

Takeaways:

- Don't invent/implement your own cryptosystem.
 - Encryption in motion: SSL/TLS, SSH, IPsec.
 - Encryption at rest: PGP/GnuPG.
 - Do not use AES, RSA, etc. algorithms directly.
 - Avoid and delegate using crypto if possible.
 - Exceptions do exist, but hire a crypto expert before you pass Go.
 - Popular programs/libraries often get it wrong. Spot check their crypto before use.
-

Types of Encryption:

	<u>Symmetric (secret key)</u>	<u>Asymmetric (public key)</u>
	Same key encrypts and decrypts	Different keys encrypt and decrypt
Typical key sizes:	128, 192, 256 bits	2048, 3072, 4096 bits (256-512 for elliptic curve)
Speed:	Fast! Fast! Fast!	Sloooooooooooooooooooo
Algorithms:	block: AES, 3-DES, Blowfish, Serpent stream: RC4, Salsa20	RSA, DSA, ElGamal, Elliptic Curve Algorithms
Typical uses:	data encryption, checksums, pseudo-random number generators, components of cryptographic hash functions	meta data encryption, authentication, verification, digital signatures, symmetric key encryption key exchange

Basic Modes of Symmetric Block Encryption:

ECB: Electronic Code Book

$\text{encrypt}(\text{plaintext1}, \text{key}) \rightarrow \text{ciphertext1}$
 $\text{encrypt}(\text{plaintext2}, \text{key}) \rightarrow \text{ciphertext2}$

$\text{decrypt}(\text{ciphertext1}, \text{key}) \rightarrow \text{plaintext1}$
 $\text{decrypt}(\text{ciphertext2}, \text{key}) \rightarrow \text{plaintext2}$

CBC: Cipher Block Chaining

Why? — Each ciphertext block is unique, even if plaintext blocks are the same.

$\text{encrypt}(\text{plaintext1}, \text{key}, \text{init_vector}) \rightarrow \text{ciphertext1}$
 $\text{encrypt}(\text{plaintext2} \oplus \text{ciphertext1}, \text{key}) \rightarrow \text{ciphertext2}$
 $\text{encrypt}(\text{plaintext3} \oplus \text{ciphertext2}, \text{key}) \rightarrow \text{ciphertext3}$

$\text{decrypt}(\text{ciphertext1}, \text{key}) \oplus \text{init_vector} \rightarrow \text{plaintext1}$
 $\text{decrypt}(\text{ciphertext2}, \text{key}) \oplus \text{ciphertext1} \rightarrow \text{plaintext2}$
 $\text{decrypt}(\text{ciphertext3}, \text{key}) \oplus \text{ciphertext2} \rightarrow \text{plaintext3}$

CFB/OFB: Cipher/Output Feedback

Why? — Turns a block cipher into a stream cipher.

$\text{keystream1} = \text{encrypt}(\text{init_vector}, \text{key})$
 $\text{keystream2} = \text{encrypt}(\text{ciphertext1}/\text{keystream1}, \text{key})$
 $\text{keystream3} = \text{encrypt}(\text{ciphertext2}/\text{keystream2}, \text{key})$

$\text{keystream1} \oplus \text{plaintext1} \rightarrow \text{ciphertext1}$
 $\text{keystream2} \oplus \text{plaintext2} \rightarrow \text{ciphertext2}$
 $\text{keystream3} \oplus \text{plaintext3} \rightarrow \text{ciphertext3}$

$\text{keystream1} \oplus \text{ciphertext1} \rightarrow \text{plaintext1}$
 $\text{keystream2} \oplus \text{ciphertext2} \rightarrow \text{plaintext2}$
 $\text{keystream3} \oplus \text{ciphertext3} \rightarrow \text{plaintext3}$

CTR: Counter

Why? — Allows fast random access to any part of the key stream.

$\text{keystream1} = \text{encrypt}(\text{nonce}, \text{key})$
 $\text{keystream2} = \text{encrypt}(\text{nonce} + 1, \text{key})$
 $\text{keystream3} = \text{encrypt}(\text{nonce} + 2, \text{key})$

$\text{keystream1} \oplus \text{plaintext1} \rightarrow \text{ciphertext1}$
 $\text{keystream2} \oplus \text{plaintext2} \rightarrow \text{ciphertext2}$
 $\text{keystream3} \oplus \text{plaintext3} \rightarrow \text{ciphertext3}$

$\text{keystream1} \oplus \text{ciphertext1} \rightarrow \text{plaintext1}$
 $\text{keystream2} \oplus \text{ciphertext2} \rightarrow \text{plaintext2}$
 $\text{keystream3} \oplus \text{ciphertext3} \rightarrow \text{plaintext3}$

Authenticated Encryption Modes of Symmetric Block Ciphers (AEAD):

GCM: Galois Counter Mode fastest free AEAD mode, becoming part of standards, complicated implementation

OFB: Offset Feedback Mode fastest AEAD mode, simple implementation, patented but royalty-free for GPL and non-commercial/non-government use

EAX: not an acronym two-pass (slow) AEAD mode, simple implementation

CCM: Counter with CBC-MAC two-pass (slow) AEAD mode, simple implementation, cannot encrypt without entire plaintext

Types of Hash Functions:

Fast Collision Resistant Hashes: MD and SHA family, use SHA-256, SHA-384, SHA-512, or SHA-3

Password Hashes: bcrypt, PBKDF2, scrypt

MAC Hashes (Message Authentication Code): CBC-MAC, HMAC