**At server side, administrator need to generate CA file, server certificate file, client certificate file**

1. Administrator need to upload CA file, server certificate file and server private key to BMC
2. Administrator will use CA file to check if the user is trusted or not
3. User will use authentication and client private key to login

**Generate CA, server certificate , client certificate files:**

common:

mkdir -p /CA

cd /CA

mkdir certs crl newcerts private

chmod 700 private

touch index.txt

echo 1000 > serial

echo 1000 > crlnumber

CA:

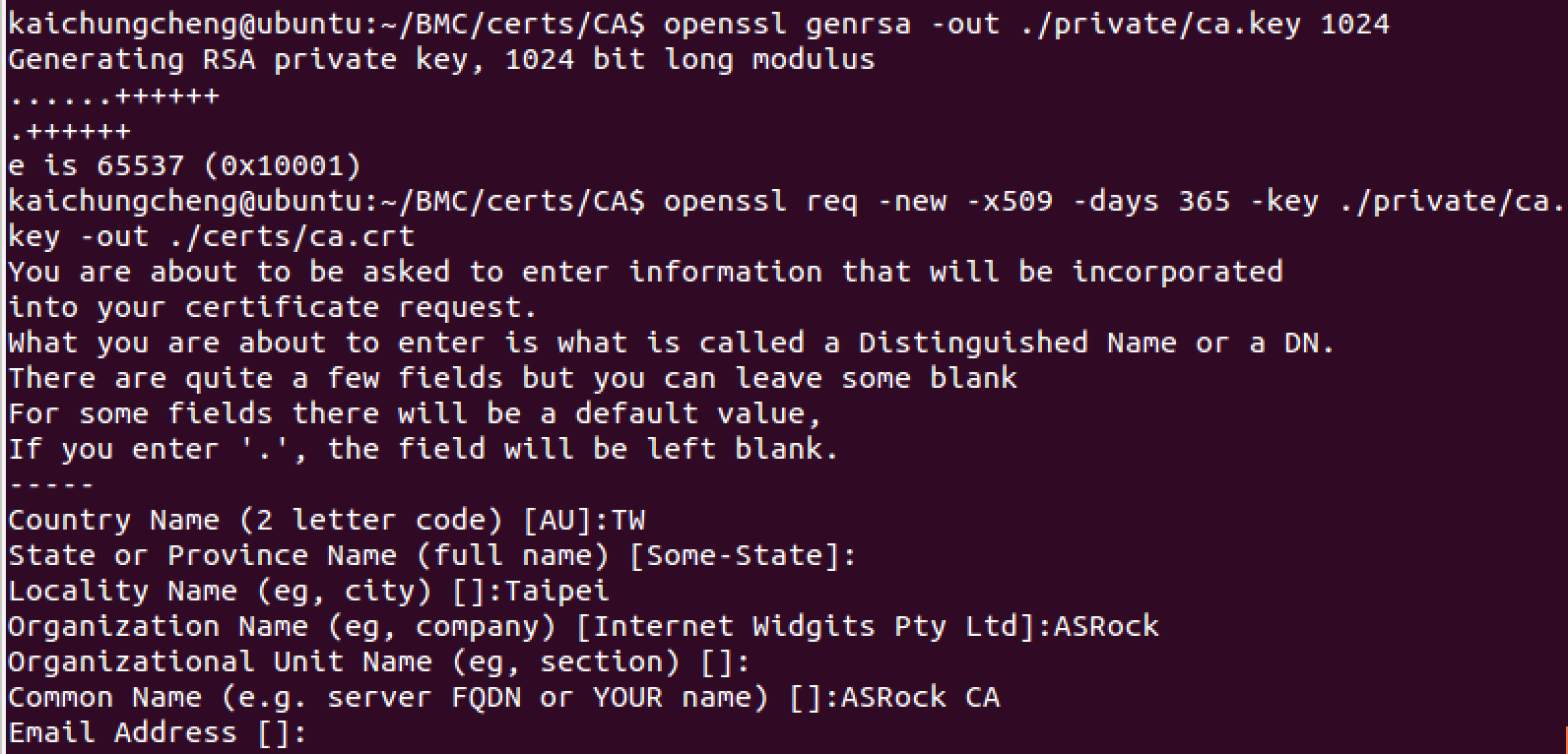
##Generate private key

openssl genrsa -out ./private/ca.key 1024

##Generate certificate file by private key (contain public key), x509 is the most common form

openssl req -new -x509 -days 365 -key ./private/ca.key -out ./certs/ca.crt

cat ./certs/ca.crt > ./certs/ca.pem



**Server certificate:**

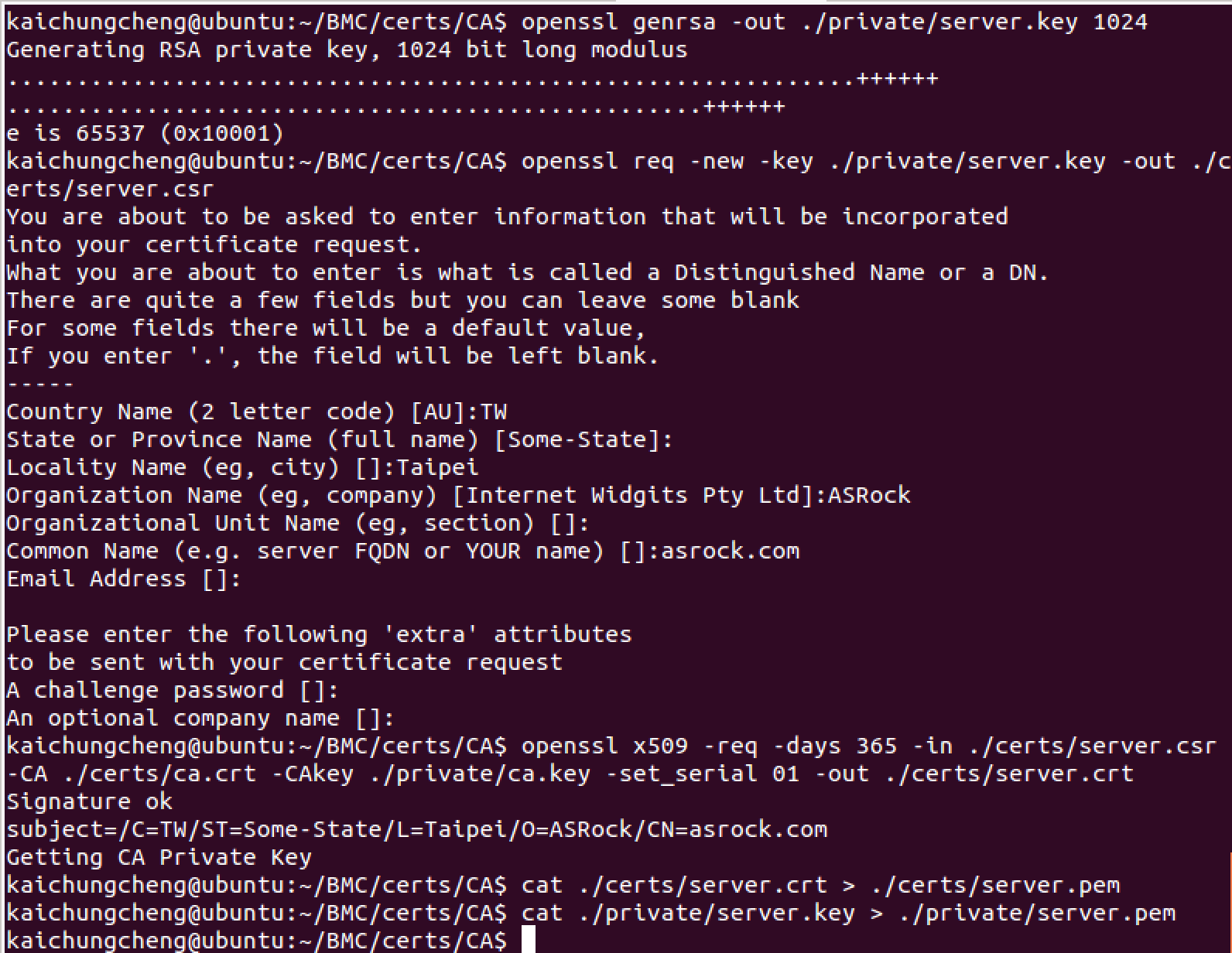
openssl genrsa -out ./private/server.key 1024

openssl req -new -key ./private/server.key -out ./certs/server.csr

openssl x509 -req -days 365 -in ./certs/server.csr -CA ./certs/ca.crt -CAkey ./private/ca.key -set\_serial 01 -out ./certs/server.crt

cat ./certs/server.crt > ./certs/server.pem

cat ./private/server.key > ./private/server\_key.pem



**Client certificate:**

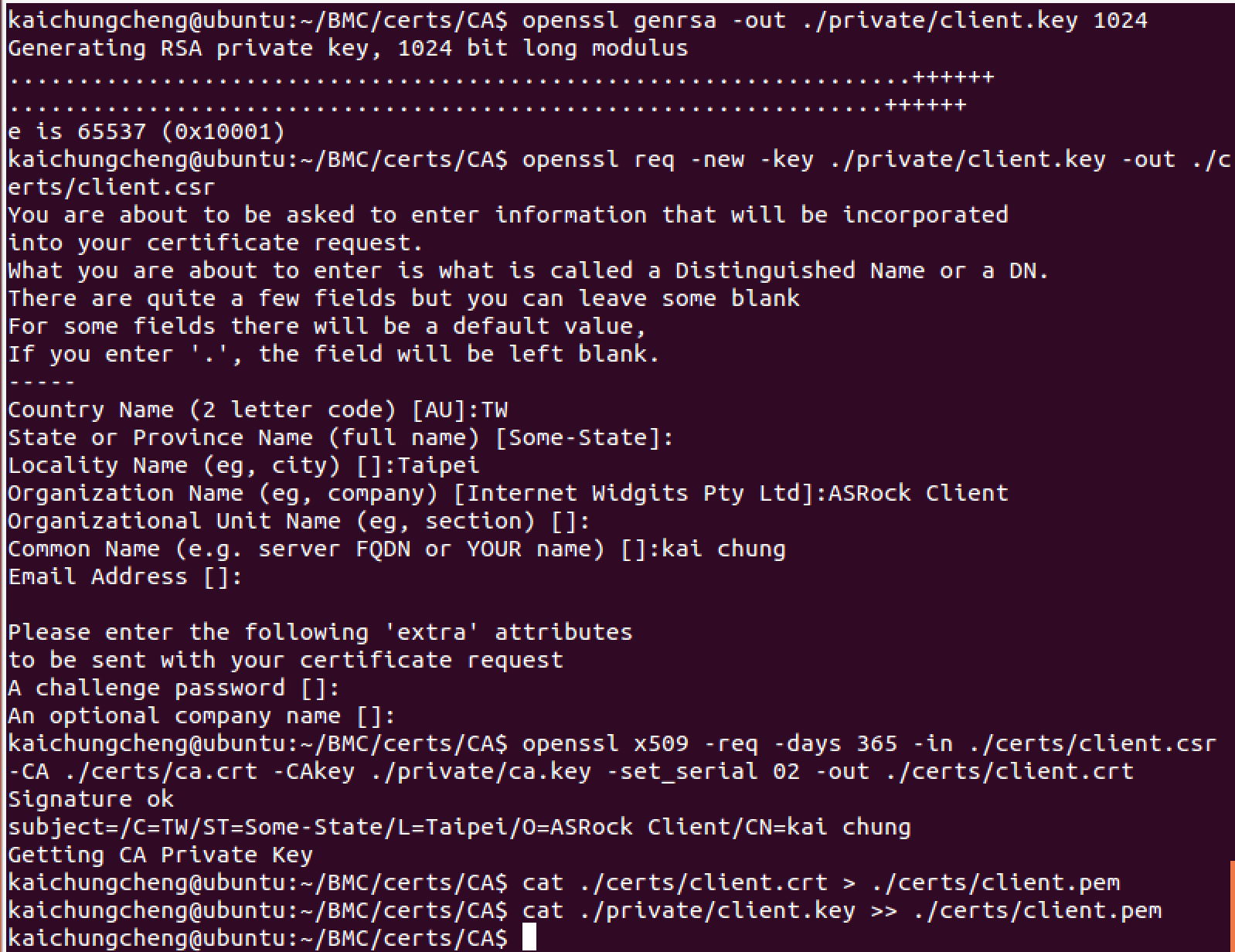
openssl genrsa -out ./private/client.key 1024

openssl req -new -key ./private/client.key -out ./certs/client.csr

openssl x509 -req -days 365 -in ./certs/client.csr -CA ./certs/ca.crt -CAkey ./private/ca.key -set\_serial 02 -out ./certs/client.crt

cat ./certs/client.crt > ./certs/client.pem

cat ./private/client.key >> ./certs/client.pem



Administrator upload ca.pem, server.pem, server\_key.pem

1.login

curl -X POST -d "WEBVAR\_USERNAME=admin&WEBVAR\_PASSWORD=admin" "<https://192.168.2.100/rpc/WEBSES/create.asp>" -k -c ./cookie1 -v

2.then upload ca.pem, server.pem, server\_key.pem

curl -F "[cacertfile=@/home/kaichungcheng/BMC/certs/CA/certs/ca.pem](mailto:cacertfile=@/home/kaichungcheng/BMC/certs/CA/certs/ca.pem)" "<https://192.168.2.100/api/oem/maintenance/SSL/cacertfile>" -H "CSRFTOKEN:Na8E4BSx60" --cookie "SessionCookie=F3mRlPz5nEGIYNWyvPSQ1zoQRRzJNZul000" -v -k

curl -F "[certfile=@/home/kaichungcheng/BMC/certs/CA/certs/server.pem](mailto:certfile=@/home/kaichungcheng/BMC/certs/CA/certs/server.pem)" "<https://192.168.2.100/api/oem/maintenance/SSL/certfile>" -H "CSRFTOKEN:Na8E4BSx60" --cookie "SessionCookie=F3mRlPz5nEGIYNWyvPSQ1zoQRRzJNZul000" -v -k

curl -F "[privkey=@/home/kaichungcheng/BMC/certs/CA/private/server\_key.pem](mailto:privkey=@/home/kaichungcheng/BMC/certs/CA/private/server_key.pem)" "<https://192.168.2.100/api/oem/maintenance/SSL/privkey>" -H "CSRFTOKEN:Na8E4BSx60" --cookie "SessionCookie=F3mRlPz5nEGIYNWyvPSQ1zoQRRzJNZul000" -v –k

3.Verify and apply them. Note: https sever will restart after applying SSL crt files

curl "https://192.168.2.100/api/oem/maintenance/SSL/validate\_cert" -H "CSRFTOKEN:Na8E4BSx60" --cookie "SessionCookie=F3mRlPz5nEGIYNWyvPSQ1zoQRRzJNZul000" -v –k

After setting up server side, the end user can connect to the server through mutual port(default 4433) with client.pem.

Command:

curl "https://192.168.2.100:4433/api/oem/maintenance/session/certlogin" -k -c ./cookie1 -v --cert ./certs/client.pem

If the client.pem is invalid or there is no such file can be send by client side application, the server will block this connection.

------------------------------------------------------------------------------------------------------------------------------------------------------------