Contents

[Uvod 1](#_Toc26914551)

[Programski jezik Python 1](#_Toc26914552)

[JetBrains PyCharm razvojno okruzenje 1](#_Toc26914553)

[Tkinter 1](#_Toc26914554)

[PyCryptoDome 2](#_Toc26914555)

[Desktop aplikacija – Chat aplikacija 3](#_Toc26914556)

[Server 3](#_Toc26914557)

[Klijent 6](#_Toc26914558)

[GUI - Tkinter 8](#_Toc26914559)

# Uvod

Ovde će biti opisane tehnologije koje su korišćene u pravljenju chat aplikacije sa delimično zaštićenom komunikacijom.

## Programski jezik Python

Python je interpretirani programski jezik visokog nivoa opšte namene. Napravio ga je Guido van Rosum i prvi put objavljen 1991. godine, Python-ova filozofija dizajna naglašava čitljivost koda uz primetno korišćenje praznog prostora. Njegove jezičke konstrukcije i objektno orijentisani pristup imaju za cilj da pomognu programerima da napišu jasan, logičan kod za male i velike projekte.

Python je dinamički jezik sa garbage collector-om. Podržava više paradigmi programiranja, uključujući proceduralno, objektno orijentisano i funkcionalno programiranje. Python se često opisuje kao jezik sa "baterijama" zbog sveobuhvatne standardne biblioteke.

Tumači Python-a su dostupni za mnoge operativne sisteme. Globalna zajednica programera razvija i održava CPython. Neprofitna organizacija, Python Softvare Foundation, upravlja i usmerava resurse za razvoj Python-a i CPython-a.

## JetBrains PyCharm razvojno okruzenje

PyCharm je integrisano razvojno okruženje (IDE) koje se koristi u računarskom programiranju, posebno za jezik Python. Razvila ga je češka kompanija JetBrains. Omogućava analizu koda, grafički program za otklanjanje grešaka, integrisani uređaj za ispitivanje jedinica, integraciju sa sistemima kontrole verzija (VCSes) i podržava web razvoj sa Djangom kao i Data Science sa Anacondom. PyCharm je cross-platforma, sa Windows, macOS i Linuks verzijama. Community Edition je izdat pod licencom Apache, a tu je i Professional Edition sa dodatnim funkcijama - objavljen pod vlasničkom licencom.

## Tkinter

Tkinter je Python alat za pravljenje Tk GUI-a. To je standardni Python interfejs za Tk GUI, i Python-ov je de facto standardni GUI.

Tkinter je uključen u standardne instalacije Pythona za Linuks, Microsoft Windows i Mac OS. Ime Tkinter dolazi iz Tk interfejsa. Tkinter je napisao Fredrik Lundh.

Tkinter je besplatni softver objavljen pod licencom Pithon.

## PyCryptoDome

PyCryptodome je samostalni Python paket kriptografskih primitiva niskog nivoa.

PyCryptodome je nadogradnja PyCripto-a koja je poboljšana kako bi se dodalo više implementacija i ispravki u originalnu PyCrypto biblioteku. Tamo gde je to moguće, većina algoritama u ovoj biblioteci je implementirana u čistom Python-u; samo komadi koji su izuzetno kritični za performanse (npr. blok šifre) su implementirani kao C ekstenzije.

Biblioteka PyCryptodome nudi implementacije za stvari poput:

* AES
* Stream ciphers kao Salsa20
* Kriptografske heševe kao što su SHA-2
* Kodove za autentikaciju poruka kao što su HMAC
* RSA asimetrično generisanje ključeva

# Desktop aplikacija – Chat aplikacija

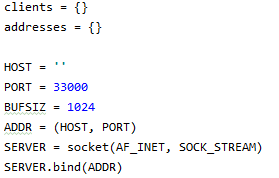
Chat aplikacija je aplikacija koja omogućuje instant razmenu poruka na bilo koju udaljenost. Program omogućuje korisniku da pokrene aplikaciju, izabere ‘nadimak’ i razmenjuje poruke sa ostalim članovima koji su nakačeni. Objasnićemo kako funkcioniše server i kako klijent, te detaljnije kako je implementirano šifrovanje. U ovom radu svi kodovi su pisani u programskom jeziku Python.

## Server

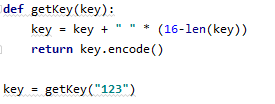
Koristićemo TCP vezu i zato koristimo *AF\_INET i SOCK\_STREAM*. Koristimo TCP pre nego UDP veze, jer su obostranije, gde primalac mora da odobri dolaznu vezu pre nego što komunikacija započne, a UDP veze su nešto kao pošta (svako može da pošalje poštu bilo kom primaocu čiju adresu on zna), pa oni zaista ne zahtevaju uspostavljanje veze pre nego što se komunikacija može dogoditi. Jasno je da TCP više odgovara našoj svrsi nego UDP, zato ga i koristimo.

Screenshot_2.png

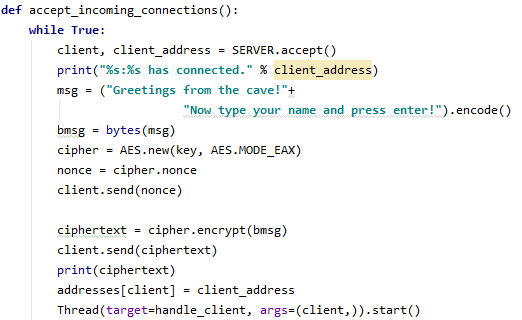
Nakon uvoza postavili smo neke konstante za kasniju upotrebu:



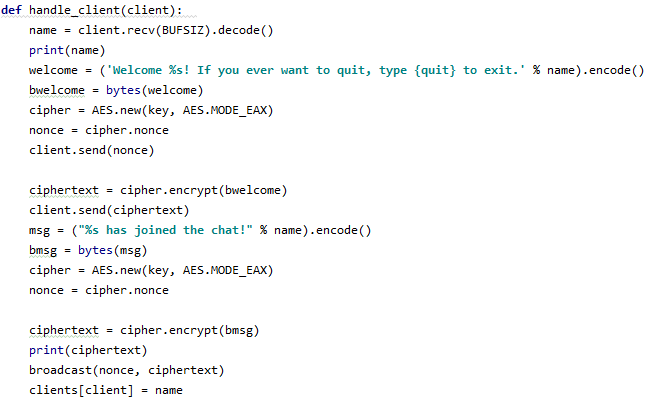
Sada definišemo ključ koji će se koristiti u šifrovanju i dešifrovanju. Ključ ne sme biti veći od 16 bajta.

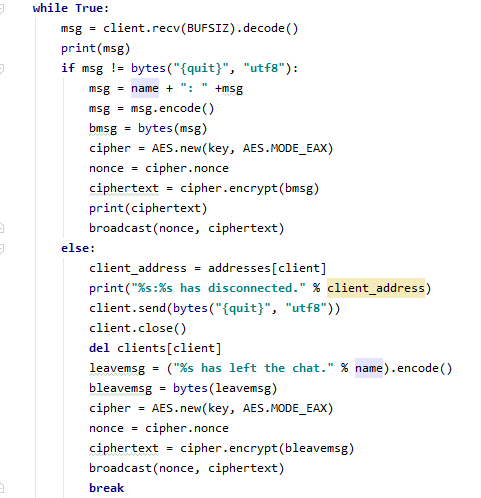


Sada razbijamo zadatak na prihvatanje novih veza, slanje poruka i rukovanju klijentima. Počnimo sa prihvatanjem veza:



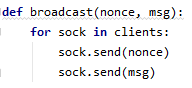
Ovo je samo petlja koja zauvek čeka dolazne veze i čim je uspostavi, registruje vezu (ispisuje neke detalje veze) i šalje klijentu poruku dobrodošlice, koja se pre slanja šifruje korišćenjem simetrične šifre. Zatim smešta adresu klijenta u rečnik adresa i kasnije pokreće nit za rukovanje za tog klijenta. Naravno, još uvek nismo definisali ciljnu funkciju(target) handle\_client (), ali evo kako to radimo:





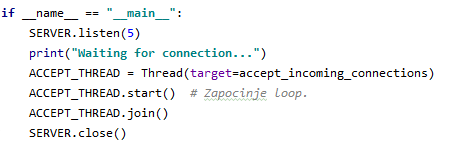
Naravno, nakon što novom klijentu pošaljemo poruku dobrodošlice, on će odgovoriti imenom koje želi da koristi za dalju komunikaciju. U funkciji handle\_client (), prvi zadatak koji radimo je da sačuvamo to ime, a zatim pošaljemo još jednu poruku klijentu(koja se takođe šifruje na isti način kao prošla poruka), u vezi daljih uputstava. Nakon toga dolazi glavna petlja za komunikaciju: ovde primamo dalje poruke od klijenta i ako poruka ne sadrži uputstva za odustajanje, jednostavno prenosimo poruku drugim povezanim klijentima (definisaćemo način emitovanja u trenu ). Ako naiđemo na poruku sa uputstvima za izlazak (tj. klijent šalje {quit}), istu poruku vraćamo klijentu (ona pokreće radnju zatvaranja na strani klijenta), a zatim zatvaramo priključnu utičnicu za nju. Potom obavimo neko čišćenje brisanjem unosa za klijenta i napokon dojavljujemo ostalim povezanim ljudima da je ta osoba napustila razgovor.

Sada je na redu broadcast() funkcija:



Ovo jednostavno šalje nonce(koji je potreban za dešifrovanje) i poruku svim povezanim klijentima.

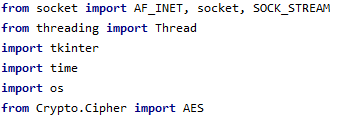
Na kraju smo uneli neki kod za pokretanje našeg servera i preslušavanje dolaznih veza:



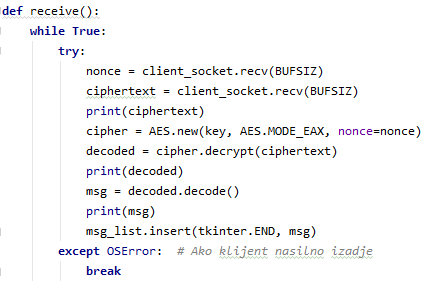
Pridružujemo ACCEPT\_THREAD() tako da glavna skripta čeka da se dovrši i ne pređe na sledeću liniju, jer bi to ugasilo server server.

## Klijent

Koristimo Tkinter, Python-ov GUI alat. Ali prvo da uvozemo šte je potrebno:

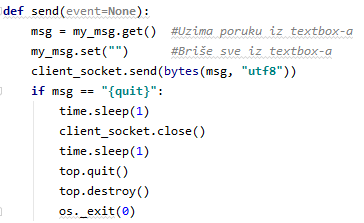


Sada ćemo pisati funkcije za rukovanje slanjem i primanjem poruka. Počinjemo sa primanjem:



Opet imamo beskonačnu petlju jer ćemo primati poruke prilično nasumino i nezavisno od toga kako i kada šaljemo poruke. Ne želimo da ovo bude voki-toki aplikacija za ćaskanje koja može istovremeno da šalje ili prima; želimo da primimo poruke kada možemo i šaljemo ih kada želimo. Funkcionalnost unutar petlje je prilično jasna; recv () je deo koji blokira. Izvršenje se zaustavlja dok se ne primi poruka, a kada se to dogodi, idemo napred i dešifrujemo poruku koristeći AES dešifrovanje i nakon toga dodajemo poruku u msg\_list. Uskoro ćemo definisati msg\_list, koji je u osnovi Tkinter funkcija za prikazivanje liste poruka na ekranu.

Nakon toga definišemo funkciju send ():



Koristimo event kao argument, jer ga Tkinter implicitno prenosi kada se pritisne dugme za slanje na GUI-u. my\_msg je polje za unos na GUI-u i zato izdvajamo poruku koja će biti poslata u msg= my\_msg.get (). Nakon toga čistimo polje za unos i zatim šaljemo poruku serveru, koja, kao što smo videli, emituje ovu poruku svim klijentima (ako to nije izlazna poruka). Ako je izlazna poruka, zatvaramo soket, a zatim aplikaciju GUI.

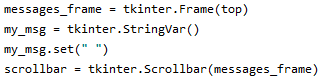
.

## GUI - Tkinter

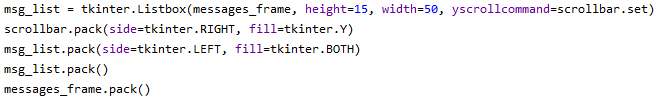
Sada započinjemo izgradnju GUI-ja, u main-u. Započinjemo definisanjem vidgeta najvišeg nivoa i postavljanjem njegovog naslova:

Screenshot_15.png

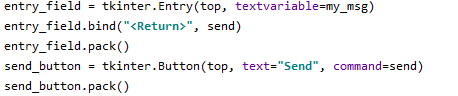
Onda kreiramo okvir za držanje liste poruka. Zatim kreiramo varijablu niza, prvenstveno za čuvanje vrednosti koju dobijemo iz polja za unos (koju ćemo uskoro definisati). Nakon toga kreiramo traku za pomicanje za kretanje kroz ovaj okvir za poruke:



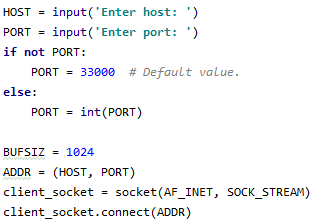
Sada definišemo listu poruka koja će biti smeštena u messages\_frame i zatim spakujemo (na odgovarajućim mestima) sve stvari koje smo do sada stvorili:



Nakon toga kreiramo polje za unos da korisnik upiše svoju poruku i vežemo ga na varijablu niza definisanu gore. Takođe ga vezujemo za funkciju send () tako da kad god korisnik pritisne Enter, poruka se šalje serveru. Zatim kreiramo dugme za slanje ako korisnik želi da pošalje svoje poruke klikom na njega. Opet, vezujemo klikom na ovo dugme na funkciju send (). I da, takođe treba da spakujemo sve ove stvari koje smo stvorili upravo sada.



Još nismo napisali kod za povezivanje sa serverom. Za to moramo pitati korisnika za adresu servera. To je učinjeno jednostavnim korišćenjem input (), tako da je korisnik dočekan sa command prompt-om koji traži adresu domaćina pre nego što GUI počne.



Nakon što dobijemo adresu i napravimo soket da se na nju povežemo, pokrećemo nit za primanje poruka, a zatim glavnu petlju za našu GUI aplikaciju:

Screenshot_21.png

1. **Zaključak**

U radu smo predstavili aplikaciju koja služi za slanje instant poruka. Komunikacija od servera ka klijentima je zaštićena, tj. šifrovana, dok je komunikacija od klijenata ka serveru nezaštićena.

Moguća poboljšanja koja se mogu uraditi su:

1. Bolji i lepši GUI
2. Jasnije konektovanje klijenta na server
3. Skrolbar ne radi

**Reference**

1.

Python,

<https://www.python.org/doc/>

2.

PyCharm,

[https://www.jetbrains.com/pycharm/](https://www.jetbrains.com/pycharm/%20)

(Pristupljeno 10.12.2019. godine)

3.

Tkinter

[https://docs.python.org/2/library/tkinter.html](https://docs.python.org/2/library/tkinter.html%20)

(Pristupljeno 12.10.2019. godine)

4.

PyCryptoDome

<https://pycryptodome.readthedocs.io/en/latest/>

(Pristupljeno 09.12.2019. godine)