Checklist

	Item	**Your assignment details**	
1	Names and ID numbers of Group Members	Cai Gwatkin 15146508 Steve Clifton 11159915	
2	Operating System(s) used for testing your codes	Windows 10	
3	Compiler used	GCC 4.8.1	
4	IDE used	SublimeText 3	
5	Required Functionalities	Sends/receives keys encrypted by CA	Yes
		Sends/receives nOnce	Yes
		Encryption/decryption using CBC with RSA	Yes
6	Snap shots of sample interactions:	Yes	

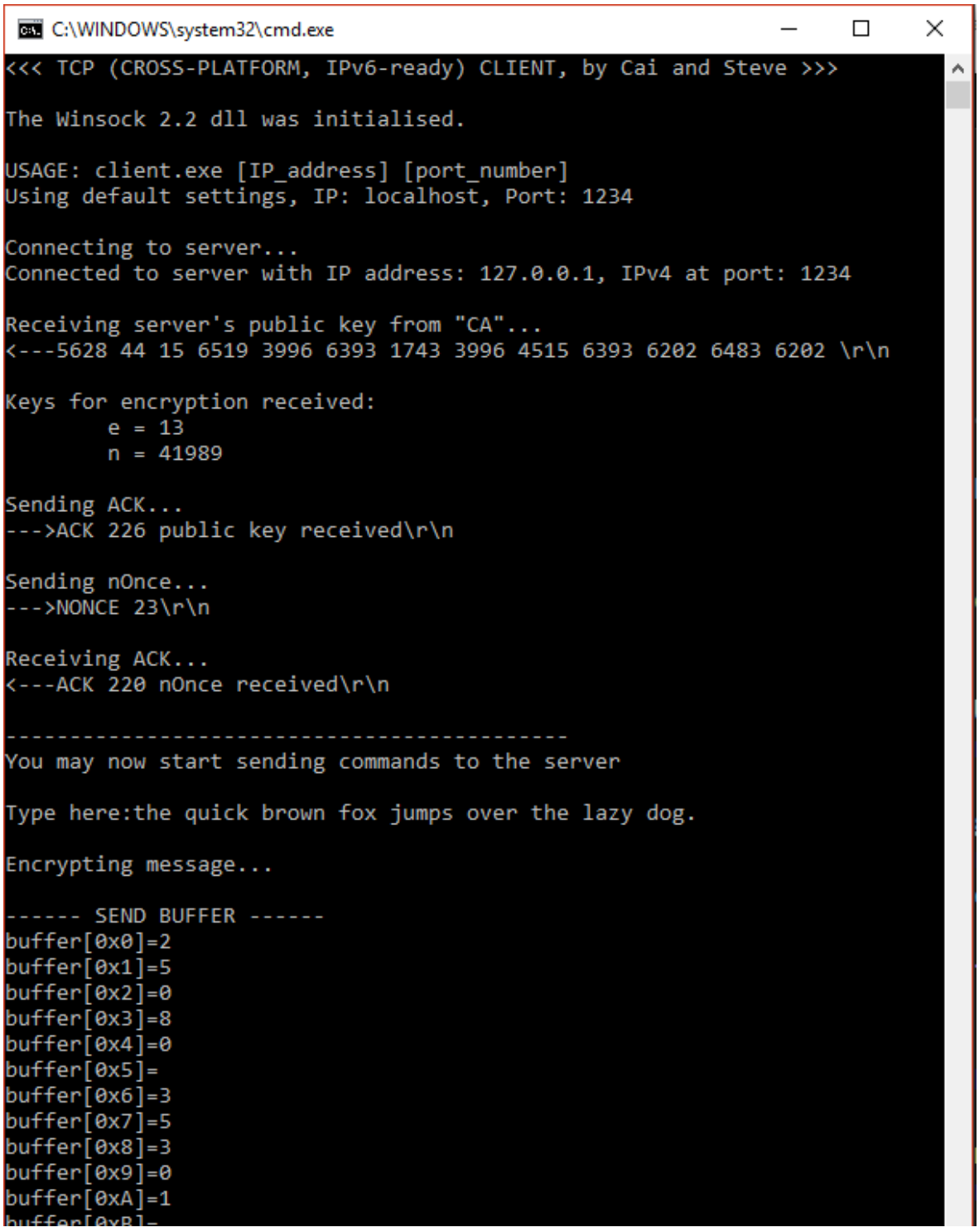
Pseudocode

Note: All keys, (eS, dS, nS) and (eCA, dCA, nCA), hardcoded in ***Server***. CA keys (eCA, nCA) and nOnce hardcoded in ***Client***.

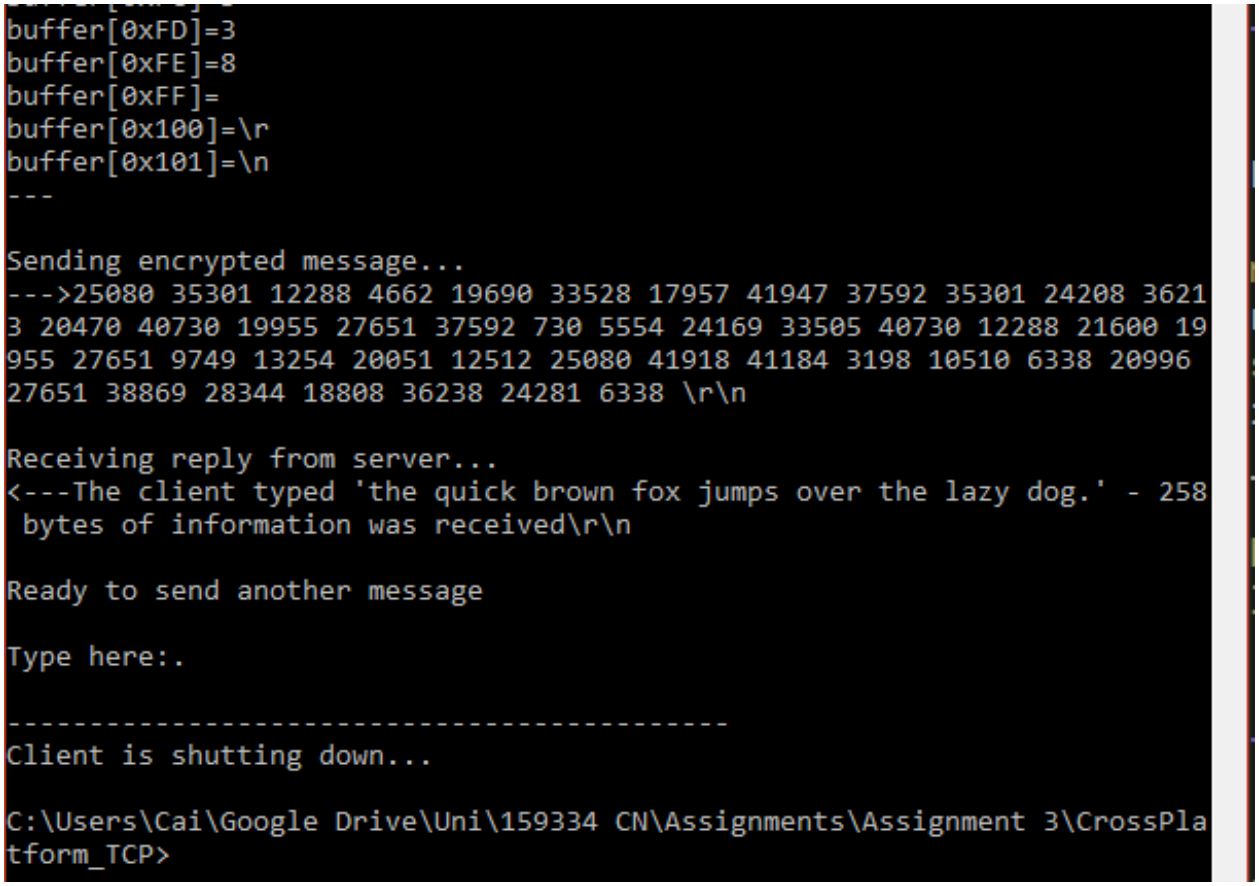
1. ***Server*** opens listening TCP port
    a. Errors in setup are handled
2. ***Client*** connects to the server using TCP/IP
    a. Errors in connection are handled
3. ***Server*** opens new communication socket for communicating with the client
4. ***Server*** encrypts the hardcoded server public keys using the hardcoded CA keys: dCA(e, n)
5. ***Server*** sends encrypted keys to client
6. ***Client*** receives message and decrypts using hardcoded eCA/nCA to get server public keys eCA(dCA(e, n))
7. ***Client*** sends ACK to server
8. ***Client*** sends the server the hardcoded nOnce in unencrypted form
9. ***Server*** receives and stores the nOnce
10. ***Server*** sends ACK to client
11. ***Client*** user enters a message to be sent, converted into array of long
12. ***Client*** encrypts the message: encryptedMessage = eS(cbc(message))
    a. CBC: the character is first XOR'd with either the nOnce (if it is the first letter in the string) or the previously encrypted value, result stored as array of long
    b. RSA: next the CBC encrypted character is RSA encrypted using Napoleon's 'repeatsquare' method, result stored as array of long
    c. The long values are converted into a sendable character string with space separation between encrypted character values
13. ***Client*** sends the encrypted message to the server
14. ***Server*** receives the message and converts the space separated values back into an array of long
15. ***Server*** decrypts message: decryptedMessage = cbc(dS(encryptedMessage))
    a. RSA: decrypts using Napoleon's 'repeatsquare' method, result stored as array of long
    b. CBC: the result is then XOR'd with either the nOnce (if it's the first letter in the string) or the previous RSA decrypted value, result stored as array of long
    c. The long values are converted back to characters and stored as the decrypted message
16. ***Server*** displays the string in the console and sends the unencrypted message back to the client for verification
17. Steps 8-13 are repeated until the client user enters '.'
18. ***Client*** closes connection with server
19. ***Server*** closes communication socket and listens for new connection, go to step 2

Snap shot interactions

Client 1

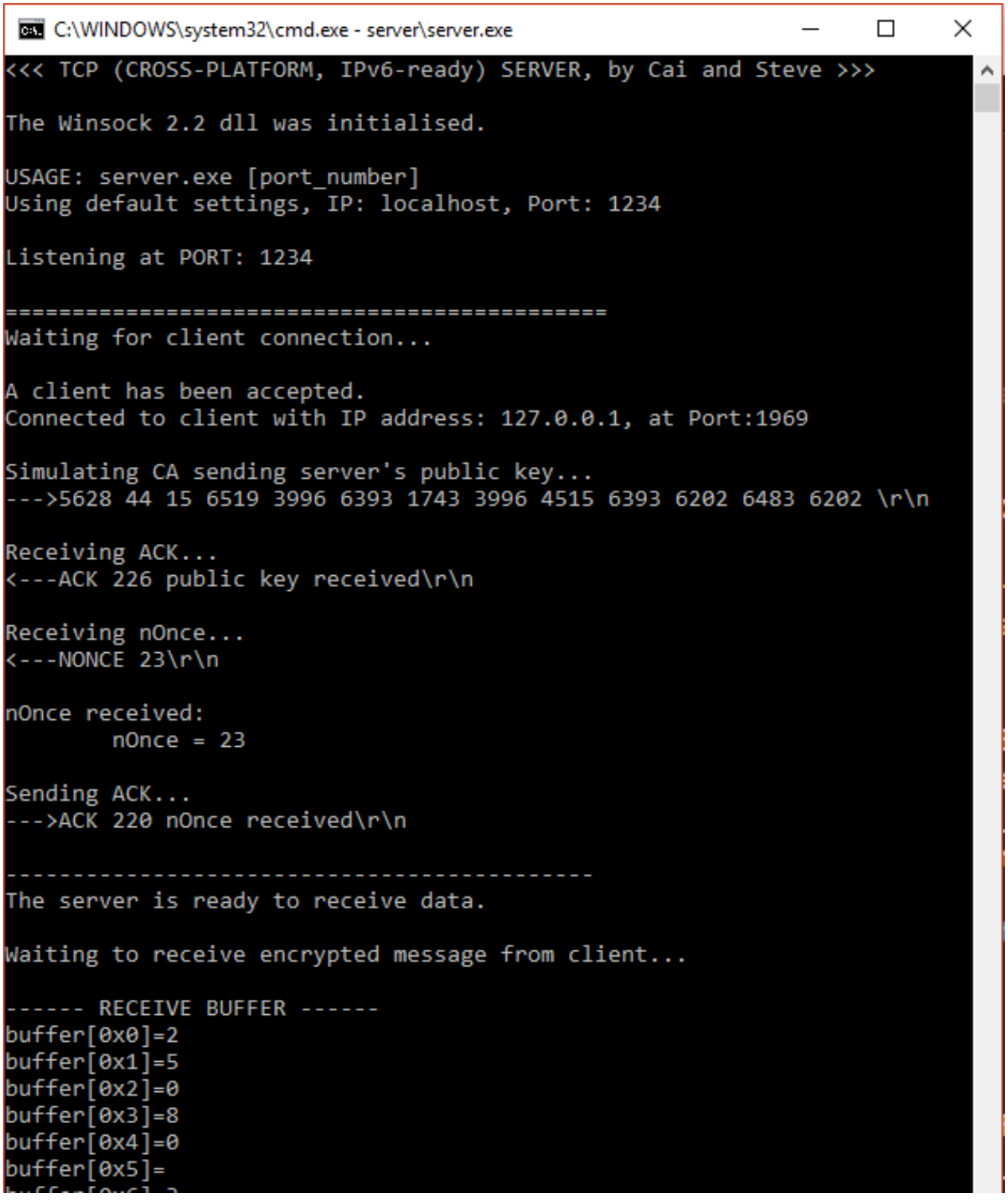


Client 2


buffer[0xFD]=3
buffer[0xFE]=8
buffer[0xFF]=
buffer[0x100]=\r
buffer[0x101]=\n

Sending encrypted message...
--->25080 35301 12288 4662 19690 33528 17957 41947 37592 35301 24208 3621
3 20470 40730 19955 27651 37592 730 5554 24169 33505 40730 12288 21600 19
955 27651 9749 13254 20051 12512 25080 41918 41184 3198 10510 6338 20996
27651 38869 28344 18808 36238 24281 6338 \r\n
Receiving reply from server...
<---The client typed 'the quick brown fox jumps over the lazy dog.' - 258
bytes of information was received\r\n
Ready to send another message
Type here:.
Client is shutting down...
C:\Users\Cai\Google Drive\Uni\159334 CN\Assignments\Assignment 3\CrossPla
tform_TCP>

Server 1


C:\WINDOWS\system32\cmd.exe - server\server.exe
<<< TCP (CROSS-PLATFORM, IPv6-ready) SERVER, by Cai and Steve >>>
The Winsock 2.2 dll was initialised.
USAGE: server.exe [port_number]
Using default settings, IP: localhost, Port: 1234
Listening at PORT: 1234
==========
Waiting for client connection...
A client has been accepted.
Connected to client with IP address: 127.0.0.1, at Port:1969
Simulating CA sending server's public key...
--->5628 44 15 6519 3996 6393 1743 3996 4515 6393 6202 6483 6202 \r\n
Receiving ACK...
<---ACK 226 public key received\r\n
Receiving nOnce...
<---NONCE 23\r\n
nOnce received:
nOnce = 23
Sending ACK...
--->ACK 220 nOnce received\r\n

The server is ready to receive data.
Waiting to receive encrypted message from client...
------ RECEIVE BUFFER ------
buffer[0x0]=2
buffer[0x1]=5
buffer[0x2]=0
buffer[0x3]=8
buffer[0x4]=0
buffer[0x5]=

Server 2

```
buffer[0xFB]=6
buffer[0xFC]=3
buffer[0xFD]=3
buffer[0xFE]=8
buffer[0xFF]=
buffer[0x100]=\r
buffer[0x101]=\n
---

Decrypting message...
Decrypted message:the quick brown fox jumps over the lazy dog.

Sending reply...
--->The client typed 'the quick brown fox jumps over the lazy dog.' - 258
 bytes of information was received\r\n

Waiting to receive encrypted message from client...
recv failed

Client has disconnected.

Disconnected from client with IP address: 127.0.0.1, Port: 1969

================================================
Waiting for client connection...
```