

DNS

Deep dive

Hosts file

- The hosts file has been around since the beginnings of DNS. And just a little background here on the hosts file, when the internet only consisted of MIT, military, and a handful of other organizations, it was an easy way to have name to IP address resolution. But as the internet grew, having a hosts file that has every single website in it, and its corresponding IP address? Not really feasible.
- The computer file hosts is an operating system file that maps hostnames to IP addresses.
- It is a plain text file. It is a common part of an OS's Internet Protocol (IP) implementation, and serves the function of translating human-friendly hostnames into numeric protocol addresses, called IP addresses, that identify and locate a host in an IP network.
- The hosts file is one of several system facilities that assists in addressing network nodes in a computer network.
- Host name to IP address resolution on Windows is a combination of the contents of the hosts file and the records returned from the specified DNS server(s) in response to DNS queries, **with the hosts file being queried first.**
- **DNS tries hard to answer DNS resolutions to IP addresses locally before it ever goes to a DNS server.**
- Unlike remote DNS resolvers, the hosts file is under the direct control of the local computer's administrator.

DNS Query

- Step 1, DNS looks to itself locally (**hosts file**) on every single machine.
- Step 2, after the hosts file is read, then **DNS cache** is looked into.
- DNS tries hard to answer DNS resolutions to IP addresses locally before it ever goes to a DNS server.
- Step 3, Last effort is DNS server service.

Fire up a Windows VM

- Download a Windows 10 Enterprise evaluation copy
 - Select the ISO – Enterprise
 - Note it is 4GB in size!
 - You may need to give this a try later
-
- <https://www.microsoft.com/en-us/evalcenter/evaluate-windows-10-enterprise>

WARNING!

- Note use a VM to mitigate making changes to your host operating system!
- Editing the hosts file is different based on the operating system. So if you're on a Windows operating system, and that could be Windows 7, 8, 10, or any of the server OS's, what you'll want to do is open an elevated Command Prompt.
- You can also edit the hosts file in Mac OS and Linux...however be careful!

Finding the hosts file...

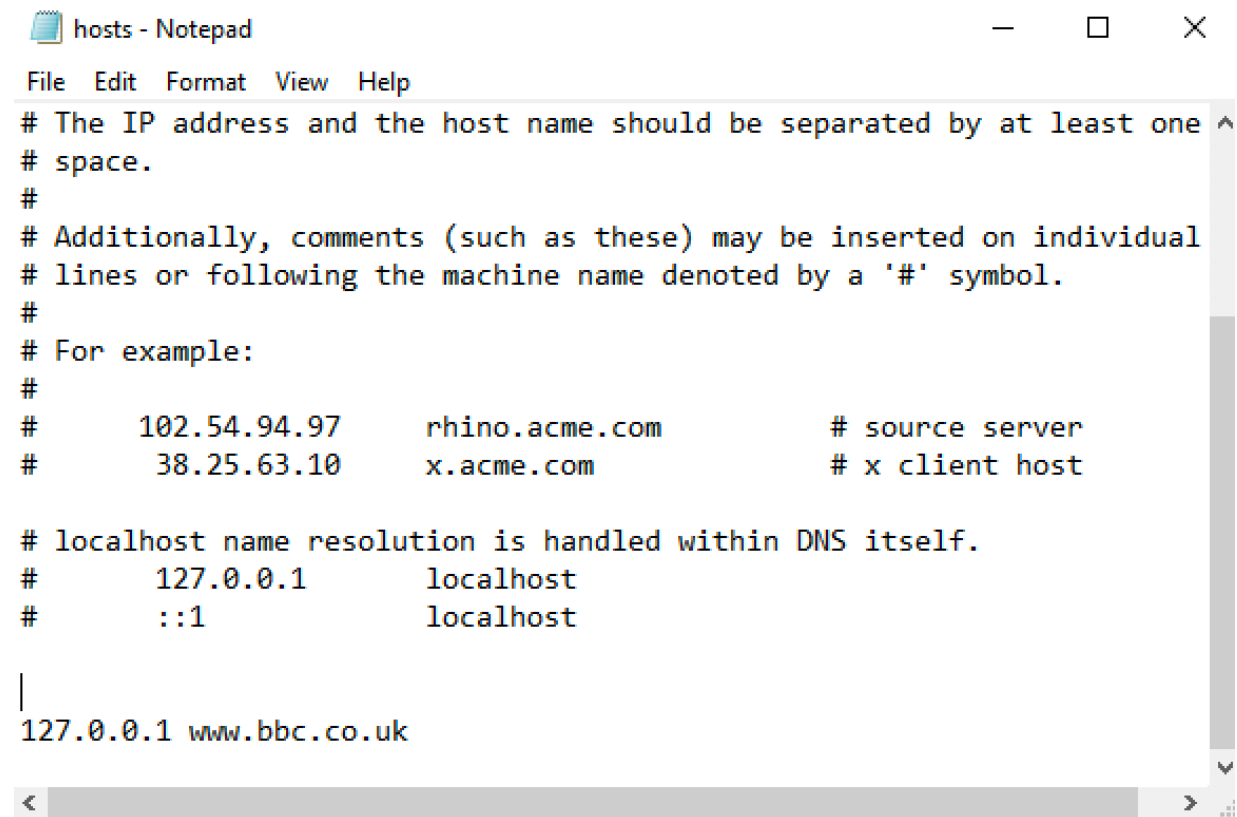
- **For Windows 10 and 8**
- Press the Windows key.
- Type Notepad in the search field.
- In the search results, right-click Notepad and select Run as administrator.
- From Notepad, open the following file:
c:\Windows\System32\Drivers\etc\hosts.
 - Note if you don't see any files in the drivers directory select the down arrow beside Text Documents (*.txt) and select All files.
- Make the necessary changes to the file.
- Click File > Save to save your changes.

Security issues

- The hosts file may present an attack vector for malicious software.
- The file may be modified, for example, by adware, computer viruses, or trojan horse software to redirect traffic from the intended destination to sites hosting malicious or unwanted content.

Making a change to the host file

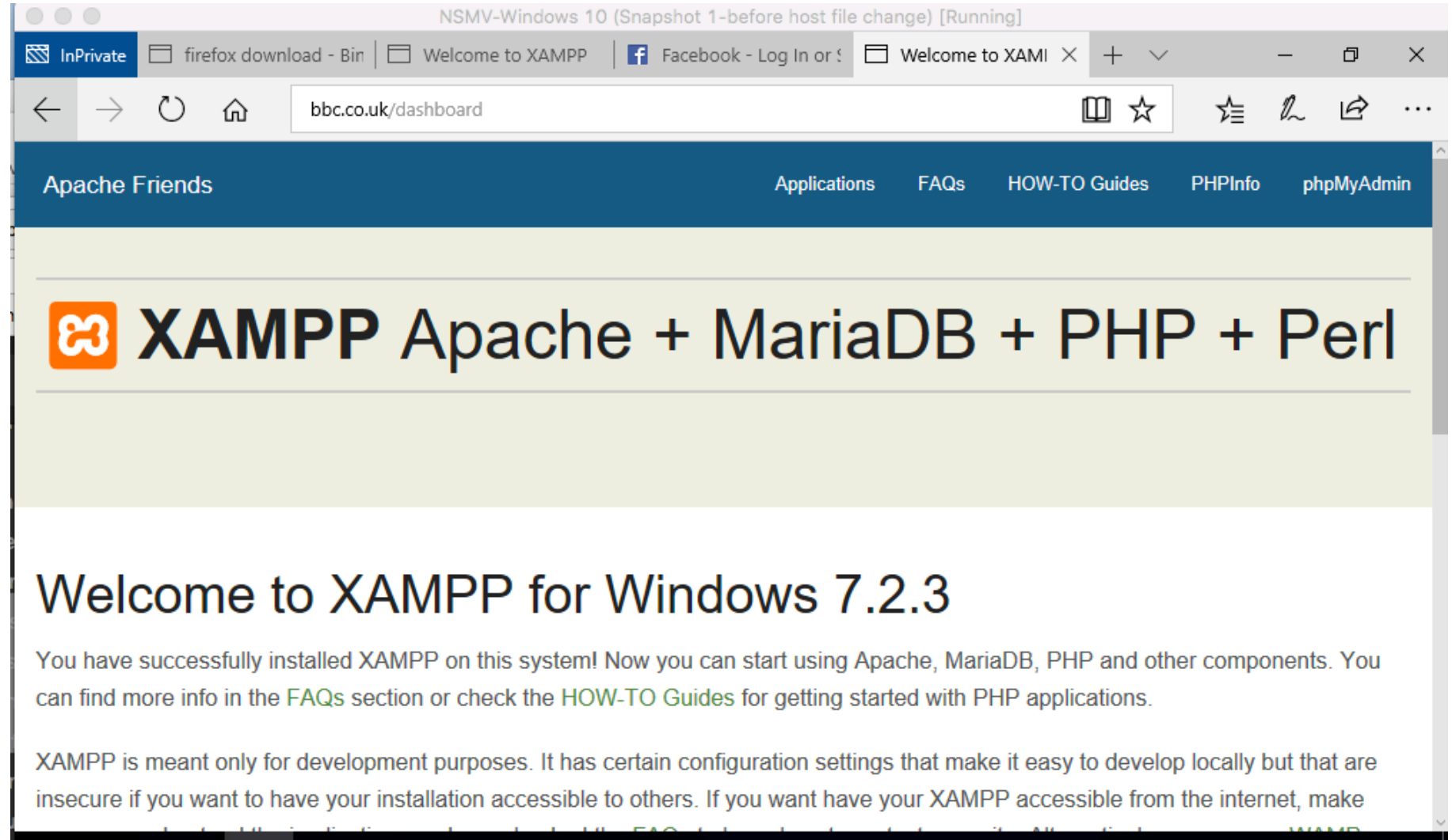
- Here I have added a new entry...
- I'm re-directing traffic from the domain www.bbc.co.uk to my local web server (localhost 127.0.0.1)
- I've installed XAMPP...
- Apache will appear on my screen



```
hosts - Notepad
File Edit Format View Help
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com           # source server
#       38.25.63.10      x.acme.com               # x client host
#
# localhost name resolution is handled within DNS itself.
#       127.0.0.1        localhost
#       ::1             localhost
|
127.0.0.1 www.bbc.co.uk
```


Redirection...

- Notice the difference?



Details about the hosts file

- The hosts file is a very, very simple file.
- There's not a lot you can do, **but it is very powerful.**
- So this starts out by telling us that this is a sample hosts file.
- It contains mappings of IP addresses to hosts names. Each entry should be kept on an individual line.
- The IP address should be placed in the first column, followed by the corresponding hosts name.
- The IP address and the hosts name should be separated by at least one space

Noting your dns cache

- Viewing **ipconfig /displaydns**

```
www.bbc.co.uk
-----
Record Name . . . . . : www.bbc.co.uk
Record Type . . . . . : 1
Time To Live . . . . . : 86400
Data Length . . . . . : 4
Section . . . . . : Answer
A (Host) Record . . . : 127.0.0.1
```

Exercise – Activity 1

- To prepare for this activity: Start Windows VM. Log in if necessary.
- **Activity 1 - View the hosts File**
- Host name to IP address resolution on Windows is a combination of the contents of the hosts file and the records returned from the specified DNS server(s) in response to DNS queries, with the hosts file being queried first. To understand the entries you will see in the DNS resolver cache, you should first examine the contents of the hosts file.
- View the hosts file.
- **Add a new record (as shown on previous slides) e.g. try re-directing traffic from the domain `www.bbc.co.uk` your local web server (`localhost 127.0.0.1`)-if you don't have apache setup on local machine it will just error can't reach this page**

Activity 2-View DNS Server Settings

- Open a command prompt
- Use **ipconfig /all** to display all IP configuration information.
- Observe the **DNS Servers** settings.
- **The DNS servers are queried for any name that has to be resolved into an IP address that does not exist in the hosts file.**
- Common examples include server names for URLs typed into an Internet browser. The DNS resolver cache will contain a history of names that have been resolved recently.

Activity 3 - Display the Contents of the DNS Resolver Cache

- To display the contents of the DNS resolver cache:
- Type **ipconfig /displaydns** and press Enter.
- Observe the contents of the DNS resolver cache.
- It is generally not necessary to view the contents of the DNS resolver cache, but this activity may be performed as a name resolution troubleshooting method.

Activity 4 - Purge the DNS Resolver Cache

- To purge the DNS resolver cache:
- Type **ipconfig /flushdns** and press **Enter**.
- Type **ipconfig /displaydns** and press **Enter** to observe the contents of the DNS resolver cache and verify that it has been purged. It may be necessary to purge the DNS resolver cache when updating records on a corporate DNS server or adjusting incorrect proxy server settings.
Note: After purging, the DNS resolver cache will still have records for any entry in the hosts file.
- Close the command prompt to complete this activity.

Activity 5 – Open a web browser

- Open a web browser and navigate to a new site
 - Type **ipconfig /displaydns** and press Enter.
-
- Notice how your dns cache has been updated to include new information

Activity 6 – Delete the hosts entry you created in activity 1

- E.g. this may have been your entry to www.bbc.co.uk
- You may now need to flush your dns cache

DNS queries

- **Iterative Queries**

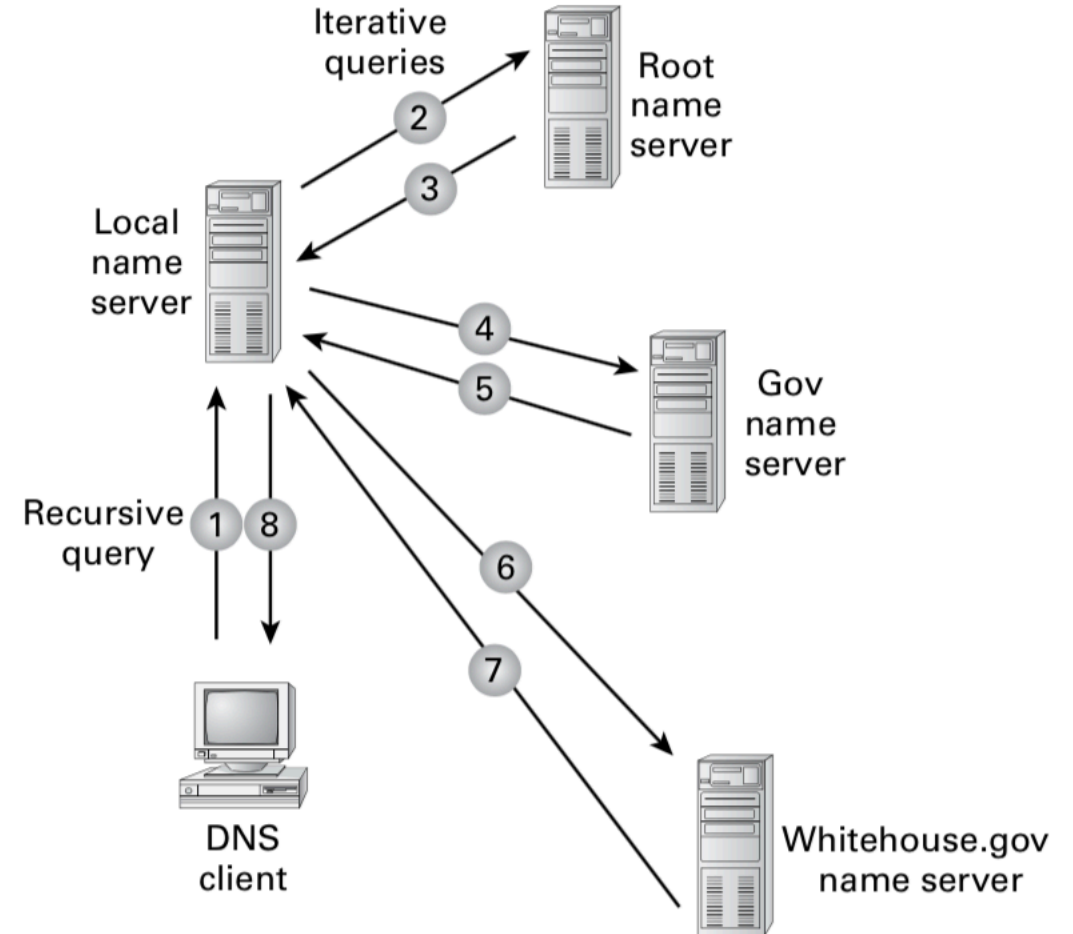
- Iterative queries are the easiest to understand: A client asks the DNS server for an answer, and the server returns the best answer. This information likely comes from the server's cache. The server never sends out an additional query in response to an iterative query. If the server doesn't know the answer, it may direct the client to another server through a referral.

- **Recursive Queries**

- In a recursive query, the client sends a query to a name server, asking it to respond either with the requested answer or with an error message. The error states one of two things: The server can't come up with the right answer. The domain name doesn't exist.
- In a recursive query, the name server isn't allowed just to refer the client to some other name server. Most resolvers use recursive queries.
- In addition, if your DNS server uses a forwarder, the requests sent by your server to the forwarder will be recursive queries.

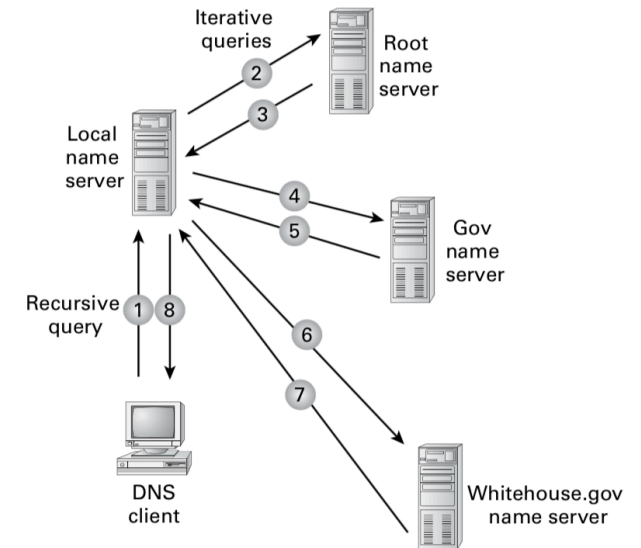
Anatomy of a DNS Query

- Here is an example of both recursive and iterative queries.
- In this example, a client within the Microsoft Corporation is querying its DNS server for the IP address for `www.whitehouse.gov`.



Local server typically does recursive

DNS query to www.whitehouse.gov



- Here's what happens to resolve the request:

1. The resolver sends a **recursive DNS** query to its local DNS server asking for the IP address of www.whitehouse.gov. The local name server is responsible for resolving the name, and it cannot refer the resolver to another name server.
2. The local name server checks its zones, and it finds no zones corresponding to the requested domain name. Sends query to . root name server domain.
3. The root name server has authority for the root domain, and it will reply with the IP address of a name server for the .gov top-level domain.
4. The local name server sends an **iterative** query for www.whitehouse.gov to the Gov name server.
5. The Gov name server replies with the IP address of the name server servicing the whitehouse.gov domain.
6. The local name server sends an iterative query for www.whitehouse.gov to the whitehouse.gov name server.
7. The whitehouse.gov name server replies with the IP address corresponding to www.whitehouse.gov.
8. The local name server sends the IP address of www.whitehouse.gov back to the original resolver.

Seeing recursion

- Step 1: Open Wireshark and start a capture.
- From client-ping www.facebook.com

Command Prompt

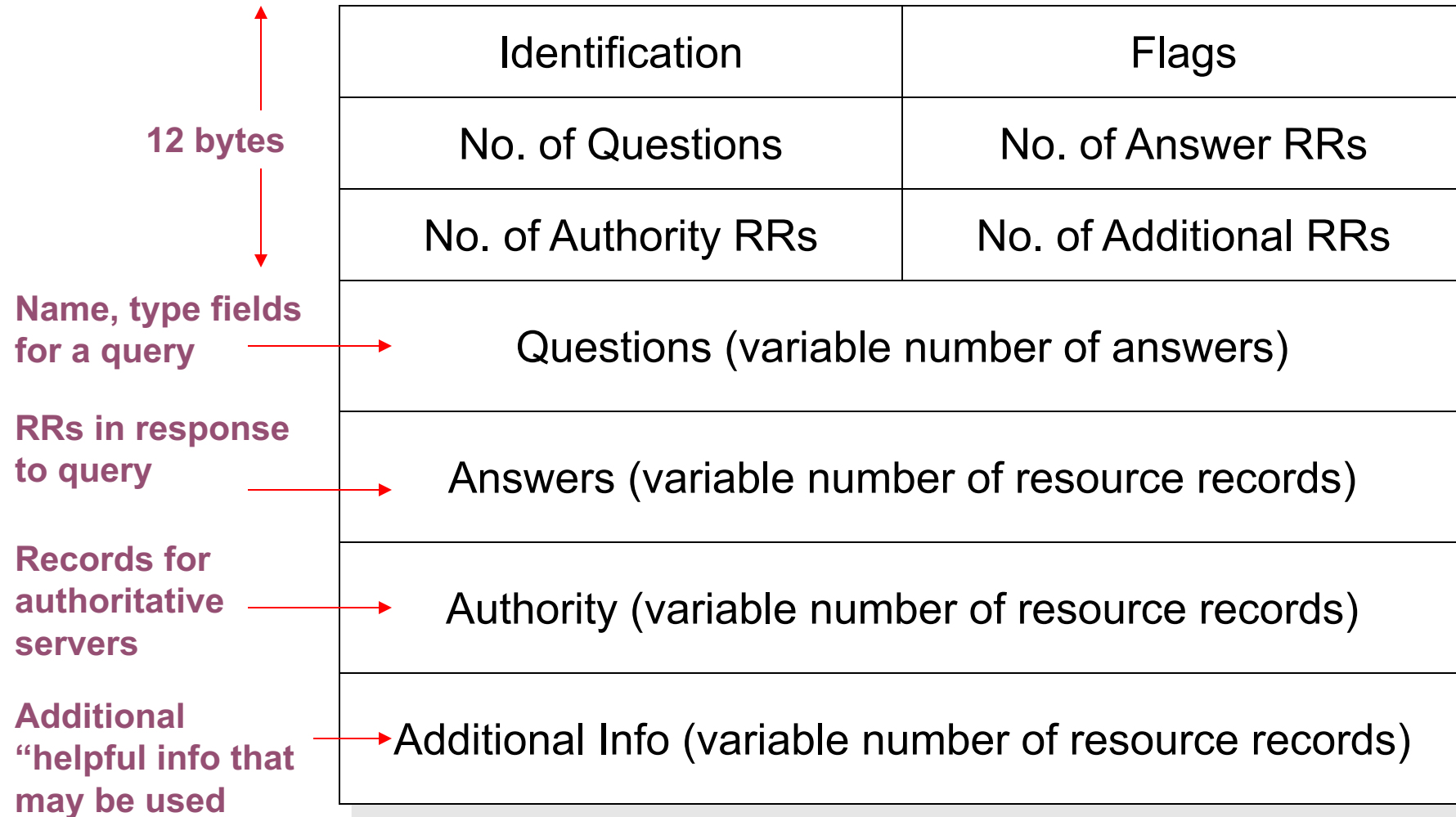
```
C:\Users\test123>
C:\Users\test123>ping www.facebook.com

Pinging star-mini.c10r.facebook.com [31.13.73.35] with 32 bytes of data:
Reply from 31.13.73.35: bytes=32 time=48ms TTL=127
Reply from 31.13.73.35: bytes=32 time=46ms TTL=127
Reply from 31.13.73.35: bytes=32 time=46ms TTL=127
Reply from 31.13.73.35: bytes=32 time=67ms TTL=127

Ping statistics for 31.13.73.35:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 46ms, Maximum = 67ms, Average = 51ms

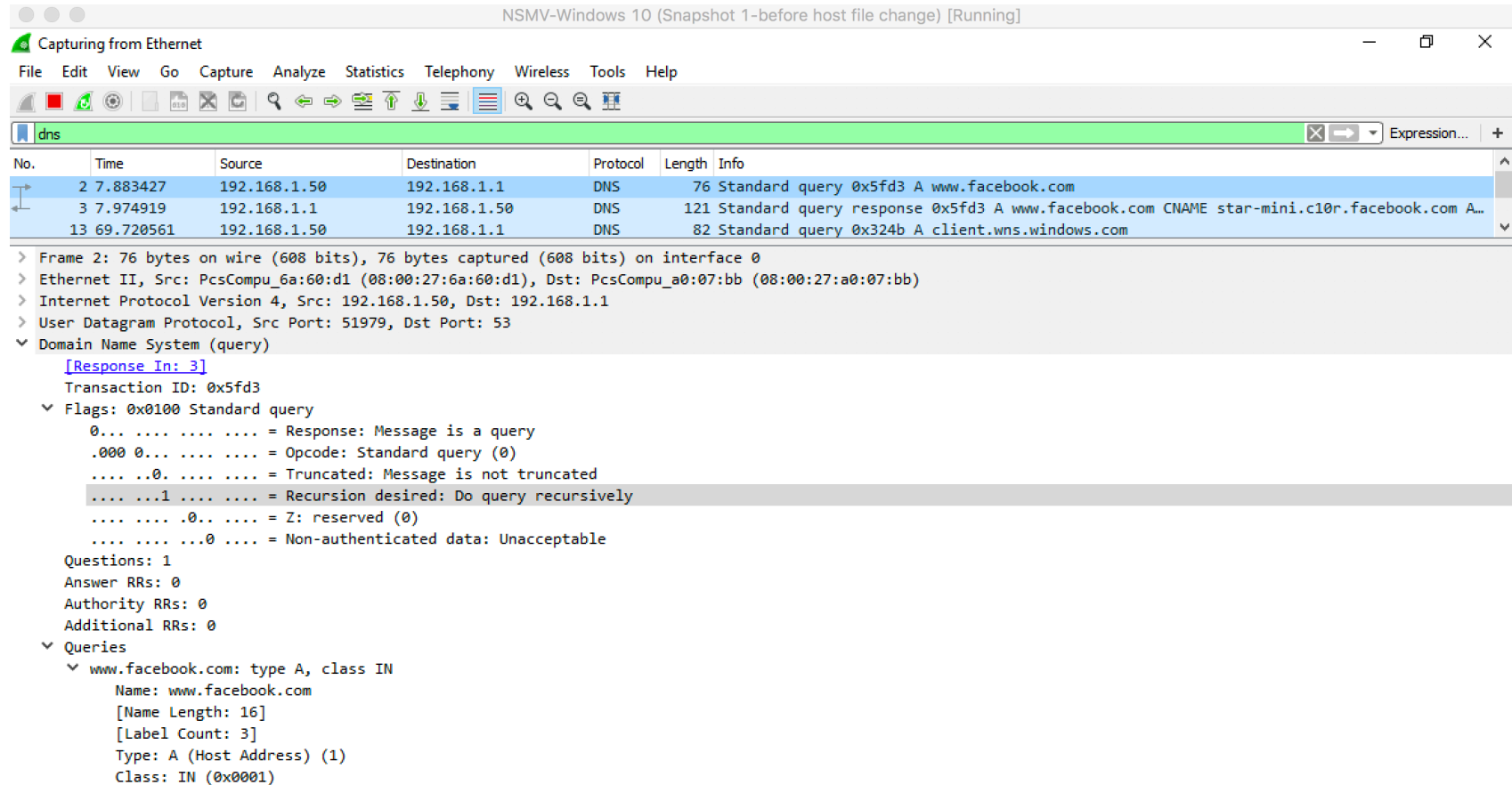
C:\Users\test123>
```

DNS Message Format



DNS query - Seeing recursion

- Step 2: Add a DNS filter in Wireshark
- You will see query (as shown) below – notice the flag 1 =Recursion desired



The screenshot shows the Wireshark interface with a filter set to 'dns'. The packet list pane displays three DNS packets:

No.	Time	Source	Destination	Protocol	Length	Info
2	7.883427	192.168.1.50	192.168.1.1	DNS	76	Standard query 0x5fd3 A www.facebook.com
3	7.974919	192.168.1.1	192.168.1.50	DNS	121	Standard query response 0x5fd3 A www.facebook.com CNAME star-mini.c10r.facebook.com A...
13	69.720561	192.168.1.50	192.168.1.1	DNS	82	Standard query 0x324b A client.wns.windows.com

The details pane for the selected packet (No. 2) shows the following information:

- Frame 2: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0
- Ethernet II, Src: PcsCompu_6a:60:d1 (08:00:27:6a:60:d1), Dst: PcsCompu_a0:07:bb (08:00:27:a0:07:bb)
- Internet Protocol Version 4, Src: 192.168.1.50, Dst: 192.168.1.1
- User Datagram Protocol, Src Port: 51979, Dst Port: 53
- Domain Name System (query)
 - [Response In: 3]
 - Transaction ID: 0x5fd3
 - Flags: 0x0100 Standard query
 - 0... .. = Response: Message is a query
 - .000 0... .. = Opcode: Standard query (0)
 -0... .. = Truncated: Message is not truncated
 -1... .. = Recursion desired: Do query recursively
 -0... .. = Z: reserved (0)
 -0... .. = Non-authenticated data: Unacceptable
 - Questions: 1
 - Answer RRs: 0
 - Authority RRs: 0
 - Additional RRs: 0
 - Queries
 - www.facebook.com: type A, class IN
 - Name: www.facebook.com
 - [Name Length: 16]
 - [Label Count: 3]
 - Type: A (Host Address) (1)
 - Class: IN (0x0001)

DNS answer

- A recursive query is when a client asks the DNS server for specific information, and it expects the answer to come from that DNS server.
- Notice the DNS answer: 31.13.73.35

The screenshot shows a Wireshark capture of a DNS response. The packet list pane shows three DNS packets. Packet 3 is the response, containing two CNAME records for www.facebook.com pointing to star-mini.c10r.facebook.com and star-mini.c10r.facebook.com pointing to 31.13.73.35.

No.	Time	Source	Destination	Protocol	Length	Info
2	7.883427	192.168.1.50	192.168.1.1	DNS	76	Standard query 0x5fd3 A www.facebook.com
3	7.974919	192.168.1.1	192.168.1.50	DNS	121	Standard query response 0x5fd3 A www.facebook.com CNAME star-mini.c10r.facebook.com A...
13	69.720561	192.168.1.50	192.168.1.1	DNS	82	Standard query 0x324b A client.wns.windows.com

```
> Frame 3: 121 bytes on wire (968 bits), 121 bytes captured (968 bits) on interface 0
> Ethernet II, Src: PcsCompu_a0:07:bb (08:00:27:a0:07:bb), Dst: PcsCompu_6a:60:d1 (08:00:27:6a:60:d1)
> Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.50
> User Datagram Protocol, Src Port: 53, Dst Port: 51979
v Domain Name System (response)
  [Request In: 2]
  [Time: 0.091492000 seconds]
  Transaction ID: 0x5fd3
  > Flags: 0x8180 Standard query response, No error
  Questions: 1
  Answer RRs: 2
  Authority RRs: 0
  Additional RRs: 0
v Queries
  v www.facebook.com: type A, class IN
    Name: www.facebook.com
    [Name Length: 16]
    [Label Count: 3]
    Type: A (Host Address) (1)
    Class: IN (0x0001)
v Answers
  > www.facebook.com: type CNAME, class IN, cname star-mini.c10r.facebook.com
  > star-mini.c10r.facebook.com: type A, class IN, addr 31.13.73.35
```


On the DNS server...(on Windows Server 2012)

- Recursion desired =0. There is not a 1, there's a 0, which means you don't have to do it recursively. You can refer me to another DNS server to help me find the answer. So that's how easy it is to tell the difference between a recursive and an **iterative query**.

The screenshot shows a Wireshark capture of a DNS query. The packet list pane shows three packets: a query from 172.20.10.2 to 69.171.239.11, a response from 69.171.239.11 to 172.20.10.2, and another query from 172.20.10.2 to 208.76.45.53. The packet details pane for the first packet (No. 2) shows the following flags: 0x0000 Standard query. The 'Recursion desired' flag is highlighted in blue, showing its value as 0. The packet bytes pane shows the raw DNS query structure, including the transaction ID 0xa3ea and the domain name star-mini.c10r.facebook.com.

No.	Time	Source	Destination	Protocol	Length	Info
2	3.645794	172.20.10.2	69.171.239.11	DNS	98	Standard query 0xa3ea A star-mini.c10r.facebook.com OPT
3	3.702146	69.171.239.11	172.20.10.2	DNS	237	Standard query response 0xa3ea A star-mini.c10r.facebook.com A 31.13.73.35 NS a.ns.c10r.facebook.co...
7	44.728558	172.20.10.2	208.76.45.53	DNS	99	Standard query 0x5ef1 A cdn.content.prod.cms.msn.com OPT

```
4 Domain Name System (query)
  [Response In: 3]
  Transaction ID: 0xa3ea
  4 Flags: 0x0000 Standard query
    0... .. = Response: Message is a query
    .000 0... .. = Opcode: Standard query (0)
    .... ..0. .... = Truncated: Message is not truncated
    .... ..0 .... = Recursion desired: Don't do query recursively
    .... ..0... .. = Z: reserved (0)
    .... ..0 .... = Non-authenticated data: Unacceptable
  Questions: 1
  Answer RRs: 0
  Authority RRs: 0

0000  ba 53 ac 99 bc 64 08 00 27 54 4d 70 08 00 45 00  .S...d.. 'TMp..E.
0010  00 54 78 86 00 00 80 11 00 00 ac 14 0a 02 45 ab  .Tx.....E.
0020  ef 0b e7 78 00 35 00 40 eb 1e a3 ea 00 00 01  .x.5.@ ...
0030  00 00 00 00 00 01 09 73 74 61 72 2d 6d 69 6e 69  .....s tar-mini
0040  04 63 31 30 72 08 66 61 63 65 62 6f 6f 6b 03 63  .c10r.fa cebook.c
0050  6f 6d 00 00 01 00 01 00 00 29 0f a0 00 00 80 00  om.....).....
0060  00 00  ..
```

Do query recursively? (dns.flags.recdesired), 2 bytes | Packets: 29 · Displayed: 18 (62.1%) | Profile: Default

Root hints

- You might be asking yourself, how did that local name server know who the Root Server was? There's something called **Root Hints**, and yes, of course, on the internet there's more than one.
- They're stored in a file called CACHE.DNS, and you'll find them on a Windows Server 2012 machine in the Windows\system32\dns folder.
- What denotes a root server? It's that special little dot at the end there. So this is a listing of all of your root servers. Notice there aren't that many. Just about 13.
- The purpose of the root server is to get you to that next level piece of the domain name, which would be com, net, org, whatever.
- In summary: Approx. 13 root name servers worldwide
- These are replicated worldwide
- Currently {a-m}.root-servers.net
- Local name servers contact root servers when they cannot resolve a name

2018111-SER2012 Properties

Debug Logging | Event Logging | Monitoring | Security
Interfaces | Forwarders | Advanced | Root Hints

Root hints resolve queries for zones that do not exist on the local DNS server. They are only used if forwarders are not configured or fail to respond.

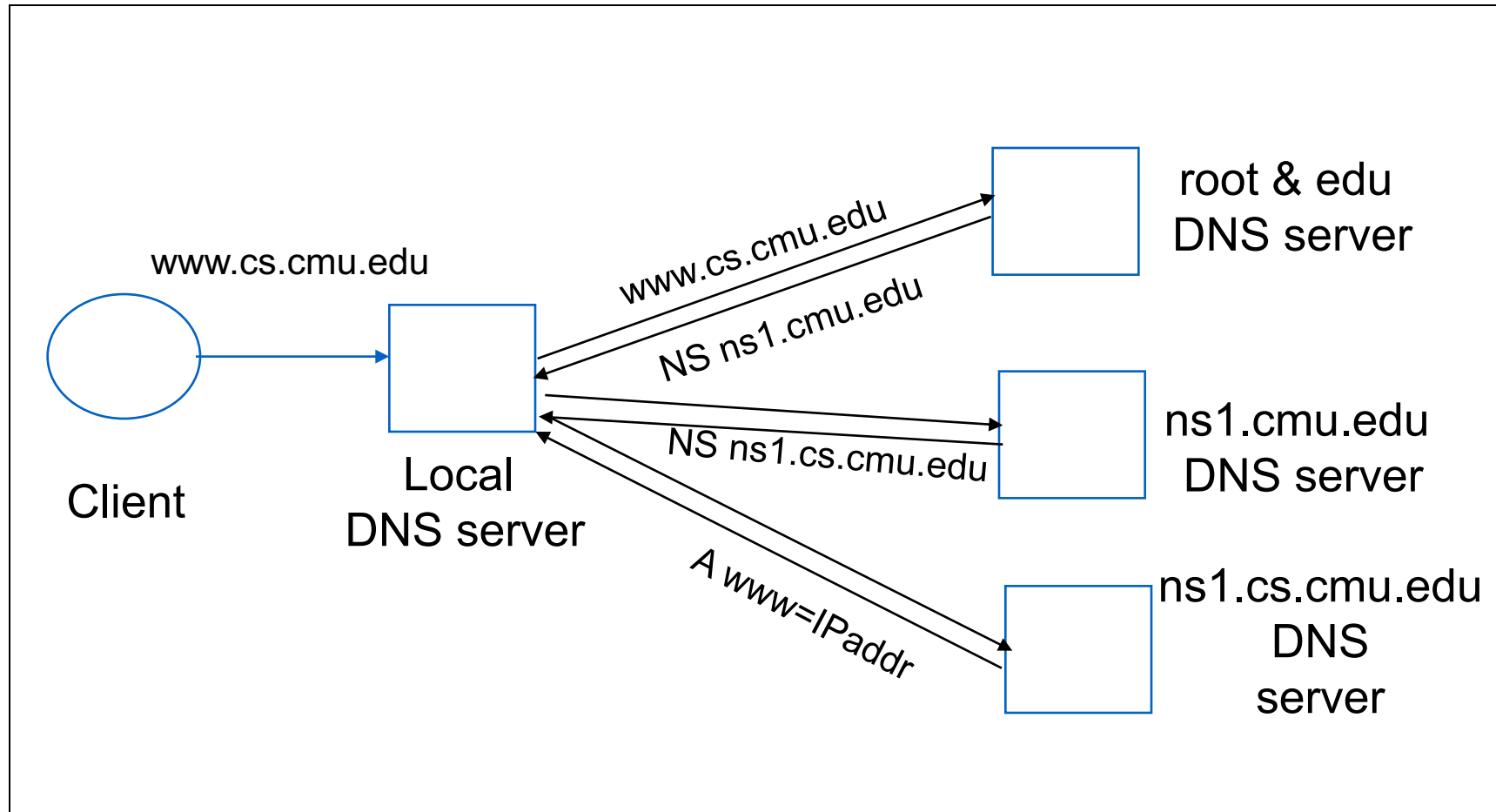
Name servers:

Server Fully Qualified Domain Name (FQDN)	IP Address
a.root-servers.net.	[198.41.0.4]
b.root-servers.net.	[192.228.79.201]
c.root-servers.net.	[192.33.4.12]
d.root-servers.net.	[199.7.91.13]
e.root-servers.net.	[192.203.230.10]
f.root-servers.net.	[192.5.5.241]
g.root-servers.net.	[192.112.36.4]
h.root-servers.net.	[128.63.2.53]
i.root-servers.net.	[192.36.148.17]

Add... Edit... Remove Copy from Server

OK Cancel Apply Help

Another Typical DNS Resolution

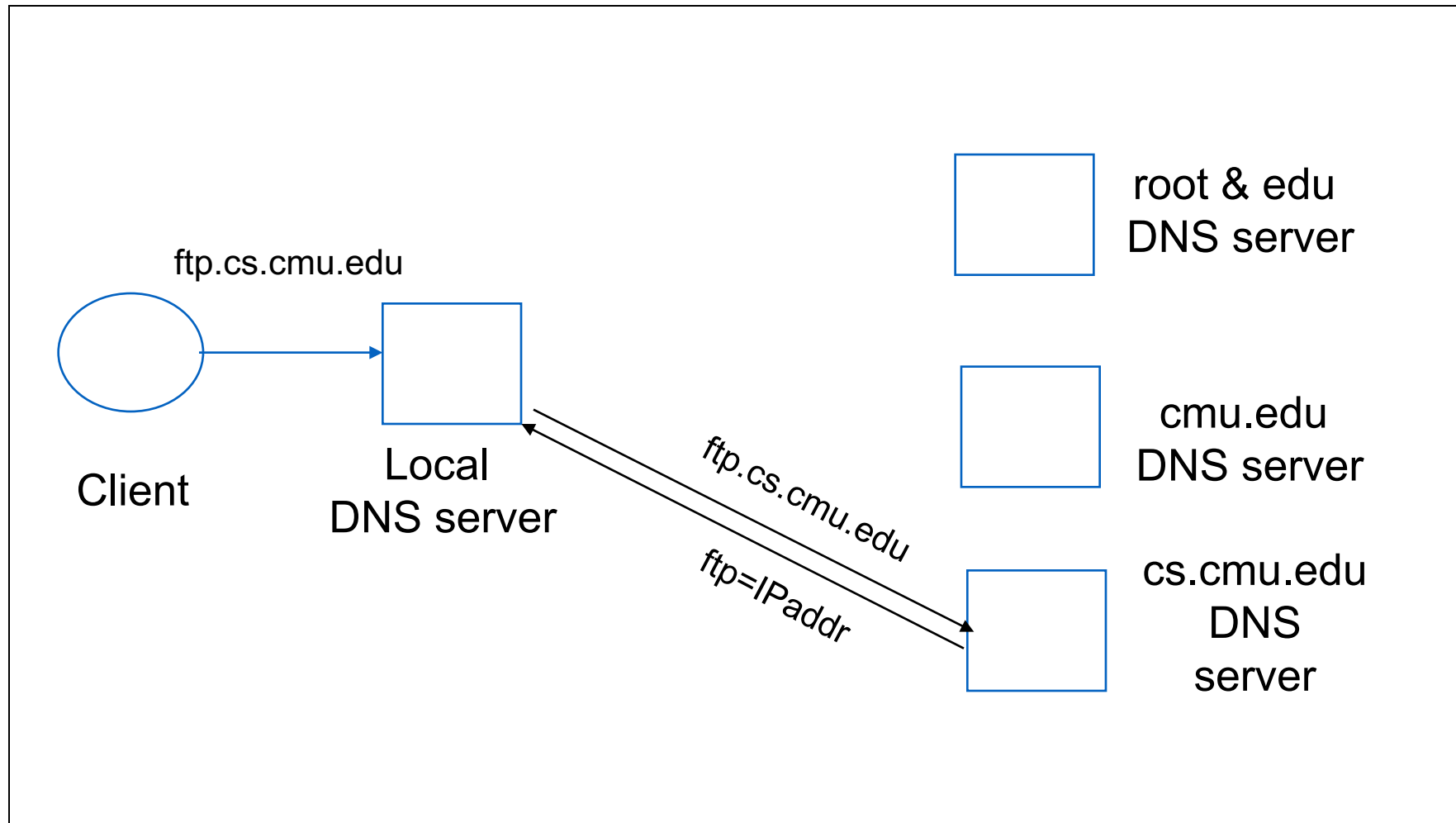


We would now like to lookup - ftp.cs.cmu.edu
Do we start the process all over again?

DNS caching

- DNS responses are cached
 - Quick response for repeated translations
 - Other queries may reuse some parts of lookup
 - NS records for domains
- Cached data periodically times out
 - Lifetime time to live (TTL) of data controlled by owner of data
 - TTL passed with every record

Subsequent Lookup Example



DNS zone

- A DNS zone is a portion of the DNS namespace over which a specific DNS server has authority.
- Within a given DNS zone, there are resource records (RRs) that define the hosts and other types of information that make up the database for the zone.

DNS Zone Types

- When it comes to DNS zones, we actually have two different types.
- The most common type is a **forward lookup zone**. Everybody knows this is how DNS works. The point to a forward lookup zone is that you have a computer name or a host name, and you'd like to get the IP address for that. So, host name to IP address resolution, that's a forward lookup.
- Forward lookup zones contain authoritative data.
- **Authoritative** data means that the data lives on the server where the client got the DNS response from.
- **Non-authoritative** answers were when the DNS server that the client asked the query of had to go out and find it someplace else, and then that DNS server would put it in its cache.

DNS Zone Types

- We also have **reverse lookup zones**.
- Now these are not as popular as forward lookup zones, but they are still used.
- The purpose here is that **you have the IP address and you need to get a resolution to the host name**, *so you're going the opposite direction*.
- Some examples of when a reverse lookup zone would be used, well, Microsoft Exchange Servers and other types of mail servers like to verify source domains of inbound emails to make sure they're coming from valid domain names. Also, there are tools, some old tools that like to use reverse lookup zones.
- A couple of those are NSLookup, traceroute, and even the simple mail transfer protocol, or SMTP.

Different types of resource records

- **Zone files consist of a number of resource records.** You need to know about several types of resource records to manage your DNS servers effectively. There are different types of resource records that could be placed in a forward lookup zone,
- One of the most common resource records out there are a **host record**. An **A record** is an IPv4 host record.
- An **AAAA**, well, that's an IPv6 host record.
- **PTR**, or pointer records, are the records that reside in a reverse lookup zone that point to the host record in the forward lookup zone.
- A **CName**, or canonical name, is really a nickname of a machine. So if you have a server in your environment, and you'd like for people to access it using different names for whatever reason, you can set up CName records to point to a host record.
- An **SRV** record, or service locator record, denotes different types of services running on a machine. For example, in a Microsoft environment, if a client machine wanted to find a domain controller, it would send a query to a DNS server asking for the SRV records for the domain, and then it would get a list of domain controllers or DCs that it could authenticate against for whatever purpose the user is doing.
- **MX records**. These are mail exchange records, and they denote a mail server. In a Microsoft environment, it would be an exchange server.

Host Record

- A host record (also called an A record for IPv4 and AAAA record for IPv6) is used to associate statically a host's name to its IP addresses. The format is pretty simple:
- Here's an example from a DNS database:
- `www IN A 192.168.1.50`
- The A or AAAA record ties a hostname (which is part of an FQDN) to a specific IP address. This makes these records suitable for use when you have devices with statically assigned IP addresses. In this case, you create these records manually using the DNS snap-in.

Pointer Record

- the host record has a lesser-known but still important twin: the *pointer (PTR) record*. The format of a PTR record appears as follows:
- reversed_address.in-addr.arpa. optional_TTL IN PTR targeted_domain_name

Pointer Record

- The A or AAAA record maps a hostname to an IP address, and the PTR record does just the opposite—mapping an IP address to a hostname through the use of the in-addr.arpa zone.
- The PTR record is necessary because IP addresses begin with the least-specific portion first (the network) and end with the most-specific portion (the host), whereas hostnames begin with the most-specific portion at the beginning and the least-specific portion at the end.
- Consider the example 192.168.1.10 with a subnet mask 255.255.255.0. The portion 192.168.1 defines the network and the final .10 defines the host, or the most-specific portion of the address. DNS is just the opposite: The hostname www.example.com. defines the most-specific portion, www, at the beginning and then traverses the DNS tree to the least-specific part, the dot (.), at the root of the tree.
- Reverse DNS records, therefore, need to be represented in this most-specific-to-least-specific manner. The PTR record for mapping 192.168.1.10 to www.example.com would look like this:
- 10.1.168.192.in-addr.arpa. IN PTR www.example.com.
- Now a DNS query for that record can follow the logical DNS hierarchy from the root of the DNS tree all the way to the most-specific portion.

Mail Exchanger Record

- The *mail exchanger (MX) record* is used to specify which servers accept mail for this domain. Each MX record contains two parameters—a preference and a mail server, as shown in the following example:
- The MX record uses the preference value to specify which server should be used if more than one MX record is present. The preference value is a number. The lower the number, the more preferred the server. Here's an example:
 - example.com. IN MX 0 mail.example.com.
 - example.com. IN MX 10 backupmail.example.com.
- In the example, mail.example.com is the default mail server for the domain. If that server goes down for any reason, emailers then use the backupmail.example.com mail server.

Other DNS Record Types

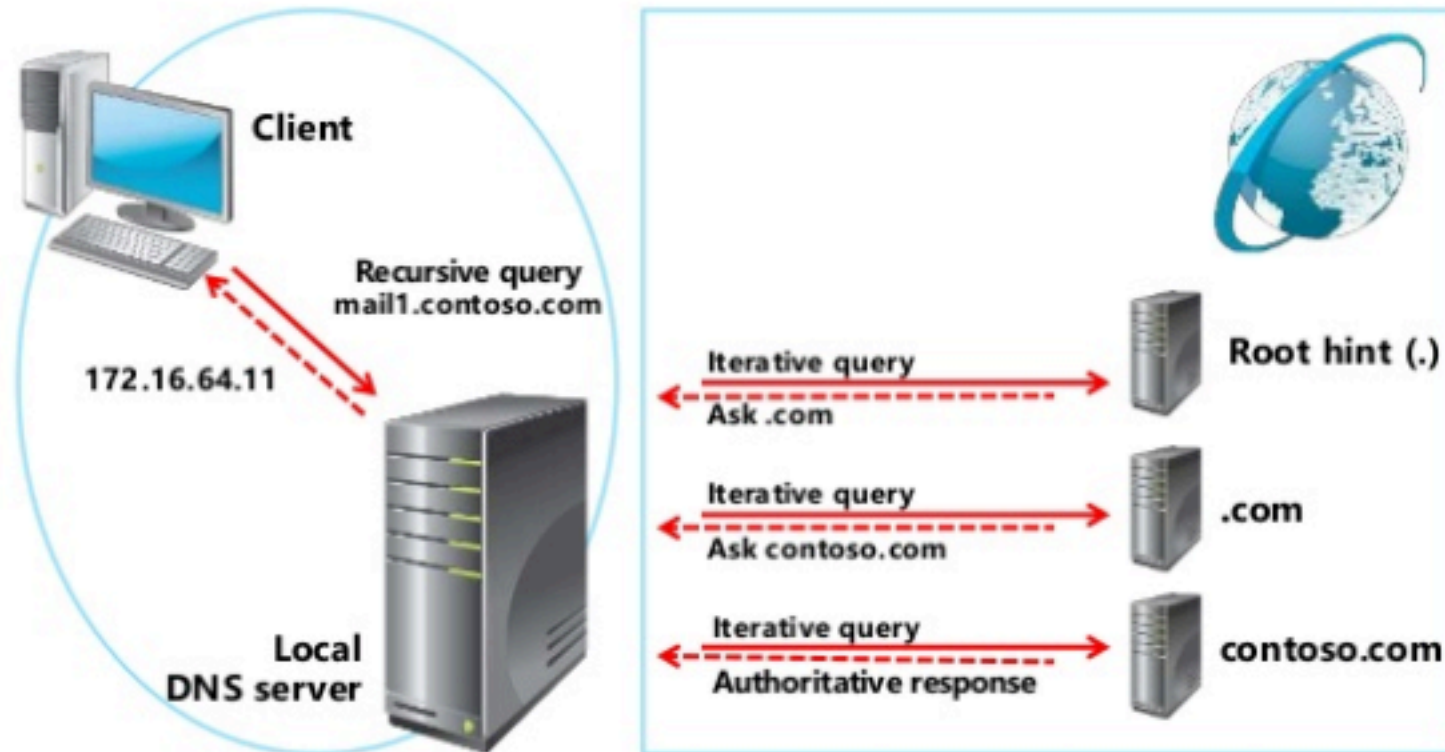
Start of Authority records (SOA)

The first record in a database file is the start of authority (SOA) record. The SOA defines the general parameters for the DNS zone, including the identity of the authoritative server for the zone.

Name Server records (NS)

- *Name server (NS) records* list the name servers for a domain. This record allows other name servers to look up names in your domain. A zone file may contain more than one name server record. The format of these records is simple:
- `example.com. IN NS Hostname.example.com`

Another example...



Using a DNS forwarder

- If a DNS server does not have an answer to a DNS request, it may be necessary to send that request to another DNS server. This is called DNS forwarding.
- When a DNS server forwards an external DNS request to a DNS server outside of your organization, this is considered external forwarding. For example, request the host `www.microsoft.com`. Most likely, your internal DNS server is not going to have Microsoft's web address in its DNS database. So, your DNS server is going to send the request to an external DNS (most likely your ISP).

