高速加入者回線（DSL）を利用したネットワークサーバの構築について

須　川　和　明

目　次

１．はじめに

近年，高速の加入者向け回線（DSL : Digital Subscriber Line）が誰にでも安価に利用できるようになり，サーバ用のハードウェアとしても，誰にでも簡単に手に入れられる部品だけを使用して，サーバ用に特化した高いコストパフォーマンスを持つサーバ専用のシステムを構築することが可能になった。

ソフトウェア面では，OS を始めサーバを利用して各種のサービスを行うために必要な多くのソフトウェアがオープンソースで公開されフリーで使用できるが，その半面，構築や維持管理，運営には多額のコストがかかるようにもなってきた。

本研究の目的は，このような回線とサーバを使用して，でき得る限りオープンソースとして公開されているソフトウェアのみを使用し，自前でサーバとネットワークの構築を行い，かつ，その維持管理，運営をも自前で行う方法について考察することである。

通信路としては，誰でも安価で手軽に加入できる高速の加入者向け回線（ADSL や FTTH など）を使用し，独自ドメインの取得には，フリーで利用できる DynamicDNS を利用する。固定IPアドレスや独自ドメインの取得・維持には経費がかかるが，DynamicDNS を利用することでその経費を削減することができる。

公開サーバを構築するためのソフトウェアとしては，OS を始め Web サーバやメールサーバなど，サーバとして必要な全てのソフトウェアをオープンソースのフリーソフトウェアのみを用いることにする。

ビジネスとしての利用を視点に入れた場合，現在では必須の条件となる通信路の暗号化とサーバの認証システムの構築にもオープンソースとして使用できるソフトウェアを利用する方法を探る。

２．ネットワークの構築

２．１　インターネットへの接続

DSL 回線を通して，LAN とインターネットの間で通信を行うためには，最低限，以下の設定が必要になる。

キーワード：オープンソース，Linux，インターネット

１．LAN の設定
２．PPPoE による ISP への接続
３．ルーティングの設定
４．ファイアウォールの設定

２．１．１ LAN の設定

サーバには２枚の NIC（Network Interface Card）が設置され，モデム（ADSL）あるいは回線終端装置（FTTH）側が eth0 で，LAN 側が eth1 であるとする。NIC の設定は，/etc/sysconfig/network-scripts/ 内の ifcfg-eth0，ifcfg-eth1 で行う。

eth0 は後述する PPPoE による接続開始後にアクティブになり IP アドレスが割り当てられるので，設定ファイル ifcfg-eth0 では，起動時には eth0 をアクティブにせず，IP アドレスも指定しない。

```
DEVICE=eth0
ONBOOT=no
IPADDR=空欄
```

一方，LAN 側のネットワークアドレスを 192.168.1.0/24 とするとき，eth1 には IP アドレスを始め，LAN の設定に必要な情報を ifcfg-eth1 に記述する。

```
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.1.1
NETMASK=255.255.255.0
GATEWAY=192.168.1.254
```

２．１．２ PPPoE による ISP への接続

DSL 回線を通して ISP（Internet Service Provider）への接続を開始するには，PPPoE クライアントを使用する。PPPoE（Point to Point Protocol over Ethernet）はイーサネット上で２点間の接続をユーザ ID やパスワードによる認証を行った上で確立するためのプロトコルである。

PPPoE クライアントは利用する回線事業者から配布されるが，ここでは標準的な RP-PPPoE を使用することにする。RP-PPPoE を使用するには PPP が必要であるが，Linux の標準的なインストールを行えば通常含まれている。なければ PPP パッケージをインストールしておかなければならない。

まず最初に/etc/ppp/pppoe.conf を編集して次の項目を指定する。ETH にはモデムあるいは回線終端装置側のイーサネットインターフェース名，USER には接続先のプロバイダから与えられたユーザ ID とドメイン名，DEMAND には接続形態を指定する。常時接続の場合は 'no' とする。DNS1，DNS2 には接続先のプロバイダから指定された DNS サーバの IP アドレスを指定する。ファイアウォールについては別途設定するので，FIREWALL は 'NONE' としておく。その他の設定項目については，とりあえずデフォルトのままで使用する。設定例［５．１．１］参照。

```
ETH='eth0'
USER='ユーザ ID@ドメイン名'
DEMAND=no
DNS1=123.45.67.8
DNS2=123.45.67.9
FIREWALL=NONE
```

次に接続に必要なユーザ ID とそのパスワードを /etc/ppp/pap-secrets および /etc/ppp/chap-secrets に記述する。このファイルは，root 以外が読めないようにファイルモードを600に設定しておく。設定例［５．１．２］参照。

```
"ユーザ ID@ドメイン名" * "パスワード"
```

接続を開始するには，root ユーザで次のコマンドを入力する。

```
# /etc/init.d/adsl start
```

システムのブート時に自動で接続を開始するには，以下のコマンドを実行して init ファイルに登録しておく。

```
# chkconfig --add adsl
```

DNS のリゾルバー用に/etc/resolv.conf にも DNS サーバの IP アドレスを記述しておく。

２．１．３　ルーティングの設定

LAN 側に接続されたコンピュータがインターネットと通信するためには，適切なルーティングとアドレスの変換が必要になる。

まず始めに，eth0 と eth1 の２つのネットワークインターフェース間でのパケットのフォワードが有効になるように /etc/sysctl.conf の設定を変更する。デフォルトでは無効になっているので，これを行わないとサーバはルーターとして機能しない。

```
# Controls IP packet forwarding
net.ipv4.ip_forward = 1
```

通常は自動的に認識されるが，デフォルトゲートウェイがインターネット側のインターフェース eth0 になるように，/etc/sysconfig/network に次の設定を記述しておく。

```
GATEWAYDEV=eth0
```

LAN 側にはインターネットからは識別できないプライベートな IP アドレスが設定されているので，LAN 側に接続された PC などのコンピュータがインターネットと通信を行うためには，IP マスカレードと呼ばれるアドレス変換を行ってグローバルアドレスを割り当てる必要がある。

アドレス変換には通常ファイアウォールとしても利用される iptables を使用する。iptables の設定は /etc/sysconfig/iptables に記述する。このファイルが存在しない場合は，/etc/sysconfig/ で次のコマンドを実行するとデフォルトの設定が保存される。

```
# iptables-save > iptables
```

このファイルの nat テーブルに POSTROUTING ルールを以下のように追加する。また，filter テーブルの FORWARD ルールのデフォルトは通常 DROP になっているが，これを ACCEPT に変更してパケットの通過を許可する。

```
*nat
:PREROUTING ACCEPT
:POSTROUTING ACCEPT
:OUTPUT ACCEPT
-A POSTROUTING
  -s 192.168.1.0/255.255.255.0
  -o eth0 -j MASQUERADE
COMMIT
*filter
:INPUT ACCEPT
:FORWARD ACCEPT
:OUTPUT ACCEPT
COMMIT
```

一方，LAN 側のコンピュータをインターネットに公開するためには，必要なIPアドレスとポートごとに PREROUTING ルールを追加する。以下の例では，プライベートアドレス192.168.1.5の80番ポートがグローバルアドレスの80番ポートに変換される。

```
-A PREROUTING -p tcp --dport 80
  -i eth1 -j DNAT --to 192.168.1.5
```

２．１．４　ファイアウォールの設定

前節の iptables の設定ではサーバは全てのパケットを受け入れ，出力，通過させてしまい，このままインターネットに接続することは危険である。

iptables は INPUT，OUTPUT，FORWARD という３つのルールを追加することで，入力，出力，通過するパケットにフィルタをかけ危険なパケットの侵入や不要なパケットの流出を防ぐことができる。設定例［５．１．３］参照。

まず filter テーブルのデフォルトを全て DROP に変更して全てのパケットを無視するように設定する。

```
*filter
:INPUT DROP
:FORWARD DROP
:OUTPUT DROP
```

次に，入ってくるパケットに対するINPUTルールについては，最低限必要なパケットのみを受け入れるように設定する。次の３行では内部から発信されたパケットのみの受け入れを許可している。

```
-A INPUT -s 127.0.0.1
    -d 127.0.0.1 -j ACCEPT
-A INPUT
    -s 192.168.0.0/255.255.255.0
    -i eth0 -j ACCEPT
-A INPUT
    -s 192.168.1.0/255.255.255.0
    -i eth1 -j ACCEPT
```

続いて，内部のコンピュータからインターネット上のコンピュータにセッションを確立した場合の応答のパケットの受け入れを許可する。これで内部のコンピュータがインターネットに接続して要求した情報を受け取ることが可能になる。

```
-A INPUT -m state
    --state RELATED,ESTABLISHED
    -j ACCEPT
```

内部のコンピュータをインターネットに公開してアクセスできるようにするには，IPアドレスとポート番号を指定して必要なルールだけを追加する。次の例は，IPアドレスが192.168.0.2であるWWWサーバに対してアクセスを許可している。

```
-A INPUT -d 192.168.0.2
    -i eth0 -p tcp -m tcp
    --dport 80 -j ACCEPT
```

通過するパケットに対するFORWARDルールは，LAN側のeth1からインターネット側のeth0への転送のみを許可する。さらにセッションを確立した後のパケットについては，eth0からeth1への転送も許可する。

```
-A FORWARD -i eth1
    -o eth0 -j ACCEPT
-A FORWARD -i eth0 -o eth1
    -p tcp -m state
    --state ESTABLISHED,RELATED
    -j ACCEPT
```

出て行くパケットに対するOUTPUTルールは，基本的に全て許可する。

```
-A OUTPUT -j ACCEPT
```

２．１．５　ブロードバンドルータによる接続

市販のブロードバンド用のルータを使用すると簡単にインターネットに接続することができる。PPPoEの設定を始め，アドレス変換，DHCPサーバ，パケットフィルタリングによるセキュリティ機能までWebブラウザを使用して設定できるようになっているものが多い。

ルータを使用する場合は，eth0とルータのNICを接続するので，eth0の設定ファイルifcfg-eth0はeth1と同様にIPアドレスなど必要な情報を記述する。eth0側のIPアドレスを192.168.0.1とすれば，以下のようになる。

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=192.168.0.1
NETMASK=255.255.255.0
GATEWAY=192.168.0.254
```

２．２　インターネットへの公開

DSL回線を使用してISPに接続した場合，接続元のIPアドレスとして通常ひとつのグローバルアドレスが割り当てられる。IPアドレスにドメイン名を付与してインターネット上に公開するためには，ドメイン名を取得して，既にインターネットに公開されているどこかのDNSサーバに登録してもらう必要がある。

ところがこのIPアドレスは，いったん接続を終了して再度接続し直した場合には，以前のアドレスと同じアドレスが再度割り当てられるという保証はなく，通常は別のアドレスが割り当てられることが多いため，その都度ドメイン名とIPアドレスの対応を変更しなければならない。

IPアドレスが変化するたびに，それを検知してドメイン名とIPアドレスの対応を自動的に変更してくれるサービスがDDNS（ダイナミックDNS）サービスである。多くのISPではこのサービスを無料で行っているが，ドメイン名の取得は通常有料であることが多い。ここでは，ドメイン名の取得とDDNSのサービスを無料で行っている代表的なサイトであるDynDNS.orgの利用法について述べる。

DynDNS.orgのDDNSサービスを利用するには，www.dyndns.orgにアクセスしてユーザ登録を行う。DnyDNS.orgで用意されている50種ほどのドメインサフィックスの中から適当なものを選択し，登録したいホストの名前を付けて公開するドメイン名を決める。希望したドメイン名が既に使用されている場合は使用できないので他のものに変更する。未使用のドメイン名が決まったら，登録画面からIPアドレスとドメイン名の組を登録する。ホスト名のワイルドカードやバックアップ用のメールサーバの指定なども行うことができる。これで，インターネット上から指定したドメイン名でサーバにアクセスすることが可能になる。

回線の切断などでIPアドレスが変更されたときには，DynDNS.orgにアクセスしてドメイン名とIPアドレスの対応を変更する必要がある。このような操作を自動的に行うプログラムがDDNSクライアントである。ここでは，Paul Burryによって開発された最も代表的なクライアントであるddclientについて述べる。ddclientはperlで書かれたプログラムで，指定した時間間隔でIPアドレスの変更をチェックして，変更があった場合にはDnyDNS.orgにアクセスしてDNSレコードの変更を行う。ddclientの設定ファイルddclient.confに，登録したドメイン名やユーザ名，パスワード，および，IPアドレスの変更を確認する時間間隔（秒数）などを指定する。設定例［５．１．４］参照。

ブロードバンドルータを使用して接続する場合には，ルータのDDNS自動変更機能を使用することができる場合もある。

一方，多くのISPでは，有料ではあるが固定IPアドレスを提供するサービスを行っているので，より安定した状態でサーバの公開を行うことも可能である。

３．各種サーバの構築

３．１　ドメインとDNSサーバ

IPアドレスとドメイン名を取得して，インターネットへの公開の準備が整ったら，次にインターネットへ公開するサーバのIPアドレスとホスト名，および，サイト内部で利用するサーバやクライアントのIPアドレスとホスト名を管理するために自前のDNSサーバを用意する必要がある。インターネットに公開するIPアドレスがひとつしかない場合であっても，サイト内には複数のサーバやクライアントがある場合には，内部用のDNSサーバを用意してホストを管理する。

３．１．１　BINDの設定

DNSサーバとしてはいくつかのプログラムが利用されているが，ここではその代表格であるBINDの設定法について述べる。BINDの最新のバージョン（2004年10月現在）であるBIND9では，ひとつのサーバで内部用，外部用など複数のサーバのように見せかけるビューという機能が導入されている。ここでは，ビューを利用してスプリットDNSサーバを構築する。他にもセキュリティの面

や IPv6 への対応など多数の機能の拡張が行われている。

BIND で必要になる設定ファイルは大きく分けて，BIND そのものの設定ファイルである /etc/named.conf と BIND が管理するゾーンファイルなどの DNS レコードのファイル群である。named.conf では，アクセス制限の設定と使用するゾーンファイルの名前を指定する。設定例［５．２．１］参照。

アクセス制限を設定するためには，アクセス制御リストを作成する。内部 LAN のネットワークアドレスを指定して localnet などという名前を付けておく。

```
// Local Network
acl localnet {
  192.168.0.0/24;
  192.168.1.0/24;
  127.0.0.1;
};
```

内部用のビューを localnet に限定することを設定。このビューで公開するゾーンファイルの列を指定する。

```
// LAN
view "inside" {
  match-clients { localnet; };
  recursion yes;

  zone "...." // ゾーンファイル
  ..........

};
```

同様に外部公開用のビューを設定し，ゾーンファイルの列を指定する。

```
// WAN
view "outside" {
  match-clients { any; };
  recursion no;

  zone "...." // ゾーンファイル
  ..........

};
```

３．１．２　ゾーンファイルの作成

公開する DNS レコードを含むゾーンファイルの構成は，BIND9 以前のものと基本的に同様である。ただし，TTL（キャッシュの有効時間）は必須になっているので，ゾーンファイルの先頭行で指定しておく。設定例［５．２．２－５］参照。

```
$TTL 86400            ;TTL の値（秒）
@ IN SOA ns.some.dom. root.some.dom. (
     2004102602       ;serial
     28800            ;refresh
     3600             ;retry
     604800           ;expire
     86400 )          ;minimum
  IN NS ns.some.dom.
<名前> <タイプ> <クラス> <値>
```

３．２　電子メール

電子メールのサービスを提供するためには，最低限でも，サーバ間でメールを転送する MTA（Mail Transfer Agency）と受信したメールをユーザの MUA（Mail User Agency）に配信するための POP または IMAP などのサーバが必要になる。サーバ間でメールを転送する MTA としては，古くから使われている sendmail を始め Postfix, qmail などが UNIX では代表的である。ここでは，設定の容易さとセキュリティで定評があり，sendmail との互換性もある Postfix の使用法について述べる。POP サーバや IMAP サーバも多数のプログラムが公開されているが，POP を拡張して IMAP の機能を持たせた Yat サーバの使用法について述べる。

電子メールのサービスを提供する場合に，ウイルス対策やスパムメール対策は欠かせない。ウイルス対策用としては，オープンソースではないがフリーでの使用が認められている AntiVir MailGate の使用法について述べる。スパムメール対策のためのプログラムも多数公開されているが，ここでは SpamAssassin の使用法について述べる。

３．２．１　Postfix サーバの設定

Postfix の設定ファイル main.conf は sendmail の sendmail.cf とは異なり，テキストで設定事項を記述できるため，設定が非常に容易になった。Postfix に付属の main.conf には詳細な説明が記述されており，数ヶ所の設定を変更するだけで，ほとんどはデフォルトのまま使用できる。設定例［５．３．１］参照。

メールサーバの FQDN（Full Qualified Domain Name）とドメイン名をそれぞれ指定する。

```
# メールサーバ名
myhostname = mx.some.domain
# ドメイン名
mydomain = some.domain
```

メールの不正な転送や不要なメールを受け取らないためには，最低でも次の設定を行っておく必要がある。

```
# 受信を許可する宛先
mydestination = $myhostname,
    localhost.$mydomain $mydomain
# 送信を許可するクライアント
mynetworks = 192.168.1.0/24,
    127.0.0.0/8
# 受信を許可するユーザリスト
local_recipient_maps =
    $alias_maps unix:passwd.byname
```

独自ドメインからのメールは携帯電話などでは受信を拒否される場合があるので，ISP のメールサーバに転送を依頼することもできる。

```
# 転送を依頼するホスト名
relayhost = mx.other.domain
```

３．２．２　POP サーバの設定

POP サーバとしては，株式会社ジェプロで開発されオープンソースとして公開されている POP3 拡張版のYatサーバを使用する。Yat サーバは POP3 の機能に加えてサーバ側に残したメールの管理とメールを振り分けるフォルダを管理するコマンド群が用意され，IMAP に似たメールの管理が可能になる。ユーザの認証のためには UNIX パスワードの他に APOP パスワードも使用可能で，APOP パスワードを管理するためのサーバ ascyd も公開されている。

yatd を使用するには，inetd.conf の pop-3 設定部分を以下のように yatd で置き換える。

```
pop-3 stream tcp nowait root
    /usr/local/sbin/yatd yatd
```

yatd の設定ファイルは yatsvrrc であるが，大半はデフォルトのままで使用できる。認証のためのパスワードは，デフォルトでは UNIX パスワードであるが，以下のようにパスワードの使用を APOP のみに変更することもできる。

```
ApopPermit: yes
UserPermit: no
```

３．２．３　ウイルス対策

メールのウイルス対策用のソフトウェアについては，そのほとんどが有料でしかも高価なものが多く，フリーで利用できるものは少ない。H+BEDV から公開されている AntiVir MailGate は，一定の条件の下ではフリーで利用でき，ウイルス定義ファイルの自動更新サービスもフリーで受けることができる。

AntiVir MailGate を利用して送受信メールのウイルスチェックを行うためには，AntiVir MailGate の avgate デーモンが25番ポートでメールを待ち受け，ウイルスチェックを行った後で，Postfix の10025番ポートにメールを渡すように設定変更する。設定例［５．３．２］参照。

まず /etc/services に10025番ポートのサービス smtp-backdoor を追加する。

```
smtp-backdoor   10025/tcp
```

続いて，通常はデフォルトのままで使用する Postfix の master.cf の１行を以下のように変更して Postfix が10025番ポートからメールを受け取るようにする。

```
#smtp inet n - n - - smtpd
localhost:smtp-backdoor inet
                    n - n - - smtpd
```

その他の設定ファイル avmailgate.conf や antivir.conf はデフォルトのままで使用できる。

ウイルス定義ファイルの自動更新サービスを受けるためには，H+BEDV のサイトにアクセスして，ユーザ登録を行っておく必要がある。ユーザ登録を行うとライセンスキーファイル hbedv.key がメールで送られてくるので /usr/lib/Antivir/ にコピーしておく。このライセンスキーは１年間有効であるが，何度でも更新できる。

３．２．４　スパム対策

スパムメール対策用の SpamAssassin はテキスト解析技術によるメールヘッダと本文に対するチェックと複数のインターネットリアルタイムブラックリストの情報を組み合わせて spam を検知する。ユーザがスパムと判定したいメールの情報を与えて以後スパムとして振り分けるよう学習させることも可能である。

SpamAssassin はユーザレベルでの利用の他にサーバにインストールしてサイトレベルで運用することもできる。ここではサイトレベルでの運用の方法について述べる。SpamAssassin をサーバレベルで利用するには，spamd と spamc を使用する。まず，デーモンプログラム spamd を起動しておき，SpamAssassin を利用したいユーザはホームディレクトリ内の procmailrc に spamc へのフィルタの設定を記述する。

```
:0fw
| spamc
:0:
* ^X-Spam-Status: Yes
spam/.    # スパム保存用フォルダ
```

スパムとして判定されたメールは，以下のヘッダを追加して Mail/spam/に保存される。

```
X-Spam-Status: Yes
```

スパムとして扱いたいメールがある場合は，例えば Mail/newspam/ などのフォルダに保存しておき，sa-learn コマンドで学習させることもできる。

```
% sa-learn --spam Mail/newspam
```

３．３　WWW サーバ

インターネットに接続してサーバを公開する目的として，電子メールと同様に重要なサービスが WWW である。ここでは，WWW サーバとして Netcraft の調査で世界の６割以上を占めると言われる Apache サーバと Apache サーバに組み込まれてさまざまなサービスの提供を可能にする Apache のモジュールについて述べる。

３．３．１　Apache

Apache には，現在（2004年10月），バージョン1.3系列とバージョン2.0系列のディストリビューションがある。ここでは，SSL モジュールの標準化を始め多くの拡張が行われているバージョン2.0系列のディストリビューションについて述べる。

Apache は Linux の多くのディストリビューションで標準的なパッケージとしてインストールされることが多いが，ここでは，必要な拡張機能を DSO（Dynamic Shared Object）モジュールとしてサーバ運用後に組み込んで使用することが可能なようにインストールしておく。

Apache サーバの設定ファイルは，基本的に

は httpd.conf のひとつであり，ほとんどの設定はデフォルトのまま使用できるが，CGI や Virtual Host の機能を利用するためには以下のような設定を追加する。

```
# The Options directive
Options ... ExecCGI

# AddHandler
AddHandler cgi-script .cgi

# Virtual Hosts
NameVirtualHost www2.some.dom

<VirtualHost www2.some.dom>
  ServerAdmin webmaster@some.dom
  DocumentRoot /www2/htdocs/
  ServerName www2.some.dom
  ErrorLog logs/www2-error_log
  CustomLog logs/www2-access_log common
</VirtualHost>
```

３．３．２　SSL 対応サーバ

Apache はバージョン2.0系列から，通信路を暗号化して情報を保護するための SSL（Secure Socket Layer）モジュール mod-SSL が標準で組み込まれるようになった。

SSL 対応サーバは通常のサーバの Virtual Host として起動する。SSL 用の設定ファイルである ssl.conf に Virtual Host の設定，および，サーバや認証局の証明書と暗号鍵のファイル名などを記述する。

```
## SSL Virtual Host Context
<VirtualHost _default_:443>

#  General setup for the virtual host
DocumentRoot "/usr/local/apache2/htdocs"
ServerName www.some.dom:443
ServerAdmin webadmin@some.dom
ErrorLog logs/ssl_error_log
TransferLog logs/ssl_access_log

#  Server Certificate:
SSLCertificateFile
    /usr/local/apache2/conf/
              ssl.crt/server.crt
#  Server Private Key:
SSLCertificateKeyFile
    /usr/local/apache2/conf/
              ssl.key/servercert.key
#  Certificate Authority (CA):
SSLCACertificateFile
    /usr/local/apache2/conf/
              ssl.crt/cacert.pem
</VirtualHost>
```

３．３．３　サーバ証明書

正式なサーバ証明書を取得するためには，商用の認証局（VeriSign, Thawte など）に証明書署名要求（CSR）を送って署名してもらう必要があるが，有料であり高額でもある。

通信路を暗号化して情報を保護するだけであれば，自前の認証局を立ち上げて，証明書を発行することもできる。認証局を作成するためには，OpenSSL の CA スクリプトを利用して，認証局のパスフレーズ，および，認証局の名前や所在地などの情報を入力する。

```
% CA -newca
```

カレントディレクトリに demoCA というディレクトリが作成され，その中に認証局の証明書 cacert.pem や認証局の秘密鍵 private/cakey.pem などが作成される。

認証局を作成したら，サーバの秘密鍵，証明書署名要求（CSR）を作成する。

```
% openssl req -new
      -keyout server.key -out server.csr
```

カレントディレクトリにサーバの秘密鍵 server.key とサーバの証明書署名要求 server.csr が作成される。

続いて，サーバの証明書署名要求に対して，自前の認証局で署名を行う。

```
% openssl ca
      -in server.csr -out server.crt
```

署名済みのサーバ証明書 server.crt が作成される。

このままでは，Apache SSL サーバを起動するたびに，サーバのパスフレーズを要求され

るので，多少セキュリティに難があるが，パスフレーズを消去した秘密鍵を作成しておく。

```
% openssl rsa -in server.key
      -out servercert.key
```

パスフレーズを消去した秘密鍵servercert.keyが作成される。

ここで作成したcacert.pem，server.crt，servercert.keyをssl.confで指定した場所にコピーする。

この間に作成された全ての秘密鍵cakey.pem，server.key，servercert.keyはファイルのモードを変更して，他ユーザから読めないようにしておかなければならない。

３．３．４　PHP

PHP（PHP：Hypertext Preprocessorの再帰的頭字語）は汎用スクリプト言語で，HTML文書中にスクリプトを埋め込む形で記述できるという特性を持ったWebアプリケーション開発用言語である。

PHPはPerlなどのスクリプト言語と同様にシェルからも使用可能で，WWWサーバと連携してCGI用言語としても利用できるが，通常は，ApacheのDSOモジュールとして組み込まれて使用される。このためCGIのように別のプロセスを起動する必要がなく，サーバの負荷も最小限に抑えることができるとともに処理速度の点でも飛躍的に向上する。

PHPをApacheのDSOモジュールとして組み込んで利用するには，httpd.confに以下のような設定を追加する。

```
# Dynamic Shared Object (DSO) Support
LoadModule php4_module modules/libphp4.so

# DirectoryIndex
DirectoryIndex index.html index.php

# Add for PHP 4.x, use:
AddType application/x-httpd-php .php
AddType
   application/x-httpd-php-source .phps
```

PHPプログラムを含むファイルには拡張子.phpを付ける。ソースコードを公開したいときは，拡張子.phpsを付ける。

３．３．５　データベースの利用

MySQLはPHPの開発の当初から標準的に使用されてきたオープンソースのデータベースエンジンとして知られ，PHPと親和性の高いデータベースエンジンとして使用されている。MySQLはマルチユーザ・マルチスレッド対応のSQLデータベースエンジンで，PHPからアクセスするための多数の関数が用意されている。

PostgreSQLはオープンソースとして公開され，トランザクション処理が可能な代表的なデータベースエンジンとして知られている。PostgreSQLもMySQL同様にPHPから多数の関数を使用してアクセスすることが可能である。

３．４　ネットワークの管理

３．４．１　NTPタイムサーバ

インターネットにサーバを公開する場合に，サーバが正確な時間を保持していることが求められるが，コンピュータ内蔵の時計機能はそれほど正確なものではないため，ネットワーク上のタイムサーバと接続して同期を取りながら正確な時間を維持する必要がある。

インターネット上にある最上位（Stratum1）のタイムサーバは通常GPSシステムを利用して正確な時間を保持している。公開されているタイムサーバやISPから提供されるタイムサーバとの間で時間の同期を取る為のプロトコルがNTP（Network Time Protocol）で，ntpdプログラムが指定したタイムサーバと接続して自サーバの時刻を自動的に修正する。同時にサイト内の他のコンピュータに対するタイムサーバとしても機能する。

ntpdの設定ファイルはntp.confで，通常は接続先のタイムサーバのIPアドレスを指定

するだけである。

```
# /etc/ntp.conf
server 123.45.67.8
```

3.4.2 SNMP エージェント

サーバのサービスの質を維持あるいは向上させるためには，ネットワークやサーバの状態についてのさまざまな情報を常に把握しておく必要がある。

ネットワークを通してサーバやルータなどの種々の情報を自動的に収集するためのプロトコルの代表的なものが SNMP （Simple Network Management Protocol）である。SNMP は，サーバやネットワーク機器などに常駐して情報を収集するエージェントとその情報を取得して管理するマネージャから構成される。エージェントは収集した情報を MIB（Management Information Base）と呼ばれるデータベースに保存し，マネージャからの要求に応じて必要なデータを送信する。

SNMP エージェントおよびマネージャとして使用するプログラムは多数公開されているが，ここではカーネギーメロン大学と UCD で開発されオープンソースとして公開されている Net-SNMP について述べる。

SNMP エージェントとして動作するプログラムは snmpd で，その設定ファイルは snmpd.conf である。ここでは，コミュニティ名，セキュリティグループ，ビュー，アクセス制限，システム情報などについて設定する。

```
#       sec.Name source          community
com2sec local    localhost       private
com2sec mynet    192.168.1.0/24 mynetwork

#     groupName   sec.Model sec.Name
group localgroup  v1        local
group localgroup  v2c       local
group localgroup  usm       local
group mygroup     v1        mynet
group mygroup     v2c       mynet
group mygroup     usm       mynet

#    name incl/excl subtree   mask
view all    included  .1      80
view system included  system  fe

#      group      context sec.model
    sec.level prefix read   write notif
access localgroup ""       any
    noauth    exact  all    none  none
access mygroup    ""       any
    noauth    exact  system none  none

# System contact information
sysname     System Site Name
syslocation System Site Location
syscontact  webadmin@some.dom
```

MIB 情報を表示してテストするには，snmpwalk コマンドを使う。

```
% snmpwalk -v2c -c private localhost
```

3.4.3 MRTG による監視

SNMP エージェントから取得した MIB 情報をグラフ化して表示するために MRTG（Multi Router Traffic Grapher）を使用する。グラフ化された MIB 情報はWebブラウザで表示して監視することができる。

MRTG の設定ファイル mrtg.cfg は，MRTG 付属の cfgmaker スクリプトを使用して作成する。

```
% cfgmaker > mrtg.cfg
```

cfgmaker ではネットワークインターフェースのトラフィック情報の設定部分だけが作成される。

```
### Global Config Options
#  for UNIX
WorkDir: /home/http/mrtg

### Interface 1 >> Descr: 'eth0'
Target[localhost_1]: 1:private@localhost:
SetEnv[localhost_1]:
        MRTG_INT_IP="192.168.1.1"
        MRTG_INT_DESCR="eth0"
MaxBytes[localhost_1]: 1250000
Title[localhost_1]: Traffic Analysis
        for 1 -- localhost
PageTop[localhost_1]: <H1>Traffic Analysis
```

```
        for 1 -- localhost</H1>
<TABLE>
 <TR><TD>System:</TD>
        <TD>localhost</TD></TR>
 <TR><TD>Maintainer:</TD>
        <TD>webadmin</TD></TR>
 <TR><TD>Description:</TD>
        <TD>eth0  </TD></TR>
 <TR><TD>ifType:</TD>
        <TD>ethernetCsmacd (6)</TD></TR>
 <TR><TD>ifName:</TD>
        <TD></TD></TR>
 <TR><TD>Max Speed:</TD>
        <TD>1250.0 kBytes/s</TD></TR>
 <TR><TD>Ip:</TD>
        <TD>192.168.1.1 (aleut)</TD></TR>
</TABLE>
```

その他の MIB 情報を取得してグラフ化するには，Target 行に MIB 情報の OID を指定する。OID は必ず２つセットで指定する。OID1 が緑色，OID2 が青色でグラフ化される。設定例［５．４．１］参照。

```
Target[統計情報名]: OID1&OID2:
    コミュニティ名@エージェントのホスト名
```

設定ファイルを作成したら，mrtg.cfg を指定して mrtg を起動する。設定ファイルで指定されたディレクトリに HTML ファイルが作成される。

```
% /mrtg_bin/mrtg /mrtg_cfg/mrtg.cfg
```

以後は一定の時間間隔でグラフが更新されるように crontab に追加する。以下は５分間隔で更新される。

```
0-55/5 * * * * /mrtg_bin/mrtg
      /mrtg_cfg/mrtg.cfg >& /dev/null
```

４　おわりに

本研究では，UNIX 系のサーバによるネットワークの構築を行い，インターネットへの公開とその維持管理の方法について考察した。この分野の宿命として，その結果はすぐにも陳腐化するであろうことは疑いないが，次の新たなる研究に向けての暫しの間の研究ノートとしたい。

世界で使用されているコンピュータの大半を占める Windows 系のコンピュータと連携して，ファイルの共有やユーザの管理などを UNIX 系のサーバで統合的に行うためのソフトウェアも多数開発され公開されている。本研究ではこのような異種の OS を持つコンピュータによるネットワークの構築については言及できなかったが，今後の課題としたい。

本研究は北星学園大学2003年度特別研究費第３号（個人研究）によるものである。

５．設定例

５．１　インターネット接続の設定例

５．１．１　pppoe.conf

```
# /etc/ppp/pppoe.conf

# Ethernet card connected to ADSL modem
ETH='eth2'

# ADSL user name.
USER='aaaa@bbb.ccc.ddd'

# Bring link up on demand?
DEMAND=no
#DEMAND=300

# DNS type: SPECIFY=use DNS1 and DNS2;
DNSTYPE=SPECIFY

# Obtain DNS server addresses from the peer
PEERDNS=no

DNS1=192.168.0.2
DNS2=192.168.0.1

# Make the PPPoE connection your default route.
DEFAULTROUTE=yes
```

５．１．２　pap-secrets, chap-secrets

```
# /etc/ppp/pap-secrets

"user-id@bbb.ccc.dom"   *   "password"
```

５．１．３　iptables の設定例

```
# Generated by iptables-save
*nat
```

```
:PREROUTING ACCEPT
:POSTROUTING ACCEPT
:OUTPUT ACCEPT
-A PREROUTING -p tcp --dport 80 -i eth1 -j DNAT
    --to 192.168.1.5
-A POSTROUTING -s 192.168.1.0/255.255.255.0
    -o eth0 -j MASQUERADE
COMMIT
# Completed
# Generated by iptables-save
*filter
:INPUT DROP
:FORWARD DROP
:OUTPUT DROP
-A INPUT -s 127.0.0.1 -d 127.0.0.1 -j ACCEPT
-A INPUT -s 192.168.0.0/255.255.255.0
    -i eth0 -j ACCEPT
-A INPUT -s 192.168.1.0/255.255.255.0
    -i eth1 -j ACCEPT
-A INPUT -s 192.51.195.3 -d 192.168.0.2
    -i eth0 -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -d 192.168.0.2 -i eth0 -p tcp -m tcp
    --dport 25 -j ACCEPT
-A INPUT -d 192.168.0.2 -i eth0 -p tcp -m tcp
    --dport 80 -j ACCEPT
-A INPUT -d 192.168.0.2 -i eth0 -p tcp -m tcp
    --dport 443 -j ACCEPT
-A INPUT -s 192.168.0.2 -j ACCEPT
-A INPUT -s 192.168.1.3 -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED
    -j ACCEPT
-A INPUT -s ! 192.51.195.3 -d 192.168.0.2 -i eth0
    -p tcp -m tcp --dport 22 -j LOG --log-level 1
-A FORWARD -i eth1 -o eth0 -j ACCEPT
-A FORWARD -i eth0 -o eth1 -p tcp -m state
    --state ESTABLISHED,RELATED -j ACCEPT
-A FORWARD -i eth0 -o eth1 -p icmp
    --icmp-type echo-reply -j ACCEPT
-A FORWARD -i eth0 -o eth1 -p icmp
    --icmp-type destination-unreachable -j ACCEPT
-A OUTPUT -s 127.0.0.1 -d 127.0.0.1 -j ACCEPT
-A OUTPUT -d 192.168.0.0/255.255.255.0 -o eth0
    -j ACCEPT
-A OUTPUT -d 192.168.1.0/255.255.255.0 -o eth1
    -j ACCEPT
-A OUTPUT -j ACCEPT
COMMIT
# Completed
```

5.1.4 ddclient.conf

```
# /etc/ddclient.conf
daemon=300     # check every 300 seconds
syslog=yes     # log update msgs to syslog
mail=root      # mail update msgs to root
pid=/var/run/ddclient.pid  # record PID

## To obtain an IP address from Web
use=web, web=checkip.dyndns.org/, \
   web-skip='IP Address'
# found after IP Address

## user login
login=username     # default login
password=password  # default password
wildcard=yes       # add wildcard CNAME?

## dyndns.org dynamic addresses
server=members.dyndns.org, \
   protocol=dyndns2 \
   some.dom
```

5.2 BIND9の設定例

```
# /etc/named.conf
options {
    directory "/var/named";
    pid-file "/var/named/run/named.pid";

include "/var/named/keys/rndc.key";
controls {
    inet 127.0.0.1 port 953
    allow { 127.0.0.1; } keys { "RNDC"; };
};

// Local Network
acl localnet {
    192.168.0.0/24;
    192.168.1.0/24;
    127.0.0.1;
};

// LAN
view "inside" {
   match-clients { localnet; };

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

   // localhost Reverse
   zone "0.0.127.in-addr.arpa" {
     type master;
     file "named.0.0.127.in-addr.arpa";
   };

   //  My Domain Normal
   zone "some.dom" {
     type master;
     file "some.dom.local";
   };

   // My Domain Reverse 0
   zone "0.168.192.in-addr.arpa" {
     type master;
     file "named.0.168.192.in-addr.arpa";
   };
};

// WAN
view "outside" {
   match-clients { any; };

   //  My Domain Normal
   zone "some.dom" {
     type master;
     file "some.dom.zone";
     allow-transfer { IP of Secondary DNS; };
   };

   // My Domain Reverse
   zone "6.121.219.in-addr.arpa" {
     type master;
     file "named.6.121.219.in-addr.arpa";
     allow-transfer { IP of Secondary DNS; };
   };
};
```

5.2.1 named.conf

```
; Local Domain
$TTL 86400
@  IN  SOA  ns.some.dom.
   root.some.dom. (
        2004022606  ;serial
        28800       ;refresh
```

```
          7200        ;retry
          604800      ;expire
          86400       ;minimum
          )

          IN  NS      ns.some.dom.
          IN  MX  10  some.dom.

          IN  A       192.168.0.2
ns        IN  A       192.168.0.2
aleut     IN  A       192.168.0.2

www       IN  A       192.168.0.2
router    IN  A       192.168.0.1
win2000   IN  A       192.168.0.11
landisk   IN  A       192.168.0.200
```

5.2.2 some.dom.local

```
; Local Domain
$TTL 86400
@  IN  SOA  ns.some.dom.
   root.some.dom. (
        2004022606  ;serial
        28800       ;refresh
        7200        ;retry
        604800      ;expire
        86400       ;minimum
        )

        IN  NS      ns.some.dom.
        IN  MX  10  some.dom.

        IN  A       192.168.0.2
ns      IN  A       192.168.0.2
aleut   IN  A       192.168.0.2

www     IN  A       192.168.0.2
router  IN  A       192.168.0.1
win2000 IN  A       192.168.0.11
landisk IN  A       192.168.0.200
```

5.2.3 some.dom.zone

```
; SOME.DOM
$TTL 86400
@  IN  SOA  ns.some.dom.
   root.some.dom. (
        2004022602  ;serial
        28800       ;refresh
        3600        ;retry
        604800      ;expire
        86400       ;minimum
        )

        IN  NS  ns.some.dom.  ;Primary
;       IN  NS  SecondaryNameServer  ;Secondary

        IN  MX  10  some.dom.

        IN  A       219.165.130.214
ns      IN  A       219.165.130.214
aleut   IN  A       219.165.130.214
www     IN  A       219.165.130.214
```

5.2.4 named.0.0.127.in-addr.arpa

```
; Localhost
$TTL 86400
@  IN  SOA  ns.some.dom.
   root.some.dom. (
        2004022601  ;serial
        28800       ;refresh
        7200        ;retry
        604800      ;expire
        86400       ;minimum
        )

        IN  NS  ns.some.dom.
1       IN  PTR localhost.
```

5.2.5 named.0.168.192.in-addr.arpa

```
; Local Domain 0
$TTL 86400
@  IN  SOA  ns.some.dom.
   root.some.dom. (
        2004022601  ;serial
        28800       ;refresh
        7200        ;retry
        604800      ;expire
        86400       ;minimum
        )

        IN  NS  ns.some.dom.
2       IN  PTR ns.some.dom.
```

5.3 Postfixの設定例

5.3.1 main.cf

```
# /etc/postfix/main.cf

# INTERNET HOST AND DOMAIN NAMES
#
myhostname = mx.some.dom

# The mydomain parameter
#
mydomain = some.dom

# SENDING MAIL
#
myorigin = $mydomain

# The mydestination parameter
#
mydestination = $myhostname,
    localhost.$mydomain $mydomain

# Alternatively,
#     you can specify the mynetworks list
#
mynetworks = 192.168.1.0/24, 127.0.0.0/8

# INTERNET OR INTRANET
#
relayhost = mx.some.other.dom

# REJECTING UNKNOWN LOCAL USERS
#
local_recipient_maps = $alias_maps
                       unix:passwd.byname

# ALIAS DATABASE
#
#alias_maps = dbm:/etc/aliases
alias_maps = hash:/etc/postfix/aliases
#alias_maps = hash:/etc/aliases,
#                   nis:mail.aliases
#alias_maps = netinfo:/aliases
```

```
# The mail_spool_directory parameter
#
mail_spool_directory = /var/spool/mail

# The mailbox_command parameter
#
mailbox_command =
    /usr/bin/procmail -a $DOMAIN -d $LOGNAME

# SHOW SOFTWARE VERSION OR NOT
#
smtpd_banner = $myhostname ESMTP $mail_name

# DEBUGGING CONTROL
#
debug_peer_level = 2

# The debugger_command
#
debugger_command =
    PATH=/usr/bin:/usr/X11R6/bin
    xxgdb $daemon_directory/$process_name
        $process_id & sleep 5

# The allow_percent_hack parameter
#
allow_percent_hack = no

# For FML
allow_mail_to_commands = alias,forward,include
```

5.3.2 master.cf

```
# /etc/postfix/master.cf

#smtp   inet  n  -  n  -     - smtpd
localhost:smtp-backdoor inet n - n - - smtpd
#628    inet  n  -  n  -     - qmqpd
pickup  fifo  n  -  n  60    1 pickup
cleanup unix  n  -  n  -     0 cleanup
qmgr    fifo  n  -  n  300   1 qmgr
#qmgr   fifo  n  -  n  300   1 nqmgr
rewrite unix  -  -  n  -     - trivial-rewrite
bounce  unix  -  -  n  -     0 bounce
defer   unix  -  -  n  -     0 bounce
flush   unix  n  -  n  1000? 0 flush
smtp    unix  -  -  n  -     - smtp
showq   unix  n  -  n  -     - showq
error   unix  -  -  n  -     - error
local   unix  -  n  n  -     - local
virtual unix  -  n  n  -     - virtual
lmtp    unix  -  -  n  -     - lmtp
```

5.4 MRTGの設定例

5.4.1 mrtg.cfg

```
### Global Config Options
#  for UNIX
WorkDir: /home/http/mrtg

### Interface 1 >> Descr: 'eth0'

Target[localhost_1]: 1:private@localhost:
SetEnv[localhost_1]: MRTG_INT_IP="192.168.1.1"
      MRTG_INT_DESCR="eth0"
MaxBytes[localhost_1]: 1250000
Title[localhost_1]:
      Traffic Analysis for 1 -- localhost
PageTop[localhost_1]:
      <H1>Traffic Analysis for 1 -- localhost</H1>
 <TABLE>
  <TR><TD>System:</TD>
      <TD>localhost</TD></TR>
  <TR><TD>Maintainer:</TD>
      <TD>webadmin</TD></TR>
  <TR><TD>Description:</TD>
      <TD>eth0  </TD></TR>
  <TR><TD>ifType:</TD>
      <TD>ethernetCsmacd (6)</TD></TR>
  <TR><TD>ifName:</TD>
      <TD></TD></TR>
  <TR><TD>Max Speed:</TD>
      <TD>1250.0 kBytes/s</TD></TR>
  <TR><TD>Ip:</TD>
      <TD>192.168.1.1 (localhost)</TD></TR>
 </TABLE>

### CPU Load Average

Target[cpu]:
    1.3.6.1.4.1.2021.10.1.5.2
    &1.3.6.1.4.1.2021.10.1.5.3:
    private@localhost
MaxBytes[cpu]: 2000
Title[cpu]: CPU Load Average
PageTop[cpu]: <H1>CPU Load Average</H1>
Options[cpu]: nopercent, integer, gauge,
                absolute, withzeroes
YLegend[cpu]: CPU Load Average
ShortLegend[cpu]: percent
Legend1[cpu]: 5 min CPU Load Average
Legend2[cpu]: 15 min CPU Load Average
LegendI[cpu]: 5min
LegendO[cpu]: 15min


### Memory Usage

Target[mem]:
    1.3.6.1.4.1.2021.4.6.0
    &1.3.6.1.4.1.2021.4.4.0:
    private@localhost
MaxBytes1[mem]: 128000
MaxBytes2[mem]: 256000
Title[mem]: Memory Usage
PageTop[mem]: <H1>Memory Usage</H1>
Options[mem]: gauge, absolute
YLegend[mem]: Mem Free[kBytes]
ShortLegend[mem]: kBytes
Legend1[mem]: Real Memory
Legend2[mem]: Swap Memory
LegendI[mem]: Real
LegendO[mem]: Swap

### Router

Target[192.168.0.1_1]: 1:public@192.168.0.1:
SetEnv[192.168.0.1_1]: MRTG_INT_IP="192.168.0.1"
    MRTG_INT_DESCR="KS_MAC1_T"
MaxBytes[192.168.0.1_1]: 1250000
Title[192.168.0.1_1]:
    Traffic Analysis for 1 -- none
PageTop[192.168.0.1_1]:
    <H1>Traffic Analysis for 1 -- none</H1>
 <TABLE>
   <TR><TD>System:</TD>
       <TD>none in </TD></TR>
   <TR><TD>Maintainer:</TD>
       <TD>Linksys</TD></TR>
   <TR><TD>Description:</TD>
       <TD>KS_MAC1_T  </TD></TR>
   <TR><TD>ifType:</TD>
       <TD>ethernetCsmacd (6)</TD></TR>
   <TR><TD>ifName:</TD>
       <TD></TD></TR>
   <TR><TD>Max Speed:</TD>
       <TD>1250.0 kBytes/s</TD></TR>
   <TR><TD>Ip:</TD>
       <TD>192.168.0.1 (router)</TD></TR>
 </TABLE>
```

［参考文献］

(1) B. Laule, P. Laurie, Apache ハンドブック, O'Reilly, 1999

(2) L. Stein, D. MacEachern, Apache 拡張ガイド(上, 下), O'Reilly, 2000

(3) S. Garfinkel, G. Spafford, Web セキュリティ & コマース, O'Reilly, 1998

(4) J. Castagnetto et al., プロフェッショナル PHP プログラミング, インプレス, 2001

(5) R. J. Yarger et al., MySQL & mSQL, O'Reilly, 2000

(6) C. Musciano, B. Kennedy, HTML, O'Reilly, 2000

(7) S. Gundavaram, CGI プログラミング, O'Reilly, 1999

(8) O. Kyas, インターネットセキュリティ, Thomson, 1997

(9) 須川和明, PHP による Web アプリケーションサーバの構築－コミュニケーションサービス－, 北星論集43巻第１号, 2003

(10) L. Mamakos et al., A Method for Transmitting PPP Over Ethernet (PPPoE), RFC2516, 1999

(11) S. Thomso et al., Dynamic Updates in the Domain Name System (DNS UPDATE), RFC2136, 1997

(12) T. Dierks et al., The TLS Protocol, RFC2246, 1999

(13) R. Fielding et al., Hypertext Transfer Protocol－HTTP/1.1, RFC2616, 1999

(14) T. Berners-Lee et al., Hypertext Transfer Protocol－HTTP/1.0, RFC1945, 1996

(15) J. Klensin Ed., Simple Mail Transfer Protocol, RFC2821, 2001

(16) J. Myers, M. Rose, Post Office Protocol - Version 3, RFC1939, 1996

(17) R. Gellens et al., POP3 Extension Mechanism, RFC2449, 1998

(18) M. Crispin, Internet Message Access Protocol - Version 4rev1, RFC3501, 2003

(19) D. L. Mills, Network Time Protocol (Version 3) Specification, Implementation and Analysis, RFC1305, 1992

(20) J. Case et al., A Simple Network Management Protocol (SNMP), RFC1157, 1990

(21) P. Mockapetris, Domain Names－Concepts and Facilities, RFC1034, 1987

(22) P. Mockapetris, Domain Names－Implementation and Specification, RFC1035, 1987