Chapter 10

Cryptography and RKI

horizontal line

# Hashing

Hashes aren’t 2-way; you can’t use the hash to reproduce the original data

**MD5**

Message digest 5, a hashing algorithm that produces a 128-bit hash. It is considered cracked now, although it is still used to check file integrity.

**SHA**

Secure hashing algorithm, with 4 variations:

* SHA-0: not used
* SHA-1: Similar to MD5, but creates 160-bit hashes instead of 128
* SHA-2: Has SHA-256/224/512, and is the improved version to overcome potential weakness.
* SHA-3: Alternate to SHA-2

**HMAC**

Hash-based message authentication code, similar to MD5 except is uses a secret key which only the sender and receiver know. This provides integrity, and authentication.

IPSec and TLS both use a version of HMAC such as HMAC MD5 and HMAC SHA1

**RIPEMD**

**RACE integrity primitives evaluation message digest** is another hash function used for integrity

**Hashing passwords**

**Key stretching**

Increases strength of stored passwords and can stop brute force and rainbow table attacks. It uses salting to increase complexity. Two common techniques are:

* **Bcrypt**: Based on blowfish block cipher. Salts password until it is a 60 char string. Can also add pepper, which is another set of random bits stored elsewhere.
* **PBKDF2**: Salts of at least 64 bits, and uses functions such as HMAC to protect passwords. Used by WPA2 and cisco OSs to increase password security. However, PBKDF2 is more susceptible to brute force attacks

**Hashing messages**

Provides integrity for messages, and the receiving system can calculate the hash and compare it to the hash it received. If it is different, the message has changed.

**Using HMAC**

The HMAC secret key can be applied to message hashes, so that an attacker cannot replace the hash, as he does not know the HMAC secret key. HMAC-MD5 and HMAC-SHA1 provides integrity and authentication

**Confidentiality with encryption**

Data at rest: Data stored on media, which can be encrypted

Data in transit: Data sent over network

Data in use: Data being used by a computer

**Encryption terms**

IV: Initialisation vector, a fixed number that provides a starting value for crypto algorithms, helping create random encryption keys. The IV should be large enough to not reuse the same IV and recreate the same encryption keys.

Nonce: A number used once, e.g. IV

XOR: If 2 inputs are same, it outputs false. If different, outputs true.

Confusion: When ciphertext is different to plaintext

Diffusion: Ensures that changing one char in plaintext will completely change the ciphertext

Secret algorithm: One that is kept private. Discouraged, because review by experts etc can discover flaws

Weak/deprecated algorithms: Easily cracked, and these algorithms should be deprecated.

High resiliency: Ensure that encryption keys are not compromised via leakage

**Block vs stream ciphers**

Both symmetric

Block cipher: Encrypts data as a block of 64 or 128 bits by separating them. More efficient when filesize is known

Stream cipher: Encrypts data as a stream of bytes rather than dividing it into blocks. More efficient when filesize is unknown, and encrypting data in a continuous stream

**Cipher modes**

**Electronic codebook**: ECB is the simplest cipher mode. It uses the same key for all data blocks. Dangerous, as it makes the data easier to crack.

**Cipher block chaining**: CBC is used by symmetric block ciphers, and uses an IV for randomising the first block. It then combines each subsequent block with the previous one using XOR. this can make it slow, as it depends on encryption of previous blocks

**Counter**: CTR/CTM/CM, converts block cipher into stream cipher. It combines IV with a counter and uses the result to encrypt each plaintext block. Combining IV with a counter results in a different key for each block, so it is widely used.

**Galois/Counter**: GCM, Combines counter method with Galois mode of authentication. Provides authenticity, integrity and confidentiality, and very efficient

**Symmetric encryption**

Uses same key to decrypt and encrypt, also called secret key. However, these keys are changed often.

Substitution cipher: Replaces plaintext with ciphertext using a fixed system

Obfuscation: Security through obscurity.

AES: Advanced encryption standard, a strong symmetric block cipher that encrypts data in 128 bits. It is fast, efficient, and strong due to being able to encrypt and decrypt in one pass, and being less resource intensive than other algorithms

DES: Data encryption standard. Encrypts in 64-bit blocks. Can be compromised via brute force

3DES: Encrypts data in 3 separate passes, in 64-bit blocks. Is a suitable alternate if hardware does not support AES

RC4: Symmetric stream cipher

Blowfish: Strong symmetric block cipher, encrypts in 64-bits. Faster than AES because it encrypts in smaller blocks.

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Type | Method | Key size |
| AES | Symmetric encryption | 128bit block cipher | 128/192/256-bit |
| 3DES | Symmetric encryption | 64bit block cipher | 56/112/168 bit |
| Blowfish | Symmetric encryption | 64bit block cipher | 32 to 448 bit |
| Twofish | Symmetric encryption | 128bit block cipher | 128/192/256-bit |
| RC4\* | Symmetric encryption | Stream cipher | 40 to 2048 bit |
| DES\* | Symmetric encryption | 64bit block cipher | 56 bit |

**Asymmetric encryption**

Uses 2 keys in a matched pair of public and private to encrypt and decrypt, and vice versa. Public keys can be shared by embedding them in a public certificate.

It is usually only used for key exchange, because of how resource intensive it is

**Certificates**

Includes the public key and author info

**RSA**

Acronym of creator’s last names. It is an asymmetric encryption method using a public and private key pair.

Relies on the fact that it is difficult to factor the product of 2 large prime numbers, so RSA is secure if the key sizes are large enough

Key size should be 2048

**Static vs ephemeral keys**

Static key: Semipermanent and stays the same over a long period of time. Used by RSA; the key is valid for the lifetime of the certificate, e.g. 1 yr

Ephemeral key: Short lifetime, and recreated for each session. They comply with perfect forward secrecy, which is when a cryptographic system generates random public keys and doesn’t use a deterministic algorithm to do so. Therefore, the compromise of a long-term key doesn’t compromise any past keys.

**Elliptic curve cryptography**

ECC does not take as many resources, so is used on low power devices. It uses equations to plot a graph and then points of the curve create keys. This is easier, and also more difficult to crack. However it is deprecated

**Diffie Hellman**

DH is a key exchange algorithm used to share a symmetric key between 2 parties. There are 2 methods that support both ephemeral and static keys.

* DHE: DH ephemeral, generates different keys for each session
* ECDHE: Elliptic curve DH ephemeral,uses ephemeral keys generated by elliptic curve method.

In DH, the higher the group the longer and and more secure the keys are.

**Steganography**

Hides data in plain sight, e.g. by manipulating it, or hiding data in white space. Hashing can be used to detect steganography.

**Using cryptographic protocols**

Email digital signals:

* Sender’s private key encrypts/signs
* Sender’s public key decrypts

Email encryption:

* Recipient’s public key encrypts
* Recipient’s private key decrypts

Website encryption:

* Web site’s public key encrypts
* Web site’s private key decrypts
* Symmetric key encrypts data in the website session

**Protecting email**

**Digital signatures**

DSA (Digital signature algorithm) uses an encrypted hash of a message, which is encrypted with the sender's private key. This provides authentication, non repudiation and integrity.

They are encrypted hashes of the message,

**Encrypting email**

The public key does not actually encrypt the email. It encrypts the symmetric key used to encrypt the email. The recipient then uses the private key to decrypt the symmetric key, and then uses the symmetric key to decrypt the email.

With Asymmetric encryption:

1. Sender receives recipient’s public key and encrypts email with that.
2. Email is sent, and recipient decrypts with their own private key.

Symmetric and asymmetric:

Asym is used to privately share a session key, and sym is used to encrypt data

1. Sender identifies a sym key to encrypt email contents
2. Sender receives copy of recipient’s public key, and uses this to encrypt the sym key
3. Encrypted email and encrypted sym key is sent to recipient
4. Recipient decrypts sym key with their private key, and uses this key to decrypt the email

S/MIME: Secure/multipurpose internet mail extensions, used to sign and encrypt email. Uses RSA for asym (therefore requires PKI) and AES for sym.

PGP/GPG: Pretty good privacy, used to encrypt, decrypt and sign emails. GNU privacy guard, PGP based standard so users don’t have to pay licensing fees to use it. Both use RSA.

**Encrypting HTTPS**

**SSL vs TLS**

Used to encrypt HTTPS, and can also encrypt FTPS. TLS is the replacement for SSL, as SSL has been deprecated. Any reference to SSL on the exam is also TLS.

They both provide certificate-based authentication, and use asym for key exchange, and sym for data encryption.

Encrypting with TLS:

Uses asym encryption to share sym key, and uses sym encryption to share session data.

1. HTTPS session is requested, and server responds with certificate containing public key.
2. Client creates a sym key (session key) and encrypts it with the server’s public key
3. Server’s private key decrypts the sym session key, so both server and client know the session key.
4. All session data encrypted with this sym key using sym encryption

**Cipher suites**

Combination of crypto algorithms that provide several layers of security for TLS and SSL, such as:

* Encryption via asym exchange of sym keys
* Authentication via certificates
* Integrity via HMAC hashing

EXAMPLE:

TLS\_ECDH\_RSA\_WITH\_AES\_128\_GCm\_SHA256

TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA256

Protocol: Both using TLS

Key exchange method: 1st is ECDH and 2nd one is RSA

Authentication: Both using RSA, although it is shortened in the 2nd line of code

Encryption: 128-bit AES, 1st one with GCM and 2nd one with CBC (cipher block chaining)

Integrity: SHA256 for hashing

**Implementation vs algorithm selection**

Crypto-module: Set of hardware/software that implements crypto functions such as hashing, encryption etc

Crypto-service providers: Software library of crypto standards and algorithms. Distributed within crypto modules.

**Downgrade attacks**

Force a system to downgrade security, making the system easier to attack. This can be done by configuring the server to SSL, which has lots of vulnerabilities to exploit. To prevent this, weak ciphers should be disabled.

**PKI components**

Public key infrastructure, a group of tech used to manage digital certificates. A key element is certificate authority

**Certificate authority**

Issues, manages, validates and revokes certificates, e.g. symantec is a public CA. The CAs must be trusted in order for their certificates to be trusted.

**Certificate chaining and trust models**

CAs are trusted by placing a root cert (their first cert) into a trusted root CA store.

Hierarchical/centralised trust model: Public CA creates root CA, which issues certs to intermediate CAs, which issue certs to child CAs, which issue certs to end users. This is certificate chaining

Decentralised trust model: Cert is verified by third party.

**Registration and CSRs**

Users can request certificates from a CA via registration. A certificate signing request (CSR) would give the purpose of the cert, details, and also the public key. The CA then validates the identity and creates a cert with the public key.

OID: object identifiers, which can be used to name all object types in a cert.

**Revoking certs**

This can happen if the private key is compromised, or if the CA is.

A certificate revocation list (CRL) revokes certs.

**Certificate issues**

Expiry: First check is to check whether the cert is still valid

Trust: If a system does not have a copy of the CA’s cert, then it will appear as not trusted, and users will be discouraged from continuing.

Improper certificate and key management: Keys should be kept private, and if they aren’t, then it can compromise the certificate.

Validation can also be done by:

1. Client initiates session requiring cert
2. Server responds with cert and public key
3. Client queries CA for CRL
4. CA responds with copy of CRL
5. Client checks serial no of cert against serial nos in CRL, to see if this cert is still valid.

We can also validate via the OCSP, the online certificate status protocol, which allows the user to query the CA. It provides a real time response.

However, OCSP resulted in a lot of traffic for CAs, so stapling was introduced. This means that during the handshake TLS process, the cert presenter appends a timestamped OCSP response, saving the client having to query the CA.

**Public key pinning**

HTTPS requests are met by an extra header which contains a list of hashes derived form valid public keys. When clients connect, they calculate the hashes and compare them. This verifies the certs.

**Key escrow**

Placing a copy of the private key in a hidden place, useful for recovery.

**Recovery agent**

A designated individual who can recover/restore crypto keys

**Certification formats**

They use x.509 v3 formats, and stored in binary. The base format is either CER or DER (canonical encoding rules, binary or distinguished encoding rules, ascii)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | Common extensions | Format | Common purpose | Can contain |
| CER | .cer | Binary | Used for binary certs | varies |
| DER | .der | ASCII | ASCII certs | varies |
| PEM | .pem .cer .crt .key | Binary (CER)  Or ASCII (DER) | Any cert purpose | Server certs, cert chains, keys, crl |
| P7B | .p7b .p7c | ASCII (DER) | Used to share public key | Certs, cert chains, crl but never private key |
| P12 PFX | .p12 .pfx | Binary (CER) | Used to store private keys with cert | Certs, cert chains, and private key |

CER: Binary format

DER: ASCII format, with header and footer of a cert.

PEM: Privacy enhanced mail format.

P7B: Used to share public keys and proof of ID for cert holder

P12: Holds cert with private key

PFX: Similar to P12, but used on Windows to import and export certs

**ACRONYMS**

MD5: Hashing algorithm producing 128bit hash

SHA: Secure hashing algorithm

HMAC: Hash-based message authentication code, similar to MD5

RIPEMD: RACE integrity primitives evaluation message digest, a hash function

PBKDF2: Key-stretching (salting) technique

**Cipher modes**

ECB: Electronic codebook, simplest cipher, uses same key for all data blocks

CBC: Cipher block chaining, uses an IV for randomising first block

CTR/CTM/CM: Counter, converts block cipher into stream cipher by combining IV with counter to get different key for each block.

GCM: Galois counter, combines counter method with galois method of authentication.

**Symmetric encryption**

AES: Advanced encryption standard, symmetric block cipher, encrypts in 128 bits

DES: Data encryption standard, encrypts in 64 bit blocks

3DES: Encrypts data in 3 passes, in 64-bit blocks

RC4: Symmetric stream cipher

**Asymmetric encryption**

RSA: Uses public and private key. Produces key based on product of 2 large prime numbers

ECC: Elliptic curve cryptography, resource-saving key generating method.

DH: Diffie-hellman, key exchange algorithm

ECDHE: EC DH ephemeral, uses ephemeral keys generated by EC method

**Encrypting email**

DSA: Digital signal algorithm, uses encrypted hash of message, which is encrypted with sender’s private key.

S/MIME: Secure/multipurpose internet mail extensions, used to sign and encrypt email.

PGP: Pretty good privacy, used to encrypt, decrypt and sign emails. Based on a standard so users don't have to pay license fee.

**PKI components**

CSR: Certificate signing request, gives purpose of cert, details, and public key. Used to request certs from a CA.

CRL: Certificate revocation list.

Ocsp: Online cert status protocol, allows user to query CA to check status of a cert. It gives a timestamped response on certs, so CA queries are reduced.

CER: Binary format

DER: ASCII format, with header and footer of a cert.

PEM: Privacy enhanced mail format.

P7B: Used to share public keys and proof of ID for cert holder

P12: Holds cert with private key. Used to install private key on a server

PFX: Similar to P12, but used on Windows to import and export certs