RSA assignment

Name : Abdelrahman Mohamed salem Hassan

Section: 1

BN: 38

Student code: 9202794

So…., I implemented 2 programs (one is called client and the other is called server) and both are chatting using RSA encryption/decryption and there is a third program called hacker which is trying to factorize the **n** which is given through the public key of client and server where each one of client and server has its own PU and PR keys and the hacker is trying to factorize **n** that's given in the public key to get the **d** through which he can decrypt the encrypted messages between them.

So I tried the number of bits for **P** and **Q** (that are prime numbers) to be the following : 27, 32, 40, 50, 60, 70, 80, 90, 128, 256, 512, 1024, 2048 and each time tried to factorize **n** which is given by equation : n = P * Q, so that I could get **d** from the equation : d = e^-1 mod(phi) where phi = (P-1) * (Q-1) and e is given from the public key of the receiver.

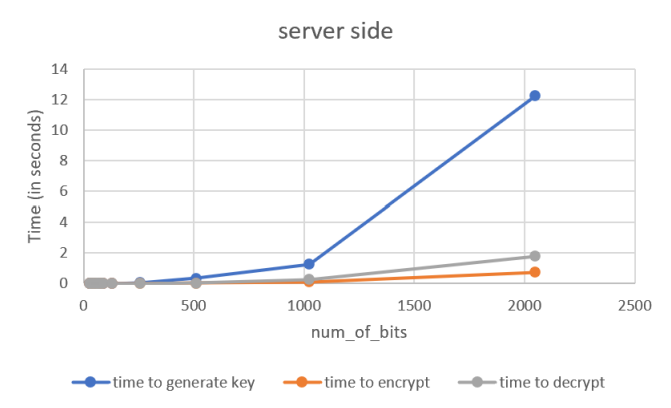
So I tried for different number of bits stated above and noticed the : (Time taken to encrypt), (Time taken to decrypt), (Time taken to generate keys), (Time taken to be hacked). For both client and server.

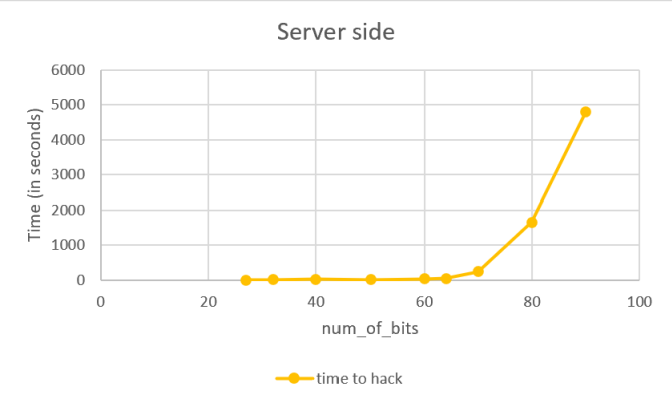
The processor of the device which was doing encryption/decryption and hacking is intel-core-i9-10980hk-8cores-16threads with 32GB RAM (doing the same procedures on another higher device will result in smaller amount of time needed especially if it was a super computer).

The next 2 tables illustrates both server side and client side time calculations (note that the time calculated in tables is in seconds), there are ∞ symbols in the table indicating that my device would so long time to factorize **n** (hack the system (passive attack)). From the tables and graphs, you would notice that time needed to generate keys (done only once) can be excluded from the calculations because it's relatively small and it's only done once while time to encrypt and decrypt message is even smaller for very large keys and choosing very big key would make it so difficult for the attacker to do an attack on the system or get the private keys, as you notice in the client side of calculations at size of key = 90 bit, it required my device about 13544 second which is equivalent to 3.76 hours to get the private key of the sender (I couldn't go higher than 90 bits as it would take my device forever to break the key), so choosing a big size key (I recommend **1024** bit or **2048** bit) would make it so hard for the attacker to do passive attacks.

Server Side time tables

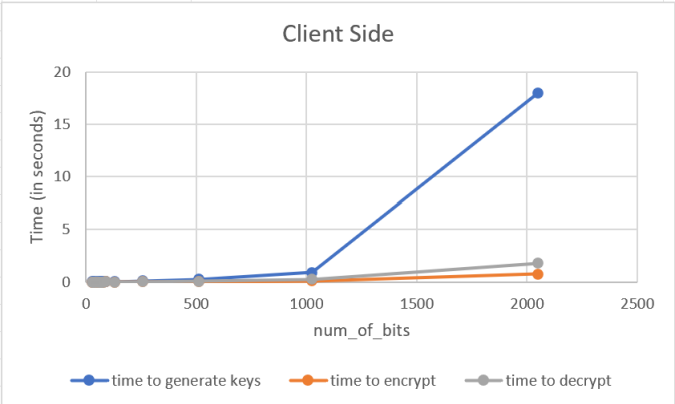
calculations done for the server side				
number of bits	time to generate keys	time to encrypt	time to decrypt	time to hack
27	0.003	0	0	0.004
32	0.004	0	0.001	0.64
40	0.005	0	0.001	15.5
50	0.009	0	0.001	1.83
60	0.005	0	0.001	26
64	0.006	0.001	0	45
70	0.009	0.001	0.001	237
80	0.007	0.001	0.001	1653.82
90	0.021	0.002	0.003	4805
128	0.01	0.001	0.001	∞
256	0.06	0.003	0.007	∞
512	0.34	0.02	0.04	∞
1024	1.26	0.1	0.26	∞
2048	12.24	0.74	1.8	∞

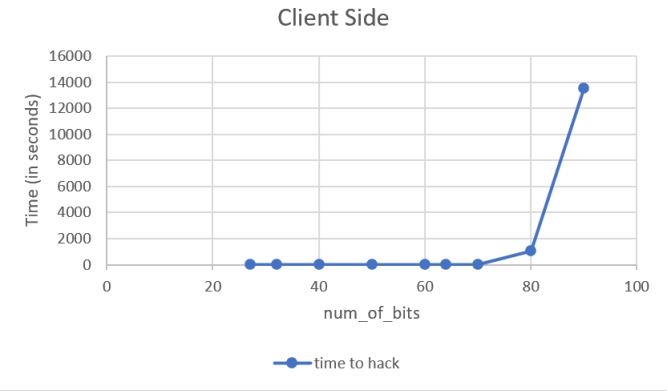




Client Side time tables

calculations done for the client side				
number of bits	time to generate keys	time to encrypt	time to decrypt	time to hack
27	0.003	0	0	0.002
32	0.005	0	0.001	0.623
40	0.009	0	0.001	15.01
50	0.005	0	0.001	4.24
60	0.005	0	0.001	4.5
64	0.006	0.001	0	6.6
70	0.007	0.001	0.001	4.8
80	0.025	0.001	0.001	1055.5
90	0.0064	0.002	0.003	13544
128	0.03	0.001	0.001	∞
256	0.1	0.003	0.007	∞
512	0.25	0.02	0.04	∞
1024	0.9	0.1	0.26	∞
2048	18	0.74	1.8	∞





Conclusion

Increasing the size of the key (number of bits) would increase slightly the time to generate the keys and have a small effect on the time to encrypt/decrypt the message but it would make it very hard for the attacker to get the private keys, notice the difference between the time to hack using 80 bit key and 90 bit key, you would notice a huge difference while the time to generate keys and encrypt/decrypt keys has a neglected change in time so I recommend choosing 1024 or 2048 bit key size which would make it very hard for the hacker to get the private key using factorization and I also noticed that any small increase in the number of bits of the key size would result in a very longer time for the hacker to factorize **n**.