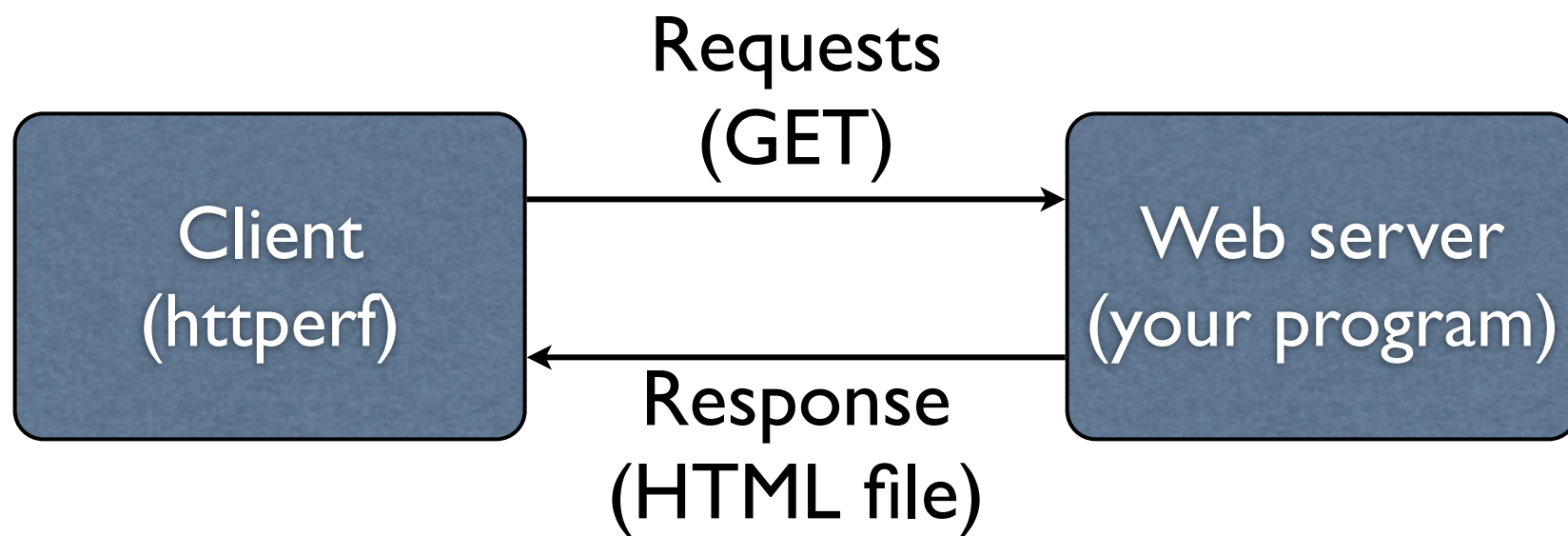


# Measuring Web Server Performance

Homework 2



# httperf

<ftp://ftp.hpl.hp.com/pub/httperf/httperf-0.9.0.tar.gz>

```
httperf --hog --server=127.0.0.1 --num-conns=10
```

```
httperf --hog --server=127.0.0.1  
--wsess=100,50,10 --burst-len=5
```

# Simple Web Server

```
listenfd = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
servaddr.sin_family = PF_INET;
servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
servaddr.sin_port = htons(80);
bind(listenfd, (struct sockaddr *)&servaddr, sizeof(servaddr));
listen(listenfd, 1000);
for (;;) {
    len = sizeof(cliaddr);
    connfd = accept(listenfd, (struct sockaddr *)&cliaddr, &len);
    read_request(connfd); /* read until "\r\n\r\n" */
    write(connfd, "200 OK HTTP/1.0\r\n\r\n"
        "That's it. You're welcome.", 19+27);
    close(connfd);
}
```

# Concurrent Server

```
listenfd = socket( ... );  
...  
bind(listenfd, ... );  
listen(listenfd, ... );  
for (;;) {  
    connfd = accept(listenfd, ... );  
    if ((pid = fork()) == 0) {  
        close(listenfd);  
        ... /* process the request */  
        close(connfd);  
        exit(EXIT_SUCCESS);  
    }  
    close(connfd);  
}
```

# Concurrent Server with File Transfer

```
listenfd = socket( ... );  
...  
bind(listenfd, ... );  
listen(listenfd, ... );  
for (;;) {  
    connfd = accept(listenfd, ... );  
    if ((pid = fork()) == 0) {  
        close(listenfd);  
        ... /* read the HTTP request */  
        ... /* send a 10-KB file */  
        close(connfd);  
        exit(EXIT_SUCCESS);  
    }  
    close(connfd);  
}
```

# What to Turn In

- Source code for three servers (simple.c, concurrent.c, file.c)
- Your environment info (CPU, Memory, OS)
- Graphs for the simple and concurrent servers
  - Throughput vs. # of connections
  - 1–2500 connections