COVER: Internet Programming 2

Student Handout 1: Internet and Web Fundamentals

MUT LOGO and notes

NOTES:

The sections in Unit 1 are all about the Internet and the Web, including some related concepts you must be familiar with to make your IP2 journey more comfortable. In general, I provide a set of Learning Outcomes and then the notes to help you navigate the section – you can use any available resource on the web, but take note that these notes serve as the official record of what I will assess you on, even though you may be required to access other resources for your own understanding of the material. The learning outcomes will provide you with a clear guide to what learners should understand, be able to do, or value after engaging with the material.

About Learning Outcomes in these notes:

The learning outcomes which appear in these notes are designed to encapsulate the key concepts and skills associated with each subsection, providing a comprehensive roadmap for understanding the foundational elements of internet technology and web communication.

Similarly, a summary at the end will reinforce these concepts, aiding in retention and comprehension.

---NOTES START next page…

Section One: A Gentle Introduction to the Internet: A Gateway to Global Connectivity

1.0 Learning Outcomes:

Upon completing this section, students will be able to:

1. Identify Different Types of Internet Connections: Students will recognize various methods for connecting devices to the internet, including wired (Ethernet) and wireless (Wi-Fi, 4G/5G) connections.

2. Understand IP Addresses: Students will explain the purpose of IP addresses, distinguish between public and private IP addresses, and understand their role in enabling device communication over the internet.

3. Describe the Function of Routers and Local Networks: Students will outline how routers create local networks, assign IP addresses, and manage data flow between devices and the broader internet.

4. Explain the Role of Servers and Domain Names: Students will understand how servers provide content and services on the internet, and how domain names and the DNS system simplify access to these resources.

5. Comprehend Basic Internet Protocols: Students will grasp the foundational principles of TCP/IP protocols, including IP, TCP, and UDP, and their significance in facilitating reliable internet communication.,

1.1 Introduction to Computer Networks and Internet Protocols

In our digital age, devices such as computers, smartphones, tablets, and even smart home appliances have become integral parts of our daily lives. While these devices offer significant standalone value, their true potential is unlocked when they are connected to the internet. This connection opens up a world of possibilities, enabling access to vast amounts of information, communication tools, and services.

1.2 Connecting Devices to the Internet

Connecting a device to the internet can be achieved through various means, including wireless connections like Wi-Fi and cellular networks (5G/4G), or wired connections using Ethernet cables. Upon connecting, each device is assigned a unique identifier known as an IP (Internet Protocol) address. This address, which consists of four number sets ranging from 0 to 255, allows your device to communicate within the vast network of the internet. For instance, an IP address might look like "192.168.1.105," where the "192.168" portion typically signifies a local network address.

1.3 Navigating the Local Network

Your local network, typically managed by a Wi-Fi or Ethernet router, serves as the intermediary between your devices and the broader internet. The router itself receives a unique IP address from your internet service provider, then assigns distinct local IP addresses to each device on your network. This structure facilitates communication between devices in your home while maintaining privacy and security from external networks.

1.4 The Role of Servers and Domain Names

At the heart of internet communication are servers: powerful computers tasked with delivering content and services. Unlike personal devices, servers possess public IP addresses to be continuously accessible. To simplify human interaction with these servers, domain names (e.g., google.com) are used. The Domain Name System (DNS) plays a crucial role here, translating easily remembered domain names into numerical IP addresses that computers use to locate each other.

1.5 Protocols: The Language of the Internet

The foundation of internet communication is built on protocols, with the Internet Protocol Suite, also known as TCP/IP, being paramount. These protocols define the rules for data exchange over the network. The suite includes:

- IP (Internet Protocol): Manages the routing of data by identifying devices through their IP addresses.

- TCP (Transmission Control Protocol): Ensures reliable delivery of data packets between devices.

- UDP (User Datagram Protocol): Offers a faster, albeit less reliable, alternative to TCP for certain applications.

1.6 How the Web works: A Conceptual View

Now, we take a brief look at "How the Web works". We will look at this from a conceptual point of view and represent the processes using a basic block diagram. This will also help you to understand how the content of IP2 is tied together. We cover Client side aspects (HTML, CSS, JavaScript) and Server side Scripting (PHP, MySQL) in IP2. In the scenario below, we assume the URL the user types actually points to a single file containing a mix of HTML, CSS, JS and PHP – yes, you read right. See Figure 1 for an example of how this is structured:

```
<!DOCTYPE html>
<html>
<style>
  h2 { color: maroon;
       margin-left: 40px; }
</style>
<body>
     <h2>What Can JavaScript Do?</h2>
    <button type="button"
    onclick="document.getElementById('demo').innerHTML = 'Hello
    JavaScript!'">Click Me!</button>

    <?php echo "My first PHP script!";?>
</body>
</html>
```

Figure 1: Sample.php file

In essence, we want to explore some of the processes that are involved when a user types a URL in the address bar of the browser (for example: https://www.Ramdeyal.com/Exercise1/Sample.php) and presses <<Enter>> until the web page gets displayed in the browser window. This is clearly a complex process, involving several activities in the background by the browser, the communication system, and the server. We will, however, focus merely on those aspects which are important to our current and later work. This means that Figure 2 below is intended to capture only what we need to understand now, and that we will return to this diagram and adapt it to contain more elements as we proceed.

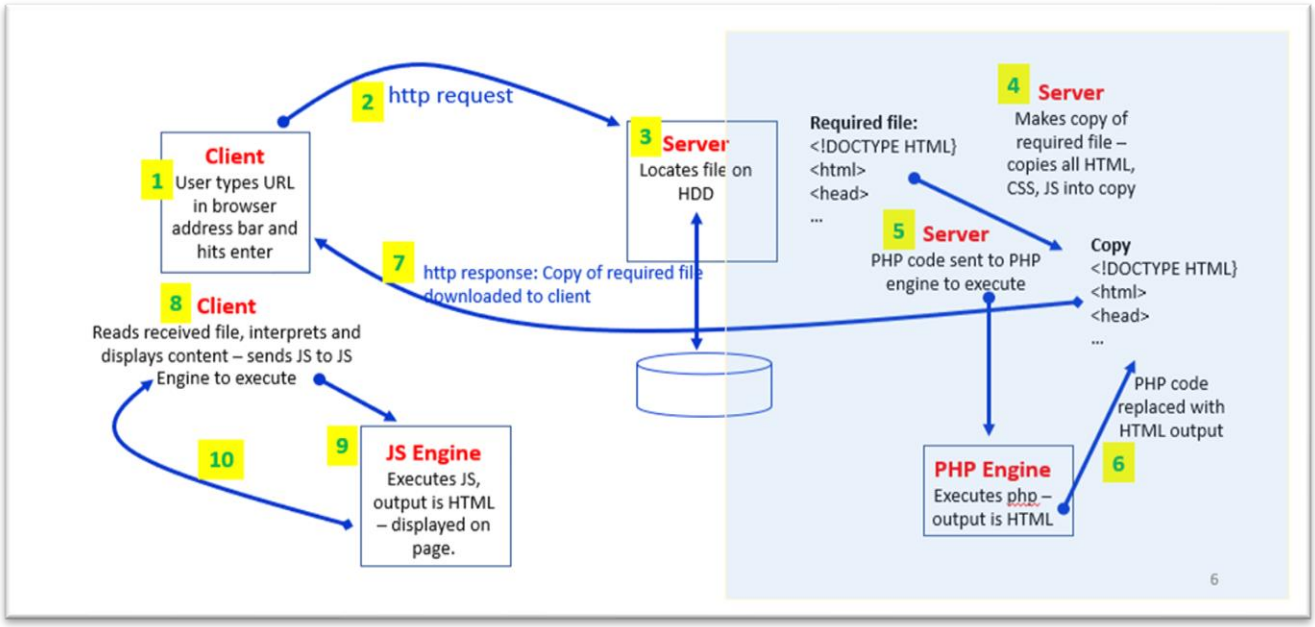

Figure 2: Conceptual View of How the Web Works

Figure 2 Explained:

1-user types URL in browser and hits enter

2-Browser formulates a message containing several parameters, the key to which are the source and destination addresses and other pertinent information that we will explore in detail later

3- server locates the required file and begins the copy process (step 4). In this example, the name of the required file is included in the URL, but if it is not, then the server attempts to find a file called default.htm or index.htm [.php, .asp]. From a legacy point of view, the file name default is Microsoft's method and Index was the file the Novel System was programmed to use for the home page. QUES: What do you think will happen if (1) both are present, and (2) neither are present in the folder that the server is looking at?

4- the server copies only those parts of the source code that the Browser "understands" (can interpret) which are only HTML, CSS and JS. If the server finds Server-side code (PHP in our case), it stops the copy process and send the PHP code for processing (step 5)

5- the PHP "Engine" / Run-time system executes the PHP code. The output from our PHP scripts is always HTML compliant (meaning that the browser will be able to interpret). This is true even if the PHP engine engages with other resources such as web services, other sites and external and internal databases.

6 -the server pastes the PHP engine output into the copied file, exactly where the PHP source code appeared in the original file. These continue until there is no more source code to copy.

7-after some housekeeping, the server send the copied file to the client

8- on receipt of the file, the browser interprets the source code and does the necessary processing to display the page content in the users browser window. Note that the file it receives contains only HTML, CSS, JS so it is able to interpret the source code and display what is needed to be displayed in the browser.

9- FYI: previously (or back in the old days), developers would have to indicate the scripting language and had to make sure that the browser has access to the “engine” to process the client side script. However, HTML5 compliant browsers are able to easily execute JS Code without any additional steps by the developer – i.e. JS has become the Standard client-side scripting language that all available browsers which are HTML5 compliant are able to handle. That said, in Figure 2, the steps 8, 9, and 10 are equivalent to how the server handles PHP scripts, the browser has a JS Engine to handle the JS Scripts…again, the JS output is always HTML-compliant and so can be displayed by the browser.

1.7 Summary of Section 1: Understanding the Internet

In this section, we've explored the foundational elements that enable our devices to connect to and communicate over the internet. We began by discussing how devices such as computers, smartphones, and smart appliances connect to the internet through various means, emphasizing the importance of IP addresses as unique identifiers for these devices. We then delved into the structure of local networks, highlighting the role of routers in assigning IP addresses and managing communication within and outside the network.

We also examined the critical function of servers in the internet ecosystem, serving content and services to users worldwide. The concept of domain names and the DNS system was introduced, illustrating how these elements simplify the process of accessing internet resources by translating human-friendly domain names into machine-readable IP addresses.

Subsequently, we covered the core protocols that govern internet communication: TCP/IP protocol suite: This is the set of rules that governs how data is transferred between computers on the internet. IP is the base layer that handles addressing and routing, while TCP is the transport layer that ensures reliable and ordered delivery of data packets. UDP is another transport protocol that is faster but less reliable than TCP.

Finally, we took a brief tour of how the web works, using a block diagram.

By understanding these fundamental concepts, students are better equipped to appreciate the complexities of internet technology and its integral role in our digital lives.

This section has laid the groundwork for understanding how our devices connect and communicate over the internet. By exploring the intricacies of networks, IP addresses, servers, and protocols, we begin to appreciate the complex yet fascinating world of internet technology. Now, we take a brief look at “How the Web works”. We will look at this from a conceptual point of view and represent the processes using a basic block-type of a diagram.

1.8 Check my Knowledge:

1. Explain the distinction between client-side and server-side processing, including examples of technologies and scenarios where each is used. How do these processes impact the performance and functionality of web applications?

2. Discuss the mechanisms web browsers employ to handle errors in HTML, CSS, and JavaScript. How do these error-handling strategies affect the user experience and website functionality?

3. Examine how browsers interpret and render white space in HTML documents. How does this understanding influence web design and development practices?"

4. Delineate the differences between the Internet and the World Wide Web (WWW). How does each contribute to what we commonly refer to as 'browsing the web'?"

5. Compare and contrast compilers and interpreters, focusing on their roles in software development. Include examples of programming languages that typically use each.

6. What occurs when a user omits the specific file name in a URL, accessing a directory instead? Discuss the server's response and the role of default documents in web servers.

(********** End$ **********)

Section Two: Exploring Internet Fundamentals: URLs and Ports

2.0 Learning Outcomes

Upon completing this section, students will be able to:

- Define what a URL is and its components: protocol, server address, and document path.
- Understand the purpose and function of ports in network communication and how they enable multiple services on a single device.
- Explain how domain names are resolved to IP addresses through DNS.
- Differentiate between commonly used ports and their purposes (e.g., HTTP, HTTPS, FTP).

2.1 Understanding URLs

A URL (Uniform Resource Locator) is essentially the address you input into a web browser to retrieve a page from a server. It's structured to include several key components:

`http://` or `https://` signal we want to use HTTP or HTTPS (secure HTTP).

1. Protocol: Indicated at the beginning of the URL by "http://" for Hypertext Transfer Protocol or "https://" for its secure version, HTTPS. This part specifies how data is exchanged between your browser and the server.

2. Server Address: This can be a domain name (like "google.com") or a direct IP address (such as "142.251.209.14"). It directs the request to the correct server on the internet.

`google.com` or `142.251.209.14` .

Combined, the protocol and address look like this: `https://google.com`

3. Document Path: Following the server address, this component specifies the exact location of the document or resource on the server. For example, in the URL "https://ramdeyal.com/debugging/", "https" is the protocol, "ramdeyal.com" represents the domain name, and "/debugging/" is the path to a specific document on the server.

For example, I have a page on my website ramdeyal.com, its URL is: https://ramdeyal.com/debugging/

- https is the protocol.
- ramdeyal.com is the domain name that points to the server
- /debugging/ is the document URL relative to the server root path.

The web server then interprets this request and serves the appropriate document back to the client, facilitating the web browsing experience.

2.2 Demystifying Ports

In addition to IP addresses or hostnames, network requests often include a port number. Ports serve as communication endpoints on a computer, allowing it to distinguish between different types of traffic. For instance, accessing a resource via "http://localhost:8080" uses port 8080, while "ftp://127.0.0.1:4321" would use port 4321.

Ports enable a single machine to run multiple server applications by assigning each a unique endpoint. For example, a standard web server might operate on port 80 for HTTP, while an alternative or secondary web server on the same machine could use port 8080 or 81. Similarly, HTTPS traffic typically uses port 443 by default.

While there are standard ports for most protocols, services are free to operate on any available port within the range from 1 to 65535. This flexibility supports multiple applications to run concurrently on the same physical server without conflict, enhancing the utility and scalability of networked services.

This exploration of URLs and ports sheds light on how the internet navigates and differentiates among the vast array of services available, ensuring that requests from your browser reach their intended destination accurately.

2.3 Summary

Summary of Section Two: Exploring Internet Fundamentals: URLs and Ports

In this section, we delved into the fundamental components that facilitate internet navigation and communication: URLs and ports. Understanding URLs, or Uniform Resource Locators, is crucial as they serve as the web addresses that direct our browsers to the specific resources we wish to access online. A URL comprises three main parts: the protocol (such as HTTP or HTTPS), which dictates how data is exchanged; the server address, which can be a domain name or an IP address, guiding the request to the correct server; and the document path, specifying the resource's exact location on the server.

Moreover, we explored the concept of ports, which act as communication endpoints on a computer. Ports allow a single device to differentiate between traffic types, enabling it to run multiple server applications simultaneously without confusion. This functionality is made possible by assigning each service a unique port number, with standard ports designated for common protocols like HTTP (port 80) and HTTPS (port 443), among others. Through the use of ports, the internet can efficiently manage and direct the vast quantities of data traffic, ensuring that each request reaches its intended destination.

2.4 Check My Knowledge

1. Break down the structure of a URL and explain how each component directs your web browser to the desired resource. Use an example URL to illustrate your point.

2. Discuss the role of ports in enabling a computer to run multiple services simultaneously. Provide examples of how ports differentiate services such as HTTP, HTTPS, and FTP.

3. Explain the process of resolving a domain name to its corresponding IP address. Why is the Domain Name System (DNS) crucial for this process?

4. Identify and differentiate between the purposes of commonly used ports, such as port 80 for HTTP and port 443 for HTTPS. How do these standards benefit the functionality of the internet?"

5. How does the use of HTTPS in a URL enhance security compared to HTTP? Discuss the underlying mechanisms that contribute to this security enhancement.

6. Consider the implications of port numbers on web development and hosting services. How might the choice of a non-standard port affect the accessibility and security of a web service?

(********** End$ **********)

Section Three: Delving into Internet Protocols: DNS, TCP, and UDP

3.0 Learning Outcomes

Upon completing this section, students will be able to:

- Describe the Domain Name System (DNS) and its role in mapping domain names to IP addresses.
- Explain the concept of DNS servers and their hierarchical structure, including root, TLD, and authoritative name servers.
- Understand TCP's role in establishing reliable connections and data transmission between clients and servers.
- Identify the differences and use cases for TCP and UDP protocols.

3.1 The Domain Name System (DNS) Protocol

Navigating the internet typically involves using domain names, such as "google.com" or "flaviocopes.com," rather than numerical IP addresses. This preference for names over numbers is facilitated by the Domain Name System (DNS), a hierarchical system that translates domain names into IP addresses. DNS operates through a network of servers, starting from your internet service provider to potentially global DNS servers, like Google's DNS at 8.8.8.8. This network is structured like a tree, with a root DNS server at its top that knows the IP addresses of other DNS servers responsible for specific domain extensions (.com, .net, .org, etc.). This architecture allows for efficient resolution of domain names to IP addresses, incorporating mechanisms for caching and redundancy to handle high volumes of requests efficiently.

3.2 The Transmission Control Protocol (TCP)

TCP, or Transmission Control Protocol, forms the backbone of the internet, supporting web browsing, email, and other applications. Established by RFC 793 in 1981, TCP is a foundational protocol that ensures reliable data transmission over the internet. Unlike its counterparts IP and UDP, TCP requires the establishment of a connection between communicating parties before data transfer can begin. This process involves a series of steps known as the handshake, which sets up the connection and ensures data integrity. One of TCP's key features is its ability to confirm that data packets are received correctly, resending any that are lost. TCP also uses ports in combination with IP addresses to facilitate process-to-process communications, enabling multiple applications to run concurrently on a single device.

3.3 The User Datagram Protocol (UDP)

UDP, or User Datagram Protocol, offers an alternative to TCP for data transmission over the internet. Defined in RFC 768 in 1980, UDP is distinguished by its connectionless nature, resulting in faster data transfer rates due to a simpler handshake process and less overhead. However, this speed comes at the cost of reliability; unlike TCP, UDP does not inherently provide mechanisms for ensuring data packets are delivered or checking their integrity. Applications using UDP must implement their own methods for verifying transmission success. Despite these limitations, UDP is critical for applications requiring rapid data exchange, such as DNS, DHCP, and the emerging HTTP/3 protocol, highlighting its continued importance in the internet's protocol suite.

Together, DNS, TCP, and UDP comprise key components of the internet's protocol ecosystem, each serving distinct but complementary roles in the seamless functioning of network communications.

3.4 Summary

In this section, we explored the intricacies of key internet protocols that enable efficient and reliable communication over the network: DNS, TCP, and UDP. The Domain Name System (DNS) plays a crucial role in translating human-friendly domain names into machine-readable IP addresses, ensuring users can easily access websites without memorizing numerical addresses. We delved into DNS's hierarchical structure, including root, TLD (Top-Level Domain), and authoritative name servers, which together facilitate the swift resolution of domain names across the internet.

We then examined the Transmission Control Protocol (TCP), a cornerstone of the internet that guarantees reliable data transmission between clients and servers. TCP's handshake process establishes a secure connection before any data exchange, ensuring integrity and orderliness in data transfer. This protocol is vital for applications where data delivery must be assured, such as web browsing and email communication.

In contrast, the User Datagram Protocol (UDP) offers a faster, though less reliable, alternative for data transmission. Its connectionless nature allows for quicker exchanges without the need for establishing a connection, making it ideal for applications where speed is paramount, such as live video streaming or online gaming. Despite its limitations in reliability, UDP remains indispensable for certain internet services, including DNS queries and real-time communications.

Through understanding DNS, TCP, and UDP, students gain insights into the foundational protocols that underpin the functionality and reliability of the internet, highlighting the balance between speed, reliability, and efficiency in digital communications.

3.5 Check my Knowledge

1. How does the hierarchical structure of DNS contribute to the efficiency of domain name resolution?

2. Describe the TCP handshake process. Why is it essential for reliable communication?

3. Compare and contrast the use cases for TCP and UDP. Give examples of applications best suited for each protocol.

4. Explain the significance of port numbers in TCP and UDP communications. How do they facilitate process-to-process communication?

5. Discuss the trade-offs between using TCP and UDP for data transmission. What factors might influence a developer's choice between the two?

(********** End$ **********)