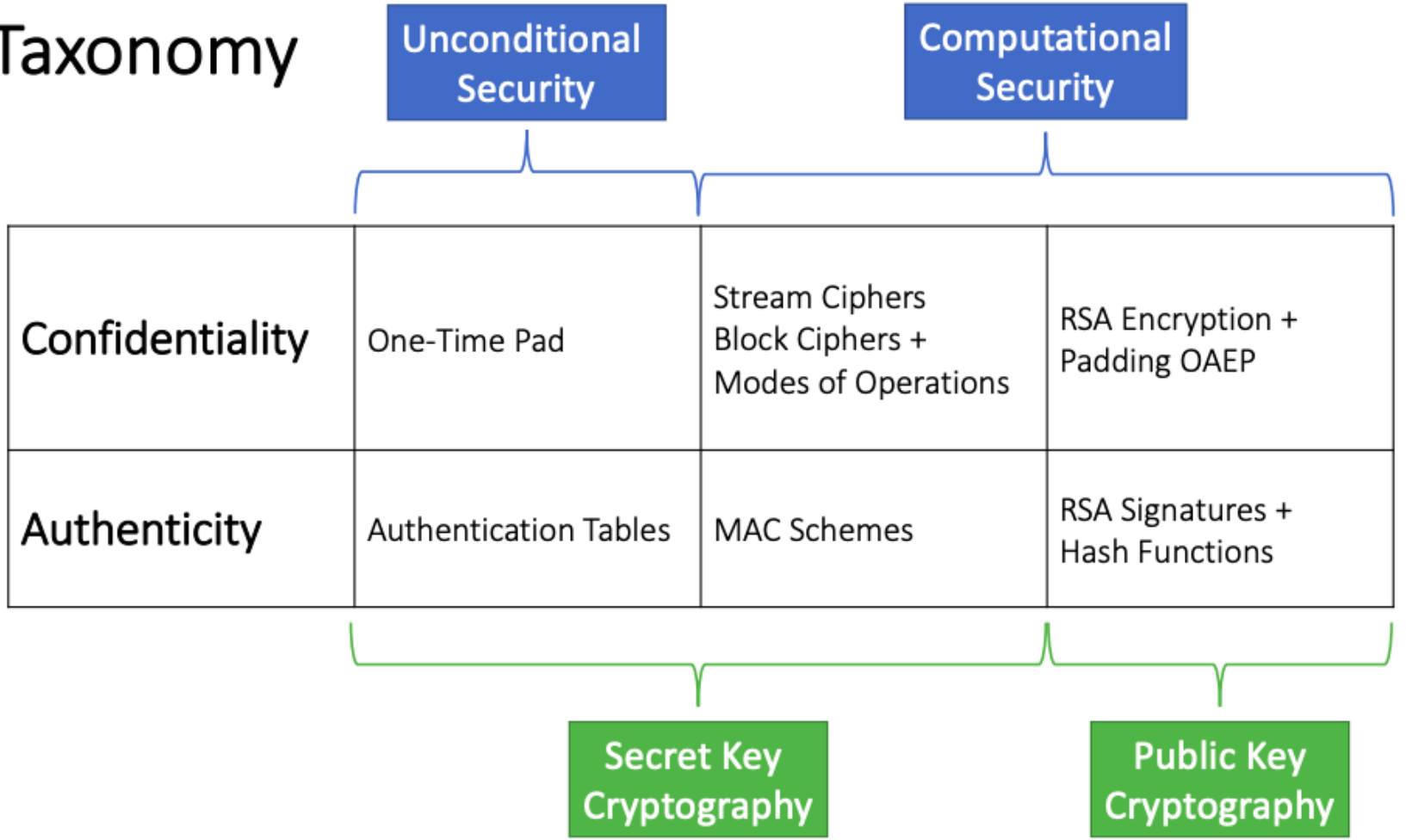
# Disposition 1: Confidentiality

Mathias Ravn Tversted

January 14, 2020

# Table of contents

# Taxonomy


Unconditional Security
Computational Security
Confidentiality
One-Time Pad
Stream Ciphers
Block Ciphers +
Modes of Operations
RSA Encryption +
Padding OAEP
Authenticity
Authentication Tables
MAC Schemes
RSA Signatures +
Hash Functions
Secret Key Cryptography
Public Key Cryptography

# Unconditional security

# Perfect Secrecy: One-time Pad

One-time pad relies on the following identity

$$c_i \bigoplus k_i = (m_i \bigoplus k_i) \bigoplus k_i = m_i \bigoplus (k_i \bigoplus k_i) = m_i$$

And has perfect secrecy[1]. The cipher text does not rely on the distribution of the message, but you need a key size that is at least as long as the cipher text, and is therefore not really useful in practice.

[1] Theorem 5.1

# Computational security, Secret-key systems

# Definition of security

We consider a case of an oracle with the secret-key, which either spits out $c = E_k(m, n)$ or $E_k(r, n)$. So either a random message or the message encrypted with a nonce.

Downside is that all parties must have the secret key.

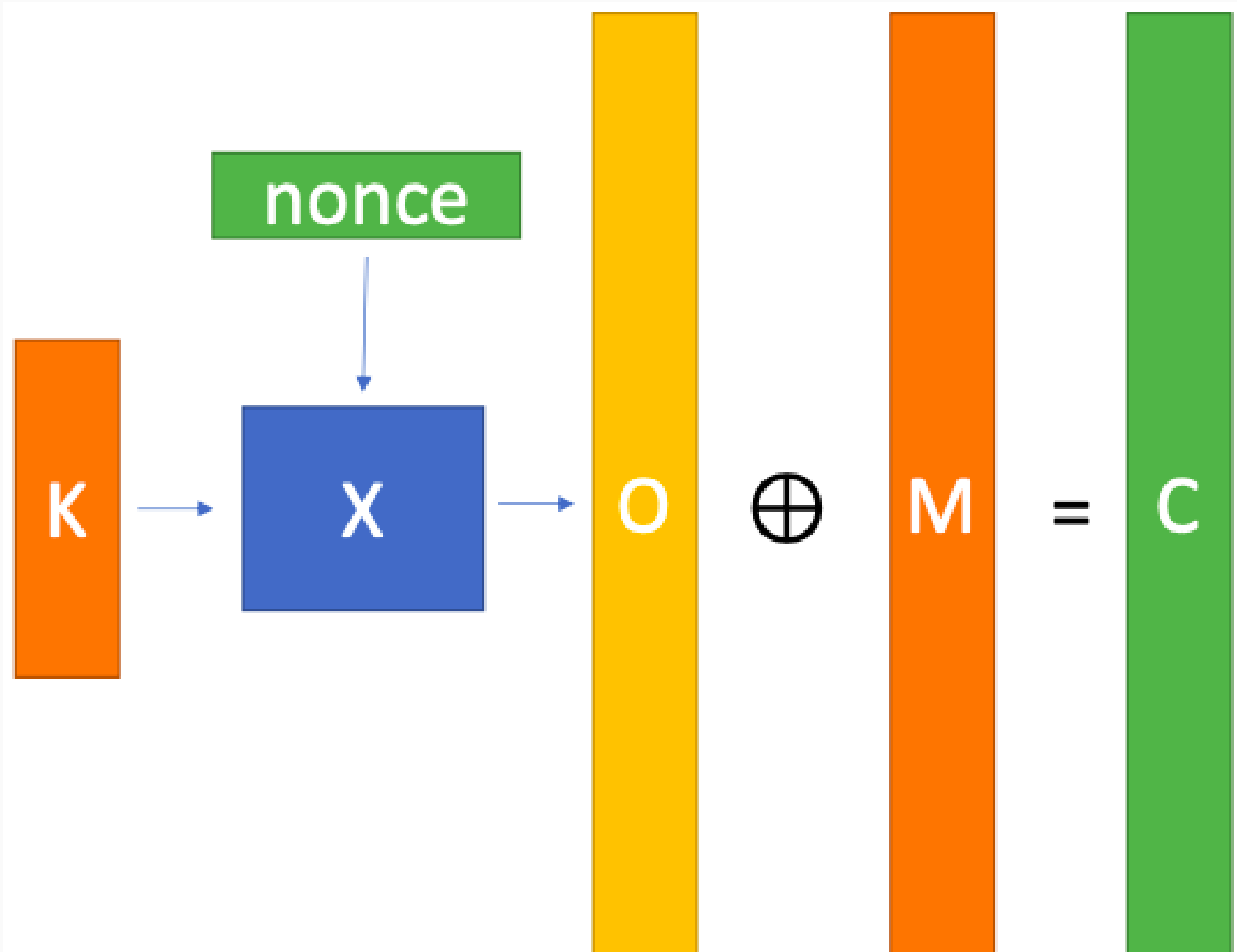
**Definition 5.3:** Consider any adversary who plays the above game, and whose computing power is limited in the sense that whatever algorithm he runs terminates in time much less than the time it takes to try all possible keys the cryptosystem. No such adversary can guess whether he is in case 1 or 2 with probability better than a random guess.

# Stream Ciphers

An algorithm $X$, which expends a short key $k$, and a nonce $n$ to a longer *looking* string $X(k, n)$. Which is then used to encrypt a message *as if* it was a one-time pad. That is, $c = m \bigoplus X(k, n)$. Decryption happens with $c = \bigoplus X(k, n) = m \bigoplus X(k, n) \bigoplus X(k, n) = m$.

Stream algorithms do not produce truly random outcomes. We cannot output more than $2^{length}$ different strings. (TJEK DET HER FARMAND SIDE 130)

# Stream Ciphers



# Block ciphers

Block ciphers encrypt a fixed size block of data, and outputs a block of the same size. Block ciphers have *modes of operation* (such as Cipher Block Chaining, Counter & Output feedback).

They have a *nonce*, which is called an *Initialisation Vector*. This must be chosen carefully and not be the same all the time.

**Examples:** 1. DES (56 bit keys, 64-bit blocks) 2. Triple-DES (112 or 168-bit keys, 64-bit blocks). 3. AES (HVAD ER SPECI?)

# OFB: Output Feedback

Generate a keystream. Take a nonce $IV$, then apply AES to it repeatedly.

$$AES_K(IV), AES_K(AES_K(IV)), \ldots,$$

Then we can XOR that with the message.

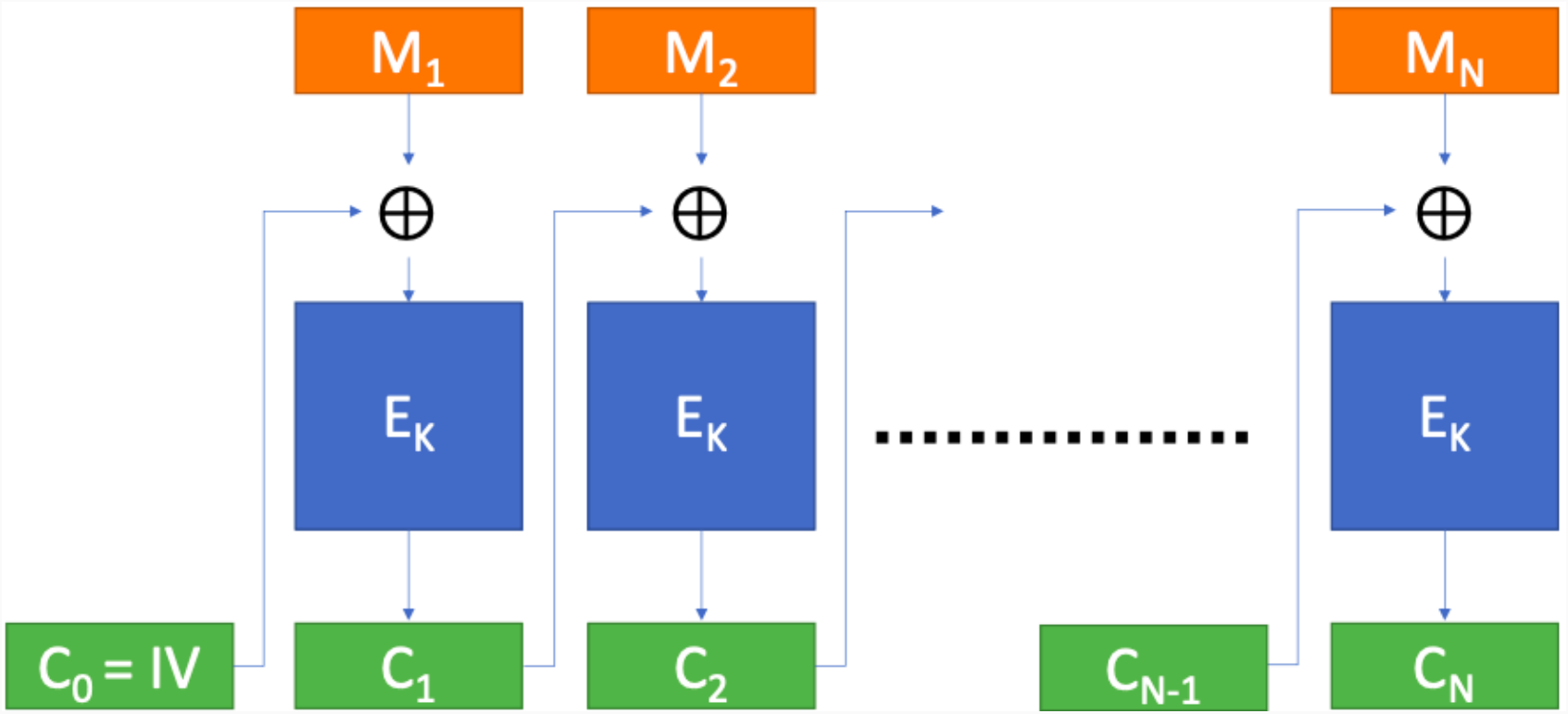
# CBC: Cipher Block Chaining

CBC is more common. Let the message consist of 128-bit blocks $M_1, \ldots, M_t$, we pad the last block. The cipher text will be $t + 1$ blocks $C_0, \ldots, C_t$ where $C_0 = IV$ and for $i = 1, \ldots, t$

$$C_i = AES_k(M_i \bigoplus C_{i-1})$$

The *IV* must be random to satisfy *indistinguishability under chosen-plaintext*. *IV* cannnot be implemented as a counter in CBC mode. (EXPLAIN!!!)

# CBC: Cipher Block Chaining



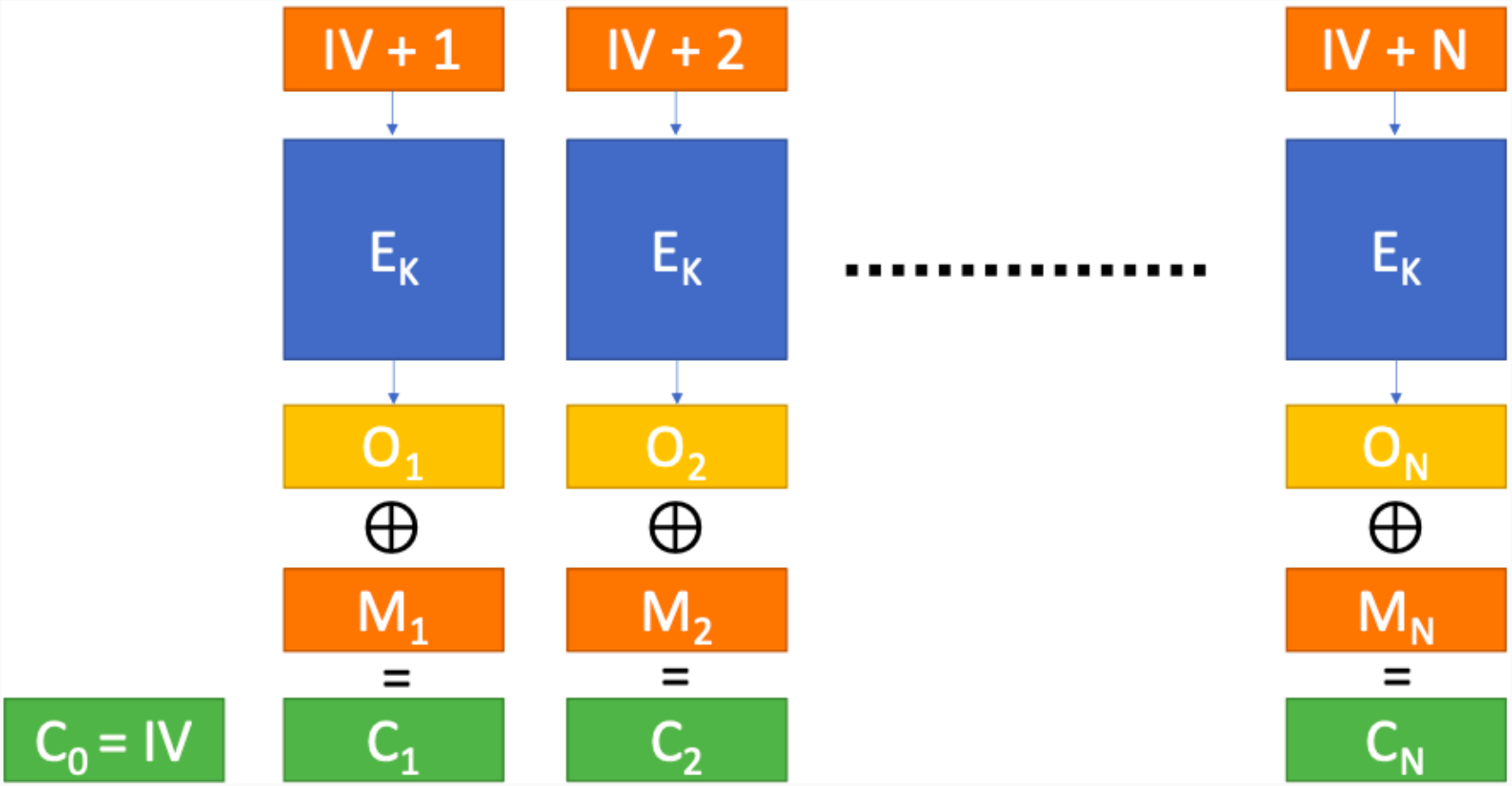
# CTR: Counter Mode

Let the message be $M_1, \ldots, M_t$ and the ciphertext depending on $IV$. The cipher text will be $t + 1$ blocks $C_0, \ldots, C_t$ where $C_0 = IV$ and for $i = 1, \ldots, t$

$$C_i = AES_K(IV + i) \bigoplus M_i$$

$IV + 1$ means that $IV$ is a 128-bit number. Add $i$ mod $2^{128}$. CTR makes it easier to decrypt several blocks than CBC in parallel.

# CTR: Counter Mode



# Computational Security, public-key systems

# Public-key Systems

Public-key systems also have three algorithms. $G, E, D$. Algorithm $G$ has input key of length $\kappa$ and outputs a pair of keys of length $\kappa$. Public key makes use of two matching keys, instead of one. It uses the following identity

$$m = D_{sk}(E_{pk}(m))$$

Secret-key systems also require the use of randomness. Otherwise they could win the oracle game.

# AES

MISSING????

# RSA

Relies on difficulty of prime factorisation. Let $N, e$ be the public key. Let $n, d$ be the private key. $N = p \cdot q$. Let $d = f(e, p, q)$. Below is the deterministic and thus insecure vanilla RSA. It relies on the following identity. Here $c = m^e \bmod n$.

$$c^d \bmod n = (m^e \bmod n)^d \bmod n = m$$

Size of keys must be big (2-3k bits), and can just grow. Let $d = e^{-1} \bmod (p-1)(q-1)$. RSA is commonly used.

# OAEP

1. Compute a padded version $OAEP(K, R)$. $R$ is a random string of bits. $|R|$ is chosen as a function of $N$. 2. Ciphertext is $c = OAEP(K, R)^e \bmod n$. 3. Receiver computes $(OAEP(K, R)^e \bmod n)^d = OAEP(K, R)$ Check that the format of the result is correct, and if so, receiver recovers $K$ from $OAE(K, R)$. Chose $OAEP$ such that this is possible.