HardnBot

Intelligent Server Hardening Software

Project ID: 19_20-J01

Preliminary Progress Review Report

W.M.K.M.W Wijekoon

IT16167742

Bachelor of Science (Hons) in Information Technology

Specialization in Cyber Security

Sri Lanka Institute of Information Technology

16th September 2019

HardnBot

Intelligent Server Hardening Software

Project ID: 19_20-J01

Preliminary Progress Review Report

W.M.K.M.W Wijekoon

IT16167742

Supervisor: Mr. Amila Senarathne

Bachelor of Science (Hons) in Information Technology

Specialization in Cyber Security

Sri Lanka Institute of Information Technology

Sri Lanka

16th September 2019

DECLARATION

I declare that this is my own work and this Preliminary Progress Review (PPR) report does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

……..

W.M.K.M.W. Wijekoon

(IT16167742)

The above candidate is carrying out research for the undergraduate Dissertation under my supervision.

Supervisor

…….

Mr. Amila Senarathne

Table of Contents

List of Figures

1. Introduction

1.1.Purpose

Preliminary progress review (PPR) is created to provide a clear understanding of the research project and its research components. And, by this document, we expect to explain in depth about the main features and their novelty and what are the hardware or software requirements needed in order to reach the full potential of the completed software (HardnBot) and how are we planning to achieve our final goal.

By using this preliminary progress review report, anyone can review the process of the workflow in each phase of the project and this document allows to identify any deviations of project outcomes.

This document is created for the research team and the supervisors to get a clear idea about functional and non-functional requirements, how to reach our research goal, and what technologies and tools to be used.

1.2.Scope

HardnBot is a software that has the capability to identify failed operating system compliances of a unix based servers and classify those failed compliances and use those data to apply industry recommended best practices or organizational required fixes to the unix based operating system of a server. Throughout this preliminary progress review document, we will explain how failed compliances classification process is going to achieve its goal and steps that needed to be taken.

Under this preliminary progress review document, following components are described.

a) Scan for compliance failures.
b) Classify them using machine learning approaches into severity levels as Critical high Medium and Low.

HardnBot consist with four main novel components,

- Issue classification
- Risk score prediction
- Intelligent hardening
- Backup and smart rollback

Within those main components there are subcomponents/functionalities and throughout this document, issue classification component and its subcomponents will be thoroughly discussed.

1.3. Overview

HardnBot – An automated unix server hardening software platform that has the capability to detect and classify poor/non/failed compliance issues and configurations of an unix server and applying hardening solutions according to the industry recommended best practices or organizational recommended best practices. HardnBot also has the capability of predicting the overall rick score by considering main and all available risk factors and if there is any abnormal behavior after the hardening process, HardnBot has the capability to roll back the modification to an accepted previous level.

Normally these hardening processes will be done by either network administrators, system administrators, outsourced professionals or server custodians by manually running scripts, commands and queries against the server and it will roughly take more than six hours to completely harden a single server in the infrastructure. Probability of a misconfiguration occurrence is higher because hardening will carry out with human interaction. Scenarios where a misconfiguration occurs, it may be hard to detect those issues since some issues cannot be identified via an error message. So, in a scenario like that, system administrators, server custodian or network administrators need to go back to the initial state of the server operating system via a backup image which will be a time-consuming task. By introducing a special feature/function for improve backup and rollback tasks, fully backup to the initial state will not be necessary. Rather than that the software itself can automatically identify the misconfigured spot and rollback only to that point. And by classifying these compliance issues, HardnBot can provide required data to the prediction algorithms as well as to the intelligent hardening features. Also classified (categorized) data can be displayed as a summary to reporting purpose.

HardnBot uses scanning mechanisms to find and identify every compliance issues and misconfiguration of a unix server and this goal is achieved by executing a structured and configured script against the target server. And it will extract all the failed compliances and format them to a CSV formatted file for later use in classification, prediction and hardening functions.

Predefined classification model will be used to the purpose of failed compliance classification, and the algorithm (model) maybe modified accordingly to maximum accuracy level. LDA (Latent Dirichlet allocation) models will be used in order to identify, process and classify text in above extracted CSV, and this models also may be modified. Research shows that LDA based Weighted Location LDA (WL-LDA) is also suitable model for this task. After the text processing is done, to continue with classification, we are going to try and select best suitable classification algorithm that supports maximum accuracy level for our dataset. According to literature reviews, SVM (Support Vector Machine) classifier is the most suitable classifier for this task, however, modified classifier called Huffman Tree SVM (HT-SVM) is also a suitable classifier for our classification problem.

After all the classification is done, a clean classified (categorized) and summarized failed compliance issues will be forwarded to the risk score prediction algorithm as a set of risk factors and later as a parameter to perform the hardening process.

2. Statement of Work

2.1.Background

Although there are vulnerability scanners which has the capability of scanning operating system compliance failures and although they can classify vulnerabilities according to the CVE scores provided by the NVD, they are not capable of providing classification for those identified compliance failures but only whether they are failed or passed.

Ratsameetip Wita and Yunyong Teng-Amnuay of Department of Computer Engineering, Chulalongkorn University, Bangkok, Thailand published a research paper in 2005 on *"Vulnerability Profile for Linux".* In this research, they talk about profiling identified vulnerabilities according to the CVE score of them. In their classification scheme, they consider four types of classification schemes namely,

1. Confidentiality violation
2. Integrity violation
3. Availability violation
4. System compromised

If a confidentiality violation occurs, it allows an attack to directly steal information from the system. Integrity violations allows an attack to directly change the information passing through the system. Availability violation results an attack that limit the genuine access to a genuine user (human or machine), Denial of service attacks (DOS) can be taken as an example. According to their research system compromised attacks gives the attacker the privilege to access the system in four different levels such as: run an arbitrary code, elevate privilege, account break-in, and finally root break in which can be the worst-case scenario.
Furthermore, these classifications are again grouped according to the severity level.

Damage Type	Severity Level		
	High	Medium	Low
Confidentiality	- Disclosure of information and system configuration in root/super user level	- Disclosure of system information and configuration in user level	- Disclosure of some no relevant information.

integrity	- Information and system configuration changed in root/super user level	-Information changed in user level	- Non-relevant information changed in another user level
availability	-Whole system crash or unavailable	- Some services unavailable -System temporary unavailable	- Some services temporary slow down with flooding
System compromised	-Root break-in -Account break-in -Run arbitrary code by root/super user privilege	- Privilege gain in some domain - Run arbitrary code by user privilege	- Run arbitrary code by another user privilege

[1]

Here they are classifying vulnerabilities of the system / server, but we are going to develop a software toolkit that is capable of identify and classify failed compliance issues of a server operating system.

Bo Shuai, Haifeng Li, Mengjun Li, Quan Zhang and Chaojing Tang of School of Electronic Science and Engineering National University of Defense Technology Changsha, Hunan, P. R. China published a research paper in 2013 on *"Automatic Classification for Vulnerability Based on Machine Learning*".* This research paper is based on vulnerability classification using machine learning methods based on LDA model and SVM. Word location information is introduced in to LDA model called WL-LDA (Weighted location LDA), which is somewhat better than typical language processing algorithms because it generate outcomes from vector space on themes other than on word, and a multi-class classifier called HT-SVM (Huffman Tree SVM) is developed that it could make a faster and more stable classification on the vulnerabilities [2].

The main idea of this research is that they classify vulnerabilities with new models based on existing LDA and SVM models and obtain more accurate and more effective outcome.

Vijaya MS, Jamuna KS and Karpagavalli S of GR Govindarajulu School of Applied Computer Technology Coimbatore, India published a research paper in 2009 on *"Password Strength Prediction using Supervised Machine Learning Techniques".* This research is targeted on the

password strength of a system and predict password strength of a system whether it's a strong password or a weak password using supervised machine learning techniques such as classification (discrete) and regression (continuous).

Here the password strength prediction is modeled as classification task and supervised machine learning techniques were used as mentioned above. In this research they used some common classification models such as,

1. Decision tree classifier
2. Multilayer Perception
3. Naïve Bayes Classifier
4. Support Vector Machine (SVM)

For testing and selecting the best classification model for the performance of the task. After some performed tests, they identified Support Vector Machine (SVM) as the most suitable model for this task [3].

Kai Liu, Yun Zhou, Qingyong Wang and Xianqiang Zhu of Science and Technology on Information Systems Engineering Laboratory National University of Defense Technology Changsha, China published a research paper in 2019 on *"Vulnerability Severity Prediction with Deep Neural Network".* Multiple deep learning methods for vulnerability text classification evaluation are proposed in this research paper. Three kinds of deep neural networks,

1. CNN,
2. LSTM,
3. TextRCNN

And one kind of traditional machine learning method

1. XGBoost

are used. Here all parameters tuned via experiments to improve the accuracy of the task. However, in this research they said that the deep neural network methods evaluate vulnerability risk levels better, compared with traditional machine learning methods but it costs more time to train. This research says that they scored 93.95% accuracy level when training the model [4].

According to these literature reviews, there are some ideas and methods that could be consider in order to achieve our primary goal in classification the compliance issues of a server operating system.

2.2.Identification and significance of the problem

When performing a hardening process, system administrators, network administrators, server custodians or outsourced expertise need to ensure security of the operating system that runs on a server (in our case its unix based servers) database and application because a single mistake can affect the whole production line which the server is in. Even though there are many scanning tools that has the capability of scan a server and identify compliance failures along with the solutions, the solutions are going to apply with a human interaction. So, the probability of mistake occurrence is higher. And manual hardening process consume much more resources such as time, human, cost likewise. As a solution for the lack of resources, organizations are tending to consider about hiring external professionals and assets to perform the hardening task. In a scenario like this, internal critical classified information might have a possible chance to expose via outsourced professionals intentionally or unintentionally and leave the server in a critical position of been hacked or data leaked. And there is a compulsory requirement of the root (administrator) access to the server terminal in order to scan and perform operating system hardening. Also, the server needs to be temporarily out of live production, because operating system hardening cannot be performed while the server is in live production environment. So, when a critical day-to-day serving server is downed for maybe more than six hours to perform operating system hardening, it will be a critical impact on the organization's day-to-day business activities.

As an example, for all these above described scenarios, we can consider an operating system hardening process of a card server of a commercial bank that provides customers with all kind of credit and debit card services. Since the hardening is done by outsourced professionals and a down time is required, all servers information are available to outsourced personals that include customer card details as well as the server details such as the root password and also because of the downtime required, legitimate customer services will be down. So, in a case like this it will be a critical situation to the organization. Likewise, there are more drawbacks to a manual operating system hardening process.

To solve these types of difficulties and prevent intentional and unintentional human errors, we are going to implement a software platform (HardnBot) which automate the server operating system hardening process and which has the capability of detect failed compliances, classify them based on their criticality/severity levels and apply industry recommended best fixes for them via CIS benchmarks or via organizational requirements. HardnBot is also consists with an automated hardening function that harden a server according to classified (categorized) compliance issues and a rollback function that rollback to a point so that it will maintain and improve the productivity and integrity of the server.

In order to identify failed compliances, we will design a shell-based script that has the capability of executing and collecting all compliance failures in an unix system. After that, there will be a classification segment where the identified failed compliances will be classified (categorized) according to a severity level which measured by the impact criticality. Later these data will used to identify the level of hardening and as a risk factor parameter for the risk score prediction algorithm.

2.3.Technical objectives

Main Programming Language: C#

Almost all operating systems support C# language. C# is a general-purpose, multi-paradigm programming language including strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented, and component-oriented programming disciplines. And, C# contains with lots of supporting third party libraries. Therefore, C# will be used as the main programming language to implement this software.

IDE: Visual Studio

Microsoft Visual Studio is an integrated development environment from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. It mainly supports for C# development and contains lots of features that improve programming experience. And because of the inbuild functions and configurations, we are going to use Microsoft visual studio as our development IDE.

Modern UI Development: Bunifu DLL

Bunifu framework is recognized for innovative interface development in C# and C++ environments. Bunifu includes a range of colors and animated controls. It also includes scripts that allows to apply animated effects into the interfaces. Therefore, we will use this framework in designing our interfaces.

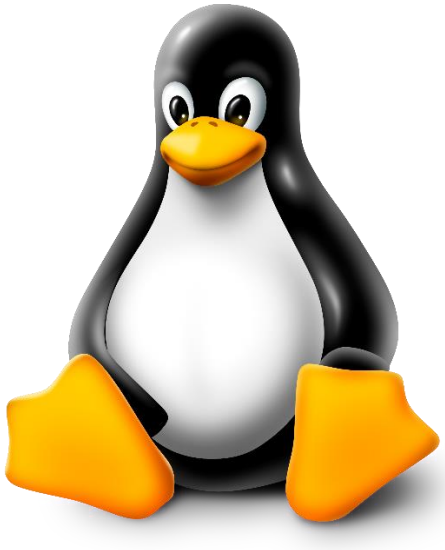
Virtual Environment: VMware Workstation

VMware Workstation is the industry standard for running multiple operating systems as virtual machines (VMs) on a single Linux or Windows PC. As the need we have to run several server-based operating systems to test our product, we will use VMware workstation for run multiple operating systems and observe the product outcomes.

Virtual Environment: Oracle VM VirtualBox

Oracle VM VirtualBox is a free and open-source hosted hypervisor for x86 visualization. In some cases where we have any difficulties to work with VMware, we will consider using Oracle VM VirtualBox as an alternative.

Operating System: Linux



Since our project is targeting linux based systems, we will implement hardening for a series of linux based operating systems as listed below.

{RedHat Enterprise Linux 7, RedHat Enterprise Linux 8, CentOS 7}

Operating System: Windows

Microsoft Windows operating systems are the most popular and wide used operating systems in the world. We will use Microsoft windows operating systems to run our IDEs and implement our software. Below listed versions will be used.

{Windows 8.1, Windows 7, Windows 10 version 1903}

Programming Language: Python

Python is an interpreted, high-level, general-purpose programming language. In our research, python will be mainly used for data training and machine learning purposes. Python includes a lot of mathematical libraries and data manipulation libraries which will be useful for our data training processes. We use following inbuild and third-party libraries. Also, we will use latest stable python version, at the time of developing (currently 3.7.4).

{Numpy, Scipy, Scikit-learn, Theano, TensorFlow, Keras, PyTorch, Pandas, Matplotlib, NLPs}

Automation Tool: Ansible

Ansible is an open-source software provisioning, configuration management, and application-deployment tool. It runs on many Unix-like systems and can configure both Unix-like systems as well as Microsoft Windows. It includes its own declarative language to describe system configuration. We will use ansible to achieve our expected automation hardening process.

Live Coding platform: Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows to create and share documents that contain live code, equations, visualizations and narrative text. Using Jupyter Notebook we can perform data cleaning and transformation, numerical simulation, statistical modeling, data visualization and all machine learning tasks. To use Jupyter notebook, we will use following techniques.

{Anaconda distribution of Jupyter, Hard installed versions of Jupyter}

GPU Based Machine Learning libraries: CUDA Toolkit

The NVIDIA CUDA Toolkit provides a development environment for creating high performance GPU-accelerated applications. The toolkit includes GPU-accelerated libraries, debugging and optimization tools, a C/C++ compiler and a runtime library to deploy our application. CUDA will mainly be used for performing text recognition and for natural language processing tasks.

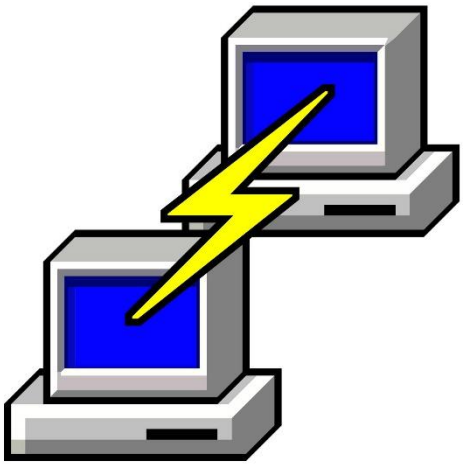
System monitoring tool: Nagios

Nagios, also known as Nagios Core, is a free and open source computer-software application that monitors systems, networks and infrastructure. Nagios offers monitoring and alerting services for servers, switches, applications and services. This tool will be used to monitor our servers for any misbehaviors after the hardening.

Script Language: Shell

Shell script A shell script is a list of commands (a program) designed to be run by the unix shell. We will use shell script for scanning purposes and other types of command executions in linux servers.

Terminal emulator: Putty



PuTTY is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. We will use putty for testing purposes and to create connections between servers.

Hardware: External GPU installed PC

A system with an externally installed GPU will be used for machine learning purposes.

3. Research Methodology

1. Retrieve failed operating system compliances and classify.

Classification is done by a trained machine learning model. Machine learning model will be selected after thoroughly test and analyze a set of classification models. Modifications and combinations may perform to the model in order to achieve the best accuracy levels. Supervised machine learning techniques and algorithms will be used to train our datasets.

Following tools and technologies will used to train the model,

1. Python 3 – programming language that contains required libraries to perform machine learning tasks.
2. Jupyter Notebook – tool that use for maintaining live python code and visualize the behavior of the data sets and algorithms.
3. TensorFlow – free and open-source software library for dataflow and differentiable programming across a range of task.
4. Anaconda – a free and open-source distribution of Python programming language for scientific computing and machine learning.

4. Test data & analysis

HardnBot's scanning procedures are done by executing scripts against a server and found compliance failures are classify using a trained machine learning classification models in help with text processing models. We will refer previous researches on vulnerability classification and text classification to gain required knowledge to train the datasets with relevant parameters. Online available genuine data sets will be used to train models. Cloud based python servers maybe used to test and retrain our models.

CIS (Center for Information Security) is a well-recognized and well reputed organization on information security and they have published many accurate and reliable reports on operating system compliances. In their website, there are many benchmark tests that they had published for many operating systems and application packages. We will refer the required and relevant reports and will gain required data to improve, test and optimize the datasets and algorithms that are going to use for model training.

We will use VMware workstation or Oracle VM Virtual Box to setup several virtual unix based server operating systems (CentOS, RHEL) and then we will test the scanning function and trained machine learning models for accuracy.

5. Anticipated benefits

HardnBot has a great commercial value because of the functionalities that it provides for users as well as organizations. Our primary goal is to automate an entire server hardening process and with the unique functionalities, HardnBot's user will gain following main benefits.

- **Fully in-depth server scanning specifically for OS compliances**
  With this functionality, HardnBot can scan thoroughly to find any compliance failures and any configuration errors.

- **Failed compliance classification and summarize**
  After collecting data from the scan, HardnBot's machine learning algorithms will provide a detailed list of compliances failures and misconfigurations with their criticality level. By using these, HardnBot will provides user with a detail percentage level of severity existence (For example: 10% Critical, 20% High, 30% Medium, 40% Low). With this information server custodians can get an overview idea about the severity level of the server.

- **Risk Score**
  By this functionality, an overall detailed risk score will be displayed to the custodian/user so that the organization can get a rough idea about server's possibility of compromise, likelihood and loss.

- **Intelligent Hardening**
  HardnBot' s hardening function is a unique function because it will harden the system for an acceptance level as required by the organization. Obviously 0% risk acceptance is not possible but in this function it's algorithms will use pre-classified compliance failure data and harden with respect to the severity levels.

- **Smart rollback functions**
  Using these functions, HardnBot will monitor applied fixes on the run as well as the server behavior parallelly and identify any misbehaviors and go back to that point where the misbehavior took place and rollback to the default or previously backed-up settings.

Besides those benefits, users will not require any down time to perform hardening because HardnBot's capability of performing stepwise hardening, stepwise error checking, stepwise backup and stepwise rollback. So, each fix will be monitored and having that, these functions will help to improve security operation center's (SOC) productivity and accuracy.

6. Project Schedule

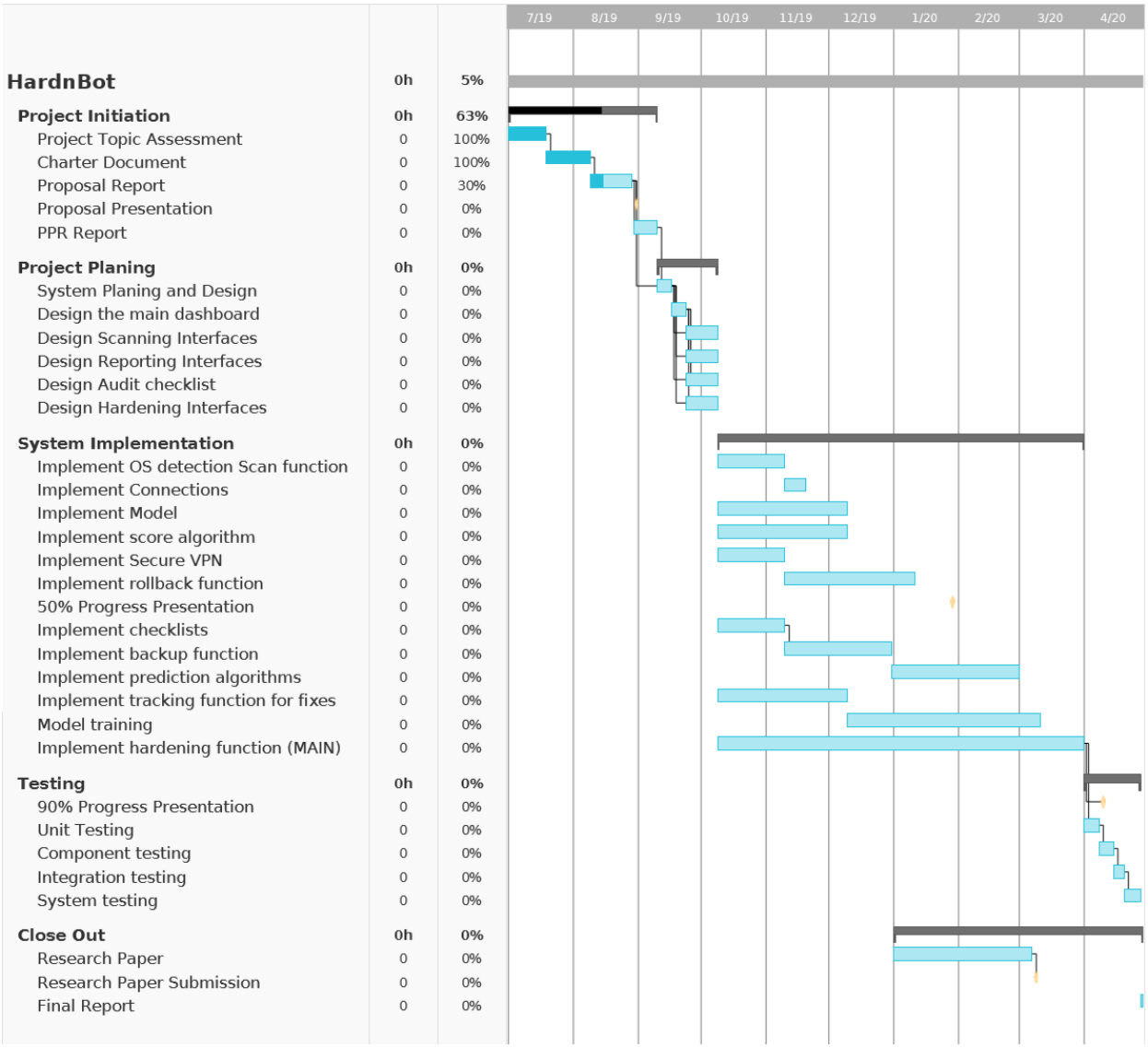
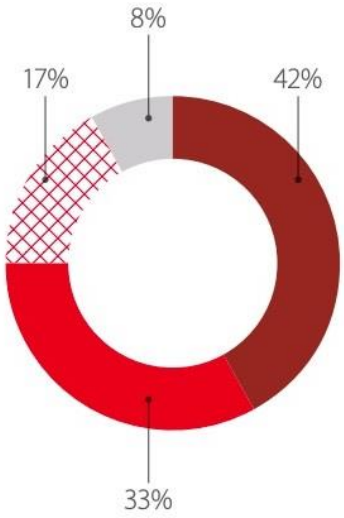


Figure 1: Project Schedule

7. Specified deliverables

- A well formatted compliance and misconfigurations list that classified according to severity levels.

- Total percentage value of compliance and misconfigurations count



- A software that will require very less human interaction

8. References

[1] Y. T.-A. Ratsameetip Wita, "Vulnerability Profile for Linux," 25 April 2005. [Online]. Available: https://ieeexplore.ieee.org/document/1423610. [Accessed August 2019].

[2] H. L. M. L. Q. Z. C. T. Bo Shuai, "Automatic Classification for Vulnerability Based on Machine Learning," 27 January 2014. [Online]. Available: https://ieeexplore.ieee.org/document/6720316. [Accessed 2019].

[3] J. K. K. S. Vijaya MS, "Password Strength Prediction using Supervised Machine Learning Techniques," 12 January 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5376606. [Accessed 2019].

[4] Y. Z. Q. W. X. Z. Kai Liu, "Vulnerability Severity Prediction With Deep Neural Network," 19 August 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8802851. [Accessed 2019].