Atlassian

Summary Report

Report created on April 09, 2024
Report date range: January 01, 2024 - March 31, 2024
Prepared by: Ben Howe, bhowe@atlassian.com
# Table of contents

- Executive summary ................................................................. 3
- Reporting and methodology ..................................................... 4
  - Background .................................................................. 4
- Targets and scope .................................................................. 5
  - Scope ........................................................................ 5
- Findings Summary .................................................................. 7
- Risk and priority key ............................................................... 8
- Appendix ............................................................................ 9
  - Submissions over time ...................................................... 9
  - Submissions signal ........................................................... 9
  - Bug types overview ......................................................... 10
- Closing Statement ................................................................. 11
Executive summary

Atlassian engaged Bugcrowd, Inc. to perform a Bug Bounty Program, commonly known as a crowd-sourced penetration test.

A Bug Bounty Program is a cutting-edge approach to an application assessment or penetration test. Traditional penetration tests use only one or two personnel to test an entire scope of work, while a Bug Bounty leverages a crowd of security researchers. This increases the probability of discovering esoteric issues that automated testing cannot find and that traditional vulnerability assessments may miss in the same testing period.

The purpose of this program was to identify security vulnerabilities in the targets listed in the targets and scope section. Once identified, each vulnerability was rated for technical impact defined in the findings summary section of the report.

This report shows testing for Atlassian's targets during the period of: 01/01/2024 – 03/31/2024.

For this Bug Bounty Program, submissions were received from 244 unique researchers.

The continuation of this document summarizes the findings, analysis, and recommendations from the Bug Bounty Program performed by Bugcrowd for Atlassian.

This report is a summary of the information available. All details of the engagement's findings — comments, code, and any tester provided remediation information — can be found in the Bugcrowd platform (https://tracker.bugcrowd.com)
Reporting and methodology

Background

The strength of crowdsourced testing lies in multiple researchers, the pay-for-results model, and the varied methodologies that the researchers implement. To this end, researchers are encouraged to use their own individual methodologies on Bug Bounty Engagements.

The workflow of every penetration test can be divided into the following four phases:

01 Reconnaissance
Gathering information before the attack

02 Enumeration
Finding attack vectors

03 Exploitation
Verifying security weaknesses

04 Documentation
Collecting results

Bugcrowd researchers who perform web application testing and vulnerability assessment usually subscribe to a variety of methodologies following the highlighted workflow, including the following:
Targets and scope

Scope

Prior to the Ongoing program launching, Bugcrowd worked with Atlassian to define the Rules of Engagement, commonly known as the program brief, which includes the scope of work. The following targets were considered explicitly in scope for testing:

- Atlassian Access (https://admin.atlassian.com/atlassian-access)
- Atlassian Admin (https://admin.atlassian.com/)
- Atlassian Identity (https://id.atlassian.com/login)
- Atlassian Start (https://start.atlassian.com)
- Bitbucket Cloud including Bitbucket Pipelines (https://bitbucket.org)
- Confluence Cloud (bugbounty-test-<bugcrowd-name>.atlassian.net/wiki)
- Confluence Cloud Premium (bugbounty-test-<bugcrowd-name>.atlassian.net/wiki)
- Confluence Cloud Mobile App for Android
- Confluence Cloud Mobile App for iOS
- Jira Cloud Mobile App for Android
- Jira Cloud Mobile App for iOS
- Jira Service Management Cloud (bugbounty-test-<bugcrowd-name>.atlassian.net)
- Jira Software Cloud (bugbounty-test-<bugcrowd-name>.atlassian.net)
- Jira Work Management Cloud formerly Jira Core (bugbounty-test-<bugcrowd-name>.atlassian.net)
- Any associated *.atlassian.com or *.atlas-paas.net domain that can be exploited DIRECTLY from the *.atlassian.net instance
- Atlassian Compass
- Atlassian Atlas
- Bitbucket Data Center
- Confluence Data Center
- Crowd
- Jira Core Data Center
- Jira Service Management Data Center
- Jira Software Data Center
- *.atlastunnel.com
- Any other *.atlassian.com or *.atl-paas.net domain that cannot be exploited directly from a *.atlassian.net instance
- Bamboo
- Confluence Companion App for macOS and Windows
- Confluence Data Center Mobile App for Android
- Confluence Data Center Mobile App for iOS
- Crucible
- FishEye
- Jira Data Center Mobile App for Android
- Jira Data Center Mobile App for iOS
- Sourcetree for macOS and Windows (https://www.sourcetreeapp.com/)
- Other - (all other Atlassian targets)
- Jira Product Discovery
- Beacon
- Forge Platform
- GraphQL API (bugbounty-test-<bugcrowd-name>.atlassian.net/gateway/api/graphql)
- https://www.npmjs.com/package/@forge/cli

All details of the program scope and full program can be reviewed in the program settings.
Findings Summary

The following chart shows all valid assessment findings from the program by technical severity.
Risk and priority key

The following key is used to explain how Bugcrowd rates valid vulnerability submissions and their technical severity. As a trusted advisor Bugcrowd also provides common "next steps" for program owners per severity category.

<table>
<thead>
<tr>
<th>Technical severity</th>
<th>Example vulnerability types</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Critical P1</strong></td>
<td>• Remote Code Execution</td>
</tr>
<tr>
<td></td>
<td>• Vertical Authentication Bypass</td>
</tr>
<tr>
<td></td>
<td>• XML External Entities Injection</td>
</tr>
<tr>
<td></td>
<td>• SQL Injection</td>
</tr>
<tr>
<td></td>
<td>• Insecure Direct Object Reference for a critical function</td>
</tr>
</tbody>
</table>

Critical severity submissions (also known as "P1" or "Priority 1") are submissions that are escalated to Bugcrowd as soon as they are validated. These issues warrant the highest security consideration and should be addressed immediately. Commonly, submissions marked as Critical can cause financial theft, unavailability of services, large-scale account compromise, etc.

| Severe P2          | • Lateral authentication bypass |
|--------------------| • Stored Cross-Site Scripting    |
|                    | • Cross-Site Request Forgery for a critical function |
|                    | • Insecure Direct Object Reference for an important function |
|                    | • Internal Server-Side Request Forgery |

High severity submissions (also known as "P2" or "Priority 2") are vulnerability submissions that should be slated for fix in the very near future. These issues still warrant prudent consideration but are often not availability or "breach level" submissions. Commonly, submissions marked as High can cause account compromise (with user interaction), sensitive information leakage, etc.

| Moderate P3        | • Reflected Cross-Site Scripting with limited impact |
|--------------------| • Cross-Site Request Forgery for an important function |
|                    | • Insecure Direct Object Reference for an unimportant function |
|                    | • Internal Server-Side Request Forgery |

Medium severity submissions (also known as "P3" or "Priority 3") are vulnerability submissions that should be slated for fix in the major release cycle. These vulnerabilities can commonly impact single users but require user interaction to trigger or only disclose moderately sensitive information.

| Low P4            | • Cross-Site Scripting with limited impact |
|                  | • Cross-Site Request Forgery for an unimportant function |
|                  | • External Server-Side Request Forgery |
|                  | • Lack of code obfuscation |
|                  | • Autocomplete enabled |
|                  | • Non-exploitable SSL issues |

| Informational P5 | • Lack of code obfuscation |
|------------------| • Autocomplete enabled |
|                  | • Non-exploitable SSL issues |

More detailed information regarding our vulnerability classification can be found at: https://bugcrowd.com/vulnerability-rating-taxonomy
Appendix

Included in this appendix are auxiliary metrics and insights into the Bug Bounty Program. This includes information regarding submissions over time, payouts and prevalent issue types.

Submissions over time

The timeline below shows submissions received and validated by the Bugcrowd team:

Submissions signal

A total of 410 submissions were received, with 106 unique valid issues discovered. Bugcrowd identified 50 informational submissions, 70 duplicate submissions, removed 232 invalid submissions, and is processing 2 submissions. The ratio of unique valid submissions to noise was 26%.

<table>
<thead>
<tr>
<th>Submission Outcome</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid</td>
<td>106</td>
</tr>
<tr>
<td>Informational</td>
<td>50</td>
</tr>
<tr>
<td>Invalid</td>
<td>232</td>
</tr>
<tr>
<td>Duplicate</td>
<td>70</td>
</tr>
<tr>
<td>Processing</td>
<td>2</td>
</tr>
<tr>
<td>Total</td>
<td>410</td>
</tr>
</tbody>
</table>

Ratio of Unique Valid Submissions to Noise

- 100%
- 75%
- 50%
- 25%
- 26%
- 0%
Bug types overview

This distribution across bug types for the Bug Bounty Program only includes unique and valid submissions.
Closing Statement

Introduction

This report shows testing of Atlassian between the dates of 01/01/2024 - 03/31/2024. During this time, 244 researchers from Bugcrowd submitted a total of 410 vulnerability submissions against Bugcrowd’s targets. The purpose of this assessment was to identify security issues that could adversely affect the integrity of Bugcrowd. Testing focused on the following:

1. Atlassian Access (https://admin.atlassian.com/atlassian-access)
2. Atlassian Admin (https://admin.atlassian.com/)
3. Atlassian Identity (https://id.atlassian.com/login)
4. Atlassian Start (https://start.atlassian.com)
5. Bitbucket Cloud including Bitbucket Pipelines (https://bitbucket.org)
6. Confluence Cloud (bugbounty-test-<bugcrowd-name>.atlassian.net/wiki)
7. Confluence Cloud Premium (bugbounty-test-<bugcrowd-name>.atlassian.net/wiki)
8. Confluence Cloud Mobile App for Android
9. Confluence Cloud Mobile App for iOS
10. Jira Cloud Mobile App for Android
11. Jira Cloud Mobile App for iOS
12. Jira Service Management Cloud (bugbounty-test-<bugcrowd-name>.atlassian.net)
13. Jira Software Cloud (bugbounty-test-<bugcrowd-name>.atlassian.net)
14. Jira Work Management Cloud formerly Jira Core (bugbounty-test-<bugcrowd-name>.atlassian.net)

15. Any associated *.atlassian.com or *.atl-paas.net domain that can be exploited DIRECTLY from the *.atlassian.net instance

16. Atlassian Compass


18. Atlassian Atlas

19. Bitbucket Data Center

20. Confluence Data Center

21. Crowd

22. Jira Core Data Center

23. Jira Service Management Data Center

24. Jira Software Data Center

25. *.atlastunnel.com

26. Any other *.atlassian.com or *.atl-paas.net domain that cannot be exploited directly from a *.atlassian.net instance

27. Bamboo

28. Confluence Companion App for macOS and Windows

29. Confluence Data Center Mobile App for Android

30. Confluence Data Center Mobile App for iOS

31. Crucible

32. FishEye

33. Jira Data Center Mobile App for Android

34. Jira Data Center Mobile App for iOS

35. Sourcetree for macOS and Windows (https://www.sourcetreeapp.com/)

36. Other - (all other Atlassian targets)

37. Jira Product Discovery

38. Beacon

39. Forge Platform

40. GraphQL API (bugbounty-test-<bugcrowd-name>.atlassian.net/gateway/api/graphql)

The assessment was performed under the guidelines provided in the statement of work between Atlassian and Bugcrowd. This letter provides a high-level overview of the testing performed, and the result of that testing.

Atlassian Program Overview

An Atlassian Program is a novel approach to a penetration test. Traditional penetration tests use only one or two researchers to test an entire scope of work, while an Ongoing program leverages a crowd of security researchers. This increases the probability of discovering esoteric issues that automated testing cannot find and that traditional vulnerability assessments may miss, in the same testing period.

It is important to note that this document represents a point-in-time evaluation of security posture. Security threats and attacker techniques evolve rapidly, and the results of this assessment are not intended to represent an endorsement of the adequacy of current security measures against future threats. This document contains information in summary form and is therefore intended for general guidance only; it is not intended as a substitute for detailed research or the exercise of professional judgment. The information presented here should not be construed as professional advice or service.

Testing Methods

This security assessment leveraged researchers that used a combination of proprietary, public, automated, and manual test techniques throughout the assessment. Commonly tested vulnerabilities include code injection, cross-site request forgery, cross-site scripting, insecure storage of sensitive data, authorization/authentication vulnerabilities, business logic vulnerabilities, and more.

Summary of Findings

During the Engagement, Bugcrowd discovered the following:

<table>
<thead>
<tr>
<th>Technical Severity</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Critical vulnerabilities</td>
<td>2</td>
</tr>
<tr>
<td>Severe vulnerabilities</td>
<td>7</td>
</tr>
<tr>
<td>Moderate vulnerabilities</td>
<td>32</td>
</tr>
<tr>
<td>Low vulnerabilities</td>
<td>19</td>
</tr>
<tr>
<td>Informational vulnerabilities</td>
<td>46</td>
</tr>
</tbody>
</table>