SolarProx

Software Requirements Specification
(SRS) Document

MUSoftwareRequirementsSpecification.doc

Final Draft 1

January 27, 2020

Force Of Mercury

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Rough Draft Number 1	Trevor Ryan Jacob Sanford Austin Carper Kurt Neimayer	This is the original rough draft for Force of Mercury's product SolarProx.	01/27/21
Final Draft Number 1	Kurt Neimayer	Edits were made to the original rough draft number 1.	02/01/21

Contents

1 Introduction

1.1 Purpose

The purpose of this product is to enable users to log into a web interface and revert snapshots on machines to a previous state, instead of having to wait for the admin to do so for them. Users will also be able to input "flags" to obtain points for class assignments. Users will also be able to see notes put onto machines by the admin that could give them hints or just a note explaining the machine. The intended audience are students in the network security classes at Mount Union.

1.2 Description

SolarProx is a web interface that will allow users to log in and be able to revert virtual machines back to snapshots created by the administrator and be able to reboot these virtual machines if they go down. SolarProx will also allow for the administrator to add notes to machines so that users can view these notes. Users will only be able to read these notes and won't be able to edit or remove the notes. Another capability of SolarProx is that the administrator will be able to make virtual machines available and unavailable for the users so that users won't have access to all the virtual machines in the lab at once. The benefits of this software are that the administrator doesn't have to be contacted every time a virtual machine goes down. This will save a lot of time for the administrator and the user. The user won't have to wait for the administrator to see a message that a virtual machine is down. Instead, the user can just log into the web interface and reboot the machine themself.

1.3 Overview

This document has been organized into four major sections: Introduction, Description, Requirements, and Life-Cycle model. In this document an analysis of important terms and descriptions of different levels of technicality describe the project within the introduction. Within the description, an analysis of a description of how the project should work, including descriptions of Proxmox, how to maintain the software, and information of requirements to run the software is included. In the requirements section, a list of both programming and non-programming requirements are analyzed. Within the final section of this document, a description of a Design-To-Schedule life-cycle model with hints of evolution life-cycle model will be provided, along with a justification of the usage of the model.

1.4 Glossary of Terms

- **Proxmox -** A software system used to manage virtual machines and allowing a connection to them over the Internet.
- **Maven** - A build tool used to compile and create user runnable files with the extension .jar.
- **Active Directory** - A type of software that allows controlling permissions of groups of users, credentials, and other important information on a network.
- **API -** Application Programming Interface. A set of software designs to allow connections to existing code, to allow integrating another program or making writing the software easier.

1.5 Business Context

Ken Smith will be the external sponsor of Solar Prox. He is a professor at the University of Mount Union and would like to implement the application in his network security course. Using the web interface, students will be able to roll-back virtual computers on the network to resolve issues that may arise while manipulating the boxes. The students will also be able to see notes for the boxes if they are stuck on a difficult step. When applied in the classroom, Solar Prox will benefit the students' learning and reduce time spent troubleshooting the boxes.

2 Overall Description

2.1 Product Perspective

2.1.1 System Interfaces

- Active Directory will be used for authentication.
- Proxmox will be used to manage the state of the boxes.
- File interactions will occur on the server to manage descriptions and box values.

2.1.2 Memory Constraints

The only limits will be the limits of the lab environment that our client has provided for us. The product will be run from the client's lab so all of the hardware is being used on the client's end.

2.1.3 Operations

- Users will need to log in at the home page of the site.
- Users will be able to view all active boxes and click in to each of the active boxes' individual pages.
- The administrative users will be able to view and activate/disable all boxes in the network.
- Users will be able to view box descriptions and revert the boxes to their original state from within the individual pages.
- Load times will occur throughout operation of the site and the user will be shown a notification of success or failure each time an external action occurs.

2.2 Product Functions

The software will first allow users to log in with a set of credentials, and then they will be able to view available virtual machines that they can reboot and revert snapshots on. Another function will be for the admin to log in and be able to view all virtual machines in the lab. Admins will be able to make machines available or not available while also being able to create snapshots on machines. Admins can also write notes for machines for the users to view. Other features will include users being able to earn points by capturing a "flag" on a machine and entering the flag value into a field to earn points. Users will also be able to view notes on machines but won't be able to edit the notes.

2.3 Similar Systems

There are no existing systems or products that use the Proxmox API to create a web interface for users to manage virtual machines.

2.4 User Characteristics

The intended user for this product will be students taking the network security classes. Their experience can range from not experienced to very experienced. The intended user will more than likely not have any technical expertise, but some might. The admin of the machine will be the professor of the class, who will have a lot of experience, and will have technical expertise as well.

2.5 User Objectives

The system user should expect to be able to login to the web interface and be presented with machines and their IPs. These are the active machines as specified by the admin. They will be able to get into any of these "active" machines and mess around in them. Machines will have "flag" values hidden in them and if a user were to find a value, they can enter it into a field to earn points if the admin specified that machine to have points. There will also be notes on each machine written by the admin, but the users can only read these notes and cannot edit them. The user will also be able to revert machines back to a certain snapshot that is created by the admin, but the user won't be able to edit snapshots or create new snapshots.

2.6 Constraints

- Must be highly reliable.
- Must interface to other virtual machines.
- Backend Database for Password Storage.
- Project Deadline.
- use of Proxmox API as a web interface.
- Space for storing screenshots to roll back to.

3 Specific Requirements

3.1 Functional Requirements

Priority Scale: Low (1) – Medium (2) – High (3)

1. Low: Items that can be eliminated should the need arise, without adversely affecting the product. These items are not urgent and not as important to the final product.
2. Medium: Items that are desired by the customer and/or users of the system, but that may be postponed until a future release. These items are not urgent but are important parts of the final product.
3. High: Items that are mission critical and without which the system cannot function in a manner that is satisfactory to the customer. These items are urgently needed and important to the success of the final product.

● Creating the Web Interface.
○ A web interface is needed for logging into the actual service and to be able to see the available virtual machines. The web interface will also be made so users can revert virtual machines back to a snapshot.
○ Lack of similar systems using Proxmox to create a web interface requires us to start from nothing, without a perfectly accurate frame of reference.
○ Priority: High

● Creating the Backend Database.
○ A backend database is needed to store passwords for users for the login feature.
○ Given that it is meant to store passwords, if it is breached then the system is put at risk.
○ Priority: High

● Connecting backend database to web interface.
○ The backend database needs to be connected to the actual web interface for the login feature.
○ As Proxmox has not been used to make a web interface, it also may impact our ability to efficiently connect the database to the Web Interface.
○ A poor connection between the interface and database could lead to the system being compromised more easily.
○ Dependent on the Web interface and Backend Database already being finished.
○ Priority: High

● Setting up permissions for users and admin.

○ Users and admins will need different permissions to do different tasks. Users should only be able to see virtual machines that the admin has made active while the admin should be able to see every virtual machine in the lab.
○ Dependent on the Interface and backend being linked to setup the Logins so permissions could be tested.
○ Priority: Medium

● Connecting the web interface to virtual machines.
○ Users and Admins will need to be able to interact with the virtual machines in order to roll them back.
○ Connecting to a virtual machine may take considerable time to figure out.
○ Dependent upon the Web interface and backend being completed and linked together.
○ Priority: Medium

● Creating a score system for virtual machine boxes.
○ A score system is to be made so when a user finds the “flag” value for a virtual machine, they can input that value and get points for that machine. The admin can specify whether a machine has points or not.
○ Dependent upon the interface being able to access virtual machines.
○ Priority: Low

● Admin notes on machines.
○ The admin should be able to write up notes for a machine for the users to view. The users should only be able to view these notes.
○ Dependent upon the users and admins being able to interact with the virtual machines.
○ Priority: Low

● Admins need the rights to create snapshots.
○ Admins should be able to create a snapshot for any machine. Users should not have the ability to create or delete snapshots, as their only permission would be to revert to a certain snapshot.
○ Dependent upon the program being able to connect to virtual machines, and the admins and users’ permissions being set up.
○ Priority: Low

● Written logs to allow capturing information and errors.
○ Admins can review data on the host machine by browsing and reading log files to help diagnose issues and monitor users.
○ Dependent upon the interface being able to connect to virtual machines.
○ Priority: Low

3.2 User Interface Requirements

3.2.1 User Interface: Graphical (GUI) or Command-Line (CLI)

This system will have a graphical interface for users, that allows users to login and see virtual machine boxes that they can attack. They will also be able to revert these virtual machines back to a snapshot in case that virtual machine breaks. Users will also be able to enter in “flag” values for some boxes that give them points. Users will be able to see only the virtual machines that are available to them specified by the admin. The admin will be able to see every box in the lab and can make certain boxes available to the users.

3.2.2 Application Programming Interface (API)

No custom API is being produced to allow connections to Solarprox.

3.2.3 Diagnostics (Error Reporting and Usage Logs)

Logging errors and information related to the system will be done through log files in a log folder in the same directory as the program. Logged data will be stored in a format based upon the date of the program launch, until it is restarted. If the program was launched twice within the same day, it will continue to write to that day's log, based upon the name of the log. All logged information will contain a date and time for each message as well as a log message state. There are 3 different states the logs allow:

- Info - general information that is used to keep track of what the system does.
- Warning - information that could hint at need of maintenance or warning of something not properly happening such as a user having a strange permission or not having weak credentials for a user.
- Errors - Information containing data about users concerning information, such as crash info, data not properly saving/loading, etc.

Logging data will also be available in the console for the current session if the user decides to launch the application with a terminal such as BASH.

3.3 System Requirements

3.3.1 Hardware Interfaces

- Web Interface, must be able to connect users to specific virtual machines.
- the primary device that should be used for connecting to this service is a PC.
- Apache Tomcat’s default ports are 8080, but they are configurable by the user.

3.3.2 Communications Interfaces

- Internet access, and LAN access to other devices.

3.3.3 Software Interfaces

- Proxmox - allow connection between the clients and the local machines in the virtual environment. - Version 6
- Windows Server 2016 - Allows access to active directory for permissions.
- Tomcat Apache - allows for user hosting. - Version 9

3.4 Domain Requirements/Constraints

There are no domain requirements or constraints.

3.5 Non-Functional Requirements

3.5.1 Reliability

This software needs to be reliable because the users should be able to use this product without having to contact the admin. That is the whole point of this software system, so that the user can revert to snapshots and reboot virtual machines without needing the admin to do so.

3.5.2 Availability

This software should be available at all times so that users can use the product anytime of the day.

3.5.3 Security

- Activity logging.
- Data must be encrypted if deemed sensitive such as passwords.
- Go through pen testing as a post requirement to find any vulnerabilities that can be exploited.

3.5.4 Maintainability

The client has requested for SolarProx to be open source after the project's initial release as to allow for further development of the project and maintaining its usage as SolarProx's dependencies update. This would be achieved by use of a Creative Commons License.

3.6 Logical Database Requirements

User credentials will be stored so that a user can log into the web interface with their own unique credentials. This database will be in the form of Windows' Active Directory. It will be used often as it will be used whenever a user or administrator logs in or requests access to a device.

● 4 Software Life Cycle Model

○ 4.1 Choice of Software Life Cycle Model

Force Of Mercury will be using the Design to Schedule Life Cycle Model with parts of the Evolutionary Delivery Life Cycle Model.

○ 4.2 Justification for Choice of Model

Force of Mercury chose the Design to Schedule Life Cycle Model because it will have certain features that need to be functional within a certain time so having the Design to Schedule Life Cycle Model will ensure that the highest priority features will be completed within the given time period. Force Of Mercury also chose to incorporate parts of the Evolutionary Delivery Life Cycle Model so that certain features can be reviewed with our client.