EECS 489 - WN 23

Discussion 4

Assignment-2

Assignment 1 ended.
Assignment 2 is out. Due date: **02/24 2023, 11:59 PM**

Much harder than A-1.

START EARLY!

Hosted in GitHub under https://github.com/eecs489

Group formation: https://forms.gle/YZRAT8zPccym4ond6

Today's Plan

- A1 Recap
- Lecture Based Questions - DNS
- A2 Prep: select()

A1 Recap: recv()

```
ssize_t recv(int sockfd, const void * buf, size_t len, int flags);

// Example
ssize_t bytes_recvd = recv(sockfd, buffer, MSG_SIZE, 0);
// here bytes_recvd is not always == to MSG_SIZE

// But we also can do
ssize_t bytes_recvd = recv(sockfd, buffer, MSG_SIZE, MSG_WAITALL);
// here bytes_recvd is == MSG_SIZE, but we block until then
```

A1 Recap: Correct?

```
int client_sd;
char buffer[1000];
ssize_t total = 0;
while (true) {
    ssize_t recvd_bytes = recv(client_sd, buffer, 1000, 0);
    total += recvd_bytes;
    // Expecting a designation that client is done sending.
    if (buffer[0] == 'F') {
        break;
    }
}
```

A1 Recap: Better

```
int client_sd;
char buffer[1000];
ssize_t total = 0;
while (true) {
    ssize_t recvd_bytes = recv(client_sd, buffer, 1000, MSG_WAITALL);
    total += recvd_bytes;
    // Expecting a designation that client is done sending.
    if (buffer[0] == 'F') {
        break;
    }
}
```

Lecture Based Questions - Q1

Suppose EECS has a DNS server for all computers in the department.

How could you determine if an external web site was likely to be accessed from another computer in EECS a couple of seconds ago?

Lecture Based Questions - Q1

Suppose EECS has a DNS server for all computers in the department.

How could you determine if an external web site was likely to be accessed from another computer in EECS a couple of seconds ago?

Perform two consecutive dig queries and compare the query time.

Lecture Based Questions - Q2

Suppose you are trying to access the page web.eecs.umich.edu/course/eecs489. You are connected to your home WiFi with its own local DNS (from your ISP), and are not connected to MWireless/umich's network. Give the order of name servers queried over time and their replies.

Assume:

- No prior caching, Recursive name resolution
- umich.edu and eecs.umich.edu are in separate zones
- eecs.umich.edu is authoritative for all hostnames ending in .eecs.umich.edu

Lecture Based Questions - Q2

Suppose you are trying to access the page web.eecs.umich.edu/course/eecs489. You are connected to your home WiFi with its own local DNS (from your ISP), and are not connected to MWireless/umich's network. Give the order of name servers queried over time and their replies.

Assume:

- No prior caching, Recursive name resolution
- umich.edu and eecs.umich.edu are in separate zones
- eecs.umich.edu is authoritative for all hostnames ending in .eecs.umich.edu

Queries: local DNS -> root -> edu -> umich -> eecs

Replies: (eecs -> umich), (umich -> edu), (edu -> root), (root -> local DNS)

Lecture Based Questions - Q3

Suppose you are trying to access the page web.eecs.umich.edu/course/eecs489. You are connected to your home WiFi with its own local DNS (from your ISP), and are not connected to MWireless/umich's network. Give the order of name servers queried over time and their replies.

Assume:

- No prior caching, Iterative name resolution
- umich.edu and eecs.umich.edu are in separate zones
- eecs.umich.edu is authoritative for all hostnames ending in .eecs.umich.edu

Lecture Based Questions - Q3

Suppose you are trying to access the page web.eecs.umich.edu/course/eecs489. You are connected to your home WiFi with its own local DNS (from your ISP), and are not connected to MWireless/umich's network. Give the order of name servers queried over time and their replies.

Assume:

- No prior caching, Iterative name resolution
- umich.edu and eecs.umich.edu are in separate zones
- eecs.umich.edu is authoritative for all hostnames ending in .eecs.umich.edu

Queries: (local DNS -> root), (local DNS -> edu), (local DNS -> umich), (local DNS -> eecs) Replies: (root -> local), (edu -> local), (umich -> local), (eecs -> local)

A2 Prep: A1 Reflection

Question:

- Can our A1 server code handle connections from multiple clients?
- How might we implement this?
  - Threads -- You may have used this in 482
  - And ?

A2 Prep: select()

select(): I/O Multiplexing

Allow a program to monitor multiple file descriptors, waiting until one or more of the file descriptors becomes "ready" for some class of I/O operation.

A2 Prep: select()

```
#include <sys/select.h>
int select(int nfds, fd_set *readfds, fd_set *writefds,
           fd_set *exceptfds, struct timeval *timeout);


// For A2, we only care about readfds, also no timeout
// so typically we invoke with
select(FD_SETSIZE, &readfds, NULL, NULL, NULL);
```

A2 Prep: macros

Some useful macros:

```
// Add fd to the set
void FD_SET(int fd, fd_set *set);
 // Remove fd from the set
void FD_CLR(int fd, fd_set *set);
// Return true if fd is in set, might not be after select()
int FD_ISSET(int fd, fd_set *set);
// Clear all entries from set
void FD_ZERO(fd_set *set);
```

A2 Prep: A demo

Discussion-4/demo

Thanks

Have a good one!