1. Generated Keys:

-----BEGIN PRIVATE KEY-----  
MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoGBAKyM5uDod5P+mGCr  
D4PuqU9NNaDT3aNhuaZmkIlTGDC6WgtEWbZ14Wqd8SlmTXjAe02KKKGN96+AGYmQ  
tAhc85yTqdsjx+ZzghI7Vj9MdfSq6P1FN22WbZFY0YUTBn1JwR8pvqmL7oZk9Rp0  
q0LDPFzQfCgrs/svpVs+D8qPMfWdAgMBAAECgYB+tvrWL+GgzVcZpoW+XYX4trqr  
YOtjEGFF0gxSzLL3FeMKYyd+1xKzOUnhHT/HbZMfXlZ4TcmXGHuoipuriJddLsLD  
iRAZpucx0gpkdRgl3r6XPcOruOw7TXRk+0P75DebKQ8HOPM6tnaITqNlempL/sXG  
o1Uiz28/S4fCLVYAAQJBAOjcXkeRAIeoYw6uU/pl9VJ/y3jid9c4pPKvYue1SdYO  
TYyv0ke0FtyEXLU6v9UswE3nyZVifsbWGIFNFG7TDq0CQQC9slObFramZiYE2KaO  
rSFvtNJ/l82YRVOU27zPLzSZjWAT+4UueHVwWUIlVa5Yh79a/uTWK6qDREnp/+pF  
AhCxAkBiwWKyk0pYPg+UoFY5MZhzHoOahuEwQ3OhAMS/Y+EmiPZWnhJeGoA/kVsH  
+Rx2Of5si8qwMg92IEJQQ/JQde4ZAkBILVnyYKoOlj9RCL4W+3kFb9/H/JKVpYsv  
0Iv+qQ9yfYBZGhKKo/rLTKsGWZuEw+HjVajsdKDkYU9eYalZqHeBAkEArN2L/mqA  
BhiHLQrHa1R82HTSkNUOuTzdS9gNn4S5TaLhe8jF5sDLrVPhm+/c5eAD0VABVZN4  
gILXlkQ/+TOH0A==  
-----END PRIVATE KEY-----

-----BEGIN PRIVATE KEY-----  
MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoGBAKGO3gfZ9tSz5ykH  
WMNf6fY+T1tClAMo6/OAL00GjU6BAWFBNXNEClYZ2Gv3Jx4BVwylEy3Z4dAhz0gO  
xIY1FTg9FlSgWybK0vq0RioOrMNM1/Py/f6vZCoe3+CbPXvuuuXRobS2uBlKjuPV  
r6bxd00KsB10zbRHuj0lwymjfjRrAgMBAAECgYBTvawUwcQRiJa/S5GU47hv7OhR  
U3dJFAMJml3qPQUtteyzLou4DYKWEpt9iwR5pxQp976zO4ayzXOV1CHFtuoiAZHi  
rJMkQ6PRkStrJsROUWGiSpStsCUzGNpYQoXmrNMwZCWOW2GuvBHzgTC7JE7coTo+  
dp8HVku31KCR1UI+oQJBAOnvFW/qtPe+2Fjx6Lq3eSGnyHeWxHAkEMcS0mkleQET  
A/Y4bzPYLG31H5wUCe9Q3/VAkgDuNOoT11ld8uKNgykCQQCwzBa7b4yOfBuRHTWZ  
t04TIie+Fw6P1ZcLSLa9nX8G16L7LwOUJaXjkulxz15Up6W9OiGnpkyT7owEys/E  
ZCFzAkA5Y+wwbrZC6VEOURq1Y/mVpiHqJNTAZjivfhCctxK+jrm0GBg/8dLEjxRY  
KBpf2cl304V+MPJOUtvMEt5ZcjoxAkEAlazClf5zFXG2eN0ssq1AGOyyjrjAaD7g  
N/pgUxkSIH3QiiycMADLd2W54tM3Ph4ML64WLfOZIygyPCBjUxWeRwJAAxphqr6W  
PVDGygH1O50+U1JTUewGc2DiiNVovihdxiaRYv6x/8fAnSvCw9BiBHgIvIqUhldN  
VRBvM55Q66E1aw==  
-----END PRIVATE KEY-----

-----BEGIN PUBLIC KEY-----  
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCsjObg6HeT/phgqw+D7qlPTTWg  
092jYbmmZpCJUxgwuloLRFm2deFqnfEpZk14wHtNiiihjfevgBmJkLQIXPOck6nb  
I8fmc4ISO1Y/THX0quj9RTdtlm2RWNGFEwZ9ScEfKb6pi+6GZPUadKtCwzxc0Hwo  
K7P7L6VbPg/KjzH1nQIDAQAB  
-----END PUBLIC KEY-----

-----BEGIN PUBLIC KEY-----  
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQChjt4H2fbUs+cpB1jDX+n2Pk9b  
QpQDKOvzgC9NBo1OgQFhQTVzRApWGdhr9yceAVcMpRMt2eHQIc9IDsSGNRU4PRZU  
oFsmytL6tEYqDqzDTNfz8v3+r2QqHt/gmz177rrl0aG0trgZSo7j1a+m8XdNCrAd  
dM20R7o9JcMpo340awIDAQAB  
-----END PUBLIC KEY-----

Encrypted text:

The plaintext is: Christopher Soto 3/24/2018

The encrypted string is: dDKfjCJ2kIwl/IdFE70rs0FfdC6DHWQdPBuzC2ciam2nDlc2hiMJxT/gdLZGeNDOtR0Rw1gY8TRqBdq+uzYrTx1O7hhAd5zurZ+DpQRt8BENCKNIFvXPIh+GU35ClT/nMgPXVdk7KPODkVr9hX3G+uMXgDuqC+UP5ahP/ruU3yc=

Q: If we encode a byte array of length *n* with Base64 encoding, what is the length of the string derived from the encoding?

A: The length derived is actually variable, following this equation: 4\*[n/3] bytes long, including padding characters. <https://en.wikipedia.org/wiki/Base64> (Wikipedia, 2018)

1. Generated certificate from website:

-----BEGIN INFORMATION-----

Date: 24-Mar-18

Name: Christopher Soto

Username: cdsoto2

-----END INFORMATION-----

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCsjObg6HeT/phgqw+D7qlPTTWg

092jYbmmZpCJUxgwuloLRFm2deFqnfEpZk14wHtNiiihjfevgBmJkLQIXPOck6nb

I8fmc4ISO1Y/THX0quj9RTdtlm2RWNGFEwZ9ScEfKb6pi+6GZPUadKtCwzxc0Hwo

K7P7L6VbPg/KjzH1nQIDAQAB

-----END PUBLIC KEY-----

-----BEGIN PUBLIC KEY-----

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQChjt4H2fbUs+cpB1jDX+n2Pk9b

QpQDKOvzgC9NBo1OgQFhQTVzRApWGdhr9yceAVcMpRMt2eHQIc9IDsSGNRU4PRZU

oFsmytL6tEYqDqzDTNfz8v3+r2QqHt/gmz177rrl0aG0trgZSo7j1a+m8XdNCrAd

dM20R7o9JcMpo340awIDAQAB

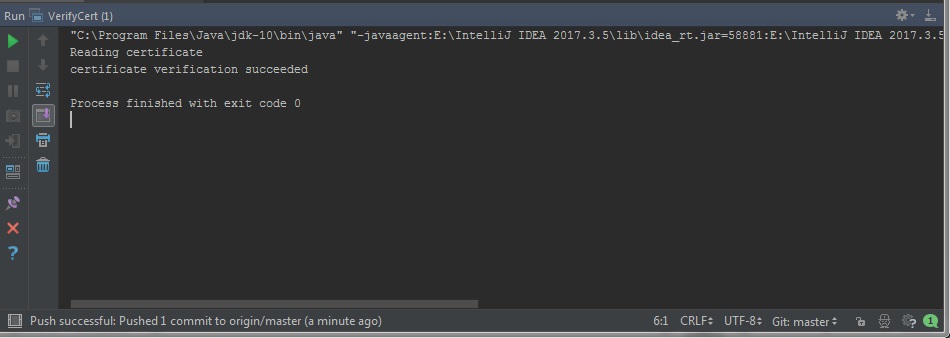
-----END PUBLIC KEY-----

-----BEGIN SIGNATURE-----

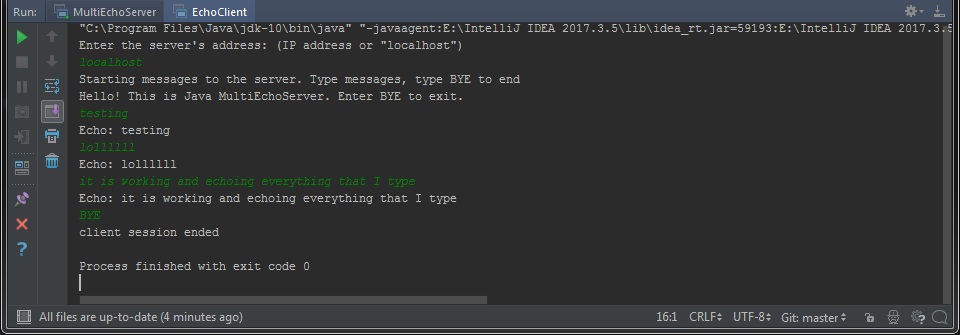
UUdd6eERrisLr9At8sT/L68MTMX36iL/jln9ZUPzEGl2Kvn3LhuLeF42fMrGOSq9cEXpOcvabKwtGIebW16xlcbW4+W/9wPKx3v4FivXAl4Labr8ch3s2OSRXgqWKXo0yTPEY/GtNKcpG00dhHUKDgV9PxKw93+2xYhntcWfTBM=

-----END SIGNATURE-----

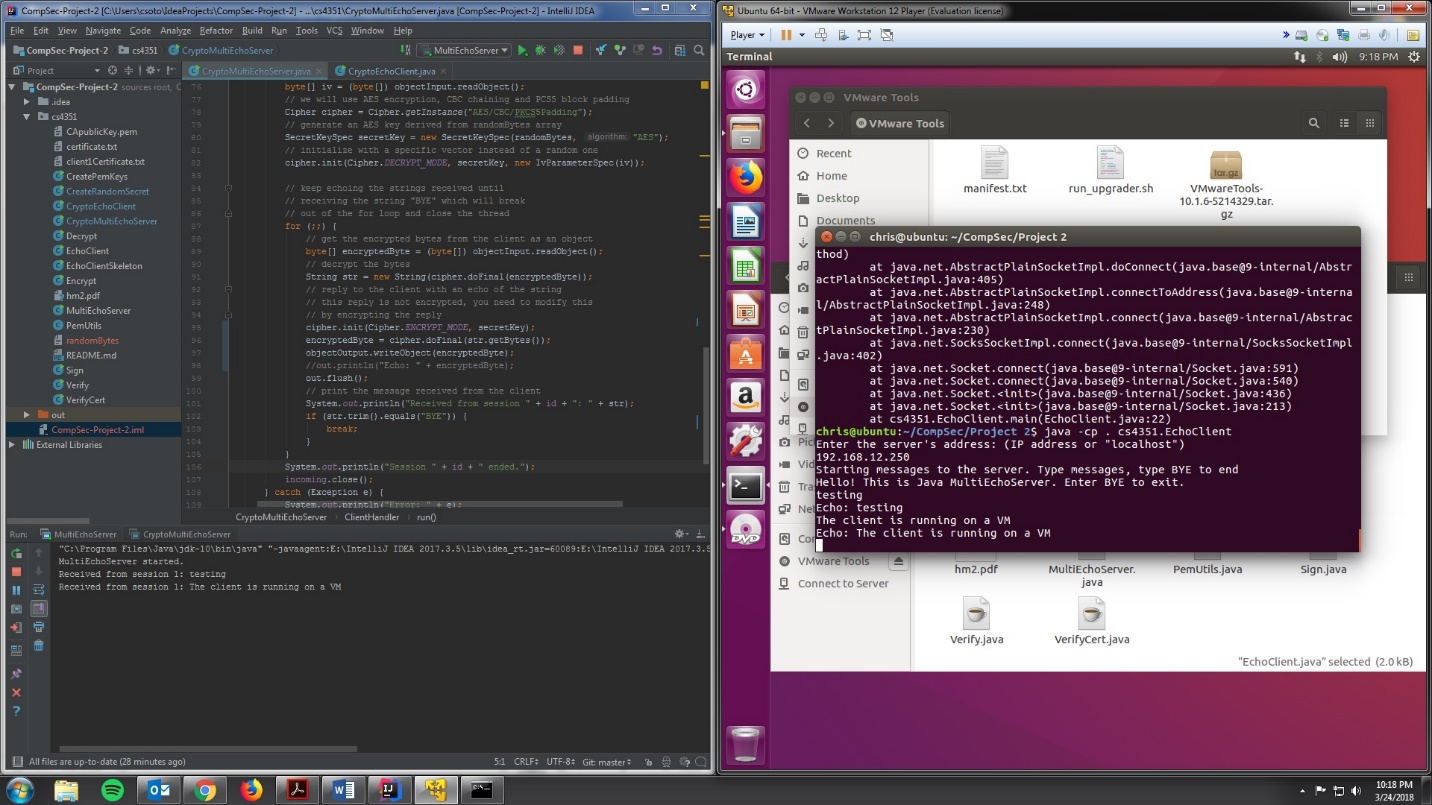
1. Certificate verification:



1. I was able to run both MultiEchoServer and EchoClient from my IDE. The client echoed everything I typed into the console. Here is a screenshot:



Here is a screenshot of running the client on a VM and the server on my work PC:



I did not have to make any changes to my VM settings. I just plugged in the IP of my work PC and it connected.

1. Here is the code I added to get the encryption working both ways:

On the server side

*/\*\* ADDED CODE FOR ENCRYPTING RESPONSE \*\*/*cipher.init(Cipher.*ENCRYPT\_MODE*, secretKey, new IvParameterSpec(iv));  
encryptedByte = cipher.doFinal(str.getBytes());  
objectOutput.writeObject(encryptedByte);  
objectOutput.flush();  
iv = (byte[]) objectInput.readObject();  
cipher.init(Cipher.*DECRYPT\_MODE*, secretKey, new IvParameterSpec(iv));  
*/\*\* END OF NEW CODE \*\*/*

On the client side

*/\*\* ADDED CODE FOR DECRYPTING RESPONSE \*\*/*cipher.init(Cipher.*DECRYPT\_MODE*, secretKey, new IvParameterSpec(iv));  
decryptedByte = (byte[]) objectInput.readObject();  
String str = new String(cipher.doFinal(decryptedByte));  
System.*out*.println("Echo:" + str);  
cipher.init(Cipher.*ENCRYPT\_MODE*, secretKey);  
iv = cipher.getIV();  
objectOutput.writeObject(iv);  
objectOutput.flush();  
*/\*\* END OF NEW CODE \*\*/*

I’ll include the whole files as part of my email.

1. Did not finish 6, just have a rough outline of the client and server code (not completely working yet).
2. Made more progress on 7 than 6, but it is still not working. I am able to connect and communicate with the server, but I am having issues getting the random bytes from the server decrypted properly. I have the next steps for encrypting and sending the client random bytes coded, and I believe that if I can get the decoding working I will be able to get the flag.