15-213 Recitation
Networking and Proxies

Your TAs

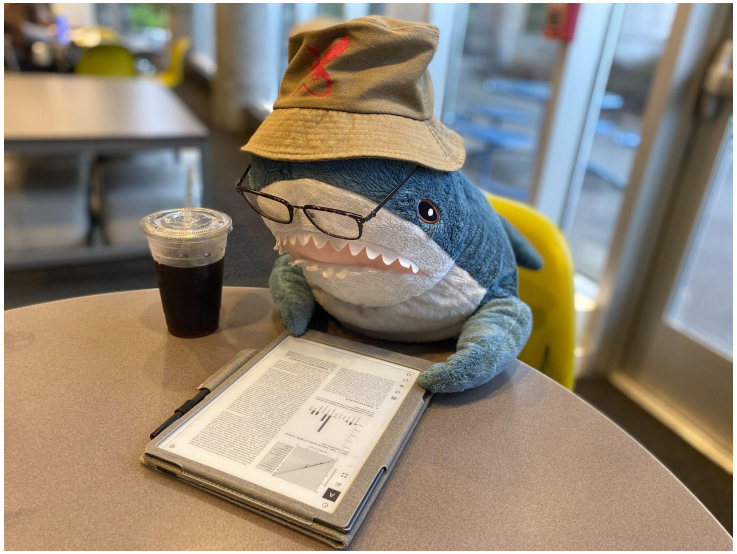
Friday, November 15th

Reminders

- **`tshlab`** is due ***Tuesday (November 19th)***
- **`proxylab`** will be released on the same day
  - Due *November 26th*
- **`sfslab`** will be released before Thanksgiving
  - Due *December 5th*
- Code Reviews:
  - **`malloc`** Final

Apply to be a TA!

- TA Applications are open! See Piazza @1104 :-)
  - First round of interviews happening in 2-3 weeks!
- What qualifications are we looking for?
  - Decent class performance
  - Strong communication skills
  - Reasonable ability to gauge schedule and responsibilities



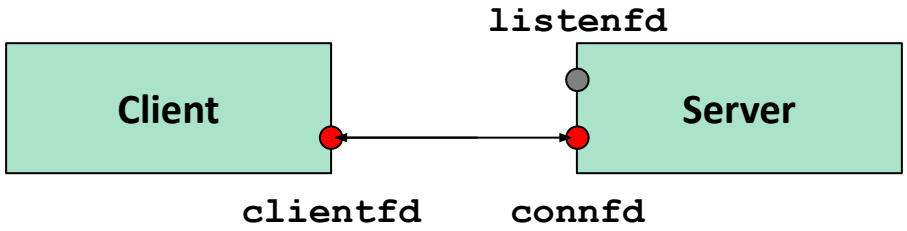
Agenda

- Review: Networking
- Proxy Lab
  - What is a proxy?
  - Getting started
  - Debugging with gdb and PxyDrive

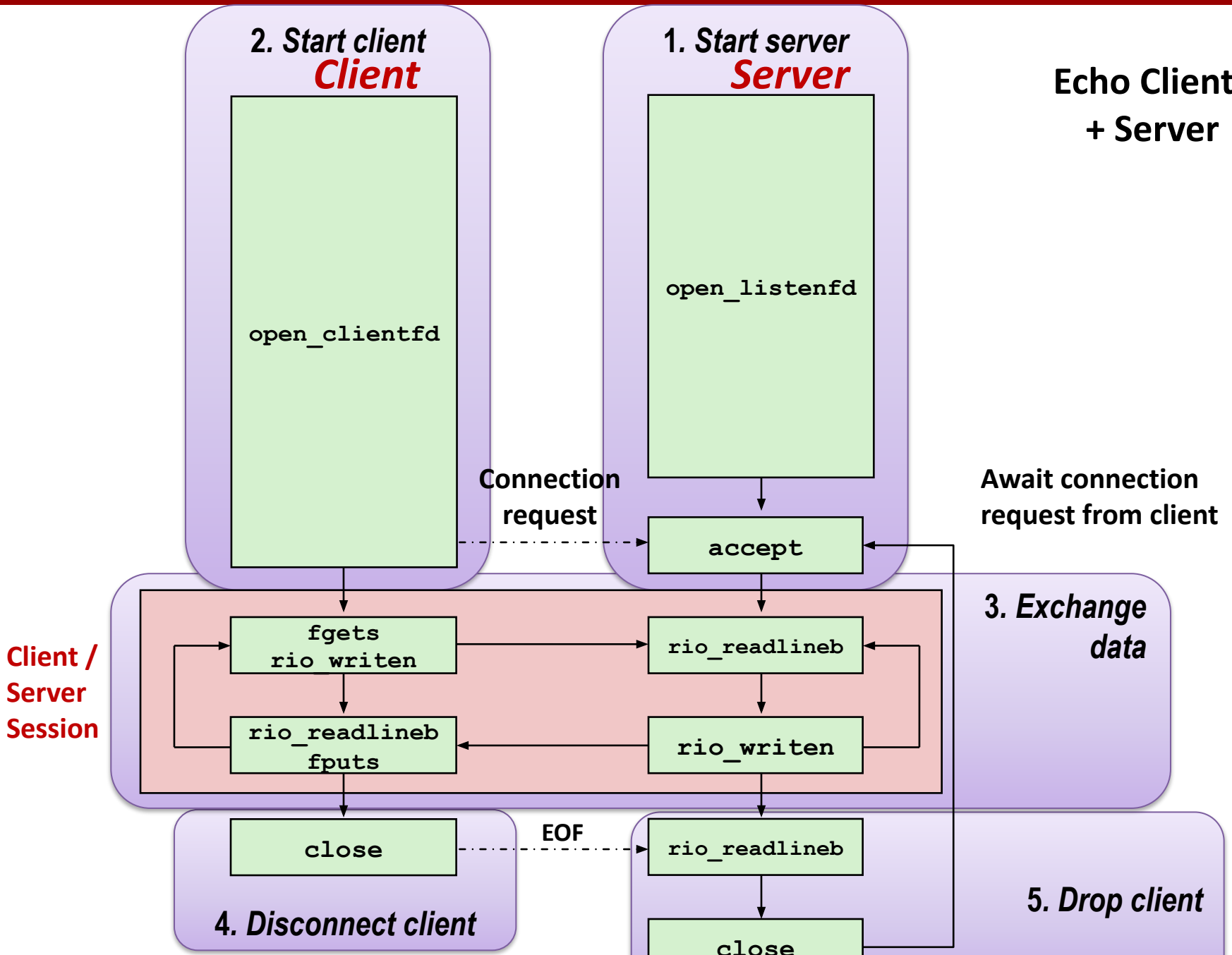
Review: Networking

Networking Refresher

- UNIX File Abstraction: communicate over the network by reading from and writing to *file descriptors*.
- **`proxylab`** – Underlying logic for setting up client and server file descriptors handled for you in **`csapp.h`**:
  - **`int open_clientfd(char *host, char *port)`**
  - **`int open_listenfd(char *port)`**
- Then just send and receive data over those file descriptors with the **`rio`** package.



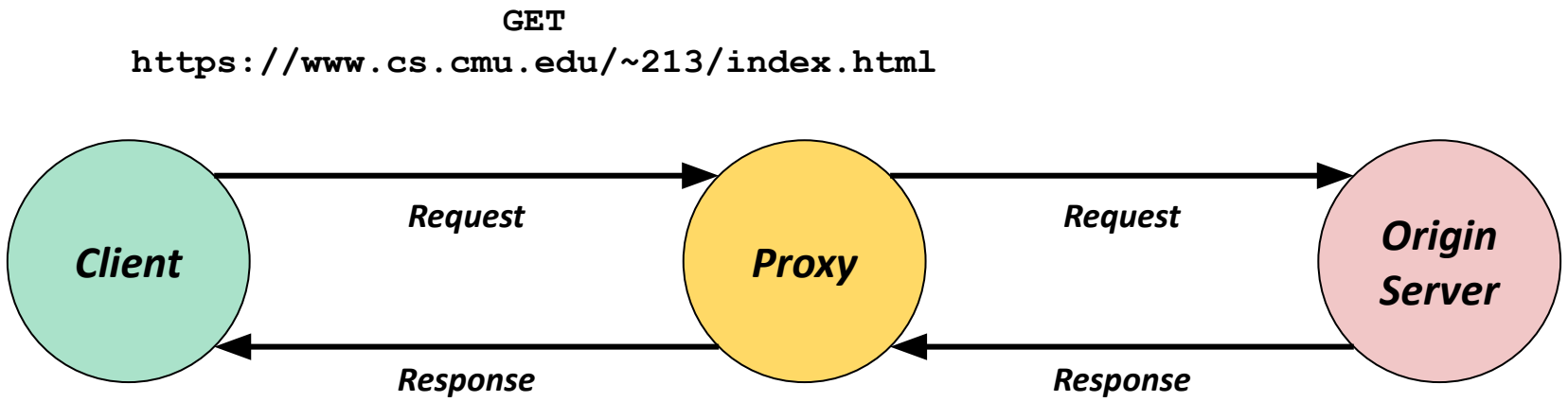
Echo Client + Server



Proxy Lab

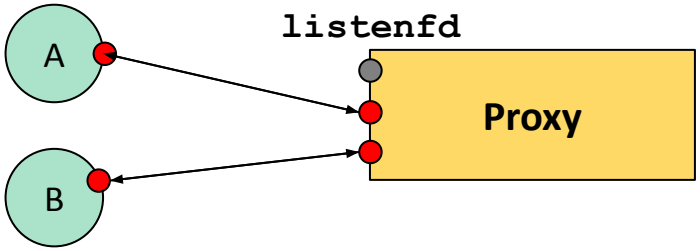
What is a proxy?

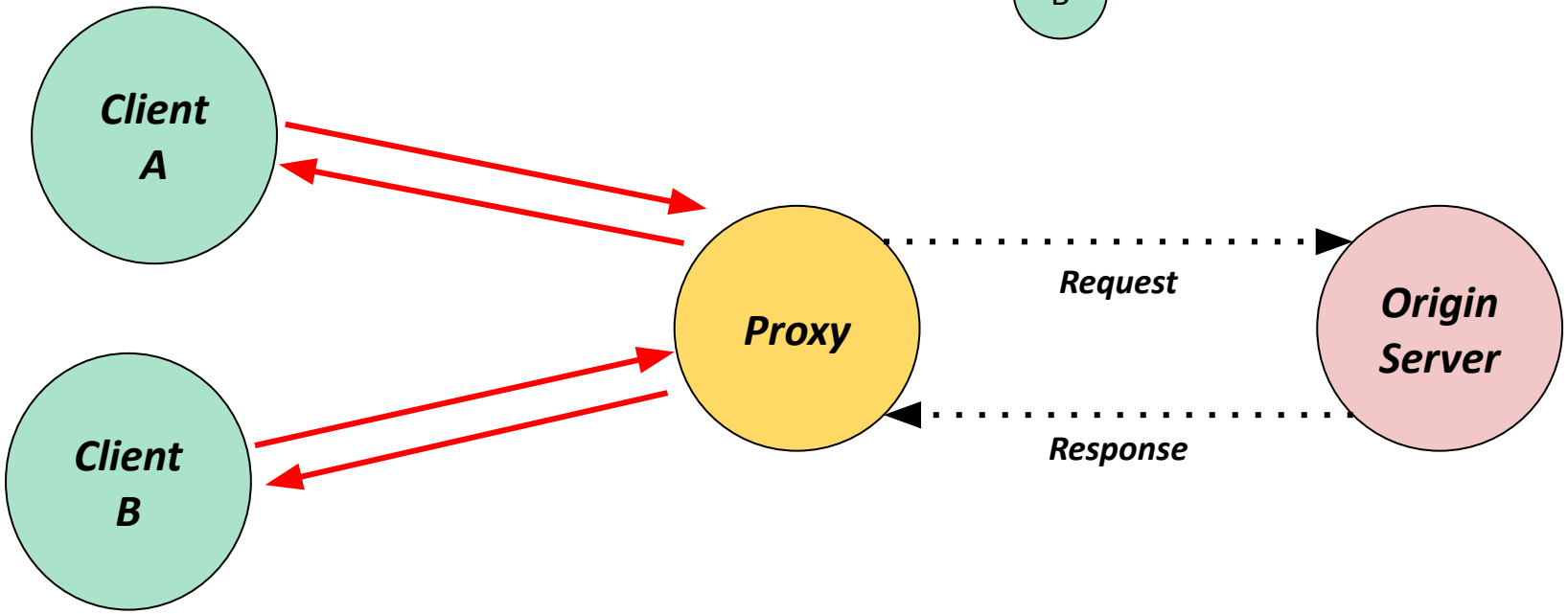
- Proxy sits between client and the server the client wants to talk to.



- Can do useful things in-between (not assessed in **`proxylab`**):
  - Caching, logging, anonymization, transcoding, etc.

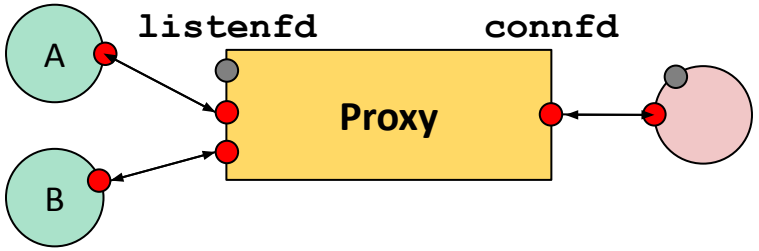
Proxies are *Servers*

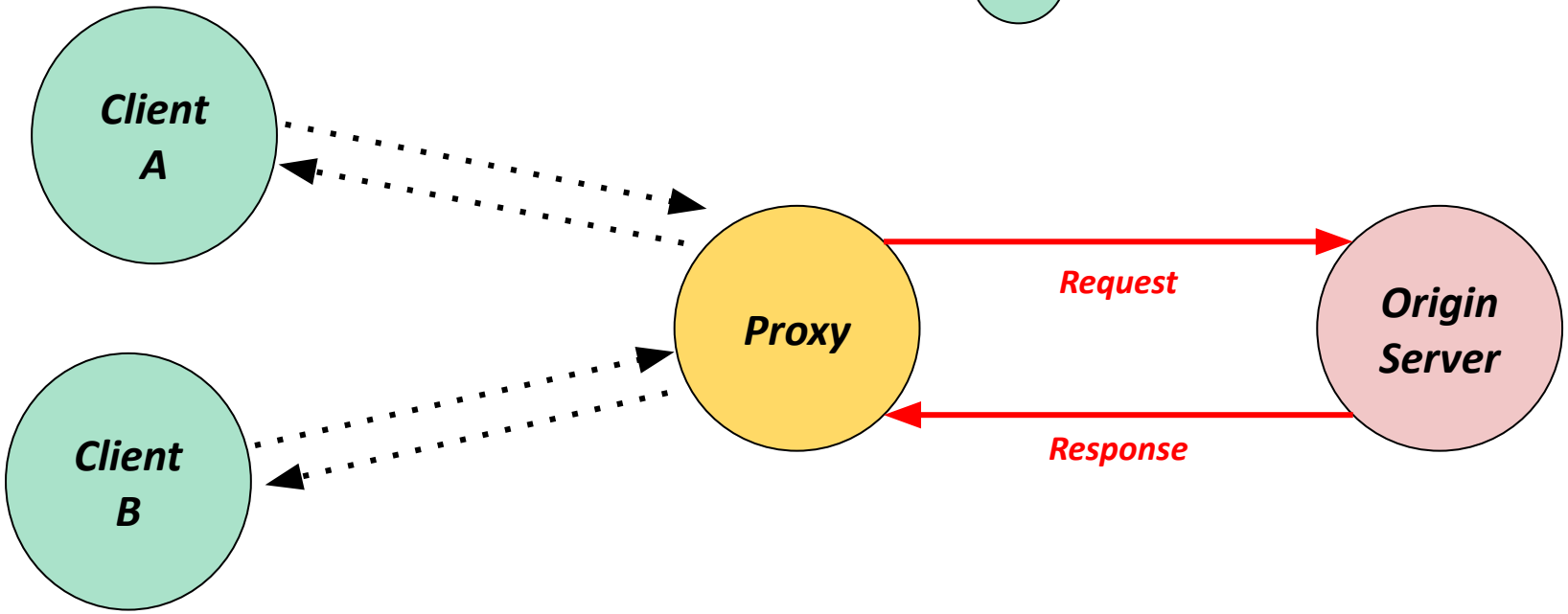




- Proxies need to listen for and handle requests from clients.
- Ideally, they should be able to do so for multiple clients at the same time!

Proxies are *Clients*





- Proxy parses headers in client's request to figure out which server to contact.
- Then connects to a server to get the data the client asked for.

Proxy Lab: Overview

- You'll implement a web proxy like the one on the previous slide!

Part I

- Accept connections from clients.
- Parse headers to determine origin server (see **`http_parser.h`**)
- Fetch data from the server, and forward response back to the client.

Part II

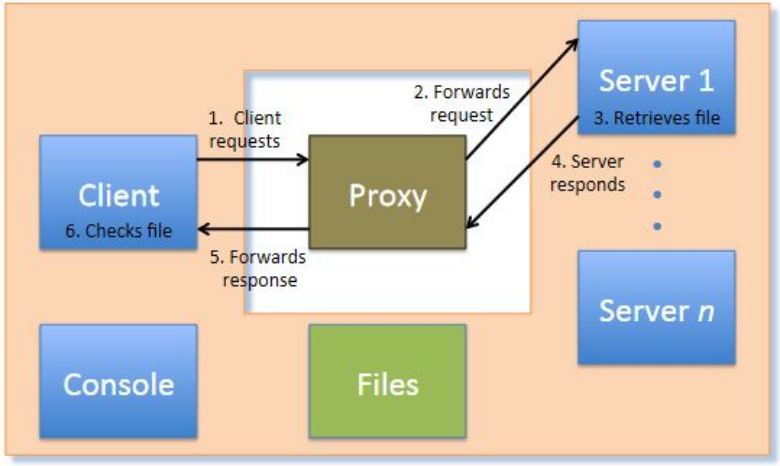
- Handle *concurrent requests* with POSIX threads (Tuesday's lecture)!

Proxy Lab: Getting Started

- Worth 4% of course grade.
- You'll have one week to complete the lab. *Start early!*
  - Grace days run into Thanksgiving
- Resources:
  - *Network Programming* Lectures
  - Textbook: Chapter 11
  - Write-up!
- Make sure you're familiar with the provided libraries before you start:
  - **`csapp.h`** – Networking Wrappers, **`rio`**
  - **`http_parser.h`** – For parsing *requests*

Debugging Proxy Lab: PxyDrive !

- Testing framework for Proxy Lab
  - Autolab uses it to grade your code…
  - You can use it to debug!
- PxyDrive workflow:
  - Generate text and binary data
  - Create server(s)
  - Build *transactions*
  - Trace transactions to inspect headers and response data.



PxyDrive Demo

- Let's run through some of the features of PxyDrive!
- If you want to follow along:

```
$ wget http://www.cs.cmu.edu/~213/activities/rec11.tar
$ tar -xvf rec11.tar
$ cd pxydrive-tutorial
```

PxyDrive – Getting Started

- Run the REPL and try entering some commands!

```
$ ./pxy/pxydrive.py
> help
  #         XXXXXXXX                Rest of line treated as comment
  check     ID [CODE]               Make sure request ID handled
properly and generated expected CODE
  delay     MS                      Delay for MS milliseconds
```

- Running with a specific proxy:

```
$ ./pxy/pxydrive.py -p ./proxy-ref
Proxy set up at nurseshark.ics.cs.cmu.edu:12168
>
```

PxyDrive – Tutorial 1

- Take a look at **`s01-basic-fetch.cmd`**
- Then try running the commands yourself in the REPL:

```
$ ./pxy/pxydrive.py -p ./proxy-ref
> generate data1.txt 1k
...
```

- **`generate data1.txt 1k`** – Generates a 1K text file called **`data1.txt`**
- **`serve s1`** – Launches a server called **`s1`**
- **`fetch f1 data1.txt s1`** – Fetches **`data1.txt`** from server **`s1`**, in a transaction called **`f1`**
- **`trace f1`** – Traces the transaction **`f1`**
- **`check f1`** – Checks the transaction **`f1`**

PxyDrive – Tutorial 1

- Try running the trace again with the **`-f`** flag:

```
$ ./pxy/pxydrive.py -f s01-basic-fetch.cmd -p ./proxy-ref
```

- Can you identify:
  - **`GET`** command
  - Host header? Other headers?
  - Request from *client to proxy*
  - Request from *proxy to server*
  - Response by *server to proxy*
  - Response by *proxy to client*

PxyDrive – Tutorial 1

- Let's try a different trace

```
$ ./pxy/pxydrive.py -f s02-basic-request.cmd  -p ./proxy-ref
```

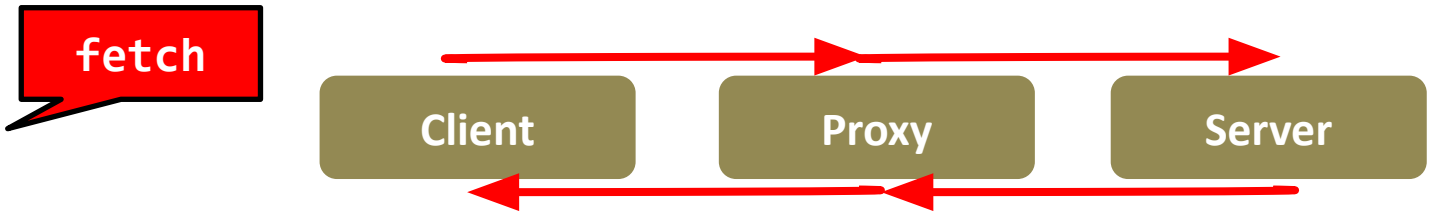
- You should get a different output.
- Why do we see "Response NOT sent by server" after the first **`trace`** command?

PxyDrive – Tutorial 1

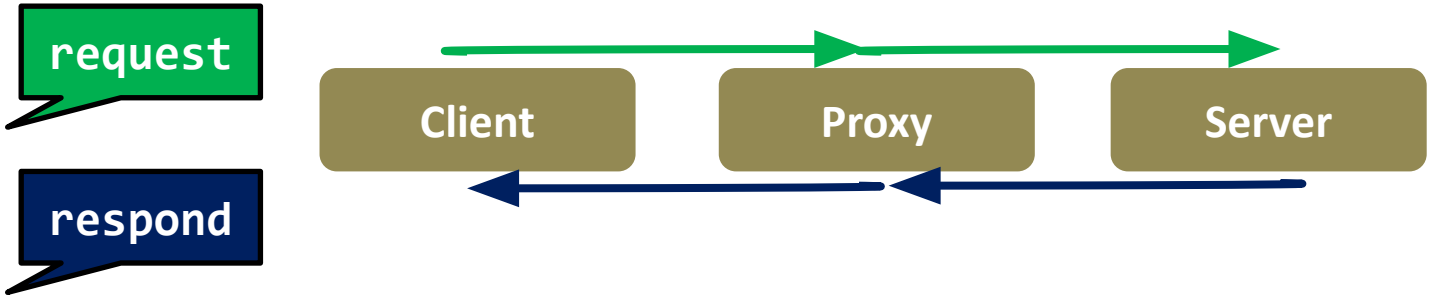
- **`generate data1.txt 1K`**
- **`serve s1`**
- **`request r1 data1.txt s1`** – Requests **`data1.txt`** from server **`s1`**, in a transaction called **`r1`**
- **`trace r1`**
- **`respond r1`**
- **`wait *`** – Allow server to respond to the transaction **`r1`**
- **`trace r1`**
- **`check r1`** – Checks the transaction **`r1`**

PxyDrive – Tutorial 1

- The **`fetch`** command makes the server *immediately* respond to a request.
- All steps of a transaction are complete after a fetch.



- The **`request`** command does not complete a transaction.
- A request needs a **`respond`** to complete its transaction.



PxyDrive – Tutorial 2

- Let's see what happens with a buggy proxy…

```
$ ./pxy/pxydrive.py -f s01-basic-fetch.cmd -p ./proxy-corrupt
```

- What happens?

```
# Make sure it was retrieved properly
> check f1
ERROR: Request f1 generated status 'error'.  Expecting 'ok'
(Mismatch between source file ./source_files/random/data1.txt and
response file ./response_files/f1-data1.txt starting at position
447: 'F' (hex 0x46) != 'G' (hex 0x47))
> quit
ERROR COUNT = 1
```

- Proxy clobbers response from server.

PxyDrive – Tutorial 3

- Let's try another buggy proxy…

```
$ ./pxy/pxydrive.py -f s01-basic-fetch.cmd -p ./proxy-strip -S 3
```

- **-S** denotes strictness level.
- What happens?

```
Response status: bad_request (Missing Request-ID header)
  Source file in ./source_files/random/data1.txt
Request status:  bad_request (Bad request)
  Result file in ./response_files/f1-status.html
```

- Proxy does not correctly forward **Request-ID** header from client to server.

PxyDrive – Tutorial 4

- Let's try another buggy proxy…

```
$ ./pxy/pxydrive.py -f s03-overrun.cmd -p ./proxy-overrun
```

- Is the error message helpful?

```
ERROR: Request f1 generated status 'error'.  Expecting 'ok'
(Socket closed after reading 106386/200000 bytes)
```

- Let's use **gdb**!

PxyDrive – Multi-Window Debugging

- Use **gdb** to run **proxy-overrun** in a fresh window.

```
$ gdb ./proxy-overrun
(gdb) run <port>
```

- Now run **pxydrive** in another window (same Shark):

```
$ ./pxy/pxydrive.py -P localhost:<port> -f s03-overrun.cmd
```

- When debugging **proxylab**, run **./port-for-user.pl** to get a unique port number, so your debugging doesn't conflict with other students.

PxyDrive – Multi-Window Debugging

- Multi-Window debugging is helpful even without **gdb**:

```
$ ./proxy-overrun <port>
```

```
$ ./pxy/pxydrive.py -P localhost:<port> -f s03-overrun.cmd
```

- Can redirect output of proxy to a file.
- If you include thread IDs in your print statements, can use **awk** to split thread outputs to different files for easier debugging.

Wrapping Up

- **`tshlab`** is due ***Tuesday (November 19th)***
- **`proxylab`** will be released on the same day
  - Due *November 26th*.
  - Start early!
- **`sfslab`** will be released before Thanksgiving
  - Due *December 5th*
- Apply to be a TA!
- Good luck on **`proxylab`** :-)

The End