# Web Server Design

## Lecture 2 – Socket Programming

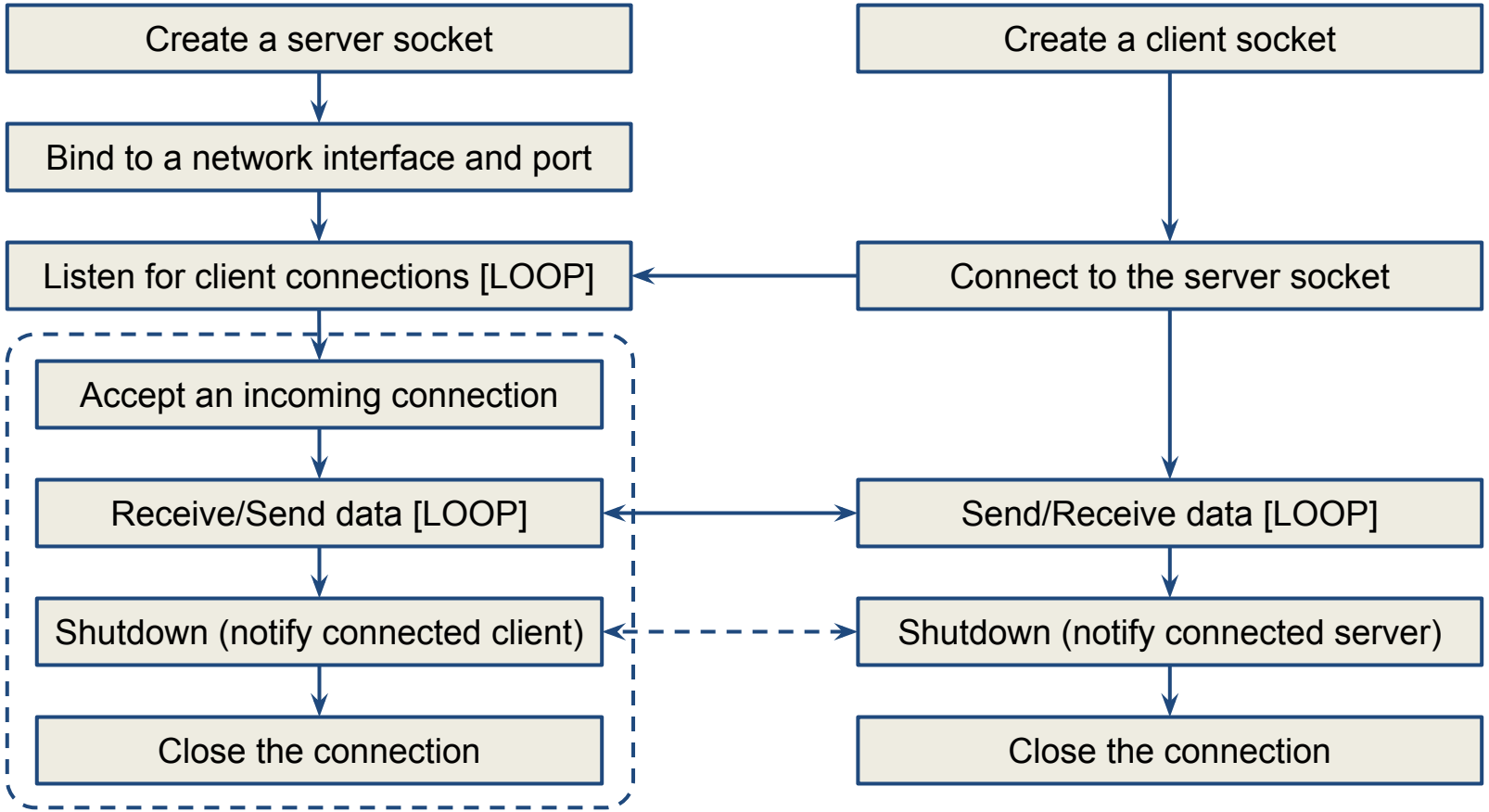
Old Dominion University
Department of Computer Science
CS 431/531 Fall 2019
**Sawood Alam** <salam@cs.odu.edu>

2019-09-07

# Interprocess Communication (IPC)

- File
- Shared memory
- Memory-mapped file
- Message passing
- Pipe
- Message queue
- Signal
- Unix socket
- Network socket

# Network Socket Workflow



# Host and Port

- Host
  - An IP address (IPv4 or IPv6)
  - A domain name mapped to an IP address
  - Identifies a host/machine on a network
- Port
  - A number from 0 to 65535
  - 0-1023 ranges is reserved (needs privileged access)
  - Identifies a process on a host for socket communication
  - Not every process is bound to a port

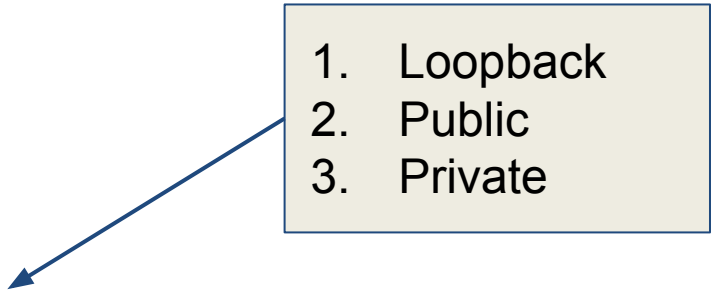
# Common Default Port Numbers

| Port Number | Service |
|---|---|
| 20/21 | File Transfer Protocol (FTP) |
| 22 | Secure Shell (SSH) |
| 23 | Telnet |
| 25 | Simple Mail Transfer Protocol (SMTP) |
| 53 | Domain Name System (DNS) |
| 80 | Hypertext Transfer Protocol (HTTP) |
| 110 | Post Office Protocol (POP3) |
| 123 | Network Time Protocol (NTP) |
| 143 | Internet Message Access Protocol (IMAP) |
| 194 | Internet Relay Chat (IRC) |
| 443 | HTTP Secure (HTTPS) |

https://en.wikipedia.org/wiki/Port_(computer_networking)

# Network Interfaces

```
$ host cs531.cs.odu.edu
cs531.cs.odu.edu has address 128.82.7.233
$ ssh cs531.cs.odu.edu
$ host localhost
localhost has address 127.0.0.1
$ ip -4 a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    inet 128.82.7.233/24 brd 128.82.7.255 scope global ens160
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
```



# Loopback Address: localhost

```
$ cat /etc/hosts
127.0.0.1 localhost.localdomain    localhost

# The following lines are desirable for IPv6 capable hosts
::1       localhost6.localdomain6  localhost6
::1       localhost                ip6-localhost       ip6-loopback
```

# 127.0.0.1 vs. 0.0.0.0

- 127.0.0.1 (or localhost)
  - Listening on loopback interface only (unless tunneled)
- 0.0.0.0
  - Listening on all network interfaces
  - Not a resolvable address

Processes running in Docker containers listening on loopback interface will not be accessible from outside of the container, run them on 0.0.0.0 instead.

# Hello Server: Python

```python
#!/usr/bin/env python3

import socket

HOST = "0.0.0.0"
PORT = 8080

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
s.bind((HOST, PORT))
s.listen()
print(f"Listening on {HOST}:{PORT} for HTTP connections")

while True:
    conn, addr = s.accept()
    print(f"Connected to {addr}")
    conn.sendall(b"Hello from server\n")
    conn.close()
```

# Hello Server: Ruby

```ruby
#!/usr/bin/env ruby

require "socket"

host = "0.0.0.0"
port = 8080

socket = TCPServer.new(host, port)
puts "Listening on #{host}:#{port} for HTTP connections")

loop do
    client = socket.accept
    puts "Connected to #{client}"
    client.write("Hello from server\n")
    client.close
end
```

# Concurrency

```ruby
#!/usr/bin/env ruby

require "socket"

host = "0.0.0.0"
port = 8080

socket = TCPServer.new(host, port)
puts "Listening on #{host}:#{port} for HTTP connections")

loop do
    Thread.start(socket.accept) do |client|
        puts "Connected to #{client}"
        client.write("Hello from server\n")
        client.close
    end
end
```

# Run the Hello Server

```
$ ./server.py
Listening on 0.0.0.0:8080 for HTTP connections
Connected to ('127.0.0.1', 46930)
```

```
$ telnet localhost 8080
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Hello from server
Connection closed by foreign host.
```