(Memoirs of the Faculty of Education and Human Studies
Akita University (Natural Science)
59, 25－33 (2004))

BIND9 のビュー機能利用を支援するゾーンファイル自動生成ツール

佐々木重雄*

A Zone File Generation Tool supporting the View Feature introduced in BIND 9

Shigeo SASAKI

abstract

This paper presents a zone file generation tool that creates forward and reverse zone files from master ones. It can maintain multiple domains and multiple *views*, while most of simular tools can do only single domain.

The *view* feature was introduced in BIND version 9 and is useful for firewall'ed sites. Such a site has multiple access lines to the (global) Internet, to its intranet, to its DMZ net and so on, and its DNS server has to reply different DNS zone information depending on where the query comes. The *view* feature supports to let a DNS server answer differently depending on the IP addresses of the query clients.

This paper also points out the common mistakes in zone file descriptions. A part of them are introduced by incorrect DNS management programs which are, in many case, old and no longer maintained. Correct zone file generation tools can reduce the number of them. The information on the mistakes is helpful for not only the users of the tool but also DNS administrators who write zone files manually.

1 動機

1.1 BIND とゾーンファイル

BIND(Berkley Internet Name Domain) は、最も広く使用されている DNS サーバの実装である。DNS サーバの管理は、しばしば難解と評価される。これは BIND の設定ファイルや、資源レコードの記述ファイルの、必要以上の複雑さによるところが大きいといえる。BIND の資源レコード記述ファイル (本稿では、以下、習慣に従い、ゾーンファイルと呼ぶ) は RFC 1034, 1035[4, 5] で定義される、DNS サーバ間でやり取りするデータ (ゾーン情報) と一対一に対応するもので、きめ細かい設定は可能であるが、データの維持管理には余計な神経を使う。たとえば、正引き情報と、単にそれを転置しただけのはずの逆引き情報を別々に管理しなければならない。データを更新したなら、シリアル番号を増加させなければならないなど、BIND では、ゾーンファイルの管理は、人手で行なうのではなく、自動生成して運用することを前提としていると考えられる。さらに、汎用ドメインを運用する場合、設定ファイルには、日本語などの非ラテン文字を記載することができず、エンコードしてからロードする必要があるなど、様々な点から、自動生成は必須であるといえる。

1.2 外向き DNS 情報と内向き DNS 情報の分離

BIND 9 で導入されたビュー機能は、外向き DNS 情報と内向き DNS 情報を一台のサーバで管理するのを容易にする、極めて有用な機能である。

BIND 9 のビュー機能は、問合わせクライアントの位置 (IP アドレス) によって、異なるゾーンファイルを参照して、その情報を返答することを可能とする機能である。これを利用することにより、イントラネット内部用の DNS 情報と、外向きの DNS 情報を区別したいとき、内部用サーバと外部用サーバの二つを分離して立てることなく、一台のサーバ、一つのデーモンでサービスを提供することが可能となる。これは、管理コストの低減に大きく寄与する。

しかし、BIND 9 のビュー機能を利用する場合、外向き、内向き、別々にゾーンファイルを用意する必要がある。正引きファイル、逆引きファイルがもともと複数あるであろうから、外向き、内向きの DNS 情報

*秋田大学教育文化学部

を分離しようとすると、もとの二倍のファイルを維持管理しなければならないことになる。

ところが、DNS のゾーンファイル生成ツールは、世の中に数多くあるにもかかわらず、外向き DNS 情報と内向き DNS 情報を分離して提供することを考慮したものは皆無と言ってよい。実際の運用を考えると、外向きと内向きの情報を分離するといっても、相違する情報はごく一部で、ほとんどが共通であると考えられる。この、共通する部分について、正引き情報 (A レコード)、および、逆引き情報 (PTR レコード) の自動生成くらいはしたいというのが、今回の自動生成ツールの作成の動機である。

ゾーンファイルの自動生成ツールは、少なくない数のものが開発されている。しかしながら、外向き DNS 情報と内向き DNS 情報を区別して提供する用途にかなうものは、皆無である。BIND 9 のビュー機能対応を謳ったものもあるが、ビューごとにマスターファイルを別に用意する必要があるなど、データの冗長性を排除しきれていない。

そもそも、ビュー機能への対応以前に、DNS の新しい規格に合致しないゾーンファイルを生成するソフトウェアが少なくない。本稿では、既存の、ゾーンファイル生成ツールについて、修正するべき事項についても指摘する。

2 BIND 9 のビュー機能

BIND 9 で導入されたビューは、問合わせ元によって、返答する名前情報を切替える機能である。この機能を使うことにより、イントラネットの内側向けの DNS 情報と、外向けの DNS 情報を分けることができる。BIND の設定ファイル (named.conf) でビュー機能を有効にする例を図 1 に示す。図 1 の設定例からわかるとおり、正引きファイル、逆引きファイルを、それぞれ、外向き用、内向き用に別々に用意しなければならない。

3 ゾーンファイル記述の誤りについて

ゾーンファイルの記述に関して、いくつかの誤った設定が、多くのサイトでなされている。自動生成ツールでは、特にこの種類の誤りの発生を避けなければな

```
view "internal" {
    match-clients {
        127.0.0.0/8;
        192.168.1.0/24;
    };
    recursion yes;

    zone "." {
        type hint;
        file "named.root";
    };

    zone "0.0.127.IN-ADDR.ARPA" {
        type master;
        file "localhost.rev";
    };

    zone "example.ac.jp" {
        type master;
        file "internal.zone";
    };

    zone "1.168.192.IN-ADDR.ARPA" {
        type master;
        file "internal.rev";
    };
};

view "external" {
    match-clients { any; };
    recursion no;

    zone "example.ac.jp" {
        type master;
        file "external.zone";
    };

    zone "1.168.192.IN-ADDR.ARPA" {
        type master;
        file "external.rev";
    };
};
```

図 1: ビューに関する named.conf の記述

```
@ IN SOA ( server.example.ac.jp.
           root.server.example.ac.jp.
           1998040100 ; Serial
           10800      ; 3 hours
           3600       ; 1 hour
           604800     ; 1 week
           86400 )    ; 1 day
```

図 2: よくある SOA レコードの誤記述例

```
$TTL 1d
@ IN SOA ( server.example.ac.jp.
           root.server.example.ac.jp.
           2004033100 ; Serial
           3h
           1h
           7d
           20m )
```

図 3: RFC 2308 に準拠した SOA レコード記述例 (BIND 8, BIND 9 用)

らない。

3.1 ネガティブキャッシュ TTL

RFC 2308[8] により、 SOA レコードの minimum TTL の意味が、そのゾーンのデフォルト TTL から、ネガティブキャッシュ TTL(名前がないことのキャッシュ期間) に変更された。しかし多くのサーバで、RFC 1035 や有力な教科書 [2] の例に挙っていた 1 日 (86400 秒) に設定されている。正しくは数十分にするべきとされる (RFC 2308 の例では 1200 秒となっている)。図 2、3 参照。

調査した範囲では、既存の自動生成ツールには、minimum TTL に 86400 秒を与えているものが少なくない。プログラムの修正は簡単なので、利用者はただちに変更するべきであろう。

なおゾーンのデフォルト TTL は、BIND 8, BIND 9 では、 SOA レコードではなく、 $TTL 指令で指定する。なお BIND 9 では $TTL 指令は必須である。既存の自動生成ツールには、 $TTL 指令を理解していないものがある。このようなツールは BIND 9 との併用に堪えないといえよう。

3.2 Lame Delegation

上位ドメインのサーバから指定されているネームサーバ (正確には上位ゾーンから NS レコードで委譲されているサーバ) が DNS サーバとして正しく稼働していないとき、この不正な (下位ゾーンへの) 委譲を Lame delegation という。原因としては、下位ゾーンの DNS サーバが正しく稼働していない事例もあるが、上位ゾーン下位ゾーン間での NS レコードの不一致や、下位ゾーンでの NS レコード CNAME レコードの不適切な使用も少なくない。

このように、 NS レコードは DNS の運用にとってデリケートな情報であるため、自動生成ツールを設計するにあたっては、 NS レコードの誤記述を起こさないよう、配慮が必要である。ソフトウェア設計の方針にもよるが、 DNS 管理者が明示的に NS レコードを指定したとき、ツールは、DNS 管理者の指定を尊重して、何もしないくらいが適切といえるだろう。

3.3 CNAME

ゾーンファイルの記述において、 CNAME レコードは濫用されがちである。これは、プログラマのセンスとして、物理レベルに近い名前よりも、抽象度の高い名前を多用するべきという意識があり (ここまでは正しい)、そのために CNAME レコードを使用しようと考える (これは正しくない) ことによる。しかし、一般に CNAME の使用はなるべく避けた方がよい。たとえば CNAME のチェーンの段数が多くなると、クライアントの実装によっては IP アドレスが引けなくなる危険がある。また NS レコードや MX レコードは CNAME を指してはいけないことになっている [7]。並列する複数のサーバに同じホスト名を与えて負荷バランスを図るラウンドロビン DNS は、複数の A レコードを記載することによって実現するのであって、CNAME によるのでない。

CNAME レコードは、更新時の変更箇所の減少を期待して使用される傾向があるが、ゾーンファイルの自動生成を前提とするなら、この目的に CNAME を使用する必要は無い。 CNAME レコードの弊害を考え

ると、自動生成ツールは、不必要な CNAME レコードの生成を控えるよう努めるべきといえる。

4 自動生成ツールの設計と実装

教育文化学部の DNS サーバ、および、同情報科学研究室の DNS サーバにおいて、ゾーン情報の管理のために使用している、ゾーンファイルの自動生成ツールの設計と実装について述べる。

4.1 設計方針

当サイトにおいて、 DNS のデータを更新する管理者は計算機の専門家とは限らない。そのため、 DNS データの更新に関しては、簡潔な処理で済むことが基本原則となる。次のような方針でツールを作成した。

- データを更新するに当たって実行するコマンドは、エディタおよび一種類のコマンド (make コマンド) のみとする。 make コマンドの実行により、ゾーンファイルの更新を行なうだけでなく、DNS サーバに、そのファイルを再読み込み (reload) させる。
- データ間の依存関係に従い、マスターファイルの更新 (もしマスターファイルが複数あれば、そのうち一つにでも更新) があれば、確実にゾーンファイルが更新されるようにする。
- 冗長性を持たないマスターファイルからゾーンファイルを自動生成する。データベース設計の原則として知られているとおり、データの冗長性は、データ更新時に矛盾が入り込む原因となる。
- ドメイン (正引きゾーン) ごとにマスターファイルを持つ。これは、マシン管理はドメインごとに行なわれると考えるからである。
- 自動生成を行なうのは、原則として A レコードおよび PTR レコードのみとする。 SOA, NS, MX, CNAME レコードは、管理者が手動で記述するべき情報と考え、マスターファイルの更新ごとに自動生成することはしないこととした。ただし管理者の負担低減のため、初期値としてのテンプレートの自動生成は行なう。
- 設定ファイル (named.conf) は自動生成しない。
- 複数ドメインを同時に管理できる。一つのサブネット (逆引きゾーン) に、複数のドメイン (正引きゾーン) のマシンが存在するようなサイトであっても、適切な逆引きファイルを生成する。
- マスターファイル、ゾーンファイル、ゾーン (ドメインおよび逆引きドメイン) 間の関係を Makefile 生成ファイルに記述し、そこから Makefile を (半) 自動生成する。

以下、若干の補足をする。

このようなサーバ管理用のツールの作成にあたっては、どこまでの作業をツールの担当範囲として設計するかの判断が、使い勝手の善し悪しに影響する。一般に、過剰な仕事をしてしまうツールは、「よけいなおせっかい」をするとして、敬遠されがちである。

(過去に自作したものを含む) 既存のツールの使用や、DNS サーバの管理経験から判断して、SOA レコード、NS レコード、 CNAME レコード、MX レコードの自動生成は、一般に、労力の軽減につながらないばかりでなく、ときとして「よけいなおせっかい」すなわち DNS 管理者に余計な手間をとらせることがある。これらの資源レコードは、自動生成する価値が低いといえる。

しかしながら、DNS サーバ運用開始時において、(特に、経験の少ない DNS 管理者にとって)、これらの資源レコードの記述の見本があることは心強い。そのため、本ツールでは、初回に一度だけ、管理するゾーンに適当と考えられるゾーンファイルのテンプレートを生成することにした。

同様に BIND 用の設定ファイルの生成も、「よけいなおせっかい」と考えて、行なわないことにした。既存のいくつかの DNS 用ツールは、BIND 用の設定ファイルを生成する。特に BIND 8 以降、設定ファイルが複雑になったため、この機能は、一見、便利であるような印象を受ける。しかし BIND を適切に運用しようとすると、たとえ設定ファイルが自動生成されようと、BIND の機能をよく理解した上で、細かい設定を記述しなければならないので、自動生成したところで、結局、二度手間となってしまう。

なお、設定ファイルを手軽に作りたい用途には、BIND 8 の配布ファイルに、 BIND 4 用設定ファイ

ルから BIND 8 用設定ファイルへの変換ツールが梱包されており、それを利用することができる。

4.2 ファイル生成の依存関係の自動抽出

本ツールの特徴として、複数ドメインの管理が可能で、一つのサブネット内に複数のドメイン (正引きゾーン) のマシンが混在するネットワーク構成であっても、適切な逆引きファイルを生成できるというものがある。マスターファイルはドメインごとに用意されるため、マスターファイルと、最終生成物であるゾーンファイルとは、多対多の関係になる。

Unix オペレーティングシステムには、古くから、make というツールが提供されていて、これを使用することで、ファイル間のデータ依存関係に従って、ファイル生成プログラムを実行する手続きを、自動化することができる。ゾーンファイルの生成も、make のパラダイムに則っており、本ツールでも、make コマンドによってゾーンファイルを自動的に生成する。

しかしながら make によって、ファイルの生成処理は自動化されるが、make コマンドの処理手順書である Makefile を、どのように作成・保守するかが問題として残る。Makefile の作成・保守には、一定規模以上のソフトウェアシステムを C 言語で作成した経験のあるプログラマ程度のスキルが要求される (正確には、そのようなプログラマ以外 Makefile を記述する経験が稀である)。これでは、DNS のゾーンファイルを直接メンテナンスするよりも、ゾーンファイルの自動生成ツールをメンテナンスする方が難しいということになってしまう。

管理するゾーンが追加になった場合、割当てられた IP アドレスブロックが変更になった場合 (これは DNS 的には、逆引きゾーンが変更になることを意味する)、また BIND 9 で複数のビューを扱うことにした (あるいは扱うビューの数を増やした) 場合、Makefile の変更が必要になる。大規模なサイトでは、これが負担になることは、容易に予想できる。また経験上も、ゾーンファイル生成のための Makefile を正しく、つまり、ファイルの依存関係を見落とさずに記述することは、それほど容易なことではない。

以上を考慮し、本ツールでは、Makefile を自動生成するプログラムを用意し、管理するゾーンが追加、変更されたり、ビューが追加されたときには、Makefile を生成し直して利用するものとした。

4.3 ファイル

本ツールが扱うファイルは、表 1 に示すとおり。

マスターファイル この形式は /etc/hosts と同じものである。すなわち、行形式のテキストファイルで、各行には、IP アドレスおよびホスト名 (複数でも可) を記載する。空行およびシャープ記号 (#) から行末は無視される。本ツールの仕様として、ホスト名が FQDN(Full Qualified Domain Name) である、すなわちドメイン名を含む完全形式である場合、正引き情報は他のドメインで登録されていると解釈し、逆引き情報、すなわち、PTR レコードのみ生成する。

インクルードファイル 拡張子 `.incl` を持つファイルは、ゾーンファイルにインクルードされるファイルである。このファイルは、通常、マスターファイルから、後述する `hosts2a` コマンド、`hosts2ptr` コマンドによって生成される。

テンプレートファイル `.tmpl` の拡張子を持つファイルは、ゾーンファイルのテンプレートである。ファイル形式はゾーンファイルと同じである。ただしシリアル番号の部分だけ `@serialno@` と記述する。本ツールを使用する場合、SOA, NS, MX レコードは、このテンプレートに記載する。

ゾーンファイル BIND によって読み込まれる、最終形式のファイルである。

4.4 実装

本ツールは、複数のシェルスクリプトを make から呼出して利用する構成になっている。本ツールを構成するコマンド群を表 2 に示す。

hosts2a コマンド /etc/hosts 形式のマスターファイルから正引きインクルードファイルを生成する。

hosts2ptr コマンド /etc/hosts 形式のマスターファイルから逆引きインクルードファイルを生成する。

表 1: 生成ツールが扱うファイル

拡張子	ファイル
.hosts 等	/etc/hosts 形式のマスターファイル
.zone 等	ゾーンファイル
.tmpl	ゾーンファイルのテンプレート (初回のみ生成)
.incl	ゾーンファイルからインクルードされる、マスターから自動生成されたファイル
.count	シリアル番号

表 2: 生成ツールを構成するコマンド

コマンド	役割
hosts2a	A レコードの生成
hosts2ptr	PTR レコードの生成
make.zone	ゾーンファイルの生成
make.tmpl	テンプレートの生成
make.depend	依存関係の分析
configure	Makefile の生成

make.zone コマンド　テンプレートファイルおよびシリアル番号ファイルから、ゾーンファイルを生成する。

make.tmpl コマンド　ゾーンごとのテンプレートファイルを作成する。

make.depend コマンド　$INCLUDE 指令の連鎖を追い、ファイルの依存関係を分析する。

configure コマンド　本ツールが正しく動作するためには、データ間の依存関係を見落とさずに、マスターファイルの更新をゾーンファイルに反映させなければならない。このため Makefile の正確さが重要である。現在の実装では、生成すべきゾーンファイル、および、中間ファイルの生成規則を、別途、ファイル (図4、5) で明示的に与えることにより、Makefile を半自動的に生成するようにしている。コマンド名は、GNU autoconf にならって configure とした。

```
zonefile external.zone example.ac.jp
zonefile internal.zone example.ac.jp
zonefile global.rev    0.0.10.IN-ADDR.ARPA
zonefile intranet.rev  1.168.192.IN-ADDR.ARPA

hostsfile /etc/hosts example.ac.jp
```

図 4: ファイル-ドメイン対応定義

```
hosts2a /etc/hosts > external.incl
hosts2a /etc/hosts > internal.incl
hosts2ptr 10.0.0    /etc/hosts > global.incl
hosts2ptr 192.168.1 /etc/hosts > intranet.incl
```

図 5: 中間ファイル生成規則

5　使用法

本ツールを利用して DNS サーバの管理を行なう場合の作業手順を述べる。

5.1　準備

DNS サーバ (BIND 8 または BIND 9; ビュー機能を利用するならば、BIND 9 が必要)、および、その設定ファイル (named.conf) を用意していることが前提である。BIND 9 であれば、コマンド制御用の鍵を準備しておく必要がある。

named.conf で指定したゾーンファイルを置くディレクトリに、本ツールを展開する。ユーティリティ・プログラム群が bin ディレクトリに置かれる。このユーティリティを利用可能にするために、bin ディレクトリで configure および make を実行する。configure 時に、ゾーンファイルの再読み込みのためのコマンドを指定しなければならない。詳細は README を参照。

5.2　メタ情報の記述

新しく運用を開始するとき、管理するドメインが追加、あるいは、変更になったとき、以下の作業を行なう。

bin ディレクトリの作業が終了したら、ゾーンファイルの置かれるディレクトリに移動する。named.conf に合わせて、Makefile.def というファイルにゾーンファイルとゾーンの対応を記述する (図4)。さらにマスターファイルの指定も行なう。複数のドメインを管理する

場合は、マスターファイルも複数になる。

次に中間ファイルの生成規則を translate.sh というファイルに記述する (図 5)。正引きの中間ファイル生成は

```
hosts2a master >include
```

逆引きの中間ファイル生成は

```
hosts2ptr address master >include
```

と記述する。ここで *master* はマスターファイルの名前、*address* は逆引きゾーンのネットワークアドレス (IP アドレスの最初の 3 オクテット)、または、 in-addr.arpa ドメインのゾーン名、*include* は生成するインクルードファイルの名前である。インクルードファイルの名前は、通常、ゾーンファイルの拡張子 (.zone や .rev) を .incl にしたものにする。これは、後述するテンプレートファイルに、この名前のファイルをインクルードするという記述が含まれるからである。インクルードの有向グラフが把握できていれば、ファイル名を他のものにして構わない。

Makefile の生成　以上の用意ができたなら、configure コマンドにより Makefile が自動生成される。Makefile だけでは、ファイルの依存関係の情報が不完全なので、"make depend" を実行する。

5.3 日常の運用

マスターファイルを編集し make コマンドを実行すれば、ファイルの依存関係に従い、ゾーンファイルが生成され、DNS サーバによって読み込み直される。通常は、これだけの作業でよい。

ゾーンファイルの再読み込み (reload) の方法は、BIND のバージョンによっても、そのサイトの運用方針によっても異なる。管理者権限を必要とする方法をとる場合、 make コマンドの実行も管理者権限が必要である。もし再読み込みの方法を変更する場合は、 bin ディレクトリで、再度、 configure および make をやり直す。

6 他のソフトウェアとの比較

6.1 h2n

DNS 管理の代表的な教科書である "DNS and BIND"[1] でも紹介されている、 /etc/hosts 形式からゾーンファイル形式への変換ツールが h2n である。/etc/hosts 形式のファイルをマスターファイルとする点で、本ツール (に含まれる hosts2a および hosts2ptr コマンド) は h2n と類似している。重要な相違点は、h2n が単独のコマンドで、すべてのゾーンファイルを生成するのに対し、本ツールは、複数のユーティリティ・コマンドの組合わせで動作することである。そのため h2n の方が、利用法の習得が容易である。一方、本ツールの方が、管理上の柔軟性が高い。

また一般に、ゾーン情報の一部には、自動生成では対応しきれないものも存在するが、そのような情報の扱いが、 h2n と本ツールでは異なる。h2n では、接頭辞として spcl. の付いた名前のファイルが存在すれば、そのファイルをインクルードする指令を、ゾーンファイルに埋め込むようになっており、自動生成しない情報は spcl. ファイルに記述する。

h2n 利用上の注意点　現在配布されている h2n のコードは、SOA レコードの minimum TTL が RFC 1035[5] で例示されている 1 日のままであること (RFC 2308[8] で意味が変更になっており、数十分が推奨値)、デフォルトではホストごとに MX レコードを生成することが、実運用上、問題となる。また、サーバの設定によるが、 NS レコードは、デフォルトで hostname コマンドの出力結果を使う。そのため、親サーバで登録した NS レコードと食い違っていると、lame delegation が発生してしまう。実際にこのツールを利用してサーバを運用する場合は、これらの点に注意する必要がある。 h2n 以外の類似ツールも同様の問題を抱えていることが多いので、同じ注意が必要である。

実運用にあたっては minimum TTL については、生成されたファイルの値を変更しておく (h2n は、既存の SOA レコードがあれば、その値を尊重するようになっている) か、 h2n のソースコード中の値を変更するべきである。また、 MX レコードや NS レコードは、自動生成に頼らず、オプションや spcl. ファイルで明示的に指定するべきである。

6.2 mkrdns

DNSのゾーンファイル管理において、特に省力化を図るべき事項が、逆引きファイルの作成であり、この逆引きファイルの生成に特化した機能を持つツールがmkrdns(http://www.mkdrns.org/)である。mkrdnsは、BINDの設定ファイル(named.conf)の情報に基づいて、どの正引きファイルからどの逆引きファイルを生成すればよいかを判断し、Aレコードを転置したPTRレコードを生成する。

mkrdnsは、他のDNSサーバで作成されたゾーン情報に基づいて、逆引きファイルを生成することができるという点で、h2nや本ツールに比べて優れている。複数のドメインが複数のサブネット間にモザイク状にまたがっているLANにおいては、適正な逆引き情報を提供するには、mkrdnsのようなツールを併用することは有益である。

6.3 djbdns

BINDの次に有力なDNSサーバの実装として知られているのがdjbdnsである。djbdnsの作者は、qmailの作者として知られるD. J. Bernsteinであり、実行性能、および、セキュリティ面においてBINDよりも優れている。

ゾーンファイルのメンテナンスの観点からもdjbdnsは優れている。ファイル形式は独自だが、理解は容易である。Aレコードの記述をするだけで、内部で自動的に逆引き情報が生成される。一方で柔軟性もあり、逆引き情報を生成しないAレコードや、Aレコードと関係しないPTRレコードの記述も可能である。

現在、djbdnsはBIND 9のビュー機能に相当する機能を持っておらず、この点では不利である。しかし、djbdns運用の最大のネックになるのが、他のUnixのシステム管理ソフトウェアとの操作性(操作感)の違いであろう。このこと自体は、ソフトウェアの短所となるものではないが、普及の妨げとなっている要因の一つである。

6.4 Webmin

WWWをインターフェイスとして、サーバのシステム管理を行なうツールがWebmin(http://www.webmin.com/)である[9]。ウェブブラウザをユーザ・インターフェイスとすることができるので、Unixの操作に熟練しなくても、システム管理を行なうことができる。

DNSの管理もWebminで行なうことができる。正引き情報から逆引き情報を自動生成することもあり、メンテナンスは、全般に容易である。しかしながら、複数ドメインや、複数ビューを持つような、中・大規模サイトの管理は、さほど容易になるわけではない。

7 今後の課題

Makefileの自動生成 本ツールは、日常の運用には、ほぼ差し支えない水準に達しているが、管理するドメイン(ゾーン)が増えたとき、Makefileを作り直さなければならない。半自動的に生成することで、Makefileの文法ミスは発生しなくなるが、本ツールの中間ファイル生成の流れをよく理解する必要があるため、やや難解である。より広く利用者を募り、コメントにそった見直しをすべきであろう。

より大規模なサイトでの運用 秋田大学全体のドメインを管理するDNSサーバも、外向き、内向きに分けて運用しているが、現行の運用は難解で、保守作業に支障を来たしかねない。本ツールを拡張して適用することで、保守作業を容易にしたいと考えている。/etc/hosts形式のマスターファイルを一箇所に用意できないので、サブドメインからゾーン転送された情報をうまく処理することが、拡張の主眼となる。

謝辞

本ツールの前バージョンも含めて、実際に利用し、コメントをくださった、徳島大学総合科学部助教授 中島浩二氏、元秋田大学教育文化学部助教授 浅沼大海氏、教育文化学部環境情報講座技官 成田堅悦氏に感謝します。

参考文献

[1] P. Albitz and C. Liu: "DNS and BIND 4th Edition," O'Reilly, 2001

[2] P. Albitz and C. Liu: "DNS and BIND in a Nutshell" O'Reilly, 1992

[3] C. Liu: "DNS and BIND Cookbook," O'Reilly, 2002

[4] P. Mockapetris: "Domain Names — Concepts and Facilities," RFC 1034, 1987

[5] P. Mockapetris: "Domain Names — Implementation and Specification," RFC 1035, 1987

[6] D. Barr: "Common DNS Operational and Configuration Errors," RFC 1912, 1996

[7] R. Elz and R. Bush: "Clarifications to the DNS Specification," RFC 2181, 1997

[8] M. Andrews: "Negative Caching of DNS Queries (DNS NCACHE)," RFC 2308, 1998

[9] Joe Cooper: "The Book of Webmin," No Starch Press, 2002

[10] 佐々木重雄: BIND9 のビュー機能を配慮したゾーンファイル生成ツール, 情報処理学会東北支部 平成 15 年度第 1 回研究会報告 A-3-5, 2003