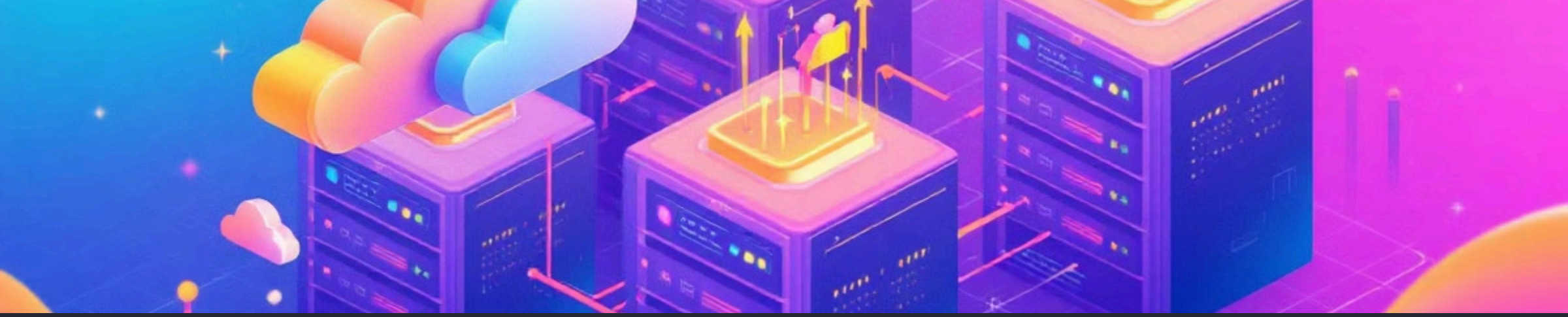
# Advanced Web Server Configuration and Optimization

Welcome to Module 4: Advanced Web Server Configuration and Optimization. In this module, we'll explore essential techniques to enhance the performance, security, and reliability of your web servers. We'll cover load balancing, caching mechanisms, performance monitoring, and best practices for securing web servers.

DK **by Dan K**



# Load Balancing with Apache and Nginx

## What is Load Balancing?

Load balancing distributes incoming traffic across multiple backend servers to improve:

## Performance

Prevents a single server from being overwhelmed.

## High Availability

Ensures uptime if one server fails.

## Scalability

Allows handling of increased traffic.

# Types of Load Balancing

## Round Robin

Requests are distributed sequentially.

## Least Connections

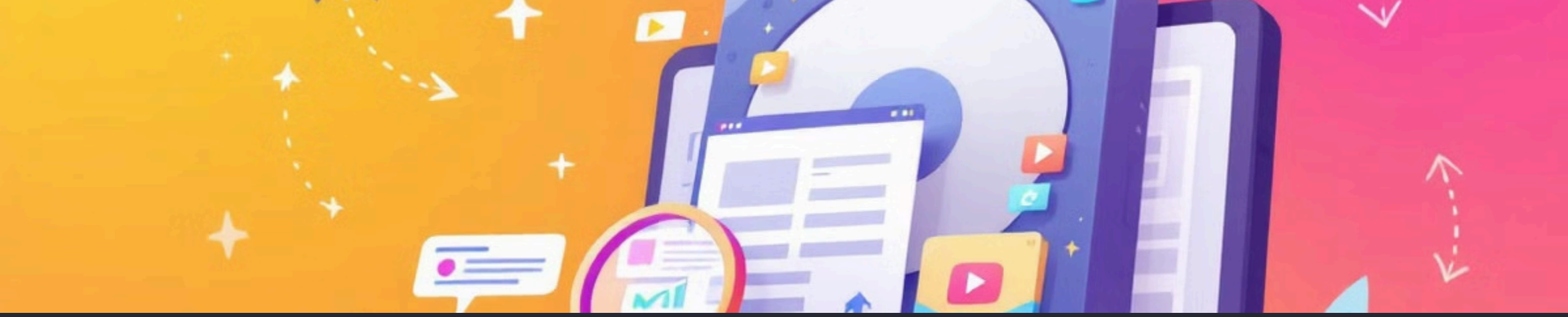
Sends traffic to the server with the fewest active connections.

## IP Hashing

Routes a specific client's request to the same backend server.

Load Balancing with Apache or Nginx.

NOTE: In this course, we shall not cover the setup of Load Balancing. This will be done in subsequent courses.



# Implementing Caching Mechanisms for Improved Performance

## What is Caching?

Caching stores frequently accessed data to reduce load times and server resource usage.

## Types of Caching

- Browser Caching – Stores resources on the client's browser.
- Page Caching – Saves fully rendered pages to serve them faster.
- Object Caching – Stores database query results.
- Opcode Caching – Uses tools like OPcache to store compiled PHP scripts.

# Configuring Caching

NOTE: In this course, we shall not cover the setup of Caching. This will be done in subsequent courses.

# Monitoring and Troubleshooting Web Server Performance

Monitoring helps detect slow response times, high CPU usage, and potential failures.

## Apache Performance Monitoring

- Check Active Connections: apachectl status
- Check Error Logs: sudo tail -f /var/log/httpd/error_log
- Monitor Requests Per Second: ab -n 1000 -c 10 **http://example.com/**

## Nginx Performance Monitoring

- Check Active Connections: sudo nginx -t
- Check Error Logs: sudo tail -f /var/log/nginx/error.log
- Monitor Real-Time Traffic: sudo apt install htop && htop

# Best Practices for Securing Web Servers

1

## Keep Software Updated

sudo apt update && sudo apt upgrade -y (Ubuntu/Debian)

sudo dnf update -y (RHEL/Almalinux/RockyLinux/Centos)

2

## Restrict Access to Sensitive Directories

apache

<Directory /var/www/>

Options -Indexes

3

## Enable a Web Application Firewall (WAF)

Install ModSecurity for Apache:

sudo apt install libapache2-mod-security2 -y

Install Fail2Ban for Nginx:

sudo apt install fail2ban -y

4

## Disable Unnecessary Modules

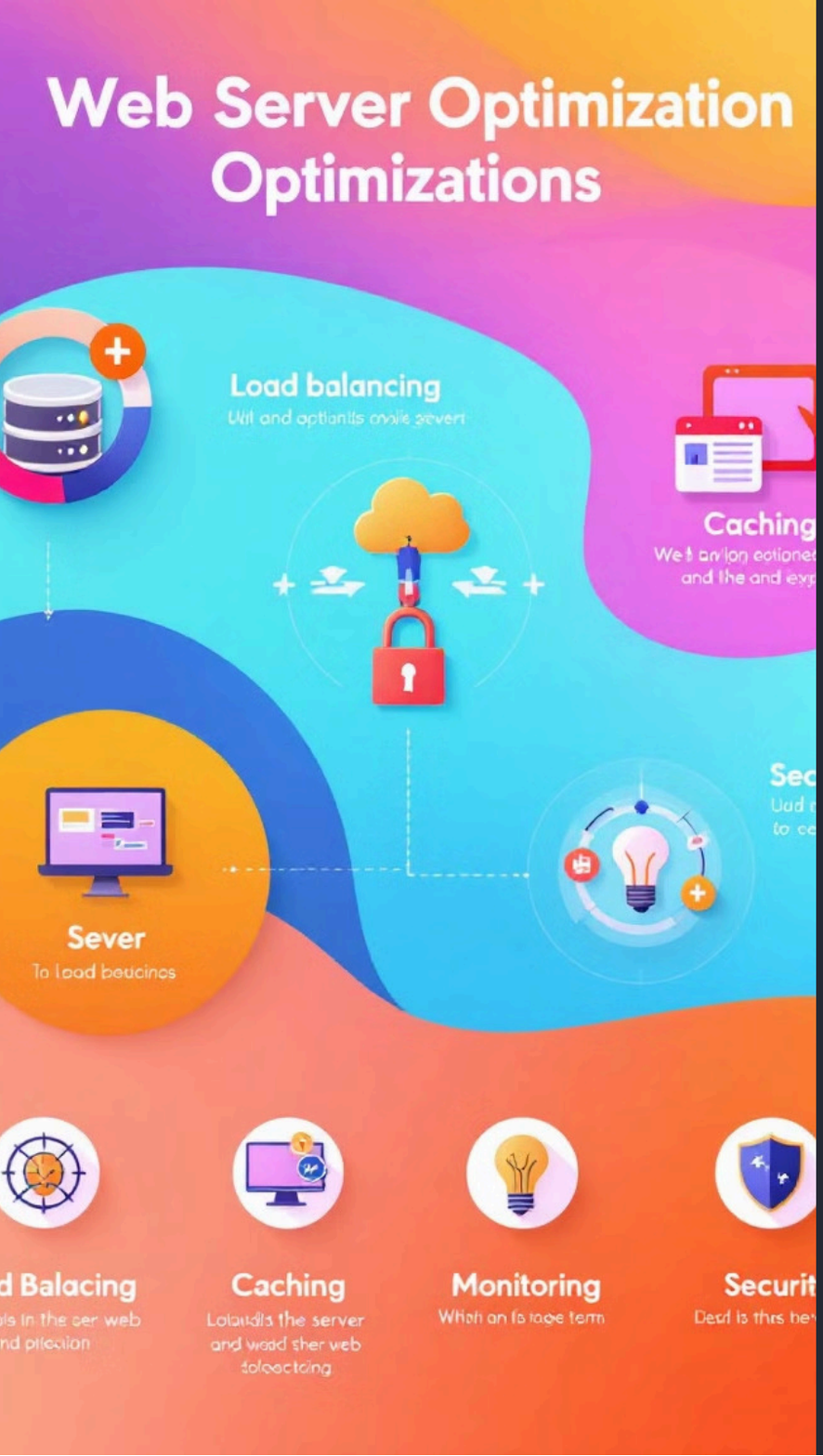
sudo a2dismod autoindex

sudo systemctl restart apache2


# Using Secure Headers

Edit the Apache or Nginx configuration:

```
Header always set X-Frame-Options "SAMEORIGIN"
Header always set X-Content-Type-Options "nosniff"
Header always set Content-Security-Policy "default-src 'self'"
```




# Summary

1

## Load Balancing

Improves availability and performance in Apache and Nginx.

2

## Caching

Reduces server load and speeds up page loads.

3

## Monitoring

Tools help detect performance bottlenecks.

4

## Security Best Practices

Protect against attacks.

# Conclusion

This concludes Module 4: Advanced Web Server Configuration and Optimization. You've learned about crucial techniques to enhance your web server's performance, reliability, and security. Remember to apply these practices in your own server configurations to ensure optimal performance and protection against potential threats.

***The End.***