System Calls

CS 360 Internet Programming

Daniel Zappala

Brigham Young University
Computer Science Department

Web Server Tasks

1. read and parse the HTTP request message
   - use supplied HTTP parser
2. translate the URI to a file name
   - use the Host header to find the host name
   - configuration file gives the root directory for each host served by the web server
   - append the URI path to the root directory to get the complete path
3. determine whether the request is authorized
   - check file permissions or other authorization procedure
4. generate and transmit the response
   - error code or file or results of script
   - must be a valid HTTP message with appropriate headers

Checking File Permissions

- call open() to determine whether you can access the file
- return value of -1 indicates failure
  - errno == EACCESS indicates you didn't have the right permissions (403 Forbidden)
  - errno == ENOENT indicates the file doesn't exist (404 Not Found)
  - any other error indicate a server failure (500 Internal Server Error)

Accessing File Attributes

- use fstat() to access file size and last modification time
- use in Content-Length and Last-Modified headers

```
1  int fstat(int filedes, struct stat *buf);
2
3  filedes = file descriptor from open()
4  buf = pointer to structure to hold statistics
```

- on success returns zero
- on error returns -1 and sets errno
- stat(2) provides details on the structure
- the st_size member gives the file size in bytes
- the st_mtime member gives the time of last modification

Sending a File

```
1  ssize_t sendfile(int out_fd, int in_fd, off_t *offset,
2                   size_t count);
3
4  out_fd = socket file descriptor
5  in_fd = file descriptor of file to send
6  offset = NULL
7  count = size of file in bytes (from fstat())
```

- on success, returns number of bytes written
- on error returns -1 and sets errno

Getting the Time

```
1  time_t time(time_t *t);
```

- returns the time since the Epoch (00:00:00 UTC, January 1, 1970), measured in seconds
- usually pass NULL as the argument to time()
- see time(2) for details

Converting to GMT

```
1  struct tm *gmtime(const time_t *timep);
2
3  timep = time given by time() or fstat()
```

- on success returns struct tm pointer
- on error returns NULL
- details of structure given in ctime(3)

Converting to RFC 822, 1123 Time Format

- the recommended date format for HTTP
- used in the Date and Last-Modified headers

```
1  size_t strftime(char *s, size_t max, const char *format,
2                  const struct tm *tm);
3
4  s = buffer to store ascii translation of time
5  max = size of buffer
6  format = format string
7  tm = pointer to time structure
```

- on success returns number of characters stored in buffer
- on error returns 0
- the magic format string for RFC 1123 time:

```
1  %a, %d %b %Y %H:%M:%S GMT
```

From Time to Time

```
1  string date(time_t t)
2  {
3      struct tm *gmt;
4      char buf[200];
5
6      memset(buf,0,200);
7      gmt = gmtime(&t);
8      if (gmt == NULL)
9          return "";
10     if (strftime(buf,sizeof(buf),
11                  "%a, %d %b %Y %H:%M:%S GMT",gmt) == 0)
12         return "";
13     return string(buf);
14 }
```

- pass in either time(NULL) for current time or the file modification time from fstat()