Sec+ 501 Get Certified Topic 10 – Understanding Cryptography and PKI

CH 10 – CRYPTOGRAPHY CONCEPTS

- Integrity – Provides assurance that data has not been modified.
  - Hashing provides assurances that data has not been modified and retains integrity
  - Hash: No. derived from performing a calculation on data such as a message, patch or file.
    - Creates fixed-size string of bits or hexadecimal characters which cannot be reversed to recreate original data
      - E.G: MD5, SHA256
- Confidentiality – Ensures data is only viewable by Authorized Users
  - Encryption scrambles or ciphers, data to make it unreadable if intercepted.
    - Symmetric – Same key to encrypt/decrypt
    - Asymmetric
      - Public and private key
      - Requires PKI to issue certificates
      - Encrypting with public = Decryption with matching private key
      - Encrypting with private = Decryption with matching public key
  - Stream ciphers encrypt data 1 bit at a time. Block ciphers encrypts data in blocks.
  - Digital Signature
    - Provides authentication to validate identity
    - Provides non-repudiation to prevent party from denying an action.
      - Used in signing emails which is a hash of an email message encrypted with sender's private key
      - Only sender's public key can decrypt hash providing verification.
- Hash – will always be the same if data is the same, if you execute the hash algorithm against data.

    - Hashes are created at least twice so that they can be compared
        - Patch File: Patch_v2_3.zip
        - SHA-1 Checksum: d0q8345q983qjowhefq894
            - In hexadecimal
- **Remember this:**
    - **Hashing verifies integrity for data such as email, downloaded files, and files stored on a disk. A hash is a number created with a hashing algorithm, and is sometimes listed as a checksum.**
- MD5 – Message Digest 5
    - Common hashing algorithm produces 128-bit hash
    - Not very good to use as it is old and crackable but it is still used in email, files downloaded etc.
    - Verifies integrity of files
- SHA – Secure Hash Algorithm
    - SHA-0: Not used
    - SHA-1: Creates 160 bit hashes.
    - SHA2- Improves SHA-1
        - SHA256 for 256 bit hashes and SHA512 for 512 bit hashes.
        - SHA224 and SHA384 create truncated versions of SHA256 and SHA512.
    - SHA-3: Alternative to SHA-2. Was created outside of NSA
        - Creates same hashes of SHA-2
    - As MD5 verifies integrity of files, SHA also verifies file integrity
    - HIDS and antivirus capture hashes of files on a system on scans
- HMAC – Hash-based authentication protocol
    - HMAC: Fixed length string of bits similar to other hashing algorithms
        - HMAC-MD5 & HMAC-SHA1
    - HMAC also uses a shared secret key to add randomness to result and only sender/receiver know the key.
    - Provides both integrity and authenticity of messages.
    - TLS and IPsec use versions of HMAC such as HMAC-MD5 & HMAC-SHA1

- **Remember this:**
  - **2 popular hashing algorithms used to verify integrity are MD5 and SHA. HMAC verifies both integrity and authenticity of a message with use of shared secret. IPSec and TLS use HMAC-MD5 and HMAC-SHA1**
- RIPEMD – Race Integrity Primitives Message Digest
  - Another hash function used for integrity but isn't widely used as MD5, SHA and HMAC
  - Can create 160,128,256,320 bit hashes. E.G: RIPEMD-160
- Hashing Files
  - Hash will always be same no matter how many times you calculate it
  - Hashing verifies file has retained integrity
  - Hashes – one way functions. Cannot use hash to reproduce original data.
- **Remember this:**
  - **Hashing is a one way function that creates a string of characters. You can't reverse the hash to recreate the original file. Passwords are often stored as hashes instead of storing actual password. Applications often salt passwords with extra characters before hashing them.**
- Key Stretching
  - Key Stretch: increases strength of stored password, salting passwords with additional random bits to make them more complex
    - Used via 'Bcrypt' and 'PBKDF2'
  - Bcrypt – Based on blowfish block cipher
    - Salts password by adding additional bits before encrypting with blowfish. Does this process multiple times.
    - Results in a 60 character string.
  - PBKDF2 – Password Based Key derivative function
    - Salts 64 bits using pseudo random function such as HMAC
    - WPA2, iOS and Cisco systems use PBKDF2.
    - Can process 1 million + times to create hash
    - Sizes range from 128,256,512 bits

- **Remember this:**
  - **Bcrypt and PBKDF2 are key stretching techniques that help prevent brute force and rainbow table attacks. Both salt password with additional random bits.**
- If hashes are changed, message loses integrity

Algorithm	Type	Comments
MD5	Hashing - Integrity	Creates 128bit hashes
SHA-1	Hashing - Integrity	Creates 160bit hashes
SHA-2	Hashing - Integrity	Creates 224,256,384,512 bit hashes
SHA-3	Hashing - Integrity	Creates 224,256,384,512 bit hashes
HMAC-MD5	Integrity/Authenticity	Creates 128bit hashes
HMAC-SHA1	Integrity/Authenticity	Creates 160bit hashes

- **Remember: Hashing algorithms don't encrypt data**
- Data at Rest- Data stored in media and is common to encrypt data
- Data in transit – Data sent over network and is common to encrypt sensitive data in transit.
- Data in use – Data being used by a computer. Data is not encrypted while in use as computer needs to process it. If data is encrypted, application will need to decrypt it.
- Algorithm – Performs mathematical calculations on data. Algorithm is always the same.
- Key – Number that provides variability for encryption. Either private/public and should be changed frequently.
- **Remember this:**
  - **Encryption provides confidentiality and helps ensure data is viewable only to authorized users. Applies to data at rest such as data in a database or data in transit being sent over the network**
- Encryption Terms
  - Random and Pseudo random numbers
    - Appears to be random but is created by an algorithm.
  - IV – Initialization Vector
    - Starting value for a cryptographic algorithm.

        - A fixed size random/pseudo-random number to create random encryption keys.
        - IV should be large
    - Nonce
        - Number used once
    - XOR
        - Logical operation used in some encryption schemes.
        - XOR compares two inputs.
            - If inputs are same they are True or binary 1.
            - If different outputs false or binary 0.
    - Confusion
        - Ciphertext is significantly different from plaintext.
    - Diffusion
        - Ensure small changes in plaintext result in large changes in ciphertext
        - E.G: Changing single character in plaintext results in completely different ciphertext.
    - Secret Algorithm
        - Algorithm kept private.
        - Experts discourage this practice as it prevents review of algorithm by experts.
        - Most algorithms are known such as SHA etc.
    - Weak/Deprecated Algorithms
        - Algorithms that can be cracked allowing attacker to easily convert ciphertext→plaintext.
            - E.G: Flaws in SSL leads to people using TLS now instead.
    - High resiliency –
        - Security of encryption key if an attacker discover part of the key.
        - High resiliency prevents leakage of key. Use Strong algorithms to combat this.
- **Remember this:**
    - **Random numbers are picked by chance. Pseudo-random numbers appear to be random but are created by deterministic algorithms, meaning that given the same input, a pseudo-random no. generator will create same**

output. Confusion indicates ciphertext is significantly different than plaintext. Diffusion cryptographic techniques ensure small changes in plaintext result in significant changes in ciphertext.

- Block cipher:
  - Encrypts data in specific sized blocks such as 64bit or 128bit blocks
    - Divides Large files or messages into these blocks and encrypts each individual block separately
  - More efficient when size of data is known such as encrypting a file or a database specific-sized field.
- Stream cipher:
  - Encrypts data a stream as a stream of bits or bytes rather than dividing it into blocks
  - Stream ciphers encrypt data 1 bit at a time.
  - More efficient than block ciphers when size of data is unknown or sent in a continuous stream
  - Encryption keys should never be reused
- **Remember this:**
  - **Stream ciphers encrypt data a single bit or single byte at a time in a stream. Block ciphers encrypt data in a specific-sized block such as 64bit or 128bit blocks. Stream ciphers are more efficient than block ciphers when encrypting data in a continuous stream.**
- Cipher Modes
  - ECB – Electronic Code Book
    - Divide plaintext into blocks and encrypts each block using same key.
    - Weakness: If plaintext blocks are the same, cipher text will be the same making it easier to crack
    - Not recommended
  - CBC – Cipher Block Chaining
    - Used in Symmetric block ciphers.
    - Uses IV for randomization when encrypting 1st block. Combines each subsequent block with previous block using XOR operation.

        - Weakness: Suffers from pipeline delays making it less efficient due to each block being dependent on the previous
    - CTM – Counter Mode
        - Converts Block cipher into stream cipher
        - Combines IV with counter and uses result to encrypt each plaintext block.
        - Each block uses same IV but CTM combines with counter value, resulting in different encryption key for each block.
        - Widely used
    - GCM – Galois/Counter Mode
        - Used in Block ciphers
        - Combines CTM with Galois.
        - Provides data authenticity (integrity) and confidentiality.
        - With encryption of data, also uses hashing techniques for integrity.
        - Widely used due to efficiency and performance
- **Remember this:**
    - **ECB (Electronic code book) mode of operation is deprecated and shouldn't be used. CBC (Cipher Block Chaining) combines each block with previous block when encrypting data and suffers from pipelining delays. CTM combines an IV with a counter to encrypt each block. GCM (Galois/Counter Mode) combines counter mode with hashing techniques for integrity.**
- Symmetric Encryption
    - Uses Same key to encrypt and decrypt data.
    - Symmetric Encryption Algorithms change keys more often than once a day.
- **Remember this:**
    - **Symmetric Encryption uses same key to encrypt and decrypt data. For example, when transmitting encrypted data, symmetric encryption algorithms use same key to encrypt and decrypt data at both ends of transmission media. RADIUS uses symmetric encryption.**

- Symmetric Encryption Summary: * - Not recommended for use

Algorithm	Type	Method	Key Size
AES	Symmetric Encryption	128-bit block cipher	128,192,256 bit key
3DES	Symmetric Encryption	64-bit block cipher	56, 112, 168 bit key
Blowfish	Symmetric Encryption	64-bit block cipher	32 to 448 bit key
Twofish	Symmetric Encryption	128-bit block cipher	128,192,256 bit key
RC4 *	Symmetric Encryption	Stream Cipher	40 to 2048 bit-key
DES *	Symmetric Encryption	64-bit block cipher	56 bit key

Remember this: Encryption keys don't hash data. Remember table.

- AES – Advanced Encryption Standard. Mostly used
  - Fast
  - Efficient
  - Strong
  - Block cipher
- DES – Data Encryption Standard
  - Symmetric but not recommended as it can be broken
  - Block cipher
- 3DES – Improvement of DES
  - Encrypts data using DES algorithm in 3 separate passes and multiple keys.
  - Like DES encrypts in 64 bit blocks
  - Not used as often as AES as AES is less resource intensive. 3DES is an alternative for Non-AES legacy hardware support
  - Block Cipher
- RC4 – Rivest Cipher
  - **Only stream cipher**
  - Use AES instead as RC4 has been broken by government agencies
- Blowfish
  - Strong symmetric block cipher

    - Faster than AES and encrypts data in smaller 64-bit blocks where AES encrypts data in 128-bit blocks
- Twofish
    - Related to blowfish but encrypts data in 128-bit blocks and supports 128,192 and 256 bit keys.
- **Remember this:**
    - **RC4 is a strong symmetric stream cipher, but most experts recommend using AES instead today. Blowfish is a 64-bit block cipher and Twofish is a 128-bit block cipher. Blowfish is faster than AES-256.**
- Asymmetric Encryption
    - Uses 2 keys in a matched pair to encrypt/decrypt data
        - Public and private key
    - Encrypting with public = Decryption with matching private key
    - Encrypting with private = Decryption with matching public key
    - Private keys are kept private and never shared.
    - Public keys are freely shared by embedding them in a shared certificate.
- **Remember this:**
    - **Only a private key can decrypt information encrypted with a matching public key. Only a public key can decrypt information encrypted with a matching private key. A key element of asymmetric encryption methods is that they require a certificate and PKI.**
    - **Asymmetric is strong but resource intensive.**
- Certificate
    - Digital document that includes public key and information on owner of certificate.
    - CA – Certificate Authorities manage certificates
    - Includes:
        - Serial Number – Uniquely identifies Certificate. CA uses serial number to validate/revoke certificate. **If CA revokes certificate, it publishes serial number in a certificate revocation list (CRL)**
        - Issuer – Identifies CA issuing certificate
        - Validity Date
            - ‘Valid from’ and ‘Valid To’

        - Subject
            - Owner of certificate
        - Public Key
            - RSA Asymmetric Encryption uses public key in combination with matching private key
        - Usage
            - Use of certificate such as encryption or authentication or multiple stages
- **Remember this:**
    - **Certificates are an important part of asymmetric encryption. Certificates include public keys with details on owner of certificate and on the CA that issued the certificate. Certificate owners share public key by sharing a copy of their certificate.**
- RSA – Asymmetric Encryption (Rivest, Shamir, Adleman)
    - RSA is widely used to protect data such as email and other data transmitted over the internet. Uses both public key and private key in a matched pair.
    - Uses recipient's public key to encrypt symmetric key and recipient's private key decrypts it.
    - Uses static keys
- Asymmetric Keys
    - Static Keys – semipermanent and stay the same over a long period of time
        - Used in RSA
        - Certificate includes public key matched to a private key and the key pair is valid for lifetime of a certificate such as for 1 year.
    - Ephemeral Keys – very short lifetime and is recreated for each session.
        - Private Ephemeral key and public ephemeral key
        - Use these key pairs for one session and then discards.
    - Diffie Hellman uses static keys and some version of ephemeral
    - Perfect Forward Secrecy – Characteristic that ephemeral keys have with asymmetric encryption

        - Cryptographic system generates random public keys for each session and doesn't use a deterministic algorithm to do so.
        - Ensures system don't reuse keys
- ECC – Elliptic Curve Cryptography
    - Used in small wireless devices as it doesn't take much processing power to achieve security.
    - Uses mathematical equations to formulate elliptical curve.
- DH – Diffie Helman
    - Key exchange algorithm used to privately share a symmetric key between 2 parties. Once 2 parties know symmetric key, they use symmetric encryption to encrypt data.
    - Supports both static and ephemeral keys
    - EDH/DHE
        - Diffie Helman Ephemeral uses ephemeral keys generating keys for each session.
    - ECDHE – Elliptic Curve Diffie Helman Ephemeral
        - Uses ephemeral keys generated using ECC
        - ECDH – uses static keys (Elliptic Curve Diffie Hellman)
- **Remember this:**
    - **Diffie-Helman is a secure method of sharing symmetric encryption keys over a public network. Elliptic curve cryptography is commonly used with small wireless devices. ECDHE is a version of Diffie-Hellman that uses elliptic curve cryptography to generate encryption keys**
- Steganography
    - Hides messages or data within a file. Use hashing to detect changes in files indicating steganography
    - Hide data by manipulating bits
    - Hide data in white space of a file.
- **Using Cryptographic Protocols – Remember this**
    - Email digital Signatures
        - Sender private key encrypts/signs
        - Sender public key decrypts
    - Email Encryption
        - Recipient public key encrypts
        - Recipient private key decrypts

    - Website encryption
        - Web site public key encrypts
        - Web site private key decrypts
        - Symmetric key encrypts data in the web site session.
- Digital Signatures:
    - Digital Signatures Provide
        - Authentication
        - Non-Repudiation
        - Integrity
- Signing email with digital signatures:
    - 1. Application hashes the message
    - 2. Application retrieves sender's private key and encrypts hash using this private key
    - 3. Application sends both encrypted hash digital signature and unencrypted message to recipient
- Verifies Digital Signature by:
    - 4. Recipients' system retrieves sender's public key, which is sender's public certificate. In some situations, sender may have sent recipient a copy of the certificate with the public key. In domain environments, retrievers system can automatically retrieve sender's certificate from a network location.
    - 5. Email application on Recipient's system decrypts the encrypted hash with Sender's public key
    - 6. Application calculates hash on received message
    - Application compares decrypted hash with calculated hash
    - If hash of calculated received message is the same as encrypted hash it validates authentication, non-repudiation and integrity
- Encrypting email with Only Asymmetric Encryption
    - 1. Sender retrieves copy of recipient's certificate that contains public key
    - 2. Sender encrypts email with recipients' public key
    - 3. Sender sends encrypted email to recipient
    - 4. recipient decrypts email with own private key

- **Remember this:**
  - **Recipients public key encrypts when encrypting an email message and recipient uses recipient's private key to decrypt email message**
- Encrypting email with asymmetric and symmetric encryption
  - 1. Sender identifies a symmetric key to encrypt mail
  - 2. Sender encrypts email contents with symmetric key of lets say 128.
  - 3. Sender retrieves copy of recipient's certificate that contains recipient's public key
  - 4. Uses recipient's public key to encrypt symmetric key
  - 5. Sender sends encrypted email and encrypted symmetric key to recipient
  - 6. Recipient decrypts symmetric key with recipient's own private key
  - 7. Then decrypts email with decrypted symmetric key
- S/MIME – Secure/Multipurpose Internet Mail extension
  - Most popular standards used to digitally sign and encrypt mail and is used in most email applications
  - Uses RSA for asymmetric encryption and AES for symmetric
  - Can encrypt email at rest and in transit
  - Requires PKI to distribute and mange certificates
- PGP/GPG
  - Pretty good privacy
  - Can encrypt, decrypt and digitally sign email
  - GNU privacy card (GPG) is a free software that is based on OpenPGP standard.
  - Uses RSA algorithm and public/private keys for encryption
  - Uses asymmetric and symmetric encryption
- HTTPS Transport encryption
- SSL vs TLS
  - Are both encryption protocols to encrypt data-in-transit
  - Both provide certificate-based authentication and encrypt data with symmetric and asymmetric encryption during a session.
    - Asymmetric encryption for key exchange
    - Symmetric encryption to encrypt data on web page

  - TLS is a replacement for SSL
  - Both require certificates so a CA is required to support TLS and SSL which can be internal or third parties.
- **Remember this:**
  - **TLS is the replacement for SSL. Both TLS and SSL require certificates issued by CA. TLS encrypts HTTPS traffic but also other traffic as well**
- HTTPS
  - Uses asymmetric encryption to transmit a symmetric key using a secure key exchange method. Then uses Symmetric key with Symmetric encryption to encrypt all data in the HTTPS session
    - 1. TLS uses asymmetric encryption to securely share symmetric key
    - 2. TLS uses symmetric encryption to encrypt session data
  - 1. Client Requests secure session
  - 2. Server responds with certificate
  - 3. Client creates symmetric key and encrypts with public key
  - 4. Encrypted symmetric key sent to server
  - 5. Server decrypts symmetric key with private key
  - 6. Session is encrypted with session key using symmetric encryption
- Cipher suites – Combination of cryptographic algorithms providing several layers of security for TLS and SSL
  - Encryption – TLS uses asymmetric cryptography to privately exchange symmetric key and encrypts data with symmetric algorithm. TLS uses symmetric encryption including 3DES and AES
  - Authentication – Require certificates for authentication
  - Integrity – TLS uses MAC (message Auth code) for integrity such as HMAC-MD5 or HMAC-SHA256.
  - E.G of Ciper suites:
    - 0x00C031 – TLS_ECDH_RSA_WITH_AES_GCM_SHA256
    - 0x00003c – TLS_RSA_WITH_AES_128_CBC_SHA256
  - Protocol
  - Key-exchange method:

        - First uses ECDH then second using RSA
    - Authentication
        - Both use RSA listed once.
    - Encryption
        - 128-bit AES w/ GCM and CBD
    - Integrity
        - SHA-256 hashing
- Crypto Module
    - Set of hardware, software, firmware that implements cryptographic functions
    - E.G: Algorithms for hashing/encryption, Digital signatures, authentication techniques etc.
- Crypto Service Provider
    - Software library of cryptographic standards and algorithms. Usually distributed with crypto modules.
- Downgrade attack
    - Attack that forces system to downgrade its security then exploits lesser security control.
- **Remember this:**
    - **Administrators should disable weak cipher suites and weak protocols on servers. When a server has both strong and weak cipher suites, attackers can launch downgrade attacks bypassing strong cipher suite and exploiting weak cipher suite.**
- PKI – Public Key Infrastructure
    - Group of technologies used to request, create, manage, store, distribute and revoke digital certificates.
    - Allows 2 entities to communicate securely without knowing each other previously.
    - Key element in PKI is certificate authority →
- CA - Certificate Authority
    - CA issues manages, validates and revokes certificates.
    - Public CA – make money by selling certificates. Must be trusted.
    - CA's are trusted by placing copy of root certificate into a trusted root CA store

- Root Certificate – first certificate created by CA that identifies it. If Root certificate is placed in store it can be trusted.
    - Issues certificates to intermediate CA's.
    - Intermediate CA's issue certificates to child CA's.
    - Child CA's issue certificates to devices or end users
- Certificate Chaining – Combines all certificates from root CA down to certificate issued to end user.
- CSR – Certificate Signing Request
    - CA require CSR's to be formatted using public-key cryptography (PKCS)
    - CSR includes public key but not private.
    - CA publishes a certificate template showing exactly how to format CSR
    - After receiving CSR, CA validates my identity and creates certificate with public key.
    - Register certificate with website with private key.
- OID – Object Identifiers
    - CA's require OIDS within CSR for certain items
    - OID – string of numbers separated by dots and can be used to name every object type in certificates
    - E.G: 1.3.6.1.4.1.11129.2.5.1
        - 1 indicates ISO (International Organization for Standardization) OID.
        - 1.3 indicates identified organization
        - 1.3.6.1.1.4.1 indicates using an IANA enterprise number.
        - 1.3.6.1.4.1.1129 – indicates organization is google
        - 2.5.1 – Google's enterprise defined within
- **Remember this:**
    - **You typically request certificates using a certificate signing request (CSR). First step is to create RSA-based private key, which is used to create public key. Then include public key in CSR and CA will embed public key in certificate. The private key is not sent to CA.**

- Revoking Certificates

    - Certificates expire based on 'Valid From' and 'Valid To' dates.
    - Can also be revoked due to:
        - Key compromise
        - CA compromise
        - Change of affiliation
        - Superseded
        - Cease of operation
        - Certificate hold
    - CA's use CRL's – Certificate Revocation Lists to revoke a certificate via serial numbers.
- Certificate Issues:
    - Expired
    - Certificate not trusted
    - Improper certificate and key management
        - Private keys should remain private and encrypted.
- Validate certificate via:
    - 1. Client initiates session requiring certificate such as a HTTPS session
    - 2. Server responds with copy of certificate including public key.
    - 3. Client queries CA for copy of CRL.
    - 4. CA responds with copy of CRL.
- OCSP – Online certificate status protocol
    - A protocol that allows you to determine the revocation status of a digital certificate using its serial number
    - Responds with 'good', 'revoked' or 'unknown' to indicate certificate forgery.
- OCSP Stapling
    - Allows the certificate holder to get the OCSP record from the server at regular intervals and include it as part of the SSL or TLS handshake
    - Eliminates need for clients to query CA where before sending, CA signs OCSP response with a digital signature. Certificate presenter appends/stables timestamped OCSP response to certificate during TLS handshake process.

- **Remember this:**
  - **CA's revoke certificates for several reasons such as when private key is compromised or CA is compromised. Certificate Revocation list (CRL) includes list of revoked certificates and is publicly available. An alternative to using a CRL is the OCSP or Online Certificate Status Protocol which returns answers such as good, revoked or unknown. OCSP stapling appends digitally signed OCSP response to a certificate.**
- Public Key Pinning
  - Allows a HTTPS website to resist impersonation attacks by presenting a set of trusted public keys to the user's web browser as part of the HTTP header
- **Remember this:**
  - **Certificate stapling is an alternative to OCSP. Certificate presenter (such as a web server) appends certificate with timestamped digitally signed OCSP response from CA. This reduces OCSP traffic to and from CA. Public key pinning helps prevent attackers from impersonating a web site with a fraudulent certificate. Web server sends a list of public key hashes that clients can use to validate certificates sent to clients in subsequent sessions.**
- Key Escrow
  - Occurs when a secure copy of a user's private key is held in case the user accidently loses their key in a safe environment
  - Organizations provide a copy of key to a third party sometimes
- Key Recovery Agent
  - A specialized type of software that allows the restoration of a lost or corrupted key to be performed
  - A designated individual who can recover or restore cryptographic keys
- Certificate Types
  - Machine/Computer
    - Certificates issued to a device or a computer are commonly called machine certificates or computer certificates

- User
  - Certificates can also be issued to Users
- Email
  - Encryption of emails and digital signatures
- Code signing
  - Signing of code to validate authenticate of application and to make sure code isn't modified
- Self-Signed
  - Certificate not issued by CA. Private CA's in enterprises are often self-signed certificates.
  - Are not trusted by default.
  - Eliminate cost of purchasing from public CA
- Wildcard
  - Start with * and can be used for multiple domains but each domain name must have same root domain
    - E.G: *.google.com, accounts.google.com, support.google.com
  - Allow all of the subdomains to use the same public key certificate and have it displayed as valid
  - Wildcard certificates are easier to manage
- SAN – Subject Alternative Name
  - Allows a certificate owner to specify additional domains and IP addresses to be supported
  - E.G: *.google.com, *.android.com, *cloud.google.com
- Domain Validation
  - Domain validation certificate indicates certificate requestor has some control over a DNS domain.
  - CA takes extra steps to contact requestor such as email or telephone
  - Is used to provide additional evidence to clients that certificate/organization are trustworthy
- Extended Validation
  - Uses additional steps beyond domain validation.
  - E.G: Include name of company before actual URL.
    - E.G: Paypal, INC[US] | before URL.

- Certificate Formats
    - X.509
        - Standard used PKI for digital certificates and contains the owner/user's information and the certificate authority's information
        - Usually in X.509v3 format
        - Uses BER,CER,DER for encoding
- Basic Encoding Rules(BER)
    - The original ruleset governing the encoding of data structures for certificates where several different encoding types can be utilized
- Canonical Encoding Rules(CER)
    - A restricted version of the BER that only allows the use of only one encoding type
    - Binary format
- Distinguished Encoding Rules(DER)
    - Restricted version of the BER which allows one encoding type and has more restrictive rules for length, character strings, and how elements of a digital certificate are stored in X.509
    - ASCII format
    - E.G:

-----BEGIN CERTIFICATE-----
MIIDdTkjlaEHAJLKWJsdnoiwDLSKDNaslkaskdnXQSxF
…. Additional ASCII Characters here…
HMUfpIBvFLSJNDK93askjnDCAF/EjJKSMp4A==
-----END CERTIFICATE-----

- Certificate Extensions and Formats

Type	Common Extensions	Format	Common Purpose	Can contain
CER	.cer	Binary	Used for Binary certificates	Varies
DER	.der	ASCII	Used for ASCII certificates	Varies
PEM	.pem, .cer, .crt, .key	Binary(DER) Or ASCII(CER)	Can be used for almost any certificate purpose	Server Certificates, certificate chains, keys, CRL
P7B	.p7b, .p7c	ASCII (CER)	Used to share public key	Certificates, certificate chains, CRL, but never private key
P12 PFX	.p12, .pfx	Binary(DER)	Commonly used to store private keys with a certificate	Certificates, certificate chains, and private keys

- PEM – Privacy Enhanced Mail
  - Can be formatted as CER(Binary Files) or DER (ASCII files)
  - PEM-based certificates are used for email only
- P7B
  - Use PKCS v7 and are DER-based (ASCII)
  - Share public keys with proof of identity of certificate holder.
  - Recipients use public keys to encrypt or decrypt data.
  - Contain certificate chain or a CRL
  - Never includes private key
- P12 – uses PKCS v12 are CER-based (binary)
  - Hold certificates with private key.
  - Used to install private key on server

- PFX – Personal Information Exchange
  - Predecessor to P12 certificate and has same usage.
  - Used on Windows to import and export certificates
- **Remember this:**
  - **CER is a binary format for certificates and DER is an ASCII format. PEM is the most commonly used certificate format and can be used for just about any certificate type. P7B certificates are commonly used to share public keys. P12 and PFX certificates are commonly used to hold private key.**