**Intro to encryption**

* Symmetric Encryption
* Asymmetric Encryption
* Hashing and Hash Digests
* MAC (Message Authentication Code)
* Digital Signatures
* Certificates
* Certificate Authorities and Certificate Trust Hierarchies
* Cipher Suites

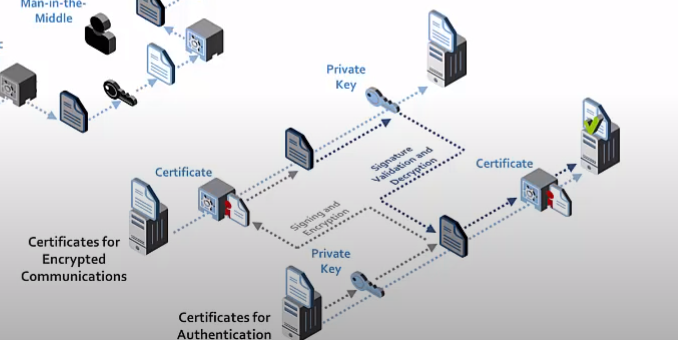
The Ceasar cypher(Old example)

* Shift/Offshoot the alphabet by 3 so that an ‘a’ becomes a ‘d’.
* Decipher the message by shifting it back by 3 so a ‘d’ becomes an ‘a’.
* The algorithm is the shift itself, and the key is the number 3.
* In our time the algorithms are much more sophisticated
  + Mathematical transforms
  + The keys are chosen from much larger domains
    - A much larger number of possible values to randomly pick from.
  + Truly random is hard to achive unpredictabley.

Symetrical and asymmetrical enchryptions

* Symetric encryption for storrage
  + If data is compromised, we don’t want anyone to be able ot read it.
  + We randomly select a key/generate a key, to encrypt the data, so no one without the key can read it
  + We need to make sure we don’t loose they.
* cumunications
  + We want to send a secury message to someone on the right
  + We want to send the other party a key so they can decrypt the message
  + We need to keep the key secret while we a distributing it.
  + We need to rotate they key occasionally to keep it secret.

Asymectric cryptography(Public pivate keys)

* Take data and encrpy it with one key and then decrypt the data with another key
* Directionaly so they we can encrpt with oene key and decrpy with the other.
* These are public and private keys.
* We can then send the public key, that can encrypt something and send it to the one with the private key.
* Public key cryptography.
* The private keys need to be kept secret.
* To avoid man in the middle attacks, where a hacker supplys their own public key in the place of the other partys and then redirects it afterwards, we are now using certificates.
  + Certificate: A name bound to the public key.
  + We can now know to send the essage by certificate.
* Certificates for authentication
  + We can validate the sender, by encrypting the message with our private key, and then on the recipiants side authenticates it with its public key, because the public key is inside a certificate with the sender name on it.
  + Want to send a confidential message tat is also authenticated.
    - Start by encroying it with their private key
    - Use the recipiants certificate and their public key to encrypt the messgage.
    - Now we have somting that can only be red bu the recibiants private key.
    - If the recipient decrypt it, they will end up with the message that was originally encrypted with the private key.
    - So notw they can authenticate thheis message with the certificate from the sender.
    - They know it was send by the proppor sender sence they were able to authenticate it with the certificate.  
      
* Challenges with distribuateing keys.
* Certificates can help us identify the correct key when distributeing.

Certificates

* Digital signature
  + Hash variable, the value that represents the certificate.
  + The hash value is then put through a mathematical function using the private key to generate a digital signature.
  + This digital signature is then added to the certificate.
  + The digital signature can later be check that the certificate has ot been altered or damaged.
  + To check the signature, it is then put through a mathematical function using the public key. Out of the comes the original hash value. If the hash value is not obtained the person knows that the signature is either curropt or has been tampered with.
    - Hash goes one way, so you cant recreate the original certificate.
* Digital signatures provides a way to identify the sender
  + A certificate is essentially a file with data in it.
  + If the certificate has been tampered with, the signature will no longer match and wil be rejected.
  + This way we can prive identity.
* Certificates wirks with a trust value.
  + Certifitcate authority
    - Its job is to issue certificates
    - Certificates are checked by the issuers, to that they don’t fall into the wrong hands. This way we can be reashured that they are valid.
    - The computer must trust the authority that the certificate came from.
      * You need to trust who issued the certificate, in order to start using it.
* Certificate hierarchy
  + CA: Certificate Authroity
  + Root CA: Will issue certificates the the second level autheorities.
    - They hold the private key to the who certificate infrastructure.
    - They are sometimes placed in removable media. Having the Root Ca offline after use helps keep it secure.
  + The sendond level Computer Certificates: Once the Root CA is not needed anymore, the second level CA will often issue certificates to uder level computers.
    - It is possible to have additionals level of autheority if needed.
  + In oder for the certificate to automatically be trusted, a certificate from the Root CA needs to be installed locally on the client computer.
    - Once the Root CA certificate is installed on the computer, the computer will trust any certificate from any of the associated CA authroities in the hirearchy.
* PIK public key infrastructure.
  + Registration authority: Asures that we are allowed to request certificates for our domain. If this is done, the CA certificate Authrority will issue a certificate to us.
    - We become a subject of the certificate.
  + When someone connects to our server, they will get the servers certificate. They will be the relying party.
  + To know the certificate is valid, they need to verify the public key.
    - The process of validateing certificate is done by the OS software we are running on.
  + A Root Certificate Authority, is used to issue a CA certificate to the Certificate authroities, as well as sending a Rood Certificate to the oberating systems that are in charge of validateing these certificates, when coming form servers.
  + If a certificate is no longer to be trusted, because the private certificate or the private key has been compromised, the Certificate authority puts the certificate on a Certificate Revocation List.
    - Every certificate has a serial number that originates from the CA that handed it out, and the Certificate Radication List is a list of all the serial numbers from that perticualar CA that cant be trusted anymore.
    - Each relying party/user involved will then go check this Certificate Radication List.
    - (Can be used) OCSP Responders works where one can ask for the status of a certain setificate based on its serial number.
    - A Root CA can also request to have a certificate listed on the Certificate Revocation List.
  + 