Vol. 44 No. 3 情報処理学会論文誌 Mar. 2003

# DNSフィルタ方式によるミラーサーバ選択法の提案と実装

横田裕思[†] 木村成伴[††] 海老原義彦[††]

サーバに対する過度の負荷集中への対処として，ミラーサーバを用いたサーバの複数化による負荷分散が広く行われている．このシステムでは，どのサーバを選択するかにより，通信効率が大きく左右される．一般に，ユーザが適切なサーバを選択することは困難であるため，各サーバに負荷を均等に分散させるDNSラウンドロビンなどを用い，サーバ周辺に負荷を分散させる方式がとられていた．しかし，これはあくまでもサーバ側から見た負荷分散であった．すなわち，この方式で選択されたサーバはクライアントの状態とは無関係に選択されるため，クライアントにとって必ずしも最適なサーバが選択されるとは限らない．この傾向は，特に，ミラーサーバがネットワーク上に分散して配置されている場合に，顕著に見られる．そこで本論文では，ローカルDNSキャッシュサーバにフィルタ機能を持たせ，クライアントからのDNS問合せに対する上位DNSサーバの応答から，ユーザにとって適切なサーバを自動的に取捨選択するDNSフィルタ方式を提案する．そして，クライアントから見て最も短い時間でファイルを転送するサーバを最適とするシステムを想定し，そのためのサーバ選択システム例を3つ示すとともに，本システムを実装する．最後に，本実装システムと従来のDNSラウンドロビン方式を用いた場合のファイル転送速度を比較評価し，本提案方式の有効性を示す．

# A Proposed of DNS Filter Methods to Select Suitable Mirror Servers for Clients and Its Implementation

Hiroshi Yokota,[†] Shigetomo Kimura[††] and Yoshihiko Ebihara[††]

In order to prevent servers from overloading, the mirror-server systems constructed by multi-servers distributing the load are widely used. Although the selection method of the server is very important for effective communication in such systems, it is too difficult to find the appropriate server for each client in general. Previously, the DNS (Domain Name System) Round-Robin mechanism is adopted for this purpose, however, the selected server is not always suitable for the client. To solve this problem, this paper proposes and implements a DNS filtering method. In this proposal, the local DNS cache server has filtering functions. When each client queries IP addresses corresponding to the server's name, the local DNS cache server forwards the queries to the exterior DNS server, and then filters the appropriate IP addresses from the replies of the queries for the client. There are various sorts of criterion to filter better servers. In this paper, for a system where the best server for clients transfers files in the minimum time, three kinds of server selection methods are introduced as examples, and then the system is implemented. Finally, we estimate the file transfer speeds of both the proposal method and Round-Robin DNS method.

## 1. はじめに

近年のインターネットの急激な普及にともない，一部の主要なサーバに対してアクセスが過度に集中する傾向が見られる．また，そのためにサーバ周辺のネットワークの混雑が激しくなるという問題も生じている．

サーバへの負荷を分散させる方法の1つとして，ミラーサーバを用いる方法がある．ミラーサーバとは，基となるマスタサーバからすべてのデータをあらかじめコピーしておき，マスタサーバとまったく同じ情報を提供するものである．これにより，特定のサーバやその周辺のネットワークに負荷が集中することを防ぐことができる．しかし，ミラーサーバを用いた場合，どのサーバを選択するのが望ましいかを何らかの手段で調べる必要がある．このような方式として，文献1)であげられているように，サーバおよびその周辺のネットワークに分散化システムを設置することにより分散を行う方法が主として提案されている．しか

† 筑波大学大学院工学研究科
Doctoral Program in Engineering, University of Tsukuba
†† 筑波大学電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba

し，これらのシステムでは，定期的にミラーサーバ群の負荷情報を収集する必要があるため，システムが大規模になる傾向がある．また，これらの方式は，サーバ側にとっての負荷分散であるため，その選択結果がクライアントにとって必ずしも最適であるとは限らなかった．

この問題を改善するため，本論文ではローカルDNS[2]キャッシュサーバにミラーサーバの取捨選択を行うフィルタ機能を持たせ，これによって，ユーザにとって適切なサーバを自動的に選択するDNSフィルタ方式を提案する．クライアントにとっての最適なサーバはそれぞれのクライアントによって異なると考えられるが，本提案方式ではそれぞれの評価方法に応じたサーバの調査システムを準備することで，これに対応している．その調査システムの例を示すため，クライアントから見て最も短い時間でファイルを転送するサーバを最適とするシステムを想定し，そのためのサーバ選択システムを3つ示すとともに，本システムを実装する．最後に，DNSラウンドロビン方式と本実装方式におけるファイル転送速度を測定し，提案方式の評価を行う[3]．なお，本提案方式はクライアントのためのサーバ選択を目指すものであり，上述したサーバのための負荷分散を主目的としていないことに注意されたい．

本論文の構成は以下のとおりである．まず，2章では既存の負荷分散方式について解説をする．3章では提案方式のDNSフィルタ方式について述べ，4章で本方式の実装について説明する．5章では提案方式の評価を行い，6章でまとめを述べる．

## 2. サーバの選択方式

ミラーサーバを用いて分散されたサーバにユーザが効率的にアクセスするためには，どのサーバにアクセスするのが最も効率的に転送できるかをあらかじめ調べる必要がある．文献1)などによれば，現在これを実現する方法として，ユーザが手動でサーバを選択する方式や，DNSによるラウンドロビン方式などのいくつかの方式がある．本章では，これらのうち，主な手法について概説する．

### 2.1 クライアントによる負荷分散

最も原始的なサーバ選択方式は，各ユーザが最適なサーバを手動で選択することである．典型的なサーバの選択基準として，以下に示すものがある．

- ネットワーク上で近いサーバをユーザ自身が知っているならば，それを選択
- そうでなければサーバにかかる負荷などの情報を得て，その情報を基に選択

この方式は，すべてのユーザがサーバの負荷状況を把握し，ネットワークに関する十分な知識を持っていることを必要とする．しかし，一般のユーザがこのような知識を持っていると期待することは大変難しい．

これを改善するため，あらかじめ内部に特定のサーバおよびプロトコル向けの負荷分散機能を内蔵させた特別なクライアントを用いて負荷分散を行う方式[4]がある．この方式では対象とならないサーバやプロトコルにこの機能を用いることはできないという問題がある．

### 2.2 中間システムによる負荷分散

手動方式を自動化する方法として中継サーバ方式がある．この方式では，まずミラーサーバ群の前に中継サーバを置き，ユーザをこのサーバにアクセスさせる．中継サーバはミラーサーバ群の中から適切なサーバを選択し，クライアントと選択されたサーバの間の通信を中継する．この方式はサーバ自体の負荷分散は可能だが，中継サーバにアクセスが集中するため，ネットワークの負荷分散はできない．また，プロトコルごとに中継サーバを作る必要があるうえに，中継サーバそれ自体の性能がミラーサーバシステム全体のボトルネックになることがある．

一方，次世代インターネットプロトコルであるIPv6には，負荷分散に使用できる“Anycast”という機能がある．これはネットワーク上でパケットを運ぶ個々のルータが，広範囲に散らばった複数のサーバのうちで最も適していると判断したある1台のサーバに向けてパケットを送り届けるというものである．

これはIPの機能であるため，プロトコルにまったく依存せず応用範囲が大変広いが，Anycastの詳細がまだ不透明であるうえ，Anycastを正しく理解できるルータが配置されている必要がある．そのため，現時点でこの方式を用いることは現実的ではない．

### 2.3 サーバによる負荷分散

HTTPサーバの機能の1つに任意のアクセスをインターネット上の任意のサーバへ振り向ける機能がある．これを「リダイレクション」と呼ぶ．この機能を利用したサーバの選択方式もある．

すなわち，**図1**のように振り向け機能を持った代表サーバを置き，まず初めにユーザをそこにアクセスさせ，代表サーバがユーザのHTTPアクセスを適切なサーバへ振り向けさせる．この方式は非常に柔軟な設定が可能だが，HTTPのプロトコルに依存しているために他の種類のプロトコルでは使えない．さらに，いったん選択させると各サーバの状況が変わってもそ

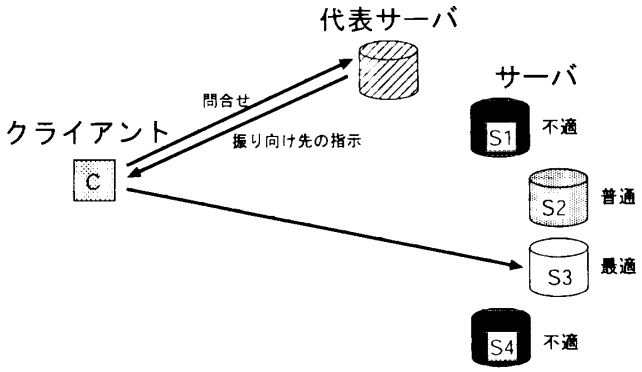


図 1 HTTP リダイレクションを用いた方式
Fig. 1 HTTP redirection method.

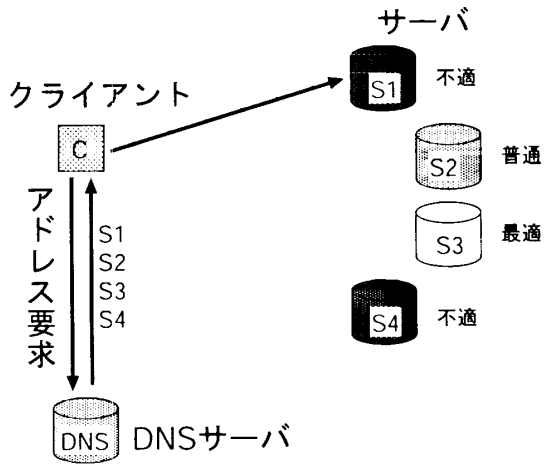


図 2 ラウンドロビン方式による負荷分散
Fig. 2 Load balancing in DNS Round-Robin method.

れに追従できないという欠点がある．

### 2.4 DNS を用いた負荷分散

一般に，ユーザがサーバを指定する際は，サーバの IP アドレスではなく，“www.example.com” などの分かりやすい名前を用いる．DNS サーバはこの名前からそれに対応する IP アドレスを調べるサーバである．

DNS サーバでは，ある名前に対して複数の IP アドレスを登録できる．たとえば図 **2** の例では，あるホスト名に対して 4 つのサーバの IP アドレス S1～S4 が登録されている．この名前の問合せに対して，DNS サーバは登録された IP アドレスすべてを提示するが，その後，IP アドレスのリストは循環される．この方式をラウンドロビン方式と呼ぶ．クライアントは提示された IP アドレスの中から 1 つの IP アドレスを選択するが，その選択は完全にランダムであり，各サーバは均等に選ばれる．現在，この方式を用いてミラーサーバの負荷分散が実現されている．しかし，この方式はサーバの選択がサーバの状況とは関係なく決定されるため，必ずしも最適なサーバが選択されるとは限らない．この問題を解決するため，TENBIN[5] と DNS Balance[6),7)] などのシステムが提案されている．

#### 2.4.1 TENBIN

TENBIN[5] は九州大学で開発された負荷分散 DNS サーバである．本システムは，負荷分散対象となるサーバ群の IP アドレスなどの情報を保持する DNS コンテンツサーバと，これらのサーバ群を定期的に調査するシステムにより構成されている．クライアントから調査対象サーバの IP アドレスおよびその他の情報要求があると，TENBIN は調査システムからの情報を基に最適なサーバの情報を選択して返答する．この選択にはさまざまな手法を採用できるが，文献 5) では BGP-4[8] を用いてサーバ付近のルータから収集した経路情報に基づいた方式などが提案されている．ただし，負荷分散の対象は調査システムがターゲットにしているシステムに限定される．

#### 2.4.2 DNS Balance

DNS Balance[6),7)] は，TENBIN と同様に，DNS コンテンツサーバに負荷分散機能を付加したものである．この DNS コンテンツサーバは各サーバの負荷状態を一定期間ごとに収集し，このデータを基にクライアントに適しているサーバの IP アドレスを 1 つまたは複数返答する．その際，ある特定のサーバに処理が集中しすぎないように，選択にはランダムな要素を加えている点が TENBIN と異なっている．

本方式では，サーバ側のある程度の負荷分散を実現しているが，TENBIN と同様に，この負荷分散は調査対象となるシステムに限定される．また，サーバとクライアントの間のネットワーク上での距離を考慮できないなどの問題点があり，本方式によっても，クライアントにとって適切なサーバが必ずしも選択されない．

## 3. DNS フィルタ方式

前章で述べた方式は，2.1 節の手動による方式を除き，すべてサーバやその付近で負荷分散を行うものである．文献 1) には，サーバおよびその近辺にさらに複雑な仕組みを置いて分散を行う仕組みがいくつか紹介されている．しかし，これらの仕組みは複雑な分，システムが大規模になりがちであり，また，その分散結果がクライアントにとって必ずしも最適であるとは限らない．クライアントにとって良いサーバは，それぞれのクライアントによって異なり，その判断はクライアント側で行う必要がある．この点については 2.1 節の手動による方式が近いといえる．

そこで，本論文では，この手動による方式を自動的に行うことを目的とする．具体的には，図 **3** に示すように，IP アドレス情報を持つ上位 DNS サーバとクラ

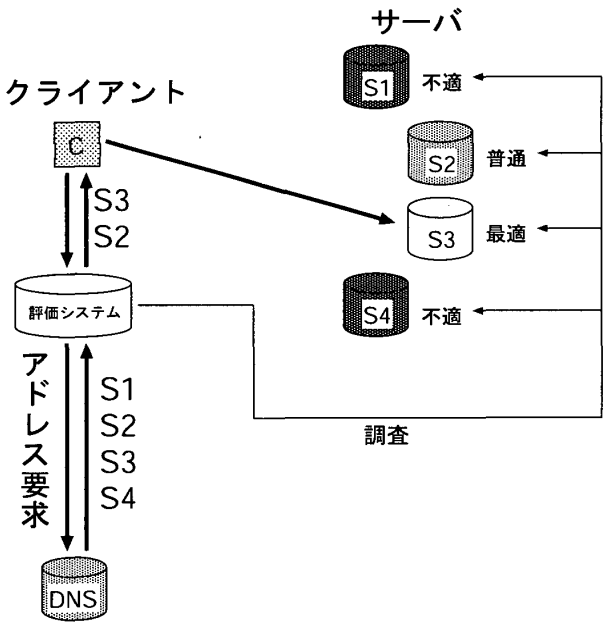


図 3 DNS フィルタ方式
Fig. 3 Load balancing in DNS filter method.

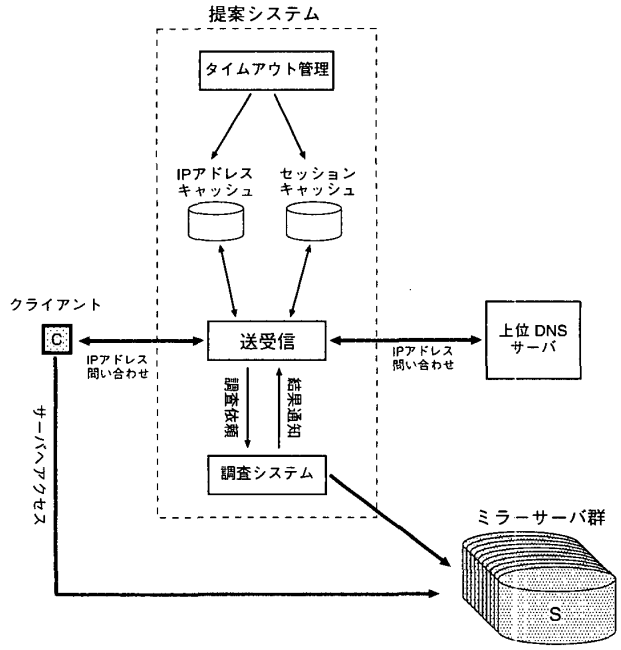


図 4 提案方式の構造
Fig. 4 Proposed method's structure.

イアントの間に評価システムを置く．このシステムが上位 DNS サーバから得た IP アドレス（A レコード）が複数あった場合は，これをミラーサーバと見なす．そして，それらの IP アドレス群からクライアントにとって適切ではないものを取捨選択し，適切なサーバの IP アドレスだけをクライアントに返す．以下では，本評価方式を DNS フィルタ方式と呼ぶ．

さて，この評価システムは，クライアントの LAN に属するローカルな DNS キャッシュサーバと置き換えることを想定している．通常，クライアントはローカルな DNS キャッシュサーバに IP アドレスの問合せを行うことから，これを提案システムで置換することでフィルタリング操作を容易に実装できる．また，DNS コンテンツサーバを構成する 2.4.1 項の TENBIN や 2.4.2 項の DNS Balance とは異なり，クライアントが属するドメインに評価システムを置くため，クライアント側に適したサーバの選択が可能になる．さらに，この評価システムでは上位 DNS サーバから IP アドレスが取得されてからそれらのサーバに対する調査を行うため，TENBIN や DNS Balance とは異なり，評価対象を特定システムのみに限定されることがないという利点を持つ．

一方，評価システムをクライアントに組み入れるのではなく，DNS キャッシュサーバを置換する形で設置するため，提案方式により得られる評価結果とクライアントで評価した結果が必ずしも一致しない可能性がある．しかし，提案方式の評価システムはクライアントの LAN に属しているため，両評価結果の差異は十分小さいものと考えられる．他方，現存するクライアントを変更する必要がなく，ローカル DNS キャッシュサーバを置換するだけでよいことから，提案方式の導入は容易である．また，提案方式には，あるクライアントのために調査した結果を，(サーバに対する同一の評価基準を持つ) 他のクライアントが利用できるという利点もある．ミラーサーバはアクセス頻度が高いために負荷を分散する目的で設置されたものであり，同一 LAN 内に所属する複数のクライアントからほぼ同時にこれらミラーサーバ群へのアクセスが起こることは十分予想される．このような場合でも，提案方式ではミラーサーバ群への調査は一度で済むが，各クライアントに調査システムを実装した場合は，クライアントの台数分だけ調査を行う必要がある．

なお，web アクセスでは cookie や SSL などのセッション管理を行うシステムがあるが，これらのシステムで提案方式を利用する際に，名前解決の一意性保持が崩れる懸念がある．しかし，本来，ミラーサーバシステムではクライアントがどのサーバを選択するかは完全にランダムであることから，接続ごとのセッション維持はサーバ側の責任であり，提案方式が考慮すべきことではない．

さて，提案システムは図 4 の点線枠内にある 3 つの処理システムと 2 つのキャッシュから構成することができる．これらの構成を用いて，本システムは従来の DNS キャッシュサーバと同じ動作を行う．ただし，クライアントから A レコードの問合せがあった場合は，以下に示すように，DNS キャッシュサーバとは異

なる動作をとる．

まず,「パケット送受信システム」は，クライアントから依頼されたIPアドレス（Aレコード）の問合せを上位DNSサーバに対して行い，その結果を「IPアドレスキャッシュ」に格納する．格納されたサーバのIPアドレスは，クライアントに回答されるとともに，次回の問合せに備えて,「クライアントが接続すべきサーバの調査システム」が各サーバの調査を行い，その結果によりサーバのIPアドレスを取捨選択する．また，クライアントとの間のセッションは，セッションキャッシュによって管理され，有効期間が過ぎたセッションは,「キャッシュのタイムアウト管理システム」によって整理される．

以下の各節では，主にAレコードの問合せがあった場合について，各構成部分の詳細について説明する．

### 3.1 DNSパケットの送受信システム

送受信システムではクライアントからのDNSパケットを受け取り，IPアドレスキャッシュに情報があれば上位DNSサーバに問い合わせることなくキャッシュ中の情報をそのまま返す．

情報がなかった場合は，上位DNSサーバに問い合わせてその返事を待つ．返事を受け取ったらセッションキャッシュを調べ，該当の情報がなければ，上位DNSサーバからの返事は捨てられる．該当する情報があれば，クライアントにそのまま返事を返すとともに，その接続状況をセッションキャッシュから削除する．さらに，DNSサーバから得たIPアドレスすべてをIPアドレスキャッシュに登録し，これらのIPアドレスが即座に回答できるようにする．また，IPアドレスを示すAレコードの問合せに対して複数のAレコードの回答があった場合は，これをミラーサーバシステムであると判断する．そして，次節の調査システムを起動し，IPアドレスのフィルタリングを行う．なお，上位DNSサーバから得たIPアドレスをクライアントにそのまま返さずに，このフィルタリングの結果をクライアントに回答する実装も可能である．しかし，調査には時間が要すると想定されるため，フィルタリング前の結果をクライアントに回答しておき，調査終了後に問い合わせたクライアントに対して，フィルタリング後のIPアドレスを回答する実装の方が好ましいと考えられる．次章の実装システムでは，この方式を採用している．

ところで，DNSサーバが提供する情報にはそれぞれ有効期限が記入されている．通常は，サーバのIPアドレスが頻繁に変わることはないので，DNSシステムではキャッシュを多用し，各情報の有効期限を長めに設定することで無駄な問合せを減らしている．

しかし，ネットワークの状況は時間とともに変化することから，過去に最適であったサーバが，ある程度の時間が経過すると，最適ではなくなっている可能性がある．このため，提案方式では，Aレコードの有効期限をある程度短くして接続サーバを再度調査させる必要がある．有効期限を短くした場合でも，ネットワーク上ですぐ近くに存在する上位DNSサーバのキャッシュに問い合わせることが可能なので，これによるネットワークへの悪影響は少ない．

### 3.2 クライアントが接続すべきサーバの調査システム

調査システムでは，送受信システムがIPアドレスキャッシュに登録したIPアドレスのリストから，適切なIPアドレスを1つまたは複数選択する．そして，その調査結果をIPアドレスキャッシュに反映する．ただし，すべてのIPアドレスへのアクセスが不可能であった場合は，すべてのサーバのIPアドレスをキャッシュに登録したままにする．IPアドレスのリストを完全にキャッシュから削除してしまうと，クライアントから再アクセスされた場合，再調査を行う必要がある．しかし，次回に同じ調査をしたとしても回答が得られないことが予想される．したがって，キャッシュを削除することは，ネットワークに不必要なオーバヘッドをかけることになる．

### 3.3 IPアドレスキャッシュ

IPアドレスキャッシュにはホスト名とそれに対応するIPアドレス，および，それぞれの情報の有効期限が保存される．

### 3.4 セッションキャッシュ

DNSの問合せに一般的に用いられるUDPプロトコルは，TCPのようにセッションを作ることができない．このため，DNSシステムではパケットにID番号を含めるなどして個々の問合せを区別し，疑似的にセッションを維持する．この接続情報を保存するのがセッションキャッシュである．このキャッシュには個々の接続情報のほか，それぞれの接続情報の有効期限も設定される．

### 3.5 キャッシュデータのタイムアウト管理システム

DNSの情報には有効期限があり，IPアドレスキャッシュ中の古くなった情報は廃棄する必要がある．また，ある問合せに対して一定期間以上回答がなかった場合も，セッションが切れたと判断して，問合せに関する情報をセッションキャッシュから廃棄する必要がある．

そのため，タイムアウト管理システムはつねに各キャッシュを監視して，有効期限が切れたデータをキャッ

シュから取り除く．

## 4. DNSフィルタ方式の実装

前章の検討をふまえ，提案システムを "DNS Trick" という名前で実装した．本システムの実装にはRuby[9]を用いた．Rubyはサーバの設計に便利なマルチスレッドや，エラー処理に便利な例外処理構文を持っており，今回の実装には最適であると判断した．

さて，クライアントにとっての最適なサーバはそれぞれのクライアントによって異なると考えられ，提案方式では，その目的にあったサーバの調査システムを実装することで，これに対応することができる．以下の実装では，クライアントから見て最も短い時間でファイルを転送するサーバを最適とするシステムを想定し，各サーバの評価の指標として，各サーバとのネットワーク上の距離，すなわち，各サーバの応答速度を測定する．このために，本実装で採用したサーバ選択システムは以下の3つであり，これらは調査の精度と速度に関して違いがある．

なお，これらのシステムで測定される応答速度は，調査システムにとっての値であり，クライアントから測定した結果とは必ずしも一致しない可能性がある．しかし，前章で述べたように，本提案方式はクライアントのLANに属するローカルDNSキャッシュサーバと置き換えることを想定しており，すべてのミラーサーバがクライアントのLANに属していない場合は，調査システムおよびクライアントから測定した応答速度の差は十分に小さいと見なすことができる．また，ミラーサーバの一部がクライアントのLANに属している場合，この事実はミラーサーバのIPアドレスなどから容易に判別できるため，これらのサーバに対する特別な対処を行うことは可能である．

### 4.1 ICMP ECHOパケットに対する応答速度を用いた方法

まず，高速かつ低精度な方法としてICMP ECHOパケットを用いる方式を使用する．すなわち，調査の依頼があると，各ミラーサーバに向けて同時にICMP ECHOパケットを送信する．これを受信したサーバはそれに対する回答としてICMP ECHO REPLYパケットを返す．これにより，最も回答の早かった1つまたは複数のサーバを選択する．

### 4.2 netselectを用いた方法

netselect[10]は，Pennarunが作成したICMPとUDPを用いて精密に応答速度を検証するツールである．ただし，その調査には約30秒程度かかる．第2の方法では，このnetselectの出力を用いて選択を行う方式を用いる．

### 4.3 TCP接続の接続時間を用いた方法

ミラーサーバシステムを用いているのはWebサーバのみであるという仮定をおき，TCP 80番ポートへの接続にかかる時間を計測し，最も短いホストを選択する．計測には状況によるが，数秒かかる．以下では，この方法を「TCP接続時間を用いた方式」と呼ぶことにする．

なお，本方式をProxyサーバを介した接続に用いると，Webサーバではなく，Proxyサーバ自体への接続時間を計測することになり，意図した測定にはならない．ただし，このProxyサーバが属するLAN内に本方式を用いたDNSキャッシュサーバを設置することで，Webサーバへ直接接続した場合の接続時間と同様な結果が得られる．

## 5. 評価実験

ミラーサーバを積極的に活用したシステムとして，RingServerプロジェクト[11]のRingServerがある．これは，インターネットなどの高速ネットワーク環境を対象に，大規模なソフトウェアライブラリとソフトウェアの分散共同開発の支援を行う共通基盤技術の構築を目的としたプロジェクトで，その研究の一環として日本各地に分散された，同一の内容を持つサーバを運営している．本章では，このRingServerから前章の実装方式（以下では，提案方式と呼ぶ）と2.4節で述べたのDNSラウンドロビン方式（従来方式と呼ぶ）の各方式でファイルを転送させたときの，転送速度の比較を行い，提案方式の評価を行う．なお，提案方式において上位DNSサーバから得られたIPアドレスから選択するIPアドレスの個数は1個とし，IPアドレスの最大有効期限を15分とした．また，セッションキャッシュの有効期限は60秒とした．

### 5.1 巨大ファイルの転送

まず，当研究室においてRINGサーバからhttpプロトコルにより巨大なファイルを取得させ，提案方式と従来方式を用いた時の転送時間の比較を行う．転送させるファイルは `http://www.ring.gr.jp/archives/GNU/emacs/emacs-20.7.tar.gz` で，ファイルサイズは約14.3メガバイトである．このファイルを提案方式と従来方式で同時に，一定時間ごとに転送させ，その転送速度を測る．測定は，指定した時間に自動的にプログラムを実行するcronプログラムを用いて，15分おきにファイル転送を行った．そしてこれらを24時間繰り返し実行させた．

提案方式に実装した各選択方式と従来方式における

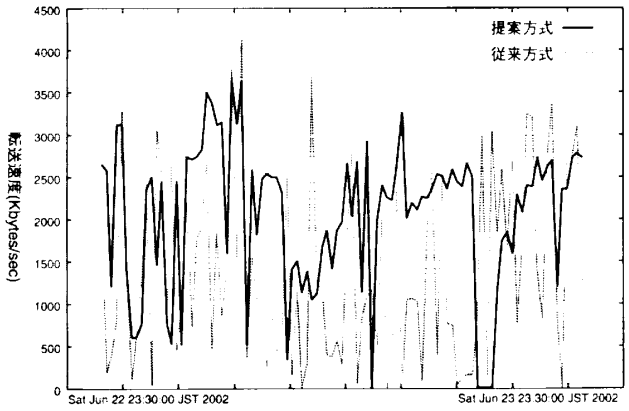


図 5 従来方式と提案方式の巨大ファイルの転送速度（ICMP ECHO を用いた場合）

Fig. 5 Compare big one file transfer speed between ordinary method and proposed method (ICMP ECHO filter).

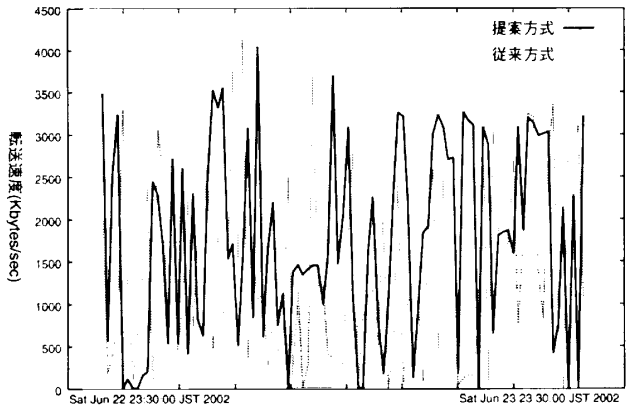


図 6 従来方式と提案方式の巨大ファイルの転送速度（netselect を用いた場合）

Fig. 6 Compare big one file transfer speed between ordinary method and proposed method (netselect filter).

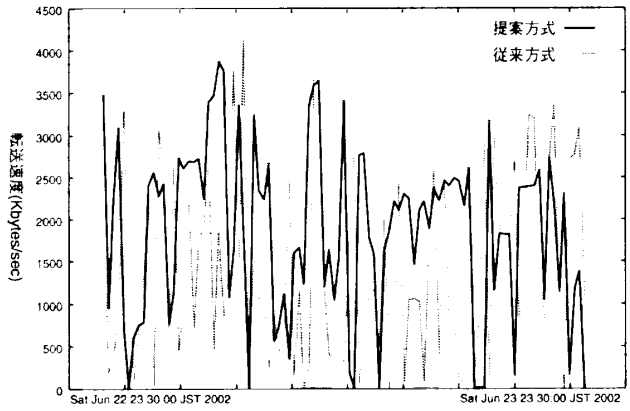


図 7 従来方式と提案方式の巨大ファイルの転送速度（TCP 接続時間を用いた場合）

Fig. 7 Compare big one file transfer speed between ordinary method and proposed method (TCP connection establishment time filter).

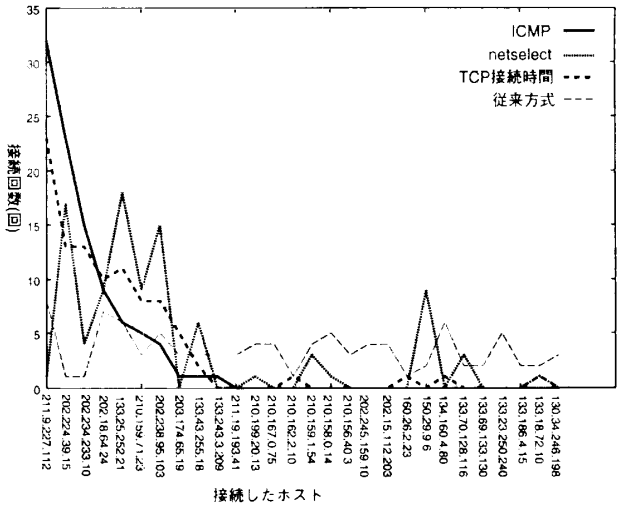


図 8 巨大ファイル転送時の転送回数比較

Fig. 8 Connected server in each method when transfering big one file.

結果をそれぞれ図 5，6，7 に示す．

この結果より，ほとんどの場合で従来方式よりも提案方式の方が転送速度が大きくなっていることが分かる．また，各方式のファイル転送 1 回あたりの平均転送速度を比較すると，ICMP ECHO パケットを用いた提案方式では 2049 K バイト/秒，netselect を用いた提案方式の場合は 1763 K バイト/秒，TCP 接続時間を用いた提案方式の場合は 1875 K バイト/秒，従来方式の場合は 1378 K バイト/秒であった．これは，図 8 に示す各サーバからのファイル転送回数を見ると分かるように，提案方式ではクライアントからネットワーク上での距離の近いサーバが数多く選択されているのに対し，従来方式では距離に関係なく選択されていることから，ネットワーク上で遠い距離にあるサーバをより多く選ぶ従来方式の方がファイル転送速度が小さくなったためである．

各選択方式を比較すると，netselet 方式では非常に精密な応答速度を測定すると期待されるが，本提案方式では他 2 種類の選択方式と比較してあまり良い結果が得られなかった．これは，各選択方式において，調査のために各サーバに送出されるパケットに対して，ICMP ECHO 方式と TCP 接続時間方式ではほとんどのサーバが応答を返したのに対し，netselect 方式の場合は半数以下のサーバのみしか応答を返さなかった．このため，netselect 方式ではうまくホストを選択できなかったものと考えられる．netselect は，わざと通信エラーを引き起こすようなパケットを送出してそれに対する応答からサーバ選択を行うため，セキュリティ上の観点からこのようなパケットがネットワーク経路の途中で排除されたと推定される．

一方，TCP接続時間方式ではTCPの接続時間のみを用いたため，各サーバでのパケット処理時間が考慮されず，ICMP ECHO方式よりもやや結果が悪くなった．これを改善するためには，データ転送時間を測定すればよいと考えられるが，httpサーバの設定の差異から，全サーバからまったく同じ応答を期待するのは困難であり，この点については検討の余地がある．ただし，ファイヤウォールの制限により，ICMP ECHOが禁止されている場合は本方式が有効であるといえる．

### 5.2 多数ファイルの転送

次に，当研究室においてRINGサーバからhttpプロトコルにより多数のファイルの一括転送を行い，提案方式と従来方式を用いたときの転送速度の比較を行う．転送対象は`http://www.ring.gr.jp/pub/doc/RFC/`にある`rfc1000.txt`から`rfc1100.txt`の101個のファイルである．これらのファイルのファイルサイズは最小の`rfc1061.txt`の29バイトから最大の`rfc1000.txt`の315,315バイトまでさまざまであり，1ファイルあたり平均で約60キロバイトある．これらのファイルを提案方式と従来方式で同時に，一定時間ごとに転送させ，その転送時間の合計を測定する．そして，総ファイルサイズをこの合計時間で割ることで，1回あたりのファイル転送速度を求める．測定はcronプログラムを用いて15分ごとにファイル転送を行い，これを24時間繰り返し実行させた．

提案方式に実装した各選択方式と従来方式における結果をそれぞれ**図9**，**10**，**11**に示す．

この結果からも，ほとんどの場合で従来方式よりも提案方式の方が転送速度が大きくなっていることが分かる．また，各方式のファイル転送1回あたりの平均転送速度を比較すると，ICMP ECHOパケットを用いた提案方式では5,564Kバイト/秒，netselectを用いた提案方式の場合は3,554Kバイト/秒，TCP接続時間を用いた提案方式の場合は5,038Kバイト/秒，従来方式の場合は786.3Kバイト/秒であった．これも，**図12**に示す各サーバからのファイル転送回数を見ると分かるように，提案方式ではクライアントからネットワーク上での距離の近いサーバが数多く選択されているのに対し，従来方式では距離に関係なく選択されていることから，ネットワーク上で遠い距離にあるサーバをより多く選ぶ従来方式の方がファイル転送速度が小さくなったためである．

ところで，図8と図12を比較すると，サーバの選択傾向が似ているにもかかわらず，1つの巨大ファイルを転送したときよりも多数ファイルを転送したときの方

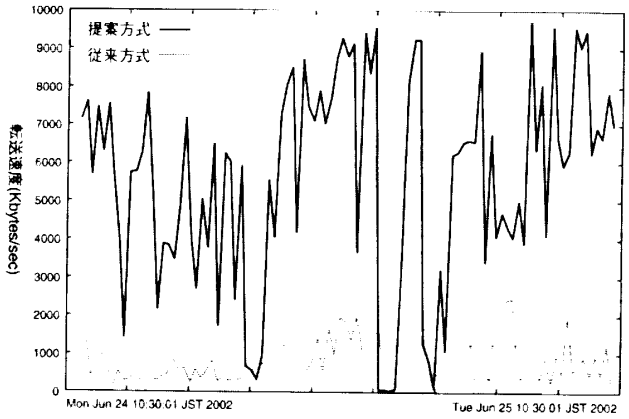


**図9** 従来方式と提案方式の多数ファイルの転送速度（ICMP ECHOを用いた場合）

Fig. 9 Compare many files transfer speed between ordinary method and proposed method (ICMP ECHO filter).

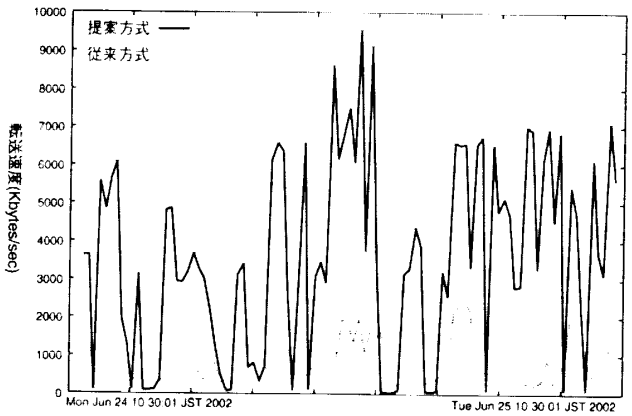


**図10** 従来方式と提案方式の多数ファイルの転送速度（netselectを用いた場合）

Fig. 10 Compare many files transfer speed between ordinary method and proposed method (netselect filter).

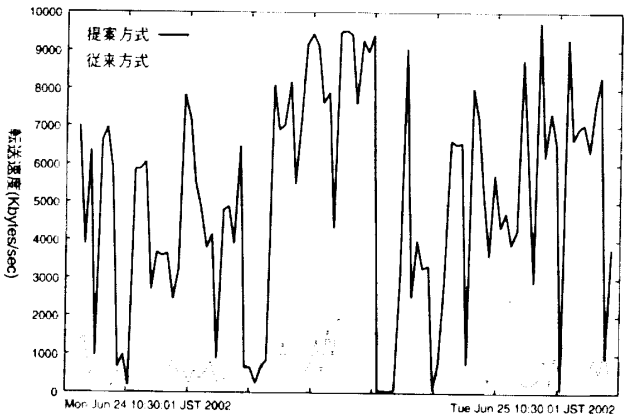


**図11** 従来方式と提案方式の多数ファイルの転送速度（TCP接続時間を用いた場合）

Fig. 11 Compare many files transfer speed between ordinary method and proposed method (TCP connection establishment time filter).

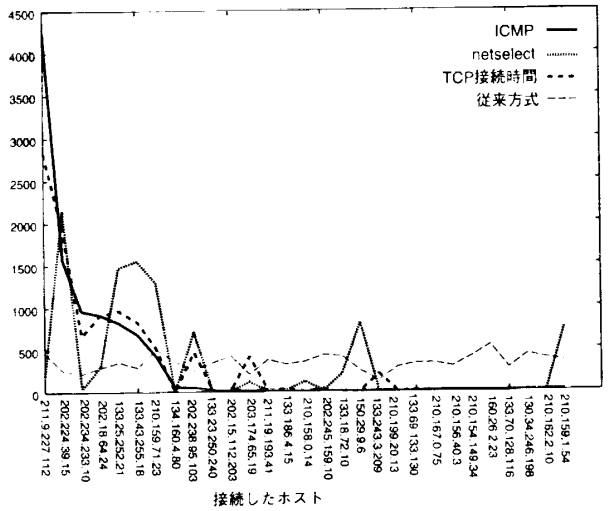


**図 12** 多数ファイル転送時の転送回数比較
Fig. 12 Connected server in each method when transfering many files.

が，従来方式と提案方式のファイル平均転送速度の差が大きい．これは後者の実験ではサーバへの接続回数が多いため，従来方式の場合に，ネットワーク上で遠いサーバに接続する回数が増加することに起因する．

## 6. まとめ

本論文では，ミラーサーバシステムなどの複数サーバシステムにおいてクライアントにとって適切なサーバを選択する DNS フィルタ方式を提案した．また，クライアントから見て最も短い時間でファイルを転送するサーバを最適とするシステムを想定し，そのためのサーバ選択システムを 3 つ示すとともに，このシステムを実装し，評価実験を行うことで，提案方式のの有効性を確かめた．

今後の課題としては，提案システムによるアクセス時間へのオーバヘッドや負荷試験などを行うことがあげられる．また，さらに優れたホスト選択手法の開発も必要である．

さて，1 章でも述べたように，本論文の提案方式はクライアントにとって適切なサーバの選択を目指している．一方，本方式単独ではサーバの負荷分散を実現することはできない．すなわち，あるクライアントにとって有利になる選択は必ずしもサーバ側の負荷分散に貢献するとは限らない．また，クライアント側からはサーバの応答速度などは測定できるものの，サーバの負荷を正確に測定することは難しい．特に，DNS による名前解決だけではサーバが提供しているサービスを判別することができないことから，サービスに特化した負荷分散は困難である．この問題は，サーバの負荷状況を評価するシステムである DNS Balance などを DNS フィルタ方式と連携させることで解決することができる．これにより，サーバおよびクライアント双方にとってより良いシステムが構築できるものと思われる．

**謝辞** 本論文の執筆にあたり，本システムの評価に協力していただいた RingServer プロジェクトの管理者グループの皆様，そして，提案方式の実装に使用させていただいた，さまざまなソフトウェアの作者の方々に深く感謝します．

## 参考文献

1) Cardellini, V., Colajanni, M. and Yu, P.S.: Dynamic load balancing on Web-server systems, *IEEE Internet Computing*, Vol.3, No.3, pp.28–39 (1999).
2) Mockapetris, P.: Domain Names — Implementation and Specification, RFC1035 (1987).
3) 横田裕思，木村成伴，海老原義彦：DNS フィルタ方式によるミラーサーバ選択法の提案と実装，インターネットコンファレンス 2001, Vol.18, pp.121–130 (2001).
4) Mosedale, D., Foss, W. and McCool, R.: Lessons Learned Administrating Netscape's Internet Site, *IEEE Internet Computing*, Vol.1, No.2, pp.28–35 (1997).
5) Shimokawa, T., Yoshida, N. and Ushijima, K.: DNS-based Mechanism for Policy-added Server Selection, *Int'l Conf. on Advances in Infrastructure for Electronic Business, Science and Education on the Internet, CD-ROM* (2000). http://www.tenbin.org/
6) 横田裕思：分散環境向け負荷分散 DNS サーバの設計と大規模サーバシステムにおける性能評価，修士論文，筑波大学大学院修士課程理工学研究科 (2001).
7) 横田裕思：DNS Balance. http://www.netlab.is.tsukuba.ac.jp/~yokota/izumi/dns_balance/
8) Rekhter, Y. and Li, T.: A Border Gateway Protocol 4 (BGP-4), RFC1771 (1995).
9) まつもとゆきひろ：オブジェクト指向スクリプト言語 Ruby. http://www.ruby-lang.org/
10) Pennarun, A.: Netselect. http://www.worldvisions.ca/~apenwarr/netselect/
11) RingServer Project: RingServer Project. http://www.ring.gr.jp/

(平成 14 年 7 月 8 日受付)
(平成 14 年 12 月 3 日採録)

**横田 裕思**（学生会員）

昭和 50 年生．平成 13 年筑波大学大学院理工学研究科修士課程修了．同年より筑波大学大学院工学研究科博士課程．現在に至る．分散ネットワーク上の負荷分散の研究に従事．著書に『情報処理発見伝—やさしい UINX!』（共著，昭晃堂），『パソコン悠悠漢字術 2001』（共著，紀伊國屋書店）等．

**木村 成伴**（正会員）

昭和 42 年生．平成 2 年東北大学工学部情報工学科卒業．平成 7 年同大学大学院博士後期課程修了．同年筑波大学電子・情報工学系講師，平成 12 年同大学助教授．現在に至る．博士（情報科学）．プロセス代数，ネットワークプロトコル，通信システムの効率評価等に関する研究に従事．電子情報通信学会，ソフトウェア科学会，IEEE，ACM 各会員．

**海老原義彦**（正会員）

昭和 22 年生．昭和 45 年東北大学工学部電子工学科卒業．昭和 50 年同大学大学院博士課程単位取得退学．同年同大学助手，昭和 50 年筑波大学電子・情報工学系助手，昭和 60 年同助教授，平成 5 年同教授．平成 10 年より同大学術情報処理センター長，平成 12 年より同大学電子・情報工学系長．現在に至る．工学博士．コンピュータネットワークアーキテクチャ，デジタル通信システムの性能評価，および知的通信システムの研究等に従事．電子情報通信学会会員．