Networking year: 1.2

Semih Can Karakoc (695258)

**Our group exists of five people:*
Semih, Marvin, Ferhat, Thom and Vince

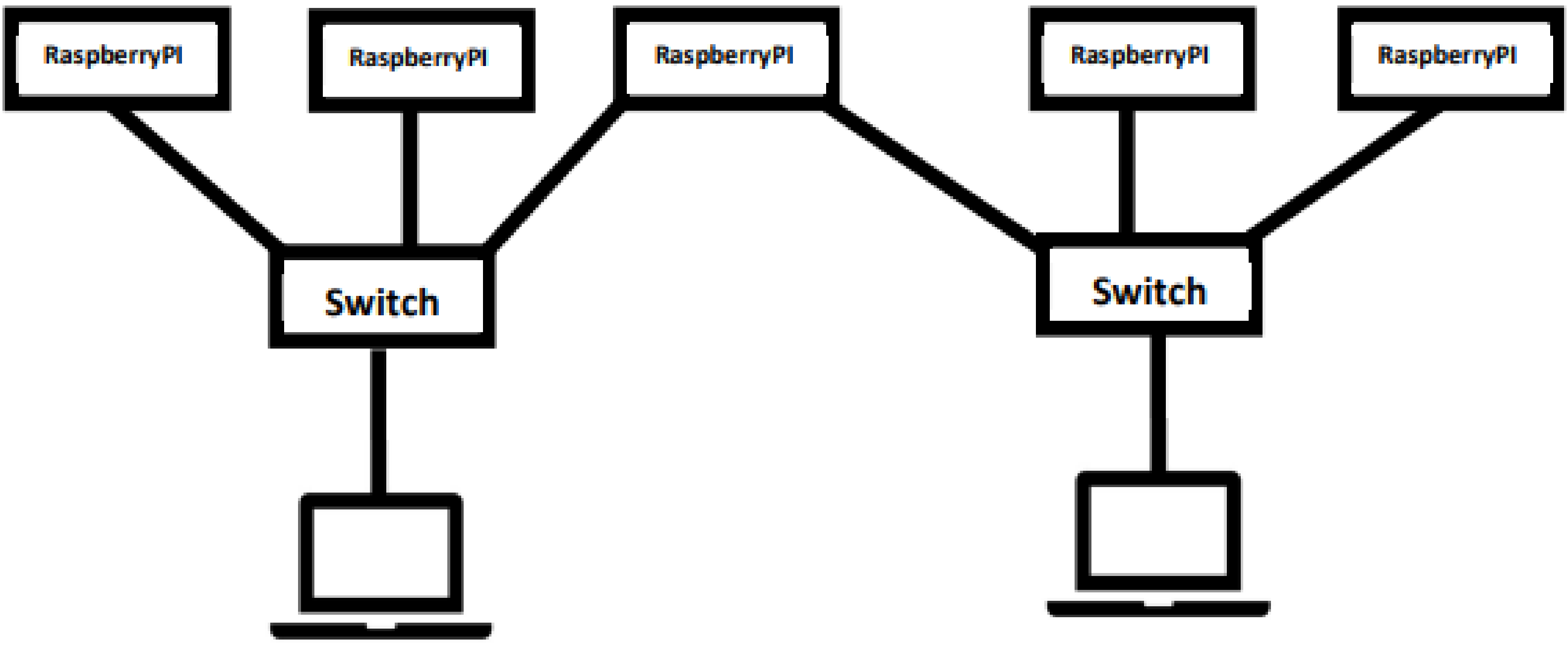
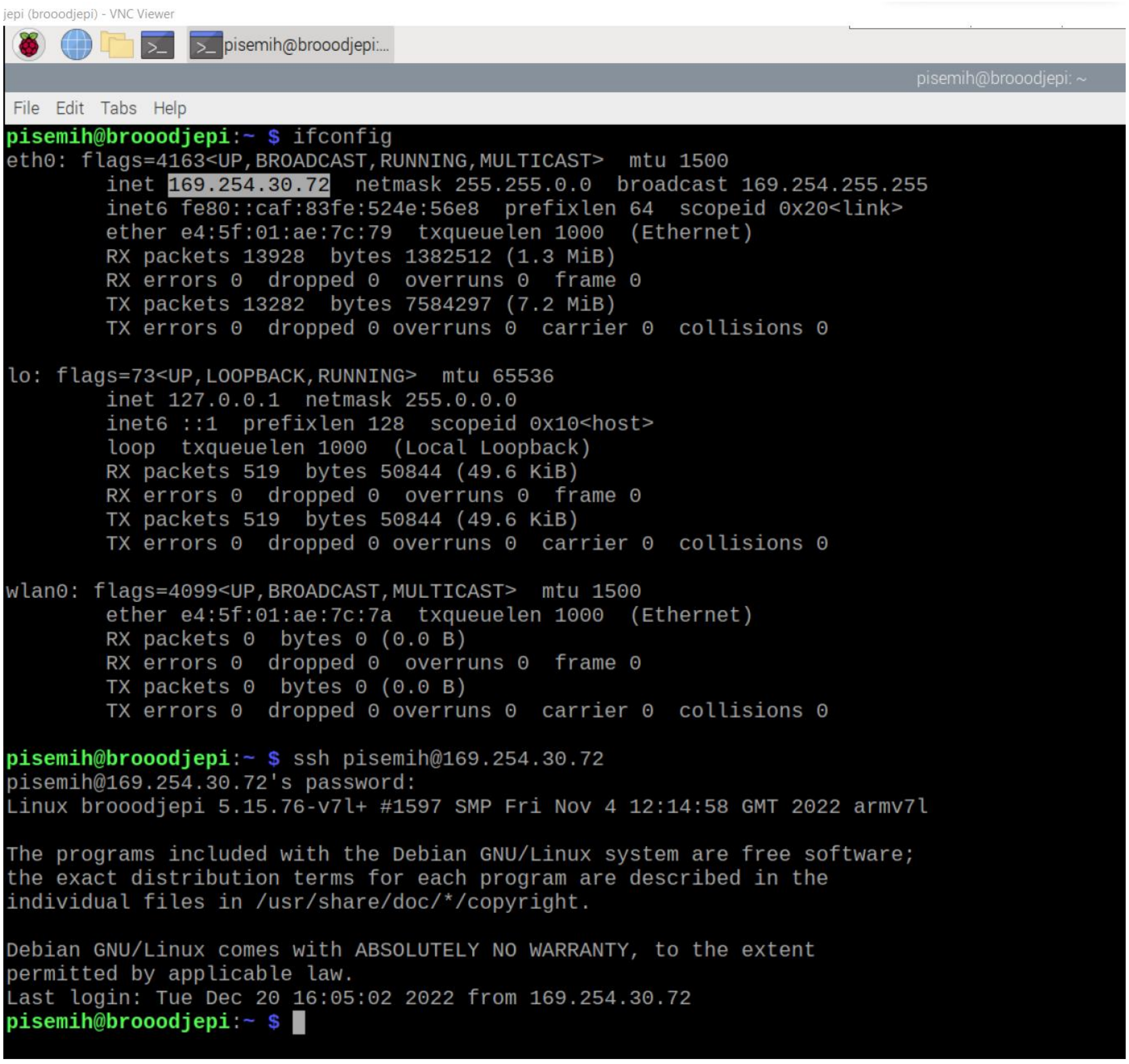


Table of Contents

Week 1 – Configuring Raspberry Pi

In week 1 we flashed/configured our new Raspberry Pi (with SSH enabled). I initially connected via PuTTY with SSH connection and enabled VNC in the Raspberry Pi settings (RasPi-config), then I installed VNC viewer on my laptop and connected via VNC to the Pi. VNC is like PuTTY, but with an user interface, which I found better to use. I actually own two Pi's, but the figure below represents the Pi we got from school (proof of SSH and Raspberry Pi working in VNC):



```
pisemih@brooodjepi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 169.254.30.72  netmask 255.255.0.0  broadcast 169.254.255.255
        inet6 fe80::caf:83fe:524e:56e8  prefixlen 64  scopeid 0x20<link>
        ether e4:5f:01:ae:7c:79  txqueuelen 1000  (Ethernet)
        RX packets 13928  bytes 1382512 (1.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13282  bytes 7584297 (7.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 519  bytes 50844 (49.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 519  bytes 50844 (49.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e4:5f:01:ae:7c:7a  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pisemih@brooodjepi:~ $ ssh pisemih@169.254.30.72
pisemih@169.254.30.72's password:
Linux brooodjepi 5.15.76-v7l+ #1597 SMP Fri Nov 4 12:14:58 GMT 2022 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Dec 20 16:05:02 2022 from 169.254.30.72
pisemih@brooodjepi:~ $ 
```

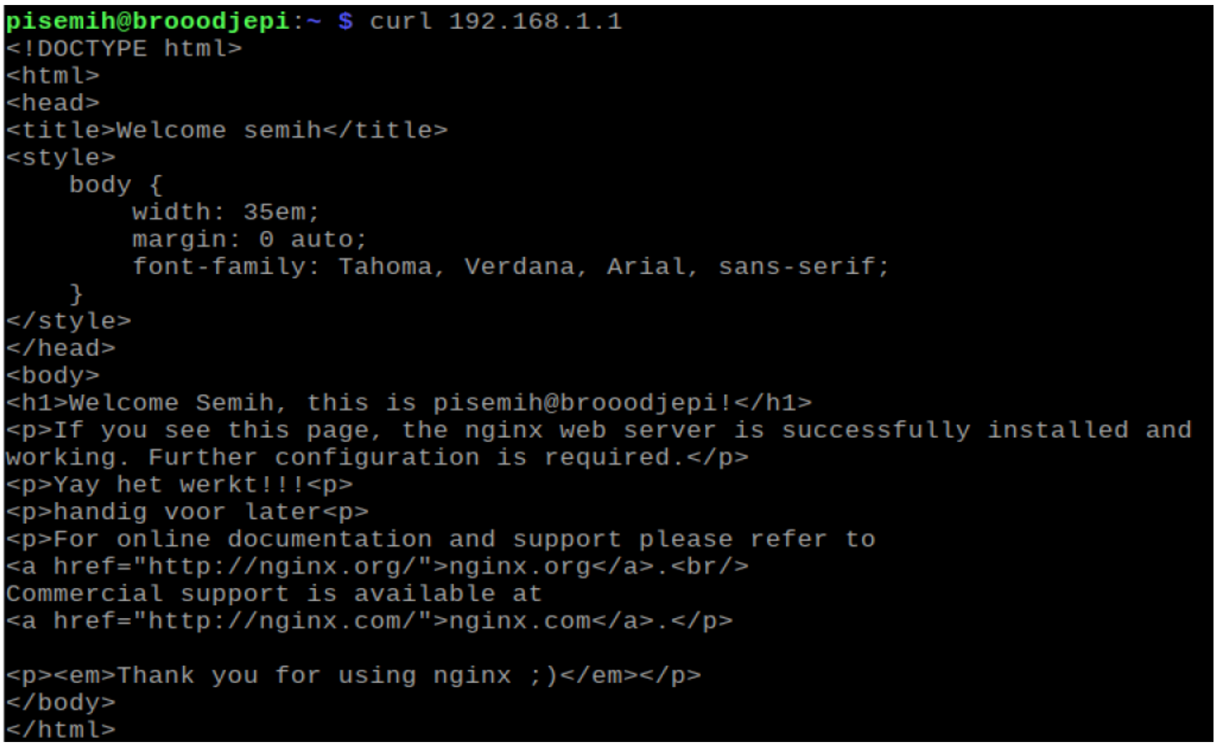
Week 2 - Webserver installation and configuration

In week 2 we tried to setup Nginx on our Raspberry Pi's. After we were done installing Nginx by using the command "`sudo apt install nginx`", it successfully worked! We also changed the HTML source code by accessing "`/var/www/html/index.nginx-debian.html`" directory.

The Nginx site is accessible by typing in your (static) IP-address (192.168.1.1) into the search bar of Chromium (or any other internet browser). If everything is setup correctly you will see "welcome to nginx!", or something else, whether the HTML has been changed or not.

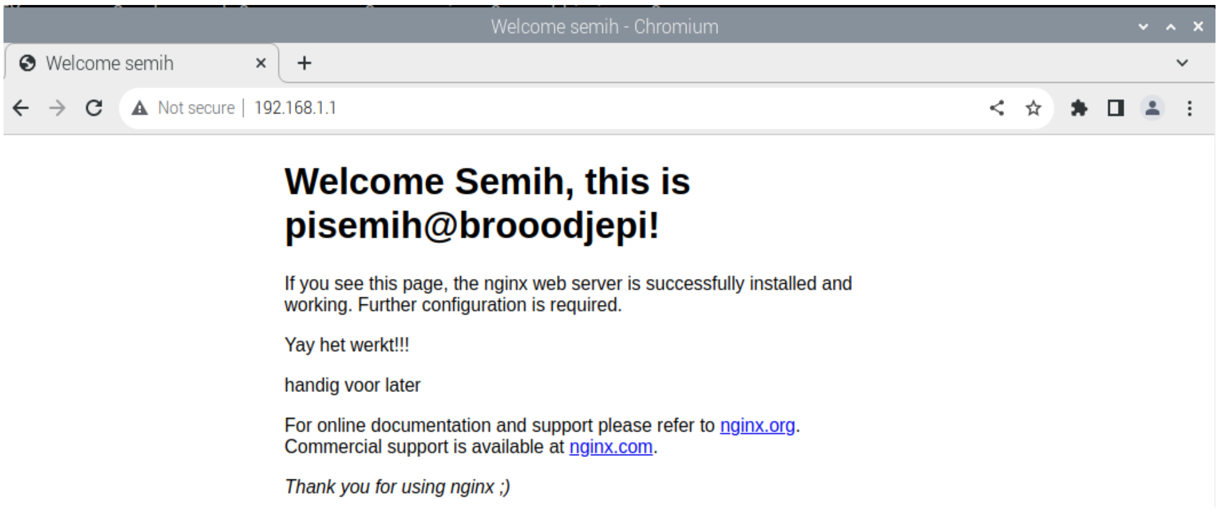
Another option is to use the command "curl", which shows the source code of the URL you've typed in. It can be used like this: "`curl 169.254.30.72`", or for in the screenshot below that I took it looks like: "`curl 192.168.1.1`".

See photo's below for proof that it worked (a different IP-address than week 1 though):



```
pisemih@brooodjepi:~ $ curl 192.168.1.1
<!DOCTYPE html>
<html>
<head>
<title>Welcome semih</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome Semih, this is pisemih@brooodjepi!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>
<p>Yay het werkt!!!<p>
<p>handig voor later<p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx ;)</em></p>
</body>
</html>
```

In the screenshot below you can see that it also works by typing in the IP-address into the search bar:



Welcome Semih, this is pisemih@brooodjepi!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

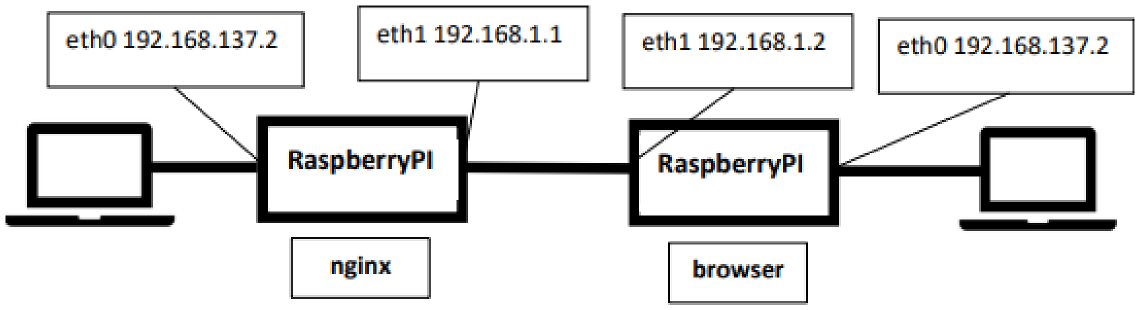
Yay het werkt!!!

handig voor later

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx ;)

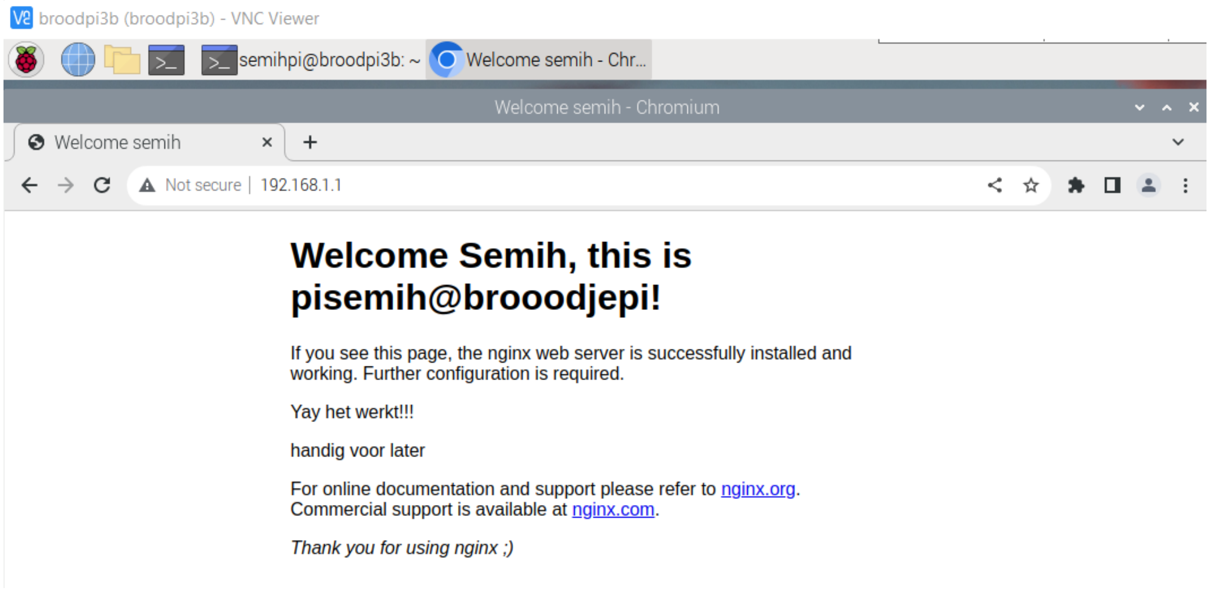
In week 2 we also had to setup “laptop1-RaspPi1-RaspPi2-laptop2” network. See figure below for a better imagination:



We managed to do this in week 2 in the lesson, but I didn’t take screenshots, so I didn’t have proof.

Anyway, I am lucky enough that I have two Pi’s, so I remade this network at home.

In this case the nginx server had a static IP-address of 192.168.1.1 (pisemih@brooodjepi) and the browser was 192.168.1.2 (semihpi@broodpi3b). See screenshot below for proof that the browser could see the nginx website and that it actually worked:



Welcome Semih, this is pisemih@brooodjepi!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

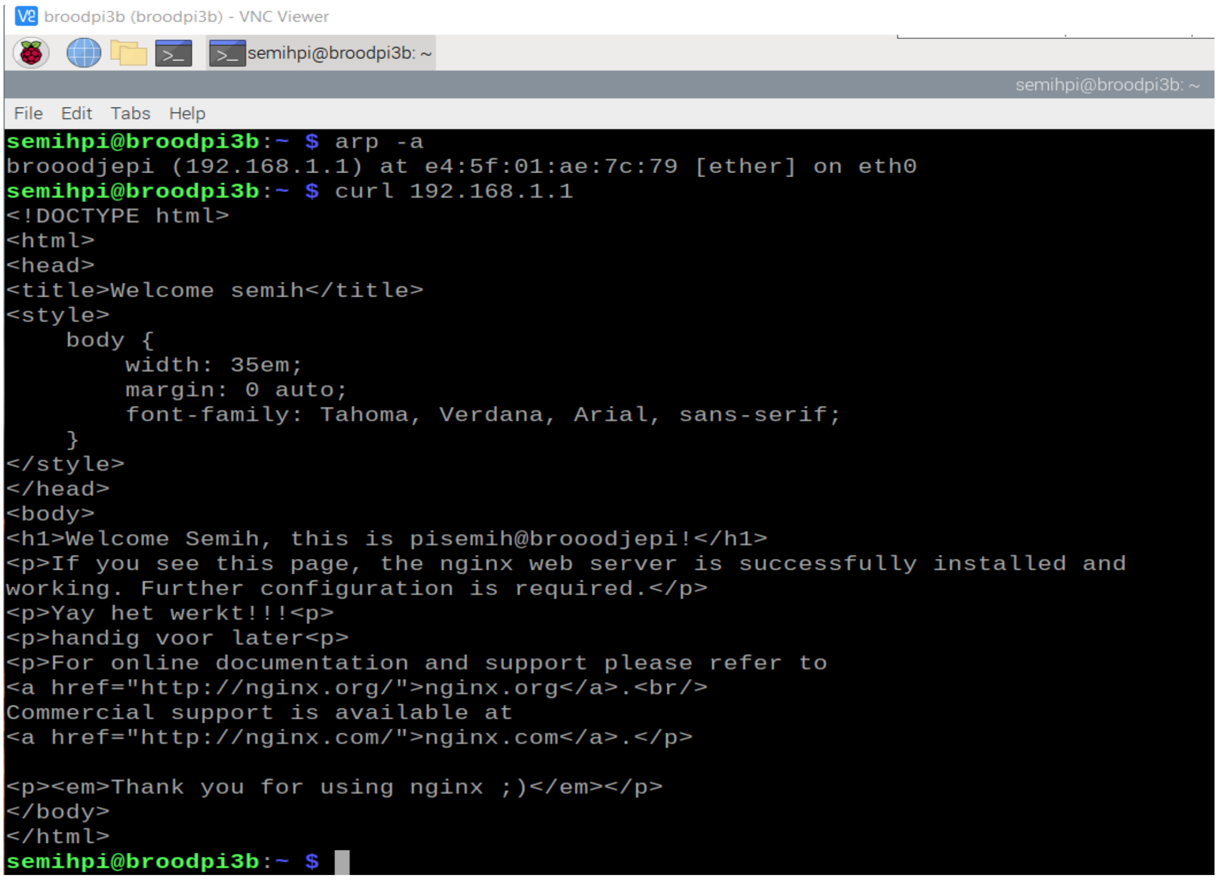
Yay het werkt!!!

handig voor later

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

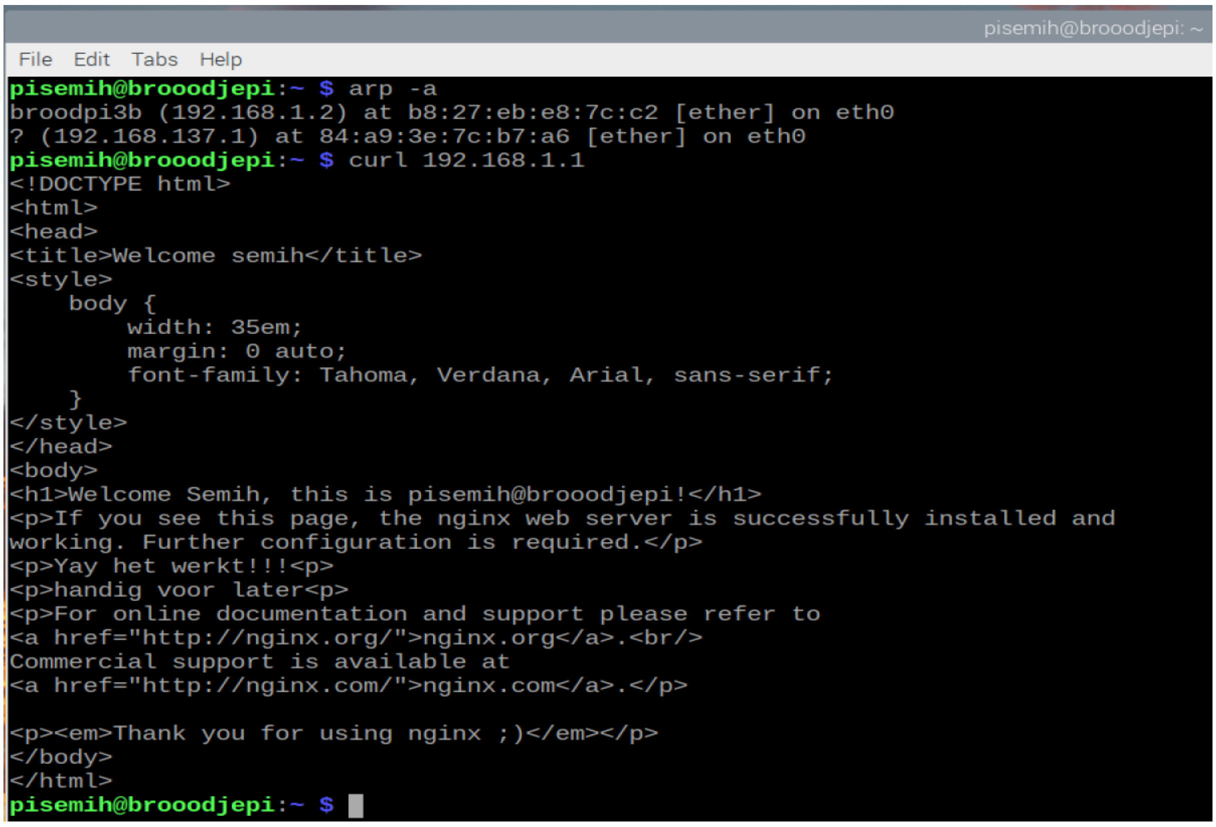
Thank you for using nginx ;)

See screenshot below for proof that 192.168.1.2 (browser) could see the nginx HTML code via curl. The screenshot also has proof that brooodjepi has an IP-address of 192.168.1.1:



```
semihpi@broodpi3b:~ $ arp -a
brooodjepi (192.168.1.1) at e4:5f:01:ae:7c:79 [ether] on eth0
semihpi@broodpi3b:~ $ curl 192.168.1.1
<!DOCTYPE html>
<html>
<head>
<title>Welcome semih</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome Semih, this is pisemih@brooodjepi!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>
<p>Yay het werkt!!!<p>
<p>handig voor later<p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx ;)</em></p>
</body>
</html>
semihpi@broodpi3b:~ $
```

See screenshot below for proof that the nginx server (192.168.1.1) was able to see the browser (192.168.1.2) and proof that the nginx server was able to see its own HTML source code via curl:

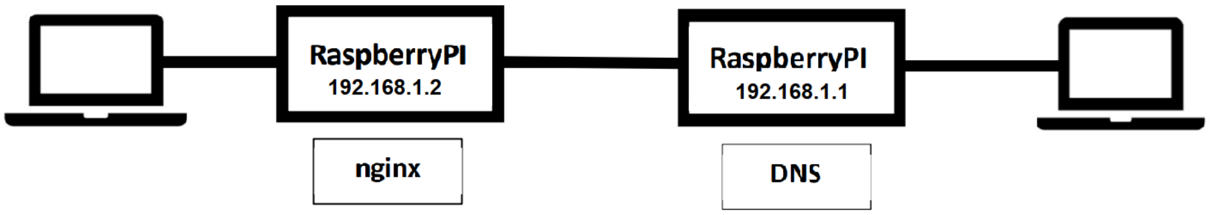


```
pisemih@brooodjepi:~ $ arp -a
broodpi3b (192.168.1.2) at b8:27:eb:e8:7c:c2 [ether] on eth0
? (192.168.137.1) at 84:a9:3e:7c:b7:a6 [ether] on eth0
pisemih@brooodjepi:~ $ curl 192.168.1.1
<!DOCTYPE html>
<html>
<head>
<title>Welcome semih</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome Semih, this is pisemih@brooodjepi!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>
<p>Yay het werkt!!!<p>
<p>handig voor later<p>
<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx ;)</em></p>
</body>
</html>
pisemih@brooodjepi:~ $
```

Week 3 - DNS configuration

In week 3 we had to co-operate in groups of two. One Pi had to be the DNS server and the other Pi had to be the nginx. The DNS server should give a domain to the nginx IP-address. In week 3 in the lesson we used “DNSmasq” to setup a DNS server, but in week 5/6 we found out that we couldn’t use DNSmasq for this project. Instead, we had to use BIND9.

Again, I was lucky that I had two Pi’s, so I remade this network at home:



I used my new Pi4 as the DNS server. I installed BIND9 with the following command: “`sudo apt-get install bind9 bind9utils dnsutils`”. Now that I have BIND9 installed, I tried to configure the settings and zones for my network with help from the syllabus. Unfortunately the code from the syllabus didn’t really work, so I researched like two full days how to configure BIND9. I followed a YouTube video on internet about how to setup BIND9 and it worked eventually.

Anyway, I had to open the configuration file: “`sudo nano /etc/bind/named.local.conf`”. In this file you can add/delete zones. There are two zones: forward lookup and reverse lookup. The forward lookup is mostly important, because the server looks for the domain name and in the zone file it tells the server which IP-address the domain name has. The inverse zone is needed for inverse queries. I used 192.168.1.1 as static IP for the DNS server and nginx server had 192.168.1.2 as IP-address.

My “named.conf.local” file looks like this:

```
zone “semih.nl” IN {
      type master;
      file “/etc/bind/db.semih.nl”;
};

zone “1.168.192.in-addr.arpa” {
      type master;
      file “/etc/bind/db.rev.1.168.192.in-addr.arpa”;
};
```

My “db.semih.nl” file looks like this:

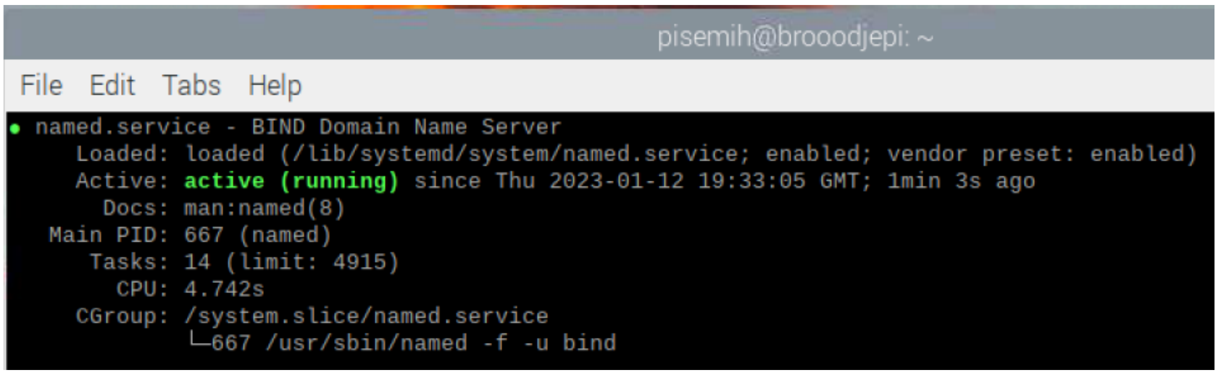
```
$TTL  1H
semih.nl. IN SOA ns1.semih.nl. admin.semih.nl. (
      2017081401 ; Serial
      2H         ; Refresh
      1H         ; Retry
      1W         ; Expire
      1D         ; Negative Cache TTL
)

      IN    NS    ns1.semih.nl.
ns1   IN    A     192.168.1.2
www   IN    A     192.168.1.2
admin IN    A     192.168.1.2
```

My “db.rev.1.168.192.in-addr.arpa” file looks like this:

```
$TTL  1H
@     IN    SOA   ns1.semih.nl. admin.semih.nl. (
      2017081401 ; serial
      2H         ; refresh
      1H         ; retry
      1W         ; expire
      1D         ; minimum
)
      IN    NS    ns1.semih.nl.
2017081401 IN    PTR   www.semih.nl.
```

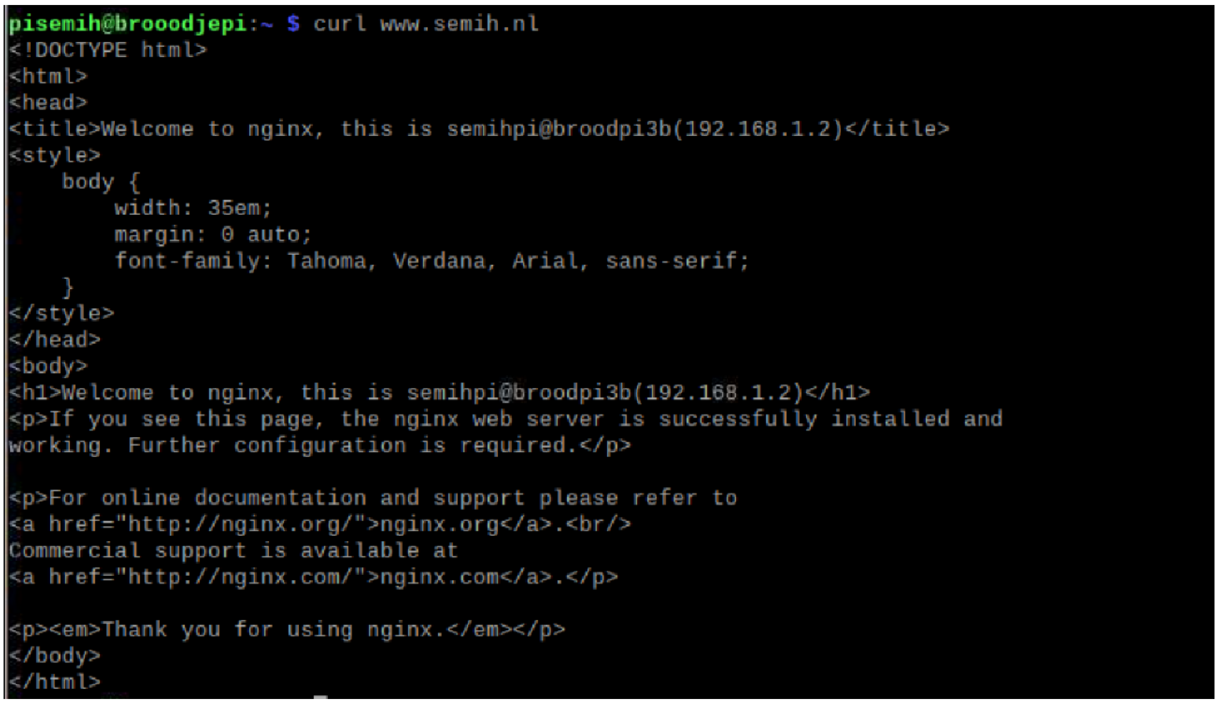
You can restart BIND9 service and you can also see the status of BIND9 service by using this command: “`sudo service bind9 restart`” or “`sudo service bind9 status`“. I also used the package “DiG” to be sure that everything is working correctly.

See screenshot below for proof that the BIND9 service was working correctly:



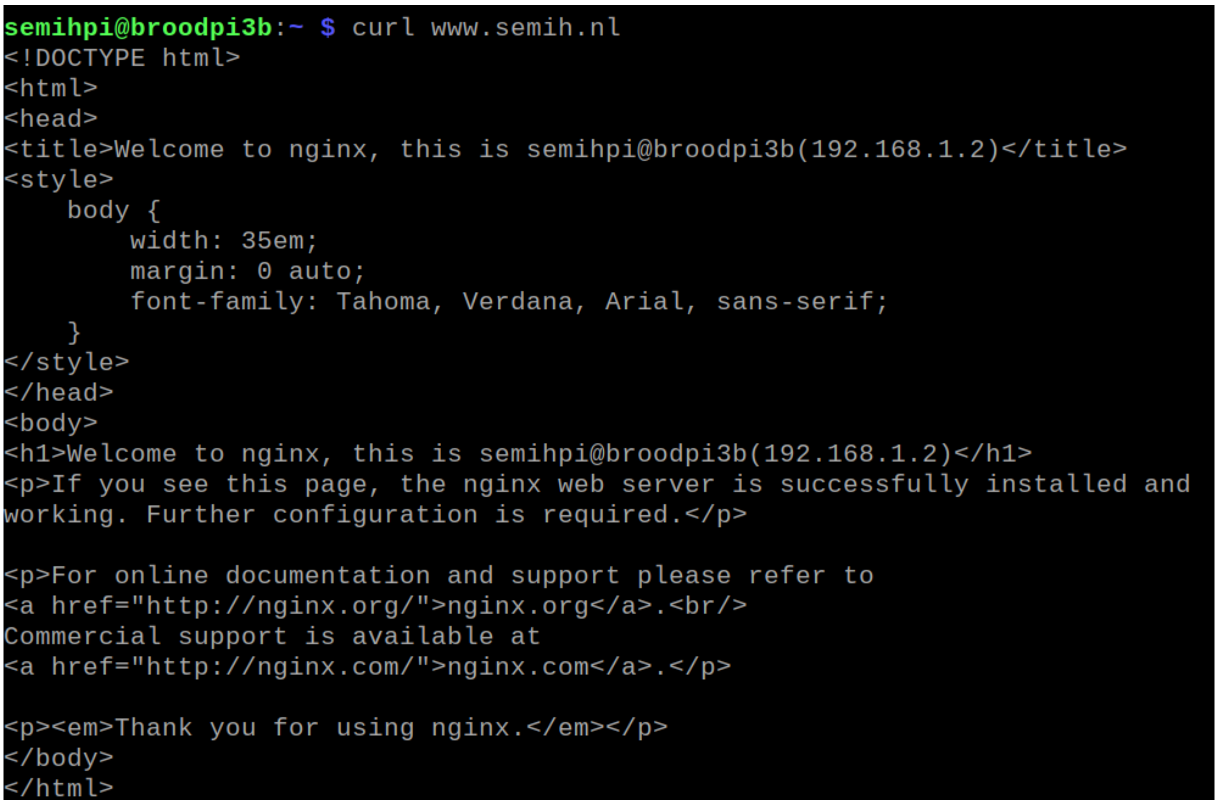
```
● named.service - BIND Domain Name Server
     Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-01-12 19:33:05 GMT; 1min 3s ago
       Docs: man:named(8)
   Main PID: 667 (named)
      Tasks: 14 (limit: 4915)
        CPU: 4.742s
     CGroup: /system.slice/named.service
             └─667 /usr/sbin/named -f -u bind
```

In the screenshot below you can see that I used curl on my domain name to see if it works, and it does!



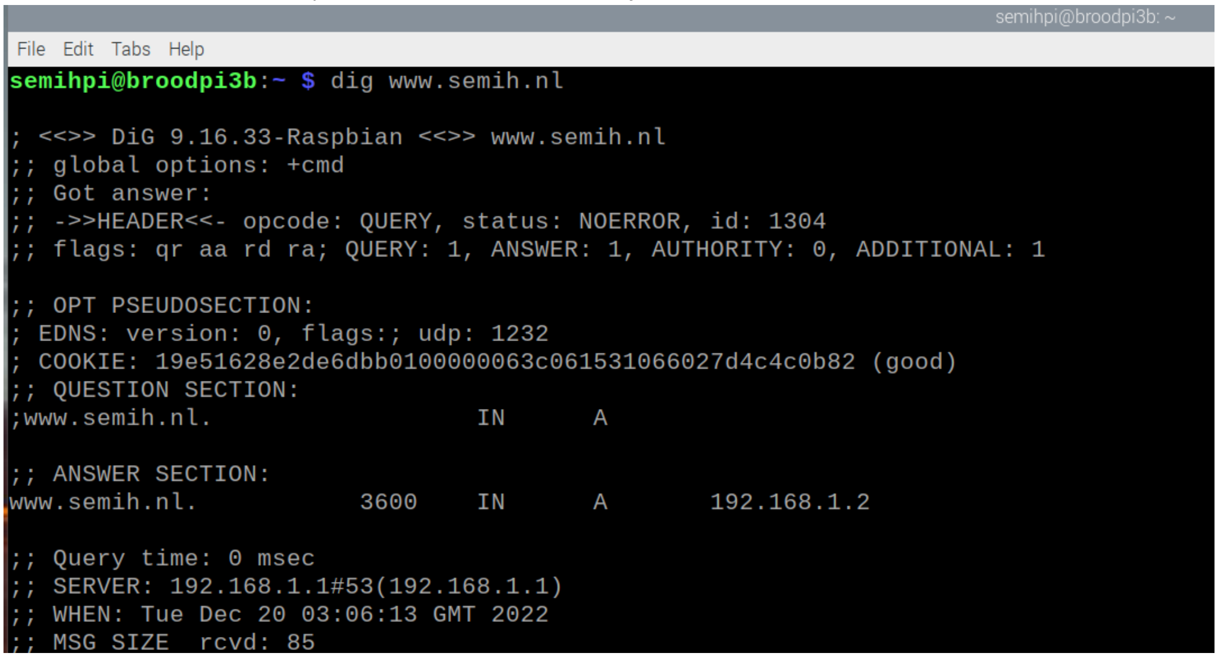
```
pisemih@brooodjepi:~ $ curl www.semih.nl
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2)</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2)</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Here the same screenshot, but then from the semihpi Raspberry Pi:



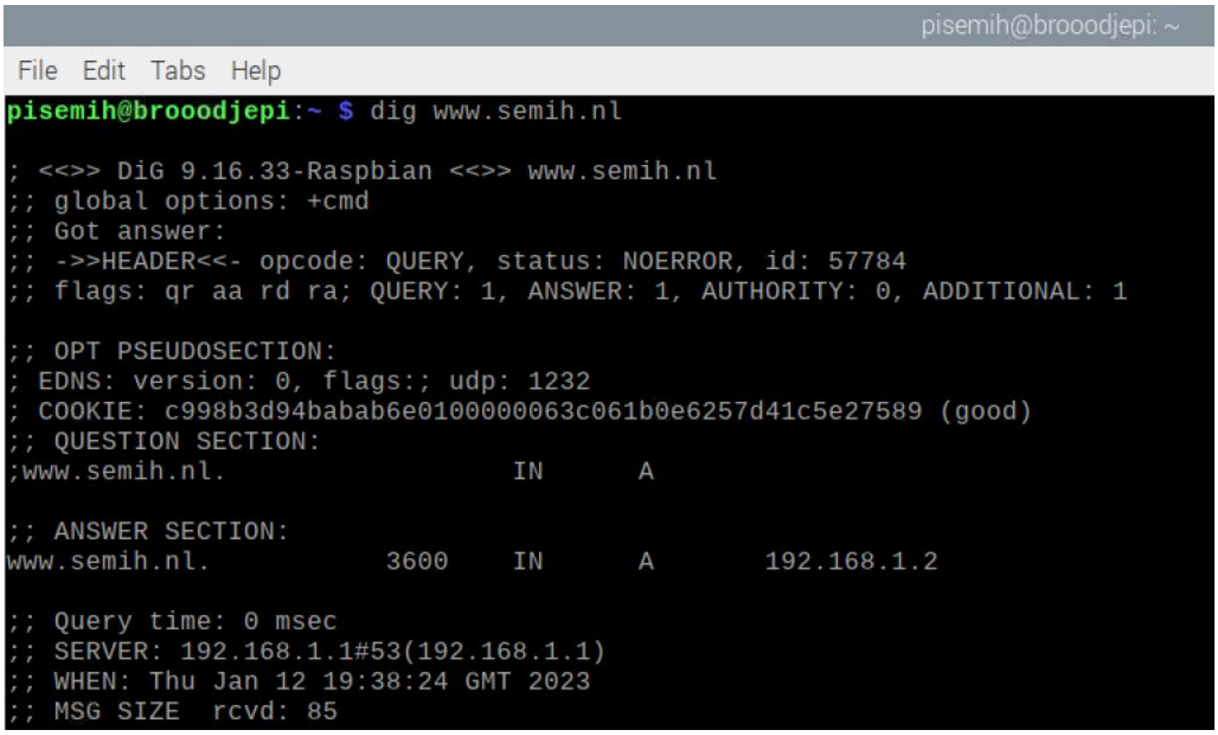
```
semihpi@broodpi3b:~ $ curl www.semih.nl
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2)</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2)</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

See screenshot below for proof that DiG also works, just like curl, but with more detail:



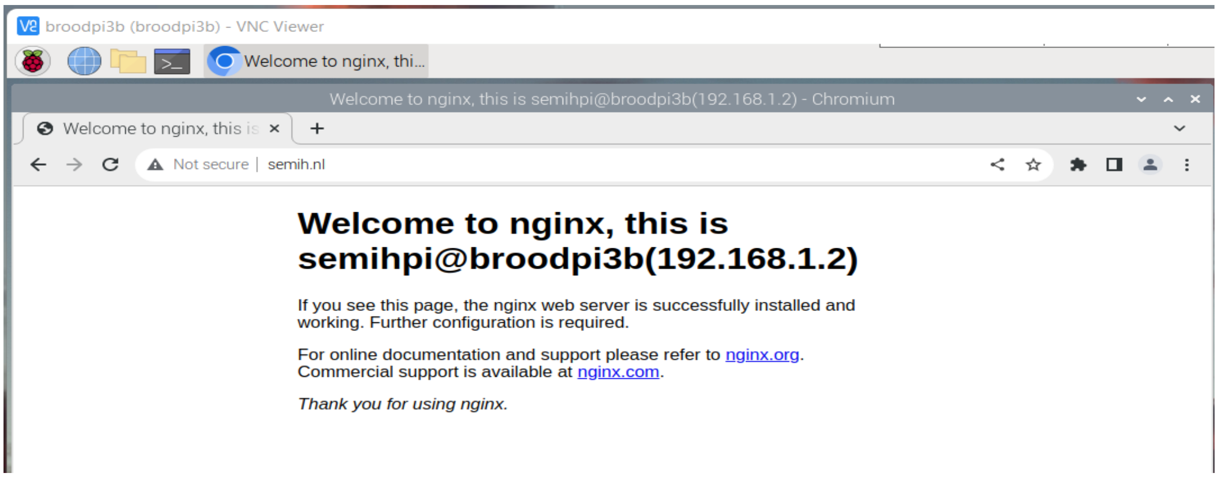
```
semihpi@broodpi3b:~ $ dig www.semih.nl

; <<>> DiG 9.16.33-Raspbian <<>> www.semih.nl
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1304
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 19e51628e2de6dbb0100000063c061531066027d4c4c0b82 (good)
;; QUESTION SECTION:
;www.semih.nl.                  IN      A

;; ANSWER SECTION:
www.semih.nl.           3600    IN      A       192.168.1.2

;; Query time: 0 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Tue Dec 20 03:06:13 GMT 2022
;; MSG SIZE  rcvd: 85
```

See screenshot below for proof that DiG worked, but then from pisemih (DNS server) perspective:



```
pisemih@brooodjepi:~ $ dig www.semih.nl

; <<>> DiG 9.16.33-Raspbian <<>> www.semih.nl
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57784
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: c998b3d94babab6e0100000063c061b0e6257d41c5e27589 (good)
;; QUESTION SECTION:
;www.semih.nl.                  IN      A

;; ANSWER SECTION:
www.semih.nl.           3600    IN      A       192.168.1.2

;; Query time: 0 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Thu Jan 12 19:38:24 GMT 2023
;; MSG SIZE  rcvd: 85
```

See the screenshots below for proof that both Pi's were able to access www.semih.nl via browser:



Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2) - Chromium

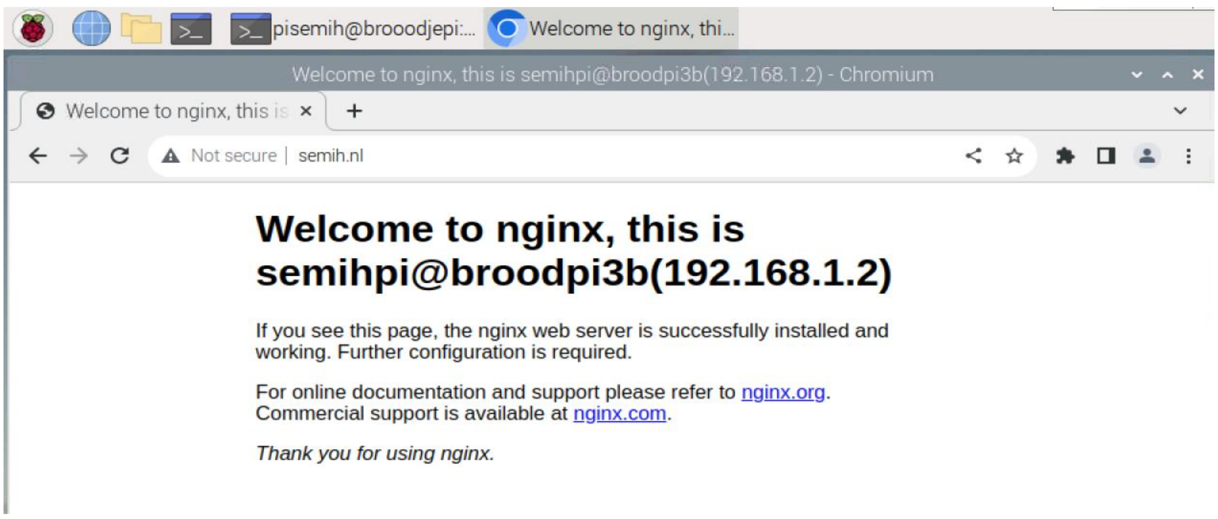
Not secure | semih.nl

Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2)

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.



Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2) - Chromium

Not secure | semih.nl

Welcome to nginx, this is semihpi@broodpi3b(192.168.1.2)

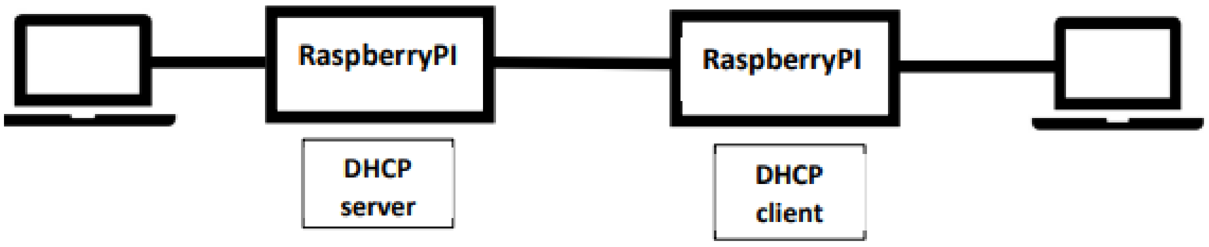
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Week 4 - DHCP configuration

In week 4 we made a DHCP server with a DHCP client network, see figure below:



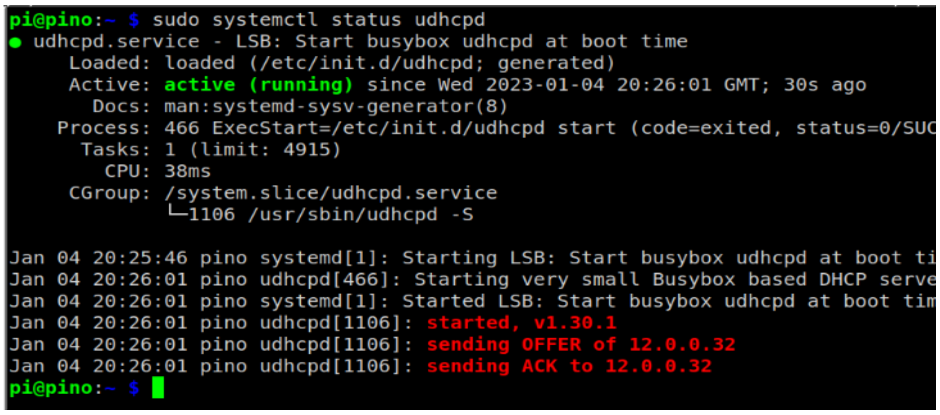
My Raspberry Pi (brooodjepi) was in this case the DHCP client, and Ferhat his Pi (called pino) was the DHCP server. He used “`sudo apt-get install udhcpd`” to install the DHCP server. Ferhat configured his server by doing “`sudo nano /etc/udhcpd.conf`”.

Here he put the following code:

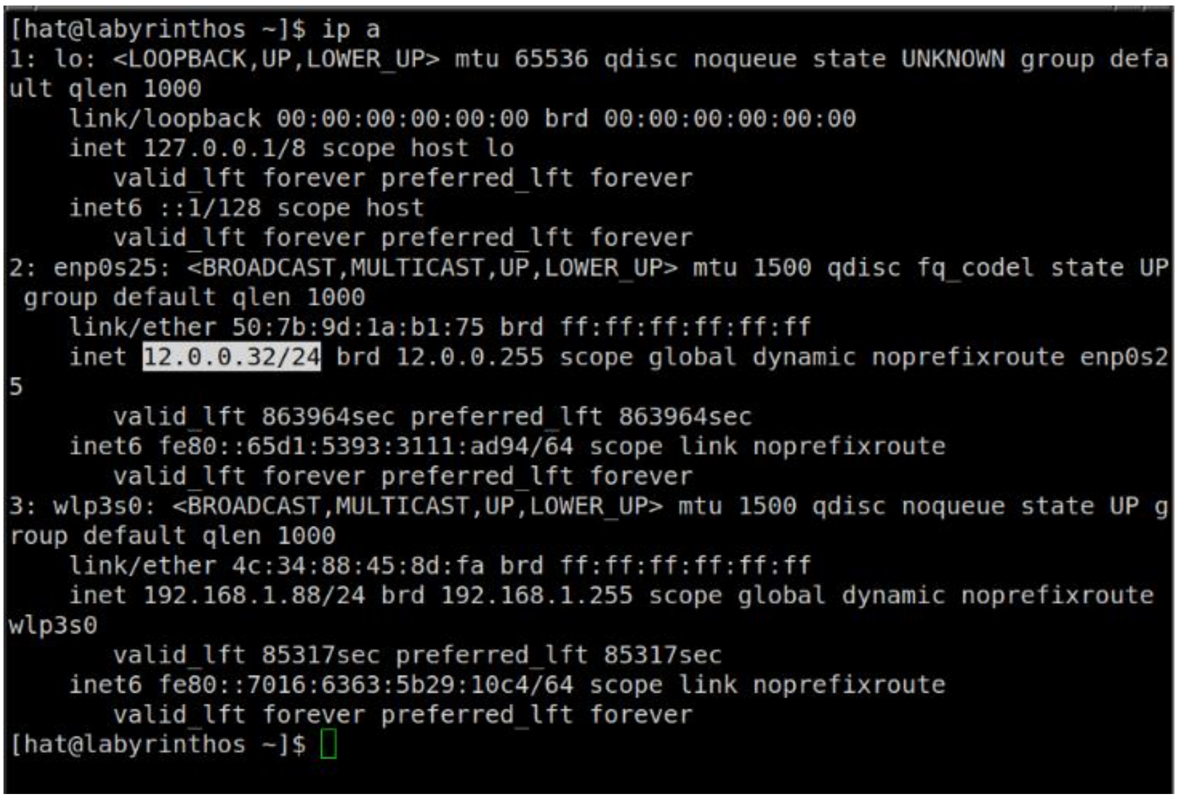
```
start           12.0.0.3
end             12.0.0.254

interface       eth0

option  dns     2.0.0.10
option  subnet  255.255.255.0
#option router  12.0.0.2
option  domain  local
option  lease   864000  # 10 days in seconds

static_lease 00:e0:4c:68:68:ba 12.0.0.10
```

Start means the starting IP lease range, and end is the end of the IP lease range. In this case the range is from 12.0.0.3 to 12.0.0.254 (251 options). In our network the router had a static-IP, which was 12.0.0.2. My Pi (DNS server) had to lease an IP from the DHCP server and it had to be a lease to work. So we added “`static_lease 00:e0:4c:68:68:ba 12.0.0.12`” to the file, as you can see. This means when there is a device (in this case my Pi) with that MAC address, it should lease 12.0.0.10 to that device.

With “`sudo nano /etc/default/udchpd`” you need to enable the DHCP sever, search in that file for `DHCPD_ENABLED="no"` and set “no” to “yes”. We also had to be sure that the laptops their internet sharing was turned off. Otherwise the laptop will give an IP-address to the Raspberry Pi! See screenshots below: proof that his DHCP server worked (he did this at home with other IP’s):

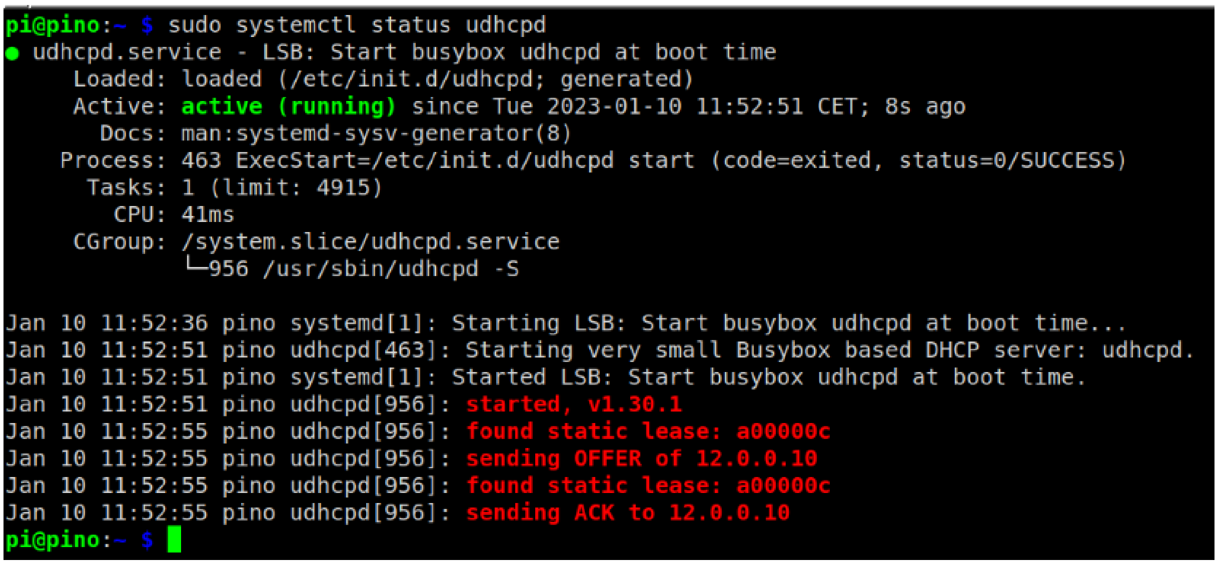


```
pi@pino:~ $ sudo systemctl status udhcpd
● udhcpd.service - LSB: Start busybox udhcpd at boot time
     Loaded: loaded (/etc/init.d/udhcpd; generated)
     Active: active (running) since Wed 2023-01-04 20:26:01 GMT; 30s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 466 ExecStart=/etc/init.d/udhcpd start (code=exited, status=0/SUC
      Tasks: 1 (limit: 4915)
        CPU: 38ms
     CGroup: /system.slice/udhcpd.service
             └─1106 /usr/sbin/udhcpd -S

Jan 04 20:25:46 pino systemd[1]: Starting LSB: Start busybox udhcpd at boot ti
Jan 04 20:26:01 pino udhcpd[466]: Starting very small Busybox based DHCP serve
Jan 04 20:26:01 pino systemd[1]: Started LSB: Start busybox udhcpd at boot tim
Jan 04 20:26:01 pino udhcpd[1106]: started, v1.30.1
Jan 04 20:26:01 pino udhcpd[1106]: sending OFFER of 12.0.0.32
Jan 04 20:26:01 pino udhcpd[1106]: sending ACK to 12.0.0.32
pi@pino:~ $
```



```
[hat@labyrinthos ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defa
ult qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s25: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
 group default qlen 1000
    link/ether 50:7b:9d:1a:b1:75 brd ff:ff:ff:ff:ff:ff
    inet 12.0.0.32/24 brd 12.0.0.255 scope global dynamic noprefixroute enp0s2
5
       valid_lft 863964sec preferred_lft 863964sec
    inet6 fe80::65d1:5393:3111:ad94/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: wlp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP g
roup default qlen 1000
    link/ether 4c:34:88:45:8d:fa brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.88/24 brd 192.168.1.255 scope global dynamic noprefixroute
wlp3s0
       valid_lft 85317sec preferred_lft 85317sec
    inet6 fe80::7016:6363:5b29:10c4/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[hat@labyrinthos ~]$
```

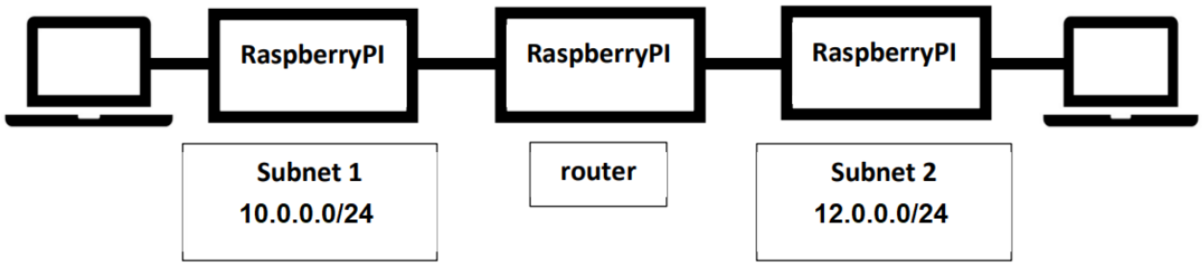
See the screenshot below for proof that the DNS server actually got the leased IP-address from the DHCP server:



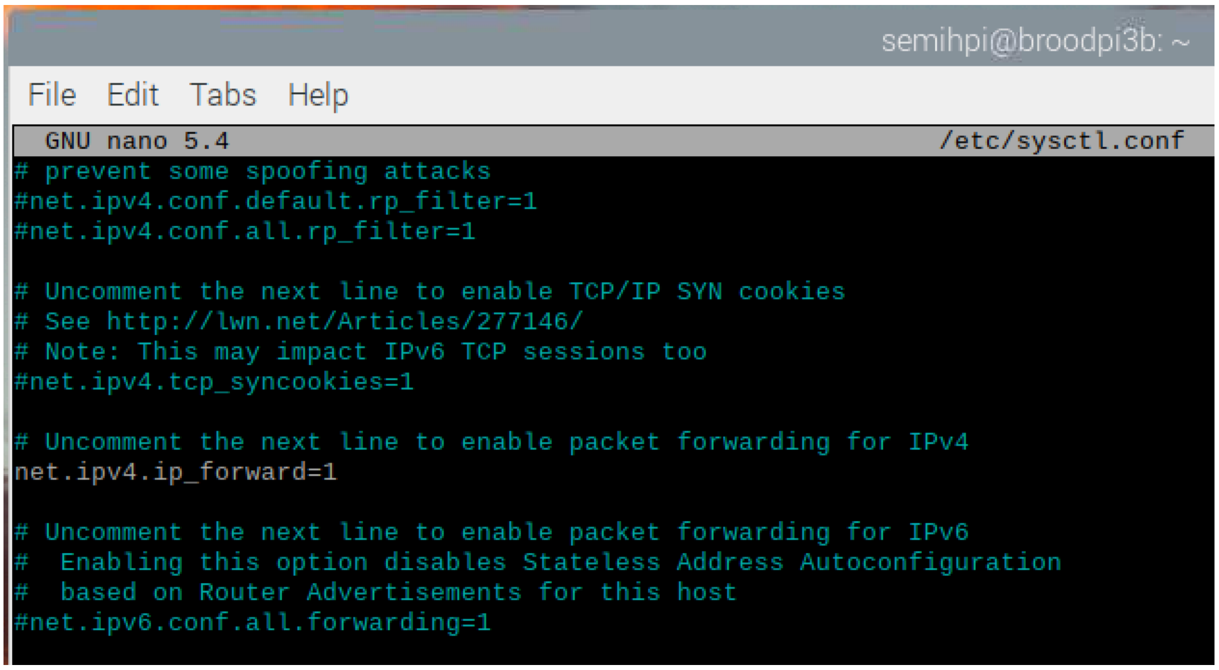
```
pi@pino:~ $ sudo systemctl status udhcpd
● udhcpd.service - LSB: Start busybox udhcpd at boot time
     Loaded: loaded (/etc/init.d/udhcpd; generated)
     Active: active (running) since Tue 2023-01-10 11:52:51 CET; 8s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 463 ExecStart=/etc/init.d/udhcpd start (code=exited, status=0/SUCCESS)
      Tasks: 1 (limit: 4915)
        CPU: 41ms
     CGroup: /system.slice/udhcpd.service
             └─956 /usr/sbin/udhcpd -S

Jan 10 11:52:36 pino systemd[1]: Starting LSB: Start busybox udhcpd at boot time...
Jan 10 11:52:51 pino udhcpd[463]: Starting very small Busybox based DHCP server: udhcpd.
Jan 10 11:52:51 pino systemd[1]: Started LSB: Start busybox udhcpd at boot time.
Jan 10 11:52:51 pino udhcpd[956]: started, v1.30.1
Jan 10 11:52:55 pino udhcpd[956]: found static lease: a00000c
Jan 10 11:52:55 pino udhcpd[956]: sending OFFER of 12.0.0.10
Jan 10 11:52:55 pino udhcpd[956]: found static lease: a00000c
Jan 10 11:52:55 pino udhcpd[956]: sending ACK to 12.0.0.10
pi@pino:~ $
```

Week 5/6 – Routing

In this week we had to setup two subnetworks with one Raspberry Pi each and one router between those subnets, see figure below:



Setting up the router wasn't that difficult to do, in my opinion. We had to turn on IP-forwarding for the router Pi. We did this by going to the sysctl.conf file: "`sudo nano /etc/sysctl.conf`". Scroll down until you find "`#net.IPv4.IP_forward=1`", delete the #, so it becomes an uncommented line. Now we have turned on IP forwarding. Run "sysctl -p" to affect changes direct.



```
 GNU nano 5.4                                                /etc/sysctl.conf
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1
```

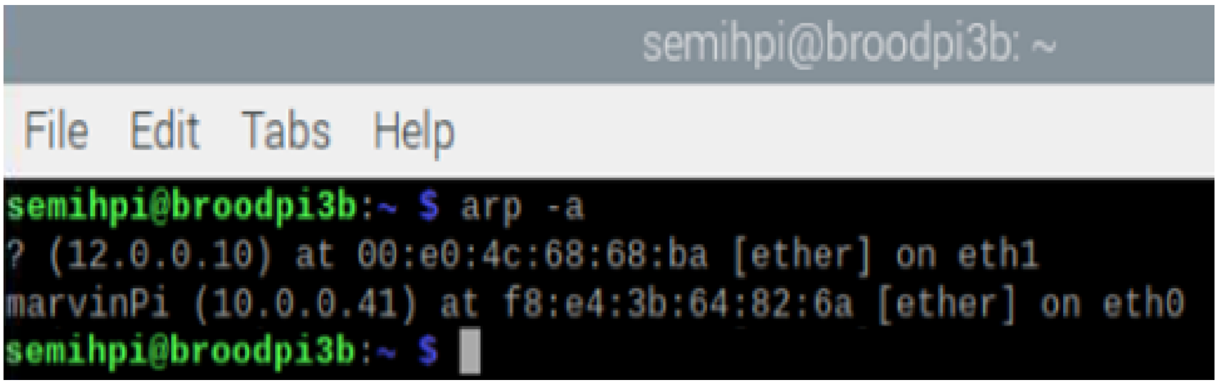
The only thing that we need to do is to configure a gateway for the router and client(s). Go to the dhcpcd.conf file and scroll to the bottom. Add the following lines (eth0=subnet1 and eth1=subnet2):

```
interface eth0
static IP_address=10.0.0.2

interface eth1
static IP_address=12.0.0.2
```

All individual clients on each subnet needs to configure their default gateway to communicate with the router, here is an example for a client from subnet 2 (in their dhcpcd.conf):

```
interface eth1
static routers=12.0.0.2
```

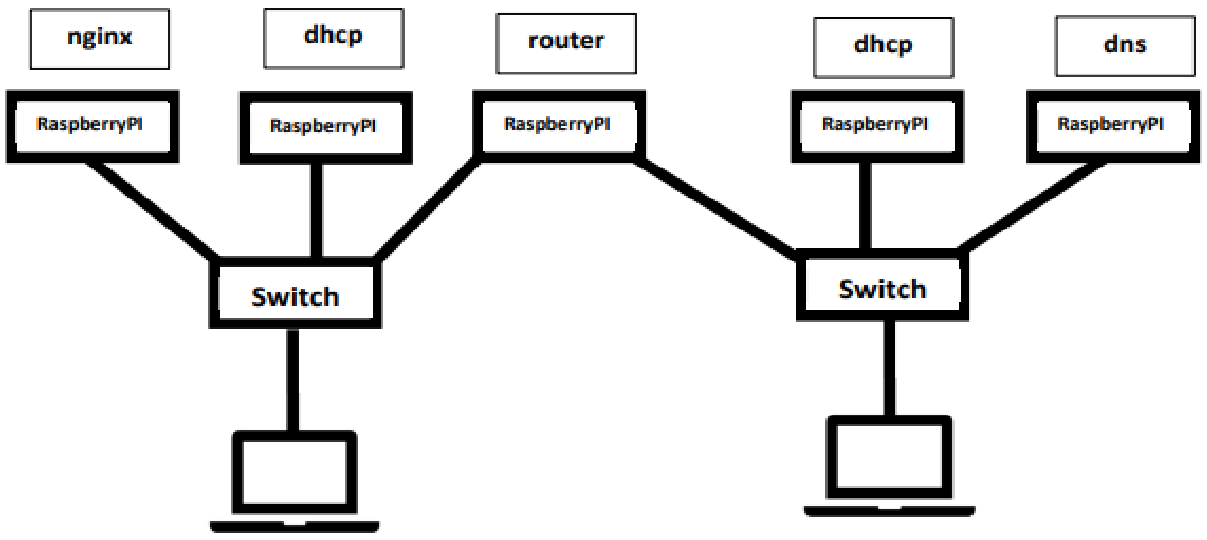
The router and clients are now setup correctly and should work. See screenshot below for proof that the router works (12.0.0.10 = pisemih@brooodjepi and 10.0.0.41 = pi@marvinPi):



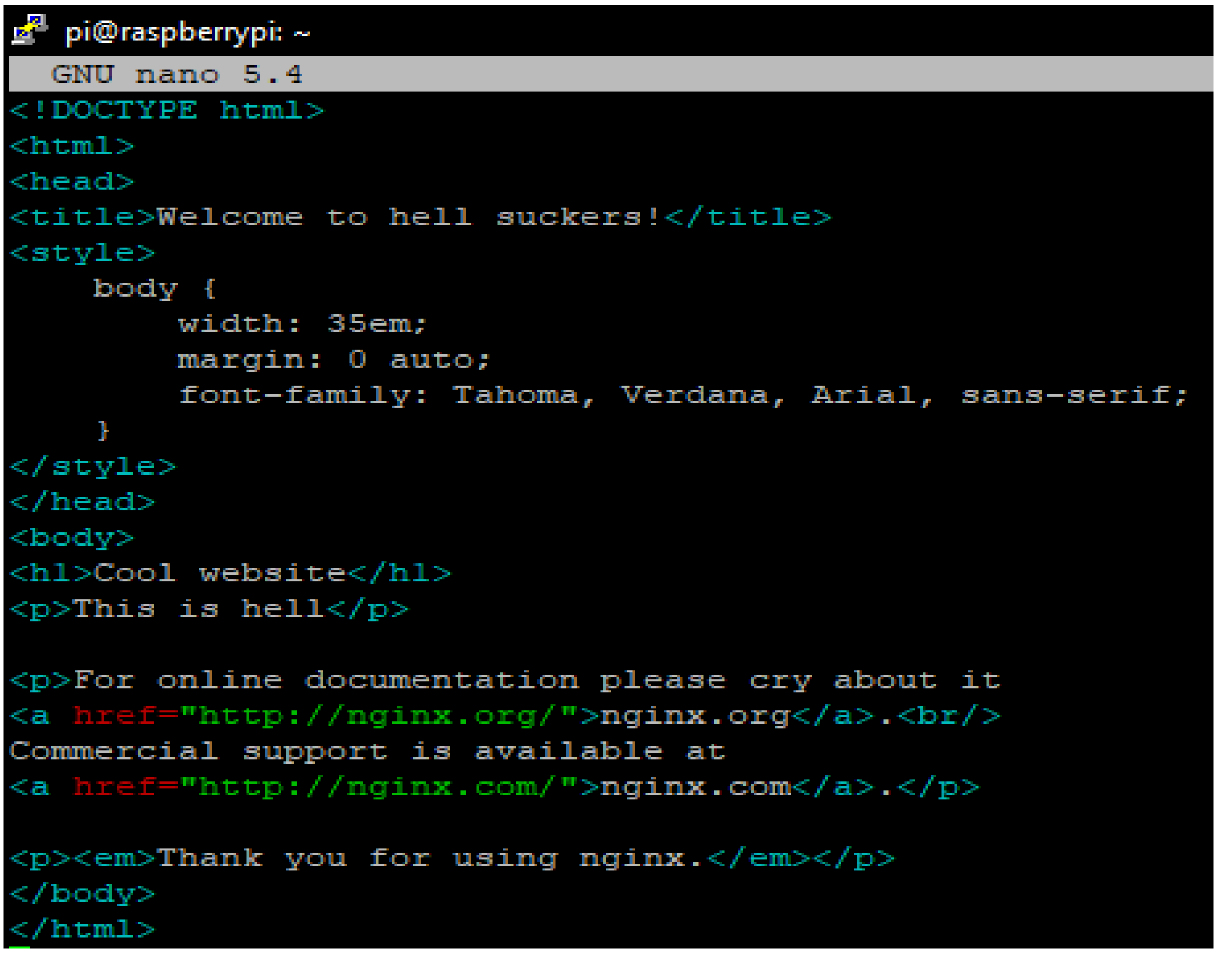
```
semihpi@broodpi3b:~ $ arp -a
? (12.0.0.10) at 00:e0:4c:68:68:ba [ether] on eth1
marvinPi (10.0.0.41) at f8:e4:3b:64:82:6a [ether] on eth0
semihpi@broodpi3b:~ $
```

Week 7 – Final Exam

In this week we had to build the following network:

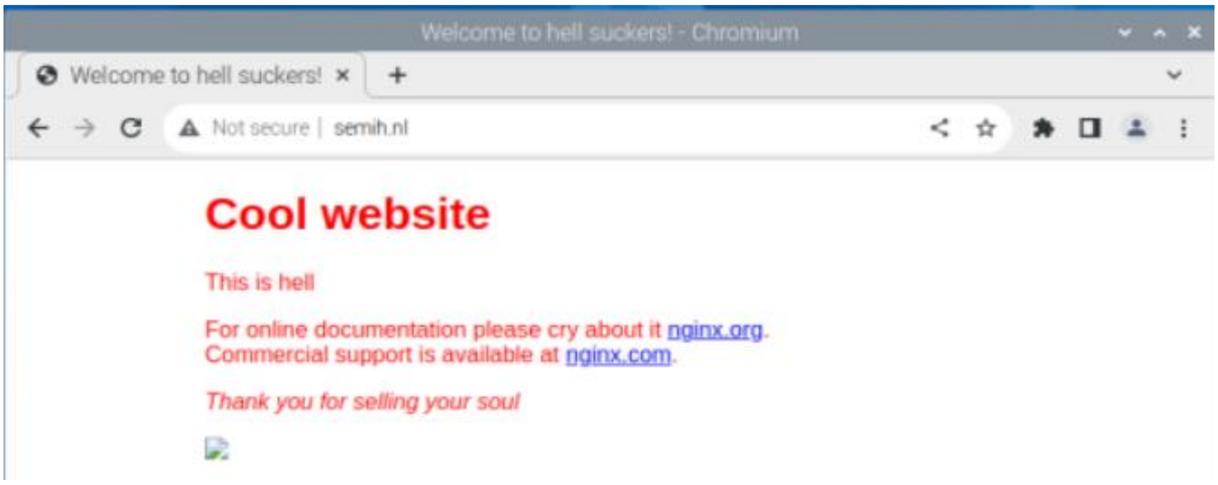


Let's start with the screenshots of the nginx server (pi@raspberrypi - 10.0.0.30). Screenshot below proves that pi@raspberrypi is the nginx server (from subnet 10.0.0.0):



```
pi@raspberrypi: ~
  GNU nano 5.4
<!DOCTYPE html>
<html>
<head>
<title>Welcome to hell suckers!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Cool website</h1>
<p>This is hell</p>

<p>For online documentation please cry about it
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Another screenshot of the nginx site and DNS working (10.0.0.30, with DNS enabled though).



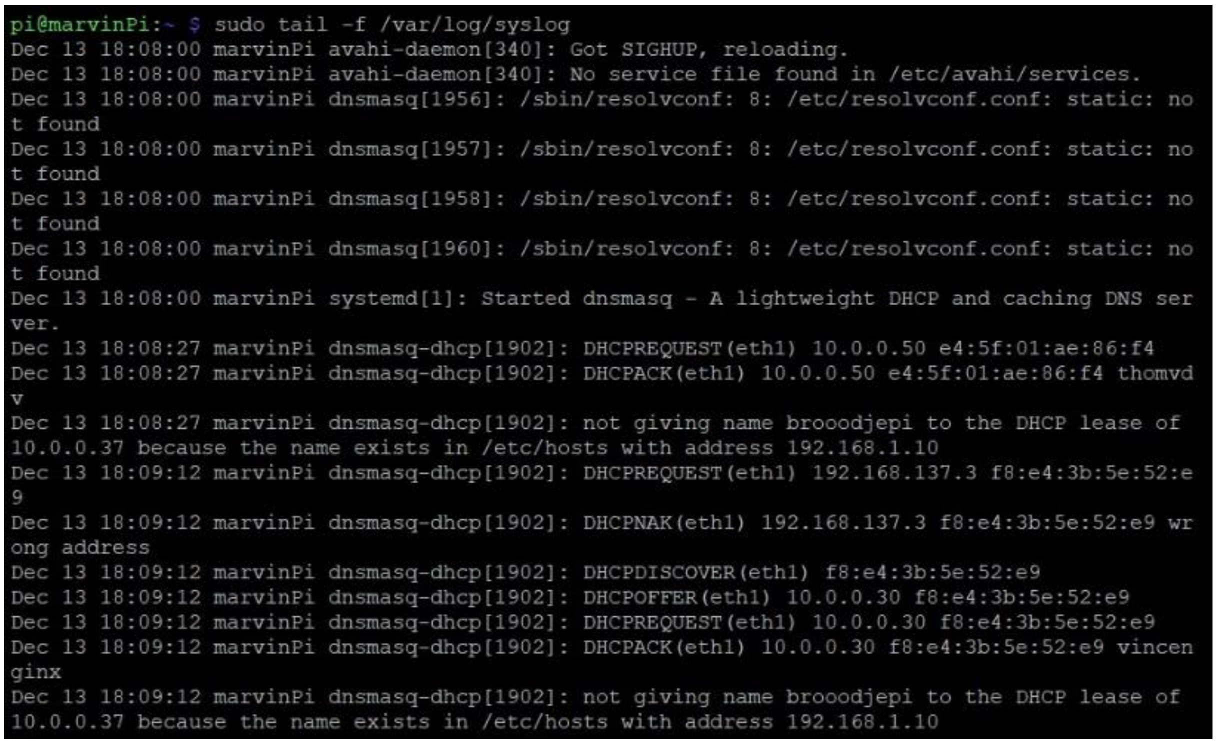
Cool website

This is hell

For online documentation please cry about it nginx.org.
Commercial support is available at nginx.com.

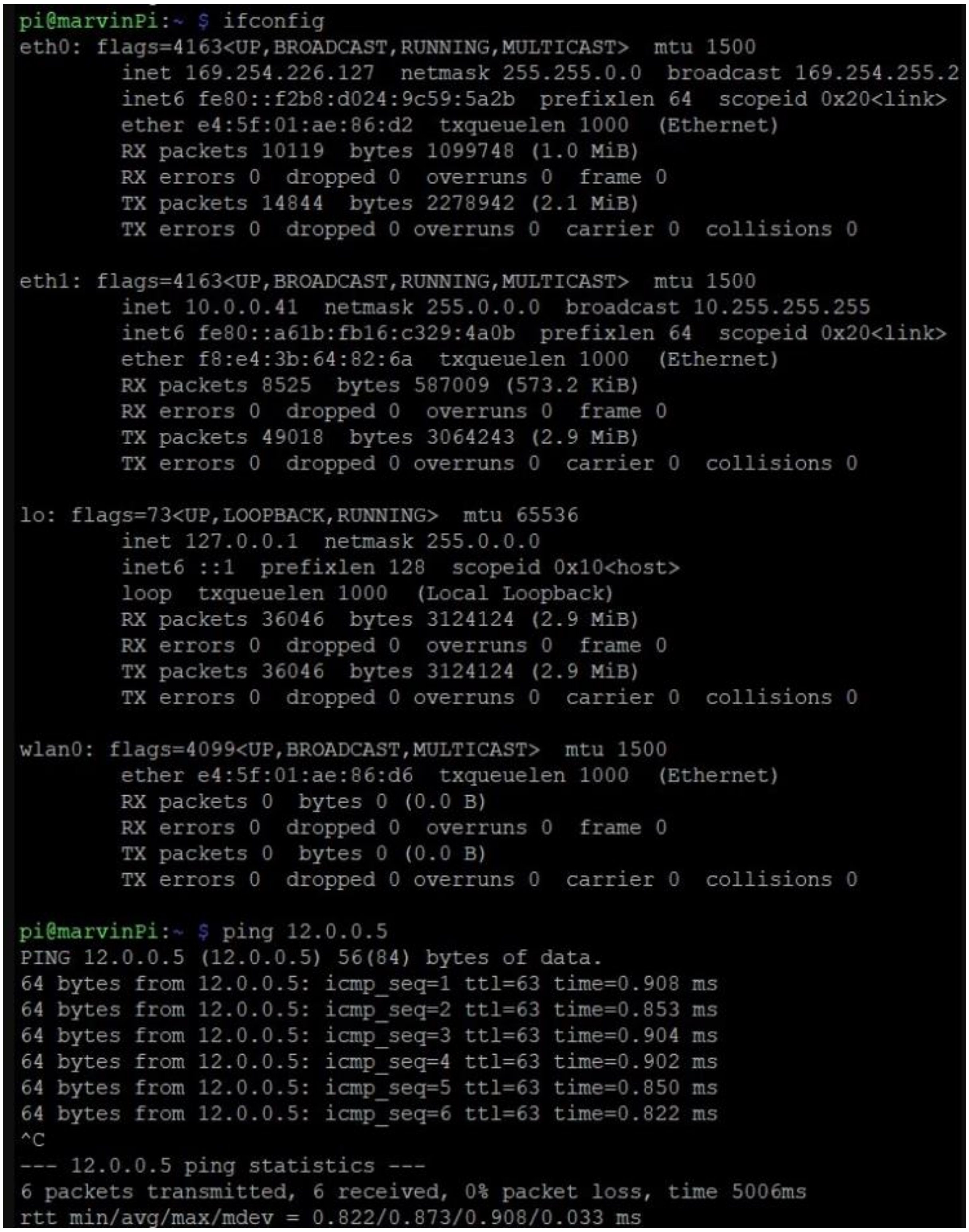
Thank you for selling your soul

Screenshot: DHCP server of subnet 10.0.0.0 (pi@marvinPi – 10.0.0.41) giving nginx server a leased IP-address (10.0.0.30).



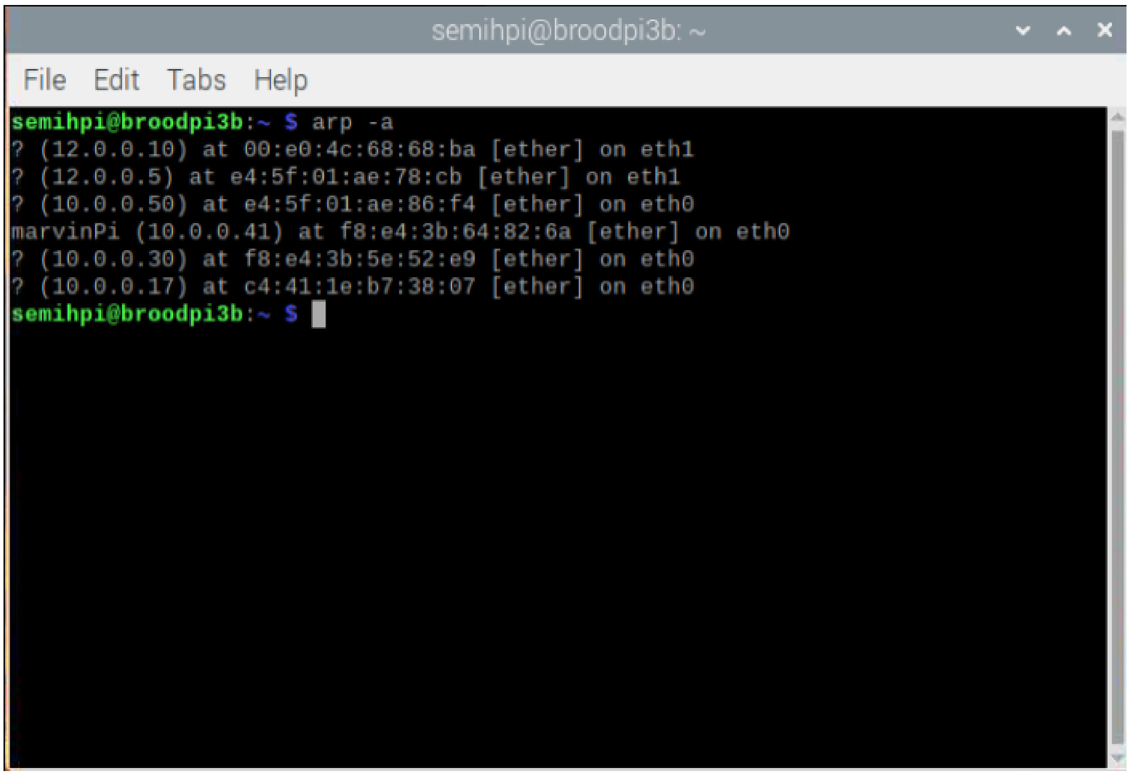
```
pi@marvinPi:~ $ sudo tail -f /var/log/syslog
Dec 13 18:08:00 marvinPi avahi-daemon[340]: Got SIGHUP, reloading.
Dec 13 18:08:00 marvinPi avahi-daemon[340]: No service file found in /etc/avahi/services.
Dec 13 18:08:00 marvinPi dnsmasq[1956]: /sbin/resolvconf: 8: /etc/resolvconf.conf: static: no
t found
Dec 13 18:08:00 marvinPi dnsmasq[1957]: /sbin/resolvconf: 8: /etc/resolvconf.conf: static: no
t found
Dec 13 18:08:00 marvinPi dnsmasq[1958]: /sbin/resolvconf: 8: /etc/resolvconf.conf: static: no
t found
Dec 13 18:08:00 marvinPi dnsmasq[1960]: /sbin/resolvconf: 8: /etc/resolvconf.conf: static: no
t found
Dec 13 18:08:00 marvinPi systemd[1]: Started dnsmasq - A lightweight DHCP and caching DNS ser
ver.
Dec 13 18:08:27 marvinPi dnsmasq-dhcp[1902]: DHCPREQUEST(eth1) 10.0.0.50 e4:5f:01:ae:86:f4
Dec 13 18:08:27 marvinPi dnsmasq-dhcp[1902]: DHCPACK(eth1) 10.0.0.50 e4:5f:01:ae:86:f4 thomvd
v
Dec 13 18:08:27 marvinPi dnsmasq-dhcp[1902]: not giving name broodjepi to the DHCP lease of
10.0.0.37 because the name exists in /etc/hosts with address 192.168.1.10
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: DHCPREQUEST(eth1) 192.168.137.3 f8:e4:3b:5e:52:e
9
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: DHCPNAK(eth1) 192.168.137.3 f8:e4:3b:5e:52:e9 wr
ong address
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: DHCPDISCOVER(eth1) f8:e4:3b:5e:52:e9
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: DHCPOFFER(eth1) 10.0.0.30 f8:e4:3b:5e:52:e9
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: DHCPREQUEST(eth1) 10.0.0.30 f8:e4:3b:5e:52:e9
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: DHCPACK(eth1) 10.0.0.30 f8:e4:3b:5e:52:e9 vincen
ginx
Dec 13 18:09:12 marvinPi dnsmasq-dhcp[1902]: not giving name broodjepi to the DHCP lease of
10.0.0.37 because the name exists in /etc/hosts with address 192.168.1.10
```

Marvin has also captured another screenshot with proof that he was able to ping Ferhat his Pi from another subnet (so a ping from 10.0.0.41 to 12.0.0.5):



```
pi@marvinPi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 169.254.226.127  netmask 255.255.0.0  broadcast 169.254.255.2
        inet6 fe80::f2b8:d024:9c59:5a2b  prefixlen 64  scopeid 0x20<link>
        ether e4:5f:01:ae:86:d2  txqueuelen 1000  (Ethernet)
        RX packets 10119  bytes 1099748 (1.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14844  bytes 2278942 (2.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.41  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::a61b:fb16:c329:4a0b  prefixlen 64  scopeid 0x20<link>
        ether f8:e4:3b:64:82:6a  txqueuelen 1000  (Ethernet)
        RX packets 8525  bytes 587009 (573.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 49018  bytes 3064243 (2.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 36046  bytes 3124124 (2.9 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 36046  bytes 3124124 (2.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether e4:5f:01:ae:86:d6  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@marvinPi:~ $ ping 12.0.0.5
PING 12.0.0.5 (12.0.0.5) 56(84) bytes of data.
64 bytes from 12.0.0.5: icmp_seq=1 ttl=63 time=0.908 ms
64 bytes from 12.0.0.5: icmp_seq=2 ttl=63 time=0.853 ms
64 bytes from 12.0.0.5: icmp_seq=3 ttl=63 time=0.904 ms
64 bytes from 12.0.0.5: icmp_seq=4 ttl=63 time=0.902 ms
64 bytes from 12.0.0.5: icmp_seq=5 ttl=63 time=0.850 ms
64 bytes from 12.0.0.5: icmp_seq=6 ttl=63 time=0.822 ms
^C
--- 12.0.0.5 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5006ms
rtt min/avg/max/mdev = 0.822/0.873/0.908/0.033 ms
```

Now it's time the router, which was my older Raspberry Pi (broodpi3b). Thom van der Veen used my older Pi in our network, because I already had it configurated for him (IP's: 10.0.0.2 and 12.0.0.2):

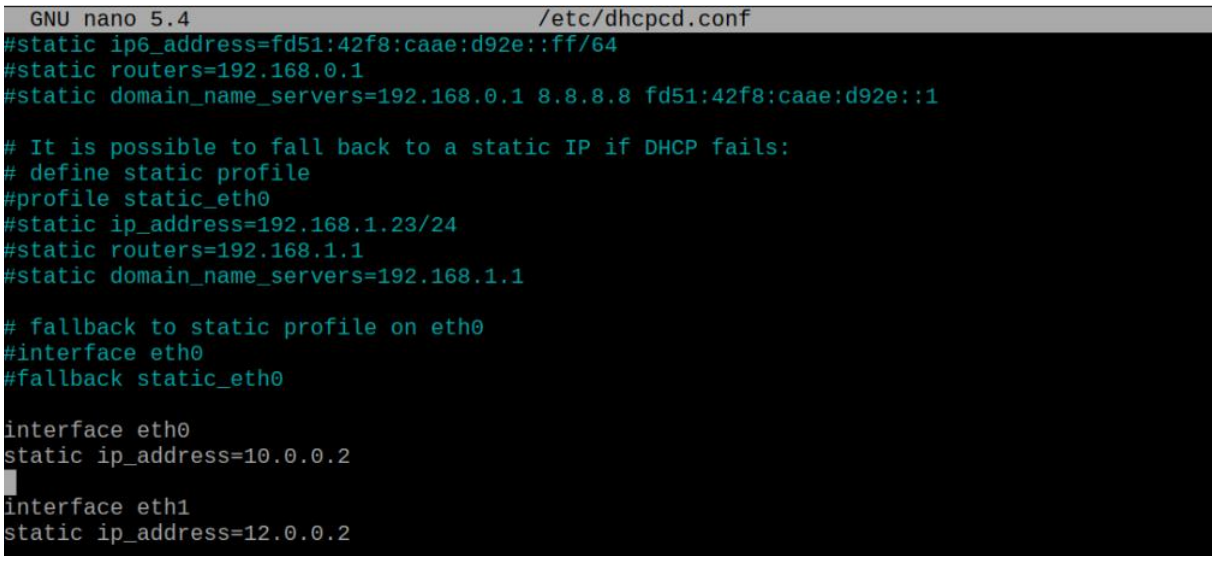


```
semihpi@broodpi3b:~ $ arp -a
? (12.0.0.10) at 00:e0:4c:68:68:ba [ether] on eth1
? (12.0.0.5) at e4:5f:01:ae:78:cb [ether] on eth1
? (10.0.0.50) at e4:5f:01:ae:86:f4 [ether] on eth0
marvinPi (10.0.0.41) at f8:e4:3b:64:82:6a [ether] on eth0
? (10.0.0.30) at f8:e4:3b:5e:52:e9 [ether] on eth0
? (10.0.0.17) at c4:41:1e:b7:38:07 [ether] on eth0
semihpi@broodpi3b:~ $
```

In this screenshot (above) you see different IP-addresses, I'll give a list of who is who:

1. 12.0.0.10 is me (Semih), the DNS server (pisemih@brooodjepi).
2. 12.0.0.5 is Ferat, the DHCP server on subnet 12.0.0.0 (pi@pino)
3. 10.0.0.50 is Thom his Raspberry Pi(, which had no function in our network, he used it for testing purposes).
4. 10.0.0.41 is Marvin, the DHCP server on subnet 10.0.0.0 (pi@marvinPi).
5. 10.0.0.30 is Vince, the nginx server (pi@raspberrypi).
6. 10.0.0.17 is also Thom, but with his other Raspberry Pi (no function, was just a DHCP server test).
7. 10.0.0.2 and 12.0.0.2, which you cannot see in this ARP list, because it is the router itself. I just wanted to mention the router too (semihpi@broodpi3b).

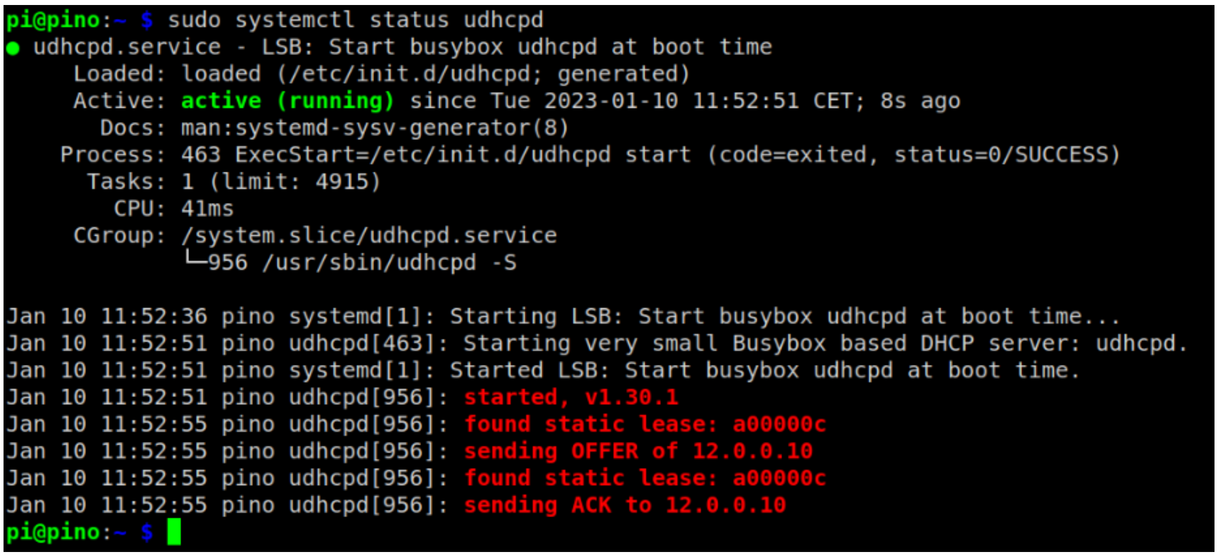
See screenshot below of dhcpcd.conf file, screenshot of configuration of router (gateways):



```
  GNU nano 5.4                        /etc/dhcpcd.conf
#static ip6_address=fd51:42f8:caae:d92e::ff/64
#static routers=192.168.0.1
#static domain_name_servers=192.168.0.1 8.8.8.8 fd51:42f8:caae:d92e::1

# It is possible to fall back to a static IP if DHCP fails:
# define static profile
#profile static_eth0
#static ip_address=192.168.1.23/24
#static routers=192.168.1.1
#static domain_name_servers=192.168.1.1

# fallback to static profile on eth0
#interface eth0
#fallback static_eth0

interface eth0
static ip_address=10.0.0.2

interface eth1
static ip_address=12.0.0.2
```

Now time for the DHCP server of subnet 12.0.0.0, which was Ferat (12.0.0.5 – pi@pino).

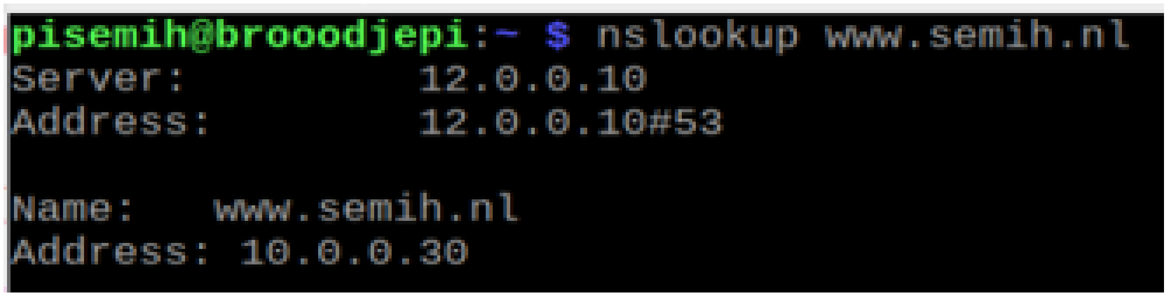
See screenshot below of proof that udhcpd DHCP server is successfully working and giving a leased IP-address to the DNS server (12.0.0.10):



```
pi@pino:~ $ sudo systemctl status udhcpd
● udhcpd.service - LSB: Start busybox udhcpd at boot time
     Loaded: loaded (/etc/init.d/udhcpd; generated)
     Active: active (running) since Tue 2023-01-10 11:52:51 CET; 8s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 463 ExecStart=/etc/init.d/udhcpd start (code=exited, status=0/SUCCESS)
      Tasks: 1 (limit: 4915)
        CPU: 41ms
     CGroup: /system.slice/udhcpd.service
             └─956 /usr/sbin/udhcpd -S

Jan 10 11:52:36 pino systemd[1]: Starting LSB: Start busybox udhcpd at boot time...
Jan 10 11:52:51 pino udhcpd[463]: Starting very small Busybox based DHCP server: udhcpd.
Jan 10 11:52:51 pino systemd[1]: Started LSB: Start busybox udhcpd at boot time.
Jan 10 11:52:51 pino udhcpd[956]: started, v1.30.1
Jan 10 11:52:55 pino udhcpd[956]: found static lease: a00000c
Jan 10 11:52:55 pino udhcpd[956]: sending OFFER of 12.0.0.10
Jan 10 11:52:55 pino udhcpd[956]: found static lease: a00000c
Jan 10 11:52:55 pino udhcpd[956]: sending ACK to 12.0.0.10
pi@pino:~ $
```

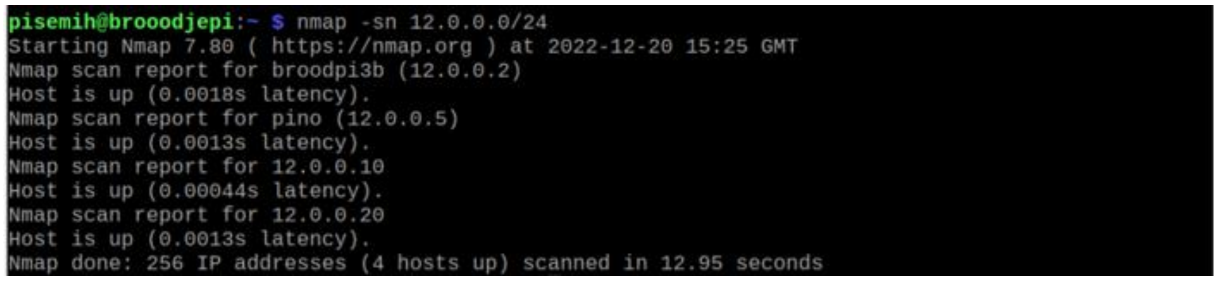
Lastly proof for the DNS server (12.0.0.10 – pisemih@brooodjepi).

Proof that DNS server works (with name server lookup):



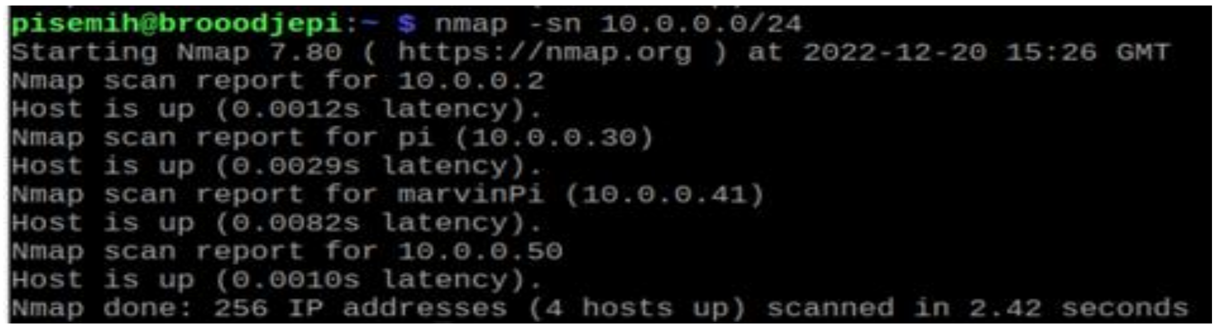
```
pisemih@brooodjepi:~ $ nslookup www.semih.nl
Server:         12.0.0.10
Address:        12.0.0.10#53

Name:   www.semih.nl
Address: 10.0.0.30
```

Proof that I was able to in see our own whole subnet (12.0.0.0/24) with the package called “nmap”:



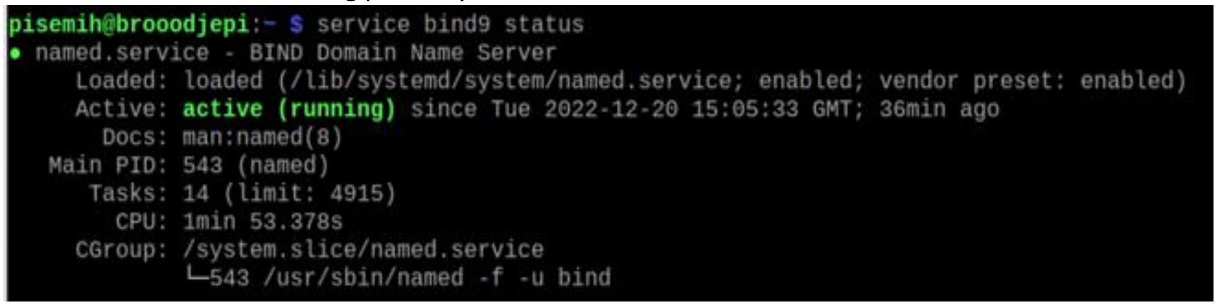
```
pisemih@brooodjepi:~ $ nmap -sn 12.0.0.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-20 15:25 GMT
Nmap scan report for broodpi3b (12.0.0.2)
Host is up (0.0018s latency).
Nmap scan report for pino (12.0.0.5)
Host is up (0.0013s latency).
Nmap scan report for 12.0.0.10
Host is up (0.00044s latency).
Nmap scan report for 12.0.0.20
Host is up (0.0013s latency).
Nmap done: 256 IP addresses (4 hosts up) scanned in 12.95 seconds
```

Proof that I was able to see everyone in the other subnet (10 .0.0.0/24):



```
pisemih@brooodjepi:~ $ nmap -sn 10.0.0.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2022-12-20 15:26 GMT
Nmap scan report for 10.0.0.2
Host is up (0.0012s latency).
Nmap scan report for pi (10.0.0.30)
Host is up (0.0029s latency).
Nmap scan report for marvinPi (10.0.0.41)
Host is up (0.0082s latency).
Nmap scan report for 10.0.0.50
Host is up (0.0010s latency).
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.42 seconds
```

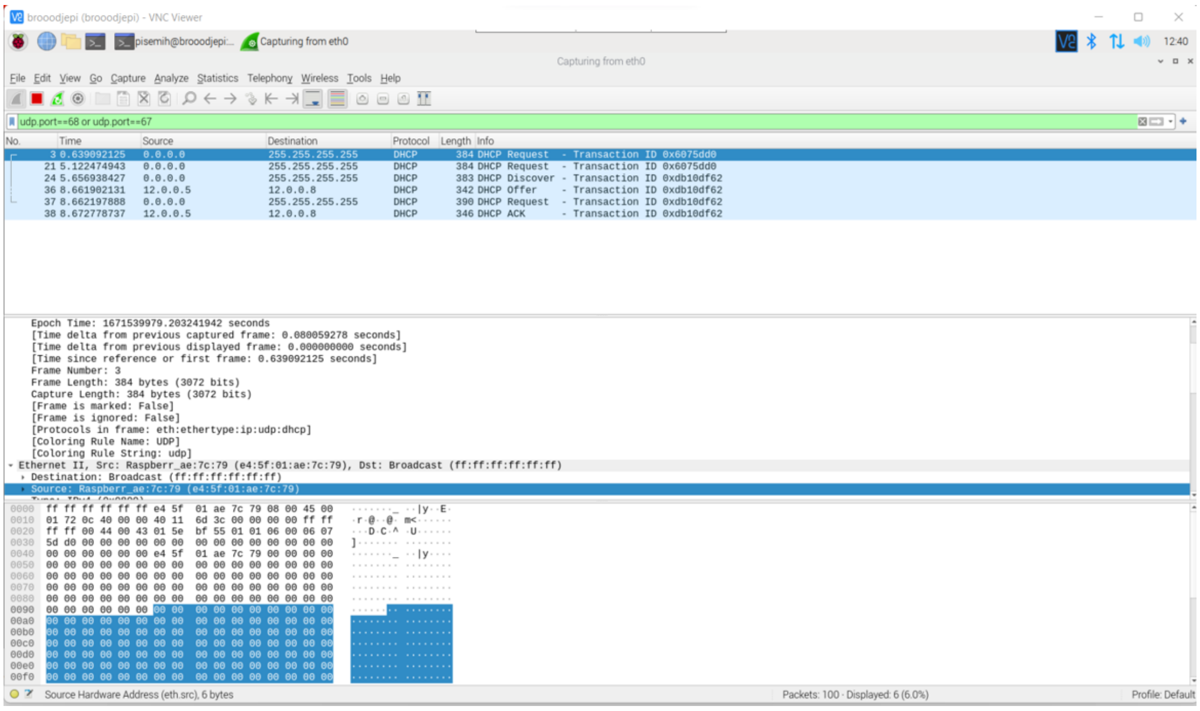
Proof that BIND9 was running perfectly:



```
pisemih@brooodjepi:~ $ service bind9 status
● named.service - BIND Domain Name Server
     Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-12-20 15:05:33 GMT; 36min ago
       Docs: man:named(8)
   Main PID: 543 (named)
      Tasks: 14 (limit: 4915)
        CPU: 1min 53.378s
     CGroup: /system.slice/named.service
             └─543 /usr/sbin/named -f -u bind
```

The settings of the DHCP and DNS servers are all explained in the weeks that we had to configure them, so week 3 and week 4. The only difference is that we had more clients and other IP-addresses.

Lastly I also have a screenshot of using Wireshark, because we had an issue that the DNS server would keep getting a weird IP-address (like 192.168 etc.) from the DHCP server. So we had to find out what the problem was and therefore we used Wireshark:



brooodjepi (brooodjepi) - VNC Viewer
pisemih@brooodjepi:...
Capturing from eth0
12:40
Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
udp.port==68 or udp.port==67
```

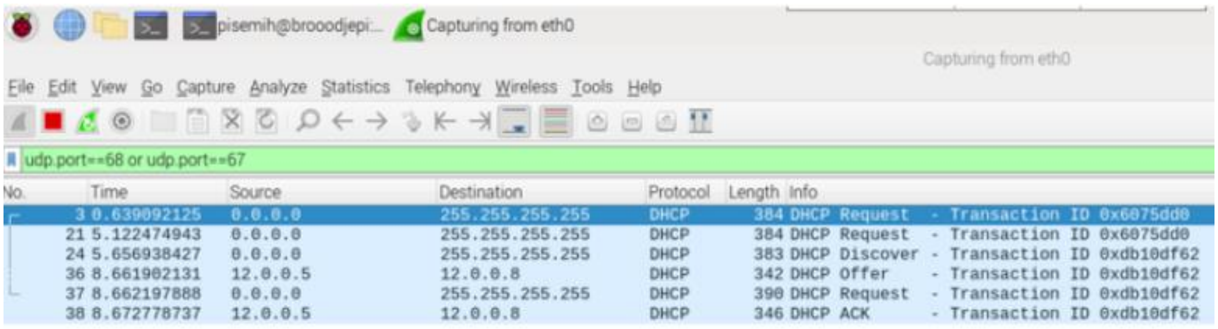
```
No.   Time          Source      Destination        Protocol  Length Info
   3 0.639092125   0.0.0.0     255.255.255.255    DHCP      384 DHCP Request  - Transaction ID 0x6075dd0
  21 5.122474943   0.0.0.0     255.255.255.255    DHCP      384 DHCP Request  - Transaction ID 0x6075dd0
  24 5.656938427   0.0.0.0     255.255.255.255    DHCP      383 DHCP Discover - Transaction ID 0xdb10df62
  36 8.661902131   12.0.0.5    12.0.0.8           DHCP      342 DHCP Offer    - Transaction ID 0xdb10df62
  37 8.662197888   0.0.0.0     255.255.255.255    DHCP      390 DHCP Request  - Transaction ID 0xdb10df62
  38 8.672778737   12.0.0.5    12.0.0.8           DHCP      346 DHCP ACK      - Transaction ID 0xdb10df62
```

```
    Epoch Time: 1671539979.203241942 seconds
    [Time delta from previous captured frame: 0.080059278 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.639092125 seconds]
    Frame Number: 3
    Frame Length: 384 bytes (3072 bits)
    Capture Length: 384 bytes (3072 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:udp:dhcp]
    [Coloring Rule Name: UDP]
    [Coloring Rule String: udp]
▾ Ethernet II, Src: Raspberr_ae:7c:79 (e4:5f:01:ae:7c:79), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Source: Raspberr_ae:7c:79 (e4:5f:01:ae:7c:79)
```

```
0000  ff ff ff ff ff ff e4 5f  01 ae 7c 79 08 00 45 00   ·······_ ··|y··E·
0010  01 72 0c 40 00 00 40 11  6d 3c 00 00 00 00 ff ff   ·r·@··@· m<······
0020  ff ff 00 44 00 43 01 5e  bf 55 01 01 06 00 06 07   ···D·C·^ ·U······
0030  5d d0 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ]······· ········
0040  00 00 00 00 00 00 e4 5f  01 ae 7c 79 00 00 00 00   ·······_ ··|y····
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00d0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00e0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
00f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
```

Source Hardware Address (eth.src), 6 bytes
Packets: 100 · Displayed: 6 (6.0%)
Profile: Default

Here is an enhanced version of that picture:



pisemih@brooodjepi:...
Capturing from eth0
Capturing from eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
udp.port==68 or udp.port==67
```

```
No.   Time          Source      Destination        Protocol  Length Info
   3 0.639092125   0.0.0.0     255.255.255.255    DHCP      384 DHCP Request  - Transaction ID 0x6075dd0
  21 5.122474943   0.0.0.0     255.255.255.255    DHCP      384 DHCP Request  - Transaction ID 0x6075dd0
  24 5.656938427   0.0.0.0     255.255.255.255    DHCP      383 DHCP Discover - Transaction ID 0xdb10df62
  36 8.661902131   12.0.0.5    12.0.0.8           DHCP      342 DHCP Offer    - Transaction ID 0xdb10df62
  37 8.662197888   0.0.0.0     255.255.255.255    DHCP      390 DHCP Request  - Transaction ID 0xdb10df62
  38 8.672778737   12.0.0.5    12.0.0.8           DHCP      346 DHCP ACK      - Transaction ID 0xdb10df62
```

In the end, we found out it was because of Ferhat's laptop. He has Linux OS and did not know he had internet sharing turned on. With Wireshark we found out that the DNS server was getting an IP address from his Laptop via internet sharing. After we fixed the problem everything worked happily. Thus, after the fix, the DNS server was getting an IP-address from the Pi's DHCP server, hurray!