Lab Exercise 5: Simple Encryption - Ciphers

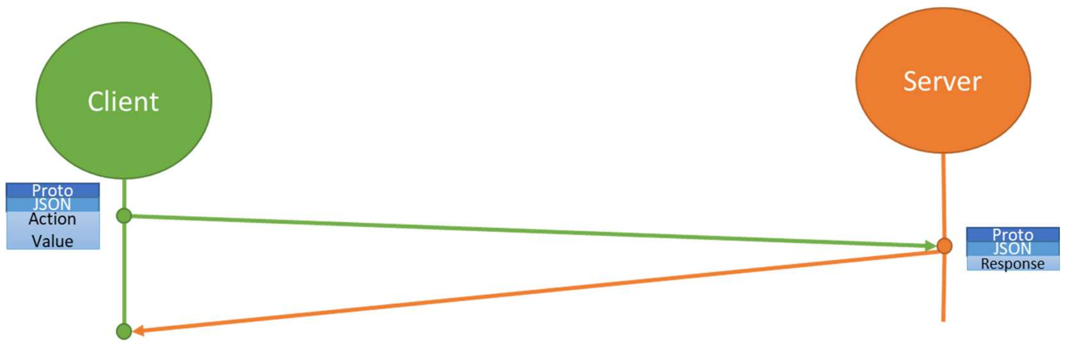
We are moving onto the security side of the course – you can continue to work on the networking aspects if you haven't yet completed the first 4 core exercises. In this lab you will be looking at the idea of simple encryption, while none of the things we look at this week are suitable for modern encryption schemes the present some of the core ideas in encryption.

We need to consider the following:

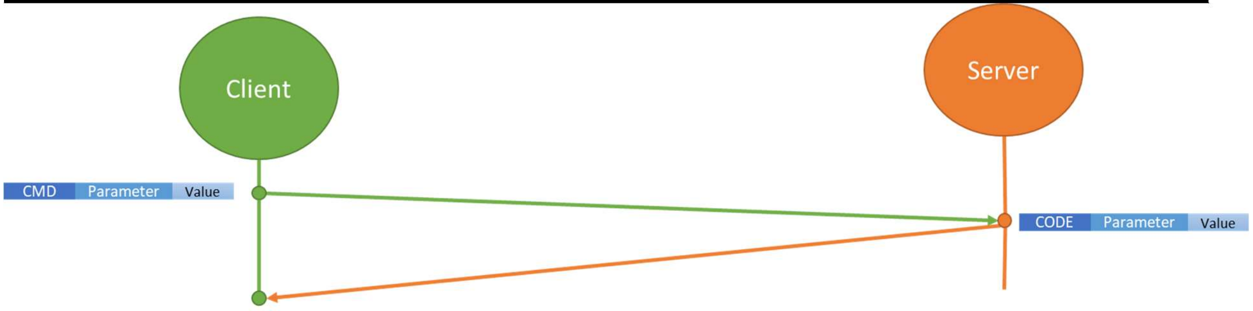
What we are protecting	Methodology
Existence of message	Steganography
Meta-data of message	Privacy enhancing techniques (e.g. TOR)
Content	Encryption
Nothing	None

The two key aspects we are looking at today are substitution (swapping characters), and transposition (moving characters around). Most modern encryption techniques will apply multiple rounds of interleaved substitutions, transpositions, and applying a key at each round. Importantly the substitutions in these systems don't just affect one character at a time – however we can look at that later. For the assessment an encryption that demonstrates both of these concepts will be the minimum acceptable standard.

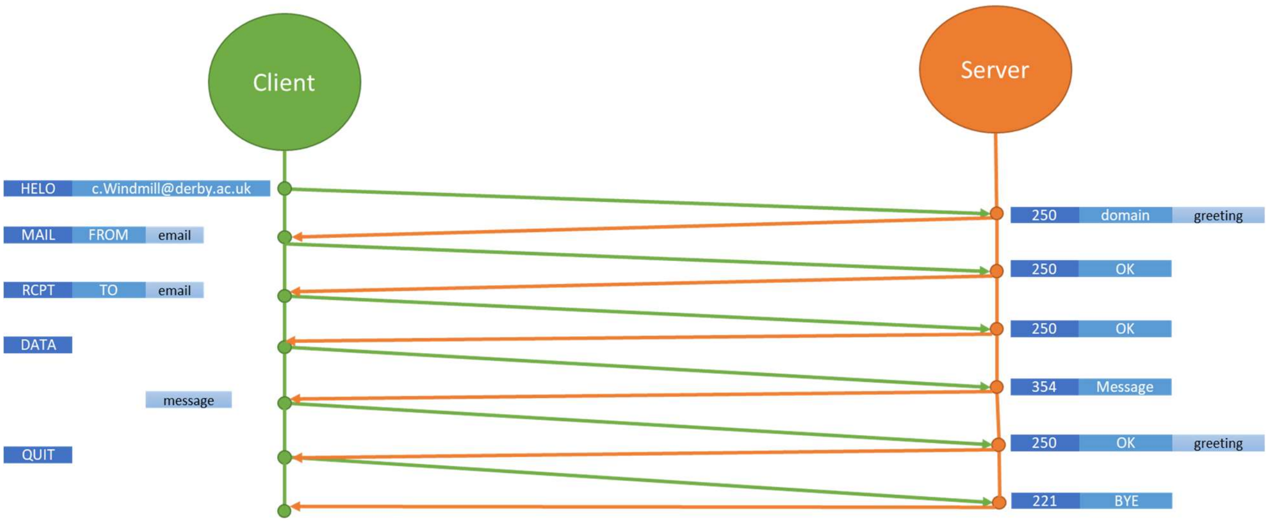
At current we have a system that looks something like the following if you have the command server working, if you are working from the SMTP code the packets will be much simpler:



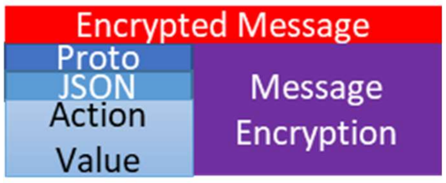
Our messages from the application server are encoded as a dictionary of action:value pairs inside the JSON header inside our simple proto header. This is more complex than we need for our SMTP server which sends and receives 4 letter commands + data, or 3 number response + data in each message as shown below.



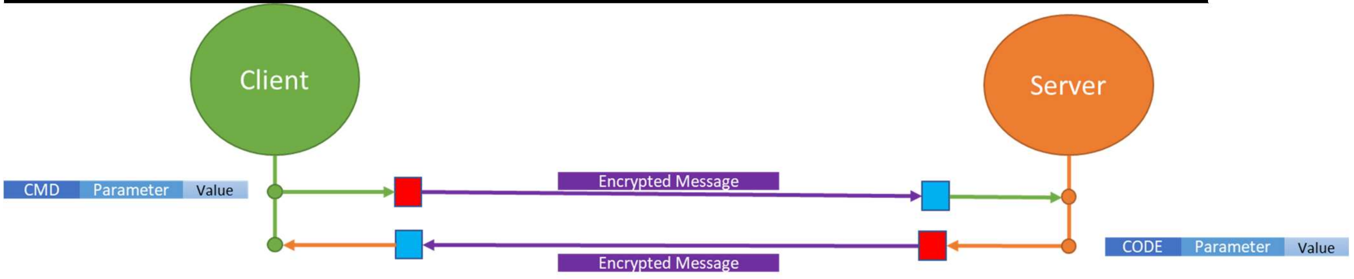
For SMTP (using RFC821) we would have a message exchange similar to the following:



This was perfectly ok (well not really but the protocol wasn't designed for security) in the 1980s, however we need to add some security – we are going to trust the server here to be nice otherwise we would want to encrypt the message separately so that the communication is encrypted, and the contents are similar to the following:

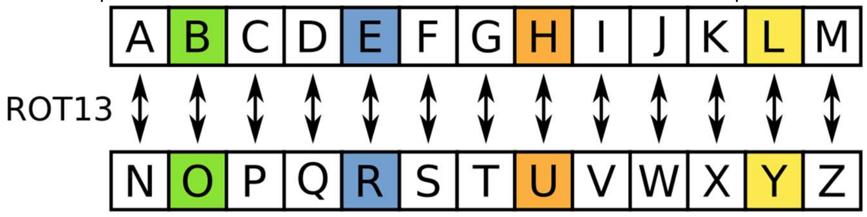


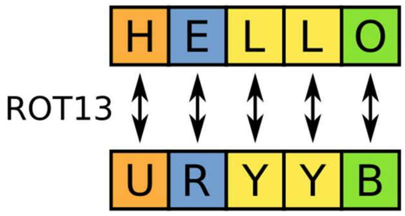
As we are trusting the server we will instead encrypt all messages onto the network, and decrypt them coming back out – the red and blue actions respectively. A brief format is provided for you in nws_encryption.py, though you can make this more efficient.



Core Exercise

- Implement a rot13 encoding system:
  - Each alpha character is moved 13 slots onwards in the alphabet





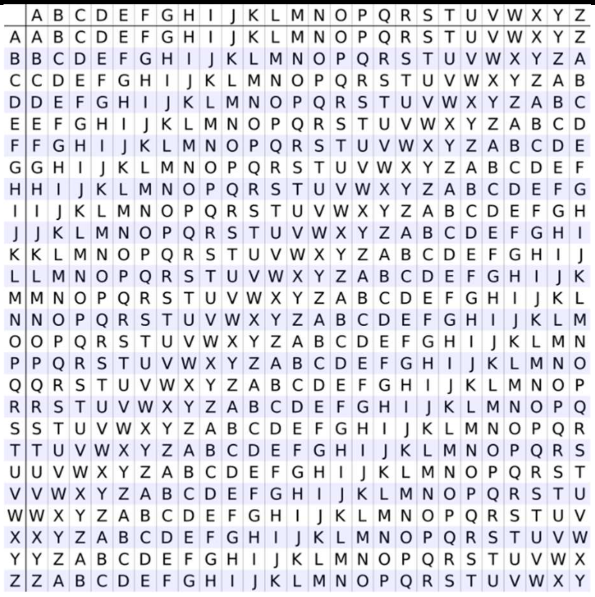
- Implement a Caesar cipher (an arbitrary rotation of x places around the alphabet)

The Caesar cipher is a simple substitution cipher – each letter of the input is shifted by N characters (for lower or upper case only) around the alphabet, this encryption is not very useful in real life as it is very vulnerable to statistical and frequency based attacks as English (like most languages) is structured. This is a generalisation of the ROT13 algorithm.

0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
5	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U

- Implement a Vigenere square cipher

The Vigenere square is equivalent to a applying a different Caesar cipher to the first N characters of a message, then repeating those through the length of the message. It is slightly more secure than a Caesar cipher where the message is significantly longer than the chosen key word, however, if the key is as long as the message it is effectively a one time pad.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Extension Exercise

- Research what a one time pad is, can you turn the Vigenere square you created into a one-time pad and what benefits would this have?
- Create a client-server program which allows for the negotiation of an encryption protocol between different clients e.g. the first sequence of messages should tell the client what encryption the server supports, and the client should pick one.
- Send and receive messages using the above three message formats then attempt to implement a solution of your own, what choices did you make?

Challenge Exercise

- Attempt to implement a Playfair Cipher – this cipher works on bi-grams (two characters) rather than single characters so is slightly more complex to implement.
- Attempt to combine a substitution cipher with a transposition one. Provide the function you create with either two integers or two strings – one as the substitution and one as the transposition.
- Research what is meant by a frequency based attack, bi-grams, tri-grams, and n-grams. What benefits does understanding these provide to attacking a Caesar cipher?