COMP 3355 Cyber Security

Assignment 1 Tutorial

Crida Wei

List of tools (java programs) provided

List of tools (java programs) provided

- **Analyser**

  > java Analyzer *<input-file>*

  Collect frequency distribution of *<input-file>* and calculate the index of coincidence

- **Factor**

  > java Factor *<number>*

  Output all the factors of *<number>*

- **Kasiski**

  > java Kasiski *<input-file> <number>*

  Arrange the text in *<input-file>* into a number columns and calculate the index of coincidence for each of them

- **PolyTranslator**

  > java PolyTranslator *<input-file> <number> <map-file 1> ... <map-file n>*

  Translate the text in *<input-file>* using multiple mapping files *<map-file 1>...<map-file n>* poly-alphabetically

- **Translator**

  > java Translator *<input-file> <map-file>*

  Translate the text in *<input-file>* using the mapping key in *<map-file>*

- **TryAllTranslators**

  > java TryAllTranslators *<input-file>*

  Translate the text in *<input-file>* using the mapping files *AA.txt ... ZZ.txt* one by one.

Monoalphabetic Ciphers Cryptanalysis - Example

1. Given the following cipher text input (*demo1.txt*):

fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

2. Analyze the cipher text by collecting frequency distribution, index of coincidence by running the Java program: *Analyzer.java*
> java Analyzer demo1.txt

You should see the following results:

```
C:\COMP3355>java Analyzer demo1.txt
reading file...
total number of letters = 48
index of Coincidence = 0.0673758865248227
frequency counting...
j 7 probability=0.14583333
n 7 probability=0.14583333
b 5 probability=0.104166664
w 4 probability=0.083333336
c 3 probability=0.0625
f 3 probability=0.0625
u 3 probability=0.0625
v 3 probability=0.0625
x 3 probability=0.0625
a 2 probability=0.041666668
h 2 probability=0.041666668
q 2 probability=0.041666668
d 1 probability=0.020833334
k 1 probability=0.020833334
m 1 probability=0.020833334
r 1 probability=0.020833334
```

3. Prepare mapping file that describes substitutions between cipher text alphabets and plain text alphabets. The format of the file is in the form of *C=P* where *C* is a cipher text alphabet and *P* is a plain text alphabet. In the given example, if we make a wild guess that cipher text j maps to plain text e, we can then try to see if any regular arrangement happen in the mapping, such as 10 (j) →5 (e), 9 (i) → 4 (d), 11 (k) → 6(f), …, etc. That is *c = (p+5) mod 26*. Then, we can prepare a text file *map_demo1.txt* to include some mappings (e.g. popular letters and vowels letters) for trial guess, such as:

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

4. Now, we can perform trial decryption based on the cipher text file *demo1.txt* and mapping
file *map-demo1.txt* by using the given Java program: Translator.java

> java Translator demo1.txt map_demo1.txt

```
C:\COMP3355>java Translator demo1.txt map_demo1.txt
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
a-e-- --e-e -ie-- -ifce -c--- i--e- ia--- h--i- -e--a ii-
```

Demonstration of Caesar ciphers cryptanalysis

- We encode 26 Caesar ciphers into 26 different mapping files. They are named as *AA.txt*, *BB.txt*, …, *ZZ.txt*. For example, the mapping file *AA.txt* encodes the Caesar cipher $c = (p+0) \bmod 26$ while the mapping file *BB.txt* encodes the Caesar cipher $c = (p+1) \bmod 26$, etc.
- We can use the program *TryAllTranslator* to try all 26 Caesar ciphers. The first command line parameter denotes the cipher text while the second parameter denotes the path
  where to locate the 26 mapping files (it is '.' for current directory). The following figure shows several trials run of the program.

> java TryAllTranslator demo1.txt .

```
C:\COMP3355>java TryAllTranslator demo1.txt .
Error: Could not find or load main class TryAllTranslator

C:\COMP3355>java TryAllTranslators demo1.txt .
Try mapping file .\AA.txt (i=0).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc
Continue [Y/N]? Y
Try mapping file .\BB.txt (i=1).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
epiba qvivi umizw amjgi vgwbp mzviu mewct laumt tiaae mmb
Continue [Y/N]? Y
Try mapping file .\CC.txt (i=2).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
dohaz puhuh tlhyv zlifh ufvao lyuht ldvbs kztls shzzd lla
Continue [Y/N]? Y
Try mapping file .\DD.txt (i=3).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
cngzy otgtg skgxu ykheg teuzn kxtgs kcuar jyskr rgyyc kkz
Continue [Y/N]? Y
Try mapping file .\EE.txt (i=4).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
bmfyx nsfsf rjfwt xjgdf sdtym jwsfr jbtzq ixrjq qfxxb jjy
Continue [Y/N]? Y
Try mapping file .\FF.txt (i=5).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
alexw mrere qievs wifce rcsxl ivreq iasyp hwqip pewwa iix
Continue [Y/N]? Y
```

```
Try mapping file .\GG.txt (i=6).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
zkdwv lqdqd phdur vhebd qbrwk huqdp hzrxo gvpho odvvz hhw
Continue [Y/N]? Y
Try mapping file .\HH.txt (i=7).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
yjcvu kpcpc ogctq ugdac paqvj gtpco gyqwn fuogn ncuuy ggv
Continue [Y/N]? Y
Try mapping file .\II.txt (i=8).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
xibut jobob nfbsp tfczb ozpui fsobn fxpvm etnfm mbttx ffu
Continue [Y/N]? Y
Try mapping file .\JJ.txt (i=9).......
ciphertext:
fqjcb rwjwj vnjax bnkhj whxcq nawjv nfxdu mbvnu ujbbf nnc

plaintext :
whats inana mearo sebya nyoth ernam ewoul dsmel lassw eet
Continue [Y/N]?
```

Recovered Plaintext!

Polyalphabetic Ciphers Cryptanalysis - Example

1. Given the following cipher text input (*demo2.txt*):

ZHYME ZVELK OJUBW CEYIN CUSML RAVSR YARNH CEARI UJPGP VARDU QZCGR NNCAW JALUH GJPJR YGEG**Q** FULUS QFFPV EYEDQ GOLKA LVOSJ TFRTR YEJZS RVNCI HYJNM ZDCRO DKHCR MMLNR FFLFN QGOLK ALVOS JWMIK QKUBP SAYOJ RRQYI NRNYC YQZSY EDNCA LEILX RCHUG IEBKO YTHGV VCKHC JEQGO LKALV OSJED WEAKS GJHYC LLFTY IGSVT FVPMZ NRZOL CYUZS FKOQR YRYAR ZFGKI QKRSV IRCEY USKVT MKHCR MYQIL **X**RCRL GQARZ OLKHY KSNFN RRNCZ TWUOC JNMKC MDEZP IRJEJ W

2. Analyze the cipher text by collecting frequency distribution, index of coincidence by
running the Java program: *Analyzer.java*

> java Analyzer demo2.txt

You should see the following results:

```
C:\COMP3355>java Analyzer demo2.txt
reading file...
total number of letters = 346
index of Coincidence = 0.043377273309876854
frequency counting...
r 30 probability=0.0867052
c 21 probability=0.06069364
y 21 probability=0.06069364
l 20 probability=0.057803467
k 18 probability=0.05202312
e 17 probability=0.049132947
j 16 probability=0.046242774
n 16 probability=0.046242774
s 15 probability=0.0433526
a 14 probability=0.040462427
g 14 probability=0.040462427
o 14 probability=0.040462427
v 14 probability=0.040462427
q 13 probability=0.037572253
z 13 probability=0.037572253
f 12 probability=0.03468208
i 12 probability=0.03468208
u 12 probability=0.03468208
h 11 probability=0.031791907
m 11 probability=0.031791907
d 7 probability=0.020231213
p 7 probability=0.020231213
t 7 probability=0.020231213
w 6 probability=0.01734104
b 3 probability=0.00867052
x 2 probability=0.0057803467
```

```
C:\COMP3355>java Analyzer demo2.txt 15
reading file...
total number of letters = 346
index of Coincidence = 0.043377733098768 54
frequency counting...
r 30 probability=0.0867052
c 21 probability=0.06069364
y 21 probability=0.06069364
l 20 probability=0.057803467
k 18 probability=0.05202312
e 17 probability=0.049132947
j 16 probability=0.046242774
n 16 probability=0.046242774
s 15 probability=0.0433526
a 14 probability=0.040462427
g 14 probability=0.040462427
o 14 probability=0.040462427
v 14 probability=0.040462427
q 13 probability=0.037572253
z 13 probability=0.037572253
f 12 probability=0.03468208
i 12 probability=0.03468208
u 12 probability=0.03468208
h 11 probability=0.031791907
m 11 probability=0.031791907
d 7 probability=0.020231213
p 7 probability=0.020231213
t 7 probability=0.020231213
w 6 probability=0.01734104
b 3 probability=0.00867052
x 2 probability=0.0057803467
--- 2-grams counting ----------
hy 4 probability=0.00591716
ez 2 probability=0.00295858
ve 2 probability=0.00295858
lk 5 probability=0.00739645
ko 3 probability=0.00443787
oj 2 probability=0.00295858
ub 2 probability=0.00295858
ce 3 probability=0.00443787
ey 3 probability=0.00443787
yi 3 probability=0.00443787
in 2 probability=0.00295858
nc 5 probability=0.00739645
us 3 probability=0.00443787
ml 2 probability=0.00295858
sr 2 probability=0.00295858
ry 5 probability=0.00739645
ya 2 probability=0.00295858
ar 5 probability=0.00739645
rn 4 probability=0.00591716
hc 4 probability=0.00591716
ea 2 probability=0.00295858
jp 2 probability=0.00295858
pv 2 probability=0.00295858
qz 2 probability=0.00295858
nn 1 probability=0.00147929
ca 2 probability=0.00295858
al 5 probability=0.00739645
lu 2 probability=0.00295858
hg 2 probability=0.00295858
gj 2 probability=0.00295858
jr 2 probability=0.00295858
gq 2 probability=0.00295858
qf 2 probability=0.00295858
```

```
--- 5-grams counting ----------
qgolk 3 position=90,141,213
golka 3 position=91,142,214
olkal 3 position=92,143,215
lkalv 3 position=93,144,216
kalvo 3 position=94,145,217
alvos 3 position=95,146,218
lvosj 3 position=96,147,219
khcrm 2 position=127,292
ilxrc 2 position=188,299
--- 6-grams counting ----------
qgolka 3 position=90,141,213
golkal 3 position=91,142,214
olkalv 3 position=92,143,215
lkalvo 3 position=93,144,216
kalvos 3 position=94,145,217
alvosj 3 position=95,146,218
--- 7-grams counting ----------
qgolkal 3 position=90,141,213
golkalv 3 position=91,142,214
olkalvo 3 position=92,143,215
lkalvos 3 position=93,144,216
kalvosj 3 position=94,145,217
--- 8-grams counting ----------
qgolkalv 3 position=90,141,213
golkalvo 3 position=91,142,214
olkalvos 3 position=92,143,215
lkalvosj 3 position=93,144,216
--- 9-grams counting ----------
qgolkalvo 3 position=90,141,213
golkalvos 3 position=91,142,214
olkalvosj 3 position=92,143,215
--- 10-grams counting ----------
qgolkalvos 3 position=90,141,213
golkalvosj 3 position=91,142,214
--- 11-grams counting ----------
qgolkalvosj 3 position=90,141,213
--- 12-grams counting ----------
--- 13-grams counting ----------
--- 14-grams counting ----------
--- 15-grams counting ----------
```

ates that the
Therefore,
cations and
locate the
butions (for
more

4. By looking at the longest repeated patterns and following the distances between the repeated sequences, we can determine the possible key lengths that used for the substitutions. In this
example, the longest repeated pattern is *qgolkalvosj* of length 11 and the distance between the first 2 sequences is 141-90 = 51, while the distance between the second 2 sequences is 213 – 141 = 72. Their factors can be easily figured out by running program *Factor*:

> java Factor 51
> java Factor 72

```
C:\COMP3355>java Factor 51
Factor of 51 include:
3 17 51

C:\COMP3355>java Factor 72
Factor of 72 include:
2 3 4 6 8 9 12 18 24 36 72
```

5. Since the common divisor between 51 and 72 is 3, we are check if the ciphertext is
encrypted with 3 monoalphabetic ciphers individually by calculating their index of
coincidence individually. To do this, we can use the program *Kasiski*:

> java Kasiski demo2.txt 3

```
C:\COMP3355>java Kasiski demo2.txt 3
reading file...
total number of letters = 346
index of Coincidence = 0.043377273309876854
frequency counting...
r 30 probability=0.0867052
c 21 probability=0.06069364
y 21 probability=0.06069364
l 20 probability=0.057803467
k 18 probability=0.05202312
e 17 probability=0.049132947
j 16 probability=0.046242774
n 16 probability=0.046242774
s 15 probability=0.0433526
a 14 probability=0.040462427
g 14 probability=0.040462427
o 14 probability=0.040462427
v 14 probability=0.040462427
q 13 probability=0.037572253
z 13 probability=0.037572253
f 12 probability=0.03468208
i 12 probability=0.03468208
u 12 probability=0.03468208
h 11 probability=0.031791907
m 11 probability=0.031791907
d 7 probability=0.020231213
p 7 probability=0.020231213
t 7 probability=0.020231213
w 6 probability=0.01734104
b 3 probability=0.00867052
x 2 probability=0.0057803467
------ sequence 1-------------
Cipher:
z--m- -v--k --u-- c--i- -u--l --v-- y--n- -e--i --p-- v--d-
-z--r --c-- j--u- -j--r --e-- f--u- -f--v --e-- g--k- -v--j
--r-- y--z- -v--i --j-- z--r- -k--r --l-- f--f- -g--k --v--
j--i- -k--p --y-- r--y- -r--c --z-- e--c- -e--x --h-- i--k-
-t--v --k-- j--g- -k--v --j-- w--k- -j--c --f-- i--v- -v--z
--z-- c--z- -k--r --y-- z--k- -k--v --c-- u--v- -k--r --q--
x--r- -q--z --k-- k--f- -r--z --u-- j--k- -d--p --j-- w
number of letters = 116
index of coincidence = 0.06746626686656672
k 16 probability=0.13793103
v 13 probability=0.112068966
r 11 probability=0.094827585
z 11 probability=0.094827585
j 10 probability=0.0862069
c 7 probability=0.060344826
f 6 probability=0.05172414
i 6 probability=0.05172414
u 6 probability=0.05172414
e 5 probability=0.04310345
y 5 probability=0.04310345
g 3 probability=0.02586207
p 3 probability=0.02586207
d 2 probability=0.01724138
l 2 probability=0.01724138
q 2 probability=0.01724138
w 2 probability=0.01724138
x 2 probability=0.01724138
h 1 probability=0.00862069
m 1 probability=0.00862069
n 1 probability=0.00862069
t 1 probability=0.00862069
------ sequence 2-------------
Cipher:
-h--e --e-- o--b- -e--n --s-- r--s- -a--h --a-- u--g- -a--u
--c-- n--a- -a--h --p-- y--g- -u--s --f-- e--d- -o--a --o--
t--t- -e--s --n-- h--n- -d--o --h-- m--n- -f--n --o-- a--o-
-w--k --u-- s--o- -r--i --n-- y--s- -d--a --i-- r--u- -e--o
--h-- v--h- -e--o --a-- o--e- -e--s --h-- l--t- -g--t --p--
n--o- -y--s --o-- y--a- -f--i --r-- i--e- -s--t --h-- m--i-
-r--l --a-- o--h- -s--n --n-- t--o- -n--c --e-- i--e- -
number of letters = 115
index of coincidence = 0.06498855835240275
o 14 probability=0.12173913
e 12 probability=0.104347825
a 11 probability=0.09565217
n 11 probability=0.09565217
h 10 probability=0.08695652
s 10 probability=0.08695652
i 6 probability=0.052173913
t 6 probability=0.052173913
r 5 probability=0.04347826
u 5 probability=0.04347826
y 4 probability=0.034782607
d 3 probability=0.026086956
f 3 probability=0.026086956
g 3 probability=0.026086956
c 2 probability=0.017391304
l 2 probability=0.017391304
m 2 probability=0.017391304
p 2 probability=0.017391304
b 1 probability=0.008695652
k 1 probability=0.008695652
v 1 probability=0.008695652
w 1 probability=0.008695652
------ sequence 3-------------
Cipher:
--y-- z--l- -j--w --y-- c--m- -a--r --r-- c--r- -j--p --r--
q--g- -n--w --l-- g--j- -g--q --l-- q--p- -y--q --l-- l--s-
-f--r --j-- r--c- -y--m --c-- d--c- -m--r --l-- q--l- -l--s
--m-- q--b- -a--j --q-- n--y- -q--y --n-- l--l- -c--g --b--
y--g- -c--c --q-- l--l- -s--d --a-- g--y- -l--y --s-- f--m-
-r--l --u-- f--q- -r--r --g-- q--s- -r--y --k-- m--c- -y--l
--c-- g--r- -l--y --n-- r--c- -w--c --m-- m--z- -r--j -
number of letters = 115
index of coincidence = 0.07597254004576659
l 16 probability=0.13913043
r 14 probability=0.12173913
c 12 probability=0.104347825
y 12 probability=0.104347825
q 11 probability=0.09565217
g 8 probability=0.069565214
m 8 probability=0.069565214
j 6 probability=0.052173913
s 5 probability=0.04347826
n 4 probability=0.034782607
a 3 probability=0.026086956
f 3 probability=0.026086956
w 3 probability=0.026086956
b 2 probability=0.017391304
d 2 probability=0.017391304
p 2 probability=0.017391304
z 2 probability=0.017391304
k 1 probability=0.008695652
u 1 probability=0.008695652
```

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

6. By observing the index of coincidence (IC), all sequences have IC close to or greater than 0.068. We are pretty sure that the cipher text is encrypted by polyalphabetic cipher. We can then prepare individual mapping files for each character sequence as in the case of monoalphabetic cipher cryptanalysis. In this example, we prepare 3 map files because the key length suggested by the *Kasiski* method is 3. Suppose we guess the first sequence is encrypted using the letter 'R' in *Vigenere Tableau*, and we still don't know the remaining two letters. We can prepare use the mapping file *RR.txt*, and use *empty.txt* as remaining mapping files.

> java PolyTranslator demo2.txt 3 RR.txt empty.txt empty.txt

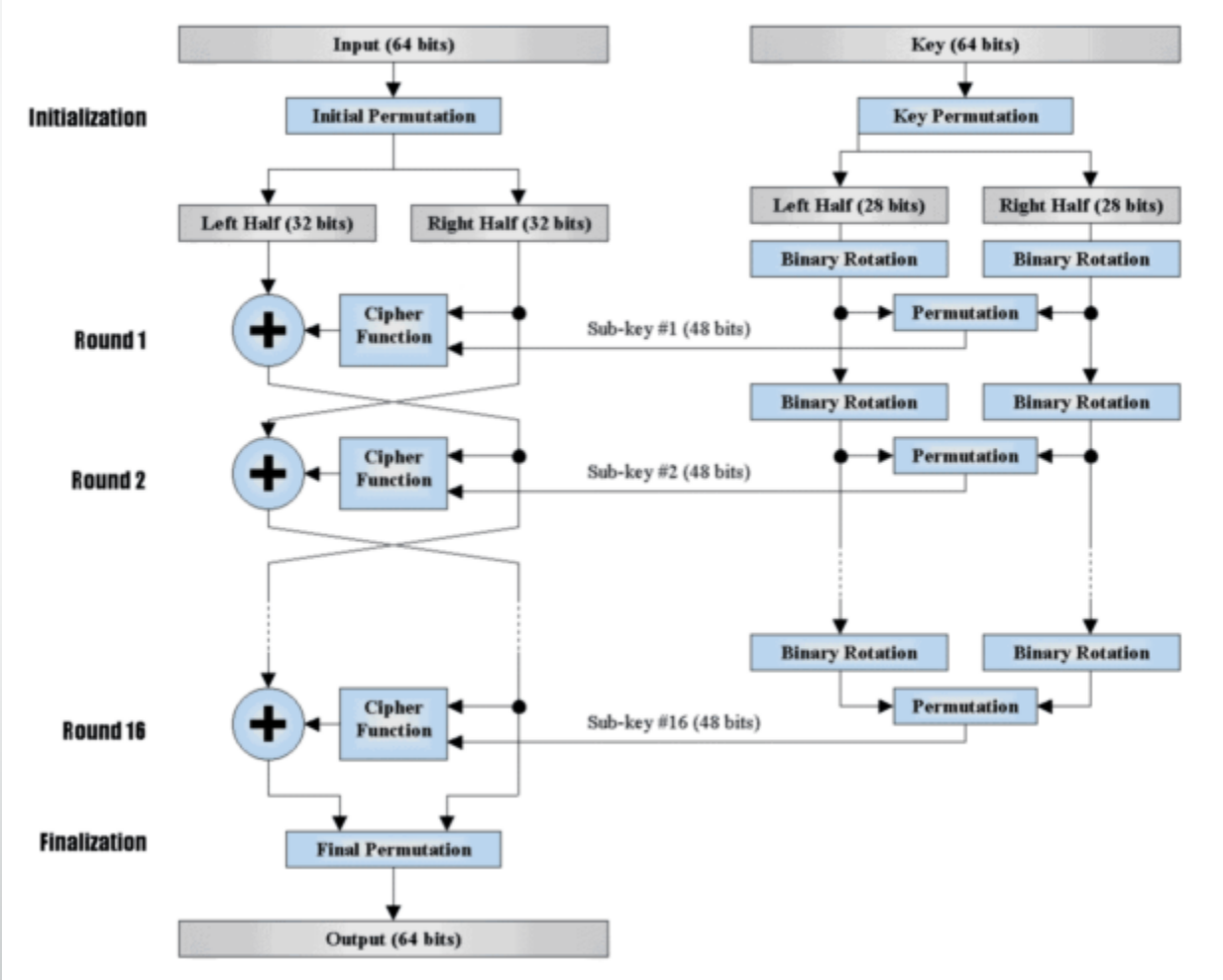
```
C:\COMP3355>java PolyTranslator demo2.txt 3 RR.txt empty.txt empty.txt
num_seq=3
mapfile 1=RR.txt
mapfile 2=empty.txt
mapfile 3=empty.txt
plaintext by mapfile1 :
i--v- -e--t --d-- l--r- -d--u --e-- h--w- -n--r --y-- e--m-
-i--a --l-- s--d- -s--a --n-- o--d- -o--e --n-- p--t- -e--s
--a-- h--i- -e--r --s-- i--a- -t--a --u-- o--o- -p--t --e--
s--r- -t--y --h-- a--h- -a--l --i-- n--l- -n--g --q-- r--t-
-c--e --t-- s--p- -t--e --s-- f--t- -s--l --o-- r--e- -e--i
--i-- l--i- -t--a --h-- i--t- -t--e --l-- d--e- -t--a --z--
g--a- -z--i --t-- t--o- -a--i --d-- s--t- -m--y --s-- f
plaintext by mapfile2 :
i--v- -e--t --d-- l--r- -d--u --e-- h--w- -n--r --y-- e--m-
-i--a --l-- s--d- -s--a --n-- o--d- -o--e --n-- p--t- -e--s
--a-- h--i- -e--r --s-- i--a- -t--a --u-- o--o- -p--t --e--
s--r- -t--y --h-- a--h- -a--l --i-- n--l- -n--g --q-- r--t-
-c--e --t-- s--p- -t--e --s-- f--t- -s--l --o-- r--e- -e--i
--i-- l--i- -t--a --h-- i--t- -t--e --l-- d--e- -t--a --z--
g--a- -z--i --t-- t--o- -a--i --d-- s--t- -m--y --s-- f
plaintext by mapfile3 :
i--v- -e--t --d-- l--r- -d--u --e-- h--w- -n--r --y-- e--m-
-i--a --l-- s--d- -s--a --n-- o--d- -o--e --n-- p--t- -e--s
--a-- h--i- -e--r --s-- i--a- -t--a --u-- o--o- -p--t --e--
s--r- -t--y --h-- a--h- -a--l --i-- n--l- -n--g --q-- r--t-
-c--e --t-- s--p- -t--e --s-- f--t- -s--l --o-- r--e- -e--i
--i-- l--i- -t--a --h-- i--t- -t--e --l-- d--e- -t--a --z--
g--a- -z--i --t-- t--o- -a--i --d-- s--t- -m--y --s-- f
ciphertext:
zhyme zvelk ojubw ceyin cusml ravsr yarnh ceari ujpgp vardu
qzcgr nncaw jaluh gjpjr ygegq fulus qffpv eyedq golka lvosj
tfrtr yejzs rvnci hyjnm zdcro dkhcr mmlnr fflfn qgolk alvos
jwmik qkubp sayoj rrqyi nrnyc yqzsy ednca leilx rchug iebko
ythgv vckhc jeqgo lkalv osjed weaks gjhyc llfty igsvt fvpmz
nrzol cyuzs fkoqr yryar zfgki qkrsv ircey uskvt mkhcr myqil
xrcrl gqarz olkhy ksnfn rrncz twuoc jnmkc mdezp irjej w
plaintext :
i--v- -e--t --d-- l--r- -d--u --e-- h--w- -n--r --y-- e--m-
-i--a --l-- s--d- -s--a --n-- o--d- -o--e --n-- p--t- -e--s
--a-- h--i- -e--r --s-- i--a- -t--a --u-- o--o- -p--t --e--
s--r- -t--y --h-- a--h- -a--l --i-- n--l- -n--g --q-- r--t-
-c--e --t-- s--p- -t--e --s-- f--t- -s--l --o-- r--e- -e--i
--i-- l--i- -t--a --h-- i--t- -t--e --l-- d--e- -t--a --z--
g--a- -z--i --t-- t--o- -a--i --d-- s--t- -m--y --s-- f
```

```
C:\COMP3355>java PolyTranslator demo2.txt 3 RR.txt AA.txt YY.txt
num_seq=3
mapfile 1=RR.txt
mapfile 2=AA.txt
mapfile 3=YY.txt
plaintext by mapfile1 :
i--v- -e--t --d-- l--r- -d--u --e-- h--w- -n--r --y-- e--m-
-i--a --l-- s--d- -s--a --n-- o--d- -o--e --n-- p--t- -e--s
--a-- h--i- -e--r --s-- i--a- -t--a --u-- o--o- -p--t --e--
s--r- -t--y --h-- a--h- -a--l --i-- n--l- -n--g --q-- r--t-
-c--e --t-- s--p- -t--e --s-- f--t- -s--l --o-- r--e- -e--i
--i-- l--i- -t--a --h-- i--t- -t--e --l-- d--e- -t--a --z--
g--a- -z--i --t-- t--o- -a--i --d-- s--t- -m--y --s-- f
plaintext by mapfile2 :
ih-ve -ee-t o-db- le-rn -ds-u r-es- ha-wh -na-r u-yg- ea-mu
-ic-a n-la- sa-dh -sp-a y-ng- ou-ds -of-e e-nd- po-ta -eo-s
t-at- he-is -en-r h-sn- id-ao -th-a m-un- of-on -po-t a-eo-
sw-rk -tu-y s-ho- ar-hi -an-l y-is- nd-la -ni-g r-qu- re-to
-ch-e v-th- se-po -ta-e o-se- fe-ts -sh-l l-ot- rg-et -ep-i
n-io- ly-is -to-a y-ha- if-ti -tr-e i-le- ds-et -th-a m-zi-
gr-al -za-i o-th- ts-on -an-i t-do- sn-tc -me-y i-se- f
plaintext by mapfile3 :
ihave beent oldby learn edsou rcest hatwh enatr ulygr eatmu
sicia nplay sandh ispla yings ounds sofre eands ponta neous
thatt helis tener hasno ideao fthea mount ofnon spont aneou
swork study schol arshi panal ysisa ndpla nning requi redto
achie vethe sespo ntane ousef fects ishal lnota rguet hepoi
ntion lywis htosa ythat ifiti strue itlea dsmet othea mazin
greal izati ontha tspon tanei tydoe snotc omeby itsel f
ciphertext:
zhyme zvelk ojubw ceyin cusml ravsr yarnh ceari ujpgp vardu
qzcgr nncaw jaluh gjpjr ygegq fulus qffpv eyedq golka lvosj
tfrtr yejzs rvnci hyjnm zdcro dkhcr mmlnr fflfn qgolk alvos
jwmik qkubp sayoj rrqyi nrnyc yqzsy ednca leilx rchug iebko
ythgv vckhc jeqgo lkalv osjed weaks gjhyc llfty igsvt fvpmz
nrzol cyuzs fkoqr yryar zfgki qkrsv ircey uskvt mkhcr myqil
xrcrl gqarz olkhy ksnfn rrncz twuoc jnmkc mdezp irjej w
plaintext :
ihave beent oldby learn edsou rcest hatwh enatr ulygr eatmu
sicia nplay sandh ispla yings ounds sofre eands ponta neous
thatt helis tener hasno ideao fthea mount ofnon spont aneou
swork study schol arshi panal ysisa ndpla nning requi redto
achie vethe sespo ntane ousef fects ishal lnota rguet hepoi
ntion lywis htosa ythat ifiti strue itlea dsmet othea mazin
greal izati ontha tspon tanei tydoe snotc omeby itsel f
```

DES details

DES

- DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration



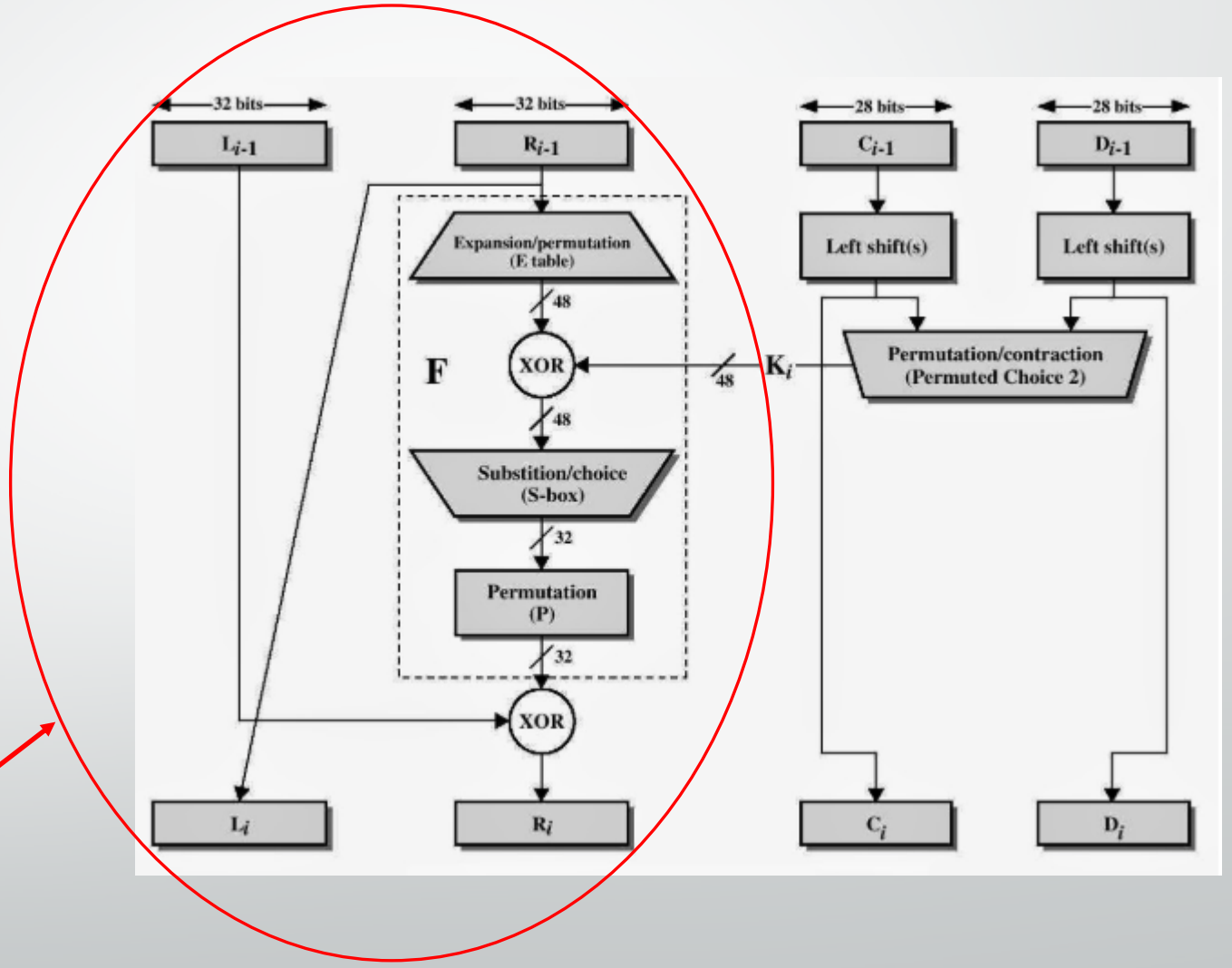
Initial Permutation

- Initial Permutation takes the plaintext as input. The table consists of 64 bits numbered from 1 to 64:

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

Round Function

- The heart of this cipher is the DES function, *f*. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



Feistel Structure

Expansion Permutation

- Since right input is 32-bit and round key is a 48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration

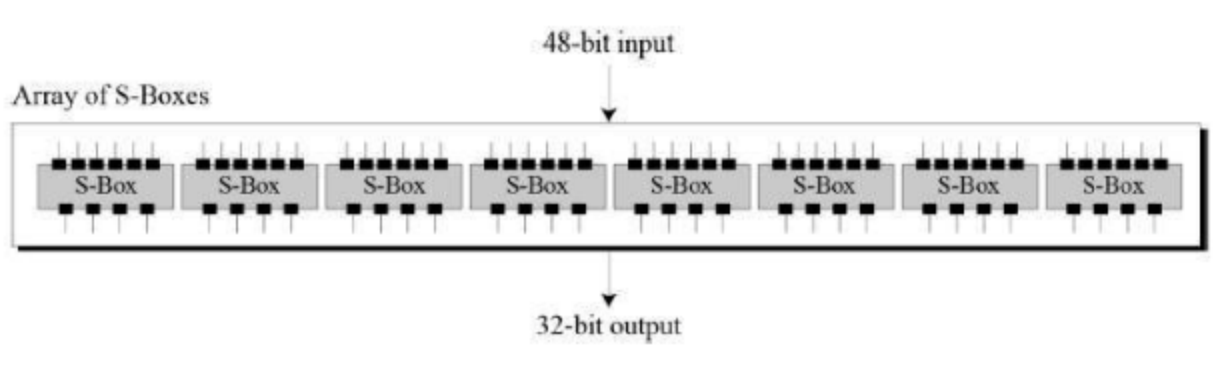
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

- **XOR**

– After the expansion permutation, DES does XOR operation on the expanded right section and the round key.

Substitution Choice (S-Box)

- The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. Refer the following illustration



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

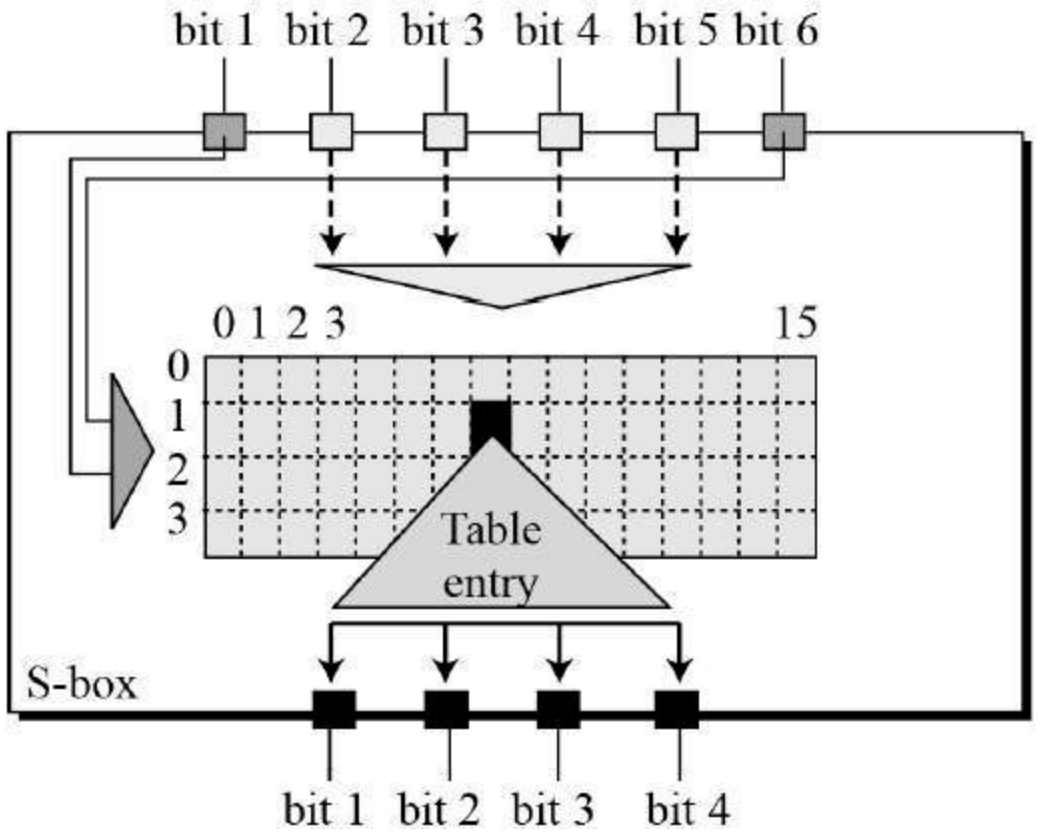
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

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



Permutation (P-Box)

- The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

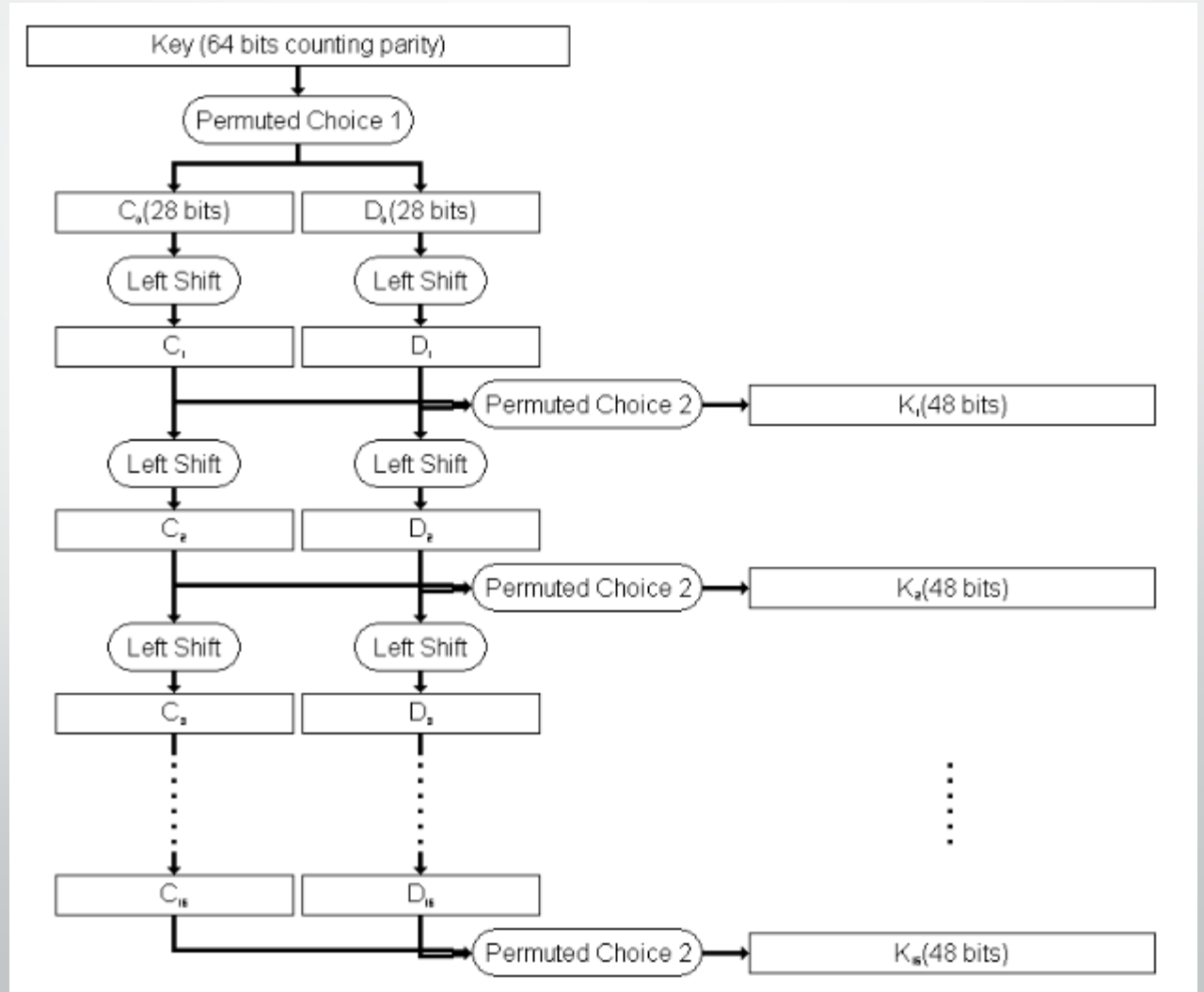
16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

Key Generation

- The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration

Shifting

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



permutation Choice one (PC-1)

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

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

- The output is separated into two 28 bits C and D. The first 28 bits are called C_0 (left part) and the last 28 bits are called D_0.
- At each round, a circular left shift is preformed on C_i and D_i by 1 or 2 bits. See the table below:

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

permutation choice two (PC-2)

- Ignore the 9, 18, 22, 25, 35, 38, 43, 54 bit of 56bits key.

The remaining 48bits key should be permuted by the following table.

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

The permutation Choice Two output in each round is uses as input to the encryption algorithm.

Inverse Initial Permutation

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

Reference

- https://www.tutorialspoint.com/cryptography/data_encryption_standard.htm
- https://www.cybrary.it/op3n/des-data-encryption-standard/

Thanks!

Q&A