

# GitKraken

## Summary Report

Report created on January 17, 2024

Report date range: February 11, 2020 - January 17, 2024

Prepared by: Adam Wride, [adam.wride@gitkraken.com](mailto:adam.wride@gitkraken.com)



---

# Table of contents

<b>Executive summary</b> .....	<b>2</b>
<b>Reporting and methodology</b> .....	<b>3</b>
Background .....	3
<b>Targets and scope</b> .....	<b>4</b>
Scope .....	4
<b>Findings Summary</b> .....	<b>5</b>
<b>Risk and priority key</b> .....	<b>6</b>
<b>Appendix</b> .....	<b>7</b>
Submissions over time .....	7
Submissions signal .....	7
Bug types overview .....	8
Spend of program rewards pool .....	8
Top 3 highest paid submissions .....	9
<b>Closing Statement</b> .....	<b>10</b>



---

# Executive summary

**BigBrassBand** engaged Bugcrowd, Inc. to perform an Ongoing Bounty Program, commonly known as a crowd-sourced penetration test.

An Ongoing 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 an Ongoing 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 **GitKraken's** targets during the period of: **02/11/2020 – 01/17/2024**.

For this Ongoing Program, submissions were received from **128** unique researchers.

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

# Reporting and methodology

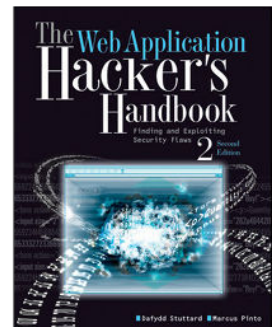
## Background

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 Ongoing Bug Bounty Engagements.

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



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 **GitKraken** 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:

- Git Integration for Jira - Cloud Hosted - <https://marketplace.atlassian.com/apps/4984/git-integration-for-jira?hosting=cloud&tab=overview>
- Git Integration for Jira - Server Hosted - <https://marketplace.atlassian.com/apps/4984/git-integration-for-jira?hosting=server&tab=overview>
- Git Integration for Jira - Data Center Hosted - <https://marketplace.atlassian.com/apps/4984/git-integration-for-jira?hosting=datacenter&tab=overview>
- <https://marketplace.atlassian.com/apps/1219270/dev-info-for-jira?hosting=cloud&tab=overview>
- CI/CD For Jira - <https://marketplace.atlassian.com/apps/1228578/ci-cd-for-jira?hosting=cloud&tab=overview>
- Team Insights for Jira (Preview) - <https://marketplace.atlassian.com/apps/1230390/team-insights-for-jira-preview?hosting=cloud&tab=overview>

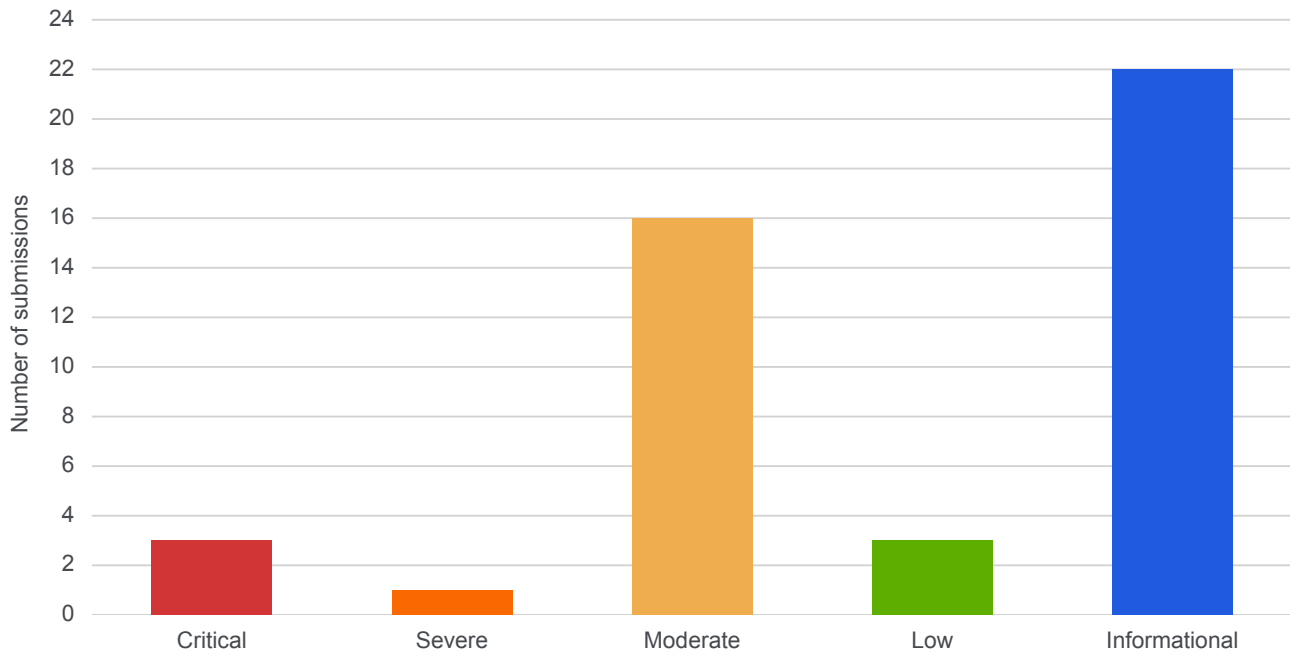
All details of the program scope and full program can be reviewed in the program settings:

<https://tracker.bugcrowd.com/bigbrassband/settings/details>



# 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.

## Technical severity

	Example vulnerability types
<div data-bbox="119 604 279 645"> <span>Critical</span> <span>P1</span> </div> <p data-bbox="119 660 981 772">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.</p>	<ul data-bbox="1013 616 1476 806" style="list-style-type: none"> <li>• Remote Code Execution</li> <li>• Vertical Authentication Bypass</li> <li>• XML External Entities Injection</li> <li>• SQL Injection</li> <li>• Insecure Direct Object Reference for a critical function</li> </ul>
<div data-bbox="119 828 279 869"> <span>Severe</span> <span>P2</span> </div> <p data-bbox="119 884 981 996">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.</p>	<ul data-bbox="1013 840 1476 1030" style="list-style-type: none"> <li>• Lateral authentication bypass</li> <li>• Stored Cross-Site Scripting</li> <li>• Cross-Site Request Forgery for a critical function</li> <li>• Insecure Direct Object Reference for an important function</li> <li>• Internal Server-Side Request Forgery</li> </ul>
<div data-bbox="119 1052 295 1093"> <span>Moderate</span> <span>P3</span> </div> <p data-bbox="119 1108 981 1198">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.</p>	<ul data-bbox="1013 1064 1476 1209" style="list-style-type: none"> <li>• Reflected Cross-Site Scripting with limited impact</li> <li>• Cross-Site Request Forgery for an important function</li> <li>• Insecure Direct Object Reference for an unimportant function</li> </ul>
<div data-bbox="119 1232 247 1272"> <span>Low</span> <span>P4</span> </div> <p data-bbox="119 1288 981 1377">Low severity submissions (also known as "P4" or "Priority 4") are vulnerability submissions that should be considered for fix within the next six months. These vulnerabilities represent the least danger to confidentiality, integrity, and availability.</p>	<ul data-bbox="1013 1243 1476 1366" style="list-style-type: none"> <li>• Cross-Site Scripting with limited impact</li> <li>• Cross-Site Request Forgery for an unimportant function</li> <li>• External Server-Side Request Forgery</li> </ul>
<div data-bbox="119 1411 335 1451"> <span>Informational</span> <span>P5</span> </div> <p data-bbox="119 1467 981 1512">Informational submissions (also known as "P5" or "Priority 5") are vulnerability submissions that are valid but out-of-scope or are "won't fix" issues, such as best practices.</p>	<ul data-bbox="1013 1422 1284 1512" style="list-style-type: none"> <li>• Lack of code obfuscation</li> <li>• Autocomplete enabled</li> <li>• Non-exploitable SSL issues</li> </ul>



## Bugcrowd's Vulnerability Rating Taxonomy

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

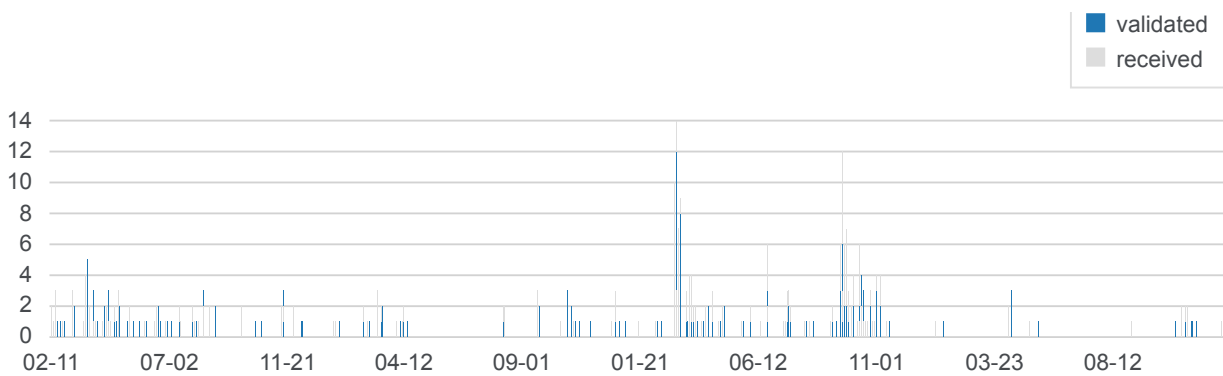


# Appendix

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

## Submissions over time

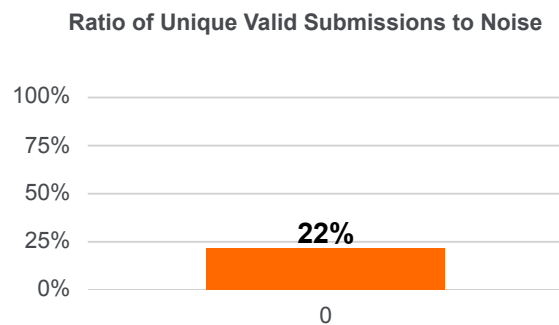
The timeline below shows submissions received and validated by the Bugcrowd team:



## Submissions signal

A total of **208** submissions were received, with **45** unique valid issues discovered. Bugcrowd identified **24** informational submissions, **16** duplicate submissions, removed **147** invalid submissions, and is processing **0** submissions. The ratio of unique valid submissions to noise was **22%**.

Submission Outcome	Count
Valid	45
Informational	24
Invalid	147
Duplicate	16
Processing	0
<b>Total</b>	<b>208</b>



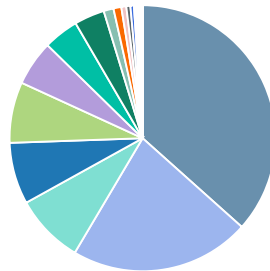




## Bug types overview

This distribution across bug types for the Ongoing Bug Bounty program only includes unique and valid submissions.

Average On-Demand Program



- Other
- Cross-Site Scripting (XSS)
- Server Security Misconfiguration
- Broken Access Control (BAC)
- Broken Authentication and Session Management
- Cross-Site Request Forgery (CSRF)
- Sensitive Data Exposure
- Server-Side Injection
- Unvalidated Redirects and Forwards
- Mobile Security Misconfiguration
- Application-Level Denial-of-Service (DoS)
- Insufficient Security Configurability
- Insecure Direct Object References (IDOR)
- Using Components with Known Vulnerabilities
- Missing Function Level Access Control
- Insecure OS/Firmware
- Insecure Data Storage
- Automotive Security Misconfiguration
- Insecure Data Transport
- Broken Cryptography
- Client-Side Injection
- Privacy Concerns
- External Behavior
- Lack of Binary Hardening
- Network Security Misconfiguration

## Spend of program rewards pool

During this Ongoing Bug Bounty Program, about **92%** of the total allocated reward pool of **\$17,800** was paid. A number of other statistics regarding the Ongoing Bug Bounty Engagement's payouts are shown below.

<b>\$16,350.00</b>	<b>\$16,350.00</b>	<b>\$1,450.00</b>
Total paid out to Researchers	Total to be paid to Researchers	Remaining prize pool
<b>\$4,400.00</b>	<b>\$100.00</b>	<b>\$743.18</b>
Highest paid reward	Lowest paid reward	Average reward



## Top 3 highest paid submissions

Title	Reward
<u>Local File Inclusion Vulnerability in Download Integration Indexing Log via 'filename=' Parameter [Manage integrations]</u> ec183207-85e7-42c1-9fd2-cc48d053cda0	\$4,400.00
<u>SSRF to EC2 Metadata   com.xiplink.jira.git.jira_git_plugin</u> ccbf8414-9ffd-4f60-9eb0-ac781c460489	\$1,500.00
<u>LFI on https://gitforjiracloud.bigbrassband.com</u> 86dde149-ae7c-45ae-aa78-1f53d113501e	\$1,500.00



Bugcrowd Inc.  
921 Front St  
Suite 100 San Francisco, CA 94111  
(888)361-9734

January 17 2024

# Closing Statement

## Introduction

This report shows testing of **GitKraken** between the dates of **02/11/2020 - 01/17/2024**. During this time, **128** researchers from Bugcrowd submitted a total of **208** 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. **Git Integration for Jira - Cloud Hosted** - <https://marketplace.atlassian.com/apps/4984/git-integration-for-jira?hosting=cloud&tab=overview>
2. **Git Integration for Jira - Server Hosted** - <https://marketplace.atlassian.com/apps/4984/git-integration-for-jira?hosting=server&tab=overview>
3. **Git Integration for Jira - Data Center Hosted** - <https://marketplace.atlassian.com/apps/4984/git-integration-for-jira?hosting=datacenter&tab=overview>
4. <https://marketplace.atlassian.com/apps/1219270/dev-info-for-jira?hosting=cloud&tab=overview>
5. **CI/CD For Jira** - <https://marketplace.atlassian.com/apps/1228578/ci-cd-for-jira?hosting=cloud&tab=overview>
6. **Team Insights for Jira (Preview)** - <https://marketplace.atlassian.com/apps/1230390/team-insights-for-jira-preview?hosting=cloud&tab=overview>

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

## GitKraken Program Overview

An GitKraken 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:

Technical Severity	Count
<b>Critical</b> vulnerabilities	3
<b>Severe</b> vulnerability	1
<b>Moderate</b> vulnerabilities	16
<b>Low</b> vulnerabilities	3
<b>Informational</b> vulnerabilities	22