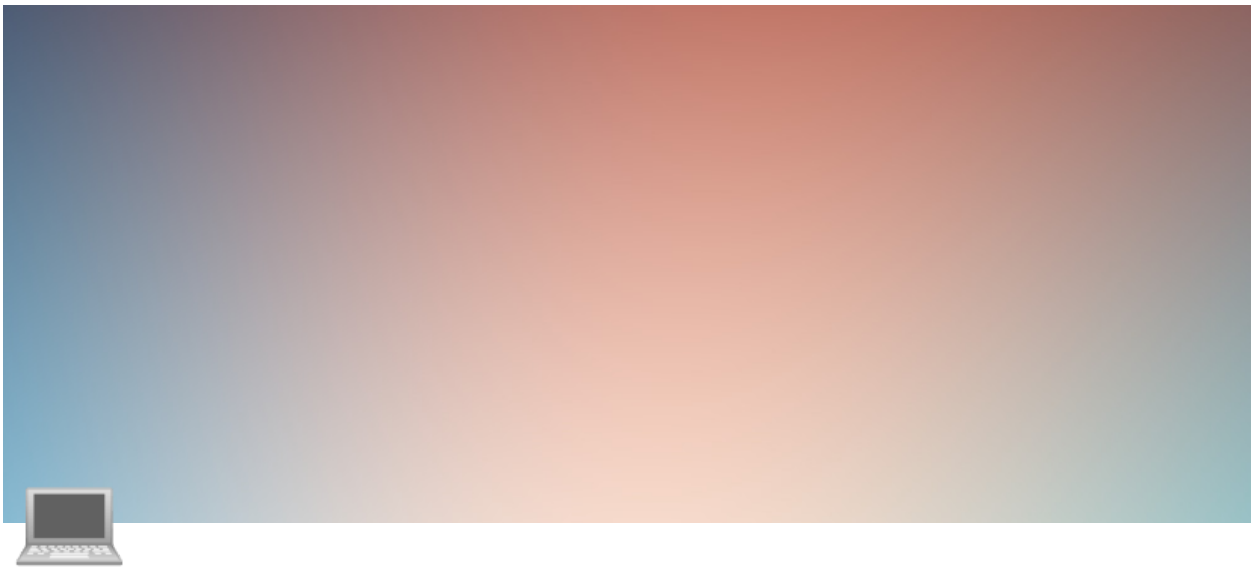

Izvještaj treće laboratorijske vježbe

> Cilj vježbe je primjeniti teoreteske spoznaje o osnovnim kritografskim mehanizmima za autentikaciju i zaštitu integriteta poruka u praktičnom primjerima. Pri tome ćemo koristiti simetrične i asimetrične krito mehanizme: *message authentication code (MAC)* i *digitalne potpise* zasnovane na javnim ključevima.

▼ Cilj je implementacija zaštite integriteta sadržaja dane poruke primjenom odgovarajućeg MAC algoritma. Koristili smo HMAC mehanizam iz Python biblioteke cryptography.

▼ Kreirali smo tekstualnu datoteku imena "message.txt" i u nju pohranili sadržaj čiji integritet smo željeli zaštititi.

▼ Na temelju pročitanog sadržaja iz datoteke i proizvoljno unesenog ključa generirali smo MAC pomoću funkcije "generate_MAC"

```
from cryptography.hazmat.primitives import hashes, hmac
from cryptography.exceptions import InvalidSignature

def generate_MAC(key, message):
    if not isinstance(message, bytes):
        message = message.encode()

    h = hmac.HMAC(key, hashes.SHA256())
    h.update(message)
    signature = h.finalize()
    return signature

def verify_MAC(key, signature, message):
    if not isinstance(message, bytes):
        message = message.encode()

    h = hmac.HMAC(key, hashes.SHA256())
    h.update(message)
    try:
        h.verify(signature)
    except InvalidSignature:
        return False
    else:
        return True
```

▼ Ako promijenimo sadržaj datoteke, MAC algoritam će uspiješno detektirati takve promjene i umjesto "true" ,funkcija će vraćati "false"