Yvette Kushmerick
Student ID: g38491974
24 May 2025
SEAS 8405 Cyber Architectures

Introduction

This report provides details on designing and implementing an Identity and Access Management (IAM) architecture using the identity provider, Keycloak, to protect a Flask application for a Docker Container environment. It is divided into two parts: Part 1: Environment Setup and Configuration, and Part 2: Security Analysis and Best Practices.

Part 1: Environment Setup and Configuration

I **c**reated a directory folder as iam-demo then created the following directory within:

- Flask-app folder containing app.py, Dockerfile, requirements.txt
- Keycloak folder containing create-realm.sh
- Docker-compose.yml file
- Makefile.txt

The Keycloak authentication server is running on localhost:8080 with a custom realm, client, and user. The Flask App API is running on localhost:5000 with a protected endpoint. The Flask App verifies a token signed by Keycloak using a public key. Keycloak is configured as a standalone server running in a Docker container on the same machine (with realm, client and user). OpenID Connect is an authentication standard built on top of OAuth 2.0, which defines an ID token type to pair with OAuth 2.0 access and refresh tokens (Okta, 2025). For Keycloak, the admin user's password is admin. The uniquely created User is Test User within the created Realm, demo-realm. The demo-realm username is testuser with ID: 0dbfefb8-9c59-4364-9808-71a8fb97df6b. The password is testpass. The Client ID is testuser at Test User using OpenID Connect. The home URL is http://localhost:8080/realms/Demo-realm/account/. Figure 1 below depicts the Docker Container architecture with Keycloak IAM.

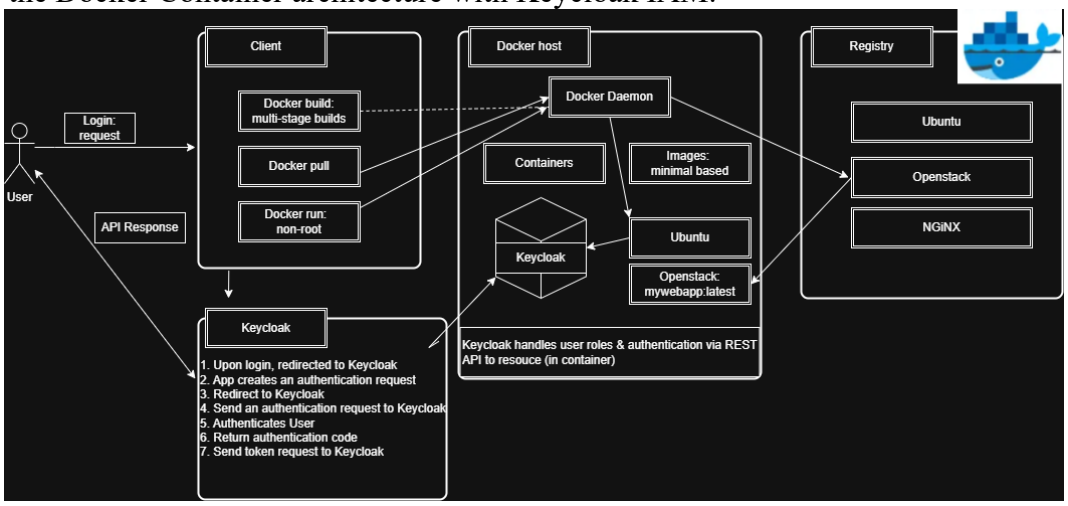

Figure 1. Docker Container with Keycloak (created by author).

Part 2: Security Analysis and Best Practices

Security Analysis

Using Docker Scout to conduct a vulnerability assessment revealed there were fifty-one vulnerabilities, of which eleven were high and all related to the version of Keycloak [24.0.1]. The highest severity vulnerability pertained to URL redirects. The 24.0.1 version of Keycloak did not properly validate URLs in redirects, possibly allowing an attacker to use the flaw to construct a malicious request to bypass validation and access other URLs and potentially sensitive information within the domain. This would allow the attacker to possibly conduct further attacks. Additionally, in the initial scan, there were thirty-four vulnerabilities discovered on the iam-demo-flask-app:latest, of which three where high, four were medium and the rest (26) are low in the severity scale. After reviewing CVE-2022-40897, CVE-2025-47273, and CVE-2024-6345, I changed the Python version in the Dockerfile from 3.9 slim to 3.13 slim. The below graphic depicts the vulnerabilities discovered to the Flask App using Python version 3.9 slim.

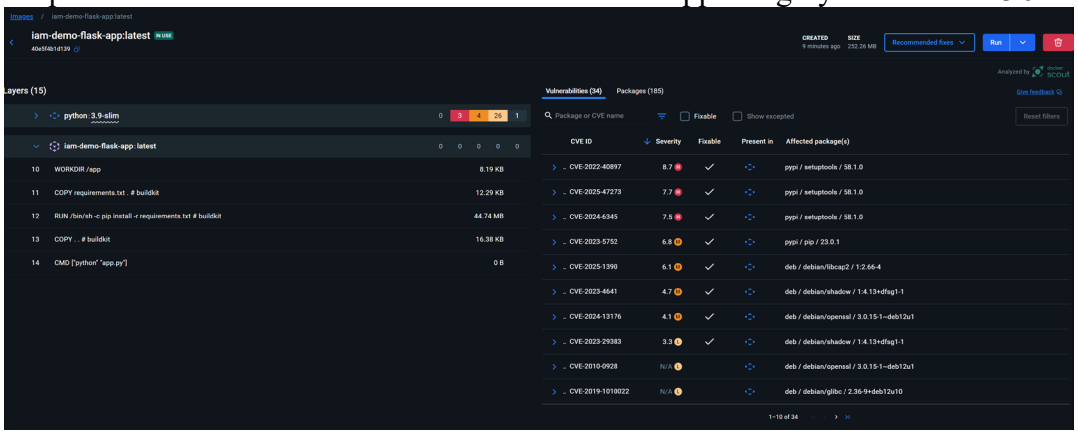

Figure 2. Vulnerabilities to Flask App (Docker Scout).

Figure 3 below shows a Docker Scout scan of the iam-demo-flask-app after changing the Python version from 3.9 slim to 3.13 slim, which depicts only twenty-five low severity vulnerabilities.

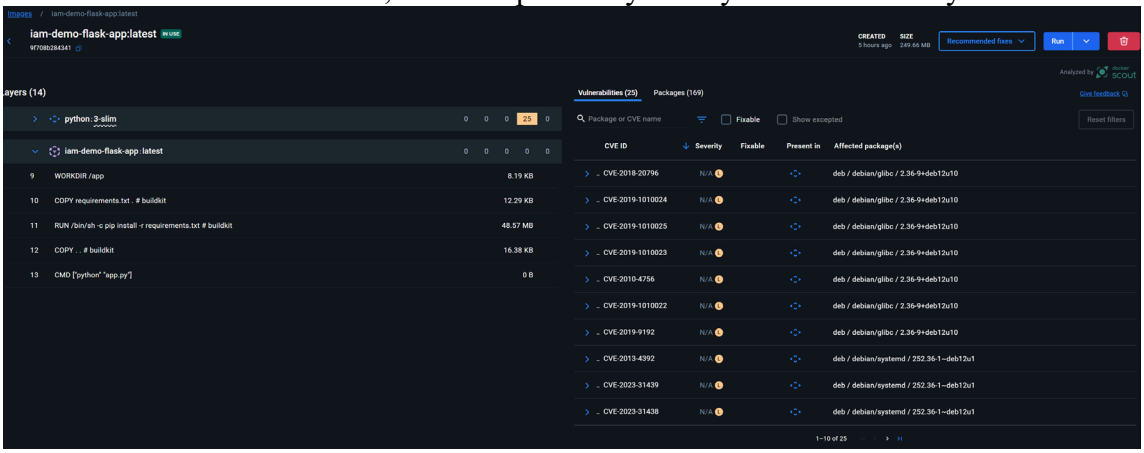

Figure 3. Vulnerabilities to Flask App (Docker Scout).

Finally, the below graphic depicts a Docker Scout scan of the original image with Keycloak version 24.0.1 included in the docker-compose.yml file.

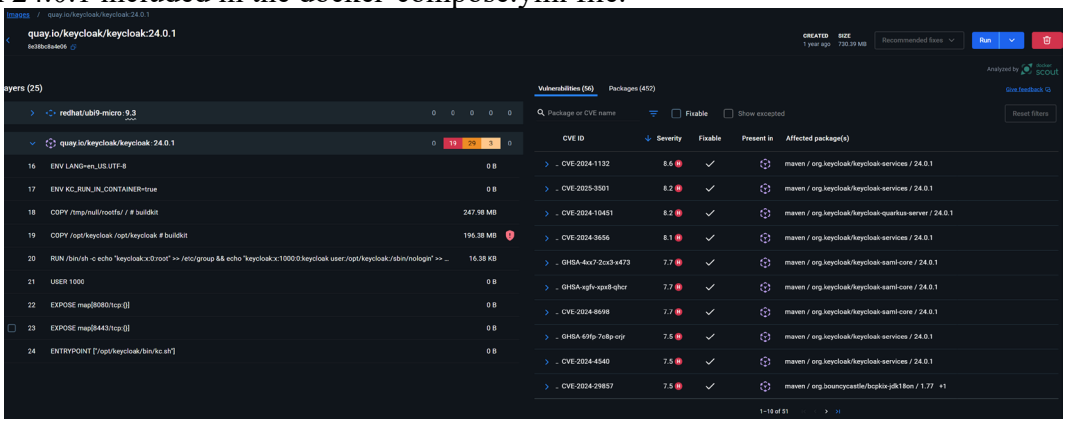

Figure 4. Vulnerabilities to Keycloak Server (Docker Scout)

After changing the Keycloak version 24.0.1 within the docker-compose.yml to 'latest' [image: quay.io/keycloak/keycloak:latest], all fifty-one vulnerabilities were mitigated.

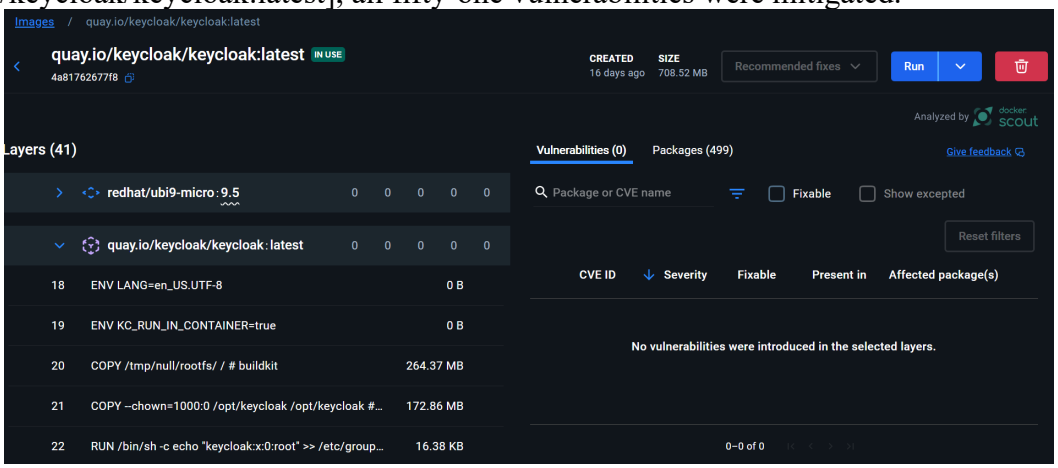

Figure 5. Vulnerabilities to Keycloak Server (Docker Scout)

Threat Modeling

STRIDE is a threat modeling methodology used to categorize potential vulnerabilities in systems. It is applied by analyzing system components against six key threats: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. This helps identify vulnerabilities early and implement security measures effectively (Allen-Addy, 2023). The below chart depicts the vulnerabilities discovered on the Flask App and Keycloak server using STRIDE Analysis.

Threat Category	Observation	Impact	Mitigation
S - Spoofing	N/A	Spoofing a user would allow unauthorized user access to environment	N/A
T - Tampering	3.9 slim version CVE-2025-47273. Users download, build, install, upgrade, and uninstall Python packages	An attacker would be allowed to write files to arbitrary locations on the filesystem with the permissions of the process running the Python code, which could escalate to remote code execution depending on the context.	Changed 3.9 slim to 3.13 slim on Flask App
R - Repudiation	N/A	N/A	N/A
I - Information disclosure	Keycloak not properly validate URLs in redirects	An attacker can construct a malicious request to bypass validation and access other URLs and potentially sensitive information within the domain or possibly conduct further attacks. Flaw affects any client that utilizes a wildcard in the Valid Redirect URIs field.	Changed from 24.0.1 to latest
D - Denial of Service	1. Improper setup denying user access 2. CVE-2022-40897 - allows remote attackers to cause a denial of service via HTML in a crafted package or custom PackageIndex page	1. Inability to access Docker environment [Flask App and Keycloak] 2. Regular Expression Denial of Service (ReDoS) in package_index.py.	1. Used proper setup 2. Changed 3.9 slim to 3.13 slim on Flask App
E - Evaluation of Privilege	3.9 slim version CVE-2024-6345. Susceptible to remote code execution	If exposed to user-controlled inputs, such as package URLs, they can execute arbitrary commands on the system.	Changed 3.9 slim to 3.13 slim on Flask App

Table 1. Flask App and Keycloak server using STRIDE Analysis (Created by the author).

Okta case study reflection.

The Okta breach from 2023 offers salient points for reflection. On 28 September 2023, a threat actor gained access to Okta's customer support case-management system due to stolen credentials from an engineer who did not set up multi-factor authentication. The attacker downloaded HTTP archive files containing session tokens to conduct session hijacking. Weeks later, BeyondTrust, a cybersecurity firm that provides protection against identity-based threats, detected unusual service-account activity. On 19 October, Okta notified affected organizations of the compromise, followed by full public disclosure of the breach a day later. In November 2023, Okta published an after-action report complete with remediation steps. The lessons learned from the Okta breach include:

1. Multi-factor authentication everywhere regardless of permissions in the system
2. Apply least privilege and segregate support from production
3. Apply short-lived session tokens with binding to client TLS
4. Automatically scrub HTTP Archive files for session cookies
5. Employ a motto of communication early and often internally and with external stakeholders in the event of a breach (SEAS8405, Reading-Material_week8)

The same lessons learned can be applied in Web 3.0 environments. Keycloak and other IAM/security methods are only as good as their implementation strategies. As of 19 May 2025, Hitachi Ltd., a Japanese financial firm, uses Keycloak to make financial-grade security easier. For API authorization in the financial sector, Financial-grade API is specified by the OpenID Foundation and widely adopted (Keycloak, 2025). The same Keycloak security capability is used for the United States Air Force's Platform One and my rudimentary Docker Container environment.

References.

Allen-Addy, C. (2023). Threat Modeling Methodology: STRIDE. IriusRisk. https://www.iriusrisk.com/resources-blog/threat-modeling-methodology-stride#:~:text=The%20STRIDE%20threat%20model%20is,and%20implement%20security%20measures%20effectively.

BeyondTrust. (2025). https://www.beyondtrust.com/.

Keycloak. (2025) Securing Applications and Services Guide. https://www.keycloak.org/docs/25.0.6/securing_apps/index.html.

NIST. (2025). CVE-2025-47273 Detail. National Vulnerability Database. https://nvd.nist.gov/vuln/detail/CVE-2025-47273.

NIST. (2025). CVE-2022-40897 Detail. National Vulnerability Database. https://nvd.nist.gov/vuln/detail/cve-2022-40897.

NIST. (2025). CVE-2024-6345 Detail. National Vulnerability Database. https://nvd.nist.gov/vuln/detail/cve-2024-6345.

Okta Website. (2025). OAuth 2.0 and OpenID Connect overview. https://developer.okta.com/docs/concepts/oauth-openid/.

System Architecture. (2020). https://students.cs.ucl.ac.uk/2020/group20/system-design/.

Reading-Material_week8.pdf