Secure and Remote 3D Printing
Test Plan

Tiffanie Petersen - tpetersen2018@my.fit.edu

Isaiah Thomas - ithomas2018@my.fit.edu

Carl Mann - cmann2013@my.fit.edu

Nick Contrell - ncontrell2019@my.fit.edu

Sponsored by
Dr. Siddhartha Bhattacharyya - sbhattacharyya@fit.edu

Table Of Contents

1. Introduction

1.1. Objective

The objective of this test plan is to provide the conditions, procedures, and results for various test cases in regards to systems and functionalities presented in the requirements document. These test cases will help describe the function and nature of the Florida Tech 3D Printing Access Portal. This webpage will allow students to submit cad files from anywhere, whether it be on campus or anywhere in the world. Students should be able to securely send their cad objects to the portal in request of printing, and administrators should be able to review and approve them for the printer.

1.2. Purpose

The purpose of the test plan is to provide test cases that will aim to verify that various interactions between the client and host function properly. This will aim as a guide to explore possible bugs or vulnerabilities within the site that could possibly lead to detrimental effects. Not only will the usual input be considered, but unusual input will be provided in order to test the limits of our systems. These tests will also attempt to find as many potential vulnerabilities to security as possible, whether it be an exploit of the website or a dos attack.

1.3. Approach

For functional testing, these proposed test cases will be performed multiple times, in multiple environments which include web browsers such as: Mozilla Firefox, Google Chrome, Apple Safari, Microsoft Edge, and Internet Explorer 11. Additionally, these test cases will be performed on mobile devices to ensure that Bootstrap is implemented properly, and that functionality remains consistent regardless of a shift in view. We will utilize automation to quickly ensure that certain functionality remains to work. However, all manual inputs will also be manually checked to ensure proper functionality.

For security testing, we will combine automated testing along with manual tests to ensure that security isn't compromised. We will utilise tools from the Kali Linux toolchain to test as many exploitable avenues as possible. Tools such as Burp Suite and Slow Loris to check for ways to break into the portal or methods of DOSing the website.

2. Test Cases

2.1. Functionality Tests

2.1.1. Test Case 1: Network connectivity to website

<table>
<tr><td colspan="2"><b>TC001 - Network connectivity to website</b></td></tr>
<tr><td colspan="2"><b>Description:</b> Due to potential networking issues, it will be important to ensure that the web portable is visible both on the network that it is hosted on and on the open internet. When a new version of the portal is implemented, it is imperative that faculty check that the web portal is accessible both on their local network and from the public internet.</td></tr>
<tr><td colspan="2"><b>Goal:</b> Connect to web portal homepage from local network and public network.</td></tr>
<tr><td colspan="2"><b>Precondition:</b> A working browser and connection to the internet.</td></tr>
<tr><td colspan="2"><b>Start of Testing:</b> Connection to Secure 3D Printing Portal</td></tr>
<tr><td colspan="2"><b>End of Testing:</b> Redirection to homepage</td></tr>
<tr><td colspan="2"><b>Test Tools:</b> Testers will be given access to the current site to ensure that any changes from migration did not cause issues. An automated tool can be made to ensure that not only the homepage is visible but that all other pages are still accessible as well.</td></tr>
<tr><td><b>Test steps:</b></td><td><b>Expected results</b></td></tr>
<tr><td>1. Local connection
1.1. Insure that you are on the same network as the web server
1.1.1. Can be checked by typing “<i>ipconfig</i>” if in windows or “<i>ip addr</i>” if using linux.
1.2. The web server will print out a URL in the form “<i>https://XXX.XXX.XXX.XXX</i>” in the output.
1.3. Navigate to the address in a browser
2. Public connection
2.1. If a domain name has been given for the website, navigate to it in a browser
2.2. If not 2.1, then navigate to public ip
2.2.1. Find out public IP here
2.3. Navigate to public IP in the same form as 1.2.</td><td>1. Local connection will be guaranteed as long as it is tested on the same network
2. If the tester is using the domain name, then the tester will be brought to the homepage. If not, then DNS was not set up properly.
3. If network settings were done correctly, then the tester can view the home page from the public IP regardless of domain names.</td></tr>
</table>

2.1.2. Test Case 2: User creation/Registration

TC002 - User creation/Registration	
Description: In order for anyone to use the website, they must have an account accessible so that they can upload cad projects.	
Goal: Log in to a newly created user.	
Precondition: Tester has a valid my.fit.edu or fit.edu email.	
Start of Testing: Create an account using the registration page.	
End of Testing: Log in using newly created user.	
Test Tools: Testers will be given access to the test branch to ensure no testing data goes into release values. A tool can be made to automatically test this.	
Test steps:	**Expected results**
1. Navigate to "*Register*" in the navbar. 2. Enter a valid username and password. 3. Navigate to "*Login*" in the navbar. 4. Log in as created user.	1. Registration will fail if the email given isn't a relevant fit.edu or my.fit.edu email. 2. Registering with an existing email will bring up an error and prevent registration. 3. Illegal characters will bring up an error and prevent registration.

2.1.3. Test Case 3: Web Page Restrictions

TC003 - Web Page Restrictions	
Description: There are pages and sections of the portal that should be accessible only to specific users. In order to do that, we add labels to the users that are administrators in contrast to the normal users. With each new release, we need to insure that pages restricted to administrators can only be accessed by administrators	
Goal: Access administrator tab as a student user	
Precondition: The user has a valid *@my.fit.edu email.	
Start of Testing: Creating new student user.	
End of Testing: Accessing admin page with student account.	
Test Tools: Testers will be given access to the test branch to prevent testing data from leaking into release versions. Testers will be provided a default admin account to test admins access to the admin page as well.	
Test steps:	**Expected results**
1. Navigate to "*Register*" in the navbar. 2. Enter a valid username and password. 3. Log in as created user. 4. Navigate to "*Admin*" in the navbar.	1. Registering with an existing email will bring up an error and prevent registration. 2. Illegal characters will bring up an error and prevent registration. 3. Existing users will not become admin unless authorized by another admin 4. Login will fail if the password is incorrect. 5. Non admins who navigate to the Admin page will be given a 403 error code. 6. Non admins who navigate to the admin page will have their user and IP logged to internal website log.

2.2. Security tests

2.2.1. Test Case 1: Cross-Site Scripting Attack

<table>
<tr><th colspan="2">TC011 - Cross-Site Scripting Attack</th></tr>
<tr><td colspan="2"><b>Description:</b> Since the Access Portal will be a website utilizing hand written code, the server could potentially be vulnerable to cross site scripting attacks.</td></tr>
<tr><td colspan="2"><b>Goal:</b> Find information valuable for exploitation. This includes obvious plain text values in cookies, params and errors caused by certain actions.</td></tr>
<tr><td colspan="2"><b>Precondition:</b> A copy of the release server including relevant data is utilized as a test environment on the test branch.</td></tr>
<tr><td colspan="2"><b>Start of Testing:</b> Read in information from crawling the portal</td></tr>
<tr><td colspan="2"><b>End of Testing:</b> Find information indented to be hidden or an unexpected error.</td></tr>
<tr><td colspan="2"><b>Test Tools:</b> The tester will be able to use a browser. The tester will be given access to multiple web fuzzing tools, including Burp Suite.</td></tr>
<tr><td><b>Test steps:</b></td><td><b>Expected results</b></td></tr>
<tr><td>1. Identify pages which may be vulnerable. (Upload forms and post requests)
2. Use Burp Suite to analyze components of pages and form submissions.
3. Begin gearing fuzzer towards potential weaknesses identified.
4. Conduct fuzzing tests.</td><td>1. Proper sanitation of all user input should prevent the execution of foreign code.
2. Checks on the files uploaded and set constraints should take effect and block all potentially harmful submissions.</td></tr>
</table>

2.2.2. Test Case 2: Input Sanitation

TC012 - Input Sanitation

Description: Due to the web server utilizing both database management as well as user input. It will be imperative to sanitize all user input incase a user enters the website attempting to exploit the website by means of Command Injection or SQL Injection.

Goal: No inputs show signs of corresponding injection attack.

Precondition: The portal has inputs that can be sanitized and checked for logic exploitation.

Start of Testing: For inputs utilizing the database, checking if “ or ‘ can cause a server error.

End of Testing: Utilizing prior knowledge of SQL statements involved in inputs to exploit including character encoding.

Test Tools: Testers will have access to a browser to test inputs. Testers will have access to source code relevant to inputs being tested. Testers can utilize tools for automated Injection testing including SQLmap.

Test steps:	**Expected results**
1. Go to any page that has inputs. 2. Check the value the input gives in the source code and use it to determine potential attacks. 2.1. If it accesses the database, prioritize SQL injection. 2.2. If it runs a command on the Raspberry pi itself, prioritize Command Injection. 3. Using knowledge of the input, attempt to exploit it.	1. If any input returns an unmanaged error statement from a ” or ‘ used as input, the input needs to be checked for vulnerabilities. 2. If login input is exploitable, it could be possible to log into a user's account without a password.

2.2.3. Test Case 3: DOS Protections

<table>
<tr><td colspan="2"><b>TC013 - DOS Protections</b></td></tr>
<tr><td colspan="2"><b>Description:</b> If we can ensure that our code is secure from being taken over by a malicious foreign entity, then our last main worry will be DOS attacks. Maintainers will have to ensure that the web server has applications and firewall rules installed that help mitigate DOS attacks. Maintainers will also have to ensure that all user inputs cannot be used to DOS the website.</td></tr>
<tr><td colspan="2"><b>Goal:</b> Fail to DOS the website using regular DOS tools and website inputs.</td></tr>
<tr><td colspan="2"><b>Precondition:</b> A copy of the release server including relevant data is utilized as a test environment on the test branch. At least two separate machines used to test DOS attacks. One to run the DOS attack, and the other to test its successfulness.</td></tr>
<tr><td colspan="2"><b>Start of Testing:</b> Run automated DOS tools against the website</td></tr>
<tr><td colspan="2"><b>End of Testing:</b> Test user inputs for large inputs or recursive actions.</td></tr>
<tr><td colspan="2"><b>Test Tools:</b> Testers will have access to a browser to test DOS attacks utilizing the web server’s code. Testers will have access to a suite of DOS attack tools, including LOIC (Low Orbit Ion Cannon), and SlowLoris.</td></tr>
<tr><td><b>Test steps:</b></td><td><b>Expected results</b></td></tr>
<tr><td>1. Run standard Dos attacks
1.1. Test effectiveness of LOIC with “<i>loic portal.fit.edu</i>”
1.2. Test effectiveness of SlowLoris with “<i>slowloris portal.fit.edu</i>”
2. Test inputs for DOS attacks
2.1. Take printer file input and give it a file >1GB in size.</td><td>1. If the test site has difficulty getting connections, then the website is vulnerable to the tested DOS attack.</td></tr>
</table>