Installation and configuration reminder for my RaspBerry Pi Model B as DNS ❗ Doesn't work, thanks to systemd that doesn't really like static IPs → 🗑️ Delete! It is now fully working under Debian!

Overview

To strengthen and add some fun at home I build a DNS out of the RaspBerry Pi.

❗ This is for home use only, this dns is not resolved from outside ❗

Here's the output of lsusb

```
lsusb
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514
Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9512 Standard Microsystems Corp. LAN9500
Ethernet 10/100 Adapter / SMSC9512/9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Here's the output of cpuinfo

```
Processor       : ARMv6-compatible processor rev 7 (v6l)
BogoMIPS        : 697.95
Features        : swp half thumb fastmult vfp edsp java tls
CPU implementer : 0x41
CPU architecture: 7
CPU variant     : 0x0
CPU part        : 0xb76
CPU revision    : 7

Hardware        : BCM2708
Revision        : 000e
```

Partition table

Disk	Partition	Name	Label	Size	Format	Comment
MMC	1	mmcblk0p1	/boot	95MB	VFAT	Boot partition
MMC	5	mmcblk0p5	/	64GB	ext4	Root

Base install

References

- Installation guide + iso
- Raspberry Pi

First install

Check the archlinuxarm.org page for latest img

```
dd if=archlinux-hf-2013-07-22.img of=/dev/mmcblk0 bs=1M
```

Extend the root partition with the free space: gparted 🙂

Move the MMC card to the Raspberry Pi

Remote access

Headless (for my case) so I check on the DHCP for a newcomer.
SSH to it

```
ssh root@192.168.1.156
```

Default password is root

Basic configuration

First change this root password

```
passwd
```

System Upgrade

Simple as

```
pacman -Syu
```

Change Vi to Vim

```
pacman -S vim
```

```
rm /usr/bin/vi &&  ln -s /usr/bin/vim /usr/bin/vi
```

LOCALE

```
vi /etc/locale.gen
```

I'll use en_US.utf-8/iso8859 so uncomment:

```
en_US.UTF-8 UTF-8
en_US ISO-8859-1
```

Then generate locales

```
 locale-gen
```

Configure them Locale:

```
vi /etc/locale.conf
```

```
LANG="en_US.UTF-8"

# Keep the default sort order (e.g. files starting with a '.'
# should appear at the start of a directory listing.)
LC_COLLATE="C"
```

Configure the console (in case of direct connection): Default keyboard in console (US variant international ... with a different mapping than on X11! Well done)

```
echo "KEYMAP=us-acentos" > /etc/vconsole.conf
```

TIMEZONE

```
ln -s /usr/share/zoneinfo/Europe/Paris /etc/localtime
```

Network configuration

Let's change the hostname

```
 echo dns1.home > /etc/hostname
```

I will use a static netctl config
Reference: https://wiki.archlinux.org/index.php/Netctl

```
cp /etc/netctl/examples/ethernet-static /etc/netctl/.
vi /etc/netctl/ethernet-static
```

```
Description='A basic static ethernet connection'
Interface=eth0
Connection=ethernet
IP=static
Address=('192.168.1.10/24')
#Routes=('192.168.0.0/24 via 192.168.1.2')
Gateway='192.168.1.1'
#DNS=('192.168.1.2')
DNS=(127.0.0.1')
SkipNoCarrier=yes
ExecUpPost='/usr/bin/ntpd -q || true'

## For IPv6 autoconfiguration
#IP6=stateless

## For IPv6 static address configuration
#IP6=static
#Address6=('1234:5678:9abc:def::1/64' '1234:3456::123/96')
#Routes6=('abcd::1234')
#Gateway6='1234:0:123::abcd'
```

Disable dhcpcd on eth0

```
systemctl disable dhcpcd@eth0
```

Make it default

```
netctl enable ethernet-static
```

```
netctl start ethernet-static
```

:!:A nice reboot is mandatory ... didn't find a way to reload network configuration

Some tools

```
pacman -S screen bash-completion glances htop lftp rsync wget curl bc lsof
strace base-devel
```

Select all base-devel tools even if some are already in place

NTP

Reference: https://wiki.archlinux.org/index.php/Ntp

```
pacman -S ntp
```

Normally already installed and started

```
vi /etc/ntp.conf
```

```
server 0.fr.pool.ntp.org iburst
server 1.fr.pool.ntp.org iburst
server 2.fr.pool.ntp.org iburst
server 3.fr.pool.ntp.org iburst
```

```
systemctl enable ntpd
systemctl start ntpd
```

DNS

Reference: https://wiki.archlinux.org/index.php/Bind
Would have been nice with a chroot but thanks to systemd no scripts are available for arch yet (and no I won't even try to make some)

Bind

Installation is pretty straight forward:

```
pacman -S bind dnsutils
```

named.conf

```
vi /etc/named.conf
```

Add your local zone and the reverse

```
zone "home" IN {
        type master;
        file "home.zone";
};

zone "1.168.192.in-addr.arpa" IN {
        type master;
        file "home.rev";
};
```

Zones

Then configure files for the zones:

- /var/named/home.zone

```
$TTL    3h
```

```
@ IN SOA dns1.home. root.home. (
    2013083101 ; serial
    3h         ; refresh after 3 hours
    1h         ; retry after 1 hour
    1w         ; expire after 1 week
    1h )       ; negative caching TTL of 1 hour

        IN NS dns1.home.

;network devices and Services [1-30]
router  IN A 192.168.1.1
dns     IN A 192.168.1.2
wifi    IN A 192.168.1.3
[...]
```

- /var/named/home.rev

```
$TTL   3h
@ IN SOA dns1.home. root.home. (
    2013083101 ; serial
    3h         ; refresh after 3 hours
    1h         ; retry after 1 hour
    1w         ; expire after 1 week
    1h )       ; negative caching TTL of 1 hour

        IN NS dns1.home.

;network devices and Services [1-30]
1       IN PTR  router.home.
2       IN PTR  dns.home.
3       IN PTR  wifi.home.
[...]
```

Resolv.conf

Change it to simply:

```
search home
nameserver 127.0.0.1
```

Search <domain> will help using network tools without the domain after

 Important, since a lot of stupid services try to rewrite resolv.conf BLOCK it 

```
chattr +i /etc/resolv.conf
```

No comments on standards - I deeply regret not having openBSD here ...

Final step

Enable it

```
systemctl enable named
```

Start it

```
systemctl start named
```

Tests

Weel first try to dig/nslookup:

```
#nslookup 192.168.1.1
Server:         127.0.0.1
Address:        127.0.0.1#53

1.1.168.192.in-addr.arpa        name = router.home.

#nslookup router.home
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   router.home
Address: 192.168.1.1
```

See the performances:

```
for i in {1..30}; do echo google.com; done | xargs -I^ -P10 dig ^ | grep
time | awk /time/'{sum+=$4} END { print "Average query = ",sum/NR,"ms"}'

Average query =  33.6667 ms
```

Reference: http://www.heystephenwood.com/2013/06/use-your-raspberry-pi-as-dns-cache-to.html

Tweaks

You can check this wiki DNS
You need

```
pacman -S dos2unix
```

Add in /etc/named.conf

```
include "blockeddomains.zones";
```

In /var/named create a file called blockeddomain.hosts with:

```
$TTL   3h
@ IN SOA dns.home. root.home. (
    2013071502 ; serial
    3h         ; refresh after 3 hours
    1h         ; retry after 1 hour
    1w         ; expire after 1 week
    1h )       ; negative caching TTL of 1 hour


              IN      NS      dns.home.
              A       127.0.0.1
*             IN      A       127.0.0.1
*             IN      AAAA    ::1
```

then use this script to generate the blockeddomains.zones file:

```
#! /bin/sh
# Script that generate a blockeddomains.zones according to a nice hosts file
on the internet

# First rm previous version of downloaded and generated files
rm blockeddomains.zones hosts.txt
# Get the hosts file
wget http://winhelp2002.mvps.org/hosts.txt
# Convert to Unix
dos2unix hosts.txt
# For each line in the file that starts with 127, doesn't contain localhost
we take the 3rd field (domain)
for line in `cat hosts.txt | grep ^127 | grep -v localhost | cut -d " " -f
3`
do
     # Create an entry in  blockeddomain.zones with the correct syntax
pretending we own this domain :P
     printf "zone \"$line\"  {type master; file
\"blockeddomain.hosts\";};\n" >> blockeddomains.zones
done
```

Then copy it to /var/named and relaunch named

```
copy blockeddomains.zones /var/named/.
systemctl reload named
```

try a nslookup/dig on one of the domain and ... it will redirect on 127.0.0.1 :D