SITE 1101: Principles of Information Systems

Week 09, 10

Internet and World Wide Web

Authors: Nilufar Ismayilova, Ismayil Shahaliyev, Rumiyya Alili

Created / Updated: Nov 21 2025 / Dec 21 2025

The Internet is a global network of interconnected computer networks that allows devices around the world to communicate. It is not controlled by any single organization; instead, many independent networks – such as home networks, university networks, company networks, and government networks – connect to each other voluntarily. Together, these connections form a large, decentralized system that enables information to travel quickly and reliably between distant points.

For communication to be possible across such a system, all connected devices must follow common rules. These rules are called **network protocols**. A *protocol* defines how data is sent, in what format, in what order, and what happens if errors occur. Without protocols, networks could be physically connected, but devices would not understand each other, making meaningful communication impossible.

The Internet follows a hierarchical structure. Local networks connect to Internet Service Providers (ISPs), which then connect to larger regional networks and finally to high-speed *backbone* networks that carry data across countries and continents. When you open a website or send a message, your data travels through this chain of networks according to agreed-upon protocols until it reaches its destination.

Internet Protocol Suite

The Internet Protocol Suite, commonly known as **TCP/IP**, is the foundational set of communication rules that defines how data is prepared, addressed, transmitted, and received across the Internet. It organizes network communication into layers, where each layer performs a specific part of the process. At the core of the suite is the Internet Protocol (IP), which assigns unique addresses to devices and routes data packets from the sender to the correct destination, even if they pass through many different networks. Above IP, the suite includes *transport protocols* that manage how data is broken into smaller pieces, how those pieces travel, and how they are reassembled when they reach the receiver. Together, these layers ensure that data moves smoothly across any combination of networks, making global communication possible.

TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are the two main transport protocols used in the Internet Protocol Suite, and both serve different purposes in network communication. TCP provides reliable, ordered, and error-checked delivery of data. It establishes a connection between the sender and receiver before any data is transferred and ensures that every packet arrives correctly. If a packet is lost, TCP automatically resends it. This makes TCP suitable for activities where accuracy matters, such as loading websites, sending emails, or downloading files. UDP, on the other hand, is a faster but less reliable protocol. It sends packets without establishing a connection and does not check whether the receiver actually gets them. Lost packets are simply skipped, not retransmitted. Because it avoids extra overhead, UDP is used for real-time applications where speed is more important than perfect accuracy, such as video calls, online gaming, and live streaming.

Example. When you visit a website such as *www.ada.edu.az*, your computer sends a request that is divided into packets and addressed to the server's IP address. Routers along the path use this address to forward the packets across multiple networks until they reach the destination. Because web traffic uses TCP, the packets arrive reliably and in order. The server receives the complete request, processes it, and sends the response back using the same transport mechanism. If the same exchange were part of a real-time application, such as a video call or an online game, UDP could be used instead. In that case, packets would be sent without waiting for acknowledgments, favoring speed over perfect delivery.

IP Address

IP addressing is one of the core functions of the Internet Protocol Suite because it allows every device on a network to be uniquely identified. Without IP addresses, routers would have no way of knowing where data packets should go. An IP address functions much like a home address: it tells the network exactly where to deliver information and where it came from. The most common earlier version, IPv4, uses a 32-bit structure divided into four decimal numbers separated by dots, such as `192.168.1.4`. Each number represents eight bits, and together they uniquely identify a device on a network.

As the number of Internet-connected devices grew rapidly, the IPv4 address space became insufficient. To solve this limitation, IPv6 was introduced. IPv6 uses 128 bits and is written as eight hexadecimal blocks separated by colons, for example `2001:db8:85a3::8a2e:370:7334`. This much larger address space allows an enormous number of unique addresses (each block has the range 0000-FFFF, instead of 0-255), removing the need for address scarcity workarounds.

Exercise. How many devices can be encoded with IPv4? What about IPv6?

Domain Name System

Domain Name System (DNS) is a global directory service that translates human-readable website names into numerical IP addresses that computers use to identify servers on the Internet. Without DNS, users would have to memorize long strings of numbers instead of simple domain names. It works automatically in the background, making it possible for browsers to locate and connect to the correct server whenever a user types a website name.

DNS has a hierarchical structure made of several levels that help organize domain names from the highest level down to specific services. The DNS hierarchy is organized in levels that together form a structured naming system. At the top is the root domain, represented by a dot (`.`), which is usually hidden from users but underlies all domain names. Below it are top-level domains (TLDs) such as `.com, .az, .org`, which classify domains by purpose or region. Under each TLD are second-level domains, which are the names registered by organizations or individuals, such as `edu.az`. These can be further divided into subdomains, which identify specific services or sections within an organization, for example `ada.edu.az` or `library.ada.edu.az` (subdomain of a subdomain).

Example. Imagine you want to visit *www.ada.edu.az*. Your computer does not know the IP address of this website, so it cannot contact the server directly. Instead, it asks DNS to look it up.

1) You type: *www.ada.edu.az*
2) Your computer asks a DNS server: *"What is the IP address for this website?"*
3) DNS checks its records or asks other DNS servers if needed.
4) DNS replies with an IP address, for example: *104.21.45.120*
5) Your browser uses this IP to contact the ADA University server and load the webpage.

So, DNS allows you to use easy names like *www.ada.edu.az* instead of memorizing long IP numbers. It works automatically in the background every time you visit any website.

Routing

Routing is the process by which routers determine the best path for data packets to travel across networks from source to destination. When you send data over the Internet, it does not travel in a straight line. Instead, it passes through multiple routers, each making decisions about where to forward the packet next. Routing ensures that data reaches its destination efficiently, even when there are many possible paths.

Static Routing involves manually configuring the routes that a router will use. Network administrators enter specific rules that tell the router exactly where to send packets for particular destinations. Static routing is predictable and uses minimal router resources, but it does not adapt to network changes. If a link fails, the router cannot automatically find an alternative path. Static routing is typically used in small, stable networks where routes do not change frequently.

Default Routing is a simplified form of routing where the router is configured to send all packets that do not match any specific route to a single next-hop router. This is useful in small networks or at the edge of larger networks, where there is only one path to reach the rest of the Internet. For example, a home router usually uses default routing, sending all external traffic to the ISP's router.

Dynamic Routing uses protocols and algorithms to automatically discover and adjust routes based on current network conditions. Routers running dynamic routing protocols share information with each other about the state of the network, such as which links are active, which are congested, and which have failed. When a path becomes unavailable, dynamic routing automatically recalculates the best alternative route. This makes dynamic routing highly adaptable and scalable, which is why it is used in large networks and across the Internet backbone.

Example. Imagine you send an email from Baku to a friend in New York. Your data packet leaves your computer and reaches your home router, which uses default routing to forward it to your ISP. The ISP's routers use dynamic routing protocols to determine the best path across the Internet. As the packet travels through multiple routers in different countries, each one reads the destination IP address and decides where to send it next based on current network conditions. If one link is congested or down, dynamic routing ensures the packet takes an alternative path. Eventually, the packet reaches your friend's ISP, then their local router (using default routing), and finally their device. This entire journey happens in milliseconds because routing protocols work automatically and efficiently to move packets across the global Internet.

OSI Model

OSI (Open Systems Interconnection) model explains how networking systems communicate across seven layers. Each layer performs a specific role and interacts only with the layers above and below it.

#	Layer	Function	Example
7	Application	Provides services directly to user applications; defines how browsers, email clients, and apps request network services; uses protocols like HTTP, DNS, FTP, and SMTP.	When you type www.ada.edu.az into your browser, the Application Layer uses HTTP to request the webpage and DNS to resolve the domain name.
6	Presentation	Translates data formats between applications and the network; handles text encoding (ASCII/Unicode), compresses files, and performs encryption/decryption such as TLS/SSL.	When the website uses HTTPS, this layer encrypts your data (TLS) so attackers cannot read login information or credit card details.
5	Session	Establishes, manages, and terminates communication sessions between computers; keeps track of ongoing dialogs so data streams don't mix.	When watching a Zoom lecture, the Session Layer maintains a stable session so your audio/video stream stays connected even if packets arrive out of order.
4	Transport	Ensures end-to-end communication; provides reliable delivery (TCP), flow control, retransmission of lost packets, or fast low-overhead delivery (UDP).	When loading a website, TCP guarantees every piece of the webpage arrives in order. For online gaming, UDP sends rapid updates without waiting for lost packets.
3	Network	Handles routing of packets between networks; assigns logical IP addresses and finds the best path through routers; uses the IP.	Routers along the Internet path use the destination IP address to forward your webpage request toward the ADA University server.
2	Data Link	Creates frames for local network communication; uses MAC addresses; handles error detection on the link (Ethernet, Wi-Fi frames).	Your laptop sends the request to your Wi-Fi router using the router's MAC address, and the router sends frames to your ISP.
1	Physical	Transmits raw bits (0s and 1s) through cables, fiber optics, or wireless signals; defines voltages, frequencies, modulation.	Wi-Fi converts frames into radio waves, or Ethernet converts them into electrical pulses that travel through a cable.

Example. When you open *www.ada.edu.az*, the process begins at the Application Layer, where the browser prepares an HTTP request. The Presentation Layer applies encryption if the connection uses HTTPS. The Session Layer sets up and maintains the communication session. The Transport Layer (TCP) breaks the request into small segments and ensures they are delivered reliably. The Network Layer (IP) attaches source and destination IP addresses so routers can forward the data across different networks. The Data Link Layer formats the data into frames addressed to your local router's MAC address. Finally, the Physical Layer converts the frames into radio signals or electrical pulses that travel across Wi-Fi or Ethernet. On the server's side, the same layers work in reverse, eventually allowing your browser to receive the website's content and display it on your screen.

World Wide Web

The World Wide Web is a global information system that allows users to access and interact with digital content through web browsers. It consists of millions of interconnected web pages stored on web servers and accessible through the Internet. Unlike the Internet, which is the physical network of computers and cables, the Web is a *service* that uses that network to deliver information. Web pages are linked using hyperlinks, which allow users to jump from one page to another, creating a web-like structure of information. Each web page is written using HTML and can include text, images, videos, animations, or interactive elements. Browsers such as *Chrome, Firefox,* and *Safari* request pages from servers and render them into a visual format the user can understand.

Example. Imagine you want to check your course grades on the ADA University website. You open your browser and type *www.ada.edu.az***.** Your browser contacts the ADA web server and requests the HTML file that contains the homepage. The server sends back this page along with images, styles, and scripts. Your browser then renders the information into a visually formatted website. All of this interaction, requesting pages, receiving files, navigating through hyperlinks, is what makes up the World Wide Web.

Hypertext Transfer Protocol (HTTP) is the foundational communication protocol used by the Web. It defines how web browsers send requests and how web servers respond. When you visit a website, your browser sends an HTTP *request* asking the server for specific resources such as HTML pages, images, or scripts. The server then sends an HTTP *response* containing the requested content, along with a status code (like `200 OK` or `404 Not Found`) that explains whether the request was successful.

Modern websites usually use HTTPS instead of HTTP. HTTPS (S stands for “secure”) adds an encryption layer through Transport Layer Security (TLS), which protects the data being sent between the browser and the server. This prevents attackers from reading or modifying information, making HTTPS essential for activities like logging into accounts, entering credit card information, or sending personal data.

Example. Suppose you log into your ADA student account. When you type your username and password and click “Sign In”, your browser sends an HTTPS request to the ADA authentication server. The server processes your credentials and sends back an HTTPS response that either grants access or shows an error. Because the connection uses HTTPS, your login information is encrypted, preventing anyone on the network, such as someone on the same Wi-Fi, from reading your password. Without HTTPS, the login request could be intercepted and viewed in plain text.

Uniform Resource Locator (URL) is the full address that identifies exactly where a resource is located on the Web. It tells the browser how to access the resource (protocol), where to find it (domain name), and which specific file or page to retrieve (path). A typical URL may look like *https://www.ada.edu.az/en/search?query=SITE* which can be broken down into clear parts:

- **https** – the protocol indicating secure communication
- **www.ada.edu.az** – the domain name of the server
- **/en/search** – the path to the specific folder and file on the website
- **?query=SITE** – a query parameter used to request a specific item or version of a page

Markup Languages

Markup languages are systems for annotating text so that computers can understand how the information should be structured, displayed, or processed. They do not describe the meaning of the content but rather its organization or formatting. The most widely used markup language is HTML (HyperText Markup Language), which forms the structure of web pages on the World Wide Web. HTML uses tags such as `<h1>`, `<p>`, and `<img>` to define headings, paragraphs, and images. Browsers read these tags and convert them into the visual layouts that users see.

Another well-known markup language is XML (Extensible Markup Language). Unlike HTML, XML is not about displaying information but about storing and transporting data in a structured, readable way. XML allows users to create custom tags that describe data, such as `<student>` or `<course>`, making it useful for data exchange between systems, configuration files, and web services.

Modern web applications often use JSON (JavaScript Object Notation) alongside or instead of XML. JSON is a lightweight data format structured as key–value pairs, making it easy for computers to parse and for humans to read. It is widely used in APIs, web applications, and cloud services because it is simpler and faster than XML.

Example. A university website uses HTML to structure all its pages. For instance, the homepage might use `<h1>Welcome to ADA University</h1>` for the main title, `<p>` for text paragraphs, and `<img>` tags to insert images. Everything the browser displays visually is built from HTML tags.

At the same time, the website's backend might use JSON to exchange data with the server. For example, when a student checks their grades, the system might send back data like: `{"student":"Badambura", "course":"SITE1101", "grade":"A"}`. This data is not meant to be shown directly to the user but allows applications to process and update information quickly.

On the other hand, a library database may store book information in XML, such as:

```
<book>
   <title>Computer Networks</title>
   <author>Andrew Tanenbaum</author>
   <year>2020</year>
</book>
```

Intranet and Extranet

An intranet is a private network used within an organization. It provides internal services such as employee portals, shared files, calendars, and communication tools. Because an intranet is restricted to internal users, it offers better security and control. Only employees or authorized personnel can access it, usually through login credentials or the company's internal Wi-Fi.

An extranet extends the intranet by allowing limited access to external users such as business partners, suppliers, contractors, or students. Organizations use extranets to collaborate with outsiders while still protecting sensitive internal information. Access to an extranet is controlled through accounts, permissions, or Virtual Private Network (VPN) connections.

Example. An *intranet* example is a company's internal management system. Only employees connected to the company's network, or logged in with corporate credentials, can access internal resources such as financial reports, staff schedules, internal announcements, and confidential documents. Customers and the general public cannot access these resources because the intranet is strictly internal and protected. An *extranet* example is when the same company allows selected business partners or contractors to access specific shared resources through a secure login. These external users might view project timelines, shared technical documents, or order statuses, but they cannot access the company's full internal systems.

Cloud Computing

Cloud computing is a model of delivering computing resources, such as storage, servers, networks, databases, and software, over the Internet instead of relying on local hardware. Instead of buying and maintaining physical equipment, users access shared resources from cloud providers on demand. This makes computing more flexible, scalable, and cost-efficient. Organizations can increase or decrease their computing power instantly, paying only for what they use, which is especially useful for businesses with changing workloads or seasonal demand.

Cloud services are often categorized into three main layers. Infrastructure as a Service (IaaS) provides virtual machines, storage, and networking. It is the closest to owning physical hardware but without the responsibility of maintaining it. Developers or IT teams use IaaS when they need control over operating systems and applications. Platform as a Service (PaaS) offers a complete environment for building and deploying applications. It hides server and system management so developers can focus solely on writing code. Software as a Service (SaaS) delivers ready-to-use applications that run entirely online. Users do not install anything; they simply log in through a browser.

Example. Imagine ADA University hosting its learning platform. Instead of buying physical servers, ADA could use Amazon Web Services (AWS). The university might run its website on IaaS, using AWS virtual machines to host pages and store data. Developers working on ADA's student mobile app could use PaaS, such as Google App Engine, to build and test new features without setting up their own servers. Students and staff would use SaaS tools like *Microsoft Outlook* or *OneDrive* to access email, documents, and files. If ADA wants higher security for sensitive student information, it might store that data in a *private cloud* while using the *public cloud* for the main website. This combination is an example of hybrid cloud computing.

Centralized, Decentralized, Distributed Systems

A centralized system relies on a single central authority or server. All users and components depend on this central point for access and coordination. This design is simple to manage but fragile: if the central server fails or is compromised, the entire system may stop working.

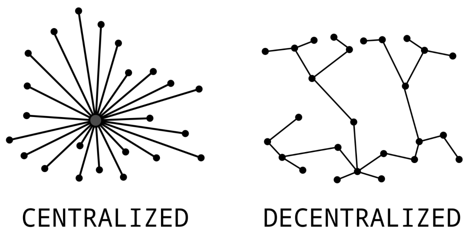

Figure 1. By Kes47 (?) - Based on File:Decentralization.jpg, by Adam Aladdin, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=35018016

A decentralized system has multiple authoritative nodes instead of one central controller. Control and data are spread across several independent points, each of which can operate on its own. If one node fails, others can continue functioning. The Internet itself is decentralized at the network level: no single organization controls it, and many independent networks cooperate to keep it running.

A distributed system goes a step further by splitting computation, storage, and tasks across many nodes that work together as a single system. These nodes coordinate and communicate to achieve a common goal, often appearing as one system to the user. Cloud platforms, distributed databases, and content delivery networks are typical examples. Distributed systems focus on scalability, performance, and fault tolerance rather than on who controls the system.

In short, centralized means one control point, decentralized means multiple independent control points, and distributed means many coordinated components working together. These concepts overlap in practice but describe different design goals and trade-offs.

Additional Material

- What is Internet?
- The Internet: Crash Course Computer Science #29
- Public vs Private IP Address
- How a DNS Server (Domain Name System) works
- TCP vs UDP Comparison
- Network Layers Model (Networking Basics) - Computerphile
- VPN (Virtual Private Network) Explained
- The World Wide Web: Crash Course Computer Science #30
- What is the world wide web? - Twila Camp
- How trees secretly talk to each other - BBC World Service
- SSL, TLS, HTTP, HTTPS Explained
- HTTP Crash Course & Exploration
- HTTP 1 vs HTTP 2 vs HTTP 3!
- DHCP Explained - Dynamic Host Configuration Protocol
- Cloud Computing Explained
- How to Setup VS Code for Web Development (2025) | HTML, CSS, JavaScript + Live Server