**Validation Plan for *P.L.A.T.Y.P.U.S***

**LockHeed Martin**

|  |  |
| --- | --- |
|  |  |
| Prepared by: | *Tyler Marolda* |
| Date: | *03-27-2020* |
| Version: | 1.0 |

Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Description** |
| 1.0 | *Tyler Marolda* | *03-27-2020* | Initial Release |

Table of Contents

[1 Introduction 4](#_Toc3503383)

[1.1 Objective 4](#_Toc3503384)

[1.2 Scope 4](#_Toc3503385)

[1.3 Definitions 4](#_Toc3503386)

[1.4 References 4](#_Toc3503387)

2 Roles and Responsibilities 5

[3 System Description 5](#_Toc3503388)

[4 Validation Activity Requirements 6](#_Toc3503389)

[4.1 Validation Approach 6](#_Toc3503390)

[4.2 Validation Methodology 6](#_Toc3503391)

[5 Assumptions, Exclusions and Limitations 7](#_Toc3503392)

[5.1 Assumptions 7](#_Toc3503393)

[5.2 Exclusions 7](#_Toc3503394)

[5.3 Limitations 7](#_Toc3503395)

[6 Testing Environment 7](#_Toc3503396)

[7 Traceability Matrix 8](#_Toc3503397)

# Introduction

## Objective

The objective is to define and outline the procedures *Team Platypus* plans to go through for validating system and user requirements of the *P.L.A.T.Y.P.U.S*. The *P.L.A.T.Y.P.U.S* is a system designed to assist business owners with the security of their computer systems.

## Scope

The scope of this document includes each component of *P.L.A.T.Y.P.U.S*. Components include:

-Network scanning tools

-GUI for user interaction

## Definitions

*P.L.A.T.Y.P.U.S* is the product being developed.

Integrated Development Environment (IDE) - A software application that provides comprehensive tools for computer programmers.

Command Line Interface (CLI) – an interface that uses a shell or command line instead of interactive graphics.

Graphic User Interface (GUI) – an interactive interface that allows users to interact with the program in an easy and intuitive manner

# 

# Roles And Responsibilities

\*More will be added to this section as project progresses

Tyler Marolda-

Development team

Patrick Moore-

Scrum master and development team

Elija Amponsah-

Development team

John Gaffne-

Product owner

Issac Duah-

Development team

Kevin Kropinack-

Development team

# System Description

The following is a high-level description of the system requirements and design choices made to fulfil them:

System Requirements: Windows Operating System

User Capabilities

* Users will be able to scan their system for vulnerabilities
* Users will be able to see the issues with their system
* Users will be able to get accurate cost reports for suggested fixes

Administrator Capabilities

* Administrators will be able to view the system’s specifications in greater detail

An in-depth description of all these high-level requirements can be found in the Requirements Document. To meet these system requirements, the following design decisions were made:

The *Team Platypus* is compatible with *windows* machines.

# Validation Activity Requirements

## Validation Approach

Our approach to validation is using unit testing to ensure each component of the *P.L.A.T.Y.P.U.S* functions as intended. We plan to use penetration testing and manual checking to test our reports. Testing will be done by the development team.

## Validation Methodology

The following sub-sections describe the methodology utilized to validate the system.

* + 1. Installation Qualification (IQ)

The Installation Qualification will validate the components of the *P.L.A.T.Y.P.U.S* related to the installation and setup of the program software on compatible workstation computers. The system will be qualified when the following conditions have been carefully tested and met:

* *Windows* is installed on the machine
* *P.L.A.T.Y.P.U.S* is connected to the machine

* + 1. Operational Qualification (OQ)

The Operation Qualification will validate that a user with a compatible workstation is able to utilize all the functionality and interactivity of the *P.L.A.T.Y.P.U.S*. The standard use cases that must be carefully tested for the operation qualifications of the application to be approved are defined below:

* User can input arguments to run *P.L.A.T.Y.P.U.S* via command line
* User can input arguments to run *P.L.A.T.Y.P.U.S* via GUI

The configured system is the system to be validated during OQ, which will be performed according to pre-approved test plans with defined acceptance criteria, using a qualified data set. OQ will generate documented test results that are subject to review and approval. All the OQ test documentation will form part of the validation package. OQ will confirm that the computerized processes perform their required function and meet the agreed business requirement and Functional Requirement Specification.

* + 1. Performance Qualification (PQ)

To ensure that user stories can be successfully completed as part of the application, our team has conducted the following performance qualification tests:

|  |  |
| --- | --- |
| **User Story** | **Validated By** |
| User’s ability to detect issues with their system | Running Nessus to check for vulnerabilities |
| User’s ability to see a detailed and accurate estimate of costs associated with fixing their issues | Displaying information relayed from database to GUI |

# Assumptions, Exclusions and Limitations

## Assumptions

It is assumed that the *P.L.A.T.Y.P.U.S* is installed on an external flash drive that a user can plug in and run the .exe file.

## Exclusions

If the user cannot figure out how to run a .exe program

If user cannot plug in a flash drive or operate a PC

User does not have general knowledge of software

## Limitations

Assumed that program will be run with no or limited network connectivity.

# Testing Environment

The Development team will be using a laptop with known vulnerabilities to unit test each component.

# Traceability Matrix

See the *P.L.A.T.Y.P.U.S* Traceability Matrix document.