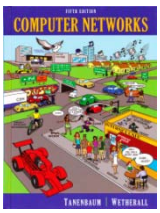
Lab Exercise – DNS



Objective

DNS (Domain Name System) is the system and protocol that translates domain names to IP addresses and more. DNS is covered in §7.1 of your text. Review that section before doing this lab.

Requirements

Wireshark: This lab uses Wireshark to capture or examine a packet trace. A packet trace is a record of traffic at some location on the network, as if a snapshot was taken of all the bits that passed across a particular wire. The packet trace records a timestamp for each packet, along with the bits that make up the packet, from the low-layer headers to the higher-layer contents. Wireshark runs on most operating systems, including Windows, Mac and Linux. It provides a graphical UI that shows the sequence of packets and the meaning of the bits when interpreted as protocol headers and data. The packets are color-coded to convey their meaning, and Wireshark includes various ways to filter and analyze them to let you investigate different aspects of behavior. It is widely used to troubleshoot networks. You can download Wireshark from www.wireshark.org if it is not already installed on your computer. We highly recommend that you watch the short, 5 minute video "Introduction to Wireshark" that is on the site.

Browser: This lab uses a web browser to find or fetch pages as a workload. Any web browser will do.

dig: This lab uses `dig` to issue DNS request and observe DNS responses. `dig` is a flexible, command-line tool for querying remote DNS servers that replaces the older nslookup program. It comes installed on Mac OS. On Window, you can download `dig` from ISC's BIND web site as part of the bind download. (Note that there may be some dependencies. Check for online instructions to set up `dig` on Windows.) On Linux, install `dig` with your package manager. It is normally part of a dnsutils or bindutils package.

Network Setup

In a typical network, your computer contacts a local DNS nameserver to resolve domain names to IP addresses. The local nameserver may be another computer in your company network, a computer at your ISP, or your wireless AP. It exchanges a series of messages with remote DNS nameservers all over the Internet to perform the resolution. The setup is as shown in the figure below.

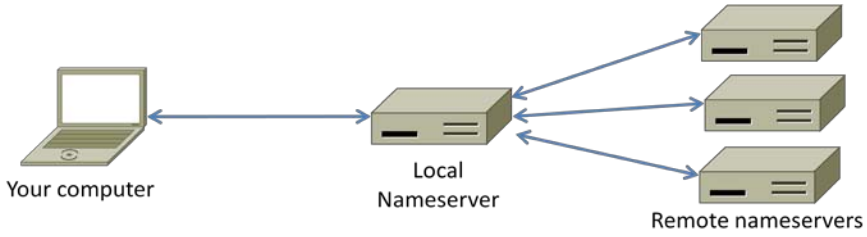


Figure 1: Typical network setup for DNS

We assume this setup for the lab, and it has an important implication: the trace we gather at our computer will see the exchanges between our computer and the local nameserver, but not between the local nameserver and the remote nameservers.

Step 1: Manual Name Resolution

Before we look at how your computer uses the DNS, we will see how a local nameserver resolves a DNS name, i.e., we will interact with remote nameservers. To do this exercise, you will pretend to be the local nameserver and issue requests to remote nameservers using the `dig` tool.

Pick a domain name to resolve, such as that of your web server. We will use `www.uwa.edu.au`. *Find the IP address of one of the root nameservers by searching the web.* For example, the Wikipedia article on root name servers includes the IP address of the root nameservers `a` through `m`. Any one of these should do, as they hold replicated information. You need this information to begin the name resolution process, and nameservers are provided with it as part of their configuration.

Use `dig` *to issue a request to a root nameserver to perform the first step of the resolution*. You are assuming that you have no cached information that will let you begin a resolution below the root. The format of a `dig` command is "`dig @aa.bb.cc.dd domainname`". It instructs `dig` to send a request to a nameserver at a given IP address (or name) for the given domain name. In the figure below, we used `dig` to send a request to the "`a`" root nameserver whose IP address is 198.41.0.4 to resolve our example web server, i.e., "`dig @198.41.0.4 www.uwa.edu.au`". The reply from the root does not provide the full name resolution, but it does tell us about nameservers closer to having the information for you to contact. In this case, it is nameservers who know about the ".`au`" domain. Multiple nameservers are given as alternative choices, and the reply helpfully includes their IP addresses; we can see IPv6 addresses as well as IPv4 addresses.

Continue the resolution process with `dig` *until you complete the resolution. When you have alternatives to choose, prefer IPv4 nameservers and select the first one in alphabetical order. If this nameserver has multiple IP addresses then select the numerically smallest IP address.* In the figure, the nameserver at IP address 58.65.254.73 that is authoritative for "au." is the remote nameserver to contact next. You can complete the resolution without these tie-breaking rules and will likely obtain the same result since the DNS information is replicated. The rules are so that everyone doing the lab follows the same path. Keep these `dig` commands handy, as you will repeat them in the next step when you capture a trace.

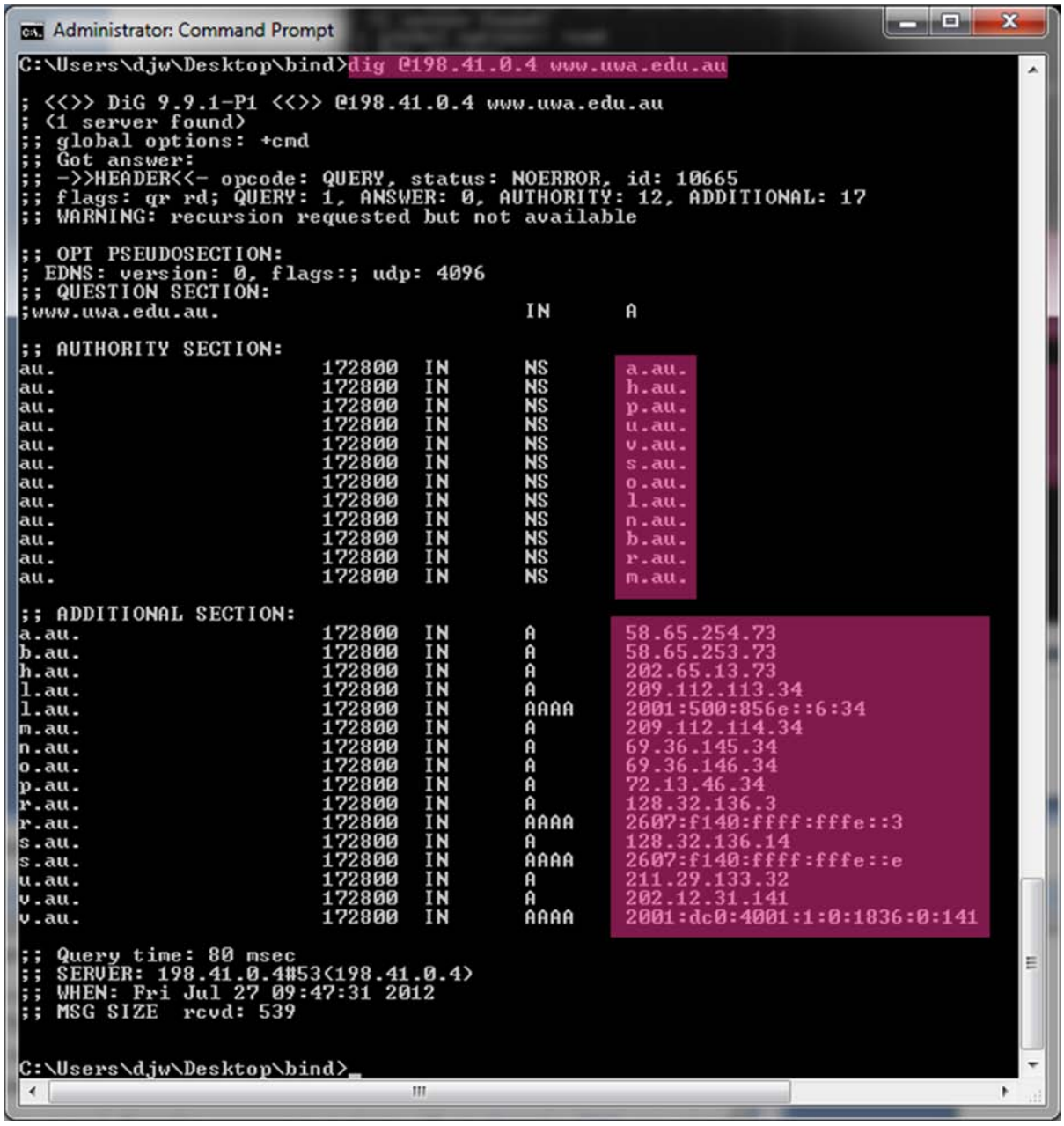


Figure 2: Using `dig` to query a nameserver

Draw a figure that shows the sequence of remote nameservers that you contacted and the domain for which they are responsible. Note that future name resolutions are likely to be a much shorter sequence because they can use cached information. For example, if you looked up a domain name in "`.edu`" then when you look up a different domain name in "`.edu`" you already know the name of the "`.edu`" nameserver. Thus you can start there, or even closer to the final nameserver depending on what you have cached; you do not need to start again at the root nameserver.

Turn-in: Hand in your drawing.

Step 2: Capture a Trace

Capture a trace of your browser making DNS requests as follows; alternatively, you may use a supplied trace. Now that we are familiar with the process of name resolution, we will inspect the details of DNS traffic. To generate DNS traffic you will both repeat the `dig` commands, and browse web sites.

1. *Close all unnecessary browser tabs and windows.* Browsing web sites will generate DNS traffic as your browser resolves domain names to connect to remote servers. We want to minimize browser activity initially so that we capture only the intended DNS traffic.
2. *Launch Wireshark and start a capture with a filter of* "`udp port 53`". We use this filter because there is no shorthand for DNS, but DNS is normally carried on UDP port 53. Your capture window should be similar to the one pictured below, other than our highlighting. Select the interface from which to capture as the main wired or wireless interface used by your computer to connect to the Internet. If unsure, guess and revisit this step later if your capture is not successful. Uncheck "capture packets in promiscuous mode". This mode is useful to overhear packets sent to/from other computers on broadcast networks. We only want to record packets sent to/from your computer. Leave other options at their default values. The capture filter, if present, is used to prevent the capture of other traffic your computer may send or receive. On Wireshark 1.8, the capture filter box is present directly on the options screen, but on Wireshark 1.9, you set a capture filter by double-clicking on the interface.
3. *Repeat the* `dig` *commands from the previous step.* This time, you should see the DNS request and reply packets that correspond to your commands captured in the trace window. Note that there may be some background DNS traffic originating from your computer if any process needs to resolve names to make a network connection. We are assuming that there will be little of this traffic so that you can
4. *Wait 10 seconds, then open your browser and browse a variety of sites.* Using your browser will generate DNS traffic as you visit new domains, and also as your browser runs its background tasks such as auto-completion. Unlike the `dig` traffic, this will be DNS traffic between your computer and the local nameserver.
5. *Stop the capture when you have a good sample of DNS traffic.* We would like enough traffic to see a variety of behavior. DNS traffic is generated fairly quickly as you browse so it should only take a short while to collect this DNS traffic.

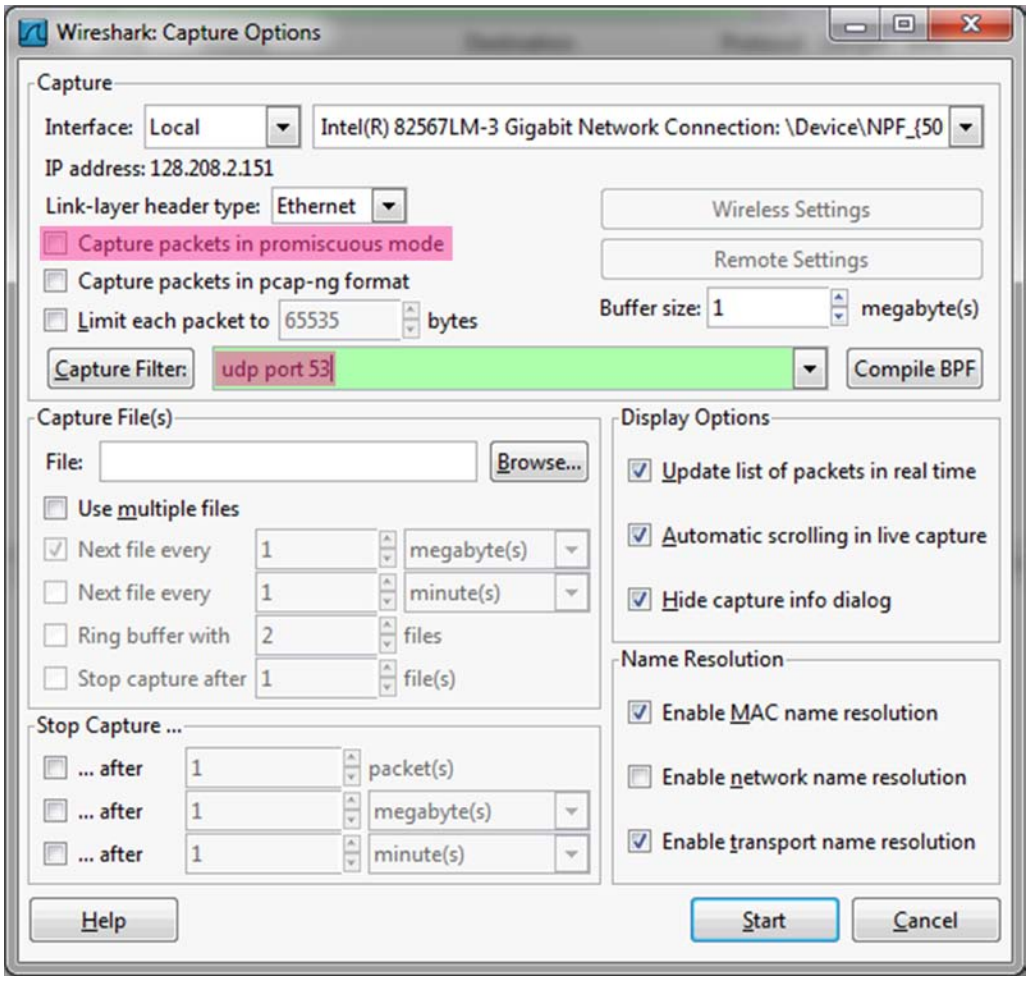


Figure 3: Setting up the capture options

Step 3: Inspect the Trace

To explore the details of DNS packets, select a DNS query expand its Domain Name System block (by using the "+" expander or icon). Your display should be similar to the one shown in our figure, with a series of packets with protocol DNS. The first packets should correspond to your `dig` commands, followed by DNS traffic produced by your browser. We have selected the first DNS message.

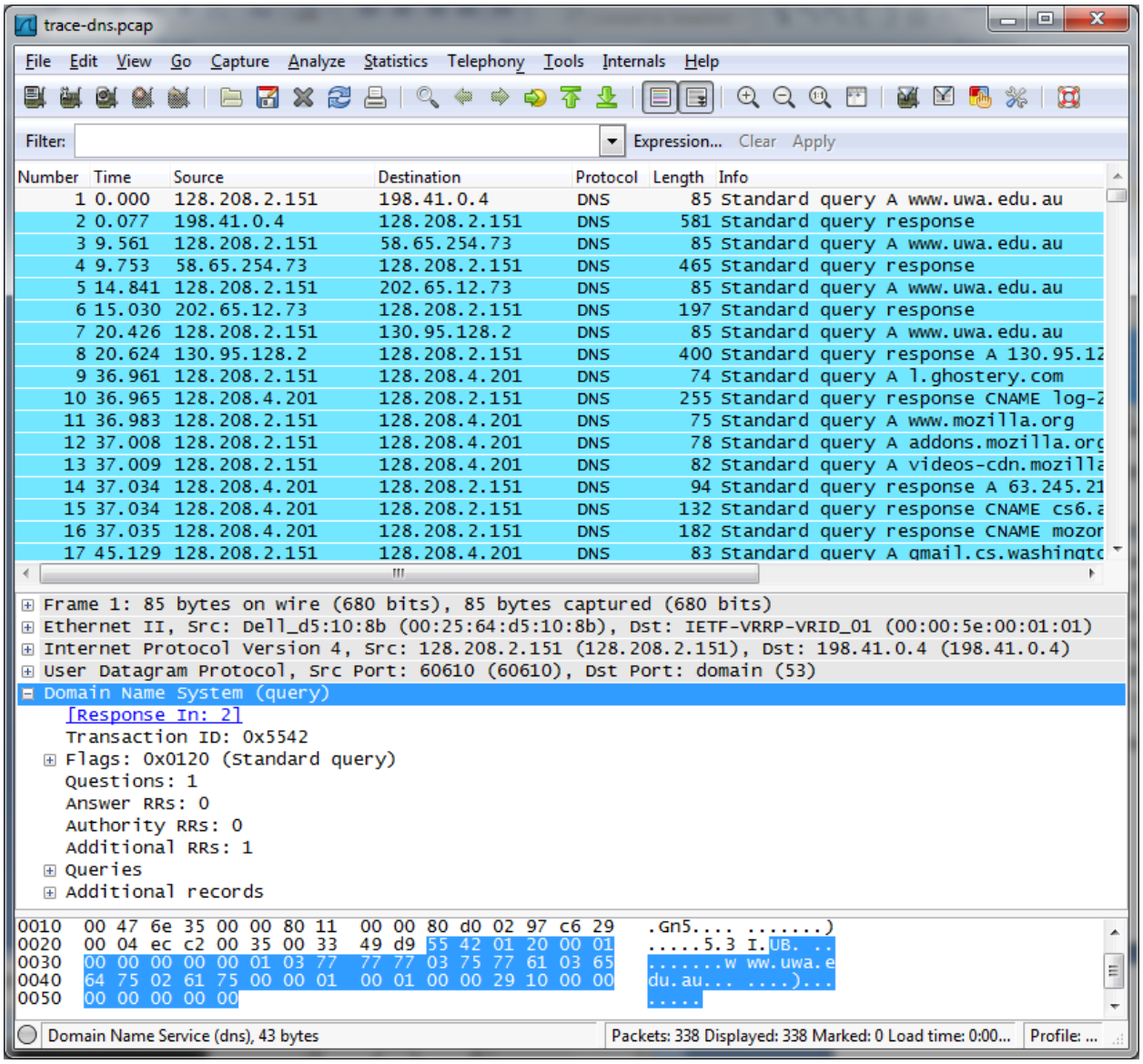


Figure 3: Trace of DNS traffic showing the details of the DNS header

Look for the following details:

- The DNS block follows the IP and UDP blocks. This is because DNS messages are carried in UDP segments within IP packets. You will see that the UDP port used by a nameserver is 53.
- The DNS header starts with a Transaction ID that is used to link a request and the corresponding reply – they both carry the same Transaction ID.
- Next come a set of flags that you can expand. They indicate whether the DNS message is a query or response, amongst other details.
- Then come the number of query, answer, authority and additional records. These fields conclude the header.
- After the DNS header, the remainder of the message consists of the indicated number of query, answer, authority and additional records. Often there will be only one query – for the IP address

of the domain name we are seeking – but there may be many of the other records. These records are grouped in sections, such as the Authority section for all of the authority records. Each query has a Type code that indicates the kind of record sought, whether an IP address or otherwise. Each of the other records also has a Type code that indicates whether it carries an IP address of a host, the name of a nameserver, or something else. The format of an individual record depends on its type. The entire DNS message is designed to fit within one UDP message.

- Wireshark may show other information, such as the number of the packet that carries the response to this request or the response time for the DNS exchange, but this is derived information. It is not actually carried on any packet.

Repeat the above to look at a DNS response. You should see a larger set of records in this message; while DNS queries mostly serve to carry the query, DNS responses often return a set of useful information.

Step 4: Details of DNS Messages

Select the first DNS query that corresponds to your `dig` *commands and expand its DNS block.* Likely this query is the first packet in your trace, with the first several packets corresponding to your `dig` commands, followed by other DNS traffic produced by your browser. To check, see if there are several queries that list the domain you chose in the Info column, each followed by a response. We will use these DNS messages to study the details of the DNS protocol. Sometimes there may be other DNS traffic interspersed with these queries due to background activity; you should ignore these extraneous packets.

Look at the DNS header, and answer the following questions:

1. *How many bits long is the Transaction ID? Based on this length, take your best guess as to how likely it is that concurrent transactions will use the same transaction ID.*
2. *Which flag bit and what values signifies whether the DNS message is a query or response?*
3. *How many bytes long is the entire DNS header?* Use information in the bottom status line when you select parts of the packet and the bottom panel to help you work this out.

Now examine the responses to the `dig` *DNS queries you made.* The initial response should have provided another nameserver one step closer to the nameserver, but not the final answer. You should find that it includes the original query in its Query section. It will also include records with both the name of the nameservers to contact next, and the IP addresses of those nameservers. The final response in this series will include the IP address of the domain name – this is the answer to the query.

Look at the body of the DNS response messages, and answer the following questions:

4. *For the initial response, in what section are the names of the nameservers carried? What is the Type of the records that carry nameserver names?*
5. *Similarly, in what section are the IP addresses of the nameservers carried, and what is the Type of the records that carry the IP addresses?*
6. *For the final response, in what section is the IP address of the domain name carried?*

Turn-in: Hand in your answers to the above questions.

Step 5: DNS Response Time

To conclude this lab, we will look at the DNS response time of the DNS queries made by your browser. This is your normal DNS usage, in which your computer sends a single query and receives the answer in the response. The response time is the delay between when your computer sends the query to the local nameserver and when it receives the response from the local nameserver. This time includes the time taken by the local nameserver to contact remote nameservers, if the answer is not cached. Since this response time can delay connections to sites, it should be as small as possible.

Proceed as follows to generate an "IO Graph" of the DNS response times. IO graphs are a standard feature of Wireshark available under the Statistics menu. By default, this graph shows the rate of packets over time. We will tweak it to show the DNS response time over the trace with the following changes:

- *On the x-axis, adjust the tick interval and pixels per tick for viewing.* The tick interval should be small enough to see into the behavior over the trace. One second is probably a good choice for your trace. The pixels per tick can be adjusted to make the graph wider or narrower to fill the window; you can also adjust the width of the window.
- *On the y-axis, change the unit to be "Advanced".* The default is Packet/Tick. "Advanced" is a special keyword that will let us access different data values to graph. Once you select it, a new "Calc:" box will appear to let us specify the data values.
- *Enter "*`dns.time`*" into the calculation box and set the pull-down menu to be "MAX(*)".* `dns.time` is a virtual field calculated by Wireshark from the query and response messages. It is shown with DNS responses, and gives the DNS response time. Choosing "MAX(*)" will let us see the largest DNS response time in every tick interval so that we can spot outliers. "AVG(*)" would also be a reasonable choice.
- *Press Enter, and click the "Graph" button if necessary*. You may need to do this to trigger a redisplay. You should now have a graph of response times similar to our graph in the figure below.

We expect that you will see many small DNS response times, and a scattering of larger DNS response times. In our graph, most times are very small, likely because the correct answer is cached by the local nameserver. In some cases, however, there is a longer delay of hundreds of milliseconds as remote nameservers must be queried. You can click a point on the graph to be taken to the nearest point in the trace if there is a feature you would like to investigate.

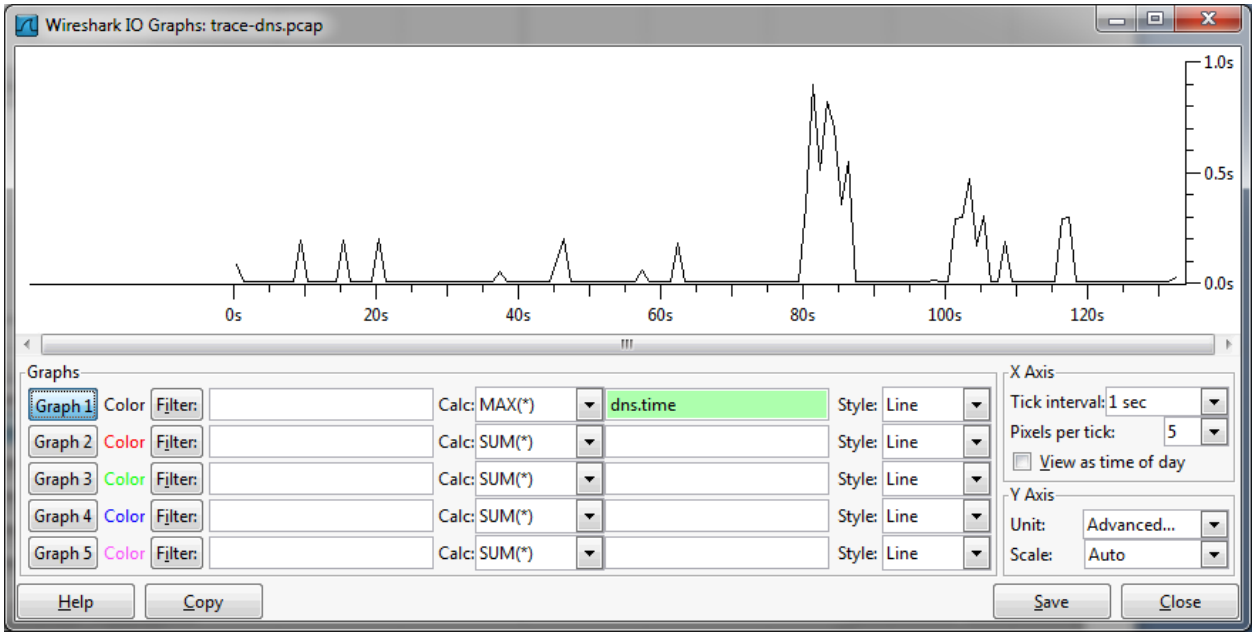


Figure 4: DNS response time via an IO graph

If you look over the DNS traffic caused by your browser, you are likely to see a greater range of behaviors than in the DNS traffic caused by the `dig` commands. This behavior might include new types of records, such as CNAME (canonical name, to provide information about aliases when one machine is known by multiple names), answers that indicate that a name does not exist, and so forth.

That's all, there are no questions in this step.

Explore Your Network

We encourage you to explore DNS on your own once you have completed this lab. Some ideas:

- Look up other types of DNS records, such as MX to find the mail server for a domain, and AAAA to find the IPv6 address of a domain.
- Google provides an alternate DNS nameserver system that you may use called "Google Public DNS". Look it up, and follow the configuration instructions to test it out. Experiment to see if this DNS service is faster than your existing DNS arrangement.
- Reverse DNS lookups determine the domain name associated with an IP address. They are often used as a security check. Read about and perform some reverse DNS lookups.
- DNSSEC is a set of security extensions for DNS. It uses additional DNS record types to return key and signature information so that nameservers can check the authenticity of responses. Read about DNSSEC and perform some DNSSEC lookups using dig. You will need to add "+dnssec" to turn on the flag requesting security.

[END]