A

RESEARCH REPORT ON WORKS OF INTERNET

SUBMITTED

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

M.SC (COMPUTER APPLICATION)

BY

MR. VIKAS SAHU

(PERMANENT REG. NO.)

40720601296

VIKAS COLLEGE STUDY CENTRE (MUMBAI, VIKHROLI)

TILAK MAHARASHTRA VIDYAPEETH, PUNE.

(2021 – 2022)

TILAK MAHARASHTRA VIDYAPEETH, PUNE

(DEEMED UNDER SECTION 3 OF UGC ACT 1956 VIDE NOTIFICATION NO. F 9-19/85-U3 DATED 24TH APRIL 1987 BY THE GOVERNMENT OF INDIA)

VIDYAPEETH BHAVAN, GULTEKADI, PUNE- 411 037

CERTIFICATE

THIS IS TO CERTIFY THE RESEARCH REPORT TITLED

“WORKS OF INTERNET” IS A BONAFIDE WORK CARRIED OUT BY

MR. VIKAS SAHU
(PERMANENT REG. NO. 40720601296)

BY STUDENT OF M.Sc. (COMPUTER APPLICATION)

SEMESTER 4TH

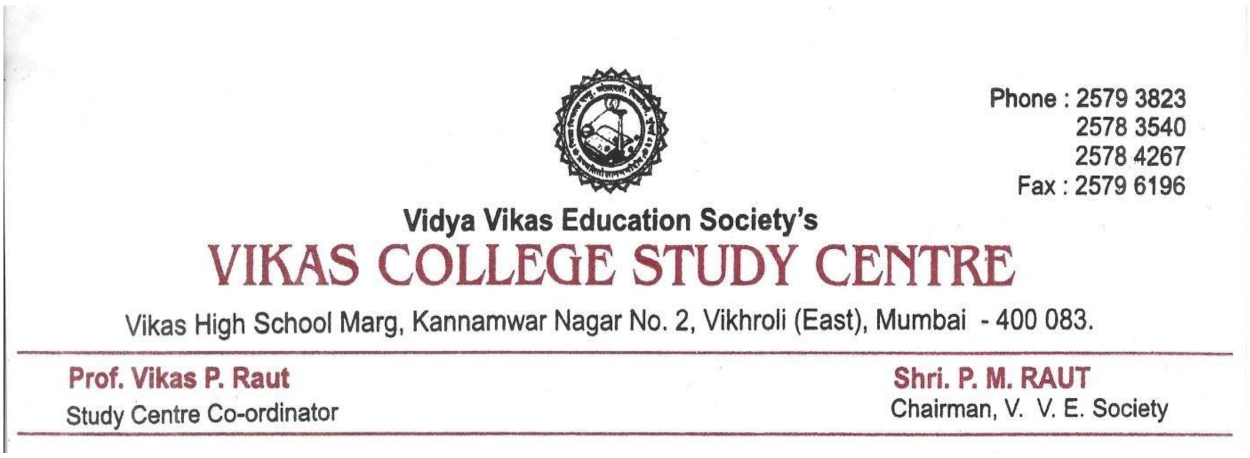
UNDER TILAK MAHARASHTRA VIDYAPEETH IN THE YEAR 2022

EXAMINER INTERNAL EXAMINER EXTERNAL HEAD OF THE DEPARTMENT

DATE:

PLACE:

SEAL



Phone : 2579 3823
2578 3540
2578 4267
Fax : 2579 6196

Vidya Vikas Education Society's

VIKAS COLLEGE STUDY CENTRE

Vikas High School Marg, Kannamwar Nagar No. 2, Vikhroli (East), Mumbai - 400 083.

Prof. Vikas P. Raut
Study Centre Co-ordinator

Shri. P. M. RAUT
Chairman, V. V. E. Society

GUIDE CERTIFICATE

THIS IS TO CERTIFY THAT THE RESEARCH REPORT "WORKS OF INTERNET" HAS BEEN SATISFACTORILY COMPLETED BY

MR. VIKAS SAHU

PERMANENT REG. NO. 40720601296

TOWARD THE PARTIAL FULFILLMENT OF THE

'M.Sc. (COMPUTER APPLICATION),

FOR THE ACADEMIC YEAR (2021-2022) AT VIKAS COLLEGE, VIKHROLI

TILAK MAHARASHTRA VIDYAPEETH, PUNE

(FACULTY OF DISTANCE EDUCATION),

AND IT IS APPROVED.

PROJECT GUIDE EXAMINER HEAD OF DEPT.

MR. VIKAS RAUT

WORKS OF INTERNET

ABSTRACT

This is an attempt to answer the age-old interview question "What Happens when you type google.com into your browser's address box and press Enter?"

INTRODUCTION

Every day you open up your browser and navigate to your favourite websites — whether it be social media, news, or e-commerce sites. You go to this page by typing in a URL or clicking on a link to the page. Have you ever thought about what happens behind the scenes? How does the news get to you when you press enter after typing in the URL? How did the images on this post show up in your browser? How does your Twitter feed and the tweet data show up in your browser securely?

We’ll look at works of internet you type a URL into your browser and press enter. End to end, the process involves the browser, your computer’s operating system, your internet service provider, the server where you host the site, and services running on that server. It’s important to understand where things can go wrong, where to look for performance issues, and ensure you’re offering a secure experience for your users.

WEB BROWSER

A **web browser** is a type of software that allows you to find and view websites on the Internet. Even if you didn't know it, you're using a web browser right now to read this page!

The browser's functionality is to present the web resource you choose, by Requesting it from the server and displaying it in the browser window. The resource is usually an HTML document, but may also be a PDF, Image, or some other type of content. The location of the resource is Specified by the user using a URI (Uniform Resource Identifier).

The way the browser interprets and displays HTML files is specified In the HTML and CSS specifications. These specifications are maintained By the W3C (World Wide Web Consortium) organization, which is the Standards organization for the web.

Browser user interfaces have a lot in common with each other. Among the Common user interface elements are:

- An address bar for inserting a URI
- Back and forward buttons
- Bookmarking options
- Refresh and stop buttons for refreshing or stopping the loading of
- Current documents
- Home button that takes you to your home page

BROWSER HIGH-LEVEL STRUCTURE

The components of the browsers are:

- User interface: The user interface includes the address bar, Back/forward button, bookmarking menu, etc. Every part of the browser Display except the window where you see the requested page.
- Browser engine: The browser engine marshals actions between the UI And the rendering engine.
- Rendering engine: The rendering engine is responsible for displaying Requested content. For example, if the requested content is HTML, the Rendering engine parses HTML and CSS, and displays the parsed content on the screen.
- Networking: The networking handles network calls such as HTTP requests, Using different implementations for different platforms behind a Platform-independent interface.
- UI backend: The UI backend is used for drawing basic widgets like combo Boxes and windows. This backend exposes a generic interface that is not Platform-specific. Underneath it uses operating system user interface methods.
- JavaScript engine: The JavaScript engine is used to parse and Execute JavaScript code.
- Data storage: The data storage is a persistence layer. The browser may Need to save all sorts of data locally, such as cookies. Browsers also Support storage mechanisms such as local Storage, Indexed DB, websql and Filesystem.

List of Top web Browsers present online

Internet is a Jungle with web Browsers acting as passages to find out what lies inside internet Websites.

There are a lot of browsers with awesome features and popularity. But which ones are the Top Web Browsers? Here we have compiled a list of top web browsers which are feature rich, faster, secure and have user popularity.

Please note that all these web browsers are unique and magnificent in their own way. There is no top browser among them but all these have been chosen based on user friendliness and popular reviews.

LIST OF TOP WEB BROWSERS

- Apple Safari.
- Internet Explorer
- Brave.
- Dolphin browser.
- Google Chrome.
- Microsoft Edge.
- Mozilla Firefox.
- Opera.

No matter which web browser you use, you have to learn the basics of browsing the Web.

LIST OF FEATURES (WEB BROWSERS)

- URLs and the address bar
- Links
- Navigation buttons
- Tabbed browsing
- Bookmarks and history
- Downloading files
- Saving images
- Plug-ins

THE "G" KEY IS PRESSED

The following sections explain the physical keyboard actions And the OS interrupts. When you press the key "g" the browser receives the Event and the auto-complete functions kick in Depending on your browser's algorithm and if you are in Private/incognito mode or not various suggestions will be presented to you in the dropdown below the URL bar. Most of these algorithms sort and prioritize results based on search history, bookmarks, cookies, and Popular searches from the internet as a whole. As you are typing "google.com" many blocks of code run and the suggestions will be refined with each keypress. It may even suggest "google.com" before you finish typing It.

THE "ENTER" KEY BOTTOMS OUT

To pick a zero point, let's choose the Enter key on the keyboard hitting the Bottom of its range. At this point, an electrical circuit specific to the enter Key is closed (either directly or capacitively). This allows a small amount of Current to flow into the logic circuitry of the keyboard, which scans the state of each key switch, debounces the electrical noise of the rapid intermittent Closure of the switch, and converts it to a keycode integer, in this case 13. The keyboard controller then encodes the keycode for transport to the computer. This is now almost universally over a Universal Serial Bus (USB) or Bluetooth Connection, but historically has been over PS/2 or ADB connections.

In the case of the USB keyboard:

- The USB circuitry of the keyboard is powered by the 5V supply provided over Pin 1 from the computer's USB host controller.
- The keycode generated is stored by internal keyboard circuitry memory in a Register called "endpoint".
- The host USB controller polls that "endpoint" every ~10ms (minimum value Declared by the keyboard), so it gets the keycode value stored on it.
- This value goes to the USB SIE (Serial Interface Engine) to be converted in One or more USB packets that follow the low-level USB protocol.
- Those packets are sent by a differential electrical signal over D+ and D-Pins (the middle 2) at a maximum speed of 1.5 Mb/s, as an HID (Human Interface Device) device is always declared to be a "low-speed device"(USB 2.0 compliance).

- This serial signal is then decoded at the computer's host USB controller, and Interpreted by the computer's Human Interface Device (HID) universal keyboard device driver. The value of the key is then passed into the operating system's hardware abstraction layer.

IN THE CASE OF VIRTUAL KEYBOARD (AS IN TOUCH SCREEN DEVICES):

- When the user puts their finger on a modern capacitive touch screen, a tiny amount of current gets transferred to the finger. This completes the Circuit through the electrostatic field of the conductive layer and creates a voltage drop at that point on the screen. The ``screen controller`` then raises an interrupt reporting the coordinate of the keypress.
- Then the mobile OS notifies the currently focused application of a press event in one of its GUI elements (which now is the virtual keyboard application Buttons).
- The virtual keyboard can now raise a software interrupt for sending a 'Key pressed' message back to the OS.
- This interrupt notifies the currently focused application of a 'key pressed' Event.

INTERRUPT FIRES [NOT FOR USB KEYBOARDS]

The keyboard sends signals on its interrupt request line (IRQ), which is mapped to an ``interrupt vector`` (integer) by the interrupt controller. The CPU uses the ``Interrupt Descriptor Table`` (IDT) to map the interrupt vectors to Functions (``interrupt handlers``) which are supplied by the kernel. When an Interrupt arrives, the CPU indexes the IDT with the interrupt vector and runs the appropriate handler. Thus, the kernel is entered.

(ON WINDOWS) A ``WM_KEYDOWN`` MESSAGE IS SENT TO THE APP

The HID transport passes the key down event to the ``KBDHID.sys`` driver which Converts the HID usage into a scan code. In this case, the scan code is ``VK_RETURN`` (``0x0d``). The ``KBDHID.sys`` driver interfaces with the ``KBDCLASS.sys`` (keyboard class driver). This driver is responsible for Handling all keyboard and keypad input in a secure manner. It then calls into ``Win32K.sys`` (after potentially passing the message through 3rd party Keyboard filters that are installed). This all happens in kernel mode.

``Win32K.sys`` figures out what window is the active window through the ``getforegroundwindow()`` API. This API provides the window handle of the Browser's address box. The main Windows "message pump" then calls ``sendmessage(hwnd, WM_KEYDOWN, VK_RETURN, lparam)``. ``lparam`` is a bitmask That indicates further information about the keypress: repeat count (0 in this Case), the actual scan code (can be OEM dependent, but generally wouldn't be For ``VK_RETURN``), whether extended keys (e.g. Alt, shift, ctrl) were also Pressed (they weren't), and some other state.

The Windows ``sendmessage`` API is a straightforward function that Adds the message to a queue for the particular window handle (``hwnd``). Later, the main message processing function (called a ``windowproc``) assigned To the ``hwnd`` is called in order to process each message in the queue.

The window (``hwnd``) that is active is actually an edit control and the ``windowproc`` in this case has a message handler for ``WM_KEYDOWN`` messages. This code looks within the 3rd parameter that was passed to ``sendmessage`` (``wparam``) and, because it is ``VK_RETURN`` knows the user has hit the ENTER Key.

(ON OS X) A ``KEYDOWN`` NSEVENT IS SENT TO THE APP

The interrupt signal triggers an interrupt event in the I/O Kit kext keyboard Driver. The driver translates the signal into a key code which is passed to the OS X ``windowserver`` process. Resultantly, the ``windowserver`` dispatches an Event to any appropriate (e.g. Active or listening) applications through their Mach port where it is placed into an event queue. Events can then be read from This queue by threads with sufficient privileges calling the ``mach_ipc_dispatch`` function. This most commonly occurs through, and is Handled by, an ``nsapplication`` main event loop, via an ``nsevent`` of ``nseventtype`` ``keydown``.

PARSE URL

The browser now has the following information contained in the URL (Uniform Resource Locator):

- ``Protocol`` "http"
  Use 'Hyper Text Transfer Protocol'
- ``Resource`` "/"
  Retrieve main (index) page

IS IT A URL OR A SEARCH TERM?

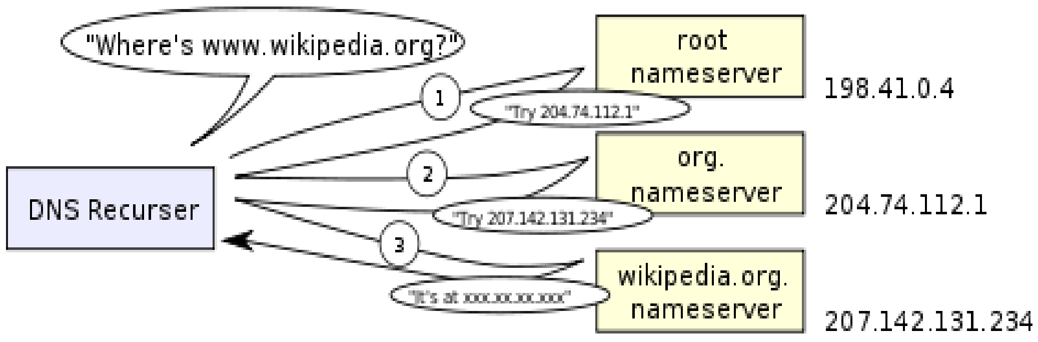
When no protocol or valid domain name is given the browser proceeds to feed The text given in the address box to the browser's default web search engine. In many cases the URL has a special piece of text appended to it to tell the Search engine that it came from a particular browser's URL bar.

DNS LOOKUP

The browser tries to figure out the IP address for the entered domain. The DNS lookup proceeds as follows:

- **Browser cache:** The browser caches DNS records for some time. Interestingly, the OS does not tell the browser the time-to-live for each DNS record, and so the browser caches them for a fixed duration (varies between browsers, 2 – 30 minutes).
- **OS cache:** If the browser cache does not contain the desired record, the browser makes a system call (gethostbyname in Windows). The OS has its own cache.
- **Router cache:** The request continues on to your router, which typically has its own DNS cache.
- **ISP DNS cache:** The next place checked is the cache ISP's DNS server. With a cache, naturally.
- **Recursive search:** Your ISP's DNS server begins a recursive search, from the root nameserver, through the .com top-level nameserver, to Google's nameserver. Normally, the DNS server will have names of the .com nameservers in cache, and so a hit to the root nameserver will not be necessary.

Here is a diagram of what a recursive DNS search looks like:



No worrying thing about DNS is that the entire domain like wikipedia.org or facebook.com seems to map to a single IP address. Fortunately, there are ways of mitigating the bottleneck:

- **Round-robin DNS** is a solution where the DNS lookup returns multiple IP addresses, rather than just one. For example, facebook.com actually maps to four IP addresses.

- **Load-balancer** is the piece of hardware that listens on a particular IP address and forwards the requests to other servers. Major sites will typically use expensive high-performance load balancers.
- **Geographic DNS** improves scalability by mapping a domain name to different IP addresses, depending on the client's geographic location. This is great for hosting static content so that different servers don't have to update shared state.
- **Anycast** is a routing technique where a single IP address maps to multiple physical servers. Unfortunately, anycast does not fit well with TCP and is rarely used in that scenario.

Most of the DNS servers themselves use anycast to achieve high availability and low latency of the DNS lookups. Users of an anycast service (DNS is an excellent example) will always connect to the 'closest' (from a routing protocol perspective) DNS server. This reduces latency, as well as providing a level of load-balancing (assuming that your consumers are evenly distributed around your network).

OPENING OF A SOCKET + TLS HANDSHAKE

- Once the browser receives the IP address of the destination server, it takes that and the given port number from the URL (the HTTP protocol defaults to port 80, and HTTPS to port 443), and makes a call to the system library function named socket and requests a TCP socket stream.
- The client computer sends a clienthello message to the server with its TLS version, list of cipher algorithms and compression methods available.
- The server replies with a serverhello message to the client with the TLS version, selected cipher, selected compression methods and the server's public certificate signed by a CA (Certificate Authority). The certificate contains a public key that will be used by the client to encrypt the rest of the handshake until a symmetric key can be agreed upon.
- The client verifies the server digital certificate against its list of trusted cas. If trust can be established based on the CA, the client generates a string of pseudo-random bytes and encrypts this with the server's public key. These random bytes can be used to determine the symmetric key.

- The server decrypts the random bytes using its private key and uses these bytes to generate its own copy of the symmetric master key.
- The client sends a Finished message to the server, encrypting a hash of the transmission up to this point with the symmetric key.
- The server generates its own hash, and then decrypts the client-sent hash to verify that it matches. If it does, it sends its own Finished message to the client, also encrypted with the symmetric key.
- From now on the TLS session transmits the application (HTTP) data encrypted with the agreed symmetric key.

WRAP UP

Now you know how the Internet works. But how long will it stay this way? The version of IP currently used on the Internet (version 4) only allows 2^{32} addresses. Eventually there won't be any free IP addresses left. Surprised? Don't worry. IP version 6 is being tested right now on a research backbone by a consortium of research institutions and corporations. And after that? Who knows? The Internet has come a long way since its inception as a Défense Department research project. No one really knows what the Internet will become. One thing is sure, however. The Internet will unite the world like no other mechanism ever has. The Information Age is in full stride and I am glad to be a part of it.

VIKAS SAHU

RESOURCES

Below are some interesting links associated with some of the topics discussed. (I hope they all still work. All open in new window.)

Http://www.ietf.org is the home page of the Internet Engineering Task Force. This body is greatly responsible for the development of Internet protocols and the like.

Http://www.internic.org is the organization responsible for administering domain names.

Http://www.faqs.org/rfcs/rfcsearch.html is an excellent RFC search engine useful for finding any RFC.

Http://navigators.com/isp.html is Russ Haynal's ISP Page. This is a great site with links to most nsps and their backbone infrastructure maps.

Https://www.nominus.com/en/dm/blog/generic-domains/a-timeline-of-important-events-in-internet-history, is a Timeline of Important Events in Internet History, sent by Lily.

Http://www.investintech.com/content/historyinternet/ is A Brief Guide to the History of the Internet.

BIBLIOGRAPHY

The following books are excellent resources and helped greatly in the writing of this paper. I believe Stevens' book is the best TCP/IP reference ever and can be considered the bible of the Internet. Sheldon's book covers a much wider scope and contains a vast amount of networking information.

- TCP/IP Illustrated, Volume 1, The Protocols.
  W. Richard Stevens.
  Addison-Wesley, Reading, Massachusetts. 1994.
- Encyclopaedia of Networking.
  Tom Sheldon.
  Osbourne McGraw-Hill, New York. 1998.

Although not used for writing this paper, here are some other good books on the topics of the Internet and networking:

- Firewalls and Internet Security; Repelling the Wiley Hacker.
  William R. Cheswick, Steven M. Bellovin.
  Addison-Wesley, Reading, Massachusetts. 1994.
- Data Communications, Computer Networks and Open Systems. Fourth Edition.
  Fred Halsall.
  Addison-Wesley, Harlow, England. 1996.
- Telecommunications: Protocols and Design.
  John D. Spragins with Joseph L. Hammond and Krzysztof Pawlikowski.
  Addison-Wesley, Reading, Massachusetts. 1992.