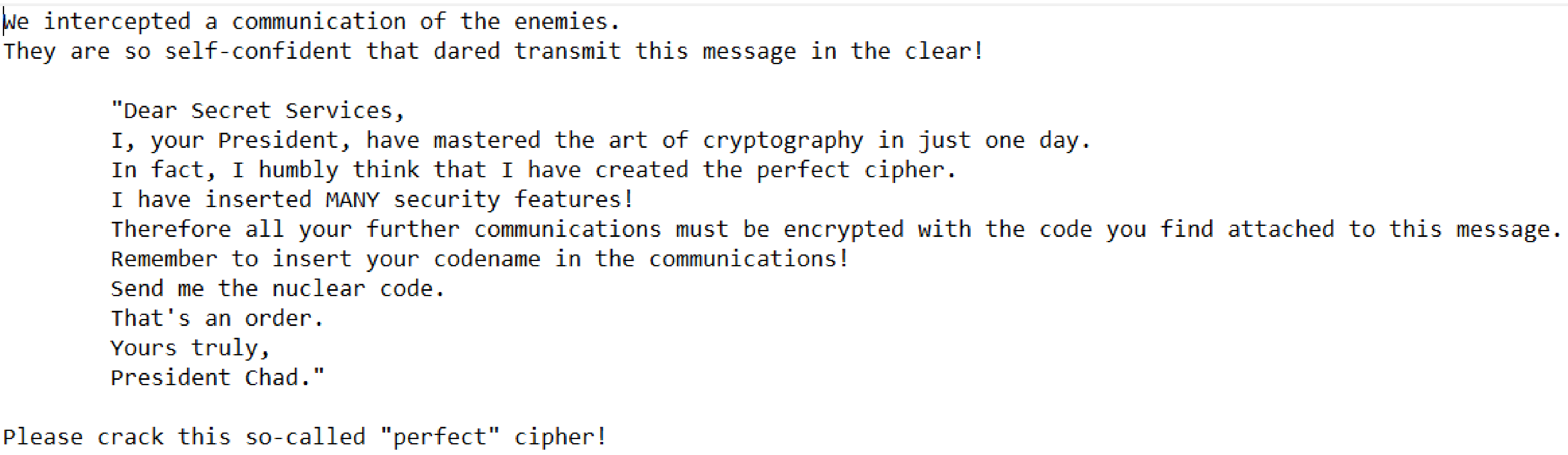
# Python3 Exercises

Michele La Manna
Dept. of Information Engineering
University of Pisa
michele.lamanna@phd.unipi.it

Version: 2022-03-14

## NUCLEAR_THREAT EASY

# Nuclear_threat Easy



```
We intercepted a communication of the enemies.
They are so self-confident that dared transmit this message in the clear!

        "Dear Secret Services,
        I, your President, have mastered the art of cryptography in just one day.
        In fact, I humbly think that I have created the perfect cipher.
        I have inserted MANY security features!
        Therefore all your further communications must be encrypted with the code you find attached to this message.
        Remember to insert your codename in the communications!
        Send me the nuclear code.
        That's an order.
        Yours truly,
        President Chad."

Please crack this so-called "perfect" cipher!
```

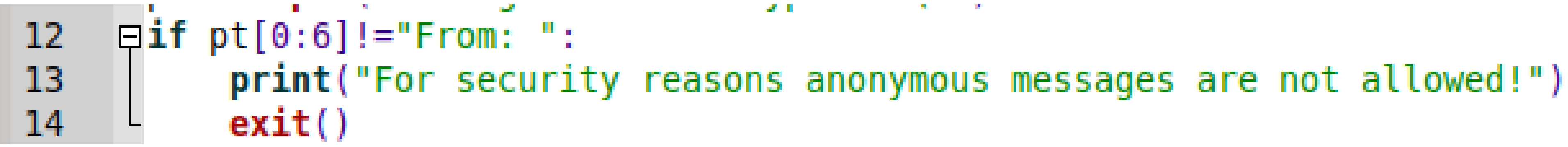
## Nuclear_threat Easy

Analyze the source code and spot the algorithm vulnerabilities.
With the gained intel, write a script that can decrypt the captured ciphertext.

30 minutes

# Hint#1: Known-Plaintext

Lines 12—14 check for a static string at the beginning of the message!



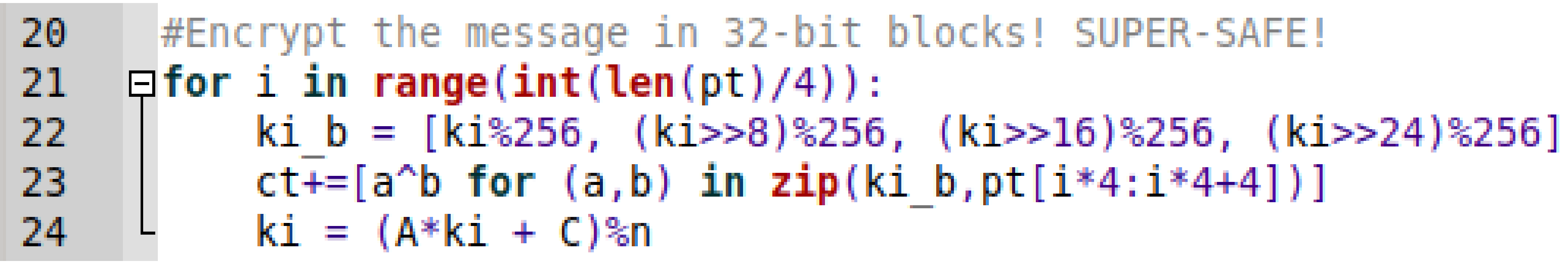
```
12  if pt[0:6]!="From: ":
13      print("For security reasons anonymous messages are not allowed!")
14      exit()
```

How can you exploit this procedure?

Answer: This can help you carry out a “Known-Plaintext-Attack”.

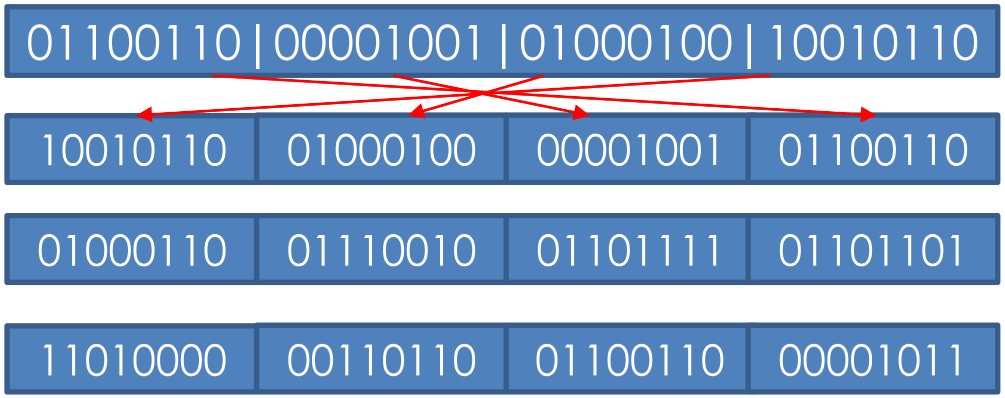
# Hint#2: Bytewise XORing

Lines 20—24 encrypts blocks of 32 bits, however the XOR is done over a single byte!



```
20      #Encrypt the message in 32-bit blocks! SUPER-SAFE!
21      for i in range(int(len(pt)/4)):
22          ki_b = [ki%256, (ki>>8)%256, (ki>>16)%256, (ki>>24)%256]
23          ct+=[a^b for (a,b) in zip(ki_b,pt[i*4:i*4+4])]
24          ki = (A*ki + C)%n
```

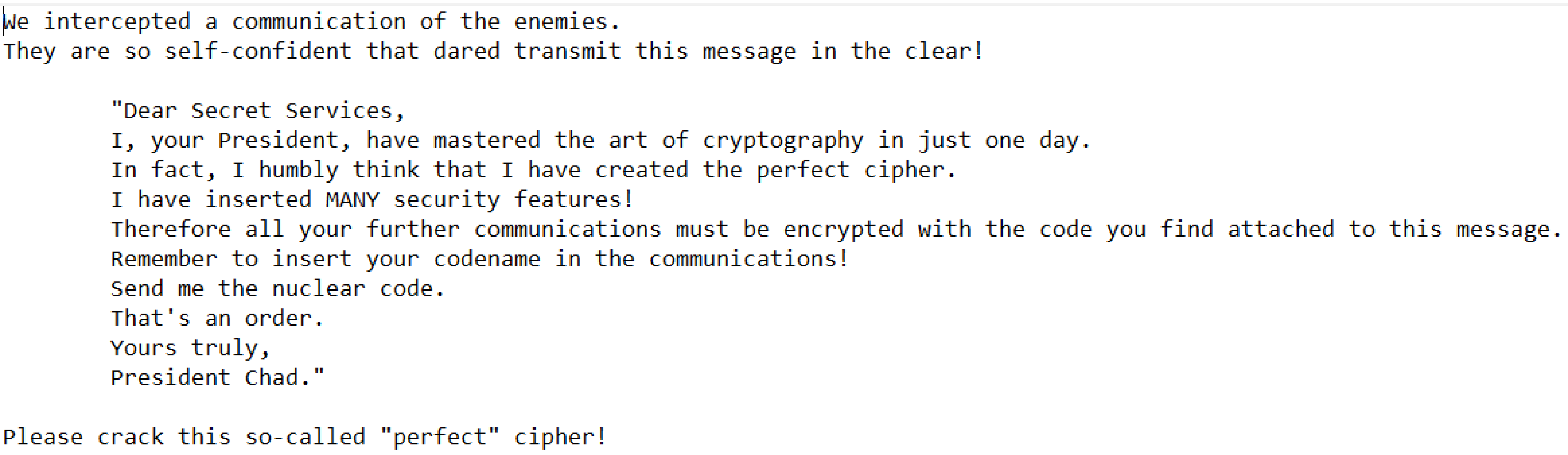
Here's the graphic representation of an example:



| | | | | | |
|---|---|---|---|---|---|
| ki = | 01100110 | 00001001 | 01000100 | 10010110 | 1711883414 (uint) |
| Ki_b[0:4] = | 10010110 | 01000100 | 00001001 | 01100110 | 150;68;9;102 (uint) |
| pt[0:4]= | 01000110 | 01110010 | 01101111 | 01101101 | b'From' (ASCII) |
| Ct[0:4]= | 11010000 | 00110110 | 01100110 | 00001011 | ciphertext |

## NUCLEAR_THREAT HARD

# Nuclear_threat Hard



We intercepted a communication of the enemies.
They are so self-confident that dared transmit this message in the clear!

> "Dear Secret Services,
> I, your President, have mastered the art of cryptography in just one day.
> In fact, I humbly think that I have created the perfect cipher.
> I have inserted MANY security features!
> Therefore all your further communications must be encrypted with the code you find attached to this message.
> Remember to insert your codename in the communications!
> Send me the nuclear code.
> That's an order.
> Yours truly,
> President Chad."

Please crack this so-called "perfect" cipher!

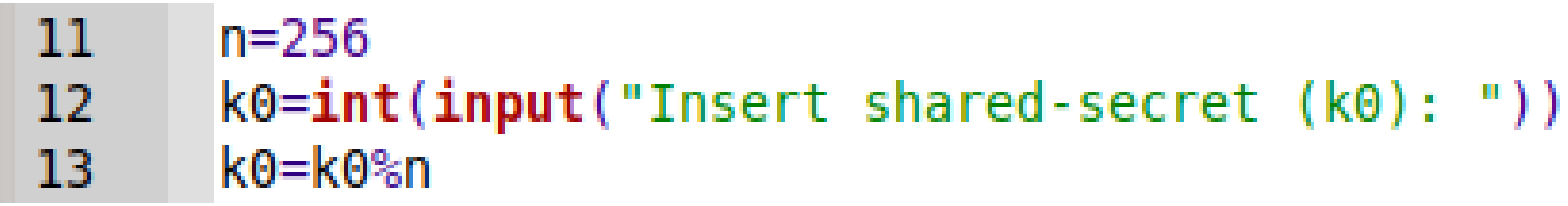
# Nuclear_threat Hard

Analyze the source code and spot the algorithm vulnerabilities.
With the gained intel, write a script that can decrypt the captured ciphertext.

30 minutes

# Hint#1: BRUTEFORCE TIME!

Lines 11—13 reveal the true vulnerability of this code!



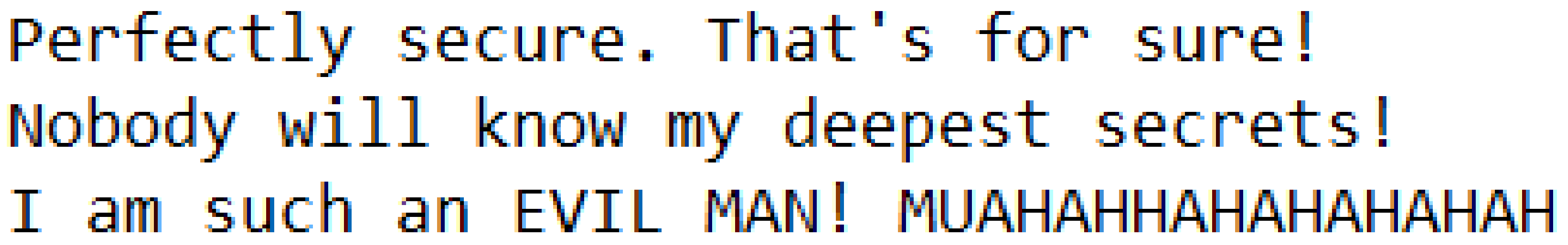
```
11    n=256
12    k0=int(input("Insert shared-secret (k0): "))
13    k0=k0%n
```

How can you exploit this small module?

Answer: A and C are now bruteforcable... But in what range?

## EVIL_MAN

## Evil_Man



```
Perfectly secure. That's for sure!
Nobody will know my deepest secrets!
I am such an EVIL MAN! MUAHAHHAHAHAHAHAH
```

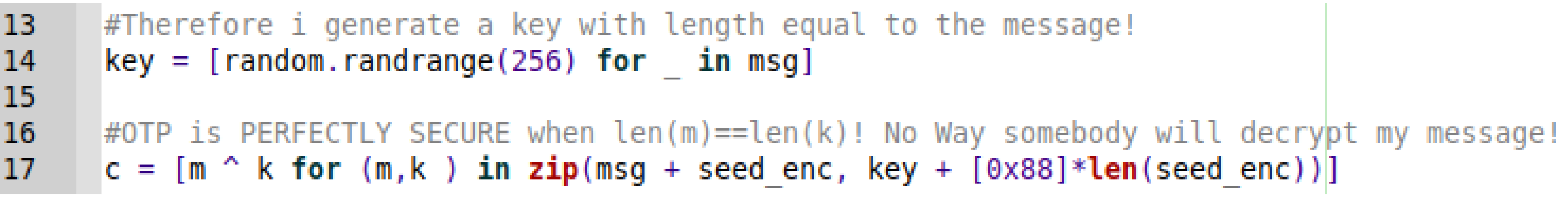
## Evil_Man

Analyze the source code and spot the algorithm vulnerabilities.
With the gained intel, write a script that can decrypt the evil secrets of the evil man!

30 minutes

# Hint#1: seeding correctly

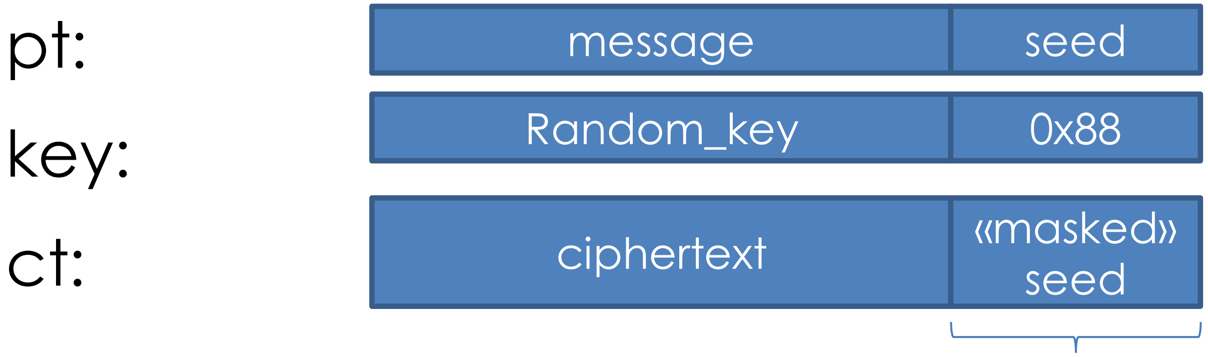
MersenneTwister generator is a really good one and is impossible to predict the generated numbers… unless you have the seed!



```
#Therefore i generate a key with length equal to the message!
key = [random.randrange(256) for _ in msg]

#OTP is PERFECTLY SECURE when len(m)==len(k)! No Way somebody will decrypt my message!
c = [m ^ k for (m,k ) in zip(msg + seed_enc, key + [0x88]*len(seed_enc))]
```

Here's the graphic representation:



How many bytes?