INSTRUCTURE

Penetration Test Results

Engineering, Security, and Operations

February 2021

In this guide, you'll learn:

How we partner with the third-party provider BugCrowd for ongoing penetration testing of our applications and services.

Table of Contents

Executive Summary

This is Instructure's 10th annual open security audit and once again Instructure engaged Bugcrowd, Inc. to perform an Ongoing Bounty Program, commonly known as a crowd-sourced penetration test for its products.

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 Canvas LMS, Bridge, Studio, Practice, and Portfolium's targets during the period of: 01/01/2020 – 12/31/2020.

For this Ongoing Program, submissions were received from 60 unique researchers.

The continuation of this document summarizes the finding, analysis, and recommendations from the Ongoing Bounty Program performed by Bugcrowd for the 2021 Penetration Test Report.

If you are interested in joining our bug bounty program as a security researcher, please contact security@instructure.com with your Bugcrowd username and we will get you hooked up!

Keep learning,

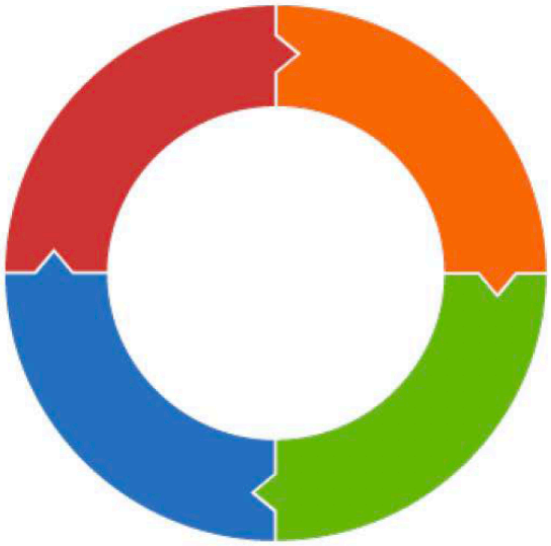
Josh Blackwelder,
Sr. Director and Head of Security
security@instructure.com

Reporting and Methodology

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.

01
Reconnaissance
Gathering information before the attack

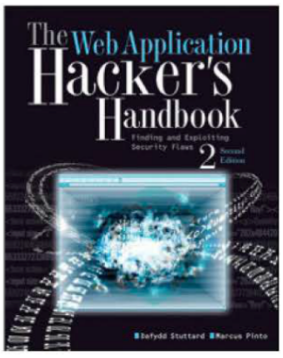
02
Enumeration
Finding attack vectors



04
Documentation
Collecting results

03
Exploitation
Verifying security weaknesses

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



NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Targets and Scope

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

iOS Applications

iOS App: Canvas Student

iOS App: Polls for Canvas

iOS App: Canvas Teacher

iOS App: Canvas Parent

Android Applications

Android App: Canvas Student

Android App: Polls for Canvas

Android App: Canvas Teacher

Android App: Canvas Parent

Canvas LMS

https://bugcrowd-tc.instructure.com

https://commons-pdx-edge.inseng.net

https://catalog-bugcrowd.inscloudgate.net

Canvas Studio

https://sectest.beta.instructuremedia.com

Portfolium

https://*.qa.ops.portfolium.net

Bridge Suite

https://*.suite.staging.bridgeapp.com

https://bugcrowd*.staging.bridgeapp.com

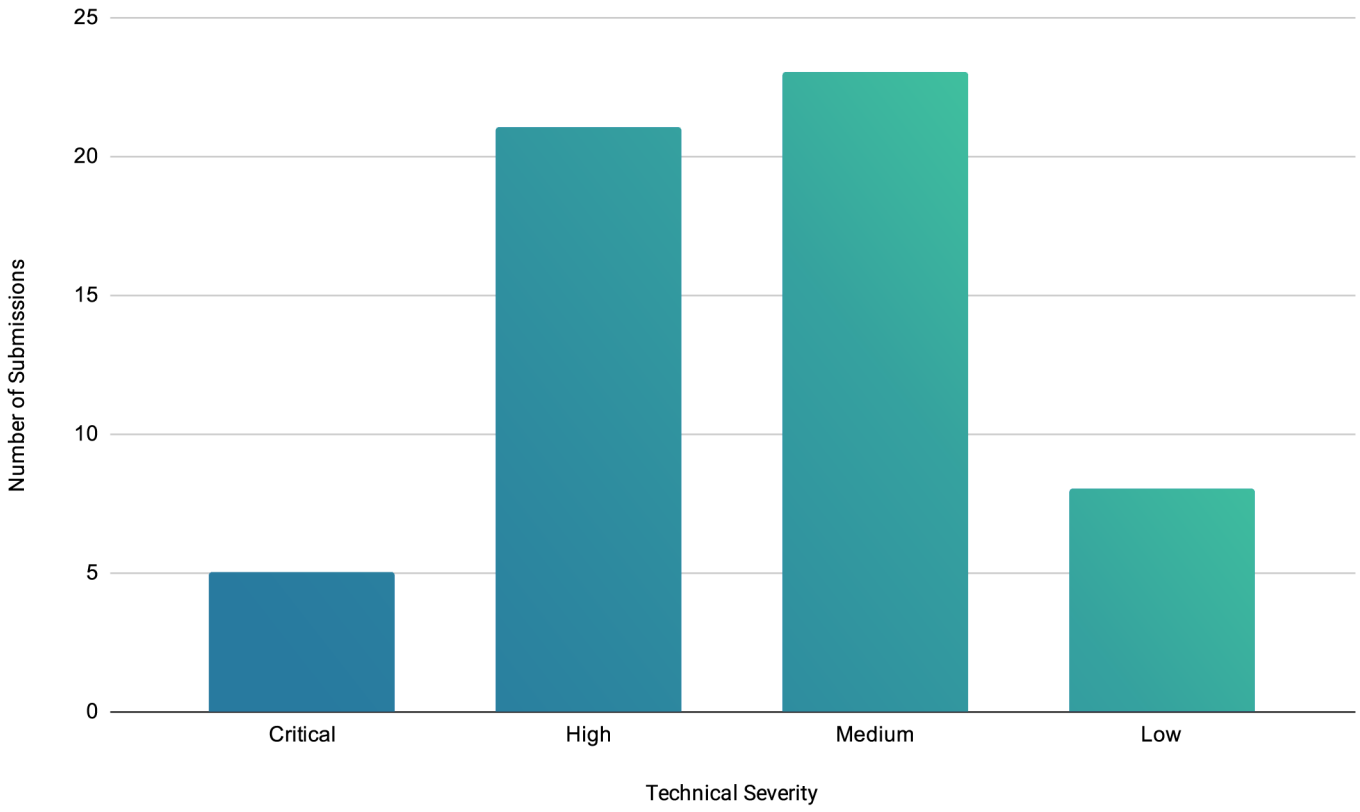
https://bugcrowd*.perform.stage.bridgeapp.com

https://*.stage.practice.xyz

Findings Summary

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 Instructure 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
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 • 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 a 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 Bugcrowd's vulnerability classification can be found at: https://bugcrowd.com/vrt

FINDINGS TABLE

The following tables list all valid assessment findings from the program.

MOBILE APPLICATIONS

Title	VRT	Duplicates	Priority	State
Stored XSS in iOS App via URL Fragment	Cross-Site Scripting (XSS)		P2	Resolved
Open redirect (Android only)	Unvalidated Redirects and Forwards		P4	Resolved

CANVAS LMS

Title	VRT	Duplicates	Priority	State
Stored XSS via Student via bypass of htmlEscape JS function	Cross-Site Scripting (XSS)		P2	Resolved
Unauthenticated Canvas DoS via GraphQL	Application-Level Denial-of-Service (DoS)		P2	Resolved
Stored XSS via Group	Cross-Site Scription (XSS)		P3	Resolved
Reflected Cross Site Scripting on Quizzes can be used to steal access tokens	Cross-Site Scripting (XSS)		P3	Resolved

Title	VRT	Duplicates	Priority	State
Stored XSS via files domain + Additional chainable vulnerabilities	Cross-Site Scripting (XSS)		P3	Resolved
Stored XSS via course link validator	Cross-Site Scripting (XSS)		P3	Resolved
Bypassing 2FA using Backup Codes	Broken Authentication and Session Management		P3	Resolved
Content-security-policy (CSP) bypass	Server-side Injection		P3	Resolved
Stored XSS via Gradinng Schema Letter	Cross-Site Scripting (XSS)		P3	Resolved
Accessing any course's rubric challenge details	Broken Access Control (BAC)		P3	Resolved
DOM XSS via postMessage	Cross-Site Scripting (XSS)		P3	Resolved
Stored XSS via Group	Cross-Site Scripting (XSS)		P3	Resolved
DOM XSS via prototype pollution	Cross-Site Scripting (XSS)		P3	Resolved
Dashboard_positions API possibility to string a number	Application-level Denial-of-service (DoS)		P3	Resolved
Open Redirection found on Quizzes	Unvalidated Redirects and Forward		P4	Resolved

CANVAS STUDIO

Title	VRT	Duplicates	Priority	State
Studio Admin Permissions	Server Security Misconfiguration		P1	Resolved
Stored XSS via Invalid Sanitize of Video Subtitles	Cross-Site Scripting (XSS)		P2	Resolved
Stored XSS via Chapters kind of Media Track	Cross-Site Scripting (XSS)		P3	Resolved
BAC issue with Studio Quizzes	Broken Access Control		P3	Resolved

PORTFOLIUM

Title	VRT	Duplicates	Priority	State
SSRF	Broken Access Control (BAC)	1	P1	Resolved
SSRF with DNS Rebinding used to steal AWS Token	Broken Access Control (BAC)		P1	Resolved
SSRF at /proxy/users/avatar	Broken Access Control (BAC)		P2	Resolved
AngularJS Injection + Stored XSS	Cross-Site Scripting (XSS)		P2	Resolved
Trigger connection request on visiting profile page (HTML injection + CSRF)	Cross-site Request Forgery (CSRF)		P2	Resolved
SSRF and Protocol Smuggling to issue commands to Elastic Search	Server Security Misconfiguration	1	P2	Resolved
Stored XSS editing project	Cross-Site Scripting (XSS)		P2	Resolved

Title	VRT	Duplicates	Priority	State
Stored XSS via project “Paste a link”	Cross-Site Scripting (XSS)		P2	Resolved
Stored XSS via Angular Template Injection in message subject	Cross-Site Scripting (XSS)		P2	Resolved
Stored XSS via Injection in MarkDown link of Project Description + disclosure of real IP	Cross-Site Scripting (XSS)	3	P2	Resolved
Reflected XSS via page_uri parameter at oauth/redirect endpoint	Cross-Site Scripting (XSS)		P3	Resolved
Sensitive info leak using JSONP callback	Sensitive Data Exposure		P3	Resolved
Reflectted XSS via Angular Template Injection in Sort Parameter	Cross-Site Scripting (XSS)		P3	Resolved
Account takeover through OAuth Sign in misconfiguration	Server Security Misconfiguration		P4	Resolved
Access Everything via API endpoints without email verification	Other		P4	Resolved
SSRF and DoS using SVG image	Broken Access Control		P4	Resolved

BRIDGE

Title	VRT	Duplicates	Priority	State
Privilege escalation – able to see any other user’s assessments	Broken Authentication and Session Management		P3	Resolved

Title	VRT	Duplicates	Priority	State
Privilege escalation using “author_id” and “assignee_id” to send/assign task as another user	Broken Authentication and session Management		P3	Resolved
APP DoS – able to block assessment function for all users	Application-level Denial-of-service (DoS)		P3	Resolved
A user must be inactive as soon as deleted from an organization	Broken Authentication and Session Management		P4	Resolved

OTHER

Title	VRT	Duplicates	Priority	State
Google Calendar access control misconfiguration	Broken Access Control (BAC)		P1	Resolved
Insecure salesforce object permissions @pay.getbridge.com	Broken Access Control (BAC)		P1	Resolved
High impact subdomain takeover via dangling NS records pointed to AWS EC2 Elastic Ips at exec.instructure.com	Server Security Misconfiguration		P2	Resolved
High impact subdomain takeover via dangling NS records pointed to AWS EC2 Elastic Ips at plotly.instructure.com	Server Security Misconfiguration	1	P2	Resolved
Subdomain takeover via dangling NS records on AWS Route 53 at try.getbridge.com	Server Security Misconfiguration		P2	Resolved
High impact subdomain takeover via dangling NS records pointed to AWS EC2 Elastic Ips at liferay-test.cisco.instructure.com	Server Security Misconfiguration		P2	Resolved

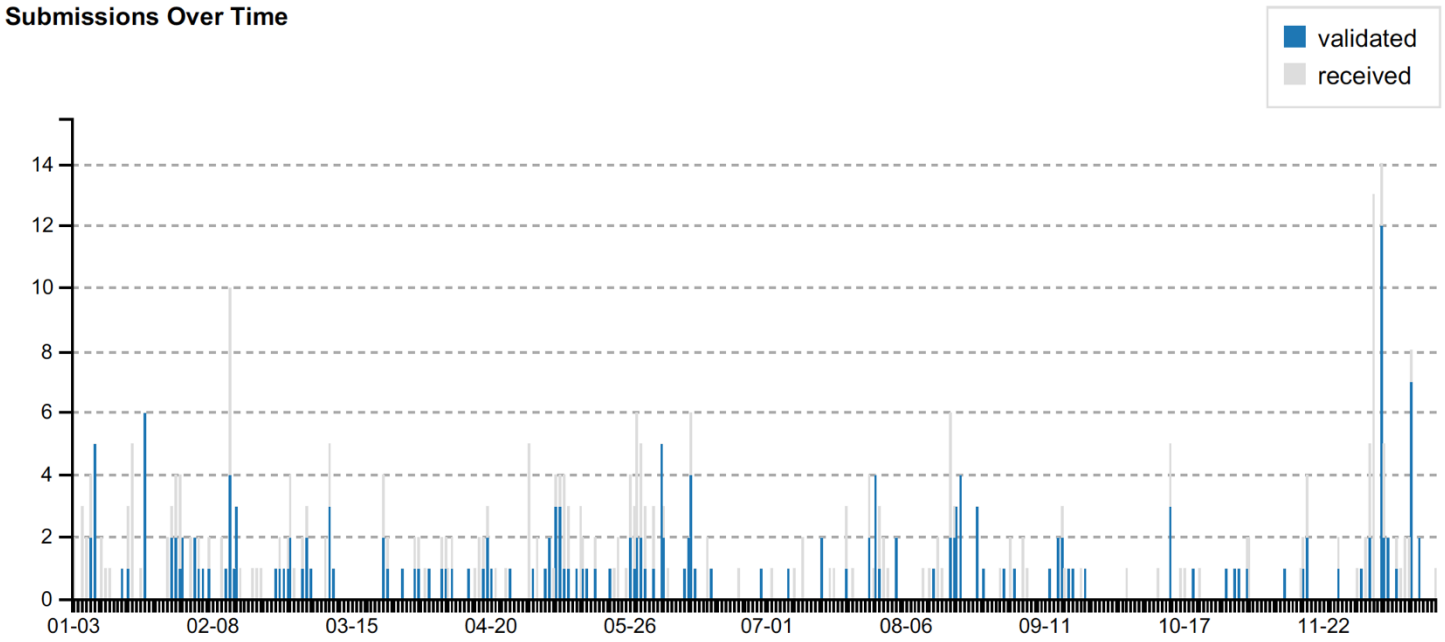
Title	VRT	Duplicates	Priority	State
Subdomain takeover via unclaimed AWS S3 Bucket for blog.portfolium.com	Server Security Misconfiguration		P2	Resolved
Subdomain takeover of faqs.instructure.com pointing to ScreenSteps	Server Security Misconfiguration		P2	Resolved
High impact subdomain takeover via dangling NS records pointed to AWS EC2 Elastic IPs at opsbox01.instructure.com	Server Security Misconfiguration		P2	Resolved
Eduappcenter organization takeover	Broken Access Control (BAC)		P2	Resolved
Subdomain takeover of HubSpot hosted domain	Server Security Misconfiguration		P2	Resolved
Subdomain takeover via unclaimed Heroku instance	Server Security Misconfiguration		P3	Resolved
Domain takeover through hubspot	Server Security Misconfiguration		P3	Resolved
Subdomain takeover	Server Security Misconfiguration		P3	Resolved
Internal Service Desk Exposed	Broken Access Control (BAC)	1	P3	Resolved
No cookie consent even after tracking leads to privacy violation	Privacy Concerns		P4	Resolved
Internal links insecure storage	Insecure Data Storage		P4	Resolved

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

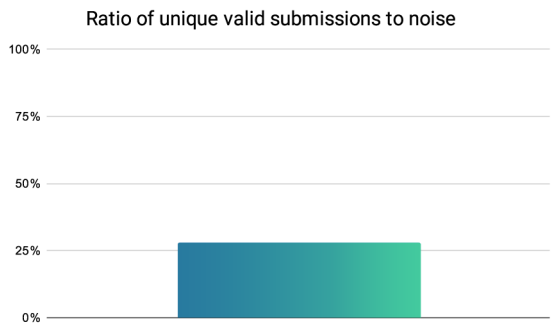
The timeline below shows submissions received and validated by the Bugcrowd team:



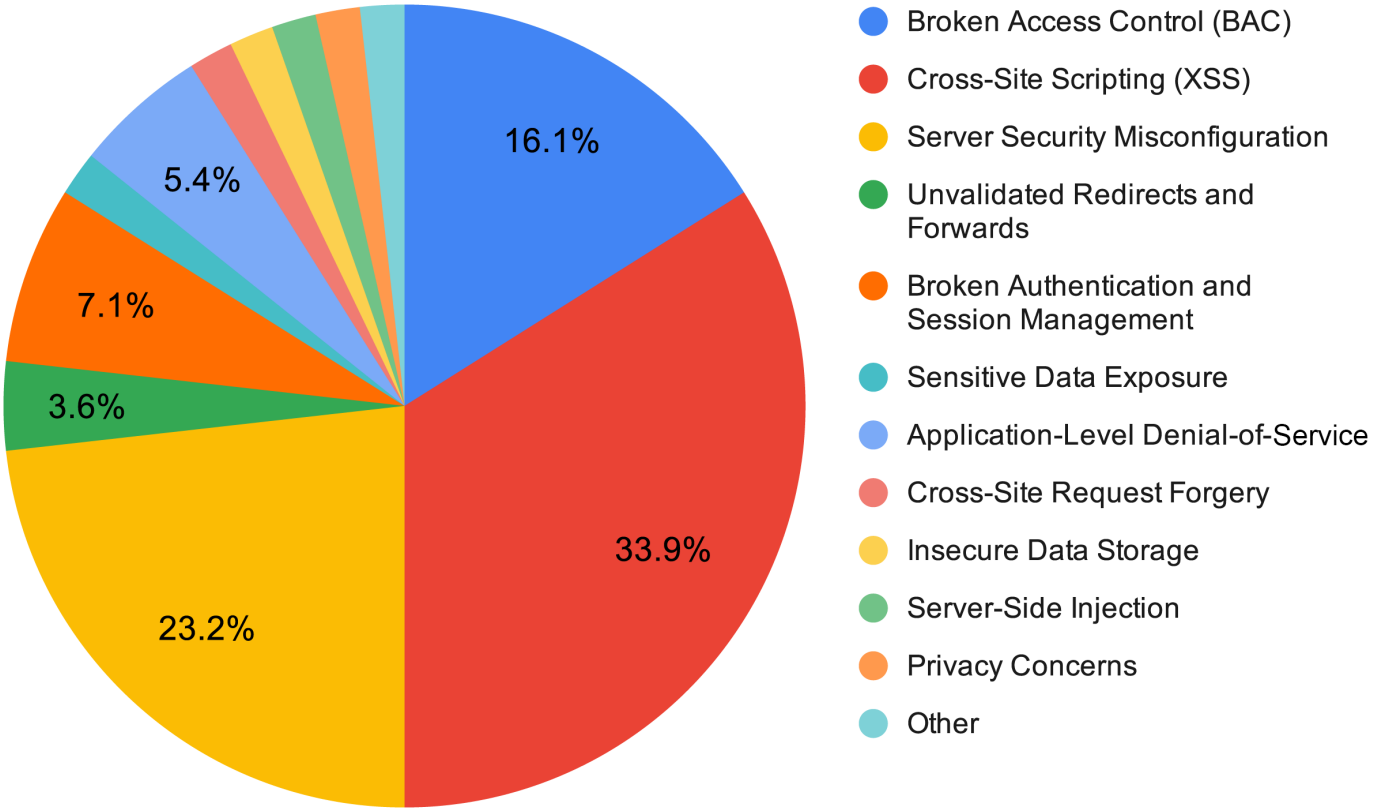
SUBMISSIONS SIGNAL

A total of **202** submissions were received, with **57** unique, valid issues discovered. Bugcrowd identified **38** duplicate submissions, removed **107** invalid submissions, and is processing **0** submissions. The ratio of unique, valid submissions to noise was **28%**.

Submission Outcome	Count
Valid	57
Invalid	107
Duplicate	38
Processing	0
Total	**202**



BUG TYPES OVERVIEW



Closing Statement

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
5	**Critical** vulnerabilities
21	**High** vulnerabilities
23	**Medium** vulnerabilities
8	**Low** vulnerabilities
0	**Informational** findings