Opsgenie

Opsgenie is a modern incident management platform for operating always-on services, empowering Dev & Ops teams to plan for service disruptions and stay in control during incidents.

Opsgenie

Bugcrowd Ongoing program results

Report created on July 09, 2020

Report date range: April 01, 2020 - June 30, 2020

bugcrowd

Prepared by

bmarriott@atlassian.com

Table of contents

Executive summary

Opsgenie 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 engagement 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 **Opsgenie's** targets during the period of: **04/01/2020 – 06/30/2020**.

For this Ongoing Program, submissions were received from **68** unique researchers.

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

This report is just a summary of the information available.

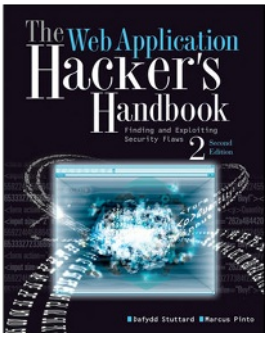
All details of the program's findings — comments, code, and any researcher provided remediation information — can be found in the Bugcrowd Crowdcontrol platform.

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:

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



Scope

Prior to the Ongoing program launching, Bugcrowd worked with Opsgenie 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:

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

```
Opsgenie (Android)
```

```
Opsgenie (IoS)
```

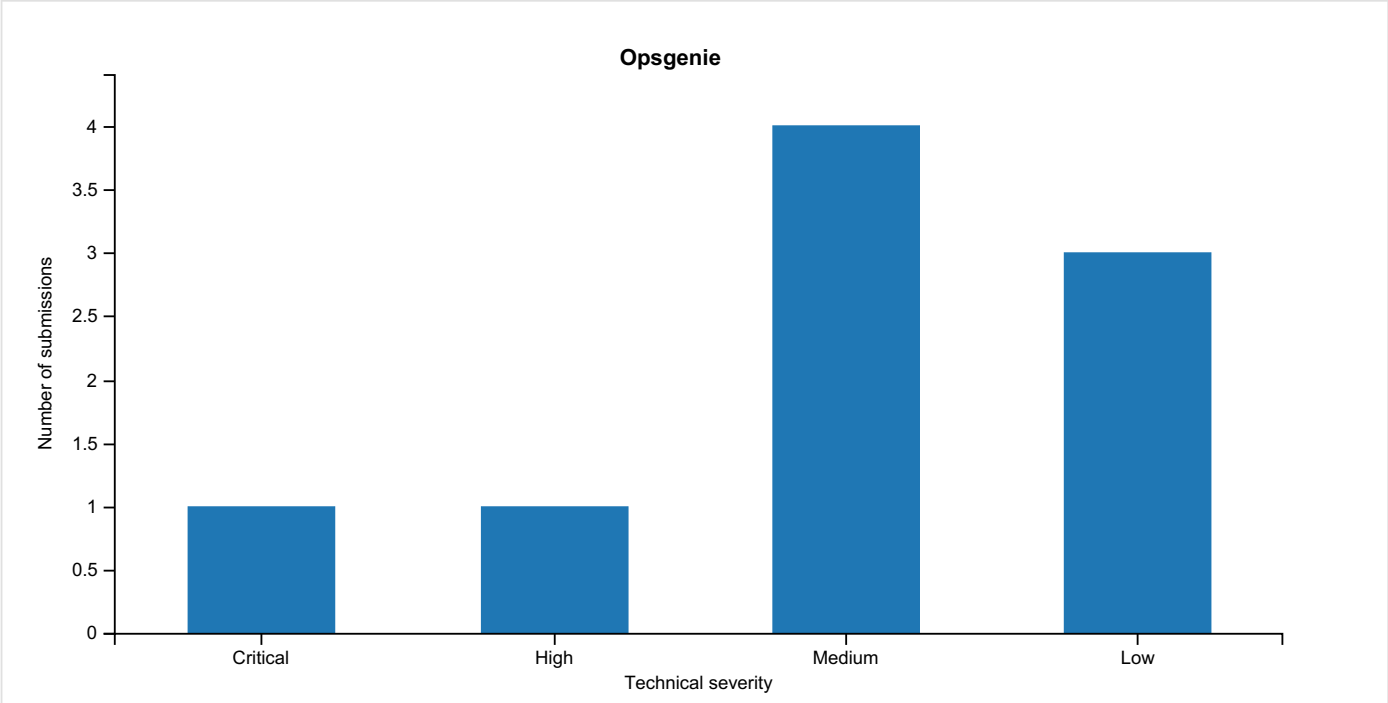
```
*.opsgeni.us
```

```
mobileapp.opsgeni.us
```

```
app.opsgeni.us
```

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	EXAMPLE VULNERABILITY TYPES
Critical Critical severity submissions (also known as "P1" or "Priority 1") are submissions that are escalated to **Opsgenie** 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.	• Remote Code Execution • Vertical Authentication Bypass • XML External Entities Injection • SQL Injection • Insecure Direct Object Reference for a critical function
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.	• 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
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.	• Reflected Cross-Site Scripting with limited impact • Cross-Site Request Forgery for an important function • Insecure Direct Object Reference for an unimportant function
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.	• Cross-Site Scripting with limited impact • Cross-Site Request Forgery for an unimportant function • External Server-Side Request Forgery
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.	• Lack of code obfuscation • Autocomplete enabled • Non-exploitable SSL issues

Bugcrowd's Vulnerability Rating Taxonomy

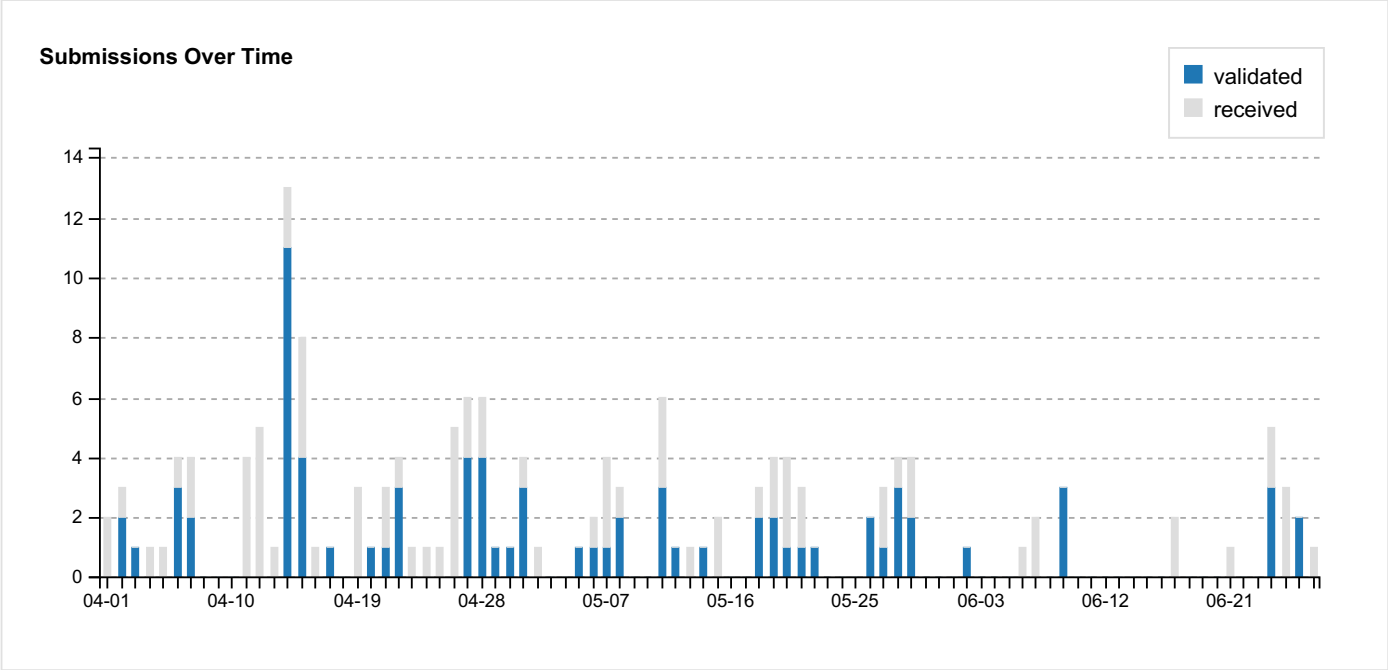
More detailed information regarding our vulnerability classification can be found at: https://bugcrowd.com/vrt

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

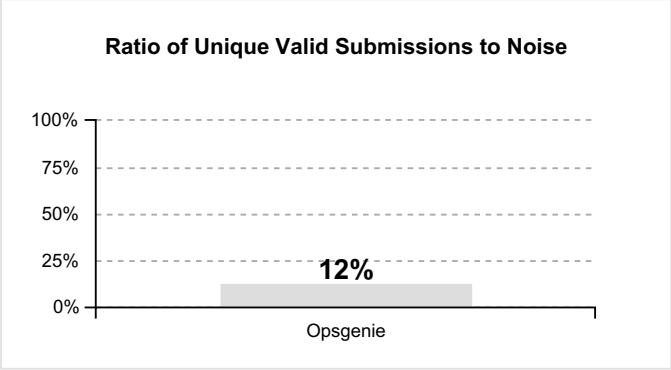
The timeline below shows submissions received and validated by the Bugcrowd team:



Submissions signal

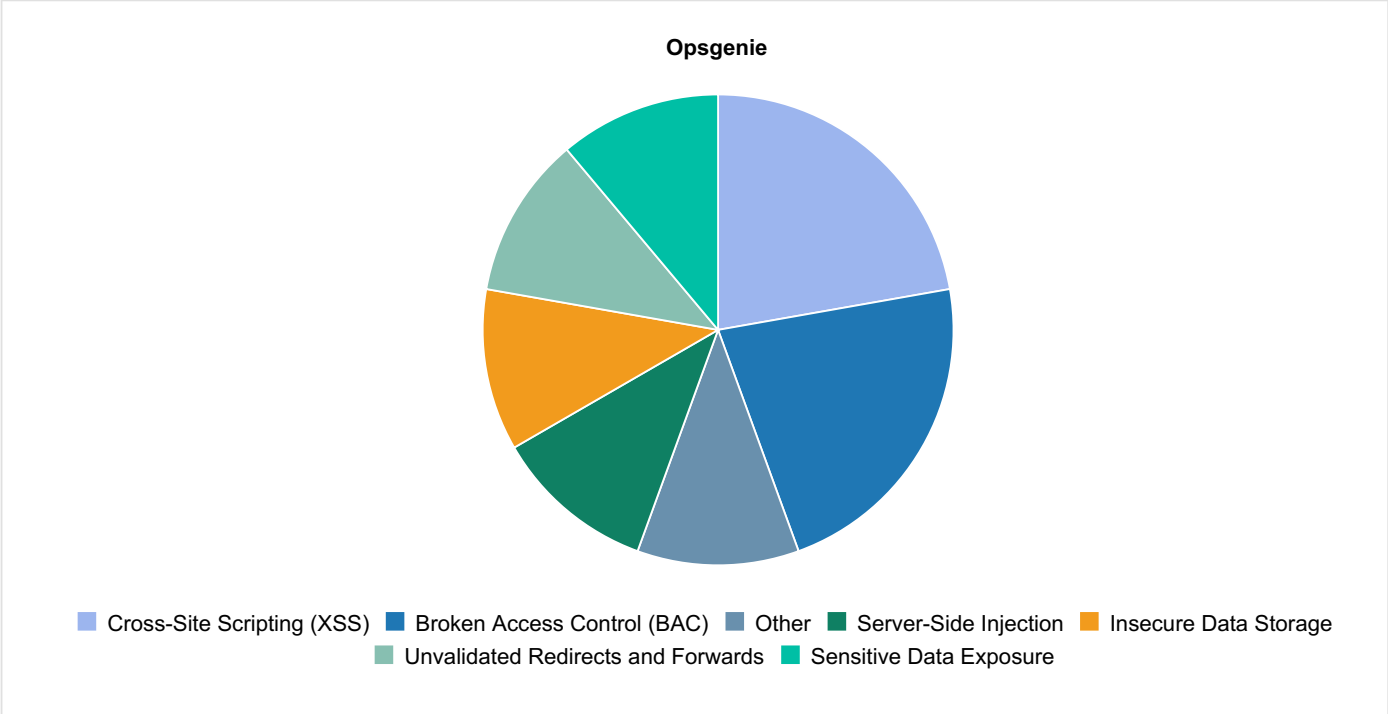
A total of **83** submissions were received, with **10** unique valid issues discovered. Bugcrowd identified **32** duplicate submissions, removed **39** invalid submissions, and is processing **2** submissions. The ratio of unique valid submissions to noise was **12%**.

Submission Outcome	Count
Valid	10
Invalid	39
Duplicate	32
Processing	2
Total	**83**



Bug types overview

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



Closing statement

bugcrowd

July 09, 2020

Bugcrowd Inc.
921 Front St
Suite 100
San Francisco, CA 94111

Introduction

This report shows testing of **Opsgenie** between the dates of **04/01/2020 - 06/30/2020.** During this time, **68** researchers from Bugcrowd submitted a total of **83** vulnerability submissions against **Opsgenie's** targets. The purpose of this assessment was to identify security issues that could adversely affect the integrity of Opsgenie. Testing focused on the following:

1. **Opsgenie (Android)**
2. **Opsgenie (IoS)**
3. ***.opsgeni.us**
4. **mobileapp.opsgeni.us**
5. **app.opsgeni.us**

The assessment was performed under the guidelines provided in the statement of work between **Opsgenie** 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 engagement, Bugcrowd discovered the following:

Count	Technical Severity
1	Critical vulnerability
1	High vulnerability
4	Medium vulnerabilities
3	Low vulnerabilities
1	Informational finding