Name : Yikwenmein Victor Magheng
Personal number: 19950218T614
course code: ET2595(Network and System Security)

Task 1: [v3_ca]

This task requires looking up for enabled extensions in the v3_ca section and then giving the meaning of each extension, values assigned to these extensions and what they mean. To complete this task, I opened the file with command "gedit openssl.cnf". I found the following enabled extensions within [v3_ca] section

1. **subjectKeyIdentifier:** It specifies how to identify the public key being certified. The only value supported for the subjectkeyidentifier is hash. This field is required if x509_extensions is specified.
2. **authorityKeyIdentifier**:It specifies how to identify the public key being used to verify the signature on this certificate, and enables keys used by the same CA to be distinguished.It has the following values;
   - **`keyid:always`**
     indicates that the subject key identifier is copied from the parent certificate and an error is returned if the copy fails.
   - **`issuer`**
     indicates that the issuer and serial number is copied from the issuer certificate if the *keyid* option fails or is not specified.

3. **basicConstraints**: indicates whether a certificate is a certificate authority (CA).
   - The value "CA:true" indicates whether the certificate is certificate authority or not indicated by either true or false.
   - "Critical" indicates the extension will be critical
4. **keyUsage**: specifies permitted key usages, where *keyusage* values are a comma-separated list of any of the following:

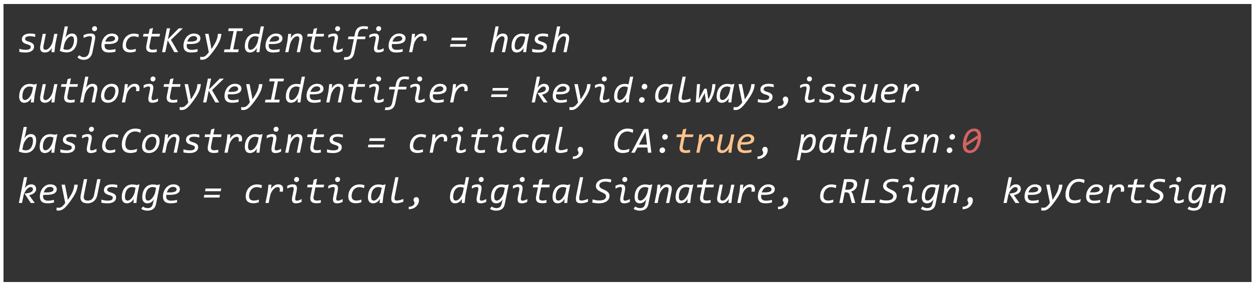
- digitalSignature
- nonRepudiation
- keyEncipherment
- dataEncipherment
- keyAgreement
- keyCertSign

- cRLSign
- encipherOnly
- decipherOnly.
- “Critical” as well indicates that the extension will be critical

Task 2:[v3_intermediate_ca]

For this task, I added the following to the to the “openssl.cnf” file

[v3_intermediate_ca]



```
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign
```

The following extensions and values can be identified

1. **subjectKeyIdentifier = hash:** It specifies how to identify the public key being certified and “hash” indicates the method which is used for generating keyIdentifiers.
2. **authorityKeyIdentifier = keyid:always,issuer:** This specifies how to identify the public key being used to verify the signature on this certificate, and enables keys used by the same CA to be distinguished.It has the following values;
    - **`keyid:always`**
      indicates that the subject key identifier is copied from the parent certificate and an error is returned if the copy fails.
    - **`issuer`**
      indicates that the issuer and serial number is copied from the issuer certificate if the *keyid* option fails or is not specified.

3. **basicConstraints:critical, CA:true, pathlen:0** It indicates whether a certificate is a certificate authority (CA).
    - The value “CA:true” indicates whether the certificate is certificate authority or not indicated by either true or false.

- “Critical” indicates the extension will be critical
- The pathlen parameter indicates the maximum number of CAs that can appear below this one in a chain. A pathlen of zero indicates the CA can only be used to sign end user certificates and not further CAs.

4. keyUsage = critical, digitalSignature, cRLSign, keyCertSign: specifies permitted key usages, where *keyusage* values are a comma-separated list of the following:

- digitalSignature
- keyEncipherment
- cRLSign
- keyCertSign
- “Critical” indicates that the extension will be critical

The only difference here is the **pathlen:0** value for the **basicConstraints** with a zero indicating the CA can only be used to sign end user certificates and not further CAs.

Task 3: [usr_cert]

As in the previous task, I looked up for enabled extensions within the [usr_cert] section, their meanings and their values and what they mean as well. So I found the following enabled extensions within [v3_ca] section

1. **keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment.** The key usage extension defines the purpose of the key contained in the certificate
   - The digitalSignature bit is asserted when the subject public key is used for verifying digital signatures, other than signatures on certificates and CRLs , such as those used in an entity authentication service, a data origin authentication service, and/or an integrity service.
   - The nonRepudiation bit is asserted when the subject public key is used to verify digital signatures, other than signatures on certificates (bit 5) and CRLs (bit 6), used to provide a non-repudiation service that protects against the signing entity falsely denying some action
   - The keyEncipherment bit is asserted when the subject public key is used for enciphering private or secret keys, i.e., for key transport
   - “Critical” indicates that the extension will be critical

2. **basicConstraints=CA:FALSE** : The basic constraints extension identifies whether the subject of the certificate is a CA and the maximum depth of valid certification paths that include this certificate
   - The CA boolean indicates whether the certified public key may be used to verify certificate signatures. FALSE in this case indicates the certified public key is not used to verify certificate signatures

3. **subjectKeyIdentifier=hash**:The subject key identifier extension provides a means of identifying certificates that contain a particular public key and “**hash**” indicates the method which is used for generating keyIdentifiers.
4. **authorityKeyIdentifier=keyid,issuer :** The authority key identifier extension provides a means of identifying the public key corresponding to the private key used to sign a certificate. This extension is used where an issuer has multiple signing keys
   - **Keyid** indicates that the subject key identifier is copied from the parent certificate.
   - **Issuer** indicates that the issuer and serial number is copied from the issuer certificate if the *keyid* option fails or is not specified
5. **extendedKeyUsage=clientAuth, emailProtection**: This extension indicates one or more purposes for which the certified public key may be used, in addition to or in place of the basic purposes indicated in the key usage extension. In general, this extension will appear only in end entity certificates
   - **clientAuth** is for TLS WWW client authentication
   - **emailProtection** is for Email protection

Task 4: [server_cert]

For this task, I added a [server_cert] section and within it,the following were added

```
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
```

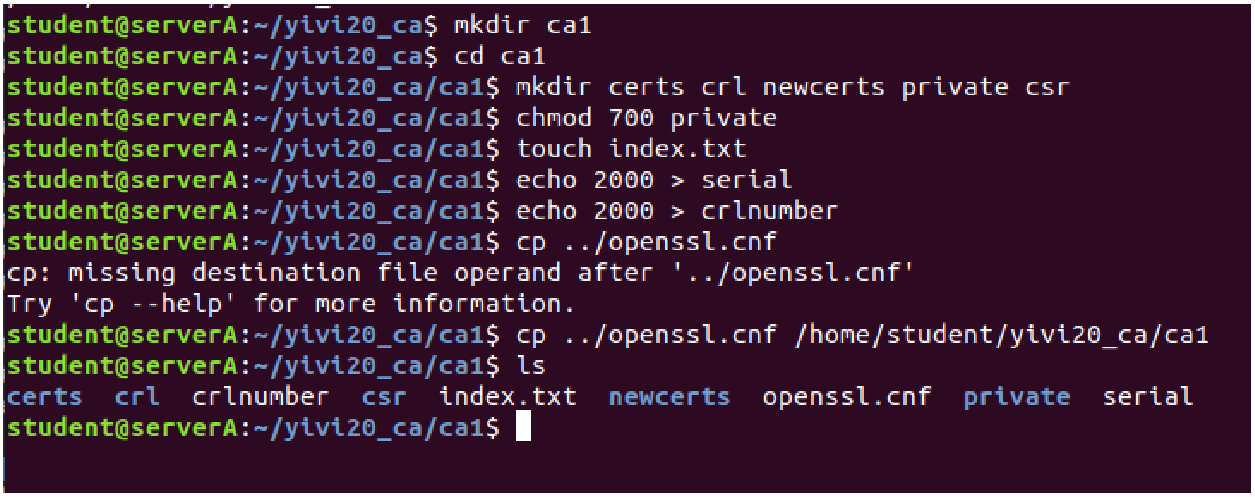
We have the following enabled extensions.

1. The basic constraints extension identifies whether the subject of the certificate is a CA and the maximum depth of valid certification paths that include this certificate
   - The CA boolean indicates whether the certified public key may be used to verify certificate signatures. FALSE in this case indicates the certified public key is not used to verify certificate signatures

2. The subject key identifier extension provides a means of identifying certificates that contain a particular public key and "**hash**" indicates the method which is used for generating keyIdentifiers.
3. The authority key identifier extension provides a means of identifying the public key corresponding to the private key used to sign a certificate. This extension is used where an issuer has multiple signing keys
   - **`Keyid`** indicates that the subject key identifier is copied from the parent certificate.
   - **`Issuer:always`** indicates that the issuer and serial number is always copied from the issuer certificate
4. The key usage extension defines the purpose of the key contained in the certificate
   - The "Critical" value indicates that the extension will be critical
   - The digitalSignature bit is asserted when the subject public key is used for verifying digital signatures, other than signatures on certificates and CRLs , such as those used in an entity authentication service, a data origin authentication service, and/or an integrity service.
   - The keyEncipherment bit is asserted when the subject public key is used for enciphering private or secret keys, i.e., for key transport

Task 5: Policies

For this task, I created the ca1 directory and subdirectories as instructed



```
student@serverA:~/yivi20_ca$ mkdir ca1
student@serverA:~/yivi20_ca$ cd ca1
student@serverA:~/yivi20_ca/ca1$ mkdir certs crl newcerts private csr
student@serverA:~/yivi20_ca/ca1$ chmod 700 private
student@serverA:~/yivi20_ca/ca1$ touch index.txt
student@serverA:~/yivi20_ca/ca1$ echo 2000 > serial
student@serverA:~/yivi20_ca/ca1$ echo 2000 > crlnumber
student@serverA:~/yivi20_ca/ca1$ cp ../openssl.cnf
cp: missing destination file operand after '../openssl.cnf'
Try 'cp --help' for more information.
student@serverA:~/yivi20_ca/ca1$ cp ../openssl.cnf /home/student/yivi20_ca/ca1
student@serverA:~/yivi20_ca/ca1$ ls
certs  crl  crlnumber  csr  index.txt  newcerts  openssl.cnf  private  serial
student@serverA:~/yivi20_ca/ca1$ 
```

I also set values for **x509_extensions** and **policy** extensions

- In the "**policy_match**" policy, all fields listed as "match" must contain the exact same contents as that field in the CA's DN. All fields listed as "supplied" must be present. All fields listed as "optional" are allowed, but not required to be there. Anything allowed must be listed! So this policy requires the same country, State, and Organization name as the CA for all certs it signs.
- In a "**policy_anything**" policy, we accept anything, and only require a CN. We can refer to this with a -policy policy_anything.

1. **countryName**

In the [policy_match] section, the sample *openssl.cnf* file has a value of ”match” for this attribute. In the [policy_anything] section, the sample *openssl.cnf* file has a value of “optional” for this attribute

2. **stateOrProvinceName.** In the [policy_match] section, the sample *openssl.cnf* file has a value of ``match" for this attribute. In the [ policy_anything ] section, the sample *openssl.cnf* file has a value of ``optional" for this attribute.
3. **localityName.** This attribute does not appear in the [policy_match] section of the sample *openssl.cnf* file. In the [policy_anything] section, the sample *openssl.cnf* file has a value of ``optional" for this attribute
4. **organizationName.** In the [policy_match] section, the sample *openssl.cnf* file has a value of ``match" for this attribute.

In the [policy_anything] section, the sample *openssl.cnf* file has a value of “optional" for this attribute

5. **organizationalUnitName.** This attribute has an ``optional" value in both the policy_match and [policy_anything] sections of the sample *openssl.cnf* file.
6. **commonName.** This attribute has a “supplied” value in both the policy_match and [policy_anything] sections of the sample *openssl.cnf* file.
7. **emailAddress.** This attribute has an “optional” value in both the policy_match and [policy_anything] sections of the sample *openssl.cnf* file.

Task 6: Options for the root certificate

For this task, I followed instructions as stated in the lab document and ran the following commands from my root CA directory(`/home/student/yivi20_ca`) to create the private RSA key
for root and ca1, respectively:

```
openssl genrsa -aes256 -out private/root.key.pem 4096
openssl genrsa -aes256 -out ca1/private/ca1.key.pem 4096
```

Protected file system access rights to private keys is restricted to the owner with :

```
chmod 400 private/root.key.pem
chmod 400 ca1/private/ca1.key.pem
```

Generated self-signed certificate entering the following command

```
openssl req -config openssl.cnf -key private/root.key.pem -new -x509
-days 7300 -sha256 -extensions v3_ca -out certs/root.cert.pem
```

I used `man req` command to check the meaning of options and values they take in the above command.

- `config` option: It allows an alternative configuration file to be specified and in this case it specifies `openssl.cnf` as the configuration file.

- key option: This specifies the file to read the private key from. So the command specifies that private key should be read from the following file `private/root.key.pem`
- new option: This option generates a new certificate request. It will prompt the user for the relevant field values
- X509 option: this option outputs a self signed certificate instead of a certificate request. This is typically used to generate a test certificate or a self signed root CA
- days option:when the -x509 option is being used this specifies the number of days to certify the certificate for. 7300 specifies that it will be valid for 7300 days.The default is 30 days
- sha256 option: specifies that the cryptographic algorithm for the process is sha256
- extension option: these options specify alternative sections to include certificate extensions (if the -x509 option is present) or certificate request extensions.
- out option:This specifies the output filename to write to or standard output by default. It specifies that output should be written to the file `certs/root.cert.pem` in this case.

Task 7: Verify the root certificate

For this task, I used the following command `openssl x509 -noout -text -in certs/root.cert.pem`
to verify root certificate The following is the output of the command

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            c2:f9:1c:6a:8d:f9:4c:c4
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = SE, ST = Blekinge, L = Karlskrona, O = 
ET2540, CN = yivi20Root
        Validity
            Not Before: May 14 13:53:34 2021 GMT
            Not After : May  9 13:53:34 2041 GMT
        Subject: C = SE, ST = Blekinge, L = Karlskrona, O =
ET2540, CN = yivi20Root
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:

00:a3:a9:f7:49:67:83:0f:dc:0b:98:e1:67:00:ce:

de:f3:e7:71:6b:41:46:a5:01:69:af:b6:5c:d3:ab:

74:07:aa:ab:84:70:a7:2f:29:2b:35:8b:c5:99:70:

f8:7b:5b:24:d0:90:1d:bf:75:d8:ad:79:17:88:73:

7f:55:4c:09:30:19:66:93:61:1b:98:e4:3f:cf:b9:

c3:0b:b3:67:1e:03:65:f4:92:58:a4:a2:69:8d:4e:

47:ee:ee:42:ea:66:e9:ee:1a:b9:35:f3:05:bb:a4:

c9:f6:9a:2f:6c:20:c2:3f:5d:fd:a5:3b:fd:83:b4:

f0:62:4e:31:d6:b3:c8:27:8d:6c:82:6a:d8:d3:29:

2b:b9:0e:36:62:80:9b:af:96:75:c8:76:f3:c7:b8:

10:5b:93:22:e1:1d:44:6e:80:99:92:e6:4e:23:47:

5e:5b:a3:93:b7:fc:b3:b4:71:d4:86:d5:bf:41:59:

33:0d:a3:d6:75:b6:95:59:1e:52:37:f7:3b:64:61:

24:ac:67:a9:7f:bb:4d:68:de:ec:5a:6a:1b:c8:34:

59:a1:b9:5a:96:8d:bb:b8:d4:5f:19:5b:7f:11:d1:
```

```
0d:65:56:ab:9c:02:94:36:6c:a6:f9:ed:74:c4:9c:
12:4a:bf:c7:9c:57:cf:55:bb:37:82:6e:4b:50:a8:
4c:75:f5:d6:55:cb:c8:88:97:7d:22:f0:de:9b:91:
16:ee:c8:2e:9d:2d:ab:fa:87:cd:5c:56:c0:74:78:
2b:c8:41:59:47:fc:51:0e:14:49:69:c1:dd:fd:ac:
f9:b7:6a:d2:e9:15:f8:73:f7:6b:91:6a:a3:5e:11:
ed:37:c6:8d:4c:25:fc:f7:5a:3a:0b:a9:dc:bb:e2:
a6:94:32:26:39:a9:3a:8a:87:90:2b:a8:cd:a7:b5:
f9:da:21:c7:28:dd:b9:62:03:16:b7:86:43:d3:43:
fa:6c:f9:bd:7e:5d:62:c5:49:12:01:6f:44:6c:4f:
c7:57:75:e8:01:09:54:ce:ad:22:70:1a:bd:2b:19:
7d:a2:97:24:c9:76:b4:a9:67:db:0f:3d:f9:95:00:
a7:f4:31:c4:b7:ec:2b:b7:f5:71:4d:fd:79:73:d6:
c5:f2:95:a1:c2:b9:b2:8a:ed:6f:88:1e:9e:c9:ba:
8a:ce:78:8f:34:97:fd:87:7f:33:b2:f5:d5:94:af:
a0:eb:ed:51:93:78:03:39:28:63:00:51:7a:ea:cc:
3c:48:95:4e:75:32:9f:3e:4f:10:45:cb:b4:76:a0:
92:74:06:b5:9f:6a:9f:a5:cc:fa:8e:00:6d:96:d0:
```

```
b2:17:ec:f9:4b:bb:73:d3:e5:83:51:cd:0e:3f:9b:
                    f4:d4:b7
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:

DB:63:33:FF:F0:6D:BB:1D:DC:90:38:04:47:0B:88:70:43:3B:8E:19
            X509v3 Authority Key Identifier:

keyid:DB:63:33:FF:F0:6D:BB:1D:DC:90:38:04:47:0B:88:70:43:3B:8E:1
9

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
    Signature Algorithm: sha256WithRSAEncryption
         5f:38:38:d8:5d:19:6f:81:0d:be:a0:ed:17:80:1a:cb:92:13:
         08:0c:8b:80:de:91:25:51:ad:0e:7c:79:b9:9c:f2:90:58:68:
         69:9f:46:2c:b6:01:82:c3:66:39:93:8c:a0:05:8a:f2:f9:e1:
         9b:df:6e:a5:91:9f:6c:50:32:d6:2b:24:9c:b7:a0:b2:07:68:
         7b:39:73:2c:2b:8d:9d:14:b0:22:dc:e4:35:9d:7b:e9:86:5c:
         fd:4a:00:05:dd:7c:69:c9:7d:ba:b7:30:22:02:93:96:c7:4a:
         e9:03:5b:e7:04:f6:65:87:c5:b7:7e:c9:e4:18:b4:a0:7e:e0:
         76:20:51:eb:60:40:7e:50:f2:d3:a8:62:27:d4:da:25:ee:bd:
         b7:dd:53:1d:58:14:9f:d5:c3:5a:22:d1:f9:64:0d:af:87:45:
         83:04:2f:b2:e0:9f:d2:e4:cd:46:12:fa:2d:93:4a:43:61:d4:
         51:00:57:9b:40:25:b0:f5:d3:42:8d:20:e5:76:17:63:94:c5:
         18:0b:ac:ef:c2:1b:ba:e3:6b:ca:6c:00:5f:e1:0c:93:95:a8:
         56:b4:a8:13:f0:8a:d4:c5:36:8d:a2:5f:09:6e:7b:f1:c7:e7:
         70:97:cb:69:f1:ea:01:65:6b:52:7f:b7:a5:d1:a5:af:a2:20:
         89:aa:69:e3:82:4f:ce:a2:e8:36:6a:18:b0:34:91:cd:da:eb:
         76:1d:8b:a4:bb:84:23:b9:dc:bd:5e:db:60:95:3b:2e:b4:76:
         a9:c5:29:fc:c1:f6:2b:47:c9:74:37:62:3f:95:87:f7:17:64:
         cc:9a:55:df:ed:6c:4a:78:b6:c1:89:10:5c:0a:ed:50:c6:cc:
```

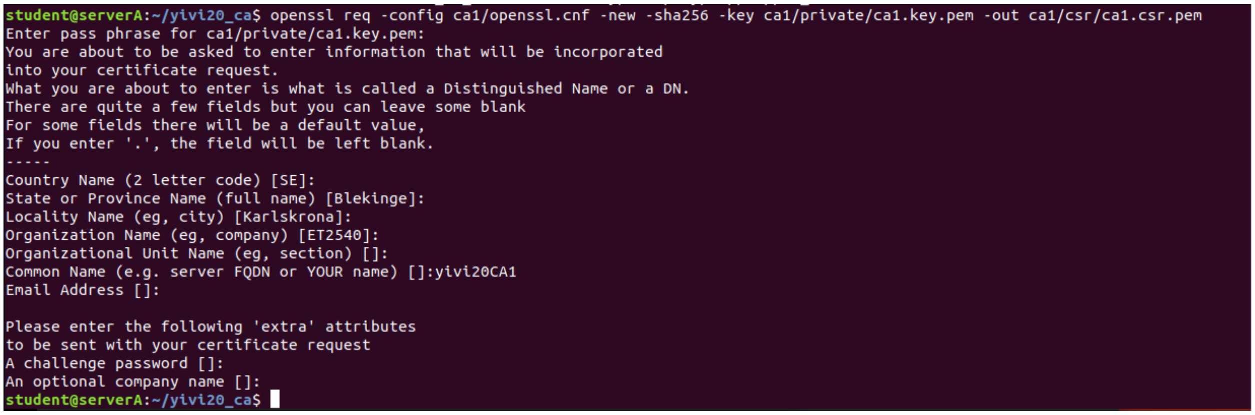
```
                    28:3a:10:49:cc:68:92:e2:41:37:16:0c:7c:d3:c5:05:17:a9:
                    b6:96:8b:80:f4:44:4e:e6:ea:8d:8c:ae:e4:23:75:6d:f5:4c:
                    24:a0:93:02:69:47:9c:ed:52:4e:da:3b:a8:65:6c:b7:0d:80:
                    f7:41:b1:c0:7b:b8:21:fc:1d:f1:cc:4d:75:9a:7d:a6:b4:e4:
                    df:3d:76:b4:c7:10:8d:8c:f1:4b:3f:17:0a:43:ca:f6:34:c9:
                    b0:dc:55:cb:16:47:aa:1c:4e:2c:4a:1e:2c:90:c8:a9:39:c8:
                    93:f8:f3:5e:81:2c:44:bc:56:84:f7:20:ea:9f:c9:fa:c2:ff:
                    7a:c9:f1:0b:69:36:14:ee:98:34:38:c3:38:08:be:34:40:23:
                    60:ff:dd:d9:84:b5:0b:50:91:b1:ee:71:c4:f5:5e:02:b0:0b:
                    98:bc:8d:c9:2b:70:0a:f0:e2:00:b5:c0:ff:31:04:51:83:5c:
                    94:cd:dd:2c:42:0d:7e:a2
```

Task 8: Verify the CSR

For this task,the following command

```
openssl req -config ca1/openssl.cnf -new -sha256 -key
ca1/private/ca1.key.pem -out ca1/csr/ca1.csr.pem
```

was first ran to create the CSR



```
student@serverA:~/yivi20_ca$ openssl req -config ca1/openssl.cnf -new -sha256 -key ca1/private/ca1.key.pem -out ca1/csr/ca1.csr.pem
Enter pass phrase for ca1/private/ca1.key.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [SE]:
State or Province Name (full name) [Blekinge]:
Locality Name (eg, city) [Karlskrona]:
Organization Name (eg, company) [ET2540]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:yivi20CA1
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
student@serverA:~/yivi20_ca$
```

Lastly, the following command

```
openssl req -text -noout -verify -in ca1/csr/ca1.csr.pem
```

Was run to verify the CSR and this is the output

```
verify OK
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = SE, ST = Blekinge, L = Karlskrona, O =
ET2540, CN = yivi20CA1
```

```
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:

00:bc:85:4f:59:b3:b7:d5:20:eb:de:2a:29:ef:6d:

d4:48:93:6d:ba:72:28:86:0f:04:e8:3c:9c:bc:0d:

e3:28:a7:66:7f:86:22:79:dd:70:85:8c:66:6f:21:

76:f9:6c:75:18:9f:06:ce:9d:16:cf:5b:ae:31:a3:

44:96:64:95:77:9f:59:93:6b:f6:e3:16:24:28:86:

8e:38:82:cf:de:b9:53:f1:04:fb:8e:21:2c:e0:02:

72:0b:7f:d2:44:25:78:a3:2e:63:dc:f4:ba:a8:56:

db:af:9d:c3:0f:06:69:80:cf:64:fe:c5:21:fe:25:

4a:06:a2:b1:b2:7d:3c:99:35:af:20:13:2a:8a:12:

d8:2e:e6:a4:7d:ab:b7:f3:b1:78:8c:79:86:9c:cc:

e1:bd:45:c2:f5:af:4a:e8:92:7e:47:40:a4:d5:7c:

0c:cf:44:ea:8a:30:49:db:06:eb:b2:79:c8:28:41:

8f:7e:db:4f:77:eb:4c:59:7c:64:01:5e:3f:04:7b:

cb:8f:1e:d2:0b:56:c2:92:3e:be:a3:67:fb:22:1d:

f9:13:c6:24:af:22:0d:4f:e1:90:63:33:f3:5e:4b:

32:fc:6c:54:c7:1e:fb:d3:d3:f1:dd:3f:59:f8:ab:
```

```
db:5c:47:d4:59:fe:11:40:ca:66:a1:f9:b0:f9:8b:

ee:fb:3c:16:ce:5e:3f:3c:49:82:84:b1:b3:48:85:

90:53:85:06:b5:75:e6:08:38:e1:97:8c:ca:d8:2c:

8b:23:e2:ff:80:d8:36:c4:d6:fe:17:63:1a:a4:43:

08:37:96:1e:04:17:5a:b7:3f:ec:13:ec:8c:55:a9:

bb:1e:71:08:e7:2e:68:f7:ac:f5:f5:3f:fe:d2:05:

d2:fe:b7:57:ab:ba:e4:b9:8d:bb:98:b8:03:1a:b2:

a3:70:e5:62:4a:e2:1d:4c:ad:b6:22:c6:ff:7d:32:

1b:bc:a9:78:8c:c3:4e:bc:43:fe:3e:73:44:e3:11:

73:ad:67:97:c6:5c:45:dd:61:7b:66:f1:4d:36:b6:

60:db:11:63:c6:01:9b:0b:1c:af:be:69:4c:ba:e0:

5d:61:70:89:43:8a:63:3c:03:e4:d2:94:71:6a:c7:

52:e9:99:63:ff:d8:85:bc:c3:f4:ec:50:37:7d:be:

55:37:d6:c4:6a:e3:c7:98:e2:eb:81:09:e6:ed:56:

74:e6:52:5c:25:a1:b0:0e:8f:57:1f:f4:9e:78:dd:

eb:ec:a1:e2:18:f3:2f:88:62:8e:8c:9e:40:cd:a8:

0b:a2:d8:8e:6b:97:84:cd:a5:0a:f6:4d:bd:54:6b:

cb:e9:0e:15:8e:aa:f4:3e:6e:4e:54:82:c3:4c:bd:
```

```
                    a1:90:d7
                Exponent: 65537 (0x10001)
        Attributes:
            a0:00
    Signature Algorithm: sha256WithRSAEncryption
         7b:61:7a:43:c3:e0:55:cd:11:77:0c:f7:e5:ee:07:51:4d:fd:
         f6:87:97:fb:58:99:e6:4d:e9:df:ea:d8:c7:83:b6:3a:a4:56:
         b7:7a:6f:a4:9c:46:21:ce:a3:ce:62:69:1a:3c:4e:cb:4f:3d:
         a7:63:13:03:fe:96:60:e2:b0:a0:ad:28:30:41:9c:27:a0:8d:
         de:76:5b:b4:cb:51:84:28:91:2b:c0:50:0c:9d:92:b6:3c:2d:
         9b:17:a1:49:29:90:f9:26:dd:ca:35:db:fd:5a:02:5a:10:bb:
         ed:99:70:c6:cd:8c:0c:de:1c:4d:07:d9:a2:74:b1:83:70:1e:
         82:30:95:29:09:df:0f:f0:52:55:d7:53:e9:86:22:6e:74:82:
         20:2e:06:58:6e:2a:e0:52:9c:95:ae:21:44:32:18:14:c6:71:
         9c:2e:aa:e5:c5:5b:b7:4a:57:19:b3:49:84:60:84:f9:2f:6f:
         18:fa:30:3b:b2:bf:d1:83:84:b2:3c:78:dd:94:a0:7b:bf:ff:
         73:37:55:fc:d8:3c:c5:89:e9:d6:d9:0a:0c:6c:04:bb:69:d7:
         26:64:71:91:dc:10:5a:e0:bf:cb:a0:83:97:ec:62:a3:78:fb:
         91:8f:9a:5d:63:16:91:72:4b:46:20:67:d3:b9:1e:86:63:e0:
         a2:b1:53:5a:17:fa:9a:a8:98:11:07:b7:a6:e3:e6:56:cd:52:
         01:c6:20:f9:3e:39:eb:89:aa:d3:94:d1:36:1d:5b:b8:80:2d:
         32:5b:16:2f:49:0f:e8:70:de:1a:e2:77:65:89:79:bd:de:cc:
         33:05:fe:dc:5a:be:f9:0c:e0:c4:fe:e7:0d:67:ed:21:e0:89:
         c4:6d:10:b7:5d:0e:63:bf:d2:60:b5:08:e2:26:af:43:e4:36:
         43:52:13:5b:7e:ee:51:cd:94:fd:19:e3:5c:aa:9f:a9:4f:7f:
         ad:db:54:89:14:51:7f:80:e7:b1:93:fa:1f:3b:91:69:69:3d:
         97:4e:09:85:4a:fb:85:db:94:ee:81:28:d4:81:a9:55:91:d5:
         31:e7:6b:b1:8b:49:9c:62:20:bd:33:90:03:83:40:55:75:a6:
         59:30:62:fd:d4:8f:38:a7:0b:3f:d3:ce:c5:52:99:db:5f:93:
         ae:46:ed:b4:5b:17:77:41:28:ba:7a:a0:f3:25:22:fe:7a:7c:
         82:15:63:c3:67:b8:db:79:02:cd:b0:9e:8c:d9:d7:b0:15:9d:
         35:7e:fc:50:f7:9b:a2:0c:6e:e9:9a:52:01:74:8f:7c:ff:14:
         3b:6d:7b:e4:96:ed:5b:17:c2:ba:93:e5:3a:0d:06:8e:6c:43:
         58:72:9e:f2:8e:a2:bb:66
```

Task 9: Options for intermediate CA certificate

The first thing here is the create of certificate for CA1 using the CSR with this command

```
openssl ca -config openssl.cnf -extensions v3_intermediate_ca
-days 3650 -notext -md sha256 -in ca1/csr/ca1.csr.pem -out
ca1/certs/ca1.cert.pem
```

and got the following output

```
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4098 (0x1002)
        Validity
            Not Before: May 14 17:24:30 2021 GMT
            Not After : May 12 17:24:30 2031 GMT
        Subject:
            countryName               = SE
            stateOrProvinceName       = Blekinge
            organizationName          = ET2540
            commonName                = yivi20CA1
        X509v3 extensions:
            X509v3 Subject Key Identifier:

A4:57:37:37:C9:8B:21:0B:E9:C0:25:88:2F:D6:0F:64:09:79:AE:79
            X509v3 Authority Key Identifier:

keyid:DB:63:33:FF:F0:6D:BB:1D:DC:90:38:04:47:0B:88:70:43:3B:8E:1
9

            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
```

```
Certificate is to be certified until May 12 17:24:30 2031 GMT
(3650 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

I then used `man ca` command to check the meaning of options and values they take in the above command

- `config` option: It allows an alternative configuration file to be specified and in this case it specifies `openssl.cnf` as the configuration file.
- `extension` option:the section of the configuration file containing certificate extensions to be added when a certificate is issued (defaults to x509_extensions unless the -extfile option is used). If no extension section is present then, a V1 certificate is created. If the extension section is present (even if it is empty), then a V3 certificate is created
- `days` option: The number of days to certify the certificate for.In this case,the certificate should be certified for 3650days
- `notext` option: This makes sure that we don't output the text form of a certificate to the output file
- `md` option:The message digest to use. Any digest supported by the OpenSSL dgst command can be used. In this case we use `sha256` as the message digest.
- `in` option: An input filename containing a single certificate request to be signed by the CA. In this case, the input file is `ca1/csr/ca1.csr.pem`
- `out` option:the output file to output certificates to. The default is standard output. In this case, the output file is `ca1/certs/ca1.cert.pem`

The effect of specifying the `v3_intermediate_ca` value for the -`extensions` option is that the certificate extensions that will be added when certificate is created will be from the `v3_intermediate_ca` section of the configuration file.

Task 10: Verify the certificate for CA1
The task is about verifying the certificate for CA1.To do that, I ran the following commands

```
openssl x509 -noout -text -in ca1/certs/ca1.cert.pem
openssl verify -CAfile certs/root.cert.pem
ca1/certs/ca1.cert.pem
```

The output of the commands is respectively as follows;

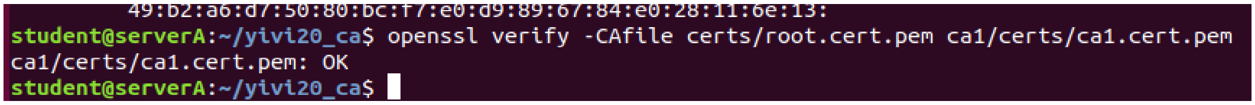
```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4098 (0x1002)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = SE, ST = Blekinge, L = Karlskrona, O =
ET2540, CN = yivi20Root
        Validity
            Not Before: May 14 17:24:30 2021 GMT
            Not After : May 12 17:24:30 2031 GMT
        Subject: C = SE, ST = Blekinge, O = ET2540, CN =
yivi20CA1
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:

00:bc:85:4f:59:b3:b7:d5:20:eb:de:2a:29:ef:6d:

d4:48:93:6d:ba:72:28:86:0f:04:e8:3c:9c:bc:0d:

e3:28:a7:66:7f:86:22:79:dd:70:85:8c:66:6f:21:

76:f9:6c:75:18:9f:06:ce:9d:16:cf:5b:ae:31:a3:

44:96:64:95:77:9f:59:93:6b:f6:e3:16:24:28:86:
```

```
8e:38:82:cf:de:b9:53:f1:04:fb:8e:21:2c:e0:02:

72:0b:7f:d2:44:25:78:a3:2e:63:dc:f4:ba:a8:56:

db:af:9d:c3:0f:06:69:80:cf:64:fe:c5:21:fe:25:

4a:06:a2:b1:b2:7d:3c:99:35:af:20:13:2a:8a:12:

d8:2e:e6:a4:7d:ab:b7:f3:b1:78:8c:79:86:9c:cc:

e1:bd:45:c2:f5:af:4a:e8:92:7e:47:40:a4:d5:7c:

0c:cf:44:ea:8a:30:49:db:06:eb:b2:79:c8:28:41:

8f:7e:db:4f:77:eb:4c:59:7c:64:01:5e:3f:04:7b:

cb:8f:1e:d2:0b:56:c2:92:3e:be:a3:67:fb:22:1d:

f9:13:c6:24:af:22:0d:4f:e1:90:63:33:f3:5e:4b:

32:fc:6c:54:c7:1e:fb:d3:d3:f1:dd:3f:59:f8:ab:

db:5c:47:d4:59:fe:11:40:ca:66:a1:f9:b0:f9:8b:

ee:fb:3c:16:ce:5e:3f:3c:49:82:84:b1:b3:48:85:

90:53:85:06:b5:75:e6:08:38:e1:97:8c:ca:d8:2c:

8b:23:e2:ff:80:d8:36:c4:d6:fe:17:63:1a:a4:43:

08:37:96:1e:04:17:5a:b7:3f:ec:13:ec:8c:55:a9:

bb:1e:71:08:e7:2e:68:f7:ac:f5:f5:3f:fe:d2:05:

d2:fe:b7:57:ab:ba:e4:b9:8d:bb:98:b8:03:1a:b2:
```

```
a3:70:e5:62:4a:e2:1d:4c:ad:b6:22:c6:ff:7d:32:

1b:bc:a9:78:8c:c3:4e:bc:43:fe:3e:73:44:e3:11:

73:ad:67:97:c6:5c:45:dd:61:7b:66:f1:4d:36:b6:

60:db:11:63:c6:01:9b:0b:1c:af:be:69:4c:ba:e0:

5d:61:70:89:43:8a:63:3c:03:e4:d2:94:71:6a:c7:

52:e9:99:63:ff:d8:85:bc:c3:f4:ec:50:37:7d:be:

55:37:d6:c4:6a:e3:c7:98:e2:eb:81:09:e6:ed:56:

74:e6:52:5c:25:a1:b0:0e:8f:57:1f:f4:9e:78:dd:

eb:ec:a1:e2:18:f3:2f:88:62:8e:8c:9e:40:cd:a8:

0b:a2:d8:8e:6b:97:84:cd:a5:0a:f6:4d:bd:54:6b:

cb:e9:0e:15:8e:aa:f4:3e:6e:4e:54:82:c3:4c:bd:
                    a1:90:d7
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:

A4:57:37:37:C9:8B:21:0B:E9:C0:25:88:2F:D6:0F:64:09:79:AE:79
            X509v3 Authority Key Identifier:

keyid:DB:63:33:FF:F0:6D:BB:1D:DC:90:38:04:47:0B:88:70:43:3B:8E:1
9

            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
            X509v3 Key Usage: critical
```

```
                Digital Signature, Certificate Sign, CRL Sign
    Signature Algorithm: sha256WithRSAEncryption
         9d:43:6c:58:87:76:bf:b9:dc:57:02:c5:a5:de:7f:3d:48:30:
         20:38:3b:c8:95:8d:85:11:a5:90:87:be:22:cc:1d:ae:17:6f:
         3a:08:0f:81:f3:64:b6:90:ef:3e:0e:d4:cc:94:d3:a3:c2:32:
         22:f6:94:54:c2:ae:59:ca:f0:bf:ec:22:43:27:5d:6d:78:04:
         bc:ba:de:c9:b0:25:0d:31:d9:c6:68:a3:44:ff:2f:34:1c:39:
         a5:5e:ad:41:33:a5:c2:d9:9d:5d:73:5c:dd:ab:fd:2e:5a:bb:
         f9:63:9f:42:6c:78:b0:6b:52:8c:b6:55:b0:b4:10:25:a9:fd:
         bc:2c:85:a4:03:85:db:c7:4e:2d:76:cd:a5:10:e4:9c:8f:0b:
         ca:27:f2:1d:1d:40:13:b0:17:10:4a:fa:d2:33:b7:eb:e9:43:
         6c:d1:b5:a7:ce:8c:27:21:8d:38:8e:57:f2:ba:d1:8a:af:4f:
         0f:3e:2c:73:5f:49:dc:f9:04:2c:de:b2:3a:cb:65:11:d8:59:
         5d:55:8a:7f:1d:c0:81:9e:60:c7:36:51:d2:69:83:b0:9e:44:
         93:f6:26:be:7a:25:e7:f2:ee:17:15:e3:86:64:05:1b:52:e6:
         b7:d9:0b:0e:f5:48:7b:e7:b5:95:0c:a4:1b:60:6c:ad:da:5f:
         f1:e2:fa:80:b4:f3:13:c5:89:12:9f:3c:68:70:7d:d4:73:3f:
         85:d5:33:de:4c:20:49:00:1a:dd:c1:2b:45:51:87:e7:ac:a8:
         6f:df:fc:3b:d2:67:52:69:70:89:19:6d:2a:d3:fb:ff:13:08:
         fe:26:9d:20:ab:36:d0:70:a8:91:f8:c8:57:f4:d7:0f:44:34:
         be:4a:d4:28:59:e5:36:ab:e5:ec:a2:bf:fb:28:c8:ae:f2:21:
         5a:42:9a:7b:bc:aa:81:d1:e4:b9:f3:9e:06:51:ce:81:08:e6:
         1f:28:a4:78:19:8b:48:54:67:7d:44:32:11:8c:f1:a2:df:82:
         7c:86:ca:a3:a0:b7:6d:0f:64:ca:2d:3d:66:ba:22:1e:2e:e5:
         d8:ba:bc:b1:5d:8b:34:22:d4:77:66:9a:59:69:e4:6f:2b:54:
         d8:00:72:ee:b9:a8:6a:6e:6c:15:a7:ed:33:f0:5a:58:a1:b7:
         c8:88:e7:9a:f1:3f:15:e9:ad:92:43:74:56:61:f9:51:e6:52:
         31:9c:2a:78:6c:74:35:a2:5c:72:dd:03:60:68:94:0e:c6:55:
         d3:72:d2:a5:aa:1d:59:07:92:69:a0:68:25:bb:38:b1:cf:40:
         25:ad:e4:13:b0:83:a8:6d:55:79:50:ed:17:1d:a3:d5:8a:9a:
         59:0b:b6:c5:39:42:25:f2
```

The verification results



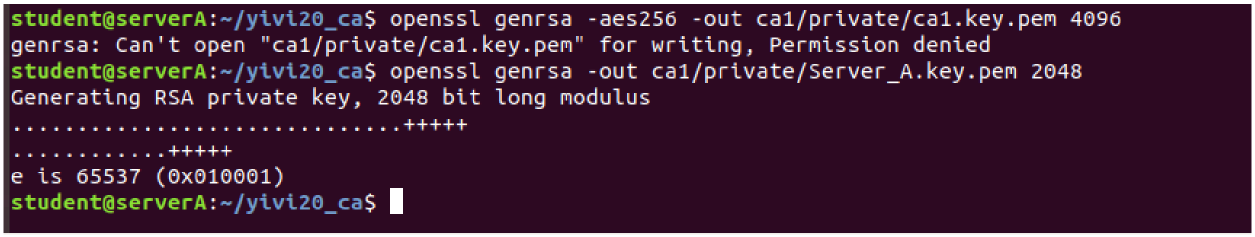
```
         49:b2:a6:d7:50:80:bc:f7:e0:d9:89:67:84:e0:28:11:6e:13:
student@serverA:~/yivi20_ca$ openssl verify -CAfile certs/root.cert.pem ca1/certs/ca1.cert.pem
ca1/certs/ca1.cert.pem: OK
student@serverA:~/yivi20_ca$ 
```

Task 11: Create server certificate

This task consisted of the following steps.

1. Creating an RSA private key for the server with

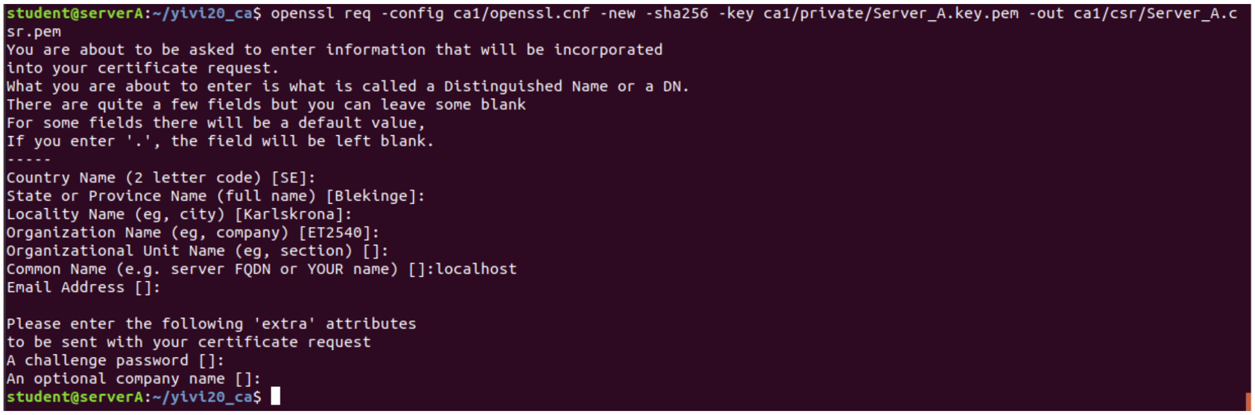
```
openssl genrsa -out ca1/private/Server_A.key.pem 2048
```



```
student@serverA:~/yivi20_ca$ openssl genrsa -aes256 -out ca1/private/ca1.key.pem 4096
genrsa: Can't open "ca1/private/ca1.key.pem" for writing, Permission denied
student@serverA:~/yivi20_ca$ openssl genrsa -out ca1/private/Server_A.key.pem 2048
Generating RSA private key, 2048 bit long modulus
..............................+++++
............+++++
e is 65537 (0x010001)
student@serverA:~/yivi20_ca$
```

2. Generating a CSR using the RSA private key from the previous step with

```
openssl req -config ca1/openssl.cnf -new -sha256 -key
ca1/private/Server_A.key.pem -out ca1/csr/Server_A.csr.pem
```



```
student@serverA:~/yivi20_ca$ openssl req -config ca1/openssl.cnf -new -sha256 -key ca1/private/Server_A.key.pem -out ca1/csr/Server_A.c
sr.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [SE]:
State or Province Name (full name) [Blekinge]:
Locality Name (eg, city) [Karlskrona]:
Organization Name (eg, company) [ET2540]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:localhost
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
student@serverA:~/yivi20_ca$
```

3. Using CA1’s private key to sign the CSR and create a certificate for your server

```
openssl ca -config ca1/openssl.cnf -extensions server_cert -days 375
-notext -md sha256 -in ca1/csr/Server_A.csr.pem -out
ca1/certs/Server_A.cert.pem
```

This command resulted to the following output

```
Using configuration from ca1/openssl.cnf
Enter pass phrase for
```

```
/home/student/yivi20_ca/ca1/private/ca1.key.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 8197 (0x2005)
        Validity
            Not Before: May 14 18:53:50 2021 GMT
            Not After : May 24 18:53:50 2022 GMT
        Subject:
            countryName               = SE
            stateOrProvinceName       = Blekinge
            localityName              = Karlskrona
            organizationName          = ET2540
            commonName                = localhost
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Key Identifier:

36:C2:3E:4F:A8:A9:E0:42:F0:A2:73:41:8D:12:B8:50:65:E9:B6:A9
            X509v3 Authority Key Identifier:

keyid:A4:57:37:37:C9:8B:21:0B:E9:C0:25:88:2F:D6:0F:64:09:79:AE:79

DirName:/C=SE/ST=Blekinge/L=Karlskrona/O=ET2540/CN=yivi20Root
                serial:10:02

            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
            X509v3 CRL Distribution Points:

                Full Name:
                  URI:https://localhost/ca1.crl.pem

Certificate is to be certified until May 24 18:53:50 2022 GMT (375
days)
Sign the certificate? [y/n]:y
```

```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Verification of the server certificate against the certificate chain.

The following command was used

```
 openssl x509 -noout -text -in ca1/certs/Server_A.cert.pem
```

It gave the following output.

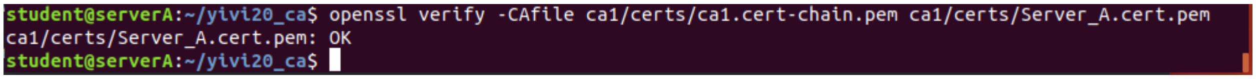
```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 8197 (0x2005)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = SE, ST = Blekinge, O = ET2540, CN = yivi20CA1
        Validity
            Not Before: May 14 18:53:50 2021 GMT
            Not After : May 24 18:53:50 2022 GMT
        Subject: C = SE, ST = Blekinge, L = Karlskrona, O = ET2540,
CN = localhost
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:bb:7f:7a:5d:b6:42:c2:e0:5d:81:5e:77:72:2e:
                    2c:a1:7b:e7:a9:d1:dd:c2:7c:57:2a:b3:79:00:01:
                    a8:0e:1a:f0:02:0c:5c:be:16:65:9a:05:51:b6:d2:
                    d2:b1:c6:55:a4:48:13:f1:a3:97:28:2f:2d:b0:5f:
                    7f:4a:fd:09:61:c6:bf:bf:6d:1b:82:af:15:12:f3:
                    46:69:16:e9:5d:3e:8b:0b:d0:ec:40:53:ff:b5:74:
                    b7:c7:2c:71:1d:04:85:a0:2d:2e:c3:ae:9c:65:ff:
                    ca:be:cf:6f:8d:b1:c0:0a:fb:32:85:4d:7c:b4:13:
                    e1:21:3f:ba:9c:1e:c7:76:85:59:93:ab:de:03:35:
                    dd:2a:c2:41:f1:01:82:e1:ae:05:01:f7:eb:1b:3c:
```

```
                    44:27:90:d2:1e:c0:fe:a3:56:20:34:fc:40:08:85:
                    5e:93:94:69:f9:f1:be:c4:69:d9:6e:c2:b2:26:cc:
                    7f:d1:a5:8d:64:f8:79:be:9b:60:55:11:3f:aa:86:
                    37:47:70:6d:f7:b6:5f:60:3f:79:f4:ea:b6:10:b8:
                    94:95:80:10:ca:56:90:4b:9b:70:2a:19:28:64:a5:
                    43:97:a7:15:d8:db:fe:2c:46:65:5a:70:7a:83:7f:
                    84:3e:9b:0c:15:61:2c:cd:26:60:41:43:75:82:6a:
                    10:ad
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Key Identifier:

36:C2:3E:4F:A8:A9:E0:42:F0:A2:73:41:8D:12:B8:50:65:E9:B6:A9
            X509v3 Authority Key Identifier:

keyid:A4:57:37:37:C9:8B:21:0B:E9:C0:25:88:2F:D6:0F:64:09:79:AE:79

DirName:/C=SE/ST=Blekinge/L=Karlskrona/O=ET2540/CN=yivi20Root
                serial:10:02

            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
            X509v3 CRL Distribution Points:

                Full Name:
                  URI:https://localhost/ca1.crl.pem

    Signature Algorithm: sha256WithRSAEncryption
         8e:b9:6e:d1:2d:69:00:98:29:bf:f8:f1:6a:36:de:4a:98:32:
         be:92:c5:c7:b6:82:1b:72:04:02:40:6e:ae:a8:ec:29:3c:bf:
         52:f7:aa:44:72:09:6f:7b:48:5e:3d:28:1a:68:bd:f7:91:96:
         1f:62:c7:e4:b7:f3:cd:4e:8c:b8:79:89:5e:25:3b:f4:75:64:
         cb:2d:d0:96:1c:ed:ef:3c:6d:44:48:26:fc:78:58:4b:a1:b5:
         c1:aa:77:0d:e9:9d:7e:f1:86:28:4c:2c:ab:98:1a:81:4e:b8:
         b7:0c:f0:00:d7:e7:be:10:02:95:5d:eb:ae:93:06:c7:d0:0b:
         27:fe:ac:1d:55:12:14:63:93:6c:b7:de:12:77:b1:c0:35:9e:
         30:d8:1b:42:35:4c:3a:62:9e:67:5d:21:db:d3:c0:4a:6b:18:
         1e:df:f0:80:35:8c:c2:13:07:05:e9:52:7b:9a:78:af:2b:cc:
         e2:a0:c5:95:48:a8:fe:08:28:5f:de:58:85:65:fb:e1:fa:8a:
         24:31:bd:c6:b9:f0:1d:4d:66:2d:5c:d8:dd:0d:68:22:b3:38:
         bd:50:b3:f1:30:8d:64:a5:7c:e5:c8:a3:cb:e1:69:48:13:0a:
         ae:ab:23:e7:19:5b:c5:be:71:c8:03:5f:14:3f:3d:70:1e:22:
         f5:3a:50:45:a7:b9:cf:4c:a9:f9:fa:b1:29:38:f9:a4:ad:84:
         a9:5e:85:08:34:7e:bb:34:00:1c:5e:a1:fb:2b:4d:b2:06:58:
         f0:69:77:3e:ac:fa:e6:72:5e:40:ef:54:8d:f4:2a:78:f8:ea:
         0f:2e:b7:d9:f7:95:27:16:6c:b7:e3:79:08:87:36:96:8a:4f:
         e3:5b:a1:7f:77:ce:1f:18:fa:a8:1e:aa:c8:6e:1f:ad:e6:d0:
         34:d2:ef:41:64:c4:e9:59:a8:b6:43:c3:c0:94:31:bd:cc:da:
         89:9a:a6:db:ae:be:80:6a:e1:d4:bf:7e:7d:96:2f:fa:14:f8:
         6f:de:c1:38:b8:60:95:37:dd:37:4c:bd:e0:b8:86:9a:5d:fb:
         47:d0:f0:8d:06:39:c2:27:53:12:50:5a:7e:94:92:8b:1b:f1:
         76:40:41:22:48:81:7d:51:c1:b6:d1:6c:fe:2b:b6:e5:ca:4f:
         ee:75:02:e6:ad:97:42:62:5f:64:ea:08:3c:6a:05:6a:3a:09:
         a7:0a:8d:82:33:16:28:8e:92:52:0a:fe:86:81:b4:c1:8b:40:
         58:17:1e:1e:14:fd:1d:96:19:c5:e2:85:07:b1:30:a6:d3:d6:
         31:6c:43:22:27:9c:8d:d9:a5:61:fc:5b:96:58:7d:14:a1:62:
         77:30:bd:55:3d:3e:3b:13
```

For verification, I ran this command

```
openssl verify -CAfile ca1/certs/ca1.cert-chain.pem
ca1/certs/Server_A.cert.pem
```

and this was the resulting output



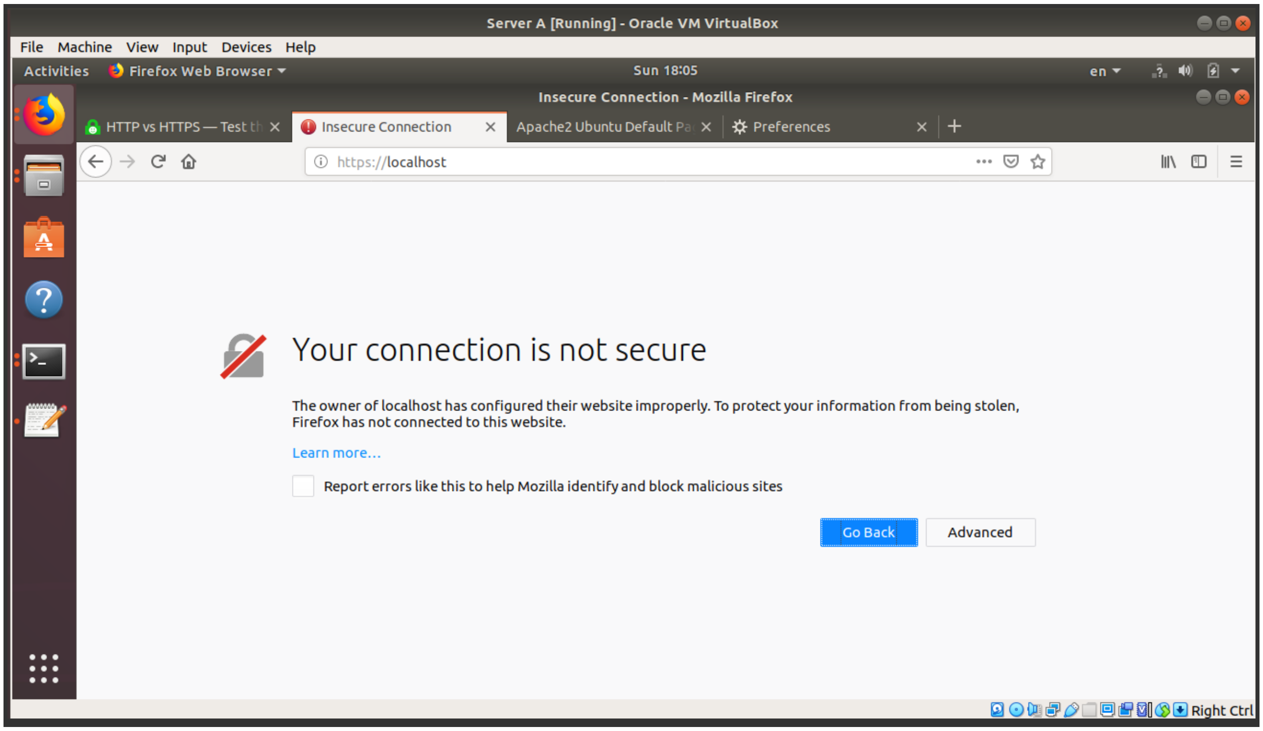
```
student@serverA:~/yivi20_ca$ openssl verify -CAfile ca1/certs/ca1.cert-chain.pem ca1/certs/Server_A.cert.pem
ca1/certs/Server_A.cert.pem: OK
student@serverA:~/yivi20_ca$
```

Task 12: Show your certificate in Firefox

To accomplish this task, I followed the instructions as in the lab manual coupled with the ones in this document https://jamielinux.com/docs/openssl-certificate-authority/sign-server-and-client-certificates.html but the process didn’t succeed.

After searching the web, I got this document https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-16-04 that helped me get specified results.

First output from running https://localhost


Server A [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox Web Browser
Sun 18:05
en
Insecure Connection - Mozilla Firefox
HTTP vs HTTPS — Test th
Insecure Connection
Apache2 Ubuntu Default Pa
Preferences
https://localhost
Right Ctrl

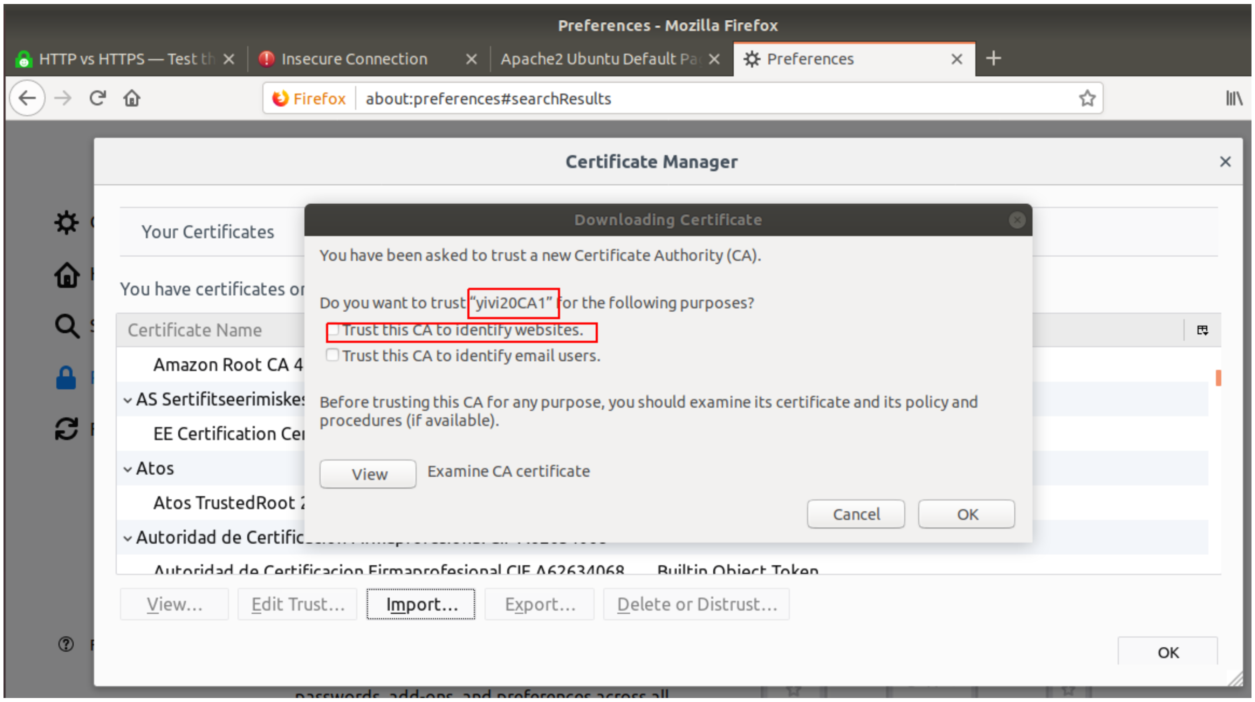
Your connection is not secure

The owner of localhost has configured their website improperly. To protect your information from being stolen, Firefox has not connected to this website.

Learn more…

□ Report errors like this to help Mozilla identify and block malicious sites

I then imported my certificate chain as seen here



You have been asked to trust a new Certificate Authority (CA).

Do you want to trust "yivi20CA1" for the following purposes?

□ Trust this CA to identify websites.

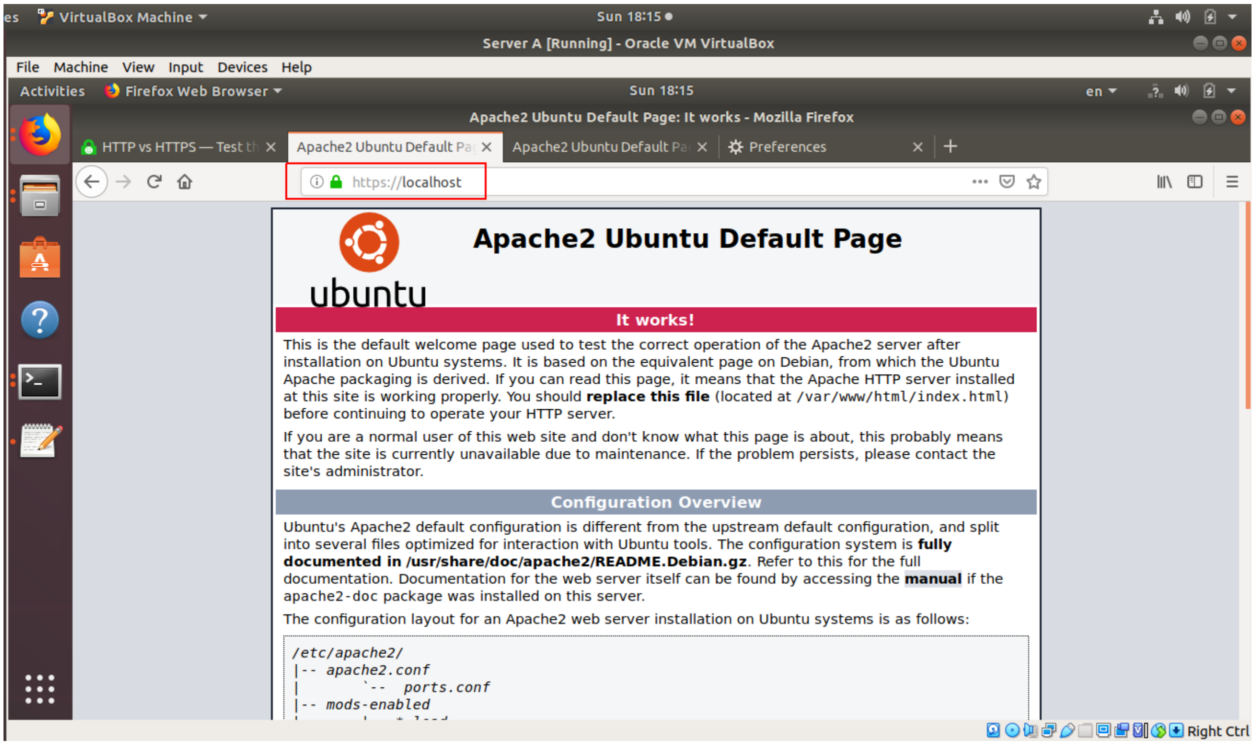
□ Trust this CA to identify email users.

Before trusting this CA for any purpose, you should examine its certificate and its policy and procedures (if available).

View Examine CA certificate

Cancel OK

Reloading the page https://localhost, I then got



VirtualBox Machine
Sun 18:15
Server A [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Firefox Web Browser
Sun 18:15
en
Apache2 Ubuntu Default Page: It works - Mozilla Firefox
HTTP vs HTTPS — Test th
Apache2 Ubuntu Default Pa
Apache2 Ubuntu Default Pa
Preferences
https://localhost
ubuntu
Right Ctrl

Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

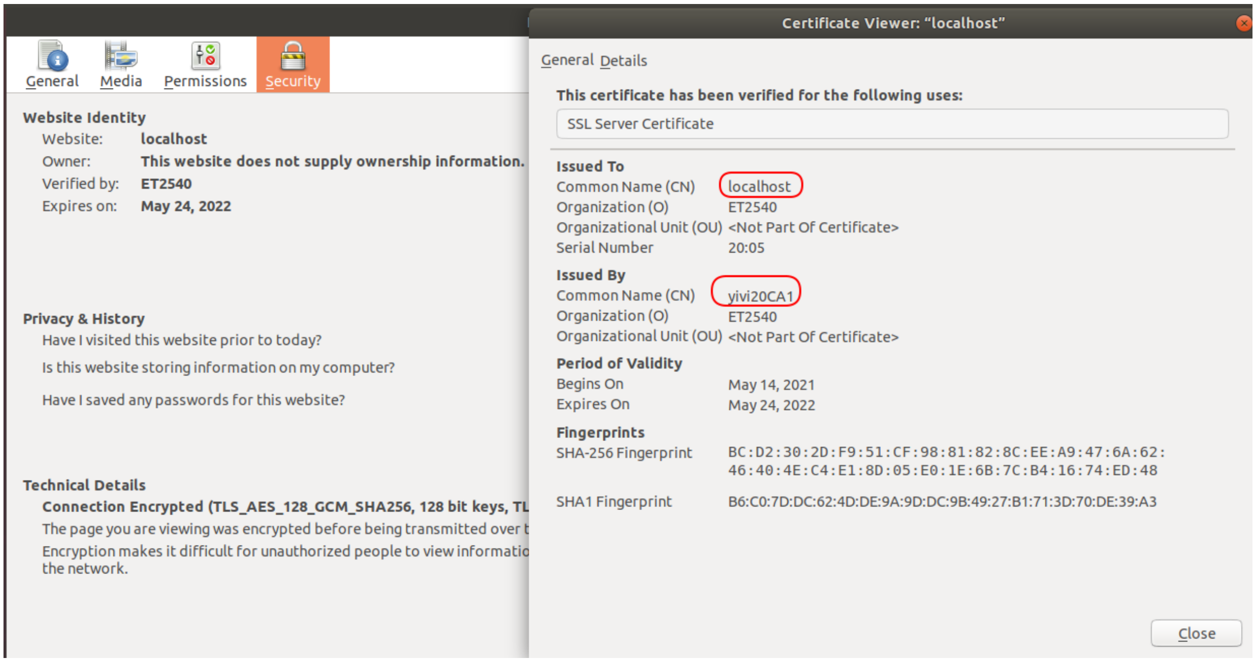
Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
```

The certificate viewer is as follows



Website Identity

Website: localhost

Owner: This website does not supply ownership information.

Verified by: ET2540

Expires on: May 24, 2022

Privacy & History

Have I visited this website prior to today?

Is this website storing information on my computer?

Have I saved any passwords for this website?

Technical Details

Connection Encrypted (TLS_AES_128_GCM_SHA256, 128 bit keys, TL

The page you are viewing was encrypted before being transmitted over t

Encryption makes it difficult for unauthorized people to view informatio the network.

Certificate Viewer: "localhost"

This certificate has been verified for the following uses:

SSL Server Certificate

Issued To

Common Name (CN) localhost

Organization (O) ET2540

Organizational Unit (OU) <Not Part Of Certificate>

Serial Number 20:05

Issued By

Common Name (CN) yivi20CA1

Organization (O) ET2540

Organizational Unit (OU) <Not Part Of Certificate>

Period of Validity

Begins On May 14, 2021

Expires On May 24, 2022

Fingerprints

SHA-256 Fingerprint BC:D2:30:2D:F9:51:CF:98:81:82:8C:EE:A9:47:6A:62:46:40:4E:C4:E1:8D:05:E0:1E:6B:7C:B4:16:74:ED:48

SHA1 Fingerprint B6:C0:7D:DC:62:4D:DE:9A:9D:DC:9B:49:27:B1:71:3D:70:DE:39:A3

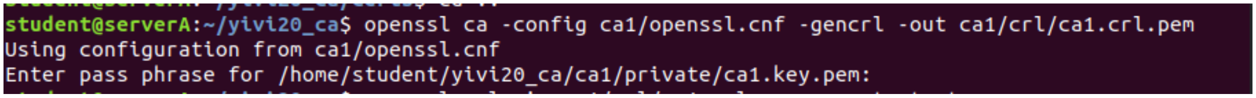
Task 13: Create a CRL for CA1

For this task, I added

```
crlDistributionPoints = URI:https://localhost/ca1.crl.pem
```

to the server_cert section of `ca1/openssl.conf` file.
Secondly, I ran these command

```
openssl ca -config ca1/openssl.cnf -gencrl -out
ca1/crl/ca1.crl.pem
```



```
student@serverA:~/yivi20_ca$ openssl ca -config ca1/openssl.cnf -gencrl -out ca1/crl/ca1.crl.pem
Using configuration from ca1/openssl.cnf
Enter pass phrase for /home/student/yivi20_ca/ca1/private/ca1.key.pem:
```

to create a CRL for CA1 and lastly this command

```
openssl crl -in ca1/crl/ca1.crl.pem -noout -text
```

The output of the last command is as follows

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=SE, ST=Blekinge, O=ET2540, CN=yivi20CA1
        Last Update: May 15 06:14:41 2021 GMT
        Next Update: Jun 14 06:14:41 2021 GMT
        CRL extensions:
            X509v3 CRL Number:
                8195
Revoked Certificates:
    Serial Number: 2001
        Revocation Date: Dec 15 14:58:52 2020 GMT
    Signature Algorithm: sha256WithRSAEncryption
         75:59:ce:15:4f:5e:d1:f0:67:76:fd:73:8e:ef:4c:04:85:91:
         69:72:ca:ba:ec:4c:07:12:50:55:08:ab:02:42:76:d9:69:3c:
         63:fd:3d:19:df:1f:fb:76:c6:13:e0:49:15:f6:3b:71:29:ab:
         80:ca:e0:9f:a7:87:fe:0b:04:bc:35:ab:fa:d0:ca:cf:0f:72:
         66:79:ad:e4:50:ec:80:6a:b3:53:87:42:e8:65:3c:df:7c:c2:
         9a:36:43:78:5a:43:49:ce:42:36:ea:41:51:3c:6c:f7:3e:7f:
         f0:8b:cc:e1:7f:00:ad:0d:db:b7:38:ec:df:b3:20:4b:a1:fb:
         28:fc:02:53:39:36:d7:ad:d6:c6:4c:8c:72:14:e3:0c:56:de:
         c7:68:ac:8d:bc:a9:4d:f3:96:e5:fc:52:a6:1a:ca:50:38:74:
         68:68:8e:22:47:b2:bb:66:23:ff:ff:19:92:71:a0:23:9e:2a:
         0c:a9:0e:cb:ea:73:2b:1a:af:90:ea:47:2b:25:ed:b9:d0:52:
         39:13:78:c0:c1:61:1d:21:02:58:d1:04:12:60:f4:23:d8:5d:
         b8:60:f5:0b:ad:d7:23:76:fa:ba:af:47:cd:ff:e9:9b:d5:f5:
         16:09:90:08:84:2a:e6:72:63:13:0e:d3:68:26:75:47:47:9e:
         bf:44:86:94:55:f8:5b:af:2b:94:51:55:81:96:2e:c1:5a:f6:
         31:b0:d2:61:28:c8:94:f1:51:52:f6:42:4b:a4:25:6f:bb:67:
         67:de:87:1f:ac:11:e0:a1:0d:a6:69:60:2d:e9:92:dd:04:4e:
         f4:6c:94:ae:10:c4:e1:20:19:dd:f7:01:3e:25:0d:1b:dc:67:
         a4:2c:60:bf:3d:e2:4f:78:a6:98:12:1a:44:f3:2f:e5:c6:50:
         57:34:6f:c0:4d:f1:8f:77:fc:b8:e0:70:17:48:ba:60:2f:f8:
         41:43:ce:f5:3b:16:b3:62:1f:06:00:10:a8:a3:c3:56:0b:3e:
         43:5c:91:b4:bf:e2:ad:36:99:0d:f9:77:d6:ed:62:e0:61:4e:
         ea:8d:0d:82:c2:a4:e4:60:64:3a:f4:9e:22:69:45:80:f5:4b:
         f3:0a:0d:35:6e:f6:d9:ef:a1:c0:2d:a5:f0:4e:2a:d6:7b:7a:
         32:36:92:f3:4a:b1:4d:ef:ca:5c:38:8a:52:a3:37:c9:7c:cb:
```

```
        dc:43:c7:3d:9b:19:37:15:6f:cb:6d:5b:6d:8f:3f:b6:09:f1:
        2e:96:97:9b:8e:e9:28:3d:3c:c3:f4:bf:bc:a6:af:1e:34:29:
        61:f7:b9:61:58:9b:7a:35:0d:5b:07:3e:00:5c:d9:87:5a:b9:
        8a:30:b2:6f:51:7d:ab:3d
```

Task 14: Revoke a certificate

For this task, I created a user certificate with the following commands

```
1. openssl genrsa -out private/dragos.ilie@bth.se.key.pem
   2048
2. openssl req -config openssl.cnf -key
   ca1/private/dragos.ilie@bth.se.key.pem -new -sha256 -out
   csr/dragos.ilie@bth.se.csr.pem
3. openssl ca -config ca1/openssl.cnf -extensions usr_cert
   -days 375 -notext -md sha256 -in
   csr/dragos.ilie@bth.se.csr.pem -out
   ca1/certs/dragos.ilie@bth.se.cert.pem
```

Results after commands

```
Using configuration from ca1/openssl.cnf
Enter pass phrase for
/home/student/yivi20_ca/ca1/private/ca1.key.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 8198 (0x2006)
        Validity
            Not Before: May 15 13:28:07 2021 GMT
            Not After : May 25 13:28:07 2022 GMT
        Subject:
            countryName               = SE
            stateOrProvinceName       = Blekinge
            localityName              = Karlskrona
            organizationName          = ET2540
            commonName                = dragos.ilie@bth.se
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage: critical
                Digital Signature, Non Repudiation, Key
Encipherment
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:

04:4A:E2:C3:2C:7C:06:14:93:B6:4D:AC:D8:4D:22:66:CF:6D:EC:2D
            X509v3 Authority Key Identifier:

keyid:A4:57:37:37:C9:8B:21:0B:E9:C0:25:88:2F:D6:0F:64:09:79:AE:7
9

            X509v3 Extended Key Usage:
                TLS Web Client Authentication, E-mail Protection
Certificate is to be certified until May 25 13:28:07 2022 GMT
(375 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

For verification, I ran this command

```
openssl x509 -noout -text -in
ca1/certs/dragos.ilie@bth.se.cert.pem
```

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 8198 (0x2006)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = SE, ST = Blekinge, O = ET2540, CN =
yivi20CA1
        Validity
            Not Before: May 15 13:28:07 2021 GMT
            Not After : May 25 13:28:07 2022 GMT
        Subject: C = SE, ST = Blekinge, L = Karlskrona, O =
ET2540, CN = "dragos.ilie@bth.se "
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:

00:cd:ca:15:2b:f8:89:d9:ad:f5:0f:14:28:5c:29:

a0:a0:58:de:7a:76:5b:6b:c3:ce:61:b6:4b:2c:3d:

eb:f0:78:53:44:d1:67:53:64:f4:ff:c7:e9:55:ba:

b2:1a:e4:07:18:50:6b:74:a7:6f:41:b3:98:a9:ba:

a7:31:dc:80:aa:2c:3d:34:f6:ea:4e:8a:f6:d9:92:

dd:26:91:b7:29:50:ed:0f:a5:a9:29:a5:7c:96:e6:

a4:04:01:f6:0a:2a:3b:28:88:04:4b:72:34:91:d3:

fd:62:4e:5d:fa:ce:7b:3b:67:fc:65:f6:36:02:ee:

30:4b:22:45:4b:7c:dd:cf:57:49:85:a2:f1:20:fc:

bb:69:12:2d:36:f8:c3:c6:0a:0e:8f:af:b8:56:33:
```

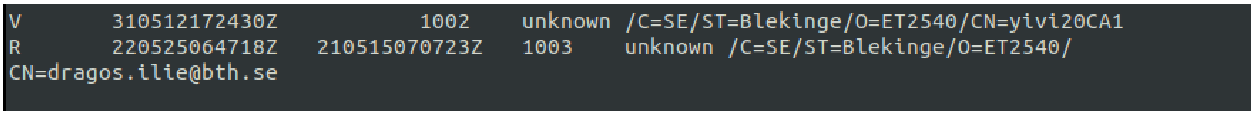
```
59:f9:0d:3d:48:89:65:89:2e:ba:2c:5a:65:e8:e6:

3a:10:e2:bd:bb:61:62:e0:c2:28:e1:8c:b6:05:da:

7b:73:59:32:6f:14:e0:a4:5e:ba:54:0d:55:98:5e:

23:61:67:b4:ad:f0:2e:0a:0e:61:bf:4f:71:d8:8c:

25:3d:db:df:b8:45:72:99:e9:87:d3:d4:f2:6f:bf:

5a:93:ef:c4:fc:97:49:ec:6a:56:90:41:28:04:31:

15:2d:b2:ed:84:e3:15:37:c2:c7:a0:40:33:16:97:
                    79:0f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage: critical
                Digital Signature, Non Repudiation, Key
Encipherment
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:

04:4A:E2:C3:2C:7C:06:14:93:B6:4D:AC:D8:4D:22:66:CF:6D:EC:2D
            X509v3 Authority Key Identifier:

keyid:A4:57:37:37:C9:8B:21:0B:E9:C0:25:88:2F:D6:0F:64:09:79:AE:7
9

            X509v3 Extended Key Usage:
                TLS Web Client Authentication, E-mail Protection
    Signature Algorithm: sha256WithRSAEncryption
         6f:fa:bf:63:ec:fb:61:39:c3:13:37:b9:a0:7e:ba:69:6b:82:
```

```
         88:1d:55:77:0f:ad:47:de:33:35:e2:ea:e5:73:d1:b2:c1:c8:
         bf:f4:0b:2b:d6:0d:82:dc:64:6a:50:b0:0b:02:5f:f3:15:90:
         58:ab:28:66:27:ef:97:b9:73:66:dc:af:6e:c7:b5:9c:52:33:
         49:73:fe:1f:05:eb:43:3f:01:95:74:13:1f:c8:71:a6:a8:d4:
         26:d4:51:94:4e:47:b1:e7:ce:c5:0b:26:20:73:28:e5:16:17:
         77:ef:91:01:df:e3:67:53:c4:76:9d:34:32:ed:84:63:84:11:
         f1:77:d8:e6:0e:ad:89:c4:e7:60:92:a1:d7:93:09:65:89:bf:
         47:46:b4:b6:4e:01:2f:07:80:70:f5:cd:1c:c3:8b:ba:89:e5:
         dd:a0:1c:63:1d:b8:6d:74:27:39:01:01:86:e7:ad:e0:20:40:
         4d:6a:78:d1:b6:a9:3b:77:ff:91:b5:36:da:49:00:3f:0f:ec:
         57:8c:2a:94:bb:db:4c:71:f3:4f:7e:78:70:7c:c1:44:9c:f3:
         80:63:df:91:13:47:05:54:2a:14:35:ef:52:a6:c7:3b:99:8f:
         87:b3:3a:d3:52:44:be:75:0d:e6:a3:10:4e:7a:b4:d9:34:72:
         9b:0d:b4:af:53:d3:62:db:63:45:72:b4:09:62:5d:f8:cb:ef:
         08:ae:dc:d1:0d:81:25:f3:76:b6:dc:2d:be:03:f0:2c:92:7c:
         6d:ab:be:1f:36:01:92:82:61:df:91:9a:77:e2:d6:1b:3e:be:
         aa:7f:a1:86:4f:64:ee:d1:65:42:eb:7b:26:e6:22:01:af:dc:
         15:d3:58:ac:d5:0a:b0:1a:e9:5e:21:7f:bd:fc:f1:9c:05:7e:
         dc:15:60:50:5a:32:cf:d2:bd:3e:8d:a0:3a:24:82:c9:a3:6a:
         15:bf:33:8f:c0:3d:d3:2f:3d:d0:76:96:d9:52:3e:f4:57:2c:
         a6:18:79:ef:03:89:ae:94:3f:90:b8:8f:74:18:06:d4:9a:d9:
         fa:d2:82:d2:f4:46:83:54:ef:bf:b0:44:f8:e0:92:8a:65:8b:
         57:22:75:00:cd:2d:9c:c5:6c:7a:9e:f9:a5:ff:79:01:05:f6:
         65:f3:2b:65:ff:3f:2d:8c:8b:e2:1c:ac:02:24:af:78:59:90:
         98:4e:4d:38:b8:22:8b:c6:44:73:55:cc:c9:a1:88:00:18:1a:
         15:ff:87:e3:0c:64:70:d2:eb:27:00:07:ff:20:a0:3d:ad:82:
         d7:fa:32:be:5b:b3:1e:0e:64:ff:b3:2a:cb:13:d4:e8:62:18:
         20:cd:14:5d:68:e9:3d:af
```

The newly created certificate is revoke by entering the following command

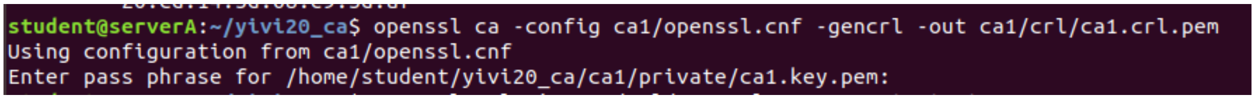
```
openssl ca -config openssl.cnf -revoke
ca1/certs/dragos.ilie@bth.se.cert.pem
```

Contents of `home/yivi20_ca/ca1/index.txt` are as shown below.



```
V       310512172430Z           1002    unknown /C=SE/ST=Blekinge/O=ET2540/CN=yivi20CA1
R       220525064718Z   210515070723Z   1003    unknown /C=SE/ST=Blekinge/O=ET2540/
CN=dragos.ilie@bth.se
```

Recreating the CRL



```
student@serverA:~/yivi20_ca$ openssl ca -config ca1/openssl.cnf -gencrl -out ca1/crl/ca1.crl.pem
Using configuration from ca1/openssl.cnf
Enter pass phrase for /home/student/yivi20_ca/ca1/private/ca1.key.pem:
```

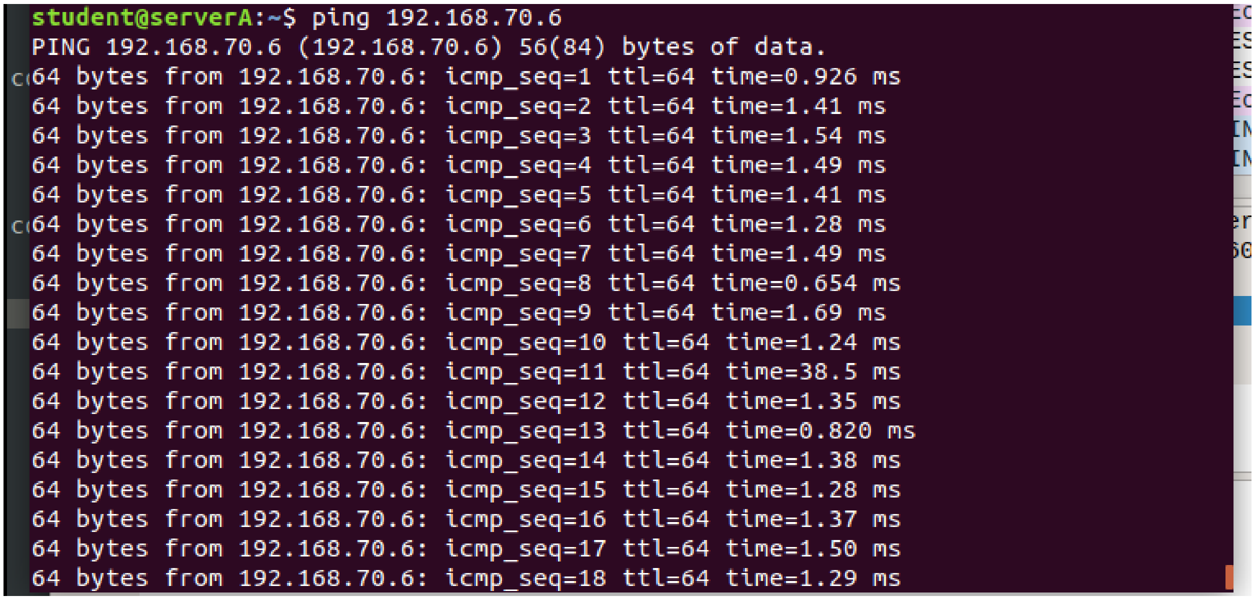
Out by running `openssl crl -in ca1/crl/ca1.crl.pem -noout -text` gives

```
Certificate Revocation List (CRL):
        Version 2 (0x1)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=SE, ST=Blekinge, O=ET2540, CN=yivi20CA1
        Last Update: May 15 13:39:51 2021 GMT
        Next Update: Jun 14 13:39:51 2021 GMT
        CRL extensions:
            X509v3 CRL Number:
                8196
Revoked Certificates:
    Serial Number: 2001
        Revocation Date: Dec 15 14:58:52 2020 GMT
    Signature Algorithm: sha256WithRSAEncryption
         b2:44:24:eb:40:d9:50:e3:6b:16:80:14:2f:b8:ec:d3:c0:08:
         68:9e:51:44:27:cf:0f:a5:d2:f4:27:3f:96:6e:61:a5:c9:cc:
         11:89:95:99:27:d2:ec:84:d9:b0:b8:23:da:96:a6:df:ec:7d:
         75:83:02:a8:10:99:f7:5a:80:8b:f7:f0:fe:2e:5c:f1:e8:97:
         e9:64:d9:14:7c:3b:41:84:7b:25:90:8a:9f:e8:93:f9:72:ad:
         1e:73:07:ba:93:d8:fb:c7:19:e8:94:15:f6:ae:06:48:2c:9a:
         83:6d:e1:6a:41:a5:d9:3c:5d:ab:7f:1a:14:ea:e5:64:e2:47:
         a9:70:6a:f7:a8:b8:7b:a4:03:42:b2:ff:b2:33:d7:32:99:c5:
         a5:76:66:e7:61:9a:f9:c3:9e:a1:bc:39:17:35:86:e3:6a:b0:
         bd:3c:1a:63:ab:77:05:45:27:e1:e5:e2:70:fc:dc:d0:74:16:
         59:e0:43:c1:3f:44:06:c9:4c:3f:2e:69:55:fb:79:72:4f:87:
         69:76:be:fc:b6:5b:66:eb:f7:42:1b:8f:5a:8e:25:65:58:77:
         09:0c:33:a4:e8:4a:bd:90:19:cb:46:5c:1c:82:8a:70:91:82:
```

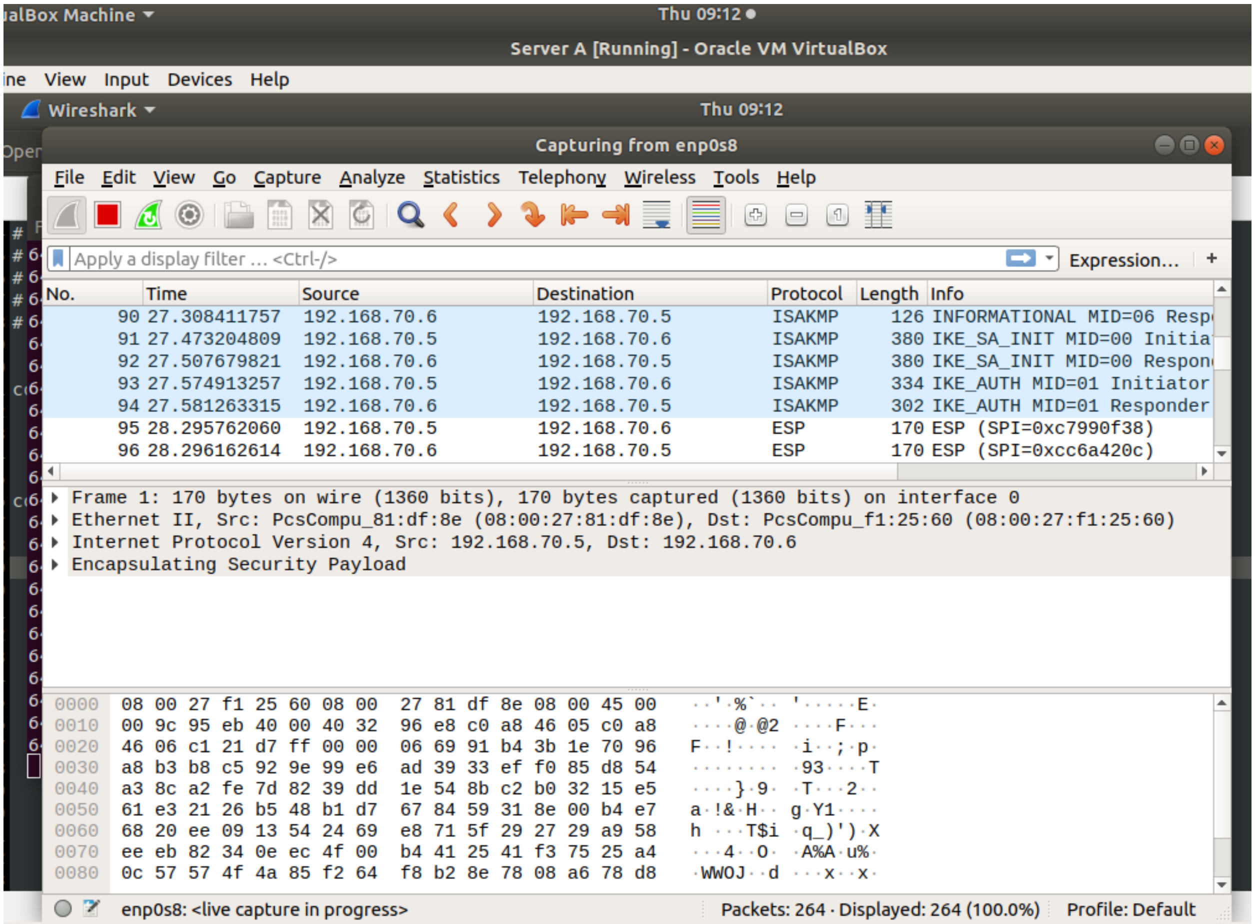
```
        ef:60:2e:8b:e8:16:da:bb:09:25:c0:b7:ce:5d:df:23:30:5c:
        91:19:11:c1:6e:73:dc:e0:4b:dc:fb:db:52:b4:be:65:13:88:
        bc:54:43:30:ca:4f:c5:6c:ed:ac:18:ac:8a:73:20:b7:00:c0:
        ae:04:b8:dc:65:cd:38:f6:92:84:56:2f:cd:a1:fc:3c:31:da:
        5d:41:b2:87:0f:4f:bf:cb:58:0a:45:6b:74:a4:f7:c9:7b:d4:
        c8:03:71:ba:af:4d:81:f4:7f:f4:41:a7:84:1b:1b:07:1b:d8:
        28:04:d1:e8:c5:fb:9d:3a:77:dc:6b:ad:53:ae:a8:47:25:56:
        d0:eb:0a:18:40:62:da:1c:e8:1e:37:8e:7a:8c:6b:86:1b:61:
        2c:01:d9:3c:f9:e7:e4:64:bf:52:42:86:93:31:29:f0:60:39:
        84:ea:5f:b5:49:2d:57:b9:0c:3b:76:df:74:15:0e:a6:4a:52:
        77:ae:ad:0e:c8:5c:9c:cf:f0:c8:dd:c5:a8:c2:cc:d4:8d:d0:
        a9:a6:3d:cf:74:af:d7:e7:76:aa:14:3e:06:c2:d1:2d:03:57:
        40:2b:d2:8f:96:57:57:52:4a:05:30:40:b8:f8:7a:f0:da:8c:
        c4:53:71:93:68:9e:8f:1c:dd:08:70:66:52:f3:f2:44:16:68:
        ce:87:49:03:f4:44:9f:31:b0:0b:05:f3:9a:15:3d:c0:b7:4d:
        99:6c:3e:1b:55:cd:57:e7
```

Task 15: Host-to-host transport mode VPN with PSK authentication

For this task, a host to host transport mode between server A and server B. This gave the following results;



```
student@serverA:~$ ping 192.168.70.6
PING 192.168.70.6 (192.168.70.6) 56(84) bytes of data.
64 bytes from 192.168.70.6: icmp_seq=1 ttl=64 time=0.926 ms
64 bytes from 192.168.70.6: icmp_seq=2 ttl=64 time=1.41 ms
64 bytes from 192.168.70.6: icmp_seq=3 ttl=64 time=1.54 ms
64 bytes from 192.168.70.6: icmp_seq=4 ttl=64 time=1.49 ms
64 bytes from 192.168.70.6: icmp_seq=5 ttl=64 time=1.41 ms
64 bytes from 192.168.70.6: icmp_seq=6 ttl=64 time=1.28 ms
64 bytes from 192.168.70.6: icmp_seq=7 ttl=64 time=1.49 ms
64 bytes from 192.168.70.6: icmp_seq=8 ttl=64 time=0.654 ms
64 bytes from 192.168.70.6: icmp_seq=9 ttl=64 time=1.69 ms
64 bytes from 192.168.70.6: icmp_seq=10 ttl=64 time=1.24 ms
64 bytes from 192.168.70.6: icmp_seq=11 ttl=64 time=38.5 ms
64 bytes from 192.168.70.6: icmp_seq=12 ttl=64 time=1.35 ms
64 bytes from 192.168.70.6: icmp_seq=13 ttl=64 time=0.820 ms
64 bytes from 192.168.70.6: icmp_seq=14 ttl=64 time=1.38 ms
64 bytes from 192.168.70.6: icmp_seq=15 ttl=64 time=1.28 ms
64 bytes from 192.168.70.6: icmp_seq=16 ttl=64 time=1.37 ms
64 bytes from 192.168.70.6: icmp_seq=17 ttl=64 time=1.50 ms
64 bytes from 192.168.70.6: icmp_seq=18 ttl=64 time=1.29 ms
```



Thu 09:12
Server A [Running] - Oracle VM VirtualBox
View Input Devices Help
Wireshark
Thu 09:12
Capturing from enp0s8
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…

```
No.   Time          Source         Destination    Protocol  Length Info
   90 27.308411757  192.168.70.6   192.168.70.5   ISAKMP       126 INFORMATIONAL MID=06 Resp
   91 27.473204809  192.168.70.5   192.168.70.6   ISAKMP       380 IKE_SA_INIT MID=00 Initia
   92 27.507679821  192.168.70.6   192.168.70.5   ISAKMP       380 IKE_SA_INIT MID=00 Respon
   93 27.574913257  192.168.70.5   192.168.70.6   ISAKMP       334 IKE_AUTH MID=01 Initiator
   94 27.581263315  192.168.70.6   192.168.70.5   ISAKMP       302 IKE_AUTH MID=01 Responder
   95 28.295762060  192.168.70.5   192.168.70.6   ESP          170 ESP (SPI=0xc7990f38)
   96 28.296162614  192.168.70.6   192.168.70.5   ESP          170 ESP (SPI=0xcc6a420c)
```

```
▸ Frame 1: 170 bytes on wire (1360 bits), 170 bytes captured (1360 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_81:df:8e (08:00:27:81:df:8e), Dst: PcsCompu_f1:25:60 (08:00:27:f1:25:60)
▸ Internet Protocol Version 4, Src: 192.168.70.5, Dst: 192.168.70.6
▸ Encapsulating Security Payload
```

```
0000  08 00 27 f1 25 60 08 00  27 81 df 8e 08 00 45 00   ··'·%`·· '·····E·
0010  00 9c 95 eb 40 00 40 32  96 e8 c0 a8 46 05 c0 a8   ····@·@2 ····F···
0020  46 06 c1 21 d7 ff 00 00  06 69 91 b4 3b 1e 70 96   F··!···· ·i··;·p·
0030  a8 b3 b8 c5 92 9e 99 e6  ad 39 33 ef f0 85 d8 54   ········ ·93····T
0040  a3 8c a2 fe 7d 82 39 dd  1e 54 8b c2 b0 32 15 e5   ····}·9· ·T···2··
0050  61 e3 21 26 b5 48 b1 d7  67 84 59 31 8e 00 b4 e7   a·!&·H·· g·Y1····
0060  68 20 ee 09 13 54 24 69  e8 71 5f 29 27 29 a9 58   h ···T$i ·q_)')·X
0070  ee eb 82 34 0e ec 4f 00  b4 41 25 41 f3 75 25 a4   ···4··O· ·A%A·u%·
0080  0c 57 57 4f 4a 85 f2 64  f8 b2 8e 78 08 a6 78 d8   ·WWOJ··d ···x··x·
```

enp0s8: <live capture in progress> Packets: 264 · Displayed: 264 (100.0%) Profile: Default

The configuration files of ipsec.conf and ipsec.secrets are as follows;

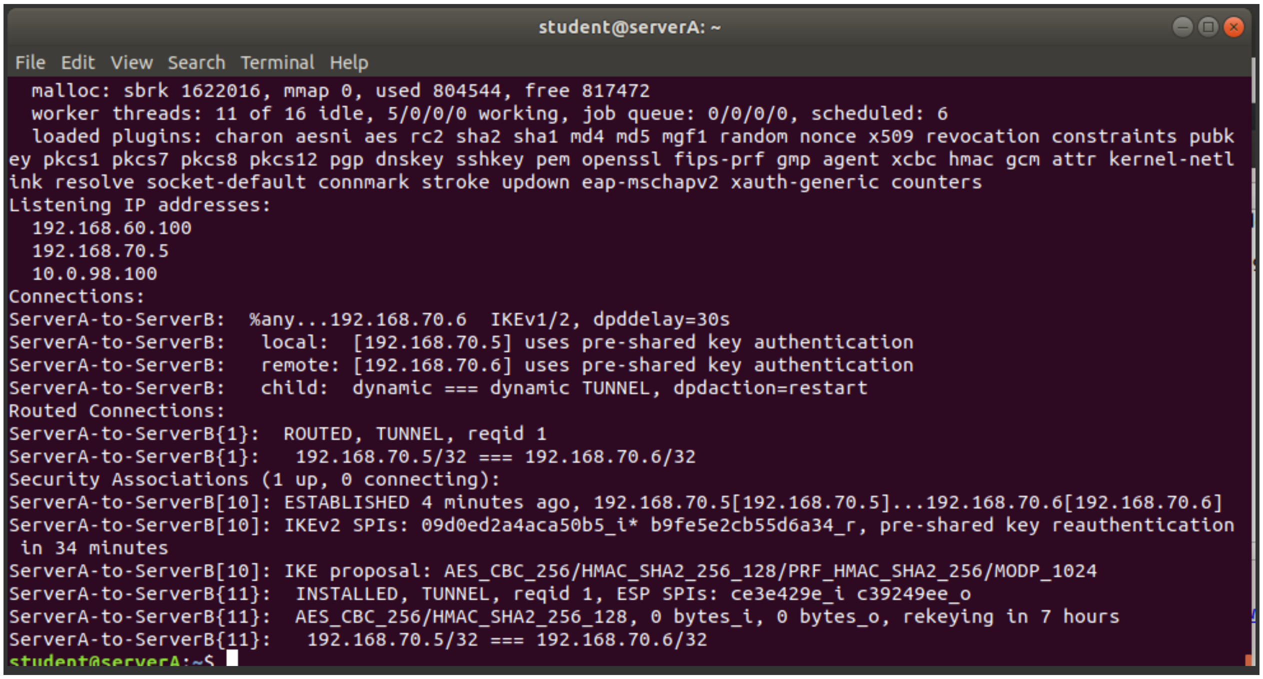
```
config setup
        #charondebug="ike 1, knl 1, cfg 0"
        charondebug="all"
        uniqueids=yes
        strictcrlpolicy=no
conn ServerA-to-ServerB
        #type=tunnel
        authby=secret
        #keyexchange=ikev2
        left=%defaultroute
        leftid=192.168.70.5
        #leftsubnet=10.10.27.1/24
        right=192.168.70.6
        #rightsubnet=10.9.141.1/24
        ike=aes256-sha2_256-modp1024!
        esp=aes256-sha2_256!
        keyingtries=0
        ikelifetime=1h
        lifetime=8h
        dpddelay=30
        dpdtimeout=120
        dpdaction=restart
        auto=route
```

ipsec.secrets

```
192.168.70.5 192.168.70.6 : PSK "pBavFPMBSCyq7g=="
```

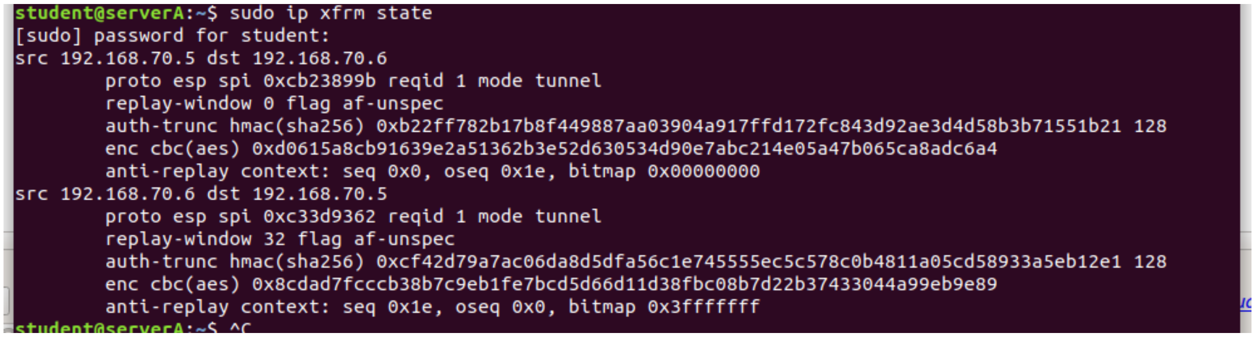
Task 16: Decrypt traffic with Wireshark

The output of `ipsec statusall`



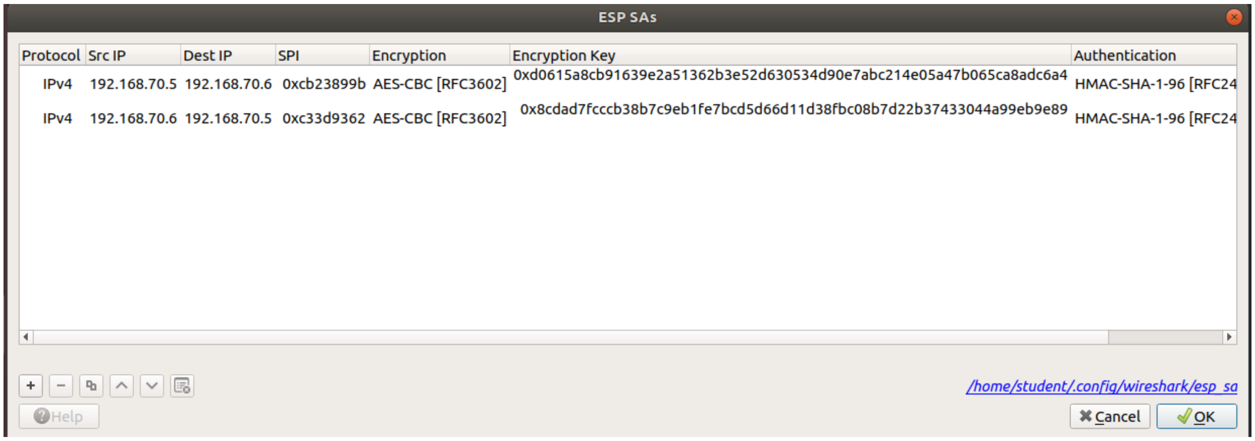
```
  malloc: sbrk 1622016, mmap 0, used 804544, free 817472
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 6
  loaded plugins: charon aesni aes rc2 sha2 sha1 md4 md5 mgf1 random nonce x509 revocation constraints pubk
ey pkcs1 pkcs7 pkcs8 pkcs12 pgp dnskey sshkey pem openssl fips-prf gmp agent xcbc hmac gcm attr kernel-netl
ink resolve socket-default connmark stroke updown eap-mschapv2 xauth-generic counters
Listening IP addresses:
  192.168.60.100
  192.168.70.5
  10.0.98.100
Connections:
ServerA-to-ServerB:  %any...192.168.70.6  IKEv1/2, dpddelay=30s
ServerA-to-ServerB:   local:  [192.168.70.5] uses pre-shared key authentication
ServerA-to-ServerB:   remote: [192.168.70.6] uses pre-shared key authentication
ServerA-to-ServerB:   child:  dynamic === dynamic TUNNEL, dpdaction=restart
Routed Connections:
ServerA-to-ServerB{1}:  ROUTED, TUNNEL, reqid 1
ServerA-to-ServerB{1}:   192.168.70.5/32 === 192.168.70.6/32
Security Associations (1 up, 0 connecting):
ServerA-to-ServerB[10]: ESTABLISHED 4 minutes ago, 192.168.70.5[192.168.70.5]...192.168.70.6[192.168.70.6]
ServerA-to-ServerB[10]: IKEv2 SPIs: 09d0ed2a4aca50b5_i* b9fe5e2cb55d6a34_r, pre-shared key reauthentication
 in 34 minutes
ServerA-to-ServerB[10]: IKE proposal: AES_CBC_256/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_1024
ServerA-to-ServerB{11}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: ce3e429e_i c39249ee_o
ServerA-to-ServerB{11}:  AES_CBC_256/HMAC_SHA2_256_128, 0 bytes_i, 0 bytes_o, rekeying in 7 hours
ServerA-to-ServerB{11}:   192.168.70.5/32 === 192.168.70.6/32
student@serverA:~$
```

The output `sudo ip xfrm state` was as follows



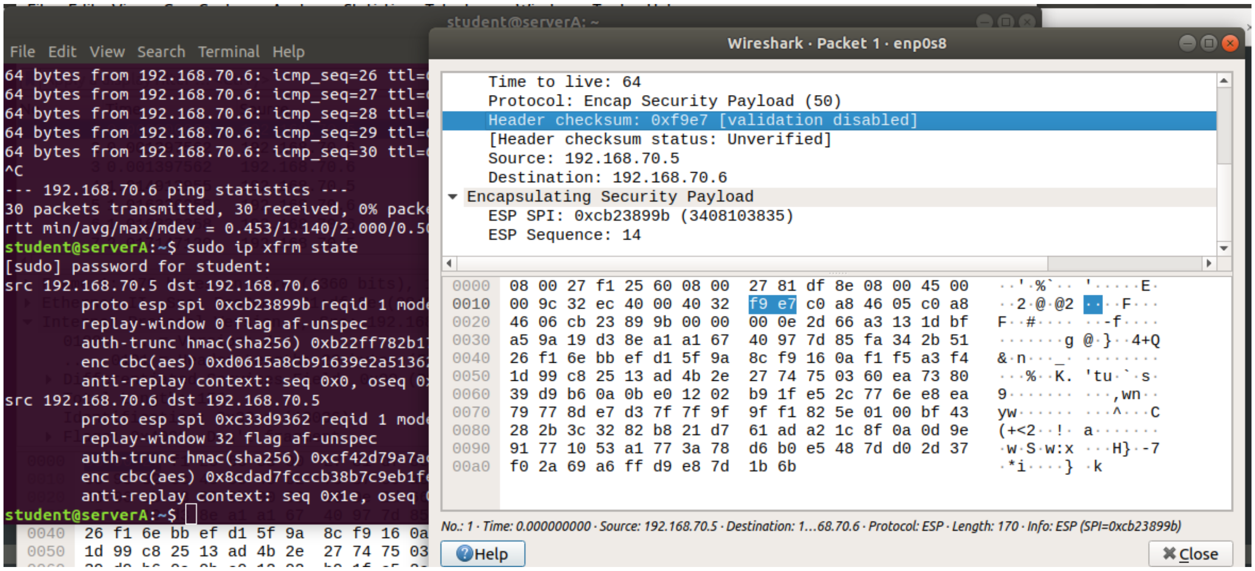
```
student@serverA:~$ sudo ip xfrm state
[sudo] password for student:
src 192.168.70.5 dst 192.168.70.6
        proto esp spi 0xcb23899b reqid 1 mode tunnel
        replay-window 0 flag af-unspec
        auth-trunc hmac(sha256) 0xb22ff782b17b8f449887aa03904a917ffd172fc843d92ae3d4d58b3b71551b21 128
        enc cbc(aes) 0xd0615a8cb91639e2a51362b3e52d630534d90e7abc214e05a47b065ca8adc6a4
        anti-replay context: seq 0x0, oseq 0x1e, bitmap 0x00000000
src 192.168.70.6 dst 192.168.70.5
        proto esp spi 0xc33d9362 reqid 1 mode tunnel
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0xcf42d79a7ac06da8d5dfa56c1e745555ec5c578c0b4811a05cd58933a5eb12e1 128
        enc cbc(aes) 0x8cdad7fcccb38b7c9eb1fe7bcd5d66d11d38fbc08b7d22b37433044a99eb9e89
        anti-replay context: seq 0x1e, oseq 0x0, bitmap 0x3fffffff
student@serverA:~$ ^C
```

Configured ESP SAs in Wireshark



Protocol	Src IP	Dest IP	SPI	Encryption	Encryption Key	Authentication
IPv4	192.168.70.5	192.168.70.6	0xcb23899b	AES-CBC [RFC3602]	0xd0615a8cb91639e2a51362b3e52d630534d90e7abc214e05a47b065ca8adc6a4	HMAC-SHA-1-96 [RFC24
IPv4	192.168.70.6	192.168.70.5	0xc33d9362	AES-CBC [RFC3602]	0x8cdad7fcccb38b7c9eb1fe7bcd5d66d11d38fbc08b7d22b37433044a99eb9e89	HMAC-SHA-1-96 [RFC24

details of a decrypted ICMP packet



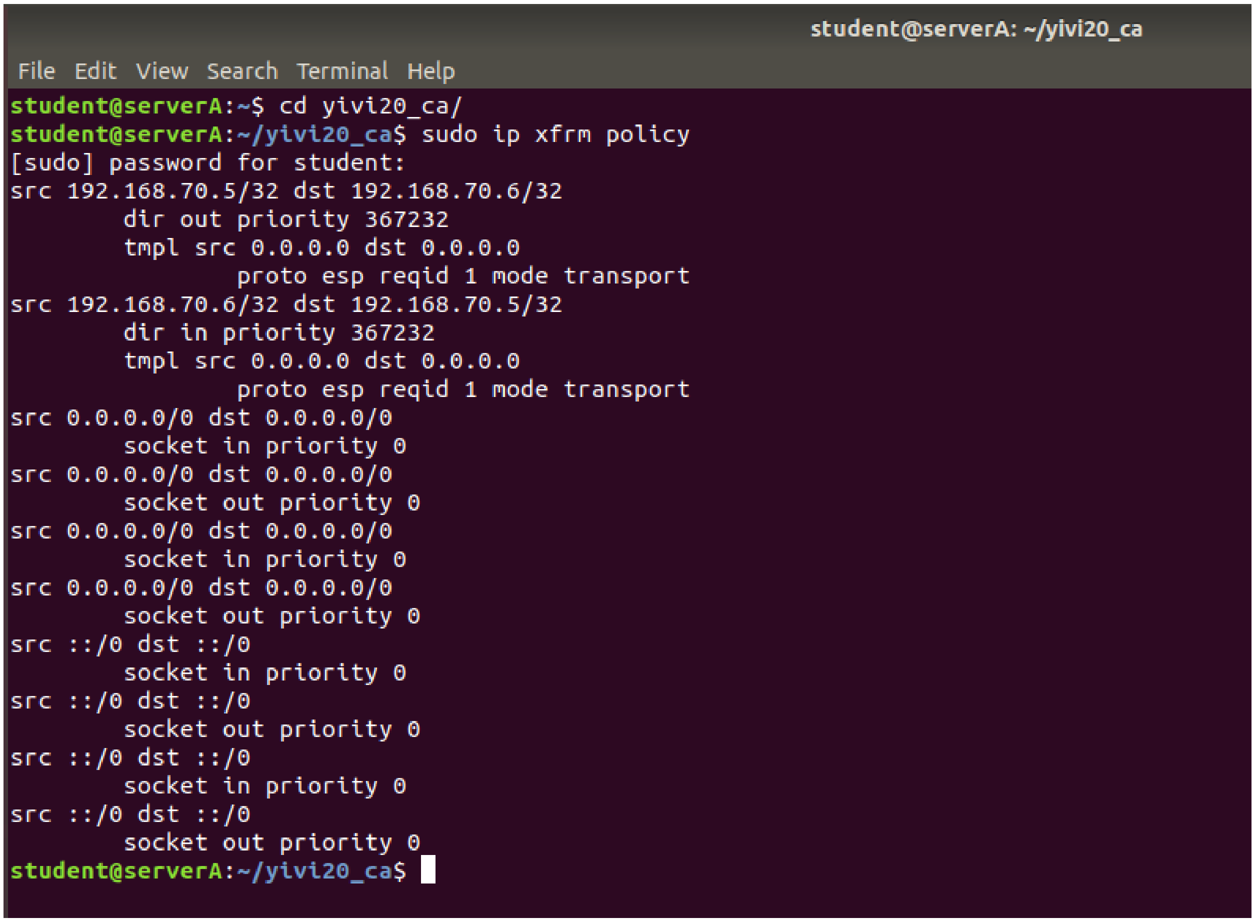
```
64 bytes from 192.168.70.6: icmp_seq=26 ttl=
64 bytes from 192.168.70.6: icmp_seq=27 ttl=
64 bytes from 192.168.70.6: icmp_seq=28 ttl=
64 bytes from 192.168.70.6: icmp_seq=29 ttl=
64 bytes from 192.168.70.6: icmp_seq=30 ttl=
^C
--- 192.168.70.6 ping statistics ---
30 packets transmitted, 30 received, 0% pack
rtt min/avg/max/mdev = 0.453/1.140/2.000/0.5
student@serverA:~$ sudo ip xfrm state
[sudo] password for student:
src 192.168.70.5 dst 192.168.70.6
        proto esp spi 0xcb23899b reqid 1 mode
        replay-window 0 flag af-unspec
        auth-trunc hmac(sha256) 0xb22ff782b1
        enc cbc(aes) 0xd0615a8cb91639e2a5136
        anti-replay context: seq 0x0, oseq 0
src 192.168.70.6 dst 192.168.70.5
        proto esp spi 0xc33d9362 reqid 1 mode
        replay-window 32 flag af-unspec
        auth-trunc hmac(sha256) 0xcf42d79a7a
        enc cbc(aes) 0x8cdad7fcccb38b7c9eb1f
        anti-replay context: seq 0x1e, oseq 
student@serverA:~$ 
 0040  26 f1 6e bb ef d1 5f 9a  8c f9 16 0a
 0050  1d 99 c8 25 13 ad 4b 2e  27 74 75 03
```

```
    Time to live: 64
    Protocol: Encap Security Payload (50)
    Header checksum: 0xf9e7 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.70.5
    Destination: 192.168.70.6
▾ Encapsulating Security Payload
    ESP SPI: 0xcb23899b (3408103835)
    ESP Sequence: 14

0000  08 00 27 f1 25 60 08 00  27 81 df 8e 08 00 45 00   ··'·%`·· '·····E·
0010  00 9c 32 ec 40 00 40 32  f9 e7 c0 a8 46 05 c0 a8   ··2·@·@2 ····F···
0020  46 06 cb 23 89 9b 00 00  00 0e 2d 66 a3 13 1d bf   F··#···· ··-f····
0030  a5 9a 19 d3 8e a1 a1 67  40 97 7d 85 fa 34 2b 51   ·······g @·}··4+Q
0040  26 f1 6e bb ef d1 5f 9a  8c f9 16 0a f1 f5 a3 f4   &·n···_· ········
0050  1d 99 c8 25 13 ad 4b 2e  27 74 75 03 60 ea 73 80   ···%··K. 'tu·`·s·
0060  39 d9 b6 0a 0b e0 12 02  b9 1f e5 2c 77 6e e8 ea   9······· ···,wn··
0070  79 77 8d e7 d3 7f 7f 9f  9f f1 82 5e 01 00 bf 43   yw······ ···^···C
0080  28 2b 3c 32 82 b8 21 d7  61 ad a2 1c 8f 0a 0d 9e   (+<2··!· a·······
0090  91 77 10 53 a1 77 3a 78  d6 b0 e5 48 7d d0 2d 37   ·w·S·w:x ···H}·-7
00a0  f0 2a 69 a6 ff d9 e8 7d  1b 6b                     ·*i···}  ·k
```

No.: 1 · Time: 0.000000000 · Source: 192.168.70.5 · Destination: 1…68.70.6 · Protocol: ESP · Length: 170 · Info: ESP (SPI=0xcb23899b)

Task 17: List the entries in the SPD

The output of `sudo ip xfrm policy` is as on the following screenshot


student@serverA: ~/yivi20_ca
File Edit View Search Terminal Help

```
student@serverA:~$ cd yivi20_ca/
student@serverA:~/yivi20_ca$ sudo ip xfrm policy
[sudo] password for student:
src 192.168.70.5/32 dst 192.168.70.6/32
        dir out priority 367232
        tmpl src 0.0.0.0 dst 0.0.0.0
                proto esp reqid 1 mode transport
src 192.168.70.6/32 dst 192.168.70.5/32
        dir in priority 367232
        tmpl src 0.0.0.0 dst 0.0.0.0
                proto esp reqid 1 mode transport
src 0.0.0.0/0 dst 0.0.0.0/0
        socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket out priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
        socket out priority 0
src ::/0 dst ::/0
        socket in priority 0
src ::/0 dst ::/0
        socket out priority 0
src ::/0 dst ::/0
        socket in priority 0
src ::/0 dst ::/0
        socket out priority 0
student@serverA:~/yivi20_ca$ 
```

- in keyword: selects the policy direction as **in**
- tmpl keyword: It is a template list specified using ID, MODE, REQID, and/or LEVEL.
- Esp keyword identifies the protocol used.
- Transport keyword: specifies a mode of operation for the transform protocol. In other words it says IPsec and IP Payload Compression mode is **transport**

Task 18: Host-to-host transport mode VPN with cert authentication

At the beginning for this task, I used this scp command

```
sudo scp -r /home/student/yivi20_ca
student@192.168.70.6:/home/student/
```

On serverA to copy certificates and necessary files to serverB.
I created server A certificate with these commands;

```
openssl genrsa -out ca1/private/192.168.70.5.key.pem 2048
```

```
openssl req -config ca1/openssl.cnf -new -sha256 -key
ca1/private/192.168.70.5.key.pem -out
ca1/csr/192.168.70.5.csr.pem
```

```
openssl ca -config ca1/openssl.cnf -extensions server_cert -days
375 -notext -md sha256 -in ca1/csr/192.168.70.5.csr.pem -out
ca1/certs/192.168.70.5.cert.pem
```

Similarly, these same commands were run on serverB changing `192.168.70.5 `to `192.168.70.6` in serverB.

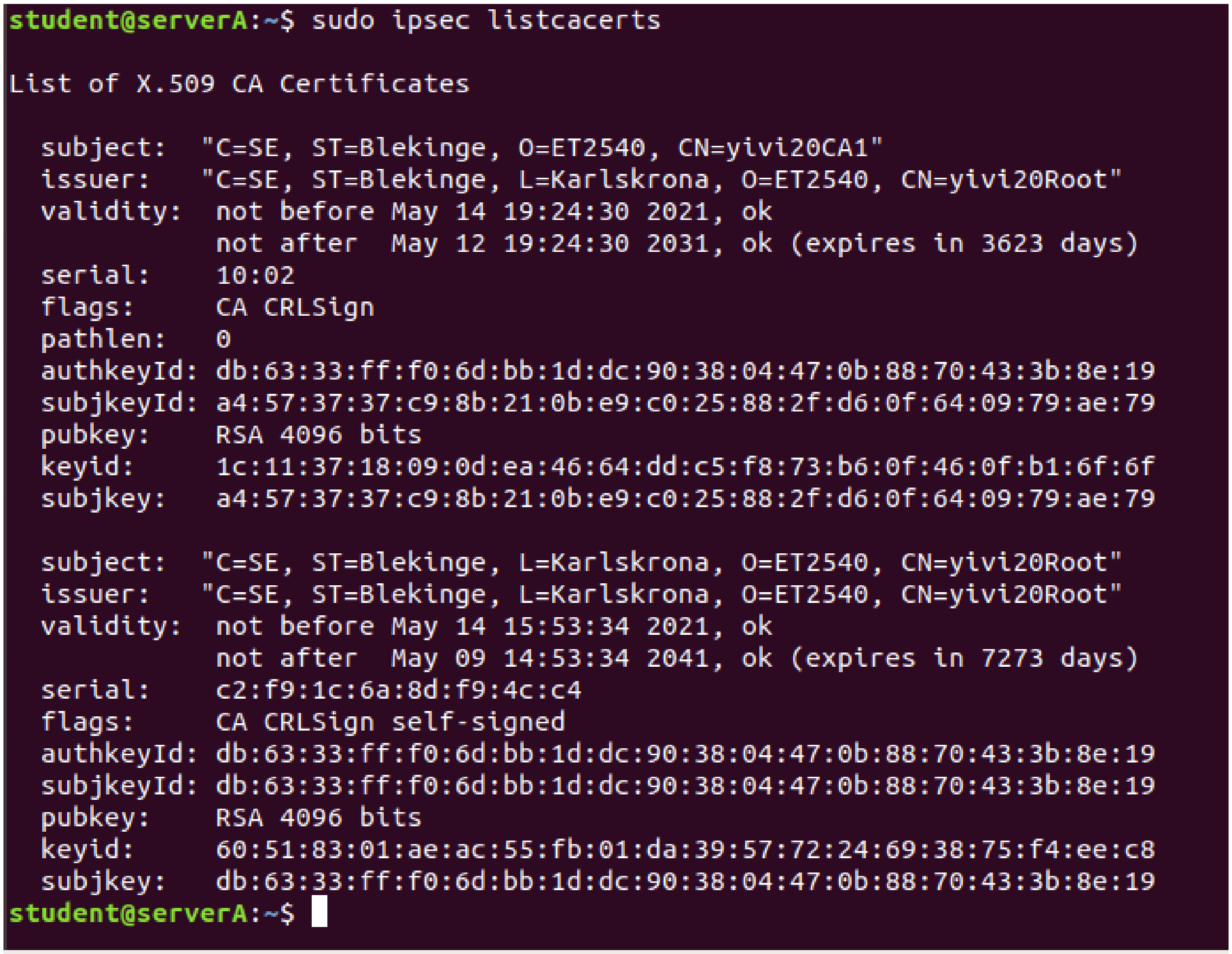
Copying certs to into strongswan directories

These commands were used on server B to do the copying

```
sudo cp root.cert.pem /etc/ipsec.d/cacerts
sudo cp ca1.cert.pem /etc/ipsec.d/cacerts
sudo cp 192.168.70.6.cert.pem /etc/ipsec.d/certs
sudo cp 192.168.70.6.key.pem /etc/ipsec.d/private/
```

On server A, the same commands were run while changing `192.168.70.6` to `192.168.70.5` for the second and third commands.

On server A, the output of `sudo ipsec rereadcacerts` and `sudo ipsec listcacerts` is as seen below



```
student@serverA:~$ sudo ipsec listcacerts

List of X.509 CA Certificates

  subject:  "C=SE, ST=Blekinge, O=ET2540, CN=yivi20CA1"
  issuer:   "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, CN=yivi20Root"
  validity:  not before May 14 19:24:30 2021, ok
             not after  May 12 19:24:30 2031, ok (expires in 3623 days)
  serial:    10:02
  flags:     CA CRLSign
  pathlen:   0
  authkeyId: db:63:33:ff:f0:6d:bb:1d:dc:90:38:04:47:0b:88:70:43:3b:8e:19
  subjkeyId: a4:57:37:37:c9:8b:21:0b:e9:c0:25:88:2f:d6:0f:64:09:79:ae:79
  pubkey:    RSA 4096 bits
  keyid:     1c:11:37:18:09:0d:ea:46:64:dd:c5:f8:73:b6:0f:46:0f:b1:6f:6f
  subjkey:   a4:57:37:37:c9:8b:21:0b:e9:c0:25:88:2f:d6:0f:64:09:79:ae:79

  subject:  "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, CN=yivi20Root"
  issuer:   "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, CN=yivi20Root"
  validity:  not before May 14 15:53:34 2021, ok
             not after  May 09 14:53:34 2041, ok (expires in 7273 days)
  serial:    c2:f9:1c:6a:8d:f9:4c:c4
  flags:     CA CRLSign self-signed
  authkeyId: db:63:33:ff:f0:6d:bb:1d:dc:90:38:04:47:0b:88:70:43:3b:8e:19
  subjkeyId: db:63:33:ff:f0:6d:bb:1d:dc:90:38:04:47:0b:88:70:43:3b:8e:19
  pubkey:    RSA 4096 bits
  keyid:     60:51:83:01:ae:ac:55:fb:01:da:39:57:72:24:69:38:75:f4:ee:c8
  subjkey:   db:63:33:ff:f0:6d:bb:1d:dc:90:38:04:47:0b:88:70:43:3b:8e:19
student@serverA:~$ 
```

Only the configuration files for server A are shown.Those of Server B are more much similar

`ipsec.cnf` is as follows

```
config setup
        charondebug="all"
        uniqueids=yes
        strictcrlpolicy=no
conn ServerA-to-ServerB
        left=192.168.70.5
        right=192.168.70.6
        ike=aes256-sha2_256-modp1024!
        esp=aes256-sha2_256!
        keyingtries=0
        ikelifetime=1h
        lifetime=8h
        dpddelay=30
        dpdtimeout=120
        dpdaction=restart
        auto=start
        keyexchange=ikev2
        type=transport
        leftcert=192.168.70.5.cert.pem
        leftid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5"
        rightid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6"
```

`ipsec.secrets` is as follows;

```
: RSA 192.168.70.5.key.pem
```

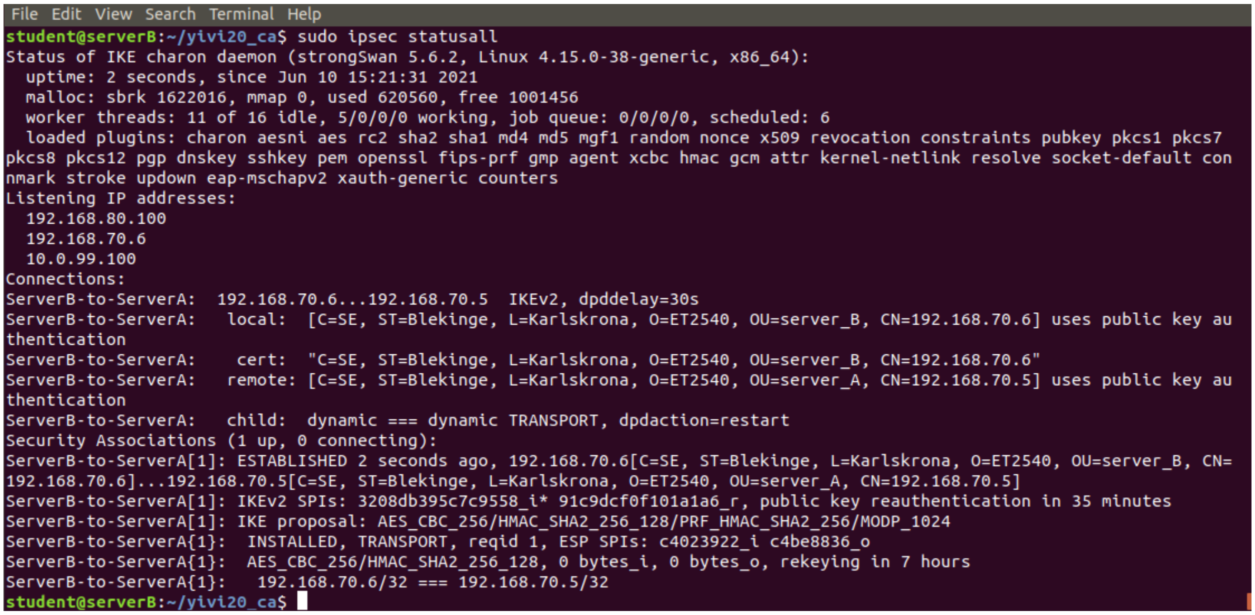
Similarly, below are copies of `ipsec.conf` and `ipsec.secrets` for serverB

```
config setup
        charondebug="all"
        uniqueids=yes
        strictcrlpolicy=no
conn ServerA-to-ServerB
        left=192.168.70.6
        right=192.168.70.5
        ike=aes256-sha2_256-modp1024!
        esp=aes256-sha2_256!
        keyingtries=0
        ikelifetime=1h
        lifetime=8h
        dpddelay=30
        dpdtimeout=120
        dpdaction=restart
        auto=start
        keyexchange=ikev2
        type=transport
        leftcert=192.168.70.6.cert.pem
        leftid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_A, CN=192.168.70.6"
        rightid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_B, CN=192.168.70.5"
```

```
: RSA 192.168.70.6.key.pem
```

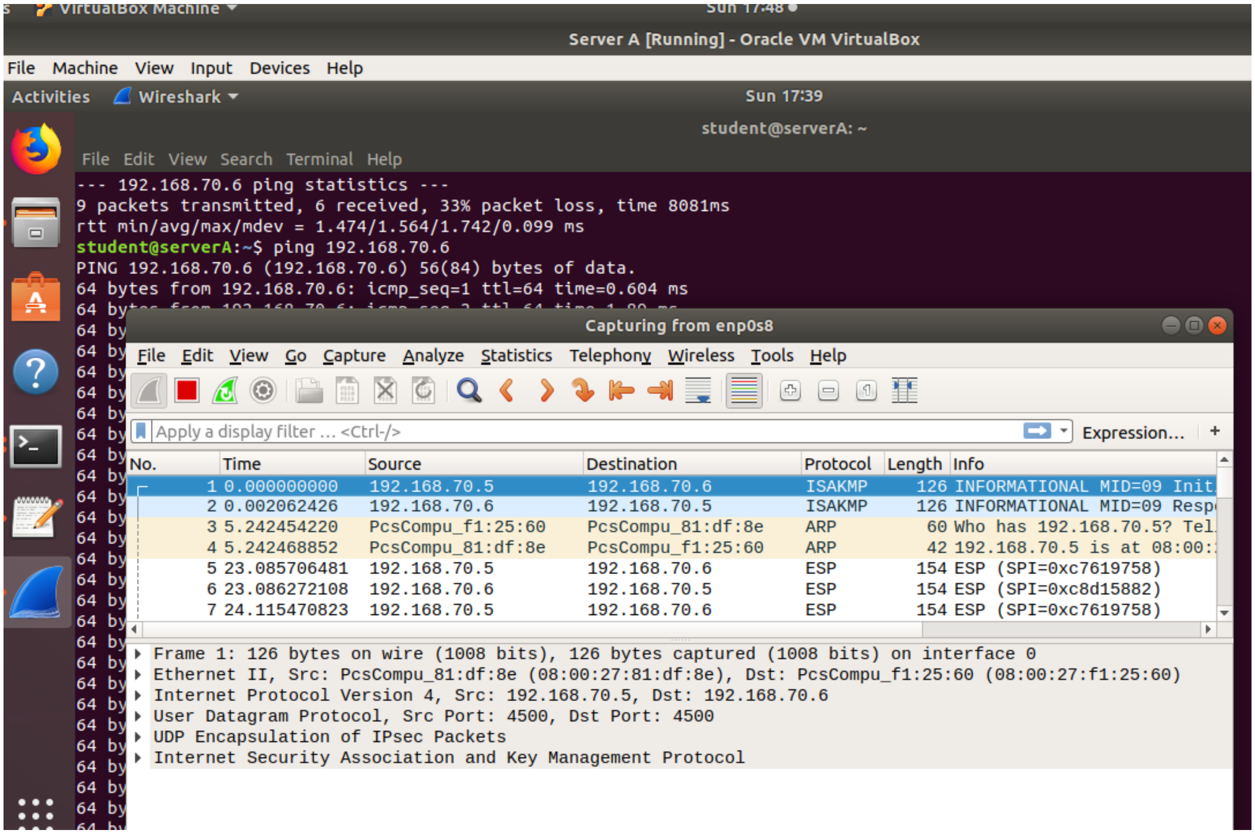
The following commands on any of the servers now shows evidence of the VPN transport mode

```
sudo ipsec restart
sudo ipsec statusall
```



```
student@serverB:~/yivi20_ca$ sudo ipsec statusall
Status of IKE charon daemon (strongSwan 5.6.2, Linux 4.15.0-38-generic, x86_64):
  uptime: 2 seconds, since Jun 10 15:21:31 2021
  malloc: sbrk 1622016, mmap 0, used 620560, free 1001456
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 6
  loaded plugins: charon aesni aes rc2 sha2 sha1 md4 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7
pkcs8 pkcs12 pgp dnskey sshkey pem openssl fips-prf gmp agent xcbc hmac gcm attr kernel-netlink resolve socket-default con
nmark stroke updown eap-mschapv2 xauth-generic counters
Listening IP addresses:
  192.168.80.100
  192.168.70.6
  10.0.99.100
Connections:
ServerB-to-ServerA:  192.168.70.6...192.168.70.5  IKEv2, dpddelay=30s
ServerB-to-ServerA:   local:  [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6] uses public key au
thentication
ServerB-to-ServerA:    cert:  "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6"
ServerB-to-ServerA:   remote: [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5] uses public key au
thentication
ServerB-to-ServerA:   child:  dynamic === dynamic TRANSPORT, dpdaction=restart
Security Associations (1 up, 0 connecting):
ServerB-to-ServerA[1]: ESTABLISHED 2 seconds ago, 192.168.70.6[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=
192.168.70.6]...192.168.70.5[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5]
ServerB-to-ServerA[1]: IKEv2 SPIs: 3208db395c7c9558_i* 91c9dcf0f101a1a6_r, public key reauthentication in 35 minutes
ServerB-to-ServerA[1]: IKE proposal: AES_CBC_256/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_1024
ServerB-to-ServerA{1}:  INSTALLED, TRANSPORT, reqid 1, ESP SPIs: c4023922_i c4be8836_o
ServerB-to-ServerA{1}:  AES_CBC_256/HMAC_SHA2_256_128, 0 bytes_i, 0 bytes_o, rekeying in 7 hours
ServerB-to-ServerA{1}:   192.168.70.6/32 === 192.168.70.5/32
student@serverB:~/yivi20_ca$ 
```

pinging 192.168.70.6 from server A shows that we have encrypted traffic.


VirtualBox Machine
Sun 17:48
Server A [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Wireshark
Sun 17:39
student@serverA: ~
File Edit View Search Terminal Help
Capturing from enp0s8
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…
+

```
--- 192.168.70.6 ping statistics ---
9 packets transmitted, 6 received, 33% packet loss, time 8081ms
rtt min/avg/max/mdev = 1.474/1.564/1.742/0.099 ms
student@serverA:~$ ping 192.168.70.6
PING 192.168.70.6 (192.168.70.6) 56(84) bytes of data.
64 bytes from 192.168.70.6: icmp_seq=1 ttl=64 time=0.604 ms
```

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.70.5	192.168.70.6	ISAKMP	126	INFORMATIONAL MID=09 Init
2	0.002062426	192.168.70.6	192.168.70.5	ISAKMP	126	INFORMATIONAL MID=09 Resp
3	5.242454220	PcsCompu_f1:25:60	PcsCompu_81:df:8e	ARP	60	Who has 192.168.70.5? Tel
4	5.242468852	PcsCompu_81:df:8e	PcsCompu_f1:25:60	ARP	42	192.168.70.5 is at 08:00:
5	23.085706481	192.168.70.5	192.168.70.6	ESP	154	ESP (SPI=0xc7619758)
6	23.086272108	192.168.70.6	192.168.70.5	ESP	154	ESP (SPI=0xc8d15882)
7	24.115470823	192.168.70.5	192.168.70.6	ESP	154	ESP (SPI=0xc7619758)

```
▸ Frame 1: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits) on interface 0
▸ Ethernet II, Src: PcsCompu_81:df:8e (08:00:27:81:df:8e), Dst: PcsCompu_f1:25:60 (08:00:27:f1:25:60)
▸ Internet Protocol Version 4, Src: 192.168.70.5, Dst: 192.168.70.6
▸ User Datagram Protocol, Src Port: 4500, Dst Port: 4500
▸ UDP Encapsulation of IPsec Packets
▸ Internet Security Association and Key Management Protocol
```

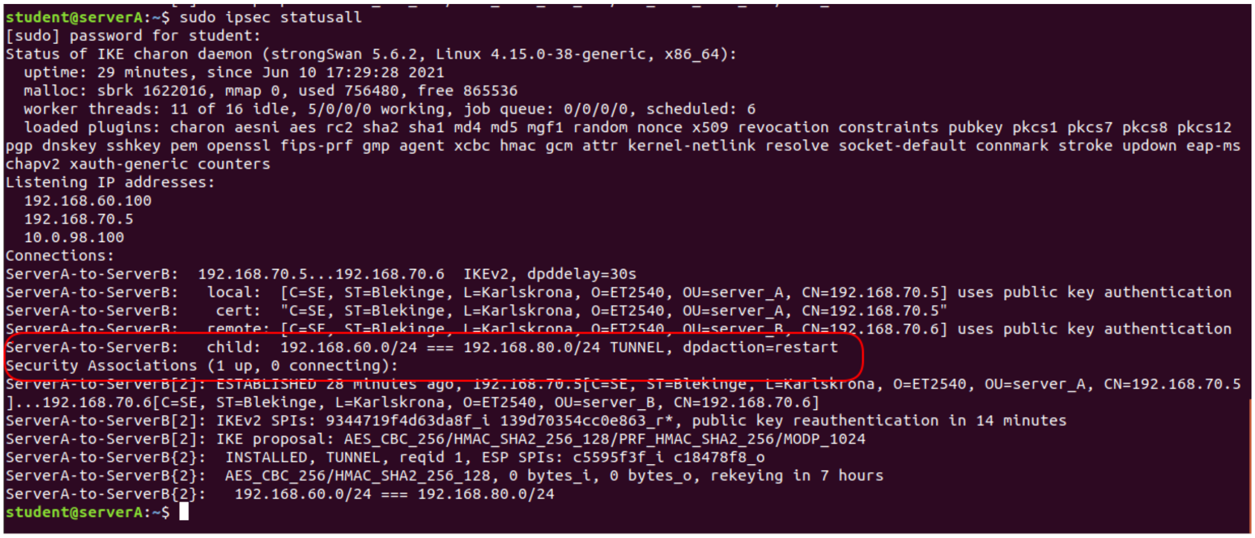
Task 19: Tunnel mode VPN with cert authentication between Server A and Server B

To accomplish this task, I modified only the `ipsec.conf` on both servers. `ipsec.secrets` has not been changed as of the last task. Below are the ipsec.conf files for servers A and B respectively.

```
config setup
        charondebug="all"
        uniqueids=yes
        strictcrlpolicy=no
conn ServerA-to-ServerB
        left=192.168.70.5
        leftsubnet=192.168.60.0/24
        right=192.168.70.6
        rightsubnet=192.168.80.0/24
        ike=aes256-sha2_256-modp1024!
        esp=aes256-sha2_256!
        keyingtries=0
        ikelifetime=1h
        lifetime=8h
        dpddelay=30
        dpdtimeout=120
        dpdaction=restart
        auto=start
        keyexchange=ikev2
        type=tunnel
        leftcert=192.168.70.5.cert.pem
        leftid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_A, CN=192.168.70.5"
        rightid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_B, CN=192.168.70.6"
```

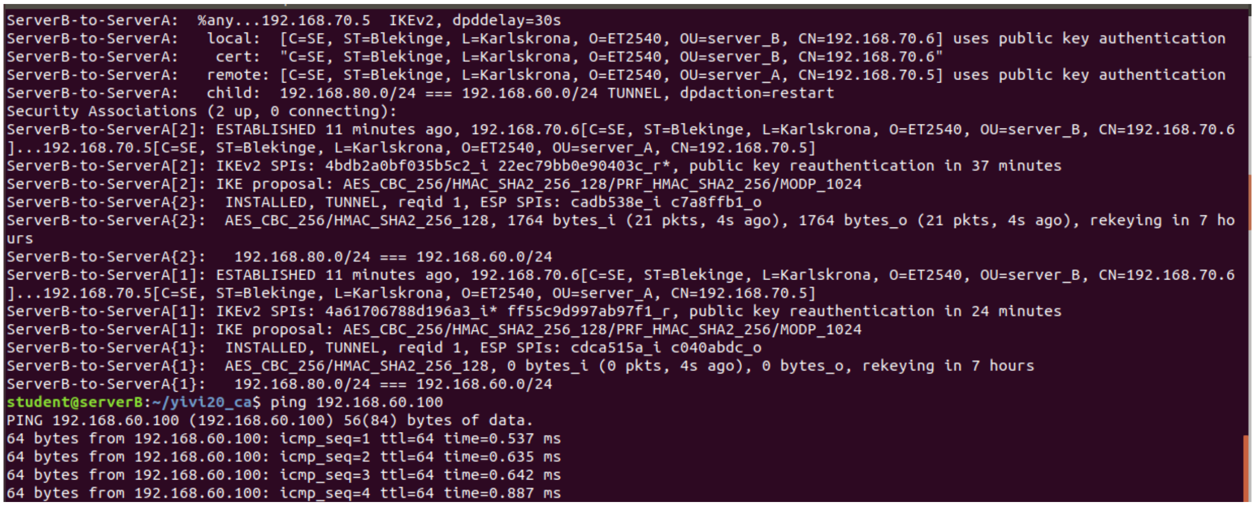
```
config setup
        charondebug="all"
        uniqueids=yes
        strictcrlpolicy=no
conn ServerA-to-ServerB
        left=192.168.70.6
        leftsubnet=192.168.80.0/24
        right=192.168.70.5
        rightsubnet=192.168.60.0/24
        ike=aes256-sha2_256-modp1024!
        esp=aes256-sha2_256!
        keyingtries=0
        ikelifetime=1h
        lifetime=8h
        dpddelay=30
        dpdtimeout=120
        dpdaction=restart
        auto=start
        keyexchange=ikev2
        type=tunnel
        leftcert=192.168.70.6.cert.pem
        leftid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_A, CN=192.168.70.6"
        rightid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_B, CN=192.168.70.5"
```

The following commands now shows `sudo ipsec restart` and `sudo ipsec statusall` now shows the following output

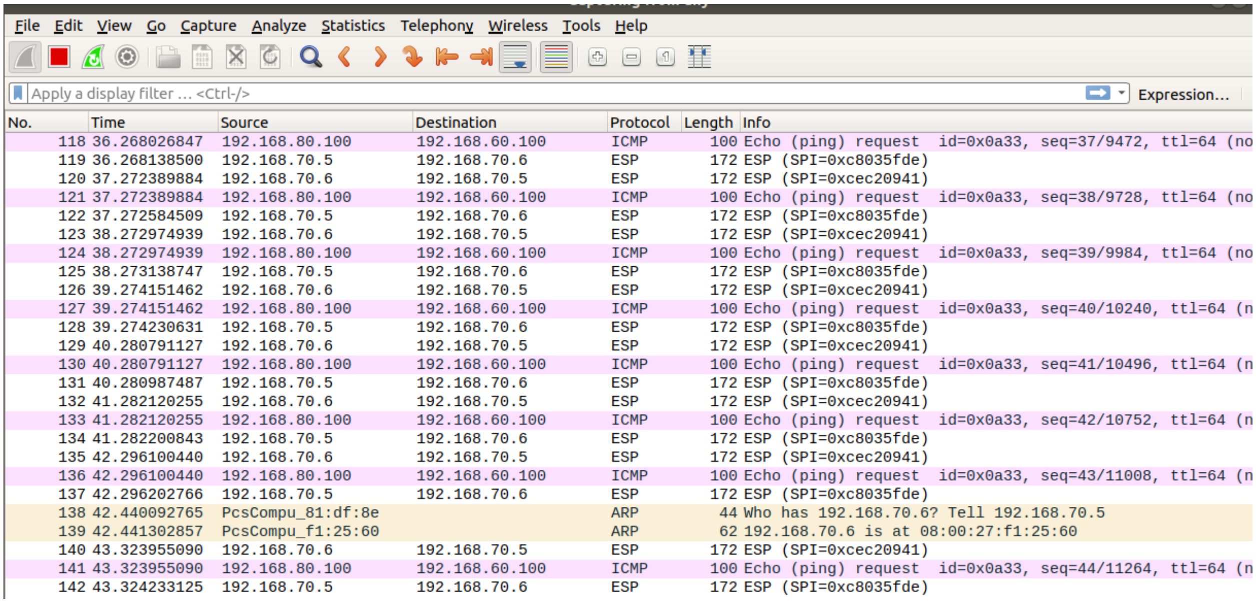


```
student@serverA:~$ sudo ipsec statusall
[sudo] password for student:
Status of IKE charon daemon (strongSwan 5.6.2, Linux 4.15.0-38-generic, x86_64):
  uptime: 29 minutes, since Jun 10 17:29:28 2021
  malloc: sbrk 1622016, mmap 0, used 756480, free 865536
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 6
  loaded plugins: charon aesni aes rc2 sha2 sha1 md4 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7 pkcs8 pkcs12
pgp dnskey sshkey pem openssl fips-prf gmp agent xcbc hmac gcm attr kernel-netlink resolve socket-default connmark stroke updown eap-ms
chapv2 xauth-generic counters
Listening IP addresses:
  192.168.60.100
  192.168.70.5
  10.0.98.100
Connections:
ServerA-to-ServerB:  192.168.70.5...192.168.70.6  IKEv2, dpddelay=30s
ServerA-to-ServerB:   local:  [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5] uses public key authentication
ServerA-to-ServerB:    cert:  "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5"
ServerA-to-ServerB:   remote: [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6] uses public key authentication
ServerA-to-ServerB:   child:  192.168.60.0/24 === 192.168.80.0/24 TUNNEL, dpdaction=restart
Security Associations (1 up, 0 connecting):
ServerA-to-ServerB[2]: ESTABLISHED 28 minutes ago, 192.168.70.5[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5
]...192.168.70.6[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6]
ServerA-to-ServerB[2]: IKEv2 SPIs: 9344719f4d63da8f_i 139d70354cc0e863_r*, public key reauthentication in 14 minutes
ServerA-to-ServerB[2]: IKE proposal: AES_CBC_256/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_1024
ServerA-to-ServerB{2}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: c5595f3f_i c18478f8_o
ServerA-to-ServerB{2}:  AES_CBC_256/HMAC_SHA2_256_128, 0 bytes_i, 0 bytes_o, rekeying in 7 hours
ServerA-to-ServerB{2}:   192.168.60.0/24 === 192.168.80.0/24
student@serverA:~$
```

To confirm, pinging 192.168.60.100 from server B and capturing wireshark traffic on server A,the observation is as seen on the following screenshot



```
ServerB-to-ServerA:  %any...192.168.70.5  IKEv2, dpddelay=30s
ServerB-to-ServerA:   local:  [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6] uses public key authentication
ServerB-to-ServerA:    cert:  "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6"
ServerB-to-ServerA:   remote: [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5] uses public key authentication
ServerB-to-ServerA:   child:  192.168.80.0/24 === 192.168.60.0/24 TUNNEL, dpdaction=restart
Security Associations (2 up, 0 connecting):
ServerB-to-ServerA[2]: ESTABLISHED 11 minutes ago, 192.168.70.6[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6
]...192.168.70.5[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5]
ServerB-to-ServerA[2]: IKEv2 SPIs: 4bdb2a0bf035b5c2_i 22ec79bb0e90403c_r*, public key reauthentication in 37 minutes
ServerB-to-ServerA[2]: IKE proposal: AES_CBC_256/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_1024
ServerB-to-ServerA{2}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: cadb538e_i c7a8ffb1_o
ServerB-to-ServerA{2}:  AES_CBC_256/HMAC_SHA2_256_128, 1764 bytes_i (21 pkts, 4s ago), 1764 bytes_o (21 pkts, 4s ago), rekeying in 7 ho
urs
ServerB-to-ServerA{2}:   192.168.80.0/24 === 192.168.60.0/24
ServerB-to-ServerA[1]: ESTABLISHED 11 minutes ago, 192.168.70.6[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6
]...192.168.70.5[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5]
ServerB-to-ServerA[1]: IKEv2 SPIs: 4a61706788d196a3_i* ff55c9d997ab97f1_r, public key reauthentication in 24 minutes
ServerB-to-ServerA[1]: IKE proposal: AES_CBC_256/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_1024
ServerB-to-ServerA{1}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: cdca515a_i c040abdc_o
ServerB-to-ServerA{1}:  AES_CBC_256/HMAC_SHA2_256_128, 0 bytes_i (0 pkts, 4s ago), 0 bytes_o, rekeying in 7 hours
ServerB-to-ServerA{1}:   192.168.80.0/24 === 192.168.60.0/24
student@serverB:~/yivi20_ca$ ping 192.168.60.100
PING 192.168.60.100 (192.168.60.100) 56(84) bytes of data.
64 bytes from 192.168.60.100: icmp_seq=1 ttl=64 time=0.537 ms
64 bytes from 192.168.60.100: icmp_seq=2 ttl=64 time=0.635 ms
64 bytes from 192.168.60.100: icmp_seq=3 ttl=64 time=0.642 ms
64 bytes from 192.168.60.100: icmp_seq=4 ttl=64 time=0.887 ms
```



No.	Time	Source	Destination	Protocol	Length	Info
118	36.268026847	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=37/9472, ttl=64 (no
119	36.268138500	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
120	37.272389884	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
121	37.272389884	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=38/9728, ttl=64 (no
122	37.272584509	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
123	38.272974939	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
124	38.272974939	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=39/9984, ttl=64 (no
125	38.273138747	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
126	39.274151462	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
127	39.274151462	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=40/10240, ttl=64 (n
128	39.274230631	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
129	40.280791127	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
130	40.280791127	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=41/10496, ttl=64 (n
131	40.280987487	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
132	41.282120255	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
133	41.282120255	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=42/10752, ttl=64 (n
134	41.282200843	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
135	42.296100440	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
136	42.296100440	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=43/11008, ttl=64 (n
137	42.296202766	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)
138	42.440092765	PcsCompu_81:df:8e		ARP	44	Who has 192.168.70.6? Tell 192.168.70.5
139	42.441302857	PcsCompu_f1:25:60		ARP	62	192.168.70.6 is at 08:00:27:f1:25:60
140	43.323955090	192.168.70.6	192.168.70.5	ESP	172	ESP (SPI=0xcec20941)
141	43.323955090	192.168.80.100	192.168.60.100	ICMP	100	Echo (ping) request id=0x0a33, seq=44/11264, ttl=64 (n
142	43.324233125	192.168.70.5	192.168.70.6	ESP	172	ESP (SPI=0xc8035fde)

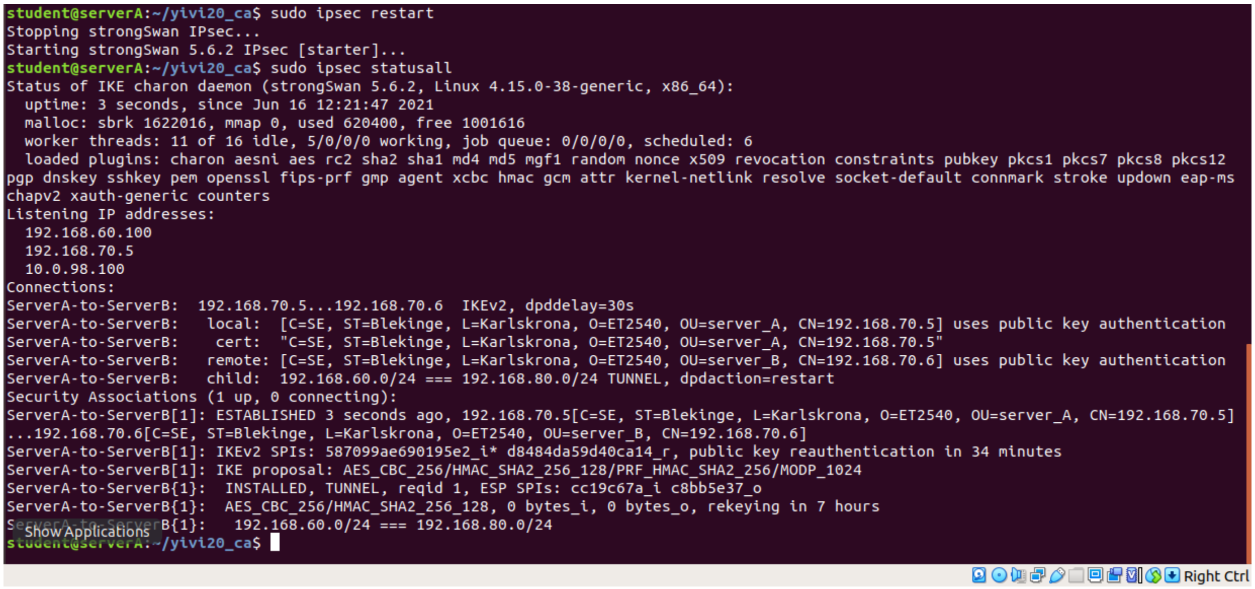
Task 20: Tunnel mode VPN with IP forwarding for client A and client B

For this task, the modifications made on the ipsec.conf files is adding **leftsourceip** and **rightsourceip** parameters to ipsec.conf files. For example, ipsec.conf file of server A is as follows.

```
config setup
        charondebug="all"
        uniqueids=yes
        strictcrlpolicy=no
```

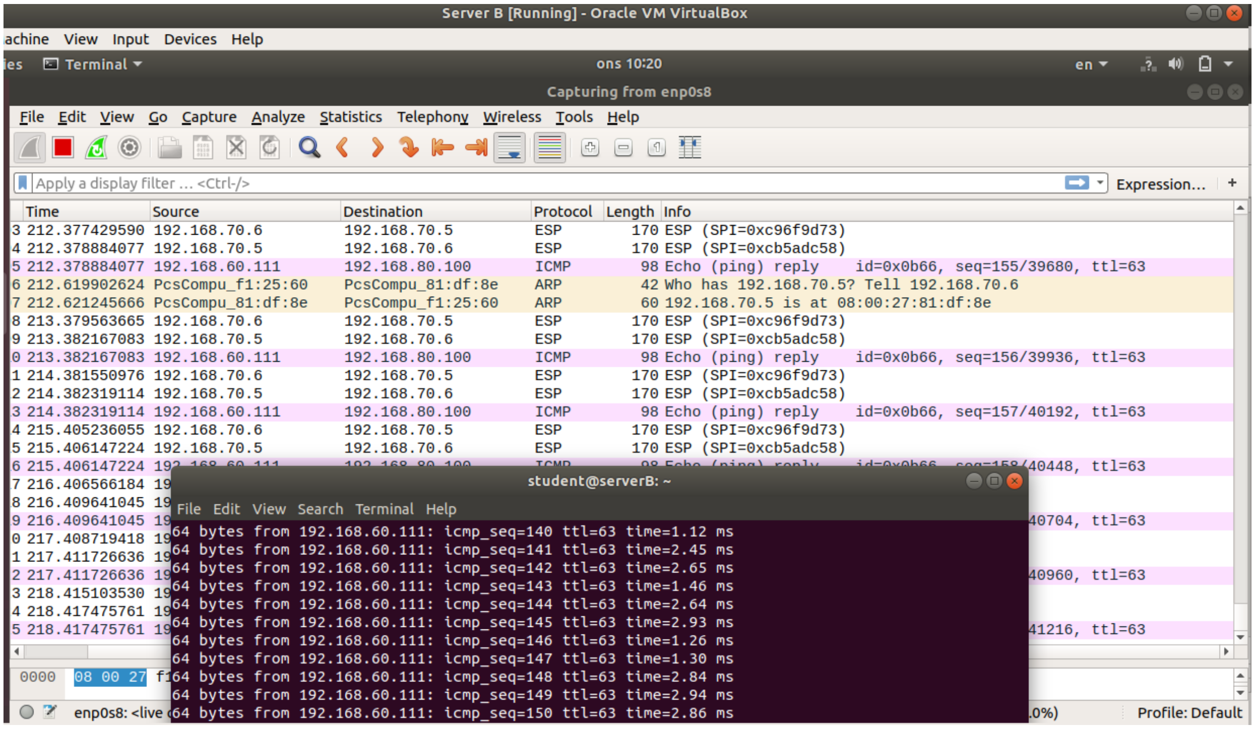
```
conn ServerA-to-ServerB
        left=192.168.70.5
        leftsubnet=192.168.60.0/24
        leftsourceip =192.168.60.111
        right=192.168.70.6
        rightsubnet=192.168.80.0/24
        rightsourceip =192.168.80.111
        ike=aes256-sha2_256-modp1024!
        esp=aes256-sha2_256!
        keyingtries=0
        ikelifetime=1h
        lifetime=8h
        dpddelay=30
        dpdtimeout=120
        dpdaction=restart
        auto=start
        keyexchange=ikev2
        type=tunnel
        leftcert=192.168.70.5.cert.pem
        leftid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_A, CN=192.168.70.5"
        rightid="C=SE, ST=Blekinge, L=Karlskrona, O=ET2540,
OU=server_B, CN=192.168.70.6"
```

sudo ipsec restart and **sudo ipsec statusall** commands gives the following output



```
student@serverA:~/yivi20_ca$ sudo ipsec restart
Stopping strongSwan IPsec...
Starting strongSwan 5.6.2 IPsec [starter]...
student@serverA:~/yivi20_ca$ sudo ipsec statusall
Status of IKE charon daemon (strongSwan 5.6.2, Linux 4.15.0-38-generic, x86_64):
  uptime: 3 seconds, since Jun 16 12:21:47 2021
  malloc: sbrk 1622016, mmap 0, used 620400, free 1001616
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 6
  loaded plugins: charon aesni aes rc2 sha2 sha1 md4 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7 pkcs8 pkcs12
pgp dnskey sshkey pem openssl fips-prf gmp agent xcbc hmac gcm attr kernel-netlink resolve socket-default connmark stroke updown eap-ms
chapv2 xauth-generic counters
Listening IP addresses:
  192.168.60.100
  192.168.70.5
  10.0.98.100
Connections:
ServerA-to-ServerB:  192.168.70.5...192.168.70.6  IKEv2, dpddelay=30s
ServerA-to-ServerB:   local:  [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5] uses public key authentication
ServerA-to-ServerB:    cert:  "C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5"
ServerA-to-ServerB:   remote: [C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6] uses public key authentication
ServerA-to-ServerB:   child:  192.168.60.0/24 === 192.168.80.0/24 TUNNEL, dpdaction=restart
Security Associations (1 up, 0 connecting):
ServerA-to-ServerB[1]: ESTABLISHED 3 seconds ago, 192.168.70.5[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_A, CN=192.168.70.5]
...192.168.70.6[C=SE, ST=Blekinge, L=Karlskrona, O=ET2540, OU=server_B, CN=192.168.70.6]
ServerA-to-ServerB[1]: IKEv2 SPIs: 587099ae690195e2_i* d8484da59d40ca14_r, public key reauthentication in 34 minutes
ServerA-to-ServerB[1]: IKE proposal: AES_CBC_256/HMAC_SHA2_256_128/PRF_HMAC_SHA2_256/MODP_1024
ServerA-to-ServerB{1}:  INSTALLED, TUNNEL, reqid 1, ESP SPIs: cc19c67a_i c8bb5e37_o
ServerA-to-ServerB{1}:  AES_CBC_256/HMAC_SHA2_256_128, 0 bytes_i, 0 bytes_o, rekeying in 7 hours
ServerA-to-ServerB{1}:   192.168.60.0/24 === 192.168.80.0/24
student@serverA:~/yivi20_ca$
```

Pinging the client A(192.168.60.111) from server B and observing the traffic on wireshark on server B shows the following



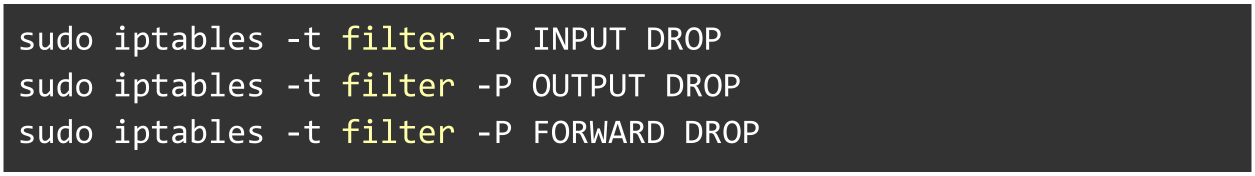
Server B [Running] - Oracle VM VirtualBox
achine View Input Devices Help
ies Terminal
ons 10:20
en
Capturing from enp0s8
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Expression…
student@serverB: ~
File Edit View Search Terminal Help
enp0s8: <live
.0%)
Profile: Default

```
  Time          Source             Destination        Protocol  Length Info
3 212.377429590 192.168.70.6       192.168.70.5       ESP          170 ESP (SPI=0xc96f9d73)
4 212.378884077 192.168.70.5       192.168.70.6       ESP          170 ESP (SPI=0xcb5adc58)
5 212.378884077 192.168.60.111     192.168.80.100     ICMP          98 Echo (ping) reply    id=0x0b66, seq=155/39680, ttl=63
6 212.619902624 PcsCompu_f1:25:60  PcsCompu_81:df:8e  ARP           42 Who has 192.168.70.5? Tell 192.168.70.6
7 212.621245666 PcsCompu_81:df:8e  PcsCompu_f1:25:60  ARP           60 192.168.70.5 is at 08:00:27:81:df:8e
8 213.379563665 192.168.70.6       192.168.70.5       ESP          170 ESP (SPI=0xc96f9d73)
9 213.382167083 192.168.70.5       192.168.70.6       ESP          170 ESP (SPI=0xcb5adc58)
0 213.382167083 192.168.60.111     192.168.80.100     ICMP          98 Echo (ping) reply    id=0x0b66, seq=156/39936, ttl=63
1 214.381550976 192.168.70.6       192.168.70.5       ESP          170 ESP (SPI=0xc96f9d73)
2 214.382319114 192.168.70.5       192.168.70.6       ESP          170 ESP (SPI=0xcb5adc58)
3 214.382319114 192.168.60.111     192.168.80.100     ICMP          98 Echo (ping) reply    id=0x0b66, seq=157/40192, ttl=63
4 215.405236055 192.168.70.6       192.168.70.5       ESP          170 ESP (SPI=0xc96f9d73)
5 215.406147224 192.168.70.5       192.168.70.6       ESP          170 ESP (SPI=0xcb5adc58)
6 215.406147224 192.168.60.111     192.168.80.100     ICMP          98 Echo (ping) reply    id=0x0b66, seq=158/40448, ttl=63
7 216.406566184 19
8 216.409641045 19
9 216.409641045 19                                                                                             40704, ttl=63
0 217.408719418 19
1 217.411726636 19
2 217.411726636 19                                                                                             40960, ttl=63
3 218.415103530 19
4 218.417475761 19
5 218.417475761 19                                                                                             41216, ttl=63

0000  08 00 27 f1
```

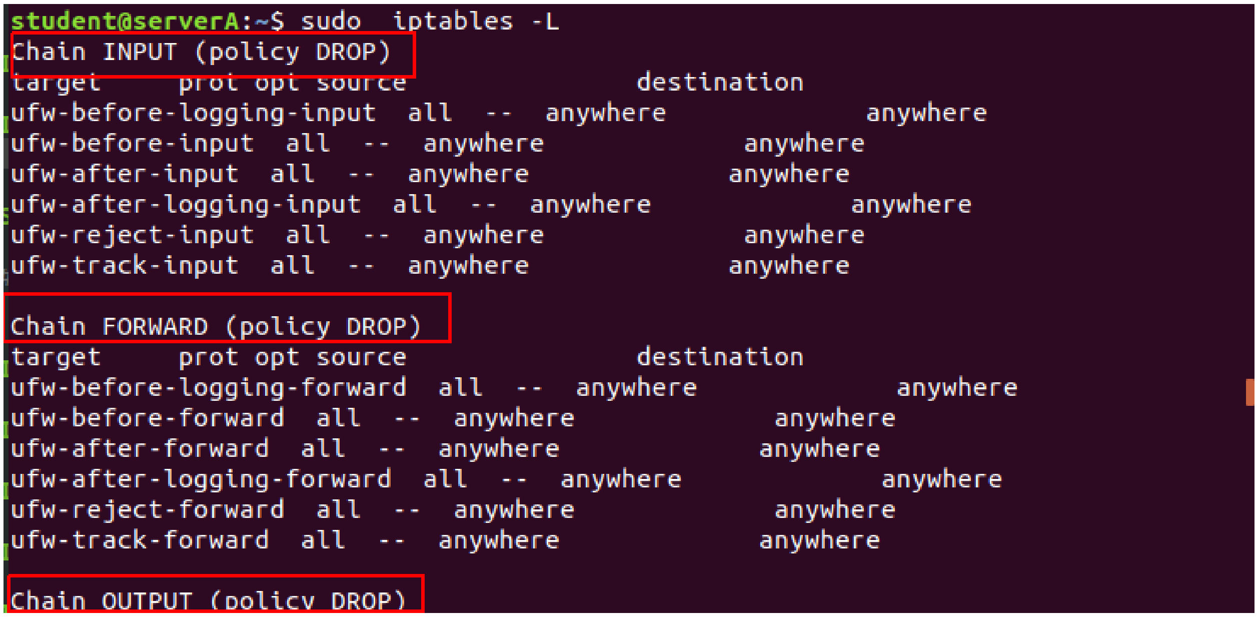
```
64 bytes from 192.168.60.111: icmp_seq=140 ttl=63 time=1.12 ms
64 bytes from 192.168.60.111: icmp_seq=141 ttl=63 time=2.45 ms
64 bytes from 192.168.60.111: icmp_seq=142 ttl=63 time=2.65 ms
64 bytes from 192.168.60.111: icmp_seq=143 ttl=63 time=1.46 ms
64 bytes from 192.168.60.111: icmp_seq=144 ttl=63 time=2.64 ms
64 bytes from 192.168.60.111: icmp_seq=145 ttl=63 time=2.93 ms
64 bytes from 192.168.60.111: icmp_seq=146 ttl=63 time=1.26 ms
64 bytes from 192.168.60.111: icmp_seq=147 ttl=63 time=1.30 ms
64 bytes from 192.168.60.111: icmp_seq=148 ttl=63 time=2.84 ms
64 bytes from 192.168.60.111: icmp_seq=149 ttl=63 time=2.94 ms
64 bytes from 192.168.60.111: icmp_seq=150 ttl=63 time=2.86 ms
```

Task 21

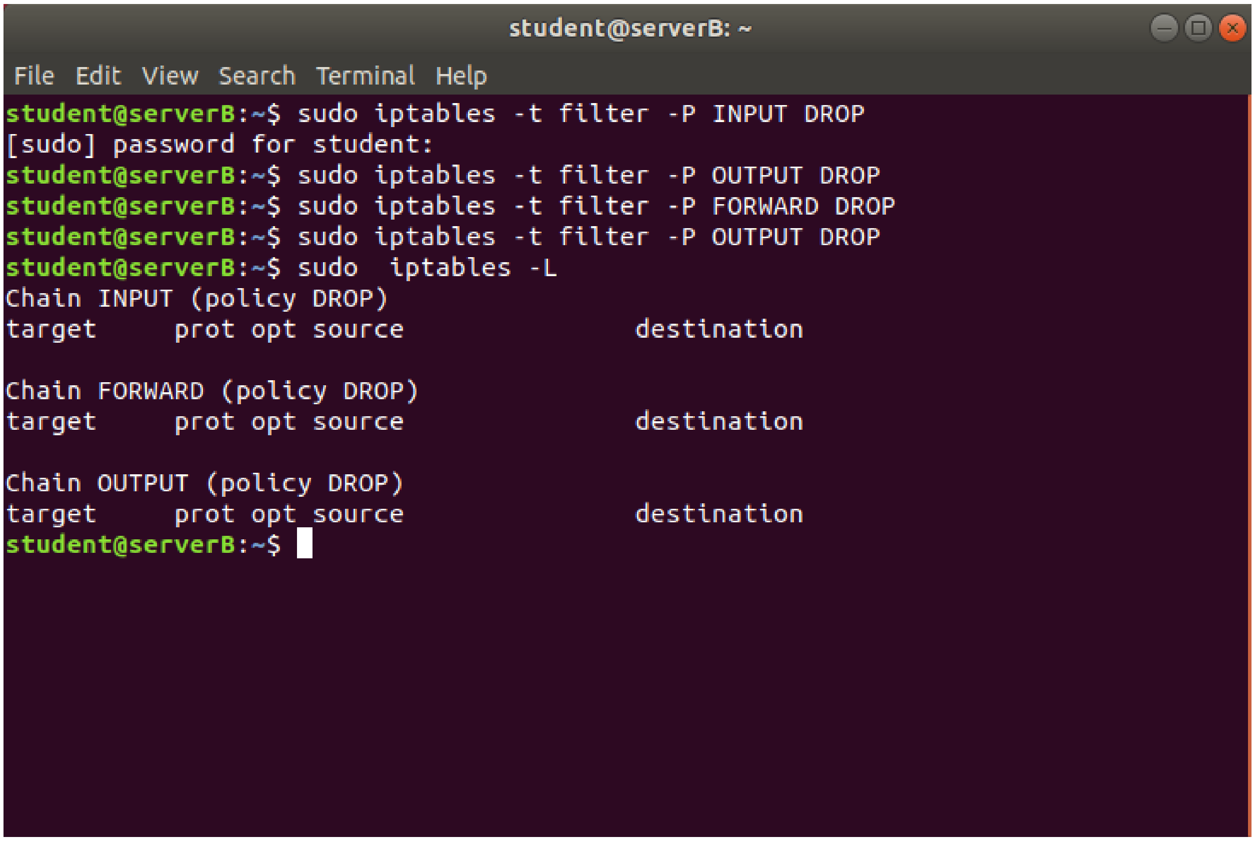
On both server A and B, I ran the following commands to change iptables default policy to DROP



```
sudo iptables -t filter -P INPUT DROP
sudo iptables -t filter -P OUTPUT DROP
sudo iptables -t filter -P FORWARD DROP
```



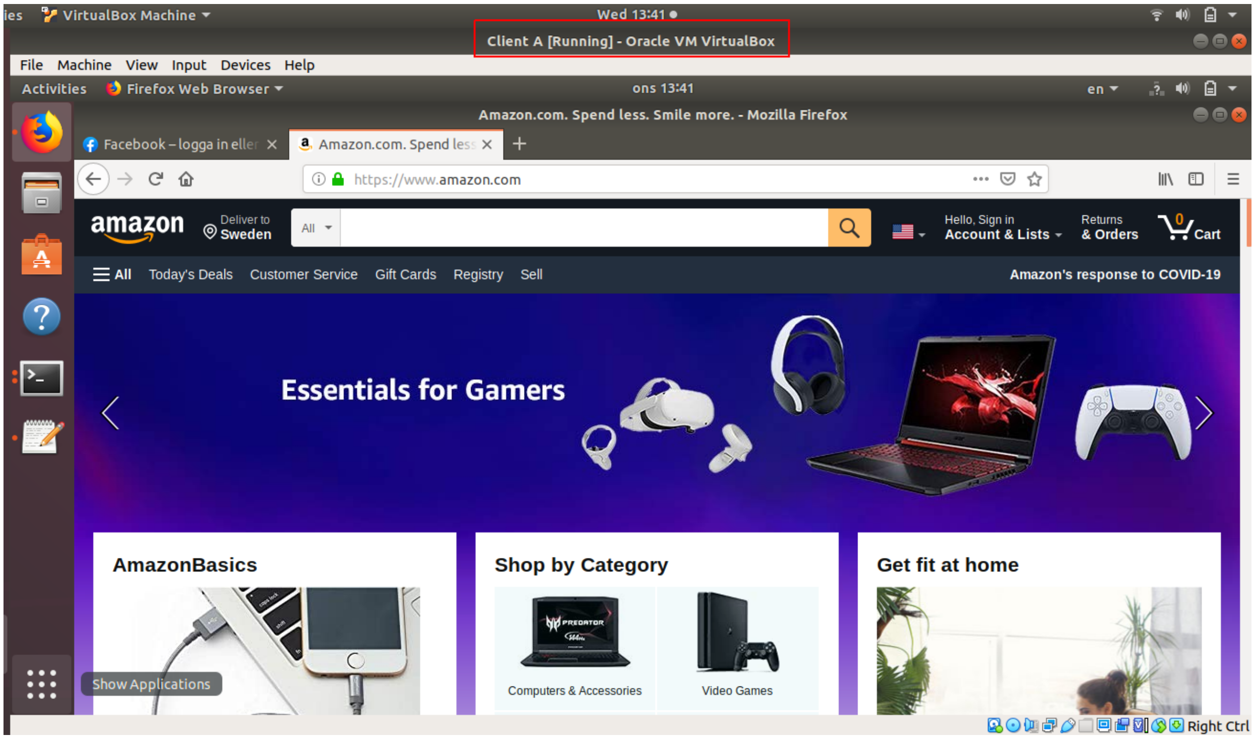
```
student@serverA:~$ sudo  iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy DROP)
```



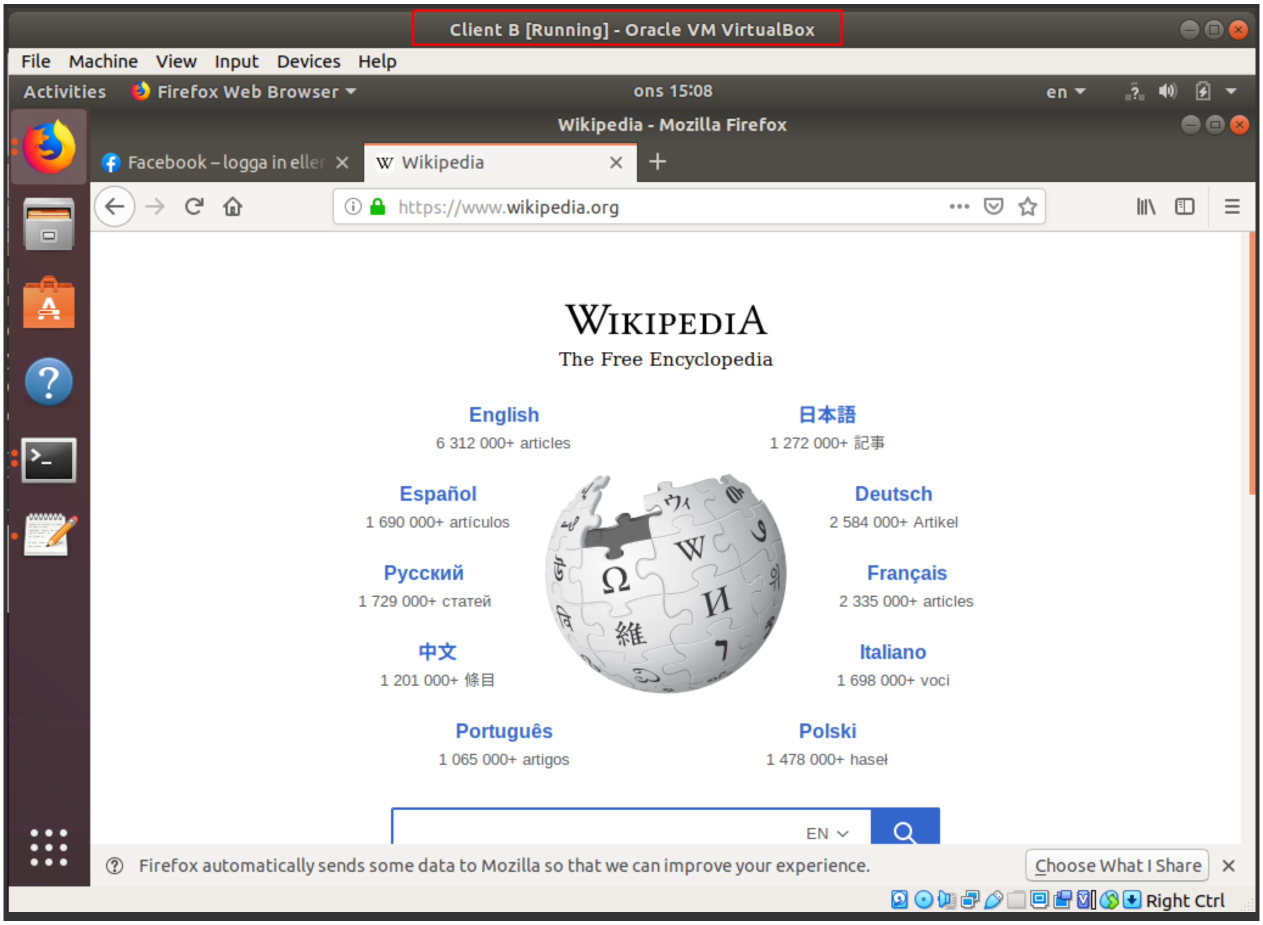
```
student@serverB:~$ sudo iptables -t filter -P INPUT DROP
[sudo] password for student:
student@serverB:~$ sudo iptables -t filter -P OUTPUT DROP
student@serverB:~$ sudo iptables -t filter -P FORWARD DROP
student@serverB:~$ sudo iptables -t filter -P OUTPUT DROP
student@serverB:~$ sudo  iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
student@serverB:~$ 
```

`gateway 192.168.60.100` and `gateway 192.168.80.100` are added to the file `/etc/network/interfaces` on client A and client B respectively.

Afterwards, `sudo ./firewall.sh` is ran on server A followed by `sudo sysctl -w net.ipv4.ip_forward=1` and `sudo sysctl -p`
After these commands, internet access is now available on client A via server A.

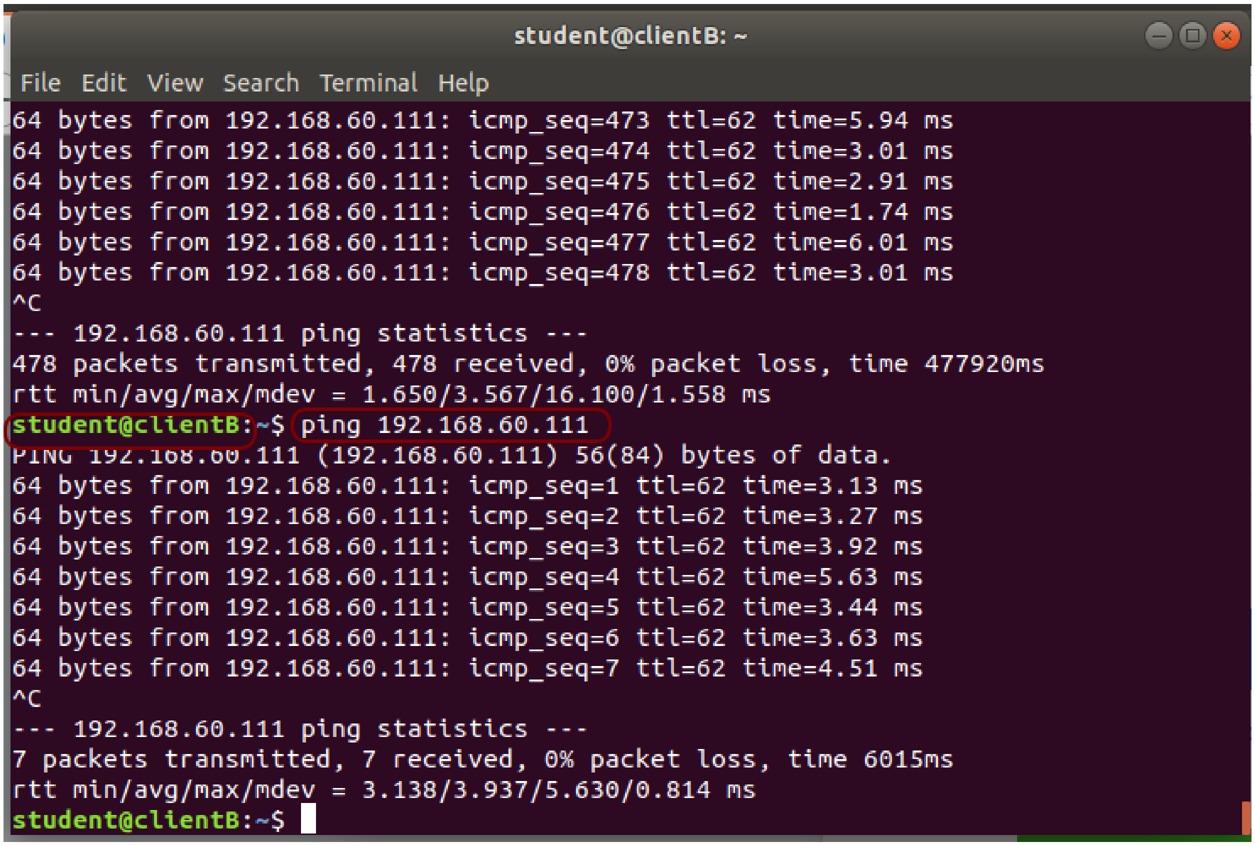


To enable client B internet access, the firewalls.sh file of server A is transferred to server B and all ip addresses are modified to correspond with the server B’s environment.Similarly as in server A, `sudo ./firewall.sh` is ran on server B followed by `sudo sysctl -w net.ipv4.ip_forward=1` and `sudo sysctl -p`
After these commands, internet access is now available on client B via server B



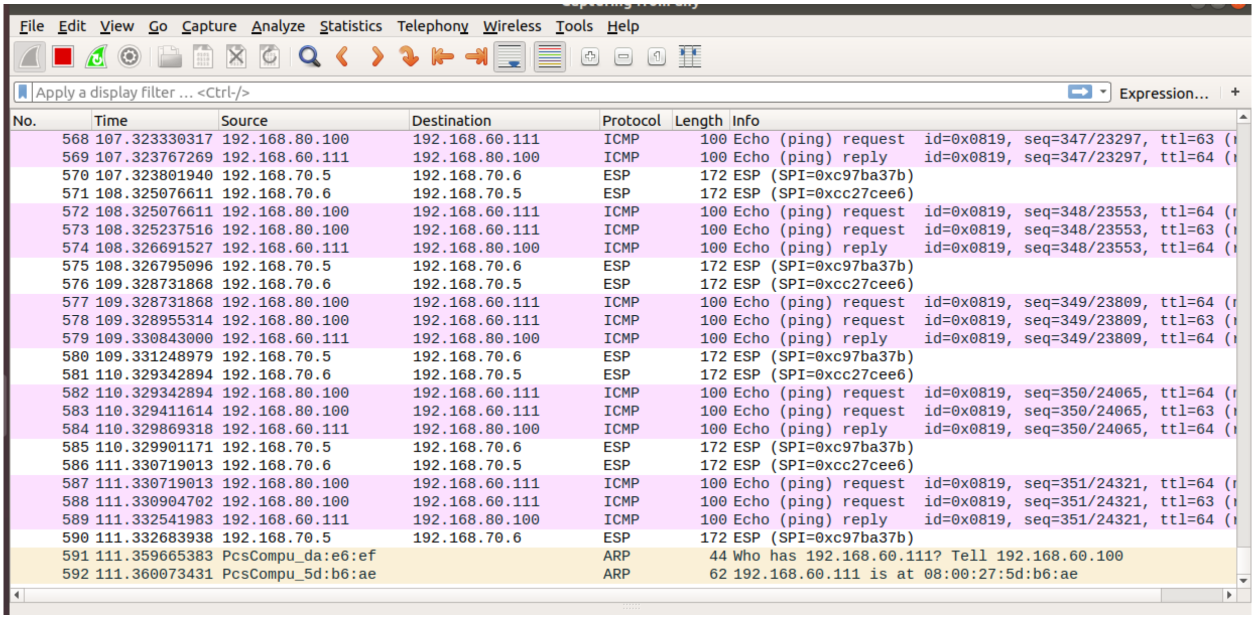
Firefox automatically sends some data to Mozilla so that we can improve your experience.

We can now as well ping client A from client B and vice versa.



```
64 bytes from 192.168.60.111: icmp_seq=473 ttl=62 time=5.94 ms
64 bytes from 192.168.60.111: icmp_seq=474 ttl=62 time=3.01 ms
64 bytes from 192.168.60.111: icmp_seq=475 ttl=62 time=2.91 ms
64 bytes from 192.168.60.111: icmp_seq=476 ttl=62 time=1.74 ms
64 bytes from 192.168.60.111: icmp_seq=477 ttl=62 time=6.01 ms
64 bytes from 192.168.60.111: icmp_seq=478 ttl=62 time=3.01 ms
^C
--- 192.168.60.111 ping statistics ---
478 packets transmitted, 478 received, 0% packet loss, time 477920ms
rtt min/avg/max/mdev = 1.650/3.567/16.100/1.558 ms
student@clientB:~$ ping 192.168.60.111
PING 192.168.60.111 (192.168.60.111) 56(84) bytes of data.
64 bytes from 192.168.60.111: icmp_seq=1 ttl=62 time=3.13 ms
64 bytes from 192.168.60.111: icmp_seq=2 ttl=62 time=3.27 ms
64 bytes from 192.168.60.111: icmp_seq=3 ttl=62 time=3.92 ms
64 bytes from 192.168.60.111: icmp_seq=4 ttl=62 time=5.63 ms
64 bytes from 192.168.60.111: icmp_seq=5 ttl=62 time=3.44 ms
64 bytes from 192.168.60.111: icmp_seq=6 ttl=62 time=3.63 ms
64 bytes from 192.168.60.111: icmp_seq=7 ttl=62 time=4.51 ms
^C
--- 192.168.60.111 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6015ms
rtt min/avg/max/mdev = 3.138/3.937/5.630/0.814 ms
student@clientB:~$
```

Observing traffic on wireshark shows that the communication is transported over the IPsec tunnel established between Server A and Server B



File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
+

```
No.     Time          Source             Destination      Protocol Length Info
    568 107.323330317 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=347/23297, ttl=63 (r
    569 107.323767269 192.168.60.111     192.168.80.100   ICMP      100 Echo (ping) reply    id=0x0819, seq=347/23297, ttl=64 (r
    570 107.323801940 192.168.70.5       192.168.70.6     ESP       172 ESP (SPI=0xc97ba37b)
    571 108.325076611 192.168.70.6       192.168.70.5     ESP       172 ESP (SPI=0xcc27cee6)
    572 108.325076611 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=348/23553, ttl=64 (r
    573 108.325237516 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=348/23553, ttl=63 (r
    574 108.326691527 192.168.60.111     192.168.80.100   ICMP      100 Echo (ping) reply    id=0x0819, seq=348/23553, ttl=64 (r
    575 108.326795096 192.168.70.5       192.168.70.6     ESP       172 ESP (SPI=0xc97ba37b)
    576 109.328731868 192.168.70.6       192.168.70.5     ESP       172 ESP (SPI=0xcc27cee6)
    577 109.328731868 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=349/23809, ttl=64 (r
    578 109.328955314 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=349/23809, ttl=63 (r
    579 109.330843000 192.168.60.111     192.168.80.100   ICMP      100 Echo (ping) reply    id=0x0819, seq=349/23809, ttl=64 (r
    580 109.331248979 192.168.70.5       192.168.70.6     ESP       172 ESP (SPI=0xc97ba37b)
    581 110.329342894 192.168.70.6       192.168.70.5     ESP       172 ESP (SPI=0xcc27cee6)
    582 110.329342894 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=350/24065, ttl=64 (r
    583 110.329411614 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=350/24065, ttl=63 (r
    584 110.329869318 192.168.60.111     192.168.80.100   ICMP      100 Echo (ping) reply    id=0x0819, seq=350/24065, ttl=64 (r
    585 110.329901171 192.168.70.5       192.168.70.6     ESP       172 ESP (SPI=0xc97ba37b)
    586 111.330719013 192.168.70.6       192.168.70.5     ESP       172 ESP (SPI=0xcc27cee6)
    587 111.330719013 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=351/24321, ttl=64 (r
    588 111.330904702 192.168.80.100     192.168.60.111   ICMP      100 Echo (ping) request  id=0x0819, seq=351/24321, ttl=63 (r
    589 111.332541983 192.168.60.111     192.168.80.100   ICMP      100 Echo (ping) reply    id=0x0819, seq=351/24321, ttl=64 (r
    590 111.332683938 192.168.70.5       192.168.70.6     ESP       172 ESP (SPI=0xc97ba37b)
    591 111.359665383 PcsCompu_da:e6:ef                   ARP        44 Who has 192.168.60.111? Tell 192.168.60.100
    592 111.360073431 PcsCompu_5d:b6:ae                   ARP        62 192.168.60.111 is at 08:00:27:5d:b6:ae
```