Student ID/Name:Jerry - 24560409
Student ID/Name:Chun Hin Carven Suen -13266578
Student ID/Name:Johristein - 14189094

Project Description Part 1

1. SSH to localhost
    - Install and run the SSH server in the VM
      OpenSSH Server | Ubuntu
      *Install the client application

```
crypto@crypto:~$ sudo apt install openssh-client
[sudo] password for crypto:
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-client is already the newest version (1:7.6p1-4).
0 to upgrade, 0 to newly install, 0 to remove and 0 not to upgrade.
```

      *Install the server application

```
crypto@crypto:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 to upgrade, 4 to newly install, 0 to remove and 0 not to upgrade.
Need to get 637 kB of archives.
After this operation, 5,321 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

    - Check the ssh configuration
      How to verify the validity of an SSH server's configuration (simplified.guide)
      (maybe)
    - SSH to localhost with your username and password

```
crypto@crypto:~$ ssh localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:9lLygJxWWObuSGmWtk6UHBCU9erNqSLgEnYYNguNvc4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
crypto@localhost's password:
Welcome to Ubuntu 18.04 LTS (GNU/Linux 4.15.0-20-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

crypto@crypto:~$ exit
logout
Connection to localhost closed.
crypto@crypto:~$
```

2. SSH to localhost with public-key authentication
   - Generate an RSA key pair and set a passphrase to protect it (passphrase=crypto)

```
crypto@crypto:~/Public$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/crypto/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/crypto/.ssh/id_rsa.
Your public key has been saved in /home/crypto/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:+MzSZ6nlDrF0egWgNyG83LyhyMU+bzEfAKSUJTcMMj0 crypto@crypto
The key's randomart image is:
+---[RSA 2048]----+
|  o.+B* o        |
|   +E+++ o       |
|    .+.=o .      |
|      =o=. .     |
|   . +..S+. .    |
|    o +*+=.o     |
|      .oO+*.     |
|       .oO.      |
|       ...o      |
+----[SHA256]-----+
```

   - Check the generated private key

```
crypto@crypto:~/.ssh$ cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,617E085F8D85A5D20BF3D0FA48D05946

aPgLJ0uIrk1SW6RaC7EP7t1hxKaup8rtCRUzwK6bPdsb9qopj0su/LMK2FXuFtaY
9JqzAaeudXIQ2KCOA5B1EpyXatCCa1YMI5D11XjIr+CpLXEF7Pud4c1ZwJHQfuoZ
z0DXznp/T10YQSabBYwTSuLWb4Mn+/KYTPj80T/nUNtBsGfj/2xwpNedAOF3nmSp
tCMlxG8qpm6G5ikIbZwM22rOWvUPSeOZPYgsYKTVkB5hPVHmp8S0wzMTf3YfnOiA
yveDDLUuwrP068jJiecO0n3IcDGkd5/jD+rtPbtZuo8mt4nJb69T+WSANxcMDddb
Le3YE/PM+f6lnE8GVGmaB7hiwyL3/+07x8mbV40Wq+5aEvXARWw+RYu7XreDFxYQ
AmYnj5zFiHr4bJVg4G4NTGCnGaZ+loj8Icq354gkcDkzoJaluDeCvACimRhvbABy
v1o9XAjGKbmrEhyl77LuOehgKJiOKOnecD8fvoTigUSC836SS8E14++Ela9WdDav
b6fA7GuXPxdDXZOGRixsmOQCoHl0BySrYVOa447HrZeSMlpu1wy6qb3HiDiTtN3G
oXpNYI6yyMaKeIQ5miQC+i7xh9m8agSSBJYIXupB5YDjYXBqMRXxbejHVGcbiSBg
gxYRTOxq/I4ryHlXb4GLwMS0QbGc85Yxf4ZEfbPz1bxP5xTqUNB6I4fdElnRzGB+
Pb03O3caTKizYtZYjm7BZUG2YrYRmZgK/rUA+818/fHcOzksyTPg1MF6qe9MuaFl
48/RDRUB1LhChuT6GyCg1t0x8XWD7W+SAoQWfC5ms6nHkATuDkg5gEXkhY/qP224
Izki26PSB51trlpqA4QgJfQWa0HS8Nid1L/rdiXwWeLph63r3q3GEGGFaA22y7q2
yrGpq/ea6uD89X7gPsVlN1cyVdHfl/Sjnia+7TLQI9H17IxpQFFOjg4sth3L92yu
oZDuXjBCDpZooiXsKHHP+OluUzjXTNImTu7KwhjiNyhPlKke3FSwmQCrk+zEKeL8
XQVnv05X4Lm/NSuuxJgSv5IdbiDYjiHJf7HqVzoMxN8qZTNuiK1maLm4REswpG+Z
iOdzZ5S/TY5boX5ZQTpwuurGSv3/wt5ysfVEtD5C6HjMREWe7u2FYH3Qpd/9RBHe
```

- Configure the ssh server accept the RSA key in authentication
  Now copy the id_rsa.pub file to the remote host (::1) and append it to ~/.ssh/authorized_keys by entering:

```
crypto@crypto:~/.ssh$ ssh-copy-id crypto@::1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/h
ome/crypto/.ssh/id_rsa.pub"
The authenticity of host '::1 (::1)' can't be established.
ECDSA key fingerprint is SHA256:9lLygJxWWObuSGmWtk6UHBCU9erNqSLgE
nYYNguNvc4.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key
(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if
 you are prompted now it is to install the new keys
crypto@::1's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'crypto@::1'"
and check to make sure that only the key(s) you wanted were added
.

crypto@crypto:~/.ssh$
```

Now to test it by logging into the machine:

```
crypto@crypto:~/.ssh$ ssh 'crypto@::1'
Enter passphrase for key '/home/crypto/.ssh/id_rsa':
Welcome to Ubuntu 18.04 LTS (GNU/Linux 4.15.0-20-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate
at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.

Last login: Fri Apr  5 12:26:50 2024 from ::1
crypto@crypto:~$ exit
logout
Connection to ::1 closed.
crypto@crypto:~/.ssh$
```

Or this:

```
crypto@crypto:~/.ssh$ ssh -i rsa localhost
Warning: Identity file rsa not accessible: No such file or directory.
Enter passphrase for key '/home/crypto/.ssh/id_rsa':
Welcome to Ubuntu 18.04 LTS (GNU/Linux 4.15.0-20-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
0 updates are security updates.

Last login: Fri Apr  5 12:30:58 2024 from ::1
crypto@crypto:~$ exit
logout
```

Project Description Part 2:

Set up the Certificate Authority on the VM (digital certificate)

- Create root CA, and use this to issue certificates for others(servers)

Using the following information to create the CA's key

- Country Name (2 letter code) [AU]:AU

- State or Province Name (full name) [Some-State]:NSW
- Locality Name (eg, city) []:SYD
- Organization Name (eg, company) [Internet Widgits Pty Ltd]:UTS
- Organizational Unit Name (eg, section) []:FEIT
- Common Name (e.g. server FQDN or YOUR name) []:utscrypto.com.au
- Email Address []:root@utscrypto.com.au

```
crypto@crypto:~$ mkdir certs
crypto@crypto:~$ cd certs
crypto@crypto:~/certs$ openssl genrsa -aes128 -out ca.key 2048
Generating RSA private key, 2048 bit long modulus
.............................................+++++
.........+++++
e is 65537 (0x010001)
Enter pass phrase for ca.key:
Verifying - Enter pass phrase for ca.key:
crypto@crypto:~/certs$ openssl req -x509 -new -nodes -key ca.key -sha256 -days
 1826 -out ca.crt
Enter pass phrase for ca.key:
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) [AU]:AU
State or Province Name (full name) [Some-State]:NSW
Locality Name (eg, city) []:SYD
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UTS
Organizational Unit Name (eg, section) []:FEIT
Common Name (e.g. server FQDN or YOUR name) []:utscrypto.com.au
Email Address []:utscrypto@netsec.com.au
crypto@crypto:~/certs$ openssl rsa -in ca.key .text
rsa: Use -help for summary.
crypto@crypto:~/certs$ openssl rsa -in ca.key -text
Enter pass phrase for ca.key:
RSA Private-Key: (2048 bit, 2 primes)
modulus:
    00:bf:aa:bc:40:a2:47:f7:22:0a:af:ea:4a:a9:3e:
    f4:97:1e:42:9c:36:ea:bc:7b:57:2d:2f:64:e7:da:
    3c:12:02:d4:6c:c2:9c:89:bf:ba:56:c9:fc:ba:f0:
    b9:59:3b:a1:c3:4b:ea:79:0a:64:2c:bd:f7:fb:fb:
```

Expected Result

```
crypto@crypto:~/certs$ openssl x509 -in ca.crt -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            d2:bf:a4:82:6f:60:f5:d6
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = AU, ST = NSW, L = SYD, O = UTS, OU = FEIT, CN = utscrypto.
com.au, emailAddress = utscrypto@netsec.com.au
        Validity
            Not Before: Apr  6 05:23:54 2024 GMT
            Not After : Apr  6 05:23:54 2029 GMT
        Subject: C = AU, ST = NSW, L = SYD, O = UTS, OU = FEIT, CN = utscrypto
.com.au, emailAddress = utscrypto@netsec.com.au
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:bf:aa:bc:40:a2:47:f7:22:0a:af:ea:4a:a9:3e:
                    f4:97:1e:42:9c:36:ea:bc:7b:57:2d:2f:64:e7:da:
```

Project Description Part 3

Generate public/private key pair for the HTTPS server

```
crypto@crypto:~/certs$ openssl genpkey -algorithm RSA -out server.key 2048
genpkey: Use -help for summary.
crypto@crypto:~/certs$ openssl genpkey -algorithm RSA -out server.key -pkeyopt
 rsa_keygen_bits:2048
.........+++++
...........+++++
crypto@crypto:~/certs$ openssl rsa -aes256 -in server.key -out serverenc.key
writing RSA key
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

```
crypto@crypto:~/certs$ openssl rsa -in serverenc.key -text
Enter pass phrase for serverenc.key:
RSA Private-Key: (2048 bit, 2 primes)
modulus:
```

Generate a Certificate Signing Request

```
crypto@crypto:~/certs$ openssl req -new -newkey rsa:2048 -nodes -keyout server
.key -out server.csr
Generating a 2048 bit RSA private key
.....................................................+++++
.....................................................................
.........+++++
writing new private key to 'server.key'
-----
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) [AU]:AU
State or Province Name (full name) [Some-State]:NSW
Locality Name (eg, city) []:SYD
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UTS
Organizational Unit Name (eg, section) []:FEIT
Common Name (e.g. server FQDN or YOUR name) []:utscrypto.com.au
Email Address []:root@utscrypto.com.au

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
crypto@crypto:~/certs$ openssl req -in sserver.csr -noout -text
req: Cannot open input file sserver.csr, No such file or directory
req: Use -help for summary.
crypto@crypto:~/certs$ openssl req -in server.csr -noout -text
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = AU, ST = NSW, L = SYD, O = UTS, OU = FEIT, CN = utscrypto
.com.au, emailAddress = root@utscrypto.com.au
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
```

Sign the Certificate and Expected Result

```
crypto@crypto:~/certs$ openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.k
ey -CAcreateserial -out server.crt -days 365
Signature ok
subject=C = AU, ST = NSW, L = SYD, O = UTS, OU = FEIT, CN = utscrypto.com.au,
emailAddress = root@utscrypto.com.au
Getting CA Private Key
Enter pass phrase for ca.key:
crypto@crypto:~/certs$ openssl x509 -in server.crt -text -noout
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            f4:4c:04:f6:3c:47:14:b4
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = AU, ST = NSW, L = SYD, O = UTS, OU = FEIT, CN = utscrypto.
com.au, emailAddress = utscrypto@netsec.com.au
        Validity
            Not Before: Apr  6 10:16:21 2024 GMT
            Not After : Apr  6 10:16:21 2025 GMT
        Subject: C = AU, ST = NSW, L = SYD, O = UTS, OU = FEIT, CN = utscrypto
.com.au, emailAddress = root@utscrypto.com.au
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
```

Project Description Part 4

Merge server.key and server.crt to a single file named server.pem

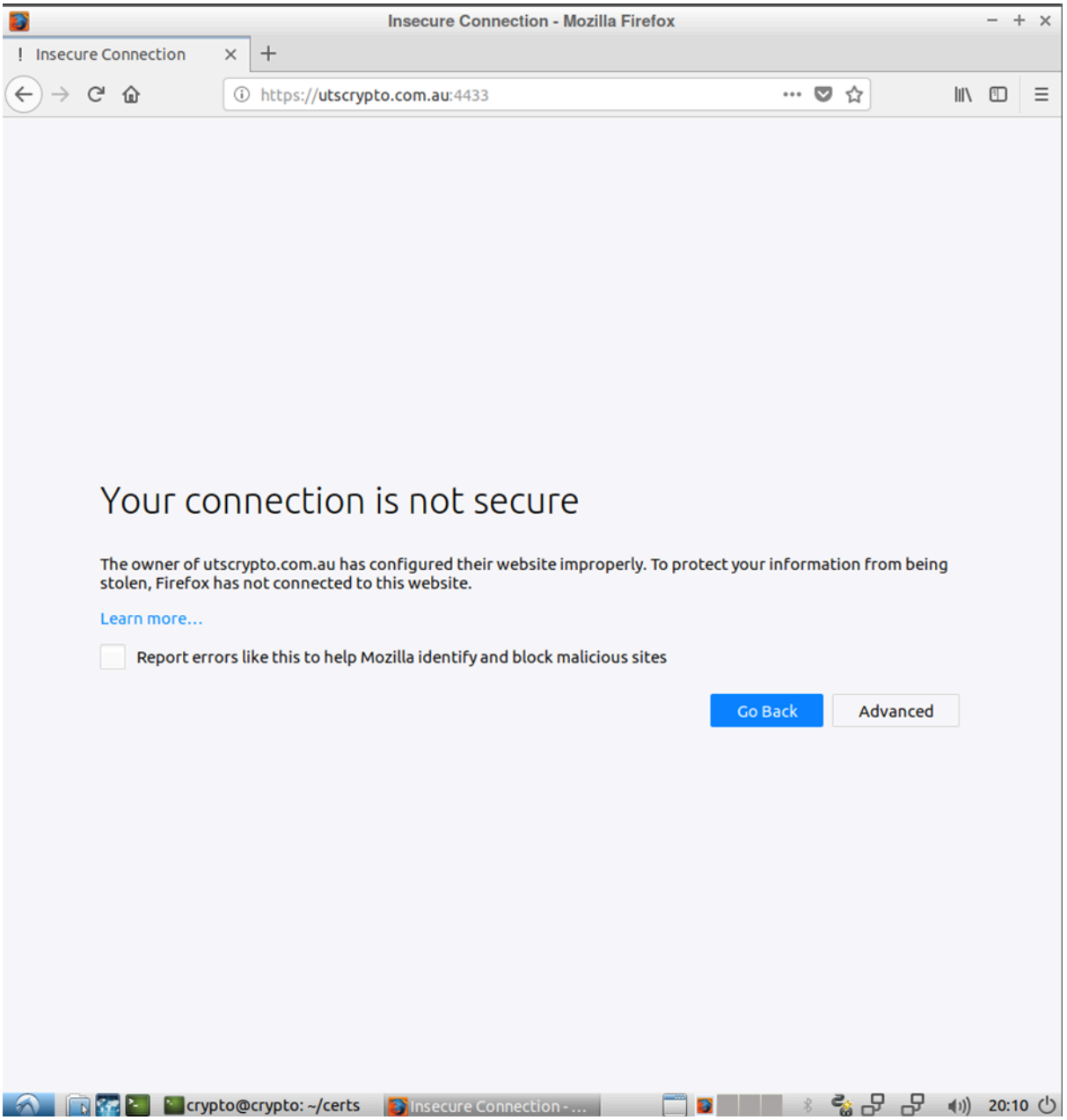
```
crypto@crypto:~/certs$ cat server.key server.crt > server.pem
```

Add the entry 127.0.0.1 utscrypto.com.au to /etc/hosts

```
crypto@crypto:~/certs$ sudo nano /etc/hosts
[sudo] password for crypto:
crypto@crypto:~/certs$ /etc/hosts
bash: /etc/hosts: Permission denied
crypto@crypto:~/certs$ cat /etc/hosts
127.0.0.1       localhost
127.0.1.1       crypto
127.0.0.1       utscrypto.com.au
```

```
crypto@crypto:~/certs$ openssl s_server -cert server.pem -www
Using default temp DH parameters
ACCEPT
```

Expected Result



After import CA


```
s_server -cert server.pem -www
Secure Renegotiation IS supported
Ciphers supported in s_server binary
TLSv1.3    :TLS_AES_256_GCM_SHA384    TLSv1.3    :TLS_CHACHA20_POLY1305_SHA256
TLSv1.3    :TLS_AES_128_GCM_SHA256    TLSv1.2    :ECDHE-ECDSA-AES256-GCM-SHA384
TLSv1.2    :ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2    :DHE-RSA-AES256-GCM-SHA384
TLSv1.2    :ECDHE-ECDSA-CHACHA20-POLY1305 TLSv1.2    :ECDHE-RSA-CHACHA20-POLY1305
TLSv1.2    :DHE-RSA-CHACHA20-POLY1305 TLSv1.2    :ECDHE-ECDSA-AES128-GCM-SHA256
TLSv1.2    :ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2    :DHE-RSA-AES128-GCM-SHA256
TLSv1.2    :ECDHE-ECDSA-AES256-SHA384 TLSv1.2    :ECDHE-RSA-AES256-SHA384
TLSv1.2    :DHE-RSA-AES256-SHA256     TLSv1.2    :ECDHE-ECDSA-AES128-SHA256
TLSv1.2    :ECDHE-RSA-AES128-SHA256   TLSv1.2    :DHE-RSA-AES128-SHA256
TLSv1.0    :ECDHE-ECDSA-AES256-SHA    TLSv1.0    :ECDHE-RSA-AES256-SHA
SSLv3      :DHE-RSA-AES256-SHA        TLSv1.0    :ECDHE-ECDSA-AES128-SHA
TLSv1.0    :ECDHE-RSA-AES128-SHA      SSLv3      :DHE-RSA-AES128-SHA
TLSv1.2    :RSA-PSK-AES256-GCM-SHA384 TLSv1.2    :DHE-PSK-AES256-GCM-SHA384
TLSv1.2    :RSA-PSK-CHACHA20-POLY1305 TLSv1.2    :DHE-PSK-CHACHA20-POLY1305
TLSv1.2    :ECDHE-PSK-CHACHA20-POLY1305 TLSv1.2    :AES256-GCM-SHA384
TLSv1.2    :PSK-AES256-GCM-SHA384     TLSv1.2    :PSK-CHACHA20-POLY1305
TLSv1.2    :RSA-PSK-AES128-GCM-SHA256 TLSv1.2    :DHE-PSK-AES128-GCM-SHA256
TLSv1.2    :AES128-GCM-SHA256         TLSv1.2    :PSK-AES128-GCM-SHA256
TLSv1.2    :AES256-SHA256             TLSv1.2    :AES128-SHA256
TLSv1.0    :ECDHE-PSK-AES256-CBC-SHA384 TLSv1.0    :ECDHE-PSK-AES256-CBC-SHA
SSLv3      :SRP-RSA-AES-256-CBC-SHA   SSLv3      :SRP-AES-256-CBC-SHA
TLSv1.0    :RSA-PSK-AES256-CBC-SHA384 TLSv1.0    :DHE-PSK-AES256-CBC-SHA384
SSLv3      :RSA-PSK-AES256-CBC-SHA    SSLv3      :DHE-PSK-AES256-CBC-SHA
SSLv3      :AES256-SHA                TLSv1.0    :PSK-AES256-CBC-SHA384
SSLv3      :PSK-AES256-CBC-SHA        TLSv1.0    :ECDHE-PSK-AES128-CBC-SHA256
TLSv1.0    :ECDHE-PSK-AES128-CBC-SHA  SSLv3      :SRP-RSA-AES-128-CBC-SHA
SSLv3      :SRP-AES-128-CBC-SHA       TLSv1.0    :RSA-PSK-AES128-CBC-SHA256
TLSv1.0    :DHE-PSK-AES128-CBC-SHA256 SSLv3      :RSA-PSK-AES128-CBC-SHA
SSLv3      :DHE-PSK-AES128-CBC-SHA    SSLv3      :AES128-SHA
TLSv1.0    :PSK-AES128-CBC-SHA256     SSLv3      :PSK-AES128-CBC-SHA
---
Ciphers common between both SSL end points:
ECDHE-ECDSA-AES128-GCM-SHA256 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-ECDSA-CHACHA20-POLY1305
ECDHE-RSA-CHACHA20-POLY1305 ECDHE-ECDSA-AES256-GCM-SHA384 ECDHE-RSA-AES256-GCM-SHA384
ECDHE-ECDSA-AES256-SHA     ECDHE-ECDSA-AES128-SHA     ECDHE-RSA-AES128-SHA
ECDHE-RSA-AES256-SHA       DHE-RSA-AES128-SHA         DHE-RSA-AES256-SHA
AES128-SHA                 AES256-SHA
Signature Algorithms: ECDSA+SHA256:ECDSA+SHA384:ECDSA+SHA512:0x04+SHA256:0x05+SHA384:0x06+SHA512:RSA+SHA256:RSA+SHA384:RSA+SHA512:ECDSA+SHA
Supported Elliptic Curves: X25519:P-256:P-384:P-521
Shared Elliptic curves: X25519:P-256:P-384:P-521
---
New, TLSv1.2, Cipher is ECDHE-RSA-AES128-GCM-SHA256
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES128-GCM-SHA256
    Session-ID:
    Session-ID-ctx: 01000000
    Master-Key: BC0E2759D8384F7704B938AD5AD9996D276746CB6992C4C74873A3D8093E4A499E4D0F9448AA406A2F10000B4CF150F7
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    Start Time: 1712398804
    Timeout   : 7200 (sec)
    Verify return code: 0 (ok)
    Extended master secret: yes
---
   0 items in the session cache
```