16 Modern Encryption

ENCRYPTION

In the mid-Twentieth Century encryption had to be done using mechanical devices such as the Enigma machine. Now we have powerful computers with software which allows us to encrypt quickly. Modern encryption schemes tend to be optimised for use in software or in devices like the chips on credit cards or SIM cards in mobile phones.

We will look at a simplified example of a modern encryption scheme, based on 'Rijndael' (also known as the Advanced Encryption Standard or AES) which is often used in smartcards and on the internet.

First we think of our message (or *plaintext*) as being a 4×4 grid and we write it in columns. The plaintext

HI HERE IS A MESSAGE

becomes

H	R	A	S
I	E	M	A
H	I	E	G
E	S	S	E

We then put the plaintext through 4 different encryption steps.

STEP 1: substitution

The first step is a *substitution*: each letter is changed into another letter as in a normal substitution cipher. In this example we will use the Caesar substitution with each letter of the alphabet being shifted along by 3. As well as the alphabet we will include a few punctuation characters.

Plaintext	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**
Caesar	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
Plaintext	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**	**,**	**.**	**?**	**'**	**!**	**_**
Caesar	T	U	V	W	X	Y	Z	,	.	?	'	!	_	A	B	C

So our message becomes

K	U	D	V
L	H	P	D
K	L	H	J
H	V	V	H

Exercise 1

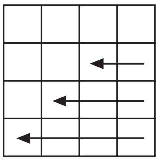
Encrypt the message

MEET ME HERE AT NINE

in a similar way. What do you get after the substitution step to this message?

STEP 2: row shift

The second step involves shifting the rows of the grid as shown in the diagram below: the first row does not shift at all; the second row shift let by 1; the third row shifts left by 2; the bottom row shifts left by 3.



So our message becomes

K	U	D	V
H	P	D	L
H	J	K	L
H	H	V	V

Exercise 2

What do you get when you have completed the row shifting step to your message?

STEP 3: column transformation

The next step is a little more complicated – it's a transformation of each column. We take each column separately and replace it with one where each entry is the sum of all the other entries in the column.

So, for example, if we have the column

A
B
C
D

we will replace it with the column

B+C+D
A+C+D
A+B+D
A+B+C

For this to make sense we need to define what we mean by 'adding' letters together. First we turn each letter into 5 binary digits ('0's or '1's) like this:

Letter	**Binary**	**Letter**	**Binary**	**Letter**	**Binary**	**Letter**	**Binary**
<space>	00000	H	01000	P	10000	X	11000
A	00001	I	01001	Q	10001	Y	11001
B	00010	J	01010	R	10010	Z	11010
C	00011	K	01011	S	10011	,	11011
D	00100	L	01100	T	10100	.	11100
E	00101	M	01101	U	10101	?	11101
F	00110	N	01110	V	10110	'	11110
G	00111	O	01111	W	10111	!	11111

Next we add together the binary digits according to these rules:

0 + 0 = 1	0 + 1 = 1
1 + 0 = 1	1 + 1 = 0

So E + F = 00101 + 00110 = 00011 = C

and A + B + C = 00001 + 00010 + 00011 = 00011 + 00011 = 00000 = < space >

[This type of 'addition' is officially known as *exclusive OR-ing,* and is an operation often performed by computers.]

Applying this to our message, the top-left entry is

H + H + H = 01000 + 01000 + 01000 = 01000 = H

The next entry along is

P + J + H = 10000 + 01010 + 01000 = 10010 = R

and so on.

Continuing in this way we get

H	R	Y	V
K	W	Y	L
K	M	V	L
K	O	K	V

Exercise 3

Apply the column transformation step to your message. What do you get?

STEP 4: add key

The final step is to add a *key* or *cryptovariable* to the entire outcome (using the same type of addition as before).

So using the key CRYPTOVARIABLEXX, our message becomes

H	R	Y	V		C	T	R	L		K	F	K	Z
K	W	Y	L	+	R	O	I	E	=	Y	X	P	I
K	M	V	L		Y	V	A	X		R	,	W	T
K	O	K	V		P	A	B	X		,	N	I	N

because H + C = 01000 + 00011 = 01011 = K, etc. So our message

HI HERE IS A MESSAGE

has become

KYR, FX,N KPWI ZITN

Exercise 4

Add the cryptovariable OURSECRETMEETING *to your outcome from Exercise 3.*

DECRYPTION

Suppose we receive the enciphered message

'FRA WYHB ,TC, BLUO

and we know that the cryptovariable is

OURTOPSECTRETKEYX

How do we decrypt the message?

We simply work backwards through the 4 steps and reverse them.

We start by writing the ciphertext and the cryptovariable in a 4×4 grid, and we 'add' the cryptovariable to the ciphertext (the type of 'addition' we are doing is *self-inverse* – to reverse it we simply do it again).

'	W	,	B		O	O	C	K		Q	?	?	?
F	Y	T	L	+	U	P	R	E	=	S	?	?	?
R	H	C	U		R	S	E	Y		_	?	?	?
A	B	,	O		T	E	T	X		U	?	?	?

So the top left entry is

' + O = 11110 + 01111 = 10001 = Q

The next one down is

F + U = 00110 + 10101 = 10011 = S

Then

R + R = 10010 + 10010 = 00000 = _

and

A + T = 00001 + 10100 = 10101 = U

Exercise 5

Continue adding the cryptovariable to the ciphertext. What do you get?

The column transformation is also self-inverse. To reverse it we just do the same thing again.

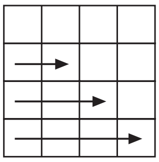
So the first column becomes

Q		S+_+U		00110		F
S	→	Q+_+U	→	00100	→	D
_		Q+S+U		10111		W
U		Q+S+_		00010		B

Exercise 6

Reverse the column transformation for the other 3 columns.

To reverse the row shifting operation we simply shift them back in the opposite direction. So the top row does not move, the second row shifts right by one square, the third row shifts right by 2 squares and the bottom row shifts right by 3 squares.



Exercise 7

Reverse the row shifting step in your message. What do you get?

Now we just have to reverse the substitution step – this is just like decrypting a Caesar enciphered message. Easy!

Exercise 8

Decode the message by undoing the substitution step. What is the message?

In practice, to make the method secure, we have to do these 4 steps 10 times to get the final ciphertext out at the end!

Activity 1

Decrypt your answer to Exercise 4 to retrieve the original message.

You may be wondering why we bother to use a complicated encryption system like this when just adding a key is so effective for hiding secret messages. The problem is that, in order to keep everything secure, keys have to be distributed using a method called *public key cryptography (PKC).* This is effective but slow, so, in practice, we send only part of the key and then use this part to calculate the rest of it. This is where simply adding the key fails – if we can guess the beginning of the message (say, if messages always start with 'TOP SECRET') then we can find the beginning of the key, then calculate the rest from it. But if we use an encryption scheme like this, knowing the beginning of the message (or even all of it!) will not help us find the key.