HardnBot

INTELLIGENT SERVER HARDENING SOFTWARE

Project ID: **19_20-J01**

Thesis

R.M.B.B Rathnayake	IT16054400
G.G.L Anjula	IT16022416
W.M.K.M.W Wijekoon	IT16022416
Aruna Shan	IT16054400

B.Sc. (Hons) Degree in Information Technology

Sri Lankan Institute of Information Technology Sri Lanka

13th May 2020

HardnBot

INTELLIGENT SERVER HARDENING SOFTWARE

Project ID: **19_20-J01**

Thesis

Supervisor:

Mr. Amila Senarathne

B.Sc. (Hons) Degree in Information Technology

Sri Lankan Institute of Information Technology Sri Lanka

13h May 2020

DECLARATION

We declare that this is my work and this Preliminary Progress Review (PPR) report does not incorporate without acknowledgment 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 acknowledgment is made in the text.

R.M.B.B Rathnayake IT16054400	G.G.L Anjula IT16022416	W.M.K.M.W Wijekoon IT16022416	R.M.B.B Rathnayake IT16054400

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

Supervisor

…….

Mr. Amila Senarathne

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF CHARTS

1. ABSTRACT

Server Hardening is one of the most important tasks to be handled on servers. Server hardening, which is also referred to as operating system hardening, is the process of making the server stronger and more resistant to security issues. Server hardening is an inexpensive and simple task to improve the overall operating system for maximum performance and to reduce expensive failures. Hardening is a Process requires many steps, all of which are critical to the success of the hardening system. The more steps a user follows, the safer and more resilient the system will be. Servers can vary, depending on their usages. As an example, an email server is used to send and store emails over the corporate network. That means each server has its unique operations [1] and to serve the most secure/reliable service to their clients, servers should have proper compliances configured based on acceptable policies. Server hardening is the approach of applying the above-mentioned compliances to a server. 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. Using strong passwords, establish a password lockout policy, establish a data encryption mechanism are sample tasks that are just the tip of the iceberg as far as server hardening is concerned. Many more tasks must be completed, and each is more complex. For proper execution and maximum protection, professional assistance from an expert is needed. The 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. In this paper, we consider developing an automated server hardening functionality to minimize the system administrator's work with the Ansible orchestration tool. In this realized scenario, more than 10 Linux servers (Red hat and CentOS) were utilized to test this developed functionality. To bring the introduced functionality closer to system administrators, the software was created and thoroughly tested.

Keywords – Server Hardening, Automation, Ansible, Linux

2. INTRODUCTION

2.1. Definitions, Acronyms, and Abbreviations

CIS	Center for Internet Security
GUI	Graphical User Interface
OS	Operating System
PCI DSS	Payment Card Industry Data Security Standard
NIST	National Institute of Standards and Technology
DAC	Discretionary Access Control
MAC	Mandatory Access Control
VPN	Virtual Private Network

2.2. Overview

Massive amounts of data are created daily across the planet. By 2021, the annual global Internet Protocol (IP) traffic is predicted to reach 3.3 zettabytes [2]. To match this huge data environment, the data center industry is anticipating unprecedented growth. Data is the most precious asset in data centers. Data centers require abilities to ensure data service works properly; many technologies are used in data centers to achieve this goal. Data centers are supported to run 24/7/365 without interruption. Planned or unplanned downtime can cause business users serious damage.

Most data centers include Linux servers, Ubuntu Server, Red Hat Enterprise Linux, and CentOS as their base operating system. A data center includes about 170+ live servers and it's very difficult to do the operating system hardening manually.

There are no fully automated hardening platforms implemented yet and, and 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, databases, application and other services because a single mistake can affect the whole production line which the server is in. Even though many scanning tools have the capability to scan a server and identify compliance failures along with the solutions, the solutions are going to apply with human interaction. So, the probability of mistake occurrence is higher. And manual hardening process consumes much more resources such as time, human, cost likewise. As a solution to the lack of resources, organizations are tending to consider 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 get exposed via outsourced professionals intentionally or unintentionally and leave the server in a critical position of been compromised. And there is a compulsory requirement of the root (administrator) access to the server 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 the 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.

To solve these types of difficulties and prevent intentional and unintentional human errors, we 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 also consists of an automated hardening function that harden a server according to classified (categorized) compliance issues, a rollback function that rollback if any abnormal behavior is detected and an overall risk score prediction mechanism that will help with management tasks in the organization.

To identify failed compliances, we designed a shell-based script that has the capability of executing and collecting all compliance failures in a UNIX system. This script is capable of search through a server and scan for any compliance failures and gets the output with the compliance id and the status. The compliance id is a unique id given by the center of internet security (CIS) for operating system compliances [3]. After that, there will be a classification process where the identified compliance issues will be classified according to severity level. Later these data will be used to identify the level of hardening and as a risk factor parameter for the risk score prediction algorithm.

In this paper, we are going to discuss the theoretical approach and comparisons regarding previous researches, experiments we conduct and obtained results based on those experiments.

2.3. Project scope

Various operating systems are used in server systems; however this research will focus on Linux Operating System, CentOS 6, 7 is selected as the operating system for this project. The basic principles, security guidelines, and best practices may apply to other operating systems as well, however, this project will only focus on the Linux based Operating Systems. The security guidelines, configuration, and settings explored by this research project may address Level 1 configuration profiles that are defined by the CIS Benchmark of a Linux Server.

Items in this Level 1 intend to:

- Be practical and prudent;
- Provide a clear security benefit; and
- Not inhibit the utility of the technology beyond acceptable means [4].

This profile is intended for servers. The project will specifically be focused on the more widely used Linux based Services, not including any custom and 3rd party applications. SSH is the most commonly used tool for remote administration and management of Linux Servers. For the sake of this project, we will explore the basic installation and configuration

of OSSEC. This project will also explore and guide on checking what services are running on the server and turning off any unwanted services. This project will not address the security and configurations of all the tools and services available for Linux operating systems and or network security. This project will only cover Linux operating systems and the most common services on Linux Servers. This study alone will not make a Linux server completely secure from attacks or vulnerabilities; however, it will try to point out common settings and configurations that will harden the server security.

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

Under this document, following components are described.

a) Scan for compliance failures.
b) Classify them using a pre-classified dataset into severity levels as 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.

2.4. Background & Literature Survey

According to some previous researches done, we found researches that we can theoretically review their approach. These researchers research about various techniques to scan a server for compliances and perform hardening of servers and systems.

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, and 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 slowdown 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

[5].

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.

A. Baith Mohamed M. of Computer Engineering Department, College of Engineering & Technology, Arab Academy for Science & Technology Alexandria, Egypt published a research paper in 2001 on "An Effective Modified Security Auditing Tool (SAT)" which is a research about software tool to audit security configurations of a system.

In this research they have explained how to identify an exploitable vulnerability of an operating system via a security audit. This tool gathers much information from a remote hosts and network services such as ftp, NFS and according to those gathered information it will check for any security flaws, misconfigurations and other poor policy implementations that will put data at risk. As solutions, this tool can either report on this output data or it can use a rule-based system to investigate any potential security problems. However, according to this research, their main function of this tool is to iterate future data collection of secondary hosts using the initial data collection and user configurations for the next audit process. Furthermore, this tool can also analyze a complicated network and make practically informed decisions about the security level of the systems involves [6].

Kuo Zhao, Qiang Li, Jian Kang, Dapeng Jiang, and Liang Hu of Department of Computer Science and Technology, Jilin University, Changchun 130012, China published a research paper in 2007 on "Designing and Implementing of Secure Auditing System in Linux Kernel" which is also a tool to audit the system kernel of a Linux based system.

This research is about a tool to audit the kernel in a unix based system. Although there is a log collection mechanism in unix based systems, they are only based on application-level. A typical example for such subsystem is the "syslogd" daemon. It mainly receives important information of restricted services and process according to the configuration files. However, in this research, there main goal is to go beyond the typical user-state auditing and provide with a more detailed security audit result which contains both name of the system calls and related object of that. Since the current log files can be accessed, it will be a security issue and, in this paper, they also discuss about the security of all audit logs of this kernel auditing component as well. Later system administrators can view a completely in-depth detailed kernel state and user state logs for taking decisions for the system [7].

ling Liul, Xiaoni Wang, Dongliang liao, and Chen Wang from Engineering teach and practice training center, Tianjing Polytechnic University, Tianjin, China Key Laboratory of Beijing Network Technology, and Beihang University, Beijing, China altogether published a research paper in 2012 on "Research and Design of Security Audit System for Compliance" which explains the concept of compliance audit, and then proposed a log-based network security audit system for compliance.

In this over-all research, they first explain the whole concept of compliance audit and then they introduce the system architecture and components of a log-based network security audit system for compliance. In the system architecture, they have main five components which are,

1. Log analyzer:
   To analyze log information and send the alarm result to correlation analysis engine.
2. Correlation analysis engine:

This uses the audit analysis method based on data mining to give a countless suggestion analysis on the network security alarm events of different sources, different time, different levels so that they could dig out the real security events, find hidden complex attacks, Identify real security threats, and finally generate the system security alarm information.

3. Query statisticser:
   Query statisticser allows users to set the conditions for their own needs to view the details stored in the database relating to the event including the log information, alarm information, policy information and other system settings information.
4. Compliance manager:
   Compliance manager can provide the policy templates for the audit terminals to generate policy configuration information to check compliance posture of the audited system according to their audit trails and ensure that they follow all policies required by an external or internal regulation.
5. Management Console:
   It provides a convenient, intuitive management interface and visually displays the event analysis results and network security status; users could use it to complete software system configuration, security policy definition and information inquiry, the management console is responsible to receive the audit data from the agents, store the data into the database and release the audit policies to the audit agents.

In the end of this paper they mentioned the used technologies and other detailed design of key components [8].

Frederick Yip, Alfred Ka Yiu Wong, Nandan Parameswaran, Pradeep Ray from School of Computer Science and Engineering School of Information Systems and Technology Management University of New South Wales, Sydney Australia published a research paper in 2007 on “Robust and Adaptive Semantic-Based Compliance Auditing” which contains a deep research about compliance management and compliance auditing.

In this research, researchers state that the compliance auditing is a child process of compliance management, where compliance rules and policies are individually check against the organization to determine the level of compliance achieved by the organization. In this research paper, they deeply describe about all Information Technology required policies and standards such as,

1. Sarbanes Oxley (SOX) Act - Government Regulations and Information Security Standards
2. HIPAA – Health Insurance Portability and Accountability Act - Government Regulations and Information Security Standards
3. ISO/IEC 17799:2005 - Information Security Standards
4. Control Objectives for Information and related Technology (CobiT) - Information Security Standards

So, this is an overall big picture of the idea of compliance. However, in our research, we are going to limit this scope form organization to a Linux system [9].

Frederick Yip from School of Computer Science & Engineering University of New South Wales Sydney, Australia, Pradeep Ray from School of Information Systems & Technology Management University of New South Wales Sydney, Australia, and Nandan Parameswaran School of Computer Science & Engineering University of New South Wales Sydney, Australia, altogether published a research paper in 2006 on "Enforcing Business Rules and Information Security Policies through Compliance Audits".

In this paper they present XISSF, an extensible information security specification format that acts as a compliance audit mechanism for enforcing business rules and information security policies [10].

Uttam Thakore from Department of Computer Science University of Illinois at Urbana-Champaign Urbana, IL 61801, Rohit Ranchal from IBM Cambridge, MA and Yi-Hsiu Wei and Harigovind V. Ramasamy from IBM Austin, TX 78758 altogether published a research paper in 2019 on "Combining Learning and Model-Based Reasoning to Reduce Uncertainties in Cloud Security and Compliance Auditing" which is a research based on cloud server compliance audit.

In this research they propose a hybrid approaches, in which formal, model-based approaches are combined with machine learning techniques to reason about evidence and historical audit data, are necessary to address any human mistakes are errors in the cloud environment [11].

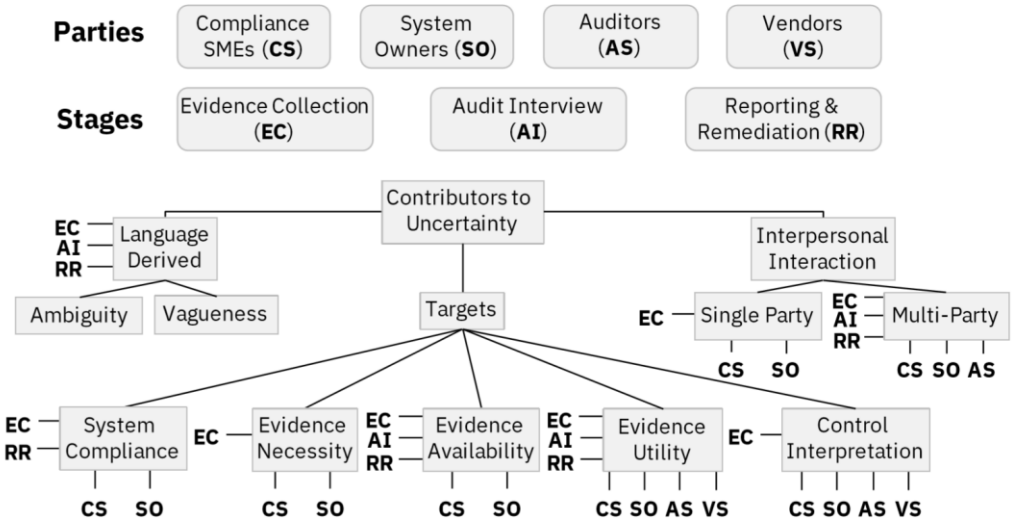


Figure 1 : System plan of cloud compliance audit

Rafael Enrique Rodriguez-Rodriguez, Andrés Felipe Quevedo Vega, Andres Felipe Sanchez, Alexandra López, and Jaime Fernando Pérez of Facultad de Ingeniería, Universidad Católica de Colombia, Bogotá D.C, Colombia published a research paper in 2018 on "Design of an Automation Model for Taking Documentary Evidence of Compliance Tests of the IT Audit" which state a method for taking documentary evidence of compliance test of the IT audit.

In this paper the researchers focused on designing a functional model for crating and executing of evidence gathering of compliance audits performed by the organization. This is basically a reporting function for compliance auditing. And in our research, we also use some reporting methodologies to collect evidence of the compliance audit performed.

In this research they categorized the audit methodologies as shown in the below table.

Table 1 : categorized the audit methodologies

Feature	ISO 27001	Megerit V3	ISO 31000
Risks Management	Identify threats Guidelines for risk analysis Analysis of risk management in ISMS Has a risk management approach	Promotes the identification of threats Systematic method for risk analysis Discover and plan risk Performs qualitative and quantitative risk analysis	Generates a stage to identify threats Poses rules for risks analysis
Senior Management Consideration	Its application demystifies that the security of information is a technical issue	Consider those responsible for the organization Prepare the organization for the application of audits	Its application helps management to control risks
Application of controls	Set default controls for its application It divides the controls by classes such as: logical security, audit trails, integrity, continuity, and physical security.	Plan the implementation of controls Categorizes controls to manage risks	Establish guidelines for the establishment of controls
Asset Management	Manage assets through inventory Establishes the owners of each asset	Guides the classification of assets Divide the assets into several groups to identify the risks more clearly	Make the classification of assets

	Manage guidelines for the acceptable use of assets Establish asset classification guidelines	Manage the description of assets to identify risks	
Type of companies	It is aimed at all types of companies	Public Organizations Non-profit organizations	It is aimed at all types of companies

[9].

A research conducted by Prowse D. In 2010 on "OS Hardening and Virtualization" describes how to perform OS hardening using the OS security audit method. To perform OS hardening, as a first step they perform vulnerability assessment over windows, then perform the security audit and fully analyze system logs. Further, they explained how important it is to perform periodic security audit over an OS to track vulnerabilities and take relevant countermeasures. As a benefit of doing OS hardening, they pointed out how it helps to reduce the risk, improve the performance, eliminates vulnerable entry points, and mitigate security risks. As this paper states, OS hardening can be done using techniques such as program clean-up, service packs, patch management, group policies, see templates, and configuration baselines. Further, for more user-friendliness, operating systems like windows provide facilities to prioritize vulnerabilities as high, medium, and low. To strengthen the security of OS, they discussed manual technology as well as semi-automated terms under manual techniques. Preparing checklist for security parameters, reviewing security configuration aspects, manually set security configuration, and explaining OS as per configuration parameter included. In a semi-automated way, they are using scripts for audits such as .bat, .ps, set security configuration using a script, exploiting OS scripted payload. In the discussion, they showed how important it is to perform periodic audits to identify security issues, prioritize those, and treat to mitigate risk over operating systems [12].

Amit Nepal published a research paper in 2013 on "Linux Server & Hardening Security" highlighting the basic security configurations that should be performed to harden the security posture of a default Linux Operating System installation. Its main approach was to keep the server as simple as possible, the complexity of the server is reduced. Also, a server with fewer services running is less vulnerable. By exploring the default installation and removing unwanted applications and services, we reduce the vulnerabilities in the server. This research carried out in four phases. In the first phase, it performs some intrusion attempts like brute force attacks, footprinting, etc. and explores the system response. Also it explores the commonly installed services and applications and identifies what applications and services we might not need and remove them. In the second phase, it makes the changes to the default configurations; put some restrictions in place with the common services and operating system. In the third phase, it configures the firewall

(IPTABLES) to restrict access to certain ports from certain IP Addresses, so that the services to the server are available only to those who need it. Finally, in the fourth phase, it installs and configures a Host Based Intrusion Detection and Prevention System (OSSEC), which will act as a proactive monitoring and intrusion preventing system [5]. According to the phases conducted by the above researcher, I followed the operating system (OS) footprinting and identified explored the OS. Secondly, the main common services and operating system were identified by following the CIS benchmark. Services were identified whether to be removed, installed, or put some restrictions.

A research conducted by Christine Bresnahan and Richard Blum In 2020 on "Understanding Basic Security" describes how to understand basic security and identify user types. Accounts enable multiple users to share a single computer without causing one another too much trouble. They also enable system administrators to track who is using system resources and, sometimes, who is doing things they shouldn't be doing. Account features help users use a computer and administrators administer it. Understanding these features is the basis for enabling you to manage accounts. Some account features help you identify accounts and the files and resources associated with them. Knowing how to use these features will help you track down account-related problems and manage the computer's users. Linux permits multiple users to access the computer simultaneously. Most often, this is done using remote access servers such as the Secure Shell (SSH); however, you can use Linux's virtual terminal (VT) feature to log in multiple times with a single keyboard and monitor. Sometimes, you might want to know who is using the computer. Linux is modeled after UNIX, which was designed as a multiuser OS. In principle, you can have thousands of accounts on a single UNIX (or Linux) computer. At least one user, though, needs extraordinary power to manage the features of the computer as a whole [6]. As said that root user privileges are needed to perform server hardening operations.

Douglas Santos and Jéferson Campos Nobre published a research paper in 2019 on "Vulnerability Identification on GNU/Linux Operating Systems through Case-Based Reasoning" highlighting automated rule-based tools are often used to support professionals with little experience in vulnerability identification activities. However, the utilization of rules sets up reliance on designers for the improvement of new standards just as to keep them updated. The inability to update rules can essentially bargain the integrity of vulnerability distinguishing proof outcomes. In this paper, inexperienced professionals are improved in conducting vulnerability identification activities by Case-Based Reasoning (CBR). The motivation behind utilizing CBR is to cause unpracticed experts to get comparable outcomes as experienced experts. Besides, the reliance on rule developers is lessened. A model was created thinking about the GNU/Linux framework to do an experimental evaluation. This assessment exhibited that the utilization of CBR improves the presentation of unpracticed experts as far as the number of identified vulnerabilities [7].

A research conducted by Mohan Krishnamurthy, Eric S. Seagren, Raven Alder, and Eli Faskha In 2020 on "Apache Web Server Hardening" describe that Apache is no different and can be negatively affected by any one of the following problems: poor application configuration; unsecured Web-based code; inherent apache security flaws; and foundational OS vulnerabilities. Apache has many default settings that require alteration

for secure operation. Nearly all configuration information for the Apache Web server exists within the httpd. conf file and associated Include files. Web developers are unquestionably more worried about business usefulness than the security of their code. Simply publishing confidential or potentially adverse information without authentication can afford attackers with resources for an attack. There are several means by which hackers can breach or damage an Apache system, such as a denial of service; buffer overflow attacks; attacks on vulnerable scripts; and URL manipulation. Code deficiencies can exist in OSs and lead to OS and application vulnerabilities. It is therefore imperative to fully patch newly deployed systems and remain current with all released functional and security patches. The Apache Web server is a powerful application through which one can deliver critical business functionality to customers. With this power comes the possibility of misuse and attack. To ensure that the Apache server is running securely, a series of steps to harden the Apache application must be followed and these are: preparing the OS for Apache Web server; acquiring, compiling, and installing the Apache Web server software; and configuring the httpd. conf file. According to the research poor Operating system configuration is identified automatically by conducting an audit according to the CIS benchmarks of the relevant operating system [8].

Doug White and Alan Rea published a research paper in 2009 on "Server hardening model development: A methodology-based approach to increased system security". This research forms a flawless model that combines information on tools, tactics, and procedures that system administrators can use to harden a server against compromise and attack [9]. Kyung Sung's research paper published in 2020 on "Analysis of Linux firewall based on Firewalld" and "Design and Implementation of Firewall Security Policies using Linux Iptables" research paper published by M. G. Mihalos, S. I. Nalmpantis, Kyriakos Ovaliadis describes A default deny all policy on connections ensures that any unconfigured network usage will be rejected [10],[11]. According to these researches, with a default accept policy the firewall will accept any packet that is not configured to be denied. It is easier to white list acceptable usage than to blacklist unacceptable usage, and Applying host-based firewalls or port filtering tools on end systems, with a default-deny rule that drops all traffic except those services and ports that are explicitly allowed.

A research conducted by Jonas Schneider, Nils Fleischhacker, Dominique Schröder and Michael Backes In 2020 on "Efficient Cryptographic Password Hardening Services from Partially Oblivious Commitments" propose a construction for password hardening services based on a novel cryptographic primitive called partially oblivious commitments, along with an efficient secure instantiation based on simple assumptions. The performance and storage evaluation of our prototype implementation shows that our protocol runs almost twice as fast as Pythia, while achieving a slightly relaxed security notion but relying on weaker assumptions [12]. In my research passwords are cryptographically ensured by the hashing algorithm is SHA-512. The SHA-512 algorithm provides much stronger hashing than MD5, thus providing additional protection to the system by increasing the level of effort for an attacker to successfully determine passwords.

Amith Raj MP, Ashok Kumar, Sahithya J Pai, Ashika Gopal published a research paper in 2016 on "Enhancing security of Docker using Linux hardening techniques" describes that Docker supports the Linux hardening capabilities and Linux Security Modules (LSM) with

AppArmor and SELinux for host system hardening. Docker interacts with the kernel security systems and LSMs. In this research work, the security depth of a popular open-source model based on containers and study on other security features and techniques to enhance the security of Docker is performed. Integrating virtualization, automated testing, Deployment tools, and security configurations management will enhance the security capability of Docker on-premise [13]. According to the above mentioned Linux security Modules, SELinux provides a Mandatory Access Control (MAC) system that greatly augments the default Discretionary Access Control (DAC) model. Under SELinux, every process and every object (files, sockets, pipes) on the system is assigned a security context, a label that includes detailed type information about the object. The kernel allows processes to access objects only if that access is explicitly allowed by the policy in effect [14].

The policy defines transitions so that a user can be allowed to run the software, but the software can run under a different context than the user's default. This automatically limits the damage that the software can do to files accessible by the calling user. The user does not need to take any action to gain this benefit. For an action to occur, both the traditional DAC permissions must be satisfied as well as the SELinux MAC rules. The action will not be allowed if either one of these models does not permit the action. In this way, SELinux rules can only make a system's permissions more restrictive and secure. SELinux requires a complex policy to allow all the actions required of a system under normal operation. Three such policies have been designed for use with RHEL7 and are included with the system: targeted, strict, and mls. These are described as follows:

- targeted: consists mostly of Type Enforcement (TE) rules, and a small number of Role-Based Access Control (RBAC) rules. Targeted restricts the actions of many types of programs, but leaves interactive users largely unaffected.
- strict: also uses TE and RBAC rules, but on more programs and more aggressively.
- mls: implements Multi-Level Security (MLS), which introduces even more kinds of labels (sensitivity and category) and rules that govern access based on these. [15]

A research conducted by Sonali Patra, N C Naveen, Omkar Prabhakar in 2016 on “An automated approach for mitigating server security issues” describes servers such as mail server, web servers, application server, etc.., store much sensitive information such as project details, media information, personal data, national security-related information, etc.., if such sensitive data gets into wrong hands. Business and the reputation of the organization will be damaged. Therefore, the need to automate security mechanisms to detect, prevent, and protect the server from the attackers. Security policies play an important role in network security and server security. An Automated Approach for Mitigating Server Security Issues proposes a framework that would ease the work of an administrator. It focuses on designing an automated tool that would perform an audit of the servers and check if it is compliant with all the prescribed security policies. As there are multiple platforms upon which the servers run, the tool is designed to adapt to a heterogeneous environment.

In this paper, an automated security tool as in Fig.1 is proposed that would ease the job of an administrator to check if the server complies with all the security policies of the organization. The security policies which were tested included the following,

1. Checking if the Windows server is running the approved up-to-date anti-malware solution.
2. Checking if the approved antivirus shield is seen in the taskbar for a system running Windows List the version of the Anti-Malware running on the systems for Linux and Windows operating system.
3. Check if any personal removable media present for both Linux and Windows Server, if so list their names and timestamp of the device insertion.
4. List all the security patches applied to the Windows operating system.
5. List the installed versions of software running on the Windows System.
6. Check if the event logs have been enabled or disabled.

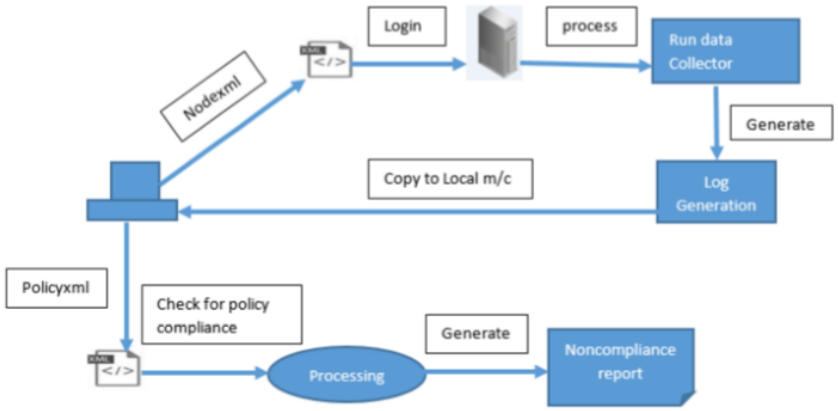


Figure 2: Proposed Security Compliance Tool

The following basic security steps which have defied in this research are applied in my research too,

- Every organization should have their security policies defined.
- There should be an apt network protection mechanism such as firewall technology, anti-virus technology, Intrusion Detection System (IDS), Virtual Private Network (VPN), and data encryption technology.
- Use of automated tools that would keep track of the server's security policies and their compliance.
- Use of secure administration and maintenance processes, which includes application of patches and upgrades, monitoring of all the logs, backing up of the server data, and operating system.
- Installation and configuration of a secure operating system and software in the server.
- Use of vulnerability scanner to perform security testing.
- Malware detection, mainly during the insertion of infected devices through USB.
- Configuring access control. [16]

Ayei Ibor, Julius Obidinnu published a research paper in 2019 on “System Hardening Architecture for Safer Access to Critical Business Data” highlighting system hardening and system hardening architecture. This will be a guide to system administrators for

implementing multi-layers of in-depth protective mechanism over the stored data." System hardening is a strategy to increase the security of the system, which applied many different security measures to different layers to detect vulnerabilities of the system layers and defeat them before it damages the system. The proactive protective mechanism of system hardening architecture applied to the host, application, operating system, user, and the physical layers. This system hardening security strategy can be implemented to the organization, to decrease the breaches and also safer access data.

Here they develop a system hardening architecture. Create security functions and combine them independently with the relevant module. These functions are applied independently and separately, however here they try to implement their security mechanism levels and use those mechanisms to relevant places which the relevant module located in the system. Because of that mechanisms if an attacker breaks the level one security he had to break several security mechanism levels to access the relevant data. Because of the higher number of security levels mechanisms, attackers had spent much more time and also wasting resources they may give up and find some other efficient way to do their malicious activities [17].

A research conducted by Hongjuan Li, Yuqing Lan in 2010 on "A Design of Trusted Operating System Based on Linux" demonstrates that a trusted operating system can help solve the information security problems. A design process of a trusted operating system based on Linux was developed by the China Standard Software Company (CS2C), and it's Still researching furthermore, Double-key authentication and architecture provide in this project. This operating system selects the method to implement Designing Method. The benefit of a trusted operating system is to offer users a trusted computing environment [18].

<table>
<tr><td rowspan="3">User layer</td><td>shell layer</td><td colspan="3">shell program</td></tr>
<tr><td rowspan="2">utility layer</td><td rowspan="2">general applications</td><td colspan="2">trusted processes</td></tr>
<tr><td>expansion of the original procedures</td><td>new programs</td></tr>
<tr><td rowspan="2">Secure core layer</td><td>system call layer</td><td>security-related system calls</td><td>security-unrelated system calls</td><td>new system calls</td></tr>
<tr><td>core layer</td><td>security-related entities</td><td>security-unrelated entities</td><td>new entities</td></tr>
<tr><td>Hardware layer</td><td colspan="4">Hardware interface</td></tr>
</table>

Figure 3: Design process of trusted OS based on Linux

The architecture of trusted operating system based Linux shown three layers of architecture, they are hardware layer in the bottom, secure core layer in middle, and application layer in the top.

The research on "Automated Provisioning of Application in IAAS Cloud using Ansible Configuration Management" mainly focuses on Cloud. Cloud has become a powerful

technology in at present's information technology background and the need arises to stand with the evolving request of clients. This paper focuses on the automation of customer application right from environment provisioning to application deployment. In this paper, the whole architecture of automated application deployment assembled using amazon as IAAS provider and Ansible as the orchestration engine [19]. This Ansible engine is used as the configuration management tool in my research.

When considering the manual server hardening approach server custodian/owner has responsibilities to manage information level risk. Therefore, accomplish that requirement he/she has to take a backup of the server before it is going to harden. Not only that but also server custodian/owner must ensure that server is working properly after the hardening process, if not that could be an operation level issue and he/she needs to fix it immediately. Let's assume he/she couldn't able to bring server to its normal working behavior, then the only answer is rollback the server to its previous stable state using backup which is previously taken. In the manual approach, all of these tasks take a considerable amount of time and active user interaction as well.

Intelligent Server Hardening Software (HardnBot) is a software that has the capability to automatically detect poor/non-compliant configurations in Linux servers which has CentOS 7 as an operating system and applying industry recommended fixes for them.

Under this final report, the following research components are described:

- Automated backup function
- Intelligence rollback function
- SSH connection to servers

Software is divided into fourteen main parts and all of them are important to the final outcome of this project. This report covers three parts of that. Through this document, these components will be described with validation.

Faraz Shakib, an employee of a Calsoft Private Limited, India and Zoheb Shivani of Pune Institute of Computer Technology, Pune published a research paper in 2006 on "Snapshot service Interface (SSI), a generic snapshot assisted backup framework for Linux". In there they were talking about an online backup solution called "SSI" for the Linux platform while talking quantitative measure on performance hits incurred due to using SSI framework in lieu of using traditional backup approaches. Mainly SSI is aimed at enterprise-level high-end critical servers (database servers and mail servers) which are providing 24x7 availability. Because that kind of servers doesn't have any downtime periods, especially when taking the backup and maintenance.

In SSI there are 9 major steps covering the whole process of taking a system backup.

1) The SSI initiates from requesting SSI based snapshot-assisted backup by the backup application.
2) SSI asks all the registered business applications to get consistent and waits until all of them respond since they need to maintain an availability of their services. Timeout mechanism is there to prevent unwanted SSI waiting.
3) Make business application's consistent by flushing their buffers.

4) Until the snapshot operation is performed, the application is paused and another timeout mechanism is also available within the business applications to prevent wait forever in case SSI fails to respond.
5) Write to disk is over and consistent file system is available for use.
6) The snapshot provider is then requested to take the snapshot since the file system is consistence. Then snapshot is take as a volume level snapshot and consistent backup can be extract from it.
7) The status of the snapshot (succeed or not) returns by the snapshot provider.
8) SSI in turn the status to the backup application.
9) In this step all the business applications are signaled to continue when the file system writes are resumed.

The major benefit of this SSI framework is that business applications can continuously write data while their snapshot assisted backup is taken [13].

*Since the HardnBot is not an online-based tool, the above approach deviates from our process of taking a backup automatically to the external block device. In the other hand, this method is most suitable for critical servers.

Alireza Tajary and Hamid R. Zarandi of Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Teheran, Iran published a research paper in 2016 on “An Efficient Soft Error Detection in Multicore Processors Running Server Applications”. According to this research, there are three categories for tolerating transient faults in server processors named “redundant execution”, “anomaly detection” and “dynamic verification”. When considering the redundant execution-based method, it has a high potential to detect any fault, but it reduces the provided throughput of the processor significantly. The impact of the Anomaly detection-based methods on processor performance is very low, but it takes more time to detect faults. When comes to the dynamic verification method it has overhead on each core of the processor negatively and implies dedicated hardware to dynamically verify the execution of instructions.

The proposed method of this research consists of three modules,

1) Configuration manager module
2) Speculative faults detection module
3) Redundant execution-based faults module

Configuration manager module is capable of handling the new threads by changing the configuration of cores. Perform this task it has a table which stores the state of each core. There are four states

1) Free core.
2) Non-coupled busy core.
3) Coupled with right core.
4) Coupled with the left core. Before handle, a new thread, status of the relevant core or cores should be checked.

Enables fault detection mode for each thread which can be run in redundant execution mode is responsible for speculative fault detection module and for that it stores the requests from

L1 cache to the L2 cache. By considering the stored value of the L1 cache the proposed solution utilizes a speculation to detect the occurrence of the fault.

The redundant execution-based faults module contains one switch for each core and one comparator module between two adjacent cores. The above switch is used to select the data path between the L1 cache and pipeline. These switches can be configured by the configuration manager module to enable or disable coupling operation.

Using this overall throughput aware redundant execution-based soft error detection method the configuration manager determines the configuration in the arrival of a new thread. If it has not enough available resources one core will be assigned to the above-mentioned thread. Then speculative faults detection method is activated to detect faults. If it has enough available resources it uses redundant execution to run the thread. This proposed method results variate from 70% to 100% with respect to resource availability for redundant execution and speculative detection methods [14].

Olumuyiwa Ibidunmoye, Ewnetu Bayuh Lakew and Erik Elmroth of Department of Computing Science Umeå University, Sweden published a research paper in 2017 on "A Black-box Approach for Detecting Systems Anomalies in Virtualized Environments". In their research, they have been proposed a black-box framework which has an unsupervised prediction-based technique to detect anomalies automatically. This mechanism uses multi-dimensional resource behavior of datacenter nodes for detecting abnormal behaviors. After that, anomalous nodes across the data center which are related to abnormal behaviors, are ranking using a graph-theoretic technique.

Mainly the system is based on usage of computer, memory, disk IO and network resources. That information collected by distributed lightweight monitoring agents deployed in each node (VMs and PMs) of the virtual infrastructure and sampled at a fixed time interval. The Node Anomaly Detection (NAD) is an independent agent which is deployed in an individual node. The main purpose of this module is adaptively detecting time-points where the node faces abnormal resource usage. It doesn't matter whether it is temporary or extended period of time. To fulfil this target NAD uses three steps. 1) estimates a continuous temporal profile of normal behavior using historical data, 2) predict expected behavior in the immediate future using the profile estimate, 3) Calculate the difference between the forecast residual of the expectation and the baseline in multidimensional space. Then NAD triggers an alarm when an anomaly is detected.

The Global Anomaly Ranking (GAR) module is for cluster-wide observability and its objective is collecting all alarms which are triggering from NAD around distributed nodes at a regular time and rank them based on spatial dependencies or how one anomaly in one node is the effect on other anomalies which are observed within the same period of time. This can facilitate root because analysis and anomaly mitigation procedures against a large number of alarms generated from NAD [15].

Yogendra Kumar Jain and Sandip S. Patil of Samrat Ashok Tech. Institute, Vidisha, India published another research paper in 2009 on "Design and Implementation of Anomalies Detection System Using IP Gray Space Analysis". They have come up with a methodology which has three steps to detect external anomalous host with their scanning behaviors using

IP gray space analysis and to scanning foreign port used by the external anomalous host. IP gray space is a collection of unassigned IP addresses in a considering network. It means IPs which are not assigned to any active host.

Step 01: Identification of anomalous external host using IP gray space and relative uncertainty. In this step, they have been maintained an IP threshold range which includes all the active IPs in it. That kind of threshold setting is called as association rule generation for supervised learning. If the source IP address is coming from this active IP space the host related to that IP is considered as a normal user. If the IP is from gray space, then it should be an anomalous host. Active space can define by our own concerns, ex (192.168.54.1 to 192.168.54.254).

Step 02: Identification of category of the anomaly using dominant scanning port (DSP). In this step it's identifying 5 main categories of anomalies using their dominant scanning port (DSP). DSP is the foreign port and port service used by scanning flaws SF (h) of anomalous host for communication with an internal host.

Types of anomalies detected using DSI:

1) Bad Scanner-I: ICMP probes, TCP/UDP scanning activities, searching for well-known ports
2) Bad Scanner-II: TCP/UDP probes on a variety of ports, ICMP ping, TCP connection request on port 113
3) Bad Scanner-III: TCP/UDP probes, receive responses from some live inside hosts, TCP port 80 scanning, performing queries to an inside DNS server, sequential scanning on TCP port 445
4) Focused Hitters: Belongs to a small number of applications like SMTO, web and some targets the web services ports like 80 and 443
5) Mixed Intruders Anomaly: A new category, having hybrid behavior of normal and abnormal behaviors. Required checking Bad scanners and Focused hitters first.

Step 03: Determination of potential behavior of each anomaly using flaws ratio [16].

Ming-Jen Chen, Chia-Chun Shih, Chien-Huei Yang, Gene Hong and Yuan-Sun Chu of Department of Electrical Engineering, National Chung-Cheng University, Chia-Yi, Taiwan have published a research paper in 2011 on "Multi-Layered Monitoring Framework with Indices Relationship Analysis for Distributed Service Performance Evaluation". As they mention a monitoring framework enables system administrators to identify and exclude abnormal system performance behaviors. There are two monitoring frameworks called centralized monitoring framework and distributed monitoring framework. The proposed multilayer monitoring process provides in-depth performance analysis. This is done by three main processors named cross-layered performance, indices relationship analysis and performance state inference. Using this three in one method administrator can reduce personal cost and increase the reaction speed of anomaly performance.

Cross-layered performance indices relationship analysis.

In this approach, the relationship between the unit performance index (UPI) and abstract performance index (API) is identified. Apart from that, it helps administrators to derive APEP according to FPEP. The process of analysis has three phases, initial phase, learning phase, and inference phase. When it comes to the initial phase it's an administrator responsibility to address the performance issues of system services and define the unit and abstract performance indices (UPIs and APIs), define detection rules of anomaly performance behavior. In the learning phase relationship analysis of the cross-layered performance, indices are processed. Results of the initial phase and the learning phase can be used to pre prediction of new performance abnormal behaviors during the inference phase [17].

Zhe Wang, Jin Zeng, Tao Lv, Bin Shi and Bo Li published a research paper in 2016 on "A Remote Backup Approach for Virtual Machine Images". In there they were talking about virtual backup on cloud storage. When we are considering cloud computing, virtualization is playing a major role, because of hosting several applications and services in virtual machines (VM) which were hosted in cloud environments. Security becomes a prior requirement in virtualized applications. In this research, the main focused area is high availability issue in virtual machines. LiveRB (Live remote backup) is the proposed remote backup approach. The purpose of the Live RB is to save the running state of the VM in an online manner known as "Live Migration". This backup process will happen the background of the hosted cloud applications of the VM and is transparent to them. A virtual block device will be designed and will be used to cache I/O Operations in memory, in order to save the incremental virtual disk data.

LiveRB will be implemented on KVM virtualization platform in order to evaluate effectiveness and efficiency using a set of comprehensive experiments. These experiments are all related to Cloud Computing and the security issues that come along with this and the key points considered in order to have successful cloud computing are security, availability & fault tolerance. The commonly used solution to handle Fault Tolerance & High Availability is using snapshots or checkpoints that periodically record the states of the software for backup and rollback the cloud applications to the previous backup upstate. This procedure will be carried out when encountering Failures or Errors of the original system.

Most currently existing VMs stop the VM to take snapshots. Some VMs need to be shut down to take snapshots which affects the ability to provide the service/ result in abnormal cloud application behavior. Some VMs suspend the current process and save the current progress onto local disks to be transferred onto remote servers later which sometimes result in data loss if a hardware failure is encountered.

The above issues can be resolved using Live RB since it works by not stopping the VM to do the backup process. Results of this process indicate that Live RB can be used on a VM to do the backup task from VM onto a Remote Server with only a slight reduction in performance [18].

In this research, it described a method that used to back up a virtual machine, but when it comes to our research area we have to consider the live server. Therefore, no need of care about any virtual machines, but when we are talking about the remote backup approach

used in here, that was Live Remote Back up, so we can consider about this technique when we are dealing with our problem.

L. Farinetti and P. L. Montessoro published a research paper in 1993 which named "An Adaptive Technique for Dynamic Rollback in Concurrent Event-Driven Fault Simulation". In here it is discussing automatic rollback based on an adaptive mechanism which is including advanced network/system status recording system. Time can be any time that means before changing of a system or after changing a system this status recording can be applied. The main feature of this research is the user can define the rate for maximum acceptable level for rollback. This approach takes the average time to a minimum level that means a very short time of the rollback process.

To come up with the proposed technique, researchers were used existing methods such as incremental backups, journal files, checkpoints, rollback, and roll forward which were found on different applications, different operating systems as well as different databases. Mainly the status of the network/system is recorded on the disk and run for the negative time period to analyze the previous status. If needed user can run for a positive time period as well. Those time periods are for comparison with the current status of the network/system. To make it happens above approaches need some fine tunes as well [19].

According to rollback techniques used by those researchers, we identified some techniques and requirement that should be in our system too. For this part of the software, it is necessary to detect abnormal behaviors of the server after the hardening process is done. For that, we need to record system status after the hardening process. Then compare with the previous status that means system status before the hardening process, but in our approach, there are pre-defined models for comparison with current system status. Apart from that, the rollback mechanism is going to adopt from this research. What are the things?

Ning Lu and Yongmin Zhao published a research paper in 2018 which named "Research and Implementation of Data Storage Backup". In this research, researchers were tried to discuss features of a reliable and secure backup and types of backup. With the use of applications which were dependent on big data, the usage of data storage backups became more important. Therefore, the methods used to backup should be more flexible and can be able to ensure of security and reliability of backup contents and also backup and restore should be in a convenient manner. There are several backup methods such as data backup, system backup, application backup etc. The backup contents are guaranteed to be confidential, complete and effective.

There are several specific performances in a backup,

i. Backup should be upgradable, capacity expansion
ii. Management without affecting other application in the system
iii. Implement a backup storage system combining SAN (storage area networks) and NAS (network attached storage) storage networks.
iv. Provide several backup methods such as data backup, system backup, application backup,

v. Backup contents should be secure and restore operations should be done in a convenient manner.

System backup

Refers to the backup of the end-point operating system, server operating system and other systems. In here core files and system's registry are backed up as a data. In a matter of system crash or operation mistaken the backup can be restored to the previous state.

Virtual tape library

Virtual tape library (VTL) considered as a world's leading modern technology to create a backup system. It can rapidly backup and rapidly recover a system that we want to backup. The main feature is no manual intervention of this technology. VTL storage media is a SATA disk and its data transfer rate is 150MS/s. That means approximately it takes 10 seconds for transferring 1.5GB data to the backup storage [20].

In our research, one of the main goals is to reduce the overall hardening time. For achieving that task, we should have to minimize the overall backup time to some acceptable level using speed backup mechanism. In this point, we are going to use the technique which is described in this research known as a virtual tape library. If we can adopt this mechanism overall hardening time will reduce averagely by 8 hours to 4 hours.

Teruaki Sakata, Teppei Hirotsu, Hiromichi Yamada and Takeshi Kataoka published a research paper in 2007 which named "A Cost-effective Dependable Microcontroller Architecture with Instruction-level Rollback for Soft Error Recovery". This tool is developed to detect soft errors using electronic design automation (EDU) which generates optimized soft error detecting logic circuits for flip-flops. When a soft error is detected that signal goes to a developed rollback control module (RCM). That RCM will reset the CPU and restores the CPU's register file from a backup register file using a rollback program guidance. After that CPU will able to restart from the state which is before the soft error occurred. In here researchers were developed another two modules called error reset module (ERM) that can restore the RCM from soft errors and error correction module (ECM) that corrects errors in RAM after error detection with no delay overhead.

In above mentioned soft error means, which are random transient errors. Those errors are the main cause of failures in microcontrollers which include reversal of a memory element's bit data due to factors such as alpha rays in a package, neutron strike and noise of the environment [21].

D. R. Avresky and M. I. Marinov published a research paper in 2011 which named "Machine Learning Techniques for Predicting Web Server Anomalies". The basic idea between servers on the web is to provide requests made by the client through the web using different transmission methods such as Services. Businesses relying on these services require the web servers to have reliability, availability and security in order to provide constant quality in the service provided. This document describes the quality ensured in these services.

The assumption made for this problem is mainly due to Resource Starvation. Resource Starvation is when a process that functions in Concurrent Computing is unendingly denied the necessary resource to continue & process the rest of its work. Resource Starvation is measured by the response time taken to cater requests under artificial workloads while collecting data on other resource parameters. The research provides proof that these recordings gathered from different artificial workloads can be applied to real world entities as well

Machine Learning is used to monitor & correlate the high response time and this is done by observing the system data. The goal of this analysis is to resolve issues of this variety in Web Servers, Operating Systems or in VM (virtual machine) Rejuvenation.

Based on the statistics provided by the Internet World Statistics, we could clearly notice a rapid rise QoS (quality of service) Internet Service Usage users and this gave several companies & industries to exist in the current world. The below listed out Companies/ Industries who gets affected by these figures since their prime business is offering Internet QoS,

- Cloud Computing
- Data Storage
- Hosting Providers
- Content Delivery
- Application Performance Management & other

Due to this high demand and dependence on network QoS, it is important for a particular service to be aware of its own deteriorating quality. Currently there are several self-monitoring network products that ensure that the QoS of services offered through the internet. The goal of this this research is to increase this area.

The benefits taken from this research can be applied to other areas as well and they have been listed down below,

- Proactive Software Rejuvenation
- Web Server Workload Balancing
- Web Server Performance Testing

Other… [22].

In this research mainly focused about detecting anomalies on a web server using machine learning technique. Hence, we are not going to use machine learning techniques for detecting anomalies in a server this research is not a good feed for us.

Risk prediction is an important part of the information security system. In accordance with the information security risk assessment process and combination of assets, threat, vulnerability and safety control measures, to strengthen the correlation among these factors and make the prediction results more objective for the target, the authors put forward a model based on the combination of the grey theory and analytic network process (ANP) with information security risk prediction. Establish the weight of each risk assessment element through the analytic network process (ANP) by analyzing interdependency and

feedback. Finally, set up systematic risk fuzzy comprehensive calculation to process data and build accurate mathematical model by combining with the risk assessment level.

firstly, the authors grasp the development law of information system through the processing of raw data and the establishment of the grey model, and confirm the preliminary scientific quantitative prediction for the system's future state; Secondly, use the network analysis method of ANP to compare each independent elements, so that the authors are able to calculate the weight value of each risk factor which affects the system security, reorder the weights, and propose more targeted and objective improvement measures; Finally ,combining with the weight value, to analyze risk objects, the authors obtain fuzzy membership matrix of judgment matrix and build the fuzzy mathematical model, calculate the value of the risk factors comprehensively, and treat it as the guidance, so that reliable guarantee for information system security can be provided. The model realized the grey theory prediction model and was applied in the field of information security, calculate accurate comprehensive weights of various risk factors in information system. In the system, internet elements are interdependent and give feedback to each other, thus combining the theory of fuzzy mathematics, satisfying the requirements of the objectivity and complex of information system, forecasting result is scientific and accurate, instructive significance as well [23].

This suggested a method for quantitative information security risk assessment and management in computer networks. This process evaluates an impact and possibility value for specific threats using fuzzy logic and analytic hierarchy process to evaluate. Using fuzzy rules and fuzzy interference system, evaluation vulnerabilities under the uncertainty.

Consider such types of assets - information, host, servers, and telecommunication equipment, IT-services (confidentiality, integrity and availability)

Consider three groups of external socio-political impact, internal impact, and direct financial losses. Most of these are qualitative, thereby they use the analytic hierarchy process for their quantitative evaluation. Evaluate priority weights of information asset regarding to confidentiality, integrity or availability.

Possibility Evaluation for Specific Threat

They suggest a method for quantitative evaluation of threat’s exercising possibility, which is based on the questionnaires. These questionnaires include questions about possibility factors for specific threat and some possible answers to these questions. Answer for every question. After the answering all questions assign number of points and they will assign possibility of the threat.

Vulnerability Evaluation

They suggest a few methods for vulnerability risk assessment. It is based on common Vulnerability Scoring system (CVSS).and they suggest new vulnerability assessment method based on expert judgments, fuzzy production rules and fuzzy logic.

Risk Assessment

They assess the information security risk for specific threat and specific vulnerability by following way.

Risk(Threat)=impact(threat).Possibility(treat).RL [24]

Cyber-attack is an attempt to exploit computer systems and networks. Cyber-attacks use malicious codes to alter algorithms, logic, or data. Securing information systems is thus critical. Multiple countermeasures need to be built The CVSS is an industry framework that helps quantify the vulnerability impact. This paper demonstrated a mathematical model to predict the impact of an attack based on significant factors that influence cyber security. Vulnerability and network traffic were selected as the influencing factors to predict CVSS score. Based on the score, the technical analyst can analyze the impact and take necessary preventive actions. This model also considers the environmental information required. It is thus generalized and can be customized to the needs of the individual organization.

TABLE 1: Project data points.

Y CVSS Score	*X*1 Vulnerability	*X*2 Network Traffic
2.1	20	324
5.3	53	623
1.0	15	235
8.0	85	932
2.9	28	438
3.0	25	498
3.8	38	391
1.0	18	132
1.2	16	177
5.9	63	823
4.3	39	579
2.8	30	455
1.1	14	231
4.2	35	725
5.4	51	740
1.9	21	345
2.0	25	432
4.1	37	467
6.2	58	845
1.1	15	111
2.3	22	191
1.2	16	182

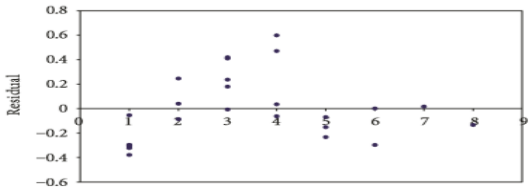


FIGURE 1: Residual plot.

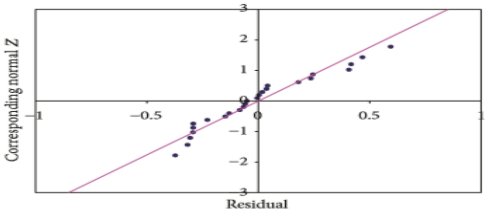


FIGURE 2: Normal probability plot.

Figure 1 :CVSS score prediction

Y is the overall CVSS score. CVSS is the predicted based on the environment and system characteristics of the target application. X1 is the number of vulnerabilities, namely, the total number of vulnerabilities detect by the static and dynamic vulnerability detection tool for target application.X2 is the average input network traffic.

In this regression model, CVSS score predict by the using two variables network traffic and vulnerability. Vulnerability and network traffic have no influence over CVSS score. No mirror pattern can be found (residual plot). Probability plot shown in figure 2is approximately linear. CVSS score is impacted positively both vulnerability and by network traffic.

Predicted CVSS score = intercept + Vulnerability * number of vulnerabilities +network traffic *average input networks

Intercept, vulnerability, network traffic can be calculate using regression equation [25].

This paper proposes a quantitative model for assessing cyber security risk in information security. The model can be used to evaluate the security readiness of firms in the marketplace through qualitative and quantitative tools. We propose a Bayesian network methodology that can be used to generate a cyber-security risk score that takes as input a firm's security profile and data breach statistics. The quantitative model enables cyber risk to be captured in a precise and comparable fashion. The objective of the scoring model is to create a common reference in the marketplace that could enhance incentives for firms to invest and improve their security systems. This paper concludes with a demonstration of scoring an intrusion detection network.

The Scoring mechanism determine from questionnaires are generated, the network is complete in both its qualitative and quantitative assessments. The scoring mechanism proceeds with a series of calculations to determine the score of a higher child node and similarly to the resource-driven security score.

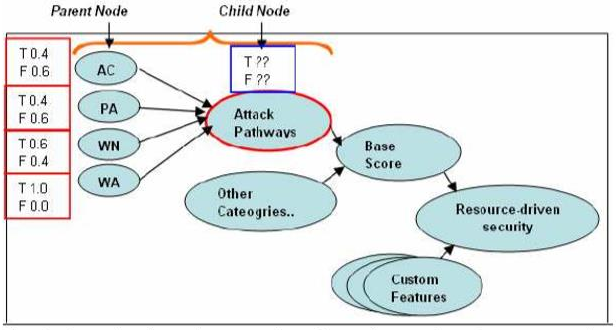


Fig. 8. Intrusion detection network to determine attack pathway prevention sub-score.

Figure 2 : Intrusion detection network prevention pathway

Scoring mechanism perform using Bayesian methodology and probability theorem [26].

2.5. Research Gap

Research Gap in Scanning Function

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 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. And, there is no system implemented that has the capability of scan a Linux server for any compliance failures and to classify them according to a criticality level.

Research Gap in Intelligent Hardening Function

Until now, there has been little work reviewed in the above researches in specifying or detailing the importance of Linux server security and Design and Implementation server hardening models to enhance the security of Linux servers. Based on researches "Automated Provisioning of Application in IAAS Cloud using Ansible Configuration Management" [27] and research on "Automated Provisioning of Application in IAAS Cloud using Ansible Configuration Management" [28] were discussed in length and defined as the following.

Development of an automated application deployment assembled using amazon as IAAS provider and Ansible as the orchestration engine. Using these metrics I introduce an alternative means of measuring its effects and impact on the field of Linux server security.

The automated Linux server hardening approach hasn't been reviewed so far. To bring the Full Potential of Ansible Framework functionality closer to system administrators, automated server hardening enabling them to do the configuration changes via the HardnBot's automated hardening functionality.

Research Gap in Backup and Automated Rollback Function

Server Hardening is one of the most important tasks to be handled on servers. Server hardening, which is also referred to as operating system hardening, is the process of making the server stronger and more resistant to security issues. Server hardening is an inexpensive and simple task to improve the overall operating system for maximum performance and to reduce expensive failures. Hardening is a Process requires many steps, all of which are critical to the success of the hardening system. The more steps a user follows, the safer and more resilient the system will be.

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/abnormal behavior occurrence is higher because hardening will carry out with human interaction. Scenarios where a misconfiguration or abnormal behavior 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. In the real world, there is no specific software or a tool which has concerned about all the difficulties met when a server is going to harden. Especially when we are talking about the abnormal behavior/misconfiguration detection, it is very hard to detect such kind of things and also very time-consuming approach after the hardening process. Maybe it will take 2 to 3 days' administrator to detect those things.

Research Gap in Risk Score Prediction Function

Until now, there has been little work reviewed in the above researches in specifying or detailing the importance of quantitative risk assessment. Based on researches "Identifying Potentially-ImpactedAreabyVulnerabilitiesinNetworkedSystemsusingCVSS" and research on "A Conditional Probability Computation Method for Vulnerability Exploitation Based on CVSS" [29]were discussed in length and defined as the following.

The Common Vulnerability Scoring algorithm, Bayesian probability theorem and CIS Benchmarks. Using these metrics, I introduce a risk score mechanism to the compliance issue for the server. The risk score prediction for compliances approach hasn't been reviewed so far. System administrators can be easily reviewed Server current state and generate report for the server current state.

3. OBJECTIVES

3.1. Main Objectives

This research aims to find an effective way to secure the datacenter Red Hat Enterprise Linux version 6, 7, and CentOS version 7, 6 servers and enhance productivity for the customers and employees with a low cost and few human interactions. Hence there is no fully automated hardening platform implemented yet, in this system implementation a novel automated open-source software is proposed.

Here fully automated free open source hardening platform is developed with the capability to detect poor or non-compliant configurations in a system (OS/DB/Application) and applying industry recommended fixes/configurations and secure systems by reducing its surface of vulnerabilities.

This research composes a great business value as it can cover 90% of an Information Systems Audit and hardening process. For internal audits, this software presents both opportunity and responsibility. By helping the organization understand and control risks and identifying opportunities/industry best practices to embrace. This software also will allow the internal audit to position themselves as trusted advisors.

i. Automatically detect failed compliances in a server.
ii. Automatically classify them according to a criticality level.
iii. Automatically detect uncompliant configurations of the server.

To make these critical servers more secure, this system can automatically perform a dry run feature. With the Dry Run feature, you can execute the playbook without having to make changes on the server [30]. With Ansible Dry Run you can see if the host is getting a configuration changed or not.

iv. Automated Linux server hardening

To prevent intentional and unintentional human errors, the automated system hardening solution explores and highlights the basic security configurations that should be performed to harden the security posture of a default Linux Operating System installation which can provide you with both scheduled or ad-hoc benchmark tests against CIS standard.

v. Bring industry best practices into the system.

For a particular parameter in the system configuration, there is an industry-recognized value. For example, a password should expire at most in 90 days. A company may not adhere to these standard values; its security policies may not describe them. On such occasions, system administrators may have assigned them with default values. Through our software, we plan to introduce industry-accepted values and configurations into the information systems.

vi. Develop an automated backup function

Provide the capability of automatically backup the server which is going to harden. The ultimate goal is to create an automated backup function which will save more time than the manual backup process.

vii. Develop an intelligence rollback function

Create a function for rollback the server when there are server misconfigurations or abnormal behaviors. Those two parameters supposed to detect automatically and after that user will be able to reset that server misconfigurations or abnormal behaviors through the function. If not, he/she can roll back the server again through the intelligence rollback function.

3.2. Specific objectives

i. Reduce manual work

Through this software, we target to reduce the manual effort needed for a complete system hardening. Almost all the parts of the audit and hardening could be performed through this software. Compliance issues are automatically detected. Reports can be generated through this software at the end of the hardening process.

ii. Ease the internal audit

This software is designed in a way that could be easily used by non-technical people. Simple interfaces, pop-up guidelines, and descriptive reports will make the user aware of the functionalities of the software. Even before applying a fix to an issue, the user can read a full detailed report of that issue including its impact and the remediation.

This software package can be handled by a single user. Therefore, the entire system's hardening process can be conducted by a single user. This requires a minimal amount of the organization’s resources. As for the internal audit, this software will cover up to 90% of their work.

4. METHODOLOGY

4.1. Setup a connection between the HardnBot and the remote server.

HardnBot has a separate interface to handle the connection to a remote UNIX server, using this interface system administrator can connect, execute UNIX terminal commands and disconnect form a server. To connect to a remote server, a system administrator must have the required remote server up and running and ssh services enabled to HardnBot to connect. Also, the system administrator must have the server IPv4 address, and root credentials.

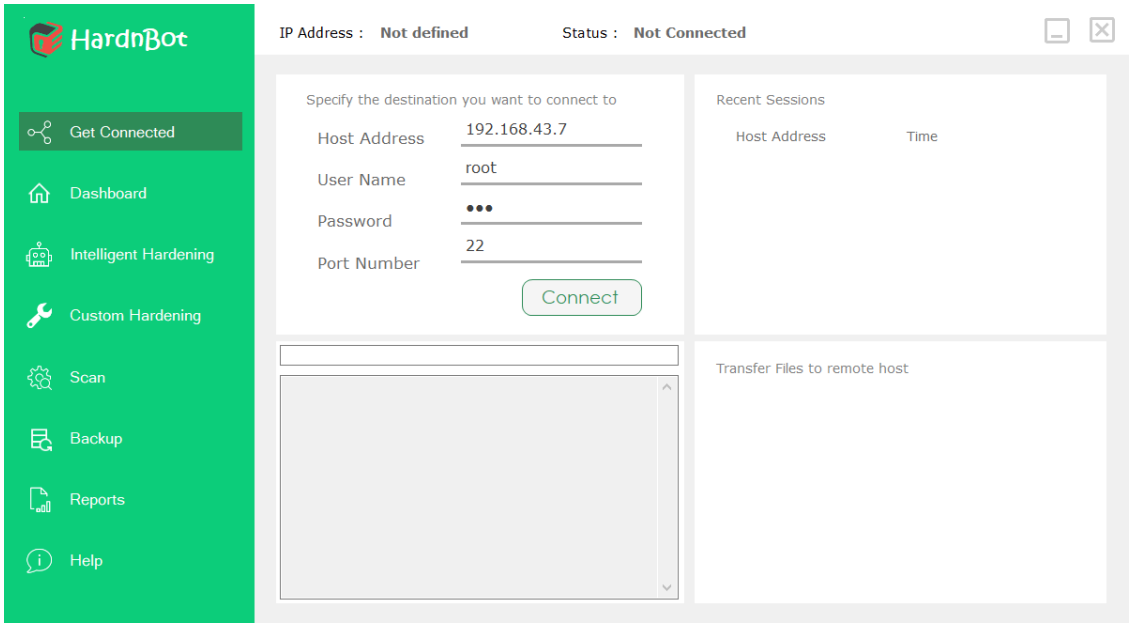


Figure 4 : HardnBot's Connection Interface

When developing the scanning interface, we used “Bunifu UI” framework to design the interface. We used C# as our main programming language of the software, but the required server-side scripts are developed using shell.

To get the connection, we used C# library named “Renci SshNet/SSH.NET [10 - 11]” and whenever the system administrator clicks on the “Connect” button HardnBot will get the user inputs (destination IP, root username, root password and the port) provided by the system administrator and pass it to the SshClient object we created. Next HardnBot will connect to the remote server and wait for 120 seconds to get a successful connection. Within this time if a connection failure occurs, HardnBot will notify the user before the 120 seconds we defined.

If the remote connection is successful, HardnBot will save the connection status to a variable, and this variable will be very useful in-order to check the connection status to other functions in HardnBot to function properly.

Below is the code segment of this functionality,

```
public static Boolean connectionstat;
```

```
this.sshClient = new SshClient(textBox1.Text,Convert.ToInt32(textBox4.Text),textBox2.
Text, textBox3.Text);
this.sshClient.ConnectionInfo.Timeout = TimeSpan.FromSeconds(120);
this.sshClient.Connect();
connectionstat = this.sshClient.IsConnected;
```

After connecting to the remote server, HardnBot will get the server's Operating system information, Kernel and Kernel released date and save them to variable that are used later to display this information to the user.

```
//get OS
SshCommand OS = this.sshClient.CreateCommand("uname -o");
OS.Execute();
os = Convert.ToString(OS.Result.ToString());
//get kernal
SshCommand KERNAL = this.sshClient.CreateCommand("uname -s");
KERNAL.Execute();
kernal = Convert.ToString(KERNAL.Result.ToString());
//get kernal_release-date
SshCommand Release = this.sshClient.CreateCommand("uname -r");
Release.Execute();
kernal_release = Convert.ToString(Release.Result.ToString());
```

After this, all parameters are saved to variables to later use. And then HardnBot will create the shell stream to get user commands form HardnBot connection interface.

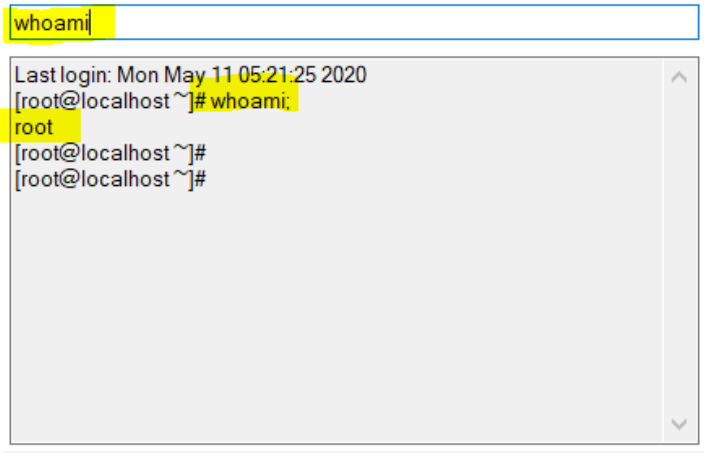


Figure 5 : HardnBot's CLI

```
IP = textBox1.Text;
UN = textBox2.Text;
PW = textBox3.Text;
port = textBox4.Text;
this.shellStreamSSH = this.sshClient.CreateShellStream("vt100",80,60,800,600,65536);
connectbtn.Visible = false;
disconnectbtn.Visible = true;
MessageBox.Show("Successfully Connected");
label5.Text = IP;
label7.Text = DateTime.Now.ToString();

//required method for the shell stream.
private void recieveSSHData()
{
while (true)
       {
              try
              {
                if (this.shellStreamSSH != null && this.shellStreamSSH.DataAvailable)
                  {
                     String strdata = this.shellStreamSSH.Read();
                     appendTextBoxInThread(txtSSHConsole, strdata);
                  }
              }catch
               {
               }
                 System.Threading.Thread.Sleep(200);
        }
}

private void appendTextBoxInThread(TextBox t, String s)
{
 if (t.InvokeRequired)
 {
  t.Invoke(new Action<TextBox, string>(appendTextBoxInThread), new oject[] { t, s });
 }
 else
 {
    t.AppendText(s);
 }
}
```

```
private void txtCommand_KeyPress(object sender, KeyPressEventArgs e)
{
    try
     {
           if (e.KeyChar == '\r')
                {
                    this.shellStreamSSH.Write(txtCommand.Text + ";\r\n");
                    this.shellStreamSSH.Flush();
                }
            }
            catch
            {
                throw;
            }
}
```

We also used accessor methods to return private static variables to other classes.

```
//accessor method for kernal
public static String getKN()
{
    return kernal;
}
//accessor method for kernal_release
public static String getKR()
{
    return kernal_release;
}
//accessor method for OS
public static String getOS()
{
    return os;
}
//accessor method for IP
public static String getIP()
{
    return IP;
}
//accessor method for UN
public static String getUN()
{
    return UN;
}
//accessor method for PW
public static String getPW()
{
    return PW;
}
//accessor method for Port
public static String getPort()
{
    return port;
}
```

4.2. Scan the connected server for security compliance misconfigurations.

After a successful connectivity to the remote server, system administrator can perform the compliance audit/scan against the server by navigating to the scanning interface.

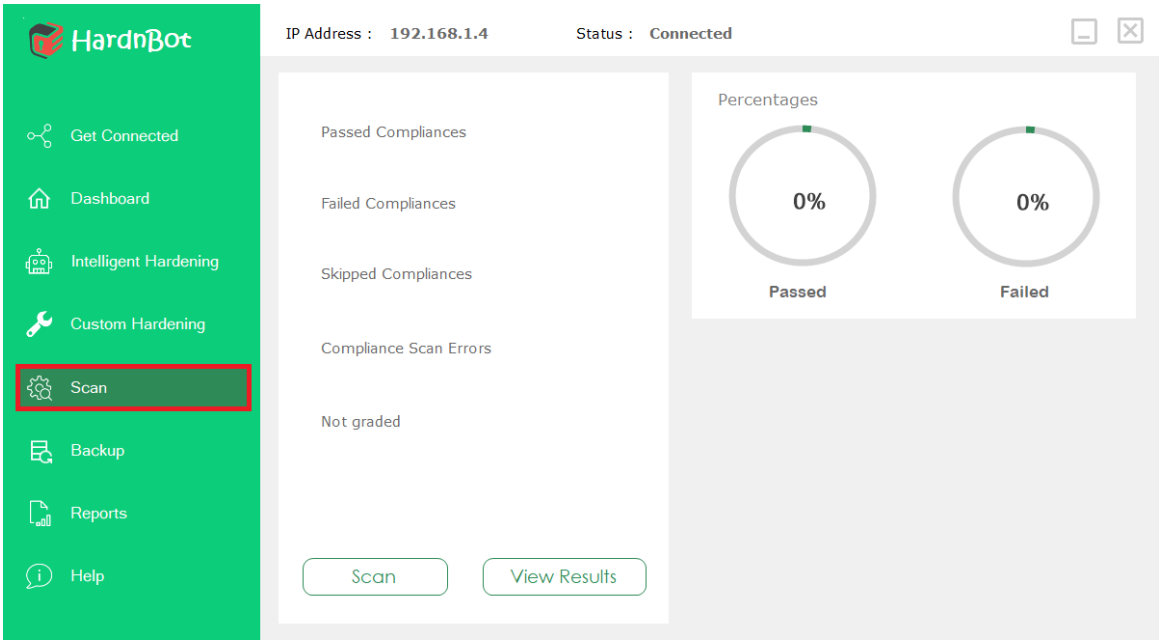


Figure 6 : HardnBot's Scan interface

After navigating to the compliance scanning interface, system administrator can click on "Scan" button and if there is a connection, HardnBot will ask the user to verify the scan before executing the scanning against the server.

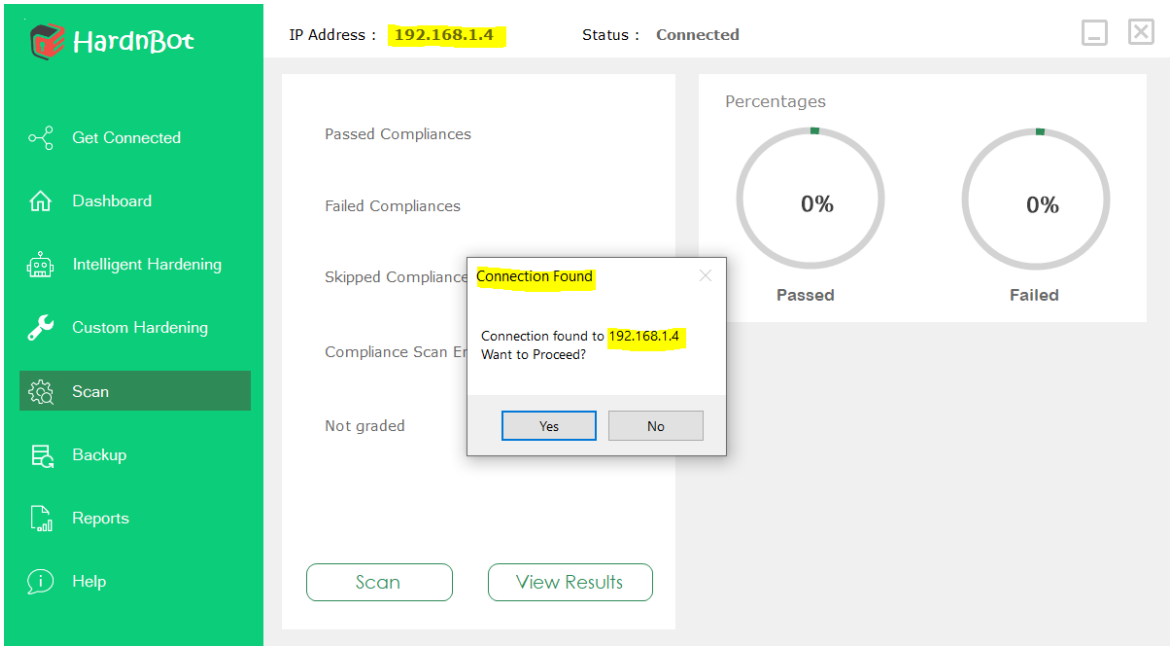


Figure 7 : Scan verification

If the user clicks on “Yes” the scanning process will begin.

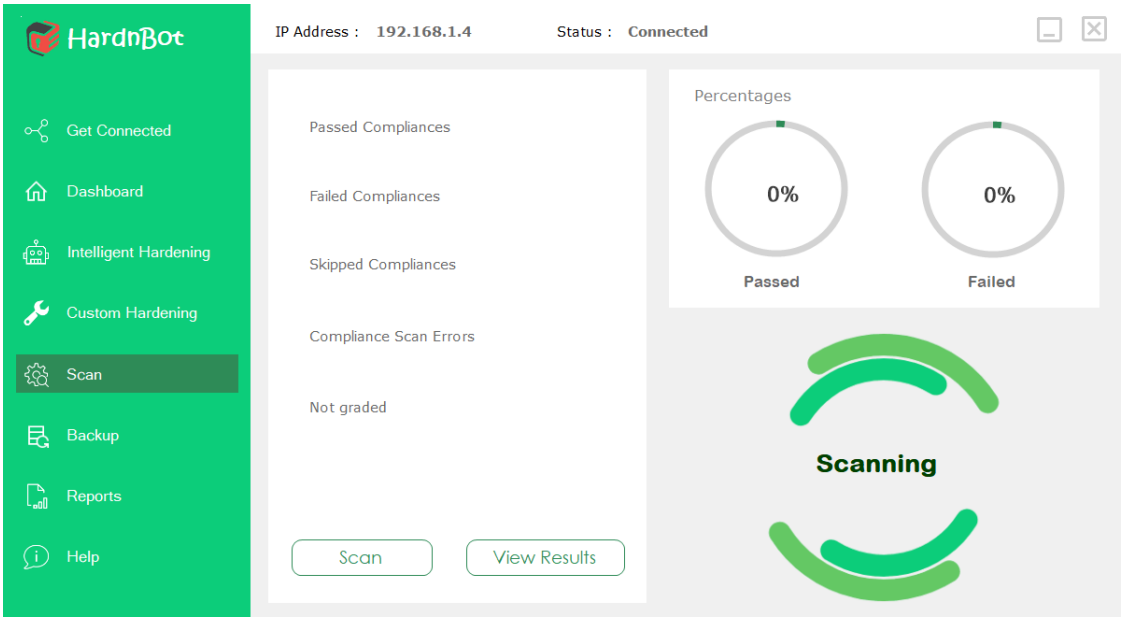


Figure 8 : Scanning process

After the scanning, HardnBot will receive the compliance audit results and display them in the scan dashboard following a message that says the “Scan Complete”.

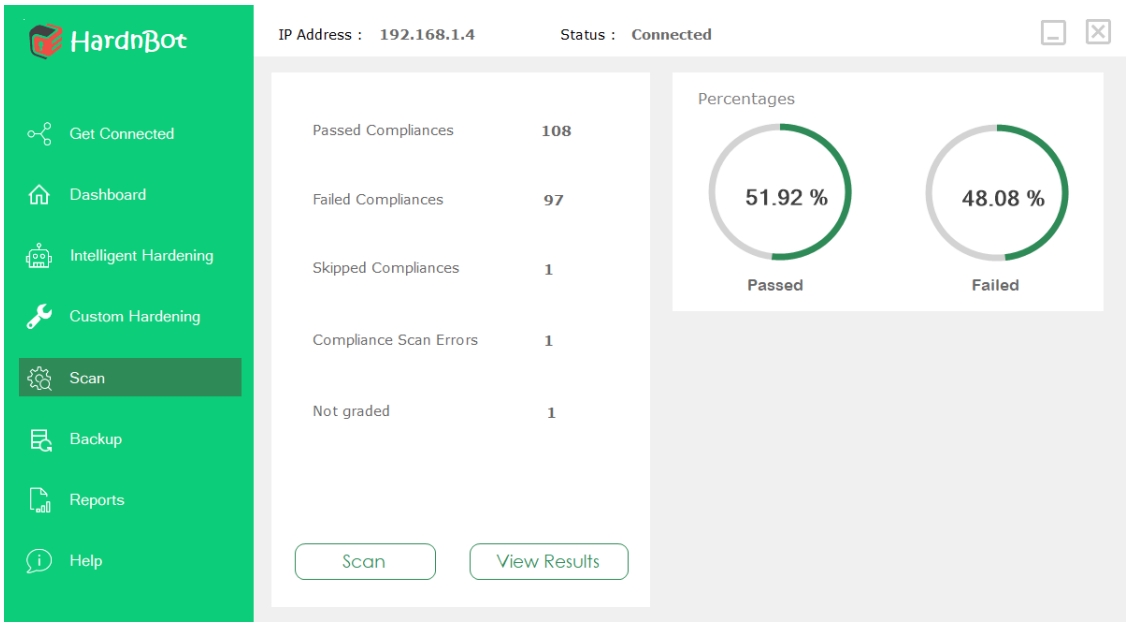


Figure 9 : Scan results

After that, user can press on the “View Results” button to get the classified compliance count according to the predefined severity levels.

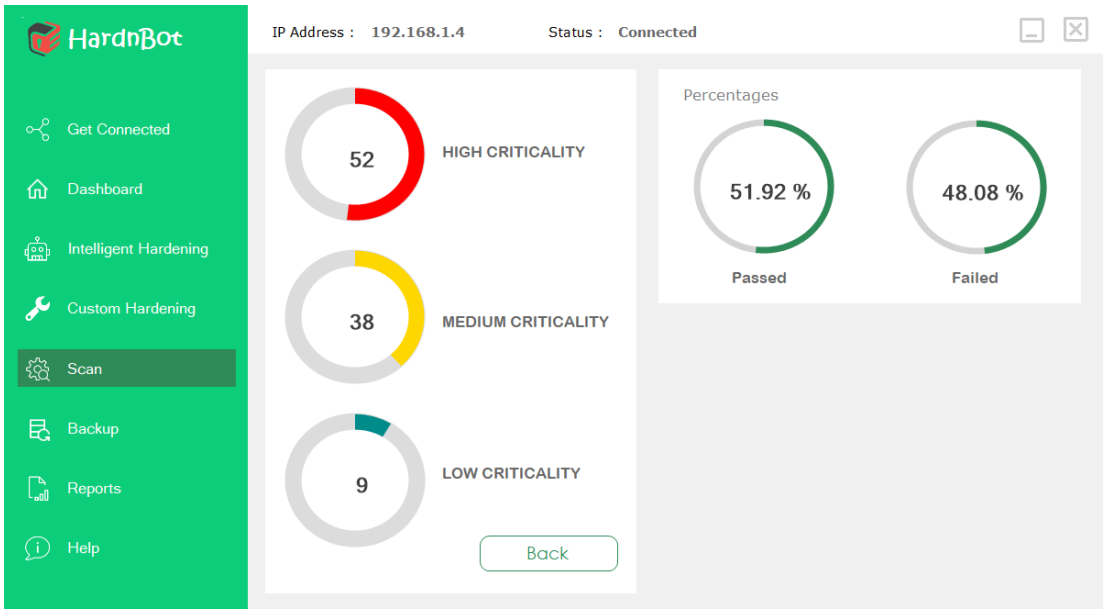


Figure 10 :Classified compliance count

Before we dive into the functionalities of “Scan” button click event, lets dive into the methods and functions defined in this scan class.

i. “Commands ( )” function.

In this function, I implemented all the commands that needs to execute when the scanning script was send to the server via the SendFilesToServer classes Send ( ) function. I will dive into this function later in this document.

So, this “Commands( )” function starts with a list of string variables assigned with some UNIX like commands to change file permission and then to execute the scanning script and after the execution, delete the scanning script form the server.

```
String chmod = "chmod 777 HardnBot_CentOS7.sh";
String runShell = "./HardnBot_CentOS7.sh > TestResults.txt";
String delShell = "rm -f HardnBot_CentOS7.sh";
```

Then, Commands ( ) function will again create a separate connection thread to perform scanning tasks. This is done so the user will not get freeze or crash if the user executes command to the server using HardnBot’s CLI in connection dashboard.

```
this.sshClient=new SshClient(con_dash.getIP(),Convert.ToInt32(con_dash.getPort()),con
_dash.getUN(),con_dash.getPW());
this.sshClient.ConnectionInfo.Timeout = TimeSpan.FromSeconds(120);
this.sshClient.Connect();
```

The connection will obviously get successful if there are not connection limitation on the server. After the connection, Commands ( ) function will execute the commands.

```
this.sshClient.RunCommand(chmod);
this.sshClient.RunCommand(runShell);
this.sshClient.RunCommand(delShell);
```

This will change the scan script's permission, execute it append the output to a text file and finally delete the script. After all this command executions, a text file will create with the output results of the scanning script. Using this text file, Commands ( ) function will get the count of all identified results including the failed compliances count, passed compliances count, skipped compliances count, and the count of compliance which are state as "Error" during the scanning process and all this results are assigned to variables defined in the scanning class.

```
SshCommand count = this.sshClient.CreateCommand("cat TestResults.txt | wc -l");
count.Execute();
CompCount = Convert.ToInt32(count.Result.ToString());
SshCommand command = this.sshClient.CreateCommand("cat TestResults.txt | grep Fail |
wc -l");
command.Execute();
FailedComp = Convert.ToInt32(command.Result.ToString());
SshCommand command1 = this.sshClient.CreateCommand("cat TestResults.txt | grep Pass |
 wc -l");
command1.Execute();
PassedComp = Convert.ToInt32(command1.Result.ToString());
SshCommand command2 = this.sshClient.CreateCommand("cat TestResults.txt | grep Skippe
d | wc -l");
command2.Execute();
SkippedComp = Convert.ToInt32(command2.Result.ToString());
SshCommand command3 = this.sshClient.CreateCommand("cat TestResults.txt | grep Error
| wc -l");
command3.Execute();
ErrorComp = Convert.ToInt32(command3.Result.ToString());
```

Then, Commands ( ) function will get all the failed, skipped and error occurred compliances IDs and pass them to an array. This is to check each failed, error occurred, and skipped compliances IDs against the pre-classified dataset of compliances which has the severity levels assigned to each compliance.

The output text file of the scan will also use here, and Commands ( ) function will grep out the failed , skipped, and error occurred compliance data and get all the relevant compliance IDs of them.

Assign these three types of string data to a single variable and assign this variable to a string array called “compz” with the split function to split the date using whitespaces.

```
private string[] compz;

SshCommand getFail = this.sshClient.CreateCommand("cat TestResults.txt | grep Fail |
cut -d' ' -f1");
SshCommand getError = this.sshClient.CreateCommand("cat TestResults.txt | grep Error
| cut -d' ' -f1");
SshCommand getSkipped = this.sshClient.CreateCommand("cat TestResults.txt | grep Skip
ped | cut -d' ' -f1");
getFail.Execute();
getError.Execute();
getSkipped.Execute();
_outputFail = Convert.ToString(value: getFail.Result.ToString());
_outputError = Convert.ToString(value: getError.Result.ToString());
_outputSkipped = Convert.ToString(value: getSkipped.Result.ToString());
String _outputAll = _outputFail + _outputError + _outputSkipped;
compz = _outputAll.Split(); //this is the array
```

Then, Commands ( ) function will assign values to it interface elements, assign some compliance counts to variables and finally disconnect the connection thread from the server.

```
HighProgressBar.Maximum = compz.Length;
MediumProgressBar.Maximum = compz.Length;
LowProgressBar.Maximum = compz.Length;
//count
int gradedCount = FailedComp + PassedComp + SkippedComp + ErrorComp;
NoGradeComp = CompCount - gradedCount;
this.sshClient.Disconnect();
```

ii. “**getResultsofCount ( )” function.**

This function was developed to cross reference the pre-classified dataset and the data in the “compz [ ]” array

Here I used a nested for loop to go through these two arrays. First outer loop used to loop through the pre-classified data set whilst the inner loop checks each compliance ID from “compz [ ]” elements with the outer loop’s current iteration’s element value to find any matching compliance ID. If the inner loop finds any matching compliance IDs, the relevant counter which has the relevant criticality level assigned will increase simultaneously.

The whole method is shown in the below code segment.

```
private void getResultsofCount()
{
    for (int x = 0; x < dataset.GetLength(0); x++)
    {
        for (int i = 0; i < compz.GetLength(0); i++)
        {
            if (compz[i] == dataset[x, 0] && dataset[x, 1] == "Medium")
            {
                Medcount++;
            }
            else if (compz[i] == dataset[x, 0] && dataset[x, 1] == "High")
            {
                 HighCount++;
            }
            else if (compz[i] == dataset[x, 0] && dataset[x, 1] == "Low")
            {
                LowCount++;
            }
        }
    }
}
```

(Here, the dataset is a two-dimensional array which has the compliance ID and its severity.)

So, these two are the main two important methods that I have implemented in the scan class of HardnBot. However, there is another important function that I implemented which handles the file transfers from C# application (HardnBot) to a remote Linux server. I named this class as “SendFilesToServer” and it has only one method called “Send”.

```
using Renci.SshNet;
using System;
using System.IO;

namespace Sherlock
{
    class SendFilesToServer
    {
        public static void Send(string host, string username, string password, string
 fileName)
        {
            using (ScpClient client = new ScpClient(host, username, password))
            {
                String Path = @".";
                client.Connect();
                client.Upload(new FileInfo(fileName), Path);
                client.Disconnect();
            }
        }
    }
}
```

To the “Send” function to work, it requires the remote host IPv4 address, root username, root password and the file name. the file needs to be in the root directory of the C# project (mostly in /bin/debug).

This function uses secure copy to copy files to the remote server. To use secure copy, I used “Renci SshNet’s ScpClient” library [31].

Here, I created a new SCP client and push the “Send” function’s arguments to it. (This was done to use the file transfer function anywhere in HardnBot.) Then I defined the path as @”.” Which will be the Linux server’s current directory that HardnBot’s “Scan” class got connected to. And, then the client connection, file upload and finally client disconnection. This class will heavily use in HardnBot’s other functions as well.

Now let us dive into the “Scan” button click event. When a user clicks on the “Scan” button it will check for any available connection via the variable assign in the connection dashboard.

```
if (con_dash.connectionstat == true)
```

If this connection status is true, HardnBot will continue its scanning by asking the user verification to the scan.

```
DialogResult result = MessageBox.Show("Connection found to " + con_dash.getIP() + "\n
Want to Proceed?", "Connection Found", MessageBoxButtons.YesNo);
if (result == DialogResult.Yes)
```

If the user clicks on “Yes” HardnBot will display the scanning circular animation and visible the required labels and call for the parallel thread called “backgroundWorker1”. I called the “SendFilesToServer ( )” function and the “Commands ( )” function inside this thread to prevent software freezing.

```
try
{
    pictureBox1.Visible = true;
    scanningLabel.Visible = true;
    backgroundWorker1.RunWorkerAsync();
}
catch (Exception exp)
{
    MessageBox.Show(exp.ToString());
    pictureBox1.Visible = false;
    scanningLabel.Visible = false;
}
```

The declaration of the parallel thread “backgroundWorker1” is as below,

```
private void backgroundWorker1_DoWork(object sender, DoWorkEventArgs e)
{
  SendFilesToServer.Send(con_dash.getIP(),con_dash.getUN(),con_dash.getPW(),"HardnBot
_CentOS7.sh");
  Commands();
}
```

In “View Results” button click event, HardnBot will check if the scan is completed otherwise it will prompt to run the scan first. If the scan is completed against the remote server, the “getResultsofCount ( )” will call here as well as three loops to display the data in a graphical manner.

```
private void bunifuThinButton21_Click(object sender, EventArgs e)
        {
            if (stat == false)
            {
                MessageBox.Show("Please run the scan first!");
            }
            else
            {
                getResultsofCount();
                outputPanel.Visible = true;
                for (int i = 1; i <= HighCount; i++)
                {
                    HighProgressBar.Value = i;
                    HighProgressBar.Text = i.ToString();
                    HighProgressBar.Update();
                }

                for (int i = 1; i <= Medcount; i++)
                {
                    MediumProgressBar.Value = i;
                    MediumProgressBar.Text = i.ToString();
                    MediumProgressBar.Update();
                }

                for (int i = 1; i <= LowCount; i++)
                {
                    LowProgressBar.Value = i;
                    LowProgressBar.Text = i.ToString();
                    LowProgressBar.Update();
                }

            }

        }
```

Scan script is written in shell (.sh) according to the CIS benchmark requirements and recommendations [1]. In CIS Benchmarks, they defined how to audit a compliance with the audit command. In HardnBot's scan script, those commands are implemented as functions and called upon the running time of the script and store the results in a temporary file.

Run the following commands and verify the output is as indicated:

```
# modprobe -n -v freevxfs
install /bin/true
# lsmod | grep freevxfs
<No output>
```

Figure 11 :Sample CIS Benchmark recommended audit command

Below, in figure 10 is a sample HardnBot's scan script's uses the CIS Benchmark recommendation to audit the service status of server audit daemon.

```
test_4.1.2() {
    id=$1
    level=$2
    description="Ensure auditd service is enabled"
    scored="Scored"
    test_start_time=$(test_start $id)


    [ $( systemd is-enabled auditd ) == "enabled" ] && result="Pass"


    duration="$(test_finish $id $test_start_time)ms"
    write_result "$id,$result"
}
```

Figure 12 :Scan Script's audit service status checking function (ID : 4.1.2)

Just like in figure 10, there are two hundred and twenty two (222) functions implemented for the scan script of HardnBot to perform the scanning operations.

4.3. Define a Benchmark for the Linux server security.

Center for Internet Security (CIS) Benchmark is referred to as the benchmark which is a consensus-driven security guideline for the Red Hat Enterprise Linux Operating Systems. CIS benchmarks are configuration standards and best practices for securely configuring a system. Each of the control references one or more CIS controls that were developed to help organizations improve their cyber defense competences. NIST Cyber security Framework (CSF) and NIST SP 800-53, the ISO 27000 series of standards, PCI DSS, HIPAA, and many more recognized standards and regulatory frameworks are mapped with CIS controls [32].

As the scope is limited to Red Hat Enterprise Linux Operating Systems version 6 and 7, following latest benchmarks were referenced to design formatted playbooks,

- CIS CentOS Linux 6 Benchmark v2.1.0

- CIS CentOS Linux 7 Benchmark v2.2.0
- CIS Red Hat Enterprise Linux 6 Benchmark v2.1.0
- CIS Red Hat Enterprise Linux 7 Benchmark v2.2.0

CIS benchmarks provide two levels of security settings:

- **Level 1** recommends essential basic security requirements that can be configured on any system and should cause little or no interruption of service or reduced functionality.
- **Level 2** recommends security settings for environments requiring greater security that could result in some reduced functionality.

The scope only focuses to find an automated way to perform all Level 1 configuration profiles of the CIS benchmark.

Items in Level 1 – Server profile intend to:

- be practical and prudent;
- provide a clear security benefit; and
- Not inhibit the utility of the technology beyond acceptable means, this profile is intended for servers.

4.4. Design formatted configuration libraries/Design Solution playbooks.

The playbooks are designed using YAML which is a human-readable data serialization language. Concerning the CIS benchmark level 1 profiles, playbook, designing is performed by six main modules. These modules perform hardening function as shown below;

1. Install Updates, Patches and Additional Security Software
2. Configurations of OS Services
3. Network Configuration and Firewalls
4. Logging and Auditing Configurations
5. System Access, Authentication, and Authorization
6. System Maintenance

HardnBot executes these modules by Ansible. Ansible works against multiple managed nodes or “hosts” in the infrastructure at the same time, using a list or group of lists in the inventory. Once the inventory is defined, we can select the hosts or groups you want Ansible to run against. The default location for inventory is located in `/etc/ansible/hosts`. Following is a basic inventory file in YAML format:

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        web1.example.com:
        web2.example.com:
    dbservers:
      hosts:
        db1.example.com:
        db2.example.com:
        db3.example.com:
```

These main sections of playbooks are scripted with CIS benchmark configurations. Before executing these modules firstly it checks OS version and family and Check ansible version and run Preliminary tasks list as follows.

```
- name: Check OS version and family
  fail:
    msg: "This role can only be run against RHEL 7. {{ ansible_distribution }}
     {{ ansible_distribution_major_version }} is not supported."
  when:
    - ansible_os_family == 'RedHat'
    - ansible_distribution_major_version is version_compare('7', '!=')
  tags:
    - always

- name: Check ansible version
  fail:
    msg: You must use ansible 2.1 or greater
  when: not ansible_version.full is version_compare('2.1', '>=')
  tags:
    - always

- include: prelim.yml
  become: yes
  tags:
    - prelim_tasks
    - always
```

Then it executes each above-mentioned playbook one after the other. These playbooks explore the following services and check what services are running on the server and turning off any unwanted services.

avahi_server	rpc_server	snmp_server	named_server	dovecot
cups_server	ntalk_server	squid_server	nfs_rpc_server	samba
dhcp_server	rsyncd_server	Samba server	mail server	squid
ldap_server	tftp_server	dovecot server	bind	net_snmp

telnet_server	rsh_server	httpd_server	vsftpd	autofs
nfs_server	nis_server	vsftpd_server	nfs_server	nfs_server

4.5. Implement Automated Hardening function.

To perform automated hardening Ansible configuration management, and application-deployment tool is used. Ansible is capable of run on many Unix-like systems and can configure both Unix-like systems as well as Microsoft Windows [33]. An interface is designed in HardnBot software to perform this function.

If any compliance gets compliant with the CIS benchmark, it skips and check and fix the rest of compliances by executing all the playbooks.

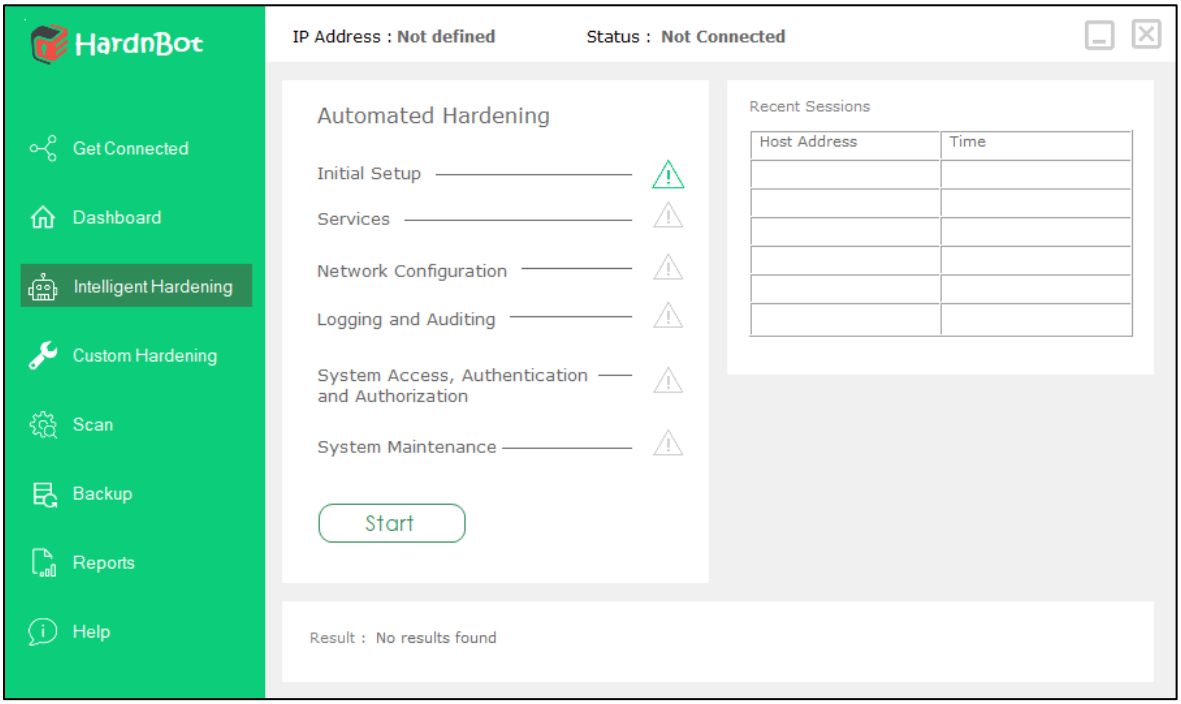


Figure 13 :Intelligent Hardening Interface

By this interface, the automated hardening function is called to execute the playbooks via Ansible engine. This intelligent hardening is more efficient for hardening default Linux Operating System installations. In this function, all the playbooks for each section are enabled and all the level 1 configurations will get executed by this function. Following is a sample code of enabled main 6 sections.

rhel7cis_section1: true
rhel7cis_section2: true
rhel7cis_section3: true
rhel7cis_section4: true
rhel7cis_section5: true
rhel7cis_section6: true

These well-formatted ansible playbooks will perform changes by checking every configuration. If an already compliant configuration is met it get skipped to the next configuration.

After navigating to the intelligent hardening interface, system administrator can click on “Start” button and if there is a connection, HardnBot will ask the user to verify the hardening before executing the hardening against the server, see Fig. (14). If the user clicks on “Yes” the intelligent hardening process will begin, see Fig. (15).

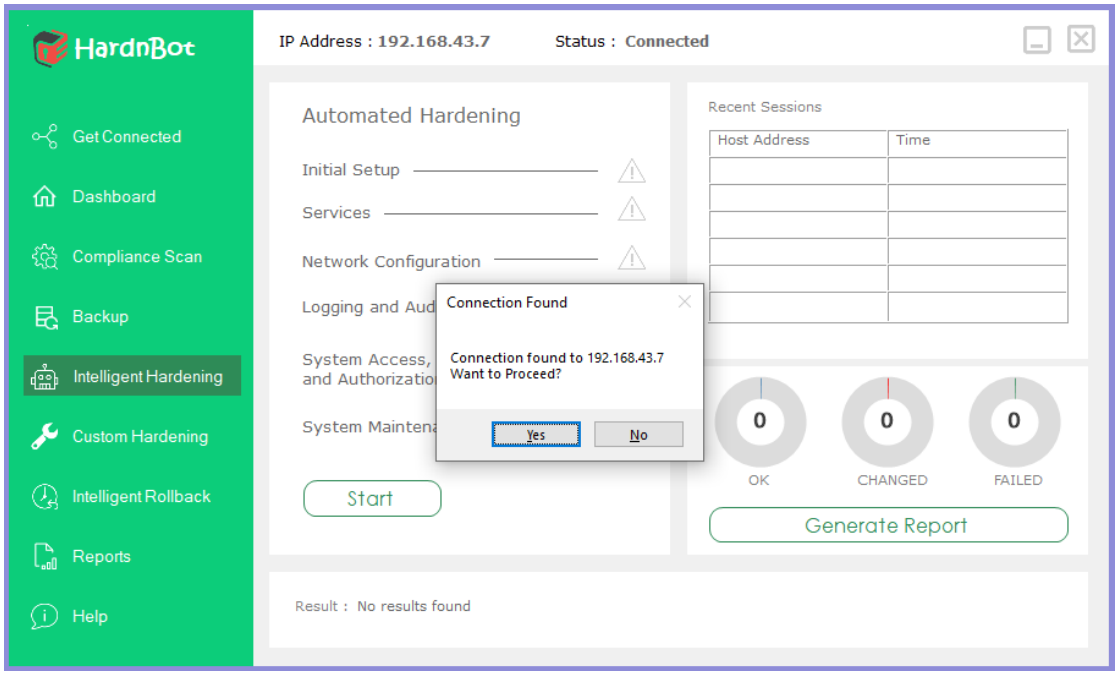


Figure 14 :Hardening verification

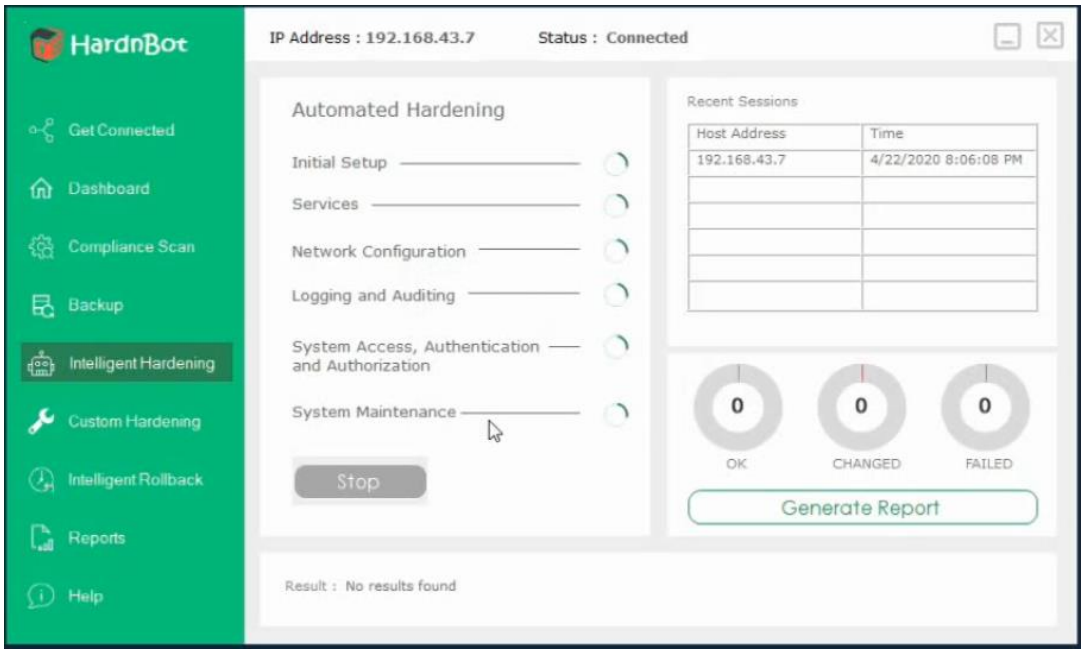


Figure 15 :Hardening in Progress

After the hardening, HardnBot will receive the hardening results and display them in the results panel following a message that says the “Scan Complete”, see Fig. (17).

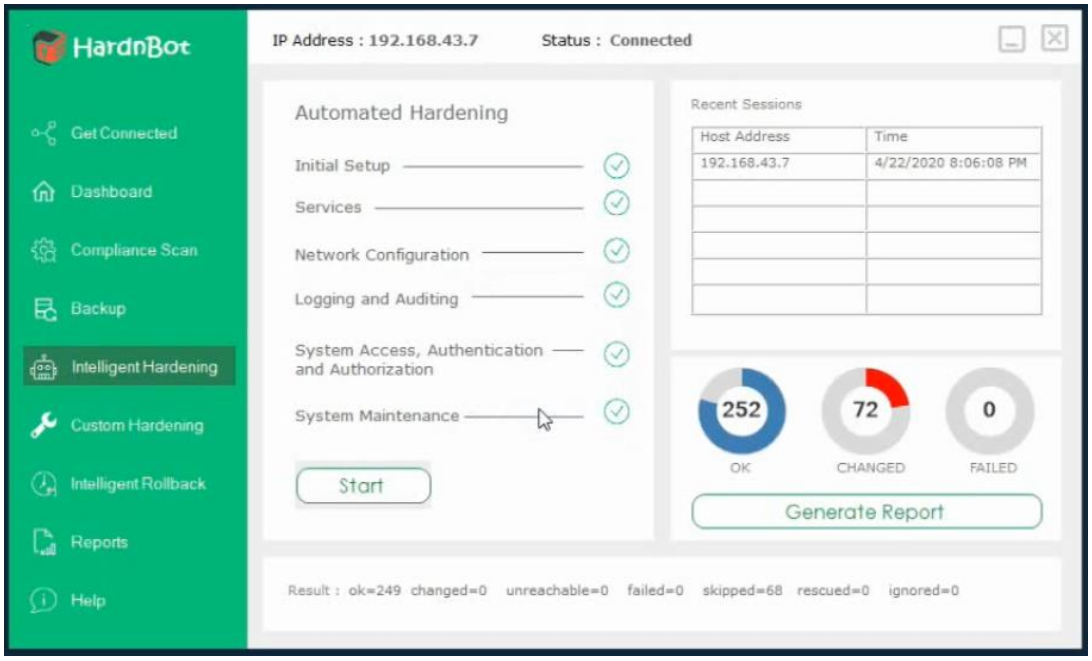


Figure 16 :Hardening results

4.6. Implement a parameterized Hardening function.

Parameterized hardening enables the system administrator to input industry-accepted values and configurations. In HardnBot there are interfaces designed to get user inputs to customize these configurations. Users can either disable or enable main sections or users are given separate interfaces to customize the configurations in each of the sections. For user's sake, when user move the mouse pointer to the relevant section's panel, it shows the configuration types which includes in the specific section, see Fig. (18).

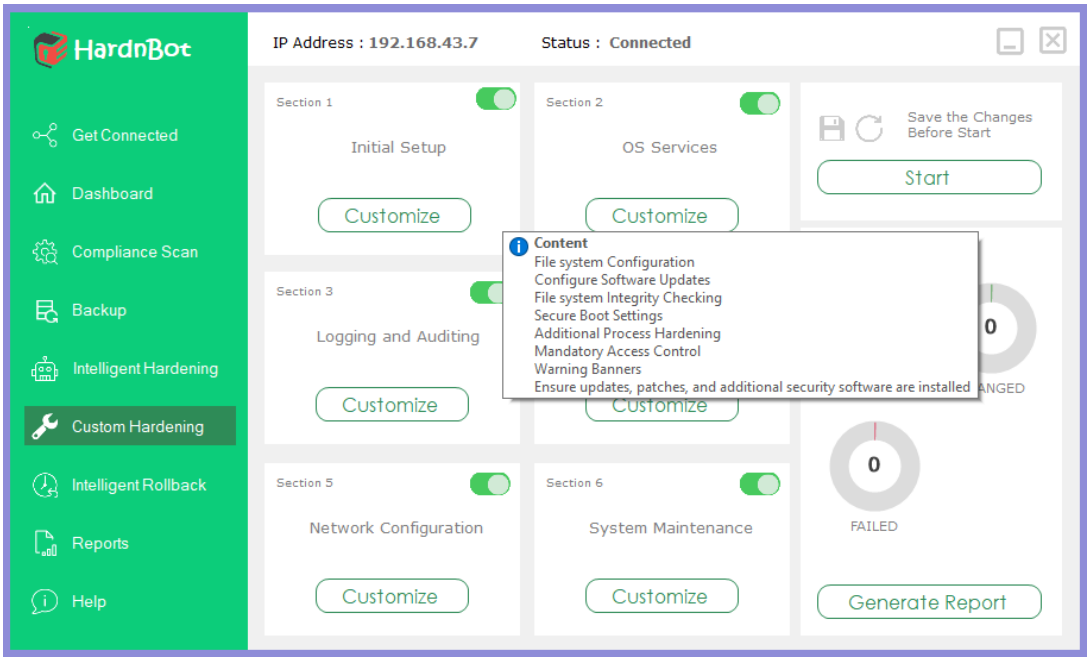


Figure 17 :Tool tip in main sections

Users can enable or disable whole sections as their requirements and users can modify the configurations inside these sections, see Fig. (19). the reset function enables users to reset the configurations to its default state.

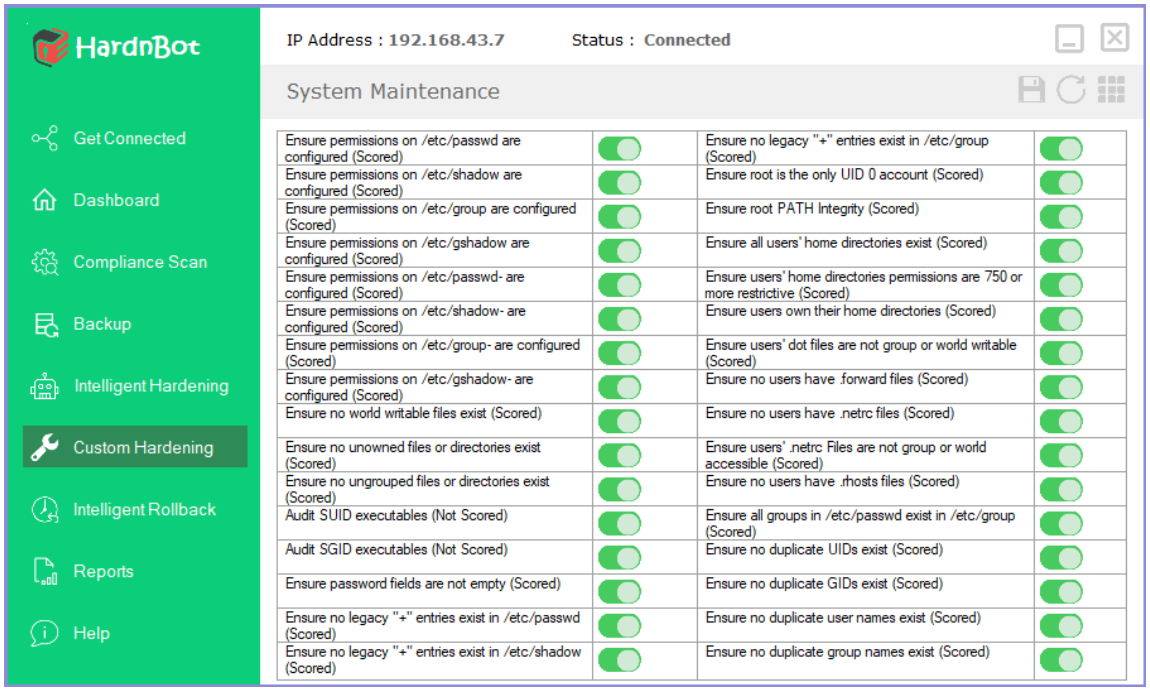


Figure 18 :Configurations in System maintenance section

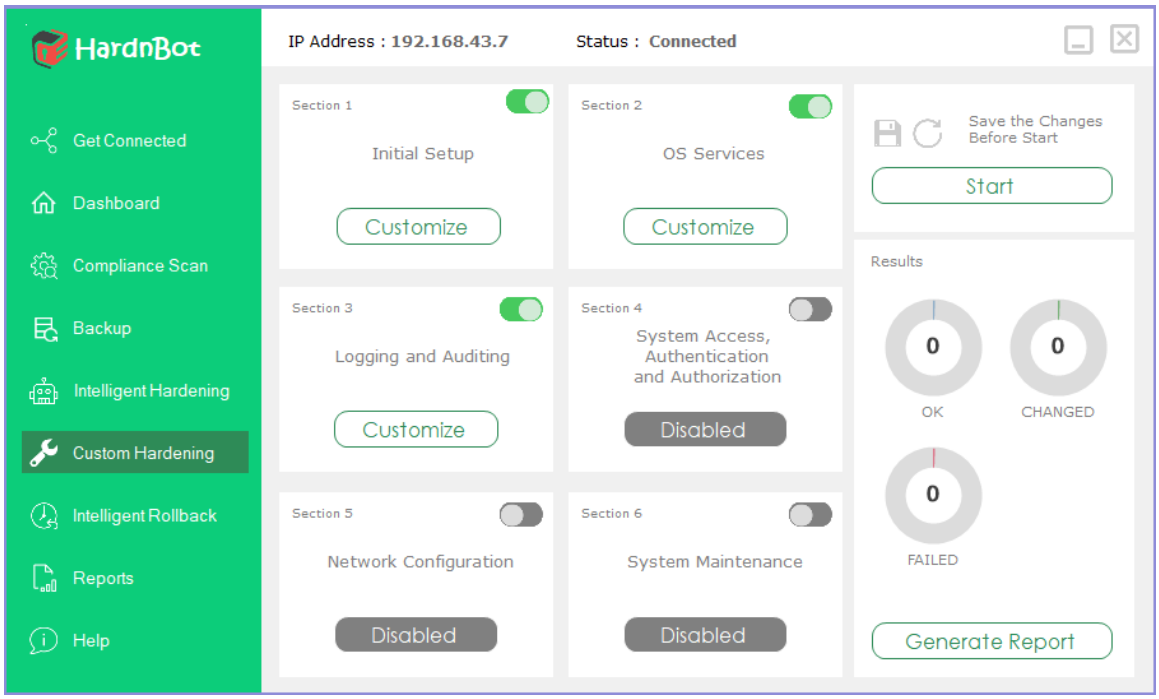


Figure 19 :Enable/Disable main section

As in the intelligent hardening function, HardnBot will ask the user to confirm the hardening before executing the hardening against the server and once it is completed HardnBot will receive the hardening results and display them in the results panel following a message that says the “Hardening Complete”, see Fig. (21).

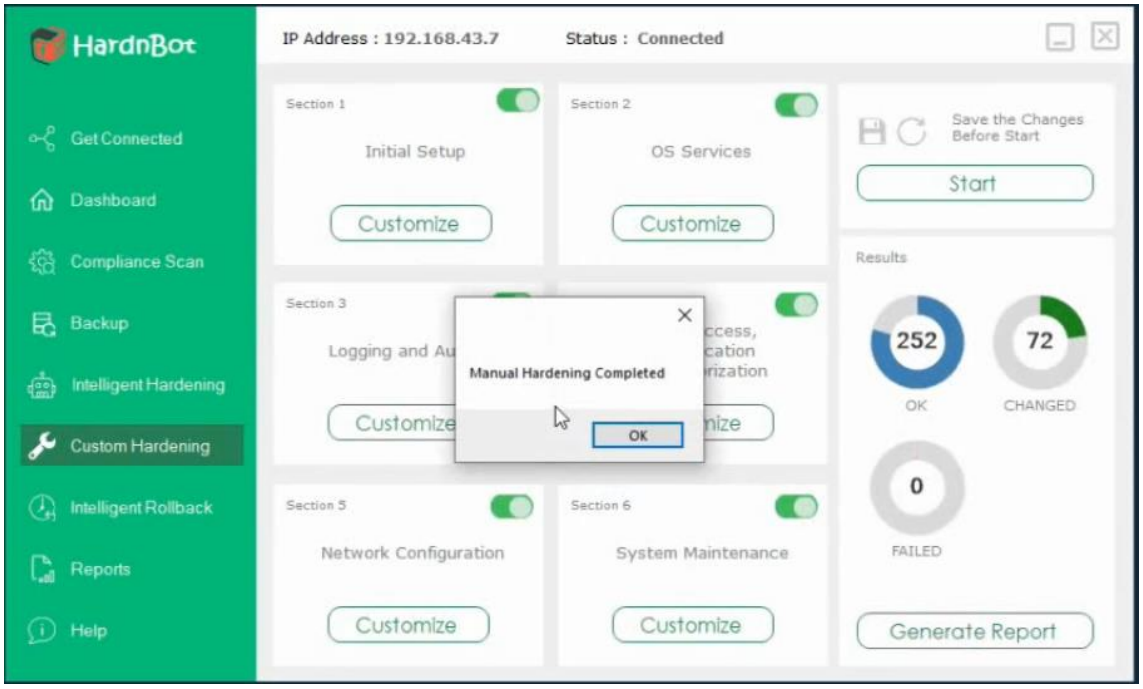


Figure 20 :Hardening results | Parameterized Hardening function

4.7. Automatic backup

In the manual process, backup is taken to either an external hard drive or mirror server and it is a time-consuming process and needs more human interaction to complete the task. When it comes to the HardnBot we have developed an automated solution for this task. The whole process is done by a shell script named “Backup_Script.sh” which includes all the instructions to collect the backup.

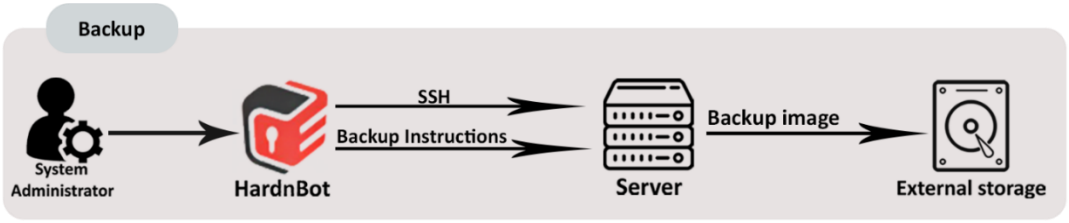


Figure 21 :System diagram of automated backup function

When the administrator needs to take a backup, simply what he/she needs to do is connect an external storage device which has enough free space to hold the backup image and select that block device ID (since Unix platform) from the HardnBot backup dashboard. Once the administrator selects that block device ID he/she can launch the backup process by click on the “Backup” button. That’s the big picture of the front end scenario.

A shell script which is used to perform this task contains three main sections.

Initially, it is creating a mount point for mount the block device which is going to store the backup image. In Linux platform, especially when we are talking about the Linux servers, before using any external storage device, it is must create an appropriate file system for the external storage device and mount it with a specific server location.

```
#!/bin/bash
DATE=`date +%a-%d-%b-%Y-%I:%M:%S-%p-%Z`
SERVER=`uname -n`
mkdir /media
mkdir /media/usb
mkfs.ext4 -F /dev/$1
```

This is the first half of the “Backup_Script.sh” script and initially in this half, it will collect the date of the backup and the server name for the future reporting purposes. Then it creates two new directories called “media” and “usb” which is inside the media directory for mounting purpose. File system creation is done by “mkfs” command which is short term of make file system. In our approach we used “ext4” file system type because it is more efficient than the other file system see Fig. 1,2; and “-F” for forcefully make the file system without any user interaction. “$1” will be taken as the first command line argument when the “Backup_Script.sh” beginning to run. In here $1 indicates the block device which has been chosen by the administrator earlier. Altogether from last line, it will make the ext4 file system forcefully for the block device which has been chosen by the administrator.

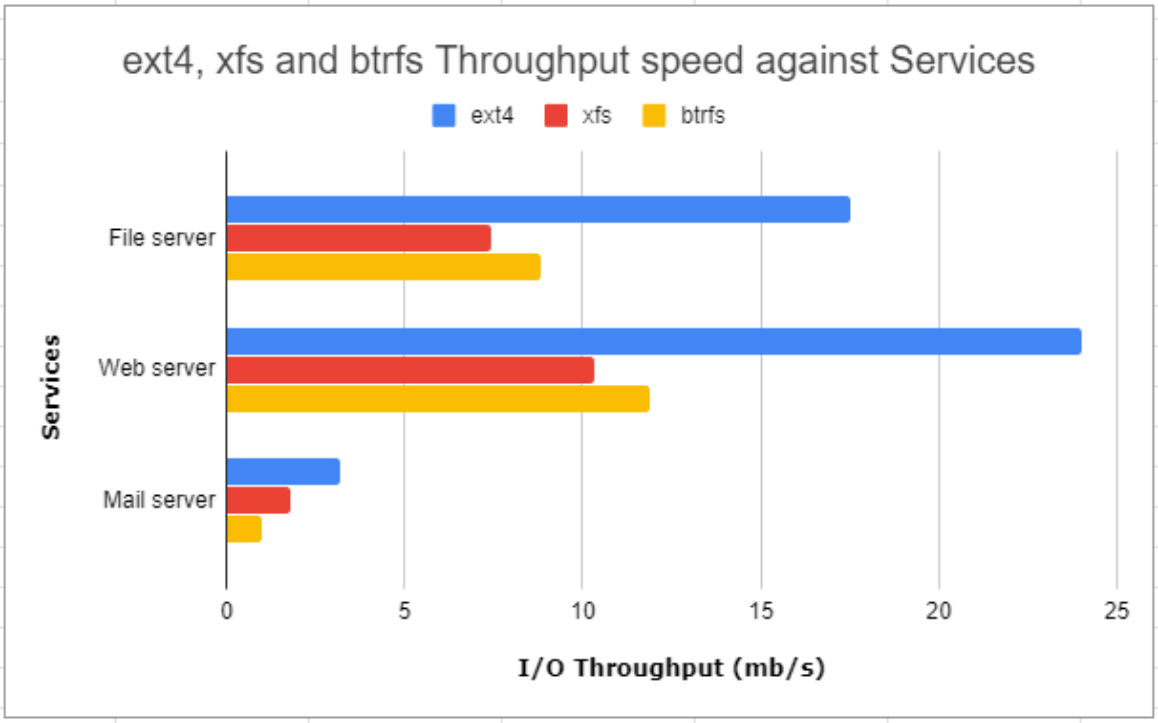


Figure 22 :ext4, xfs and btrfs throughput speed comparison against with services

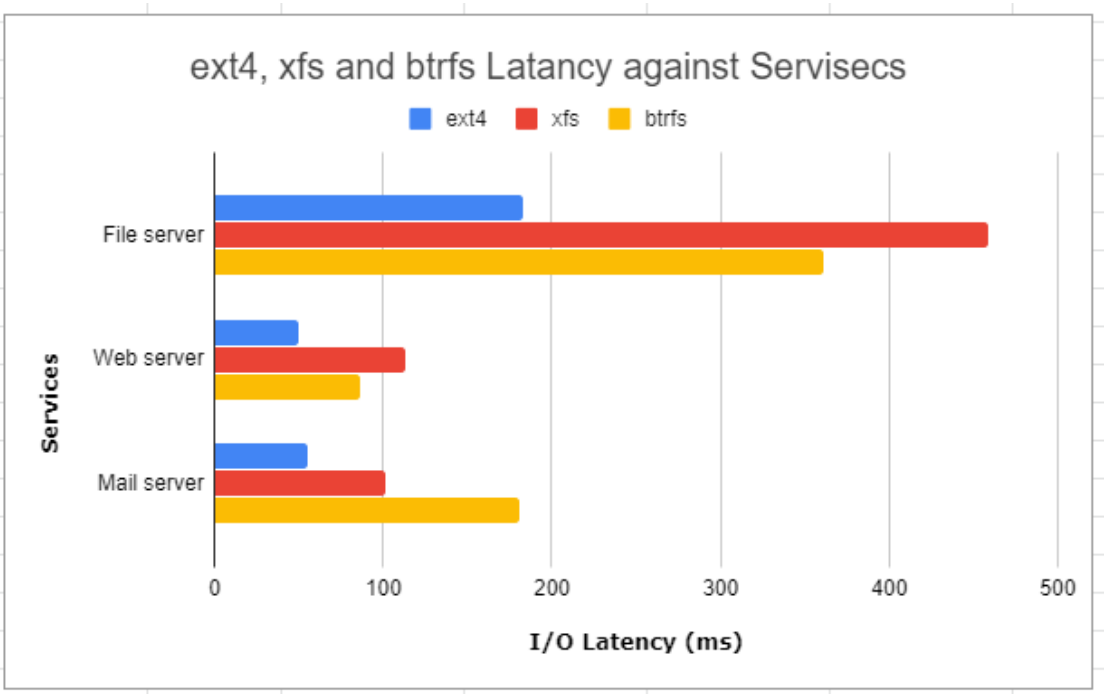


Figure 23 :ext4, xfs and btrfs latency comparison against with services

```
echo "/dev/$1 /media/usb ext4 defaults 0 0" >> /etc/fstab
mount –a
```

Middle part of the script is to mount the block device ($1) to created mount point (/media/usb). This configuration code must save inside the server "/etc/fstab" location to prevent discard of the changes after the reboot of the server. Finally, "mount –a" command will commit the changes.

```
echo "Starting backup for $SERVER..
dd if=/dev/sda conv=sync,noerror bs=64K | lz4 -z -f -B4 >
/media/usb/backup_image.img.lz4
echo "Done"
```

Last part of the script will take the backup using "dd" (disk dump) command and output of the dd command will go through the "lz4" compression algorithm and make a compressed backup image named "backup_image.img.lz4". Backup image will redirect to the mount point which is mounted with external storage device.

Lz4 is a comparatively most efficient and lossless compressing algorithm [34] see Fig. 3; and that is the process which is used to speed backup mechanism other than the file system formatting process.

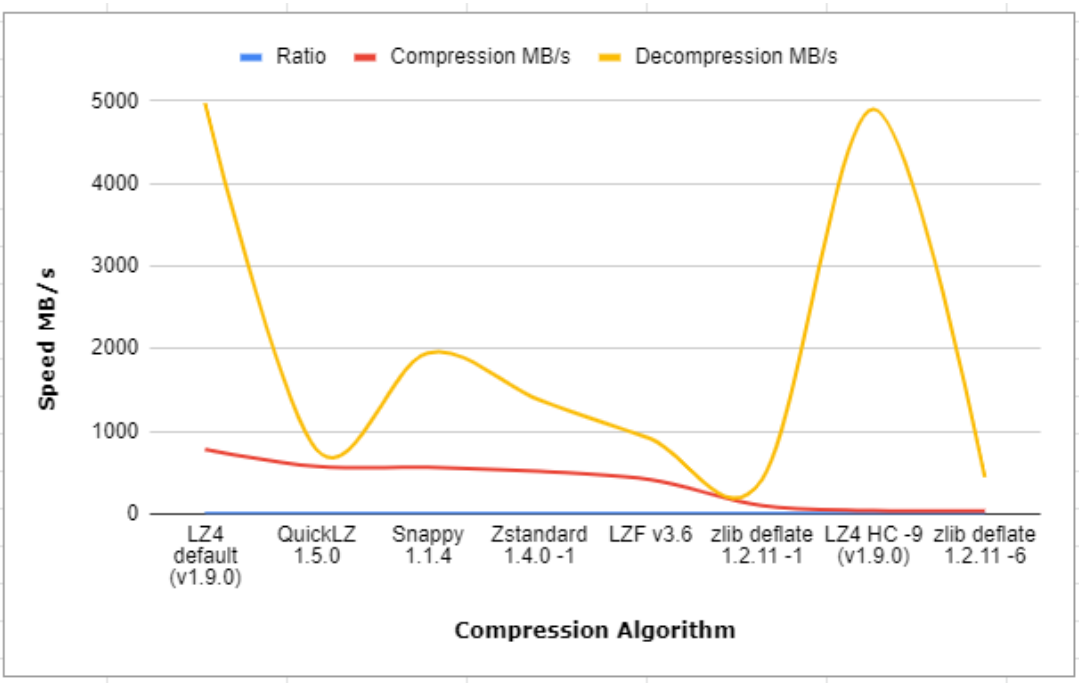


Figure 24 :Compression algorithms comparisons against the speed (MB/s)

4.8. Intelligence rollback

When we are considering the intelligence rollback function whole process is done through altogether 10 shell script. The whole process can be break down to four main parts.

1) Overall RAM usage comparison
2) Service wise RAM usage analysis
3) Check misconfigurations (enabled or disabled services during the hardening process)
4) Rollback the server

Overall RAM usage comparison

Two shell scripts named “int-ram.sh” and “fin-ram.sh” have been used to accomplish this overall RAM usage comparison task. Initially int-ram.sh script is executing before the hardening process is done. Output of the below command will be stored in “ram1.txt” file. Using “free” command user can get the in detail output of the RAM usage. Among them in here HardnBot will extract the total amount of used memory and free memory using “awk” command

```
free | awk '/Mem/{printf("Used          Memory:%.2f%\n"),$3/$2*100}/Mem/{printf("Free Memory: %.2f%\n"),  $4/$2*100}' > ram1.txt
```

Same pipe line command is executing after the hardening process and output will be stored in “ram2.txt” file. When the user wants to compare the RAM usage against before the hardening and after the hardening, he/she can click “Analyze” button and it will display the free and used RAM usages respect to the above two stages (before the hardening, after the hardening) on circular progress bar. User will be able to get an idea about RAM usage of the server by using this function.

Service wise RAM usage analysis

HardnBot is capable of detecting abnormal RAM usages of the services which are running on server. For this it will use three shell scripts named “ramusage.sh, int-ram.sh, fin-service.sh”. int-ram.sh is a same script which is used in Overall RAM usage comparison function as well. That means it utilizes
For both functions (2.2.1 and 2.2.2).

As said in previously int-ram.sh script is running before the hardening process. Inside this script it contains following commands to collect the RAM usage of each services before the hardening.

```
./ramusage.sh > ramusage1.txt
sed -e 's/ \+/\t/g' ramusage1.txt | cut -f13,14,17 | grep -w RAM > ram-        service1.txt
sed -e 's/ \+/\t/g' ramusage1.txt | cut -f15,16,19 >> ram-service1.txt
```

The ramusage.sh shell script is using inside the int-ram.sh script. First line executes the running command for the ramusage.sh script and output is redirect to the “ramsusge.txt”. By running the ramusage.sh script it provides services vice RAM usage. In here initially it will collect “private” and “shared memory” for each and every service running on the server. Overall RAM usage regarding each service is calculate using above private and shared memory.

```
Pss=`cat /proc/$PID/smaps | grep -e "^Pss:" | awk '{print $2}'| paste -        sd+ | bc `
Private=`cat /proc/$PID/smaps | grep -e "^Private" | awk '{print $2}'|        paste -sd+ | bc
let Shared=${Pss}-${Private}
echo -e "Private \t + \t Shared \t = \t RAM used \t Program"
```

Only the used memory and relevant service is extract from the ramusgae1.txt file and that output will redirect to the “ram-service1.txt” file.

Same process will be executing after the hardening process using fin-ram.sh script for collect the RAM usage of each services after the hardening and output will be redirect to the “ram-service2.txt” file.

When the user wants to clarify what are the services which is utilizing more RAM usage than before the hardening process, he/she needs to click on the increased services or decreased services buttons. Then following set of commands will execute and display to

the user if there are any services which are utilizing more RAM usage than the way it used before the hardening process or vice versa.

```
awk '
   NR==FNR{s1[$2]=$1;next}
     {
          s2[$2]=$1
   }
   END{
      for (value in s2)
      {
         if ((s1[value]!=s2[value] && s2[value] > s1[value]*1.5 ))
          {
                    print s2[value]-s1[value]  "MB" " Increased",$2,value
                    }
          else if (( s1[value]!=s2[value] && s1[value] > s2[value] ))
                    {
                              print (s1[value]-s2[value]) "MB" " Decreased",$2,value
                    }
      }
   }
' ram-service1.txt ram-service2.txt | column -t > f-outfile
```

Input files are ram-service1.txt and ram-service2.txt. This will get each service and its RAM usages respect to the two stages (before hardening and after hardening). Then compare if they (RAM usages) were increased or decreased and save them in an “f-outputfile” file. Either user clicks on increased services or decreased services button HardnBot will filter relevant services and display to the user with the increased or decreased RAM usage values.

Check misconfigurations (enabled or disabled services during the hardening process)

As mentioned in earlier, server hardening may cause to occur misconfigurations on the server, especially in services. Some disabled services can be enabled as well as some enabled services can be disabled during the hardening approach. We cannot prevent this issue, but we can detect and provide a solution for this issue.

HardnBot provides intelligence solution for this problem. For accomplish this whole task HardnBot uses five shell scripts (initialstate.sh, enfinstate.sh, disfinstate.sh, enable.sh, disable.sh).

At the beginning of the backup (before the hardening) the initialstate.sh script will run and collect the status of all services running on the server. Based on the status of the services they are divided into two parts (enabled and disabled). Basically that two status are the main status of any service. Then enabled services list redirect to enlist1.txt file and disabled services list redirect to dislist1.txt file.

```
mkdir /rollback

touch /rollback/enlist1.txt
touch /rollback/dislist1.txt

systemctl list-unit-files | grep enabled > /rollback/enlist1.txt
systemctl list-unit-files | grep disabled > /rollback/dislist1.txt
```

Same commands include in the enfinstate.sh and disfinstate.sh script with few changes. Those two scripts used to collect status of the services after the hardening process. Except that HardnBot will identify the enabled and disabled services during the hardening process using above two scripts. Then identified services respectively redirect to the two files named enable.txt and disable.txt.

```
touch /rollback/dislist2.txt

systemctl list-unit-files | grep disabled > /rollback/dislist2.txt
diff /rollback/dislist1.txt /rollback/dislist2.txt >    /rollback/disdiff.txt
awk '{print $2}' /rollback/disdiff.txt > /rollback/disable1.txt
awk 'NF' /rollback/disable1.txt > /rollback/disable.txt
touch /rollback/enlist2.txt
systemctl list-unit-files | grep enabled > /rollback/enlist2.txt
diff /rollback/enlist1.txt /rollback/enlist2.txt > /rollback/endiff.txt
awk '{print $2}' /rollback/endiff.txt > /rollback/enable1.txt
awk 'NF' /rollback/enable1.txt > /rollback/enable.txt
```

Once the user clicks “check enabled services” or “check disabled services” button from the “intelligence rollback” dashboard, HardnBot will display those services and asks to reset the services to their previous status. If the user wants to reset those services he/she can give the permission to the HardnBot and it will reset above services status using “enable.sh” and “disable.sh” shell scripts.

```
Cat /rollback/disable.txt | while read line
do
systemctl disable $line
done
```

Rollback the server

If the user unsatisfied with the overall server status after the hardening, then he/she can roll back the server to its previous stable status using the backup taken by earlier. This is done by “rollback.sh” script and basically it is almost same to the Backup_Script.sh script with few changes.

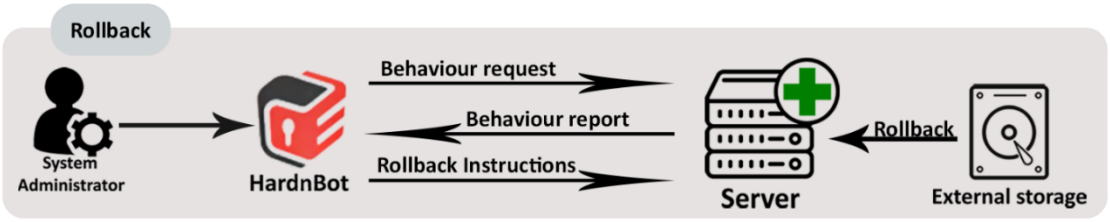


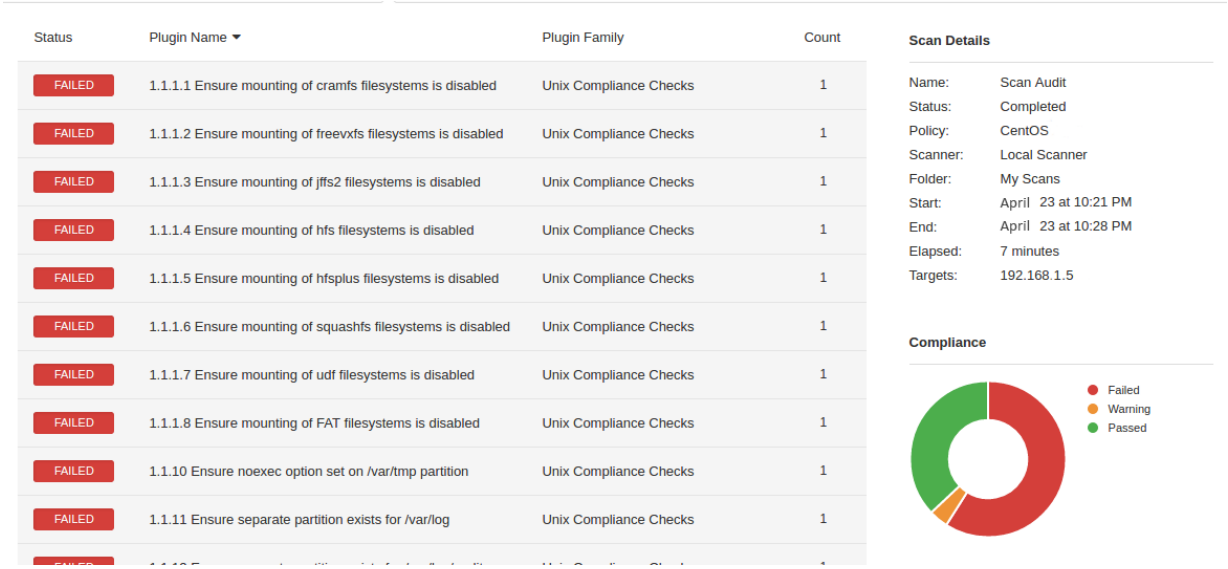
Figure 25: System diagram of intelligence rollback function

5. TEST RESULTS AND DISCUSSIONS

5.1. Compliance Audit

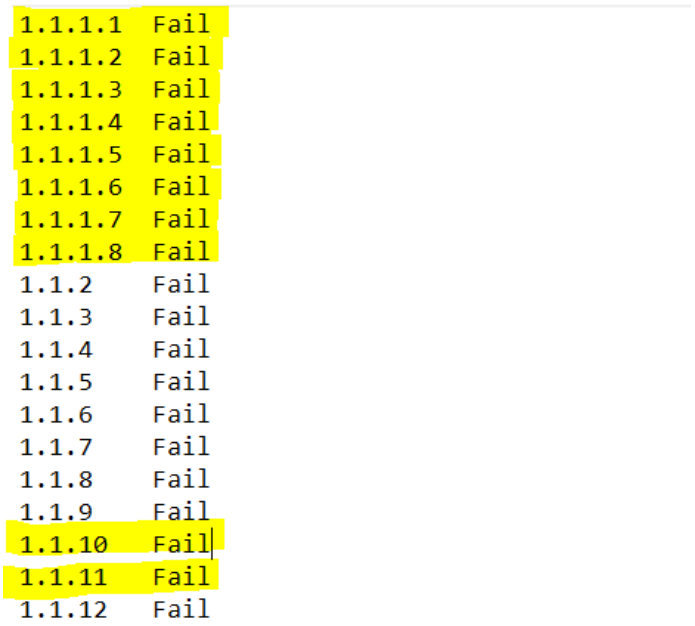
HardnBot’s scan functions show 99% correctness and accuracy when cross-referencing a standard Nessus compliance audit report.

Furthermore, we are digging deep into compliances in-order-to classify our pre-classified data set more accurately.



Status	Plugin Name	Plugin Family	Count
FAILED	1.1.1.1 Ensure mounting of cramfs filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.2 Ensure mounting of freevxfs filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.3 Ensure mounting of jffs2 filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.4 Ensure mounting of hfs filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.5 Ensure mounting of hfsplus filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.6 Ensure mounting of squashfs filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.7 Ensure mounting of udf filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.1.8 Ensure mounting of FAT filesystems is disabled	Unix Compliance Checks	1
FAILED	1.1.10 Ensure noexec option set on /var/tmp partition	Unix Compliance Checks	1
FAILED	1.1.11 Ensure separate partition exists for /var/log	Unix Compliance Checks	1

Figure 26 : Sample Nessus report



```
1.1.1.1  Fail
1.1.1.2  Fail
1.1.1.3  Fail
1.1.1.4  Fail
1.1.1.5  Fail
1.1.1.6  Fail
1.1.1.7  Fail
1.1.1.8  Fail
1.1.2    Fail
1.1.3    Fail
1.1.4    Fail
1.1.5    Fail
1.1.6    Fail
1.1.7    Fail
1.1.8    Fail
1.1.9    Fail
1.1.10   Fail
1.1.11   Fail
1.1.12   Fail
```

Figure 27 : HardnBot's Scan results

As in figure 11, we have a Nessus compliance audit report of one of our testing servers (Hosted in a hypervisor) and in figure 12 we have HardnBot's scan result of the same server. As you can see all the compliance IDs are matched with their status in both the Nessus compliance audit report and the HardnBot scan report.

In figure 11, it shows that Nessus took approximately seven minutes to scan this server. By using HardnBot's scan function the same server can be scanned within 28 seconds. However, this time may vary according to server content and policies.

Figure 28 : HardnBot's scan time

Figure 13 shows the time it took to scan the server by using a custom build HardnBot's scanning script. You can see the time it took on the left side of figure 13 (highlighted in yellow).

I copy the scanning script to the target server and execute it manually to obtain the results in figure 13.

Table 2 : Time comparison of scanning function

Time Nessus took (in seconds)	Time HardnBot took (in seconds)
420	< 30

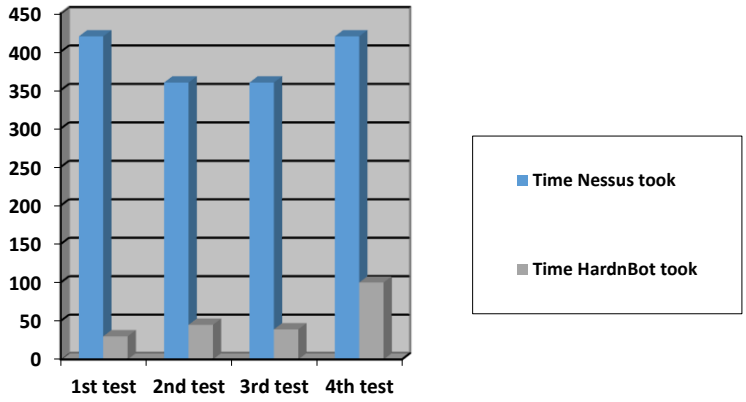


Figure 29 : Time comparison of scanning function

The above graph shows the test results of four tests ran on four servers and the time it took for both HardnBot and Nessus scanner. By analyzing these data, we can conclude that HardnBot's scan is faster than the Nessus scanner's scanning function.

However, the accuracy/correctness of HardnBot's scan results are not 100% accurate when comparing with Nessus reports but in the range of 95% - 100%.

5.2. Intelligent Hardening

To attest the functionality and the performance of the Hardening function, some trial series has been run and the trial results are reviewed in this section.

Experimental setup: The intelligent hardening software was run and tested using windows 10 client and it uses a Linux virtual machine as a jump server. The servers that were tested included Centos 7 and Red Hat 7 Linux distributions. These tested servers were virtual machines and were hosted in VMware Workstation 15.5.1.

Effectiveness: Intelligent Hardening was tested with freshly installed Centos 7 and Red Hat 7 servers and successfully hardened within 20-30 minutes. The following chart shows the order of the functionalities evaluated and its performance evaluation compared with the manual process and the HardnBot software. Following chart shows the time evaluated to complete a single server hardening successfully compared with manual process and automated intelligent process,

Table 3 : Time comparison of hardening function

	Manual Process	HardnBot
Hardening	< 5 hours	> 30 minutes

5.3. Backup and Intelligent Rollback

As mentioned in previously here we have utilized the “dd” command to get the backup and “LZ4” compression algorithm for compressing the backup. As mentioned previously for the experiment purpose, we have used Centos 7 and Red Hat 7 servers (virtual machines) which have 20GB hard drives on each server. When considering the time consumption, servers took 25 - 30 minutes (average) to take a backup.

Furthermore, the rollback function also examined using those virtual machines. Since LZ4 decompression rate is higher than the compression rate it only took less than 20 minutes to take servers to their initial state (rollback). Rollback is depending (not the speed) on abnormal behavior detection. Therefore, pretending as abnormal behavior to execute the rollback function, we have done some configuration changes to the servers. Therefore, it's mandatory to test abnormal behavior detection function as well. After configuring a few changes (appeared to be unusual changes) in servers (virtual machines) we executed an abnormal detection function and it took less than five minutes to identify and correct those unusual changes.

5.4. Risk score Prediction

HardnBot’s risk score prediction functions show 95% correctness and accuracy.

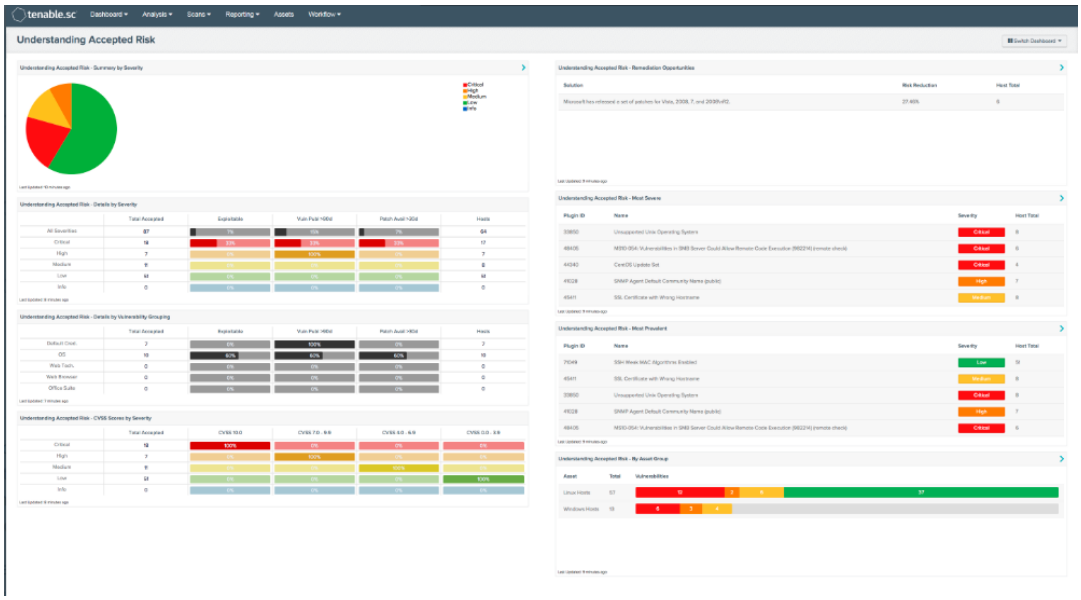


Figure 30: Nessus Scan results

	ID	Security	Category	RiskScore	Serverity
▷	1.1.1.2	Ensure mounti...	1.1.1(Disable u...	6.9	High
	1.1.1.3	Ensure mounti...	1.1.1(Disable u...	5.5	Medium
	1.1.1.4	Ensure mounti...	1.1.1(Disable u...	5.5	Medium
	1.1.2	Ensure /tmp is ...	1.1.1(Disable u...	7.3	High
	1.1.3	Ensure nodev ...	1.1.1(Disable u...	7.8	High

Figure 31: HardnBot's Scan results

As in figure 11, we have a Nessus vulnerability scanning report of one of our testing servers (Hosted in a hypervisor) and in figure 12 we have HardnBot's report of the same server.

In figure 11, it shows that Nessus took approximately 2 to 5 minutes to generate risk report this server. By using HardnBot's risk score prediction function the same server can be scanned within 1 to 2 minutes. However, this time may vary according to server content and policies. Nessus is not provide the risk report to the compliance issues in server.

The above details are shows the test results HardnBot and Nessus scanner. By analyzing these data, we can conclude that HardnBot's risk assessment is faster than the Nessus risk assessment function.

However, the accuracy/correctness of HardnBot's risk report are not 100% accurate when comparing with risk report but in the range of 80% - 90%.

6. CONCLUSIONS

In this research, we have blended into the automation of industry server operating system security hardening via scanning a server and identifying operating system compliance failures, classifying them according to the severity levels defined by a proper and thorough search in each and every security compliance in an operating system and then harden those identified issues with an efficient manner which is capable of hardening only required compliances automatically and ignoring unwanted compliances such as compliances related with IP tables manipulation. Then we predict the overall risk score of the server with the required parameters. We also implemented an intelligence rollback function that is capable of roll backing any predefined abnormal behavior occurring when hardening. The experiment results confirm the effectiveness of the system. However, the classification process is not that much a success since the data set is limited to 208 data. But another overall intelligent hardening process, intelligent backup, and rollback process and the risk score prediction process is a proven success.

7. REFERENCES

[1] V.Beal, "https://www.webopedia.com," 10 02 2011. [Online].

[2] "www.cisco.com," 28 February 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

[3] "Center for Internet Security," [Online]. Available: https://www.cisecurity.org/home.

[4] CIS, "Center for Internet Security," [Online]. Available: https://www.cisecurity.org/cis-benchmarks/cis-benchmarks-faq/.

[5] Y. T. A. R. Wita, "Vulnerability Profile for Linux," 2005. [Online]. Available: https://ieeexplore.ieee.org/document/1423610. [Accessed August 2019].

[6] A. B. M. M., "An Effective Modified Security Auditing Tool (SAT)," 2001. [Online]. Available: https://ieeexplore.ieee.org/document/937994. [Accessed August 2019].

[7] K. Zhao, "Design and Implementation of Secure Auditing System in Linux Kernel," 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4244819. [Accessed August 2019].

[8] J. Liu, "Research and Design of Security Audit System for Compliance," 2012. [Online]. Available: https://ieeexplore.ieee.org/document/6291450. [Accessed August 2019].

[9] F. Yip, "Enforcing Business Rules and Information Security Policies through Compliance Audits," 2006. [Online]. Available: https://ieeexplore.ieee.org/document/1649214. [Accessed August 2019].

[10] F. Yip, "Towards Robust and Adaptive Semantic-Based Compliance Auditing," 2007. [Online]. Available: https://ieeexplore.ieee.org/document/4566971. [Accessed August 2019].

[11] U. Thakore, "Combining Learning and Model-Based Reasoning to Reduce Uncertainties in Cloud Security and Compliance Auditing," 2019. [Online]. Available: https://ieeexplore.ieee.org/document/9049549. [Accessed August 2019].

[12] "webopedia," [Online]. Available: https://www.webopedia.com/quick_ref/servers.asp..

[13] Faraz Shaikh, Zoheb Shivani , "Snapshot service interface (ssi), a generic snapshot assisted backup framework for linux," in *International Conference on Digital Information Management*, 2007.

[14] Alireza Tajary, Hamid R. Zarandi , "An Efficient Soft Error Detection in Multicore Processors Running Server Applications," in *Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*, 2016.

[15] Olumuyiwa Ibidunmoye, Ewnetu Bayuh Lakew, Erik Elmroth , "A Black-Box Approach for Detecting Systems Anomalies in Virtualized Environments," in *International Conference on Cloud and Autonomic Computing (ICCAC)*, 2017.

[16] Yogendra Kumar Jain, Sandip S. Patil , "Design and Implementation of Anomalies Detection System Using IP Gray Space Analysis," in *International Conference on Future Networks*, 2009.

[17] Ming-Jen Chen, Chia-Chun Shih, Chien-Huei Yang, Gene Hong, Yuan-Sun Chu , "Multi-layered Monitoring Framework with Indices Relationship Analysis for Distributed Service Performance Evaluation," in *International Conference on Technologies and Applications of Artificial Intelligence*, 2011.

[18] Z.Wang, J.Zeng , " A Remote Backup Approach for Virtual Machine Images," in *IEEE*, Beijing, 2016.

[19] L. Farinetti, P.L. Montessoro , "An Adaptive Technique for Dynamic Rollback in Concurrent Event-Driven Fault Simulation," in *IEEE International Conference on Computer Design ICCD'93*, 1993.

[20] Yongmin Zhao, Ning Lu , "Research and Implementation of Data Storage Backup," in *IEEE International Conference on Energy Internet (ICEI)*, 2018.

[21] Teruaki Sakata, Teppei Hirotsu, Hiromichi Yamada, Takeshi Kataoka , "A Cost-effective Dependable Microcontroller Architecture with Instruction-level Rollback for Soft Error Recovery," in *Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*, 2007.

[22] M.I. Marinov, D.R. Avresky , "Machine Learning Techniques for Predicting Web Server Anomalies," in *International Symposium on Network Cloud Computing and Applications*, 2011.

[23] Qian Yu, Yongjun Shen, "Research of Information Security Risk Prediction," in *IEEE*, 2016.

[24] I. V. Anikin, "Information Security Risk Assessment and," in *IEEE*, 2015.

[25] Venkatesh Jaganathan, Priyesh Cherurveettil, Premapriya Muthu Sivashanmugam, "Using a Prediction Model to Manage Cyber Security Threats," in *ResearchGate*, 2015.

[26] Sheung Yin Kevin Mo, Peter A. Beling, Kenneth G. Crowther, "Quantitative Assessment of Cyber Security Risk using Bayesian," in *IEEE*, 2009.

[27] Nishant Kumar Singh ,Sanjeev Thakur , Himanshu Chaurasiya ,Himanshu Nagdev , "Automated provisioning of application in IAAS cloud using Ansible configuration management," in *IEEE*, Dehradun, 2015.

[28] Pavel Masek ,Martin Stusek , Jan Krejci ,Krystof Zeman, Jiri Pokorny, Marek Kudlacek , "Unleashing Full Potential of Ansible Framework: University Labs Administration," in *IEEE*, Jyvaskyla, 2018.

[29] L. LeylaBilge, RiskTeller:PredictingtheRiskofCyberIncidents, USA, 2017.

[30] admin, "mw[i]," [Online]. Available: https://www.middlewareinventory.com/blog/ansible-dry-run-ansible-check-mode/.

[31] Nudoq.org, "ScpClient (Class)," [Online]. Available: http://www.nudoq.org/#!/Packages/SSH.NET/Renci.SshNet/ScpClient. [Accessed August 2019].

[32] "docs.microsoft.com," [Online]. Available: https://docs.microsoft.com/en-us/microsoft-365/compliance/offering-cis-benchmark?view=o365-worldwide.

[33] "Ansible," [Online]. Available: https://www.ansible.com/.

[34] "LZ4 Benchmark," [Online]. Available: https://lz4.github.io/lz4/.