Gabriela Czarska
Sieci Komputerowe

TFTP

Projekt TFTP składa się z dwóch części. Plik *tftp_client7440.py* przyjmuje dwa parametry z lini poleceń nazwę serwera z którym ma się połaczyć oraz nazwę pliku który ma pobrać. Następnie pobiera zadany plik i zapisuje go w bieżącym folderze jako 'message.txt' oraz wypisuje sumę kontrolną md5. Plik *tftp_server7440.py* przyjmuje port(6969) na którym ma nasłuchiwać oraz ścieżkę do folderu z którego ma wysyłać pliki.

Klient

Klient początkowo otwiera plik message.txt w trybie 'a' aby móc do niego dopisywać w miarę otrzymywania kolejnych paczek. Następnie wysyła RRQ do servera, aż otrzyma OACK lub pierwszy datagram. Jeżeli przekroczy ustalloną ilość prób (MAXRETRY) poddaje się. Jeśli otrzymał pierwszy datagram oznacza to, że będzie nadawał w trybie bez 'windowsize', jesli otrzymał oack to z.

Tryb zwykły:

Początkowo sprawdzane jest czy otrzymany datagram nie jest ostatnim, jesli jest to oznaczamy to specjalną flaga (rec_flag). Następnie dopisujemy do pliku oraz sumy kontrolnej. Wysyłamy potwierdzenie oraz oczekujemy na kolejne datagramy. Jeśli otrzymamy zły datagram to ponawiamy wysłanie poprzedniego ack. Jesli otrzymamy dobry datagram to updatujemy wszystko i sprawdzamy czy aby nie jest to ostatni. Jeśli przerwie nam timeout to o ile nie przekroczylismy ilości prob, ponawiamy ack.

Tryb windowsize:

W tym przypadku ack bedziemy wysylac gdy cos pojdzie nie tak/ po otrzymaniu zadanej liczby datagramow/ na poczatku ack0. Do odliczenia czy juz jest moment w ktorym powinnismy wyslac potwierdzenie służy licznik blocks_counter. Oczekujemy na datagram i sprawdzamy czy to ten na ktory czekaliśmy. Jeśli nie to odsylamy ack z potwierdzeniem ostatniego datagramu jaki otrzymalismy. Jesli datagram jest ok to aktualizujemy wszystko i sprwdzamy czy nie jest to ostatni datagram. Jesli przerwał nam timeout to odsylamy ten ack co poprzednio.

Na końcu wipisujemy sumę kontrolną oraz zamykamy plik.

Server

Początkowo server czeka na request ze strony klienta. Z niego wyczytuje nazwę pliku który ma serwować oraz po ilu paczkach ma pojawiać się ack. Następnie przygotowywuje plik który bedzie wysylal. Tworzy pierwszy datagram i sprawdza czy nie jest to ostatni. Następnie tworzy Oack i wysyla je az nie otrzyma ack0. Następnie rozpoczyna się wysylanie pliku. sended_counter jest zmienna odpowiadającą za to kiedy powinniśmy oczekiwac na potwierdzenie. Jesli nadejdzie taki moment to aktualizujemy last_ack(trzyma numer ostatniego potwierdzenia jakie otrzymalismy) oraz sprawdzamy czy to potwierdzenie ktorego sie spodziewalismy. Jesli nie to ponownie wysylamy datagramy zaczynajac od miejsca miejsca gdzie dostlismy potwierdzenia. Gdy ack jest ok, lub gdy nie musimy na nie czekac na ack to wysylamy datagram i sprawdzamy czy przypadkiem nie jest juz koniec. Następnie aktualizujemy wszystko, tworzymy nowy datagram i sprawdzamy czy jeszcze jakis zostal nam do wyslania. Gdy przerwie nam timeout rowniez cofamy sie z wysylaniem do momentu gdzie wiemy ze ta część na pewno doszła.

Na końcu zamykamy plik.