DNS

What is DNS?

- The **Domain Name System** (**DNS**) is used to resolve human-readable hostnames like **www.google.com** into machine-readable IP addresses like **8.8.8.8** . **DNS** also provides other information about domain names, such as mail services

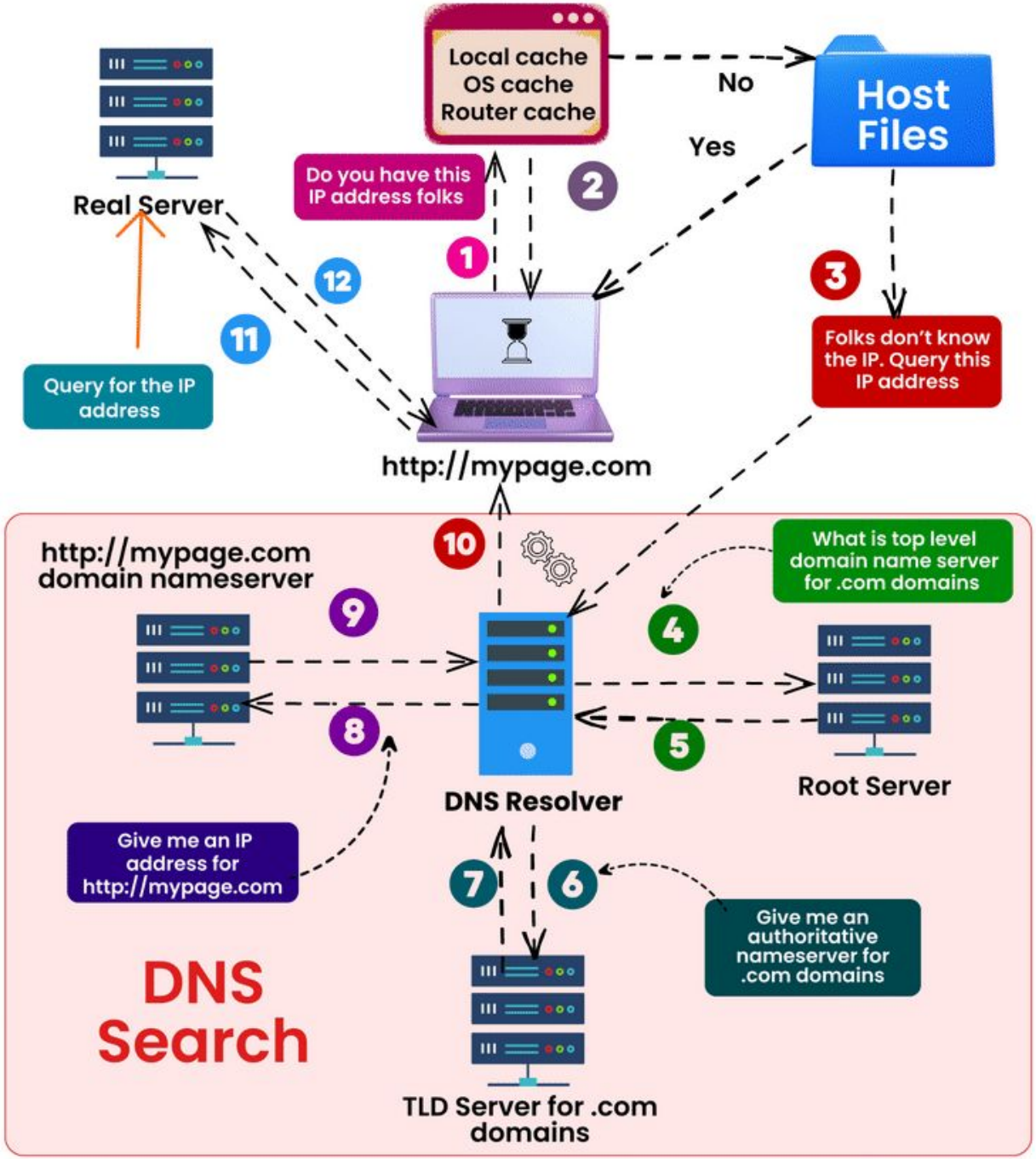
Why is DNS important?

- DNS is like a phone book for the Internet. If we know a person's name but don't know their telephone number, we can simply look it up in a phone book. DNS provides this same service to the Internet.

- When you visit **http://www.google.com** in a browser, your computer uses DNS to retrieve the website's IP address of **8.8.8.8**. Without DNS, you would only be able to visit the website (or any website) by visiting its IP address directly, such as **8.8.8.8**.

How does DNS work?

- When you visit a domain such as ***google.com***, your computer follows a series of steps to turn the human-readable web address into a machine-readable IP address. This happens every time you use a domain name, whether you are viewing websites, sending email


Local cache
OS cache
Router cache
No
Host Files
Yes
Do you have this IP address folks
2
1
3
Folks don't know the IP. Query this IP address
Real Server
12
11
Query for the IP address
http://mypage.com
10
http://mypage.com domain nameserver
What is top level domain name server for .com domains
9
4
8
5
DNS Resolver
Root Server
Give me an IP address for http://mypage.com
7
6
Give me an authoritative nameserver for .com domains
DNS Search
TLD Server for .com domains

Step 1: Request information

- The process begins when you ask your computer to resolve a hostname, such as visiting *google.com*. The first place your computer looks is its **local DNS cache**, which stores information that your computer has recently retrieved.

- If your computer doesn't already know the answer, it needs to perform a **DNS query** to find out.

Step 2: Ask the Recursive DNS Servers

- If the information is not stored locally, your computer queries (contacts) your ISP's **Recursive DNS servers**.
- These specialized computers perform the legwork of a DNS query on your behalf.
- Recursive servers have their own caches, so the process usually ends here and the information is returned to the user.

Step 3: Ask the root Nameservers

- If the recursive servers don't have the answer, they query the **root nameservers**.
- A **nameserver** is a computer that answers questions about domain names, such as IP addresses.
- The twenty two root name servers act as a kind of telephone switchboard for DNS.
- They don't know the answer, but they can direct our query to someone that knows where to find it.

Step 4: Ask the TLD nameservers

- The root nameservers will look at the first part of our request, reading from right to left — **www.google.*com*** — and direct our query to the **Top-Level Domain (TLD) nameservers** for ***.com***.

- Each TLD, such as ***.com*, *.org*,** and ***.org***, have their own set of nameservers, which act like a receptionist for each TLD. These servers don't have the information we need, **but they can refer us directly** to the servers that **do have the information**.

Step 5: Ask the Authoritative DNS servers

- The TLD nameservers review the next part of our request — ***www.google.com*** — and direct our query to the nameservers responsible for this specific domain.

- These **Authoritative nameservers** are responsible for knowing all the information about a specific domain, which are stored in **DNS records**.

- There are many types of records, which each contain a different kind of information. In this example, we want to know the IP address for *www.google.com*, so we ask the authoritative nameserver for the **Address Record (A)**.

Step 6: Retrieve the record

- The recursive server retrieves the **A** record for ***google.com*** from the authoritative nameservers and stores the record in its local cache.

- If anyone else requests the host record for ***google.com***, the recursive servers will already have the answer and will not need to go through the lookup process again.

- All records have a **time-to-live** value, which is like an expiration date. After a while, the recursive server will need to ask for a new copy of the record to make sure the information doesn't become out-of-date.

Step 7: Receive the Answer

- Armed with the answer, recursive server returns the **A** record back to your computer.
- Your computer stores the record in its cache, reads the IP address from the record, then passes this information to your browser.
- The browser then opens a connection to the webserver and receives the website.
- **This entire process, from start to finish, takes only milliseconds to complete.**

Cont'd…

- **Domain Name System (DNS)** translates between ***domain names*** and ***IP addresses***, and is supported by nearly every operating system.

- All Internet based name resolution utilizes DNS. DNS is organized as a hierarchy. Consider the following translation.

- Consider the following translation:

  www.google.com = 8.8.8.8

Cont'd…

- The above domain name represents a **Fully Qualified Domain Name (FQDN):**
- **.com** represents a top level domain.
- **.google** represents a secondary level domain
- **www** represents a host computer in the .google.com domain

- Other top level domains include **.org, .net,** and **.gov.**
- Top level domains can also include country codes, such as **.in**, **.us**, **.ca**, **.cn**, and **.ru.**

DNS Record Types

There are a variety of DNS record types, including:

- **NS (Name Server)** – identifies a DNS server for the domain.
- **SOA (Start of Authority)** – identifies the primary (authoritative) DNS server for the domain.
- **A (Address)** – identifies an individual host in the domain.
- **CNAME (Canonical Name)** – assigns an alias for another host name.
- **MX (Mail Exchanger)** - identifies a mail server in the domain.
- **PTR (Pointer)** - used for *reverse* DNS lookups.

The number defined in the MX record is a *priority*. A *lower* priority is more preferred.

Nslookup

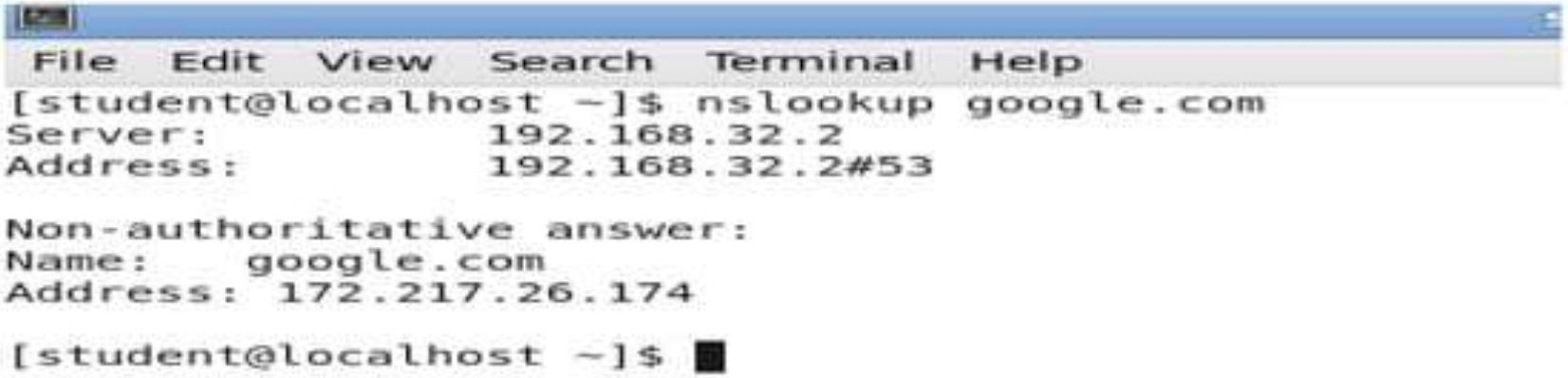
- Nslookup is a network administration tool for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or any other specific DNS record.

- It is also used to troubleshoot DNS related problems.

Cont'd..

- nslookup can operate on both "**Interactive mode**" and "**Non-Interactive mode**".
- Interactive mode allows the user to **query the DNS-Server** about various host, and domains.
- Non-Interactive mode allows the user to **query** the information for **a host or domain**.
- All the commands explained are "**Non-Interactive mode**".

- nslookup followed by the domain name will display the "**A Record**" (IP Address) of the domain.



Server refers to the IP address of the DNS server. Then the below section provides the "A Record" (IP Address) of the domain "google.com"

1. Query the MX Record using -query=mx

- MX (Mail Exchange) record maps a domain name to a list of mail exchange servers for that domain. The MX record tells that all the mails sent to "@google.com" should be routed to the Mail server in that domain.

```
$ nslookup -query=mx redhat.com
Server:     192.168.19.2
Address:    192.168.19.2#53

Non-authoritative answer:
redhat.com  mail exchanger = 10 mx2.redhat.com.
redhat.com  mail exchanger = 5 mx1.redhat.com.

Authoritative answers can be found from:
mx2.redhat.com  internet address = 66.187.233.33
mx1.redhat.com  internet address = 209.132.183.28
```

In the above example, we have 2 MX records for the domain "redhat.com". The number (5, 10), associated with the MX records tells the preference of mail server. Lower the number, higher the preference. So when a mail is sent to "@redhat.com", first preference will be "mx1.redhat.com", then "mx2.redhat.com".

Authoritative Answer vs. Non Authoritative Answer

- we may have also noticed the keyword "Authoritative Answer" and "Non-Authoritative Answer" in the above output.

- Any answer that originates from the DNS Server which has the complete zone file information available for the domain is said to be authoritative answer.

- In many cases, DNS servers will not have the complete zone file information available for a given domain. Instead, it maintains a cache file which has the results of all queries performed in the past for which it has got authoritative response. When a DNS query is given, it searches the cache file, and return the information available as "Non-Authoritative Answer".

2. Query the NS Record using -query=ns

- NS (Name Server) record maps a domain name to a list of DNS servers authoritative for that domain. It will output the name servers which are associated with the given domain.

```
nslookup -type=ns redhat.com
Server:         192.168.19.2
Address:        192.168.19.2#53

Non-authoritative answer:
redhat.com      nameserver = ns4.redhat.com.
redhat.com      nameserver = ns2.redhat.com.
redhat.com      nameserver = ns1.redhat.com.
redhat.com      nameserver = ns3.redhat.com.

Authoritative answers can be found from:
ns4.redhat.com      internet address = 209.132.188.218
ns2.redhat.com      internet address = 209.132.183.2
ns1.redhat.com      internet address = 209.132.186.218
ns3.redhat.com      internet address = 209.132.176.100
```

3. Query the SOA Record using -query=soa

- SOA record (Start of Authority), provides the authoritative information about the domain, the e-mail address of the domain admin, the domain serial number, etc…

```
$ nslookup -type=soa redhat.com
Server:         192.168.19.2
Address:        192.168.19.2#53

Non-authoritative answer:
redhat.com
        origin = ns1.redhat.com
        mail addr = noc.redhat.com
        serial = 2012071601
        refresh = 300
        retry = 180
        expire = 604800
        minimum = 14400
```

Cont'd..

- **mail addr** – specifies the mail address of the domain admin (noc@redhat.com)
- **serial** – sort of revision numbering system. The standard convention is to use "YYYYMMMYYNN" format. (2012-07-16. 01 will be incremented, if more than one edit has taken place on a same day)
- **refresh** – specifies (in seconds), when the secondary DNS will poll the primary to see if the serial number has been increased. If increased, secondary will make a new request to copy the new zone file.
- **retry** – specifies the interval to re-connect with the Primary DNS
- **expire** – specifies the time that the secondary DNS will keep the cached zone file as valid
- **minimum** – specifies the time that the secondary DNS should cache the zone file

4. Reverse DNS lookup

- You can also do the reverse DNS look-up by providing the IP Address as argument to nslookup.

```
$ nslookup 209.132.183.181

Server:     192.168.19.2

Address:    192.168.19.2#53


Non-authoritative answer:

181.183.132.209.in-addr.arpa    name = origin-www2.redhat.com.
```

Host File

- The computer file **hosts** is an operating system file that maps hostnames to IP addresses. It is a plain text file. Originally a file named HOSTS.txt.
- Assists in addressing network nodes in a computer network
- 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.

Applications

- **Redirecting local domains-**accessing the company's internal resources or to test local websites in development.

- **Internet resource blocking-**achieved by adding entries for those sites to redirect requests to another address that does not exist or to a harmless destination, such as the IP address "0.0.0.0"

Dig

- Using dig command you can query DNS name servers for your DNS lookup related tasks.
- When you pass a domain name to the dig command, by default it displays the A record (the ip-address of the site that is queried).
- In this example, it displays the A record of redhat.com in the "ANSWER SECTION" of the dig command output.

```
$ dig redhat.com

; <<>> DiG 9.7.3-RedHat-9.7.3-2.el6 <<>> redhat.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62863
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 4, ADDITIONAL: 3

;; QUESTION SECTION:
;redhat.com.                    IN      A

;; ANSWER SECTION:
redhat.com.             37      IN      A       209.132.183.81
;; AUTHORITY SECTION:
redhat.com.              73       IN       NS       ns4.redhat.com.
redhat.com.              73       IN       NS       ns3.redhat.com.
redhat.com.              73       IN       NS       ns2.redhat.com.
redhat.com.              73       IN       NS       ns1.redhat.com.

;; ADDITIONAL SECTION:
ns1.redhat.com.          73       IN       A        209.132.186.218
ns2.redhat.com.          73       IN       A        209.132.183.2
ns3.redhat.com.          73       IN       A        209.132.176.100

;; Query time: 13 msec
;; SERVER: 209.144.50.138#53(209.144.50.138)
;; WHEN: Thu Jan 12 10:09:49 2012
;; MSG SIZE  rcvd: 164
```

The dig command output has the following sections:

- **Header:** This displays the dig command version number, the global options used by the dig command, and few additional header information.
- **QUESTION SECTION**: This displays the question it asked the DNS. i.e This is your input. Since we said 'dig redhat.com', and the default type dig command uses is A record, it indicates in this section that we asked for the A record of the redhat.com website.
- **ANSWER SECTION**: This displays the answer it receives from the DNS. i.e This is your output. This displays the **A record of redhat.com**.
- **AUTHORITY SECTION**: This displays the DNS name server that has the authority to respond to this query. Basically this displays available name servers of redhat.com.
- **ADDITIONAL SECTION**: This displays the ip address of the name servers listed in the AUTHORITY SECTION.
- **Stats section** at the bottom displays few dig command statistics including how much time it took to execute this query

1. Query MX Records Using dig -t MX

- To query MX records, pass MX as an argument to the dig command as shown below

```
$ dig redhat.com  MX +noall +answer


; <<>> DiG 9.7.3-RedHat-9.7.3-2.el6 <<>> redhat.com MX +noall +answer
;; global options: +cmd
redhat.com.             513     IN      MX      5 mx1.redhat.com.
redhat.com.             513     IN      MX      10 mx2.redhat.com.
```

2.Query NS Records Using dig -t NS

- To query the NS record use the type NS as shown below

```
$ dig redhat.com NS +noall +answer

; <<>> DiG 9.7.3-RedHat-9.7.3-2.el6 <<>> redhat.com NS +noall +answer
;; global options: +cmd
redhat.com.             558     IN      NS      ns2.redhat.com.
redhat.com.             558     IN      NS      ns1.redhat.com.
redhat.com.             558     IN      NS      ns3.redhat.com.
redhat.com.             558     IN      NS      ns4.redhat.com.
```

3. View ALL DNS Records Types Using dig -t ANY

- To view all the record types (A, MX, NS, etc.), use ANY as the record type as shown below

```
$ dig redhat.com ANY +noall +answer

; <<>> DiG 9.7.3-RedHat-9.7.3-2.el6 <<>> redhat.com ANY +noall +answer
;; global options: +cmd
redhat.com.             430     IN      MX      5 mx1.redhat.com.
redhat.com.             430     IN      MX      10 mx2.redhat.com.
redhat.com.             521     IN      NS      ns3.redhat.com.
redhat.com.             521     IN      NS      ns1.redhat.com.
redhat.com.             521     IN      NS      ns4.redhat.com.
redhat.com.             521     IN      NS      ns2.redhat.com.
```

4. View Short Output Using dig +short

- To view just the ip-address of a web site (i.e the A record), use the short form option as shown below

```
$ dig redhat.com +short

209.132.183.81
```

Cont'd..

- we can also specify a record type that you want to view with the +short option.

```
$ dig redhat.com ns +short
ns2.redhat.com.
ns3.redhat.com.
ns1.redhat.com.
ns4.redhat.com.
```

5. DNS Reverse Look-up Using dig -x

- To perform a DNS reverse look up using the ip-address using dig -x as shown below

```
$ dig -x 209.132.183.81 +short

www.redhat.com.
```

Caching(TTL)

- TTL (Time to Live) is a setting for each DNS record that specifies how long a resolver is supposed to cache (or remember) the DNS query before the query expires and a new one needs to be done.

- The benefits of caching are:

- It is a lot faster to check your local resolver's cache then having to look up a DNS record that isn't already cached.

- This speeds up your Internet experience when visiting a site which we go often (since less time is needed to complete DNS lookups) and also helps lower the load on DNS servers around the world

Zone Files

- Zone files contain Resource Records that describe a domain or sub-domain. Almost any sensible DNS software should be able to read zone files. A zone file will consist of the following types of data:

- Data that indicates the top of the zone and some of its general properties (**a SOA Record**).
- Authoritative data for all nodes or hosts within the zone (**typically A (IPv4) or AAAA (IPv6) Records**).
- Data that describes global information for the zone (including mail **MX Records** and **Name Server NS Records**).

Cont'd..

- In the case of **sub-domain delegation** the name servers responsible for this sub-domain (one or more **NS Records**).
- In the case of **sub-domain delegation** one or more **glue** records that allows a name server to reach the sub-domain (typically one or more **A or AAAA Records**) for the sub-domain name servers.
- The individual **Resource Records** are described and numerous **sample configuration files** are provided and documented.

AXFR vs. IXFR

- **AXFR** is a mechanism for replicating DNS data across DNS servers. If, for example, the **bvb.edu** administrator has two DNS servers, **a.ns.bvb.edu** and **b.ns.bvb.edu**, he can edit the **bvb.edu** data on **a.ns.bvb.edu,** and rely on **AXFR** to pull the same data to **b.ns.bvb.edu**

- **Incremental zone transfer, or IXFR** for short, solves the problem by allowing slave name servers to tell their master servers which version of a zone they currently hold and to request just the changes to the zone between that version and the current one. This can reduce the size and duration of a zone transfer dramatically.

Cont'd..

- When the master name server receives an incremental zone transfer request, it looks for the record of the changes to the zone between the slave's version of the zone and the version the master holds.

- If that record is missing, the master sends a full zone transfer. Otherwise, it sends just the differences between the versions of the zone.

Different Zones: Primary/Secondary/Stub

- **Primary Zone:** The DNS server is the primary source for information about this zone, and it stores the master copy of zone data in a local file.

- **Secondary Zone**:

- The zone at this server must be obtained from another remote DNS server computer that also hosts the zone.

- This DNS server must have network access to the remote DNS server that supplies this server with updated information about the zone.
- A secondary zone is merely a copy of a primary zone that is hosted on another server

Cont'd..

- **Stub:** This DNS server is a source only for information about the **Authoritative Name Servers** for this zone.

- The zone at this server must be obtained from another DNS server that hosts the zone.

- This DNS server must have network access to the remote DNS server to copy the authoritative name server information about the zone.

DNS Forwarding

- DNS forwarding is the process by which particular sets of DNS queries are handled by a designated server, rather than being handled by the initial server contacted by the client.

- All DNS servers that handle address resolution within the network are configured to forward requests for addresses that are outside the network to a dedicated forwarder.

Cont'd…

- **Internal DNS information can be exposed on the open Internet.** It's far better to have a strict separation between internal and external DNS. Exposing internal domains on the **open Internet creates a potential security and privacy vulnerability**.

- Without forwarding, all DNS servers will query external DNS resolvers if they don't have the required addresses cached. This can result in **excessive network traffic**

Open DNS

- It's domain servers for you to use are 208.67.222.222, and 208.67.220.220.
- It has servers in Amsterdam, Chicago, Dallas, London, LA, Miami, New York, Palo Alto, Seattle and Washington.
- OpenDNS is currently serving 35 Billion DNS requests per day.

Pros

- With servers all around the world it allows your DNS to resolve faster, and if one goes down, another will work for you.
- It has phishing protection so you are protected from fake websites.
- You can block websites individually or by one of the 40+ filters.Much more reliable then your ISP.
- If another OpenDNS user visited a website, but now its offline, if you have smart cache enabled it will continue to resolve (assuming it's a dns issue).
- Many high end company's, schools, and governments are using the service daily.

- Has plans (paid) for Enterprises.
- Its free basic service is only lacking faster customer support, and larger website blocking features.
- Websites can be voted by the users with category's so that websites can be blocked if they are 18+, have ad ware, are gaming, or are high network load.
- Stats (Charts, requests, domains, blocked, etc).
- Has guides to help you set it up on your router or computer.
- Has an application to deal with dynamic IP's.
- Hasn't ever completely gone down in its 4 years.

Cons

- Unless you purchase a plan if a url does not resolve you will see a page like Google with ads.
- The website voting can be highly inaccurate.
- Its phishing service Phishtank which is run by its users can take DAYS or even months to flag a url.

Phishing


From:
Amazon <management@mazoncanada.ca> on behalf of
not an Amazon email address
(note the missing A in Amazon)
To: @sheridanc.on.ca
Cc:
Subject: Suspension
amazon.com
Dear Client,
Generic non-personalized greeting
We have sent you this e-mail, because we have strong reason to belive, your account has been used by someone else.In order to prevent any fraudulent activity from occurring we are required to open an investigation into this matter. We've locked your Amazon account, and you have 36 hours to verify it, or we have the right to terminate it.
To confirm your identity with us click the link bellow:
https://www.amazon.com/exec/obidos/sign-in.html
Hovering over the link reveals it points to a non-Amazon site - "http://redirect.kereskedj.com"
Sincerely,
The Amazon Associates Team
© 1996-2013, Amazon.com, Inc. or its affiliates

Google DNS

- It's domain servers are 8.8.8.8, and 8.8.4.4
- Is run on Google's servers.
- Extremely reliable
- Privacy is kept in tact
- Phishing protection
- Handles 100 billion DNS querys a day
- Set and go. No fuss no configuring.

Pros

- It's free
- It's simple
- It's from Google
- It's reliable

Cons

- Business users may find it too simple (i.e. no domain blocking)

Extension mechanisms for DNS (EDN0)

- **Extension mechanisms for DNS (EDNS)** is a specification for expanding the size of several parameters of the Domain Name System (DNS) protocol which had size restrictions that the Internet Engineering community deemed too limited for increasing functionality of the protocol.

- The restrictions in the size of several flags fields, return codes and label types available in the basic DNS protocol prevented the support of some desirable features.

- Moreover, DNS messages carried by UDP were restricted to 512 bytes, not considering the Internet Protocol (IP) and transport layer headers

Cont'd..

- Since no new flags could be added in the DNS header, EDNS adds information to DNS messages in the form of ***pseudo-resource-records*** included in the "**additional data**" section of a DNS message.

Extra info…..

- The pseudo-RR EDNS introduces for this are of type OPT.
- As pseudo-RRs, OPT type RRs never appear in any zone file; they exist only in messages, fabricated by the DNS participants.
- The mechanism is backward compatible, because older DNS responders ignore any RR of the unknown OPT type in a request and a newer DNS responder never includes an OPT in a response unless there was one in the request. The presence of the OPT in the request signifies a newer requester that knows what to do with an OPT in the response.
- The OPT pseudo-record provides space for up to 16 flags and it extends the space for the response code. The overall size of the UDP packet and the version number (at present 0) are contained in the OPT record. A variable length data field allows further information to be registered in future versions of the protocol. The original DNS protocol provided two label types, which are defined by the first two bits in DNS packets (RFC 1035): 00 (standard label) and 11 (compressed label). EDNS introduces the label type 01 as *extended label*. The lower 6 bits of the first byte may be used to define up to 63 new extended labels.

Extra info…..

- An example of an OPT pseudo-record, as displayed by the *Domain Information Groper* (dig) utility tool:: OPT PSEUDOSECTION: ; EDNS: version: 0, flags: do; udp: 4096.

- The result of "EDNS: version: 0" indicates full conformance with EDNS0.The result "flags: do" indicates that "DNSSEC OK" is set.

- EDNS is essential for the implementation of DNS Security Extensions

THANK YOU