

小さなネットワークの お手軽DNS管理ツール

2025-06-27

民田 雅人

ぶらっとホーム

DNS Summer Day 2025

About me

民田 雅人(みんだまさと) aka. みんな

- 2014年まではドメイン名レジストリの会社に所属
- 2015年にふらっとホーム(コンピューター系製造業)に転職

転職して初期の仕事で、あるお客様向け専用 Dynamic DNSサービスを構築

- BIND 9のDynamic DNSってちゃんと動く(もちろん知ってた 😊)

還暦過ぎて、初めてSQLでCREATE TABLE文使いました ⇒ 初めてのDB構築

- SQLとの付き合いは簡単なSELECT文ぐらいなら使ったことある程度
- Oracleのインストールや起動停止、HDDのチューニングなんかは経験有

Githubの [tzdiff](#) にStarをつけていただいた皆様、ありがとうございます

ぷらっとホームにやってきて

当初は社内ネットワークでも、更にNATルータの裏側に住んでいた

- 当時のOpenBlocks(ぷらっとホーム社の小型Linuxサーバー)はデフォルトでDHCPサーバーが動くので、一般社員のネットワークに同居するのは危険
- 当初は自分の管理しているサーバーは無かった

ある日NATルーターやめて普通のルーターに切り換えた

- 自分の管理しているサーバーもある
- 社内側から自分のサーバーへ名前でアクセスしたい
- 実験機(主にOpenBlocks)にも名前でアクセスしたい
- 接続して(動いて)いる機器を把握したい ⇒ ちゃんと逆引き欲しい

DNSサーバー用意しないと

DNSサーバーは必要だけど DNSの管理(ホストの追加削除)はめんどくさい

ホストが増えたり減ったりすると、その都度

- ゾーンデータへのRR(リソースレコード)の追加や削除
- SOAのシリアルの更新 (忘れやすい)
- ゾーンデータのreload
- さらに逆引きでも同作業
⇒ ピリオド忘れる問題も付きまとう

正直言って登録内容の割に手数が多く
機器がちよくちよく入れ替わる環境では
「いちいちDNS登録なんてやってられるか！」
になりがち

実験機器が並んでる様子



ホストとIPアドレスのDNS登録

obs1.example.jpで192.0.2.30をDNS登録するにはAとPTR RRを用意する

A (正引き)

```
obs1.example.jp.      86400   IN  A      192.0.2.30
```

PTR (逆引き)

```
30.2.0.192.in-addr.arpa. 86400   IN  PTR    obs1.example.jp.
```

- ホスト名とIPアドレスが決まればRRは機械的に生成できる
- nsupdateを使えばDynamic DNSでDNSへ登録できる
- Dynamic DNSを使うとSOAのSerial管理から開放される

ひょっとして...

手間のかかるDNS登録も
Dynamic DNSを使えば楽できそう

つまり

全てDynamic DNSで管理すればいい

Dynamic DNSでの登録削除のためのツールを作る

Dynamic DNSで登録

```
phddns-add host1 192.0.2.30
```

Dynamic DNSで削除

```
phddns-del host1
```

正引きと逆引きをnsupdateコマンド経由で登録・削除するシェルスクリプト

- 当時のバージョンは社内の自ネットワークのサブドメインとネットワークアドレスの /24 1個だけをサポート

これらのコマンドのおかげで簡単にDNS更新ができるようになる

⇒ 当初の目的は達成！

数年経過して...

社内の他のサブドメイン・ネットワーク対応が必要になる

hostname.subdomain形式で、社内のサブドメイン一式に対応

```
phddns-add host1.subC 198.51.100.50
```

```
phddns-add host2.subB 203.0.113.70
```

```
phddns-del host3.subA
```

この時点ではプログラムに対応ドメイン名やネットワークが埋め込み

⇒ ドメイン名やネットワークを増やすにはコードを修正する必要がある

他のドメイン名でも使いたい！

⇒ 頑張ればなんとかかなりそう

⇒ 頑張りました

プログラムと設定を分離して

ddns-utils

<https://github.com/belgianbeer/ddns-utils>

できました

ddns-utilsではこうなってます その1

ホストの追加: `ddns-add obs1.example.jp 192.0.2.30`

- AレコードとPTRレコードを登録
- `obs1.example.jp`が既に登録済なら、同時に正引き・逆引きとも削除
つまりデフォルトではIPアドレスの新規登録または変更
- `-a`を指定すると削除は実行しない (2nd IPアドレスの追加)
- 正引きのIPアドレスは管理外のネットワークでも登録可

ホストの削除: `ddns-del obs1.example.jp`

- `obs1.example.jp`の正引き・逆引きとも削除する
- もちろん管理外ネットワークの逆引きは削除できません

ddns-utilsではこうなってます その2

設定ファイルddns.confに、1ゾーンあたり5つのシェル変数を設定する

KEYWORD_zone	対象のゾーン名を指定
KEYWORD_nsrv	そのゾーンを管理している権威サーバー
KEYWORD_dttl	デフォルトのTTL
KEYWORD_keyu	Dynamic DNSのアップデート用TSIGキーのファイル
KEYWORD_keyt	権威サーバーからゾーン転送する際のTSIGキーのファイル

- TSIGを使っていない場合、KEYWORD_keyuやKEYWORD_keytは省略可
- " _ "に続くキーワードは可読性向上のため4文字で統一

ddns-utilsではこうなってます その3

KEYWORDは何を設定するのか

ドメイン名の場合(正引き)

⇒ ドメイン名の"."(ピリオド)を"_"(アンダースコア)に置換する

example.jp ⇒ example_jp

IPアドレスの場合(逆引き)

⇒ オクテット区切りの"."を"_"に置き換えて、先頭に"ipv4_"をつける

192.168.37.0/24 ⇒ ipv4_192_168_37

172.23.0.0/16 ⇒ ipv4_172_23

ddns-utilsではこうなってます その4

管理対象のキーワードをシェル変数に設定する

ドメイン名2個 example.jp example.org
 `db_zone_list=" example_jp example_org "`

IPアドレス2種類 192.168.37.0/24 172.23.0.0/16
 `db_ipv4_list=" ipv4_192_168_37 ipv4_172_23 "`

設定ファイルddns.confは本体(ddns-add等)と同じディレクトリに置く

- /etcとか/usr/local/etcとかに置くことは考えていない
- そもそも管理者しか使わないしTSIGキーへのアクセスも制限が必要
⇒ 管理者が~/binに置いて使うのを想定している

ddns-add を使ってみる (新規追加)

```
$ ddns-add obs1.example.jp 192.168.37.10
server ns0.example.jp
zone example.jp
update add      obs1.example.jp.      86400  IN      A      192.168.37.10
send
server ns0.example.jp
zone 37.168.192.in-addr.arpa
update add      10.37.168.192.in-addr.arpa. 86400  IN      PTR
obs1.example.jp
send
$
```

- デフォルトではnsupdateの内容を表示するが-qで抑止できる
- -nでnsupdateを実行せず表示のみ
- 他に-tでTTLを個別に変更できる(ようにしたい、現時点で未実装😅)

ddns-add を試してみる (既存の変更)

```
$ ddns-add obs1.example.jp 192.168.37.20
server ns0.example.jp
zone 37.168.192.in-addr.arpa
update delete 10.37.168.192.in-addr.arpa. 86400 IN PTR obs1.example.jp.
send
server ns0.example.jp
zone example.jp
update delete obs1.example.jp. 86400 IN A 192.168.37.10
update add obs1.example.jp. 86400 IN A 192.168.37.20
send
server ns0.example.jp
zone 37.168.192.in-addr.arpa
update add 20.37.168.192.in-addr.arpa. 86400 IN PTR
obs1.example.jp
send
```

ddns-del を試してみる

```
$ ddns-del obs1.example.jp
server ns0.example.jp
zone example.jp
update delete obs1.example.jp. 86400 IN A 192.168.37.20
send
server ns0.example.jp
zone 37.168.192.in-addr.arpa
update delete 20.37.168.192.in-addr.arpa. 86400 IN PTR obs1.example.jp.
send
$
```

ゾーンデータのバックアップはどうする？

ddns-backupを実行すると、カレントディレクトリに管理対象ゾーンのデータを一式ダウンロードする。ちなみに実装はこんな感じ

```
dig ゾーン名 axfr | sort -u --version-sort > ゾーン名
```

```
$ ls
$ ddns-backup
$ ls
23.172.in-addr.arpa      example.jp
37.168.192.in-addr.arpa example.org
$
```

個別に見るだけなら ddns-zone で (dig ゾーン名 axfr をそのまま出力)

- ddns-zone example.jp
- ddns-zone 192.168.37.0

IPアドレスとホストの一覧(/etc/hosts形式)を見たい

ddns-hostsで一覧を確認できる

```
$ ddns-hosts example.jp
127.0.0.1      ns0.example.jp
192.168.37.100 obs100.example.jp
192.168.37.101 obs101.example.jp
192.168.37.102 obs102.example.jp
192.168.37.103 obs103.example.jp
192.168.37.104 obs104.example.jp
192.168.37.105 obs105.example.jp
192.168.37.106 obs106.example.jp
192.168.37.107 obs107.example.jp
192.168.37.108 obs108.example.jp
192.168.37.109 obs109.example.jp
$
```

NSとかTXT等他のRRを編集するにはどうするの？

nsupdateを直接使えばよく、SOAやNS等も変更できる

SOAのシリアルを9999に変更する

```
$ nsupdate -k ~/.tsig/example.jp-update.key
server ns0.example.jp.
zone example.jp.
update add example.jp. 86400 IN SOA ns0.example.jp. minmin.example.jp. 9999 3600 300 3600000 300
send
$
```

メモ: namedはDynamic DNS経由の異常な更新をチェックする

- SOAを削除する、シリアル値を古くする、NSを全て削除する等はできない ⇒ 無視あるいは拒否となる

汎用に使える ddns-add-raw、ddns-del-raw みたいなものがあるといいのかも

ddns-utilsではこうなってます その5

```
$ ls -l ddns-*  
-rwxr-xr-x  1 minmin minmin 7413 Jun 24 10:16 ddns-add  
lrwxr-xr-x  1 minmin minmin   8 Jun 24 10:16 ddns-backup -> ddns-add  
lrwxr-xr-x  1 minmin minmin   8 Jun 24 10:16 ddns-del -> ddns-add  
lrwxr-xr-x  1 minmin minmin   8 Jun 24 10:16 ddns-hosts -> ddns-add  
lrwxr-xr-x  1 minmin minmin   8 Jun 24 10:16 ddns-zone -> ddns-add  
$
```

- ddns-addがプログラム本体で、他はddns-addへのシンボリックリンク
- makeの実行でシンボリックリンクを作成 (別途Makefile有り)
- FreeBSDならsh (/bin/sh) 、Linuxならdash (or ash?)のシェルスクリプト
⇒ BashやZsh依存無し

ddns-utilsのデモ

デモ環境のddns.conf

```
$ cat ddns.conf
db_zone_list=" example_jp example_org "
db_ipv4_list=" ipv4_192_168_37 ipv4_172_23 "

example_jp_zone=example.jp
example_jp_nsrv=ns0.example.jp
example_jp_dttl=86400
example_jp_keyu=~/.tsig/example.jp-update.key
example_jp_keyt=~/.tsig/example-transfer.key

example_org_zone=example.org
example_org_nsrv=ns0.example.jp
example_org_dttl=86400
example_org_keyu=~/.tsig/example.org-update.key
example_org_keyt=~/.tsig/example-transfer.key
$
```

ところで「小さなネットワーク」とは？

dns-addやdns-delの処理で気軽(?)にゾーン転送を利用しているため、ゾーンデータが大きいと転送時間と負荷が無視できなくなる

- RR数の合計が数千個程度であれば今時問題にはならない(多分)
⇒ 少なくとも2000個程度では実用上問題無し
- 今のままではRR数で1万を超えてくると厳しいんじゃないかな
- プログラムを工夫すればゾーン転送は減らせるのかも
⇒ 削除時に逆引き側に残っても良いなら...

他にも複数ゾーンをシーケンシャルサーチ

- シェルスクリプトなので仕方ない ⇒ ゾーン数が増えると厳しい

ddns-utilsとDynamic DNSによるDNS管理

メリット: 従来のゾーンファイル管理に比べて

- ホストの登録と削除が簡単になりSOAのシリアル管理からも開放された
- nsupdateを直接使う場合も差分だけを扱うので意外に悪くない
- Let's EncryptのDNS認証時にTXT RRの手作業が不要になった
⇒ acme.shでdns_myapi.shをDynamic DNS環境に合わせる

デメリット: 従来のゾーンファイル管理に比べて

- 特に無し(個人の感想です)

今後の機能追加等

- IPv6対応: やればできるはずだけど、どう実装すべきかは要検討
- /24より小さいアロケーションでの逆引き: 汎用的実装は多分無理
⇒ 正引きはddns-add、逆引きはnsupdateが正解

おわりに

現時点のddns-utilsは
主に必要な機能を実装した段階

バグ修正や機能追加(IPv6 / DNSSEC)等
皆様からのコントリビューションを
お待ちしております