HTTP Parsing

pallabi ghosh (PALLABIG@ANDREW.cmu.edu)

15-441 Computer Networks

Recitation 3
Feedback for Checkpoint 1

15-441: Computer Networks
Basic code structure

```python
bind(), listen()
while true select(r_fds, w_fds)
    for fd in all current fds
        if fd is ready to accept
            accept() → new_fd
        if fd is ready to read
            recv() → buffer[fd]
        if fd is ready to write && buffer[fd]
            buffer[fd] → send()
```
Timeout error: why is it slow?

bind(), listen(s, 5?)
while true select(r_fds, w_fds_with_data)
  for fd in all current fds
    if fd is ready to accept
      accept() → new_fd
    if fd is ready to read
      recv() → buffer[fd]
    if fd is ready to write && buffer[fd]
      buffer[fd] → send()
About Makefile

• At least have `make` and `make clean` working
• `make`: Nothing to be done for 'all'?  
  • `all`: lisod
  • `lisod`:  
    ```bash
    @gcc echo_server.c -o lisod -Wall -Werror
    ```  
  • `clean`: `rm -f lisod`
  – Added .PHONY lisod at the beginning
About code style

• Use *meaningful comment* for git commit
• And in your source code of course
• *No magic number* in your code
• Do not version-control your *.o or lisod
• Do not print debug info
Your submission should be *camera-ready*
PJ 2 CP 2

• GET, POST and HEAD requests
• Read the documents first to understand the rules.
• RFC 2616
• Check the annotated RFC
HTTP request

• Request =
  Request-Line ;
  *( ( general-header ;
    request-header ;
    entity-header ) CRLF ) ;
  CRLF [ message-body ] ;
HTTP request

• Request =
GET /path/file.html HTTP/1.1
Host: www.host1.com:80
User-Agent: MyBrowser/1.0
[blank line here]
HTTP request

• Request =

POST /path/script.cgi HTTP/1.1
Host: www.host1.com:80
User-Agent: MyBrowser/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 32

name=mukerjee&age=25
HTTP response

•Response =
  Status-Line ;
  *((general-header ;
    response-header ;
    entity-header ) CRLF) ;
  CRLF [ message-body ] ;
HTTP response

• Response =
HTTP/1.1 200 OK
Date: Fri, 20 Sep 2013 23:59:59 GMT
Content-Type: text/html
Content-Length: 1354

<html>
...
</html>
Tools

• Use tools to look at these requests and see the pattern for real
  – Wireshark
    ◦ http://www.wireshark.org/
  – Use the dumper code (dumper.py)
  – Play with the headers
  ◦ Tamperdata
  ◦ Poster
  – Online tools
    ◦ http://web-sniffer.net/
Sample GET request

GET request was done on this page.

Do you see the pattern?
Sample POST request

POST method called while submitting this form. Can you identify the values that were submitted?
HEAD is similar to GET! (just without the data)
Sample GET response

| Frame 29: 509 bytes on wire (4072 bits), 509 bytes captured (4072 bits) |
| Transmission Control Protocol, Src Port: http (80), Dst Port: 52444 (52444), Seq: 1, Ack: 403, Len: 455 |

**HTTP/1.1 200 OK**

- **Date:** Wed, 28 Sep 2011 04:34:20 GMT
- **Server:** Apache/2.2.16 (Ubuntu)
- **Last-Modified:** Wed, 28 Sep 2011 04:30:52 GMT
- **ETag:** "14391f-79-4adf8db71b700"
- **Accept-Ranges:** bytes
- **Vary:** Accept-Encoding
- **Content-Encoding:** gzip
- **Content-Length:** 116
- **Keep-Alive:** timeout=15, max=100
- **Connection:** keep-Alive
- **Content-Type:** text/html

Content-encoded entity body (gzip): 116 bytes -> 121 bytes

**Line-based text data:** text/html

```html
<html>
<head>
<title>Athula's apache webserver</title>
</head>
<body>
<h1>It works!</h1>
welcome to Athula's page
</body>
</html>
```
Sample POST response

For this checkpoint you just have to send status code. More on dynamic content later!
Minimal Implementation

• Status codes
  – 200_OK
  – 404_NOT_FOUND
  – 411_LENGTH_REQUIRED
  – 500_INTERNAL_SERVER_ERROR
  – 501_NOT_IMPLEMENTED
  – 503_SERVICE_UNAVAILABLE
  – 505_HTTP_VERSION_NOT_SUPPORTED
Minimal Implementation

- General Headers
  - Connection
  - Date
- Response Headers
  - Server should always be: Liso/1.0
- Entity Headers
  - Content-Length
  - Content-Type
  - Last-Modified
Be careful with the buffers!

• Requests may straddle multiple recv calls
  – Need to maintain state information.

• If request header size > 8192 bytes
  – For now, send error message and disconnect.
Flex Tutorial

15-441: Computer Networks
What && Why

• Flex (fast lexical analyzer generator) is a lexer. It scans strings to identify keywords, numbers and tokens.

• Your http parser need a lexer.

• You have two options for a lexer:
  A. hand write your lexer
  B. flex generates C code of your lexer
How

• Some knowledge of regular expression
• Write your lex file
• Compile it to a c file
• Compile your c file

flex -o foo.c foo.lex
gcc foo.c
Example
All questions?