DNS&Mail

トランス・コスモス（株）技術本部
安藤一憲
ando@rnd.trans-cosmos.co.jp

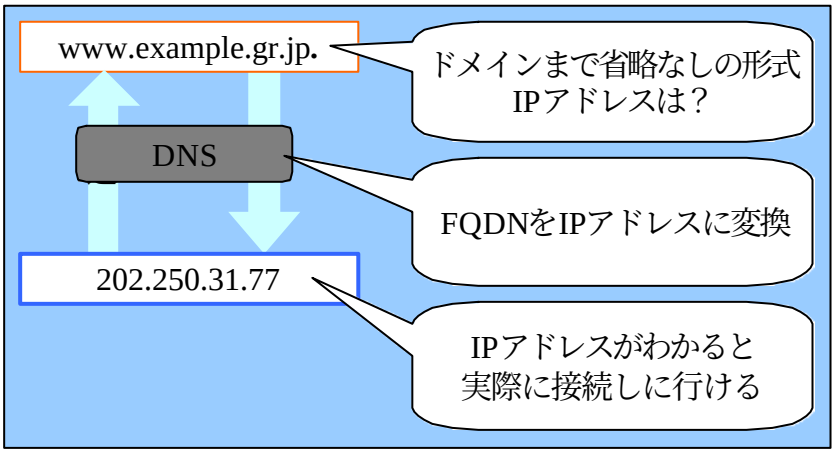
このチュートリアルの構成

- DNSのしくみ
  - 階層構造を持った分散データベース
- namedの設定
  - bind-8.2.2における設定と新機能
- メール配送のモデル
  - MX配送、static配送の使い分け
- メールの抱える問題
  - SPAM対策、ウイルス、チェーンメール

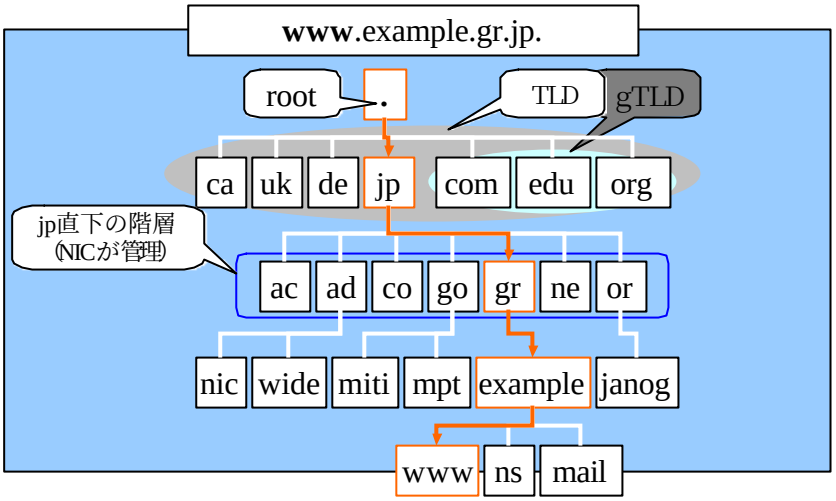
ＤＮＳの重要性

- あるドメインのリソース情報へ到達する鍵
  - 順次ＮＳを手繰ってデータを引きに来る仕組み
    - 手繰れないと破綻
  - ＩＰアドレス付け替え時、ドメイン変更時に注意
  - ほぼ全てのサービスに影響
    - FQDNを用いるもの全て
  - ＮＳを死守すべし
    - 全ての基礎となるサービスのひとつという位置付け

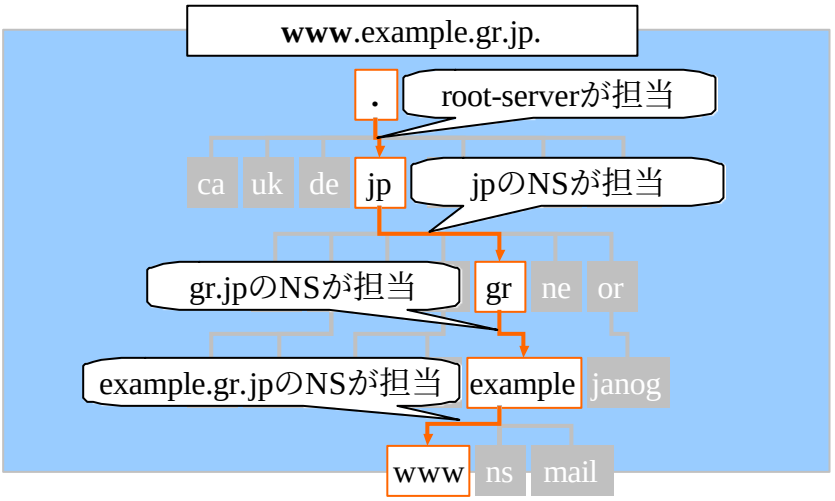
黒子として働くDNS

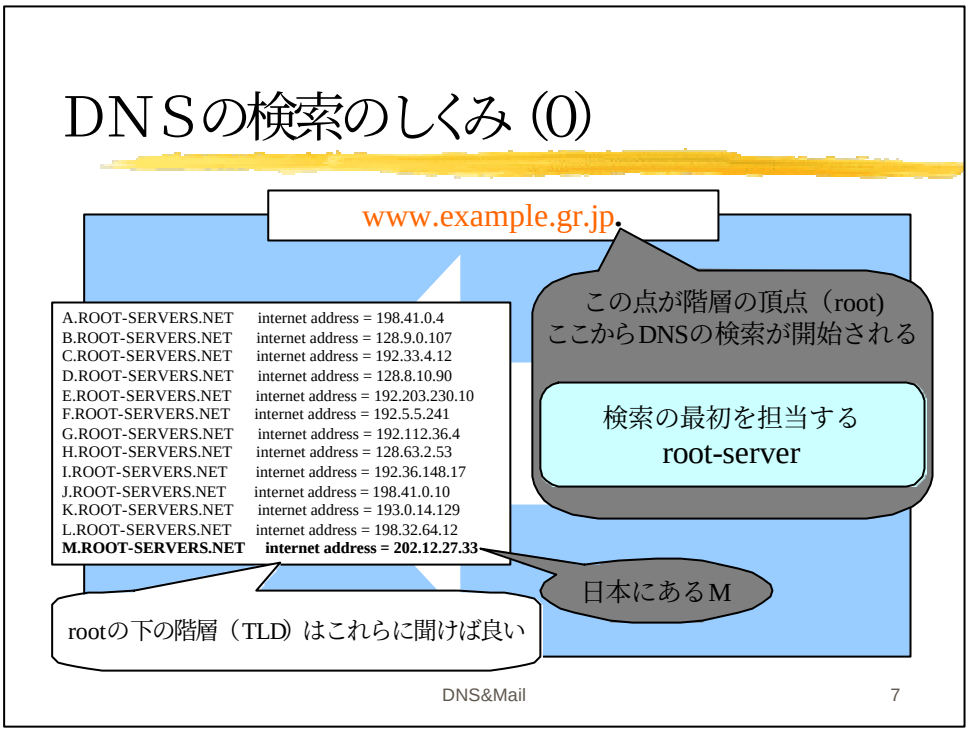


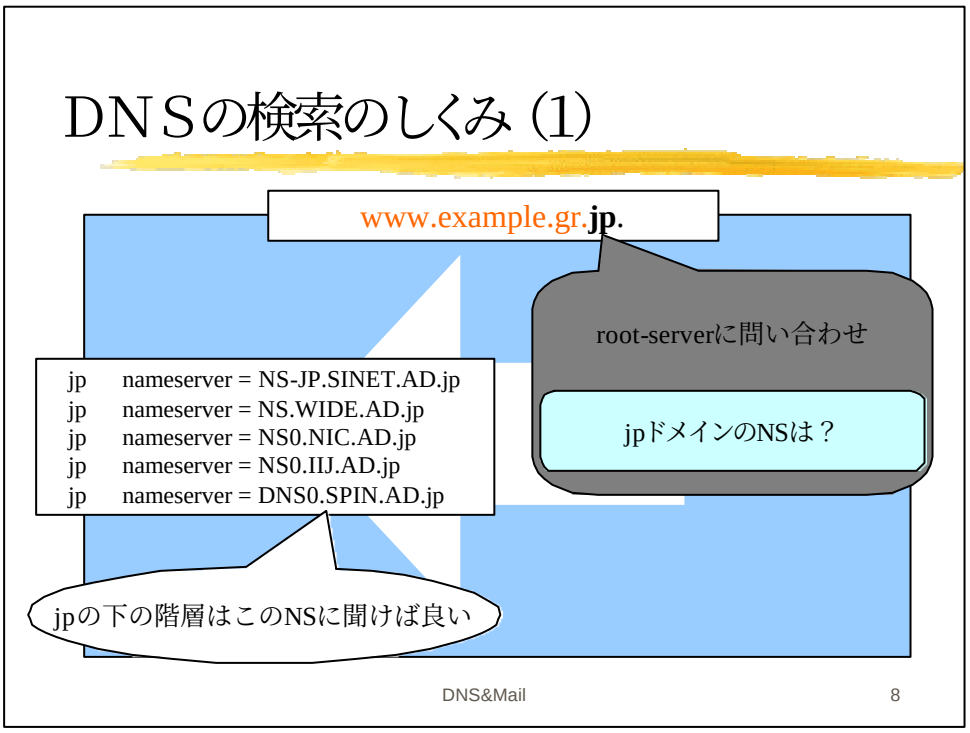
ドメイン名の階層構造



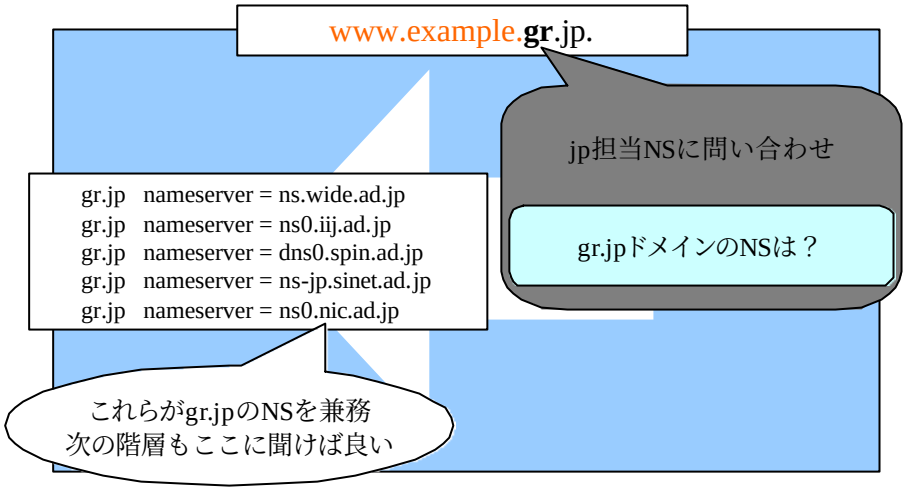
階層ごとの担当サーバ (NS)



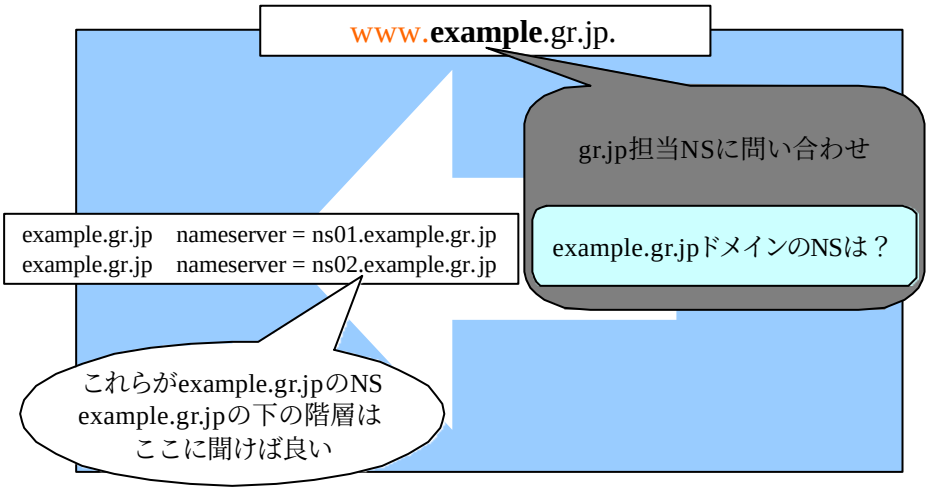

DNSの検索のしくみ (0)
www.example.gr.jp.
この点が階層の頂点（root)
ここからDNSの検索が開始される
検索の最初を担当する
root-server
A.ROOT-SERVERS.NET internet address = 198.41.0.4
B.ROOT-SERVERS.NET internet address = 128.9.0.107
C.ROOT-SERVERS.NET internet address = 192.33.4.12
D.ROOT-SERVERS.NET internet address = 128.8.10.90
E.ROOT-SERVERS.NET internet address = 192.203.230.10
F.ROOT-SERVERS.NET internet address = 192.5.5.241
G.ROOT-SERVERS.NET internet address = 192.112.36.4
H.ROOT-SERVERS.NET internet address = 128.63.2.53
I.ROOT-SERVERS.NET internet address = 192.36.148.17
J.ROOT-SERVERS.NET internet address = 198.41.0.10
K.ROOT-SERVERS.NET internet address = 193.0.14.129
L.ROOT-SERVERS.NET internet address = 198.32.64.12
M.ROOT-SERVERS.NET internet address = 202.12.27.33
日本にあるM
rootの下の階層（TLD）はこれらに聞けば良い
DNS&Mail
7


DNSの検索のしくみ (1)
www.example.gr.jp.
root-serverに問い合わせ
jpドメインのNSは？
jp nameserver = NS-JP.SINET.AD.jp
jp nameserver = NS.WIDE.AD.jp
jp nameserver = NS0.NIC.AD.jp
jp nameserver = NS0.IIJ.AD.jp
jp nameserver = DNS0.SPIN.AD.jp
jpの下の階層はこのNSに聞けば良い
DNS&Mail
8

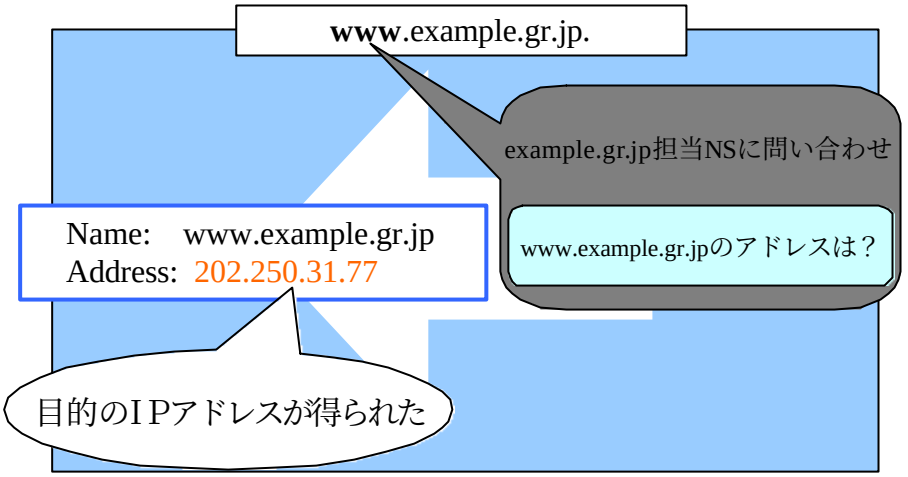
DNSの検索のしくみ (2)



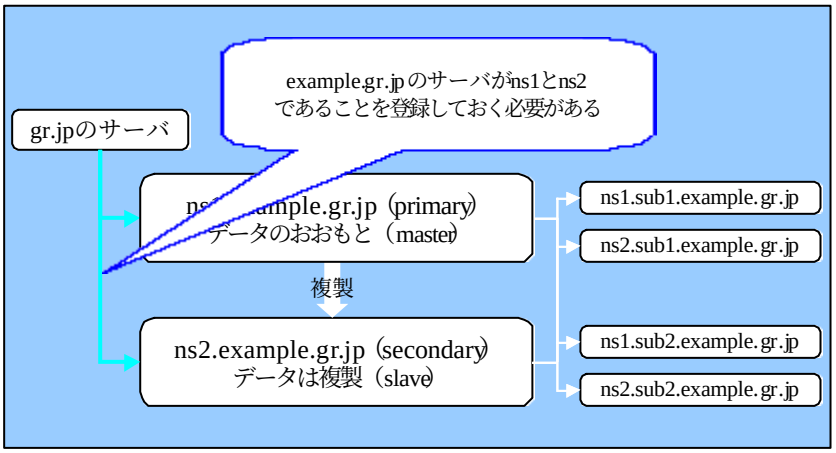
DNSの検索のしくみ (3)



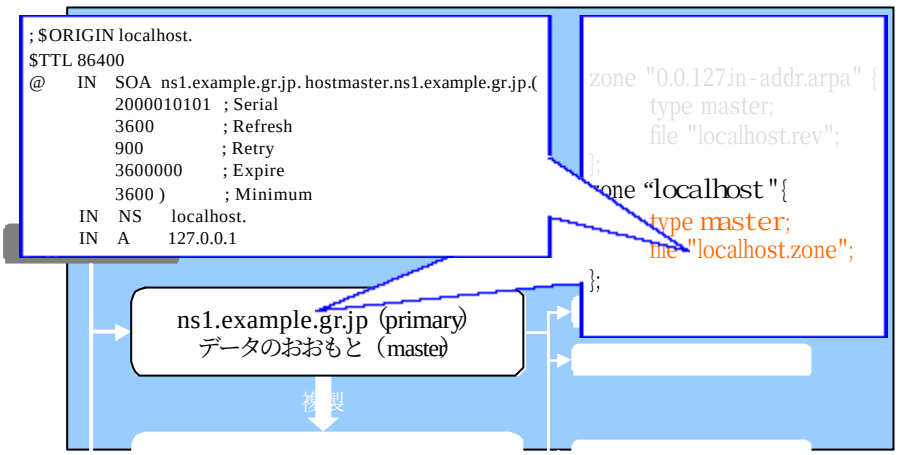
DNSの検索のしくみ (4)



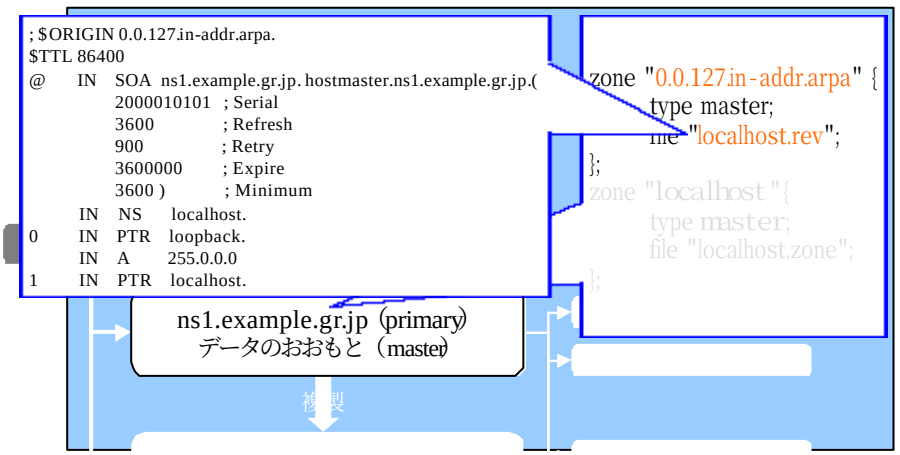
ドメインを支えるネームサーバ群



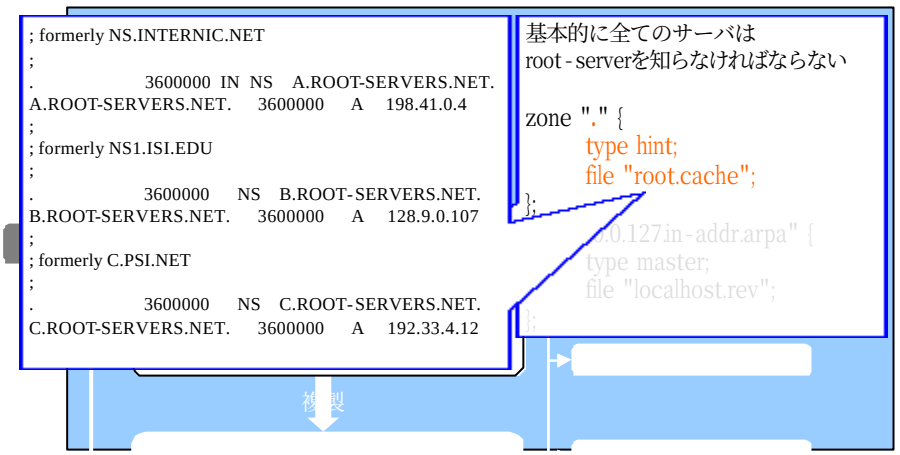
localhostの設定（正引き）



localhostの設定（逆引き）

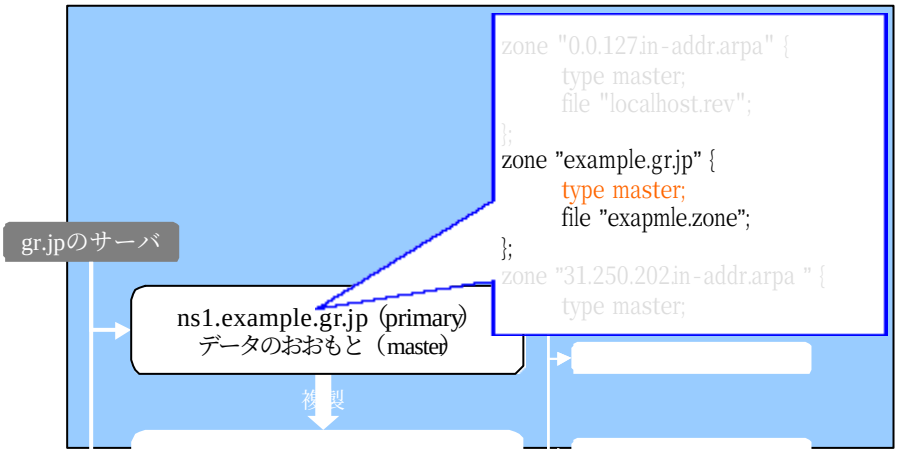


root-serverの設定



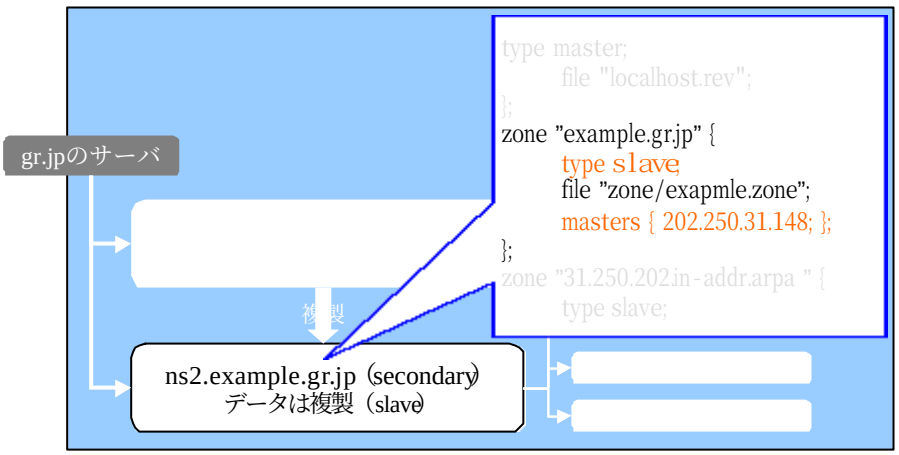
15

masterサーバの設定

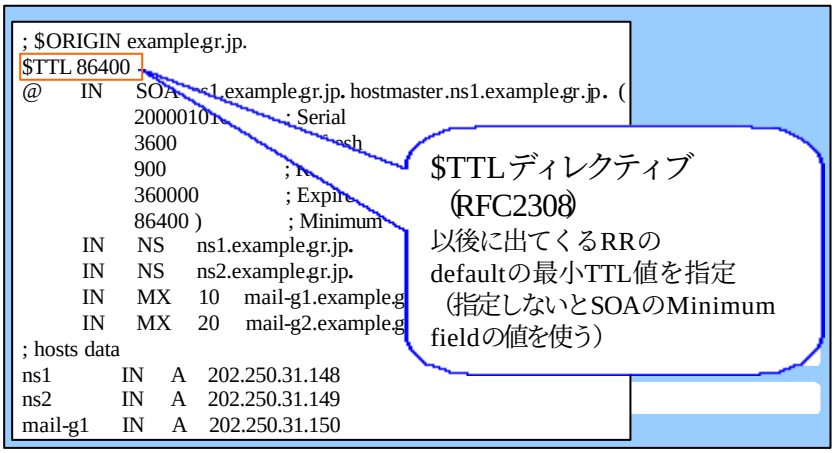


16

slaveサーバの設定



マスター（ゾーン）ファイルの設定



ディレクティブの種類

- $ORIGIN <origin domain>
  - current originを指定する
- $INCLUDE <filename>　[origin]
  - ファイルを読み込む
- $TTL <value>
  - defaultのTTLを指定する

RR（Resource Record）

- 文法

{<domain>|@|<blank>} [<ttl>] [<class>] <type> <rdata> [<comment>]

たとえば

```
@          86400  IN  SOA   ns1.example.gr.jp.  hostmaster.example.gr.jp. (
                             2000010101 3600 900 3600000 86400)
<blank>    86400  IN  NS    ns1.example.gr.jp.
<blank>    86400  IN  NS    ns2.example.gr.jp.
<blank>    86400  IN  MX    mail-g1.example.gr.jp.
ns1        86400  IN  A     202.250.31.148
<blank>    86400  IN  MX    mail-g1.example.gr.jp.
<blank>    86400  IN  HINFO  PC FreeBSD3.3
```

SOA RR

```
; $ORIGIN example.gr.jp.
$TTL 86400
@       IN      SOA  ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
                     2000010101         ; Serial
                     3600               ; Refresh
                     900                ; Retry
                     360000             ; Expire
                     86400 )            ; Minimum
        IN      NS      ns1.example.gr.jp.
        IN      NS      ns2.example.gr.jp.
        IN      MX   10    mail-g1.example.gr.jp.
        IN      MX   20    mail-g2.example.gr
; hosts data
ns1             IN    A    202.250.31.148
ns2             IN    A    202.250.31.149
mail-g1         IN    A    202.250.31.150
```

the Start of a zone Of Authority
マスターファイルの原本のありかと
管理者、設定パラメータを記入する

SOA RR

```
@       IN      SOA  ns1.example.gr.jp. hostma
                     20
                     3600
                     900                ; Retry
                     360000             ; Expire
                     86400 )            ; Minimum
        IN      NS      ns1.example.gr.jp.
        IN      NS      ns2.example.gr.jp.
        IN      MX   10    mail-g1.example.g
        IN      MX   20    mail-g2.example.g
; hosts data
ns1             IN    A    202.250.31.148
ns2             IN    A    202.250.31.149
mail-g1         IN    A    202.250.31.150
mail-g2         IN    A    202.250.31.151
www             IN    A    202.250.31.77
```

@ = current origin

named.confで指定したドメイン
（例ではexample.gr.jp）

```
zone "example.gr.jp" {
        type master;
        file "exapmle.zone";
};
```

SOA RR

```
@       IN      SOA  ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
                     2000010101     ; Serial
                     3600           ; Refresh
                     900            ; Retry
                     360000         ; Expire
                     86400 )        ; Minimum
        IN      NS   ns1.example.gr.jp.
        IN      NS   ns2.example.gr.jp.
        IN      MX   10   mail-g1.example.g
        IN      MX   20   mail-g2.example.gr.jp.
; hosts data
ns1             IN   A    202.250.31.148
ns2             IN   A    202.250.31.149
mail-g1         IN   A    202.250.31.150
mail-g2         IN   A    202.250.31.151
www             IN   A    202.250.31.77
```

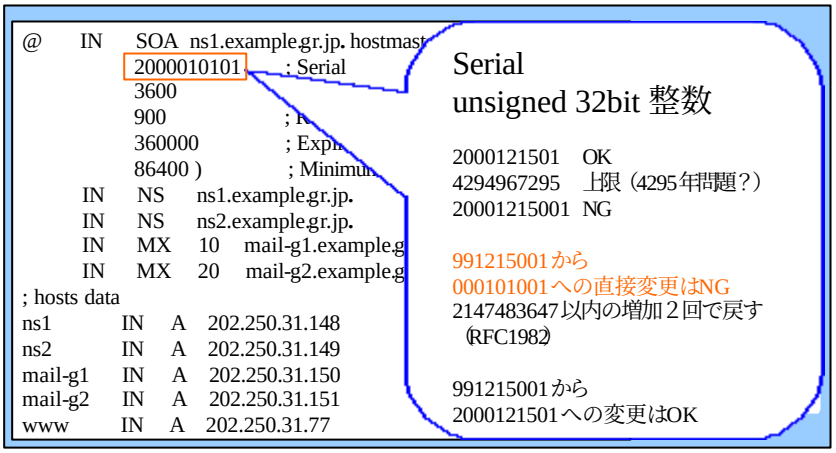
原本（master）のあるサーバ

SOA RR

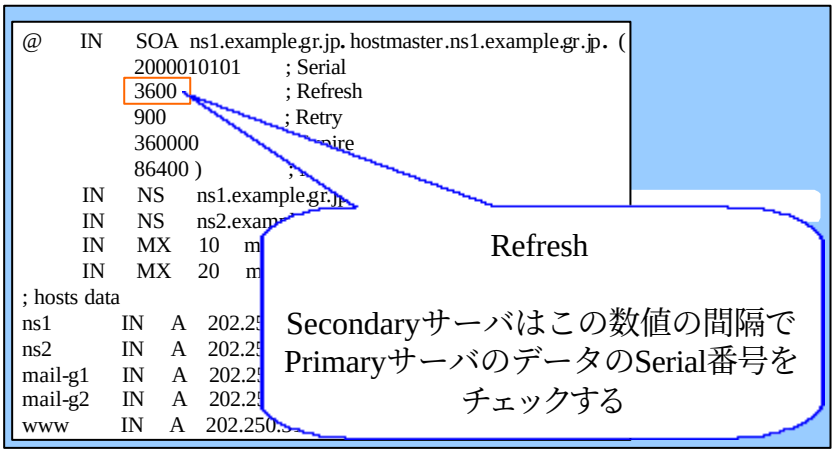
```
@       IN      SOA  ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
                     2000010101     ; Serial
                     3600           ; Refresh
                     900            ; Retry
                     360000         ; Expire
                     86400 )        ; Minimum
        IN      NS   ns1.example.gr.jp.
        IN      NS   ns2.example.gr.jp.
        IN      MX   10   mail-g1.example.g
        IN      MX   20   mail-g2.example.g
; hosts data
ns1             IN   A    202.250.31.148
ns2             IN   A    202.250.31.149
mail-g1         IN   A    202.250.31.150
mail-g2         IN   A    202.250.31.151
www             IN   A    202.250.31.77
```

管理者のメールアドレス
hostmaster@ns1.example.gr.jp
（RFC2142）

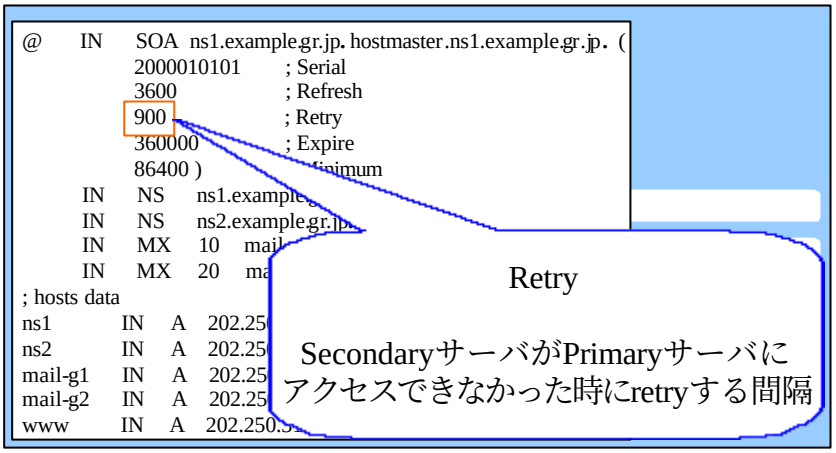
SOA RR


@ IN SOA ns1.example.gr.jp. hostmast
2000010101 ; Serial
3600
900 ; R
360000 ; Expi
86400) ; Minimu
IN NS ns1.example.gr.jp.
IN NS ns2.example.gr.jp.
IN MX 10 mail-g1.example.g
IN MX 20 mail-g2.example.g
; hosts data
ns1 IN A 202.250.31.148
ns2 IN A 202.250.31.149
mail-g1 IN A 202.250.31.150
mail-g2 IN A 202.250.31.151
www IN A 202.250.31.77
Serial
unsigned 32bit 整数
2000121501 OK
4294967295 上限（4295年問題？）
20001215001 NG
991215001から
000101001への直接変更はNG
2147483647以内の増加２回で戻す
(RFC1982)
991215001から
2000121501への変更はOK

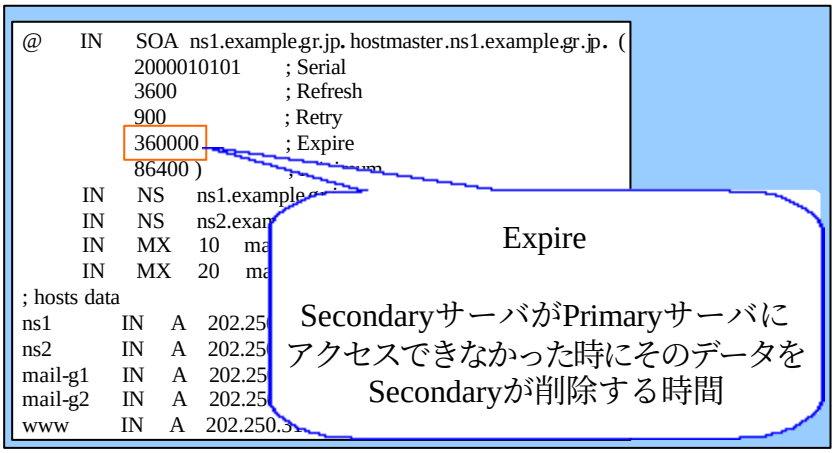
SOA RR


@ IN SOA ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
2000010101 ; Serial
3600 ; Refresh
900 ; Retry
360000 ire
86400) ;
IN NS ns1.example.gr.jp
IN NS ns2.exam
IN MX 10 m
IN MX 20 m
; hosts data
ns1 IN A 202.25
ns2 IN A 202.25
mail-g1 IN A 202.25
mail-g2 IN A 202.25
www IN A 202.250.3
Refresh
Secondaryサーバはこの数値の間隔で
Primaryサーバのデータの Serial番号を
チェックする

SOA RR


@ IN SOA ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
2000010101 ; Serial
3600 ; Refresh
900 ; Retry
360000 ; Expire
86400)
IN NS ns1.example.gr.jp.
IN NS ns2.example.gr.jp.
IN MX 10 mail
IN MX 20 ma
; hosts data
ns1 IN A 202.25
ns2 IN A 202.25
mail-g1 IN A 202.25
mail-g2 IN A 202.25
www IN A 202.250.3
Retry
SecondaryサーバがPrimaryサーバに
アクセスできなかった時にretryする間隔

SOA RR


@ IN SOA ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
2000010101 ; Serial
3600 ; Refresh
900 ; Retry
360000 ; Expire
86400)
IN NS ns1.example.gr.jp.
IN NS ns2.exam
IN MX 10 ma
IN MX 20 ma
; hosts data
ns1 IN A 202.25
ns2 IN A 202.25
mail-g1 IN A 202.25
mail-g2 IN A 202.25
www IN A 202.250.3
Expire
SecondaryサーバがPrimaryサーバに
アクセスできなかった時にそのデータを
Secondaryが削除する時間

SOA RR

```
@       IN      SOA  ns1.example.gr.jp. hostmaster.ns1.example.gr.jp. (
                     2000010101     ; Serial
                     3600           ; Refresh
                     900            ; Retry
                     360000         ; Expire
                     86400 )        ; Minimum
        IN      NS   ns1.example.gr.jp.
        IN      NS   ns2.example.gr.jp.
        IN      MX   10 mail-g1.example.gr.jp.
        IN      MX   20 mail-g2.example.gr.jp.
; hosts data
ns1             IN   A   202.250.3.1
ns2             IN   A   202.250.3.2
mail-g1         IN   A   202.250.3.3
mail-g2         IN   A   202.250.3.4
www             IN   A   202.250.3.5
```

Minimum

nagative responseにおける
最小TTL（Time To Live/有効時間）
→ RFC2308 section 4

NS（Name Server） RR

```
@       IN      SOA  ns1.example.gr.jp. root.ns1.example.gr.jp. (
                     2000010101     ; Serial
                     3600           ; Refresh
                     900            ; Retry
                     360000         ; Expire
                     86400 )        ; Minimum
        IN      NS   ns1.example.gr.jp.
        IN      NS   ns2.example.gr.jp.
        IN      MX   10 mail-g1.example.gr.jp.
        IN      MX   20 mail-g2.example.gr.jp.
; hosts data
ns1             IN   A   202.250.3.1
ns2             IN   A   202.250.3.2
mail-g1         IN   A   202.250.3.3
mail-g2         IN   A   202.250.3.4
www             IN   A   202.250.3.5
```

example.gr.jpのprimary/secondaryの
ネームサーバ（NS）を記述しておく

MX（Mail eXchanger） RR

```
        IN    NS    ns1.example.gr.jp.
        IN    NS    ns2.example.gr.jp.
        IN    MX    10    mail-g1.example.gr.jp.
        IN    MX    20    mail-g2.example.gr.jp.
; hosts data
ns1         IN    A    202.250.31.148
ns2         IN    A    202.250
mail-g1     IN    A    202.25
mail-g2     IN    A    202.25
www         IN    A    202.25
proxy       IN    A    202.25
cache       IN    CNAME
; subdomain sub1 (delegatio
sub1        IN    NS   ns1.s
            IN    NS   ns2.s
```

@example.co.jp のメールアドレスに
対するメールの送り先サーバを指定

数字はpreferenceで小さい方が優先
0～32767の値を指定する

サーバ名にCNAMEを書いてはいけない

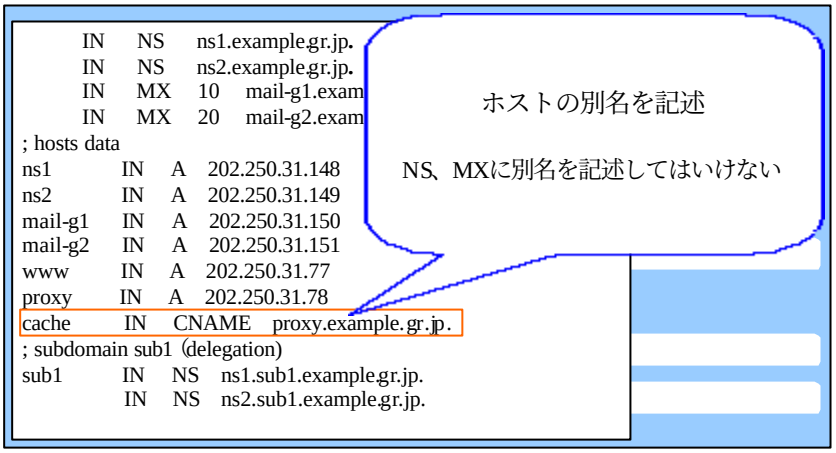
A（Address） RR

```
        IN    NS    ns1.example.gr.jp.
        IN    NS    ns2.example.gr.jp.
        IN    MX    10    mail-g1.exam
        IN    MX    20    mail-g2.exam
; hosts data
ns1         IN    A    202.250.31.148
ns2         IN    A    202.250.31.149
mail-g1     IN    A    202.250.31.150
mail-g2     IN    A    202.250.31.151
www         IN    A    202.250.31.77
proxy       IN    A    202.250.31.78
cache       IN    CNAME    proxy.example.gr.jp.
; subdomain sub1 (delegation)
sub1        IN    NS    ns1.sub1.example.gr.jp.
            IN    NS    ns2.sub1.example.gr.jp.
```

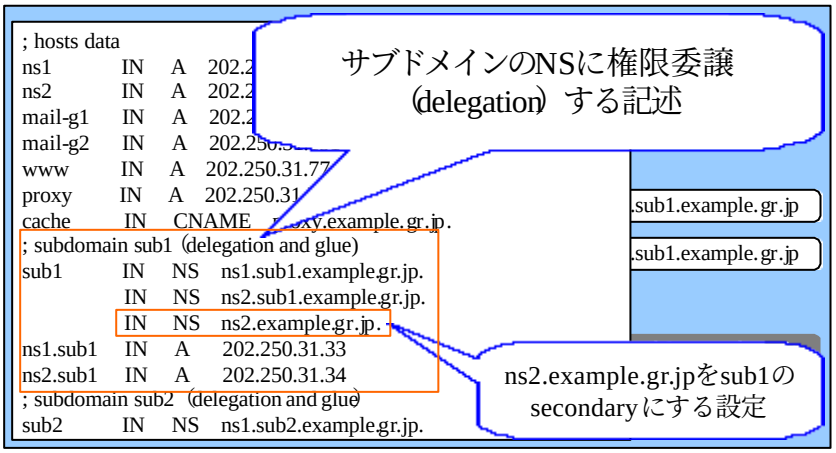
各ホストの名称とIPアドレスを記述

NS RR、MX RRに記述したホストの
アドレスは必ず記述しておく

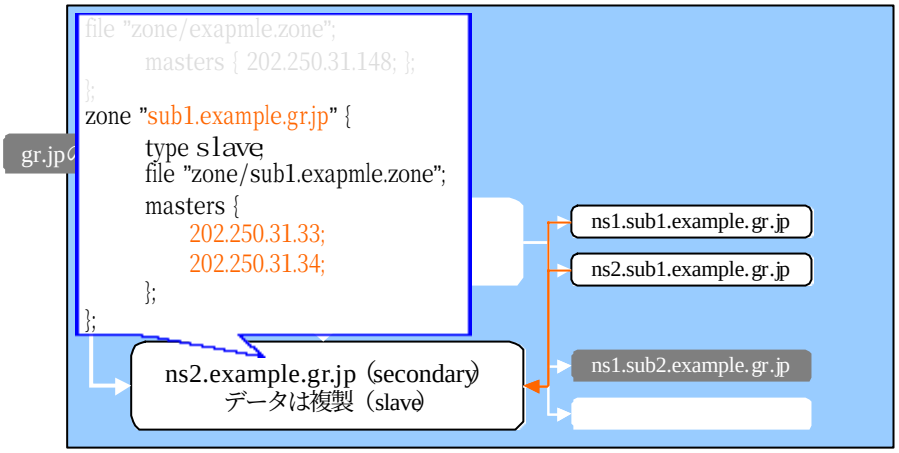
CNAME（Canonical Name）RR



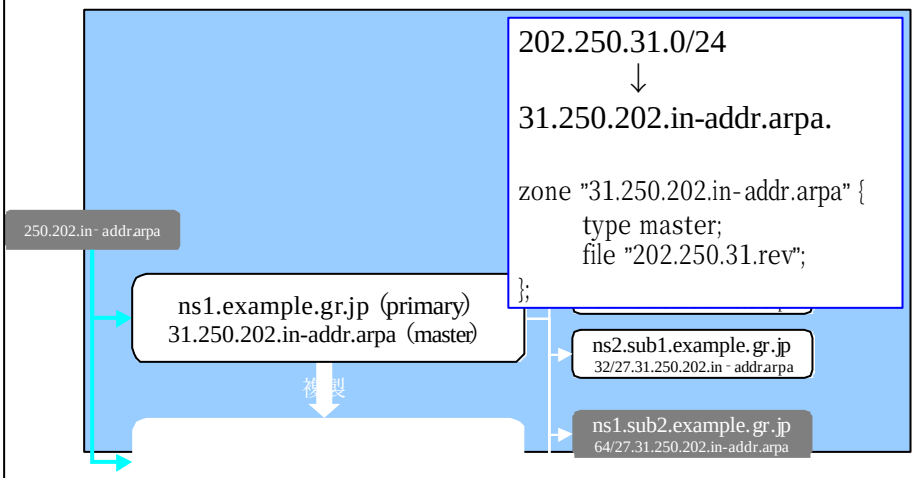
Delegation



sub1のsecondaryにする設定



逆引きアドレス



ネットワーク名のマッピング

```
; $ORIGIN 31.250.202.in-addr.arpa.
$TTL 86400
@   IN  SOA  ns1.example.gr.jp.  host
                2000010101      ; Serial
                3600            ; Refresh
                900             ; Retry
                360000          ; Expire
                86400 )         ; Minimu
        IN      NS      ns1.example.gr.jp.
        IN      NS      ns2.example.gr.jp.
0       IN      PTR     example-net.example.gr.jp.
        IN      A       255.255.255.0
example-net.example.gr.jp.   IN  PTR   0.31.250.202.in-addr.arpa.
1       IN      PTR     gw.example.gr.jp.
32/27   IN      NS      ns1.sub1.example.gr.jp.
        IN      NS      ns2.sub1.example.gr.jp.
```

ネットワーク名のマッピング
(A RRはnetmask)
netstatコマンド等で参照される
→ RFC1101 Section 4

.sub1.example.gr.jp
7.31.250.202.in - addr.arpa

.sub2.example.gr.jp
7.31.250.202.in-addr.arpa

.sub2.example.gr.jp
7.31.250.202.in-addr.arpa

Classless IN-ADDR.ARPA delegation

```
        IN      NS      ns1.example.gr.jp.
        IN      NS      ns2.example.gr.jp.
0       IN      PTR     example-net.example.gr.jp.
        IN      A       255.255.255.0
1       IN      PTR     gw.example.gr.jp.
; 202.250.31.32/27 delegation
32/27   IN      NS      ns1.sub1.example.gr.jp.
        IN      NS      ns2.sub1.example.gr.jp.
$GENERATE 33-62 $ CNAME $.32/27.31.250.202.in-addr.arpa.
77      IN      PTR     www.example.gr.jp.
; 202.250.31.96/27 delegation
96/27   IN      NS      ns1.sub2.
        IN      NS      ns2.sub2.
$GENERATE 97-126 $ CNA
148     IN      PTR     ns1.exam
149     IN      PTR     ns2.exam
```

.sub1.example.gr.jp
7.31.250.202.in - addr.arpa

.sub1.example.gr.jp
7.31.250.202.in - addr.arpa

202.250.31.32/27を
ns1.sub1.example.gr.jpに
権限委譲（delegation）

$GENERATE directive

- $GENERATE <range> <lhs> <type> <rhs>
  - ex.

$GENERATE 33-62 $ CNAME pc$.example.gr.jp.

↓

```
33    CNAME   pc33.example.gr.jp.
34    CNAME   pc34.example.gr.jp.
...
62    CNAME   pc62.example.gr.jp.
```


Classless IN-ADDR.ARPA delegation

```
; $ORIGIN 32/27.31.250.202.in-addr.arpa.
$TTL 86400
@   IN  SOA  ns1.sub1.example.gr.jp. hostmaster.example.gr.jp. (
                2000010101      ; Serial
                3600            ; Refresh
                900             ; Retry
                360000          ; Expire
                86400 )         ; Minimum
    IN   NS    ns1.sub1.example.gr.jp.
    IN   NS    ns2.sub1.example.gr.jp.
32  IN   PTR   example-sub1.example.gr.jp.
    IN   A     255.255.255.32
33  IN   PTR   ns1.sub1.example.gr.jp.
34  IN   PTR   ns2.sub1.example.gr.jp.
35  IN   PTR   mail.sub1.example.gr.jp.
36  IN   PTR   www.sub1.example.gr.jp.
```

Classless IN-ADDR.ARPA delegation

- RFC2317
  - 202.250.31.33の逆引き
    - 33.31.250.202.in-addr.arpa
      - ↓ CNAME on ns1.example.gr.jp
    - 33.32/27.31.250.202.in-addr.arpa
      - ↓ PTR on ns1.sub1.example.gr.jp
    - ns1.sub1.example.gr.jp

master fileについての注意点

- データを更新したら、必ずそのmaster fileのSerialを増加させること.
  - secondaryは更新をSerialでチェックしている
- 変更が終わったら、namedをrestartする
  - # ndc restart
  - bind8ではこれで更新要求がsecondaryに伝えられ、secondaryのデータも更新される.(RFC1996)

bind8の新しい仕様

- RFC1995：IXFR
  - zoneファイルの変更部分だけを転送して更新の効率を上げる。
- RFC1996：DNS NOTIFY
  - masterファイルの更新をsecondaryに通知する

bind8の新しい仕様

- RFC2065：DNSSEC
  - 3つの目的に使われる
    - key distribution
    - data origin authentication
    - transaction and request authentication

その他の注意点

- NSのマシンは他から同時に到達できなくならない場所に設置する
- Authorized Server
  - 上位サーバから権限委譲を受けたNS
- Unauthorized Server
  - 権限委譲を受けていないNS
- Lame Server
  - 権限委譲されているのにPrimary/Secondaryでない
  - メールが配送されない

さらなる拡張

- RFC2535（RFC2065の続き）
  - SIG RR
  - NXT RR
  - RFC2536～2539（各暗号形式のDNSへの適用）
- RFC2672（DNAME RR）
  - ドメイン名のCNAMEに相当するもの
- draft-ietf-dnsind-tsig-11.txt
  - TSIG RR（transaction signature）

DNSとメール

- user@example.gr.jp
  - ここから配送先をどう見つけるか?
  - 手がかりはドメイン名の部分
- example.gr.jpのMXレコードを調べる
  - 配送にはMXとサーバのAレコードが必要
    - 最も効率が良いのは、 MXを聞いたらMXだけではなくAが同時に返ってくる場合
    - 答えるnamedがMXとAを両方知っているのがベスト

MXはCNAMEではいけない

- O DontExpandCnames=False
  - RFC822,1123的にはたぐるのが正しい
  - IETFはCNAMEをたぐらない方向に動いている
  - というわけでsendmailはオプションにする模様
- DNSのMXはCNAMEを指定してはいけない
  - RHSにはAを書く
  - そのAはMXを答えるnamedが知っていると良い

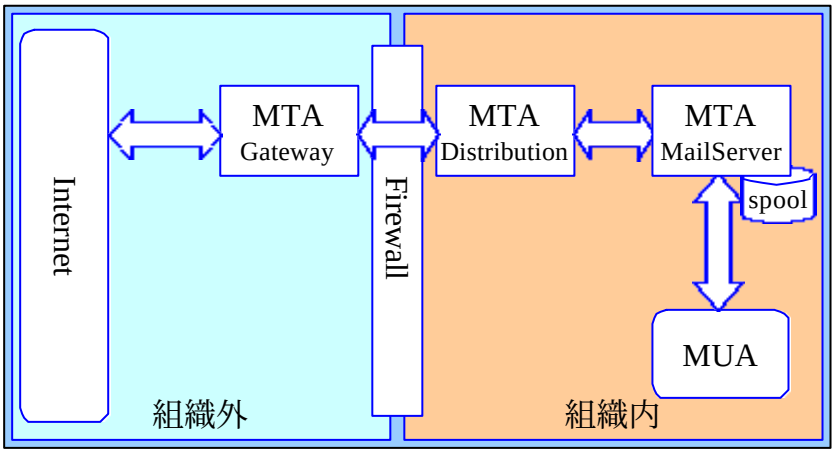
ワイルドカードMXは使わない

- 手抜きはいけません
  - sendmailのREADMEには…
    - ワイルドカードが自分のドメインを含まない特殊なDB
    - 指している先がfirewall

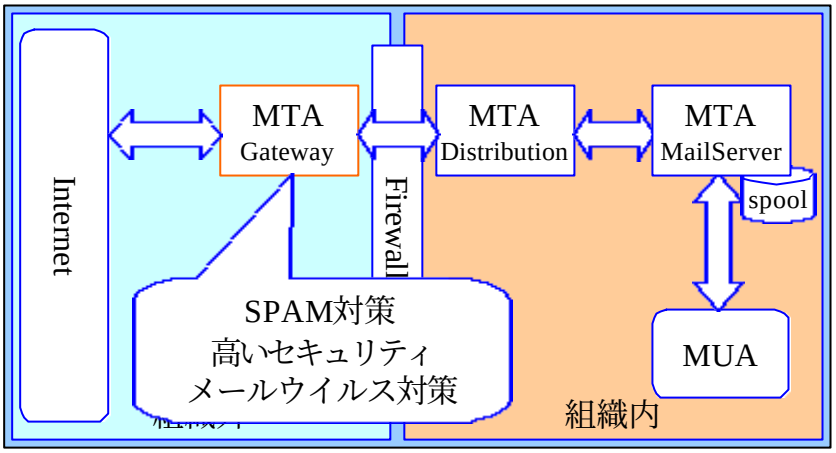
  という状況でしかうまく動かない
  これ以外だと頭痛のタネにしかならない

  と書いてあったりする

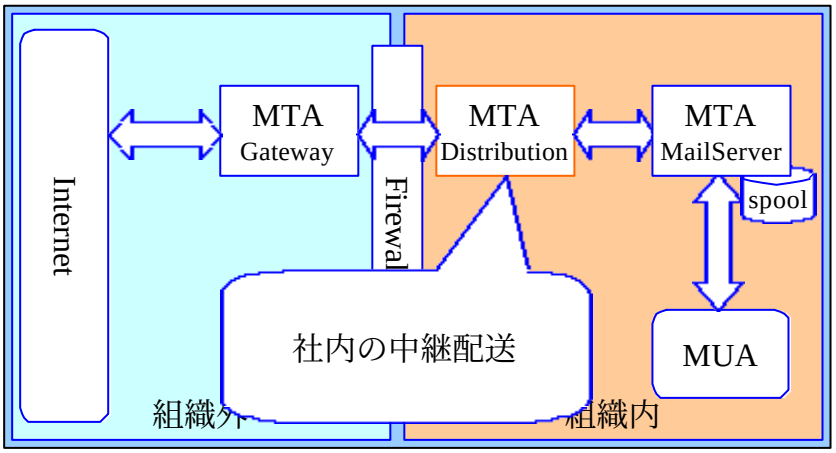
メール配送モデル



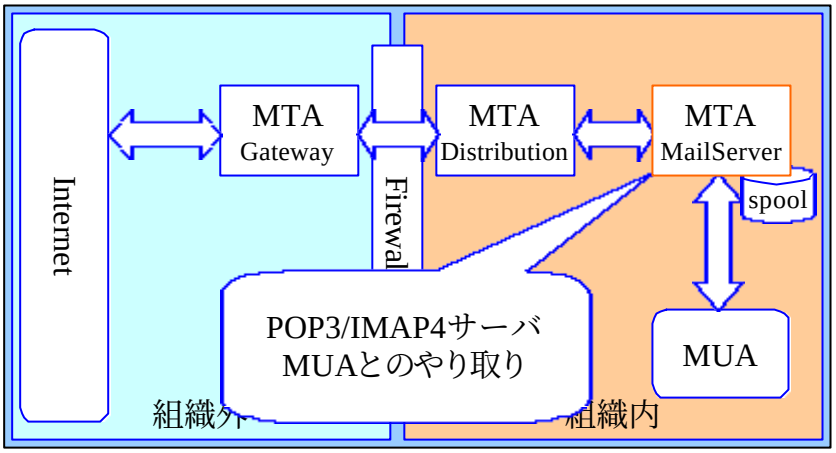
メール配送モデル



メール配送モデル

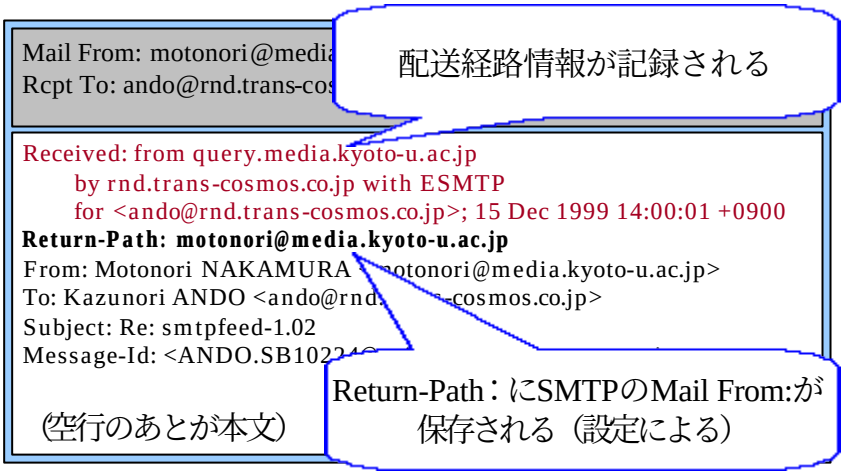


メール配送モデル



メール本体とエンベロープ

メール本体とエンベロープ (2)



メール本体とエンベロープ (3)

- え？ To: に書くとそこに送られるじゃん...
  - それはMUAの仕業
  - To: ヘッダに書いたアドレスをSMTP的な配送先情報としてMTAに渡しているだけ
  - 例えば、To:ヘッダがメーリングリストのアドレスなのに自分にメールが届くのはこのため

ヘッダの話 (1)

- Field-name: Field-body (standard)
  - From: 差出人アドレス
  - Sender：差出人アドレスが不明確な場合に差出人を明示
  - To: 宛先アドレス
  - Cc: カーボンコピー
  - Reply-To: 返信先アドレス
  - Message-Id: ５年間固有のID
  - Subject: タイトル
  - Date: 差出時間
  - Return-Path: エラー返信先アドレス

ヘッダの話 (2)

- Field-name: Field-body (standard)
  - Received: 配送経路
  - In-Reply-To: どのメールに返信したかを示す
  - References： どのメールに返信したかを示す
- Resent系（メールを再配信する場合の）
  - Resent-From: 差出人アドレス
  - Resent-Sender： 差出人アドレスが不明確な場合に明示
  - Resent-Reply-To: 返信先アドレス
  - Resent-Message-Id: ５年間固有のID
  - Resent-Date: 再配信日時

ヘッダの話 (3)

- Field-name: Field-body
  - Precedence: 配送優先度
  - X-Authentication-Warning: アドレス詐称（?）
- (おまけ) MLドライバ等の付けるヘッダ
  - X-MLServer: fml
  - X-ML-System : ppml
  - X-Ml-Version: kkml
  - X-Distribute: distribute
  - Delivered-To: qmail

文字の話

- 機種依存文字を使ってはいけない
  - ①②③④⑤⑥⑦⑧⑨⑩
  - Ⅰ Ⅱ ⅢⅣ Ⅴ ⅥⅦⅧⅨ Ⅹ
  - ﾄｳｾﾞﾝﾊﾝｶｸｶﾀｶﾅﾓﾀﾞﾒ
- 漢字コードはISO-2022-JPを使用する
  - SJISだめ、EUCだめ、UNICODEだめ

配送の実際 (1)

- 宛先アドレスのMXを引いてみる

MX 10 mail-g1.example.gr.jp
MX 10 mail-g2.example.gr.jp

- この場合はランダムでどちらかに配送

MX 10 mail-g1.example.gr.jp
MX 20 mail-g2.example.gr.jp

- この場合は10の方に配送して駄目だったら

配送の実際 (2)

- MX RRの他にも答がいっぱい返ってくる

```
example.gr.jp   MX  10   mail-g1.example.gr.jp
example.gr.jp   MX  20   mail-g2.example.gr.jp
```

MX RR

```
example.gr.jp   NS   ns1.example.gr.jp
example.gr.jp   NS   ns2.example.gr.jp
mail-g1.example.gr.jp   A   202.250.31.150
mail-g2.example.gr.jp   A   202.250.31.151
ns1.example.gr.jp    A  202.250.31.148
ns2.example.gr.jp    A  202.250.31.149
```

additinal information

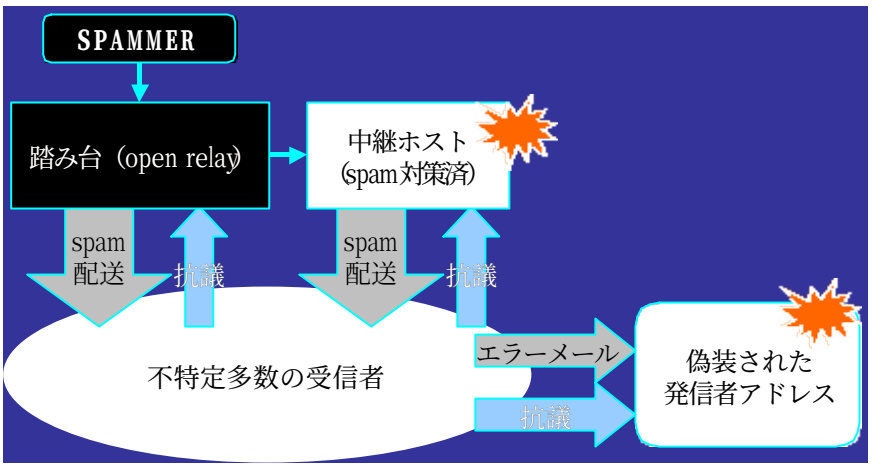
実際の配送 (3)

- Additional Information
  - MXを聞かれたNSがMXのAも知っている
    - MXとAがわかれば実際に接続しにいける
    - １回のqueryで済むので効率が良い
    - MX RRを保持しているNSがAも保持することが重要
  - Additional InformationとMTA
    - 例えばSMTPfeedはAdditional Informationを利用
    - MTAが利用しなくても手元のnamedがcache
      - Aを聞きに行くとそこが答えるので速い

配送トラフィック

- 回線帯域を示さないベンチマークは無意味
  - 「n万通しか配送できない」？
    - 多くは対外回線の帯域がボトルネック
    - 例えばsendmailは1.5Mbpsの回線を埋められる
  - 機能まで考慮しない比較は無意味
    - 総合性能を評価する場合そろえるべき条件
      - 実現している機能（binary-codeはMIME-encodeとか）
      - 計算機リソース・OS
      - 使用回線帯域

SPAM中継の被害の構図



SPAM対策 (1)

- RBL（Realtime Blackhole List）
  - SPAMの発信源を登録する閻魔帳
    - DNSと同じ枠組みで作られている
      - MTAがメール送信元のIPアドレスを照会
    - 類似のものが何種類もある
      - MAPS RBL、MAPS RSS、MAPS DUL、ORBS、IMRSS等
    - 自分のサーバが登録された場合
      - メールを受け取らない所が出てくる
    - RBLのサーバが落ちると「**全てのIPアドレスはクロ**」
      - 全部受け取り拒否して全然メールが来なくなる

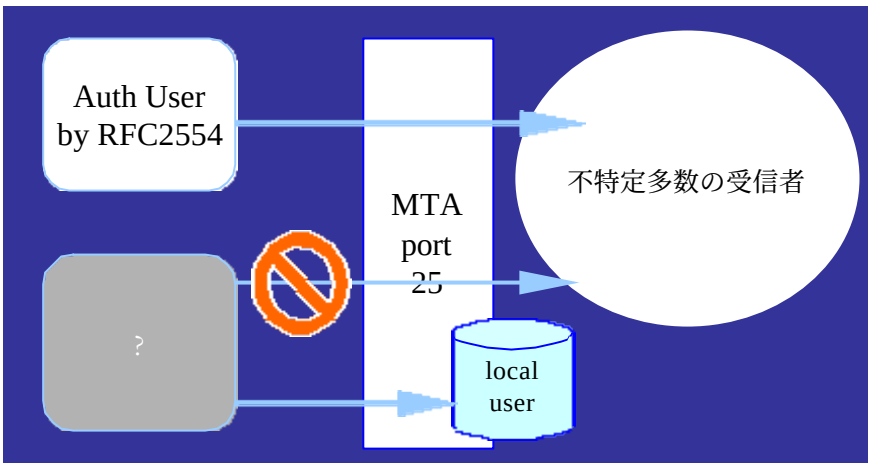
SPAM対策 (2)

- SPAMLIST
  - 発信元についていずれかを指定して排除
    - メールアドレス（envelope from)
    - ドメイン
    - IPアドレス
- POP before SMTP
  - ISPで取り入れられている手法
    - POPアクセスするとSMTP接続を許可する
    - 例えばqpopperにパッチを当てて実現する

SMTP Authentication（RFC2554)

- SASL（RFC2222）を利用したRelay認証
  - sendmail-8.10では
    - cyrus SASLライブラリを利用
    - SASLのデフォールトは/etc/passwdを利用した認証
    - 認証を通るとそのサーバ経由のRelay配送を許可
  - POP before SMTPの置き換え
    - MUAが対応が条件ではある

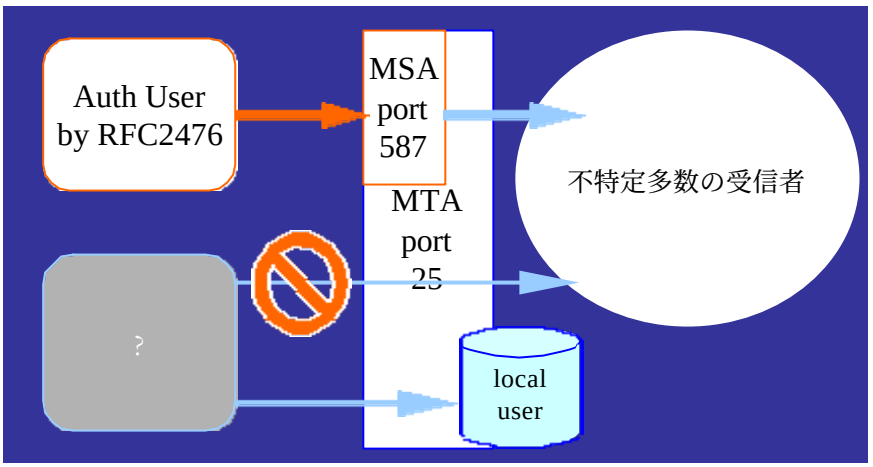
SMTP Authentication (RFC2554)



Message Submission (RFC2476)

- MSA (Message Submission Agent)
  - メールを「出す」新たな枠組み
    - Relayと区別することでSPAMを防止
      - SMTPではlocal宛のメールしか受けない
      - Submissionによる発信は自分のサイトからの接続だけを許可してさらに認証をかける
    - port 587
      - sendmail-8.10はMSAになれる

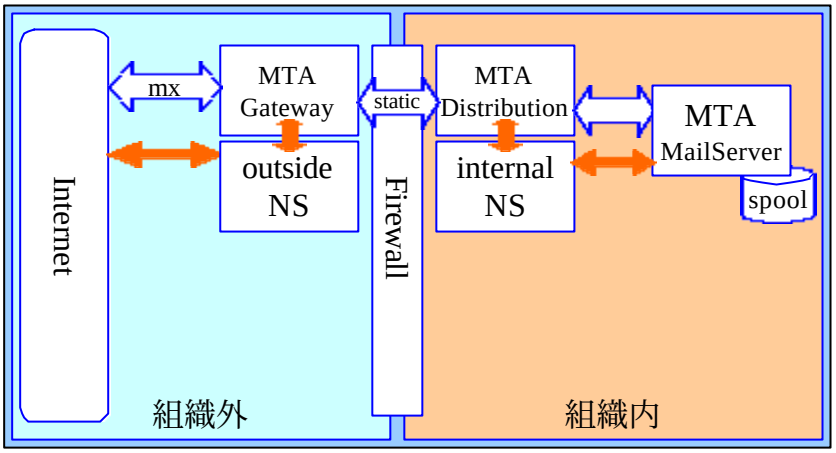
Message Submission (RFC2476)



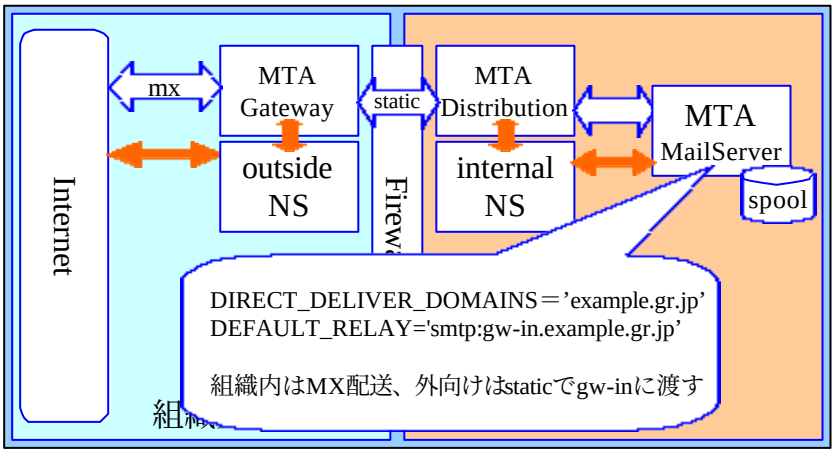
配送設定の基本要素

- MX配送かstatic（静的）配送か?
  - 対外配送はMX配送
  - 組織内部の配送はどちらか選択
    - 組織内部で独自のDNSの定義をしている場合
    - 集中サーバならstaticでもいける
  - resolv.confで参照するDNSサーバを指定

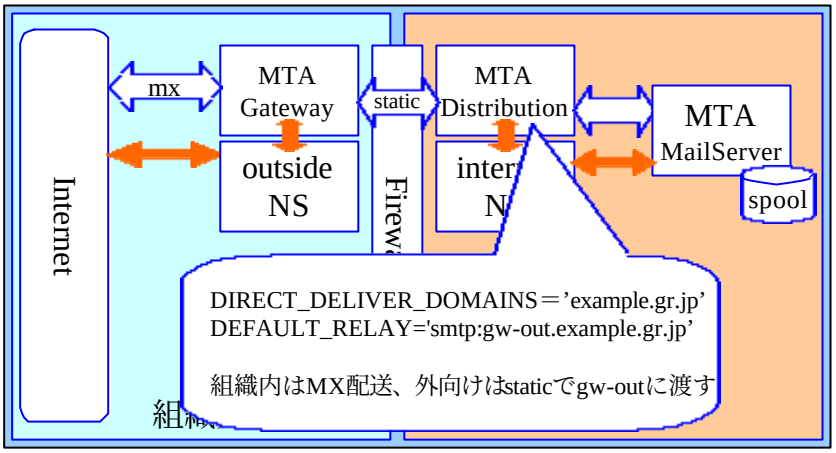
メール配送モデル



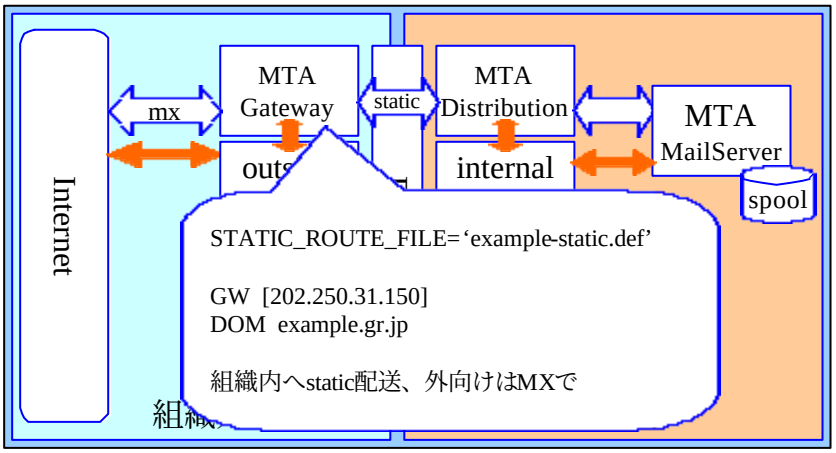
内部MSの設定



内側distributionサーバの設定



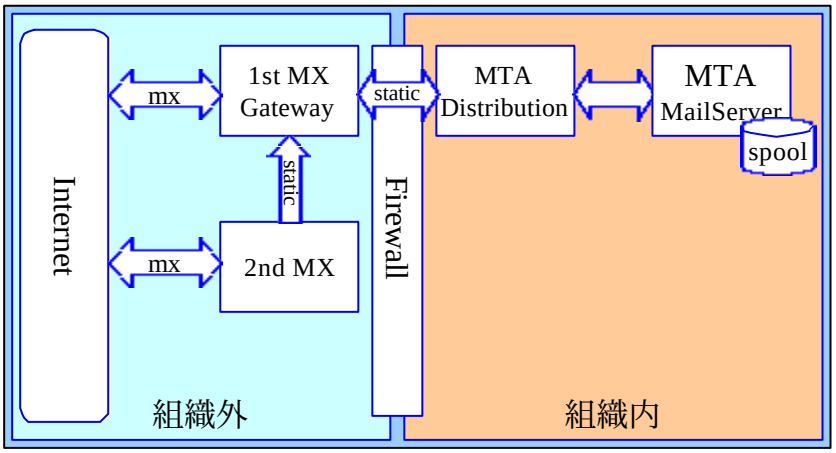
外側サーバの設定



対外受信ホストの多重化

- MXを複数にする理由
  - メールが集中して負荷が高い場合
    - 一時的にため込む
    - 可能なら2nd MXは1st MXとは独立に配信
  - メンテナンス用
    - 片方が停止しても受け取りに支障を出さない

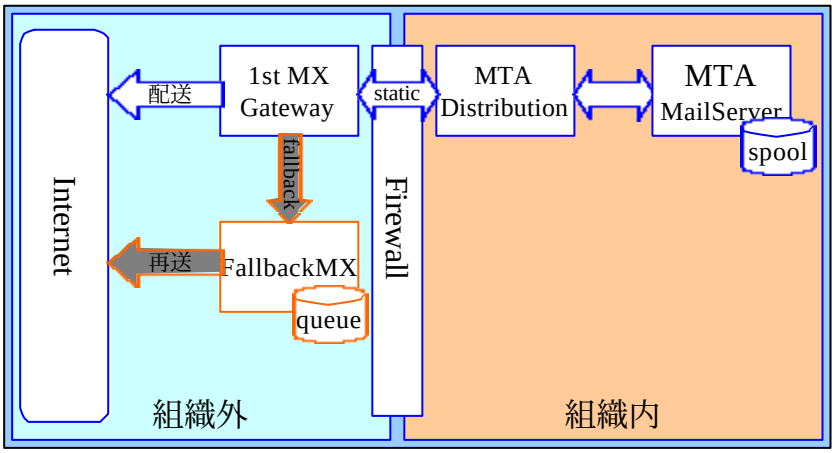
2nd MXのあるメール配送モデル



FallbackMX

- 再送専用ホスト
  - 再送queueを特定のホストに集める
    - DNSが引けなかった場合
    - 全MXに対してメールが送れなかった場合
  - ネットワーク的なトラブルがすぐわかる
  - 再送を試みる期間の調整
  - 大量配信を行う場合は負荷を分散できる

FallbackMXのある配送モデル



MTAの種類（1）

- Sendmail（商用版：Sendmail Pro）
  - さまざまな意味で最も手堅い選択
  - 処理限界はファイルシステムのスピードに依存
    - 途中で電源が落ちても....の対策をちゃんとしている
  - 大規模MLの配送ではsmtpfeedを併用
    - 外部メーラへの受け渡しでLMTPを話せる
  - 難解なsendmail.cf
    - GUIで設定ができる商用版Sendmail Proもある

MTAの種類（2）

- qmail
  - 単機能の小さいプログラムの集合体
    - 比較的パフォーマンスは良い
    - 個々の機能を見ると必ずしもベストを尽くしていない
      - 同一MX宛てのメールのまとめ送りがない
      - ひらたく言えば配送戦略は「力任せ」
  - トラブルが発生しても原因がさっぱりわからない
    - ログ情報の不足、エラーメッセージの不備

MTAの種類（3）

- その他商用製品
  - Netscape Messaging Server
  - InterMail
  - SIMS（Sun Internet Mail Server）
    - オールインワンサーバ（MTA＋仲介サーバ）
    - ユーザ数ベースの課金形態
      - データベースを併用しているから？
    - ユーザ数の上限を拡大しているのが売り

MTA選び（1）

- アカウント管理
  - OSに依存： Sendmail
  - LDAPを利用： SIMS等商用製品
    - 実はSendmailでも頑張ればLDAPは利用できる
- 仲介サーバはpopかimapか？
  - popはメールをクライアントが保持する
    - リスク分散と見ることができる
  - imapは便利だがサーバがメールを保持
    - パフォーマンスもリスクもサーバに依存

MTA選び（2）

- サーバの実際の収容数は...
  - 定期メールチェックのアクセス頻度に依存
    - ネットワーク接続の同時接続数上限はOS依存
      - メールチェックはネットワークとファイルI/Oのリソースを消費
    - 100Mbpsのリソースも１００００人で使えば１０kbps
      - リソースの上限近くでは急激にパフォーマンスが悪化する
    - セールストークにありがちなごまかし :)
      - × 少数ユーザでテストして大人数の場合を推測する
      - × ストライピング（RAID0)を利用したストレージの利用
        - 製品の性能ではなくストレージの性能が高い

MTA選び（3）

- セキュリティホール
  - そもそも「あると仮定しておくべきもの」である
    - 一般にシェアの大きい方が発見が早い
    - ＯＳ自体のセキュリティホールの方が圧倒的に多い
      - 商用製品なら安心かというと必ずしもそうではない
      - MTA自体が完璧なら安心というものでもない
    - セキュリティが心配ならそういう情報を得る努力を
      - 実際、セキュリティホールは呆れるほど多い
      - 例えば、http://www.securityfocus.com/

エラーメールの基礎

- エラーメール配信の枠組み
  - DSN (Delivery Status Notification)
    - Envelope From は null address (<>)
      - エラーメールに返信アドレスはない
- トラブルの種類を判定する手段
  - RFC1893 (Status Code)
    - Status: 5.1.1
      - 5.X.X　Permanent Failure
      - X.1.1　Bad destination mailbox address

エラーハンドリング問題

- 配送エラーコード (status code) の実装
  - 実際にRFCを守っているか？
    - sendmailやSIMS等守っているものも多い
    - その他の対応はいまいち
      - MTAの数だけエラーハンドリングのプログラムが必要
      - 標準を守ろうとしないMTAは迷惑なんだけど...
    - 大量にメールを配るところでは頭痛のタネ

メール経由のウイルス

- 添付ファイルが感染源であることが多い
  - マクロウイルス（Excel、Word、PowerPoint）
    - 中に忍ばせてあるOfficeオブジェクトが曲者
    - 代表は勝手にウイルスメールをばら撒くMelissa
    - 大規模MLでは添付ファイルは許可しない
  - 実行形式ファイル
    - 不用意に実行してはいけない
    - 代表はメール発信するとそのあて先にもう１通ウイルスメールを送付するHappy99

チェインメール

- 善意の協力依頼を装う（あるいは本物）
  - 「このメールを転載して下さい」が曲者
    - 無制限の転載を意図している場合には無視
    - 本来の目的を達成するには、期間や範囲を限定して一定数しか転載されない工夫を
- 不幸・幸福のメール
  - 「このメールを５人に転送しないと．．．」
    - 初心者の多い環境で流行りやすい

メール爆撃 (Bombing)

- ２種類ある
  - 巨大なサイズのメールを送付
  - 膨大な数のメールを送付
    - どちらもspoolを膨らませる結果になる
    - loopと見分けがつきにくい場合がある
- サイズ制限、通数制限等の防御
  - メーリングリストではさらに深刻な問題に
  - O MaxMessageSize=500000

知っておくべきメールアドレス

- MAILBOX NAMES FOR COMMON SERVICES, ROLES AND FUNCTIONS（RFC2142)で挙げられているもの
- 例えば、
  - abuse@example.gr.jp
    - いざという場合の問い合わせ先
  - postmaster@example.gr.jp
    - メール配送についての問い合わせ先
  - hostmaster@example.gr.jp
    - DNSについての問い合わせ先

MLの周辺アドレス

- 周辺アドレスの例
  - owner-hoe@example.gr.jp
    - sendmail的にちょっと考慮されたMLの発信者アドレス
  - hoe-admin@example.gr.jp
    - 管理者のaliasとして使われることがある
  - hoe-request@example.gr.jp
    - RFC2142的管理者アドレス
  - hoe-errorsto@example.gr.jp
    - エラーメールの専用受信アドレスを用意している場合

アドレス詐称・隠蔽問題

- bombing等では発信者アドレスが偽装される
  - SPAM発信者を偽装して発信者をbombing
- MLに他人のアドレスを登録する
  - 自動登録でConfirmなしだとアウト
- 無料メールアドレスの転送機能
  - 誰に届くかわからないという意味で曲者

まとめと展望

- SPAM対策
  - 「来たときの対策」から「出させない対策」へ
    - SMTP Authentication（RFC2554）
    - Message Submission（RFC2476）
    - SMTP over TLS（RFC2487）
  - メーリングリストではアドレス一覧を出さないこと
    - 例えばPPMLは一般参加者のwhoコマンドに対して GECOSの一覧を出す