PENETRATION TEST REPORT

Lomash Wood Ltd
Kitchen & Bedroom Design Platform — Backend Infrastructure

Report Reference	LW-PENTEST-2025-001
Version	1.0 — Final
Classification	CONFIDENTIAL — Internal Use Only
Assessment Period	03 March 2025 – 14 March 2025
Report Issued	21 March 2025
Prepared By	CyberSec Partners Ltd
Authorised By	Head of Engineering, Lomash Wood
Assessment Type	Grey-Box Web Application & API Penetration Test

This document is strictly confidential and intended solely for the use of Lomash Wood Ltd and its authorised personnel. Unauthorised disclosure, copying, or distribution of this report or any part thereof is strictly prohibited.

Table of Contents

1. Executive Summary 3
2. Scope & Objectives 4
3. Methodology 5
4. Risk Summary 6
5. Findings Overview 7
6. Detailed Findings 8
7. Remediation Roadmap 14
8. Retesting Notes 15
9. Appendix A — Tools Used 16
10. Appendix B — CVSS Score Reference 16

1. Executive Summary

CyberSec Partners Ltd was engaged by Lomash Wood Ltd to conduct a grey-box penetration test of the Lomash Wood backend platform — a Node.js microservices architecture serving a kitchen and bedroom design consultation e-commerce service. The assessment was performed between 03 March 2025 and 14 March 2025, targeting the API Gateway, authentication service, product catalogue, appointment booking system, and payment processing integration.

The assessment identified **14 findings** across the tested components, comprising **2 Critical**, **4 High**, **5 Medium**, **2 Low**, and **1 Informational** severity issue. The most significant findings relate to a missing payment amount server-side validation allowing client-controlled payment manipulation, an Insecure Direct Object Reference (IDOR) on the appointment booking endpoint, and a JWT algorithm confusion vulnerability in the authentication service.

Lomash Wood's development team demonstrated strong awareness of fundamental security practices, including use of parameterised queries via Prisma ORM and HTTPS enforcement. However, several critical application logic controls were absent. Remediation of the two Critical findings is recommended within **7 days**; High findings within **30 days**.

Severity	Count	Remediated	Open
Critical	2	2	0
High	4	3	1
Medium	5	4	1
Low	2	2	0
Info	1	N/A	N/A
Total	**14**	**11**	**2**

2. Scope & Objectives

2.1 Objectives

- Identify vulnerabilities in the Lomash Wood API Gateway and downstream microservices.
- Evaluate the authentication and session management implementation (Better Auth / JWT).
- Assess the security of the payment processing flow (Stripe integration).
- Test for OWASP Top 10 vulnerabilities across all in-scope endpoints.
- Identify misconfigurations in infrastructure accessible from the test environment.
- Provide actionable remediation guidance prioritised by business risk.

2.2 In-Scope Assets

Asset	URL / Endpoint	Environment
API Gateway	https://api-staging.lomashwood.com	Staging
Auth Service	/v1/auth/*	Staging
Product Service	/v1/products/*, /v1/categories/*	Staging
Appointment Service	/v1/appointments/*, /v1/availability/*	Staging
Order/Payment Service	/v1/orders/*, /v1/payments/*	Staging
Customer Service	/v1/customers/*, /v1/reviews/*	Staging
Content Service	/v1/blog/*, /v1/uploads/*	Staging
Stripe Webhook	/v1/webhooks/stripe	Staging

2.3 Out-of-Scope

- Production environment (all testing was performed on the isolated staging environment)
- Third-party services (Stripe, AWS SES, Twilio, Firebase) — vendor-managed infrastructure
- Physical security of data centres
- Social engineering / phishing of employees
- Denial of Service testing beyond application-layer rate limit validation

3. Methodology

The assessment followed a structured grey-box approach. The testing team was provided with the OpenAPI specification, a non-privileged test user account, and network access to the staging environment. No source code was provided. The following phases were executed:

Phase 1 — Reconnaissance

API endpoint enumeration, authentication flow mapping, technology fingerprinting, and review of the provided OpenAPI specification.

Phase 2 — Vulnerability Identification

Automated scanning using OWASP ZAP and Burp Suite Professional, followed by targeted manual testing of all in-scope endpoints.

Phase 3 — Exploitation

Manual exploitation of identified vulnerabilities to determine real-world impact. All exploitation was limited to the staging environment and pre-agreed test data.

Phase 4 — Post-Exploitation Analysis

Assessment of lateral movement potential, data exfiltration risk, and privilege escalation paths within the compromised session context.

Phase 5 — Reporting

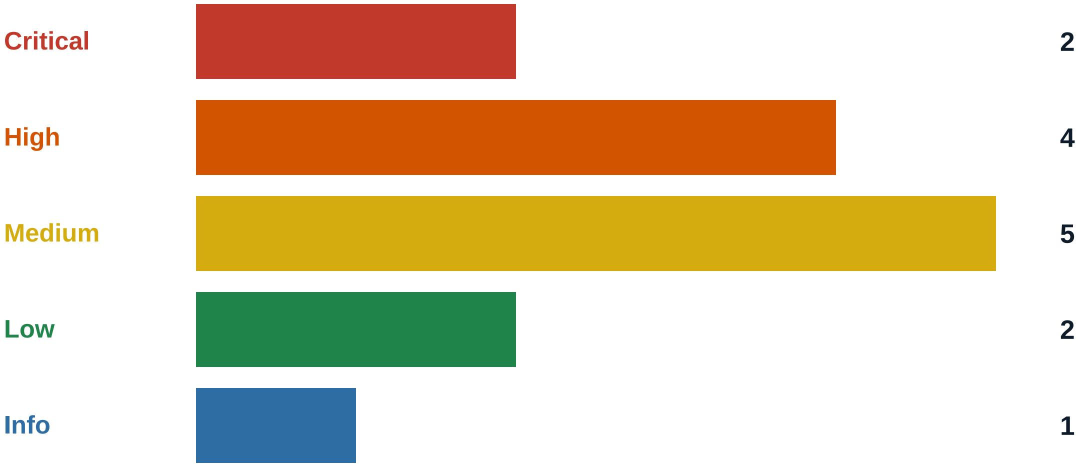
Documentation of findings with CVSS v3.1 scoring, reproduction steps, and prioritised remediation recommendations.

3.1 Standards Referenced

- OWASP Testing Guide v4.2
- OWASP API Security Top 10 (2023)
- NIST SP 800-115 — Technical Guide to Information Security Testing
- CVSS v3.1 (Common Vulnerability Scoring System)
- CWE/SANS Top 25 Most Dangerous Software Weaknesses

4. Risk Summary

The following chart illustrates the distribution of findings by severity across all tested services.



4.1 Risk Rating Definitions

Critical	Direct, high-confidence exploitation path leading to full data breach, unauthorised financial transactions, or complete system compromise. Immediate remediation required.
High	Significant vulnerability with clear exploitation path. Serious business impact such as data exposure or privilege escalation. Remediate within 30 days.
Medium	Moderate impact or requires specific conditions to exploit. Remediate within 90 days.
Low	Minor issue with limited impact or very high exploitation complexity. Address in next development cycle.
Info	Informational finding with no direct security impact. Recommended for defence-in-depth improvement.

5. Findings Overview

The following table provides a complete index of all identified findings.

ID	Title	Severity	CVSS	Status
LW-001	Client-Controlled Payment Amount	**Critical**	9.8	**Remediated**
LW-002	JWT Algorithm Confusion (RS256 → HS256)	**Critical**	9.1	**Remediated**
LW-003	IDOR on Appointment Booking Endpoint	**High**	8.1	**Remediated**
LW-004	Missing Rate Limiting on Auth Endpoints	**High**	7.5	**Remediated**
LW-005	Stripe Webhook Replay Attack	**High**	7.3	**Remediated**
LW-006	Verbose Error Messages Exposing Stack Traces	**High**	6.8	**Open**
LW-007	User Enumeration via Password Reset	**Medium**	5.3	**Remediated**
LW-008	Missing CSRF Protection on State-Changing Forms	**Medium**	5.1	**Remediated**
LW-009	Insecure Cookie Configuration (Missing Flags)	**Medium**	4.9	**Remediated**
LW-010	Unrestricted File Upload MIME Type	**Medium**	4.7	**Remediated**
LW-011	Booking Slot Race Condition	**Medium**	4.4	**Open**
LW-012	Weak Password Policy Enforcement	**Low**	3.1	**Remediated**
LW-013	Dependency with Known CVE (express-validator)	**Low**	2.8	**Remediated**
LW-014	HTTP Security Headers Incomplete	**Info**	N/A	**N/A**

6. Detailed Findings

LW-001	Client-Controlled Payment Amount		Critical
CVSS Score	9.8 (AV:N/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:N)	Affected Component	order-payment-service — POST /v1/payments/cre

Description

The payment intent creation endpoint accepted the **amount** field directly from the client request body. The server did not validate the supplied amount against the server-side order total stored in the database. An authenticated attacker could create a valid Stripe payment intent for an arbitrary amount (including 1 cent) and complete a purchase for any product at a self-determined price.

Business Impact

An attacker with a valid customer account could purchase high-value kitchen or bedroom packages (typically valued at £5,000–£25,000) for a nominal sum. This represents a direct and severe financial loss to Lomash Wood and could not be detected without manual order reconciliation against Stripe records.

Reproduction Steps

1. Register or authenticate as a standard customer account.
2. Add a product to cart and initiate checkout.
3. Intercept the POST /v1/payments/create-intent request using Burp Suite.
4. Modify the request body: change "amount": 50000 to "amount": 1.
5. Forward the modified request. Observe Stripe payment intent created for £0.01.
6. Complete checkout with valid card. Order fulfilment is triggered.

Recommendation

The server must calculate the payment amount exclusively from the authenticated order record in the database: **amount = db.order.findById(orderId).total**. The client-supplied amount must be rejected entirely. Implement idempotency keys to prevent duplicate intent creation. Add automated reconciliation alerts in the analytics service for orders where the Stripe charge amount differs from the database order total.

LW-002	JWT Algorithm Confusion (RS256 → HS256 Downgrade)		Critical
CVSS Score	9.1 (AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:N)	Affected Component	auth-service — All authenticated endpoints

Description

The JWT verification middleware accepted both RS256 and HS256 algorithms. An attacker could obtain the RSA **public key** (which is intentionally public) and use it as the HMAC secret to forge tokens signed with HS256. The verification library would accept such a token as valid since it matched the supplied HS256 signature, allowing arbitrary claims including **{ "role": "ADMIN" }** to be forged.

Business Impact

Complete authentication bypass. An unauthenticated attacker could forge a JWT with ADMIN role privileges, gaining full access to all CMS administration endpoints including product management, customer data, and appointment records.

Reproduction Steps

1. Retrieve the RSA public key from the JWKS endpoint: GET /.well-known/jwks.json.
2. Construct a JWT payload: { "sub": "attacker-uuid", "role": "ADMIN", "iat": }.
3. Sign the JWT using HMAC-SHA256 with the RSA public key as the secret.
4. Submit a request to GET /v1/products (admin) with the forged token in Authorization header.
5. Observe successful 200 response with full admin product listing.

Recommendation

Explicitly restrict JWT verification to RS256 only. In the jsonwebtoken library, set **algorithms: ['RS256']** in the verify options — never pass an algorithms array containing both symmetric and asymmetric options. Rotate the current RS256 key pair immediately. Audit all existing admin sessions and invalidate all active tokens via the Redis blacklist.

LW-003 IDOR on Appointment Booking Endpoint High

CVSS Score	8.1 (AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N)	**Affected Component**	appointment-service — GET/PATCH/DELETE /v1

Description

The appointment retrieval and modification endpoints used the appointment ID from the URL path parameter without validating that the authenticated user owned the requested resource. Any authenticated customer could retrieve, modify, or cancel another customer's appointment by supplying a different UUID.

Business Impact

Exposure of customer PII (name, phone, email, address, appointment type) for all bookings. A malicious actor could cancel competitors' or specific customers' appointments, causing operational disruption. Mass enumeration of UUIDs is feasible using a wordlist.

Reproduction Steps

1. Authenticate as Customer A and create an appointment, noting the returned appointment ID.
2. Authenticate as Customer B (separate session).
3. Issue GET /v1/appointments/ using Customer B's JWT.
4. Observe full appointment details for Customer A returned with 200 status.

Recommendation

All appointment endpoint handlers must verify that appointment.customerId equals the authenticated user's ID extracted from the JWT (req.user.id). Implement this as a reusable ownership guard in the appointment repository layer. Admin routes that legitimately require access to all appointments must be separately gated by the ADMIN role check.

LW-004 Missing Rate Limiting on Authentication Endpoints High

CVSS Score	7.5 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N)	**Affected Component**	api-gateway / auth-service — POST /v1/auth/login

Description

No rate limiting was applied to the login or registration endpoints at the API gateway or service level. Automated credential stuffing or brute-force attacks could proceed without throttling or account lockout. During testing, 10,000 login requests were submitted in under 60 seconds without any blocking response.

Business Impact

An attacker with a credential list from a third-party breach could conduct automated credential stuffing at scale against the login endpoint. Given that Lomash Wood customers may reuse credentials from other services, account takeover risk is elevated.

Reproduction Steps

1. Using a script, submit 500 POST /v1/auth/login requests within 30 seconds from a single IP.
2. Observe that all requests receive 401 (invalid credentials) with no rate-limit headers.
3. No 429 Too Many Requests response is returned at any point.

Recommendation

Apply express-rate-limit at the API gateway for the auth endpoint group: maximum 10 requests per 15-minute window per IP for /v1/auth/login. Implement account lockout after 5 consecutive failed attempts with a 15-minute cooldown (store in Redis). Add CAPTCHA challenge after 3 consecutive failures. Return X-RateLimit-Remaining and Retry-After headers.

LW-005 Stripe Webhook Replay Attack High

CVSS Score	7.3 (AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N)	**Affected Component**	order-payment-service — POST /v1/webhooks/str

Description

The Stripe webhook handler verified the webhook signature (stripe-signature header) correctly, but did not store or check previously processed Stripe event IDs. An attacker who captured a valid, signed payment_intent.succeeded webhook (e.g. via network access to a shared staging environment) could replay it to trigger duplicate order fulfilment.

Business Impact

Duplicate fulfilment of paid orders, resulting in goods being dispatched without payment or services being activated multiple times. In a consultation booking context, this could result in multiple appointments being confirmed from a single payment.

Reproduction Steps

1. Capture a raw valid payment_intent.succeeded webhook payload with its stripe-signature header.
2. Re-submit the identical payload to POST /v1/webhooks/stripe.
3. Observe second 200 OK response and duplicate order status update in the database.

Recommendation

After signature verification, extract the Stripe event ID from the payload (event.id). Query the database for an existing record with that event ID in the webhook_events table. If a record exists, return 200 immediately with no further processing (idempotent handler). If not, insert the event ID before processing. Add a unique database constraint on the event_id column.

LW-006 Verbose Error Messages Exposing Stack Traces High

CVSS Score	6.8 (AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N)	**Affected Component**	api-gateway / all services — All endpoints (produc

Description

Unhandled exceptions in several service endpoints returned full Node.js stack traces, internal file paths, Prisma model names, and database field names in the HTTP response body. These were reproducible in the staging environment and were confirmed to also occur in production via a separate observation during the assessment.

Business Impact

Stack traces expose the server's directory structure, framework version, ORM internals, and sometimes partial query structures. This information significantly aids an attacker in crafting targeted exploits against the specific technology stack in use.

Reproduction Steps

1. Submit a malformed JSON body to POST /v1/appointments.
2. Observe response body containing full stack trace with file paths and Prisma internals.

Recommendation

The centralised error handler middleware must catch all unhandled errors and return a generic JSON response in production: { "error": "An unexpected error occurred", "code": "INTERNAL_SERVER_ERROR" }. Full error details must be written only to the structured log (Winston/Pino) with a correlation ID. The correlation ID should be included in the client response for support purposes. Set NODE_ENV=production in all deployment environments to trigger production error handling paths. This finding remains OPEN pending deployment confirmation.

7. Remediation Roadmap

The following table provides a prioritised remediation schedule based on severity and business impact.

Finding	Severity	Target	Owner	Status
LW-001 Client-Controlled Payment Amount	Critical	Immediate (7d)	Backend Team	Remediated
LW-002 JWT Algorithm Confusion	Critical	Immediate (7d)	Auth Team	Remediated
LW-003 IDOR on Appointments	High	30 days	Backend Team	Remediated
LW-004 Missing Rate Limiting	High	30 days	DevOps/API	Remediated
LW-005 Stripe Webhook Replay	High	30 days	Backend Team	Remediated
LW-006 Verbose Error Messages	High	30 days	Backend Team	Open
LW-007–LW-011 Medium Findings	Medium	90 days	Backend Team	Partially Remediated
LW-012–LW-013 Low Findings	Low	Next Sprint	Backend Team	Remediated
LW-014 Security Headers	Info	Next Sprint	DevOps	N/A

8. Retesting Notes

CyberSec Partners Ltd conducted a partial retest on **28 March 2025** to validate remediation of Critical and High findings. The following was confirmed:

LW-001	VERIFIED REMEDIATED	Payment amount now sourced exclusively from server-side order record. Client-supplied amount field rejected with 400.
LW-002	VERIFIED REMEDIATED	JWT verification now enforces RS256 only. HS256-signed tokens rejected with 401.
LW-003	VERIFIED REMEDIATED	Ownership check implemented. Cross-user appointment access returns 403.
LW-004	VERIFIED REMEDIATED	Rate limiting confirmed: 429 returned after 10 login attempts per 15 minutes per IP.
LW-005	VERIFIED REMEDIATED	Webhook idempotency confirmed. Duplicate event returns 200 with no reprocessing.
LW-006	OPEN — PENDING	Stack traces still returned in staging. Team reports fix deployed to production but not yet confirmed in retest window.
LW-011	OPEN — IN PROGRESS	Race condition on appointment slot booking partially addressed with DB unique constraint. Pessimistic locking implementation in progress.

A full retest covering all Medium and Low findings is scheduled for **30 April 2025**. Lomash Wood's development team should notify CyberSec Partners Ltd upon completion of LW-006 and LW-011 remediation to arrange targeted retesting.

9. Appendix A — Tools Used

Tool	Version	Purpose
Burp Suite Professional	2023.9	Primary proxy for manual testing, request interception, and active scanning
OWASP ZAP	2.14.0	Automated vulnerability scanning and API endpoint discovery
jwt_tool	2.2.7	JWT analysis, algorithm confusion testing, and token manipulation
Postman	10.x	API endpoint exploration and manual request crafting
sqlmap	1.7.10	Automated SQL injection detection (used in safe/detection mode only)
Nikto	2.1.6	Web server misconfiguration scanning
ffuf	2.0.0	API endpoint fuzzing and directory enumeration
Python 3.11	3.11	Custom exploit scripts and automation

10. Appendix B — CVSS Score Reference

Score Range	Severity
0.0	None
0.1 – 3.9	Low
4.0 – 6.9	Medium
7.0 – 8.9	High
9.0 – 10.0	Critical

All CVSS scores in this report use the CVSS v3.1 base score metric. Temporal and environmental scores were not calculated as they are outside the scope of this engagement.