# GetFile Server

- Functions in blue are provided as skeleton code.
- This diagram shows only normal code path for simplicity. Make sure you handle every possible error cases.

## Boss thread

```
parse command line arguments

// create a server instance and set parameters
gfserver_t *gfs;
gfs = gfserver_create();
gfserver_set_port();
gfserver_set_maxpending();
gfserver_set_handler();
gfserver_set_handlerarg();

initialize global resource such as request queue, mutexes, and
condition variables

start worker threads

// start the server
gfserver_serve();
    create a socket and bind to an endpoint
    listen for incoming connections
    repeat forever
        accept a new client
        read and parse header according to the GetFile protocol
        // enqueue the request
        gfs->handler()
```

## Worker threads

```
repeat forever
    dequeue a request from the request queue
    fildes = content_get(path)
    gfs_sendheader(ctx, GF_OK, file_len);
    while (until finished)
        gfs_send(ctx, buffer, len);
```

# GetFile Client

## Boss thread

```
parse command line parameters

gfc_global_init();

initialize global resource such as request queue, mutexes, and
condition variables

create and run worker threads

load workload file

for (nrequests)
    select a file path from the workload to make a request
    enqueue the file path to the queue

wait for all requests to be served

gfc_global_cleanup();
```

## Worker threads

```
repeat until all requests are processed
    wait until there is an item in the queue
    dequeue an item

    // create a request
    gfcrequest_t *gfr = gfc_create();
    gfr = gfc_create();
    gfc_set_server();
    gfc_set_path();
    gfc_set_port();
    gfc_set_writefunc();
    gfc_set_writearg();

    // send the request and receive the response
    // according to the GetFile protocol
    gfc_perform();

    // check the status of the request
    gfc_get_status();
```