MODUŁ 3:
Protokoły internetowe a system WWW

Cel modułu

Moduł dostarcza podstawowych informacji o architekturze i protokołach Internetu, istotnych z punktu widzenia programisty aplikacji WWW. Wyjaśniono założenia i zasady na których oparto protokoły internetowe. Wskazano aspekty decentralizacji architektury i jej rolę w zapewnieniu elastyczności i skalowalności Sieci.

Efekty kształcenia

Niniejsza lekcja służy osiągnięciu następujących efektów kształcenia.

Wiedza

Po ukończonym kursie student / studentka:

- Potrafi umiejscowić w czasie kluczowe wydarzenia historyczne związane z rozwojem Internetu,
- Charakteryzuje warstwowy model protokołów OSI i konfrontuje go z rodziną protokołów Internetu,
- Charakteryzuje najważniejsze protokoły internetowe,
- Objaśnia zasady działania systemu DNS,
- Charakteryzuje rodzaje adresowania w sieci Internet,
- Charakteryzuje protokół HTTP oraz zmiany, jakie wprowadzono w toku jego rozwoju,

Umiejętności

Po ukończonym kursie student / studentka:

- Korzysta z anglojęzycznych dokumentów specyfikacji standardów protokołów internetowych,
- Posługuję się narzędziami diagnostycznymi celem analizy komunikatów żądań i odpowiedzi HTTP.

Uzasadnienie

Podstawowa wiedza o protokołach jest istotna dla twórcy aplikacji WWW z co najmniej dwóch względów. Po pierwsze, wyznacza wspólny, jednolity niezależnie od użytych technologii programistycznych kontekst programowania aplikacji (w tym m.in. zestaw informacji na temat klienta, którym może dysponować aplikacja WWW). Po drugie – narzuca określone ograniczenia na model komunikacji i jej koszty, które należy uwzględnić dbając o należytą wydajność aplikacji.

Rola protokołów Internetu w budowie aplikacji WWW

W dotychczasowych modułach zajmowaliśmy się dokumentami znacznikowymi i ich formatowaniem, czyli, w kontekście aplikacji WWW, warstwą prezentacyjną oprogramowania. Związane z nią przetwarzanie (deklaratywne wiązanie formatowań z dokumentem znacznikowym) odbywało się bezpośrednio w ramach programu-konsumenta (czyli – przeglądarki WWW). Nie podejmowaliśmy jeszcze tematu programowania w JavaScript po stronie przeglądarki – jednak i tutaj sytuacja byłaby analogiczna: przetwarzanie po stronie konsumenta. Mogliśmy więc całkowicie abstrahować od tego, jak dokument i związane z nim zasoby trafiają do klienta. Sytuacją zmienia się diametralnie, gdy rozpatrujemy całość aplikacji WWW:

- Dyspozycje generowane po stronie klienta, które muszą trafić na serwer;
- Zakres informacji nt. żądania i nt. klienta, którymi dysponuje działające na serwerze oprogramowane;
- Dostarczenie do klienta treści przeznaczonych do zaprezentowania mu.

We wszystkich tych aspektach kluczową rolę odgrywa stosowany w komunikacji protokół.

Kilka istotnych dat

Zanim przejdziemy do szczegółów technicznych - przyjrzyjmy się kilku znaczącym datom wyznaczającym okres rozwoju omawianych tu technologii.

1945 - **Vannevar Bush – "As We May Think"**

Doniosły (przynajmniej - z dzisiejszej perspektywy) artykuł w czasopiśmie "Atlantic Monthly". Jest uważany za źródło koncepcji hipertekstu, na której oparto system WWW.

1958 - **ARPA**

Powstaje Advanced Research Project Agency (później przemianowana na DARPA). Podległa Departamentowi Obrony USA agencja powołana celem prowadzenia dużych przedsięwzięć badawczych o znaczeniu dla obronności.

1965 - **"On Distributed Communication Network"**

Autor - Paul Baran z RAND Corporation. Praca nakreśliła koncepcję zdecentralizowanej i odpornej na awarię/zniszczenie licznych węzłów sieci komputerowej.

1969 - **ARPA-NET**

W Stanach Zjednoczonych powstaje (uruchomienie pierwszych węzłów) eksperymentalna sieć komputerowa ARPA-NET.

1971 - **eMail**

Ray Tomilson tworzy pierwszy program poczty elektronicznej

1972 – **Telnet**

Kolejna ważna usługa sieciowa.

1973 – **Pierwsze połączenia międzynarodowe**

ARPA-NET osiąga zasięg międzynarodowy: połączenia do Wielkiej Brytanii i Norwegii

1974 – **"A protocol for Packet Intercommunication"**

Praca Vintona Cerfa i Boba Kahna opisująca koncepcję TCP; wprowadza też termin "Internet"

1975 - **ARPA-NET siecią na użytkową**

Sieć zmienia status z eksperymentalnej na użytkową.

1982 - **TCP/IP**

ARPA-NET wprowadza rodzinę protokołów TCP/IP

1984 - **DNS**

Wprowadzenie usługi DNS. Odtąd można stosować zamiast numerów IP nazwy domenowe dla identyfikacji komputerów w Internecie

1991 – **Polska dołączona do Internetu**

Pierwsze połączenie - z Danią

1990 - **Koncepcja WWW**

Tim Berners Lee (CERN) specyfikuje koncepcję systemu WWW i języka znaczników

1993 - **Mosaic**

Pierwsza przeglądarka WWW

1994 - **Spam**

Pierwszy znaczący przypadek spamu internetowego: f-ma prawnicza Canter i Siegel wysyła list do 6 tys. grup dyskusyjnych (usługi prawne związane z loterią pozwoleń na pracę w US)

1995 - **Netscape, Java**

Pojawia się przeglądarka Netscape oraz język Java

1995 - **WWW najpopularniejszą usługą**

WWW wyprzedza FTP stając się najpopularniejszą usługą Internetu

1996 - **XML**

Powstaje specyfikacja języka XML

1996 – **HTTP 1.1**

Znaczące udoskonalenie stosowanego w systemie WWW protokołu HTTP

2000 - **Web Services**

Zapoczątkowanie standaryzacji Web Services

Kontrola bieżąca	( )	*komentarz odpowiedzi*
System WWW narodził się:		
W drugiej połowie lat 1940-tych, wkrótce po pojawieniu się pierwszych komputerów		Nie. WWW stworzono już po narodzinach sieci Internet.
W pierwszej połowie lat 1970-tych – stanowił pierwszą usługę Internetu		Nie. Poczta elektroniczna, Usenet, czy usługi transferu pliku są znacznie starsze.
W połowie lat 1980-tych – wraz z usługą DNS		Nie. DNS jest bardzo ważny dla użyteczności WWW, ale te dwa systemy rozwinięto niezależnie.
W pierwszej połowie lat 1990-tych – przed stworzeniem języka Java	x	Tak. Ta niezwykle popularna i doniosła usługa jest wciąż stosunkowo młodym wynalazkiem.
Na przełomie XX i XXI wieku – jako składnik rodziny technologii Web Services		Nie. Web Services utworzono później, bazując już na ugruntowanych elementach systemu WWW.

Warstwowy model protokołów internetowych

Charakteryzując protokoły internetowe nie sposób pominąć klasycznego modelu warstwowego OSI (*Open Software Interconnection*). Nie stanowi on gotowego przepisu do powielenia przy projektowaniu warstwowego zestawu protokołów, jednakże w systematyczny i dogodny sposób dekomponuje złożony problem komunikacji elektronicznej na ułożone w hierarchię zagadnienia. Model ten wyodrębnił następujące warstwy problemu, począwszy od najniższej a skończywszy na najbardziej wyspecjalizowanej:

1. Fizyczna: jak sama nazwa wskazuje, jest reprezentowana przez fizyczne elementy infrastruktury; zapewnia przesyłanie nieprzetworzonych bitów danych przez nośnik
2. Łącza: zapewnia przesyłanie możliwych do zinterpretowania zestawów bitów pomiędzy dwoma węzłami sieci
3. Sieciowa: tworzy warstwę abstrakcji pozwalającą korzystać z intersieci mogącej składać się z różniących się technologiami i sposobami łączności sieci fizycznych; logiczne adresy zamienia na sprzętowe, pakiety logiczne dzieli na (stosowane w ramach warstwy 2) ramki danych
4. Transportu: zapewnia podział danych, gwarancję dostarczenia, regulowanie tempa przesyłu
5. Sesji: nawiązywanie i zamykanie połączeń pomiędzy współpracującymi węzłami
6. Prezentacji danych: format wymiany danych, ich ewentualna kompresja czy szyfrowanie
7. Aplikacji: usługi realizowane na rzecz konkretnych aplikacji.

Warstwy od 1 do 4 bywają łącznie nazywane transportowymi, zaś warstwy od 5 do 7 - aplikacyjnymi. Jak widać, model ten obejmuje zagadnienia komunikacyjne począwszy od przesyłania sygnałów rozpoznawalnych jako elementarne jednostki informacji w danym medium, poprzez ich grupowanie w większe porcje, następnie - zapewnienie ich jednolitej, niezależnej od platformy docelowej postaci, zapewnienie niezawodności przekazu, jego właściwej interpretacji a wreszcie - funkcji specyficznych dla danego zastosowania. Na szczególną uwagę zasługuje w tym modelu warstwa sieciowa, gdyż ujednolica ona komunikację i izoluje rozwiązania wyższego poziomu od heterogeniczności rozwiązań sprzętowych w integrowanych przezeń sieciach fizycznych.

Kontrola bieżąca	( )	*komentarz odpowiedzi*
Warstwa w modelu OSI, która dostarcza wspólnej abstrakcji ukrywającej szczegóły różnorodnych sieci fizycznych i wprowadzająca jednolity system logicznych adresów to:		
Warstwa aplikacji		Nie. Problem ten rozwiązujemy w niższej warstwie
Warstwa prezentacji danych		Nie. Problem ten rozwiązujemy w niższej warstwie
Warstwa sesji		Nie. Problem ten rozwiązujemy w niższej warstwie
Warstwa transportu		Nie. Problem ten rozwiązujemy w niższej warstwie
Warstwa sieciowa	x	Tak jest. To w tej warstwie występuje pojęcie intersieci
Warstwa łącza		Nie. Problem ten rozwiązujemy w wyższej warstwie
Warstwa fizyczna		Nie. Problem ten rozwiązujemy w wyższej warstwie

Internet i TCP/IP

Internet jest określany jako sieć wirtualna, realizująca model jednolitych usług. Oznacza to, że pozwala łączyć sieci fizyczne i dostarcza poziomu abstrakcji ukrywającego ich heterogeniczność, czyniąc z nich (z punktu widzenia budowanych nań usług) sieć wirtualną o wspólnym systemie adresowania oraz protokole. Rolę tego "wspólnego mianownika" pełni tu protokół IP (Internet Protocol) ze swym systemem abstrakcyjnych adresów protokołowych – numerów IP.

Architektura Internetu zapewnia daleko posuniętą decentralizację i odporność na usterki. Awaria pojedynczego komputera, czy nawet niedostępność pewnej sieci fizycznej mogą wciąż pozwalać na wyznaczenie alternatywnej ścieżki (*route*) pomiędzy komunikującymi się ze sobą maszynami. Scentralizowana do pewnego stopnia musi pozostać kontrola nad unikalnością adresów i nazw. Czuwa nad tym Internet Corporation for Assigned Names and Numbers (ICANN). Jak sama nazwa sugeruje, do obowiązków tego prywatnego podmiotu non-profit należy zapewnienie unikalnego przydziału adresów IP a także (zobacz dalej) nazw domenowych. Na bazie stworzonej przez protokoły komunikacyjne zdefiniowano protokoły wyspecjalizowane dla poszczególnych usług dla użytkowników (m.in. poczta, transfer plików, www ...).

Stos protokołów internetowych wyodrębnia mniej warstw niż wspomniany wcześniej model OSI, jednak opiera się na tej samej filozofii. Możemy tu wyróżnić 4 warstwy:

Połączeniowa - sterowniki urządzenia i karta interfejsu sieciowego; np. Ethernet, token ring. Realizuje ARP (Address Resolution Protocol) i RARP (Reverse Address Resolution Protocol), zapewniających w sieci lokalnej odwzorowania pomiędzy adresami logicznymi (jak numer IP) a adresami fizycznymi (jak MAC). Inaczej zwana warstwą interfejsu sieciowego. Pokrywa wszelkie szczegóły fizycznej komunikacji.

Sieciowa - Zwana też warstwą internetową. Obsługuje ruch pakietów poprzez sieć. Między innymi zachodzi tu routing pakietów. Obejmuje Internet Protocol (IP), a także m.in. Internet Control Message Protocol (ICMP), Internet Group Management Protocol (IGMP).

Transportowa - Obejmuje protokoły TCP (Transmission Control Protocol) oraz UDP (User Datagram Protocol). W przeciwieństwie do IP, którego funkcje realizowane są przez każdy węzeł na trasie pakietu, protokoły transportowe mają charakter end-to-end, co oznacza, że tylko nadawca o ostateczny odbiorca muszą realizować czynności związane z protokołem transportowym. TCP zapewnia wiarygodny (reliable) przepływ danych (dokonuje potwierdzenia ich dostarczenia, działa w trybie połączeniowym). Realizuje w tym celu ich porcjowanie, kontrolę czasu odpowiedzi, zestawianie połączenia. UDP przesyła datagramy bez gwarancji dostarczenia.

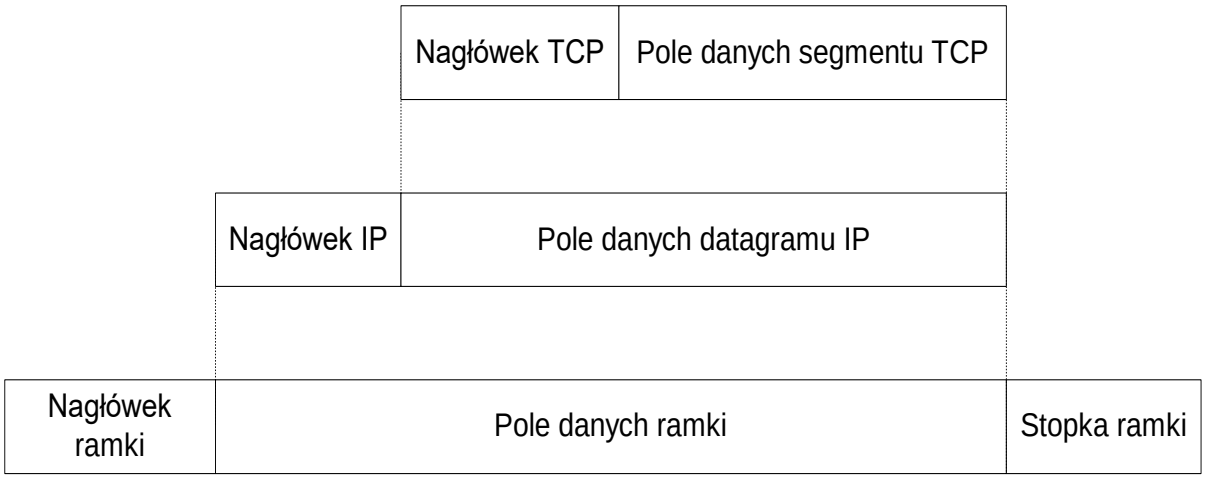
Aplikacji - Obejmuje specjalizowane protokoły dla aplikacji użytkowych (m.in. Telnet, FTP, SMTP, HTTP...).

Kontrola bieżąca	( )	*komentarz odpowiedzi*
Na podstawie powyższego zestawienia można zgadnąć, że popularna instrukcja diagnostyczna ping korzysta z protokołu pochodzącego z internetowej warstwy:		
Połączeniowej. Jego zasięg ogranicza się do sieci lokalnej.		Nie. Problem ten rozwiązujemy w wyższej warstwie
Sieciowej. Wykorzystuje protokół ICMP	x	Tak.
Transportowej. Dopiero ona pozwala dotrzeć do węzła docelowego		Nie. Problem ten rozwiązujemy w niższej warstwie
Aplikacji. Wykorzystuje protokół HTTP.		Nie. Problem ten rozwiązujemy w niższej warstwie

Hermetyzacja

Nawiązując do warstwowej organizacji protokołów internetowych, przytoczyć trzeba pojęcie hermetyzacji (czy enkapsulacji) przekazywanych komunikatów.

Protokół każdej warstwy wymaga (by mógł zrealizować ciążące na nim zadania) dodania do powierzonej mu wiadomości danych pomocniczych – np. adresu, informacji o sposób zakodowania danych, wymagania odnośnie sposobu dostarczenia, identyfikacja programu lub protokołu odpowiedzialnego za przetworzenie komunikatu po stronie adresata itp. Umieszczane są w nagłówku i ew. w stopce przekazywanej jednostki danych. Zob. poniższy rysunek.



Warstwa sieciowa umieszcza w nagłówku m.in. adresy IP nadawcy i odbiorcy, pozwalające na skuteczne dostarczenie datagramu do adresata. Nagłówek zawiera też informację, jakiego rodzaju zawartość znajduje się w ciele datagramu – pozwala to rozstrzygnąć, który protokół po stronie docelowego odbiorcy powinien zostać użyty celem zinterpretowania wiadomości.

Warstwa transportowa natomiast umieszcza (obok szeregu informacji pomocniczych zapewniających niezawodność i odpowiednie tempo przesyłania) w nagłówku 16-bitowe numery źródłowego i docelowego portu. Identyfikują one, w ramach zidentyfikowanych numerami IP maszyn, konkretne procesy, które ze sobą się komunikują.

Z kolei demultiplexing jest procesem odwrotnym po stronie odbiorcy: po otrzymaniu ramki kolejne warstwy „rozpakowują" dane i w oparciu o identyfikatory w znanych sobie nagłówkach przekazują odpowiedniemu protokołowi warstwy wyższej.

Prześledźmy w uproszczeniu ten proces enkapsulacji i demultiplexingu na przykładzie żądania pobrania strony WWW. Po stronie maszyny klienta, wykonywane jest, co następuje:

1. Na podstawie podanego w pasku adresowym przeglądarki adresu URL – ustalany jest numer IP maszyny, na której działa serwer, a także numer portu, na którym ów serwer nasłuchuje.
2. Przygotowywana jest, jako łańcuch tekstowy, odpowiednia treść żądania pobrania strony w protokole HTTP
3. Treść ta jest umieszczana jest w ciele segmentu TCP, zaś do nagłówków TCP trafia informacja o portach nadawcy i odbiorcy.

4. Segment IP trafia jako ciało utworzonego dlań datagramu IP; nagłówki są wypełniane m. in. informacjami o adresach IP nadawcy i odbiorcy oraz o wskazanie, że zawartością datagramu jest segment TCP.
5. Tak skonstruowany datagram jest powierzany ramce sieci zgodnie z technologią danej sieci fizycznej (np. Ethernet); nagłówek ramki identyfikuje w ramach danej sieci fizycznej węzeł docelowy (może nim być ów finalny adresat – u nas: serwer WWW), bądź też router, który przekaże datagram poza granice danej sieci fizycznej.

Proces demultipiexingu przebiega analogicznie.

Widzimy więc, że powyższy rysunek nie wyczerpuje pełnego stosu enkapsulacji omawianego w tym przykładzie. Wewnątrz segmentu TCP mielibyśmy jeszcze nagłówki i ewentualnie również (zależnie od rodzaju komunikatu) ciało komunikatu (a w nim z kolei – jeśli byłaby to odpowiedź z serwera – moglibyśmy napotkać elementy nagłówkowy i ciała składające się na dokument HTML).

Zauważmy, że pojęcie hermetyzacji, choć ma tu nieco odmienne znaczenie, jest motywowane tym samym, co np. hermetyzacja w językach programowania: dany protokół traktuje zawartość komunikatu z jej informacjami przeznaczonymi dla innych protokołów jako nieprzejrzystą całość i może abstrahować od jej budowy.

Kontrola bieżąca	( )	*komentarz odpowiedzi*
Numer portu identyfikującego adresata znajdziemy w:		
Stopce ramki		Nie. Dane te nie są potrzebne warstwie połączeniowej
Nagłówku ramki		Nie. Dane te nie są potrzebne warstwie połączeniowej
Nagłówku IP		Nie. Dane te nie są potrzebne warstwie sieciowej
Nagłówku TCP	x	Tak. Warstwa transportowa identyfikuje za pomocą numeru portu proces-adresata
Nagłówku HTTP		Nie. Dane te muszą być przetworzone w niższej warstwie

Adresowanie

Adres IP zapewnia jednolity sposób identyfikacji maszyn w sieci Internet. Stanowi on adres logiczny, który celem zidentyfikowania adresata musi być ostateczne odwzorowany na adres sprzętowy w danej sieci fizycznej. Jeśli identyfikuje on maszynę w bieżącej sieci, można dokonać odwzorowania. W przeciwnym razie - datagram trafi do maszyny zwanej routerem działającej na granicy danej sieci fizycznej, która na podstawie adresu IP określi kolejnego pośrednika, któremu należy go przekazać.

Adres IP w stosowanym aktualnie powszechnie protokole IPv4 składa się z 4 oktetów (dla poprawy czytelności dla człowieka - zapisywanych zwykle jako 4 liczby dziesiętne z przedziału 0-255) - łącznie z 32 bitów. Adres ów musi zawierać rozróżnienie adresu sieci i adresu maszyny w tej sieci. W zależności od wielkości sieci - podział w sensie liczby bitów identyfikujących sieć i maszynę w tej sieci może być różny. Pierwotnie był to sztywny podział, to znaczy - wynikający z samej wartości adresu. Mowa tu o tzw. podejściu klasowym, które wyróżniało 5 klas adresów. Ponadto, cała pula 127.*.*.* stanowi adresy pętli zwrotnej - identyfikujące maszynę, która dokonuje odwołania do takiego adresu.

Wraz z szybkim rozrostem Internetu, powyższe podejście klasowe wykazało znaczący brak elastyczności. Było nazbyt rozrzutne w sensie przydzielania zakresów adresów IP (których w IP v4 nie jest znów tak wiele); okazało się również nieoptymalne z punktu widzenia rozrostu tabel routingu

utrzymywanych przez routery. Dla podniesienia efektywności wykorzystania puli adresów w IPv4 wprowadzono specyfikację bez-klasowego (classless) podejścia do adresowania, w którym liczba bitów określających tzw. maskę sieci jest określana explicite – np. 192.9.205.22 /18

Wzięto też pod uwagę potrzeby zaadresowania maszyn w sieciach lokalnych połączonych z Internetem tak, aby miały one możliwość dostępu do jego zasobów, ale nie były pełnoprawnymi członkami Sieci z unikalnym adresem. Zrodziło to potrzebę wyodrębnienia adresów, które mogłyby się powtarzać pomiędzy różnymi sieciami prywatnymi poszczególnych podmiotów, ale zarazem - by nie powodowały konfliktu z publicznymi adresami w Internecie. Wyodrębniono zatem 3 pule adresów (zaszłość historyczna – aby adresy takie były dostępne w każdej z klas od A do C):

od 10.0.0.0 do 10.255.255.255

od 172.16.0.0 do 172.31.255.255

od 192.168.0.0 do 192.168.255.255

Maszyny dysponujące takimi adresami mogą korzystać z zasobów Internetu dzięki zastosowaniu odpowiednich technik NAT (Network Address Translation).

Sposoby adresowania sieci, maszyn i ich zasobów w Internecie możemy podsumować w postaci następującej listy:

Adresy fizyczne (np. Ethernet-owy MAC – liczba 48 bitowa) - pozwalają one identyfikować maszyny w lokalnej sieci. Dzięki kolejnym warstwom modelu protokołów, programista aplikacji internetowej jest odizolowany od tych szczegółów adresowania.

Adresy IP (w IPv4 – np. 148.81.141.16) – umożliwiają odwołanie się do odległej maszyny bez znajomości jej adresu sprzętowego. Są podstawą routingu.

Numery portów – określają sposób interpretowania komunikatu. Pozwalają mianowicie odbiorcy rozstrzygnąć, który z działających po jego stronie programów powinien zająć się przetworzeniem komunikatu.

Nazwy domenowe – stanowią przyjazny człowiekowi ekwiwalent adresów IP (przybierają postać tekstową i są zorganizowane hierarchicznie w ramach zdefiniowanych domen głównych). Są translowane przez usługę DNS (zob. dalej) na adresy IP.

Identyfikatory w protokołach aplikacyjnych – precyzują odwołania określając zasób np. nazwa skrzynki e-mail na danym serwerze pocztowym, lokalny adres konkretnej strony na serwerze WWW.

<< ANIMACJA A1: wysyłka żądania pobrania strony zilustrowana w poszczególnych warstwach komunikacji – zob. załącznik >>

Zasada lokalności w sieciach komputerowych

W tradycyjnych usługach w sieci można zaobserwować lokalność i powtarzalność odwołań pomiędzy maszynami. Obserwację tę opisano jako zasadę lokalności odwołań: dany komputer częściej komunikuje się z jednostkami fizycznie nieodległymi. Uzupełnieniem tej prawidłowości jest czasowa lokalność odwołań: dany komputer często wielokrotnie (tj. ponownie) komunikuje się z tymi samymi maszynami. Zasada ta motywuje rozwijanie sieci lokalnych oraz optymalizowanie komunikacji na okoliczność ponownego odwołania do tej samej maszyny. Jak zobaczymy, przejawem tej zasady są decyzje projektowe w protokołach a także sposób organizacji usługi DNS.

Protokoły aplikacyjne

DNS

System Domain Name System (DNS) zaliczany jest do warstwy aplikacyjnej, choć faktycznie kojarzy się z infrastrukturą Internetu (jest intensywnie wykorzystywany przez szereg innych protokołów aplikacyjnych; jest też logicznie mocno związany z warstwą sieciową). Słowem – jest realizowany w warstwie aplikacyjnej, ale służy zdobyciu informacji służących komunikacji w warstwie sieciowej.

DNS stanowi rozproszoną bazą danych, zawierającą odwzorowania nazw oraz ich aliasów na adresy IP. Można powiedzieć, że jest to jedna z większych i najintensywniej wykorzystywanych rozproszonych baz danych. Nazwy domenowe tworzą hierarchię:

- Serwery główne DNS (są one również zreplikowane - 13 sztuk) znają adresy serwerów odpowiedzialnych za poszczególne domeny główne itd.
- Serwer obsługujący domenę może oddelegować obsługę „poddrzewa" hierarchii do innego serwera.
- Organizacja, której przydzielono daną domenę może zatem samodzielnie i elastycznie obsługiwać swoje pod-domeny.

Zgodnie z zasadą lokalności – odwzorowanie nazwy zaczyna się na serwerze DNS lokalnym dla danego użytkownika (który samodzielnie dysponuje wiedzą o odwzorowaniach nazw z lokalnej domeny). Z kolei, zgodnie z zasadą czasowej lokalności odwołań wyniki odwzorowania są buforowane przez zadany czas. Buforowanie radykalnie poprawia wydajność funkcjonowania systemu - zwłaszcza w ewaluacji adresów popularnych domen.

Kontrola bieżąca	( )	*komentarz odpowiedzi*
System DNS scharakteryzujemy jako:		
Rozproszoną, hierarchicznie zdekomponowaną bazę danych	X	Tak.
Scentralizowaną bazę danych zreplikowaną na 13 serwerów		Nie. Rozmiar i dynamika zmian zbioru nazw domenowych nie byłyby do udźwignięcia przez pojedynczy serwer
Rozproszoną bazę danych podzieloną na serwery według adresów sieci fizycznych, których dotyczą wpisy		Nie. Kluczem podziału danych nie jest tutaj fizyczna budowa sieci

SMTP

Simple Mail Transfer Protocol (SMTP) - realizuje niezawodne przesłanie wiadomości (nadawca przechowuje kopię do momentu upewnienia się, że odbiorca zapisał list w pamięci nieulotnej). Ponadto następuje sprawdzenie, czy skrzynka docelowa istnieje.

Protokół ten bazuje na funkcjonalności dostarczanej przez niższe warstwy, toteż jego specyfikacja może się skoncentrować na określaniu sposobu kodowania znaków, oraz informacji specyficznej dla wiadomości e-mail. Format zawartości zdefiniowano jako dokument tekstowy o postaci:

- Nagłówek - stanowiący słowo kluczowe - np. `From`, `To`, `CC`, `Date`, `Subject`, `Reply-To`, `X-Charset`, `X-Mailer`, `X-Sender`, `X-Face`; po nim dwukropek i tekst stanowiący wartość nagłówka
- pusty wiersz
- właściwa zawartość

W tej postaci standard został sformułowany w roku 1982 i spełnił ówczesne wymogi - ograniczone do wymiany tradycyjnych tekstowych wiadomości.

MIME

Upowszechnienie usługi poczty elektronicznej oraz zwiększenie możliwości przetwarzania otworzyły potrzeby na przesyłanie plików binarnych, w tym programów, obrazów czy dźwięków. Oznaczało to, że przyjęte w oryginalnej wersji SMTP ograniczenia na objętość przesyłek, zbiory znaków czy też format wiadomości muszą zostać zrewidowane. W szczególności, potrzebne stało się umożliwienie przesyłania wiadomości z załącznikami oraz określenie sposobu kodowania załączników binarnych. W tym celu, odrębna specyfikacja zwana Multipurpose Internet Mail Extensions zdefiniowała dodatkowe wiersze nagłówkowe oraz ustaliła otwarty system definiowania różnego typu zawartości. Umożliwiło to:

- przesyłanie wielu odrębnych obiektów w ramach pojedynczego listu
- usunięcie ograniczeń na długość wiersza oraz rozmiar całej przesyłki
- stosowanie zbiorów znaków innych niż ASCII
- przesyłanie w załącznikach różnych mediów i formatów dokumentów

MIME określa:

- Wiersz nagłówkowy identyfikujący rozszerzenie: MIME-Version: 1.0
  Predefiniowany zestaw głównych typów zawartości: text, multipart, application, message, image, audio, video
- Podtypy zawartości i nagłówek `Content-Type:` służący określaniu rodzaju zawartości
  Mechanizm kodowania dla zapewnienia zgodności z ograniczeniami transportów na znaki spoza zestawu ASCII. Format ów, zwany BASE64 reprezentuje każde trzy bajty w postaci 4 drukowalnych znaków ASCII. Stąd znane nam zjawisko zwiększenia objętości przesyłki ponad rozmiar zajmowany na dysku przez plik załącznika. Określeniu sposobu zakodowania załącznika służy nagłówek `Content-Transfer-Encoding:`
  Specjalny tym zawartości mieszanej, pozwalający na dowolne zagnieżdżanie załączników. Pozwala definiować separator. Wynikowa definicja w nagłówku może przybrać postać:

```
Content-Type: Multipart/Mixed; Boundary=”do70ciu-znakow-drukowanych”
```

Przykłady innych protokołów aplikacyjnych

Poniżej przedstawiamy kilka kolejnych, popularnych protokołów aplikacyjnych:

POP (Post Office Protocol)

Umożliwia pobieranie poczty z odległego serwera

FTP (File Transfer Protocol)

Transfer plików. Oparty zwykle na TCP. Połączenie sterujące pozostaje otwarte podczas całej sesji. Połączenia danych są tworzone oddzielnie dla każdego polecenia przesłania pliku.

TFTP (Trivial FTP)

Minimalistyczny zamiennik FTP, minimalizujący ruch w sieci. Oparty na UDP. Brak interakcyjności. Możliwość przesłania w dowolnym kierunku kopii pliku. Może być wykorzystany jako protokół sprzętowego ładowania systemu.

NFS (Network File System)

Oferuje zwykłe operacje plikowe. Zintegrowanie usługi z lokalnym systemem plików sprawia, że różne programy mają możliwość pracy na plikach odległych. Możliwość blokowania pliku dla zapewnienia bezpieczeństwa współbieżnego dostępu. Znacznie bardziej efektywny przy konieczności dostępu i np. modyfikacji odległego pliku.

System WWW

System WWW (World Wide Web) powstał stosunkowo niedawno (początek lat 1990-tych) i szybko stał się najpopularniejszą usługą Internetu (już w roku 1995 wyprzedził usługę FTP). Z powodu swej popularności oraz braku wygodnego polskiego odpowiednika terminu Web – w mowie potocznej termin ten często mieszany jest z pojęciem Internetu. Do specyfiki WWW należy m.in. to, że zrywa on z zasadą lokalności odwołań. Wraz z rozwojem tego systemu obserwujemy jego potężny wpływ na działalność gospodarczą na kolejno osiąganych płaszczyznach:

- Nowy kanał komunikacji marketingowej
- Platforma dla handlu elektronicznego (e-commerce)
- Narzędzie integracji informatycznej w modelu business-to-business (B2B)

Warstwa komunikacji z użytkownikiem systemu WWW zakłada tzw. interfejs point and click, a więc wykorzystanie urządzeń wskazujących. WWW jest określany jako system hipermedialny, tzn. rozszerzenie (obecność innych niż tekst postaci informacji) koncepcji hipertekstu. Dokumenty administrowane są niezależnie, stąd musimy się liczyć z potencjalnie niską stabilnością odsyłaczy prowadzących do cudzych zasobów. Zauważmy, że scenariusz działania serwera jest dość prosty: udostępnianie w odpowiedzi na komunikat klienta żądanych przezeń dokumentów. (Zastosowanie technologii dokumentów dynamicznych WWW znacząco zmienia jednak ten stan rzeczy. Do tego wątku powrócimy w jednym z kolejnych wykładów.) Przeglądarka stanowi zaś stosunkowo złożone narzędzie. Oczekujemy od niej zrealizowania następujących komponentów:

- moduł sterujący, reagujący na sygnały z urządzeń wejściowych
- klient protokołu HTTP
- interpreter języka HTML
- inne, opcjonalne interpretery i klienty (FTP, POP, SMTP)

Zasoby w systemie WWW są identyfikowane poprzez adres URL (Uniform Resource Locator). W jego budowie można wyróżnić następujące elementy:

- określenie schematu adresowania: "http"
- adres hosta (nazwa domenowa lub – nie zalecane – numer IP)
- Numer portu (domyślnie 80 - można wówczas pominąć)
- Ścieżka bezwzględna do zasobu (wirtualna!)
- Ew. parametry żądania (tzw. *query string* - umieszczane po symbolu "?")

Przykładowo, adres URL może przybrać więc np. postać:

```
http://www.example.org:8080/publications.php?lang=pl&yr=2000
```

Należy też dodać, że dla adresów URL obowiązują reguły składniowe, które nie pozwalają na bezpośrednie użycie pewnych znaków. Znaki taki wymagają wobec tego zakodowania (tzw. *URL-encoding*) za pomocą symbolu "%" liczby zapisanej w systemie heksadecymalnym. Np. symbol spacji może zostać zastąpiony przez "`%20`" Treść znajdująca się w systemie WWW nie musi ograniczać się do zawartości o charakterze statycznej. Strony WWW mogą zawierać kod programów wykonywanych

po stronie serwera przy okazji pobrania przez klienta strony spod danego adresu, albo po stronie klienta - po załadowaniu zawartości do środowiska przeglądarki.

Protokół HTTP

Standard określa zastosowanie protokołu HTTP (Hypertext Transfer Protocol) w dość ogólny sposób, jako "transfer różnego rodzaju zasobów poprzez sieć". Projekt tego protokołu musi uwzględnić specyfikę systemu WWW, a mianowicie fakt, że przeglądanie dokumentów hipermedialnych nie wykazuje istotnej lokalności odwołań. Mamy tu zatem do czynienia z inny wzorcem odwołań niż dla pozostałych rodzajów programów użytkowych. Protokół został wobec tego zaprojektowany oryginalnie jako bezstanowy (brak pojęcia sesji grupującej interakcje). Interakcja przeglądarki z serwerem WWW odbywa się według modelu bezpołączeniowego:

- żądanie jest wysyłane przez klienta
- Serwer przekazuje (zawsze z inicjatywy klienta) żądane zasoby lub informacja o ich niedostępności
- połączenie zostaje zamknięte

Komunikacja w protokole HTTP jest przekazywana w warstwie transportu za pośrednictwem protokołu TCP, który gwarantuje niezawodność przesyłania, ale też generuje istotne narzuty na nawiązanie połączenia (tzw. *handshake*) i jego późniejsze zamknięcie.

Protokół określa format żądania oraz odpowiedzi. Budowa komunikatów wykazuje podobieństwa z omawianym wcześniej protokołem SMTP. Podobnie jak tam, mamy tu do czynienia z częścią nagłówkową oraz (opcjonalnie) z oddzielonym od niej za pomocą pustego wiersza ciałem komunikatu.

Zarówno nagłówki jak i ciało komunikatu mogą wystąpić zarówno w żądaniu od klienta, jak i w odpowiedzi przesyłanej przez serwer WWW (przy czym niektóre rodzaje żądań i odpowiedzi wykluczają przez swoją naturę obecność ciała komunikatu). Ponieważ zadanie zlokalizowania serwera ciąży na protokołach niższych warstw, żądanie pobrania dokumentu WWW (w starszym protokole w wersji 1.0) może przybrać formę tak prostą jak:

```
GET lokalna_sciezka_i_nazwa_pliku_na_serwerze HTTP/1.0
[pusty wiersz]
```

Należy podkreślić, że zakres funkcjonalności protokołu http znacząco wykracza poza samo pobieranie zasobów znajdujących się na serwerze. Między innymi, projektanci przewidzieli obsługę poprzez żądania HTTP pełnego zestawu operacji „CRUD” (czyli Create, Retrieve, Update, Delete), definiując odpowiednie metody żądań:

PUT

Umieszczenie zasobu na serwerze. Ciało komunikatu zawiera zasób, zaś polecenie precyzuje URL, pod którym zasób ów powinien zostać umieszczony.

GET

Służy pobieraniu zasobów. Nie przewiduje ciała komunikatu.

POST

Przekazuje dane, które mają zmodyfikować stan zasobu wyspecyfikowanego w żądaniu. Komunikat zawiera ciało.

DELETE

Usuwa zasób skojarzony z adresem URL podanym w żądaniu

Ten zestaw funkcjonalności upodabnia nieco HTTP do protokołu FTP. Ponieważ jednak aktualizacja zasobów na serwerze przebiega obecnie często za pomocą innych środków, powinniśmy w pierwszej kolejności zwrócić uwagę na następujące trzy metody szeroko wykorzystywane przez oprogramowanie klienckie:

GET

Posiada jedynie wiersz polecenia i nagłówki. Przypomnijmy, że parametry precyzujące żądanie mogą zostać dołączone do URL w postaci query string. Zgodnie z nazwą ma służyć jedynie do pobierania zasobów, bez powodowania efektów ubocznych. Takie założenie znajduje odzwierciedlenie w sposobie obsługi żądania przez serwer i przez oprogramowanie klienckie (w szczególności wiąże się to z możliwością buforowania pobranych wcześniej rezultatów identycznego żądania).

POST

Służy zaktualizowaniu zasobu na serwerze. Parametry mogą być wysyłane w ciele komunikatu. Opisany dodatkowo nagłówkami `Content-Type` (np. `application/x-www-form-urlencoded`) oraz `Content-Length`. Wołana lokalizacja (URL) wskazuje zwykle na program obsługujący, który jest konsumentem polecenia aktualizacyjnego.

HEAD

Działa podobnie jak GET, ale służy jedynie sprawdzeniu dostępności zasobu: zwracany w odpowiedzi jest komunikat nieposiadający ciała. Nagłówki pozwalają określić datę ostatniej modyfikacji zasobu oraz sugerowany czas, przez jaki pobierający go klienci powinni buforować aktualną wersję. Udogodnienie to odgrywa istotną rolę w zredukowaniu ruchu w sieci. Więcej na temat protokołu HTTP, jego rozszerzeń oraz sposobu wykorzystania przez programistów – dowiemy się w toku kolejnych wykładów.

Kontrola bieżąca	( )	*komentarz odpowiedzi*
Spośród wymienionych wyżej metod żądań HTTP następujące metody przewidują występowanie ciała komunikatu:		
PUT, POST, HEAD, GET, DELETE (wszystkie wymienione)		Odpowiedź błędna.
GET, POST, HEAD		Odpowiedź błędna
PUT, POST	x	Tak.
POST		Odpowiedź niekompletna.
PUT		Odpowiedź niekompletna.

HTTP 1.1 i dalszy rozwój protokołów dla aplikacji WWW

Pierwotny kształt specyfikacji HTTP, sformalizowany w wersji 1.0, powstawał w zupełnie innych realiach użytkowania WWW. Tworzono go z myślą o relatywnie prostych (być może wręcz czysto hipertekstowych stronach), które:

- mogły być pobrane pojedynczym żądaniem;
- zawierały hiperłącza prowadzące do innych dokumentów tej samej witryny, lub do
- zewnętrznych serwisów.

Dla tego typu zastosowań bezpołączeniowa i bezstanowa natura HTTP 1.0 była wystarczająca, a wręcz – optymalna. Sytuację tę zmienił radykalnie następujące czynniki:

- coraz bogatsze strony WWW (zawierające powiązane zasoby: arkusze CSS, skrypty, elementy graficzne, zdjęcia itp.) – wymagające często dziesiątek żądań do pobrania kompletnego dokumentu (realizowane odrębnie wymagałyby dla każdego transferu wynegocjowania na nowo połączenia TCP);
- szybki wzrost popularności WWW – także wśród mniejszych firm, które nie potrzebowałyby osobnej dedykowanej maszyny dla swojego serwera WWW;
- realizowanie funkcjonalności aplikacyjnej poprzez interfejs WWW – konieczność zapamiętania kontekstu interakcji z klientem (np. fakt zalogowania, ustawienia preferencji itp.).

Potrzebom tym wyszedł naprzeciw HTTP 1.1, oferując:

- Na potrzeby hostingu wielu witryn na tym samym serwerze – dodatkową identyfikację żądania poprzez dodanie nagłówka `Host:` .
- Na potrzeby pobierania wielu zasobów związanych z dokumentem – możliwość zadysponowania podtrzymania otwartego połączenia TCP na potrzeby przyszłych żądań.
- Celem zaoszczędzenia na nadmiarowych transferach – dodatkowe informacje nagłówkowe i opcje żądań wspierające buforowanie wcześniej pobranych zasobów.
- Dla usprawnienia transferu dokumentów dynamicznych – możliwość przesyłania w częściach (serwer może wysłać fragment strony w czasie, gdy jeszcze generowana jest pozostała część jej zawartości).
- Dla podtrzymania wybranych danych o przebiegu dotychczasowej interakcji danego klienta z witryną – mechanizm przekazywania w nagłówkach protokołowych tzw. cookies.

Część powyższych zagadnień zostanie rozwinięta w kolejnych modułach. W szczególności będziemy posługiwać się mechanizmami opartymi na cookies.

<<SCREENCAST S3 – analiza komunikacji HTTP z serwerem WWW przy użyciu narzędzia typu web-debugger>>

Na koniec zauważmy, że ww. usprawnienia protokołu HTTP nie wyczerpują tematu pożądanych udoskonaleń. Protokół pozostaje nadal stosunkowo rozrzutny w sensie generowanego ruchu w sieci, zaś nowe zastosowania (m.in. gry, transmisje na żywo, narzędzia współpracy zdalnej itp.) rodzą nowe zapotrzebowania dotyczące protokołu. To tych zagadnień powrócimy w jednym z kolejnych modułów.

Kontrola bieżąca	[ ]	*komentarz odpowiedzi*
Którym nowym wyzwaniom wychodzi naprzeciw specyfikacja protokołu HTTP w wersji 1.1?		
Kompresja danych		Odpowiedź błędna
Usprawnienie pobierania stron składających się z wielu powiązanych zasobów	x	Poprawna odpowiedź. / Niekompletna odpowiedź.
Wsparcie dla przesyłania XML		Odpowiedź błędna
Hosting wielu witryn pod pojedynczym adresem IP	x	Poprawna odpowiedź. / Niekompletna odpowiedź.
Rozpoczęcie wysyłania odpowiedzi w trakcie, gdy nie ukończono jeszcze całości przetwarzania żądania	x	Poprawna odpowiedź. / Niekompletna odpowiedź.

Zadanie

Pozyskaj i zapoznaj się z podstawową funkcjonalnością wybranego narzędzia diagnostycznego do analizy komunikatów protokołu HTTP (narzędzia te zwane są zwykle Web-debugger). Może to być jeden z komponentów narzędzi programistycznych stanowiący integralną część przeglądarki, lub też odrębne narzędzie (np. https://www.telerik.com/fiddler/fiddler-classic).

Za pomocą tego narzędzia sprawdź, jaka liczba i jakich wywołań protokołu http następuje przy pobieraniu strony początkowej wybranej witryny.

Sprawdź, jak zachowuje się deklarowana w nagłówkach data ostatniej modyfikacji pobieranego zasobu dla stron HTML (pamiętajmy, że obecnie bardzo często są to dokumenty dynamiczne, czyli generowane po stronie serwera WWW w odpowiedzi na żądanie), a jak dla innych, zapewne rzadziej zmieniających się zasobów (arkusze CSS, elementy graficzne itp.).

Na przykładzie którejś z publicznie dostępnych witryn znajdź dokument, który posłuży za przykład ilustrujący pobieranie warunkowe zasobu (żądanie „pobierz, o ile zmodyfikowany po <data, godz>”). Przedstaw, jak jest zbudowane żądanie, które zostało wysłane przez klienta na serwer oraz załącz nagłówki (plus informacja, czy wystąpiło ciało komunikatu, czy nie) odpowiedzi dla dwóch wariantów:

a) Gdy klient podaje datę wcześniejszą niż data ostatniej modyfikacji tego zasobu na serwerze;
b) Gdy klient podaje datę późniejszą.

Do ww. rozwiązania zadania dołącz objaśnienia znaczenia wszystkich nagłówków protokołowych, które znalazły się w żądaniu i obu wersjach odpowiedzi.

W miarę potrzeb – staraj się sprawdzić znaczenie odpowiednich konstrukcji w specyfikacji protokołu HTTP 1.1 dostępnej m.in. poprzez witrynę http://www.rfc-editor.org