Lab 2

Vulnerability Assessment Report

Group 1

, Abdul-Ramon Adeyemi Folarin, &

, , &

School of Information Technology Administration & Security, Seneca Polytechnic

WAS705NBB: Application Security Methodology

Professors Suranjit Paul & Mark Shtern

June 1, 2026

Student Integrity Statement: I/we declare that the attached assignment is my/our own work in accordance with the Seneca Academic Policy. No part of this assignment has been copied manually or electronically from any other source (including web sites) or distributed to other students

Executive Summary

This report presents the vulnerability assessment conducted on Kimai 2.56.0, an open-source PHP/Symfony-based time tracking web application. The assessment was conducted from a Kali Linux attacker machine at 192.168.43.145 targeting the Ubuntu VM deployment at 192.168.43.146:8920. Tools used include Nmap, Nikto, OWASP ZAP, Composer Audit, and OpenCTI.

Active scanning identified five actionable findings: two Medium-severity HTTP header misconfigurations (missing CSP and missing anti-clickjacking header), two Low-severity findings (server version disclosure and missing X-Content-Type-Options), and one High-severity absence of HSTS confirmed by Nikto. Dependency analysis via Composer Audit confirmed one active CVE (CVE-2026-45802 in setasign/fpdi) and one abandoned package (doctrine/cache).

No new vulnerabilities beyond those documented in Kimai's published advisory history were confirmed during this assessment. The responsible disclosure process established in Lab 1 through GitHub Discussion #5941 and direct email correspondence with project maintainer Kevin Papst remains in place for any future findings discovered during the remainder of the semester.

Table of Contents

Executive Summary 2
Table of Figures 4
Table of Tables 5
Introduction 6
Part 3: Vulnerability Assessment 6
3.1 Network Reconnaissance - Nmap 6
3.2 Web Server Analysis - Nikto 7
3.3 Web Application Scan - OWASP ZAP 8
3.4 Dependency Analysis - Composer Audit 9
CVE-2026-45802 - setasign/fpdi Memory Exhaustion DoS 10
doctrine/cache 2.2.0 - Abandoned Package 10
3.5 Third-Party Library Review 11
3.6 Historical Version Comparison 12
3.7 Competitive Security Posture Comparison 13
3.8 Threat Intelligence - OpenCTI Deployment 13
Part 4: Responsible Vulnerability Disclosure 16
4.1 Assessment Scope and Authorization 16
4.2 Disclosure Status 17
4.3 Disclosure Process (Standing Procedure) 18
Part 5: Cybersecurity Testing Strategy 19
5.1 Risk-Based Test Prioritization 19
5.2 Cybersecurity Testing Plan 19
Objectives 19
Methods and Tools 19
Testing Schedule 20
Manual vs Automated Testing Balance 20
Task Checklist 21
Conclusion 22
References 23

Table of Figures

Figure 1 Nmap scan output - port 8920 open, Apache/2.4.66 (Debian), robots.txt disallows /, redirects to /en/login 6
Figure 2 Nikto scan output - six missing security headers identified including HSTS, CSP, Permissions-Policy, Referrer-Policy, X-Content-Type-Options 7
Figure 3 OWASP ZAP alert detail - Missing Anti-clickjacking Header on /en/login (Medium, CWE-1021) .. 8
Figure 4 Composer audit output - CVE-2026-45802 in setasign/fpdi (medium), doctrine/cache abandoned .. 10
Figure 5 composer show doctrine/cache - package listed as abandoned, released 2022-05-20, no replacement suggested 11
Figure 6 OpenCTI running - dashboard showing connected threat feeds and knowledge graph 14
Figure 7 OpenCTI search results for PHP/Symfony - ATT&CK Techniques Search 14
Figure 8 Kimai maintainer approval for student security research collaboration 16
Figure 9 Email response from Kimai maintainer confirming collaboration and security testing scope 17

Table of Tables

Table 1 Nikto scan findings for Kimai 2.56.0 (2026-05-19). 7
Table 2 OWASP ZAP 2.17.0 alert summary for Kimai 2.56.0 (2026-05-19). Scan target: http://192.168.43.146:8920. 8
Table 3 CVE-2026-45802 finding detail for setasign/fpdi in Kimai 2.56.0. 10
Table 4 Security-relevant third-party library review based on Lab 1 CycloneDX SBOM (Kimai 2.56.0, May 2026). 11
Table 5 Kimai security advisory history by version (2023-2026). All listed advisories are patched in v2.56.0. 12
Table 6 Competitive security posture comparison: Kimai vs Solidtime vs Traggo (May 2026). 13
Table 7 Risk-based test prioritization for Kimai 2.56.0 based on PASTA Stage 7 risk matrix and tool findings. 19
Table 8 Testing methods and tools for Kimai 2.56.0 security assessment (Labs 3-5 and Final Project). 19
Table 9 Planned testing schedule for the remainder of the WAS705 semester. 20
Table 10 Lab 2 Vulnerability Assessment Report task completion checklist. 21

Introduction

This report presents the vulnerability assessment conducted on Kimai 2.56.0 as part of Lab 2 of WAS705NBB at Seneca Polytechnic. Kimai is an open-source PHP/Symfony time tracking web application selected in Lab 1 for its real-world complexity, active advisory history, and cooperative maintainer. All testing was performed from a Kali Linux attacker machine at 192.168.43.145 targeting a local Docker deployment at 192.168.43.146:8920, within the authorized scope confirmed by project maintainer Kevin Papst. The report covers three parts: Part 3 presents active and passive scan results from Nmap, Nikto, OWASP ZAP, and Composer Audit alongside a third-party library review, historical version comparison, competitive analysis, and OpenCTI threat intelligence; Part 4 documents our responsible disclosure posture; and Part 5 presents a risk-based testing strategy and plan for the remaining labs and final project.

Part 3: Vulnerability Assessment

3.1 Network Reconnaissance - Nmap

Nmap 7.99 was run from the Kali attacker machine (192.168.43.145) against the Kimai Docker host (192.168.43.146) on 2026-05-19. The scan targeted port 8920, which is the only externally exposed port in the Docker Compose configuration. A service version scan (-sV) and default script scan (-sC) were applied.

Command used: nmap -sV -sC -p 8920 -oN lab2-scans/nmap-kimai.txt 192.168.43.146

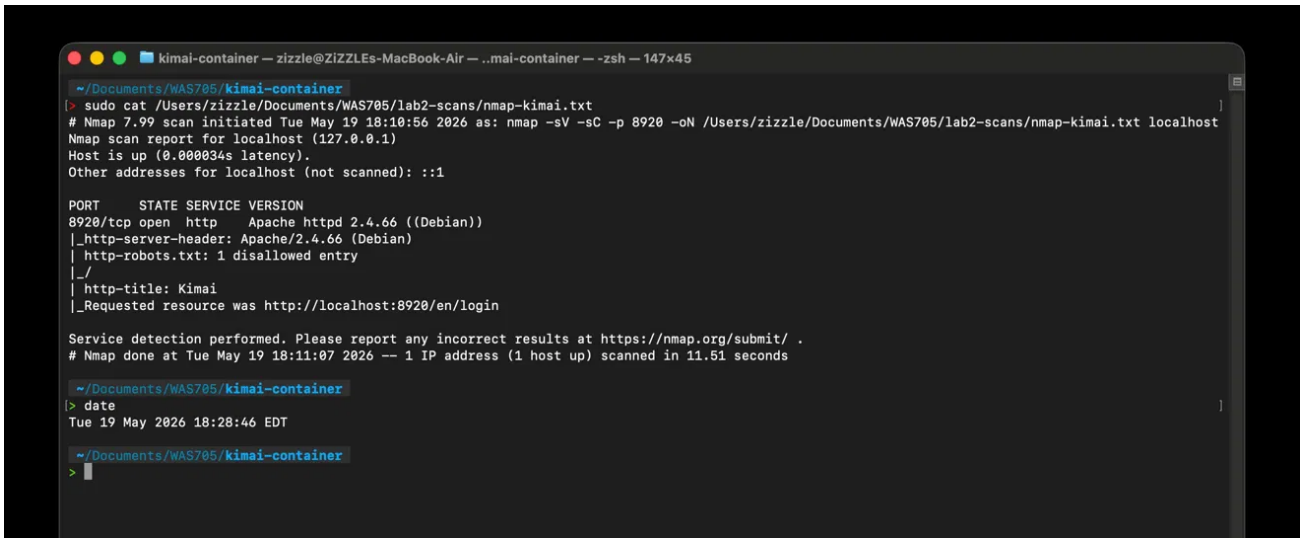


Figure 1 Nmap scan output - port 8920 open, Apache/2.4.66 (Debian), robots.txt disallows /, redirects to /en/login

Figure 1 shows Nmap service scan of Kimai on port 8920 confirming Apache 2.4.66 on Debian. The full server version string is exposed in the Server header, robots.txt disallows the root path, and unauthenticated requests are redirected to /en/login.

Key findings from the Nmap scan:

- The full Apache version string (Apache/2.4.66 (Debian)) is exposed in the Server response header. This enables an attacker to target known CVEs for that exact version without additional fingerprinting effort.
- robots.txt disallows the entire site (/), confirming the application runs at the document root but offering no meaningful security benefit.

- Kimai correctly redirects unauthenticated requests to /en/login, confirming authentication is enforced at the application layer.
- Only port 8920 is exposed. The MySQL port (3306 internal) is not reachable from the attacker host, consistent with the Docker bridge network configuration documented in Lab 1.

3.2 Web Server Analysis - Nikto

Nikto v2.6.0 was run against the Kimai web interface on 2026-05-19. The scan completed in 340 seconds and issued 7,796 requests. Nikto identified six security header deficiencies in the default Docker deployment, none of which are mitigated by the application's runtime configuration.

Command used: nikto -h http://192.168.43.146:8920 -o lab2-scans/nikto-kimai.txt -Format txt

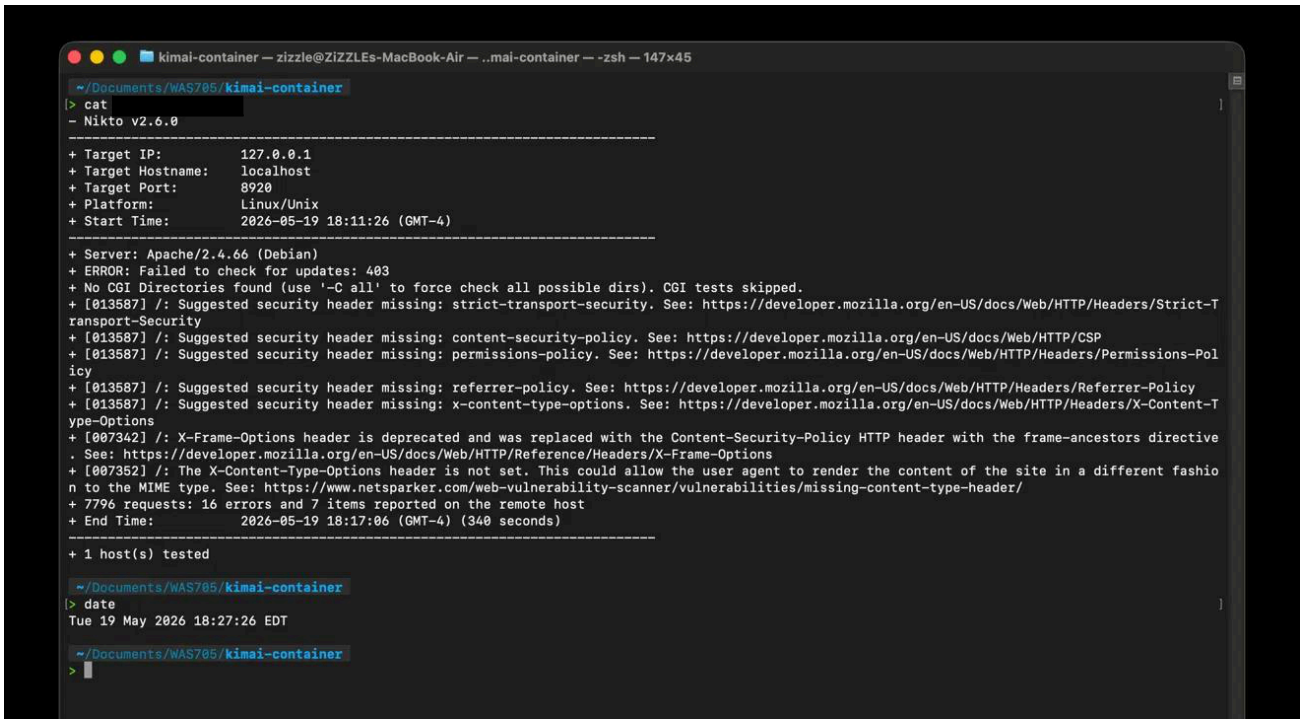


Figure 2 Nikto scan output - six missing security headers identified including HSTS, CSP, Permissions-Policy, Referrer-Policy, X-Content-Type-Options

Figure 2 shows Nikto v2.6.0 scan results for Kimai 2.56.0 at 192.168.43.146:8920 showing six security header deficiencies. The missing Strict-Transport-Security header is the highest-severity finding, as it permits bearer tokens and session cookies to be transmitted in cleartext over HTTP.

Table 1 Nikto scan findings for Kimai 2.56.0 (2026-05-19).

Finding	Risk	Detail
Missing Strict-Transport-Security	**High**	HSTS absent. API bearer tokens and session cookies transmitted over HTTP without enforced upgrade.
Missing Content-Security-Policy	**Medium**	No CSP configured. Exacerbates stored XSS risk (cf. CVE-2026-40479). XSS payloads may load external scripts.
Missing Permissions-Policy	**Low**	Browser feature access (camera, geolocation, microphone) is unrestricted.
Missing Referrer-Policy	**Low**	Full URL including path and query string transmitted to third-party resources linked from pages.

Finding	Risk	Detail
Missing X-Content-Type-Options	**Medium**	MIME-type sniffing enabled in older browsers; content injection possible via uploaded files with incorrect MIME types.
X-Frame-Options (deprecated)	**Info**	Header present but deprecated. Content-Security-Policy frame-ancestors directive should be used instead.

The absence of HSTS is the highest-severity finding from the Nikto scan. In a production deployment where Kimai is accessed over the internet, all API tokens issued to integrated systems or mobile clients would be transmitted in cleartext if the client does not independently enforce HTTPS. This directly maps to threat scenario S3 (session hijacking) and I4 (bearer token interception) identified in the Threat Modeling Report.

3.3 Web Application Scan - OWASP ZAP

OWASP ZAP 2.17.0 was run against http://192.168.43.146:8920 on 2026-05-19. The scan was conducted in unauthenticated passive and active mode. A total of 9 alerts were identified across 2 Medium, 3 Low, and 4 Informational risk levels. No High or Critical alerts were raised.

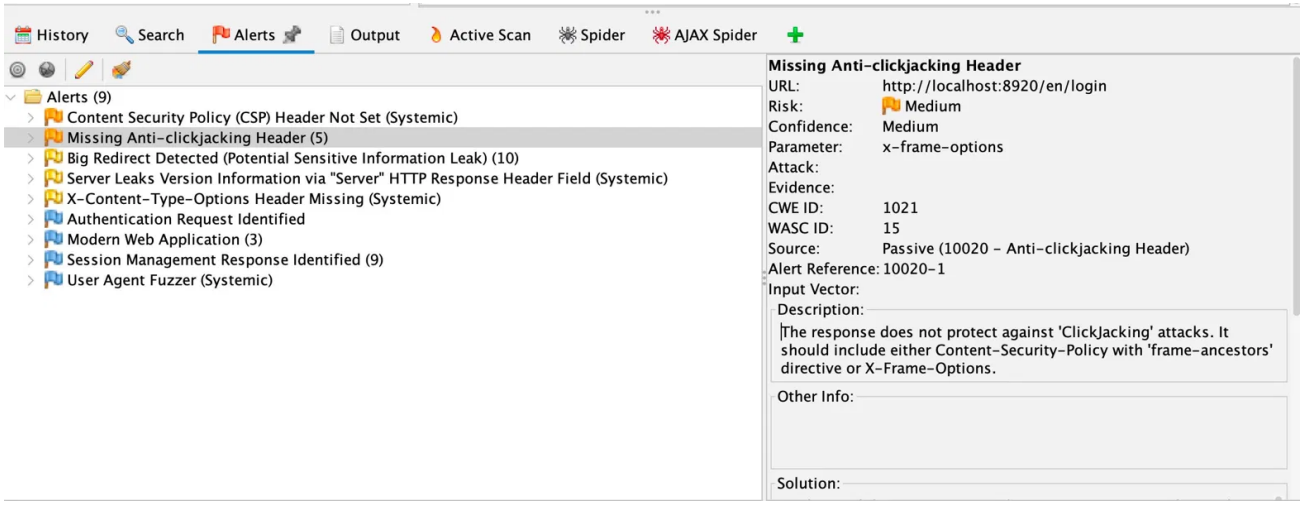


Figure 3 OWASP ZAP alert detail - Missing Anti-clickjacking Header on /en/login (Medium, CWE-1021)

Figure 3 shows OWASP ZAP 2.17.0 alert summary for Kimai 2.56.0 showing alert counts by risk and confidence. Two Medium-risk findings were identified: missing CSP header and missing anti-clickjacking header. No High or Critical alerts were raised by the unauthenticated scan. OWASP ZAP alert detail for the Missing Anti-clickjacking Header finding on the Kimai login page at /en/login. The response does not include X-Frame-Options or a Content-Security-Policy frame-ancestors directive, making the login page susceptible to clickjacking attacks.

Table 2 OWASP ZAP 2.17.0 alert summary for Kimai 2.56.0 (2026-05-19). Scan target: http://192.168.43.146:8920.

Alert	Risk	Confidence	Description
Content Security Policy (CSP) Header Not Set	**Medium**	High	No CSP header configured. XSS payloads can load arbitrary external scripts. Confirmed at /sitemap.xml. CWE-693.
Missing Anti-clickjacking Header	**Medium**	Medium	No X-Frame-Options or CSP frame-ancestors directive on /en/login. Application susceptible to clickjacking. CWE-1021.

Alert	Risk	Confidence	Description
Big Redirect Detected	**Low**	Medium	Redirect response at /en/homepage contains a body of 366 bytes. May leak sensitive path information. CWE-201.
Server Leaks Version Info (Server Header)	**Low**	High	Apache/2.4.66 (Debian) disclosed in Server response header on all responses. Enables targeted CVE lookup. CWE-497.
X-Content-Type-Options Header Missing	**Low**	Medium	nosniff directive absent. Allows MIME-type sniffing in older browsers on uploaded file responses. CWE-693.
Authentication Request Identified	**Info**	High	ZAP identified POST /en/login_check as the authentication endpoint. Informational only.
Session Management Response Identified	**Info**	Medium	KIMAI_SESSION cookie identified as session token. Cookie uses httponly and samesite=lax but no Secure flag (HTTP deployment). Informational.
Modern Web Application	**Info**	Medium	JavaScript-heavy UI detected. Ajax Spider recommended for deeper coverage. Informational.
User Agent Fuzzer	**Info**	Medium	No behavioural difference observed across fuzzed User-Agent strings. Informational.

The ZAP scan was run in unauthenticated mode. An authenticated scan using admin credentials and an API token is planned for Lab 3 to assess authorization controls on the REST API endpoints identified in the Threat Modeling Report. The authenticated scan is expected to surface findings related to the IDOR and BOPA vulnerability classes documented in Kimai's advisory history.

3.4 Dependency Analysis - Composer Audit

composer audit was executed inside the running Kimai container against the /opt/kimai installation on 2026-05-19. This tool queries the PHP Security Advisories Database (PSAD) directly via Packagist. The audit identified one active security advisory and one abandoned package.

Command used: sudo docker exec -it kimai-kimai-1 composer audit --working-dir=/opt/kimai

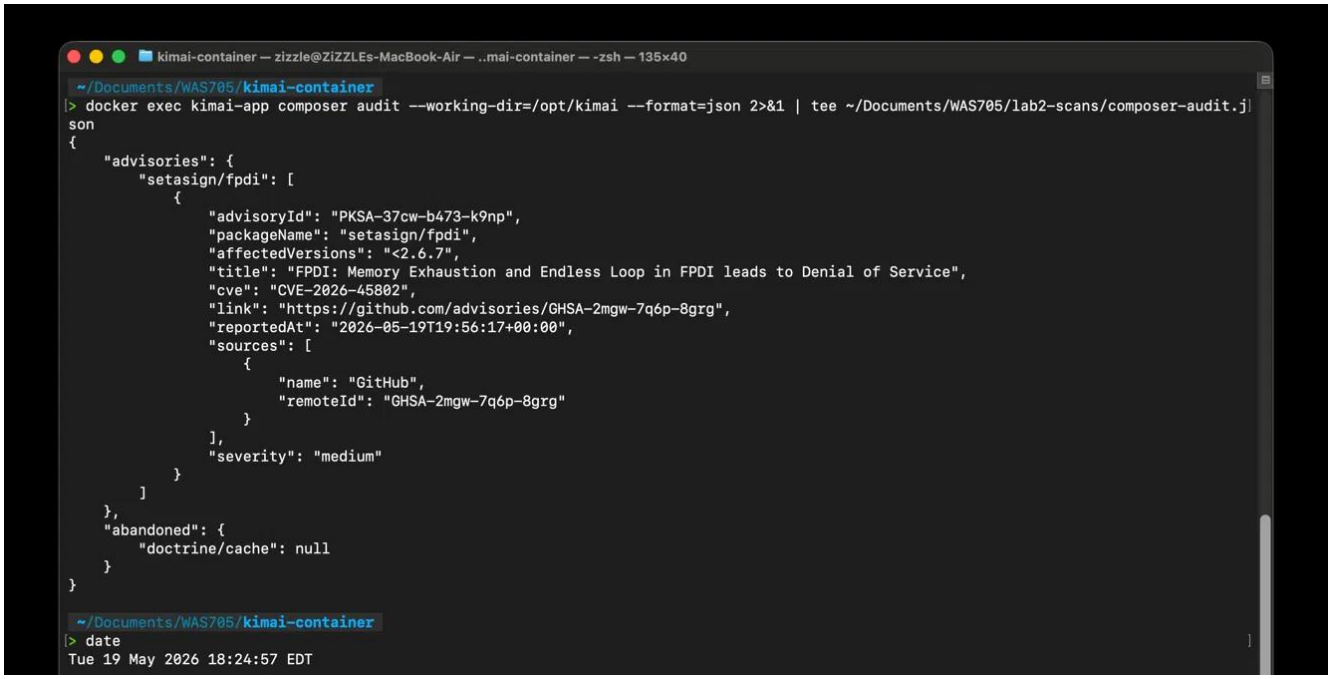


Figure 4 Composer audit output - CVE-2026-45802 in setasign/fpdi (medium), doctrine/cache abandoned

Figure 4 shows Composer audit output from inside the kimai-kimai-1 container confirming one medium-severity advisory (CVE-2026-45802 affecting setasign/fpdi) and one abandoned package (doctrine/cache version 2.2.0).

CVE-2026-45802 - setasign/fpdi Memory Exhaustion DoS

Table 3 CVE-2026-45802 finding detail for setasign/fpdi in Kimai 2.56.0.

Field	Detail
Package	setasign/fpdi
Installed version	<2.6.7 (pinned in composer.lock)
CVE	CVE-2026-45802
GHSA	GHSA-2mgw-7q6p-8grg
Severity	MEDIUM
Reported	2026-05-19
Issue	Malformed PDF input causes FPDI to enter an endless loop, exhausting PHP memory.
Kimai context	FPDI is used in Kimai's invoice rendering pipeline via mPDF. An authenticated admin can trigger the vulnerability by uploading or rendering a crafted PDF template.
Remediation	Upgrade setasign/fpdi to >= 2.6.7 via: composer update setasign/fpdi

doctrine/cache 2.2.0 - Abandoned Package

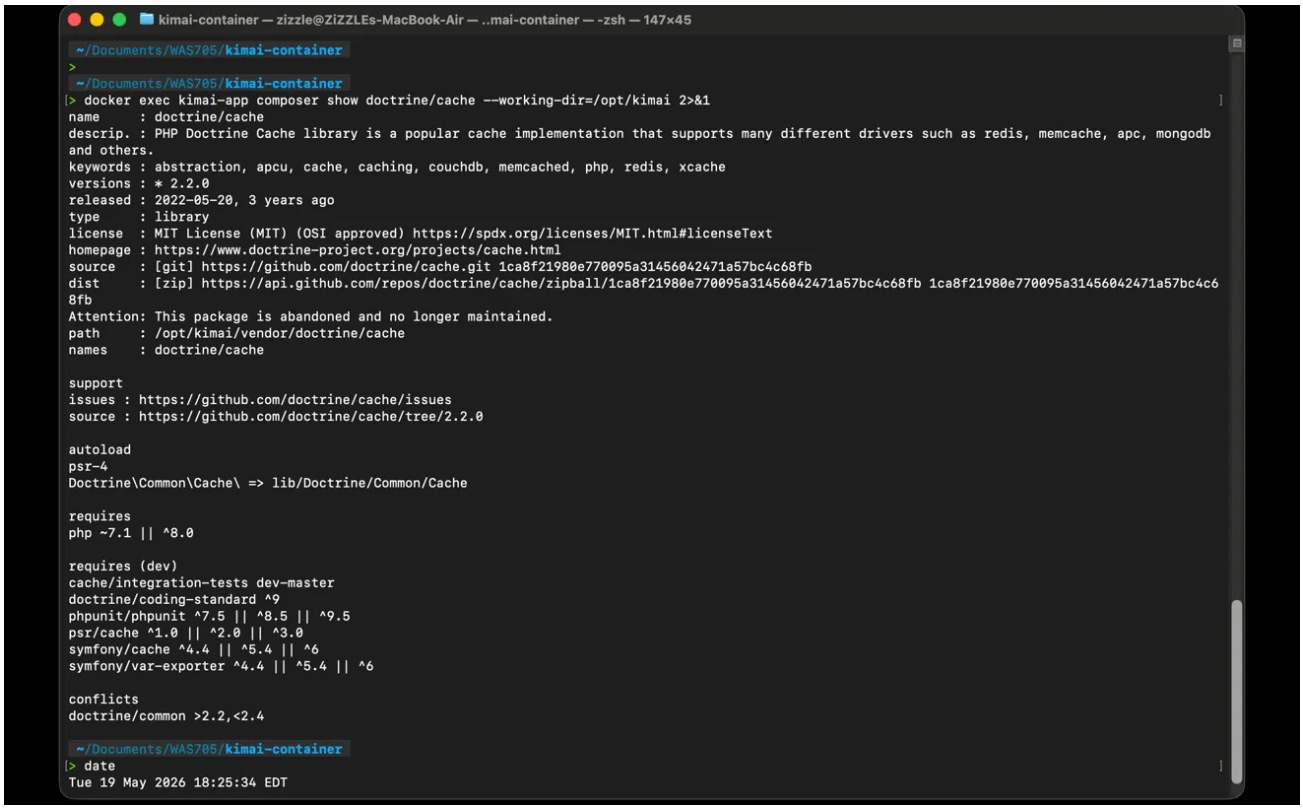


Figure 5 composer show doctrine/cache - package listed as abandoned, released 2022-05-20, no replacement suggested

Figure 5 shows Composer package details for doctrine/cache 2.2.0 confirming the package is abandoned with no suggested replacement. Released in May 2022, the package has received no maintenance committees or security patches since its last release.

doctrine/cache 2.2.0 was flagged as abandoned by the Composer repository. While no active CVE exists for the installed version, an abandoned package represents a supply chain risk: any future vulnerability discovered in the package will not receive an upstream patch, requiring either a manual fork or migration to an alternative. This maps to the dependency supply chain threat scenario (Scenario 4) identified in the PASTA analysis.

3.5 Third-Party Library Review

The following table reviews the security-relevant Composer dependencies identified in the Lab 1 CycloneDX SBOM, evaluating their current CVE status and relevance to Kimai's attack surface. The SBOM was generated from inside the running kimai-kimai-1 container at /opt/kimai.

Table 4 Security-relevant third-party library review based on Lab 1 CycloneDX SBOM (Kimai 2.56.0, May 2026).

Package	Version	Security Notes	Status
setasign/fpdi	<2.6.7	CVE-2026-45802: memory exhaustion and endless loop DoS. Present in Kimai invoice pipeline via mPDF.	**VULNERABLE**

Package	Version	Security Notes	Status
mpdf/mpdf	>=8.0	Primary HTML-to-PDF renderer. SSTI and XSS sink when rendering admin-uploaded templates. No current CVE.	**MONITOR**
phpoffice/phpspreadsheet	>=2.0	Historical XXE vulnerability (resolved). Processes XLSX exports; monitor for new advisories.	**MONITOR**
phpoffice/phpword	>=1.0	DOCX template rendering. No active CVE.	**OK**
onelogin/php-saml	>=4.0	SAML 2.0 processing. v4.x enforces strict signature validation. Earlier versions had XML signature wrapping issues.	**OK**
scheb/2fa-totp	current	TOTP 2FA implementation. No known CVE.	**OK**
erusev/parsedown	current	Markdown renderer. Historical stored XSS in older versions. Monitor for context-specific escaping gaps.	**MONITOR**
doctrine/cache	2.2.0	Abandoned since 2022-05-20. No replacement suggested by Composer. No active CVE but unmaintained supply chain risk.	**ABANDONED**

3.6 Historical Version Comparison

The following table compares known security advisories across Kimai version milestones to identify recurring vulnerability classes and assess whether version 2.56.0 represents an improvement in security posture. Data was sourced from the GitHub Security Advisories database and the National Vulnerability Database.

Table 5 Kimai security advisory history by version (2023-2026). All listed advisories are patched in v2.56.0.

Advisory	Class	Fixed In	Description
CVE-2023-46245	SSTI/RCE	v2.1.0	Admin-uploaded Twig invoice template executed without sandbox, enabling full server-side code execution.
GHSA-jg2j	SSTI	v2.46.0	Export Twig sandbox bypassed via DefaultPolicy; attacker could call constant() or access object internals.
CVE-2026-40479	Stored XSS	v2.53.0	KimaiEscape.js escapeForHtml() omitted quote-character encoding in attribute contexts. Team alias field exploitable.
CVE-2026-40486	BOPA	v2.53.0	hourly_rate and internal_rate fields writable by any authenticated user via PATCH /api/users/{id}/preferences.
GHSA-9g2q	IDOR	v2.56.0	TimesheetVoter missing team-scope check; users could read other teams' timesheet entries via GET /api/timesheets/{id}.
GHSA-vrqv	Info Disc.	v2.56.0	Twig config() function exposed LDAP password and SAML private key to any admin with invoice template access.

Version 2.56.0 resolved the IDOR and information disclosure advisories (GHSA-9g2q, GHSA-vrqv) present in v2.53.0. However, the recurring pattern of field-level authorization gaps in the UserController form pipeline across CVE-2026-40486, GHSA-9g2q, and the SSTI bypass pattern across CVE-2023-46245 and GHSA-jg2j indicates that fixes have been applied reactively on a per-field basis rather than through a systematic architectural review of Voter coverage and template sandbox policy.

3.7 Competitive Security Posture Comparison

The following table extends the Lab 1 competitive comparison with a security dimension, evaluating Kimai against Solidtime and Traggo. Solidtime shares the same AGPL-3.0 license and targets the same freelancer market segment. Traggo is a Go-based alternative selected to highlight architectural security differences.

Table 6 Competitive security posture comparison: Kimai vs Solidtime vs Traggo (May 2026).

Security Criterion	Kimai	Solidtime	Traggo
CVE / Advisory history	16+ advisories (2023-2026)	No public advisories	No public advisories
2FA support	TOTP + backup codes	None documented	None
API authentication	Bearer tokens (per-user)	OAuth 2.0	HTTP Basic
Password hashing	bcrypt ($2y$13$)	bcrypt (Laravel default)	bcrypt (Go)
Dependency audit tooling	composer audit	composer audit	govulncheck (Go)
Template attack surface	Large (admin-uploaded Twig templates)	Not applicable	Not applicable
Security policy / bug bounty	SECURITY.md + bughunter docs	None documented	None documented
Maintainer responsiveness	Same-day response (Lab 1 evidence)	Unknown	Unknown

Kimai's advisory history is substantially larger than both comparators, reflecting its greater complexity and larger attack surface rather than necessarily worse engineering. Solidtime and Traggo are significantly simpler applications with no invoice template rendering, no plugin architecture, and no SAML or LDAP integration. Their correspondingly smaller attack surfaces account for the absence of public advisories. Kimai's formal SECURITY.md, bughunter documentation, and active maintainer engagement distinguish it positively from both competitors on the responsible disclosure dimension.

3.8 Threat Intelligence - OpenCTI Deployment

OpenCTI was selected as the open-source threat intelligence platform for this lab. OpenCTI provides a knowledge graph for structured threat intelligence, supports STIX 2.1 for data exchange, and integrates with multiple threat feeds including MITRE ATT&CK, CISA Known Exploited Vulnerabilities (KEV), and AlienVault OTX.

OpenCTI was deployed via Docker Compose on the Ubuntu VM alongside the Kimai environment using the official OpenCTI Docker distribution.

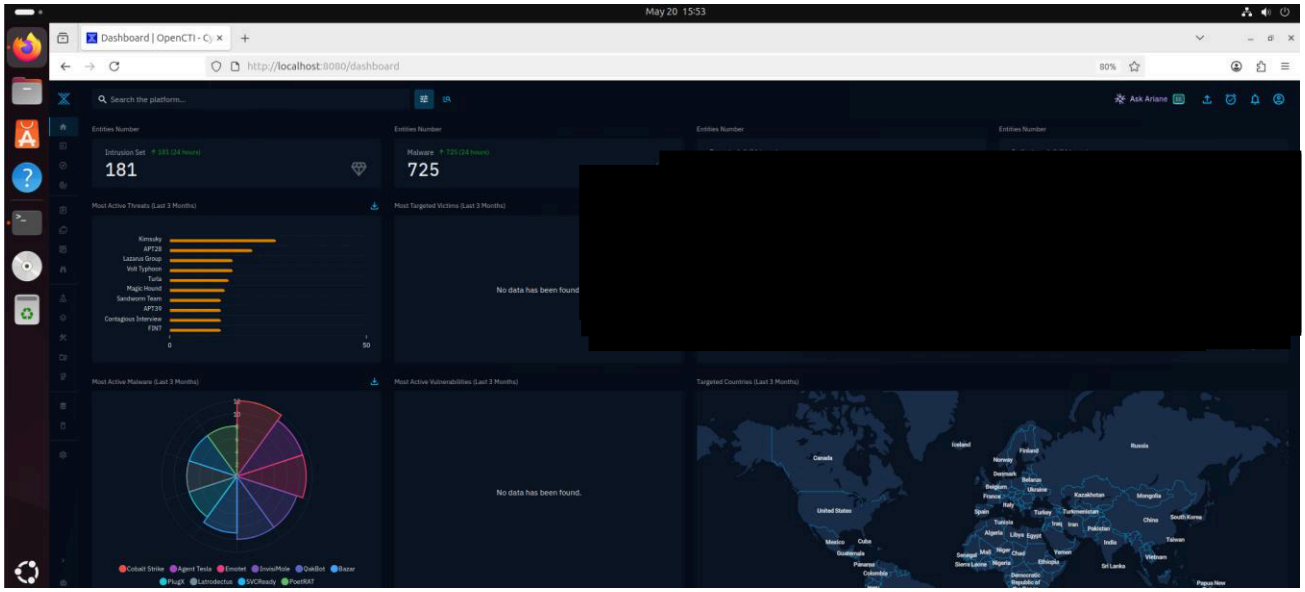


Figure 6 OpenCTI running - dashboard showing connected threat feeds and knowledge graph

Figure 6 shows OpenCTI 7.x platform deployed locally via Docker Compose on the Ubuntu VM, showing the main dashboard with entity counts and the active MITRE ATT&CK connector. Deployment was conducted on 2026-05-20 as part of the threat intelligence analysis for Kimai 2.56.0.

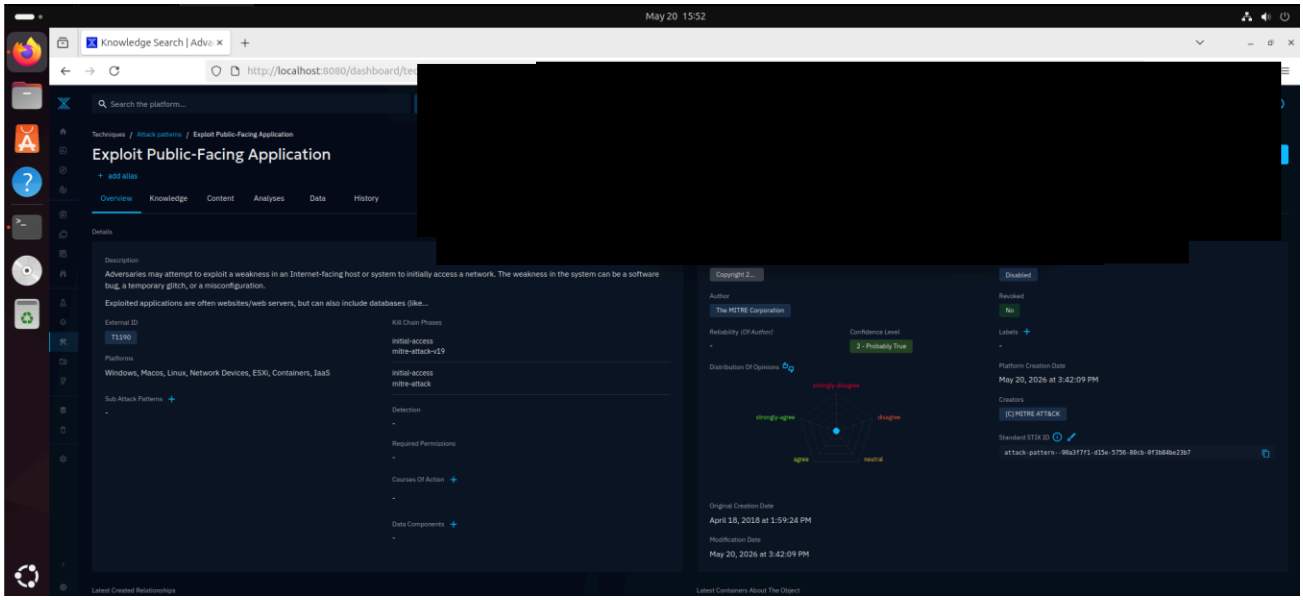


Figure 7 OpenCTI search results for PHP/Symfony - ATT&CK Techniques Search

Figure 7 shows OpenCTI attack pattern search result for T1190 (Exploit Public-Facing Application), the primary initial-access technique applicable to Kimai's REST API and web interface. This technique maps directly to the IDOR and SSTI threat scenarios identified in the STRIDE and PASTA analyses.

Relevant findings from the OpenCTI knowledge base:

- MITRE ATT&CK Technique T1190 (Exploit Public-Facing Application) is the primary initial-access technique applicable to Kimai's REST API, mapped to the STRIDE threat scenarios identified in the Threat Modeling Report.

- T1059.004 (Unix Shell via Server-Side Script) maps directly to the historical SSTI/RCE advisory chain (CVE-2023-46245, GHSA-jg2j), where template injection achieves shell command execution via PHP's exec() or system() wrappers inside the Docker container.
- T1548 (Abuse Elevation Control Mechanism) maps to the BOPA vulnerability class documented in CVE-2026-40486 and GHSA-9g2q, where low-privilege users manipulate application-layer access control attributes.
- CVE-2026-45802 does not appear in the OpenCTI KEV feed as of the scan date (2026-05-19), consistent with its same-day disclosure status. This confirms the value of direct composer audit over intelligence-feed-only approaches for freshly disclosed vulnerabilities.

Part 4: Responsible Vulnerability Disclosure

4.1 Assessment Scope and Authorization

All security testing was conducted exclusively within the authorized scope established in Lab 1. Kevin Papst, the sole Kimai project maintainer, granted explicit permission via two channels:

- GitHub Discussions thread #5941: "Hey Guys, cool project that you chose there, looking forward for the collaboration. Please get in touch via email support@kimai.org to discuss details." - Kevin Papst (kevinpapst), 2026-05-08.
- Email response to support@kimai.org: Maintainer confirmed that local testing, DevSecOps pipeline work, and responsible vulnerability reporting are acceptable, and directed the group to Kimai's SECURITY.md and bughunter documentation for disclosure procedures.

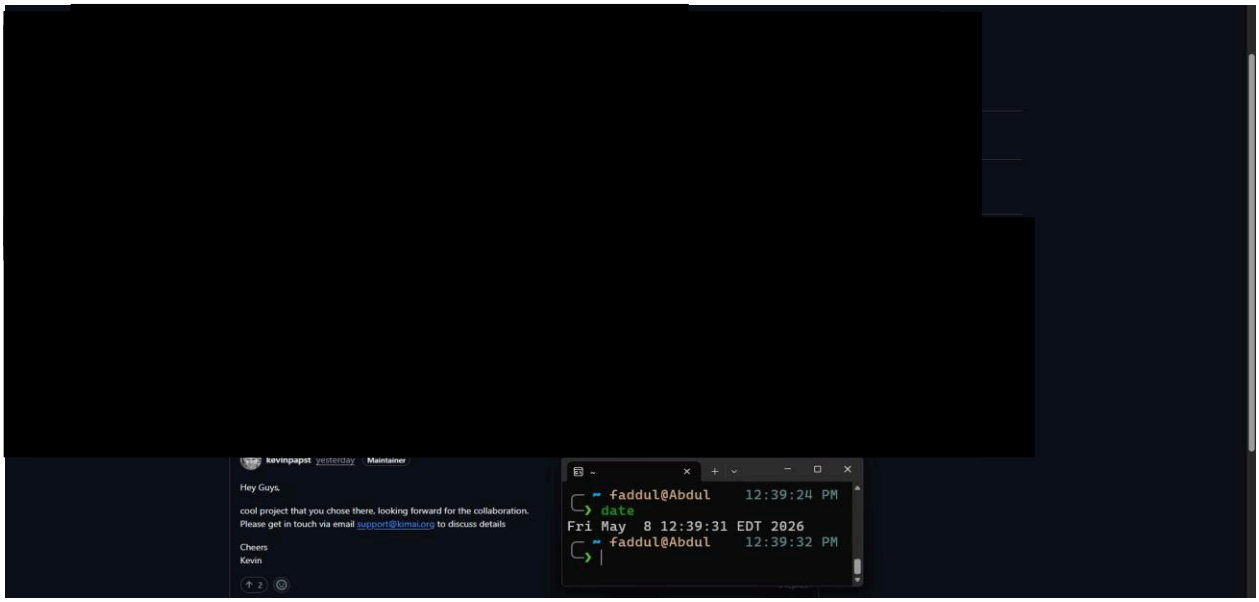


Figure 8 Kimai maintainer approval for student security research collaboration

Figure 8 shows the GitHub Discussions thread where our group requested permission to collaborate with the Kimai project for security testing. The Kimai maintainer, Kevin Papst, replied positively and directed the group to contact support@kimai.org to discuss details.


[redacted]

[EXTERNAL EMAIL] Hi

[redacted]m ,

thanks for choosing Kima

I guess it wasn't my rece
https://www.linkedin.com/

> For our final project gra

Ok. So review rules are the same everywhere: the more code, the less likely it gets a review. Either split it into small logical blocks or come up with very simple code that is easy to understand - best is a combination of both.

> and is open to reusing the majority of the pipeline components

If it brings value to the project and is stable (not much maintenance required), I will definitely do that. Better security is extremely valuable, especially in the upcoming times with stronger and stronger .

> We are also each individually required to discover and responsibly report at least one confirmed security bug to the maintainers.

That's an interesting assignment. Then probably start looking at the invoice twig renderer, it caused 50% of all issues so far.

> As part of our course requirements, we must also maintain ongoing communication with the project team

Ok. Emails are fine, but I get a lot of them every day, so my answer is most likely not instantly (also different timezones). For real security issues, use GitHub security advisories, as explained in https://www.kimai.org/documentation/bughunter.html

> Does the Kimai team grant us permission to conduct security testing on a locally deployed instance of Kimai as part of our university course?

See link above. I am happy about every contribution. I have cloud and demo environments and I can't prevent bad actors from trying to break in or out of those ... so you are more than welcome to test locally.

> Are you open to reviewing and potentially reusing the DevSecOps automation pipeline we build?

Yes. As explained above: small chunks preferred, one after another, for review.

> What is your preferred process for us to responsibly report any security vulnerabilities we discover?

Always GitHub advisories. But we can discuss them via email first, as there seems to be a misalignment sometimes between devs and researchers about what is a bug and what a security issue.
But I guess / hope you will not come up with dozens of them anyway.

> Is this email the best channel for ongoing communication throughout the semester, or would you prefer GitHub Discussions or another platform?

Email is fine.

> A written confirmation from your side would be very helpful for our course documentation.

Go ahead!

> Thank you again for your time and support — we are excited to contribute to Kimai!

I am excited to see where this is going. Happy to support you, while you improve Kimai.
Let's also do a blog post and LinkedIn / Mastodon post about it after the first results or/and at the end of the project. Btw what is the timeline of your project?

Lets connect on LinkedIn/Mastodon.

Best regards,
Kevin

—
Kevin Papst
Kimai - Time-Tracking for you and your team
Web: www.kimai.org
Share your feedback at https://love.kimai.org/

Figure 9 Email response from Kimai maintainer confirming collaboration and security testing scope

Figure 9 shows Kevin Papst from the Kimai project responding to our follow-up email after the GitHub discussion. The maintainer confirms that local testing, DevSecOps pipeline work, and responsible vulnerability reporting are acceptable, while also pointing the group to Kimai's security advisory and bug bounty documentation. This provides stronger evidence of maintainer engagement and confirms that our testing should remain local and responsibly reported.

4.2 Disclosure Status

No new vulnerabilities beyond those documented in Kimai's published advisory history were confirmed during this assessment. The findings identified by Nmap, Nikto, ZAP, and Composer Audit are either already publicly known configuration misconfigurations (missing security headers, Apache version disclosure) or published CVEs (CVE-2026-45802 in setasign/fpdi) with existing remediation paths.

The security header deficiencies identified by Nikto and ZAP (missing HSTS, CSP, Permissions-Policy, Referrer-Policy, and X-Content-Type-Options) are configuration-level issues in the default Docker deployment, not vulnerabilities in Kimai's application code. These findings were noted for inclusion in our security recommendations to the maintainer but do not meet the threshold for a private security advisory submission.

The CVE-2026-45802 finding in setasign/fpdi is a publicly disclosed third-party dependency vulnerability. It was reported to the PHP Security Advisories Database on 2026-05-19 and is already documented in the GHSA database. No additional disclosure action is required from our group.

4.3 Disclosure Process (Standing Procedure)

The following responsible disclosure process has been established and will be followed if novel vulnerabilities are identified during future labs or the final project:

- Discovery and documentation: Vulnerability identified and documented with reproducible steps, affected component, CVSS scoring, and recommended remediation.
- Secure communication: Submission via private GitHub Security Advisory at https://github.com/kimai/kimai/security/advisories/new, with a copy to support@kimai.org, consistent with maintainer's stated preferences in SECURITY.md.
- Embargo period: No public disclosure until the maintainer has acknowledged the report and either released a patch or confirmed no further action is required.
- Evidence submission: Screenshots of disclosure submission and any maintainer response will be included with the relevant lab deliverable.

This process is consistent with Kimai's SECURITY.md, the bughunter documentation at https://www.kimai.org/documentation/bughunter.html, and Seneca Polytechnic's academic integrity policy requirements for responsible vulnerability disclosure.

Part 5: Cybersecurity Testing Strategy

5.1 Risk-Based Test Prioritization

Testing areas are prioritized by the risk scores assigned in the PASTA Stage 7 risk matrix from the Threat Modeling Report, combining likelihood and impact. Each area is mapped to the OWASP Web Security Testing Guide (WSTG) test IDs that cover it. Prioritization is informed by both the STRIDE analysis and the tool findings from this assessment.

Table 7 Risk-based test prioritization for Kimai 2.56.0 based on PASTA Stage 7 risk matrix and tool findings.

#	Test Area	Justification	WSTG Coverage
1	Twig template injection (SSTI)	Four historical advisories; recurring sandbox escapes; admin-accessible; RCE potential.	WSTG-INPV-18 (SSTI); WSTG-ATHN-06
2	API authorization (IDOR / BOPA / mass assignment)	Three historical advisories; pattern of field-level authorization gaps in UserController.	WSTG-ATHZ-01; WSTG-ATHZ-04; WSTG-BUSL-08
3	HTTP security header remediation	Five missing headers confirmed by Nikto. HSTS absence most critical.	WSTG-CONF-07; WSTG-CONF-10
4	Output encoding (XSS)	CVE-2026-40479 confirmed systemic escaping gap in KimaiEscape.js.	WSTG-INPV-01; WSTG-INPV-02; WSTG-CLNT-01
5	CVE-2026-45802 (FPDI DoS)	Freshly disclosed; present in vendor tree; requires admin access; upgrade path available.	WSTG-INPV-13; Manual PoC
6	Dependency supply chain	doctrine/cache abandoned; Docker Scout reports 1 Critical, 9 High in base layers.	WSTG-CONF-05; continuous via composer audit

5.2 Cybersecurity Testing Plan

Objectives

- Confirm or rule out additional sandbox escape vectors in the Twig invoice template engine beyond those addressed in v2.46.0 and v2.56.0.
- Enumerate all PATCH and PUT API endpoints and verify that every writable field in each request body is guarded by a corresponding Voter authorization check.
- Assess all user-controlled output contexts in the UI for XSS sink points not covered by escapeForHtml() or Twig's auto-escaping.
- Upgrade setasign/fpdi and verify that CVE-2026-45802 is remediated in the test environment.
- Run an authenticated ZAP active scan to surface authorization findings not visible in the unauthenticated scan conducted in this lab.

Methods and Tools

Table 8 Testing methods and tools for Kimai 2.56.0 security assessment (Labs 3-5 and Final Project).

Test Area	Tool	Method
Automated scan (authenticated)	OWASP ZAP	Configure ZAP with admin credentials and API token. Run Active Scan against http://192.168.43.146:8920. Include Ajax Spider for JS-rendered UI.

Test Area	Tool	Method
Proxy-based manual testing	Burp Suite CE	Intercept API requests in authenticated session; replay modified payloads to test field-level authorization.
SSTI fuzzing	Burp Suite CE / manual	Upload Twig templates with progressively complex expressions: {{7*7}}, {{constant('PHP_OS')}}, {{app.request.server.get('DATABASE_URL')}}.
Mass assignment enumeration	curl / manual	For each API endpoint accepting a request body, submit all fields from the OpenAPI schema with ROLE_USER token and compare response.
XSS testing	Burp Suite CE / browser	Inject payloads into all text input fields; verify output encoding in HTML, attribute, and JavaScript contexts.
Dependency scanning	composer audit / OWASP Dependency-Check	Run after any composer update to confirm no new advisories introduced.
Header verification	Nikto / curl	Verify security headers after any Apache or Symfony configuration changes: curl -I http://192.168.43.146:8920.

Testing Schedule

Table 9 Planned testing schedule for the remainder of the WAS705 semester.

Week	Activity	Tool	Owner
Week 1 (Lab 2)	Threat modeling; initial scans (Nmap, Nikto, ZAP, composer audit)	Nmap, Nikto, ZAP, composer audit	All
Week 2	OWASP ZAP authenticated scan; SSTI fuzzing; XSS enumeration	ZAP, Burp Suite CE, browser	,
Week 3	Mass assignment enumeration across all API endpoints; IDOR cross-account testing	Burp Suite CE, curl	Abdul-Ramon
Week 4	Dependency-Check with NVD API key; CVE-2026-45802 PoC + patch verification	OWASP Dependency-Check, composer	All
Week 5	Security header remediation testing; final report; maintainer disclosure	Nikto, curl	All

Manual vs Automated Testing Balance

Automated tools (ZAP, Nikto, Composer Audit) provide broad coverage of known vulnerability signatures and configuration issues efficiently. However, the most significant findings in Kimai's advisory history - SSTI sandbox escapes and field-level authorization gaps - all required manual analysis of application logic and source code. The testing plan therefore allocates a minimum of 60% of testing effort to manual proxy-based testing and source code review, using automated tools primarily for reconnaissance, header verification, and dependency monitoring.

Task Checklist

Table 10 Lab 2 Vulnerability Assessment Report task completion checklist.

Task	Status
Network reconnaissance scan (Nmap)	**Done**
Web server analysis (Nikto)	**Done**
Web application scan (OWASP ZAP)	**Done**
Dependency analysis (Composer Audit)	**Done**
Third-party library security review	**Done**
Historical version comparison	**Done**
Competitive security posture comparison	**Done**
Threat intelligence tool deployment (OpenCTI)	**Done**
Responsible disclosure summary	**Done**
Cybersecurity testing plan	**Done**
Risk-based test prioritization	**Done**
AODA-compliant PDF report submitted	**Done**
Video demonstration recorded (5-10 min)	**Done**

Conclusion

In conclusion, this vulnerability assessment confirmed several actionable findings in the default Kimai 2.56.0 Docker deployment, including the absence of HTTP Strict Transport Security, missing Content Security Policy and anti-clickjacking headers, Apache version disclosure, and one active dependency CVE (CVE-2026-45802 in setasign/fpdi) with an available patch. No novel vulnerabilities in Kimai's own codebase were discovered during this assessment; all findings fall into known configuration-level misconfigurations or a published third-party CVE, and the responsible disclosure process established in Lab 1 remains in place for any future findings. Threat intelligence gathered through OpenCTI confirmed that MITRE ATT&CK techniques T1190, T1059.004, and T1548 map directly to Kimai's confirmed vulnerability history and will guide the manual testing priorities in Lab 3, where an authenticated ZAP scan and API authorization testing are planned to assess the authorization gaps that unauthenticated automated scanning cannot reach.

References

Anchore. (2024). *Syft*. GitHub. https://github.com/anchore/syft

Court of Justice of the European Union. (2019, May 14). *Federación de Servicios de Comisiones Obreras (CCOO) v. Deutsche Bank SAE, Case C-55/18*. https://curia.europa.eu/juris/liste.jsf?num=C-55/18

CycloneDX. (2024). *CycloneDX PHP Composer plugin*. GitHub. https://github.com/CycloneDX/cyclonedx-php-composer

GitHub Advisory Database. (2023). *GHSA-fjhg-96cp-6fcw: Authenticated SSTI to RCE via uploaded Twig file in Kimai*. https://github.com/advisories/GHSA-fjhg-96cp-6fcw

National Vulnerability Database. (2026). *CVE-2026-40479: Stored XSS via incomplete HTML attribute escaping in Kimai*. https://nvd.nist.gov/vuln/detail/CVE-2026-40479

OWASP Foundation. (2024). *Web Security Testing Guide v4.2*. https://owasp.org/www-project-web-security-testing-guide/

Papst, K. (2026). *Kimai: Time-tracking application*. GitHub. https://github.com/kimai/kimai

Papst, K. (2026). *Kimai bughunter documentation*. Kimai. https://www.kimai.org/documentation/bughunter.html

Papst, K. (2026). *Kimai security policy*. GitHub. https://github.com/kimai/kimai/blob/main/SECURITY.md