1. Wprowadzenie

1.1. Komunikacja w warstwie transportowej

Protokoły UDP i TCP są protokołami występującymi warstwy transportowej modelu OSI, umożliwiają one komunikację pomiędzy **aplikacjami** działającymi na różnych hostach sieci . Komunikacja taka może zostać podzielona na połączeniową (ang. connection-oriented) i bezpołączeniową (ang. connectionless).

Komunikacja połączeniowa wymaga poprzedzenia procesu transmisji danych etapem połączania, w trakcie którego zostaje zestawione połączenia (które w przypadku sieci komputerowych ma charakter wirtualny) pomiędzy dwoma punktami końcowymi. Cała komunikacja między hostami odbywa się wtedy w ramach zestawionego połączenia. Połączenie po zakończeniu transmisji musi zostać zamknięte.

W przypadku komunikacji bezpołączeniowej, komunikaty mogą być transmitowane w dowolnym czasie, przerwa pomiędzy kolejnymi komunikatami może mieć dowolną długość, ale też nie ma informacji o dostarczeniu komunikatu do odbiorcy.

Komunikacja może również zostać rozróżniona ze względu na wiarygodność dostarczenia informacji, na podstawie tego kryterium można mówić o komunikacji niezawodnej (ang. reliable) oraz zawodnej (ang. unreliable). Pierwszy rodzaj komunikacji kontroluje proces transmisji i w razie wystąpienia błędów, niedostarczenia pakietów wprowadza mechanizmy umożliwiające retransmisję zagubionych, niedostarczonych lub błędnych komunikatów. W przypadku transmisji protokołami zawodnymi, nie ma żadnej kontroli nad transmitowanymi pakietami, które mogą zostać błędnie dostarczone, dostarczone nie w kolejności wysłania. W tym przypadku cały nadzór nad procesem komunikacji spoczywa na aplikacji warstwy wyższej korzystającej z zawodnych protokołów komunikacyjnych.

W warstwie transportowej istnieje mechanizm umożliwiający rozróżnienie do której aplikacji ma zostać dostarczony przesyłany komunikat. Protokoły UDP i TCP dysponują niezależnymi numerami – tzw. portami. Które umożliwiają niezależną od systemu operacyjnego pracującego na danym hoście identyfikację procesu docelowego i nadawczego komunikatu.

1.2. Protokół UDP

Protokół UDP (ang. User Datagram Protocol) jest protokołem warstwy transportowej modelu OSI. Tak jak protokół warstwy sieci umożliwia rozpoznawanie hosta docelowego w sieci, tak protokoły warstwy transportowej umożliwiają rozpoznanie docelowej aplikacji działającej na hoście. Protokół UDP pełni rolę usługi, która zapewnia programom możliwość wysyłania i odbierania niezależnych komunikatów, z których każdy jest przenoszony w oddzielnym datagramie.

Rozwiązania UDP bazują na rozwiązaniach komunikacji bezpołączeniowej, nie jest nawiązywane połączenie przed wysłaniem datagramu, nie ma informacji o zakończeniu wymianie danych, nie ma informacji o stanie wysłanych danych i nie ma informacji sterujących czy reagujących na błędy transmisji.

Komunikaty UDP nie są defragmentowane podczas przesyłania przez sieć, stąd maksymalny możliwy rozmiar komunikatu UDP organiczna maksymalny rozmiar pola danych pakietu IP w którym są enkapsulowane. Może się tak zdarzyć, że MTU będzie mniejszy niż transportowany datagram UDP, w takim wypadku zostanie on sfragmentowany w warstwie IP.

Protokół komunikacji UDP nie specyfikuje mechanizmów gwarantujących dostarczenie komunikatów, każdy z datagramów może zostać utracony, powielony, opóźniony, dostarczony poza kolejnością wysyłania lub uszkodzony. Poprawność komunikacji musi zapewnić warstwa aplikacji korzystająca z protokołu UDP.

Programy wykorzystujące komunikacje bazującą na protokole UDP mogą wybrać jeden z możliwych rodzajów interakcji:

- Jeden do jednego
- Jeden do wielu
- Wielu do jednego
- Wielu do wielu

Głównymi usługami korzystającymi z protokołu UDP są usługi DNS, VoIP, strumieniowe przesyłanie dźwięku, komunikatory oraz gry sieciowe

1.3. Format datagramu UDP

Format datagramu UDP został przedstawiony na Rys 1, definiuje on nagłówek, który zawiera numery portów umożliwiające rozróżnienie aplikacji działających na hoście oraz pole danych przenoszonych w datagramie.

Rys 1. Format datagramu UDP

<table>
<tr><td>Numer bitu</td><td>0 15</td><td>16 31</td></tr>
<tr><td rowspan="2">Nagłówek UDP</td><td>Port źródłowy komunikatu UDP</td><td>Port docelowy komunikatu UDP</td></tr>
<tr><td>Długość komunikatu UDP</td><td>Suma kontrolna</td></tr>
<tr><td rowspan="2">Pole danych komunikatu UDP</td><td colspan="2">Transmitowane dane</td></tr>
<tr><td colspan="2">...</td></tr>
</table>

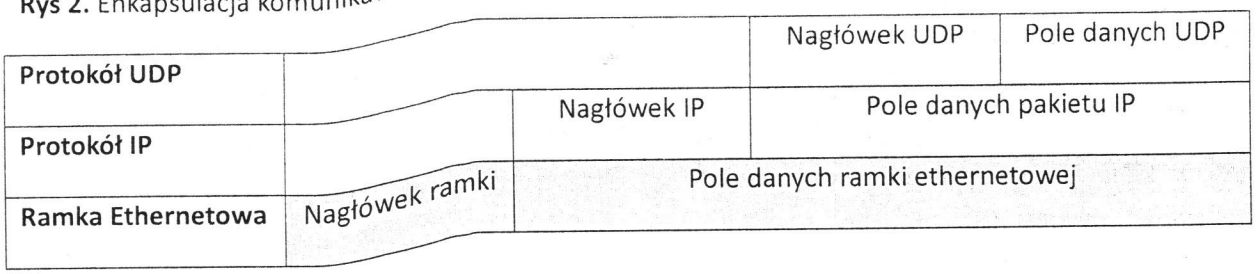
W celu wyznaczenia sumy kontrolnej oprogramowanie UDP tworzy tak zwany *pseudo nagłówek*, który zawiera informacje o źródłowym i docelowym adresie IP, pole typu danych datagramu IP (pola z warstwy intersieci) oraz długość datagramu UDP. Stworzenie pseudo nagłówka i wyznaczenie na jego podstawie sumy kontrolnej umożliwia sprawdzenie czy komunikat został dostarczony do właściwego hosta.

Suma kontrolna w komunikacie UDP jest opcjonalna, nadawca może ustawić bity w tym polu na zero. Odbiorca uwzględnia pole sumy kontrolnej tylko w przypadku gdy jest niezerowe.

1.4. Enkapsulacja komunikatów UDP

Komunikatu protokołu UDP są kapsułkowane w polu danych pakietu IP, który z kolej kapsułkowany jest w polu danych ramki ethernetowej. Przedstawia to Rys 2.

Rys 2. Enkapsulacja komunikatów UDP w datagramie IP i ramce ethernetowej.



1.5. Protokół TCP

Protokół TCP (ang. Transmission Control Protocol) zapewnia niezawodny transport danych w sieciach internetowych bazujących na zawodnym mechanizmie przekazywania pakietów IP. W odróżnieniu od UDP jest to protokół połączeniowy który ustanawia połączenie między dwoma punktami końcowymi. Protokół TCP gwarantuje dostarczenie danych w takiej samej formie i kolejności w jakiej zostały wysłane. Wykorzystuje w tym celu kilka mechanizmów takich jak: sumy kontrolne, numery sekwencyjne, retransmisje pakietów. Każde połączenie jest ustanawiane w sposób jawny i rozłączane po zakończeniu komunikacji.

Protokół TCP umożliwia kontrolę przeciążenia urządzeń znajdujących się na sieci, w wypadku wykrycia zbyt dużego obciążenia zmniejsza prędkość nadawania segmentów przez urządzenia nadawcze.

Protokół TCP dzieli dane pochodzące z wyższej warstwy na porcje zwane segmentami, które następnie enkapsułuje w pakietach IP, które w czasie transmisji na sieci mogą dotrzeć do odbiorcy w kolejności innej niż nadane. Poprzez mechanizm porządkowania segmentów, który nadaje numery sekwencyjne możliwe jest poprawne złożenie segmentu u odbiorcy.

Połączenie ustanowione pomiędzy dwoma punktami jest połączeniem wirtualnym, relegalizowany i obsługiwanym prze oprogramowanie. Złudzenie istnienia połączenia jest realizowane wyłącznie przez moduły TCP, oprogramowanie intersieci (rutery i ich oprogramowanie) nie uczestniczy w zestawieniu połączenia poza dostarczeniem medium – pakietów IP.

W celu nawiązania połączenia protokół TCP definiuje trójetapewe połączenie (ang. 3-way handshake), w czasie którego obydwie strony wysyłają komunikaty sterujące (segment synchronizujący, SYN) zawierające informacje o rozmiarze bufora i numer sekwencyjny obowiązującym na danym komputerze. Do rozłączenia połączenia wymagane jest użycie segmentów FIN.

Algorytm ustanawiania połączenia (opis z zasobów mimuw.edu.pl):

1) Klient inicjuje połączenie poprzez wysłanie segmentu z ustawioną flagą w SYN (=1). Ustawiona wartość pola SYN=1 oznacza, że hosty nie zostały jeszcze zsynchronizowane i segment ten jest żądaniem nawiązania połączenia. Jednocześnie wysyłany jest w tym samym segmencie numer sekwencyjny pakietu o wartości 'x' oraz rozmiar okna, który jest informacją dla serwera jakiej wielkości bufor został zarezerwowany (po stronie klienta) na segmenty przesyłane z serwera.

2) Serwer odbiera pakiet. Po analizie flagi rozpoznaje, że jest to próba nawiązania połączenia. Serwer rejestruje zatem numer sekwencyjny 'x', przydziela dla tego połączenia bufory i zmienne stanu. Odpowiedź będzie zawierała oprócz ustawionego bitu flagi SYN również ustawiony bit flagi ACK. Ustawione bity flag sygnalizują, że będzie to początek konwersji zwrotnej. Serwer wysyła swój własny numer 'y' w polu numer sekwencyjny oraz w polu numer sekwencyjny potwierdzenia wysyła wartość 'x+1'. Wartość wpisana w to ostatnie pole informuje klienta, którą następną porcję bajtów oczekuje serwer. Dodatkowo serwer wysyła rozmiar okna. Wielkość ta informuje klienta o rozmiarze bufora, który został zarezerwowany po stronie serwera na segmenty, które będą przesyłane od klienta.

3) Klient odbiera segment inicjalizuje po swojej stronie bufor na segmenty pochodzące od serwera i na zmienne stanu tego połączenia. W ramach potwierdzenia odebrania segmentu od serwera wysyła pakiet z ustawioną wartością flagi ACK, wyzerowaną wartością pola SYN oraz w polu następny numer sekwencyjny wpisywana jest wartość 'y+1'. Oznacza to, że połączenie zostało nawiązane i teraz klient oczekuje od serwera porcji bajtów od numeru 'y+1'. Wartość SYN=0 oznacza, że połączenie zostało nawiązane i nastąpiła synchronizacja.

Po nawiązaniu połączenia może nastąpić transmisja danych, protokół TCP w tym celu wykorzystuje mechanizm przesuwnego okna, którego rozmiar może się zmieniać.

1.6. Format segmentu TCP

Wszystkie rodzaje wiadomości mają taki sam format (przenoszące dane jaki i potwierdzenia). Format segmentu przedstawia Rys 3.

Rys 3. Format segmentu TCP

Bity	0 4 12 15			16 24 31
Nagłówek TCP	Port źródłowy			Port docelowy
	Numer sekwencyjny			
	Numer sekwencyjny			
	Długość nagłówka	Nieużywane	Bity sterujące	Rozmiar okna
	Suma kontrolna			Wskaźnik ważności
	Opcje (jeśli są zdefiniowane)			
Transmitowane dane	Pole danych			
	...			

Analizując format segmentu należy pamiętać, że na połączenie TCP składają się dwa strumienie danych. Niektóre pola nagłówka odnoszą się do strumienia odbiorcy a inne do nadawcy. Nadawca segmentu ustawia pole **numer potwierdzenie** (odpowiada numerowi sekwencyjnego w następnym segmencie, którego spodziewa się odbiorca) oraz **rozmiar okna** (wskazuje ilość wolnej pamięci w buforze odbiorczym komputera). Odbiorca wykorzystuje numer sekwencyjny do złożenia segmentów które były transportowane w pakietach, które zostały dostarczone poza kolejnością.

2. Cel ćwiczenia

Celem ćwiczenia jest zapoznanie się z podstawowymi aspektami pracy protokołu UDP i TCP

3. Ćwiczenia

1. Korzystając z Wiresharka zidentyfikuj usługi działające na sieci korzystające z protokołów UTP i TCP
2. W programie Wireshark sprawdź jaki port został przydzielony przez system operacyjny na Twoim hoście podczas komunikacji z serwerem „http://www.pk.edu.pl"?

3. Za pomocą Wireshark przeanalizuj róch pomiędzy serwerem a klientem, w tym celu uruchom dwie konsole, w pierwszej wydaj komendę „nc –l 1234" (spowoduje to uruchomienie netcata w trybie serwera nasłuchującego na porcie 1234 protokołu TCP). W drugiej konsoli wydaj komendę „nc 127.0.0.1 1234" – spowoduje to nawiązanie połączenie z localhostem na porcie 1234 protokołu TCP.

   Wpisany tekst w drugiej konsoli zostanie przesłany protokołem TCP do serwera i wypisanie w pierwszej konsoli.

   Zbadaj ruch, nawiązanie połączenia oraz jego zakończenie w programie Wireshark. Wyróżnij segmenty odpowiedzialne za nawiązanie połączenia, transmisje danych i rozłączenie. Wykonaj analogiczne ćwiczenie z wykorzystaniem protokołu UDP. Porównaj przebieg otrzymanych sesji.

   Jak można przesłać plik z wykorzystaniem netcata?

4. Przebadaj jaki segment TCP jest generowany w przypadku gdy jest próba nawiązania połączenia na zamknięty port

5. Korzystając z programu Wiresharka przeanalizuj sesję telnet. W tym celu zaloguj się na serwer consull.ull.ac.uk wykorzystując z programu telnet, hasło: library.

6. Przeanalizuj zawartość pliku"/proc/sys/net/ipv4/tcp_keepalive_time", co ta zmienna zawiera?

   Zmień jej wartość na 100, czy można zauważyć jakąś zmianę w programie Wireshark w porównaniu z poprzednim ustawieniem w przypadku komunikacji z serwerem? Jakie inne zmienne konfiguracyjne znajdują w katalogu „"/proc/sys/net/ipv4/" ?

Ponadto proszę w sprawozdaniu odpowiedzieć na pytania:

- Jakie pole ramki Ethernetowej trzeba sprawdzić w celu określenia czy transmituje ona komunikat UDP?
- Ile ramek zostanie przekazanych przez sieć jeśli będzie transmitowany komunikat UDP wielkości 10kb?
- Czy suma kontrolna TCP jest niezbędna? Czy poprawność transmisji nie można stwierdzić po sumie kontrolnej datagramu IP?
- Opisz mechanizm przesuwnego okna

4. Bibliografia

- „Sieci komputerowe i intersieci" D. E. Comer
- „Sieci komputerowe TCP/IP. Zasady, protokoły i architektura" D. E. Comer
- „Wprowadzenie do CCNA" A. Józefiok

Instrukcja opracowana przez:

mgr inż. Kazimierz Kiełkowicz