Sigurnost računala i podataka (Lab 2)

Symmetric key cryptography - a crypto challenge

Zadatak: U sklopu vježbe student će riješiti odgovarajući *crypto* izazov, odnosno dešifrirati odgovarajući *ciphertext* u kontekstu simetrične kriptografije. Izazov počiva na činjenici da student nema pristup enkripcijskom ključu.

Teorija

Simetrična enkripcija-univerzalna tehnika za pružanje povjerljivosti u komunikaciji i pohrani

OSNOVNI ELEMENTI

- plaintext P
- encryption algorithm E
- secret symmetric key K
- ciphertext C
- decryption algorithm D

DVIJE VRSTE NAPADA

1. kriptoanaliza
2. brute-force - naš zadatak

Uvod

Za pripremu *crypto* izazova, odnosno enkripciju korištena je Python biblioteka `cryptography`. *Plaintext* koji student treba otkriti enkriptiran je korištenjem *high-level* sustava za simetričnu enkripciju iz navedene biblioteke - Fernet.

Crypto challenge

Koristimo virtualno okruženj te instaliramo biblioteku cryptography naredbom: `pip install cryptography`

Pratimo upute (https://cryptography.io/en/latest/fernet/)

```
from cryptography.fernet import Fernet
key = Fernet.generate_key()
```

> key = Fernet.generate_key()
> key
> b'2BKp9GNLIwRpqVYD-CbxciA4ksJdCxgdIf_y5ilEAdw='

```
f = Fernet(key)
```

```
f = Fernet(key)
f
<cryptography.fernet.Fernet object at 0x000002418348D4C0>
```

`f.encrypt(b"my deep dark secret")`

```
f.encrypt(b"my deep dark secret")
b'gAAAAABhdpYb6ELnfsd1JQaRncAsDXgrabNGik4SjqF6aXqYH_VSJJtXENPIamQ7A531Smvhv
```

```
ciphertext=f.encrypt(b"my deep dark secret")
ciphertext
b'gAAAAABhdpZdraSbykublcnCjtMkUR4U6BhG5q7uYM8FWsGG2zCU27lJOOPhe1d98ToJv3-
lYGncT877fqlIoHhoIZurgVdnhd5DL9ONkmHGNJc4YZGB8ew='
```

`f.decrypt(ciphertext)`

```
f.decrypt(ciphertext)
b'my deep dark secret'
```

Prvo smo ekriptirali plaintext (b'my deep dark secret) pa ga onda dekriptirali.

Sada je zadatak otkriti koji file s a507 local serverae je moj. Napišemo kod u VS.

```
from cryptography.hazmat.primitives import hashes

def hash(input):
    if not isinstance(input, bytes):
        input = input.encode()

    digest = hashes.Hash(hashes.SHA256())
    digest.update(input)
    hash = digest.finalize()

    return hash.hex()


print(hash('zupanovic_karmen'))
```

Dobije **naziv**:

Directory of C:\Users\A507\KarmenZ

10/25/2021 01:52 PM <DIR> .
10/25/2021 01:52 PM <DIR> ..
10/25/2021 01:52 PM 12,280 **3f7699d1bc4ee53a3e8f24bfd2577a150260f938f45b8d6a538819129263bd13.encrypted**
10/25/2021 01:20 PM <DIR> karmenz
10/25/2021 01:49 PM 304 my_izazov.py
2 File(s) 12,584 bytes
3 Dir(s) 118,552,190,976 bytes free

Sada možemo preuzeti svoj izazov.

Za enkripciju smo koristili **ključeve ograničene entropije - 22 bita**. Ključevi su generirani na sljedeći način:

```
# Encryption keys are 256 bits long and have the following format:
#
#             0...000b[1]b[2]...b[22]
#
# where b[i] is a randomly generated bit.
```

```
key = int.from_bytes(os.urandom(32), "big") & int('1'*KEY_ENTROPY, 2)

# Initialize Fernet with the given encryption key;
# Fernet expects base64 urlsafe encoded key.
key_base64 = base64.urlsafe_b64encode(key.to_bytes(32, "big"))
fernet = Fernet(key_base64)
```

Znamo plaintext i ciphertext, trebamo dekriptirati naš file. Koristit ćemo brute-force pristup.

Unutar petlje prevrtimo ključeve dok ne nađemo naš.

```
import base64
from cryptography.hazmat.primitives import hashes
from cryptography.fernet import Fernet

def hash(input):
  if not isinstance(input, bytes):
    input = input.encode()
  digest = hashes.Hash(hashes.SHA256())
  digest.update(input)
  hash = digest.finalize()
  return hash.hex()

def test_png(header):
  if header.startswith(b"\211PNG\r\n\032\n"):
    return True

def brute_force():
# Reading from a file
  filename = "f7699d1bc4ee53a3e8f24bfd2577a150260f938f45b8d6a538819129263bd13.encrypted"
  with open(filename, "rb") as file:
    ciphertext = file.read()
# Now do something with the ciphertext
  ctr = 0
  while True:
    key_bytes = ctr.to_bytes(32, "big")
    key = base64.urlsafe_b64encode(key_bytes)
    if not (ctr + 1) % 1000:
      print(f"[*] Keys tested: {ctr +1:,}", end = "\r")
    try:
      plaintext = Fernet(key).decrypt(ciphertext)
      header = plaintext[:32]
      if test_png(header):
        print(f"[+] KEY FOUND: {key}")
        # Writing to a file
        with open("BINGO.png", "wb") as file:
          file.write(plaintext)
            break
    except Exception:
      pass
# Now initialize the Fernet system with the given key
# and try to decrypt your challenge.
# Think, how do you know that the key tested is the correct key
# (i.e., how do you break out of this infinite loop)?
    ctr += 1
if __**name__** == "__**main__**":
  brute_force()
```

Napomena: Nama na satu nije radilo pa sam popričala s kolegama iz drugih grupa i kopirala njihov kod.