Protokół serwera upload/download plików

Autorzy: Paweł Kowalczyk i Dominik Dudek

Informacje wstępne

Serwer służy do przechowywania plików. Po zalogowaniu, klient może wysłać na serwer dowolny plik. Klient ma możliwość pobrać swoje pliki oraz pliki innych użytkowników które są publicznie dostępne. Połączenie jest szyfrowane asymetrycznie.

Protokół

Każda wiadomość wysyłana pomiędzy klientem a serwerem zakończona jest ”\r\n\r\n”.
Format przesyłanych danych pomiędzy serwerem a klientem wygląda następująco:

```
class Main:
    def __init__(self, typ, length, content):
        self.typ = typ
        self.length = length
        self.content = content
```

Gdzie:
typ: nazwa klasy przesyłanej w „content”
length: długość pola content
content: zawiera klasę

nazwy klas obiektów jakie mogą być przesyłane w polu „content” oraz wartości jakie może przyjąć pole „typ” to:
„login”, ”logout”, „register”, „download_file”, „send_file”, „ls”.

Wartości jakie mogą przyjąć obiekty poszczególnych klas:

Login	
username	Nazwa użytkownika
password	Hasło

Logout	
username	Nazwa użytkownika
sid	Identyfikator sesji

Register	
username	Nazwa użytkownika
password	Hasło

Ls	
username	Nazwa użytkownika
own	Pole przyjmuje wartość True/False w zależności czy chcemy wyświetlić własne pliki
public	Pole przyjmuje wartość True/False w zależności czy chcemy wyświetlić publicznie dostępne pliki
files_list	Pole służące do przechowywania listy z nazwami plików

Download_file	
filename	Nazwa pliku
username	Nazwa użytkownika

Send_file	
filename	Nazwa pliku
username	Nazwa użytkownika
ispublic	Wartość True/False wskazuje czy plik ma być publicznie dostępny na serwerze
Size	Wielkość przesyłanego pliku

Ponad to pole „typ” może przyjąć wartość „service_code” a w polu „content” znajdzie się kod błędu. Kody serwisowe są wysyłane tylko z serwera do klienta w celu poinformowania o zrealizowaniu lub nie zrealizowaniu jego żądania.

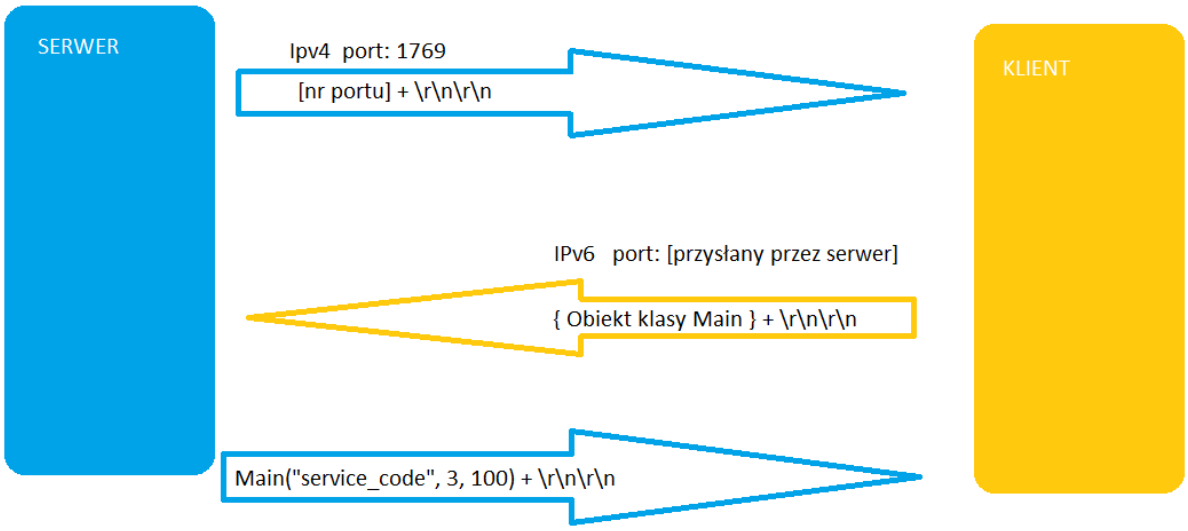
Np.

Main	
typ	service_code
length	3
content	100

Zaraz po uruchomieniu

Po uruchomieniu programu serwer tworzy socket IPv4 i nasłuchuje na porcie 1769, połącznie jest szyfrowane SSL. Gdy klient nawiąże polaczenie serwer uruchamia funkcję obsługującą go w dzielnym wątku. Pierwszą wiadomość jaką serwer wysyła klientowi to numer losowego portu. Wiadomość ta nie jest opakowywana w żadną klasę. Jest to po prostu numer portu zakodowany w formacie UTF-8.

Następnie serwer zamyka polaczenie i tworzy nowy socket IPv6 i nasłuchuje na wcześniej wylosowanym porcie. Połączenie jest szyfrowane SSL. Po połączeniu z klientem serwer czeka na żądanie od klienta. Serwer rozpoznaje rodzaj żądania klienta za pomocą pola „typ”. Dzięki temu wie gdzie przekazać dalej obiekt zawarty w polu „content”.



Klient na podstawie kodu informuje użytkownika.

Service_code	
100	"Successfully"
101	"Successfully Logged in"
102	"Successfully Logged out"
103	"Username available"
104	"Registration successful"
201	"Incorrect password"
202	"Incorrect username"
203	"Username unavailable"
204	"Unauthorized logout!"
205	"User not logged in!"

206	"Operation cancelled. Illegal username."
207	"Login error."
301	"Successfully. Start sending file. "
401	"File not exist.",
402	"Permission denied."

Przesyłanie pliku

Klient wysyła obiekt klasy „download_file”, w polu „content” klasy Main. Obiekt klasy „download_file” zawiera informacje na temat pliku który użytkownik chce pobrać z serwera.
Serwer odsyła service_code 301 i wysyła plik w postaci binarnej. Plik nie jest opakowywany w żadne klasy.