DNS & mail

InternetWeek ’98 チュートリアル

1998/12/15

中村 素典

motonori@econ.kyoto-u.ac.jp

スケジュール

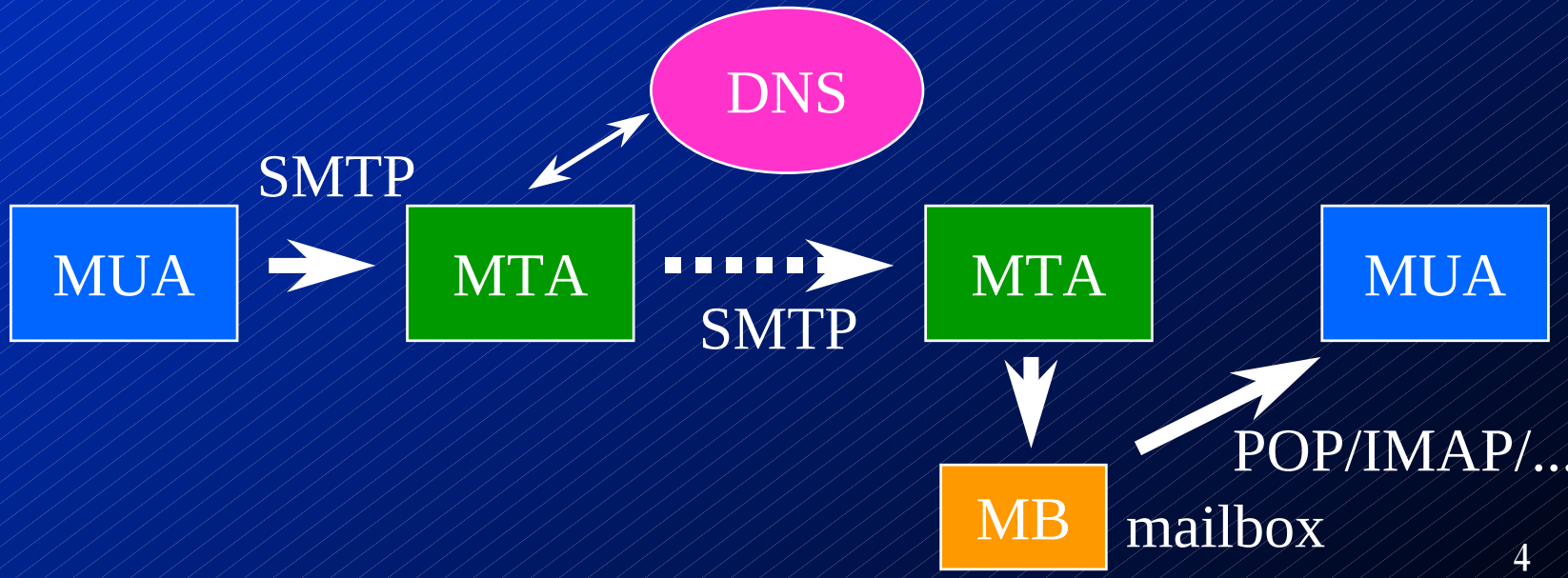
- インターネットメールの基礎
- DNSの仕組みと管理

- メールシステムのデザイン
- SPAM対策

1. インターネットメールの基礎

電子メールシステム

- MUA (Mail User Agent)
- MTA (Mail Transfer Agent)
- DNS (Domain Name System)



MUA (Mail User Agent)

ユーザ・アプリケーション

- メールを読む
- メールを書く
- メールを保存/検索する

- UNIX
  - ucbmail, RMAIL, mush, MH (mh-e), mew,....
- Windows
  - OutLook, Netscape Mail, Eudora,....

MTA (Mail Transfer Agent)

- メールの受信
- 配信先の決定
- メールの配信
  - リモートへ、ローカルへ、発信者へ（エラー）

- Store and Forward
  - 一旦受信した後、次のホストへの転送を試みる

MTA Programs

- sendmail http://www.sendmail.org/
- qmail http://www.qmail.org/
- SMAIL (GNU)
- MMDF (Multi-channel Memo Distribution, CSNET)
- exim http://www.exim.org/
- VMail http://wzv.win.tue.nl/vmail/
- LSMTP http://www.lsoft.com/LSMTP.html
- PP (X.400)

インターネットでのメールの送受信

- SMTP - Simple Mail Transfer Protocol RFC821(S)
- TCP のポート 25 番

- ほとんどのMTAはSMTPの実装をもつ
  - DNSとの連携機能をもつ

SMTPの様子

220 r.domain SMTP Server ready (サーバからのメッセージ)

HELO s.domain (サーバへのメッセージ)

250 r.domain Hello s.domain

MAIL FROM:<sender@s.domain> (発信者のアドレス)

250 sender ok

RCPT TO:<recipient@r.domain> (受信者のアドレス)

250 recipient ok

DATA

354 Enter mail, end with "." on a line by itself

電子メールのデータがここにくる

. (データの終端を示す)

250 Message accepted for delivery

QUIT

221 r.domain closing connection

インターネットにおける メールの送信先の決定方法

- あて先メールアドレスのホスト名を抽出

  `user@host`

- ホスト名からIPアドレスを取得

  host → 12.34.56.78

  – /etc/hosts

  – NIS (YP)

  – DNS (Domain Name System)

DNS (Domain Name System)

- 広域分散ディレクトリ・サービス
  - 分散配置
  - 分散管理

- ホスト名 → IPアドレス
- メールアドレス $\xrightarrow[\text{MX}]{}$ ホスト名 → IPアドレス

  - 同じドメイン空間を共用している

用語

- 配信
  - ローカルに配信 → メールボックス
  - リモートに配信 → 別の MTA へ渡す
- 転送
  - リモートに配信
- 受理 (たぶん一般的な用語ではない)
  - ローカルに配信
- 受信
  - リモートから配信

メールアドレス

- 発信者/受信者情報として利用
- ユーザ部 @ ドメイン部
  - motonori @ wide.ad.jp
- その他の形式
  - %-Hack
  - Route Address
  - UUCP addressing

%-Hack

- RFC1123(S)

user % host @ relay

sender → relay → host

➧ relay に届いた時点で user @ host に書き換え

user % host % relay2 @ relay1

sender → relay1 → relay2 → host

Route Address

- RFC822(S)

@relay: user @ host

sender → relay → host

➧ relay **に届いた時点で** user @ host **に書き換え**

@relay1, @relay2: user @ host

sender → relay1 → relay2 → host

UUCP addressing

- host ! user
- relay ! host ! user

- host ! user @ domain の解釈
  - “host ! user” @ domain　(Internet的)
    - » sender → domain → host
  - host ! “user @ domain”　(UUCP 的)
    - » sender → host → domain

コメント形式

- Full Name <user@domain>
- user@domain (Full Name)
- user(User Name)@domain(Company Name)
  - ( ) のコメントはどこに入ってもよい

ドメイン部

- Fully Qualified Domain Name
  - インターネットドメイン形式の完全なホスト/ドメイン名
- Fully Qualified Mail Address
  - user@mailhost.wide.ad.jp
  - user@mailhost ではないことを意味する
- Not Qualified Mail Address
  - user
- Generic Address
  - user@wide.ad.jp

メッセージの形式

- ヘッダ(header)と本文(body)

  RFC822(S): Standard for the format of arpa internet text messages

- 最初の空行が区切り

```
From: announce@nic.ad.jp
To: motonori@wide.ad.jp
Subject: InternetWeek ’98
      ← 空行 (空白もなし)
InternetWeek ’98 のお知らせ
```

発信者と受信者

- 発信者 (Sender)
  - 1人
  - ヘッダの発信者は複数のことも
    - » 意味上の発信者
- 受信者 (Recipient)
  - 1人または複数人

ヘッダとエンベロープ(cont.)

- 封書に似ている
- エンベロープ (envelope)
  - 投函した人/届け先
  - 封書の表書きの送り主/宛先
    - » 実際に事務作業を行なう人
  - 配送に際に書き換えられていく
- RFC821(S): Simple Mail Transfer Protocol
  - エンベロープはコマンドで指定
- UUCP
  - エンベロープは rmail のコマンド･ラインに指定

ヘッダとエンベロープ(cont'd)

- ヘッダ (header)
  - 本文を書いた人/読んで欲しい人
  - 内封された書面の送り主/宛て先
  - 基本的に書き換えられない
- ヘッダとエンベロープの送信者/受信者
  - 一緒の場合もある
    - » 個人宛て
  - 異なる場合もある
    - » メーリングリストなど

エンベロープはいつ作られるか

- ヘッダから抽出される
  - 送信するMUAが行う
  - 最初に処理を行なうMTAが行う
- エンベロープは配信処理で書き換えられる
  - 転送
  - メーリング・リスト

返信に利用するアドレス

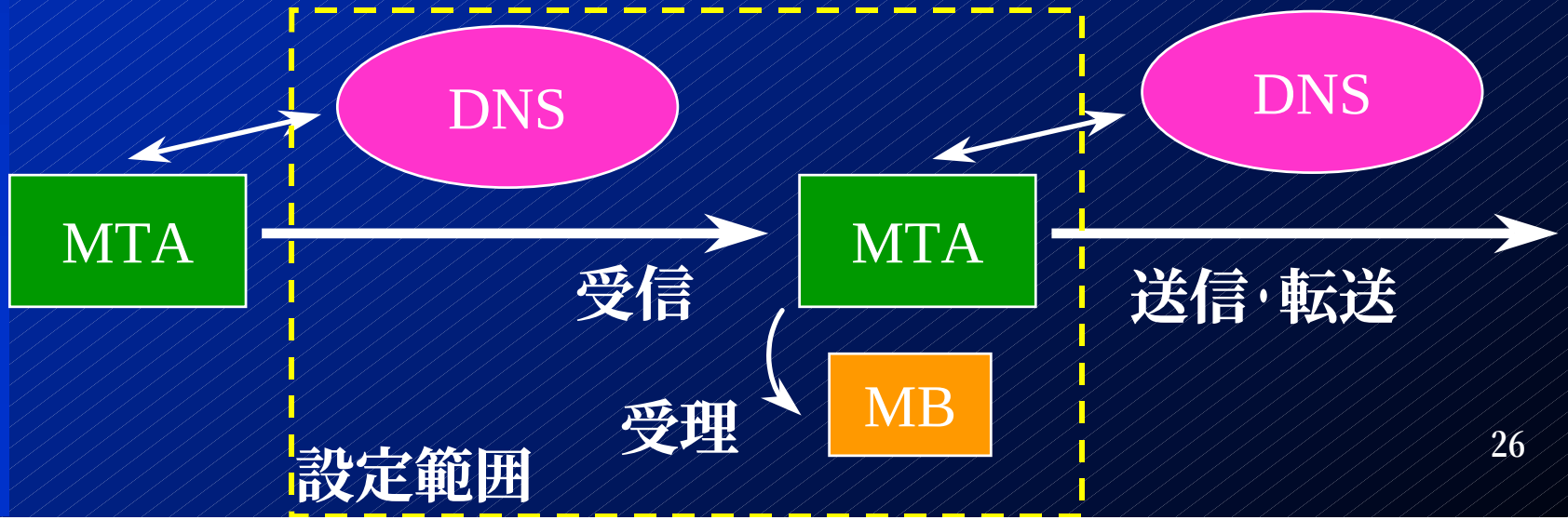
- 配信エラー通知の返送(自動)
  - エンベロープの発信者
  - Errors-To: ヘッダ
    - » エンベロープの概念がないシステム用（まだあるの？）
- 内容への返事(人が介在)
  - ヘッダの発信者
    - » From:, Reply-To:
    - » (To:, Cc:)

メールボックスから MUA へ

- ローカル・メールボックス
  - UNIX など
- POP
- IMAP

メールの配信の3つのポイント

1) 受信(リモートからの配信)
   – リモートのメールサーバから送ってもらう
2) 受理(ローカルへの配信)
3) 送信・転送(リモートへの配信)
   – 受信者のメールサーバに送りつける



1) メールを送ってもらうための設定

どうやって送り先を教えるか

- Internet
  - SMTPによる直接配信
  - → DNSに配信先を定義
- バケツリレー・システム
  - UUCPなど(JUNET時代)
  - → 経路上の(すべての)ホストに配信先を設定
  - mailconfの活躍
    - » sendmail.cf作成ツール

メール配信時に参照されるDNSのレコード

- A (Address) RR (Resource Record)
  - ホスト名からIPアドレスを取得
- MX (Mail eXchanger) RR
  - メールアドレスから配信先ホスト名を取得
- CNAME (Canonical NAME) RR
  - ホストの別名を取得

nslookupでAを確認(1)

% nslookup sh.wide.ad.jp.

Server: localhost

Address: 127.0.0.1

Name: *sh.wide.ad.jp*

Address: *203.178.137.73*

複数のIPアドレスを持つホスト

```
mail.x.co.jp        IN  A  12.34.56.78
                    IN  A  12.34.54.32
```

- 最初のアドレスへの配信に失敗した場合、順に全てのアドレスを試みる (実装依存)
- DNSのラウンドロビン機能により、検索で得られる順序は毎回変わる
  - 負荷分散
  - 最初のアドレスしか試みない実装でも、そのうち届く(?)

nslookupでAを確認(2)

% nslookup jp-gate.wide.ad.jp

Server: localhost

Address: 127.0.0.1

Name: *jp-gate.wide.ad.jp.*

Addresses: *203.178.137.17, 203.178.136.81,*
203.178.137.75, 203.178.136.89

Generic なメールアドレス

- ホスト名部分を持たない
  - ホストの改廃に依存しない
- MX (Mail eXchanger) RR を利用
- user@x.co.jp 宛てのメールは
  指定されたホストに送られてくる
  - MX を引いて、得られた右辺のホスト名について、さらに A を引き IP アドレスを取得

nslookupでMXを確認

% nslookup -q=mx wide.ad.jp.

Server: localhost

Address: 127.0.0.1

wide.ad.jp preference = 10, mail exchanger = sh.wide.ad.jp

:

(additional information)

sh.wide.ad.jp internet address = 203.178.137.73

- 送信先は、MXがみつからないときはAに従い、両方あればMXが優先されることに注意
  - つまりMXによってメールを別のホストに向けることも可能

障害に備える(MX編)

- メールの受信の代行

  x.co.jp preference=10, mx=mail1.x.co.jp

  preference=50, mx=mail2.x.co.jp

- 数字が小さい程優先度が高い(コスト値)
  - 送り側は配信に成功するまで
    順にコストの大きなものへと配信を試みる
- mail2 は mail1 の回復後に mail1 へ転送
  - mail2 のメールの保存期間に注意

Lower MXの条件
(メール･ループを防ぐための条件)

- MX RR の右辺ある自分の名前の認識
  - 自分自身への接続の防止
    - » sendmail -bt において $=w で確認
    - » インタフェースのアドレス応じた名前は自動登録
    - » qmail は IP アドレスで確認がおこなわれる
      - 自分のIPアドレスには接続にいかない
- 自分の名前に対する MX RR のプレファレンス以上のコストのRRを捨てる
  - Lower MX 間でのピンポンの防止

負荷分散

x.co.jp preference=10, mx=mail1.x.co.jp.
Preference=10, mx=mail2.x.co.jp.

- 同じコストの場合は送り側が乱数で選ぶ
- 最終的に一つのメールボックスへ
  - 受信側に仕掛けが必要
    - » 静的な配送定義など

2) 送ってもらったメールの受理

- 届いたメールを自分宛てとして認識
  - ローカルに配信(受理)
  - 「送られてきた = 自分あてである」ではない
- 自分宛でないと判断した場合
  - 転送先を探す

受理するアドレスの設定

- Sendmail (CF)
  - ACCEPT_ADDRS に定義
- qmail
  - /var/qmail/control/locals に定義

受信設定のまとめ

- 相手に送り先を教えること
  - MX レコードを定義
- 自分宛てだと解釈すること
  - ローカルへの配信 (受理)

別個に設定が必要

3) メールの配信設定

配信方法のバリエーション

- DNSのMX RR参照による配信
  - MX を参照する MTA の準備
- ホスト名のみによる配送
- 固定ルールによる配信
  - DNS を参照する必要性の検討

DNS参照のための基本設定

- /etc/resolv.conf
- サービススイッチファイル

/etc/resolv.conf

- ネームサーバの指定

  nameserver 0.0.0.0 (localhost - 127.0.0.1 と解釈される)

  nameserver 12.34.56.78

  nameserver 12.34.56.79

  – 3つまで (MAXNS in resolv.h)

    » いくつでもタイムアウト時間は変わらない (75s)

  domain sub.x.co.jp

  search sub1.x.co.jp sub2.x.co.jp x.co.jp

  – アドレス補完の際に利用

サービススイッチファイル

- Solaris
  - /etc/nsswitch.conf
    - » hosts: files dns
- DEC
  - /etc/svc.conf
- その他
  - ServiceSwitchFile オプション (sendmail.cf)
  - デフォルト: /etc/service.switch

    hosts dns files nis

DNSのMXを参照する場合

- MX を参照する MTA
  - sendmail.mx
    - » libresolv.a のリンク
  - MX 参照用 sendmail.cf
    - » MX_SENDMAIL=yes (CF)
    - » (実際には Wildcard MX 対策だけ)
      - → アドレスの補完

固定ルールによる配信

- sendmail.cf に固定ルールを書く
  - mailconf
  - CF
    - » STATIC_ROUTE_FILE

配信の動作確認

- アドレスの解釈が正しいか
  - sendmail -bv あるいは sendmail -bt の /parse
- MX が正常に検索できているか
  - sendmail -bt で /mx コマンド
- 実際に送ることができるか
  - sendmail -v

配信設定のまとめ

- ホストが DNS を参照できるようにする
  - resolv.conf
  - サービス･スイッチ･ファイル
- メールアドレスごとの配信先を考える
  - DNS (MX) を参照してそのまま配信する
    - » どのネームサーバを見るか (後述)
  - 静的に配信先を設定する

DNSの仕組みと管理

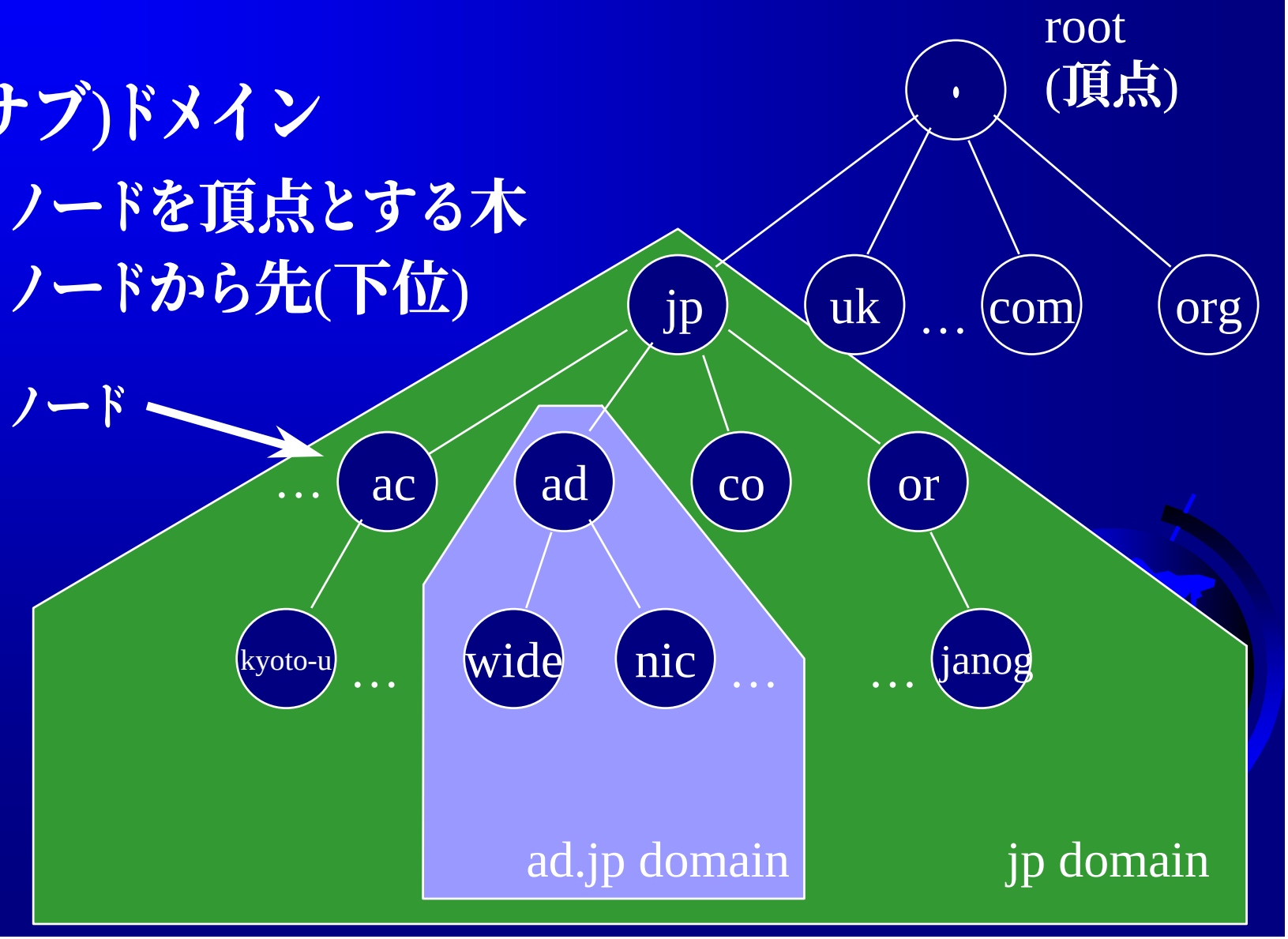
- 内容
  - ドメインとゾーン
  - サーバの種類
  - サーバの設定
  - レコード詳説
  - アドレスの補完
  - Wildcard MX
  - CIDRと逆引き
  - よくある設定の誤り

DNS (Domain Name System)

- 広域分散データベース
- ホスト名とIPアドレスの対応表
- 組織ごとに自主管理
  - 大昔は /etc/hosts で集中管理

ドメイン・ツリー

- (サブ)ドメイン
  - ノードを頂点とする木
  - ノードから先(下位)



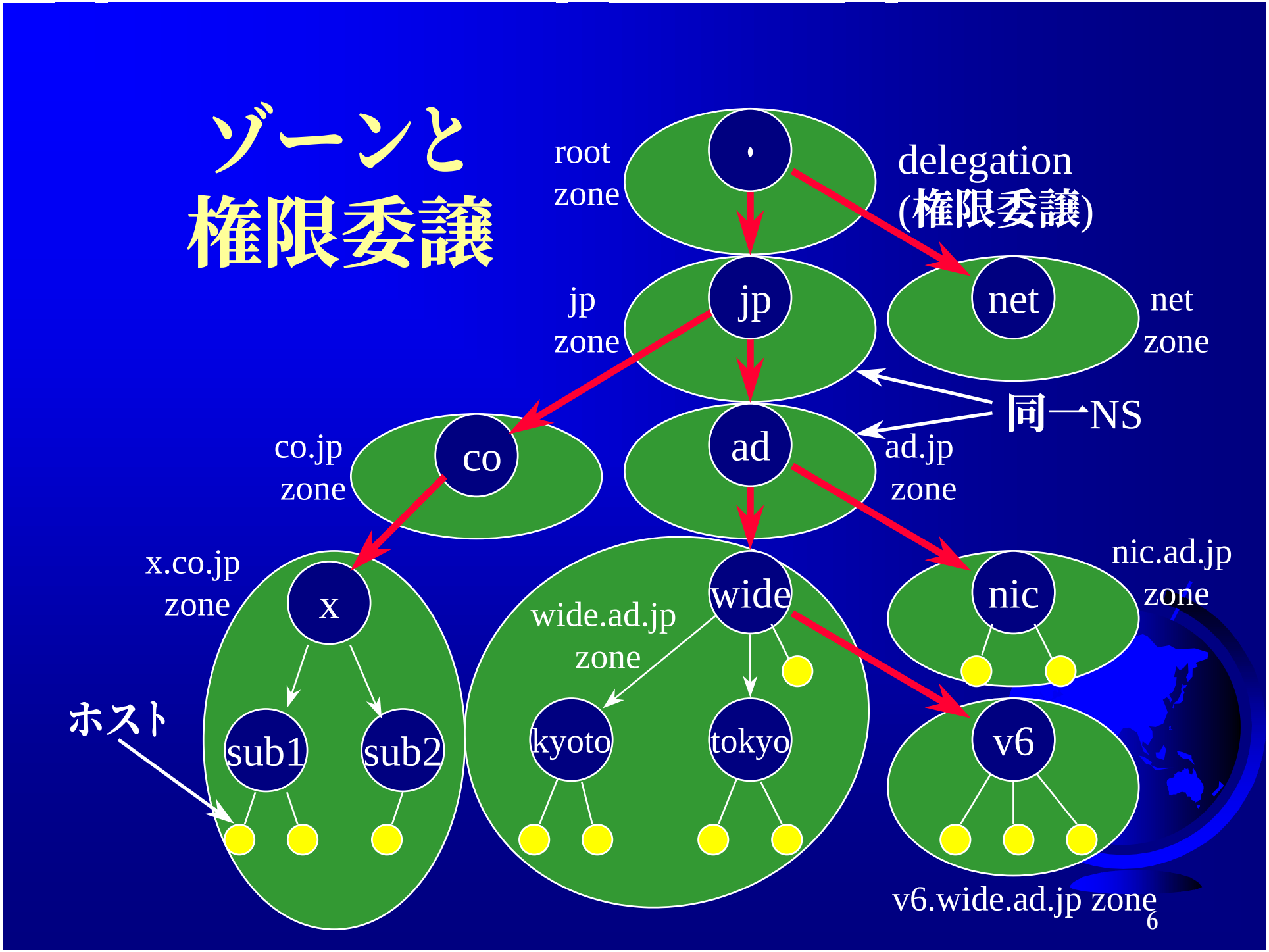
分散管理と検索

- 必要に応じてノード間の上下リンクで分割
  - ノードの下流へのリンク
    - Delegation(権限委譲)
  - TOP domain, 2nd(3rd)-level domain
    - NIC が管理
- 単方向リンク(上から下へ)
  - 上位へはrootまで戻ってから辿る
  - 全サーバはrootを知っている

ゾーンとドメイン

- 必ずしもノード単位で分割管理する必要なし
- ゾーン
  - 共同管理される隣接ノードの集合
  - 必ずしもドメインとは一致しない
    - 1ゾーン内に複数サブドメインを定義可能
  - データの管理単位
    - 部所単位/地域単位に分割
    - 1つのネームサーバに対応
  - 末端ではドメインと一致

ゾーンと権限委譲


root
zone
.
delegation
(権限委譲)
jp
zone
jp
net
net
zone
同一NS
co.jp
zone
co
ad
ad.jp
zone
x.co.jp
zone
x
wide.ad.jp
zone
wide
nic
nic.ad.jp
zone
ホスト
sub1
sub2
kyoto
tokyo
v6
v6.wide.ad.jp zone

サーバの種類

- サービスの種類で分類
  - データ提供用 (検索もする) / 検索専用
- データ(ゾーン)の管理方法で分類
  - そこで編集(Primary) / 他からコピー(Secondary)
- 権限で分類
  - Authorized / Unauthorized
- サービス対象で分類
  - 組織外向け / 組織内向け

提供するデータ(ゾーン)の管理 (cont.)

- プライマリ(マスタ)・サーバ
  - データベース・ファイルの編集を行なう
- セカンダリ(スレーブ)・サーバ
  - プライマリのサービス・バックアップ
  - プライマリ・サーバからデータをコピー
    - 別のセカンダリからでも一応可
      - コピーのチェイン
    - コピー元サーバを複数指定可能
  - 同時に到達不能にならない場所に配置

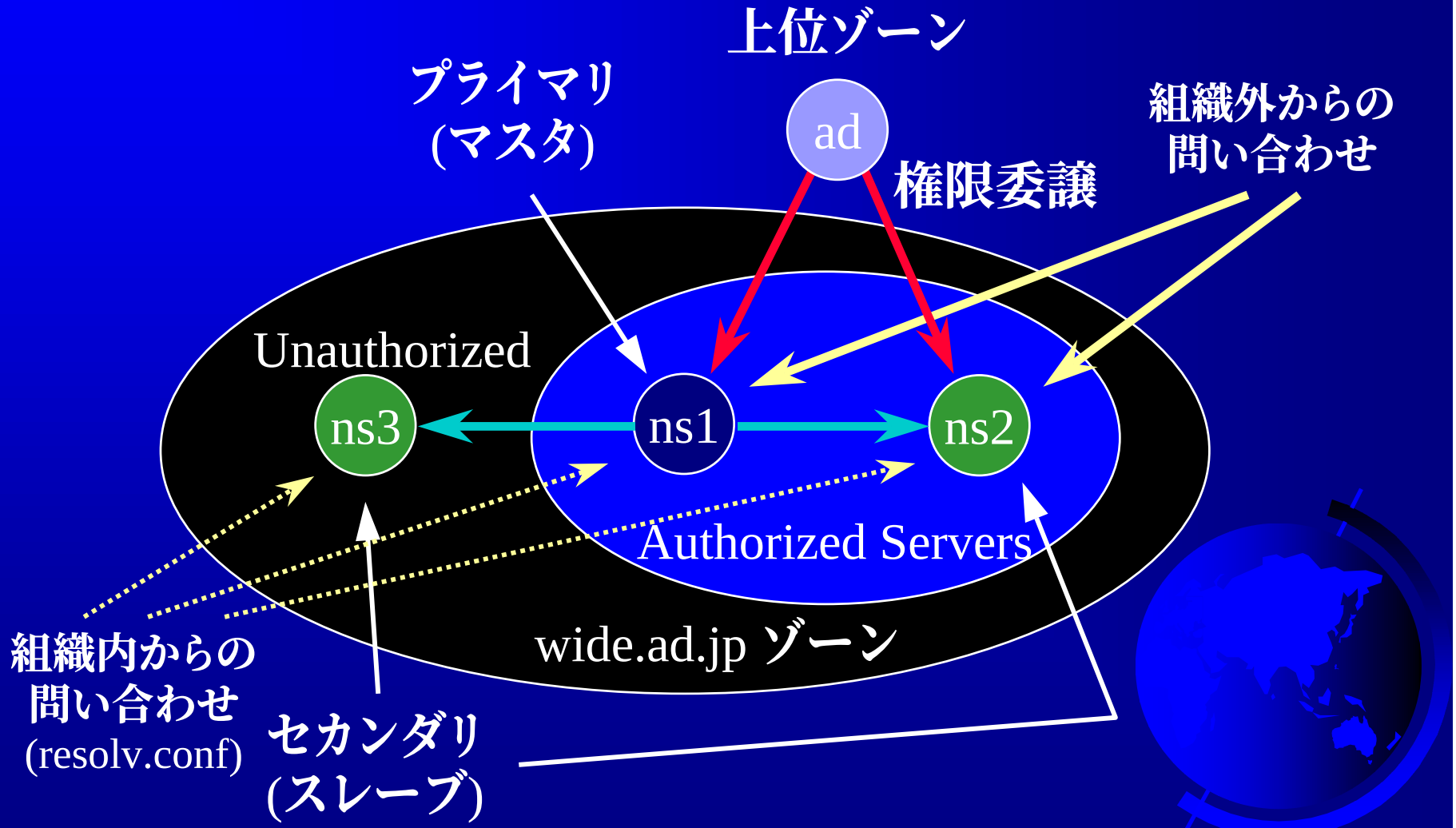
提供するデータ(ゾーン)の管理 (cont'd)

- 検索要求は平等に来る
  - プライマリ･セカンダリの区別はない
- ゾーンに対する区別
  - 一つのサーバで複数のゾーンを管理
    - ゾーンAに対してはプライマリ
    - ゾーンBに対してはセカンダリ
  - サーバ個体に対する区別ではない

データの提供に関する権限

- Authorized Server
  - データをインターネットに提供
  - 上位ゾーンからのリンク(権限委譲)がある
- Unauthorized Server
  - 手元の恒常的キャッシュ
  - データを近隣クライアントに提供
  - 上位ゾーンからのリンク(権限委譲)がない
- ゾーンに対する区別

サーバの権限とゾーン


上位ゾーン
プライマリ
(マスタ)
ad
権限委譲
組織外からの
問い合わせ
Unauthorized
ns3
ns1
ns2
Authorized Servers
wide.ad.jp ゾーン
組織内からの
問い合わせ
(resolv.conf)
セカンダリ
(スレーブ)

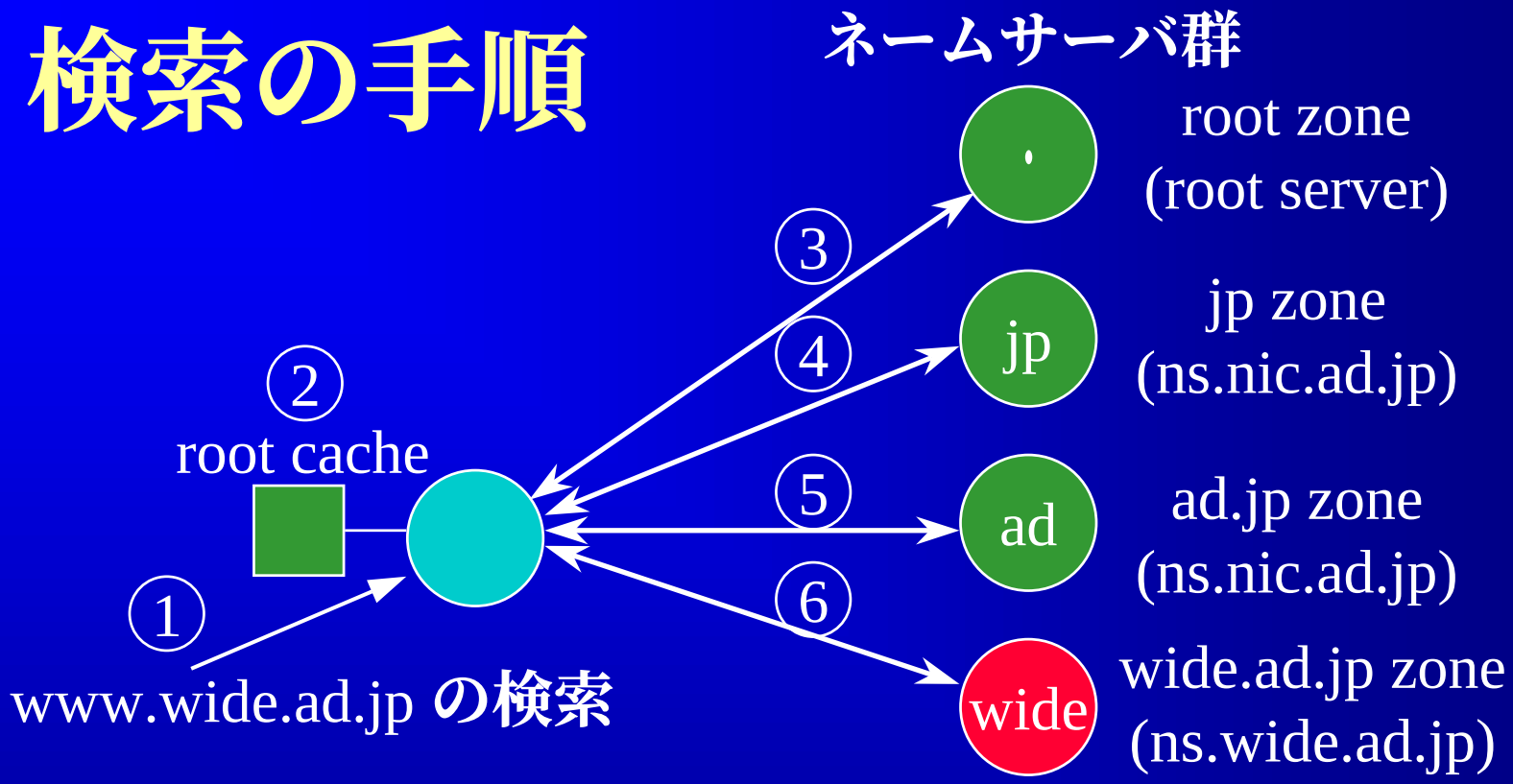
検索専用

- キャッシュサーバ
  - 一度検索したデータをしばらく記憶
    - 2度目以降はUnauthoritative Answer として応答
  - プライマリでもセカンダリでもない
    - どのゾーンに対しても

参考：ネガティブ・キャッシュ

- 該当レコードが存在しなかったことを保持 (全サーバ)

検索の手順



- root server への到達性がなければ引けない
  - 国際線の安定性問題
  - 国内に root server が必要 (m.root-servers.net)
  - jp zone の Unauthorized Secondary に

DNS Servers

- Berkeley Internet Name Domain (BIND) Server
  - bind 4.9.7
  - bind 8.1.2
    - できるだけ最新版を
      - セキュリティ、パフォーマンス、信頼性、新機能
  - http://www.isc.org/bind.html
- Windows NT のネームサーバなど
  - 信頼性は大丈夫(?)

サーバの設定ファイル

- /etc/named.boot (bind 4)
- /etc/named.conf (bind 8)
  - named-bootconf.pl でフォーマット変換
    - named.boot から named.conf に
    - bind 8 に添付

BIND では ‘; ’ がコメントの開始

sample of named.boot (bind 4)

```
; デフォルトディレクトリ
directory /etc/namedb
; 起動時に知っておくべきデータ (ルートサーバ情報)
cache      .                          root.cache
; localhost に関する情報
primary    localhost                  localhost
primary    0.0.127.in-addr.arpa       127.rev
; プライマリとして提供するゾーン
primary    wide.ad.jp                 wide
primary    136.178.203.in-addr.arpa   203.178.136.rev
; セカンダリとして提供するゾーン
secondary v6.wide.ad.jp               203.178.136.188     sec/v6
```

sample of named.conf (bind 8)

```
options {
        directory "/etc/namedb";
};

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

zone "localhost" {
        type master;
        file "localhost";
};
```

```
zone "0.0.127.in-addr.arpa" {
        type master;
        file ”127.rev";
};

zone "wide.ad.jp" {
        type master;
        file "wide";
};

zone "136.178.203.in-addr.arpa" {
        type master;
        file ”203.178.136.rev";
};
```

```
zone "v6.wide.ad.jp" {
        type slave;
        file "sec/v6";
        masters {
                203.178.136.188;
        };
};
```

root cache

- ルートサーバに関する情報
  - ルートサーバさえ知れば全てを検索可能
- ftp://ftp.rs.internic.net/domain/named.root
- 13番目が日本で稼働開始(1997/8)
  - m.root-servers.net
- Firewall の内側では
  - 内部向け root server を用意
  - Forwarders でがんばる

sample of root.cache

```
; formerly NS.INTERNIC.NET
.                        3600000  IN  NS  A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.  3600000      A   198.41.0.4
;
; formerly NS1.ISI.EDU
.                        3600000      NS  B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.  3600000      A   128.9.0.107
     :
     :
; housed in Japan, operated by WIDE
.                        3600000      NS  M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET. 3600000      A   202.12.27.33
```

forwarders

- 組織内から外部のアドレスの問い合わせ
  - 外部のネームサーバに問い合わせを転送
    - socks 対応 firewall などの場合
  - slave とともに指定

    forwarders 12.34.56.79 (内外両側からアクセス可能なサーバ)

    slave (options forward-only - 4.9.3 or later)
- キャッシュの有効利用
  - データを特定のサーバに集約
  - トラフィックを抑える
    - 回線が細いときなど

sample of localhost

```
; $ORIGIN       localhost.
@       IN      SOA     ns.wide.ad.jp.      postmaster.wide.ad.jp. (
                1       ; Serial number
                172800  ; Refresh every 2 days
                3600    ; Retry every hour
                1728000 ; Expire every 20 days
                172800 ); Minimum 2 days
;
        IN      NS      localhost .
;
        IN      A       127.0.0.1
```

sample of 127.rev

```
; $ORIGIN       0.0.127.in-addr.arpa.
@       IN      SOA     ns.wide.ad.jp.      postmaster.wide.ad.jp. (
                1       ; Serial number
                172800  ; Refresh every 2 days
                3600    ; Retry every hour
                1728000 ; Expire every 20 days
                172800 ); Minimum 2 days
;
        IN      NS      localhost.
;
0       IN      PTR     loopback-net.       ; ネットワークの名前
        IN      A       255.0.0.0           ; ネットマスク
1       IN      PTR     localhost.
```

sample of wide (cont.)

```
; $ORIGIN wide.ad.jp.
@                   IN  SOA  ns.wide.ad.jp. two.wide.ad.jp. (
                             1998112301   ; Serial
                             3600         ; Refresh
                             900          ; Retry
                             3600000      ; Expire
                             3600         ; Minimum
                        )
                    IN  NS     ns
                    IN  NS     ns.tokyo
                    IN  MX 10  sh.wide.ad.jp.
                    IN  MX 20  jp-gate.wide.ad.jp.
ns                  IN  A      203.178.136.63
ns.tokyo            IN  A      203.178.136.61
```

sample of wide (cont'd)

```
sh              IN  A       203.178.137.73
jp-gate         IN  A       203.178.137.75
                IN  A       203.178.136.81

www             IN  CNAME   endo
endo            IN  A       203.178.137.71
                IN  MX  10  endo

localhost       IN  CNAME   localhost.

v6              IN  NS      ns1.v6
                IN  NS      ns2.v6
ns1.v6          IN  A       163.221.11.21
ns2.v6          IN  A       203.178.136.188
```

sample of 203.178.136 (cont.)

```
; $ORIGIN 136.178.203.in-addr.arpa.
@                   IN  SOA  ns.wide.ad.jp. two.wide.ad.jp. (
                             1998100401   ; Serial
                             3600         ; Refresh
                             900          ; Retry
                             3600000      ; Expire
                             3600         ; Minimum
                    )
                    IN  NS   ns.wide.ad.jp.
                    IN  NS   ns.tokyo.wide.ad.jp.

61                  IN PTR   ns.wide.ad.jp.
63                  IN PTR   ns.tokyo.wide.ad.jp.
188                 IN PTR   ns2.v6.wide.ad.jp.
```

レコード定義の基本型

key [ttl] IN r-id value1 value2 ...

<左辺> <右辺>

- ttl (Time To Live) - 省略可
  - 当該レコードのキャッシュ期間
- IN (class-ID) - Internet Domain
- r-id (resource-ID)
  - レコードの種類 (SOA, NS, A, MX, ...)
- value
  - そのレコードの値 (r-id によって形式が違う)

レコード定義の基礎知識

- 同一 key に対する定義の連続
  - 後続の定義の key は省略可
- $ORIGIN <domain>
  - デフォルトのドメイン名の指定
  - 初期デフォルトは named.{boot,conf} で指示されるもの
- $INCLUDE <filename> [<domain>]
  - ファイルの挿入
- FQDN表記のホスト名の末尾には . を

SOA (Start Of Authority) RR

```
@  IN  SOA  <Pri-NS名>  <管理者メールアドレス>  (
        1           ; Serial
        172800   ; Refresh (2d)
        3600       ; Retry
        1728000 ; Expire (20d)
        172800   ; Minimum TTL (2d)
        )
```

- **管理者メールアドレスは @ を . に変える**
  - motonori.wide.ad.jp

SOA パラメータ (cont.)

- Serial
  - Sec-NSのデータ更新判定用
- Refresh (秒)
  - Sec-NSのSerialチェック間隔
- Retry (秒)
  - Refresh経過後のチェック間隔

SOA パラメータ (cont'd)

- Expire (秒)
  - サービス停止までのチェック不能期間
  - サービス停止後に nslookup をすると...

    *** ns.provider.ad.jp can't find x.co.jp.: Server failed
- Minimum TTL (time to live) (秒)
  - ゾーン内に定義される全レコードの
    デフォルト・キャッシュ期間

    (キャッシュする全NSに対して効果を持つ)

Serialについて

- Secondary の Primary との同期のため
  - 内容を変更したら必ず Serial を増加させる
- 32ビット
- . による混乱に注意(使わない方がよい?)
  - 1.01 = 100001 ("." は "000" と同値)
- 1997122501 など日付を使うと明瞭
  - 一日100回更新で4294年まで
- 上限なし(ループ状):RFC1912(I)
  - 1に戻すことが可能
  - 2147483647(7fffffff)以内を2回足す

データの再読み込み

- データ更新後、named に SIGHUP を送る

  # ndc reload

- bind8 以降の場合は、BIND_NOTIFY 機能により、Secondary に更新要求が送られる(Serial が増加している場合)
  - Secondary も bind8 以降が必要

Secondaryでの手動更新

- FORCED_RELOAD機能
  - SIGHUP を受けるとシリアルをチェック
- バックアップファイルを消してから named の再起動
  - named-xfer で転送がおこなわれる

```
# mv mydomain.zone mydomain.zone.bak
# ndc restart
```

NS (Name Server) RR

- Pri-NS および Sec-NS を記述
  - 上位ゾーンでの記述が重要
    - Authorized Server
  - 上位ゾーンに記述がない
    - Unauthorized Server
- 該当する NS に対する A RR も記述
  - glue record (逆引き zone には不要)

```
$ORIGIN     ad.jp.
wide        IN  NS  ns.wide.ad.jp.    ;  ad.jp.zone からの delegation
ns.wide     IN  A   203.178.136.63  ; ← glue record
```

lame (不完全な) NS

- Authorized だと思って問い合わせたら Unauthoritative answer が返ってきた
  - Delegation されているのに
  - Primary/Secondary NS ではない
- 実際の Authorized NS にアクセス不能な状況になると、存在するはずのデータが存在しないとみなされる
  - メールが落ちる

A (Address) RR

- A RR
  - ホスト名からIPアドレスのマッピング

```
$ORIGIN         wide.ad.jp.
sh              IN A 203.178.137.73
```

「ホスト名」に利用できる文字

- アルファベット (A-Z, a-z)
- 数字 (0-9)
- ハイフン (-)
- 注意すべき文字
  - アンダースコア (_)
    - RFC1035(S), RFC1123(S)は許していない
    - 新しい(4.9.4以降の)bindのresolverは、_ を含むホスト名を無視する (res_hnok)
      - メールが落ちる

MX (Mail eXchanger) RR

- MX RR
  - メールアドレスから配信先ホスト名へのマップ

```
$ORIGIN         wide.ad.jp.
@               IN MX 10 sh.wide.ad.jp.
```

- 末尾の . に注意
- MX は A より優先(メールの配信)
- A を優先させたいとき
  - 1st-MX で転送

MXのプレファレンス

- DNS の MX RRに指定するコスト値
- コスト最小
  - Primary MX / Primary Mail Server
  - First MX / First Mail Server
- コスト準最小
  - Secondary MX / Secondary Mail Server
- コスト最小以外
  - Lower MX (優先度が低いという意味)

MX RR の右辺と CNAME

- MX RR の右辺に CNAME の左辺となる名前を書くべきではない
- Lower MX が MX RR の右辺にある自分の名前を認識できないと問題
  - 回避策が講じられていれば動くけど…
  - namedが警告を出す

ワイルドカードMX (cont.)

*.x.co.jp. IN MX 10 mail.x.co.jp.

- Firewall がある場合(直接通信できない)
  - 外: 個々のレコードを外部に見せたくない
    - でもホスト宛のメールアドレスを利用したい
  - 内: 外界をひとつのレコード定義で代表
    - root に Wildcard MX を定義し、GW に集める
- nohost.x.co.jp や host.nosubdom.x.co.jp にマッチ
  - 無駄なメールが飛ぶ

ワイルドカードMX (cont'd)

- specific なレコードが存在すると参照されない

```
ns.x.co.jp.        IN A  12.34.56.78
*.x.co.jp.         IN MX 10 mail.x.co.jp.
ns.x.co.jp.        IN MX 10 mail.x.co.jp. (必要)
```

  - サブドメインが存在する場合も同様

Wildcard MX の弊害

- 存在しないアドレスにもメールが飛ぶ
  - 送信時に存在しないアドレスであることが不明
- 存在しないアドレスに補完される

  user@mail.x.co.jp.x.co.jp
  - 回避するには sendmail.cf で ResolverOptions に HasWildcardMX を定義
- 配信先に対応するMX RRが引けない
  - 配信先ホスト名の最後に必ず . を補う

→ どうしても必要な場合にのみ利用する

CNAME (Canonical NAME) RR

- ホストの別名定義

```
$ORIGIN  wide.ad.jp.
archie       IN CNAME sun3.tokyo.wide.ad.jp.
```

  - 末尾の . に注意
  - CNAME チェインはできるだけ避ける
  - 同一 key に別の種類のレコードを定義しない
  - 同一 key に複数の CNAME は定義しない
- NS, MX の右辺に CNAME で定義される名前を使わない

CNAME のチェイン

- CNAME RR の右辺がさらに別の CNAME RR の左辺

```
alias1          IN  CNAME  alias2
alias2          IN  CNAME  real-name
```

- RFC1034(S)
  - チェインの定義は非推奨(should not)
  - 実装では辿れること(should)
    - sendmail では10回までたどる (MAXCNAMEDEPTH)
    - named は8回までたどる (MAXCNAMES)

メールアドレスと CNAME

- エンベロープのエイリアスは本名に書き換えられるべき(RFC1123(S))
- 多くの(古い)sendmailはヘッダも本名に書き換える
  - どのアドレス宛に届いたのかがわからなくなる
  - 経路上のsendmail.cf の設定次第
- 書き換えられたくないときは、MX か A を
  - IETFはCNAMEによる書き換えをしない方に動いている
  - DontExpandCnames オプション (8.7以降)

メール配信時のDNS検索手順 (cont.)

1. CNAMEを解決
   - CNAME でなくなるまでチェインをたどる
     - 上限あり(無限ループ防止)
2. MXで検索
   - 複数あれば、preference でソート
   - preference が同じ時はさいころを振る
   - ➧ MXが見つかった時、Aも同時にAdditional Information として返される (DNS の仕様)

メール配信時のDNS検索手順 (cont'd)

3. Aで検索
   - MXが得られなかった時
   - 個々のMXに対して
     (Additional Info.でAが得られなかった時)

- A しか定義されていなければ、検索処理は2回必要 (MX と A)
  - ホストにも MX を定義すべき
    - 通信トラフィックの削減

ホストにもMXレコードを

- 障害に備えるため
  - Aレコードだけでは Secondary MXが指定できない
  - 異なるホストに対するIPアドレスを定義した Aレコード(仮想ホスト)
    - 障害対策としては弱い/負荷分散にしかならない
- DNS検索の効率化
  - そのホストしか受信しなくても定義すべき
    - DNS検索が一度で完了する

メールアドレスの補完(cont.)

- MX RR と A RR を用いる
  - ワイルドカードMX問題に注意
- /etc/resolv.conf を参照

domain sub.x.co.jp

  - search sub.x.co.jp x.co.jp co.jp と同値
    - 遡って3階層分調べる (MAXDFLSRCH)
    - 最短は2レベル (LOCALDOMAINPARTS)
      - JP domain の実状にあわない
  - RFC1535(I) では暗黙の遡りを禁止
    - 新しいbindのresolverは遡らない

メールアドレスの補完(cont'd)

search sub1.x.co.jp sub2.x.co.jp x.co.jp

- LOCALDOMAIN 環境変数によるユーザ設定
  - 最大6ドメイン (MAXDNSRCH)
- 検索の順序

    nic.ad.jp

    nic.ad.jp.sub.x.co.jp

    nic.ad.jp.x.co.jp

    nic.ad.jp.co.jp

  - RFC1535(I)より前は nic.ad.jp を最後に検索

PTR (domain name PoinTeR) RR

- IP アドレスからホスト名へのマッピング
  - いわゆる逆引き

```
$ORIGIN      137.178.203.in-addr.arpa.
73           IN PTR sh.wide.ad.jp.
```

  - PTR レコード検索によるサービス制限
    - 引けないホストからのアクセス拒否
    - ドメイン名の確認
- うそつき問題
  - アドレス→ホスト名 の単方向だと騙れる
  - 引き直しでチェック

nslookup で逆引きの確認

- ホストのIPアドレスが 1.2.3.4 のとき

  % nslookup -q=ptr 4.3.2.1.in-addr.arpa.

- 新しい (4.8.3 以降) nslookup では次の指定も可能

  % nslookup 1.2.3.4

ネットワーク名の定義

- RFC1101(?): DNS Encoding of Network Names and Other Types
- netstat -i, -r などで参照される

```
0.0.54.130.in-addr.arpa.    IN PTR kuins.kyoto-u.ac.jp.
                            IN A   255.255.0.0
kuins.kyoto-u.ac.jp.        IN PTR 0.0.54.130.in-addr.arpa.

0.0.0.224.in-addr.arpa.     IN PTR BASE-ADDRESS.MCAST.NET.
```

その他のレコード

- HINFO, TXT, WKS
  - HINFO は必ず2つ以上のパラメータを書く!
- NULL, MB, MG, MR, MINFO (experimental)
  - RFC1035(S)
- AFSDB, ISDN, RP, RT, X25
  - RFC1183(E)
- PX
  - RFC1664(E)

localhost/127.in-addr.arpa zone

- すべてのネームサーバに設定すべき
  - root server まで問い合わせるのは無駄

```
$ORIGIN         my.domain.jp.
localhost       IN CNAME localhost.
```

- 引き直しの際の不整合の防止
  - 127.0.0.1 → localhost.my.domain.jp にならないように

CIDRと逆引き管理

- class less なアドレスの割り当て
  - 192.0.2.0/25 - 組織Aに
  - 192.0.2.128/26 - 組織Bに
- 逆引きゾーンの管理単位問題
  - オクテット(8ビット)単位の権限委譲との不整合
- 解決策
  - CNAME で散らす
    - RFC2317(BCP)
      - Classless IN-ADDR.ARPA delegation
  - NS で散らす

Classless IN-ADDR.ARPA delegation (cont.)

- **上位ゾーンからの権限委譲**

```
$ORIGIN 2.0.192.in-addr.arpa.
;  <<0-127>> /25
0/25  NS  ns.A.domain.jp.
1       IN  CNAME  1.0/25.2.0.192.in-addr.arpa.
2       IN  CNAME  2.0/25.2.0.192.in-addr.arpa.
 :
126    IN  CNAME  126.0/25.2.0.192.in-addr.arpa.
```

Classless IN-ADDR.ARPA delegation (cont'd)

- **当該ゾーンでの定義**

```
$ORIGIN 0/25.2.0.192.in-addr.arpa.
@       IN  SOA ...
        IN  NS  ns.A.domain.jp.
1       IN  PTR  host1.A.domain.jp.
2       IN  PTR  host2.A.domain.jp.
 :
126     IN  PTR  host126.A.domain.jp.
```

Classless IN-ADDR.ARPA delegation (cont'd)

- すなわち…

1.2.0.192.in-addr.arpa.

↓ CNAME

1.0/25.2.0.192.in-addr.arpa.

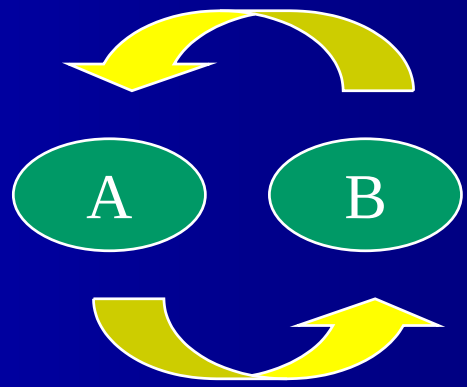
↓ PTR

host1.A.domain.jp.

古いglueレコードが消えない?

- アドレスのつけかえ時の問題
- bind4.8.3以前?
- server A: primary of x.co.jp
- server B: primary of sub.x.co.jp
  - お互いにセカンダリになっている
- x.co.jp の NS (server C)のアドレスを変更
- server C の古い glue レコードが消えない
  - server A で消しても
  - server B のからの zone transfer で甦る

  → セカンダリコピーからも消す



サーバが報告するエラー (cont.)

- bad referral
  - NS があるのに SOA がない
- NS points to a CNAME
- MX points to a CNAME
- dangling CNAME pointer
  - CNAME の先が何も指していない
- Lame server on 'x.co.jp'
  - Authorized サーバのはずなのに、Unauthoritative answer が返ってきた

サーバが報告するエラー (cont'd)

- Response from unexpected source
  - 違うインターフェースアドレスからの応答?
  - アタック?
- zone "xxx" (class 1) SOA serial# (nn) is < ours (mm)
  - SOA serial が減った!

RFC1912(I): Common DNS Operational and Configuration Errors

DNSの今後

- Dynamic Update
  - レコード単位のデータ更新
- Incremental Zone Transfer (IXFR)
  - トラフィックの削減と更新速度の向上
- Security Extention
  - SIG RR, NXT RR

メールシステムのデザイン

Contents

- ドメインマスタ
- NULL クライアント
- PPP クライアント
- バックアップメールサーバ
- Firewall とメールサーバ
- バーチャルホスト

ドメインマスタ

- user@mail.x.co.jp の他に user@x.co.jp も受理する。
- メールの発信時に user@x.co.jp を発信者アドレスにする。

- 計算機を特定すべきメール(rootなどからのメール)は、user@mail.x.co.jp が望ましい

ドメインマスタの設定

ACCEPT_ADDRS='x.co.jp'

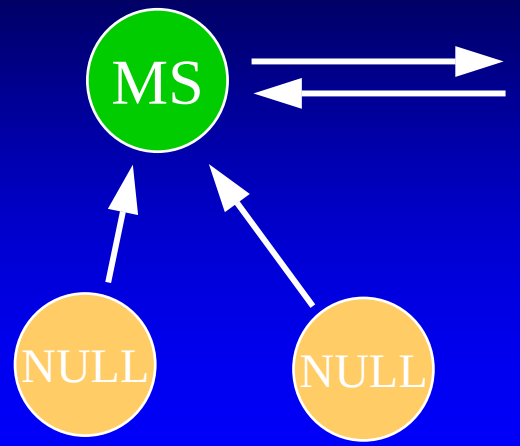
- 受理すべきアドレス部 (これがポイント!)

FROM_ADDRESS='x.co.jp'

- 送信時のデフォルトのドメイン部
- 管理用アドレスにはホスト名が付与される
  - root, daemon, postmaster,...

- 複数ドメイン宛を受理
  - ACCEPT_ADDRS='sub1.co.jp sub2.x.co.jp'

NULL Client



- スプールを持たない
- 全てのメールをメールサーバへ
  - メールサーバのアドレスの定義のみが必要

CF_TYPE=R8V7-null

SPOOL_HOST=mail.x.co.jp

- メールサーバにだけ届くアドレスを記述
  - [] で囲む (lower MX があるときに A RR を参照)
  - [IP アドレス]を記述

PPPクライアント

- ダイアルアップ環境なので常に接続されているわけではない
- IPアドレスは接続の際に決まる
- DNS的ホスト名も固定的ではない
- 内部的ホスト名が表に出るのはまずい
  - メールの返事が返らない
- 発信者アドレスはプロバイダのメールサーバのものを使用
- 受信はPOP (popclient など)

PPPクライアントの設定

プロバイダのメールサーバが発信用に使えるとき

DIRECT_DELIVER_DOMAINS=none

DEFAULT_RELAY=mail.provider.ne.jp

FROM_ADDRESS=po.provider.ne.jp

CON_EXP=True

SMTP_MAILER_FLAG_ADD=e

すぐに発信しない工夫

- 必ず一旦 mqueue にためる
- 接続時に sendmail -q でまとめて配信
- デーモンsendmailは-bdのみで起動(必要なら)

高度なPPPクライアント

- 発信者アドレスの書き換え
  - ローカルのユーザ名と契約ユーザ名の変換
  - userdb, usertable の利用
- 契約していないアドレスからの発信の抑制
  - check_compat ルールセットの利用
- 自動ダイアルアップ時のタイムアウト

  O DialDelay=15s

バックアップメールサーバ(cont.)

- 1st-MXの障害発生時に代わりに受信
  - 2nd-MXとして動作
- 可能なら1st-MX と独立に直接配信
  - 1st-MX と aliases ファイルを共有
  - 同じアドレスを受理
    - 全てのユーザ

      ACCEPT_ADDRS=
    - 特定のユーザ

      SECONDARY_*=

      – 指定したもの以外は、1st-MX の回復を待つ

バックアップメールサーバ (cont'd)

- aliases を共有
  - NIS などの共有手段の利用
  - rdist して newaliases する方法もある
  - ローカル aliases と共有用 aliases に分離
    - R8 sendmail は複数ファイルの aliases が扱える
    - OA/etc/aliases, nis: mail.aliases
- ML 配送のバックアップでの問題
  - アーカイブをどうするか
  - 連番をどうするか

Firewallとネームサーバ (cont.)

- 組織外向けネームサーバ
  - Wildcard MX を定義する場合

    $ORIGIN x.co.jp.

    * IN MX 10 ext-mail.x.co.jp.

    - 内部のホストの存在を見せない
  - 存在するメールアドレスをすべて定義
    - Wildcard MX を使わない方法
    - エラーになるべきメールがGWまでやってこない

Firewallとネームサーバ (cont'd)

- 組織内向けネームサーバ
  - 組織外のアドレスを内部に見せない方法
    - あらゆるサービスはすべてproxy経由
    - DNSタイムアウトを避けるため
      組織内にrootサーバを設置
  - 組織外のアドレスをforwardersで拾ってくる方法
    - 内部から外部に直接接続できる場合
    - socks

Firewallとメールサーバ(1)

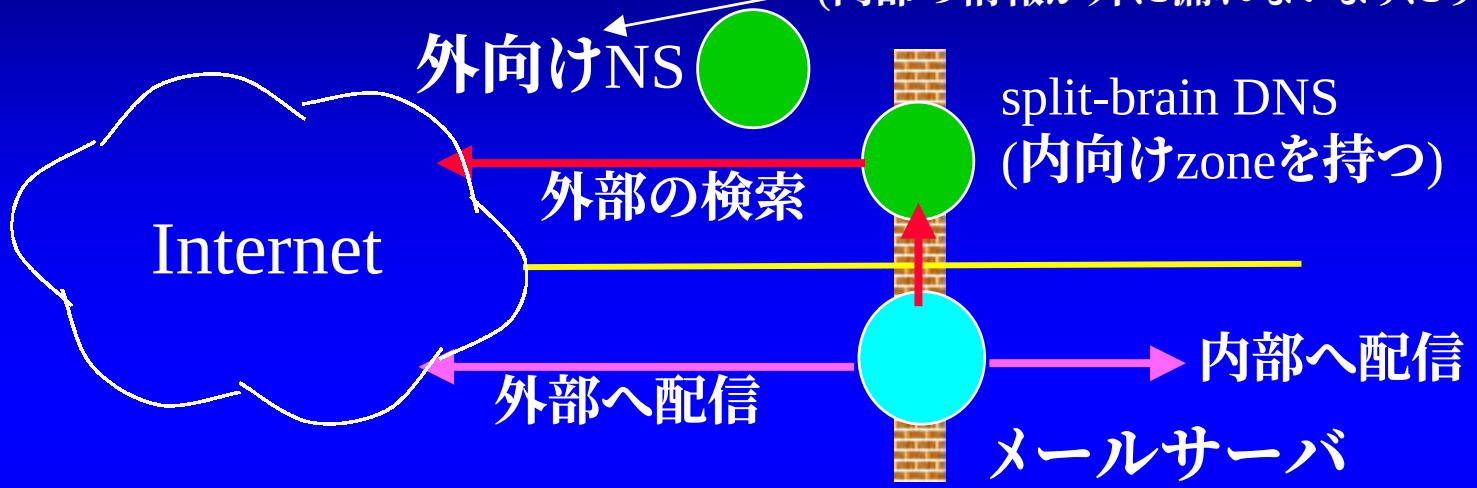
ネームサーバとメールサーバの構成

- メールサーバを1台でまかなう
  a. 内向けzoneを持った外部検索用ネームサーバを参照する方法
     - split-brain DNS
  b. 内部は固定ルールで配信する方法

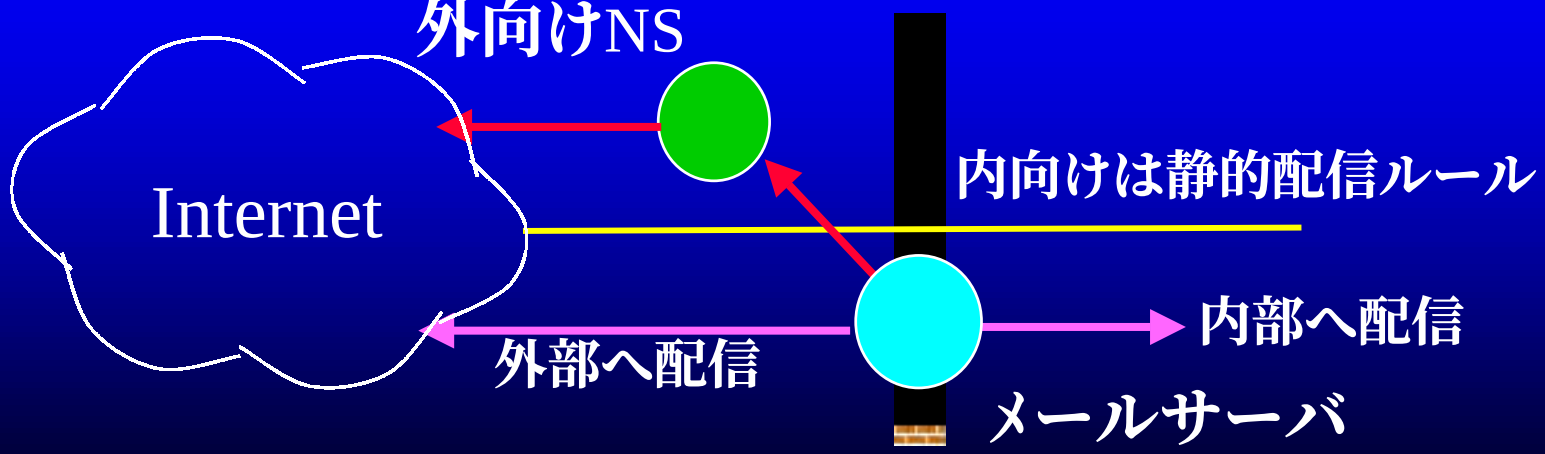
メールサーバ を1台で

a

(内部の情報が外に漏れないようにするため)



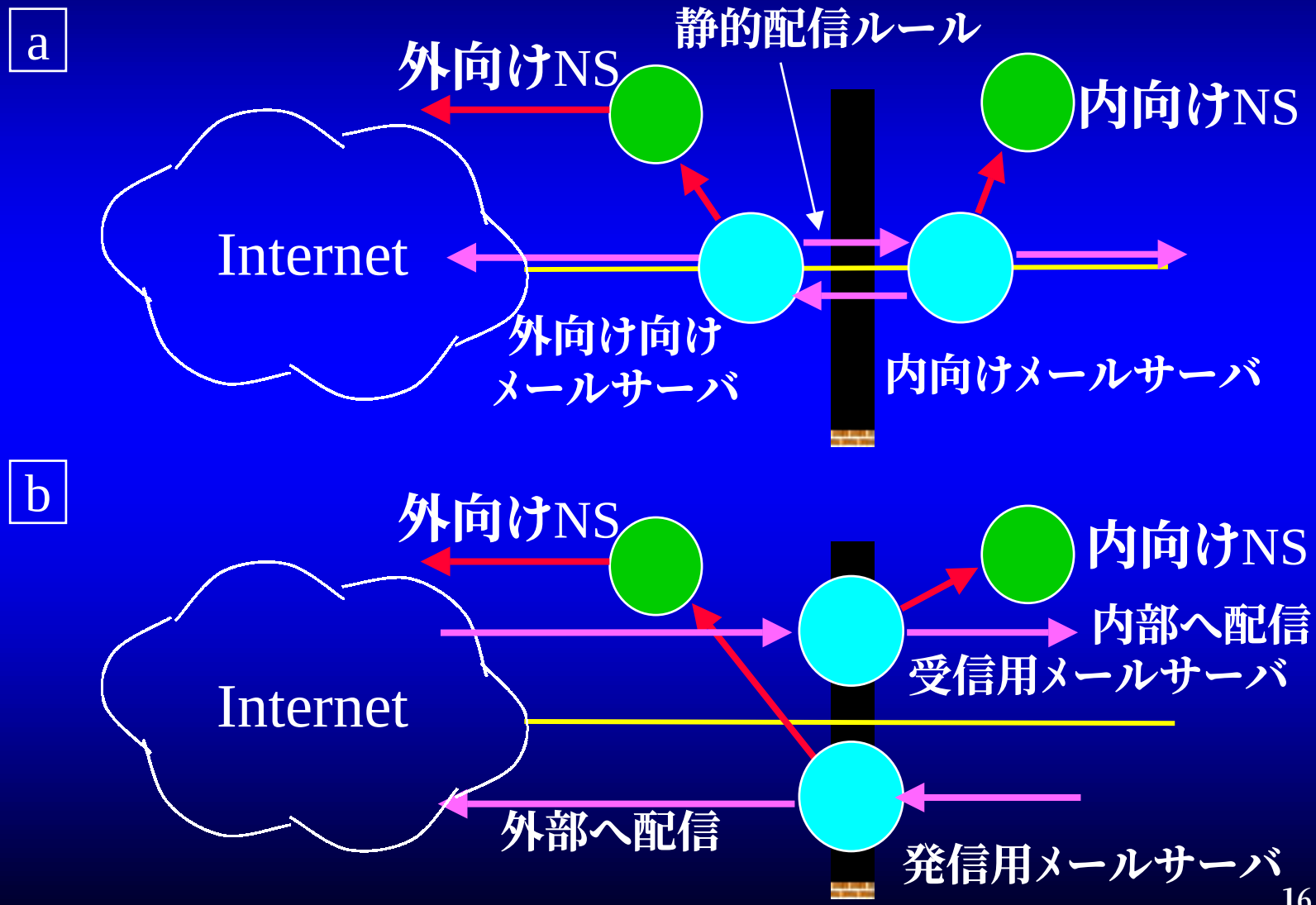
b



Firewallとメールサーバ(2)

- メールサーバを2台
  - 外部DNSを参照するメールサーバ
  - 内部DNSを参照するメールサーバ
    - 方式 a
      - 両者間は静的経路設定
    - 方式 b
      - 受信専用メールサーバ
      - 発信専用メールサーバ

メールサーバを2台で


a
静的配信ルール
外向けNS
内向けNS
Internet
外向け向け
メールサーバ
内向けメールサーバ
b
外向けNS
内向けNS
内部へ配信
受信用メールサーバ
Internet
外部へ配信
発信用メールサーバ

内部向けメールサーバの設定

- 外部への静的配信ルール

DIRECT_DELIVER_DOMAINS=x.co.jp

DEFAULT_RELAY=external.x.co.jp

  - 外部向けメールサーバ

外部向けメールサーバの設定

- 内部への静的配信ルール

STATIC_ROUTE_FILE=x.static

x.staticの内容:

```
GW    [12.34.56.78]
#         (internal.x.co.jp)
DOM  x.co.jp
```

– メールサーバ宛てのメールは受理可能

ネームサーバを1台で(1)

- NS, MS とも1台でなんとかするには...
  - a. NSで内側に first MX を向けておく

```
inner-host  IN  MX  10  inner-host
            IN  MX  20  gw
```

    - 外部からは 1st-MX と直接通信できず、タイムアウトが発生
      - 送信側のストレスになるのでよくない
  - b. GWで内側へは A RR を参照して配信

```
inner-host  IN  A  12.34.56.78
            IN  MX  10  gw
```

ネームサーバを1台で(2)

c. 内側を別の枝にマップ

- inner.domain.jp → inner.domain.jp.local
  - sendmail.cf でアドレス変換をおこなう
    - STATIC_ROUTE_FILE の MAP 行 (CF)

d. 1台に複数のデーモンを起動する

- 外側/内側のIP アドレスにバインドさせる
  - 外向け named と 内向け named
    - listen-on, query-source, transfer-source (bind8.1.2)
  - 外向け sendmail と内向け sendmail
    - O DaemonPortOptions=Address=12.34.56.78
- いわゆる virtual host の設定

ネームサーバを1台で(3)

- a, b 方式の問題
  - 内外の直接の通信は不可なのに
  - 外から内部ホストの情報が見える
    - bind8 の allow-query だけでは役不足

ネームサーバを1台で(4)

- b 方式の具体的設定
  - ゲートウェイから内部への配信
    - 静的経路定義
    - A RR を見る
      - 1st-MX が自分だった場合の挙動の変更

      TRY_NULL_MX_LIST=True (CF)

      O TryNullMXList=True (sendmail.cf)
  - 正しく設定できていないと

    local configuration error になる

ゲートウェイ内クライアント

- なんでもGWへ

  DIRECT_DELIVER_DOMAINS=none

  DEFAULT_RELAY=internal.x.co.jp

- 内部については直接配送

  DIRECT_DELIVER_DOMAINS=x.co.jp

  DEFAULT_RELAY=internal.x.co.jp

  - qmail の場合は、control/smtproutes に定義

バーチャル・ホスト(cont.)

- 1台のホストで複数アドレスを利用

a) ユーザ空間の共有

```
USERTABLE_MAPS='domain1=hash:/etc/map1 ¥
                domain2=hash:/etc/map2'
```

b) ユーザ空間の分離(1)

- 1つのホストに複数のIPアドレス
- アドレスごとにsendmailをバインド
  - O DaemonPortOptions=Address=1.2.3.4
- chroot で環境も分離すると良い

バーチャル・ホスト(cont'd)

c) ユーザ空間の分離(2)

- sendmail.cf でがんばる
  - アドレスごとに local mailer を切り替え
- /etc/passwd とは別のデータベースを利用
- POP 等専用サービス
  - ドメイン名を含んだ識別子でユーザ認証

システムデザインのまとめ

- どのアドレスを受理するか
- アドレスごとの配信方法の選択
  - 配信先を静的に定義
  - ネームサーバ (MX) を参照

これらをしっかり整理する

付録:意外と知られていない(?) sendmail テクニック

- 送信待ちメールをひとつのホストに集める
- 巨大なメッセージを拒否する
- 発信者アドレスによって処理を変える
- SMTPに失敗したらUUCPで送る
- メーラの処理順序の指定
- 個人レベルのML設定

送信待ちメールを ひとつのホストに集めるには

- FallBackMX オプションを利用する
  - DNSが引けなかった場合
  - すべてのMXにメールを送ることができなかった場合

  に指定したホストにメールを転送する
- mqueueの管理等が楽になる
  - 最長保存期間の調整
- 経路のトラブルに気がつきやすくなる

巨大なメッセージを拒否する

- MaxMessageSize オプションを利用する場合
  - 受信時にサイズを超過したメッセージを拒否
  - ESMTP の場合は、MAIL FROMの時点でサイズが通知されるので、そこで拒否
- メーラ定義に M= を指定する場合
  - 一旦メッセージを受信してしまう
  - 送信直前にサイズをチェックする
    - メーラごとに許容サイズが異なる可能性があるため

発信者アドレスによって処理を変える

- エラーメールやSPAMの分類をsendmail.cfレベルで実現

```
CT root news postmaster MAILER-DAEMON uucp cron
S0
:                                                    エンベロープ発信者
R $*                          $: $1 $| $>3 $&f
R motonori $| <@>             $: trash          <> のとき
R motonori $| $=T<@$*>        $: trash          クラスTで検査
R $* $| $*                    $: $1
:
```

SMTPに失敗したらUUCPで送る (3.1Wpatch)

- 複数のメーラを順に起動する機能を利用

```
S0
 :
R $*<@x.co.jp>$*      $# smtp $@ x.co.jp $: $1<@x.co.jp>$2
(スペース)            $# uucp $@ uucp-x $: $1<@x.co.jp>$2
 :
```

メーラの処理順序の指定 (3.1Wpatch)

- 大規模MLなどで、SMTPより前にlocalへの配信（自分あて、アーカイブ）を先に処理させたい場合
  - メーラ定義に %= で優先度（コスト）を指定

Mlocal …, %=0

Msmtp …, %=10 (local mailer の後に処理される)

個人レベルでのML設定

- CF の localdeliver 機能を利用
  - user@host だけでなく、user+*opt*@host も利用可能
  - さらに、*opt*@user.host もサポート
  - ユーザは .forward のほかに .forward+*opt* や .forward+default が利用可能
  - したがって、.forward+ML を設定することで、ML@user.host を運用可能
  - 参考：Samples/virt-domain+.def

sendmailの重要なオプション(1)

- EightBitMode=pass8
  - MIMEでない8bitデータをそのまま通過させる
- SendMimeErrors
  - RFC1894 - DSN (Delivery Status Notification) に従ったエラー通知を返すかどうか
- ConnectionCacheSize
  - 接続したままにするSMTPコネクションの数
  - run queue の時に役立つ

sendmailの重要なオプション(2)

- PostMasterCopy
  - エラーの発生したメールのヘッダのみをpostmasterにも送る
  - 致命的なトラブルを未然に防ぐため
- DoubleBounceAddress
  - エラーメールを発信者に送り返せなかった場合の送り先
  - 本文も届いてしまうことに注意

sendmailの重要なオプション(3)

- MeToo
  - alias展開したときに発信者が含まれていた場合、送り返すかどうか
- PrivacyOption
  - SMTPのEXPN, VRFY等の禁止
- QueueSortOrder
  - mqueueにたくさんたまったメールの採草順序
  - timeにするとうれしいかも(届いた順)

sendmailの重要なオプション(4)

- MinQueueAge
  - mqueueにたまったメールの最短再送間隔
  - 手動再送時に無視したい場合は -qI などで
- ConnectionRateThrottle
  - 許容する一秒あたりの受信コネクション数
  - デフォルトは0（無制限）
  - DoSアタックに有効かも

sendmailの重要なオプション(5)

- DontBlameSendmail
  - 8.7後半からは、ホスト内セキュリティの向上のため、ファイルのモードなどにシビアになった
  - チェックを緩和したいときに指定する
  - 詳しくはhttp://www.jpcert.or.jp/tech/98-0001/を
- DontProbeInterface
  - 起動時にインタフェースアドレスの逆引を禁止

sendmailの重要なオプション(6)

- Timeout.queuereturn.*
  - 送信先への到達不能などの原因によるメールの送信の再試行の最大期限
  - この期限を過ぎると、エラーとして発信者に返送される
  - 長期休暇などを考慮して期間を決めること
- Timeout.queuewarn.*
  - メールの送信が再試行状態にあることを通知するまでの時間
  - 不要な場合は0にする

メールの配信制限
～ SPAM対策にむけて ～

内容

- 不正中継（踏み台利用）の防止
- SPAMの排除

従来のメールシステムの問題

- メールを送り届けることが第1目標だった
- 中継やソースルーティングにも寛大
  - ネットワークの相互接続のため
    - ゲートウェイを明示的に指定

      user%domain@gateway

      @gateway:user@domain
  - 不安定・不完全な経路制御への補助
    - 適当な途中のホストをゲートウェイに指定
  - 細い回線の補助
    - 他組織によるバックアップMX

着信・配信制限への要求(1)

- 意図しないメールの受信
  - メール爆撃 (Mail Bombing)
  - 宣伝メール (Spam)
    - Unsolicited Commercial Email (UCE)
- 意図しないメールの中継
  - 組織外から組織外へ(踏み台)
    - Third-Party Mail Relay

→ CPU、ネットワーク、ディスクの圧迫

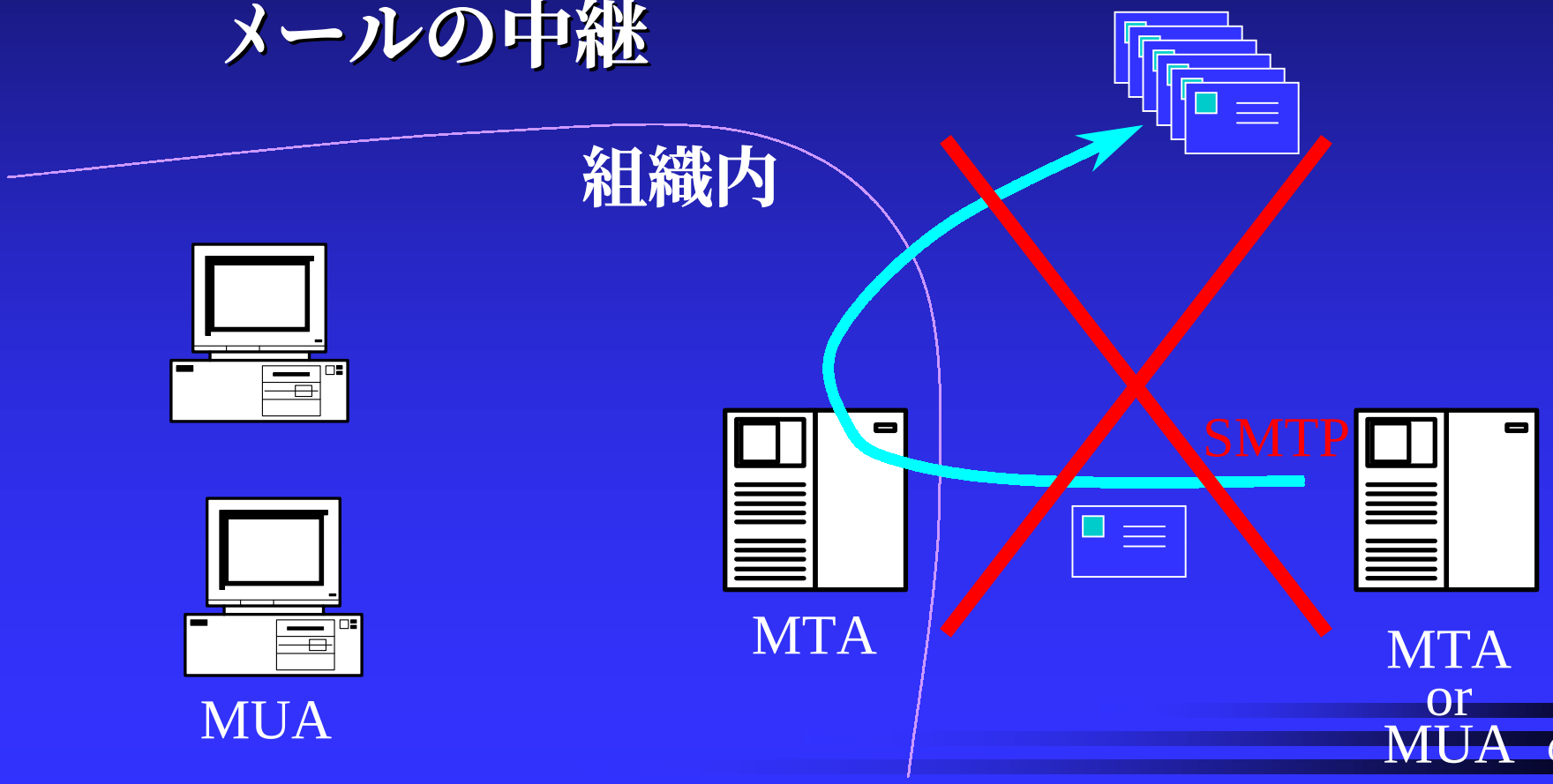
→ 時間の浪費 (転送時間、メールの整理)

着信・配信制限への要求(2)

- ポリシーによる通信先制限
  - 組織内のみ
    - 学生実験、アルバイト、出向社員
- メールボックスのサイズ問題
  - SPAMで溢れ、重要なメールが受け取れない
    - 一種の Dos アタック
  - ディスク管理の問題
    - POPで「サーバに残す」、IMAP
    - メールを読まない人たち
      - expire 処理

不正中継（踏み台利用）の拒否

- 自組織に関係のないメールの中継



SPAMの踏み台の役割

- たくさんの宛先をもったメール1通から宛先の数だけのメールを複製する
  - 踏み台ホストへの影響
    - 発信者の回線費用、CPU資源などをかたがわり
    - SPAM発信の嫌疑、苦情対応の手間
    - 組織に対するマイナスイメージ
    - ブラックリストに載り、メールが届かなくなる
  - 対策
    - 明示的な転送（ソースルーティング）を拒否
    - 通常不必要な組織外 → 組織外 の転送を拒否

配信制限のための判断材料

- SMTP接続元/先のホストの－
  - IPアドレス
  - ドメイン名　（DNSの逆引きによる）
- エンベロープの－
  - 発信者のメールアドレス
  - 受信者のメールアドレス
- ヘッダの内容
- 本文の内容

ヘッダとエンベロープ

* 封書に似ている

- ヘッダ(header)のアドレス
  - 内封された書面の送り主/宛て先
  - 配信処理中は基本的に書き換えなし
- エンベロープ(envelope)のアドレス
  - 封書の表書きの送り主/宛先
  - 配信処理中に状況に応じて書き換え
  - 着信後は UNIX From や Return-Path: に残る
    - 単に残るだけであることに注意

ヘッダとエンベロープの関係

- 一緒のとき
  - ユーザからの発信（個人宛て、投函直後）
    - ユーザはヘッダのみを指定
    - エンベロープはヘッダからコピーされる
- 異なるとき
  - メーリングリストからの配信
    - エンベロープの発信者・受信者が書き換わる
  - フォワード処理後・エイリアス処理後
    - エンベロープの受信者が(受信者も?)書き換わる

SMTPでのヘッダとエンベロープ

```
HELO mx1.s.domain
250 post.r.domain Hello mx1.s.domain
MAIL FROM:<sender@s.domain>
250 sender ok
RCPT TO:<recipient@r.domain>
250 recipient ok
DATA
354 Enter mail, end with "." on a line by itself
From: announce@s.domain
To: list @s.domain
Subject: Newsletter
      ← 空行 (空白もなし)
チュートリアルのお知らせ
[後略]
.
250 Message accepted for delivery
```

エンベロープ発信者

エンベロープ受信者

ヘッダ

本文

この部分が
受信者に届く

SMTPのどの時点で制限するか

- SMTP コネクションの接続時
- SMTP セッション中の応答
  - HELO/EHLO ホスト名
  - MAIL FROM: <エンベロープ発信者アドレス>
  - RCPT TO: <エンベロープ受信者アドレス>
  - DATA (ヘッダ、本文の内容)
- SMTP 受信完了後・配信前
  - è 負荷を抑えるためにもSMTPの時点で拒否したい
    - エラーメールの生成は送信側の仕事になる

制限すべきパターンは何か

- 組織外から組織外へ
  - 「From:組織内」は許す?
- 「From:組織内」が組織外からは?
- 「From:組織外」が組織内からは?

転送制限とメーリングリスト、転送設定

- 組織外MLに参加
  組織外アカウントからの転送
  - FROMが組織内のものが、組織外からくる
    - → 宛先が組織内なら許可？
- 組織内MLに組織外の参加者
  組織外への転送
  - FROMが組織外のものが、組織内からくる
    - → 組織内のホストからの接続は許可
    - → 発信者の書き換えを義務づけ・転送の禁止？
  - エンベロープ発信者を書き換えるMLはOK

中継形態の分類

- 組織内のホストからの（組織内は信頼できる？）
  - FROM 内 TO 外　　OK
  - FROM 外 TO 外　　OK?　(きっと内部ML)
  - FROM 内 TO 内　　OK?　(直接送ってほしいかも)
  - FROM 外 TO 内　　OK?
- 組織外のホストからの
  - FROM 内 TO 外　　NG?　(ISP利用)
  - FROM 外 TO 外　　NG　(不正中継)
  - FROM 内 TO 内　　OK　(外部ML, NGにしたいかも)
  - FROM 外 TO 内　　OK

ここに注意!

中継判定の一般化

- 3つのパラメータを持つ判定関数

$$f(h, s, r) = OK, NG$$

h - SMTP接続元ホストのアドレス
s - SMTPエンベロープ発信者
r - SMTPエンベロープ受信者

組織外からの「From内」許可の是非

- ISPにダイアルアップして、
  Fromが自組織のメールを出したい
- アドレスを騙られることで、踏み台になる
  - ISPのメールサーバによっては、内部からは、付与したアドレスをFromに持つメールしかだせない
- エンベロープ発信者を書き換えないMLの存在
  - 戻ってきたメールが拒否される

組織外からの「To 外」について

- 他組織のバックアップ MX を引き受けている場合に注意！
  - プライマリ MX が落ちたときに問題が発覚する
  - http://www.wide.ad.jp/~motonori/mtachecker.html などで確認を

実際の設定

- sendmail
- qmail

sendmail では

- sendmail 8.8/8.9 から柔軟な拒否設定が可能
  - sendmail.cf から拒否機能を呼び出す必要あり
    - オリジナル sendmail 付属 m4 版生成ツール
    - CF-3.6W 以降
  - sendmail 8.9 用の sendmail.cf にはデフォルトで拒否設定が組み込まれる (m4, CF-3.7W)
- 設定は従来の sendmail.cf に単純に追加可能
  - CF-3.7Wでは追加部分のみの生成も可能

sendmail 8.8 でできること

- 接続元ホストの選別
  - check_relay
- エンベロープ発信者アドレスのチェック
  - check_mail
- エンベロープ受信者アドレスのチェック
  - check_rcpt
- 受信後のアドレスチェック
  - check_compat

sendmail 8.9 で可能になったこと

- ヘッダの内容をチェックして拒否
  - 複数ヘッダの組み合わせによる判定はできない…
- 正規表現によるパターンマッチ機能
  - 存在し得ないアドレスをまとめて表現
- DNSで「存在しない」と「引けなかった」の区別
  - すぐエラーにするか、しばらくキューに保存するか
- バックアップ MX 指定の自動認識
  - 勝手に MX に指定されてしまう可能性...

check_rcpt でのチェックの手順

- ${client_addr}（接続元ホストのIPアドレス）が組織内
  - OK（組織内を信頼）
- ${client_addr} が組織外
  - 自ホスト宛て
    - OK
  - 宛先が組織内
    - OK
  - 宛先が組織外
    - NG（SMTPの際にエラーを返す）

CFを利用する場合のパラメータ (cont.)

- LOCAL_HOST_＊
  - f (src_host,＊,＊) = OK/NG
- CLIENT_＊
  - f (src_host, from_domain,＊) = OK/NG
- ROAM_＊
  - f (src_host, from_user,＊) = OK/NG

CFを利用する場合のパラメータ (cont'd)

- ALLOW_RELAY_FROM
  - f (*, from_domain,＊) = OK/NG
    - from_domainで偽装されると
      どこからでもメールをリレーしてしまうので危険
- ALLOW_RELAY_TO
  - f (＊,＊, to_domain) = OK/NG
    - lower MX を引き受けている場合は
      定義を忘れずに

ネットワークアドレスの一部へのクラスマッチング(3.1W機能)

- Sendmail では文字列によるパターンマッチングで判定を行う
- IPアドレスマッチングは、オクテット単位の判定
- CIDR時代には、もっと細かな判定が必要
- 8.9.1+3.1Wパッチではネットマスク表記が利用可能

  C{Network} 200.3.4.64/27

  C{Network} _MASKED_ADDRESS_MATCH_
  - この場合、200.3.4.64 - 200.3.4.91 がマッチする
- マップの場合はmaskedaddr mapを利用

qmail では(cont.)

- デフォルトで中継を拒否
- 中継を許可させるには
  - /var/qmail/control/rcphosts に転送を許すアドレスを追加
    - Lower MX を提供しているアドレスも忘れずに

qmailでは(cont'd)

- ローカルクライアントからのメールの転送
  - qmail-smtpd の起動時に RELAYCLIENT を設定する
    - tcp_wrapper や tcpserver (ucspi-tcp) から起動
  - tcpserver を利用すれば、クライアントのアドレスを見てリレーの可否を制御することも可能
    - IDENT情報も利用可能
    - tcprulesコマンドでルールを即時反映

設定後の動作確認

- 組織外のアカウントを利用
- テスト用 WWW ページを利用
  - http://maps.vix.com/tsi/ar-test.html
  - http://www.wide.ad.jp/~motonori/mtachecker.html

- 正規のメールを拒否させないように注意！
  - 設定ミスをしているホストへの連絡は困難

メールの発信者の認証

- 組織外からのメールの発信の認証
  - POP認証を併用
    1) POPサーバにアクセス
    2) アクセス元のアドレスをデータベースに登録
    3) SMTPの際にデータベースを検索
    4) データベースに載っている場合はアクセスを許可
    5) 一定時間後にデータベースから削除
  - sendmailではmakemapでDBを作るとsendmailの再起動が不要

SPAMとは

- Hormel Foods Corporation の食品の名前
- 意図しないメールの受信
  - メール爆撃 (Mail Bombing)
  - 宣伝メール (Spam)
    - Unsolicited Commercial Email (UCE)
    - Unsolicited Bulk Email (UBE)
    - 受信者の興味等に関係なく無差別に送られてくる
    - 不法なものも
  - 宣伝コストが非常に安い!

SPAMの予防

- なぜメールアドレスが知られるのか？
  - NetNews への投稿
  - メーリングリストの参加者リスト
  - Web に記載されたアドレス （情報収集ロボット）
- SPAM 予防策
  - `user@domain.nospam` といったアドレスの細工
    - 機械的自動返信機能が利用できない
    - 返信の際に手間がかかる
    - 初心者にわかりづらい
  - SPAM リストからの削除要求の返送をしない？

SPAMはどこからくるか

- 不正中継を許すホストから
- 中継されず直接に
  - 不正中継ではないので、完璧な対策が困難

SPAMのフィルタリング

- 判断材料
  - 発信者のメールアドレス(常連アドレス)
  - 発信者のドメイン名
  - 発信ホスト・ネットワークのアドレス
    - DNS逆引き設定の有無？
  - ヘッダの内容
    - SPAM特有の特徴をとらえる
  - 本文の内容？

フィルタリングの問題点

- 定義したホストからのメールが届かなくなる
  - ✓ 完全に拒否せず、マークをつけて受信者側で分類する方法もある
  - ✓ リレーでない(その組織からの)メールを制限しない工夫
- 未知の踏み台が利用されると効果が下がる
- ブラックリストのメンテナンスが面倒
  - → MAPS RBL などの参照

メーリングリストとSPAM対策

- 最近のメーリングリストサーバの標準的(?)な機能
  - 登録メンバ以外からのメールを拒否
    - 投稿、アーカイブ検索と入手
    - メンバリストの入手
  - ヘッダの宛先にMLのアドレスがなければ拒否
- MTAとは独立したサーバプログラムで対応
  - プログラムの機能整備が進んでいる？

SPAMホストのデータベース

- MAPS RBL
- ORBS
- DUL

MAPS RBL

- Mail Abuse Protection System Realtime Blackhole List
  - http://maps.vix.com/rbl/
- DNS で 4.3.2.1.rbl.maps.vix.com に対する A レコードが存在すれば拒否する
  - IP アドレス 1.2.3.4 からの接続要求の場合
  - DNS 参照のテスト用アドレス：127.0.0.2

    2.0.0.127.rbl.maps.vix.com
- BGPもやっている

ORBS

- Open Relay Blocking System
- http://www.dorkslayers.com/orbs/
- リストの参照方法は、MAPS RBL と同様
  - 4.3.2.1.orbs.dorkslayers.com に対する A レコードの存在を確認

DUL

- ORCA Dial-up User List
  - ダイアルアップユーザからの直接のメールを拒否するためのもの
- http://www.orac.bc.ca/dul/
- リストの参照方法は、MAPS RBL と同様
  - 4.3.2.1.dul.orac.bc.ca に対する A レコードの存在を確認
  - ゾーン転送も可能

常連spammerをローカルに登録

- sendmail (CF)
  - SPAM_LIST*
- qmail
  - control/badmailfrom

もちろん手動でメンテナンス

発信者アドレスに関するチェック
～ ごみメールをへらすために ～

- 返信無用のSPAMは、発信者アドレスの不正が多い
  - @domain が省略されているもの
  - user@host 形式の(FQDNでない) もの
  - DNSに登録されていない (返送不能な) もの
    - たまたま引けないものは受信エラーにしない (一時的拒否)
  - 偽造がみえみえなもの
    - ユーザ部が数字だけ、長すぎるもの
      - 正規表現でチェック
- でも、アドレスが騙られると見分けられない

正規表現による拒否 (sendmail 8.9以降)

```
Kcheckaddress regex -a@MATCH
  ^([0-9]+<@(aol|msn)¥.com|[0-9][^<]*
  <@juno¥.com|.{10}[^<]+<@aol¥.com)¥.?>

R $+          $: $(checkaddress $1 $)
R @MATCH      $#error $: "553 Header error"
```

ヘッダの内容による拒否 (8.9以降)

```
HTo: $>CheckTo
SCheckTo
R friend@$*    $#error $: "553 Header error"
```

```
HMessage-Id: $>CheckMessageId
SCheckMessageId
R<$+@$+>$@OK
R$*            $#error $: "553 Header error"
```

- SpamCan という実装もある

  http://consult.ml.org/~timb/spamcan/

苦情窓口の開設

- abuse@domain の作成
  - (RFC2142 Mailbox Names for Common Services, Roles and Functions)
- domain@abuse.net
  - Network Abuse Clearinghouse (http://www.abuse.net)

さらなるSPAM対策

- エンベロープ発信者が <> のものをどうするか
  - Mailer_daemon からのメール
- 発信者を実在のアドレスに偽装されたら...
- 最終的には個人レベルで対処するしかない？
  - 自分のアドレスがヘッダにあるものを通す
  - 参加しているメーリングリストだけ通す
    - procmail などの利用

さらなる技術的対策

- メールの内容をチェックするMTAも
  - ファイアウォル製品など
- 電子署名を普及させる
  - 署名の無い物は捨てる
- MTA間認証？