Sigurnost računala i podataka

Lab 3: Symmetric key cryptography (a crypto challenge)

Prvo smo morali preuzeti osobni izazov sa servera, odnosno pronaći vlastitu datoteku. Imena datoteka su bile hash vrijednosti SHA-256 funkcije koja je kao argument uzimala ime i prezime studenta u formatu prezime_ime.

```
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()

filename = hash('prezime_ime') + ".encrypted"
```

Pomoću ovog isječka koda smo dobili ime vlastite datoteke tako što smo u posljednjoj liniji umjesto ‘prezime_ime’ upisali svoje prezime i ime u odgovarajućem formatu. Nakon što smo preuzeli odgovarajuću datoteku počeli smo s dekripcijom.

Za enkripciju datoteka korišteni su ključevi generirani pomoću librarya Fernet. Ključevi su bili ograničene entropije - 22 bita (zato da bi ih bilo moguće pogoditi brute force pristupom u razumnom vremenu).

Znali smo da je datoteka slika u png formatu pa smo zaključili da plaintext (datoteka u binarnom obliku) započinje sa “\211PNG\r\n\032\n”, prvih 8 byteova karakterističnih za png format.

Tako smo znali plaintext i cyphertext, pa smo samo morali otkriti ključ. Odlučili smo ga otkriti brute-force napadom.

Postavili smo counter ctr = 0 (odnosno počeli smo od ključa 0), te smo za svaki ključ provjeravali je li odgovarajući, odnosno je li file koji dobijemo dekripcijom s tim ključem png formata. Dekripciju smo izvršavali pomoću funkcije iz Fernet librarya:

plaintext = Fernet(key).decrypt(ciphertext)

Za provjeru formata smo koristili jednostavno implementiranu funkciju test_png. U slučaju da ključ nije ispravan, povećali bi counter za 1 i pokušali ponovo.

Cijeli kod:

```
from cryptography.hazmat.primitives import hashes
from os import path
from cryptography.fernet import Fernet, InvalidToken
import base64


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
    return False


def brute_force_attack(ciphertext):
    ctr = 0
    while True:
        key_bytes = ctr.to_bytes(32, "big")
        key = base64.urlsafe_b64encode(key_bytes)

        try:
            plaintext = Fernet(key).decrypt(ciphertext)

            header = plaintext[:32]
            if test_png(header):
                print(f"KEY FOUND: {key}")
                with open("BINGO.png", "wb") as file:
                    file.write(plaintext)
                break
        except InvalidToken:
```

```
            pass

        ctr += 1
        if not ctr % 1000:
            print(f"[*] Keys tested: {ctr:,}", end="\r")


if __name__ == "__main__":
    filename = hash("biuk_ivan") + ".encrypted"
    print(filename)

    if not path.exists(filename):
        with open(filename, "wb") as file:
            file.write(b"")

with open(filename, "rb") as file:
    encrypted_challenge = file.read()

brute_force_attack(encrypted_challenge)
```

Dekriptirana datoteka: