**Simple Network Scanner Web App (nmap)**

# **Introduction**

Network security is a cornerstone of modern IT infrastructure management. Ensuring that all devices on a network are accounted for and secure is crucial for preventing unauthorized access and mitigating potential threats. This project aims to create a simple, automated network scanner web application using Nmap, accessible from any web browser on the network. By scheduling regular scans and presenting the results through a web interface, the project provides an effective tool for network monitoring.

# **Motivation**

Maintaining network security often involves manual scans and checks, which can be labor-intensive and prone to oversight. This project addresses these challenges by automating the scanning process using Nmap, a robust and widely-used network scanning tool. The web application provides a straightforward interface for accessing scan results, making it easier for network administrators and security professionals to keep track of the network’s health and promptly identify issues such as unauthorized devices or open ports.

# Area of Work

The project integrates elements of network administration and web development. Specifically, it utilizes:

* **Cron Jobs** for scheduling automated tasks.
* **Nmap** for executing network scans.
* **PHP** for processing and displaying scan results.
* **Apache** for serving the web application.

This combination allows for continuous, automated network monitoring with a user-friendly web interface for reviewing scan results.

# Software Requirements

To set up this project, the following software needs to be installed on a Linux system:

1. **Apache2**: A web server that serves the PHP scripts and displays the network scan results.

* sudo apt-get install apache2

1. **PHP**: A server-side scripting language used to format and display the scan results.

* sudo apt-get install php

1. **Nmap**: A powerful network scanning tool used to perform the actual network scans.

* sudo apt-get install nmap

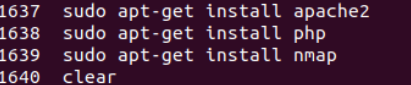
1. **Cron**: A job scheduler used to automate the execution of Nmap scans at regular intervals.

# Setting Up the Project

**Step 1: Install Prerequisites**

Start by installing Apache2, PHP, and Nmap. These are essential components for running the web server, processing scripts, and performing network scans.

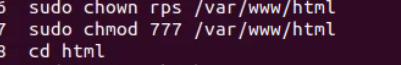
* sudo apt-get install apache2
* sudo apt-get install php
* sudo apt-get install nmap



**Step 2: Configure Ownership and Permissions**

Change the ownership and permissions of the web server’s root directory to facilitate easy access. **Note:** Setting permissions to 777 is insecure for production environments and should only be used in controlled, testing scenarios.

* sudo chown ubuntu /var/www/html
* sudo chmod 777 /var/www/html

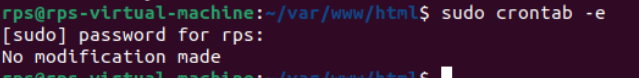


# Step 3: Set Up Cron Job

Configure a Cron Job to automate network scans. This job will run every 10 minutes, scanning the network and saving the results in a text file located in the web server’s root directory.

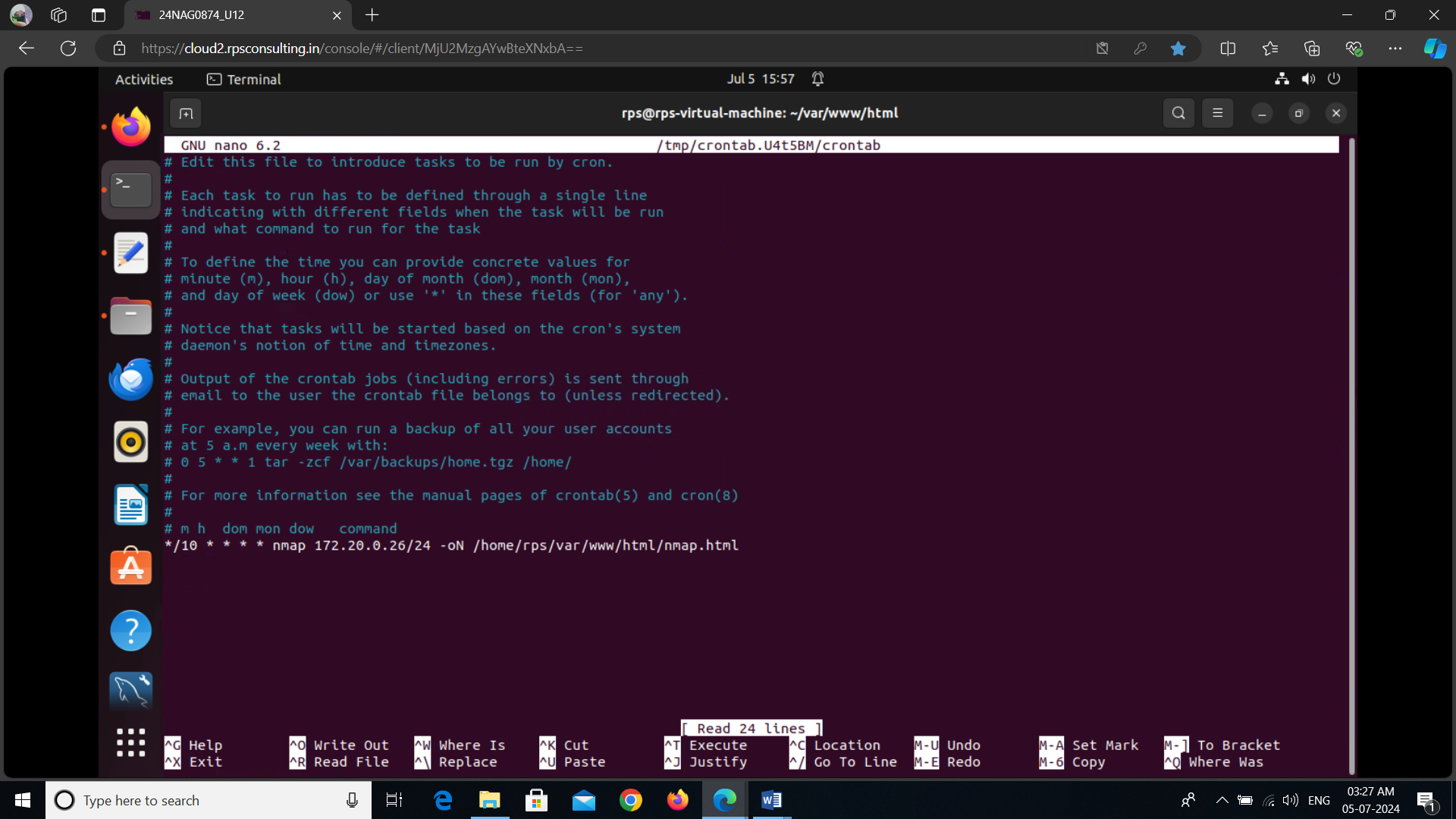
Open the cron table for editing:

* sudo crontab –e



Add the following line to the cron table:

* \*/10 \* \* \* \* nmap 192.168.1.0/24 -oN /var/www/html/nmap.html



This command schedules Nmap to scan the subnet 192.168.1.0/24 every 10 minutes and outputs the results to /var/www/html/nmap.html.

# Step 4: Create the PHP File

Create a PHP file named network.php that reads the scan results and formats them for display in a web browser.

Create and edit network.php in /var/www/html/:\

* <?php

echo "Server Timestamp: ";

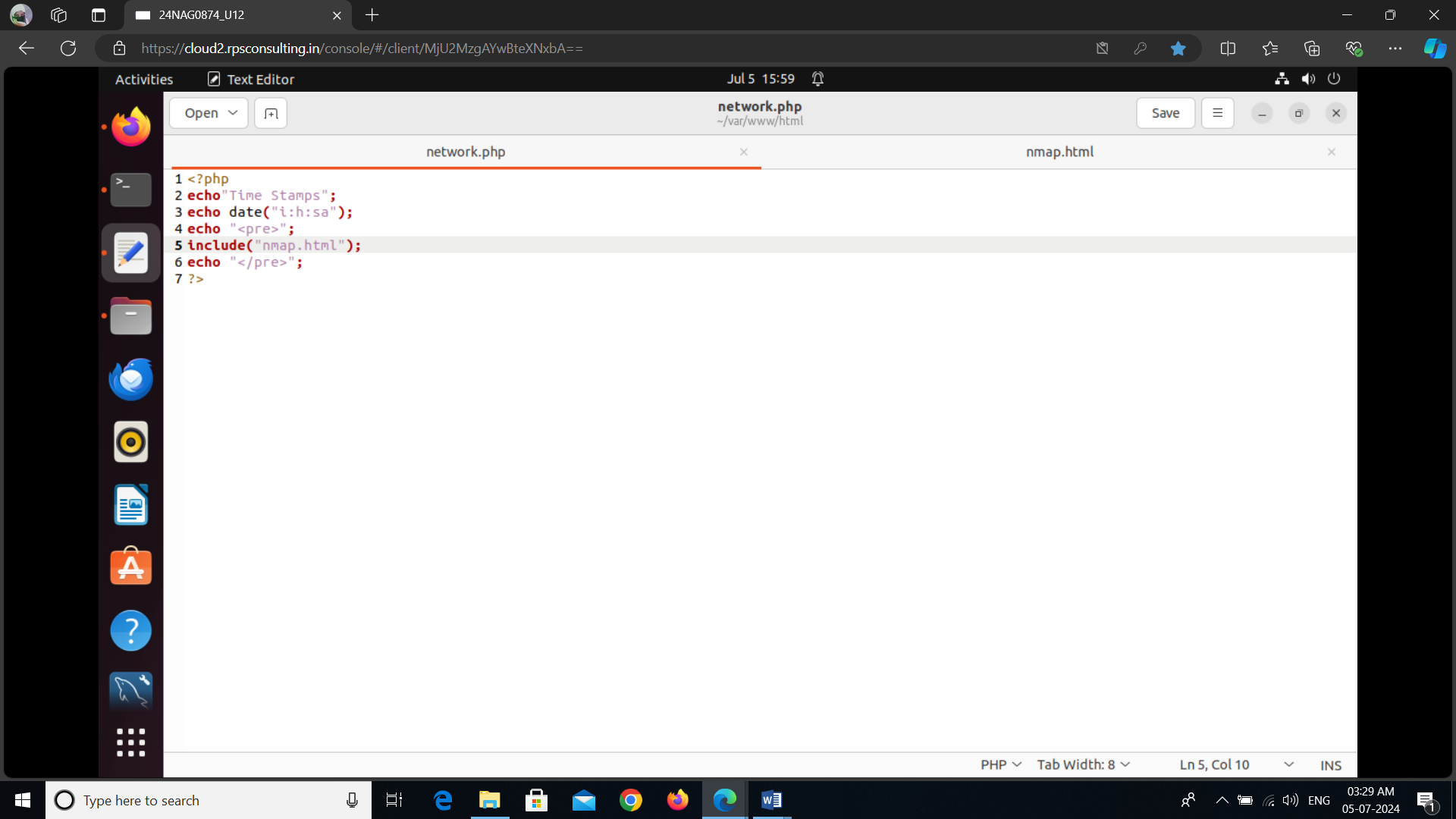
echo date("h:i:sa");

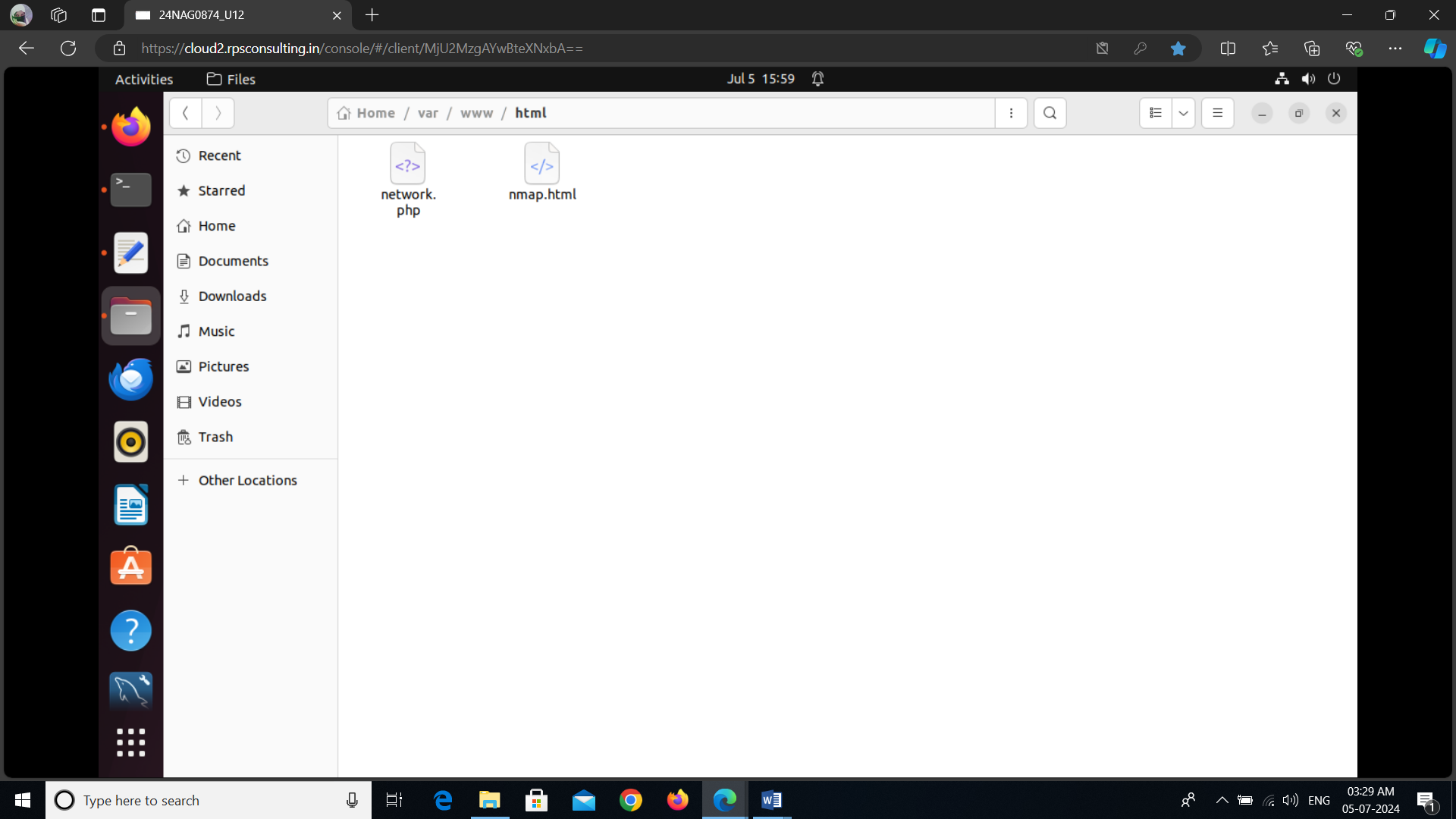
echo "<pre>";

include("nmap.html");

echo "</pre>";

?>





Save this file in /var/www/html/.

# Explanation of the PHP File

The PHP script serves two main functions:

1. **Server Timestamp**: It displays the current server time using PHP’s date function. This provides a reference for when the scan results were last updated, adding context to the displayed information.
2. **Include Nmap Output**: It reads and includes the contents of nmap.html in the web page using the include function. The <pre> HTML tag ensures that the formatting of the text (such as line breaks and spaces) is preserved, making the output easier to read and understand.

# How PHP Works in This Context

PHP is a widely-used server-side scripting language that excels at generating dynamic content. When a user accesses the network.php file via a web browser, the Apache server processes the PHP script. The script fetches the latest scan results from nmap.html and formats them for display in the browser. This allows for real-time or near-real-time updates on the network’s status without manual intervention.

# Understanding Cron Jobs

**Cron** is a time-based job scheduler in Unix-like operating systems, allowing users to schedule tasks (referred to as "cron jobs") to run at specific times or intervals. Cron jobs are defined in a file called the crontab, which specifies the schedule and the command to execute.

# Cron Syntax

Each line in the crontab represents a scheduled task and follows a specific format:

* \* \* \* command\_to\_execute

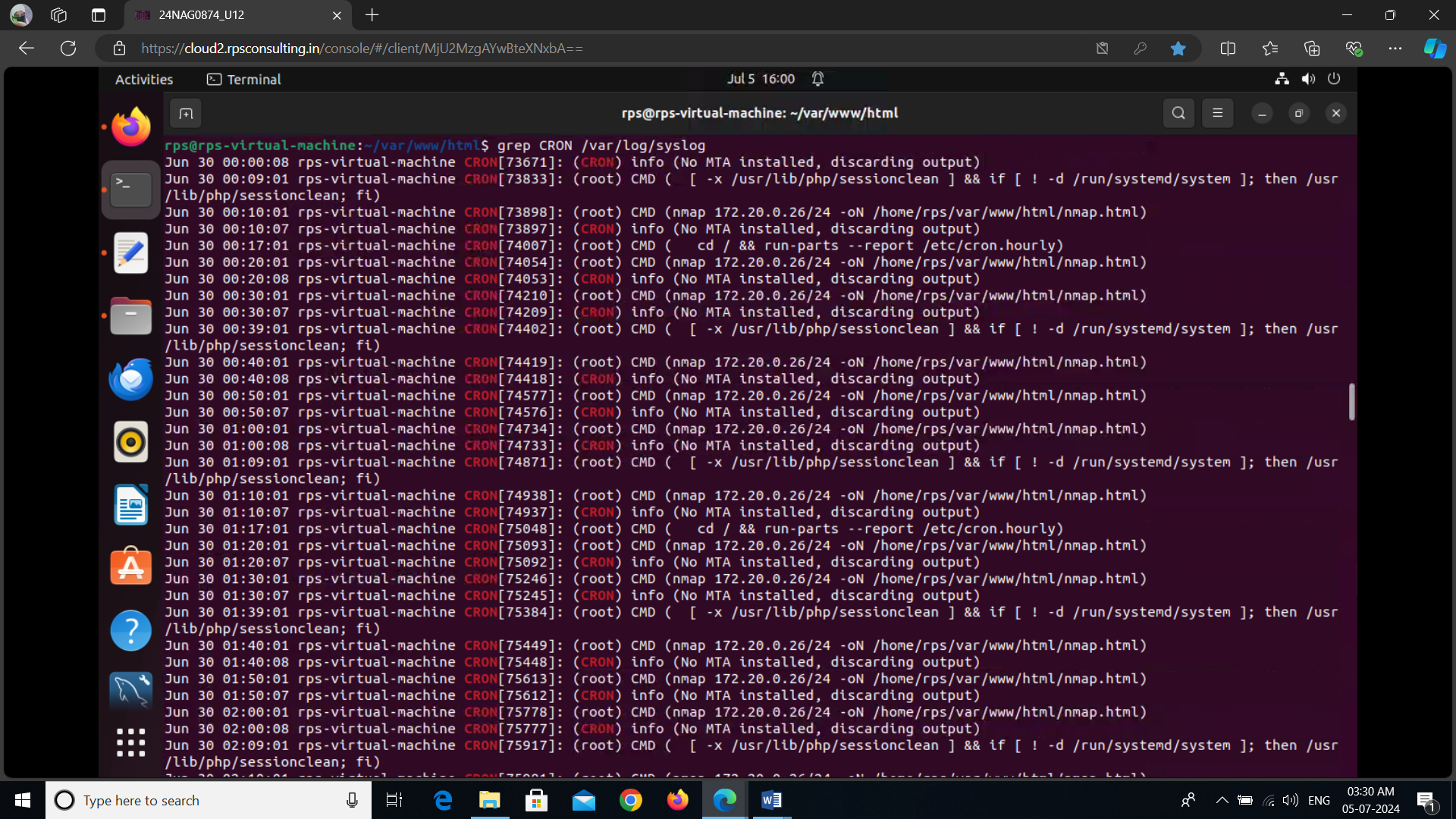
The five fields represent the time and date of execution:

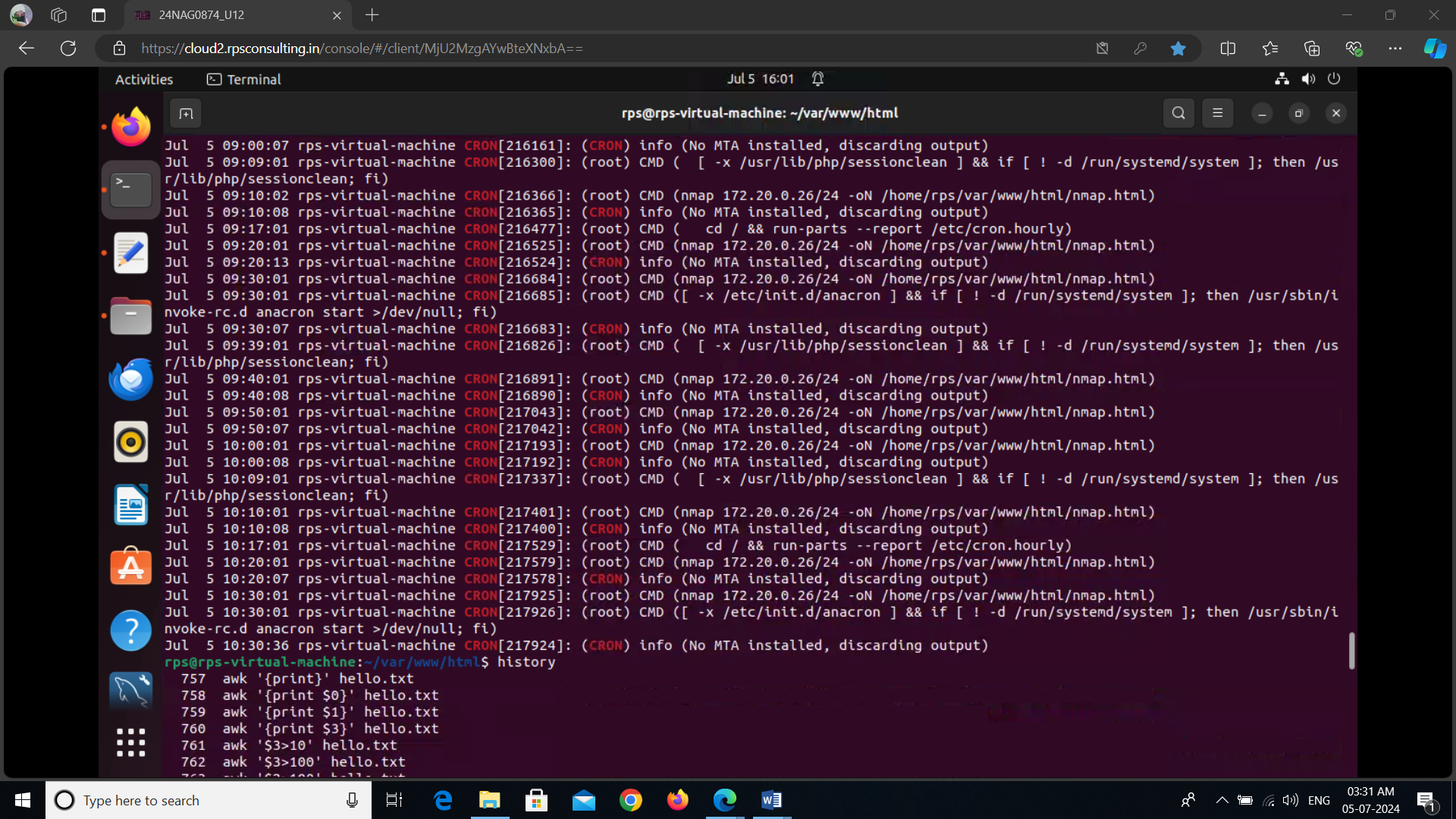
* **Minute**: \*/10 indicates every 10 minutes.
* **Hour**: \* means every hour.
* **Day of Month**: \* means every day of the month.
* **Month**: \* means every month.
* **Day of Week**: \* means every day of the week.

In our project, the Cron job runs Nmap every 10 minutes. It scans the subnet 192.168.1.0/24 and outputs the results to nmap.html, which is located in the web server's root directory. This automation ensures that the scan results are updated regularly without manual intervention, providing up-to-date information on the network's status.

# Benefits of Using Cron Jobs

1. **Automation**: Cron jobs automate repetitive tasks, such as network scanning, which reduces manual effort and minimizes the risk of missing critical updates.
2. **Consistency**: They ensure tasks are executed at regular intervals, providing consistent updates and information.
3. **Efficiency**: Cron jobs can be scheduled during off-peak hours to avoid disrupting regular operations and to ensure resource availability.



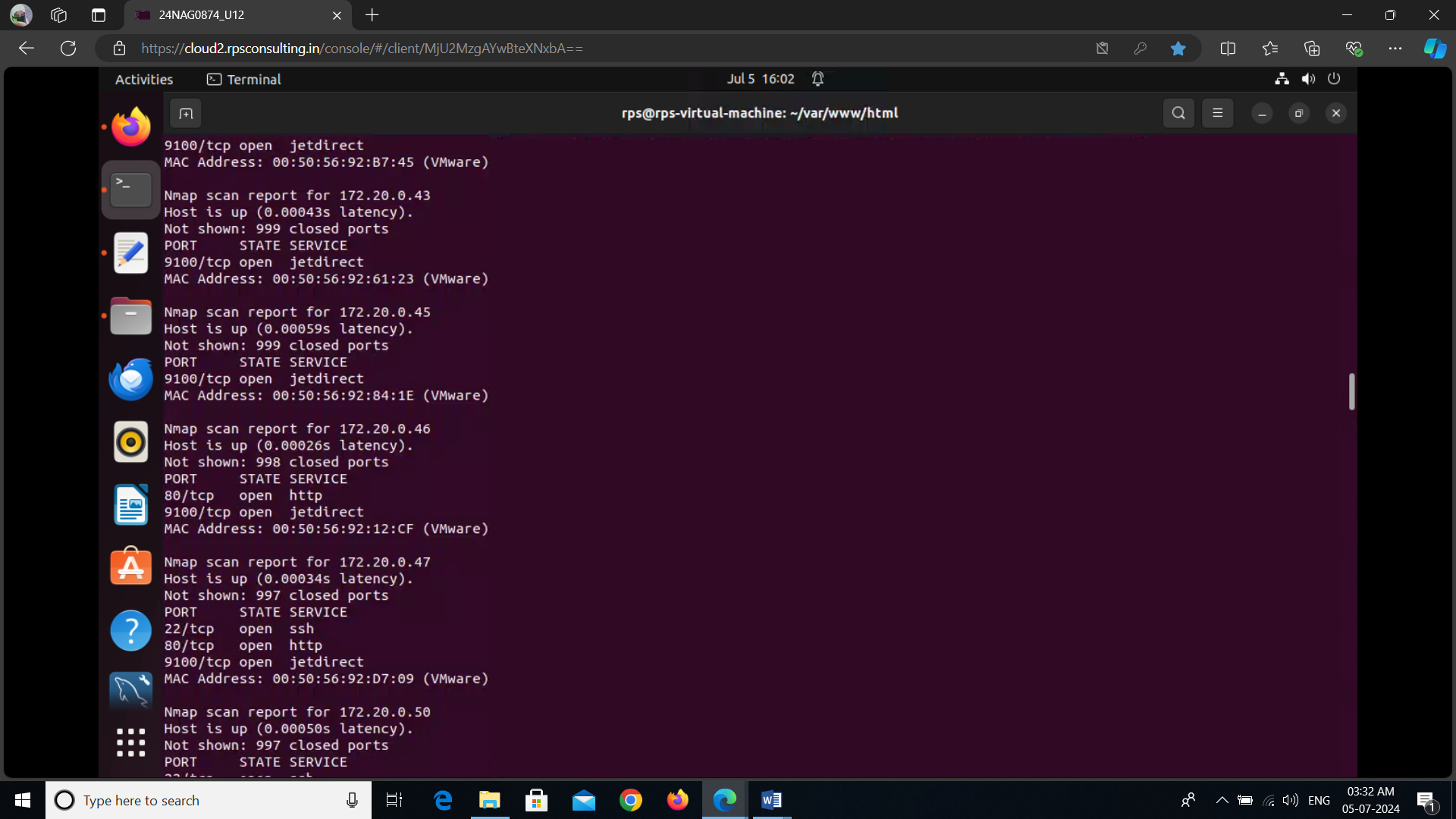


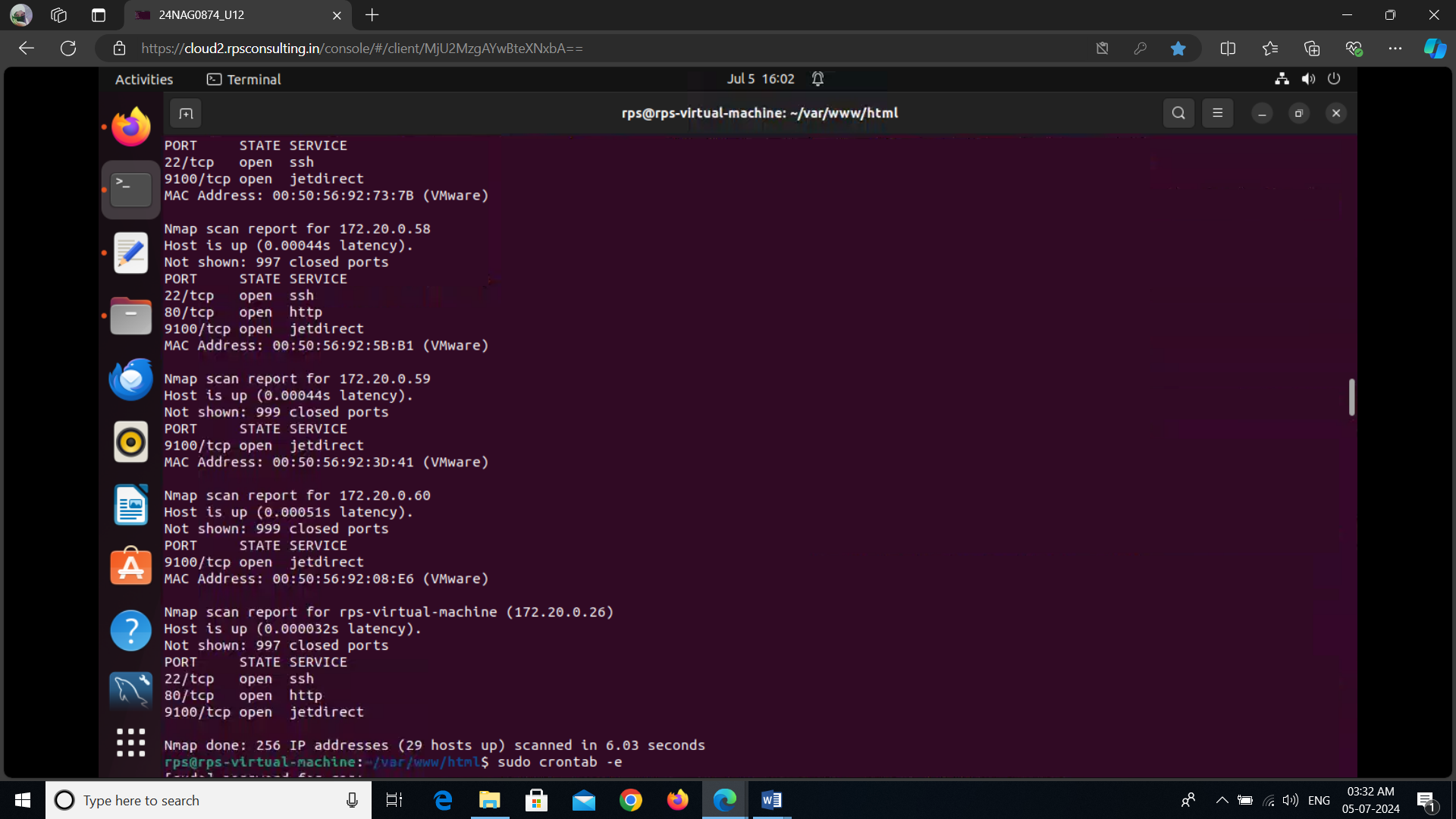
# Final Output

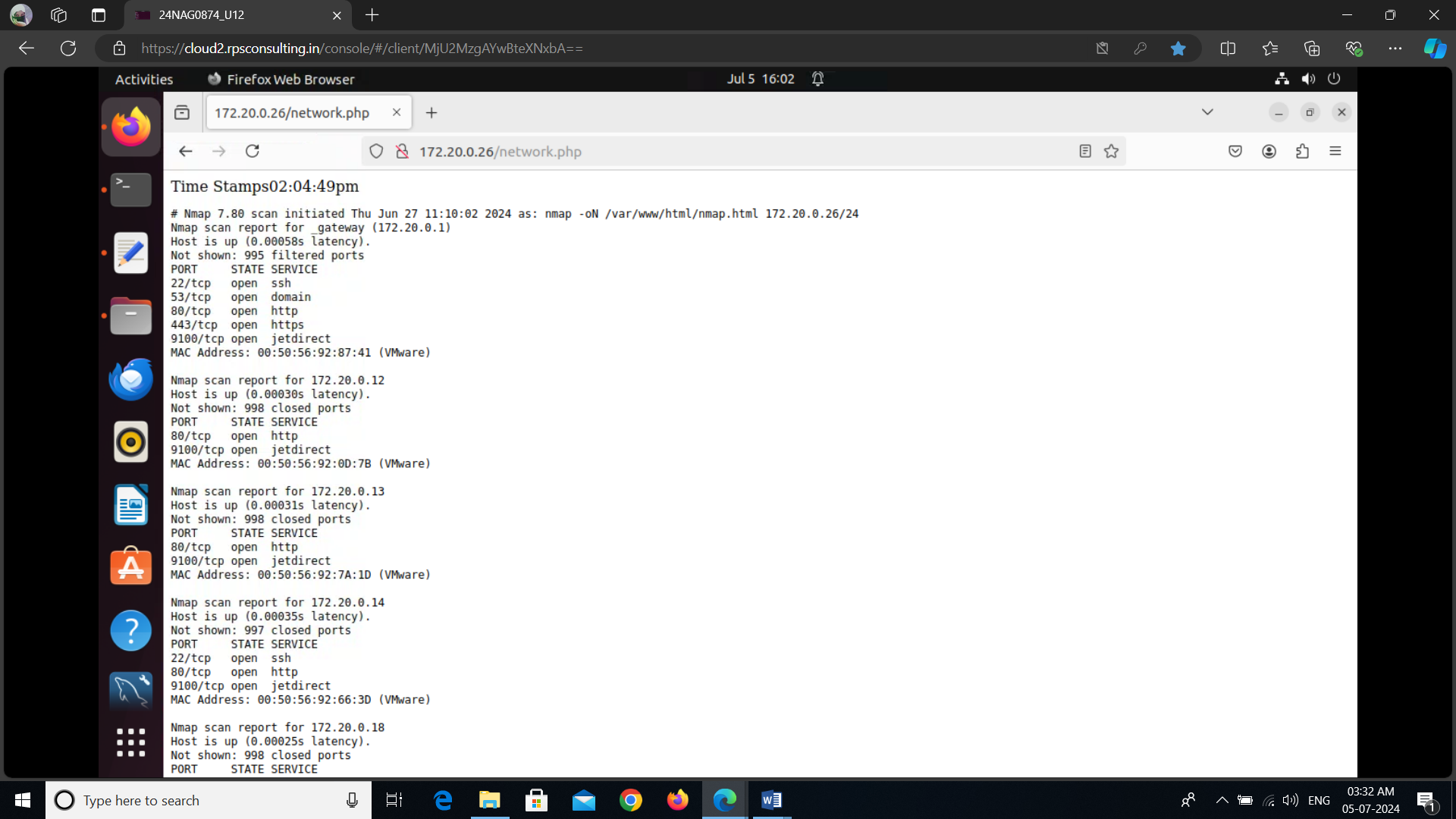
The final output of this project is a web page that displays the results of the Nmap scans in a user-friendly format. When users navigate to the web page hosting network.php, they see:

1. **Server Timestamp**: Displayed at the top, it indicates the current time on the server, providing context for the scan results.
2. **Network Scan Results**: Below the timestamp, the results of the latest Nmap scan are shown. These results include information about active hosts on the network, open ports, and potentially running services.









By providing a web-based interface for viewing scan results, this project simplifies network monitoring and enhances the accessibility of network security data.

Top of Form

Bottom of Form