Recitation 3

Internet Technology (Section 01)

ASCII TCP and TELNET

- A protocol that uses only ASCII characters over TCP
  - Telnet, HTTP, etc
- Telnet is protocol for accessing virtual terminals over a network
  - Developed in 1969!
  - Uses ASCII characters. Lots of existing clients
- Example: Play a text-based game
  - `$ telnet mtrek.com 1701`

Toy Example: A Remote Text Editor

- Need to support some ASCII commands
  - "**OPEN**", "**CLOSE**", "**INSERT**",
  - On close we send client contents of the edited file
- Specify how to differentiate between commands, data, and end-of-line. Need a format
  - "**:<COMMAND> <DATA>\n**"
- Examples
  - "**:OPEN file.txt\n**"
  - "**:INSERT This is a line of text\n**"
  - "**:CLOSE\n**"

TCP Remote Text Editor (Server)

```
import socket
HOST, PORT="127.0.0.1", 1234
sock = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR,
1)
sock.bind((HOST,PORT))
sock.listen()
buffer = ""
alive = False
CMD_OPEN = ":OPEN "
CMD_INSERT = ":INSERT "
CMD_CLOSE = ":CLOSE "
file = None
while True:
    #try to accept connection if ones doesn't exist
    if not alive:
        (conn, address) = sock.accept()
        alive = True
    else:
        #read data byte by byte
        data = conn.recv(1)
        if not data:
            alive = False
            continue
        data = chr(data[0])
```

```
        #read data until newline character
        if data != '\n':
            buffer+=data
            Continue
        #state machine
        if(buffer[:len(CMD_OPEN)] == CMD_OPEN):
            filename = buffer[len(CMD_OPEN)+1:]
            file = open(filename, "a+")
            buffer = ""
        if(buffer[:len(CMD_INSERT)] == CMD_INSERT):
            text = buffer[len(CMD_INSERT)+1:]
            file.write(text + "\n")
            buffer = ""
        if(buffer[:len(CMD_CLOSE)] == CMD_CLOSE):
            file.seek(0)
            contents = file.read()
            file.close()
            buffer = ""
            conn.sendall(contents.encode())
```

TCP Remote Text Editor (Client)

```
import socket
HOST="127.0.0.1"
PORT=1234
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.connect((HOST,PORT))
sock.sendall(b":OPEN test file.txt\n")
sock.sendall(b":INSERT A line of text\n")
sock.sendall(b":INSERT Another line of text\n")
sock.sendall(b":CLOSE\n")
data = sock.recv(1024)
print(data.decode())
sock.close()
```

- We don't actually need this client!
  - Our protocol is ASCII based so Telnet will suffice (need to use "`\n`" instead of "`\r\n`" for end-of-line)
- Example:

```
$ telnet 127.0.0.1 1234
  > :OPEN test.txt
  > :INSERT some text
  > :CLOSE
  > some text
```

Limitations

- Our remote text editor has many limitations, mainly:
  - Can’t actually edit existing contents of a file
  - Not robust if client disconnects mid-operation
  - Only 1 client can edit some file at a time
  - What if the client edits “`server.py`”?
  - ***The list goes on…***