Password Storage Lab: Breaking and Fixing Weak Hashing Systems

Idea Overview

This lab explores the security implications of password storage mechanisms, contrasting weak hashing practices with modern, secure alternatives like Argon2 and scrypt. Students will learn how poor hashing choices can lead to password recovery via rainbow tables or brute force attacks, and how to harden systems against these threats.

The lab simulates a vulnerable login system and guides students through identifying weaknesses, exploiting them, and then implementing secure fixes

Didactic Objective

Targeted at final year Computer Science or MSc students in security focused courses, this lab helps students:

- Understand the difference between hashing and encryption
- Analyze password storage vulnerabilities
- Use rainbow tables and brute force tools to recover weakly hashed passwords
- Implement salting and modern hashing algorithms (Argon2, scrypt)
- Evaluate the impact of these defenses on attack feasibility

Background and Motivation

Real-world breaches often stem from poor password storage practices. Examples include:

- Storing unsalted MD5 or SHA1 hashes
- Using fast hash functions without key stretching
- Reusing predictable salts or none at all
- Failing to upgrade legacy systems

This lab simulates these pitfalls and guides students toward secure design.

Lab Environment

Runs on SEED Labs VM.

Components:

- A login system with configurable hashing (MD5, SHA1, bcrypt, Argon2, scrypt)
- A rainbow table generator
- Scripts to simulate brute force attacks
- A challenge system: retrieve the password or flag from each vulnerable configuration

Possible Scenarios:

1. Login system using unsalted MD5
2. Login system using salted SHA1
3. Login system using bcrypt with low cost factor
4. Login system using Argon2id with proper parameters
5. Login system using scrypt with memory-hard settings

Student Tasks

Students must:

- Analyze how passwords are stored and verified
- Identify weaknesses in the hashing scheme
- Use rainbow tables or brute force tools to recover passwords
- Modify the code to add salts and upgrade to secure hashing
- Re-test attacks and observe increased resistance

Expected Learning Outcomes

By the end of the lab, students will be able to:

- Explain why hashing is preferred over encryption for password storage
- Identify insecure hashing practices
- Demonstrate password recovery using rainbow tables
- Implement salting and modern hashing algorithms
- Evaluate the effectiveness of stronger hashing methods

Implementation Approach

- Base environment: SEED Labs VM
- Docker containers for vulnerable servers
- A lightweight web application framework for the login system with pluggable hash models

- Tools: Hashcat, custom rainbow table scripts, bcrypt, argon2 and scrypt libraries
- Flags: Stored in protected resources accessible only after successful login