JSS MAHAVIDYAPEETHA

JSS Science and Technology University

JSS Technical Institutions Campus, Mysore – 570006

"IMPLEMENTATION OF HTTP PROTOCOL USING CISCO PACKET TRACER"

A technical project report submitted in partial fulfilment of the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE & ENGINEERING

BY,

ADITYA VINOD KOPPIKAR	01JST21IS005
CHINMAY ANKOLEKAR	01JST21IS012
NAGESH BHAVI	01JST21IS029
NANDAN K N	01JST21IS030

Under the Guidance of

Sindhu G

Assistant Professor

Department of Information Science & Engineering

JSS STU, Mysore

INTRODUCTION TO PROTOCOLS IN DATA COMMUNICATION

In data communication, protocols play a crucial role in enabling the reliable and efficient exchange of information between systems. A protocol can be defined as a set of rules and conventions that govern the format, timing, sequencing, and error control of data transmission.

Protocols establish a common language and standardized procedures that ensure seamless communication and interoperability between different entities, such as computers, networks, and devices. They define how data is structured, transmitted, received, and interpreted at both the sender and receiver ends.

Protocol Stack

A protocol stack, also known as a protocol suite or network stack, refers to a collection of protocols that are designed to work together to support communication between systems.
Examples of well-known protocol stacks include TCP/IP, OSI (Open Systems Interconnection), and Bluetooth protocol stacks.

Layered Architecture

Many protocols follow a layered architecture, where different protocols are organized into distinct layers based on their functionalities. This layered approach allows for modular design, ease of implementation, and abstraction of complexities. Each layer provides specific services and interacts with adjacent layers through well-defined interfaces.

OSI Model

The OSI (Open Systems Interconnection) model is a conceptual framework that defines a standardized way of understanding and organizing communication protocols. It consists of seven layers, namely Physical, Data Link, Network, Transport, Session, Presentation, and Application layers. Each layer has its own specific responsibilities and protocols associated with it.

TCP/IP

is the most widely used protocol suite for data communication over the internet. It comprises several protocols, including IP (Internet Protocol), TCP (Transmission Control Protocol), UDP (User Datagram Protocol), and others. TCP/IP provides end-to-end routing.

TYPES OF PROTOCOL IN DATA COMMUNICATION

There are various types of protocols in data communication, each serving different purposes and operating at different layers of the protocol stack. Here are some common types of protocols

1. **Transport Layer Protocols**
    - Transmission Control Protocol (TCP): Provides reliable, connection-oriented data delivery with error detection, flow control, and congestion control.
    - User Datagram Protocol (UDP): Offers connectionless, unreliable data delivery without error recovery or flow control.

2. **Network Layer Protocols**
    - Internet Protocol (IP): Facilitates the routing and delivery of packets across networks.
    - Internet Control Message Protocol (ICMP): Handles error reporting, diagnostics, and control messages for IP.

3. **Link Layer Protocols**
    - Ethernet: Defines the physical and data link layer protocols for wired local area networks (LANs).
    - Point-to-Point Protocol (PPP): Establishes a direct connection between two nodes over various physical media.

4. **Application Layer Protocols**
    - Hypertext Transfer Protocol (HTTP): Used for communication between web browsers and servers.
    - Simple Mail Transfer Protocol (SMTP): Enables email transmission between mail servers.
    - File Transfer Protocol (FTP): Supports file transfer between client and server systems.
    - Domain Name System (DNS): Resolves domain names to IP addresses.

HYPER TEXT TRANSFER PROTOCOL

Introduction

The Hypertext Transfer Protocol (HTTP) is an application-layer protocol used for communication on the World Wide Web. It enables the exchange of hypertext (text with embedded links) between clients (such as web browsers) and servers. HTTP operates on top of the TCP/IP protocol suite and follows a client-server model.

At its core, HTTP is a stateless protocol, meaning that each request-response cycle is independent, and the server does not retain any information about past requests. This statelessness allows for scalable and efficient communication between clients and servers. To maintain stateful behaviour, mechanisms like cookies or session management are employed.

HTTP operates on top of the TCP/IP protocol suite, utilizing the reliable transmission capabilities of TCP for transferring data over the internet. It uses a simple and flexible requestresponse model. Clients initiate HTTP requests by specifying a method (such as GET, POST, PUT, or DELETE), a Uniform Resource Locator (URL) to identify the desired resource, and optional headers for additional information. Servers respond with an HTTP response that includes a status code indicating the outcome of the request, headers with metadata, and the requested data.

Over the years, different versions of HTTP have been introduced, with HTTP/1.1 being the most widely used. However, the emergence of HTTP/2 and later HTTP/3 (based on the QUIC protocol) brings improvements in terms of performance, multiplexing, and security.

FEATURES

Client-Server Communication: HTTP involves communication between a client and a server. The client initiates an HTTP request, and the server responds with an HTTP response containing the requested data.

Request-Response Model:

HTTP follows a request-response model. The client sends an HTTP request to the server, specifying the desired action (such as retrieving a web page) and providing any necessary parameters. The server processes the request and sends back an HTTP response containing the requested data or an error message.

HTTP Methods:

HTTP defines various methods (also known as verbs) that indicate the desired action to be performed on a resource. Common methods include:

GET: Retrieves a resource from the server.

POST: Submits data to be processed by the server.

PUT: Stores or replaces a resource at a specific URL.

DELETE: Removes a resource from the server.

And more.

URLs: Uniform Resource Locators (URLs) are used to identify resources on the web. A URL typically consists of a protocol identifier (such as "http://"), followed by the domain name or IP address of the server, and the path to the specific resource.

Headers: HTTP requests and responses contain headers that provide additional information about the request or response. Headers include details such as the content type, cache control, cookies, authentication information, and more.

Status Codes: HTTP responses include status codes that indicate the outcome of a request. Some common status codes include:

200 OK: The request was successful.

404 Not Found: The requested resource could not be found.

500 Internal Server Error: An unexpected error occurred on the server.

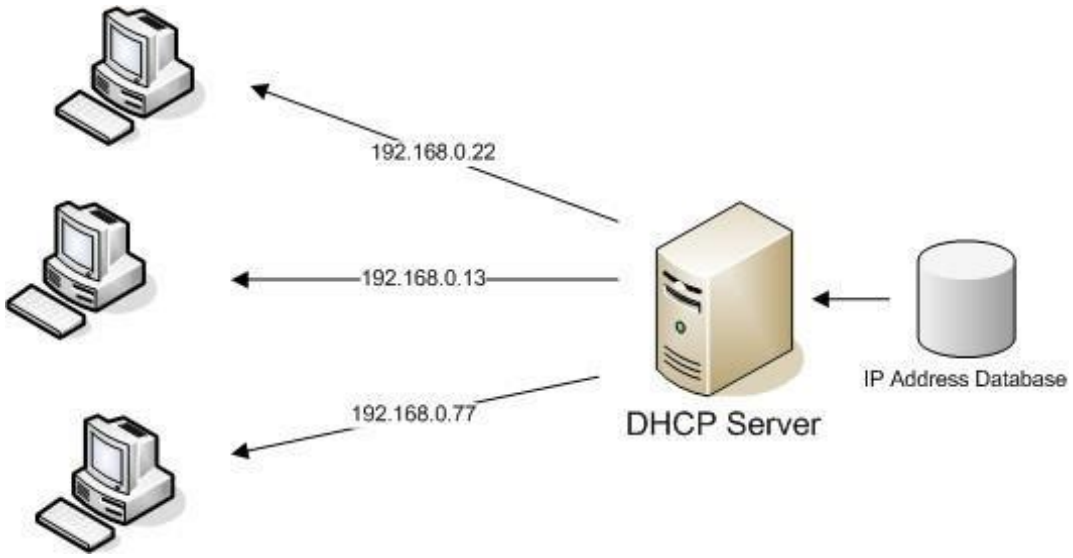
Statelessness: HTTP is a stateless protocol, meaning that each request-response cycle is independent. The server does not retain information about past requests from the same client. To maintain stateful behaviour, mechanisms like cookies or session management are often used.

HTTP is a foundational protocol of the web, enabling the retrieval and delivery of web pages, images, documents, and other resources. It provides a standardized framework for client-server communication, allowing web browsers to interact with web servers and retrieve content from across the internet

DYNAMIC HOST CONFIGURATION PROTOCOL

Dynamic Host Configuration Protocol (DHCP) is a network management protocol used to dynamically assign an IP address to nay device, or node, on a network so they can communicate using IP (Internet Protocol). DHCP automates and centrally manages these configurations.

There is no need to manually assign IP addresses to new devices. Therefore, there is no requirement for any user configuration to connect to a DHCP based network.



DHCP does the following ○ DHCP manages the provision of all the nodes or devices added or dropped from the network. ○ DHCP maintains the unique IP address of the host using a DHCP server. ○ It sends a request to the DHCP server whenever a client/node/device, which is configured to work with DHCP, connects to a network. The server acknowledges by providing an IP address to the client/node/device. ○ DHCP is also used to configure the proper subnet mask, default gateway and DNS server information on the node or device.

Working of DHCP

DHCP runs at the application layer of the TCP/IP protocol stack to dynamically assign IP addresses to DHCP clients/nodes and to allocate TCP/IP configuration information to the DHCP clients. Information includes subnet mask information, default gateway, IP addresses and domain name system addresses.

DHCP is based on client-server protocol in which servers manage a pool of unique IP addresses, as well as information about client configuration parameters, and assign addresses out of those address pools.

DHCP is based on client-server protocol in which servers manage a pool of unique IP addresses, as well as information about client configuration parameters, and assign addresses out of those address pools.

The DHCP lease process works as follows

- First of all, a client (network device) must be connected to the internet.
- DHCP clients request an IP address. Typically, client broadcasts a query for this information.
- DHCP server responds to the client request by providing IP server address and other configuration information. This configuration information also includes time period, called a lease, for which the allocation is valid.
- When refreshing an assignment, a DHCP clients request the same parameters, but the DHCP server may assign a new IP address. This is based on the policies set by the administrator.

Components of DHCP

- **DHCP Server:** DHCP server is a networked device running the DCHP service that holds IP addresses and related configuration information. This is typically a server or a router but could be anything that acts as a host, such as an SD-WAN appliance.
- **DHCP client:** DHCP client is the endpoint that receives configuration information from a DHCP server. This can be any device like computer, laptop, IoT endpoint or anything else that requires connectivity to the network. Most of the devices are configured to receive DHCP information by default.
- **IP address pool:** IP address pool is the range of addresses that are available to DHCP clients. IP addresses are typically handed out sequentially from lowest to the highest.

  Subnet: Subnet is the partitioned segments of the IP networks. Subnet is used to keep networks manageable.
- **Lease:** Lease is the length of time for which a DHCP client holds the IP address information. When a lease expires, the client has to renew it.
- **DHCP relay:** A host or router that listens for client messages being broadcast on that network and then forwards them to a configured server. The server then sends responses back to the relay agent that passes them along to the client.

Benefits of DHCP

Centralized administration of IP configuration: DHCP IP configuration information can be stored in a single location and enables that administrator to centrally manage all IP address configuration information.

Dynamic host configuration: DHCP automates the host configuration process and eliminates the need to manually configure individual host. When TCP/IP (Transmission control protocol/Internet protocol) is first deployed or when IP infrastructure changes are required.

Seamless IP host configuration: The use of DHCP ensures that DHCP clients get accurate and timely IP configuration IP configuration parameter such as IP address, subnet mask, default gateway, IP address of DND server and so on without user intervention.

Flexibility and scalability: Using DHCP gives the administrator increased flexibility, allowing the administrator to move easily change IP configuration when the infrastructure changes.

DNS - Domain Name System

An application layer protocol defines how the application processes running on different systems, pass the messages to each other. • DNS stands for Domain Name System.

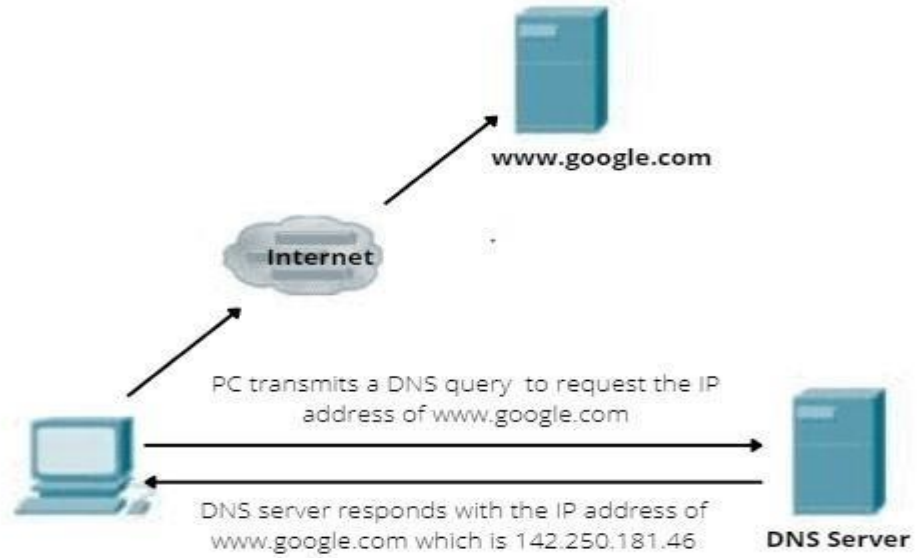
- DNS is a directory service that provides a mapping between the name of a host on the network and its numerical address.
- DNS is required for the functioning of the internet.
- Each node in a tree has a domain name, and a full domain name is a sequence of symbols specified by dots.
- DNS is a service that translates the domain name into IP addresses. This allows the users of networks to utilize user-friendly names when looking for other hosts instead of remembering the IP addresses.

DNS is a TCP/IP protocol used on different platforms. The domain name space is divided into three different sections: generic domains, country domains, and inverse domain.

Generic Domain: It defines the registered hosts according to their generic behavior. It uses three-character labels, and these labels describe the organization type. Examples: govGovernment institutions, edu-Educational institutions, com-Commercial Organization, orgNonprofit Organizations.

Country Domain: The format of country domain is same as a generic domain, but it uses twocharacter country abbreviations (e.g., us for the United States) in place of three-character organizational abbreviations.

Inverse domain: The inverse domain is used for mapping an address to a name.



Working of DNS

- DNS is a client/server network communication protocol. DNS clients send requests to the. server while DNS servers send responses to the client.
- Client requests contain a name which is converted into an IP address known as a forward DNS lookup while requests containing an IP address which is converted into a name known as reverse DNS lookups.
- DNS implements a distributed database to store the name of all the hosts available on the internet.
- If a client like a web browser sends a request containing a hostname, then a piece of software such as DNS resolver sends a request to the DNS server to obtain the IP address of a hostname. If DNS server does not contain the IP address associated with a hostname, then it forwards the request to another DNS server. If IP address has arrived at the resolver, which in turn completes the request over the internet protocol.

IMPLEMENTATION OF HTTP PROTOCOL IN CISCO PACKET TRACER

Tools used:

Software used: cisco packet tracer

Operating System: Ensure that the chosen programming language and tools are compatible with your operating system (e.g., Windows, mac-OS, Linux).

Internet Connectivity: While not essential, internet access can be useful for searching online resources, documentation, and forums related to the programming language and its ecosystem.

By fulfilling these software requirements, you will have the necessary tools and environment to successfully implement the hypertext transfer protocol.

The http protocol can be implemented in cisco packet tracer as shown below

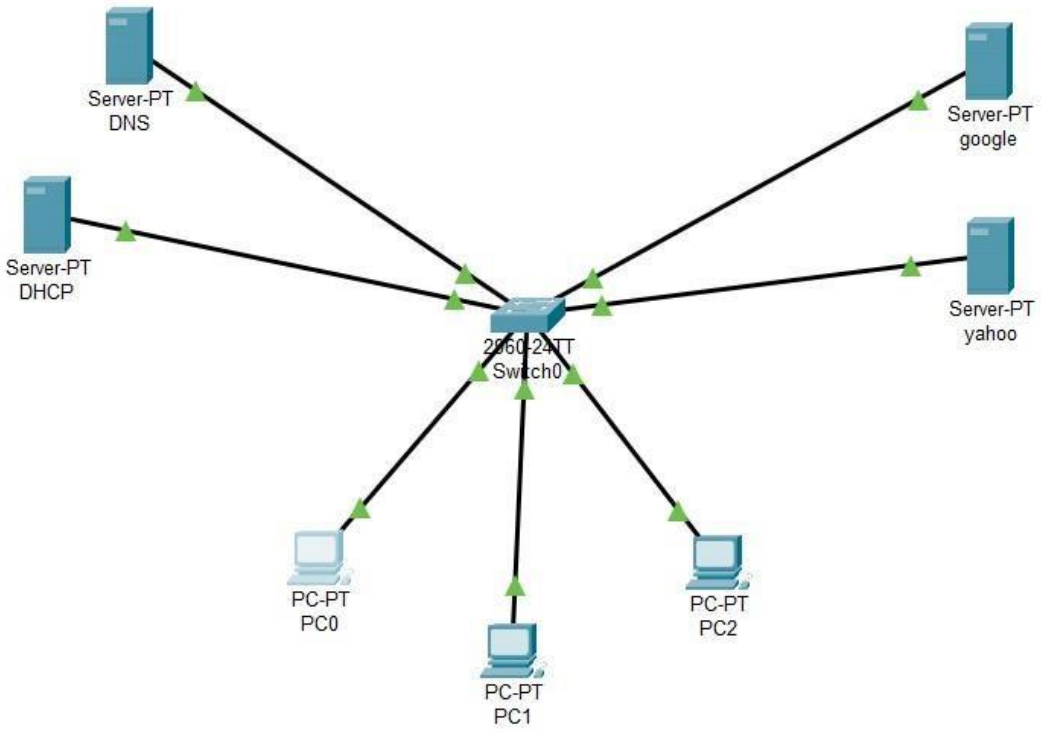


Fig: HTTP protocol implementation

Working:

Devices used:

1. Servers:
   In this mini project we have used 4 servers out of those two are web servers for yahoo and google and the others are to make use of services like DNS and DHCP.

   The google and yahoo servers help in storing web pages.
2. PC:
   We have used 3 personal computers.
3. Networking devices:
   We have used a switch to connect the servers and pcs together, they are all in the same network.

   Firstly, we assign IP addresses to the web servers, in this context we have assigned google with IP address of 192.168.1.1 and yahoo with 192.168.1.2, they are statically assigned. This can be seen in the picture below

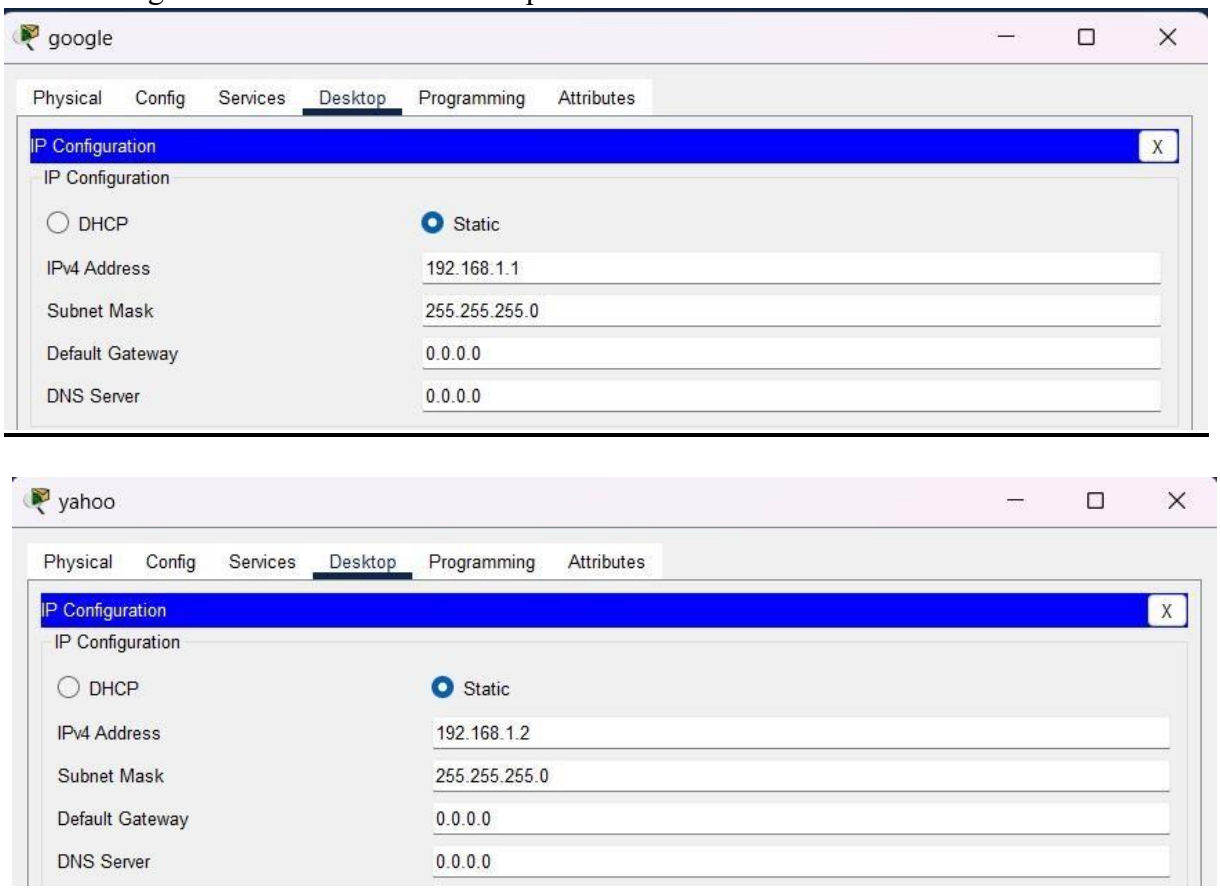


Fig . IP configurations of web servers

Next, to configure the IP addresses of the pcs, we make use of the DHCP servers which dynamically allocate the IP addresses. We have to provide the start IP addresses in the DHCP

servers to be allocated to the host computers, we have chosen the start IP address as 192.168.1.10. we turn on the IP configuration to DHCP in all the host computers.

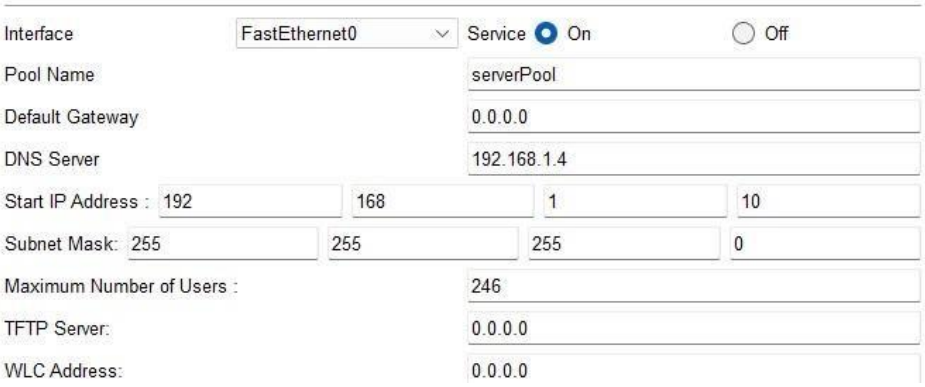


Fig . DHCP server configuration

Now we can see that the hosts have been assigned the IP addresses dynamically, starting from 192.168.1.10.

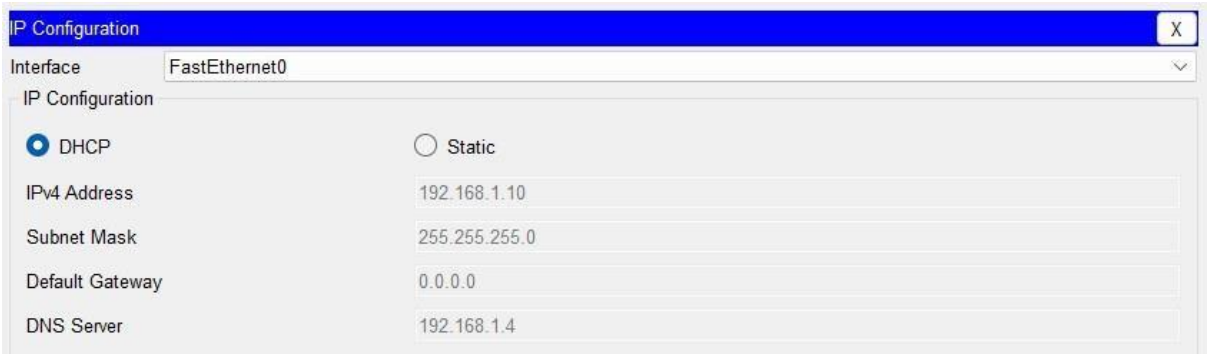


Fig . IP configurations of host computers

The first pc has been assigned with the IP address of 192.168.1.10 which cant be changed as it has been assigned dynamically.

Now, we need to create HTML web pages for each of the domains www.google.com which we store in the google server and www.yahoo.com which we store in the yahoo server. It can be seen in the picture below.

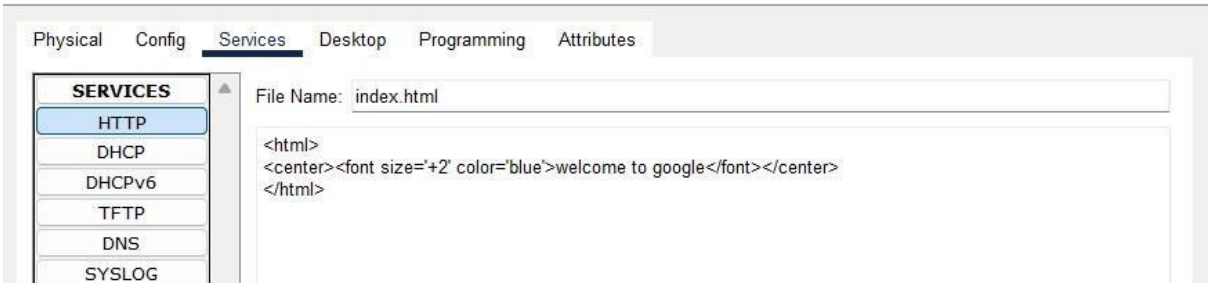


Fig . Web pages stored in the web servers

Next, we need to map the domain name with their respective IP addresses in the DNS servers.

This can be done in the DNS service section of the DNS server where we map the domains with their respective IP addresses. This is important because the user cannot remember all the IP addresses of all the webpages. This can be done as shown below.

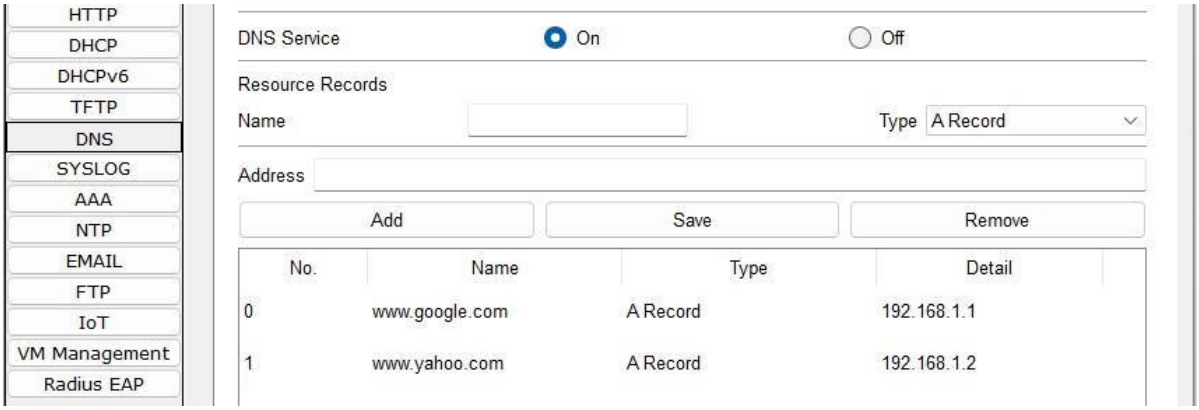


Fig . Mapping of domain names with their IP addresses(DNS)

This makes our configuration complete, now in order to fetch the web pages, we need to open the web browser of one of our host computers and type the respective domain name in the browser, as shown below.

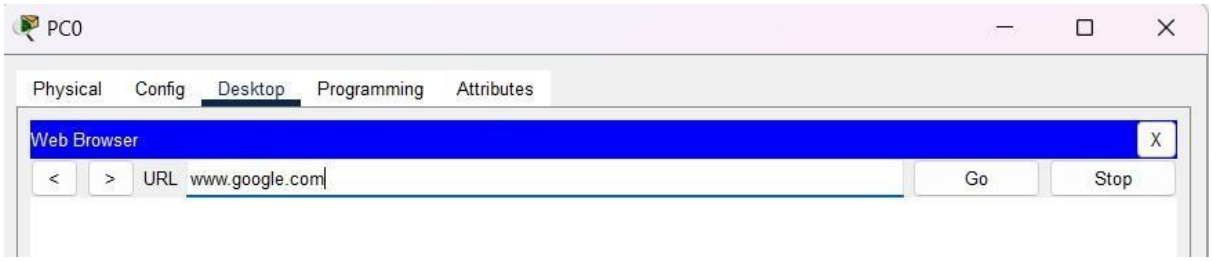


Fig . Requesting web page through domain name in web browsers

Firstly, the user request is sent to the DNS server where the domain name is mapped to the respective IP address and the corresponding IP address is sent back to the requesting host, the host sends the IP address to the respective web servers and web servers in return fetch the webpage and returns to the requesting host, which appears as graphical interface in the web browser. The simple HTML web page is displayed as shown below.

Fig . Fetching webpages from servers and loading them on local web browsers

Conclusion

T0he mini project of implementing the HTTP protocol has been a valuable learning experience. Throughout this project, I gained a deep understanding of the fundamental principles and functionalities of the HTTP protocol. I successfully implemented the basic features of HTTP, including establishing connections, sending requests, and receiving responses. I also became familiar with the different HTTP methods, status codes, and header fields, allowing me to effectively communicate with web servers. This project has not only enhanced my technical

skills but has also provided me with insights into the inner workings of the internet and how web applications interact. Moving forward, I can utilize this knowledge to develop more sophisticated applications and contribute to the ever-evolving field of web development. Overall, this mini project has been a significant stepping stone in my journey to becoming a proficient developer.