Sieci komputerowe:

WYŻSZE WARSTWY MODELU OSI

Agata Półrola

Katedra Informatyki Stosowanej UŁ

polrola@math.uni.lodz.pl

http://www.math.uni.lodz.pl/~polrola

Protokoły TCP i UDP

Adresowanie komunikatów

- Adresatem datagramów IP był konkretny komputer, identyfikowany poprzez adres IP
- Protokoły wyższej warstwy umożliwiają rozróżnienie między różnymi programami czy użytkownikami na danym komputerze

Adresowanie komunikatów – c.d.

- Systemy operacyjne są zazwyczaj wieloprogramowe – wiele procesów jest wykonywanych równocześnie
- Zazwyczaj adresatem komunikatów nie jest proces, ale port

Porty protokołów

- każda maszyna posiada zbiór abstrakcyjnych punktów docelowych, zwanych portami protokołów
- porty protokołów identyfikowane są przez liczby całkowite dodatnie
- porty zazwyczaj są buforowane
- procesy korzystają z portów

Porty protokołów – c.d.

- System operacyjny zawiera mechanizmy określania portów i dostępu do nich
- Każda aplikacja negocjuje z systemem operacyjnym port którego używa do przesyłania komunikatów

Porty protokołów – c.d.

- Sposoby przypisywania numerów portów:
  - centralny

    tzw. *well-known ports* – numery portów są przyznawane centralnie,

    (najczęściej przeznaczone dla serwerów konkretnych usług)
  - dynamiczny

    numery portów przyznawane są aplikacjom lokalnie na danym komputerze

Porty protokołów – c.d.

- W celu skomunikowania się z aplikacją na odległym komputerze należy znać:
  - adres IP komputera
  - numer portu docelowego
- Każdy komunikat powinien przenosić numery portu źródłowego i docelowego *(source & destination port)*
  - numer portu źródłowego jest wykorzystywany przy przesyłaniu odpowiedzi

Protokoły warstwy transportu

- Używanymi w sieciach TCP/IP protokołami warstwy transportu są:
  - UDP – *User Datagram Protocol*
  - TCP – *Transmission Control Protocol*

Umożliwiają one przesyłanie danych między portami

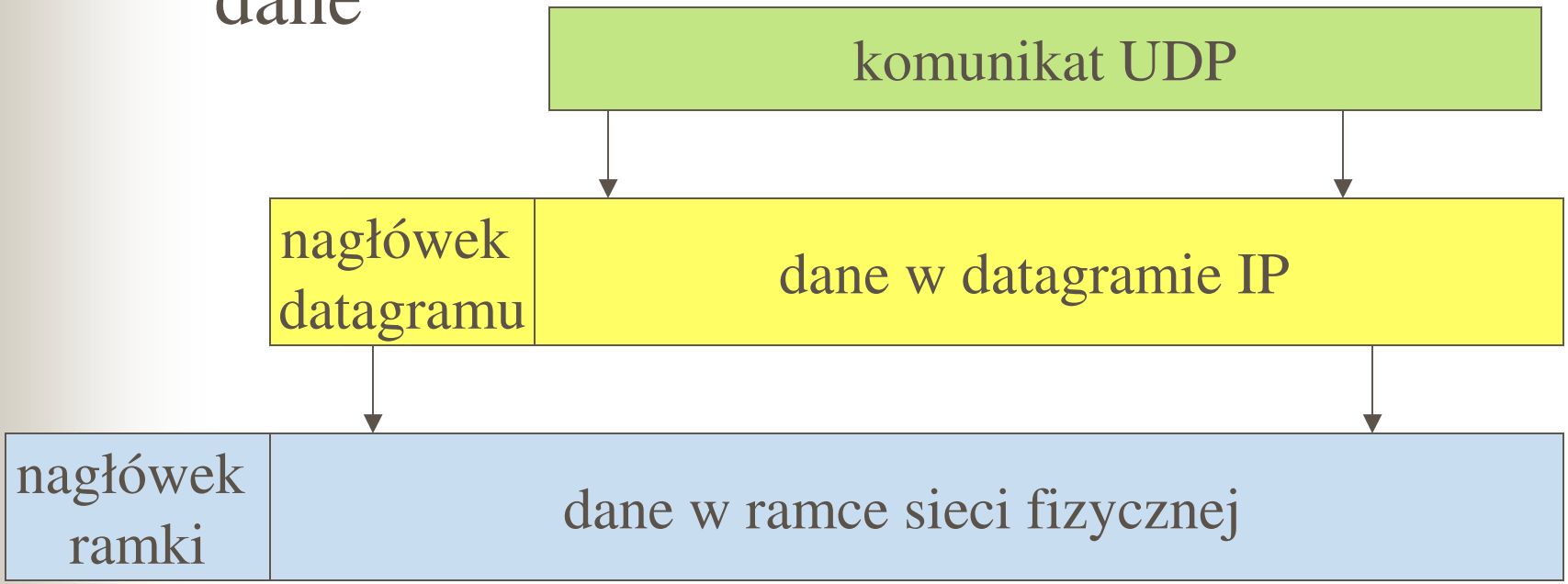
Protokół UDP

Protokół UDP

- Właściwości UDP:
  - Protokół bezpołączeniowy
  - Nie gwarantuje dostarczenia danych
- Porty UDP:
  - część numerów portów jest przyznawana centralnie *(well-known ports)*, część przypisywana dynamicznie
  - komunikat UDP (zwany *datagramem użytkownika*) zawiera numer portu źródłowego i docelowego

Protokół UDP – c.d.

- Komunikat UDP jest przesyłany siecią w części datagramu IP przeznaczonej na dane



Protokół UDP – c.d.

- Oprogramowanie UDP dokonuje przenoszenia danych między warstwami:
  - „zbiera” datagramy UDP z różnych aplikacji i przekazuje je IP do przesłania
  - odbiera otrzymane datagramy od IP i przekazuje je odpowiednim aplikacjom

(multiplexing / demultiplexing UDP)

- Rozróżnianie między aplikacjami bazuje na mechanizmie portów protokołów

Format komunikatów UDP

źródłowy port UDP	docelowy port UDP
długość komunikatu UDP	suma kontrolna UDP
dane	

- numery portów – 16-bitowe
- długość – liczba oktetów datagramu UDP, razem z nagłówkiem i danymi.
  Minimalna wartość – 8, tzn. sam nagłówek
- suma kontrolna – opcjonalna; obliczana na podstawie datagramu UDP i jego pseudonagłówka

Pseudonagłówek UDP

adres IP nadawcy		
adres IP odbiorcy		
zero	protokół (17)	długość datagramu UDP

- długość datagramu IP – długość bez pseudonagłówka
- Suma kontrolna UDP pozwala sprawdzić, czy datagram UDP dotarł do właściwego adresata.
  - Odbiorca datagramu wykorzystuje do obliczenia sumy kontrolnej adresy IP nadawcy i odbiorcy, które otrzymał w datagramie IP

Protokół TCP

Protokół TCP

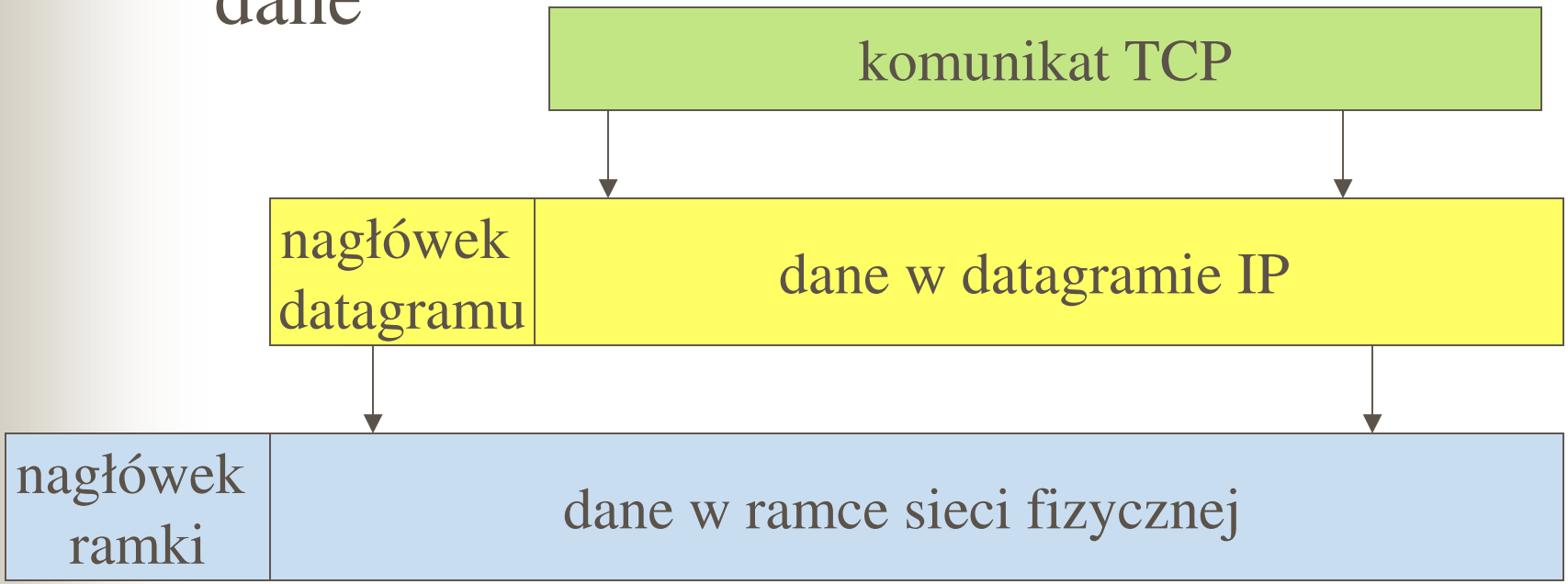
- TCP *(Transmission Control Protocol)* jest kolejnym protokołem umożliwiającym przesyłanie danych między portami

Protokół TCP – c.d.

- Porty TCP:
  - część numerów portów jest przyznawana centralnie *(well-known ports)*, część przypisywana dynamicznie
  - komunikat TCP (zwany *segmentem*) zawiera numer portu źródłowego i docelowego

Protokół TCP – c.d.

- Komunikat TCP jest przesyłany siecią w części datagramu IP przeznaczonej na dane



Protokół TCP – c.d.

- Właściwości TCP:
  - protokół zorientowany połączeniowo
  - niezawodność przesyłania danych
  - interfejs strumieniowy
  - komunikacja w pełni dwukierunkowa
  - transfer buforowany

Protokół TCP – c.d.

- Połączenie TCP jest zdefiniowane przez parę swoich punktów końcowych, będących parami *(host, port)*

Protokół TCP – c.d.

- Sposób zapewnienia niezawodności – mechanizm tzw. pozytywnego potwierdzania z retransmisją *(positive acknowledgement with retransmission)*
  - wymaga od odbiorcy skomunikowania się z nadawcą przez odesłanie potwierdzenia
  - nadawca przechowuje kopię wysłanego pakietu; jeśli w odpowiednim czasie nie otrzyma potwierdzenia, to retransmituje pakiet

Protokół TCP – c.d.

- Strumieniowe przesyłanie danych i potwierdzeń jest efektywne dzięki mechanizmowi przesuwających się okien *(sliding windows)*
  - mechanizm ten pozwala lepiej wykorzystać przepustowość sieci (można wysłać wiele pakietów przed otrzymaniem potwierdzenia)

Protokół TCP – c.d.

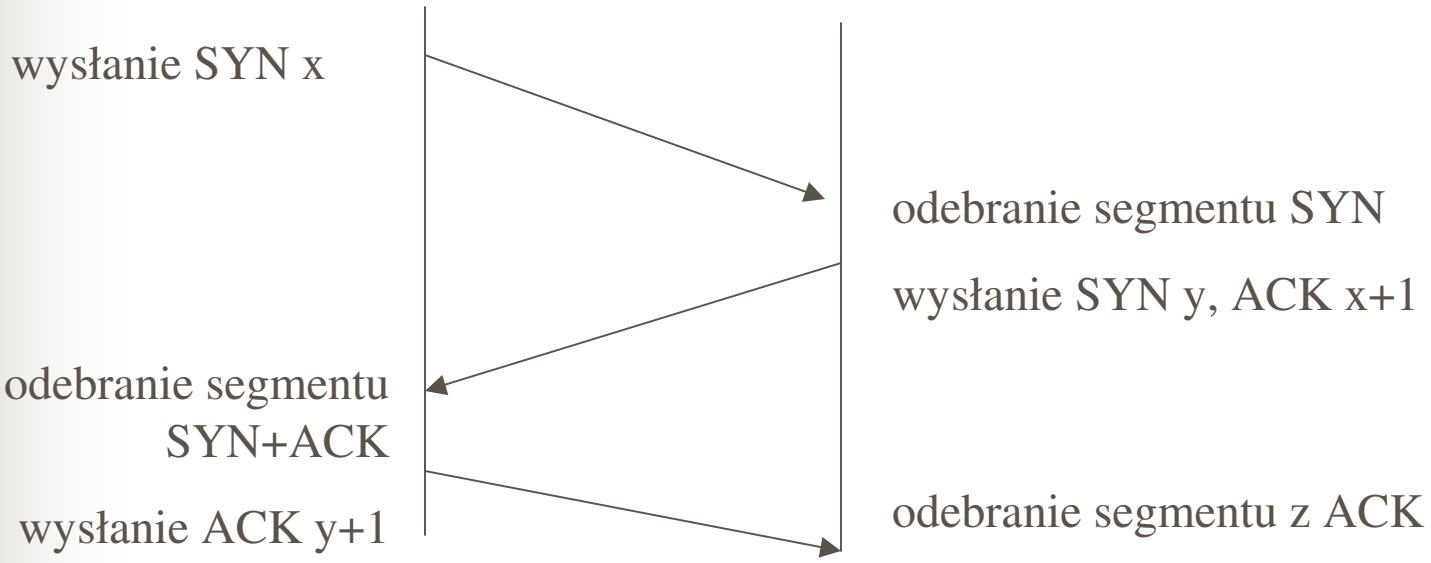
- Protokół TCP definiuje m.in:
  - sposób nawiązywania i zamykania połączenia
  - format komunikatów TCP i potwierdzeń
  - mechanizmy obsługi błędów (jak zduplikowane czy zgubione pakiety)

Protokół TCP: połączenie

- Połączenie definiuje para „końców” *(host, port)*
  - dany port TCP może być dzielony między kilka połączeń
- Oba końce połączenia uzgadniają, że chcą „rozmawiać”:
  - z jednej strony wykonywane jest tzw. *positive open* – program komunikuje się ze swoim systemem operacyjnym, informuje że będzie przyjmował dane i dostaje numer portu TCP
  - z drugiej strony – *active open request* – program komunikuje się ze swoim systemem operacyjnym informując, że chce nawiązać połaczenie
  - Moduły IP na obu końcach komunikują się z sobą w celu ustanowienia połączenia, którym będzie można przesyłać dane

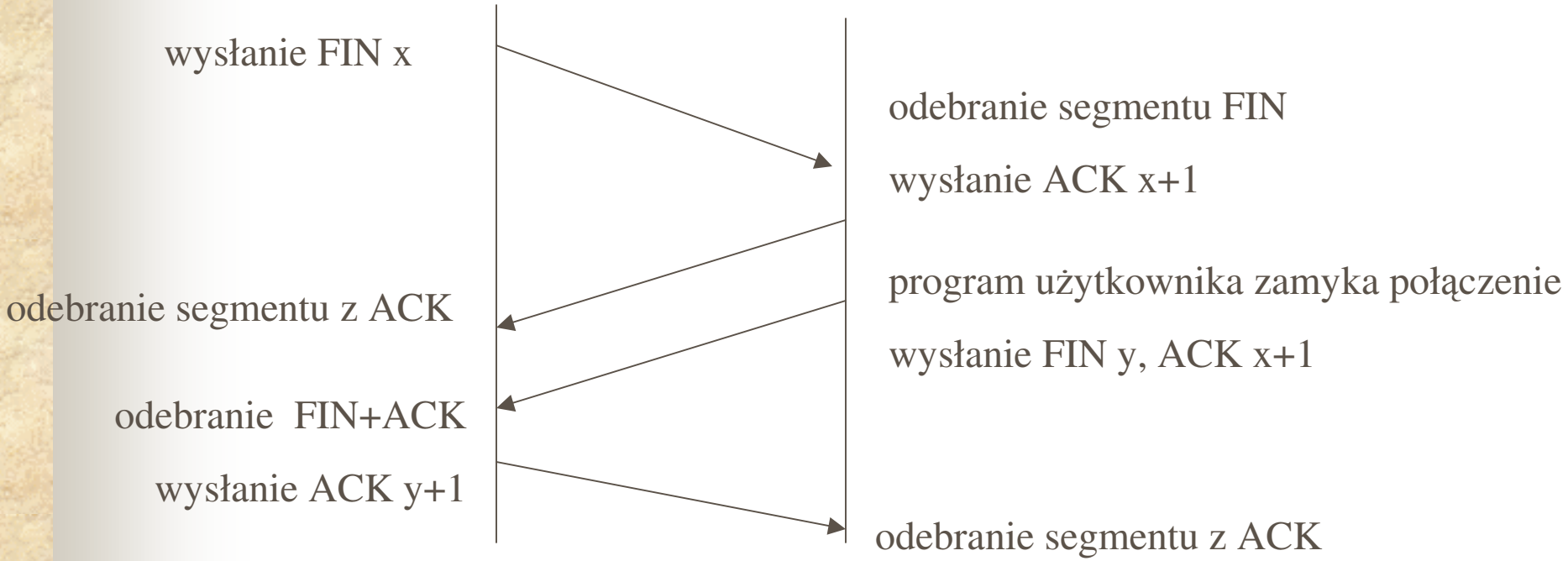
Protokół TCP: otwieranie połączenia

- Nawiązanie połączenia wymaga przesłania trzech komunikatów *(three-way handshake)*



Protokół TCP: zamykanie połączenia

- Zamykanie połączenia również jest wielostopniowe



Format segmentu TCP

Format segmentu TCP – c.d.

- **nr porządkowy** – pozycja danych segmentu w strumieniu oktetów nadawcy
- **nr potwierdzenia** – nr oktetu który nadawca spodziewa się dostać w następnej kolejności
- **bity kodu** – określają zawartość i przeznaczenie segmentu
- **okno** – rozmiar okna sugerowany odbiorcy komunikatu przez jego nadawcę
  - pozwala to dostosować rozmiar okna (a zatem liczbę transmitowanych segmentów) do możliwości odbiorcy, np. do stopnia zapełnienia jego buforów

Format segmentu TCP – c.d. : bity kodu

- URG – zawartość pola **wskaźnik pilnych danych** jest istotna
- ACK – pole **nr potwierdzenia**jest istotne
- PSH – segment z żądaniem „wypchnięcia” (wysłania segmentu TCP mimo że bufor jeszcze nie jest pełny)
- RST – resetowanie połączenia
- SYN – synchronizacja numerów porządkowych
- FIN – nadawca doszedł do końca strumienia danych do wysłania

Format segmentu TCP – c.d.

- Chociaż TCP jest protokołem strumieniowym, ważne jest, aby można było przesyłać dane poza głównym strumieniem transmisji, nie czekając aż program na drugim końcu połączenia przyjmie wszystkie dane znajdujące się w strumieniu
- TCP umożliwia określenie, że dane są *pilne*:
  - przy transmisji pilność danych zaznacza się bitem kodu URG; **wskaźnik pilnych danych** określa koniec takich danych w segmencie
  - program odbiorcy powinien przejść do „trybu pilności” i obsłużyć otrzymane pilne dane

Format segmentu TCP – c.d.

- **opcje** umożliwiają m.in. wynegocjowanie maksymalnego rozmiaru segmentów TCP przesyłanych w danym połączeniu
  - nie wszystkie segmenty wysyłane podczas połączenia muszą mieć ten sam rozmiar
  - zarówno zbyt małe, jak i zbyt duże segmenty prowadzą do nieefektywności

Pseudonagłówek TCP

- Suma kontrolna TCP obliczana jest na podstawie segmentu i tzw. pseudonagłówka:

<table>
<tr><td colspan="3">adres IP nadawcy</td></tr>
<tr><td colspan="3">adres IP odbiorcy</td></tr>
<tr><td>zero</td><td>protokół (6)</td><td>długość segmentu TCP</td></tr>
</table>

  - umożliwia sprawdzenie, czy segment dotarł bez uszkodzeń i do właściwego odbiorcy

Potwierdzanie i retransmisja

- Ponieważ TCP wysyła dane w segmentach o zmiennej długości i ponieważ retransmitowane segmenty mogą zawierać więcej danych niż segmenty pierwotne, więc potwierdzenia nie mogą odnosić się do segmentów, tylko do oktetów
- Odbiorca musi być w stanie zrekonstruować strumień oktetów nadawcy

Potwierdzanie i retransmisja – c.d.

- Potwierdzenie TCP określa numer oktetu, który spodziewa się otrzymać odbiorca (numer pierwszej „dziury” w rekonstruowanym strumieniu)
  (schemat skumulowanego potwierdzania)
  - potwierdzanie łatwe i jednoznaczne
  - zgubienie potwierdzenia nie musi powodować retransmisji
  - wada – nadawca nie ma informacji o wszystkich poprawnie przesłanych danych

Potwierdzanie i retransmisja – c.d.

- Oprogramowanie TCP przesyłając dane każdorazowo ustawia zegar. Jeżeli ustalony czas zostanie przekroczony zanim przybędzie potwierdzenie, to dane są retransmitowane
- Potrzebna jest przy tym obsługa różnych, zmieniających się opóźnień (oprogramowanie TCP obsługuje komunikację w różnych sieciach, różnymi łączami i na różne odległości)

Potwierdzanie i retransmisja – c.d.

- Zamiast stałego czasu oczekiwania na potwierdzenie stosuje się retransmisję z adaptacją:
  - TCP śledzi aktualne opóźnienia występujące w danym połączeniu i dostosowuje do tego czas po jakim następuje retransmisja
  - jest to wykonywane niezależnie dla każdego połączenia

Potwierdzanie i retransmisja – c.d.

- Śledzenie połączenia polega na szacowaniu tzw. RTT *(round-trip time)* – czasu upływającego od wysłania danych do uzyskania potwierdzenia
- Na podstawie RTT kolejnych transmitowanych segmentów oblicza się średnie opóźnienie, a na jego podstawie – czas po jakim następuje retransmisja

Obsługa przeciążeń sieci

- Przeciążenia *(congestions)* mają zazwyczaj miejsce na routerach (gdy nie nadążają one z obsługą nadchodzących pakietów)
- Routery likwidują wówczas pakiety
- Jeśli reakcją na przeciążenia byłaby retransmisja, to przeciążenie by się zwiększało
- TCP musi więc reagować na przeciążenia zmniejszeniem intensywności transmisji

Obsługa przeciążeń sieci – c.d.

- Dwie metody reagowania na przeciążenia:
  - metoda powolnego startu
  - metoda wielokrotnego zmniejszania

Metoda wielokrotnego zmniejszania

- dla każdego połączenia TCP pamięta rozmiar okna odbiorcy (rozmiar bufora proponowanego w potwierdzeniach)
  - w celu kontroli przeciążeń utrzymywane jest okno przeciążeniowe
    - okno przeciążeniowe jest w normalnej sytuacji równe oknu odbiorcy; zgubienie segmentu powoduje zmniejszenie go o połowę (aż do osiągnięcia minimalnego rozmiaru jednego segmentu)
  - rozmiar bieżącego okna nadawcy jest równy mniejszemu z powyższych
  - dla segmentów pozostałych w oknie zwiększa się wykładniczo czas po którym ma nastąpić retransmisja

Metoda powolnego startu

- Po wyjściu ze stanu przeciążenia (a także przy rozpoczynaniu ruchu w ramach nowego połączenia) stosowana jest metoda powolnego startu:
  - na początku okno przeciążeniowe ma rozmiar jednego segmentu
  - rozmiar ten jest zwiększany o jeden segment po otrzymaniu potwierdzenia
    - w przypadku gdy rozmiar okna przeciążeniowego osiąga połowę swojej wartości sprzed przeciążenia, okno zwiększane jest tylko wtedy, gdy wszystkie segmenty w oknie zostały potwierdzone (stan unikania przeciążenia)

Modele pracy w sieci

Model klient - serwer

- Podstawowym modelem interakcji między programami użytkowymi jest model klient – serwer

Model klient – serwer – c.d.

- Serwer – każdy program oferujący usługę dostępną przez sieć. Przyjmuje przez sieć zamówienia, wykonuje usługę i zwraca wyniki zamawiającemu
- Klient – program wysyłający zamówienia do serwera i korzystający z jego usług

Różnice między klientem a serwerem

- Serwer rozpoczyna działanie zanim rozpocznie współpracę przez sieć; zazwyczaj działa w sposób ciągły, przyjmując zlecenia i odpowiadając na nie
- Klient wysyła zlecenia i czeka na ich realizację, zwykle kończąc działanie po kilkakrotnym skorzystaniu z usługi udostępnianej przez serwer

Różnice między klientem a serwerem – c.d.

- Serwer oczekuje na zlecenia korzystając z zarezerwowanego portu, przeznaczonego dla usługi którą oferuje
- Klient na potrzeby swojej komunikacji rezerwuje dowolny, nie zarezerwowany i nie używany port
  - Przypisanie każdej usłudze jednoznacznego identyfikatora portu ułatwia tworzenie zarówno klientów, jak i serwerów

Alternatywa dla modelu klient - serwer

- Alternatywą jest rozgłaszanie informacji na dany temat przez określone maszyny
- Każdy z komputerów przechowuje w pamięci podręcznej uzyskane informacje
  - dane te można łatwo udostępnić
  - rozwiązanie zużywające czas procesora oraz obciążające sieć

Inicjowanie działania i autokonfiguracja

protokoły BOOTP i DHCP

Zapotrzebowanie

- Każdy komputer dołączony do sieci TCP/IP musi znać swój adres IP aby móc wysyłać i odbierać datagramy;
  - do poprawnej komunikacji potrzebuje również: maski podsieci, adresu routera, adresu serwera DNS
- W niektórych przypadkach informacje te nie są pamiętane przez komputer (przykład: stacje bezdyskowe)

Rozwiązanie:

- Uzyskanie potrzebnych informacji może nastąpić w drodze interakcji klient – serwer
- Potrzebny jest zatem protokół umożliwiający komputerowi uzyskanie adresu IP (i ewentualnie innych potrzebnych informacji) od świadczącego odpowiednią usługę serwera

Rozwiązanie I: protokół RARP

- Pierwsze rozwiązanie tego rodzaju to (omówiony już) protokół RARP
  - Wady:
    - działa na niskim poziomie, używanie go wymaga bezpośredniego dostępu do sprzętu sieciowego
    - odpowiedź RARP zawiera mało informacji (tylko adres IP którego ma używać klient)
    - używa do identyfikacji klienta adresu sprzętowego, co może być problemem w sieciach, w których adresy sprzętowe nie są stałe

Rozwiązanie II: protokół BOOTP

- Protokół BOOTP *(Bootstrap Protocol)* został opracowany jako alternatywa dla RARP
- Klient i serwer komunikują się używając protokołu UDP (i IP), zatem oprogramowanie obsługujące tę komunikację może być zaimplementowane jako program użytkowy
  - nie jest potrzebne dostosowywanie oprogramowania do specyfiki konkretnej sieci (rozwiązań sprzętowych)

Protokół BOOTP – c.d.

- Komputer – klient BOOTP wysyła swoją prośbę używając adresu ograniczonego rozgłaszania (255.255.255.255)
  - bezpośrednia komunikacja ograniczona jest zatem do sieci lokalnej
- Serwer odsyła odpowiedź klientowi
  - używając adresu ograniczonego rozgłaszania

albo

  - wykorzystując adresu sprzętowy klienta otrzymany w zapytaniu BOOTP

Protokół BOOTP – c.d.

- Ponieważ komunikacja bazuje na UDP, więc nie ma gwarancji dostarczenia komunikatów
- Odpowiedzialność za poprawny przebieg komunikacji spoczywa na kliencie
- Gubienie datagramów obsługiwane jest w standardowy sposób: klient wysyłając prośbę uruchamia zegar, jeśli nie otrzyma odpowiedzi w odpowiednim czasie, to wysyła prośbę ponownie

Protokół BOOTP – c.d.

- BOOTP wymaga używania przez UDP sum kontrolnych
- BOOTP wymaga od IP przesyłania prośb i odpowiedzi BOOTP z ustawionym bitem „nie fragmentuj”, aby można było obsłużyć również klientów mających zbyt mało pamięci na złożenie datagramu
- BOOTP obsługuje wielokrotne odpowiedzi – przetwarzana jest tylko pierwsza

Protokół BOOTP – c.d.

operacja	typ sprzętu	dł.adr.sprz.	etapy
identyfikator transakcji			
sekundy		nieużywane	
adres IP klienta			
twój adres IP			
adres IP serwera			
adres IP routera			
adres sprzętowy klienta (16 oktetów)			
nazwa serwera (64 oktety)			
nazwa pliku startowego (128 oktetów)			
obszar opcji (64 oktety)			

Protokół BOOTP – c.d.

- **operacja** – prośba (1) czy odpowiedź (2) (prośby i odpowiedzi mają ten sam format)
- **typ sprzętu**, **długość adresu sprzętowego** – jak w protokole ARP (Ethernet – typ 1)
- **etapy** – maksymalna liczba routerów przez które może przejść komunikat; klient umieszcza 0; jeśli serwer – pośrednik odbierze komunikat i zdecyduje się przekazać go do innej maszyny, to zwiększa wartość pola
- **identyfikator transakcji** – liczba używana przez klienta do powiązania odpowiedzi z pytaniem
- **sekundy** – liczba sekund jaka upłynęła od momentu rozpoczęcia przez klienta procedur startowych

Protokół BOOTP – c.d.

- **adres IP klienta** i dalsze pola – główne dane w komunikacie. Klient wypełnia wszystkie pola które zna, pozostałym nadaje wartość zero.
  - Jeśli klient zna nazwę lub adres serwera, to może ją umieścić w pytaniu, wtedy odpowiedzi udzieli mu tylko ten serwer. W przeciwnym wypadku odpowiedzi udzielają wszystkie serwery, które otrzymały zapytanie
  - Protokół BOOTP może być używany także przez klientów znających już swoje adresy IP, np. w celu otrzymania nazwy pliku startowego. W takim przypadku wypełniają oni pole **adres IP klienta**.
  - Jeśli w prośbie BOOTP pole **adres IP klienta** jest zerowe, to w odpowiedzi serwer odsyła mu adres IP umieszczony w polu **twój adres IP**

Protokół BOOTP – c.d.

- W przypadku, gdy celem klienta jest znalezienie pliku z obrazem systemu, który miałby być u niego załadowany, protokół BOOTP umożliwia wykonanie dwuetapowej procedury startowej:
  - BOOTP dostarcza klientowi danych pozwalających znaleźć obrazu systemu (pole **nazwa pliku startowego**)
  - klient pobiera plik startowy o podanej nazwie, używając innego protokołu (plik ten może być umieszczony na innym serwerze niż serwer BOOTP)
    - Umożliwia to klientowi poproszenie o określony system operacyjny (np. UNIX – nazwa ta jest umieszczana przez niego w polu **nazwa pliku startowego**). Serwer wpisuje w to pole rzeczywistą nazwę pliku (nazwy plików mogą być różne dla różnych klientów, np. w zależności od architektury)

Protokół BOOTP – c.d.

- pole **obszar opcji** umożliwia przesłanie przez serwer pewnych dodatkowych informacji:
  - pierwsze cztery oktety nazywane są *„magicznym ciasteczkiem” (magic cookie)* i definiują format elementów w dalszej części pola. Standardowy format elementów (opisywany przez ciasteczko o wartości 99.130.83.99) to jeden oktet **typu**, jeden oktet **długości** i pewna liczba oktetów będących **wartością** elementu.
  - opcje pozwalają na przesłanie np. maski podsieci, bieżącego czasu, nazwy klienta, adresów IP routerów itp.

Protokół BOOTP – c.d.

- Protokół BOOTP został zaprojektowany z myślą o względnie stałym środowisku, w którym każdy host jest podłączony na stałe do sieci:
  - administrator tworzy na serwerze plik konfiguracyjny BOOTP, określający zestaw parametrów BOOTP dla każdego klienta
  - klient identyfikowany jest zazwyczaj na podstawie adresu sprzętowego przysłanego w pytaniu. Pozwala to serwerowi odesłać odpowiednie dane pobrane z powyższego pliku
  - BOOTP umożliwia szybkie określenie konfiguracji hosta na podstawie jego identyfikatora, ale konfiguracja jest zapisana w pliku i zawsze taka sama

Rozwiązanie III: protokół DHCP

- W przypadku zmieniającego się środowiska (np. różne komputery przenośne dołączane do sieci) protokół BOOTP nie jest wystarczający
- Problem pojawia się również, jeżeli adresów IP w danej sieci jest zbyt mało na to, by przydzielić stały adres każdemu komputerowi, ale liczba komputerów pracujących równocześnie jest zazwyczaj mniejsza (posiadana pula adresów mogłaby być więc w danej chwili wystarczająca)
- Pojawia się potrzeba dynamicznej konfiguracji

Protokół DHCP – c.d.

- Dynamiczne konfigurowanie węzłów umożliwia protokół DHCP *(Dynamic Host Configuration Protocol)*
- DHCP rozszerza BOOTP na dwa sposoby:
  - umożliwia przesłanie wszystkich potrzebnych klientowi informacji w jednym komunikacie
  - pozwala przydzielić klientowi adres w sposób dynamiczny i na krótki czas:
    - klient włączając się do sieci wysyła prośbę o przydzielenie adresu IP
    - serwer wybiera wolny adres IP z puli adresów do przydziału i odsyła go klientowi

Protokół DHCP – c.d.

- DHCP obsługuje trzy metody przydziału adresów:
  - „ręczna” konfiguracja przydziału
    - administrator ustala w pliku konfiguracyjnym, jaki adres ma otrzymywać dany klient (jak w BOOTP)
  - konfiguracja automatyczna
    - gdy komputer podłącza się po raz pierwszy do sieci serwer przypisuje mu adres IP; przy kolejnych podłączeniach klient otrzymuje zawsze ten sam adres
  - przydział dynamiczny
    - serwer „wynajmuje” adres klientowi na określony czas

Protokół DHCP – c.d.

- Klient rozgłasza prośbę DHCP, używając adresu ograniczonego rozgłaszania.
- Komunikacja klient – serwer bazuje na protokole UDP
- Jeśli po wysłaniu prośby klient nie dostanie odpowiedzi, to ponawia prośbę
- Serwery DHCP w sieci lokalnej oferują klientowi adresy IP; klient negocjuje z odpowiednim serwerem wypożyczenie wybranego adresu
- Adres jest wypożyczany na pewien ustalony czas (zależny od specyfiki sieci i potrzeb klienta)
- Pod koniec czasu wynajmu klient musi albo wynegocjować przedłużenie czasu wynajmu, albo zwrócić adres

Protokół DHCP – c.d.

- Staranie o przedłużenie wynajmu rozpoczynane jest po upływie 50% czasu. Klient wysyła do serwera prośbę o przedłużenie zawierającą obecne IP, może też zasugerować czas przedłużenia. Serwer może się zgodzić lub nie.
- Jeśli serwer nie wyrazi zgody na przedłużenie wynajmu, klient musi przestać używać tego adresu i rozpocząć starania o nowe IP
- Jeśli klient wysłał prośbę o przedłużenie, ale nie otrzymuje odpowiedzi, to po upływie 87,5% czasu wynajmu zakłada, że serwer jest nieosiągalny i rozpoczyna procedurę przewiązywania adresu – stara się skontaktować z innym serwerem DHCP w sieci lokalnej i uzyskać od niego potwierdzenie wynajmu tego adresu

Protokół DHCP – c.d.

- Jeśli klient nie uzyska żadnej odpowiedzi przed upływem czasu wynajmu, to musi przestać używać adresu i rozpocząć starania o nowe IP
- Przy następnym dołączeniu do sieci klient może poprosić o przydzielenie tego samego adresu który miał poprzednio (jeśli go zapamiętał)
- Wynajem adresu można zakończyć przed czasem (klient informuje serwer że już nie będzie korzystał z przydzielonego IP)
- Podobnie jak w BOOTP, serwer DHCP nie musi się znajdować w sieci lokalnej; można użyć serwera pośredniczącego, który przekaże prośbę serwerowi DHCP umieszczonemu w innej sieci fizycznej, a po otrzymaniu odpowiedzi przekaże ją klientowi

Protokół DHCP – c.d.

operacja	typ sprzętu	dł.adr.sprz.	etapy
identyfikator transakcji			
sekundy		znaczniki	
adres IP klienta			
twój adres IP			
adres IP serwera			
adres IP routera			
adres sprzętowy klienta (16 oktetów)			
nazwa serwera (64 oktety)			
nazwa pliku startowego (128 oktetów)			
opcje (zmienna długość)			

Protokół DHCP – c.d.

- Większość pól komunikatu DHCP jest taka sama jak w BOOTP
- zamiast pola **nieużywane** występuje pole **znaczniki**. Nadanie pierwszemu bitowi tego pola wartości 1 oznacza, że klient prosi o rozgłoszenie odpowiedzi (zazwyczaj serwer odsyła ją bezpośrednio do klienta korzystając z adresu sprzętowego zawartego w prośbie)
- Pole **opcje** ma taki sam format jak w przypadku BOOTP i uwzględnia wszystkie opcje używane w tamtym protokole. Dodatkowe opcje umożliwiają np. określenie typu komunikatu DHCP (prośba o przydział adresu, prośba o przedłużenie wynajmu itp)

Protokół DHCP – c.d.

- Protokół DHCP może (ale nie musi) współpracować z DNS (starsze wersje serwerów DNS nie umożliwiały w ogóle takiej współpracy)

System odwzorowywania nazw

Domain Name System

Zapotrzebowanie

- Użytkownicy sieci wolą korzystać z nazw (np. www.wp.pl) zamiast z adresów IP
- Protokoły sieciowe wymagają do komunikacji adresów IP
- Nazwy należy zatem przetłumaczyć na odpowiadające im adresy IP
  - w większości przypadków tłumaczenie to jest wykonywane automatycznie, a wynik nie jest przedstawiany użytkownikowi, tylko zachowywany w pamięci i wykorzystywany do przesyłania datagramów

Interakcja klient - serwer

- Działanie oprogramowania dokonującego tłumaczenia nazw na adresy IP jest przykładem interakcji klient – serwer
- Gdy program ma przetłumaczyć nazwę, staje się klientem systemu nazw:
  - baza danych z nazwami jest rozproszona po serwerach w cały, Internecie
  - klient wysyła do serwera nazw komunikat z zapytaniem, serwer odnajduje odpowiedni adres i wysyła komunikat z odpowiedzią
  - jeśli serwer nie potrafi udzielić odpowiedzi, to staje się tymczasowo klientem innych serwerów, dopóki nie znajdzie odpowiedzi na pytanie

- DNS – Domain Name System – system nazw domen (dziedzin)

Struktura nazw

- Nazwa – ciąg znaków alfanumerycznych. którego poszczególne człony pooddzielane są kropkami
- Nazwy domen są zhierarchizowane. Najbardziej znacząca część nazwy znajduje się po prawej stronie, skrajna lewa część jest zazwyczaj nazwą konkretnego komputera
- Liczba członów nazwy może być dowolna, system nazw nie narzuca również co reprezentują poszczególne człony nazwy. Decyzje o postaci nazwy podejmowane są lokalnie

Struktura nazw – c.d.

- Najbardziej znaczący człon nazwy to tzw. główny poziom DNS
- Najważniejsze domeny głównego poziomu (tzw. *top-level domains*):
  - .com, .edu, .gov, .mil, .net, .org, .arpa, .int
  - kody krajów

System DNS

- Oprócz reguł dotyczących nazw i podziału odpowiedzialności za nazwy, system DNS zawiera również rozproszony system odwzorowywania nazw na adresy

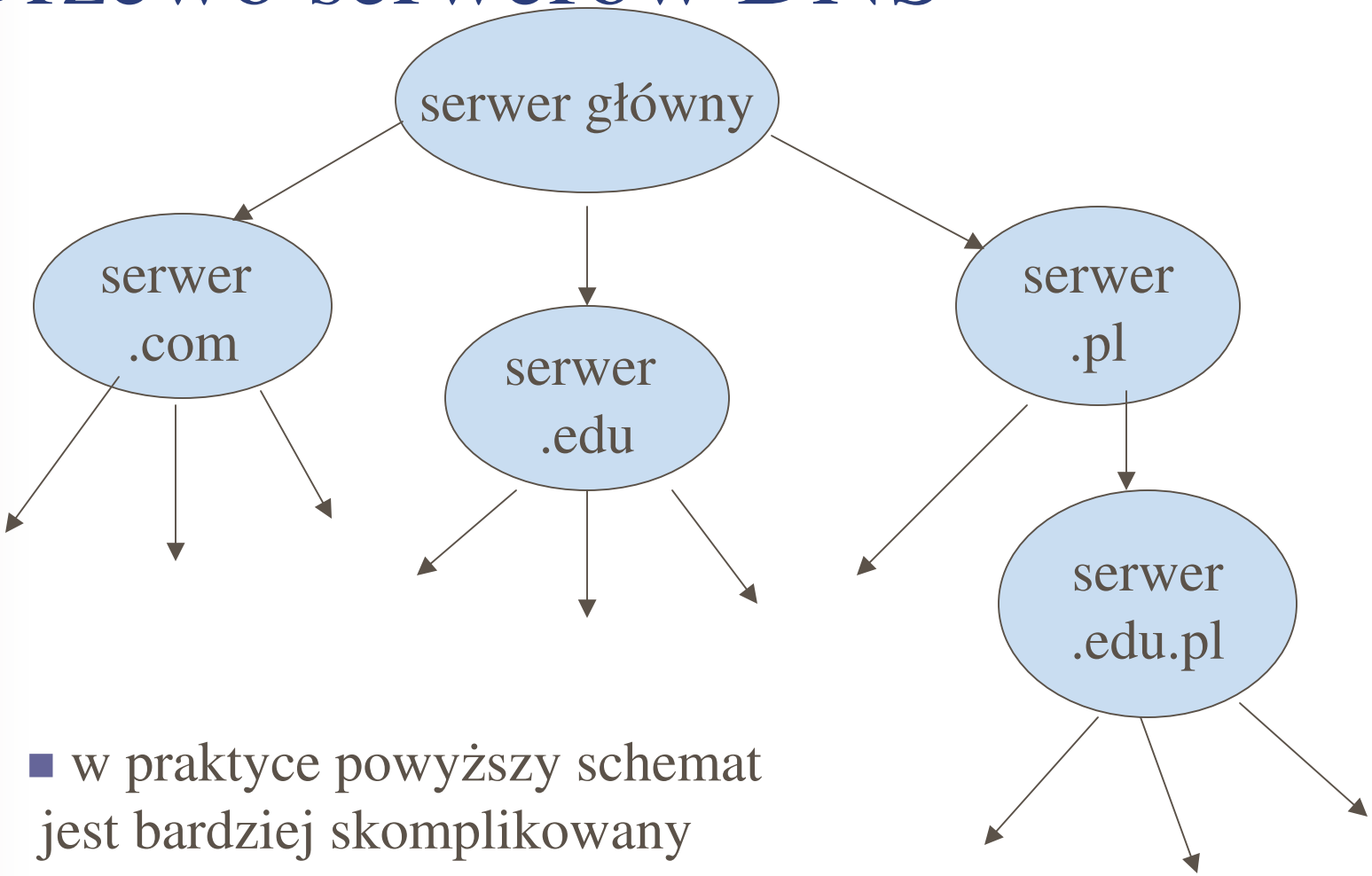
System DNS – c.d.

- System odwzorowywania nazw działa w następujący sposób:
  - większość nazw można odwzorować lokalnie, tylko niewiele zapytań wymaga przesyłania komunikatów w intersieci
  - system określa się jako niezawodny – awaria pojedynczej maszyny nie powoduje niepoprawnej pracy systemu

System DNS – c.d.

- System rozproszony – złożony z wielu serwerów znajdujących się w różnych ośrodkach.
- Serwery są niezależne, ale współpracują ze sobą przy odwzorowywania nazw
- Dla każdej domeny istnieje serwer DNS, który jest za nią odpowiedzialny

Drzewo serwerów DNS



- w praktyce powyższy schemat jest bardziej skomplikowany

Drzewo serwerów DNS – c.d.

- Ze względu na efektywność działania systemu istnieje kilka serwerów głównych *(root servers)*
- Serwery główne przechowują listy serwerów DNS odpowiedzialnych za domeny głównego poziomu
- Serwer danej domeny przechowuje informacje o serwerach odpowiedzialnych za poddomeny tej domeny (jest tak na każdym poziomie drzewa)

Drzewo serwerów DNS – c.d.

- Krawędzie drzewa nie mają nic wspólnego z fizycznymi połączeniami między serwerami; serwery mogą znajdować się w dowolnym miejscu Internetu
- W praktyce drzewo serwerów jest dość płytkie; pojedynczy serwer może przechowywać informacje dotyczące większej części hierarchii nazw niż wynika z przedstawionego schematu

Odwzorowywanie nazw

- Klient wysyła zapytanie do „znanego mu” serwera DNS
- Serwer DNS sprawdza, czy otrzymane zapytanie dotyczy poddomeny za którą odpowiada.
  - jeśli tak – udziela odpowiedzi, korzystając ze swojej bazy danych
  - jeśli nie – podejmuje jedno z dwóch możliwych działań, w zależności od typu zapytania które przesłał klient:

Odwzorowywanie nazw – c.d.

- jeżeli klient zażądał pełnego tłumaczenia (tzw. rekurencyjnego odwzorowywanie nazw), to serwer DNS kontaktuje się z innym serwerem DNS, który potrafi obsłużyć zapytanie klienta, uzyskuje od niego odpowiedź i przesyła ją klientowi
- jeżeli klient zażądał nierekurencyjnego (tzw. iteracyjnego) odwzorowania nazw, serwer DNS informuje klienta o adresie następnego serwera z którym klient powinien się skontaktować

Odwzorowywanie nazw – c.d.

- Nielokalne odwzorowanie nazwy wymaga zazwyczaj albo skontaktowania się z serwerem głównym, albo skorzystania z informacji uzyskanej wcześniej od takiego serwera

Odwzorowywanie nazw – c.d.

- klient powinien znać adres przynajmniej jednego serwera DNS
- każdy serwer DNS musi znać adres przynajmniej jednego serwera głównego
- serwer DNS zazwyczaj zna również adres serwera DNS domeny bezpośrednio nadrzędnej (tzw. domeny macierzystej)

  (np. serwer domeny math.uni.lodz.pl zna adres serwera domeny uni.lodz.pl)

Odwzorowywanie nazw – c.d.

- W celu zapewnienia większej efektywności działania systemu serwery DNS przechowują w pamięci podręcznej informacje uzyskane od innych serwerów DNS (są to ostatnio odwzorowane nazwy wraz z informacjami, skąd została uzyskana odpowiedź).
- Mechanizm ten jest skuteczny, jeśli powiązania nazw z adresami nie zmieniają się zbyt często

Odwzorowywanie nazw – c.d.

- W przypadku otrzymania zapytania, którego nie można odwzorować lokalnie, sprawdzana jest zawartość pamięci podręcznej. Jeśli znajduje się w niej odpowiedź, serwer odsyła ją klientowi, informując, że jest ona nie w pełni wiarygodna (tzw. *non-authoritative answer*). Podaje również adres serwera, od którego ją otrzymał

Odwzorowywanie nazw – c.d.

- Serwery przechowują uzyskane informacje tylko przez jakiś czas. Jeśli po jego upływie serwer ponownie otrzyma zapytanie o usuniętą z pamięci podręcznej nazwę, to musi ponownie skontaktować się z odpowiednim serwerem DNS i uzyskać potrzebne informacje
- Czas przechowywania informacji w pamięci podręcznej określany jest przez serwer, od którego ta informacja została uzyskana

Rodzaje serwerów DNS

- Daną domenę może obsługiwać:
  - serwer „pierwszorzędny” dla tej domeny (tzw. *primary* lub *master server*), posiadający bazę danych opisującą zawartość domeny
  - jeden lub kilka serwerów pomocniczych (drugorzędnych, tzw. *secondary* lub *slave servers*), udzielające odpowiedzi na podstawie przechowywanej kopii bazy danych serwera głównego. Kopia ta jest okresowo uaktualniana
- Istnieją również tzw. serwery „keszujące” *(cache servers)*, nie posiadające własnych baz, a jedynie pamięć podręczną

Domeny odwrotne

- Serwery domen tzw. prostych (jak math.uni.lodz.pl, wp.pl) umożliwiają przetłumaczenie nazwy na adres IP
- Serwery domen tzw. odwrotnych *(reversed domains)* umożliwiają przetłumaczenie adresu IP na nazwę