SSN Lab Assignment:

Asymmetrical encryption

RSA was first described in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman of the Massachusetts Institute of Technology. Public-key cryptography, also known as asymmetric cryptography, uses two different but mathematically linked keys, one public and one private. Many protocols like SSH, OpenPGP, S/MIME, and SSL/TLS rely on RSA for encryption and digital signature functions. It is also used in software programs - browsers are an obvious example, which need to establish a secure connection over an insecure network like the Internet or validate a digital signature. RSA signature verification is one of the most commonly performed operations in IT.

1. Create a 2048 bit RSA key pair using *openssl*. Write your full name in a text file and encrypt it with your **private key**. Using OpenSSL extract the public modulus and the exponent from the public key. Publish your public key and the base64 formatted encrypted text in your report. Verify that it is enough for decryption.

Hint: study openssl params and encryption vs verification vs decryption vs signing

2. Assuming that you are generating a 1024 bit RSA key and the prime factors have a 512 bit length, what is the probability of picking the same prime factor twice? Explain your answer.

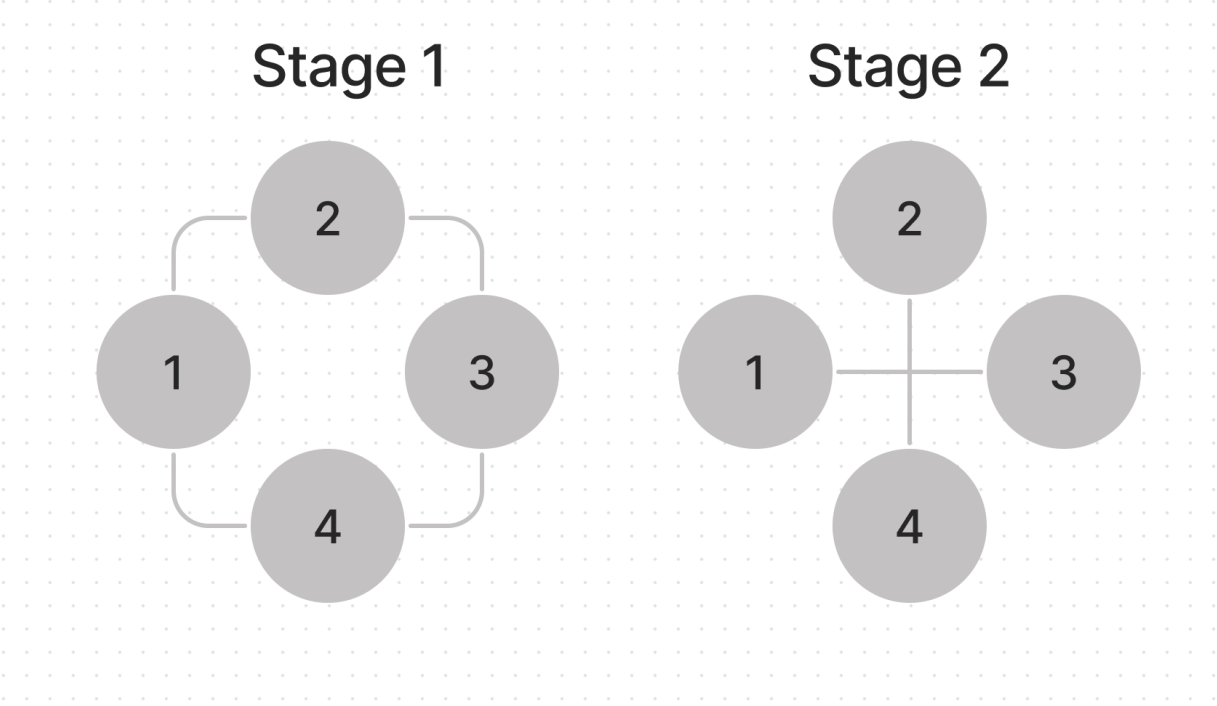
Hint: How many primes with length 512 bit or less exist?

3. Here you can find the modulus (public information) of two related 1024bit RSA keys. Your keys are numbered using the list. Your task is to factor them i.e. retrieve ***p*** and ***q***. You may use any tools for this. Explain your approach.

Hints: study the RSA algorithm. What private information can two keys share? What practical attacks exist? You may have to write code or use existing code for simple arithmetic operations. Be careful with bigInt values!

4. Now that you have the ***p*** and ***q*** for both keys, recreate the first public and private key using this script. Encrypt your name with the private key and post the public key and the base64 formatted encrypted data in your report.

5. Using GPG implement a small web of trust in your group of 4 people:



- Stage 1: create your PGP keys and exchange and sign each other's key as in the picture above (as you can see on the picture, at this stage the 1st participant exchanges with 2nd and 4th, but not with 3rd). After exchange, verify that you can communicate securely with that person.

- Stage 2: obtain a public key of a person that you have not contacted yet and send them a private message alongside your public key. At this point you do not need to sign each other's keys, since your keys are already signed by 2 other trusted members, so you should be able to securely communicate without it.

Describe what you have done to implement this model.

(If your group size is < 4 - please some person from another group to also participate in your group)