Tutorials - Servers > Telnet as a Diagnostic Aid

Tutorials & FAQs: Servers: Telnet as a Diagnostic Aid

In this tutorial I will be explaining the use of Telnet for accessing your cgi webspace shell as well as using it as a very simple and useful diagnostic aid in determining if a server or service is responding to requests.

In the following sections I will explain what Telnet is and show how it can be used for:

- Accessing the shell on a CGI server (port 23)
- Checking a web server (port 80)
- checking a POP3 mail server (port 110)
- Checking a SMTP mail server (port 25)
- checking a Autoturn MX backup mail server (port 79)

The port numbers will become obvious in the section descriptions and are included here as a quick reference.

Special Note

This tutorial assumes you are running telnet on a Windows PC. While in most cases, the information below can be equally applied to running telnet on other systems (like Linux/Unix), there may be differences that mean it will not work as described. There are also differences between the Windows versions so be sure to check you are using the correct syntax or settings for your telnet version.

In the following text, <CR> is used to indicate the return key should be pressed. Any information contained within < and > e.g. <username> indicates where you must substitute your own information like login name, password etc. Within the example dialogs, any text between [and] is a comment by me and not part of the text you will see when enter or view information.

Using an alternative telnet client

As with all software, there are alternatives around that will often function better or have more features than the Windows telnet client application. While this tutorial uses the Windows version because it is included as standard, it should be possible to use alternative software to perform the same tasks. An example of an alternative telnet application is PuTTY but it's use is outside the scope of this document.

It is also possible to use the telnet client on the cgi server by logging on to your CGI webspace (explained later in this tutorial) and running telnet from the shell prompt ($).

What is telnet

Telnet is a text based communication program that allows you to connect to a remote server over a network. It is normal use is to login to a server that has shell access - in our case your CGI server - to allow you to run commands on the server. Telnet comes with all versions of Windows, as either a DOS mode command line program (Win2k/XP) or as a GUI text window (Win9x/Me/NT).

The telnet program has the following basic parameters, for a full list type telnet /? at a command prompt, but this is all we need to know for this tutorial:

> **telnet <hostname> <port>**
>
> **<hostname>** is the name or IP address of the remote server to connect to.
> **<port>** is the port number of the service to use for the connection. The default is 23 (TELNET service).
>
> Both parameters are optional and <port> can only follow <hostname>. If <hostname> is specified, telnet will attempt to connect to the remote system <hostname> on port <port> (default telnet port 23 if not specified). Running telnet without any parameters will invoke the telnet command mode prompt where you can enter commands to configure it.

Here is an example of Telnet running in command mode and showing the commands supported: (The example below is from WinXP, other Windows versions may be different; Win9x/Me/NT has a GUI front-end to telnet with menu selections for setting the options available):

```
C:>telnet <CR>
Welcome to Microsoft Telnet Client

Escape Character is 'CTRL+]'

Microsoft Telnet> help <CR>
Commands may be abbreviated. Supported
commands are:

c   - close                  close
current connection
d   - display                display
operating parameters
o   - open hostname [port]   connect to
hostname (default port 23).
q   - quit                   exit telnet
set - set                    set options
(type 'set ?' for a list)
sen - send                   send
strings to server
st  - status                 print
status information
u   - unset                  unset
options (type 'unset ?' for a list)
?/h - help                   print help
information
Microsoft Telnet>
```

- **open <hostname> <port>** will connect to remote system <hostname> on port <port> or port 23 if not specified.
- Once connected to a remote host, it is possible to get back to the local telnet prompt by pressing the escape character (CTRL+] (hold down the control (Ctrl) key and press the] key). To go back to the remote system just press <CR> at the local prompt.
- **close** to close the connection. This must precede any open command.
- To terminate the connection at the remote end (at a shell prompt) just type **exit <CR>** and you will be returned to the local telnet prompt.
- **quit** will terminate telnet.
- **set ?** will show the options that can be set options. The main one to remember is localecho which will be used later. Note: this differs depending on the Windows version. I.e. it may be local_echo.

If telnet is not able to connect to the remote host specified in <hostname> or establish a connection on the <port> specified, it will report an error similar to the following:

```
C:> telnet fred.plus.com <CR>
Connecting To fred.plus.com...Could not
open connection to the host, on port
23: Connect failed
C:> telnet mail.plus.net 60 <CR>
Connecting To mail.plus.com...Could not
```

```
open connection to the host, on port
60: Connect failed.
C:> telnet 60.92.12.56 74 <CR>
Connecting To 60.92.12.56...Could not
open connection to the host, on port
74: Connect failed.
```

If this happens check you have the correct hostname / IP address and port and try again.

There are several ways in which the telnet program can be run in Windows:

1. **From a DOS/CMD/COMMAND box**

   Click start->run, enter **command** <CR> (for Win9x/Me) or **cmd** <CR> (for WinNT/2k/XP) Note: **command** will actually work on all windows versions. This will bring up a DOS box. At the C:> prompt enter **telnet <hostname> <port>** <CR>. Telnet will attempt to connect to <hostname> on port <port>. If <hostname> is not specified you will get a telnet command prompt.

2. **From the run command**

   From the desktop, click start->run then enter **telnet <hostname> <port>** <CR> (e.g. telnet cgi.username.plus.com). This will bring up a telnet window and a connection with <hostname> will be attempted or a telnet command prompt if <hostname> is not specified. With this method, when you exit telnet, the window will be closed.

3. **From a browser address bar**

   Enter the following in the address bar: **telnet://<hostname>:<port>** <CR> (e.g. telnet://cgi.username.plus.com:23). This will bring up a telnet window and connect to <hostname> on port <port> if specified or a telnet command prompt if <hostname> is not specified. With this method, when you exit telnet, the window will be closed.

Accessing the shell on a CGI server

Here is an example session from a WinXP system showing the login/password entry and the shell prompt, from where you can enter *nix commands.

```
c:>telnet cgi.username.plus.com <CR>
Connecting to cgi.username.plus.com

[connecting]

Linux 2.4.22-5um (shellx.cgi.plus.net)
(00:49 on Saturday, 24 January 2004)
Login: <loginname> <CR>
Password: <password> <CR>
[password not shown]
Last Login: Fri Jan 23 23:17:52 from
username.plus.com

tutorialsteam@shellx tutorialsteam $
pwd <CR>
/files/home2/tutorialteam
tutorialsteam@shellx tutorialsteam $
exit <CR>
Logout

Connection to host lost.

c:>
```

Note: shellx will be **shell1** or **shell2** depending on which CGI server you are routed to at the time of connection.

That concludes the basic telnet usage. Now we come to the part that many telnet users may not be aware of. Telnet normally connects on port 23 (the default Telnet port) but it is possible to use Telnet to connect to a number of other ports or services on a remote server to check they are working. We will start with a basic web server check.

Checking a web server

If you do not have access to a browser, it is possible to use Telnet to perform some very basic checks to confirm if your web server is responding to requests. A web server normally listens for connections on port 80 (often referred to as HTTP) - Customer websites use this port. On other systems alternative ports may be used: Port 8080 is a common alternative or where a proxy is involved, Port 443 is normally used for secure (HTTPS) connections etc. Note: secure connections cannot be tested with Telnet because it has no knowledge of initiating a secure encrypted connection with the web server.

A web server does not produce a shell prompt like the CGI server, it just responds to HTTP requests/commands sent to it, which is exactly what a browser does. In simple terms a browser makes a connection to the web server and requests a page using a set of HTTP protocol commands. So lets simulate a basic connection to a web server and get some data back. In the example I will be using the tutorialsteam website and connect on port 80. Because the web server has no prompt and does not echo anything you type you have to type exactly what I have shown even though you will not see it when typing. Once a connection to the web server is established, we need to enter some HTTP commands to get the web server to send back some information. Note: You cannot use the backspace key to correct typing errors, if you do you will get an error reply - which is actually still testing the web server. If this happens just repeat the whole telnet connection again. Also the case of the text must be exactly as specified or it will not be recognised as a HTTP command:

```
c:>telnet www.tutorialsteam.plus.com 80
<CR>
Connecting to
www.tutorials.plus.com....

[at this point the screen will go blank
(wait for this to happen) and anything
you type will not be echoed back]

HEAD / HTTP/1.1<CR>
Host: www.tutorialsteam.plus.com<CR>
<CR>
```

The server will then reply with header information related to the default html page as defined by the web designer (normally something like index.html) or an error if you entered the HTTP commands wrong. The telnet connection will also close automatically. The important info is the 200 OK line, which indicates the commands were entered successfully and the web server has responded to the HTTP request.

A successful request will look similar to this:

```
HTTP/1.0 200 OK
Date: Sun, 25 Jan 2004 01:55:06 GMT
Server: Apache/1.3.26 (Unix)
Last-Modified: Sat, 06 Sep 2003
09:45:13 GMT
ETag: "bfde27-9ac-3f59aca9"
Accept-Ranges: bytes
Content-Length: 2476
Connection: close
Content-Type: text/html
```

If you entered any of the commands wrong you will get an error reply similar to this:

```
HTTP/1.1 400 Bad Request
Date: Sun, 25 Jan 2004 02:00:23 GMT
Server: Apache/1.3.26 (Unix)
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>
Your browser sent a request that this
server could not understand.<P>
The request line contained invalid
characters following the protocol
string.<P>
<P><HR>
<ADDRESS>Apache/1.3.26 Server at
hp1.plus.net Port 80</ADDRESS>
</BODY></HTML>
```

The above should work as detailed on PlusNet web servers. If you enter the HTTP commands correctly but still get an error, replace the command HEAD / HTTP/1.1 with GET / HTTP/1.1 and see if that works. With GET, you do receive additional text if the command was successful.

That concludes a basic test of the web server. We will now try testing a POP3 email server.

checking a POP3 mail server

If you are having problems receiving incoming emails from you mailbox, the following instructions will allow you to check if the POP3 mail server (which is used to receive and store email for your accounts) is responding to requests. This is the basic operation that your email client will be performing to download received email to your system.

A POP3 server normally accepts connections on port 110 so we will use this port when using telnet. In the following Dialog, we will connect to the mail server, login using your <username> and <password> and check to see if you have any messages waiting to be downloaded to your email client.

```
c:>telnet mail.plus.net 110 <CR>
Connecting to mail.plus.net

+OK Hello There
user <username> <CR>
+OK Password required.
pass <password> <CR>
+OK Logged in.
stat <CR>
+OK a b
quit <CR>
+OK goodbye
```

In the reply to the **stat** command, **a** is the number of messages (emails) waiting to be sent to you and **b** is the size in bytes of all the messages.

If you enter an unknown command you will see:

```
-ERR Invalid command.
```

If you enter an invalid username or the wrong password you will see:

```
-ERR Login failed.
```

Just repeat the user and pass commands using the correct information.

There are several other POP3 commands but the above is enough to prove the email server is responding to a connection request.

That concludes the test for a POP3 email server, we will now test a SMTP server, which is used for sending email to other people.

Checking the SMTP server

The server used to send email out to other email addresses is called an SMTP (Simple Mail Transport Protocol) server. Your email client will perform a similar activity to what is described here to send mail out. Connection to an SMTP server is normally done on port 25 and we will use Telnet to connect to port 25 on the server.

Editorial Note
Please see the Community Support Library article Testing SMTP Authentication Using Telnet for updated information on how to login to the SMTP server. The rest of this section is being left here for the moment since it might prove useful.
End of Note

The following dialog will simulate the creation and sending of a simple email message. All messages from the SMTP server will start with a number, all other text you should type. <email account> I suggest you use postmaster:

```
c:>telnet relay.plus.net 25 <CR>

220 relay.plus.net ESMTP Exim <today's
date>
helo username.plus.com <CR>
250-<server>.plus.net Hello
username.plus.com <ipaddress>
mail from: <email
account>@username.plus.com <CR>
250 OK
rcpt to: <email
account>@username.plus.com <CR>
250 Accepted
data <CR>            [start data (email
text) entry]
354 Please start mail input.
Date: 2 Jan 04 12:00:00
Subject: test email from SMTP
This is a test mail from the SMTP
```

```
server <CR>
. <CR>               [single dot on
it's own terminates message text]
250 Mail queued for delivery.
Quit <CR>
221 Closing connection
```

If you enter a command that is not supported you will get:

```
500 unrecognised command
```

Just check the command syntax and re-enter it correctly.

If you get stuck the following sequence of commands will reset and terminate the connection:

```
<CR>
.<CR>
RSET<CR>
Quit<CR>
```

All being well, you should be able to use the instructions for checking your POP3 server with telnet given earlier to see if you have received the message you sent above. Or just use your normal email client or webmail to view the message you have just sent to yourself.

checking an Autoturn MX backup mail server

Autoturn is a backup server for storing SMTP email messages that are waiting to be sent (dequeued) to a local SMTP server that you have configured. The following test is only necessary if you have configured your PlusNet email account to forward all emails to the IP address of your own SMTP server. If you have not done this then you will only see a blank screen.

```
C>: telnet autoturn.plus.net 79 <CR>

dequeuing X messages [only shown if
there are messages waiting to be sent
to your SMTP server]

[press Telnet Escape key CTRL+] to get
back to the local telnet prompt - there
is no other way to disconnect]

Microsoft Telnet>close <CR>
Microsoft Telnet>quit <CR>
```

That completes this tutorial. If you have any questions or comments regarding the contents of this document please PM me or one of the other tutorials team members.

Original Article by: petervaughan - Edited by: spraxyt