Introduction to Cryptography 1

Definitions and Introduction to Substitution Ciphers

Yicheng Wang

White Hat Academy

2015-03-06

What is Cryptography?

- Cryptography is the study of encodings.
- Cryptographic algorithms are the algorithms used to encode messages such as "Hello World" into "b10a8db164e0754105b7a99be72e3fe5."
- Some necessary definitions:
  - Plaintext (message): the original message.
  - Key: the encryption/decryption agent (sometimes non-existent).
  - Ciphertext: the result of a crypto algorithm.

- Any cryptographic algorithm has an inverse with respect to the plaintext, the original function is called the encryption algorithm and the inverse is called the decryption algorithm.
- Good cryptographic algorithms are bijective with respect to the plaintext message, which means that each ciphertext is unique to a message-key pair.
- In mathematical terms:

$$\exists Encryption : \{Plaintext\} \times \{Keys\} \rightarrow \{Ciphertext\}$$

$$\exists Decryption : \{Ciphertext\} \times \{Keys\} \rightarrow \{Plaintext\}$$

Why cryptography?

- Cryptography is very useful in our modern society, with the end of the age of privacy, everything one does on the internet is strictly monitored by everyone, thus the only way of protecting one's privacy is by encryption.
- Different algorithms offer different degrees of security, but some is still better than none.

Substitution Ciphers

- One of the earliest forms of encryption is the substitution cipher. A substitution cipher is a method of encoding that divide the plaintext into pieces (most commonly letters) and substitute each piece with its corresponding ciphertext.

There are a lot of substitution ciphers. Their advantage lies in how easy it is to make them, but that also means that they are easily cracked. For this reason, a lot of substitution ciphers are not designed for security reasons but rather as a means to transmit data. Morse code is an example of this.

Character	Morse Code	Character	Morse Code	Number	Morse Code
A	·−	N	−·	1	·−−−−
B	−···	O	−−−	2	··−−−
C	−·−·	P	·−−·	3	···−−
D	−··	Q	−−·−	4	····−
E	·	R	·−·	5	·····
F	··−·	S	···	6	−····
G	−−·	T	−	7	−−···
H	····	U	··−	8	−−−··
I	··	V	···−	9	−−−−·
J	·−−−	W	·−−	0	−−−−−
K	−·−	X	−··−		
L	·−··	Y	−·−−		
M	−−	Z	−−··		

Interpretation of Data

- In cryptography, the most common ways of interpreting data is actually as numbers!
- For computers, numbers are easily manipulatable and there does exist a basic connection between numbers and strings, this is known as ASCII (American Standard Code for Information Interchange) encoding, as shown in the next slide.

ASCII encoding

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

Source: www.LookupTables.com

One of the more "useful" ciphers out there is the rot-N cipher, or Cesser Shift. It takes the alphabet, shifts it N units forward and then overlays it with the original alphabet to create the substitution. rot-13 functions as follows:

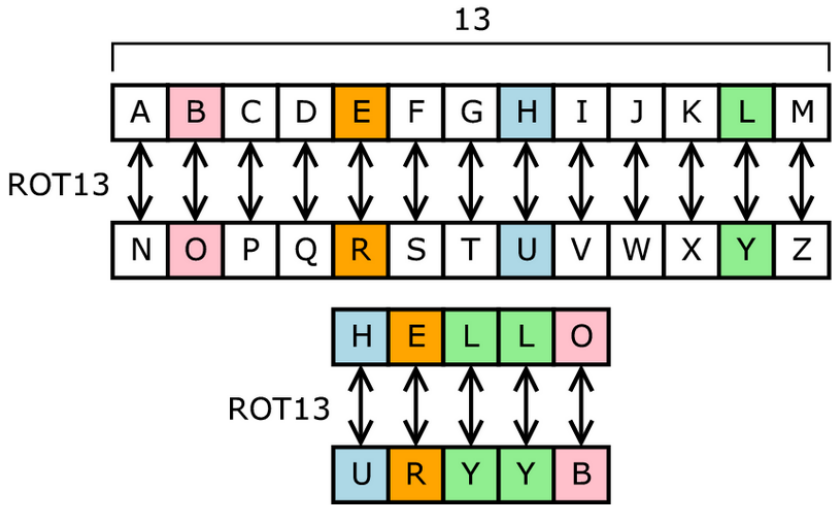

FIGURE : Credit goes to Matt Crypto of Wikipedia.

ROT-13

Following is a sample code for a rot-N algorithm.

```
1  def encrypt(data, N):
2      result = []
3      for c in list(data):
4          # Upper Case Letters
5          if ord(c) > 64 and ord(c) < 91:
6              result.append(chr(65 + (ord(c) - 65 + N) % 26))
7
8          # Lower Case Letters
9          if ord(c) > 96 and ord(c) < 123:
10             result.append(chr(97 + (ord(c) - 97 + N) % 26))
11
12         # We ignore everything else
13         else:
14             result.append(c)
15
16     return "".join(result)
```

Mono-alphabetic Substitution Ciphers

What we just discussed is called a **mono-alphabetic** substitution cipher because it uses the same encryption scheme for each letter. Note that this is not particularly safe and can be easily broken, as long as one has figured out the complete encryption table, one has cracked the entire algorithm. The following is another rather old algorithm, let's see what it does:

Challenge algorithm

You know that "the quick brown fox jumps over the lazy dog" encrypts to:

tsv jfrxp yildm ulc qfnkh levi gsv ozab wlt

Using that info, try to figure out what this means:
uozt: gsrh_rh_gsv_zgyzhs_xrksvi
As an additional challenge, try to code it!

Plain-text Attack

- What we just did was called a "plain-text attack" or a crib attack. It works because we knew a part of the text and then can use it to find the encryption algorithm.
- However, that raises the question of what if we don't know anything about the text?
- As an exercise: go to this url: http://tinyurl.com/encodedMessage
- Grab the file, it looks like gibberish, you know that it is an encryption based on monoalphabetic substitution... But what else do you know?

Frequency Analysis

You know that this made sense before encryption, and that is a huge huge help to you. Now you can use what is known as Frequency Analysis to crack the script.
The process is quite simple:

- go to www.tinyurl.com/ltrFreq
- you know the text used to be in English, you also know that each letter has its own UNIQUE CIPHERED PAIR... huh? What could this mean?
- Have fun cracking it!