CRYPTOGRAPHIC PRIMITIVES

AN INTRODUCTION TO THE THEORY AND PRACTICE BEHIND MODERN CRYPTOGRAPHY

Reactive.IO

Robert Sosinski

Founder & Engineering Fellow

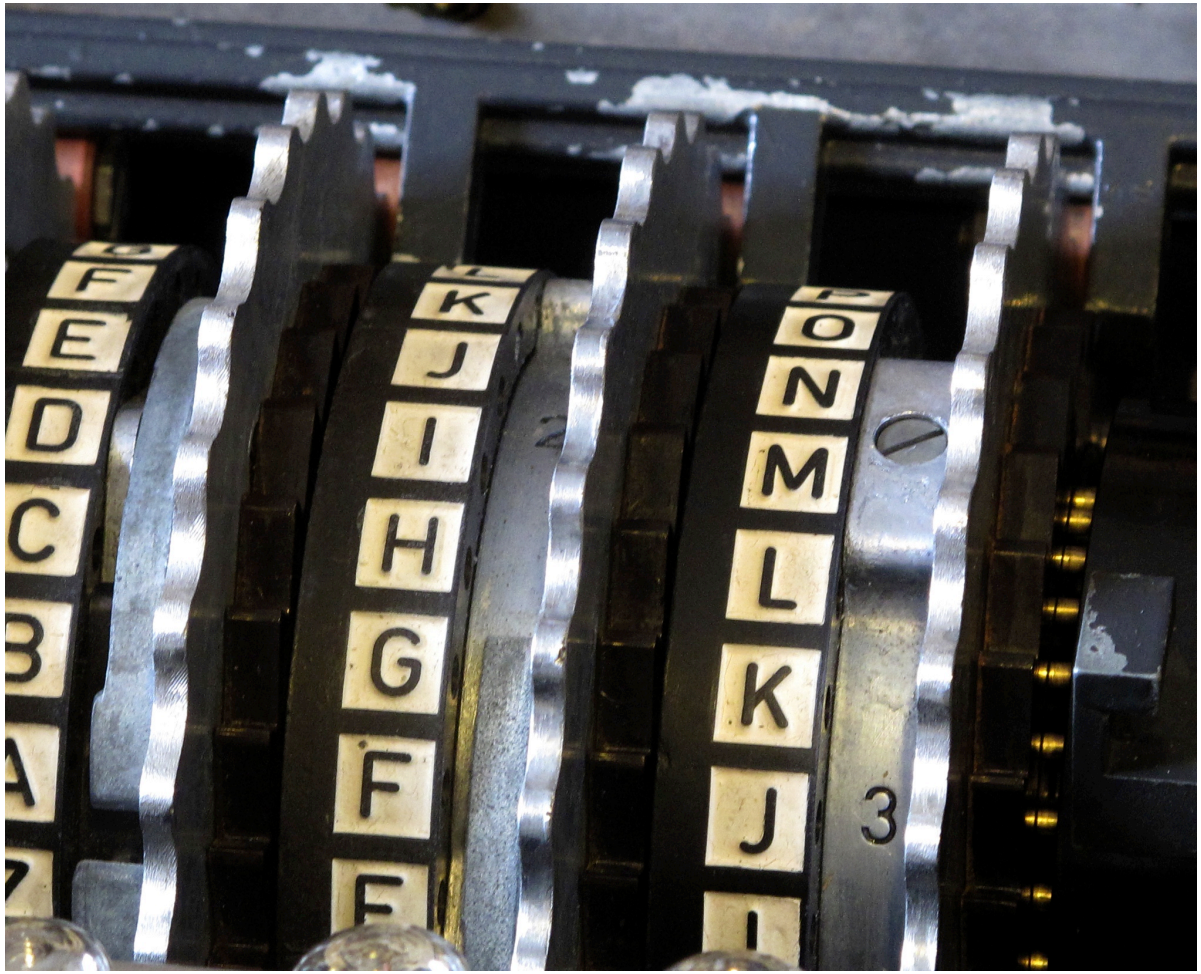
Known as "America's Cryptologic Wing",
is the only Air Force wing that supports the National Security Agency,
Air Intelligence Agency and the entire United States Air Force (USAF)
with cryptologic intelligence.

~ wikipedia.org

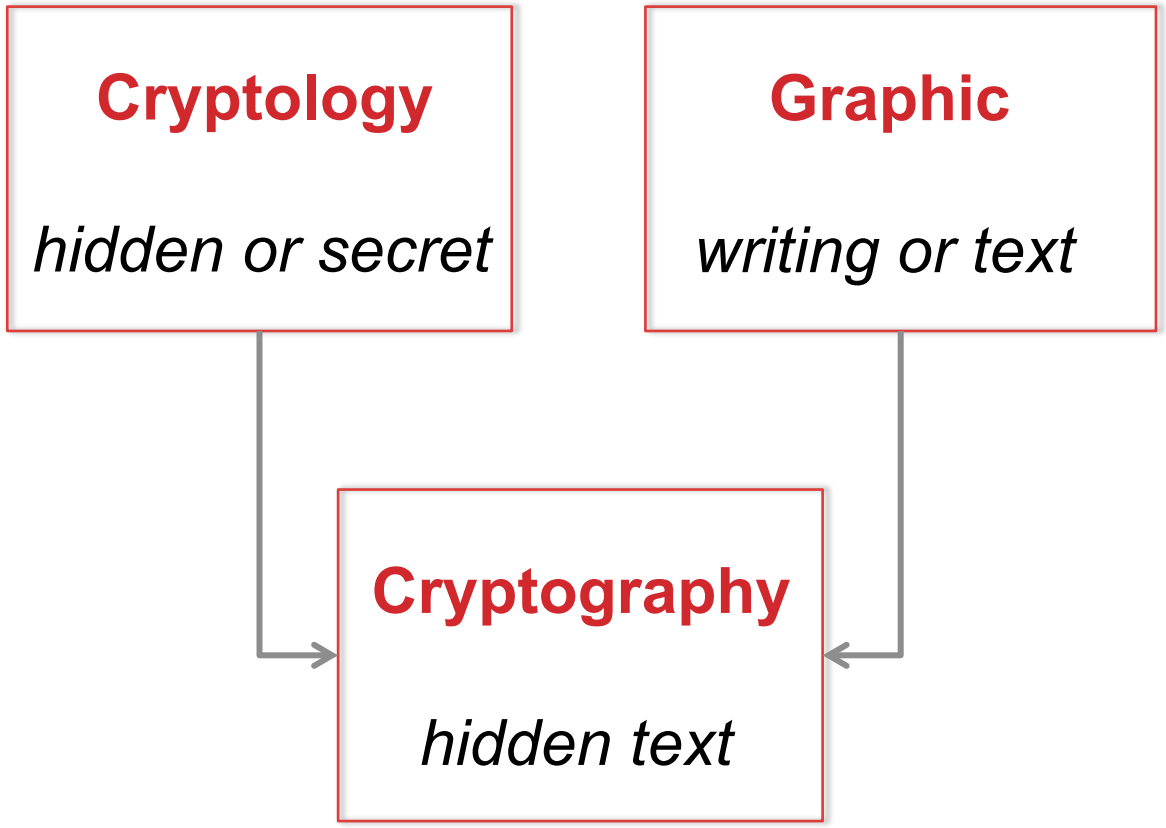
AGENDA

- **Definition:** what cryptography is and is not
- **The Basics:** making your first algorithm
- **Cryptographic Attacks:** break your first code
- **Symmetric Encryption:** one key goes both ways
- **Message Digests:** many sizes in, one size out
- **Asymmetric Encryption:** two keys go many ways
- **Randomness:** don’t just pick a number
- **Summary:** bringing everything together
- **Questions:** should be a couple

WHAT IS CRYPTOGRAPHY?



WHAT IS CRYPTOGRAPHY?



WHAT IS CRYPTOGRAPHY?

The study and practice of securing communication in the presence of adversaries.

Cryptography is a tool that can help provide:

- **Confidentiality: adversary cannot read**
- **Integrity: adversary cannot change**
- **Authenticity: recipient can trust**
- **Acknowledged: sender cannot repudiate**

GETTING PRIMITIVE

Cryptography has many disciplines, known as cryptographic primitives

- **Encryption and Decryption**
- **One-Way Functions (Hash, Digest)**
- **Authentication**
- **Digital Signatures**
- **Entropy and Randomness**

MAKING (OR BREAKING) THE SYSTEM

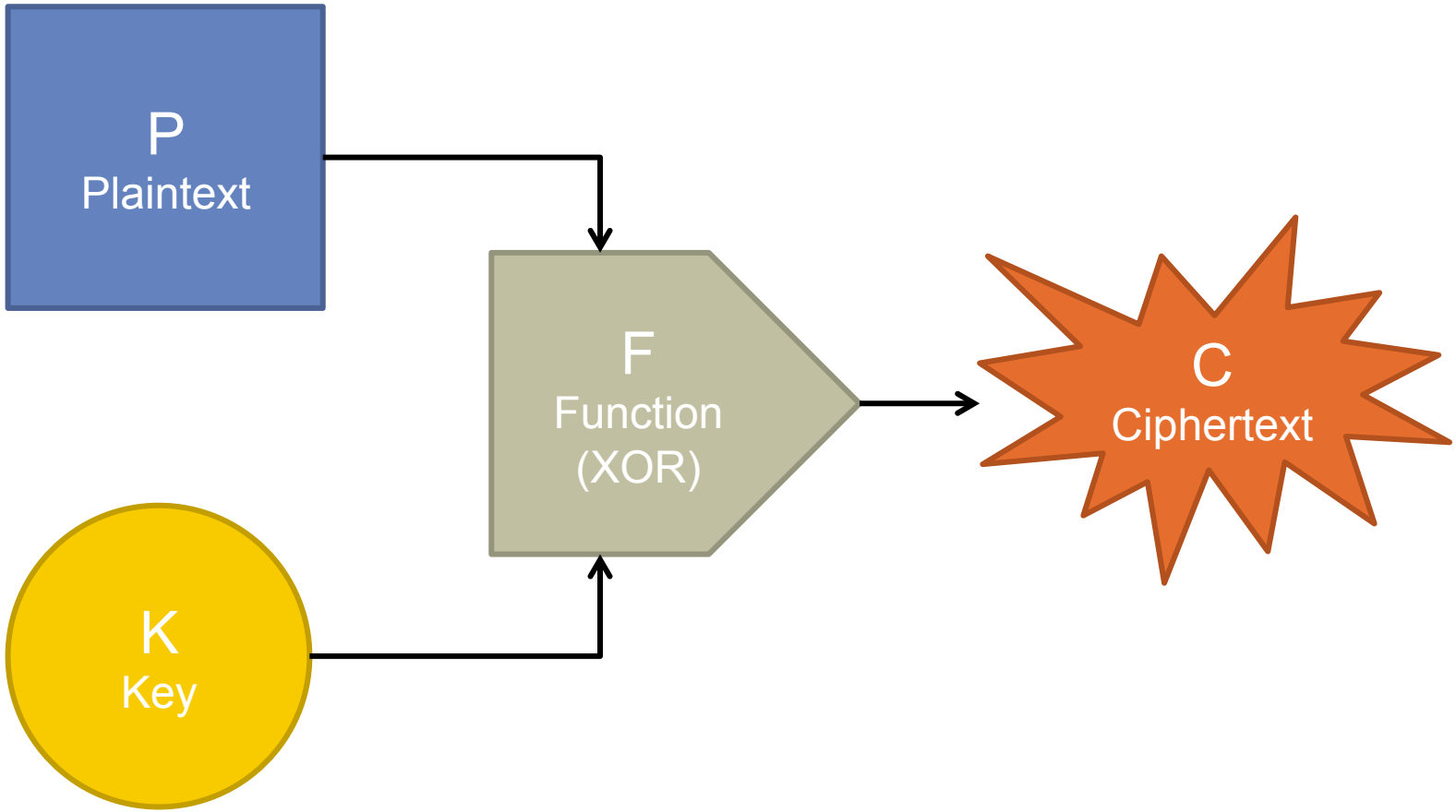
Combining multiple cryptographic primitives together makes a cryptographic system

- **SSL and TLS**
- **PGP and GnuPG**
- **SSH and VPN**
- **SFTP and FTPS**
- **OAuth and SAML**
- **Even Bitcoin**

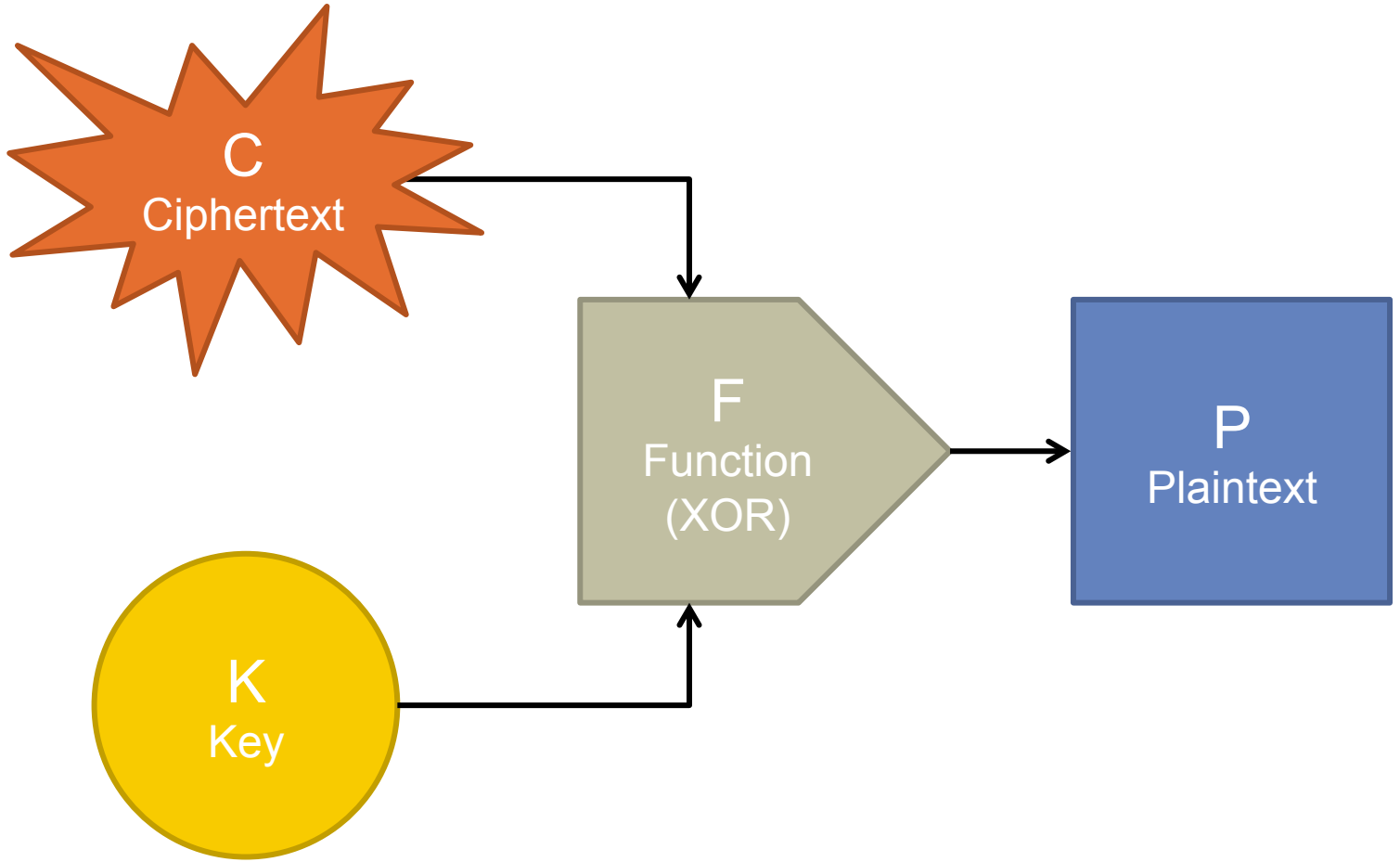
LET'S MAKE AND BREAK YOUR VERY FIRST CODE

Let's make a basic cryptographic algorithm and see how it fares

OUR ENCRYPTION ALGORITHM



OUR DECRYPTION ALGORITHM



CIPHER AND DECIPHER “A”

A	0100 0001
K	0100 1011
\n	0000 1010

\n	0000 1010
K	0100 1011
A	0100 0001

CIPHER AND DECIPHER “Z”

z	0111 1010
K	0100 1011
1	0011 0001

1	0011 0001
K	0100 1011
z	0111 1010

PLAIN TEXT ATTACK!

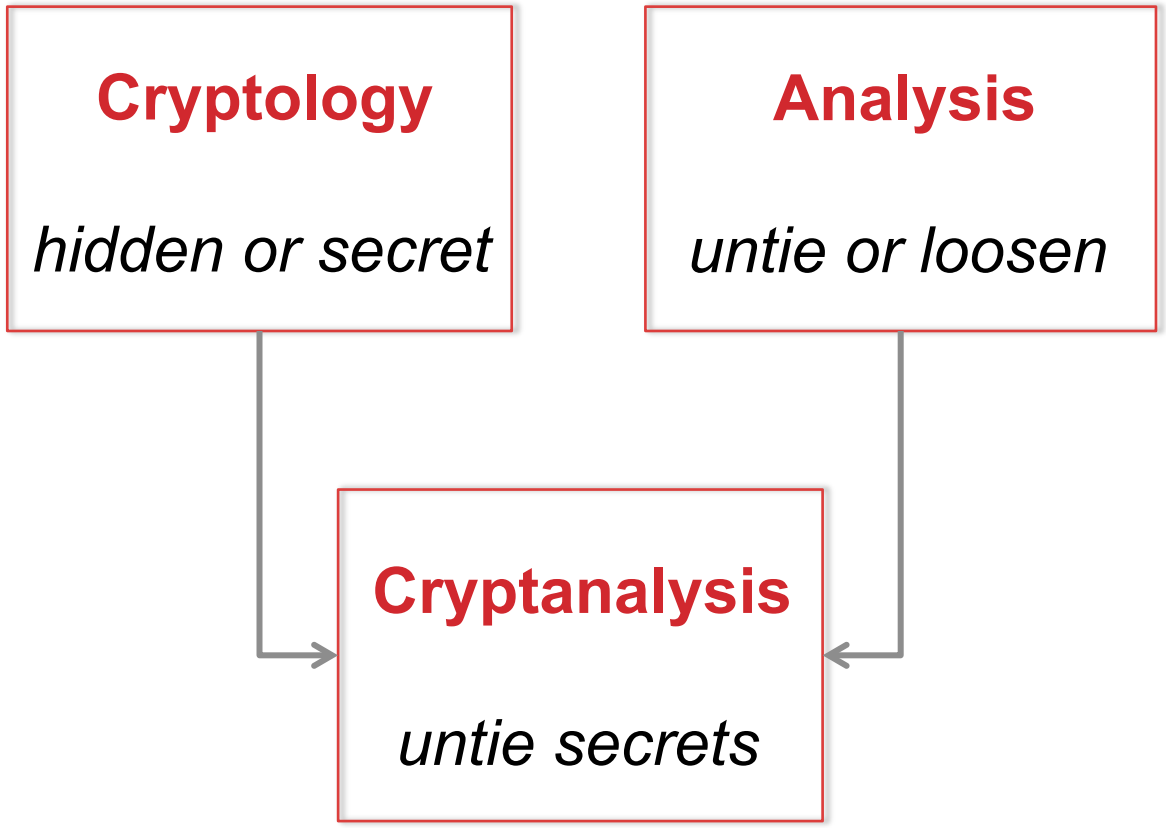
A	0100 0001
\n	0000 1010
K	0100 1011

z	0111 1010
1	0011 0001
K	0100 1011

ENSURING KEY PROTECTION

A good cryptographic algorithm protects the key

CRYPTANALYSIS



CRYPTOGRAPHIC ATTACKS

Known Plaintext Attacks

Attacker knows the plaintext associated with the ciphertext.

- **Encrypting information already made public**
- **Using the same email signature for all recipients**

Chosen Plaintext Attacks

Attacker has the message sender encrypt plaintext of his choosing, and gets access to the corresponding ciphertext

- **“Gardening” during WWII with minesweeper ships**
- **Sending queries to a system and matching their output**

SYMMETRIC ENCRYPTION

Stream Ciphers

Each plaintext character is encrypted with a different key character.

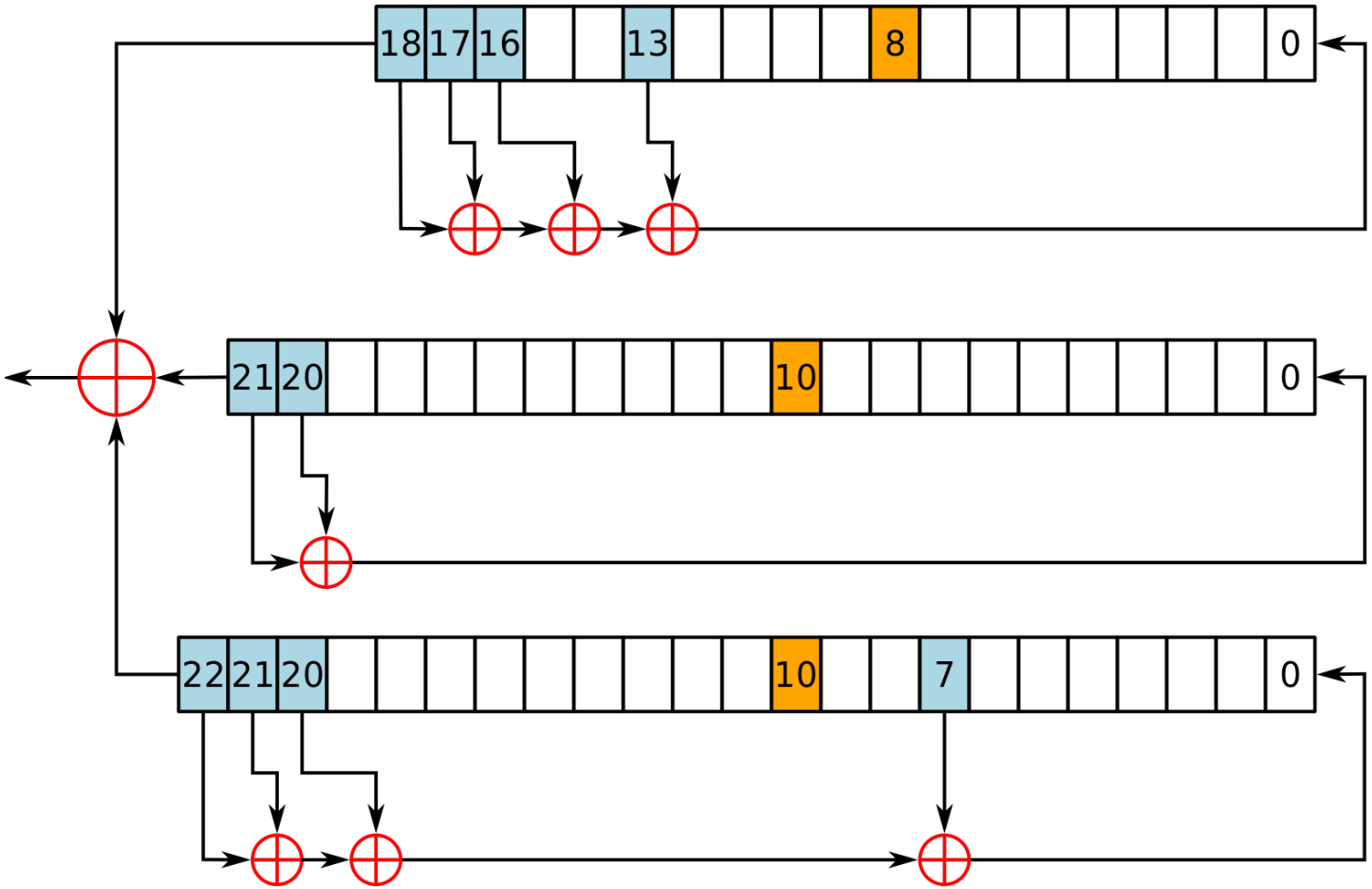
Benefits are increased speed

Block Ciphers

A block of plaintext is manipulated, often several times, with a different block of key material.

Benefits are increased security

A5/1 – USED IN GSM PHONES



HARDENING THE ALGORITHM

Confusion

- Finding the relationship between the a key and ciphertext should be as complex and involved as possible.
- The key should be protected from exposure even when an attacker has large amounts of ciphertext to analyze.

Diffusion

- The statistical structure of plaintext should be dissipated over the bulk of ciphertext.
- Changing 1 bit of plaintext should change 50% of the overall ciphertext structure (an avalanche).

MIXING THINGS UP WITH S&P NETWORKS

Substitution

- Take one unique set of input bits and returns another completely different set output bits
- Adds confusion to a cryptographic algorithm

Permutation

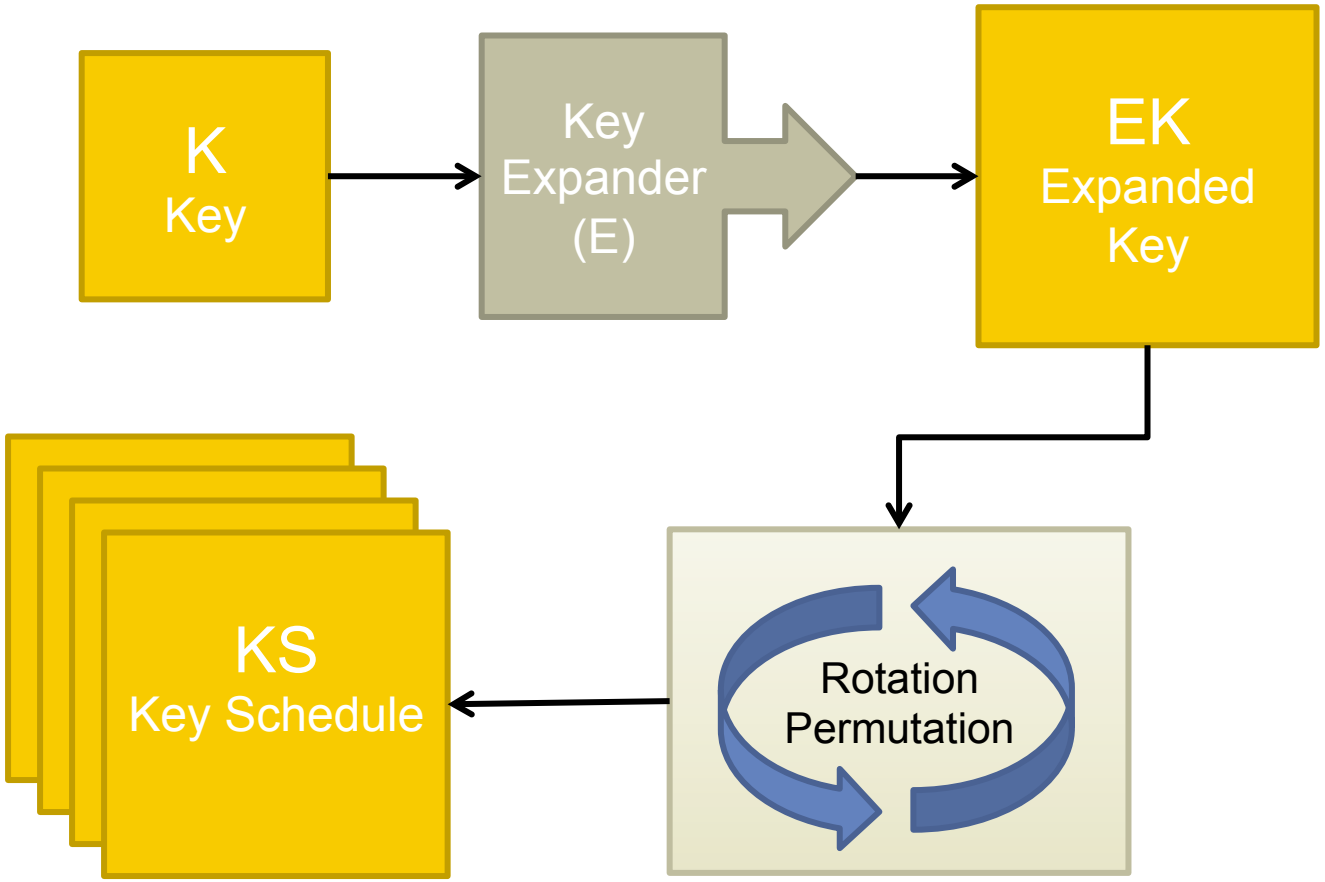
- Take one unique set of input bits and returns the same set of output bits, however with different ordering
- Adds diffusion to a cryptographic algorithm

PRODUCT CIPHERS

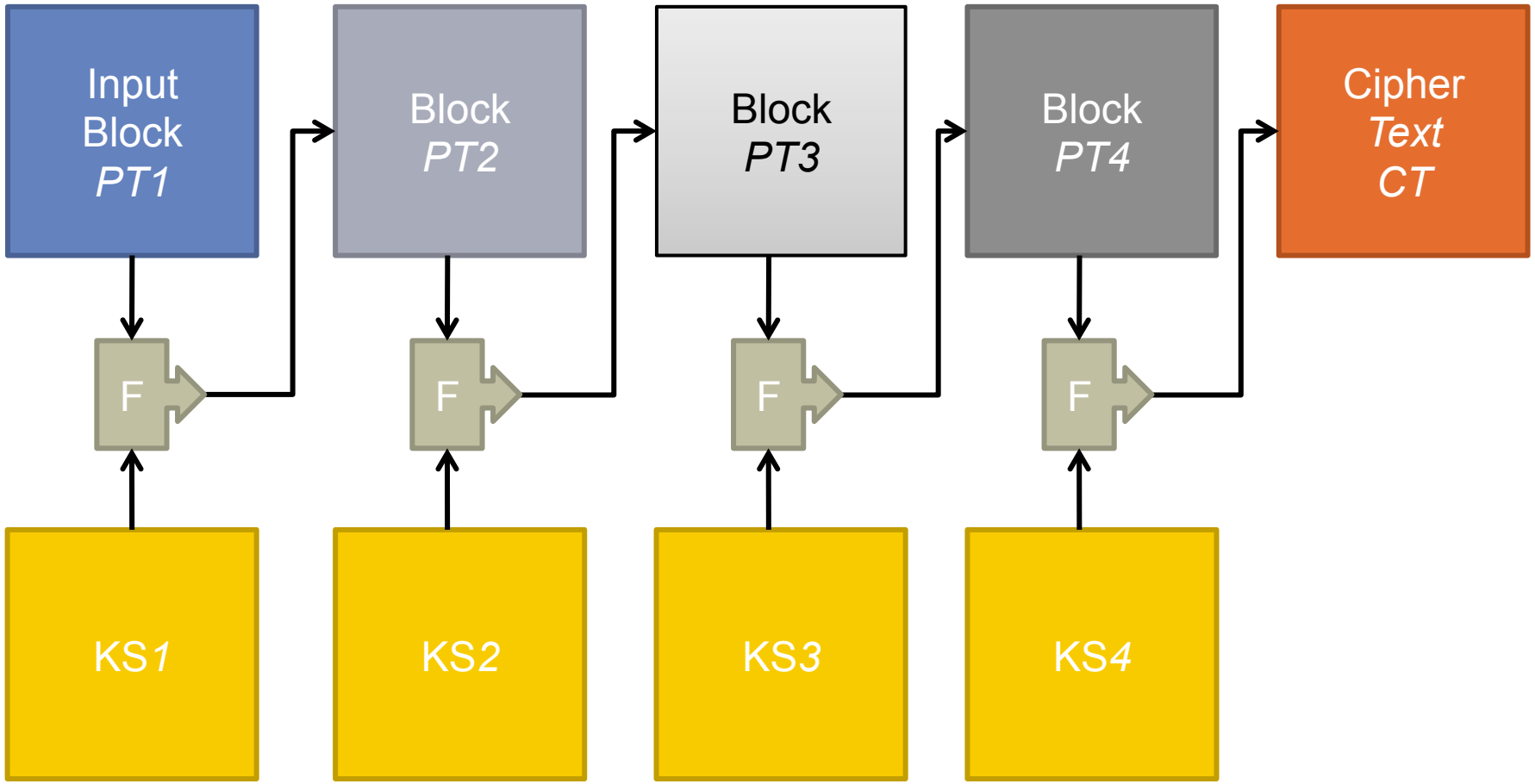
Block Ciphers that also have the following criteria:

- **Operate on a fixed size block of text (e.g. 128bit)**
- **Utilizes a specific key size (e.g. 128bit, 192bit, 256 bit)**
- **Undergo multiple rounds of encryption (rounds)**
- **Keys are expanded via a key schedule**
- **Each round of encryption performs:**
  - **Substitution: adding confusion**
  - **Permutation: adding diffusion**
  - **Key Mixing: encrypting data**
  - **Expansion and reduction**

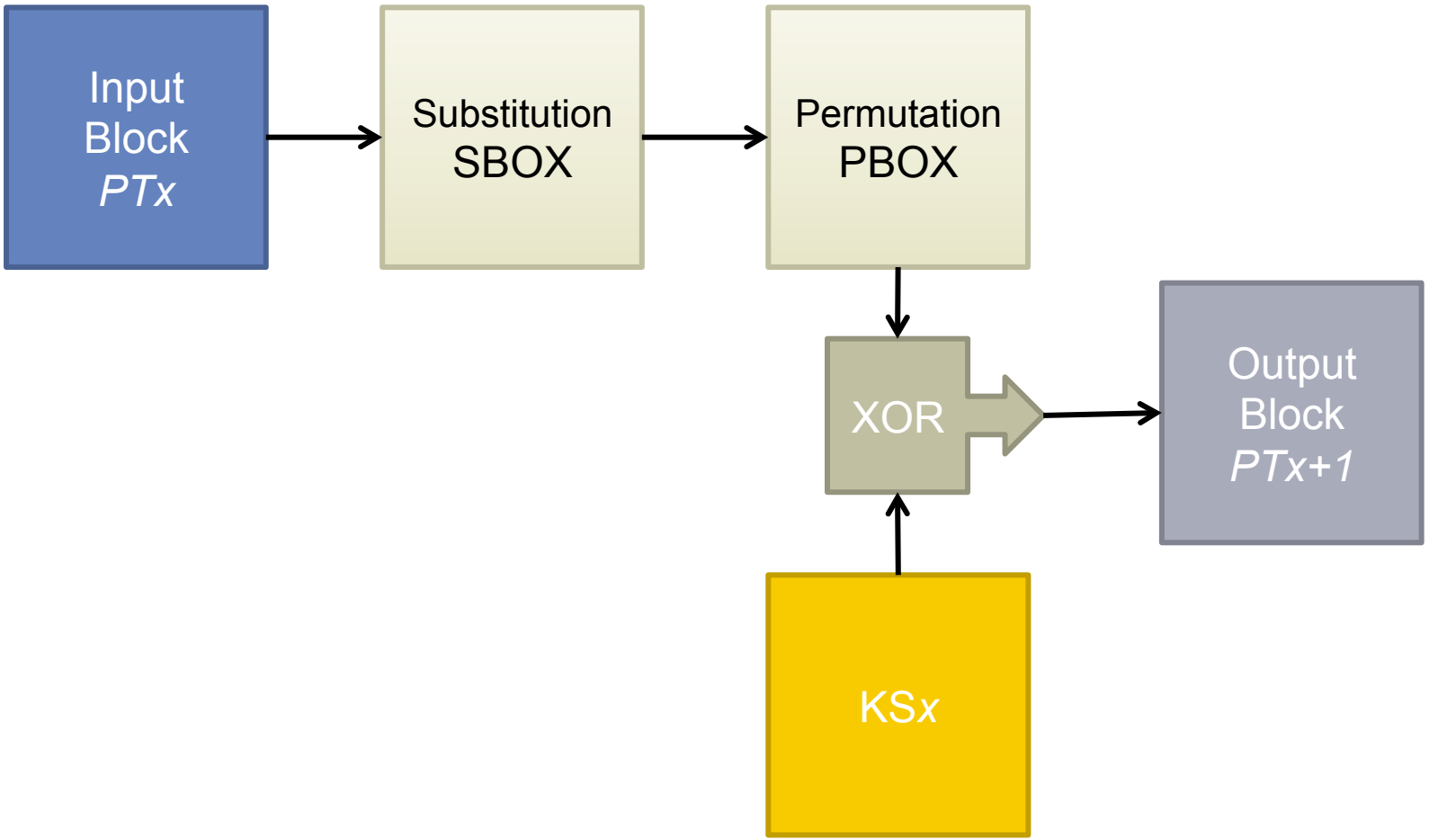
KEY EXPANSION



ROUNDS OF ENCRYPTION



EACH ROUND OF ENCRYPTION IS PROCESSED



PERMUTATION BOX - P

Input Block *PTx*

14	20	15	21	6	10	29	4
5	32	19	23	26	16	13	24
25	1	31	17	11	2	39	7
27	8	9	18	28	12	22	3

Permutation PBOX

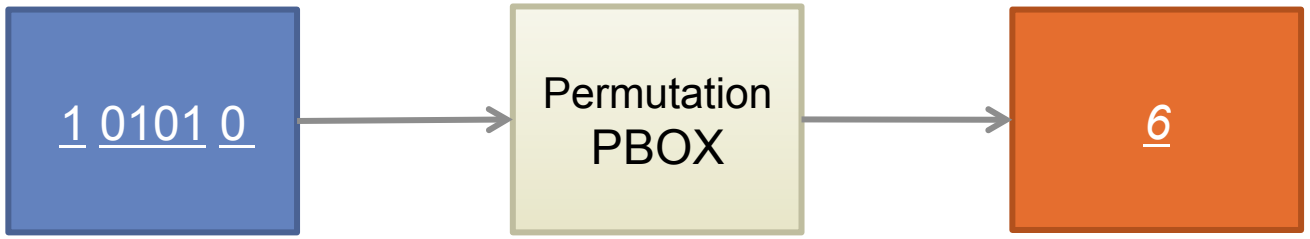
16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Output *Block PTx+1*

5	25	39	22	26	18	20	1
7	24	12	10	8	17	32	14
4	16	27	20	21	28	19	31
3	23	2	29	6	9	13	11

SUBSTITUTION BOX - S1

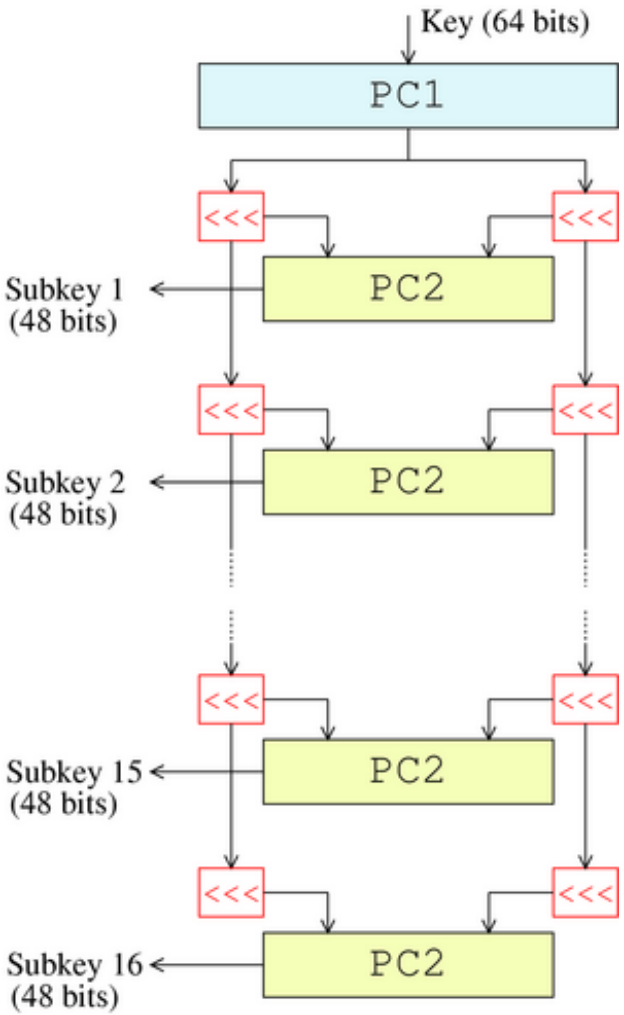
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
01	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
10	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
11	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

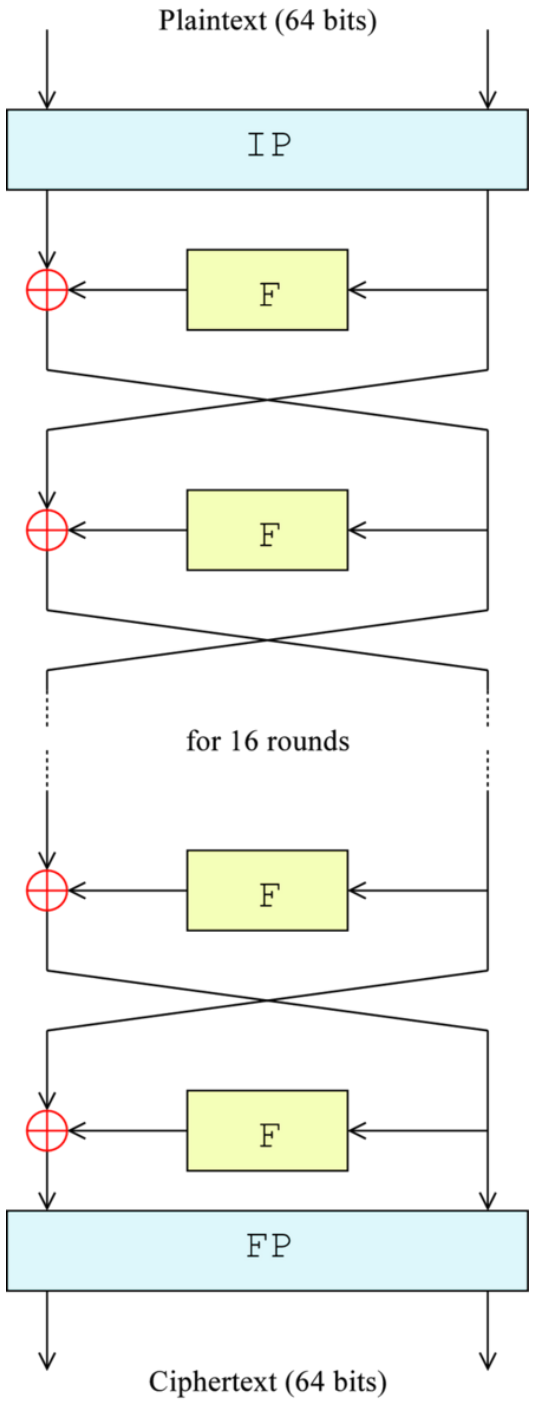


DES - DATA ENCRYPTION STANDARD

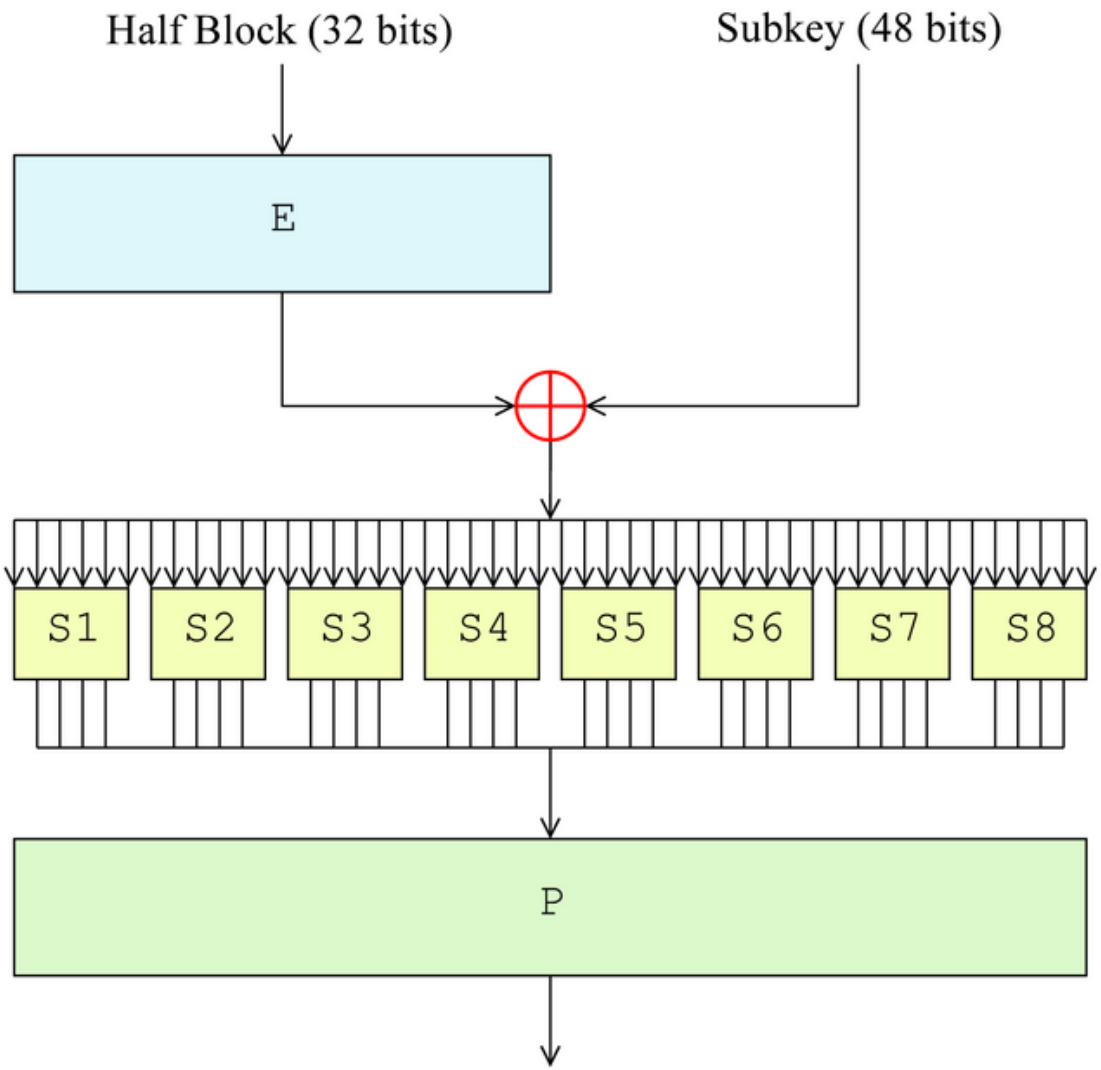
- Developed in 1976 by IBM
- Selected by the National Bureau of Standards
- 64 Bit key, although really only 56 Bits, as every 8th Bit is used for error-checking purposes
- 64 Bit block and 16 Rounds of processing
- Often used three times, known as Triple DES
- Not secure since 1999, use AES instead
- You can learn more by reading FIPS46-3

KEY SCHEDULE AND ROUNDS





FEISTEL NETWORK

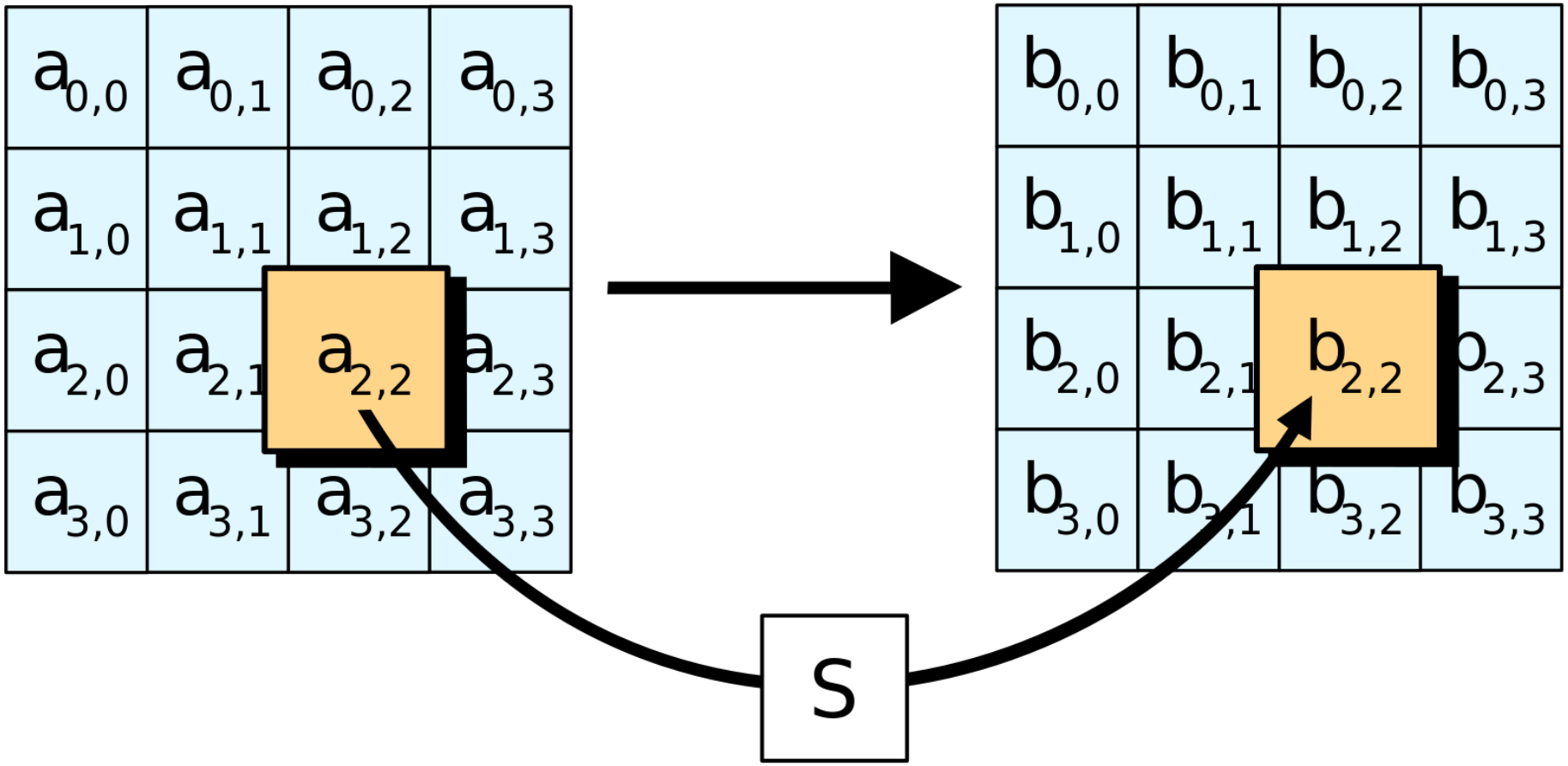


AES - ADVANCED ENCRYPTION STANDARD

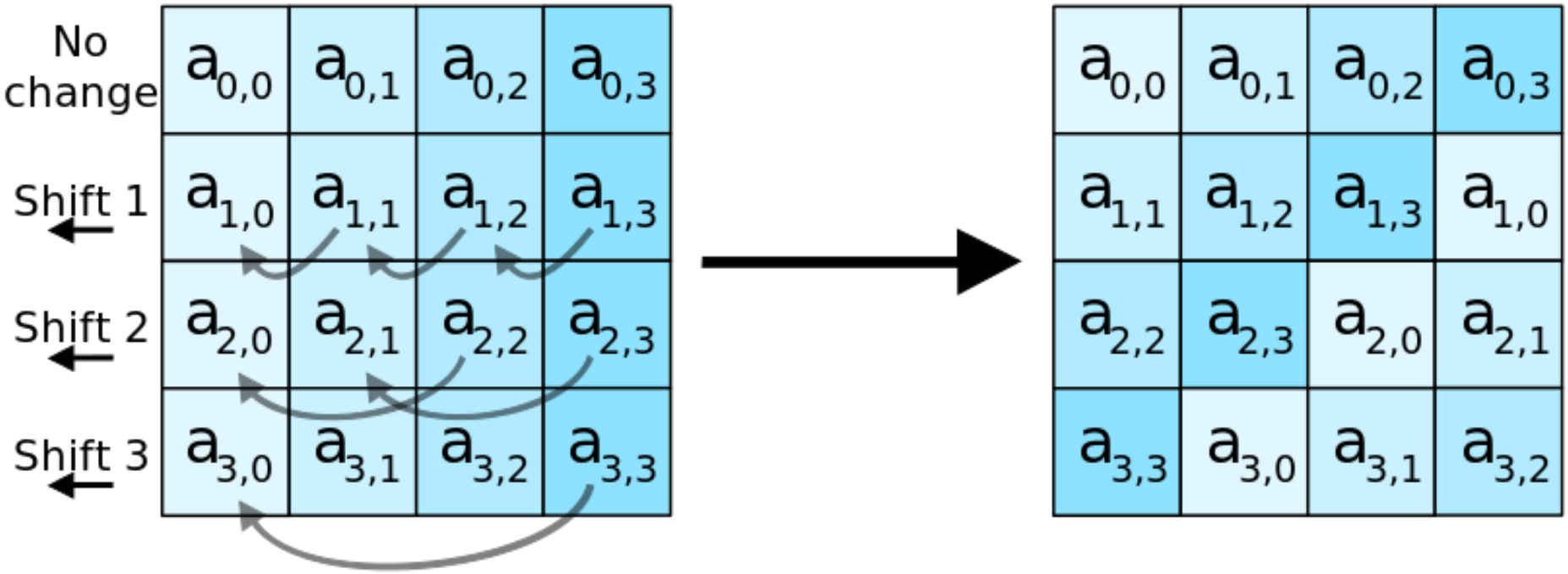
- Known as Rijndael, it was developed by Belgian cryptographers Vincent Rijmen and Joan Daemen during the AES competition
- Selected by the NIST to deprecate DES in 2001
- Supports a 128, 192, or 256 Bit Key
- 128 Bit block, that performs 10, 12, or 14 rounds
- When you need to use symmetric encryption, you should probably use AES
- You can learn more by reading FIPS 197

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00	63	7c	77	7b	f2	6b	6f	d5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	fb	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d3
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	d6	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	c8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	94
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

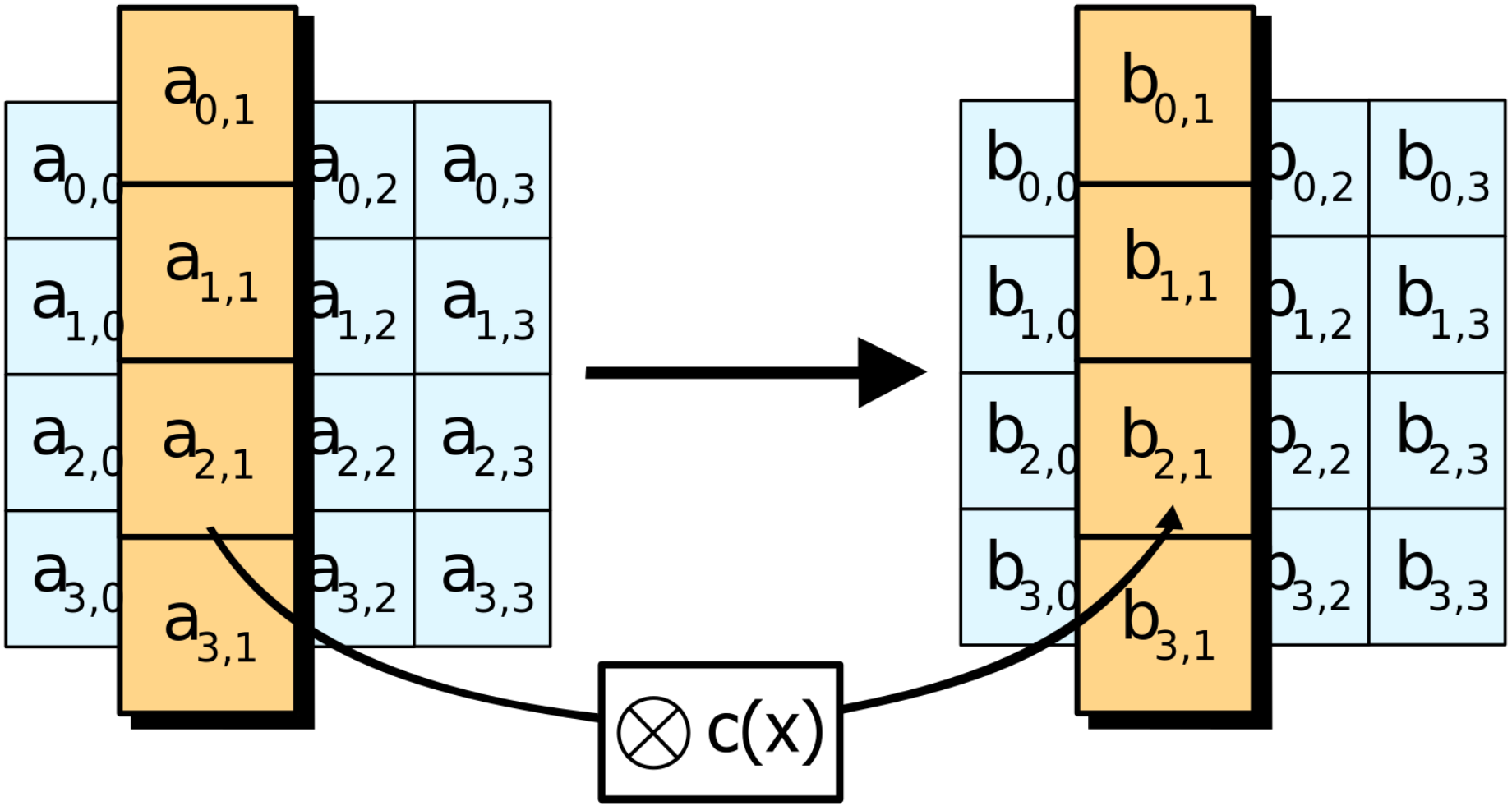
SUB BYTES

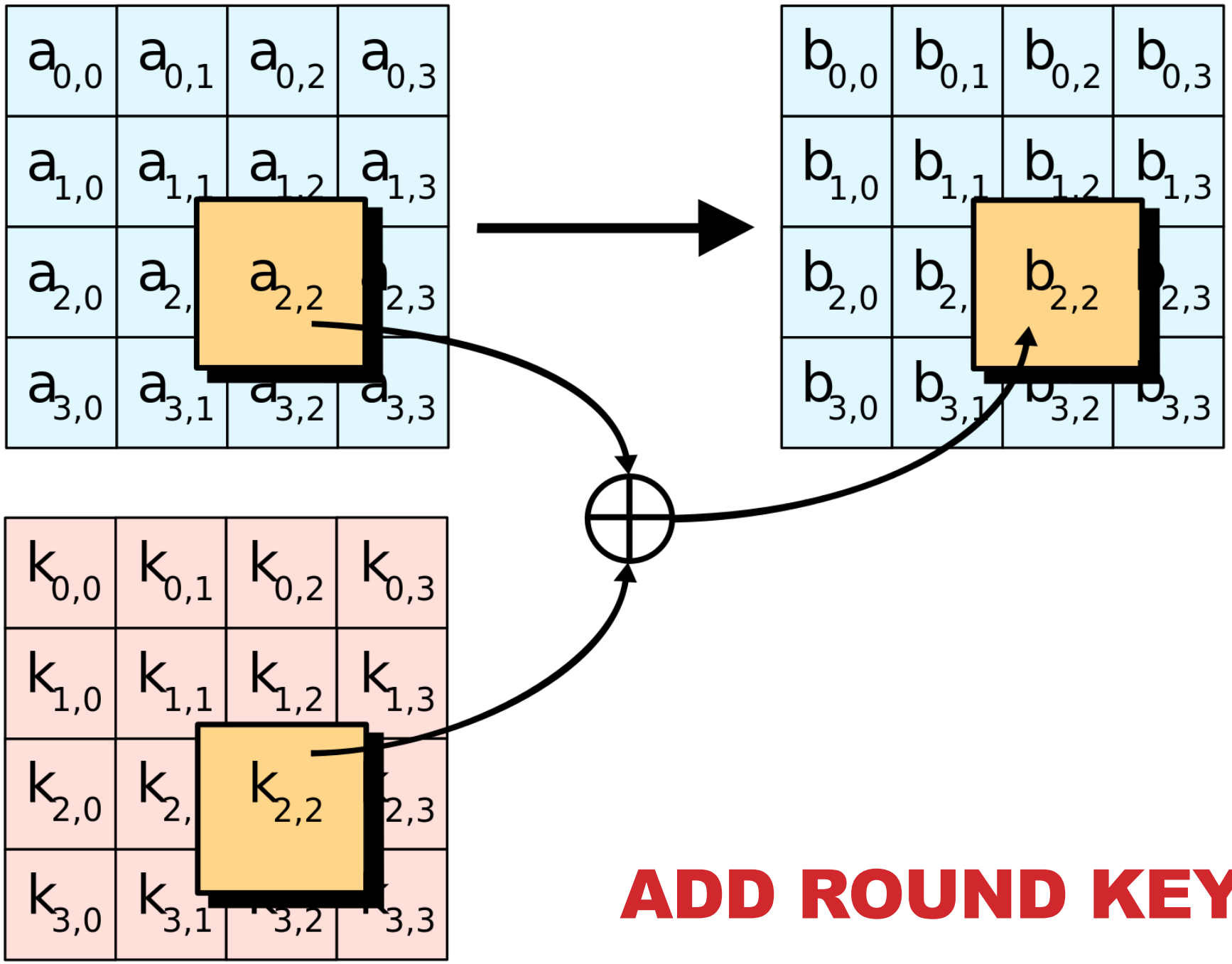

a0,0 a0,1 a0,2 a0,3
a1,0 a1,1 a1,2 a1,3
a2,0 a2,1 a2,2 a2,3
a3,0 a3,1 a3,2 a3,3
b0,0 b0,1 b0,2 b0,3
b1,0 b1,1 b1,2 b1,3
b2,0 b2,1 b2,2 b2,3
b3,0 b3,1 b3,2 b3,3
S

SHIFT ROWS


No change
Shift 1
Shift 2
Shift 3
a0,0 a0,1 a0,2 a0,3
a1,0 a1,1 a1,2 a1,3
a2,0 a2,1 a2,2 a2,3
a3,0 a3,1 a3,2 a3,3
a0,0 a0,1 a0,2 a0,3
a1,1 a1,2 a1,3 a1,0
a2,2 a2,3 a2,0 a2,1
a3,3 a3,0 a3,1 a3,2

MIX COLUMNS


a0,1
a1,1
a2,1
a3,1
a0,0
a0,2
a0,3
a1,0
a1,2
a1,3
a2,0
a2,2
a2,3
a3,0
a3,2
a3,3
b0,1
b1,1
b2,1
b3,1
b0,0
b0,2
b0,3
b1,0
b1,2
b1,3
b2,0
b2,2
b2,3
b3,0
b3,2
b3,3
c(x)



ADD ROUND KEY

CIPHER MODES

When the amount of plaintext exceeds the block size of an algorithm, a cipher mode must be used

ECB - Electronic Code Book

- **Message is divided into blocks and encrypted**

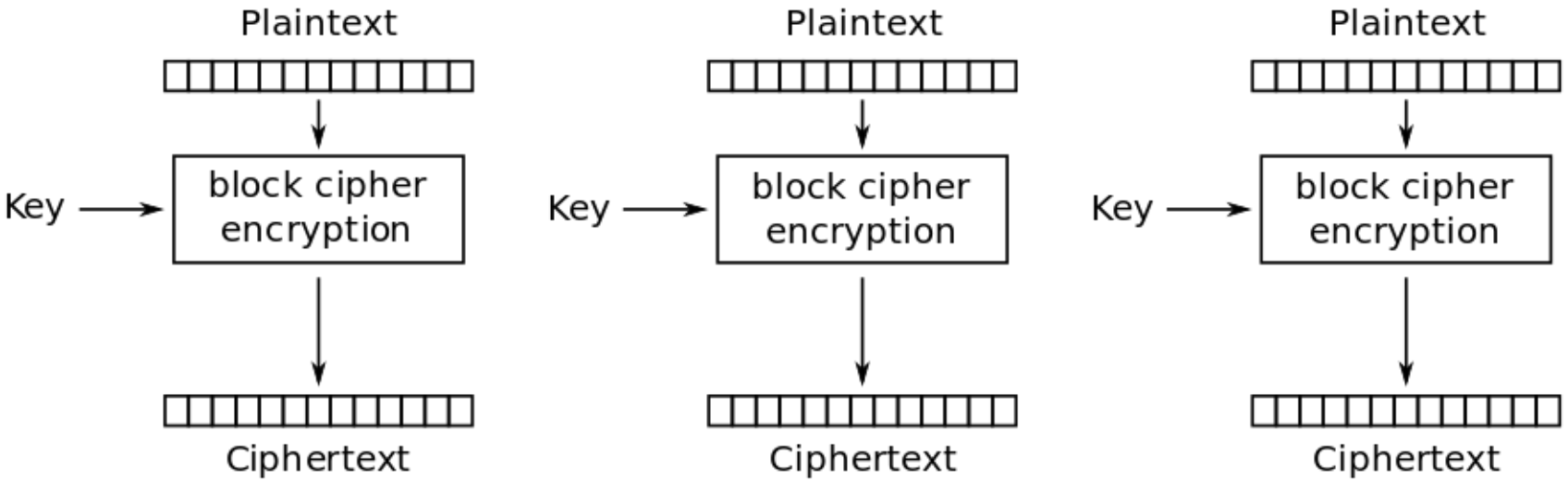
CBC - Cipher Block Chaining

- **Each block is XORed with previous block**

Counter

- **Generates a key stream by incrementing a counter or making a nonce (number used once)**

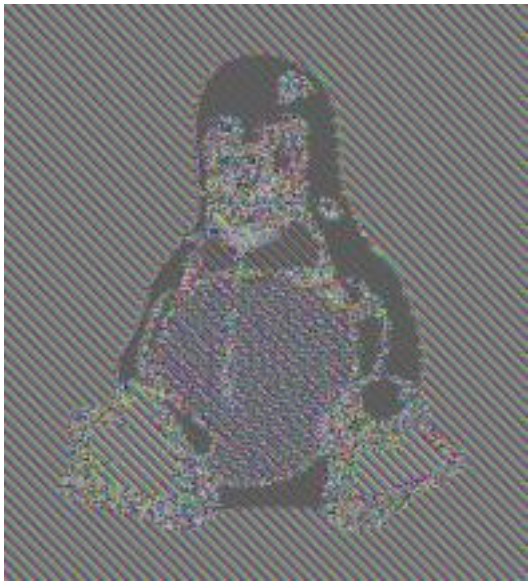
ELECTRONIC CODE BOOK



Electronic Codebook (ECB) mode encryption

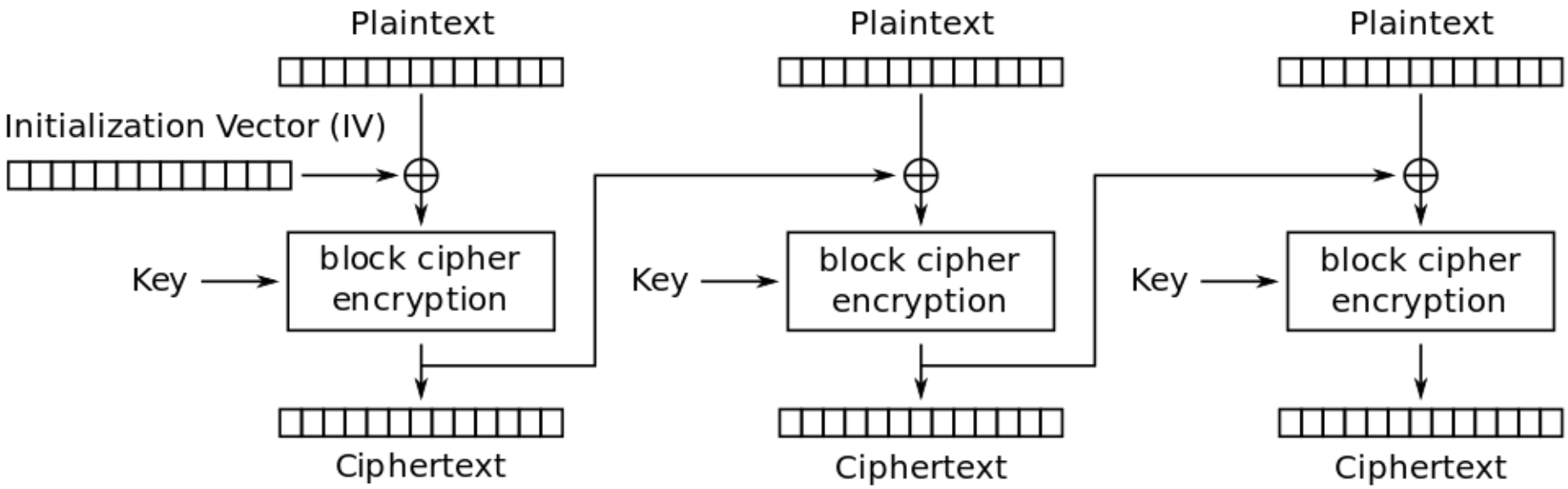
QUALITY OF ENCRYPTION ECB

Before



After

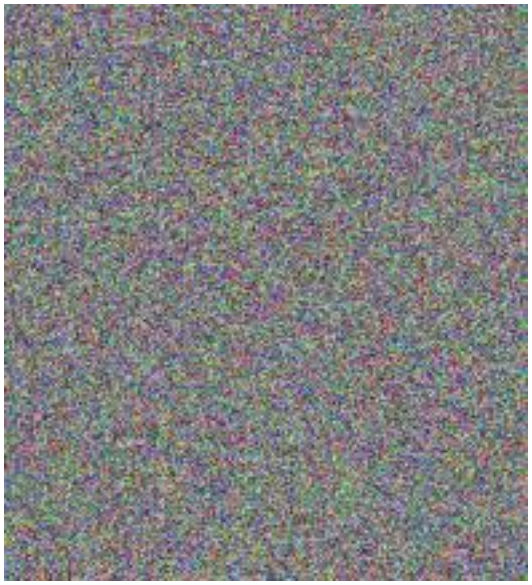
CIPHER BLOCK CHAINING



Cipher Block Chaining (CBC) mode encryption

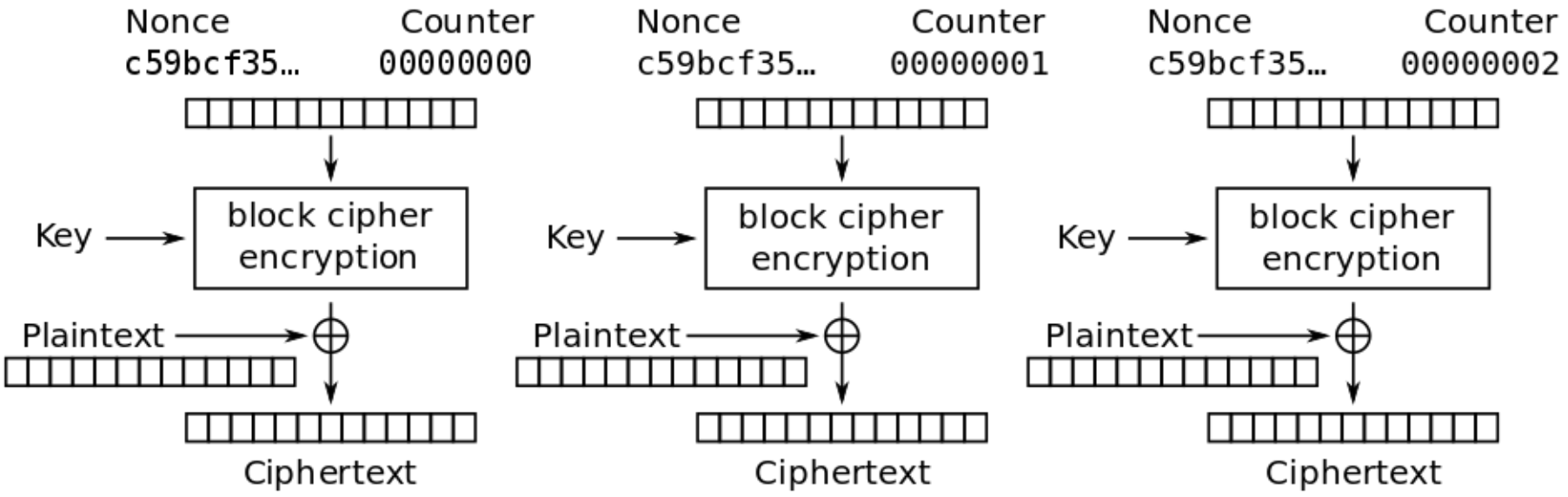
QUALITY OF ENCRYPTION CBC

Before



After

COUNTER



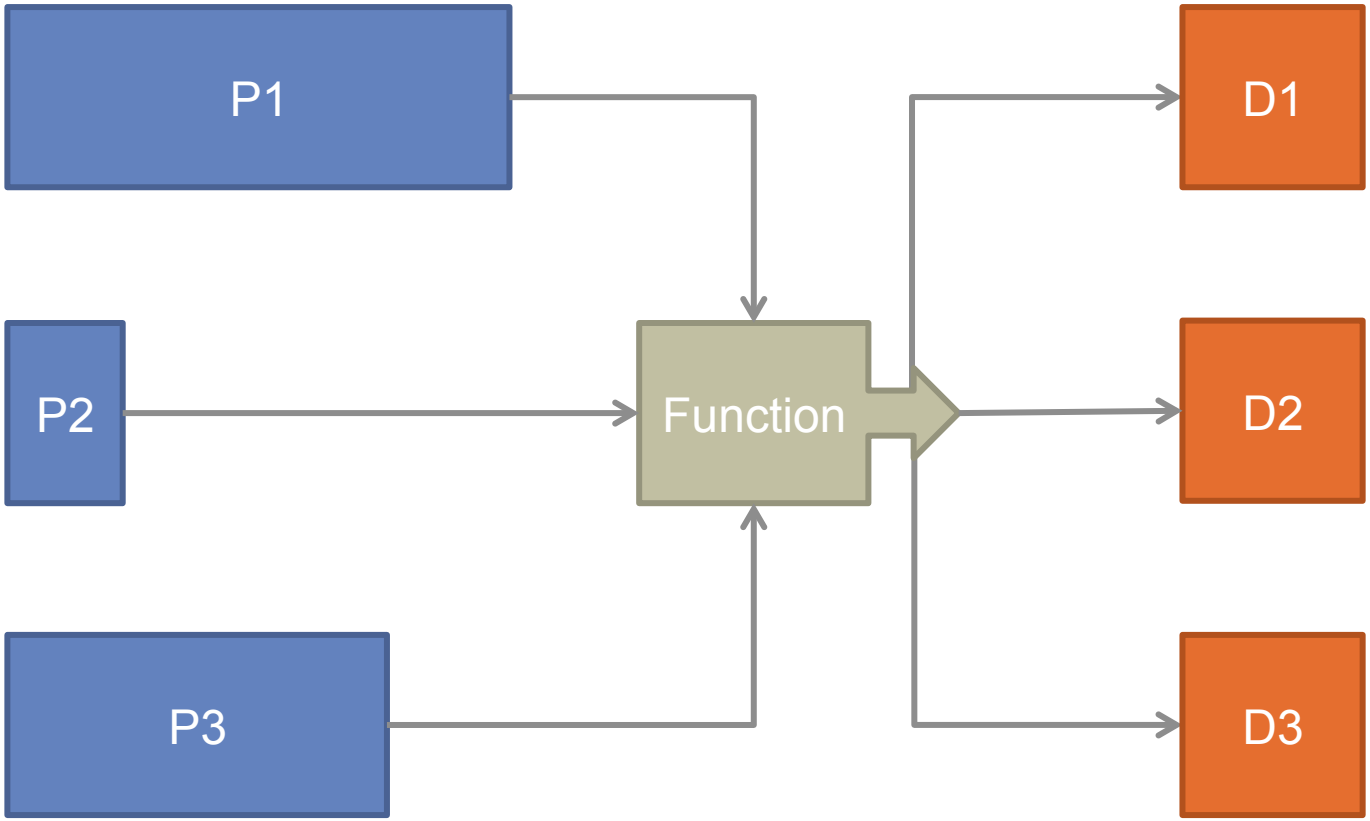
Counter (CTR) mode encryption

ONE WAY FUNCTIONS

Cryptographic Digests, also known as Hashing Functions or One-Way Functions:

1. **A variable length input (Pre Image) relates deterministically to a fixed length output (Digest)**
2. **The process of computing a pre image to a digest should be computationally easy**
3. **The process of inversely relating a digest to the it's corresponding pre image should be extremely difficult and computationally expensive**

DIGESTS AT WORK



TYPES OF DIGEST FUNCTIONS

- **Any Block Cipher:** Such as AES or DES
- **MD5:** Machine Digest (version 5)
- **SHA1:** Secure Hashing Algorithm)
- **SHA2:** 224 Bit through 512 Bit
- **Whirlpool:** Made by the designers of AES
- **RIPEMD160:** Designed in the open

WHEN TO USE DIGESTS

Generating unique “fingerprints” for complex pre images

- Giving each multi-gigabyte video a unique id
- Compiling multiple data points into a single opaque

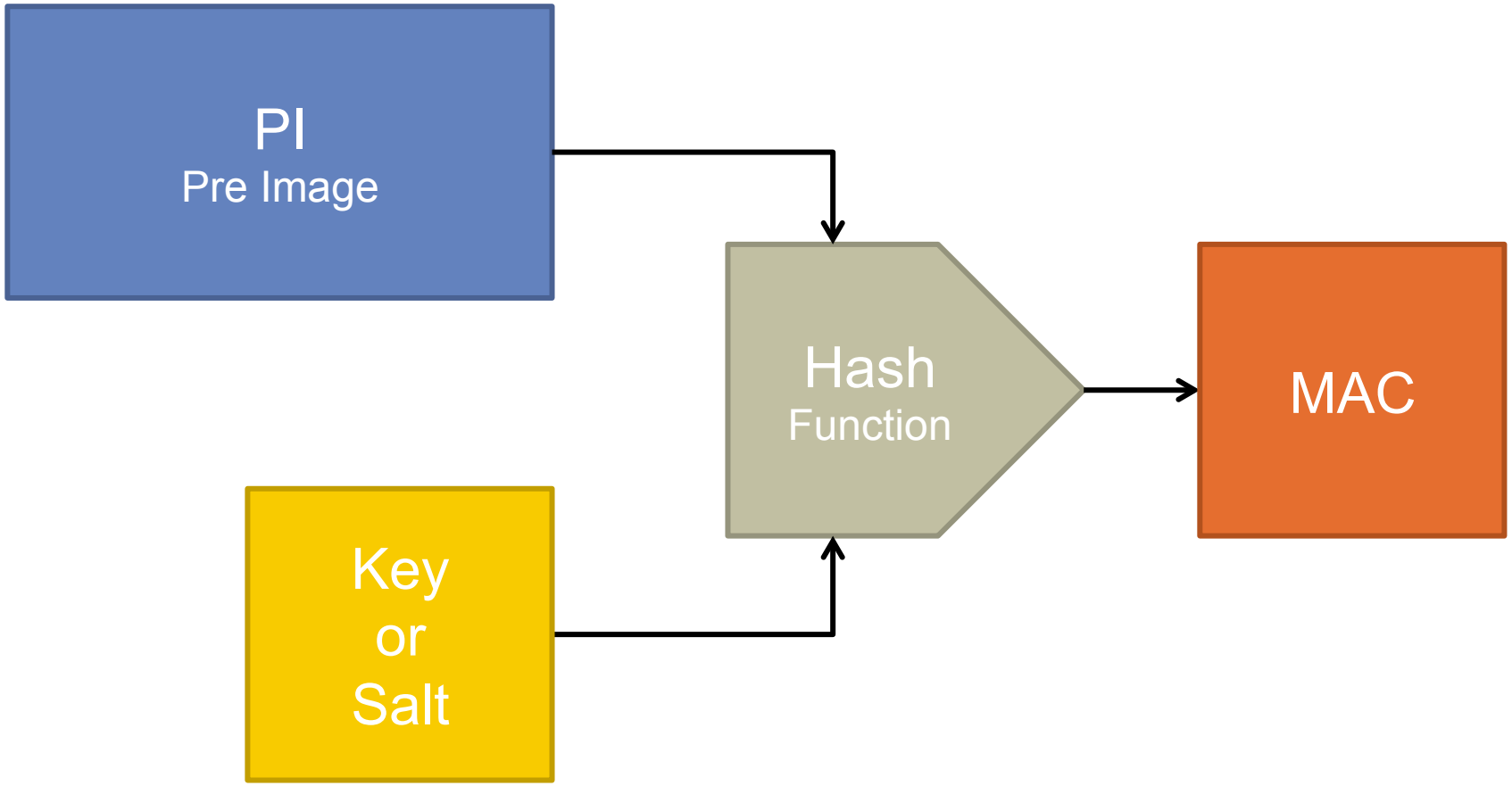
Concealing pre images while preserving relationships

- Storing passwords securely in a database
- Storing personally identifiable information as an opaque

Verifying data integrity

- Digest large files before and after transmission; if both digests match, then the file is not corrupt.
- Send the digest though a different medium then the file, if both digests match, then the file was not modified.

MESSAGE AUTHENTICATION CODES



WHEN TO USE MACS

More secure digests

- Using a unique “Salt” before digesting password

Verifying authenticity

- If the MAC of the received file matches the MAC sent with the file, then the sender can be authenticated

Verifying data integrity

- Any modification of the data by an adversary while in transit will result in an different MAC

Key Recycling

- Having two keys, a “Key Encryption Key” and “Transmission Key”. When you digest both, the resulting output is the key.

ASYMMETRIC ENCRYPTION

Two different, but mathematically linked keys (one public, one private) that secure data when used in pairs.

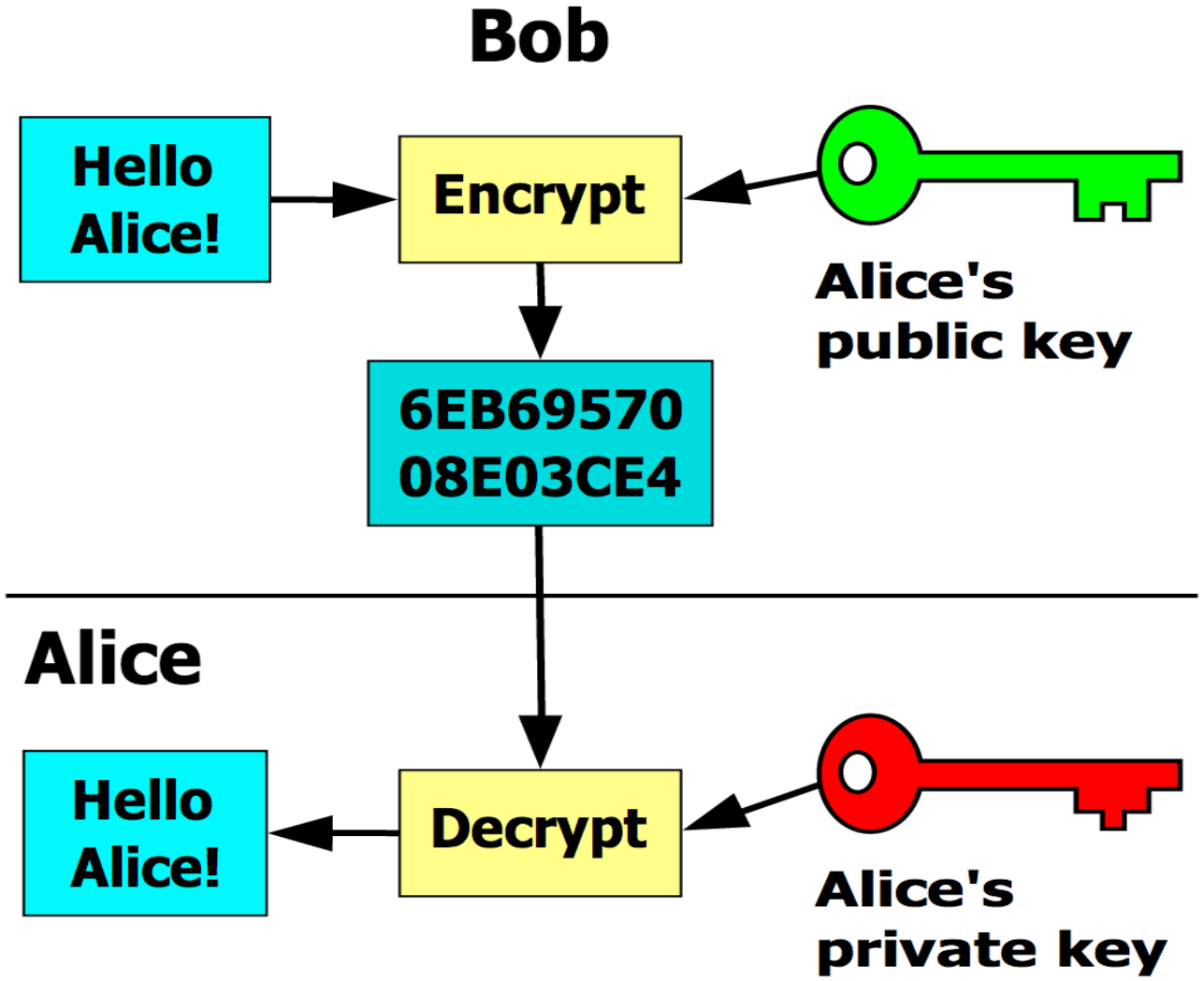
Benefits:

- **Key Exchange can be performed within a system that is compromised by an adversary.**
- **Keys can be used to encrypt or to sign data.**

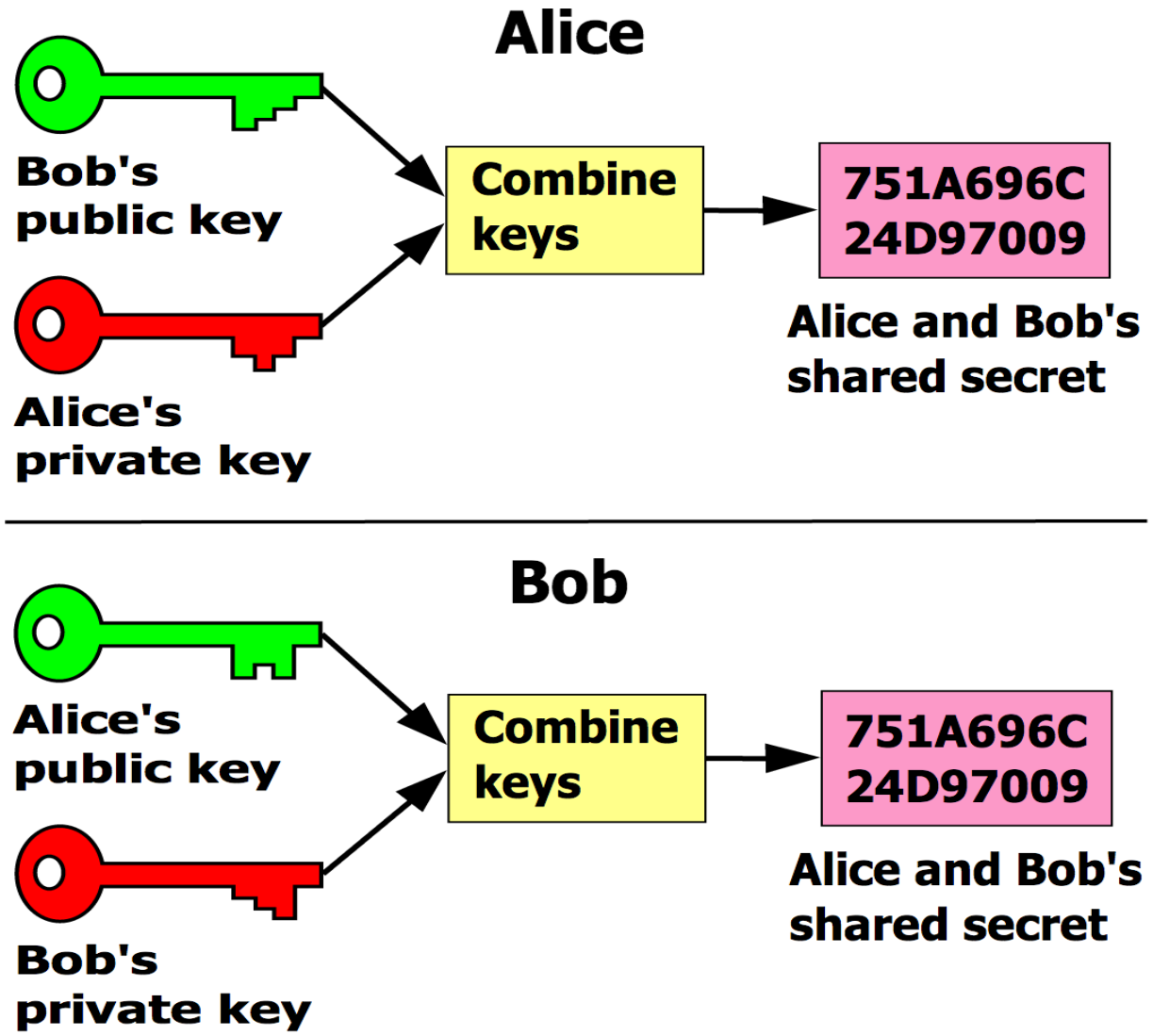
Drawbacks:

- **Key Generation is computationally expensive**
- **Encryption is slow and relies on integer factorization**
- **Not possible to use ciphering modes**

PUBLIC / PRIVATE ENCRYPTION



SHARED SECRET



RSA

The first practicable public-key cryptographic algorithm.

- **Invented by Ron Rivest, Adi Shamir, and Leonard Adleman**
- **Published in 1977, used in many cryptosystems including SSL, SSH and PGP/GnuPG**
- **Typical Key Sizes are from 1024 through 4096**
- **Relies on the mathematical difficulty of factoring the product of two prime numbers**
- **RSA have been broken with key sizes of 768 bits and lower, 768 was broken on December 12th 2009**

GENERATING RSA KEYS

1. Pick two large, random prime numbers; called *p* and *q*
   p = 11
   q = 5

2. Multiply p and q to calculate the RSA modulus; called *n*
   $n = p * q$
   $n = 11 * 5$
   $n = 55$

3. Calculate the Totient of the RSA modulus; called *z*
   $z = (p - 1) * (q - 1)$
   $z = (11 - 1) * (5 - 1)$
   $z = 10 * 4$
   $z = 40$

GENERATING RSA KEYS CONTINUED

4. Select a co-prime for z for your public key; called e
   $1 < e < z$ and $\gcd(e, z) = 1$
   $e = 7$

5. Find the multiplicative inverse of *e* mod *z* by extended Euclidian algorithm for your private key; called d
   $e * d \bmod z = 1$
   $7 * d \bmod 40 = 1$
   $d = 23$

PUBLIC AND PRIVATE KEY

The Public Key is the modulus *n* and exponent *e*

***pub* = (n, e)**

***pub* = (55, 7)**

The Private Key is the modulus *n* and exponent *d*

***pri* = (*n*, *d*)**

***pri* = (55, 23)**

p	11
q	5
n	55
z	40
e	7
d	23

You can now share your Public Key with everyone, however must ensure that your Private Key stays a closely held secret.

ENCRYPTING DATA WITH RSA

1. **Choose an integer value for the message; *m* where 0 <= *m* < *n***
   ***m* = 42**

2. **Solve for the ciphered message; *c* with the exponent e and modulus n**
   c* = *m*^e mod *n
   ***c* = 42^7 mod 55**
   ***c* = 48**

p	11
q	5
n	55
z	40
e	7
d	23
pub	(55,7)
pri	(55,23)

DECRYPTING DATA WITH RSA

1. **Solve for the plaintext message; *m* with the exponent d and modulus *n***

   m = *c*^*d* mod *n*

   m = 48^23 mod 55

   m = 42

p	11
q	5
n	55
z	40
e	7
d	23
pub	(55,7)
pri	(55,23)
m	42
c	48

HOW TO USE RSA

Transmit Keying Material

1. Randomly choose a key for a symmetric algorithm
2. Encrypt the key with an RSA public key and transmit
3. Utilize the symmetric cipher for transmitting data

Digitally Sign Documents

1. Digest the document into a smaller fingerprint
2. Encrypt the digest with the senders RSA private key
3. Recipient decrypts the signature with the senders public key
4. Match the digests of the received document with the signature

DIFFIE-HELLMAN KEY EXCHANGE

Instead of generating a key that can encrypt or sign data, generates material that allows two parties to generate a shared secret for use by symmetric algorithms.

- **Invented by Whitfield Diffie and Martin Hellman in 1976**
- **Key generation is faster, as only one large prime number is needed and the process is simpler**
- **As keys are quicker to generate, they can be once per transmission session; aiding in Forward Security**
- **By itself cannot be used to digitally sign documents, but is used by other cryptosystems that do so**

GENERATING A SHARED SECRET

1. Both parties choose a shared prime number; p and base; g
   p = 23
   g = 5

2. Both parties generate a secret integer; a and b
   a = 6
   b = 15

3. The A side calculates their public transport; A
   A = g^a mod *p*
   A = 5^6 mod 23
   A = 8

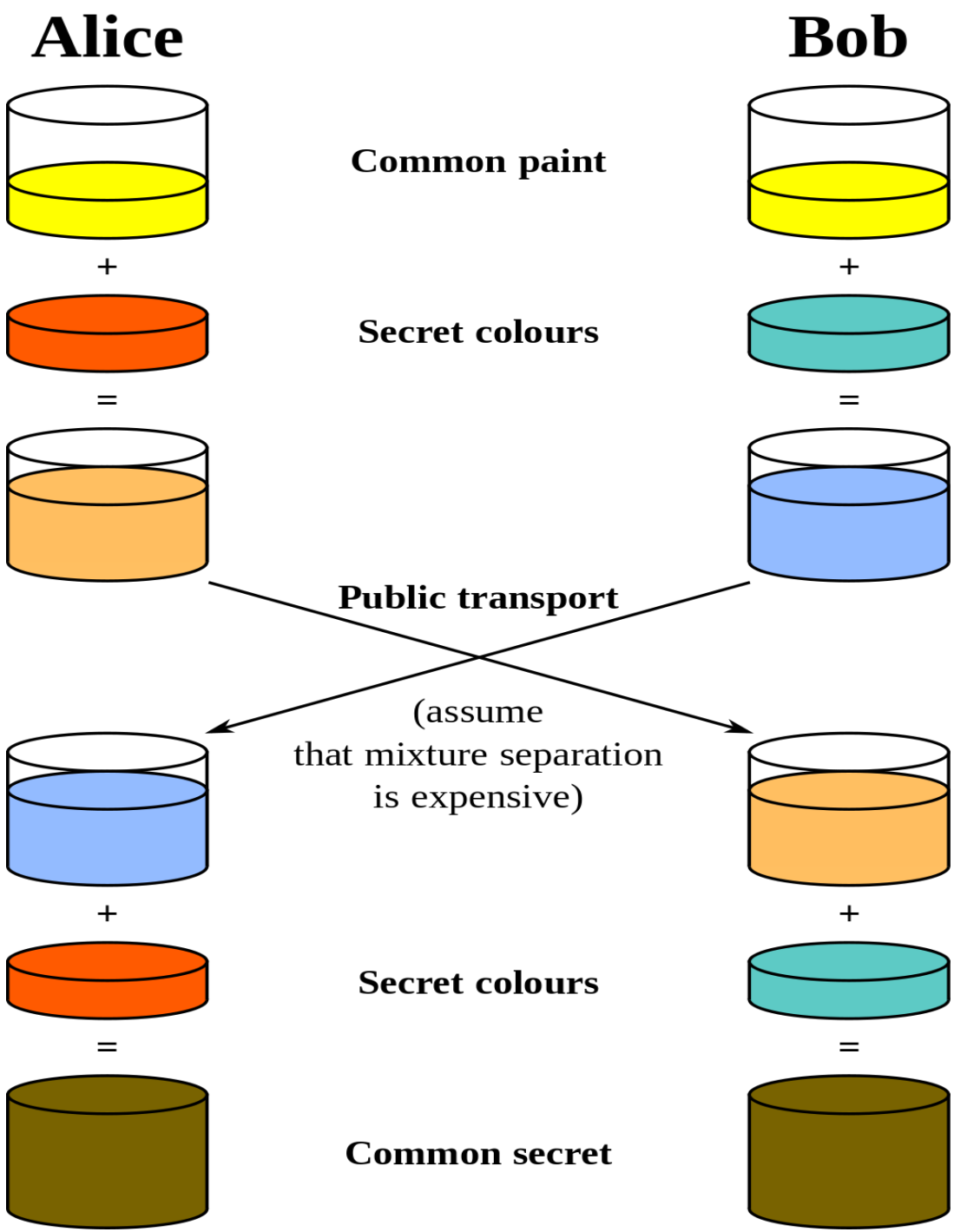
GENERATING A SHARED SECRET CONTINUED

4. The B side calculates their public transport; *B*
   B = g^b mod *p*
   B = 5^15 mod 23
   B = 19

5. The A side calculates the secret key; s
   s = *B*^a mod *p*
   s = 19^6 mod *p*
   s = 2

6. The B side calculates the same secret key; s
   s = A^b mod *p*
   s = 8^15 mod *p*
   s = 2

p	23
g	5
a	6
b	15
A	8
B	19
s	2


Alice
Bob
Common paint
+
+
Secret colours
=
=
Public transport
(assume
that mixture separation
is expensive)
+
+
Secret colours
=
=
Common secret

OTHER ASYMMETRIC ALGORITHMS

ElGamal

- Optimized for encrypting data due to being probabilistic
- A plaintext message can produce multiple ciphertexts
- Based off of the Diffie-Hellman key exchange protocol

DSA - Digital Signature Algorithm

- Optimized for securely producing digital signatures
- Is a variant of ElGamal and is used in PGP and GnuPG
- Chosen by NIST as the Digital Signature Standard

LAST BUT NOT LEAST, SOMETHING RANDOM

Creating proper cryptographic keys high levels of entropy

1. If there is a pattern used to generate a key, then the pattern, and thus the key, can be guessed
2. If there are a limited number of keys, the surface area for an attack is smaller, and the impact is larger
3. If an attacker controls the random generator, then the attacker controls key generation
4. Randomness is very hard to obtain with computers, so we often we settle with pseudo-randomness
5. Finding pseudo-random prime numbers is very costly

GETTING RANDOM DATA

`/dev/random`

High Quality and High Cost

Will block if entropy pool is empty

`/dev/urandom`

Medium Quality and Low Cost

Will digest new data if pool is empty

SUMMARY

The Basics: what cryptography is at a fundamental level

Symmetric Encryption: fast and well understood algorithms

Digest Functions: great for fingerprints and opaque tokens

Asymmetric Encryption: important part of any crypto system

Randomness: hard to generate but very important to have

CONDENSED SUMMARY

Understanding

your tools

lets you make make more

secure systems

QUESTIONS

Reactive.IO

Robert Sosinski

robert.sosinski@reactive.io