SECURITY ASSESSMENT

Tuesday
04/11/2025

ANSHUL SHUKLA
CBS-0417

Project Scenario

Overview

As the lead security engineer for CryptoV4ult, a prominent international cryptocurrency platform, you're tasked with ensuring the security and integrity of our newly established infrastructure. With over 1 million users relying on our services, it's imperative that we maintain the highest standards of security to protect their digital assets.

Your role involves a comprehensive review of the security landscape for our new application technology stack, identifying potential vulnerabilities, and running scans to assess any existing threats. Your scope encompasses various entities within our architecture, including the application itself, containerized services, and the external-facing API.

Ultimately, your objective is to develop a robust remediation plan that not only addresses current vulnerabilities but also strengthens our overall security posture, safeguarding both user data and the platform's reputation. This critical mission presents an exciting opportunity to leverage your skills and expertise in cybersecurity to fortify our infrastructure and uphold our commitment to providing a secure and reliable platform for our users. Let's embark on this journey together to ensure CryptoV4ult remains a trusted leader in the cryptocurrency industry!

Section 1:
Integrating SDLC

Transitioning to Secure SDLC

Requirements Analysis

- *Conduct user interviews to gather functional requirements.*
- *Write a requirements document for task management features.*

Security Task: *Perform a Security Requriements Assessment (SRA) to identify data protection, authentication and regulatory compliace needs*

Design

- *Create a high-level architecture diagram for the application.*
- *Design the database schema for tasks.*

Security Task: *Conduct a Threat Modeling Session using DREAD or STRIDE frameworks to identify potential attacks vectors across APIs, user interfaces and database*

Transitioning to Secure SDLC

Development

- *Code the user interface using HTML and CSS.*
- *Implement interactive elements using JavaScript.*
- *Set up a Flask application to handle API requests.*
- *Implement CRUD operations for tasks.*

Security Task: *Integrate Static Application Security Testing(SAST) tools like Bandit or SonarQube in CI/CD to detect insecure code patterns before deployment.*

Testing

- *Write and exercute functional test cases.*
- *Conduct browser compatibility testing.*

Security Task: *Perform Dynamic Application Security Testing(DAST) and penetration testing to identify runtime vulnerabilities including input validation*

Transitioning to Secure SDLC

Deployment

- *Deploy the application to Heroku.*
- *Perform smoke testing on the deployed application.*

Security Task: *Implement container and environment hardening practices ensuring secrets are stored securely and TLS is enforced end-to-end*

Maintenance

- *Monitor application logs and fix reported issues.*
- *Gather user feedback for future feature additions.*

Security Task: *Establish a Continuous Vulnerability Management process with path management, automated scans and regular review of API security reports*.

Advocating for Secure SDLC

1. Early Identification and Mitigation of Risks
Incorporating security from the intitial phases help detect vulnerabilities early reducing remediation costs and preventing cricitical flaws before deployment.
2. Enhanced Data Protection and Compliance
A Secure SDLC ensures encryption, compliance checks and access control are embedded throughout development safeguarding sensitive user data and meeting global regulatory standards.
3. Improved Code Quality and System Resilience
Continuous security teting and code reviews within SDLC workflows enhance code reliability and builds a strong attack-resistant infrastrucure.
4. Cost and Time Efficiency in Remediation
Addressing vulnerabilities during development rather than post-release minimizes incident response time, reduces downtime and preserves CryptoV4ult's operational continuity.
5. Strengthened Customer Trust and Brand Reputation
By proactively integrating security into every phase we demonstrate CryptoV4ult's commitment to safeguarding the user assets.

Section 2:
Vulnerabilities and Remediation

Vulnerabilities and remediation

1. Weak Authentication and Credential Storage
Description
Our current authentication stack relies predominantly on single-factor passwords. Password strength enforcement is inconsistently applied across services and credential storage uses non-standardized hashing parameters.
Risk
Compromised credentials enable account takeover and unauthorized transfers. Given the financial nature of our service this can directly result in user fund loss, regulatory exposure, remediation costs and significant reputational damage.
Remediation
• *Enforce a single corporate password policy (minimum entropy, banned-password lists, no plaintext storage).* • *Standardize credential storage on Argon2id (documented parameters) and ensure per-user salts across all services.*

Vulnerabilities and remediation

2. Account enumeration and exposure to brute-force / credential-stuffing
Description
Authentication and recovery endpoints return distinguishable messages Rate-limiting and anti-automation controls are not consistently applied. There is no integration with breached-credential feeds.
Risk
This enables attackers to enumerate user accounts and run credential-stuffing or large-scale brute-force attacks increasing the probability of account compromise and subsequent fraud.
Remediation
• *Normalize API responses for login and recovery endpoints to avoid revealing account existence.* • *Apply consistent rate-limiting and progressive backoff on authentication and recovery endpoints.* • ***Implement anti-automation controls**: IP reputation checks, device fingerprinting and selective CAPTCHA or equivalent friction for suspicious flows.*

Vulnerabilities and remediation

3. Insecure session management and token handling
Description
Session cookies and API tokens are not consistently protected with secure cookie attributes and token lifetimes/rotation policies vary by service. There is limited detection for token replay, theft or anomalous session behavior.
Risk
Weak session handling can lead to session hijacking or replay attacks that bypass authentication controls and MFA. Successful exploitation can immediately enable unauthorized transfers or elevated actions.
Remediation
• *Ensure session cookies are set with Secure and HttpOnly flags and appropriate SameSite attributes issue tokens only over TLS.* • *Implement short-lived access tokens with rotating refresh tokens and a documented token revocation endpoint.* • *Introduce session anomaly detection (geographic/device anomalies) and require re-authentication on flagged events.*

Threat Matrix

Pathway (Vulnerability)	Impact Level	Likelihood Level
Weak authentication and inconsistent credential storage	*High*	*Medium*
Account enumeration and credential-stuffing	*Medium*	*High*
Insecure session management and token handling	*High*	*Medium*

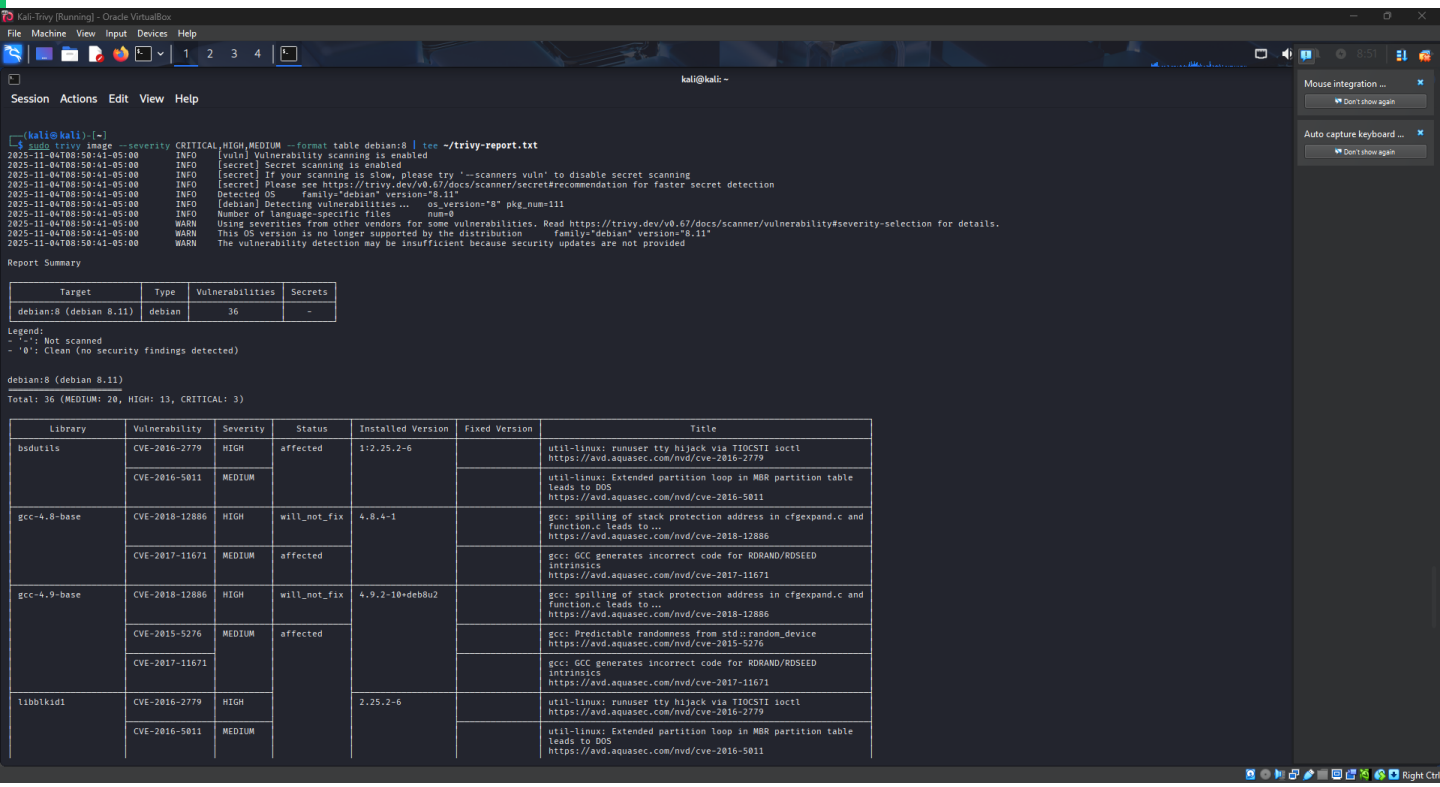
Project Information Slide

Impact / Likelihood	Low	Medium	High
High	--	***Account enumeration / credential-stuffing:*** *High priority (automation risk)*	--
Medium	--	--	***Weak authentication and inconsistent credential storage****: High priority (impact mitigation required)* ***Insecure session management and token handling:*** *High priority (impact mitigation required)*
Low	--	--	--

Section 3:
Container Security

Project Information Slide

Trivy scan screenshot



Project Information Slide

						https://avd.aquasec.com/nvd/cve-2020-14155
libsmartcols1	CVE-2016-2779	HIGH		2.25.2-6		util-linux: runuser tty hijack via TIOCSTI ioctl https://avd.aquasec.com/nvd/cve-2016-2779
	CVE-2016-5011	MEDIUM				util-linux: Extended partition loop in MBR partition table leads to DOS https://avd.aquasec.com/nvd/cve-2016-5011
libstdc++6	CVE-2018-12886	HIGH	will_not_fix	4.9.2-10+deb8u2		gcc: spilling of stack protection address in cfgexpand.c and function.c leads to ... https://avd.aquasec.com/nvd/cve-2018-12886
	CVE-2015-5276	MEDIUM	affected			gcc: Predictable randomness from std::random_device https://avd.aquasec.com/nvd/cve-2015-5276
	CVE-2017-11671					gcc: GCC generates incorrect code for RDRAND/RDSEED intrinsics https://avd.aquasec.com/nvd/cve-2017-11671
libuuid1	CVE-2016-2779	HIGH		2.25.2-6		util-linux: runuser tty hijack via TIOCSTI ioctl https://avd.aquasec.com/nvd/cve-2016-2779
	CVE-2016-5011	MEDIUM				util-linux: Extended partition loop in MBR partition table leads to DOS https://avd.aquasec.com/nvd/cve-2016-5011
login	CVE-2017-12424	CRITICAL		1:4.2-3+deb8u4		shadow-utils: Buffer overflow via newusers tool https://avd.aquasec.com/nvd/cve-2017-12424
mount	CVE-2016-2779	HIGH		2.25.2-6		util-linux: runuser tty hijack via TIOCSTI ioctl https://avd.aquasec.com/nvd/cve-2016-2779
	CVE-2016-5011	MEDIUM				util-linux: Extended partition loop in MBR partition table leads to DOS https://avd.aquasec.com/nvd/cve-2016-5011
passwd	CVE-2017-12424	CRITICAL		1:4.2-3+deb8u4		shadow-utils: Buffer overflow via newusers tool https://avd.aquasec.com/nvd/cve-2017-12424
perl-base	CVE-2018-6797		will_not_fix	5.20.2-3+deb8u12		perl: heap write overflow in regcomp.c https://avd.aquasec.com/nvd/cve-2018-6797
util-linux	CVE-2016-2779	HIGH	affected	2.25.2-6		util-linux: runuser tty hijack via TIOCSTI ioctl https://avd.aquasec.com/nvd/cve-2016-2779
	CVE-2016-5011	MEDIUM				util-linux: Extended partition loop in MBR partition table leads to DOS https://avd.aquasec.com/nvd/cve-2016-5011

Key Findings:

- **36 total vulnerabilities** detected: 3 Critical, 13 High, 20 Medium.
- **Root cause:** outdated base image (Debian 8) and presence of development/runtime packages (perl, gcc, libstdc++).
- **Immediate risk:** account / privilege escalation and potential remote code execution if exploited.

Remediation summary

- **Upgrade base image** to a supported release (e.g., debian:stable) and rebuild images.
- **Remove compilers and dev libs from runtime** using multi-stage builds so only application artifacts are shipped.
- **Enforce CI gating**: fail builds on HIGH/CRITICAL Trivy results and schedule weekly registry scans.

Report to Fix Container Issues

Vulnerability Name	Unpatched Software Version	Patched Software Version (recommended)
perl-base:CVE-2018-6797	5.20.2-3+deb8u12	No fix in this image: Upgrade base image to a supported Debian release or remove perl from runtime
passwd:CVE-2017-12424	1:4.2-3+deb8u4	No fix in this image: Upgrade base image / upgrade shadow-utils in newer distro
login:CVE-2017-12424	1:4.2-3+deb8u4	No fix in this image: Upgrade base image / upgrade shadow-utils in newer distro
libstdc++6:CVE-2018-12886	4.9.2-10+deb8u2	No fix in this image: Use newer base image or remove runtime dev libs
libgcc1:CVE-2018-12886	1:4.9.2-10+deb8u2	No fix in this image: Use newer base image or remove runtime dev libs
gcc-4.9-base:CVE-2018-12886	4.9.2-10+deb8u2	No fix in this image: Remove compiler runtime artifacts; use multi-stage build
gcc-4.8-base:CVE-2018-12886	4.8.4-1	No fix in this image: Same as above: remove compilers or upgrade base image

Section 4:
API Security

Vulnerabilities and remediation

1. Excessive data exposure and lack of data minimization (Privacy risk)
Description
The proposed API is intended to export user telemetry to a third party. Without a data-minimization policy and strong transformation/pseudonymization controls.
Risk
Unauthorized or unnecessary disclosure of PII will create direct regulatory liability (GDPR, CCPA, etc.) increase the likelihood of data breaches, expose user funds or identities and materially damage customer trust and the company brand.
Remediation
• *Implement a transformation layer in the API gateway that enforces field-level allowlists and performs pseudonymization or hashing for any identifiers (e.g., replace email/user_id with irreversible token).* • *Instrument data loss prevention (DLP) on outbound flows to detect and block accidental PII leakage.* • ***Owner:** Product (data requirements) / Identity & API Platform (implementation).* • ***Target:** Basic allowlist/pseudonymization and consent checks in 2–4 weeks full DLP integration and audit logging in 8–12 weeks.* • ***Verification:** Automated tests proving disallowed fields are removed DLP alerts configured and tested sample exports demonstrating pseudonymization and consent enforcement.*

Vulnerabilities and remediation

2. Inadequate authentication, authorization, and scope control for third-party access

Description

If the vendor is issued long-lived API keys or broad-scoped credentials (or if we rely on simple shared secrets) the vendor will possess excessive access. There is likely no fine-grained OAuth2 scope model no client identity binding (mTLS) and no short-lived credentials/rotation in the proposed approach.

Risk

Compromised or misused vendor credentials can be used to access data beyond the intended scope, allow lateral movement to other APIs or enable large-scale exfiltration. This increases both operational and regulatory risk and can lead to catastrophic data breaches.

Remediation

- *Adopt an OAuth2.0 / OpenID Connect model for third-party access with fine-grained scopes (least privilege) and short-lived access tokens. Use refresh tokens sparingly and only with strong controls.*
- *Implement an API gateway that enforces scope checks, per-client rate limits and data allowlists per client. Configure token introspection and token revocation endpoints.*
- ***Owner:** API Platform (implementation) with Security sponsorship.*
- ***Target:** Gatekeeping with scope enforcement and short-lived tokens in 3–6 weeks mTLS onboarding within 6–10 weeks.*
- ***Verification:** Penetration tests demonstrating inability to access out-of-scope data; automated test suite verifying token expiry and revocation behavior; audit demonstrating credential rotation.*

Vulnerabilities and remediation

3. Insecure third-party integration and inadequate vendor controls (governance and operational risk)

Description

The business has engaged an external sales vendor without documented security requirements, no Data Processing Agreement (DPA) and without conducting a vendor security assessment or penetration test of the vendor's ingest processes and storage.

Risk

Weak vendor controls (insufficient encryption at rest poor key management lax access control) or poor operational hygiene at the vendor can result in a downstream breach that implicates CryptoV4ult legally and publicly. Lack of contractual obligations also limits our ability to enforce remediation and to meet regulatory obligations.

Remediation

- *Execute a mandatory Vendor Security Assessment before any data exchange. Required checks should include SOC 2 / ISO 27001 evidence, encryption and key management practices, access control policies, incident response capability and retention policies.*
- *Sign a comprehensive Data Processing Agreement (DPA) that defines permitted uses, retention limits, security requirements (encryption in transit and at rest) breach notification timelines, and audit rights.*
- *Limit data retention and require secure deletion at contract termination or per retention policy.*
- ***Owner:** Legal (DPA) and Vendor Management / Security (assessment and enforcement).*
- ***Target:** Vendor assessment and DPA negotiation prior to any data sharing initial remediation items completed before go-live.*
- ***Verification:** Signed DPA completed vendor assessment report with remediation plan, and evidence of implemented technical controls on vendor side.*