

WAPH-Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: LakshmaReddy Bandi

Email: bandild@mail.uc.edu



Figure 1: LakshmaReddy Bandi

Lab 1 - Foundations of the WEB

Overview: This lab explores web technologies, focusing on HTTP protocol and web application programming. It uses Wireshark, TELNET, CGI programs, HTML templates, and PHP. Examines HTTP requests using Wireshark and curl.

<https://github.com/nakkantm-uc/waph-nakkantm/blob/main/labs/lab1/README.md>.

Part 1 : The WEB and the HTTP Protocol

Task 1. Familiar with the Wireshark tool and HTTP protocol

Wireshark is a protocol analyzer tool used for network analysis and troubleshooting. Installed on a Ubuntu VM, it captures packets and filters HTTP protocols. It analyzes HTTP requests and responses, providing information on request type, target URL, version, content-type, authorization, and data sent to web servers and back to the web browser.

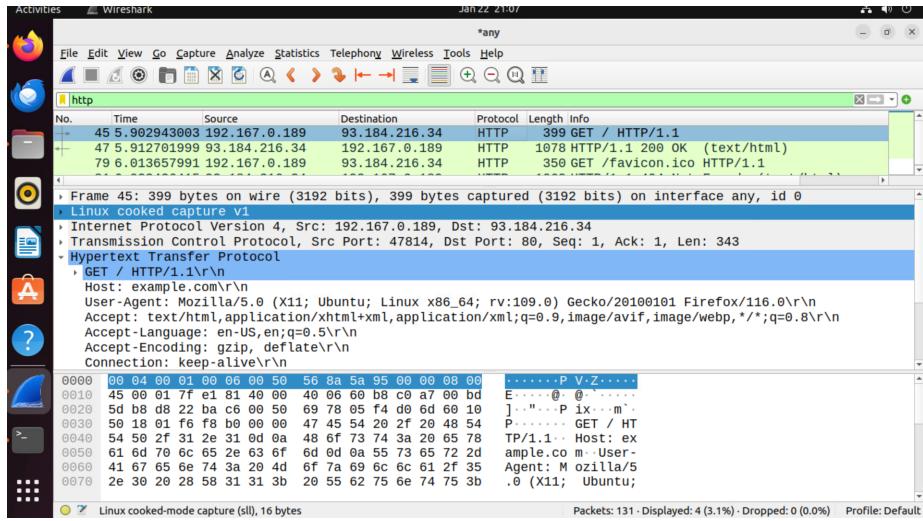


Figure 2: Wireshark HTTP Request

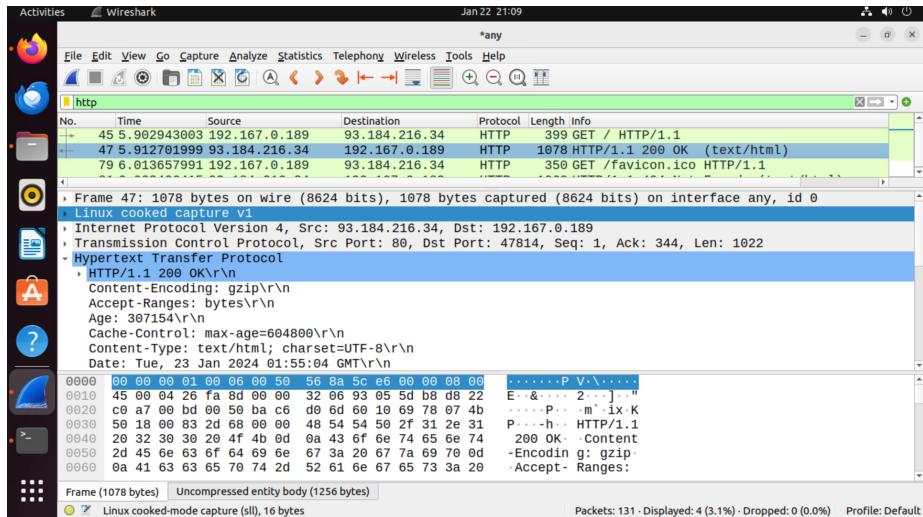


Figure 3: Wireshark HTTP Response

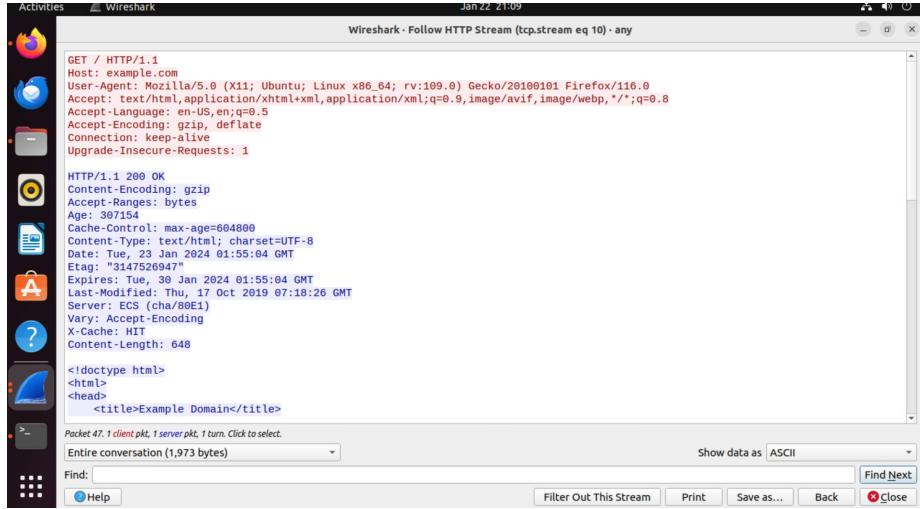


Figure 4: Wireshark HTTP Stream

Task 2. Understanding HTTP using telnet and Wireshark

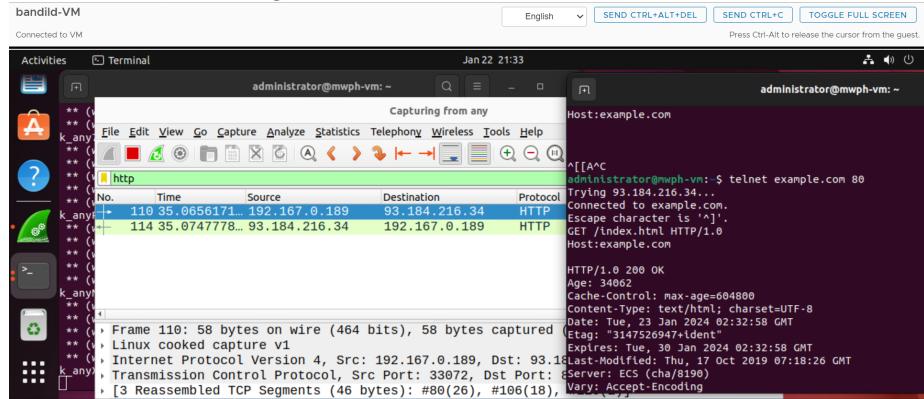
Wireshark captured network packets before making an HTTP request to exmaple.com/index.html via TELNET. Connection was established, request type, path file, http version, and host name were provided, and response received.

```
** (wireshark:119040) 20:55:10.745574 [Capture MESSAGE] -- Capture stopped.
administrator@mwph-vm:~/waph-band1d$ telnet example.com 80
Trying 93.184.216.34...
Connected to example.com.
Escape character is '^].
GET /index.html HTTP/1.0
Host:example.com

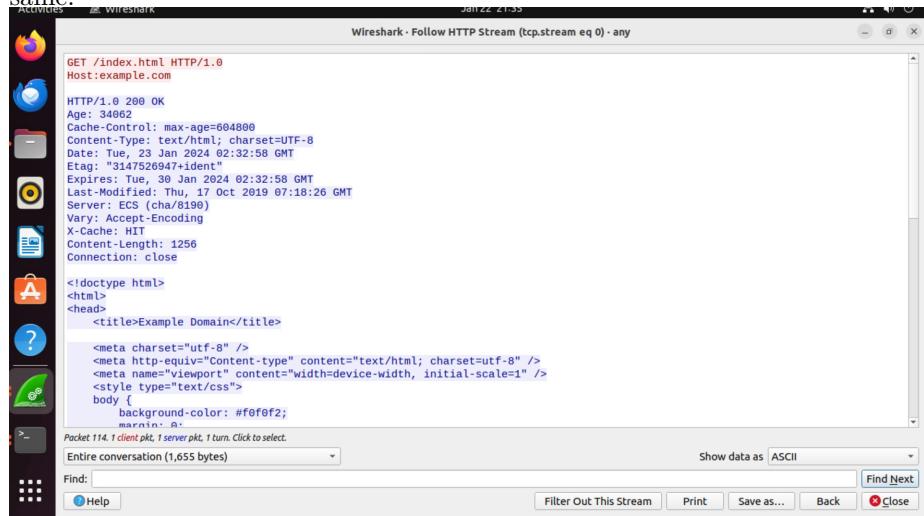
HTTP/1.0 200 OK
Age: 33388
Cache-Control: max-age=604800
Content-Type: text/html; charset=UTF-8
Date: Tue, 23 Jan 2024 02:21:44 GMT
Etag: "3147526947+ident"
Expires: Tue, 30 Jan 2024 02:21:44 GMT
Last-Modified: Thu, 17 Oct 2019 07:18:26 GMT
Server: ECS (cha/8190)
Vary: Accept-Encoding
X-Cache: HIT
Content-Length: 1256
Connection: close

<!doctype html>
<html>
<head>
    <title>Example Domain</title>
<meta charset="utf-8" />
<meta http-equiv="Content-type" content="text/html; charset=utf-8" />
```

The telnet HTTP request in wireshark lacks server fields, while the browser-sent request automatically populates headers like user-agent, accept, accept-language, authorization, encoding, and content.



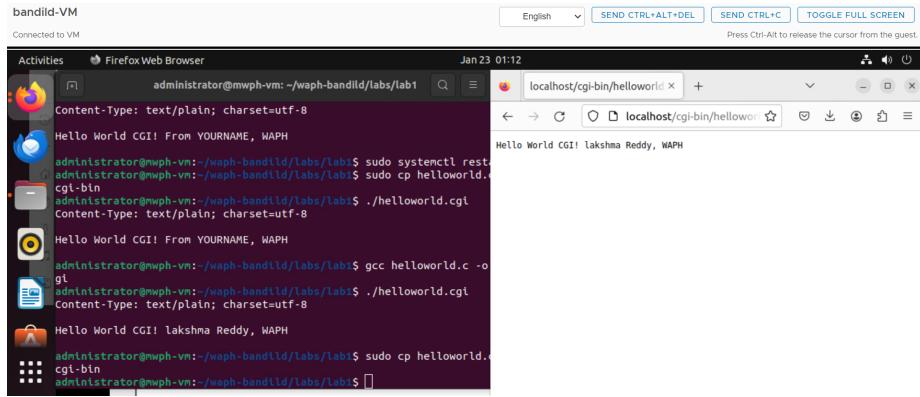
both the HTTP responses in wireshark through browser and TELNET were same.



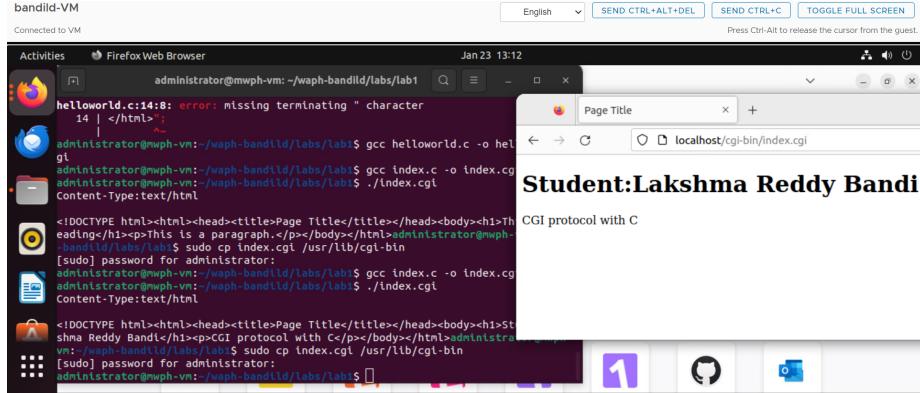
Part II - Basic Web Application Programming

Task 1: CGI Web applications in C

A.CGI program in C, compiled using gcc, was deployed to the browser by copying it to `/usr/lib/cgi-bin` and accessing it in the browser.



B.I created a CGI program in C, incorporating an HTML template with course title, student name, and paragraph details, compiled using gcc, and saved for browser access.

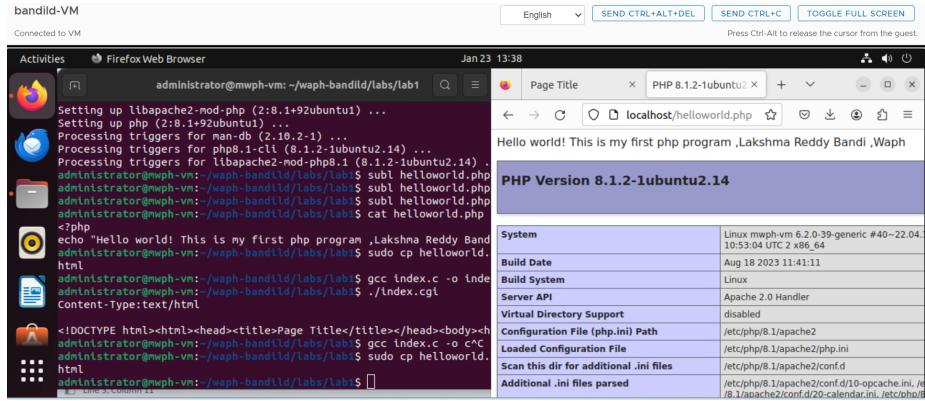


Included file helloworld.c:

```
#include<stdio.h>
int main() {
    const char *htmlContent = "<!DOCTYPE html> <html> <head> <title>page title</title></head> <body> <h1>Student: LakshmaReddy Bandi</h1><p>CGI protocol with c</p></body></html>";
    printf("Content-Type: text/html\n\n");
    printf("%s", htmlContent);
    return 0;
}
```

Task 2: A simple PHP Web Application with user input.

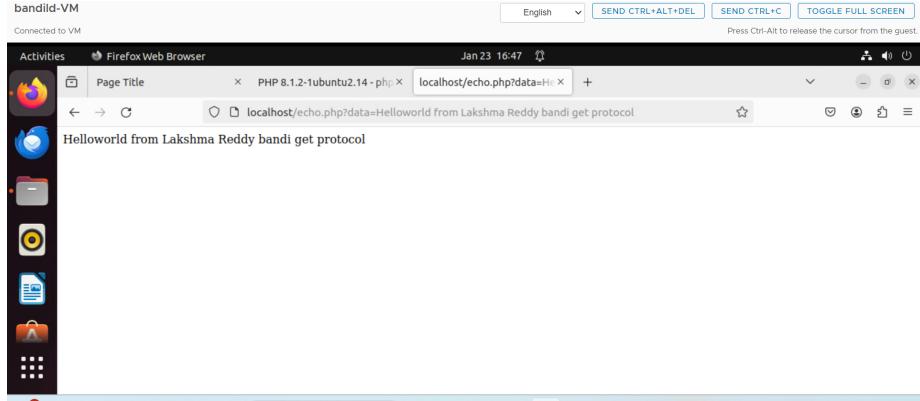
A.A PHP web application was developed, deployed to the root apache2 var/www/html directory, and accessed via the URL IP address/helloworld.php.



Included file `helloworld.php`:

```
<?php
    echo "Hello World! This is my first PHP program, LakshmaReddy Bandi , WAPH";
?>
```

B.PHP's `$_REQUEST` function captures path variables in HTTP requests, but it exposes security flaws like data tampering, SQL injections, and remote code execution. Reducing these risks involves sanitizing user inputs.

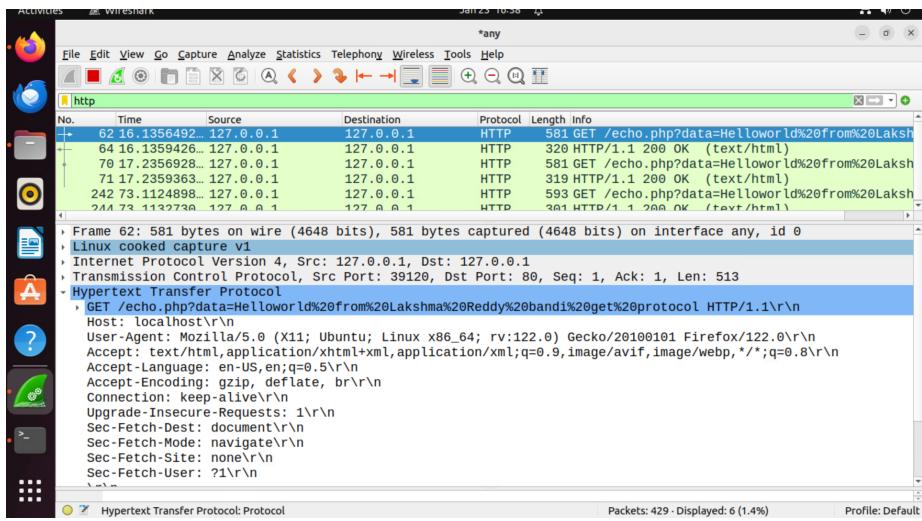


Included file `echo.php`:

```
<?php
    echo "_REQUEST['data']";
?>
```

Task 3: Understanding HTTP GET and POST requests.

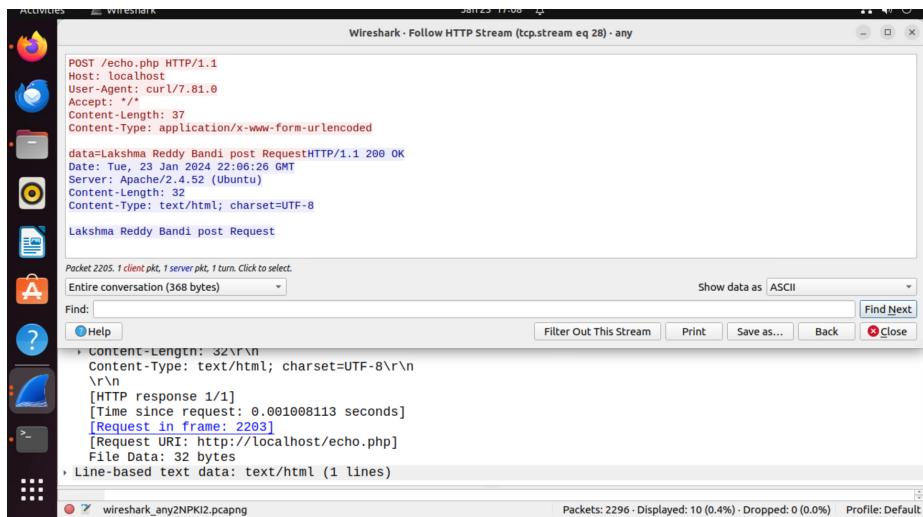
A.The browser's HTTP GET call, passing a path variable, was analyzed using wireshark, revealing the input variable was displayed as part of the response.



In B.A command-line programme called Client URL of CURL is used to process data utilising several n/w protocols. I have made a post request to echo.php using CURL in the terminal.

```
curl -X POST localhost/echo.php -d "data=LakshmaReddy Bandi"
```

```
** (wireshark:4019) 16:56:15.543776 [Capture MESSAGE] -- Capture Start ...
** (wireshark:4019) 16:56:15.006812 [Capture MESSAGE] -- Capture started
** (wireshark:4019) 16:56:15.006860 [Capture MESSAGE] -- File: "/tmp/wireshark_any6TFCI2.pcapng"
** (wireshark:4019) 16:56:34.419197 [Capture MESSAGE] -- Capture Stop ...
** (wireshark:4019) 16:56:34.451841 [Capture MESSAGE] -- Capture stopped.
** (wireshark:4019) 16:56:34.895749 [Capture MESSAGE] -- Capture Started ...
** (wireshark:4019) 16:56:47.993536 [Capture MESSAGE] -- Capture started ...
** (wireshark:4019) 16:56:47.995178 [Capture MESSAGE] -- File: "/tmp/wireshark_any2NPKI2.pcapng"
administrator@wph-vn: ~$ /waph-bandit/labs/lab1 curl -X POST http://localhost/echo.php -d "data=Lakshma Reddy Bandi post Request"
Command 'curl' not found, but can be installed with:
sudo snap install curl # version 8.1.2, or
sudo apt install curl # version 7.81.0-ubuntu1.14
See 'snap info curl' for additional versions.
administrator@wph-vn: ~$ /waph-bandit/labs/lab1$ sudo apt install curl
Reading package lists... Done
Building dependency tree... Done
Reading status information... Done
The following NEM packages will be installed:
  curl
0 upgraded, 1 newly installed, 0 to remove and 29 not upgraded.
Need to get 194 kB of archives.
After this operation, 454 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 curl amd64 7.81.0-ubuntu1.15 [194 kB]
Fetched 194 kB in 0s (632 kB/s)
Selecting previously unselected package curl.
(Reading database ... 33900 files and directories currently installed.)
Preparing to unpack .../curl_7.81.0-ubuntu1.15_amd64.deb ...
Unpacking curl (7.81.0-ubuntu1.15)...
Setting up curl (7.81.0-ubuntu1.15)...
Processing triggers for man-db (2.10.2-1) ...
administrator@wph-vn: ~$ /waph-bandit/labs/lab1$ curl -X POST http://localhost/echo.php -d "data=Lakshma Reddy Bandi post Request"
Lakshma Reddy Bandi post Requestadministrator@wph-vn: ~$ ** (wireshark:4019) 17:06:51.370348 [Capture MESSAGE]
[ ] -- Capture Stop ...
** (wireshark:4019) 17:06:51.403710 [Capture MESSAGE] -- Capture stopped.
administrator@wph-vn: ~$ /waph-bandit/labs/lab1$
```



C.The similarities and differences between HTTP GET/ POST requests and responses
Similarities: Both HTTP techniques are employed in client-server communication. Instruments like as Wireshark can be used to record and examine both. Both have request headers, however they could be different. Responses with headers and status codes are sent to both. Differences: While data is supplied in the HTTP body in a POST request, it is sent in the URL for a GET request. Since data is not accessible to regular users, POST is more secure. Information can be retrieved using GET, while updates can be made using POST.

Because the echo.php web site is a mirror application that is, it simply prints the input it receives the results from the HTTP GET and HTTP POST requests are the same.

The project report was placed in the Labs/Lab1 folder, and the modifications were pushed. The project report was created using the Pandoc tool using the README.md file.