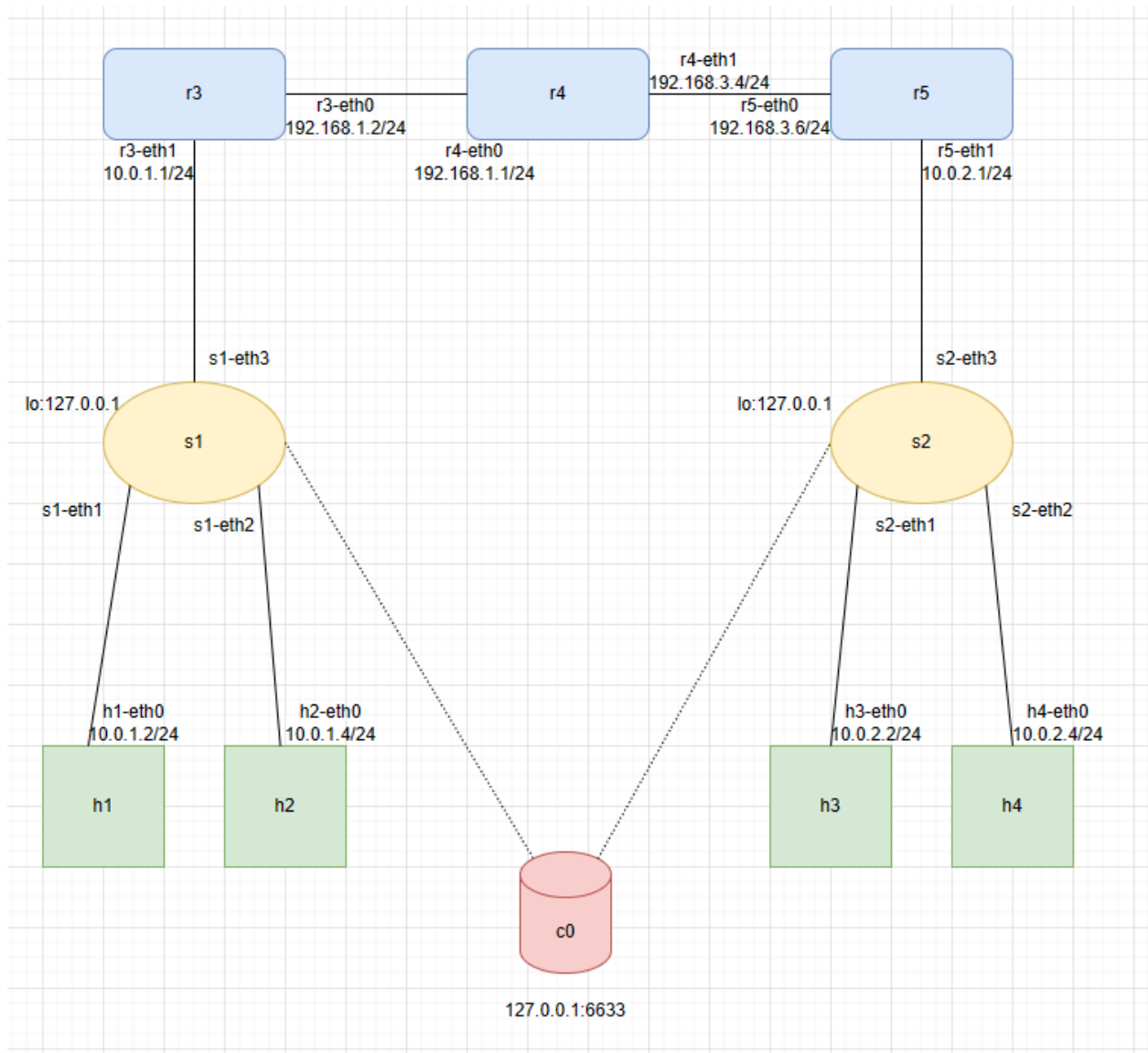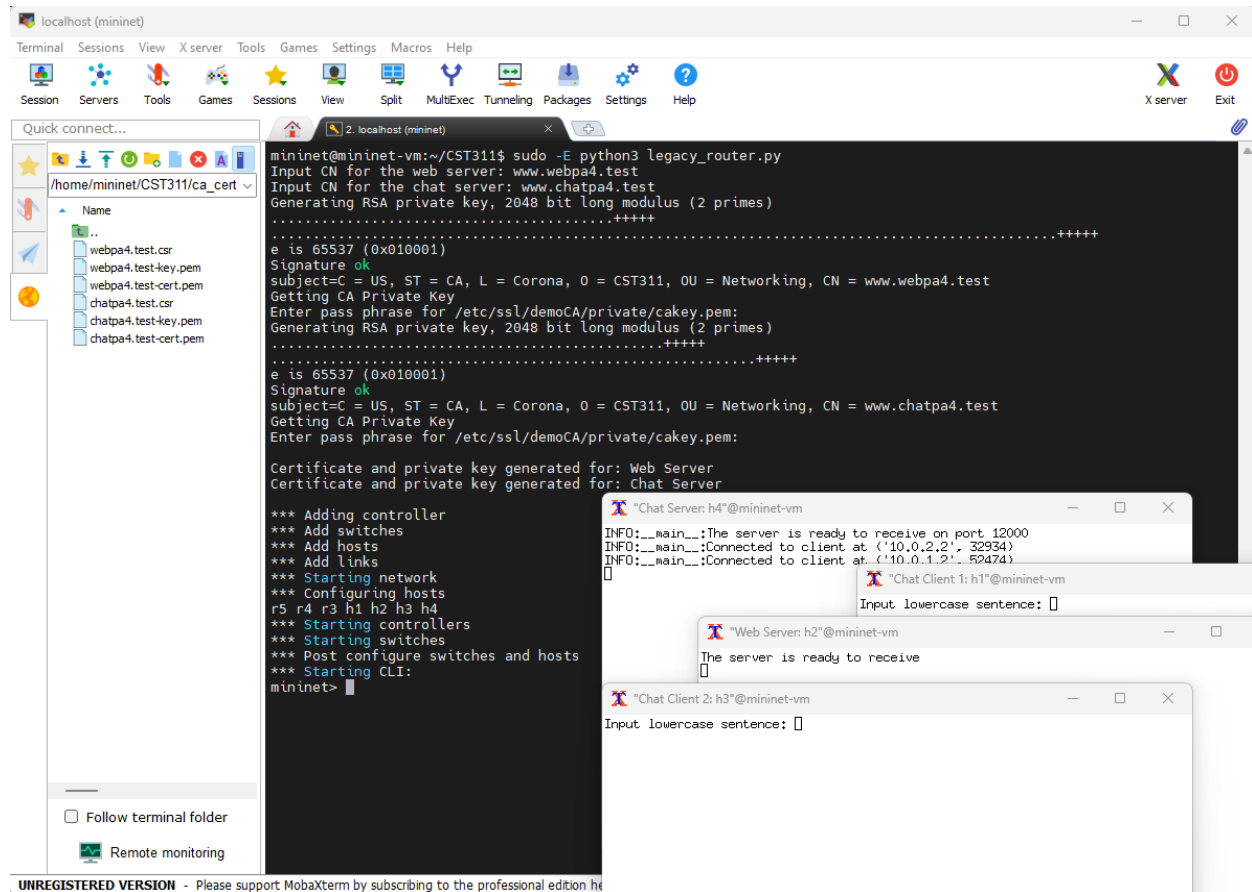Diagram

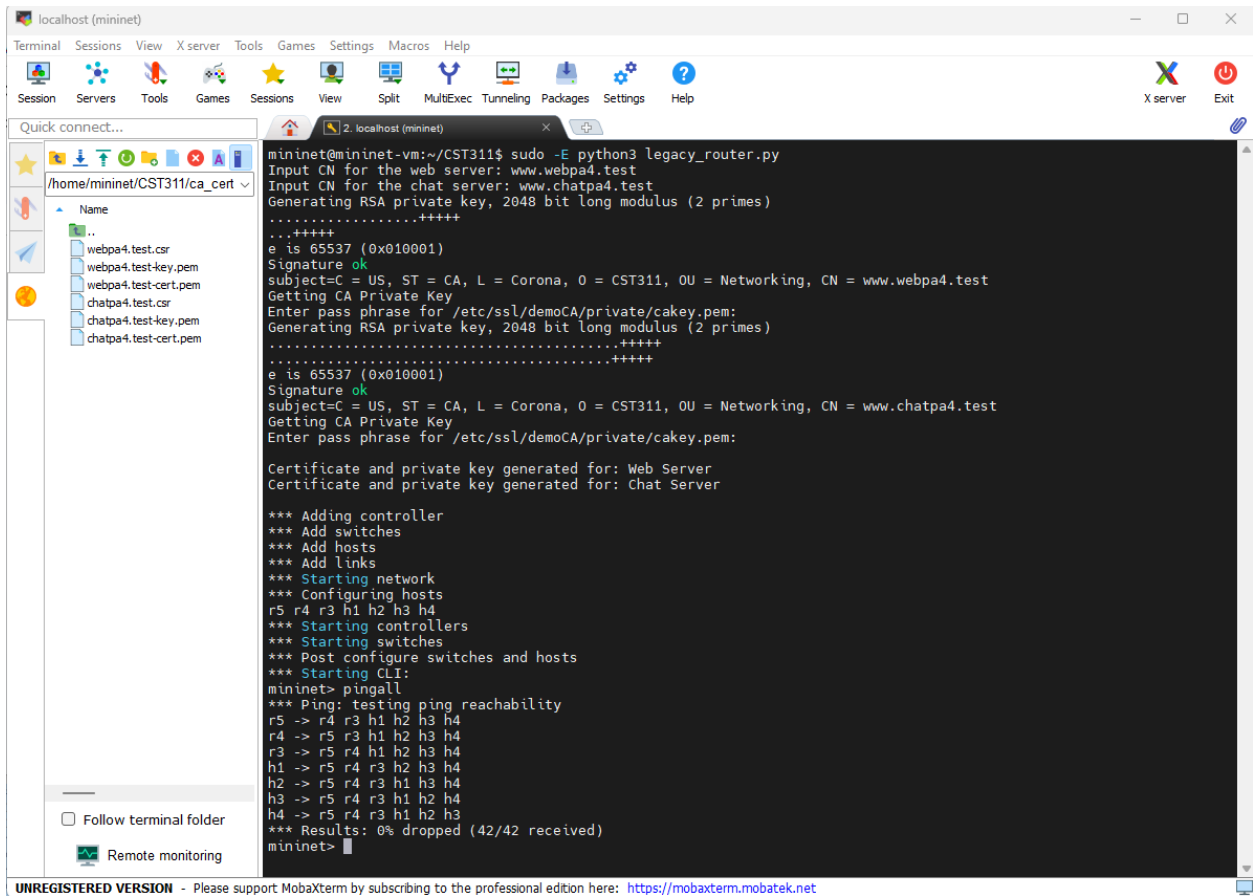## Screenshots

Screen capture of the program that runs with no Python errors:

Screen capture of successful pingall at the mininet> prompt:

Screen capture of a successful chat session between the two chat clients:

"Web Server: h2"@mininet-vm

```
The server is ready to receive
```

root@mininet-vm: ~/CST311

```
INFO:__main__:The server is ready to receive on port 12000
INFO:__main__:Connected to client at ('10.0.2.2', 32828)
INFO:__main__:Connected to client at ('10.0.1.2', 52364)
INFO:__main__:Recieved query test "hi"
INFO:__main__:Recieved query test "hello"
root@mininet-vm:~/CST311#
```

root@mininet-vm: ~/CST311

```
Input lowercase sentence: hi
From Server:
Y : "hi", X : "hello"
root@mininet-vm:~/CST311#
```

root@mininet-vm: ~/CST311

```
Input lowercase sentence: hello
From Server:
Y : "hi", X : "hello"
root@mininet-vm:~/CST311#
```

Screen capture of a Wireshark trace of the communication between a chat client and the chat server:
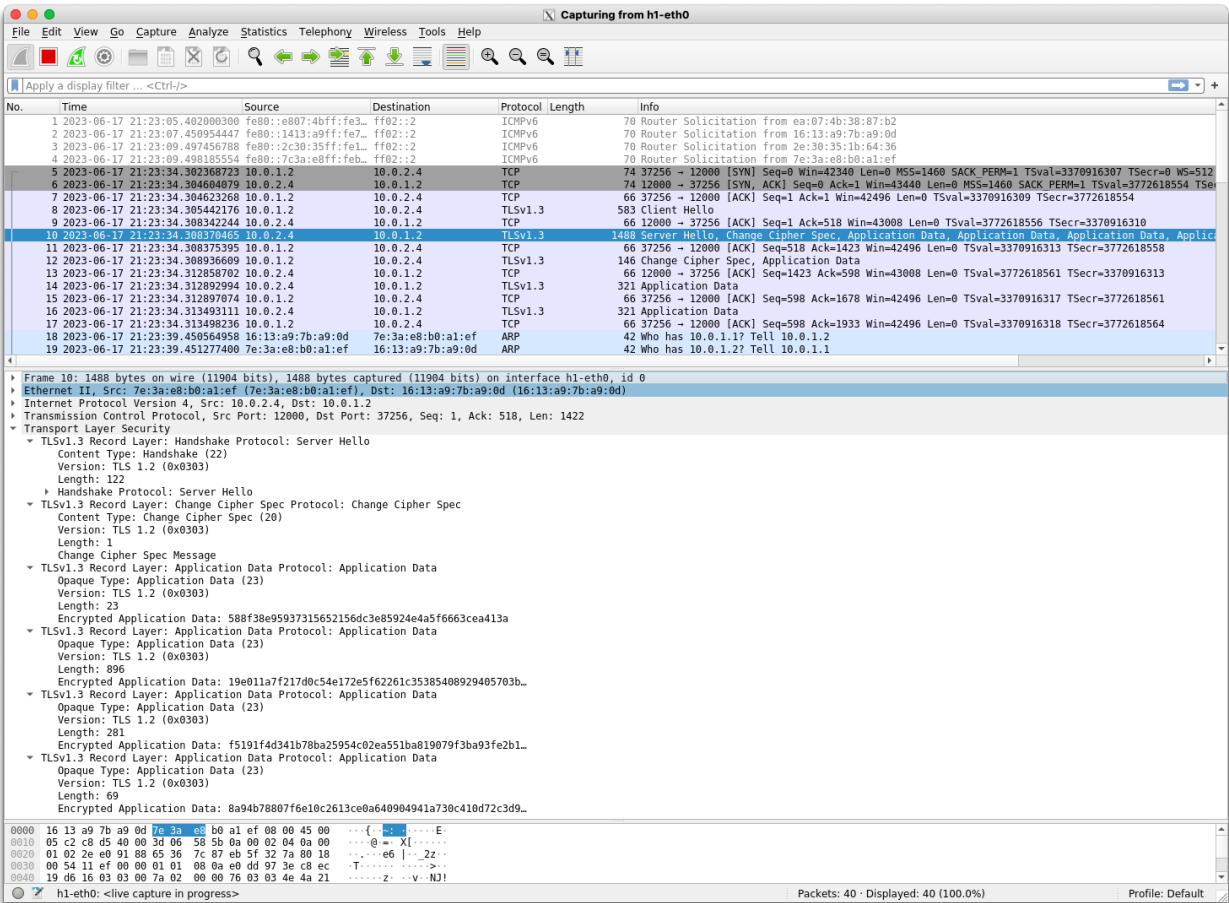
Screen capture of the successful wget (or curl) of the web server index file:


```
root@mininet-vm: ~/CST311

INFO:__main__:The server is ready to receive on port 12000
INFO:__main__:Connected to client at ('10.0.1.2', 52400)
INFO:__main__:Connected to client at ('10.0.2.2', 32864)
INFO:__main__:Recieved query test "hi"
INFO:__main__:Recieved query test "hello"
root@mininet-vm:~/CST311# []
```
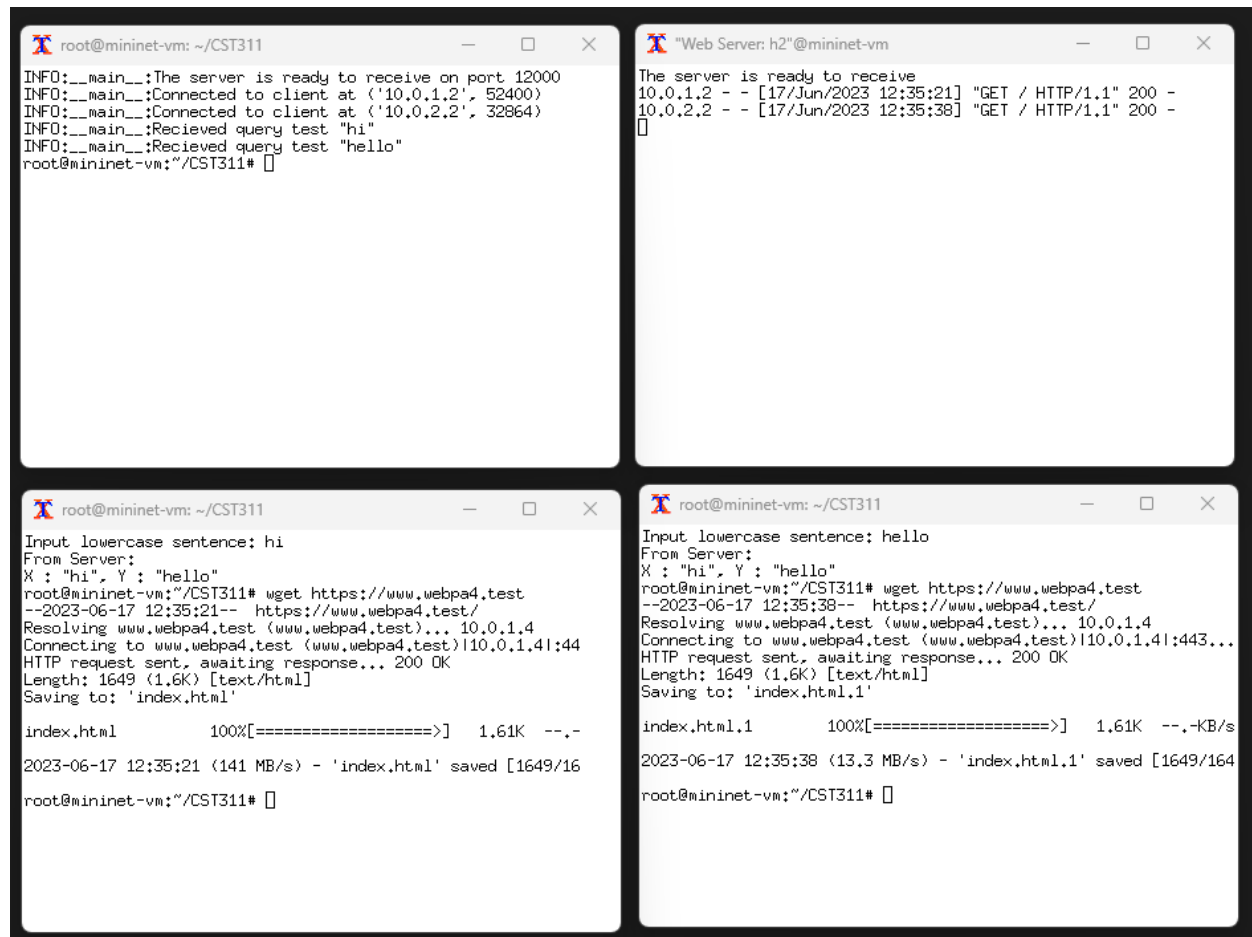
```
"Web Server: h2"@mininet-vm

The server is ready to receive
10.0.1.2 - - [17/Jun/2023 12:35:21] "GET / HTTP/1.1" 200 -
10.0.2.2 - - [17/Jun/2023 12:35:38] "GET / HTTP/1.1" 200 -
[]
```

```
root@mininet-vm: ~/CST311

Input lowercase sentence: hi
From Server:
X : "hi", Y : "hello"
root@mininet-vm:~/CST311# wget https://www.webpa4.test
--2023-06-17 12:35:21--  https://www.webpa4.test/
Resolving www.webpa4.test (www.webpa4.test)... 10.0.1.4
Connecting to www.webpa4.test (www.webpa4.test)|10.0.1.4|:44
HTTP request sent, awaiting response... 200 OK
Length: 1649 (1.6K) [text/html]
Saving to: 'index.html'

index.html          100%[===================>]   1.61K  --.-

2023-06-17 12:35:21 (141 MB/s) - 'index.html' saved [1649/16

root@mininet-vm:~/CST311# []
```

```
root@mininet-vm: ~/CST311

Input lowercase sentence: hello
From Server:
X : "hi", Y : "hello"
root@mininet-vm:~/CST311# wget https://www.webpa4.test
--2023-06-17 12:35:38--  https://www.webpa4.test/
Resolving www.webpa4.test (www.webpa4.test)... 10.0.1.4
Connecting to www.webpa4.test (www.webpa4.test)|10.0.1.4|:443...
HTTP request sent, awaiting response... 200 OK
Length: 1649 (1.6K) [text/html]
Saving to: 'index.html.1'

index.html.1        100%[===================>]   1.61K  --.-KB/s

2023-06-17 12:35:38 (13.3 MB/s) - 'index.html.1' saved [1649/164

root@mininet-vm:~/CST311# []
```
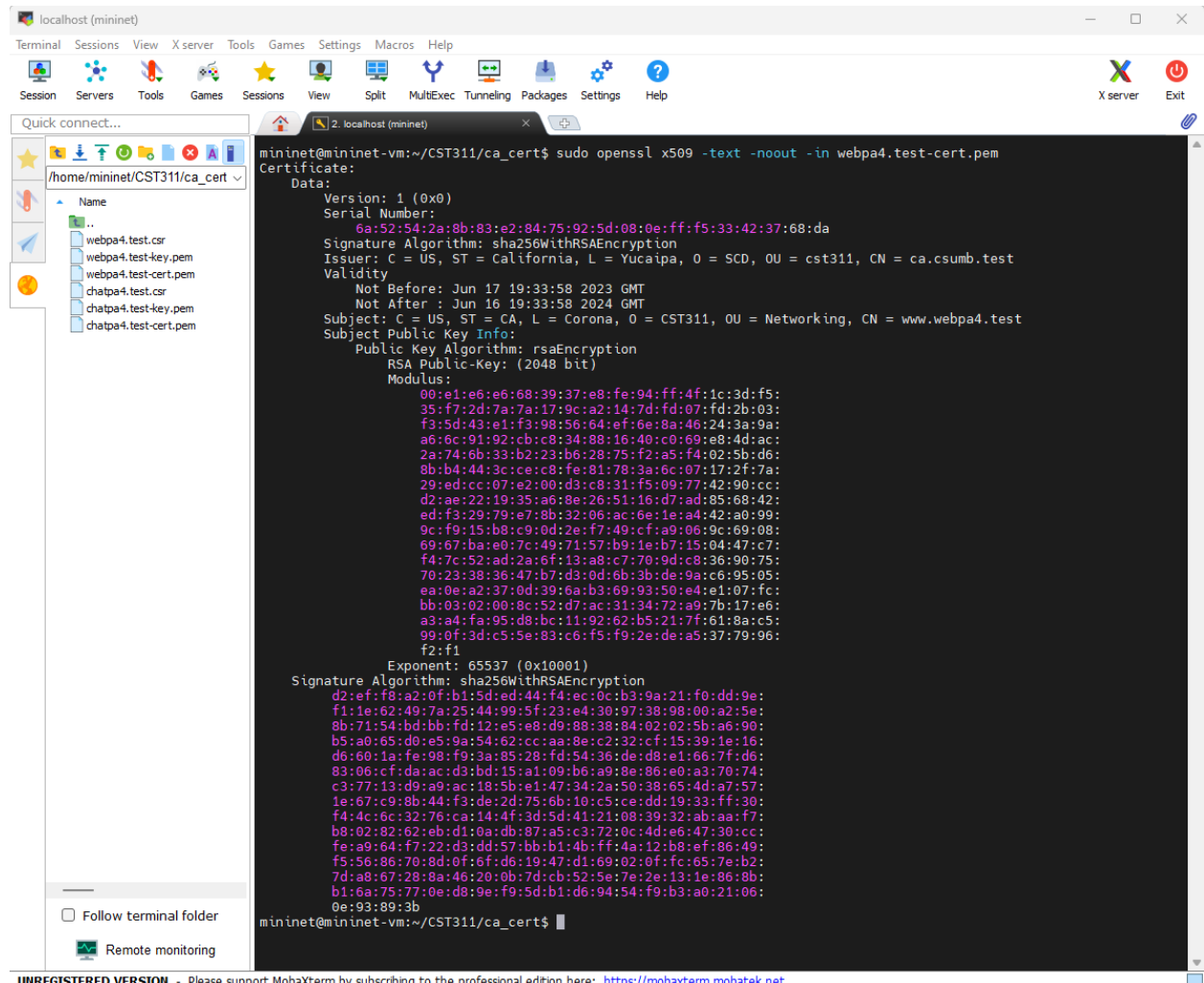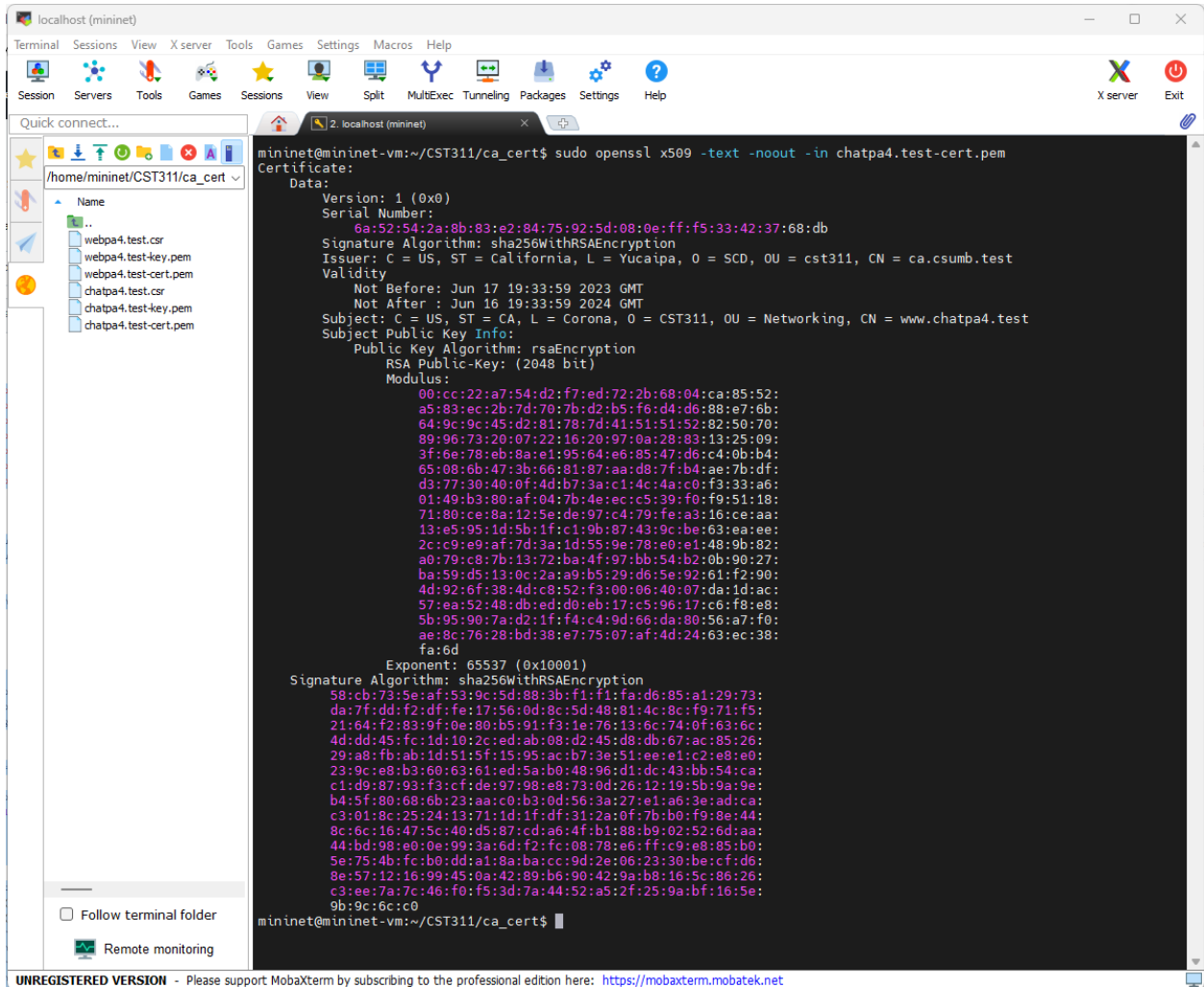
Screenshot of both decrypted server (web and chat) certificates.
Web server certificate:

Chat server certificate:


```
mininet@mininet-vm:~/CST311/ca_cert$ sudo openssl x509 -text -noout -in chatpa4.test-cert.pem
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            6a:52:54:2a:8b:83:e2:84:75:92:5d:08:0e:ff:f5:33:42:37:68:db
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = US, ST = California, L = Yucaipa, O = SCD, OU = cst311, CN = ca.csumb.test
        Validity
            Not Before: Jun 17 19:33:59 2023 GMT
            Not After : Jun 16 19:33:59 2024 GMT
        Subject: C = US, ST = CA, L = Corona, O = CST311, OU = Networking, CN = www.chatpa4.test
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:cc:22:a7:54:d2:f7:ed:72:2b:68:04:ca:85:52:
                    a5:83:ec:2b:7d:70:7b:d2:b5:f6:d4:d6:88:e7:6b:
                    64:9c:9c:45:d2:81:78:7d:41:51:51:52:82:50:70:
                    89:96:73:20:07:22:16:20:97:0a:28:83:13:25:09:
                    3f:6e:78:eb:8a:e1:95:64:e6:85:47:d6:c4:0b:b4:
                    65:08:6b:47:3b:66:81:87:aa:d8:7f:b4:ae:7b:df:
                    d3:77:30:40:0f:4d:b7:3a:c1:4c:4a:c0:f3:33:a6:
                    01:49:b3:80:af:04:7b:4e:ec:c5:39:f0:f9:51:18:
                    71:80:ce:8a:12:5e:de:97:c4:79:fe:a3:16:ce:aa:
                    13:e5:95:1d:5b:1f:c1:9b:87:43:9c:be:63:ea:ee:
                    2c:c9:e9:af:7d:3a:1d:55:9e:78:e0:e1:48:9b:82:
                    a0:79:c8:7b:13:72:ba:4f:97:bb:54:b2:0b:90:27:
                    ba:59:d5:13:0c:2a:a9:b5:29:d6:5e:92:61:f2:90:
                    4d:92:6f:38:4d:c8:52:f3:00:06:40:07:da:1d:ac:
                    57:ea:52:48:db:ed:d0:eb:17:c5:96:17:c6:f8:e8:
                    5b:95:90:7a:d2:1f:f4:c4:9d:66:da:80:56:a7:f0:
                    ae:8c:76:28:bd:38:e7:75:07:af:4d:24:63:ec:38:
                    fa:6d
                Exponent: 65537 (0x10001)
    Signature Algorithm: sha256WithRSAEncryption
         58:cb:73:5e:af:53:9c:5d:88:3b:f1:f1:fa:d6:85:a1:29:73:
         da:7f:dd:f2:df:fe:17:56:0d:8c:5d:48:81:4c:8c:f9:71:f5:
         21:64:f2:83:9f:0e:80:b5:91:f3:1e:76:13:6c:74:0f:63:6c:
         4d:dd:45:fc:1d:10:2c:ed:ab:08:d2:45:d8:db:67:ac:85:26:
         29:a8:fb:ab:1d:51:5f:15:95:ac:b7:3e:51:ee:e1:c2:e8:e0:
         23:9c:e8:b3:60:63:61:ed:5a:b0:48:96:d1:dc:43:bb:54:ca:
         c1:d9:87:93:f3:cf:de:97:98:e8:73:0d:26:12:19:5b:9a:9e:
         b4:5f:80:68:6b:23:aa:c0:b3:0d:56:3a:27:e1:a6:3e:ad:ca:
         c3:01:8c:25:24:13:71:1d:1f:df:31:2a:0f:7b:b0:f9:8e:44:
         8c:6c:16:47:5c:40:d5:87:cd:a6:4f:b1:88:b9:02:52:6d:aa:
         44:bd:98:e0:0e:99:3a:6d:f2:fc:08:78:e6:ff:c9:e8:85:b0:
         5e:75:4b:fc:b0:dd:a1:8a:ba:cc:9d:2e:06:23:30:be:cf:d6:
         8e:57:12:16:99:45:0a:42:89:b6:90:42:9a:b8:16:5c:86:26:
         c3:ee:7a:7c:46:f0:f5:3d:7a:44:52:a5:2f:25:9a:bf:16:5e:
         9b:9c:6c:c0
mininet@mininet-vm:~/CST311/ca_cert$
```

List of lines that were changed and why:
The lines that add the routers (r1, r2, and r3) (lines 40-45) were updated to have a valid IP address with their subnet mask, which is needed to identify each router. The subnet mask allows the routers to be on the same network since the first 24 bits of their IP address are the same. The lines that add the hosts (h1, h2, h3, and h4) (lines 48-51) were updated to include the subnet mask as well and the default route for each host. The subnet masks were added to also allow the hosts to be on the same network since the first 24 bits of their IP addresses will be the same. The hosts will also be on the same network as the routers since they are both given the same subnet mask. Default routes are given to forward a packet to a location if the packet does not have the same local subnet.
The lines 58-61 were modified to assign IP addresses to the links between the routers and switches. Without these IP addresses the routers and switches would not be connected together. The lines to create 6 static routes were added in lines 66-82 to provide static routes between the routers. The static routes allow the packets to have a path from one host to another host.

Questions:
1. What were any interesting findings and lessons learned?
   We learned that configuring IP addresses and routes are important in order for the network to function correctly. Static routes need to have subnets and the next hop in order to route properly. It is also important for the newly created certificate and key to be placed in the same path that the web and chat server have in order for the servers to use the certificate. We also learned that the makeTerm function allows for hosts to run commands that are given to it so the user does not have to manually input the commands.

2. Why didn't the original program forward packets between the hosts?

   The original program does not forward packets because the IP addresses in lines 26, 30, and 32 are '0.0.0.0'. If all the routers had the same IP address of 0.0.0.0, packets will not be able to be forward to a specific router, but providing a specific IP fixes this. There are also no routing algorithms in the original program so the packets are not directed where to go.

3. Is the line ' r3.cmd('sysctl -w net.ipv4.ip_forward=1') ' required?
   The line is required because it allows packets to be forward onto the network. This line allows the packets to move past r3. Without this line the packets would not be forward to the next router which would cause the packets to never get to the destination. This would cause routing issues and h3 and h4 would never receive the packets from h1 and h2. If packets are not able to reach their destination, then the networks would not communicate properly and the "internet" would not work.

4. Intentionally break your working program, e.g.: change a subnet length, IP address, or default route for a host. Explain why your change caused the network to break.

   Changing any one of these examples would cause the network to break because the packets would not be routed correctly. If s subnet length is changed then the subnet of the

hosts would not match. Changing the IP address would cause a routing issue because the packet would not be sent to the correct location or the packet might not be sent anywhere if the IP address is invalid. Changing a default route for a host would cause the network to break because packets will not be forwarded correctly to the next hop.