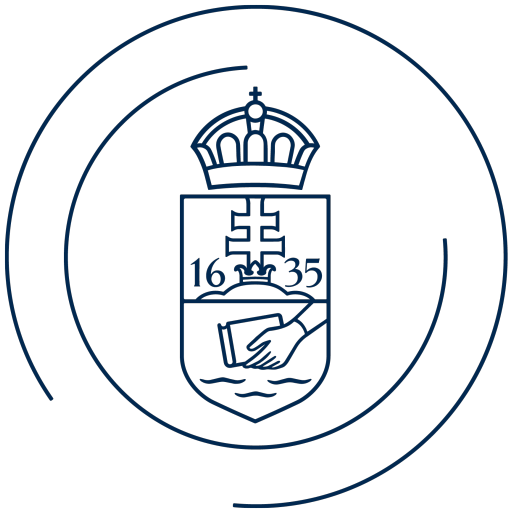

ELTE
EÖTVÖS LORÁND
UNIVERSITY

Cryptography and security
(IPM-18sztKVSZKRBG)

Assignment II

Minimum requirements

- There are 7 different IT security related problems described in this paper with varying difficulties;
- You need to collect at least **10 points** from this assignment;
- Always provide step by step solutions;
- Submit your final solution in **one PDF** file including every necessary source code;
- Please send your solutions to **ntihanyi@inf.elte.hu** till **15th April 2022**;
- It is **strictly prohibited** to share your solutions with others.

Challenge #1 - Simple factorization

We have the following 256-bit number:

```
N=450843386254514383254234904819564314133047200507653780729741006356265116334 43
```

Questions:

a. Provide the prime factorization of N. (**1 point**)

b. Implement the solution in any chosen programming language (**2 points**)

Challenge #2 - Special prime numbers

We have the following 2048-bit number:

```
N=22311140989820914550000986626313649557247770212361679138331581359889844086641006434638927599137456840774923340635937929656278287765519657407156093608742702815890690039283703218479832880846766137483119331597676869176397230051282616283038805155392350907968441315249545440736908368329882012764550410514806228109178699532952225563465463648637779935978713730794113893058678857727752118507890696573569149769436760361000516604217723497737161022058424978089719431075642078566217626644560471991389891197867565872364584063174387002655505428530638739514738691250640336477678105795431139283491632703173102223358411228793542047717
```

Questions:

a. Provide the prime factorization of N. (**2 points**)

b. Implement the solution in any chosen programming language. (**3 points**)

c. Why is it possible to factorize N? (**1 point**)

Challenge #3 - RSA factorization of a 2048-bit N modulus

The RSA (Rivest–Shamir–Adleman) algorithm is a public-key cryptosystem publicly described in 1977. The security of RSA is based on the problem of factoring large integers. As of today no efficient algorithm exists for solving the factorization problem. The following random number generator was used to generate a 2048-bit RSA N modulus:

```
from Crypto.Util import number
# Generate 1024-bit P prime
x= number.getRandomNumber(1024)
while True:
    x=x+2
    if (number.isPrime(x)==True):
        P=x
        break
# Generate 1024-bit Q prime
while True:
    x=x+2
    if (number.isPrime(x)==True):
        Q=x
        break
N=P*Q
print(N)
```

Output:

```
155355812253529420414638216075447729177064335988
234493840142817233685295530583281678982690891776 2
000791268397390610001671024565003536712972449937 3
495769314222669243742736648616879764786103502017 3
796390022665777386815046135035949387719693615621 8
569210023718759878041910536745641395789995483638 1
965722968369202450681646999512357558452958618233 9
942875317926811037514175102414344370850616293767 2
150318431515228140070417967917887674350832781749 0
684463540577250388181258613843343413429747533407 9
678515685196128198065071015271737968667703278313 8
068399207659526118075577650936962908393680573715 0
396192964071516312072320164 79
```

Questions:

a. Recover P and Q prime numbers. (**3 points**)

b. Make the python code secure with a simple modification. (**1 point**)

Challenge #4 - Encrypted text

We have the following RSA encrypted text:

```
Encrypted text: 25638920938252605568895546745915864430032601449246389102003905595590281535614494083115285466424934888722445856680322471966324253613179786613909109393468757365506827907544722018003777040276697115524948537504116130838520034397491469004478806730598503593969539017431415343224253330093172038891720598510126234425381204670005066424654823235126373391281293899982560510742145411191323345651012335044957173926825573691423859516454384362389818720561877499366050844916793847290840460938497939358354057176088478166060565796668442948448805498970991098488521722959778012607576789719059863048236773059701554299604319833626518330576
```

The RSA paramaters are the following:

```
N=30479915487669326930154938380968766431833949993346991132694149723187817935339528962855546989243335804010036419422987797217836348288026014284489006519127280859366514645387949472747931071629724217991985973546443890005626067483439373338965791299306567969307130124133997845851509755838943259384017879067734826353016270127156768168167794991449166504474185759354999418577081672681139858664445489086000031773896480822893296682174804689134484817843679020394838926915616313757974443043422503488635552736371022127054455105212424089722728800842893299361711485229280789384148449349631381448714241530299081398302331503199102701259

d=11653497144735497884994618170070866540770302042865794235499295201396419640902125169328340340715477749074255497833007388725259401239804504935020538571337904946719360933052868607636066784592306631829732708838568206537848427192769586306276848689850384875824782329377688695294280787220293898872168942670556045347189823824894151854920078650742791530141827190011712249760979061729445384263029983575431519903421329627975147558091383199470264796128084370015805972377702537067596481532759675574858239622528439695446531885371755719959956839104110377238118593431128430936964712765080357811966133828100326883105932717510977473649

e=65537
```

Questions:

a. Recover the original cleartext message. (**4 points**)

b. Implement the solution in any chosen programming language. (**4 points**)

Challenge #5 - Creating the RSA private key

We have the following two prime numbers:

```
P=4710707783152652963131393039062535568792811521273534852738842882577711199862 7
Q=2153688799415487013196539089099588576672202370242854179869245695416258432896 1
```

Questions:

a. Recreate the the RSA public and private key from P and Q prime numbers. (**2 points**)

b. Implement the solution in any chosen programming language. (**4 points**)

Challenge #6 - Creating RSA private key from $\phi(n)$

We have the Euler's totient function from an RSA key generation:

$Phi = \phi(N) = (P-1) \times (Q-1)$

```
Phi=9310759785104321947994765912354331857647523302309790378091286977801859079534382061575084536844264024705919624493712131765625613234448097330190150457225633429250079100423433516751031312413788281860214860846815791111608302699489356548437243271956303529374144023766085787903673585968742256120688386792 2357842063
```

```
N=9310759785104321947994765912354331857647523302309790378091286977801859079534382061575084536844264024705919624493712131765625613234448097330190150457225633429250079100423433516751031312413788281860214860846815791111608302699489356548437243271956303529374144023766085787903673585968742256120688386792 2357842063
```

Questions:

a. Recreate the the RSA public and private key from $\phi(n)$. (**6 points**)

Challenge #7 - Diffie-Hellman key exchange

Diffie–Hellman key exchange is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols as conceived by Ralph Merkle and named after Whitfield Diffie and Martin Hellman.

Questions:

a. Implement the Diffie–Hellman key exchange protocol in python. (**4 points**)

b. Demonstrate your python solution with real examples. (**3 points**)