Make and Sockets

UNX511 Week 8 Class 1

John Sellens

July 7, 2025

Seneca Polytechnic

Outline

More About Make

More About Make

- make is a handy tool for building software
    - Or running commands based on file timestamps ...
- Up to now, we've used some fairly simple Makefiles
- Let's have a look at some more advanced functionality
- See unx511_samples/week8_1/1_makefiles

Sockets and Network Ports

Remember Sockets?

- In week 4, class 2, we had a UNIX domain (filesystem) socket example
- Sockets are a method fof inter-process communication (IPC)
- You know, like web browsers and servers
- Let’s recap from week 4 class 2

Recap: What Is a Socket?

- Sockets allow bi-directional communication between processes
- They can be local only, or available across the network
- There are many different socket types (or families) – see `socket(2)`
- The most common are AF_INET (IPv4 internet protocols) and AF_INET6
- Today we will look at AF_UNIX – “UNIX Domain Sockets”
    - For local communication on a single machine
- A UNIX domain socket appears in the file system
- Similar to named pipes (FIFOs), but named pipes are unidirectional
- Sockets Tutorial: `https://www.linuxhowtos.org/C_C++/socket.htm`

Recap: How to Use Sockets

- The general method for using sockets is similar across families
- Connections are made by a client process connecting to a server process
- The server process gets ready
    - socket() – returns a file descriptor
    - bind() – attach to a network port or UNIX domain socket
    - listen() – wait for a client to ask to connect
    - accept() – accept a connection, returns a read/write file descriptor
- The client process initiates a connection to the server
    - socket() – returns a file descriptor
    - bind() – only if network, establishes local network port
    - connect() – connect to a server
- Processes then read/write over the connection until `close()`

Streams and Datagrams

- Every sockets implementation provides at least two types of sockets: stream and datagram. These socket types are supported in both the UNIX and the Internet domains.
- Stream sockets (SOCK_STREAM) provide a reliable, bidirectional, byte-stream communication channel. – e.g. TCP/IP
- Datagram sockets (SOCK_DGRAM) allow data to be exchanged in the form of messages called datagrams. With datagram sockets, message boundaries are preserved, but data transmission is not reliable. Messages may arrive out of order, be duplicated, or not arrive at all. – e.g. UDP

FYI: Host Names and IP Addresses

- In our examples, we have been using IPv4 addresses for connections
    - We use `inet_pton(3)` to convert an address string to 32 bit integer
- In real applications, we're much more likely to use host names
    - Or fully-qualifed domain names (FQDNs)
- We could look up host names and get their addresses with `gethostbyname(3)`
    - Which uses local hosts files and DNS (and maybe other methods)
    - A host can have multiple addresses
- Once we have an address, we can use it in `bind(2)` or `connect(2)` calls

Port Numbers

- Network sockets bind to a network interface (or all interfaces) and to a port number
    - Port numbers are 16 bit unsigned integers, from 0 to 65535
- Common port numbers include 22 SSH, 80 HTTP, 443 HTTPS, 25 SMTP, ...
- Standard port numbers are assigned by the Internet Assigned Numbers Authority (IANA)
- On linux, see the file `/etc/services` for a list
- `https://en.wikipedia.org/wiki/Port_(computer_networking)`

Port Numbers Currently in Use

- The `netstat(1)` and `ss(1)` commands can tell you what ports are currently open or in use
- e.g. `netstat -lntup` or `ss -lntup`
  - -l listening sockets,
  - -n numeric values rather than service names,
  - -t all TCP connections,
  - -u all UDP connections,
  - -p application that is listening on a port.
- Examples `https://www.binarytides.com/linux-netstat-command-examples/`

Socket Code Examples

- Let's have a look in unx511_samples
    - `https://github.com/jsellens/unx511_samples`
- `week8_1/2_unix_sockets` – UNIX client/server pair
- `week8_1/3_inet_local` – local network stream
- `week8_1/4_inet_network` – two host network datagram
- `week8_1/5_inet_select` – local UNIX stream multi-client select

Socket References

- `socket(2)` `bind(2)` `listen(2)` `connect(2)` `accept(2)`
- `select(2)` – includes FD_SET() and FD_ISSET()
- `inet_pton(3)` `inet_ntop(3)`
- `read(2)` `write(2)` `close(2)` `unlink(2)`
- `send(2)` – includes sendto()
- `recv(2)` – includes recvfrom()
- `getsockopt(2)` – includes setsockopt()
- `https://github-pages.senecapolytechnic.ca/unx511/Week8/Week8.html`

Summary

- Make is handy, and flexible
- Sockets are used everywhere!