Computer Networking Project Report: Hosting a Website & Understanding Networking Concepts

Team Members

- Zeba Karobi
- Sebastian Geer
- Rian Merritt
- BhaskarTeja Pidugu

Project Overview

This project aims to demonstrate the process of setting up a website using GitHub Pages while incorporating key networking concepts such as DNS, IP addressing, and communication protocols. The goal is to gain practical experience in web hosting, domain name resolution, and network security fundamentals.

Website Details

- **Website URL:** https://bhaskartej.github.io/csc_computer_networks_proj1/
- **GitHub Repository:** https://github.com/Zkarobi/csc_computer_networks_proj1

Steps Taken

1. Set Up a GitHub Repository

- Created a public repository on GitHub.
- Uploaded website files (HTML, CSS).
- Configured repository settings for hosting.

2. Built the Website

- Designed a simple webpage using HTML and CSS.
- Ensured responsiveness for different screen sizes.
- Applied a clean and structured layout.

3. Hosted on GitHub Pages

- Enabled GitHub Pages in the repository settings.
- Configured the correct branch for deployment.
- Verified successful hosting by accessing the website.

4. Configured DNS & IP Addressing

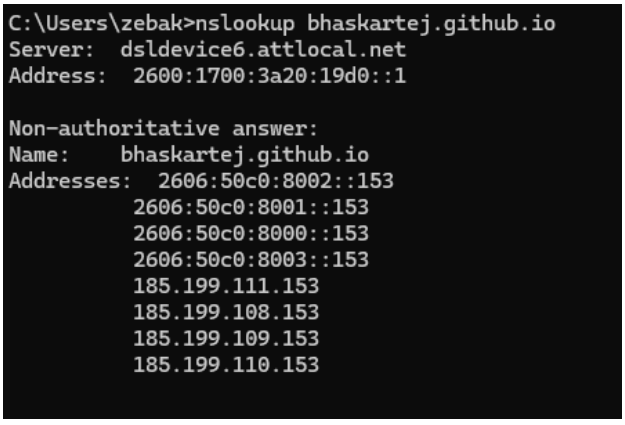
- GitHub Pages assigned a subdomain (e.g., `bhaskartej.github.io`).
- DNS translates the domain into an IP address associated with GitHub's servers.
- Used `ping` and `traceroute` commands to verify IP addresses and routing paths.

5. Implemented Security Measures

- Enabled HTTPS for secure communication (default in GitHub Pages).
- Verified SSL/TLS encryption using browser security checks:
    - Checked the padlock icon in the browser address bar to confirm HTTPS encryption.
    - Clicked on the padlock to view SSL certificate details and ensure validity.
    - Used browser developer tools (Security tab) to inspect TLS settings and encryption protocols..
- Ensured only necessary files were exposed publicly to mitigate security risks.

6. Verified Networking Aspects

- **DNS Lookup:** Used `nslookup` to verify domain resolution and obtain IPv4 and IPv6 addresses.



- **IP Address Validation:** Ran `ping bhaskartej.github.io` to retrieve the assigned IP and check network latency.
    - **Results:**
        - IP Address returned: `2606:50c0:8001::153`
        - Response times: `22-23ms`
        - No packet loss observed.

```
C:\Users\zebak>ping bhaskartej.github.io

Pinging bhaskartej.github.io [2606:50c0:8001::153] with 32 bytes of data:
Reply from 2606:50c0:8001::153: time=22ms
Reply from 2606:50c0:8001::153: time=22ms
Reply from 2606:50c0:8001::153: time=22ms
Reply from 2606:50c0:8001::153: time=23ms

Ping statistics for 2606:50c0:8001::153:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 22ms, Maximum = 23ms, Average = 22ms
```

- **Traceroute Analysis:** Used `traceroute bhaskartej.github.io` (Linux/macOS) or `tracert bhaskartej.github.io` (Windows) to analyze the path packets take to reach the GitHub server.
    - **Results:**
        - Network path included ISP and backbone nodes.
        - Final hop resolved to `2606:50c0:8001::153`.
        - First hop timeout likely due to firewall restrictions on the local network.

```
C:\Users\zebak>tracert bhaskartej.github.io

Tracing route to bhaskartej.github.io [2606:50c0:8001::153]
over a maximum of 30 hops:

  1     *        *        *     Request timed out.
  2    21 ms    21 ms    21 ms  2001:506:6000:120:76:253:216:103
  3    21 ms    21 ms    21 ms  2001:506:6000:120:76:253:216:97
  4    22 ms    21 ms    21 ms  2606:50c0:8001::153

Trace complete.
```

Networking Concepts Applied

1. Domain Name System (DNS)

- GitHub Pages assigns a subdomain like `bhaskartej.github.io`.
- The DNS converts the human-readable domain into an IP address.
- Verified using `nslookup` and browser-based WHOIS lookups.

2. IP Addressing

- GitHub Pages serves websites through multiple predefined IP addresses.
- Checked assigned IPs using `nslookup bhaskartej.github.io` and confirmed IPv4 and IPv6 resolution.

- These IP addresses are mapped to GitHub's content delivery network (CDN) for optimized distribution.

3. HTTP & HTTPS Protocols

- GitHub Pages enforces HTTPS for secure communication.
- SSL/TLS encryption protects against man-in-the-middle attacks.
- Verified through browser security indicators and SSL certificate inspection.

4. Ping & Traceroute for Network Analysis

- **Ping Command:** Verified website availability and response times.
- **Traceroute Command:** Mapped the network path from our local machine to GitHub's servers.
- Showed multiple hops and network nodes traversed during data transmission.

Challenges & Solutions

1. Challenge: GitHub Pages Site Not Loading Immediately

- **Issue:** After enabling GitHub Pages, the website was initially unreachable.
- **Solution:** We waited for DNS propagation and ensured the correct branch was selected for deployment.

2. Challenge: Understanding How GitHub Assigns IP Addresses

- **Issue:** Needed to verify how GitHub Pages resolves domain names to IPs.
- **Solution:** Used `ping`, `nslookup`, and `traceroute` to analyze IP mappings and network routing.

Conclusion

This project provided hands-on experience in web hosting, DNS resolution, IP addressing, and security concepts. By setting up a website with GitHub Pages, we gained a deeper understanding of networking principles and practical troubleshooting techniques. The ability to analyze network routes, understand domain resolution, and implement security features was invaluable in strengthening our knowledge of computer networking.