ソケットAPIを用いた簡易FTPサーバの実装

目次

目次

FTPとは？

File Transfer Protocol の略

- 1970年代(インターネット初期)から登場
- インターネットでよく用いられるプロトコルの一種
- 最近はそうでもないかも・・・

File　Transfer　Protocol

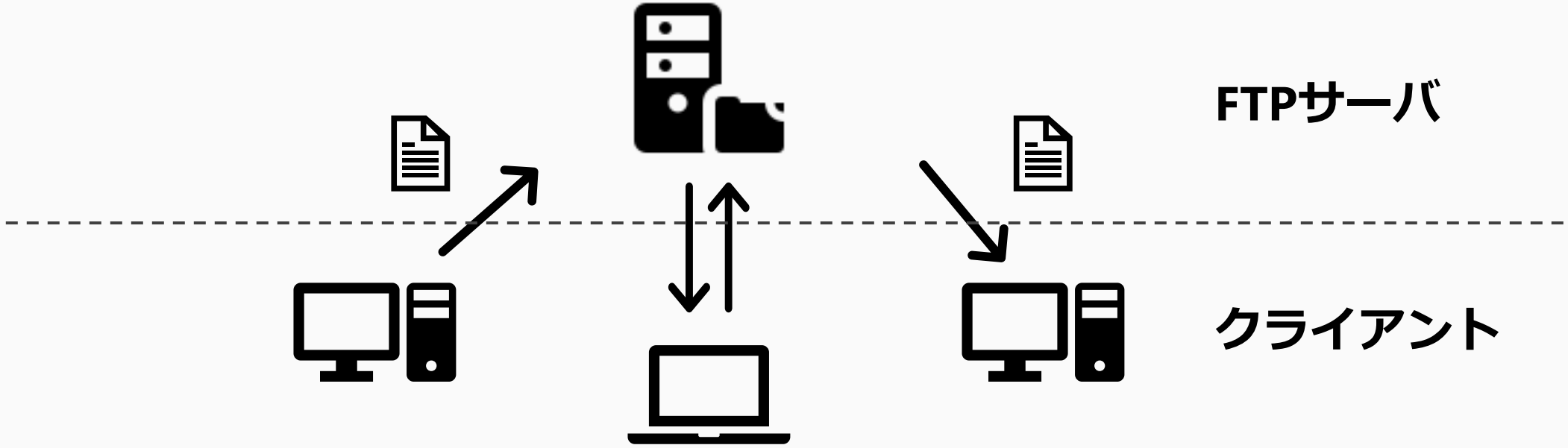
文字通り**ファイル**を**転送**するための**プロトコル**

FTPとは？

典型的な**クライアントサーバモデル**

サーバにファイルを**アップロード**

サーバからファイルを**ダウンロード**



目次

FTPの仕組み

クライアントとサーバ間で**2**本のTCPコネクションを確立

- **コントロールコネクション**

  クライアントからサーバへのコマンド送信

  サーバからクライアントへの応答送信などを行う

- **データコネクション**

  実際のファイルデータのやり取りのみ行う

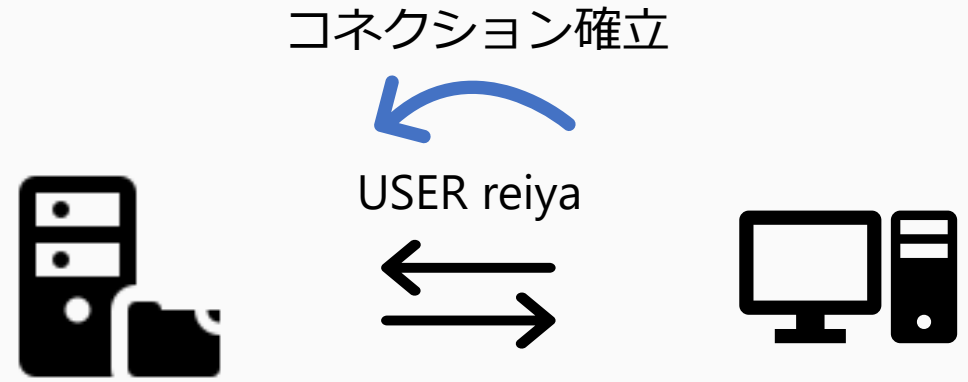
FTPの仕組み

コントロールコネクション

・**クライアント**から**サーバ**に対してコネクション確立

クライアント→サーバ：コマンド名 （+内容）を送信

サーバ→クライアント：レスポンスコード + メッセージを送信



FTPの仕組み

コントロールコネクション

・コマンド例

USER <ユーザー名>：指定したユーザー名でログイン

PASS <パスワード>：ユーザーのパスワードを送信

QUIT　　：ログアウトする

RETR <ファイル名>：ファイルをサーバからダウンロード

STOR <ファイル名>：ファイルをサーバへアップロード

FTPの仕組み

データコネクション

・ファイルの送受信毎にコネクションを確立(2つの方法)

1. **アクティブモード**：
   ・クライアントからサーバにIPアドレスとポート番号を通知
   ・**サーバ**から**クライアント**に対してコネクションを確立

2. **パッシブモード**：
   ・サーバからクライアントに使用するポート番号を通知
   ・**クライアント**から**サーバ**に対してコネクションを確立

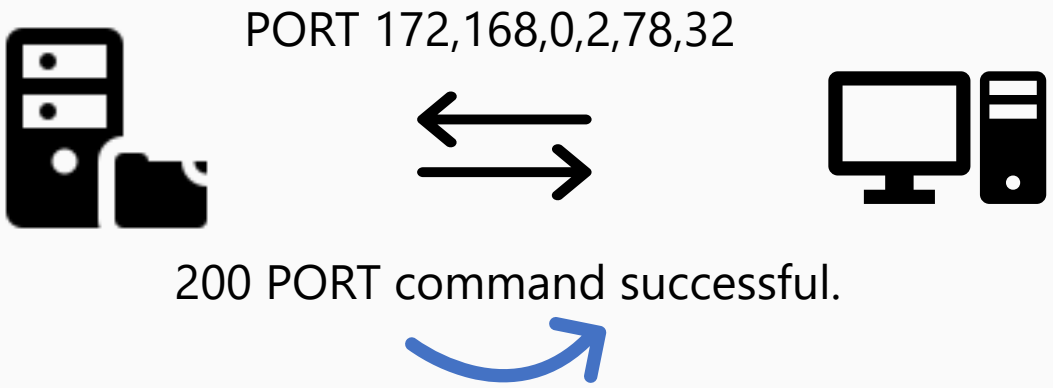
FTPの仕組み

アクティブモード

・クライアントからサーバへ**PORT**コマンドを用いる

例：IPアドレスが172.168.0.2でポート番号が20000の時

PORT　172,168,0,2,78,32



FTPの仕組み

アクティブモード

例：IPアドレスが172.168.0.2でポート番号が20000の時

PORT　172,168,0,2 , 78,32

IPアドレス：172.168.0.2 → 172,168,0,2

PORT番号　：20000 → 78,32？

- PORT番号は**16ビット**
- 上位**8ビット**と下位**8ビット**に分けて送信
- つまり、**78** × **256**(8ビットシフト) + **32**=**20000**

目次

実装

実装したコマンド（クライアント）

fetch <ファイル名>：ファイルをサーバからダウンロード

store <ファイル名>：ファイルをサーバへアップロード

exit　：ログアウト（プログラムの終了）

ls　　：サーバのファイル一覧を取得

myls：自分のファイル一覧を表示

cd <ディレクトリ名>：自分のカレントディレクトリを変更

man　：コマンド一覧を表示

実装

fetchコマンドとstoreコマンド

- **アクティブモード**で**データコネクション**を確立
  - **PORT**コマンドを使用
- コマンド実行時にデータコネクション確立
- コマンド終了時にデータコネクション破棄

- **コントロールコネクション**を用いて送受信の制御
  - 送信中または送信完了の合図をやり取り

実装

exitコマンド

- **QUIT**コマンドをサーバに送信
- QUITの応答受信後にコントロールコネクションを破棄

lsコマンドとmylsコマンド

- **fork()**で子プロセスを生成し**exec()**でls –lを実行させる
  - サーバ：**stdout**の出力先をネットワークのソケットにつなげる

実装

実装したコマンド（サーバ）

USER：ユーザーネームをクライアントから受け取る
PASS：パスワードをクライアントから受け取る
FILE　：該当のファイルが存在するかを確認
PORT：クライアントとのデータコネクションを確立
RETR：ファイルをクライアントに送信
STOR：ファイルをクライアントから受信
QUIT：クライアントとの接続を切断

実装

USERコマンドとPASSコマンド

- ユーザー名とパスワードを受け取る
- **ユーザーログイン機能**は**未実装**のため受け取るだけ

FILEコマンド

- **fetch**コマンドまたは**store**コマンド実行時に実行
- **stat()**を用いてファイルの存在をチェック
  - 存在した場合はファイル送受信の準備に移行
  - 存在しなかった場合はその旨をクライアントに通知

実装

PORTコマンド

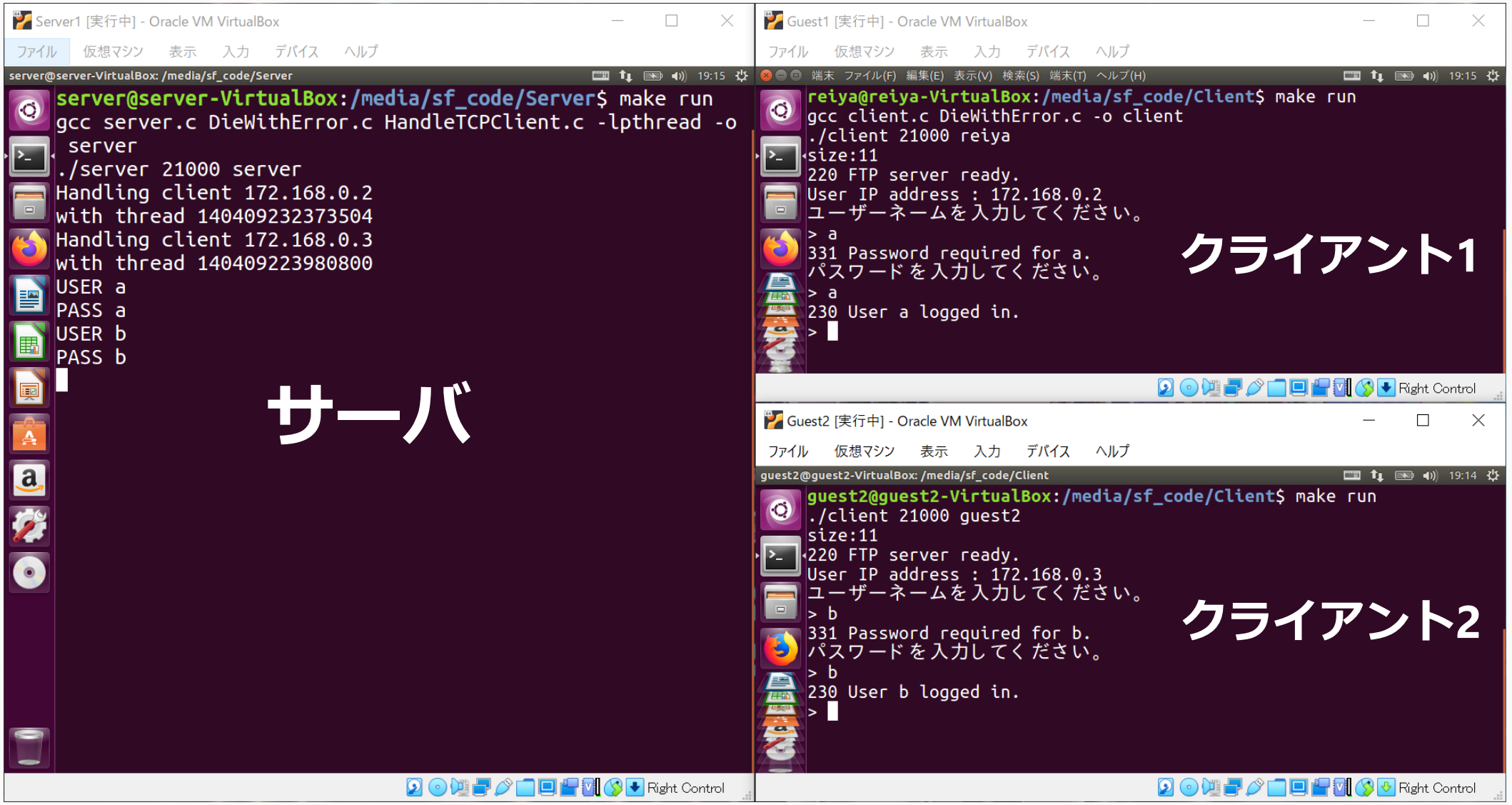
- 受信したメッセージからIPアドレスとポート番号を復号
- **データコネクション**を**サーバ**から**クライアント**へ確立
  - アクティブモード

RETRコマンドとSTORコマンド

- **データコネクション**を用いてファイルデータを送受信
- コマンド完了時に**データコネクション**を破棄

目次

デモ


Server1 [実行中] - Oracle VM VirtualBox
ファイル 仮想マシン 表示 入力 デバイス ヘルプ
server@server-VirtualBox: /media/sf_code/Server
19:15
server@server-VirtualBox:/media/sf_code/Server$ make run
gcc server.c DieWithError.c HandleTCPClient.c -lpthread -o
server
./server 21000 server
Handling client 172.168.0.2
with thread 140409232373504
Handling client 172.168.0.3
with thread 140409223980800
USER a
PASS a
USER b
PASS b
サーバ
Right Control
Guest1 [実行中] - Oracle VM VirtualBox
ファイル 仮想マシン 表示 入力 デバイス ヘルプ
端末 ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
19:15
reiya@reiya-VirtualBox:/media/sf_code/Client$ make run
gcc client.c DieWithError.c -o client
./client 21000 reiya
size:11
220 FTP server ready.
User IP address : 172.168.0.2
ユーザーネームを入力してください。
> a
331 Password required for a.
パスワードを入力してください。
> a
230 User a logged in.
>
クライアント1
Right Control
Guest2 [実行中] - Oracle VM VirtualBox
ファイル 仮想マシン 表示 入力 デバイス ヘルプ
guest2@guest2-VirtualBox: /media/sf_code/Client
19:14
guest2@guest2-VirtualBox:/media/sf_code/Client$ make run
./client 21000 guest2
size:11
220 FTP server ready.
User IP address : 172.168.0.3
ユーザーネームを入力してください。
> b
331 Password required for b.
パスワードを入力してください。
> b
230 User b logged in.
>
クライアント2
Right Control

実装

実装できなかった仕様

- ユーザーのログイン機能
  - サーバにユーザーの登録機能を実装する必要がある
  - パーミッションの問題(アクセス許可やファイルのオーナー等)
- サーバのディレクトリ移動
  - 複数クライアントが同時に移動させようとした時の挙動が未定義
  - 同様にパーミッションの問題(ルートディレクトリ移動等)
- 大容量ファイルの送受信
  - 書き込み中に次のデータを読み込んでしまう(flushができない)

目次