情報通信システム論　レポート課題

学籍番号: 1240293
氏名: 植田蓮

2023 年 10 月 30 日

概要

この文書は 2023 年度の情報通信システム論レポート課題である．“クライアントプログラムを C 言語で作成せよ” という課題で作成したプログラムの説明，実行例，考察，感想を記述する．

1　課題の要件

作成するプログラムの要件は以下を満たすことである．

- netcat ツールのクライアント機能を実現する．
- Socket API を用い，指定された IP アドレス w ポートのサーバーに TCP で接続する．
- サーバーと接続したソケット及び標準入力の 2 つを select() で監視する．
- もし標準入力から入力があったら，その内容をサーバーへ送信する．
- もしサーバーからデータを受信したらそれを標準出力へ出力する
- どちらかが EOF になるか，エラーが発生するまで上記を繰り返す．
- サーバとの接続を切断して終了する

2　プログラムの説明

このプログラムは，実行時にコマンドライン引数としてホスト名と，ポート番号を指定する必要がある．どちらか一方でも不足している場合には argument error をとしてプログラムを終了する．また，指定できるポート番号は TCP に則り，1 から 65535 までの整数に限定している．

指定されたサーバーの名前解決には，getaddrinfo を用いている．getaddrinfo の hints には，PF_UNSPEC を指定しているため，IP のバージョンは関係なくサーバーに接続する．一方で，SOCK_STREAM を指定しているため，UDP と TCP どちらもサービスしているようなサーバーであっても，TCP で接続を行う．指定された，サーバーへの接続が確立できない，すなわち，socket を作成することができない場合には，例外処理を行い，プログラムを終了する．接続が確立できた婆には，接続が確認できたソケット及び標準入力を select によって監視する．select のタイムアウト時間は指定していないため，指定された記述子がブロックしない状態になるまで必ずブロックを続ける．読み取りがブロックしない状態になった記述子があれば，その記述子から read を行う．

サーバーとの接続ソケットから read した場合には，読み込んだ内容を標準出力へ出力を行う．標準入力から read した場合には，読み込んだ内容をサーバーへと送信する．また，どちらの記述子から read した場合でも，読み込んだ結果のサイズが 0 の場合には，EOF を読み込んだとして，サーバーとの接続ソケットを close して，プログラムを終了する．

3 プログラムの実行例

高知工科大学情報学群の Web サーバへと接続を試みた例を示す.

```
------------------------------------------------
user@kut % ./a.out www.info.kochi-tech.ac.jp 80
hoge
HTTP/1.1 400 Bad Request
Date: Sun, 29 Oct 2023 03:26:56 GMT
Server: Apache
Content-Length: 226
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
</body></html>
EOF
------------------------------------------------
```

“hoge” というリクエストを送った結果として “Bad Request” のレスポンスが返された．また，Web サーバ側が接続を close したことによって，プログラムが終了している．

さらに，2022 年度 4Q のネットワーク設計で作成した，しりとりサーバーとの接続例を以下に示す．しりとりサーバーに複数回のリクエストを送信することができていることが確認できる．

```
------------------------------------------------
user@kut % ./a.out localhost 12345
i
current_word:i
in
current_word:in
now
current_word:now
word
current_word:word
drive
current_word:drive
eeeeee
I don't know that word
iiiiii
Terminating and initial letters do not match
^C
------------------------------------------------
```

4　考察

今回作成したプログラムでは read するためのバッファのサイズを 1024 に設定している．そのため，1024 バイト以上のデータのリクエストは送信することができない．テキストだけであれば，小規模のプログラムで用いるプロトコルであれば 1024 バイトでも十分かもしれないが，プロトコルによってはテキストのみでも 1024 バイトを超える可能性は十分に考えられる．クライアントプログラムは，エンドユーザ本人が使うものだと，限定して，大規模なデータのを受け取るか受け取らないかは，サーバ側に任せて，クライアント側のバッファサイズは十分に大きくするという手段も有効だと考えられる．しかし，バッファサイズの設定についてはそのクライアントプログラムの利用用途を十分に考えてどの程度が適当なサイズか，十分に検討して設計すべきである．

また，今回の実装では，read について確実に read ができているかのチェック機構を設けていない．もし，一度で read できなかった場合には，次のループで read を行うようになっている．プロトコルによっては 1 回のリクエストと判定されずに，エラーが起きる可能性がある．対策としては，記述子をノンブロッキングに設定した上で，再度 read を行い読み込めるデータが無いかを確認するなどがある．

5　感想

今回の課題を通して，C 言語の構造体の強力さを感じた．inet_pton では sockaddr_in 型の構造体を sockaddr 型にキャストして渡しているが，これは，それぞれの構造体のメンバ変数をうまく並べて作っているからできている．Java などのオブジェクト指向言語では継承などを用いてこれを提供しているが，C 言語では継承を明示的に使わずとも，構造体の設計者が継承を意識して作るだけで実現できている．メモリ上に何が展開されているかイメージできていればあたりまえのことだが，C 言語の強力さを感じた．

また，getaddrinfo の返り値が連結リストになっているのも興味深かった．対象のサーバが何個のサービスをしているかわからない時でも，動的にメモリを確保することで実現されており面白いと思った．シンプルな考え方であれば，ある程度の大きさの配列を静的に用意しておいて，そこに各構造体へのポインタを格納すれば良いと考えられるが，そうではなく動的に用意されている．連結リストは，知識としては知っているが，正直これまでまともに使ったことがなかったが，こんなところで用いられているのかと思った．連結リストの使い方として一つ知識が増えた．

ヘッダファイルの include について，少し疑問点があった．select のリファレンスを見ると，sys/select.h を include するように記述があったが，netdb.h や stdlib.h を include している場合には sys/select.h を include 指定なくても select を用いることが可能であった．このような場合には sys/select.h は冗長なものとして include に指定しないほうが良いのか，それとも明示的に include をしておいたほうが良いのか至らなかった．

参考文献

[1] W.Richard Stevens(著), Stephen A. Rage(著), 大木 敦雄 (翻訳), “詳解 UNIX プログラミング 第 3 版”, 翔泳社，2014/4/22

[2] W. リチャード・スティーヴンス (著), 篠田陽一 (訳), “UNIX ネットワークプログラミング第 2 版 Vol.1 ネットワーク API: ソケットと XTI”, ピアソンエデュケーション，1998/7/30

[3] W. リチャード・スティーヴンス (著), 篠田陽一 (訳), “UNIX ネットワークプログラミング第 2 版 Vol.2 IPC: プロセス間通信”, ピアソンエデュケーション，2004/2/2

付録 A　プログラム
