

1 Sending emails and SMTP

1. What filter did you use to catch the traffic and explain why you chose that filter?

Tcp.port == 465: we specified port 465 in the command

2. What is the standard port for SMTP and why do we use port 465 in the example above?

The standard port for SMTP is port 25. We used 465 so we can isolate our activity for observation

3. Explain each line used in the command line and what it does and why it is needed?

We connect to gmail using SMTP, then “Client hello” then we log into the account. After that we , enter the “from” email followed by the recipient. Then we enter the subject and the email before quitting.

4. How much back and forth communication do you see for establishing the connection?

Establishing a connection sent 8 packets back and forth.

5. What is the port your local machine is using between sending the two emails when communicating with the SMTP server?

Port 465

6. Explain who sends the first FIN flag and how the quitting process works.

The server sends the first FIN flag and the client responds with their own

7. Add a screenshot of your Wireshark output and add it to your document.

*enp0s31f6

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.port == 465

No.	Time	Source	Destination	Protocol	Length	Info
369	30.827467333	192.168.50.70	142.250.141.108	TCP	74	36342 → 465 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 ...
374	30.858039854	142.250.141.108	192.168.50.70	TCP	74	465 → 36342 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1430 S...
375	30.858083609	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=200955736...
376	30.858596933	192.168.50.70	142.250.141.108	TLsv1.3	372	Client Hello
381	30.887586548	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=1 Ack=307 Win=66816 Len=0 TSval=2368775...
382	30.887586807	142.250.141.108	192.168.50.70	TCP	66	[TCP Dup ACK 381#1] 465 → 36342 [ACK] Seq=1 Ack=307 Win=66816...
383	30.887869939	142.250.141.108	192.168.50.70	TLsv1.3	1484	Server Hello, Change Cipher Spec
384	30.887888606	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=307 Ack=1419 Win=64128 Len=0 TSval=2009...
385	30.888159309	142.250.141.108	192.168.50.70	TLsv1.3	1288	Application Data
386	30.888179454	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=307 Ack=2641 Win=63360 Len=0 TSval=2009...
387	30.891048053	192.168.50.70	142.250.141.108	TLsv1.3	146	Change Cipher Spec, Application Data
390	30.924844193	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=2641 Ack=387 Win=66816 Len=0 TSval=2368...
391	30.952173616	142.250.141.108	192.168.50.70	TLsv1.3	679	Application Data, Application Data
392	30.952194366	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=387 Ack=3254 Win=64128 Len=0 TSval=2009...
481	36.765327367	192.168.50.70	142.250.141.108	TLsv1.3	104	Application Data
486	36.798137690	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3254 Ack=425 Win=66816 Len=0 TSval=2368...
495	36.824276581	142.250.141.108	192.168.50.70	TLsv1.3	124	Application Data
496	36.824303255	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=425 Ack=3312 Win=64128 Len=0 TSval=2009...
568	41.940313706	192.168.50.70	142.250.141.108	TLsv1.3	100	Application Data
569	41.972349312	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3312 Ack=459 Win=66816 Len=0 TSval=2368...
570	41.996933800	142.250.141.108	192.168.50.70	TLsv1.3	106	Application Data
571	41.996954674	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=459 Ack=3352 Win=64128 Len=0 TSval=2009...
609	48.882268057	192.168.50.70	142.250.141.108	TLsv1.3	122	Application Data
610	48.916450141	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3352 Ack=515 Win=66816 Len=0 TSval=2368...
611	48.940119129	142.250.141.108	192.168.50.70	TLsv1.3	106	Application Data
612	48.940141341	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=515 Ack=3392 Win=64128 Len=0 TSval=2009...
672	55.434222666	192.168.50.70	142.250.141.108	TLsv1.3	106	Application Data
673	55.469151068	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3392 Ack=555 Win=66816 Len=0 TSval=2368...
674	55.635451229	142.250.141.108	192.168.50.70	TLsv1.3	108	Application Data
675	55.635474536	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=555 Ack=3434 Win=64128 Len=0 TSval=2009...
719	61.952159029	192.168.50.70	142.250.141.108	TLsv1.3	126	Application Data
720	61.980175337	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3434 Ack=615 Win=66816 Len=0 TSval=2368...
721	62.010438385	142.250.141.108	192.168.50.70	TLsv1.3	129	Application Data
722	62.010461261	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=615 Ack=3497 Win=64128 Len=0 TSval=2009...
769	67.572198673	192.168.50.70	142.250.141.108	TLsv1.3	124	Application Data
770	67.606397105	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3497 Ack=673 Win=66816 Len=0 TSval=2368...
776	67.629260940	142.250.141.108	192.168.50.70	TLsv1.3	129	Application Data
777	67.629282623	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=673 Ack=3560 Win=64128 Len=0 TSval=2009...
806	71.808974877	192.168.50.70	142.250.141.108	TLsv1.3	94	Application Data
807	71.841715689	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3560 Ack=701 Win=66816 Len=0 TSval=2368...
808	71.951227984	142.250.141.108	192.168.50.70	TLsv1.3	130	Application Data
809	71.951254375	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=701 Ack=3624 Win=64128 Len=0 TSval=2009...
1079	90.883957151	192.168.50.70	142.250.141.108	TLsv1.3	115	Application Data
1080	90.911776595	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3624 Ack=750 Win=66816 Len=0 TSval=2368...
1082	92.266869706	192.168.50.70	142.250.141.108	TLsv1.3	91	Application Data
1083	92.294145067	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3624 Ack=775 Win=66816 Len=0 TSval=2368...
1095	92.610709240	142.250.141.108	192.168.50.70	TLsv1.3	141	Application Data
1096	92.610738842	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=775 Ack=3699 Win=64128 Len=0 TSval=2009...
1179	94.078926822	192.168.50.70	142.250.141.108	TLsv1.3	94	Application Data
1181	94.106545582	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [ACK] Seq=3699 Ack=803 Win=66816 Len=0 TSval=2368...
1184	94.136100224	142.250.141.108	192.168.50.70	TLsv1.3	145	Application Data
1185	94.136120096	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [ACK] Seq=803 Ack=3778 Win=64128 Len=0 TSval=2009...
1186	94.137603929	142.250.141.108	192.168.50.70	TCP	66	465 → 36342 [FIN, ACK] Seq=3778 Ack=803 Win=66816 Len=0 TSval...
1187	94.137719983	192.168.50.70	142.250.141.108	TLsv1.3	90	Application Data
1188	94.137737387	192.168.50.70	142.250.141.108	TCP	66	36342 → 465 [FIN, ACK] Seq=827 Ack=3779 Win=64128 Len=0 TSval...
1200	94.170514600	142.250.141.108	192.168.50.70	TCP	60	465 → 36342 [RST] Seq=3778 Win=0 Len=0
1201	94.170514687	142.250.141.108	192.168.50.70	TCP	60	465 → 36342 [RST] Seq=3779 Win=0 Len=0
1202	94.170514705	142.250.141.108	192.168.50.70	TCP	60	465 → 36342 [RST] Seq=3779 Win=0 Len=0

....0. = Urgent: Not set
1 = Acknowledgment: Set
 0 = Push: Not set

0000 a8 5e 45 10 98 20 1c 1b 0d 6f bd cb 08 00 45 00 ^E... ..o...E
 0010 00 34 3f 19 40 00 00 06 ec 55 c0 a8 32 46 8e fa .4?@.@.U..2F..

wireshark_enp0s31f6_20210527104726_D8VOqP.pcapng Packets: 1316 · Displayed: 58 (4.4%) · Dropped: 0 (0.0%) Profile: Default

2 Understanding HTTP

1. Explain the specific API calls you used.

curl -H "Accept: application/vnd.github.cloak-preview+json"

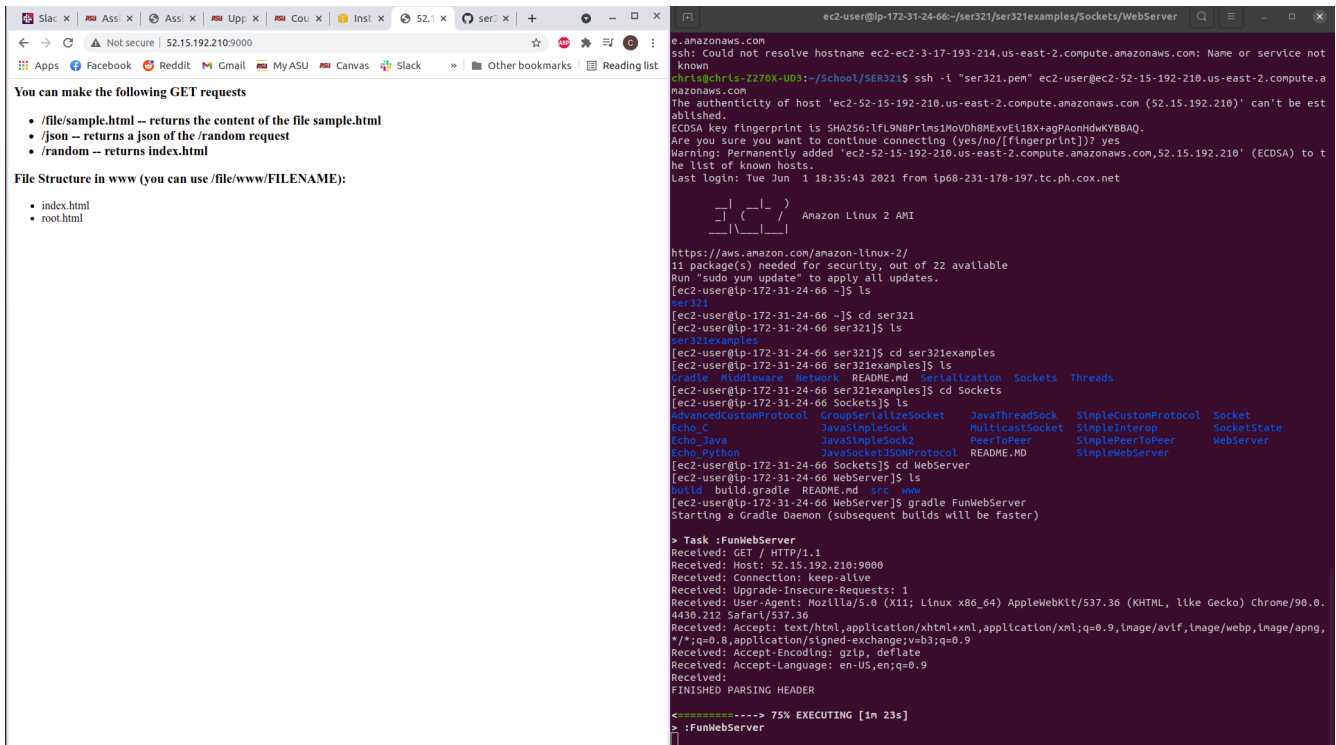
<https://api.github.com/search/commits?q=author:amehlhase316>

2. Explain the difference between stateless and a state-full communication.

In stateless communication, the client sends a request to the server and the server replies back depending on the state. In state-full communication the client sends a request to the server and if the server does not respond, the client continues to resend the request.

```
chris@chris-Z270X-U03:~$ curl -H "Accept: application/vnd.github.cloak-preview+json" https://api.github.com/search/commits?q=author:amehlhase316
{
  "total_count": 221,
  "incomplete_results": false,
  "items": [
    {
      "url": "https://api.github.com/repos/ARossiter0/Memoranda/commits/a0053b1b7dc04e967a3efb5c8e94c98e20b4ac6e",
      "sha": "a0053b1b7dc04e967a3efb5c8e94c98e20b4ac6e",
      "node_id": "MDY6Q29tbWl0MzMONzU3Mjk00MEWMDUzYjFlN2RjMDRlOTY3YTlZnI1YzhlOTRjOTlMjB1NGFjNmU=",
      "html_url": "https://github.com/ARossiter0/Memoranda/commit/a0053b1b7dc04e967a3efb5c8e94c98e20b4ac6e",
      "comments_url": "https://api.github.com/repos/ARossiter0/Memoranda/commits/a0053b1b7dc04e967a3efb5c8e94c98e20b4ac6e/comments",
      "commit": {
        "url": "https://api.github.com/repos/ARossiter0/Memoranda/git/commits/a0053b1b7dc04e967a3efb5c8e94c98e20b4ac6e",
        "author": {
          "date": "2020-10-15T18:06:29.000-07:00",
          "name": "amehlhase316",
          "email": "amehlhase316@asu.edu"
        },
        "committer": {
          "date": "2020-10-15T18:06:29.000-07:00",
          "name": "amehlhase316",
          "email": "amehlhase316@asu.edu"
        },
        "message": "Project dump",
        "tree": {
          "url": "https://api.github.com/repos/ARossiter0/Memoranda/git/trees/84c2d704ea4bf7e4cd3054efb36429bccccf298a6",
          "sha": "84c2d704ea4bf7e4cd3054efb36429bccccf298a6"
        },
        "comment_count": 0
      },
      "author": {
        "login": "amehlhase316",
        "id": 46384989,
        "node_id": "MDQ6VXNlcjQ2Mzg0OTg5",
        "avatar_url": "https://avatars.githubusercontent.com/u/46384989?v=4",
        "gravatar_id": "",
        "url": "https://api.github.com/users/amehlhase316",
        "html_url": "https://github.com/amehlhase316",
        "followers_url": "https://api.github.com/users/amehlhase316/followers",
        "following_url": "https://api.github.com/users/amehlhase316/following{/other_user}",
        "gists_url": "https://api.github.com/users/amehlhase316/gists{/gist_id}",
        "starred_url": "https://api.github.com/users/amehlhase316/starred{/owner}/{/repo}",
        "subscriptions_url": "https://api.github.com/users/amehlhase316/subscriptions",
        "organizations_url": "https://api.github.com/users/amehlhase316/orgs",
        "repos_url": "https://api.github.com/users/amehlhase316/repos",
        "events_url": "https://api.github.com/users/amehlhase316/events{/privacy}",
        "received_events_url": "https://api.github.com/users/amehlhase316/received_events",
        "type": "User",
        "site_admin": false
      },
      "committer": {
        "login": "amehlhase316",
        "id": 46384989,
        "node_id": "MDQ6VXNlcjQ2Mzg0OTg5",
        "avatar_url": "https://avatars.githubusercontent.com/u/46384989?v=4",
        "gravatar_id": "",
        "url": "https://api.github.com/users/amehlhase316"
      }
    }
  ]
}
```

3 Setup your second system and run Server on it



1. What filter did you use? Explain why you chose that filter.

ip.addr == 52.15.192.210 && http

2. What happens when you are on /random and click the refresh button compared to the browser refresh (you can also use the command line output that the WebServer generates to answer this)?

Hitting refresh in the browser produces 2 GET requests
Hitting random button produces 1 get request

3. What kinds of response codes are you able to get through different requests to your server?

200 OK
400 Bad requests
404 Not Found

4. Explain the response codes you get and why you get them?

200: When there were no issues the code was OK.
400: Entering wrong URL
404: Trying to access something that does not exists on the site

5. When you do a ipOfSecondMachine:9000 take a look what Wireshark generates as a server response. Are you able to find the data that the server sends back to you?

Yes, we can see everything in plain text

6. Based on the above question explain why HTTPs is now more common than HTTP.

HTTPs encrypts what is sent and received so it is much more difficult to see what is being sent.

7. What port does the server listen to for HTTP requests in our case and is that the most common port for HTTP?

22

8. What local port is used when sending different requests to the WebServer? How does it differ to the traffic to your SMTP server from part 1?

TCP: 37934

HTTP: 42266

3.4 Setting up a "real" Web server

1. Check your traffic to your Webserver now. What port is the traffic going to now? Is it the same as previously used or is it and should it be different?

The ports are the same as before.

2. Is it still HTTP or is it now HTTPs? Why?

It is still HTTP. I did not set up HTTPs

You can make the following GET requests

- `/file/sample.html` -- returns the content of the file `sample.html`
- `/json` -- returns a json of the `/random` request
- `/random` -- returns `index.html`

File Structure in `www` (you can use `/file/www/FILENAME`):

- `index.html`
- `root.html`

```
42 php7.4 available [=stable]
43 livepatch available [=stable]
44 python3.8 available [=stable]
45 haproxy2 available [=stable]
46 collectd available [=stable]
47 aws-nitro-enclaves-ctl available [=stable]
48 R4 available [=stable]
49 kernel-5.4 available [=stable]
50 selinux-ng available [=stable]
51 php8.0 available [=stable]
52 toncat9 available [=stable]
53 unbound1.13 available [=stable]
54 mariadb10.5 available [=stable]
55 kernel-5.10 available [=stable]

[ec2-user@ip-172-31-24-66 WebServer] sudo nginx
[ec2-user@ip-172-31-24-66 WebServer] sudo vim /etc/nginx/nginx.conf
[ec2-user@ip-172-31-24-66 WebServer] sudo nginx -s reload
[ec2-user@ip-172-31-24-66 WebServer] gradle funWebServer
Starting a Gradle Daemon, 1 busy and 1 stopped Daemons could not be reused, use --status for details

> Task :FunWebServer
Received: GET / HTTP/1.1
Received: Host: 18.116.34.196:9000
Received: Connection: keep-alive
Received: Upgrade-Insecure-Requests: 1
Received: User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/537.36
Received: 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
Received: Accept-Encoding: gzip, deflate
Received: Accept-Language: en-US,en;q=0.9
Received: FINISHED PARSING HEADER
Received: null
Received: FINISHED PARSING HEADER
Received: GET / HTTP/1.0
Received: Host: localhost:9000
Received: Connection: close
Received: Content-Length: 0
Received: User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36
Received: FINISHED PARSING HEADER
Received: GET /random HTTP/1.1
Received: Host: 18.116.34.196:9000
Received: Connection: keep-alive
Received: Upgrade-Insecure-Requests: 1
Received: User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/537.36
Received: 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
Received: Accept-Encoding: gzip, deflate
Received: Accept-Language: en-US,en;q=0.9
Received: FINISHED PARSING HEADER
```

Time	Source	Destination	Protocol	Length	Info
6127.150	159999166	192.168.50.70	HTTP	378	GET /json HTTP/1.1
6131.150	235122033	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (application/json)
6141.151	092450093	192.168.50.70	HTTP	378	GET /json HTTP/1.1
6145.151	166373659	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (application/json)
6221.151	843990119	192.168.50.70	HTTP	378	GET /json HTTP/1.1
6231.151	919774457	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (application/json)
6244.153	894956600	192.168.50.70	HTTP	378	GET /json HTTP/1.1
6252.153	908441935	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (application/json)
6313.158	940572820	192.168.50.70	HTTP	493	GET / HTTP/1.1
6317.159	019544168	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (text/html)
15113.900	140108472	192.168.50.70	HTTP	480	GET / HTTP/1.1
15217.995	626826021	192.168.50.70	HTTP	519	GET / HTTP/1.1
15227.995	703201302	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (text/html)
15301.996	016101701	192.168.50.70	HTTP	66	HTTP/1.1 200 OK (text/html)
15398.1075	9066464	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (text/html)
16000.1076	0743548	192.168.50.70	HTTP	391	GET /json HTTP/1.1
16015.1076	1500410	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (application/json)
16300.1101	5430889	192.168.50.70	HTTP	512	GET / HTTP/1.1
16312.1101	6234935	18.116.34.196	HTTP	66	HTTP/1.1 200 OK (text/html)

Frame 15985: 512 bytes on wire (4096 bits), 512 bytes captured (4096 bits) on interface enp0s31f6, id 0
Ethernet II, Src: 61ga-byt-6f:bd:cb (1c:1b:0d:6f:bd:cb), Dst: ASUSTekC 10:96:20 (a8:5e:45:10:96:20)
Internet Protocol Version 4, Src: 192.168.50.70, Dst: 18.116.34.196
Transmission Control Protocol, Src Port: 8080, Dst Port: 9000, Seq: 1, Ack: 1, Len: 440
Hypertext Transfer Protocol

3.6.1 Multiply

If there is one input, the result is multiplied by one. If there are two inputs, the result is the product of the two inputs. If there are more than two inputs, A message is shown to the client explaining that there was more than two inputs received.