By Falko Timme

Published: 2008-01-22 17:54

How To Install And Use The djbdns Name Server On Debian Etch

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 01/15/2008

djbdns is a very secure suite of DNS tools that consists out of multiple parts: dnscache, a DNS cache that can be used in *`/etc/resolv.conf`* instead of your ISP's name servers and that tries to sort out wrong (malicious) DNS answers; axfrdns, a service that runs on the master DNS server and to which the slaves connect for zone transfers; and tinydns, the actual DNS server, a very secure replacement for BIND.

I do not issue any guarantee that this will work for you!

1 Preliminary Note

I have tested djbdns on a Debian Etch system with the IP address *`192.168.0.100`*. I'll explain how to use dnscache and tinydns (as a master DNS server), but not how to use axfrdns - maybe I'll cover that in another tutorial.

dnscache will listen on the local IP address *`127.0.0.1`*, tinydns on the external IP address *`192.168.0.100`*.

2 Installing djbdns

djbdns is not available as a binary package in the Debian repositories due to its "license" (until December 28, 2007, djbdns was license-free software), however there's a *`djbdns-installer`* package in the repositories that can be used to install djbdns. djbdns depends on daemontools and ucspi-tcp; again, there are only installer packages available for these programs. The installers are available in the Debian Etch *`contrib`* and *`non-free`* repositories, so we must make sure first that these are included in our *`/etc/apt/sources.list`*:

```
vi /etc/apt/sources.list
```

```
[...]
deb http://ftp2.de.debian.org/debian/ etch main contrib non-free
[...]
```

Update your packages database afterwards:

```
apt-get update
```

Next we install the *daemontools-installer*:

```
apt-get install daemontools-installer
```

Now we can install the daemontools like this:

```
build-daemontools
```

You will be asked a few questions. You can always accept the default value by pressing *ENTER*:

```
Enter a directory where you would like to do this [/tmp/daemontools] <-- ENTER

Which format would you like to use? [fD] <-- ENTER

Press ENTER to continue... <-- ENTER

Do you want to remove all files in /tmp/daemontools,
  except daemontools_0.76-9_i386.deb now? [Yn] <-- ENTER
```

Do you want to install daemontools_0.76-9_i386.deb now? [Yn] <-- ENTER

Do you want to purge daemontools-installer now? [yN] <-- ENTER

To install *ucspi-tcp*, we run

```
apt-get install ucspi-tcp-src
```

and then:

```
build-ucspi-tcp
```

You'll be asked a few questions again, and again you can accept the default values:

Enter a directory where you would like to do this [/tmp/ucspi-tcp] <-- ENTER

Press ENTER to continue... <-- ENTER

Do you want to remove all files in /tmp/ucspi-tcp,
except ucspi-tcp_0.88-10_i386.deb now? [Yn] <-- ENTER

Do you want to install ucspi-tcp_0.88-10_i386.deb now? [Yn] <-- ENTER

Do you want to purge ucspi-tcp-src now? [yN] <-- ENTER

Finally we install djbdns as follows:

```
apt-get install djbdns-installer
```

```
build-djbdns
```

Again, you'll be asked a few questions - accept the default values:

```
Enter a directory where you would like to do this [/tmp/djbdns] <-- ENTER

Press ENTER to continue... <-- ENTER

Do you want to remove all files in /tmp/djbdns,
  except djbdns_1.05-11_i386.deb now? [Yn] <-- ENTER

Do you want to install djbdns_1.05-11_i386.deb now? [Yn] <-- ENTER

Do you want to purge djbdns-installer now? [yN] <-- ENTER
```

Next we configure dnscache, axfrdns, and tinydns (make sure you replace `192.168.0.100` with the external IP address of your system):

```
mkdir /var/lib/svscan

dnscache-conf dnscache dnslog /var/lib/svscan/dnscache

axfrdns-conf axfrdns dnslog /var/lib/svscan/axfrdns /var/lib/svscan/tinydns 192.168.0.100

tinydns-conf tinydns dnslog /var/lib/svscan/tinydns 192.168.0.100
```

```
ln -s /var/lib/svscan/dnscache /service

ln -s /var/lib/svscan/axfrdns /service

ln -s /var/lib/svscan/tinydns /service
```

Then we start djbdns:

```
/etc/init.d/djbdns restart
```

3 Using dnscache

To use dnscache, we replace the existing name servers in `/etc/resolv.conf` with `127.0.0.1`, the IP address that dnscache is listening on.

Make a backup of `/etc/resolv.conf`:

```
cp /etc/resolv.conf /etc/resolv.conf-original
```

Then run the following commands to create a new `/etc/resolv.conf` (make sure you replace `example.com` with your own domain):

```
echo "domain example.com" > /etc/resolv.conf

echo "nameserver 127.0.0.1" >> /etc/resolv.conf
```

To test if dnscache is working, we can try to resolve a hostname, e.g. `www.google.com`:

```
dnsip www.google.com
```

If all goes well, it should display the IP addresses of `www.google.com`:

```
server1:~# dnsip www.google.com
  66.249.93.104 66.249.93.147 66.249.93.99
server1:~#
```

4 Configuring tinydns

All tinydns records are stored in the file `/service/tinydns/root/data`. This file can either be edited by hand, or you can use some helper scripts that are in the `/service/tinydns/root` directory, e.g. `add-ns`, `add-host`, `add-alias`, etc.

I will now create some records for the domain `example.com` using these helper scripts. To use these helper scripts, we must go to the `/service/tinydns/root` directory:

```
cd /service/tinydns/root
```

Now I want this server (`192.168.0.100`) to be a name server for the `example.com` domain, so I run:

```
./add-ns example.com 192.168.0.100
```

The name of the name server is not directly specifiable. Names are automatically assigned by `add-ns` itself, following the pattern `[a-z].ns.name`, i.e. the `192.168.0.100` name server is named `a.ns.example.com` (you don't have to create an A record for `a.ns.example.com`, this has been created automatically by the previous `add-ns` command).

Now let's make the server with the IP address `192.168.0.101` our second name server for the `example.com` domain - this is `b.ns.example.com`:

```
./add-ns example.com 192.168.0.101
```

Next let's create A records for the servers that will host `example.com` - let's name them `server1.example.com` and `server2.example.com`:

```
./add-host server1.example.com 192.168.0.100

./add-host server2.example.com 192.168.0.101
```

A single IP address can be used only once in an add-host command. To create further hostnames that use the IP address, we must now use the `add-alias` command:

```
./add-alias www.example.com 192.168.0.100
```

```
./add-alias example.com 192.168.0.100
```

Let's make `192.168.0.100` the mail exchanger for `example.com`:

```
./add-mx example.com 192.168.0.100
```

The name of the SMTP server is not directly specifiable. Names are automatically assigned by `add-mx` itself, following the pattern `[a-z].mx.name`, in this case `a.mx.example.com`. It is not possible to specify the distance value (i.e., the priority) for the SMTP server.

After you've created all wanted records, you must run

```
make
```

so that your changes can take effect.

There are no helper scripts to create CNAME and TXT records (e.g. for SPF records), so if you want to create such records, you must modify `/service/tinydns/root/data` manually, e.g. like this:

```
vi /service/tinydns/root/data
```

```
[...]
'example.com:v=spf1 a mx ~all:3600
Cftp.example.com:www.example.com
```

You can use the SPF wizard on **http://old.openspf.org/wizard.html** to create an SPF record for your domain - the wizard shows the record in BIND and tinydns syntax so that you can copy & paste the record.

Don't forget to run

```
make
```

afterwards.

If you take a look at the `/service/tinydns/root/data` file...

```
cat /service/tinydns/root/data
```

```
server1:/service/tinydns/root# cat /service/tinydns/root/data
  .example.com:192.168.0.100:a:259200
  .example.com:192.168.0.101:b:259200
  =server1.example.com:192.168.0.100:86400
  =server2.example.com:192.168.0.101:86400
  +www.example.com:192.168.0.100:86400
  +example.com:192.168.0.100:86400
  @example.com:192.168.0.100:a::86400
  'example.com:v=spf1 a mx ~all:3600
  Cftp.example.com:www.example.com
server1:/service/tinydns/root#
```

... you'll notice that the records begin with signs such as `.`, `=`, `+`, `@`, `'`, `C`, etc. You can find explanations of the different record types on **http://www.fefe.de/djbdns/#recordtypes** and **http://www.pjvenda.org/linux/doc/tinydns/**.

Instead of using the `add-*` helper scripts, you can of course specify all records manually in `/service/tinydns/root/data`. This way you are more flexible, for example you can assign individual names to your name servers and mail exchangers, e.g. `ns1.example.com` instead of `a.ns.example.com`:

```
cd /service/tinydns/root

vi data
```

```
#define the authoritative nameserver
.example.com::ns1.example.com
#mail exchanger
@example.com::mail.example.com
#IP for machine1,2,3,4,5
=machine1.example.com:1.2.3.1
=machine2.example.com:1.2.3.2
=machine3.example.com:1.2.3.3
=machine4.example.com:1.2.3.4
=machine5.example.com:1.2.3.5
#machine5 is also known as ns1
+ns1.example.com:1.2.3.5
#machine1 is our mailserver
+mail.example.com:1.2.3.1
#and our webserver
+www.example.com:1.2.3.1
```

```
make
```

To test your records, you can use the dig command, e.g.

```
dig @192.168.0.100 example.com
```

```
dig @192.168.0.100 ns example.com
```

```
dig @192.168.0.100 mx example.com
dig @192.168.0.100 txt example.com
```

```
dig @192.168.0.100 www.example.com
```

etc.

To learn more about djbdns, you should definitely take a look at the following web sites:

- **http://cr.yp.to/djbdns.html**
- **http://www.tinydns.org**
- **http://www.lifewithdjbdns.com**
- **http://www.djbdnsrocks.org/**
- **http://www.fefe.de/djbdns/#recordtypes**
- **http://www.pjvenda.org/linux/doc/tinydns/**
- **http://smarden.org/pape/djb/manpages/**