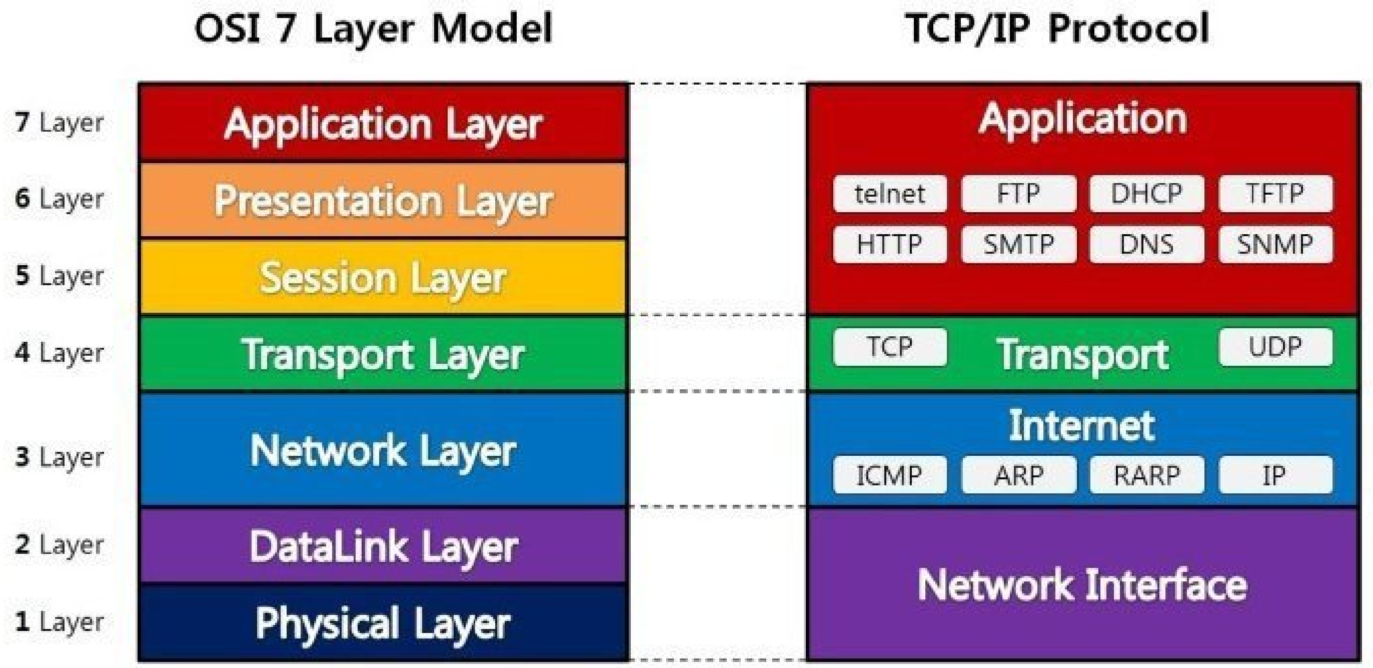
Computer Network

rohyunsang · 약 2시간 전

tcp protocol :
is a protocol used in computer networking to provide reliable, ordered, and error-checked delivery of data between applications running on hosts. It is a connection-oriented protocol

osi 7 layer :

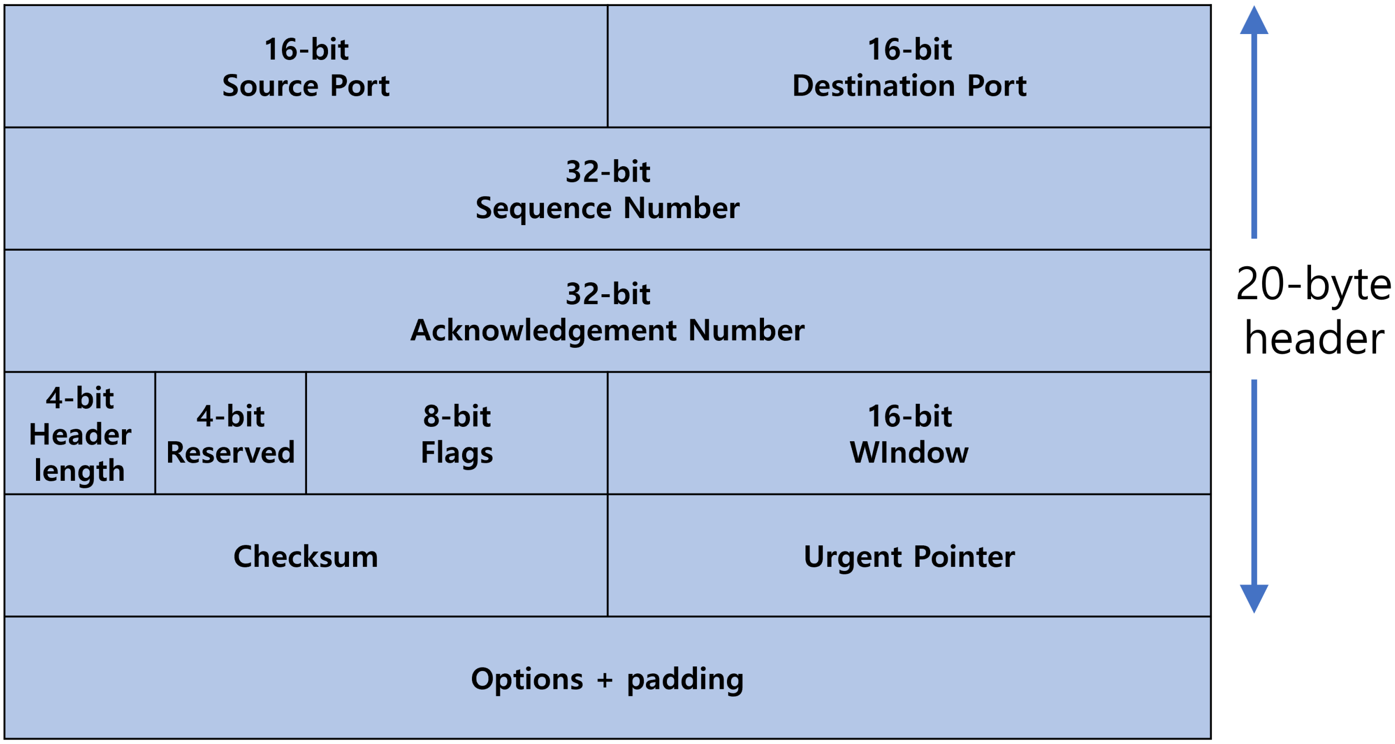


Physical Layer: This layer deals with the physical transmission of data, including the transmission medium, connectors, and electrical signals.

Data Link Layer: This layer provides error-free transmission of data over a physical link.
Network Layer: This layer is responsible for routing data between different networks.
Transport Layer: This layer provides reliable end-to-end data delivery between applications running on different hosts. It ensures that data is delivered in the correct order, without errors or duplication.
Session Layer: This layer establishes, manages, and terminates sessions between applications on different hosts.
Presentation Layer: This layer deals with the format and presentation of data to applications.
Application Layer: This layer provides access to network services for end-user applications. It includes protocols such as HTTP, FTP, SMTP, and Telnet,

internet :
Network's Network

packet :



<table>
<tr><td colspan="3">16-bit
Source Port</td><td>16-bit
Destination Port</td></tr>
<tr><td colspan="4">32-bit
Sequence Number</td></tr>
<tr><td colspan="4">32-bit
Acknowledgement Number</td></tr>
<tr><td>4-bit
Header
length</td><td>4-bit
Reserved</td><td>8-bit
Flags</td><td>16-bit
WIndow</td></tr>
<tr><td colspan="3">Checksum</td><td>Urgent Pointer</td></tr>
<tr><td colspan="4">Options + padding</td></tr>
</table>

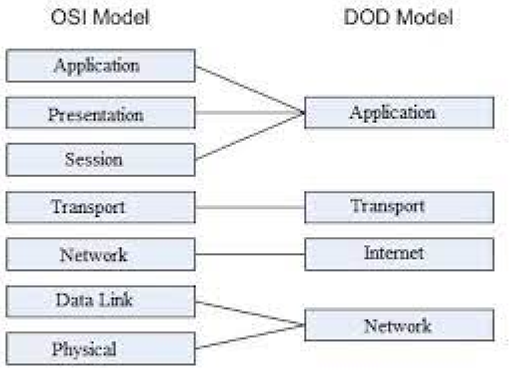
In computer networking, a packet is a unit of data that is transmitted between devices over a network. It consists of a header and a payload. The header contains information such as the source and destination addresses, while the payload contains the actual data being transmitted.

Application :
In the context of computer networking, an application refers to a software program or set of programs designed to perform a specific function or set of functions

Protocol :
In computer networking, a protocol refers to a set of rules and procedures governing communication between devices over a network.

A protocol specifies how data is transmitted, received, and processed between devices, as well as how errors are detected and corrected, and how devices are identified and authenticated.

osi 7 layer vs dod model :



there are conceptual frameworks understanding and designing computer networks.

The OSI model includes the following layers:

Physical Layer
Data Link Layer
Network Layer
Transport Layer
Session Layer
Presentation Layer
Application Layer

The DOD model includes the following layers:

Network Access Layer
Internet Layer

Transport Layer
Application Layer

protocol layer :
In computer networking, a protocol layer refers to a specific level of communication within a protocol stack. A protocol stack is a set of protocols organized in a hierarchical manner, with each protocol layer responsible for a specific aspect of network communication.

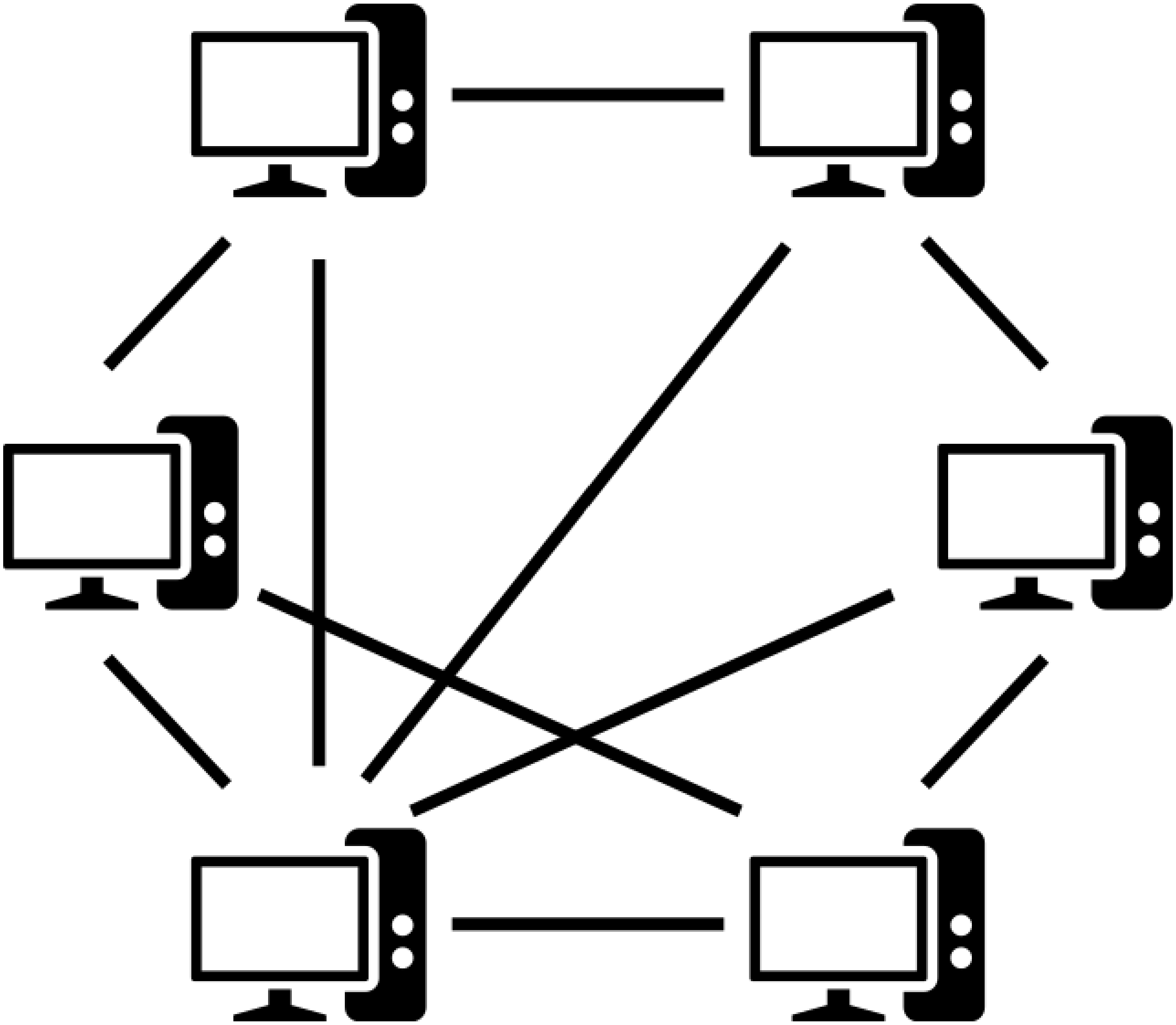
internet protocol stack :
The Internet Protocol (IP) stack, also known as the TCP/IP protocol stack, is a set of protocols used to establish communication between devices over the Internet or other networks.

The TCP/IP protocol stack consists of four layers:

1. Network Access Layer
2. Internet Layer
3. Transport Layer
4. Application Layer

application layer :
you know ? application layer is OSI top layer
application architectures :
The application architecture refers to the design of an application system and how different components of the system interact with each other. It involves decisions about the choice of programming languages, the use of libraries and frameworks, the design of the user interface, and the implementation of various services and functionality.

peer to peer architectures :



each other computer acting client and server

client server architectures :
It enables the sharing of data and services such as email and file transfer

socket :
We want to access it at the user-mode application level, but how do we do this? We abstracting TCP as a file-type interface.

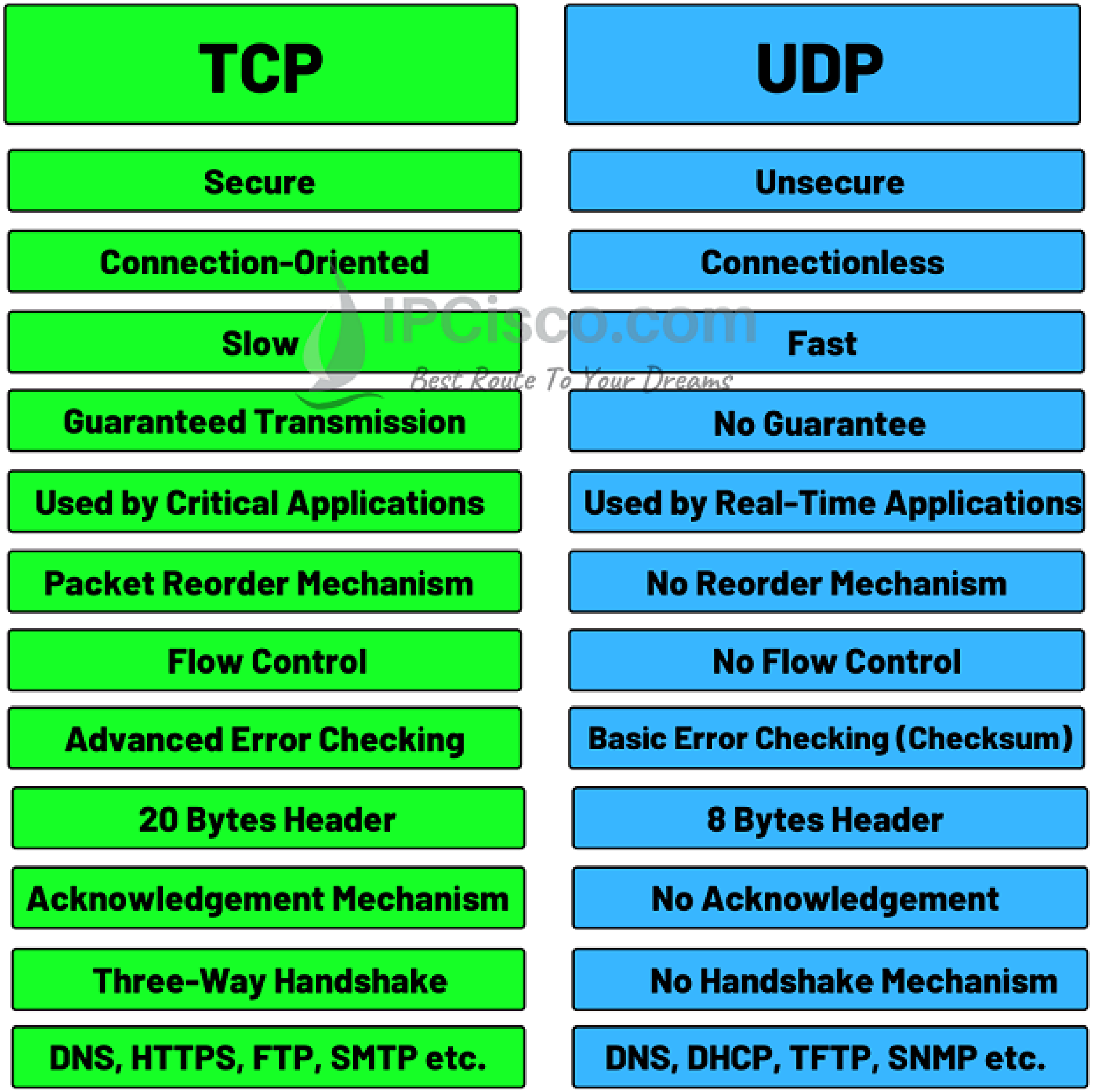
socket structure :
used to define a communication endpoint
socket = address + IP + Port Number + Protocol

Http / Tcp protocol :
this two guys are network protocol use web communication

Http is based on a request response model

Tcp is reliable

Tcp vs Udp :



TCP	UDP
Secure	Unsecure
Connection-Oriented	Connectionless
Slow	Fast
Guaranteed Transmission	No Guarantee
Used by Critical Applications	Used by Real-Time Applications
Packet Reorder Mechanism	No Reorder Mechanism
Flow Control	No Flow Control
Advanced Error Checking	Basic Error Checking (Checksum)
20 Bytes Header	8 Bytes Header
Acknowledgement Mechanism	No Acknowledgement
Three-Way Handshake	No Handshake Mechanism
DNS, HTTPS, FTP, SMTP etc.	DNS, DHCP, TFTP, SNMP etc.

Udp is fast but not reliable

Tcp is slow but provide reliable

So, we are going to pick one, considering our system.

Assign
1.
HTML, HTML5, CSS, Javascript
Jsp Ruby,, etc

HTML5 is the latest version of HTML and includes new features such as support for video and audio playback, new form controls, and improved semantic elements.

CSS (Cascading Style Sheets) is a style sheet language used to describe the presentation and layout of web pages, including the colors, fonts, and spacing.

JavaScript is a scripting language used to create interactive and dynamic web pages, such as adding animations, responding to user input, and manipulating the content of a web page in real-time.

JSP (JavaServer Pages) is a technology used to create dynamic web pages using Java programming language. It allows embedding Java code into HTML pages, which enables the creation of web applications with server-side functionality.

Ruby is a programming language often used for web development, especially with the Ruby on Rails framework. It is known for its readability and conciseness and is often used to build dynamic and scalable web applications.

Node.js is a JavaScript runtime environment built on Chrome's V8 JavaScript engine. It allows developers to run JavaScript code on the server-side, which enables building scalable and efficient web applications.

React is a JavaScript library used for building user interfaces for web applications. It allows developers to create reusable UI components and provides a fast and efficient way to update the user interface in response to changes in the application state.

2. C/S architecture P2P's Pros and Cons

lets make long sentense upper one =>
Client/Server architecture Peer to Peer Adventages and DisAdventages

Pros :
Provides a decentralized system that is more resilient and fault-tolerant
Offers better scalability and performance since there is no central server bottleneck
Enables more efficient use of network resources

Cons :

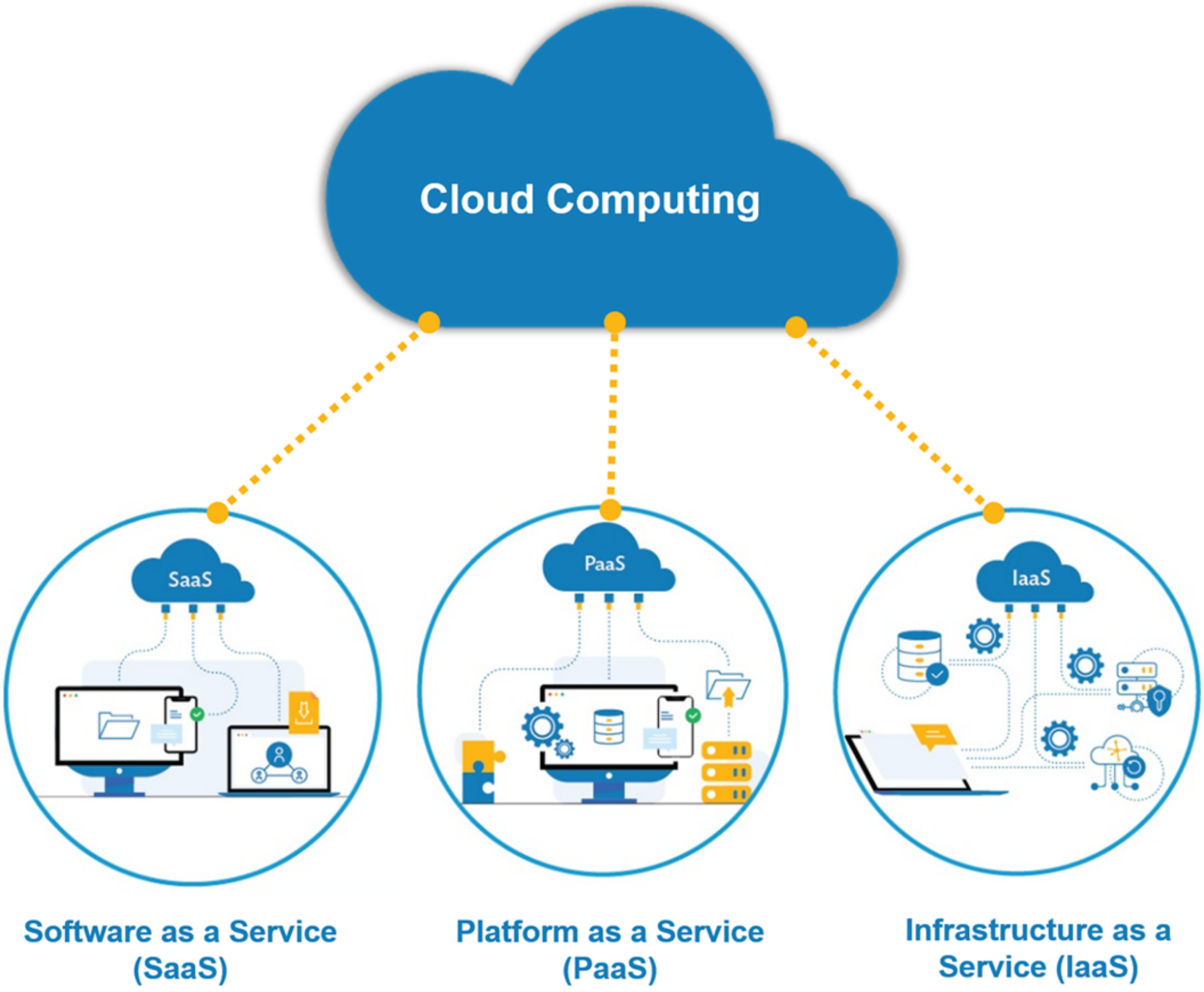
Can be more difficult to manage and maintain due to the lack of central control

May be less secure due to the lack of centralized authentication and access control

May suffer from slower data transfer speeds due to the distributed nature of the network

p2p responds quickly, so it is used a lot in fighting games, but there is a problem with not being able to catch the hacks

3. cloud computing


Cloud Computing
SaaS
PaaS
IaaS
Software as a Service (SaaS)
Platform as a Service (PaaS)
Infrastructure as a Service (IaaS)

relies on remote servers hosted on the Internet to store, manage, and process data

IaaS : Infrastruct as a Service :

provides virtualized computing resources such as servers, storage, and network resources

Customers can use these resources using by make Customers Service

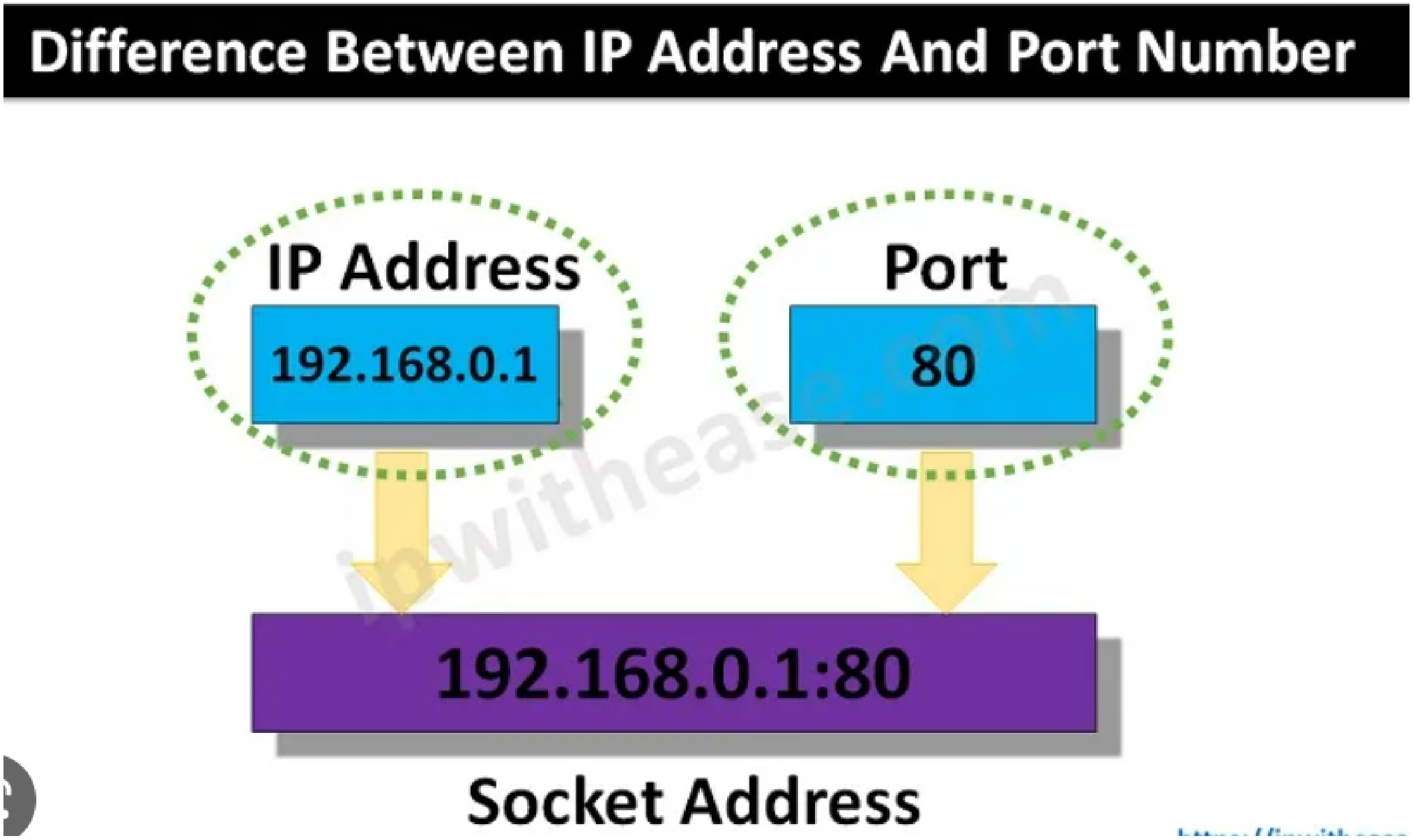
PaaS : Platform as a Service :
Provide manages the infrastructure and customers focus on the application.

SaaS : Software as a Service :
provides ready-to-use applications and services

4. IPC :
   It is Inter Process Communication
   using socket and Provide supports "access control" for shared resources through communication between processors.

URL structure :
URL = Scheme + Domain + Port + Path + Query + Fragment
=> scheme://domain:port/path?query#fragment

Web Server Port Number :

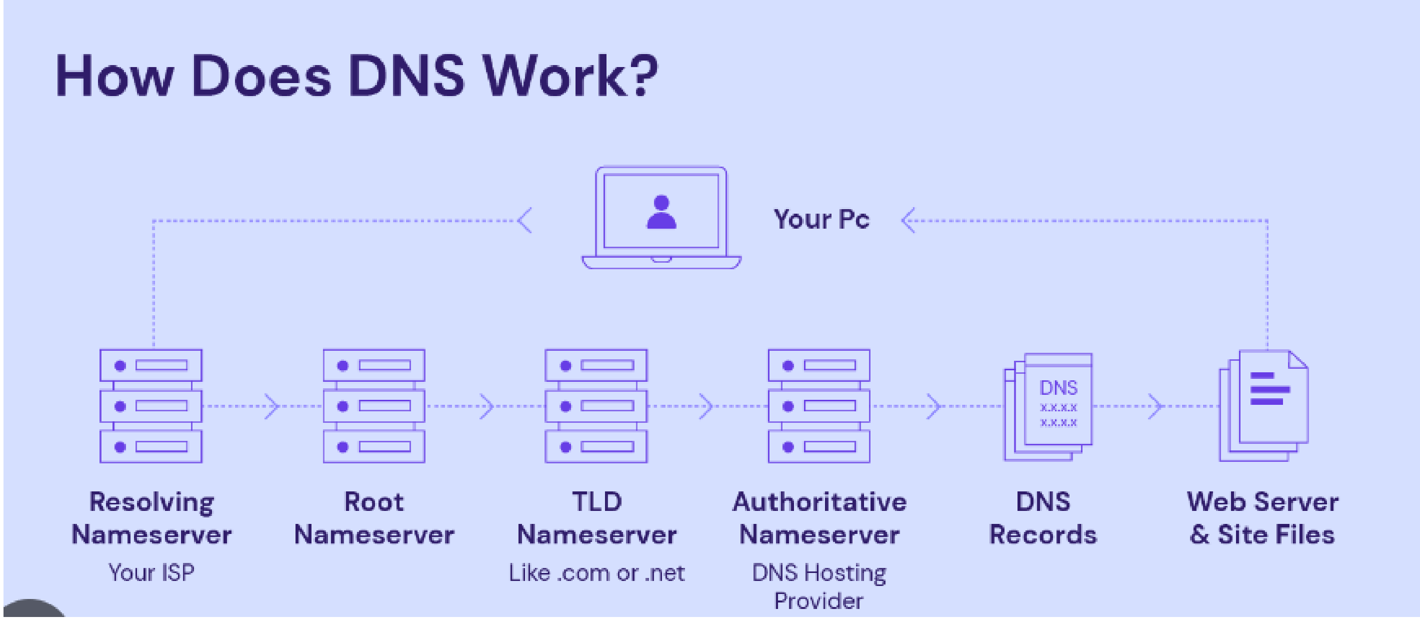


why is important? if we imagine receive some data but we have wholota many process running in your computer so we gonna pick that one receive process

HTTP Request message :
packet of information that is sent by a client

Web caches :
cache's purpose we need faster one So we predict working guys job
it is correct? => fast
and use cache system in web? Web Cache

DNS :



Domain Name System, not Service i was know this is Service
if join some site we cant read longlonglong ip adress so we make easy one

DNS Protocol :
Dns query => Dns resolver => request Dns Server => respone ip address

FTP Protocol :
these day we dont use this
why? it is slow... but until use multi all in one printer scan

Host and Domain :

guys are related terms that are used to identify resources on the internet.

Video streaming Service :

down divided video file and run, down, run

Tcp socket programming :

```
#include <iostream>
#include <cstring>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>

using namespace std;

int main(int argc, char const *argv[]) {

    int sock = 0, valread;
    struct sockaddr_in serv_addr;
    char *hello = "Hello from client";
    char buffer[1024] = {0};

    // Create socket
    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0) {
        cout << "Socket creation error" << endl;
        return -1;
    }

    // Set server details
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(8080);

    // Convert IPv4 and IPv6 addresses from text to binary form
    if (inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr) <= 0) {
        cout << "Invalid address/ Address not supported" << endl;
        return -1;
    }

    // Connect to server
    if (connect(sock, (struct sockaddr *)&serv_addr, sizeof(serv_addr)) < 0) {
        cout << "Connection Failed" << endl;
        return -1;
    }

    // Send message to server
    send(sock, hello, strlen(hello), 0);
    cout << "Hello message sent to server" << endl;

    // Receive message from server
    valread = read(sock, buffer, 1024);
    cout << buffer << endl;

    return 0;
}
```

Roh

이전 포스트
백준 10815 [c++] 숫자 카드

0개의 댓글

댓글을 작성하세요

댓글 작성