**Sigurnost računala i podataka**

Vježba 3: Symmetric key cryptography (a crypto challenge)

Preuzeli smo osobni izazov sa servera, odnosno pronalazak vlastite datoteke. Imena datoteka su hash vrijednosti SHA-256 funkcije kojoj je argument 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 koda smo dobili ime vlastite datoteke tako što smo u posljednjoj liniji umjesto ‘prezime\_ime’ upisali svoje prezime i ime. Preuzeli smo odgovarajuću datoteku i počeli s dekripcijom.

Ključevi za enkripciju datoteka su bili ograničene entropije, 22 bita. Datoteka je slika u png formatu pa smo zaključili da plaintext započinje sa “\211PNG\r\n\032\n”, prvih 8 byteova karakterističnih za png format.

Znali smo plaintext i cyphertext, pa smo ključ odlučili otkriti brute-force napadom.

Kod koji smo koristili:

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("munivrana\_luka") + ".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)

Dekriptiranu datoteku koju smo dobili:

