"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them."

Co-funded by the European Union

Internet i računarska sigurnost

Amel Toroman, Ermin Bajramović

UNBI

Co-funded by the European Union

Internet i računarska sigurnost

UVOD

Internet i računarska sigurnost su ključni faktori za sigurnost digitalnog svijeta u kojem živimo.

Kako tehnologija napreduje, tako raste i potreba za adekvatnom zaštitom računarskih sistema, podataka i korisničkih informacija na internetu.

U ovom kursu, kroz detaljnu teoretsku osnovu i praktične primjere, obradit će se ključne teme poput zaštite računara na internetu, sigurnosnih prijetnji, metoda zaštite, te kriptografije, koja je osnova sigurnosti podataka i komunikacija

Internet i računarska sigurnost

UVOD

Posebnu pažnju posvetit će se primjeni Pythona u implementaciji zaštitnih metoda, kao i analizi i implementaciji različitih kriptosistema, kao što su simetrična i asimetrična kriptografija, hash funkcije i generatori slučajnih brojeva.

Kurs je osmišljen kako bi polaznicima omogućio razumijevanje teorijskih osnova i praktičnih vještina potrebnih za upravljanje sigurnošću u digitalnom okruženju.

Internet i računarska sigurnost

INTERNET

Internet je globalna mreža računara koja omogućava međusobnu komunikaciju i razmjenu podataka koristeći različite tehnologije i protokole.

Nastao je kao rezultat razvoja računarskih mreža i danas predstavlja osnovu modernog digitalnog svijeta. Milijarde uređaja su povezane putem interneta, a njegova uloga je prisutna u gotovo svim sferama ljudskog života – od obrazovanja i poslovanja do zabave i društvenih interakcija.

Internet i računarska sigurnost

INTERNET

Internet funkcioniše na osnovu niza pravila i standardizovanih protokola koji omogućavaju komunikaciju između računara. Ključni elementi interneta uključuju:

- IP adrese (eng. Internet Protocol Address),
- DNS (eng. Domain Name System),
- Protokoli za prijenos podataka,
- Servisi na internetu.

Međutim, upotreba interneta nosi i određene sigurnosne rizike. Zbog toga je važno razumjeti metode zaštite koje omogućavaju sigurno korištenje interneta.

Co-funded by the European Union

Internet and computer security

Načini zaštite računara i korisnika na internetu

Malveri (Malware) - virusi, trojanci, ransomware su zlonamjerni softveri dizajnirani s ciljem narušavanja sigurnosti korisnika.

Najčešće vrste malvera su:

- **Virusi** - Računarski programi koji se prikače na druge legitimne programe i izvršavaju štetne radnje.

- **Trojanci** - Maliciozni softveri koji se prikrivaju kao korisni programi, ali zapravo omogućavaju hakerima pristup sistemu.

- **Ransomware** – Malver koji šifrira podatke na računaru i zahtijeva otkupninu za njihovo vraćanje.

Co-funded by the European Union

Internet i računarska sigurnost

Načini zaštite računara i korisnika na internetu

Phishing i socijalni inženjering

Phishing napadi se zasnivaju na prevari korisnika pomoću lažnih e-mailova ili web stranica koje oponašaju legitimne servise. Cilj je navesti korisnika da unese svoje povjerljive podatke, kao što su lozinke i brojevi kreditnih kartica.

Napadi na mreže (DDoS, MITM):

- **DDoS napadi** – Napadači koriste mreže zaraženih računara (botnet) kako bi preopteretili servere velikim brojem zahtjeva i učinili ih nedostupnim.
- **MITM (eng. Man-in-the-Middle) napadi** – Napadači presreću komunikaciju između korisnika i web servisa, čime mogu ukrasti podatke ili manipulisati informacijama.

Internet i računarska sigurnost

Metode zaštite na internetu

Antivirusni softver i vatrozid

Antivirusni programi su ključna zaštita od malvera, dok vatrozid filtrira mrežni saobraćaj i sprečava neovlaštene pristupe.

Primjer: Windows Defender aktivacija

```
Settings → Update & Security → Windows Security → Virus & threat protection
```

Primjer: Gmail 2FA aktivacija

```
Google Account Settings → Security → 2-Step Verification
```

Co-funded by the European Union

Internet i računarska sigurnost

Metode zaštite na internetu

Enkripcija podataka i VPN

VPN servisi pružaju dodatnu sigurnost šifriranjem internet saobraćaja i sakrivanjem IP adrese korisnika.

Primjer: Instalacija Windscribe VPN-a

https://windscribe.com/

Co-funded by the European Union

Internet i računarska sigurnost

Primjer zaštite na internetu pomoću Pythona

Provjera sigurnosti web stranice pomoću Python skripte

Ova skripta koristi VirusTotal API za provjeru da li je web stranica maliciozna:

```
import requests

API_KEY = "YOUR_VIRUSTOTAL_API_KEY"
url_to_check = "http://example.com"

def check_url_safety(url):
    params = {"apikey": API_KEY, "resource": url}
    response = requests.get("https://www.virustotal.com/vtapi/v2/url/report",
params=params)

    if response.status_code == 200:
        result = response.json()
        if result["positives"] > 0:
            print(f"Upozorenje: {url} je označen kao nesiguran!")
        else:
            print(f"{url} je siguran za posjetu.")
    else:
        print("Greška pri provjeri URL-a.")

check_url_safety(url_to_check)
```

Internet i računarska sigurnost

Primjer zaštite na internetu pomoću Pythona

Pokretanje Python skripte

Ova skripta šalje URL na VirusTotal i provjerava da li je označen kao maliciozan.

Prvobitno je potrebno prijaviti se na stranicu VirusTotal, te kopirati API za provjeru i zamijeniti ga u kôdu sa YOUR_VIRUSTOTAL_API_KEY.
Nakon toga, za testiranje je potrebno postaviti URL koji se želi provjeriti na mjesto url_to_check = "http://example.com".

Sljedeći korak je otvaranje terminala i pokretanje fajla preko naredbe:

```
python NameFile.py
```

Internet i računarska sigurnost

Primjer zaštite na internetu pomoću Pythona

Ako je stranica prijavljena kao zlonamjerna dobije se sljedeća poruka koja je prikazana na Slici.

```
C:\Users\AzraKT\Desktop>python zastita.py
Warning: https://web.tfb.unbi.ba has been marked as unsafe!
```

Ako stranica nije prijavljena kao zlonamjerna dobije se sljedeća poruka koja je prikazana na Slici.

```
C:\Users\AzraKT\Desktop>python zastita.py
https://www.bhtelecom.ba is safe to visit.
```

Co-funded by the European Union

Internet i računarska sigurnost

RAČUNARSKA SIGURNOST I ZAŠTITA PODATAKA

Razvojem matematičkih nauka, tehnologije i računarstva, došlo je do pojave složenih algoritama koji su sadržavali složene matematičke proračune.

Kriptovanje, ograničavanje pristupa i zaštita dokumenata iza tzv. vatrozida (eng. firewall) neke su od uobičajenih tehnika zaštite osjetljivih informacija.

Uz takve metode koristi se i digitalno potpisivanje dokumenata i digitalni vodeni žigovi (watermarks).

U suštini napadi se mogu definisati kao akcije koje su usmjerene na ugrožavanje sigurnosti informacija, računarskih sistema i mreža.

Internet i računarska sigurnost

RAČUNARSKA SIGURNOST I ZAŠTITA PODATAKA

Kada se govori o konkretnim primjerima veliku opasnost predstavljaju zlonamjerni programi, u koje spadaju:

1) virusi,
2) crvi,
3) trojanski konji,
4) špijunski i reklamni programi,
5) alati za dobijanje administrativnih ovlasti na sistemu (eng. rootkit) itd.

Neke od značajnijih softverskih metoda zaštite su:

1) kriptovanje,
2) upotreba vodenih žigova, šifri, kriptografskih sažetaka,
3) korištenje digitalnih potpisa.

Co-funded by the European Union

Internet i računarska sigurnost

KRIPTOGRAFIJA

Kriptografija je naučna disciplina koja se bavi proučavanjem metoda za slanje poruka u takvom obliku da ih samo onaj kome su namijenjene može pročitati.

Djelotvorna komunikacija je bila vrlo bitan faktor kraljevima, kraljicama i vojskovođama stoljećima, zbog svjesnosti o posljedicama koje bi mogle rezultirati ukoliko njihove poruke dođu u krive (suparničke) ruke.

Na taj način bi se otkrile zabranjene i vrlo bitne, te dragocjene tajne.

Ova opasnost je potaknula razvoj šifri i kôdova koje se mogu definirati kao sredstva kojima se postiže da poruku može pročitati samo osoba kojoj je namijenjena.

Co-funded by the European Union

Internet i računarska sigurnost

Osnovni pojmovi u kriptografiji

Kriptografija je naučna disciplina koja se bavi proučavanjem metoda za slanje poruka u oblicima čitljivim samo onima kojima su i namijenjene.

Sama riječ kriptografija je grčkog porijekla i mogla bi se doslovno prevesti kao tajnopis.

Cilj kriptografije je omogućiti nesmetano komuniciranje pošiljaoca i primaoca poruke preko nesigurnog komunikacijskog kanala tako da treća osoba ne može razumijeti njihove poruke.

Co-funded by the European Union

Internet i računarska sigurnost

Osnovni pojmovi u kriptografiji

Kriptografski algoritam ili šifra je matematička funkcija koja se koristi za šifriranje i dešifriranje.

Općenito, radi se o dvije funkcije, jednoj za šifriranje, a drugoj za dešifriranje.
Te funkcije preslikavaju osnovne elemente otvorenog teksta (najčešće su to slova, bitovi, grupe slova ili bitova) u osnovne elemente šifrata i obratno.

Funkcije se biraju iz određene familije funkcija u ovisnosti o ključu. Skup svih mogućih vrijednosti ključeva naziva se prostor ključeva

Co-funded by the European Union

Internet i računarska sigurnost

Osnovni pojmovi u kriptografiji

Enkripcija je postupak koji se primjenjuje na strani pošiljatelja. Njome se otvoreni tekst pretvara u šifrirani ili kodirani tekst pomoću odgovarajućeg ključa.

Zbog toga se enkripcija dijeli na šifriranje i kodiranje. Kod šifriranja, bazična jedinica je jedno slovo, par slova (bigram), a ponekad i više slova (poligram), a kod kodiranja, bazična jedinica je riječ ili skup riječi. Zbog toga, ne postoji neka oštra teorijska granica izmedu kôdova i šifri.

Šifriranje je postupak koji se dijeli na transpoziciju (svako slovo zadržava svoj identitet, ali mijenja mjesto) i supstituciju (svako slovo zadržava svoje mjesto, ali mijenja identitet).

Internet i računarska sigurnost

Osnovni pojmovi u kriptografiji

Dekripcija je postupak kojim se šifrirani ili kodirani tekst pretvara u otvoreni tekst pomoću unaprijed poznatog ključa i algoritma.

Primjenjuje se na strani primatelja poruke. Dakle, ovaj postupak je legalan, jer primatelj komunicira sa pošiljateljem.

Samim postupkom nastaje dekriptat.

Co-funded by the European Union

Internet i računarska sigurnost

Osnovni pojmovi u kriptografiji

Kriptosistem se sastoji od kriptografskog algoritma ili šifre, te svih mogućih otvorenih tekstova, šifrata i ključeva. Kriteriji za klasifikaciju kriptosistema:

1) **Tip operacija koji se koristi pri šifriranju:**
   a) Supstitucijske šifre
   b) Transpozicijske šifre

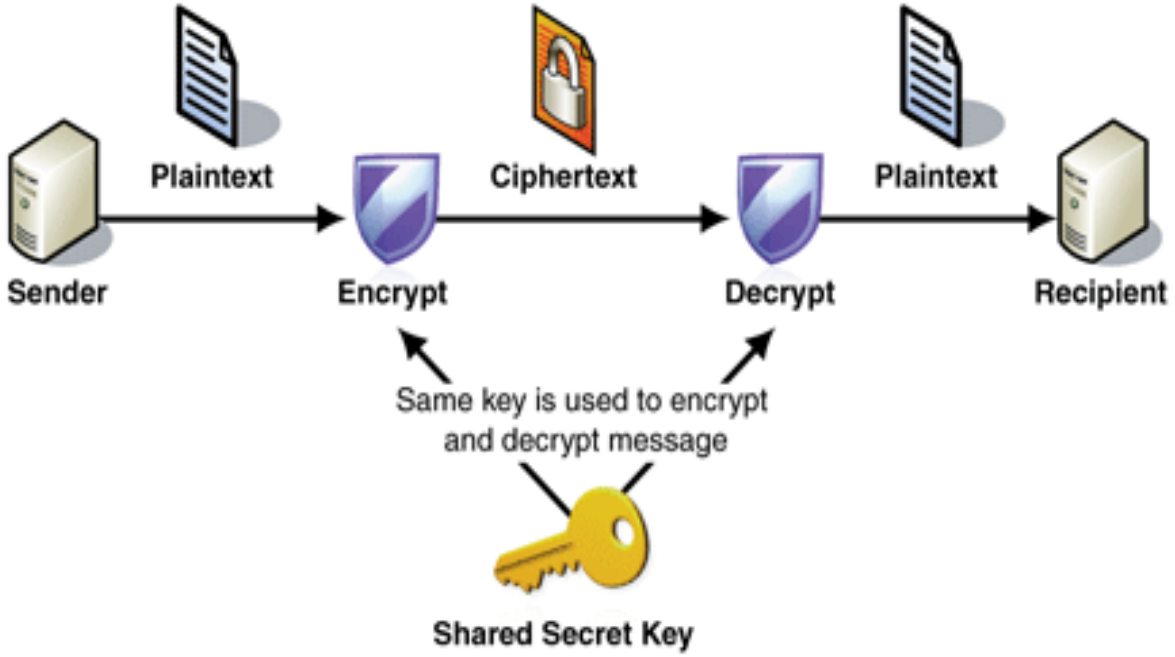
2) **Način na koji se obrađuje otvoreni tekst:**
   a) blokovne šifre obrađuje se jedan po jedan blok elemenata otvorenog teksta koristeći isti ključ
   b) protočne šifre elementi otvorenog teksta obrađuju se jedan po jedan koristeći pri tome niz ključeva engl. Keystream koji se paralelno generira

3) **Tajnost i javnost ključeva**
   a) Kriptosistemi s tajnim ključem
   b) Kriptosistemi sa javnim ključem

Internet i računarska sigurnost

Kriptosistemi sa tajnim ključem

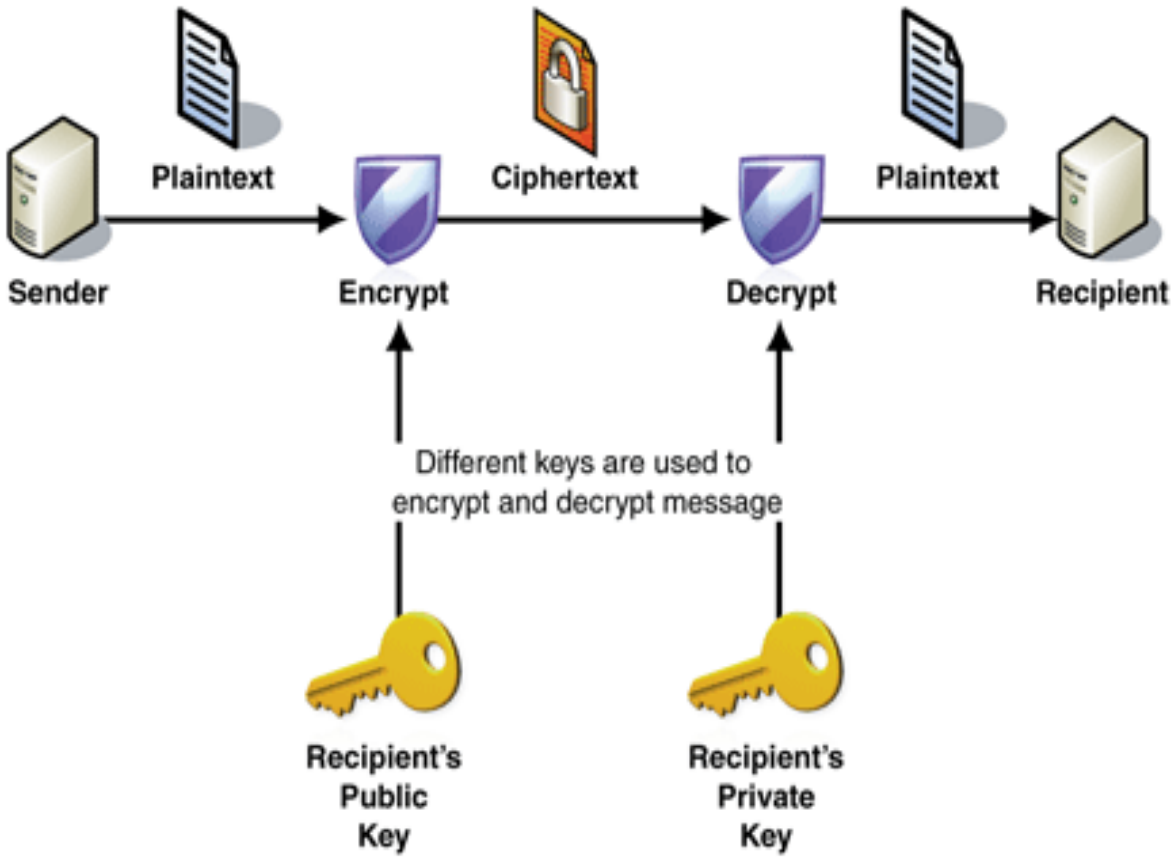
Kod simetričnih, konvencionalnih kriptosistema tj. kriptosistema sa tajnim ključem, pošiljatelj i primatelj trajno izabiru ključ K pomoću kojeg generiraju funkcije za šifriranje i dešifriranje. U ovom slučaju su funkcije iste te su lake za „provaljivanje“.



Internet i računarska sigurnost

Kriptosistemi sa javnim ključem

Kod kriptosistema sa javnim ključem, odnosno asimetričnih kriptosistema je nemoguće, u nekom razumnom vremenu, odrediti ključ za dešifriranje (uprkos tome što je ključ za šifriranje poznat) koji je javni ključ. Naime, bilo ko može šifrirati poruku pomoću njega, ali samo osoba koja ima odgovarajući ključ za dešifriranje (privatni ili tajni ključ) može dešifrirati tu poruku.



Co-funded by the European Union

Internet i računarska sigurnost

Vrste kriptografskih algoritama

Kriptografski algoritmi dijele se na:

1. **Simetrične algoritme**
   isti ključ se koristi za šifriranje i dešifriranje (AES, DES, 3DES).

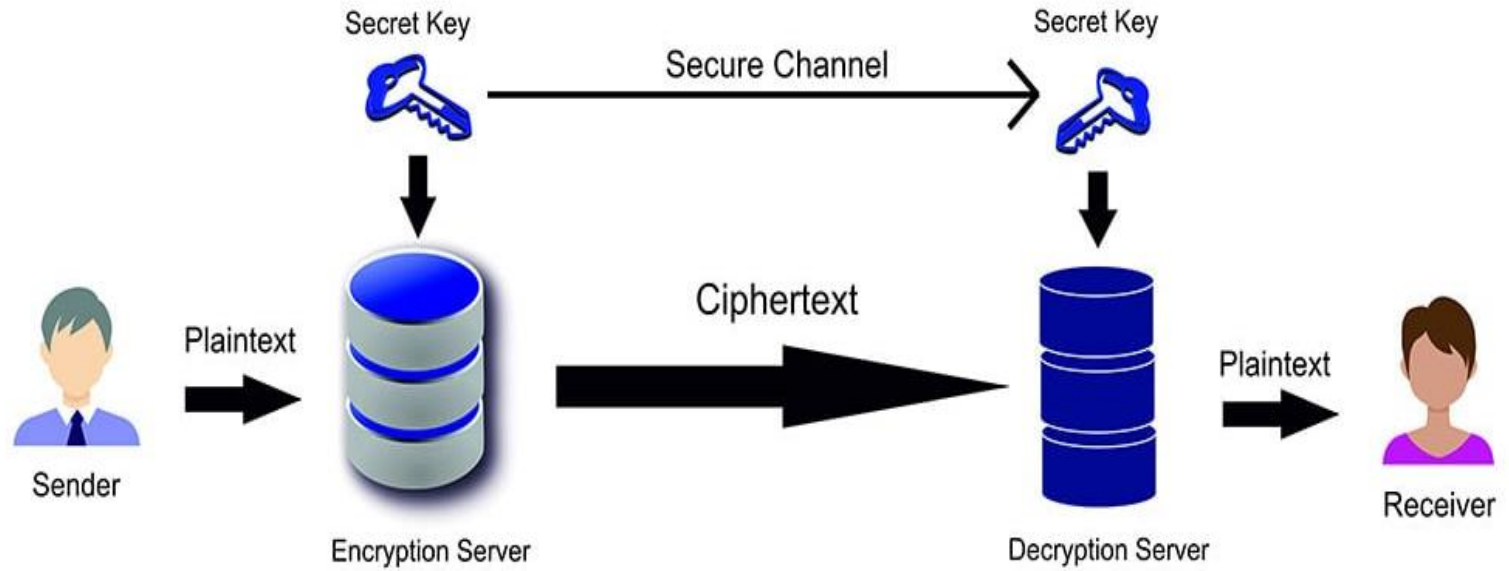
2. **Asimetrične algoritme**
   koristi se javni ključ za šifriranje i privatni ključ za dešifriranje (RSA, ECC).

3. **Hash funkcije**
   nepovratna matematička transformacija podataka u jedinstveni otisak (MD5,SHA-256, SHA-3).

Internet i računarska sigurnost

Simetrična kriptografija

Simetrična kriptografija je tradicionalni način zaštite podataka koji se prenose od jednog do drugog učesnika u komunikaciji. Ovaj način zaštite podataka star je gotovo koliko i ljudska komunikacija. Suština simetrične kriptografije je da je za sigurnu komunikaciju neophodno da oba učesnika imaju isti ključ koji im omogućava šifriranje i dešifriranje podataka.



Internet i računarska sigurnost

Simetrična kriptografija

Ključne karakteristike simetrične kriptografije:

- Jedan ključ – isti ključ se koristi za obje operacije.
- Brza i efikasna – posebno pogodna za šifriranje velikih količina podataka.
- Sigurnosni izazov – ključ mora biti sigurno podijeljen između pošiljaoca i primaoca.

Najpoznatiji simetrični algoritmi su:

- AES (eng. Advanced Encryption Standard)
- DES (eng. Data Encryption Standard)
- 3DES (eng. Triple DES)
- Blowfish, Twofish, RC4

Co-funded by the European Union

Internet i računarska sigurnost

PRIMJER: Matematički model simetrične kriptografije

Uz pretpostavku da postoji poruka (plaintext) P, ključ K, i enkripcijsku funkciju E.

Matematički model šifriranja izgleda ovako:

$$C = E_K(P)$$

gdje je C šifrirani tekst (ciphertext).

Za dešifriranje koristi se obrnuti proces:

$$P = D_K(C)$$

gdje je D funkcija dešifriranja koja koristi isti ključ K.

U nastavku će biti prikazan primjer AES algoritma za enkripciju i dekripciju koristeći Python.

Co-funded by the European Union

Internet i računarska sigurnost

```python
from Crypto.Cipher import AES
import base64

# Function to add padding
def pad(text):
    return text + (16 - len(text) % 16) * ' '

# The key must be 16 bytes long
key = b"1234567890123456"  # Tačno 16 bajtova


# Text to encrypt
plain_text = "Osjetljivi podaci"

# Create AES object and encrypt the text
cipher = AES.new(key, AES.MODE_ECB)
encrypted_text = cipher.encrypt(pad(plain_text).encode())

# Display encrypted text
print("Encrypted text:", base64.b64encode(encrypted_text).decode())

# Decrypt the text
decrypted_text = cipher.decrypt(encrypted_text).decode().strip()
print("Decrypted text:", decrypted_text)
```

Co-funded by
the European Union

Internet i računarska sigurnost

PRIMJER: Matematički model simetrične kriptografije

Prikaz nakon pokretanja programa (AES algoritam):

```
C:\Users\AzraKT\Desktop>python aess.py
Encrypted text: jVvk8kTGrMWJriqnxQUjHw==
Decrypted text: Sensitive data
```

Internet i računarska sigurnost

Asimetrična kriptografija

Računarske mreže omogućavaju brzu razmjenu podataka, ali u svojim počecima nisu pravljene da budu veoma sigurne.

Umjesto jednog istog ključa za šifriranje i dešifriranje predloženo je postojanje para ključeva: javnog i privatnog.

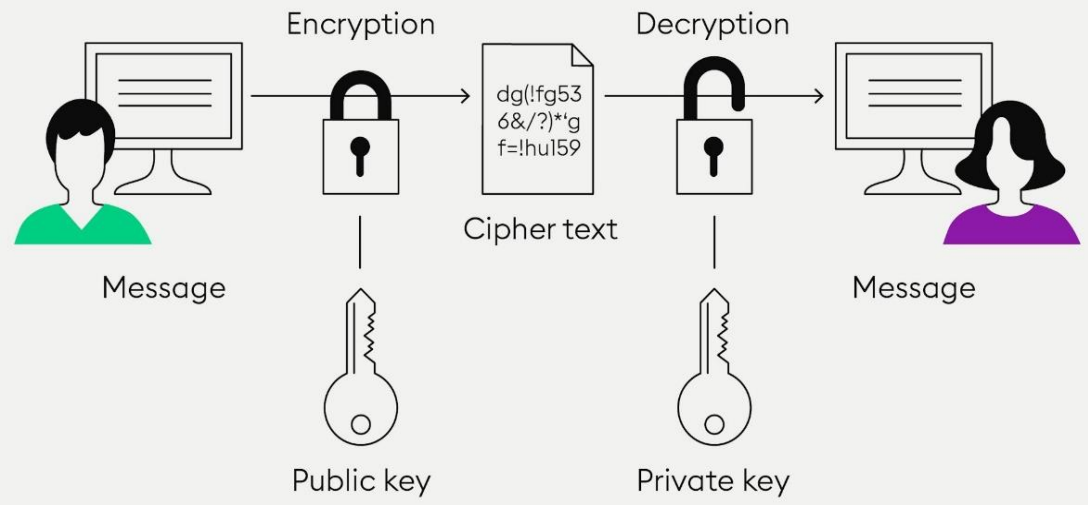
Javni ključ (KJ) je dostupan svima i to je ključ koji se koristi za šifriranje podataka.

Odgovarajući privatni ključ (KP) je tajan i samo taj ključ omogućava dešifriranje podataka šifriranih javnim ključem iz para.

Svaki subjekat koji želi sigurnu komunikaciju može objaviti svoj javni ključ i svako ko mu želi poslati šifriranu poruku koristi taj ključ za šifriranje. Pošto jedino ovaj subjekt ima privatni ključ koji može dešifrirati podatke obezbjeđena je povjerljivost podataka.

Internet i računarska sigurnost

Asimetrična kriptografija



Asimetrična kriptografija koristi dva ključa: javni ključ i tajni ključ.

Razlikuje se od simetrične kriptografije, koja koristi isti ključ za šifriranje i dešifriranje podataka.

- **Javni ključ** se koristi za šifriranje podataka i može se slobodno dijeliti s drugima.
- **Tajni ključ** se koristi za dešifriranje podataka i mora ostati privatno i sigurno.

Co-funded by the European Union

Internet i računarska sigurnost

RSA algoritam

RSA (eng. Rivest-Shamir-Adleman) je jedan od najpoznatijih asimetričnih kriptografskih algoritama. Temelji se na teoriji brojeva i koristi se za šifriranje i digitalne potpise.

Javni ključ je sastavljen od brojeva e i n, gdje je e eksponent i n produkt dvaju velikih prostih brojeva.

Tajni ključ je broj d, koji je veza između e i n, i koji se koristi za dešifriranje podataka.
RSA algoritam uključuje tri glavna koraka:

1. Generisanje ključeva
2. Šifriranje
3. Dešifriranje

Internet i računarska sigurnost

PRIMJER 1: Implementacija RSA algoritma u Pythonu

```python
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
from base64 import b64encode, b64decode

# 1. Generating RSA keys
def generate_keys():
    key = RSA.generate(2048)  # Generate a 2048-bit key
    private_key = key.export_key()
    public_key = key.publickey().export_key()

    # Saving the keys to files
    with open("private.pem", "wb") as f:
        f.write(private_key)

    with open("public.pem", "wb") as f:
        f.write(public_key)

    print("Keys have been successfully generated and saved.")

# 2. Encrypting a message
def encrypt_message(plain_text):
    # Load the public key
    with open("public.pem", "rb") as f:
        public_key = RSA.import_key(f.read())

    # Create a PKCS1_OAEP encryption object
    cipher = PKCS1_OAEP.new(public_key)
```

Internet i računarska sigurnost

PRIMJER 1: Implementacija RSA algoritma u Pythonu

```
# Encrypt the message
    encrypted_message = cipher.encrypt(plain_text.encode())

    # Show the encrypted message in Base64 format
    encrypted_message_b64 = b64encode(encrypted_message).decode()
    print("Encrypted message:", encrypted_message_b64)

    return encrypted_message_b64

# 3. Decrypting the message
def decrypt_message(encrypted_message_b64):
    # Load the private key
    with open("private.pem", "rb") as f:
        private_key = RSA.import_key(f.read())

    # Create a PKCS1_OAEP decryption object
    cipher = PKCS1_OAEP.new(private_key)

    # Convert the Base64 encoded encrypted message back to binary format
    encrypted_message = b64decode(encrypted_message_b64)
```

```
    # Decrypt the message
    decrypted_message = cipher.decrypt(encrypted_message).decode()
    print("Decrypted message:", decrypted_message)

# Main program
def main():
    print("Generating keys...")
    generate_keys()  # Generate the keys

    # Encrypting the message
    message = "This is a secret message"
    print("\nEncrypting the message...")
    encrypted_message = encrypt_message(message)

    # Decrypting the message
    print("\nDecrypting the message...")
    decrypt_message(encrypted_message)

if __name__ == "__main__":
    main()
```

Internet i računarska sigurnost

Prikaz nakon pokretanja programa (RSA algoritam):

```
C:\Users\AzraKT\Desktop>python rsaa.py
Generating keys...
Keys have been successfully generated and saved.

Encrypting the message...
Encrypted message: Z/SJI3pFAmzOiPOH1iFF4nXwkOg5E9CxOAC/rLD6RpwmBQp/nr9YSuNCEDTXQWtXSg
7XhUqRVBYpFFKX/kcVC7tn3+clhOUEE7WnGNrrGrZJmtRZaAoZkf1/5uegGZuKFVgAw/N9pTn01/oQtqsVEUa
/M/8Uw9+lMbUXeRIJ5IbDdx9IOh6jIHjFJOCzFfITz6t5jhKR+Nm1iqzt6glmQH3uKJKCLoKdJn242xt/v/aQ
C99vKUiNY84pNXjAJa/Vst2kBlisbH0Jv5AeIsfMthCpeW7A13Apd4eSA16MbBizLZQZJI6iZ7TABMK3htAyJ
EN9XAULmCv5ai9qc8UEcA==

Decrypting the message...
Decrypted message: This is a secret message
```

Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

Primjer upotrebe RSA algoritma (Java):

- unos teksta koji se želi šifrirati,
- kreiranje javnih i privatnih ključeva,
  - javni ključ prikazuje se unutar samog interfejsa programa,
  - privatni ključ zapisuje se u datoteku PK.txt.
- unešeni tekst se enkriptuje (šifrira) te se šifrirana poruka ispisuje u unutar interfejsa,
- unutar programa je predviđena i dekripcija dobivene šifrirane poruke,
- ispis na interfejsu dobivenog rezultata.

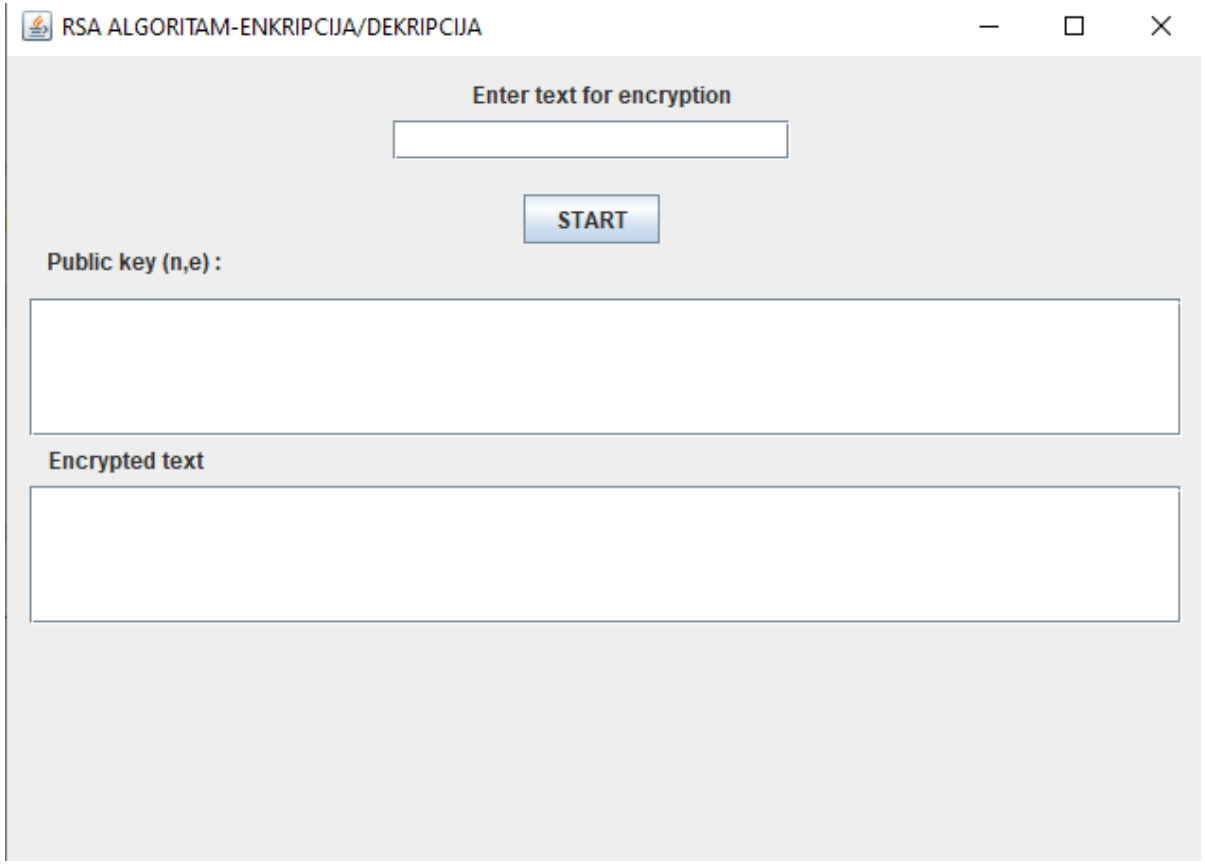
Za kreiranje programa korišten je *Apache NetBeans IDE* kao i *Java JDK*.

Za kreiranje samog UI-a korišten je *Swing GUI Forms*, a za realizaciju programske logike korištena je *Java*.

Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

Prikaz interfejsa programa RSA Algoritam Test:



Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

Prikaz funkcije RSA Algoritam Test:

```
private void button1ActionPerformed(java.awt.event.ActionEvent evt)
        RSA rsa = new RSA(1024);
        String tekst = this.unosTeksta.getText();
        /* Displaying the public key */
        this.jTextArea1.setText(rsa.getN() + "\n" + rsa.getE());
        BigInteger tekstUBroj = new BigInteger(tekst.getBytes());
        /* Encryption of the entered text */
        BigInteger sifriraniTekst = rsa.enkripcija(tekstUBroj);
        this.jTextArea2.setText(sifriraniTekst.toString());

        /* Decryption of the encrypted text to the original */
        tekstUBroj = rsa.dekripcija(sifriraniTekst);
        String text2 = new String(tekstUBroj.toByteArray());
        this.ispisDesifriranogTeksta.setText("Decrypted text: " + text2);
    }
```

Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

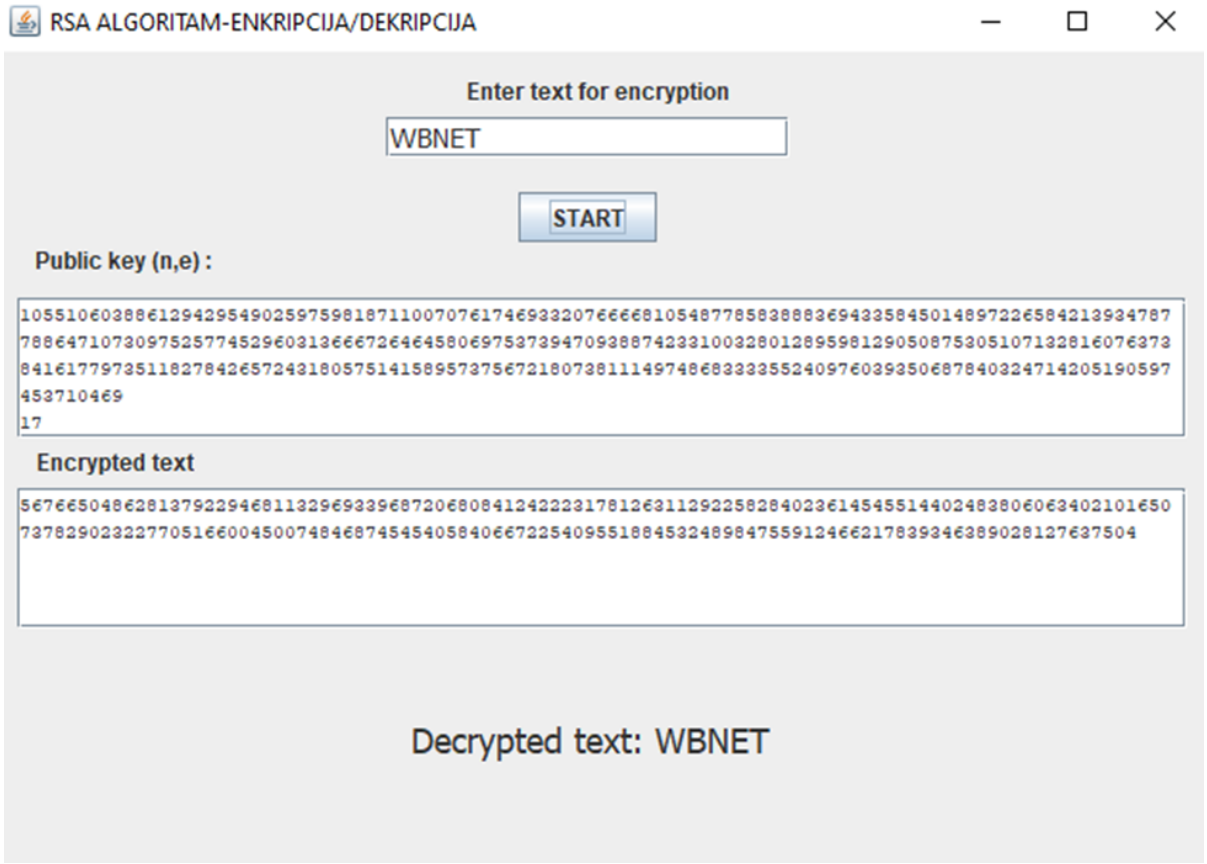
Prikaz RSA klase:

```
import java.math.BigInteger;
import java.security.SecureRandom;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

public class RSA {

    private int bitlen = 1024;
    BigInteger n, d, e;

    /**
     * Creating a class instance for encryption and decryption.
     */
    public RSA(int bits) {
        bitlen = bits;
        this.generirajKljuceve();
    }

    /**
     * Generate a new set of encryption keys
     */
    private synchronized void generirajKljuceve() {
        SecureRandom r = new SecureRandom();
        BigInteger p = new BigInteger(bitlen / 2, 100, r);
        BigInteger q = new BigInteger(bitlen / 2, 100, r);
        n = p.multiply(q);
        BigInteger m = (p.subtract(BigInteger.ONE)).multiply(q
                .subtract(BigInteger.ONE));
        e = new BigInteger("17");

        /**
         * Use a fixed public key (e.g., e=17) for encryption
         */
        while (m.gcd(e).intValue() > 1) {
            e = e.add(new BigInteger("2"));
        }
        d = e.modInverse(m);


        /**
         * saving the private key to the file PK.txt (p,q,d)
         */
    pisanjePKuFajl(p.toString() + "\n" + q.toString() + "\n" + d.toString());
    }

    private static void pisanjePKuFajl(String data) {
        try {
```

Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

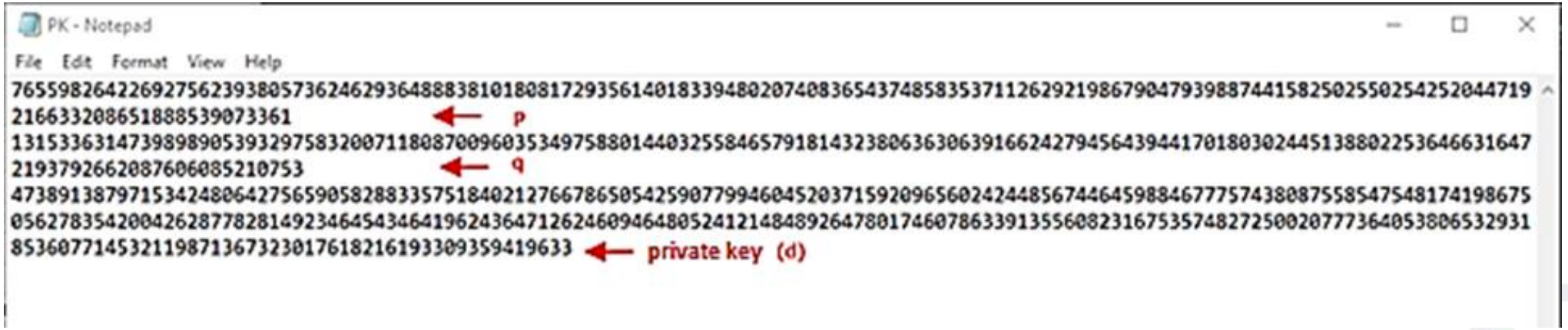
Pokretanje i testiranje aplikacije (unos teksta):



Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

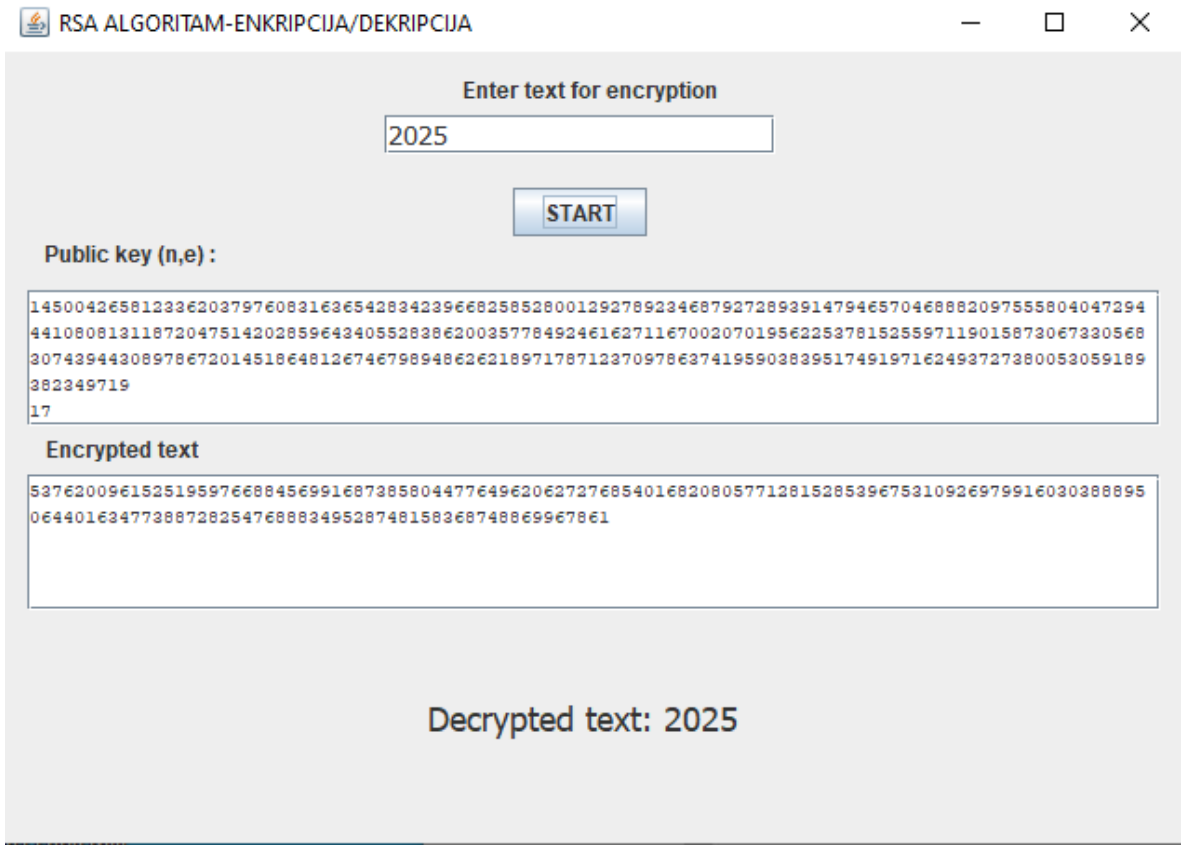
Zapis privatnog ključa u datoteci PK.txt:



Internet i računarska sigurnost

PRIMJER 2: Implementacija RSA algoritma u Java aplikaciji

Pokretanje i testiranje aplikacije (unos broja):



Internet i računarska sigurnost

Hash funkcije

Hash funkcija je matematička funkcija koja uzima ulazne podatke (ili poruku) i generira izlaznu vrijednost fiksne dužine, koja se obično naziva "hash vrijednost" ili "hash". Hash funkcije se koriste u mnogim sigurnosnim aplikacijama, kao što su provjera integriteta podataka, digitalni potpisi, autentifikacija, pohrana lozinki, itd.

Osnovne karakteristike hash funkcija su:

1. Jednosmjernost
2. Fiksna dužina izlaza
3. Unikatnost
4. Osjetljivost na promjene

Co-funded by the European Union

Internet i računarska sigurnost

Hash funkcije

Popularne *hash funkcije* uključuju:

- **MD5** (eng. Message Digest Algorithm 5)
  - smatra se zastarjelim zbog sigurnosnih ranjivosti.
- **SHA-1** (eng. Secure Hash Algorithm 1)
  - smatra se nesigurnim zbog mogućnosti sudara.
- **SHA-256**
  - dio SHA-2 porodice, vrlo siguran i široko korišten.
- **SHA-3**
  - najnovija verzija SHA hash funkcija.

Co-funded by the European Union

Internet i računarska sigurnost

PRIMJER: Implementacija RSA algoritma u Java aplikaciji

Implementacija i pokretanje SHA-256 hash funkcije:

```
import hashlib

# Input text
input_text = "Sensitive data"

# Create SHA-256 hash object
sha256_hash = hashlib.sha256()

# Update the hash object with the input data
sha256_hash.update(input_text.encode())

# Get the hash value
hashed_text = sha256_hash.hexdigest()

# Displaying the original text and its SHA-256 hash
print("Original text:", input_text)
print("SHA-256 Hash value:", hashed_text)
```

```
C:\Users\AzraKT\Desktop>python hash.py
Original text: Sensitive data
SHA-256 Hash value: 474de7276ecc120b83bc42291a96a36c648a0f6428bf7c88bfdccf32bfbb4339
```

Co-funded by the European Union

Internet i računarska sigurnost

RNG generatori

Generatori slučajnih brojeva (**RNG**, eng. Random Number Generators) ključna su komponenta u mnogim aspektima sigurnosti računarskih sustava i kriptografije.

Služe za generiranje brojeva koji su, prema definiciji, nasumični i nepredvidivi. RNG-ovi su temelj za stvaranje sigurnih ključeva, digitalnih potpisa, autentifikacije i mnogih drugih operacija u savremenim informacijskim tehnologijama. S obzirom na važnost njihove uloge, kvaliteta i sigurnost RNG-a direktno utječu na cijeli sistem u kojem se koriste.

Postoje dva osnovna tipa RNG generatora:

1. Pseudo-slučajni generatori (**PRNG**) i
2. Pravi slučajni generatori (**TRNG**).

Internet i računarska sigurnost

Pseudo-slučajni generatori (PRNG)

Pseudo-slučajni brojevi nisu "pravi" slučajni brojevi, već su generirani pomoću determinističkih algoritama.

To znači da, iako se ponašaju kao slučajni brojevi, oni su zapravo predvidljivi ako se zna početni uslov ili "sjeme" (eng. seed).

Pseudo-slučajni generatori široko se koriste u računarskim sistemima zbog svoje brzine i učinkovitosti.

Jedan od najpoznatijih PRNG-a je *Mersenne Twister*. Iako vrlo brz i učinkovit, PRNG-ovi nisu pogodna opcija za sigurnosne aplikacije jer ako napadači dođu do sjemena, mogu predvidjeti cijeli niz generiranih brojeva.

Co-funded by the European Union

Internet i računarska sigurnost

PRIMJER: Implementacija PRNG-a u Pythonu

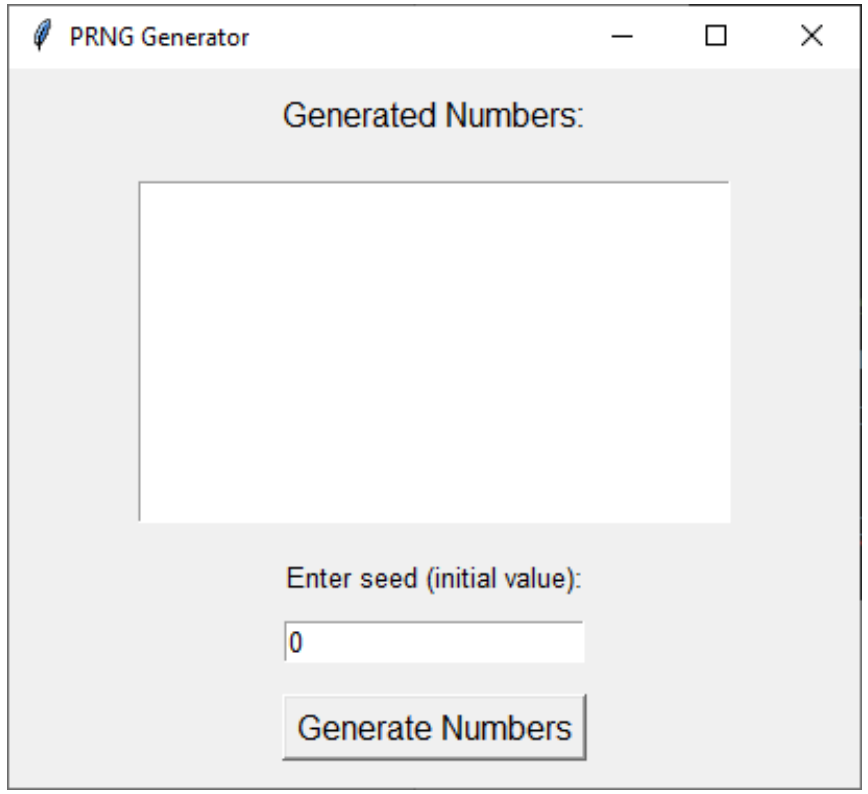
```
import tkinter as tk
from tkinter import messagebox

# PRNG - Linear Congruential Generator (LCG)
class PRNG:
    def __init__(self, seed=0):
        self.a = 1664525  # Multiplier
        self.c = 1013904223  # Increment
        self.m = 2**32  # Modulus
        self.state = seed

    def generate(self):
        self.state = (self.a * self.state + self.c) % self.m
        return self.state

# GUI Application
class PRNGApp:
    def __init__(self, root):
        self.root = root
        self.root.title("PRNG Generator")
        self.root.geometry("400x300")

        self.prng = PRNG()  # Create an instance of PRNG
```

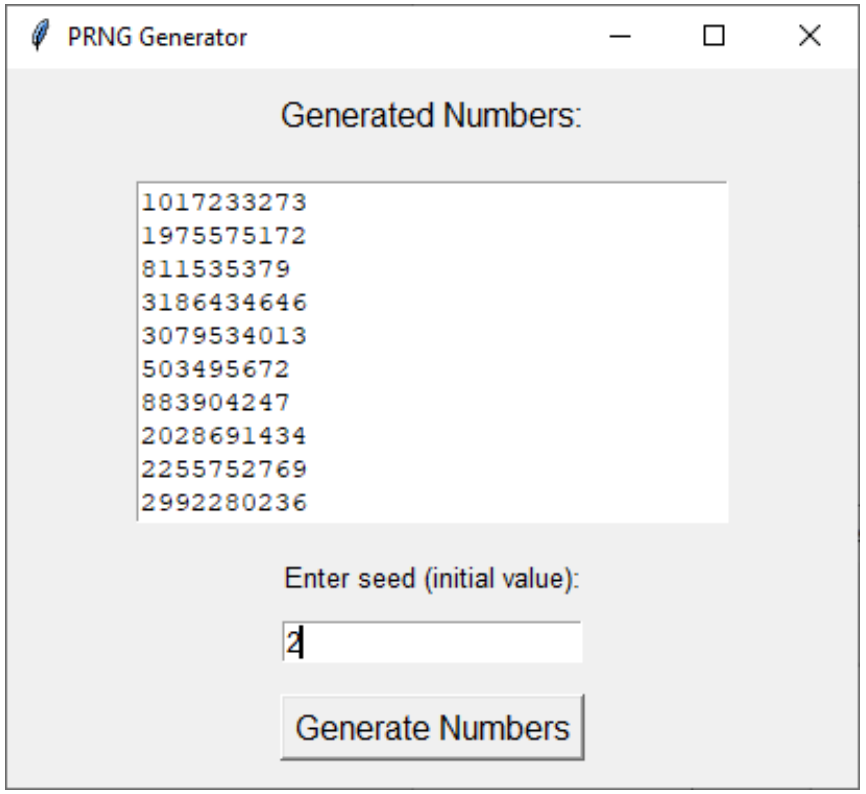
```
        # Label to display generated numbers
        self.output_label = tk.Label(root, text="Generated Numbers:", font=("Arial",
12))
        self.output_label.pack(pady=10)

        # Text box to display the numbers
        self.number_display = tk.Text(root, height=10, width=35)
        self.number_display.pack(pady=10)

        # Seed input field
        self.seed_label = tk.Label(root, text="Enter seed (initial value):",
font=("Arial", 10))
        self.seed_label.pack(pady=5)

        self.seed_entry = tk.Entry(root, font=("Arial", 10))
        self.seed_entry.pack(pady=5)
        self.seed_entry.insert(0, "0")  # Default value for seed

        # Button to generate numbers
        self.generate_button = tk.Button(root, text="Generate Numbers",
font=("Arial", 12), command=self.generate_numbers)
```

Internet i računarska sigurnost

PRIMJER: Implementacija PRNG-a u Pythonu

Pokretanje i testiranje aplikacije





Co-funded by the European Union

Internet i računarska sigurnost

Pravi slučajni generatori (TRNG)

Pravi slučajni generatori temelje se na fizičkim procesima, kao što su elektronički šum ili radioaktivni raspad, kako bi generirali nepredvidljive brojeve.

Ovi generatori su u pravilu sigurniji jer koriste izvor nesigurnosti koji nije deterministički i koji se ne može lako predvidjeti.

Međutim, TRNG generatori su često sporiji i skuplji od PRNG-a, te se rjeđe koriste zbog zahtjeva za specifičnim hardverom.

Co-funded by the European Union

Internet i računarska sigurnost

PRIMJER: Implementacija TRNG u Pythonu

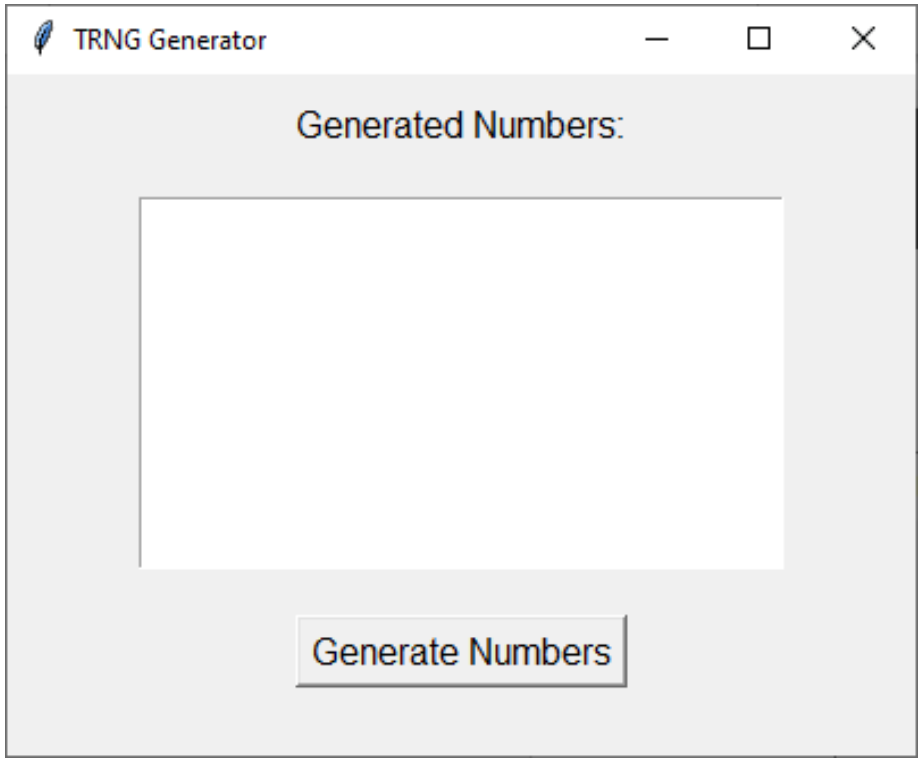
```
import tkinter as tk
from tkinter import messagebox
import os

# TRNG (True Random Number Generator) using os.urandom()
class TRNG:
    def generate(self):
        # Using os.urandom() which uses entropy from the operating system to generate
random numbers
        random_bytes = os.urandom(4)  # Generate 4 bytes of random data
        random_number = int.from_bytes(random_bytes, "big")  # Convert bytes into an
integer
        return random_number

# GUI Application
class TRNGApp:
    def __init__(self, root):
        self.root = root
        self.root.title("TRNG Generator")
        self.root.geometry("400x300")

        self.trng = TRNG()  # Create an instance of TRNG
```

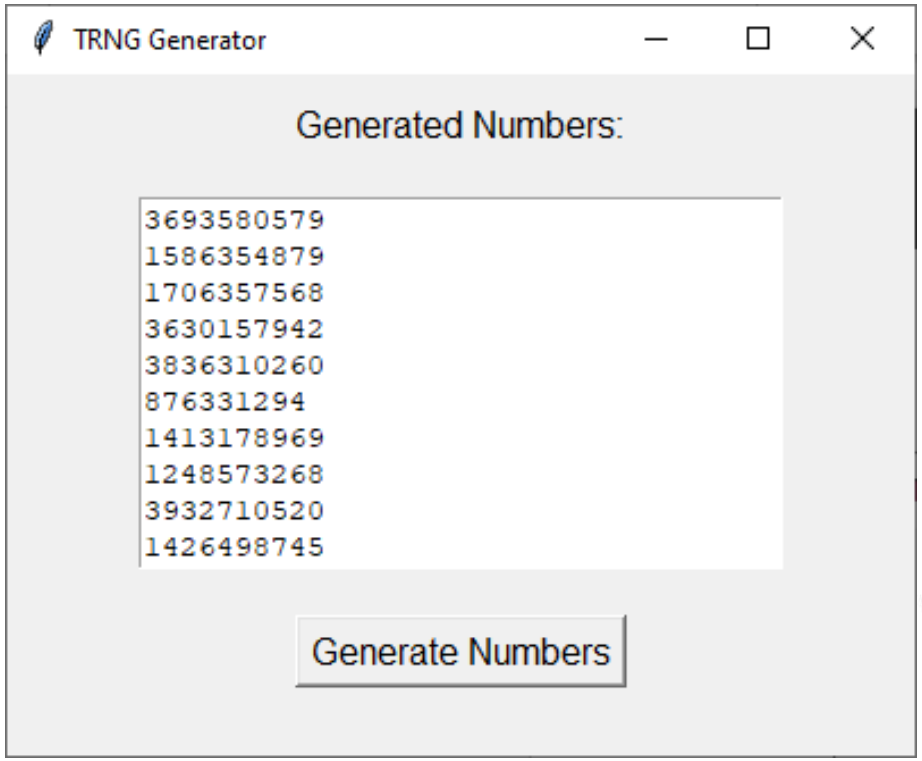
```
        # Label to display generated numbers
        self.output_label = tk.Label(root, text="Generated Numbers:", font=("Arial",
12))
        self.output_label.pack(pady=10)

        # Text box to display the numbers
        self.number_display = tk.Text(root, height=10, width=35)
        self.number_display.pack(pady=10)

        # Button to generate numbers
        self.generate_button = tk.Button(root, text="Generate Numbers",
font=("Arial", 12), command=self.generate_numbers)
        self.generate_button.pack(pady=10)

    def generate_numbers(self):
        self.number_display.delete(1.0, tk.END)  # Clear previous numbers
        for _ in range(10):  # Generate 10 numbers
            random_number = self.trng.generate()
            self.number_display.insert(tk.END, f"{random_number}\n")

# Create the main Tkinter window for the application
if __name__ == "__main__":
```

Internet i računarska sigurnost

PRIMJER: Implementacija TRNG u Pythonu

Pokretanje i testiranje aplikacije





Internet i računarska sigurnost

Primjena RNG generatora u kriptografiji

RNG generatori imaju ključnu ulogu u kriptografiji, gdje sigurnost sistema u velikoj mjeri ovisi o stvaranju nasumičnih brojeva koji su teški za predvidjeti.

Glavne primjene RNG-a u kriptografiji uključuju:

- Generiranje kriptografskih ključeva
- Salt i IV (Inicijalizacijski Vektor)
- Digitalni Potpisi i Certifikati
- Generiranje Session Ključeva

Internet i računarska sigurnost

ZAKLJUČAK

Završetkom ovog kursa, polaznici će steći sveobuhvatno znanje iz oblasti internet i računarske sigurnosti, uključujući:

- zaštitu korisnika i računara na internetu,
- prepoznavanje i prevenciju sigurnosnih prijetnji, kao i
- implementaciju zaštite putem kriptografskih tehnika.

Kroz praktične primjere u Pythonu, polaznicima će biti omogućeno da stečena teorijska znanja primijene u realnim situacijama, čime će razviti potrebne vještine za implementaciju sigurnosnih rješenja u digitalnom okruženju. Savladavanje kriptografskih algoritama i metoda zaštite podataka doprinosi njihovoj sposobnosti da prepoznaju i adresiraju sigurnosne izazove, čineći ih spremnim za suočavanje sa stalnim prijetnjama u današnjem dinamičnom internetu.

WBnet

Co-funded by the European Union

Questions & Answers

"Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them."

Network of centers for regional short study programs in the countries of the Western Balkans

Call: ERASMUS-EDU-2023-CBHE

Project number: 101128813

ATYCC

UNIVERSITY OF L]UBL]ANA
Faculty of Electrical Engineering

POLITÉCNICA

University of Pristina
Kosovska Mitrovica