COMP 755 Assignment 3

Bassey Ude

October 2022

1 Assignment Description

Implement following encryption techniques and provide corresponding cipher text for the following plain text message:
Plaintext: “This assignment is the third out of five assignments in the course COMP 755.”
a. DES
b. 3DES
c. MD5 hash
d. RSA
Use toolboxes provided by any programming language and use a key that can be generated by those toolboxes.

2 DES

2.1 DES Code

```python
from Crypto.Cipher import DES
from secrets import token_bytes
key = token_bytes(8)
def encrypt(msg):
cipher = DES.new(key, DES.MODE_EAX)
nonce = cipher.nonce
ciphertext, tag = cipher.encrypt_and_digest(msg.encode('ascii'))
return nonce, ciphertext, tag
def decrypt(nonce, ciphertext, tag):
cipher = DES.new(key, DES.MODE_EAX, nonce=nonce)
plaintext = cipher.decrypt(ciphertext)
try:
cipher.verify(tag)
return plaintext.decode('ascii')
except:
return False
nonce, ciphertext, tag = encrypt(input('Enter a message: '))
plaintext = decrypt(nonce, ciphertext, tag)
print(f'Cipher text: {ciphertext}')
if not plaintext:
print('Message is corrupted!')
else:
print(f'Plain text: {plaintext}')

```

2.2 DES Output

```
Enter a message: This assignment is the third out of five assignments in
the course COMP 755

Cipher text:

b'\x8fX\xb2\xc1\xb0+\x90\xf2\xd4\xf1N\xa4\x12MK1\xd2\xd9\xa5\xc9\x062V\x8d\
xd3\xffa\xb5*\x17\xaf4/\x95\xc9\x8bi\xa2>\xf6\x9a4\xa5\xcac\xc7\xa0\xa4\xd9
K\xe5\xfcN\x1f\x9c-c\x90\x88eRyW\xaaE\xfe\x8a|\xd0\xa4\xf0OIV\xd5\xa2'
Plain text: This assignment is the third out of five assignments in the
course COMP 755
```

3 3DES

3.1 3DES Code

```python
from Crypto.Cipher import DES3
from Crypto.Random import get_random_bytes
while True:
try:
key = DES3.adjust_key_parity(get_random_bytes(24))
break
except ValueError:
pass
def encrypt(msg):
cipher = DES3.new(key, DES3.MODE_EAX)
nonce = cipher.nonce
ciphertext = cipher.encrypt(msg.encode('ascii'))
return nonce, ciphertext
def decrypt(nonce, ciphertext):
cipher = DES3.new(key, DES3.MODE_EAX, nonce=nonce)
plaintext = cipher.decrypt(ciphertext)
return plaintext.decode('ascii')
nonce, ciphertext = encrypt(input('Enter a message: '))
plaintext = decrypt(nonce, ciphertext)
print(f'Cipher text: {ciphertext}')
print(f'Plain text: {plaintext}')

```

3.2 3DES Output

```
Enter a message: This assignment is the third out of five assignments in the course
COMP 755

Cipher text:
b"4\x06\xda\xebV5Q\xab2|\x93\xef\xdes'\x07Z\nPh\xb3\xca~?SK\xb8\xd7~\xe1\x88`7\xe1\
xbb\xf9I\xb6\xbe]\x96\xc0\xc7C\x88,\xfe\x06G\xdf\x1d\xe5\xc2\xe7\xde\r\xd6\xf4^\xa3
c\xf1\x8aH\xd5\x0f\xff\xba9p\x88\x97\xa1R\x9f"
Plain text: This assignment is the third out of five assignments in the course COMP
755
```

4 MD5

4.1 MD5 Code

```
from hashlib import md5
class MD5:
def __init__(self, data = "This assignment is the third out of five assignments
in the course COMP 755."):
self.data = data
def encrypt(self):
self.data = md5(self.data.encode()).hexdigest()
return "Crypted: "+self.data
def decrypt(self, data):
if md5(data.encode()).hexdigest() == self.data:
return "Decrypted: "+data
del self.data
else:
return "Error"
crypt = MD5()
print(crypt.encrypt()) # Encrypt
print(crypt.decrypt("This assignment is the third out of five assignments in the
course COMP 755."))
```

4.2 MD5 Output

```
Crypted: 3a8a7eebfbe46f826c2b35a55224dfe5
Decrypted: This assignment is the third out of five assignments in the course COMP
755.
```

5 RSA

5.1 RSA Code

```python
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
import binascii
keyPair = RSA.generate(3072)
pubKey = keyPair.publickey()
print(f"Public key: (n={hex(pubKey.n)}, e={hex(pubKey.e)})")
pubKeyPEM = pubKey.exportKey()
print(pubKeyPEM.decode('ascii'))
print(f"Private key: (n={hex(pubKey.n)}, d={hex(keyPair.d)})")
privKeyPEM = keyPair.exportKey()
print(privKeyPEM.decode('ascii'))
msg = b'This assignment is the third out of five assignments in the course COMP
755.'
encryptor = PKCS1_OAEP.new(pubKey)
encrypted = encryptor.encrypt(msg)
print("\nEncrypted:", binascii.hexlify(encrypted))
decryptor = PKCS1_OAEP.new(keyPair)
decrypted = decryptor.decrypt(encrypted)
print('\nDecrypted:', decrypted)
```

5.2 RSA Output

```
Public key:
(n=0x8de6138e7651b862269fe9e1ebf640ad299ddb70ffea32c6ef5c9bb0f5bfc9f9dbecfa3fddea1b
0e815f0de22dc96d046bafe26b7558a27b2597b66a7b25a08ca42cbb767f5cd1a79e14addc1ca670180
32e915b1f5435258c2a67ba9e5d2c6339c303df96b7b1bc6c6b06b42ee284adeded0d5223dbc46db759
2649c0dd94068f675239fb3e0370b41d705607c07e8d2b23850994c59049cbfd3f16c23d65c1628e409
85f2884da9d84de4dfcbe2f60eb0cc4a29eab67cd4f8676f256ae7c4dc1bb71502f7807c894d5d8fdf7
021ffbc9288ce185050d1175f4f4348e42d6d0e1d8d48afd3eacaa44711737f99090cd22fc8a2100d26
5ca1d9aa8e3da643e83310ba1fcded17d01e645db58bf7f4976a66be4516ce63eedc578a2209e3f3973
49148aaff9ff14b50320ef807b680d9bbf6465215d99024638b8b3cecf33c00ba10b8410771b12529a0
5deca6f7fba16fcd2651246c8a7aeb53866be6efbc37fa36e4591598df86dd296a4b2e0ba772a2105f1
33b32f701fa19bb7b7d1da9a53, e=0x10001)
-----BEGIN PUBLIC KEY-----
MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEAjeYTjnZRuGImn+nh6/ZA
rSmd23D/6jLG71ybsPW/yfnb7Po/3eobDoFfDeItyW0Ea6/ia3VYonsll7ZqeyWg
jKQsu3Z/XNGnnhSt3BymcBgDLpFbH1Q1JYwqZ7qeXSxjOcMD35a3sbxsawa0LuKE
re3tDVIj28Rtt1kmScDdlAaPZ1I5+z4DcLQdcFYHwH6NKyOFCZTFkEnL/T8Wwj1l
wWKOQJhfKITanYTeTfy+L2DrDMSinqtnzU+GdvJWrnxNwbtxUC94B8iU1dj99wIf
+8kojOGFBQ0RdfT0NI5C1tDh2NSK/T6sqkRxFzf5kJDNIvyKIQDSZcodmqjj2mQ+
gzELofze0X0B5kXbWL9/SXama+RRbOY+7cV4oiCePzlzSRSKr/n/FLUDIO+Ae2gN
m79kZSFdmQJGOLizzs8zwAuhC4QQdxsSUpoF3spvf7oW/NJlEkbIp661OGa+bvvD
f6NuRZFZjfht0paksuC6dyohBfEzsy9wH6Gbt7fR2ppTAgMBAAE=
-----END PUBLIC KEY-----
```

5.3 RSA OUTPUT CONT'D

```
Private key:
(n=0x8de6138e7651b862269fe9e1ebf640ad299ddb70ffea32c6ef5c9bb0f5bfc9f9dbecfa3fddea1b
0e815f0de22dc96d046bafe26b7558a27b2597b66a7b25a08ca42cbb767f5cd1a79e14addc1ca670180
32e915b1f5435258c2a67ba9e5d2c6339c303df96b7b1bc6c6b06b42ee284adeded0d5223dbc46db759
2649c0dd94068f675239fb3e0370b41d705607c07e8d2b23850994c59049cbfd3f16c23d65c1628e409
85f2884da9d84de4dfcbe2f60eb0cc4a29eab67cd4f8676f256ae7c4dc1bb71502f7807c894d5d8fdf7
021ffbc9288ce185050d1175f4f4348e42d6d0e1d8d48afd3eacaa44711737f99090cd22fc8a2100d26
5ca1d9aa8e3da643e83310ba1fcded17d01e645db58bf7f4976a66be4516ce63eedc578a2209e3f3973
49148aaff9ff14b50320ef807b680d9bbf6465215d99024638b8b3cecf33c00ba10b8410771b12529a0
5deca6f7fba16fcd2651246c8a7aeb53866be6efbc37fa36e4591598df86dd296a4b2e0ba772a2105f1
33b32f701fa19bb7b7d1da9a53,
d=0x77fc2c5d323f37a206f701553fa9225749f16934cbb0c4eee247747ed5ce2e83502bba0f771e157
2d68201c356e41382c16c2010ad511d52f41eace8ace8cc31204192e2481fbc47f7ab94b7a2a239f693
fa347ed0a6eba9ac33cf3245e69339f405aaa8cd50d6c1750baac7813d98e3596430f75bcae0f5c52b6
4ecc860946257b00e1f690aaeee05df3eb600280cda61b7cf2ea2d5cb841b3130bf95cf44d398faa034
3782939427cdca0c37c2b95d4263b181b0994f54940227621359c750d5f02e4c42dfbb0915e7b8e9ded
e5b112190aedc903557d2041ab41b0550c63e2c88dc024ae81afd15e7b335fb85d1fad59a339cd102c3
796c42ccbeca6279b4131f31cafc233a18137f9bca1ccb9e2692acb4dd36c03a184fb2991a10a63f12d
fa3f80b2f479bcf3d179a40b15f11f3a676fa1d82725d40ed886faf109f102e27ef3b04012fa4165cdc
2a77b77ebe97ff5633e7d8fb9fe82095f5fee464a44c121a665bd1e9ceaf934990d5a4b0f902eaf6e96
5e4ecf957343c3cda6c48b31)
-----BEGIN RSA PRIVATE KEY-----
MIIG4gIBAAKCAYEAjeYTjnZRuGImn+nh6/ZArSmd23D/6jLG71ybsPW/yfnb7Po/
3eobDoFfDeItyW0Ea6/ia3VYonsll7ZqeyWgjKQsu3Z/XNGnnhSt3BymcBgDLpFb
H1Q1JYwqZ7qeXSxjOcMD35a3sbxsawa0LuKEre3tDVIj28Rtt1kmScDdlAaPZ1I5
+z4DcLQdcFYHwH6NKyOFCZTFkEnL/T8Wwj1lwWKOQJhfKITanYTeTfy+L2DrDMSi
nqtnzU+GdvJWrnxNwbtxUC94B8iU1dj99wIf+8kojOGFBQ0RdfT0NI5C1tDh2NSK
/T6sqkRxFzf5kJDNIvyKIQDSZcodmqjj2mQ+gzELofze0X0B5kXbWL9/SXama+RR
bOY+7cV4oiCePzlzSRSKr/n/FLUDIO+Ae2gNm79kZSFdmQJGOLizzs8zwAuhC4QQ
dxsSUpoF3spvf7oW/NJlEkbIp661OGa+bvvDf6NuRZFZjfht0paksuC6dyohBfEz
sy9wH6Gbt7fR2ppTAgMBAAECggGAB3/CxdMj83ogb3AVU/qSJXSfFpNMuwxO7iR3
R+1c4ug1Arug93HhVy1oIBw1bkE4LBbCAQrVEdUvQerOis6MwxIEGS4kgfvEf3q5
S3oqI59pP6NH7QpuuprDPPMkXmkzn0BaqozVDWwXULqseBPZjjWWQw91vK4PXFK2
TsyGCUYlewDh9pCq7uBd8+tgAoDNpht88uotXLhBsxML+Vz0TTmPqgNDeCk5Qnzc
oMN8K5XUJjsYGwmU9UlAInYhNZx1DV8C5MQt+7CRXnuOne3lsRIZCu3JA1V9IEGr
QbBVDGPiyI3AJK6Br9FeezNfuF0frVmjOc0QLDeWxCzL7KYnm0Ex8xyvwjOhgTf5
vKHMueJpKstN02wDoYT7KZGhCmPxLfo/gLL0ebzz0XmkCxXxHzpnb6HYJyXUDtiG
+vEJ8QLifvOwQBL6QWXNwqd7d+vpf/VjPn2Puf6CCV9f7kZKRMEhpmW9Hpzq+TSZ
DVpLD5Aur26WXk7PlXNDw82mxIsxAoHBALYnYj6j15vqBB2XiUFprL+PL1IPtVvQ
LG4/e1BBcZaCGlZSEBt4GY0K20xllxOlkdJx06B/SSk3g1QR+wtUpFsIiKrjjkKc
yN49bex5ERyv7r0PGrCv78TMxuexbyvbPL+0vzpnp0FT5nwtpq+c5c/z3YsAS4Xk
h9ec9OroEDwelGPz/weN0u/o0xagmmvpB9FXNcBsTTd2v6WOYGXLg+nrgG4WGrkj
oxoqj28UZoQvjEar+jWqaH4F2I40pdwu0QKBwQDHbN1zsoYUrzghZ0Fgl/9qN2DR
W4RCpUOCt2Sc7grjLm3k2WzE+aii0dy1EKA5q0Gi28Dtp7blAKlMM8ipED+ei7BJ
30PabH7kE0vIxn8M8j+0tgziJEU3VuI/t1DHYtPUspRcS9p9pRJDmcnG6pcia4Ay
eOVbhBlRNeyIxBNJnvWIGU+T5oLN018rjzIkTucxLky1nUP/vdoDPMsojGpgp+WZ
JLptDbxos3byBiv48qdm4u8kEiOWm0QZerQXZ+MCgcATbZPu9m6ytHOH4vfCwFCq
hc0YRZCkYEm84Ix4fOJumSXR+yK4q1CeWIAXX+aDM++fsIBP6AOXn9IkD17lUHvP
MdPR1j+AH1TVT1wAEtGea61dANVp7vn11ZmJcMVYuiN6lv6mbhPKxfYZXI0gwe1J
BsukXb2wZDWQ6667t1Cz+Nb+6jFTTu8mo4CQM0UFKW+qBlE3WtXJiqLaN5tUgCZr
trCWDUmAYaqtychOorBkfOXLS+B+BNTsTBdS446lM2ECgcBluPUYr+ZOpLXX7Ahg
EggZkNfU0n6bVBskkRFELvPkv+kUQ4eEzegj7TJMNYfVArL/NGKrltHm67hvgzaG
biBVaeFpPPYqcw8inWkZp45k9yhhsD7QFzorKSlM4N/WZGWy58hfb92qG0o+qzJO
QZkH2JfSpvIvBSm4z+2wMXu5INkTK/34bisoe/neiTNFa+3nmztLpEVsLNYIsrja
HQ2h8eJehGqHd+sz6N7yQw2o9XVIONdMqef3SYmqCXnVOosCgcBCyWHJbBe2id5T
MvRejSKQWvJPFPsFLqaaadnQFgk0hKGpSr+pMtbHcdRaBBFteIjOCB2yKvOh0oHX
mRINx0xpO8diXr9vQ1ePQF7QRgfRK+OAEsAIVW9VIhtUpJmdO92JYRjGIW1WfWM7
ZRD/pHGZZ2AgWTsyd6AC9oEH9XSdHAV7a5annAECmryXSFz1hF6otvg+pgR2hrtx
oSo4uDu0YFuW8zc6QOAbCqyUkHNUr+DLRnITkzXPCC3ndJ1kbwQ=
-----END RSA PRIVATE KEY-----
```

```
Encrypted:
b'77b2363f375913504e3375c6c9cb7098dd80f98d806d2c7055c25baa55d1c87962458bb4b9cd445dd
578676ffc97f4d05c3df1949ed789964e0e4077e22b368e9ada94fbce72a3c8d2859d33daa93e241090
57771b85fc1149b9e451c0ac5dea5f792275f9ac50e57d6498794b2cae1fd11d4222d334430bf6e5a34
d9d650ca72bc815b71551653aa87f2f2d57771e5adeef4c93f22197eb054f94c7ac0ac1a687a12107b9
13f12912a8bd25cb7f05a6e2a421d4b047d0765ecfd6eb4771d0ca8eed0f93ee9e4fa5cb4ef5e4980cd
47cd47c45565fc1d44b1722322593afb0046322b3be4059afa783c24d13d1b1d81c14d91861a3ab0c6a
03f10ce42ad968e680ef2926e43855c6e52518e9835ab8104dfeb12ee350116a672ecf2a62098ed7f47
cab6ac7883e1b501e3faceda4bf31a3ab5fac9dd1493575b2cc73d263a121fab6f1188031d0c0485db8
87c17ff9a92da7d3d3cd7b96415f6b088c6ebe4895847e7dfb7d656ef1e24d9d676b057cc3324c746f5
1b71a19e707140e500760e4'

Decrypted: b'This assignment is the third out of five assignments in the course
COMP 755.'
```

6 Conclusion

Used python toolbox to demonstrate DES, 3DES, MD5 and RSA with corresponding outputs.