PART 1

Goal: To decrypt “ciphertext1”

Available Data

- **Plaintext: ASCII file**
- **Key: Upper – lower case characters and numerals**
- **Map [16] [16]: The ciphertext’s 4 higher or lower bits (ch,cl)**
  - Rows: 4 higher or lower bits of the plaintext (ph, pl)
  - Columns: 4 higher or lower bits of the key (kh, kl)

PART 1

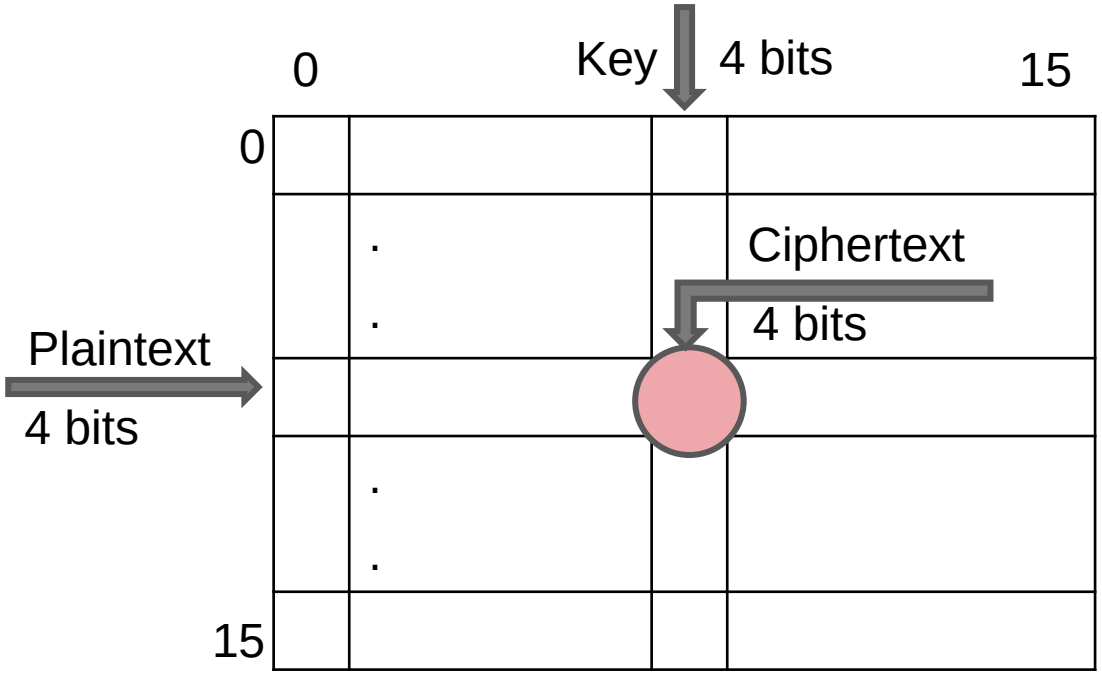
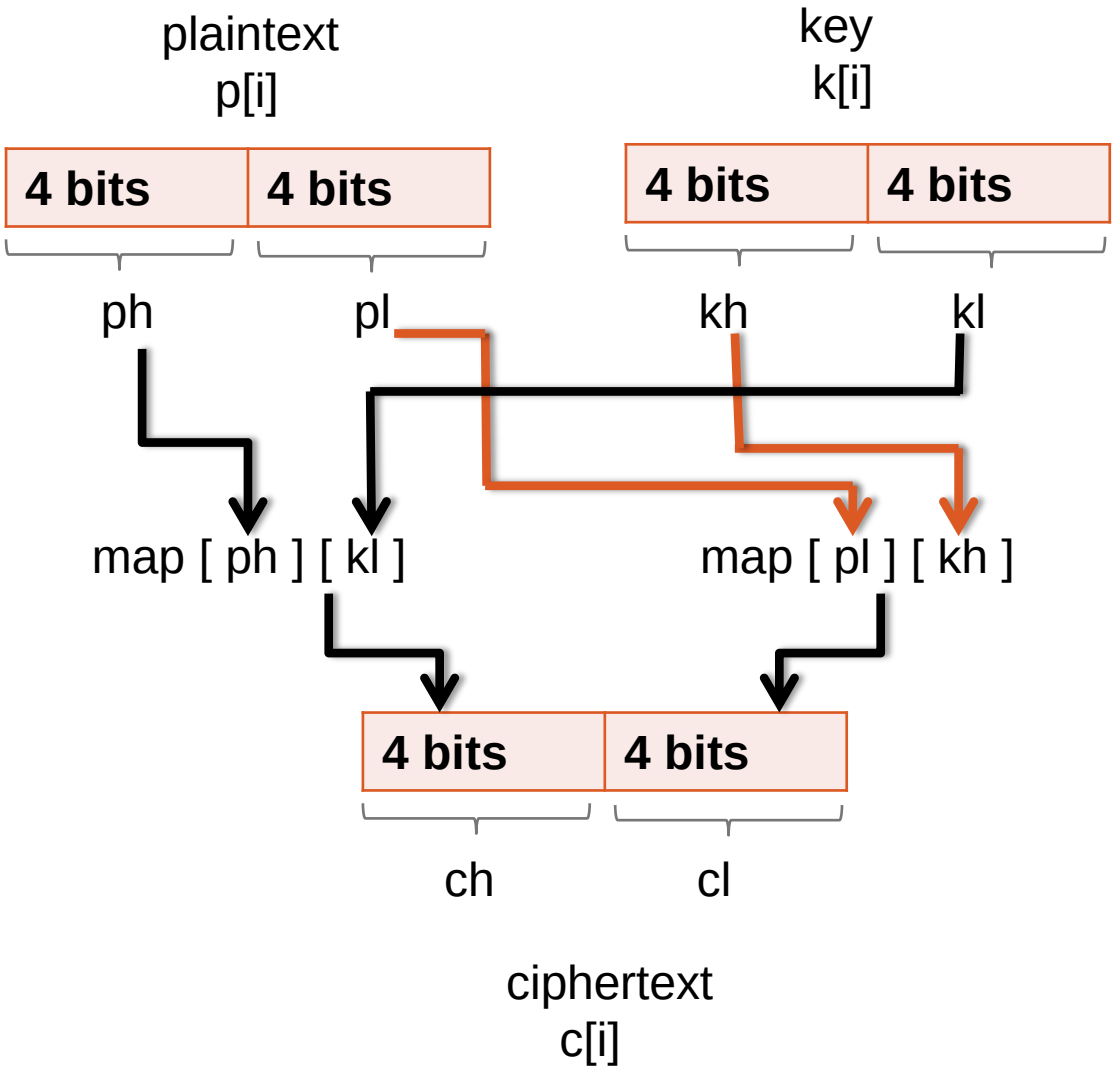


Figure 2: Encryption table

PART 1


plaintext
p[i]
key
k[i]
4 bits
4 bits
4 bits
4 bits
ph
pl
kh
kl
map [ph] [kl]
map [pl] [kh]
4 bits
4 bits
ch
cl
ciphertext
c[i]

PART 1

- **How is encryption performed?**

Example: Suppose that the plaintext to be encrypted is “hello!” and the keyword is “key”.

Step 1: Extract ph, pl from the byte p and kh, kl from the byte k

Step 2: Use these formulas to find the values of ch and cl in the map

ch <- map [ph] [kl]

cl <- map [pl] [kh]

Step 3: Combine ch and cl into the byte c

PART 1

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Figure 3: ASCII table

PART 1

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	{0x7,	0x5,	0x0,	0x4,	0x2,	0x3,	0xb,	0x6,	0xa,	0x8,	0x9,	0xd,	0xc,	0xf,	0xe,	0x1}
1	{0x3,	0x8,	0xd,	0xa,	0xc,	0xe,	0xf,	0xb,	0x7,	0x6,	0x4,	0x5,	0x1,	0x2,	0x0,	0x9}
2	{0x4,	0x0,	0x3,	0x1,	0xb,	0xa,	0x8,	0x5,	0x9,	0xd,	0xc,	0xe,	0xf,	0x6,	0x7,	0x2}
3	{0x9,	0xe,	0x7,	0xc,	0x6,	0x4,	0x5,	0xd,	0x1,	0x0,	0x2,	0x3,	0xb,	0x8,	0xa,	0xf}
4	{0x1,	0x3,	0xa,	0x2,	0x8,	0x9,	0xd,	0x0,	0xc,	0xe,	0xf,	0x7,	0x6,	0x5,	0x4,	0xb}
5	{0xe,	0x6,	0x5,	0x7,	0x1,	0x0,	0x2,	0xf,	0x3,	0xb,	0xa,	0x8,	0x9,	0xc,	0xd,	0x4}
6	{0x2,	0xa,	0x9,	0xb,	0xd,	0xc,	0xe,	0x3,	0xf,	0x7,	0x6,	0x4,	0x5,	0x0,	0x1,	0x8}
7	{0x6,	0x1,	0x2,	0x5,	0x3,	0xb,	0xa,	0x4,	0x8,	0x9,	0xd,	0xc,	0xe,	0x7,	0xf,	0x0}
8	{0xb,	0x9,	0xc,	0x8,	0xe,	0xf,	0x7,	0xa,	0x6,	0x4,	0x5,	0x1,	0x0,	0x3,	0x2,	0xd}
9	{0x0,	0xb,	0x8,	0x3,	0x9,	0xd,	0xc,	0x2,	0xe,	0xf,	0x7,	0x6,	0x4,	0x1,	0x5,	0xa}
A	{0x8,	0xc,	0xf,	0xd,	0x7,	0x6,	0x4,	0x9,	0x5,	0x1,	0x0,	0x2,	0x3,	0xa,	0xb,	0xe}
B	{0x5,	0x2,	0xb,	0x0,	0xa,	0x8,	0x9,	0x1,	0xd,	0xc,	0xe,	0xf,	0x7,	0x4,	0x6,	0x3}
C	{0xd,	0xf,	0x6,	0xe,	0x4,	0x5,	0x1,	0xc,	0x0,	0x2,	0x3,	0xb,	0xa,	0x9,	0x8,	0x7}
D	{0xc,	0x7,	0x4,	0xf,	0x5,	0x1,	0x0,	0xe,	0x2,	0x3,	0xb,	0xa,	0x8,	0xd,	0x9,	0x6}
E	{0xa,	0xd,	0xe,	0x9,	0xf,	0x7,	0x6,	0x8,	0x4,	0x5,	0x1,	0x0,	0x2,	0xb,	0x3,	0xc}
F	{0xf,	0x4,	0x1,	0x6,	0x0,	0x2,	0x3,	0x7,	0xb,	0xa,	0x8,	0x9,	0xd,	0xe,	0xc,	0x5}

Figure 4: The map table

PART 1

Example: Plaintext letter is ‘h’ and key letter is ‘k’

ASCII code for ‘h’ is 0x68 and for ‘k’ is 0x6B

So, ph = 0x06, pl = 0x08, kh = 0x06, kl = 0x0B

ch = map [ph] [kl] = map[0x06][0x0B] = 0x04

cl = map [pl] [kh] = map[0x08][0x06] = 0x07

So, the ciphertext byte is 0x47

POINTERS

- Part 1: The plaintext is printable ASCII, and the key is a combination of upper and lower case characters and numerals. Use these facts when searching for the key.
- Part 2: The plaintext is a common file format that may include non-printable ASCII characters
- Online sources
  - Google
    - http://www.simonsingh.net/The_Black_Chamber/vigenere_cipher.html
  - YouTube

Make sure you cite your references!

QUESTIONS ?