# みかん - ミラー選択機能付き代理 FTP サーバ

尾藤 正人 † 舟阪 淳一 ‡
† 株式会社ホライズン・デジタル・エンタープライズ ‡ 広島市立大学 情報科学部

## 概要

FTP サーバに対する負荷の集中を緩和するために, ミラーサーバの設置による負荷分散が行われている. しかし従来の方式では, どのサーバが近いのか, 更新が遅れてないか, どこのディレクトリにミラーされているのか等が不明であり, ユーザが適切なミラーサーバを選択することができなかった. そこで本論文では, ミラーサーバの自動選択を行う代理 FTP サーバ「みかん」を提案し, 実装する. また実際に「みかん」を用いて, 評価を行う.

## 1 はじめに

ファイルを一般に配布する技術として, FTP(File Transfer Protocol) [1] が広く利用されている. また, トラフィックの集中を避けるため, マスターサーバと同一ツリーを提供するミラーサーバも広く利用されている.

このような複数のミラーサーバを選択する技術が既にいくつか提案されている. しかし, 既存の技術ではミラーサーバの選択を柔軟に行うことができず, 結果としてその技術が利用されず, マスターサーバの負荷が増加してしまう現象が発生している.

そこで本論文では, ミラーサーバ選択機能付き代理 FTP サーバ「みかん」を提案する. 「みかん」を使用することで従来のクライアントを利用しつつ, 最新ファイルを近傍のミラーサーバから取得することができるようになる.

以下, 2 章では研究背景についてミラーサーバを利用する上での問題点を挙げる. 3 章では関連研究とその問題点について述べる. 4 章では, ミラーサーバの更新状況についての調査結果を示す. その結果従来方式の問題点を指摘することができた. 5 章では, 従来方式の問題点を解決するために, ミラーサーバ選択機能つき代理 FTP サーバ「みかん」を提案する. 6 章では, 実際に実装した「みかん」の実装体系について述べる. 7 章では, 「みかん」の評価結果を示す. その結果, 「みかん」の有用性を示すことができた. 最後に 8 章では, 全体のまとめと今後の課題を示す.

## 2 背景

本章では, ミラーサーバをユーザが利用する場合にどのような問題点があるかを述べる.

ミラーサーバを利用する際, あらかじめユーザは情報を収集しておかなければならない. この情報収集は, 一般的に手動で行われるため手順が面倒である. 以下にユーザが収集しなければならない情報を述べる.

- ミラーサーバの存在
  通常, ミラーサーバの存在は, WWW, 電子メール, ネットニュース等の媒体でアナウンスされる. これらのアナウンスは全て手動で行われている. ミラーサーバの存在を確認する技術は存在しないため, ユーザが手動でその存在を確認しなければならない.
- 何をミラーしているのか
  ミラーサーバは複数のサーバのミラーを行っていることが多い. しかし, ディスク容量やネットワーク帯域の制限から, 全てのミラーを行うことはできない. ユーザは何がミラーされているかあらかじめ調べておく必要がある.
- ミラーしているディレクトリ

ミラーサーバがミラーを行うディレクトリは, ミラーサーバの管理者に委ねられている. そのためユーザは, あらかじめどのディレクトリにミラーが行われているかを調べなければならない.

- ネットワーク的な距離
  ネットワーク的な距離は, 経由するルータの数や帯域で決定される. たとえ地理的に近くても, ネットワーク的に遠いことは十分に考えられる. また, ネットワーク上のトラフィックは刻一刻変化するため, ネットワーク的な距離も刻一刻変化する. そのため, ネットワーク的な距離を把握するのはかなり困難である.
- ミラーサーバの更新遅延
  ミラーサーバはある一定の間隔で更新を行うので, 更新間隔程度の遅延が生じてしまう. また, ミラーサーバの管理者の設定ミスや, ディスク容量以上のファイルの流入等の原因によって, 更新が止まってしまうこともある. よって最新のファイルを入手するためには, 1 度マスターサーバにアクセスし比較を行って, 最新のファイルであることを確認しなければならない.

以上に述べたようにミラーサーバの利用には様々な問題が存在する. ユーザにしてみれば, マスターサーバにアクセスすれば確実に最新ファイルを入手することができる. そのためミラーサーバが利用されず, マスターサーバに負荷が集中する現象が発生している.

このように, ミラーサーバは効率良く利用されていない. そこで現在, サーバを自動選択する技術が注目されている.

## 3 関連研究

ミラーサーバを用いて分散化されたサーバにアクセスする場合, サーバを選択する必要がある. この選択の実現方法は, 既にいくつか提案されてきている. これらの関連研究は前提の違いにより大きく二つに分けることができる.

まず, 前提として選択対象の全てのミラーサーバが同じディレクトリ構造を持つような管理ポリシーを考える. 例えば RingServer Project[*1] のような運用形態である. この場合, ミラーサーバの保持している内容の相違は想定していないので, DNS を用いて複数のホストから最適なホストを選択する方法や IPv6 Anycast による方法が可能である. 例として, Tenbin [2] [3], DNS Balance [4], DNS Trick [5] [6] が挙げられる. しかし実際にはミラーの更新には遅延が発生しており, ソフトウェアの発展に伴う配布ファイルの増加は, サーバ間の同期をますます悪化させていくと予想される.

次に, 前提として選択対象のミラーサーバがそれぞれ独自のディレクトリ構造を持つことを考える. この場合, 必要な配布ファイルのみをミラーすることができ, 同期のずれもあまり大きくならないことが期待される. また管理ポリシの異なるサーバを含んでよいので, より多くのミラーサーバを選択の対象とすることができる. ただし DNS や IPv6 Anycast による方法は使えず, 各ミラーサーバの内容を確認して選択する方法が必要となる. これは, 同期のずれが発生していた場合にも有効である.

このような同一性の保証されないミラーサーバからファイルを選択する方法としては, まずクライアントにおける実装がある. 例として, GetRight [7], apt[*2], Smart Client [8] が挙げられる. この方法では, 各ユーザが各マスターサーバに対して, ミラーサーバ存在と更新状況を調査しておかなければならない. この情報を一元管理することができないため, 各ユーザ毎に調査のためのトラフィックが発生してしまう.

また中間サーバにおいてファイルを選択する方式としては, FTP Mirror Tracker [9] [10] が挙げられる. FTP Mirror Tracker は FTP サーバのディレクトリツリーを解析し, 各ディレクトリの内容を反映した一意の識別子 (MD5 [11]) を生成する. この識別子をもとに MySQL を用いてデータベースを構築する. ユーザから URL が入力されると, データベー

[*1] http://www.ring.gr.jp/
[*2] Debian; Advanced Packaging Tool

スを参照し, それと同じ中身を持つ複数の URL を返す. FTP Mirror Tracker はディレクトリごとの同一性しか判断できないため, ファイル単位で相違が生じているディレクトリは丸ごと利用できず非効率である. この問題の重要性については, 次章ミラーサーバ更新状況の調査によって明らかにする.

本研究では, 同一性の保証されないミラーサーバを, ファイル単位の非同期まで考慮して選択することを目標とする.

# 4 ミラーサーバの更新状況

ミラーサーバの更新状況を把握しておかなくては, 選択単位の異なる各自動選択技術の効果が不明である. そこで、 実際にインターネット上のミラーサーバに接続して調査した. 本章では調査方法と結果について述べる.

## 4.1 調査方法

Linux ディストリビューションの 1 つである Vine Linux[*3]のディレクトリツリーに対して, 11 台のミラーサーバ[*4]の更新状況を調査した. これらのミラーサーバは, Vine Linux の Web ページに掲載されているミラーサーバと RingServer Project のサーバの中から無作為に選択したものである.

ftp コマンドとして “ls -lR” を入力して, Vine Linux のディレクトリツリーの情報を収集し, 比較を行なった. Vine Linux では, マスターサーバが非公開であるので, 同期が早く, ミラーリングポリシー (後述) で最も多くのファイルを保存している北陸先端科学技術大学院大学 (ftp.jaist.ac.jp) と比較した.

調査結果を示す前に, まずはミラーリングポリシーを定義する. ミラーサーバがミラーを行なうときに, 先頭に “.” のつくファイルや, 最後に “~” のつくファイル等をミラーを行なう必要が無いと判断した場合, そのファイルに関してはミラーを行なわない場合がある. また, 圧縮方式の変換を行なっても支障がないと判断される場合, 圧縮方式の変換が行なわれる場合がある. これらのポリシーを本論文では, ミラーリングポリシーと呼ぶことにする.

[*3] http://vinelinux.org/

[*4] ftp.jaist.ac.jp, ftp.nuie.nagoya-u.ac.jp, ftp.ryukyu.ad.jp, ftp.ics.es.osaka-u.ac.jp, ftp.kddlabs.co.jp, ftp.osn.u-ryukyu.ac.jp, ftp.rpmlinux.com, ftp.riken.go.jp, SunSITE.sut.ac.jp, ring.asahi-net.or.jp, ring.iwate-pu.ac.jp

表 1: 11 台のミラーサーバの更新状況 (2002/6/3 現在)

| ディレクトリ | 同一 | 差異あり | ミラーなし |
|---|---|---|---|
| TestPkg | 4 | 3 | 4 |
| Vine-1.0 | 2 | 2 | 7 |
| Vine-1.1 | 2 | 2 | 7 |
| Vine-2.0 | 2 | 3 | 6 |
| Vine-2.1 | 2 | 9 | 0 |
| Vine-2.1.5 | 3 | 7 | 1 |
| Vine-2.5 | 7 | 1 | 3 |
| VinePlus | 4 | 6 | 1 |
| VineSeed | 2 | 8 | 1 |
| apt | 5 | 2 | 4 |

## 4.2 調査結果

表 1 に 11 台のミラーサーバに対しての調査結果を示す. 表 1 中の各列で同一, 差異あり, ミラーなしは, それぞれディレクトリが完全に同一な状態, ディレクトリに差異が生じている状態, ミラーが行われていない状態のサーバ数を表す. 各行は Vine Linux のディレクトリツリーに存在するディレクトリを表している.

表 1 を見ると, 多くのミラーサーバのディレクトリツリーに差異が生じていることが分かる. この差異はミラーリングポリシーによるものがほとんどであり, ほとんどのミラーサーバでは, 実用上同期がとれている.

これらのディレクトリの差異は, ユーザにとっては全く問題ないものであり, 同一内容であると判断された方が都合がよい. しかし, FTP Mirror Tracker のようにディレクトリの単位でサーバ選択を行なった場合, 違うものと判断されてしまう. ファイル単位の細かい粒度なら, 問題は発生しない.

# 5 「みかん」の提案

これまでに述べてきたように, 最新のファイルをネットワーク的に近いサーバから取得するという目的に対して, 従来の実装には様々な問題があった. そこで本論文ではこれらの問題を解決ために, ミラーサーバの選択機能を持った代理 FTP サーバである「みかん」を提案する. 以下では,「みかん」について詳しく述べ, その有用性を示す.

## 5.1 「みかん」の概要

ミラーされたファイルを選択する際の問題点をまとめると次のようになる.

- DNS や IPv6 Anycast も用いてサーバを選択する手法では, ミラーの更新遅延に対応することができない. また, ディレクトリ構造が同一でないといけないため, 限られた用途でしか使用することができない.
- 同一性の保証されないミラーサーバからファイルを選択する方法としては, クライアントにおけるがある. しかし, この実装だとクライアント毎に調査のためのトラフィックが発生してしまう. また, 専用クライアントを導入する必要があり, 普及に難がある.
- 中間サーバにおける実装として FTP Mirror Tracker が挙げられるが, FTP Mirror Tracker はディレクトリ単位でしかサーバ選択を行うことができない. 柔軟にサーバ選択を行うためには, ファイル単位での細かい粒度のサーバ選択が必要である.

次に, 用途, 利用形態を以下のようなものとする.

- 組織ネットワークの出入口に設置
  ここに設置することで, 組織内から組織外への FTP 要求を処理することができる. また更新状況の情報を共有することもできる.
- 取得ファイルに偏りがある
  組織ネットワークが取得要求を行う情報には偏りがあることが知られている. これは, FTP においても同様のことがいえる. これを仮定することにより, 更新状況の調査のためのトラフィックを減らすことができる.
- 完全には同一でないディレクトリツリーを想定
  Ring Server Project のミラーサーバでは, ディレクトリツリーが同一になっている. しかし, その他のミラーサーバはディレクトリツリーが各々異なる. 多くのミラーサーバをサーバ選択の対象とするため, 柔軟なサーバ選択の機構が必要である.

以上を踏まえた上で, 次のような特色を持つ「みかん」を提案する.

- 中間サーバにおける実装
  DNS における実装とクライアントにおける実装は, そもそも想定している用途が異なる. よって, 実装モデルは中間サーバにおける実装である. また, 中間サーバにおける実装は更新状況の状況を一元管理することができ, 無駄が少ない.
- ファイル単位の細かい粒度によるサーバ選択
  ファイル単位の細かい粒度でサーバ選択を行なうので, ディレクトリに僅かな差異が生じても的確にサーバ選択を行なうことができ, 最新ファイルを取得することができる.
- 従来のクライアントでアクセス可能
  クライアントとのやりとりを行なうためのプロトコルには, FTP を使用する. これにより, 従来の FTP クライアントを使用することができ, ユーザが新たなソフトウェアを導入する必要がなくなる.
- 更新状況の収集機能
  ディレクトリツリーを解析することにより, 更新状況の把握を行なう. この情報をもとにサーバ選択が行なわれる.
- ネットワーク状況の調査
  RTT(Round Trip Time) やスループットを調査することにより, ネットワーク的に近いサーバを調べる. この情報をもとにサーバ選択を行い, 最も近傍のミラーサーバからファイルを取得す

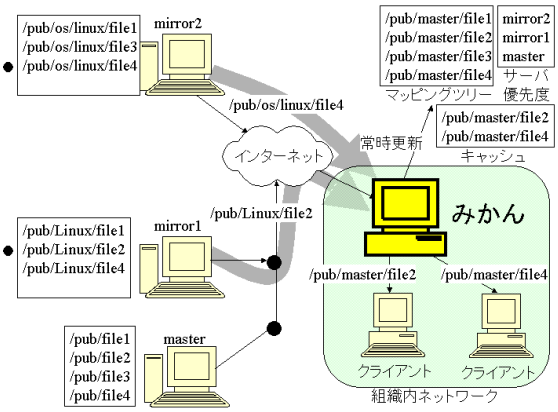


図 1: 「みかん」の動作の仕組

ることができる.

- マッピングツリー
  「みかん」はクライアントとのやりとりを全て FTP で行なうので, 自分自身が FTP サーバのように振るわなければならない. そのために仮想的なディレクトリツリー (マッピングツリー) を生成する. マッピングツリーは収集した更新状況を基に生成される.
- データベース構築機能
  更新状況やネットワーク状況等の収集したデータを, 次に使用するためにデータベース化しなければならない.
- FTP クライアント機能
  「みかん」は, ユーザからの取得要求を受け取るとサーバ選択を行ない, 代わりにファイルを取得する.
- キャッシング機能
  ユーザの代わりに取得してきたファイルを保存しておくと, 次回以降高速にファイルを転送することができる. また, インターネット上のトラフィックの減少にもなる.

## 5.2 「みかん」の仕組

図 1 に「みかん」の動作の仕組を示す. ここで, 「みかん」はある組織ネットワークの出入口に置き, マスターサーバに集中するアクセスをミラーサーバに振り分けるために用いる.

図 1 中の master, mirror1, mirror2 はそれぞれ FTP サーバで, master がマスターサーバ, mirror1, mirror2 がミラーサーバである. それぞれのサーバの左枠には, サーバが所持しているファイルを示している. サーバ優先度は, 高い方から順に mirror2, mirror1, master となっている. 優先度は, 例えばネットワーク往復時間 (RTT) の短い順に高くしておく. 「みかん」はマスターサーバのツリー構造をあらかじめマッピングしておく. サーバ優先度とマッピングツリーは, 定期的なモニタリングにより, 最新の情報に更新しておく.

例えば図 1 においてクライアントが「みかん」に接続して, /pub/master/file2 を取得しようとしたとする. 「みかん」はミラーサーバの更新状況とサーバ優先度により, mirror1 から/pub/Linux/file2 を取得する. 取得したファイルはキャッシュされ, クライアントに返される.

「みかん」とクライアント間のやり取りは, 拡張のない FTP であるので, クライアントは従来の FTP クライアントでよい. また, クライアントには「みかん」がマッピングしているツリー構造しか見えないので, 「みかん」からファイルを取得したようにしか見えない. よってクライアントはネットワーク状況を把握する必要が無く, 「みかん」を最寄りのミラーサーバとみなして利用することができる.

## 5.3 提案方式の問題点

問題点を以下に挙げる.

- サーバの負荷は間接的にしかわからない
  アクセスを行うミラーサーバの負荷がどの程度のものかを判断するための機構がない. しかし, ファイルの取得時間を計測することにより, 負荷の高いサーバは選ばれにくくなると考えられる.
- オーバーヘッド
  「みかん」の機能を利用するためには, 必ず「みかん」を経由しなければならない. よってどうしてもオーバーヘッドが発生してしまう. これに関しては, 第 7 章の評価で実用上問題ないこ

とを示す.

- ディレクトリ単位より処理が複雑
  ファイル単位の細かい粒度で処理を行うので, どうしても複雑になってしまう. しかし, ディレクトリ単位でのサーバ選択には重要な問題があることは既に示した. ファイル単位でのサーバ選択の方がより利点が大きい.
- ディレクトリの対応関係をユーザが知らなければならない
  「みかん」は, マスターサーバのディレクトリツリーを仮想的なマッピングツリーにマッピングする. そのためユーザは, どのマスターサーバのどのディレクトリが, マッピングツリーのどのディレクトリに対応するかを知らなければならない. しかしこの問題は, 現在においてミラーサーバを利用する際に発生する問題と同じである. よってユーザの手間が増えることにはつながらないであろう.
- 耐故障性の低下
  「みかん」を利用する場合は, 外部ミラーサーバに向けての FTP 接続がすべて「みかん」を経由する. このため, 「みかん」に障害が起きると外部への FTP 接続が不可能になる. すなわち, 耐故障性の低下が考えられる. これは, 複数台のサーバを導入することで回避することができる. 複数台のサーバは独立に運用することも, 協調して運用することも考えられる. 協調する場合には保持データの整合性等について検討する必要がある.

# 6 「みかん」の実装

本論文で提案した「みかん」を実際に実装した [12]. 以下の環境で動作を確認している.

- Vine Linux 2.5
- FreeBSD 4.5-RELEASE
- FreeBSD 4.6-RELEASE

以下では「みかん」をどのように実装したのかについて詳しく述べる. 「みかん」の実装体系を図 2 に示す.「みかん」は 2 つのプログラム, mikan と mkdb で構成される.

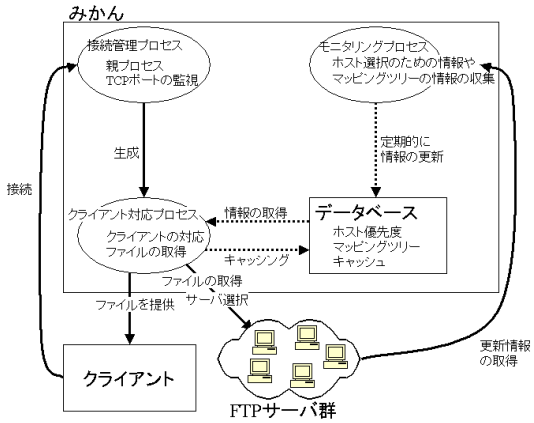


図 2: 「みかん」の実装体系

## 6.1 mkdb

データベースの生成, 更新を行うプログラムである. 図 2 のモニタリングプロセスに該当する. 実行するとマスターサーバのディレクトリツリーを走査し, 仮想的なディレクトリツリーであるマッピングツリーの更新を行う. マッピングツリーは, mikan がクライアントに対して FTP サービスを提供するときに参照するディレクトリツリーのデータベースである.

mkdb はマスターサーバ上のディレクトリ構造をそのまま再現する. ディレクトリ内にあるファイルの情報は, “.dir_info” というディレクトリ内に保存する. “.dir_info” はそのディレクトリに関する情報を保存するために使用し, FTP クライアントに対しては, 表示を行わないようにする.

mkdb はスケジューリングの機能をもっておらず, データベースの更新を行なうと終了してしまう. そのため, 定期的に更新を行なうためには cron 等でスケージューリングしなければならない.

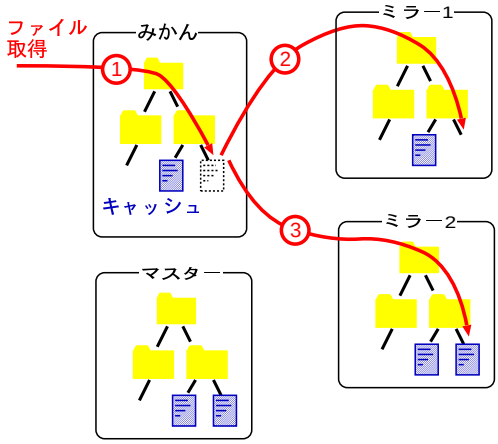


図 3: みかんのファイル取得方法

### 6.2 mikan

「みかん」のデーモンプログラムである．図 2 の接続管理プロセスとクライアント対応プロセスから成る．実行すると接続管理プロセスとなり，TCP ポートの監視を行なう．TCP ポートにクライアントから接続があった場合は，クライアント対応プロセスを生成し，再び TCP ポートの監視を行う．

クライアント対応プロセスは，クライアントと FTP でやりとりを行う．mkdb によって作成されたマッピングツリーを基に，ディレクトリサービスを提供する．この時，“.dir_info” はクライアントに対して表示しない．“.dir_info” は，「みかん」がディレクトリの情報を保存するために使用する，特別なディレクトリだからである．

クライアントからファイルの取得要求があった場合は，クライアントの代わりにファイルの取得を行い，取得したファイルをクライアントに返す．

mikan がファイルを取得する際の具体的な動作を図 3 に示す。mikan はクライアントからのファイル取得要求を受けつけると，ローカルファイルシステムにおける該当ファイルにアクセスする．既にキャッシュが存在すればファイルにアクセスできるので，その内容をすぐにクライアントに返すことができる．しかしみつからない場合は，その実体をミラーサーバから取得しなければならない (図中 1)．mikan はミラーサーバ優先順位に従って，対応するディレクトリにアクセスする (図中 2)．選択されたミラーサーバにおいてもみつからない場合は，次の候補にアクセスする (図中 3). 最終的に該当ファイルを発見すると，FTP により取得し，クライアントにそのファイルを転送する．また同時にローカルファイルシステムにはキャッシュとしてファイルを保存する．

一方，ファイル取得以外の FTP コマンドはできるだけ mikan において処理し，ミラーサーバへのアクセス数を低減する．

## 7 評価及び考察

本論文で実装した「みかん」について評価を行った．まず「みかん」を経由することによって生じると予想される処理時間 (オーバーヘッド) について実測し，従来通りの FTP の利用に影響がないことを確かめる．次に，実際に「みかん」を運用したときにユーザが認識できるメリットについて調べる．最後に，ミラーサーバを運用し全てのファイルを取得する際に生じるトラフィックと「みかん」を運用する際に必要なトラフィックを比較する．

### 7.1 オーバーヘッドの評価

まず，「みかん」を経由した場合のオーバーヘッドの評価を行った．10 個のファイル (合計約 15MB) を，直接 FTP サーバから取得した場合と，「みかん」経由で取得した場合の時間を測定した．測定の対象としたサーバは，ftp.jaist.ac.jp(以下 JAIST) と ring.atr.co.jp(以下 ATR) である．

ftp コマンドに自動的に一定の処理を行うスクリプトを作成し，そのスクリプトの実行に要する時間を測定した．測定は 1 時間毎に 24 時間行い，各測定においては 10 回繰り返した．

ATR での結果を図 4 に,JAIST での結果を図 5 に，示す．ここで，縦軸はファイル取得時間，横軸は 1 日での時間帯，各点は 10 回の平均値である．オーバヘッドの平均，直接 FTP サーバから取得した場合の平均と標準偏差をまとめたものを，表 2 に示す．

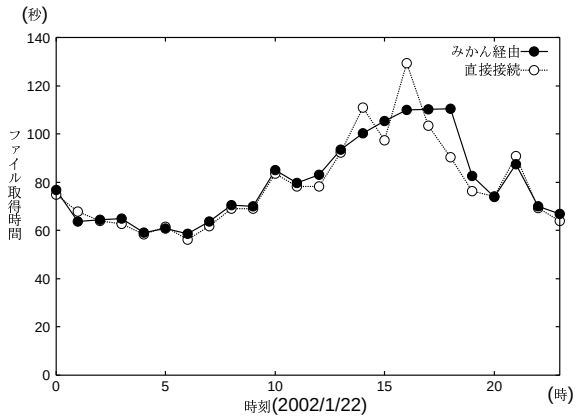


図 4: ATR からのファイル取得時間

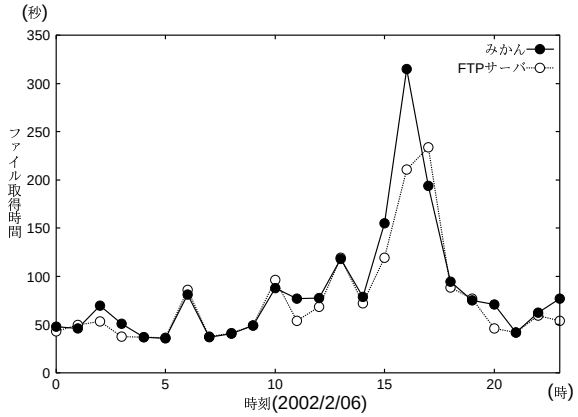


図 5: JAIST からのファイル取得時間

ファイルの取得時間の平均, 標準偏差とオーバーヘッドの平均と比較すると, オーバーヘッドは実用上ほぼ支障がないといえる [13] [14].

## 7.2 可用性の評価

「みかん」を運用した場合の効果について調べる. まず広島市立大学内で「みかん」を動作

表 2: 平均オーバーヘッドと平均取得時間と標準偏差

| サーバ名 | ATR | JAIST |
|---|---|---|
| 平均オーバーヘッド (秒) | 4.93 | 8.64 |
| 取得時間 (秒) | 78.9 | 75.5 |
| 標準偏差 (秒) | 34.1 | 50.4 |

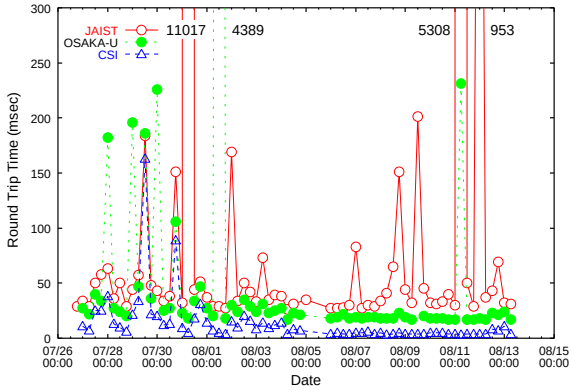


図 6: ネットワーク状況の調査結果

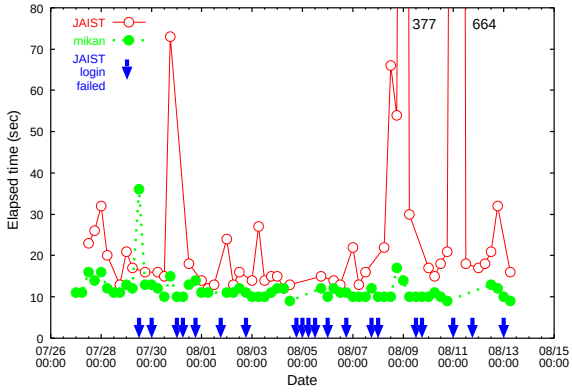


図 7: FTP サーバから直接取得した場合と「みかん」を経由した場合のファイル取得時間

させ, 3 つのミラーサーバ (ring.csi.ad.jp(以下 CSI), ftp.ics.es.osaka-u.ac.jp(以下 OSAKA-U), ftp.jaist.ac.jp(以下 JAIST)) を選択するようにした.

各ミラーサーバへの RTT を ping により計測した結果が図 6 である. この図をみると同一 AS 内の CSI への RTT が常に最短であり, このミラーサーバが常に選択されることがわかる.

動作させた「みかん」に対し, 6 時間おきに 10 個のファイル (約 7MByte) の取得に要する時間を計測した. 結果を図 7 に示す. また同時に直接 JAIST から取得した場合の所要時間も示した. この図をみると, 「みかん」を経由して CSI から取得する時間が常に JAIST から取得するより短いことがわかり, ミラーサーバ選択の有効性が確認できる. この場合, 「み

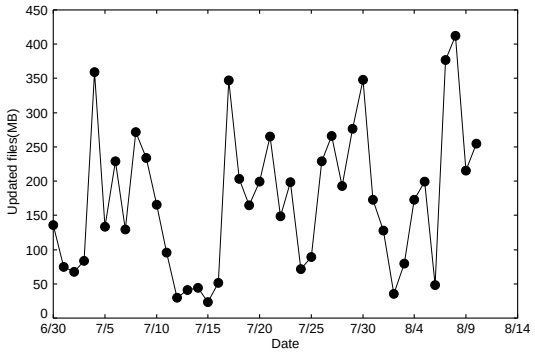


図 8: FTP サーバの更新状況

かん」を利用することは, 常に CSI に接続することとほとんど相違ないが, 存在しないファイルを他のサーバへ探しに行く機能がある分, 有利である.

また図 7 中の矢印は, JAIST における最大利用クライアント数の制限によりログインできなかった時刻を示す. 「みかん」への接続では常にファイルが取得できたことを考慮すると, 「みかん」の利用により可用性が大きく向上できたといえる.

### 7.3 モニタリングにかかるトラフィックの評価

モニタリングにかかるトラフィックの評価を行った. ミラーサーバはミラーの更新を行う際に, 1 日にどの程度のトラフィックが発生するのかを測定した.

ftp.jaist.ac.jp における Vine Linux のディレクトリツリーのファイルリストを, 1 日おきに取得する. それを比較することで, 1 日に更新されるファイルの総容量を測定した. 結果を図 8 に示す.

図をみると, 毎日更新が行われていることが分かる. Vine Linux は開発が活発に行われているディストリビューションの 1 つであり, ほぼ毎日開発が行われているためである.

1 日平均の更新量は, 約 172MB であった. ミラーサーバを設置した場合は, 毎日この程度のトラフィックが発生することになる. これに対し, ファイルリストの転送量はわずか約 5.4MB であった. 「みかん」はファイルの更新を行わずファイルリストの取得のみを行うので, ミラーサーバと比較してネットワークトラフィックを大幅に減らすことができることが分かる.

## 8 おわりに

本論文のまとめと今後の課題について述べる.

### 8.1 まとめ

現在, インターネット上でファイルを配布するために FTP が広く使用されている. またファイルの配布は, 1 台のマスターサーバではなく, 複数のミラーサーバによって行われている.

しかし, 最新のファイルを保持するミラーサーバの中からネットワーク的に近いサーバを選んで, ファイルを取得するための方式はまだ確立されていない. また従来の方式では, 専用クライアントが必要であったり, サーバ選択の単位がディレクトリ単位であったりと問題が多かった.

本論文では, これらの問題を解決するために「みかん」を提案した. 「みかん」を用いることによって, 従来のクライアントに手を加えずに, 最新ファイルをできるだけ近いサーバから取得可能になることを示した.

また, 実際に「みかん」の実装を行い, その上で評価を行った. 「みかん」を使用することによるオーバーヘッドは実用上問題ないことを示した. 「みかん」を使用することによる可用性を示した. 実際にミラーサーバを設置するよりもトラフィックを大幅に減らせることを示した.

### 8.2 今後の課題

今後の課題を以下に述べる.

- ファイル取得要求の誘導方法
  「みかん」は, クライアントから接続が行わなければ, サーバ選択を行うことができない. そのため, URL などで直接 FTP サーバを指定された場合は「みかん」によるサーバ選択機能を利用することができない. そこで, 「みかん」の方に

ファイル取得要求を誘導する方法が必要になる.

- モニタリング頻度の最適性についての考察
  ファイルの最新性を保証するために, どの程度のモニタリングを行う必要があるのかが確かめられていない. 頻度が多いとネットワークトラフィックを増加させることになり, 頻度が少ないと最新性を保証することができなくなる. そこで適度な頻度のみきわめが必要である.
- 近いサーバの選択方法
  近いサーバを決定するための手法の考察を十分行っていない. ネットワーク状況は刻一刻変化するので, このみきわめは重要である.

## 参考文献


[1] J. Postel and J. Reynolds, File transfer protocol (ftp), RFC 959, October 1985.

[2] 下川俊彦 吉田紀彦 牛島和夫, ネームサーバを用いた柔軟な負荷分散, インターネットコンファレンス '99, pp.107-116, December 1999.

[3] T. Shimokawa, N. Yoshida, and K. Ushijima, DNS-based Mechanism for Policy-added Server Selection, SSGRR2000, 2000.

[4] 横田 裕思, DNS Balance, http://openlab.ring.gr.jp/dns_balance/dns_balance.html.

[5] 横田 裕思, DNS Trick, http://openlab.ring.gr.jp/dns_balance/dns_trick.html.

[6] 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 論文集, pp.121-130, 2001.

[7] M. Burford, GetRight, http://www.getright.com/.

[8] C. Yoshikawa, B. Chun, P. Eastham, A. Vahdat, T. Anderson, and D. Culler, Using Smart Clients to Build Scalable Services, USENIX Annual Technical Conference, 1997.

[9] M. Hamilton and A. Novikov, FTP Mirror Tracker: First Steps towards URN, Proc. 5th International Web Caching and Content Delivery Workshop, September 2000.

[10] A. Novikov and M. Hamilton, FTP Mirror Tracker: A Few Steps towards URN, USENIX LISA 2000, 2000.

[11] R. Rivest, MIT Laboratory for Computer Science and RSA Data Security Inc, The MD5 Message-Digest Algorithm, RFC 1321, April 1992.

[12] 尾藤正人, みかん, http://www.nets.ce.hiroshima-cu.ac.jp/~masato/mikan/

[13] 舟阪淳一, 尾藤正人, 石田賢治, 天野橘太郎, 最新ファイルの提供を保証する FTP 代理サーバの開発, 2002 年電子情報通信学会総合大会, SB-4-5(pp.768-769), 2002.

[14] Junichi Funasaka, Masato Bito, Kenji Ishida, and Kitsutaro Amano, PFTPD: An FTP Proxy System to Assure the Freshness of Files, Proc. 22nd ICDCS Workshops (1st International Workshop on Assurance in Distributed Systems and Networks), pp.57-62, 2002.