File Transfer in Matlab using FTP

[1]Aravind.D, [2]Ishwarya Pandian
[1,2]Department of Electronics and Communication Engineering
Kingston Engineering College,Vellore, Tamilnadu, India
Email: sharewitharavind@gmail.com

Abstract-**The file transfer between persons can be accomplished by the use of E-Mails, social networks or other external devices such as pen drives, DVDs, CDs, memory cards, etc. The major problem with this kind of file transfer is insecurity. The password can be easily hacked by hackers. This problem can be overcome by use of File Transfer Protocol (FTP) in MATLAB. File Transfer Protocol (FTP) is a standard network protocol used to transfer Web page files from their creator to the computer that acts as their server for everyone on the Internet. It is also used to download programs and other files to your computer from other servers. This file transfer is made secure by the use of username and password which is same for all clients using MATLAB.**

I. INTRODUCTION

The file transfer between computers plays a vital role in modern technology. The goal is to transfer files using File Transfer Protocol(FTP) in MATLAB. This file transfer can be made secure by the use of username and password. File Transfer Protocol (FTP) is a standard Internet protocol for transmitting files between computers on the Internet.

A. Performing FTP file operations

From MATLAB, one can connect to an FTP server to perform remote file operations. The following procedure uses a public Math Works FTP server (ftp.mathworks.com). To perform any file operation on an FTP server, follow these steps:

1. Connect to the server using the ftp function.
2. Perform file operations using appropriate MATLAB FTP functions. For all operations, specify the server object. For a complete list of functions.
3. When you finish working on the server, close the connection object using the close function.

B. FTP Class

Connect to an FTP server by calling the ftp function, which creates an FTP object. Perform file operations using methods on the FTP object, such as mput and mget. When finish accessing the server, call the close method to close the connection.

C. Construction

f = ftp(host,username,password) connects to the FTP server host and creates FTP object f. If the host supports anonymous connections, one can use the host argument alone. To specify an alternate port, separate it from host with a colon (:).

D. Input Arguments

Host	String enclosed in single quotation marks that specifies the FTP server.
Username	String enclosed in single quotation marks that specifies your user name for the FTP server.
Password	String enclosed in single quotation marks that specifies your password for the FTP server. Because FTP is not a secure protocol, others can see your user name and password.

II. MATLAB SOFTWARE

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

1. Math and computation
2. Algorithm development
3. Modeling, simulation, and prototyping
4. Data analysis, exploration, and visualization
5. Scientific and engineering graphics
6. Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name MATLAB stands for *matrix laboratory*. MATLAB was originally written to provide easy access to

matrix software developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of application-specific solutions called *toolboxes*. Very important to most users of MATLAB, toolboxes allow you to *learn* and *apply* specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

A. *About Simulink*

Simulink, a companion program to MATLAB, is an interactive system for simulating nonlinear dynamic systems. It is a graphical mouse-driven program that allows you to model a system by drawing a block diagram on the screen and manipulating it dynamically. It can work with linear, nonlinear, continuous-time, discrete-time, multivariable, and multirate systems.

Block sets are add-ins to Simulink that provide additional libraries of blocks for specialized applications like communications, signal processing, and power systems.
Real-time Workshop is a program that allows you to generate C code from your block diagrams and to run it on a variety of real-time systems.

B. *About Toolboxes*

MATLAB features a family of application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment in order to solve particular classes of problems. Many toolboxes are available from The Math Works.

C. *Exchanging Data Files Between Platforms*

It's sometimes necessary to work with MATLAB implementations on different computer systems, or to transmit MATLAB applications to users on other systems. MATLAB applications consist of *M-files*, containing functions and scripts, and *MAT-files*, containing binary data. Both types of files can be transported directly between different computers:

1. M-files consist of ordinary text. They are machine independent. While different platforms terminate lines with various combinations of CR (carriage return) and LF (line feed) characters, the MATLAB interpreter tolerates all possible combinations. (However, text editors and other tools may not work correctly with M-files from other platforms.)
2. MAT-files are binary and machine dependent, but they can be transported between machines because they contain a machine signature in the file header. MATLAB checks the signature when it loads a file and, if a signature indicates that a file is foreign, performs the necessary conversion. To use MATLAB across different platforms, you need a program for exchanging both binary and text data between the machines. When using these programs, be sure to transmit MAT-files in *binary file mode* and M-files in *ASCII file mode*. Failure to set these modes correctly usually corrupts the data.

D. *MATLAB's Memory Management*

MATLAB uses the standard C functions malloc and free to allocate dynamic memory. These routines maintain a pool of memory that is allocated from the operating system relatively slowly. malloc and free allocate memory from this pool for MATLAB much more quickly. If the pool runs low, malloc asks the operating system for another large chunk of memory to replenish the pool. As MATLAB releases memory, the pool can grow very large.

To maintain speed, malloc and free do not return the additional memory to the operating system. These routines make the assumption that if you need a large amount of memory once, you will need it again. A side effect of this algorithm is that, once MATLAB has used a certain amount of memory, it is no longer available to other programs even if MATLAB is no longer using it. The memory in the pool only returns to the operating system when MATLAB terminates.

E. *Editor/Debugger*

The Editor/Debugger provides basic text editing operations as well as access tovM-file debugging tools. The Editor/Debugger offers a graphical user interface. It supports automatic indenting and syntax highlighting.

To specify the default editor for MATLAB, select Preferences from the File menu in the Command Window. On the General page, select MATLAB's Editor/Debugger or specify another.

To start the Editor/Debugger, select New from the File menu, or click the new file (page icon) button on the toolbar, or type edit at the command line.

To start the Editor/Debugger, opening it to a particular file, select Open from the File menu, or click the open file (folder icon) button on the toolbar, or type edit filename at the command line. Do not use the Editor/Debugger while you are running an M-file in the Command Window or you will get an error.

While running the Editor/Debugger without MATLAB open, it becomes a pure text editor; one cannot use it as a debugger. To use the debugger, launch the Editor/Debugger from within MATLAB or with MATLAB open.

F. Profiling

Profiling is a way to measure where a program spends its time. Measuring is a much better method than guessing where the most execution time is spent. One can probably deal with obvious speed issues at design time and can then discover unanticipated effects through measurement.

One key to effective coding is to create an original implementation that is as simple as possible and then use a profiler to identify bottlenecks if speed is an issue. Premature optimization often increases code complexity unnecessarily without providing a real gain in performance.

Use a profiler to identify functions that are consuming the most time, then determine why you are calling them and look for ways to minimize their use. It is often helpful to decide whether the number of times a particular function is called is reasonable. Because programs often have several layers, your code may not explicitly call the most expensive functions. Rather, functions within your code may be calling other, time-consuming functions that can be several layers down in the code. In this case it's important to determine which of your functions are responsible for such calls.

The profiler often helps to uncover problems that can solve by:

1. Avoiding unnecessary computation, which can arise from oversight.
2. Changing your algorithm to avoid costly functions.
3. Avoiding recomputation by storing results for future use.
4. When the point is reached where most of the time is spent on calls to a small number of built-in functions, one can probably optimized the code as much as expected.

G. Data Types

There are six fundamental data types (classes) in MATLAB, each one a multidimensional array. The six classes are double, char, sparse, storage, cell, and struct. The two-dimensional versions of these arrays are called *matrices* and are where MATLAB gets its name.

One will probably spend most of the time working with only two of these data types: the double precision matrix (double) and the character array (char) or string. This is because all computations are done in double-precision and most of the functions in MATLAB work with arrays of double-precision numbers or strings.

The other data types are for specialized situations like image processing,sparse matrices (sparse), and large scale programming (cell and struct).

One can't create variables with the types numeric, array, or storage. These *virtual* types serve only to group together types that share some common attributes. The storage data types are for memory efficient storage only.

One can apply basic operations such as subscripting and reshaping to these types of arrays but they can't perform any math with them. One must convert such arrays to double via the double function before doing any math operations.

One can define user classes and objects in MATLAB that are based on the struct data type.

H. Matrices in MATLAB

Informally, the terms matrix and array are often used interchangeably. More precisely, a matrix is a two-dimensional rectangular array of real or complex numbers that represents a linear transformation.

The linear algebraic operations defined on matrices have found applications in a wide variety of technical fields. (The Symbolic Math Toolboxes extend MATLAB's capabilities to operations on various types of nonnumeric matrices.)

MATLAB has dozens of functions that create different kinds of matrices. Two of them can be used to create a pair of 3-by-3 example matrices for use throughout this chapter.

The first example is symmetric.

```
A = pascal(3)
A =
1 1 1
1 2 3
1 3 6
```

The second example is not symmetric.

```
B = magic(3)
B =
8 1 6
3 5 7
4 9 2
```

Another example is a 3-by-2 rectangular matrix of random integers.

```
C = fix(10*rand(3,2))
C =
9 4
2 8
6 7
```

A c*olumn vector* is an *m*-by-1 matrix, a *row vector* is a 1-by-*n* matrix and a *scalar* is a 1-by-1 matrix. The statements

```
u = [3; 1; 4]
v = [2 0 —1]
s = 7
```

produce a column vector, a row vector, and a scalar.

```
u =
3
1
4
v =
2 0 —1
s =
7
```

I. LU, QR, and Cholesky Factorizations

MATLAB's linear equation capabilities are based on three basic matrix factorizations.

1. Cholesky factorization for symmetric, positive definite matrices
2. Gaussian elimination for general square matrices

3. Orthogonalization for rectangular matrices. These three factorizations are available through the chol, lu, and qr functions.

All three of these factorizations make use of *triangular* matrices where all the elements either above or below the diagonal are zero. Systems of linear equations involving triangular matrices are easily and quickly solved using either *forward* or *back substitution*.

III. BROWSERS

A. System Browser

The system browser that MATLAB uses depends on your platform:

1. On Microsoft Windows and Apple Macintosh platforms, MATLAB uses the default browser for your operating system.
2. On UNIX platforms, MATLAB uses the Mozilla Firefox browser. You can specify a different system browser for MATLAB using Web preferences.

B. Display Pages in Web Browsers

To display an HTML document in the MATLAB Web browser, double-click the document name in the Current Folder browser.

To display a Web page or any file type in the MATLAB Web browser:

1. Open the browser using the web command.
2. Type a URL or full path to a filename in the Location field.

IV. MATLAB SERVER SESSION

The MATLAB server session creates a data set to be transferred out, opens a TCP/IP server socket and waits for the client to connect to it. When the connection is established, the data is written out to the socket.

Prepare the data we want to send over to the client MATLAB session. In this case our data is created by a call to the membrane function.

```
data = membrane(1);
```

Let us list details of the data set we want to transfer. We will use this information later to set up some parameters on the server socket and in the client.

```
s = whos('data')
```

```
s =
        name: 'data'
        size: [31 31]
        bytes: 7688
        class: 'double'
        global: 0
        sparse: 0
        complex: 0
        nesting: [1x1 struct]
        persistent: 0
```

Get the dimensions of the data array we will be transferring.

```
s.size
```

```
ans =
    31    31
```

Get the number of bytes of data we will be transferring.

```
s.bytes
```

```
ans =
    7688
```

Start a TCP/IP server socket in MATLAB. By setting the IP address to '0.0.0.0' the server socket will accept connections on the specified port (arbitrarily chosen to be 55000 in our case) from any IP address. You can restrict the TCP/IP server socket to only accept incoming connections from a specific IP address by explicitly specifying the IP address. Note the new property NetworkRole.

```
tcpipServer = tcpip('0.0.0.0',55000,'NetworkRole','Server');
```

Set the OutputBufferSize property to a value large enough to hold the data. This is the first place where we use the output of the whos function, specifically the value of s.bytes.

```
set(tcpipServer,'OutputBufferSize',s.bytes);
```

Open the server socket and wait indefinitely for a connection. This line will cause MATLAB to wait until an incoming connection is established.

```
fopen(tcpipServer);
```

Since the MATLAB server code is running in a separate MATLAB session than the client, you may notice the Busy status next to the MATLAB Start Button in the server session until the following commands have been executed. You may stop the MATLAB server socket creation and break out of this busy state by using the Control-C key combination to close the server socket. Note that once you close the server socket clients will no longer be able to connect to it until it has been re-opened.

Once the connection is made by the client, write the data out and close the server.

```
fwrite(tcpipServer,data(:),'double');
fclose(tcpipServer);
```

V. MATLAB CLIENT SESSION

The MATLAB server session is running on a computer with a known IP address or hostname. In this case, this is the address '127.0.0.1'. The second MATLAB session that acts as the client application creates a TCP/IP client, connects to the server and retrieves the data. Once retrieved, the data will be visualized in the client session.

Create a MATLAB client connection to MATLAB server socket. Note the value of the NetworkRole property on the client. Also note that the port number of the client matches that selected for the server.

```
tcpipClient = tcpip('127.0.0.1',55000,'NetworkRole','Client')
```

```
  TCPIP Object : TCPIP-127.0.0.1

Communication Settings
    RemotePort:        55000
    RemoteHost:        127.0.0.1
    Terminator:        'LF'
    NetworkRole:       client

 Communication State
    Status:            closed
    RecordStatus:      off

Read/Write State
    TransferStatus:    idle
    BytesAvailable:    0
    ValuesReceived:    0
    ValuesSent:        0
```

Set the InputBufferSize property so we have sufficient room to hold the data that will be sent to us by the server. The number 7688 is the number of bytes in the data array. For more general purpose code, you can parametrize this code by using the value in s.bytes instead of the hard-coded value of 7688.

```
set(tcpipClient,'InputBufferSize',7688);
```

I will define a long value for the Timeout; the waiting time for any read or write operation to complete. Adjust this value to ensure that any data that is being transferred to the client will be read back within the selected timeout.

```
set(tcpipClient,'Timeout',30);
```

Open a TCPIP connection to the server.

```
fopen(tcpipClient);
```

Read the data that is being written out on the server. Note that the data types need to be matched up on the client and server. The number of double values to be read back from the server socket is 961. For more general purpose code, you may parametrize the second argument using prod(s.size), for example.

```
rawData = fread(tcpipClient,961,'double');
```

Close the connection to the server once we've retrieved the data.

```
fclose(tcpipClient);
```

VI. MATLAB CODINGS

```
close all;
clear all;
clc;

% FTP Create an FTP object.
%FTP(host,username,password) returns an FTP object.

f =ftp('162.202.67.157','bensingh','revolution');

%DISP Display array.
% DISP(X) displays the array, without printing the array name.
 disp(f);

%MPUT Upload to an FTP site.
% MPUT(FTP,FILENAME (or) DIRECTORY) uploads a file.
paths = mput(f,'E:\Magesh\GPS\screencapture.m');
 %MGET Download from an FTP site.
% MGET(FTP,FILENAME (or) DIRECTORY) downloads a file.
mget(f,'tcp.c');

close(f);
```

VI. PICTORIAL PROCEDURE

A. Step 1

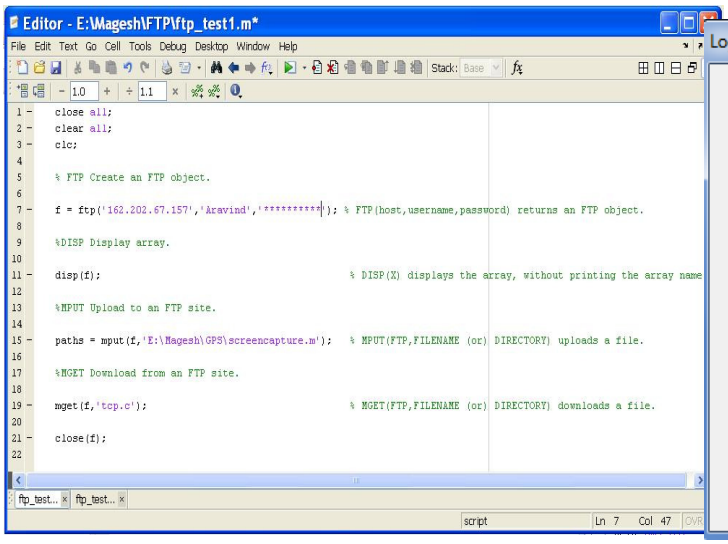

Fig.1. Enter the codings in the MATLAB Editor Window

B. Step 2

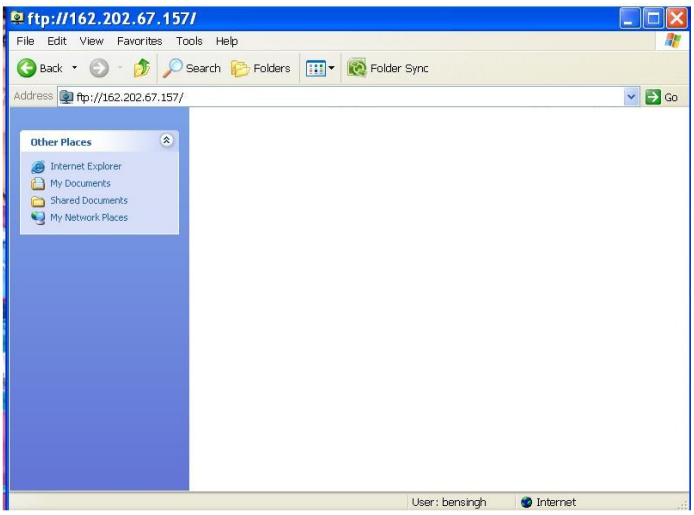

Fig.2. Address bar showing the IP address

C. Step 3

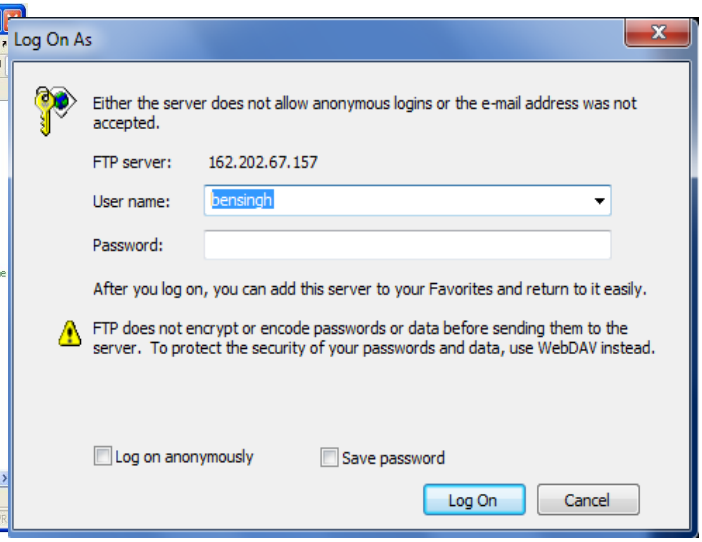

Fig.3. Log on with the user name and password

D. Step 4

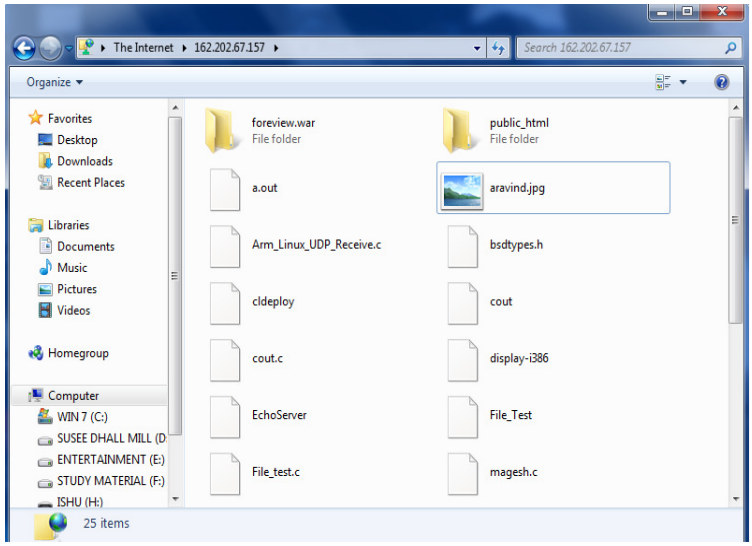

Fig.4. Files that are transferred can be accessed

CONCLUSION

The files that are uploaded in the MATLAB server session can be accessed by the client through FTP protocol with the help of Internet. The uploading of files requires a MATLAB platform but the accessing of files doesn't require it. The file transfer can be made secure by the use of user name and password which is common for all users. In the same way, the files which are uploaded by the client can also be accessed by the server over the Internet.

REFERENCES

[1] http://www.google.com/

[2] MATLAB 2010 software

[3] http://www.mathwork.com/help/

[4] Using MATLAB Version 5
COPYRIGHT 1984 - 1999 by The Math Works, Inc.

[5] MATLAB Data Analysis COPYRIGHT 2005 2013 by The Math Works, Inc.