Lab 3: Building an OTP Authentication Server

Course: FSCT 8561 – Security Applications
Instructor: Dr. Maryam R. Aliabadi
Lab Duration: 3 Hours

Overview

This lab introduces authentication mechanisms used in secure networked systems, with a focus on password hashing and One-Time Passwords (OTP). Students will design and implement a simple client–server authentication system that combines static password verification with Time-Based One-Time Passwords (TOTP). The lab emphasizes secure credential handling, attack mitigation, and correct authentication workflows.

While the textbook covers static HTTP methods like Basic and Digest Auth, this lab focuses on dynamic, time-based authentication, showing how hashing and OTPs can mitigate threats like replay attacks and credential theft in Python beyond HTTP protocols.

Learning Objectives

- Understand the role of authentication in network security
- Apply secure password hashing using Python hashlib
- Generate and verify OTPs using the PyOTP library
- Implement a basic OTP-based authentication server
- Analyze security benefits and limitations of OTP-based authentication

Pre-requisite

- Completed Lab1 and Lab2

Required Reading & Tutorials

- Mastering Python for Networking and Security – **Chapter 4**
    - https://learning.oreilly.com/library/view/mastering-python-for/9781839217166/
    - Source code on: https://github.com/PacktPublishing/Mastering-Python-for-Networking-and-Security-Second-Edition
- Python hashlib documentation : https://docs.python.org/3/library/hashlib.html
- PyOTP Documentation : https://pyauth.github.io/pyotp/
- **getpass module** for secure password input: https://docs.python.org/3/library/getpass.html

Lab Scenario

You are tasked with implementing a secure authentication service for a networked application. Users must authenticate using a username, password, and a time-based one-time password (TOTP). The server validates credentials and grants or denies access accordingly. The goal is to demonstrate a secure authentication workflow and analyze its security properties.

Part 1 – Password Hashing

Implement secure password storage using cryptographic hash functions. Passwords must never be stored or transmitted in plaintext.

- Use hashlib with SHA-256 or stronger
- Store only password hashes on the server
- Verify passwords by comparing hashes

Part 2 – OTP Secret Generation

Generate a unique OTP secret for each user using PyOTP. This secret is shared between the client and server.

- Generate base32 secrets
- Associate secrets with user accounts
- Display provisioning URI (optional)

Part 3 – OTP Authentication Flow

Implement the authentication workflow combining password and OTP verification.

Workflow steps:

1. User submits username and password
2. Server verifies password hash
3. User submits OTP
4. Server verifies OTP

Part 4 – Client Server Interaction

Implement a simple client to interact with the authentication server.

- Prompt user for credentials
- Send authentication data to server
- Display server responses clearly

Part 5 – Security Testing

Test your authentication system against common failure and attack scenarios. You must demonstrate handling of the following:

- Incorrect password
- Expired or invalid OTP
- Repeated failed attempts
- Clock desynchronization

Part 6 – Reflection Questions

1. Why is hashing required for password storage?
2. How does OTP mitigate replay attacks?
3. What happens if the client and server clocks are not synchronized?
4. What are the limitations of OTP-based authentication?

Part 7 – Security Analysis

In 300–400 words, analyze the security implications of your authentication system. Your analysis must reference specific implementation details and address:

- Threats mitigated by OTP
- Remaining attack vectors
- Why authentication alone does not provide full security
- Suggested improvements for real-world deployment

Deliverables

- `Auth_server.py`
- `Auth_client.py`
- A recording demonstrating successful and failed authentication as well as security testing (mp4 format)
- Security analysis and reflection report
- Submit **one PDF file only**. This PDF is the Security analysis and Reflection Report and must include cloud links to all required technical artifacts (code base, recordings.).
- Filename format:

  Lab3-FirstName-LastName-StudentNumber.pdf

Good Luck!

Lab 3 Grading Rubric (Total: 100 points)

Component	Points	Full Credit	Partial Credit	Minimal / No Credit
Password Hashing Implementation	20	Correctly hashes passwords using SHA-256 or stronger; stores only hash; compares hashes for verification	Minor errors in hash comparison; partial security (e.g., prints passwords)	Passwords stored or transmitted in plaintext; hashing not implemented
OTP Secret Generation	15	Generates unique base32 secret for each user; associated correctly in user DB	Secret generated but misassigned; minor errors in storage	Secret missing or incorrect; OTP unusable
OTP Verification	20	Server correctly verifies OTP using TOTP; rejects invalid OTP; proper error messages	OTP verification works but inconsistent handling; minor errors in messages	OTP verification missing or incorrect; server accepts invalid OTP
Client-Server Interaction	15	Client prompts correctly for username/password/OTP; sends data; prints server responses; clean exit	Client works but may mishandle inputs or display errors	Client does not function or fails to connect
Server Robustness / Error Handling	15	Handles invalid usernames, passwords, OTPs; server does not crash; multiple clients supported	Handles some errors but crashes on edge cases	Server crashes on invalid input; no multi-client support
Security Analysis & Reflection	15	Clearly explains security implications, threats mitigated, remaining risks, and improvement suggestions; 300–400 words	Partial coverage of analysis; minor missing points	Analysis missing, generic, or does not reference implementation