STUDIA INFORMATICA 2001
Volume 22 Number 1 (43)

Leszek BORZEMSKI
Politechnika Wrocławska, Instytut Sterowania i Techniki Systemów

POMIARY WYDAJNOŚCI KLASTRA SERWERÓW WWW Z RÓWNOWAŻENIEM OBCIĄŻENIA

Streszczenie. W pracy przedstawiono wyniki pomiarów wydajności serwerów WWW pracujących w konfiguracji lokalnego klastra komputerów IBM RS/6000 z równoważeniem obciążeń. Zapytania do serwisu WWW dystrybuowane były pomiędzy poszczególne serwery webowe w klastrze przez system równoważenia obciążeń IBM Network Dispatcher. Badania potwierdziły skuteczność algorytmów równoważenia obciążeń w uzyskiwaniu przez klaster webserwerów wysokiej wydajności przetwarzania zapytań.

PERFORMANCE MEASUREMENTS OF CLUSTER-BASED WEB SERVERS WITH LOAD BALANCING

Summary. This paper presents results from a comprehensive empirical study of Web servers running in cluster configuration with load balancing mechanisms. We also discuss how different attributes of the whole solution, i.e. HTTP protocol, WWW server design and load balancing strategy influence the system's performance. In particular, we present the results of experiments performed for the clusters of WWW Apache servers installed on IBM RS/6000 stations under AIX operating system.

1. Wstęp

Po mechanizacji, automatyzacji, komputeryzacji, informatyzacji nadeszła era internetyzacji rozumiana jako wprowadzanie informatycznych technologii internetowych do technologii przetwarzania danych. Technologie internetowe, a inaczej mówiąc technologie webowe budowane są wokół bardzo prostej, a jakże nośnej koncepcji przetwarzania sieciowego, w której wyróżniamy lekkiego uniwersalnego klienta – przeglądarką stron WWW oraz serwerem webowym usługi *www*. Dzięki szerokiej akceptacji tej technologii

przetwarzania Internet uległ przeobrażeniu w sieć WWW, a technologie webowe stały się podstawą budowy współczesnych systemów informatycznych i możemy dzisiaj mówić o następnym etapie rozwoju informatyki, a mianowicie o postępującej webyfikacji systemów informatycznych.

Posługiwanie się na szeroką skalę technologią webową spowodowało, że w Internecie, obok badań nad wydajnością i niezawodnością podsystemów transmisji danych (zarówno w sieciach dostępowych, jak i szkieletowych), pojawiła się potrzeba badań nad metodami sprawnego i wydajnego przetwarzania zapytań kierowanych przez klientów WWW do serwerów WWW. Parę lat temu, przed powstaniem usługi *www*, problematyka przetwarzania w Internecie praktycznie nie istniała, ponieważ jedynymi wykorzystywanymi usługami były usługi pocztowe *mail* czy też transferu plików *ftp*, które nie angażują w sposób zauważalny lokalnych zasobów przetwarzania. Usługi *telnet* i *rexec* ze względów bezpieczeństwa nie były szeroko wykorzystywane. Budowa systemów informatycznych z wykorzystaniem usługi *www* spowodowała naturalny wzrost zainteresowania problematyką badawczą dotyczącą organizacji przetwarzania w ośrodkach webowych, w których instalowane są serwery WWW. W szczególności, najpopularniejszym dzisiaj podejściem do tej problematyki jest zastosowanie metod i algorytmów równoważenia obciążeń (ang. *load balancing*). Problematyka równoważenia obciążenia procesorów w systemach obliczeniowych nie jest nowa i była w okresie wprowadzania usługi *www* już znacząco rozwinięta dla różnych architektur przetwarzania w systemach komputerowych (tj. SMP, maszyny równoległe i systemy rozproszone), [1-3]. Niemniej jednak, specyfika zagadnienia wynikająca z zasad funkcjonowania usługi *www* w sieci TCP/IP spowodowała, że wyodrębnił się nowy kierunek badań nad efektywnym funkcjonowaniem ośrodków webowych [7]. Tematyką tą interesuje się wielu znanych producentów, którzy opracowują konkretne rozwiązania systemowe, mające stać się niezbędnym elementem rozwiązań ośrodków webowych między innymi dla handlu elektronicznego, bankowości internetowej czy rozrywki w Internecie. Są to rozwiązania zarówno sprzętowe, jak i programowe. Szereg codziennych przykładów pokazuje, jak takie rozwiązania mogą być skuteczne utrzymując dostępność serwisu nawet w bardzo trudnych warunkach obciążenia.

Wśród polskich ośrodków badawczych problematyką projektowania i eksploatacji webowych systemów informatycznych zajęto się między innymi w Zakładzie Rozproszonych Systemów Komputerowych w Instytucie Sterowania i Techniki Systemów Politechniki Wrocławskiej. Przedmiotem zainteresowania tych prac jest w szczególności ocena wydajności serwerów usług WWW pracujących w systemie klastra komputerowego na podstawie różnych mechanizmów równoważenia obciążeń [4-6]. Z prowadzonymi

badaniami związane są również prace dydaktyczne w zakresie wykładów, projektów, seminariów i prac dyplomowych, w tym między innymi [12-14].

W niniejszym artykule przedstawimy część wyników eksperymentów, w trakcie których mierzono wydajność obsługi zapytań WWW przez ośrodek webowy organizowany w oparciu o różne konfiguracje sprzętowe i programowe, począwszy od pojedynczej maszyny, przez maszynę dwuprocesorową, a skończywszy na klastrze wielu maszyn. Eksperymenty pomiarowe realizowane były z udziałem studentów specjalności systemy sterowania na kierunku informatyka na Wydziale Informatyki i Zarządzania [12, 14].

2. Problemy wydajności usługi *www*

2.1. Protokół HTTP i jego usprawnienia

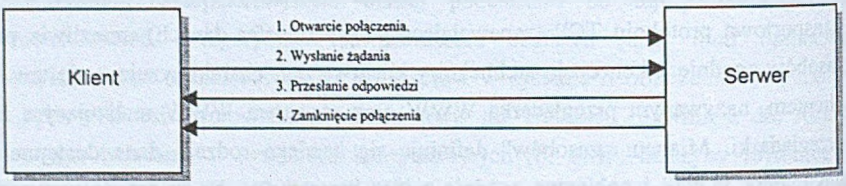
Protokół HTTP (ang. *HyperText Transfer Protocol*) jest podstawowym protokołem usługi *www*. Umiejscowiony w 7. warstwie modelu ISO/OSI i wykorzystujący 4. warstwę transportową protokołu TCP (zapewniającą pewny transfer danych) umożliwia przesłanie zasobów zgodnie z koncepcją architektury klient-serwer, charakteryzującą się tzw. cienkim klientem, nazywanym przeglądarką WWW oraz serwerem WWW realizującym zapytania przeglądarki. Mianem „zasobów" definiuje się każdego rodzaju dane dostępne poprzez dokumenty WWW i pobierane zgodnie z ideą hipertekstu. Są to przede wszystkim pliki opisu stron HTML, obrazy (np. GIF, JPG), wynik działania aplikacji po stronie serwera, skrypty uruchamiane po stronie klienta (np. Java Applet, JavaScript, ActiveX), definicje stylów (np. CSS) itd. Koncepcja hipertekstu organizacji zbioru dokumentów wykorzystuje powiązania między dokumentami. Dokument hipertekstowy zawierać może obok tekstu również grafikę, animacje, dźwięki oraz odsyłacze do innych dokumentów hipertekstowych, które mogą znajdować się na innych serwerach WWW, w konsekwencji często, aby skompletować cały dokument, przeglądarka WWW musi otworzyć wiele połączeń z kilkoma serwerami WWW.

Część zasobów dostępnych poprzez przeglądarki WWW używa innych protokołów, opartych często na odmiennych protokołach transportowych (np. UDP) – dotyczy to głównie strumieni audio i wideo (np. Real Networks, MS NetShow).

Definicja protokołu HTTP w wersji 1.0 została zapisana w dokumencie RFC 1945 [9], a w wersji 1.1 w dokumencie RFC 2616 [10]. Dokumenty te przedstawiają cechy, które muszą być zaimplementowane po stronie przeglądarki i serwera, aby transakcje HTTP v.1.0/v.1.1 mogły zostać bezbłędnie przeprowadzone.

Podstawowa zasada wymiany informacji z wykorzystaniem protokółu HTTP jest następująca. Przeglądarka otwiera połączenie i zgłasza do serwera zapotrzebowanie na określony zasób, serwer w odpowiedzi przesyła poszukiwane dane, ewentualnie zwraca status błędu, po czym zamyka połączenie. Transakcja przebiega w czterech etapach:

- **Etap 1** – klient otwiera połączenie TCP/IP; domyślnym portem docelowym jest 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.



Rys. 1. Przebieg transakcji HTTP
Fig. 1. HTTP transaction

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).

Metody HTTP określają sposób wywołania danego zasobu i reakcji serwera. Dokument RFC1945 definiuje trzy podstawowe metody wraz ze wskazaniem na możliwość użycia metod rozszerzonych. Podstawową i najpopularniejszą metodą jest GET. Serwer po otrzymaniu tego polecenia wyszuka/uruchomi i zwróci zasób do klienta, a w razie problemów odpowie właściwym kodem statusu. Odmianą tej metody jest metoda *Conditional GET*, która umożliwia *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`. Metoda HEAD jest „podmetodą"

metody GET – serwer wykonuje w tym przypadku identyczne czynności jak przy metodzie GET, z wyjątkiem przesłania treści odpowiedzi - klient otrzymuje jedynie nagłówek obiektu. W praktyce metodę tę stosuje się do rozpoznania obiektu (data utworzenia, rozmiar, typ) bez pobierania jego zawartości. Definicja metody POST jest bardzo ogólna – jej użycie 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, przy czym efekt jest taki sam jak przy przesłaniu typu GET, natomiast dane nie są umieszczane w sposób jawny w identyfikatorze zasobu. Dodatkowe metody to: PUT, DELETE, LINK oraz UNLINK. Są one rzadziej implementowane w serwerach, gdyż nie są wymagane przez specyfikację, a ponadto zaleca się unikanie ich ze względu na wymogi bezpieczeństwa (każda z tych metod ingeruje w zawartość zasobów). W protokole HTTP v1.1 wprowadzono ponadto nową metodę OPTIONS, która pozwala 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.

Protokół HTTP, jak wiele innych rozwiązań komunikacyjnych, podlega modyfikacji między innymi w celu podwyższenia wydajności oraz bezpieczeństwa komunikacji. Podstawowe modyfikacje wprowadzone w wersji 1.1 protokołu HTTP dotyczą: wykorzystania rozszerzeń protokołu, użycia pamięci podręcznej *cache,* optymalizacji wykorzystania pasma, zarządzania połączeniem, przekazu komunikatów, przechowywania adresu, informacji o błędach, bezpieczeństwa oraz negocjacji zawartości. W zakresie poprawy wydajności najważniejsze są udoskonalenia w zakresie pamięci *cache,* wykorzystania pasma oraz zarządzania połączeniem.

Użycie pamięci *cache* ma istotny wpływ na zwiększenie wydajności przesyłania dokumentów WWW. Podstawowy mechanizm wykorzystania pamięci *cache* pojawił się już w wersji 1.0 w postaci wspomnianej wcześniej metody *Conditional* GET. Wersja 1.1 wprowadza nowe udogodnienia. Przede wszystkim zrezygnowano z oznaczania momentu aktualizacji zasobu poprzez łańcuchowy zapis daty – w tym przypadku 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 *cachem* umożliwia pole `Cache-Control`; określa ono, co może być umieszczone w pamięci podręcznej i na jak długo - można również zakazać/nakazać aktualizację zawartości *cache*.

Optymalizacja wykorzystania protokołu była bardzo istotnym czynnikiem motywującym wprowadzenie zmian w protokole HTTP. Wprowadzono możliwość przesłania zasobu w częściach, niwelując tym samym skutki przerwanych połączeń i dając możliwość pobrania początku treści zasobu (np. paletę barw i rozmiar grafiki) czy też doczytywania końcówki rosnącego obiektu (np. odczyt plików logowania i statystyk). 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.

Wśród metod poprawienia wydajności w zakresie zarządzania połączeniem stworzono przede wszystkim możliwość utworzenia trwałego połączenia dla przesyłu wielu obiektów oraz ich strumieniowania. Trwałe połączenia (*persistent connections*) są zachowaniem domyślnym protokołu w wersji 1.1 – połączenie trwa aż do wystawienia dyrektywy przez klienta `Connection: close` lub do chwili upłynięcia okresu przeterminowania. HTTP w wersji 1.1 umożliwia stosowanie stałego, podzielonego na sesje, połączenia TCP do pobrania całego dokumentu. Podstawowym problemem wydajności HTTP wydaje się być jednak niedopasowanie protokołu TCP – zorientowanego na przesyłanie strumienia bajtów - i usługi WWW, zorientowanej na przesyłanie wyodrębnionych komunikatów. Okazuje się, że otwieranie zbyt wielu połączeń równoczesnych nie przynosi spodziewanych korzyści. Strumieniowanie pozwala przeglądarce wystawić wiele nowych żądań, nawet, jeśli nie nadeszło potwierdzenie dla poprzednich. Ponadto, w przekazie komunikatów wprowadzono możliwość określania wielkości udostępnianego zasobu, co polepsza warunki transmisji i buforowania zasobów.

2.2. Inne metody poprawienia wydajności serwisu WWW

Podstawowym sposobem zwiększenia wydajności serwisu WWW jest zwiększenie przepustowości przyłącza. Sprzętowa modernizacja samego serwera nie przyniesie rezultatów w przypadku, gdy „wąskim gardłem" jest przyłącze do Internetu. Jeśli jednak dysponujemy odpowiednio wydajnym przyłączem (np. ATM), „wąskim gardłem" jest na pewno serwer.

Wydajność każdego komputera zwiększyć można dokupując większą ilość pamięci RAM i szybsze dyski twarde. W przypadku serwerów WWW jest to jednak rozwiązanie prowizoryczne. Lepsze efekty przynosi instalacja maszyny wieloprocesorowej. Niestety, obecne implementacje TCP/IP nie wykorzystują wielowątkowości w stopniu, który umożliwiałby pełne wykorzystanie cech technologii wieloprocesorowych. Należy się spodziewać, że otwartych połączeń TCP będzie zazwyczaj więcej niż procesorów w

serwerze. Gdyby obsługę stosu TCP/IP podzielić można na niezależne wątki, których wykonanie przebiegać by mogło z podziałem czasu, to efektywność wieloprocesorowych serwerów WWW byłaby znacznie większa niż obecnie. Współcześnie stosowanie serwerów wieloprocesorowych nie przynosi korzyści tak dużych, jak można by się spodziewać, aczkolwiek pewne nowe architektury wieloprocesorowe, takie jak np. NUMA są szczególnie polecane przez producentów jako maszyny dla ośrodków ebiznesowych.

Dobrą metodą zwiększenia wydajności WWW jest stosowanie serwerów proxy bądź różnego rodzaju akceleratorów internetowych, które na zasadzie pamięci cache przechowują część dokumentów serwisu, uzupełniają i kopiują potencjalnie pożądane zasoby i w ten sposób częściowo wyręczają serwer główny bądź przejmują na siebie część zadań przetwarzania (np. w zakresie połączeń SSL), zwalniając tym samym zasoby serwera WWW. Podobnie obiecujące jest oddawanie przez frontowe serwery WWW zadań wymagających zaawansowanego przetwarzania do serwerów zaplecza. W takich przypadkach zadania realizowane przez serwery zaplecza mogą dotyczyć np. przeszukiwania dużych baz danych, przetwarzania w hurtowni danych bądź zaawansowanego przetwarzania numerycznego.

Najbardziej efektywną i jednocześnie najbardziej obecnie zaawansowaną technologicznie metodą zwiększenia wydajności serwisu WWW jest powielenie jego zawartości na kilka dodatkowych serwerów i udostępnienie tej grupy serwerów pod jednym wirtualnym adresem IP. Rozwiązanie takie wymaga sprzętu bądź oprogramowania, które z jednej strony ukrywałoby przed użytkownikami istnienie kilku serwerów, a z drugiej dokonywałoby dystrybucji obciążenia w celu maksymalizacji ich łącznej wydajności (kierując się kryterium zrównoważenia obciążeń poszczególnych serwerów). Obecne badania są ukierunkowane na dwie kategorie rozwiązań. Pierwsza z nich dotyczy dystrybucji obciążenia (przełączania połączeń) bazując na poziomie TCP/IP, bez wykorzystania informacji, jakie dane przeglądarka chce pobrać z serwera. Wiele podejść i algorytmów z tego zakresu zostało przeanalizowanych między innymi w pracach [4-7]. Druga grupa rozwiązań, które stają się coraz popularniejsze, polega na przełączaniu w ramach siódmej warstwy modelu ISO/OSI (ang. *Level 7 web switching)*, gdzie za podstawę przełączenia możemy przyjmować typ zasobów, o jakie ubiega się klient (szerzej na ten temat pisze K. Zatwarnicki w tym zeszycie).

Serwis WWW będzie wydajniejszy, jeśli prawidłowo zaprojektujemy naszą witrynę. W trakcie jej projektowania należy zastanowić się i przeanalizować warunki, w jakich witryna będzie wykorzystywana. Użycie narzędzi wysokiego poziomu, za pomocą których można przyspieszyć projekt, w większości przypadków prowadzi do uzyskania nadmiarowego i bardzo rozbudowanego kodu strony. Taki efekt można zaobserwować, kiedy stronę stworzoną takim wygodnym narzędziem poddamy weryfikacji i optymalizacji. Budowa strony w szczególności powinna współgrać z możliwościami przyspieszenia jej udostępniania

przy użyciu protokołu HTTP v1.1, np. poprzez podział dużych tabel znajdujących się na stronie na mniejsze obiekty. Takie techniki projektowania, a w tym powrót do korzeni, tj. wykorzystanie możliwości, jakie daje język HTML, powinny być podstawą dobrego projektu serwisu WWW.

3. Metody pomiaru wydajności serwerów WWW

Podstawową metodą oceny wydajności serwerów WWW jest stosowanie syntetycznych programów testujących, tzw. benchmarków, które generując sztuczny strumień zapytań rejestrują parametry wydajności serwera, takie jak np. ilość obsługiwanych zapytań na sekundę lub szybkość transferu. Benchmarki wykorzystują do pomiarów własną witrynę, której konstrukcja powstała specjalnie dla potrzeb takich pomiarów i uwzględnia w pewnym sensie typowe rozwiązania w tym zakresie. Oczywistą zaletą tego podejścia jest możliwość łatwego porównania wyników uzyskanych dla różnych serwerów. Podstawową wadą jest to, że najlepszy nawet benchmark tylko w przybliżeniu naśladuje zachowania rzeczywistych użytkowników, pomimo że w konstrukcji wielu benchmarków wykorzystano informacje pochodzące z rzeczywistych logów znanych i obleganych serwerów WWW. Możemy zastosować również metodę bieżącego pomiaru parametrów serwera WWW i przez ustalony okres czasu obserwować konkretny serwer pod względem wielkości transferów, których dokonał i ilości dziennie obsługiwanych zapytań. Tego typu badania mają jednak mało ogólny charakter, gdyż wiele uzyskanych wyników zależy od lokalnej specyfiki (np. charakteru publikowanych informacji).

Przy implementacji benchmarków serwerów WWW napotykamy między innymi na następujące problemy:

1. Chociaż dany jest zarówno rozkład rozmiaru plików na serwerze, jak i rozkład rozmiaru plików pobieranych z serwera, to nie jest znana zasada pozwalająca określić, ile zapytań o konkretny plik należy skierować do serwera podczas testu. Podstawą dokonania właściwego wyboru mogą być wyniki doświadczeń.
2. Przeprowadzone dotychczas badania nie wskazują jednoznacznie, w jaki sposób uszeregować kolejne odstępy pomiędzy odwołaniami. Stąd do symulacji tego zjawiska używa się rzeczywistych logów.
3. Wygenerowanie możliwie realistycznego obciążenia wymaga również znajomości częstotliwości, z jaką przeciętny użytkownik przełącza się między stronami. W rzeczywistych warunkach można zaobserwować w różnych serwisach znaczne skoki obciążenia w czasie. Jest to skutek wspomnianych przełączeń między stronami. Kiedy przeprowadzane są testy z użyciem generatorów wysyłających maksymalną możliwą ilość

zapytań (zależną od wydajności komputera klienta), obciążenie testowanych serwerów jest bardziej stabilne.

W naszych eksperymentach wykorzystany został program testujący Ziff – Davis WebBench 3.0 dostępny w Internecie [11]. Jest to kompletny pakiet pozwalający mierzyć wydajność serwerów WWW w ustalonym środowisku. Jest on prosty w obsłudze i dostarcza wielu możliwości. WebBench pozwala zdefiniować własne zestawy plików testowych oraz, co ważniejsze, przebieg i metody badania. Każde badanie może się składać z wielu etapów (*mixes*), etap definiuje liczbę uczestniczących w nim klientów, czas trwania, liczbę wątków w obrębie komputera-klienta, pauzę pomiędzy kolejnymi żądaniami, opóźnienie startu testu po otrzymaniu sygnału od kontrolera WebBench, udział żądań HTTP v1.1 (*persistent connection* – stałe połączenie dla wielu żądań oraz *pipelining* – żądanie bez oczekiwania na potwierdzenie) w ogólnej liczbie zapytań. Ponadto użytkownik ma możliwość wskazania serwera *proxy*, portu usługi WWW oraz zdefiniowania testu dla wielu serwerów równocześnie.

4. System IBM SecureWay Network Dispatcher

IBM SecureWay Network Dispatcher jest to jedno z pierwszych komercyjnych rozwiązań programowych, które pozwala zwiększyć wydajność i zapewnić ciągłą pracę serwisów WWW [8]. Znalazło ono szerokie zastosowanie w znaczących serwisach WWW, takich jak serwisy prowadzone w trakcie olimpiad w Atlancie, Nagano oraz Sydney. System jest przeznaczony do obsługi serwerów webowych zbudowanych w technologii klastrowej. Klaster jest grupą serwerów webowych obsługujących jedną witrynę WWW. Wszystkie serwery w klastrze posiadają tę samą zawartość i zapytanie może być obsłużone przez dowolny serwer w klastrze. Pojedyncze zapytanie obsługiwane jest przez jeden z aktualnie sprawnych serwerów. System może funkcjonować w sieci WAN i LAN. W eksperymentach wykorzystano konfigurację klastra lokalnego. Funkcjonowanie systemu przedstawiono na rys. 2. Dokładniejszy opis systemu zawiera między innymi praca [4].

Rozdział zleceń między serwery bazuje na wartościach wag wskazujących na możliwość potencjalnego obciążenia serwera – jeżeli np. jeden serwer ma wagę 2, a drugi ma wagę 1, to serwer o wadze 2 powinien dostać dwa razy więcej żądań niż ten o wadze 1. Wagi mogą być ustawiane ręcznie lub przez Menadżera. Najczęściej do ustawiania wag korzysta się z Menadżera, który ustawia wagi automatycznie i w sposób adaptacyjny, z uwzględnieniem aktualnych warunków pracy klastra. Menadżer może korzystać z wewnętrznych liczników systemowych oraz z informacji dostarczanych przez inne komponenty systemu, takich jak ISS (Interactive Session Support).

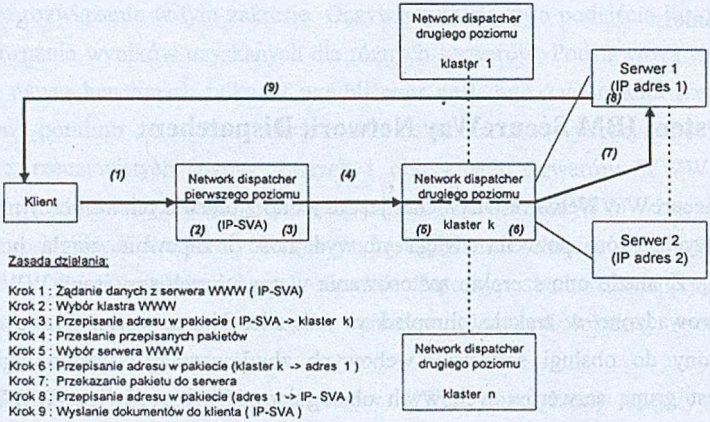
W trakcie eksperymentów przeprowadzono badania następujących konfiguracji klastra lokalnego złożonego z trzech serwerów jednoprocesorowych:

1. Konfiguracja standardowa, w której do obliczania wag serwerów wykorzystywane były tylko dane z tablicy otwartych połączeń TCP i tablicy nowych połączeń przydzielonych od ostatniego odświeżenia wag serwerów.

2. Konfiguracja, w której wagi serwerów nie były odświeżane (były stałe i równe), a nowe połączenia przydzielane były do serwerów cyklicznie.

3. Konfiguracja, w której wagi serwerów obliczane były wyłącznie na podstawie danych z systemu ISS oraz doradcy HTTP.

4. Konfiguracja, w której wagi serwerów obliczane były na podstawie wszystkich dostępnych danych, tzn. tablicy otwartych połączeń, tablicy połączeń nowo przydzielonych, danych z monitora ISS oraz doradcy HTTP.



Rys. 2. Funkcjonowanie IBM SecureWay Network Dispatcher w sieci WAN/LAN
Fig. 2. Functioning of IBM SecureWay Network Dispatcher in WAN/LAN

Monitor ISS określał obciążenie poszczególnych serwerów poprzez pomiar wykorzystania różnych zasobów, takich jak ilość wolnej pamięci, użycie procesora czy licznik procesów. Zadaniem doradców jest zbieranie informacji na temat obciążenia serwerów. Doradcy okresowo otwierają połączenia TCP i wysyłają do serwera wiadomość z żądaniem specyficznym dla danego typu doradcy. Po wysłaniu wiadomości doradcy czekają na odpowiedź. Po jej otrzymaniu szacują wartość obciążenia serwera na podstawie czasu, jaki upłynął, nim serwer zwrócił odpowiedź na żądanie i zgłaszają ten czas (w milisekundach) Menadżerowi, aby na podstawie tej i innych informacji oszacował wartość wag

poszczególnych serwerów. System posiada doradców między innymi działających w zakresie: HTTP, FTP, Telnet, NNTP, POP3, SMTP, SSL, TCP i Ping.

5. Badania wydajności serwerów WWW

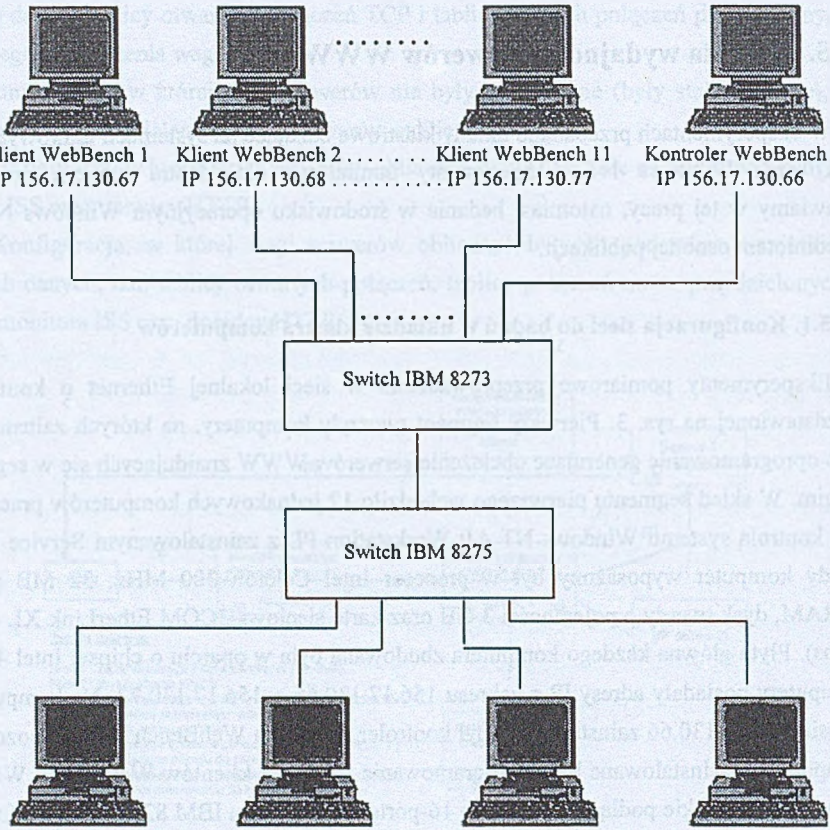
W eksperymentach przebadano układy klastrowe bazujące na systemach unixowych (IBM AIX) oraz Windows NT. Eksperymenty pomiarowe dla klastra maszyn unixowych omawiamy w tej pracy, natomiast badania w środowisku operacyjnym Windows NT będą przedmiotem osobnej publikacji.

5.1. Konfiguracja sieci do badań w układzie klastra komputerów

Eksperymenty pomiarowe przeprowadzono w sieci lokalnej Ethernet o konfiguracji przedstawionej na rys. 3. Pierwszy segment tworzyły komputery, na których zainstalowane było oprogramowanie generujące obciążenie serwerów WWW znajdujących się w segmencie drugim. W skład segmentu pierwszego wchodziło 12 jednakowych komputerów pracujących pod kontrolą systemu Windows NT 4.0 Workstation PL z zainstalowanym Service Pack 5. Każdy komputer wyposażony był w procesor Intel Celeron 300 MHz, 32 MB pamięci SDRAM, dysk twardy o pojemności 3 GB oraz kartę sieciową 3COM EtherLink XL (10/100 Mbps). Płyta główna każdego komputera zbudowana była w oparciu o chipset Intel 430 LX. Komputery posiadały adresy IP z zakresu 156.17.130.66 ÷ 156.17.130.77. Na komputerze o adresie 156.17.130.66 zainstalowany był kontroler programu WebBench 3.0. Na pozostałych komputerach zainstalowane było oprogramowanie jedenastu klientów WebBench. Wszystkie komputery klienckie podłączone były do 16-portowego switcha IBM 8273 High Performance Switch (10/100 Mbps). Do jednego z wolnych portów tego switcha przyłączony był 24-portowy switch IBM 8275 (10/100 Mbps), który stanowił podstawę budowy drugiego segmentu opisywanej sieci.

Do switcha 8275 podłączane były komputery klastra z zainstalowanymi serwerami WWW oraz programem zarządzającym alokacją zapytań, którym był system IBM Network Dispatcher, co z punktu widzenia klientów WWW prowadziło do stworzenia wirtualnego serwera WWW. Sprzętową platformę serwerów WWW tworzyły komputery IBM RS/6000 43P Model 260, pracujące pod kontrolą systemu operacyjnego AIX 4.3. Trzy komputery posiadały procesor RISC PowerPC Power3 taktowany częstotliwością 200 Mhz, 256 MB pamięci RAM, dysk twardy Ultra Wide SCSI o pojemności 4.3 GB oraz zintegrowaną kartę sieciową IBM 10/100 Mbps. Jeden komputer wyposażony był w dwa procesory PowerPC Power3 taktowane częstotliwością 200 Mhz, pracujące w architekturze SMP. Pozostała

konfiguracja tej maszyny była taka sama jak dla maszyn jednoprocesorowych. Na każdym z badanych serwerów zainstalowane było oprogramowanie serwera WWW Apache for AIX.



Rys. 3. Schemat sieci wykorzystanej w eksperymentach pomiarowych
Fig. 3. Local area network used in measurements

Eksperymenty przeprowadzono dla następujących konfiguracji serwera WWW:

- serwer WWW zbudowany w oparciu o pojedynczą maszynę jednoprocesorową,
- serwer WWW zbudowany w oparciu o maszynę dwuprocesorową,
- serwer WWW zbudowany w oparciu o klaster złożony z dwóch maszyn jednoprocesorowych, pracujący pod kontrolą programu IBM Network Dispatcher zainstalowanego na komputerze dwuprocesorowym,

- serwer WWW zbudowany w oparciu o klaster złożony z trzech maszyn jednoprocesorowych pracujący pod kontrolą programu IBM Network Dispatcher zainstalowanego na komputerze dwuprocesorowym.

Badane konfiguracje serwera WWW dobrze modelują możliwe konfiguracje rzeczywistych serwerów WWW z przyłączem do Internetu o przepustowości 100 Mbps.

Poszczególne komputery posiadały adresy IP z zakresu 156.17.130.80 ÷ 156.17.130.83, przy czym maszyna dwuprocesorowa posiadała adres IP 156.17.130.80. W przypadku rozproszonych serwerów WWW (konfiguracje 3 i 4) klaster posiadał adres IP 156.17.130.88, a program Network Dispatcher był konfigurowany zgodnie z zaleceniami producenta dla typowej konfiguracji.

Aby przebadać wpływ stosowanej wersji protokołu HTTP na wydajność serwera WWW dla każdej konfiguracji serwera opisanej powyżej przeprowadzono trzy rodzaje testów, różniące się używaną w nim wersją protokołu HTTP, a mianowicie:

- test wykorzystujący wyłącznie protokół HTTP v1.0;
- test wykorzystujący wyłącznie protokół HTTP v1.1 (z ograniczeniem maksymalnej liczby plików pobieranych równocześnie w trakcie jednego połączenia TCP do 20);
- test wykorzystujący w 50% zapytania HTTP v1.0 i w 50% HTTP v1.1 (z ograniczeniem jw.).

W każdym z powyższych testów klienci programu WebBench pracowali w trybie trzech wątków.

Pomiary wykonywane w trakcie eksperymentu przez program WebBench obejmowały: średnią liczbę żądań HTTP obsługiwanych przez serwer w ciągu sekundy oraz transfer (bajty/sekundę) w kierunku od serwera do klientów.

5.2. Wybrane wyniki pomiarów

Przedstawiamy jedynie wybrane wyniki pomiarów. Całość wyników zawierają prace [12, 14]. W analizie wyników przyjęto za punkt odniesienia testy przeprowadzone w konfiguracji „standardowej", tj. takiej, w której wagi serwerów wyliczane były na podstawie danych z tablicy otwartych połączeń i tablicy połączeń nowo przydzielonych (rys. 4 i 5). Maksymalna liczba obsłużonych zapytań na sekundę wyniosła 1616, a maksymalny transfer przekroczył 10 MB/sek. Wyniki te osiągnięto stosując wyłącznie zapytania HTTP v1.1. Jeżeli używano wyłącznie zapytań HTTP v1.0, to pomiary te pokazały odpowiednio 1169 zap./sek. i 7.2 MB/sek. Tym samym można było zaobserwować, jak duże znaczenie w podniesieniu wydajności serwisu ma zastosowanie nowszej wersji protokołu HTTP. W badaniach liczba plików, które klient WebBench mógł pobrać z wykorzystaniem pojedynczego połączenia TCP, ograniczono do 20, ale i w tych warunkach wydajność klastra serwerów WWW

obciążanego wyłącznie zapytaniami HTTP 1.1 była ok. 38% wyższa od wydajności uzyskanej dla protokołu HTTP 1.0 zarówno pod względem liczby obsłużonych zapytań, jak i szybkości transferu. Podobne zachowanie się systemu zaobserwowano we wszystkich innych konfiguracjach pomiarowych.

Jak należało się spodziewać, pojedynczy serwer jednoprocesorowy osiągnął najsłabsze wyniki dla każdego rodzaju protokołu. Pewną niespodzianką wydawać się mogą wyniki porównania rezultatów osiągniętych przez serwer dwuprocesorowy w stosunku do konfiguracji rozproszonych dwu – i trzy serwerowych. Serwer dwuprocesorowy był zawsze lepszy od konfiguracji dwu - serwerowej, a w przypadku stosowania wyłącznie protokołu HTTP 1.0 osiągnął najlepszy wynik spośród badanych serwerów (dla protokołu HTTP 1.1 i dla równomiernej mieszanki protokołów najlepsza okazała się konfiguracja złożona z trzech serwerów). Wynik taki może dziwić, jeśli zauważyć, że w konfiguracji dwuserwerowej mamy do dyspozycji większą ilość zasobów systemowych (osobne podsystemy dyskowe i osobna pamięć na każdym serwerze). Pierwszym nasuwającym się wyjaśnieniem takiej sytuacji jest możliwość wystąpienia wąskiego gardła w postaci połączenia pomiędzy oboma switchami. Aby sprawdzić tę hipotezę zestawiono połączenie pomiędzy serwerami a klientami WebBench w ramach tego samego switcha IBM 8275. W takiej konfiguracji wyniki klastra dwu - jak i trzyserwerowego nie poprawiły się (różnice wyniosły poniżej 2%). Zauważony stan możemy wyjaśnić na podstawie oceny obciążeń procesorów poszczególnych serwerów. Zauważono, że procesory serwera dwuprocesorowego w końcowej fazie testu obciążone były maksymalnie (tzn. obserwowano 0% czasu bezczynności procesora), podczas gdy w konfiguracjach rozproszonych obciążenie żadnego z procesorów serwerów nie przekraczało 85%. Obserwujemy więc, że klaster posiadał jeszcze spory zapas wolnej mocy przetwarzania. Jest to wynik zasady działania programu Network Dispatcher w konfiguracji standardowej. Pomiar obciążenia procesorów, który był przeprowadzony co sekundę, wykazał, że w mniej więcej pięciosekundowych ostępach czasowych jeden z serwerów w klastrze był w pełni obciążony, podczas gdy pozostałe obsługiwały mniej niżby mogły zapytań nadchodzących do klastra. Wynika to z faktu, że przy ustawieniach standardowych Network Dispatcher oblicza wagi każdego z serwerów co pięć sekund, kierując największy udział zapytań do serwera o najwyższej wadze (najwyższą wagę uzyskuje serwer, który był najmniej obciążony w ciągu poprzednich pięciu sekund). Dalsze badania niestandardowych konfiguracji programu Network Dispatcher pokazały możliwość uzyskania większych wydajności klastra niż w przypadku ustawień zalecanych przez producenta.

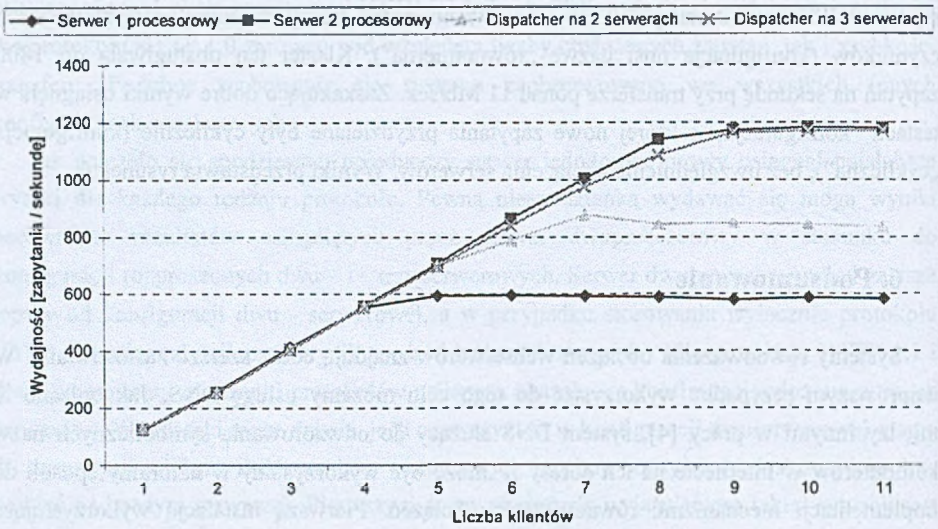
Najwyższe osiągi spośród wszystkich badanych konfiguracji uzyskał w teście protokołu HTTP 1.1 klaster trzyserwerowy, w którym wagi serwerów były obliczane na podstawie danych z tablicy otwartych połączeń, tablicy połączeń nowo przydzielonych, danych z

monitora ISS oraz doradcy HTTP - z równomiernym uwzględnieniem wszystkich tych czynników (konfiguracja nosi nazwę „równomierna"). Klaster ten obsługiwała do 1800 zapytań na sekundę przy transferze ponad 11 Mb/sek. Zaskakująco dobre wyniki osiągnęła w testach konfiguracja, w której nowe zapytania przydzielane były cyklicznie (konfiguracja „cykliczna"), bez uwzględnienia obciążenia serwerów. Wyniki przedstawia rysunek 6.

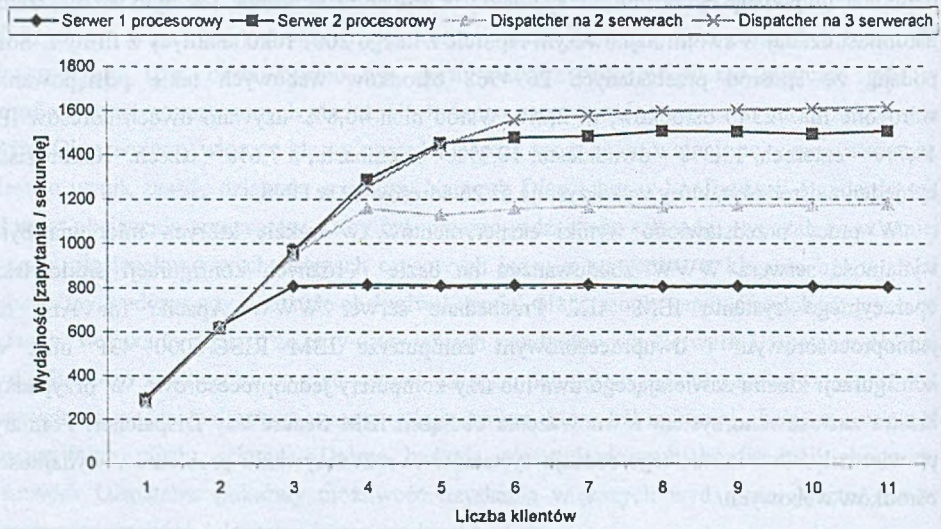
6. Podsumowanie

Systemy równoważenia obciążeń webserwerów znajdują coraz szersze zastosowanie. W najprostszym przypadku wykorzystać do tego celu możemy usługę DNS. Jak opisano to między innymi w pracy [4], system DNS służący do odwzorowania symbolicznych nazw komputerów w Internecie na ich adresy IP może być wykorzystany w naturalny sposób do implementacji mechanizmu równoważenia obciążeń. Pierwszą instalacją wykorzystującą DNS do równoważenia obciążeń był serwis WWW NCSA. Zestawiono tam klaster dziewięciu serwerów WWW, który stanowił odrębną domenę i był udostępniany pod nazwą www.ncsa.ninc.edu. Na podstawowym serwerze DNS domeny ncsa.ninc.edu skonfigurowano oprogramowanie, które na zapytania o odwzorowanie nazwy www.ncsa.ninc.edu odpowiadało podając cyklicznie adresy kolejnych serwerów z klastera. Tak było w 1994 roku, natomiast dzisiaj w swoim najnowszym raporcie z lutego 2001 roku analitycy z firmy E-Soft podają, że spośród przebadanych 2694968 ośrodków webowych takie postępowanie wdrożone ma 72340 ośrodków, tj. 2,7%. Wśród nich 40,8% używało dwóch adresów IP, 18,7% - czterech, 13,3% - dwudziestu, 10,3% - osiemnastu, a 7,6% - trzech. Rekordzistą był jeden z ośrodków, który wykorzystuje aż 26 adresów IP w DNS.

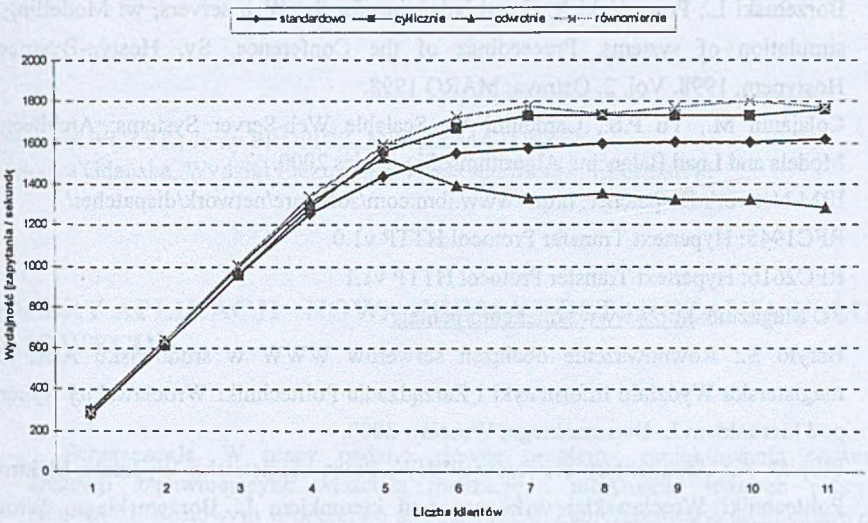
W pracy przedstawiono wyniki eksperymentów, w trakcie których mierzona była wydajność serwera WWW zbudowanego na bazie różnych konfiguracji środowiska operacyjnego systemu IBM AIX. Przebadano serwer WWW Apache for AIX na jednoprocesorowym i dwuprocesorowym komputerze IBM RISC/6000 43P oraz w konfiguracji klastra zawierającego dwa lub trzy komputery jednoprocesorowe. W przypadku klastra zastosowano system równoważenia obciążeń IBM SecureWay Dispatcher. Pomiary potwierdziły skuteczność tego rodzaju systemów w rozwiązywaniu problemów wydajności ośrodków webowych.



Rys. 4. Wydajność obsługi zapytań HTTP v1.0 w konfiguracji standardowej
Fig. 4. Performance of HTTP v1.0 in standard configuration



Rys. 5. Wydajność obsługi zapytań HTTP v1.1 w konfiguracji standardowej
Fig. 5. Performance of HTTP v1.1 in standard configuration



Rys. 6. Wydajność obsługi zapytań HTTP v1.1 dla różnych algorytmów równoważenia obciążeń

Fig. 6. Performance of HTTP v1.1. for different load balancing algorithms

LITERATURA

1. Borzemski L.: Load balancing in parallel and distributed processing of tree-based multiple-task jobs. Proceedings of the Euromicro Workshop on Parallel and Distributed Processing. Los Alamitos: IEEE Computer Society Press 1995.
2. Borzemski L., Gajewski D., Głowacz S., Szczypiński M., Wojtczak K.: A load balancing system for Unix-based area networks, Microprocess. & Microprogram, 1993 vol. 39 nr 2-5.
3. Borzemski L., Kieda A.: A load balancing system for Windows NT networks, Beyond 2000: hardware and software design strategies. Proceedings of the 22nd EUROMICRO Conference, IEEE Computer Society Press, Los Alamitos 1996.
4. Borzemski L., Nowak Z., Porczyński R.: Metody, algorytmy i rozwiązania systemowe równoważenia obciążeń serwerów WWW, Studia Informatica, Vol. 21, no 1, Wydawnictwo Politechniki Śląskiej, Gliwice 2000.
5. Borzemski L., Nowak Z., Porczyński R.: Load balancing for WEB systems, Information Systems Architecture and Technology ISAT '2000, Proceedings of the 22th International Scientific School, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2000.

6. Borzemski L., Porczyński R.: Load balancing for the Web servers, w: Modelling and simulation of systems. Proceedings of the Conference. Sv. Hostyn-Bystrice p. Hostynem, 1998. Vol. 2. Ostrava: MARQ 1998.
7. Colajanni M., Yu P.S., Cardellini V.: Scalable Web-Server Systems: Architectures, Models and Load Balancing Algorithms. Sigmetrics 2000.
8. IBM Network Dispatcher: http://www.ibm.com/software/network/dispatcher/.
9. RFC1945: Hypertext Transfer Protocol HTTP v1.0
10. RFC2616: Hypertext Transfer Protocol HTTP v1.1
11. PC Magazine: http://www.zdnet.com/pcmag/
12. Baryło S.: Równoważenie obciążeń serwerów WWW w środowisku AIX, Praca magisterska Wydziału Informatyki i Zarządzania Politechniki Wrocławskiej wykonana pod kierunkiem L. Borzemskiego, Wrocław 2000.
13. Filipiak A.: Architektury serwerów WWW, Praca magisterska Wydziału Elektroniki Politechniki Wrocławskiej wykonana pod kierunkiem L. Borzemskiego, Wrocław 1999.
14. Kasprzyk G.: Charakterystyka obciążenia i wydajności serwerów WWW w środowisku AIX, Praca magisterska Wydziału Informatyki i Zarządzania Politechniki Wrocławskiej wykonana pod kierunkiem L. Borzemskiego, Wrocław 2000.

Recenzent: Prof. dr hab. inż. Andrzej Grzywak

Wpłynęło do Redakcji 30 marca 2001 r.

Abstract

This paper presents results from a comprehensive empirical study of Web servers running in cluster configuration with load balancing mechanisms. The paper shows results of measurements of Apache for AIX Web server running under AIX operating system in different computer configurations: single-processor system, two-way processor system, two-server and three-server clusters. Benchmark studies were performed with Ziff-Davis's WebBench 3.0 software under control of the IBM SecureWay Network Dispatcher used for load balancing of web servers due to servicing WWW requests. Our empirical results illustrated that highly efficient WWW service can be implemented on the basis of load balancing mechanisms.