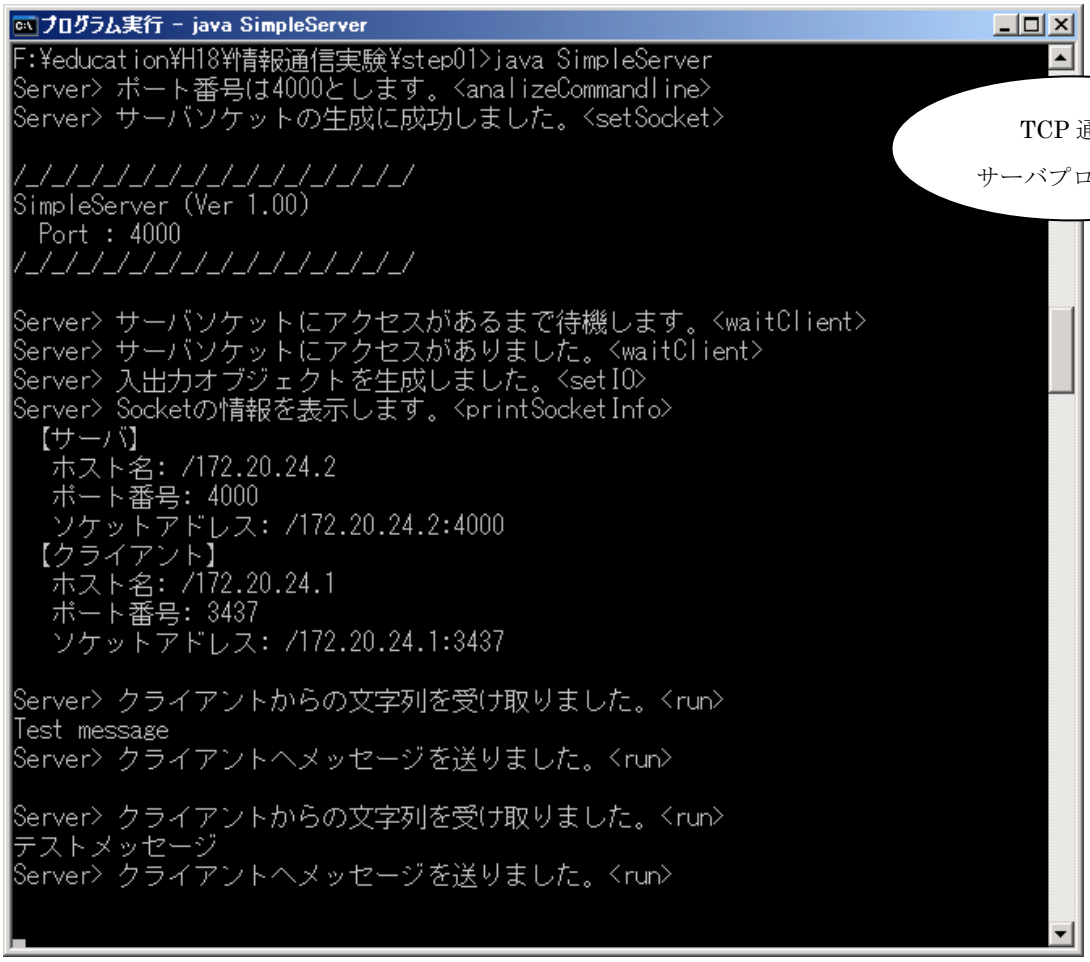
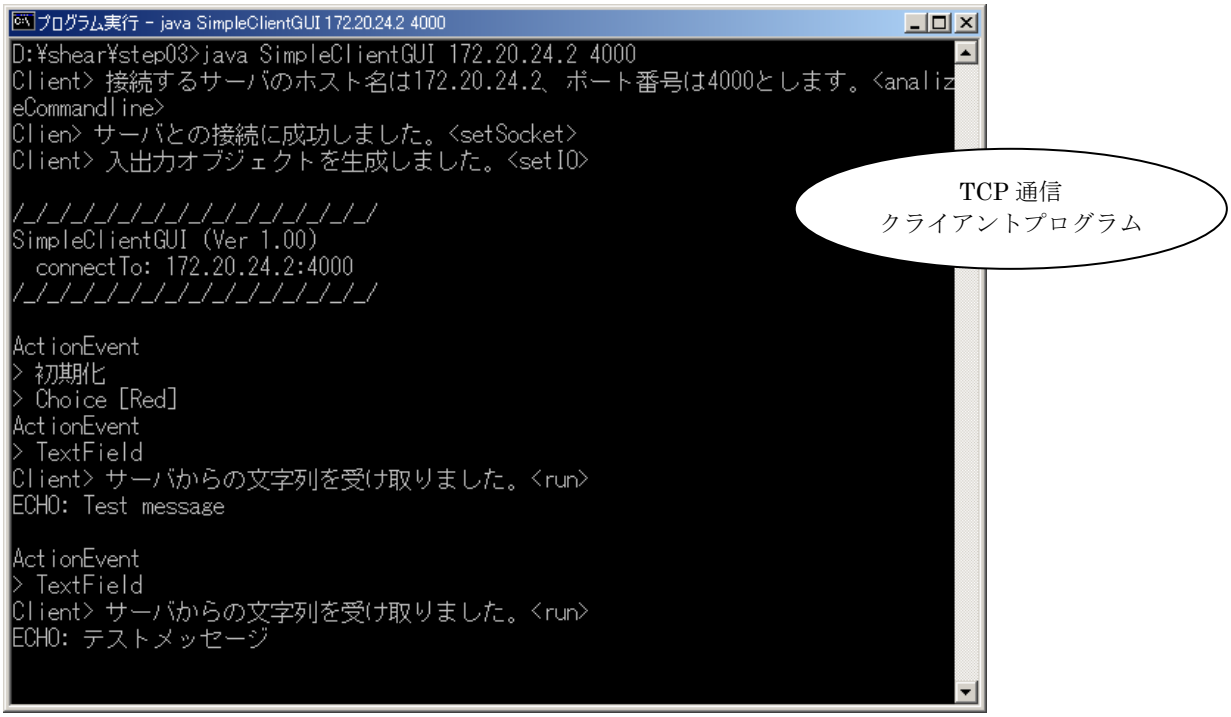
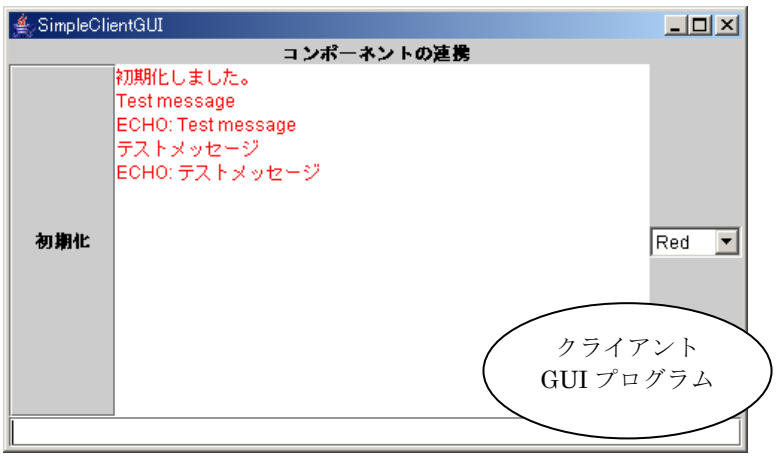
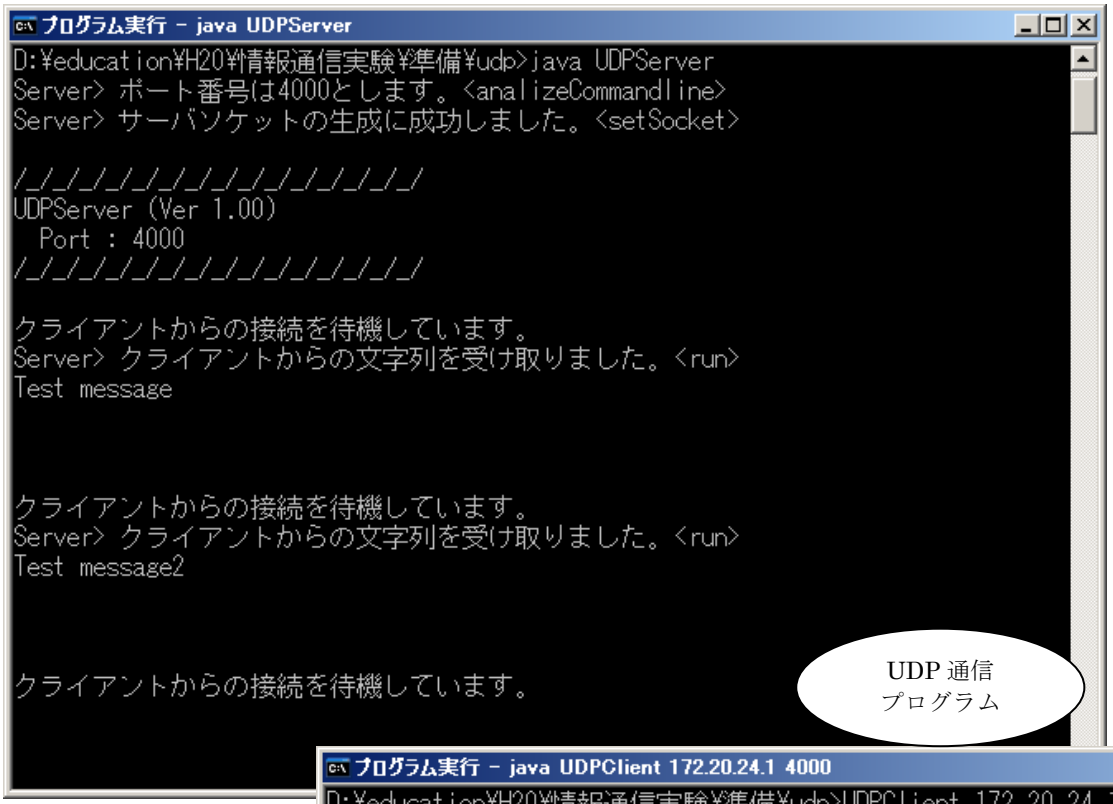
情報通信実験

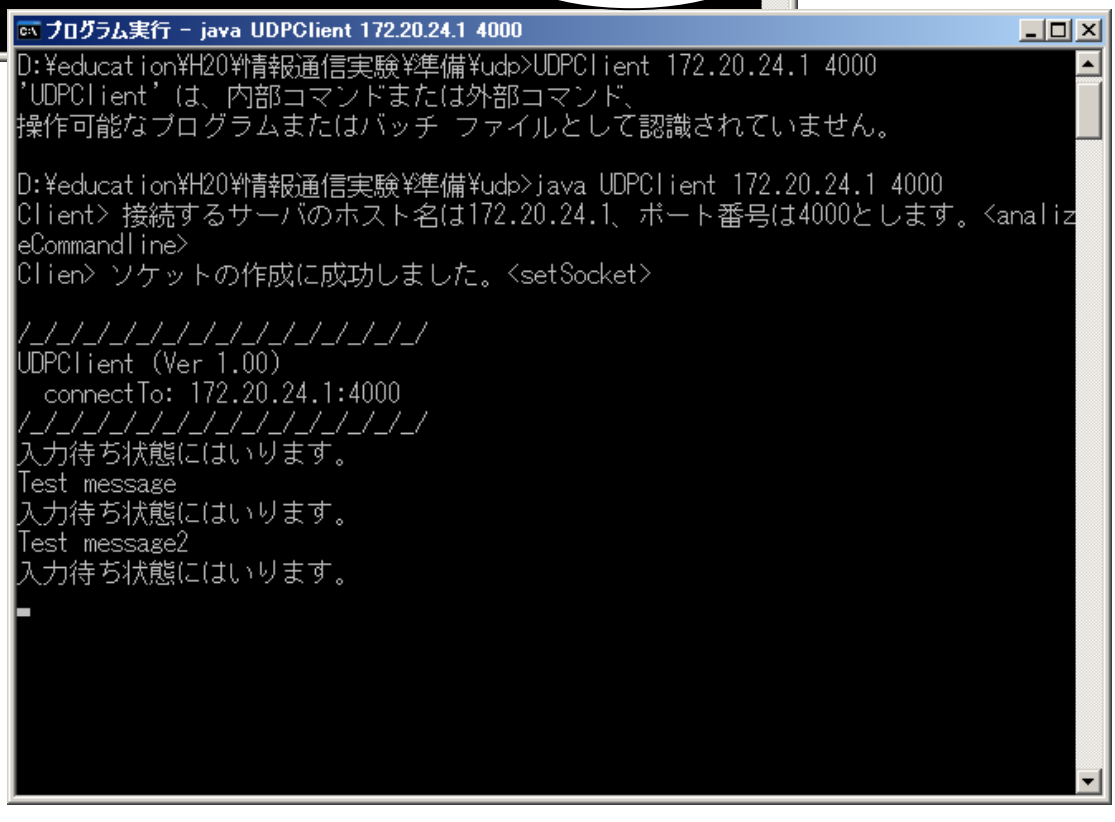
【JAVA 言語による TCP/IP 通信】






SimpleClientGUI
コンポーネントの連携
初期化しました。
Test message
ECHO: Test message
テストメッセージ
ECHO: テストメッセージ
初期化
Red
クライアント
GUI プログラム


プログラム実行 - java UDPServer
D:¥education¥H20¥情報通信実験¥準備¥udp>java UDPServer
Server> ポート番号は4000とします。<analizeCommandline>
Server> サーバソケットの生成に成功しました。<setSocket>
/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/
UDPServer (Ver 1.00)
Port : 4000
/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/
クライアントからの接続を待機しています。
Server> クライアントからの文字列を受け取りました。<run>
Test message
クライアントからの接続を待機しています。
Server> クライアントからの文字列を受け取りました。<run>
Test message2
クライアントからの接続を待機しています。
UDP 通信
プログラム


プログラム実行 - java UDPClient 172.20.24.1 4000
D:¥education¥H20¥情報通信実験¥準備¥udp>UDPClient 172.20.24.1 4000
'UDPClient' は、内部コマンドまたは外部コマンド、
操作可能なプログラムまたはバッチ ファイルとして認識されていません。
D:¥education¥H20¥情報通信実験¥準備¥udp>java UDPClient 172.20.24.1 4000
Client> 接続するサーバのホスト名は172.20.24.1、ポート番号は4000とします。<analiz
eCommandline>
Clien> ソケットの作成に成功しました。<setSocket>
/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/
UDPClient (Ver 1.00)
connectTo: 172.20.24.1:4000
/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/
入力待ち状態にはいります。
Test message
入力待ち状態にはいります。
Test message2
入力待ち状態にはいります。

1. 実験の目的

ソケットを用いたクライアント／サーバアプリケーション作成を通して、TCP／IP 通信、ならびに、暗号化技術に関する理解を深める。

2. 背景

アプリケーションプログラムとネットワークモジュールとの間の仲立ちをするインタフェースの代表に、ソケット（Socket）がある。ソケットは、アプリケーションが使用している IP アドレスやポート番号を管理したり、パケットの送受信に必要なバッファの管理したりするという、ネットワークモジュールが動作するために必要な処理を行う概念といえる。また、ソケットによる通信を実現するにあたり、トランスポート層のプロトコルとして、信頼性を保証しないコネクションレス型の UDP ではなく、信頼性を保証するコネクション型の TCP を採用することで、2 つのホスト間での信頼性のあるデータ交換を行うことが可能となる。

3. 基本的な原理

TCP/IP 通信における基本的な原理として、TCP/IP、パケット、ソケットについて以下に説明する。

3.1. TCP／IP

TCP/IP は、「いろいろな通信ハードウェアを利用できるように設計された通信ソフトウェア」である。この TCP/IP の仕組みのおかげで、様々な異なった仕組みの OS やハードウェアや通信経路を使っていてもインターネットによるデータの交換を行えている。TCP/IP に対応している具体的なアプリケーションには、HTML（WWW）をはじめ、SMTP（電子メール）、ストリーミング（動画配信）、Telnet（遠隔ログイン）、FTP（ファイル転送）、SNMP（ネットワーク管理）などがある。

TCP/IP は、具体的には TCP（Transmission Control Protocol）と IP（Internet Protocol）の 2 つのプロトコルを中心とするプロトコルの集まりである。さらに、TCP/IP を用いた通信では、通信の処理を、①アプリケーション層（図 1 中の「アプリケーション」）、②トランスポート層（図 1 中の「TCP」）、③インターネット層（図 1 中の「IP」）、④ネットワーク層（図 1 中の「Ethernet」）、の 4 つの処理段階

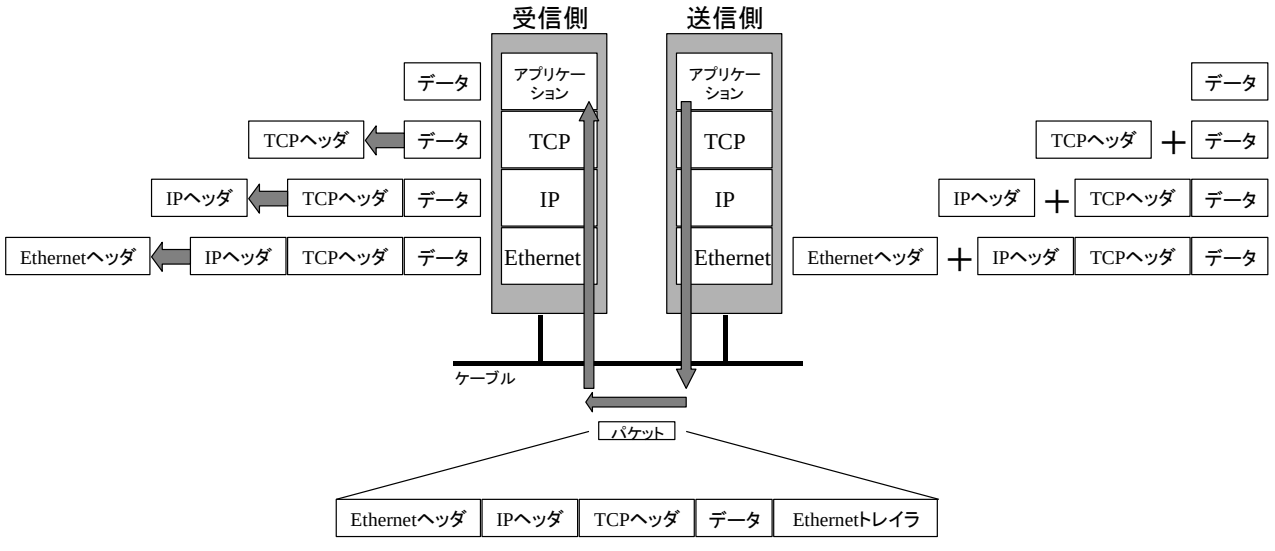


図 1 ：TCP/IP による通信の概要

に分けて行っている。

図１に TCP/IP による通信の概要を示す。TCP/IP による通信の基本は、クライアントサーバモデルと呼ばれるサービス形態となっている。送信側となるホスト（クライアント／サーバ）からデータが送信される際には、まず、アプリケーション層にてデータが作られ、さらに OS 内の処理として上述の 3 つの下位層に順々に渡される。各層においてヘッダが付加された後に、受信側のホストへ向けての送信がスタートする。データにヘッダを付加させたものをパケットと呼び、送り先のホストがデータを受け取る際には、逆に、下位層から順々に、ヘッダの確認が行われる。

3.2. パケット

TCP/IP を用いた通信では、パケットの構造が決まっており、この決められた構造のことをパケットフォーマットと呼ぶ。上述したように、TCP/IP におけるパケットフォーマットは、図２のような形をしている。先頭に Ethernet ヘッダが付けられ、次に IP ヘッダ、TCP ヘッダが付いた後に、実際にアプリケーションが利用するデータ（TCP データ）が格納されている、また、最後尾には Ethernet トレイラが付けられている。パケットの中身は図２に示すような構造となっているが、実際のネットワークを流れている最中においては、2 進数で直線的な形をしている。しかしながら、図３のように折り返して表現するのが一般的といえる。

図２：パケットフォーマット

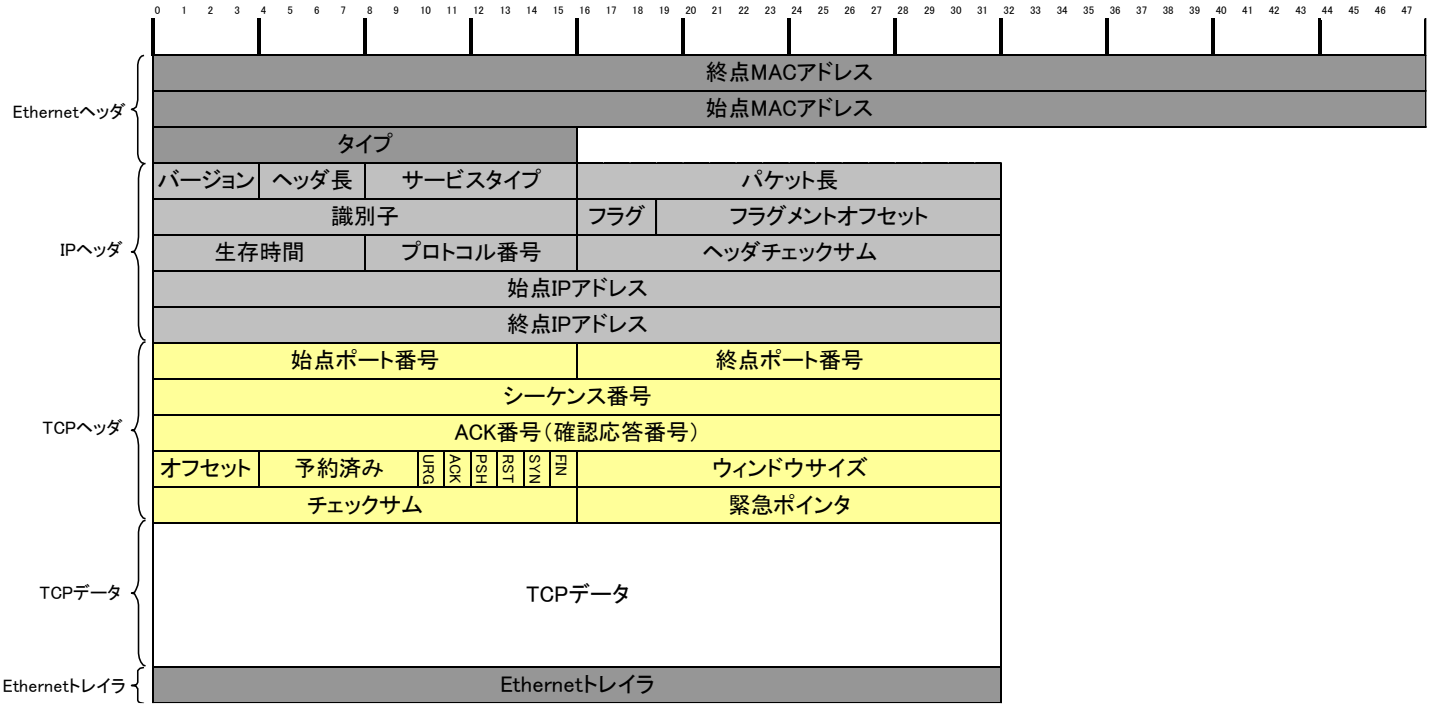


図３：パケットヘッダの表記

3.3. ソケット

ソケットには、サービスを提供するプログラム（サーバ）と、サービスを要求するプログラム（クライアント）とがある。ソケットによる通信を行うためには、まず、サーバが、相手を特定しない受動的なオープンを行っていなければならない。この受動的なオープンをしている状態を待機（listen）と呼ぶ。サーバがこの待機の状態にあると、クライアントは相手を指定した能動的なオープン（connect）を行うことができる。このクライアントの connect に対して、サーバが accept システムコールを実行し、接続要求を受諾する。このようにソケットが結合したことを、コネクションの確立と呼ぶ。コネクションの確立後は、相互にメッセージの送受信が可能となる。通信を終了するときには、close システムコールを使用する。図４は、ソケットによる通信の流れの概要を示している。

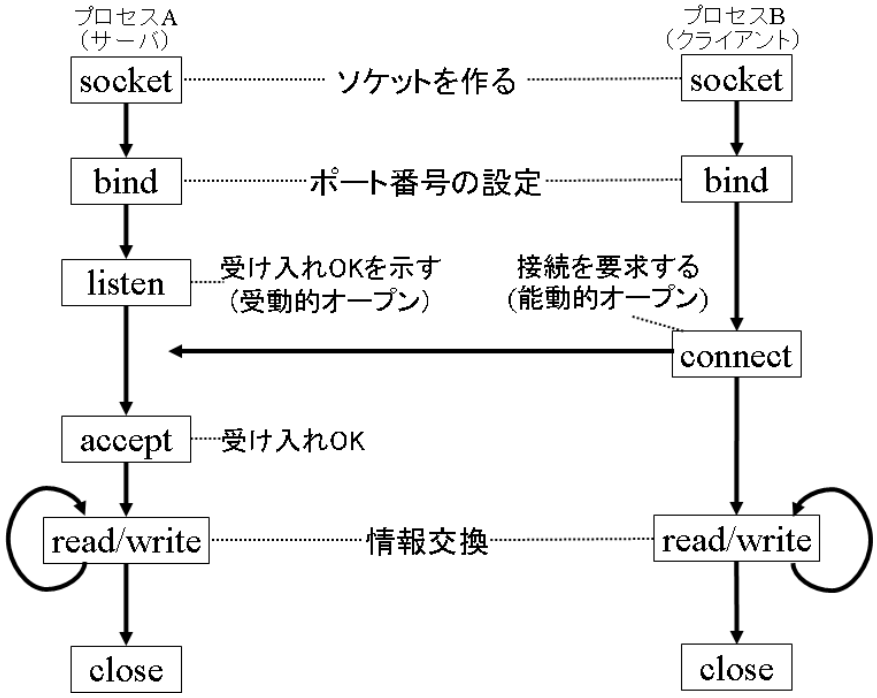


図４：ソケットを用いた通信の概要

3.4. 暗号化技術

暗号化技術は、パケット盗聴、データの改ざん、ユーザなりすまし、などの様々なインシデントから個人、組織、社会を守るための技術である。前述の TCP/IP には、通信の内容を暗号化する機能はなく、通信内容をそのままの形（平文）で送受信する。そのため、通信経路の途中の機器などから、パケットを観測するば、容易に通信内容を知ることが可能である。本節では、文字列のデータを別の文字列に置き換える暗号化技術の一種「Base64」と、データ通信の際に鍵を用いることで鍵の所有者のみに暗号化・復号を許す通信方式の一種「共通鍵暗号方式」について説明する。

3.4.1. Base64

Base64 は、文字列をバイナリーデータ（2 進数の列）とみなし、6bit ごとに文字定数に置き換える暗号化技術である。Base64 は、単に暗号化としてだけではなく、通信においてマルチバイトやバイナリーデータを送れない場合に、送信可能な文字へ変換する目的で利用される。例えば、MIME 規格の電子メールなどでの利用がそれである。

3.4.2. 共通鍵暗号方式

共通鍵暗号方式は、情報の暗号化・復号に共通の鍵である「共通鍵」を用いる暗号方式である。データ通信に用いる際には、情報の送信側と受信側とで同じ共通鍵を持っておく必要がある。そのため、鍵を共有する手段や、鍵の管理が非常に重要となる。処理の流れを図 5 に示す。

また、共通鍵暗号方式の特徴として、暗号化・復号にかかる計算負荷が軽い。そのため、大量データの通信には適した方式といえる。本実験では、共通鍵暗号方式の中で代表的な共通鍵暗号化アルゴリズムである AES (Advanced Encryption Standard) を用いる。AES は、WPA2 などの無線 LAN の通信、SSL/TLS などの電子メールアプリなど、身近な通信機器や技術の中で用いられている。

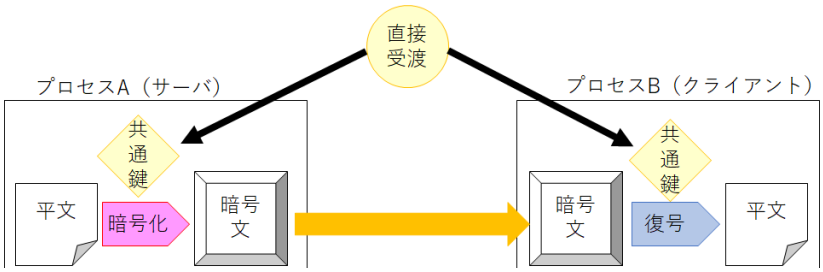


図 5 ：共通鍵暗号方式の概要

4. 実験に用いる具体的技術の概説

本実験のアプリケーション開発は Java 言語を、作成したプログラムの実行確認に 2 つのソフトウェアを用いる。それぞれについて簡単に説明する。

4.1. Java 言語

本実験のアプリケーション開発において使用する主要なクラスは以下の 5 つである。それぞれについて簡単に説明する。（記述例として示しているプログラムのコードの変数名や宣言方法は、本演習と異なっているものあるため、演習時には適宜修正して用いること。）

① ServerSocket クラス

ServerSocket クラスは、サーバソケットを実装しているクラスである。サーバソケットという概念は、ネットワークを介して要求が送られてくるのを待ち、接続要求があった際には、その要求に基づいた処理を行うソケットのことを指す。

ServerSocket クラスのオブジェクト生成においては、ポート番号を選択する。使用するポート番号は標準サービスでは使わないものを利用する。ここではポート番号として 4000 番を使用する。コンストラクタの記述は以下となる。

```
ServerSocket s = new ServerSocket(4000);
```

上述のコードは、ソケットを生成し TCP とアプリケーションの間の通信経路を作る処理と、ポート番号を指定する処理との両方を行うことを示している。別けて記述する場合には、以下のように記述する。(InetSocketAddress クラスは IP ソケットアドレス(IP アドレス + ポート番号)の情報を管理するクラス。)

```
ServerSocket s = new ServerSocket();
s.bind(new InetSocketAddress(4000));
```

次に、上述で作成した ServerSocket クラスのオブジェクトを使って、クライアントからの接続要

求を待つ。この処理は、ServerSocket クラスのメソッド accept()を用いる。accept()メソッドの実行後は、指定のポートに接続するまで無期限に待機することとなる。接続要求があった後は、接続要求をしてきたクライアントとの通信を行うためのソケットを準備する、つまり、Secket クラスのオブジェクトを生成する。記述は以下のようになる。s.accept()は、前述した ServerSocket のオブジェクト s による accept()メソッドの呼び出しである。s.accept()の実行では、クライアントからの接続があった後、Socket クラスのオブジェクトを戻り値として返し、左辺の Socket クラスのオブジェクト c_socket に代入される。

Socket c_socket = s.accept() ;

ServerSocket クラスを利用する際には、様々な例外処理が不可欠である。例えば、コンストラクタや accept()メドッドにおいては、ソケットの生成中に入出力エラーが発生した場合に発行される IOException への対応をしなければならない。

② Socket クラス

Socket クラスはクライアントソケット（単に「ソケット」とも呼ぶ）を実装しているクラスである。ソケットという概念は、2 つのマシン間の通信の両端に位置し、マシンの OS やハードウェア、通信経路には依存しない仕組を指す。

Socket クラスのオブジェクト生成においては、リモートコンピュータ（つまり、接続希望先）のホスト名（または、IP アドレス）とポート番号を選択する。コンストラクタの記述は以下となる。

Socket s = new Socket(“www.kurume-nct.ac.jp”, 4000);

上述のコードは、久留米高専の Web サーバコンピュータにおける 4000 番ポートとの通信を行うためのソケットを生成している。また、このコード一行で、TCP とアプリケーションの間の通信経路を作る処理と、指定したリモートコンピュータへの接続要求を行う処理との両方を行うことを示している。

また、リモートコンピュータ（つまり、接続希望先）からデータを受信するローカルアドレスとローカルポートを指定する場合のコンストラクタの記述は以下となる。

Socket s = new Socket(“www.kurume-nct.ac.jp”, 4000 , InetAddress.getLocalHost(), 5000);

上述のコードは、久留米高専の Web サーバコンピュータにおける 4000 番ポートとの通信を、5000 番ポートで受信するソケットを生成している。

Socket クラスを利用する際にはおいても ServerSocket クラスと同様に、様々な例外処理が不可欠である。例えば、ホストの IP アドレスを判定できなかった場合に発行される UnknownHostException、ソケットの生成中に入出力エラーが発生した場合に発行される IOException への対応をしなければならない。

③ BufferedReader クラス

BufferedReader クラスは Reader クラスを拡張したクラスである。文字、配列、行をバッファリングすることによって、文字型入力ストリームからテキストを効率良く読み込む。

入力ストリームに対して、この BufferedReader を利用する際には以下のように使う。

```
BufferedReader bf = new BufferedReader( new InputStreamReader( InputStream クラスのオブジェクト ) );
```

テキストファイルに対して、この BufferedReader を利用する際には以下のように使う。

```
BufferedReader bf = new BufferedReader( new FileReader( ファイル名 ) );
```

このオブジェクトは以下のように使う。

```
String str = bf.readLine();  // 1 行単位の受信。
```

④ PrintWriter クラス

PrintWriter クラスは Writer クラスを拡張したクラスである。フォーマットされたオブジェクトの表現をテキスト出力ストリームに出力します。このクラスでは、Object クラスを含む全ての型についての print()メソッドと println()メソッドをサポートしているため、特に重宝される。

出力ストリームに対して、この PrintWriter を利用する際には以下のように使う。

```
PrintWriter pw = new PrintWriter( new OutputStreamWriter( OutputStream クラスのオブジェクト ) );
```

テキストファイルに対して、この PrintWriter を利用する際には以下のように使う。

```
PrintWriter pw = new PrintWriter( new BufferedWriter( new FileWriter( ファイル名 ) ) );
```

このオブジェクトは以下のように使う。

```
String str = "ABC";
pw.println( str );  // 1 行（改行付き）の送信。
pw.flush( );  // バッファのフラッシュ（必須）
```

⑤ Thread クラス

Thread クラスは、アプリケーションの並列実行を実現するクラスである。並列実行を実現する方法の 1 つとして、並列実行したい処理を持っているクラスを、Thread クラスのサブクラスと宣言する（つまり、Thread クラスを継承させる）方法がある。継承したクラスにおいて、Thread クラスの持っている run()メソッドをオーバーライドし、その run()メソッド内に、並列実行したい処理を記述する。スレッドの起動(run()メソッドの実行)は、start()メソッドによって行う。

4.2. ネットワークアプリケーション開発に用いるソフトウェア

① Telnet

Telnet（Telecommunication network protocol）は、TCP/IP 階層のアプリケーション層に区分されているプロトコルの一つである。Telnet は、TCP/IP を用いた通信を行い、遠隔ログインを実現するアプリケーションの一つである。ネットワークアプリケーションの開発においては、クライアントプログラムの代用としてサーバプログラムのデバッグツールとして利用することができる。本実験でも、Step01 においてサーバプログラムを作成した後、その動作確認のために用いる。この Telnet は、WindowsOS に標準実装されているので、コマンドプロンプト上から利用することができる。

② Wireshark

Wireshark ネットワーク上を流れるデータをモニタリングするネットワークアナライザツールの一つである。10 年前は Tcpdump コマンドなどにより調べていたパケットを、Wireshark では自動での収集・解析することが可能となっている。

5. 実験準備

以下の手順にて、実験の準備を行う。

① クラスやメソッドに関して検索するための準備として、「Java Platform Standard Edition API 仕様」の HP を確認する。
② 圧縮ファイルをダウンロードし、C ドライブに展開する。
③ 展開してできたフォルダの下位フォルダ「ans」と「exe」があることを確認する。
④ フォルダ「ans」からサンプルプログラムの実行を行う。
⑤ 実験は、フォルダ「exe」の各課題フォルダ(Step01～Step06)に格納されている資料を用いて行う。
⑥ 各課題用の動画を、必ず事前に確認しておくこと。

6. 実験課題

ソケットを用いた通信を行うプログラムの作成を行う。尚、プログラミングは Java 言語を使用する。具体的な実験課題を以下に示す。

課題①： TCP 通信のためのサーバプログラムの作成

"SimpleServer.java" を修整・追記し、サーバプログラムを完成させよ。動作確認は、telnet を用いて行うこと。確認事項として少なくとも、以下の点については調べよ。

- ポート番号を指定せずにプログラムの実行
- ポート番号を指定して(プログラム内部で指定した値以外)プログラムの実行
- 複数のクライアント(telnet)のアクセス
- 半角文字の送信・受信
- 全角文字の送信・受信
- 受信内容「bye」での動作
- 同じポート番号指定のサーバプログラムを複数実行

プログラムが完成したら、上述の確認事項のチェックを教員の前で行うこと。

課題①のレポートの実験結果としては、ソースコードの要点のみを説明すること。**ソースコードそのものの記載は不可とする。**また、実験結果として上述の**確認事項を表にまとめ記述すること。**独自の確認事項があれば記述すること。実際にプログラムを動作させた画面をレポートに添付する必要はない。

課題②： TCP 通信のためのクライアントプログラムの作成

"SimpleClient.java" を修正・追記し、クライアントプログラムを完成させよ。動作確認は、

課題①にて作成した SimpleServer.java によって行うこと。確認事項として少なくとも、以下の点については調べよ。

- 指定したホストとポート番号へのアクセス
- 半角文字の送信・受信
- 全角文字の送信・受信
- 送信内容「bye」での動作
- 同ホスト・同ポートへの複数のクライアントプログラムの実行

プログラムが完成したら、上述の確認事項のチェックを教員の前で行うこと。

課題②のレポートの実験結果としては、ソースコードの要点のみを説明すること。**ソースコードそのものの記載は不可とする。**また、実験結果として上述の**確認事項を表にまとめ記述すること。**独自の確認事項があれば記述すること。実際にプログラムを動作させた画面をレポートに添付する必要はない。

課題③： クライアントプログラムの GUI 化

既存の GUI プログラムは、“Main.java”と“GUIPanel.java”の２つのプログラムで構成されている。これら２つのプログラムと、課題②で作成した“SimpleClient.java”を基に、クライアントプログラムの GUI 化を行う。オブジェクト指向プログラミングの「継承」を活用し、プログラムを完成させよ。動作確認は、課題①にて作成した SimpleServer.java によって行うこと。確認事項として少なくとも、以下の点については調べよ。

- 指定したホストとポート番号へのアクセス
- 半角文字の送信・受信
- 全角文字の送信・受信
- 送信内容「bye」での動作
- 複数のクライアントプログラムの実行

プログラムが完成したら、上述の確認事項のチェックを教員の前で行うこと。

課題③のレポートの実験結果としては、クラス間の関係について説明すること。**ソースコードそのものの記載は不可とする。**

課題④： Wireshark を用いた TCP 通信のパケット内容の観測と解析

Wireshark を用いて PC が送受信したパケットの内容の観測と解析をせよ。実験の際には 2 台の PC を用いて、サーバ・クライアントの接続を行うこと。観測と解析を行うパケットは以下の 2 つとする。指定の用紙を受け取りレポートに添付すること。

課題④－１：コネクションの管理に伴うパケットの観測と解析

以下の TCP の通信におけるコネクションの管理に伴うパケットを確認することができる。

1. コネクションを確立させる際に「(クライアントからの)確立要求」を行うパケットの受信
2. コネクションを確立させる際に「(クライアントからの)確立要求に対する確認応答＋(サーバからの)確立要求」を行うパケットの送信
3. コネクションを確立させる際に「(サーバからの)確立要求に対する確認応答」を行うパケット受信

4． コネクションを切断させる際に「(サーバからの)切断要求」を行うパケットの送信

5． コネクションを切断させる際に「(サーバからの)切断要求に対する確認応答」を行うパケットの受信

6． コネクションを切断させる際に「(クライアントからの)切断要求」を行うパケット受信

7． コネクションを切断させる際に「(クライアントからの)切断要求に対する確認応答」を行うパケットの送信

上記 7 つのパケットのうち担当の番目（出席番号を 7 で割った余りの数字＋１）のパケットを観測と解析の対象とする。

課題④－２：クライアントから送信された文字が格納されたパケットの観測と解析

クライアントから送信するテキストは、苗字（全て小文字の英字）とする。

課題④のレポートの実験結果としては、指定の用紙に 2 つのパケット（課題④－1、課題④－２）の内容の記録と解析を記述する。【観測結果】には、16 進数の値をそのまま書き取ったもの、【解析結果】には、読み易い値（2 進数、10 進数、16 進数、文字）を適宜使用した分かりやすい記述に書き直したものを記述すること。レポートの最後に観測結果を繋げた状態の 1 つの pdf ファイルにして提出すること。(エクセルファイルを pdf にして、pdf 同士を連結させる)

課題⑤： Base64 による暗号化と復号を用いた通信（クラスとメソッドの自作課題）

クライアントの送信直前と、サーバの受診直後に、Base64 の変換を行うことで、暗号化・復号を行う送受信を実現する。

課題⑤－１： Java 言語の Base64 の文字列変換を行うクラスとメソッドの作成

C 言語の Base64 に関するサンプルプログラムを参考に作成する。クラスは事前に準備している MyBase64 とする。メソッドは static メソッドにすること。メソッド名は、以下とする。

static String encode(String str1)

平文の文字列を受け取って、Base64 で暗号化した文字列を戻り値とするメソッド

static String decode(String str1)

Base64 で暗号化された文字列を受け取って、復号した文字列を戻り値とするメソッド

課題⑤－１の実験結果として、少なくとも以下の点について調べよ。

・入力「a」の変換結果
・入力「hello」の変換結果
・入力「あ」の変換結果
・入力「制御」の変換結果
・入力「Σ」の変換結果

【チャレンジ課題（＋10 点)】課題⑤－２：通信プログラムへの組み込み

これまでに作成した通信プログラムの送信側（クライアント側）に自作した Base64 を、受信側（サーバ側）に java のライブラリにある Base64 を組み込み、暗号化・復号を利用した通信プログラムに拡張させる。

課題⑤－２の実験結果として、サーバ側で、暗号化されたデータを受信しているパケットを観測し、指定用紙に記録する。**ただし、記録は TCP ヘッダと TCP データ部分のみとする。**送信するテキストは、課題④－２と同様に、苗字（全て小文字の英字）とする。レポートの最後に観測結果を繋げた状態の１つの pdf ファイルにして提出すること。(エクセルファイルを pdf にして、pdf 同士を連結させる)

※チャレンジ課題に到達していなくても 100 点満点で評価します。チャレンジ課題をしている場合は、加点します。

課題⑤－１では、encode メソッドのみの完成を目指し行う。課題⑤－1 の作成が終わったら課題⑤－１の動作のチェックを教員の前で行うこと。(課題⑤－1 は必ず終わらせること）実験を通じてメソッドの検索、API の利用の技術を身に着けること。補助として java の文字列変換に関するメソッドのいくつかを以下に示す。下記以外のメソッドが必要な場合は、各人で調べ、プログラムに組み込むこと。

・String オブジェクトから char 配列への変換
```
char[]　cary = str.toCharArray( );
```
・int 型から String オブジェクトへの変換
```
String　str = String.valueOf( num );
```
・char 変数から 2 進数表記の String オブジェクトに変換
```
String　str_b = ( Integer.toBinaryString( c ) ).toString( );
```
・String オブジェクトの 2 番目から 7 番目までの要素の部分文字列の作成
```
String　str_sub = str.substring( 2, 7 );
```
注意：0 から数えて 2 番目の文字からスタートし、7 番目の直前の文字までを切り出す。
つまり、2,3,4,5,6 番目までの 5 文字の部分文字列が戻り値となる。

課題⑥：　共通鍵暗号方式による暗号化と復号を用いた通信（ライブラリの利用）

クライアントの送信直前と、サーバの受診直後に、共通鍵暗号方式の ABS を用いた変換を行うことで、鍵を用いたより安全な文字列の送受信を実現する。

課題⑥－１：　AES を用いた文字列変換を行うクラスとメソッドの作成

クラス名は MyCrypt.java とする。メソッドは static メソッドにすること。メソッド名は以下とする。

static　String　encode(String str,　String strK,　String strV)
　平文の文字列を受け取って、以下の設定で暗号化した文字列を戻り値とするメソッド

static　String　decode(String str,　String strK,　String strV)
　暗号化された文字列を受け取って、復号した文字列を戻り値とするメソッド

本実験での共通鍵暗号方式の設定

暗号化アルゴリズム：　AES（Advanced Encryption Standard）
　暗号化のアルゴリズムまたは規格の名称。

ブロックモード：　CBC（Cipher Block Chaining mode）

ブロック暗号（特定のビット数ごとに暗号化の処理を行う）を用いた暗号化の一種。

パディング方式： PKCS5Padding

8 バイトのブロック用のパディング方式。PKCS7Padding だと 16 バイトのブロック用。

encode／decode メソッドの処理の流れ

2 つの処理の流れは基本的に同じである。

1．鍵の作成

鍵となる文字列は、任意の**英数字の 16 文字**（16byte/128bit）でなければならない。

charset(文字コード)は、UTF-8。

2．初期化ベクトルの作成

この文字列も、鍵と同様に任意の**英数字の 16 文字**（16byte/128bit）でなければならない。

charset(文字コード)は、UTF-8。

3．変換器の作成（暗号化・復号アルゴリズムを実行するオブジェクトの作成）

上述の設定に従い、"AES/CBC/PKCS5Padding"と指定して作成する。

4．３で作った変換器の初期設定（モード（encode/decode）と１と２の指定）

5．変換器を使って暗号化または復号の実行しバイト配列を作成する。

6．５で作ったバイト配列を文字列に変換し、戻り値とする。

※encode メソッドでは変換後に Base64 のエンコードを行い、decode メソッドでは変換前に Base64 のデコードを行う。Base64 の変換には既存ライブラリ（java.util パッケージの Base64 クラス）を利用すること。

課題⑥－１の実験結果として、少なくとも以下の点について調べよ。

・入力「a」の変換結果

・入力「hello」の変換結果

・入力「あ」の変換結果

・入力「制御」の変換結果

・入力「Σ」の変換結果

【チャレンジ課題（＋10 点)】課題⑥－２：通信プログラムへの組み込み

これまでに作成した通信プログラムの送信側（クライアント側)、受信側（サーバ側）に自作したクラスのメソッドを利用した AES の変換処理を組み込み、暗号化・復号を利用した通信プログラムに拡張させる。

課題⑥－２の実験結果として少なくとも、以下の点については調べよ。

- クライアント側とサーバ側とで同じ鍵、同じ初期化ベクトルを使用した場合
- クライアント側とサーバ側とで同じ鍵、異なる初期化ベクトルを使用した場合
- 鍵や初期化ベクトルに英数字以外の文字を使用した場合

課題⑥－１の作成が終わったら動作のチェックを教員の前で行うこと。（課題⑥－１は必ず終わらせること）実験を通じてメソッドの検索、API の利用の技術を身に着けること。補助とし

て java の文字列変換に関するメソッドのいくつかを以下に示す。下記以外のメソッドが必要な場合は、各人で調べ、プログラムに組み込むこと。

・String オブジェクトから byte 配列への変換
変換の際に、String を指定の文字セット（＝文字コード）を引数で指定する。
byte[]　bary = str.getBytes(String charset);

・byte 配列から文字列への変換
String　str = new String(bary , String charset);

7. 検討課題

全体を整理するために、以下の検討課題を課する。テンプレート（全体-出席番号-名前.docx）のフォーマットに従って、以下の項目について調べて、検討ならびに考察を行うこと。

１．TCP におけるデータの信頼性について、パケットの内容に触れた上で説明せよ。
２．課題⑥にて、Base64 の処理を行う理由について説明せよ。
３．実験に使用した以外の暗号化技術について、技術の概要、特徴または利点、利用例の 3 項目に分けて説明せよ。

8. 参考図書

・村田正幸著　　マルチメディア情報ネットワーク　共立出版　（3 年科目「計算機ネットワーク I」の教科書）
・村山公保著　　基礎からわかる TCP/IP ネットワークコンピューティング入門　オーム社
・井口信和著　　基礎から分かる TCP/IP ネットワークツール活用　オーム社
・結城浩著　　暗号技術入門－秘密の国のアリス　ソフトバンククリエイティブ社
・神永正博著　　Java で作って学ぶ暗号技術　森北出版社

その他

未実験があった場合（公欠、病欠など）、また、レポート提出遅れ、授業への大幅な遅刻があった場合には、補講として追加実験を行う。追加実験の内容は補講時に知らせる。

以上