RSA Encryption and Signature Lab

Task 6: Manually Verifying an X.509 Certificate

Step 1: Download a certificate from a real web server.

```
[04/24/22]seed@VM:~/.../Task6$ openssl s_client -connect www.example.org:443 -showcerts
CONNECTED(00000003)
depth=1 C = US, O = DigiCert Inc, CN = DigiCert TLS RSA SHA256 2020 CA1
verify error:num=20:unable to get local issuer certificate
verify return:1
depth=0 C = US, ST = California, L = Los Angeles, O = Internet\C2\A0Corporation\C2\A0for\C2\A0Assigned\C2\A0Names\C2
\A0and\C2\A0Numbers, CN = www.example.org
verify return:1
---
Certificate chain
 0 s:C = US, ST = California, L = Los Angeles, O = Internet\C2\A0Corporation\C2\A0for\C2\A0Assigned\C2\A0Names\C2\A0
and\C2\A0Numbers, CN = www.example.org
   i:C = US, O = DigiCert Inc, CN = DigiCert TLS RSA SHA256 2020 CA1
-----BEGIN CERTIFICATE-----
MIIHRzCCBi+gAwIBAgIQD6pjEJMHvD1BSJJkDM1NmjANBgkqhkiG9w0BAQsFADBP
MQswCQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMSkwJwYDVQQDEyBE
aWdpQ2VydCBUTFMgUlNBIFNIQTI1NiAyMDIwIENBMTAeFw0yMjAzMTQwMDAwMDBa
Fw0yMzAzMTQyMzU5NTlaMIGWMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZv
cm5pYTEUMBIGA1UEBxMLTG9zIEFuZ2VsZXMxQjBABgNVBAoMOUludGVybmV0wqBD
b3Jwb3JhdGlvbsKgZm9ywqBBc3NpZ25lZMKgTmFtZXPCoGFuZMKgTnVtYmVyczEY
MBYGA1UEAxMPd3d3LmV4YW1wbGUub3JnMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A
MIIBCgKCAQEAlV2WY5rlGn1fpwvuBhj0nVBcNxCxkHUG/pJG4HvaJen7YIZ1mLc7
/P4snOJZiEfwWFTikHNbcUCcYiKG8JkFebZOYMc1U9P1EtVWGU4kuYuxiXpD8oMP
in1B0SgrF7gKfO1//I2weJdAUjgZuXBCPAlhz2EnHddzXUtwm9XuOLO/Y6LATVMs
bp8/lXnfo/bX0UgJ7C0aVqOu07A0Vr6OkPxwWmOvF3cRKhVCM7U4B51KK+IsWRLm
8cVW1IaXjwhGzW7BR6EI3sxCQ4Wnc6HVPSgmomLWWWkIGFPAwcWUB4NC12yhCO5i
W/dxNMWNLMRVtnZAyq6FpZ8wFK6j4OMwMwIDAQABo4ID1TCCA9EwHwYDVR0jBBgw
FoAUt2ui6qiqhIx56rTaD5iyxZV2ufQwHQYDVR0OBBYEFPcqCdAkWxFx7rq+9D4c
PVYSiBa7MIGBBgNVHREEejB4gg93d3cuZXhhbXBsZS5vcmeCC2V4YW1wbGUubmV0
ggtleGFtcGxlLmVkdYILZXhhbXBsZS5jb22CC2V4YW1wbGUub3Jngg93d3cuZXhh
bXBsZS5jb22CD3d3dy5leGFtcGxlLmVkdYIPd3d3LmV4YW1wbGUubmV0MA4GA1Ud
DwEB/wQEAwIFoDAdBgNVHSUEFjAUBggrBgEFBQcDAQYIKwYBBQUHAwIwgY8GA1Ud
HwSBhzCBhDBAoD6gPIY6aHR0cDovL2NybDMuZGlnaWNlcnQuY29tL0RpZ2lDZXJ0
VExTUlNBU0hBMjU2MjAyMENBMS00LmNybDBAoD6gPIY6aHR0cDovL2NybDQuZGln
aWNlcnQuY29tL0RpZ2lDZXJ0VExTUlNBU0hBMjU2MjAyMENBMS00LmNybDA+BgNV
HSAENzA1MDMGBmeBDAECAjApMCcGCCsGAQUFBwIBFhtodHRwOi8vd3d3LmRpZ2lj
```

โดยใน step นี้ จะทำการดึงข้อมูล certificate มาจากเว็ปที่เราต้องการ โดย เราต้องการ Certificate ของเว็ปไซต์ 2 อันโดยดูที่ Certificate chain และหลังจากจบ certificate ของแต่ละอัน โดยจะนำ certificate ที่เริ่มด้วย Begin certificate ถึง End certificate มาเก็บไว้เป็น 2 ไฟล์คือ c0.pem และ c1.pem

Step 2: Extract the public key (e, n) from the issuer's certificate.

```
[04/24/22]seed@VM:~/.../Task6$ openssl x509 -in c1.pem -noout -modulus
Modulus=C14BB3654770BCDD4F58DBEC9CEDC366E51F311354AD4A66461F2C0AEC6407E52EDCDCB90A20EDDFE3C4D09E9AA97A1D8288E51156DB
1E9F58C251E72C340D2ED292E156CBF1795FB3BB87CA25037B9A52416610604F571349F0E8376783DFE7D34B674C2251A6DF0E9910ED57517426
E27DC7CA622E131B7F238825536FC13458008B84FFF8BEA75849227B96ADA2889B15BCA07CDFE951A8D5B0ED37E236B4824B62B5499AECC767D6
E33EF5E3D6125E44F1BF71427D58840380B18101FAF9CA32BBB48E278727C52B74D4A8D697DEC364F9CACE53A256BC78178E490329AEFB494FA4
15B9CEF25C19576D6B79A72BA2272013B5D03D40D321300793EA99F5
```

```
[04/24/22]seed@VM:~/.../Task6$ openssl x509 -in c1.pem -text -noout | grep Exponent
            Exponent: 65537 (0x10001)
```

โดยใน step นี้ จะนำค่า Public Key ออกมาจาก certificate โดยใช้ openssl x509 -in c1.pem -noout – modulus ก็จะได้ Public Key มา และจะนำค่า e ออกมาจาก certificate โดยใช้ openssl x509 -in c1.pem -text -noout | grep Exponent เพื่อจะได้ค่า e ออกมา โดยจะออกมาเป็นค่า dec

Step 3: Extract the signature from the server's certificate.

```
[04/24/22]seed@VM:~/.../Task6$ openssl x509 -in c0.pem -text -noout
```

```
    Signature Algorithm: sha256WithRSAEncryption
         aa:9f:be:5d:91:1b:ad:e4:4e:4e:cc:8f:07:64:44:35:b4:ad:
         3b:13:3f:c1:29:d8:b4:ab:f3:42:51:49:46:3b:d6:cf:1e:41:
         83:e1:0b:57:2f:83:69:79:65:07:6f:59:03:8c:51:94:89:18:
         10:3e:1e:5c:ed:ba:3d:8e:4f:1a:14:92:d3:2b:ff:d4:98:cb:
         a7:93:0e:bc:b7:1b:93:a4:42:42:46:d9:e5:b1:1a:6b:68:2a:
         9b:2e:48:a9:2f:1d:2a:b0:e3:f8:20:94:54:81:50:2e:ee:d7:
         e0:20:7a:7b:2e:67:fb:fa:d8:17:a4:5b:dc:ca:00:62:ef:23:
         af:7a:58:f0:7a:74:0c:bd:4d:43:f1:8c:02:87:dc:e3:ae:09:
         d2:f7:fa:37:3c:d2:4b:ab:04:e5:43:a5:d2:55:11:0e:41:87:
         5f:38:a8:e5:7a:5e:4c:46:b8:b6:fa:3f:c3:4b:cd:40:35:ff:
         e0:a4:71:74:0a:c1:20:8b:e3:54:47:84:d5:18:bd:51:9b:40:
         5d:dd:42:30:12:d1:3a:a5:63:9a:af:90:08:d6:1b:d1:71:0b:
         06:71:90:eb:ae:ad:af:ba:5f:c7:db:6b:1e:78:a2:b4:d1:06:
         23:a7:63:f3:b5:43:fa:56:8c:50:17:7b:1c:1b:4e:10:6b:22:
         0e:84:52:94
[04/24/22]seed@VM:~/.../Task6$ cat signature | tr -d '[:space:]:'
cat: signature: No such file or directory
[04/24/22]seed@VM:~/.../Task6$ cat signature | tr -d '[:space:]:'
aa9fbe5d911bade44e4ecc8f07644435b4ad3b133fc129d8b4abf3425149463bd6cf1e4183e10b572f83697965076f59038c51948918103e1e5c
edba3d8e4f1a1492d32bffd498cba7930ebcb71b93a4424246d9e5b11a6b682a9b2e48a92f1d2ab0e3f820945481502eeed7e0207a7b2e67fbfa
d817a45bdcca0062ef23af7a58f07a740cbd4d43f18c0287dce3ae09d2f7fa373cd24bab04e543a5d255110e41875f38a8e57a5e4c46b8b6fa3f
c34bcd4035ffe0a471740ac1208be3544784d518bd519b405ddd423012d13aa5639aaf9008d61bd1710b067190ebaeadafba5fc7db6b1e78a2b4
```

โดยใน step นี้มันจะดึง signature ออกจาก certificate โดยเมื่อได้ signature มาแล้ว แต่จะยังติดโคล่อนอยู่ ก็จะต้องกำจัดออกโดยใช้ cat signature | tr -d '[:space:]:'

Step 4: Extract the body of the server's certificate.

```
[04/24/22]seed@VM:~/.../Task6$ openssl asn1parse -i -in c0.pem -strparse 4 -out c0_body.bin -noout
[04/24/22]seed@VM:~/.../Task6$ sha256sum c0_body.bin
7061df0a50b8f2ba3367ecfabab273a16f3bb1378dbe1fe524e6dfd90dfa3b91  c0_body.bin
```

โดยใน step นี้ต้องการรับ body ของ certificate โดยใช้คำสั่งจากภาพด้านบนโดย ผลลัพท์ที่ได้จะสามารถนำไปคำนวนได้

Step 5: Verify the signature

```c
// Task 6 Manually Verifying an X.509 Certificate

// assign values
BN_hex2bn(&n, "C14BB3654770BCDD4F58DBEC9CEDC366E51F311354AD4A66461F2C0AEC6407E52EDCDCB90A20EDDFE3C4D09E9AA97A1D8288E51156DB1E9F58
C251E72C340D2ED292E156CBF1795FB3BB87CA25037B9A52416610604F571349F0E8376783DFE7D34B674C2251A6DF0E9910ED57517426E27DC7CA622E131B7F2
38825536FC13458008B84FFF8BEA75849227B96ADA2889B15BCA07CDFE951A8D5B0ED37E236B4824B62B5499AECC767D6E33EF5E3D6125E44F1BF71427D588403
80B18101FAF9CA32BBB48E278727C52B74D4A8D697DEC364F9CACE53A256BC78178E490329AEFB494FA415B9CEF25C19576D6B79A72BA2272013B5D03D40D321300793EA99F5"); //modulus
BN_dec2bn(&e, "65537"); //from grep Exponent
BN_hex2bn(&M, "0001FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF000FAA63109307BC3D414892640CCD4D9A7061df0a50b8f2ba3367ecfabab273a16f3bb1378dbe1fe524e6dfd90dfa3b91"); //hash of the certificate body
BN_hex2bn(&S, "aa9fbe5d911bade44e4ecc8f07644435b4ad3b133fc129d8b4abf3425149463bd6cf1e4183e10b572f83697965076f59038c51948918103e1e
5cedba3d8e4f1a1492d32bffd498cba7930ebcb71b93a4424246d9e5b11a6b682a9b2e48a92f1d2ab0e3f820945481502eeed7e0207a7b2e67fbfad817a45bdcc
a0062ef23af7a58f07a740cbd4d43f18c0287dce3ae09d2f7fa373cd24bab04e543a5d255110e41875f38a8e57a5e4c46b8b6fa3fc34bcd4035ffe0a471740ac1208be3544784d518bd519b405ddd423012d13aa5639aaf9008d"); //cat signature

// Solution: S^e mod n = (M^d mod)^e mod n = M for Decrypt
BN_mod_exp(C, S, e, n, ctx);

// verify the signature
if (BN_cmp(C, M) == 0)
{
    printf("Valid Signature! \n");
}
else
{
    printf("Verification fails! \n");
}
```

```
[04/25/22]seed@VM:~/.../Task6$ gcc CS234_Security_Lab03-Cryptography-RSA_S40_task-6.c -lcrypto
[04/25/22]seed@VM:~/.../Task6$ ./a.out
Verification fails!
```

โดยใน step นี้จะทำ ทุกอย่างที่ได้มาเช็คว่า Signature ถูกต้องไหม ซึ่งในที่นี้เป็น verification fails