PostgreSQL における匿名通信システムを介した外部データラッパ機能の実現と評価

2015SC058 松山到生
指導教員:石原靖哲

1 はじめに

現在注目されている技術に匿名通信システムがある．その中の一つとして Tor (The Onion Router) がある．これは，日本では 2012 年に起きた「パソコン遠隔操作事件」をきっかけに広く知られるようになった．世界では自由にインターネットで発信したり，メッセージのやり取りをしたりすることが難しい国もある．Tor はそういった国で活動する人権活動家にとってはなくてはならないシステムである．Tor は複数のプロキシを経由することで，高い匿名性を実現しているため，第三者が OR(Onion Router) を流れているデータの内容及び送受信者を直接把握することはできない．

本研究は PostgreSQL[2] で SELECT 文や UPDATE 文を使用し，外部にあるデータにアクセスすることができるようにする外部データラッパ (Foreign Data Wrapper，以下 FDW と省略) という拡張機能を Tor を介して行うことを目標にする．この機能を実現することによって，隠れた通信を行わないといけない人でも，関係データのやり取りが可能になる．例えば，高校と大学が生徒の成績を関係データとしてやりたいとする．その際関係データのやり取りを行っている学校以外の外部には知られたくないことであり，秘匿するべきことである．生徒の成績関係データを外部に漏らさず安全にやり取りするために，本研究の目標である FDW を Tor を介して行うことは有用な手段といえる．

2 The Onion Router

Tor は元々は米海軍調査研究所で開発された技術であり，現在最も利用されている匿名通信システムである．Tor ネットワークから無作為に選ばれた複数のプロキシ (OnionRouter: OR) を経由し，「Onion Routing」と呼ばれる仮想回線接続により匿名性を高めた通信をすることができる．そして，この OR は世界に 6000 以上あるといわれている．図 1 に Tor による通信の様子を示す．また Tor は SOCKS プロキシを経由して通信を行っている．本研究では，SOCKS プロキシを介し FDW を行うことが最終的な目標である．

3 SQL と PostgreSQL

SQL[3] は構造化問い合わせ言語 (Structured Query Language) であり，リレーショナルデータベース (RDB) の作成や操作等を行うことのできる問い合わせ言語である．「表」形式という分かりやすい関係データベースを基本としており，使用するにあたって使用者にとって理解のしやすい言語である．SQL はデータベース言語として国際標準化機構 (ISO) で規格化されているため，一つのデータベース用の SQL を覚えればほとんどの RDB で同じように使用することができる．

PostgreSQL はリレーショナルデータベースの作成や操作等ができるオープンソースのオブジェクトリレーショナルデータベース管理システム (ORDBMS) の一つである．様々な OS に対応しており，無料で使用することができる．他にも無料で使用できる DBMS のなかには PostgreSQL と同じほど人気のある MySQL があるが，本研究では拡張性に優れている PostgreSQL に着目し，その中の機能の一つである FDW を扱う．

4 外部データラッパ

外部データラッパは外部にあるデータにアクセスするための PostgreSQL の拡張機能である．PostgreSQL からアクセスできる外部データは様々あり，PostgreSQL 以外の RDB，NoSQL データベース，ビッグデータ，ファイル (CSV ファイル，プレーンテキスト等) などがある，外部のデータにアクセスするためには，対応した FDW を PostgreSQL の拡張機能としてインストールし，外部データを外部テーブルとして定義する．そして，外部テーブルに対し，SQL 文で問い合わせることで FDW を介して外部データにアクセスすることができる．図 2 に FDW の通信の様子を示す．

5 実現

5.1 実現環境

表 1 に実現環境を示した．また，本研究ではサーバ側の PC で実現用のデータベースを用意し，クライアント側の PC からサーバ側の PC のデータベースに FDW を行うとする．

5.2 実現の詳細

FDW を Tor を介して行うために以下のような手順で実現を目指した．

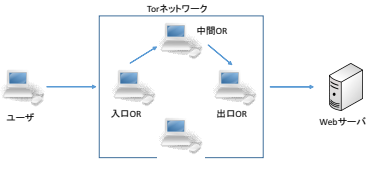

図 1 Tor による通信

表 1　実現環境

	クライアント	サーバ
CPU	Intel 1.80GHz	Intel 2.60GHz
OS	Linux20.04	Linux20.04
Tor	v0.4.2.7	v0.4.2.7
PostgreSQL	v12.5	v12.5

表 2　SQL の問合せ実行時間

	問い合わせ実行時間 [ms]
FDW	819
FDW over SSH	2039
FDW over SSH over Tor	100470

1. PC2 台に PostgreSQL をインストール・起動
2. 家のネットワーク内だけで FDW が動作することを確認
3. FDW のサーバ側のサービスを研究室 SSH サーバにリモートポートフォワーディングし，SSH サーバ経由でもう 1 台の PC から FDW サーバにアクセスできることを確認
4. FDW のクライアント側で Tor を起動し，リモートポートフォワーディングした FDW のサーバにアクセスできることを確認

サーバ側とクライアント側においての FDW は成功したが最初はうまくいかなかった．理由としては PostgreSQL の 2 つの設定ファイルを変更する必要があったためである．1 つ目のファイル postgresql.conf では接続をうけるホストをクライアントに変更し，2 つ目のファイル pg_hba.conf では，クライアントのアドレスとロール名を指定し FDW を行うデータベースに対して接続を許可をする変更を行ったところ，クライアント側からサーバ側への FDW が成功した．その後ポートフォワーディングを行い SSH サーバを介した FDW も同じく成功した．今回 SSH サーバと接続するにあたって公開鍵認証を用いて接続を行い成功した．しかし，Tor を介し FDW を行ったところうまくいかなかった．間違いなくサーバ側とクライアント側で Tor は起動していたが，その状態で FDW を行っても Tor を介すことはできなかった．そのため，データベースがある側のサーバ側で Tor を起動し，さらに SSH サーバを介してポートフォワーディングによってクライアント側から FDW を行ったところうまく繋げることに成功した．

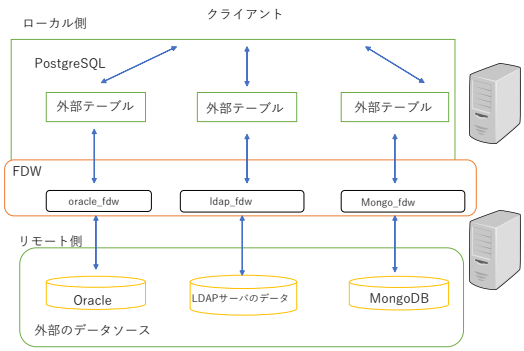

図 2　fdw のイメージ

6　評価

FDW のしかたとして以下の 3 通りを実現し，SQL を実行してから結果が返ってくるまでの総時間の比較を行った．データベースとして，DBMS のベンチマークを測るために用いられるオープンソースの HammerDB[1] を使用した．

- 何も介さず FDW を行った場合 (FDW)
- SSH サーバを介して FDW を行った場合 (FDW over SSH)
- SSH サーバと Tor を介して FDW を行った場合 (FDW over SSH over Tor)

その結果 Tor を介した場合はかなりの遅延が発生していた．FDW のみと SSH サーバを介したときと比較すると倍以上の遅延が発生していた．以下の表 2 には HammerDB 内の tpcc というデータベースから item というテーブルに対して i_id が 10000 以下という条件を付けて問い合わせをした場合の実行時間 (5 回試行したときの平均) を示している．

```
select * from item where i_id <=10000
```

7　まとめ

本来は Tor を介して直接 FDW を行えるようにすることが目標であったが，本研究では実現することができなかったため，これを実現することが今後の目標である．さらに，本研究からも Tor を介して通信を行った場合かなりの時間がかかることが分かった．これは，SSH サーバを介さず直接 Tor を繋いだとしても同じように時間はかかると思われる．そのため，Tor を繋ぐことと同時に FDW の圧縮も今後行うべきことの 1 つである．

参考文献

[1] Hammerdb. https://www.hammerdb.com/.

[2] PostgreSQL グローバル開発グループ．PostgreSQL 12.4 文書, 第 12 版, 2020.

[3] 川越恭二. 楽しく学べるデータベース. 共立出版, 2016.