(?<internal_query>[^",]*),
                    (?<internal_query_pos>[^",]*),(?<context>[^",]*),"?(?<query>(?:[^"]|"")*)"?,
                    (?<query_pos>[^",]*),(?<location>[^",]*),"?(?<application_name>(?:[^",]|"")*)"?$/
</source>
```

fluent-plugin-tail-multiline

TIS IT Holdings Group | Go Beyond

- パースされたJSONデータ


```
{
 "tag": "postgresql",
 "time": "2014-06-30 01:12:02+00",
 {"user_name":"","database_name":"","process_id":"27136","connection_from":"","session_id":"53b0b980.6a00",
  "session_line_num":"10","command_tag":"","session_start_time":"2014-06-30 01:12:32 GMT",
  "virtual_transaction_id":"","transaction_id":"0","error_severity":"LOG","sql_state_code":"00000",
  "message":"checkpoint starting: xlog","detail":"","hint":"","internal_query":"","internal_query_pos":"","context":"",
  "query":"","query_pos":"","location":"","application_name":""}
}
```

Fluentdでログを収集する

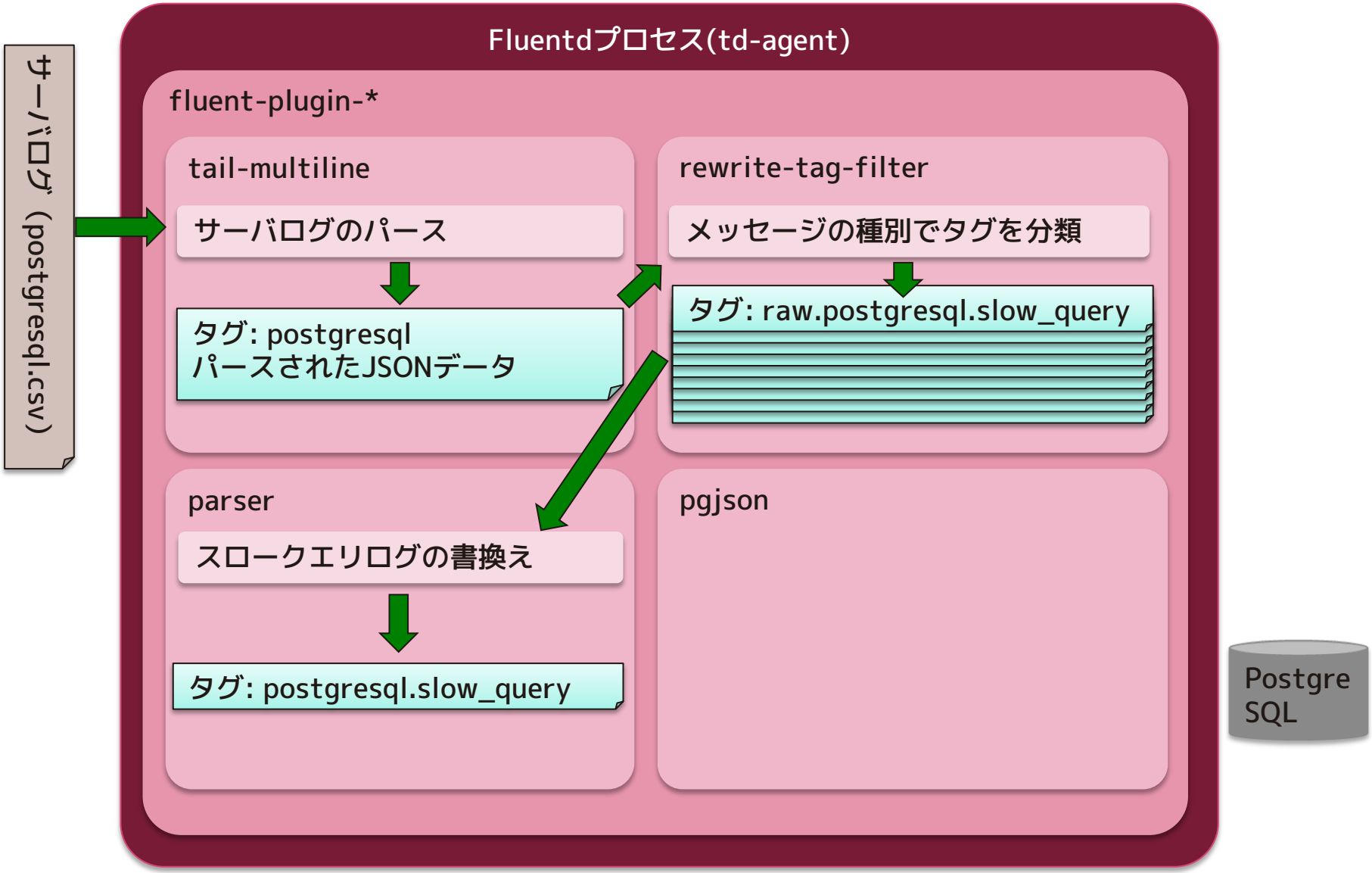
fluent-plugin-rewrite-tag-filter

TIS IT Holdings Group | Go Beyond

- データ内容を条件に任意のタグに書き換えるプラグイン
- Fluentdの設定
  - ログのメッセージ内容でタグを分別

```
<match      postgresql>
 type       rewrite_tag_filter
 rewriterule1   message   ^duration:                                raw.postgresql.slow_query
 rewriterule2   message   ^checkpoints¥sare¥soccurring¥stoo¥sfrequently
                                                                   postgresql.checkpoints.frequently
 rewriterule3   message   ^checkpoint¥sstarting:                   postgresql.checkpoint.start
 rewriterule4   message   ^checkpoint¥scomplete:                   postgresql.checkpoint.complete
 rewriterule5   message   ^automatic                               postgresql.vacuum
 rewriterule6   message   ^temporary file:                         postgresql.tempfiles
 rewriterule7   message   ^process.*detected¥sdeadlock             postgresql.deadlock
 rewriterule8   message   ^process.*(still waiting|acquired)       postgresql.lockwait
 rewriterule9   message   .*                                       postgresql.others
</match>
```

Fluentdでログを収集する



fluent-plugin-parser

TIS IT Holdings Group | Go Beyond

- **Fluentd上に流れるデータの任意のフィールドをさらに分解するプラグイン**
- **Fluentdの設定**
  - スロークエリログから処理時間,SQL文を分解

```
<match raw.postgresql.slow_query>
  type              parser
  remove_prefix     raw
  reserve_data      yes
  key_name          message
  format            /^duration: (?<duration>[0-9¥.]+) ms  statement: (?<statement>.+)$/
</match>
```

fluent-plugin-parser

TIS IT Holdings Group | Go Beyond

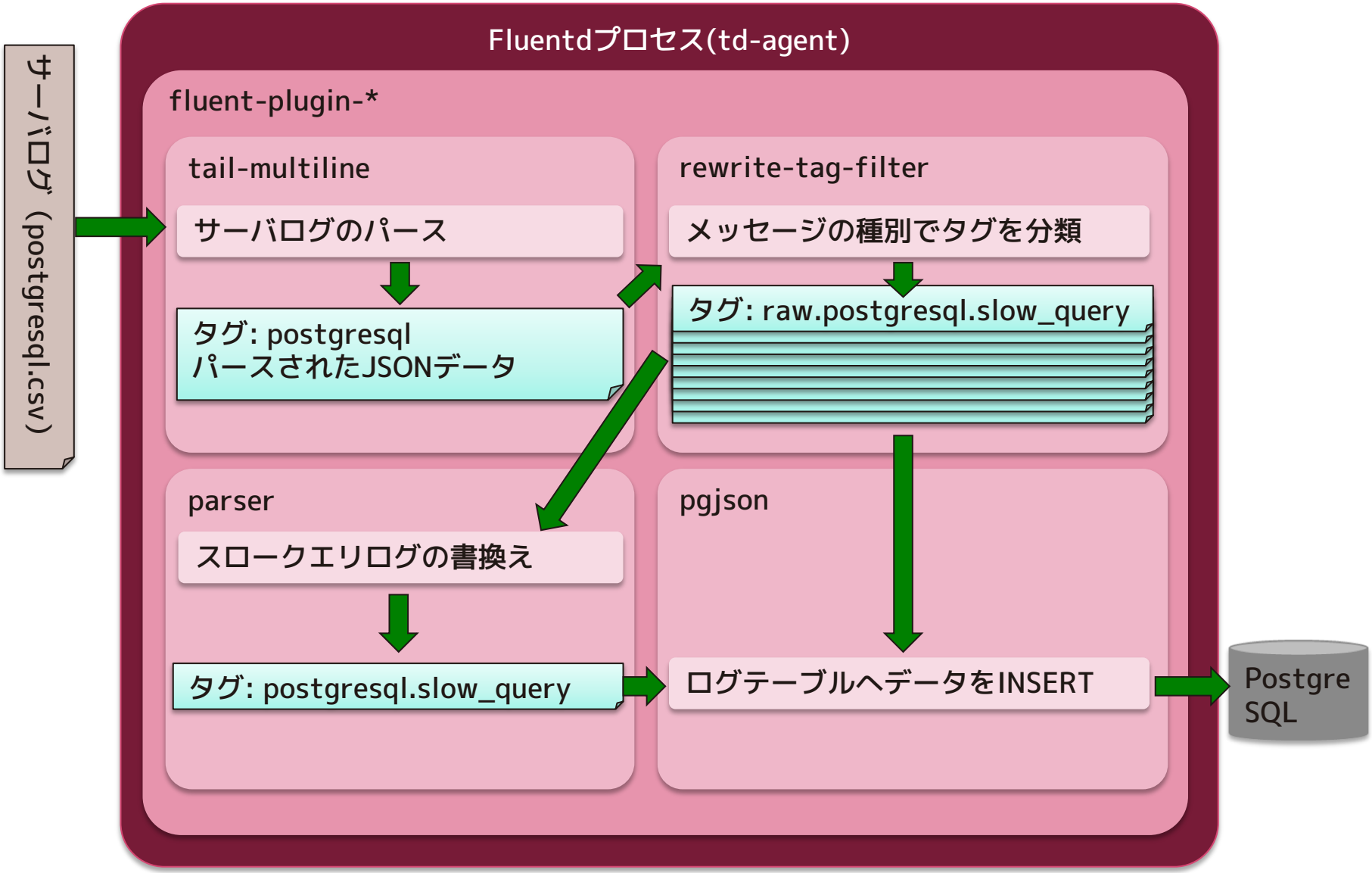
- スロークエリログの変化
  - Before

```
{
 "tag": "raw.postgresql.slow_query",
 "time": "2014-06-30 01:12:02+00",
 "user_name":"postgres","database_name":"fluentd", ･･･,
 "message":"duration: 4004.423 ms  statement: select pg_sleep(4);", ･･･ ,"application_name":"psql"}
}
```

  - After

```
{
 "tag": "postgresql.slow_query",
 "time": "2014-06-30 01:12:02+00",
 "user_name":"postgres","database_name":"fluentd", ･･･,
 "message":"duration: 4004.423 ms  statement: select pg_sleep(4);", ･･･ ,"application_name":"psql",
 "duration":"4004.423",
 "statement":"select pg_sleep(4);"}
}
```

Fluentdでログを収集する



fluent-plugin-pgjson

- **データをPostgreSQLのテーブルに出力するアウトプットプラグイン**
  - 出力先にPostgreSQLのJSON型カラムを利用
- **Fluentdの設定**
  - postgresqlで始まるタグのデータをINSERT

```
<match {postgresql.**}>
  type        pgjson
  host        localhost
  port        5432
  sslmode     prefer
  database    fluentd
  table       fluentd
  user        postgres
  password    postgres
  time_col    time
  tag_col     tag
  record_col  record
</match>
```

fluent-plugin-pgjson

TIS IT Holdings Group | Go Beyond

- **ログデータを格納するテーブル**
  - fluentdデータベースにテーブルを用意

```
CREATE TABLE fluentd (
   tag         Text,
   time        Timestamptz,
   record      Json
);
```

PostgreSQLのJSON型とは？

TIS IT Holdings Group | Go Beyond

- PostgreSQL 9.3でJSONサポートが強化
- スキーマレスなデータの格納先に使える
- Fluentdで収集したログデータをJSON型のカラムに格納すれば、フォーマットが非定型なログデータをSQLで検索できる

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- 深刻度がERRORのメッセージを検索

```
# SELECT time, record->>'user_name' as user, record->>'database_name' as database,
        record->>'error_severity' as severity, record->>'message' as message, record->>'query' as query,
        record->>'query_pos' as pos
  FROM  fluentd
  WHERE record->>'error_severity'  =  'ERROR';

-[ RECORD 1 ]----------------------------------------------------------------------------------
time         | 2014-06-30 09:49:12+00
user         | postgres
database     | fluentd
severity     | ERROR
message      | column ""datname"" does not exist
query        | select count(*) from fluentd where tag = 'postgresql.slow_query' and datname = 'postgres'
pos          | 70
```

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- スロークエリを検索（時間が長い10件）

```
# SELECT time, record->>'user_name' as user, record->>'database_name' as db,
        record->>'duration' as duration, record->>'statement' as statement
  FROM fluentd
  WHERE tag = 'postgresql.slow_query'
  ORDER BY (record->>'duration')::numeric desc LIMIT 10;

          time            |   user   |   db    | duration  |      statement
--------------------------+----------+---------+-----------+---------------------
2014-07-01 07:32:03+00 | postgres | fluentd | 10011.009 | SELECT pg_sleep(10);
2014-07-01 07:34:36+00 | postgres | fluentd | 10010.756 | SELECT pg_sleep(10);
2014-07-01 07:32:40+00 | postgres | fluentd | 10010.394 | SELECT pg_sleep(10);
2014-07-01 07:37:26+00 | postgres | fluentd | 10010.393 | SELECT pg_sleep(10);
2014-07-01 07:32:27+00 | postgres | fluentd | 10010.384 | SELECT pg_sleep(10);
2014-07-01 07:34:04+00 | postgres | fluentd | 10010.358 | SELECT pg_sleep(10);
2014-07-01 07:34:21+00 | postgres | fluentd | 10010.342 | SELECT pg_sleep(10);
2014-07-01 07:36:09+00 | postgres | fluentd | 10010.327 | SELECT pg_sleep(10);
2014-07-01 07:36:27+00 | postgres | fluentd | 10010.279 | SELECT pg_sleep(10);
2014-07-01 07:37:12+00 | postgres | fluentd | 10010.259 | SELECT pg_sleep(10);
```

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- チェックポイント頻発の警告メッセージ

```
# SELECT time, record->>'message' as message  FROM  fluentd
   WHERE  tag  = 'postgresql.checkpoints.frequently';

             time              |                                  message
-------------------------------+-----------------------------------------------------------
 2014-06-30 01:30:21+00 | checkpoints are occurring too frequently (8 seconds apart)
 2014-06-30 01:30:29+00 | checkpoints are occurring too frequently (8 seconds apart)
 2014-06-30 01:30:37+00 | checkpoints are occurring too frequently (8 seconds apart)
 2014-06-30 01:30:46+00 | checkpoints are occurring too frequently (9 seconds apart)
 2014-06-30 01:30:55+00 | checkpoints are occurring too frequently (9 seconds apart)
 2014-06-30 01:31:04+00 | checkpoints are occurring too frequently (9 seconds apart)
 2014-06-30 01:31:14+00 | checkpoints are occurring too frequently (10 seconds apart)
 2014-06-30 01:31:23+00 | checkpoints are occurring too frequently (9 seconds apart)
```

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- チェックポイントの実行状況

```
# SELECT time, record->>'message' as message  FROM fluentd  WHERE  tag  like  'postgresql.checkpoint.%';

                time                    |                                   message
----------------------------+-------------------------------------------------------------------------
2014-06-30 01:17:32+00  | checkpoint starting: time
 2014-06-30 01:17:33+00  | checkpoint complete: wrote 5 buffers (0.0%); 0 transaction log file(s) added, 0
removed, 0 recycled; write=0.403 s, sync=0.463 s, total=0.885 s; sync files=3, longest=0.460 s,
average=0.154 s
2014-06-30 01:27:32+00  | checkpoint starting: time
 2014-06-30 01:27:32+00  | checkpoint complete: wrote 1 buffers (0.0%); 0 transaction log file(s) added, 0
removed, 0 recycled; write=0.000 s, sync=0.002 s, total=0.022 s; sync files=1, longest=0.002 s,
average=0.002 s
 2014-06-30 01:30:13+00  | checkpoint starting: xlog
 2014-06-30 01:30:16+00  | checkpoint complete: wrote 1006 buffers (6.1%); 0 transaction log file(s)
added, 0 removed, 0 recycled; write=2.019 s, sync=0.272 s, total=2.396 s; sync files=19, longest=0.161 s,
average=0.014 s
 2014-06-30 01:30:21+00  | checkpoint starting: xlog
 2014-06-30 01:30:24+00  | checkpoint complete: wrote 1894 buffers (11.6%); 0 transaction log file(s)
added, 0 removed, 1 recycled; write=2.223 s, sync=0.209 s, total=2.518 s; sync files=9, longest=0.148 s,
average=0.023 s
```

収集したログをSQLで検索する

Go Beyond

- 自動VACUUMの実行状況

```
# SELECT time, record->>'message' as message  FROM  fluentd  WHERE  tag = 'postgresql.vacuum';

            time                |                                        message
--------------------------------+------------------------------------------------------------------------------------------
2014-06-30 01:31:03+00 |
automatic analyze of table ""fluentd.public.pgbench_history"" system usage: CPU 0.00s/0.02u sec elapsed 0.43 sec
2014-06-30 01:32:03+00 |
automatic analyze of table ""fluentd.public.fluentd"" system usage: CPU 0.00s/0.00u sec elapsed 0.06 sec
2014-06-30 01:32:04+00 |
automatic analyze of table ""fluentd.public.pgbench_history"" system usage: CPU 0.00s/0.05u sec elapsed 1.29 sec
2014-06-30 01:33:05+00 |
automatic analyze of table ""fluentd.public.pgbench_history"" system usage: CPU 0.01s/0.06u sec elapsed 2.03 sec
2014-06-30 01:33:09+00 |
automatic analyze of table ""fluentd.public.pgbench_accounts"" system usage: CPU 0.02s/0.05u sec elapsed 3.68 sec
2014-06-30 01:33:32+00 |
automatic vacuum of table ""fluentd.public.pgbench_tellers"": index scans: 1                    +
                       | pages: 0 removed, 30 remain                                           +
                       | tuples: 469 removed, 80 remain                                        +
                       | buffer usage: 104 hits, 0 misses, 33 dirtied                          +
                       | avg read rate: 0.000 MB/s, avg write rate: 0.002 MB/s                 +
                       | system usage: CPU 0.00s/0.00u sec elapsed 148.59 sec
```

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- ロック待ちが発生したセッション情報

```
# SELECT time, record->>'message' as message  FROM  fluentd  WHERE  tag  =  'postgresql.lockwait';

              time              |                                  message
--------------------------------+------------------------------------------------------------------------------------------------
 2014-06-29 12:04:57+00 |
process 13505 still waiting for RowExclusiveLock on relation 16385 of database 16384 after 1001.725 ms
 2014-06-29 12:05:12+00 |
process 13505 acquired RowExclusiveLock on relation 16385 of database 16384 after 16021.973 ms
 2014-06-29 12:06:18+00 |
process 13397 still waiting for RowExclusiveLock on relation 16385 of database 16384 after 1000.910 ms
 2014-06-29 12:06:29+00 |
process 13397 acquired RowExclusiveLock on relation 16385 of database 16384 after 11992.269 ms
 2014-06-29 12:06:52+00 |
process 13505 still waiting for RowExclusiveLock on relation 16391 of database 16384 after 1000.680 ms
 2014-06-29 12:06:57+00 |
process 13505 acquired RowExclusiveLock on relation 16391 of database 16384 after 6870.196 ms
```

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- 一時ファイルの生成情報

```
# SELECT time, record->>'message' as message, record->>'query' as query
  FROM fluentd WHERE tag = 'postgresql.tempfiles';

-[ RECORD 1 ]----------------------------------------------------------------
time    | 2014-07-02 09:34:33+00
message | temporary file: path ""base/pgsql_tmp/pgsql_tmp8184.0"", size 10723328
query   | explain analyze select * from pgbench_accounts order by bid;
```

収集したログをSQLで検索する

TIS IT Holdings Group | Go Beyond

- デッドロックの発生情報

```
# SELECT  time, record->>'database_name' as db, record->>'command_tag' as command,
        record->>'message' as message, record->>'query'  as query
  FROM  fluentd
  WHERE  tag  =  'postgresql.deadlock';

-[ RECORD 1 ]----------------------------------------------------------------------------------------------------
time        | 2014-07-02 09:45:02+00
db          | fluentd
command  | LOCK TABLE waiting
message   | process 8184 detected deadlock while waiting for ExclusiveLock on relation 16394 of database
16384 after 1000.235 ms
query       | LOCK TABLE pgbench_history IN EXCLUSIVE MODE;
```

pg_monzのログ監視を強化

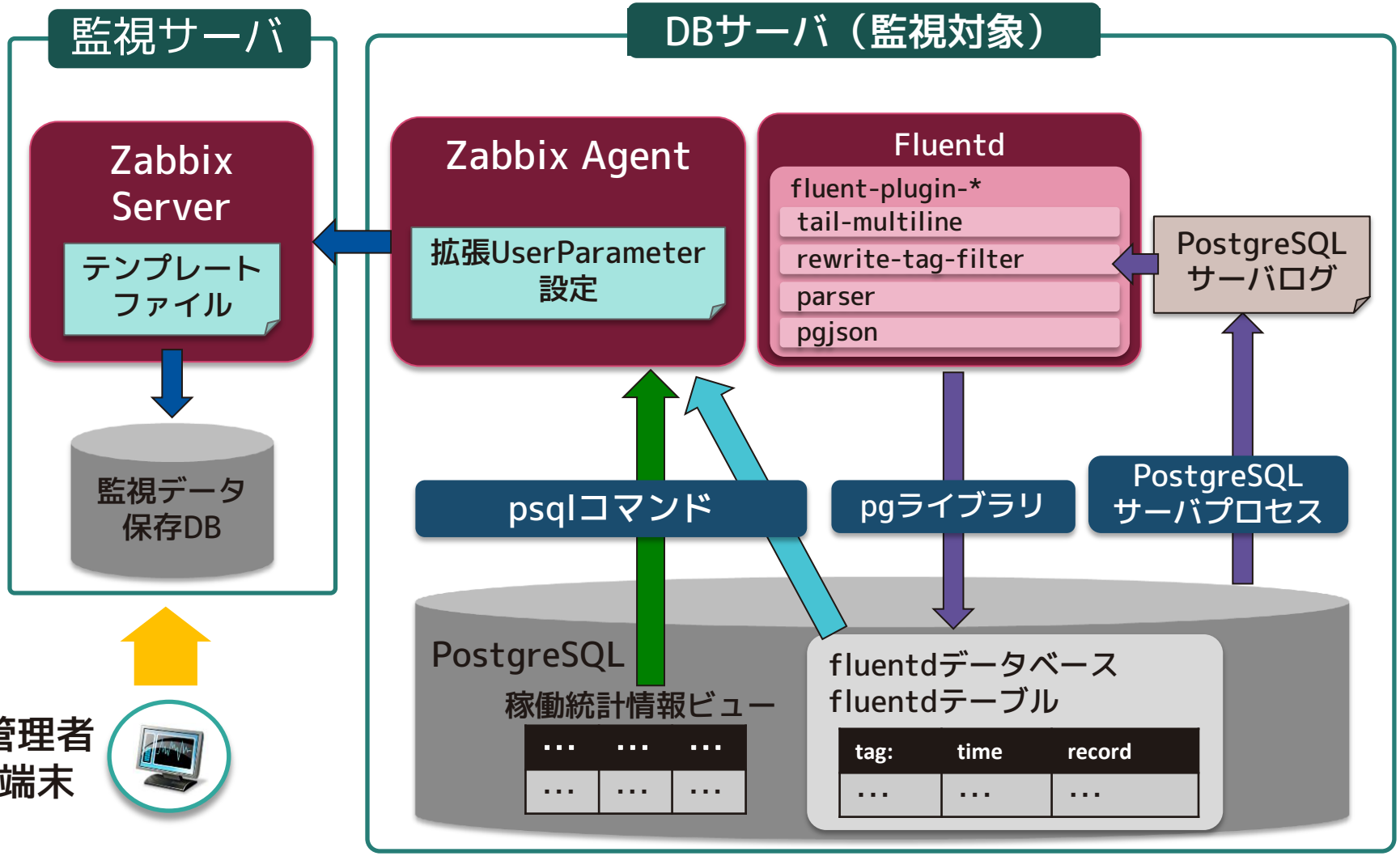
TIS IT Holdings Group | Go Beyond

- pg_monzは稼働統計情報ビューからSQLで必要な情報を取得
- ログの情報はログファイルから直接抽出せざるを得ず、細かな情報取得が困難

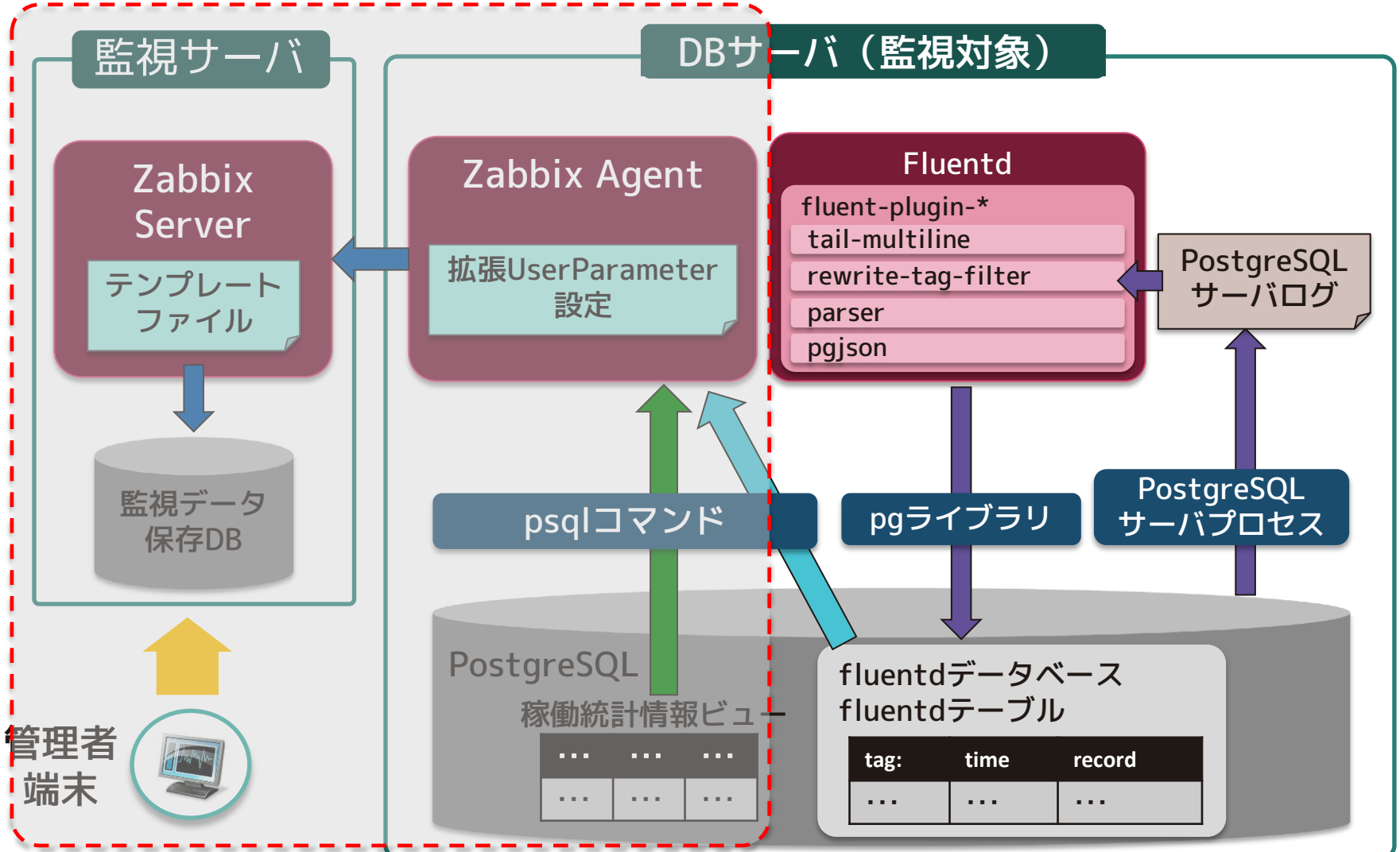
FluentdによりログもSQLで取得可能に

スロークエリの発生状況をpg_monzで監視してみよう

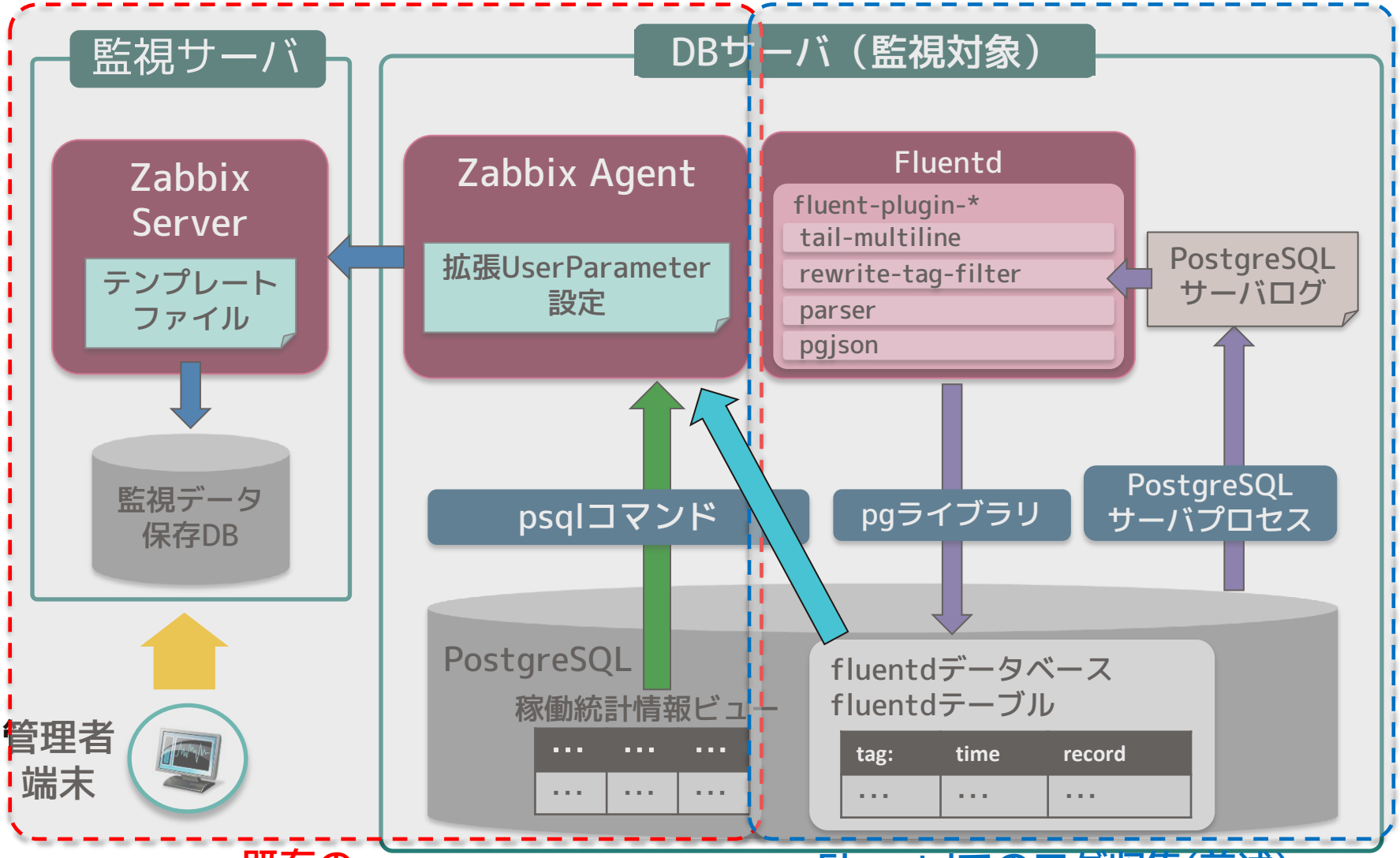
pg_monz + Fluentd



pg_monz + Fluentd



pg_monz + Fluentd

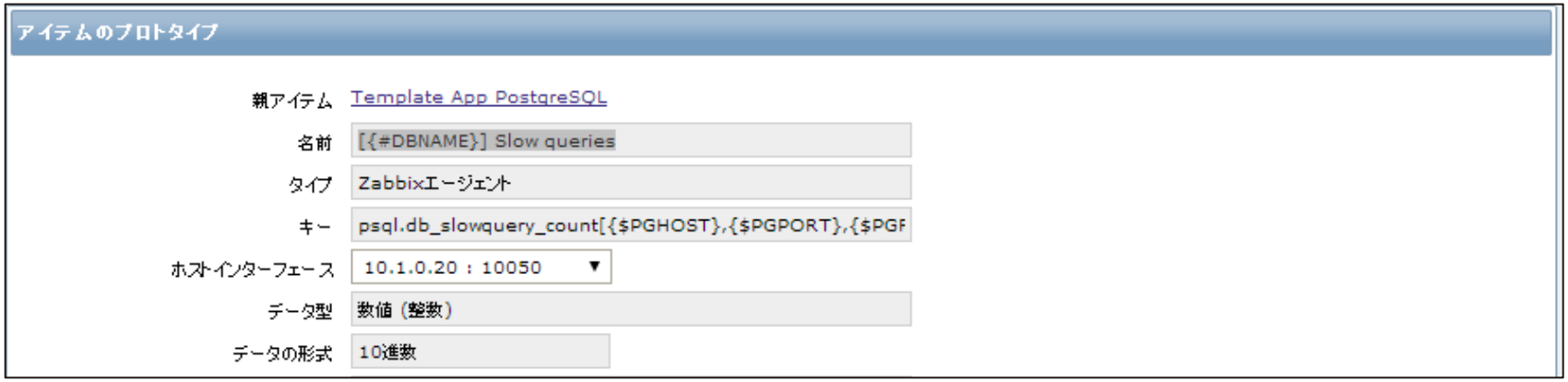


スロークエリの監視設定

- **Zabbix AgentのUserParameterを追加**
  - userparameter_pgsql.conf

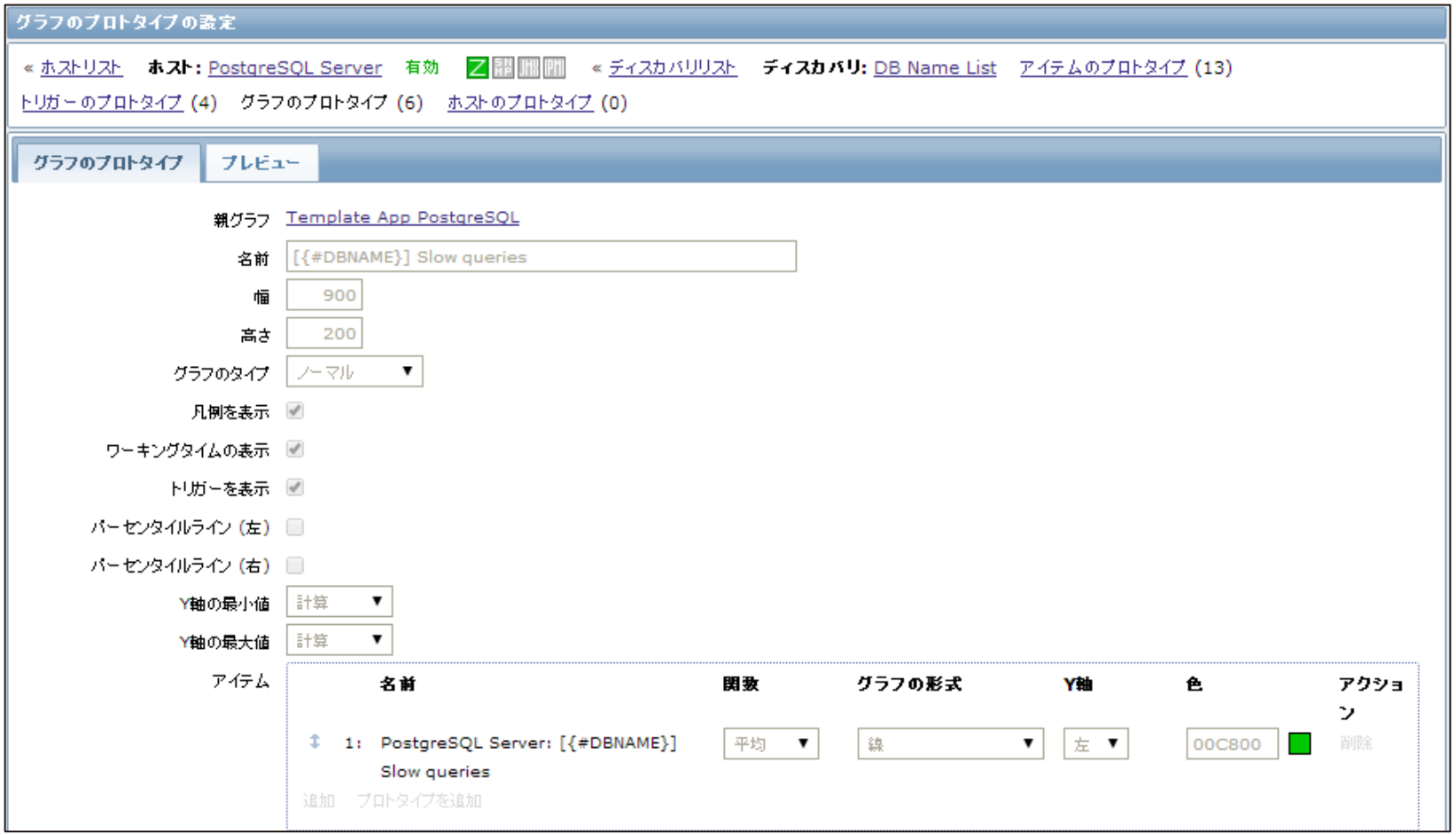
```
UserParameter = psql.db_slowquery_count[*], psql -h $1 -p $2 -U $3 -d $4 -t -c "select count(*)
from fluentd where tag = 'postgresql.slow_query' and record->>'database_name' = '$5'"
```

- **Zabbix Serverのアイテム設定を追加**
  - pg_monzではLLDのプロトタイプで作成

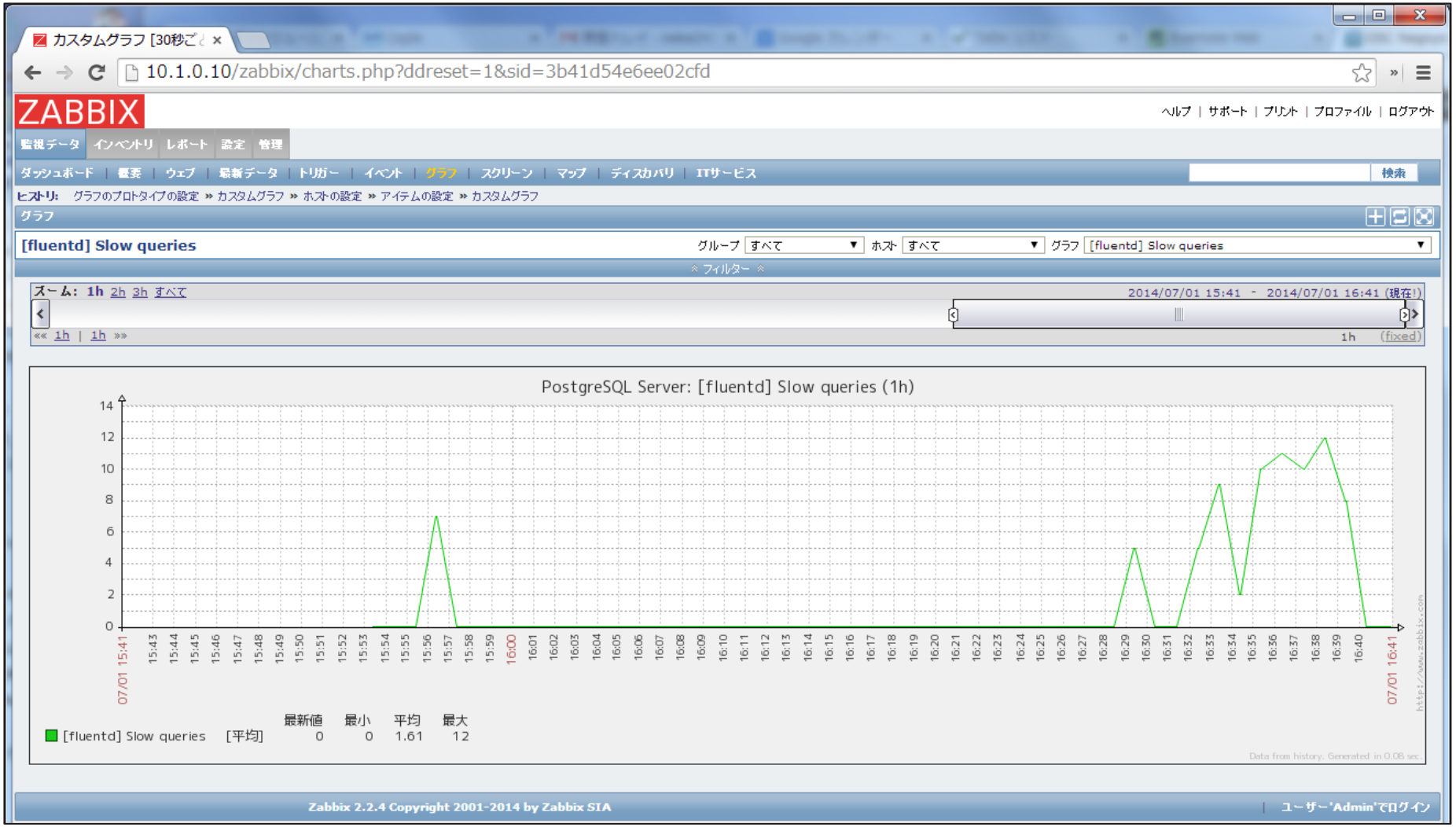


スロークエリの監視設定

- **Zabbix Serverのグラフ設定を追加**
  - pg_monzではLLDのプロトタイプで作成



スロークエリ発生回数のグラフ



まとめ

- **運用で使えるPostgreSQLの標準機能**
  - 稼働統計情報
  - ログ
- **標準機能だけではいろいろダルい**
  - 稼働統計情報：SQLを書くのがダルい
  - ログ：分析に必要な情報を取出すのがダルい
- **ツールを組合せて楽をしよう**
  - pg_monz で楽々監視
  - fluentd で楽々ログ管理

pg_monzの入手、問合せ先

- 入手先や使い方
  - http://pg-monz.github.io/pg_monz/
- 問い合わせ
  - pg_monz ユーザーグループ
  - pg_monz@googlegroups.com

参考情報

TIS IT Holdings Group | Go Beyond

- 明日から使えるPostgresql運用管理テクニック(監視編)
  - http://www.slideshare.net/kasaharatt/postgre-sql-26186128
- Let's Postgres - 稼動統計情報を活用しよう(1)
  - http://lets.postgresql.jp/documents/technical/statistics/1
- pg_perf
  - http://pgsqldeepdive.blogspot.jp/2012/12/blog-post.html
- pg_stats_info
  - http://pgstatsinfo.projects.pgfoundry.org/pg_statsinfo-ja.html
- SIOS “OSSよろず”ブログ出張所
  - PostgreSQL9.3 の新機能 〜JSON編〜
  - http://sios-oss.blogspot.jp/2013/09/postgresql93-json.html

参考情報

- 今さら聞けないfluentd〜クラウド時代のログ管理入門
  - http://www.atmarkit.co.jp/ait/articles/1402/06/news007.html
- PostgreSQLのログをfluentdで回収する設定
  - http://chopl.in/blog/2013/06/07/postgresql_csv_log_with_fluentd.html
- fluent-plugin-tail-multiline
  - https://github.com/tomohisaota/fluent-plugin-tail-multiline
- fluent-plugin-rewrite-tag-filter
  - https://github.com/fluent/fluent-plugin-rewrite-tag-filter
- fluent-plugin-parser
  - https://github.com/tagomoris/fluent-plugin-parser
- fluent-plugin-pgjson
  - https://github.com/choplin/fluent-plugin-pgjson


TIS
IT Holdings Group
Go Beyond