TNE30019/TNE80014 – Unix for Telecommunications

Sockets API

Dr. Jason But

Swinburne University

Outline

Sockets API

- Access to OS network transport protocols
- Initially developed by Berkeley University
- Many implementations but all conform to the standard API
- Allows network code to be cross-platform

Blocking vs Non-blocking Sockets

- By default sockets are **blocking**
- Execution of many functions will block until data can be sent/received, connections are established etc.
- Need multi-threading to support concurrency
- Complicates things, no gain on single processor core

- Can we get "concurrency" without multi-threading?
- Set socket to **non-blocking** (socket option)
- With non-blocking sockets functions will never block but instead return error E_WOULDBLOCK
- Use `select()` and others to check for unblocking

Basic Socket API Functions

socket()

Create new socket in system

- Allocates resources and returns handle (file descriptor) to newly created socket
- Descriptor is used in all future function calls on socket
- Parameters include protocol (e.g. TCP) and network type (e.g. IP version 4)

bind()

Attach socket to particular IP address and TCP/UDP port number

- Can bind to specified or wildcard IP Address
- Can bind to specific or system selected port number

Basic Socket API Functions cont'd

listen()

Tell system to begin listening for incoming connections on socket

- Only valid for TCP sockets – usually server side
- OS performs entire TCP connection handshake

accept()

Accept connections on listening socket (may block)

- TCP handshake has already been completed
- Returns socket descriptor to new socket created
- Original socket still listening for new connections
- New socket is bound to same IP address/port number pair as listening socket

Basic Socket API Functions cont'd

connect()

Attempt to establish connection to remote socket (may block)

- Parameters include IP address/port number pair of remote socket
- For UDP this only stores IP address/port number pair

send()

Send data over connected socket (may block)

- Socket must be connected to remote

sendto()

Send data over unconnected socket (may block)

- UDP sockets if socket is not connected
- Specify destination socket details for each datagram

Basic Socket API Functions cont'd

recv()

Retrieve data from connected socket (may block)

- Socket must be connected to remote

recvfrom()

Retrieve data from connected or unconnected socket (may block)

- Receive message as well as information about sender (source IP address and port)

close()

Close and release the socket connection

- Releases any allocated resources

Basic Socket API Functions cont'd

setsockopt()

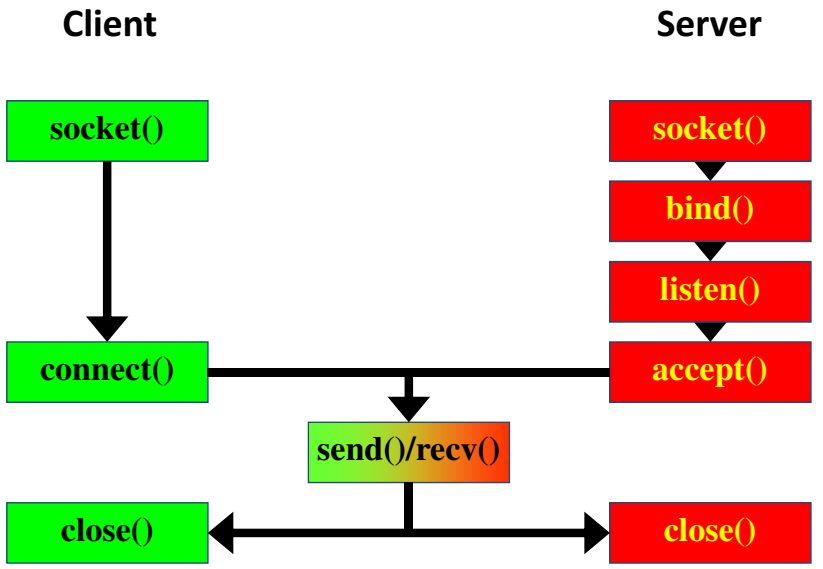
Set options on socket after it has been created with socket()

- Set size of socket buffer for input/output
- Enable reception of timestamps with datagrams
- Many more

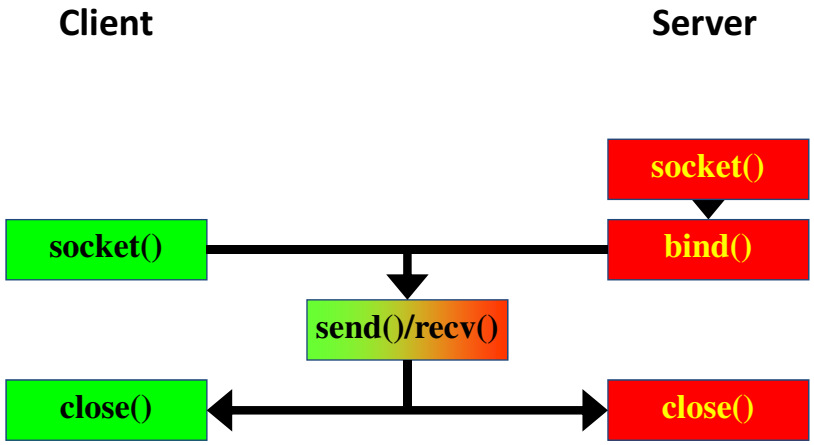
getsockopt()

Query options set on socket

Socket API Usage – TCP



Socket API Usage – UDP



Other Socket Programming Related Functions

getaddrinfo(), getnameinfo()
deprecated: gethostbyname(), gethostbyaddr()

Get object describing Internet host based on host name or address

- Deprecated functions are non-reentrant and need to be protected when called from concurrent threads

select()

Blocks on list of sockets until some sockets would be unblocked or set timer expires

- Useful for single-threaded applications with concurrency and non-blocking sockets

inet_XXXX()

Convert IP addresses from one representation to another, e.g. convert binary to string