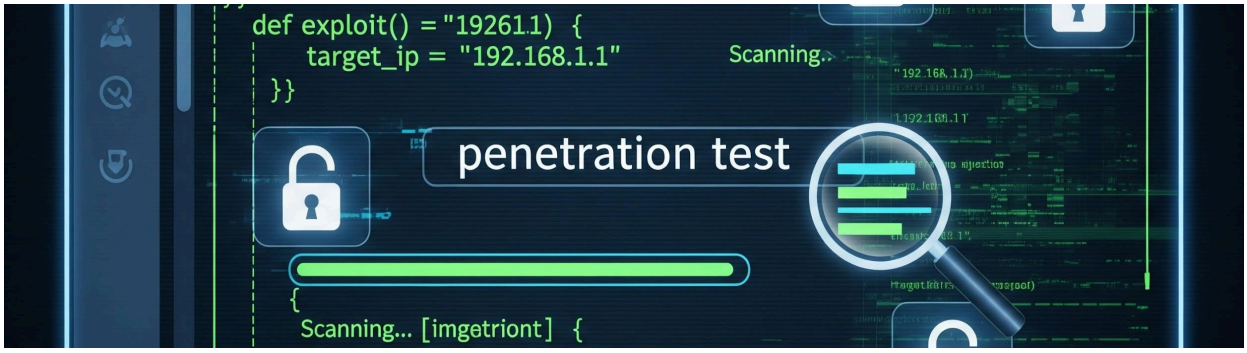

Penetration Testing Report for HackThisSite

Executive Summary

This report details the findings and recommendations from a penetration test conducted on the HackThisSite website (hackthissite.org). The primary goal of this assessment was to identify potential security vulnerabilities that could be exploited by malicious actors. The testing focused on common web application security flaws and network configuration issues.

Overall, the assessment identified several areas of concern, including a critical vulnerability in the challenge section, which could potentially lead to unauthorized access or data compromise. Immediate remediation is strongly recommended for the high and critical severity findings.

Scope of Assessment

The penetration test was conducted against the publicly accessible web application hosted at hackthissite.org. The scope was limited to the "Basic Missions" challenge section, specifically levels 1 through 11, as these levels comprehensively demonstrate fundamental security weaknesses. The assessment included:

- **Vulnerability Identification:** Testing for common flaws such as input validation issues, misconfigurations, and improper handling of client-side data.
- **Authentication/Authorization Mechanisms:** Examining challenge-specific login and access control mechanisms where applicable.

- **Client-Side and Server-Side Interaction:** Analyzing how the application handles data submitted from the user's browser, including URL parameters, form data, and cookies.

The testing did not extend to the site's underlying infrastructure, internal network components, or other sections of the website (e.g., forums, other challenge categories).

Vulnerability Description and Key Findings

Level	Name	Description
1	**The Idiot Test**	The password is "hidden" in the HTML code itself. It teaches you how to use "View Source" in your browser.
2	**Network Security**	The administrator forgot to upload the password file, but the script still looks for it. It teaches how server-side misconfigurations can lead to bypasses.
3	**The Password File**	The password is stored in a hidden file on the server. This level introduces the concept of file paths and looking for common "hidden" directories or files.
4	**Mail To**	A form sends the password to an email address. You must intercept and modify the HTML (specifically a hidden input field) to send the email to yourself instead.
5	**The Email Change**	Similar to Level 4, but involves a script that emails the password to the administrator. You must manipulate the form to change where the automated email is sent.

6	**Encryption**	You are given a password that has been encrypted with a simple algorithm. This level is a primer on cryptography and pattern recognition (specifically a stream cipher).
7	**The Unix Command**	This level introduces Command Injection. You are given a tool that executes a system command (like `cal` for a calendar), and you must trick it into running extra commands (like `ls`).
8	**SSI Injection**	Focuses on Server Side Includes (SSI). You must use a guestbook-style input to inject a command that reveals files on the server using the `` tag.
9	**The Scrambled Script**	A test of your ability to read and understand obfuscated JavaScript. You have to reverse-engineer a script to figure out what the correct password should be.
10	**Cookies**	This level teaches about Cookie Manipulation. The site uses a cookie to determine if you are an "authorized" user. You need to modify your browser's cookies to gain access.
11	**The Hidden Directory**	A more advanced version of Level 3. It involves finding a hidden directory by observing server behavior or hints left in configuration files (like `.htaccess`).

Recommendations on How to Better Secure the Web Application

The basic HackThisSite challenges highlight foundational web application security weaknesses, primarily emphasizing the necessity of robust, server-side defenses. Key recommendations include implementing strict input validation and context-aware output encoding to prevent injection flaws like Command and SSI Injection (Levels 7, 8) and Cross-Site Scripting. Secure configuration and adherence to the principle of least privilege are critical to prevent unauthorized access to sensitive files and directories (Levels 2, 3, 11). Furthermore, developers must never store secrets in client-side code (Level 1), rely on strong session

management (Level 10), avoid basing critical functionality on easily manipulated client-side controls (Levels 4, 5), and utilize established cryptographic libraries instead of custom or simple encryption methods (Level 6) to build resilient and secure applications.