HowTo: UNIX / Linux Open TCP / UDP Ports

Posted By nixCraft <webmaster@cyberciti.biz> On September 3, 2010 @ 2:25 am [8 Comments]

[1]

How do I open the TCP or UDP ports under UNIX / Linux like operating systems?

A port is an application-specific or process-specific software construct serving as a communications endpoint and it is identified by its number such as TCP port number 80 . It is used by TCP and UDP [2] of the Internet Protocol Suite. A port number is a 16-bit unsigned integer, thus ranging from 0 to 65535.

```
      UNIX / Linux
   +------------------+
   | Networking stack |
   |      eth0        |
   +------------------+
          |
   +------------------+
   |  Apache process  |--> Binding port 80 @ 202.54.1.1 IP
   +------------------+
```


[3]

In the above example Apache process associates its input and output channel file descriptors (fd) with a port number 80 and an IP address 202.54.1.1. This is known as binding. It is used to send and receive web pages via UNIX / Linux operating system's networking stack (software). In other words communication is done using application ports. When you start the Apache you open port 80 for communication. Common services such as web, mail, pop3 et all use use specifically reserved, well-known port numbers for receiving service requests from client hosts. The well-known ports are defined the Internet Assigned Numbers Authority (IANA). Type the following command to see list well-known of TCP and UDP port numbers:

```
$ less /etc/services
grep -w 80 /etc/services
```

Sample outputs:

```
www  80/tcp  http  # WorldWideWeb HTTP
www  80/udp    # HyperText Transfer Protocol
```

Privileged Ports

Typically port number less than 1024 are used by well know network servers such as Apache. Under UNIX and Linux like oses root (super user) privileges are required to open privileged ports. Almost all clients uses a high port numbers for short term use. This is also known as an ephemeral port. For example Apache use TCP port 80

```
 Server                          Client w/ Firefox
+-----------+                    +-----------------+
| Apache    |                    | connects using  |
| TCP Port  |                    | an ephemeral    |
| 80 @      |<-----> eth0 <----> | port #          |
|202.54.1.2|                     | 46025           |
+-----------+                    +-----------------+
```

The port numbers are divided into three ranges:

1. Well Known Ports: those from 0 through 1023.
2. Registered Ports: those from 1024 through 49151
3. Dynamic and/or Private Ports: those from 49152 through 65535

You can increase local port [4] range by typing the following command (Linux specific example):

```
# echo 1024 65535 > /proc/sys/net/ipv4/ip_local_port_range
```

You can also increase or decrease [5] socket timeout (Linux specific example):

```
# echo 2000 > /proc/sys/net/ipv4/tcp_keepalive_time
```

Common Well Known Port Numbers

The following are used by UNIX / Windows / Linux / BSD / OS X and all other server operating systems or network devices (see /etc/services file):

- 21: FTP Server
- 22: SSH Server (remote login)
- 25: SMTP (mail server)
- 53: Domain Name System (Bind 9 server)
- 80: World Wide Web (HTTPD server)
- 110: POP3 mail server
- 143: IMAP mail server
- 443: HTTP over Transport Layer Security/Secure Sockets Layer (HTTPDS server)
- 445: microsoft-ds, Server Message Block over TCP

How Do I See Open Ports and Socket Information Under UNIX or Linux?

You can use the netstat command [6]:

```
# netstat -tulpn
```

FreeBSD specific [7] example:

```
# sockstat -l
```

To list open IPv4 connections use the lsof command:

```
# lsof -Pnl +M -i4
```

The ss command is used to dump socket statistics [8]. It allows showing information similar to netstat command. It can display more TCP and state information than other tools

```
# ss -s
# ss -l
# ss -pl
# ss -o state established '( dport = :smtp or sport = :smtp )'
```

Examples

Each TCP or UDP port is opened using a UNIX service or daemon such as Apache web server. You can also write a program using C, C++, Perl, Shell or Bash to open any port. You can also use utilities such as nc command .

Apache Server Example (open TCP port 80)

Start the Apache web server under FreeBSD as follows to open TCP port 80:

```
# /usr/local/etc/rc.d/apache22 forcestart
```

OR

```
# /usr/local/etc/rc.d/apache22 start
```

To displays listening sockets (open ports) under FreeBSD, enter:

```
# sockstat -l
```

OR

```
# netstat -nat | grep LISTEN
```

You should see port 80 opened under FreeBSD. Under CentOS or Redhat (RHEL) Linux, you can open port 80 using the following commands:

```
# service httpd start [9]
# chkconfig httpd on [10]
# netstat -tulpn | grep :80
```

Firewall Configuration

All port numbers are encoded in the transport protocol packet header, and they can be read by other components of the network stack such as firewall. Firewall can be used for port forwarding or denying access to open port. For example, block an abusing IP address called 1.2.3.4 using UNIX firewall. In other words, Apache port is open but it may be blocked by UNIX (pf) or Linux (iptables) firewall. You also need to open port at firewall level. In this example, open tcp port 80 using Linux iptables firewall tool:

```
# /sbin/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT
# service iptables save
```

See also:

1. CentOS / Redhat Linux Iptables Firewall Configuration Tutorial
2. Redhat / CentOS / Fedora Linux Open Port [11]
3. FreeBSD Setting up Firewall using IPFW [12]
4. OpenBSD PF Firewall Script – /etc/pf.conf File [13]

nc Command Example

The nc (or netcat utility) is used for just about anything under the sun involving TCP or UDP. It can open TCP connections, send UDP packets, listen on arbitrary TCP and UDP ports, do port scanning, and deal with both IPv4 and IPv6. In this example, open port 5000 using nc command:

```
$ nc -l 5000
```

On a second console or from a second UNIX / Linux machine, connect to the machine and port being listened on:

```
$ nc localhost 5000
```

OR

```
$ nc unix.system.ip.here 5000
```

In this example, send data from one computer to another:

```
$ nc -l 5555 > output.txt
```

Using a second machine, connect to the listening nc process (@ port 5555), feeding it the file which is to be transferred:

```
$ nc your.unix.systems.ip.here 5555 < input.txt
```

You can run netstat command to view open ports:

```
$ netstat -a
$ netstat -nat | grep LISTEN
```

Sample outputs:

```
tcp4       0      0  *.5555                 *.*                    LISTEN
tcp4       0      0  10.1.3.29.53           *.*                    LISTEN
tcp4       0      0  192.168.56.1.53        *.*                    LISTEN
tcp4       0      0  115.242.47.238.53      *.*                    LISTEN
tcp4       0      0  127.0.0.1.953          *.*                    LISTEN
tcp4       0      0  127.0.0.1.53           *.*                    LISTEN
tcp4       0      0  127.0.0.1.631          *.*                    LISTEN
tcp6       0      0  ::1.631                *.*                    LISTEN
```

Python Example

Create a file called echo_server.py:

```
#!/usr/bin/python

# Demo server to open port 8888
# Modified from Python tutorial docs
import socket

HOST = '127.0.0.1'       # Hostname to bind
PORT = 8888              # Open non-privileged port 8888

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))
s.listen(1)
conn, addr = s.accept()
print 'Connected by', addr
while 1:
    data = conn.recv(1024)
    if not data: break
    conn.send(data)
conn.close()
```

Create a file called echo_client.py:

```
#!/usr/bin/python

# Demo client program
# Modified from Python tutorial docs
import socket

HOST = '127.0.0.1'      # Set the remote host, for testing it is localhost
PORT = 8000             # The same port as used by the server
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
s.connect((HOST, PORT))
s.send('Where there is love there is life')
data = s.recv(1024)
s.close()
print 'Received', repr(data)
```

Save and close the file. Run it as follows:

```
$ chmod +x *.py
```

Start server, enter:

```
$ ./echo_server.py
$ netstat -nat | grep LISTEN
```

On a second console connect to the localhost and port being listened on using echo_client.py:

```
$ ./echo_client.py
```

Programming Language Specific Examples

Discussion regarding Sockets and TCP/IP network programming is beyond the scope of this FAQ. I suggest you visit the following web-pages:

Recommended readings:

- Port Numbers [14]: The Internet Assigned Numbers Authority (IANA).
- Python documentation [15]: See networking and sockets section.
- TCP [16]: Transmission Control Protocol.
- Perl specific TCP/IP [17] networking using IO::Socket::INET.
- UNIX socket [18]: Programming in C UNIX System Calls and Subroutines using C.
- man pages - ss, netstat, lsof, sockstat, nc, services, ntsysv

URL to article: **http://www.cyberciti.biz/faq/linux-unix-open-ports/**

URLs in this post:

[1] Image: **http://www.cyberciti.biz/faq/category/unix/**
[2] TCP and UDP: **http://www.cyberciti.biz/faq/key-differences-between-tcp-and-udp-protocols/**
[3] Image: **http://www.cyberciti.biz/faq/category/linux/**
[4] increase local port: **http://www.cyberciti.biz/tips/linux-increase-outgoing-network-sockets-range.html**
[5] increase or decrease: **http://www.cyberciti.biz/tips/linux-increasing-or-decreasing-tcp-sockets-timeouts.html**
[6] netstat command: **http://www.cyberciti.biz/tips/netstat-command-tutorial-examples.html**
[7] FreeBSD specific: **http://www.cyberciti.biz/tips/freebsd-lists-open-internet-unix-domain-sockets.html**
[8] ss command is used to dump socket statistics: **http://www.cyberciti.biz/tips/linux-investigate-sockets-network-connections.html**
[9] service httpd start: **http://www.cyberciti.biz/faq/check-running-services-in-rhel-redhat-fedora-centoslinux/**
[10] chkconfig httpd on: **http://www.cyberciti.biz/faq/rhel5-update-rcd-command/**
[11] Redhat / CentOS / Fedora Linux Open Port: **http://www.cyberciti.biz/faq/howto-rhel-linux-open-port-using-iptables/**
[12] FreeBSD Setting up Firewall using IPFW: **http://www.cyberciti.biz/faq/howto-setup-freebsd-ipfw-firewall/**
[13] OpenBSD PF Firewall Script – /etc/pf.conf File: **http://bash.cyberciti.biz/firewall/pf-firewall-script/**
[14] Port Numbers: **http://www.iana.org/assignments/port-numbers**
[15] Python documentation: **http://docs.python.org/index.html**
[16] TCP: **http://en.wikipedia.org/wiki/Transmission_Control_Protocol**
[17] Perl specific TCP/IP: **http://perldoc.perl.org/IO/Socket/INET.html**
[18] UNIX socket: **http://www.cs.cf.ac.uk/Dave/C/**