Name: ADITYA BAHETI
Reg. No:22BCE10521
Department: BTECH
Course: COMPUTER SCIENCE AND ENGINEERING

Project Title

Web Vulnerability Scanning Use Case

Problem Statement / Use Case

A company has recently developed an internal web application to manage sensitive employee data and internal communications. Before deploying this application in a production environment, it is critical to ensure that no basic vulnerabilities exist that could be exploited by internal or external attackers. This project simulates a basic vulnerability scan using common tools to uncover issues such as exposed admin panels, outdated software, and misconfigurations that could pose security risks.

Project Objective(s)

Some of the objectives for the projects are:

- To identify common misconfigurations or exposed sensitive paths in the internal web application.
- To detect outdated or vulnerable software versions that the app might be using.
- To analyze the technologies behind the application and how they may influence security posture.
- To provide simple yet effective suggestions to secure the application before going live.

Tools and Technologies

I used a few popular open-source tools available on Kali Linux to perform the scans:

- **Nikto** – to perform a basic vulnerability scan on the web server.
- **Dirb** – to brute-force hidden directories and find admin panels or backup folders.
- **WhatWeb** – to identify the technologies powering the web application, such as server type, CMS, or frameworks.

Methodology / Approach

The process I followed was straightforward:

1. I started by launching the internal web application in a secure lab environment and made sure it was accessible via Kali Linux.

2. I used **WhatWeb** first to understand the technologies behind the web application — this helped me learn if it was running on Apache, Nginx, or IIS and what programming languages or plugins were used.

3. Next, I used **Dirb** with a default wordlist to scan the application for any hidden directories like `/admin`, `/test`, `/backup`, etc. This is important because such directories often contain sensitive information or entry points.

4. Finally, I ran **Nikto** against the site to check for common vulnerabilities, outdated server components, insecure headers, or known exploits related to the technologies used.

5. All results were carefully reviewed, and I documented my findings, including screenshots and logs of potential vulnerabilities.

6. Based on the results, I listed a few recommendations for improving the application's security.

Innovation & Uniqueness

What I liked most about this project is that it doesn't rely on heavy enterprise tools — just some simple open-source utilities that anyone with basic Linux skills can run. It's practical and can easily be applied to real-world use cases. The uniqueness lies in its low-cost, hands-on approach to testing internal applications that are often assumed to be "safe" just because they're not publicly accessible.

Relevance to Cybersecurity Field

This project helped me build confidence in using essential cybersecurity tools and taught me how to think like a security analyst. Scanning web applications for vulnerabilities is one of the core skills in penetration testing, ethical hacking, and even defensive security roles. The work aligns closely with real-world practices taught in certifications like CEH (Certified Ethical Hacker) and OSCP (Offensive Security Certified Professional). It also helped reinforce the importance of always testing applications — even internal ones — before deployment.

Expected Deliverables

At the end of this project, I compiled the following deliverables:

- Screenshots showing the outputs from **WhatWeb**, **Dirb**, and **Nikto**.
- A written report detailing the vulnerabilities found and what each one means.
- Suggested remediation steps for each issue discovered.
- (Optional) A short video walkthrough or screen recording of the scanning process, if required.