CS305-Computer-Networks-Lab

> Name: 刘仁杰
>
> SID: 11911808

Design Idea

- Simulate a proxy server using python lib "socket" bind to specific (ip, port), this process monitor the request from the client server and transpond the request to the original server after nodal processing.
- This proxy server receives response from original requested server, it will auto transpond the response the original client.
- python lib **threading** are used to support concurrent request from multiple clients, moreover, python lib **Lock** is used to dealing with concurrency problem.
- **Json File** is used in this proxy server for flashing cache from memory to disk, every request in nodal processing stage will wait for the cache reply.
- **If-Modified-Since** request header is added to the original monitored client request to ensure client will receive the updated version.

Run The Proxy Server

> Use **sys.argv** to judge the **ip** and **port** parameters, this processing include string matching to distinguish between **ip** and **port**. If we do not pass any specific ip and port to this proxy script, their default value are 127.0.0.1 and 8080.

```
if __name__ == "__main__":
    ip = '127.0.0.1'
    port = 8080
    args = sys.argv[1:]
    for arg in args:
        if '.' in arg:
            ip = arg
        else:
            port = int(arg)
    try:
        start_proxy(ip, port)
    except KeyboardInterrupt:
        pass
```

```
PS E:\SUSTech_Codes\PyCharmWorkshop\ComputerNetworks\Lab_Assignment1> python .\web_proxy.py 172.8.1.92 2400
172.8.1.92 :  2400
PS E:\SUSTech_Codes\PyCharmWorkshop\ComputerNetworks\Lab_Assignment1> python .\web_proxy.py 2400
127.0.0.1 :  2400
PS E:\SUSTech_Codes\PyCharmWorkshop\ComputerNetworks\Lab_Assignment1> python .\web_proxy.py 172.8.1.92
172.8.1.92 :  8080
PS E:\SUSTech_Codes\PyCharmWorkshop\ComputerNetworks\Lab_Assignment1> python .\web_proxy.py
127.0.0.1 :  8080
PS E:\SUSTech_Codes\PyCharmWorkshop\ComputerNetworks\Lab_Assignment1> python .\web_proxy.py 2400 172.8.1.92
172.8.1.92 :  2400
PS E:\SUSTech_Codes\PyCharmWorkshop\ComputerNetworks\Lab_Assignment1>
```

Proxy GET, HEAD and POST Request

> First we create a socket `socket(AF_INET, SOCK_STREAM)` and use `socket.bind((ip, port))` to bind with user defined specific ip and port , then `socket.listen(10)` is triggered to wait for client requests.
>
> Then `socket.accept()` is triggered to get the connection with the client and the ip address and we initiate a proxy process instance to monitor this client's request.
>
> Every proxy process uses natively defined `receive_all()` to receive response both from the server in case of data loss.

- We here use `receive_all()` but not socket native function `socket.reveive()` is that when I try to receive response from **www.baidu.com**, I always get partial data rather than the whole with similar error message as "leaving xxx bytes to be sent" displayed in terminal. There are two reasons for this to happen:
  - pre-defined buffer size in `socket.receive()` is not large enough, leading to data loss.
  - since some response html body may be in large scale, server may send chunks rather than the whole data in one chunk, which will resulting in some latency in chunk transmission, so we will need to receive the response message regularly until the whole data transfer has completed.
- Source code `receive_all()` is shown below:

```
def receive_all(the_socket, time_limit=1):
    the_socket.setblocking(0)
    total_data = b''
    begin = time.time()
    while 1:
        if total_data and time.time() - begin > time_limit:
            break
        elif time.time() - begin > time_limit * 2:
            break
        try:
            data = the_socket.recv(2048)
            if data:
                total_data += data
                begin = time.time()
            else:
                time.sleep(0.1)
        except error:
            pass
    return total_data
```

Test results

- Use "curl -x 127.0.0.1:8080 http://www.example.com/"

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://www.example.com/
<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sans", "Helvetica Neue", Helvetica,
 Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
```

```
Ready to serve...
Received a connection from: ('127.0.0.1', 11933)
Ready to serve...
Read From Cache
b'HTTP/1.1 304 Not Modified\r\nAge: 491882\r\nCache-Control: max-age=604800\r\nDate: Tue, 12 Oct 2021 08:34:43 GMT\r\nEtag
```

- Use "curl -x 127.0.0.1:8080 http://www.baidu.com/ --head"

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://www.baidu.com/ --head
HTTP/1.1 200 OK
Accept-Ranges: bytes
Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform
Connection: keep-alive
Content-Length: 277
Content-Type: text/html
Date: Tue, 12 Oct 2021 08:35:30 GMT
Etag: "575e1f59-115"
Last-Modified: Mon, 13 Jun 2016 02:50:01 GMT
Pragma: no-cache
Server: bfe/1.0.8.18


C:\Users\21548>
```

```
Received a connection from: ('127.0.0.1', 1387)
Read From Cache
Ready to serve...
b'HTTP/1.1 200 OK\r\nAccept-Ranges: bytes\r\nCache-Control: private, no-cache,
```

- Use "curl -x 127.0.0.1:8080 http://ecosimulation.com/cgi-bin/testpost.cgi -X POST -d "firstname=san&lastname=zhang""

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://ecosimulation.com/cgi-bin/testpost.cgi -X POST -d "firstname=san&lastname=z
hang""
hello!
key:lastname, value:zhang
key:firstname, value:san
goodbye!

C:\Users\21548>
```

```
Received a connection from: ('127.0.0.1', 1390)
Ready to serve...
Read From Cache
```

Support Multithreading

> In this proxy server script, python lib **threading** is used for concurrency support

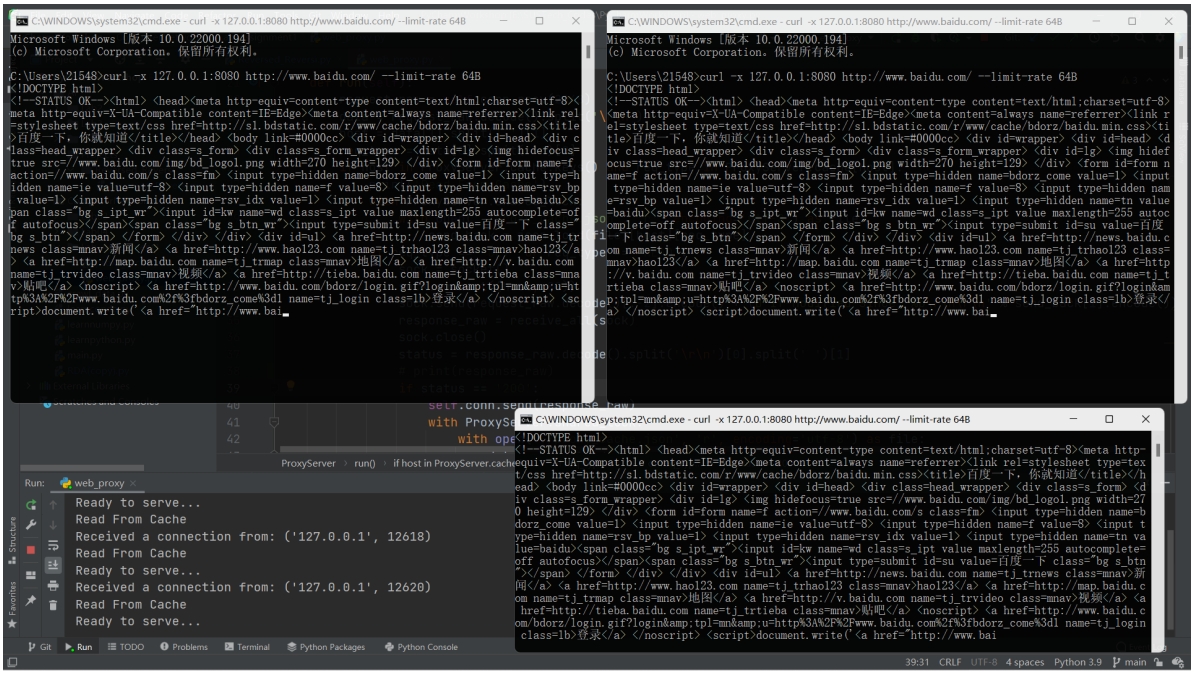
```
class ProxyServer(threading.Thread):
    cache = {}
    GMT_FORMAT = '%a, %d %b %Y %H:%M:%S GMT'
    shared_lock = threading.Lock()

    def __init__(self, conn, address):
        super().__init__()
        self.conn = conn
        self.address = address

    def run(self):
```

Test Results

- Use "curl -x 127.0.0.1:8080 http://www.baidu.com/ --limit-rate 64B"
- **Also, the picture below verify our implementation of concurrency control, multiple client are reading from the same cache file and it's OK**



Caching

> `json file` is used to store a dictionary, whose key is **host name** and the value is **[response, type(GET/HEAD/POST)]** and `json.load()` and `json.dump()` is to load and flash the cache from and into disk.
>
> `datetime.utcnow()` is adapted to get the time when the response was received and use `strftime()` to transfer it into GMT format.
>
> `threading.Lock()` is added to prevent concurrency reader-writer problems.

Test Results

- We first clear the cache file and use "curl -x 127.0.0.1:8080 http://www.example.com/"

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://www.example.com/
<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open Sa
Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
```

```
Ready to serve...
Received a connection from: ('127.0.0.1', 7873)
Ready to serve...
```

- we can see the cache file as shown which has stored the record we just request

```
{
  "www.example.com": {
    "GET": [
      "HTTP/1.1 200 OK\r\nAccept-Ranges: bytes\r\nAge: 3478
      "Tue, 12 Oct 2021 09:10:42 GMT"
    ]
  }
}
```

- This time we again request the same host with GET method, the request is read from cache

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://www.example.com/
<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI",
 Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
```

```
Ready to serve...
Received a connection from: ('127.0.0.1', 7873)
Ready to serve...
Received a connection from: ('127.0.0.1', 8984)
Read From Cache
Ready to serve...
```

- Before the third request, we modify the source code to display the response message in terminal, and this time we can see that, the server response with `HTTP/1.1 304 Not Modified` with no html body attached.

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://www.example.com/
<!doctype html>
<html>
<head>
    <title>Example Domain</title>

    <meta charset="utf-8" />
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "Segoe UI", "Open
 Arial, sans-serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
```

```
Ready to serve...
Received a connection from: ('127.0.0.1', 11979)
Read From Cache
Ready to serve...
b'HTTP/1.1 304 Not Modified\r\nAge: 271951\r\nCache-Control: max-age=6
```

- Also, having cached "**www.w3.org --HEAD**", we now request www.w3.org again:

```
C:\Users\21548>curl -x 127.0.0.1:8080 http://www.w3.org/ --head
HTTP/1.1 200 OK
date: Tue, 12 Oct 2021 09:20:49 GMT
content-location: Home.html
vary: negotiate,accept,Accept-Encoding,upgrade-insecure-requests
tcn: choice
last-modified: Tue, 12 Oct 2021 08:40:09 GMT
etag: "6e7c-5ce23cb078c40;89-3f26bd17a2f00"
accept-ranges: bytes
content-length: 28284
cache-control: max-age=600
expires: Tue, 12 Oct 2021 09:30:49 GMT
content-type: text/html; charset=utf-8
x-backend: www-mirrors

C:\Users\21548>
```

```
Received a connection from: ('127.0.0.1', 10193)
Ready to serve...
Read From Cache
b'HTTP/1.1 304 Not Modified\r\ndate: Tue, 12 Oct 2021 09:28:25 GMT\r\netag: "6
```