PLAGIARISM STATEMENT

I certify that this assignment/report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they be books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment/report has not previously been submitted for assessment in any other course, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that I have not copied in part or whole or otherwise plagiarised the work of other students and/or persons. I pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, I understand my responsibility to report honour violations by other students if I become aware of it.

Name: Akash Tadwai
Date: 28-01-2022
Signature: Akash Tadwai

Assignment 1: ABCs of Digital Certificates

Individual Assignment

PART-A: Comparison of Digital Certificates in the chain of trust of a website

- Visit the website #N in this list of top-100 most visited websites globally where is #N is the last two digits in your roll number and download all the certificates in .CER format in the chain of trust from the root Certificate, intermediate certificate(s), to the end-user (website) certificate at the leaf in the hierarchy.
- Compare the digital certificates in the chain in terms of various field values by filling this table.

Field Name	Subject (CN) of certificate holder (website)	Subject (CN) of certificate holder (intermediate)	Subject (CN) of certificate holder (root)	Remarks/observations
Issuer	DigiCert TLS RSA SHA256 2020 CA1	DigiCert Global Root CA	DigiCert Global Root CA	- There are 3 certificates from root to end user in chain of trust. - - Both intermediate and root certificates are given by the same CA. - Issuer of root certificate is itself

Version No.	Version 3	Version 3	Version 3	
Signature Algo	PKCS #1 SHA-256 With RSA Encryption	PKCS #1 SHA-256 With RSA Encryption	PKCS #1 SHA-1 With RSA Encryption	Root CA used SHA-1 encryption hence size of digest decreases
Size of digest	256 bits	256 bits	160 bits	SHA 256 generates 256 bit signature whereas SHA 1 generates 160 bit digest.
Signature Value	69 7E 8B DD F3 9B BE 97 C3 2B 92 2B BF C0 39 51 22 F6 7B A7 97 60 A3 44 57 35 73 B1 8D CE E1 5B 3F A5 43 A7 96 79 F2 8E 38 E1 DF 88 E5 67 49 53 94 97 23 0A 44 1F 62 41 FB AE F6 E2 F0 D2 F6 44 AA E5 46 5F B0 84 BF 86 C1 6A AC 5B AD C8 91 6F E3 FB 97 08 91 24 B5 6B 54 8F 6F D5 2F 0E 63 C1 20 AA 54 4C 35 DA 77 76 39 47 96 66 9D 2A C5 79 89 39 CB 2E 84 81 FD B8 2D 04 A8 93 45 B9 08 7C C8 27 B9 7C 56 FF 25 94 CE DB D1 6C A0 8E 11 92 8E C4 2B 5E 7C 78 7E 6F 2F 84 B9 C8 5A 39 00 F7 48 E2 F1 72 29 16 26 E4 0F DE DD 35 44 7B AE 42 A9 CA EC 1A 67 7D A5 EC E9 4D 7C 4C ED B5 6B F2 7E F1 E7 98 FE DA C9 C5 C2 55 46 30 D0 A5 72 65 36 8E B0 F9 97 0A 94 8D 66 C7 78 BB E0 F2 25 1F 9E 8B A3 9F 3B 35 C0 CC D2 1A 78 63 8B E0 5D 3D 39 B4 6C BD 2E B2 41 95 17 08 3F D9 EF 41 7D 1E	77 AB B7 7A 27 3D AE BB F6 7F E0 5A 56 C9 84 AA CA 5B 71 17 DD 22 47 FC 4E 9F EE D0 C1 A4 04 E1 A3 EB C5 49 C1 FD D1 C9 DF 8C AF 94 45 2C 46 2A A3 63 39 20 F9 9E 4A 24 94 41 C8 A9 D9 E2 9C 54 05 06 CB 5C 1C BE 00 1B 0F A8 5A FF 19 BB 65 C7 16 AF 21 56 DD 61 05 C9 E9 8F 98 76 DF 6B 1B D0 72 0C 50 B9 30 29 7A BF 60 59 10 66 13 3A 2D AC 15 11 6C 2D 23 0C 02 3E 05 3B FE E5 A1 9C E2 8A DB 87 D7 4A E8 5E E7 48 06 EB AB 12 9A F2 AF 84 C3 5B 83 4A 99 81 83 AB 00 A1 CA 0A 3C 4C A2 25 89 2A 22 A7 A4 F3 33 4C 5B 8C 2E 1A 02 97 0F 9D 8F 6D 2D 95 08 FB 4F DA F1 91 38 25 E1 9C 6E 61 18 87 6A CE B1 BB 00 30 6A 9B B7 AF DA F1 C5 97 FE 8A 78 24 AA EA 93 80 BA 33 65 7A BC A1 77 E9 7F 69 14 0B 00 3F 77 92 B1 4D 5B 73 87 0A 13 D0 9C C8 F2 4B 39 4F 52 84 49 A6 4C 90 4E 1F F7 B4	CB 9C 37 AA 48 13 12 0A FA DD 44 9C 4F 52 B0 F4 DF AE 04 F5 79 79 08 A3 24 18 FC 4B 2B 84 C0 2D B9 D5 C7 FE F4 C1 1F 58 CB B8 6D 9C 7A 74 E7 98 29 AB 11 B5 E3 70 A0 A1 CD 4C 88 99 93 8C 91 70 E2 AB 0F 1C BE 93 A9 FF 63 D5 E4 07 60 D3 A3 BF 9D 5B 09 F1 D5 8E E3 53 F4 8E 63 FA 3F A7 DB B4 66 DF 62 66 D6 D1 6E 41 8D F2 2D B5 EA 77 4A 9F 9D 58 E2 2B 59 C0 40 23 ED 2D 28 82 45 3E 79 54 92 26 98 E0 80 48 A8 37 EF F0 D6 79 60 16 DE AC E8 0E CD 6E AC 44 17 38 2F 49 DA E1 45 3E 2A B9 36 53 CF 3A 50 06 F7 2E E8 C4 57 49 6C 61 21 18 D5 04 AD 78 3C 2C 3A 80 6B A7 EB AF 15 14 E9 D8 89 C1 B9 38 6C E2 91 6C 8A FF 64 B9 77 25 57 30 C0 1B 24 A3 E1 DC E9 DF 47 7C B5 B4 24 08 05 30 EC 2D BD 0B BF 45 BF 50 B9 A9 F3 EB 98 01 12 AD C8 88 C6 98 34 5F 8D 0A 3C C6 E9 D5 95 95 6D DE	
Validity period	2 Years 14/12/2021, 05:30:00 GMT+5:30 to 15/01/2023, 05:29:59 GMT+5:30	10 Years 24/09/2020, 05:30:00 GMT+5:30 to 24/09/2030, 05:29:59 GMT+5:30	25 Years 10/11/2006, 05:30:00 GMT+5:30 to 10/11/2031, 05:30:00 GMT+5:30	The period of validity increases as we go up in the chain of trust i.e, root certificates have the longest period of validity.
Is Subject field (CN), FQDN?	Yes it is FQDN as there are no ‘*’s in CN	Yes it is FQDN as there are no ‘*’s in CN	Yes it is FQDN as there are no ‘*’s in CN	
Certificate	OV certificate. The	OV certificate. The	OV certificate. The	All the

type: DV, IV, OV or EV? Tell also how you are able to determine the type!	subject field has the common name, as well as details about the company running the website	subject field has the common name, as well as details about the company running the website	subject field has the common name, as well as details about the company running the website	certificates are organization validated(OV).
Subject Alternative Name (SAN/UCC), if any	**DNS Name:** account.netflix.com ca.netflix.com netflix.ca netflix.com signup.netflix.com www.netflix.ca www1.netflix.com …	**None**	None	
Certificate category: Single domain, wildcard o Multi-domain SAN/UCC cert?	Multi Domain Cert.	Single Domain	Single Domain	The end entity in chain of trust Has multi-domain certificate i.e, it has many domains registered under a single certificate whereas the other certificates have only one domain under a certificate
Public Key Info like key algo, key length, public exponent (e) in case of RSA	**Algo:** PKCS #1 RSA Encryption Key Length: 2048 bits Exponent: 65537	**Algo:** PKCS #1 RSA Encryption Key Length: 2048 bits Exponent: 65537	**Algo:** PKCS #1 RSA Encryption Key Length: 2048 bits Exponent: 65537	
Public key or modulus (n) in case of RSA	C7 DB 2B A5 47 2B 3C 48 7D 2F AE 3C 6C 3E 82 66 45 51 B1 B3 BE EA EC 1E 31 8B 9F 12 42 77 BB 51 19 08 AD E3 7E 1A C9 FD 2C 21 FD A3 61 6A 8E C7 87 39 BA 2E 07 82 1F 0C 0D E9 5E 43 BE ED 00 6B A9 47 00 4C FB 22 79 01 14 F6 F8 82 8C E2 31 B9 27 C3 81 9D AC 57 7F 91 7A 21 72 17 C8 C8 C3 CF 60 86 CE 8E 96 54 4F 1A 82 A5 5E 03 F0 EB 6D 89	C1 4B B3 65 47 70 BC DD 4F 58 DB EC 9C ED C3 66 E5 1F 31 13 54 AD 4A 66 46 1F 2C 0A EC 64 07 E5 2E DC DC B9 0A 20 ED DF E3 C4 D0 9E 9A A9 7A 1D 82 88 E5 11 56 DB 1E 9F 58 C2 51 E7 2C 34 0D 2E D2 92 E1 56 CB F1 79 5F B3 BB 87 CA 25 03 7B 9A 52 41 66 10 60 4F 57 13 49 F0 E8 37 67 83 DF E7 D3 4B 67 4C 22 51 A6 DF 0E 99 10 ED 57 51 74 26	E2 3B E1 11 72 DE A8 A4 D3 A3 57 AA 50 A2 8F 0B 77 90 C9 A2 A5 EE 12 CE 96 5B 01 09 20 CC 01 93 A7 4E 30 B7 53 F7 43 C4 69 00 57 9D E2 8D 22 DD 87 06 40 00 81 09 CE CE 1B 83 BF DF CD 3B 71 46 E2 D6 66 C7 05 B3 76 27 16 8F 7B 9E 1E 95 7D EE B7 48 A3 08 DA D6 AF 7A 0C 39 06 65 7F 4A 5D 1F BC 17 F8 AB BE EE 28 D7 74 7F 7A 78 99 59 85 68	Length of modulus: 2048 bits

	2A C8 F0 71 29 41 64 CD 26 94 8D 6D 8A D0 5A 90 9E 22 FD 22 18 B7 C0 70 E2 86 18 87 F9 42 86 CA 74 79 EF 13 8B 2D 98 A2 73 2F 43 16 53 BC 2D AD C1 A6 DC 28 F5 60 C5 E7 86 FA A9 4C BE C8 90 A3 52 C8 34 50 9E 0F 79 F2 99 C2 2B C0 F1 7D 36 15 1A DD 7D 73 5E 55 7C 03 38 2B 8E 26 27 57 16 D8 98 19 6A 31 A7 8A 41 D8 36 7A 8B 0D B5 C5 FF 80 D4 D6 05 74 A4 DD DE 9A C4 5F BF F9 FA 99 A0 E7 92 25 D7 B3 32 B8 E1 D0 77 7E 11 39 43 12 8E 61	E2 7D C7 CA 62 2E 13 1B 7F 23 88 25 53 6F C1 34 58 00 8B 84 FF F8 BE A7 58 49 22 7B 96 AD A2 88 9B 15 BC A0 7C DF E9 51 A8 D5 B0 ED 37 E2 36 B4 82 4B 62 B5 49 9A EC C7 67 D6 E3 3E F5 E3 D6 12 5E 44 F1 BF 71 42 7D 58 84 03 80 B1 81 01 FA F9 CA 32 BB B4 8E 27 87 27 C5 2B 74 D4 A8 D6 97 DE C3 64 F9 CA CE 53 A2 56 BC 78 17 8E 49 03 29 AE FB 49 4F A4 15 B9 CE F2 5C 19 57 6D 6B 79 A7 2B A2 27 20 13 B5 D0 3D 40 D3 21 30 07 93 EA 99 F5	6E 5C 23 32 4B BF 4E C0 E8 5A 6D E3 70 BF 77 10 BF FC 01 F6 85 D9 A8 44 10 58 32 A9 75 18 D5 D1 A2 BE 47 E2 27 6A F4 9A 33 F8 49 08 60 8B D4 5F B4 3A 84 BF A1 AA 4A 4C 7D 3E CF 4F 5F 6C 76 5E A0 4B 37 91 9E DC 22 E6 6D CE 14 1A 8E 6A CB FE CD B3 14 64 17 C7 5B 29 9E 32 BF F2 EE FA D3 0B 42 D4 AB B7 41 32 DA 0C D4 EF F8 81 D5 BB 8D 58 3F B5 1B E8 49 28 A2 70 DA 31 04 DD F7 B2 16 F2 4C 0A 4E 07 A8 ED 4A 3D 5E B5 7F A3 90 C3 AF 27	
Key usages; how do they vary in the chain?	Signing, Key Encipherment	Signing Certificate Signer CRL Signer	Signing Certificate Signer CRL Signer	The intermediate and root certificate holders can use the certificate for signing other certificates, whereas the end user cannot.
Basic constraints, how do they vary in the chain?	Is not a Certification Authority	Is a Certification Authority Maximum number of intermediate CAs: 0	Is a Certification Authority Maximum number of intermediate CAs: unlimited.	The end certificate is not a CA, but the intermediate and root CAs have the maximum number of intermediate CAs length mentioned.
Name constraints (if any), how are these useful?	None	None	None	When a request is placed it succeeds if it is present in the permitted namespace else it fails if it is present in the excluded namespace.
Size of the certificate	2.8 kB	1.8 kB	1.4 kB	The size of the certificate decreases as we go up along the chain of trust, this is due to the

				absence of some fields such as SAN.
Any other parameters that you found interesting?				

Answer the following queries after filling out the above table:

1. Which certificate type (DV/OV/IV/EV) is more trustable and expensive?
   A: The highest-ranking and most expensive certificate type is an EV (Extended Validation) Certificate. To get this certificate Orgainsations need to go through an extended verification process. OV and IV certificates are more trustworthy than DV certificates.

2. What is the role of the Subject Alternative Name (SAN) field in the X.509 certificate?
   A: SAN is a structured way to indicate all of the domain names and IP addresses that are secured by the certificate. This is useful for organizations that have multiple domains as they can issue the same SSL certificate for all those domains.

3. Why are key usages and basic constraints different for root, intermediate, and end certificates?
   A: ***Key usages*** are different because, end certificates can be used for digital certification and key encipherment, they cannot be used for certificate signing whereas an essential usage of root and intermediate certificates is certificate signing along with the other usages.
   Since end certificates cannot be used for signing, the ***basic constraints*** just tell that whereas for intermediate certificates and root CAs it would also tell the *maximum length* of the intermediate CAs . Mostly there would be some limited path length for intermediate CAs and root CA can have any path length as a basic constraint.

4. What is the difference between the Signature value and the Thumbprint of a digital certificate?
   A: Signature Value: It is a CA-signed value of the information encoded in the certificate including things like the subject, the issuer, and the public key of the certificate.
   Thumbprint: It is a Cryptographic Hash Value (usually sha1 or sha256) computed on the entire certificate [9]

5. Why do RSA key lengths increase over the years? Why is ECDSA being preferred over RSA nowadays?
   A: RSA key lengths are increasing because vulnerabilities have been found in shorter key lengths. Attackers have found a brute force algorithm to break this in sub-exponential time. Due to high intense computations involved in RSA the time taken is much longer compared to ECDSA. By using ECDSA over RSA the key size is drastically

reduced and the performance is much higher hence it is preferred.

6. What are the pros and cons of pre-loading root and intermediate certificates in the root stores of browsers and OSes?
   A:
   Pros:
   - Browsers can save a lot of time during checking certificates if the root and intermediate certificates are pre-loaded since they will be used quite frequently. And also this will be quite efficient when there is a larger chain of trust.

   Cons:
   - Browsers need to periodically check and update the certificates stored. And also sometimes the certificates can take up large memory space.

7. Why are root CAs kept offline?
   A: Since the chain of trust begins with the root CAs, unauthorized access to root CAs can lead to very severe consequences. So, in order to maintain the integrity and security of root CAs, they are kept offline.

8. List out names of OS/Browser/Company whose root stores pre-populated with Root and Intermediate CA certificates of the website #N?
   A: DigiCert SSL Certificates are trusted virtually by every browser in use today as well as dozens of smartphones and handheld computing devices. Different browsers and devices supported are listed in the link [10].

PART-B

1. You have received the digital certificate of the website #N over email. How do you verify whether the certificate is valid without using any online tools or browsers? Write a pseudo-code of your verifier function named myCertChecker( ) and explain how it works by picking the entire chain of trust of an end-user cert (of the website #N) in PART-A of this assignment.
   A:

```
VALID_CERT=True
INVALID_CERT=False
def mycertChecker(domain d, certificate C):
    if regex_match(d, C.SANList):
        return VALID_CERT
    if C.subject == C.Issuer: # Root CA
        return VALID_CERT
    curTime = datetime.now()
    if C.Validity.notBefore > curTime or C.Validity.notAfter < curTime:
        return INVALID_CERT
    if not mycertChecker(C.subject.CN, C.IssuerCert): # recursively
checking for certificate in upper hierarchy
        return INVALID_CERT
    if C.sign != signatureAlgo(C,C.Issuer.PublicKey):
        return INVALID_CERT
    if not ( opensslVerify(C,C.Issuer.PublicKey) or openssl_ocsp(C)) :
        return INVALID_CERT

    return VALID_CERT
```

2. Consider the scenario in which evil Trudy has used the digital certificate of the website (Bob) named abc.com to launch her own web server with the domain name, **xyz.com**. Does your function myCertChecker( ) return valid or invalid for this when someone like Alice tries to access Trudy's website xyz.com from a browser like Chrome/Edge/Firefox?
   A: My myCertChecker() function returns an ***invalid*** certificate error as Trudy's domain name xyz.com doesn't match the common name or subject alternative name (SAN) in the digital certificate.

3. Consider the scenario in which evil Trudy has used the digital certificate of Bob's website abc.com to launch her own web server with the domain name, **xyz.com**. When a web client (Alice) tries to connect with Bob's website abc.com by sending a DNS query, Trudy responds with her IP address by launching MITM attack (What is DNS cache poisoning? | DNS spoofing | Cloudflare) Does your function myCertChecker( ) returns valid or invalid for this and what are the consequences? What kind of attacks can Trudy launch in this scenario?

A: myCertChecker() returns a *valid* certificate as Trudy copies the certificate of Bob's website and due to DNS Spoofing the domain name of Trudy is still visible as abc.com in web browser.

Consequences:

- Eventhough Trudy was able to spoof bob's website she can't decrypt any of the messages as she doesnt have the private key of Bob.

Attacks:

- DDOs attacks are always possible
- Integrity may be lost as even though Trudy can't Decrypt the messages she can tamper them and send to Alice.

Deliverables in GC:

- Certificates used for completing this assignment and a readable PDF Report with name "DCAsg-<RollNo>.PDF" compressed and encrypted with AES-256 using open source 7-zip file archiver tool with your RollNo (UPPERCASE) as the password.
    - In your report, also briefly explain how 7-zip uses the password to encrypt compressed files using secure hash and symmetric algorithms. What role does the password length play in brute force attacks to decrypt the encrypted files?

7Zip:

7-zip encrypts the file using the AES 256 algorithm. The key for this encryption is generated using SHA 256 function which would be iterated exponential times so that it would be harder to execute brute force attack. And the password for this encrypted file is the passphrase used in the hash function.
As password length increases, the time taken for brute force attack will be very large (exponential) and the computation is also intensive.

References:

1. https://crt.sh/
2. https://ahrefs.com/blog/most-visited-websites/
3. http://lapo.it/asn1js/#
4. http://phpseclib.sourceforge.net/x509/decoder.php
5. https://www.ssl.com/article/dv-ov-and-ev-certificates/
6. https://www.ccadb.org/
7. DV, OV, IV, and EV Certificates - SSL.com
8. 7-Zip (7-zip.org)
9. Digital Certificate "Signature and Fingerprint" - Information Security Stack Exchange
10. DigiCert SSL Certificate Browser Compatibility