CS 306 Homework 2

Problem 1 (30%)

- (1.1) Completed in `./Mac.java`

```
josh@josh:~/.../$java Main
## Key ##
CS-306_Test_Key

## Message ##
This is a test message.  There are many like it but this one is mine.

## Tag ##
29BA1525FA2E2E390574CEAB96BF3F3F

Success, tag was verified.
```

- (1.2) `29BA1525FA2E2E390574CEAB96BF3F3F`
- (1.3) I implemented CBC-MAC, where I broke up the message into blocks, xor-ing with the previous block output and then encrypting the new block, iterating through the whole message to produce one final tag. This algorithm can handle a message of any length with padding, and produces a tag of the set block size to use for verification.
- (1.4) The domain-extension involves using blocks consecutively with each other, so the order and integrity of each block is ensured. This prevents reordering attacks, truncation attacks, and max-and-match attacks.

Problem 2 (20%)

- (2.1) Mallory and Eve could collaborate by having Mallory continue to send out Bob's old public key even after he registered a new one. Then Eve will continue to be able to use the secret key she stole to decrypt Bob's message. After that they could use his new public key and send the message to him, which he can then decrypt with his public key without knowing anything is wrong. This is a MITM attack, because the integrity of the communication line is compromised by the attacker.
- (2.2) The timestamp could be used to make sure the most recent record for the public key is being sent out. If there are multiple different public keys being sent out in the same day then something is wrong.

Problem 3 (30%)

- (3.1) The split architecture means you need to have the data from both servers in order to get information about a users password. If an attacker only compromises the user server they would know the location of a real password, and if they compromise the password server they would have a huge list of mostly fake passwords, with no way to associate them back to users
- (3.2) If a lot of the fake passwords are being used it means that someone likely got the list of the mostly fake passwords, and is trying to use them to log in for other user accounts.
- (3.3) paSSword5 -> {paSSword6, PassWord5, password6}
- (3.4) Blink128. It's probably one of the correct ones along with Blink123. These are similar and the third is completely different, choosing this likely gives me a 50% chance of success

---

Problem 4 (30%)

- (4.1) Doing the modular exponentiation in binary makes it much more efficient. Signing is made more efficient through the use of certificate authorities.
- (4.2) Completed in `./rsa_skeleton.py`

```
josh@josh:~/.../$ python rsa_skeleton.py
########################
('Public key:', [1081, 35])
('Private key:', [1081, 347])
('Original message:', 'Hello world!')
('Encrypted message:', [653L, 863L, 72L, 72L, 1042L, 357L, 982L, 1042L, 528L, 72L, 225L, 774L])
('Decrypted message:', 'Hello world!')
------------------------
('Public key:', (8926157, 6183893))
('Private key:', (8926157, 384125))
('Original message:', 'Hello world!')
('Encrypted message:', [2935765L, 6585186L, 4181891L, 4181891L, 7149857L, 4952722L, 5948679L, 7149857L, 4110962L, 4181891L, 2812361L, 578800L])
('Decrypted message:', 'Hello world!')
```