Bugcrowd Ongoing program results

Report created on October 05, 2021

Report date range: July 01, 2021 - September 30, 2021

Prepared by

bmarriott@atlassian.com
**Executive summary**

**Statuspage** 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 **Statuspage's** targets during the period of: 07/01/2021 – 09/30/2021.

For this Ongoing Program, submissions were received from **12** unique researchers.

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

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

1. **Reconnaissance**
   - Gathering information before the attack

2. **Enumeration**
   - Finding attack vectors

3. **Exploitation**
   - Verifying security weaknesses

4. **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 Statuspage 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:

- `manage.statuspage.io`
- `*.statuspage.io`

All details of the program scope and full program brief can be reviewed in the Program Brief.
Findings by severity

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

Critical

Critical severity submissions (also known as "P1" or "Priority 1") are submissions that are escalated to Statuspage 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.

High

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.

Medium

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

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.

Informational

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.

EXAMPLE VULNERABILITY TYPES

Remote Code Execution
Vertical Authentication Bypass
XML External Entities Injection
SQL Injection
Insecure Direct Object Reference for a critical function

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

Cross-Site Scripting with limited impact
Cross-Site Request Forgery for an unimportant function
External Server-Side Request Forgery

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
Appendix

Included in this appendix are auxiliary metrics and insights into the Ongoing 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 17 submissions were received, with 3 unique valid issues discovered. Bugcrowd identified 1 informational submission, 4 duplicate submissions, removed 8 invalid submissions, and is processing 1 submissions. The ratio of unique valid submissions to noise was 19%.

<table>
<thead>
<tr>
<th>Submission Outcome</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid</td>
<td>3</td>
</tr>
<tr>
<td>Informational</td>
<td>1</td>
</tr>
<tr>
<td>Invalid</td>
<td>8</td>
</tr>
<tr>
<td>Duplicate</td>
<td>4</td>
</tr>
<tr>
<td>Processing</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>17</td>
</tr>
</tbody>
</table>
Bug types overview

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

This report shows testing of **Statuspage** between the dates of **07/01/2021 - 09/30/2021**. During this time, **12** researchers from Bugcrowd submitted a total of **17** vulnerability submissions against **Statuspage's** targets. The purpose of this assessment was to identify security issues that could adversely affect the integrity of Statuspage. Testing focused on the following:

1. manage.statuspage.io
2. *.statuspage.io

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

Ongoing Program Overview

An Ongoing 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 program, Bugcrowd discovered the following:
<table>
<thead>
<tr>
<th>Count</th>
<th>Technical Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><strong>Critical</strong></td>
</tr>
<tr>
<td>1</td>
<td><strong>High</strong></td>
</tr>
<tr>
<td>0</td>
<td><strong>Medium</strong></td>
</tr>
<tr>
<td>1</td>
<td><strong>Low</strong></td>
</tr>
<tr>
<td>1</td>
<td><strong>Informational</strong></td>
</tr>
</tbody>
</table>