UNIT–2

*Symmetric Key Ciphers: Block Cipher Principles and Algorithms (DES, AES, and Blowfish),Differential and Linear Cryptanalysis, Block Cipher Modes of Operations, Stream Ciphers,RC4,LocationandPlacementof encryptionfunction,KeyDistribution.*

Asymmetric Key Ciphers: *Principles of Public Key Cryptosystems, Algorithms (RSA, Diffie-Hellman,ECC), KeyDistribution.*

CONVENTIONALENCRYPTIONPRINCIPLES

AConventional/Symmetricencryptionschemehasfiveingredients

1. ***PlainText***:Thisistheoriginalmessageordatawhichisfedintothealgorithmasinput.

2. ***Encryption Algorithm***: This encryption algorithm performs various substitutions andtransformationsonthe plain text.

3. ***SecretKey***:Thekeyisanotherinputtothealgorithm.Thesubstitutionsandtransformationsperformed byalgorithm dependonthe key.

4. ***CipherText:***Thisisthescrambled(unreadable)messagewhichisoutputoftheencryptionalgorithm. This cipher text is dependent on plaintext and secret key. For a given plaintext,twodifferentkeysproduce two differentcipher texts.

5. ***DecryptionAlgorithm***:Thisisthereverseofencryptionalgorithm.Ittakestheciphertextandsecretkeyasinputsandoutputsthe plain text.

Simplified Model of Conventional Encryption
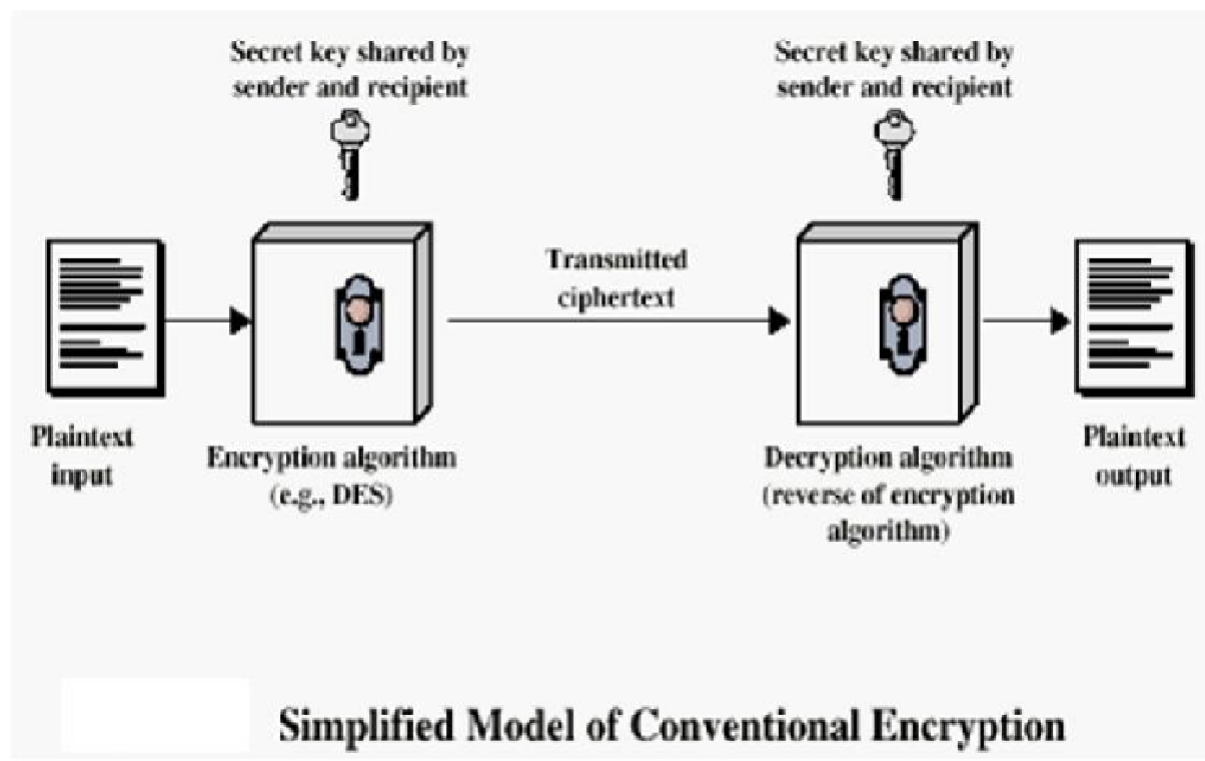
The important point is that the security of conventional encryption depends on the secrecyof the key, not the secrecy of the algorithm i.e. it is not necessary to keep the algorithmsecret, but only the key is to be kept secret. This feature that algorithm need not be keptsecret made it feasible for wide spread use and enabled manufacturers develop low costchip implementation of data encryption algorithms. With the use of conventional algorithm,theprincipalsecurityproblemismaintainingthesecrecyofthe key.

## FEISTELCIPHERSTRUCTURE

The input to the encryption algorithm are a plaintext block of length 2w bits anda key K. the plaintext block is divided into two halves L_0 and R_0. The two halves of thedata pass through „n" rounds of processing and then combine to produce the ciphertextblock. Each round „i" has inputs L_{i-1} and R_{i-1}, derived from the previous round, as well asthesubkeyK_i,derivedfromtheoverallkeyK.ingeneral,thesubkeys$K_i$aredifferentfromKandfromeachother.

Allrounds have the same structure.Asubstitution is performedon the lefthalfof thedata(assimilartoS-DES).ThisisdonebyapplyingaroundfunctionFtotherighthalf

of the data and then taking the XOR of the output of that function and the left half of thedata.Theroundfunctionhasthesamegeneralstructureforeachroundbutisparameterized by the round subkey k_i. Following this substitution, a permutation isperformedthatconsistsoftheinterchangeofthetwohalvesofthedata.Thisstructureisa particular form of the substitution-permutation network. The exact realization of aFeistelnetworkdependsonthechoiceofthefollowingparametersanddesignfeatures:

- **Blocksize**-Increasingsizeimprovessecurity,butslowscipher
- **Keysize**-

  Increasingsizeimprovessecurity,makesexhaustivekeysearchingharder,butmayslow cipher

- **Numberofrounds**-Increasingnumberimprovessecurity,butslows cipher
- **Subkeygeneration**-Greatercomplexitycanmakeanalysisharder,butslowscipher
- **Roundfunction**-Greatercomplexitycanmakeanalysisharder,butslowscipher
- **Fast software en/decryption & ease of analysis -** are more recent concerns forpracticaluseandtesting
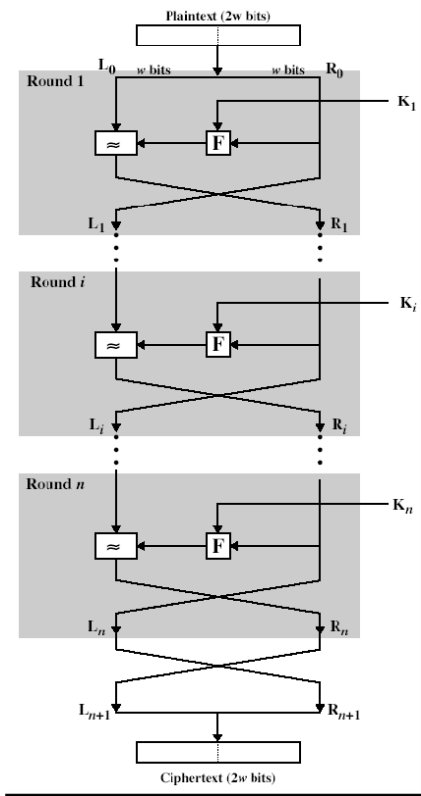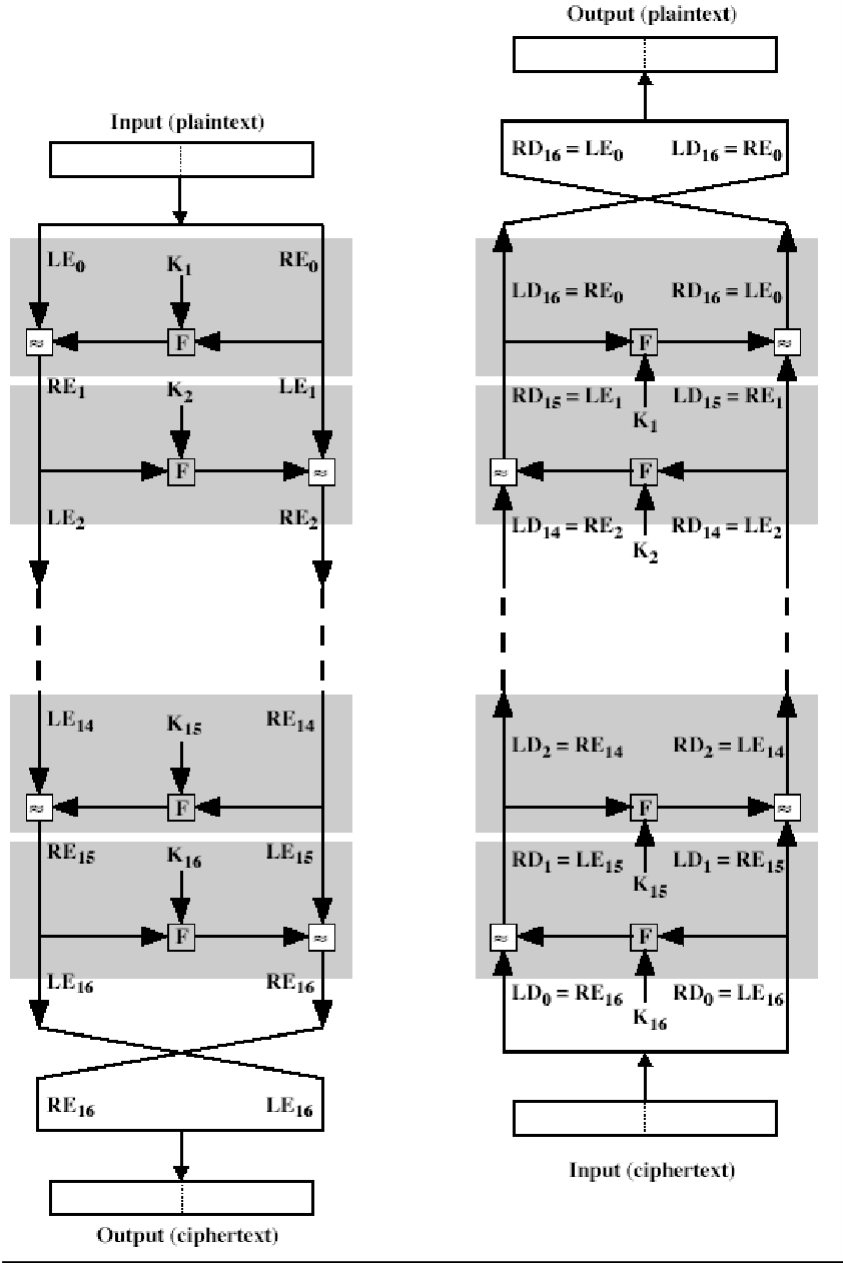
The process of decryption is essentially the same as the encryption process. The rule isas follows: use the cipher text as input to the algorithm, but use the subkey k_i in reverseorder. i.e., k_n in the first round, k_{n-1} in second round and so on. For clarity, we use thenotation LE_i and RE_i for data traveling through the decryption algorithm. The diagrambelow indicates that, at each round, the intermediate value of the decryption process issame(equal)tothecorrespondingvalueoftheencryptionprocesswithtwohalvesofthevalues wapped.

i.e., $RE_i || LE_i$ (or) equivalently $RD_{16-i} || LD_{16-i}$

Afterthelastiterationoftheencryptionprocess,thetwohalvesoftheoutputareswapped,sothatth eciphertextis$RE_{16}||LE_{16}$.Theoutputofthatroundistheciphertext.Now take the cipher text and use it as input to the same algorithm. The input to the firstroundis$RE_{16}||LE_{16}$,whichisequaltothe32-bitswapoftheoutputofthesixteenthroundof the encryption process. Now we will see how the output of the first round of thedecryption process is equal to a 32-bit swap of the input to the sixteenth round of theencryptionprocess.

First consider the encryption

process, $LE_{16}=RE_{15}$

$RE_{16}=LE_{15}(+)F(RE_{15}, K_{16})$

On the decryption side, $LD_1 = RD_0 = LE_{16}$

$= RE_{15} RD_1 = LD_0(+)F(RD_0, K_{16})$

$= RE_{16}F(RE_{15}, K_{16})$

$= [LE_{15}F(RE_{15}, K_{16})]F(RE_{15}, K_{16})$

$= LE_{15}$

Therefore, $LD_1 = RE_{15} RD_1 = LE_{15}$ Ingeneral,forthei_{th}iteration oftheencryptionalgorithm,$LE_i = RE_{i-1} RE_i = LE_{i-1}F(RE_{i-1}, K_i)$

Finally,theoutputofthelastroundofthedecryptionprocessis$RE_0||LE_0$.A32- bitswaprecoverstheoriginalplaintext.

DEFINITIONS

Encryption:Convertingatext intocodeorcipher.

Converting computer data and messages into something, incomprehensible use a

key,sothatonlyaholder ofthematchingkey can reconvertthem.

ConventionalorSymmetricorSecretKeyorSingleKeyencryption:

Usesthesamekeyforencryption&decryption.

PublicKeyencryption: Usesdifferentkeysforencryption&decryption

ConventionalEncryptionPrinciples

- Anencryptionschemehasfiveingredients:

1. Plaintext–Originalmessageordata.

2. Encryptionalgorithm–performssubstitutions&transformationsonplaintext.

3. SecretKey–exactsubstitutions&transformationsdependonthis

4. Ciphertext-outputiescrambledinput.

5. Decryptionalgorithm-convertsciphertextbacktoplaintext.

SIMPLIFIEDDATAENCRYPTIONSTANDARD(S-DES)
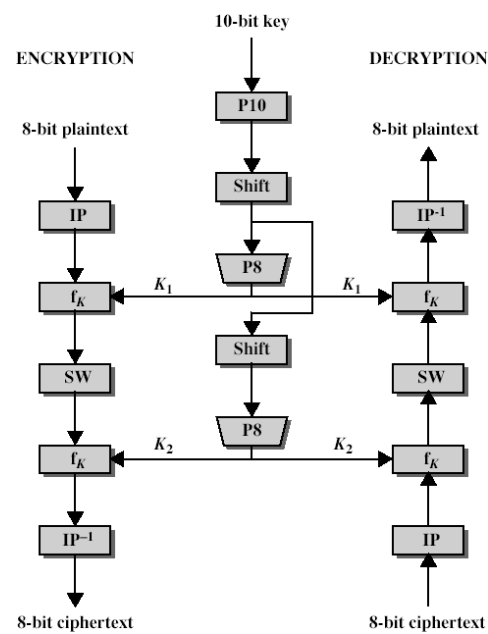
The figure above illustrates the overall structure of the simplified DES. The S-DESencryption algorithm takes an 8-bit block of plaintext (example: 10111101) and a 10-bitkey as input and produces an 8-bit block of ciphertext as output. The S-DES decryptionalgorithmtakesan8-bitblockofciphertextandthesame10-bitkeyusedtoproducethatciphertextas inputandproduces theoriginal8-bit blockofplaintext.

Theencryptionalgorithminvolvesfivefunctions:

- aninitialpermutation(IP)

- acomplexfunctionlabeledf$_k$,whichinvolvesbothpermutationandsubstitutionoperationsanddepends on a key input

- asimplepermutationfunctionthatswitches(SW)thetwohalvesofthedata

- thefunctionf$_k$again

- apermutationfunctionthatistheinverseoftheinitialpermutation

Thefunctionf$_k$takesasinputnotonlythedatapassingthroughtheencryptionalgorithm,butalsoan8-bitkey. Herea10-bitkeyis usedfromwhichtwo8-bitsubkeysaregenerated. The key is first subjected to a permutation (P10). Then a shift operation isperformed.Theoutputoftheshiftoperationthenpassesthroughapermutationfunctionthat produces an 8-bit output (P8) for the first subkey (K1). The output of the shiftoperation also feeds into another shift and another instance of P8 to produce the secondsubkey(K2). The encryption algorithm can be expressed as a composition composition1 of functions:IP-1o f$_{K2}$o SWo f$_{k1}$o IP

Whichcanalsobewrittenas

Ciphertext=IP$_{-1}$ (f$_{K2}$ (SW(f$_{k1}$ (IP(plaintext)))))

Where

K1=P8 (Shift (P10(Key)))

K2=P8 (Shift(shift(P10(Key))))

Decryptioncanbeshownas

Plaintext=IP$_{-1}$ (f$_{K1}$(SW(f$_{k2}$ (IP(ciphertext)))))
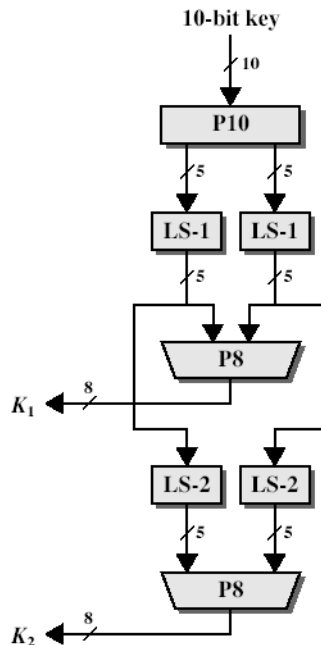
S-DES depends on the use of a 10-bit key shared between sender and receiver.Fromthiskey,two8-bitsubkeysareproducedforuseinparticularstagesoftheencryptionanddecryptionalgorithm.First,permutethekeyinthefollowingfashion.

Let the 10-bit key be designated as (k1, K2, k3, k4, k5, k6, k7, k8, k9, k10).Thenthepermutation P10is definedas:

P10(k1,K2,k3,k4,k5,k6,k7,k8,k9,k10)=(k3,k5,K2,k7,k4,k1010,k1,k9,k8,k6)P10 canbeconciselydefinedbythedisplay:

P10									
3	5	2	7	4	10	1	9	8	6

This table is read from left to right; each position in the table gives the identity of theinputbitthatproducestheoutputbitinthatposition.Sothefirstoutputbitisbit3oftheinput;thesecondoutputbitisbit5oftheinput,andsoon.Forexample,thekey(1010000010) is permuted to (10000 01100). Next, perform a circular left shift (LS-1),or rotation, separately on the first five bits and the second five bits. In our example, theresultis(0000111000).NextweapplyP8,whichpicksoutandpermutes8ofthe10bitsaccordingtothe followingrule:

P8							
6	3	7	4	8	5	10	9

The result is subkey 1 (K1). In our example, this yields (10100100). We then go back tothe pair of 5-bit strings produced by the two LS-1 functions and performs a circular leftshift of 2 bit positions on each string. In our example, the value (00001 11000) becomes(00100 00011). Finally, P8 is applied again to produce K2. In our example, the result is(01000011).

S-DESencryption

Encryptioninvolvesthesequentialapplicationoffivefunctions.

Initial and Final Permutations The input to the algorithm is an 8-bit block of plaintext,whichwe first permuteusingtheIP function:

IP							
2	6	3	1	4	8	5	7

This retains all 8 bits of the plaintext but mixes them

up.Considertheplaintext tobe 11110011.

Permutedoutput=10111101

Attheendofthealgorithm, theinversepermutation isused:

IP−1							
4	1	3	5	7	2	8	6

TheFunctionf_k

ThemostcomplexcomponentofS-

DESisthefunctionf_k,whichconsistsofacombinationofpermutationandsubstitutionfunctions.Thefunctionscanbeexpressedasfollows.LetL and R be the leftmost 4 bits and rightmost 4 bits of the 8-bit input to f K, and let F be amapping(not necessarilyone toone) from4-bit stringsto4-bit strings.

Thenweletf$_k$(L, R)=(L(+)F(R,SK),R)

WhereSKisasubkeyand(+)isthebit-by-bitexclusive-ORfunction.

e.g.,permutedoutput=10111101andsupposeF(1101,SK)=(1110)forsomekeySK.ThenfK(10111101) = 10111110, 1101= 01011101

WenowdescribethemappingF.Theinputisa4-bitnumber(n1n2n3n4).Thefirstoperationis anexpansion/permutation operation:

E/P							
4	1	2	3	2	3	4	1

R=1101E/Poutput=11101011Itisclearertodepicttheresultin thisfashion:

$$\begin{array}{cc|cc|c} n_4 & n_1 & n_2 & n_3 \\ n_2 & n_3 & n_4 & n_1 \end{array}$$

The8-bitsubkeyK1=(k11,k1212,k1313,k1414,k1515,k1616,k1717,k18)isaddedtothis valueusingexclusive-OR:

$$\begin{array}{c|cc|c} n_4 \oplus k_{11} & n_1 \oplus k_{12} & n_2 \oplus k_{13} & n_3 \oplus k_{14} \\ n_2 \oplus k_{15} & n_3 \oplus k_{16} & n_4 \oplus k_{17} & n_1 \oplus k_{18} \end{array}$$

Letusrenamethese8bits:

$$\begin{array}{c|cc|c} p_{0,0} & p_{0,1} & p_{0,2} & p_{0,3} \\ p_{1,0} & p_{1,1} & p_{1,2} & p_{1,3} \end{array}$$

Thefirst4bits(firstrowoftheprecedingmatrix)arefedintotheS-boxS0toproducea2-bitoutput, and the remaining 4 bits (second row) are fed into S1 to produce another 2- bitoutput. Thesetwoboxesaredefined asfollows:

$$S0 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{cccc} 0 & 1 & 2 & 3 \\ \left[\begin{array}{cccc} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{array}\right] \end{array} \qquad S1 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{cccc} 0 & 1 & 2 & 3 \\ \left[\begin{array}{cccc} 0 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{array}\right] \end{array}$$

The S-boxes operate as follows. The first and fourth input bits are treated as a 2-bitnumber that specify a row of the S-box, and the second and third input bits specify acolumn of the S-box. The entry in that row and column, in base 2, is the 2-bit output. Forexample, if (p0,0 p0,3) =) (00) and (p0,1 p0,2) = (10), then the output is from row 0,column 2 of S0, which is 3, or (11) in) binary. Similarly, (p1,0 p1,3) and (p1,1 p1,2) areusedtoindexintoarowandcolumnofS1toproduceanadditional2bits.Next,the4bitsproduce dbyS0andS1 undergo afurtherpermutationas follows:

P4			
2	4	3	1

TheoutputofP4istheoutputof the function F.

The Switch Function The function f K only alters the leftmost 4 bits of the input. The switchfunction(SW)interchangestheleftandright4bitssothatthesecondinstanceoffKoperates

onadifferent4bits.Inthissecondinstance,theE/P,S0,S1,andP4functionsarethesame.Thekeyinputis K2. Finallyapplyinversepermutationto getthe ciphertext

DATAENCRYPTIONSTANDARD(DES)

The main standard for encrypting data was a symmetric algorithm known as theDataEncryptionStandard(DES).However,thishasnowbeenreplacedbyanewstandardknow n as the Advanced Encryption Standard (AES) which we will look at later. DES is a64bitblockcipherwhichmeansthatitencryptsdata64bitsatatime.Thisiscontrastedtoastream cipherinwhichonlyonebitatatime(orsometimessmallgroupsofbitssuchas   a   byte)   is encrypted.   DES   was   the   result   of   a   research   project   set   up   by InternationalBusinessMachines(IBM)corporationinthelate1960'swhichresultedinacipherkn ownas LUCIFER. In the early 1970's it was decided to commercialise LUCIFER and a numberof significant changes were introduced. IBM was not the only one involved in thesechanges as they sought technical advice from the National Security Agency (NSA) (otheroutsideconsultantswereinvolvedbutitislikelythattheNSAwerethemajorcontributors from   a   technical   point   of   view).   The   altered   version   of   LUCIFER   was putforwardasaproposalforthenewnationalencryptionstandardrequestedbytheNationalBure auofStandards(NBS)3.Itwasfinallyadoptedin1977astheDataEncryption   Standard   -   DES (FIPS PUB 46). Some of the changes made to LUCIFER havebeen the subject of much controversy   even   to   the   present   day.   The   most   notable   of thesewasthekeysize.LUCIFERusedakeysizeof128bitshoweverthiswasreducedto56bitsfor DES. Even though DES actually accepts a 64 bit key as input, the remaining eight bitsareusedforparitycheckingandhavenoeffectonDES'ssecurity.Outsiderswereconvinced that the 56 bit key was an easy target for a brute force attack4 due to itsextremely small size. The need for the parity checking scheme was also questionedwithout satisfying answers.   Another   controversial   issue   was   that   the   S-boxes   used weredesignedunderclassifiedconditionsandnoreasonsfortheirparticulardesignwereevergiv en.ThisledpeopletoassumethattheNSAhadintroduceda"trapdoor"throughwhichthey   could decrypt any data encrypted by DES even without knowledge of the key. Onestartling discovery   was   that   the   S-boxes   appeared   to   be   secure   against   an   attack knownasDifferentialCryptanalysiswhichwasonlypubliclydiscoveredbyBihamandShamirin1 990.ThissuggeststhattheNSAwereawareofthisattackin1977;13yearsearlier!In

facttheDESdesignersclaimedthatthereasontheynevermadethedesignspecificationsfor the S-boxes available was that they knew about a number of attacks that weren'tpublicknowledgeatthetimeandtheydidn'twantthemleaking - thisisquiteaplausibleclaim as differential cryptanalysis has shown. However, despite all this controversy, in1994 NIST reaffirmed DES for government use for a further five years for use in areasother than "classified". DES of course isn't the only symmetric cipher. There are manyothers,eachwithvaryinglevelsofcomplexity.Suchciphersinclude:IDEA,RC4,RC5,RC6and the new Advanced Encryption Standard (AES). AES is an important algorithm andwas originally meant to replace DES (and its more secure variant triple DES) as thestandard algorithm for non-classified material. However as of 2003, AES with key sizesof 192 and 256 bits has been found to be secure enough to protect information up to topsecret.Sinceitscreation,AEShadunderdoneintensescrutinyasonewouldexpectforanalgorithm that is to be used as the standard. To date it has withstood all attacks but thesearch is still on and it remains to be seen whether or not this will last. We will look atAESlater in the course.

INNERWORKING OFDES

DES (and most of the other major symmetric ciphers) is based on a cipher known as the Feistelblock cipher. It consists of a number of rounds where each round contains bit-shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. As with most encryption schemes,DES expects two inputs - the plaintext to be encrypted and the secret key. The manner in whichthe plaintext is accepted, and the key arrangement used for encryption and decryption, bothdetermine the type of cipher it is. DES is therefore a symmetric, 64 bit block cipher as it uses thesamekeyforbothencryptionanddecryptionandonlyoperateson64bitblocksofdataatatime5(bethey plaintextorciphertext).Thekeysizeusedis56bits,howevera64bit(oreight-byte)keyisactuallyinput.Theleastsignificantbitofeachbyteiseitherusedforparity(oddforDES)orsetarbitrarily and does not increase the security in any way. All blocks are numbered from left torightwhichmakesthe eightbitof each bytetheparitybit.

Once a plain-text message is received to be encrypted, it is arranged into 64 bit blocks required for input. If the number of bits in the message is not evenly divisible by 64, then the last block will be padded. Multiple permutations and substitutions are incorporated throughout in order to increase the difficulty of performing a cryptanalysis on the cipher

OVERALL STRUCTURE

Figure below shows the sequence of events that occur during an encryption operation. DES performs an initial permutation on the entire 64 bit block of data. It is then split into 2 , 32 bit sub-blocks, Li and Ri which are then passed into what is known as a round (see figure 2.3), of which there are 16 (the subscript i in Li and Ri indicates the current round). Each of the rounds are identical and the effects of increasing their number is twofold - the algorithms security is increased and its temporal efficiency decreased. Clearly these are two conflicting outcomes and a compromise must be made. For DES the number chosen was 16, probably to guarantee the elimination of any correlation between the ciphertext and either the plaintext or key6. At the end of the 16th round, the 32 bit Li and Ri output quantities are swapped to create what is known as the pre-output. This [R16, L16] concatenation is permuted using a function which is the exact inverse of the initial permutation. The output of this final permutation is the 64 bit ciphertext.
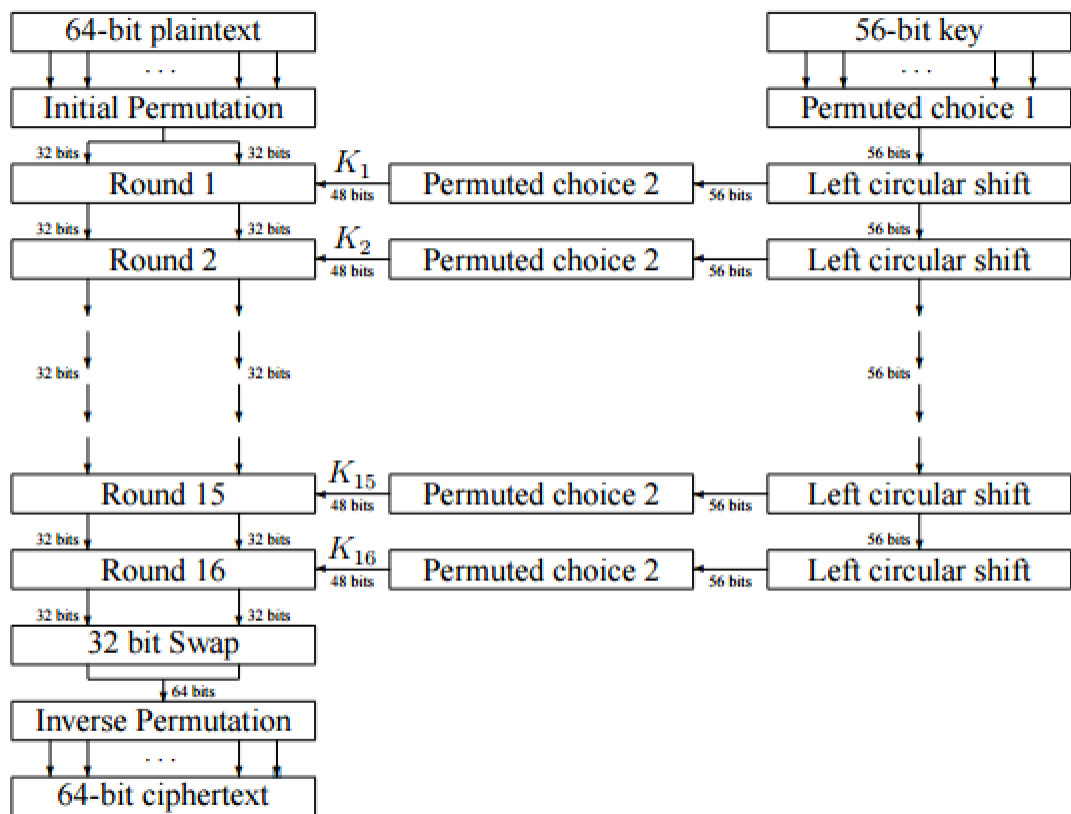
Figure   : Flow Diagram of DES algorithm for encrypting data.

So in total the processing of the plaintext proceeds in three phases as can be seen fromthelefthandside of figure

1. Initial permutation (IP - defined in table 2.1) rearranging the bits to form the"permutedinput".

2. Followed by 16 iterations of the same function (substitution and permutation). Theoutput of the last iteration consists of 64 bits which is a function of the plaintext andkey.The leftandrighthalves areswappedtoproduce thepreoutput.

3. Finally, the preoutput is passed through a permutation (IP−1 - defined in table 2.1)which is simply the inverse of the initial permutation (IP). The output of IP−1 is the 64-bitciphertext

(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

(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

(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

(d) Permutation Function (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

Table 2.1: Permutation tables used in DES.

As figure shows, the inputs to each round consist of the Li , Ri pair and a 48 bit subkeywhichisashiftedandcontractedversionoftheoriginal56bitkey.Theuseofthekeycanbe seen in the right hand portion of figure 2.2: • Initially the key is passed through apermutationfunction(PC1-definedintable2.2)•Foreachofthe16iterations,asubkey(Ki)isproducedbyacombinationofaleft circularshiftandapermutation(PC2-definedintable2.2)whichisthesameforeachiteration.However,theresultingsubkeyisdifferentforea chiterationbecause of repeatedshifts.

(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

Table 2.2: DES key schedule.

DETAILSOFINDIVIDUALROUNDS
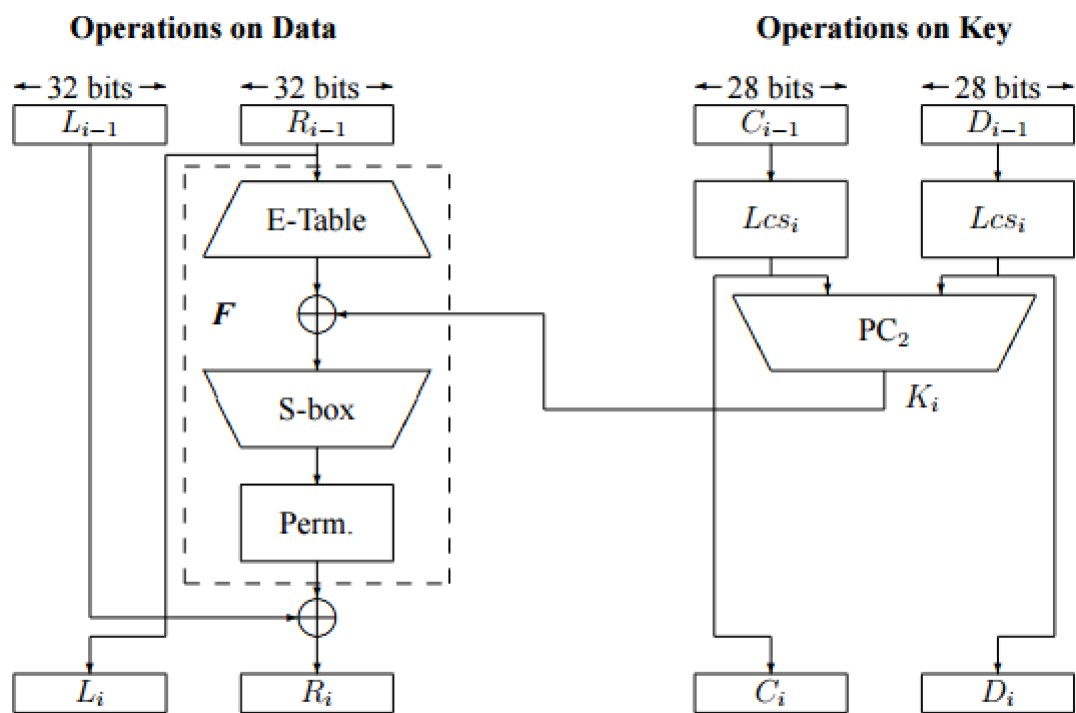
Figure   : Details of a single DES round.

The main operations on the data are encompassed into what is referred to as the cipher functionand is labeled F. This function accepts two different length inputs of 32 bits and 48 bits andoutputs a single 32 bit number. Both the data and key are operated on in parallel, however theoperations are quite different. The 56 bit key is split into two 28 bit halves Ci and Di (C and Dbeing chosen so as not to be confused with L and R). The value of the key used in any round issimplyaleftcyclicshift anda permuted contractionofthatusedintheprevious round. Mathematically,thiscanbewrittenas

$$Ci = Lcsi(Ci-1), Di = Lcsi(Di-1)Ki$$

$$=PC2(Ci,Di)$$

where Lcsi is the left cyclic shift for round i, Ci and Di are the outputs after the shifts, P C2(.) is afunction which permutes and compresses a 56 bit number into a 48 bit number and Ki is theactual key used in round i. The number of shifts is either one or two and is determined by theroundnumberi.Fori={1,2,9,16}thenumberofshiftsisoneandforeveryotherrounditistwo

S_1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Table 2.3: S-box details.
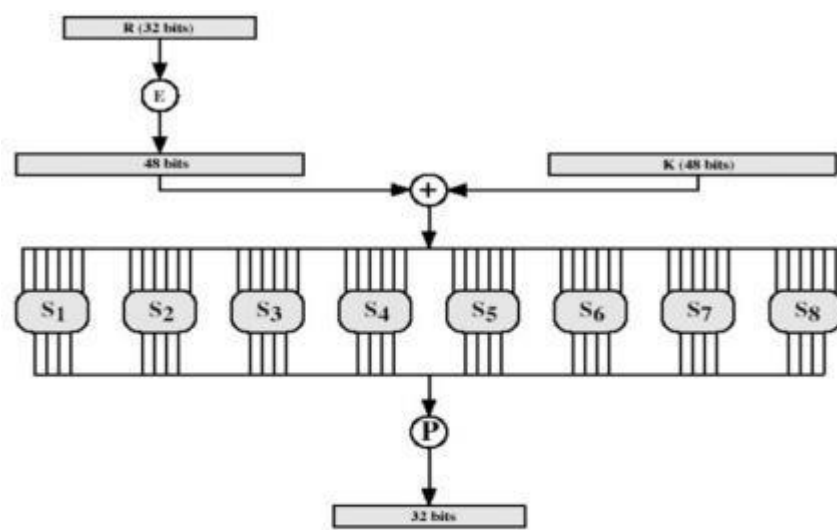
Figure 2.4: The complex **F** function of the DES algorithm.

ADVANCED ENCRYPTION ALGORITHM (AES)

- AES is a block cipher with a block length th of 128 bits.

- AES allows for three different key lengths: 128, 192, or 256 bits. Most of our discussion will assume that the key length this 128 bits.

- Encryption consists of 10 rounds of processing for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys.

- Except for the last round in each case, all other rounds are identical.

- Each round of processing includes one single-byte based substitution step, a row-wise permutation step, a column-wise mixing step, and the addition of the round key. The order in which these four steps are executed is different for encryption and decryption.

- To appreciate the processing steps used in a single round, it is best to think of a 128-bit block as consisting of a 4 ×4 matrix of bytes, arranged as follows:

$$\begin{bmatrix} byte_0 & byte_4 & byte_8 & byte_{12} \\ byte_1 & byte_5 & byte_9 & byte_{13} \\ byte_2 & byte_6 & byte_{10} & byte_{14} \\ byte_3 & byte_7 & byte_{11} & byte_{15} \end{bmatrix}$$

Therefore, the first four bytes of a 128-bit input block occupy the first column in the 4 × 4 matrix of bytes. The next four bytes occupy the second column, and so on. The 4×4 matrix of bytes shown above is referred to as the state array in AES.
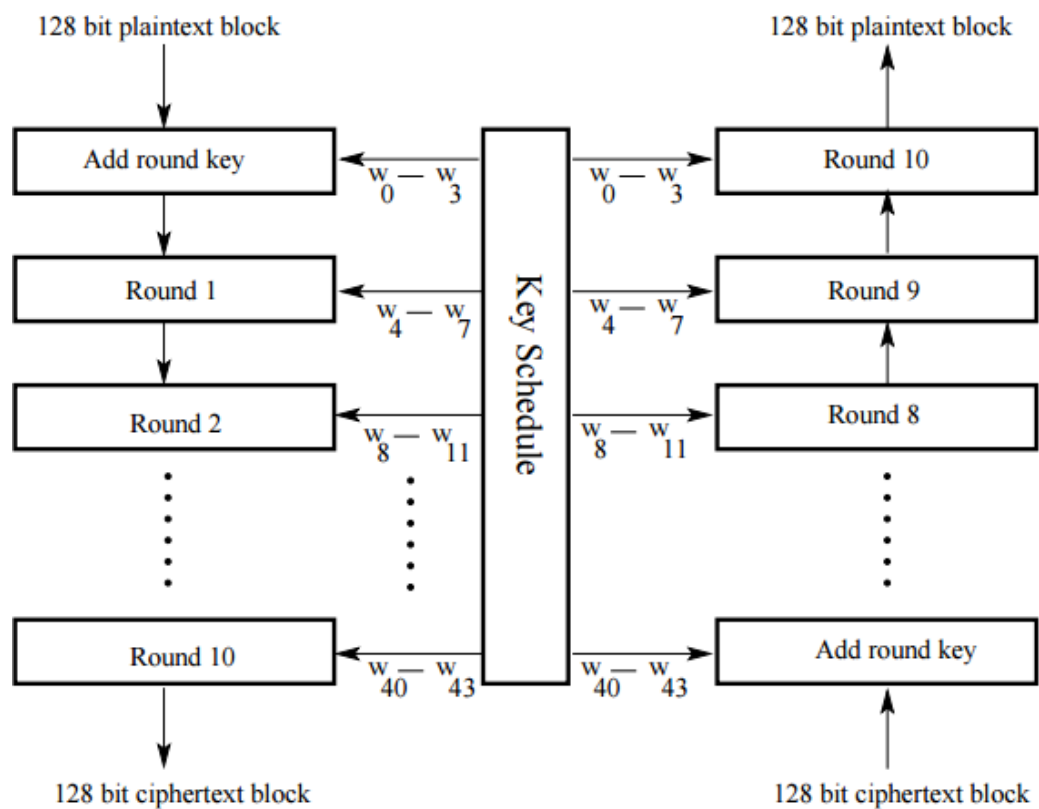
AES Encryption   AES Decryption

The algorithm begins with an Add round key stage followed by 9 rounds of four stagesandatenthroundofthree stages.

This applies for both encryption and decryption with the exception that each stage of aroundthedecryptionalgorithmistheinverseofitscounterpartintheencryptionalgorithm.

The four stages are as follows: 1. Substitute bytes 2. Shift rows 3. Mix Columns 4. AddRoundKey

SubstituteBytes

• This stage(knownasSubBytes)is simplya table lookupusinga

16×16matrixofbytevaluescalledans-box.

• This matrixconsistsofall the possiblecombinations ofan 8 bit sequence(28= 16× 16

=256).

• However,thes-

boxisnotjustarandompermutationofthesevaluesandthereisawelldefinedmethodforcreatingt

hes-boxtables.

• ThedesignersofRijndaelshowedhowthiswasdoneunlikethes-boxesinDESforwhichnorationalewasgiven.Ourconcernwillbehowstateiseffectedineachround.

• Forthisparticularroundeachbyteismappedintoanewbyteinthefollowingway:theleftmost nibble of the byte is used to specify a particular row of the s-box and therightmostnibblespecifies a column.

• Forexample,thebyte{95}(curlybracketsrepresenthexvaluesinFIPSPUB197)selectsrow9column5 whichturnsout tocontainthe value{2A}.

• Thisisthenusedtoupdatethestatematrix.

S-box
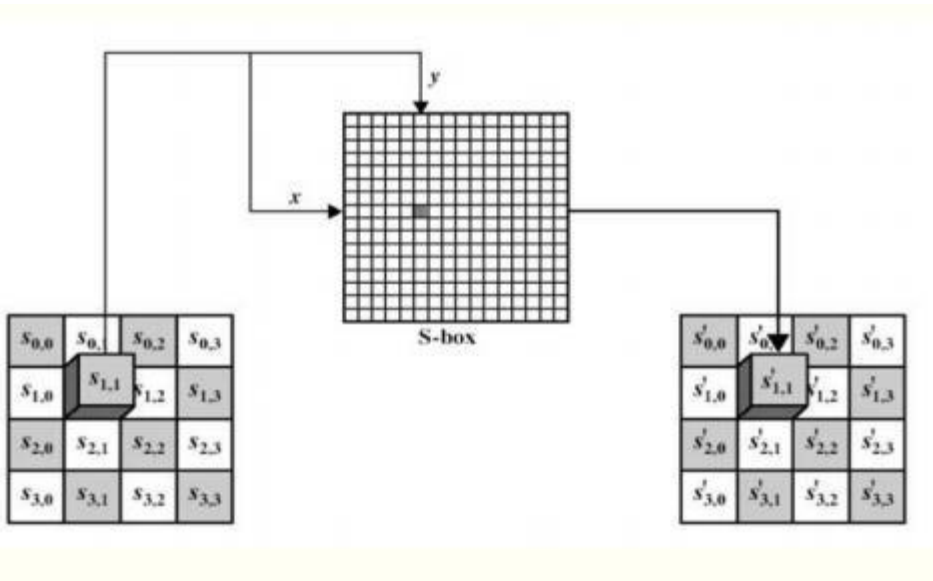
ShiftRowTransformation

• Thisstage(knownas ShiftRows)isshowninfigurebelow.

• Simplepermutationannothingmore.

• It works as follow: – The first row of state is not altered. – The second row is shifted 1bytes to the left in a circular manner. – The third row is shifted 2 bytes to the left in acircularmanner. – Thefourthrowis shifted3bytestotheleftin acircularmanner.
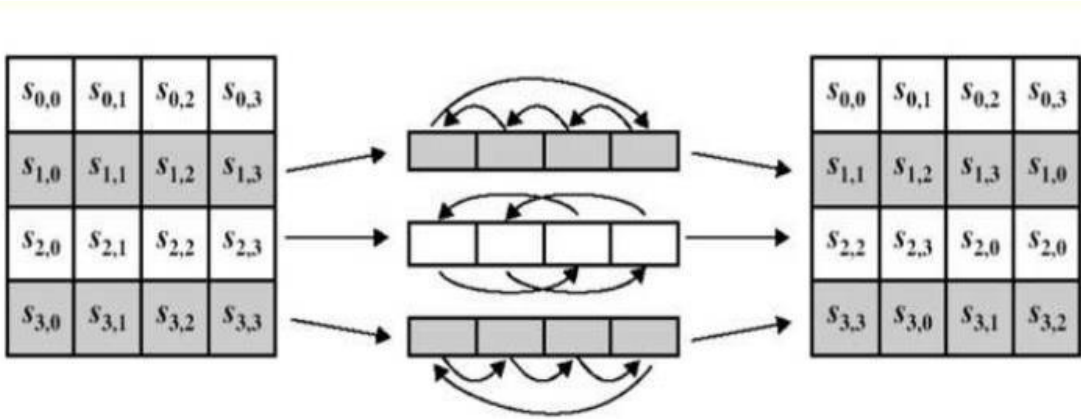
MixColumnTransformation

• This stage (known as MixColumn) is basically a substitution

• Each column is operated on individually. Each byte of a column is mapped into a new value that is a function of all four bytes in the column.

• The transformation can be determined by the following matrix multiplication on state

• Each element of the product matrix is the sum of products of elements of one row and one column.

• In this case the individual additions and multiplications are performed in GF(2^8).

• The MixColumn transformation of a single column j ($0 \le j \le 3$) of state can be expressed as:

$s'_{0,j} = (2 \bullet s_{0,j}) \oplus (3 \bullet s_{1,j}) \oplus s_{2,j} \oplus s_{3,j}$

$s'_{1,j} = s_{0,j} \oplus (2 \bullet s_{1,j}) \oplus (3 \bullet s_{2,j}) \oplus s_{3,j}$

$s'_{2,j} = s_{0,j} \oplus s_{1,j} \oplus (2 \bullet s_{2,j}) \oplus (3 \bullet s_{3,j})$

$s'_{3,j} = (3 \bullet s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \bullet s_{3,j})$
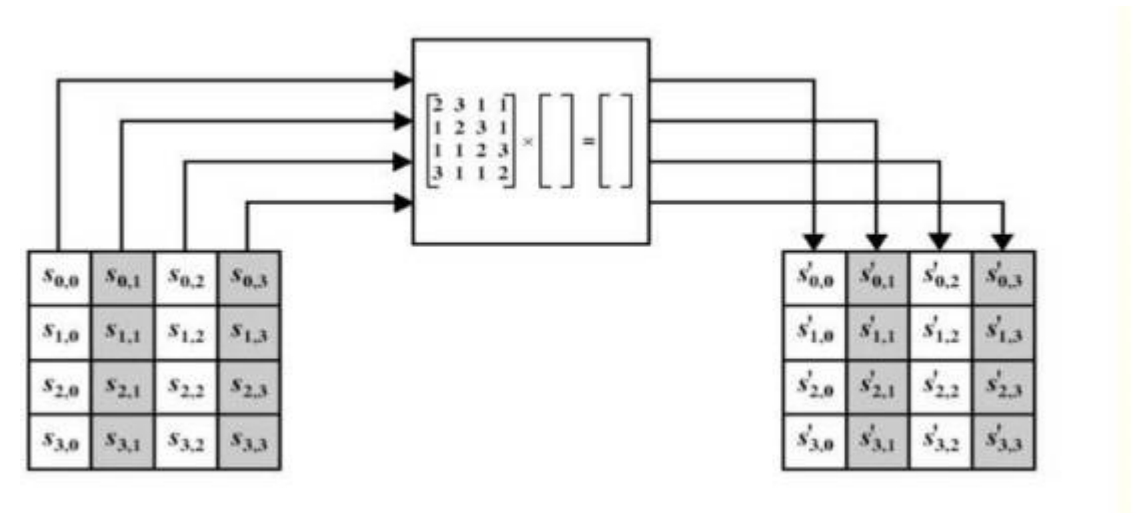
AddRoundKeyTransformation

• In this stage (known as AddRoundKey) the 128 bits of state are bitwise XORed with the 128 bits of the round key.

• The operation is viewed as a column wise operation between the 4 bytes of a state column and one word of the round key.

• This transformation is as simple as possible which helps in efficiency but it also effects every bit of state.

• The AES key expansion algorithm takes as input a 4-word key and produces a linear array of 44 words. Each round uses 4 of these words as shown in figure.

• Each word contains 32 bytes which means each subkey is 128 bits long. Figure 7 show pseudocode for generating the expanded key from the actual key.

BLOWFISH ALGORITHM

- asymmetric block cipher designed by Bruce Schneier in 1993/94
- characteristics
  - fast implementation on 32-bit CPUs
  - compact in use of memory
  - simple structure for analysis/implementation
  - variable security by varying key size
- has been implemented in various products

BLOWFISH KEY SCHEDULE

- uses a 32 to 448 bit key, 32-bit words stored in K-array K_j, j from 1 to 14
- used to generate
  - 18 32-bit subkeys stored in P array, $P_1 \ldots P_{18}$
  - four 8x32 S-boxes stored in $S_{i,j}$, each with 256 32-bit entries

Subkeys and S-Boxes Generation:

1. initialize P-array and then 4 S-boxes in order using the fractional part of pi P_1 (leftmost 32-bit), and so on,,, $S_{4,255}$.

2. XOR P-array with key-Array (32-bit blocks) and reuse as needed: assume we have up to k_{10} then $P_{10} XOR K_{10}$,, $P_{11} XOR K_1 \ldots P_{18} XOR K_8$

3. Encrypt 64-bit block of zeros, and use the result to update P_1 and P_2.

4. encrypting output form previous step using current P&S and replace P_3 and P_4. Then encrypting current output and use it to update successive pairs of P.

5. After updating all P's (last: $P_{17} P_{18}$), start updating S values using the encrypted output from previous step.
   - requires 521 encryptions, hence slow in re-keying
   - Not suitable for limited-memory applications.

BLOWFISHENCRYPTION

- usestwomainoperations:additionmodulo2^{32},andXOR
- dataisdividedintotwo 32-bithalvesL_0&R_0

for$i$=1to16do

R_i=L_{i-1}XORP_i;

L_i = F[R_i] XOR R_{i-1};L_{17}

= R_{16} XOR

P_{18};R_{17}=L_{16}XORP_{17};

- where

F[a,b,c,d]=(($S_{1,a}$+$S_{2,b}$)XOR$S_{3,c}$) +$S_{4,d}$
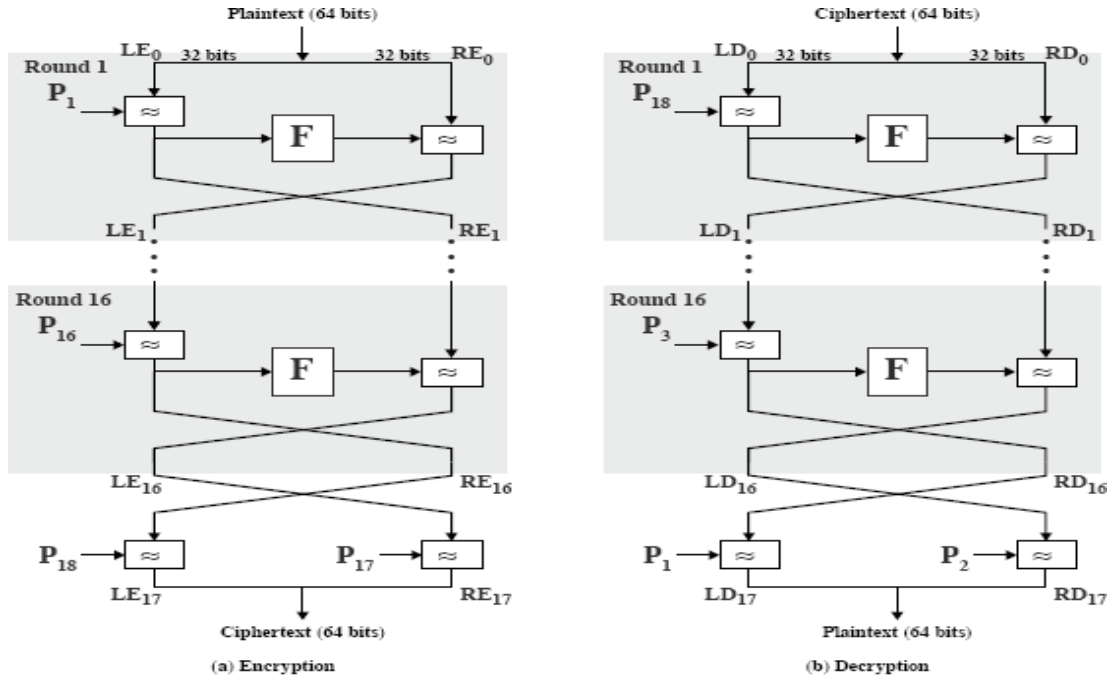
Figure 6.3 Blowfish Encryption and Decryption
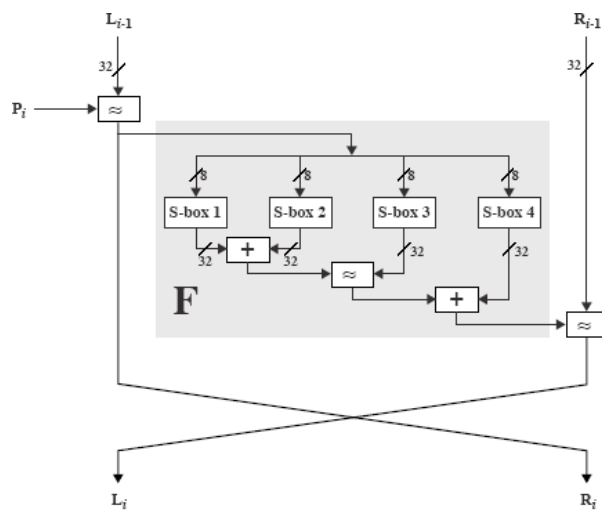
Figure 6.4 Detail of Single Blowfish Round

BLOCKCIPHERMODESOF OPERATIONS

- Directuseofablockcipheris inadvisable
- Enemycanbuildup"codebook"ofplaintext/ciphertextequivalents
- Beyond that, direct use only works on messages that are a multiple of the cipher blocksizeinlength
- Solution: five standard Modes of Operation: Electronic Code Book (ECB), Cipher BlockChaining(CBC),CipherFeedback(CFB),OutputFeedback(OFB),andCounter(CTR).

ElectronicCodeBook

- Directuseoftheblockcipher
- Usedprimarilytotransmitencryptedkeys
- Veryweakifusedforgeneral-purposeencryption;neveruse itforafileoramessage.
- Attackercanbuildupcodebook;nosemanticsecurity
- Wewrite{P}k→Ctodenote"encryptionofplaintextPwithkeyktoproduceciphertextC"
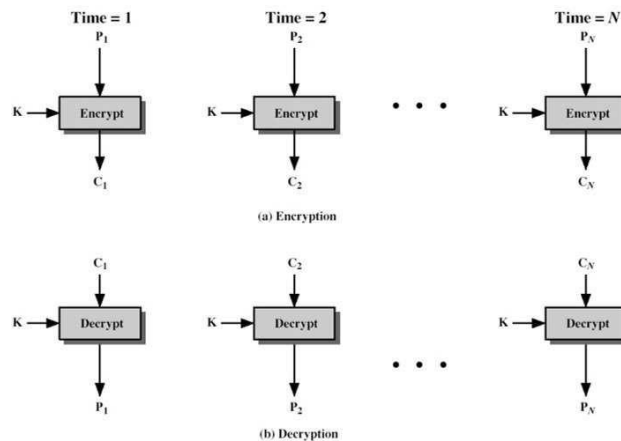
CipherBlockChaining

• Wewouldlikethatsameplaintextblocksproducedifferentciphertextblocks.

• CipherBlockChaining(seefigure)allowsthisbyXORingeachplaintextwiththeCiphertextfrom

thepreviousround(thefirstroundusinganInitialisationVector(IV)).

• Asbefore, thesamekeyisusedforeachblock.

• DecryptionworksasshowninthefigurebecauseofthepropertiesoftheXORoperation,
i.e.IV$\oplus$ IV$\oplus$ P=PwhereIVistheInitialisationVectorandPistheplaintext.
• ObviouslytheIVneedstobeknownbybothsenderandreceiveranditshouldbekeptsecretalong
with thekeyformaximumsecurity.

(a) Encryption

(b) Decryption
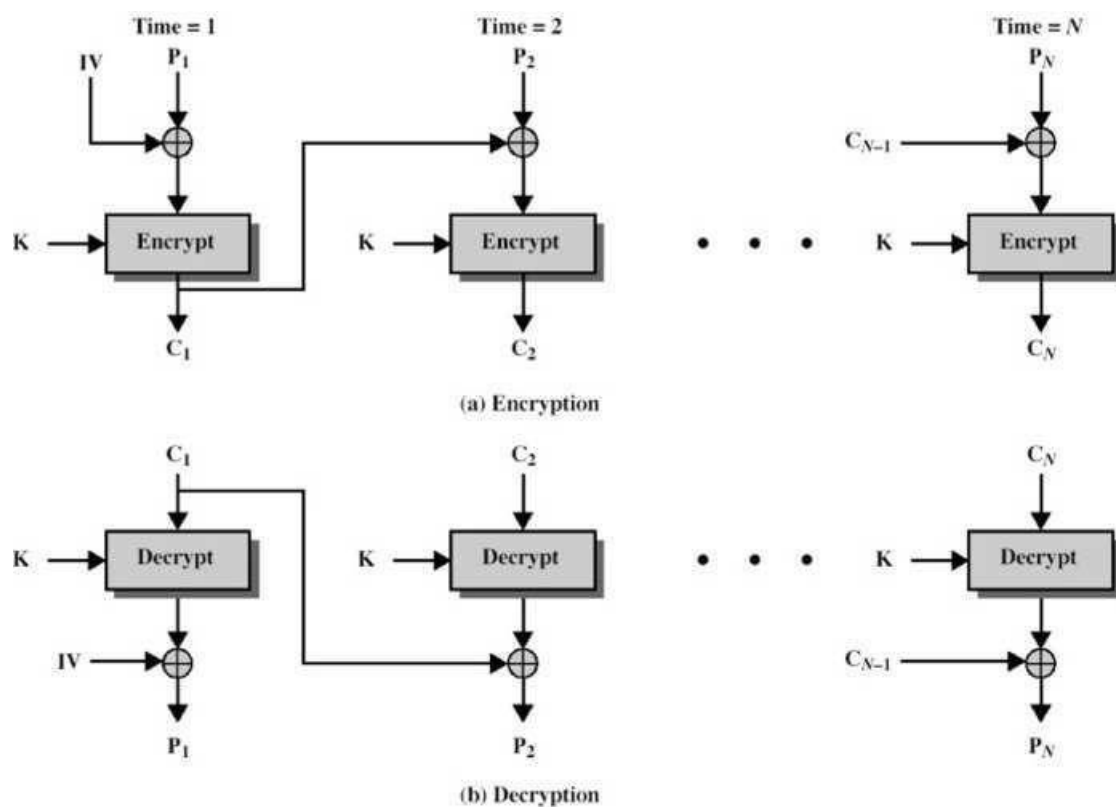
CipherFeedback(CFB)Mode
• TheCipherFeedbackandOutputFeedbackallowsablockciphertobeconvertedintoastream

cipher.

• Thiseliminatestheneedtopadamessagetobeanintegralnumberofblocks.Italsocanoperateinr

eal time.

• FigureshowstheCFB scheme.

• Inthisfigureitassumedthattheunitoftransmissionissbits;acommonvalueiss=8.

• As with CBC, the units of plaintext are chained together, so that the ciphertext of anyplaintextunitisafunctionofalltheprecedingplaintext(whichissplitintosbitsegments).

• Theinputtotheencryptionfunctionisashiftregisterequalinlengthtotheblockcipherof the algorithm (although the diagram shows 64 bits, which is block size used by DES,thiscanbe extendedtoother blocksizes suchasthe128bits ofAES).

• ThisisinitiallysettosomeInitialisationVector(IV).

(a) Encryption
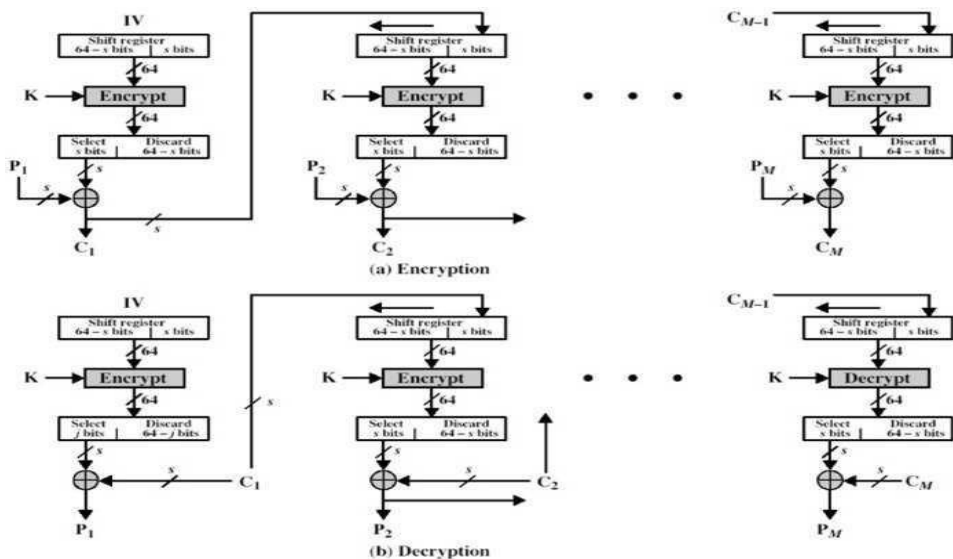
(b) Decryption

OUTPUTFEEDBACK (OFB)MODE

• TheOutputFeedbackModeissimilarinstructuretothatofCFB,asseeninfigure13.

• Ascanbeseen,itistheoutputoftheencryptionfunctionthatisfedbacktotheshiftregisterinOFB, whereasin CFBtheciphertextunitis fedbacktotheshiftregister.

• OneadvantageoftheOFBmethodisthatbiterrorsintransmissiondonotpropagate.

• Forexample,ifabiterroroccursinC1onlytherecoveredvalueofP1isaffected;subsequentplaint extunitsare notcorrupted.

WithCFB,C1alsoservesasinputtotheshiftregisterandthereforecausesadditionalcorruptiondownstre am.

(a) Encryption

(b) Decryption

CounterMode

(a) Encryption

(b) Decryption
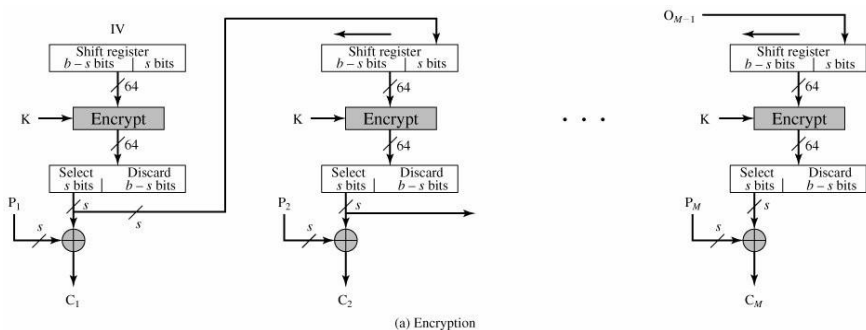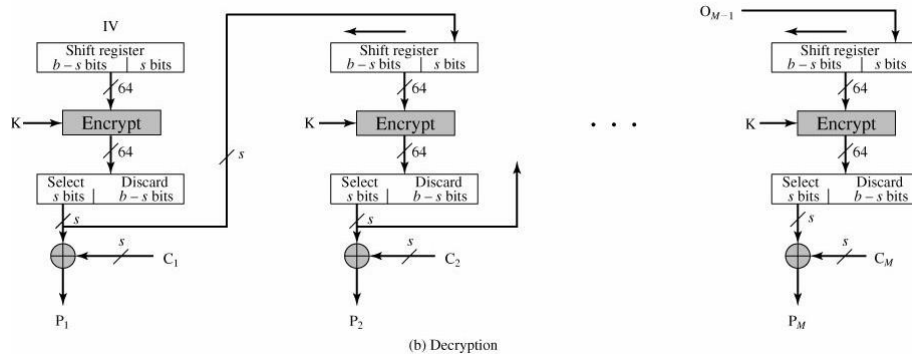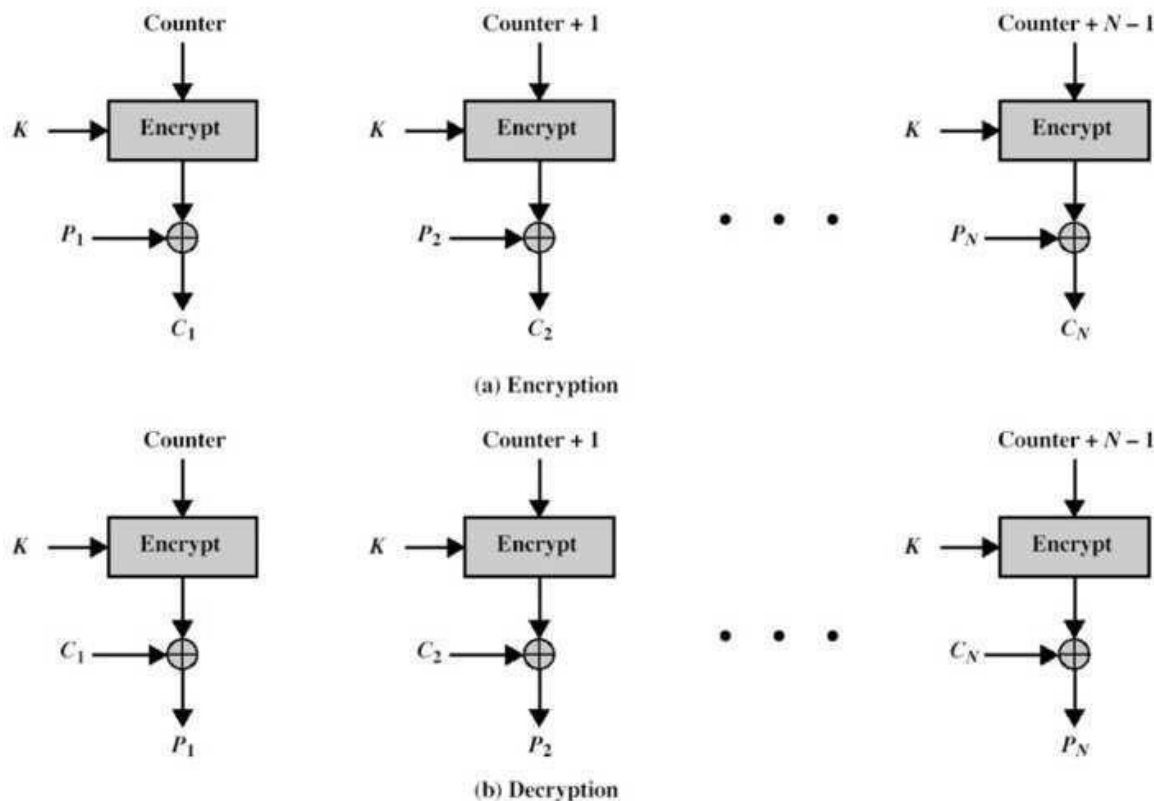
PUBLICKEYCRYPTOGRAPHY

Thedevelopmentofpublic-keycryptographyisthegreatestandperhapstheonlytrue revolution in the entire history of cryptography. It is *asymmetric*, involving the useof two separate keys, in contrast to symmetric encryption, which uses only one key.Public key schemes are neither more nor less secure than private key (security dependson the key size for both). Public-key cryptography *complements rather than replaces*symmetriccryptography.Bothalsohaveissueswithkeydistribution,requiringtheuse

of some suitable protocol. The concept of public-key cryptography evolved from anattempttoattacktwoofthemostdifficultproblemsassociatedwithsymmetricencryption:

1.) *key distribution* – how to have secure communications in general without having totrusta KDCwith yourkey

2.)*digitalsignatures* –howtoverifyamessagecomesintactfromtheclaimedsender

Public-key/two-key/asymmetriccryptographyinvolvestheuseof**two**keys:

- a*public-key*,whichmaybeknownbyanybody,andcanbeusedto**encryptmessages**,and**verify signatures**
- a*private-key*,knownonlytotherecipient,usedto**decryptmessages**,and**sign** (create)**signatures**.
- is**asymmetric**becausethosewhoencryptmessagesorverifysignaturescannotdecrypt messages orcreate signatures

Public-Keyalgorithmsrelyononekeyforencryptionandadifferentbutrelatedkeyfordecryption. Thesealgorithmshavethefollowingimportantcharacteristics:

- itiscomputationallyinfeasibletofinddecryptionkeyknowingonlyalgorithm&encryptionkey
- itiscomputationallyeasytoen/decryptmessageswhentherelevant(en/decrypt)keyis known
- either of the two related keys can be used for encryption, with the other used fordecryption (for some algorithms likeRSA)

The following figure illustrates public-key encryption process and shows that a public-keyencryptionschemehassixingredients:plaintext,encryptionalgorithm,public&privatekeys,ciphertext& decryptionalgorithm.
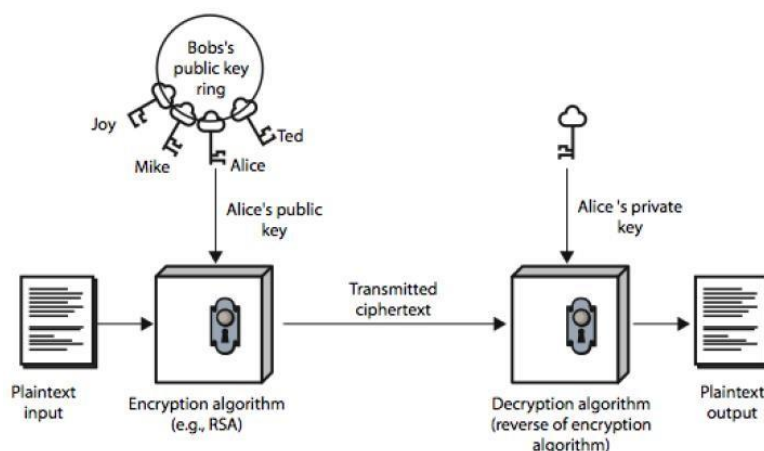
The essential steps involved in a public-key encryption scheme are given below:

1.) Each user generates a pair of keys to be used for encryption and decryption.

2.) Each user places one of the two keys in a public register and the other key is kept private.

3.) If B wants to send a confidential message to A, B encrypts the message using A's public key.

4.) When A receives the message, she decrypts it using her private key. Nobody else can decrypt the message because that can only be done using A's private key (Deducing a private key should be infeasible).

5.) If a user wishes to change his keys –generate another pair of keys and publish the public one: no interaction with other users is needed.

Notations used in Public-key cryptography:

- The public key of user A will be denoted **KUA**.
- The private key of user A will be denoted **KRA**.
- Encryption method will be a function E.
- Decryption method will be a function D.
- If B wishes to send a plain message X to A, then he sends the cryptotext $Y = E(KU_A, X)$
- The intended receiver A will decrypt the message: $D(KR_A, Y) = X$

The first attack on Public-key Cryptography is the attack on Authenticity. **An attacker may impersonate user B**: he sends a message $E(KU_A, X)$ and claims in the message to be B – A has no guarantee this is so. To overcome this, B will encrypt the message using his private key: $Y = E(KR_B, X)$. Receiver decrypts using B's public key KR_B. This shows the authenticity of the sender because (supposedly) he is the only one who knows the private key. The entire encrypted message serves as a digital signature. This scheme is depicted in the following figure:
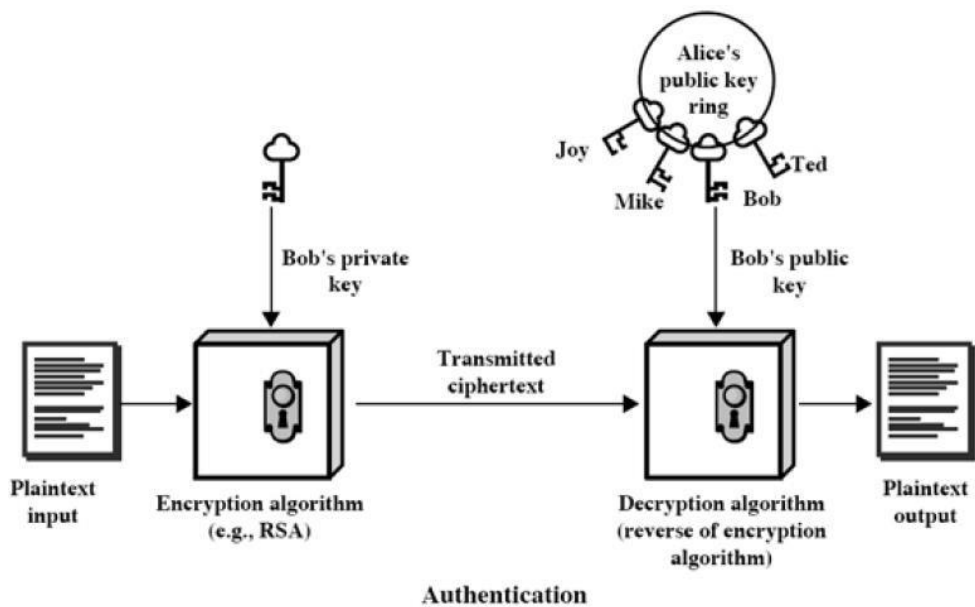
Authentication

But, a drawback still exists. Anybody can decrypt the message using B's public key. So, secrecy or confidentiality is being compromised. One can provide both *authentication and confidentiality* using the public-key scheme twice:

Public-Key Cryptosystem: Secrecy and Authentication
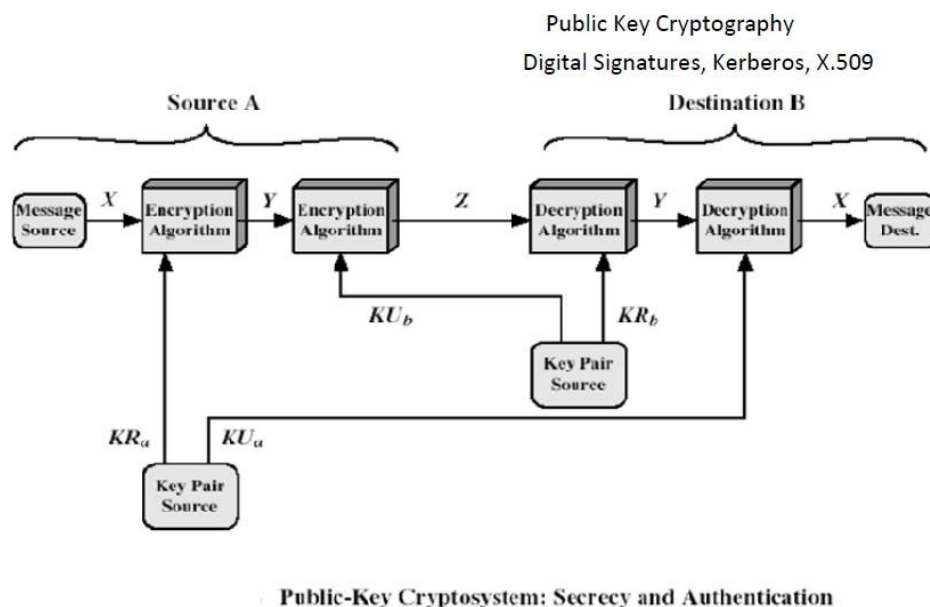
B encrypts X with his private key:

Y=E(KRB,X) B encrypts Y with A's public key: Z=E(KU

A,Y)

A will decrypt Z (and she is the only one capable of doing it): Y=D(KRA,Z)

A can now get the plaintext and ensure that it comes from B (he is the only one who knows his private key): decrypt Y using B's public key: X=E(KUB,Y).

Applications for public-key cryptosystems:

1.)**Encryption/decryption**: sender encrypts the message with the receiver's public key.

2.)**Digital signature**: sender "signs" the message (or a representative part of the message) using his private key

3.) **Key exchange**: two sides cooperate to exchange a secret key for later use in a secret-key cryptosystem.

The main requirements of Public-key cryptography are:

1. Computationally easy for a party B to generate a pair (public key KUb, private key KRb).

2. Easy for sender A to generate ciphertext:

3. Easy for the receiver B to decrypt ciphertect using private key:

4. Computationally infeasible to determine private key (KRb) knowing public key (KUb)

5. Computationally infeasible to recover message M, knowing KUb and ciphertext C

6. Either of the two keys can be used for encryption, with the other used for decryption:

$$M = D_{KRb}[E_{KUb}(M)] = D_{KUb}[E_{KRb}(M)]$$

Easy is defined to mean a problem that can be solved in polynomial time as a function of input length. A problem is infeasible if the effort to solve it grows faster than polynomial time as a function of input size. Public-key cryptosystems usually rely on difficult math functions rather than S-P networks as classical cryptosystems. **One-way function** is one, easy to calculate in one direction, infeasible to calculate in the other direction (i.e., the inverse is infeasible to compute). **Trap-door function** is a difficult function that becomes easy if some extra information is known. Our aim to find a ***trap-door one-way function***, which is easy to calculate in one direction and infeasible to calculate in the other direction unless certain additional information is known.

Security of Public-key schemes:

- Like private key schemes brute force **exhaustive search** attack is always theoretically possible. But keys used are too large (>512 bits).

- Security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalyse) problems. More generally the **hard** problem is known, its just made too hard to do in practise.

- Requires the use of **very large numbers,** hence is **slow** compared to private key schemes

RSA ALGORITHM

RSA is the best known, and by far the most widely used general public key encryption algorithm, and was first published by Rivest, Shamir & Adleman of MIT in 1978 [RIVE78]. Since that time RSA has reigned supreme as the most widely accepted and implemented general-purpose approach to public-key encryption. The RSA scheme is a block cipher in which the plaintext and the ciphertext are integers between 0 and n-1 for some fixed n and typical size for n is 1024 bits (or 309 decimal digits). It is based on exponentiation in a finite (Galois) field over integers modulo a prime, using large integers (eg. 1024 bits). Its security is due to the cost of factoring large numbers. RSA involves a public-key and a private-key where the public key is known to all and is used to encrypt data or message. The data or message which has been encrypted using a public key can only be decryted by using its corresponding private-key. Each user generates a key pair i.e. public and private key using the following steps:

- *each user selects two large primes at random - p, q*
- *compute their system modulus n=p.q*
- *calculate ø(n), where **ø(n)=(p-1)(q-1)***
- *selecting at random the encryption key e, where 1<e<ø(n), and **gcd(e,ø(n))=1***
- *solve following equation to find decryption key d: **e.d=1mod ø(n)** and 0≤d≤n*
- *publish their public encryption key: **KU={e,n}***
- *keep secret private decryption key: **KR={d,n}***

Both the sender and receiver must know the values of n and e, and only the receiver knows the value of d. Encryption and Decryption are done using the following equations. To encrypt a message M the sender:

– obtains **public key** of recipient *KU={e,n}*

– computes: **C=Me mod n**, where

0≤M<n To decrypt the ciphertext C the owner:

– uses their private key *KR={d,n}*

– computes: **M=Cdmodn=(Me)dmodn=Medmodn**

Forthisalgorithmtobesatisfactory,thefollowingrequirementsaretobemet.

a) Itspossibletofindvaluesofe, d,nsuchthat**Med=Mmodn**forallM<n

b) ItisrelativelyeasytocalculateMeandCforallvaluesofM<n.

c) Itisimpossibletodeterminedgiveneandn

The way RSA works is based on Number theory: **Fermat's little theorem**: if **p** isprime and **a** is positive integer not divisible by **p**, then **ap-1 ≡ 1 mod p. Corollary**: Foranypositive integer **a**andprime **p,ap≡a mod p.**

Fermat'stheorem,asusefulaswillturnouttobedoesnotprovideuswithintegersd,e   we are looking for –Euler's theorem (a refinement of Fermat's) does. Euler's functionassociates to any positive integer **n,** a number φ**(n)**: the number of positive integerssmaller than **n** and relatively prime to **n.** For example, φ**(37) = 36** i.e. φ**(p) = p-1** for anyprime**p.**Foranytwoprimes**p,q**,φ**(pq)=(p-1)(q-**

1).Euler'stheorem:foranyrelativelyprime integers a,n we have **a**φ**(n)≡1 mod n. Corollary:**   **F**or   any   integers   a,n   we have**a**φ**(n)+1≡a**modn**Corollary:**Letp,qbetwooddprimesandn=pq.Then:φ(n)=(p-1)(q-1) For any integer m with 0<m<n, m(p-1)(q-1)+1 ≡ m mod n For any integers k,m with0<m<n, mk(p-1)(q-1)+1 ≡ m mod n Euler's theorem provides us the numbers d, e suchthat Med=M mod n. We have to choose d,e such that ed=kφ(n)+1, or equivalently, d≡e-1modφ(n)

An example of RSA can be given

as,Select primes: *p*=17

&*q*=11Compute*n*=*pq* =17×11=187

Compute ø(*n*)=(*p*–1)(*q*-

1)=16×10=160Select*e:*gcd(e,160)=1;choo

se *e*=7

Determined:*de*=1mod160and*d*<160Valueisd=23since23×7=161=10×160+1Publishpublic

keyKU={7,187}

Keep secret private key KR={23,187}Now,

given message M = 88 (nb.

88<187)encryption:C=887mod187=11

decryption:M=1123mod187=88

Anotherexampleof RSA is given as,

Let **p=11,q=13,e=11,m=7**

n=pq i.e. n=11*13=143

ø(n)=(p-1)(q-1) i.e.(11-1)(13-1)=120

e.d=1 mod ø(n) i.e. 11d mod 120 = 1 i.e. (11*11) mod

120=1;so d=11 public key:{11,143} and private key:{11,143}

C=Me mod n, so ciphertext=711 mod143=727833 mod143;i.e. C=106

M=Cd mod n, plaintext=10611 mod143=1008 mod 143;i.e. M=7
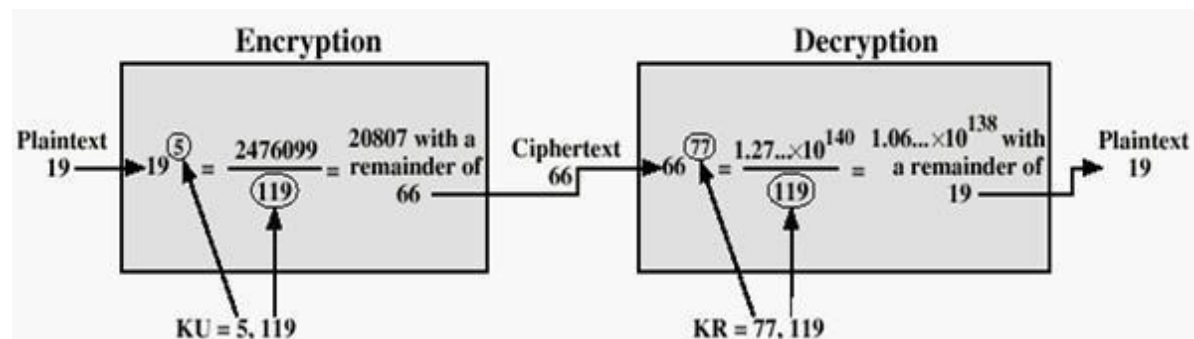
ForRSAkeygeneration,

🞐 users of RSA must:

– determinetwoprimesatrandom -p,q

– selecteithereordandcomputethe other

🞐 primes p,q must not be easily derived from modulus N=p.q

– meansmustbesufficientlylarge

– typicallyguessanduseprobabilistictest

🞐 exponents e, d are inverses, so use Inverse algorithm to compute the other

SecurityofRSA

TherearethreemainapproachesofattackingRSAalgorithm.

Brute force key search (infeasible given size of numbers) As explained before, involvestryingallpossibleprivate keys.Bestdefenceisusinglarge keys.

Mathematical attacks (based on difficulty of computing ø(N), by factoring modulus N)There are several approaches, all equivalent in effect to factoring the product of twoprimes. Some of themare given as:

– factorN=p.q,hencefindø(N)andthend

– determineø(N)directlyandfindd

– findddirectly

The possible defense would be using large keys and also choosing large numbers for pandq,whichshoulddifferonlybyafewbitsandarealsoontheorderofmagnitude10^{75}to10^{100}. Andgcd(p-1, q-1) shouldbe small.

DIFFIE-HELLMAN KEYEXCHANGE

Diffie-Hellman key exchange (**D-H**) is a cryptographic protocol that allows two partiesthat have no prior knowledge of each other to jointly establish a shared secret key overan insecure communications channel. This key can then be used to encrypt subsequentcommunicationsusingasymmetrickeycipher.TheD-Halgorithmdependsforitseffectivenesson thedifficultyofcomputingdiscrete logarithms.

First,aprimitiverootofaprimenumberp,canbedefinedasonewhosepowersgenerateall the integers from 1 to p-1. If a is a primitive root of the prime number p, then thenumbers,**$a\,mod\,p, a^2\,mod\,p, ..., a^{p-1}\,mod\,p$**,aredistinctandconsistoftheintegersfrom1throughp1 in some permutation.

Foranyinteger**b**andaprimitiveroot**a**ofprimenumber**p**,wecanfindauniqueexponent

isuchthat $b \equiv a^i \pmod{p}$ where $0 \leq i \leq (p\ 1)$ .Theexponentiisreferredtoasthediscrete logarithm of b for the base a, mod p. We express this value as **$dlog_{a,p}\,(b)$.** Thealgorithmis summarizedbelow:

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \bmod q$

Generation of Secret Key by User A
$K = (Y_B)^{X_A} \bmod q$

Generation of Secret Key by User B
$K = (Y_A)^{X_B} \bmod q$
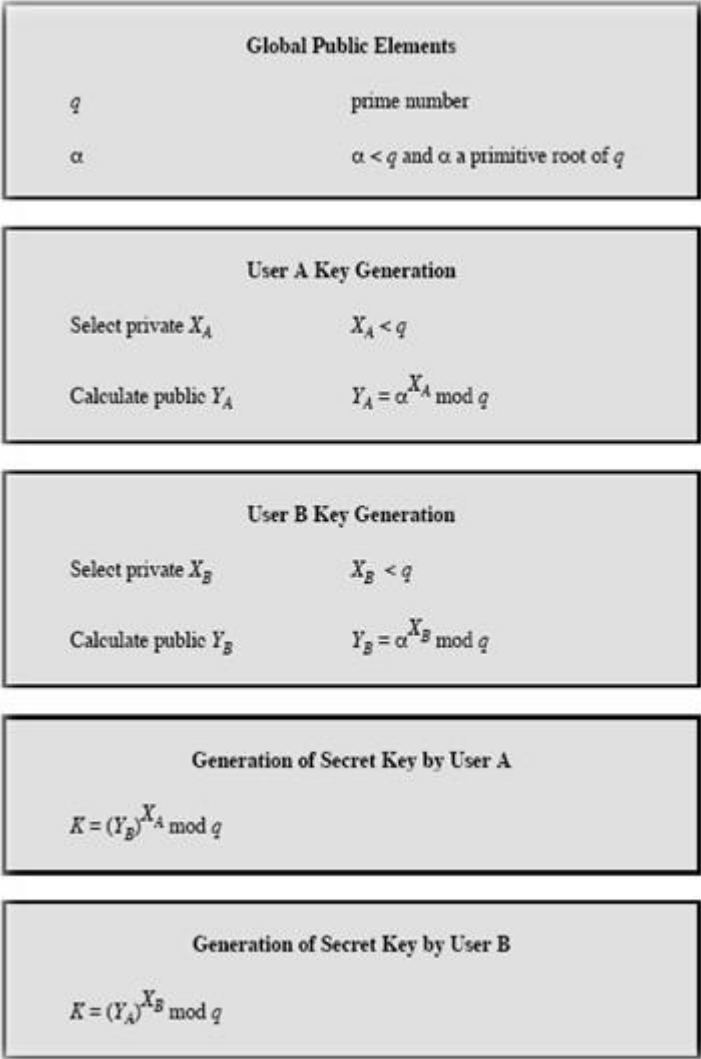
For this scheme, there are two publicly known numbers: a prime number q and an integer α that is a primitive root of q. Suppose the users A and B wish to exchange a key. User A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{X_A} \bmod q$. Similarly, user B independently selects a random integer $X_A < q$ and computes $Y_B = \alpha^{X_B} \bmod q$. Each side keeps the X value private and makes the Y value available publicly to the other side. User A computes the key as $K = (Y_B)^{X_A} \bmod q$ and user B computes the key as $K = (Y_A)^{X_B} \bmod q$. These two calculations produce identical results.

DiscreteLogProblem

The (discrete) exponentiation problem is as follows: Given a base a, an exponent b and a modulus p, calculate c such that $a^b \equiv c \pmod{p}$ and $0 \le c < p$. It turns out that this problem is fairly easy and can be calculated "quickly" using fast-exponentiation. The discrete log problem is the inverse problem: Given a base a, a result c ($0 \le c < p$) and a modulus p,

calculate the exponent b such that $a^b \equiv c \pmod{p}$. It turns out that no one has found a quick way to solve this problem With DLP, if P had 300 digits, X_a and X_b have more than 100 digits, it would take longer than the life of the universe to crack the method.

Examples for D-H key distribution scheme:

1) Let p = 37 and g = 13.

Let Alice pick a = 10. Alice calculates $13^{10} \pmod{37}$ which is 4 and sends that to Bob. Let Bob pick b = 7. Bob calculates $13^7 \pmod{37}$ which is 32 and sends that to Alice. (Note: 6 and 7 are secret to Alice and Bob, respectively, but both 4 and 32 are known by all.)

⬚ Alice receives 32 and calculates $32^{10} \pmod{37}$ which is 30, the secret key.

⬚ Bob receives 4 and calculates $4^7 \pmod{37}$ which is 30, the same secret key.

2) Let p = 47 and g = 5. Let Alice pick a = 18. Alice calculates $5^{18} \pmod{47}$ which is 2 and sends that to Bob. Let Bob pick b = 22. Bob calculates $5^{22} \pmod{47}$ which is 28 and sends that to Alice.

⬚ Alice receives 28 and calculates $28^{18} \pmod{47}$ which is 24, the secret key.

⬚ Bob receives 2 and calculates $2^{22} \pmod{47}$ which is 24, the same secret key

Man-in-the-Middle Attack on D-H protocol

Suppose Alice and Bob wish to exchange keys, and Darth is the adversary. The attack proceeds as follows:

1. Darth prepares for the attack by generating two random private keys X_{D1} and X_{D2} and then computing the corresponding public keys Y_{D1} and Y_{D2}.

2. Alice transmits Y_A to Bob.

3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculates $K2 = (Y_A)^{X_{D2}} \bmod q$.

4. Bob receives Y_{D1} and calculates $K1 = (Y_{D1})^{X_E} \bmod q$.

5. Bob transmits X_A to Alice.

6. Darth intercepts X_A and transmits Y_{D2} to Alice. Darth calculates $K1 = (Y_B)^{X_{D1}} \bmod q$.

7. Alice receives Y_{D2} and calculates $K2 = (Y_{D2})^{X_A} \bmod q$.

At this point, Bob and Alice think that they share a secret key, but instead Bob and Darthshare secret key K1 and Alice and Darth share secret key K2. All future communicationbetweenBob andAliceis compromisedinthefollowingway:

1. AlicesendsanencryptedmessageM:E(K2,M).

2. Darthinterceptstheencryptedmessageanddecryptsit,torecoverM.

3. DarthsendsBobE(K1,M)orE(K1,M'),whereM'isanymessage.Inthefirstcase,Darthsimply wants to eavesdrop on the communication without altering it. In the second case,Darthwantsto modify themessagegoingtoBob.

Thekeyexchangeprotocolisvulnerabletosuchanattackbecauseitdoesnotauthenticate the participants. This vulnerability can be overcome with the use of digitalsignaturesandpublic-key certificates.

ELLIPTICCURVECRYPTOGRAPHY(ECC)

Elliptic curve cryptography (ECC) is an approach to public-key cryptography based onthe algebraic structure of elliptic curves over finite fields. The use of elliptic curves incryptography was suggested independently by Neal Koblitz and Victor S. Miller in 1985.The principal attraction of ECC compared to RSA is that it appears to offer equal securityforafarsmallerbitsize,therebyreducingtheprocessingoverhead.

EllipticCurveoverGF(p)

LetGF(p)beafinitefield,p>3, andleta, b

☐ GF(p) are constant such that **4a3 + 27b2 ≡ 0 (mod p).** An elliptic curve,

E(a,b)(GF(p)),isdefinedastheset ofpoints(x,y) $\mathbf{C}$GF(p) *GF(p)whichsatisfytheequation

y2 ≡ x3 + ax + b (mod p), together with a special point, O, *called the point at infinity*. LetPandQbetwo points onE(a,b)(GF(p))andOisthepoint atinfinity.

• P+O= O+P =P

• IfP=(x1,y1)then-P= (x1,-y1)andP+(-P)= O.

• IfP=(x1,y1)andQ =(x2,y2),andPandQare notO.

thenP+Q=(x3,y3)where

x3=λ2- x1- x2

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & for \_ x_1 \neq x_2 \\ \dfrac{3x_1^2 + a}{2y_1} & for \_ x_1 = x_2 \end{cases}$$

86

y3=λ (x1- x3) -y1and

λ=(y2-y1)/(x2-x1)ifP≠Q

λ=(3x12+a)/2y1 ifP=Q

An elliptic curve may be defined over any finite field GF(q). For GF(2_m), the curve has adifferentform:- *y2+ xy=x3+ ax2+ b, whereb* !=0.

CryptographywithEllipticCurves

The addition operation in ECC is the counterpart of modular multiplication in RSA, andmultiple addition is the counterpart of modular exponentiation. To form a cryptographicsystemusingellipticcurves,somekindofhardproblemsuchasdiscretelogarithmorfac torization of prime numbers is needed. Considering the equation, Q=kP, where Q,P arepoints in an elliptic curve, it is "easy" to compute Q given k,P , but "hard" to find k given Q,P.This is known as the elliptic curve logarithm problem. k could be so large as to make brute-force fail.

ECCKey Exchange

Pickaprimenumberp=2180andellipticcurveparametersaandbfortheequation

y2 ≡ x3 + ax + b (mod p) which defines the elliptic group of points Ep(a,b). Selectgenerator point G=(x1,y1) in Ep(a,b) such that the smallest value for which nG=O be averylargeprimenumber.Ep(a,b)andGareparametersofthecryptosystemknowntoallparticip ants. The followingsteps takeplace:

• A&BselectprivatekeysnA<n,nB<n

• computepublickeys:PA=nA×G,PB=nB×G

• Computesharedkey:K=nA×PB,K=nB×PA{samesinceK=nA×nB×G}

ECC Encryption/Decryption As with key exchange system, an encryption/decryptionsystem requires a point G and and elliptic group Ep(a,b) as parameters. First thing to bedone is to encode the plaintext message m to be sent as an x-y point **Pm**. Each userchooses private key nA<n and computes public key PA=nA×G. To encrypt and send amessage to Pm to B, A chooses a random positive integer k and produces the

ciphertext**Cm**consistingofthepairofpoints**Cm={kG,Pm+kPb}.**here,AusesB'spublickey.To

decrypt the ciphertext, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point **Pm+kPb – nB(kG) = Pm+k(nBG) – nB(kG) =Pm** A has masked the message **Pm** by adding **kPb** to it. Nobody but A knows the value of k, so even though **Pb** is a public key, nobody can remove the mask k**Pb.** For an attacker to recover the message, he has to compute k given G and kG, which is assumed hard.

Security of ECC To **protect** a 128 bit AES key it would take a RSA Key Size of 3072 bits whereas an ECC Key Size of 256 bits.

Computational Effort for Cryptanalysis of Elliptic Curve Cryptography Compared to RSA

Key Size	MIPS-Years
150	3.8×10^{10}
205	7.1×10^{18}
234	1.6×10^{28}

Key Size	MIPS-Years
512	3×10^{4}
768	2×10^{8}
1024	3×10^{11}
1280	1×10^{14}
1536	3×10^{16}
2048	3×10^{20}

(a) Elliptic Curve Logarithms using the Pollard rho Method

(b) Integer Factorization using the General Number Field Sieve

Hence for similar security ECC offers significant computational advantages.

Applications of ECC:

- Wireless communication devices
- Smartcards
- Web servers that need to handle many encryption sessions
- Any application where security is needed but lacks the power, storage and computational power that is necessary for our current cryptosystems

KEY MANAGEMENT

One of the major roles of public-key encryption has been to address the problem of key distribution. Two distinct aspects to use of public key encryption are present.

 The distribution of public keys.

 Use of public-key encryption to distribute secret keys.

Distribution of Public Keys The most general schemes for distribution of public keys are given below

PUBLICANNOUNCEMENTOFPUBLICKEYS

Hereanyparticipantcansendhisorherpublickeytoanyotherparticipantorbroadcastthe key to the community at large. For example, many PGP users have adopted thepracticeofappendingtheirpublickeytomessagesthattheysendtopublicforums.

Uncontrolled Public-Key Distribution

Thoughthisapproachseemsconvenient,ithasamajordrawback.Anyonecanforgesucha public announcement. Some user could pretend to be user A and send a public key toanotherparticipantorbroadcastsuchapublickey.UntilthetimewhenAdiscoversaboutthefor geryandalertsotherparticipants,theforgerisabletoreadallencryptedmessagesintendedforAa ndcanusetheforgedkeysforauthentication.

PUBLICLYAVAILABLEDIRECTORY

Agreaterdegreeofsecuritycanbeachievedbymaintainingapubliclyavailabledynamicdirectory ofpublickeys.Maintenanceanddistributionofthepublicdirectorywouldhaveto be the responsibility of some trusted entity or organization. It includes the followingelements:

1. Theauthoritymaintainsadirectorywitha{name,publickey}entryforeachparticipant.

2. Eachparticipantregistersapublickeywiththedirectoryauthority.Registrationwouldhaveto bein personorbysomeformofsecureauthenticatedcommunication.
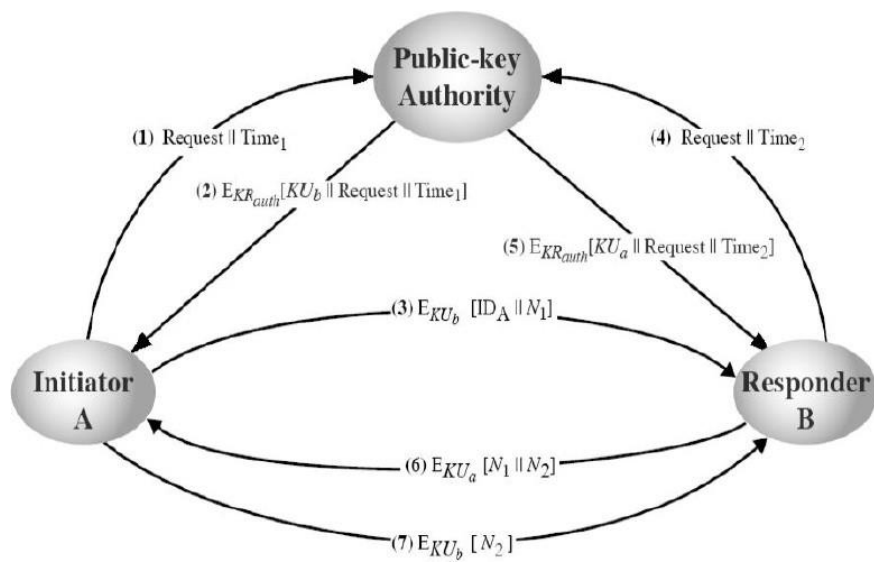
Public-Key Publication
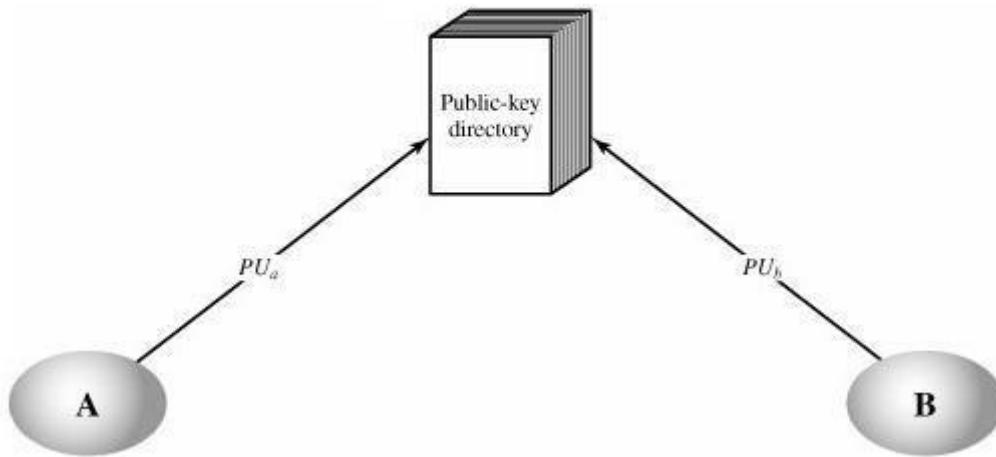
3. A participant may replace the existing key with a new one at any time, eitherbecauseofthedesiretoreplaceapublickeythathasalreadybeenusedforalargeamountofdata,orbecausethecorrespondingprivatekeyhasbeencompromisedinsome way.

4. Participantscouldalsoaccessthedirectoryelectronically.Forthispurpose,secure, authenticated communication from the authority to the participant ismandatory.Thisschemehasstillgotsomevulnerabilities.Ifanadversarysucceeds in obtaining or computing the private key of the directory authority, theadversarycouldauthoritativelypassoutcounterfeitpublickeysandsubsequentlyimpersonate any participant and eavesdrop on messages sent to any participant.Orelse, theadversary maytamperwiththerecordskeptbytheauthority.

PUBLIC-KEYAUTHORITY

Strongersecurityforpublic-keydistributioncanbeachievedbyprovidingtightercontroloverthedistributionofpublickeysfromthedirectory.Thisscenarioassumestheexistenceofapublicauthority(whoeverthatmaybe)thatmaintainsadynamicdirectoryof public keys of all users. The public authority has its own (private key, public key) thatit is using to communicate to users. Each participant reliably knows a public key for theauthority, with only the authority knowing the corresponding private key. For example,considerthatAliceandBobwishtocommunicatewitheachotherandthefollowingsteps takeplaceandarealsoshowninthe figure below:

1.) Alice sends a **timestamped** message to the central authority with a request for Bob's public key (the time stamp is to mark the moment of the request)

2.) The authority sends back a message encrypted with its private key (for authentication) – message contains Bob's public key and the original message of Alice –this way Alice knows this is not a reply to an old request;

3.) Alice starts the communication to Bob by sending him an encrypted message containing her identity ID_A and a nonce N_1 (to identify uniquely this transaction)

4.) Bob requests Alice's public key in the same way (step 1)

5.) Bob acquires Alice's public key in the same way as Alice did. (Step-2)

6.) Bob replies to Alice by sending an encrypted message with N_1 plus a new generated nonce N_2 (to identify uniquely the transaction)

7.) Alice replies once more encrypting Bob's nonce N_2 to assure bob that its correspondent is Alice

Thus, a total of seven messages are required. However, the initial four messages need be used only infrequently because both A and B can save the other's public key for future use, a technique known as caching. Periodically, a user should request fresh copies of the public keys of its correspondents to ensure currency.

PUBLIC-KEY CERTIFICATES

The above technique looks attractive, but still has some drawbacks. For any communication between any two users, the central authority must be consulted by both users to get the newest public keys i.e. the central authority must be online 24 hours/day. If the central authority goes offline, all secure communications get to a halt. This clearly leads to an undesirable bottleneck. A further improvement is to use certificates, which can be used to exchange keys without contacting a public-key authority, in a way that is as reliable as if the keys were obtained directly from a public-key authority. A certificate binds an **identity** to **public key**, with all contents **signed** by a trusted Public-Key or Certificate Authority (CA). A user can present his or her public key to the authority in a secure manner, and obtain a certificate. The user can then publish the certificate. Anyone needed this user's public key can obtain the certificate and verify that it is valid by way of the attached trusted signature. A participant can also convey its key information to another by transmitting its certificate. Other participants can verify that the certificate was created by the authority. This certificate issuing scheme does have the following requirements:

1. Any participant can read a certificate to determine the name and public key of the certificate's owner.

2. Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.

3. Only the certificate authority can create and update certificates.

4. Any participant can verify the currency of the certificate.
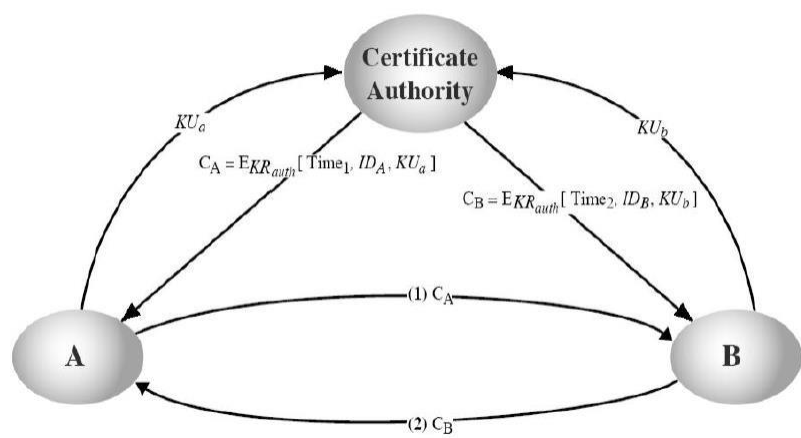
Application must be in person or by some form of secure authenticated communication. For participant A, the authority provides a certificate of the form
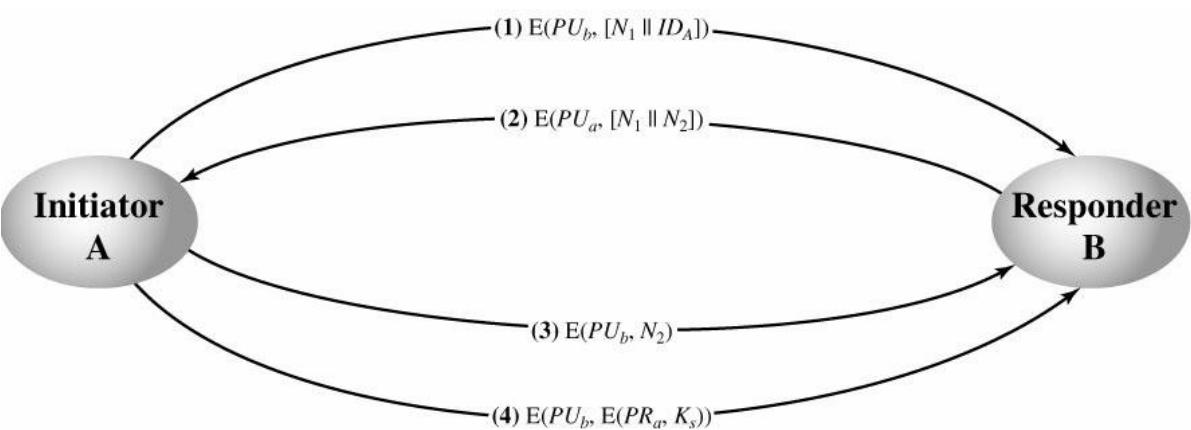
$C_A = E(PR_{auth}, [T||ID_A||PU_a])$ where PR_{auth} is the private key used by the authority and T is a timestamp. A may then pass this certificate on to any other participant, who reads and verifies the certificate as follows: $D(PU_{auth}, C_A) = D(PU_{auth}, E(PR_{auth}, [T||ID_A||PU_a]))$ $= (T||ID_A||PU_a)$ The recipient uses the authority's public key, PU_{auth} to decrypt the certificate.

Because the certificate is readable only using the authority's public key, this verifies that the certificate came from the certificate authority. The elements ID_A and PU_a provide the recipient with the name and public key of the certificate's holder. The timestamp T validates the currency of the certificate. The timestamp counters the following scenario. A's private key is learned by an adversary. A generates a new private/public key pair and applies to the certificate authority for a new certificate. Meanwhile, the adversary replays the old certificate to B. If B then encrypts messages using the compromised old public key, the adversary can read those messages. In this context, the compromise of a private key is comparable to the loss of a credit card. The owner cancels the credit card number but is at risk until all possible communicants are aware that the old credit card is obsolete. Thus, the timestamp serves as something like an expiration date. If a certificate is sufficiently old, it is assumed to be expired.

One scheme has become universally accepted for formatting public-key certificates: the X.509 standard. X.509 certificates are used in most network security applications, including IP security, secure sockets layer (SSL), secure electronic transactions (SET), and S/MIME.

SECRET KEY DISTRIBUTION WITH CONFIDENTIALITY AND AUTHENTICATION

It is assumed that A and B have exchanged public keys by one of the schemes described earlier. Then the following steps occur:

(1) $E(PU_b, [N_1 || ID_A])$

(2) $E(PU_a, [N_1 || N_2])$

Initiator A   Responder B

(3) $E(PU_b, N_2)$

(4) $E(PU_b, E(PR_a, K_s))$

1. A uses B's public key to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.

2. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2) Because only B could have decrypted message (1), the presence of N_1 in message (2) assures A that the correspondent is B.

3. A returns N_2 encrypted using B's public key, to assure B that its correspondent is A.

4. A selects a secret key K_s and sends $M = E(PU_b, E(PR_a, K_s))$ to B. Encryption of this message with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.

5. B computes $D(PU_a, D(PR_b, M))$ to recover the secret key.

The result is that this scheme ensures both confidentiality and authentication in the exchange of a secret key.