SYS2 (NET) Laboratory 3 : DNS, DHCP and NTP

These weekly lab scripts have been written as a self-study resource i.e. an activity to be worked on at your own pace, in your own time. The timetabled practical sessions are an opportunity to get advice and guidance. Therefore, you may not always finish each lab during the timetabled session, but do finish each lab's tasks in your own time. Hardware labs are open Mon-Fri, 9:00-17:00.

The aim of this lab is to introduce some of the Internet's "housekeeping" protocols. Protocols that allow users to connect and navigate through a network. Before starting this lab make sure you have watched video **Lecture 2B**. Intuitively you would guess that this core functionality would be implemented by lower level protocols within our protocol stack shown in figure 1, but actually we are still looking at protocols in the application layer. Its surprising how many protocols live in this layer. Its all a question of the functionality used i.e. rule number 1 of a protocol stack: don't replicate functionality. Like the protocols we looked at in the last lab these new protocols use a client-server model, however, as the information transferred is at a machine-to-machine level we see a move away from plain text messages, as used in HTTP and SMTP, to binary data. Therefore, we will be looking at a range of different network tools and Wireshark, to convert and display this information in a human readable format. At the end of this practical you will understand how :

- IP addresses and host names are registered and how / where this data is stored.
- The DNS protocol is used to resolve host names into IP addresses, and the different types of records that are stored in this distributed database.
- Static and dynamic IP addresses are assigned to hosts on a network.
- Computers know what time it is.

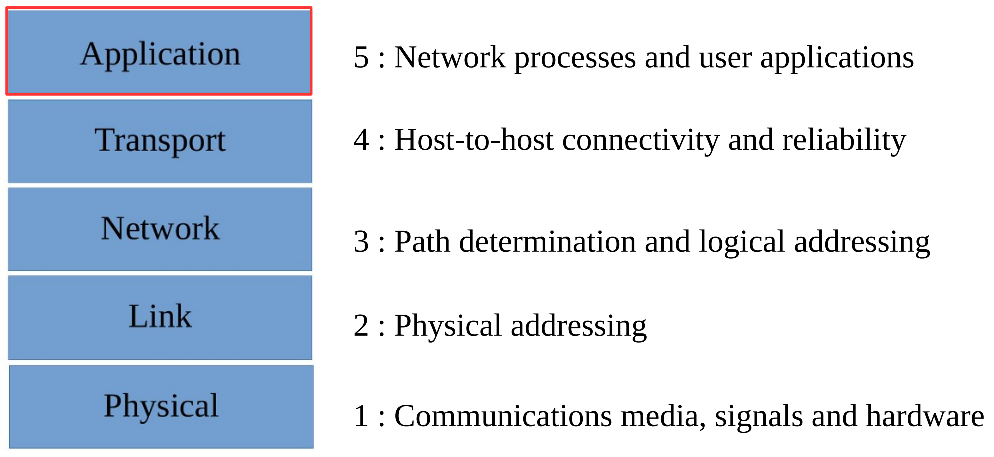

Application
5 : Network processes and user applications
Transport
4 : Host-to-host connectivity and reliability
Network
3 : Path determination and logical addressing
Link
2 : Physical addressing
Physical
1 : Communications media, signals and hardware

Figure 1 : The Internet protocol stack

Task 1

In lab 1 we saw how a host connected to a network is identified by its IP address. In lab 2 we introduced the concept of an Uniform Resource Locator (URL) shown in figure 2, a human readable label that can be used to identify machines and other resources connected to our local network, or the wider Internet. This raises the questions how do we map a domain name to an IP address, or vice-versa?

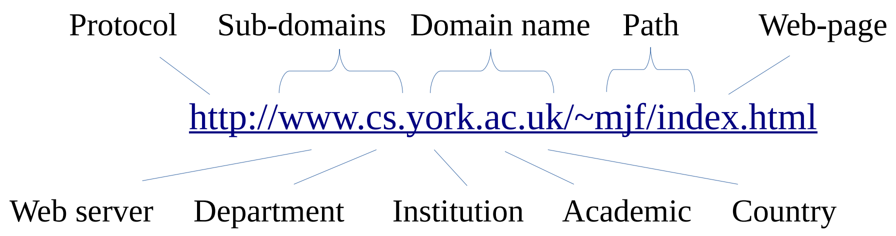


Figure 2 : an URL (uniform resource locator)

When the Internet first started this was done manually via a `hosts` file. A file containing a list of user friendly domain names and their associated IP addresses (plain text) e.g. the entry for the department's web server could of looked like this:

```
144.32.128.93       www.york.ac.uk
```

Note, `hosts` files are still used on your computer today, simplifying the conversion of frequently used names / machines that may not be Internet visible. However, as today's Internet is more of a dynamic place than its original beginnings, static mappings like these may not always be appropriate e.g. a web server could be moved to a different host/IP address owing to a hardware fault or increased network traffic (load balancing) etc.

Open a Secure Shell (SSH) on Pi-1, as described in lab 1, click on the start button and select the command prompt.

```
-> Command Prompt
```

In the command prompt enter :

```
ssh pi@192.168.X.1              (X=Box/Desk)
```

To view the contents of Pi-1's `hosts` file enter the command:

```
cat /etc/hosts
```

Task : examine this file, if you were to enter the command "`ping pi-2`" would this work? Is this name present in this file? Also test out the telnet and SSH commands e.g. "`telnet pi-3`" etc.

As the internet grew the `hosts` file moved online (onto a server) and was stored in the WHOIS directory. This made the `hosts` file globally accessible, but it was still manually controlled. By the 1980s as the Internet expanded this approach became impractical and needed to be automated. However, the core concept of the domain name remains.

One of the original aims of the domain name was to allow users to determine the hosts physical location and purpose (however, this has been relaxed over time) e.g. `uk`, `fr`, `us`, `de` etc. Domain names are read right to left i.e. the top level domain in figure 2 is `uk`. This is then broken down into further sub-domains until you reach the authoritative-domain for that organisation e.g. for the University this is `york.ac.uk` i.e. a common root that all following sub-domains share.

Note, the WHOIS directory still lives on in the sense of the `whois` protocol, allowing users to identify who owns a domain name or a block of IP addresses.

RFC 3912: https://datatracker.ietf.org/doc/html/rfc3912

This protocol can be used at the command line by typing: `whois <domain-name>` or via a web browser. Open your browser of choice on the PC and enter the URL:

`https://www.whois.com/whois/`

Task : to test this protocol enter the domain name: `york.ac.uk` and click on the search button. This will return the registrar data. You can also use `whois` to determine who owns a block of IP addresses. Enter the department's web server IP address: 144.32.128.40, who is this address registered to?

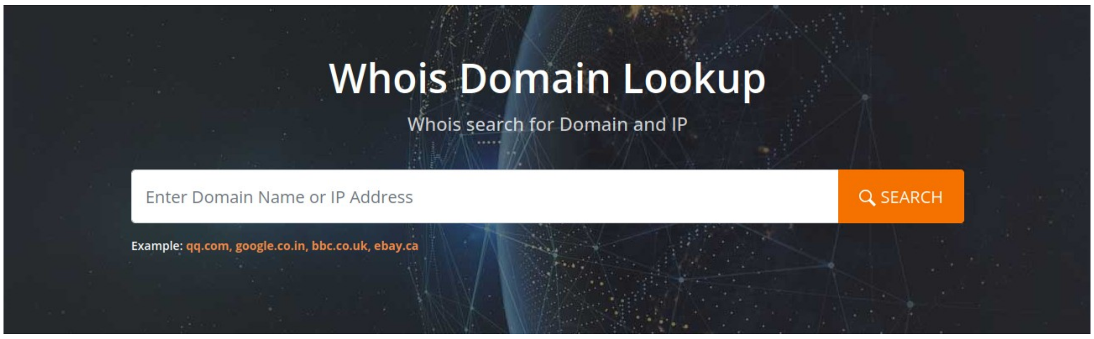

Whois Domain Lookup
Whois search for Domain and IP
Enter Domain Name or IP Address
SEARCH
Example: qq.com, google.co.in, bbc.co.uk, ebay.ca

Figure 3 : who owns an URL

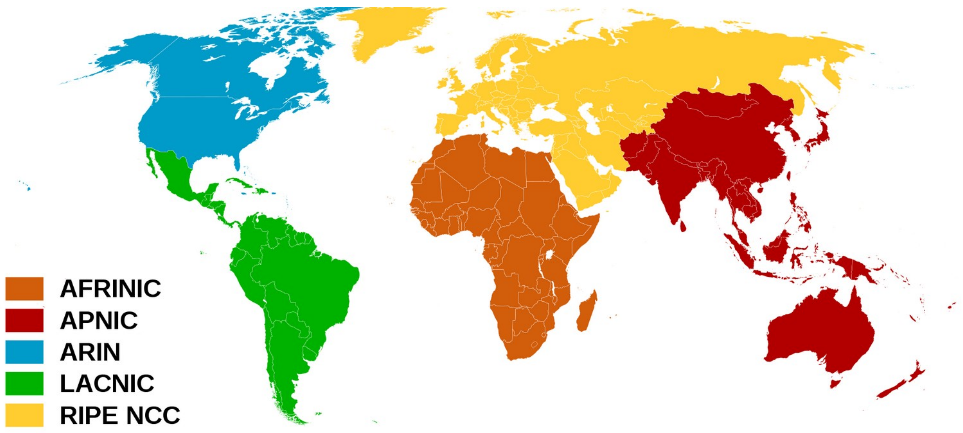


Figure 4 : regional Internet registries

Listed within all the `whois` data you will find references to JISC: Joint Information Systems Committee, which is a member of RIPE: Réseaux IP Européens, the network coordination centre. To find out more information about this organisation you can watch this video on your the PC, but do keep the volume on low :).

https://www-static.ripe.net/static/videos/misc/What_is_the_RIPE_NCC.mp4

Alternatively, you can find out more on their website (video also available there):

https://www.ripe.net/

There are five such Internet registries as shown in figure 4, controlling the allocation of IP addresses and other Internet resources, and also maintaining the WHOIS registries.

Task 2

In the previous task we have seen how domain names are represented and registered, and how IP addresses are allocated to hosts. The protocol that now ties these two elements together is the : Domain Name System (DNS), a distributed database that maps an IP address to a domain name i.e. the protocol that replaced the static `hosts` file.

The Domain Name System (DNS) protocol (RFC 1035: https://tools.ietf.org/html/rfc1035) was created in 1983 and uses a client server model, default port 53. This protocol allows hosts to translate domain names into IP addresses and vice versa. This information is stored in a hierarchical distributed database i.e. replicated servers, in order to:

- Improves reliability, removing Single Points of Failure (SPF).
- Reduces network load, queries resolved by multiple machines.
- Reduces delays, reduces RTT by using local servers.
- Reduce processing load, database management and updates.

For a DNS client (your PC) to perform a forward lookup i.e. convert a domain name into an IP address, it first has to know the IP address of its local DNS server. This can be hard-coded into a config file (`/etc/resolv.conf`), or passed to the PC when it connects to a network (we will see this when we look at DHCP later). The DNS client passes a query to its DNS server. If the server has already previously performed this lookup it can fulfil this request from its cache, otherwise it will need to consult a root DNS server.

Conceptually there are 13 logical root name servers A – M, as shown in figure 5, being assigned the domain names: `a.root-servers.net` to `m.root-servers.net`. As previously discussed in the lecture these are then replicated across the world, using approximately 1300 servers. These 13 root servers are assigned "well know" IP addresses i.e. you have to know where to start. To find the location of these root servers, open your preferred web browser and enter the flowing URL :

https://root-servers.org/

Using the zoom in button + zoom in on the UK. Then click on the hub icon closet to York, finally click on the location icons to see each DNS servers location, as shown in Appendix B.

Note, when the Internet first started most root DNS servers were located in America, as a result if you lived in the UK the round trip delays on DNS queries were quite significant. To overcome this issue caching was heavily used. Each cache entry has a Time To Live (TTL) field to force updates after a specified time i.e. to ensure IP addresses are up to date. Nowadays these servers are replicated around the world so you are never that far from a root server. However, DNS caching is still very important to help reduce server loads i.e. number of queries to any one server.

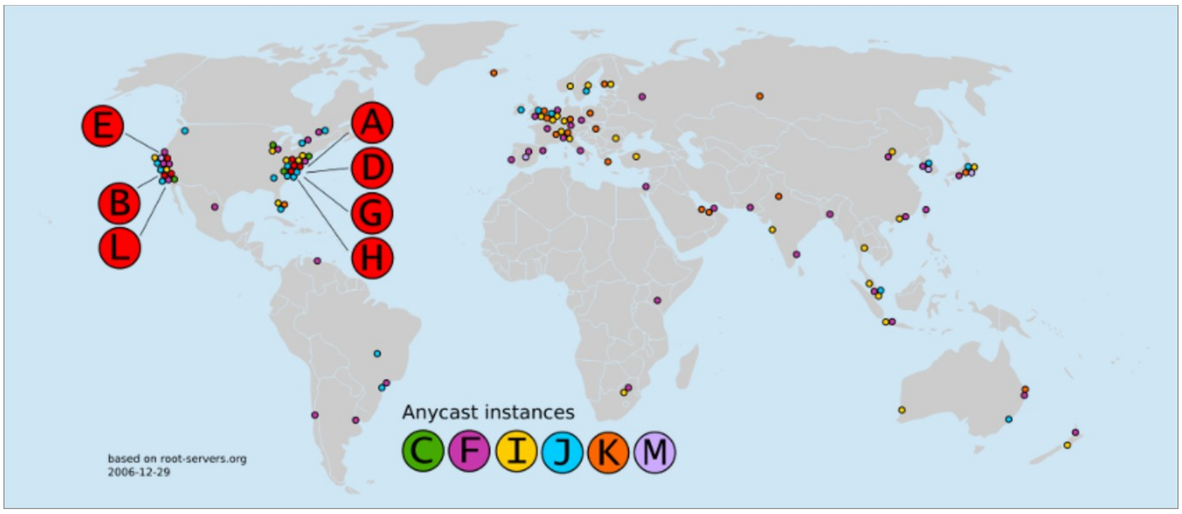


Figure 5 : root name servers

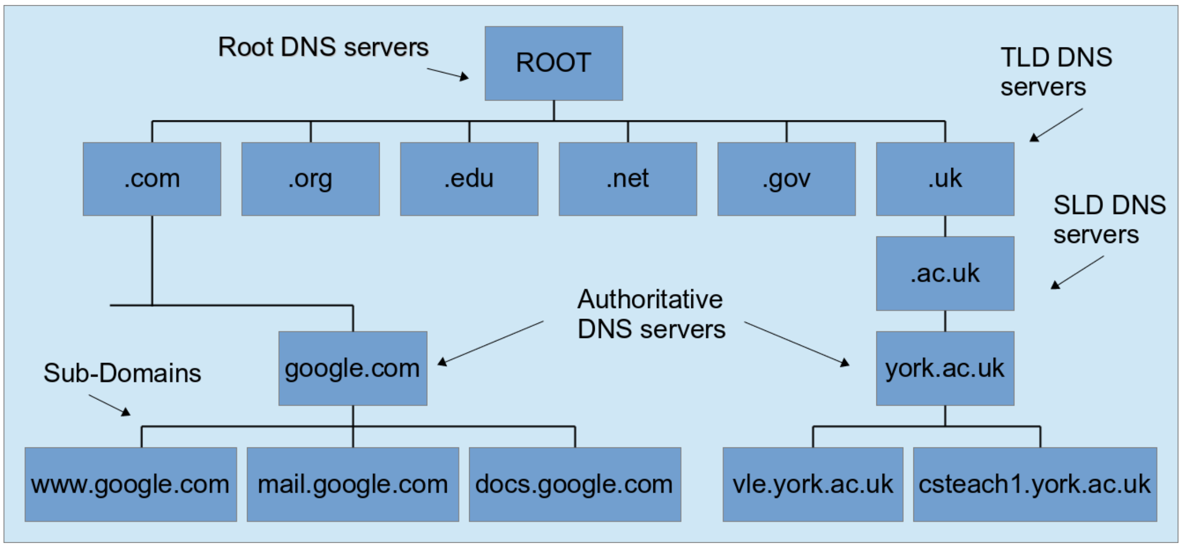


Figure 6 : DNS server hierarchy

DNS uses a distributed database, therefore, root DNS servers will not know the IP address of the requested domain name, but they do know the IP addresses of Top Level Domain (TLD) name servers that may, as shown in figure 6. The DNS query is passed onto the correct TLD server e.g. for `york.ac.uk` this will be `.uk`. This process is repeated until you get to an authoritative DNS server that does know the IP address of the domain name you are looking for. This returns the requested IP address to your local DNS server, that then passes it to the DNS client.

Note, the local DNS server will also add this IP address to its cache, such that the next time this DNS lookup is requested it can be fulfilled locally without having to go through the previous process. Our local DNS server is: `ofns0.york.ac.uk.` (144.32.128.242).

Task : using https://root-servers.org what is the closest root DNS server to York? As the crow flies :)

DNS forward lookups can be performed by a user using the command line tool: `dig`. On the PC click on the start button and select the command prompt.

```
-> Command Prompt
```

At the command prompt enter the following command :

```
dig
```

Running `dig` without any parameters will return the domain names and IP addresses of the root DNS servers, as shown in Appendix C. This output is arranged in the following columns :

NAME	: fully qualified domain name
TTL	: record time to live in seconds.
CLASS	: record class, normally IN – Internet.
TYPE	: record type e.g. NS – name server, A – IPv4, AAAA – IPv6, MX – mail exchange ...
RDATA	: record additional data.

Note, for a list of record-type codes : https://en.wikipedia.org/wiki/List_of_DNS_record_types

To identify the DNS servers used to convert the domain name of the universities webserver : `www.york.ac.uk`, we just need to work backwards from the top level domain i.e. dig through different DNS zones and their name servers by entering the following `dig` commands :

```
dig NS uk
dig NS ac.uk
dig NS york.ac.uk
```

Different DNS zones

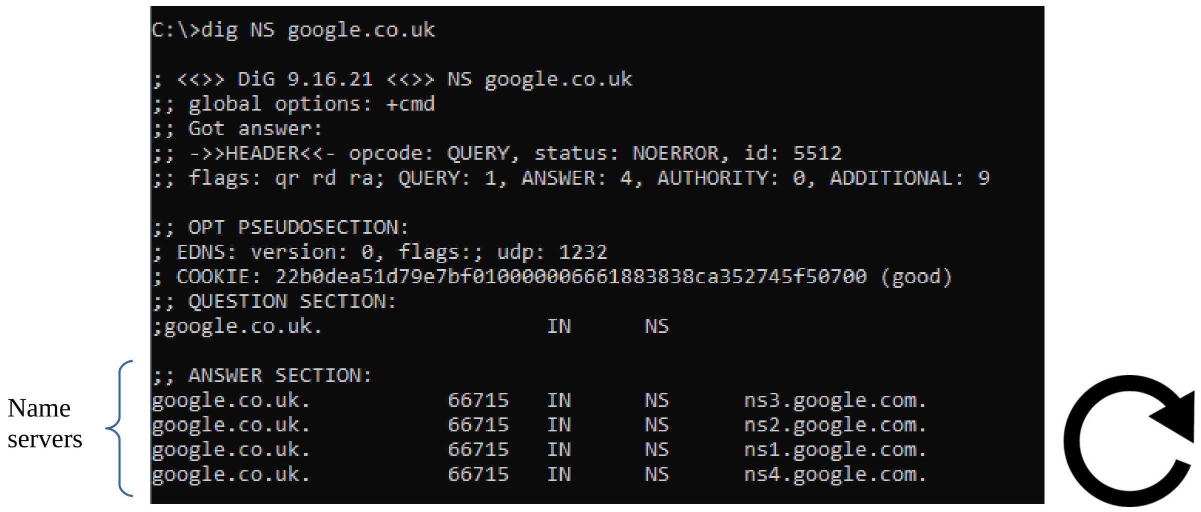
<u>Task</u> : perform the above dig commands. Can you identify what each section is telling you? What do the various record <u>types</u> indicate, as described in the above link.

For high load DNS servers a common technique to help reduce loading is to perform round-robin DNS server allocation from a pool of authoritative DNS servers i.e. each time a DNS query is performed the DNS server will return a different server IP address i.e. the next one in the list. To see round-robin DNS in action perform the command :

```
dig NS google.co.uk
```

In the `ANSWER SECTION` of this query will be a list of name servers for the `google.co.uk` domain, as shown in figure 7. Record this output and then perform the same command four more times, recording the output each time. You should see that the name at the top of the list changes each time, spreading the load across these multiple machines.

Note, the four name servers listed are identical i.e. replicated to spread load and increase reliability. You will also see that the time to live (TTL) field has been decremented each time you perform a dig i.e. time left until the local DNS server will refresh this cached entry. The TTL for each entry can vary depending on how critical it is. If an TTL entry is 86400 seconds, then if this server was to go down the IP address may be held in some DNS caches for 24 hours. Therefore, for mission critical systems 300 seconds may be more appropriate i.e. updated every 5 minutes.



```
C:\>dig NS google.co.uk

; <<>> DiG 9.16.21 <<>> NS google.co.uk
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 5512
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 9

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 22b0dea51d79e7bf01000000006661883838ca352745f50700 (good)
;; QUESTION SECTION:
;google.co.uk.                  IN      NS

;; ANSWER SECTION:
google.co.uk.           66715   IN      NS      ns3.google.com.
google.co.uk.           66715   IN      NS      ns2.google.com.
google.co.uk.           66715   IN      NS      ns1.google.com.
google.co.uk.           66715   IN      NS      ns4.google.com.
```

Figure 7 : Google Name Servers (NS)

Task : using the `dig` command identify the IP address of the google web-server, name-server and mail-server.

```
dig A google.co.uk
dig NS google.co.uk
dig MX google.co.uk
```

Note, when digging you will need to issue the correct `dig` classes e.g. `A`, `NS`, `MX` etc, you need to tell the DNS server what you are digging for :). Also during some searches you may need to dig multiple times i.e. use the results from one dig as the inputs of your next dig.

Task 3

As the Raspberry Pi system used in the lab is not connected to the Internet they can not perform the types of DNS lookups we have previously seen i.e. they can't access the root servers. However, they can perform DNS lookups using the local name servers implemented in the MikroTik router, shown in figure 8 and lab's Pi-Hole DNS server.

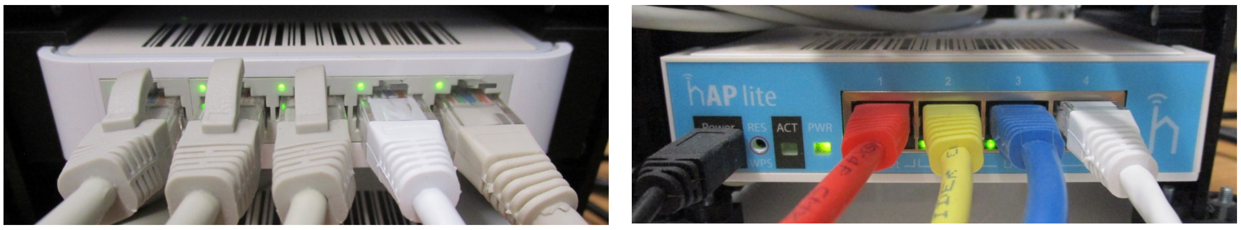


Figure 8 : switch (left) MikroTik router (right)

Note, hubs, switches and routers may look the same, but they are very different network devices and have specific purposes. However, for the moment you can think of these devices as something that electrically joins network cables together, the devices that connect our hosts together to form a network. We will be looking at these devices in a lot more detail in a later lab.

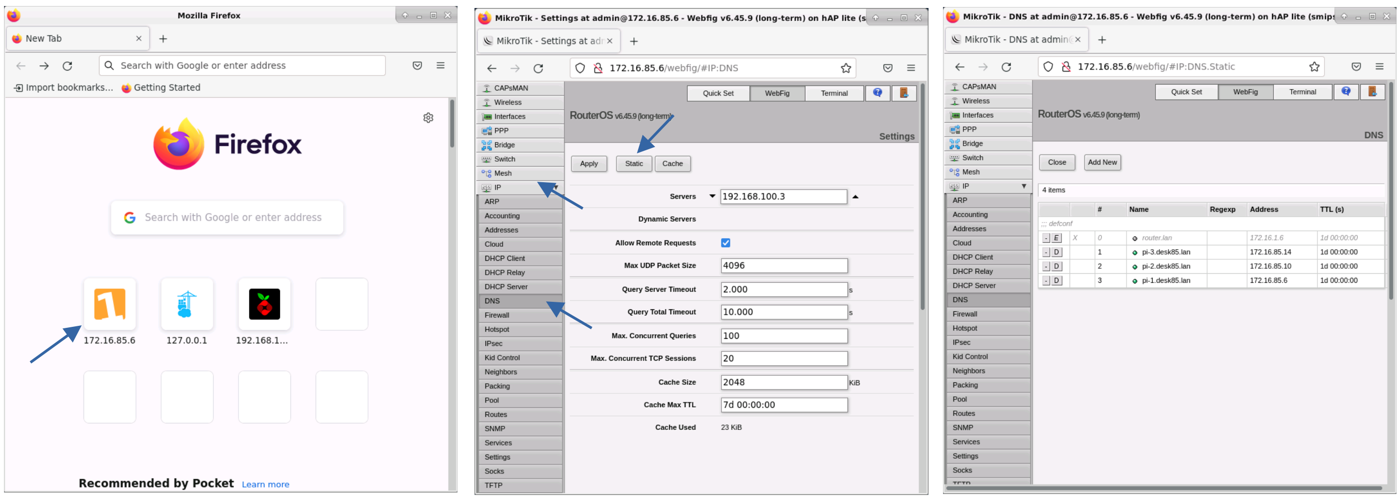
To view the MicroTik's configuration open a VNC session on Pi-1, then launch a web browser on Pi-1 by clicking on the icon on the lower toolbar. If needed refer back to lab 1 for more information on how to start a VNC session. Within the browser either click on the short cut icon, or enter the routers IP address:

```
172.16.X.6                    (X=Box/Desk)
```

As always the password is : `12345`. This will automatically open the router's Web Config page, as shown in figure 9. On the side options panel click on:

```
IP -> DNS
```

Finally click on the `Static` button to view the fixed DNS entries for your Raspberry Pi system.



172.16.85.6/webfig/#IP:DNS

Servers	192.168.100.3
Dynamic Servers	
Allow Remote Requests	✓
Max UDP Packet Size	4096
Query Server Timeout	2.000 s
Query Total Timeout	10.000 s
Max. Concurrent Queries	100
Max. Concurrent TCP Sessions	20
Cache Size	2048 KiB
Cache Max TTL	7d 00:00:00
Cache Used	23 KiB

172.16.85.6/webfig/#IP:DNS.Static

4 items

		#	Name	Regexp	Address	TTL (s)
;;; defconf						
- E	X	0	router.lan		172.16.1.6	1d 00:00:00
- D		1	pi-3.desk85.lan		172.16.85.14	1d 00:00:00
- D		2	pi-2.desk85.lan		172.16.85.10	1d 00:00:00
- D		3	pi-1.desk85.lan		172.16.85.6	1d 00:00:00

Figure 9 : MikroTik router DNS server setup

Task : using the terminal on Pi-1 check that the local DNS server is working correctly by pinging:

```
pi-Y.deskX.lan     (X=Box/Desk=04 to 85, Y=1,2,3)
```

that is, ping pi-1, pi-2 and pi-3 using their fully qualified names rather than their IP addresses. What IP address does the ping command use for each Pi, do these match the static DNS entries on the router? Do you understand why pinging the name `pi-2.deskX.lan` and pinging the name `pi-2` are resolved to different IP addresses?

For the moment each desk in the lab can be considered a separate DNS zone e.g. `desk98.lan` and `desk99.lan` are separate parts of the DNS name space.

Task : What happens when you ping another desk e.g. `pi-1.desk90.lan`?

What you should see is that the ping responds with a DNS failure error message as it is unable to resolve this name. In these situations the MikroTik will perform a **recursive** DNS lookup i.e. the query is offloaded to other DNS servers to track down the IP address and return it to the client. This is in contrast to an **iterative** DNS query, where the local DNS server communicates directly with each DNS server involved in the lookup, as shown in figure 10.

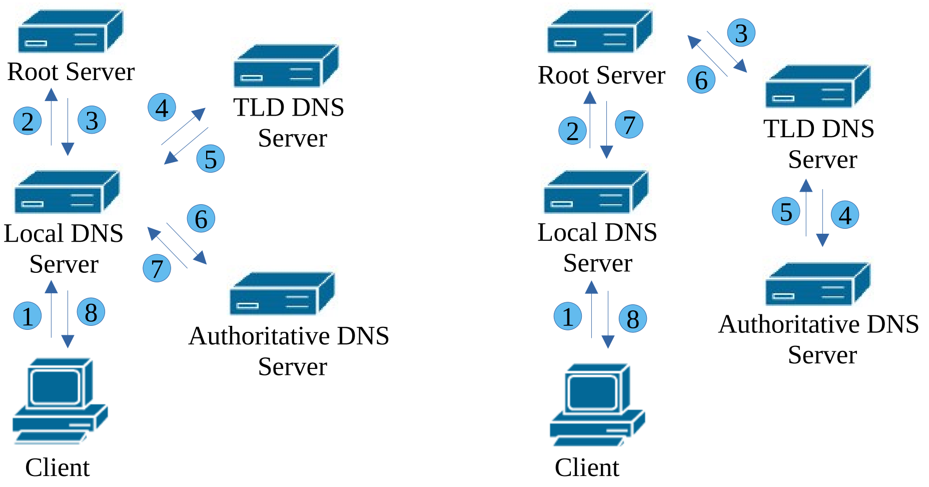


Figure 10 : iterative (left) and recursive (right) DNS queries.

In the lab we are replicating the recursive DNS lookup using a Pi-hole server. For more information on Pi-hole:

https://en.wikipedia.org/wiki/Pi-hole

If you have a old Raspberry Pi at home, or if you are running Docker on a PC, Pi-hole is definitely something to have a look at. To steal a quote from Pi-hole Wiki page :

"*The application acts as a DNS server for a private network (replacing any pre-existing DNS server provided by another device or the ISP), with the ability to block advertisements and tracking domains for users' devices …*"

To connect to the lab's Pi-hole server we will need to connect the Pi system to the lab's network, this is accessed using the **GREEN** network cable, as described in lab 2.

Open a VNC session on Pi-1, then launch a web browser on the Pi-1 by clicking on the icon on the lower toolbar. Within the browser either click on the short cut icon, or enter the Pi-hole's fully qualified domain name or IP address (defined in `/etc/hosts`):

`http://raspberrypi-dns.lan/admin`
`http://192.168.100.3/admin`

This will open the login page as shown in figure 11. Click on the Login button. If prompted the password is: 12345678. Once you have access to the main Pi-hole page you can see a log of what

DNS lookups have been performed and who is using the DNS server.

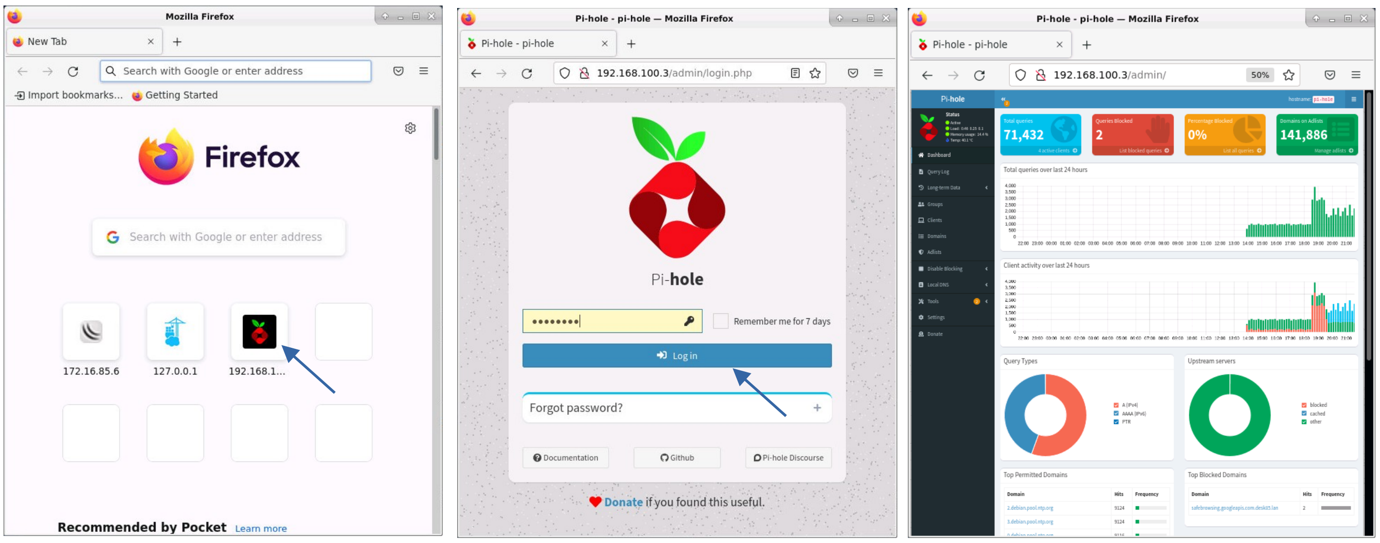
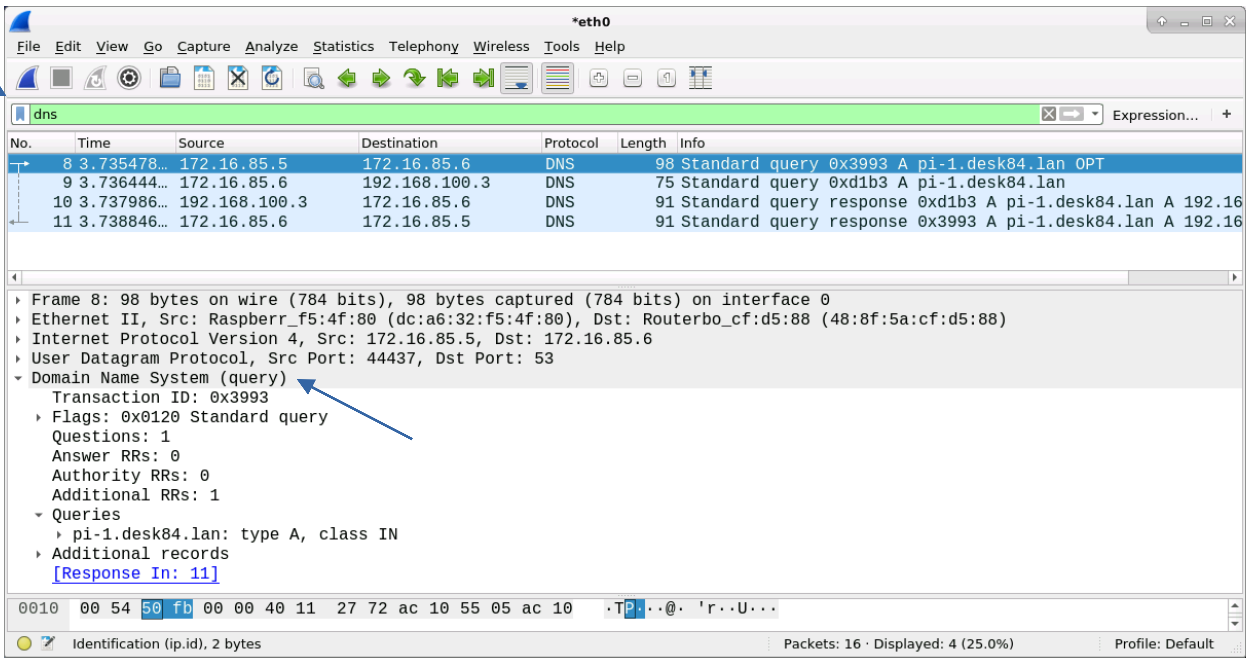


Figure 11 : Pi-hole DNS server


*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
dns
Expression...

```
No.  Time         Source          Destination     Protocol Length Info
  8 3.735478…  172.16.85.5     172.16.85.6     DNS       98 Standard query 0x3993 A pi-1.desk84.lan OPT
  9 3.736444…  172.16.85.6     192.168.100.3   DNS       75 Standard query 0xd1b3 A pi-1.desk84.lan
 10 3.737986…  192.168.100.3   172.16.85.6     DNS       91 Standard query response 0xd1b3 A pi-1.desk84.lan A 192.16
 11 3.738846…  172.16.85.6     172.16.85.5     DNS       91 Standard query response 0x3993 A pi-1.desk84.lan A 192.16
```

```
Frame 8: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
Ethernet II, Src: Raspberr_f5:4f:80 (dc:a6:32:f5:4f:80), Dst: Routerbo_cf:d5:88 (48:8f:5a:cf:d5:88)
Internet Protocol Version 4, Src: 172.16.85.5, Dst: 172.16.85.6
User Datagram Protocol, Src Port: 44437, Dst Port: 53
Domain Name System (query)
  Transaction ID: 0x3993
  Flags: 0x0120 Standard query
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0
  Additional RRs: 1
  Queries
    pi-1.desk84.lan: type A, class IN
  Additional records
  [Response In: 11]
```

```
0010  00 54 50 fb 00 00 40 11  27 72 ac 10 55 05 ac 10   ·TP··@· 'r··U···
```

```
Identification (ip.id), 2 bytes          Packets: 16 · Displayed: 4 (25.0%)          Profile: Default
```

Figure 12 : DNS packet capture

To capture the network packets associated with a DNS lookup, within the VNC session start Wireshark, click on the menu icon and select :

```
Applications -> Internet -> Wireshark
```

This will open the Wireshark GUI. Select the network interface (NIC) to capture packets on, left click on (highlight) the Ethernet 0 (`eth0`) in the Interface list. If needed refer back to lab 2 for more information on how to do this.

In the Wireshark GUI enter the packet protocol filter for DNS : `dns`, as shown in figure 12 and click on the Apply button. To start capturing network packets click on the Start icon on the top toolbar. To generate a recursive DNS lookup enter the following `dig` command:

```
dig pi-1.deskX.lan              (X = 04 to 85, but not your Desk/Box)
```

this will perform a DNS lookup for a different desk. Finally, click on the Stop icon on the top toolbar to stop capturing network traffic.

Task : examine your DNS capture in Wireshark, expand the DNS section, examine the request and response data, do you understand what steps are being performed in this sequence of packets? What sequence of DNS servers have been interrogated to resolve this request? What are the Type, Class and TTL values of this lookup?

For more information on how to setup your own Pi-hole:

http://simplecpudesign.com/networking_name_server/index.html

Task 4

DNS allows a host to translate a domain name into an IP address. However, to be able to do this the host must first have an IP address so that it can connect to the network. As discussed in lab 2 each Raspberry Pi has three Network Interface Controllers (NIC) and each of these can be assigned an IP address. These IP addresses can be static i.e. defined in a configuration file, or assigned using the the Dynamic Host Configuration Protocol (DHCP). This protocol was first proposed in 1985 (RFC 2131: https://datatracker.ietf.org/doc/html/rfc2131), default ports 67/68. DHCP differs from previous protocols as it uses specific ports for server (67) and client (68) as discussed in the lecture. To help illustrate these ideas we will look at the Raspberry Pi system's set-up, shown in figure 13.

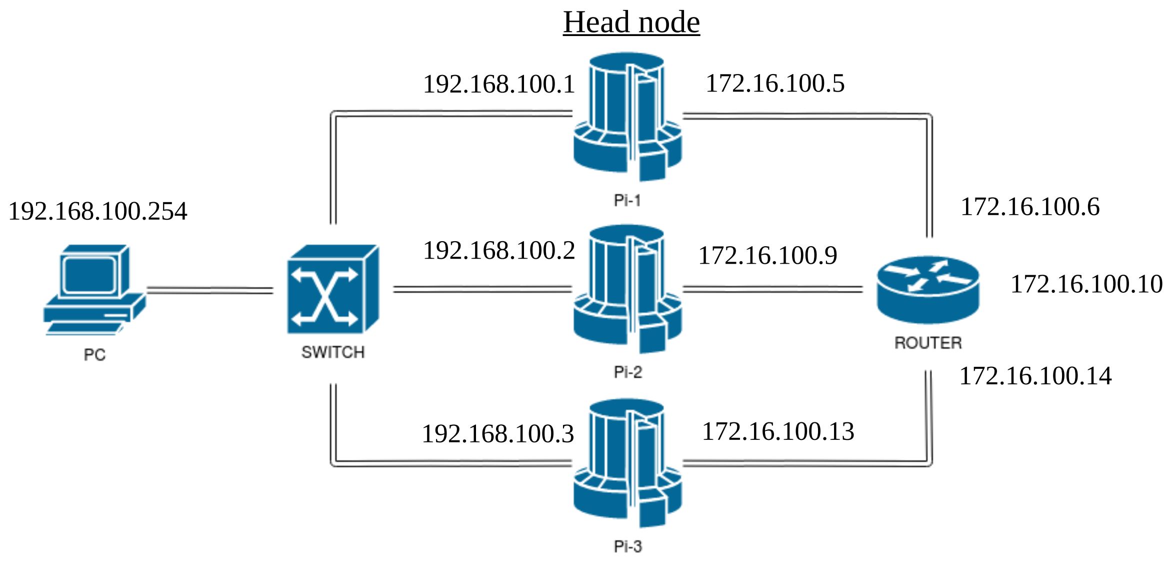


Figure 13 : Example Raspberry Pi system, desk 100

Note, each NIC on a Pi is assigned an IP address. Each network port on the Router is assigned an IP address. The ports on the Switch are not assigned IP addresses. This is because IP addresses work at layer 3 i.e. allowing packets to be routed between these IP addresses. Switches and hubs work on layer 2, which uses the Ethernet protocol, rather than IP. As always we will be looking at this in more detail in a later lab :).

Open a Secure Shell (SSH) on Pi-1 (the head node), as described in lab 1, click on the start button and select the command prompt.

```
-> Command Prompt
```

In the command prompt enter :

```
ssh pi@192.168.X.1                (X=Box/Desk)
```

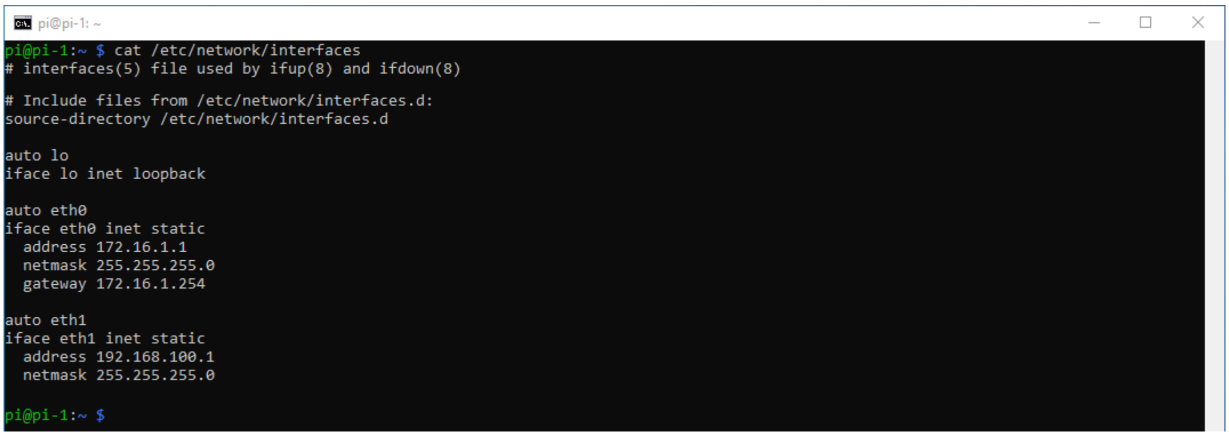
We shall refer to this terminal as **Terminal-1**. In any network some hosts need to have static IP addresses i.e. know their IP address on boot. This allows them to start different network services e.g. a DHCP server. Also, in small networks from a set-up point of view, it can be easier to assigned a NIC a static IP address as it is unlike to change. For Pi-1 this information is stored in the configuration file:

```
/etc/network/interfaces
```

To view this file enter the following command in Terminal-1:

```
cat /etc/network/interfaces
```

This will concatenate (read) this file onto the standard output (display), as shown in figure 14. Your output may contain slight variations.



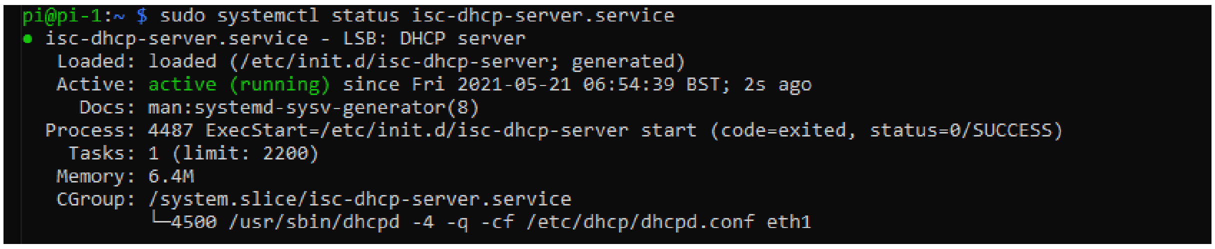
```
pi@pi-1:~ $ cat /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)

# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
  address 172.16.1.1
  netmask 255.255.255.0
  gateway 172.16.1.254

auto eth1
iface eth1 inet static
  address 192.168.100.1
  netmask 255.255.255.0

pi@pi-1:~ $
```

Figure 14 : interfaces configuration file

Next, on the PC, open a second command prompt (**Terminal-2**) and `SSH` into Pi-3 by entering the command :

```
ssh pi@192.168.X.3            (X=Box/Desk)
```

Task : examine the interfaces file for Pi-3 in this terminal. Can you spot any differences between this file and the one on Pi-1? If you would like to check you have the correct file the interfaces file for Pi-3 for my system is shown in Appendix D.



```
pi@pi-1:~ $ sudo systemctl status isc-dhcp-server.service
● isc-dhcp-server.service - LSB: DHCP server
   Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
   Active: active (running) since Fri 2021-05-21 06:54:39 BST; 2s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 4487 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, status=0/SUCCESS)
    Tasks: 1 (limit: 2200)
   Memory: 6.4M
   CGroup: /system.slice/isc-dhcp-server.service
           └─4500 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf eth1
```

Figure 15 : DHCP server status

Pi-1 is the head node in the Pi network and is configured as a DHCP server for the network address range `192.168.X.2` to `192.168.X.253` i.e. used on `eth1`. The network interface connected to the switch. Therefore, on Pi-1 `eth1` is set up with a static IP address so that system can start the DHCP service, or daemon. A background process controlled by the OS that allocates IP addresses to other hosts. To see the status of this service on Pi-1, in Terminal-1 enter the following command:

```
sudo systemctl status isc-dhcp-server.service
```

The response should look like that shown in figure 15, the key thing to check is that the service is active (running).

Note, if something has gone wrong and this or other services are not running you can try and restart them using the `systemctl` command. To restart DHCP service enter the following command:

```
sudo systemctl restart isc-dhcp-server.service
```

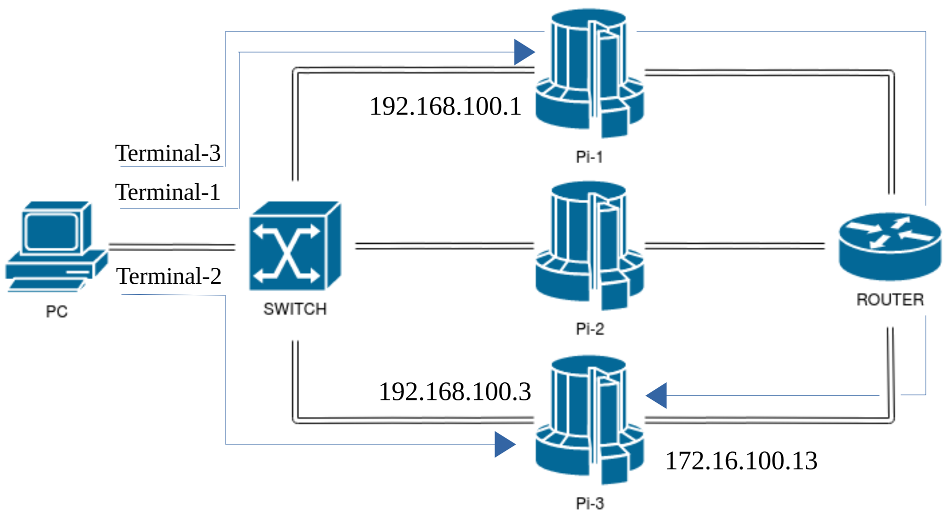


Figure 16 : Pi-1 and Pi-3 network addresses.

As discussed in the lecture the DHCP protocol is performed using a series of handshakes between the server (Pi-1) and the client (Pi-2 or Pi-3), which can be viewed / captured using Wireshark.

To see the DHCP protocol in action we will need to open a terminal across a different network (172.16.X.13), so that we can remotely take down and bring up the network interface connected to the DHCP server i.e. 192.168.X.3, as shown in figure 16.

Note, as we are running each Pi headless i.e. without a display or keyboard, we can't use an `SSH` session across the 192 network, as once it is taken down you will be disconnected.

To connect to Pi-3's 172 network interface open a third command prompt and open another `SSH` connections to Pi-1 by entering the command :

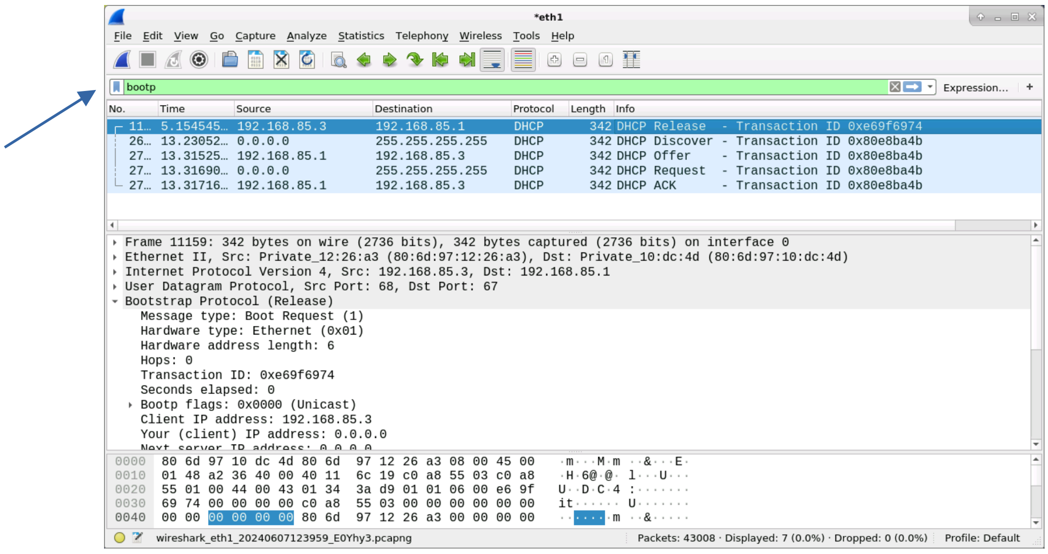
```
ssh pi@192.168.X.1        (X=Box/Desk)
```

Then within this new terminal `SSH` into Pi-3 across the 172.16.X.13 network by enter the following command:

```
ssh pi@172.16.X.13        (X=Box/Desk)
```

We shall refer to this terminal as **Terminal-3**.

Note, you can open as many terminals as you like on each Pi, each terminal is a process running in parallel with the other services controlled / managed by the OS. There will be a limit on the number of terminals that a Pi can have running at the same time, but i'm not sure what will run out / fall over first i.e. memory, ports or OS :).


*eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
bootp
Expression...

```
No.   Time        Source          Destination      Protocol  Length Info
11…   5.154545…   192.168.85.3    192.168.85.1     DHCP      342 DHCP Release  - Transaction ID 0xe69f6974
26…   13.23052…   0.0.0.0         255.255.255.255  DHCP      342 DHCP Discover - Transaction ID 0x80e8ba4b
27…   13.31525…   192.168.85.1    192.168.85.3     DHCP      342 DHCP Offer    - Transaction ID 0x80e8ba4b
27…   13.31690…   0.0.0.0         255.255.255.255  DHCP      342 DHCP Request  - Transaction ID 0x80e8ba4b
27…   13.31716…   192.168.85.1    192.168.85.3     DHCP      342 DHCP ACK      - Transaction ID 0x80e8ba4b

Frame 11159: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0
Ethernet II, Src: Private_12:26:a3 (80:6d:97:12:26:a3), Dst: Private_10:dc:4d (80:6d:97:10:dc:4d)
Internet Protocol Version 4, Src: 192.168.85.3, Dst: 192.168.85.1
User Datagram Protocol, Src Port: 68, Dst Port: 67
Bootstrap Protocol (Release)
  Message type: Boot Request (1)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0xe69f6974
  Seconds elapsed: 0
  Bootp flags: 0x0000 (Unicast)
  Client IP address: 192.168.85.3
  Your (client) IP address: 0.0.0.0
  Next server IP address: 0.0.0.0

0000  80 6d 97 10 dc 4d 80 6d  97 12 26 a3 08 00 45 00   ·m···M·m ··&···E·
0010  01 48 a2 36 40 00 40 11  6c 19 c0 a8 55 03 c0 a8   ·H·6@·@· l···U···
0020  55 01 00 44 00 43 01 34  3a d9 01 01 06 00 e6 9f   U··D·C·4 :·······
0030  69 74 00 00 00 00 c0 a8  55 03 00 00 00 00 00 00   it······ U·······
0040  00 00 00 00 00 00 80 6d  97 12 26 a3 00 00 00 00   ·······m ··&·····

wireshark_eth1_20240607123959_E0Yhy3.pcapng   Packets: 43008 · Displayed: 7 (0.0%) · Dropped: 0 (0.0%)   Profile: Default
```

Figure 17 : DHCP packets

To capture the network packets associated with a DHCP request, open a VNC session on Pi-1 and start Wireshark, click on the menu icon and select :

Applications `-> Internet -> Wireshark`

This will open the Wireshark GUI. To select the network interface (NIC) to capture packets on, left click on (highlight) the Ethernet 1 (`eth1`) icon in the Interface list. In the Wireshark GUI enter the packet protocol filter for DHCP : `bootp`, as shown in figure 17 and click on the Apply button. This protocol filter name refers back to the original bootstrap protocol that has been effectively taken over by DHCP. For more info on this older protocol refer to:

https://en.wikipedia.org/wiki/Bootstrap_Protocol

Note, you can also filter packets on the port numbers used i.e. for DHCP this is 67/68, by enter the filter expression :

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

To start capturing network packets click on the Start icon on the top toolbar. You will now see some DHCP activity in the main window as Wireshark captures packets associated with managing the various DHCP connections. However, to trigger a new DHCP request we need to force Pi-3 to release its connection. In Terminal-3 i.e. the terminal connected to Pi-3 using the 172.16.X.13 network interface, type in the following command:

```
sudo dhclient -r -v
```

This will force Pi-3 to release the IP address 192.168.X.3 associated with its NIC `eth1`. You can see this if you now enter the command below on Pi-3:

```
ifconfig
```

The entry associated with `eth1` is no longer assigned an IP address. Also, if you try and type something in Terminal-2 nothing will happen, eventually it will time-out and close. To force Pi-3 to request a "new" IP address for `eth1` enter the following command in Terminal-3:

```
sudo dhclient -v
```

Finally, click on the Stop icon on the top toolbar to stop capturing network traffic. Within the DHCP packets captured you will find the RELEASE, DISCOVER, OFFER, REQUEST and ACK packets, as shown in figure 17.

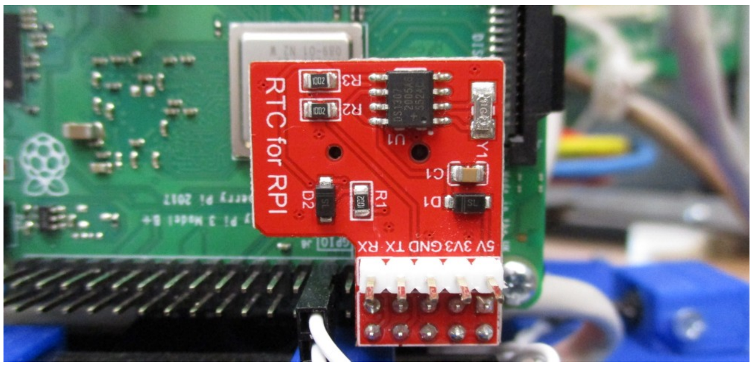


Figure 18 : Real Time Clock (RTC)

Note, the server will normal suggest the previously assigned IP address, therefore, the “new” IP address may not be that new.

Task : examine your DHCP capture in Wireshark, expand the Bootstrap section, examine the request and response data, do you understand what steps are being performed in this sequence of packets?

Task 5

The Network Time Protocol (NTP) (RFC 1305: https://datatracker.ietf.org/doc/html/rfc1305) is another early protocol created in 1985 and again uses a client server model, default port 123. This protocol allows a computer to adjust its local internal clock by querying an online reference clock e.g. a very accurate atomic clock. In your Raspberry Pi system this functionality is simulated on the head node (Pi-1) by using a Real Time Clock (RTC) module, as shown in figure 18. This local time is then used to synchronise the clocks on the clients (Pi-2 and Pi-3).

Note, typically there would be a number of different reference clocks. These clocks are organised in a hierarchical structure i.e. the stratum level 0 – 16, as discussed in the lecture. More information on the real time clock module (RTC) used is available here :

http://www.simplecpudesign.com/networking_base_node/index.html

To read the real time clocks value, in Terminal-1 (Pi-1) enter the following command:

```
sudo hwclock -r
```

This will return the current time stored in the DS1307 RTC. This RTC has a battery backup so even if the Pi is powered down the clock keeps on ticking. To see the time used by the OS enter this command:

```
date
```

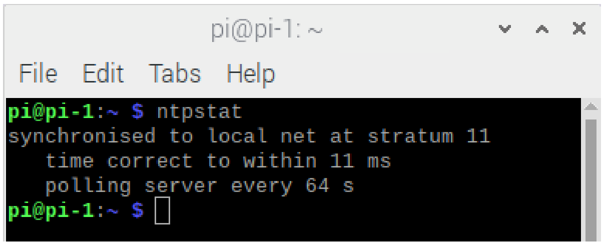
This will return the local date and time. This should match the RTC, plus the time it took you to type the command. It may also differ owing to day light saving settings. The NTP server runs as a service on Pi-1. To check it is running correctly enter the following command.

```
sudo systemctl status ntp.service
```

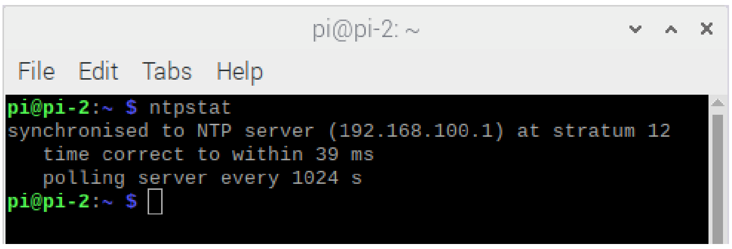
If all is good you should see the normal “active (running)” message. To get a little more extra detail run the following command in Terminal-1 (Pi-1) and Terminal-3 (Pi-2):

```
ntpstat
```

This will return details of the stratum level (quality of source clock), accuracy (of local clock given network delays) and polling (update) rates as shown in figure 19. A key thing to remember about NTP is that it is **not** designed to update a system’s clock, it is designed to synchronise a system’s clock to one or more reference clocks. The way I think about the NTP service is that it adjusts the frequency of the clock signal driving the OS’s clock e.g. if the local time is ahead of the reference time it will lower this frequency, if the local time is behind the reference time it will increase this frequency. The reason why you do not simply update the system time to the “correct” time is that there may be system events scheduled at specific times. Therefore, if you jumped to a new time you could “miss” an event i.e. jumping over these times.



```
pi@pi-1:~ $ ntpstat
synchronised to local net at stratum 11
   time correct to within 11 ms
   polling server every 64 s
pi@pi-1:~ $
```



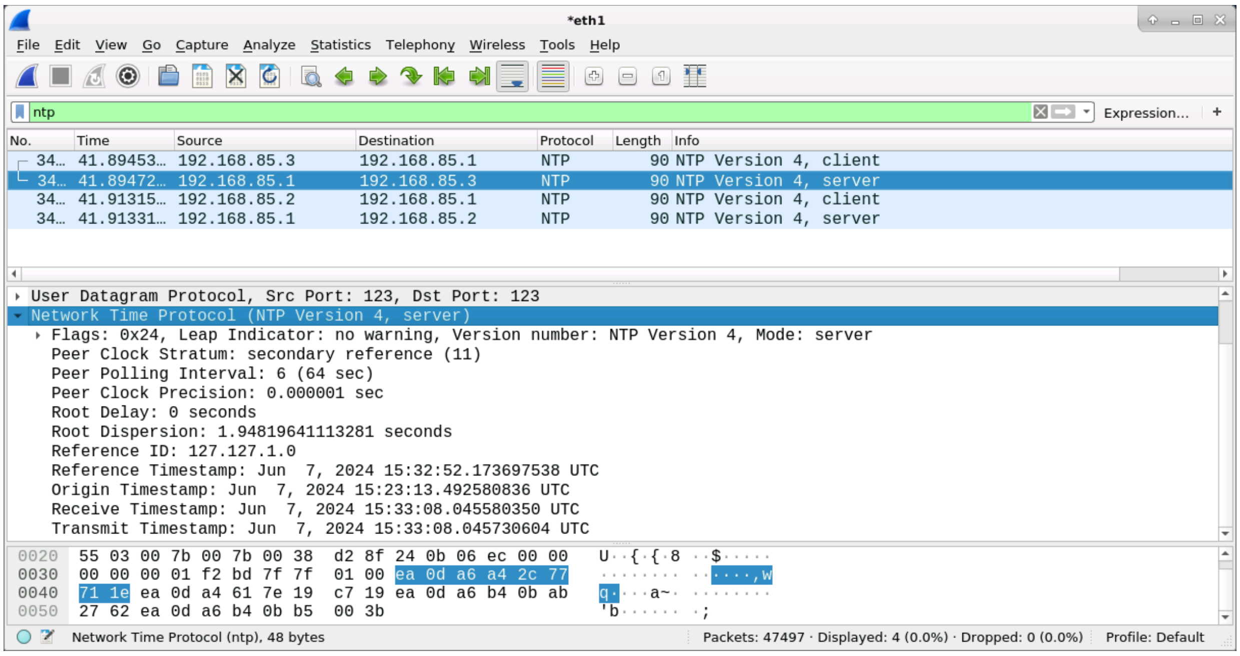
```
pi@pi-2:~ $ ntpstat
synchronised to NTP server (192.168.100.1) at stratum 12
   time correct to within 39 ms
   polling server every 1024 s
pi@pi-2:~ $
```

Figure 19 : NTPstat, server (left), client (right)

To capture the network packets associated with an NTP request, within a VNC session start Wireshark, click on the menu icon and select :

Applications `-> Internet -> Wireshark`

This will open the Wireshark GUI. To select the network interface (NIC) to capture packets on, left click on (highlight) the Ethernet 1 (`eth1`) icon in the Interface list. Enter the packet protocol filter for NTP : `ntp`, as shown in figure 20 and click on the Apply button.


*eth1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ntp
Expression...

```
No.   Time        Source         Destination    Protocol  Length Info
 34…  41.89453…   192.168.85.3   192.168.85.1   NTP           90 NTP Version 4, client
 34…  41.89472…   192.168.85.1   192.168.85.3   NTP           90 NTP Version 4, server
 34…  41.91315…   192.168.85.2   192.168.85.1   NTP           90 NTP Version 4, client
 34…  41.91331…   192.168.85.1   192.168.85.2   NTP           90 NTP Version 4, server

User Datagram Protocol, Src Port: 123, Dst Port: 123
Network Time Protocol (NTP Version 4, server)
  Flags: 0x24, Leap Indicator: no warning, Version number: NTP Version 4, Mode: server
  Peer Clock Stratum: secondary reference (11)
  Peer Polling Interval: 6 (64 sec)
  Peer Clock Precision: 0.000001 sec
  Root Delay: 0 seconds
  Root Dispersion: 1.94819641113281 seconds
  Reference ID: 127.127.1.0
  Reference Timestamp: Jun  7, 2024 15:32:52.173697538 UTC
  Origin Timestamp: Jun  7, 2024 15:23:13.492580836 UTC
  Receive Timestamp: Jun  7, 2024 15:33:08.045580350 UTC
  Transmit Timestamp: Jun  7, 2024 15:33:08.045730604 UTC

0020  55 03 00 7b 00 7b 00 38  d2 8f 24 0b 06 ec 00 00   U··{·{·8 ··$·····
0030  00 00 00 01 f2 bd 7f 7f  01 00 ea 0d a6 a4 2c 77   ········ ····,w
0040  71 1e ea 0d a4 61 7e 19  c7 19 ea 0d a6 b4 0b ab   q····a~· ········
0050  27 62 ea 0d a6 b4 0b b5  00 3b                     'b······ ·;

Network Time Protocol (ntp), 48 bytes     Packets: 47497 · Displayed: 4 (0.0%) · Dropped: 0 (0.0%)     Profile: Default
```

Figure 20 : NTP packets

To start capturing network packets click on the Start icon on the top toolbar. You may have to wait a minute for the client (Pi-2 / Pi-3) to request a time update from the NTP server (Pi-1). When you see a client / server handshake click on the Stop icon on the top toolbar to stop capturing network traffic. As the Raspberry Pi system has been running for a reasonable period of time the clocks on Pi-2 and Pi-3 should be reasonably synchronised.

Note, to force the NTP client to request an update you can restart the NTP service on Pi-2 / Pi-3 using the following command:

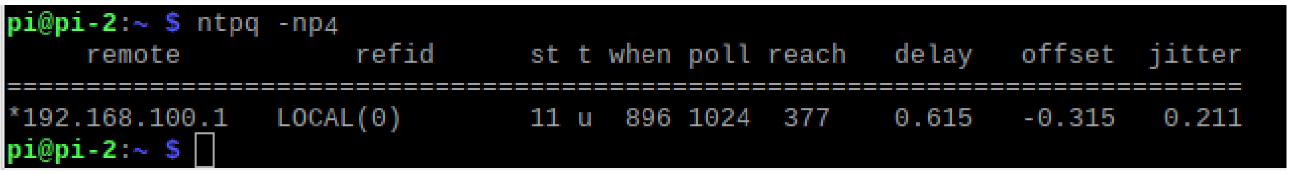
```
sudo systemctl restart ntp.service
```

Time within a computer can be represented using a number of different formats. The Pi OS uses Unix epoch time i.e. the number of seconds that have elapsed since the “birth” of Unix on January 1, 1970 midnight (UTC). This time is not quite the “birth” of Unix, it was a date chosen for simplicity at Bell labs, so not quite its birthday. At the time of writing this script the Unix epoch time is : 1717762029. To display this “time” on a Pi, in a terminal enter the command :

```
date +%s
```

The NTP protocol uses as similar approach, its packet contain a fixed point value counting the number of seconds from its epoch time, the number of seconds since 00:00:00 on January 1, 1900 (UTC).

Task : examine your NTP packet capture in Wireshark. Can you identify the binary data used to represent each timestamp?



```
pi@pi-2:~ $ ntpq -np4
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*192.168.100.1   LOCAL(0)        11 u  896 1024  377    0.615   -0.315   0.211
pi@pi-2:~ $ 
```

Figure 21 : NTPQ table

Another useful command to look at is the NTP query program: `ntpq`, allowing you to view your system’s time statistics. In Terminal-3 enter the following command:

```
ntpq -np4
```

This will return a table of information as shown in figure 21.

A quick summary of these columns:

- Remote : NTP servers IP (as always will vary with desk).
- Refid : NTP servers that the NTP uses, in this case as its the RTC this is LOCAL.
- St : NTP server’s stratum level.
- T : type i.e. unicast, broadcast, multicast, or manycast.
- When : last time the server was polled in seconds.
- Poll : how often the server will be polled.
- Reach : status of the last eight NTP updates.
- Delay : round trip network delay (ms)
- Offset : mean offset between local and server times (ms)
- Jitter : mean deviation between local and server times (ms)

The lab’s raspberry-ntp time server is equipped with GPS module, making it a stratum 1 NTP server. The GPS module used is a GT-U7, shown in figure 22, allowing the Pi to access the atomic clocks on one or more satellites. For more information on this system refer here:

http://www.simplecpudesign.com/networking_time_node/index.html

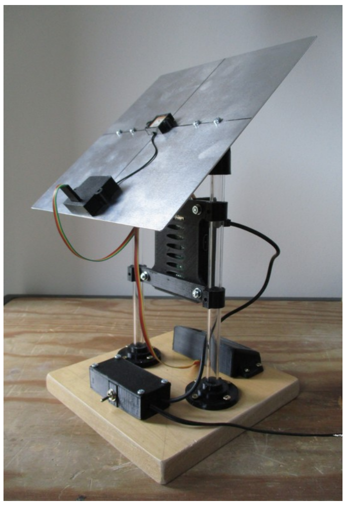
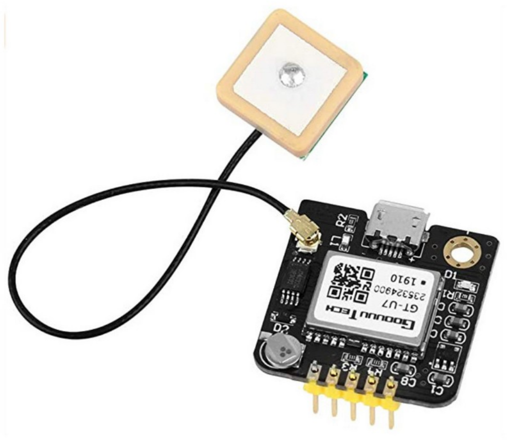



Figure 22 : GPS server (left), GPS module (right)

Optional Task : configure Pi-1 to use the lab's NTP time server. To do this you will need to first stop the NTP service on Pi-1 using the `systemctl` command. Update the NTP service configuration file: `/etc/ntp.conf`. Restart the NTP service on Pi-1 and check its working correctly using the `systemctl` commands. Then use the program: `ntpq`, to test the "correctness" of the time on Pi-1. Sooooo, to set Pi-1s RTC to GPS time:

STEP 1 : update `/etc/ntp.conf`

```
server 192.168.100.1 prefer
server 127.127.1.0
fudge 127.127.1.0 stratum 10
```

STEP 2 : stop the NTP service

```
sudo systemctl stop ntp.service
```

STEP 3 : sync to NTP server. Note, this may take a couple of minutes

```
sudo ntpd -gq
```

STEP 4 : restart the NTP service

```
sudo systemctl start ntp.service
```

STEP 5 : check time

```
ntpq -np4
```

STEP 6 : write date / time to the RTC using the command:

```
sudo hwclock -w
```

You can also restart the NTP services on Pi-2 and Pi-3. SSH into each pi and run the following commands:

STEP 7 : stop the ntp service

```
sudo systemctl stop ntp.service
```

STEP 8 : sync to ntp server. Note, this may take a couple of minutes

```
sudo ntpd -gq
```

STEP 9 : restart the ntp service

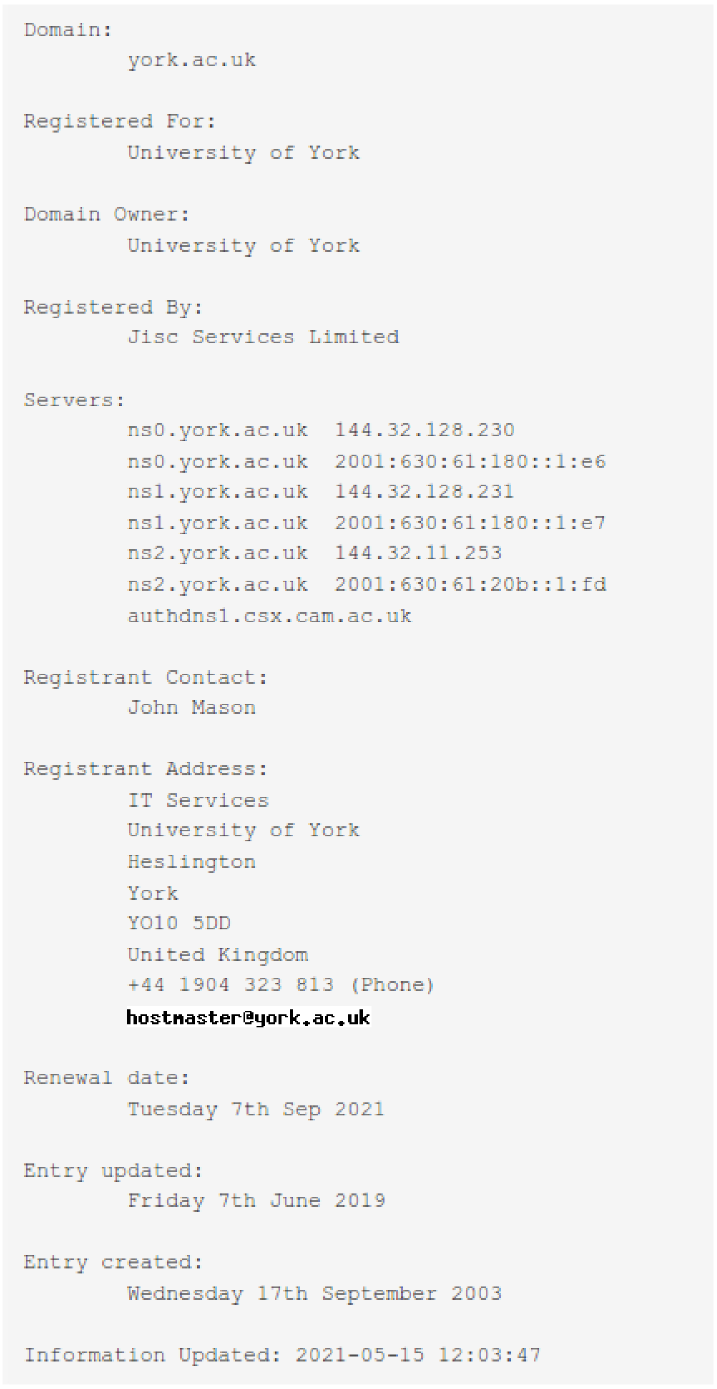
```
sudo systemctl start ntp.service
```

STEP 10 : check time, you should see that the time source is now listed as a stratum 2.

```
ntpq -np4
```

Note, if GPS receiver can not receive a signal and looses synchronisation it will cause the lab’s NTP server to be listed as a stratum 16 time server i.e. do not trust. Confess the UV filters on the department’s windows do block / attenuate the GPS signal, so this can occur, but for some reason, I’m guessing better line of site to the satellite it seems to work better in the afternoons.

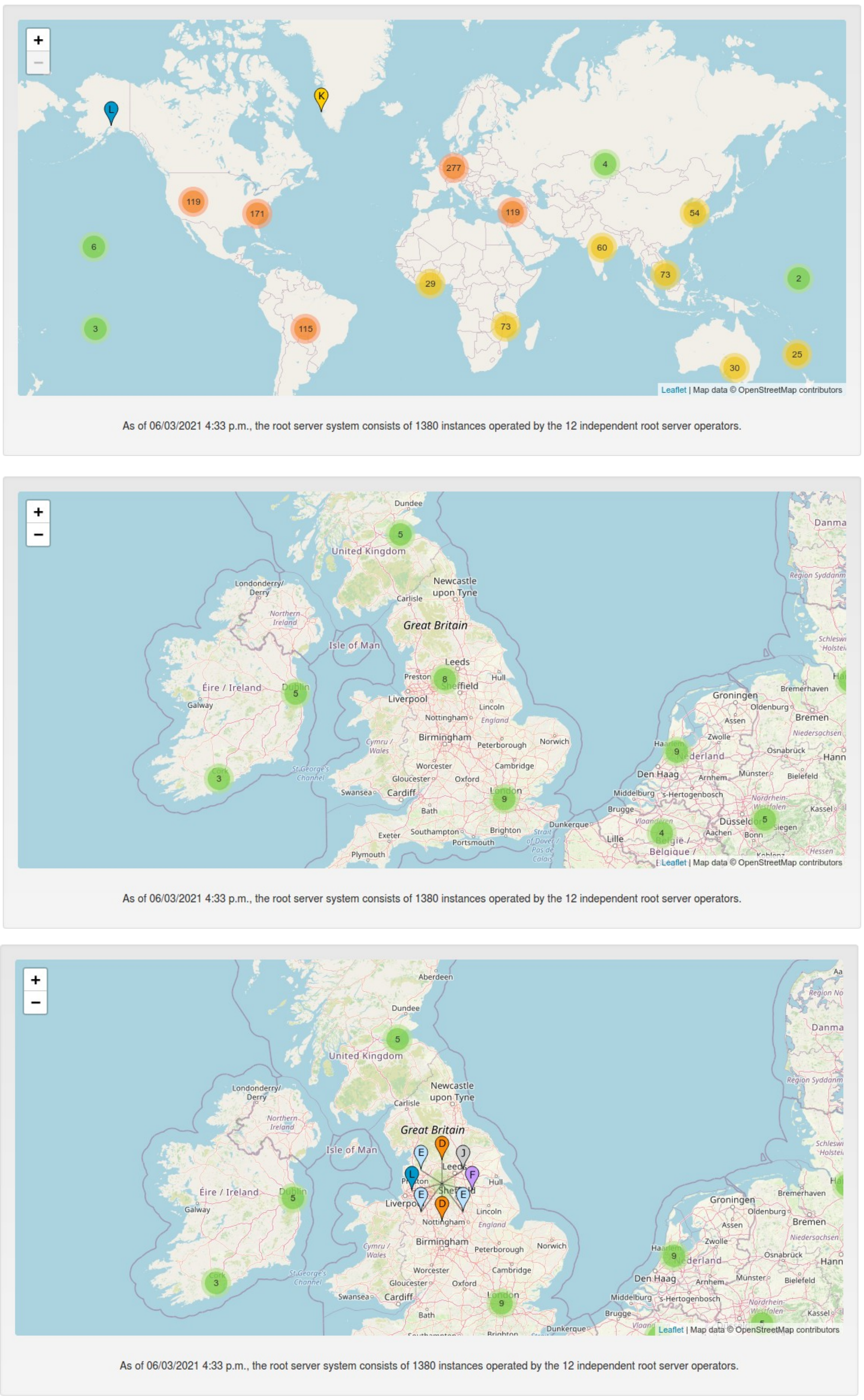
Appendix A : WHOIS



```
Domain:
        york.ac.uk

Registered For:
        University of York

Domain Owner:
        University of York

Registered By:
        Jisc Services Limited

Servers:
        ns0.york.ac.uk  144.32.128.230
        ns0.york.ac.uk  2001:630:61:180::1:e6
        ns1.york.ac.uk  144.32.128.231
        ns1.york.ac.uk  2001:630:61:180::1:e7
        ns2.york.ac.uk  144.32.11.253
        ns2.york.ac.uk  2001:630:61:20b::1:fd
        authdns1.csx.cam.ac.uk

Registrant Contact:
        John Mason

Registrant Address:
        IT Services
        University of York
        Heslington
        York
        YO10 5DD
        United Kingdom
        +44 1904 323 813 (Phone)
        hostmaster@york.ac.uk

Renewal date:
        Tuesday 7th Sep 2021

Entry updated:
        Friday 7th June 2019

Entry created:
        Wednesday 17th September 2003

Information Updated: 2021-05-15 12:03:47
```

Figure A1 : whois for york.ac.uk

Appendix B : Root DNS server locations



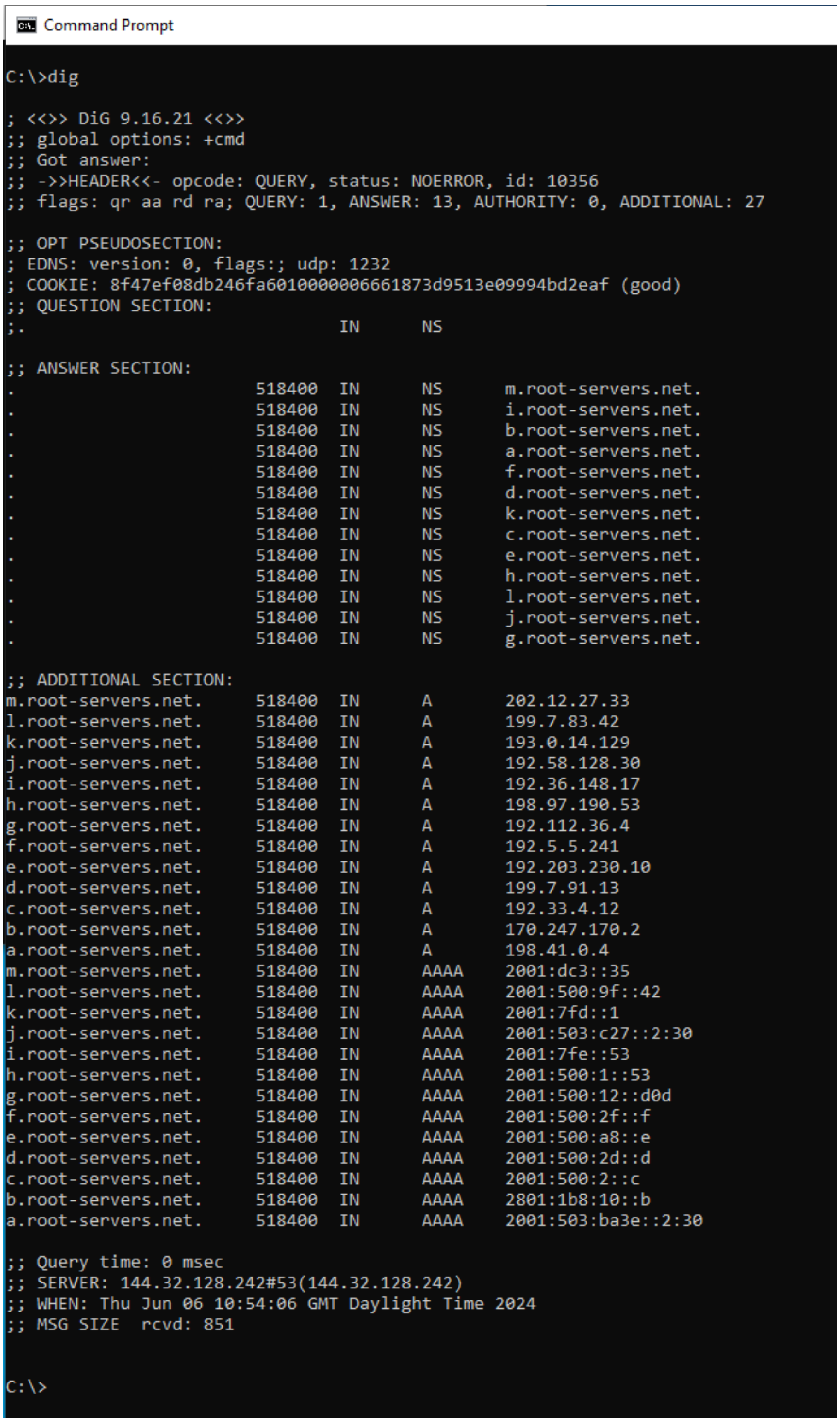
As of 06/03/2021 4:33 p.m., the root server system consists of 1380 instances operated by the 12 independent root server operators.

As of 06/03/2021 4:33 p.m., the root server system consists of 1380 instances operated by the 12 independent root server operators.

As of 06/03/2021 4:33 p.m., the root server system consists of 1380 instances operated by the 12 independent root server operators.

Figure B1 : root DNS server locations

Appendix C : Root DNS servers IP addresses



```
C:\>dig

; <<>> DiG 9.16.21 <<>>
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10356
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 8f47ef08db246fa6010000006661873d9513e09994bd2eaf (good)
;; QUESTION SECTION:
;.                              IN      NS

;; ANSWER SECTION:
.                       518400  IN      NS      m.root-servers.net.
.                       518400  IN      NS      i.root-servers.net.
.                       518400  IN      NS      b.root-servers.net.
.                       518400  IN      NS      a.root-servers.net.
.                       518400  IN      NS      f.root-servers.net.
.                       518400  IN      NS      d.root-servers.net.
.                       518400  IN      NS      k.root-servers.net.
.                       518400  IN      NS      c.root-servers.net.
.                       518400  IN      NS      e.root-servers.net.
.                       518400  IN      NS      h.root-servers.net.
.                       518400  IN      NS      l.root-servers.net.
.                       518400  IN      NS      j.root-servers.net.
.                       518400  IN      NS      g.root-servers.net.

;; ADDITIONAL SECTION:
m.root-servers.net.     518400  IN      A       202.12.27.33
l.root-servers.net.     518400  IN      A       199.7.83.42
k.root-servers.net.     518400  IN      A       193.0.14.129
j.root-servers.net.     518400  IN      A       192.58.128.30
i.root-servers.net.     518400  IN      A       192.36.148.17
h.root-servers.net.     518400  IN      A       198.97.190.53
g.root-servers.net.     518400  IN      A       192.112.36.4
f.root-servers.net.     518400  IN      A       192.5.5.241
e.root-servers.net.     518400  IN      A       192.203.230.10
d.root-servers.net.     518400  IN      A       199.7.91.13
c.root-servers.net.     518400  IN      A       192.33.4.12
b.root-servers.net.     518400  IN      A       170.247.170.2
a.root-servers.net.     518400  IN      A       198.41.0.4
m.root-servers.net.     518400  IN      AAAA    2001:dc3::35
l.root-servers.net.     518400  IN      AAAA    2001:500:9f::42
k.root-servers.net.     518400  IN      AAAA    2001:7fd::1
j.root-servers.net.     518400  IN      AAAA    2001:503:c27::2:30
i.root-servers.net.     518400  IN      AAAA    2001:7fe::53
h.root-servers.net.     518400  IN      AAAA    2001:500:1::53
g.root-servers.net.     518400  IN      AAAA    2001:500:12::d0d
f.root-servers.net.     518400  IN      AAAA    2001:500:2f::f
e.root-servers.net.     518400  IN      AAAA    2001:500:a8::e
d.root-servers.net.     518400  IN      AAAA    2001:500:2d::d
c.root-servers.net.     518400  IN      AAAA    2001:500:2::c
b.root-servers.net.     518400  IN      AAAA    2801:1b8:10::b
a.root-servers.net.     518400  IN      AAAA    2001:503:ba3e::2:30

;; Query time: 0 msec
;; SERVER: 144.32.128.242#53(144.32.128.242)
;; WHEN: Thu Jun 06 10:54:06 GMT Daylight Time 2024
;; MSG SIZE  rcvd: 851


C:\>
```

Figure C1 : root DNS server IP addresses

Appendix D : Pi-3 /etc/interfaces

```
# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:
#source-directory /etc/network/interfaces.d

# NOTE : replace X with your desk number

auto lo
iface lo inet loopback

auto lo:1
iface lo:1 inet static
address 10.X.3.1
netmask 255.255.255.0

auto eth0
iface eth0 inet static
address 172.16.X.13
netmask 255.255.255.252
gateway 172.16.X.14

auto eth1
iface eth1 inet dhcp
```