به نام خدا

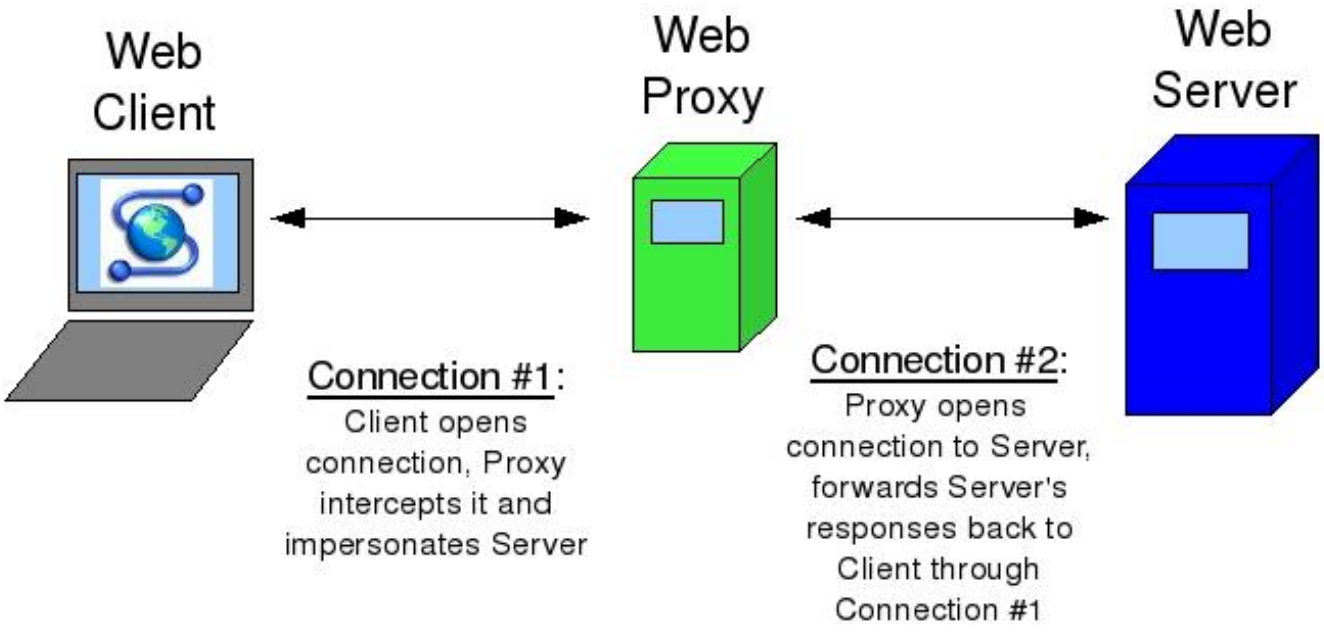
گزارش پروژه ۲

webproxy

امیرحسین محمد امری

۹۵۲۷۱۱۳

در این پروژه قصد داریم webproxy را شبیه سازی کنیم. شکل زیر عملکرد وب پراکسی را نشان می‌دهد.



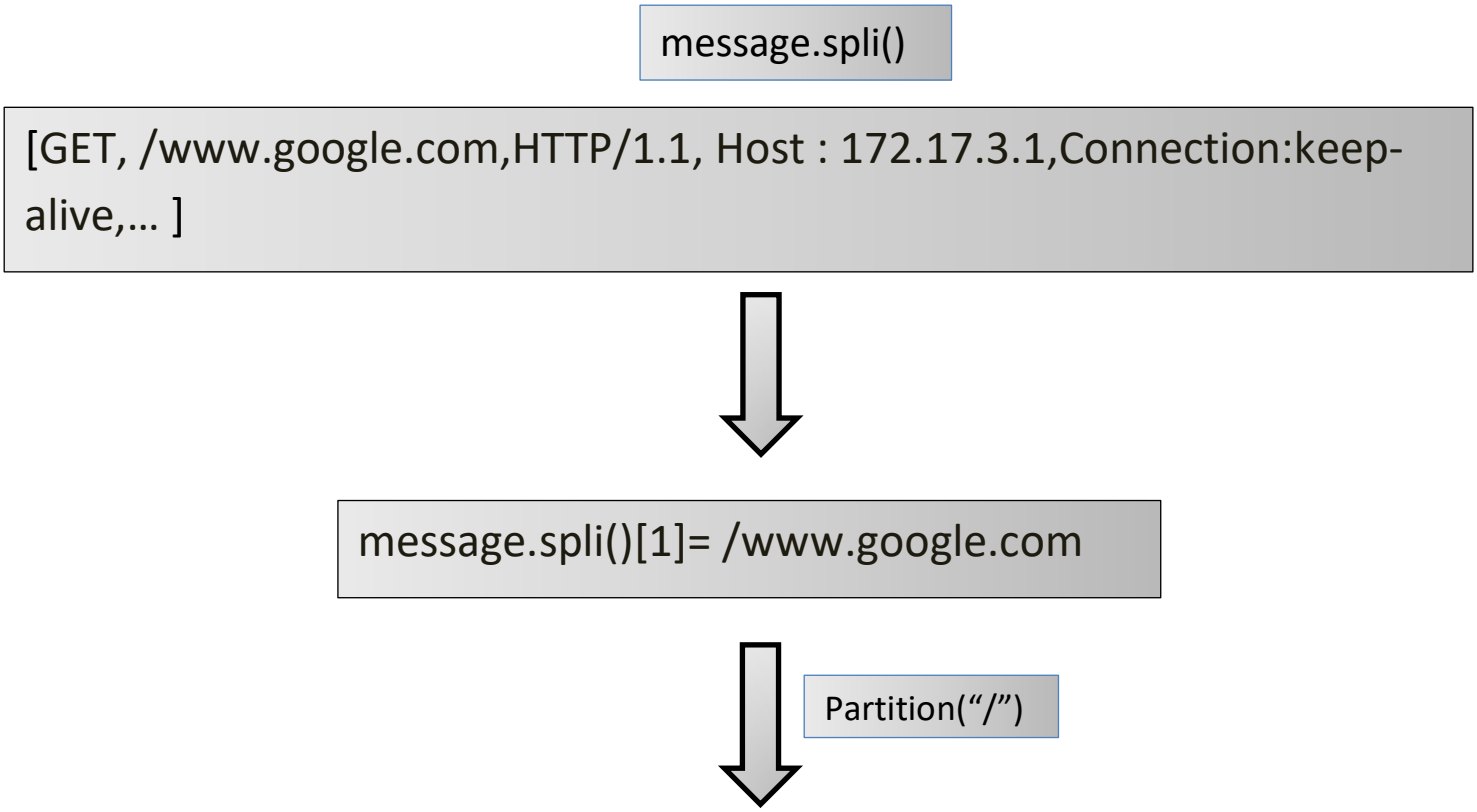
این پروژه را به دو صورت می‌توان اجرا کرد، یکی این که در قسمت کلاینت برای ارسال در خواست از مرورگر اینترنت استفاده کرد یا این که یک کد برای ارسال و دریافت درخواست استفاده کرد.

ابتدا کد وب‌پراکسی را توضیح داده و سپس کد سمت کلایت و در نهایت کار با مرورگر را توضیح می-دهم.

در تصویر زیر کد وب‌پراکسی را می‌بینید

```
#!/usr/bin/python
import socket
address= ('10.10.0.1',8899)
web_server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
web_server_socket.bind(address)
web_server_socket.listen(5)
while True:
    print('server is ready...')
    connection_from_client, addr = web_server_socket.accept()
    print('Received a connection from: ', addr)
    message = connection_from_client.recv(4096).decode()
    print(message)
    filename = message.split()[1].partition("/")[2]
    print(filename)
    file_Exist = "false"
    try:
        f = open(filename,"r")
        data = f.readlines()
        file_Exist = "true"
        connection_from_client.send("HTTP/1.1 200 OK \r\n".encode())
        connection_from_client.send("Content-Type:text/html\r\n")
        for i in range(0, len(data)):
            connection_from_client.send(data[i].encode())
        print("Read from cache")
    except IOError:
        if file_Exist == "false":
            print('-----------------------------------------------')
            print("creating a socket on proxy")
            c = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            host_name = filename.replace("www.","",1)
            print("Host name : ",host_name)
            try:
                c.connect((host_name, 80))
                c.sendall(message.encode())
                buff = c.recv(4096)
                print(buff)
                connection_from_client.send(buff)
            except:
                print("Illegal request")
        else:
            print("file notFound")
    connection_from_client.close()
web_server_socket.close()
```

- در خط ۳ آدرس و پورت وب‌پراکسی تعریف شده است.
- در خط ۴ سوکت از نوع tcp ایجاد شده است.
- در خط ۵ به آدرس و پورت تعریف شده در خط ۳ بایند کردیم.
- در خط ۶ تعداد درخواست هایی که قرار است به صورت همزمان دریافت شود مشخص شده است.
- بدنه اصلی کد در یک حلقه while همواره درست اجرا می‌شود.
- در خط ۹ اتصال از سمت کلایت قبول می‌شود .
- در خط ۱۱ داده ارسال شده توسط کلاینت دریافت شده و از دی‌کد می‌شود.
- در خط ۱۳ اسم سایت یا فایلی که کلاینت می‌خواهد به آن متصل شود یا آن را دریافت کند از پیام دریافتی استخراج می‌شود.
- در شکل زیر پیام دریافتی و روش استخراج پیام مشخص شده است.



message.spli()[1].partition=(‘’,’/’,’www.google.com’)

message.spli()[1].partition[2]=’www.google.com’

- حال که با تجزیه کردن پیام دریافتی نام فایل یا سایت خواسته شده را بدست آوردیم در حافظه‌ی خود پراکسی نگاه کرده تا ببینم محتویات این فایل یا سایت خواسته شده در حافظه‌ی پراکسی هست یا خیر و در صورت وجود آن را ارسال می-کنیم و در صورتی که وجود نداشته باشد با ایجاد یک اتصال به پورت ۸۰ و آدرس بدست آمده از تجزیه‌ی پیام دریافت شده جواب را از سرور اصلی دریافت و آن را به کلاینت می‌فرستیم، که این کار توسط خطوط ۱۶ تا ۴۱ صورت گرفته است.(توجه شود که در حقیقت webproxy چنین کاری را نمی‌کند و فقط درخواست خواسته شده را با ip خود به سرور اصلی میفرستد و جواب را برای کلاینت ارسال می‌کند.)
- فرمت try : except: به این صورت است که ابتدا دستورهای موجود در بلوک try انجام شده ولی اگر هنگام اجرای این دستورها به هر نوع خطایی برخورد کنیم کد های موجود در بلوک except را اجرا می‌کند. به صورت مثال در خط ۱۷ خواسته شده تا فایلی به نام بدست امده از پیام کلاینت باز شود اگر چنین فایلی در محل وب پراکسی موجود نباشد با خطا مواجه می‌شویم و باعث می‌شود به بلوک except برویم و یک سوکت tcp به سرور اصلی ایجاد کنیم و فایل خواسته شده را از سرور اصلی درخواست کنیم.
- در خط ۱۸ در صورت وجود ، فایل درخواستی بازشده و هر خط آن را در یک لیست قرار می‌گیرد.و در توط یک حلقه‌ی for در خطوط ۲۲ و ۲۳ هر عضو این لیست به کلاینت ارسال می‌شود.
- وقتی وارد بلوک except در خط ۲۵ می‌شویم که فایل در خود پراکسی وجود نداشته باید در این صورت همان‌طور که گفته شده است یک اتصال tcp به پورت و سرور اصلی برقرار می‌شود. در خط ۲۹ سوکت ایجاد شده و در خط ۳۳ به آدرس مد نظر متصل شده است.
- در خط ۳۴ تمام پیام دریافتی از سمت کلاینت به سرور فرستاده می‌شود.
- و جواب آن در خط ۳۵ در متغییری به نام buffer ذخیره می‌شود.و در خط ۳۶ به کلاینت فرستاده می‌شود.
- در انتها هر دو سوکت tcp بسته می‌شوند.

در تصویر زیر کد سمت کلاینت را مشاهده می‌کنید.

```
1  from socket import *
2  import webbrowser
3  import sys
4  target_host = '10.10.0.1'
5  web=sys.argv[1]
6  print(web)
7  file_path = "/"+web
8  target_port = 8899  # create a socket object
9  try:
10     client = socket(AF_INET, SOCK_STREAM)
11
12     # connect the client
13     client.connect((target_host, target_port))
14 except:
15     print("can't connect to the server")
16     sys.exit()
17
18 # send some data
19 request = "GET %s HTTP/1.1\r\nHost:%s\r\n\r\n" % (file_path, target_host)
20 client.send(request.encode())
21 with open("file.html", 'wb') as f:
22     print("recieving data...")
23     while True:
24         data = client.recv(1024)
25         if not data:
26             f.close()
27             break
28         print(data)
29         temp= data.partition("Content-Type:text/html\r\n")
30         f.write(temp[2])
31 deliver_state = client.recv(1024).decode()
32 print("deliver_state = " + deliver_state[0: deliver_state.find("\r")])
33 client.close()
34 webbrowser.open('file.html')
35 print("data connection disconnected")
```

- در این کد آدرس سایت به عنوان آرگمان دریافت می‌شود و خط ۱۹ در یک درخواست http قرار می‌گیرد و در خط ۲۰ به وب پراکسی فرستاده می‌شود و در خطوط ۲۱ تا ۳۰ جواب این درخواست در فایلی به نام file.html ذخیره می‌شود و در نهایت توسط خط ۳۴ این فایل باز می‌شود و در خط ۳۳ اتصال tcp ایجاد شده بسته می‌شود.