情報通信応用実験 ネットワークプログラミング

実験指導書

１．実験課題と手順

課題１は一人で取り組み、課題２～４は二人一組で取り組む。ただし、全体の人数が奇数の場合は、一つのグループは三人とする。

課題 1．ネットワークオペレーションの基礎

まず、基本的なネットワークコマンドについて理解する。班内で相談しても良いが、実験結果は各自で取る。課題２以降はグループで取り組むので、相方の進捗が遅ければ指導する。（※ ただし、実験結果は各自のパソコンで取る。）

課題 1.1　自分の IP アドレス及びマシン名の調査

手順：

- ifconfig コマンドを実行 （/sbin/ifconfig）して、自分のマシンの IP アドレスを調べる。IP アドレスは inet アドレスとして表示されている。

  IP アドレス：　133.10.____.____

- uname コマンドを実行 （uname -n と入力）して，自分のマシンのマシン名を表示する．

  表示内容：　._____

課題 1.2　ping コマンドを用いた、PC 稼働状況の調査

手順：

- ping コマンドのオプションについて調べる。（本研究では Unix 系の ping を用いるが、Unix と Windows 系ではオプションが異なるので注意する。）
- 相手方の PC は 133.10.235.2 ~133.10.235.135 のいずれかの IP アドレスとする。 ping コマンドを入力し、前述の IP アドレスの中から稼働している PC を探す。この時、パケット送信回数は 1 回とする。オプションを指定せずに ping を打つと、強制終了するまで、永遠にパケットを送信し続けるので注意する。シェルスクリプトを書く、若しくは 1 つずつ調べても良い。応答がない場合は、当該 IP アドレスを持つ PC が稼働していない、若しくは存在しないことを意味する。
- 稼働している PC からターゲットを 1 台選び、ターゲット IP アドレスに対し ping コマンドを打つ。送信するパケット数は「4」とする。結果を保存し、各項目の意味を調べる。また、平均応答時間を求める。

課題 1.3　ping コマンドを用いた、PC 稼働状況の調査

手順：

- パケットサイズを指定するためのオプションについて調べる。
- IP アドレス 133.10.105.51 に対し、パケットサイズが６４、１２８、２５６、５１２バイトの ping を各４回ずつ送信する。パケットサイズと平均応答時間の表を作る。

課題 1.4　traceroute コマンドを用いた、他の PC までの経路調査

手順：

- IP アドレス 133.10.105.51 に対し、traceroute コマンドを用いてターゲットとなる PC までの経路を調べる。結果を保存し、各項目の意味を調べる。
- traceroute の仕組みを調べ、ping コマンドだけを用いて traceroute を実装する。実装方法は、シェルスクリプトを書く、若しくはコマンドの列挙により行う。

課題 2．ソケットプログラミング

C 言語によってプロセス間通信を実現するプログラムを作成し、ネットワークプログラミングの基礎について理解する。本課題では、UDP 及び TCP を用いた Echo プログラムを作成する。なお、Echo プログラムとは、サーバがクライアントから送信されてきた文字列をそのまま返信（Echo）するものである。

課題 2.1　UDP による Echo プログラムを実現するクライアント・サーバプログラム

本課題では、UDP を用いた Echo プログラムを作成する。

サーバ（udp_sever.c）の機能：

- ポートをオープンにし、クライアントからの文字列送信を待機する。サーバは永遠に稼働するものとし、無限ループで待機して良い。
- 文字列を受信した際に、１）受信した文字列、２）クライアントの IP アドレス、３）ポート番号をターミナルに表示する。

クライアント（udp_client.c）の機能：

- scan(.) でユーザからの文字列入力を待機し、入力された文字列をサーバに送信する。ループ文を用い、何回でも文字列を入力できるようにする。
- クライアントプログラムを終了するための入力文字列（'q'、'quit' や'exit' 等）を定義し、ループを抜けてプログラムを終了できるようにする。以降、「終了コマンド」と呼ぶ。

手順：

- UDP に基づいた Echo サーバプログラム（udp_server.c）を作成する。
- UDP に基づいた Echo クライアントプログラム（udp_client.c）を作成する。
- 作成したプログラムを用いて、クライアントから文字列を数回送信し、送信した文字列とサーバからエコーされた文字列が一致することを確認する。
- さらに、RTT（Round Trip Time）を計算し、ターミナルに表示する。RTT は、クライアントが文字列を入力してから、サーバからエコーを受信するまでの時間とする。
- ターミナルを２つ立ち上げ、クライアントプログラムを２つ同時に実行し、サーバが複数のクライアントからの文字列を正しくエコーできることを確認する。 **（TA に確認してもらう）**

ヒント：

- クライアントの IP アドレスとポートは、struct sockaddr_in 構造体の sin_addr と sin_port から取得する。また、IP アドレス（sockaddr_in の sin_addr）を文字列に変換するには、arp/inet.h（Unix 系の場合）で定義されている inet_ntoa(.) を用いる。
- 文字化けが起こる場合は、バッファの最後に終端記号 '\0' を入れる等で対処する。また、送受信したバイト数（read() と write() システムコールが返す値）が同じかどうか確認する。
- クライアントからの２回目以降の文字列送信が上手くいかない場合は、クライアントプログラムで送信した文字列のバイト数とサーバからの返信された文字列のバイト数の整合性を確認し、対処する。

課題 2.2　TCP による Echo プログラムを実現するクライアント・サーバプログラム

本課題では、TCP を用いた Echo プログラムを作成する。

サーバ（tcp_sever.c）の機能：

- TCP を用いた Echo サーバでは、複数のクライアントからの接続要求に対処するため、子プロセスを作成して並列処理を行う。（ヒントを参照）
- コネクションを受け付けた際に、クライアントの IP アドレスとポート番号をターミナルに表示する。
- クライアントから文字列を受信するごとに、受信した文字列を表示する。
- 担当しているクライアントプログラムがコネクションを閉じれば、当該子プロセスは終了する。

クライアント（udp_client.c）の機能：

- scan(.) でユーザからの文字列入力を待機し、入力された文字列をサーバに送信する。ループ文を用い、何回でも文字列を入力できるようにする。
- クライアントプログラムを終了するための終了コマンド（'q'、'quit' や'exit' 等）を定義し、ループを抜けてプログラムを終了できるようにする。

手順：

- TCP に基づいた Echo クライアントプログラム（tcp_client.c）を作成する。TCP と UDP では、ソケットの扱い方が異なるため、その違いを理解する。特にソケットへの入出力に注意する。
- TCP に基づいた Echo サーバプログラム（tcp_server.c）を作成する。
- 作成したプログラムを用いて、クライアントから文字列を数回送信し、送信した文字列とサーバからエコーされた文字列が一致することを確認する。
- さらに、RTT（Round Trip Time）を計算し、ターミナルに表示する。RTT は、クライアントが文字列を入力してから、サーバからエコーを受信するまでの時間とする。
- ターミナルを２つ立ち上げ、クライアントプログラムを２つ同時に実行し、サーバが複数のクライアントを処理できることを確認する。**（TA に確認してもらう）**

ヒント：

- サーバ側では、accept()が成功した後に、fork()で子プロセスを作成する。子プロセスはクライアントとの通信を行い、接続終了後にソケット記述子を閉じ、exit()で終了する。
- TCP 通信では、サーバはクライアントごとにソケット記述子が必要であるため、クライアントが終了した際に、サーバ側での当該ソケットを閉じるようにする。これを実現するには、サーバが終了コマンド（'q'、'quit' や'exit' 等）を受信した場合に、ループを抜け、当該ソケットを閉じて、子プロセスを終了する。

課題 3．簡易ファイル送受信プロトコル

TCP を用いたファイル転送プロトコルを設計・実装を行い、アプリケーションプロトコルの開発について理解する。

※　ファイル入出力に関しては、システムコール（open(.) 、 write(.) 、 lseek(.) など）を用いる。その他ライブラリ（fopen 等）などは許可しない。

課題 3.1　簡易ファイル転送プログラムの設計

本課題では、クライアントが任意のファイルをサーバへ送信する簡易ファイル転送プロトコルの設計を行う。トランスポートプロトコルは TCP を用いる。サーバとクライアントのやり取りは図１．を参照する。また、エラー処理（ファイルが存在しない場合に起こるエラー等）は無視して良い。

サーバ（ft_server.c）の機能：

- クライアントを待ち受け、接続があれば、子プロセスを作成する。
- ファイル名、ファイルサイズ、ファイルを順に受信し、それぞれ ACK を返す。

- クライアントから終了コマンドを受信すれば、子プロセスを終了する。

クライアント（ft_cilent.c）の機能：

- scan(.) でファイル名を受け取り、ファイル名、ファイルサイズ、ファイル順にサーバへアップロードする。この時、サーバからそれぞれの ACK を受信する。
- 複数のファイルをアップロードできるように、ループ文でユーザからのコマンドを受けつける。この時、TCP コネクションを切断しない。
- 終了コマンドが入力されると、サーバに終了コマンドを送信し、クライアントプログラムを終了する。
- バッファサイズの上限は６４バイトとする。一度にファイル全体を読み込んで、データを送信しない。従って、ファイルを６４バイトずつ読み込み、送信する。

手順：

- パケットの種類を定義する。クライアントからファイル名、ファイルサイズ、ファイルデータを送信するためのパケットをそれぞれ「fila_name」、「file_size」、「data」、サーバからファイル受信完了をクライアントに知らせる「ACK」の４種類のパケットが必要である。
- サーバとクライアントの状態遷移図を、それぞれ図２と図３に示す。それらを参考にして、サーバプログラムとクライアントプログラムがどのように動くか考察する。
- サーバプログラムとクライアントプログラムを書く。（同一 PC で実験を行う場合、クライアントプログラムが送信したファイルをサーバプログラムが上書きしないように気をつけること）
- diff コマンドを用いて、ファイルが正しく送信できるか確認する。
- クライアントから、サイズの異なるファイル（1.jpg, 2.jpg, 3.jpg, 4.jpg, 5.jpg）を計３回送信し、それぞれ RTT の平均値を計算する。横軸をファイルのサイズ、縦軸を RTT 平均値として、グラフを作成する。なお、各画像ファイルのサイズは、それぞれ 1M、2M、3M、4M、5M である。
- 複数のクライアントが同時にサーバにアクセスできることを確認する。また、各クライアントが同じ TCP コネクションで他のファイルも送信できることを確認する。**（TA に確認してもらう）**

ヒント：

- ファイル入出力は、open(.) と write(.) を用いる。ファイルサイズの取得には、lseek(.) 、ファイルの存在判定には、 access(.) を用いる。ファイル入出力関連のシステムコールに関しては、sys_calls.tar.gz 内のソースコードを参考にする。

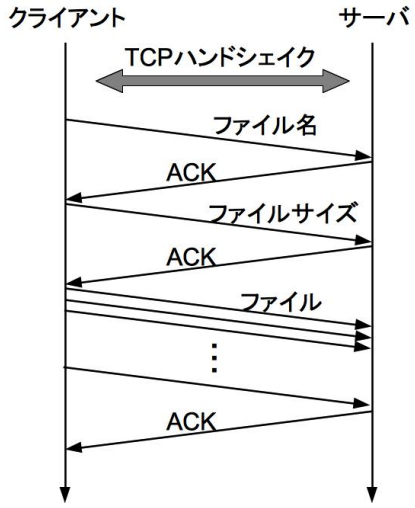
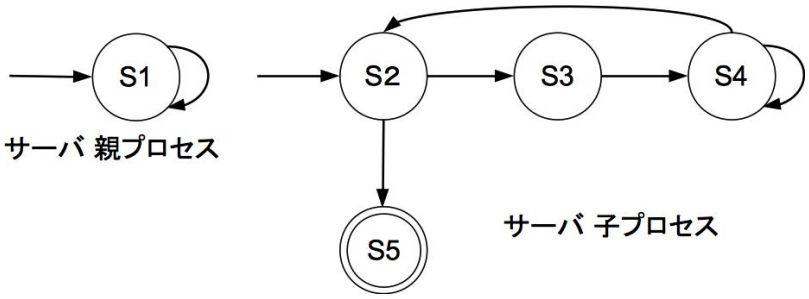


図１．時間ダイアグラム

補足：

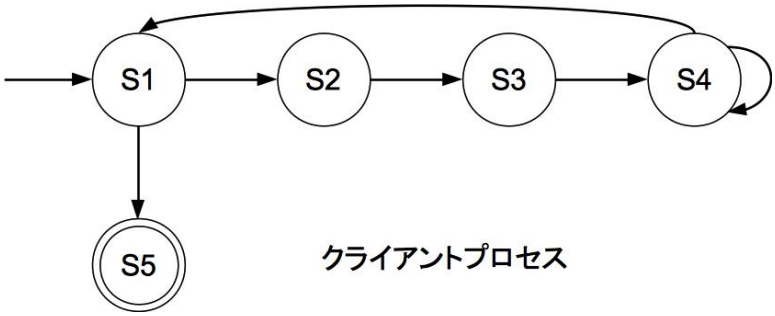
クライアント側	サーバ側
１．	１．read(.) でファイル名を待機
２．write(.) でファイル名を送信	２．
３．read(.) で ACK を待機	３．read(.) でファイル名を受信
４．	４．write(.) で ACK を送信
５．read(.) で ACK を受信	５．read(.) でサイズを待機

以下、ファイルサイズ、ファイルデータに関しても同様の動作を繰り返す。



状態遷移
S1→S2：クライアントからの接続要求
　　子プロセスを作成する
S2→S3：ファイル名受信、ACK 送信
S3→S4：ファイルサイズ受信、ACK 送信
S4→S4：ファイル受信
S4→S2：ファイル受信完了、ACK 送信
S1→S5：終了コマンド受信

図２．サーバプログラムの状態遷移図



状態遷移
S1→S2：ユーザからファイル名入力を検知
　　ファイル名送信
S2→S3：ACK 受信（ファイル名受信）
　　ファイルサイズ送信
S3→S4：ACK 受信（ファイルサイズ受信）
　　ファイルデータ送信
S4→S4：ファイルデータ送信
S4→S2：ACK（ファイル受信完了）
S1→S5：終了コマンド入力を検知

図３．クライアントプログラムの状態遷移図

課題 3.2　簡易ファイル共有プログラムの設計

本課題では、コマンドラインベースの簡易ファイル共有プロトコルの設計を行う。クライアントとサーバともに、課題 3.1 より複雑な機能もつが、 符号化・復号化を組み入れ、柔軟で保守性の高いプログラムを作成する。また、エラー処理（存在しないファイルをリクエストした場合に起こるエラー）などは無視して良い。

サーバ（share_server.c）の機能：

- 課題 3.1 を拡張し、クライアントからのリクエストに対し、ファイルの送受信機能を実現する。

クライアント（share_cilent.c）の機能：

- ターミナルからのコマンドを受け付け、ファイルの送受信機能を実現する。
- バッファサイズの上限は６４バイトとする。

手順：

- パケットフォーマットを定義する。 パケットの定義は、「フィールド名」、「ビット長」、「オフセット」からなる。（ヒントを参照）
- クライアントとサーバの状態遷移を考える。また、クライアントとサーバ間のやり取りを示す、タイアグラムを書く。
- プロトコルを設計した時点で、担当教員若しくは TA に確認してもらう。
- サーバプログラムとクライアントプログラムを書く。
- 作成したプログラムを実行し、ファイルが正しく送受信できるか確認する。diff コマンドを用いて、ファイルが正しく送信できるか確認する。また、クライアントプログラムを２つ以上起動させ、一方のクライアントがアップロードしたファイルを他方のクライアントがダウンロードできるか確認する。 **（TA に確認してもらう）**

ヒント：

- クライアントがサーバからファイルをダウンロード、若しくはファイルをアップロードするための「request」コマンド、ファイルをアップロードする「data」を定義する。サーバ側も同様に「ACK」や「reply」や「data」を定義する。課題 3.1 と異なり、 各パケットタイプを識別するためのフィールドが必要である。また、今回は終了コマンドも文字列として扱わず、フィールドで識別する。
- パケットフォーマットの例を図４に示す。オフセットをビット単位にすると複雑になるので、バイト単位で良い。また、各パケットは固定長でも可変長でも良い。

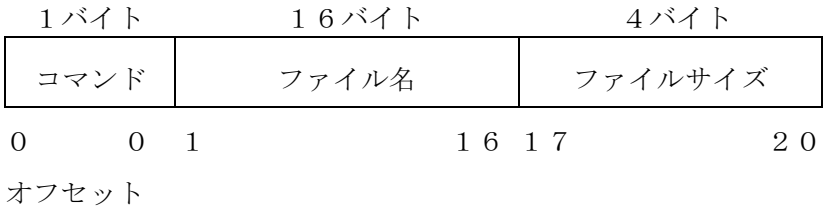


図４．パケットフォーマットの例

課題 4．アプリケーション層における伝送制御プロトコルの実装

アプリケーション層ベースの伝送制御プロトコルの設計・実装を行い、アプリケーションプロトコルの開発について理解する。

課題 4.1　アプリケーション層における伝送制御プロトコル（Stop-and-Wait 若しくは Go-Back-N）の実装

サーバ（sw_server.c / gbn_sever.c）の機能：

- 課題 3.1 と同様の機能を持つ。
- クライアントを待ち受け、接続があればファイルを受信する。受信したパケットに対し、P%（P は０～１００）の確率でパケット内のデータをファイルに保存して ACK を返す。（１−P）%の確率で、パケットを破棄する。

クライアント（sw_cilent.c / gbn_client.c）の機能：

- 課題 3.1 と同様の機能を持つ。
- サーバからの ACK がなければ、送信したパケットは破棄されたと判断し、伝送制御プロトコルに従い、再送を行う。
- バッファサイズは４０９６バイト、タイムアウトを３秒とする。ファイルサイズが大きい場合は、適時、バッファサイズとタイムアウトを調整して良い。

手順：

- パケットを定義し、ダイアグラムと状態遷移を考える。
- サーバプログラムとクライアントプログラムを作成する。
- 作成したプログラムを実行する。サーバの受信確率を９０%、９２%、９４%、９６%、９８%、１００%として、クライアントからファイル（ファイルサイズは自由）を送信し、RTT を計算する。横軸を受信確率 P、縦軸を RTT としてグラフを作成する。
- 作成したプログラムが正しく動作するか確認する。**（担当教員に確認してもらう）**

２．注意事項

- システムコール及び関数の仕様については、参考資料若しくは Google 等で調べる。
- プログラムのバグに関する質問は一切受け付けない。また、TA もバグ取りに関する補助は一切しない。デバッグの仕方（バグが起こった箇所の探し方など）は別途、教える。
- 課題２と３は TA にプログラムを確認してもらい、課題４は担当教員が直接確認を行う。応用実験の進捗状況はプログラム動作確認時に記録するため、必ず TA に確認してもらう。（課題１は確認を取らなくても良い。）
- 作成したプログラムのソースコードは、必ずバックアップを取る。「PC の故障などによって、ソースコードを失くした」場合などの特別処置は一切しない。
- レポート提出は、「３．レポート提出要項」の要領で書く。従わない場合は、採点しない。

３．レポート提出要領

レポート構成：

レポートは各自、個別に書く。レポートを書くときのフォントサイズは 10.5 とする。レポートは、次の要領で構成する。

- 表紙
  タイトル，学修番号，氏名，実験日，共同実験者
- 各課題の報告（詳細は、下記参照）
- 参考文献リスト
- 作成したソースコード（レポートの最後に付録として提出）

各課題の報告：

課題１は、指定のフォーマットに書き、課題２～４は、書式は自由とする。ただし、各課題がどこに書かれているか一目で分かるように、見出しをつける。

課題２～４は次の要領でまとめる。各課題のページ数制限に注意する。また、図や表はページ数制限に入れるが、ソースコードは付録として扱うためページ数制限に入れない。

- 課題 2.1（２ページ以内）
  １）作成したプログラムの説明、２）実行結果の説明、３）RTT の計算方法の説明・考察を書く。
- 課題 2.2（２ページ以内）
  １）作成したプログラムの説明、２）実行結果の説明、３）RTT の計算方法の説明・考察を書く。特に UDP との違いを書く。
- 課題 3.1（３ページ以内）
  １）設計したプロトコルの説明（パケットフォーマット及びパケットを受信した時の動作）、２）ファイル送信実験の説明、グラフの説明・考察を書く。
- 課題 3.2（４ページ以内）
  １）設計したプロトコルの説明（パケットフォーマット及びパケットを受信した時の動作を、パケットフォーマット図、状態遷移図、ダイアグラムを用いて説明する）、２）ファイル送信実験の説明、グラフの説明・考察を書く。
- 課題 4.1（３ページ以内）
  １）設計したプロトコルの説明（パケットフォーマット及びパケットを受信した時の動作をパケットフォーマット図、状態遷移図、ダイアグラムを用いて説明する）、２）プログラムの仕様を書く。

注意事項：

- 付録はアルファベットで番号を付ける。例）付録 A. UDP クライアントプログラム、など。
- ソースコードは付録として添付する。ソースコードは図として扱い、当該課題から参照する。例）サーバプログラムを図１に示す、など。
- 図、表には番号とキャプション（説明）をつける。（図は下に、表は上につける。）
- 図、表には通し番号をつけ、適時参照する。

- 参考にした文献，web ページ等は，必ず最後の参考文献リストに掲載する。指導書と参考資料は参考文献に掲載不要。

レポート再提出：

- 第 1 回目に提出したレポートに、赤ペンで印がついてある者は、レポート再提出対象者となる。
- 再提出の際には、１）修正した新しいレポート、２）一回目のレポートの両方を提出する。

４．採点基準

- 最低条件として課題 2 を終え、レポートの出来が良好であれば、ネットワークプログラミングの成績を「４」とする。さらに課題を進めレポートの出来が良好であれば点数を加算する。
- 説明・考察が不十分、若しくはレポートの出来が悪い場合は、減点対象となる。
- ソケット及びファイル入出力は、システムコールを用いて作成する。外部ライブラリを用いた場合は減点対象とする。
- その他、遅刻・欠席は内規に従う。

５．参考資料と補足用のソースファイル

参考資料（コースのホームページからダウンロードする）：

- networking_instruction_v2_0.docx　：このファイル
- networking_sup.doc　：情報通信応用実験 参考資料
- networking_problem1.docx　：課題 1 をレポートにまとめるときのフォーマット
- networking_tutorial.pdf　：チュートリアル用のスライド
- 0.jpg、1.jpg、2.jpg、3.jpg、4.jpg、5.jpg：データ送信に用いるサンプル画像ファイル

補足用ソースファイル：

- ex_shell.tar.gz　：シェルスクリプトのサンプルコード
- daytime.tar.gz　：ソケットプログラミングのサンプルコード
- sys_calls.tar.gz　：システムコールのサンプルコード（課題 3.1 及び課題 4 で用いる参考プログラム）
- ex_codec.tar.gz　：符号化・復号化のサンプルコード（課題 3.2 で用いる参考プログラム）
- ex_gdb.tar.gz　：GDB 使い方のサンプルコード

--

作成日：２０１５年2月７日

更新日：２０１６年４月５日

更新日：２０１７年４月１日

作成者：助教・酒井 和哉