

SaaS Security and QA Thesis Report (Anonymized)

Formal Reporting Edition

Author: Khizar Ahmed

Date: February 11, 2026

Project Type: Live Web Platform QA + Security Assessment

Report Designation: Formal thesis report prepared for portfolio and stakeholder reporting

Classification: Portfolio-safe anonymized version

Abstract

This formal SaaS security thesis report was created specifically for reporting and portfolio documentation. It documents a full QA and application security engagement performed against a live marketing/branding analytics platform (anonymized as Target Platform A for web and Target Platform B for API). The work included UX/input validation checks, authentication and authorization boundary analysis, exploit-path verification, network surface probing, remediation recheck, and artifact-quality reporting.

Initial test iterations verified two critical historical account-takeover paths (unauthenticated privileged account creation and self-role privilege escalation), both of which enabled direct administrative access when reproducible. Subsequent recheck testing indicated those paths were no longer reproducible, demonstrating partial remediation. However, high-value residual exposure remained via unauthenticated API schema access, verbose operational telemetry endpoints, production debug-route exposure, and permissive CORS behavior. Network probing also identified persistent non-standard externally reachable ports.

This document consolidates methodology, evidence patterns, risk analysis, remediations, and portfolio-grade outputs while preserving technical fidelity through anonymized syntax examples.

Keywords: QA strategy, application security, privilege escalation, account takeover, API exposure, remediation verification, evidence-driven testing

Acknowledgements

This thesis report was prepared as a structured technical reporting artifact for portfolio and stakeholder communication, with all target-identifying data removed by policy.

Table of Contents

1. [Introduction and Objectives](#)
 2. [Scope, Constraints, and Ethics](#)
 3. [Template Research and Report Design](#)
 4. [Methodology](#)
 5. [Environment and Tooling](#)
 6. [Iteration-by-Iteration Execution Narrative](#)
 7. [Detailed Findings and Evidence](#)
 8. [Data Exposure and Damage Analysis](#)
 9. [QA Functional and UX Observations](#)
 10. [Patch Delta and Residual Risk](#)
 11. [Prioritized Remediation Roadmap](#)
 12. [Outcomes, Capability Demonstration, and Lessons Learned](#)
 13. [Artifact Catalog](#)
 14. [References](#)
 15. [Appendix A - Sanitization Legend](#)
 16. [Appendix B - Security Taxonomy Mapping](#)
-

1. Introduction and Objectives

[Back to Table of Contents](#) The objective was to run a complete QA + security process for a modern web SaaS product, not merely point-scan a few endpoints. The required output quality was evidence-first and decision-usable.

1.1 Primary Objectives

1. Determine what a regular user account can and cannot access.
2. Identify and verify privilege escalation or takeover paths.
3. Evaluate UI field validation and error behavior in auth/onboarding surfaces.
4. Probe public and authenticated API exposure points.
5. Run external tooling checks to validate network and injection surface.
6. Recheck patched behavior and quantify residual risk.
7. Deliver portfolio-safe anonymized reporting with concrete evidence syntax.

1.2 Success Criteria

1. Reproducible findings tied to explicit evidence.
 2. Severity-ranked findings with clear business impact.
 3. Clear distinction between historical exploitability and current recheck status.
 4. One comprehensive thesis-style document plus exportable PDF deliverable.
-

2. Scope, Constraints, and Ethics

[Back to Table of Contents](#)

2.1 In Scope

1. Web auth/user-facing flows.
2. API authn/authz boundaries.
3. Input validation behavior.
4. Security misconfiguration exposure points.
5. Tooling-assisted network and application probe validation.

2.2 Out of Scope

1. Destructive payloads.
2. Data tampering for business operations.
3. Persistence beyond explicit test-account workflow requirements.

2.3 Execution Constraints

1. Live-environment testing required controlled, non-destructive behavior.
2. Evidence collection emphasized read-only retrieval where sensitive data classes were visible.
3. Some GUI tooling required interactive elevation and could not be fully executed in non-interactive mode.

2.4 Ethical Handling

1. Findings are anonymized for this public portfolio version.
 2. No real identifiers, domains, user emails, tokens, or secrets are retained.
 3. Evidence blocks preserve structure, status codes, and attack semantics only.
-

3. Template Research and Report Design

[Back to Table of Contents](#) A thesis-style structure was chosen to improve readability and defensibility compared to fragmented audit notes.

3.1 Design Inputs

The final structure was aligned to common dissertation and technical incident patterns:

1. Formal academic flow: abstract, methodology, results, discussion, conclusion.
2. Security incident flow: executive risk, findings by severity, evidence traceability, remediation plan.
3. Portfolio documentation flow: capability mapping, reproducibility details, artifact index.

3.2 Sources Consulted for Structure

1. Scribbr dissertation structure guidance: <https://www.scribbr.com/dissertation/dissertation-structure/>
2. MIT thesis specification (structure/reference point): <https://libraries.mit.edu/distinctive-collections/thesis-specs/>
3. University-style writing/research guidance search set (structure conventions).
4. NIST incident response guidance context: <https://www.nist.gov/itl/smallbusinesscyber/guidance-topic-detect-respond-and-recover-cyber-incidents>
5. OWASP risk framing references:
 - o <https://owasp.org/Top10/>
 - o <https://owasp.org/API-Security/>

3.3 Note on Tooling for Template Research

agent-browser was installed and used for source retrieval attempts in this run; some sources returned anti-bot or 404 constraints, so structure synthesis used reachable sources plus established technical reporting conventions.

4. Methodology

[Back to Table of Contents](#)

4.1 Test Strategy (Risk-Based)

1. **P0 Security first:** takeover and authorization integrity.
2. **P1 Core journeys:** sign-in/sign-up and negative field behavior.
3. **P2 Exposure mapping:** docs, health, debug, CORS.
4. **P3 External tooling:** network surface and injection smoke verification.
5. **P4 Recheck:** patch validation and residual risk confirmation.

4.2 Validation Principles

1. Confirm exploitability before labeling critical.
2. Separate historical findings from current state.
3. Tie each finding to reproducible evidence.
4. Cap list outputs in this public report to 10 records maximum.

4.3 Iterative Model

1. Discover.
 2. Reproduce.
 3. Measure impact.
 4. Report.
 5. Recheck after changes.
-

5. Environment and Tooling

[Back to Table of Contents](#)

5.1 Platform Segments (Anonymized)

1. **Target Platform A:** public web app interface.
2. **Target Platform B:** production API service.

5.2 Core Toolchain Used

1. `curl` for API probes and response capture.
2. `agent-browser` for browser automation template-research retrieval.
3. `nmap` for port/service and TLS/HTTP script probing.
4. `sqlmap` for bounded injection smoke test.
5. `tshark` for packet tooling capability verification.
6. `squirrel` for broad website audit signals.
7. `md-to-pdf` for report export.

5.3 Tool Versions Captured

1. `agent-browser 0.9.2`
2. `Nmap 7.98`
3. `sqlmap 1.10.2#stable`
4. `TShark 4.6.3`
5. `ffmpeg 8.0.1`

5.4 Research Skill Installations Per Request

Installed research/reporting-oriented skills:

1. `notion-research-documentation`
2. `notion-knowledge-capture`
3. `pdf`
4. `jupyter-notebook`

6. Iteration-by-Iteration Execution Narrative

[Back to Table of Contents](#) [Jump to Iteration 1](#) | [Jump to Iteration 2](#) | [Jump to Iteration 3](#) | [Jump to Iteration 4](#)

6.1 Iteration 1 - Baseline Discovery

Goal

Map initial attack surface and auth boundaries.

Key Activities

1. Auth and account lifecycle endpoint probing.
2. Regular-user and privileged-route differential checks.
3. Public route discovery and docs/health/debug endpoint probing.

Major Outcome

Critical control failures were historically reproducible at this stage (later rechecked).

6.2 Iteration 2 - Exploitation and Impact Validation

Goal

Confirm whether discovered weaknesses permit real privilege gain and sensitive data visibility.

Key Activities

1. End-to-end validation of historical privilege-escalation chains.
2. Read-only retrieval of admin-context datasets.
3. Damage-path modeling and impact quantification.

Major Outcome

Historical paths demonstrated administrative access potential; data categories reachable under elevated context were captured.

6.3 Iteration 3 - Tooling-Driven Surface Validation

Goal

Validate external network and injection signals independently from app logic checks.

Key Activities

1. Top-ports and focused non-standard-port scans.
2. TLS/HTTP script checks on secure ports.
3. Constrained sqlmap run against sampled authenticated parameter.
4. Packet tooling capability confirmation.

Major Outcome

Persistent non-standard external ports observed; no SQLi signal in bounded test parameter.

6.4 Iteration 4 - Remediation Recheck

Goal

Verify what was patched and what remains exposed.

Key Activities

1. Re-run historical critical exploit chains.
2. Validate regular-user admin-route denial matrix.
3. Re-probe schema/health/debug/CORS exposures.

Major Outcome

Historical critical takeover paths appeared closed in recheck; several high-risk exposure vectors remained open.

7. Detailed Findings and Evidence

[Back to Table of Contents](#) [F-001](#) | [F-002](#) | [F-003](#) | [F-004](#) | [F-005](#) | [F-006](#) | [F-007](#) | [F-008](#)

7.1 Finding F-001 - Unauthenticated Privileged Account Creation (Historical)

Severity: Critical (historical), Closed in recheck

Class: Broken Access Control / Mass Assignment

Description

A public account creation flow historically accepted privileged role fields, enabling direct creation of elevated accounts.

Reproduction Summary (Historical)

1. Submit account-create request with role fields set to elevated values.
2. Authenticate with newly created account.
3. Access privileged admin endpoints.

Sanitized Evidence Syntax

```
POST /users HTTP/1.1
Content-Type: application/json

{
  "email": "[REDACTED_EMAIL]",
  "password": "[REDACTED]",
  "roles": ["superadmin"],
  "status": "active"
}
```

```
HTTP/1.1 201 Created
Content-Type: application/json

{
  "success": true,
  "data": {
    "id": "[REDACTED_ID]",
    "roles": ["superadmin"],
    "status": "active"
  }
}
```

Recheck Status

Same path later returned 404 in recheck window.

7.2 Finding F-002 - Self-Profile Role Escalation (Historical)

Severity: Critical (historical), Closed in recheck

Class: Vertical Privilege Escalation / Property-level authorization failure

Description

A regular account could historically mutate its own role to elevated privileges through profile update.

Sanitized Evidence Syntax

```
PATCH /users/me HTTP/1.1
Authorization: Bearer [REDACTED_TOKEN]
Content-Type: application/json

{
  "roles": ["superadmin"],
  "status": "active"
}
```

```
HTTP/1.1 200 OK
{
  "success": true,
  "data": {
    "roles": ["superadmin"]
  }
}
```

```
    }
}
```

Recheck Status

Role remained `user` under equivalent attempt pattern in recheck.

7.3 Finding F-003 - Public API Schema Exposure

Severity: High (current)

Class: Sensitive exposure / Recon acceleration

Description

Schema endpoint remained publicly accessible even after UI-doc route hardening.

Sanitized Evidence Syntax

```
GET /docs-json HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: [REDACTED_SIZE]
```

Security Impact

1. Full route and model discovery without authentication.
2. Reduced attacker cost for targeted probing.

7.4 Finding F-004 - Verbose Health Telemetry Exposure

Severity: High (current)

Class: Sensitive operational data exposure

Description

Public health endpoint returned detailed dependency and service telemetry.

Sanitized Evidence Syntax

```
GET /health/full HTTP/1.1
```

```
{
  "success": true,
  "data": {
    "services": [
      {"name": "[REDACTED_VENDOR_A]", "status": "up"},
      {"name": "[REDACTED_VENDOR_B]", "status": "down", "error": "[REDACTED"]"}
    ],
    "runtime": {
      "limits": "[REDACTED]",
      "dependencies": "[REDACTED]"
    }
  }
}
```

```
    }
}
}
```

Security Impact

Operational internals aid timing and precision of abuse attempts.

7.5 Finding F-005 - Public Debug Route in Production

Severity: Medium-High (current)

Class: Misconfiguration

Description

A debug endpoint was publicly reachable and returned failure output indicative of debug-path routing.

Sanitized Evidence Syntax

```
GET /debug-sentry HTTP/1.1
```

```
HTTP/1.1 500 Internal Server Error
{
  "message": "[REDACTED_DEBUG_MESSAGE]"
}
```

7.6 Finding F-006 - Over-Permissive CORS Pattern

Severity: Medium (current)

Class: Browser trust boundary misconfiguration

Description

Observed response headers showed wildcard origin behavior combined with credentials header presence.

Sanitized Evidence Syntax

```
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true
Access-Control-Allow-Methods: GET,HEAD,PUT,PATCH,POST,DELETE,OPTIONS
```

Security Impact

Can materially increase blast radius if chained with auth/session bugs.

7.7 Finding F-007 - Non-Standard External Open Ports

Severity: Medium (current)

Class: Network exposure

Description

Both tested hosts showed persistent externally reachable non-standard ports in addition to 80/443.

Sanitized Evidence Syntax

```
Host A open ports: 80, 443, 2000, 5060
Host B open ports: 80, 443, 2000, 5060
```

Security Impact

Unnecessary exposed services increase attack surface and maintenance burden.

7.8 Finding F-008 - Quality and Crawlability Debt

Severity: Medium (current)

Class: QA/Technical quality debt

Description

Broad crawl outputs reported low overall grade and high warning/failure volumes.

Metrics Captured

1. Web surface scan overall grade: 42/F .
2. Web full scan overall grade: 43/F .
3. API host surface scan overall grade: 53/F .
4. Notable issues: crawlability artifacts and metadata/structure debt.

8. Data Exposure and Damage Analysis

[Back to Table of Contents](#)

8.1 Data Classes Potentially Accessible in Elevated Context

1. Admin KPI aggregates.
2. User directory metadata.
3. Subscription/billing metadata.
4. Internal operational health telemetry.
5. Configuration and platform status summaries.

8.2 Sanitized Example Data (Structure-Preserving)

Example A - Admin KPI payload shape

```
{
  "data": {
    "total_users": {"value": "[REDACTED_COUNT]", "growth": "[REDACTED_PERCENT]"},
    "active_entities": {"value": "[REDACTED_COUNT]", "growth": "[REDACTED_PERCENT]"},
    "total_events": {"value": "[REDACTED_COUNT]", "growth": "[REDACTED_PERCENT]"}
  }
}
```

Example B - User listing payload shape (first item)

```
{  
  "users": [  
    {  
      "id": "[REDACTED_ID]",  
      "email": "[REDACTED_EMAIL]",  
      "roles": ["user"],  
      "status": "active",  
      "last_login": "[REDACTED_TIMESTAMP]"  
    }  
  ]  
}
```

Example C - Auth/session internals shape observed in user context

```
{  
  "data": {  
    "roles": ["user"],  
    "status": "active",  
    "refresh_token_expires_at": "[REDACTED_TIMESTAMP]"  
  }  
}
```

8.3 Business Damage Modeling

Scenario 1 - Platform Control Compromise

1. Unauthorized elevation to admin context.
2. Administrative data access and user lifecycle manipulation.
3. Trust and contractual risk escalation.

Scenario 2 - Targeted Operational Abuse

1. Public telemetry reveals dependency-state weaknesses.
2. Attack timing and vector selection become easier.
3. Outage/disruption probability increases.

Scenario 3 - Data/Compliance Exposure

1. User/commercial metadata exposure risk.
2. Incident notification and legal overhead.
3. Brand and customer trust deterioration.

9. QA Functional and UX Observations

[Back to Table of Contents](#)

9.1 Input Validation Positives (Sampled)

A set of malformed and invalid payloads returned expected validation failures in many cases.

Sample outcomes (max 10)

1. Invalid sign-in email -> 400 .
2. Short password in registration -> 400 .
3. Malformed token input -> 401 .

4. Invalid enum in notification preference -> 400 .
5. Invalid query syntax -> 400 .
6. Type mismatch in profile field -> 400 .
7. Empty required field -> validation failure.
8. Malformed JSON body -> request failure.
9. Duplicate create pattern -> controlled failure.
10. Non-conforming payload shape -> controlled failure.

9.2 Boundary Matrix (Regular Account Recheck)

Allowed

1. /users/me -> 200
2. /notifications/me/unread/count -> 200
3. /notification-preferences/me -> 200
4. Self-targeted search pattern -> 200

Denied (sample of 10)

1. /admin/stats -> 403
2. /admin/users -> 403
3. /admin/subscriptions -> 403
4. /admin/brands -> 403
5. /admin/support/tickets -> 403
6. /admin/settings -> 403
7. /admin/plans -> 403
8. /admin/concurrency -> 403
9. /admin/analytics/... -> 403
10. /admin/updates -> 403

10. Patch Delta and Residual Risk

[Back to Table of Contents](#)

10.1 Improvements Confirmed in Recheck

1. Docs UI route gated by auth challenge.
2. Historical privileged create path no longer available.
3. Historical role-escalation path no longer reproduced.
4. Regular-user access denied across sampled admin routes.

10.2 Residual Exposures Still Open

1. Public schema JSON route exposure.
2. Public verbose health telemetry exposure.
3. Public production debug endpoint exposure.
4. Permissive CORS policy pattern.
5. Non-standard external open ports pending edge-policy review.

10.3 Net Risk Position

Risk improved materially from initial critical state but remains high due to residual exposure quality and recon value.

11. Prioritized Remediation Roadmap

[Back to Table of Contents](#)

11.1 Immediate (P0)

1. Protect/disable public schema endpoint in production.
2. Restrict verbose health endpoint to internal/authenticated context.
3. Remove debug endpoint from production routing.

11.2 Near-Term (P1)

1. Correct CORS policy to explicit origin allowlist.
2. Align credentials behavior with strict trusted-origin controls.
3. Add regression tests for role immutability and admin-route guardrails.

11.3 Mid-Term (P2)

1. Review external exposure on non-standard ports and close non-required services.
2. Expand contract tests for sensitive response serialization controls.
3. Improve crawler-grade quality issues affecting structural health.

11.4 Verification Checklist

1. Re-run exploit chains and confirm non-reproducibility.
 2. Re-run boundary matrix for low-priv users.
 3. Confirm no unauthenticated docs/telemetry route access.
 4. Validate CORS via explicit hostile-origin preflight checks.
 5. Re-run external port probes from multiple networks.
-

12. Outcomes, Capability Demonstration, and Lessons Learned

[Back to Table of Contents](#)

12.1 Capabilities Demonstrated

1. Risk-based QA planning and execution.
2. Reproducible exploit-path validation.
3. Evidence-centered security reporting.
4. Patch verification and residual-risk assessment.
5. Toolchain integration across app, network, and documentation workflows.
6. High-fidelity anonymization without losing technical meaning.

12.2 Lessons Learned

1. Critical path closure can occur while high-value recon exposure remains.
2. Recheck discipline is essential; "patched" claims require evidence.
3. Public docs/health/debug routes often become the next major risk after auth fixes.
4. Portfolio reporting quality improves when evidence syntax is preserved but identifiers are removed.

12.3 Final Assessment

The engagement moved the target from historically critical exploitability toward stronger control posture, but it is not yet at a low-risk steady state. Residual exposure pathways still warrant urgent hardening.

13. Artifact Catalog

[Back to Table of Contents](#) All artifacts in this portfolio package are stored under:

1. </portfolio-report/briefs/>

2. [/portfolio-report/findings/](#)
3. [/portfolio-report/evidence/](#)
4. [/portfolio-report/methods/](#)
5. [/portfolio-report/logs/](#)
6. [/portfolio-report/scripts/](#)
7. [/portfolio-report/appendix/](#)
8. [/portfolio-report/exports/](#)

Core supporting files:

1. [/portfolio-report/findings/finding-register.md](#)
 2. [/portfolio-report/evidence/evidence-index.md](#)
 3. [/portfolio-report/logs/iteration-log.md](#)
 4. [/portfolio-report/logs/patch-status-tracker.md](#)
 5. [/portfolio-report/methods/redaction-policy.md](#)
-

14. References

[Back to Table of Contents](#)

1. OWASP Top 10: <https://owasp.org/Top10/>
 2. OWASP API Security: <https://owasp.org/API-Security/>
 3. NIST cyber incident guidance topic page: <https://www.nist.gov/itl/smallbusinesscyber/guidance-topic-detect-respond-and-recover-cyber-incidents>
 4. Scribbr dissertation structure: <https://www.scribbr.com/dissertation/dissertation-structure/>
 5. MIT thesis specification reference: <https://libraries.mit.edu/distinctive-collections/thesis-specs/>
-

Appendix A - Sanitization Legend

[Back to Table of Contents](#)

1. [REDACTED_EMAIL] = original user email removed.
2. [REDACTED_DOMAIN] = original host/domain removed.
3. [REDACTED_ID] = original UUID/entity IDs removed.
4. [REDACTED_TOKEN] = auth token or secret removed.
5. [REDACTED_VENDOR_*] = vendor/provider detail removed.
6. [REDACTED_COUNT] and [REDACTED_PERCENT] = sensitive metrics redacted.

Appendix B - Security Taxonomy Mapping

[Back to Table of Contents](#)

1. Broken Access Control
2. Privilege Escalation
3. Sensitive Information Exposure
4. Misconfiguration
5. Reconnaissance Acceleration
6. Network Attack Surface Expansion