www.getastra.com

Security Assessment Report

Automated Full Scan

Inserve VOF

Inserve

Report dated **July 15, 2024**

About Astra

Astra is a leading Penetration Testing as a Service (PTaaS) platform which combines continuous automated scanning with on-demand manual pentests by security experts. Astra follows the highest standards for security testing, vulnerability scanning and is an active contributor to industry leading Open source security standards and tools (OWASP WSTG, OWASP ZAP).

The assessment was performed within the predefined scope of this engagement, and its findings and recommendations have been shared with the customer. A penetration test is considered a snapshot in time. The findings and recommendations solely reflect the information gathered during the assessment period and do not account for any subsequent changes or modifications.

Astra IT Inc.
help@getastra.com

2093 Philadelphia Pike 4080,
Claymont, Delaware, 19703,
United States

Table of Contents

Overview

Executive Summary

Astra was engaged by Inserve VOF to perform a security assessment of **1** target during the period **12th July 2024** to **12th July 2024**. Manual pentest was performed on **0** targets , and automated vulnerability scanning was performed on **1** target.

The testing was performed from a remote attacker's perspective with the following goals:

- To perform automated vulnerability scanning and identify security loopholes, known vulnerabilities, and evaluate effectiveness of existing security controls in the application.
- Recommend technical security best practices to improve security posture of the target applications audited.
- Explain the potential impact of the identified vulnerabilities, such as the extent of data exposure, potential financial losses, or reputational damage that could occur if they were exploited by malicious actors.
- Provide clear and actionable recommendations for addressing the identified vulnerabilities.

A total of **0 vulnerabilities/recommendations** were reported. Out of a score of 10, the highest risk score assigned to a vulnerability was **0**, the lowest was **0**, and the average score was **0**.

The reported vulnerabilities have been found during an automated vulnerability scan, and have not been vetted by Astra's security analysts.

Scope of the Assessment

The assessment was performed within the predefined scope of this engagement as listed below. No assumptions about the application were made.

Type	Name	Scope	Start Grade	Closure Grade
Web App	Inserve	https://test.inservebeta.nl	A+	A+

Resolution Statistics

Severity	Solved	Unsolved	Help Wanted	Under Review	Accepted Risk	Grand Total
Critical	0	0	0	0	0	0
High	0	0	0	0	0	0
Medium	0	0	0	0	0	0
Low	0	0	0	0	0	0
Info	0	0	0	0	0	0
Grand Total	0	0	0	0	0	0

Overall vulnerability statistics

Scan Details

Assessment Methodology

An automated scan was performed using Astra's cloud-based vulnerability scanner on the targets.

Using the same techniques as sophisticated real-world attackers, the scanner scans for **9300+ vulnerabilities** based on recently found CVEs, and industry standards such as OWASP Web Security Testing Guide (WSTG), OWASP Top 10, OWASP Application Security Verification Standard (ASVS), NIST 800-115 etc.

Scan Authentication

Scan was performed using suitable authenticated test accounts. For this assessment you can see the number of user roles tested in the table below.

Testers have full access to information about the platform being tested. This often includes accounts (including administrative users), and access to discuss functionality with developers during the testing process.

Assessment Duration and Dates

Scan Mode	Target Name	Authentication	Started	Completed
Automated (Full)	Inserve	1 user	12th Jul 2024	12th Jul 2024

Certificates

No certificates have been issued for the scope covered as per this report. Certificates are issued when 90% or more vulnerabilities have been fixed after a manual pentest. The remaining vulnerabilities have to be of Info or Low severity. They can also be of Medium severity, if they're not immediately fixable.

Rule Exclusion

1. Weak Password Policy Implementation - The test doesnt register of create a new password for the user therefore it cant check the password policy Which is currently implemented as by the guidelines of a strong password
2. NoSQL Injection - MongoDB - The ne value is sent is the request but its ignored by our parser
3. Unexpected Content-Type Not Being Rejected - An invalid ContentType will just be ignored by our application
4. Spring Actuator Endpoints Publicly Available - A 200 response is returned but that happens on every invalid page
5. Permissions Policy Header Not Set - Not applicable
6. Cookie without SameSite Attribute - Accepted
7. Advanced SQL Injection - AND boolean-based blind - WHERE or HAVING clause - Theres no SQL injection the response is a 403 and the passed sleep method is not executed
8. [Recommendation] Implement Idempotency Header - Wont fix for now
9. Incomplete or No Cache-control Header Set - Accepted
10. Session Token Not Invalidated on Logout - Test on inservebeta copied me request as cUrl request with Bearer Token Logged out in browser and executed request again response was Unauthenticated
11. Use of Symmetric Algorithm in JSON Web Tokens (JWTs) - Wont fix for now
12. CORS Misconfiguration - Since the Inserve API is a public api with an APIkey theres no need to configure allowed domains for CORS
13. SQL Injection - No SQL injection takes place the request parameter is ignored
14. GET for POST - GET and POST methods are both allowed on this route
15. Bypassing 403 Forbidden Response Status Code - Our frontend application does nothing with this request and just displays an empty page with a 200 response However that doesnt imply a 403 is bypassed
16. name Hash Found - The hash that is found in the response is the md5 of the email address since this is used in the gravatarcom URL to obtain an avatar for the userFrom the JSON response gravatar jsonhttpswwwgravatarcomavatare3a1d71cd3c73d0f176d89b2db67fc29d404

Vulnerabilities

Overview Table

No vulnerabilities found.

Appendix

APPENDIX A — MEASUREMENT SCALES

Astra determines severity ratings using in-house expertise and industry-standard rating methodologies such as the Open Web Application Security Project (OWASP) and the Common Vulnerability Scoring System (CVSS).

The severity of each finding in this report was determined independently of the severity of other findings. Vulnerabilities assigned a higher severity have more significant technical and business impact and achieve that impact through fewer dependencies on other flaws.

Critical: Vulnerability is an otherwise high-severity issue with additional security implications that could lead to exceptional business impact. Findings are marked as critical severity to communicate an exigent need for immediate remediation. Examples include threats to human safety, permanent loss or compromise of business-critical data, and evidence of prior compromise.

High: Vulnerability introduces significant technical risk to the system that is not contingent on other issues being present to exploit. Examples include creating a breach in the confidentiality or integrity of sensitive business data, customer information, or administrative and user accounts.

Medium: Vulnerability does not in isolation lead directly to the exposure of sensitive business data. However, it can be leveraged in conjunction with another issue to expose business risk. Examples include insecurely storing user credentials, transmitting sensitive data unencrypted, and improper network segmentation.

Low: Vulnerability may result in limited risk or require the presence of multiple additional vulnerabilities to become exploitable. Examples include overly verbose error messages, insecure TLS configurations, and detailed banner information disclosure.

Informational: Finding does not have a direct security impact but represents an opportunity to add an additional layer of security, is a deviation from best practices, or is a security-relevant observation that may lead to exploitable vulnerabilities in the future. Examples include vulnerable yet unused source code and missing HTTP security headers.

APPENDIX B - RESOLUTION STATUS

Unsolved: This status indicates that the security team has reported an issue or vulnerability to the customer, but it is yet to be resolved by the customer. Further actions are required to address the reported security concern.

Under Review: This status is assigned when the customer has fixed the reported issue or vulnerability. The security team will now evaluate and validate the fix to ensure it has been implemented correctly and effectively mitigates the identified risk.

Accepted Risk: This status reflects the customer's decision not to address or resolve the reported issue or vulnerability. By accepting the associated risk, the customer has chosen not to pursue any further action in mitigating the identified security concern.

Help Wanted: When assigned this status, it indicates that the customer has requested additional clarification or assistance from the security team. This could involve seeking further guidance, recommendations, or expertise to better understand and address the reported security issue.

Solved: This status signifies that the customer has successfully resolved the reported vulnerability, and it has been verified by the security team. The necessary measures have been taken to mitigate the risk, ensuring that the identified security concern is no longer present.

APPENDIX C — RISK SCORE

Security recommendations/long-term tasks are marked as “Unsolved” or "Accepted Risk". Astra has evaluated the risk based on information provided and has provided feedback on a case-to-case basis as requested.

Security grades are assigned for vulnerabilities needing immediate attention and does not include security best practices, although reported. For each vulnerability, a risk score is assigned which signifies real world possibility of an attack being orchestrated using the vulnerability. The risk score is calculated using a correlation of multiple factors including potential loss value of a vulnerability, CVSS score, historic data about attacks performed using similar vulnerability & severity of such vulnerabilities assigned by our security engineers in the past.

APPENDIX D — TEST CASES

The following lists of tests are suggestive & not limited to the ones listed. Most importantly, every test case has multiple sub-test cases ranging from a few to sometimes 1000+ sub tests. Additional test cases will be performed based on factors such as:

1. Business Logic
2. Technology Stack
3. Framework/CMS/APIs
4. Application specific features

OWASP Top 10

#	For Applications
1	Broken Access Control
2	Cryptographic Failures
3	Injection
4	Insecure Design
5	Security Misconfiguration
6	Vulnerable and Outdated Components
7	Identification and Authentication Failures
8	Software and Data Integrity Failures
9	Security Logging and Monitoring Failuresies
10	Server-Side Request Forgery

SANS 25 Software Errors/Tests

#	SANS 25
1	Improper Restriction of Operations within the Bounds of a Memory Buffer
2	Improper Neutralization of Input During Web Page Generation ('XSS')
3	Improper Input Validation
4	Information Exposure
5	Out-of-bounds Read
6	Improper Neutralization of Special Elements used in an SQL Command (SQLi)
7	Use After Free
8	Integer Overflow or Wraparound
9	Cross-Site Request Forgery (CSRF)ies
10	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
11	Improper Neutralization of Special Elements used in an OS Command
12	Out-of-bounds Write
13	Improper Authentication
14	NULL Pointer Dereference
15	Incorrect Permission Assignment for Critical Resource
16	Unrestricted Upload of File with Dangerous Type
17	Improper Restriction of XML External Entity Reference
18	Improper Control of Generation of Code ('Code Injection')
19	Use of Hard-coded Credentials
20	Uncontrolled Resource Consumption
21	Missing Release of Resource after Effective Lifetime
22	Untrusted Search Path
23	Deserialization of Untrusted Data
24	Improper Privilege Management
25	Improper Certificate Validation

174 Other Test Cases

#	Test Performed	Typical Severity
1	OS Command Injection	High
2	SQL Injection (Second Order)	High
3	XML External Entity Injection	High
4	LDAP Injection	High
5	XPath Injection	High
6	XML Injection	High
7	ASP.NET Debugging Enabled	High
8	DoS Locking Customer Accounts	Medium
9	DoS Buffer Overflows	Medium
10	Storing too much data in session (DoS)	High
11	Writing user-provided data to disk (DoS)	High
12	HTTP Insecure methods available on Server	High
13	Out of band resource load (HTTP)	High
14	File path manipulation	High
15	Server-site JavaScript code injection	High
16	Perl code injection	High
17	Ruby code injection	High
18	Python code injection	High
19	Expression Language injection	High
20	Unidentified code injection	High
21	Server-side template injection	High
22	SSL injection	High
23	Stored XSS	High
24	HTTP response header injection	High
25	Reflected XSS	High
26	Client-side template injection	High
27	DOM-based XSS	High
28	Reflected DOM-based XSS	High
29	Stored DOM-based XSS	High
30	DOM-based JavaScript Injection	High
31	Reflected DOM-based JavaScript Injection	High
32	Stored DOM-based JavaScript Injection	High
33	Path-relative style sheet import	Information
34	Client-side SQLi (DOM-based)	High
35	Client-side SQLi (Reflected DOM-based)	High
36	Client-side SQLi (Stored DOM-based)	High
37	WebSocket Hijacking (DOM-based)	High
38	WebSocket Hijacking (Reflected DOM-based)	High
39	WebSocket Hijacking (Stored DOM-based)	High
40	Local Path Manipulation (DOM-based)	High
41	Local Path Manipulation (Reflected DOM)	High
42	Local Path Manipulation (Stored DOM-based)	High
43	Client-side XPATH Injection (DOM-based)	Low
44	Client-side XPATH Injection (Reflected DOM)	Low
45	Client-side XPATH Injection (Stored DOM)	Low

#	Test Performed	Typical Severity
46	Client-side JSON Injection (DOM-based)	Low
47	Client-side JSON Injection (Reflected DOM)	Low
48	Client-side JSON Injection (Stored DOM-based)	Low
49	Flash cross-domain policy	High
50	Cross-origin resource sharing	
51	Cross-origin resource sharing (arbitrary)	High
52	Cross-origin resource sharing (encrypted)	Low
53	Cross-origin resource sharing (all sub-domains)	Low
54	Cross-site Request Forgery (CSRF)	Medium
55	SMTP header injection	Medium
56	Cleartext submission of password	High
57	External service interaction (DNS)	High
58	External service interaction (HTTP)	High
59	External service interaction (SMTP)	Information
60	Referrer dependent response	Information
61	Spoofable client IP address	Information
62	User-agent dependent response	Information
63	Password returned in a later response	Medium
64	Password submitted using GET method	Low
65	Password returned in URL query string	Low
66	SQL statement in request parameter	Medium
67	Cross-domain POST	Information
68	ASP.NET ViewState without MAC Enabled	
69	XML entity expansion	Medium
70	Long redirection response	Information
71	Serialized object in HTTP message	
72	Duplicate cookies set	Information
73	WebSocket Hijacking (DOM-based)	High
74	WebSocket Hijacking (Reflected DOM-based)	High
75	WebSocket Hijacking (Stored DOM-based)	High
76	Local Path Manipulation (DOM-based)	High
77	Local Path Manipulation (Reflected DOM)	High
78	Local Path Manipulation (Stored DOM-based)	High
79	Client-side XPATH Injection (DOM-based)	Low
80	Client-side XPATH Injection (Reflected DOM)	Low
81	Client-side XPATH Injection (Stored DOM)	Low
82	Client-side JSON Injection (DOM-based)	Low
83	Client-side JSON Injection (Reflected DOM)	Low
84	Client-side JSON Injection (Stored DOM-based)	Low
85	Flash cross-domain policy	High
86	Cross-origin resource sharing	
87	Cross-origin resource sharing (arbitrary)	High
88	Cross-origin resource sharing (encrypted)	Low
89	Cross-origin resource sharing (all sub-domains)	Low
90	Cross-site Request Forgery (CSRF)	Medium
91	SMTP header injection	Medium
92	Cleartext submission of password	High
93	External service interaction (DNS)	High

#	Test Performed	Typical Severity
94	External service interaction (HTTP)	High
95	External service interaction (SMTP)	Information
96	Referrer dependent response	Information
97	Spoofable client IP address	Information
98	User-agent dependent response	Information
99	Password returned in a later response	Medium
100	Password submitted using GET method	Low
101	Password returned in URL query string	Low
102	SQL statement in request parameter	Medium
103	Cross-domain POST	Information
104	ASP.NET ViewState without MAC Enabled	
105	XML entity expansion	Medium
106	Long redirection response	Information
107	Serialized object in HTTP message	
108	Duplicate cookies set	Information
109	Input returned in response (stored)	Information
110	Input returned in response (reflected)	Information
111	Suspicious input transformation (reflected)	Information
112	Suspicious input transformation (stored)	Information
113	Open redirection (stored)	Low
114	Open redirection (reflected)	Medium
115	Open redirection (DOM-based)	Low
116	Open redirection (Stored DOM-based)	Low
117	Open redirection (Reflected DOM-based)	Medium
118	SSL cookie without secure flag set	Medium
119	Cookie scoped to parent domain	Low
120	Cross-domain referrer leakage	Information
121	Cross-domain script include	Information
122	Cookie without HTTPOnly flag set	
123	Session token in URL	
124	Password field with autocomplete enabled	
125	Password value set in cookie	Medium
126	Browser cross-site scripting disabled	Information
127	HTTP TRACE method is enabled	Information
128	Cookie manipulation (DOM-based)	Low
129	Cookie manipulation (reflected DOM-based)	Low
130	Cookie manipulation (DOM-based)	Low
131	Ajax request header manipulation (DOM-based)	Low
132	Ajax request header manipulation (reflected)	Low
133	Ajax request header manipulation (stored DOM)	Low
134	Denial of service (DOM-based)	Information
135	Denial of service (reflected DOM-based)	Information
136	Denial of service (stored DOM-based)	Low
137	HTML5 web message manipulation DOM-based	Information
138	HTML5 web message manipulation (reflected)	Information
139	HTML5 web message manipulation (stored DOM)	Information
140	HTML5 storage manipulation (DOM-based)	Information
141	HTML5 storage manipulation (reflected DOM)	Information

#	Test Performed	Typical Severity
142	HTML5 storage manipulation (stored DOM)	Information
143	Link manipulation (DOM-based)	Low
144	Link manipulation (reflected DOM-based)	Low
145	Link manipulation (stored DOM-based)	Low
146	Link manipulation (reflected & stored)	Information
147	Document domain manipulation (DOM-based)	Medium
148	Document domain manipulation reflected DOM	Medium
149	Document domain manipulation (stored DOM)	Medium
150	DOM data manipulation (DOM-based)	Information
151	CSS Injection (reflected & stored)	Medium
152	Client-side HTTP parameter pollution (reflected)	Low
153	Client-side HTTP parameter pollution (Stored)	Low
154	Form action hijacking (reflected)	Medium
155	Form action hijacking (stored)	Medium
156	Database connection string disclosed	Medium
157	Source code disclosure	
158	Directory listing	Information
159	Email addresses disclosed	Information
160	Private IP addresses disclosed	Information
161	Social security numbers disclosed	Information
162	Credit card numbers disclosed	Information
163	Private key disclosed	Information
164	Cacheable HTTPS response	Information
165	Base64 encoded data in parameter	Information
166	Multiple content types specified	Information
167	HTML does not specify charset	Information
168	HTML uses unrecognized charset	Information
169	Content type incorrectly stated	Low
170	Content type is not specified	Information
171	SSL certificate	Medium
172	Unencrypted communications	Low
173	Strict transport security not enforced	Low
174	Mixed content	Low