第 12 部

ネットワーク運用技術

第 1 章

Cnews

ネットワーク間におけるさまざまな情報交換、あるいはメールよりもより開放的、広域的な議論を行なうための媒体として、従来より電子ニュースシステムが使用されてきた。従来この電子ニュースシステムとして Rick Adams が開発した Bnews システムが用いられてきたが、この Bnews システムは、

1. 流入するニュースの処理速度の低下。
   NNTP と組み合わせて使用した場合に、そのままでは複数のアーティクルをまとめて処理することができないため、ニュースが相手先から一度に大量に流入した場合に処理速度がかなり低下してしまう。またその際のシステムに与える負荷も大きくなりがちである。またそのことがさらにニュースシステムの処理全体に負荷をかけることになり、全体としてかなりパフォーマンスが低下してしまう。

2. ニュースアーティクルの処理能力の限界。
   ニュースファイル管理データベースとして dbm を使用しているため、ニュースの流量の増加に伴いスプールの容量を増加させると、dbm ではその処理能力に限界を生じる。そのため、例えば expire 等の処理能力がかなり低下し、またその際のシステムの負荷も大きくなる。

3. ニュースを送出する際のシステムに与える負荷の上昇。
   アーティクルを batch+compress して隣接サイトへ送出する場合、ファイル 1 つ 1 つに対して batch する際にサイズをチェックしているため、batch の処理がシステムにかなりの負荷を与えてしまう。

等、最近の広域 IP ネットワーク環境、あるいは近年のニュース流通量の急激な増加に対し不都合な点が生じて来た。そのため、Bnews にかわり、特に処理速度の面で劇的な改良がなされた **Cnews** を用いてニュースシステムを運用するサイトが近年増加しつつある。

そのような背景に鑑み、ここでは Cnews システムに関し、その特徴、導入、運用などについて記述し、また特に Bnews から移行する場合の注意点についても記述する。

本章では、筆者が特に強調したいことがら、あるいはファイル名等については**/usr/lib/newsbin** のように太字で表記し、UNIX システムにおけるファイル名、パス名等については`/usr/lib/news` のようにタイプライタ書体を用いて表記している。

1.1 Cnews とは

Cnews は Univercity of Toronto の Geoff Collyer と Henry Spencer が作成した広域ネットワークにおける電子ニュースシステムである。

Cnews には従来の Bnews(Bnews version 2.11 patchlevel 19) と比較して以下のような特徴がある。

細分化されたディレクトリ構造

従来の Bnews では、ニュースに関連する設定ファイル、コマンドなどのほとんど全てが/usr/lib/news ディレクトリに格納されていた。Cnews ではこれがさらに細分化され、ニュース関連のコマンドは/**usr/lib/newsbin** ディレクトリに移された。また /usr/lib/newsbin ディレクトリはそのコマンドの目的毎にさらに細分化されており、例えばニュースシステム管理関連のコマンドはすべて/usr/lib/newsbin/maint ディレクトリに格納されている。

dbz のサポート

従来 Bnews で使用されていた dbm に代わって、より処理速度が早く、かつデータファイル (.pag ファイル) の大きさが小さい **dbz** データベースマネージャに対応した。これにより history ファイルの検索が高速になっており、expire 等の処理が劇的に向上している。またこのため、Bnews から Cnews へ移行する場合には、nntpd を dbz 対応で再インストールする必要がある。

システム設定の変更が容易

/usr/lib/news 下の設定ファイルを vi 等でエディットすることにより、従来の Bnews では再インストールでしか変更できなかった設定をインストール後に容易に変更することが可能になった。例えば whoami ファイルではニュースシステム名を、rnews.immed ファイルでは到着したニュースの処理に関する設定を再インストールすることなく簡単に変更することができる。

フロントエンド化された inews

Cnews では Bnews における inews/rnews に代わって、**relaynews** というプログラムが実際のニュースアーティクルの処理を行なっている。従来のニュースシステムとの互換性を保つため、inews および rnews は relaynews を呼び出すフロントエンドプログラムとして存在している。また Bnews では inews と rnews の実体は同一のプログラムであったが、Cnews では inews と rnews は別のプログラムとなっている。

batched input のサポート

nntpd と組み合わせて使用する場合、Cnews は batched input(複数のアーティクルを一度に処理することでシステムの負荷を軽減する機能) をサポートしている。この機能を利用するには nntpd を batched input 機能付きでインストールする必要がある (詳細は nntpd キットの common/conf.h ファイルを参照のこと)。

sendbatch の高速化

UUCP サイトにニュースを batch で送出する場合の処理が大幅に向上している。このため、複数の UUCP サイトを抱えており、かつニュースは NNTP で流入するようないわゆる Hub の役割を果たしているようなサイトでは、Cnews はかなり有効である。

expire の高速、高機能化

Bnews と比較して expire(古い記事の消去) 処理が劇的に高速化されている。これは dbz の導入によるところが大きい。また Cnews では 1 度の expire でニュースグループ毎に異なった expire の期間を設定することが可能である。従来、Bnews で同一のことをしようとした場合、expire をその回数分だけ動作させる必要があった。

また Cnews ではニュースの保存を `/usr/spool/oldnews` だけでなく、任意の場所に、かつ任意のニュースグループ単位で保存することが可能になっている。また設定も **explist**(後述) ファイルに設定するだけで容易に変更することが可能である。

管理手法の大幅な変更

Bnews と比較すると、以上のようなかなり大規模な変更が行なわれているため、管理の手法がかなり異なっている。また、`sys` ファイル等の設定ファイルも Bnews とは若干異なる点があるため、特に Bnews から移行する場合に注意が必要である。この点については後述する。

1.2　Cnews の入手

本報告書執筆時の Cnews の最新のバージョンは **“22-Dec-1991”** 版である。これは Cnews キットの **PATCHDATES** というファイルを参照することにより判別することができる。本報告書もこのバージョンの Cnews について記述している。また Cnews は頻繁にアップデート、およびバグフィックスが行なわれており、またその情報は USENET の **news.software.b** ニュースグループ等を用いて告知されている。そのため定期的にこのニュースグループをチェックしておくことが管理上好ましい。

1.2.1　日本語への対応

Bnews の inews とは異なり、Cnews の relaynews はそのままでも本文の ESC キャラクタを通す。すなわちそのままで現在のネットワーク漢字コードを用いたニュースの運用が可能である。しかし Cnews に付属している inews シェルスクリプトには Bnews との互換性のために **ESC キャラクタを strip してしまう**コードが入っている。そのため、そのままではローカルサイトから inews 経由で投稿されたニュースは ESC キャラクタが strip されてしまうことになる。これは Cnews キットに付属の inews シェルスクリプトに以下のようなパッチを当てることで回避が可能である。

```
*** relay/sh/inews.ORG Tue Mar 17 23:16:01 1992
--- relay/sh/inews Fri Feb 21 19:04:45 1992
*** 178,185 ****

   # strip invisible chars from body, a la B news.  bells and escapes are right
out.
   case "$trversion" in
!  v7)  tr -d  '\1-\7\13\15-\37' ;;
!  v6)  tr -d '[\1-\7]\13[\15-\37]' ;;
   esac <$inbody

   if test -r $HOME/.signature; then
--- 178,185 ----

   # strip invisible chars from body, a la B news.  bells and escapes are right
out.
   case "$trversion" in
!  v7)  tr -d  '\1-\7\13\15-\32\34-\37' ;;
!  v6)  tr -d '[\1-\7]\13[\15-\32][\34-\37]' ;;
   esac <$inbody

   if test -r $HOME/.signature; then
```

このパッチの当たった Cnews は、*sh.wide.ad.jp* [133.4.11.11] をはじめ、国内の主な anonymous ftp サイトから入手することが可能である。

1.3 UNIX システムファイルの設定

ここでは、Cnews の運用を行なう場合の UNIX システムの設定手順について説明する。実際の Cnews システムの生成の詳細、およびそのコンパイルの手順等については Cnews キットの doc ディレクトリ下のドキュメント等を参照されたい。

1.3.1 システム起動時の設定

システム起動時に **newsboot** コマンドを実行するようにする。これは news の権限で実行される必要がある。具体的には以下のような記述を rc.local あるいはそれに相当するファイルに加える。

```
#
# news startup
#
if [ -f /usr/lib/newsbin/maint/newsboot ]; then
        su news -c /usr/lib/newsbin/maint/newsboot &
```

```
        (echo "news cleanup")                          >/dev/console
fi
```

1.3.2　定期的に実行すべきコマンド

newswatch コマンドは、ニュースシステムが正常に動作しているかをチェックするコマンドである。結果が標準出力にメール形式で報告されるようになっているため、crontab でニュースの権限で 1 日に数回実行するとよい。また/usr/lib/news/watchtime ファイルの時間をチェックすることによって、最近の newswatch が実行された時間を知ることができる。

newsdaily コマンドは、ログファイルの処理、それまでに起こったニュースシステム上のエラーの報告、古いゴミファイルの処理等を行なうコマンドである。通常 1 日に 1 回起動するのがよい。

doexpire コマンドは、古い記事を消去するコマンドである。通常 1 日に 1 回起動される。このコマンドは**/usr/lib/news/explist** というファイルを参照し、そのファイルの記述にしたがって expire を行なう。

upact コマンドは、/usr/lib/news/active ファイルの最小値をアップデートするコマンドである。通常 expire が終了した後に起動されるようにしておくのが適当である。

また Cnews のキットには upact よりも高速、かつ負荷の低い **updatemin** というコマンドもオプションで用意されており、このコマンドが正常に動作するシステム (SunOS 4.1.1 など) では upact に替えてこちらを使用してもよい。updatemin コマンドは Cnews キットのあるディレクトリで、

```
% cd expire
% make updatemin
% cp updatemin /usr/lib/newsbin/expire
```

とすることによりインストールされる。updatemin コマンドは通常の Cnews のインストール手順では生成、インストールされないので注意が必要である。

sendbatches コマンドは、Bnews における sendbatch コマンドにあたる。すなわち batch を利用して隣接サイトにニュースを転送する際に用いられるコマンドである。sendbatch と異なっている点は、アーギュメントを省略した場合には**/usr/lib/news/batchparms** に記述された全てのサイトに対して順番に batch が行なわれるということである。また sendbatches は自分自身が 2 つ以上同時に起動されないように排他制御をしているため、従来の Bnews のように sendbatch を 2 つ以上並行して動作させるような記述はできない点に注意すべきである。通常のサイトでは 1 時間に 1 回程度アーギュメントなしで起動すれば十分である。

newsrun コマンドは、Bnews における rnews -U コマンドに相当する。すなわち到着したニュースアーティクルをすぐに処理せず、後でまとめて処理するような設定を行なっ

ている場合、具体的には/usr/lib/news/rnews.immed ファイルの内容を “yes” に設定していない場合にこのコマンドを 10 分に 1 回程度定期的に実行する必要がある。

しかしながら特にニュースの転送速度が要求されるサイト (例えば NNTP で組織外からのニュースを組織内のサーバに配布しているようなサイト) では通常 rnews.immed ファイルの内容を “yes” に変更して設定し、newsrun コマンドを起動する設定にしないのが普通である。また現在 WIDE プロジェクトの各 NOC のニュースサーバはニュースの転送速度を向上させるため、全て rnews.immed ファイルを “yes” に設定してある。

以上をまとめると、news に関連する crontab ファイルの例は以下のようになる。

```
# newswatch
0 2,8,14,20 * * *       /usr/lib/newsbin/maint/newswatch | /bin/mail usenet
# newsdaily
0 1 * * *               /usr/lib/newsbin/maint/newsdaily
# doexpire
0 3 * * *               /usr/lib/newsbin/expire/doexpire
# upact (or updatemin)
0 4 * * *               /usr/lib/newsbin/expire/upact
#0 4 * * *              /usr/lib/newsbin/expire/updatemin
# sendbatches
0 * * * *               /usr/lib/newsbin/batch/sendbatches
# newsrun
# If your /usr/lib/news/rnews.immed sets ‘‘no’’, you should enable this line.
#0,10,20,30,40,50 * * * * /usr/lib/newsbin/input/newsrun
```

1.4 Cnews の設定

ここでは、Cnews システムを実際に運用する場合の留意点を、特に Bnews と異なる点、あるいは誤りやすい点に重点をおいて記述する。

1.4.1 config ファイル

Cnews では、/usr/lib/news/bin/config というファイルで実際に Cnews 上で用いられる環境について記述されている。Cnews で提供されるシェルスクリプトでは、例えば、

```
#! /bin/sh
# doexpire - overall administration for expire

# =()<. ${NEWSCONFIG-@<NEWSCONFIG>@}>()=
. ${NEWSCONFIG-/usr/lib/news/bin/config}
```

のように最初にこのファイルを読み込むことで、Cnews の環境変数を設定している。ここでは設定される環境変数の主なもの、および標準的なインストールを行なった際に設定される値を以下に記述する。

NEWSARTS ニュースアーティクルの木構造が実際に置かれるディレクトリ
　Bnews における SPOOLDIR である。通常は/usr/spool/news が代入される。

NEWSCTL Cnews の設定ファイルが置かれるディレクトリ
　Bnews における LIBDIR に相当する。通常は/usr/lib/news である。

NEWSBIN Cnews を構成するプログラム群が置かれるディレクトリ
　通常は/usr/lib/newsbin になり、さらにこの下にいくつかのサブディレクトリが存在する。

NEWSUMASK Cnews が用いる umask 値
　通常は 002 が代入されている。

NEWSPATH Cnews が用いるコマンドサーチパス
　通常は/bin:/usr/bin:/usr/ucb が代入されている。**/usr/ucb が/usr/bin の後でなければならない**ことに注意されたい。

NEWSMASTER ニュースシステムの管理者
　通常は usenet が代入されている。この usenet ユーザは `aliases` ファイルで然るべき管理者にエリアスしておく必要がある。

また、本報告書においても以後この記述を用いて、例えば **NEWSCTL/sys** のように記述することとする。

1.4.2　JLE 環境等における `config` ファイルの応用

また、この `config` ファイルにおいては、ニュースシステムの全てに影響を与える動作環境についても記述することができる。例えば JLE 等の日本語環境下では通常、

```
JLE% date
1992 年 03 月 26 日（木）14 時 37 分 22 秒 JST
JLE%
```

のように日本語で日付が表示されてしまうため、Cnews の標準的なシェルスクリプトが正常に動作しない。これを防ぐためにはこの `config` ファイルに、

```
LANG=C; export LANG
```

という行を追加することによって date コマンドの出力を英語形式に変更することができ、Cnews システムを正しく動作させることができるようになる。

1.4.3 NEWSCTL 下の設定ファイル

前段でも若干説明したように、Cnews は Bnews とは各システム設定ファイルがかなり異なっている。以下順を追って説明する。

NEWSCTL/active

Bnews と比べて第 4 フィールドに用いられる記述子が拡張されている。また第 2 フィールドの最大桁数も 10 桁に変更されている。active の例を以下に示す。

```
fj.general 0000000051 00049 y
fj.binaries.msdos 0000000528 00527 m
alt.binaries.pictures.erotica.female 0000000000 00001 x
alt.sys.sun 0000000000 00001 =comp.sys.sun.misc
```

このように Cnews では、第 4 フィールドに **"x"** という記述子を書くことができる。これはこのニュースグループに配布されて来たニュースは処理せずそのまま捨てるということを意味する。また上記のように、**"=realgroup"** という記述をすることもでき、この場合そのニュースグループに配布されて来たアーティクルは、=で指定されたニュースグループに配送される。すなわちこれは Bnews における aliases ファイルの機能に相当する。しかしこの場合は Bnews のそれとは異なり、配送されたニュースアーティクル自身には何の影響も与えない (Bnews では alias されたニュースの Newsgroups: フィールドの書換えが行なわれる)。

NEWSCTL/active.times

Cnews では、active.times というファイルが拡張されている。このファイルには外部からの newgroup コントロールメッセージ、あるいはローカルサイトにおける addgroup コマンドによって作成されたニュースグループについて記述されている。active.times の例を示す。

```
local.test 687430866 yasuhiro@sh.wide.ad.jp
bionet.plants 706490375 shibumi@genbank.bio.net (Kenton A. Hoover)
rec.games.frp.archives 706942328 tale@uunet.uu.net (David C Lawrence)
fj.soc.environment 705950032 news@icsts1.osaka-u.ac.jp (News manager)
```

このように active.times には、

ニュースグループ名 作成された時間 作成した人

が記録される。すなわちこのファイルをチェックすることによってニュースグループの作成履歴をチェックすることができることになる。

NEWSCTL/batchparms

Cnews になって、sendbatches コマンドによって参照される batchparms というファイルが追加された。batchparms の例を以下に示す。

```
#
# site                  size    queue   builder muncher sender
# ----                  ----    -----   ------- ------- ------
target                  300000  5       batcher compcun viauuxz
target.ihave            50000   20      batchih nocomp  viainews
target.sendme           50000   20      batchsm nocomp  viainews
```

batchparms ファイルは 6 つのフィールドで構成される。

第 1 フィールドには相手のサイト名を記述する。通常 **NEWSCTL/sys** で記述したサイト名を記述すれば良いが、相手のニュースシステム名 (Path:フィールドに現れる名前) と UUCP システム名が異なる場合には sys で記述したニュースシステム名ではなく、UUCP システム名を記述する必要があるため、注意が必要である。

第 2 フィールドには 1 つの batch パケットのサイズを記述し、第 3 フィールドには 1 回の batch で一度に送られるパケットの数を記述する。すなわち上記の例では *target* に対するエントリは 300000 バイトのパケットを一度に 5 つ送るということを意味する。

またこのサイズは、compress **される前の**サイズであるということに注意すべきである。

第 4 フィールド以降には順に、batch パケットの生成方法、および batch の際に compress 等のデータの加工をするか否か、また相手のシステムに送る際にどの方法でニュースを送るかについて記述する。実際にはこれらは **NEWSBIN/batch** ディレクトリ下にあるその名前のプログラムが実行されている。詳細についてはそれぞれのプログラムの内容を参照されたい。また NEWSBIN/batch 下のプログラムはそのほとんどがシェルスクリプトで書かれているため、設定の変更、改良等が非常に容易であるということも注目すべきであろう。

以下に各々のフィールドに書かれる代表的なプログラムについて記述する。

- batcher
  通常の batch を行なう場合の builder として第 4 フィールドに指定する。
- batchih/batchsm
  ihave/sendme プロトコルを行なう場合に指定する。batchih は相手のシステムに対して ihave メッセージを送る際に指定し、batchsm は sendme メッセージを送る際に指定する。

- compcun/compc7
  compcun は “#! cunbatch” を付加し、compress を用いて batch をする場合 (Bnews の “sendbatch -c” と同様) に第 5 フィールドに指定する。また compc7 は同様に “#! c7unbatch” を付加し 7 ビット圧縮を用いて batch をする場合 (Bnews の “sendbatch -c7” と同様) に指定する。
  ここで注意すべきことは、Cnews のデフォルトでは compress プログラムが **12 ビット圧縮で起動されている**ことである。そのため Cnews を用いて通常の (Bnews と同じ) 圧縮効率で batch を行ないたい場合は、NEWSBIN/batch/compcun シェルスクリプトを 16 ビット圧縮を行なうように修正しなければならない。

- nocomp
  nocomp は圧縮を用いない場合に第 5 フィールドに指定する。通常、ihave/sendme プロトコルのために使用される。すなわち batchih や batchsm を第 4 フィールドに指定した場合に使用される。

- viauux/viauuxz
  これは相手のシステムへのニュースの転送に UUCP プロトコルを用いる場合に第 6 フィールドに指定する。通常は viauuxz を用いればよいが、もし相手の UUCP システムに-z オプションが存在しない場合には viauux シェルスクリプトに-n オプションを追加して用いることになる。

- viarsh
  これは相手へのニュースの転送に rsh を用いる場合に用いる。Bnews ではこのような設定は若干面倒であったが、Cnews ではこのような設定も容易に行なうことが可能である。

- viainews
  ニュースの処理に inews を用いるということを意味する。これは ihave/sendme プロトコル、すなわち第 4 フィールドに batchih あるいは batchsm を指定した場合に第 6 フィールドにこれを指定する。

NEWSCTL/explist

explist は、batchparms 同様 Cnews で追加された設定ファイルである。explist の例を以下に示す。

```
# hold onto history lines 20 days, nobody gets >90 days
/expired/                       x       20      -
/bounds/                        x       0-1-90  -
# alt and rec are 7 days.
alt                             x       7       -
rec                             x       7       -
# fj is longer days.
```

```
fj                              x       14      -
fj.sources                      x       14      =/archive/fj-sources
fj.sources.d                    x       14      -
# archiving moderated NG only.
comp.sources                    m       10      /archive/oldnews
# default:  10 days
all                             x       10      -
```

explist ファイルは 4 つのフィールドで構成される。

第 1 フィールドにはニュースグループ名を記述する。all は全てのニュースグループを表す。またここに/expired/ を指定した場合には、第 3 フィールドの日付は NEWSCTL/history ファイルにおける履歴の保存期間を表す。すなわち Bnews の expire コマンドにおける-E オプションに相当する。また/bounds/ を指定した場合には第 3 フィールドには Expires:を指定した場合の記事の保存期間に関する指定を行なうことができる。これらの詳細については Cnews のドキュメントを参照されたい。

第 2 フィールドには m,u, および x が指定できる。m を指定した場合その行は第 1 フィールドで指定されたニュースグループのうちの moderated なニュースグループにのみ適応され、u を指定した場合には unmoderated なニュースグループにのみ適応される。また x を指定した場合双方に適応される。すなわち全てのニュースグループに適応されることを意味する。

第 3 フィールドには expire を行なう日付を指定する。ここには 3 つの日付を指定できるが、通常は 1 つで十分である。3 つの日付はハイフンを用いて連結して (0-1-20 のように) 記述するが、Expires: ヘッダを用いた記事以外では真中の数字以外は意味を持たない。ちなみに例のように数字を 1 つだけ指定した場合には真中の数字が指定されたとみなされる。これは expire の対象となる日付を表す。すなわちこの例では alt および rec ニュースグループは 7 日で expire を行ない、fj は 14 日間保存し、その他のニュースグループは 10 日で expire されることを意味する。

第 4 フィールドには記事を保存するディレクトリを絶対パスで記述する。“-” を指定した場合には記事の保存は行なわれない。またパス名の前に “=” を指定すると記事はツリー構造ではなく、そのディレクトリ上にそのまま保存される。すなわちこの例では fj.sources ニュースグループの記事は/archive/fj-sources/fj/sources ディレクトリではなく、/archive/fj-sources ディレクトリに保存されることになる。

NEWSCTL/log,errlog

Cnews のログファイルは Bnews のそれとはフォーマットが大きく異なる。また Bnews と異なり **junk に落ちても errlog にはその記録が残らない**ので注意が必要である。log の例を以下に示す。

```
Mar 26 02:02:26.979 sh.wide + <1018@eagle.ukc.ac.uk> ihave-comp green-hill
```

```
 astecgw dclsic cckgw rena etlpost lkbreth hitachi-cable.co.jp iwanami nec
-tyo isr ricohgwy scslwide hoffman sranha sriva cs.titech ccut virgil toum
on
Mar 26 04:37:20.055 nec-tyo + <1992Mar25.160748.27165@teleride.on.ca> ihav
e-rec dclsic cckgw rena etlpost lkbreth hitachi-cable.co.jp iwanami isr sc
slwide hoffman sranha sriva ccut sh.wide
Mar 26 04:37:20.595 nec-tyo j <1992Mar25.160748.27165@teleride.on.ca> junk
ed due to groups 'rec.arts.sf.science'
Mar 26 04:58:38.354 sh.wide - <ASAMI.92Mar26040906@pukeko.cs.Berkeley.EDU>
 duplicate
```

このようにログファイルは、

日付 受信システム名 {+|-|j} メッセージ ID 送信システム名 [送信システム名 ...]

という形式になっている。第 3 フィールドが "+" である記事は正常に受信されたことを表し、"-" の場合は何らかの理由により受信されなかったことを表す。理由は上記のようにログファイルを参照することによってチェック可能である。

第 3 フィールドが "j" になっている場合、該当するニュースグループがシステム中に存在しなかったため、その記事が junk ニュースグループに落ちたことを表す。また Cnews では Bnews と異なり、**ローカルサイトにおいて junk に落ちた記事であっても周辺サイトには** sys **ファイルの記述にしたがって配布される**という点に留意すべきである。上記の例においても同一のメッセージ ID を持つ行が 2 行にわたってログファイルに記録されている。

NEWSCTL/mailname

mailname には、自ホスト名を含んだフルドメイン名、あるいは自ホスト名を含まないフルドメイン名がセットされる。これはそのニュースシステムを動作させるマシンから発信されるニュース、あるいは sendsys 等のコントロールメッセージに対する応答としてそのマシンから発信されるメールの From: フィールドに影響を与える。このことからこのファイルの内容は、Cnews システムが広域ネットワーク上において自分自身のシステム名を表す名称として用いられていることになる。すなわち電子メールシステム等の設定においてホスト名を隠蔽して運用しているサイトでは、この mailname ファイルをドメイン名のみの内容に設定しておくことにより、そのサイトから発信されるニュースの From: フィールドを Bnews でいうところのいわゆる GENERICFROM 形式にすることができる。

またこのファイルの設定内容はニュースシステムによって付加される Message-ID: フィールドにも使用される。mailname の設定内容における挙動の相違を以下に示す。

```
sh.wide.ad.jp% cat mailname
sh.wide.ad.jp   (一般的な設定)
wide.ad.jp      (ホスト名を隠蔽する設定)
```

この場合、一般的な設定の場合にそのサイトから発信された記事の From: ヘッダおよび Message-ID: ヘッダは、

```
From: user@sh.wide.ad.jp
Message-ID: <1992Mar27.042446.29491@sh.wide.ad.jp>
```

という形式となり、ホスト名を隠蔽する設定を行なった場合には

```
From: user@wide.ad.jp
Message-ID: <1992Mar27.042446.29491@wide.ad.jp>
```

という形式となる。

このように Cnews では、mailname の内容を変更するだけで From: ヘッダの設定を容易に変更することが可能になっている (Bnews では同様のことを行なう場合にはシステムの再インストールを行なう必要がある)。

NEWSCTL/mailpaths

mailpaths は、moderated newsgroups への投稿に対する処理の設定を行なうファイルである。Bnews にも同様の機能をもつファイルが存在するが、Bnews のそれとは若干設定が異なっているので注意が必要である。現在の日本の広域ネットワークにおける望ましい mailpaths の内容を以下に示す。

```
fj.all          %s@junet.ad.jp
jp.inet.all     %s@utsun.s.u-tokyo.ac.jp
all             %s@ames.arc.nasa.gov
```

この設定は、

1. fj 下の moderated newsgroup(例えば fj.announce) へのポストは

   fj-announce@junet.ad.jp

   というアドレスに変換されてメールで送られる
2. 同様に、例えば jp.inet.announce へのポストは

   jp-inet-announce@utsun.s.u-tokyo.ac.jp

   にメールで送られる
3. その他全ての moderated newsgroup は、USENET において moderated newsgroups を処理可能なサイトの 1 つである ames.arc.nasa.gov に対して、

   news-announce-important@ames.arc.nasa.gov

というアドレスでメールが送られる

ということを意味している。

moderated newsgroup を処理できるサイトのリストは、news.lists ニュースグループ等に定期的にポストされているので参照されたい。また現在の日本 Internet における設定ではそれらのホストの中で日本から Internet 的に最も近距離である

ames.arc.nasa.gov

を設定するのが適当であると考えられる。また InetClub を利用して海外への電子メールを運用しているサイトでは上記の all 行を、

```
all             %s@uunet.uu.net
```

と設定してもよい。

従来の Bnews において設定していた **internet 行や backbone 行は不要**であることに注意されたい。また all 行はその時点で指定していない全てのニュースグループに該当してしまうため、必ず **mailpaths ファイルの最終行で設定しなければならない**ことに注意すべきである。

NEWSCTL/rnews.immed

`rnews.immed` は、ローカルサイトに到着したニュースが rnews によって処理される際に、即座に全ての処理を行なうか、あるいはいったん `NEWSARTS/in.coming` ディレクトリ (Bnews における.rnews ディレクトリに相当する) に集めたうえで newsrun コマンド (Bnews における rnews -U コマンド) でまとめて処理をするかの設定を変更するファイルである。すなわちこれは Bnews における SPOOLNEWS の機能にほかならない。

これに関しては、本章において既に説明したように、通常広域ネットワーク環境においては rnews.immed ファイルは “yes” に設定しておくのが一般的であるが、システムの負荷が大きい等の理由により “no” に設定する場合もある。その場合はニュースアーティクルの処理速度は当然低下する。

また Cnews では、このように rnews.immed の内容を変更するだけで rnews の挙動の設定を容易に変更することが可能になっている (Bnews では同様のことを行なう場合にはシステムの再インストールを行なう必要がある)。

NEWSCTL/whoami

whoami は、自分のニュースシステム名を設定するファイルである。ニュースシステム名とは、自システムを経由するニュースアーティクルの Path:フィールドに現れる名前のことである (Bnews における GENERICPATH に相当する)。`whoami` における設定と `mailname` におけるそれとを混同しないよう注意されたい。

通常 whoami ファイルの設定は、そのシステムにおける UUCP システムの名称、すなわち “uuname -l” コマンドの出力と一致させておくとシステム設定上の混乱を避けやすいであろう。またこのシステム名は後述する ihave/sendme プロトコルを設定する場合に非常に重要になるので、自システムにおいて whoami ファイルにどのような設定がなされているかということは必ずチェックしておく必要がある。

NEWSCTL/sys sys ファイルは、Bnews と設定方法がかなり異なっているため、ここで若干詳細に説明する。また Cnews では Bnews と異なり、ニュースシステムにおいて ihave/sendme プロトコルを設定する場合には **sys ファイルにおいてそのための設定をしなければならない**。この設定については後段「ihave/sendme プロトコルの設定について」を参照されたい。ここでは最も基本的な設定について説明する。

Cnews における最も基本的な sys ファイルの設定の例を以下に示す。

```
##
# My own entry.
##
ME:all,!alt.sex,!alt.binaries.pictures.erotica/all
##
# foo, as NNTP site. Sending by nntpsend.
##
foo:all,!local,!to,to.foo\
  /all,!local:F:
##
# bar, as UUCP site. Sending by UUCP batch.
# If articles are already relayed host ‘‘baz’’, do not feed to bar.
# rec.* newsgroups dont send except to rec.arts.anime.
##
bar/baz:all,!local,!rec,rec.arts.anime,!to,to.bar\
  /all,!local:f:
```

Cnews の sys は、Bnews のそれと比較して第 2 フィールドと第 3 フィールド (それぞれのフィールドはコロンで区切られる) の記述が拡張されている。以下それについて説明する。sys ファイルにおけるその他のフィールドの意味、およびその設定の詳細については、Cnews に付属のドキュメント等を参照されたい。

Cnews の sys では第 2 フィールドにディストリビューションを “/” 記号で区切って記述するように拡張された。これに伴って従来の Bnews システムにおいて sys ファイルを記述する際のトラブルの一因となっていた “world” というエントリは**廃止された**。また “control” というニュースグループも Cnews システムにおいてはもはや**存在せず、そのかわりにコントロールメッセージはそれぞれのディストリビューショ**

ンを持つ通常の記事として処理されるように変更がなされた (control は Cnews システムでは “pseudo newsgroup” として扱われている)。

すなわち Cnews では ihave コントロールメッセージは “ihave” というディストリビューションを持つ 1 つのアーティクルとして、同様に sendme コントロールメッセージは “sendme” というディストリビューションを持つ 1 つのアーティクルとして扱われることになる。

また Cnews の sys では第 2 フィールドのニュースグループ名、およびディストリビューションに対し、“all” という指定を行なうことが可能となった。この指定はそれぞれ全てのニュースグループ、全てのディストリビューションに対応している。しかしながらこの “all” という指定を行なった場合には、**“local” ニュースグループや “local” ディストリビューションも含まれる**ということに注意しなければならない。これは Cnews では **“local” という名称はもはや特殊ではない**ということを意味している。すなわち第 2 フィールドに “all” を使用した場合には上記の例のように必ず **“!local”** というエントリを加える必要がある。また **“to” ももはや特殊ではない**ため、コントロールメッセージ等を正しく処理するためにニュースグループフィールドにおいて、**“!to,to.***sysname***”** という記述をしなければならない。これらの設定を誤ると、本来その団体内部で運用されるべき記事や他サイトから流れて来たコントロールメッセージが外部サイトへ流出してしまい、ニュースシステムに悪影響を与える場合があるため、注意が必要である。

Cnews では第 3 フィールドの記述に “f” というエントリが拡張された。これは従来の “F” の機能に加えて、そのニュースアーティクルの実際のサイズもファイルに記録するように拡張されたものである。Cnews ではこれを sendbatches の際のバッチサイズの算出に用いることによって、バッチ動作の際のシステムに対する負荷を大幅に軽減することに成功している。

Cnews では “F”、“I”、あるいは “f” を第 3 フィールドに書いた場合にバッチファイルが書かれる場所 (Bnews システムにおける BATCHDIR に相当) が Bnews とは変更されているため、注意が必要である。Cnews ではこのファイルは

`NEWSARTS/out.going/`*sysname*`/togo`

に書かれる (*sysname* は `sys` における第 1 フィールドに対応する)。そのため nntpsend を用いてニュースの送信を行なう場合には NNTP キットに付属の nntpsend.sh スクリプトの変更が必要になる。また `sys` ファイルに新たな隣接システムに対するエントリを追加した場合には、あらかじめ `NEWSARTS/out.going/`*sysname* ディレクトリを news の権限で作成しておかなければならない。これらの作業を行なう際のコマンドとして Cnews では **NEWSBIN/maint/addfeed** コマンドが用意されている。addfeed コマンドの使用法については Cnews に付属のドキュメントを参照されたい。

1.4.4　Cnews に付属のコマンドについて

Cnews はその特徴として、Cnews システム自体にシステム管理上有用な各種のコマンドが最初から付属してくるということを挙げることができる。ここではそれらのコマンドについて前段までに説明しなかったもので、かつ運用上有用ないくつかのものについてそれぞれ説明する。

NEWSCTL/inews

inews は、前段までに説明したように、従来の rn 等のニュースリーダ、あるいは従来の管理手法でニュースを管理する際の互換を取るために Cnews システムによって用意されたシェルスクリプトである。以前に説明したように日本語対応とするためには若干のパッチが必要となる。

またこの inews を用いてポストされたニュースには Lines:ヘッダが付加されない。これは RFC において Lines:が必須ヘッダと定義されていないためである (と Cnews の作者は説明している)。しかしながら inews シェルスクリプトには Lines:ヘッダを付加する部分がコメントアウトされて記述されているので、この部分のコメントをはずしてやることによって Lines:ヘッダを付加するように設定することができる。以下にそのパッチを示す。

```
*** inews.ORG   Sun Feb 23 23:36:25 1992
--- inews       Thu Mar 21 19:05:18 1992
***************
*** 176,182 ****
  (anne.jones <$inhdrs >/tmp/in$$realtmp        # bash headers
   status=$?
   mv /tmp/in$$realtmp $inhdrs
!  cat $inhdrs          # grep -v "^Lines:"   # uncomment if generating
 Lines:
   case $status in
   0)   ;;
   *)
--- 176,182 ----
  (anne.jones <$inhdrs >/tmp/in$$realtmp        # bash headers
   status=$?
   mv /tmp/in$$realtmp $inhdrs
!  cat $inhdrs          | grep -v "^Lines:"     # uncomment if generating
 Lines:
   case $status in
   0)   ;;
   *)
***************
*** 184,196 ****
        exit $status            # anne.jones was unhappy - bail out
```

```
        ;;
   esac
! # lines="`                    # sop to msb, just uncomment to use
! # (sed 1d $inbody;            # take out the first (blank) line
! # if test -r $HOME/.signature; then
! #     echo '-- '
! #     sed 4q $HOME/.signature
! # fi) | wc -l `"
! # echo Lines: $lines

   # strip invisible chars from body, a la B news.  bells and escapes are
 right out.
   case "$trversion" in
--- 184,196 ----
        exit $status            # anne.jones was unhappy - bail out
        ;;
   esac
! lines="`                      # sop to msb, just uncomment to use
! (sed 1d $inbody;              # take out the first (blank) line
! if test -r $HOME/.signature; then
!       echo '-- '
!       sed 4q $HOME/.signature
! fi) | wc -l `"
! echo Lines: $lines

   # strip invisible chars from body, a la B news.  bells and escapes are
right out.
   case "$trversion" in
```

また、Cnews 版の inews は 64512 バイト以上の大きさを持ったニュースアーティクルを受け付けない。これは通常の運用においてはほとんど障害にならないが、例えば inews 経由で (Bnews のように) 冗長 ihave を自前のシェルスクリプトで送出しようとした場合にうまく機能しない場合がある。これを防ぐには inews シェルスクリプトの 212 行目あたりにある max 変数の値、具体的には、

```
# 64512 = 63*1024 to allow for Path: growth
max=64512
```

をさらに大きな値に変更してやればよい。

NEWSBIN/ctime,getabsdate,getdate

ctime/getabsdate/getdate は、時間のフォーマットの変換を行なうコマンドである。ctime は 1970 年 1 月 1 日からの秒数 (NEWSCTL/log 等はこれを使用している) から ctime 形式 (UNIX の date コマンドの形式) への変換を行なう。getabsdate はそ

の逆を行なう。また gatdate は ARPADATE 形式 (ニュースアーティクルやメールの Date:フィールドに用いられる時間表記) から 1970 年 1 月 1 日からの秒数への変換を行なうプログラムである。それらの使用例を以下に示す。

```
% ctime 675496241
Wed May 29 14:50:41 1991
% getabsdate "Wed May 29 14:50:41 1991"
675496241
% getdate "Wed, 29 May 91 14:50:41"
675496241
```

NEWSBIN/expire/mkhistory

mkhistory は、history ファイルの再構築を行なうコマンドである。従来の Bnews システムでは、この機能は expire コマンドの一部 (expire -r) に組み込まれていたが、Cnews ではこの機能は expire とは独立している。

なおこの mkhistory コマンドはシェルスクリプトであり、また実行に非常に時間がかかる (find, awk 等を使用している) ことに留意すべきである。

NEWSBIN/expire/recovact

recovact は、active ファイルの第 2 フィールドを修復するコマンドである。これもかなりのシステム時間を要することに留意すべきである。

NEWSBIN/maint/newshist,histfrom,newsfrom

これらは、history の検索を行なうコマンド群である。newshist はメッセージ ID から history を検索し、該当するニュースアーティクルに関するエントリを表示する。使用例を以下に示す。

```
% newshist "<1992Mar26.150820.17632@wnoc-tyo-news.wide.ad.jp>"
<1992Mar26.150820.17632@wnoc-tyo-news.wide.ad.jp>       706892934~-
    control/47007
```

histfrom はある時間からある時間までのセグメント単位で history ファイルを検索、表示するコマンドである。時刻の format は ARPADATE 形式が使用される。使用例を以下に示す。

```
% histfrom "Thr, 26 Mar 1992 16:42:36" "Fri, 27 Mar 1992 00:00:00"
<1992Mar26.163359.23969@athena.mit.edu> 706898566~-     talk.politics.mis
c/24591 misc.legal/12022 soc.culture.usa/4697
<1992Mar26.164146.9285@news.ysu.edu>    706898569~-     news.software.rea
ders/1570
...
```

newsfrom はある時間からある時間までのセグメント単位で history ファイルを検索し、該当するニュースアーティクルの絶対パスを表示するコマンドである。時刻の format は ARPADATE 形式が使用される。使用例を以下に示す。

```
% newsfrom "Thr, 26 Mar 1992 16:42:36" "Fri, 27 Mar 1992 00:00:00"
/usr/spool/news/talk/politics/misc/24591
/usr/spool/news/news/software/readers/1570
...
```

NEWSBIN/maint/addgroup,delgroup

addgroup/delgroup はそれぞれローカルサイトにおいてニュースグループを追加、削除するコマンドである。すなわちこれらのコマンドは Bnews における inews -C コマンドおよび rmgroup コマンドに相当する。addgreoup/delgroup コマンドはローカルサイトにおいてのみ機能し、これらのコマンドによる設定変更は他の隣接接続サイト等には影響を与えない。この点で Bnews のそれに比較してより安全であるといえる。また delgroup コマンドは Bnews の rmgroup コマンドとは異なりそのグループ内のニュースアーティクル、およびそのグループの実体 (ディレクトリ) は消去されない。そのためもしそのニュースグループのアーティクル自体を消去したい場合は手でそれらの操作を行なう必要がある。

NEWSBIN/maint/addmissing

addmissing は、ニュースアーティクルの history への登録を行なうコマンドである。これにより従来の Bnews では history の再構築や rnews への入力等によって行なっていた history ファイルへの登録を、より簡便に行なうことができるようになっている。

NEWSBIN/maint/locknews

locknews は、ニュースシステムのロックを行なう際に使用するコマンドである。Cnews では Bnews と異なり、ニュースシステムのロックを単に NEWSCTL ディレクトリ下に **NEWSCTL/LOCKinews** 等のロックファイルを作成することによって行なっている。これを行なうコマンドが locknews である。

locknews コマンドを起動すると、ニュースシステムをロックした上で子シェルが起動される。すなわち管理者はこの環境を利用してニュースシステムのメンテナンスを行なうことができるわけである。これにより従来の Bnews システムに比較して、active ファイルの等の修正などの管理作業をより容易、かつ安全に行なうことができる。起動された子シェルを終了するとニュースシステムのロックは解除される。

1.4.5 ihave/sendme プロトコルの設定について

Cnews では、ニュースシステムにおける冗長 ihave/sendme プロトコル (以下単に「ihave/sendme プロトコル」とする) の実装メカニズム、およびその設定方法が Bnews と

は根本的に異なっている。ここでは Cnews システムにおいて ihave/sendme プロトコルを設定する際の方法、および運用の際の留意点について説明する。

従来の Bnews システムでは、ihave/sendme プロトコルは sys ファイルに対象システムが登録された時点で自動的に (Cnews の作者によると、これは強制的にということらしい) 機能していたが、Cnews システムにおいては単に sys に相手のシステム名を登録するだけでなく、自システムにおいて然るべき設定を行なわない限り ihave/sendme プロトコルは全く機能しない。

Cnews で ihave/sendme プロトコルを動作させるためには、以下の設定を行なう必要がある。

1. ディレクトリの作成
2. NEWSCTL/sys の設定
3. NEWSCTL/batchparms の設定

以下にこれらについて順に説明する。

ディレクトリの作成

NEWSARTS/out.going ディレクトリに *sysname*.ihave 、および *sysname*.sendme というディレクトリを news ユーザの権限で作成する。ここに ihave/sendme コントロールメッセージの中身 (アーティクルのメッセージ ID で構成されたファイル) が蓄積される。

NEWSCTL/sys の設定

次に sys ファイルを設定する。設定は基本的に以下のように行なう。

```
##
# foo, as NNTP site. Sending by nntpsend.
# And ihave/sendme protocol is now enable.
##
foo:all,!local,!to,to.foo\
  /all,!local,!ihave:F:
foo.ihave:all,!local,!to\
  /all,!local,!ihave,!sendme:I:foo.ihave/togo
foo.sendme:to.foo/ihave:I:foo.sendme/togo
##
# bar, as UUCP site. Sending by UUCP batch.
# If articles are already relayed host ‘‘baz’’, do not feed to bar.
# rec.* newsgroups dont send except to rec.arts.anime.
# And ihave/sendme protocol is now enable.
```

```
##
bar/baz:all,!local,!rec,rec.arts.anime,!to,to.bar\
  /all,!local,!ihave:f:
bar.ihave:all,!local,!rec,rec.arts.anime,!to\
  /all,!local,!ihave,!sendme:I:bar.ihave/togo
bar.sendme:to.bar/ihave:I:bar.sendme/togo
```

それぞれ 2 行目の *sysname*.ihave は相手システム (この例では foo および bar) に発信する ihave コントロールメッセージの素を作成するためのエントリである。また 3 行目の *sysname*.sendme は相手システムから発信されてきた ihave コントロールメッセージに応答して自システムから相手システムに発信する sendme コントロールメッセージの素を作成するためのエントリである。これらのエントリに対して sendbatches コマンドを用いて batch を行なうことによって、実際の ihave、ないしは sendme コントロールメッセージが作成され、該当する相手システムに送出される。

ここで、それぞれの sys ファイルのエントリの設定の意味について解説しておく。

1 行目の通常のエントリでは、ディストリビューションの部分に “!ihave” を追加して、他システムから送信されてきた ihave コントロールメッセージを中継してしまわないように設定している。

また *sysname* あて (to.*sysname*) の記事 (*sysname* あてのコントロールメッセージ) はこの行で処理されるため、ローカルシステムの inews によって作成された、相手システムに対する ihave/sendme メッセージはこの行によって相手に送信される。

2 行目のエントリでは、ihave コントロールメッセージを作成する素ファイルが作成される。ここではディストリビューションの部分の記述に “!ihave” および “!sendme” を追加して他システムから送信されてきた ihave コントロールメッセージや、あるいは自システムから送信される sendme コントロールメッセージ自身のメッセージ ID を ihave コントロールメッセージに含めないようにしている。

3 行目は、ihave ディストリビューションを持った記事 (ihave コントロールメッセージ) のみを処理し、ニュースグループ名フィールドで指定された相手システム (to.*sysname*) に送信する sendme コントロールメッセージの素を作成する処理を行なうための記述である。

NEWSCTL/batchparms の設定

ihave/sendme コントロールメッセージの送出処理を行なわせるために batchparms の設定を行なう。設定は基本的に以下のようになる。

```
##
# site                  size    queue   builder muncher sender
# ----                  ----    -----   ------- ------- ------
```

```
##
# foo, NNTP site, but ihave/sendme needs batchparms entry.
##
foo.ihave               50000   20      batchih nocomp  viainews
foo.sendme              50000   20      batchsm nocomp  viainews
##
# bar, UUCP site.
##
bar                     300000  5       batcher compcun viauuxz
bar.ihave               50000   20      batchih nocomp  viainews
bar.sendme              50000   20      batchsm nocomp  viainews
```

このように、*sysname*.ihave のエントリでは builder として batchih を、muncher として nocomp を、sender として viainews を指定する。同様に *sysname*.sendme のエントリでは builder として batchsm を指定する。あとはこれらのエントリに対して sendbatches を実行することで然るべき ihave/sendme コントロールメッセージが作成され、該当システムに送出される。

またこのように、ニュースのリンクに NNTP を用いているサイトであっても Cnews において ihave/sendme プロトコルを実装する場合には、NEWSCTL/batchparms の設定を行ない、定期的に sendbatches を起動するように設定しなければならないということに注意すべきである。

1.4.6　Cnews におけるコントロール・メッセージの処理メカニズム

Cnews では、cancel、ihave、sendme 以外のコントロールメッセージが受信された場合、NEWSBIN/ctl 下の同名のプログラムが起動され、そのプログラムにおいて実際の処理が行なわれる。すなわちこのプログラムの設定を改変することで、コントロールメッセージに対する応答を再設定することが可能である。現在の Cnews で用意されているものは以下の通りである。全てシェルスクリプトで提供されている。

```
checkgroups             rmgroup                 senduuname
newgroup                sendsys                 version
```

1.4.7　awk プログラムについて

Cnews では、ニュースアーティクル、あるいは active 等のシステムファイルの検索、加工に awk/grep/sed 等、UNIX に標準で付属しているテキスト処理プログラムが用いられている。それらのうちでも特に awk は Cnews システムのさまざまな部分から非常に頻繁に使用されているため、awk プログラムを GNU awk(gawk)、mawk 等のより高性能なも

のに差し替えるだけで、ニュースシステム自体のパフォーマンスがかなり (場合によっては数 10%以上) 向上することがわかっている。

Cnews システムにおいて、これらのより高性能なプログラムを標準提供のプログラムと差し替えて利用するには、Cnews が NEWSBIN(/usr/lib/newsbin) を常にコマンドサーチパスの先頭として使用していることを利用するのが適当である。例えば/usr/bin/awk のかわりに/usr/local/bin/mawk を使用したい場合には、/usr/lib/newsbin に cd したうえで、

```
news% cd /usr/lib/newsbin
news% ln -s /usr/local/bin/mawk awk
```

のようにシンボリックリンクを張ることで、Cnews 自体に変更を加えることなく mawk を awk に替えて利用することが可能になる。

1.5 Bnews からの移行

Cnews のドキュメントによると、Bnews から Cnews へ移行する場合にはいったん双方のシステムを同時に動作させておいて移行する方法をとることが可能であると述べられている。しかし実際にはシステム資源の都合、あるいはさまざまな混同を防ぐため、いったん Bnews システムを停止したうえで Cnews をインストール、設定することが多い。そこでここでは Bnews システムから Cnews システムへ移行する際の手順、およびさまざまな注意事項について説明する。

1.5.1 移行の手順例

Bnews から Cnews への移行の際の一連の手順について説明する。これはあくまで一例であり、よりスマートな移行方法も考えられるであろう。

設定ファイルの作成

まず Cnews のインストール以前に Bnews のそれを参考にしながら Cnews の NEWSCTL 下の設定ファイルを**あらかじめ作成しておく**ことが Cnews への移行作業をよりスムーズなものにするであろう。そのためにはまず Cnews のドキュメントをまずよく読んでおくことが必要である。対象になるファイルとしては、active、batchparms、explist、mailpaths、sys などがこれにあたる。

mailname、whoami は Cnews のインストール時に自動作成されるため、ここでは作成しておく必要はない。

Bnews のシステムの停止

その際 BATCHDIR に残っているファイルは、後で送信を再開する際に使用できるように保存しておくのがよい。

nntpd が立ち上がらないように inetd.conf を修正する。また UUCP も一時的に停止しておく。crontab を修正して sendbatch 等が動作しないようにする。また SPOOLDIR/.rnews ディレクトリの内容がその時点で空であることを確認しておく。そうでない場合は rnews -U を実行して空にしておく。

Bnews 環境の保存

LIBDIR(/usr/lib/news) の内容を tar 等で保存しておく。しばらくはいつでも内容をとりだせるようディスク上に保存しておくのがよいであろう。

Cnews システムの構築

Cnews システムをインストールする。

この際気をつけなければならない点として、Bnews では inews と rnews はリンクされていたため、

/usr/bin/inews → /usr/lib/news/inews
や
/usr/local/bin/rnews → /usr/lib /news/inews

といったシンボリックリンクが存在している場合がある。この環境で Cnews をそのままインストールしてしまうと Cnews では inews と rnews は別のプログラムであるため、inews が rnews に上書きされてしまうことになる。そのためこのようなシンボリックリンクが存在するおそれがある場合には doit.bin 等のシェルスクリプトを実行する以前に必ずこのようなリンクが存在しないことをチェックしておかなければならない。

nntpd の再インストール

nntpd を Cnews/dbz に対応させ、再インストールする。具体的には nntp キットの conf.h を修正する。

設定ファイルのインストール

事前にあらかじめ作成しておいた NEWSCTL 下の設定ファイルを然るべき位置にインストールする。

history の再構築

Cnews の history ファイルを作成する。mkhistory コマンドは非常にシステム時間を要するため、従来の Bnews で使用していた history ファイル (テキスト) を元に Cnews の history ファイル (history.dir,history.pag) を作成してもよい。しかしながら Bnews の history テキストファイルと Cnews のそれとでは若干違いがあることがあるため、Cnews に付属の dbz コマンドによる再構築がうまく機能しない場合もあることに注意すべきである。その場合でも sed や awk を history テキストファイルに対してうまく使うことによって、dbz コマンドで処理することが可能な形式に変換することはそれほど難しいことではない。

SPOOLDIR/.rnews,.bad,.tmp ディレクトリの削除

これらのディレクトリは Cnews ではもはや使用されないため、内容が空であることを確認したうえで削除しておく。

また BATCHDIR を/usr/spool/news/batched 等、/usr/spool/news 下のディレクトリに設定していた場合はこのディレクトリも忘れずに削除しておく。

UNIX システムファイルの設定

crontab,rc.local 等を設定し、inetd.conf 等の設定を元に戻す。

1.6 nntplink

近年、広域IP ネットワーク環境の普及に伴い、従来よりもニュースアーティクルの転送速度をより高速化、かつ高信頼化しようという試みが広域ネットワーク上で行なわれるようになってきた。その試みの 1 つとして最近 Cnews とともにかなりの割合で普及しはじめているソフトウェアパッケージに nntplink と呼ばれるものがある。ここでは nntplink パッケージについてその特徴、導入、設定などについて説明を行なう。

1.6.1 nntplink とは

nntplink は Ohio State Univercity の David Alden が作成した NNTP プロトコルを用いたニュース転送をより円滑に行なうためのソフトウェアパッケージである。

nntplink には以下のような特徴がある。

nntplink サーバの存在

従来の nntpsend 等に代わって、nntplink という NNTP によるニュースの送信を専門的に行なうサーバが実際のニュースの処理を行なっている。

NNTP リンク数に対応した nntplink サーバ

nntplink では、NNTP のリンク 1 つに対して nntplink サーバが 1 つ常駐する形をとっている。すなわち nntplink がインストールされたシステムでは nntplink サーバが NNTP のリンクの数だけ常駐している。

ニュースシステムにアプライするパッチの存在

nntplink には、Cnews システムに対してアプライするパッチパッケージが付属している。このパッチをあてると NEWSCTL/log が nntplink に対応した形式となり、また NEWSBIN/relay/relaynews がデーモンとして常駐するように変更される。このため若干ニュースシステムの管理手法が標準の Cnews と異なる点が存在する。これについては後述する。

また nntplink は Bnews にも対応しておりパッチパッケージも付属しているが、現在までに Bnews システムでの nntplink の運用実績は WIDE プロジェクトにおいては報告されていない。

1.6.2　nntplink の入手

nntplink は、*sh.wide.ad.jp*[133.4.11.11] をはじめ、国内の主な anonymous ftp サイトから入手することが可能である。本報告書執筆時の nntplink の最新バージョンは **3.0.2** である。3.0.1beta 以降 nntplink はかなりのバグフィックス、および Cnews における設定方法がかなり変更されているため、注意が必要である。本報告書では nntplink バージョン 3.0.2 を “22-Dec-1991” 版の Cnews と組み合わせて使用した場合の設定について説明する。

1.6.3　nntplink の導入

nntplink は通常、`NEWSBIN/local` ディレクトリにインストールされる。nntplink 本体、および nntplink キット中にある `sup/links` シェルスクリプト、および `links-list` ファイルがインストールされる。それらのうち `links-list` ファイルはニュースシステムの設定に関するものなので、他の Cnews における設定ファイルと同様 `NEWSCTL` からも参照可能なように、

```
% cd /usr/lib/news
% ln -s /usr/lib/newsbin/local/links-list .
```

のようにシンボリックリンクを張っておくとよい。

また nntplink 3.0.2 を標準の設定でコンパイルした場合、`NEWSBIN/local/links` シェルスクリプトを以下のように修正しておく必要がある。これは nntplink が使用するステータスファイルの位置が `links` シェルスクリプトが想定しているそれと異なっているためである。

```
*** links.ORG   Mon Feb 24 02:54:59 1992
--- links       Wed Feb 26 22:52:47 1992
***************
*** 48,55 ****
  # What goes on the end of "${batchdir}/${site}" to get the link datafile?
  # (ie: should we use "${batchdir}/${site}/togo.link" or should we use
  # "${batchdir}/${site}.link"?)
! rest=/togo.link
! # rest=.link

  #
  # command to display all of the processes currently running
--- 48,55 ----
  # What goes on the end of "${batchdir}/${site}" to get the link datafile?
  # (ie: should we use "${batchdir}/${site}/togo.link" or should we use
  # "${batchdir}/${site}.link"?)
! # rest=/togo.link
```

```
! rest=.link

  #
  # command to display all of the processes currently running
```

1.6.4 nntplink の設定

Cnews に nntplink に付属のパッチをアプライし、nntplink を導入した後の nntplink の設定を以下に説明する。

NEWSCTL/links-list ファイル

nntplinkでは、links-list というファイルでその挙動の設定を行なっている。links-list ファイルの例を以下に示す。

```
#
# List of sites which "links" should start up nntplink processes with
#
# NOTE: Comments must be of the form "# ..." - meaning there must be
#       a space after the "#" character.
#
# pri   site                                    options
# ---   ------------------------------------  ----------------------
# WNOC-TYO-NEWS
    0   wnoc-tyo-news.wide.ad.jp               -s wnoc-tyo-news
# WNOC-KYO
    0   wnoc-kyo.wide.ad.jp                    -s wnoc-kyo
# WNOC-SND-SS2
    0   wnoc-snd-ss2.wide.ad.jp                -s wnoc-snd-ss2
```

links-list ファイルは 3 つのフィールドで構成される。

第 1 フィールドは nntplink サーバが起動される際のプライオリティを記述する。これは links シェルスクリプトによって nntplink サーバ起動時の/bin/nice コマンドに引き渡されている。通常は 0 を指定しておく。

第 2 フィールドは nntplink サーバがニュースを送る相手のホスト名を記述する。

第 3 フィールドは nntplink サーバに渡されるオプションを指定する。通常の設定では上記の例のように、"-s *sysname*" と指定すれば充分である。この設定における *sysname* は、相手のニュースシステム名を指定する。ニュースシステム名とは相手のニュースシステムが Path: フィールドに自分のシステム名として付加している名称のことである。

NEWSCTL/sys ファイル

この設定を行なった場合、nntplink サーバは「logfile モード」で動作する。logfile

モードとは NEWSCTL/log に出力されるシステムログを nntplink サーバがチェックし、それによってターゲットシステムにニュースを 1 つずつ送出するモードのことを指している。

この他にも nntplink には batchfile モード、stdin モードという 2 つの動作モードが存在する。それらの設定方法については nntplink に付属のドキュメント類を参照されたい。

logfile モードで nntplink を動作させる場合、NEWSCTL/sys ファイルを nntplink 用に再設定する必要がある。具体的には、従来 nntpsend 等を使用して送っていた行のエントリを、単に logfile にシステム名が表示されるだけで Cnews 自体は何もしないように設定することとなる。すなわち以下の通りである。

```
##
# foo, as NNTP site. Sending by nntplink.
# And ihave/sendme protocol is now enable.
##
foo:all,!local,!to,to.foo\
  /all,!local,!ihave:F:/dev/null
foo.ihave:all,!local,!to\
  /all,!local,!ihave,!sendme:I:foo.ihave/togo
foo.sendme:to.foo/ihave:I:foo.sendme/togo
```

上記のようにバッチファイルとして/dev/null を指定する。また ihave/sendme に関連するエントリに関しては nntplink への変更に際し特に変更点はない。このことは nntplink によるニュースの転送を行なう場合においても、ihave/sendme プロトコルを使用する場合には、NEWSCTL/batchparms の設定を行ない、定期的に sendbatches を起動するように設定しなければならないということを意味している。

rc.local への設定の追加

システム起動時に nntplink の立ち上げ、およびデーモン化された relaynews の立ち上げを行なうよう、rc.local あるいはそれに相当するファイルを変更する。具体的には以下の行を追加すればよい。

```
#
# relayrun daemon
#
if [ -f /usr/lib/newsbin/relay/relayrun ]; then
        (echo "relayrun daemon")                        >/dev/console
        rm -f /usr/lib/news/LOCK*
        /usr/lib/newsbin/relay/relayrun -D
fi
#
```

```
# nntplink start
#
if [ -f /usr/lib/newsbin/local/links -a -f /usr/lib/news/links-list ]; then
        (echo "nntplink daemon")                        >/dev/console
        /usr/lib/newsbin/local/links start
fi
```

nntpd の再インストール

relaynews をデーモン化した場合、nntpd の再設定を行なう必要がある。具体的には、到着したニュースが蓄積するディレクトリを変更する必要がある。

```
*** common/conf.h.dist Sat Feb  9 09:31:07 1991
--- common/conf.h      Mon Feb 24 03:35:01 1992
***************
*** 262,268 ****
  /*
   * Support for C-News style batching
   */
! #undef  NONEWSRUN             /* define this if you are using the
daemon */
                                /* version of relaynews */
  #ifdef NONEWSRUN
  #define TOOBIG 1L
--- 262,268 ----
  /*
   * Support for C-News style batching
   */
! #define  NONEWSRUN            /* define this if you are using the dae
mon */
                                /* version of relaynews */
  #ifdef NONEWSRUN
  #define TOOBIG 1L
***************
*** 276,282 ****
  #define COPYSIZE 8192         /* bytes to copy at one time */
  #define MAXDIGITS 25          /* lg(maxlongint) + epsilon */
  #define MAXSTR 1024
! #define INDIR                 "/usr/spool/news/in.coming"
  /* You may wish to delete the pathname from the front of BATCH_FILE */
  #define BATCH_FILE            "/usr/spool/news/in.coming/nntp.XXXXXX"

--- 276,282 ----
  #define COPYSIZE 8192         /* bytes to copy at one time */
  #define MAXDIGITS 25          /* lg(maxlongint) + epsilon */
  #define MAXSTR 1024
```

```
! #define INDIR                  "/usr/spool/news/in.coming/daemon"
  /* You may wish to delete the pathname from the front of BATCH_FILE */
  #define BATCH_FILE             "/usr/spool/news/in.coming/nntp.XXXXXX"
```

1.6.5　relaynews について

relaynews がデーモン化されているため、NEWSCTL/sys 等の Cnews 設定ファイルを変更した場合には、必ず常駐している relaynews デーモンに対して HUP シグナルを送らなければならない。

このためのファイルとしてデーモン化された relaynews は、NEWSCTL/relaynews.pid というファイルを作成している。すなわち

```
news% kill -HUP ‘cat /usr/lib/news/relaynews.pid‘
```

のような形式で使用することが可能である。

第 2 章

sendmail

広域 IP ネットワークに接続されたホスト間において電子メールの送受信を支援するプロトコルとして、DARPA インターネットサービスの一つである SMTP [?] が用いられている。ここでは UNIX における SMTP の実装である sendmail についてその設定などについて記述する。

2.1 さまざまな sendmail

現在各ベンダにより各マシンに実装されている sendmail にはさまざまなバージョンが存在することが確認されている。それらを大別すると、(1)Berkeley 系、(2)NFS 系の 2 つに大別することができる。また WIDE プロジェクトで開発された (3)WIDE 版 sendmail もかなり広く用いられているようになってきた。以下それぞれのバージョンの相違による機能および動作の違いについて説明する。

なお、sendmail のバージョンは通常の sendmail.cf を用いている限り、各メールヘッダの Received:行に、

```
Received: from ucbvax.Berkeley.EDU by jp-gate.wide.ad.jp (5.67+1.6W/2.8Wb)
        id AA04871; Mon, 18 Mar 92 18:25:11 JST
```

のように、(sendmail のバージョン/sendmail.cf のバージョン) という形式で表示されるので、それによって判断することもできる。ローカルな sendmail のバージョンは、

```
% /usr/lib/sendmail -bt -d21.8 < /dev/null
```

によっても知ることができる。

2.1.1 Berkeley 系 sendmail

Berkeley 系 sendmail とは、4.xBSD UNIX のディストリビューションに含まれている sendmail を示す。1992 年 3 月時点で公開されているこの版の sendmail の最新版はバージョン **5.67** である。これは U.C.Berkeley より無償で公開されている Net2 リリース中に含まれているものである。すなわち sendmail は、U.C.Berkeley が AT&T フリーである

と宣言したネットワークコードの中に含まれており、無償で入手することが可能である。以下、現在出回っている主なマシンに実装されている Berkeley 系 sendmail のバージョン、およびその機能の相違について説明する。

4.12 4.2BSD のディストリビューションに付属の sendmail のバージョンである。NEWS-OS 2.x 等、4.2BSD 相当のネットワークコードを持つ UNIX に実装されている。また一部のΣ規格のマシンには、現在もこのバージョンの sendmail が実装されていることが確認されている。この sendmail はかなり古く、さまざまな不都合や障害があることが判明しているため、手元のマシンの sendmail がこのバージョンであった場合、至急新しいバージョンのバイナリをベンダより入手するか、または新しい sendmail のソースを入手、インストールする必要がある。

5.51 4.3BSD のディストリビューションに付属の sendmail のバージョンである。初期の 4.3BSD 相当のネットワークコードを持つ UNIX ワークステーションに実装されている。この sendmail は、以前のバージョンである 4.12 と比較してかなり品質も向上しており、比較的安定に動作するが、debug コマンドというネットワーク上非常に大きなセキュリティホールが存在しているため、新しい sendmail を入手し、入れ換えを行なう必要がある。

5.61 U.C.Berkeley から、4.3BSD-Tahoe Release の network free software として 1988 年に公開された sendmail のバージョンがこれである。このバージョンは 5.51 に存在した様々な障害の修正が行なわれており、またライセンスフリーであるため、近年まで広域ネットワークにおいてかなり広く用いられていた。またこのバージョンは広域ネットワーク環境においてメールを配送する際に非常に重要な要素となるネームサーバの MX レコードをサポートしており、より効率的なメール配送を行なうことが可能である。しかしながら MX の検索を有効にするには、コンパイル時にあらかじめ conf.h で NAMED_BIND フラグを定義しておく必要がある (当然 MX レコードはそれ以降のバージョンにおいてもサポートされている)。

5.64 このバージョンはメールをキューイングする際に、メールファイルの保護のために flock() を用いていることが特徴である。そのため NFS が実装された UNIX でこの版の sendmail を用いる場合、sendmail の起動を inetd および rpc.lockd の起動よりも後で行なうように変更しなければならない。このため例えば SunOS 等では rc.local を修正する必要がある。

5.65 5.65 は 5.64 に対する軽微なバグを修正したものであり、5.67 とともに現在広域ネットワーク環境において最も広く使用されているものである。

5.67 現在 U.C.Berkeley より公開されている最新版のものである。5.67 では 5.65 と比較して、メールの内部キューの扱いに若干変更が見られるほか、一度に多数の宛先を指定してメールを送信する際の信頼性を向上させるためのコードを含んでいることに特徴がある。

2.1.2 NFSSRC に付属の sendmail

NFSSRC パッケージには Berkeley 系 sendmail を元に Sun Microsystems がいくつかの改良および機能の追加を行なった sendmail が含まれている。これは NFS 系 sendmail と呼ばれており、やはりいくつかのバージョンが存在している。またバージョン番号の管理は Berkeley 系とは別個に行なわれており、対応する NFSSRC のバージョンに合わせている。この sendmail のソースを利用するには、対応する NFSSRC のライセンスが必要になる。以下、主なバージョンの相違について説明する。

3.2 NFSSRC 3.2 に付属の NFS 系 sendmail である。SunOS 3.x に標準実装されている。Berkeley 4.12+α程度のバージョンを元にしているため、4.0 以降のものにリプレースした方が良い。

4.0 NFSSRC 4.0 に付属の NFS 系 sendmail である。SunOS 4.0.x、NEWS-OS 3.x/4.0 等に標準実装されている。このバージョンは当初 Berkeley 5.51+αのバージョンを元にしていたが、上記のセキュリティホールが明らかになった時点で Sun から Berkeley 5.59 に準拠したオフィシャルパッチがリリースされた。またこの 4.0 版から MX レコードをサポートした sendmail.mx が付属しているが、オリジナルの SunOS 4.0.x に付属のものにはバグがあり、sendmail.mx であるにもかかわらず MX レコードを参照しないという不具合が知られている。これを修正したパッチもオフィシャルパッチとして入手可能である。

4.1 NFSSRC 4.1 に付属の NFS 系 sendmail である。SunOS 4.1.1 に含まれているものには、ある種の security hole が存在することが知られており、Sun から sun-fixes として Berkeley 5.64 に準拠したオフィシャルパッチがリリースされている。また SunOS 4.1.1 に付属の sendmail.mx には、MX 検索の際の_res 構造体のフラグの設定が間違っているため、MX 検索が正常に動作しないというバグがあった。これを修正したパッチも前述の sun-fixes に含まれている。

NFS 系 sendmail の 4.0 以降のバージョン (NFSSRC 4.0 以降のもの) には、Sun Microsystems により独自に付加されたいくつかの拡張機能が含まれている。以下それについて説明する。

1. sendmail.cf 中で定義されるクラスにドメイン名を用いることが可能である。Berkeley 系 sendmail では、アドレス解釈の際に “.” がデリミタとして解釈されてしまうため、クラスにドメイン名を用いることが出来ない。

2. $m マクロにそのホストが属する NIS ドメイン名が代入される。このドメイン名はネームサーバにおけるドメイン名ではないことに注意されたい。また後述の WIDE 版 sendmail の sendmail.mx では、起動時に$m マクロにネームサーバにおけるドメイン名が代入されるように改良されている。

3. いくつかの NIS 関連の拡張機能をサポートしている。これについては SunOS 等に付属のドキュメントを参照されたい。

4. 宛先アドレス別のキューの処理、メッセージ ID を指定したキューの処理をサポートしている。Berkeley 系では linear なキューの処理以外はサポートされていない。

5. postmaster へ届くフェイルメールに本文が入らず、ヘッダだけが送られる。これによってある程度のプライバシーを保護することができる。

6. “-bt” オプションを用いてデバッグを行なう際に、“`^V`”, “`^W`” といった出力の代わりに “`$#`”, “`$@`” のように、出力が読みやすくなっている。

2.1.3　WIDE 版 sendmail

国内の広域ネットワークでは、WIDE Project で Berkeley 系 sendmail にパッチを加えることにより開発された、“WIDE 版 sendmail” もかなりの割合で使用されている。これは前述の NFS 系 sendmail において存在する付加機能を、NIS 関連以外は全てサポートしており、またその他 mail キューの処理、より細かな SMTP のロギング機能等、独自のさまざまな拡張機能を含んでいる。また WIDE 版 sendmail は Berkeley 版 sendmail と完全に上位互換を保つように設計されており、従来の Berkeley 系と同一の sendmail.cf で動作させることが可能である。この WIDE 版 sendmail に新たに付加された機能のコードは NFSSRC とは無関係であるため、このソースコードは NFSSRC 等のライセンスによる制限を受けない。

現在の WIDE 版 sendmail は Berkeley 版 sendmail 5.67 に対してアプライするパッチパッケージとして提供されており、現時点における最新バージョンは 1.6W である。この sendmail のバージョン番号は、「元になった Berkeley 版のバージョン番号+WIDE 版コードのバージョン」という形式のバージョン番号を出力するように設定されている。

2.2　sendmail を動作させる際のポイント

ここでは、広域ネットワーク環境で Berkeley 系 sendmail を動作させる際に注意すべきいくつかの点について説明する。

2.2.1　コンパイル時の注意

sendmail をコンパイルする際に、**conf.h** というヘッダファイルでの設定がその sendmail の動きを決定する。以下、コンパイル時に重要な意味を持つ conf.h 中の 2 つのフラグについて説明する。

NAMED_BIND

SMTP によるメール配送の際にネームサーバの MX レコードをサポートするか否か

を決定するフラグである。これを定義しておくと SMTP によるメール配送の際に MX レコードが用いられる。

NO_WILDCARD_MX
sendmail.cf 中で$[$] マクロでホスト名をサーチする際に、デフォルトのドメイン名を付加するか否かのフラグと考えてよい。日本の広域ネットワーク環境で用いる場合には、このフラグは off (すなわちコメントアウト) しなければならない。

また conf.h 中では各フラグは

```
#define NAMED_BIND 1
```

のように 1 と定義している。ところが、sendmail のソース内では全てのフラグは#ifdef で参照されているため、各フラグの設定を disable にする場合には、1 を 0 にするのではなく、**該当する行全体をコメントアウトしなければならない**点に注意すべきである。

MX レコードをサポートした sendmail を作成する場合には、

```
#include <resolv.h>
#include <arpa/nameser.h>
#include <netdb.h>
```

等のヘッダも忘れずにアップデートしておく必要がある。また勿論リンクされる resolver ライブラリ (libresolv.a) も最新のものに入れ換えておくの必要がある。

バージョン 5.64 以降の sendmail では、4.3BSD-Reno の pmake 環境用の Makefile しか付属していないため、これをコンパイルするためにはバージョン 5.61 に付属の Makefile を修正して使用するのがよい。

ある種の OS では、未解決な名前のためリンクが正常に行なわれない場合がある。その場合は配布キット中の./support ディレクトリに含まれているライブラリをリンクしてみるとうまくいく場合が多い。この./support ディレクトリには、4.3BSD-Tahoe の libc ライブラリの一部、および関連するヘッダファイル等が含まれている。これらはいずれもライセンスフリーである。

sendmail がロードアベレージを取得する際に用いている getloadavg( ) がサポートされていない OS の場合は、X-Window の配布キットに含まれている getloadavg.c を用いると良い。このソースはほとんどの UNIX ワークステーションに対応している。またこれを用いる場合、例えば NEWS-OS のように OS のバージョンの相違、および RISC/CISC アーキテクチャによってカーネル内のロードアベレージの持ち方が異なっている場合があるので設定には十分注意する必要がある。

2.3 sendmail.cf ファイル

ここでは、sendmail の挙動を決定する sendmail.cf ファイルの設定について記述する。

2.3.1　広域ネットワーク環境におけるメールの配送

広域ネットワーク環境においては、外部組織へ発信されるメールに関してはよほどの特殊な事情がない限り MX レコードを用いた配送が行なわれるべきである。これによって、従来の static な経路設定に比較して経路設定/経路変更が容易になる上、不慮のリンクダウン等に対しての対応が容易になる。また、sendmail.cf ファイルを小型化できるため、sendmail が消費するメモリを節約することもできる。sendmail は頻繁に fork するため、パフォーマンスに関してもこれは有効である。MX レコードに関する詳細は、ネームサーバの章を参照されたい。

2.3.2　sendmail.fc ファイルについて

フリーズファイル (sendmail.fc) は sendmail が何度も exec(fork ではない) される環境、たとえば実ユーザが非常に多いホスト、ないしは UUCP のゲートウェイホストなどでない場合、作成しない方が安全である。特に、

```
# sendmail -bt -C./test.cf
```

のように、標準ではない構成ファイルのテストを行なう際に/etc/sendmail.fc ファイルが存在すると、一部のデータをオリジナルの/etc/sendmail.fc から取り込んでしまうという障害があることが報告されている。

また SunOS 4.0 以降で sendmail を make する場合、sendmail.fc ファイルを使用した場合には、暫くは正常に動作するが、時間が経過するにつれ、SMTP の RCPT TO コマンドなどの動作がおかしくなることがあることが報告されている。従って、sendmail を Dynamic Linking ではなく Static Linking (-Bstatic) で生成するか、sendmail.fc ファイルを使用しないように注意しなければならない。

また SunOS 4.1.2 に付属の sendmail には、sendmail.fc ファイルを作成した場合に local ユーザにメールが正常に配送されないというバグが存在することが報告されている。sendmail.fc ファイルを作成しない限りこれは発生しない。

2.3.3　MX に対応した sendmail.cf

現在 WIDE Internet をはじめとする国内広域ネットワークで主に用いられている MX 検索に対応した sendmail.cf には大きく分けて次の 3 つがある。

WIDE 系

WIDE Project で開発されているもので、もともとは Berkeley の sendmail のディストリビューションに含まれている ucbarpa の sendmail.cf を基に、“user@host.BITNET” 等のアドレスの処理、およびさまざま改良を重ねて現在に至っているバージョンである。現在広く利用されている WIDE 系 sendmail.cf のバージョンは “2.7W” というバージョンである。これは一般には「WIDE 版 sendmail.cf」と呼ばれている。ま

た現在、WIDE 版および NFS 版 sendmail の拡張機能がサポートされていることを前提にシェイプアップされた新バージョンの sendmail.cf(2.8W-beta) も数箇所で試験的に運用されており、現在の所ほぼ問題なく動作しているようである。なお、NEWS-OS 4.x では/usr/lib/sendmail.mx.cf に MX 対応の sendmail.cf が標準で提供されている。

TISN 系
TISN で使用されているものである。これは 4.3BSD に付属の sendmail.cf に TISN で改良を加えたものである。詳細は TISN に問い合わせたい。

mailconf 系
従来の JUNET 用 sendmail.cf 生成ツールである mailconf 6.4J に簡単な改良を加え、MX レコードの検索サポートを追加したものである。設定が比較的容易であるのと、従来の環境の延長で使用できるためにこれもかなり広範囲に用いられている。この mailconf はこのパッチの作者である東京大学の平原正樹氏の名前から、「平原パッチ」とも呼ばれている。

以下では、WIDE Project において最も一般的に用いられている 2.7W を例に実際の sendmail.cf の設定について説明する。mailconf 版については、mailconf および平原正樹氏のドキュメントを参照されたい。

2.3.4 2.7W の設定

2.7W においては mailconf とは異なり、サンプルの sendmail.cf のみが提供されている。それをファイルの説明にあるコメントを頼りに自分のドメインに合わせてカスタマイズしていくことになる。ここでは 2.7W をカスタマイズする際にポイントとなるいくつかの点について説明する。またいずれにせよ現在 2.7W を用いる場合、sendmail.cf に関するある程度の知識が必要になる。

1. $w マクロには自分の名前がドメイン形式で代入される 2.7W に限らず、MX 対応の sendmail.cf では hostname(1) で出力されるホスト名がドメイン形式かそうでないかに関わらずドメイン形式の名前が代入される。ドメイン形式の名前というのは、例えば *jp-gate.wide.ad.jp* のような "." で区切られた名前のことをいう。このため、例えば *jp-gate!user*(あるいは user@jp-gate.UUCP) のような形式のアドレスを扱うためには、あらかじめ予約クラス w に対して、

   ```
   Cwjp-gate
   ```

   という記述を sendmail.cf で追加してやる必要がある。なおこれは、現在の WIDE 版 sendmail では自動的に行なわれているため、明示的にこのような記述を行なう必要はない。

2. “*user@mydomain.jp*” という形式をローカルに処理させたい場合 (ドメインマスタにおける設定)、ルールセット 0 にある、

```
# domain that I take it as local        XXXXXXXX
#R<@$D>:$*      $@$>0$1                 @mydomain:... -> ...
#R$*<@$D>       $@$>0$1                  ...@mydomain -> ...
```

という行のコメントを外す必要がある。これによりそのホストはそのドメインのドメインマスタとして振舞うようになる。

3. 2.7W はローカルな UUCP 接続によるメールの配送も処理することが出来る。そのためには、ruleset 0 にある

```
# domain that connects this host by uucp to gateway mapping XXXXXXXX
#R$*<@$*DOMAIN_NAME>$* $:<@GATEWAY.UUCP>:$1<@$2DOMAIN_NAME>$3
```

というエントリの後に、

```
R$*<@$*dom1.xx.jp>$*    $:<@dom1gw.UUCP>:$1<@$2dom1.xx.jp>$3
R$*<@$*dom2.yy.jp>$*    $:<@dom2gw.UUCP>:$1<@$2dom2.yy.jp>$3
```

のようにエントリを追加してやればよい。

2.3.5　海外 reahcble でないホストにおける設定

海外 reachable でない IP 接続ホストからの海外向けメールの送信に関しては、適切な該当ホスト (WIDE では jp-gate.wide.ad.jp、TISN では utsun.s.u-tokyo.ac.jp など) に static にリレーする形式を取るが、この場合の設定に関しては必ずそのリレーとして指定するホストの管理者、およびそのホストが所属するネットワークコミュニティの承認を得た上で行なわれるべきである。その場合、ruleset 0 の後半部で mailer へ渡す部分を改造することにより行なわれることになる。

2.3.6　_ip_forwarding を off にした場合

一般的な MX レコードによるメールの配送は全てのホストに IP パケットが届くことを想定している。そのため UNIX カーネルの_ip_forwarding を off にした場合、さまざまな特殊な細工を sendmail.cf 等の設定ファイルで行なう必要がある。

以下にいくつかの例を示す。

1. 自ドメイン内へのメールの配送には MX レコードを用いない方法。
   static に自ドメイン内の特定ホストへメールを配送するように ruleset 0 を書き換えることで実現することができる。この際注意すべきことは、そのホストにホスト名をそのまま記述してしまうと、今度はそのホスト名に対する MX の参照が起こってしまうことである。これは 2.7W では ruleset 0 を書き換える (IP アドレスを直接 sendmail.cf に記述する。ないしは WIDE 版 sendmail を用いている場合には [hostname.domainname] のようにホスト名を [ ] 内に記述する) ことによって回避で

きる。また mailconf 6.4J では配送先を IP アドレスで設定する (直接 IP アドレスを host.dat に記述する) ことで実現することができる。

2. MX レコードを記述する際に組織外から unreachable な内部のホストを最初に記述してしまう方法。
   最初にトライするホストに、外から届かないホストを書いてしまうという方法が用いられることがある。この場合外部からの MX レコードの参照の際に、最初の試行は必ず失敗することになり、スマートな方法ではない。しかし、そのホストへのアクセスに対して、ゲートウェイにおいて ICMP Host Unreach を返すよう設定しておくことによって、ロスタイムを実用上問題にならない程度に抑えることができる。

3. “中向け” の MX を別系統とする方法。
   中向けの配送に際しては、自らのホストで動作しているネームサーバを参照せず、resolv.conf の記述により内部の別ホスト上のサーバを参照する、という方法が用いられることがある。この場合、内部用と外部用のネームサーバを 2 系統管理しなければならないというオーバーヘッドがあるが、内部、外部を問わずにすべてのメールの配送を MX レコードによることができるという利点がある。すなわちこの方法では sendmail.cf において特殊な記述をする必要がないわけである。

4. SMTP をリレーする方法。
   SMTP を内部の特定のホストの sendmail にリレーするデーモンがゲートウェイホストの SMTP のポートをオープンし、SMTP のメッセージをすべて内部のホストに転送し、そこで処理する、という方式である。この方式は NNTP 等、他のネットワークプロトコルにも応用が可能である。

現在の所、「これが一番良い」という方法は今だ確立されておらず、各参加組織がそれぞれの都合に一番合致したさまざまな方法を用いているのが現状である。米国の Internet においてもその状況は同様のようである。

これらの非標準の設定を行なう際には、いずれも設定を誤った場合、sendmail が自分自身に SMTP してしまう “mail loops back to myself” (5.65 までの “Local Configuration Error”) によるメールフェイルが起こりやすいため、いずれの場合も設定には細心の注意が必要である。

2.3.7 その他のメモ

- I オプション
  5.61 以降の MX レコード対応の Berkeley 版 sendmail に対して有効であり、named と “Connection refused” エラーで通信できなかった際に、そのメールを “temporary fail” にするためのフラグである。この sendmail を使用している場合、sendmail.cf で OI を定義しておいた方がよい。

また現在の WIDE 版 sendmail では、sendmail.mx の場合にはこのフラグの設定いかんにかかわらず自動的に OI フラグが設定されたのと同様の動作を行なうようコンパイル時に設定される。

第 3 章

BIND ネームサーバ

DNS(Domain Name System) [?] [?] は階層的な名前空間を提供し、それに対する分散的な名前サーバを提供しているものである。広域ネットワークにおいては、ホスト名や IP アドレス、メールサーバホストなどの情報がネームサーバによって管理・運用されている。ここでは 4.3BSD における DNS の実装の 1 つである BIND ネームサーバ (以下 BIND) についてその設定などについて説明する。

なお、現在の BIND のバージョンは **4.8.3** である。このバージョンは zone データの転送などで 4.8 以前のものに比べて大きく改善されているので、手元の BIND が古い場合、最新版を入手しそれをインストールする必要がある。なお SunOS 4.1.x、NEWS-OS 4.x では BIND は 4.8.3 相当のものになっている。

3.1 zone データ作成上のヒント

3.1.1 ファイルの位置

BIND の各ファイルは全部を一つのディレクトリにまとめておくと管理が楽である。例えば、/etc/ns というディレクトリに集めておくことにする場合、まず named が立ち上がるときに読み込むファイル (/etc/named.boot) をこのディレクトリに移し、さらに/etc からもそのファイルがアクセスできるように Symbolic Link を張っておくのがよい。

次に、named.boot の先頭で、それ以降のファイルアクセスのディレクトリを次のように指定しておく。

```
directory               /etc/ns
```

自らが primary に管理する zone データは通常それほど大きくならないので、root ファイルシステムでも差し支えないが、きちんとバックアップが取られるファイルシステムに保存するべきである。一方 secondary zone データは場合によってはかなり (2MB 程度) 大きくなる場合があるので、それを保存するディレクトリを bak とした場合、これは/var のようなファイルシステムにおいた方がよいこともある。secondary として zone データのバックアップファイル名は named.boot の secondary の行の最後のフィールドに指定されるが、そこで

```
bak/jp.zone
```

などとすればよい。この場合、directory として/etc/ns が指定されている場合にはバックアップファイルとして、*/etc/ns/bak/jp.zone* というファイルが使用される。

3.1.2　named.boot ファイル

BIND の挙動を決定するのが named.boot である。このファイルには、前述のように関連ファイルのディレクトリを記す directory 行を初めとしていくつかの制御情報を書くことができ、それによって BIND の挙動が決定される。

forwarders
: あるサーバが問い合わせを処理する場合、サーバで保持しているデータで解決できなかった場合、root サーバに対して問い合わせを行うか、root サーバに対して問い合わせを行う指示を返答する。この場合、root サーバではないが、ある程度の情報を収集しているネームサーバが近くにある場合、それに対して次の問い合わせを行う方が root サーバの負荷やネットワークトラフィックなどの点で望ましい場合がある。

  このようなサーバを指定するのが forwarders である。BIND で保持しているキャッシュデータのみでは質問に対応することができない場合に、通常は root.cache で指定されている root サーバに問い合わせを行うが、forwarders が指定されている場合、まずここで指定されているサーバに対して recursive に問い合わせを行なう。

  forwarders には複数のサーバを指定することができるので、forwarders を指定する場合最寄りのサーバからアクセスが順に行われるように順番を考慮すべきである。そして forwarders に指定されたサーバすべてから回答が得られなかった場合には root.cache のサーバに対して問い合わせが行なわれる。

slave
: slave を指定したサーバでは、問い合わせは forwarders に指定したマシンに対してしか行なわれない。そのため、直接外部のサーバに問い合わせを行なうことがない。通常 forwarders を指定した場合、事故を防ぐために slave も指定した方が安全である。

3.1.3　zone データのシリアル番号

各ネームサーバ間では、各々の zone データがすでに更新されているかどうかの判断を、その SOA レコード中に記述されたシリアル番号を比較することによって行なっている。このシリアル番号は “*234*” のような数字、または “*3.26*” のように小数点を一つ伴う数字のいずれかである必要がある。

WIDE Internet の各参加組織で多く用いられている方法は、RCS と併用し、RCS のリビジョン番号と BIND のシリアル番号とを一致させるという方法である。この場合チェックインを行なう毎にシリアル番号を書き替えることが必要になるが、シリアル番号の変更忘れのため更新されたデータが伝搬しないというトラブルを防ぐことができる。

3.1.4 絶対名と相対名

BIND で設定が必要なファイルで、RR(Resource Record) を書くものに関しては、名前の末尾の “.” が重要な役割を担っている。例えば、

```
wide.ad.jp
```

と

```
wide.ad.jp.
```

は異なった意味を持つ。従ってゾーンデータ作成の際には名前の末尾の “.” の有無に充分注意する必要がある。

後者のように名前の末尾に “.” がある場合はその字面通りの解釈がなされるが、前者のように名前の末尾に “.” がない場合には、current origin からの相対的な名前として解釈され、current origin が “keio.ac.jp” である場合、実際には “wide.ad.jp.keio.ac.jp” という名前であるものと解釈されてしまうことになる。つまり current origin が省略されたものとして取り扱われる。

current origin は$ORIGIN によって変更することができるが、一般には/etc/named.boot でそのゾーンファイルを引用するときのドメイン名

```
primary domainname filename
```

の第 2 フィールドで決定される。

末尾の “.” を忘れやすい場所としては、例えば SOA レコードの管理者の Email アドレスフィールドの末尾などがある。また in-addr.arpa ドメインの PTR レコードの RR データ部には通常 “.” を記述しなければならない。

3.1.5 localhost の定義

ある種のソフトウェアは、“localhost” に対する IP アドレスを求める際、RES_DEFNAMES を定義している。そのために resolver で指定された default のドメイン (/etc/resolv.conf の domain 行の記述に対応している) を付加し、

```
localhost.default-domain
```

に対する検索を行なってしまうことになる。このような場合にも対応するため、全てのドメインについて

```
localhost       IN      CNAME   localhost
```

というエントリを追加するか、あるいはその zone の記述ファイルに、

```
localhost       IN      A       127.1
```

というエントリを入れておくと良い。

3.2 BIND のクラス

3.2.1 プライマリサーバ

プライマリサーバはその zone データのマスタであり、その zone の記述ファイルの実体の管理を行なっている。ここで、ある BIND サーバがある zone“*wide.ad.jp*” のプライマリサーバであり、その zone 情報が “*wide.zone*” というファイルに格納されているとしよう。このとき、named.boot ファイルでは次のように指定する。

```
primary         wide.ad.jp              wide.zone
```

3.2.2 セカンダリサーバ

セカンダリサーバはプライマリサーバで設定された zone データをキャッシュし、その zone データに関しては プライマリサーバと同等の権限をもつように設定されたサーバである。例えば *wide.ad.jp*zone のセカンダリサーバになるためには、named.boot ファイルで以下のように指定する。

```
secondary       wide.ad.jp              133.4.1.1 bak/wide.zone
```

なお、3 番目のフィールドはその zone データをどのホストから転送するかということを示しており、通常はその zone データのプライマリサーバ を指定する。

ところで、セカンダリサーバには、ある zone データの内容を (プライマリサーバの設定に関係なく引用している “unauthorized” セカンダリサーバと、プライマリサーバによってサーバとして承認されている “authorized” セカンダリサーバの 2 種類が存在している。

“authorized” セカンダリサーバ は、その zone データ中にプライマリサーバによってそのサーバが NS レコードによって公式に指定されているものである。従って、その zone に関しては “authorized” セカンダリサーバはプライマリサーバと同様の権限を持つことになる。“authorized” セカンダリサーバはその zone の管理者が戦略的に配置するものであって、root から NS レコードを順にたどることによってその存在を知ることができる。

3.2.3 caching only server

各組織のプライマリサーバ、あるいは “authorized” セカンダリサーバ以外のサーバは、caching only server として設定することになる。そのためには、forwarders によってその組織のプライマリ/セカンダリサーバを指定する。他の組織のサーバを forwarders として指定するのはあまり好ましいことではない。やむを得ない場合でも、forwarders の第 3 位以降に設定するべきであろう。

caching only server には必要に応じて local に問い合わせを解決できるように、ドメイン内部の zone データを保持するように設定しておくことが望ましい。

3.3 日本のネームサーバの系列

ネームサーバは US Internet を含めて世界的に一つの体系で運用されている。このための条件は

IP パケットが相互に届くこと

であり、海外 Internet に IP reachable でない組織の場合にはこれとは異なった設定が必要になる。

日本の広域ネットワークでは、ネームサーバを次のように 3 つのクラスに分類することによって、この状況下で必要な範囲の情報を検索できるように対応している。

系列 A 国内の情報で、国外から reachable な部分のみを保持する。

系列 B 国内で reachable な情報を保持する。

系列 C 国内部分は系列 B の情報を用い、海外部分は適切な海外のサーバへのアクセスを行なう。

系列 A のネームサーバは海外からアクセスできる情報のみを取扱い、海外から日本への問い合わせに答える役割を担っている。このサーバは root サーバに jp. ドメインのサーバとして登録されている。現在このサーバが走っているホストとしては、

jp-gate.wide.ad.jp	133.4.1.1
ns.tisn.ad.jp	133.11.11.12
orion.arc.nasa.gov	128.102.18.10
trantor.umd.edu	128.8.10.14

が挙げられる。下の 2 つは海外に存在している。この系列のサーバは、通常海外のホストからの問合せのみを処理し、国内一般のサーバからアクセスされることはない。

系列 B のネームサーバ は国内のデータのみを管理しており、海外の zone のデータは全く保持していない。現在、系列 B のネームサーバのマスタは

ccut.cc.u-tokyo.ac.jp	192.41.197.4

が担当している。

系列 C は国内の海外にアクセスできるところで運用される。この系列のサーバは、海外の部分は海外のサーバから情報を得、国内部分は系列 B のサーバから得ることによって、IP で接続されるすべてのホスト等に関する情報を検索することができる。

このような複雑な状況がでてくるのは、国内のすべての組織から海外にアクセスすることができないということにある。そのため、ネームサーバを立ち上げる場合には、IP reachability によってどのクラスのサーバを立ち上げるべきであるか、あるいはどのクラスのプライマリサーバにそのドメインの情報を登録してもらうかということが異なってくることに注意しなければならない。

3.3.1　海外から reachable な場合

root.cache　海外に対して IP パケットが届く場合の設定は比較的簡単である。つまり BIND の配布キットに付属している root.cache をそのままインストールすればよい。ただし 現在の root.cache は古い配布キットのそれとは異なっているため、古い配布キットに含まれている `root.cache` を使う場合には内容を変更する必要がある。1992 年 3 月現在の root.cache を表 3.1 に示す。

表 3.1: 海外到達可能なホスト用の root.cache

```
;
; Initial cache data for root domain server.
;
.                       99999999  IN  NS  ns.nasa.gov.
                        99999999  IN  NS  ns.nic.ddn.mil.
                        99999999  IN  NS  aos.brl.mil.
                        99999999  IN  NS  kava.nisc.sri.com.
                        99999999  IN  NS  c.nyser.net.
                        99999999  IN  NS  terp.umd.edu.
                        99999999  IN  NS  nic.nordu.net.
;
; order does not matter.
;
ns.nic.ddn.mil.         99999999  IN  A   192.112.36.4
aos.brl.mil.            99999999  IN  A   192.5.25.82
kava.nisc.sri.com.      99999999  IN  A   192.33.33.24
c.nyser.net.            99999999  IN  A   192.33.4.12
terp.umd.edu.           99999999  IN  A   128.8.10.90
ns.nasa.gov.            99999999  IN  A   128.102.16.10
                        99999999  IN  A   192.52.195.10
nic.nordu.net.          99999999  IN  A   192.36.148.17
```

jpドメイン　各組織では、少なくとも１つjpドメインとin-addr.arpaドメインの “unauthorized” セカンダリサーバを持つべきである。この場合、jp および in-addr.arpa の各 zone データは最寄りの “authorized” セカンダリサーバから取り寄せるように設定する。このようにすることによって、組織を跨る問い合わせを減らすことができる。1992 年 3 月時点での jp の authorized セカンダリサーバ を表 3.2 に示す。また表中の utnet.nc.u-tokyo.ac.jp は “authorized” セカンダリサーバではないが、現在歴史的理由によりこの目的で使用されている。これに関しては近いうちに変更される可能性がある。またマスタサーバとして ccut.cc.u-tokyo.ac.jp が使用されているが、ここからデータを取り寄せるよう設定することは、現在禁止されているので注意が必要である。

この操作を行なわない場合、現在の BIND の設定では root.cache によって jp のデータの検索を行なう際にの root サーバに問い合わせを行ってしまい、結果として海外向けのデータを教えられてしまい、障害が起こることになるので注意が必要である。すなわち結果として系列 A の海外向けサーバに問い合わせが行なわれることになり、国内向けの情報を得られないことになってしまうわけである。

表 3.2: 国内の jp. の各ドメインのサーバ

地区	ホスト名	IP アドレス
マスタ	ccut.cc.u-tokyo.ac.jp	192.41.197.4
東京	utnet.nc.u-tokyo.ac.jp	192.41.197.3
TISN	utsun.s.u-tokyo.ac.jp	133.11.11.11
藤沢	endo.wide.sfc.keio.ac.jp	133.4.11.2
		133.27.48.2
大阪	vanilla-ice.gw.osaka-u.ac.jp	133.1.192.4
福岡	nakasu.csce.kyushu-u.ac.jp	133.5.19.3

3.3.2 海外から reachable でない場合

海外にアクセスできない組織の場合、root.cache に海外のサーバを指定することはできないので、それに代わる設定をする必要がある。1992 年 3 月時点での国内向けのサーバのマスタは表 3.3 に示すとおりであるため、BIND の配布キットに含まれている root.cache をそのまま使用することはできない。

また、トラフィック等を軽減するため、組織で１つは jp ドメインの zone データを cache しているサーバを設定しておくべきである。このサーバは表 3.2 に示すサーバの中で適当なものから Zone データを得るように設定する。これに関しては先程と同様である。

ただし in-addr.arpa ドメインに関しては、表 3.2のサーバは海外の情報も含んでおり、その zone データには海外を指している NS レコードが存在しているので、国内向けサーバとしては適当でないため、セカンダリサーバとして設定してはならない。

またこのサーバでは、海外に対する情報を国内向けのものとして正しく取り扱う必要があるので、表 3.2 のサーバを forwarders として指定してはならないことに注意しなければならない。

また、系列 B のサーバは海外のドメインに関する情報は全く含まれていないので、Berkeley.EDU 等のアドレスへメールを送る際には ネームサーバ の MX レコードを使用することはできない。そのため sendmail.cf などで直接海外向けゲートウェイを指定する必要がある。

表 3.3: 海外到達不能なホスト用の root.cache

```
.                       99999999  IN  NS  ccut.cc.u-tokyo.ac.jp.
ccut.cc.u-tokyo.ac.jp. 99999999  IN  A   192.41.197.4
```

3.4　MX レコード

MX レコードは電子メールの交換を行なう場合に、どのマシンに送信すればよいかという示唆を与えるために運用されている BIND のリソースの 1 つである。

some.domain.	IN	MX	*20*	*ms.some.domain.*
ms.some.domain.	IN	A		*ppp.qqq.rrr.sss*

という記述があった場合、

user@*some.domain*

宛にメールを出す場合、“*some.domain.*” という名前を持つリソースで type が MX であるものを検索する。そして、そのデータ部に指定されている *ms.some.domain.* というマシンがそのアドレスのメールを処理できることが分かるので、そのマシンに対して SMTP [?] でメールを送信することができる。

この時、1 つのアドレスに対して複数の MX が指定されていた場合、最も優先順位の高い (優先順位の数字の小さい) ホストから順にメールの送信が試行される。

3.4.1　MX ホスト

MX レコードで指定されているホストは、外部に対して IP reachable でなければならない。少なくとも国内での IP reachability は確保されていなければならないし、系列 A のサーバに登録されているドメインとそのサブドメインに関する MX は海外も含めて IPreachable でなければならない。

組織として一部のホストしか IP reachability が確保されていない場合、それらのホストにも MX レコードを定義しておくことが大切である。この場合、組織内部向けと外部向けと異なった系列のサーバを用意することも可能であるが、煩雑になる問題点がある。組織内部のメールの送信に MX レコードを使わない場合は単に IP reachable なマシンのみに MX レコードを設定しておけば済むが、組織内部でも MX レコードを使用している場合には工夫が必要になる。

このような場合、最終的な MX ホストに対する優先順位を上げて (数字を小さくして) おき、IP reachable なホストの MX の優先順位を下げて (数字を大きくして) おくという方法がある。外部の計算機はメールを送る際、数字のもっとも小さな MX ホストに送信を試みるが、そのホストは外部から IP reachable でないため、SMTP のコネクションを

張ることができず、timeout した後、IP reachable な次善の MX ホストにメールを届けることになる。

この方法は外部からメールを送る場合、必ず一度 timeout を起こす点を除いては正常に稼働する。この timeout を防ぐには、IP reachable でない MX ホストに SMTP コネクションを張る段階で、適当なゲートウェイが例えば ICMP Host Unreach を返答することにより、即座に IP reachable な次善の MX ホストへメールを送信することができる。メールの送信に必要なパケット数は若干増加するが、timeout を起こさないのでよりエレガントであると言える。

3.4.2 wildcard MX

メールを送るとき、メールアドレスのドメイン部と MX レコードの名前が正確に一致した場合のみ MX レコードとして採用される。例えば、

user@*host.domain*

にメールを出す場合、

domain IN MX *20* *ms.domain*

という MX レコードは使用されない。*host.domain* と *domain* が一致していないからである。従って、全てのホストに対して MX を宣言しておく必要があるが、これは結構大変である。

このような場合、MX の wildcard(“*”) を使用して MX レコードを宣言する方法がある。この場合では、

*.*domain* IN MX *25* *ms.domain*

のように宣言しておけばよい。

また、zone が定義するドメイン自身に対する MX レコード、具体的には “@” に対する MX レコードも忘れないように注意すべきである。一般的には、次のような記述 (あるいはそれと同等なもの) が各 zone データにまれるようにする必要がある。

@ IN MX *30* *ms.domain*
* IN MX *30* *ms.domain*

ここで *ms.domain* はそのドメインのメール配送を行う、いわゆるドメインマスタである。

3.4.3 A レコードと MX レコード

ところで、あるホストのアドレスが A レコードで宣言されていた場合、そのレコードに対して陽に MX が宣言されていない限り sendmail はそのホストに直接メールを送ろうとする。この場合、例え wildcard(*) で MX レコードが定義されていてもそれは参照されない。

従ってメールを送ってほしくないホストのエントリに対しては、適当な MX を一つ一つ指定する必要がある。

3.4.4　安定に IP reachable でない場合

64Kbps 以上の速度で外部組織と接続されている場合にはあまり問題ではないが、一部の 3.4KHz 帯音声回線や X.25 網など IP 接続の安定性が悪い場合がある。このような場合、その組織の近隣で安定して IP reachable なホストに対してより低い (数字が大きい) 優先順位で MX レコードを宣言しておいた方が望ましい。

この場合、必ずその MX ホストの管理者に事前に承認を得ておくようにすることは言うまでもない。

3.5　セカンダリサーバの setup

このようにドメイン名からアドレスへの変換や電子メールの送信にネームサーバ は重要な役割を果たしている。従って zone データにアクセスできなくなるような事態は極力避けなければならない。

例えばリンクやゲートウェイがダウンしても、zone データが外部からアクセスできる場合、“Host Name Lookup Failure” によるメール配送処理の遅延を防ぐことができる。場合によっては次善の MX ホストから UUCP 等でメールを転送することができるし、その MX ホストの mqueue を調べれば何通のメールが pending になっているかを知ることもできる。

zone データを常時アクセスできるようにしておく方法は、“authorized” セカンダリサーバを組織外部に依頼することである。それもなるべくネットワークの単一故障に対して影響を受けないようにある程度離れた組織に依頼することが望ましい。

現在 WIDE Internet では、WNOC-SFC に接続されているドメインに関しては東京方面に、WNOC-TYO に接続されているドメインに関しては藤沢や京阪神方面に “authorized” セカンダリサーバを設定するようにしている。

3.6　ネームサーバの運用

ネームサーバ が安定に動作していることは安定なネットワークサービスを提供する上で重要である。適切に “authorized” セカンダリサーバを設定することは、ホスト等の障害を考える上で無論重要であるが、ネームサーバ単体としてもきちんと動作させておくことも重要である。

ネームサーバで、特に海外部分を含めた in-addr.arpa のセカンダリサーバとして BIND を設定した場合、named プロセスはおよそ 3MB 近くの仮想空間を消費することになることにも注意が必要である。

また定期的に (例えば 1 時間に 1 回程度)cron 等で named が実際に動作しているか (何らかの原因でダウンしていないか) ということを調べ、必要なら自動的に再起動するようにすることが望ましい。

3.7 named の起動

named は経路情報が確立した後に起動するべきである。しかしながら、ホスト名からアドレスへの変換を伴う他のプロセスよりは前に起動しておくべきである。特に named を頻繁に利用する sendmail が named より先に立ち上がらないように注意しなければならない。

gated.conf などの経路情報に関するプログラムの設定ファイルはホスト名からアドレスへの変換を伴わないように、シンボリックな形式ではなく IP アドレスを直接指定するのが無難である。やむを得ない場合には、gated などより先に named を起動する必要があるが、その場合には必要な経路情報を静的に設定しておき、named が正しく情報の授受をできるようにしておくことが大切である。

3.8 resolver

3.8.1 resolver の設定

ユーザの要求に応じてネームサーバに対して問い合わせを行なうのが resolver の役割であり、通常 libresolv.a というライブラリとして提供され、プログラムのリンク時に同時にリンクされる。SunOS 4.x では dynamic linking がサポートされているので、その機能を活用し shared library を resolver 対応のものと差し替えることで resolver の機能を利用することも可能である。

resolver の挙動は/etc/resolv.conf というファイルによって決定される。このファイルには、resolver が問い合わせるネームサーバや default ドメイン名等が指定されている。このファイルの内容は例えば次のようになっている。

```
domain      wide.ad.jp
nameserver  133.27.48.2
nameserver  131.4.11.2
```

domain 行ではこのホストが発生する ネームサーバ の問い合わせが相対的なもの (末尾に “.” を含まない) であった場合、この行で指定された default ドメイン名を付加して問合せを行なう際に使用されるドメイン名を表している。nameserver 行は resolver が問合せを行なうネームサーバが存在するホストの IP アドレスを順に指定する。この nameserver 行は最大 MAXNS 個 (現在は 3) 指定することができ、指定された順番に問合せを試みる。最初のサーバからの応答がない場合には次に指定されたサーバ、さらにその次というように問合せを行なう。従って、ネームサーバが動いているホストのうち、問合せを行なうホストに近い順に指定するのがよい。

domain 行のドメイン名の後に不必要な空白文字を残さない方がよい。あるバージョンの resolver は、この不要な空白文字をドメイン名の一部であるように取り扱ってしまい、

正しい検索ができなくなる[1]ことが知られている。

3.8.2　NIS との共存

NIS を運用している場合、ホスト名と IP アドレスの変換は NIS の hosts.byname、あるいは hosts.byaddr の各マップによって行なわれる。従って、全てのホストのアドレス情報をこのマップに含めなければならないが、ネームサーバへ問合せを行なうことによってこのことを実現する方法がある。

条件として、NIS の全てのサーバでネームサーバへの問合せをサポートしていなければならない。現在知られているネームサーバをサポートしている `ypserv` を含む OS としては、SunOS 4.x[2]、NEWS-OS 4.x などがある。

ことも必要である。

ネームサーバ への問合せを行なうようにするためには、まずすべての ypserv が動作しているマシンで正しく /etc/resolv.conf が設定されており、組織外部に IP reachable であることを確認する。ついで NIS のマスタサーバマシンの NIS の Makefile(/var/yp/Makefile) を次のように修正する。SunOS 4.1.1 以降ではこの設定はコメントアウトされた形で既に含まれている。

> `hosts.time` を make するところの `$(MAKEDBM)` のオプションに “`-b`” を追加する。`hosts.byname` に 1 箇所、`hosts.byaddr` に 1 箇所修正箇所がある。

SunOS の場合は NIS マップを検索し、発見できなかった場合にはネームサーバ にアクセスするようになっているが、NEWS-OS の場合は resolv.conf の設定によってこの順序を変更することができる。詳しくは NEWS-OS に付属のマニュアルを参照されたい。

3.9　ネームサーバ の問題点

3.9.1　PTR に関するセキュリティホール

ネームサーバ に関するセキュリティホールとして、in-addr.arpa レコードの PTR レコードの問題が知られている。IP アドレスからホスト名を求める際、ネームサーバでは PTR レコードの検索を行なうが、この検索結果のドメイン名をそのまま信用することは危険である。例を示すと次のようになる。

いま仮に *133.4.1.1* の正しいドメイン名は *jp-gate.wide.ad.jp* であるとしよう。これに対応したレコードは

> *1.1.4.133*`.in-addr.arpa.  IN  PTR  `*jp-gate.wide.ad.jp*

[1] この情報は東京理科大学の右手氏による。

[2] SunOS 3.5 に添付されている `ypserv` はそのままでは動作しない。また SunOS 4.x で 4.0.3 以前のものは 4.0.3 に更新が必要である。

であるが、これを

1.1.4.133.in-addr.arpa. IN PTR *fake.fooled.edu*

と書き換えた場合、ネームサーバを経由したアドレスードメイン変換の結果は、*fake.fooled.edu* に (*fooled.edu* の許可なく) なってしまう。~/.rhosts 等のファイルにこれが含まれている場合などは、第 3 者に無断で login されてしまうことになる。

これを防ぐため、ネームサーバ経由で得られたドメイン名を用いて、再度そのアドレスを検索し、最初のアドレスが含まれていた場合に限りそのドメイン名を信用するという方法が提案されている。

これ方法では、個々のネームサーバを用いるアプリケーション毎にそれぞれ対応しなければならない。また NIS を経由して検索を行なっている場合、ypserv がこの機構をサポートしていることが要求される。SunOS 4.1.1 の ypserv では対応しているようだが、不完全なようである。従って個々のアプリケーション毎に、きちんと対応を取った resolver をリンクし直す必要がある。

3.9.2 シリアル番号の問題

BIND の各 zone では、そのシリアル番号を SOA レコードに記述し、zone データが更新されたことをセカンダリサーバなどが知ることができる。管理者は zone データを更新した場合、シリアル番号の更新を忘れないようにしなければならない。

ところで、このシリアル番号には小数点を一つ含めることができるが、その比較は次のようなアルゴリズムで行なわれる。

シリアル番号の一番左の桁から順に、

- 一桁数字を読むたびに値を 10 倍したものにその数字を加える。
- 小数点が現れたら整数部を 1000 倍する。

従って、1 は 1 であるが、1.0 は 10000 になる。また 1.9 は 10009 であるが 1.10 は 100010 になる。

このことは、通常はあまり問題ではないが、小数点ありのシリアル番号の体系を使っている場合、整数部 (メジャー番号として用いられることが多い) を変えた場合に問題になる。例えば 1.23 というシリアル番号の zone データを更新し、2.0 というシリアル番号にしたとしよう。この場合の BIND 内部での比較は、前者が 100023 であるのに対して後者は 20002 となってしまい、100023 $>$ 20002 であるため zone データが更新されたことを伝搬することができない。この場合、例えば 2.00 などのように小数部分の桁数を減らすことがないようにすればこのようなことを防ぐことができる。

また、整数部を更新した日付にすることもよく行なわれるが、この場合小数部分があるとその値が Named の内部で unsigned int の範囲で表現できない。従って、この場合には小数部分を使用しないようにしなければならない。

結局次のような方法がお勧めであろう。

- 小数部分を初めから（例えば）3 桁取って、1.034、2.001 のようにして用いる。
- 一日に十回もデータを更新することはないと仮定して、9104011 のように日付に、その日でのシリアル番号を一桁付け加える。一日に十回以上データ更新の可能性がある場合には、91040101 のように二桁付け加えればよい。

但し、中途で下の方法から上の方法に変更するようなことは避けるべきである。

3.9.3　RFC1101 の RR

RFC1101 によると、ネットワーク名や subnet mask などの表示を PTR を用いて積極的に行なうことが提案されている。この提案によると、IP ネットワークアドレスとドメイン名の対応を DNS を通じてとることができ、ネットワーク管理上有益であるため、それぞれの組織のトップレベルで次のような RR を導入することをお推めしたい。

IP アドレス −→ ネットワーク名
ある組織 *someorg.co*.jp がクラス *B* の IP ネットワークアドレス *YYY.XXX.0*.0 を割り当てられている場合、*XXX.YYY*.in-addr.arpa のゾーンに次のような RR を追加する。ただし、*YYY.XXX.0*.0 の申請の際に用いたネットワーク名が *someorg-net* であり、そのネットワークの subnet mask が 255.*255.255.0* であるものとする：

```
0.0.XXX.YYY.in-addr.arpa. IN PTR someorg-net.someorg.co.jp.
                          IN A   255.255.255.0
```

ネットワーク名の表現
ネットワーク名は PTR を用いてその IP ネットワークアドレスを表現する：

```
someorg-net.someorg.co.jp.  IN PTR 0.0.XXX.YYY.in-addr.arpa.
```

組織名と IP アドレス
組織が使用している IP アドレスへの PTR レコードを記述する。複数の IP アドレスを使用している場合には、複数記す。

```
someorg.co.jp.  IN PTR 0.0.XXX.YYY.in-addr.arpa.
```