Activity 5 : Public Key Infrastructure

1. From the two given openssl commands, what is the difference?

- The result of the first command (without certificate)

```
parallels@ubuntu-linux-20-04-desktop:~$ openssl s_client -connect twitter.com:443 -CApath /
CONNECTED(00000003)
depth=1 C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
verify error:num=20:unable to get local issuer certificate
verify return:1
depth=0 C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com
verify return:1
---
Certificate chain
 0 s:C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com
   i:C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
 1 s:C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
   i:C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert Global Root CA
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIFbTCCBPSgAwIBAgIQCxxca7CYoYZhknbWP5XwSTAKBggqhkjOPQQDAzBWMQsw
CQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMTAwLgYDVQQDEydEaWdp
Q2VydCBUTFMgSHlicmlkIEVDQyBTSEEzODQgMjAyMCBDQTEwHhcNMjMwMTAzMDAw
MDAwWhcNMjQwMTAzMjM1OTU5WjBoMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2Fs
aWZvcm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEWMBQGA1UEChMNVHdpdHRl
ciwgSW5jLjEUMBIGA1UEAxMLdHdpdHRlci5jb20wWTATBgcqhkjOPQIBBggqhkjO
PQMBBwNCAATBaScaGRxLMLZWvebpgniMdyeJAPd5S8rrkHQAGF3VZgYqe5A2LskC
hc80OAj1GR/Ws42ksCxMSeimFafMsvYbo4IDkDCCA4wwHwYDVR0jBBgwFoAUCrwI
KReMpTlteg7OM8cus+37w3owHQYDVR0OBBYEFAGdUg9XjselYGygjXob4/9XfIxn
MCcGA1UdEQQgMB6CC3R3aXR0ZXIuY29tgg93d3cudHdpdHRlci5jb20wDgYDVR0P
AQH/BAQDAgeAMB0GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjCBmwYDVR0f
BIGTMIGQMEagRKBChkBodHRwOi8vY3JsMy5kaWdpY2VydC5jb20vRGlnaUNlcnRU
TFNIeWJyaWRFQ0NTSEEzODQyMDIwQ0ExLTEuY3JsMEagRKBChkBodHRwOi8vY3Js
NC5kaWdpY2VydC5jb20vRGlnaUNlcnRUTFNIeWJyaWRFQ0NTSEEzODQyMDIwQ0Ex
LTEuY3JsMD4GA1UdIAQ3MDUwMwYGZ4EMAQICMCkwJwYIKwYBBQUHAgEWG2h0dHA6
Ly93d3cuZGlnaWNlcnQuY29tL0NQUzCBhQYIKwYBBQUHAQEEeTB3MCQGCCsGAQUF
BzABhhhodHRwOi8vb2NzcC5kaWdpY2VydC5jb20wTwYIKwYBBQUHMAKGQ2h0dHA6
Ly9jYWNlcnRzLmRpZ2ljZXJ0LmNvbS9EaWdpQ2VydFRMU0h5YnJpZEVDQ1NIQTM4
NDIwMjBDQTEtMS5jcnQwCQYDVR0TBAIwADCCAX8GCisGAQQB1nkCBAIEggFvBIIB
awFpAHYAdv+IPwq2+5VRwmHM9Ye6NLSkzbsp3GhCCp/mZ0xaOnQAAAGFdXNqOgAA
BAMARzBFAiEAnxcMnInk6eUd0CDpAFMbhhBIMPEyFyNnlMfo2wnDa/UCIHPEh9A4
WpNIo5/CgwwnGe3gKQj7pqh7sstRMTLjXx9fAHcASLDja9qmRzQP5WoC+p0w6xxS
ActW3SyB2bu/qznYhHMAAAGFdXNqMQAABAMASDBGAiEAmr217SflCPjuTlQ97tg8
ZWmsyc5AucJM+UHaPThIlM0CIQCUY5gNk15yQ5sAbd9wogcV2PxD14GRwxDmpGnJ
lLa0AQB2ADtTd3U+LbmATosWWb+QDtn2E/D9Me9AA0tcm/h+tQXAAABhXVzai8A
AAQDAEcwRQIgF5MxpEzc0ynB3tR7S2J0BsCxJgXJOUsjsHsTf/Y6LKACIQClZbM+
tJGWw9xfqZ6VievbVrxiC+ME0Ufy9HJEgEwfnTAKBggqhkjOPQQDAwNnADBkAjBN
i7g7j+zuqx/cTZtpyKmC6vU5chBf1CEZujVEIJ9DWOQ6ieiJ3OP8U8eLCxzXpF4C
MGj2yWKCTRG1tHX7y883nO4m4fTmqUBORBrrLPGuDJF2MUO+dJz6MpOVZzhA056T
PA==
-----END CERTIFICATE-----
subject=C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com

issuer=C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1

---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: ECDSA
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 2754 bytes and written 383 bytes
Verification error: unable to get local issuer certificate
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Server public key is 256 bit
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 20 (unable to get local issuer certificate)
---
```

- The result of the second command (with certificate)

```
parallels@ubuntu-linux-20-04-desktop:~$ openssl s_client -connect twitter.com:443 -CAfile /etc/ssl/certs/ca-certificates.crt
CONNECTED(00000003)
depth=2 C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert Global Root CA
verify return:1
depth=1 C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
verify return:1
depth=0 C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com
verify return:1
---
Certificate chain
 0 s:C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com
   i:C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
 1 s:C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
   i:C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert Global Root CA
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIFbTCCBPSgAwIBAgIQCxxca7CYoYZhknbWP5XwSTAKBggqhkjOPQQDAzBWMQsw
CQYDVQQGEwJVUzEVMBMGA1UEChMMRGlnaUNlcnQgSW5jMTAwLgYDVQQDEydEaWdp
Q2VydCBUTFMgSHlicmlkIEVDQyBTSEEzODQgMjAyMCBDQTEwHhcNMjMwMTAzMDAw
MDAwWhcNMjQwMTAzMjM1OTU5WjBoMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2Fs
aWZvcm5pYTEWMBQGA1UEBxMNU2FuIEZyYW5jaXNjbzEWMBQGA1UEChMNVHdpdHRl
citwgSW5jLjEUMBIGA1UEAxMLdHdpdHRlci5jb20wWTATBgcqhkjOPQIBBggqhkjO
PQMBBwNCAATBaScaGRxLMLZWvebpgniMdyeJAPd5S8rrkHQAGF3VZgYqe5A2LskC
hc80OAj1GR/Ws42ksCxMSeimFafMsvYbo4IDkDCCA4wwHwYDVR0jBBgwFoAUCrwI
KReMpTlteg7OM8cus+37w3owHQYDVR0OBBYEFAGdUg9XjselYGygjXob4/9XfIxn
MCcGA1UdEQQgMB6CC3R3aXR0ZXIuY29tgg93d3cudHdpdHRlci5jb20wDgYDVR0P
AQH/BAQDAgeAMB0GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjCBmwYDVR0f
BIGTMIGQMEagRKBChkBodHRwOi8vY3JsMy5kaWdpY2VydC5jb20vRGlnaUNlcnRU
TFNIeWJyaWRFQ0NTSEEzODQyMDIwQ0ExLTEuY3JsMEagRKBChkBodHRwOi8vY3Js
NC5kaWdpY2VydC5jb20vRGlnaUNlcnRUTFNIeWJyaWRFQ0NTSEEzODQyMDIwQ0Ex
LTEuY3JsMD4GA1UdIAQ3MDUwMwYGZ4EMAQICMCkwJwYIKwYBBQUHAgEWG2h0dHA6
Ly93d3cuZGlnaWNlcnQuY29tL0NQUzCBhQYIKwYBBQUHAQEEeTB3MCQGCCsGAQUF
BzABhhhodHRwOi8vb2NzcC5kaWdpY2VydC5jb20wTwYIKwYBBQUHMAKGQ2h0dHA6
Ly9jYWNlcnRzLmRpZ2ljZXJ0LmNvbS9EaWdpQ2VydFRMU0h5YnJpZEVDQ1NIQTM4
NDIwMjBDQTEtMS5jcnQwCQYDVR0TBAIwADCCAX8GCisGAQQB1nkCBAIEggFvBIIB
awFpAHYAdv+IPwq2+5VRwmHM9Ye6NLSkzbsp3GhCCp/mZ0xaOnQAAAGFdXNqOgAA
BAMARzBFAiEAnxcMnInk6eUd0CDpAFMbhhBIMPEyFyNnlMfo2wnDa/UCIHPEh9A4
WpNIo5/CgwwnGe3gKQj7pqh7sstRMTLjXx9fAHcASLDja9qmRzQP5WoC+p0w6xxS
ActW3SyB2bu/qznYhHMAAAGFdXNqMQAABAMASDBGAiEAmr217SflCPjuTlQ97tg8
ZWmsyc5AucJM+UHaPThIlM0CIQCUY5gNk15yQ5sAbd9wogcV2PxD14GRwxDmpGnJ
lLa0AQB2ADtTd3U+LbmATosWWb+QDtn2E/D9Me9AA0tcm/h+tQXAAABhXVzai8A
AAQDAEcwRQIgF5MxpEzc0ynB3tR7S2J0BsCxJgXJOUsjsHsTf/Y6LKACIQClZbM+
tJGWw9xfqZ6VievbVrxiC+ME0Ufy9HJEgEwfnTAKBggqhkjOPQQDAwNnADBkAjBN
i7g7i+zuqx/cT7tpvKmC6vU5chBf1CEZuiVEIJ9DWOO6iejJ3OP8U8eLCxzXpF4C
PA==
-----END CERTIFICATE-----
subject=C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com

issuer=C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1

---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: ECDSA
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 2755 bytes and written 383 bytes
Verification: OK
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Server public key is 256 bit
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 0 (ok)
---
```

The first command trust and use the system's default CA, while the second command trusts only the certificates provided in “ca-certificates.crt” .
the first command gives Verify return code: 20 (unable to get local issuer certificate) while the second results in Verify return code: 0 (ok).

2. What does the error (verify error) in the first command mean? Please explain.

The "verify error" in the initial command indicates that the system couldn't validate the received public key's authenticity. This is due to the absence of the necessary certificate for verification.

3. Copy the server certificate (beginning with -----BEGIN CERTIFICATE----- and ending with -----END CERTIFICATE-----) and store it as twitter_com.cert. Use the command openssl x509 -in twitter_com.cert -text to show a text representation of the certificate content. Briefly explain what is stored in an X.509 certificate (i.e. data in each field).

1. **Version**: Specifies the X.509 version used. Here, it's version 3 (0x2 in hexadecimal).
2. **Serial Number**: A unique number assigned by the Certificate Authority (CA) during issuance. This helps in identifying the certificate uniquely among others issued by the same CA.
3. **Signature Algorithm**: The algorithm used by the CA to sign this certificate. In this case, it's ecdsa-with-SHA384.
4. **Issuer**: Provides details about the Certificate Authority (CA) that issued the certificate. Here, the issuer is "DigiCert Inc" and the specific certificate used for signing is "DigiCert TLS Hybrid ECC SHA384 2020 CA1".
5. **Validity**:
   - **Not Before**: The start date of the certificate's validity.
   - **Not After**: The expiration date of the certificate.
6. **Subject**: Details about the entity for which the certificate was issued. It indicates that this certificate is for twitter.com owned by "Twitter, Inc." based in San Francisco, California.
7. **Subject Public Key Info**: Contains the public key details:
   - **Public Key Algorithm**: Specifies the type of public key, here it's an elliptic curve public key (id-ecPublicKey).
   - **Public-Key**: The actual public key value.
   - **ASN1 OID & NIST CURVE**: Identify the specific elliptic curve used. Here, it's prime256v1 or commonly known as P-256.
8. **X509v3 extensions**: A set of additional properties and metadata for the certificate:
   - **Authority Key Identifier & Subject Key Identifier**: These are identifiers that help in linking certificates in a chain.
   - **Subject Alternative Name**: Alternative names for which this certificate is valid. Here, it's valid for both twitter.com and www.twitter.com.
   - **Key Usage & Extended Key Usage**: Define purposes for which the public key can be used. This certificate is primarily for server and client authentication over TLS.
   - **CRL Distribution Points**: URLs where browsers can check if this certificate has been revoked.
   - **Certificate Policies**: Identifies the policy under which the certificate has been issued and a link (CPS) to a detailed Certification Practice Statement.
   - **Authority Information Access**: Provides links to the OCSP server for real-time revocation checking and to the CA's certificate.
   - **Basic Constraints**: Indicates if the certificate is a CA certificate. Here, it's not (CA:FALSE).
   - **CT Precertificate SCTs**: These are Signed Certificate Timestamps, evidence that the certificate has been logged in public Certificate Transparency logs. This helps in detecting mistakenly or maliciously issued certificates.
9. **Signature Algorithm**: This is repeated at the end, just before the actual signature. It indicates the algorithm used to create the signature below.
10. **Signature**: The digital signature generated by the CA. This is used to verify that the certificate was genuinely issued by the stated CA.

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            0b:1c:5c:6b:b0:98:a1:86:61:92:76:d6:3f:95:f0:49
        Signature Algorithm: ecdsa-with-SHA384
        Issuer: C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
        Validity
            Not Before: Jan  3 00:00:00 2023 GMT
            Not After : Jan  3 23:59:59 2024 GMT
        Subject: C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:c1:69:27:1a:19:1c:4b:30:b6:56:bd:e6:e9:82:
                    78:8c:77:27:89:00:f7:79:4b:ca:eb:90:74:00:18:
                    5d:d5:66:06:2a:7b:90:36:2e:c9:02:85:cf:34:38:
                    08:f5:19:1f:d6:b3:8d:a4:b0:2c:4c:49:e8:a6:15:
                    a7:cc:b2:f6:1b
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Authority Key Identifier:
                keyid:0A:BC:08:29:17:8C:A5:39:6D:7A:0E:CE:33:C7:2E:B3:ED:FB:C3:7A

            X509v3 Subject Key Identifier:
                01:9D:52:0F:57:8E:C7:A5:60:6C:A0:8D:7A:1B:E3:FF:57:7C:8C:67
            X509v3 Subject Alternative Name:
                DNS:twitter.com, DNS:www.twitter.com
            X509v3 Key Usage: critical
                Digital Signature
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 CRL Distribution Points:

                Full Name:
                  URI:http://crl3.digicert.com/DigiCertTLSHybridECCSHA3842020CA1-1.crl

                Full Name:
                  URI:http://crl4.digicert.com/DigiCertTLSHybridECCSHA3842020CA1-1.crl

            X509v3 Certificate Policies:
                Policy: 2.23.140.1.2.2
                  CPS: http://www.digicert.com/CPS

            Authority Information Access:
                OCSP - URI:http://ocsp.digicert.com
                CA Issuers - URI:http://cacerts.digicert.com/DigiCertTLSHybridECCSHA3842020CA1-1.crt

            X509v3 Basic Constraints:
                CA:FALSE
            CT Precertificate SCTs:
                Signed Certificate Timestamp:
                    Version   : v1 (0x0)
                    Log ID    : 76:FF:88:3F:0A:B6:FB:95:51:C2:61:CC:F5:87:BA:34:
                                B4:A4:CD:BB:29:DC:68:42:0A:9F:E6:67:4C:5A:3A:74
                    Timestamp : Jan  3 02:26:16.250 2023 GMT
                    Extensions: none
                    Signature : ecdsa-with-SHA256
                                30:45:02:21:00:9F:17:0C:9C:89:E4:E9:E5:1D:D0:20:
                                E9:00:53:1B:86:10:48:30:F1:32:17:23:67:94:C7:E8:
                                DB:09:C3:6B:F5:02:20:73:C4:87:D0:38:5A:93:48:A3:
                                9F:C2:83:0C:27:19:ED:E0:29:08:FB:A6:A8:7B:B2:CB:
                                51:31:32:E3:5F:1F:5F
                Signed Certificate Timestamp:
                    Version   : v1 (0x0)
                    Log ID    : 48:B0:E3:6B:DA:A6:47:34:0F:E5:6A:02:FA:9D:30:EB:
                                1C:52:01:CB:56:DD:2C:81:D9:BB:BF:AB:39:D8:84:73
                    Timestamp : Jan  3 02:26:16.241 2023 GMT
                    Extensions: none
                    Signature : ecdsa-with-SHA256
                                30:46:02:21:00:9A:BD:B5:ED:27:E5:08:F8:EE:4E:54:
                                3D:EE:D8:3C:65:69:AC:C9:CE:40:B9:C2:4C:F9:41:DA:
                                3D:38:48:94:CD:02:21:00:94:63:98:0D:93:5E:72:43:
                                9B:00:6D:DF:70:A2:07:15:D8:FC:43:D7:81:91:C3:10:
                                E6:A4:69:C9:94:B6:B4:01
                Signed Certificate Timestamp:
                    Version   : v1 (0x0)
                    Version   : v1 (0x0)
                    Log ID    : 3B:53:77:75:3E:2D:B9:80:4E:8B:30:5B:06:FE:40:3B:
                                67:D8:4F:C3:F4:C7:BD:00:0D:2D:72:6F:E1:FA:D4:17
                    Timestamp : Jan  3 02:26:16.239 2023 GMT
                    Extensions: none
                    Signature : ecdsa-with-SHA256
                                30:45:02:20:17:93:31:A4:4C:DC:D3:29:C1:DE:D4:7B:
                                4B:62:74:06:C0:B1:26:05:C9:39:4B:23:B0:7B:13:7F:
                                F6:3A:2C:A0:02:21:00:A5:65:B3:3E:B4:91:96:C3:DC:
                                5F:A9:9E:95:89:EB:DB:56:BC:62:0B:E3:04:D1:47:F2:
                                F4:72:44:80:4C:1F:9D
    Signature Algorithm: ecdsa-with-SHA384
         30:64:02:30:4d:8b:b8:3b:8f:ec:ee:ab:1f:dc:4d:9b:69:c8:
         a9:82:ea:f5:39:72:10:5f:d4:21:19:ba:35:44:20:9f:43:58:
         e4:3a:89:e8:89:dc:e3:fc:53:c7:8b:0b:1c:d7:a4:5e:02:30:
         68:f6:c9:62:82:4d:11:b5:b4:75:fb:cb:cf:37:9c:ee:26:e1:
         f4:e6:a9:40:4e:44:1a:eb:2c:f1:ae:0c:91:76:31:43:be:74:
         9c:fa:32:93:95:67:38:40:d3:9e:93:3c
```

4. From the information in exercise 3, is there an intermediate certificate? If yes, what purpose does it serve?

Yes, there is an intermediate certificate. Its function is to validate the intermediate TLS Server, namely "DigiCert TLS Hybrid ECC SHA384 2020 CA1

5. Is there an intermediate CA, i.e. is there more than one organization involved in the certification? Say why you think so.

Yes there is, base on the results from running openssl s_client -connect twitter.com:443

```
parallels@ubuntu-linux-20-04-desktop:~$ openssl s_client -connect twitter.com:443 -CAfile /etc/ssl/certs/ca-certificates.crt
CONNECTED(00000003)
depth=2 C = US, O = DigiCert Inc, OU = www.digicert.com, CN = DigiCert Global Root CA
verify return:1
depth=1 C = US, O = DigiCert Inc, CN = DigiCert TLS Hybrid ECC SHA384 2020 CA1
verify return:1
depth=0 C = US, ST = California, L = San Francisco, O = "Twitter, Inc.", CN = twitter.com
verify return:1
```

6. What is the role of ca-certificates.crt?

The file “ca-certificates.crt” contains the trusted root certificates of various Certificate Authorities (CAs). It plays a pivotal role in the trust framework by containing the public keys of these trusted CAs. Our devices rely on these keys to verify certificates presented by websites or services. Essentially, if a certificate can be traced back to one of the trusted roots in this file, then the entity presenting the certificate is deemed trustworthy by our device.

7. Explore the ca-certificates.crt. How many certificates are in there? Give the command/method you have used to count.

openssl storeutl -noout -certs /etc/ssl/certs/ca-certificates.crt Total found: 137

8. Extract a root certificate from ca-certificates.crt. Use the openssl command to explore the details. Do you see any Issuer information? Please compare it to the details of twitter's certificate and the details of the intermediate certificate.

```
parallels@ubuntu-linux-20-04-desktop:~/Desktop/Parallels Shared Folders/Home/desktop$ openssl x509 -in '/home/parallels/Desktop/Parall
els Shared Folders/Home/Desktop/class-lecture/2023S12110413-Computer-Security-Activity/activity5/certficates/root.cert' -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 6828503384748696800 (0x5ec3b7a6437fa4e0)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: CN = ACCVRAIZ1, OU = PKIACCV, O = ACCV, C = ES
        Validity
            Not Before: May  5 09:37:37 2011 GMT
            Not After : Dec 31 09:37:37 2030 GMT
        Subject: CN = ACCVRAIZ1, OU = PKIACCV, O = ACCV, C = ES
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (4096 bit)
                Modulus:
                    00:9b:a9:ab:bf:61:4a:97:af:2f:97:66:9a:74:5f:
                    d0:d9:96:fd:cf:e2:e4:66:ef:1f:1f:47:33:c2:44:
                    a3:df:9a:de:1f:b5:54:dd:15:7c:69:35:11:6f:bb:
                    c8:0c:8e:6a:18:1e:d8:8f:d9:16:bc:10:48:36:5c:
                    f0:63:b3:90:5a:5c:24:37:d7:a3:d6:cb:09:71:b9:
                    f1:01:72:84:b0:7d:db:4d:80:cd:fc:d3:6f:c9:f8:
                    da:b6:0e:82:d2:45:85:a8:1b:68:a8:3d:e8:f4:44:
                    6c:bd:a1:c2:cb:03:be:8c:3e:13:00:84:df:4a:48:
                    c0:e3:22:0a:e8:e9:37:a7:18:4c:b1:09:0d:23:56:
                    7f:04:4d:d9:17:84:18:a5:c8:da:40:94:73:eb:ce:
                    0e:57:3c:03:81:3a:9d:0a:a1:57:43:69:ac:57:6d:
                    79:90:78:e5:b5:b4:3b:d8:bc:4c:8d:28:a1:a7:a3:
                    a7:ba:02:4e:25:d1:2a:ae:ed:ae:03:22:b8:6b:20:
                    0f:30:28:54:95:7f:e0:ee:ce:0a:66:9d:d1:40:2d:
                    6e:22:af:9d:1a:c1:05:19:d2:6f:c0:f2:9f:f8:7b:
                    b3:02:42:fb:50:a9:1d:2d:93:0f:23:ab:c6:c1:0f:
                    92:ff:d0:a2:15:f5:53:09:71:1c:ff:45:13:84:e6:
                    26:5e:f8:e0:88:1c:0a:fc:16:b6:a8:73:06:b8:f0:
                    63:84:02:a0:c6:5a:ec:e7:74:df:70:ae:a3:83:25:
                    ea:d6:c7:97:87:93:a7:c6:8a:8a:33:97:60:37:10:
                    3e:97:3e:6e:29:15:d6:a1:0f:d1:88:2c:12:9f:6f:
                    aa:a4:c6:42:eb:41:a2:e3:95:43:d3:01:85:6d:8e:
                    bb:3b:f3:23:36:c7:fe:3b:e0:a1:25:07:48:ab:c9:
                    89:74:ff:08:8f:80:bf:c0:96:65:f3:ee:ec:4b:68:
                    bd:9d:88:c3:31:b3:40:f1:e8:cf:f6:38:bb:9c:e4:
                    d1:7f:d4:e5:58:9b:7c:fa:d4:f3:0e:9b:75:91:e4:
                    ba:52:2e:19:7e:d1:f5:cd:5a:19:fc:ba:06:f6:fb:
                    52:a8:4b:99:04:dd:f8:f9:b4:8b:50:a3:4e:62:89:
                    52:a8:4b:99:04:dd:f8:f9:b4:8b:50:a3:4e:62:89:
                    f0:87:24:fa:83:42:c1:87:fa:d5:2d:29:2a:5a:71:
                    7a:64:6a:d7:27:60:63:0d:db:ce:49:f5:8d:1f:90:
                    89:32:17:f8:73:43:b8:d2:5a:93:86:61:d6:e1:75:
                    0a:ea:79:66:76:88:4f:71:eb:04:25:d6:0a:5a:7a:
                    93:e5:b9:4b:17:40:0f:b1:b6:b9:f5:de:4f:dc:e0:
                    b3:ac:3b:11:70:60:84:4a:43:6e:99:20:c0:29:71:
                    0a:c0:65
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            Authority Information Access:
                CA Issuers - URI:http://www.accv.es/fileadmin/Archivos/certificados/raizaccv1.crt
                OCSP - URI:http://ocsp.accv.es

            X509v3 Subject Key Identifier:
                D2:87:B4:E3:DF:37:27:93:55:F6:56:EA:81:E5:36:CC:8C:1E:3F:BD
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Authority Key Identifier:
                keyid:D2:87:B4:E3:DF:37:27:93:55:F6:56:EA:81:E5:36:CC:8C:1E:3F:BD

            X509v3 Certificate Policies:
                Policy: X509v3 Any Policy
                  User Notice:
                    Explicit Text:
                  CPS: http://www.accv.es/legislacion_c.htm

            X509v3 CRL Distribution Points:

                Full Name:
                  URI:http://www.accv.es/fileadmin/Archivos/certificados/raizaccv1_der.crl

            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Subject Alternative Name:
                email:accv@accv.es
    Signature Algorithm: sha1WithRSAEncryption
         97:31:02:9f:e7:fd:43:67:48:44:14:e4:29:87:ed:4c:28:66:
         d0:8f:35:da:4d:61:b7:4a:97:4d:b5:db:90:e0:05:2e:0e:c6:
         79:d0:f2:97:69:0f:bd:04:47:d9:be:db:b5:29:da:9b:d9:ae:
         a9:99:d5:d3:3c:30:93:f5:8d:a1:a8:fc:06:8d:44:f4:ca:16:
         95:7c:33:dc:62:8b:a8:37:f8:27:d8:09:2d:1b:ef:c8:14:27:
         20:a9:64:44:ff:2e:d6:75:aa:6c:4d:60:40:19:49:43:54:63:
         da:e2:cc:ba:66:e5:4f:44:7a:5b:d9:6a:81:2b:40:d5:7f:f9:
         01:27:58:2c:c8:ed:48:91:7c:3f:a6:00:cf:c4:29:73:11:36:
         de:86:19:3e:9d:ee:19:8a:1b:d5:b0:ed:8e:3d:9c:2a:c0:0d:
```

The issuer and subject are identical, which is a distinction from the intermediate and Twitter certificates, where the issuer and subject represent different entities.

9. If the intermediate certificate is not in a PEM format (text readable), use the command to convert a DER file (.crt .cer .der) to PEM file.
openssl x509 -inform der -in certificate.cer -out certificate.pem.
(You need the pem file for exercise 10.)

10. From the given python code,[1] implement the certificate validation.

```python
from OpenSSL import crypto
import pem


def verify(target_filename, intermediate_filenames, root_filename):
    with open(target_filename, "r") as cert_file:
        cert = cert_file.read()
    int_certs = []
    for filename in intermediate_filenames:
        with open(filename, "r") as cert_file:
            int_certs.append(cert_file.read())
    pems = pem.parse_file(root_filename)
    trusted_certs = [str(mypem) for mypem in pems] + int_certs

    verified = verify_chain_of_trust(cert, trusted_certs)
    if verified:
        print("Certificate verified")
    else:
        print("Certificate verification failed")


def verify_chain_of_trust(cert_pem, trusted_cert_pems):
    certificate = crypto.load_certificate(crypto.FILETYPE_PEM, cert_pem)
    store = crypto.X509Store()

    for trusted_cert_pem in trusted_cert_pems:
        trusted_cert = crypto.load_certificate(crypto.FILETYPE_PEM, trusted_cert_pem)
        store.add_cert(trusted_cert)

    store_ctx = crypto.X509StoreContext(store, certificate)
    result = store_ctx.verify_certificate()
    if result is None:
        return True
    else:
        return False


print("Verfiying Twitter certificate...")
verify("twitter_com.cert", ["int_twitter_com.cert"], "ca-certificates.cert")
print("Verfiying Google certificate...")
verify(
    "google_com.cert",
    ["int_google_com.cert", "int2_google_com.cert"],
    "ca-certificates.cert",
)
print("Verfiying Chula certificate...")
verify("chula_ac_th.cert", ["int_chula_ac_th.cert"], "ca-certificates.cert")
print("Verfiying Classdeedee certificate...")
verify("classdeedee.cert", ["int_classdeedee.cert"], "ca-certificates.cert")
```

```
pawankanjeam@Pawans-MacBook-Pro ~/Desktop/class-lecture/2023S12110413-Computer-Security-Activity/activity5/for#10 main ? /opt/homebrew/bin/python3 /Users/pawankanjeam/Desktop/class-lecture/2023S12110413-Computer-Security-Activity/activity5/for#10/certificate_validation.py
Verfiying Twitter certificate...
Certificate verified
Verfiying Google certificate...
Certificate verified
Verfiying Chula certificate...
Certificate verified
Verfiying Classdeedee certificate...
Certificate verified
```

11. Nowaday, there are root certificates for class 1 and class 3. What uses would a class 1 signed certificate have that a class 3 doesn't, and vice versa?

Class 1 is a legacy root certificate encompassing both low and high-security validations, while Class 3 represents a more recent, elevated-security certificate. Class 1 is compatible with older browsers, but Class 3 caters to modern browsers with enhanced security requirements.

12. Assuming that a Root CA in your root store is hacked and under the control of an attacker, and this is not noticed by anyone for months.

A. What further attacks can the attacker stage? Draw a possible attack setup.
B. In the attack you have described above, can we rely on CRLs or OCSP for protection? Please explain

A. **Attack Setup :**
If a Root CA is compromised:
- Fraudulent Certificates: Attackers can issue fake certificates for any domain.
- Man-in-the-Middle: They can intercept and alter communications without users realizing.
- Data Theft: Users, thinking they're on genuine sites, might give away personal data.
- Malware Distribution: Users could be redirected to malicious sites that look legitimate.
- Reputation Damage: Fake sites can spread misinformation.

B. **CRLs and OCSP :**
If the Root CA breach is undetected, CRLs and OCSP might not help much. There's often a delay in revocation, not all sites use the OCSP "must-staple" feature, and if the CA itself is compromised, its revocation data might be manipulated. Thus, relying solely on these tools in such a scenario would be inadequate.