

IZVJEŠTAJ 3. LABORATORIJSKE VJEŽBE

Na 3. laboratorijskim vježbama, naš izazov se ticao teme simetrične kriptografije. Naš prvi zadatak je bio od svih izazova pronaći naš izazov tako što smo *hashirali* naše ime i prezime te to još i dodali *string* `'encrypted'` kako bismo saznali koji od ponuđenih izazova je bio baš naš. Ovo smo morali raditi zato što ne postoji neka funkcija kojom bismo saznali iz *hash valuea* što je zapravo *hashirano* (*one-way property*).

Nakon toga, prelazimo na naš izazov, treba saznati što je enkriptirano u našem samom izazovu. Naime, brute forceom napadom mi smo mogli saznati što se nalazi unutar samog izazova ako znamo nešto o tome što je enkriptirano ili na koji je način to enkriptirano. Znajući da se radi o slici u `.png` formatu, mogli smo prilagoditi naš algoritam za pronalaženje ključa kojim ćemo dekriptirati naš izazov. Pošto sada to znamo, možemo *exploitati* 'slabosti' samog `.png` formata. Na taj način i pravi hackeri prilagođavaju svoj algoritam na temelju patterna svoje žrtve.

Kod `.png` formata znamo da on počinje sa *8-bytenim* potpisom koji glasi: 137 80 78 71 13 10 26 10 (u decimalnom formatu). Kada smo saznali, prilagodili smo naš brute force algoritam te smo uspješno saznali našu skrivenu poruku.

OBJAŠNJENJE KODA

1. Saznajemo naš izazov

```
if __name__ == "__main__":  
    filename = hash('juric-pesic_mijo') + ".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)
```

Ovdje hashiramo naše ime i prezime te mu dodajemo string '.encrypted' te saznajemo ime našeg filea. Zatim kad je file otvoren, sve to spremamo u varijablu encrypted_challenge nad kojim ćemo vršiti brute force napad.

2. *Exploit* .png formata

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

Ovdje provjeravamo header filea tj. da li file počinje s stringom : "\211PNG\r\n\032\n" te ako počinje sa tim, znamo da radimo sa pravim fileom i pravim formatom.

3. Brute force

```
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"[*] Key Tested: {ctr}", end="\r")
```

Testiramo od ključa 0 (varijabla crt) te provjeravamo kada dekriptiramo da li dobijamo sliku u png formatu. Kada pronađemo odgovarajući ključ, napravimo file BINGO.png u koji spremamo dekriptirani izazov koji je slika s našim imenom i prezimenom.

4. Dekriptirani izazov



Congratulations Juric-Pesic Mijo!

You made it!