Systemy webowe

2011/2012

Leszek Borzemski

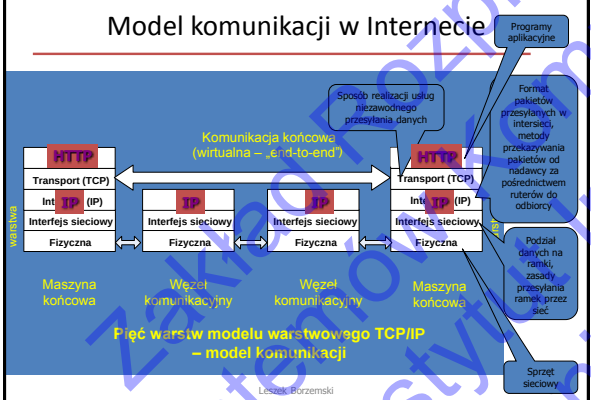
Zapraszamy do współpracy!

Badania Internetu prowadzone w Zakładzie Rozproszonych Systemów Komputerowych (ZRSK) w Instytucie Informatyki Politechniki Wrocławskiej

(lata 2001+)

www.zrsk.pwr.wroc.pl

Model komunikacji w Internecie



Badania Internetu w ZRSK (2001+)

- Platformy pomiarowe WING i MWING
- Systemy Webowe z Jakością Usług
- Rozproszone Laboratorium Data Mining
- Laboratorium Pomiarów i Eksploracji Internetu
- Projekt COST ComplexHPC
- ...

Wykaz projektów (wybór)

- **Badania Internetu na poziomie IP oraz HTTP – własne systemy pomiarowe TRACE, WING, MWING, IMES i VEGA - wykorzystanie klasycznej analizy danych i technik eksploracji danych**
- **Poziom HTTP**
  - **Pomiary i wizualizacja transakcji webowych - usługa sieciowa WING**
  - **WING w badaniach wydajności i niezawodności Internetu na poziomie protokołu HTTP**
  - **Efektywne pozyskiwanie zasobów WWW**
  - **Predykcja przepustowości połączenia klient-serwer WWW**
  - **Badania popularności serwerów WWW – projekty VEGA i Athena**
  - **Dystrybucja żądań HTTP - algorytm FARD, algorytm GARDiB dla CDN, przełącznik WebDispatcher, badania ewaluacyjne przełączników webowych**
  - **Badania jakości usługi WWW – sterowanie dostępem i szeregowanie żądań HTTP na serwerze WWW**
- **Poziom IP**
  - **System TRACE w badaniach niezawodności i wydajności Internetu na poziomie protokołu IP**
  - **Odkrywanie struktury Internetu (tomografia internetowa)**
- **Inne: np. wyszukiwarka SearchSystem, sieci MPLS**
- **Grant „Systemy webowe z jakością usług" (2006-2009) – m.in. system MWING**
- **Projekt w Programie COST ComplexHPC (2009-2013)**
- **Laboratorium Distributed Data Mining oraz Internet Measurement & Exploration**

Pomiary i Eksploracja Internetu

- Pomiary wydajności
- Pomiary topologii
- Pomiary stanu

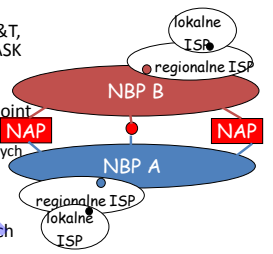
- Odkrywanie topologii
- Analiza predykcyjna wydajności transferu danych
- Dystrybucja żądań HTTP
- Efektywne dostarczanie treści internetowych
- Technologie HPC z użyciem Cloud Computing

Co to jest Internet?

- Każdy to przecież wie!
- Definicja (wg RFC 1462; Internet Engineering Task Force, maj 1993 r.):
  - Aspekt techniczny: Połączone sieci (*ang. network of networks*) oparte na protokołach TCP/IP;
  - Aspekt społeczny: Społeczność (*ang. community of people*), która używa i rozwija tę sieć;
  - Aspekt informacyjny: Zbiór zasobów (*ang. collection of resources*), które znajdują się w tej sieci;
  - „Używanie Internetu" to nic innego jak działanie członków społeczności przy pomocy sieci, mające na celu odnalezienie i wykorzystanie znajdujących się w niej zasobów informacyjnych.

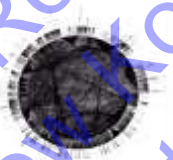
Struktura Internetu

- słabo zhierarchizowana globalna sieć sieci
- narodowi/międzynarodowi dostawcy sieci szkieletowych (backbone) (NBPs, IBPs)
  - np. w USA: BBN/GTE, Sprint, AT&T, IBM, UUNet; w Polsce: TPSA, NASK
  - połączone jak równy z równym prywatnie lub za pomocą państwowych Network Access Point (NAPs)
- regionalni dostarczyciele usług internetowych ISPs
  - podłączone do NBPs
- lokalne ISP
  - firmy podłączone do regionalnych ISPs

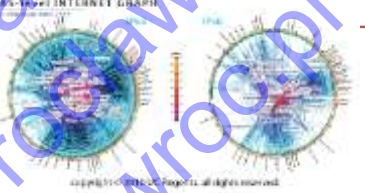


Struktura Internetu

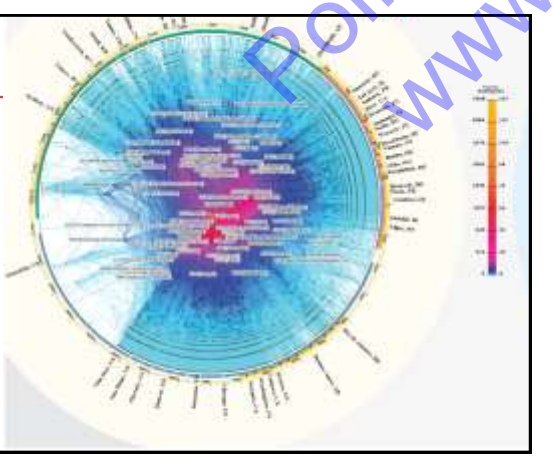
- Systemy (obszary) Autonomiczne (ang. Autonomous Systems, AS)
- ASN – numer AS
- 50 K aktywnych AS



www.caida.org

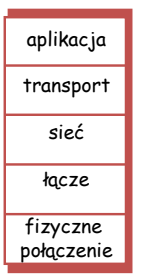


„This visualization represents macroscopic snapshots of IPv4 and IPv6 Internet topology samples captured in June 2010. For the IPv4 map, CAIDA collected data from 45 monitors located in 24 countries on 6 continents. For the IPv6 map, CAIDA collected data from 12 Ark monitors located in 6 countries on 3 continents.
Date added: 2011-04-21"



Stos protokołów internetowych

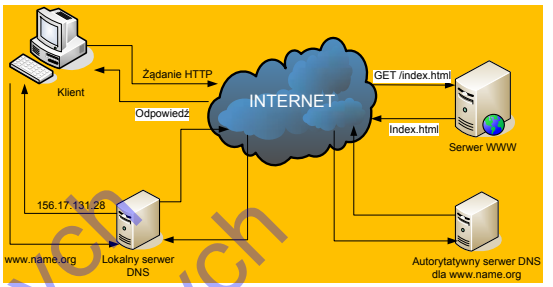
- **aplikacja: umożliwiająca sieciowe zastosowania**
  - **FTP, SMTP, HTTP**
- **transport: transfer danych od hosta do hosta**
  - **TCP, UDP**
- **sieć: routing datagramów od źródła do przeznaczenia**
  - **IP, protokoły routingu**
- **łącze: przepływ danych pomiędzy sąsiednimi elementami sieci**
  - **PPP, Ethernet**
- **fizyczne połączenie: bity "po drucie ew.,po światłowodzie, eterze"**



Odkrywanie zasobów internetowych

- WWW: strony internetowe (**W**orld **W**ide **W**eb)
- Email: poczta elektroniczna
- Telnet: zdalny dostęp do hosta
- FTP: transfer plików (**F**ile **T**ransfer **P**rotocol)
- News/Usenet: grupy dyskusyjne
- Chat/IRC: pogawędki
- Gopher: tekstowe wyszukiwanie
- WAIS (**W**ide **A**rea **I**nformation **S**ervice): usługa informacyjna serwerów WAIS

Usługa WWW



Internet w statystykach

Informacja jest podawana na bieżąco

- ***Internet***
  Internet description from Wikipedia, history, creation, growth, structure, uses and other basic data.
- ***Internet Traffic Report***
  The Internet Traffic Report monitors the flow of data around the world. It then displays a value between zero and 100. Higher values indicate faster and more reliable connections.
- ***The CAIDA Web Site***
  CAIDA, the Cooperative Association for Internet Data Analysis, provides tools and analyses promoting the engineering and maintenance of a robust, scalable global Internet infrastructure.
- ***Internet News***
  Internet dot com provides enterprise IT and Internet Industry professionals with the news, information resources and community they need to succeed in today's rapidly evolving IT and business environment.
- ***Detailed Domain Count***
  Statistics on the number of active domains and those deleted from the Internet each day.
- ***Web Browser Statistics***
  Statistics and trends in browser usage, operating systems.....

Obserwujemy zwrot ku Internetowi mobilnemu!

Internet może utracić neutralność!

Rozwój Internetu

Historia Internetu

1972-1980: Łączenie sieci, nowe oraz prywatne

- 1970: ALOHAnet - satelitarna sieć na Hawajach
- 1973: Metcalfe proponuje w swoim doktoracie Ethernet
- 1974: Cerf and Kahn - architektura do łączenia sieci
- późne lata 70-te: prywatne architektury: DECnet, SNA, XNA
- późne lata 70-te: przełączanie pakietów o ustalonej długości (prekursor ATM)
- 1979: ARPAnet ma 200 węzłów

Zasady Cerf'a oraz Kahn'a dotyczące łaczenia sieci:

- minimalizm, autonomia - żadne wewnętrzne zmiany nie mogą być wymagane przy łączeniu sieci
- routery bez stanów wewnętrznych
- zdecentralizowana kontrola

Historia Internetu

1980-1990: nowe protokoły, szybki i niekontrolowany rozwój sieci

- 1983: rozpowszechnienie TCP/IP
- 1983: protokół SMTP dla e-mail
- 1983: usługa DNS dla tłumaczenia nazw-do-IP-adresów
- 1985: protokół FTP
- 1988: TCP - protokół do kontroli przepływu datagramów
- Powstają nowe narodowe sieci (w USA): Csnet, BITnet, NSFnet, Minitel
- 100,000 hostów podłączonych do skonfederowanych sieci - w porównaniu do 200 pod koniec lat 70-tych

Historia Internetu

Lata 1990-te: komercjalizacja oraz WWW

- Wczesne lata 1990-te: ARPAnet bez opłat
- 1991: NSF nakłada restrykcje na komercyjne używanie NSFnet (anulowane w 1995)
- wczesne lata 1990-te: WWW
  - hypertext [Bush 1945, Nelson 1960's]
  - HTML, http: Berners-Lee
  - 1994: pierwsza przeglądarka Mosaic, później Netscape
  - późne lata 1990-te: komercjalizacja www

Późne lata 1990-te:

- ok. 50 milionów komputerów w Internecie
- ponad 100 milionów (+) użytkowników
- połączenia sieci szkieletowych osiągają przepustowość rzędu 1 Gbps

10 najważniejszych wydarzeń w historii Internetu w Polsce (wg internautów)

1. Podłączenie Polski do Internetu (wrzesień 1991 r.)
2. Pierwsza wymiana poczty elektronicznej pomiędzy Warszawą a Kopenhagą (17 sierpnia 1991 r.)
3. Uruchomienie przez TP SA bezabonamentowego dostępu do Internetu przez numer 0202122 (kwiecień 1996 r.) - pierwsza tego rodzaju usługa w Europie!
4. Powstanie NASK (Naukowa i Akademicka Sieć Komputerowa - 1993 r.)
5. Udostępnienie darmowych kont e-mail (Polbox - grudzień 1996 r.)
6. Stworzenie ogólnopolskiej szkieletowej pakietowej sieci X.25 TP SA pod nazwą Polpak (1992) i Polpak-T (1995 r.)
7. Powstanie Międzyuczelnianej Sieci Komputerowej, obejmującej Gliwice, Warszawę i Wrocław (1981 - 1983 r.)
8. "Internet dla szkół" - program podłączania szkół do Internetu (od 1994 r.)
9. Powstanie portalu Onet.pl (czerwiec 1996 r.)
10. Wprowadzenie SDI do oferty TP SA (listopad 1999 r.)

Źródło:PCWK

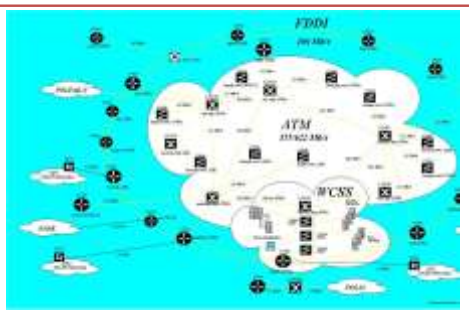
10 lat Internetu w Polsce

- "*Jak widać, opłaciło się finansować polską naukę i projekt fizyków, i astronomów sprzed 10 lat*" - stwierdził prof. Maciej Kozłowski, współtwórca polskiego Internetu, obecnie dyrektor NASK, komentując wyniki ankiety PCWK.
- "*Nie ukrywam jednak, że tak szybki rozwój sieci nas zaskoczył. W 1991 r. nie myśleliśmy, że za 10 lat Internet stanie się tak powszechny, że korzystać będą z niego nie tylko naukowcy, ale także firmy, a nawet dzieci*" - dodaje prof. M. Kozłowski w 10. rocznicę Internetu w Polsce.

Źródło:PCWK

Internet dla nauki w Polsce

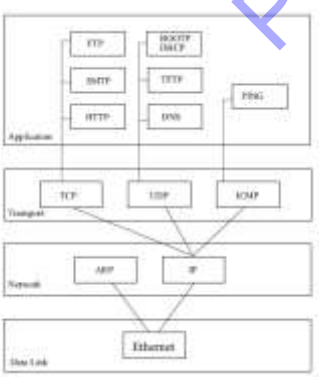
Sieć WASK we Wrocławiu – tak zaczynaliśmy



Sieć WASK we Wrocławiu – teraźniejszość

PIONIER – stan w roku 2011

TCP/IP



Protokoły rodziny TCP/IP

TCP - *Transmission Control Protocol* – protokół kontroli transmisji

- Protokół komunikacyjny między dwoma komputerami.
- Został stworzony przez Vintona Cerfa i Roberta Kahna.
- Jest on częścią większej całości określanej jako stos TCP/IP.
- W modelu OSI TCP odpowiada warstwie transportowej.

Połączenie/Gniazdo TCP – nawiązanie połączenia

- Połączenie TCP jest tworzone przez potrójne potwierdzenie (handshaking) pomiędzy klientem a serwerem.
- Uproszczona procedura postępowania jest następująca:
  - Host inicjujący połączenie wysyła pakiet zawierający segment TCP z ustawioną flagą SYN (*synchronize*).
  - Host odbierający połączenie, jeśli zechce je obsłużyć, odsyła pakiet z ustawionymi flagami SYN i ACK (*acknowledge* – potwierdzenie).
  - Inicjujący host powinien teraz wysłać pierwszą porcję danych, ustawiając już tylko flagę ACK (i gasząc SYN).
  - Jeśli host odbierający połączenie nie chce lub nie może odebrać połączenia, powinien odpowiedzieć pakietem z ustawioną flagą RST (*reset*).
-

Połączenie/Gniazdo TCP – nawiązanie połączenia

- Połączenie jest wtedy utworzone i jest identyfikowane przez „czwórkę" znaną jako tzw. *gniazdo* albo *gniazdko:*

  (adres IP przeznaczenia, numer portu przeznaczenia)

  (adres IP źródła, numer portu źródłowego)
- Podczas fazy ustanawiania połączenia, te wartości są wprowadzone do tabeli i zapisane na czas trwania połączenia.

Transmisja danych

- W celu weryfikacji wysyłki i odbioru TCP wykorzystuje sumy kontrolne i numery sekwencyjne pakietów.
- Odbiorca potwierdza otrzymanie pakietów o określonych numerach sekwencyjnych ustawiając flagę ACK.
- Brakujące pakiety są retransmitowane. Host odbierający pakiety TCP defragmentuje je i porządkuje według numerów sekwencyjnych tak, by przekazać wyższym warstwom pełen złożony segment.

Zakończenie połączenia

- Prawidłowe zakończenie połączenia może być zainicjowane przez dowolną stronę.
- Polega ono na wysłaniu pakietu z ustawioną flagą FIN (*finished*).
- Pakiet taki wymaga potwierdzenia flagą ACK. Najczęściej po otrzymaniu pakietu z flagą FIN, druga strona również kończy komunikację wysyłając pakiet z flagami FIN i ACK.
- Pakiet taki również wymaga potwierdzenia przez przesłanie ACK.
- Dopuszcza się również awaryjne przerwanie połączenia poprzez przesłanie pakietu z flagą RST (*reset*). Pakiet taki nie wymaga potwierdzenia.

Nagłówek TCP

Port nadawcy – 16-bitowy numer identyfikujący port nadawcy.
Port odbiorcy – 16-bitowy numer identyfikujący port odbiorcy.
Numer sekwencyjny – 32-bitowy identyfikator określający miejsce pakietu danych w pliku przed fragmentacją (dzięki niemu, można "poskładać" plik z poszczególnych pakietów).
Numer potwierdzenia – 32-bitowy numer będący potwierdzeniem otrzymania pakietu przez odbiorcę, co pozwala na synchronizację nadawanie-potwierdzenie.
Długość nagłówka – 4-bitowa liczba, która oznacza liczbę 32-bitowych wierszy nagłówka, co jest niezbędne przy określaniu miejsca rozpoczęcia danych.
Dlatego też nagłówek może mieć tylko taką długość, która jest wielokrotnością 32 bitów.
Zarezerwowane – 4-bitowy ciąg zer, zarezerw. dla ewentualnego przyszłego użytku.

Flagi 8-bitowa informacja/polecenie dotyczące bieżącego pakietu:
CWR – (ang. Congestion Window Reduced) flaga potwierdzająca odebranie powiadomienia przez nadawcę, umożliwia odbiorcy zaprzestanie wysyłania echa.
ECE – (ang. ECN-Echo) flaga ustawiana przez odbiorcę w momencie otrzymania pakietu z ustawioną flagą CE, **URG** – informuje o istotności pola "Priorytet„,
ACK – informuje o istotności pola "Numer potwierdzenia"
PSH – wymusza przesłanie pakietu, **RST** – resetuje połączenie (wymagane ponowne uzgodnienie sekwencji), **SYN** – synchronizuje kolejne numery sekwencyjne, **FIN** – oznac zakończenie przekazu danych
Szerokość okna – 16-bitowa informacja o tym, ile danych może aktualnie przyjąć odbiorca. Wartość 0 wskazuje na oczekiwanie na segment z innym numerem tego pola. Jest to mechanizm zabezpieczający komputer nadawcy przed zbyt dużym napływem danych.
Suma kontrolna – 16-bitowa liczba, będąca wynikiem działań na bitach całego pakietu, pozwalająca na sprawdzenie tego pakietu pod względem poprawności danych.
Wskaźnik priorytetu – jeżeli flaga URG jest włączona, informuje o ważności pakietu.
Opcje – czyli ewentualne dodatkowe informacje i polecenia: **0** – koniec listy opcji, **1** – brak działania, **2** – ustawia maksymalna długość segmentu. W przypadku opcji **2** to tzw. **Uzupełnienie**, które dopełnia zerami długość segmentu do wielokrotności 32 bitów (patrz: informacja o polu "Długość nagłówka")

Dobrze znane porty

Numer portu	Protokół/Aplikacja
20 (dane) 21 (sterowanie)	FTP (File Transfer Protocol)
22	SSH (Secure Shell Login)
23	TELNET (Network Terminal Protocol)
25	SMTP (Simple Mail Transfer Protocol)
53	DNS (Domain Name System)
69	TFTP (Trivial File Transfer Protocol)
79	FINGER
80	HTTP (Hyper Text Transfer Protocol)
109	POP (Post Office Protocol)
119	NNTP (Network News Transfer Protocol)
143	IMAP (Internet Message Access Protocol)
161	SNMP (Simple Network Management Protocol)
...	...

HTTP

Specyfikacja protokołu

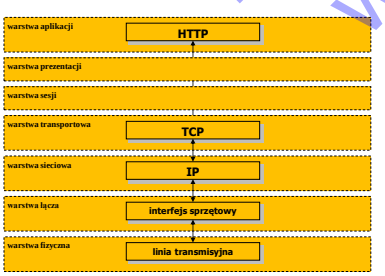
Protokół HTTP - wstęp

- Protokół **HTTP** (***HyperText Transfer Protocol***) jest podstawowym protokołem usługi WWW. Położony na 7. poziomie aplikacji w modelu ISO/OSI i wykorzystujący domyślnie 4. warstwę transportową TCP (zapewniającą pewny przesył informacji), umożliwia przesłanie zasobów w układzie klient-serwer.
- Mianem „zasobów" definiuje się każdego rodzaju dane dostępne poprzez dokumenty WWW i pobierane zgodnie z ideą hipertekstu: będą to przede wszystkim pliki opisu stron HTML, obrazy (GIF, JPG, PNG), wynik działania aplikacji po stronie serwera, skrypty uruchamiane po stronie klienta (Java Applet, JavaScript, ActiveX), definicje styli (CSS) itd.

Protokół HTTP - wstęp

- Część danych dostępnych poprzez przeglądarki WWW używa innych protokołów, opartych często na odmiennych warstwach transportowych (np. UDP) – dotyczy to głównie strumieni audio i wideo (Real Networks, MS NetShow).
- Pierwotna wersja protokołu - **HTTP/0.9** – przesyłała komunikaty pozbawione nagłówków (*raw data*) i nie zawierała żadnych rozszerzeń funkcjonalnych. Ewolucję protokołu od wersji **1.0** do **1.1** wraz z ogólnymi założeniami przyszłościowego rozwiązania (**HTTP-NG**) przedstawiono w dalszej części wykładu.

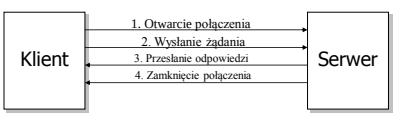
Protokół HTTP - wstęp



Położenie HTTP w skali modelu ISO/OSI

Przebieg transakcji HTTP

- Protokół HTTP jest bezstanowym protokołem sieci WWW.
- Zasada wymiany informacji jest bardzo prosta i charakterystyczna dla systemów klient-serwer:
  - przeglądarka otwiera połączenie i zgłasza do oczekującego na żądania serwera zapotrzebowanie na określony zasób;
  - serwer w odpowiedzi przesyła poszukiwane dane, ewentualnie zwraca status błędu, po czym zamyka połączenie.
  - przebieg transakcji jest 4-etapowy:



Przebieg transakcji HTTP

- **Etap 1 –** klient otwiera połączenie TCP/IP; domyślny port docelowy to port 80.
- **Etap 2 –** klient wysyła żądanie HTTP o zasób wskazany lokalizatorem URL w 1. linii nagłówka, przy użyciu wybranej metody i wersji HTTP.
- **Etap 3 -** serwera przesyła odpowiedź HTTP: może to być nagłówek i treść żądanego zasobu, bądź też komunikat o błędzie (brak zasobu, błąd autoryzacji, serwer przeciążony itd.).
- **Etap 4** – serwer zamyka połączenie TCP/IP; obie komunikujące się aplikacje muszą odpowiednio reagować na niespodziewane zamknięcie połączenia.

Założenia protokołu HTTP/1.0

- Definicja protokołu HTTP w wersji 1.0 została zapisana w dokumencie **RFC1945**. Dokument ten przedstawia cechy, które muszą być zaimplementowane po stronie przeglądarki i serwera, aby transakcje HTTP/1.0 mogły zostać zainicjowane i bezbłędnie przeprowadzone.
- Celem było stworzenia szybkiego i łatwego w implementacji protokołu, przeznaczonego do hipermedialnych systemów rozproszonych.
- Dalej: podstawowe mechanizmy protokołu, postać żądania i odpowiedzi wraz ze statusem, najważniejsze pola nagłówków, przebieg podstawowego uwierzytelniania.

Komunikaty HTTP

- Całość informacji wymienianych jednorazowo pomiędzy klientem a serwerem lub odwrotnie określa się mianem komunikatu HTTP (*HTTP message*) – w zależności od kierunku transmisji jest to komunikat żądania (kierunek: **klient→serwer**) lub odpowiedzi (kierunek: **serwer→klient**).
- Każdy komunikat złożony jest z nagłówka i treści.

Komunikaty HTTP

- W przypadku **komunikatu żądania** w pierwszej linii podawana jest metoda HTTP wraz ze wskazaniem zasobu oraz wersją protokołu, którego używa przeglądarka – wszystkie pola mogą być oddzielone wielokrotnymi spacjami lub tabulacjami.
- Kolejne linie nagłówka żądania zawierają wymagane lub nadobowiązkowe informacje, związane z wystawionym zapytaniem - przekazywane są w formie tzw. pól nagłówka w formacie typowym dla MIME (*Multipurpose Internet Mail Extensions*).
- Po przekazaniu nagłówka wymagane jest wstawienie jednej pustej linii – specyfikacja dopuszcza podwójne użycie znaków CRLF lub tylko LF. Przestrzeń po pustej linii jest przeznaczona na treść żądania – jest ona przesyłana w przypadku niektórych metod HTTP, np. POST.

Komunikaty HTTP

- Struktura nagłówka **komunikatu odpowiedzi** jest analogiczna – różnice dotyczą pierwszej linii oraz zestawu wymaganych i nadobowiązkowych pól nagłówka.
- Pierwsza linia to tzw. linia statusu: zawiera oznaczenie wersji HTTP oraz liczbowy kod (status) odpowiedzi wraz z reprezentacją łańcuchową.
- Kolejne linie to pola nagłówka zakończone linią pustą, po której może być przesłana treść odpowiedzi zgodnie z zadeklarowanym w nagłówku typem MIME – jej zawartość może być tekstowa (np. dokument HTML) lub binarna (np. grafika GIF).

Metody HTTP

- Metody HTTP określają sposób wywołania danego zasobu i reakcji serwera.
- Dokument RFC1945 definiuje **trzy podstawowe metody** wraz ze wskazaniem na możliwość użycia metod rozszerzonych – zgodnie z ogólnym założeniem architektury protokołu opisującym jego otwartość.

Metody HTTP

- **Podstawową i najpopularniejszą metodą jest GET**.
- Serwer po otrzymaniu tego polecenia wyszuka/uruchomi i zwróci zasób do klienta; w razie problemów odpowie właściwym kodem statusu.
- Odmianą tej metody jest tzw. ***Conditional GET***, który umożliwia najprostszy *cacheing* (wykorzystanie pamięci podręcznej) dokumentów po stronie przeglądarki: serwer zwróci zasób tylko wtedy, gdy data jego aktualizacji jest nowsza od tej sygnalizowanej przez klienta w polu If-Modified-Since.

Metody HTTP

- **Metoda HEAD** jest „podmetodą" GET – serwer wykonuje w tym przypadku identyczne czynności za wyjątkiem przesłania treści odpowiedzi; klient otrzymuje jedynie nagłówek. W praktyce metodę tą używa się do rozpoznania obiektu (data utworzenia, rozmiar, typ) bez pobierania jego zawartości.

Metody HTTP

- Definicja **metody POST** w RFC1945 jest najbardziej ogólna – oznacza przesłanie pewnych informacji od klienta (w przypadku tej metody przeglądarka ma prawo dołączyć treść komunikatu) oraz ich odczyt i interpretację przez serwer.
- Zalecenie określa, że może to być komentarz do istniejących zasobów, przesłanie komunikatu do grupy dyskusyjnej lub metoda interakcji z bazą danych. W praktyce metoda POST wykorzystywana jest do przesyłania danych z formularza HTML do aplikacji po stronie serwera.

Metody HTTP

- **Dodatkowe metody** to: PUT, DELETE, LINK, UNLINK - rzadziej implementowane w serwerach, gdyż nie są wymagane przez specyfikację, ponadto zaleca się ich unikanie ze względu na wymogi bezpieczeństwa (każda z tych metod ingeruje w zawartość zasobów).

Identyfikator zasobu

- Identyfikator zasobu (URI – *Uniform Resource Identifier*) pozwala na jasne i precyzyjne określenie jego położenia w sieci Internet. URI, określany potocznie jako adres WWW, jest kombinacją definicji lokalizacji URL (*Uniform Resource Localizator*) i nazwy URN (*Uniform Resource Name*).
- Dla celów usługi WWW używa się jego uszczegółowionej postaci: HTTP URL, złożonej z następujących znaczników:

  http_URL = "http:" "//" host [":" port] [abs_path]

  gdzie:

Identyfikator zasobu

- **host** – określa prawidłową nazwę domeny (www.serwer.pl) lub adres IP w postaci czteroczęściowej numerycznej (195.116.34.2).
- **port** – określa port HTTP serwera; jest parametrem opcjonalnym, w przypadku pominięcia zakłada się domyślny port 80.
- **abs_path** – absolutna ścieżka do zasobu w obrębie danego serwera w odniesieniu do jego katalogu źródłowego; jej brak skutkuje zwrotem zasobu domyślnego, określonego w konfiguracji serwera.
- Wskazanie zasobu może mieć postać:

  http://www.serwer.pl/zasob.html
- Zalecenie RFC1945 wymaga umieszczenia w 1. linii komunikatu żądania wskazania zasobu, ale tylko ostatniego elementu z powyższej definicji: abs_path (dla zasobu domyślnego przeglądarka zobowiązana jest podać znak „/").
- Pełny zapis HTTP URL jest dostarczany tylko do serwerów HTTP Proxy, które przed przesłaniem do serwera docelowego dokonują konwersji do postaci abs_path.
- W przypadku pobrania zasobów nie korzystających z warstwy transportowej TCP (strumienie audio/wideo), wykorzystuje się indywidualne metody adresowania zasobów.

Pola nagłówka

- Integralną częścią komunikatów HTTP są pola nagłówka. Umieszczone po pierwszej linii nagłówka w nieograniczonej przez specyfikację liczbie, przenoszą dodatkowe, obowiązkowe lub nie, informacje dotyczące przesyłanego komunikatu.
- Pola są podzielone na **cztery** kategorie: ogólne, żądania, odpowiedzi i zawartości. Komunikat żądania może zawierać pola ze wszystkich grup za wyjątkiem pól odpowiedzi; analogicznie komunikat odpowiedzi nie może zawierać tylko pól żądania.

Pola nagłówka – pola ogólne

- Ogólne pola nagłówka dotyczą zarówno komunikatów przesyłanych przez klienta jak i przez serwer:

  Date – data wysłania komunikatu;

  Pragma – pozwala na dołączenia specyficznych dla implementacji rozkazów; najpopularniejszy to dyrektywa no-cache, zakazująca umieszczania dokumentu w pamięci podręcznej;

Pola nagłówka – pola żądania

- Pola nagłówka żądania przekazują dodatkowe informacje bądź o zapytaniu, bądź o samym kliencie:
- **Authorization –** przy użyciu tego pola klient określa typ oraz podaje niezbędne do uwierzytelnienia dostępu dane; w przypadku protokołu HTTP/1.0 jest to zawsze uwierzytelnienie podstawowe *Basic Authentication*;
- **From –** adres e-mail osoby korzystającej z przeglądarki;
- **If-Modified-Since** – pole zawiera datę wykorzystywaną w metodzie *Conditional GET* do walidacji zawartości dokumentu;
- **Referer –** pole przekazuje lokalizację odnośnika, który wskazał żądany zasób;
- **User-Agent** – nazwa i wersja oprogramowania przeglądarki;

Pola nagłówka – pola odpowiedzi

- Pola nagłówka odpowiedzi przekazują dodatkowe informacje o odpowiedzi, których nie zawarto w statusie:
- **Location** – serwer podaje w tym polu przekierowanie na inny zasób HTTP URL; pole to łączy się zazwyczaj z odpowiedzią o kodzie statusu 3xx;
- **Server –** nazwa i wersja oprogramowania serwera;
- **WWW-Authenticate –** bierze udział w procesie uwierzytelnienia dostępu.

Pola nagłówka – pola opisujące zawartość

- Pola opisujące zawartość dotyczą cech szczególnych treści przesyłanego komunikatu:
- **Allow –** przekazuje spis dopuszczalnych metod HTTP powiązanych z wybranych zasobem;
- **Content-Encoding, Content-Length, Content-Type –** pola te przekazują informację o sposobie zakodowania treści komunikatu (w celu polepszenia wydajności WWW czasem zasoby są dodatkowo kompresowane przez serwer, o długości treści (brak tej informacji nie musi wpływać na błąd w transmisji – serwery WWW zazwyczaj pobierają zasób aż do zamknięcia połączenia; problem z wyliczeniem długości zasobu pojawia się często w przypadku stron generowanych dynamicznie) oraz o typie zasobu, do którego określenia stosuję się notację MIME;
- **Expires –** data przeterminowania ważności zasobu;
- **Last-Modified –** data ostatniej aktualizacji zasobu.

Status odpowiedzi

- Dwa ostatnie elementy linii statusu w nagłówku odpowiedzi to 3-cyfrowy kod oraz zapis przyczyny. Pozwalają one przekazać serwerowi informację o przebiegu wyszukania zasobu. Kod powoduje właściwą reakcję przeglądarki, natomiast tekstowa reprezentacja przyczyny statusu, często konfigurowalna przez administratora serwera, może, lecz nie musi być wyświetlona w oknie oprogramowania klienta.

Status odpowiedzi

- Kody statusu zostały zakwalifikowane do pięciu kategorii:
  - **1xx –** kody informacyjne; w wersji HTTP/1.0 zarezerwowane, wykorzystane zostały dopiero w wersji 1.1;
  - **2xx –** operacja zakończyła się sukcesem: żądanie zostało przyjęte, zrozumiane i wykonane; zasób został przesłany (**200**), utworzony (**201**), jego wykonywanie rozpoczęło się (**202**) lub nie ma potrzeby przesłania informacji zwrotnej (**204**);
  - **3xx –** przekierowanie: przeglądarka musi ponowić żądanie pod inny, wskazany przez serwer adres, przy czym serwer może wskazać kilka możliwości (**300**), może też poinformować, że zasób został przeniesiony na stałe (**301**) lub czasowo (**302**); kod **304** określa, że dokument nie był modyfikowany (odpowiedź na metodę *Conditional Get*;
  - **4xx –** błąd po stronie klienta: przesłane żądanie zawierało błędy (**400**), dostęp do zasobu wymaga autoryzacji (401) lub jest zabroniony (**403**); kod 404 informuje o braku żądanego zasobu;
  - **5xx –** błąd po stronie serwera: wystąpił błąd wewnętrzny (**500**), metoda żądania nie została zaimplementowana (**501**), błąd wystąpił po stronie serwera proxy (**502**) lub też serwer docelowy jest chwilowo przeciążony lub konserwowany (**503**).

Uwierzytelnienie dostępu

- Protokół HTTP/1.0 wprowadził mechanizm uwierzytelnienia dostępu do zasobów, zwany *Basic Authentication*. Jego idea opiera się na zdefiniowaniu przez administratora obszarów (*realms*) plików i katalogów ograniczonego dostępu, do których odnoszą się listy uprawnionych użytkowników wraz z hasłami; czasem dodatkowemu ograniczeniu/zezwoleniu podlegają komputery pochodzące ze wyspecyfikowanych grup adresów internetowych.
- Podstawowe uwierzytelnienie dostępu zakłada, że połączenie klient-serwer jest zabezpieczone przed podsłuchem – nazwa użytkownika i jego hasło są przesyłane w sposób jawny (ściślej rzecz biorąc, dane te podlegają prostemu kodowaniu Base64, które konwertuje znaki 8-bitowe w pakiety 6-bitowe).
- W przypadku sieci Internet przekonanie to jest oczywiście mylne i o ile uwierzytelnienie tego rodzaju jest wystarczające w większości przypadków, o tyle nie zapewnia wystarczającego poziomu bezpieczeństwa tam, gdzie to jest naprawdę ważne (operacje bankowe oraz w handlu elektronicznym) – w takich sytuacjach należy stosować uwierzytelnienie rozszerzone protokołu HTTP/1.1, protokół HTTPS, zabezpieczenia poza protokołem HTTP.

Specyfikacje MIME

- MIME (*Multipurpose Internet Mail Extensions*) jest zbiorem specyfikacji, wykorzystywanych nie tylko w usłudze WWW, pozwalającym na identyfikację typu przesyłanego zasobu, zastosowanego zestawu znaków, czasem również na kodowanie znaków 8 i więcej bitowych do postaci 7-bitowej. Określenie typu dokumentu (pole Content-Type nagłówka) pozwala na prawidłową interpretację treści komunikatu przez oprogramowanie klienta czy serwera.
- Rodzaj dokumentu jest opisywany przez typ i podtyp oraz opcjonalnie zestaw dodatkowych parametrów. Podstawowe typy zasobów to: text, image, audio, video, application, message i multipart; ponadto możliwe jest definiowanie typów rozszerzonych. W obrębie każdego typu określono zestaw podtypów, przykładowo: dokument tekstowy – text/plain, dane z formularza HTML – application/x-www-form-urlencoded, obraz GIF – image/gif itd. Parametry, dodawane po średniku, opisują dodatkowe cechy dokumentu, np. zestaw użytych znaków.
- Pole w nagłówku odpowiedzi, której treść stanowi dokument HTML w języku polskim (zestaw znaków Latin-2) powinno mieć postać:

  Content-Type: text/html; charset=iso-8859-2
- Serwer zapisuje typ dokumentu na podstawie danych konfiguracyjnych i rozszerzeń plików, rzadziej na podstawie ich wewnętrznej struktury.

Metody kontroli stanu połączenia

- Protokół HTTP jest bezstanowy, tzn. nie utrzymuje informacji o połączeniu pomiędzy kolejnymi żądaniami – serwer odpowiada na każde z nich bez odniesienia do poprzednich zapytań.
- Pozwoliło to znacznie uprościć konstrukcję serwerów oraz zwiększyć ich wydajność (serwer nie jest obciążony śledzeniem i przekazywaniem parametrów sesji).
- Niestety, wprowadziło też znaczne ograniczenia, a najlepszym tego przykładem są transakcje w handlu elektronicznym, które wymagają dostarczania na bieżąco informacji o kliencie.

Metody kontroli stanu połączenia

- Potrzeba wprowadzenia pojęcia sesji do protokołu HTTP zaowocowała oddzielnym dokumentem RFC2109 oraz zaleceniem grupy IETF (*Internet Engineering Task Force*), które opisuje metody zarządzania stanem, oparte na wczesnych rozwiązaniach firmy Netscape – tzw. *cookies*, przyjętych później przez innych producentów oprogramowania przeglądarek WWW.
- Idea polega na dodaniu dwóch definicji pól nagłówka komunikatu. Serwer przekazując klientowi zestaw informacji wymaganych przy późniejszych połączeniach, wpisuje w nagłówku odpowiedzi pole Set-Cookie wraz z listą parametrów; możliwe jest również przekazanie kilku tego rodzaju pól.
- Aż do zakończenia sesji klient jest zobowiązany odpowiadać tym samym zestawem informacji przekazując pole/pola Cookie w nagłówku żądania - naturalnie, w czasie trwania sesji serwer może modyfikować dowolne wartości atrybutów. Wśród opcjonalnych parametrów warto wymienić Max-Age, który determinuje czas przechowywania informacji (czas życia *cookie*) – serwer kończąc sesję ustawia Max-Age=0, zwalniając klienta z obowiązku odsyłania wcześniej przekazanych informacji.

Metody kontroli stanu połączenia

- Implementacje *cookies* zawsze pełne były różnego rodzaju niedociągnięć i dziur w bezpieczeństwie (np. łatwy dostęp do numerów kart kredytowych na dysku lokalnym klienta) – stąd panuje powszechna niechęć do tego sposobu utrzymywania stanu połączenia, a duża część użytkowników zablokowała ten mechanizm w swoich przeglądarkach.
- Istnieją też bardziej wyszukane metody utrzymywanie informacji o kliencie poprzez wiele połączeń: niektóre bazują na jego adresie i porcie internetowym, w przypadku stron z formularzami generowanymi dynamicznie stosuje się tzw. pola ukryte.

Przykłady wymiany komunikatów żądania i odpowiedzi – różne wersje HTTP

Pobranie zasobu przy użyciu protokołu HTTP/0.9:

Klient:
GET /index.html

Serwer:
<tresc dokumentu>

Pobranie statycznego zasobu przy użyciu protokołu HTTP/1.0:

Klient:
GET /index.html HTTP/1.0
From: praca@dyplomowa
User-Agent: Przegladarka/0.1
Referer: http://www.serwer.pl/poprzednia.html

Serwer:
HTTP/1.0 200 OK
Content-Type: text/html
Content-Length: 1360
Last-Modified: Fri, 31 Dec 1998 23:59:59 GMT
Server: HWS/0.9
<CRLF>
<tresc dokumentu HTML>

Przykłady wymiany komunikatów żądania i odpowiedzi – różne wersje HTTP

Walidacja ważności dokumentu w pamięci *cache* przeglądarki (metoda *Conditional GET*):

Klient:
GET /index.html HTTP/1.0
If-Modified-Since: Fri, 10 Jan 1999 22:14:32 GMT
User-Agent: Przegladarka/0.1

Serwer:
HTTP/1.0 304 Not Modified
Content-Type: text/html
Content-Length: 1360
Last-Modified: Fri, 31 Dec 1998 23:59:59 GMT
Server: HWS/0.9

Przesłanie danych z formularza HTML za pomocą metody POST:

Klient:
POST /cgi-bin/skrypt.pl HTTP/1.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 28
<CRLF>
imie=Jan&drugie+imie=Zbyszek

Serwer:
HTTP/1.0 200 OK
Content-Type: text/plain
<CRLF>
<wydruk tekstowy ASCII ze skryptu CGI>

Próba użycia niezaimplementowanej w serwerze metody (HTTP/1.1):

Klient:
DELETE /kosz.html HTTP/1.1

Serwer:
HTTP/1.1 501 Not Implemented

Zmiany i rozszerzenia protokołu HTTP/1.1

- Wykorzystanie rozszerzeń protokołu
- Użycie pamięci podręcznej cache
- Optymalizacja wykorzystania pasma
- Zarządzanie połączeniem
- Przekaz komunikatów
- Przechowywanie adresu
- Informacje o błędach
- Bezpieczeństwo
- Negocjacja zawartości

RFC 2616 (June 1999)

Wykorzystanie rozszerzeń protokołu

- Jednym z podstawowych założeń protokołu HTTP było właściwe rozpoznanie bieżących cech klienta/serwera oraz kompatybilność z wcześniejszymi wersjami – serwer i klient identyfikują nawzajem swoje możliwości na bazie zapisanego w nagłówku numeru protokołu. W wersji 1.0 pojawił się jednak istotny problem: numer protokołu zawsze odnosił się do ostatniego nadawcy (*hop-by-hop sender*) np. serwera proxy, a nie do nadawcy źródłowego (*end-by-end sender*). Tym samym odbiorca nie mógł rozpoznać wersji protokołu właściwego nadawcy.
- W protokole HTTP/1.1 wprowadzono zatem pole Via, które, modyfikowane przez każdy węzeł w łańcuchu połączenia, opisuje drogę przebytą przez komunikat.
- W protokole HTTP/1.1 wprowadzono ponadto nową metodę – OPTIONS – pozwalającą na rozpoznanie przez klienta cech i wymagań związanych z wywoływaniem zasobu bez przesłania jego treści – przeglądarka może przykładowo dowiedzieć się, jakim protokołem posługuje się serwer oraz czy dostęp do zasobów jest autoryzowany.
- Klient może zasygnalizować serwerowi chęć użycia nowszego lub niestandardowego protokołu poprzez pole Upgrade. Mechanizm ten ma pozwolić na łatwiejsze wprowadzanie do użycia aktualnych wersji protokołu.

Użycie pamięci podręcznej cache

- Użycie pamięci *cache* ma niebagatelny wpływ na zwiększenie wydajności przesyłania dokumentów WWW; stąd zmiany w wersji 1.1.
- Podstawowe mechanizmy wykorzystania pamięci *cache* pojawiły się już w protokole HTTP/1.0 – metoda *Conditional GET* wraz z polem If-Modified-Since pozwalały na sprawdzenie ważności zawartości pamięci podręcznej przeglądarki; jeśli data utworzenia/aktualizacji dokumentu jest po stronie serwera nowsza od tej posiadanej przez klienta, to przeglądarka uzupełnia zawartość. Pola Pragma: no-cache zapobiegało zapamiętywaniu dokumentu w pamięci klienta.
- W HTTP/1.1 zrezygnowano przede wszystkim z oznaczania chwili aktualizacji zasobu poprzez łańcuchowy zapis daty – problemem była synchronizacja zegarów klienta i serwera oraz niewystarczająca rozdzielczość (1 sek.). Zamiast tego w wersji 1.1 do bieżącej postaci każdego zasobu przypisane jest unikalne pole Etag, którego zawartość pozwala przeglądarce jednoznacznie stwierdzić czy ważność umieszczonego w jej *cache'u* dokumentu została przeterminowana. Ponadto powstały pochodne pola If-Modified-Since: If-Unmodified-Since oraz If-Match.
- Rozbudowane sterowanie *cache'em* umożliwia pole Cache-Control; określa ono co może być umieszczone w pamięci podręcznej i na jak długo, można zakazać/nakazać aktualizację zawartości *cache.* Do najważniejszych dyrektyw tego pola należą: no-cache – zakaz przechowywania, max-age – czas przeterminowania, private – zakaz użycia publicznej pamięci *cache.*

Optymalizacja wykorzystania pasma

- Optymalizacja wykorzystania protokołu była bardzo istotnym czynnikiem motywującym zmiany w wersji 1.1. Wykorzystanie pól Range i Content-Range dało możliwość przesłania zasobu w częściach, niwelując tym samym skutki przerwanych połączeń, dając możliwość pobrania początku treści zasobu (np. paletę barw i rozmiar grafiki) czy też doczytywania końcówki rosnącego obiektu (np. odczyt plików logowania i statystyk).
- Ponadto wprowadzono nowy kod statusu – 100 (*Continue*). Zapobiega on zbędnemu przesłaniu treści komunikatu przez klienta (POST, PUT) w wypadku, gdy np. dostęp do zasobów jest autoryzowany.
- W protokole HTTP/1.1 umożliwiono również kompresję treści komunikatu; o ile pliki graficzne i dźwiękowe zazwyczaj są już w postaci spakowanej, to dokumenty tekstowe mogą na kompresji wiele zyskać.

Zarządzanie połączeniem

- Występują wady wykorzystania warstwy transportowej TCP w procesach przesłania dokumentów WWW (krótkie połączenia, małe rozmiary zasobów). Wśród metod poprawienia wydajności wymieniono przede wszystkim możliwość utworzenia **trwałego połączenia** dla przesyłki wielu obiektów oraz ich **strumieniowanie**.
- Projektanci HTTP/1.1 uwzględnili te zalecenia. **Trwałe połączenia** (*persistent connections*) są zachowaniem domyślnym w tej wersji protokołu – połączenie trwa aż do wystawienia dyrektywy Connection: close przez klienta lub do chwili upływu okresu przeterminowania.
- **Strumieniowanie** pozwala przeglądarce wystawić wiele nowych żądań nawet jeśli nie nadeszło potwierdzenie dla poprzednich.

Przekaz komunikatów

- Sporym problemem w wersji 1.0 było określenie rozmiaru (pole Content-Length) zasobów generowanych dynamicznie; skrypty przed wyemitowaniem treści komunikatu musiałyby ją buforować, co nie zawsze jest możliwe i wprowadza dodatkowe opóźnienia. Wskutek tego klient nie był w stanie określić z góry długości zasobu, a przy przerwaniu połączenia nie było pewności czy obiekt został przesłany w całości.
- W wersji 1.1 serwer może użyć pola Transfer-Encoding: chunked, aby poinformować przeglądarkę, że zasób będzie przesyłany w mniejszych partiach o znanym rozmiarze.

Przechowywanie adresu

- Ważnym rozszerzeniem serwerów WWW było wprowadzenie wirtualnych domen, czyli możliwości umieszczenia wielu serwisów internetowych na jednym komputerze wyposażonym w pojedynczy interfejs sieciowy. Tym samym konieczny dla serwera stał się sposób wyboru właściwego zasobu dla nadchodzącego żądania. Warto przypomnieć, że identyfikator zasobu podaje tylko ścieżkę lokalizacji, natomiast nie przekazuje informacji o nazwie domeny, z której pochodzi (jest ona oczywista w przypadku umieszczenia na serwerze pojedynczego serwisu).
- W wersji 1.1 wprowadzono w nagłówku żądania pole Host, zawierające wspomnianą nazwę domeny – na jej podstawie oraz adresu URL serwer jest w stanie jednoznacznie zdeterminować położenia obiektu we własnym drzewie plików.

Informacje o błędach

- HTTP/1.0 definiuje 16 kodów statusu.
- Protokół HTTP/1.1 zawiera już łącznie 24 definicje, są wśród nich następujące nowe definicje: 100 (klient może kontynuować przesłanie treści komunikatu), 402 (wymóg opłaty), 405 (niedozwolona metoda), 410 (zasób został trwale wyrzucony), 505 (nieznana wersja protokołu).
- Inną nowością jest możliwość umieszczenia przez serwer ostrzeżenia w polu Warning. Protokół wspomina tu o sygnalizacji błędu rewalidacji dokumentu w pamięci *cache*, czasowym jej odłączeniu itp.

Bezpieczeństwo

- Największą wadą wprowadzonego w wersji HTTP/1.0 uwierzytelnienia podstawowego (*Basic Authentication*) było przesyłanie nazwy użytkownika i hasła w postaci niezaszyfrowanej. Słabości tej pozbawiony jest nowy mechanizm zalecany dla wersji 1.1 o nazwie *Digest Authentication*).
- Koncepcja uwierzytelnienia jest bardzo podobna: serwer ogranicza dostęp do pewnych zasobów poprzez zdefiniowanie obszarów (*realms*) oraz przypisanie im dozwolonych użytkowników wraz z hasłami.
- Klient próbujący pobrać zasoby o ograniczonym dostępie otrzymuje odpowiedź 401 i żąda od użytkownika wpisania nazwy i hasła. Przesyłając w drodze powrotnej zaszyfrowane dane użytkownika, podaje dodatkowo sumę kontrolną - domyślnie MD5 (*Message Digest 5*) - obliczoną na podstawie identyfikatora, hasła, metody HTTP, lokalizatora zasobu oraz specjalnej wartości (*nonce*) otrzymanej z serwera wraz z kodem 401. Oprócz szyfrowania danych mechanizm ten zabrania pobrania więcej niż jeden raz tego samego zasobu z użyciem tych samych parametrów, minimalizując ryzyko podsłuchu.

Negocjacja zawartości

- Podstawą wprowadzenia negocjacji zawartości treści komunikatu odpowiedzi był wybór najwłaściwszej reprezentacji zasobu zgodnej z życzeniami klienta – zalecenia przeglądarki mogą przykładowo dotyczyć zastosowanego zestawu znaków i języka dokumentu. Wstępne mechanizmy negocjacji wprowadzono w HTTP/1.0 – wersja 1.1 uporządkowała je i rozszerzyła.
- Rozróżnia się dwa typy negocjacji: klient przekazuje swoje preferencje do serwera (*server-driven*) lub odwrotnie: serwer proponuje klientowi kilka możliwości odnoszących się do żądanego zasobu (*agent-driven*).
- W pierwszym przypadku przeglądarka przekazuje swoje preferencje poprzez pola typu Accept-* (Accept-Language, Accept-Charset itd.), w drugim - serwer przekazuje kilka możliwości do wyboru przy użyciu kodu odpowiedzi 300 (*Multiple Choices*).
- W praktyce drugi wariant nie został jeszcze dopracowany i zazwyczaj stosuje się negocjację typu *server-driven*; przy jej użyciu klient może wybrać najlepszy wariant zasobu – względem języka (użytkownik może nawet zadeklarować znajomość kilku o różnym stopniu zaawansowania), zestawu znaków etc.
- W przyszłości planuje się wprowadzenie możliwości negocjacji zawartości pod względem formy: rozmiaru ekranu, głębi kolorów.

Projekt HTTP-NG

- Wraz z rozwojem HTTP/1.1 narodził się pomysł stworzenia nowego protokołu dla WWW.
- Projekt ten – nazwany protokołem nowej generacji HTTP-NG (*HTTP Next Generation*) – zakładał stworzenie niezależnego rozwiązania pozbawionego błędów wcześniejszych wersji i ograniczeń wprowadzanych przez zasadę kompatybilności wstecznej.
- Zalecenia dotyczące HTTP-NG były próbą nie do końca sprecyzowanych ogólników, zarówno w kwestii celów jakie ma osiągnąć przyszły protokół, jak i metod, którymi cele te zostaną osiągnięte.
- Natomiast prowadzone są aktualnie prace nad zweryfikowaną postacią HTTP/1.1 - **Hypertext Transfer Protocol Bis (httpbis)**

Badania mechanizmów protokołu TCP i HTTP (projekt DominoHTTP)

- nawiązanie połączenia TCP
- zakończenie połączenia TCP
- tradycyjna metoda połączeniowa protokołu HTTP/1.0
- mechanizm *persistent connection* w HTTP
- mechanizm *pipelining* w HTTP/1.1

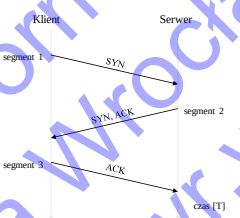
Wybrane zagadnienia

Przebadane mechanizmy protokołu TCP i HTTP:

- nawiązanie połączenia TCP
- zakończenie połączenia TCP
- tradycyjna metoda połączeniowa protokołu HTTP/1.0
- mechanizm *persistent connection* w HTTP
- mechanizm *pipelining* w HTTP/1.1

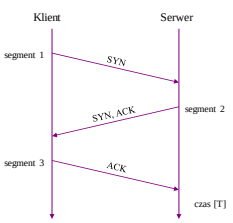
Wybrane zagadnienia protokołu TCP

Nawiązanie połączenia TCP

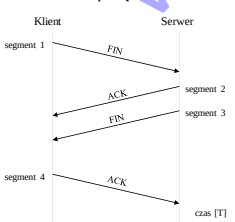


Wybrane zagadnienia protokołu TCP

Nawiązanie połączenia TCP



Zakończenie połączenia TCP



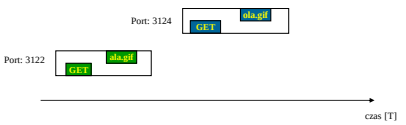
Wybrane zagadnienia

Przebadane mechanizmy protokołu TCP i HTTP:

- nawiązanie połączenia TCP
- zakończenie połączenia TCP
- tradycyjna metoda połączeniowa protokołu HTTP/1.0
- mechanizm *persistent connection* w HTTP
- mechanizm *pipelining* w HTTP/1.1

Metoda tradycyjna protokołu HTTP/1.0

- przesłanie każdego obiektu ze strony WWW w ramach osobnego połączenia TCP



Lp.	Obiekt	Oznaczenie
1	ala.gif	
2	ola.gif	

Wybrane zagadnienia

Przebadane mechanizmy protokołu TCP i HTTP:

- nawiązanie połączenia TCP
- zakończenie połączenia TCP
- tradycyjna metoda połączeniowa protokołu HTTP/1.0
- mechanizm *persistent connection* w HTTP
- mechanizm *pipelining* w HTTP/1.1

Mechanizm *persistent connection*

- przesłanie wielu obiektów ze strony WWW w ramach tego samego połączenia TCP

Lp.	Obiekt	Oznaczenie
1	ala.gif	
2	ola.gif	

Wybrane zagadnienia

Przebadane mechanizmy protokołu TCP i HTTP:

- nawiązanie połączenia TCP
- zakończenie połączenia TCP
- tradycyjna metoda połączeniowa protokołu HTTP/1.0
- mechanizm *persistent connection* w HTTP
- mechanizm *pipelining* w HTTP/1.1

Mechanizm *pipelining*

- przesłanie wielu obiektów ze strony WWW na tym samym połączeniu TCP
- wysłanie wielu żądań bez oczekiwania na odpowiedź serwera

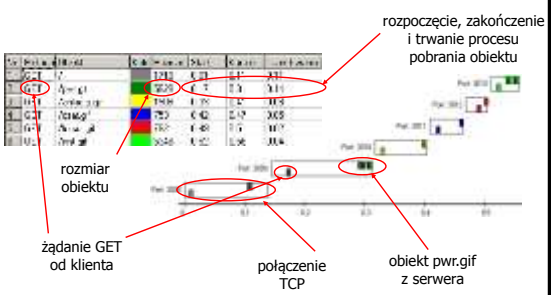
Lp.	Obiekt	Oznaczenie
1	ala.gif	
2	ola.gif	

Program DominoHTTP

Cel powstania:
- łatwa analiza wyników wykonanych doświadczeń

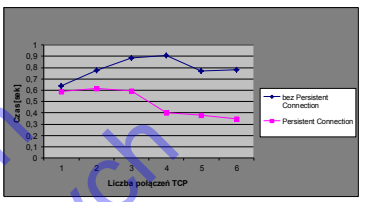
Wynik działania programu DominoHTTP:
- obserwacja liczby połączeń TCP
- prezentacja zastosowania metod: *persistent connection, pipelining*
- ukazanie informacji o obiektach zagnieżdżonych na stronie WWW (czas, rozmiar, nazwa, typ żądania, kolejność)

Informacje TCP, HTTP w programie DominoHTTP



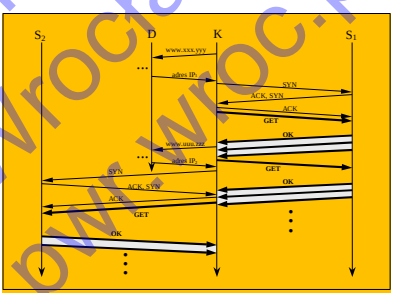
Przeprowadzony eksperyment

Pomiar efektywności użycia mechanizmu *persistent connection* oraz zmiennej liczby jednoczesnych połączeń TCP w procesie pobrania prostej strony WWW w wydzielonej sieci LAN.

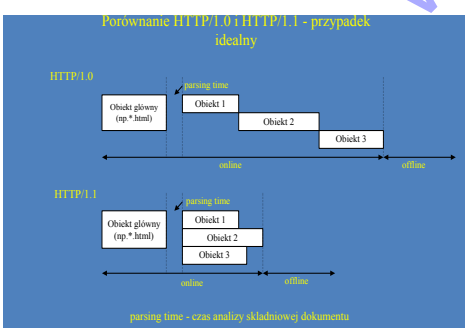


Jakość usług WWW - QoWS

Schemat przebiegu złożonej transakcji webowej



Jakość usług WWW - QoWS



Źródła spadku wydajności WWW

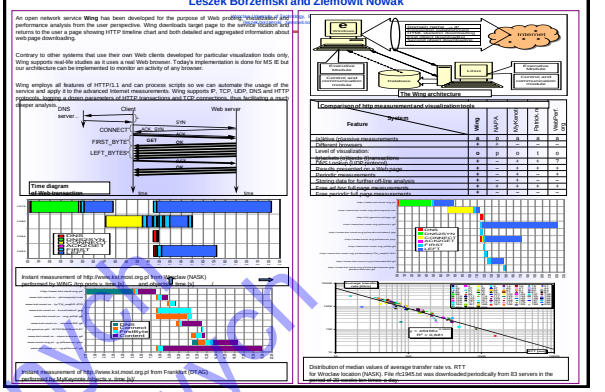
Fazy transakcji WWW	Źródła spadku wydajności
DNS	Zapytania DNS
CONNECT	Serwery WWW
FIRST_BYTE	Łącza
LEFT_BYTES	Routery
	Protokół TCP

Pomiary wydajności Internetu/Weba

- Opóźnienie (*latency*) [ms]
- Utrata pakietów (*packet loss*) [%]
- Przepustowość (*throughput*) [b/s]
- Wykorzystanie łącza (*link utilization*) [%]
- Pomiary na poziomie IP oraz HTTP

WING: A Web Probing, Visualization and Performance Analysis Service

Leszek Borzemski and Ziemowit Nowak

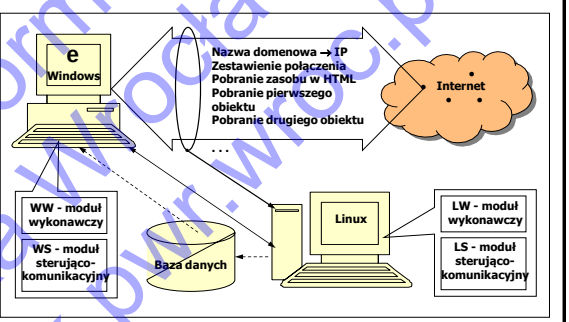


Systemy pomiarowe (wybór)

Comparison of http measurement and visualization tools

Feature \ System	Wing	NAPA	MyKenole	Patrick.net	WebPerf.org
(a)ktive (p)assive measurements	a	p	a	a	a
Different browsers	+	+	–	–	–
Level of visualization: (p)ackets (o)bjects (t)ransactions	o	p	o	t	o
DNS Lookup (UDP protocol)	+	–	+	+	?
Results presented on a Web page	+	–	+	+	+
Periodic measurements	+	–	+	–	+
Storing data for further off-line analysis	+	–	+	–	–
Free instant full page measurement	+	+	+	+	+
Free periodic full page measurements	+	–	–	–	–

System/Usługa Wing (Web ping)



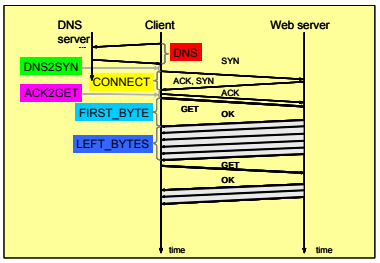
Założenia

- Ocena z punktu widzenia użytkownika końcowego posługującego się przeglądarką: IE, Mozilla, Opera,…
- Rozwiązanie programowe, nie sprzętowe
- Do dyspozycji jest jedynie „strona” użytkownika, ale bez instalacji czegokolwiek u użytkownika
- Potrzeba wysokiej rozdzielczości pomiaru czasu (1 ns)

Rozwiązanie

- Pomiary aktywne (nie bierne)
- Serwis webowy typu *request&result*
- Dialog z rzeczywistymi przeglądarkami
- Rozwiązanie typu *sonar*
- Wykorzystanie sniffera
- Gromadzenie pomiarów w bazie danych
- Pomiary *ad hoc* i programowane skryptami
- Przetwarzanie *on-line* i *off-line*

Schemat przebiegu prostej transakcji webowej



Co mierzy Wing?

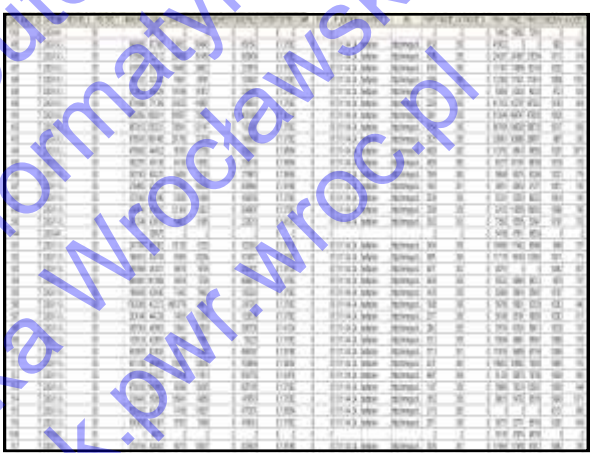
W wyniku analizy zgromadzonych danych powstaje drzewiasta struktura obiektów, zawierająca takie dane jak:

- czasy nadejścia każdego z pakietów należącego do obiektu,
- liczby poprawnych i niepoprawnych (wymagających retransmisji) pakietów,
- rozmiary każdego obiektu ustalone poprzez zliczenie bajtów przynależących do ciał obiektów,
- rozmiary obiektów pobrane z nagłówków HTTP,
- kolejności wystąpień podczas pobierania z uwzględnieniem równoległego pobierania obiektów na wielu portach przez przeglądarkę.

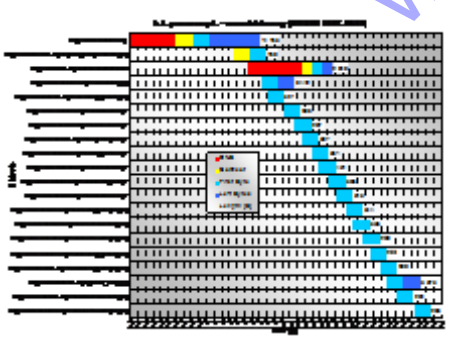
Co mierzy Wing?

Dla każdego z obiektów system mierzy i zapisuje w bazie danych m.in.:

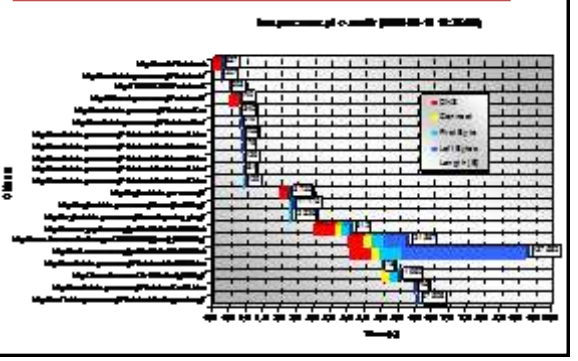
- ***DateTime*** – data i czas przeprowadzonego badania;
- ***Measurement*** – całkowity czas pobierania obiektu;
- ***DNS*** – czas potrzebny na "rozwinięcie" nazwy hosta do postaci adresu IP;
- ***Connect*** – czas potrzebny do ustanowienia połączenia z serwerem WWW na bazie protokołu TCP;
- ***First_Byte*** – czas jaki upłynął od wysłania żądania HTTP do momentu nadejścia pierwszego pakietu z odpowiedzią;
- ***IndexFile*** – czas potrzebny na pobranie obiektu zakodowanego w HTML;
- ***Content*** – sumaryczny czas potrzebny na załadowanie osadzonych obiektów;
- ***Start_MS*** – czas jaki upłynął od rozpoczęcia badania do momentu wysłania pakietu związanego z danym obiektem;
- ***URL*** – identyfikator URL obiektu;
- ***DNS2SYN*** – czas jaki upłynął pomiędzy otrzymaniem przez klienta ostatniego pakietu związanego z obsługą DNS a wysłaniem pakietu inicjującego połączenie TCP/IP;
- ***ACK2GET*** – czas jaki upłynął od momentu otrzymania przez klienta pakietu potwierdzającego zestawienie połączenia do momentu wysłania pakietu z żądaniem HTTP danego obiektu;



Wizualizacja prostej transakcji webowej



Wizualizacja złożonej transakcji WWW



Wing: ists.pwr.wroc.pl

- ..\index.htm

Testy publicznej wersji Winga

- IBM
- Microsoft
- Cisco
- Google
- Yahoo
- Onet
- Wirtualna Polska
- Gaz Wyborcza

WING – czy tylko ocena pojedynczego pobrania strony?

- WING – prezentacja ładowania strony WWW
- WING – długoterminowe pomiary i gromadzenie danych do dalszych badań metodami analizy danych i eksploracji danych
- WING – rozbudowa do wersji rozproszonej MWING

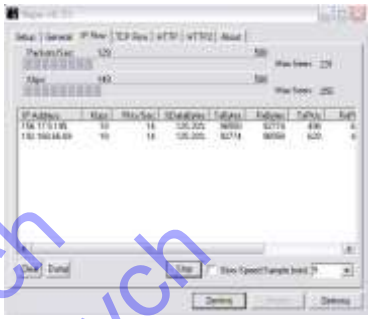
NAPA - Network Application Performance Analyzer (Intel)

- Podgląd protokołów sieciowych IP, TCP, HTTP
- Wizualizacja przechwyconych danych
- Wizualizacja transakcji webowych (porty, połączenia TCP, żądania, pobieranie zasobów)
- Uniwersalny wgląd „po stronie" klienta (dla dowolnej przeglądarki pod Windows)
- Implementacja u nas

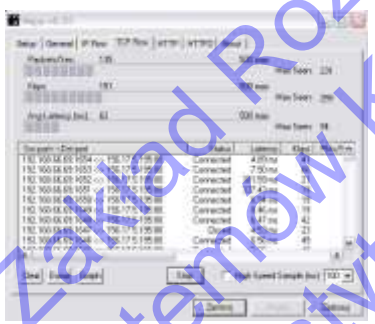
Setup

General

IP Flow



TCP Flow



HTTP Data Breakdown NAPA

- Categorizes HTTP data bytes sent
- Shows benefits of compressing text
- Helps analyze Get latencies

Good analysis for webpage developers

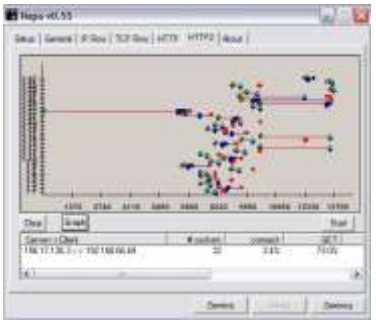
HTTP

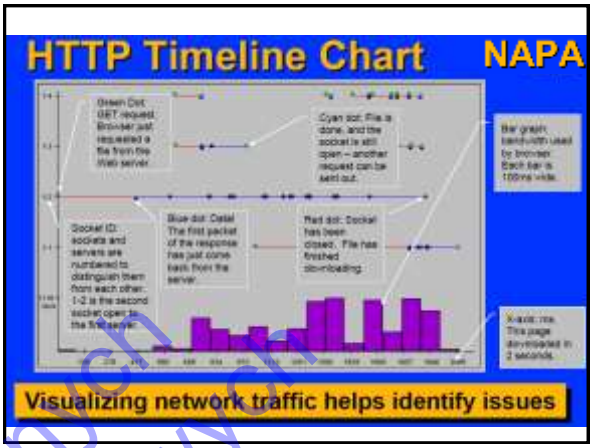
HTTP Time Breakdown NAPA

- Application socket usage
- Shows packets wrt time
- Easily identify latency & pipelining issues

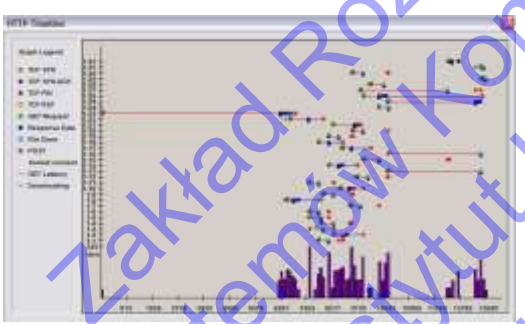
Good analysis for network app developers

HTTP 2

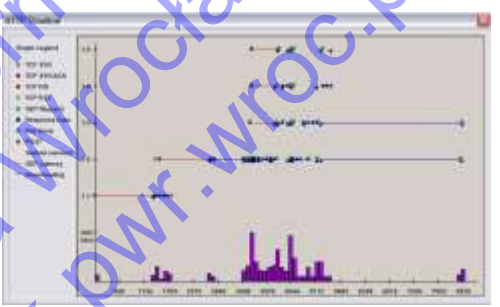




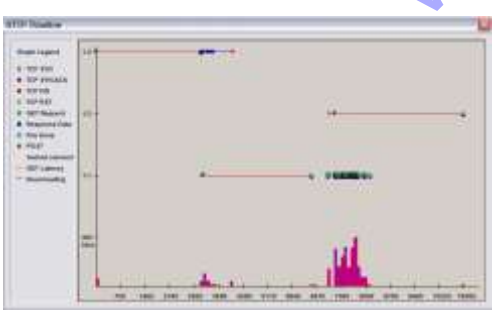
Graph- www.ists.pwr.wroc.pl- Opera



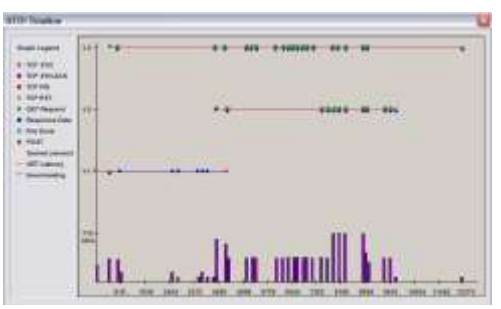
Graph – www.ists.pwr.wroc.pl - IE



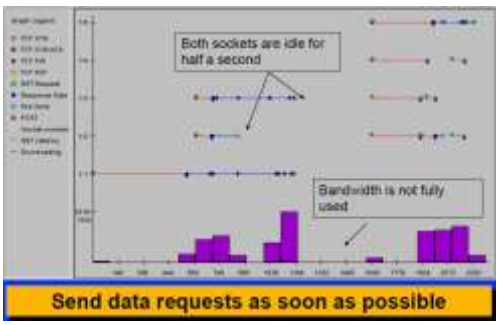
Graph – www.ists.pwr.wroc.pl - Netscape



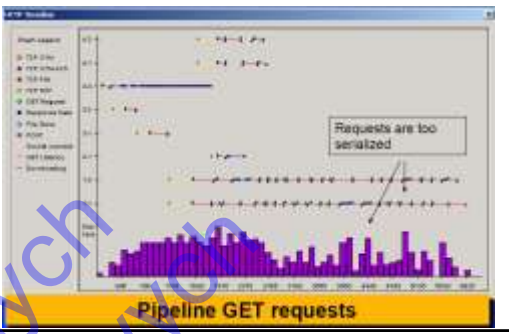
Graph – www.ists.pwr.wroc.pl - Mozilla



Przeglądarka C: Sieć LAN

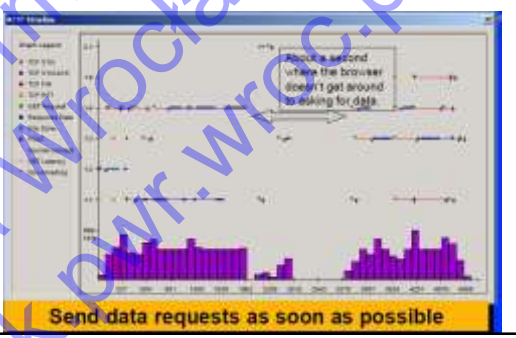


Przeglądarka C: Sieć DSL

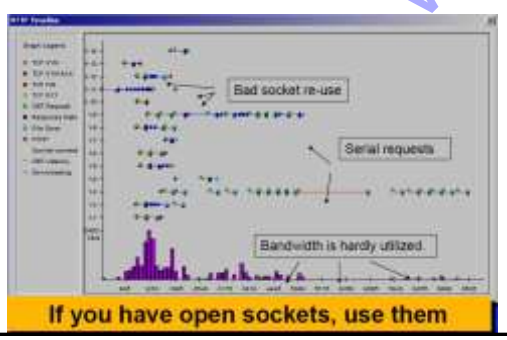


Przeglądarka B: DSL

Przeglądarka A: DSL



Przeglądarka A: DSL

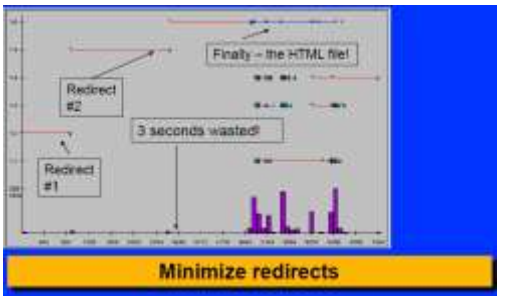


Przyspieszenie działania przeglądarek

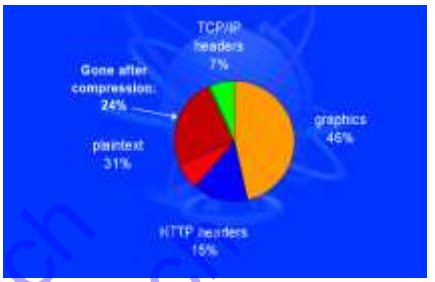
- Należy wysyłać żądania tak szybko jak to tylko jest możliwe
- Stosować strumieniowanie – przeglądarki nie są agresywne w tym względzie, a powinny
- Należy wykorzystywać otwarte już połączenia

- Ww działania mogą przynieść zwiększenie wydajności o 25%

Przekierowania



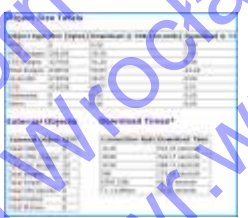
Dane przesyłane przez sieć



Wnioski dla serwerów WWW

- Kompresja HTML zmniejsza o 24% ruch HTTP – oszczędność czasu i pieniędzy – nawet na łączach szerokopasmowych!
- Serwer WWW ma standardowo wyłączoną opcję kompresji danych
- Nagłówki HTTP nie mogą być kompresowane – należy używać jak najmniejsze nagłówki – standardowo każdy pobierany plik ma ~800 Bajtów nagłówka
- Strony WWW powinny być małe!
- Duża liczba małych grafik zabija wydajność – każda grafika to nagłówek – wysyłane bez strumieniowania skutkuje opóźnieniami na serwerze

Web Page Analyzer



Wykaz elementów strony,
wraz z sugestiami dotyczącymi
ich wpływu na szybkość ładowania strony

Czasy ładowania dla różnych szybkości transmisji łącza

http://www.websiteoptimization.com

Web Page Analyzer

Por. plik pdf

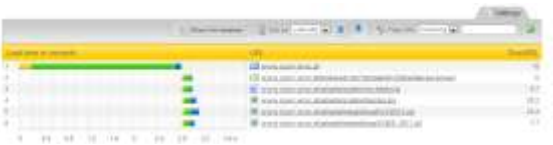
WatchScript - test ładowania www

Test WWW *- sprawdź jaki jest dokładny czas ładowania Twojej strony www, wpisz w poniższym polu adres i rozpocznij test.*
www.ii.pwr.wroc.pl. Całkowity czas ładowania: **2.989 sekundy**

Szczegóły testu:
Sprawdzany adres www: http://www.ii.pwr.wroc.pl
Data testu: 2010-10-19 20:49:07
Tytuł strony: Instytut I-32 Status: **OK**
Nagłówki serwera www:
Server: Apache/2.0.48 (Unix) PHP/4.3.4 mod_jk2/2.0.2
X-Powered-By: PHP/4.3.4
Transfer-Encoding: chunked
Czas odp. DNS: 0.008 sekundy
Czas połączenia: 0.008 sekundy
Czas zapytania: 3.075 sekundy
Czas odpowiedzi: 0.397 sekundy
Ilość pobranych danych: 85.15 kB
Prędkość pobierania: 28.49 kB/s

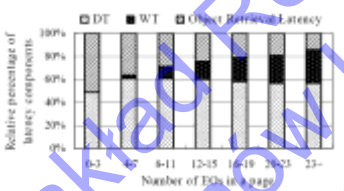
WatchScript - test ładowania www

- **Test the load time of a web page**
- **Testing finished: www.ii.pwr.wroc.pl**



Website information: Total loading time: 3.2 seconds, Total objects: 6 (85.1 KB) External objects: 0, (X)HTML: 1 (18KB) , RSS/XML: 0, CSS: 1 (0KB), Scripts: 1 (6.7KB), Images: 3 (60.4KB), Plugins: 0, Other: 0, Redirected: 0

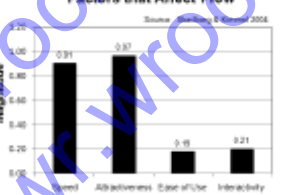
Składowe całkowitego czasu pobierania strony



DT – download time, WT – waiting time, EO – embedded object

Wraz ze wzrostem liczby obiektów rośnie udział czasu WT!

Motywacja wyboru strony



Skadberg, Y., and J. Kimmel. 2004. "Visitors' flow experience while browsing a Web site: its measurement, contributing factors and consequences. "*Computers in Human Behavior* 20 (3): 403–422

Speed wins…

•W celu zbudowania konkurencyjnego serwisu nie wystarcza już tylko wysoki poziom graficzny strony.

• Należy zapewnić jak najkrótszy czas dostępu do informacji na stronie.

Speed wins…

Amazon: każde 100 milisekund opóźnienia w otwieraniu strony kosztuje 1% sprzedaży.

Google: zwiększenie liczby wyników wyszukiwania wyświetlanych na stronie z 10 do 30 spowodowało, że ruch oraz przychody z reklamy spadły o 20% przy wzroście czasu ładowania strony z 0,4 do 0,9 sekundy.

Drobne ułamki sekund mogą decydować o wielkich pieniądzach!

Speed wins…

- Ze względu na architekturę protokołu http i sposób obsługi stron WWW trudno jest oszacować jednoznacznie czas ładowania strony.
- Liczenie czasu od chwili wpisania i zatwierdzenia adresu w przeglądarce do zakończenia ładowania widocznego w przeglądarce nie daje wiarygodnych wyników.
- Na przeszkodzie stoją m.in..:
  - rodzaj transmisji (synchroniczna/asynchroniczna),
  - ilość obsługiwanych połączeń,
  - szybkość transmisji,
  - zakłócenia sygnału,
  - rodzaj elementów stron (statyczne/dynamiczne),
  - konfiguracja przeglądarki, etc…

Podsumowanie

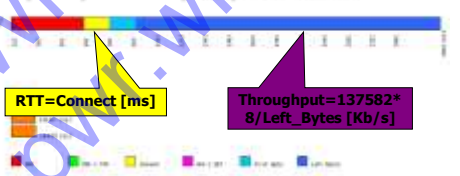
- Jest wiele różnych inicjatyw pomiarowych dla WWW o różnym zakresie i celach
- Jedynie niektóre z nich pozwalają na gromadzenie danych do dalszej analizy
- Gromadzenie danych wymaga opracowania specjalizowanych metod oraz narzędzi (baz danych, hurtowni danych)
- Ważne jest prowadzenie badań cyklicznych

WING – badania cykliczne

Badania cykliczne WING

- Wyznaczono obiekt (rfc1945.txt, 137 582 B) oraz lokalizacje jego kopii w kilkuset miejscach na świecie
- Zaprogramowano Winga tak, aby co kilka godzin pobierał obiekt z tych miejsc i odnotowywał w BD zaobserwowane zjawiska
- Po odpowiednim czasie zwrócono się o analizę

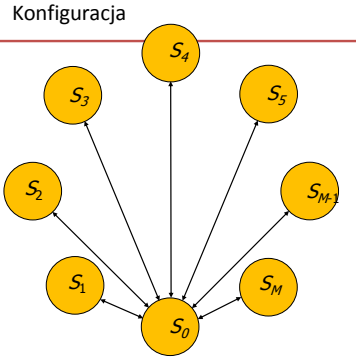
Czy RTT może być predyktorem przepustowości?



Badania wydajności i niezawodności usługi WWW

- Zaplanowano eksperyment aktywny z wykorzystaniem usługi Wing.
- Wylosowano próbkę 83 rozproszonych po całym świecie serwerów WWW.
- Z serwerów pobierano dokument o jednakowej wielkości **132 kB (rfc1945.txt**), dla którego:
  - czas transmisji był znacząco większy w stosunku do popełnianego błędu pomiarowego,
  - transmisja nie obciążała zbytnio sieci i serwerów.
- Zaprogramowano system **Wing** tak, aby transmisja dokumentu z każdego z 83 serwerów była inicjowana 10 razy na dobę przez 47 tygodni (od 21.09.02 do 28.07.2003). Zarejestrowano blisko 150 tys. transakcji webowych. [Pomiary są kontynuowane].
- Mierzono czas pomiędzy momentami nadejścia pierwszego i ostatniego pakietu zawierającego dokument.

Konfiguracja

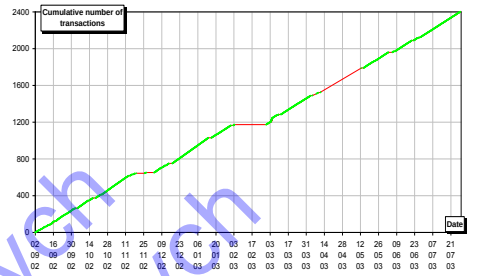


Sieć WASC we Wrocławiu

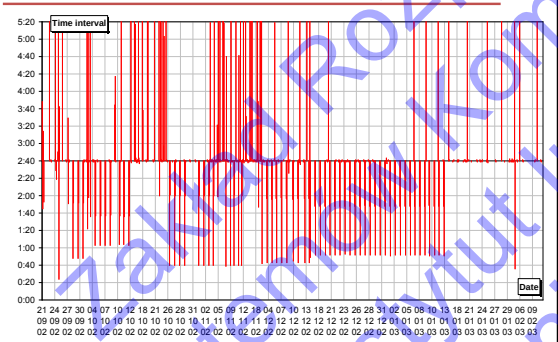
Lista serwerów (fragment)

#	SERVER NAME	COUNTRY	CITY	DISTANCE [km]
2	199.125.85.46	US	MANCHESTER	6350
4	www.fiuba6662.com.ar	NL	-	788
5	ftp.univie.ac.at	AT	VIENNA	323
9	ironbark.bendigo.latrobe.edu.au	AU	BENDIGO	15617
14	cs.anu.edu.au	AU	CANBERRA	15844
16	files.ruca.ua.ac.be	BE	ANTWERP	876
19	www.deadly.ca	CA	CALGARY	7754
21	www.munet.mun.ca	CA	-	6286
24	tecfa.unige.ch	CH	GENEVE	962
27	www.embed.com.cn	CN	BAO_AN	8552
33	www.cgisecurity.com	US	MANCHESTER	6350
37	www.gordano.com	UK	BRISTOL	1358
40	www.networksorcery.com	US	HERNDON	7017
41	www.q-linux.com	PH	MAKATI	9697
43	www.sasknet.com	US	PROVO	8702
44	docs.securepoint.com	US	LANEXA	7109
47	www.soldierx.com	US	PANGUITCH	8970
51	salsero.ibp.cz	CZ	BRNO	210
52	bbs1.biz-worms.de	DE	WORMS	634
54	www.deadlyzone.de	DE	KARLSRUHE	654

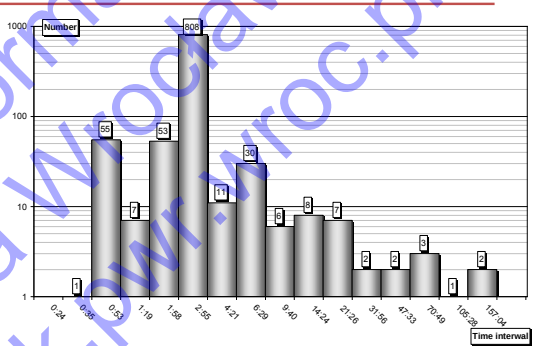
Wykres ciągłości badań serwera # 2



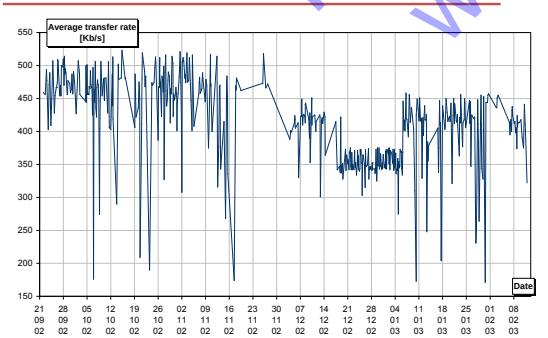
Rozkład interwałów pomiarowych dla serwera #2



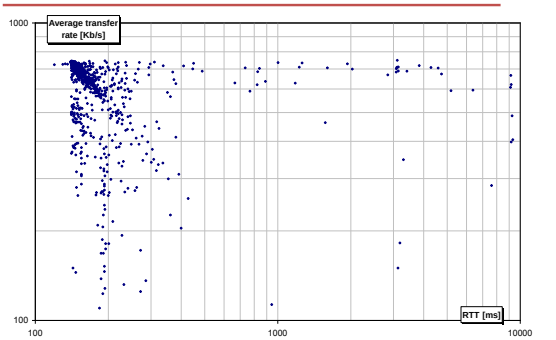
Histogram interwałów pomiarowych dla serwera #2



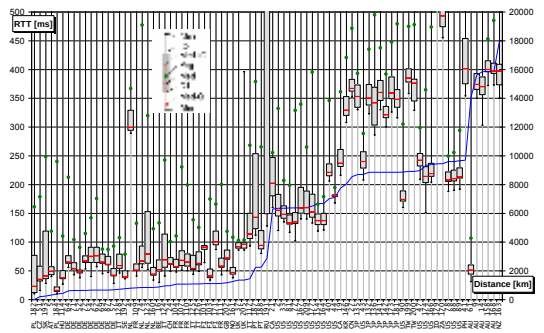
Rozkład średnich czasów transmisji dla serwera #2



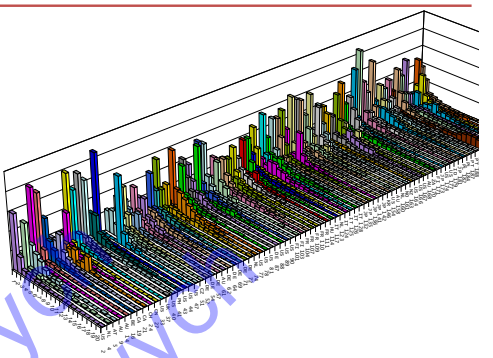
Rozkład czasów transmisji vs RTT dla serwera #2



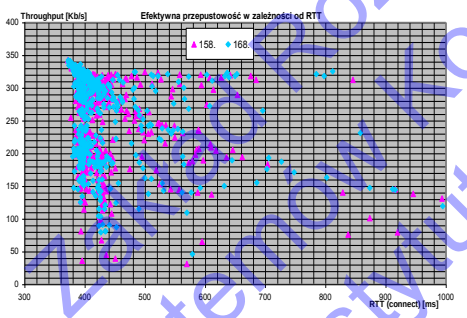
Miary pozycyjne czasów transmisji -wykres „Box & whisker” dla RTT



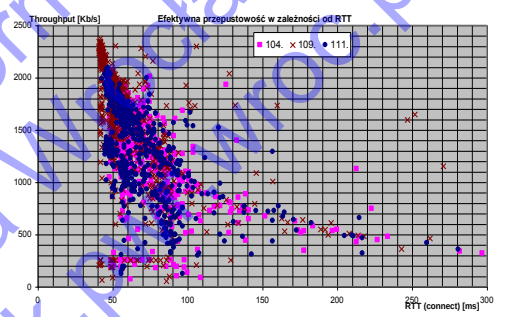
Rozkłady czasów transferu (mediany)



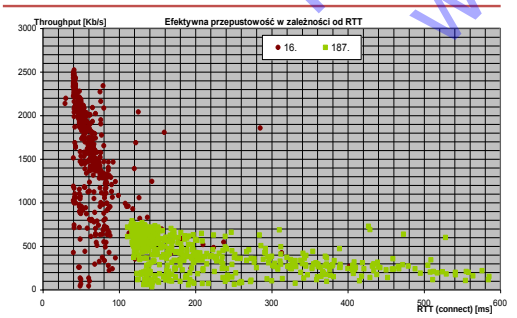
158. http://www.potaroo.net/ietf/rfc/rfc1945.txt AS1221
168. http://ietfreport.isoc.org/rfc/rfc1945.txt AS1221



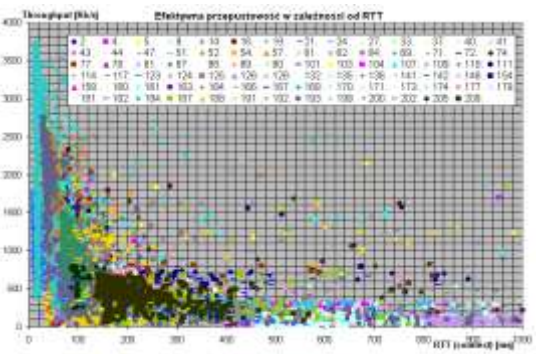
104. http://www.tls.cena.fr/~boubaker/WWW/rfc1945.txt AS2200
109. http://www.loria.fr/~fleury/CPS/I33/rfc1945.txt AS2200
111. http://eurise.univ-st-etienne.fr/infsci/rfc1945.txt AS2200

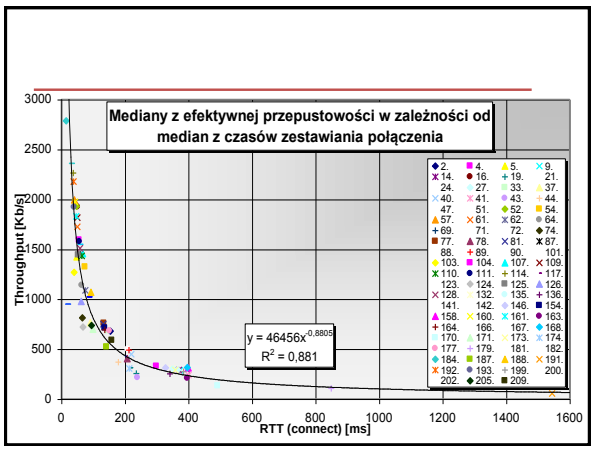


16. http://files.ruca.ua.ac.be/.../rfc1945.txt AS11042 BE
187. http://www.ave.dee.isep.ipp.pt/.../rfc1945.txt AS11042 PT



Zależność przepustowości od RTT – wszystkie wyniki





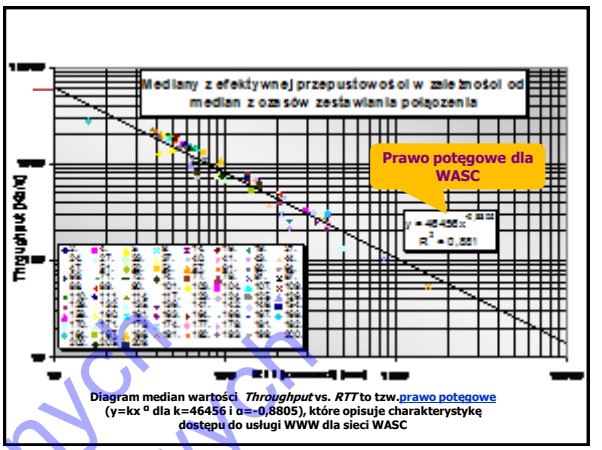
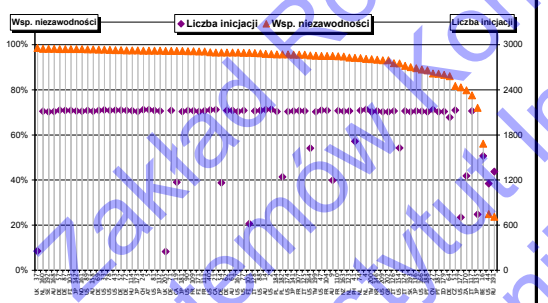
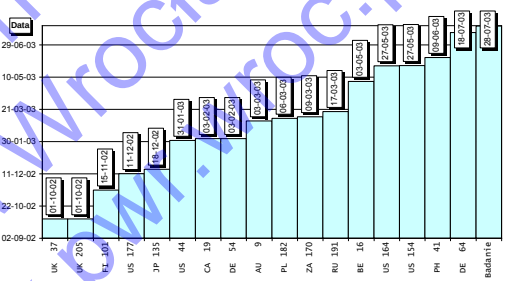


Diagram median wartości *Throughput* vs. *RTT* to tzw. prawo potęgowe ($y = kx^{\alpha}$ dla $k = 46456$ i $\alpha = -0{,}8805$), które opisuje charakterystykę dostępu do usługi WWW dla sieci WASC

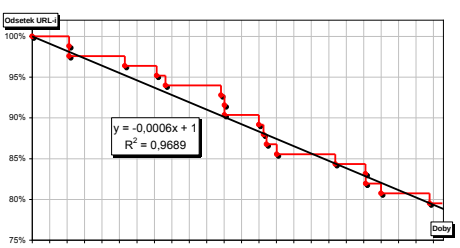
Współczynniki niezawodności na tle liczby inicjacji poszczególnych serwerów WWW



Wykres momentów „śmierci" odnośników webowych na serwerach WWW

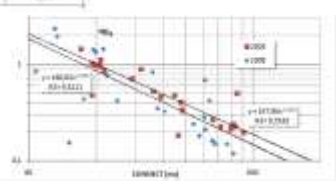


Model śmiertelności linków



Internet – Wrocław - zmiany po pięciu latach

Czas trwania fazy	Średnia arytmetyczna [ms]		Współczynnik przyspieszenia
	2003 r.	2008 r.	
DNS	466	135	2,75
CON	269	232	1,16
FIRST_BYTE	498	219	2,27
LEFT_BYTES	2672	3143	0,85
TOTAL	4268	3930	1,16



Źródła danych o Internecie

Wstęp

- Systemy wyszukiwarek mogą być użyteczne w analizie zachowań użytkowników w Internecie, np. w celu:
  - eksploracji liczby zapytań przypadających na jedną sesję użytkownika
  - modelowania zachowań zależnych od pory dnia
  - predykcji, które strony są relewantne
- Zastosowania analiz:
  - zrozumienie wpływów społecznych na użycie Weba
  - projektowanie lepszych narzędzi dostępu do informacji
  - zastosowanie e-biznesowe

Proces przygotowania danych

- Dane wykorzystywane w badaniach Weba mogą być gromadzone na różnych poziomach:
  - na poziomie serwerów WWW,
  - na poziomie klientów,
  - na poziomie serwerów pośredniczących (*Proxy*),
  - na poziomie specjalnych systemów pomiarowych

Jak rejestrowana jest nawigacja po Webie

- Logi webowe (logi serwerów WWW)
  - Rekordy o aktywności pomiędzy przeglądarką a danym serwerem WWW
  - Łatwo dostępne
  - Ich wartość może być „wzmocniona" przez zastosowanie ciasteczek zawierających informację o „stanie"
- Rekordy wyszukiwarek
  - Tekst w kwerendach; odsyłacze, które aktywowaliśmy, itp..
- Rekordy po stronie przeglądarek klienckich
  - Produkowane dla określonych celów badawczych
  - Automatycznie rejestrowane przez oprogramowanie klienckie
  - **Trudniejsze do uzyskania, ale mogą być wartościowsze do analiz aniżeli logi po stronie serwera**
- Inne źródła
  - Informacja rejestracyjna na stronach webowych, zakupy, maile, itp
  - Informacja rejestrowana przez ISP na temat odwiedzanych stron
  - Informacja rejestrowana przez *infrastruktury pomiarowe* podobne do WINGa

Problemy pomiarów webowych

- Należy rozumieć jak dane są zbierane
- Dane webowe są zbierane automatycznie przez odpowiednie narzędzia
  - plusy:
    - Niepotrzebne jest nasze zaangażowanie aby dokonać pomiarów
  - minusy:
    - Dane mogą być zaszumione (ze względu na roboty)
- Należy umieć zidentyfikować i odfiltrować ruch pochodzący od robotów

Pliki serwerów webowych

- Log dot. pobierania zasobów z danego serwera:
  - Transakcje pomiędzy przeglądarką a serwerem
  - Adres IP, czas żądania
  - Metoda użyta w żądaniu (GET, HEAD, POST…)
  - Kod statusu dla odpowiedzi z serwera
  - Rozmiar (w bajtach) wielkości transakcji
- Log dot. adresu referującego (odsyłającego) (Referrer):
  - adres, z jakiego nastąpiło przekierowanie użytkownika na naszą stronę internetową
- Log dot. agenta :
  - Oprogramowanie przeglądarki (pająka) wysyłającego żądanie
- Log błędu:
  - Żądanie zakończone błędem (404)

Standardy logów serwerów webowych

- Pliki z logami WWW tworzone są według różnych formatów (30+).
- Do najczęściej stosowanych należą ***Common Logfile Format*** i ***Extended Logfile Format***.
- W obu formatach w plikach zapisywane są kolejne żądania skierowane do serwera - w jednej linii znajdują się informacje dotyczące pojedynczego żądania.
- http://publib.boulder.ibm.com/tividd/td/ITWSA/ITWSA_info45/en_US/HTML/guide/c-logs.html - informacja o różnych logach

Common Logfile Format (CLF)

- Według standardu ***Common Logfile Format*** (CLF) pojedynczy zapis w logu powinien mieć następującą postać:
  `remotehost rfc931 authuser date:time „request” status bytes`
- Pole **`remotehost`** oznacza nazwę lub adres IP komputera, z którego nastąpiło odwołanie
- Pole **`rfc931`** zawiera nazwę użytkownika na danym komputerze
- W polu **`authuser`** znajduje się informacja za kogo podaje się użytkownik
- W polu **`date:time`** znajduje się stempel czasowy oznaczający moment żądania (zawiera datę i czas z różną dokładnością, najczęściej do jednej sekundy)
- Pole **`request`** zawiera żądanie przesłane do serwera w takiej formie w jakiej zostało wygenerowane przez klienta, dodatkowo zawiera informacje o użytej metodzie i wersji protokołu
- Pole **`status`** informuje o kodzie odpowiedzi zwracanej klientowi zgodnie z protokołem HTTP
- Pole **`bytes`** zawiera rozmiar przesłanego zasobu.
- Jeżeli niemożliwe jest określenie wartości w polach: `rfc931`, `authuser`, `status`, `bytes`, to wstawiany jest znak „-”

Przykład rekordu w CLF

```
192.168.0.125 - dsmith [10/Oct/1999:21:15:05
+0500] "GET /index.html HTTP/1.0" 200 1043
```

- **`remotehost`** (**`192.168.0.125`** w przykładzie)
- **`rfc931`** ("-" w przykładzie)
- **`authuser`** (dsmith w przykładzie)
- **`date:time timezone`** ([10/Oct/1999:21:15:05 +0500] w przykładzie)
- **`request`** ("GET /index.html HTTP/1.0" w przykładzie)
- **`status`** (200 w przykładzie)
- **`bytes`** (1043 bajtów w przykładzie, bez nagłówka HTTP)

Czy to jest czas obsługi żądania?

- Pola w opisie daty i czasu są zdefiniowane następująco:

  `[dd/MMM/yyyy:hh:mm:ss +-hhmm]`

gdzie:

- **`dd`** – dzień miesiąca
- **`MMM`** - miesiąc
- **`yyy`** - rok
- **`:hh`** - godzina
- **`:mm`** - minuta
- **`:ss`** – **sekundy!**
- **`+-hhmm`** – strefa czasowa
- **Jaki to jest czas?**

Combined Logfile Format – rozszerzenie CLF

- Combined Logfile Format przewiduje dodatkowe dwa pola umieszczane na końcu linii:
  - **`referrer`**
  - **`user_agent`**
  - pierwsze zawiera informacje o stronie z jakiej nastąpiło aktualne odwołanie,
  - drugie pole informuje o typie przeglądarki, która wygenerowała żądanie.
- Dane gromadzone na poziomie serwera nie zawierają jednak wszystkich informacji o aktywności danego użytkownika na witrynie, spowodowane to jest stosowaniem technik *cache* na różnych poziomach w środowisku sieciowym.
- W plikach z logami z nie ma żadnych informacji na temat stron pobieranych z ukrytych zasobów (*cache*).

Extended Logfile Format

- Dwie dodatkowe dyrektywy: **`Version`** i **`Fields`**, które mówią jak przetwarzać log. Są one umieszczone na początku rekordu
  - `#Version: 1.0`
  - `#Fields: date time c-ip sc-bytes time-taken cs-version`
- Poza tym administrator może dodawać swoje pola
- W **IIS:** `c-ip cs-username s-sitename s-computername s-ip s-port cs-method cs-uri-stem cs-uri-query sc-status sc-win32-status sc-bytes cs-bytes time-taken cs-version cs-host cs(User-Agent) cs(Cookie) cs(Referer)`

W3C Extended Log File Format

Field	Date	Description
Date	date	The date that the activity occurred
Time	time	The time that the activity occurred
Client IP address	c-ip	The IP address of the client that accessed your server
User Name	cs-username	The name of the authenticated user who access your server, anonymous users are represented by -
Servis Name	s-sitename	The Internet service and instance number that was accessed by a client
Server Name	s-computername	The name of the server on which the log entry was generated
Server IP Address	s-ip	The IP address of the server that accessed your server
Server Port	s-port	The port number the client is connected to
Method	cs-method	The action the client was trying to perform
URI Stem	cs-uri-stem	The resource accessed
URI Query	cs-uri-query	The query, if any, the client was trying to perform
Protocol Status	sc-status	The status of the action, in HTTP or FTP terms
Win32 Status	sc-win32-status	The status of the action, in terms used by Microsoft Windows
Bytes Sent	sc-bytes	The number of bytes sent by the server
Bytes Received	cs-bytes	The number of bytes received by the server
Time Taken	time-taken	The duration of time, in milliseconds, that the action consumed
Protocol Version	cs-version	The protocol (HTTP, FTP) version used by the client
Host	cs-host	Display the content of the host header
User Agent	cs(User Agent)	The browser used on the client
Cookie	cs(Cookie)	The content of the cookie sent or received, if any
Referrer	cs(Referrer)	The previous site visited by the user. This site provided a link to the current site

s = server actions
c = client actions
cs = client-to-server actions
sc = server-to-client actions

Przykład zapisów w logu (Apache)

```
205.188.209.10 - - [29/Mar/2002:03:58:06 -0800] "GET
   /~sophal/whole5.gif HTTP/1.0" 200 9609
   "http://www.csua.berkeley.edu/~sophal/whole.html"
   "Mozilla/4.0 (compatible; MSIE 5.0; AOL 6.0; Windows 98;
   DigExt)"
216.35.116.26 - - [29/Mar/2002:03:59:40 -0800] "GET
   /~alexlam/resume.html HTTP/1.0" 200 2674 "-" "Mozilla/5.0
   (Slurp/cat; slurp@inktomi.com;
   http://www.inktomi.com/slurp.html)"
202.155.20.142 - - [29/Mar/2002:03:00:14 -0800] "GET
   /~tahir/indextop.html HTTP/1.1" 200 3510
   "http://www.csua.berkeley.edu/~tahir/" "Mozilla/4.0
   (compatible; MSIE 6.0; Windows NT 5.1)"
202.155.20.142 - - [29/Mar/2002:03:00:14 -0800] "GET
   /~tahir/animate.js HTTP/1.1" 200 14261
   "http://www.csua.berkeley.edu/~tahir/indextop.html"
   "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
```

Bogactwo informacji?

lbz000.ust.hk - - [16/Nov/2009:12:03:26 +0800] "GET /catalog/ HTTP/1.1" 200 20283 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5 (.NET CLR 3.5.30729)"

lbnxyz.ust.hk - - [16/Nov/2009:12:03:27 +0800] "GET /catalog/?s=brandy&feed=rss HTTP/1.1" 304 - "-" "Feedfetcher-Google; (+http://www.google.com/feedfetcher.html; 1 subscribers; feed-id=10486796160015392754)"

lbz222.ust.hk - - [16/Nov/2009:12:03:30 +0800] "GET /stream/xml/stream.xml HTTP/1.1" 304 - "-" "Mozilla/5.0 (Windows; U; Windows NT 6.0; zh-TW; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5"

lbz333.ust.hk - - [16/Nov/2009:12:03:33 +0800] "GET /catalog/?s=brandy HTTP/1.1" 304 - "-" "Mozilla/5.0 (Windows; U; Windows NT 6.0; zh-TW; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5"

lbz444.ust.hk - - [16/Nov/2009:12:03:35 +0800] "GET /stream/xml/stream.xml HTTP/1.1" 304 - "-" "Mozilla/5.0 (Windows; U; Windows NT 6.0; zh-TW; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5"

Jeden z rekordów

lbz000.ust.hk - - [16/Nov/2009:12:03:26 +0800] "GET /catalog/ HTTP/1.1" 200 20283 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5 (.NET CLR 3.5.30729)"

Pola	Wartość
Remote host field	lbz000.ust.hk
Date/Time field	[16/Nov/2009:12:03:26 +0800]
HTTP request	"GET /catalog/ HTTP/1.1"
Status code field	200
Transfer Volume (Bytes) Field	20283
User agent field	"Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5 (.NET CLR 3.5.30729)"

Common Log Format – zwykle: Apache Web server, Apache Tomcat,

lbz000.ust.hk - - [16/Nov/2009:12:03:26 +0800] "GET /catalog/ HTTP/1.1" 200 20283 "-" "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5 (.NET CLR 3.5.30729)"

Microsoft IIS Log Format

2009-07-20 01:22:44 GET /ce/ - 66.249.71.201 HTTP/1.1 Mozilla/5.0+(compatible;+Googlebot/2.1;++http://www.google.com/bot.html) - 401 1891 0

zawiera:
- Remote host field
- Date field
- Time field
- HTTP request field
- Status code field
- Transfer Volume (Bytes)
- Referrer field
- User agent field

185

Microsoft Streaming Server np. Streaming video

143.89.160.133 2009-09-02 10:21:20 - /arc-open/oudpa/OUDPA-2008-Sobel-Adventures_in_Science_Writing.wmv 0 6 5 200 {3300AD50-2C39-46c0-AE0A-41B7139D4722} 11.0.5721.5251 en-US WMFSDK/11.0.5721.5251_WMPlayer/11.0.5721.5268 - wmplayer.exe 11.0.5721.5145 Windows_XP 5.1.0.2600 Pentium 3816 216613290 2830093 rtsp TCP - - - 2244972 2244972 398 398 0 0 0 0 0 0 1 1 100 143.89.105.168 lbms07.ust.hk 1 0 - 245 file://C:\wmhome\hkust\arc-open\oudpa/OUDPA-2008-Sobel-Adventures_in_Science_Writing.wmv mms://stream.ust.hk/arc-open/oudpa/OUDPA-2008-Sobel-Adventures_in_Science_Writing.wmv OUDPA-2008-Sobel-Adventures_in_Science_Writing.wmv - - 0

Pola tylko dla serwera strumieniowego:
- kodek Video
- kodek Audio
- czas trwania
- odtwarzacz kliencki

186

Narzędzia do analizy logów

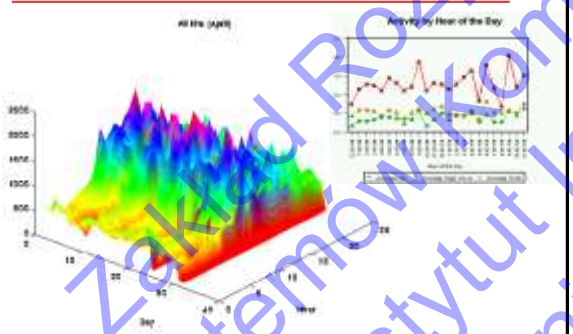
- AccessWatch v1.33
- Analog 6.0
- Pwebstats
- RefStats 1.2
- INNOPAC Millennium Web Report – Search Statistics
- AWStats
- Sawmill Analytics
- Webalizer
- WebTrends
- Live Stats
- HTTP-Analyze
- Nihuo Web Log Analyzer
- …

187

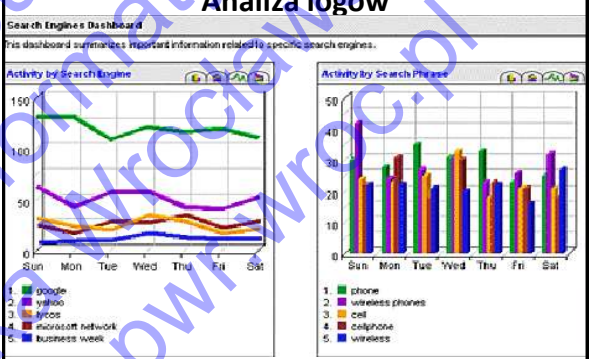
Typowe analizy logów serwerów

- Najczęściej i najrzadziej odwiedzane strony
- Strony poprzez które weszliśmy i poprzez które wyszliśmy (Entry and exit pages)
- Odwołania z innych ośrodków lub wyszukiwarek
- Jakie słowa kluczowe są używane
- Raport z błędów, nieprawidłowych odwołań
- Do jakich innych analiz mogą być przydatne logi?

Wizualizacja danych (w osi czasu)

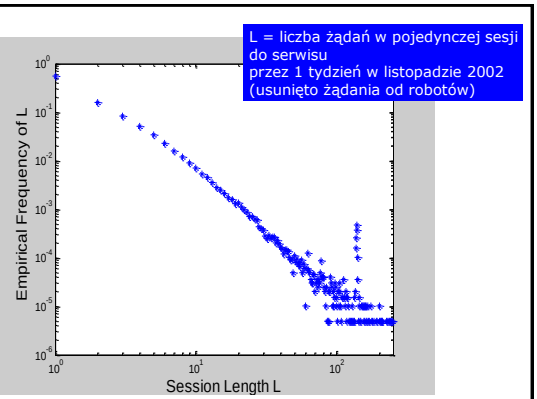


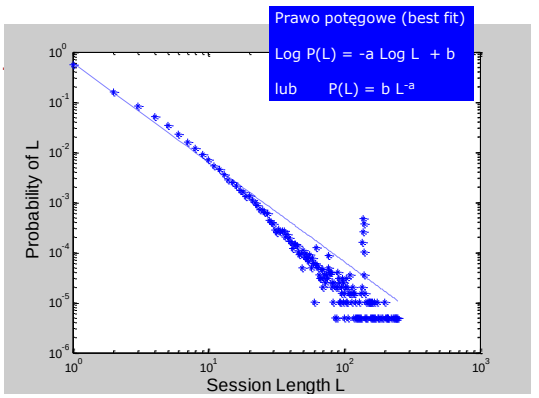
Analiza logów

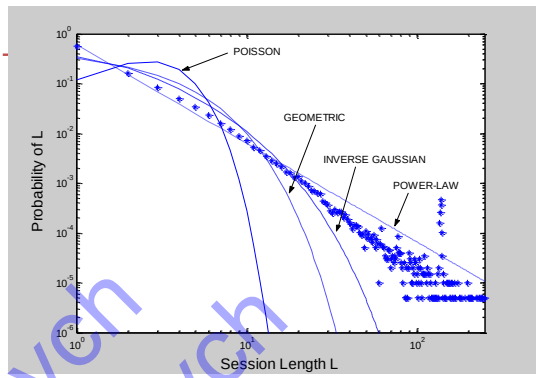


Statystyki

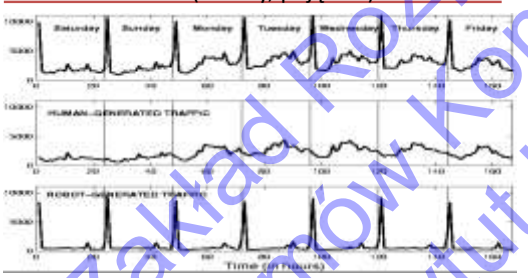
- Histogramy, wykresy czasowe
  - Bardzo ważne!
  - Pomagają zrozumieć Web
  - Dostarczają uzupełniającą wiedzę przy modelowaniu
    - Modele agregują zachowanie, nie pokazują indywidualnych sytuacji
- Przykłady
  - Długość sesji (np. prawo potęgowe)
  - Ilość kliknięć w funkcji czasu







Żądania generowane są przez ludzi i automaty (roboty, pająki….)



Liczba żądań na godzinę w funkcji czasu dla stron pobieranych z www.ics.uci.edu (logi z pierwszego tygodnia kwietnia 2002)

1998 World Cup Web Site

Mundial w Internecie

- Internet był jedną z form bezpośredniego przekazu rozgrywek piłkarskich na tegorocznych mistrzostwach świata
- Infrastruktura techniczna była przygotowana do obsługi ok. miliarda połączeń podczas mundialu.
- Planowana dziennie liczba odwołań do stron umieszczonych na serwerze www.france98.com - według przewidywań – miała wynosić 25-50 mln.
- Serwis obsługiwały trzy osobne zespoły serwerów w Ameryce, Azji i Europie.
- Wykorzystanie serwerów WWW regulowło oprogramowanie bilansujące połączenia z adresem www.france98.com.

Mundial w Internecie

- Do tworzenia stron wykorzystano dane umieszczone w bazie równoległej rozdzielonej między grupę serwerów.
- Bazę zasilały dane replikowane przez cały czas z centralnego serwera - tzw. Content Engine.
- Było to źródło wszystkich serwisów informacyjnych mistrzostw, - zarówno intranetu wykorzystywanego jedynie przez akredytowanych dziennikarzy, jak i serwera WWW .

Analiza logów na przykładzie witryny www.france98.com

- Analiza logów na przykładzie pracy „*Workload Characterization of the 1998 World Cup Web Site*” autorstwa Martina Arlitta oraz Tai Jina.
- Strona www.france98.com była przez wiele lat największym źródłem danych dotyczących odwiedzin ośrodka webowego, miała ona ich aż 1,35 miliarda, w przeciągu zaledwie 3 miesięcy.
- Logi zostały poddane anomizacji do postaci:

```
Adres IP hosta, identyfikator
  użytkownika, data żądania, żądany zasób,
  odpowiedź w kodzie HTTP oraz wielkość
  przesłanej informacji
```

Podsumowanie analizy

Table 1 Summary of Access Log Characteristics (Raw Data)

Duration	May 1st - July 23, 1998
Total Requests	1,352,804,107
Avg Requests/Minute	10,796
Total Bytes Transferred (GB)	4,991
Avg Bytes Transferred/Minute (MB)	40.8

0,02 % użytkowników nie miało poprawnie sformułowanego protokołu HTTP

Table 2 Breakdown of HTTP Version Supported by Client

HTTP Version	% of Requests	% of Content Data Transferred
0.9	0.00	0.00
1.0	79.08	79.83
1.1	21.32	20.09
x.x	0.02	0.08
Total	**100.00**	**100.00**

Kody odpowiedzi serwera HTTP

Table 4 Breakdown of Server Response Codes

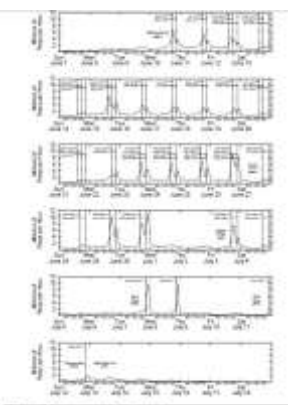
Response Code	% of Requests	% of Content Data Transferred
200 (Successful)	80.32	97.86
206 (Partial Content)	0.09	2.08
304 (Not Modified)	18.75	0.00
4xx (Client Error)	0.64	0.06
5xx (Server Error)	0.00	0.00
Other Codes	0.00	0.00
Total	**100.00**	**100.00**

Zasoby głównie statyczne!

Table 5 Breakdown by File Type

File Type	% of Requests	% of Content Data Transferred
HTML	9.85	38.60
Images	88.16	35.02
Audio	0.02	0.10
Video	0.00	0.82
Compressed	0.08	20.33
Java	0.82	0.83
Dynamic	0.02	0.38
Other Types	1.05	3.92
Total	**100.00**	**100.00**

Ruch godzinowy



Ruch dzienny

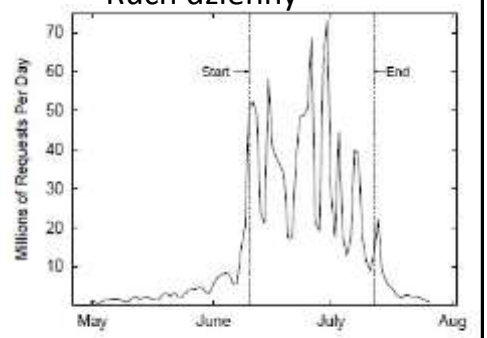


Figure 1 Daily Traffic Volume to the World Cup Web Site

A co z World Cup 2014?

Problemy:

- Jak odetkać Internet?
- Jak nie dopuścić do zatkania Internetu?

Badania wydajnościowe - konkluzje

- Wykorzystanie logów serwerowych jest wątpliwe, a logi są praktycznie nieprzydatne w analizie wydajnościowej
- Należy prowadzić pomiary aktywne zamiast biernych
- Należy sterować eksperymentem pomiarowym
- Należy oceniać z punktu widzenia klienta

WWW

WWW = World Wide Web

WWW = World Wide Wait

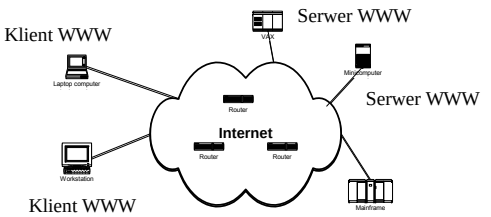
Obciążenie serwisów WWW

Ośrodek/wydarzenie	Okres obserwacji /całkowita liczba żądań	Max obciążenie w ciągu jednego dnia	Szczytowe obciążenie minutowe
Serwer NCSA	październik 1995	2 mln	b.d.
Letnie Igrzyska Olimpijskie, USA 1996	180 mln	8 mln	b.d.
Wybory prezydenckie w USA	listopad 1996	9 mln	b.d.
Projekt NASA Pathfinder – lipiec 1997	942 mln (14 dni)	40 mln	b.d.
Zimowe Igrzyska Olimpijskie, Japonia, 1998	635 mln (15 dni)	57 mln	110 000
Piłkarskie Mistrzostwa Świata, Francja, 1998	1 350 mln (90 dni)	73 mln	209 000
Wimbledon, Wielka Brytania, Czerwiec 1999	942 mln (14 dni)	125 mln	430 000

Geneza problemu

- Znaczenie usługi WWW
- Potrzeba zapewnienia wydajnego działania serwisów WWW
- Potrzeba zagwarantowania jakości usług (ang. *Quality of Service*) i QoWS (ang. ***Quality of Web Service***)
  - Wydajność
  - Niezawodność
  - Dostępność z różnych klientów
  - Bezpieczeństwo

Jakość usług WWW – QoWS i QoS

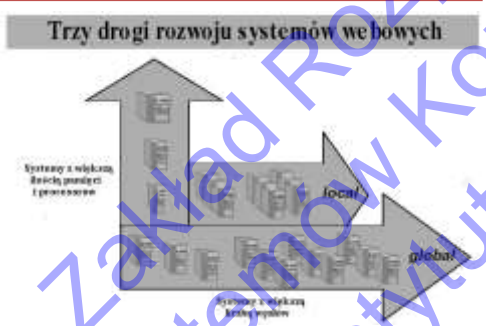


QoS – jakość transmisji
QoWS – jakość usługi WWW

Sposoby zapewniania jakości usług WWW

- Zapewnienie nadwyżki zasobów
  - Pamięć, procesor
- Wykorzystanie rozproszonych i klastrowych systemów webowych
- Specjalna (dodatkowa) infrastruktura w Internecie
  - Proxy, cache
- Regulacja ruchu
- Specjalne usługi globalne i lokalne
  - CDN
- Zwiększanie wydajności serwerów WWW
  - Sterowanie dostępem, szeregowanie żądań
  - Pamięć podręczna

Jakość usług WWW - QoWS



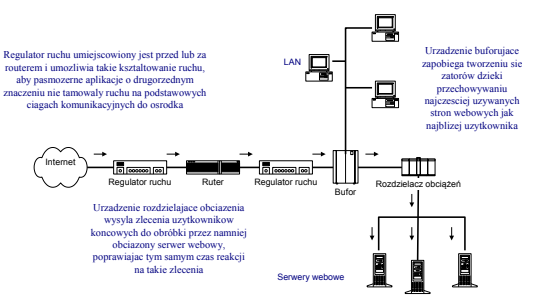
Mechanizmy poprawiające QoWS

- Regulatory ruchu webowego –kształtowanie ruchu i aktywna kontrola nad ruchem „niepożądanym”
- Urządzenia buforujące dane webowe – oszczędzanie pasma przez zaniechanie transmisji
- Rozdzielacze obciążeń warstwy 4 i 7 – dystrybucja zapytań i równoważenie obciążeń serwerów webowych
- Globalnie rozmieszczone serwery z zawartością internetową – np. sieć serwerów Akamai - fizyczne zbliżenie zawartości Internetu do użytkowników
- Nie-neutralna organizacja obsługi żądań HTTP (tam gdzie jest to możliwe i przydatne)

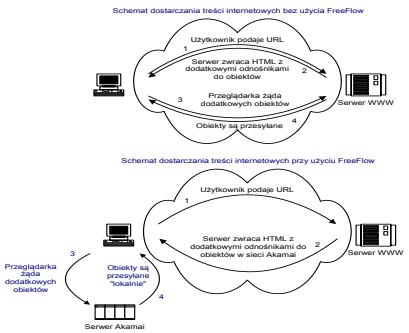
Lokalizacja mechanizmów poprawiających jakość usługi WWW

- Strona kliencka (przeglądarka) (QoWS)
- Strona ośrodka webowego (serwer WWW) (QoWS)
- Infrastruktura szkieletowa Internetu (QoS)
- Infrastruktura sieci dostępowej ośrodka (QoS)
- Infrastruktura sieci dostępowej klienta (QoS)
- „Krawędź Internetu” (QoWS)
- Inne …

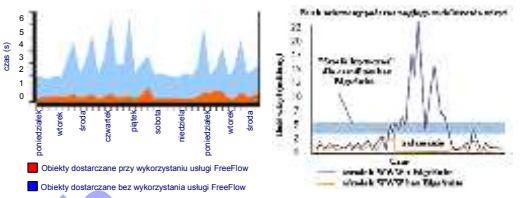
Mechanizmy poprawiające QoWS



Mechanizmy poprawiające QoWS - przykład

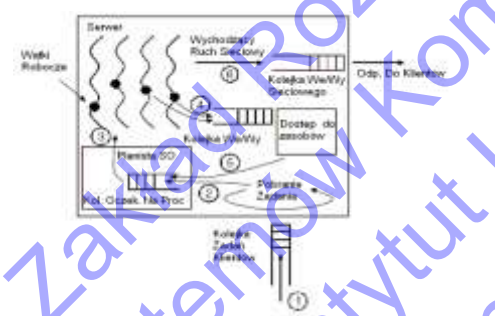


Mechanizmy poprawiające QoWS - specjalne usługi

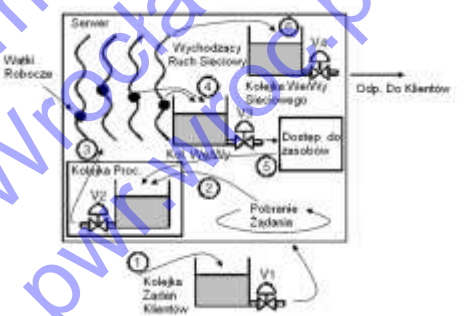


Usługi Akamai: FreeFlow i EdgeSuite

Architektura serwera WWW



Model przepływowy serwera WWW



Zarządzanie usługą WWW i rodzaje „manipulatorów”

- Przyszłość to zarządzanie usługą WWW
  - Kryteria wydajnościowe (efektywnościowe)
  - Kryteria niezawodnościowe
  - Kryteria energetyczne i ekologiczne
  - Kryteria ustawione na użytkownika końcowego lub właściciela serwisu np. biznesowe
- Sterowanie przyjęciem żądań na wejściu systemu
- Sterowanie wykonaniem żądań na procesorze
- Sterowanie wykonaniem żądań na serwerach zaplecza

Sterowanie dostępem, szeregowanie żądaniami w serwerze WWW

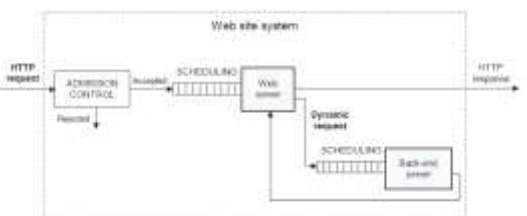


Fig. 1.4 A general model of request service in a B2C Web site system

Dystrybucja żądań HTTP

Klastry vs. rozproszenie

- Stosowanie grupy serwerów do obsługi jednego serwisu webowego jest obecnie najbardziej zaawansowaną technologicznie metodą zwiększania wydajności serwisu
- Trzy podstawowe rozwiązania:
  - Serwery **rozmieszczone globalnie** w różnych rejonach geograficznych. Żądania klienckie są kierowane najczęściej do najbliższego ośrodka webowego.
  - Serwery **rozmieszczone lokalnie** (skupione geograficznie), pracujące w ramach jednej sieci lokalnej i tworzące **klaster webowy**.
  - Trzecie rozwiązanie łączy dwa poprzednie i polega na **rozmieszczeniu globalnym klastrów serwerów**. Rozwiązanie to przeznaczone jest dla bardzo popularnych serwisów o charakterze globalnym, których klienci pochodzą z różnych rejonów świata.
- Podział na powyższe klasy rozwiązań dokonany jest na podstawie sposobu, w jaki klient (a właściwie jego przeglądarka) widzi grupę serwerów.
- W każdej z wymienionych klas rozwiązań wykorzystywane są zupełnie inne mechanizmy przekierowania żądań HTTP i algorytmy dystrybucji żądań.

Rozproszony system webowy

- **Rozproszony system webowy** (ang. *Distributed Web System*) - adresy IP poszczególnych serwerów WWW pracujących w grupie są widoczne dla klientów.
- Architektura rozproszonych systemów webowych jest starszym rozwiązaniem; w szczególności była i jest bardzo często stosowana jej odmiana wykorzystująca system serwerów DNS do przekierowania żądań. Rozwiązania używane dla rozproszonych systemów webowych nierzadko znajdują swoje zastosowania w systemach globalnie rozmieszczonych serwerów WWW.

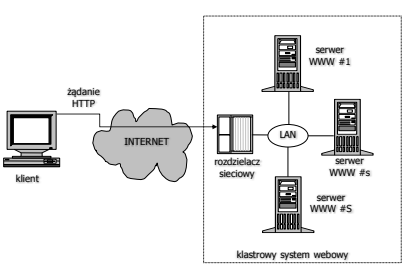
Klastrowy system webowy

- **Klastrowy system webowy** (ang. *Cluster-based Web System*) (w skrócie **klaster serwerów**) - grupa serwerów pracująca jako jedna witryna.
- Mimo, że klaster może być zbudowany z dziesiątków serwerów WWW znajdujących się w jednej lokacji, i połączonych razem poprzez sieć o wysokiej przepustowości, to posiada on pojedynczy zewnętrzny adres URL (np. www.site.pl), do którego przypisany jest dokładnie jeden adres IP (np. 156.17.24.5). Jest to zazwyczaj adres specjalnego urządzenia sieciowego, które dystrybuuje żądania na poszczególne serwery.
- Rozdzielacz sieciowy (dystrybutor sieciowy, switch/przełącznik webowy) może być specjalizowanym urządzeniem lub programem.
- Innym terminem oznaczającym klaster webowy jest *farma serwerów webowych* (ang. *web farm*).

Klastrowy system webowy

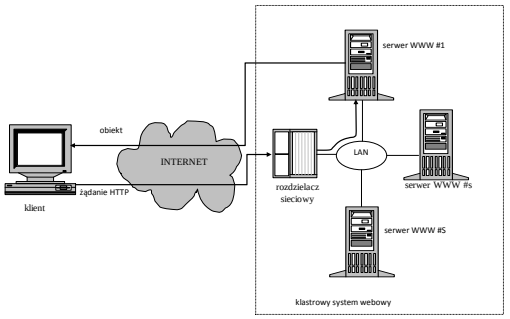
- Architektury klastrowych systemów webowych są młodszym rozwiązaniem (w stosunku do rozproszonego systemu webowego), w którym trasowanie żądań użytkownika odbywa się w całości w ramach klastra.
- Klastry serwerów są obecnie preferowane dla systemów lokalnie rozmieszczonych, ponieważ umożliwiają podział zadań z dowolną granulacją, lepszą dostępność oraz wyższy poziom bezpieczeństwa.

Architektura klastrowego systemu webowego

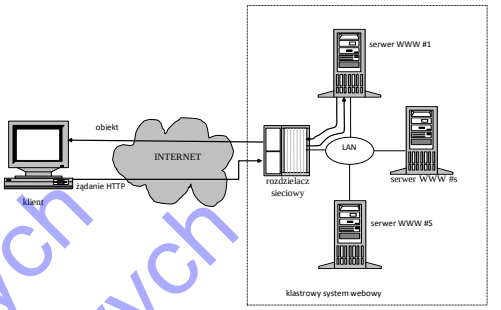


Uwaga: brak drogi powrotnej odpowiedzi na żądania

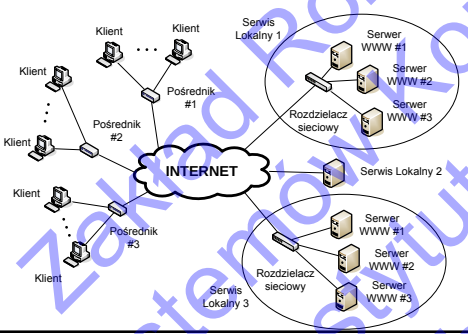
Architektura jednokierunkowa



Architektura dwukierunkowa



System globalnej dystrybucji treści z serwerami pośredniczącymi



Główne komponenty wieloserwerowych systemów webowych

- ***Mechanizm przekierowania*** żądań użytkowników. Mechanizm ten umożliwia dostarczanie żądań użytkownika do serwera WWW oraz wysyłanie odpowiedzi z serwera do klienta. Od przyjętego *mechanizmu przekierowania żądań* zależy architektura połączeń stosowana w klastrze.
- ***Algorytm dystrybucji*** żądań. Algorytm decyduje, do którego serwera przekierowane zostanie żądanie użytkownika.
- ***Egzekutor***. Jest odpowiedzialny za przesłanie żądania użytkownika do wybranego przez *algorytm dystrybucji* żądań serwera zgodnie z *mechanizmem przekierowania* żądań. Często też egzekutor ma za zadanie przesłanie odpowiedzi z serwera do użytkownika. Inne nazwy: *rozdzielacz sieciowy*, *dystrybutor* , *przełącznik webowy* (ang. *web switch*).

Mechanizmy przekierowania

- Lokalizacja mechanizmu
  - wykorzystanie DNS
  - wykorzystanie serwerów WWW
  - wykorzystanie dedykowanego urządzenia
  - złożenie modeli dystrybucji
- Podział rozwiązań ze względu na:
  - umiejscowienie mechanizmu
  - rozmieszczenie serwerów webowych
  - poziom szczegółowości
  - poziom kontroli żądań
  - strategię rozmieszczenie
  - poziom zaangażowania egzekutora
  - liczbę poziomów przekierowania

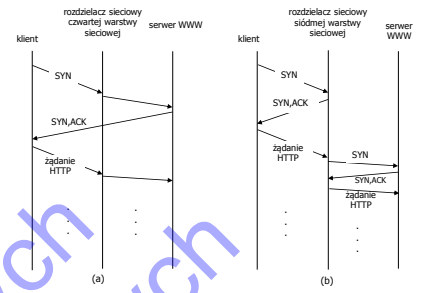
Przekierowanie podczas dostępu do serwera WWW

301 (Przeniesiono na stałe)	**Żądana strona została na stałe przeniesiona do innej lokalizacji. Gdy serwer zwraca tę odpowiedź (na żądanie GET lub HEAD), automatycznie przekieruje żądającego do nowej lokalizacji.**
302 (Tymczasowo przeniesiono)	Serwer aktualnie odpowiada na żądanie przy użyciu strony z innej lokalizacji, ale w przyszłości należy nadal żądać użycia oryginalnej lokalizacji.
303 (Sprawdź inną lokalizację)	Serwer zwraca ten kod wówczas, gdy żądający powinien w celu pobrania odpowiedzi wysłać oddzielne żądanie GET do innej lokalizacji. W przypadku wszystkich żądań innych niż HEAD serwer automatycznie przekieruje do innej lokalizacji.
307 (Tymczasowe przekierowanie)	Serwer aktualnie odpowiada na żądanie przy użyciu strony z innej lokalizacji, ale w przyszłości należy nadal żądać użycia oryginalnej lokalizacji. Ten kod jest podobny do kodu 301 pod tym względem, że na żądanie GET lub HEAD automatycznie przekieruje żądającego do innej lokalizacji.

Mechanizmy przekierowania żądań w klastrowych systemach webowych

- Jest wiele różnych mechanizmów przekierowania żądań.
- Możliwości zastosowania konkretnych rozwiązań zależą od budowy i działania rozdzielacza sieciowego.
- Podstawowa klasyfikacja wyróżnia rozdzielacze w zależności od warstwy stosu protokołowego ISO/OSI, na której rozdzielacz realizuje funkcję przekierowania przychodzących od klienta pakietów z żądaniami HTTP.
- Lokalizacja mechanizmów przekierowania żądań: warstwa 4 (L4) lub warstwa 7 (L7)

Sposób nawiązywania połączenia dla (a) rozdzielaczy czwartej i (b) siódmej warstwy



Mechanizmy przekierowania żądań w klastrowych systemach webowych

Rozdzielacze sieciowe czwartej warstwy sieciowej.

- Przeprowadzają przekierowanie „ślepe" na treść żądania użytkownika.
- Rozdzielacz podejmuje decyzje o przekierowaniu żądania w momencie, gdy klient prosi o nawiązanie połączenia z serwisem poprzez wysłanie pierwszego pakietu TCP SYN.
- Pakiety klienta nigdy nie są przesyłane do siódmej, tj. warstwy aplikacyjnej rozdzielacza.

Mechanizmy przekierowania żądań w klastrowych systemach webowych

Rozdzielacze sieciowe siódmej warstwy sieciowej.

- Podejmują decyzje o przekierowaniu na podstawie nagłówka HTTP żądania.
- W rozwiązaniu tym klient nawiązuje połączenie TCP z rozdzielaczem, rozdzielacz analizuje żądanie HTTP i przekierowuje żądanie do docelowego serwera.
- Taki mechanizm przekierowania żądań jest mniej wydajny od poprzedniego, lecz pozwala na zastosowanie bardzie wyszukanych algorytmów dystrybucji żądań.
- Inaczej: **URL routing/dispatching**

Mechanizmy przekierowania żądań w klastrowych systemach webowych

- Wybór mechanizmu przekierowania żądań ma zasadniczy wpływ na wybór algorytmu dystrybucji żądań, ponieważ wymagane informacje w obu rodzajach rozdzielaczy są różne.
- Architektury połączeń systemów klastrowych wykorzystujących rozdzielacze czwartej i siódmej warstwy sieciowej mogą być klasyfikowane w oparciu o przepływ danych pomiędzy klientem, a serwerem webowym, a zwłaszcza drogą powrotną danych od serwera do klienta.
- We wszystkich rozwiązaniach pakiety od klienta muszą przejść przez rozdzielacz sieciowy, natomiast serwer WWW może wysłać pakiety TCP z odpowiedzią bezpośrednio do klienta, gdy klaster pracuje w architekturze jednokierunkowej lub za pośrednictwem rozdzielacza sieciowego, gdy klaster pracuje w architekturze dwukierunkowej.

Rozdzielacze warstwy czwartej

- **Architektura dwukierunkowa.** Wszystkie serwery w klastrze posiadają własne adresy IP natomiast pakiety przychodzące od klienta, jak i wychodzące z serwisu przechodzą przez rozdzielacz i są w nim przepisywane.
- **Architektura jednokierunkowa.** Pakiety idące od klienta przechodzą przez rozdzielacz, natomiast pakiety wychodzące kierowane są bezpośrednio z serwera WWW do klienta. Technika ta wymaga stosowania wysoko przepustowej sieci dla pakietów wychodzących.
- Istnieją następujące mechanizmy umożliwiające wykorzystanie architektury jednokierunkowej:
  - **Packet single-rewriting**
  - **Packet tunneling**
  - **Packet forwarding**

Rozdzielacze warstwy czwartej (4L)

- Rozdzielacze 4-Layer pracują na poziomie warstwy 4 OSI.
- Aby możliwe było prawidłowe przekierowanie pakietów pochodzących od jednego klienta z jednego połączenia TCP do tego samego serwera WWW, rozdzielacz sieciowy wykorzystuje *tablice połączeń* (ang. *binding table*).
- Gdy do rozdzielacza przychodzi pakiet TCP od klienta, analizowany jest nagłówek pakietu.
  - Jeśli pakiet należy do już istniejącego połączenia, to przesyłany jest do serwera WWW zgodnie z tablicą połączeń.
  - Jeśli pakiet ma dopiero utworzyć połączenie (jest pakietem typu SYN), to wybierany jest najpierw docelowy serwer WWW zgodnie z algorytmem dystrybucji żądań, następnie dodawany jest zapis do tablicy połączeń i pakiet przesyłany jest do serwera.
- Zazwyczaj w każdym wierszu tablicy połączeń zawarte są następujące dane: adres IP klienta, port TCP klienta, adres IP serwera WWW, port TCP rozdzielacza w połączeniu z serwerem.

241

Rozdzielacze warstwy czwartej

Wady i zalety.

- W architekturze dwukierunkowej wymagane jest przepisywanie pakietów przychodzących i o wiele większej liczby pakietów wychodzących, przez co rozdzielacz może łatwo stać się „wąskim gardłem".
- Architektura ta nie wymaga jednak żadnej modyfikacji w oprogramowaniu serwerów.
- Zazwyczaj architektury jednokierunkowe są bardziej złożone, ale zarazem umożliwiają osiągnięcie wyższej wydajność klastra, ponieważ przez rozdzielacz sieciowy przechodzą jedynie pakiety przychodzące od klienta.
- Architektury dwukierunkowe natomiast są łatwiejsze do zaimplementowania, lecz rozdzielacz sieciowy może stać się „wąskim gardłem" serwisu ze względu na to, że przetwarza pakiety przychodzące jak i wychodzące z serwisu.

242

Rozdzielacze warstwy czwartej

- W rozwiązaniu **Packet single-rewriting** pakiety idące do klienta muszą być przepisywane przez serwery WWW, co powoduje dodatkowe ich obciążenie i wymaga modyfikacji stosu protokołowego serwerów.
- Tunelowanie pakietów (**Packet tunneling**) wymaga, aby serwery miały zaimplementowany mechanizm tunelowania, co nie jest jeszcze standardem w systemach operacyjnych.
- Z kolei **Packet forwarding** ograniczone jest jedynie do jednej sieci lokalnej.

243

Rozdzielacze warstwy siódmej

- Rozdzielacze warstwy siódmej pracują w warstwie aplikacji, dzięki czemu decyzja o przekierowaniu żądania użytkownika może być podjęta w oparciu o zawartość żądania.
- W rozwiązaniu tym klient nawiązuje połączenie TCP z rozdzielaczem sieciowym, następnie przesyła żądanie HTTP. Wówczas rozdzielacz wybiera docelowy serwer WWW (w oparciu o żądanie klienta) i nawiązuje z nim połączenie.
- W rozdzielaczach czwartej warstwy połączenie z docelowym serwerem nawiązywane było zaraz po przesłaniu przez klienta pakietu SYN inicjalizującego połączenie.

244

Rozdzielacze warstwy siódmej

- Podobnie jak przy rozdzielaczach warstwy czwartej istnieje kilka rozwiązań umożliwiających przekierowanie pakietów w klastrach stosujących rozdzielacze warstwy siódmej.
- Dalsza klasyfikacja związana jest z mechanizmami przekierowania pakietów od serwera do klienta.
- Mamy architekturę dwukierunkową bądź jednokierunkową.

245

Rozdzielacze warstwy siódmej

Architektura dwukierunkowa. W architekturze dwukierunkowej pakiety przychodzące do serwisu jak i wychodzące przechodzą przez rozdzielacz sieciowy.

- **TCP gateway**. Kiedy klient nawiązuje połączenie z rozdzielaczem, ten podejmuje decyzję o przekierowaniu połączenia, następnie nawiązuje drugie połączenie tym razem z serwerem i wysyła dane przekazane przez klienta. Dane z serwera trafiają do rozdzielacza i są przesyłane do klienta wcześniej nawiązanym z nim połączeniem. Dane przychodzące i wychodzące z serwisu przechodzą przez warstwę aplikacji rozdzielacza.
- **TCP splicing**. W rozwiązaniu tym dane napływające od klienta do serwisu przechodzą przez warstwę aplikacji rozdzielacza i trafiają przez drugie połączenie do serwera. Pakiety idące z serwera do klienta przechodzą przez warstwę czwartą rozdzielacza, gdzie są przepisywane (zmieniany jest adres źródłowy i docelowy pakietu).

246

Rozdzielacze warstwy siódmej

Architektura jednokierunkowa. W tej architekturze dane z serwera przekazywane są bezpośrednio do klienta.

- **TCP handoff**. Po nawiązaniu połączenia klienta z rozdzielaczem i wybraniu serwera WWW rozdzielacz przekazuje połączenie serwerowi. Wszystkie pakiety płynące od klienta przechodzą przez warstwę aplikacji rozdzielacza i są wysyłane do serwera. Z serwera dostarczane są one bezpośrednio do klienta. W rozwiązaniu tym możliwe jest przekazywanie tego samego stałego połączenia HTTP w wersji 1.1 z rozdzielacza do różnych serwerów WWW dla kolejnych żądań użytkownika. Rozwiązanie to wymaga modyfikacji stosu protokołowego, zarówno rozdzielacza jak i serwerów WWW.
- **TCP connection hop**. Rozwiązanie to zostało opracowane i oprogramowane przez firmę Resonate.

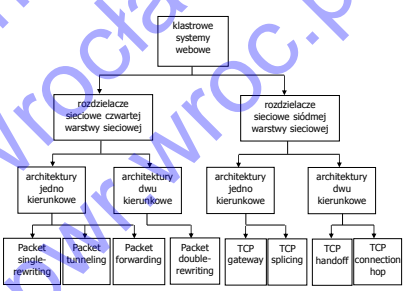
Rozdzielacze warstwy siódmej

- Architektury dwukierunkowe są łatwe do zaimplementowania, ponieważ wymagają jedynie odpowiedniej konstrukcji rozdzielacza, który jednak może stać się „wąskim gardłem" klastra.
- Oba rodzaje architektur jednokierunkowych wymagają natomiast modyfikacji w stosach protokołowych serwerów. Ponieważ wykorzystywane są różne systemy operacyjne serwerów, architektura jednokierunkowa stosowana jest w jednym tylko produkcie sprzętowym znajdującym się na rynku i jest nim *Central Dispatch* firmy *Resonate*.

Rozdzielacze 4-Layer vs. 7-Layer

- Rozdzielacze warstwy czwartej umożliwiają stosowanie większej liczby serwerów w klastrze oraz dają dużo większą przepustowość klastra.
- Rozdzielacze warstwy siódmej umożliwiają stosowanie algorytmów dystrybucji żądań wykorzystujących informacje zawartą w nagłówku HTTP, co w znaczny sposób może podnieść jakość pracy klastra.

Taksonomia mechanizmów przekierowania żądań w klastrach



Algorytmy dystrybucji żądań w klastrach webowych

- Omówione zostaną jedynie algorytmy, które stosowane są w komercyjnych lub prototypowych rozdzielaczach sieciowych.
- W klastrowych systemach webowych za politykę dystrybucji żądań odpowiedzialny jest rozdzielacz sieciowy.
- Algorytmy dystrybucji żądań mogą mieć wiele celów działania.
- Dwa z nich najczęściej spotykane, to:
  - równoważenie obciążeń serwerów (ang. load balancing) tak, aby wszystkie serwery były jednakowo obciążone, oraz
  - dzielenie obciążenia (ang. load sharing), które polega na takim rozdziale żądań, aby żaden z serwerów nie był przeciążony.

Algorytmy dystrybucji żądań w klastrach webowych

- Algorytmy dystrybucji żądań powinny mieć cechy umożliwiające sprawną i wydajną pracę serwisu. Powinny one spełniać postulaty:
  - **małej złożoności obliczeniowej.** Decyzje o przekierowaniu żądania muszą być podjęte w czasie rzeczywistym - algorytm powinien mieć czas podejmowania decyzji znacznie mniejszy od czasu obsługi żądania przez serwer.
  - **kompatybilności.** Musi być zachowana zgodność proponowanego rozwiązania ze standardami panującymi w Internecie, a w szczególności w WWW.
  - **lokalność informacji.** Wszystkie informacje potrzebne do podjęcia decyzji o przekierowaniu żądania powinny być dostępne w rozdzielaczu lub w jednym z elementów klastra.

Algorytmy dystrybucji żądań w klastrach webowych

- Podobnie jak mechanizmy przekierowania żądań, algorytmy dystrybucji żądań dzielimy na a. warstwy czwartej oraz a. warstwy siódmej.
- W pierwszej grupie algorytmów decyzja o przekierowaniu żądania użytkownika nie jest podejmowania w oparciu o informację zawartą w żądaniu.
- W grupie algorytmów dystrybucji żądań siódmej warstwy decyzja jest częściowo lub całkowicie oparta o zawartość nagłówka HTTP wysłanego przez użytkownika.

Algorytmy dystrybucji żądań w klastrach webowych

- Algorytmy mogą być statyczne, dynamiczne i adaptacyjne.
- **Algorytmy statyczne** nie wykorzystują w swym działaniu żadnych informacji o stanie serwerów ani o żądaniu użytkownika. Dzięki temu możliwe jest bardzo szybkie podjęcie decyzji.
- W **algorytmach dynamicznych** do podjęcia decyzji mogą być wykorzystane różne informacje o stanie obciążenia serwerów lub informacje o obiekcie, który użytkownik chciałby pobrać. W algorytmach tych decyzja o przekierowaniu żądania podejmowana jest wolniej, lecz jakość tej decyzji jest wyższa niż w poprzednim rozwiązaniu.
- W **algorytmach adaptacyjnych** polityka dystrybucji żądań może się zmieniać w zależności od obciążenia serwerów lub żądań użytkowników.

Algorytmy dystrybucji żądań w klastrach webowych

- Algorytmy dynamiczne mogą być klasyfikowane według pochodzenia informacji, na podstawie której podejmowana jest decyzja. Wyróżnia się następujące trzy klasy algorytmów:
  - **Algorytmy posiadające informacje o kliencie**. Decyzja o przekierowaniu żądania podejmowana jest na podstawie informacji o kliencie.
    - **Rozdzielacze warstwy czwartej** posiadają jedynie informacje o adresie IP oraz porcie TCP klienta.
    - **Rozdzielacze siódmej warstwy** mogą oprócz informacji o adresie i porcie klienta korzystać z danych zawartych w nagłówku HTTP oraz informacji, które można wywnioskować z nagłówka np. czy żądany obiekt jest statyczny czy dynamiczny.
  - **Algorytmy posiadające informacje o serwerze**. Rozdzielacz podejmuje decyzje posiadając informacje o aktualnym i przeszłym obciążeniu serwera.
  - **Algorytmy posiadające informacje o kliencie i serwerze**. Decyzja podejmowana jest w oparciu o informacje pochodzące od serwera i klienta. Obecnie większość algorytmów posiadających informacje o kliencie należy do tej grupy, ponieważ rozdzielacze zazwyczaj sprawdzają dostępność serwerów pracujących w klastrze.

Algorytmy dystrybucji żądań warstwy czwartej - *Algorytmy statyczne*

- **Random**. Algorytm rozdziela żądania równomiernie na wszystkie serwery z równym prawdopodobieństwem wyboru każdego z nich.
- **Round-Robin** (RR). W algorytmie tworzona jest lista serwerów oraz wskaźnik do ostatnio wybranego serwera np. *sn*. Każde kolejne żądanie przydzielane jest następnemu serwerowi z listy *sn+1*.
- **Static Weighted Round-Robin** (WRR). Algorytm ten wraz z algorytmem RR jest najczęściej stosowanym algorytmem w rozwiązaniach praktycznych. Swą popularność zyskuje dzięki dużej prostocie i wystarczającym w większości rozwiązań możliwościom równoważenia obciążeń. W algorytmie WRR poszczególne serwery mają przypisane wagi wyrażone w procentach, które określają procent liczby żądań, które powinny być przekierowane do poszczególnych serwerów.

Algorytmy dystrybucji żądań warstwy czwartej - *Algorytmy dynamiczne*

Algorytmy posiadające informacje o kliencie; Algorytmem należącym do tej klasy jest ***Client partition***. W algorytmie tym klienci są rozróżniani po adresie IP oraz porcie TCP. Dzięki temu możliwe jest przydzielanie tym samym klientom tych samych serwerów WWW, przez co możliwe jest często kontynuowanie transakcji np. zakupów w Internetowym sklepie.

- Rozwiązanie to ma jednak bardzo dużą wadę, ponieważ nie jest możliwe dokładne określenie po adresie IP i porcie TCP tożsamości klienta. Często zdarza się, że klienci korzystają z serwerów Proxy lub pracują za ścianą ogniową. W takich przypadkach z tego samego adresu IP może korzystać wielu użytkowników i określanie ich tożsamości lub pogrupowanie jest niemożliwe.

Algorytmy dystrybucji żądań warstwy czwartej - *Algorytmy dynamiczne*

Algorytmy posiadające informacje o serwerze

- O ile informacje o kliencie są dostępne w chwili, gdy przychodzi żądanie użytkownika, o tyle informacje o stanie serwerów są trudniejsze do osiągnięcia.
- Jest wiele problemów związanych z informacjami o serwerach, np. jakie miary obciążenia serwerów są potrzebne, jak często przesyłać informacje z serwerów na rozdzielacz, w jaki sposób przesyłać informacje, czy informacja przesłana do rozdzielacza jest aktualna.
- Miary obciążenia serwerów mogą być: miarami wejściowymi (ang. input indexes), miarami serwera (ang. server indexes), miarami ekspediowanymi (ang. forward indexes).

Algorytmy dystrybucji żądań warstwy czwartej - *Algorytmy dynamiczne*

Algorytmy posiadające informacje o serwerze

- Miary wejściowe są obliczane przez rozdzielacz i nie wymagają przesyłania jakichkolwiek informacji z serwera. Często wykorzystywaną miarą tego typu jest **czas obsługi żądania**, który może być zmierzony przez egzekutor lub też **liczba aktywnych połączeń** z danym serwerem.
- Miary serwera są przesyłane z serwera na rozdzielacz i wymagane jest, aby wysyłane dane były zbierane na serwerze periodycznie przez wyspecjalizowanie oprogramowanie. Często stosowanymi miarami serwera są: obciążenie procesora, obciążenie dysku, obciążenie karty sieciowej, wielkość dostępnej pamięci operacyjnej, liczba procesów na serwerze, itd.
- Miary ekspediowane uzyskiwane są poprzez wysłanie z rozdzielacza na serwer próbnych żądań HTTP. Miary te są często wykorzystywane dla klastrów używających architekturę jednokierunkową.

259

Algorytmy dystrybucji żądań warstwy czwartej - *Algorytmy dynamiczne*

- Algorytmy wykorzystujące informacje o stanie serwera:
  - **Least Loaded**. Wybierany jest zawsze serwer, dla którego wartość określonej miary obciążenia serwera jest najmniejsza.
  - **Weighted Round-Robin (WRR)**. Algorytm ten, podobnie jak algorytm static WRR, wybiera serwery ze statycznie utworzonej listy z prawdopodobieństwem wyboru danego serwera równym wadze przypisanej serwerowi. Wagi w algorytmie WRR zmieniają się dynamicznie w zależności od obciążenia poszczególnych serwerów.
- **Algorytmy posiadające informacje o kliencie i serwerze**; Algorytm należący do tej klasy nazywany **Client affinity** łączy w sobie cechy algorytmu Client partition oraz algorytmu posiadającego informacje o serwerze. Dzięki temu nowi klienci przekierowani są do serwerów najmniej obciążonych, natomiast klienci, którzy pracują w serwisie od jakiegoś czasu przekierowani są do serwerów, na których już pracowali i przeprowadzali transakcje.

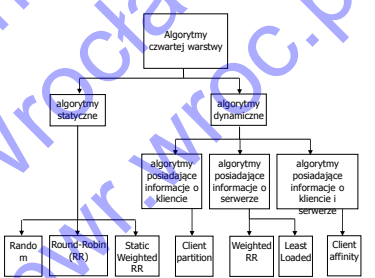
260

Algorytmy dystrybucji żądań czwartej warstwy

- W rozwiązaniach wykorzystujących rozdzielacze czwartej warstwy algorytmy statyczne są najszybszymi i najmniej obciążającymi rozdzielacz algorytmami doboru serwera.
- Algorytmy te w zadowalającym stopniu umożliwiają równoważenie obciążeń na serwerach.
- Algorytmy dynamiczne umożliwiają uzyskanie jeszcze większej wydajności klastra serwerów oraz w prymitywny sposób wspomagają stałe połączenia klienta z serwerem.
- Stosując algorytmy dynamiczne należy pamiętać o pewnych trudnościach związanych z wyborem miar obciążenia serwerów, sposobem przekazywania wartości tych miar do rozdzielacza oraz doborem częstości ich przesyłania.
- Jak prezentują to badania od wyboru opisywanych wartości w zasadniczy sposób zależy wydajność klastra serwerów.

261

Taksonomia algorytmów warstwy czwartej



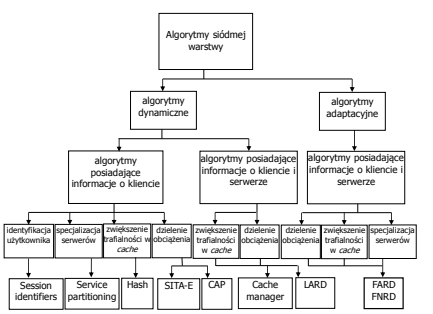
262

Algorytmy dystrybucji żądań warstwy siódmej

- Są klasyfikowane w pierwszym rzędzie na dynamiczne oraz adaptacyjne. Następnie klasyfikacja dokonywana jest według pochodzenia informacji na podstawie, której podejmowana jest decyzja o przekierowaniu (np. od klienta i z serwera). Dalszy podział związany jest z przeznaczeniem poszczególnych algorytmów.
- Proponowane są następujące cele działania algorytmów:
  - **Dzielenie obciążenia**. Celem jest wygładzenie przelotnych szczytowych przeciążeń serwerów.
  - **Specjalizacja serwerów**. Celem jest wykorzystanie cech i różnic jakości pracy serwerów, które umożliwia minimalizację czasu obsługi żądania.
  - **Zwiększenie trafialności** w cache. Celem jest zwiększenie prawdopodobieństwa występowania obiektu w pamięci serwera, na który jest przekierowane żądanie.
  - **Identyfikacja użytkownika**. Celem jest precyzyjna identyfikacja użytkowników.

263

Taksonomia algorytmów warstwy siódmej



264

Algorytmy dynamiczne posiadające informacje o kliencie

Session identifiers. Poprzez identyfikację sesji klienta rozdzielacz może połączyć klienta z odpowiednim serwerem WWW. Który to ma być serwer zależy od przyjętej strategii. Najczęściej są stosowane strategie:

- **Cookie-Based Scheduling.** Poprzez stosowanie informacji, która może być dodana do nagłówka HTTP w cookie, możliwa jest jednoznaczna identyfikacja użytkowników, a co za tym idzie podzielenie ich na klasy.
- **Persistence.** Poprzez stosowanie cookie (wtedy algorytm przyjmuje nazwę Cookie-based persistence) lub nie zaszyfrowanego identyfikatora SSL (wtedy algorytm przyjmuje nazwę SSL persistence) możliwa jest jednoznaczna identyfikacja użytkownika. Identyfikacja ta może być użyteczna, gdy klient wykonuje bardziej złożone operacje w serwisie WWW i konieczne jest, aby łączył się zawsze z tym samym serwerem.

Algorytmy dynamiczne posiadające informacje o kliencie

- **Service partitioning**. Rozdzielacz odczytuje nagłówek HTTP, który otrzymał od klienta, sprawdza URL żądanego obiektu i klasyfikuje obiekty np. na podstawie rozszerzenia pliku. Następnie wysyła żądanie do wyspecjalizowanego serwera. Wszystkie serwery WWW obsługują konkretne typy obiektów np. obiekty statyczne, dynamiczne, multimedialne itp.
- **Hash**. Podział wszystkich obiektów na serwery odbywa się poprzez funkcje mieszające. Rozdzielacz posiadając informacje o URL pobieranego obiektu lub też o jego wielkości, podejmuje decyzje o przydzieleniu go do konkretnego serwera WWW na podstawie funkcji mieszających. Stosowanie tej strategii powoduje zmniejszenie liczby plików pobieranych z konkretnego serwera, a co za tym idzie, zwiększenie trafialności występowania danego obiektu w pamięci cache serwera. Algorytm przeznaczony jest dla serwisów obsługujących w głównej mierze obiekty statyczne.

Algorytmy dynamiczne posiadające informacje o kliencie

- **SITA-E**. Żądania użytkowników rozdzielane są na podstawie wielkości żądanych obiektów statycznych.
- Do każdego serwera pracującego w klastrze przyporządkowany jest pewien zakres wielkości obiektów.
- Kiedy przychodzi żądanie, rozdzielacz określa wielkość żądanego obiektu na podstawie tabeli, w której każdy wiersz zawiera informacje o nazwie obiektu i jego wielkości.
- Następnie ustalane jest, do którego z serwerów przyporządkowany jest zakres wielkości obiektów, do których należy żądany obiekt.

Algorytmy dynamiczne posiadające informacje o kliencie

- **CAP** (Client-Aware Policy). Przyjmuje się, że obiekty w serwisie można podzielić na następujące klasy: statyczne i lekko dynamiczne publikacje WWW, serwisy obciążające dysk, serwisy obciążające procesor oraz serwisy obciążające dysk i procesor. Rozdzielacz przełącza cyklicznie żądanie z każdej z klas do innego serwera WWW tak, aby żaden serwer nie był przeciążony jednym rodzajem żądań, a co za tym idzie, nie miał przeciążonego żadnego ze swoich zasobów (procesora i dysku). W tej strategii każdy z serwerów WWW może obsłużyć każde z żądań. Wszystkie serwery otrzymują podobną liczbę żądań należących do tej samej klasy.
- Wadą CAP jest to, że przed uruchomieniem klastra serwerów konieczne jest dokładne określenie, które obiekty są statyczne lub dynamiczne i w jakim stopniu obsługa, każdego z obiektów obciąża poszczególne elementy serwerów.

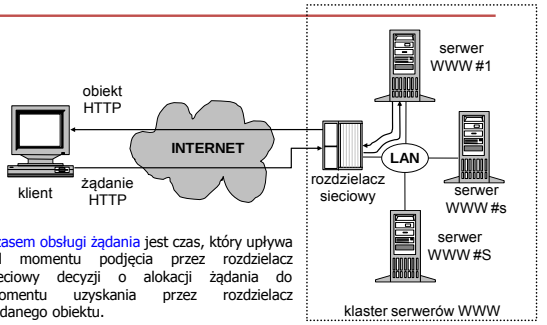
Algorytmy dynamiczne posiadające informacje o kliencie i serwerze

- **Cache manager.** Specjalny program nazywany zarządcą pamięci podręcznej posiada informacje o zawartości pamięci podręcznej poszczególnych serwerów.
- Kiedy przychodzi żądanie użytkownika, które dotyczy obiektu jeszcze nie znajdującego się w pamięci cache żadnego z serwerów, to wybierany jest serwer najmniej obciążony.
- Gdy przychodzi żądanie dotyczące obiektu znajdującego się w pamięci podręcznej serwerów, to wybierany jest ten serwer spośród serwerów posiadających obiekt w pamięci, który jest najmniej obciążony.
- Jeśli jednak obciążenie wybranego serwera jest większe niż ε procent w stosunku do najmniej obciążonego serwera w klastrze, to wybierany jest serwer najmniej obciążony w całym klastrze.

Algorytmy dynamiczne posiadające informacje o kliencie i serwerze

- **LARD** (Locality-Aware Request Distribution). Rozdzielacz podejmuje decyzje o przekierowaniu żądania na podstawie informacji od klienta o żądanym obiekcie oraz informacji o obciążeniu serwerów WWW.
- Jako obciążenie serwera brana jest pod uwagę liczba aktywnych połączeń TCP na danym serwerze.
- Kroki algorytmu:
  - Jeśli obiekt pobierany jest po raz pierwszy, to przekieruj go na najmniej obciążony serwer.
  - Jeśli obiekt pobierany jest po raz kolejny, to przekieruj go na serwer, z którego był już pobierany, o ile serwer nie jest przeciążony, tzn. nie ma zbyt dużej różnicy między obciążeniem tego serwera a innymi.
  - Jeśli obiekt pobierany jest po raz kolejny i serwer, z którego był już pobierany, jest przeciążony, to przekieruj go na najmniej obciążony.
- Jak pokazują badania jest jednym z najskuteczniejszych algorytmów podnoszących wydajność klastra serwerów

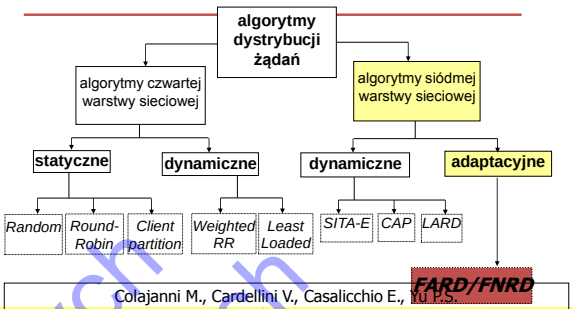
Klaster serwerów WWW – inne kryterium – czas obsługi



Czasem obsługi żądania jest czas, który upływa od momentu podjęcia przez rozdzielacz sieciowy decyzji o alokacji żądania do momentu uzyskania przez rozdzielacz żądanego obiektu.

271

Klasyfikacja algorytmów dystrybucji żądań w klastrowych systemach webowych



Colajanni M., Cardellini V., Casalicchio E., Yu P.S.
The state of the art in locally distributed Web-server systems ACM Computing Surveys, Vol. 34, No. 2, June 2002

Fuzzy **A**daptive **R**equest **D**istribution

Fuzzy **N**eural **R**equest **D**istribution