最終更新　2017.10.20

# インターネットとプロトコル

　本章では代表的なプロトコルについて学習します。

## インターネットとプロトコル

　インターネットには多数のコンピューターが接続されています。コンピューターを役割で分けると、サーバー（何らかのサービスを提供するコンピューター）とクライアント（何らかのサービスを受けるコンピューター）の2つに分けられます。

　我々が使うパソコンやスマートフォンなどはクライアント、GoogleやYahooなどのサイトの実体はサーバー群です。

　クライアントとして使うパソコンのOSとしてはWindows, Mac OS, Unix（ほとんどはLinux）があります。スマートフォンのOSとしてはiOS, Androidなどがあります。

　サーバーコンピューターのOSとしてはUnix（多くはLinux）, Windows Serverなどがあります。

　パソコンの中では多数のプログラムが同時に動作しています。Webブラウザをはじめとする多くのプログラムはネットワークを通して他のプログラムと通信を行います。このイメージを図5.1に示します。



図5.1　インターネットのしくみ

　Webブラウザ（ex. IE, Firefox, Safari）はWebサーバー上で動作しているサーバープログラムと通信を行います。メールソフト（例えばThunderbird）はメールを送る時はsmtpサーバープログラムと通信を行い、メールを受け取るときはpop3サーバープログラムと通信をします。ネットワークを使うアプリ（例えばLine）はインターネット上にあるアプリ専用サーバーと通信を行うと思われます。

　前章で、通信を行うプログラムは相手を識別するのにIPアドレスとポート番号を使用することを学習しました。ポート番号のうち、0～1023番はWell known portと呼ばれ、用途が決まっています。有名なポート番号として表5.1のような番号があります。

表5.1　代表的なポート番号

|  |  |  |
| --- | --- | --- |
| ポート番号 | プロトコル名 | 用途 |
| 20, 21 　 | 　ftp | 　ファイルの送受信 |
| 22 　 | 　ssh | 　遠隔ログイン（暗号化付き） |
| 23 　 | 　telnet | 　遠隔ログイン |
| 25 　 | 　smtp | 　メール配送 |
| 80 　 | 　http | 　Web |
| 110 　　　 | 　pop3 | 　メール受信 |
| 114 　 | 　imap4 | 　メール受信 |
| 115 　 | 　sftp | 　ファイルの送受信（暗号化付き） |
| 443 　 | 　https | 　Web（暗号化付き） |
| 587 　 | 　smtp AUTH | 　認証付きメール配送 |

　本章ではsmtpとhttpについて学習します。各プロトコルの詳細は

http://www.atmarkit.co.jp/fnetwork/rensai/netpro00/netpro01.html

などで分かりやすく解説されています。

## メール配送用プロトコル smtp



図5.2　メールソフトを使う場合のメール配送のしくみ

　今ではメールの読み書きはWebブラウザを通して行うことが多いと思いますが、ここでは古典的な場合を学習します[[1]](#footnote-1)。Thunderbirdのようなメーラー（メールを読み書きするアプリ）を使ってパソコンでメールを読み書きする場合、図5.2のような感じになります。メールを送るときはsmtpサーバー、受け取るときはpop3サーバーとやりとりします。メールの実体はパソコン内にあります。

　なお図5.2ではsmtpサーバーは25番ポートを使用しています。IDとパスワードによる認証は不要です。初期のインターネットではメール配送に25番ポートが使われていましたが、現在（2007年頃以降？）のプロバイダは認証付きのメール配送を587番ポートで受けています。smtp認証のしくみはhttp://www.kanadas.com/investigation-j/2007/04/smtp.htmlで分かりやすく説明されています。今でも25番ポートで認証なしの接続を許可しているのは、大学内や会社内など信用できるメンバーのみで構成された組織の内側だけでしょう。

　ここでは、smtpプロトコルについて学習するため、最も単純なケースである図5.2においてメールソフトが行うやりとりについて勉強します。メーラーとメールサーバー間の通信例を表5.2に示します。

表5.2　メールの送信例

|  |  |  |
| --- | --- | --- |
| 送信者 | 通信内容 | 意味 |
| SCSCSCSCSCCCSCS | 220 mail2.nara-edu.ac.jp ESMTP（以下略）HELO machine-name250 mail2.nara-edu.ac.jpMAIL FROM:<xxx@nara-edu.ac.jp>250 2.1.0 OkRCPT TO:<yyy@nara-edu.ac.jp>250 2.1.5 OkDATA354 End data with <CR><LF>.<CR><LF>ここにメールの本文を書く.250 2.0.0 Ok: queued as F0A0F22FCBDQUIT221 2.0.0 Bye | 自分の名を名乗る250はackの意味メールの送信元宛先内容を送る合図ピリオドが終わりの印コネクションを切断する |

　送信者欄のSはサーバー、Cはクライアントを表します。メールソフトの「送信」ボタンを押すとメールソフトとメールサーバーは表5.2のような通信を行います。本実習ではこのやりとりを手作業で行います。

　HELO, MAIL FROMなどの命令を小文字で入力してもエラーにはなりませんが、MAIL FROM: と RCPT TO:　のコロンの直前に空白を入れるとエラーになるので注意して下さい。

　手紙に例えると、表5.2の「RCPT TOとMAIL FROM」はそれぞれ「封筒の表面に書いてある宛先と裏面に書いてある差出人」に該当し、「DATA～.（ピリオド）」の本文は「手紙の中身」に該当します。

　e-mailの本文は「このような形式にしなさい」というルールがあります。そのルールはRFC822と名付けられています。それによるとe-mailの本文は「ヘッダブロック」と「メッセージブロック」の2つのブロックからなり、2つのブロックは空行（CR LF）で区切られることになっています。ヘッダブロックの中には差出人を表すフィールド（行）や件名を表すフィールドなどを書きます。ただし、本校のメールサーバーの場合、ヘッダブロックを省略してもメールを受け取ってくれます。本文の例を示します。

|  |
| --- |
| From: Tetsuro Yabu <yabu@略.ac.jp>To: (Tetsuro Yabu) yabu@略.ac.jpSubject: Test mailReply-To: hanako@mydomain.comX-My-header: dokuji no jyouhouTest mail.last line. |

　最初の5行がヘッダブロックで、最後の2行がメッセージブロックです。この例では、ヘッダブロックに5個のフィールドがあります。1個目のフィールドの「From:」の部分をヘッダと言います。ヘッダは「:（コロン）」までの文字列で、1桁目から始まります。行頭にスペースを入れてはいけません。スペースかタブから始まる行は継続行とみなされます。ヘッダの大文字と小文字は区別されません。アドレスを表すフィールドにおいては、(　　) で囲んだ部分はコメントと見なされます。また、< > で囲んだ部分がある場合、その中がアドレスで、それ以外はコメントと見なされます。

　From: の後に差出人のアドレス、To: の後に宛先のアドレス、Subject: の後に件名を書きます。

　メールソフトにおいて受信したメールに対して「返信」の操作をした場合、ヘッダブロックの中にReply-To: というヘッダが存在する場合はそのアドレスを返信先アドレスに設定し、無い場合はFrom: アドレスを返信先アドレスに設定します。上のメールに対して返信しようとすると、hanako@mydomain.com が返信先アドレスに設定されます。メーリングリストを運用する場合などにReply-To: フィールドは使われます。

　開封確認を求めるメールがあります。Disposition-Notification-To: というヘッダがある場合、メールソフトは指定されたアドレスへ開封確認を送って良いかどうかユーザーに問い合わせます。X- で始まるヘッダは個々のメールソフトが独自に設定したヘッダです。

　メールをやりとりするためのプロトコル（メールサーバーとメールソフトが通信するときの規則を定めたもの）はSMTP（Simple Mail Transfer Protocol）と呼ばれます。SMTPで定義されている主な命令を表5.3に示します。先ほどの表5.2の通信はこの命令に従って行っています。

表5.3　smtpの命令表

|  |  |
| --- | --- |
| 命令 | 意味 |
| HELO | サーバーに自分自身の正体を知らせる |
| MAIL FROM | 送信側のメールボックスを知らせる。宛先不明などの時のエラーメッセージはこのアドレスへ返される。アドレスは < > で囲む |
| RCPT TO | 宛先のアドレスを指定する。アドレスは < > で囲む |
| DATA | メッセージを送る準備が出来たことをサーバーに知らせる。返答コードが354のときデータを送信することが出来る。 |
| QUIT | 通信を終了する |

　以下の課題は最初はclient.exeを使って行いますが、途中からは図5.3のmail.exeを使います。「宛先」「メールサーバ名」「差出人」「送信内容」を入れて「メール送信」ボタンを押します。



図5.3　mail.exeの外観

課題1

　各小問においてアンダーラインが引かれた部分に答えなさい。

(1) 表5.2の手順に従って、「汎用通信プログラム」を用いて自分自身のアドレスにメールを送りなさい。奈良教育大学の学生用のメールサーバーのホスト名とポート番号は

student.nara-edu.ac.jp　　25

です。アドレスは

xxxx@nara-edu.ac.jp

です。HELOコマンドの後に名乗るホスト名は、本来なら現在操作しているコンピューターのホスト名を入れるべきですが、本校のメールサーバーの場合、何を入れても受け付けてくれるようです。MAIL FROMコマンドの後のアドレスはエラーが発生したときにエラー発生を通知するメールを送る時の送り先などに使われます。本校のメールサーバーの場合、学内から出す場合は何を入れても受け付けてくれるようですが、一般のプロバイダではFrom: フィールドと異なるアドレスに設定するとメール配送を拒否されると思われます。

　まずは、本文として This is a test mail という一行だけのメール（すなわち、ヘッダブロックは存在しない）を自分自身に送って下さい。

　本校のWebメールソフトであるActive Mailは送られてきたメールを読みやすい形に整形して表示します。送られてきたメールの素のままの姿を見るには、右端の「▼操作を選択」→「ソース表示」と操作します。すると、メールのヘッダブロックも含めて全て表示します。

　今回送ったメールはヘッダブロックなしのメールでしたが、「ソース表示」をすると、ヘッダブロックがあり、多数のフィールドがあります。このヘッダブロックはメールサーバーが自動的に付加したものです。メールサーバーによって自動的に付加されたヘッダにはどのようなものがありますか？　ヘッダ名を全て書きなさい。このうち、メールの配送経路を表すヘッダ名はどれですか？

　送信者はどうなっていますか？

(2) この課題以降はmail.exeを用いてよい。

　本文のヘッダブロックを以下のように書いて下さい。

　　From: aaa <aa@bb.cc>

この通りでなくても良い

　　To: bbb <xx@yy.zz>

　　Subject: ccc

というフィールドを含めて下さい。差出人を表す From: と宛先を表すTo: にわざと間違った名前とアドレスを書いています。本来は From: の後には MAIL FROM: で指定したアドレス、To: の後にはRCPT TO: で指定したアドレスを書くべきです。受信したメールの「件名」「送信者」はどうなりますか？

＜注意＞

　奈良教育大学のメールサーバーはヘッダブロックの From: と送信時の MAIL FROM: が異なっていてもメールを受け付けてくれますが、一般のプロバイダではそのようなメールは拒否されます。

(3) ヘッダブロックの中に

　　Reply-To: nazo no hito <nazo@himitu.com>

というフィールドを含めて下さい。そのメールに返信しようとすると、返信先のアドレスはどこになりますか。また、そのフィールドを入れない場合、返信先のアドレスはどこになりますか。

　ここまでの実験より、メールにおいて差出人の偽装方法が分かったと思います。手紙に例えると、「封筒に書いてある宛先に従って配送が行われるが、受取人に渡されるものは手紙の中身であり、封筒に書いてある情報は伝わらない」となります。そして、手紙の中身に書いてある「差出人、受取人」は「真の差出人、真の受取人」と異なっている場合があります。

## Web用プロトコルhttp

　httpはWebブラウザがWebサーバーと通信するためのプロトコルです。まずは、Webブラウザで次のページを開いて下さい。

http://denki.nara-edu.ac.jp/~yabu/edu/jyoho/html/simple.html

ブラウザのurl（Uniform Resource Locator）入力欄に上のようなアドレスを打つと、ブラウザは次のような動作を行います。

1. コンピューター denki.nara-edu.ac.jp の80番ポートに接続する。

2. ファイル /~yabu/edu/jyoho/html/simple.html を要求する。

3. 取得した内容を表示する。

　Webブラウザが行っている作業を手作業で行ってみましょう。汎用通信プログラムclient.exeを使います。Webサーバーに接続した後、以下のような文字列を送信します（現在のブラウザはHTTP/1.1プロトコルを用いています。HTTP/1.1はHOSTヘッダなどが必須となりタイプ量が増えるため、ここでは単純なHTTP/1.0を使用します）。

GET /~yabu/edu/jyoho/html/simple.html HTTP/1.0

　　（空行）

　GET はファイルを取得する命令です。取得したいファイル名をその後に書き、使うプロトコルのバージョンを書きます。GETを含む行の次に、空行を1行送ることに注意して下さい。空行は命令の終わりを表します。次のような文字列が送られてくるはずです。

HTTP/1.1 200 OK

Date: Fri, 08 May 2009 04:30:09 GMT

Server: Apache

Last-Modified: Thu, 02 Oct 2003 03:04:29 GMT

ETag: "75295b-67-81932940"

Accept-Ranges: bytes

Content-Length: 103

Connection: close

Content-Type: text/html

<html>

<head>

<title>The simplest page</title>

</head>

<body>

This is a simplest page.

</body>

</html>

　Webブラウザはこの文字列を受け取って、内容を画面に表示します。ブラウザで「表示」→「ソース」と操作すると、<html>から下の部分を見ることが出来ます。

　ここでは簡略化のため、client.exeからWebサーバーに送る文字列は1行だけにしましたが、実際のWebブラウザは色々な情報をWebサーバーに送っています。

　組織によってはWebを利用するときプロキシサーバーを経由することを義務づけている場合があります。そのような場合は以下のようにして下さい。プロキシサーバーの名前とポート番号が以下のように与えられていると仮定します。

proxy.abc-univ.ac.jp　　8080

プロキシサーバーのポート番号としては8000番や8080番が使われることが多いです。プロキシサーバーに対して、次のような文字列を送信して下さい。

GET http://xxx.nara-edu.ac.jp/~yabu/edu/jyoho/simple.html HTTP/1.0（ここまで1行）

Pragma: no-cache

　　（空行）

　Webサーバーは1回通信すると、コネクションを切断してしまいます。従って、Webサーバーに対して何かをGETするときは、その都度、接続を要求する必要があります。「汎用通信プログラム」の「接続」ボタンは毎回押してください。

　GETは指定したファイルを取得する命令でした。httpの主な命令は2つしかありません。1つはここで紹介したGETであり、もう1つはブラウザからWebサーバーへデータを送るときに使われるPOSTです。ただし、GETを使ってもブラウザからWebサーバーへデータを送れます。検索サイトでキーワードを入れて「検索」ボタンを押したときはGETが使われます。

課題2

(1) server.exeを起動し80番ポートを開けて待機します。すなわちserver.exeはWebサーバーのふりをすることになります。IEを起動し、http://localhost/ へアクセスして下さい。localhostは自マシンを意味します。IEがWebサーバーに送信した文字列を全て書きなさい。

　どのような情報が送られていますか？

　次にserver.exeにおいて、送信先ソケット番号をIEのソケット番号に設定し、以下の文字列を送りなさい。

HTTP/1.0 200 OK

Content-Length: 3

（空行）

abc

　結果として得られたIEの画面をコピペしてWordに貼り付けなさい。

　操作をやり直す場合は、IEで「再読み込み」のボタンをクリックして下さい。

(2)「サーバーが受け取る情報」と書かれたリンクをクリックすると、「サーバーが受け取ることができる情報を表示します。例えば、HTTP\_REFERER = xxxxxx という表示は、= の左側が変数名、右側がその内容です。

　自分のパソコンのIE, Firefoxからクリックしなさい。また、自分のスマホからもアクセスしなさい。

　「パソコンからのアクセス」「スマホからのアクセス」のそれぞれの場合について、変数名を列挙し、各変数の内容が何を表しているか述べなさい。

　例えばYahooのサイトに行くとパソコンから行くかスマホで行くかで表示が異なります。接続元のパソコンのOSによって表示するページの体裁を変えていると思われます。

　ここでは、httpプロトコルを用いて通信を行いましたが、現在はhttps（暗号化されたhttp）を使う場合が多くなっています。

1. Webメールはどこででも読めるというメリットがあります。Webメールを使う場合、ユーザーが接続するのはWebサーバーであり、その向こうがどうなっているかは、ユーザーには見えません。このあたりについては、私もよく分かりません。 [↑](#footnote-ref-1)