社団法人　情報処理学会　研究報告
IPSJ SIG Technical Reports

2004－QAI－13　(4)
2004／10／28

ルータクラスタにおける二重パケット処理冗長方式

狩野秀 一　鈴木一哉　地引昌弘

NEC システムプラットフォーム研究所　〒211-8666 神奈川県川崎市中原区下沼部 1753

E-mail:　{karino@da, kazuya@ax, jibiki@bx}.jp.nec.com

あらまし　ルータクラスタにおける二重パケット処理冗長方式を提案する。高機能ルータクラスタではフロー状態の冗長化が必要だが、従来方式では冗長化のための状態交換のオーバーヘッドが大きいという問題がある。本研究では、パケットを二重処理して状態を冗長化するために、状態交換のオーバーヘッドがかからない方式を提案する。また、メンバ数および故障回復時間をパラメータとして従来方式とスループット比較を行い、提案方式の優位性を検証する。

キーワード　クラスタ, ルータ, スケーラビリティ, 高可用性

Duplicate Redundancy and Packet Processing Method in Router Clustes

Shuichi KARINO,　Kazuya SUZUKI　and　Masahiro JIBIKI

NEC Sysytem Platforms Laboratories　1753, Shimonumabe, Nakahara-ku, Kawasaki, Kanagawa, Japan

E-mail:　{karino@da, kazuya@ax, jibiki@bx}.jp.nec.com

Abstract　A duplicate redundancy and packet processing method in router clusters is proposed. In advanced router clusters, redundancy of flow states is necessary. Conventional methods have a problem that state exchange overhead for redundancy is high. We propose a method that has no state exchange overhead. In this method flow states are redupicated with duplicate packet processing. We have evaluated throughput dependent on node number and failover processing time to validate the efficiency of our method.

Keyword　Clustering, Router, Scalability, High-Availability

1. はじめに

近年インターネットがインフラ化し、ネットワークへの要求が増すにつれ、ルータに付加機能が搭載されることが多くなってきている。とくに、エッジのネットワークと ISP の接続点や、エンドノードが接続されるアクセスネットワークのルータは、VPN の終端やファイアウォールなどの高度な機能が搭載されている。

これらの機能は、パケット通過によって参照・更新される状態をルータ内部に保持する必要がある。障害によりそれらの状態が失われると、該当状態を参照・更新するパケットが通過できなくなることがあり、TCP などの上位層プロトコルの性能や、アプリケーションの動作に大きな支障を来たすリスクが高くなっている。このため、従来のルータと比較して、より高い可用性が必要とされている。またこれらの機能では、従来のルータでのパケット転送処理と比較して処理量が多いため、複数装置による分散処理により性能をスケールさせる必要がある。

可用性と負荷分散を同時に実現する技術としてクラスタリングが知られている。ルータにおいてもクラスタリングによる冗長化とスケーラビリティの確保が提案されているが、従来の方式には効率上の問題があるため、本稿では、効率的に負荷を分散させ、かつ冗長構成を実現できる、次の特長を持つルータクラスタリング方式を提案する。

状態交換なしの冗長化　ブロードキャストディスパッチを利用し、現用・予備で独立に状態を更新するため効率的に状態を冗長化することが可能である。各クラスタメンバがトラフィックの異なる部分を担当して分散処理する従来の方式と比較して、メンバ数の増加による通信オーバーヘッドを大幅に削減できる。

常時状態冗長化　状態の同期に時間がかからないため、一定間隔で更新状態を交換する従来の方式と比較して、高速なフェイルオーバが可能である。

これらにより、メンバ数についてスケーラブルであり、かつ高速な復旧を可能にした。

また本稿では、パケット送受信および制御通信の処

理量により、提案方式と従来方式の性能を分析し、実測値に基づいて定量的な性能見積もりを行う。上記により提案方式の優位性を検証した結果、故障回復50msec,メンバ数16以上の場合において、提案方式のクラスタの性能が従来方式を上回ることを示す。

2. ルータクラスタリング

2.1. ステートフルなパケット転送

企業網と外部網の境界や、キャリア網のアクセスリンクを収容するルータには、各種のアクセスサービスの収容や、セキュリティの確保の目的で、ステートフルファイアウォール[1]や、IPsec ESP[2]トンネル終端などの付加機能が搭載されている。また近年、不正アクセスやSPAMメール等の遮断すべきトラフィックが増加してきており、これらの機能の利用頻度が増えるとともに多機能化も進んでいる。

これらの機能は、パケットの属するフローを識別し、フローごとに内部に状態を保持し、かつパケット通過時に同状態の参照・更新を行うような処理が必要になる。たとえば、TCPステートフルファイアウォールでは、観測されるパケットから計算できる送信側および受信側ウインドウの範囲から外れたセグメントは転送しないよう、コネクション情報を保持している。IPsecトンネルの終端処理では、再送攻撃を防止するため、同一シーケンス番号をもつパケットは破棄する必要がある。これらの機能を実現するには、パケット通過のたびにフロー状態を更新・参照する、ステートフルな処理が必要である。

ステートフルなパケット転送処理は、フローごとに状態を保持し、パケット転送時にそれを検索・更新する必要があるため、従来のルータにおけるパケット転送処理と比較してコストが高い。このためある程度性能を向上させるには分散処理を行う必要がある。

また状態が失われると、既設フローの情報が失われ、同フローのパケットが通過できなくなる。ステートフルなパケット転送処理装置が故障すると、一般にセッションを張り直さない限り復旧しないため、エンドノードやアプリケーションへの影響が大きい。セッションの状態はエンドノードの送信するパケットにより作成・更新されるため、一度失われると復旧が困難であり、状態の複製を保持するなどして、冗長化する必要がある。

2.2. ルータクラスタリング

ハイエンドルータ装置では、複数のコンポーネントによる分散処理と冗長構成を行えるのが一般的である。同装置はスイッチファブリックにラインカードやルーティングカードを複数枚接続して冗長化や負荷分散を行う[3]。ただし、この構成には次の問題がある。

1. 一般にハードウエアを専用に開発する必要があるため高価であり、中小規模のルータ装置には採用するのは困難である。
2. ハードウエアの構成に合わせてソフトウエアを構成する必要があるため、ファイアウォールなどの付加機能を実現する既存のソフトウエアが分散化・冗長化に対応していない場合、コンポーネント間で機能を分離するように大規模に変更する必要があり、既存ソフトウエアの再利用が困難である。

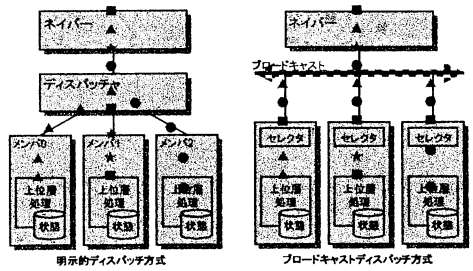

図 1: クラスタの二つのディスパッチ方式

一方、単独で動作する小規模な装置を複数利用し、負荷分散と冗長化を同時に満たす手法として、クラスタリングが知られている。ルータクラスタは、トラフィックの分配方式で、明示的なディスパッチ方式とブロードキャストディスパッチ方式に大別できる。

明示的なディスパッチ方式 ネイバーとの間にディスパッチャを置き、ディスパッチャにより負荷の調整と死活監視を行う(図 1左)。

ブロードキャストディスパッチ方式 ネイバーと、クラスタを構成する各メンバをイーサネットなどのマルチアクセスデータリンクに接続し、同データリンク上でネイバーがパケットをブロードキャストし、各メンバがその一部ずつを拾得して処理する(図 1右)。

このうち明示的ディスパッチ方式は、ディスパッチャがボトルネックになる可能性があり、また単一障害点となるのを避けるためにディスパッチャを冗長化する必要があるため構成が複雑になる問題がある。このため、近年ブロードキャストディスパッチ方式のルータクラスタが提案されている[4]。

従来のブロードキャストルータクラスタである多重冗長・状態交換方式の構成を図 2に示す。各メンバにはパケットセレクタがあり、同セレクタにてブロードキャストされたパケットを選択して拾得し、後段の上位層処理に回す。

パケットの選択はパケットのヘッダ等から計算するハッシュ値にて行う。同ハッシュ値はメンバごとに異なる値になるように割り当てられる。図 2では各メ

ンバにハッシュ値を1つずつ割り当てているが、分割を細かくして1台に複数のハッシュ値を割り当てるようにしてもよい。転送対象のパケットは、該当するハッシュ値を割り当てられたセレクタを持ついずれか1台のメンバで処理される。

各メンバは、自身の処理範囲のフローについて、フロー状態を作成・維持する。この処理状態を冗長化するため、各メンバは状態情報を互いに交換する。具体的には、所定の同期間隔、たとえば 50msec ごとに、以前の状態交換の時点から更新されている状態を、他のメンバに通知する。この手順により、各メンバは同期間隔以前の他のメンバの状態を保持することができる。

クラスタ内には、負荷バランスを制御するマスターメンバがおり、マスターメンバは他のメンバの負荷情報を監視し、ハッシュ値の割り当てを制御する。マスターメンバは他のメンバの死活を監視し、いずれかのメンバの故障を検知すれば、他のメンバに故障メンバの担当処理範囲を割り当て直す。これにより、故障回復が実現される。

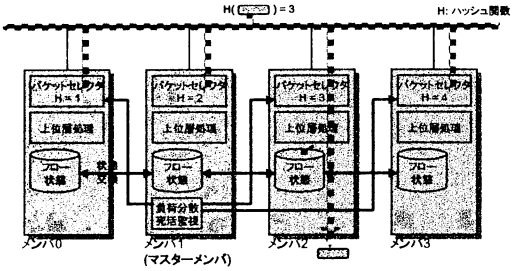

図 2: 状態交換・多重冗長方式の構成

2.3. 問題点

前節の手順によりフロー状態の冗長化と分散処理が可能となるが、従来方式では、次のように状態交換のオーバーヘッドが大きいため、メンバの増設および故障回復時間の短縮を行うのが難しいという問題が存在する。

- 従来方式では、各メンバが状態を共有するため、所定の同期間隔の期間内に更新されたフロー状態を相互に交換する。たとえば図 2のメンバ2は、パケット通過により更新された自身の状態情報を他のメンバに通知する必要がある。メンバを増設してスループットを上げようとすれば、クラスタ全体で処理するフロー状態数は増加することになる。メンバ数が増加するほど状態交換が増加するため、スループットの増加率はメンバ数の増加により低下すると考えられる。
- 故障回復を短時間で行うためには、その期間より短い間隔で状態を同期させている必要がある。図2においてメンバ 2 の処理を他のメンバが引き継ぐには、故障前の状態を受領している必要があり、故障回復時間よりも短い間隔で、メンバ間での状態交換を行う必要がある。状態交換を頻繁に行うと、そのオーバーヘッドのためにデータパケットの転送処理能力は低下すると考えられる。

これらより、スループット向上および故障回復時間の短縮を効率的に行うことができないことが予想される。

3. 二重パケット処理冗長方式

上記の問題を解決するため、本稿では二重パケット処理冗長方式を提案する。本方式では、ブロードキャストされるパケットを現用・予備の2台で独立に処理することにより、メンバ間の通信量を従来方式より大幅に削減する。これにより、多数のメンバによる負荷の分散と、短時間での故障回復が可能となる。

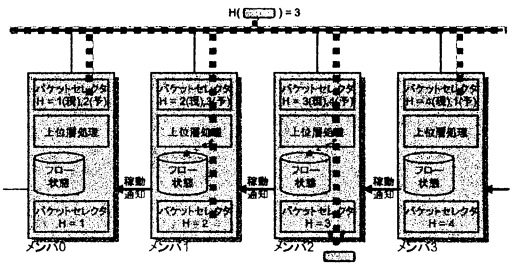

図 3: 二重パケット処理冗長方式の構成

3.1. 二重パケット転送処理

従来方式では、あるハッシュ値で表されるトラフィックの範囲を単一のメンバにより処理していたが、本方式では、ハッシュ値ごとにあらかじめ現用メンバと予備メンバを割り当てる。また、各メンバに受信画だけでなく送出側のパケットセレクタも用意する。予備メンバでは、パケット受信側のセレクタで予備処理に割り当てされたハッシュ値のパケットを拾得し、該当パケットの上位層処理を行う。その後、パケットをインタフェースより送出する直前に、送出側のセレクタにて、予備処理に該当するパケットのみを破棄する。現用メンバでは、従来と同様にパケットを拾得・処理・送出する。

これにより、予備メンバは現用メンバと同様にパケットを拾得・処理することができるため、現用メンバと同一のフロー状態を保持・更新できることになる。

たとえば図 3では、ハッシュ値3に該当するパケットはメンバ1および2で処理され、各々のメンバがもつフロー状態が独立に更新される。ハッシュ値3にか

んする予備であるメンバ1では、送出側のフィルタでパケットを破棄するため、パケットが重複して送出されることはない。

3.2. 死活監視と故障回復

予備処理を担当するメンバは、現用メンバの死活を監視し、現用メンバの故障を検知すると、自身の担当している予備処理を現用処理に切り替える。死活監視手順は、現用メンバが所定の間隔で送信する稼動通知パケットの監視などにより行えるため、現用と予備のメンバ間の通信コストは小さい。よって故障回復を速めるために稼動通知を頻繁に交換することができ、高速な復旧が可能となる。

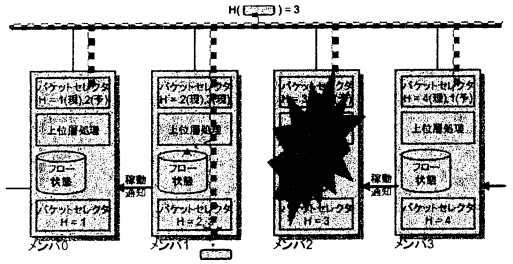

図 4: 二重パケット処理冗長方式での故障回復

図 3に示す構成の場合は、メンバ2はメンバ1に稼動通知を送信しており、それが一定時間受信されないとメンバ1は送出側のセレクタにハッシュ値3を追加して、故障回復処理を行う。故障回復後の状態を図 4に示す。

図 4の例では、故障回復後にはハッシュ値 3,4 の冗長度がなくなるため、低負荷のメンバに予備処理を割り当てる。トラフィックを細かく分割してハッシュ値割り当てを行うことにより、特定のメンバに負荷が偏ることはなく、故障回復後も分散処理を継続できる。

3.3. パケット転送処理負荷

ここで、提案方式では、全トラフィックを現用, 予備の2台のメンバで処理する必要があるため、パケット転送時の上位層処理のコストは、クラスタ全体で見れば従来方式に比して約2倍になる。ただし状態同期のための通信が不要であるため、メンバの増設によるスループットの向上や、故障回復期間の短縮を行うと、従来方式より高い性能を発揮するものと期待できる。

3.4. 実装

提案方式のクラスタを NetBSD 1.6.1 上に実装した。実装の概略を図 5に示す。図にある通り、カーネル内のイーサネット共通ルーチンにパケットセレクタを組み込み、IP 派毛とフィルタの実装である ipfilter と連携して動作するよう実装した。本実装により、提案方式の負荷分散および状態冗長化が動作することを確認した。

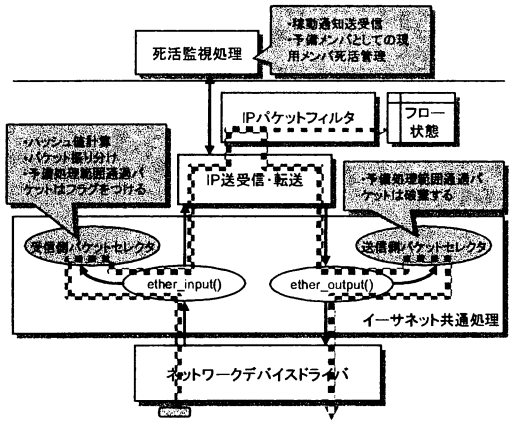

図 5: 実装の概略

4. 評価

多重冗長・状態交換方式(従来方式)は、メンバ数の増加および状態更新間隔の短縮により、状態交換のコストが増加する問題がある。二重パケット転送冗長方式(提案方式)では、状態交換のための通信は不要だが、全トラフィックを現用, 予備の2台のメンバで処理するため、パケット転送時の上位層処理のコストが従来方式に比して2倍になる。メンバ数および同期間隔の値により、従来方式と提案方式がどのような性能を示すかを分析し、実測値に基づいた比較を行う。

4.1. 性能モデル

クラスタにおける同期通信およびパケット転送処理のコストを比較するため、機能毎に処理を分割してそのコストを見積もる性能モデルを導入する。見積もりに利用する各パラメータを表 1に示す。ここでは、同一性能の各メンバが均等にトラフィックを分散して処理していると仮定する。具体的には、スループット t (pps)のフロー f 本を、各メンバで処理するものとする。

表 1: 性能パラメータ

項目名	単位	意味
n	台	メンバ数
f	本	メンバ当たりの処理フロー数
t	pps	フローあたりのトラフィック
s	cycles	フロー状態送信コスト
r	cycles	フロー状態受信・マージのコスト
c	cycles /packet	パケット転送・上位総処理コスト
k	cycles	稼動通知送受信コスト
q	秒	故障回復時間
C	cycles /秒	メンバの CPU 処理能力

4.1.1. 従来方式の処理コスト

クラスタの各メンバが保持するフロー状態のうち、ある期間に更新される状態数は、パケットが通過したフローの数 f と等しい。このとき、状態交換のオーバーヘッドは次のように表せる。フロー状態は一般にハッシュ表として実装されているため、その参照・更新等のコストはフロー数によらずほぼ一定と仮定する。自身の状態ひとつを他のメンバに送信するコストと、他のメンバからの状態更新ひとつを受信し、自身の保持する状態にマージするコストを CPU クロックサイクルの消費量で表現し、各々 s, r (cycles) とすると、自身のフロー状態を他のメンバに送信し、他の全メンバよりフロー状態を受け取るため、状態交換コストは $fs + (n - 1)fr$ (cycles) である。

また各メンバでの転送対象パケット量は、フローごとのスループット t ,フロー数 f を用いて ft (pps) と書ける。パケットごとの転送および上位層処理コストを c (cycles/packet) とすると、各メンバでの単位時間あたりのパケット転送コストは cft (cycles/sec) となる。

これより、メンバあたりの総処理量は、状態交換の間隔を q (sec) とすれば、

$$cft + \frac{fs + (n-1)fr}{q} \quad \text{(cycles/sec)}$$

となる。

各メンバの処理能力を C (cycles/sec) とすると、処理を能力一杯まで行うときは、

$$C = cft + \frac{fs + (n-1)fr}{q}$$

が成り立つ。f について上式を解くと、1 台あたりの処理可能フロー数は次のように表せる。

$$f = \frac{qC}{ctq + s + (n-1)r}$$

クラスタの総スループット T_c は、各メンバのフロー処理能力, フロー毎のスループット, メンバ数の積をとればよいから、

$$T_c = nft = \frac{nqtC}{ctq + s + (n-1)r} \quad \text{(pps)}$$

となる。

4.1.2. 二重パケット処理冗長方式の処理コスト

現用と予備のパケット転送処理のために、提案方式では従来方式の倍のパケットを処理する必要がある。本稿では、パケット転送のコストはほとんどが上位層の処理に費やされるような用途を想定しているため、予備処理でのパケット破棄処理はパケット転送処理と比べてコストが十分低いと仮定できる。すると 1 台あたりの処理量は

$2cft$ (cycles/sec)

となる。

また、故障回復時間を q, 稼動通知 5 回消失にて故障検知とすると、故障回復時間 q 内に少なくとも 5 個の稼動通知を送受信する必要があるため、1 秒あたりの稼動通知送受信の回数は $5/q$ である。死活監視のための通信コスト(送信・受信含め)を 1 回あたり k とすれば、死活監視のコストは $5k/q$ (cycles/sec)となる。メンバの処理能力を C とすると、能力一杯まで処理する場合は、

$$C = 2eft + 5\frac{k}{q}$$

となる。これを、従来方式の場合と同様に f について解けば、クラスタの総スループット T_p は次のように表せる。

$$T_p = nft = \frac{n(qC - 5k)}{2cq} \quad \text{(pps)}$$

4.2. 評価パラメータの実測

上記の性能モデルに基づいて従来方式と提案方式の性能を比較するため、前節で示した各パラメータのうち、CPU 消費クロックサイクルを実測した。

測定は次のように行った。ハードウエアとしてインテルの IA-32 プロセッサを利用し、同プロセッサが具備するクロックサイクルカウンタにより、先に示した各処理に該当する区間の命令を実行するのに要したクロックサイクルを算出した。

- 提案方式については、3.4節で記した実装を用いて測定を行った。　死活監視用の稼動通知パケットの送受信処理のうち、送受信ルーチンから　デバイスドライバまでの処理コストを測定した。
- 従来方式については、入手しやすい OpenBSD[5] の pfsync を用いて　状態交換コストを測定した。状態送信側においては、状態送信のためのデータ蓄積, IP パケットへのカプセル化, プロトコルスタックからデバイスドライバまでの、パケットの送出処理のコストを測定し、更新メッセージの個数で消費クロックを割って状態更新の送信コストを算出した。
  状態受信側についても、デバイスドライバにてパケットがメモリに読み込まれてから、同期通信処理が終了するまでのコストを同様に測定した。
- パケット転送コストについては提案方式の実装を用いて、NetBSD 1.6.1 上でパケットが受信側デバイスドライバに入ってから、送出側デバイスドライバから出るまでのコストを測定した。

いずれも測定には Pentium3 500MHz のマシンを利用し、パケット転送コストと従来方式の状態交換には、片方向 25pps の TCP フローを利用した。

表 2に上記各項目の測定結果を示す。

表 2: CPU 消費クロックの実測値

項目	消費クロックサイクル
パケット転送処理(c)	28278.02
フロー状態送信(s)	4628.48
フロー状態受信(r)	2119.96
稼動通知送受信(k)	28956.88

4.3. 有効性の検証

先に示した性能モデルに上記の測定結果をあてはめると、メンバ数と状態同期,死活監視間隔についてクラスタの総スループットは次のようになる。1 フローあたりのスループット t を平均 25pps, クラスタメンバの動作クロック周波数 C を 1GHz として計算した。このとき、故障回復時間 q を 50msec に固定してメンバ数 n を変化させると、クラスタの総スループットは図 6のようになる。また、メンバ数を 16 に固定して故障回復時間を変化させた場合、スループットは図 7のようになる。

これより、従来方式では、メンバ数の増加により、総スループットの増加率が低下している様子が見受けられるが、提案方式ではほぼメンバ数に比例してスループットが向上しており、メンバ数 16 程度で、提案方式のクラスタの性能が従来方式を上回ることがわかる。

また、故障回復時間については、従来方式では 1sec から 50msec まで、間隔が短くなるにつれて性能が急激に劣化するのに対し、提案方式ではほとんど劣化が見られないことがわかる。

これらより、実際に PC を利用したルータクラスタでも、メンバ数 16 台以上, 故障回復 50msec 程度の条件では、提案方式の性能が従来方式を上回ることが期待できる。

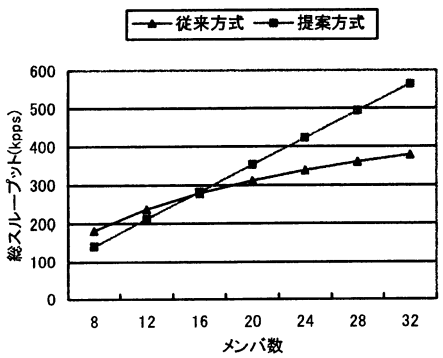

図 6: メンバ数を変数とした性能比較

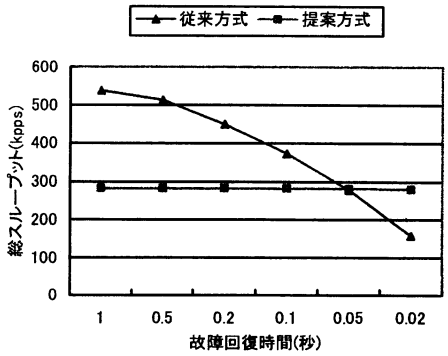

図 7: 故障回復時間を変数とした性能比較

5. むすび

本稿では、スケーラブルで高速復旧可能なルータクラスタリング方式として、従来の状態交換・多重冗長方式では、メンバ間でフロー状態を冗長化するために、一定間隔での状態の交換が必要であり、そのためにメンバ数の増加と故障回復時間の短縮を、低いオーバーヘッドでは実現できないという問題が存在した。提案方式はブロードキャストディスパッチされたパケットを現用メンバと予備メンバで独立に処理し、フロー状態を維持するため、状態交換のオーバーヘッドがかからない。これによりメンバ数の増加や故障回復時間の短縮を、少ないオーバーヘッドで可能にした。

また本稿では、従来方式と提案方式のクラスタリング方式について性能モデルをたて、その各パラメータを実測することにより、メンバ数および故障回復間隔を変化させた場合の予測性能の定量的比較を行った。この結果、故障回復間隔 50msec, メンバ数 16 以上で提案方式のスループットが従来方式を上回る結果がえられた。

文　献

[1] Guido van Rooij, Real Stateful TCP Packet Filtering in IP Filter, SANE 2000.

[2] Kent, S., and R. Atkinson, IP Encapsulating Security Payload (ESP), RFC 2406, November 1998.

[3] Requirements for Next-Generation Core Routing Systems, Cisco Carrier Routing System White Paper, http://www.cisco.com/warp/public/cc/pd/rt/12000/clc/prodlit/reqng_wp.pdf

[4] K. A. Adelman, D. L. Kashtan, W. L. Palter and D. D. Piper II, Method and apparatus for a TCP/IP load balancing and failover process in an internet protocol (IP) network clustering system, U.S. Patent 6,078,957, June, 2000.

[5] OpenBSD, http://www.openbsd.org