Documentation

Scott Jin TCP Port Scanner

Usage

The Program was a simple implementation of port scanner as a command line tool. It works by trying to establish a connection with every port that is to be scanned. If a connection is established then the port is open otherwise closed.

```
USAGE: PortScanner hostname [-p 15:25]
INPUT : IPv4 address, Port Range
FUNCTION : Enter an IP address and a port range
       where the program will then attempt to
       find open ports on the given computer
       by connecting to each of them. On any
       successful connection ports, mark the
       port as open.
OUTPUT : Status of port (open/closed)
```

- [] means optional option as the defualt search will using port range from 0 to 2014.

Implementation

1. takes in the required arguments (hostname, start*port, end*port).
2. traverse through all the ports in range provided and then check against each one of them. [the second implementation put them in

   ```
   struct addrinfo
   ```

   ]

3. Connecting against the sockets
   the network host is assumed to be multihomed, accessible over multiple protocols (e.g., both AF*INET and AF*INET6); or the same service is available from multiple socket types.

4. try connecting the socket using socket function. if creation fails, then continue with another port. Once

socket creation succeeds, try connecting to it using the

```
connect().
```

If the connection is a success, then the socket is OPEN, else try with continue & exit.

5. The program tries to boost performance with forking bunches of processes and the defualt port range however is not tried as fork resources may not be available. The range for each process is hard-coded to one port but can always be change by makeing the program more robust by adding another option to specify port range for each processes.

Reflection

The port scanning for each port takes the complete three-way hand shake between the 2 hosts. It is the slowest and consumes the maximum time.

```
    Local ----> sends tcp syn packet -----> Server
    Local <---- replies with a syn+ack packet <----- Server
    Local ----> sends ack packet -----> Server
```

After all that, the connection is considered established. TCP syn port scanning should be used to avoid the wased full connection process and boost the perfomance by only performing partial connection.