# CSE 484 / CSE M 584 - Homework 2

This homework is focused on cryptography.

## Overview

- **Due Date:** Friday, May 6, 2016, 8pm.
- **Group or Individual:** Do this assignment as an **individual**. But you are allowed to talk with others in advance of actually doing the assignment.
- **How to Submit:** Submit a PDF to Catalyst. Your assignment does not need to be entirely typed / developed with computer software. You could hand-write your assignment, and hand-draw some diagrams, and then submit a PDF scan of your hand-written assignment, just make sure it is legible.
- **Total Points:** 46 (+12 extra credit)

**Q1 (3 points).** What is the main concern cryptographers have with the Encrypt-and-MAC method for combining a symmetric encryption scheme with a symmetric MAC to create a symmetric authenticated encryption scheme?

**Q2 (5 points).** This message was encrypted with the RSA primitive, where N=33 and e=3. Decrypt it and submit the corresponding plaintext.

Tips: You are welcome to write a program to aid in the decryption, and you might want to compute the private decryption exponent d.

For this cryptogram 'A' is encoded as a 1 before encryption, 'B' as a 2, and so on.

Here is the cryptogram: 14 17 3 28 27 24 16 4 14 9 13 24 1 19 23 1 28 26 5 27 24 16 4 14 26 31 23 3 14 17 14 17 26 24 28 1 4 24 3 19 3 14 3 22 26

**Q3 (8 points).** The following question has you use RSA, but with larger values (but still not anywhere close to the size of the numbers one would use in a secure cryptographic protocol like TLS/SSL).

You may use a program that you write, Wolfram Alpha, or any other computer program to help you solve this problem.

For all of these, it is sufficient to just include your number in the answer, unless the question explicitly asks for additional detail.

Let p = 9497 and  q =7187 and e = 3.

- Compute N = p * q.  What is N?
- Compute Phi(N) = (p-1)(q-1).  What is Phi(N)?
- Verify that e is relatively prime to Phi(N).  What method did you use to verify this?
- Compute d as the inverse of e modulo Phi(N).  What is d?
- Encrypt the value P = 12345678 with the RSA primitive and the values for N and e above. Let C be the resulting ciphertext.  What is C?
- Verify that you can decrypt C using d as the private exponent to get back P.  What method did you use to verify this?
- Decrypt the value C' = 12345679 using the RSA primitive and your values for N and d above.  Let P' be the resulting plaintext.  What is P'?
- Verify that you can encrypt P' using e as the public exponent to get back C'.  What method did you use to verify this?

**Q4 (5 points).** Suppose you, as an attacker, observe the following 32-byte (3-block) ciphertext C1 (in hex)

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03
46 64 DC 06 97 BB FE 69 33 07 15 07 9B A6 C2 3D
2B 84 DE 4F 90 8D 7D 34 AA CE 96 8B 64 F3 DF 75
```

and the following 32-byte (3-block) ciphertext C2 (also in hex)

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03
51 7E CC 05 C3 BD EA 3B 33 57 0E 1B D8 97 D5 30
7B D0 91 6B 8D 82 6B 35 B7 8B BB 8D 74 E2 C7 3B
```

Suppose you know these ciphertexts were generated using CTR mode, where the first block of the ciphertext is the initial counter value for the encryption.  You also know that the plaintext P1 corresponding to C1 is

```
43 72 79 70 74 6F 67 72 61 70 68 79 20 43 72 79
70 74 6F 67 72 61 70 68 79 20 43 72 79 70 74 6F
```

Compute the plaintext P2 corresponding to the ciphertext C2.  Submit P2 as your response, using the same formatting as above (in hex, with a space between each byte).

**Q5 (5 points).** Consider an insecure version of SSH that uses ECB mode for encryption. Whenever a user types a key into the ssh client, that key is immediately encrypted and sent over the wire to the server. This immediate encrypt-after-key-press procedure is what enables the interactivity of a remote shell. Now consider the following sequence of plaintext packets (written in hex):

P1 = 6C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII l
P2 = 73 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII s
P3 = 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII space
P4 = 2A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII *
P5 = 2D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII -
P6 = 66 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII f
P7 = 72 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII r
P8 = 6F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII o
P9 = 6D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII m
P10 = 0D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 // ASCII <enter>

This corresponds to a user typing “ls *-from<enter>” into their ssh client.

Suppose an attacker knows what the user is typing via some out-of-band channel (e.g., shoulder surfing) and also eavesdrops on this communications and intercepts the corresponding ciphertexts:

C1 = 4E B6 48 B2 E0 BE A5 B1 21 2F 07 54 DF CF A4 39
C2 = 11 70 78 65 88 89 06 62 82 0C 0A 6A 55 6F 87 46
C3 = EF 7F 1F 25 3E 99 98 8D 1A FC BE 7A D9 D6 ED 7E
C4 = 5B 40 2B 18 0B 94 E8 13 DA F3 DE 21 A0 27 2E C4
C5 = 93 80 19 1F 06 B4 4B 19 9D 70 86 28 34 12 26 DC
C6 = 68 74 EB 1B 16 5F 70 45 05 29 B9 66 0A CC D3 6C
C7 = 56 E8 77 E1 7E BF 01 19 27 87 03 FE E1 1D 65 A8
C8 = 9D 37 51 F0 68 C8 F7 BA 44 B2 E9 5C 09 94 1D 5A
C9 = 62 30 38 8F A4 D7 C1 68 56 88 CE 2C 29 2D F5 23
C10 = D5 89 74 7E 45 89 08 FA 5B 63 98 42 E6 B2 31 85

The attacker can now inject messages into the communications channel from the client to the server. One thing an attacker might try to do: generate a sequence of ciphertext packets that, when decrypted, are interpreted as “rm -rf *<enter>” on the server. Give such a sequence of ciphertext packets in your answer below.

**Q6 (3 points).** Consider a Diffie-Hellman key exchange with p=29 and g=2. Suppose that Alice picks x=5 and Bob picks y=3. What will each party send to the other, and what shared key will they agree on? Show your work.

**Q7 (6 points).** Below we give you the entry for a password stored on a Linux machine. The password is weak. Your task: find the password.

To do this, we recommend using either john or hashcat. We strongly recommend using Linux for this question (Use attu or a VM if you don't have a native Linux). You can likely install John the Ripper from the repository using apt-get or yum. The Linux package name is most likely john, e.g., for Ubuntu, run "sudo apt-get install john". Otherwise, you can download john and build it from source (you will have to use this option if you are using attu, during build specify the architecture as linux-x86-64): John the Ripper 1.8.0 (sources, tar.gz, 5.2 MB) and its signature

Here is the password entry, from a Linux machine:

> ahaha:$6$FlItSjGy$54IaMBy6ThxAbvnUztWzrl4FjtEwn1sX81/V8Ll7PtMpPAiy57QM4q.oyUD2cHFL4nwhguDk7eP7c3t0ArKep.:16769:0:99999:7:::

**Q8 (4 points).** The goal of this task is to give you a better understanding of Certificate Authorities (CA) and certificates.

Take a look at the CAs certificates that your computer trusts.

- Mac: Spotlight search 'Keychain Access'
- Windows: Control Panel -> Search 'Internet Options' -> Content -> Certificates

Answer these questions:

1. How many root CA certificates does your computer have?
2. What is something that you found interesting from looking at the root CA certificates?
3. Go to google.com using your favorite browser, and find a way to look at the certificates for google.com. List the chain of certificates your browser sees.
4. What is a possible risk of trusting a CA?

**Q9 (10 points).** For this task, the goal is to give you experience with sending encrypted emails. To successfully complete this task, you will need to set up your email client and send/receive an encrypted email to/from the TAs. For this assignment, you can reach your super secret agent TA "Batman" at brucewayne@uwctf.ninja.

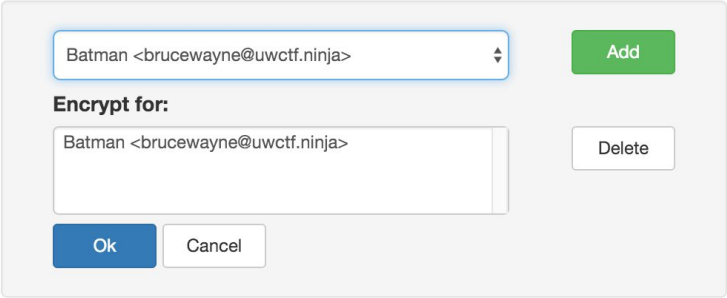
### Setup information

Setting up your email client to send encrypted emails is a bit complicated, and the following link is helpful if you don't want to follow the recommended setup below or you know what you are doing: How to Encrypt Your Email and Keep Your Conversations Private

## Recommended setup

We found that using Mailvelope extension for the browser is easiest since you will be able to use your Gmail web client.

1. Get Mailvelope, which works with Chrome or Firefox. The rest of this guide assumes you are using Chrome.
2. Once installed, you should see a  icon on the top right corner of the browser, click on it then click 'Options'
3. Generate a key for yourself
    - Name: Your name
    - Email: The email address you're using for this part of the assignment
    - Password: Optional but **recommended** password to secure your key
4. Import Batman's public key from here by pasting the contents into Mailvelope (Options -> Import Keys)
5. You are now set up! Go to your email page, hit Compose and you should see  . Click on it to start composing an encrypted email, click 'encrypt' when you are done.
6. You will see a dialog box that allows you to choose who you want to encrypt the email for. Make sure to click "Add" and my key is in the "Encrypt for: " field! **Note** that you can "Encrypt for: " you and me.




7. Using the email account that you associated with your key above (**note that you will NOT get a response if your email account and key don't match**), send an email to the TA in this format:

    To: brucewayne@uwctf.ninja
    Subject: [CSE 484] Encrypted email
    Content: Whatever secret message you'd like to send us :)
    Attachment: <your public key> (Select your key from 'Display Keys' on the Mailvelope site, and export the public key **only.** Download <yourkey>_pub.asc and attach it.)

Once you are OK with the contents, hit encrypt, select the correct key for the recipient and transfer the encrypted contents.

**Note:** In order for the TA to send you an encrypted email, you will need to attach **your** public key with your email.

If you don't want to use your main email account, you can use this with a throw away email address inside a virtual machine.

Once the TA receives this email, a secret reply will be sent back to you. Submit the content of this email to your writeup.

Though we have an automated bot to respond to your emails, **please start early on this** in case we have to fall back to a manual process. *Please email the TA at least 48 hours before the deadline.*

## Deliverables

1. The email address you used
2. Secret value provided by the TA
3. Answers to short answer questions
    a. Does this process (PGP encryption) involve the use of symmetric or asymmetric encryption or both?
    b. We recommended a browser extension for ease of use, but what are the security risks of enabling this browser extension? (Hint: what permissions did the extension ask for during install?)

***----- (Extra Credits Below) -----***

**Extra Credit 1 (6 points).** Download the required files here: link

(a) What is the modulus for the RSA pubkey? In your answer, explain how you obtained the modulus. We recommend using command line openssl on Linux. Hint: Run openssl rsa --help. You will have to give OpenSSL an input file, tell it what type of file (e.g., public key or private key), and tell it what to output. (Also remember the course's Gilligan's Island rule, and the fact that you can talk with others in class.)

(b) Factor this RSA modulus and give us the factors in format p=?, q=? (p < q). In your answer, explain how you factored the modulus.
We recommend the following options:

1. use Sage Math to factor,

e.g., print factor(<number>). To use Sage Math, you need to create an account, start a new worksheet, and then run the worksheet; you might test with small numbers; to factor the full modulus may take some time (e.g., start before a meal and check afterwards)

2. use yafu. You might need to compile from source if you choose this
3. use factordb. (Note that factor db only recognize Decimal Number)

(c) Generate the private key and decrypt the encrypted message. What's the decrypted message? In your answer, explain how you generated the private key and provide the decrypted message. You might use rsatool.py to generate private key once you figure out the factors and use openssl to decrypt it.

Hint: Run python rsatool.py --help to find out your options when generating private key
Run openssl rsautl --help to find out how to decrypt a file using a private key
For rsautl, you will need to tell it the input file (the encrypted file), tell it to decrypt, and tell it what private key to use.
For rsatool.py, on the departmental machines you may need to set up a virtual python environment with the following, before running rsatool.py (if you use C shell, run source <environment_name>/bin/activate.csh instead):

```
virtualenv <enviroment_name>
source <environement_name>/bin/activate
pip install pyasn1 gmpy
```

**Extra Credit 2 (6 points)**. I have a secret backdoor on my mail server, it allows you to create profiles. However, there's some functionality that only admins are able to perform. ECB is my favorite block cipher mode!

The service is available here**: 128.208.6.104:31713**. Find the flag.

You can download the server code here: link

If you use netcat, then run the following to connect
**nc 128.208.6.104 31713**

(I have timeout for the service, so it might be hard to solve the problem without scripting)
(If you are scripting and don't want to deal with raw sockets, then try pwnlib)

There are hints down there in white color! Read it if you need it.