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 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 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 (intermediate, if applicable)	Subject (CN) of certificate holder (root)	Remarks/observations
Issuer	**C = US O = Google Trust Services CN = GTS CA 101**	**OU - GlobalSign Root CA - R2 O = GlobalSign CN = GlobalSign**	**-**	**OU = GlobalSign Root CA - R2 O = GlobalSign CN = GlobalSign**	**Root certificate issuer is itself (self signed certificate) Thus issuer of intermediate and root certificate are same.**
Version No.	**3**	**3**	**-**	**3**	**All certificates are using latest version i.e. version 3 of X.509.**
Signature Algo	**sha256WithRSAEncryption**	**sha256WithRSAEncryption**	**-**	**sha1WithRSAEncryption**	**After 2016, all CA's sign the**

					certificate using sha2 algorithm. Thus intermediate and end-user certificates are signed using sha2 while root certificate which is quite old (issued in 2006) is signed with sha1.
Size of digest	**256 bits**	**256 bits**		**160 bits**	**Root certificate is using sha1 digest of length 160 bits while end-user and intermediate certificate is using sha2 digit of size 256 bits.**
Signature Value	**81:1f:a6:e0:af:21:0a:51:2f:a7:73:cf:16:fd:62:ec:ae:6f: da:ca:b5:7f:b7:16:26:79:1b:9a:d5:bf:b1:98:41:43:5e:74: 80:db:a6:7b:1f:d1:78:28:20:4f:05:90:53:79:bc:cc:98:a7:**	**1a:80:3e:36:79:fb:f3:2e:a9:46:37:7d:5e:54:16:35:ae:c7: 4e:08:99:fe:bd:d1:34:69:26:52:66:07:3d:0a:ba:49:cb:62: f4:f1:1a:8e:fc:11:4f:68:96:4c:74:2b:d3:67:de:b2:a3:aa:**		**99:81:53:87:1c:68:97:86:91:ec:e0:4a:b8:44:0b:ab:81:ac: 27:4f:d6:c1:b8:1c:43:78:b3:0c:9a:fc:ea:2c:3c:6e:61:1b: 4d:4b:29:f5:9f:05:1d:26:c1:b8:e9:83:00:62:45:b6:a9:08:**	**Signature value is used to verify integrity of certificate. If signature value of certificate matches with signature value obtained after applying signature**

	f3:9a:03:7a:5b:4e:b4:3f:3b:b5:4c:51:df:02:13:7b:13:ab: ff:c3:43:b5:00:31:98:19:85:49:20:af:06:5a:fb:70:a3:85: 76:57:90:9b:0d:00:6d:e9:b7:aa:21:97:fe:94:c2:cc:de:7d: f1:47:60:dd:8c:5f:87:d5:f8:9c:3b:1b:83:5c:81:f0:6b:72: 7d:5e:a2:1f:c0:4c:01:26:ef:13:77:cc:eb:93:5c:ce:dc:96: 9b:6b:50:3e:5e:3c:78:3f:0f:b1:3f:7d:d4:65:d6:7b:80:7f: 9d:26:80:82:44:98:13:eb:07:00:e7:bd:47:2b:23:8f:8c:55: 1c:07:b3:e1:30:b8:8b:7f:b9:67:99:e6:d9:c1:ac:8b:63:26: 03:84:0e:eb:	05:8d:84:4d:4c:20:65:0f:a5:96:da:0d:16:f8:6c:3b:db:6f: 04:23:88:6b:3a:6c:c1:60:bd:68:9f:71:8e:ee:2d:58:34:07: f0:d5:54:e9:86:59:fd:7b:5e:0d:21:94:f5:8c:c9:a8:f8:d8: f2:ad:cc:0f:1a:f3:9a:a7:a9:04:27:f9:a3:c9:b0:ff:02:78: 6b:61:ba:c7:35:2b:e8:56:fa:4f:c3:1c:0c:ed:b6:3c:b4:4b: ea:ed:cc:e1:3c:ec:dc:0d:8c:d6:3e:9b:ca:42:58:8b:cc:16: 21:17:40:bc:a2:d6:66:ef:da:c4:15:5b:cd:89:aa:9b:09:26: e7:32:d2:0d:6e:67:20:02:5b:10:b0:90:09:9c:0c:1f:9e:ad: d8:3b:ea:a1:		93:b9:a9:33:4b:18:9a:c2:f8:87:88:4e:db:dd:71:34:1a:c1: 54:da:46:3f:e0:d3:2a:ab:6d:54:22:f5:3a:62:cd:20:6f:ba: 29:89:d7:dd:91:ee:d3:5c:a2:3e:a1:5b:41:f5:df:e5:64:43: 2d:e9:d5:39:ab:d2:a2:df:b7:8b:d0:c0:80:19:1c:45:c0:2d: 8c:e8:f8:2d:a4:74:56:49:c5:05:b5:4f:15:de:6e:44:78:39: 87:a8:7e:bb:f3:79:18:91:bb:f4:6f:9d:c1:f0:8c:35:8c:5d: 01:fb:c3:6d:b9:ef:44:6d:79:46:31:7e:0a:fe:a9:82:c1:ff: ef:ab:6e:20:c4:50:c9:5f:9d:4d:9b:17:8c:0c:e5:01:c9:a0: 41:6a:73:53:f	algorithm applied on certificate, it means the certificate is not tampered with.

	42:9e:27:18:56:a9:4c:d6:2f:1d:1b:df:ed:a4: **f0:2a:e0:df:7b:1d:0b:80:ac:ea:b4:b7:3d:13:7f:4b:4b:ec:** **85:15:55:21:3f:c5:40:dc:74:de:fb:81:76:13:04:e3:33:90:** **62:d6:5a:60**	**fc:6c:e8:10:5c:08:52:19:51:2a:71:bb:ac:7a:** **b5:dd:15:ed:2b:c9:08:2a:2c:8a:b4:a6:21:ab:63:ff:d7:52:** **49:50:d0:89:b7:ad:f2:af:fb:50:ae:2f:e1:95:0d:f3:46:ad:** **9d:9c:f5:ca**		**a:a5:50:b4:6e:25:0f:fb:4c:18:f4:fd:52:d9:** **8e:69:b1:e8:11:0f:de:88:d8:fb:1d:49:f7:aa:de:95:cf:20:** **78:c2:60:12:db:25:40:8c:6a:fc:7e:42:38:40:64:12:f7:9e:** **81:e1:93:2e**	
Validity period	**85 days**	**4 years 6 months**		**15 years**	**End-user certificates usually have low validity period and intermediate certificates have more validity period and root certificates last much longer (in order of 10-15 years). That is as we go up in the hierarchy, validity period increases.**
Subject (CN)	***.google.com**	**GTS CA 101**		**GlobalSign**	**End user certificate has wildcard domain. (*.google.co**

					m) while intermediate and root certificates have single domain.
Certificate type: DV, IV, OV or EV? Tell also how you are able to determine the type!	**OV OV certificates mentioned organization details in subject field, since this certificate contains subfields like C, ST, L, O, CN under Subject name field, it is OV certificate.**	**OV OV certificates mentioned organization details in subject field, since this certificate contains subfields like C, O, OU, CN under Subject name field, it is OV certificate.**		**OV OV certificates mentioned organization details in subject field, since this certificate contains subfields like C, O, OU, CN under Subject name field, it is OV certificate.**	**All the three certificates are organization validated. Thus, it ensures the identity of organization. This adds an extra layer of security over DV certificates and thus user can trust the organization.**
Subject Alternative Name (SAN), if any	***.google.com, DNS:*.android.com, DNS:*.appengine.google.com, DNS:*.bdn.dev, DNS:*.cloud.google.com, DNS:*.crowdsource.google.com, DNS:*.datacompute.google.com, DNS:*.flash.android.com, DNS:*.g.co, DNS:*.gcp.gvt2.com,**	**-**		**-**	**SAN of intermediate and root certificates are none. While google (end-user) certificates control a lot of wildcard domains like *.google.com, *android.com, etc. Thus all these domains are certified under same end-user certificate.**

	DNS:*.gcpcdn.gvt1.com, DNS:*.ggpht.cn, DNS:*.gkecnapps.cn, DNS:*.google-analytics.com, DNS:*.google.ca, DNS:*.google.cl, DNS:*.google.co.in, DNS:*.google.co.jp, DNS:*.google.co.uk, DNS:*.google.com.ar, DNS:*.google.com.au, DNS:*.google.com.br, DNS:*.google.com.co, DNS:*.google.com.mx, DNS:*.google.com.tr, DNS:*.google.com.vn, DNS:*.google.de, DNS:*.google.es, DNS:*.google.fr, DNS:*.google.hu, DNS:*.google.it, DNS:*.google.nl, DNS:*.google.pl, DNS:*.google.pt, DNS:*.googl				

	eadapis.com, DNS:*.googleapis.cn, DNS:*.googlecnapps.cn, DNS:*.googlecommerce.com, DNS:*.googlevideo.com, DNS:*.gstatic.cn, DNS:*.gstatic.com, DNS:*.gstaticcnapps.cn, DNS:*.gvt1.com, DNS:*.gvt2.com, DNS:*.metric.gstatic.com, DNS:*.urchin.com, DNS:*.url.google.com, DNS:*.wear.gkecnapps.cn, DNS:*.youtube-nocookie.com, DNS:*.youtube.com, DNS:*.youtubeeducation.com, DNS:*.youtubekids.com, DNS:*.yt.be, DNS:*.ytimg.com, DNS:android.clients.google.com, DNS:android.com, DNS:developer.android.				

	google.cn, DNS:developers.android.google.cn, DNS:g.co, DNS:ggpht.cn, DNS:gkecnapps.cn, DNS:goo.gl, DNS:google-analytics.com, DNS:google.com, DNS:googlecnapps.cn, DNS:googlecommerce.com, DNS:source.android.google.cn, DNS:urchin.com, DNS:www.goo.gl, DNS:youtu.be, DNS:youtube.com, DNS:youtubeeducation.com, DNS:youtubekids.com, DNS:yt.be				
Certificate category: Single domain, wildcard or SAN/UCC cert?	**SAN(Subject alternate names)**	**Single domain**		**Single domain**	**End-user certificate is UCC (unified communication certificate)/SAN since under that certificate, multiple**

					domains are protected. While root and intermediate certificates are single-domain.
Public Key Info like key algo, key length, public exponent (e) in case of RSA	**Algo - Elliptic Curve, Key length - 256 bit**	**Algo - RSA, Key length - 2048 bit, Public exponent = 65537 (0x10001)**		**Algo - RSA, Key length - 2048 bit, Public exponent = 65537 (0x10001)**	**Public exponent used in RSA algo is 65537. This is known as a short public exponent and it significantly improves the performance of RSA verification. Also, public exponent is compatible with existing hardware and software.**
Public key or modulus (n) in case of RSA	**-**	**00:d0:18:cf:45:d4:8b:cd:d3:9c:e4:40:ef:7e:b4: dd:69:21:1b:c9:cf:3c:8e:4c:75:b9:0f:31:19:84: 3d:9e:3c:29:ef:50:0d:10:93:6f:05:80:80:9f:2a: a0:bd:12:4b:**		**00:a6:cf:24:0e:be:2e:6f:28:99:45:42:c4:ab:3e: 21:54:9b:0b:d3:7f:84:70:fa:12:b3:cb:bf:87:5f: c6:7f:86:d3:b2:30:5c:d6:fd:ad:f1:7b:dc:e5:f8: 60:96:09:92:**	**ECC is used in end-user certificate. Thus no public key modulus. While RSA is used in intermediate and root key certificates. Key length is 2048 bits.**

		02:e1:3d:9f:58:16:24:fe:30:9f:0b: 74:77:55:93:1d:4b:f7:4d:e1:92:82:10:f6:51:ac: 0c:c3:b2:22:94:0f:34:6b:98:10:49:e7:0b:9d:83: 39:dd:20:c6:1c:2d:ef:d1:18:61:65:e7:23:83:20: a8:23:12:ff:d2:24:7f:d4:2f:e7:44:6a:5b:4d:d7: 50:66:b0:af:9e:42:63:05:fb:e0:1c:c4:63:61:af: 9f:6a:33:ff:62:97:bd:48:d9:d3:7c:14:67:dc:75: dc:2e:69:e8:f8:6d:78:69:d0:b7:10:05:b8:f1:31: c2:3b:24:fd:1a:33:74:f8:23:e0:ec:6b:19:8a:16: c6:e3:cd:a4:cd:0b:db:b3:a4:59:60:38:88:3b:ad: 1d:b9:c6:8c:		10:f5:d0:53:de:fb:7b:7e:73:88:ac: 52:88:7b:4a:a6:ca:49:a6:5e:a8:a7:8c:5a:11:bc: 7a:82:eb:be:8c:e9:b3:ac:96:25:07:97:4a:99:2a: 07:2f:b4:1e:77:bf:8a:0f:b5:02:7c:1b:96:b8:c5: b9:3a:2c:bc:d6:12:b9:eb:59:7d:e2:d0:06:86:5f: 5e:49:6a:b5:39:5e:88:34:ec:bc:78:0c:08:98:84: 6c:a8:cd:4b:b4:a0:7d:0c:79:4d:f0:b8:2d:cb:21: ca:d5:6c:5b:7d:e1:a0:29:84:a1:f9:d3:94:49:cb: 24:62:91:20:bc:dd:0b:d5:d9:cc:f9:ea:27:0a:2b: 73:91:c6:9d:1b:ac:c8:cb:e8:e0:a0:f4:2f:90:8b: 4d:fb:b0:36:	

		a7:53:1b:fc: bc:d9:a4:ab: bc:dd:3c: 61:d7:93:15: 98:ee:81:bd: 8f:e2:64:47:2 0:40:06: 4e:d7:ac:97: e8:b9:c0:59: 12:a1:49:25: 23:e4:ed: 70:34:2c:a5: b4:63:7c:f9: a3:3d:83:d1: cd:6d:24: ac:07		1b:f6:19:7a: 85:e0:6d:f2: 61:13:88: 5c:9f:e0:93:0 a:51:97:8a:5 a:ce:af:ab:d 5:f7:aa: 09:aa:60:bd: dc:d9:5f:df:7 2:a9:60:13:5 e:00:01: c9:4a:fa:3f:a 4:ea:07:03:2 1:02:8e:82:c a:03:c2: 9b:8f	
Key usages	**Critical Digital signature**	**Critical Digital Signature Certificate Sign CRL Sign**		**Critical Certificate Signer, CRL Sign**	**Intermediate and root certificates have “certificate sign” as extension which means they have the ability to sign other certificates. Also, they have CRL sign extensions which they have ability to sign CRL list of other CAs.**
Basic constraints	**Critical CA: FALSE**	**Critical CA: TRUE Max Path length: 0**		**Critical CA: TRUE Max path length:**	**Basic constraints are used to mark**

				Unlimited	**certificates as belonging to a CA giving them the ability to sign other certificates. Intermediate and root certificates have CA as true that is they are CA's and other certificates can be signed by them. Maxpath length determines how many subordinate CAs can exist beneath it. Root certificate has unlimited max path length here while for intermediate certificate, it has 0 max path length which means no other subordinate CA can exists beneath it.**
Name constraints, how these	**Not present**	**Not present**		**Not present**	**-**

are useful?					
Size of the certificate	**3414 bytes**	**1574 bytes**		**1376 bytes**	**Size of certificate decreases as we go up in hierarchy (lowest size of root certificate)**
Any other parameter?	**Certificate Fingerprint: SHA1:0F 7A F7 71 EF 6D 79 66 02 B3 13 71 05 79 98 BA 02 3C A6 DF MD5: 9D CA 93 64 4C 2F 4A D8 95 F9 15 1E E3 F9 55 3E Serial Number: c4:ea:98:ea:7e:5e:1f:43:02:00:00:00:00:87:01:82**	**Certificate Fingerprint: SHA1: A0 31 C4 67 82 E6 E6 C6 62 C2 C8 7C 76 DA 9A A6 2C CA BD 8E MD5: AA EE 5C F8 B0 D8 59 6D 2E 0C BE 67 42 1C F7 DB Serial Number: 01:e3:b4:9a:a1:8d:8a:a9:81:25:69:50:b8**		**Certificate Fingerprint: SHA1:5F B7 EE 06 33 E2 59 DB AD 0C 4C 9A E6 D3 8F 1A 61 C7 DC 25 MD5: D4 74 DE 57 5C 39 B2 D3 9C 85 83 C5 C0 65 49 8A Serial Number: 04:00:00:00:00:01:0f:86:26:e6:0d**	**SHA1 fingerprint length is 160 bit and MD5 length is 128 bit. Fingerprints are used for assurance that the certificate is not tampered with. Serial Number is unique to each certificate.**

Answer the following queries after filling out the above table:

1. Which certificate type (DV/OV/IV/EV) is more trustable and expensive?
   OV (Organization validation), IV(Individual validation) are trustable and expensive as they require more validation aur effort by CA to validate. While EV (Extended validation) certificates are most trustable and expensive. It requires maximum effort by CA for issuing this certificate. It requires 18 validation checks by CA. It may be issued to businesses and other organizations and not to individuals. It has most details under the subject field in the certificate as compared to other types of certificates.

2. What is the role of Subject Alternative Name (SAN) field?
   Subject alternative name field specifies multiple host names (domain names) to be protected by that particular single certificate. As in above table, end-user certificate google trust services (GTS) include multiple domain names like google.com, google.fr, google.us, youtube.com, android.com etc in SAN field.

3. Why are key usages and basic constraints different for root, intermediate and end certificates?
   Key usage extensions restrict what a certificate can be used for. If the extensions are present, then only those listed usages are allowed. If no extensions are present, then there are no restrictions. Different types of key usage extensions are digital signature, key encipherment, data encipherment, key cert sign, crl sign. For CA certificates, key usage “Certificate signing” is used when subject public key is used to verify signature on certificates. “CRL signing” is used when subject public key is used to verify signature on revocation information like CRL. It is also applicable for CAs. Since root, intermediate and end certificates have different restrictions, key usages are different on each of them. Basic constraints extension are used to mark certificates as belonging to a CA, giving them the ability to sign other certificates. They are also used to express the maximum depth of the trust path from the CA.
   Basic constraints of certificates include subfields like CA. So for end-user certificate, CA subfield is false which means it can’t issue certificates to others while usually for intermediate and always for root certificates, CA is set to true which means it can issue certificates to others.

4. Why do RSA key lengths increase over the years? Why ECDSA is being preferred over RSA now-a-days?
   RSA key lengths are increasing over years due to increased computational power and improved efficient integer factorization algorithms. Length of key in RSA determines its strength. Factorization of large numbers is becoming possible due to increased computational power. Challenge for breaking RSA-155 (512 bits) was broken in 1999 which took 6 months on advanced hardware. (part of RSA factoring challenge)
   ECDAS offers same level of security that RSA provides for smaller key lengths. For example, 160 bit length ECC key offers same level of security as RSA key of 1024 bit length. Thus, if requires less computational power and overhead. As a result, it offers great performance and scalability. Thus it is appropriate for low-end devices with less computational power. In TLS certificates, it reduces handshake time and thus speeds up the loading of website. Thus, ECDSA is preferred over RSA now-a-days due to its performance and same level of security for smaller key lengths which make it desirable for every range of devices which include lower-end devices also and due to faster response times.

5. What are pros and cons of pre-loading root and intermediate certificates in the root stores of browsers and OSes?
   Main advantage of root store is to ensure that trusted root certificates are present in OSes and browsers which will further ensure secure connections with other websites and networks. These popular and famous vendors like Microsoft, Google, Apple, Mozilla can be trusted by customers. So if they preinstalled the certificates of root CAs and trusted intermediate CAs, it would be convenient and secure for customers to connect to other websites.

The main disadvantage of preloaded root stores is that we have to trust them and it may happen that they may issue fake certificates to some websites or sometimes untrustworthy root certificates may come preinstalled on some OSes due to negligence of some root CAs which is rare. Some incidents have happened like this. For example - Lenovo and Dell were selling computers with adware of Superfish preinstalled which installed a self-signed root HTTPS certificate in the computer and which made it possible for attackers to hijack web sessions and carry out man-in-the-middle attacks. Another incident where CNNIC (Chinese internet network information center) issued intermediate certificate was used by security business in Egypt to create unauthorized SSL certificates that tricked people into connecting to fake gmail.com and google.com and stole password. After this Mozilla and Google removed their certificates from their root stores. Another such incident happened with Symantec where they issued fake Google security certificates.

6. Why are root CAs kept offline?
   Root CAs issue certificates to intermediate or subordinate CAs. Thus now intermediate CAs can now issue, revoke, redistribute certificates to others. Since root CAs are most trustworthy and thus, any threat or unauthorized access to root CA would have far reaching repercussions. Because then root CA would then have to re-issue certificates to each intermediate CA in the chain of trust. So to prevent this, root CAs are mostly kept in offline state. It is brought online when tasks like issuing certificates to intermediate CAs are required which are very rare. Thus root CAs are kept offline to ensure security and integrity of PKI.

PART-B

1. You have received a digital certificate of a firm M over email. How do you verify whether the certificate is valid without using any of online tools or browsers? Write a psuedo-code of your verifier function named myCertChecker( ) and explain how it works by picking the chain of trust of an end-user cert in PART-A of this assignment.
   Following is the function myCertChecker which takes digital certificate of firm M and verifies if its is valid.
   Function myCertChecker(certificate)
   1. Check if it is still valid by checking if current date lies inside validity (expiration) period mentioned in certificate. If current data doesn't lie in between validity period, it means certificate is expired and is unlikely to be trusted.
   2. Check if domain name specified or in SAN fields (CN in subject) is same as domain name of website (web server).
   3. Check if issuer of certificate is trusted CA by checking the CA in browser or OS root stores.
   4. Check if this certificate does not lie in CRL list of browser and OS root stores'

certificates. If it lies in CRL list of any CA, then it can't be trusted.

5. Check if the hash obtained by applying hashing algorithm mentioned in certificate is same as digital signature mentioned in certificate. If two hashes doesn't match, it means certificate has been tampered and can't be trusted.
6. Traverse from its issuer (intermediate) certificate of end-user upto root certificate verifying all above steps for each certificate.
7. If any of the above checks fails, return invalid else return valid.

2. Consider the scenario in which evil Trudy has used the digital certificate of firm M to launch her own web server. Does your function myCertChecker( ) returns valid or invalid for this when someone tries to access Trudy's website from a browser like Chrome/Edge/Firefox?
   Since, Trudy is launching web server on his own domain using digital certificate of firm M, subject (CN) will be different from Trudy's website domain (step 2 will fail), function myCertChecker() will return invalid. Also since his domain is new, it wouldn't be listed in any of CA, so can't be trusted.

Explain how 7-zip uses the password to encrypt compressed files using secure hash and symmetric algorithms. What role password length plays in brute force attacks to decrypt the encrypted files?

7zip uses symmetric encryption algorithm AES-256 in CBC mode which uses cipher key of length 256 bits. For generating this, 7zip uses key derivation function (PBKDF) based on SHA-2. The derivation function takes password as input from user and generates the key which is used by AES algorithm. 7zip uses a large number of iterations to generate cipher key from password. PBKDF also uses salt (512 bit) to decrease vulnerabilities to dictionaries and rainbow table attacks. 7zip uses 2^{18} iterations to derive key from password. As the length of password increases, the computational power and time required on an average to perform brute force attack successfully increases exponentially.

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. 7-Zip (7-zip.org)
8. https://www.digicert.com/difference-between-dv-ov-and-ev-ssl-certificates
9. https://www.northeastern.edu/securenu/sensitive-information-2/how-to-use-7-zip-to-encrypt-files-and-folders/
10. https://en.wikipedia.org/wiki/Offline_root_certificate_authority#:~:text=A%2

0common%20method%20to%20ensure,of%20certificates%20authorizing%20intermediate%20CAs.
11. https://expeditedsecurity.com/blog/control-the-ssl-cas-your-browser-trusts/
12. https://arstechnica.com/information-technology/2015/02/lenovo-pcs-ship-with-man-in-the-middle-adware-that-breaks-https-connections/
13. https://arstechnica.com/information-technology/2015/11/dell-does-superfish-ships-pcs-with-self-signed-root-certificates/
14. https://www.theregister.com/2015/04/02/mozilla_revokes_cnnic_cert_trust/
15. https://www.itpro.co.uk/security/25315/symantec-employees-fired-over-fake-security-certificates
16. https://sectigostore.com/blog/ecdsa-vs-rsa-everything-you-need-to-know/#:~:text=Compared%20to%20RSA%2C%20ECDSA%20is%20a%20less%20adopted%20encryption%20algorithm.&text=RSA%20requires%20longer%20keys%20to,RSA%20slows%20down%20the%20performance.
17. https://www.feistyduck.com/library/openssl-cookbook/online/ch-openssl.html
18. https://developer.mozilla.org/en-US/docs/Mozilla/Security/x509_Certificates
19. https://crypto.stackexchange.com/questions/3110/impacts-of-not-using-rsa-exponent-of-65537
20. https://security.stackexchange.com/questions/210069/encryption-using-7z-or-zip-file
21. https://security.stackexchange.com/questions/29375/is-7-zips-aes-encryption-just-as-secure-as-truecrypts-version
22. https://security.stackexchange.com/questions/240072/are-files-encrypted-with-7zip-vulnerable-to-password-crackers
23. https://help.hcltechsw.com/domino/11.0.0/admin/conf_keyusageextensionsandextendedkeyusage_r.html#:~:text=Note%3A%20The%20digital%20signature%20and,default%20for%20all%20Internet%20certificates.&text=Use%20when%20the%20public%20key,certificate%20signing%2C%20or%20CRL%20signing.
24. https://crypto.stackexchange.com/questions/3931/is-512-bit-rsa-still-safe-for-signature-generation
25. https://en.wikipedia.org/wiki/RSA_numbers\
26. http://www.math.ucsd.edu/~crypto/students/cert.html#:~:text=To%20validate%20the%20digital%20signature%20person%20authenticating%20the%20certificate%20will,and%20the%20certificate%20is%20authenticated.
27. https://en.wikipedia.org/wiki/Brute-force_attack

PLAGIARISM STATEMENT <Include it in your report>

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: Ajinkya Bokade
Date: 03/02/2021
Signature: AB