DIY RFC2136 dyndns with bind

ever since dyndns stopped to be completely free (including hassle-free) i was looking for alternatives. i recently stumbled across RFC2136 which can be used to provide dynamic dns services. since i have access to two nameservers running bind i decided to try it out.. it works pretty nicely :)

there is a howto in the pfsense wiki, however, that did not work for me. i had to use allow-update reather than update-policy.. don't know why, somehow it just seemed to have been ignored by the version of bind9 i am running on the servers. I've used in general the setup described in this very detailed webpage about dyndns with bind9.

this following howto will explain how i did my setup so that i could have a little bash script that would allow me to add new hosts to my dyndns with a single command. all my hosts will end with .dyn.mydomain.ch.

i can run

```
/etc/bind/dyn/add_new_host.sh myhost
```

and it will add a new host called myhost.dyn.mydomain.ch to the configuration and return an authorization key which i can use on the client side.

so here is how i did it:

first of all i wanted to be able to have a simple script that would allow me to add new hosts with a minimum amount of work. so i split my config into different files, so i could later edit them automatically. also, you want to make sure the file where the keys are stored is not world readable..

- create a directory that holds all the dynamic dns stuff:

  ```
  mkdir /etc/bind/dyn
  cd /etc/bind/dyn
  ```

- create a basic zonefile for the dynamic dns zone. **Important**, you should use a dedicated subdomain with its own zone file for the dyndns stuff, as the zone file will be rewritten by bind later on and after that it is an absolute mess. so make sure you don't do this with your main zone file for your main domain! save the file as db.dyn.mydomain.ch in your dyn directory. here are the contents:

  ```
  $ORIGIN .
  $TTL 30 ; 30 seconds
  dyn.mydomain.ch     IN SOA    ns1.mydomain.ch. hostmaster.mydomain.ch.
  (
                  2013102704 ; serial
                  900        ; refresh (15 minutes)
                  600        ; retry (10 minutes)
                  604800     ; expire (1 Week)
                  30         ; minimum (30 seconds)
                  )
              NS    ns1.mydomain.ch.
  ```

```
            NS    ns3.mydomain.ch.
```

- create an empty keys.conf file

```
touch keys.conf
```

- create a file named.conf with the following contents

```
include "/etc/bind/dyn/keys.conf";

zone "dyn.mydomain.ch" {
    type master;
    file "/etc/bind/dyn/dyn.mydomain.ch";
    allow-update {
        //add_keys_here//
    };
    allow-query {
        ANY;
    };
};
</code bash add_new_host.sh> **note** keep the //add_keys_here//
comment exactly as it is, this is the marker for our script so it knows
where to add new keys
  * edit your main named.conf file, usually in /etc/bind/named.conf and
add an include line at the end of your zone definitions like so:
<code>include "/etc/bind/dyn/named.conf";
```

- create the “add_new_host.sh” script that will add new hosts to our setup. here are the contents of the script:

```
#!/bin/bash
if [ -z "$1" -o "$1" == " " ]; then
        echo "usage: add_new_host.sh <hostname>"
        echo "EXAMPLE: add_new_host.sh myhost will add
myhost.dyn.mydomain.ch"
        exit 1
fi
cd /etc/bind/dyn/
mkdir tmp
cd tmp
hostname=${1}.dyn.mydomain.ch.
echo "generating key for ${hostname}"
keyfile=`dnssec-keygen -a HMAC-MD5 -b 128 -n HOST ${hostname}`
key=`grep "Key" ${keyfile}.private | awk '{ print $2; }'`
echo "here is the key i have generated, use this to configure your
client: $key"
cd ..
rm -rf tmp
echo "adding key to named.conf..."
cat named.conf | sed -e "s/\/\/add_keys_here\/\//key
${hostname};\n\t\t\/\/add_keys_here\/\//" | tee named.conf > /dev/null
```

```
echo "key ${hostname} { algorithm hmac-md5; secret \"${key}\"; };" >>
keys.conf
echo "done"
echo "reload bind";
/etc/init.d/bind9 reload
echo "currently active hosts:"
grep "key " named.conf | awk '{ print $2; }' | tr -d ";"
```

- now set the permissions so that especially the keys.conf file is only readable by bind and editable by root. also the dyn directory must be writeable by bind or if you don't want that, touch a file called dyn.mydomain.ch.jnl and make it writeable for bind, as well as making the dyn.mydomain.ch file writeable for bind. here is how i've set the permissions on my server:

```
drwxrwxr-- 2 root bind 4096 Oct 29 13:45 ./
drwxr-sr-x 3 root bind 4096 Oct 29 11:47 ../
-rwx------ 1 root root  904 Oct 29 13:45 add_new_host.sh*
-rw-r--r-- 1 bind bind  434 Oct 29 13:20 dyn.mydomain.ch
-rw-r--r-- 1 bind bind 1230 Oct 29 13:15 dyn.mydomain.ch.jnl
-rw-r----- 1 root bind  356 Oct 29 13:45 keys.conf
-rw-r--r-- 1 root bind  322 Oct 29 13:45 named.conf
```

- now use the script to add your first hostname.

```
./add_new_host myhost
```

  if you did everything correctly (and if i described it all correctly) your client should now be able to update it's own dns entry with the key you received back from the script.

script to remove hosts

optionally you can also create a little script to remove hosts just as easily. create a file called remove_hosts.sh with the following contents

```
#!/bin/bash
if [ -z "$1" -o "$1" == " " ]; then
        echo "usage: remove_host.sh <hostname>"
        echo "EXAMPLE: remove_host.sh myhost will remove
myhost.dyn.mydomain.ch"
        exit 1
fi
cd /etc/bind/dyn/
hostname=${1}.dyn.mydomain.ch.
echo "old keys.conf entry: "
grep ${hostname} keys.conf
echo "remove key for ${hostname}"
cat named.conf | sed -e "/^\t\tkey ${hostname}.*$/d" | tee named.conf >
/dev/null
cat keys.conf | sed -e "/^key ${hostname}.*$/d" | tee keys.conf > /dev/null
echo "reload bind";
```

```
/etc/init.d/bind9 reload
echo "currently active hosts:"
grep "key " named.conf | awk '{ print $2; }' | tr -d ";"
```

make it executable and run it to remove hotsts. **warning** make a backup of your keys.conf and your named.conf file before testing this :)

```
./remove_host.sh myhost
```