Please include the following in your assignment writeup:
- Assignment number, your full name and Student ID.
- Clearly labelled answers to the questions (Core 1, Challenge 99, etc).
- Report is submitted in electronic form as a PDF file.

ASSIGNMENT NUMBER: 1
FULL NAME: Eli Jones
STUDENT ID: 300573902

PART 1: Brute Forcing Keys
- Question 1.1:
    (2^40)/(10^6) = 1099511.6 second or 18325.2 mins or 763.6 hours or 2.9 years
- Question 1.2:
    (2^128)/(10^6) = 3.4028237e+32 seconds or 6.4697385e+26 years or roughly 647 Septillion years.
- Question 1.3:
    6.4697385e+26 = 6.4697385 *10^26 years or roughly 647 Septillion years. 3.2348693e+19 times older than the maximum estimated end of the universe.
- Question 1.4 [completion]:
    It takes roughly 647 Septillion (24 0s at end of number) years for AES 128bits to encrypt at 1 million bits a second. It takes roughly 6.27 Sexdillion (51 0s at end of number) years for AES 192bits to encrypt at 1 million bits a second. It takes roughly 157 Duovigintillion (71 0s at the end) years for AES 128bits to encrypt at 1 million bits a second. So by this logic, having more keys to encrypt to would mean that the time taken would be exponentially longer to brute force, rendering brute force attacks nearly obsolete in that they will never work.
- Question 1.5 [challenge]:
Even with Grover's algorithm halving the brute force halved (so 2^64 instead of 2^128), it would still take 18 trillion years to brute force at 1million bits a second. So yes it would still be safe to use against an attack as it would take another 900000 times longer than the max estimate end of the universe to decrypt.

PART 2: Classical Cryptography
- Question 2.1:
    Most common letter was Z.
- Question 2.2:
    Caesar cyphers work by shifting all letter of the alphabet over by x amount (i.e. e with x=2 would become g), based on this, you are able to compare the frequency of letters in the English alphabet to the frequencies of letters in the cyphertext in order to determine the value of the key or x.
- Question 2.3:
Decrypts with key of 19 to:

welcome to the fractured future, the first century following the singularity. earth has a population of roughly a billion hominids. for the most part, they are happy with their lot, living in a preserve at the bottom of a gravity well. those who are unhappy have emigrated, joining one or another of the swarming dense thinker clades that fog the inner solar system with a dust of molecular machinery so thick that it obscures the sun. except for the solitary lighthouse beam that perpetually tracks the earth in its orbit, the system from outside resembles a spherical fogbank radiating in the infrared spectrum; a matryoshka brain, nested dyson spheres built from the dismantled bones of moons and planets.

- Question 2.4:

Show your working by completing the table below (you will need more than one).

Key	M	U	D	M	U	D	M	U	D	M	U	D	M
Key	13	21	3	13	21	3	13	21	3	13	21	3	13
Plaintext	T	O	B	E	O	R	N	O	T	T	O	B	E
Plaintext	20	15	2	4	15	18	14	15	20	20	15	2	4
Ciphertext	33	36	5	17	36	21	27	26	23	33	36	5	17
Ciphertext	F	i	e	q	i	u	z	i	w	f	i	e	q

- Question 2.5 [completion]:

```
WIBSI MSQZM BXJTL FBWFE G
```

Rain is the first 4 characters.
R = 18, A = 1, I = 9, N = 14
W = 23, I = 9, B = 28, S = 19
Key = 6,9,20,6 = FITF

RAINA TNIGH TEELS ATDAW N
RAIN AT NIGHT EELS AT DAWN

- Question 2.6 [completion]:
Combining classical won't lead to better security as it involves no pseudo-randomness. Solving a combined cypher would be as simple as applying a Vigenère brute force then looking for normal suffixes (i.e. double letters), then using the Caesar cypher to find the plaintext. Meaning that no real additional security would be added.

- Question 2.7 [chalenge]:
    - Known plaintext attack: The attacker knows at least one sample of both the plaintext and the ciphertext. In most cases, this is recorded real communication. for example, If the XOR cipher is used, this will reveal the key as plaintext xor ciphertext.
    - Chosen plaintext attack: The attacker can specify his own plaintext and encrypt or sign it. They can carefully craft it to learn characteristics about the algorithm. For example, they can provide an empty text, a text which consists of one "a", two "aa". (E.g. if the Vigenère cipher is used, it is very easy to extract the key length and recover the key by repeating one letter).

PART 3: Symmetric Cryptography
- Question 3.1:

```
barretts% openssl enc -des-cbc -pbkdf2 -in ciphers.txt -out ciphers.des.enc
enter des-cbc encryption password:
Verifying - enter des-cbc encryption password:
barretts%
```

openssl enc -des-cbc -pbkdf2 -in ciphers.txt -out ciphers.des.enc uses to encrypt file.
Password is 300573902

- Question 3.2:
openssl enc -aes-256-ecb -pbkdf2 -in ciphers.txt -out ciphers.aes.enc uses to encrypt file.

Password is 300573902

```
barretts% openssl enc -aes-256-ecb -pbkdf2 -in ciphers.txt -out ciphers.aes.enc
enter aes-256-ecb encryption password:
Verifying - enter aes-256-ecb encryption password:
barretts%
```

- Question 3.3:

openssl enc -blowfish -pbkdf2 -d -in treasure.bf.enc -pass pass:lucre

```
barretts% openssl enc -blowfish -pbkdf2 -d -in treasure.bf.enc -pass pass:lucre
I have deposited in the county of Bedford, about four miles from Buford's, in an excavation or
vault, six feet below the surface of the ground, the following articles: The deposit consists
of two thousand nine hundred and twenty one pounds of gold and five thousand one hundred pounds
of silver; also jewels, obtained in St. Louis in exchange for silver to save transportation.
The above is securely packed in iron pots, with iron covers. The vault is roughly lined with
stone, and the vessels rest on solid stone, and are covered with others.
barretts%
```

- Question 3.4:

barretts% curl -O http://ftp.sh.cvut.cz/slax/Slax-9.x/slax-ipxe.iso

barretts% md5sum slax-ipxe.iso
b347608199f2a0a70fb7a31beb460c20 slax-ipxe.iso

Slax iPXE	9.11.0	slax-ipxe.iso	b347608199f2a0a70fb7a31beb460c20	0.88 MB	changelog

Hashes are the same so files are verified to publisher.

- Question 3.5 [completion]:

Proving the integrity does not verify the authenticity as the file may have been hacked and had an alternate hash placed under the existing hash.

- Question 3.6 [challenge]:

PART 4: Asymmetric Cryptography

- Question 4.1:

As potentially everyone has your public key, anyone can encrypt a message to you using said key and pretend that it is from someone you know. If you verify (via the digital signature) that the message was signed by someone whose key you have verified, can you trust the message received and that is not from a malicious source.

- Question 4.2:

Copy and paste key info into a new document and imported it using:

barretts% gpg2 --import key.asc

outputs:

gpg: key FC8D7EE4F3B2AC61: "CYBR171 2021 <cybr171-staff@ecs.vuw.ac.nz>" not changed
gpg: Total number processed: 1
gpg: unchanged: 1

Key checked:

barretts% gpg2 --fingerprint cybr171-staff@ecs.vuw.ac.nz
pub rsa3072 2021-03-02 [SC] [expires: 2023-03-02]
08EA 5577 3F32 5954 2E69 BBBE FC8D 7EE4 F3B2 AC61
uid [unknown] CYBR171 2021 <cybr171-staff@ecs.vuw.ac.nz>
sub rsa3072 2021-03-02 [E] [expires: 2023-03-02]

Fingerprint supplied in assignment: `08EA 5577 3F32 5954 2E69 BBBE FC8D 7EE4 F3B2 AC61.`

Key's match so key can be trusted in that regards. However, there is no way to 100% verify a file handed over text and not given in person or in conversation with said person.

- Question 4.3:

```
barretts% gpg2 --verify document1.asc
gpg: Signature made Thu 04 Mar 2021 14:14:50 NZDT
gpg:                using RSA key 08EA55773F3259542E69BBBEFC8D7EE4F3B2AC61
gpg:                issuer "cybr171-staff@ecs.vuw.ac.nz"
gpg: Good signature from "CYBR171 2021 <cybr171-staff@ecs.vuw.ac.nz>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 08EA 5577 3F32 5954 2E69  BBBE FC8D 7EE4 F3B2 AC61
barretts% gpg2 --verify document2.asc
gpg: Signature made Thu 04 Mar 2021 14:15:19 NZDT
gpg:                using RSA key 08EA55773F3259542E69BBBEFC8D7EE4F3B2AC61
gpg:                issuer "cybr171-staff@ecs.vuw.ac.nz"
gpg: BAD signature from "CYBR171 2021 <cybr171-staff@ecs.vuw.ac.nz>" [unknown]
barretts%
```

```
barretts% stat document1.asc
  File: document1.asc
  Size: 796           Blocks: 8          IO Block: 65536  regular file
Device: 36h/54d Inode: 3940688     Links: 1
Access: (0644/-rw-r--r--)  Uid: (39367/joneselis1)   Gid: (   25/students)
Access: 2021-04-01 22:20:07.464035676 +1300
Modify: 2021-04-01 18:05:20.000000000 +1300
Change: 2021-04-01 18:05:29.427774583 +1300
 Birth: -
barretts% stat document2.asc
  File: document2.asc
  Size: 825           Blocks: 8          IO Block: 65536  regular file
Device: 36h/54d Inode: 3940689     Links: 1
Access: (0644/-rw-r--r--)  Uid: (39367/joneselis1)   Gid: (   25/students)
Access: 2021-04-01 22:20:07.464039082 +1300
Modify: 2021-04-01 18:05:20.000000000 +1300
Change: 2021-04-01 18:05:31.923401873 +1300
 Birth: -
barretts% ls -l document1.asc
-rw-r--r-- 1 joneselis1 students 796 Apr  1 18:05 document1.asc
barretts% ls -l document2.asc
-rw-r--r-- 1 joneselis1 students 825 Apr  1 18:05 document2.asc
```

I have used the function stat and ls to attempt to see what dates the files were changed after the key was put into place. However, upon trying to see this change, the files would only come up with the date downloaded. To check whether this issue would persist, I redownloaded the files and attempted to try using the stat and ls -l commands to check the modification dates (see command lines above), however the same issue persisted but both outputted the same results. I posted in the assignment 1 forum which Aaron Chen replied to, but he said that the stat command should have been able to tell me. Therefore, I am unable to check what file was changed after being signed.

- Question 4.4 [completion]:

Pgp works by the following steps.

You download the public key of the software author. Check the public key's fingerprint to ensure that it is the correct key. Import the correct public key to your GPG public keyring. Download the software's signature file. Use public key to verify PGP signature.

So by these steps the PGP signature must include information about the signers public key and verify it against the encrypted information in the signature.

- Question 4.5 [challenge]:

PART 5: Ransomware

- Question 5.1:

As the CIA is an intelligence agency, its property might come under fire as a hacktivist or hacker may intend to leak information to the world. (“So instead of just locking data away, attackers are threatening the exact opposite: publish it for all the world to see” found at the bottom of the article)

- Question 5.2:

Crypto ransomware differs from normal ransomware in that encrypted all the content on the infected server or software. As well as that the ransom was done in the form of crypto currency so ransomware authors could take payment without involving the trappings of the conventional banking system such as credit cards and avoid detection via bank transfers.

- Question 5.3:

Marcus discovered that affected computers tried to access a particular web address after infection. The address was not registered to anyone, so he bought the domain which stopped the virus.

- Question 5.4 [completion]:

Instead of just encrypting file, doxware threatens to publish the data online if the ransom is not met. The software could be potentially used to threaten to leak sensitive data from the CIA databases including investigation information, sensitive information about people in witness protection, etc.

- Question 5.5 [challenge]: