

Kriptografija

III (11) gimnazijos klasė

Rekomenduojamas turinys ir užduotys

Tatjana Balvočienė

Informatikos mokytoja ekspertė
Šilutės Vydūno gimnazija

Antanas Balvočius

Kompetencijų aprašo, Bendrųjų
programų (BP) įvado ir Informatikos
BP bei rekomendacijų bendraautorius

2023 m. rugsėjis

Kriptografija. III (11) gimnazijos klasė

Informatikos bendroji programa

30.3.4. Kriptografinės sistemos, viešasis ir privatusis raktas. Prisimenamos kriptografinės sistemos, simetrinis ir asimetrinis šifravimas. Apibrėžiamos viešojo ir privačiojo rakto, sertifikato sąvokos. Išbandomi kriptografinių sistemų pavyzdžiai (pavyzdžiui, OpenPGP). Diskutuojama apie sertifikato patikimumo požymius.

Informatikos bendroji programa (2022). *Teisės aktų registras*:

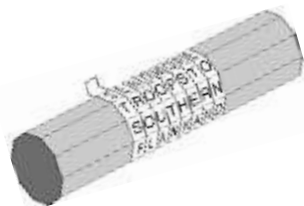
<https://e-tar.lt/rs/aesupplement/1a764050239511edb4cae1b158f98ea5/PibGuMnJPz/b52ac6c04foe11ee81b8b446907f594f/>
(2023-09-20)

Kriptografinių sistemų apibūdinimas

- ❑ **Kriptografija** – iš graikų kalbos **kripto** (*paslėptas*) ir **grafos** (*rašymas*).
- ❑ Kriptografija reiškia paslėptą rašymą, kurio **tikslas** aiškus – taip norima **apsaugoti informaciją** ir neleisti trečiosioms šalims prie jos prieiti.
- ❑ **Kriptografinės sistemos** – tai matematikos ir informatikos disciplinos, kurioje nagrinėjami duomenų užšifravimo ir šifro atkūrimo (iššifravimo) metodai.
- ❑ **Jų tikslas** – užtikrinti informacijos saugumą nuo nepageidaujamų asmenų (nuo asmenų, kurie neturi teisės matyti ar modifikuoti šiuos duomenis).

Kriptografinių sistemų pagrindiniai komponentai

- ❑ **Raktas** – informacija, metodas, sistema, būdas, kurie leidžia užšifruoti arba iššifruoti žinutę.
- ❑ **Šifravimo algoritmas** – algoritmas, kuris naudojamas norint paversti pradinę žinutę į užšifruotą formą naudojant raktą.
- ❑ **Iššifravimo (dešifravimo) algoritmas** – algoritmas, kuris naudojamas norint atkurti pradinę žinutę iš užšifruotos formos naudojant raktą.



1	2	3	4

I	S	Š	T
K	S	E	T
S	Y	A	R
S	L	A	A
T	A	P	S

Prislapis Eilutė Žodis
 ↓ ↓
 21-7-5
 ↓
 Eilutė
 ↓
 34:18:22
 ↑ ↑
Poslapis Raidės Nr.
 eilutėje



Pagrindinės šiuolaikinių kriptografinių sistemų kategorijos

- ❑ **Simetrinė kriptografija** – tai kriptografijos tipas, kai **tie patys šifravimo raktai** naudojami tiek užšifravimui, tiek iššifravimui. Taip šifruojant, pranešimo siuntėjas ir gavėjas turi turėti tą patį raktą, kad užšifruotas pranešimas būtų perskaitytas.
- ❑ **Asimetrinėje kriptografijoje** – pranešimo užšifravimui ir iššifravimui šiame **naudojami skirtingi raktai**. Paprastai yra **viešasis raktas**, kuris yra prieinamas visiems ir naudojamas užšifravimui, bei **privatūs raktai**, kurie naudojami tik iššifravimui ir yra laikomi paslapyje.
- ❑ **Hibridinė kriptografija** kombinuoja simetrinės ir asimetrinės kriptografijos privalumus. Paprastai naudojama asimetrinė kriptografija saugiam simetrinio rakto perdavimui, o po to su šiuo simetriniu raktu šifruojami duomenys.

Kurį pasirinkti?

- ❑ **Simetrinis šifravimas** yra greitas ir paprastas, bet kyla problemų, kai reikia saugiai pasidalyti ir saugoti raktus.
- ❑ **Asimetrinis šifravimas** šias problemas sprendžia, bet yra lėtesnis ir reikalauja daugiau kompiuterio resursų.

Kriptografinės sistemos naudojamos daugelyje modernių technologijų, tokių kaip elektroninis paštas, interneto svetainės, elektroninių mokėjimų sistemos, virtualūs privatūs tinklai (VPT), kriptovaliutos, karyba ir kt.

Simetrinė kriptografija gali būti skirstoma

- ❑ **Šifravimas nenaudojant kompiuterių** – dažniausiai yra istoriniai šifravimo metodai, kurie buvo naudojami prieš kompiuterių erą, pavyzdžiui, *Cezario šifras*, *Skytale*, *Knyginis šifras*, *Perstatų šifras* ir kt.
- ❑ **Šifravimas naudojant kompiuterius**, pavyzdžiui, *Advanced Encryption Standard* (AES – *Išplėstinis šifravimo standartas*), kuris yra vienas iš populiariausių ir plačiausiai naudojamų simetrinio šifravimo algoritmų.

Simetrinis šifravimas (kriptografija)

Problema

Kuo daugiau žmonių turės jūsų suteiktą raktą, tuo didesnė bus saugos rizika.



Simetrinio šifravimo naudojant kompiuterius schema



Užduotis

Simetrinio šifravimo atveju sauga ypatingai priklauso nuo šifravimo rakto (metodo), šio rakto saugojimo ir jo saugaus perdavimo komunikacijos dalyviui. Pakomentuokite simetrinio šifravimo galimas saugos spragas ir veiksmus, didinančius šifravimo saugą.



Simetrinė kriptografija turi įvairių taikymų sričių, kuriose ji gali būti vertinama dėl savo greitaveikos ir efektyvumo.

Simetrinė kriptografija – taikymas (1)

Duomenų saugojimas

- ❑ Šifruoti standieji diskai (naudojama siekiant apsaugoti duomenis diskuose nuo neautorizuotos prieigos).
- ❑ Duomenų saugojimo debesijos paslaugos (kai kurie tiekėjai leidžia klientams šifruoti savo failus prieš juos įkeliant į debesį).

Komunikacija:

- ❑ Šifruotos žinutės ir pokalbiai (kai kurios žinučių siuntimo programėlės naudoja simetrinę kriptografiją šifruoti pokalbiams ir nuotraukoms).
- ❑ Saugus VoIP (Voice over IP) (simetrinė kriptografija gali būti naudojama balso ir vaizdo pokalbiams šifruoti).

Simetrinė kriptografija – taikymas (2)

Tinklų saugumas

- ❑ Wi-Fi saugumas (WPA ir WPA2 – bevielio tinklo saugumo protokolai naudoja simetrinę kriptografiją (pavyzdžiui, AES) saugumui užtikrinti).
- ❑ VPN (VPT – *virtualūs privatūs tinklai*) paslaugos (simetrinė kriptografija yra naudojama greitai ir saugiai perduoti duomenis).

Finansų sektorius

- ❑ Elektroniniai mokėjimai (simetrinė kriptografija yra naudojama piniginių transakcijų duomenims saugoti).
- ❑ ATM transakcijos (kai kurios bankomato operacijos gali naudoti simetrinę kriptografiją saugoti piniginei informacijai).

Simetrinė kriptografija – taikymas (3)

Pramoninės ir kontrolinės sistemos

- ❑ SCADA ir ICS (naudojama simetrinę kriptografiją saugoti komunikacijai tarp valdymo centų ir nuotolinių įrenginių).

Elektroninė prekyba

- ❑ Saugus duomenų perdavimas (simetrinė kriptografija gali būti naudojama užtikrinti konfidencialumą perduodant klientų informaciją ir mokėjimo duomenis).

Šifruotas IPTV ir kiti šifruoti transliacijos protokolai

- ❑ Naudojama apsaugoti nuo neautorizuotos prieigos prie turinio.

Simetrinė kriptografija – taikymas (4)

Automobilių pramonė

- ❑ Saugus nuotolinis valdymas (simetrinė kriptografija gali būti naudojama saugiai komunikacijai tarp nuotolinio rakto ir automobilio užtikrinti).

Medicinos įrenginiai

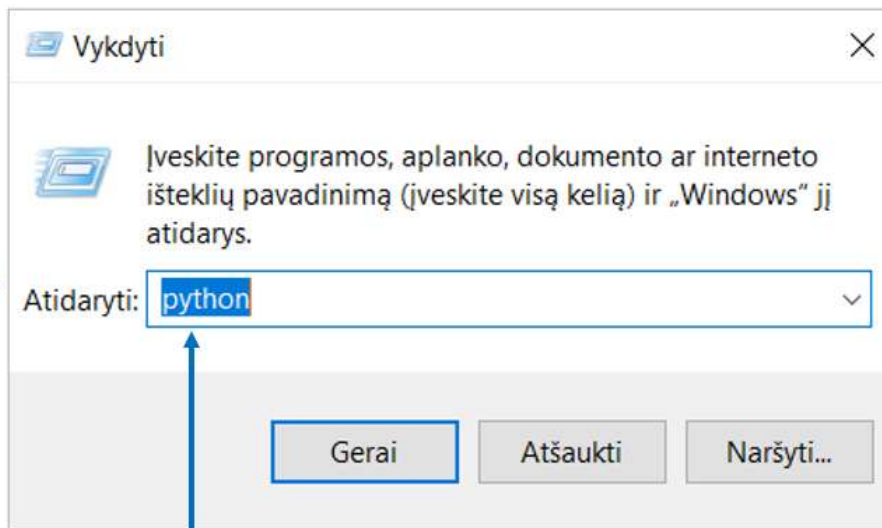
- ❑ Paciento duomenų apsauga (simetrinė kriptografija gali būti naudojama saugoti jautrius paciento duomenis medicinos įrenginiuose).

Ir kitos taikymo sritys.

Panagrinėkime pavyzdžius, kaip gali būti panaudota programavimo kalba *Python* kuriant kompiuterio programas, naudojančias simetrinio *Advanced Encryption Standard (AES)* šifravimo biblioteką tekstui užšifruoti ir iššifruoti.

Pasiruošimas (1)

Klaviatūroje
paspauskite:



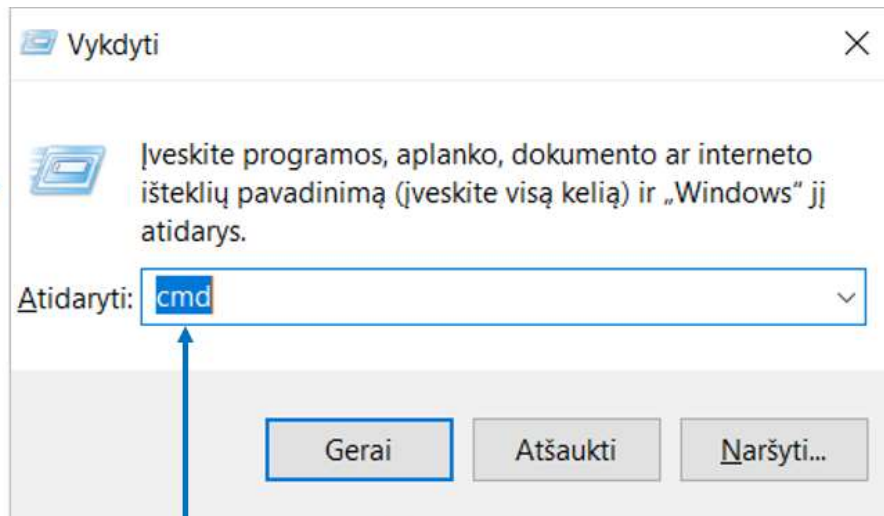
Atsivėrusiame langelyje parašykite
python ir paspauskite klavišą **Enter ↵**



Instaliuojama
Python programa

Pasiruošimas (2)

Klaviatūroje paspauskite:

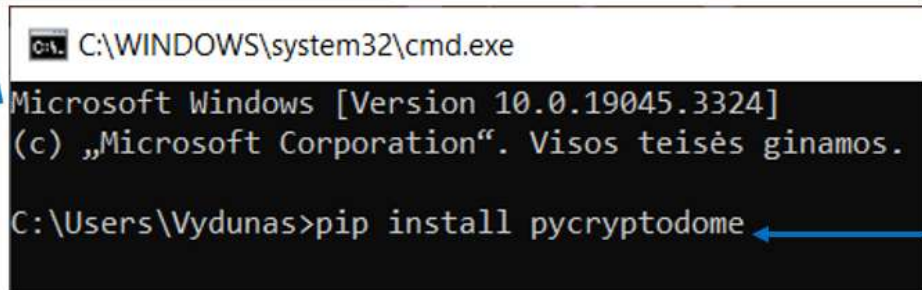


Atsivėrusiame langelyje parašykite **cmd** ir paspauskite klavišą **Enter ↵**

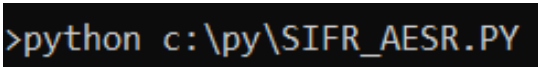
Kopijuoti: **pip install pycryptodome**

Pereinama į Windows komandų eilutę. Joje parašykite (nukopijuokite) **pip install pycryptodome** ir paspauskite **Enter ↵**

Taip įdiegiama Python biblioteka **PyCryptodome**, reikalinga šifravimo veiksams atlikti.



Išbandykite simetrinio užšifravimo įrankį (1)

- ❑ Nusikopijuokite čia pateiktą programos kodą į „Užrašinę“ ir įrašykite vardu **SIFR_AESR.PY** į **C:** disko aplanką **py**.
- ❑ Įvykdysite programą, komandų eilutėje nurodydami* **python c:\py\SIFR_AESR.PY**

- ❑ Panagrinėkite programos darbo rezultatą – informaciją, pateiktą įvykdžius programą (žr. kitą skaidrę).

***Pastaba.** Jei įrašysite programą kitų vardu ir (arba) kitame aplanke, komandų eilutėje rašykite savo informaciją – nurodykite savo aplanką ir savo failo vardą.

```
# PROGRAMOS SIFR_AESR.PY KODAS
from Crypto.Cipher import AES
```

```
def main():
    # Naudokime pastovų 32 baitų ilgio raktą
    key = b"ThisIsASecretKey_32bytesLongAbCd"

    # Inicijuoti AES šifro objektą su EAX režimu
    cipher = AES.new(key, AES.MODE_EAX)

    # Teksto užšifravimas
    plaintext = b"Slapta: Kompiuteris yra tavo ir mano geriausias draugas :-)."
    ciphertext, tag = cipher.encrypt_and_digest(plaintext)

    # Atspausdiname raktą, užšifruotą tekstą, tag ir nonce reikšmes
    print(f"Raktas: {key.decode('utf-8')}")
    print(f"Užšifruotas tekstas: {ciphertext.hex()}")
    print(f"Tag: {tag.hex()}")
    print(f"Nonce: {cipher.nonce.hex()}") # Atspausdiname nonce reikšmę

    # Teksto iššifravimas
    cipher_dec = AES.new(key, AES.MODE_EAX, nonce=cipher.nonce)
    decrypted = cipher_dec.decrypt(ciphertext)

    # Patikriname, ar iššifruotas tekstas sutampa su pradiniais duomenimis
    try:
        cipher_dec.verify(tag)
        print(f"Pradinis: {decrypted.decode('utf-8')}")
    except ValueError:
        print("Iššifruotasis tekstas yra neteisingas!")

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

Išbandykite simetrinio užšifravimo įrankį (2)

- ❑ Įvykdytos programos **SIFR_AESR.PY** darbo rezultatas:

```
Raktas: ThisIsASecretKey_32bytesLongAbCd
Užšifruotas tekstas: e7883d8abd0243e2636be5c52899b1d3c749ba080807a22c959394750e24d72d35547cd15e570627517c700e1148f590208f79f7f7afee7fbe7076073
Tag: 733d3cfd1421fdc76095bc8f65175d73
Nonce: b05e6383dee9e5e6406f112795d7276c
Pradinis: Slapta: Kompiuteris yra tavo ir mano geriausias draugas :-).
```

- ❑ Rezultato teksto dalys: **Raktas**, **Užšifruotas tekstas**, **Tag**, **Nonce**, **Pradinis**.
- ❑ Apie **Nonce** ir **Tag**:
 - **Nonce** garantuoja, kad tam pačiam pranešimui, net ir naudojant tą patį raktą, bus sugeneruotas skirtingas šifro tekstas kiekvieno šifravimo metu.
 - **Tag** yra simbolių seka, kuri garantuoja pranešimo autentiškumą ir integralumą. Ji leidžia gavėjui patikrinti ir įsitikinti, ar šifruotasis pranešimas arba duomenys nebuvo pažeisti ar pakeisti.
 - **Išvada:** **Nonce** užtikrina, kad kiekvienas šifro tekstas yra unikalus, o **Tag** patvirtina, kad šifruotasis pranešimas yra autentiškas ir nepažeistas.

Išbandykite simetrinio užšifravimo įrankį (3)

- ❑ Užšifruokite kitą tekstą – įrašykite norimą užšifruoti tekstą* atitinkamoje pateiktos programos vietoje ir įvykdysite programą.
- ❑ Įvykdytos programos su kitu užšifruojamu tekstu darbo rezultatas:

```
Raktas: ThisIsASecretKey_32bytesLongAbCd
Užšifruotas tekstas: dbb6fd975fafc6282d8a27dd9841cc26bf67929926e7648fa
0b50fdc88bff2641ab5e68b7c447f310d527b6ff2df1f5c4498c4be4ede13
Tag: 265df1dfa03b68cd368605e2660c763d
Nonce: 78db0191dcadb80bb318579ad3a535d3
Pradinis: Kas nedirba, mielas vaike, tam ir duonos duot nereikia.
```

***Pastaba.** Šioje užšifravimo kompiuterio programoje naudokite tik ASCII kodo raides (raides be diakritinių ženklų).

```
# PROGRAMOS SIFR_AESR.PY KODAS
from Crypto.Cipher import AES
```

```
def main():
    # Naudokime pastovų 32 baitų ilgio raktą
    key = b"ThisIsASecretKey_32bytesLongAbCd"
```

```
# Inicijuoti AES šifro objektą su EAX režimu
cipher = AES.new(key, AES.MODE_EAX)
```

```
# Teksto užšifravimas
plaintext = b"Kas nedirba, mielas vaike, tam ir duonos duot nereikia."
ciphertext, tag = cipher.encrypt_and_digest(plaintext)
```

```
# Atspausdiname raktą, užšifruotą tekstą, tag ir nonce reikšmes
print(f"Raktas: {key.decode('utf-8')}")
print(f"Užšifruotas tekstas: {ciphertext.hex()}")
print(f"Tag: {tag.hex()}")
print(f"Nonce: {cipher.nonce.hex()}") # Atspausdiname nonce reikšmę
```

<...>

Čia pateiktas programos teksto fragmentas

Nukopijuokite programos darbo rezultato informaciją į, pavyzdžiui, programą „Užrašinė“ ir įrašykite failą– šią informaciją panaudosite iššifruodami užšifruotą pranešimą (žr. kitą skaidrę).

Išbandykite simetrinio iššifravimo įrankį (1)

- ❑ Nusikopijuokite čia pateiktą programos kodą į užrašinę.
- ❑ Į programos teksto atitinkamas vietas įkelkite anksčiau išsaugotą užšifravimo informaciją*:
 - Raktas → **key**
 - Užšifruotas tekstas → **ciphertext_hex**
 - Tag → **tag_hex**
 - Nonce → **nonce_hex**
- ❑ Įrašykite programą vardu **SIFR_AESS.PY** į **C:** disko aplanką **py**.

*Pavyzdyje panaudota informacija, gauta užšifravus tekstą
Kas nedirba, mielas vaike, tam ir duonos duot nereikia.

```
# PROGRAMOS SIFR_AESS.PY KODAS
from Crypto.Cipher import AES
from binascii import unhexlify

def decrypt_aes_eax(ciphertext_hex, key, tag_hex, nonce_hex):
    # Konvertuojame šiuos duomenis iš šešiolyktainės į baitų sekas
    ciphertext = unhexlify(ciphertext_hex)
    tag = unhexlify(tag_hex)
    nonce = unhexlify(nonce_hex)

    # Sukuriamas šifravimo objektas su pateiktu raktu ir nonce
    cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)

    # Bandoma iššifruoti tekstą ir patikrinti MAC su pateiktu tag'u
    try:
        decrypted_data = cipher.decrypt_and_verify(ciphertext, tag)
        return decrypted_data.decode('utf-8')
    except ValueError:
        return "Užšifruotas tekstas buvo pažeistas. Iššifravimas nutrauktas."

def main():
    # Pateikti duomenys
    key = b"ThisIsASecretKey_32bytesLongAbCd"
    ciphertext_hex =
    "dbb6fd975fafc6282d8a27dd9841cc26bf67929926e7648fa0b5ofdc88bff26
    41ab5e68b7c447f310d527b6ff2df1f5c4498c4be4ede13"
    tag_hex = "265df1dfa03b68cd368605e2660c763d"
    nonce_hex = "78db0191dcadb80bb318579ad3a535d3"

    decrypted_message = decrypt_aes_eax(ciphertext_hex, key, tag_hex,
    nonce_hex)
    print(f'Iššifruotas tekstas: {decrypted_message}')

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

Išbandykite simetrinio iššifravimo įrankį (2)

- ❑ Įvykdykite programą, komandų eilutėje nurodę*:

python c:\py\SIFR_AESS.PY

```
>python c:\py\SIFR_AESR.PY
```

- ❑ Programos darbo rezultatas – iššifruotas pranešimas:

```
Iššifruotas tekstas: Kas nedirba, mielas vaike, tam ir duonos duot nereikia.
```

***Pastaba.** Jei įrašysite programą kitų vardu ir (arba) kitame aplanke, komandų eilutėje rašykite savo informaciją – nurodykite savo aplanką ir savo failo vardą.

```
# PROGRAMOS SIFR_AESS.PY KODAS
from Crypto.Cipher import AES
from binascii import unhexlify

def decrypt_aes_eax(ciphertext_hex, key, tag_hex, nonce_hex):
    # Konvertuojame šiuos duomenis iš šešiolyktainės į baitų sekas
    ciphertext = unhexlify(ciphertext_hex)
    tag = unhexlify(tag_hex)
    nonce = unhexlify(nonce_hex)

    # Sukuriamas šifravimo objektas su pateiktu raktu ir nonce
    cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)

    # Bandoma iššifruoti tekstą ir patikrinti MAC su pateiktu tag'u
    try:
        decrypted_data = cipher.decrypt_and_verify(ciphertext, tag)
        return decrypted_data.decode('utf-8')
    except ValueError:
        return "Užšifruotas tekstas buvo pažeistas. Iššifravimas nutrauktas."

def main():
    # Pateikti duomenys
    key = b"ThisIsASecretKey_32bytesLongAbCd"
    ciphertext_hex =
    "dbb6fd975fafc6282d8a27dd9841cc26bf67929926e7648fa0b50fdc88bff26
    41ab5e68b7c447f310d527b6ff2df1f5c4498c4be4ede13"
    tag_hex = "265df1dfa03b68cd368605e2660c763d"
    nonce_hex = "78db0191dcadb80bb318579ad3a535d3"

    decrypted_message = decrypt_aes_eax(ciphertext_hex, key, tag_hex,
    nonce_hex)
    print(f"Iššifruotas tekstas: {decrypted_message}")

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


Išbandykite simetrinio iššifravimo įrankį (3)

- ❑ Išstirkite, koks bus iššifravimo programos darbo rezultatas, jei joje pateiksime šiek tiek pakeistą užšifruotą tekstą, pavyzdžiui, tekste bus pakeistas vienas simbolis: dbb6 → dbb8

- ❑ Išsiaiškinkite, kas vyks iššifruojant, jei pakeisite programos kitų dalių užšifravimo informaciją:
 - key
 - tag_hex
 - nonce_hex

```
<...>  
  
def main():  
    # Pateikti duomenys  
    key = b"ThisIsASecretKey_32bytesLongAbCd"  
    ciphertext_hex =  
    "dbb6fd975fafc6282d8a27dd9841cc26bf67929926e764  
    8fa0b50fdc88bff2641ab5e68b7c447f310d527b6ff2df1f5  
    c4498c4be4ede13"  
    tag_hex = "265df1dfa03b68cd368605e2660c763d"  
    nonce_hex = "78db0191dcadb80bb318579ad3a535d3"  
  
<...>  
Čia pateiktas tik programos teksto fragmentas.
```



Išbandykite simetrinio iššifravimo įrankį (5)

- ❑ Naudodami pateiktą simetrinio iššifravimo įrankį – programą **SIFR_AESS.PY** – iššifruokite čia pateiktus užšifruotus pranešimus:

1.

Raktas: ThisIsASecretKey_32bytesLongAbCd

Užšifruotas tekstas:

47b3ec03eacbe4aaddb22192c685f69da5c0fab86d5b7c0fb56e9c00f1035ef59eb1146cbd44138d9e9fa69e7b3b8957a65ffa3
464a3eebb5762b9172e64844cfe380807c2baf7c4b11a23af2e25969a52e7275e2e1591dfd3cd0392c18c7db1c6397b69800

Tag: 692a8142327e1fd3de00bd76a1833931

Nonce: 3150360edd204dbcb610883b168b5958

2.

Raktas: ThisIsASecretKey_32bytesLongAbCd

Užšifruotas tekstas:

a8cfed33a6ca1fd8bc16f5f712544a9c6350f20b5597495d46b4222b81c0cd7c4ead46e9811bcd7040d814348123cbc4718df4e03
403106d7dc846e009755cd2943ed6a044761c410e6e88c5f2

Tag: 1d7fce75962fa02764f15b57769d097a

Nonce: 6900c5899fa76800ba7bff77e0fe6cb6

Simetrinis šifravimas naudojant šifravimo raktą (1)

Patarimas. Prieš atliekant tolesnius veiksmus, patartina pakartotinai įdiegti / atnaujinti kriptografijos biblioteką. Tam komandų (CMD) eilutėje rašome **pip install --upgrade cryptography**

1. Naudodami Python SIM.PY programą, aplanke **c:\py** sukurkime simetrinį šifravimo raktą:

- nusikopijuokite čia pateiktą programos kodą į užrašinę ir įrašykite vardu **SIM.PY** į **C:** disko aplanką **py**;
- įvykdysite programą, komandų eilutėje nurodydami komandas:
cd c:\py
python SIM.PY

PROGRAMOS SIM.PY KODAS

```
# sim.py generuoja simetrinį raktą key.bin
from cryptography.fernet import Fernet
# Sugeneruojame simetrinį raktą
key = Fernet.generate_key()
# Išsaugome raktą į key.bin failą
with open("key.bin", "wb") as key_file:
    key_file.write(key)
print(f"Raktas sugeneruotas ir išsaugotas į key.bin:
{key}")
```

- aplanke **py** bus sukurtas **simetrinio šifravimo raktas** – failas **key.bin**.

Simetrinis šifravimas naudojant šifravimo raktą (2)

2. Naudodami **Python** programą **Simetrinis_e_d.py** sukurtu simetrinio šifravimo raktu **key.bin** užšifruokite posakį:

If at first you don't succeed, call it version 1.0.

(Jeigu iš pirmo karto nepasisekė, pavadink tai versija 1.0. – Programuotojų išmintis)

- Šį posakį įkelkite į programą „Užrašinė“ ir įrašykite į aplanką **c:\py** vardu, pavyzdžiui, **posakis.txt**.
- Į tą patį aplanką įrašykite ir simetrinio šifravimo / iššifravimo programą **Simetrinis_e_d.py** (jos kodas pateiktas kitoje skaidrėje)
- Komandų eilutėje nurodykite*:
cd c:\py
python simetrinis_e_d.py e key.bin posakis.txt posakis.bin

***Pastaba.** Jei įrašysite programą ir šifravimo raktą kitų vardu ir (arba) kitame aplanke, komandų eilutėje rašykite savo informaciją – nurodykite savo aplanką ir savo failų vardus.

Simetrinis šifravimas

naudojant šifravimo raktą (3)

3. Iššifruokite užšifruotą tekstą:

- Naudodami programą „Užrašinė“ atverkite failą **posakis.bin**. Kaip galvojate, ar įmanoma be rakto jį iššifruoti?
- Ištrinkite arba pervardykite aplanke **c:\py** pradinio teksto failą **posakis.txt**. Pabandykite jį atstatyti iš užšifruoto failo vykdydami komandas komandų (CMD) eilutėje:

python simetrinis_e_d.py d key.bin posakis.bin posakis.txt

- Perskaitykite iššifruotą tekstą, kurį programa įrašė į failą **posakis.txt**. Ar pastebėjote kokius nors neatikimus ar iškraipymus palyginus su pradiniu tekstu?

```
# python Simetrinis_e_d.py e|d key.bin Tekstas.txt Tekstas.bin
from cryptography.fernet import Fernet
import sys
```

```
def encrypt(key, plaintext):
    cipher_suite = Fernet(key)
    return cipher_suite.encrypt(plaintext)
```

```
def decrypt(key, ciphertext):
    cipher_suite = Fernet(key)
    return cipher_suite.decrypt(ciphertext)
```

```
def main():
    action = sys.argv[1]
    key_file = sys.argv[2]
    text_file = sys.argv[3]
    output_file = sys.argv[4]
```

```
    with open(key_file, 'rb') as f:
        key = f.read()
```

```
    with open(text_file, 'rb') as f:
        text = f.read()
```

```
    if action == 'e':
        encrypted_text = encrypt(key, text)
        with open(output_file, 'wb') as f:
            f.write(encrypted_text)
    elif action == 'd':
        decrypted_text = decrypt(key, text)
        with open(output_file, 'wb') as f:
            f.write(decrypted_text)
```

```
    else:
        print("Invalid command. Use 'e' for encryption or 'd' for decryption.")
```

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

Simetrinis šifravimas naudojant šifravimo raktą (4)

- ❑ Naudodami programą **Simetrinis_e_d.py**, pabandykite iššifruoti čia pateiktą „labai slapta“ užšifruotą tekstą:

```
gAAAAABIFHwoMeVcPQwYOn2Cln9Uq41Kr2Gr_ND5e9_YKm2IVFqkL8MktROpvZdvJV9LoesjVWG6BTrUdN1EpSSBAIXQRgzOat4TMI  
5RclK8PjCoxXRElyosiOq8rGcvRLJqHJjH_WDYsFuTfOcWpSOY8zMqmlLZK7MzHXZwH8v_YQkEVstb5zuQ83OuhEBlgob8yqNfHnPodg  
EyYsjHxg_o3R116LLg==
```

- Pateiktą užšifruotą tekstą įkelkite į programą „Užrašinė“ ir įrašykite į **c:\py** vardu **slaptas_tekstas.bin**.
- Į programą „Užrašinė“ įkelkite šį simetrinį raktą: **ucE_arr6D5lyPcjNMU_CuCUwl1PsfwBW7uEDBgVx2yU=** ir įrašykite į **c:\py** vardu **key.bin**.
- Iššifruokite šį tekstą naudodami programą **Simetrinis_e_d.py**, ir simetrinį raktą **key.bin**, komandų (CMD) eilutėje nurodydami:
python simetrinis_e_d.py d key.bin slaptas_tekstas.bin slaptas_tekstas.txt
- Perskaitykite iššifruotą tekstą. Ar tikrai jį buvo galima laikyti labai slaptu? 😊

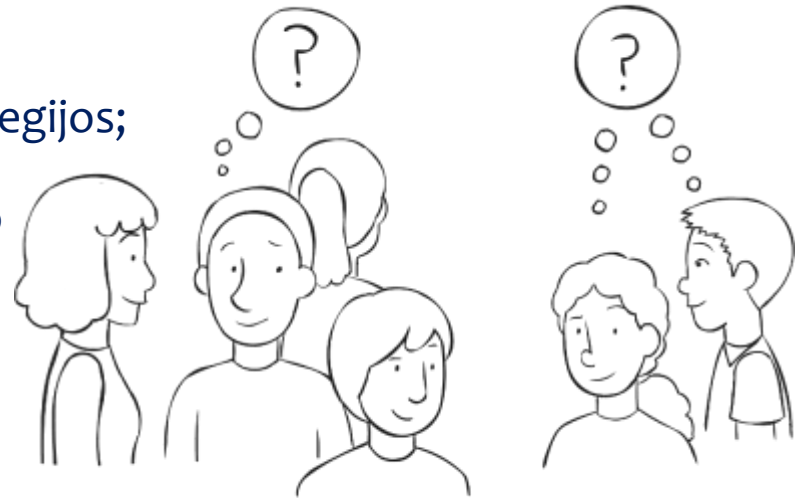
Užduotis darbui grupėse: pranešimo užšifravimas ir iššifravimas **naudojant savo sukurtą šifravimo raktą (1)**

- ❑ Mokytojui padedant pasidalinkite į mažas grupes po 2–3 mokinius.
- ❑ Kiekvienas grupės narys:
 - naudodamas **Python** programą **SIM.PY**, sukurkite savo šifravimo raktą **key.bin**;
 - programa **Simetrinis_e_d.py** ir savo sukurtu šifravimo raktu **key.bin** užšifruokite norimą informaciją ir perduokite kitam grupės nariui (būtina perduoti ir šifravimo raktą – pagalvokite, kaip tai atlikti laikantis saugumo reikalavimų);
 - iššifruokite iš grupės draugo gautą informaciją (naudokite programą **Simetrinis_e_d.py** ir grupės draugo perduotą raktą **key.bin**).

Užduotis darbui grupėse: pranešimo užšifravimas ir iššifravimas **naudojant savo sukurtą šifravimo raktą** (2)

❑ Grupelėse, po to ir visi kartu su mokytoju aptarkite:

- kaip pavyko atlikti užduotį;
- ar pakankamai saugios buvo jūsų pasirinktos šifravimo rakto perdavimo strategijos;
- ką galima būtų patobulinti siekiant didesnio užšifruotos informacijos saugumo.



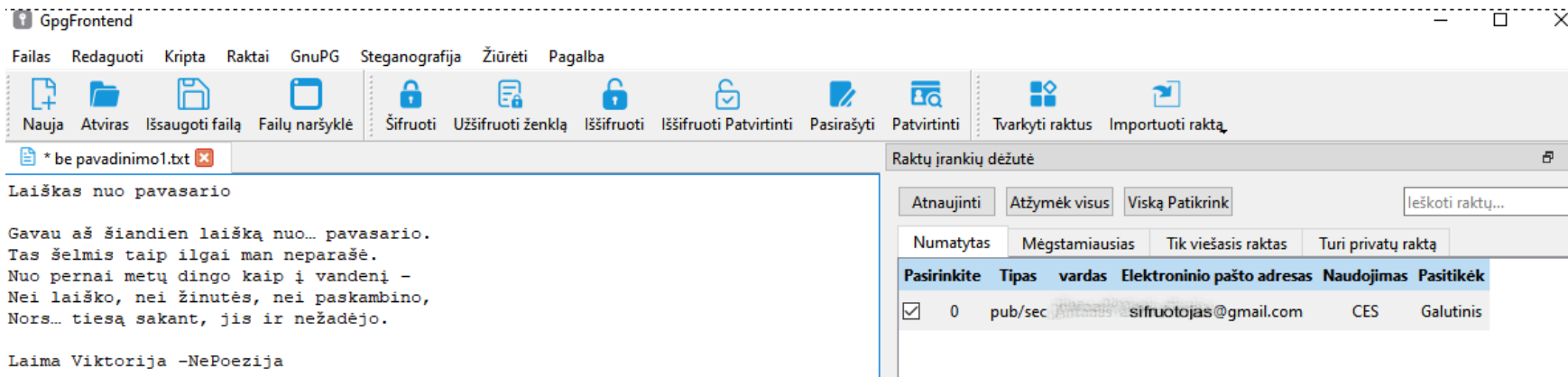
Profesionalioje praktikoje naudojamos simetrinės kriptografijos sistemos pavyzdys (1)

- ❑ **GpgFrontend** yra kelių platformų šifravimo įrankis, kuris atitinka **OpenPGP standartą**. Jos tikslas – supaprastinti OpenPGP naudojimą, taip suteikiant galimybę daugiau asmenų apsaugoti savo privatumą.
 - **OpenPGP yra duomenų šifravimo ir iššifravimo standartas**, kurį palaiko GpgFrontend.
 - GpgFrontend labai gerai dokumentuota programa.
 - Programos svetainėje randama patarimų, kaip pasirašyti šifruotą pranešimą bei saugiai elgtis su raktais.

Atsisiųsti įrankį GpgFrontend: <https://www.gpgfrontend.bktus.com/#/downloads>
Įrankio naudojimosi aprašas: <https://www.gpgfrontend.bktus.com/manual/encrypt-decrypt-text>

Profesionalioje praktikoje naudojamos simetrinės kriptografijos sistemos pavyzdys (2)

Atsisiuntę ir įdiegę įrankį GpgFrontend, galite peržiūrėti ReadME instrukcijas (rodomas diegimo metu) ir pradėti ją naudoti atlikę vos kelis veiksmus.



GpgFrontend programos lango fragmentas

SIMETRINIO ŠIFRAVIMO PRIVALUMAI IR TRŪKUMAI

Simetrinis šifravimas – privalumai

- ❑ **Greitaveika.** Simetrinis šifravimas yra daug greitesnis nei asimetrinis. Tai reiškia, kad jūs galite užšifruoti ir iššifruoti daugiau informacijos per mažesnį laiką.
- ❑ **Mažiau resursų.** Simetriniai algoritmai paprastai reikalauja mažiau kompiuterio procesoriaus pajėgumo ir atminties, todėl jie yra patogūs naudoti įvairiose sistemose, net ir senesnėse ar mažesnio galingumo.
- ❑ **Paprastumas.** Nereikia rūpintis dviem raktų poromis kaip asimetriniame šifravime. Vienas raktas veikia tiek užšifravimui, tiek iššifravimui, tai palengvina procesą.

Simetrinis šifravimas – trūkumai

- ❑ **Rakto pasidalinimo problema.** Didžiausias trūkumas yra būtinybė saugiai pasidalyti raktu tarp siuntėjo ir gavėjo. Jei kas nors nepageidaujamas gaus šį raktą, jis galės atšifruoti visą informaciją.
- ❑ **Rakto saugojimas.** Kadangi užšifravimo ir iššifravimo raktas yra tas pats, tiek išsiųstos, tiek gautos žinutės tampa pažeidžiamos, jei šis raktas prarastas ar pavogtas.
- ❑ **Mažesnis lankstumo lygis.** Simetrinio šifravimo algoritmai yra mažiau lankstūs pasirinkimų atžvilgiu, palyginti su asimetriniais. Jie dažniausiai yra tinkami tik duomenų šifravimui, bet ne identifikavimui ar elektroniniams parašams, kas puikiai atliekama naudojant asimetrinį šifravimą.

ASIMETRINĖ KRIPTOGRAFIJA

Asimetrinis šifravimas

- **Asimetrinio šifravimo sistema yra kriptografijos algoritmas**, kuriame naudojami du matematiškai susiję, bet skirtingi raktai – **viešasis raktas** ir **privatus raktas**.
 - **Viešasis raktas** yra prieinamas visiems ir gali būti **naudojamas** informacijos **užšifravimui** arba el. parašui tikrinti.
 - **Privatus raktas** yra saugomas paslapyje ir naudojamas informacijos iššifravimui arba el. parašo sukūrimo procese.
 - Viešasis ir privatusis raktai yra tarpusavyje susiję taip, **kad informacija, užšifruota viešuoju raktu, gali būti iššifruota tik atitinkamu privačiu raktu**. Tačiau naudojant šiuos raktus, negalima lengvai atkurti vieno ar kito iš jų – turint viešąjį raktą, negalima lengvai gauti privataus rakto.

Prisiminkite (arba atlikite) ir aptarkite užduotį, demonstruojančią, kaip vyksta keitimasis informacija naudojant asimetrinio šifravimo metodą (užduotis aprašyta 9 – 10 klasės metodinėje medžiagoje – pateiktyje ir užduočių faile).

Užduotis (veikla), demonstruojanti, kaip vyksta asimetrinis šifravimas

Aprašomos veiklos esmė

- Parodoma, kad informacijai šifruoti asimetriniu šifravimu atliekami du skirtingi veiksmai: vienas, kuris atliekamas (prieš išsiunčiant) informacijai užrašyti (užšifruoti), ir kitas, kuris atliekamas (gavus siuntą) informacijai atrinkinti (iššifruoti), ir kad šie veiksmai atliekami naudojant atitinkamus skirtingus raktus – viešąjį ir privaatų.
- Aptariant šios užduoties veiklas, pateikiamas pranešimo užšifravimo bei iššifravimo realaus proceso žingsnių imitavimas.
- Šis pavyzdys yra paprastas, bet jis leidžia suprasti asimetrinio šifravimo esmę: galimybę saugiai perduoti informaciją naudojant skirtingus raktus užšifravimui ir iššifravimui.

Priemonės

- 3 dėžutės su dviem užraktais („viešuosius“ ir „privatų“).
- 1 „privatus“ + 2 „viešieji“ raktai kiekvienai dėžutei.
- Etiketės, žymekliai arba kitos priemonės dėžučių ir raktų žymėjimui.
- Lapeliai užrašams, rašymo priemonės.



Priemonių alternatyvos

Įvairius dėžučių, galima naudoti simbolinius „dėžučių“ (tai gali būti vokai, kartoninės dėžutės arba suolemi popieriniai lapai) ir simbolinius „raktus“ (pavyzdžiui, „viešieji raktai“ ir „privatus raktas“ gali būti nedideli spalvoti ir (arba) atitinkamai pažymėti lapeliai).



Veikla (1)

Pasiruošimas:

- Sudaromos trys mokinių grupės, kiekviena grupė gali sugalvoti savo grupės pavadinimą. Mes paprastumo dėlei grupes pavadinsime A, B ir C.
- Kiekviena iš trijų grupių gauna grupės vardą (A, arba B, arba C) pažymėtą dėžutę bei grupės vardą (A, arba B, arba C) pažymėtus raktus – „viešuosius“ ir vieną „privatų“ (jį reikia dar papildomai pažymėti, pavyzdžiui, grupės vardą „A“ arba „B“ arba „C“).

Veikla (2)

Raktų mainai:

- Grupė A perduoda grupėms B ir C po vieną savo „viešąjį“ raktą – šis raktas kitų grupių bus naudojamas tik tam, kad grupės B ir C grupės A dėžutę įdėtų užrašytą „paslaptį“ („užšifruotą“ pranešimą). „Privatų“ raktą (A) grupė A pasilekia sau.
- Tokius pat veiksmus atlieka ir grupės B bei C – perduoda savo „viešuosius“ raktus kitoms grupėms, sau palikdami „privatų“ raktą ir vieną „viešąjį“.

Veikla (3)

„Paslapčių“ įdėjimas ir užrašymas (užšifravimas):

- Grupė A atidaro grupių B ir C dėžutes naudodamiesi atitinkamais tų grupių „viešaisiais“ raktais, įdėda „paslaptį“ (informaciją, kurią užšifruojama ir perduodama kitai grupei, pavyzdžiui, pranešimą „Ši ketvirtadienį mokykloje vyks geografinis naktis. Renkames 20 valandą sporto salėje. Nepamirškite pasiimti vandens ir maisto.“) ir vėl užrakina („užšifruoja“).
- Taip pat elgiasi ir grupės B bei C.

Veikla (4)

„Paslapčių“ atrakinimas (iššifravimas):

- Grupė A naudoja savo „privatų“ raktą (raktą, kurio neatidavė kitoms grupėms, o pasiūlė sau „A“), kad atrinktų savo dėžutę ir raktų kitų grupių įdėtą (atsiųsčius) „paslaptį“ („iššifruoja“ perduotą pranešimą).
- Grupės B bei C daro tą patį.

Veikla (5)

Aptarimas:

- Kiekviena grupė perskaito atsiųstas „iššifruotas paslaptis“ ir visi kartu su mokytoju aptaria, kokie yra atliktos veiklos ir asimetrinio šifravimo žingsnių atitikimai (pavyzdžiui, vienu – viešuoju – raktu užrakiname („užšifruojame“), kitu – privatu – raktu atrakiname („iššifruojame“)).



Užduotis su išsamiau aprašymu pateikta metodinės medžiagos 9–10 klasių pateikties 29–37 skaidrėse.

Viešojo ir privataus rakto sąvokos

Viešasis raktas (Public Key)

Viešasis raktas yra asinchroninio šifravimo kriptografinės raktų poros dalis, kuri yra viešai prieinama. Šis raktas naudojamas šifruoti duomenis arba patikrinti elektroninio parašo autentiškumą. Kiekvienas, turintis viešąjį raktą, gali šifruoti duomenis taip, kad tik asmuo, turintis atitinkamą privatų raktą, galėtų juos iššifruoti.

Privatus raktas (Private Key)

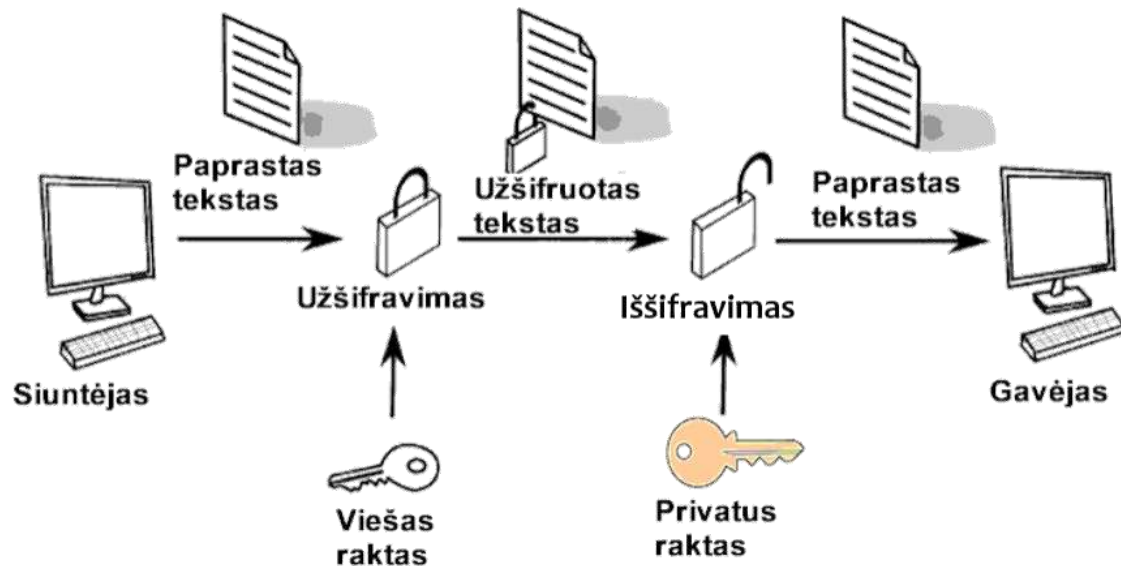
Privatusis raktas yra asinchroninio šifravimo kriptografinės raktų poros dalis, kuri saugoma kaip paslaptis. Šis raktas naudojamas iššifruoti duomenis, kurie buvo užšifruoti naudojant atitinkamą viešąjį raktą, arba sukurti elektroninį parašą. Privačiojo rakto saugumas yra labai svarbus, nes su juo galima atkurti visus užšifruotus viešuoju raktu duomenis ir atlikti kitas saugumo kritines operacijas.

Sertifikato sąvoka

Sertifikatas (Digital Certificate)

Sertifikatas yra elektroninis dokumentas, kuriame patvirtinama viešojo rakto ir identifikatoriaus (pvz., vartotojo vardo ar organizacijos pavadinimo) sąsaja. Sertifikatai išduodami ir patvirtinami pasitikėta trečiaja šalimi, vadinama sertifikavimo centru (CA - Certificate Authority). Sertifikatai naudojami patikrinti asmenų, sistemų ar tarnybų tapatybę internete.

Asimetrinio šifravimo schema



Užduotis. Yra sakoma, kad simetrinio šifravimo atveju sauga ypatingai priklauso nuo šifravimo rakto saugojimo ir jo saugaus perdavimo komunikacijos dalyviui. Pakomentuokite, kodėl šios galimos simetrinio šifravimo spragos iš esmės dingsta naudojant *asimetrinį šifravimą*.

- ❑ Asimetrinio šifravimo metodai yra ypač naudingi saugiam duomenų perdavimui per nesaugius tinklus, tokius kaip internetas, taip pat tapatybės patvirtinimui, elektroniniams parašams ir kitoms panašioms operacijoms.
- ❑ Asimetrinė kriptografija yra plačiai naudojama įvairiose pramonės ir technologijų srityse dėl jos gebėjimo užtikrinti aukštą saugumo lygį ir konfidencialumą.

Asimetrinė kriptografija – taikymas (1)

- ❑ **Interneto saugumas (TLS/SSL protokolai)** – užtikrina saugų informacijos perdavimą tarp interneto naršyklių ir serverių.
- ❑ **Tinklo technologijos (SSH – Secure Shell)** – suteikia saugią prieigą prie nuotolinių serverių.
- ❑ **VPN (VPT – Virtualūs privatūs tinklai)** – naudoja asimetrinę kriptografiją saugiam duomenų perdavimui tarp kliento ir serverio.
- ❑ **Elektroniniai laiškai ir komunikacija (PGP/GPG – Pretty Good Privacy/GNU Privacy Guard)** – šifravimas saugiai perduodant elektronines žinutes.

Asimetrinė kriptografija – taikymas (2)

- ❑ **Duomenų saugojimas, debesijos duomenų saugojimo** paslaugos – kai kurios debesijos paslaugos naudoja asimetrinę kriptografiją failų šifravimui ir prieigos kontrolės mechanizmams.
- ❑ **Skaitmeniniai parašai.**
- ❑ **E. dokumentų patvirtinimas** – dokumentų ir sutarčių skaitmeniniai parašai, patvirtinami naudojant asimetrinę kriptografiją.
- ❑ **Finansinės paslaugos.**
- ❑ **Mokėjimų sistemos** – asimetrinė kriptografija užtikrina, kad piniginės operacijos atliekamos saugiai.

Asimetrinė kriptografija – taikymas (3)

- ❑ **Blockchain ir kriptovaliutos** – naudoja asimetrinę kriptografiją transakcijų autentifikavimui.
- ❑ **Identifikacija ir autentifikacija.**
- ❑ **Skaitmeniniai sertifikatai** – asimetrinė kriptografija naudojama elektroninėse tapatybės kortelėse, vairuotojo pažymėjimuose ir kituose tapatybės dokumentuose.
- ❑ **Įrenginių autentifikacija ir saugus duomenų perdavimas** – dėl ribotų skaičiavimo galimybių ypač svarbu naudoti efektyvius asimetrinės kriptografijos algoritmus.

Asimetrinė kriptografija – taikymas (4)

- ❑ **IoT (Internet of Things)** – daiktų internetas.
- ❑ **Mašininis mokymasis ir duomenų analizė.**
- ❑ **Homomorfinė kriptografija** – leidžia atlikti skaičiavimus su šifruotais duomenimis, kuriuos vėliau galima dešifruoti ir gauti teisingus rezultatus.
- ❑ **Valdžios ir karinės institucijos.**
- ❑ **Ryšių šifravimas, skaitmeniniai sertifikatai** – naudojami saugiai komunikacijai ir duomenų laikymui.
- ❑ **Ir kitos taikymo sritys.**

Programavimo kalbos Python panaudojimo pavyzdys kuriant programą, kuri naudoja asimetrinio RSA (Rivest–Shamir–Adleman) šifravimo biblioteką, kad užšifruotų ir iššifruotų pateiktą tekstą.

Šifravimo programos SIFR_RSA.PY darbo rezultatas

(programos SIFR_RSA.PY kodas pateiktas kitoje skaidrėje)

```

-----BEGIN PUBLIC KEY-----
MIIEIjAMBgqhkiG9w0BQEIAAQCABQgIBCCgKCAQEAk8B0wLF#ZYw3H7xrQbHv
acGTMD3ONvgf19k6e5Yco1ch5TevYNo6aMahaH3xartm1GofS1373K19UG
d5SBrenYK4e17H01adKR2q1G1QVa/66B11sVQREY7bY8B/1VtMk/w+AR2MUEgG2
Q0+Gq/kJ1Idrwk/jwlpVFqbyQKZJesJt00uNoz/p7o+TK71yXNqUQIDKYXq+Qh
Yhu8BV85SRN1u4tptCDSZEPmwv5+Ht14bnfKRXunSCEH1CEBwbZ+QE7gXFz1Ew/
J1eG11dpdhmc2paxmSe2yU14DPAeHohZ1p5JrhxfBRTFWMIQ#LXIZ/4ffqbo7U+6
PwIDAQAB
-----END PUBLIC KEY-----

Privatusis raktas:
-----BEGIN RSA PRIVATE KEY-----
MIIEowTAAKCAQEAg8DalFe2Yw3H7xrQbHvYwGTMD3ONvgf19k6e5Yco1ch5Tev
YNo6aMahaH3kxmt1GofS1373K19UGd5SBrenYK4e17H01adKR2q1G1QVa/
66B11sVQREY7bY8B/1VtMk/w+AR2MUEgG2Q0+Gq/kJ1Idrwk/jwlpVFqbyQKZJ
esJt00uNoz/p7o+TK71yXNqUQIDKYXq+QhYhu8BV85SRN1u4tptCDSZEPmwv5
+Ht14bnfKRXunSCEH1CEBwbZ+QE7gXFz1Ew/J1eG11dpdhmc2paxmSe2yU14DPA
eHohZ1p5JrhxfBRTFWMIQ#LXIZ/4ffqbo7U+6PwIDAQABAAIABAN/KKPAw1A5tKM22
g7M3KyT1M8hze2Nakb3e0V2H1c851cMAG1KBc33Tcm973Ez+80WbzmgcVH633F
wChuLumfj1xwM2BvJno047y72QwX69Kcu1B8rhcz+gETC0rH51yxXp32d83Kyf/
xbZFA5B0x+d46g17xt1x1+Mhxy7L7B+7VycQvab1JctHf45q3jba1Fc+XQc0IBZ
k8+Zr75ELJsJw05uayf1Jxc0A106Pg1B0K80F5Y3a5vVWKTtqg4w1HMAmapCKH
sMBQjbr3j7fVn5TGRa4d66pFVMA6pC1J3Gaf/rHIT+0DBusahg541xgbl+R73C
Q19ozgECgYEAzQd1p4Uv1tmvcen9kn32knpZebOPdAnt39E4HeqmxVLQnE7Ur
en0deuQv+151832+Xsr5Ldd1p3B6F1Xh2gXMMN0WfYhu+/fCLv8e6FFJmJ1T8H0
uNS1nzK31rj7Z8hrKPCyn3dkqE4x4sSol1utpK51s9msFQT3Rn1k3cEGYEAcmW
PM58vPa0nAE86ng8PuQKxrf415wXV171wYjy8e/m110/1Re6+R5x6W1WczuUxyx
tJ5da08Gvrao1pf08kplbyVTc0UyVLR1M1uM9SR0mfQBA6C11h0Wm4hgn17ndTD
B93J517r8CazTbQRL1caU1cKv9LPSBCE152K1/BcYgBXH/522t6721PF+pPP3s8K
Am6+2WYmfde0jJ79ppwF7dUZXhaC/j3sFo1J/X0KruX+UC+415/LV5g0bd01U9
Se7BFAena1Uyrh1NA2UEVA6VJd/f09Aqcas3dn3GzQXHZ304yH1V/kyuFMY34
cAvTBKb6amf0uV4g11uWAKQBgQCXnuZE10EMjwJ5/4q9e5zcnyes3n09NgJusNP
//115sefj1hVpBepC2G0o30n5blkkCfE8CBwnwX25GdBoNspEYxx4+e8s351uY4
EHgK0K8nO189Y7V/c8d8BxsZ8M1KLmPQ123dVWkXQUPU3wCQqQRuHEdYTY6
y0DY2xK8Gw7y04g5v8B8Hh1sGaou94q1kY7ktuY5qsh700t807Yz13d1Ecngw0
WdLET3Cn/c8C356rdbvWnc1wVtBT+ry236PLUdanyNg24N1kARq3B3uaj1Vb
c4HG0dyKprkqRhuLr6qeh3u0fwrhoJ2FMQza7LLJH0Z1+9wV1Q
-----END RSA PRIVATE KEY-----

Uffisfurota 2!notu6: b'\\x157p8\\xacc8B\\x8B\\x0b\\x912\\xae2\\xb4\\xb1\\xc9\\xbd\\x9f\\x5d\\xae2-\\xc35\\xff\\xae3\\xc1\\xb0Q\\x99\\xbf\\x95\\xda\\xae\\xf1\\xae6\\xec\\xbf\\x127\\xaea\\x0\\xf6\\xce\\x8d\\xc2g\\x9a\\xb1t\\x97\\x13Q\\xf7m\\x8\\xa2P\\x8f\\xac\\xbfy\\x9f\\xae6\\xf3f\\xae
B0K\\xc1a\\x9b\\x8f\\x8a\\x0d21n\\x9P\\xb8\\xae2H\\xf6\\x0c6\\xb489f\\xb8\\x07\\x08\\xb6\\xd9d\\x39FDka\\xc2\\x112\\xadda\\x07e\\xb8\\xc3e\\xf4f\\x3942v\\t5\\x08\\xaa\\xb8\\xae3\\xa8\\xaps\\xf4\\x86\\xa0f\\xb9\\xb0g\\xb7\\xd5\\xf7\\x9B7b\\xadd\\x85\\xo13\\x18f\\xb7\\x09Q6b863V\\xafQ\\x
912\\xae2\\xb8\\x97\\xd3d\\xae2\\x0d\\xc1\\x12\\xbdc\\x08\\xb83\\x88Bh3\\xbdb1+\\x95\\x0f5\\xafW6\\xbfbf\\x7b9-U\\x92\\xc8\\xf6\\xb0\\x1b0\\xb4\\xb3\\xbaf2\\xb3Q\\xde\\xb9\\xb9e\\xf4f\\xae9\\xb0V\\xad\\xbcb\\x55\\xaa0\\xc0c8K\\xf4f\\xa11\\x1c1\\xc85B\\x9f_1\\xc0a\\xc18K\\x0d\\x91\\xb0\\x116q
```


Asimetrinis šifravimas naudojant Python kalba sudarytą programą SIFR_RSA.PY

(prieš naudojant įdiekite biblioteką – komandų eilutėje nurodykite `pip install cryptography`)

```
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives import serialization, hashes
from cryptography.hazmat.primitives.asymmetric import rsa,
padding
```

```
# Generuojame RSA privatuji ir viešąjį raktą
private_key = rsa.generate_private_key(
    public_exponent=65537,
    key_size=2048,
    backend=default_backend()
)
public_key = private_key.public_key()

# Eksportuojame viešąjį raktą PEM formatu
public_pem = public_key.public_bytes(
    encoding=serialization.Encoding.PEM,
    format=serialization.PublicFormat.SubjectPublicKeyInfo
)
print("Viešasis raktas:")
print(public_pem.decode("utf-8"))
```

```
# Eksportuojame privatuji raktą PEM formatu
private_pem = private_key.private_bytes(
    encoding=serialization.Encoding.PEM,
    format=serialization.PrivateFormat.TraditionalOpenSSL,
    encryption_algorithm=serialization.NoEncryption()
)
```

```
print("Privatusis raktas:")
print(private_pem.decode("utf-8"))
```

```
# Užšifruojame žinutę su viešuoju raktu
original_message = b"Svarbi informacija"
ciphertext = public_key.encrypt(
    original_message,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)
print("Užšifruota žinutė:", ciphertext)
```

```
# Iššifruojame žinutę su privačiuoju raktu
decrypted_message = private_key.decrypt(
    ciphertext,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)
print("Iššifruota žinutė:", decrypted_message.decode('utf-8'))
```

Kurį pasirinkti?

- ❑ **Simetrinis šifravimas** yra greitas ir paprastas, bet kyla problemų, kai reikia saugiai pasidalyti ir saugoti raktus.
- ❑ **Asimetrinis šifravimas** šias problemas sprendžia, bet yra lėtesnis ir reikalauja daugiau kompiuterio resursų.

Kiekvienu atveju reikia pasirinkti labiausiai tinkamą šifravimo strategiją arba šifruojant informaciją derinti abi galimybes – simetrinį šifravimą su asimetriniu.

Simetrinio ir asimetrinio šifravimo derinimo pavyzdys (1)

- ❑ Kadangi asimetrinė kriptografija yra daug lėtesnė nei simetrinė kriptografija, ji retai naudojama dideliems duomenų kiekiams tiesiogiai užšifruoti.
- ❑ Vietoj to, asimetrinis šifravimas paprastai naudojamas tokiu būdu simetriniams raktams užšifruoti.

Simetrinio ir asimetrinio šifravimo derinimo pavyzdys (2)

- ❑ Agnė ir Benas dažnai susirašinėja rengdami bendrą projektą.
- ❑ Išnagrinėkite Agnės ir Beno, kurie keičiasi informacija, susitarimus ir veiksmus:
 - Agnė reikalauja, kad Benas el. paštu siųstų jai tik užšifruotas žinutes.
 - Ji sukuria privataus/viešojo rakto porą, saugo savo privatų raktą paslapyje ir paskelbia savo viešąjį raktą.
 - Benas turi žinutę, kurią nori nusiųsti Agnei.
 - Jis sukuria naują simetrinį raktą ir šį raktą panaudoja, kad užšifruotų savo pranešimą Agnei.
 - Po to Benas naudoja Agnės viešąjį raktą, kad užšifruotų savo simetrinį raktą.

(tęsinys kitoje skaidrėje) →

Simetrinio ir asimetrinio šifravimo derinimo pavyzdys (3)

- Benas siunčia užšifruotą pranešimą ir užšifruotą simetrinį raktą Agnei (toks raktas vadinamas **suvyniotu** arba **apgaubtu** raktu).
 - Agnė naudoja savo privatų raktą (iš privačios/viešos poros), kad iššifruotų simetrinį Beno raktą.
 - Po to ji naudoja simetrinį Beno raktą, kad iššifruotų pranešimą.
- Kaip manote, ar pakankamai saugiai buvo perduotas Agnei Beno sukurtas simetrinis šifravimo raktas?
 - Aptarkite tai bendroje grupės diskusijoje.

(adaptuota pagal <https://learn.microsoft.com/>)