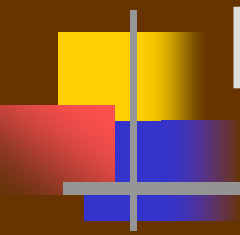
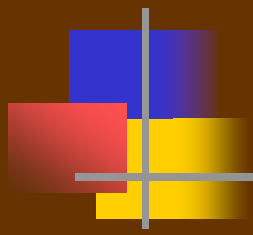


DNSを「きちんと」設定しよう

民田雅人
WIDE Project

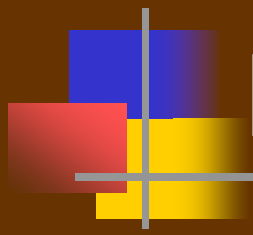
DNS DAY - Internet Week 2002



はじめに

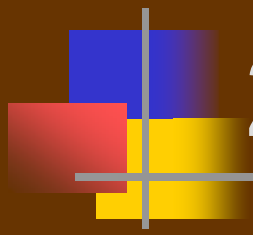
- ネームサーバーのおさらい
- ネームサーバーの設定
  - 主に、BINDで運用する場合の注意点
- セカンダリーネームサーバーを運用する上で注意点

ネームサーバーのおさらい



DNSの復習

- DNS(Domain Name System)は、
  サーバーとクライアントから成り立つ
- ネームサーバー
  - 専用のサービスプログラム
    - named(BIND), tinydns(djbdns), MicrosoftDNS(Windows), etc…
- リゾルバ
  - ライブラリ
  - サービスプログラム

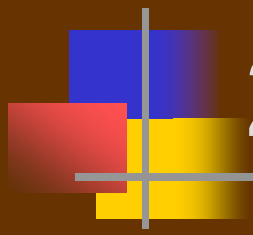


2種類のネームサーバー(1)

- 管理しているドメインへの問い合わせに答える
  - www.example.jpのIPアドレスは10.100.200.1
  - 10.20.30.40のホスト名はftp.example.jp
- 上位ドメインに登録するネームーサーバー

```
% dig example.jp ns
;; ANSWER SECTION:
example.jp.  1D IN NS  ns0.example.jp.
example.jp.  1D IN NS  ns1.example.jp.
;; ADDITIONAL SECTION:
ns0.example.jp. 1D IN A 10.10.10.10
ns1.example.jp. 1D IN A 192.168.10.10
```

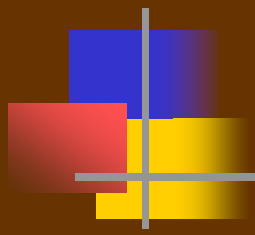
- 以下「ドメインサーバー」と呼ぶ



2種類のネームサーバー(2)

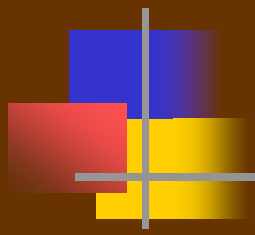
- クライアントの要求により検索を行う
  - www.example.jpのIPアドレスはいくつ?
  - IPアドレスが10.20.30.40のホスト名は?
- /etc/resolv.confでnameserverに設定
- DHCPサーバーでクライアントに配布
- 結果をキャッシュしてトラフィックを削減
- 以下「キャッシュサーバー」と呼ぶ

ドメイン・キャッシュ それぞれのサーバーの区別



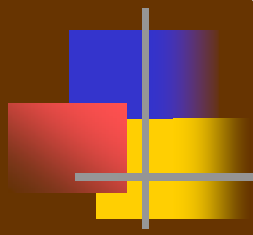
- BINDは明示的な区別が無い
  - namedで両方を兼用
  - WindowsのDNSサーバーもBINDと同様
    - BINDを元にWindows用に変更?
- djbdnsは別のプログラムとして実装
  - tinydns – ドメインサーバー
  - dnscache – キャッシュサーバー
  - 兼用はできない

ドメイン・キャッシュ兼用ネームサーバー



- BINDで極めて多い設定(Windowsも?)
  - 便利だから?
  - そういう設定例ばかりだから?
  - resolv.conf の設定がわかりやすくなるから?
- キャッシュ情報管理の問題
  - 検索結果のキャッシュによるメモリ肥大
  - キャッシュした情報がゾーンに混ざる可能性
  - ドメインサーバーに影響が出る可能性

第三者のキャッシュサーバーの不正利用



- 普通に使われて必要なドメインを検索するならほとんど問題は発生しないが…
- 不正にドメインの検索を大量に行われる
  - 負荷の増大
  - キャッシュの増大
    - BIND9ならキャッシュメモリに制限をかけられるけど…
  - プログラムの穴を突く可能性もありうる
- いずれもDOS攻撃につながる
  - 兼用の場合、ドメインサーバーへ影響

Cache Poisoning

- キャッシュサーバーへの不正情報の注入
  - example.gr.jp側でexample.jpへのいやがらせ

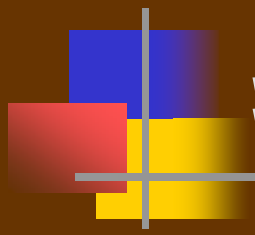
```
www.example.gr.jp  NS     ns.example.jp
ns.example.jp          A      1.2.3.4
```

  - dig @<nameserver> www.example.gr.jp

```
;; AUTHORITY SECTION:
www.example.gr.jp.    1D IN NS     ns.example.jp.
;; ADDITIONAL SECTION:
ns.example.jp.          1D IN A      1.2.3.4
```

  - 結果を鵜呑みにするキャッシュサーバー
- 古くから知られている問題
  - 過去のBINDでは確実に問題が発生する

BINDを設定する



ドメインサーバーとキャッシュサーバーは分離する

- 広く指摘されている運用TIPS
  - 「DNS and BIND」にも記述がある
- キャッシュサーバーにトラブルがあってもドメインサーバーを守る
  - あるいはその逆もある
- djbdnsではもともと分離する設計
  - セキュアなアプローチ

BINDでのドメインサーバー

- named.confで
  自ドメインのみを記述
- recursion no;
- fetch-glue no;
  - BIND9**では常に**no
- hint**情報不要**(zone “.”)
- **セカンダリは、**zone**の記述が変わるだけ**

```
options {
    ...
    recursion  no;
    fetch-glue no;
    ...
} ;
zone "example.jp" {
    type master ;
    file "example.jp.zone" ;
} ;
```

BINDでのキャッシュサーバー

- recursion yes;
- hint情報が必要
- allow-queryでアクセス制限して第三者に不正に利用させない
- 127.0.0.1 (localhost) の情報も加える。
  - ::1もお忘れなく。

```
options {
    ...
    recursion yes;
    allow-query { 10.0.0.0/8 ; };
    ...
};
zone "." {
    type hint;
    file "named.root";
};
zone "0.0.127.IN-ADDR.ARPA" {
    type master;
    file "localhost.rev";
};
```

1台でキャッシュサーバーとドメインサーバーを運用(1/3)

- namedプロセスを2つ起動する
  - 但しBIND9はv6 を有効にすると1プロセスのみ listen-on-v6 {any;}; のみ機能 → 将来修正される(?)
- ドメインサーバー用/etc/named.conf

```
options {
    ...
    recursion  no;
    fetch-glue no;
    listen-on { 10.10.10.1 ; } ;
    ...
} ;
```

- listen-onでサーバーのIPアドレスのみ
- /etc/resolv.confでは“nameserver 127.0.0.1”

1台でキャッシュサーバーとドメインサーバーを運用(2/3)

- キャッシュサーバー用/etc/cache.conf
  - named –c /etc/cache.conf で起動

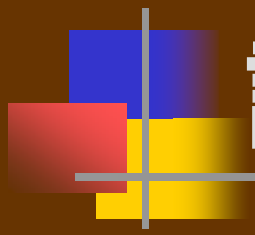
```
options {
    ...
    pid-file     "/var/run/cache-named.pid" ;
    listen-on { 127.0.0.1 ; } ;
    ...
};
controls {
    unix "/var/run/cache-ndc" perm 0600 owner 0 group 0;
} ;
```

- 127.0.0.1だけなのでアクセス制限は不要

1台でキャッシュサーバーとドメインサーバーを運用(3/3)

- dump-file, memstatistics-file, statistics-fileにも注意
  - 2つのnamedプロセスで上書きの可能性があるため一方を名前を変更する
  - 例 (BIND8の場合)

```
dump-file           "cache_dump.db" ;
memstatistics-file  "cache.memstats" ;
statistics-file     "cache.stats" ;
```



設定の確認

- dig @10.10.10.1 example.jp ns
  - ;;flags **に注目**
    - ;; flags: qr aa rd; **なら正常**
    - ;; flags: qr aa rd ra; **なら** recursion yes ; **のまま**
- dig @10.10.10.1 **<適当なドメイン名>**
  - **管理ドメイン以外は検索できないのを確認**
- dig @127.0.0.1 **<適当なドメイン名>**
  - **正常に検索できるか**
- **可能なら、自ネットワーク外から動作チェック**

やむを得ず 1プロセスのnamedで兼用

- 再帰的検索は管理対象ネットワークのみに制限
- 管理するゾーンへの問い合わせは何処からでも

```
options {
    ...
    allow-query {
        localhost ;
        10.0.0.0/8 ;
    } ;
    ...
};
```

```
zone "." {
    type hint;
    file "named.root" ;
};
zone "0.0.127.IN-ADDR.ARPA" {
    type master ;
    file "localhost.rev" ;
};
zone "example.jp" {
    type master ;
    file "example.jp.zone" ;
    allow-query { any; };
};
```

キャッシュサーバーに加えるべき逆引きゾーンの設定

- Private Address Space - RFC 1918
  - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
- IPv4 Link-Local Address
  - Dynamic Configuration of IPv4 Link-Local Addresses
    - draft-ietf-zeroconf-ipv4-linklocal-07.txt
  - 169.254.0.0/16
- **特にISPのネームサーバー担当の方は是非!**

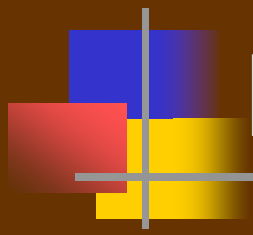
キャッシュサーバーに加えるべき逆引きゾーンの設定例

- named.conf

```
zone "10.in-addr.arpa" {
    type master; file "dummy.zone"; };
zone "16.172.in-addr.arpa" {
    type master; file "dummy.zone"; };
.......
.......
zone "31.172.in-addr.arpa" {
    type master; file "dummy.zone"; };
zone "168.192.in-addr.arpa" {
    type master; file "dummy.zone"; };
zone "254.169.in-addr.arpa" {
    type master; file "dummy.zone"; };
```

- dummy.zone
  - SOAとNSを記述
  - 他は不要

```
$TTL 1D
@   IN SOA ns.example.jp.
                    root.example.jp. (
            1
            1H
            15M
            1W
            1D )
    IN NS  ns.example.jp.
```

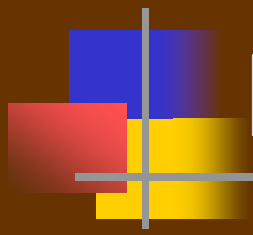


BINDをよりセキュアに(1/2)

- ゾーン転送可能なホストを制限する
  - セカンダリー以外は転送できないように
  - optionsやzoneにallow-transfer

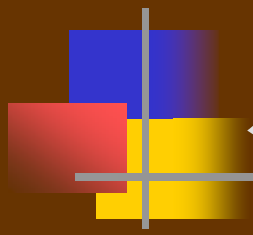
```
zone “xxx.zone” {
... allow-transfer { x.x.x.x ; y.y.y.y ; };
};
```

  - BIND8まではゾーン転送はforkで実装されている
- named –u bind ……
  - bind というnamed専用ユーザーを用意し その権限で起動する
  - 万が一、named経由で侵入されたときへの備え



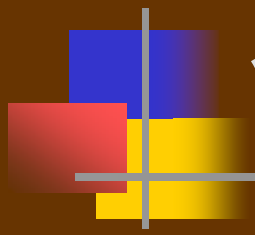
BINDをよりセキュアに(2/2)

- chroot環境でnamedを動かす
  - named経由で侵入されたときへの備え
- named –t <chrootディレクトリ>
  - 設定は少々手間がかかる
  - Rob‘s DNS Data Page http://www.cymru.com/DNS/ の Secure BIND Template に設定例
- djbdnsはchroot環境下で動作する



ルータでのacl や IDS

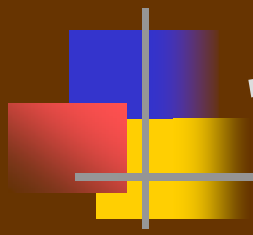
- ネームサーバーへのacl
  - 設定するのはかまわないけど…
  - 動作が妨げられない程度に
- IDS
  - 正常なパケット侵入と検出したりしない
  - 語検出によって、しなくてもいい問い合わせ
- 生半可な設定は世間へ迷惑
  - 設定した本人も余計なコストがる



ゾーンファイル

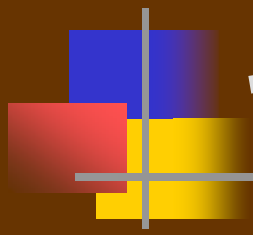
- 一般的な注意
  - SOAの各フィールドの値は適正か
  - NSの値の指すものがCNAMEになっていたりしないか
  - MXの値の指すものがCNAMEになっていたりしないか
  - MXの値がIPアドレスになっていたりしないか
  - 不必要にCNAMEを多用してないか?
- NS RRに記述してあるドメインは、上位に登録しているドメインと一致しているか
  - 不一致でも動作することも多いが、不具合になることもある
- 自ドメイン外の情報を記述していたりしないか

セカンダリーネームサーバー



セカンダリーの誤解

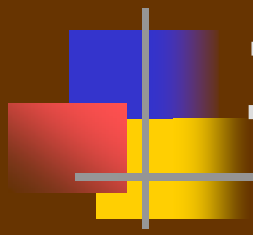
- セカンダリーネームサーバーは
  プライマリーネームサーバーが
  停止したときのための予備である
- プライマリネームサーバーが動いてる間は
  セカンダリーネームサーバーは働かない
- 大きな誤り!



セカンダリーは予備では無い!

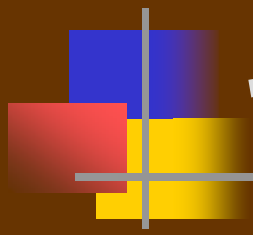
- 稼動中はマスターネームサーバーと同じ
  - 常時マスターネームーサーバーと同じデータを保持する
  - 常時クライアントからの問い合わせがある
- プライマリとの違いはゾーンデータをプライマリからコピーすること
  - ゾーン転送

よくあるモデル
専用線接続のエンドユーザー



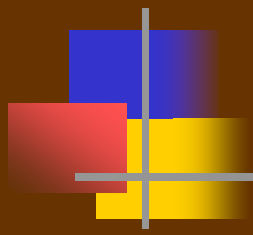
専用線にトラブル発生!

- 専用線(or その接続ルータ)トラブル
- www.example.jpへのアクセス不能
- ns.example.jpへのアクセス不能
- ns.provider.domがあるので
  www.example.jpのIPアドレスは検索可能
  - しかしながら、その意義は?



セカンダリーの効果的配置

- 「セカンダリーネームサーバーは絶対必要」というのはウソである
  - もちろんプライマリーを一時停止する場合には役立つ
- 用意するなら違うネットワークに配置する
  - 純粋に負荷分散目的なら同一ネットワークもありうる
- 第三者(接続先プロバイダ等)に任せるなら十分信頼できるところへ
  - プライマリがセキュアでも、セカンダリが...
- セカンダリサーバーの情報が正常かどうかを、必ず確認する



まとめ

- 「なんとなく動いてる」ではいけません
- 古き良き時代は終わりました
  - アクセス制限しなくても問題は発生しない
  - キャッシとドメインサーバー兼用している
    - メールサーバーのオープンリレーと同じ話
- 過去の遺物に頼っていてはいけない
- 今一度、自分の管理してる
  ネームサーバーの点検を