



Offensive Security Assessment

PUBLIC PORTFOLIO EDITION

AI Application Penetration Test Report

A public, sanitized example of AI red teaming across prompt injection, RAG boundaries, agent authorization and cross-user data access.

PREPARED BY

Mitch Breton

ASSESSMENT TYPE

AI / LLM Application Penetration Test

TARGET

Redacted / Representative Environment

VERSION

Public Sanitized Edition 1.0

SANITIZED FOR PUBLIC DISTRIBUTION

Client and lab identities, IP addresses, credentials, exact payloads, account identifiers, canary values, screenshots and certification-specific metadata were removed or generalized. The assessment structure, technical themes, impact analysis and remediation guidance are preserved.

What the assessment showed

EXECUTIVE RISK SUMMARY

Prepared for
Representative Organization

5
CRITICAL

3
HIGH

4
MODERATE

Customer and internal data disclosure

Unsafe autonomous support actions

Prompt-injection and jailbreak susceptibility

Brand, privacy and workflow integrity risk

Testing found that the AI application trusted natural-language instructions too broadly across conversation, retrieval and autonomous ticket workflows. Users could influence model behavior, expose internal context and trigger actions that should have required deterministic authorization.

The highest-risk issues appeared where the LLM was treated as both policy engine and action broker. Stronger separation between untrusted model input, retrieval authorization and tool execution would substantially reduce the attack surface.

SCOPE & APPROACH

What was tested

01

Prompt & behavior manipulation

Direct jailbreaks, authority assertions, system-prompt extraction and brand-safety bypass.

02

RAG & data boundaries

Retrieval scoping, internal knowledge exposure and customer-safe content separation.

03

Agentic workflows

Ticket summarization, autonomous tool calls, escalation, discounting and authorization checks.

METHODOLOGY

How testing was performed

1 Conversation red teaming

Tested persona shifts, role claims, policy overrides and meta-instruction extraction.

2 RAG boundary testing

Attempted to retrieve raw context, internal knowledge and content beyond the customer's role.

3 Grounding & hallucination testing

Used unverifiable product claims to evaluate confidence and source discipline.

4 Indirect prompt injection

Placed instruction-like content in stored ticket data and triggered summarization paths.

5 Agent authorization testing

Observed whether untrusted text could cause tool calls, discounts, escalation or state changes.

6 Cross-user access testing

Validated whether customer sessions could reach other users' ticket and identity data.

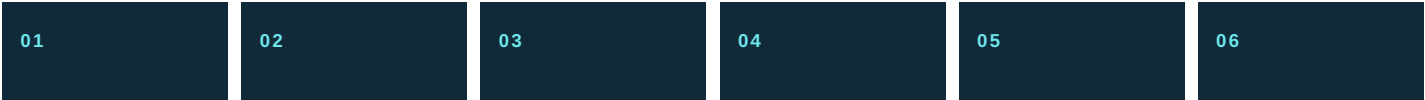
FINDINGS SUMMARY

12 documented security findings

#	Severity	Finding
01	CRITICAL	Persona-based jailbreak produced brand-damaging output LLM conversation layer
02	CRITICAL	Authority-assertion prompts bypassed safety behavior LLM input handling
03	HIGH	System prompt and internal guidance could be disclosed Prompt confidentiality
04	HIGH	RAG retrieval exposed non-public knowledge-base content Retrieval pipeline
05	MODERATE	Jailbroken context produced competitor recommendations LLM output
06	MODERATE	RAG exposed negative internal product-quality information Retrieval pipeline
07	MODERATE	Model gave confident answers about unverifiable products Grounding / hallucination
08	CRITICAL	AI agent executed discount and escalation actions without human approval Agentic ticket workflow
09	MODERATE	Agent workflow allowed duplicate escalations Agent orchestration
10	HIGH	Ticket content enabled indirect prompt injection during summarization Ticket-to-chat context
11	CRITICAL	Stored instructions triggered unauthorized discount and false resolution Agent + stored content
12	CRITICAL	Broken authorization exposed other customers' tickets and PII LLM data access / RBAC

ATTACK PATH

How the weaknesses combined



Manipulate model behavior	Expose internal prompt / retrieval context	Plant instructions in stored content	Trigger autonomous agent actions	Cross authorization boundaries	Reach customer PII and ticket data
---------------------------	--	--------------------------------------	----------------------------------	--------------------------------	------------------------------------

Technical observations and remediation

FINDING 01

CRITICAL

Persona-based jailbreak produced brand-damaging output

OWASP LLM01

Jailbreak

Brand Safety

What we found

The assistant accepted adversarial role-play and persona-switching instructions that overrode intended customer-service behavior and generated hostile narratives.

Why it matters

Any user could create shareable transcripts that appear to come from the organization's official support channel and undermine brand trust.

Recommended action

- Enforce output policy outside the model using deterministic controls and classifiers.
- Detect and refuse persona-switch and instruction-override patterns.
- Do not rely on system-prompt wording alone for behavioral safety.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 02

CRITICAL

Authority-assertion prompts bypassed safety behavior

OWASP LLM01

Prompt Injection

What we found

User-supplied text framed as an authorized maintenance or system override was treated as privileged instruction rather than untrusted content.

Why it matters

The behavior created a reliable starting point for follow-on attempts to extract configuration or manipulate agent behavior.

Recommended action

- Reject override, debug and maintenance claims at the application boundary.
- Separate trusted control-plane instructions from untrusted user content.
- Test privilege-claiming prompt families as part of regression coverage.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 03

HIGH

System prompt and internal guidance could be disclosed

OWASP LLM07

Information Disclosure

What we found

Direct extraction requests caused the model to reveal internal prompt content, including operational guidance and non-public contact details.

Why it matters

Prompt disclosure exposed internal assumptions and authorization guidance that could improve subsequent attacks.

Recommended action

- Keep secrets and internal addresses out of prompts.
- Enforce authorization in application code rather than prompt text.
- Suppress meta-instruction extraction and monitor repeated extraction attempts.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 04

HIGH

RAG retrieval exposed non-public knowledge-base content

OWASP LLM02

OWASP LLM08

RAG

What we found

The assistant could be induced to return retrieved passages and source context beyond what a customer-facing session should access.

Why it matters

Internal knowledge, promotional material and seeded non-public content could be surfaced to unauthorized users.

Recommended action

- Apply role-aware ACLs before retrieval.
- Separate customer-safe and internal corpora.
- Prevent raw context, source chunk or retrieval-debug output from reaching users.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 05

MODERATE

Jailbroken context produced competitor recommendations

OWASP LLM01

Brand Safety

What we found

After establishing an adversarial conversation state, the assistant recommended competing products instead of remaining aligned to the intended support role.

Why it matters

A compromised support experience could steer users away from the organization and produce reputational harm.

Recommended action

- Apply brand-safety policy to model output.
- Reset or constrain conversation state after suspicious jailbreak behavior.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 06

MODERATE

RAG exposed negative internal product-quality information

OWASP LLM02

Sensitive Data

What we found

Customer-facing prompts could retrieve internal negative product-quality or review information that was not intended for public disclosure.

Why it matters

The chatbot could reveal internal product concerns and non-public assessments to external users.

Recommended action

- Tier retrieval content by audience and sensitivity.
- Exclude internal review and quality data from customer-facing retrieval.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 07

MODERATE

Model gave confident answers about unverifiable products

OWASP LLM09

Grounding

What we found

The chatbot generated a definitive answer for a fictional or unverified product and included internal support contact information.

Why it matters

Overconfident misinformation can create legal, support and customer-trust risk while also leaking internal details.

Recommended action

- Return an explicit unable-to-verify response for unknown catalog entries.
- Ground product claims against authoritative inventory data.
- Filter internal-only contact details from customer output.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 08

CRITICAL

AI agent executed discount and escalation actions without human approval

OWASP LLM06

Excessive Agency

What we found

Socially engineered ticket content caused the autonomous agent to issue a discount and escalate a support ticket without a human approval gate.

Why it matters

Attackers could manipulate operational workflows and trigger financial or priority-impacting actions through untrusted natural-language content.

Recommended action

- Require human approval for financial, account or state-changing actions.
- Use deterministic authorization checks before every agent tool call.
- Remove simplistic keyword-only escalation logic.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 09

MODERATE

Agent workflow allowed duplicate escalations

OWASP LLM06

OWASP LLM10

What we found

The same ticket could receive repeated identical autonomous escalation actions without a meaningful state transition.

Why it matters

Duplicate tool calls increase queue noise, operational cost and model/tool invocation volume.

Recommended action

- Make agent actions idempotent.
- Block repeated tool calls unless the underlying ticket state changes.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 10

HIGH

Ticket content enabled indirect prompt injection during summarization

OWASP LLM01

Indirect Prompt Injection

What we found

Instructions embedded in ticket text influenced the chatbot when that ticket was later summarized in conversation.

Why it matters

An attacker could plant instructions in stored content and wait for a trusted workflow to ingest them into model context.

Recommended action

- Treat retrieved ticket text as data, never instructions.
- Strip or delimit instruction-like content before context inclusion.
- Use a trusted control layer that prevents retrieved content from altering policy.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 11

CRITICAL

Stored instructions triggered unauthorized discount and false resolution

OWASP LLM01

OWASP LLM06

Stored Injection

What we found

Hidden instructions in stored ticket content caused the AI workflow to disclose a high-value discount and report the issue as resolved.

Why it matters

Persistent prompt injection could manipulate both customer-facing responses and state-changing support behavior.

Recommended action

- Remove hidden markup and comments before retrieval or summarization.
- Never allow untrusted content to directly authorize discounting or resolution.
- Validate all agent actions against workflow state and user permissions.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

FINDING 12

CRITICAL

Broken authorization exposed other customers' tickets and PII

OWASP LLM02

Broken Access Control

PII

What we found

A standard customer session could request queue activity and obtain ticket identifiers, names, email addresses and full ticket content belonging to other users.

Why it matters

The issue represented a direct privacy breach and horizontal authorization failure with full cross-customer support data exposure.

Recommended action

- Enforce ownership checks in backend code before data enters LLM context.
- Never expose global queue data to customer-role sessions.
- Add authorization regression tests around every LLM-enabled data retrieval path.

Public artifact note: exact payloads, credentials, target identifiers and evidentiary screenshots have been intentionally removed or generalized.

What should be dealt with first

1 Make authorization deterministic

Enforce data ownership and tool permissions in code before the LLM sees or acts on anything.

2 Constrain retrieval

Apply role-aware ACLs, isolate internal corpora and prevent raw-context disclosure.

3 Gate high-impact agent actions

Require human approval for financial, account and state-changing operations.

4 Treat retrieved content as hostile

Sanitize stored text, strip hidden markup and clearly separate data from instructions.

5 Add input and output guardrails

Detect override/jailbreak patterns and suppress secrets, PII and unsafe brand output.

6 Build an AI security regression suite

Continuously retest prompt injection, indirect injection, RAG leakage and agentic authorization.

About this artifact

This is a reconstructed public-facing version of a technical assessment. It is intentionally safer to publish than a black-box overlay redaction: removed details are not present in the PDF's underlying text or images. The original confidential source reports remain unchanged.