

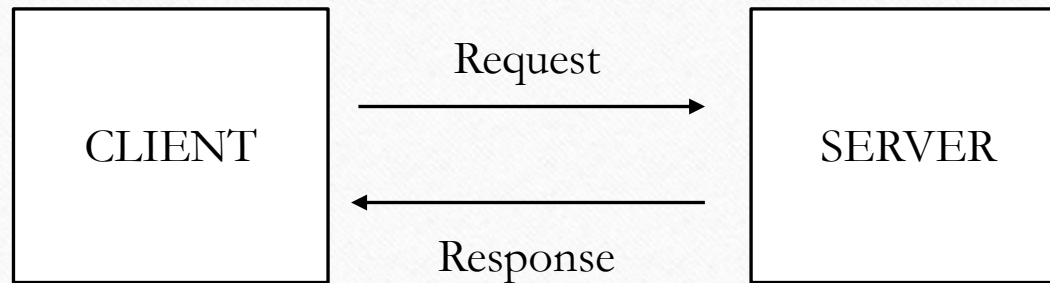
Project 3: HTTP Pseudo Streaming

Carnegie Mellon University

March 20 2020
Deeptha Anil Kumar

BASICS

- What is a HTTP Server ?
 - A computer program or a software that provides contents needed for web browsers like images, videos, text file etc.
 - It's a Request/Response Model:



HTTP Protocol - Hypertext Transfer Protocol

- Is an application layer protocol
- Used to virtually transmit files and other data on the World Wide Web like HTML files, image files, query results.
- Uses TCP/IP

URI – Universal Resource Identifier

- An URI is a sequence of characters that identifies a logical or physical resource.

Example:

- <https://www.cmu.edu/>
- <https://www.cmu.edu/academics/index.html>

Project Implementation



- For our project: Localhost must act as webserver
Example: http://localhost:8012/index.html

HTTP/1.1

Will need to implement HTTP/1.1 GET Method for:

- 404 Not Found
- 200 OK
- 206 Partial Content – Range Requests

<http://techslides.com/demos/sample-videos/small.mp4>

Very Important:

- Your server should keep the connection alive after receiving a request

Headers Cookies Params Response Timings Stack Trace

Request URL: http://localhost:8080/video.ogg
Request method: GET
Remote address: 127.0.0.1:8080
Status code: 200 OK ?
Version: HTTP/1.1
Referrer Policy: no-referrer-when-downgrade Edit and Resend

Filter headers

Response headers (223 B) Raw headers

- Accept-Ranges: bytes
- Connection: keep-alive
- Content-length: 1734919
- Content-type: video/ogg
- Date: Sun Apr 21 21:19:23 EDT 2019
- Last-Modified: Wed, 10 Apr 2019 21:15:58 EDT
- Server: Java HTTP Server

Request headers (333 B) Raw headers

- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
- Accept-Encoding: gzip, deflate
- Accept-Language: en-US,en;q=0.5
- Connection: keep-alive
- Host: localhost:8080
- Upgrade-Insecure-Requests: 1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; ...) Gecko/20100101 Firefox/66.0

Content-Type

- Below are example of content type that your web server needs to handle:
 - text/plain (.txt)
 - text/html (.htm, .html)
 - image/gif (.gif)
 - image/jpeg (.jpg, .jpeg)
 - video/webm (.mp4, .webm, .ogg)
- Refer Handout to cover all types

Project 3: Deliverables

- `vodserver <#port no>`
 - By default all the content files will need to be stored in a directory called “content” relative to the directory from where the server files are called.
 - `/.../project1 $> vodserver #port`
 - On browser: `http://<server-ip-address>:<server-port>/video/video.webm`
 - folder from where files needs to be accessed:
`/.../project1/content/video/video.webm`
- Makefile
- Design document

Grading

Items	Points (100 – 441 + 110 – 741)
Logistics	
- Successful submission & compilation	10
- Design documents	10
Server Correctness	
- Handle standard HTTP GET	10
- Handle HTTP error (404)	10
- Handle HTTP Byte-range request	20
- Play video from browser	10
- Allowing video seek from browser	10
Server Performance	
- < 10% CPU utilization when idle	5
- Handle 10 clients simultaneously	15
Required for 18-741 only (bonus for 18-441)	
- Handle 5000 concurrent clients!	10

Closing Comments

- Firefox 9+ Browser will be used for testing.
- If you are using Chrome, the browser will use multiple connections for a single video transfer.
- Understand the headers that need to be created.
- Your server should be able to easily handle concurrent connections.
- More details in the write up.
- Use Piazza/Office Hours for any clarifications