

INTERNET PROTOCOL LAB ASSIGNMENT -3

Name: Siriparapu Sparshika

Roll No: CYS22006

Date: 20-10-2022

Analyzing HTTP requests and responses using Wireshark

Aim:

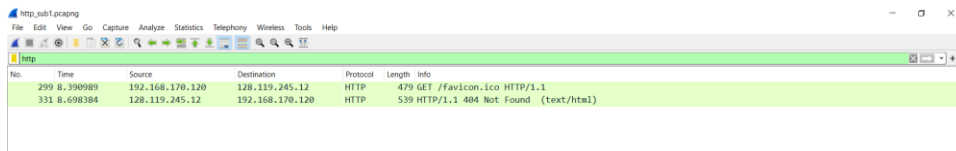
To explore the web application protocols using protocol analyzer

1.)

By looking at the information in the HTTP GET and response messages, answer the following questions.

1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server running?

A. HTTP version is 1.1



No.	Time	Source	Destination	Protocol	Length	Info
299	8.390989	192.168.170.120	128.119.245.12	HTTP	479	GET /favicon.ico HTTP/1.1
331	8.698384	128.119.245.12	192.168.170.120	HTTP	539	HTTP/1.1 404 Not Found (text/html)

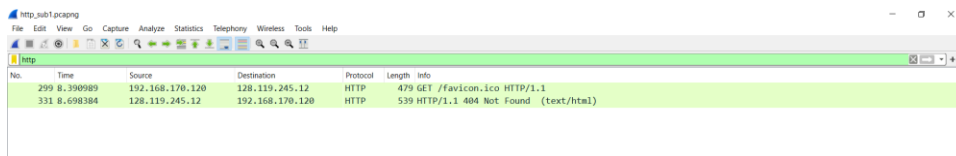
2. What languages (if any) do your browser indicate that it can accept to the server?

```
Accept-Language: en-US,en;q=0.9\r\n\r\n[Full request URI: http://gsia.cs.umass.edu/favicon.ico]\r\n[HTTP request: 1/1]\r\n[Response in frame: 331]
```

3. What is the IP address of your computer? Of the gaia.cs.umass.edu server?

Source:192.168.17.120

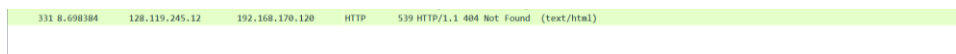
Destination:128.119.245.12



No.	Time	Source	Destination	Protocol	Length	Info
299	8.390989	192.168.170.120	128.119.245.12	HTTP	479	GET /favicon.ico HTTP/1.1
331	8.698384	128.119.245.12	192.168.170.120	HTTP	539	HTTP/1.1 404 Not Found (text/html)

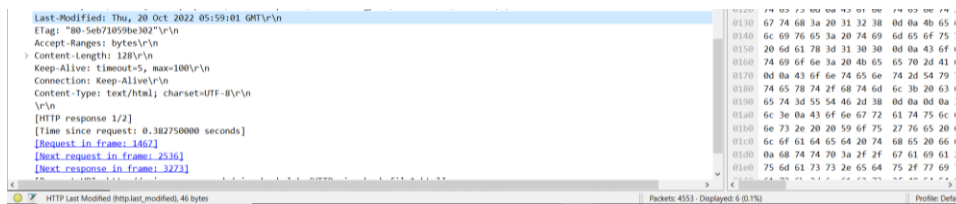
4. What is the status code returned from the server to your browser?

404 not found



No.	Time	Source	Destination	Protocol	Length	Info
331	8.698384	128.119.245.12	192.168.170.120	HTTP	539	HTTP/1.1 404 Not Found (text/html)

5. When was the HTML file that you are retrieving last modified at the server?



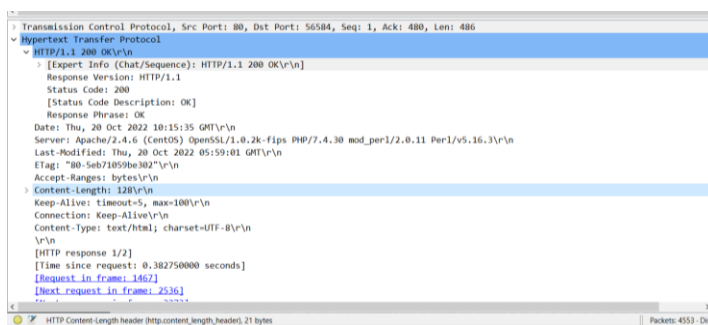
No.	Time	Source	Destination	Protocol	Length	Info
331	8.698384	128.119.245.12	192.168.170.120	HTTP	539	HTTP/1.1 404 Not Found (text/html)

HTTP response 1/2
[Time since request: 0.382750000 seconds]
[Request in frame: 1467]
[Next request in frame: 2536]
[Next response in frame: 3273]

6. How many bytes of content are being returned to your browser?

Last modified

: Identify malicious activities on n/w



No.	Time	Source	Destination	Protocol	Length	Info
331	8.698384	128.119.245.12	192.168.170.120	HTTP	539	HTTP/1.1 404 Not Found (text/html)

HTTP response 1/2
[Time since request: 0.382750000 seconds]
[Request in frame: 1467]
[Next request in frame: 2536]

7. By inspecting the raw data in the packet content window, do you see any headers within the data

that are not displayed in the packet-listing window? If so, name one.

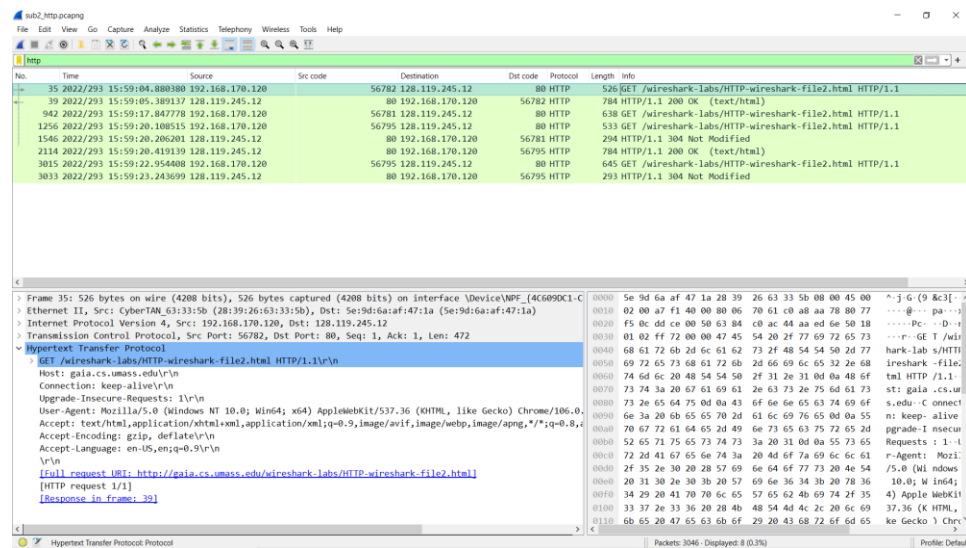
A. No

2.)

8. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an

“IF-MODIFIED-SINCE” line in the HTTP GET?

A. No



9. Inspect the contents of the server response. Did the server explicitly return the contents of the

file? How can you tell?

A. Yes

Destination address = source address should be same

```
<html>
<body>
  Congratulations again! Now you've downloaded the file lab2-2.html. <br>
  This file's last modification date will not change. <p>
  Thus, if you download this multiple times on your browser, a complete copy <br>
  will only be sent once by the server due to the inclusion of the IF-MODIFIED-SINCE<br>
  field in your browser's HTTP GET request to the server.
</body>
</html>
```

10. Now inspect the contents of the second HTTP GET request from your browser to the server. Do

you see an “IF-MODIFIED-SINCE:” line in the HTTP GET? What information follows the “IF-MODIFIED

SINCE:” header?

A. Yes

It contains: Thu, 20 Oct 2022 05:59:01 GMT\r\n

Wireshark packet capture showing an HTTP GET request and response. The packet list shows a GET request to /wireshark-labs/HTTP-wireshark-file2.html. The packet details show the request headers, including 'If-Modified-Since: Thu, 20 Oct 2022 05:59:01 GMT\r\n'. The packet bytes show the raw data of the request and response.

11. What is the HTTP status code and phrase returned from the server in response to this second

HTTP GET? Did the server explicitly return the file's contents? Explain.

For the first request the content sent request with html

For second it didn't modify so it doesn't return the content

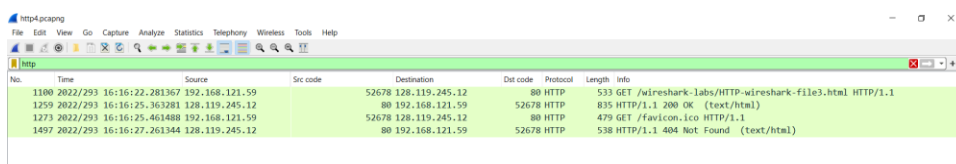
Info

```
GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
HTTP/1.1 200 OK (text/html)
GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
HTTP/1.1 304 Not Modified
HTTP/1.1 200 OK (text/html)
```

3.)

12. How many HTTP GET request messages did your browser send? Which packet number in the

trace contains the GET message for the Bill or Rights?



No.	Time	Source	Src code	Destination	Dist code	Protocol	Length	Info
1100	2022/293	16:16:22.281367 192.168.121.59		52678 128.119.245.12	80	HTTP	533	GET /wireshark-labs/HTTP-wireshark-file3.html HTTP/1.1
1259	2022/293	16:16:25.363281 128.119.245.12		80 192.168.121.59	52678	HTTP	835	HTTP/1.1 200 OK (text/html)
1273	2022/293	16:16:25.463480 192.168.121.59		52678 128.119.245.12	80	HTTP	479	GET /favicon.ico HTTP/1.1
1497	2022/293	16:16:27.263344 128.119.245.12		80 192.168.121.59	52678	HTTP	538	HTTP/1.1 404 Not Found (text/html)

-First packet number in the trace contains the GET message for the Bill or Rights

13. Which packet number in the trace contains the status code and phrase associated with the response to the HTTP GET request?

1259	2022/293	16:16:25.363281	128.119.245.12	80	192.168.121.59	52678	HTTP	835	HTTP/1.1 200 OK (text/html)
------	----------	-----------------	----------------	----	----------------	-------	------	-----	-----------------------------

14. What is the status code and phrase in the response?

Status code – 200, phrase - Ok

1259	2022/293	16:16:25.363281	128.119.245.12	80	192.168.121.59	52678	HTTP	835	HTTP/1.1 200 OK (text/html)
------	----------	-----------------	----------------	----	----------------	-------	------	-----	-----------------------------

15. How many data-containing TCP segments were needed to carry the single HTTP response and the text of the Bill of Rights?

A. 3

Filter btw http response we got should be servers ip address. Reassembled- updated data. We have only 3 tcp reassembled because it depends on the size of the data and there is no fix number of reassembled data.

1254	2022/293	16:16:25.360163	128.119.245.12	80	192.168.121.59	52678	TCP	54	80 → 52678 [ACK] Seq=1 Ack=480 Win=30336 Len=0	
+	1255	2022/293	16:16:25.360249	128.119.245.12	80	192.168.121.59	52678	TCP	1414	80 → 52678 [ACK] Seq=1 Ack=480 Win=30336 Len=1360 [TCP segment]
+	1256	2022/293	16:16:25.360294	128.119.245.12	80	192.168.121.59	52678	TCP	1414	80 → 52678 [ACK] Seq=1361 Ack=480 Win=30336 Len=1360 [TCP segment]
+	1257	2022/293	16:16:25.360463	192.168.121.59	52678	128.119.245.12	80	TCP	54	52678 → 80 [ACK] Seq=480 Ack=2721 Win=66560 Len=0
+	1258	2022/293	16:16:25.362219	128.119.245.12	80	192.168.121.59	52678	TCP	1414	80 → 52678 [ACK] Seq=2721 Ack=480 Win=30336 Len=1360 [TCP segment]
+	1259	2022/293	16:16:25.363281	128.119.245.12	80	192.168.121.59	52678	HTTP	835	HTTP/1.1 200 OK (text/html)

16. What is the server's response (status code and phrase) in response to the initial HTTP GET

message from your browser?

Status code 401, phrase - Unauthorized

The screenshot shows the Wireshark interface with a packet capture of an HTTP 401 Unauthorized response. The packet list on the left shows three packets. The selected packet (packet 3) is an HTTP 401 Unauthorized response from 192.168.170.120 to 192.168.170.126. The packet details pane on the right shows the structure of the HTTP response, including the status bar (401 Unauthorized) and the response body (text/html).

No.	Time	Source	Destination	Protocol	Length	Info
222	293.16	192.168.170.120	192.168.170.126	HTTP	542	GET /wireshark-labs/protected_pages/HTTP-wireshark-file5.html HTTP/1.1
223	293.16	192.168.170.120	192.168.170.126	HTTP	771	HTTP/1.1 401 Unauthorized (text/html)

17. When your browser sends the HTTP GET message for the second time, what new field is included

in the HTTP GET message?

For first get request we wont have any authorization header

```
GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1
Host: gaia.cs.umass.edu
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36 Edg/106.0.1370.47
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

But whereas for second get we have Authorization

```
GET /wiresark-labs/protected_pages/HTTP-wiresark-file5.html HTTP/1.1
Host: gais.cs.unsw.edu
Connection: keep-alive
Cache-Control: max-age=0
Authorization: Basic d2lyZ00wYXJrLXN0dWR1bWZ0eS1ldHdvcms=
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Result:

Thus, explored the web application protocols using protocol analyzer successfully.

Notes:

To analyze we use tcp not http

Ip.addr==128.119.245.12 && tcp

The image shows a Wireshark packet capture window with the filter 'Ip.addr==128.119.245.12 && tcp'. The packet list shows 19 packets. Packets 642-648 are TCP SYN and ACK exchanges. Packets 649-650 are TCP ACKs. Packets 651-652 are HTTP GET requests for /favicon.ico and /wirehark-labs/HTTP-wirehark-f1e3.html. Packets 653-654 are HTTP 200 OK responses. Packets 655-656 are TCP ACKs. Packets 657-658 are TCP RST and ACK exchanges. Packets 659-660 are TCP ACKs. Packets 661-662 are TCP RST and ACK exchanges. Packets 663-664 are TCP ACKs. Packets 665-666 are TCP RST and ACK exchanges. Packets 667-668 are TCP ACKs. Packets 669-670 are TCP RST and ACK exchanges. Packets 671-672 are TCP ACKs. Packets 673-674 are TCP RST and ACK exchanges. Packets 675-676 are TCP ACKs. Packets 677-678 are TCP RST and ACK exchanges. Packets 679-680 are TCP ACKs. Packets 681-682 are TCP RST and ACK exchanges. Packets 683-684 are TCP ACKs. Packets 685-686 are TCP RST and ACK exchanges. Packets 687-688 are TCP ACKs. Packets 689-690 are TCP RST and ACK exchanges. Packets 691-692 are TCP ACKs. Packets 693-694 are TCP RST and ACK exchanges. Packets 695-696 are TCP ACKs. Packets 697-698 are TCP RST and ACK exchanges. Packets 699-700 are TCP ACKs. Packets 701-702 are TCP RST and ACK exchanges. Packets 703-704 are TCP ACKs. Packets 705-706 are TCP RST and ACK exchanges. Packets 707-708 are TCP ACKs. Packets 709-710 are TCP RST and ACK exchanges. Packets 711-712 are TCP ACKs. Packets 713-714 are TCP RST and ACK exchanges. Packets 715-716 are TCP ACKs. Packets 717-718 are TCP RST and ACK exchanges. Packets 719-720 are TCP ACKs. Packets 721-722 are TCP RST and ACK exchanges. Packets 723-724 are TCP ACKs. Packets 725-726 are TCP RST and ACK exchanges. Packets 727-728 are TCP ACKs. Packets 729-730 are TCP RST and ACK exchanges. Packets 731-732 are TCP ACKs. Packets 733-734 are TCP RST and ACK exchanges. Packets 735-736 are TCP ACKs. Packets 737-738 are TCP RST and ACK exchanges. Packets 739-740 are TCP ACKs. Packets 741-742 are TCP RST and ACK exchanges. Packets 743-744 are TCP ACKs. Packets 745-746 are TCP RST and ACK exchanges. Packets 747-748 are TCP ACKs. Packets 749-750 are TCP RST and ACK exchanges. Packets 751-752 are TCP ACKs. Packets 753-754 are TCP RST and ACK exchanges. Packets 755-756 are TCP ACKs. Packets 757-758 are TCP RST and ACK exchanges. Packets 759-760 are TCP ACKs. Packets 761-762 are TCP RST and ACK exchanges. Packets 763-764 are TCP ACKs. Packets 765-766 are TCP RST and ACK exchanges. Packets 767-768 are TCP ACKs. Packets 769-770 are TCP RST and ACK exchanges. Packets 771-772 are TCP ACKs. Packets 773-774 are TCP RST and ACK exchanges. Packets 775-776 are TCP ACKs. Packets 777-778 are TCP RST and ACK exchanges. Packets 779-780 are TCP ACKs. Packets 781-782 are TCP RST and ACK exchanges. Packets 783-784 are TCP ACKs. Packets 785-786 are TCP RST and ACK exchanges. Packets 787-788 are TCP ACKs. Packets 789-790 are TCP RST and ACK exchanges. Packets 791-792 are TCP ACKs. Packets 793-794 are TCP RST and ACK exchanges. Packets 795-796 are TCP ACKs. Packets 797-798 are TCP RST and ACK exchanges. Packets 799-800 are TCP ACKs. Packets 801-802 are TCP RST and ACK exchanges. Packets 803-804 are TCP ACKs. Packets 805-806 are TCP RST and ACK exchanges. Packets 807-808 are TCP ACKs. Packets 809-810 are TCP RST and ACK exchanges. Packets 811-812 are TCP ACKs. Packets 813-814 are TCP RST and ACK exchanges. Packets 815-816 are TCP ACKs. Packets 817-818 are TCP RST and ACK exchanges. Packets 819-820 are TCP ACKs. Packets 821-822 are TCP RST and ACK exchanges. Packets 823-824 are TCP ACKs. Packets 825-826 are TCP RST and ACK exchanges. Packets 827-828 are TCP ACKs. Packets 829-830 are TCP RST and ACK exchanges. Packets 831-832 are TCP ACKs. Packets 833-834 are TCP RST and ACK exchanges. Packets 835-836 are TCP ACKs. Packets 837-838 are TCP RST and ACK exchanges. Packets 839-840 are TCP ACKs. Packets 841-842 are TCP RST and ACK exchanges. Packets 843-844 are TCP ACKs. Packets 845-846 are TCP RST and ACK exchanges. Packets 847-848 are TCP ACKs. Packets 849-850 are TCP RST and ACK exchanges. Packets 851-852 are TCP ACKs. Packets 853-854 are TCP RST and ACK exchanges. Packets 855-856 are TCP ACKs. Packets 857-858 are TCP RST and ACK exchanges. Packets 859-860 are TCP ACKs. Packets 861-862 are TCP RST and ACK exchanges. Packets 863-864 are TCP ACKs. Packets 865-866 are TCP RST and ACK exchanges. Packets 867-868 are TCP ACKs. Packets 869-870 are TCP RST and ACK exchanges. Packets 871-872 are TCP ACKs. Packets 873-874 are TCP RST and ACK exchanges. Packets 875-876 are TCP ACKs. Packets 877-878 are TCP RST and ACK exchanges. Packets 879-880 are TCP ACKs. Packets 881-882 are TCP RST and ACK exchanges. Packets 883-884 are TCP ACKs. Packets 885-886 are TCP RST and ACK exchanges. Packets 887-888 are TCP ACKs. Packets 889-890 are TCP RST and ACK exchanges. Packets 891-892 are TCP ACKs. Packets 893-894 are TCP RST and ACK exchanges. Packets 895-896 are TCP ACKs. Packets 897-898 are TCP RST and ACK exchanges. Packets 899-900 are TCP ACKs. Packets 901-902 are TCP RST and ACK exchanges. Packets 903-904 are TCP ACKs. Packets 905-906 are TCP RST and ACK exchanges. Packets 907-908 are TCP ACKs. Packets 909-910 are TCP RST and ACK exchanges. Packets 911-912 are TCP ACKs. Packets 913-914 are TCP RST and ACK exchanges. Packets 915-916 are TCP ACKs. Packets 917-918 are TCP RST and ACK exchanges. Packets 919-920 are TCP ACKs. Packets 921-922 are TCP RST and ACK exchanges. Packets 923-924 are TCP ACKs. Packets 925-926 are TCP RST and ACK exchanges. Packets 927-928 are TCP ACKs. Packets 929-930 are TCP RST and ACK exchanges. Packets 931-932 are TCP ACKs. Packets 933-934 are TCP RST and ACK exchanges. Packets 935-936 are TCP ACKs. Packets 937-938 are TCP RST and ACK exchanges. Packets 939-940 are TCP ACKs. Packets 941-942 are TCP RST and ACK exchanges. Packets 943-944 are TCP ACKs. Packets 945-946 are TCP RST and ACK exchanges. Packets 947-948 are TCP ACKs. Packets 949-950 are TCP RST and ACK exchanges. Packets 951-952 are TCP ACKs. Packets 953-954 are TCP RST and ACK exchanges. Packets 955-956 are TCP ACKs. Packets 957-958 are TCP RST and ACK exchanges. Packets 959-960 are TCP ACKs. Packets 961-962 are TCP RST and ACK exchanges. Packets 963-964 are TCP ACKs. Packets 965-966 are TCP RST and ACK exchanges. Packets 967-968 are TCP ACKs. Packets 969-970 are TCP RST and ACK exchanges. Packets 971-972 are TCP ACKs. Packets 973-974 are TCP RST and ACK exchanges. Packets 975-976 are TCP ACKs. Packets 977-978 are TCP RST and ACK exchanges. Packets 979-980 are TCP ACKs. Packets 981-982 are TCP RST and ACK exchanges. Packets 983-984 are TCP ACKs. Packets 985-986 are TCP RST and ACK exchanges. Packets 987-988 are TCP ACKs. Packets 989-990 are TCP RST and ACK exchanges. Packets 991-992 are TCP ACKs. Packets 993-994 are TCP RST and ACK exchanges. Packets 995-996 are TCP ACKs. Packets 997-998 are TCP RST and ACK exchanges. Packets 999-1000 are TCP ACKs.

Http-tcp working for transferring protocol, tcp sharing on behalf of http,
so we get the same data
in some cases

tcp.stream eq 10

we get the same thing for follow for both tcp and http

Ex: website to YouTube

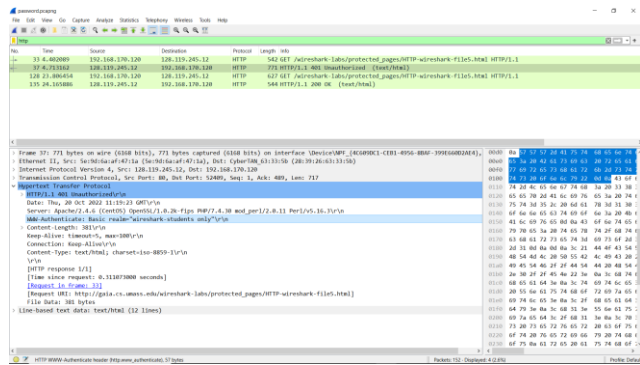
In the obj if we see

there's one page we see html page and another is some data

I.e., due to vulnerabilities.

- means path and it contains data to be explored

authorization



401,304,200

We are getting credentials, and password is encrypted using bay 64

We are using basic authorization tech