web fundermentals try hack me

What happens when you make a DNS request?

1. When you request a domain name, your computer first checks its local cache to see if you've previously looked up the address recently; if not, a request to your Recursive DNS Server will be made.

2. A Recursive DNS Server is usually provided by your ISP, but you can also choose your own. This server also has a local cache of recently looked up domain names. If a result is found locally, this is sent back to your computer, and your request ends here. If the request cannot be found locally, a journey begins to find the correct answer, starting with the internet's root DNS servers.

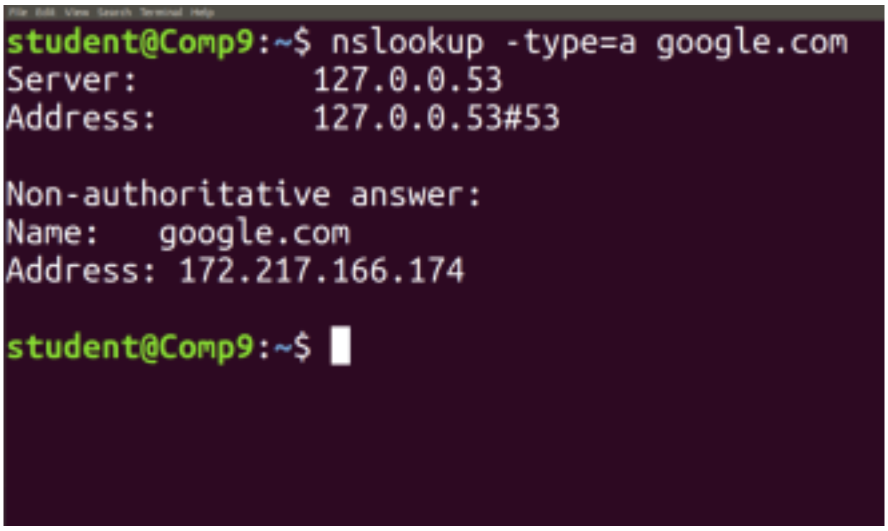
3. The root servers act as the DNS backbone of the internet; their job is to redirect you to the correct Top Level Domain Server, depending on your request. If, for example, you request www.tryhackme.com, the root server will recognise the Top Level Domain of .com and refer you to the correct TLD server that deals with .com addresses.

4. The TLD server holds records for where to find the authoritative server to answer the DNS request. The authoritative server is often also known as the nameserver for the domain. For example, the name server for tryhackme.com is kip.ns.cloudflare.com and uma.ns.cloudflare.com. You'll often find multiple nameservers for a domain name to act as a backup in case one goes down.

5. An authoritative DNS server is the server that is responsible for storing the DNS records for a particular domain name and where any updates to your domain name DNS records would be made. Depending on the record type, the DNS record is then sent back to the Recursive DNS Server, where a local copy will be cached for future requests and then relayed back to the original client that made the request. DNS records all come with a TTL (Time To Live) value. This value is a number represented in seconds that the response should be saved for locally until you have to look it up again. Caching saves on having to make a DNS request every time you communicate with a server.

nslookup command: -queries internet domain name servers

Syntax: **nslookup** [- option] [name | -

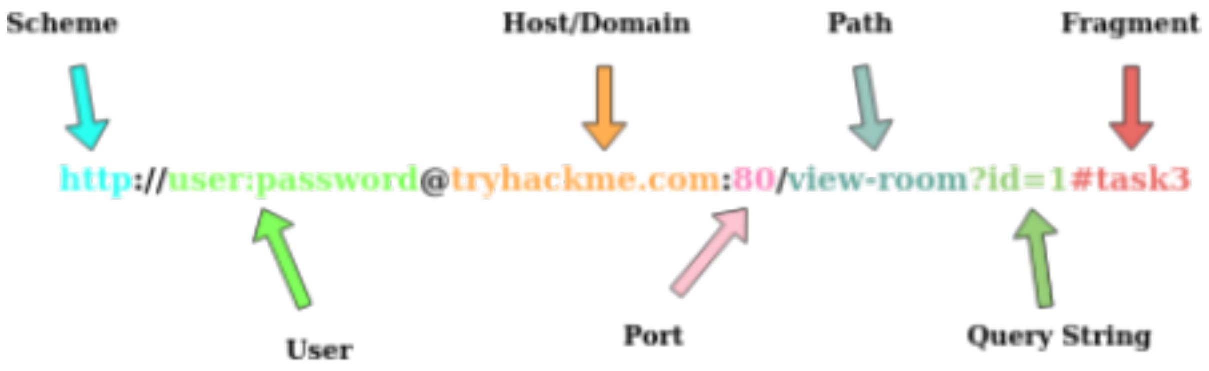


```
student@Comp9:~$ nslookup -type=a google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 172.217.166.174

student@Comp9:~$
```

URL: stands for Uniform Resource Locator



Scheme: This instructs on what protocol to use for accessing the resource such as HTTP, HTTPS, FTP (File Transfer Protocol).

User: Some services require authentication to log in, you can put a username and password into the URL to log in.

Host: The domain name or IP address of the server you wish to access.

Port: The Port that you are going to connect to, usually 80 for HTTP and 443 for HTTPS, but this can be hosted on any port between 1 - 65535.

Path: The file name or location of the resource you are trying to access.

Query String: Extra bits of information that can be sent to the requested path. For example, /blog?**id=1** would tell the blog path that you wish to receive the blog article with the id of 1.

Fragment: This is a reference to a location on the actual page requested. This is commonly used for pages with long content and can have a certain part of the page directly linked to it, so it is viewable to the user as soon as they access the page.

HTTP Methods:

- **GET Request**

This is used for getting information from a web server.

- **POST Request**

This is used for submitting data to the web server and potentially creating new records

- **PUT Request**

This is used for submitting data to a web server to update information

- **DELETE Request**

This is used for deleting information/records from a web server.

COMMON HTTP STATUS Codes

- HTTP Status Code 200 - OK.
- HTTP Status Code 301 - Permanent Redirect.
- HTTP Status Code 302 - Temporary Redirect.
- HTTP Status Code 404 - Not Found.
- HTTP Status Code 410 - Gone.
- HTTP Status Code 500 - Internal Server Error.
- HTTP Status Code 503 - Service Unavailable.

Common Request Headers

These are headers that are sent from the client (usually your browser) to the server.

Host: Some web servers host multiple websites so by providing the host headers you can tell it which one you require, otherwise you'll just receive the default website for the server.

User-Agent: This is your browser software and version number, telling the web server your browser software helps it format the website properly for your browser and also some elements of HTML, JavaScript and CSS are only available in certain browsers.

Content-Length: When sending data to a web server such as in a form, the content length tells the web server how much data to expect in the web request. This way the server can ensure it isn't missing any data.
Accept-Encoding: Tells the web server what types of compression methods the browser supports so the data can be made smaller for transmitting over the internet.

Cookie: Data sent to the server to help remember your information (see cookies task for more information).

Common Response Headers

These are the headers that are returned to the client from the server after a request.

Set-Cookie: Information to store which gets sent back to the web server on each request (see cookies task for more information).

Cache-Control: How long to store the content of the response in the browser's cache before it requests it again.

Content-Type: This tells the client what type of data is being returned, i.e., HTML, CSS, JavaScript, Images, PDF, Video, etc. Using the content-type header the browser then knows how to process the data.

Content-Encoding: What method has been used to compress the data to make it smaller when sending it over the internet.