D o k u m e n t a c j a p r o j e k t u

Tytuł projektu	Program tracepath/traceroute w postaci biblioteki	
Wykonawcy	Patryk Janowski	Jan Czernecki

1. Sposób i cel użycia:

Program służy do badania trasy pakietów w sieciach IP. Przyjmuje nazwę docelowego hosta i opcję związaną z protokołem jako argumenty. Umożliwia wybór protokołu do swojego działania pomiędzy ICMP oraz UDP.

Aby sprawdzić działanie programu należy uruchomić plik *test.py* z podaniem nazwy hosta docelowego oraz ewentualnie opcji -i (lub --send_over_icmp) jeśli chcemy wykorzystać protokół ICMP. W przeciwnym razie użyte zostaną puste pakiety UDP.

2. Struktura projektu:

Projekt składa się z trzech plików *IcmpLib.py*, *TracerouteLib.py* oraz *test.py*, które zawierają kod w języku *Python3*.

Funkcje *IcmpLib.py*:

- **findChecksum** - Oblicza sumę kontrolną dla pakietu ICMP. Wymagana do sprawdzenia integralności pakietu.
- **createPacket** - Tworzy pakiet ICMP korzystając z identyfikatora, numeru sekwencji oraz przesyłanych danych. Zwraca stworzony nagłówek ICMP połączony z przesyłanymi danymi.
- **buildEchoRequestPacket** - Tworzy pakiet typu EchoRequest korzystając z funkcji createPacket z wylosowanym numerem pakietu i pustymi przesyłanymi danymi. Zwraca gotowy pakiet typu ICMP Echo Request.
- **getIcmpSocket** - Tworzy gniazdo surowe korzystające z rodziny adresów AF_INET i protokołu ICMP. Ustawia timeout i przypisuje gniazdo do dostępnego portu. Zwraca skonfigurowane gniazdo.
- **setSockTTl** - Ustawia przekazaną w argumencie wartość pola Time To Live dla gniazda przekazanego w argumencie.
- **sendEchoRequest** - Wysyła pakiet ICMP Echo Request na adres wskazany w argumencie korzystając z wybranego gniazda.

Funkcje *TracerouteLib.py*:

- **tracerouteUtil** - Implementacja funkcjonalności traceroute. W zależności od ustawienia zmiennej *send_over_icmp* korzysta z protokołu ICMP lub UDP. Funkcja jak argument przyjmuje adres docelowy. Zwraca generator zwracający obecny adres i upłynięty czas dla każdego węzła.
- **printHeader** – Wypisuje nagłówek programu.
- **traceroute** – Wywołuje odpowiednie funkcje i prezentuje ich wyniki w odpowiedni sposób. Przyjmuje adres docelowy jako argument.

Program *test.py* służy do prezentowania działania dwóch poprzednich bibliotek. Obsługuje też przyjmowanie argumentów z linii komend.

3. Wykorzystywane moduły:

- ***socket***: Moduł został wykorzystany do tworzenia gniazda ICMP, tworzenia gniazda UDP, ustawiania wartości TTL oraz pobierania nazw hostów.
- ***struct***: Moduł został wykorzystany do tworzenia nagłówków pakietów ICMP.
- ***random***: Moduł został wykorzystany do generowania losowych identyfikatorów pakietów ICMP.
- ***time***: Moduł został wykorzystany do obliczania czasów odpowiedzi dla poszczególnych węzłów przy użyciu funkcji *perf_counter_ns()*.
- ***collections***: Z modułu wykorzystano *Generator*, który zwraca iterator z sekwencją wartości. Pozwolił zredukować długość kodu i zwiększyć jego przejrzystość.
- ***contextlib***: Z modułu wykorzystano *ExitStack*, który automatycznie zamyka wszystkie otwarte pliki na końcu swojego działania. Pozwolił zredukować długość kodu i zwiększyć jego przejrzystość.
- ***argparse***: Moduł został użyty do obsługi argumentów wiersza poleceń, czyli nazwy docelowej i opcji decydującej o użytym protokole.

4. Źródła:

- https://pl.wikipedia.org/wiki/Traceroute
- https://www.slashroot.in/how-does-traceroute-work-and-examples-using-traceroute-command
- https://rednafi.com/python/implement_traceroute_in_python/#writing-a-crappier-version-of-traceroute-in-python
- https://www.youtube.com/watch?v=xW_ALxfop7Y&ab_channel=CodingTech
- https://www.speedguide.net/port.php?port=33434