Lab 1: Socket Programming - Understanding Networking Functions

Christian Garcia

December 4, 2023

1 Questions

1.1 Creating a socket

```
int new_socket;
if ((new_socket = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
  perror("The socket was failed to be created!");
  exit(EXIT_FAILURE);
}
```

The above code is used to create a new socket in C and C++. Using this sample code, determine...

1.1.1 What does AF_INET mean when creating a socket?

1.1.2 What does SOCK_STREAM mean when creating a socket?

1.2 Setting a port number and IP address

```c
#define DEFAULT_PORT 4200

socket_addr->sin_family = AF_INET;
if (port == 0) {
    socket_addr->sin_port = htons(DEFAULT_PORT);
    port = DEFAULT_PORT;
} else {
    socket_addr->sin_port = htons(port);
}
if (ip != NULL) {
    socket_addr->sin_addr.s_addr = inet_addr(ip);
} else {
    socket_addr->sin_addr.s_addr = INADDR_ANY;
}
```

The above code is used to assign an IP address and port number to the socket.

1.2.1 What does htons do to a numeric value like a port number?

1.2.2 What does inet_addr() do in the code?

1.2.3 What does INADDR_ANY mean when setting up a socket?

1.3   More Socket Operations

```c
if (bind(*socket, (struct sockaddr *)&addr, sizeof(addr)) < 0) {
  perror("The socket failed to bind!");
  exit(EXIT_FAILURE);
}
```

1.3.1   What does it mean for a socket to bind to another socket?

```c
if (listen(*socket, 1) < 0) {
  perror("The socket failed to listen!");
  exit(EXIT_FAILURE);
}
```

1.3.2   What does it mean for a socket to listen to another socket?

```c
if (connect(*socket, (struct sockaddr *)&ip_addr, sizeof(ip_addr))
    < 0) {
  perror("The socket failed to connect!");
  exit(EXIT_FAILURE);
}
```

1.3.3 Although this one is fairly straightforward, please technically explain what it means for a socket to connect to another socket?

```c
socklen_t sock_len = sizeof(sock_addr);
if ((*socket = accept(*socket, (struct sockaddr *)&sock_addr,
    &sock_len)) <
0) {
  perror("There was an error accepting a socket");
  exit(EXIT_FAILURE);
}
```

1.3.4 What does it mean for a socket to accept another socket?

```c
close(*socket);
```

1.3.5 Why is it important to close a socket when you are done using it?

1.4 Reading and Writing Data Over the Network

There are two primary methods of exchanging data over the network in C and C++. These methods are write()/read() and send()/recv(). Using online resources, please explain...

1.4.1 How do write() and read() exchange data over two sockets?

1.4.2 How do send() and recv() exchange data over two sockets?

1.4.3 After researching these two methods of exchanging data over the network, which one seems more reliable? Please explain your stance.

1.5   Additional Questions

1.5.1   Notice how the sample code contains an if statement comparing if the result of the function is less than 0. This is one of the main ways to perform error handling in C/C++. Why is error handling important when implementing a networked application?