**CYPTOGRAPHY**

* **Cryptography:** study of the techniques used for encryption and secure communications
* **Cryptographers:** people who study and analyze cryptography. They are always constructing and analyzing protocols for preventing unauthorized users from reading private messages as well as the following areas of information security: data confidentiality, data integrity, authentication, nonrepudiation
* Examples of use of cryptography: **virtual private networks (VPNs), ecommerce, secure email transfer, and credit card chips**.
* **Cryptanalysis:** study of how to crack encryption algorithms or their implementations
* **Cipher**: set of rules, which can also be called an algorithm, about how to perform encryption or decryption.

**Methods that cipher use:**

* Substitution: this type of cipher substitute one character for another
* Polyalphabetic: similar to substitution, but instead of using a single alphabet, it can use multiple alphabets and switch between them by some trigger character in the encoded message.
* Transposition: This method uses many different options, including the rearrangement of letters.

**Keys**:

* instructions for how to reassemble the characters.
* For someone to know how to encrypt/decrypt this correctly, the correct key is needed.

**Block ciphers:**

Encryption algorithms can operate on blocks of data at time, or bits and bytes of data, based on the type of cipher.

* Block cipher is a symmetric key cipher (meaning the same key is used to encrypt and decrypt) that operates on a group of bits called a block
* Block cipher encryption algorithm may take a 64-bit block of plaintext and generate a 64-bit block of ciphertext
* In this type of encryption, the key to encrypt is also used to decrypt
* Examples of symmetric block cipher algorithms include the following: AES (Advanced Encryption Standard), 3DES (Triple Digital Encryption Standard), Blowfish, DES (Digital Encryption Standard), IDEA (International Data Encryption Algorithm)
* Block ciphers might add padding in cases where there is not enough data to encrypt to make a full block size
* This might result in a very small amount of wasted overhead, because the small padding would be processed by the cipher along with the real data.

**Stream cipher**

* It is a symmetric key cipher where the plaintext data to be encrypted is done a bit at a time against the bits of the key stream. The resulting output is a ciphertext stream.
* Because a given algorithm ciphertext stream does not have to fit in a given block size, there may be slightly less overhead than with a block cipher that requires padding to complete a block size

**Cryptographic Algorithms**

**Symmetric encryption algorithms:**

* Examples of symmetric encryption algorithms: AES, 3DES, DES, Blowfish, IDEA, RC2, RC4, RC5, RC6
* The same key is used to encrypt and decrypt
* There are used for most of the data we protect in VPNs today because they are much faster to use and take less CPU than asymmetric algorithms.
* More difficult the key, the more difficult it is for someone who does not have the key to intercept and understand the data.
* Typical key length is 112 bits to 256 bits
* Minimum key length should be at least 128 bits for symmetric encryption algorithms to be considered fairly safe.

**Asymmetric algorithm**:

* It is a public key algorithm
* They use two different keys that mathematics work together as a pair: **public key and the private key**
* very high CPU cost when using key pairs to lock and unlock data; required more CPU processing power than symmetric algorithms
* More secure
* Typical length: between 2048 and 4096
* A key length that is shorter than 2048 is considered unreliable and not as secure as a longer
* we use asymmetric algorithms for things such as authenticating a VPN peer or generating keying material that we can use for our symmetric algorithms
* The public key is available to anyone who wants to use it and the private key is known only to the device that owns the public-private key pair

**Examples of asymmetric algorithms:**

* **RSA**
  + The primary use of this asymmetric algorithm is for authentication
  + key length is 512 bits to 2048 bits
  + Minimum key length for good security is 1024.
* **DH**
  + Diffie Hellman is used to allow two devices to negotiate and establish shared keying material over an untrusted network
  + The keys generated by the exchange are symmetric keys that can then be used with symmetric algorithms such as 3DES, AES
* **ELGamal:** This asymmetric encryption system is based on the DH exchange
* **DSA Digital Signature Algorithm** was developped by the US
* **ECC: Elliptic Curve Cryptography**

**Hashes**

**Hashing:** method used to verify data integrity

**Cryptographic hash** **function**: is a process that takes a block of data and creates a small fixed-sized hash value. If two different computers take the same data and run the same hash function, they should get the same fixed-sized hash value

Three most popular types of hashes are:

* **Message digest 5 MD5**: this hash creates a 128-bit digest. MD5 hashing has several vulnerabilities
* **Secure Hash Algorithm 1 SHA**: this hash creates a 160-bit digest. SHA-1 has several vulnerabilities and attacks.
* **Secure Hash Algorithm 2 SHA2**: include a digest between 224-bits and 512-bits
* It is recommended to use SHA-2 with 512bits

**Hashed Message Authentication code (HMAC)**

* HMAC is a construction that uses a secret key and hash function to provide a message authentication code for a message
* Instead of using a hash that anyone can calculate, it includes in its calculation a secret key of some type
* Only the other party who also knows the secret key and can calculate the resulting hash can correctly verify the hash
* We have a HMAC-MD5, which uses MD5 as its hash function
* NIST recommended HMAC function is HMAC-SHA-1

**Digital Signatures**

* Digital signature provides three core benefits: authentication, data integrity, and nonrepudiation
* We can define Digital signature as a: encrypted hash that use public-private keys and digital certificates
* Certificate Authority CA: trusted entity that hands out digital certificates

**Key management**: deals with generating keys, verifying keys, exchanging keys, storing keys and at the end of their lifetime, destroying keys.

**Keyspace**: refers to all the possible values for a key. Bigger the key, the more secure the algorithm will be. But the more the key is longer, the more the CPU is used.

**Next-Generation Encryption Protocols**

* Elliptic Curve Cryptogaphy (ECC) replaces RSA signatures with the ECDSA algorithm and replaces the DH key exchange with ECDH. ECDSA is an elliptic curve variant of the DSA algorithm, which has been a standard since 1994. The new key exchange uses DH with P-256 and P-384 curves.
* AES in the Galois/Counter Mode (GCM) of operation.
* ECC digital signature algorithm.
* SHA-256, SHA-384, and SHA-512.

**IPsec**

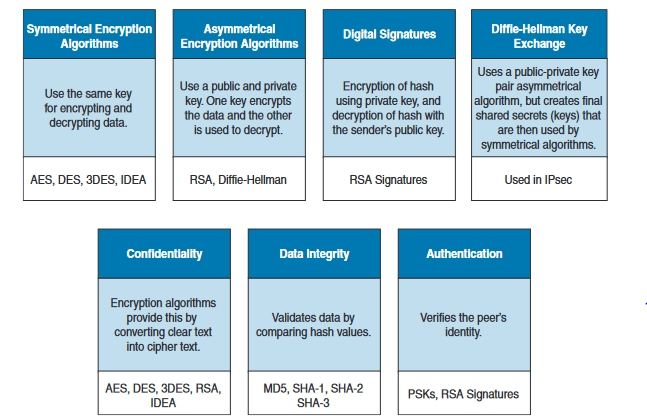
* Suite of protocols used to protect IP packets at layer 3 that is use today for both remote-access VPNs and site-to-site VPNs
* IPsec provides the core benefits if confidentiality through encryption, data integrity through hashing and HMAC, authentication using digital signatures or using a pre-shared key (PSK)
* IPsec components:
  + ESP and AH: two primary methods for implementing IPsec
  + ESP: Encapsulating Security Payload which can perform all the features of IPsec
  + **AH** stands for Authentication Header: it can do many parts of IPsec objectives except encryption of the data. It is not frequently used
  + Encryption algorithms for confidentiality: DES, 3DES, and AES
  + Hashing algorithms for integrity: MD5 and SHA
  + Authentication algorithms: pre-shared keys (PSKs), and RSA digital signatures
  + Key management:
    - Example: include Diffie-Hellman DH, which can be used to dynamically generate symmetric keys to be used by symmetric algorithms
    - Example: PKI which supports the function of digital certificates issued by trusted CAs
    - Example: Internet Key Exchange IKE, which does a lot of the negotiating and management needed for IPsec to operate

**SSL/TLS**

IPsec can be used to encrypt the data, perform integrity checking and authentication of the server we are connect to, but not everyone has an IPsec client or software running on their computer.

* We can use SSL (Secure Sockets Layer) to perform encryption and authentication because all computer can use it.
* To use SSL, the users connect to an SSL server by using HTTPS
* SSL may also be called Transport Layer Security, depending on whom you talk.
* TLS is the preferred method of encrypt communication
* TLS 1.3 is the latest version of TLS
* TLS 1.3 benefits: separating key agreement and authentication algorithms from the cipher suites, removes support weak and less-used names elliptic curves, removes the uses of MD5 and SHA-224 cryptographic hash function

**Fundamental Encryption components**



**Fundamentals of PKI (Public Key Infrastructure)**

PKI is a set of identities, roles, policies, and actions for the creation, use, management, distribution, and revocation of public and private keys.

**Public and Private Key pair**

* **Key pair**: is a set of two keys that work in combination with each other as a team

Example: key pair: one public key and one private key

* Public key may be shared with everyone and the private key is not shared
* If you use the public key to encrypt data, the corresponding private key is used to decrypt the data. The inverse is also true

**Certificate Authorities**: computer or entity that creates and issues digital certificates.

**Digital certificate** is information about the identity of device, such as its IP address, fQDN and the public key of that device. (Electronic document)

We have **root certificates** (which identify the CA) and **identity certificates** (which identify devices such as servers and others devices that want to participate in PKI).

**Root certificates**

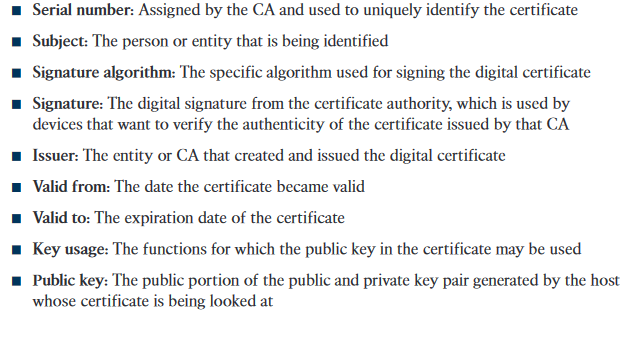
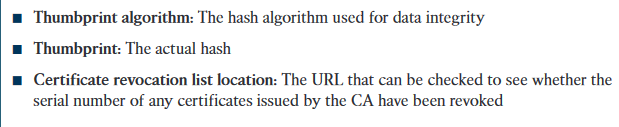
They contain public key of the CA server and the other details about the CA server:

* **Serial number**: issued and tracked by the CA that issued the certificate
* **Issuer:** the CA issued this certificate
* **Validity dates**: time window during which the certificate is considered valid
* **Subject of the certificates:** this includes organization(O), country (C) etc found in an X.500 structured directory. It is the CA itself
* **Public key:** content and length of the key are often both shown
* **Thumbprint algorithm and thumbprint:** Hash algorithm andhash for the certificate

**Identity Certificate:** similar to a root certificate, but it describes the client and contains the public key of the client (ex: web server that want to support SSL) or router that want to use digital signature for authentication of VPN tunnel.

Any device that wants to verify a digital signature must have the public key of the sender.

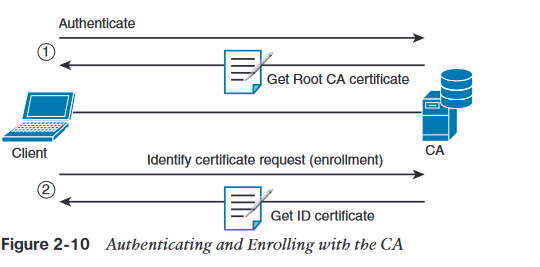
**Digital certificates information:**

**x.500 and x.500v3**

X.500: is a series of standards focused on directory services and how those directories are organized.

Authenticating and enrolling with the CA



**Public Key Cryptography Standards PKCS**: control the format and use of certificates

* PKCS #10: format of a certificate request sent to a CA that want to receive its identity certificate; include the public key for the entity desiring a certificate
* PKCS #7: format can be used by a CA as a response to a PKCS #10 request
* PKCS #1: RSA cryptography standard
* PKCS #12: A format for storing both public and private keys using a symmetric password-based key to “unlock” the data whenever the key needs to be used or accessed.
* PKCS #3 Diffie-hellman key exchange

**Simple certificate Enrollment Protocol (SCEP):**

* They can be used to automate most of the process for requesting and installing an identity certificate; it was developed by cisco and few other vendors

**Revoking Digital Certificates**

If we decommission a device that has been assigned an identity certificate, or if the device assigned a digital certificate has been compromised you could request from the CA that the previously issued certificate be revoked. To know that the certificate we just received has been revoked we must check and see. A digital certificate contains information on where an updated lit of revoked certificates can be obtained.

Three basic ways to check whether certificates have been revoked are as follows:

**Certificate revocation list (CRL):**

* List of certificates, based on their serial numbers, that had initially been issued by a CA but have since been revoked and as a result should not be trusted
* Could be very large and the client would have to process the entire list to verify a particular certificate is not on the list
* Primary protocol used for this purpose
* CRL can be accessed by several protocols, including LDAP, HTTP, SCEP

**OSCP: Online Certificate Status Protocol:**

* this is an alternative to CRLs.
* The client sends a request to find the status of a certificate and gets a response without having to know the complete list of revoked certificates

**AAA Authentication, authorization and accounting**: also provide support for validating digital certificates, including a check to see whether a certificate has been revoked. It is not often used in PKI because this a proprietary solution

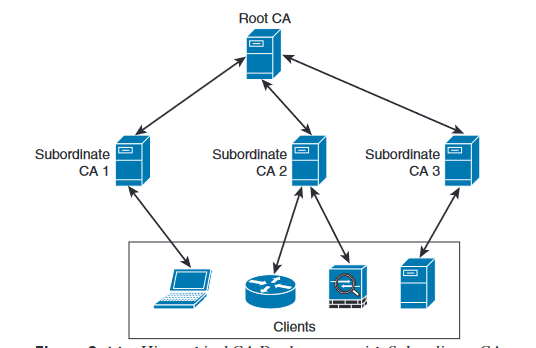
**Digital Certificates in Pratice:**

* Digital certificates can be used when you do online banking from your pc to the bank’s website
* can **be used if you use SSL technology for your remote-access VPNs**
* we can also **use digital certificates with the protocol family of IPsec** (for the authentication portion)
* can be **used with protocols such as 802.1X; ex : wireless network, controlling access and requiring authentication**, using digital certificates for the PCs/Users, before allowing them in on the network

**PKI Topologies**

**Single CA server** might not provide availability and fault tolerance in a large network.

* To offload some of the workload form single server, you could publish CRLs on others servers.
* Another option for supporting fault tolerance and increased capacity is to use intermediate or subordinate CAs to assist the root CA. The root CA is the king of the hill. The root CA delegates the authority (to the subordinate CAs) to create and assign identity certificates to clients. This is called a **hierarchical PKI topology**. The root CA signs the digital certificates of its subordinate or intermediate CAs, and the subordinate CAs are the ones to issue certificates to clients. For a client to verify the “chain” of authority, the client needs both the subordinate CA’s certificate and the root certificate.



* **Cross-Certifying CAs**Another approach to hierarchical PKIs is called cross-certification. With cross-certification, you would have a CA with a horizontal trust relationship over to a second CA so that clients of either CA can trust the signatures of the other CA.