Title: Implementation of DNS lookup

Objectives:To demonstrate DNS forward lookup and DNS reverse lookup .

Problem Statement:

Write a program for DNS lookup. Given an IP address input, it should return URL and vice-versa.

Outcomes:

For given an IP address input, it should return URL and vice-versa.

Software Requirements: Python / JAVA
Operating System: Open source (Ubuntu)

Theory:

Domain:

Domain: Domain names are used to identify one or more IP addresses. For example, the domain name microsoft.com represents about a dozen IP addresses. Domain names are used in URLs to identify particular webpages. For example, in the URL http://www.pcwebopedia.com/index.html, the domain name is pcwebopedia.com.

Every domain name has a suffix that indicates which top level domain (TLD) it belongs to. There are only a limited number of such domains. For example:

gov - Government agencies

edu - Educational institutions

org - Organizations (nonprofit)

mil - Military

com - commercial business

net - Network organizations

ca - Canada

th - Thailand

Because the Internet is based on IP addresses, not domain names, every Web server requires a Domain Name System (DNS) server to translate domain names into IP addresses.

DNS lookup:

DNS stands for domain name system, which is the database responsible for storing all of the information pertaining to IP addresses and domain names online. DNS servers are used to carry and transmit this data from one computer to another. All of this data is stored on a network that is backed up by thousands of separate DNS servers and stored on single root DNS servers in the United States, Japan, London and Sweden.

DNS Forward lookup:

The forward lookup, or simple DNS lookup, is the most commonly used approach to DNS. The forward approach to DNS is simply finding out the IP address of a domain. People tend to find it difficult to remember long strings of numbers. Instead, it's easier to remember a domain name that uses words. However, electronic devices use streams of 1's and 0's to communicate. The only way for one computer to communicate with another is by uniquely identification. The method identification used on the Internet is by IP addresses.

Here are the simple steps for DNS resolution:

1. A user enters a domain name into their Internet browser. (www.whatismyip.com)
2. The computer sends the domain name as a DNS request to the user's Internet Service Provider (ISP).
3. The ISP determines if it has the IP address associated with that name;
4. If not, the ISP forwards the request to other providers in an effort to located the DNS record that contains the data.
5. Once the record is found, the IP address of the domain is returned to the user.
6. Now, the user's computer can communicate directly with the server.

DNS Backward lookup:

Reverse DNS lookups for IPv4 addresses use the special domain `in-addr.arpa`. In this domain, an IPv4 address is represented as a concatenated sequence of four decimal numbers, separated by dots, to

which is appended the second level domain suffix `.in-addr.arpa`. The four decimal numbers are obtained by splitting the 32-bit IPv4 address into four octets and converting each octet into a decimal number. These decimal numbers are then concatenated in the order: least significant octet first (leftmost), most significant octet last (rightmost). It is important to note that this is the reverse order to the usual dotted-decimal convention for writing IPv4 addresses in textual form.

For example, to do a reverse lookup of the IP address `8.8.4.4` the PTR record for the domain name `4.4.8.8.in-addr.arpa` would be looked up, and found to point to `google-public-dns-b.google.com`.

If the A record for `google-public-dns-b.google.com` in turn pointed back to `8.8.4.4` then it would be said to be forward-confirmed.

Working of DNS:

When you visit a domain such as *dyn.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 or listening to Internet radio stations like Pandora.

Step 1: Request information

The process begins when you ask your computer to resolve a hostname, such as visiting *http://dyn.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 thirteen root nameservers 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.dyn.***com** — and direct our query to the **Top-Level Domain (TLD) nameservers** for .com. Each TLD, such as .com, .org, and .us, 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.***dyn.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.dyndns.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 *dyn.com* from the authoritative nameservers and stores the record in its local cache. If anyone else requests the host record for *dyn.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

Conclusion: Hence we have implemented DNS lookup.