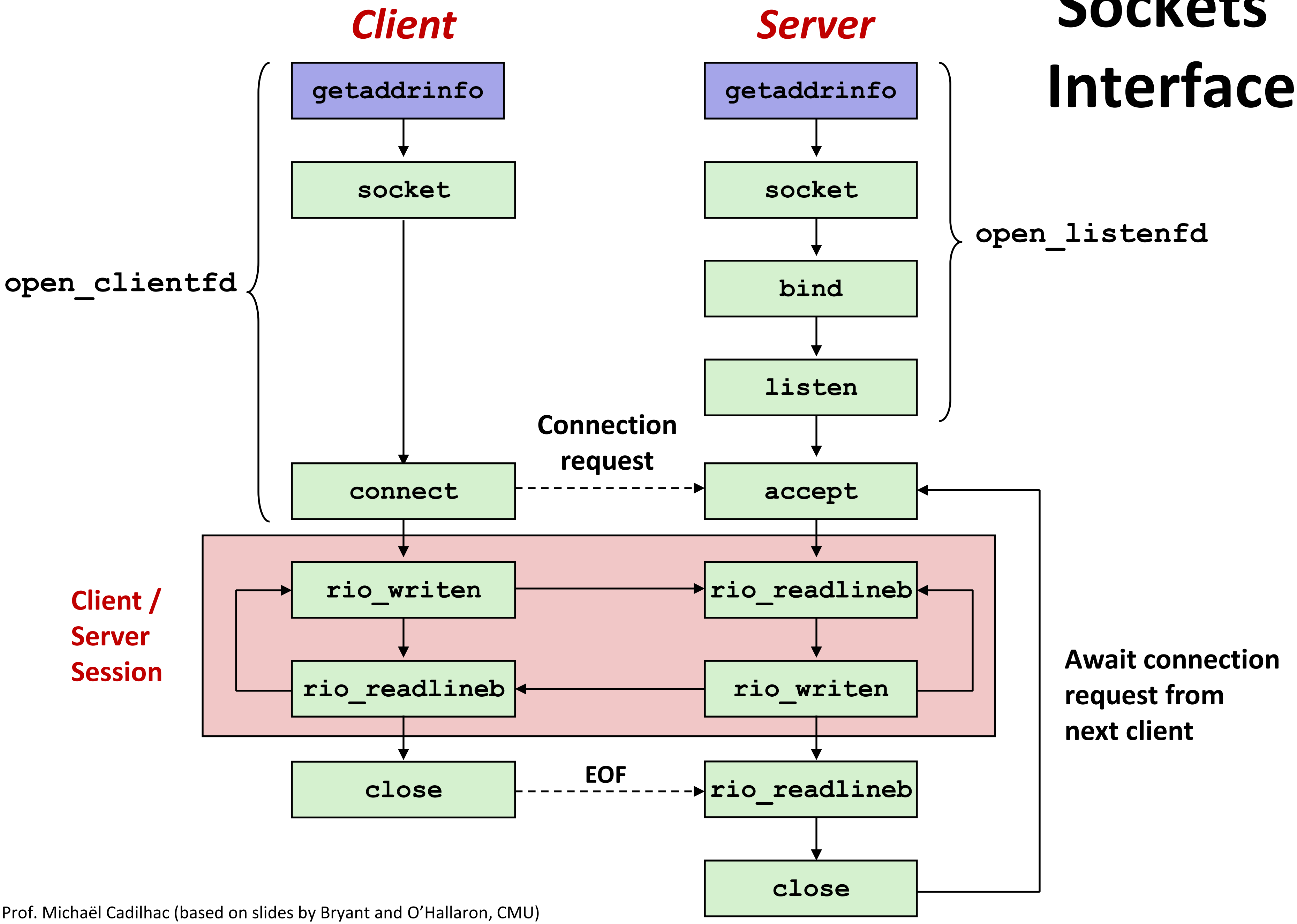
Network Programming:
Basic client/server application:
1. `getaddrinfo`

Sockets Interface



Host and Service Conversion: `getaddrinfo`

- **`getaddrinfo(3)` is the modern way to convert string representations of hostnames, host addresses, ports, and service names to socket address structures.**
  - Replaces obsolete `gethostbyname` and `getservbyname` funcs.
  - Resolving implemented by libc, *not* kernel (which implements TCP/IP).

- **Advantages:**
  - Reentrant (can be safely used by threaded programs).
  - Allows us to write portable protocol-independent code
    - Works with both IPv4 and IPv6

- **Disadvantages:**
  - Somewhat complex
  - Fortunately, a small number of usage patterns suffice in most cases.

Host and Service Conversion: `getaddrinfo`

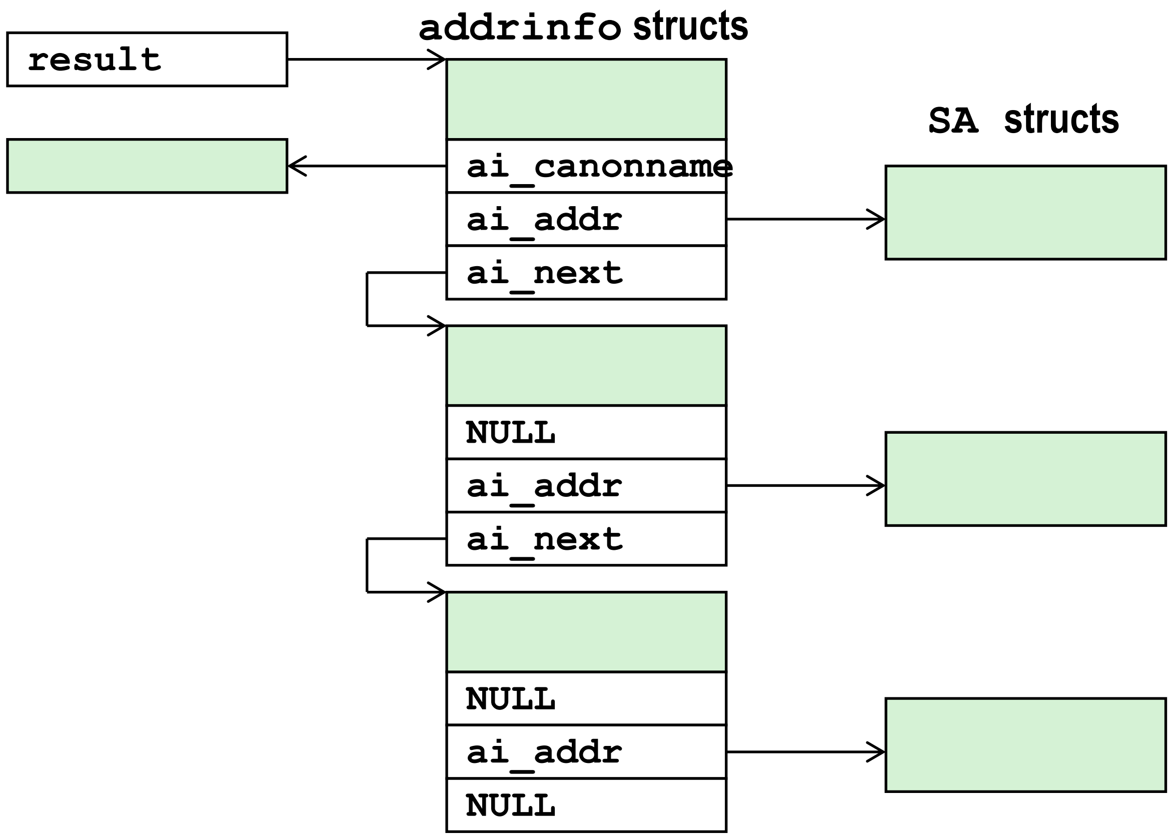
```
int getaddrinfo(const char *host,            /* Hostname or address */
                const char *service,         /* Port / service name */
                const struct addrinfo *hints,/* Input parameters */
                struct addrinfo **result);   /* Output linked list */

void freeaddrinfo(struct addrinfo *result);  /* Free linked list */

const char *gai_strerror(int errcode);       /* Return error msg */
```

- **Given `host` and `service`, `getaddrinfo` sets `result` to point to a linked list of `addrinfo` structs, each of which points to a corresponding SA, and which contains arguments for the sockets interface functions.**
- **Helper functions:**
  - `freeadderinfo` frees the entire linked list.
  - `gai_strerror` converts error code to an error message.

Linked List Returned by `getaddrinfo`



- Clients: walk this list, trying each socket address in turn, until the calls to `socket` and `connect` succeed.
- Servers: walk the list until calls to `socket` and `bind` succeed.

addrinfo Struct

```
struct addrinfo {
    int              ai_flags;     /* Hints argument flags */
    int              ai_family;    /* First arg to socket function */
    int              ai_socktype;  /* Second arg to socket function */
    int              ai_protocol;  /* Third arg to socket function  */
    char            *ai_canonname; /* Canonical host name */
    size_t           ai_addrlen;   /* Size of ai_addr struct */
    SA              *ai_addr;      /* Ptr to socket address structure */
    struct addrinfo *ai_next;      /* Ptr to next item in linked list */
};
```

- **Each `addrinfo` struct returned by `getaddrinfo` contains arguments that can be passed directly to `socket` function.**
- **Also points to a `SA` struct that can be passed directly to `connect` and `bind` functions.**

Host and Service Conversion: getnameinfo

- **getnameinfo is the inverse of getaddrinfo, converting a socket address to the corresponding host and service.**
  - Replaces obsolete gethostbyaddr and getservbyport funcs.
  - Reentrant and protocol independent.

```
int getnameinfo(const SA *sa, socklen_t salen, /* In: socket addr */
                char *host, size_t hostlen,    /* Out: host */
                char *serv, size_t servlen,    /* Out: service */
                int flags);                    /* optional flags */
```

Conversion Example

```
#include "csapp.h"

int main(int argc, char **argv)
{
    struct addrinfo *p, *listp, hints;
    char buf[MAXLINE];
    int rc, flags;

    /* Get a list of addrinfo records */
    memset(&hints, 0, sizeof(struct addrinfo));
    hints.ai_family = AF_INET;       /* IPv4 only, remove for any */
    hints.ai_socktype = SOCK_STREAM; /* TCP Connections only */
    if ((rc = getaddrinfo(argv[1], NULL, &hints, &listp)) != 0) {
        fprintf(stderr, "getaddrinfo error: %s\n", gai_strerror(rc));
        exit(1);
    }
```

hostinfo.c

Conversion Example (cont)

```
    /* Walk the list and display each IP address */
    flags = NI_NUMERICHOST; /* Display address instead of name */
    for (p = listp; p; p = p->ai_next) {
        Getnameinfo(p->ai_addr, p->ai_addrlen,
                    buf, MAXLINE, NULL, 0, flags);
        printf("%s\n", buf);
    }

    /* Clean up */
    Freeaddrinfo(listp);

    exit(0);
}
```

hostinfo.c

Running hostinfo

Running hostinfo

```
$ ./hostinfo localhost
127.0.0.1
```

Running hostinfo

```
$ ./hostinfo localhost
127.0.0.1

$ ./hostinfo mc.cdm.depaul.edu
216.220.181.74
```

Running hostinfo

```
$ ./hostinfo localhost
127.0.0.1

$ ./hostinfo mc.cdm.depaul.edu
216.220.181.74

$ ./hostinfo twitter.com
199.16.156.230
199.16.156.38
199.16.156.102
199.16.156.198
```