Cryptography

Cryptography is a technique of securing information and communications through use of codes. Thus preventing an unauthorized access to information. The prefix "crypt" means "hidden" and suffix graphy means "writing".

Cryptography Types

1) Symmetric Key Cryptography:

The sender and receiver of message use a single common key to encrypt and decrypt messages.

2) Asymmetric Key Cryptography:

A pair of keys is used to encrypt and decrypt information. A public key is used for encryption and a private key is used for decryption. Even if the public key is known by every one the intended receiver can only decode it because he alone knows the private key.

3) Hash Functions:

There is no usage of any key in this algorithm. A hash value with fixed length is calculated as per the plain text which makes it impossible for contents of plain text to be recovered.

Classification is based on following points:

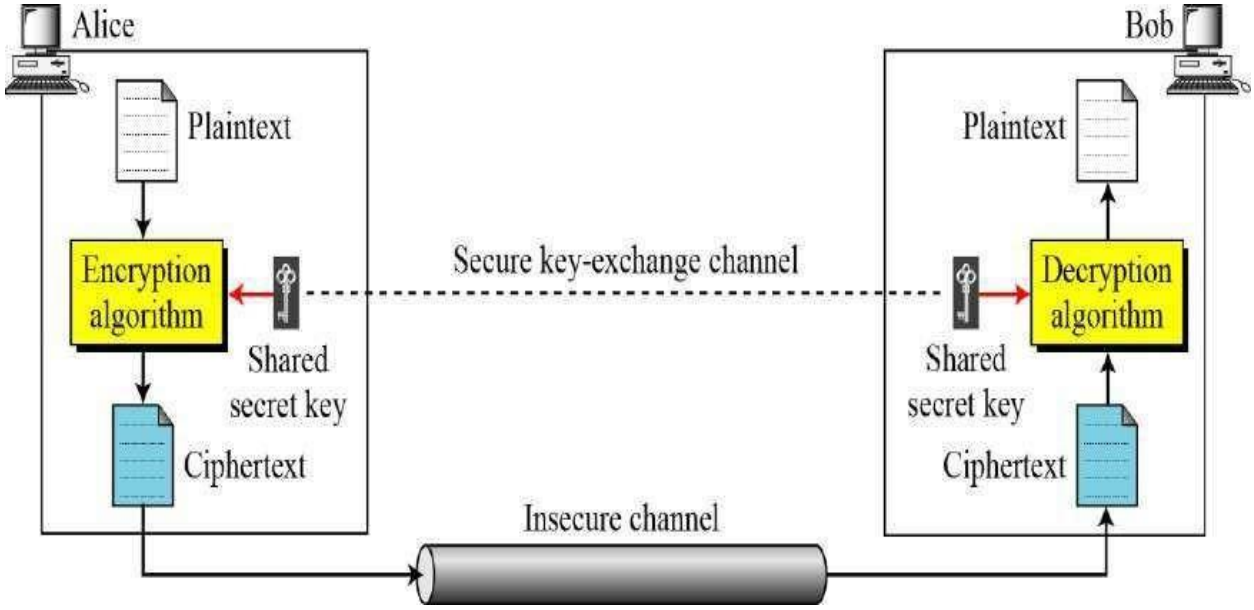
- Number of keys used
  - Hash functions: no key
  - Secret key cryptography: one key
  - Public key cryptography: two keys - public, private
- Type of encryption operations used
  - substitution / transposition / product
- Way in which plaintext is processed
  - block / stream

Steganography

- An alternative to encryption
- Hides existence of message
  - using only a subset of letters/words in a longer message marked in some way
  - using invisible ink
  - hiding in graphic image or sound file
- Has drawbacks
  - high overhead to hide relatively few info bits

Introduction to Symmetric Encryption

Alice: $C = E_k(P)$ **Bob:** $P_1 = D_k(C) = D_k(E_k(P)) = P$



Encryption: $C = E_k(P)$ Decryption: $P = D_k(C)$

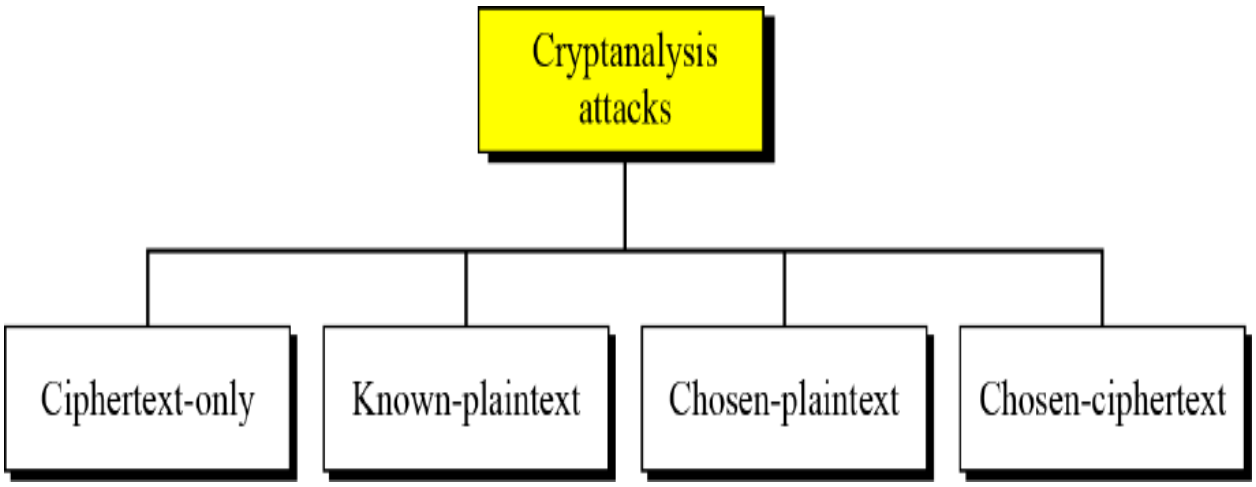
In which, $D_k(E_k(x)) = E_k(D_k(x)) = x$

Kerckhoff's Principle

It states that the security of a cryptographic system shouldn't rely on the secrecy of the algorithm. Instead, it should be based on the secrecy of the cryptographic key.

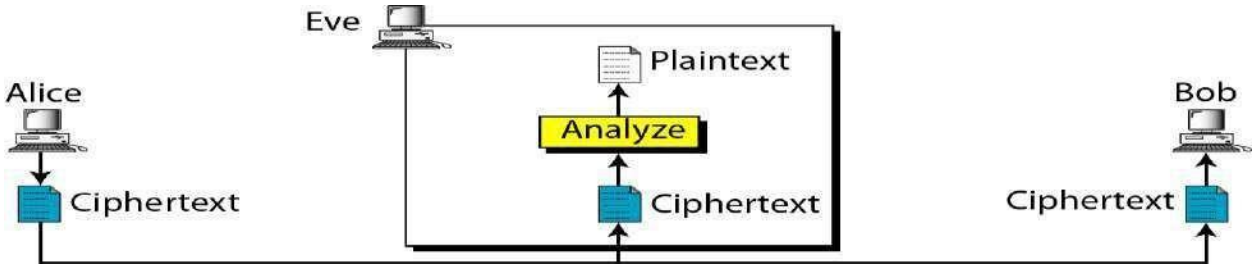
Cryptanalysis

As cryptography is the science and art of creating secret codes, cryptanalysis is the science and art of breaking those codes.



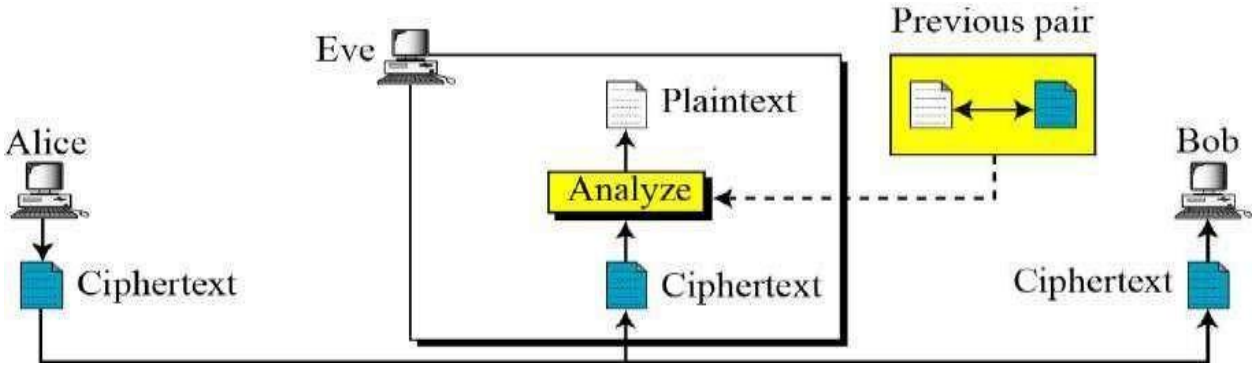
Cipher text Only attack

-Brute-Force attack & Statistical attack

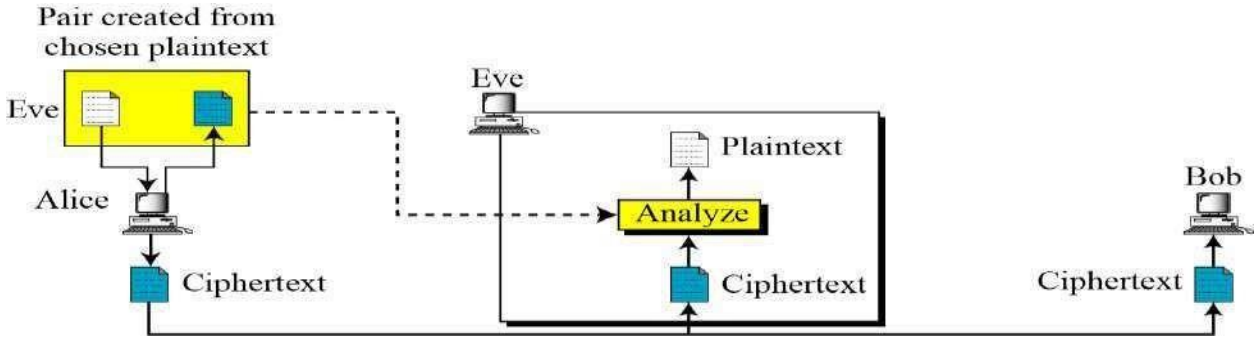


Known Plain text Only attack

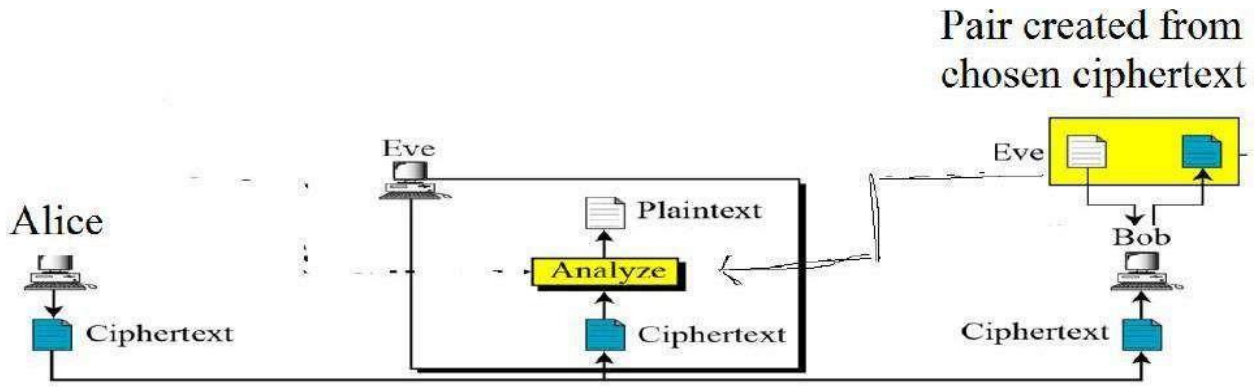
-Tries to find next secret message



Chosen Plain text attack

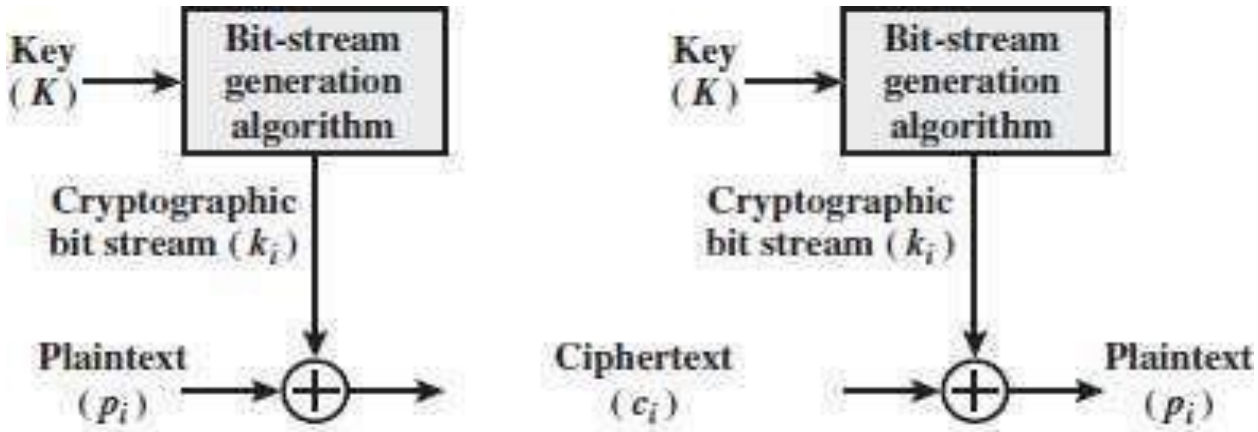


Chosen Cipher text attack



Stream ciphers

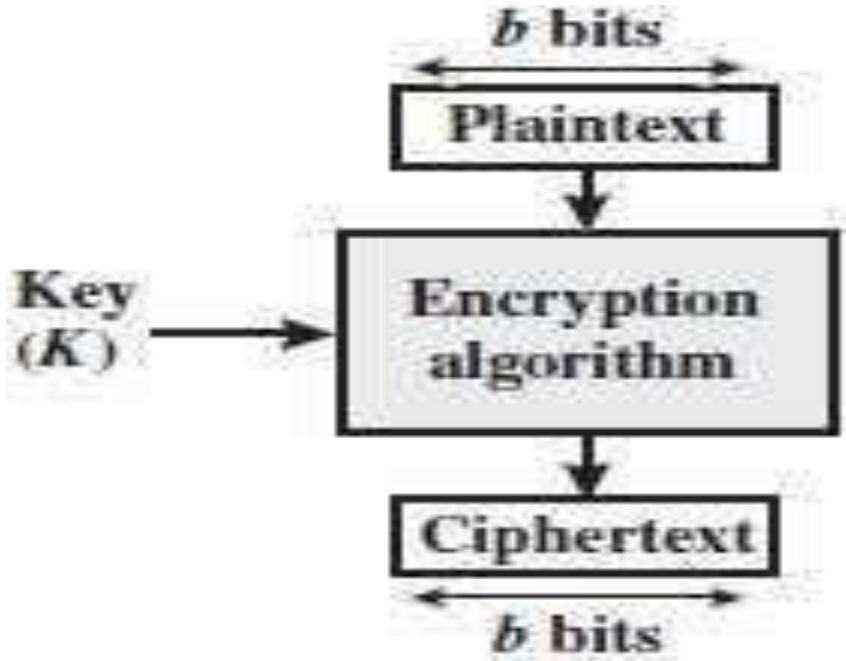
A stream cipher is one that encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the **auto keyed Vigenère cipher and the Vernam cipher.**



(a) Stream cipher using algorithmic bit-stream generator

Block ciphers

A **block cipher** is one in which a block of plain text is treated as a whole and used to produce a cipher text block of equal length. Typically, a block size of 64 or 128 bits is used



(b) Block cipher

Mathematics of Symmetric Key Cryptography

AlgebraicStructures:

Cryptography requires set of integers and specific operations that are defined for those sets. The combination of the set and the operations that are applied to the elements of the set is called an **algebraic structure**.

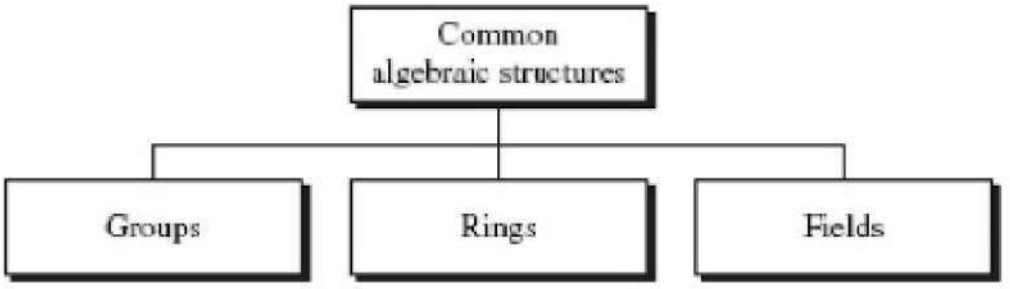
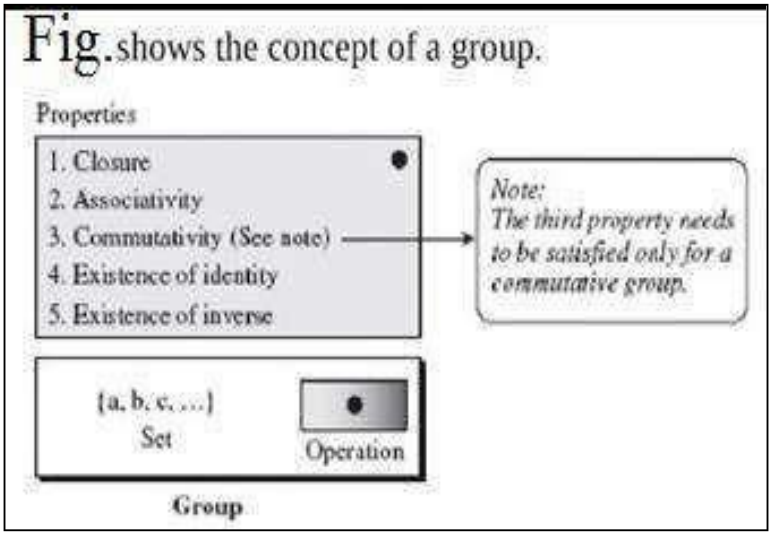


Fig. *Common algebraic structures*

1. Groups

A **Group(G)** is a set elements with a binary operation "•" usually Addition or multiplication that satisfies four properties(Axioms).

- A **Commutative Group**, also called an **abelian group**, is a group in which the operator satisfies the four properties for groups plus an extra property, commutativity.
- Closure Property: if a and b are elements of G, then c=a•b is also an element of G.
- Associatively Property: if a,b, and c are elements of "G, then (a•b)•c = a•(b•c).
- Existence of Identity Property: For all a in G, there exists an element e, called the identity element, such that e • a = a • e = a
- Existence of Inverse Property: For each a in G, there exists an element a^1, called the inverse of a, such that a • a^1 = a^1• a = e



- Commutativity Property: For all a and b in G, **we have a•b=b•a.**

EXAMPLE1

The set of residue integers with the addition operator, G=<Z_n,+>, is a commutative group

1. Closure is satisfied. The result of adding two integers in Z_n is another integer in Zn
2. Associativity is satisfied. The result of 4+(3+2) is same as (4+3)+2
3. Commutative is satisfied. We have 3+4=4+3
4. The identity element is 0. We have 3+0=0+3=3
5. Every element has an additive inverse. The inverse of 3 is 7($3+7 \bmod 10=0 \bmod 10 \text{ in } Z_{10}$) and Inverse of 7 is 3($7+3 \bmod 10=0 \bmod 10 \text{ in } Z_{10}$), so inverse property satisfied

EXAMPLE2

The set Z_n * with multiplication operator, G=<Z_n*,x>, is also an abelian group. We can perform multiplication and divisions on the elements. Identity element as 1.

Finite Group: A group is called a finite group if the set has a finite number of elements; otherwise, it is an infinite group.

Order of a Group: The order of group, **|G|**, is the number of elementsinthegroup.Ifthegroupisnot finite,itsorderisinfinite;ifthegroupisfinite,theorderisfinite.

Subgroups:A subset **H** of a group **G** isa subgroup of G if H itselfisa group withrespect tothe operation on**G**.Inotherwords,if **G=<S,•>**isagroup, **H=<T,•>**isagroup underthesameoperation,andTisa non-empty subset of **S**, then **H** is a subgroup of **G**. The above definition implies that:

1. Ifaandbaremembersofbothgroups,thenc=a•bisalsoamemberofbothgroups
2. Thegroupsharethesameidentityelement
3. Ifaisamemberofbothgroups,theinverseofaisalsoamemberofbothgroups
4. Thegroupmadewiththeidentityelementof G,H=<{e},•>,isasubgroupofG
5. Eachgroupisasubgroupofitself

CyclicSubgroup: Ifasubgroupofagroupcanbegeneratedusingthepowerofanelement,thesubgroupis called the cyclic subgroup.

Thetermpowermeansrepeatedlyapplyingthegroupoperationtotheelement:

a^n-> a.a.a.a ..... a(ntimes)

Example:ThegroupG=<Z3,+>containscyclicsubgroupsfor0,1and2: If generated using 0: $0^0 \bmod 3=0, 0^1 \bmod 3=0, 0^2 \bmod 3=0$.so,H1=<{0},+>

Ifgeneratedusing1:

$1^0 \bmod 3=0, 1^1 \bmod 3=1, 1^2 \bmod 3 =(1+1) \bmod 3=2$.so,H2=G If generated using 2:

$2^0 \bmod 3=0, 2^1 \bmod 3=2, 2^2 \bmod 3=(2+2) \bmod 3=1$.so,H3=G

CyclicGroup:ACyclicgroupisagroupthatisitsowncyclicsubgroup.Theelementthatgeneratescyclic subgroupcanalsogeneratesgroupitself.Thiselementisreferredasgenerator‘g’.

Example:Inthepreviousexample,ThegroupG=<Z3,+>isacyclicgropwithtwogeneratorsg=1andg=2

Lagrange’sTheorem:

ItrelatedtheorderofagrouptotheorderofitssubgroupAssmethatGisgroupandHisitssubgroup.

IforderofGandHare|G|and|H|,respectively,basedonthistheorem|H|divides|G|.
EXAMPLE:Asper thepreviouscyclicsubgroupexample,|H1|=1,|H2|=3,|H3|=3, Obviously,allof theseordersdivide theorderof|G|.

❑ ***Order of an Element*** The **order of an element** a in a group, ord(a), is the smallest integer n such that $an = e$. The definition can be paraphrased: the order of an element is the order of the cyclic group it generates.

Example:

InthegroupG=<Z3,+>,ord(0)=1,ord(1)=3,ord(2)=3

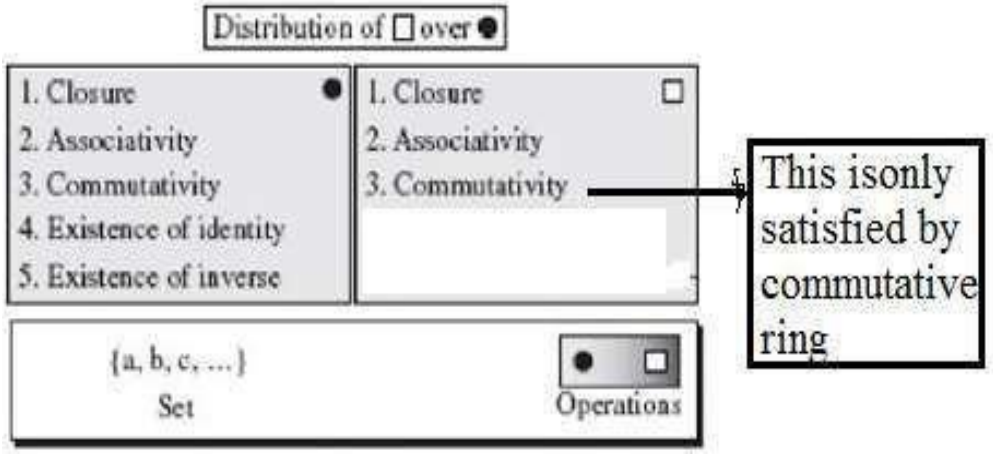
2. RING

A**Ring**,denotedas **R=<{.....},•,□>**,isanalgebraicstructurewithtwooperations(additionand multiplication).
The firstoperationmustsatisfy allfivepropertiesrequired for anabeliangroup. Thesecondoperationmustsatisfy only thefirst two.
Inaddition,thesecondoperationmustbedistributedoverthefirstoperation.
Distributivitymeansthatforalla,bandcelementsof**R**,wehave

a□(b•c) = (a□b)•(a□c) and **(a•b) □c=(a□c) •(b□c)**

CommutativeRing: Ifaringsatisfiescommutativeproperty,thenwesaytheringisa*commutativering*.

- **Ringsdonotneedtohaveamultiplicativeinverse.**



Example: Z an Integer set is a Ring structure. ExplainwhyZ(setofIntegernumbers)isaring?
Suppose that2,3,4∈Z.

- Bothadditionandmultiplicationareassociativesince

2+(3+4)=(2+3)+4,and2(3x4)=(2x3)*4*.

- Itfollowsthat

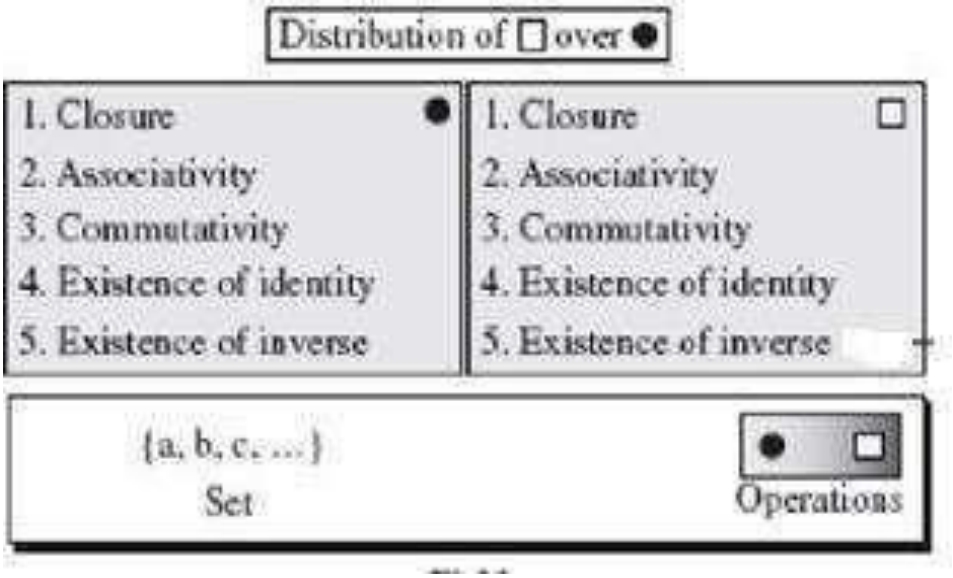
Theidentityelementforadditionis0since,2+0=2.
Theidentityelementformultiplicationis1since1x2=2.

• Addition is commutative too since 2+3=3+2 Multiplication is also commutative since 2x3=3x2,so, Z can be called a *commutative ring*).
Additionhastheinverseof-2since2+(−2)=0
(Notethatmultiplicationdoesnotneedtohaveamultiplicativeinverse.Becausemultiplicativeinverseof2 is ½. It is not an integer.
Lastly,multiplicationalsodistributesoveraddition,thatis2(3+4)=2x3+2x4.

Ringsdonotneedtohaveamultiplicative inverse.

3. FIELDS

Afield, denotedby **F=<{...},•,□>,**is acommutativeringin whichfirstandsecond operationssatisfies all five properties.
Inotherwords:
Afieldisasetwiththetwobinaryoperationsofadditionandmultiplication,bothofwhichoperationsare commutative,associative,containidentityelements,andcontaininverseelements.
Theidentityelementforadditionis0,andtheidentityelementformultiplicationis1.
Application: Afieldisastructurethatsupportstwopairsofoperations:addition/subtractionand multiplication/division



FIELDS-Example

ExplainwhyR isafield.(Rissetofrealnumbers) :Supposethat $a,b,c,d \in R$.WeknowthatR hasaddition and multiplication as binary operations since $(a+b)=c$for some c, and$ab=d$ for some d. Furthermore, we know that additionand multiplication defined onreal numbers is both commutativeandassociative.

Additionally,theidentityelementforadditionis0,$x+0=x$,andtheidentityelementformultiplicationis1, since $1x=x$.
Lastly,theinverseelementforadditionis-x,since$x+(-x)=0$(0beingtheidentityforaddition),andthe inverseelementformultiplication1/xsince$x \cdot 1/x=1$whenx≠0.

ComparisonofGroup,RingandField:

Algebraic Structure	Supported Typical Operations	Supported Typical Sets of Integers
Group	(+ –) or (× ÷)	Z_n or Z_n^*
Ring	(+ –) and (×)	Z
Field	(+ –) and (× ÷)	Z_p

Checkwhether$Z_p$isfieldstructureor not?

Z_5 {0,1,2,3,4} (0,0),(1,4),(2,3)

Additive Inverses table

	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Z_5^* {1,2,3,4} (1,1),(2,3),(4,4)

multiplicative Inverses tabl

	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Finite Fields:

A finite field, a field with a finite number of elements. The finite fields are usually called **Galois fields** and denoted as **GF(p^n)**.

Note:AGaloisfield,**GF(p^n)**,isafinitefieldwith**p^n**elementswherepisprime.

GF(p) Fields: When n=1, we have GF(p) field. Tis field can be the set Z, (0,1,2,p-1), with two operations addition and multiplication. Each element has an additive inverse and that nonzero elements have amultiplicative inverse for prime p.

ExampleforGF(p)Field: AverycommonfieldinthiscategoryisGF(2)withtheset{0,1}andtwo operationsadditionandmultiplicationashownbelow:

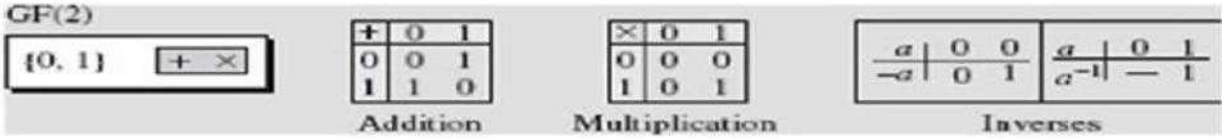


Fig. *GF(2) field*

NOTE: AdditionissameastoXORandmultiplicationisANDoperation GF(2^n) Fields:-

GF(2^n)isaFiniteFieldwith2^nelements.Theelementsinthissetaren-bitwords.Forexample,ifn=3,the set is: {000, 001, 010, 011, 100, 101,110, 111}

Example:ifn=2,thenGF(2^2)fieldinwhichthesethasfour2-bitwords:

{00,01,10,11}.

WhyGF(2^n)?

Generallycomputerstorespositiveintegersasn-bitwords,canbe8-bit,16-bit,32-bit,64-bit. Thismeansthatrangeofwords(integers)is0to2^n-1.So,themodulusis2^n

Wehavetwochoicesifweuseafieldstructure1)usingGF(p)orGF(2^n)

1) Ifwe use GF(p)withthesetZ$_p$,wherepisthelargestprimenumberlessthan2^n. Thisis Ifn=3,thelargestprimelessthan2^3is7.Thismeansthatwecannotuseinteger7,8.

2) IfweGF(2^n)withtheset2^nelements.Theelementsinthissetaren-bitwords.Example:Ifn=3,,theset is {000,001,010,011,100,101,110,111}

Polynomials

Thedataisshownasn-bitwordsinthecomputersthatsatisfythepropertiesinGF(2^n).Thesen-bitwords areeasilyrepresentedbyPolynomialofdegreen-1.

A polynomial of degree n-1 is an expression of the form: Where x^i is called the i_{th}term and $a_i$is calledcoefficient of the i_{th}term.

$$f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x^1 + a_0x^0$$

EXAMPLE Figure shows how we can represent the 8-bit word (10011001) using a polynomials.

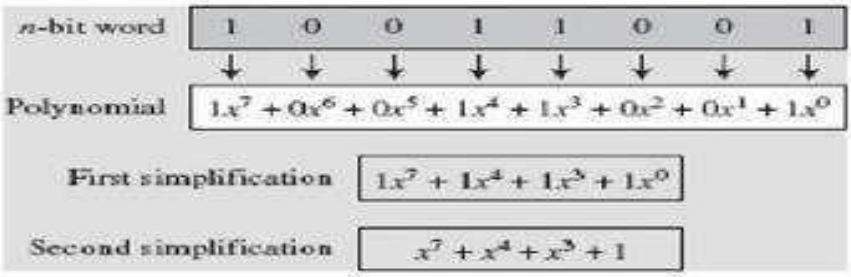


Fig. · *Representation of an 8-bit word by a polynomial*

EXAMPLE To find the 8-bit word related to the polynomial $x^5 + x^3 + x$, we first supply the omitted terms. Since $n = 8$, it means the polynomial is of degree 7. The expanded polynomial is

$$0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0$$

This is related to the 8-bit word 00100110.

Note: Polynomials representing n-bit words use two fields: GF(2) for Coefficients and GF(2^n) for terms.

Modulus:

Addition of two polynomials never creates a polynomial out of the set. However, multiplication of two polynomials may create a polynomial with degrees more than **n-1**. This means that we need to divide the result by a modulus and keep only the remainder.

A ***Prime Polynomial*** cannot be factored into a polynomial with degree of less than **n**. Such polynomials are referred to as ***Irreducible polynomials***.

Table *List of irreducible polynomials*

Degree	Irreducible Polynomials
1	$(x + 1), (x)$
2	$(x^2 + x + 1)$
3	$(x^3 + x^2 + 1), (x^3 + x + 1)$
4	$(x^4 + x^3 + x^2 + x + 1), (x^4 + x^3 + 1), (x^4 + x + 1)$
5	$(x^5 + x^2 + 1), (x^5 + x^3 + x^2 + x + 1), (x^5 + x^4 + x^3 + x + 1),$ $(x^5 + x^4 + x^3 + x^2 + 1), (x^5 + x^4 + x^2 + x + 1)$

Operations on Polynomials: Addition:

Addition and Subtraction operations on polynomials are the same operation.

The addition operation for polynomials with coefficient in GF(2) is add the coefficients of the corresponding term in GF(2).

Adding two polynomials of degree **n-1** always create a polynomial with degree **n-1**, which means that we do not need to reduce the result using the modulus.

AdditiveIdentity: Theadditiveidentityinapolynomialisazeropolynomial(apolynomialwithall coefficients set to zero).

Additiveinverse: TheadditiveinverseofapolynomialwithcoefficientsinGF(2)isthepolynomialitself. Thismeansthatthesubtractionoperationisthesameastheadditionoperation.

Polynomials-Addition

EXAMPLE Let us do $(x^5 + x^2 + x) \oplus (x^3 + x^2 + 1)$ in $GF(2^8)$. We use the symbol $\oplus$ to show that we mean polynomial addition. The following shows the procedure:

$$\begin{array}{ll} 0x^7 + 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0 & \oplus \\ 0x^7 + 0x^6 + 0x^5 + 0x^4 + 1x^3 + 1x^2 + 0x^1 + 1x^0 & \\ \hline 0x^7 + 0x^6 + 1x^5 + 0x^4 + 1x^3 + 0x^2 + 1x^1 + 1x^0 & \rightarrow \quad x^5 + x^3 + x + 1 \end{array}$$

There is a short-cut: keeps the uncommon terms and delete the common terms. In other words, x^5, x^3, x, and 1 are kept and x^2, which is common in the two polynomials, is deleted.

Polynomials-Multiplication

- Multiplicationinpolynomialsisthesumofthemultiplicationofeachtermofthefirstpolynomial with each term of the second polynomial.
- Themultiplicationmaycreatetermswithdegreemorethan**n-1**,whichmeanstheresultneedstobe reduced using a modulus polynomial

EXAMPLE Find the result of $(x^5 + x^2 + x) \otimes (x^7 + x^4 + x^3 + x^2 + x)$ in $GF(2^8)$ with irreducible polynomial $(x^8 + x^4 + x^3 + x + 1)$. Note that we use the symbol $\otimes$ to show the multiplication of two polynomials.

$$P_1 \otimes P_2 = x^5(x^7 + x^4 + x^3 + x^2 + x) + x^2(x^7 + x^4 + x^3 + x^2 + x) + x(x^7 + x^4 + x^3 + x^2 + x)$$

$$P_1 \otimes P_2 = x^{12} + x^9 + x^8 + x^7 + x^6 + x^9 + x^6 + x^5 + x^4 + x^3 + x^8 + x^5 + x^4 + x^3 + x^2$$

$$P_1 \otimes P_2 = (x^{12} + x^7 + x^2) \bmod (x^8 + x^4 + x^3 + x + 1) = x^5 + x^3 + x^2 + x + 1$$

Divisionbymodulustoreduce Polynomial

$$\begin{array}{r|l} & x^4 + 1 \\ \hline x^8 + x^4 + x^3 + x + 1 & x^{12} + x^7 + x^2 \\ & x^{12} + x^8 + x^7 + x^5 + x^4 \\ \hline & x^8 + x^5 + x^4 + x^2 \\ & x^8 + x^4 + x^3 + x + 1 \\ \hline \text{Remainder} & \boxed{x^5 + x^3 + x^2 + x + 1} \end{array}$$

Fig. *Polynomial division with coefficients in GF(2^8)*

Multiplicativeidentity&Multiplicativeinverse

Multiplicativeidentity:-Themultiplicativeidentityisalways1.Forexample,inGF(2^8),themultiplicative inverse is the bit pattern 00000001

Multiplicativeinverse:-UseextendedEuclideanAlgorithmtofindthemultiplicativeinverseofa polynomial.Thisprocessisexactlysameasforintegers.

Example: In GF(24), find the inverse of (x2+1) modulo (x4+x+1)Solution:-UsetheextendedEuclideanalgorithmasinTable:

q	r_1	r_2	r	t_1	t_2	t
(x^2+1)	(x^4+x+1)	(x^2+1)	(x)	(0)	(1)	(x^2+1)
(x)	(x^2+1)	(x)	(1)	(1)	(x^2+1)	(x^3+x+1)
(x)	(x)	(1)	(0)	(x^2+1)	(x^3+x+1)	(0)
	(1)	(0)		(x^3+x+1)	(0)	

This means that $(x^2 + 1)^{-1}$ modulo $(x^4 + x + 1)$ is $(x^3 + x + 1)$. The answer can be easily proved by multiplying the two polynomials and finding the remainder when the result is divided by the modulus.

$$[(x^2 + 1) \otimes (x^3 + x + 1)] \bmod (x^4 + x + 1) = 1$$

Polynomials-Multiplicationusingacomputer

Ifwemultiplytwopolynomials,wealsoneedtoperformdivisionoperationthatreducesanefficiency. Computerusesanalgorithmformultiplythepolynomialsthatshouldnotusedivisionoperation,instead repeatedly multiplying areducedpolynomialbyx.

Example:Insteadoffindingtheresultof$x^2 \otimes P_2$,itcandonelike $x \otimes (x \otimes P_2)$

Example:

Power	Operation	New Result	Reduction
$x^0 \otimes P_2$		$x^7+x^4+x^3+x^2+x$	No
$x^1 \otimes P_2$	$x \otimes (x^7+x^4+x^3+x^2+x)$	x^5+x^2+x+1	Yes
$x^2 \otimes P_2$	$x \otimes (x^5+x^2+x+1)$	$x^6+x^3+x^2+x$	No
$x^3 \otimes P_2$	$x \otimes (X^6+x^3+x^2+x)$	$x^7+x^4+x^3+x^2$	No
$x^4 \otimes P_2$	$x \otimes (x^7+x^4+x^3+x^2)$	x^5+x+1	Yes
$x^5 \otimes P_2$	$x \otimes (x^5+x+1)$	x^6+x^2+x	No

Resultis **$P1xP2=(x^6+x^2+x)+(x^6+x^3+x^2+x)+(x^5+x^2+x+1)=x^5+x^3+x^2+x+1$**

Simplealgorithm

1. Ifthemostsignificantbitofthepreviousresultis0,justshiftthepreviousresultonebittotheleft.
2. Ifthemostsignificantbitofthepreviousresultis1.

a) Shiftitonebittotheleft,and

b) Exclusive-ORitwiththemoduluswithoutthemostsignificantbit.

Example:MultiplyP_1=(x^5+x^2+x)byP_2=($x^7+x^4+x^3+x^2+x$)inGF(2^8)withirreducible($x^8+x^4+x^3+x+1$)
Binary representation of P2=10011110,
Irreduciblepolynomial=100011011(9bits)

Power	**Shift-leftOperation**	**Exclusive-OR**
$x^0 \otimes P_2$		10011110
$X^1 \otimes P_2$	111100	(00111100)+(00011011)=<u>00100111</u>
$X^2 \otimes P_2$	1001110	1001110
$X^3 \otimes P_2$	10011100	10011100
$X^4 \otimes P_2$	111000	(00111000)+(00011011)=00100011
$X^5 \otimes P_2$	01000110	<u>01000110</u>
$P_1 \otimes P_2$=(00100111)+ 01001110+01000110=00101111		

Multiplicationofpolynomialsin GF(2^n)canbeachievedusingshift-leftandexclusive-oroperations

Classical Substitution Ciphers

- Letters of plaintext are replaced by other letters or by numbers or symbols
- Plaintext is viewed as a sequence of bits, then substitution replaces plaintext bit patterns with ciphertext bit patterns

Caesar Cipher

- Earliest known substitution cipher
- Replaces each letter by 3rd letter on

  Example: meet me after the toga party

  PHHW PH DIWHU WKH WRJD SDUWB

Define transformation as:

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

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

- Mathematically give each letter a number

a b c d e f g h i j k l m

0 1 2 3 4 5 6 7 8 9 10 11 12

n o p q r s t u v w x y Z

13 14 15 16 17 18 19 20 21 22 23 24 25

- Then have Caesar cipher as:

$C = E(p) = (p + k) \bmod (26)$

$p = D(C) = (C - k) \bmod (26)$

Cryptanalysis of Caesar Cipher

- Only have 25 possible ciphers
  - A maps to B,..Z
- Given ciphertext, just try all shifts of letters

- Do need to recognize when have plaintext
- E.g., break ciphertext "GCUA VQ DTGCM"

Monoalphabetic Cipher

- Rather than just shifting the alphabet
- Could shuffle (jumble) the letters arbitrarily
- Each plaintext letter maps to a different random ciphertext letter
- Key is 26 letters long

Plain: abcdefghijklmnopqrstuvwxyz

Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

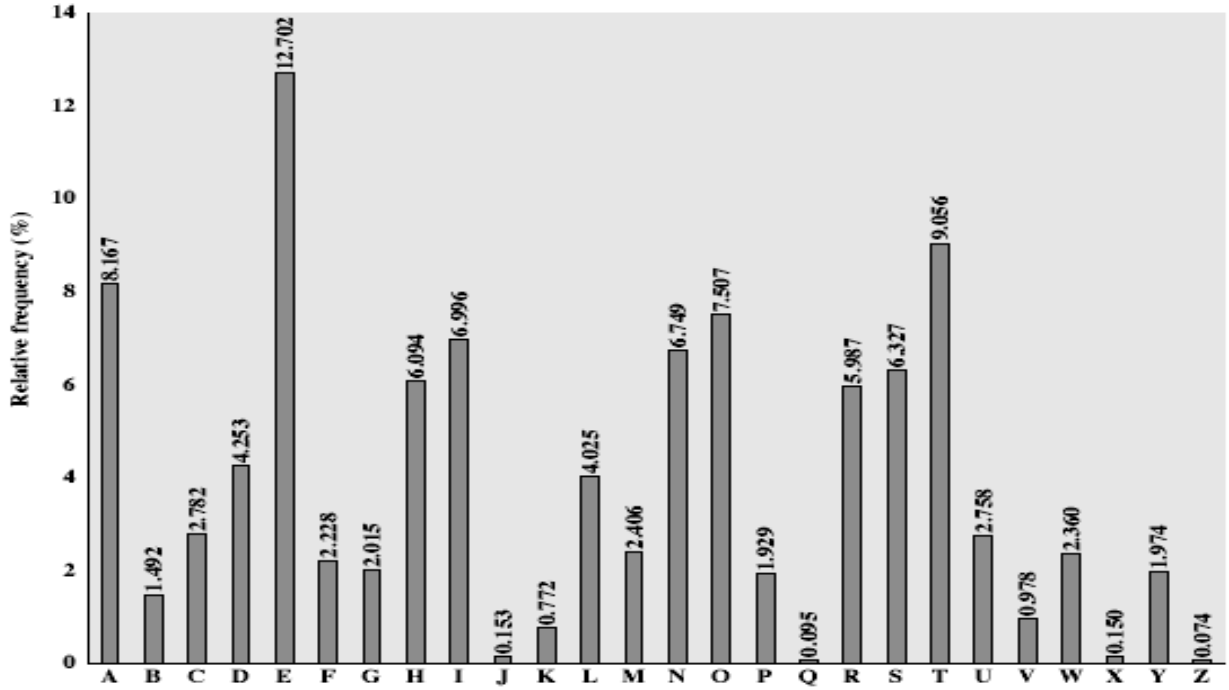
Plaintext: ifwewishtoreplaceletters

Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA

Monoalphabetic Cipher Security

- Now have a total of 26! = 4 x 10^{26} keys
- Is that secure?
- Problem is language characteristics
  - Human languages are **redundant**
  - Letters are not equally commonly used

English Letter Frequencies



Note that all human languages have varying letter frequencies, though the number of letters and their frequencies varies.

Example Cryptanalysis

- Given ciphertext:

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ

VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX

EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

- Count relative letter frequencies (see text)
- Guess P & Z are e and t
- Guess ZW is th and hence ZWP is the
- Proceeding with trial and error finally get:

it was disclosed yesterday that several informal but

direct contacts have been made with political

representatives of the viet cong in moscow

One-Time Pad- Vernam Cipher

- If a truly random key as long as the message is used, the cipher will be secure - One-Time pad
- E.g., a random sequence of 0's and 1's XORed to plaintext, no repetition of keys
- Unbreakable since ciphertext bears no statistical relationship to the plaintext
- For **any** plaintext**,** it needs a random key of the same length
  - Hard to generate large amount of keys
- Have problem of safe distribution of key
- It is an unbreakable cipher.
- The key is exactly same as the length of message which is encrypted.
- The key is made up of random symbols.
- As the name suggests, key is used one time only and never used again for any other message to be encrypted.

Example: In alphabet sequence from 0 to 25 find the alphabet value in plain text and add with key text. If the result is greater than 25 than do modulo 26.

Consider,

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
13	14	15	16	17	18	19	20	21	22	23	24	25
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

plain text Hi how are you

key dwxcvtypljk

Plain Text :	H(7)	I(8)	H(7)	O(14)	W(22)	A(0)	R(17)	E(4)	Y(24)	O(14)	U(20)
Key :	d(3)	w(22)	x(23)	c(2)	v(21)	t(19)	y(24)	p(15)	l(8)	j(9)	k(10)
Result:	**K(10)**	**E(30%26=4)E**		**Q(16)**	**R(17)**	**T(19)**	**P(15)**	**T(19)**	**G(6)**	**X(23)**	**E(4)**

Cipher Text : KEEQRTPTGXE

Decryption: Apply result – key, if the value is less than 0, apply negative modulo concept.

Use in Cryptanalysis

- key concept - monoalphabetic substitution ciphers do not change relative letter frequencies
- discovered by Arabian scientists in 9th century
- calculate letter frequencies for ciphertext
- compare counts/plots against known values
- if caesar cipher look for common peaks/troughs
    - peaks at: A-E-I triple, NO pair, RST triple
    - troughs at: JK, X-Z
- for monoalphabetic must identify each letter
    - tables of common double/triple letters help

Classical Substitution Ciphers

- Letters of plaintext are replaced by other letters or by numbers or symbols
- Plaintext is viewed as a sequence of bits, then substitution replaces plaintext bit patterns with ciphertext bit patterns

Polyalphabetic Ciphers

- Improve security using multiple cipher alphabets
- Make cryptanalysis harder with more alphabets to guess and flatter frequency distribution
- Use a key to select which alphabet is used for each letter of the message and uses each alphabet in turn, repeats from start after end of key is reached
- Improve security using multiple cipher alphabets
- Make cryptanalysis harder with more alphabets to guess and flatter frequency distribution
- Use a key to select which alphabet is used for each letter of the message and uses each alphabet in turn, repeats from start after end of key is reached

Vigenère Cipher

- Simplest polyalphabetic substitution cipher
- Effectively multiple caesar ciphers
- Key is multiple letters long $K = k_1\ k_2\ ...\ k_d$
- i^{th} letter specifies i^{th} alphabet to use
- Use each alphabet in turn and repeat from start after d letters in message.

Example of Vigenère Cipher

- write the plaintext out
- write the keyword repeated above it
- use each key letter as a caesar cipher key
- encrypt the corresponding plaintext letter
- eg using keyword *deceptive*

key: deceptivedeceptivedeceptive

plaintext: wearediscoveredsaveyourself

ciphertext:ZICVTWQNGRZGVTWAVZHCQYGLMGJ

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

Playfair Cipher

- Not even the large number of keys in a monoalphabetic cipher provides security
- One approach to improving security was to encrypt multiple letters. The **Playfair Cipher** is an example
- Invented by Charles Wheatstone in 1854, but named after his friend Baron Playfair

Playfair Key Matrix

- A 5X5 matrix of letters based on a keyword
- Fill in letters of keyword (Except duplicates)
- Fill rest of matrix with other letters

Eg. using the keyword MONARCHY

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

Playfair Encrypting and Decrypting

Plaintext is encrypted two letters at a time

1. if a pair is a repeated letter, insert filler like 'X'
2. if both letters fall in the same row, replace each with letter to right (wrapping back to start from end)
3. if both letters fall in the same column, replace each with the letter below it (again wrapping to top from bottom)
4. otherwise each letter is replaced by the letter in the same row and in the column of the other letter of the pair

Security of Playfair Cipher

- Security much improved over monoalphabetic, since have 26 x 26 = 676 digrams
- would need a 676 entry frequency table to analyse (verses 26 for a monoalphabetic) and correspondingly more ciphertext
- It was widely used for many years
  - eg. by US & British military in WW1
- It **can** be broken, given a few hundred letters.

Transposition Ciphers

- Now consider classical **transposition** or **permutation** ciphers
- These hide the message by rearranging the letter order, without altering the actual letters used
- Can recognise these since have the same frequency distribution as the original text

There are 3 transposition ciphers such as

- **Rail Fence Cipher**
- **Row Cipher**
- **Route Cipher**

Rail Fence Cipher

- Write message letters out diagonally over a number of rows
- Then read off cipher row by row
- E.g., write message “meet me after the toga party” as:

m e m a t r h t g p r y

e t e f e t e o a a t

Giving ciphertext : MEMATRHTGPRYETEFETEOAAT

Row Cipher

- A more complex transposition
- write letters of message out in rows over a specified number of columns
- Then reorder the columns according to some key before reading off the rows

Key: 3 4 2 1 5 6 7

Plaintext: a t t a c k p

o s t p o n e

d u n t i l t

w o a m x y z

Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

Route Cipher

- A more complex transposition
- write letters of message out in rows and columns (Square or rectangle).
- select a route (diagonal, spiral, orthogonal, crab and so on) to enter characters and frame ciphertext top-down and left-right approach.
- Consider plain text
  - "attack postponed until next call be cool"

Plaintext:

a t a p p u

t c o o n e

k s n t x l

t e i t l c

d l c b o l

n a e o y z

Ciphertext: atapputcooneksntxlteitlcdlcbolnaeoyz

Product Ciphers

- Ciphers using substitutions or transpositions are not secure because of language characteristics
- Hence consider using several ciphers in succession to make harder, but:
  - two substitutions make a more complex substitution
  - two transpositions make more complex transposition
  - but a substitution followed by a transposition makes a new much harder cipher
- This is bridge from classical to modern ciphers

Transposition Ciphers

- Now consider classical **transposition** or **permutation** ciphers
- These hide the message by rearranging the letter order, without altering the actual letters used
- Can recognise these since have the same frequency distribution as the original text

There are 3 transposition ciphers such as

- **Rail Fence Cipher**
- **Row Cipher**
- **Route Cipher**

Rail Fence Cipher

- Write message letters out diagonally over a number of rows
- Then read off cipher row by row
- E.g., write message “meet me after the toga party” as:

m e m a t r h t g p r y

e t e f e t e o a a t

Giving ciphertext : MEMATRHTGPRYETEFETEOAAT

Row Cipher

- A more complex transposition
- write letters of message out in rows over a specified number of columns
- Then reorder the columns according to some key before reading off the rows

Key: 3 4 2 1 5 6 7

Plaintext: a t t a c k p

o s t p o n e

d u n t i l t

w o a m x y z

Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

Route Cipher

- A more complex transposition
- write letters of message out in rows and columns (Square or rectangle).
- select a route (diagonal, spiral, orthogonal, crab and so on) to enter characters and frame ciphertext top-down and left-right approach.
- Consider plain text
  - "attack postponed until next call be cool"

Plaintext:

a t a p p u

t c o o n e

k s n t x l

t e i t l c

d l c b o l

n a e o y z

Ciphertext: atapputcooneksntxlteitlcdlcbolnaeoyz

Product Ciphers

- Ciphers using substitutions or transpositions are not secure because of language characteristics
- Hence consider using several ciphers in succession to make harder, but:
  - two substitutions make a more complex substitution
  - two transpositions make more complex transposition
  - but a substitution followed by a transposition makes a new much harder cipher
- This is bridge from classical to modern ciphers

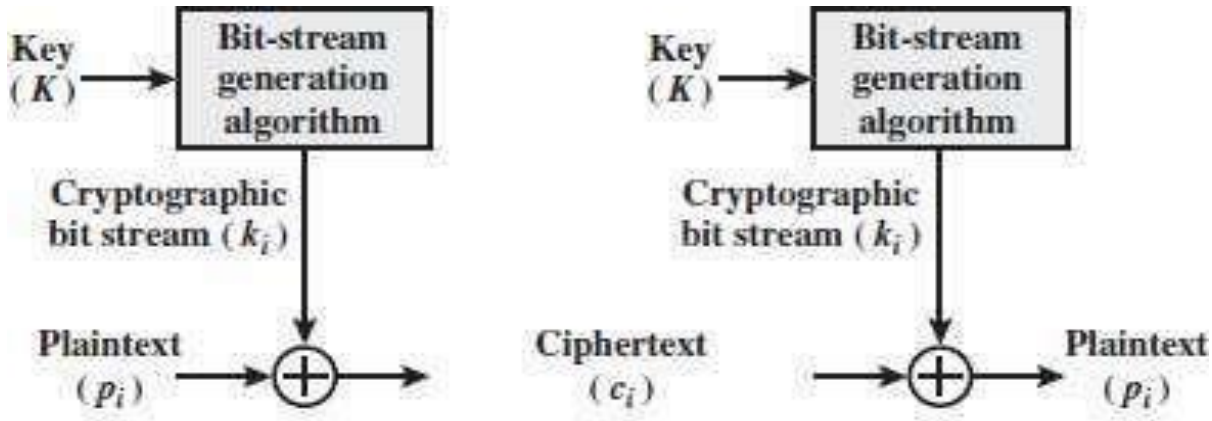
Stream and Block ciphers &

Introduction to Modern Symmetric Key Ciphers

- **Stream and Block ciphers**
- **Block cipher design principles**
- **Ideal block cipher**
- **S-P network**
- **Confusion & Diffusion**
- **Fiestal Cipher Structure**
- **Fiestal Cipher design elements**

Stream ciphers

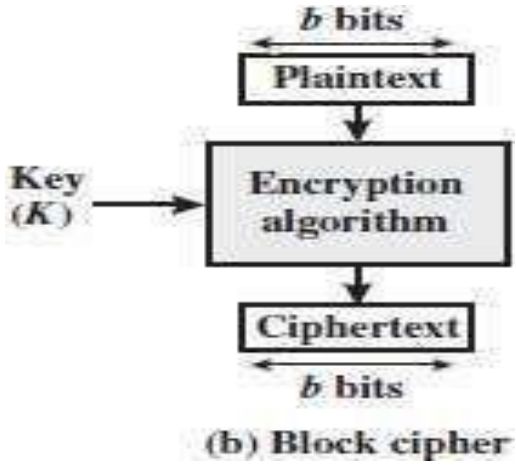
A stream cipher is one that encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the **auto keyed Vigenère cipher and the Vernam cipher.**



(a) Stream cipher using algorithmic bit-stream generator

Block ciphers

A **block cipher** is one in which a block of plain text is treated as a whole and used to produce a cipher text block of equal length. Typically, a block size of 64 or 128 bits is used.

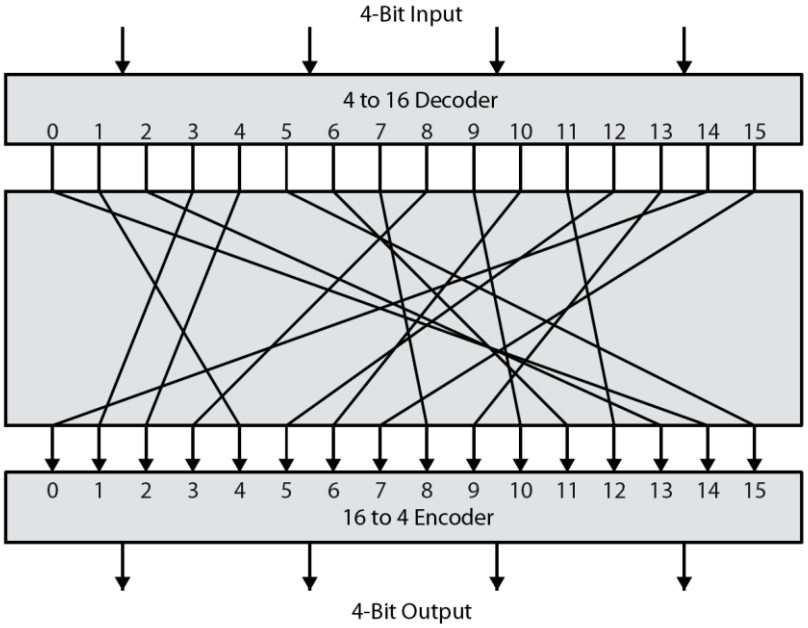


(b) Block cipher

Block Cipher Principles

- Most symmetric block ciphers are based on a **Feistel Cipher Structure**
- Block ciphers look like an extremely large substitution
- Would need table of 2^{64} entries for a 64-bit block
- Instead create from smaller building blocks
- Using idea of a product cipher
- provide secrecy /authentication services

An ideal block cipher is as follows:



Feistel refers to an *n*-bit general substitution as an ideal block cipher, because it allows for the maximum number of possible encryption mappings from the plaintext to ciphertext block. A 4-bit input produces

one of 16 possible input states, which is mapped by the substitution cipher into a unique one of 16 possible output states, each of which is represented by 4 ciphertext bits. The encryption and decryption mappings can be defined by a tabulation. It illustrates a tiny 4-bit substitution to show that each possible input can be arbitrarily mapped to any output - which is why its complexity grows so rapidly.

Substitution-Permutation Ciphers

- Substitution-permutation (S-P) networks [Shannon, 1949]
  - modern substitution-transposition product cipher
- These form the basis of modern block ciphers
- provide *confusion* and *diffusion* of message
- S-P networks are based on the two primitive cryptographic operations
  - *substitution* (S-box)
  - *permutation* (P-box)

Claude Shannon's 1949 paper has the key ideas that led to the development of modern block ciphers. Critically, it was the technique of layering groups of S-boxes separated by a larger P-box to form the S-P network, a complex form of a product cipher. He also introduced the ideas of *confusion* and *diffusion*, notionally provided by S-boxes and P-boxes (in conjunction with S-boxes).

Confusion and Diffusion

Cipher needs to completely obscure statistical properties of original message. A one-time pad does this and more practically Shannon suggested S-P networks to obtain:

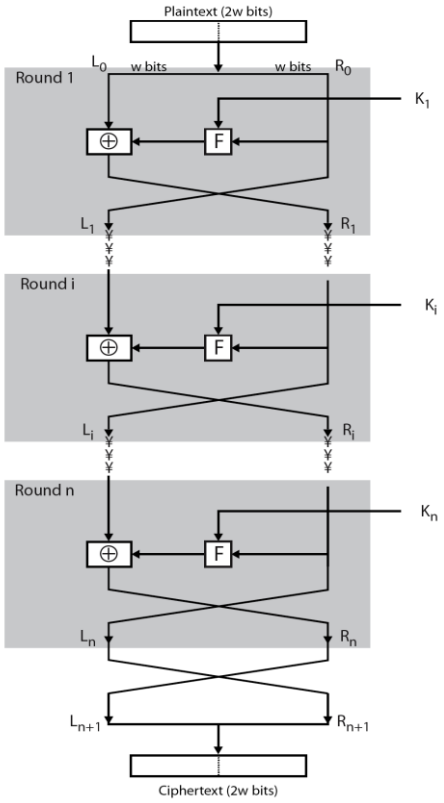
- **Diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext
- **Confusion** – makes relationship between ciphertext and key as complex as possible

Every block cipher involves a transformation of a block of plaintext into a block of ciphertext, where the transformation depends on the key. The mechanism of diffusion seeks to make the statistical relationship between the plaintext and ciphertext as complex as possible in order to thwart attempts to deduce the key. confusion seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible, again to thwart attempts to discover the key. So successful are diffusion and confusion in capturing the essence of the desired attributes of a block cipher that they have become the cornerstone of modern block cipher design.

Feistel Cipher Structure

Horst Feistel, working at IBM Thomas J Watson Research Labs devised a suitable invertible cipher structure in early 70's. One of Feistel's main contributions was the invention of a suitable

structure which adapted Shannon's S-P network in an easily inverted structure. Essentially the same h/w or s/w is used for both encryption and decryption, with just a slight change in how the keys are used. One layer of S-boxes and the following P-box are used to form the round function. The structure is as follows:



- **Feistel cipher** implements Shannon's S-P network concept
  - based on invertible product cipher
- Process through multiple rounds which
  - partitions input block into two halves
  - perform a substitution on left data half
  - based on round function of right half & subkey
  - then have permutation swapping halves

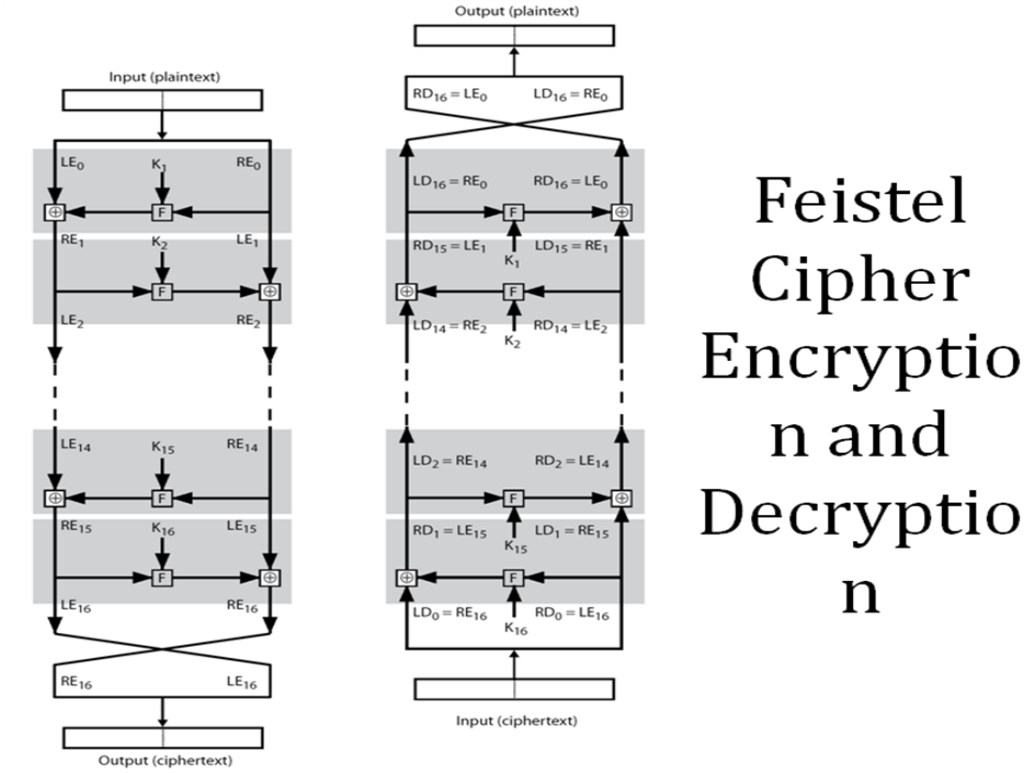
Feistel Cipher Design Elements

The exact realization of a Feistel network depends on the choice of the following parameters and design features:

- **block size** - increasing size improves security, but slows cipher
- **key size** - increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- **number of rounds** - increasing number improves security, but slows cipher
- **subkey generation** algorithm - greater complexity can make analysis harder, but slows cipher
- **round function** - greater complexity can make analysis harder, but slows cipher
- **fast software en/decryption -** more recent concern for practical use
- **ease of analysis - for easier validation &** testing of strength

Feistal Encryption and Decryption

The process of decryption with a Feistel cipher, as shown in Stallings Figure 3.3, is essentially the same as the encryption process. The rule is as follows: Use the ciphertext as input to the algorithm, but use the subkeys *Ki* in reverse order. That is, use *Kn* in the first round, *Kn*–1 in the second round, and so on until *K*1 is used in the last round. This is a nice feature because it means we need not implement two different algorithms, one for encryption and one for decryption.



DES (DATA ENCRYPTION STANDARD)

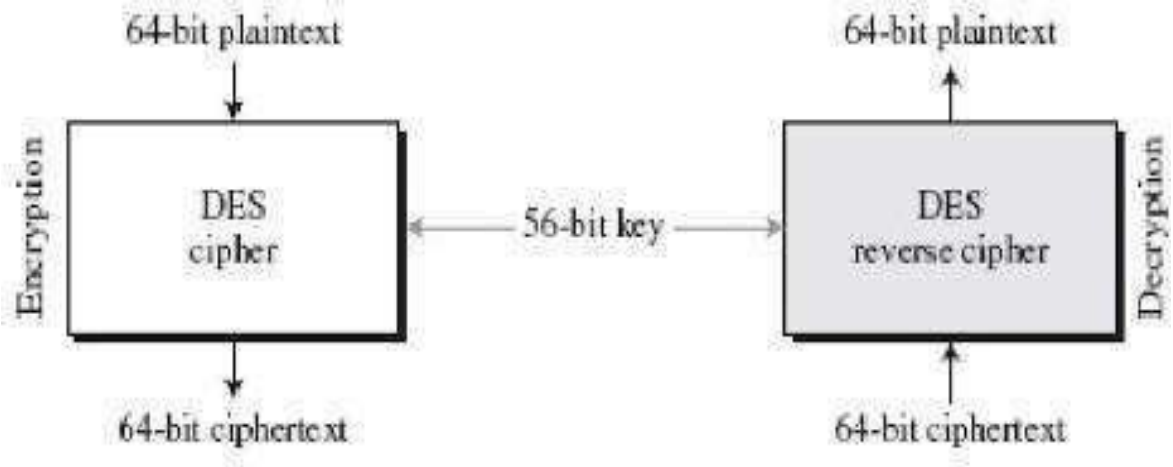
- Most widely used block cipher in world
- adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46
- encrypts 64-bit data using 56-bit key
- has been widespread use and also considerable controversy over its security

The most widely used private key block cipher, is the Data Encryption Standard (DES). It was adopted in 1977 by the National Bureau of Standards as Federal Information Processing Standard 46 (FIPS PUB 46). DES encrypts data in 64-bit blocks using a 56-bit key. The DES enjoys widespread use. It has also been the subject of much controversy its security.

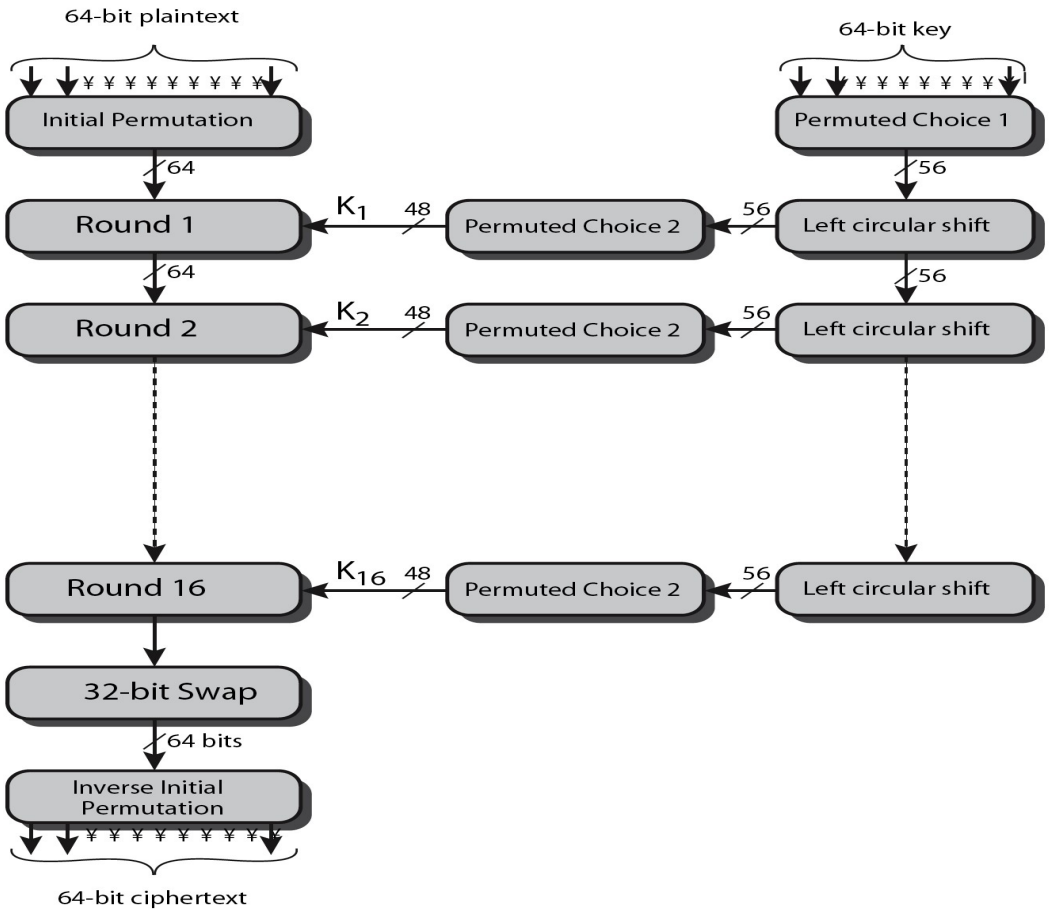
DES History

- IBM developed Lucifer cipher
  - by team led by Feistel in late 60's
  - used 64-bit data blocks with 128-bit key
- Redeveloped as a commercial cipher with input from NSA and others
- In 1973 NBS issued request for proposals for a national cipher standard
- IBM submitted their revised Lucifer which was eventually accepted as the DES
- In the late 1960s, IBM set up a research project in computer cryptography led by Horst Feistel. The project concluded in 1971 with the development of the LUCIFER algorithm. LUCIFER is a Feistel block cipher that operates on blocks of 64 bits, using a key size of 128 bits.
- Because of the promising results produced by the LUCIFER project, IBM embarked on an effort, headed by Walter Tuchman and Carl Meyer, to develop a marketable commercial encryption product that ideally could be implemented on a single chip. It involved not only IBM researchers but also outside consultants and technical advice from NSA. The outcome of this effort was a refined version of LUCIFER that was more resistant to cryptanalysis but that had a reduced key size of 56 bits, to fit on a single chip.
- In 1973, the National Bureau of Standards (NBS) issued a request for proposals for a national cipher standard. IBM submitted the modified LUCIFER. It was by far the best algorithm proposed and was adopted in 1977 as the Data Encryption Standard.

DES En/Decryption TOP VIEW



DES Encryption Overview



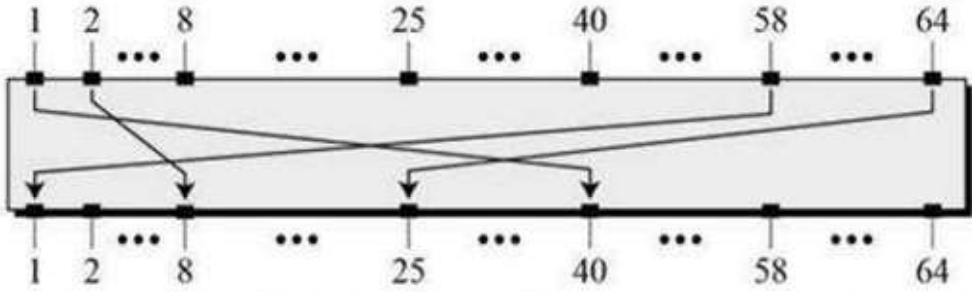
Initial Permutation IP

- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half

(a) 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

Inverse Initial Permutation



(b) Inverse Initial Permutation (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 Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

$L_i = R_{i-1}$

$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$

- F takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using perm E
  - adds to subkey using XOR

– passes through 8 S-boxes to get 32-bit result

– finally permutes using 32-bit perm P

Here the internal structure of the DES round function F, which takes R half & subkey, and processes them through E, add subkey, S & P. This follows the classic structure for a feistel cipher. Note that the s-boxes provide the "confusion" of data and key values, whilst the permutation P then spreads this as widely as possible, so each S-box output affects as many S-box inputs in the next round as possible, giving "diffusion".

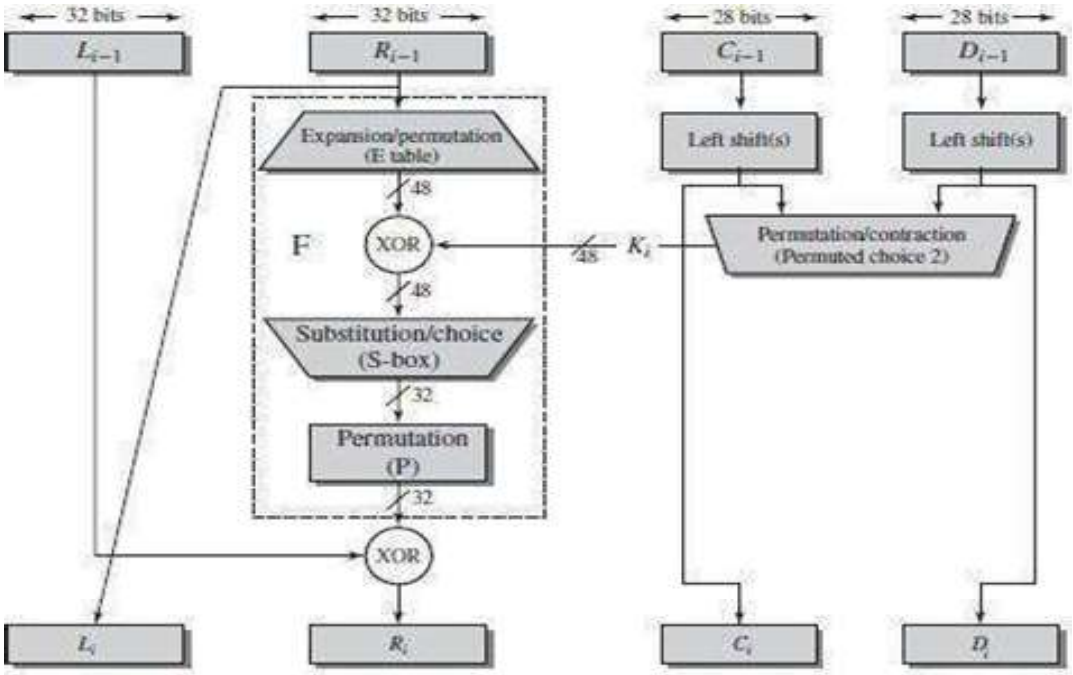
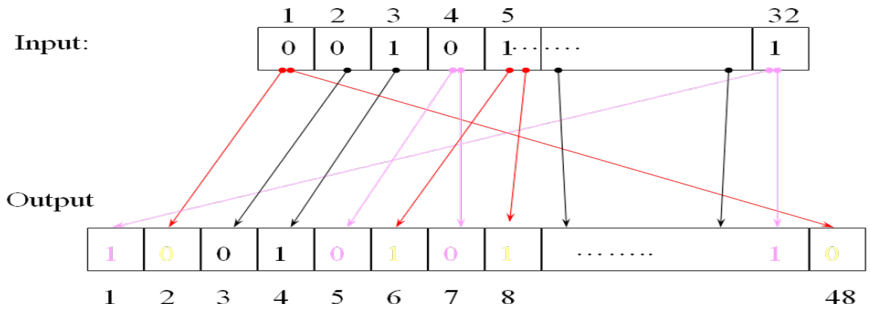


Figure 3.6 Single Round of DES Algorithm

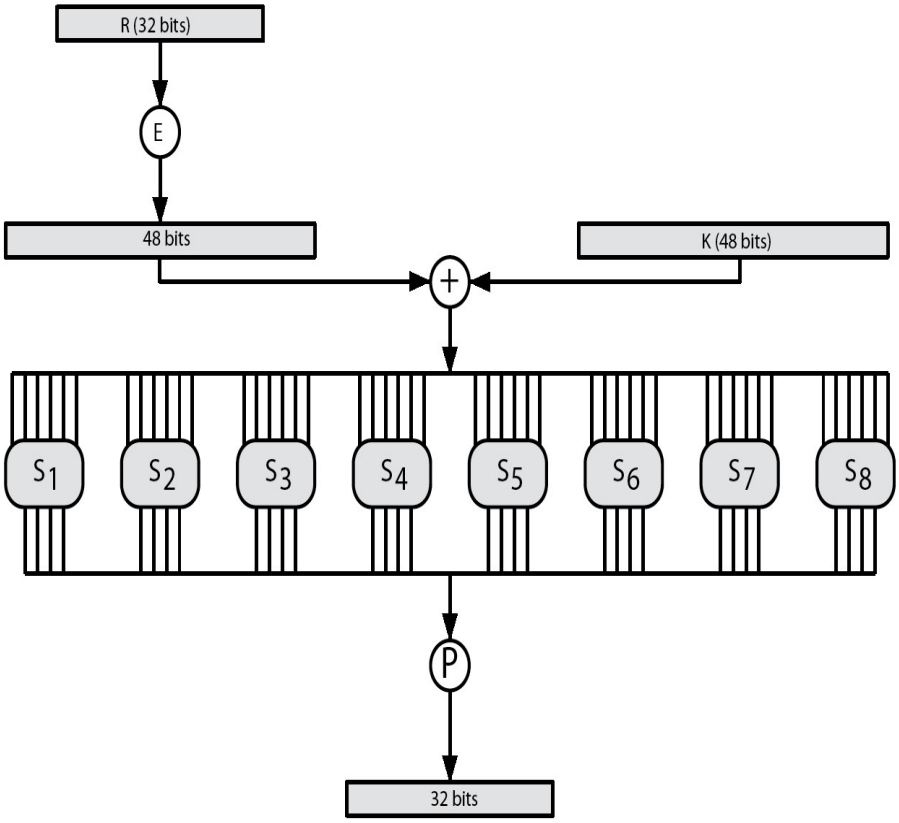
(c) Expansion Permutation (E)

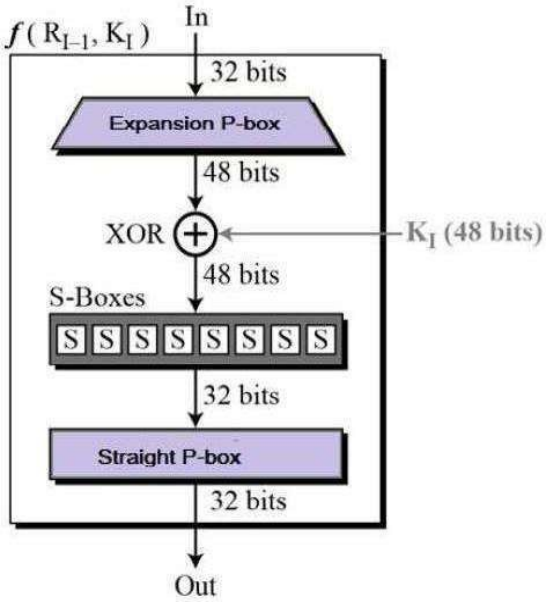
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Bits Expansion



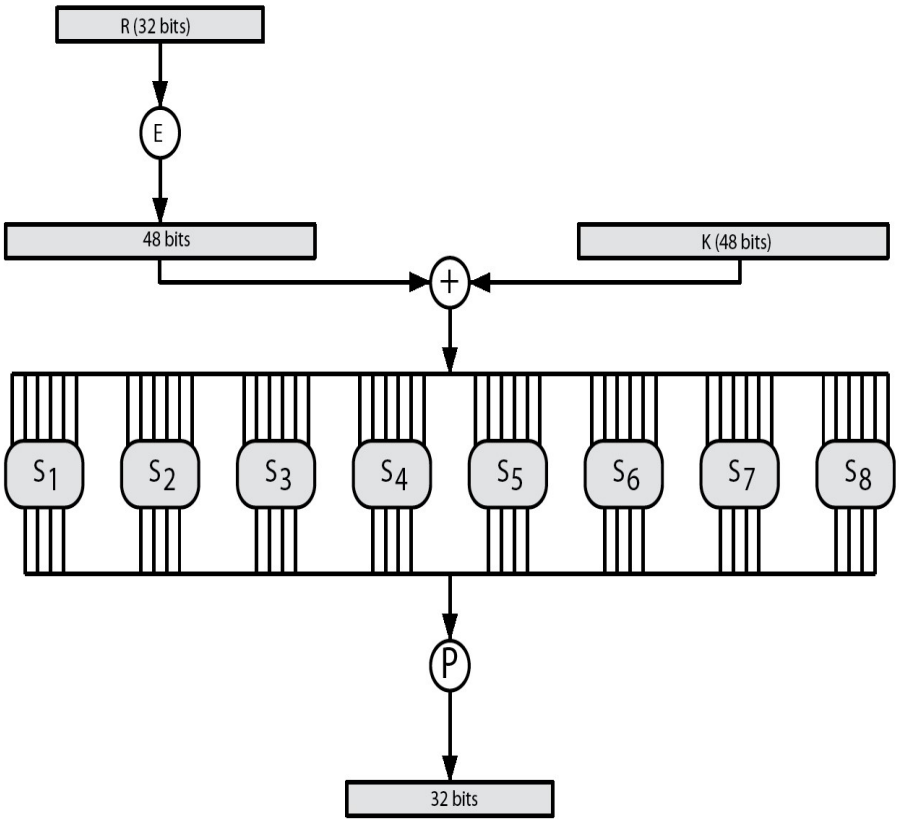
DES Round Structure

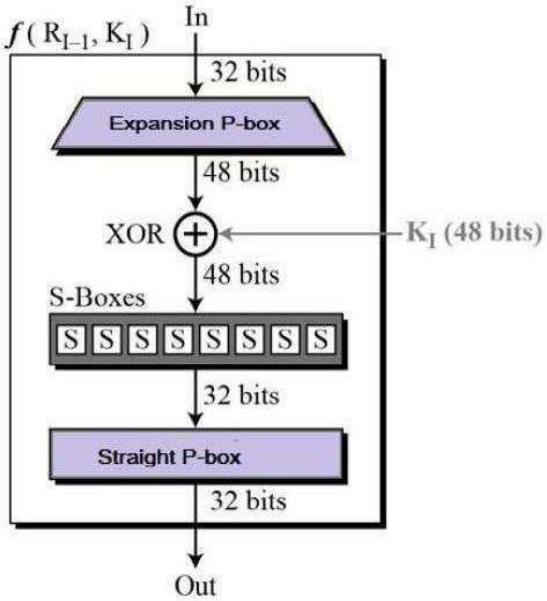




DES (DATA ENCRYPTION STANDARD)

DES Round Structure





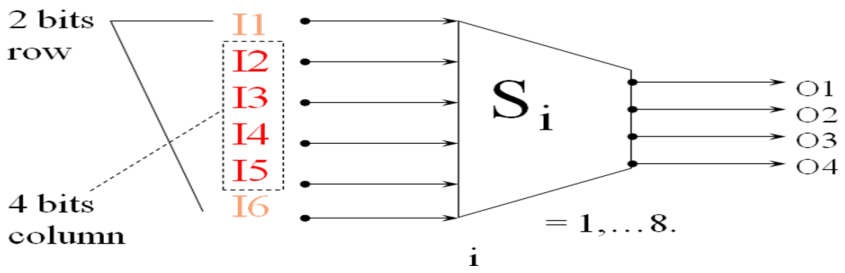
Substitution Box

- Have eight S-boxes which map 6 to 4 bits
- each S-box is actually 4 little 4 bit boxes

  - outer bits 1 & 6 (row bits) select one row of 4
  - inner bits 2-5 (col bits) are substituted
  - result is 8 lots of 4 bits, or 32 bits
- row selection depends on both data & key
  - feature known as autoclaving (autokeying)
- example:
  - S(18 09 12 3d 11 17 38 39) = 5fd25e03
- The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. These transformations are defined in Stallings Table 3.3, which is interpreted as follows: The first and last bits of the input to box S*i* form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for S*i*. The middle four bits select one of the sixteen columns. The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output. For example, in S1, for input 011001, the row is 01 (row 1) and the column is 1100 (column 12). The value in row 1, column 12 is 9, so the output is 1001.
- The example lists 8 6-bit values (ie 18 in hex is 011000 in binary, 09 hex is 001001 binary, 12 hex is 010010 binary, 3d hex is 111101 binary etc), each of which is replaced following the process detailed above using the appropriate S-box. ie
- S1(011000) lookup row 00 col 1100 in S1 to get 5
- S2(001001) lookup row 01 col 0100 in S2 to get 15 = f in hex
- S3(010010) lookup row 00 col 1001 in S3 to get 13 = d in hex
- S4(111101) lookup row 11 col 1110 in S4 to get 2 etc

S-Box (Substitute and Shrink)

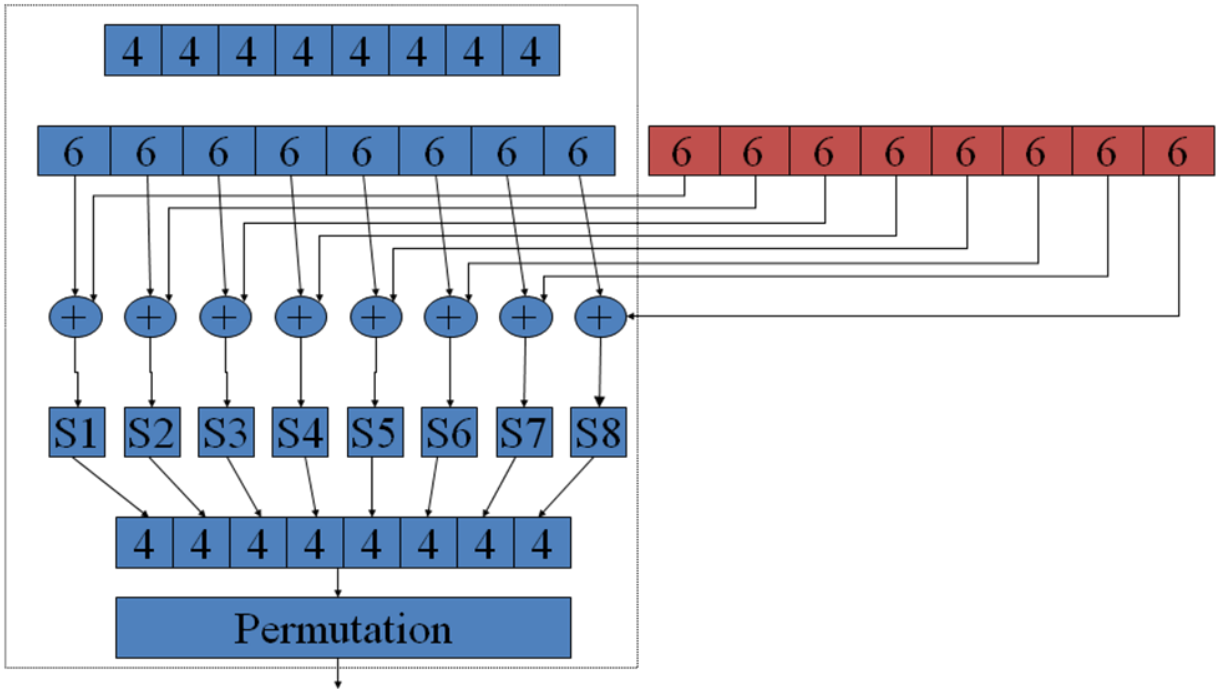
- **48 bits ==> 32 bits. (8*6 ==> 8*4)**
- **2 bits used to select amongst 4 substitutions for the rest of the 4-bit quantity**



S-Box Examples

Each row and column contain different numbers

Mangler Function



DES Key Schedule

The DES Key Schedule generates the subkeys needed for each data encryption round. The 64-bit key input is first processed by Permuted Choice One (Stallings Table 3.4b). The resulting 56-bit key is then treated as two 28-bit quantities C & D. In each round, these are separately processed throgh a circular left shift (rotation) of 1 or 2bits as shown in Stallings Table 3.4d. These shifted values serve as input to the next round of the key schedule. They also serve as input to Permuted Choice Two (Stallings Table 3.4c), which produces a 48-bit output that serves as input to the round function F.

The 56 bit key size comes from security considerations as we know now. It was big enough so that an exhaustive key search was about as hard as the best direct attack (a form of differential cryptanalysis called a T-attack, known by the IBM & NSA researchers), but no bigger. The extra 8 bits were then used as parity (error detecting) bits, which makes sense given the original design use for hardware communications links. However we hit an incompatibility with simple s/w implementations since the top bit in each byte is 0 (since ASCII only uses 7 bits), but the DES key schedule throws away the bottom bit! A good implementation needs to be cleverer!

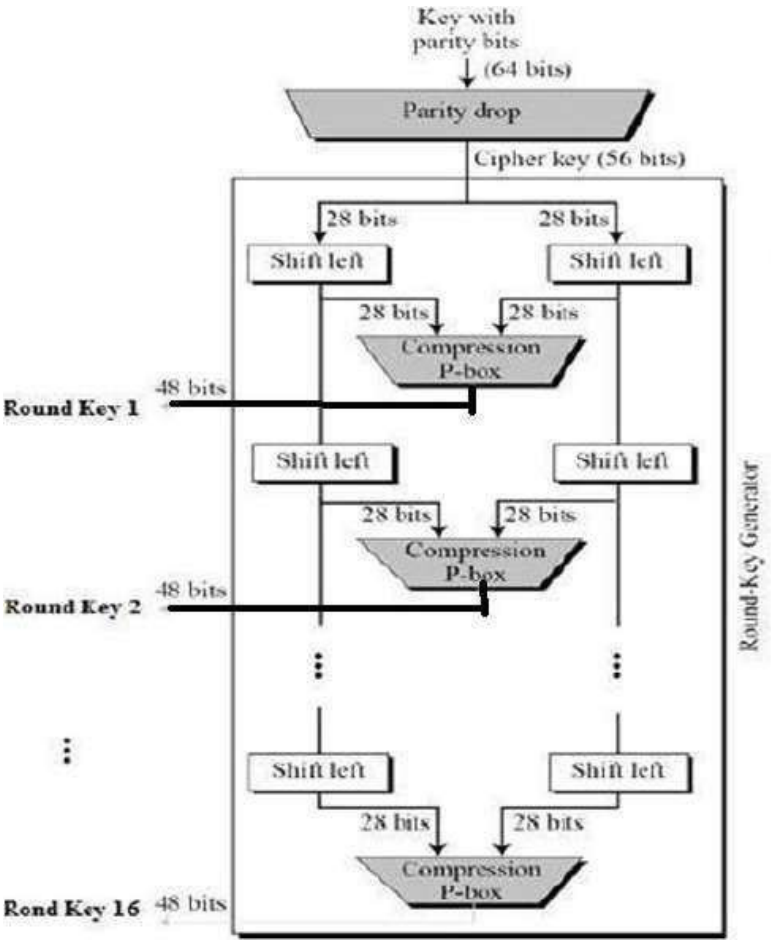
- forms subkeys used in each round
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves

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

– 16 stages consisting of:

- rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K
- selecting 24-bits from each half & permuting them by PC2 for use in round function F

Per-Round Key Generation



Shifting

Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits

DES Key generation steps

Table 3.4 DES Key Schedule Calculation

(a) Input Key

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

(b) Permuted Choice One (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

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(d) Schedule of Left Shifts

Round Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits Rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

DES Decryption

- Decrypt must unwind steps of data computation
- With Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)
  - IP undoes final FP step of encryption
  - 1st round with SK16 undoes 16th encrypt round
  - 16th round with SK1 undoes 1st encrypt round
  - then final FP undoes initial encryption IP
  - thus recovering original data value

As with any Feistel cipher, DES decryption uses the same algorithm as encryption except that the subkeys are used in reverse order SK16 .. SK1. If you trace through the DES overview diagram can see how each decryption step top to bottom with reversed subkeys, undoes the equivalent encryption step moving from bottom to top.

Multiple DES

(Double and Triple DES)

Multiple Encryption/Decryption

Given the potential vulnerability of DES to a brute-force attack,there has been considerable interest in finding an alternative. One approach is to design a completely new algorithm, of which AES is a prime example. Another alternative, which would preserve the existing investment in software and equipment, is to use multiple encryption with DES and multiple keys. We examine the widely accepted triple DES (3DES) approach.

Clear a replacement for DES was needed

  - theoretical attacks that can break it
  - demonstrated exhaustive key search attacks

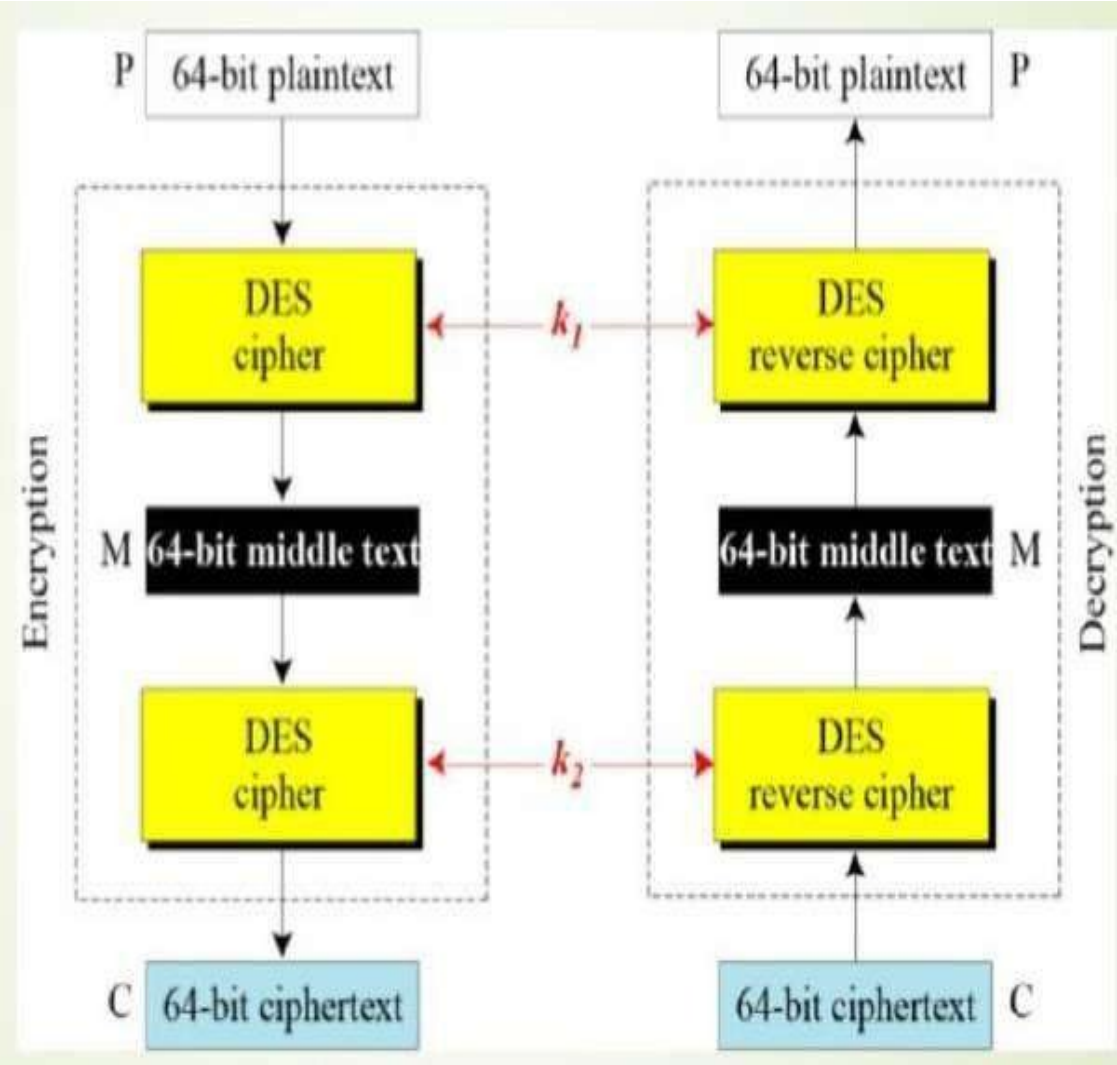
- AES is a new cipher alternative
- prior to this alternative was to use multiple encryption with DES implementations
- Triple-DES is the chosen form
- The simplest form of multiple encryption has two encryption stages and two keys - Double-DES.
- Have concern that there might be a single key that is equivalent to using 2 keys as above, not likely but only finally proved in 1992.

More seriously have the “meet-in-the-middle” attack, first described by Diffie in 1977. It is a known plaintext attack (ie have know pair (P,C), and attempts to find by trial-and-error a value X in the “middle” of the double-DES encryption of this pair, and chances of this are much better at O(2^56) than exhaustive search at O(2^112).

Double-DES

- could use 2 DES encrypts on each block
  - $C = E_{K2}(E_{K1}(P))$
- issue of reduction to single stage
- and have “meet-in-the-middle” attack
  - works whenever use a cipher twice

- can show takes $O(2^{56})$ steps



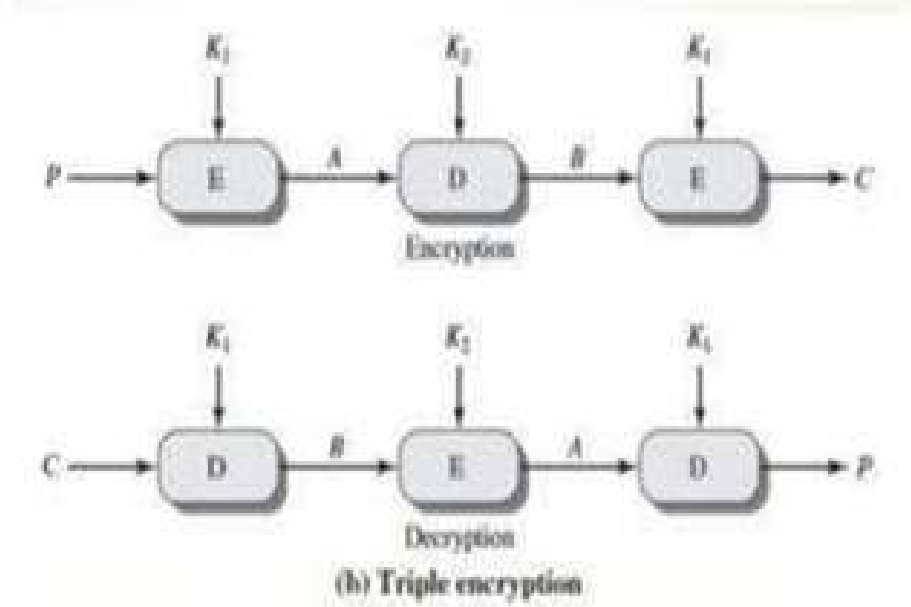
Triple-DES with Two-Keys

- Triple DES was developed in 1999 by IBM–by a team led by Walter Tuchman. DES prevents a meet-in- the-middle attack. 3-DES has a 168-bit key and enciphers blocks of 64 bits.
- hence must use 3 encryptions
  - would seem to need 3 distinct keys
- but can use 2 keys with E-D-E sequence
  - $C = E_{K1}(D_{K2}(E_{K1}(P)))$
  - nb encrypt & decrypt equivalent in security

  - if K1=K2 then can work with single DES

- standardized in ANSI X9.17 & ISO8732
- no current known practical attacks

Triple-DES with two keys is a popular alternative to single-DES, but suffers from being 3 times slower to run. The use of encryption & decryption stages are equivalent, but the chosen structure allows for compatibility with single-DES implementations. 3DES with two keys is a relatively popular alternative to DES and has been adopted for use in the key management standards ANS X9.17 and ISO 8732. Currently, there are no practical cryptanalytic attacks on 3DES. Coppersmith notes that the cost of a brute-force key search on 3DES is on the order of 2^112 (=5*10^33) and estimates that the cost of differential cryptanalysis suffers an exponential growth, compared to single DES, exceeding 10^52.



(b) Triple encryption

- The function follows an encrypt-decrypt-encrypt (EDE) sequence.

$$C = E(K_1, D(K_2, E(K_1, P)))$$

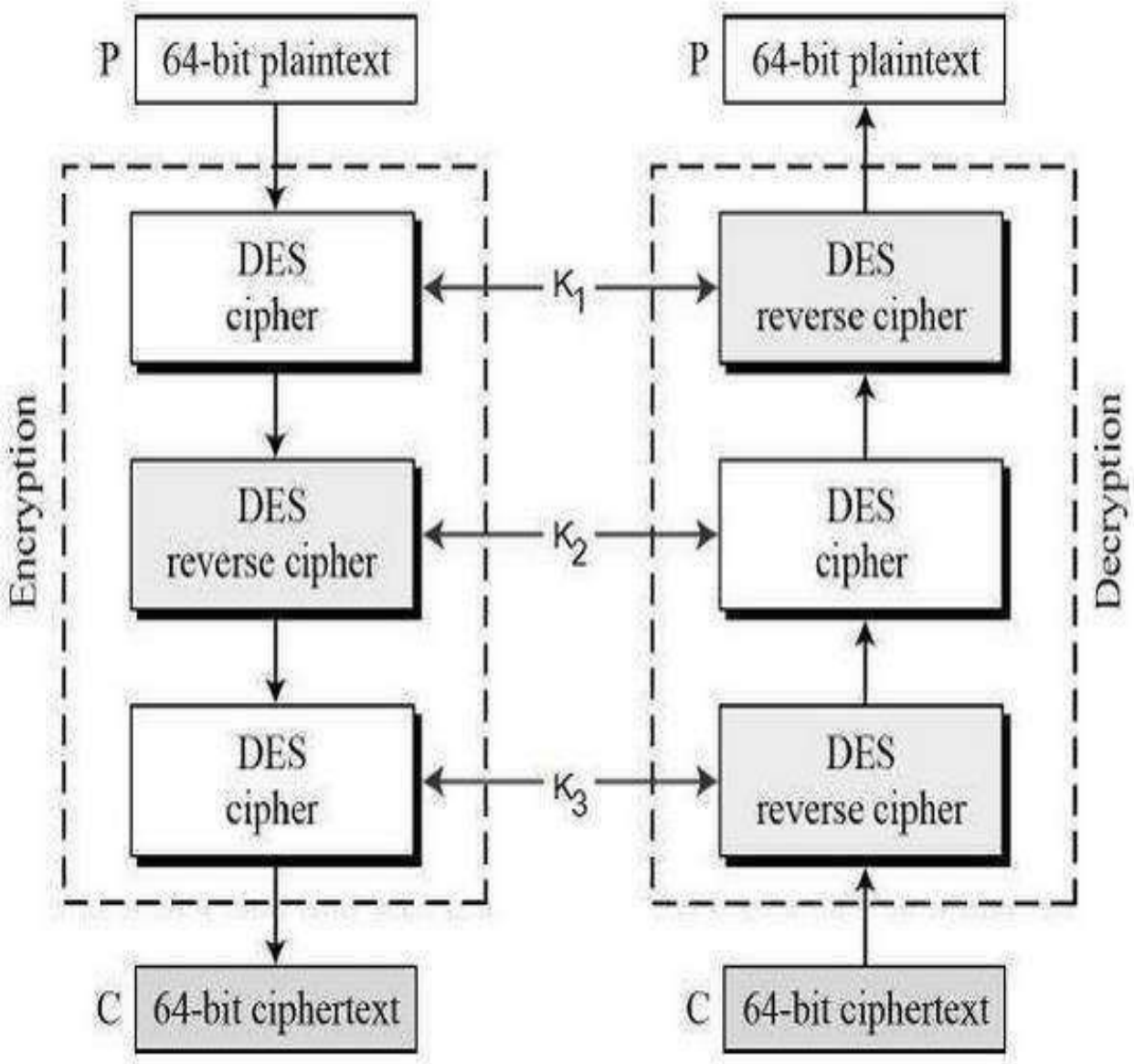
$$P = D(K_1, E(K_2, D(K_1, C)))$$

- By the use of triple DES with 2-key encryption, it raises the cost of meet-in-the-middle attack to 2^{112}.
- It has the drawback of requiring a key length of $56 \times 3 = 168$bits which may be somewhat unwieldy.

Triple-DES with Three-Keys

- although are no practical attacks on two-key Triple-DES have some indications
- can use Triple-DES with Three-Keys to avoid even these
  - $C = E_{K3}(D_{K2}(E_{K1}(P)))$
- has been adopted by some Internet applications, eg PGP, S/MIME

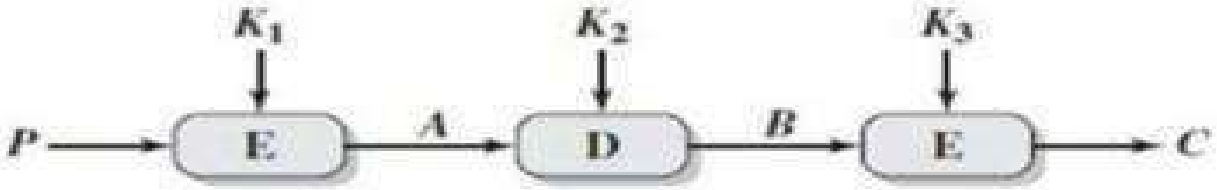
Have some indications of possible attacks on 2-key Triple-DES. Triple-DES with three keys (168-bits) is now being used in some applications, including PGP and S/MIME, for greater security.



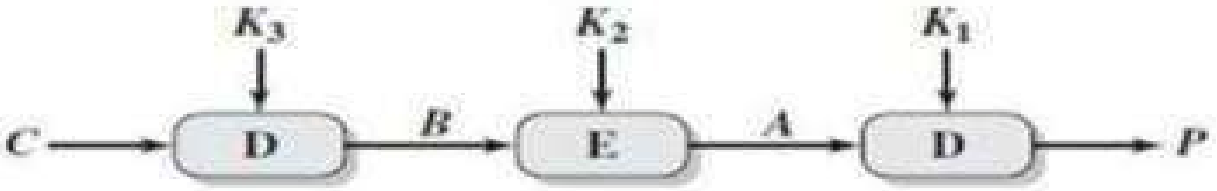
- Although the attacks just described appear impractical, anyone using two-key 3DES may feel some concern.
- Thus, many researches now feel that 3-key 3DES is the preferred alternative.
- Use three stages of DES for encryption and decryption with three different keys.
- 3-key 3DES has an effective key length of 168 bits and is defined as,

$$C = E(K_3, D(K_2, E(K_1, P)))$$

$$P = D(K_1, E(K_2, D(K_3, C)))$$



Encryption



Decryption

Advanced Encryption Standard (AES)

AES Origins, Requirements, Criteria and shortlists

The Advanced Encryption Standard (AES) was published by NIST (National Institute of Standards and Technology) in 2001. AES is a symmetric block cipher that is intended to replace DES as the approved standard for a wide range of applications. The AES cipher (& other candidates) form the latest generation of block ciphers, and now we see a significant increase in the block size - from the old standard of 64-bits up to 128-bits; and keys from 128 to 256-bits. In part this has been driven by the public demonstrations of exhaustive key searches of DES. Whilst triple-DES is regarded as secure and well understood, it is slow, especially in s/w. In a first round of evaluation, 15 proposed algorithms were accepted. A second round narrowed the field to 5 algorithms. NIST completed its evaluation process and published a final standard (FIPS PUB 197) in November of 2001. NIST selected Rijndael as the proposed AES algorithm. The two researchers who developed and submitted Rijndael for the AES are both cryptographers from Belgium: Dr. Joan Daemen and Dr.Vincent Rijmen.

Origin of AES

- A replacement for DES was needed
  - have theoretical attacks that can break it
  - have demonstrated exhaustive key search attacks
- can use Triple-DES – but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in Jun 98
- 5 were shortlisted in Aug-99
- Rijndael was selected as the AES in Oct-2000
- issued as FIPS PUB 197 standard in Nov-2001

AES Shortlist

The AES shortlist of 5 ciphers was as shown. Note mix of commercial (MARS, RC6, Twofish) verses academic (Rijndael, Serpent) proposals, sourced from various countries.

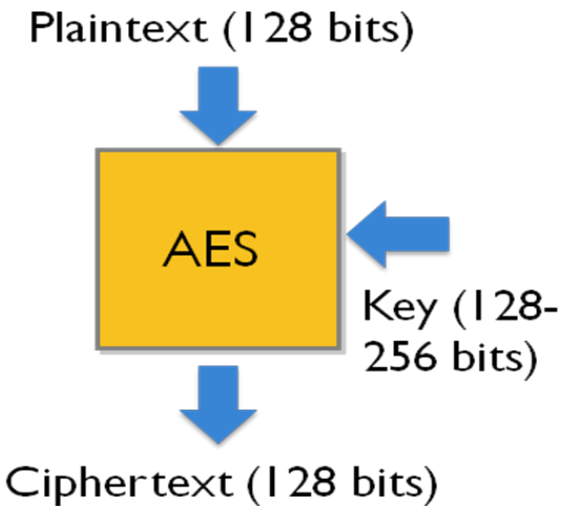
All were thought to be good – it came down to the best balance of attributes to meet criteria, in particular the balance between speed, security & flexibility.

- after testing and evaluation, shortlist in Aug-99:
  - MARS (IBM) - complex, fast, high security margin

  - RC6 (USA) - v. simple, v. fast, low security margin
  - Rijndael (Belgium) - clean, fast, good security margin
  - Serpent (Euro) - slow, clean, v. high security margin
  - Twofish (USA) - complex, v. fast, high security margin
- then subject to further analysis & comment
- saw contrast between algorithms with
  - few complex rounds verses many simple rounds
  - which refined existing ciphers verses new proposals

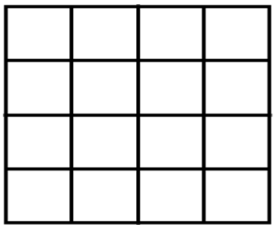
AES Cipher Top View

- Designed by Rijmen-Daemen in Belgium
- has 128/192/256 bit keys, 128 bit data
- an **iterative** cipher which processes data as block of 4 columns of 4 bytes
- operates on entire data block in every round

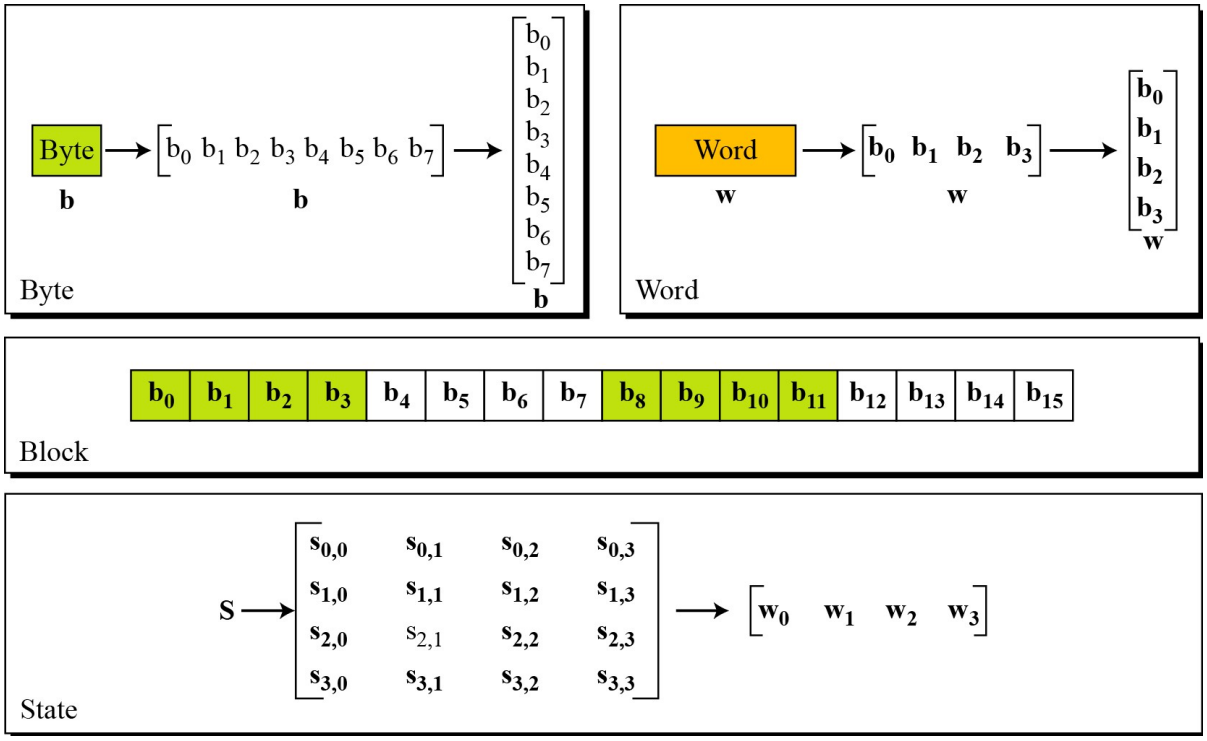


128-bit values

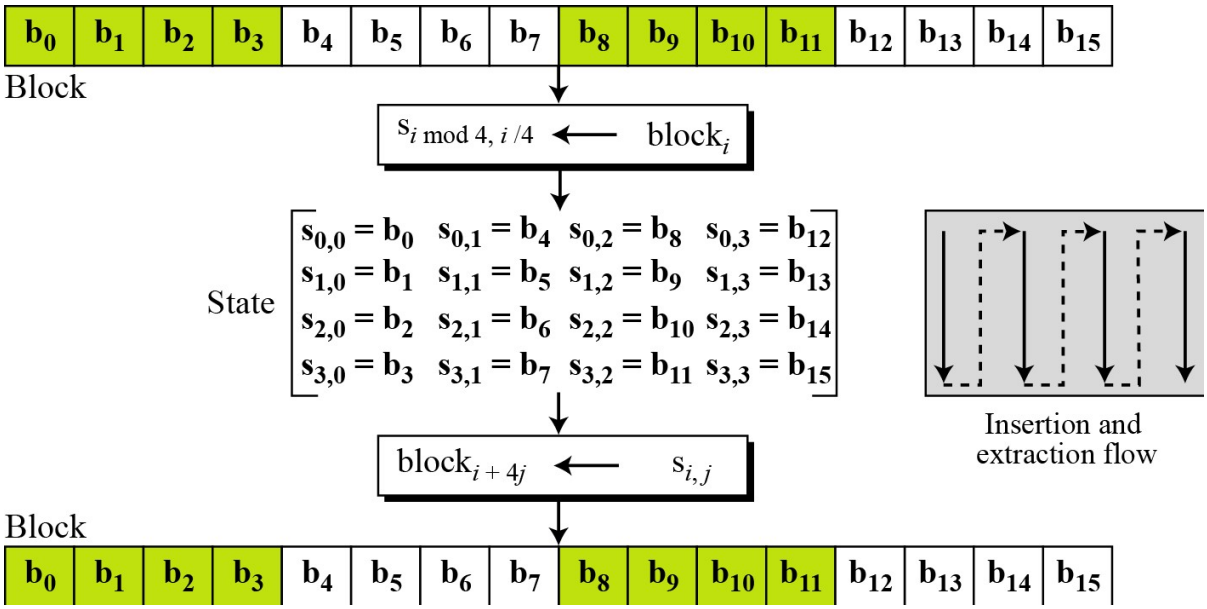
- Data block viewed as 4-by-4 table of bytes
- Represented as 4 by 4 matrix of 8-bit bytes.
- Key is expanded to array of 32 bits words



Data Unit



Unit Transformation

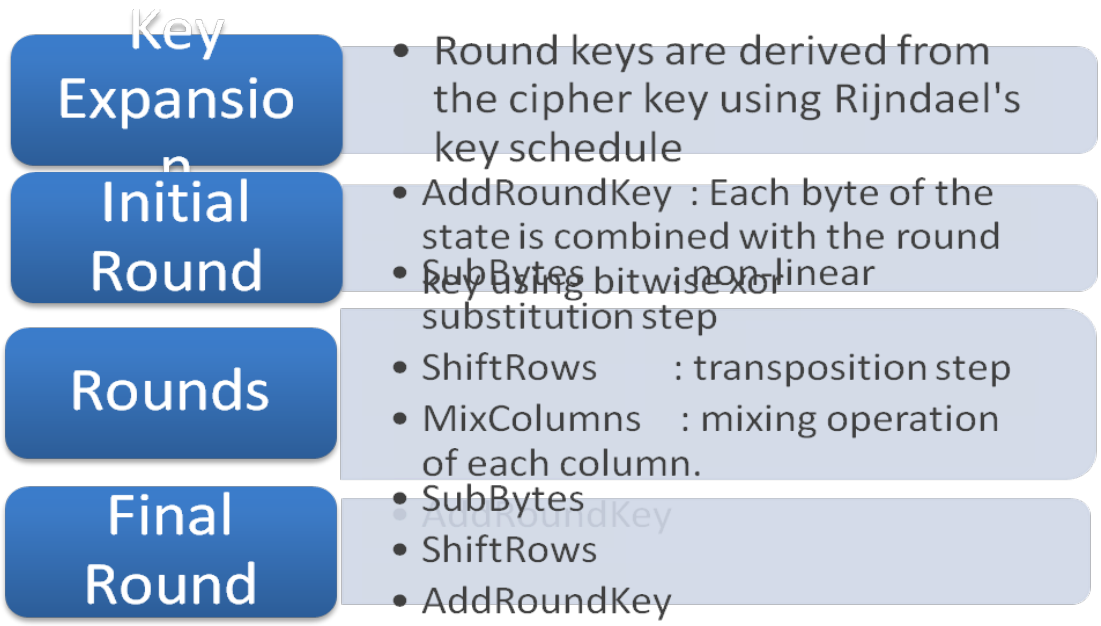


Changing Plaintext to State in AES

Text	A	E	S	U	S	E	S	A	M	A	T	R	I	X	Z	Z
Hexadecimal	00	04	12	14	12	04	12	00	0C	00	13	11	08	23	19	19

$$\begin{bmatrix} 00 & 12 & 0C & 08 \\ 04 & 04 & 00 & 23 \\ 12 & 12 & 13 & 19 \\ 14 & 00 & 11 & 19 \end{bmatrix}$$ State

AES High Level Description



AES Structure

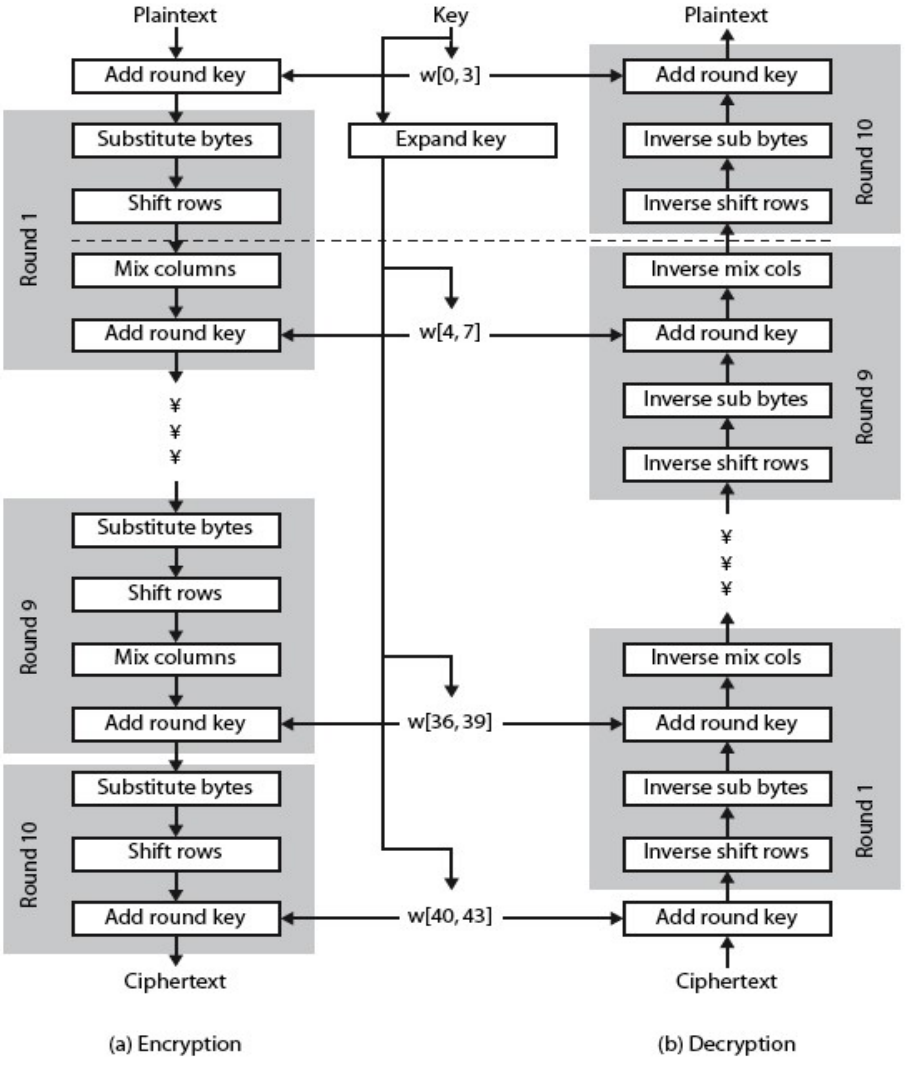
The input to the AES encryption and decryption algorithms is a single 128-bit block, depicted in FIPS PUB 197, as a square matrix of bytes .This block is copied into the State array, which is modified at each stage of encryption or decryption. After the final stage, State is copied to an output.

The key is expanded into 44/52/60 lots of 32-bit words (see later), with 4 used in each round.

The data computation then consists of an "add round key" step, then 9/11/13 rounds with all 4 steps, and a final $10^{th}/12^{th}/14^{th}$ step of byte subs + mix cols + add round key. This can be viewed as alternating XOR key & scramble data bytes operations. All of the steps are easily reversed, and can be efficiently implemented using XOR's & table lookups.

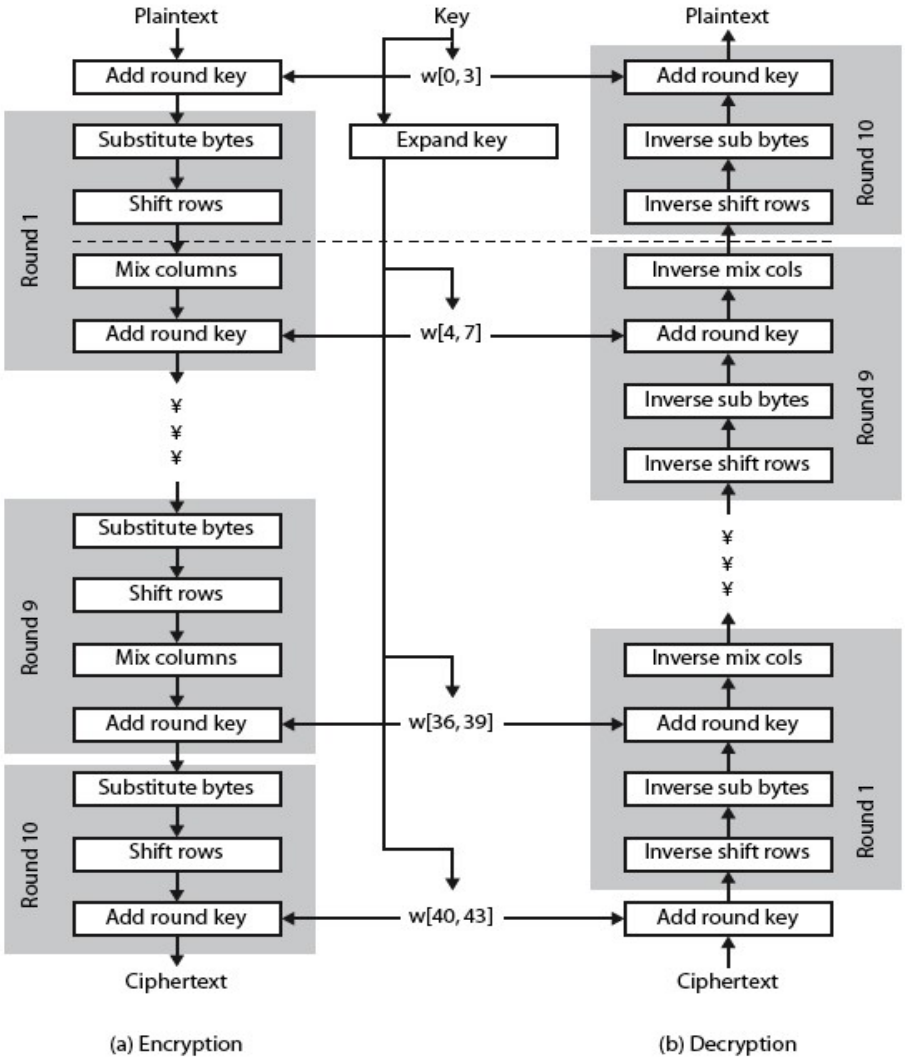
- data block of 4 columns of 4 bytes is state

- key is expanded to array of words
- has 10/12/14 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multipy of groups)
  - add round key (XOR state with key material)
  - view as alternating XOR key & scramble data bytes
- initial XOR key material & incomplete last round
- with fast XOR & table lookup implementation



Advanced Encryption Standard (AES)

AES Structure



The input to the AES encryption and decryption algorithms is a single 128-bit block, depicted in FIPS PUB 197, as a square matrix of bytes .This block is copied into the State array, which is modified at each stage of encryption or decryption. After the final stage, State is copied to an output.

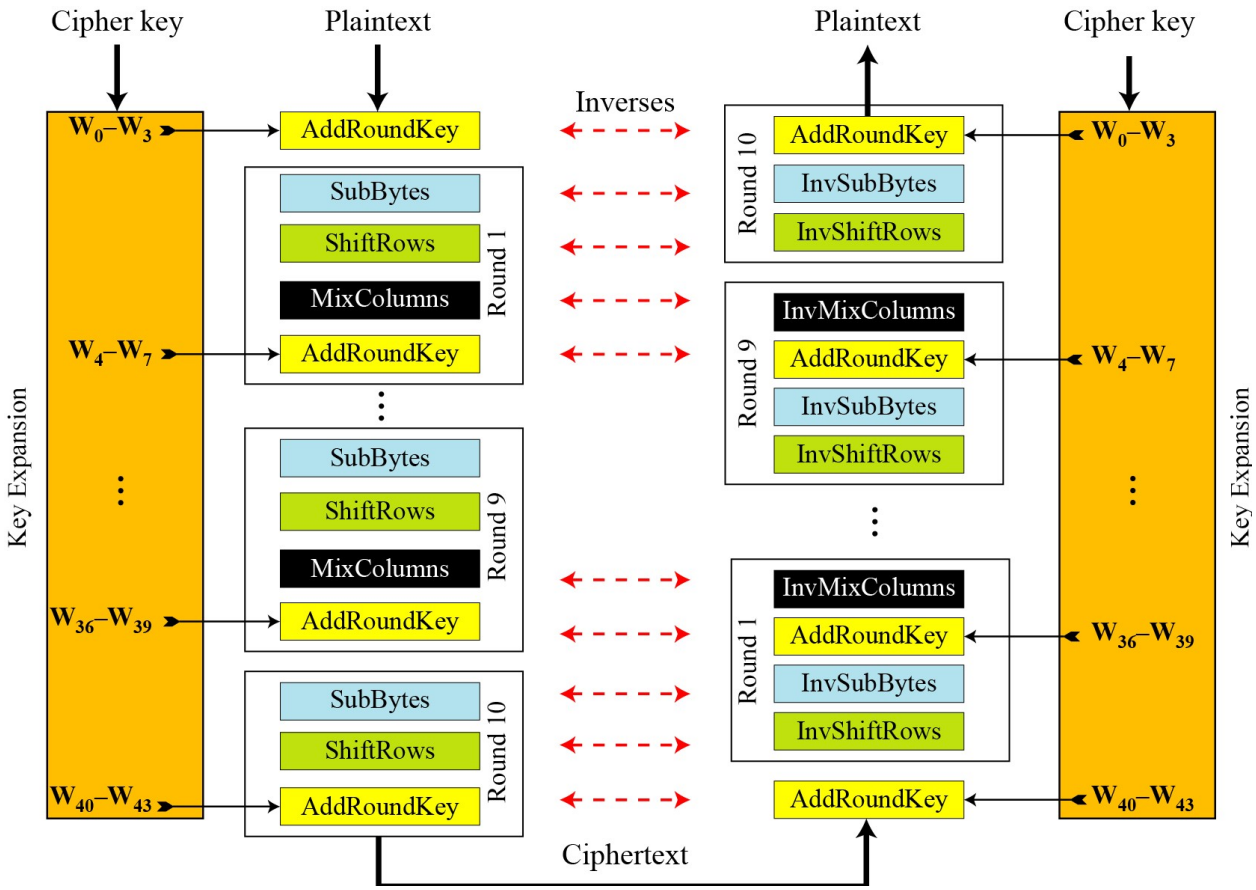
The key is expanded into 44/52/60 lots of 32-bit words (see later), with 4 used in each round.

The data computation then consists of an "add round key" step, then 9/11/13 rounds with all 4 steps, and a final $10^{th}/12^{th}/14^{th}$ step of byte subs + mix cols + add round key. This can be viewed as alternating XOR key & scramble data bytes operations. All of the steps are easily reversed, and can be efficiently implemented using XOR's & table lookups.

- data block of 4 columns of 4 bytes is state

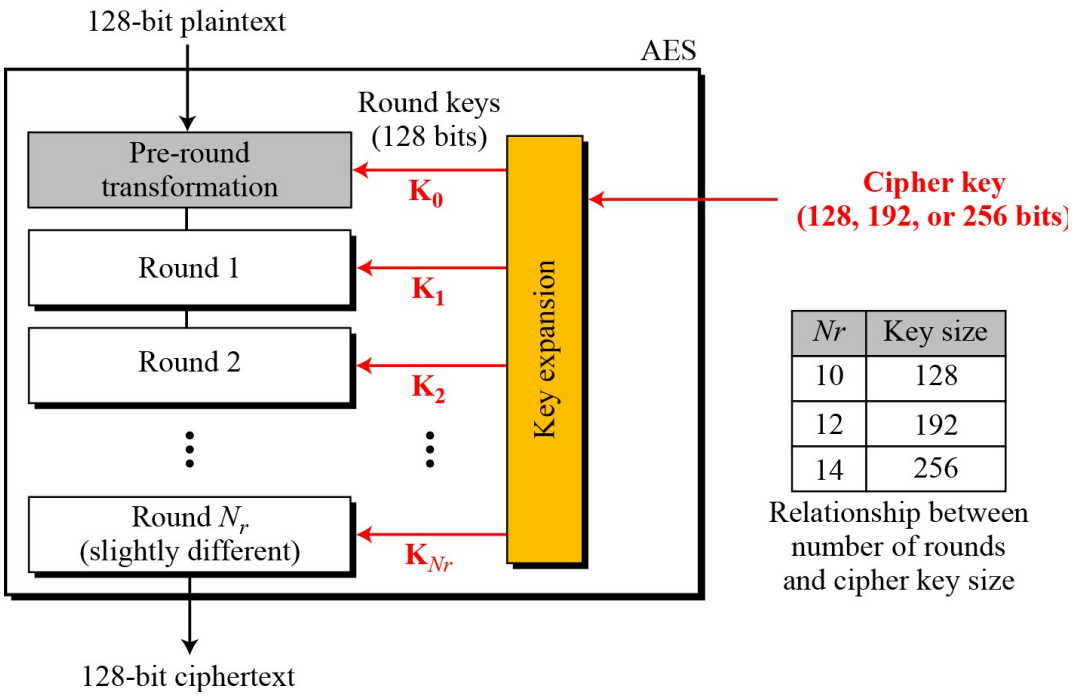
- key is expanded to array of words
- has 10/12/14 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multipy of groups)
  - add round key (XOR state with key material)
  - view as alternating XOR key & scramble data bytes
- initial XOR key material & incomplete last round
- with fast XOR & table lookup implementation

AES Working Structure



Multiple rounds

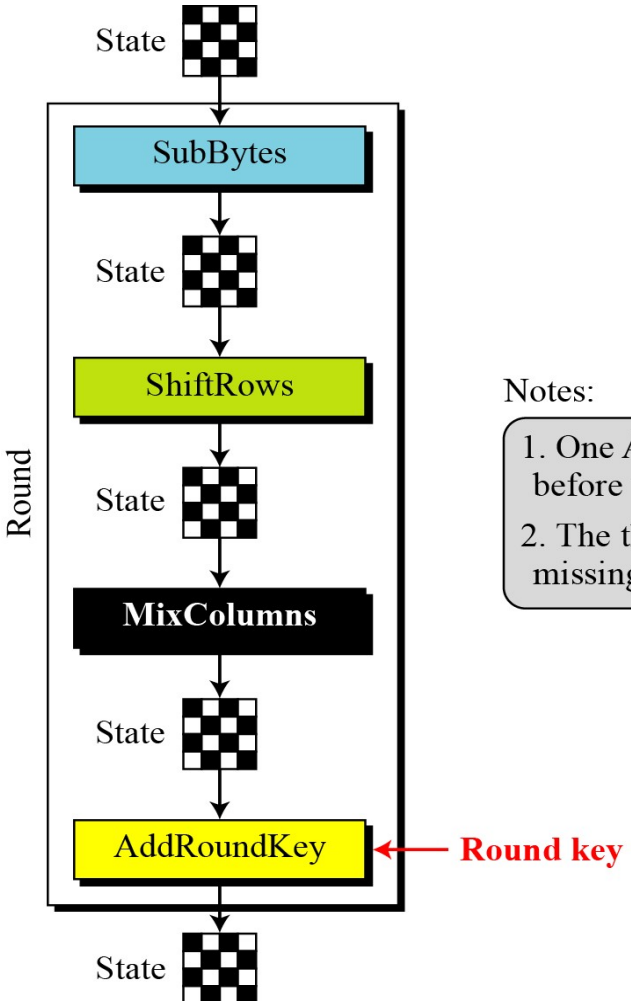
- Rounds are (almost) identical
  - First and last round are a little different



Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Details of Each Round



Notes:

1. One AddRoundKey is applied before the first round.
2. The third transformation is missing in the last round.

Byte Substitution

SubBytes table

		y														
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
x	**0**	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Inverse SubBytes table

		y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
x	**0**	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
	1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
	2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
	3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
	4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
	5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
	6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
	7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
	8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
	9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
	A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
	B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
	C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
	D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
	E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
	F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Now discuss each of the four stages used in AES. The Substitute bytes stage uses an S-box to perform a byte-by-byte substitution of the block. There is a single 8-bit wide S-box used on every byte. This S-box is a permutation of all 256 8-bit values, constructed using a transformation which treats the values as polynomials in GF(2^8) – however it is fixed, so

really only need to know the following table when implementing. Decryption requires the inverse of the table.

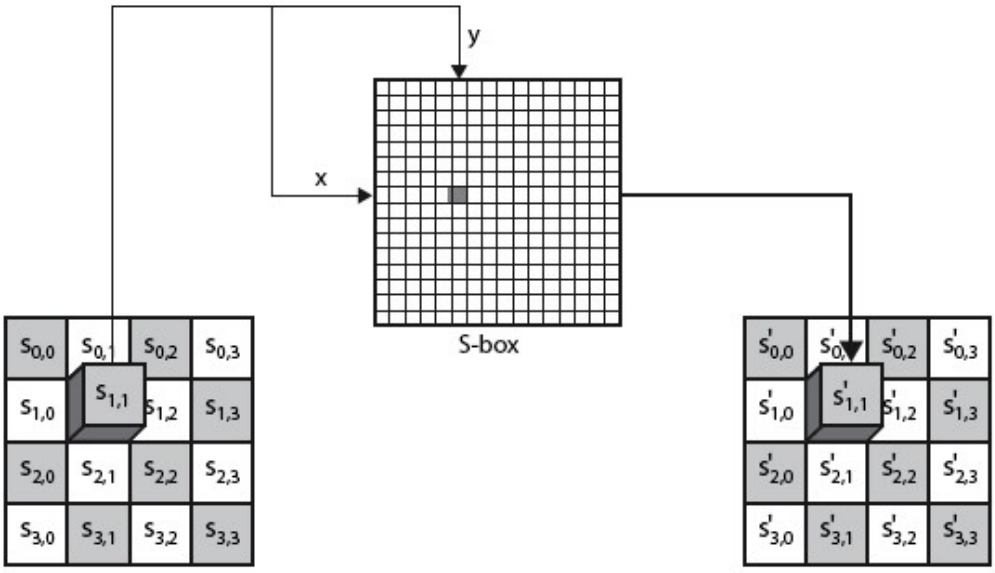
The table was designed to be resistant to known cryptanalytic attacks. Specifically, the Rijndael developers sought a design that has a low correlation between input bits and output bits, with the property that the output cannot be described as a simple mathematical function of the input, with no fixed points and no “opposite fixed points”.

- a simple substitution of each byte
  - provide a confusion
- uses one table of 16x16 bytes containing a permutation of all 256 8-bit values
- each byte of state is replaced by byte indexed by row (left 4-bits) & column (right 4-bits)
  - eg. byte {95} is replaced by byte in row 9 column 5
  - which has value {2A}

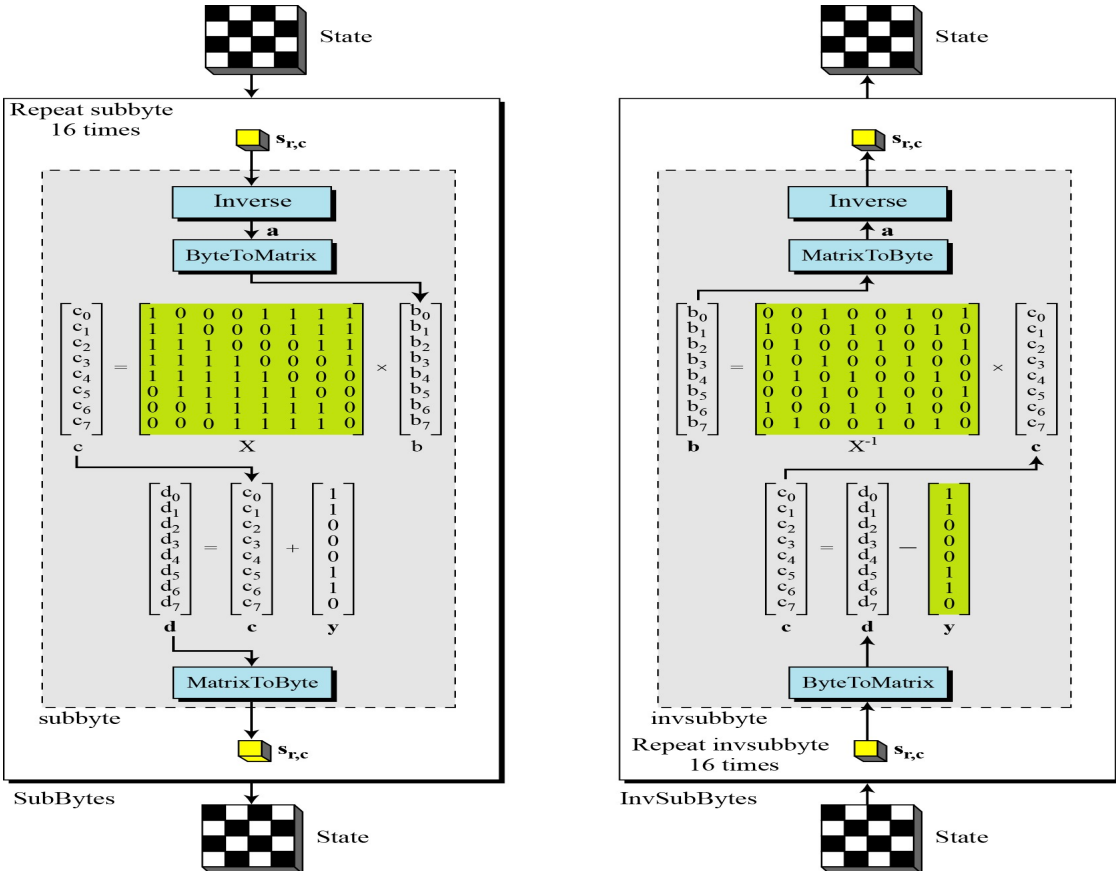
- S-box constructed using defined transformation of values in GF(2^8) designed to be resistant to all known attacks

SubBytes Operation

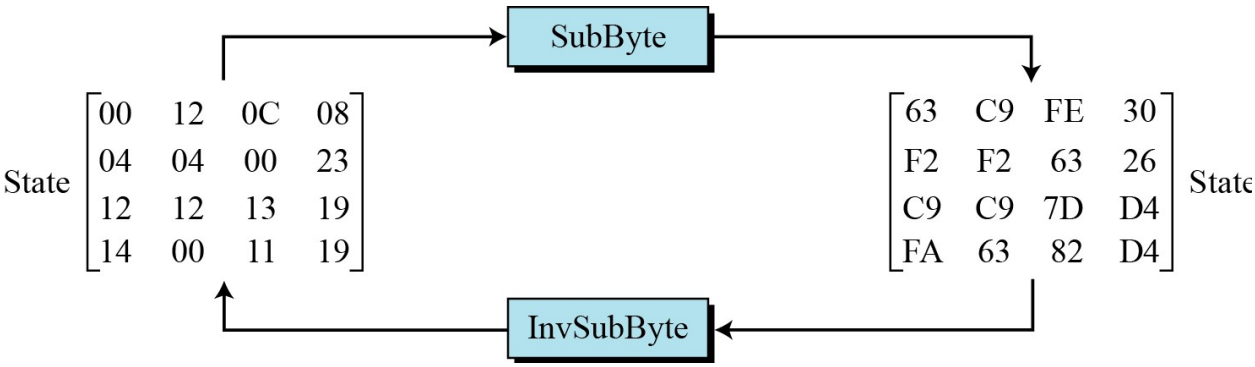
The SubBytes operation involves 16 independent byte-to-byte transformations



SubBytes and InvSubBytes

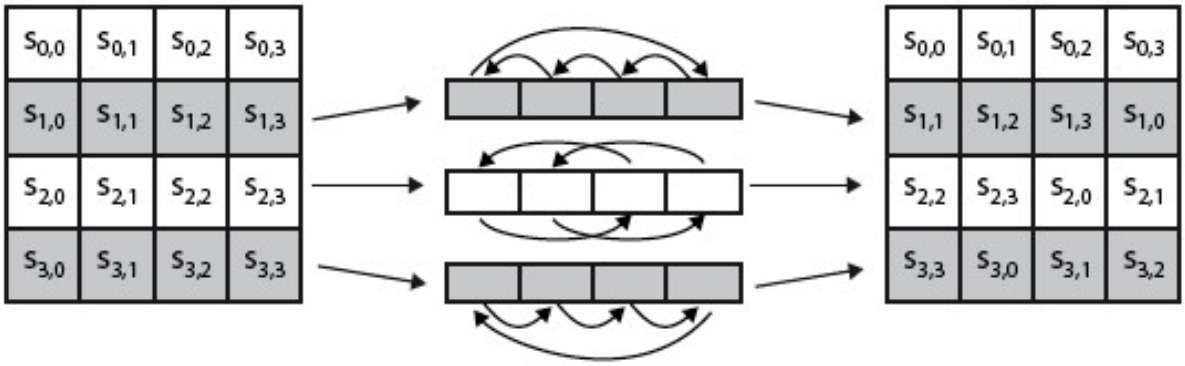


Sample SubByte Transformation

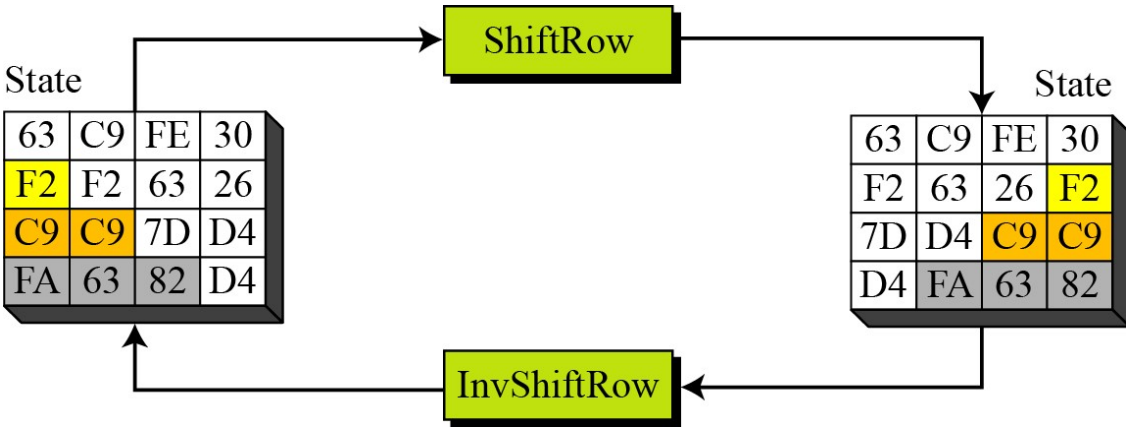


ShiftRows

The ShiftRows stage provides a simple “permutation” of the data, whereas the other steps involve substitutions. Further, since the state is treated as a block of columns, it is this step which provides for diffusion of values between columns. It performs a circular rotate on each row of 0, 1, 2 & 3 places for respective rows. When decrypting it performs the circular shifts in the opposite direction for each row. This row shift moves an individual byte from one column to another, which is a linear distance of a multiple of 4 bytes, and ensures that the 4 bytes of one column are spread out to four different columns.



- Shifting, which permutes the bytes.
- A circular byte shift in each each
  - 1st row is unchanged
  - 2nd row does 1 byte circular shift to left
  - 3rd row does 2 byte circular shift to left
  - 4th row does 3 byte circular shift to left
- In the encryption, the transformation is called ShiftRows
- In the decryption, the transformation is called InvShiftRows and the shifting is to the right



MixColumns

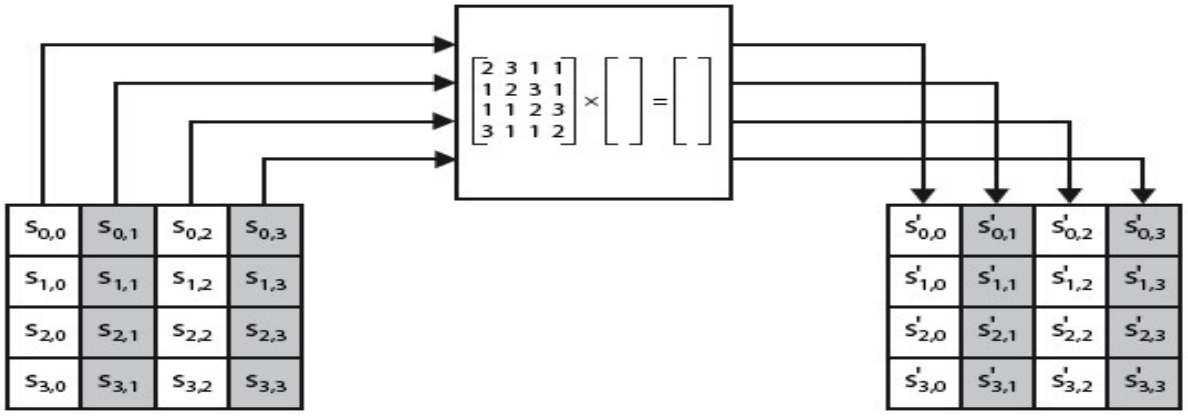
The MixColumns stage is a substitution that makes use of arithmetic over GF(2^8). Each byte of a column is mapped into a new value that is a function of all four bytes in that column. It is designed as a matrix multiplication where each byte is treated as a polynomial in GF(2^8). The inverse used for decryption involves a different set of constants.

The constants used are based on a linear code with maximal distance between code words – this gives good mixing of the bytes within each column. Combined with the "shift rows" step provides good avalanche, so that within a few rounds, all output bits depend on all input bits.

- ShiftRows and MixColumns provide diffusion to the cipher
- Each column is processed separately
- Each byte is replaced by a value dependent on all 4 bytes in the column
- Effectively a matrix multiplication in GF(2^8) using prime poly m(x) =$x^8+x^4+x^3+x+1$

$$\begin{matrix} ax + by + cz + dt \\ ex + fy + gz + ht \\ ix + jy + kz + lt \\ mx + ny + oz + pt \end{matrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix}$$

New matrix **Constant matrix** Old matrix

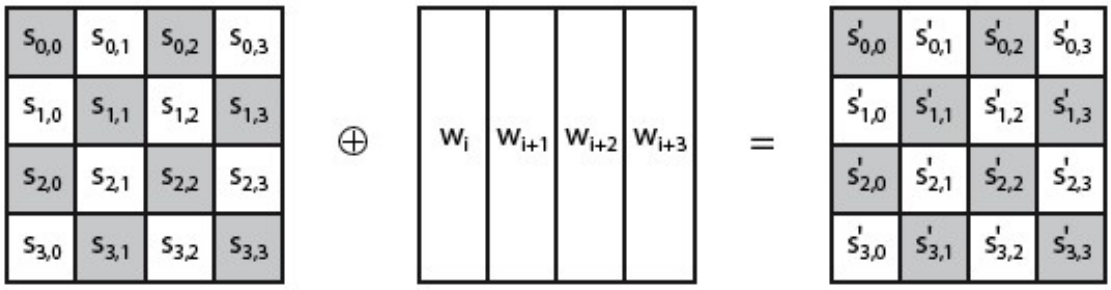


- can express each col as 4 equations
  - to derive each new byte in column
- decryption requires use of inverse matrix
  - with larger coefficients, hence a little harder
- have an alternate characterization
  - each column a 4-term polynomial
  - with coefficients in GF(2^8)
  - and polynomials multiplied modulo (x^4+1)

Add Round Key

Lastly is the Add Round Key stage which is a simple bitwise XOR of the current block with a portion of the expanded key. Note this is the only step which makes use of the key and obscures the result, hence MUST be used at start and end of each round, since otherwise could undo effect of other steps. But the other steps provide confusion/diffusion/non-linearity. That us you can look at the cipher as a series of XOR with key then scramble/permute block repeated. This is efficient and highly secure it is believed.

- XOR state with 128-bits of the round key
- AddRoundKey proceeds one column at a time.
  - adds a round key word with each state column matrix
  - the operation is matrix addition
- Inverse for decryption identical
  - since XOR own inverse, with reversed keys
- designed to be as simple as possible
  - a form of Vernam cipher on expanded key
  - requires other stages for complexity / security

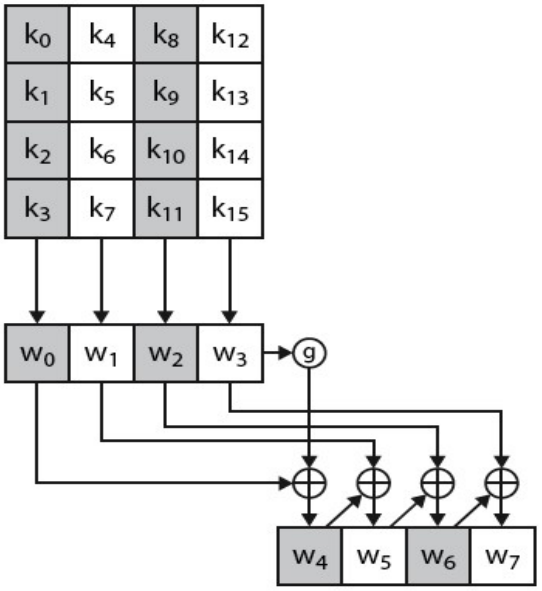


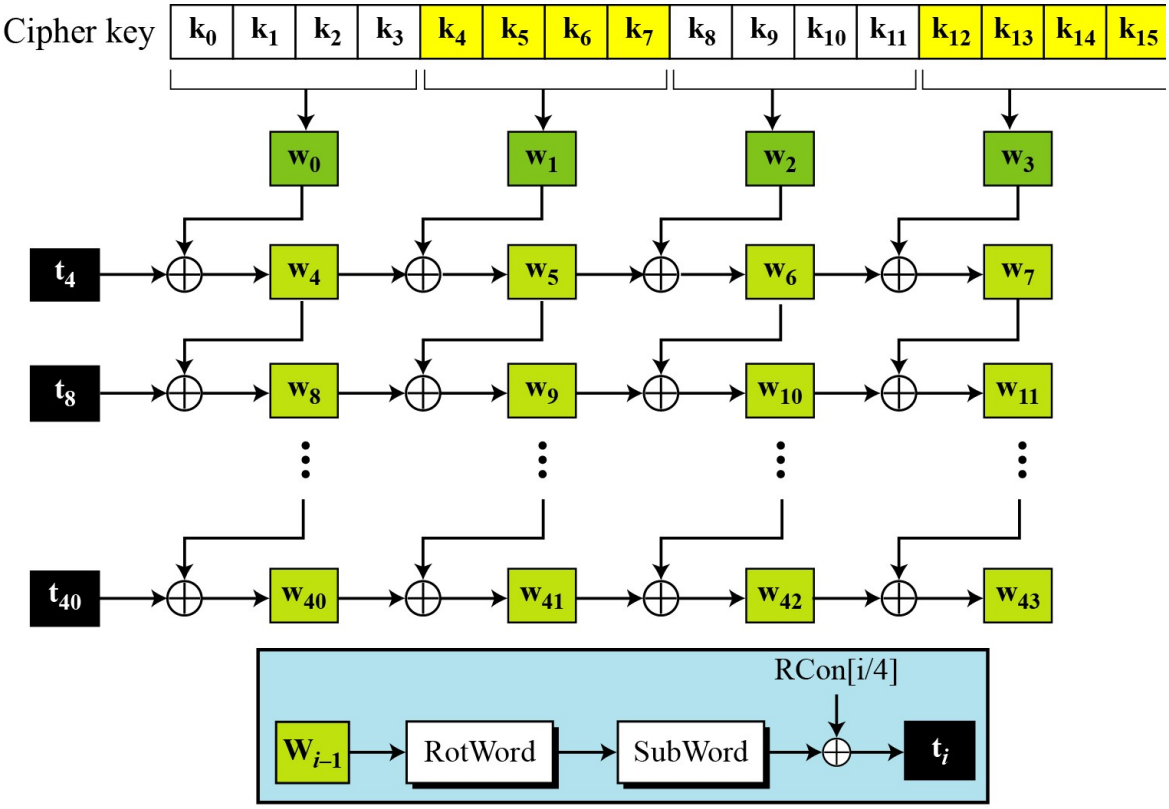
AES Key Expansion (Key expansion Scheme)

The AES key expansion algorithm takes as input a 4-word (16-byte) key and produces a linear array of words, providing a 4-word round key for the initial AddRoundKey stage and each of the 10/12/14 rounds of the cipher. It involves copying the key into the first group of 4 words, and then constructing subsequent groups of 4 based on the values of the previous & 4th back words. The first word in each group of 4 gets “special treatment” with rotate + S-box + XOR constant on the previous word before XOR’ing the one from 4 back. In the 256-bit key/14 round version, there’s also an extra step on the middle word.

- takes 128-bit (16-byte) key and expands into array of 44/52/60 32-bit words

- start by copying key into first 4 words
- then loop creating words that depend on values in previous 4 places back
  - in 3 of 4 cases just XOR these together
  - 1st word in 4 has rotate + S-box + XOR round constant on previous, before XOR 4th back





Making of t_i (temporary) words $i = 4\ N_r$

Key Expansion submodule

- **RotWord** performs a one byte circular left shift on a word For example:

  RotWord[b0,b1,b2,b3] = [b1,b2,b3,b0]

- **SubWord** performs a byte substitution on each byte of input word using the S-box
- **SubWord(RotWord(temp))** is XORed with RCon[j] – the round constant

AES Key Scheduling

- takes 128-bits (16-bytes) key and expands into array of 44 32-bit words

Round	*Words*			
Pre-round	$\mathbf{w}_0$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_3$
1	$\mathbf{w}_4$	$\mathbf{w}_5$	$\mathbf{w}_6$	$\mathbf{w}_7$
2	$\mathbf{w}_8$	$\mathbf{w}_9$	$\mathbf{w}_{10}$	$\mathbf{w}_{11}$
. . .	. . .			
N_r	$\mathbf{w}_{4N_r}$	$\mathbf{w}_{4N_r+1}$	$\mathbf{w}_{4N_r+2}$	$\mathbf{w}_{4N_r+3}$

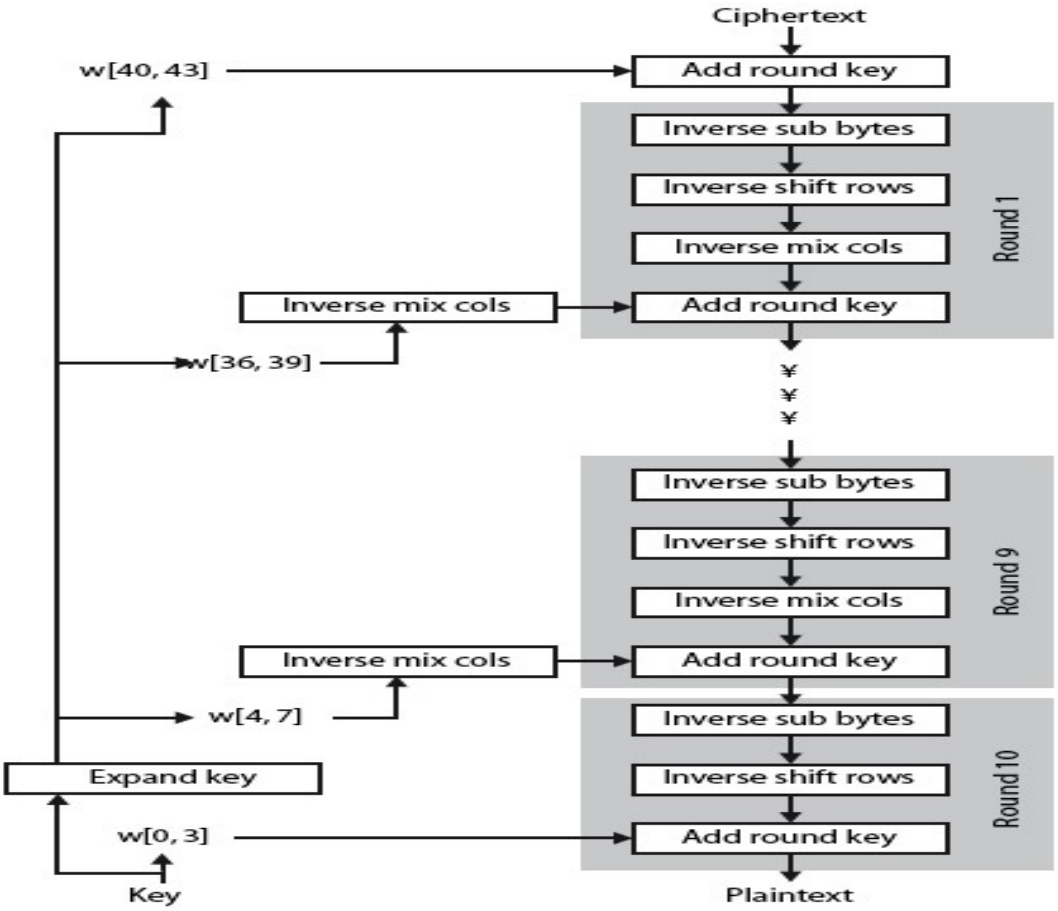
AES Decryption

The AES decryption cipher is not identical to the encryption cipher. The sequence of transformations for decryption differs from that for encryption, although the form of the key schedules for encryption and decryption is the same. This has the disadvantage that two separate software or firmware modules are needed for applications that require both encryption and decryption. There is, however, an equivalent version of the decryption algorithm that has the same structure as the encryption algorithm, with the same sequence of transformations as the encryption algorithm (with transformations replaced by their inverses). To achieve this equivalence, a change in key schedule is needed.

By constructing an equivalent inverse cipher with steps in same order as for encryption, we can derive a more efficient implementation. Clearly swapping the byte substitutions and shift rows has no effect, since work just on bytes. Swapping the mix columns and add round key steps requires the inverse mix columns step be applied to the round keys first – this makes the decryption key schedule a little more complex with this construction, but allows the use of same h/w or s/w for the data en/decrypt computation.

- AES decryption is not identical to encryption since steps done in reverse

- but can define an equivalent inverse cipher with steps as for encryption
  - but using inverses of each step
  - with a different key schedule
- works since result is unchanged when
  - swap byte substitution & shift rows
  - swap mix columns & add (tweaked) round key



Analysis & Security of DES & AES

Concepts of the Analysis & Security of DES & AES

- DES Avalanche & completeness Effect
- DES Weakness analysis
- Strength of DES
- DES Security
- DES replacement
- Strength of AES
- AES Evaluation criteria
- AES Security

DES Avalanche & completeness Effect

- A change of **one** input or key bit results in changing more than **half** output bits
- DES exhibits strong avalanche
- Completeness is w.r.t diffusion & confusion.

EXAMPLE 6.7 To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the number of bits in each round.

Plaintext: 0000000000000000	Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1	
Plaintext: 0000000000000001	Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3	

DES Weakness analysis

- **Weakness in Cipher Design:** It is not clear why the designers of DES used the initial and final permutations; these have no security benefits.
- **Weakness in Cipher Key:** DES Key size is 56 bits. To do Brute force attack on a given cipher text block, the adversary needs to check 2^{56} keys. With available technology it is possible to check 1 million keys per second

Strength of DES – Key Size

DES finally and definitively proved insecure in July 1998, when the Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose "DES cracker" machine that was built for less than $250,000. The attack took less than three days. The

EFF has published a detailed description of the machine, enabling others to build their own cracker [EFF98].

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- Brute force search looks hard
- Recent advances have shown is possible
  - in 1997 on a huge cluster of computers over the Internet in a few months
  - in 1998 on dedicated hardware called “DES cracker” by EFF in a few days ($220,000)
  - in 1999 above combined in 22hrs!
- Still must be able to recognize plaintext
- No big flaw for DES algorithms

Strength of DES – Security Analytic Attacks

Another concern is the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm. The focus of concern has been on the eight substitution tables, or S-boxes, that are used in each iteration. These techniques utilise some deep structure of the cipher by gathering information about encryptions so that eventually you can recover some/all of the sub-key bits, and then exhaustively search for the rest if necessary. Generally these are statistical attacks which depend on the amount of information gathered for their likelihood of success. Attacks of this form include differential cryptanalysis. linear cryptanalysis, and related key attacks.

- Now have several analytic attacks on DES
- these utilise some deep structure of the cipher
  - by gathering information about encryptions
  - can eventually recover some/all of the sub-key bits

  - if necessary then exhaustively search for the rest
- generally these are statistical attacks include
  - differential cryptanalysis
  - linear cryptanalysis
  - related key attacks

Strength of DES – Security Timing Attacks

A timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts. A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs. The AES analysis process has highlighted this attack approach, and showed that it is a concern particularly with smartcard implementations, though DES appears to be fairly resistant to a successful timing attack.

- attacks actual implementation of cipher
- use knowledge of consequences of implementation to derive information about some/all subkey bits
- specifically use fact that calculations can take varying times depending on the value of the inputs to it
- particularly problematic on smartcards

DES Replacement

The AES candidates are the latest generation of block ciphers, and now we see a significant increase in the block size - from the old standard of 64-bits up to 128-bits; and keys from 128 to 256-bits. In part this has been driven by the public demonstrations of exhaustive key searches of DES. Whilst triple-DES is regarded as secure and well understood, it is slow, especially in s/w.

- Triple-DES (3DES)
  - 168-bit key, no brute force attacks
  - Underlying encryption algorithm the same, no effective analytic attacks
  - Drawbacks

- Performance: no efficient software codes for DES/3DES
- Efficiency/security: bigger block size desirable

- Advanced Encryption Standards (AES)
  - US NIST issued call for ciphers in 1997
  - Rijndael was selected as the AES in Oct-2000

AES

- Secret key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger & faster than Triple-DES
- Active life of 20-30 years
- Provide full specification & design details
- Evaluation criteria
  - Security: effort to practically cryptanalysis
  - Cost: computational efficiency and memory requirement
  - Implementation characteristics: flexibility to apps, hardware/software suitability & simplicity

Strength of AES

The Rijndael proposal for AES defined a cipher in which the block length and the key length can be independently specified to be 128,192,or 256 bits. The AES specification uses the same three key size alternatives but limits the block length to 128 bits. Rijndael is an academic submission, based on the earlier Square cipher, from Belgium academics Dr Joan Daemen and Dr Vincent Rijmen. It is an iterative cipher (operates on entire data block in every round) rather than feistel (operate on halves at a time), and was designed to have characteristics of: Resistance against all known attacks, Speed and code compactness on a wide range of platforms, & Design simplicity.

- Rijndael was selected as the AES in Oct-2000
- The algorithms used in AES are so simple that they can be easily implemented using cheap processors and a minimum amount of memory.

- Very efficient
- Implementation was a key factor in its selection as the AES cipher
- Rijndael design:
  - simplicity
  - has 128/192/256 bit keys, 128 bits data
  - resistant against known attacks
  - speed and code compactness on many CPUs

AES Evaluation Criteria

In fact, two set of criteria evolved. When NIST issued its original request for candidate algorithm nominations in 1997, the request stated that candidate algorithms would be compared based on few factors, which were used to evaluate field of 15 candidates to select shortlist of 5. These had categories of security, cost, and algorithm & implementation characteristics.

The final criteria evolved during the evaluation process, and were used to select Rijndael from that short-list, and more details are given in Stallings Table 5.2, with categories of: general security, ease of software & hardware implementation, implementation attacks, & flexibility (in en/decrypt, keying, other factors).

- initial criteria:
  - security – effort for practical cryptanalysis
  - cost – in terms of computational efficiency
  - algorithm & implementation characteristics
- final criteria
  - general security
  - ease of software & hardware implementation
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)

AES Security

- AES was designed after DES to be :
  - resistant against known attacks
  - speed and code compactness on many CPUs
  - design simplicity
- Most of the known attacks on DES were already tested on AES.
- Brute-Force Attack
  - AES is definitely more secure than DES due to the larger-size key.
- Statistical Attacks
  - Numerous tests have failed to do statistical analysis of the ciphertext
- Differential and Linear Attacks
  - There are no differential and linear attacks on AES as yet.