CS443 — Cryptography

Note Title

1/16/2013

Cryptography is old:

Caesar Cyphers:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Q W E R T Y U I O P A S D F G H J K L Z X C V B N M

plain

EXAMPLE → TBQ ...

cipher

KQFRGD → RANDOM

Q:

How would you attack a Caesar cipher?

- Pattern analysis
- Frequency analysis
- Brute force

Today:

1. Symmetric encryption
   - ↳ secret shared key
   - DES, AES,

2. Assymetric encryption - early 70's
   - public key crypto
   - Diffie-Hellman, RSA, ...

3 Goals in Cryptography:

1. Confusion: Obfuscate the relationship between the plaintext + ciphertext.
2. Diffusion: Dissipate the redundancy in plaintext by spreading it over the ciphertext.
3. Secrecy only in the key.

A bad example: CSS

In 1996, DVDs began using Content Scrambling System to protect DVDs from unauthorized copying.

Secrecy depended on users not knowing the handshake protocol and where in memory keys were stored.

In 1999, a group in Norway reverse-engineered it and made DeCSS, a tool to break this encryption.

Even worse:

POLITICS : LAW

DVD Lawyers Make Secret Public

Declan McCullagh 01.26.00

Lawyers representing the DVD industry got caught in an embarrassing gaffe when they filed a lawsuit and accidentally publicized the computer code they wanted to keep secret.

The DVD Copy Control Association included its "trade secret" source code in court documents, but forgot to ask the judge to seal them from public scrutiny.

Whoops.

In a hastily arranged hearing Wednesday morning, DVD CCA lawyers asked Santa Clara Superior Court Judge William J. Elfving to correct their oversight, and he agreed to keep the document confidential.

It may be a little late. The document is dated 13 January and is widely available on the Web. The owner of one site that placed the 140KB declaration online says over 21,000 people have downloaded it so far.

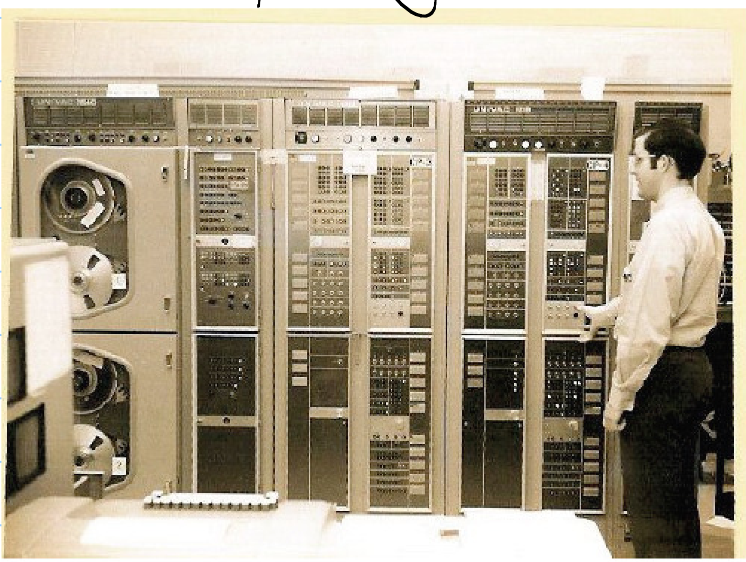
The 11KB "CSSscramble" source code, part of the larger declaration of DVD CCA president John Hoy, cannot be readily compiled into a DVD viewer or copier.

But if it had not been released online last October, the DVD encryption scheme likely would not have been penetrated.

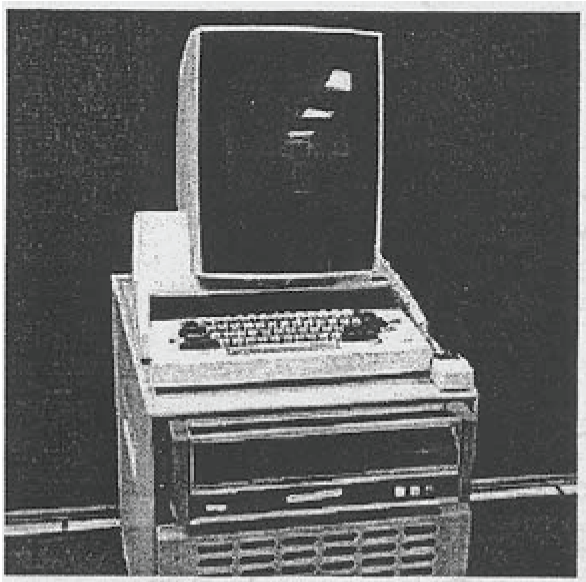
Elfving granted an injunction last Friday, ordering 21 defendants to stop posting DeCSS software -- which allows compressed video images to be copied from a DVD disc onto a hard drive -- on their Web sites.

Good example: DES - Data Encryption Standard

In 1972, NBS (now NIST - National Institute of Standards & Technology) issued a request for a standard cryptographic protocol.



UNIVAC Type 1218 Computer



Xerox Alto

Design criteria:

- high level of security
- completely specified
- easy to understand
- adaptable
- economically implementable
- efficient
- able to be validated
- exportable.

Result: Complete Failure

DES

- In 1974, they try again.
- IBM produces "Lucifer"; with some edits, this becomes DES, officially adopted in 1977.
- Encrypts 64 bits of plaintext using a key of 64 bits

Essential element: XOR (permuting)

DES steps

1) Perform an initial permutation (IP)
2) Perform initial key transformation
3) Perform 16 identical rounds of key-dependent computation using a function f.
4) Perform inverse initial permutation.

Visually:

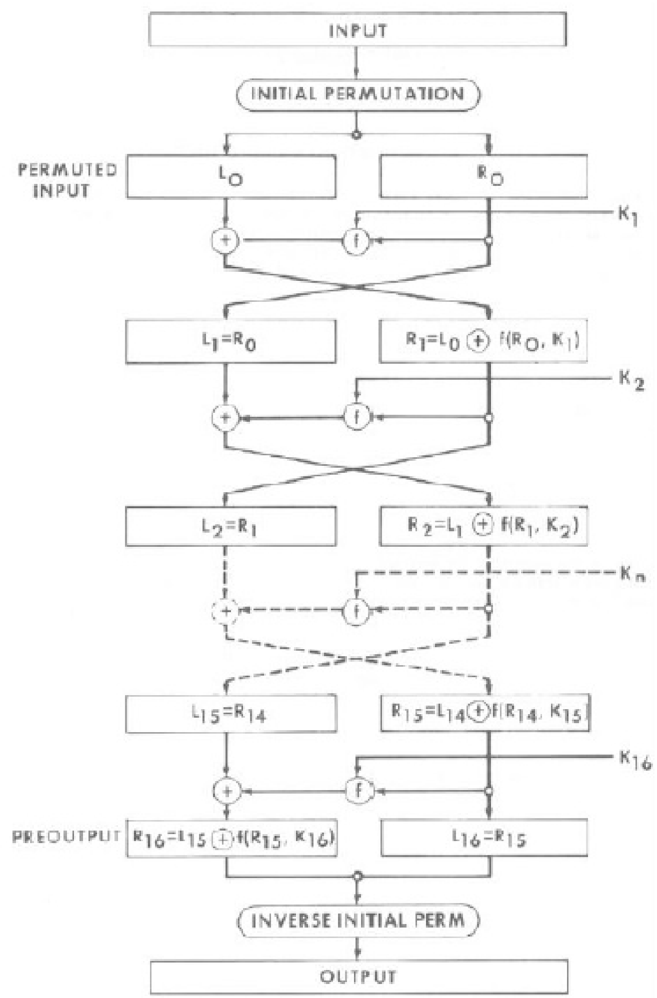


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

Step 1: IP

The 64 input bits are permuted using the following initial permutation:

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Step 2: Key xform

Reduce 64-bit key to 56-bit key by ignoring every 8^{th} bit.
(Yes, really.)

Then permute:

PC-1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

Step 3: 16 Rounds

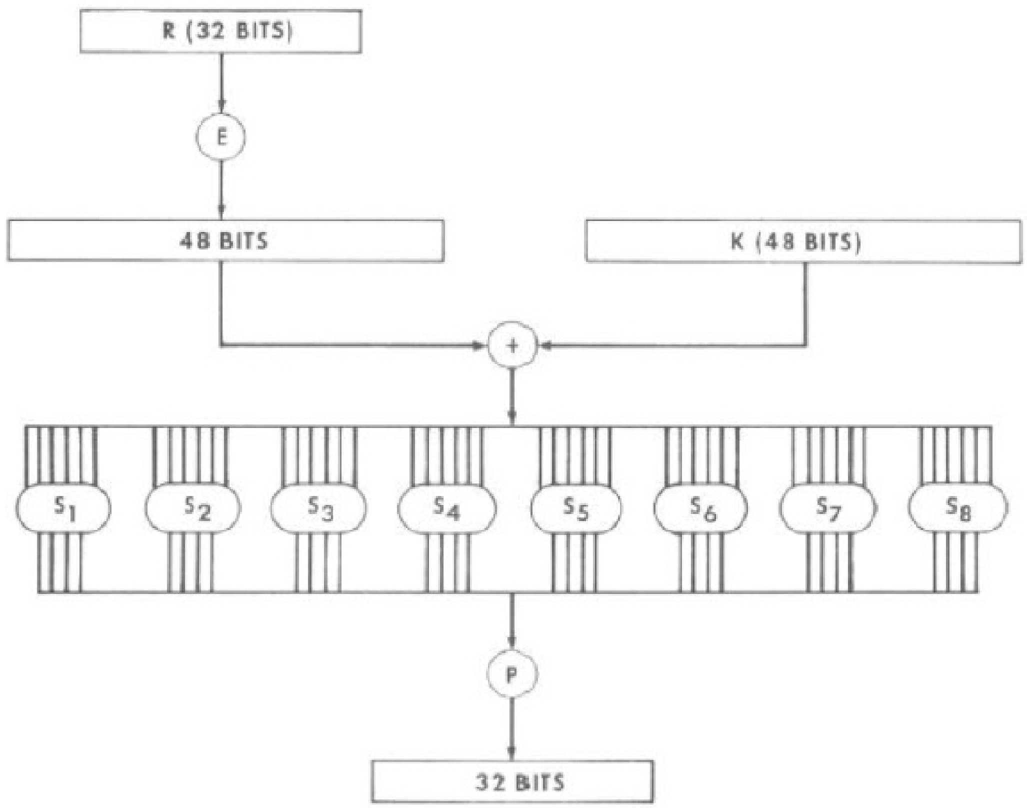


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

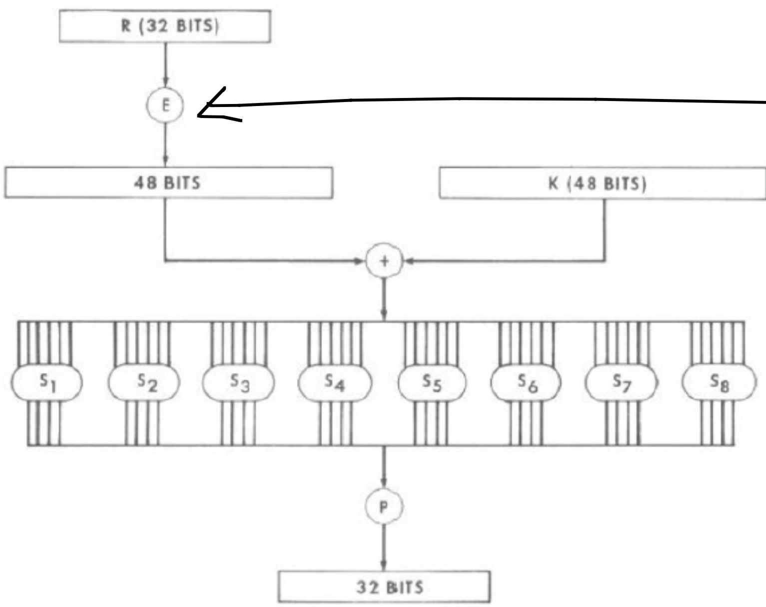


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

E:
32-bits expanded to 48 by duplicating half of the bits

Called expansion permutation.

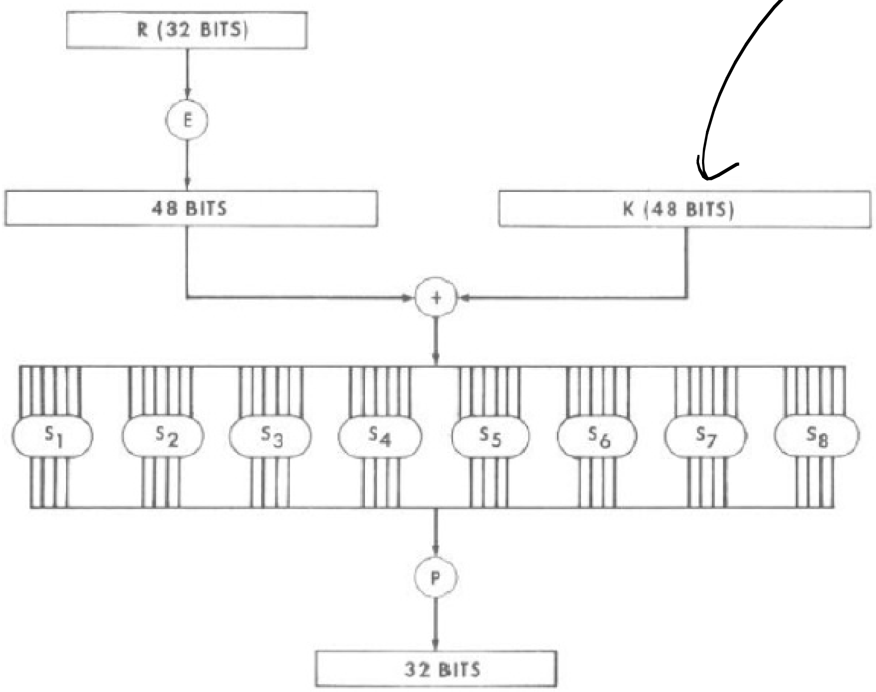


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

Get a subkey K.

There are 16 of these, each based on secret key.

(Calculation is done according to set algorithm.)

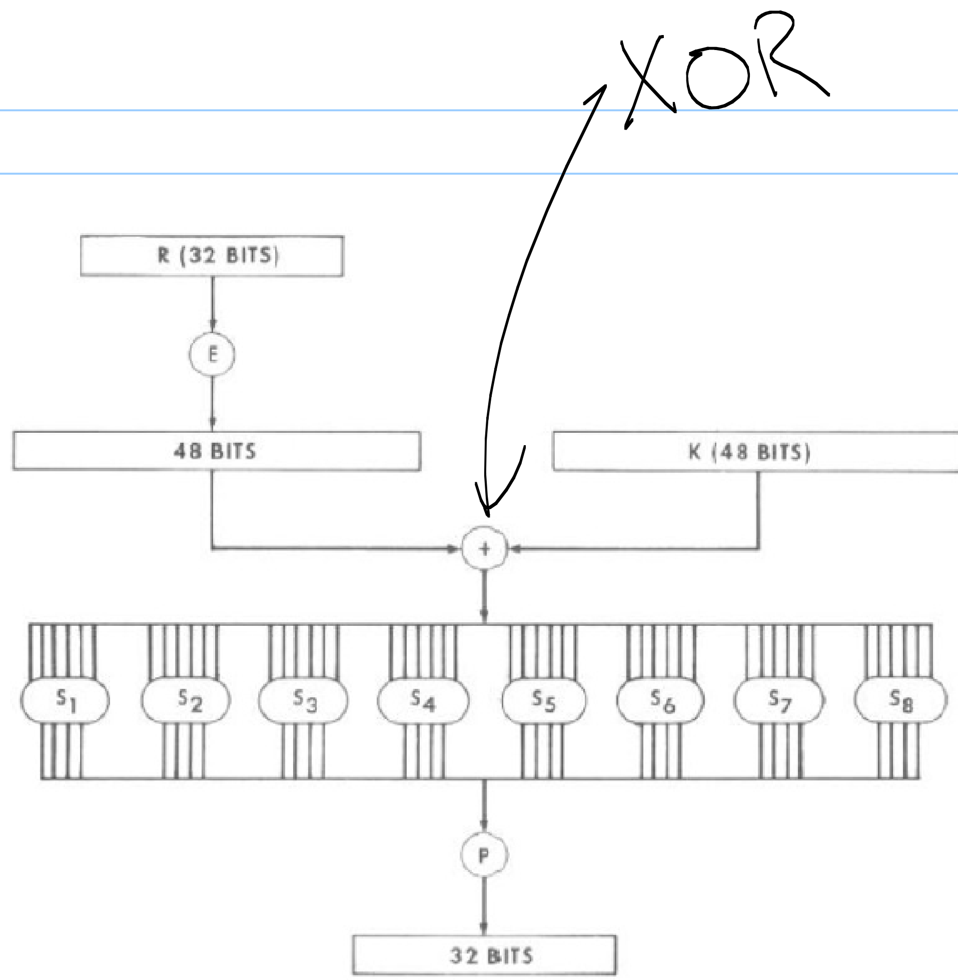


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

The Key:

"S-box substitution"

Block divided into 8 6-bit pieces.

Processed by the 8 S-boxes: non-linear transformation from 6 to 4 bits, in look-up table.

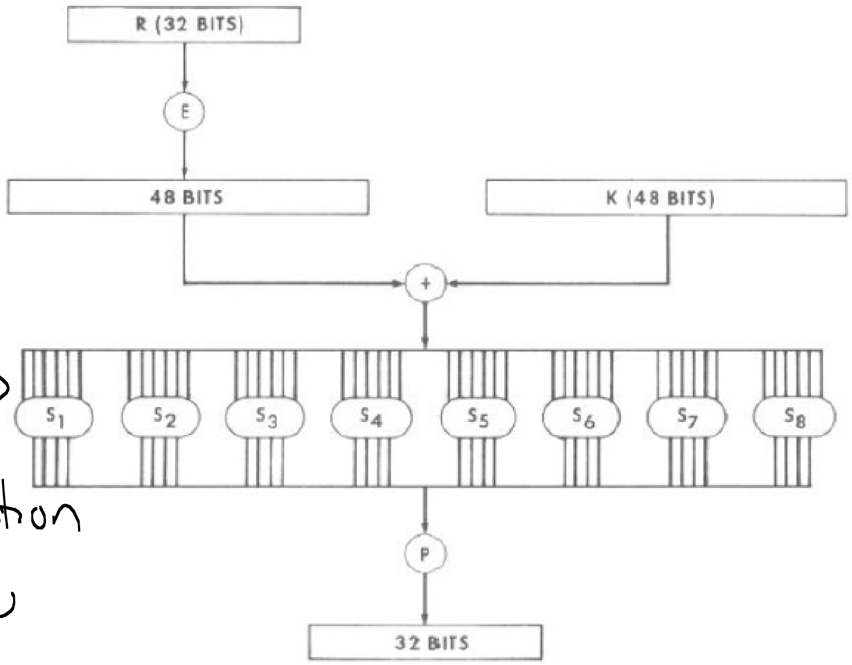


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

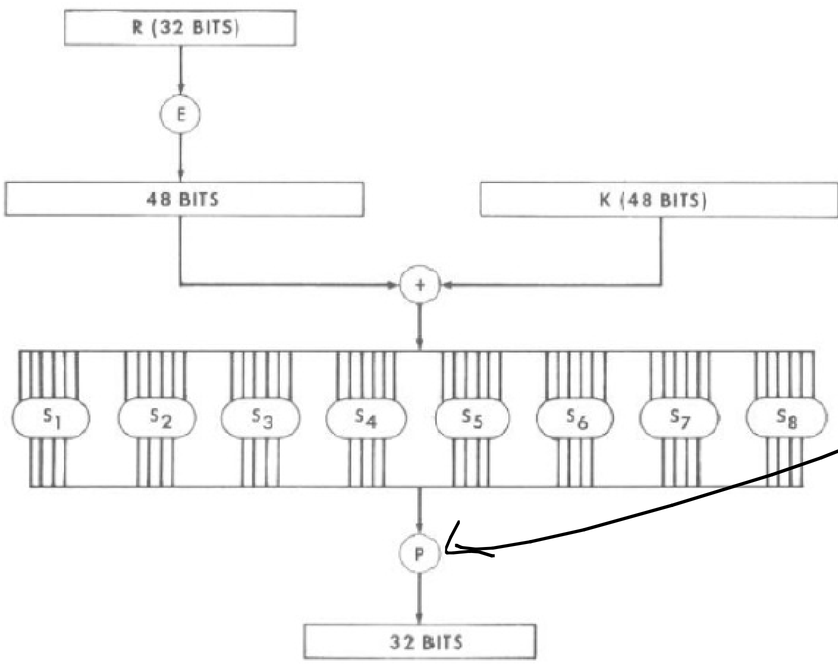


Image taken from FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, FIPS PUB 46-3

Finally, the 'P-box' permutation.

Permutes outputs to "mix" the outputs of the 8 boxes together.

(This is "diffusion".)

P

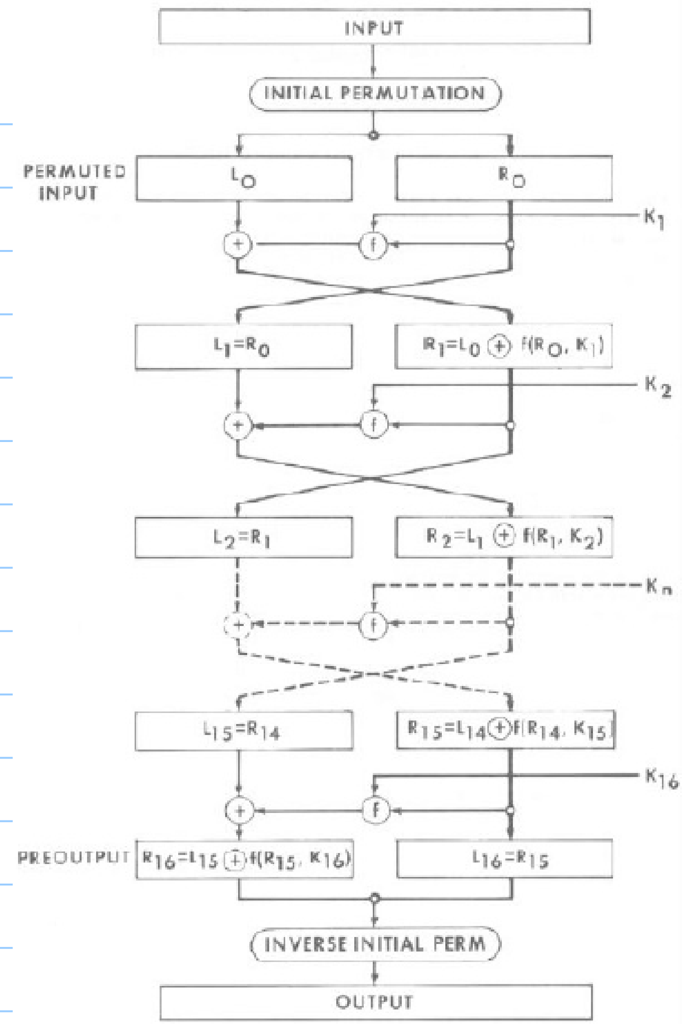
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

After one round, XOR this new right half with the left half.

XOR is new right half.

Old right half is new left half.

Repeat 16 times:



Step 4: Inverse IP

IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

DES:

Looks simple from the outside:

```
Data:    0x0DEA.D0C0.FFEE.00FF
Key:     0x0123.8104.75AA.F41E

Result:  0x0EEF.E446.0E9B.19FF
```

or:

```
Data:    0x0EEF.E446.0E9B.19FF
Key:     0x0123.8104.75AA.F41E

Result:  0x0DEA.D0C0.FFEE.00FF
```

Why does it work??

```
Data:    0x0EEF.E446.0E9B.19FF
Key:     0x0123.8104.75AA.F41E

Result:  0x0DEA.D0C0.FFEE.00FF
```

```
Data:    0x0EEF.E446.0E9B.19FF
Key:     0x0023.8104.75AA.F41E

Result:  0x1AE0.0386.B2FF.1D94
```

The conspiracy theory : S-boxes

The S-boxes make DES unusually resistent to an attack known as differential cryptanalysis, a technique discovered in 1990.

IBM revealed later that they knew of this technique, but we asked by the NSA to keep it quiet.

Also, many contend that the NSA changed the S-boxes so they could break all DES traffic.

How did DES break?

1990: Biham & Shamir develope (publically) differential cryptanalysis.

Consider ciphertext pairs: pairs of ciphertexts where plaintexts have particular differences.

1992: Broke a ciphertext using 2^{47} pairs ← HUGE

1994: Matsui used linear cryptanalysis to do better.

Took 50 days and 12 workstations, as well as 2^{43} known plaintexts!

In July 1998, the EFF built a machine for $250,000 that performed a brute force attack in 56 hours.

(So computational speed broke DES.)

In 2002, DES was officially retired.

Triple DES: 3DES

- Last ditch effort to save it.
- Repeat DES 3 times with different keys - total of 168 bits.

Actually - still secure!

Drawback: SLOW

AES: Advanced Encryption Standard

History:

- In 1996, NIST issued a call to replace 3DES.
- In 1998, 15 algorithms were submitted.
- NIST spent years having open tests done on all submissions.
- The winner was Rijndael, developed by 2 Belgian cryptographers.
- Officially approved in 2001.

AES: Details

- Block length is 128 bits, & keys are 128, 192, or 256 bits.

- Works in a finite field Z_{256}.

  (Huh?)

Definitions

AES operates in a finite field.

Dfn: A group is a set equipped with an operation (such as addition or multiplication).

The operation must:

- be associative $(ab)c = a(bc)$
- have (unity) identity element $a \cdot 1 = a$, $b + 0 = b$
- provide inverses $a \cdot \frac{1}{a} = 1$, $a + (-a) = 0$

Ex: $\mathbb{R}$ is a group under addition:

- $x + (y + z) = (x + y) + z$
- $x + 0 = x$
- $x + (-x) = 0$

Identity element: 0
(or unity)

What about multiplication? ~~Yes~~ - for $\mathbb{R}$
no - b/c of 0

0?

What about $\mathbb{Z}$?

Ex: Is $\mathbb{Z}$ a group?

addition: yes

multiplication: no (inverses)

Abelian group:

A group where commutativity also holds:

$$a \times b = b \times a$$

Ring: A set R with **both** + and ×, where

- R is an abelian group under +
- R is commutative & associative under ×
- multiplicative identity ≠ additive identity
- connection between + & ×:

$$x(y+z) = xy + xz$$

Examples: • $\mathbb{Z}$, the integers

• integers modulo any n

eg $\mathbb{Z}_3 = \{0, 1, 2\}$

$\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$

Note:

Rings may not have multiplicative inverses!

Example: $\mathbb{Z}_3 = \{0, 1, 2\}$

$\mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$

what is mult inverse of 5?

yes $\rightarrow$ 5

what about 3?

Field: A ring where we have multiplicative inverses also.

Called a finite field if finite # of elements.

Ex: $\mathbb{Z}_3 = \{0, 1, 2\}$

$\mathbb{Z}_n$

Note: $\mathbb{Z}_p$, where p is ~~prime~~ or $p = q^k$ & q is prime.

Side note: Why do we care?

1. Why not use $\mathbb{R}$?

   infinite!

   numeric precision

2. Why finite?

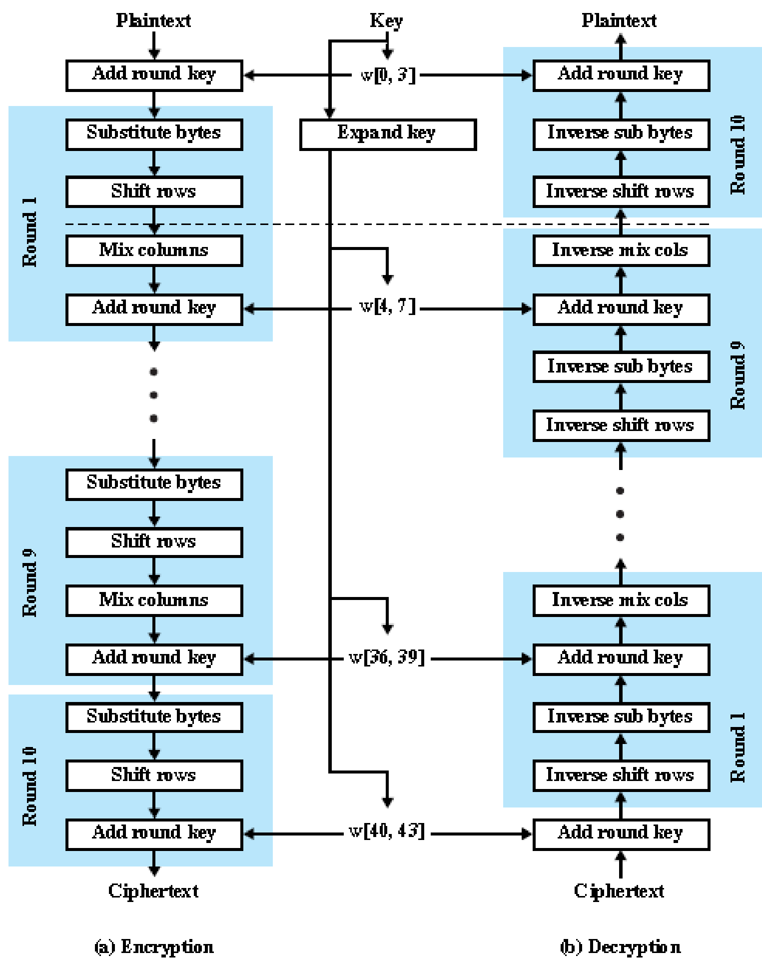
3. Why good for cryptography?

AES: Details

Essentially, 4 operations:
(performed repeatedly)

1) Substitute bytes
2) Permute
3) Mix Columns
4) Add round key (an XOR with part of secret key - changes each round)

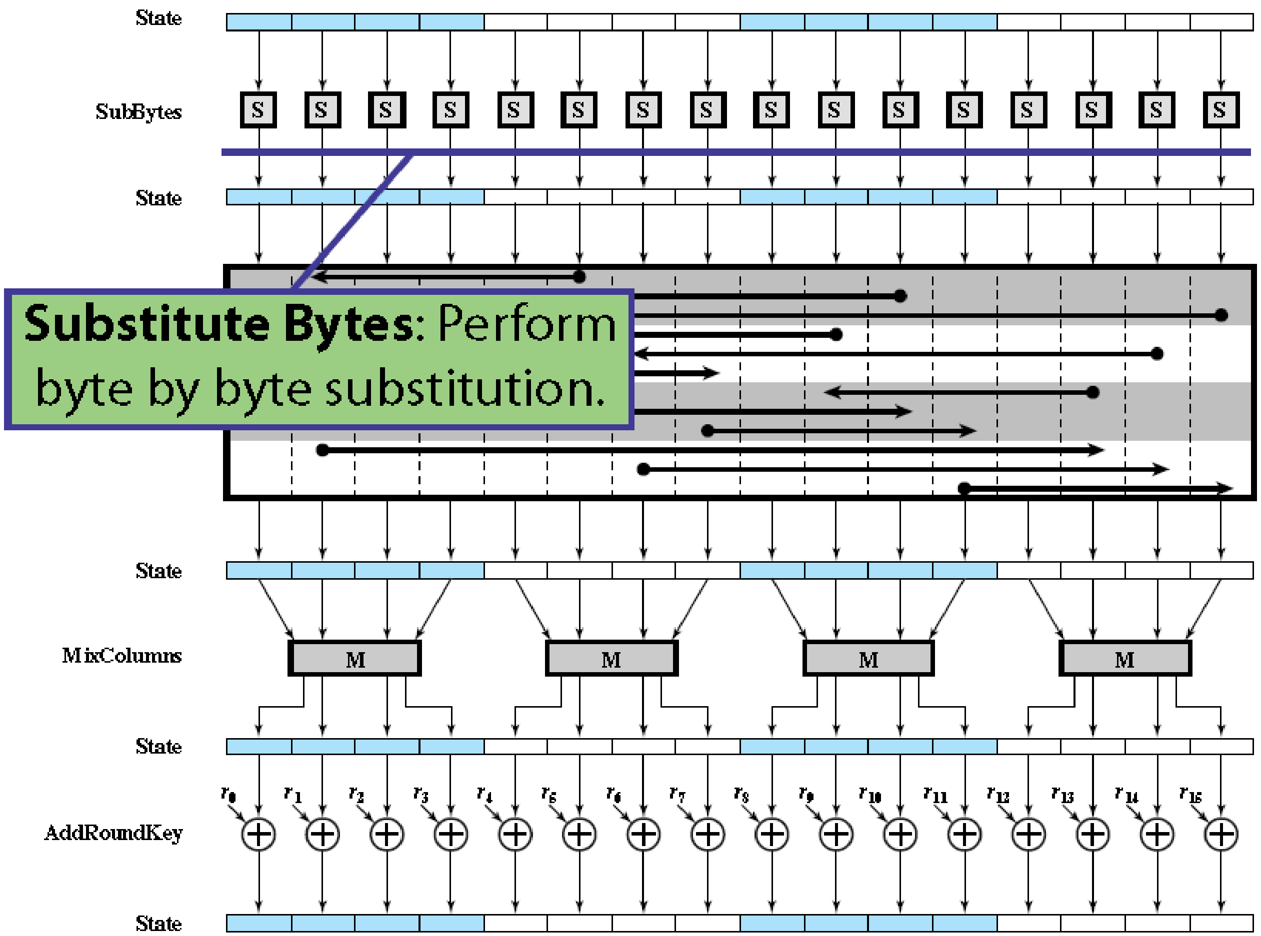
Details:



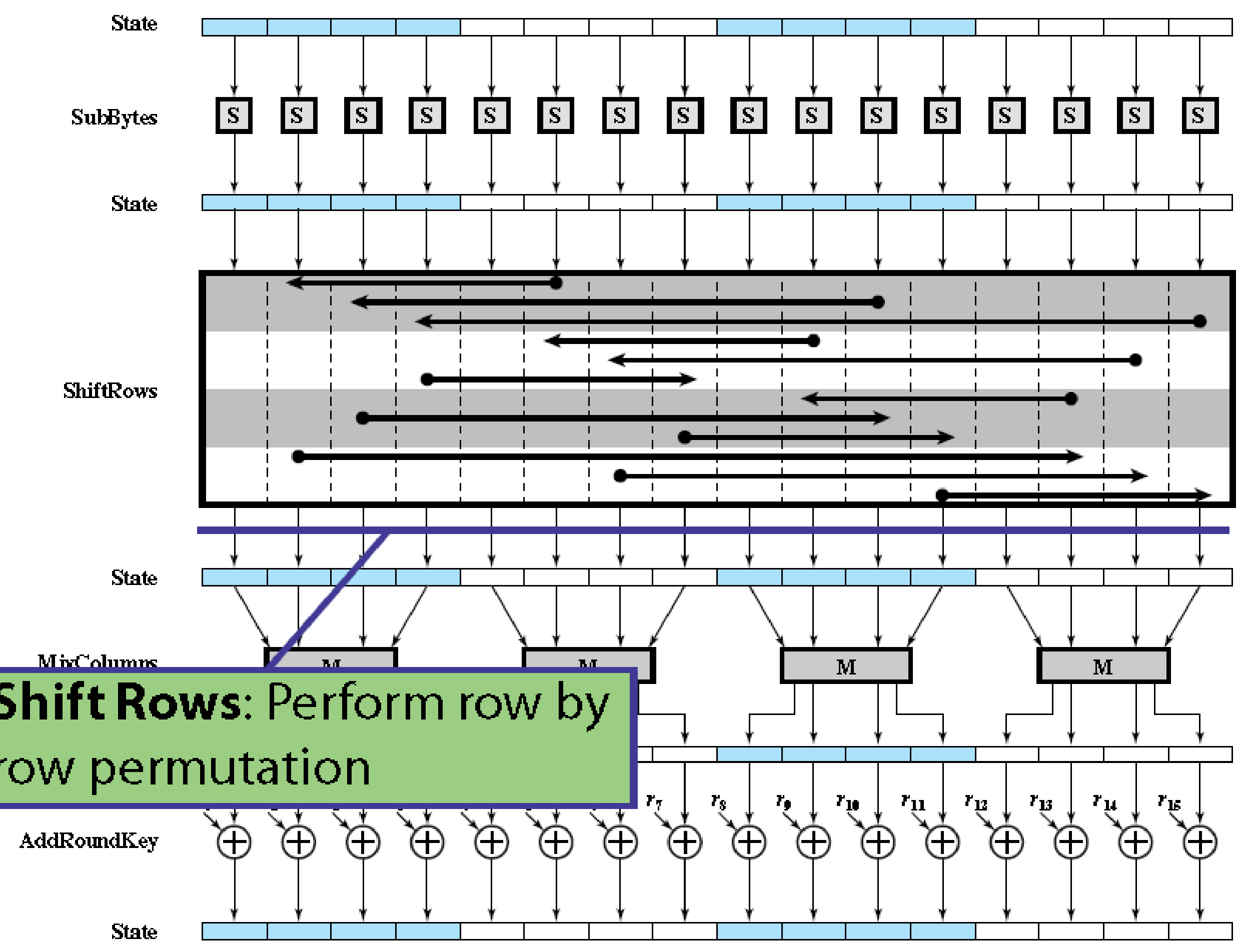
AES Algorithm

128-bits of message at a time are operated upon in 10 rounds of four stages.

Step 1 (in a round):

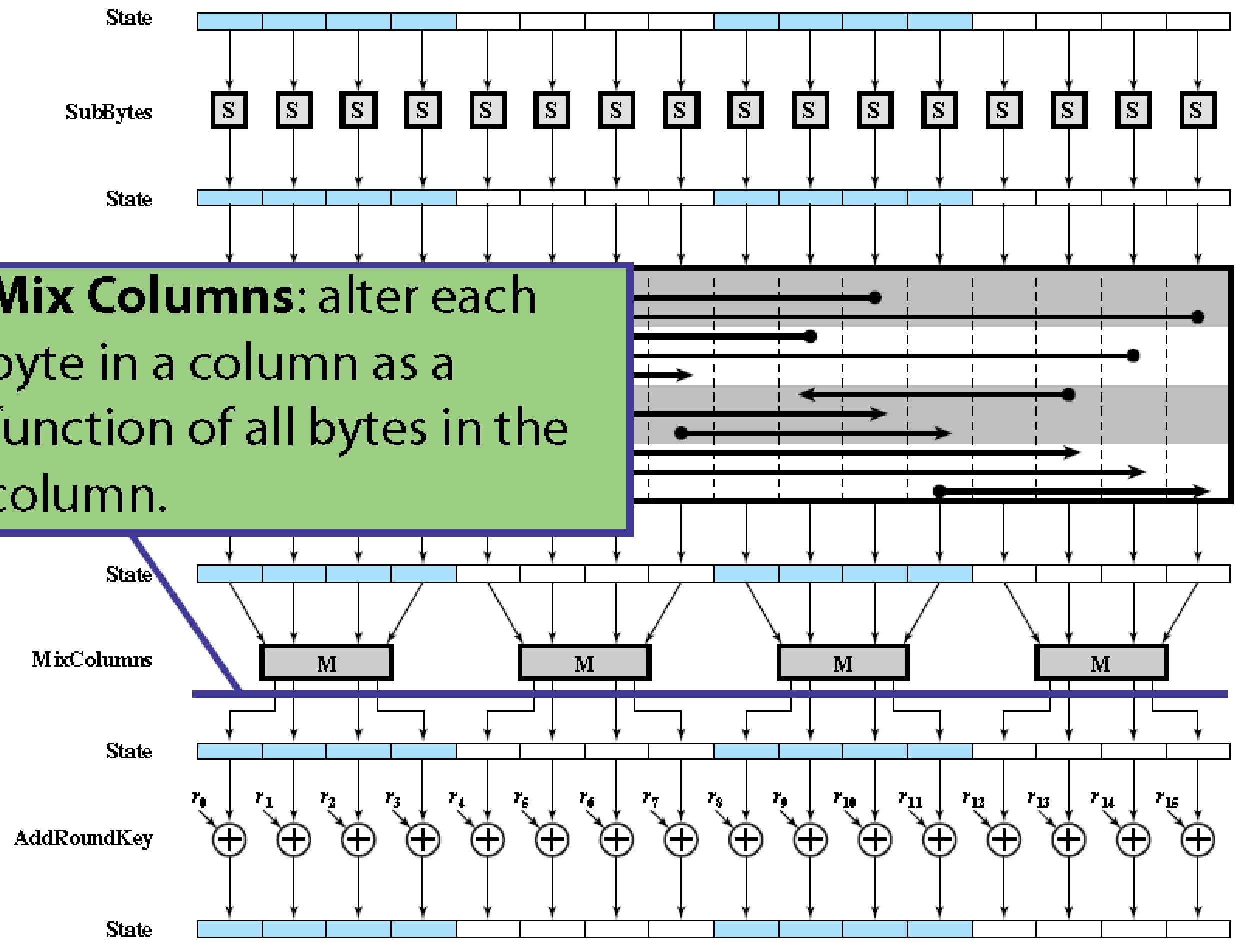


Step 2:



Shift Rows: Perform row by row permutation

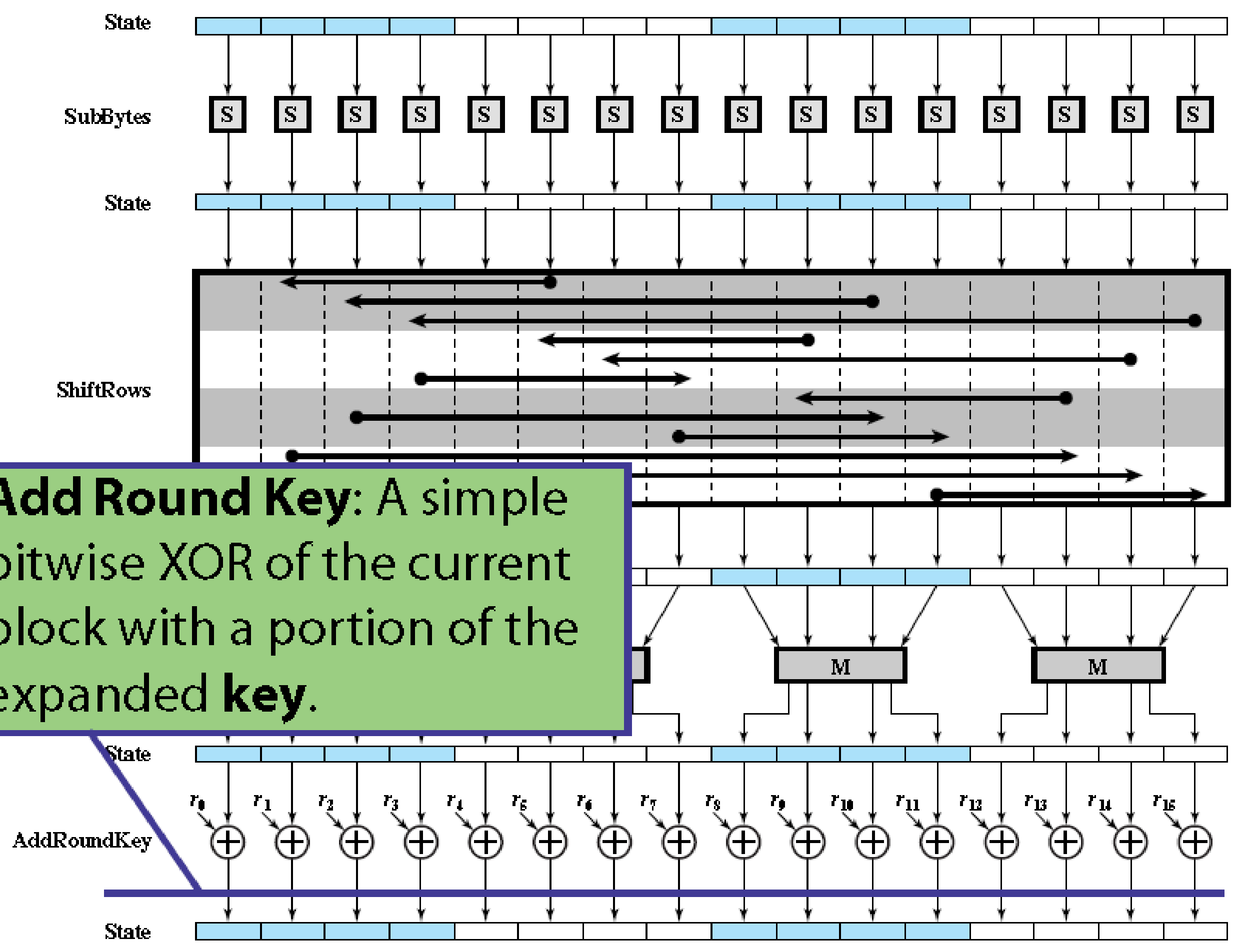
Step 3:



Mix Columns: alter each byte in a column as a function of all bytes in the column.

Step 4:

(in 20)



Add Round Key: A simple bitwise XOR of the current block with a portion of the expanded **key**.