Scientific Research Publishing

Journal of Computer and Communications, 2023, 11, 108-136
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Strengthening the Security of Supervised Networks by Automating Hardening Mechanisms

Patrick Dany Bavoua Kenfack, Alphonse Binele Abana, Emmanuel Tonye, Genevieve Elvira Ndjana Leka

Department of Electrical and Telecommunications Engineering, National Advanced School of Engineering of Yaounde, University of Yaounde I, Yaounde, Cameroon
Email: danybavoua@gmail.com

How to cite this paper: Kenfack, P.D.B., Abana, A.B., Tonye, E. and Leka, G.E.N. (2023) Strengthening the Security of Supervised Networks by Automating Hardening Mechanisms. *Journal of Computer and Communications*, **11**, 108-136.
https://doi.org/10.4236/jcc.2023.115009

Received: March 3, 2023
Accepted: May 26, 2023
Published: May 29, 2023

Abstract

In recent years, the place occupied by the various manifestations of cybercrime in companies has been considerable. Indeed, due to the rapid evolution of telecommunications technologies, companies, regardless of their size or sector of activity, are now the target of advanced persistent threats. The Work 2035 study also revealed that cyber crimes (such as critical infrastructure hacks) and massive data breaches are major sources of concern. Thus, it is important for organizations to guarantee a minimum level of security to avoid potential attacks that can cause paralysis of systems, loss of sensitive data, exposure to blackmail, damage to reputation or even a commercial harm. To do this, among other means, hardening is used, the main objective of which is to reduce the attack surface within a company. The execution of the hardening configurations as well as the verification of these are carried out on the servers and network equipment with the aim of reducing the number of openings present by keeping only those which are necessary for proper operation. However, nowadays, in many companies, these tasks are done manually. As a result, the execution and verification of hardening configurations are very often subject to potential errors but also highly consuming human and financial resources. The problem is that it is essential for operators to maintain an optimal level of security while minimizing costs, hence the interest in automating hardening processes and verifying the hardening of servers and network equipment. It is in this logic that we propose within the framework of this work the reinforcement of the security of the information systems (IS) by the automation of the mechanisms of hardening. In our work, we have, on the one hand, set up a hardening procedure in accordance with international security standards for servers, routers and switches and, on the

other hand, designed and produced a functional application which makes it possible to: 1) Realise the configuration of the hardening; 2) Verify them; 3) Correct the non conformities; 4) Write and send by mail a verification report for the configurations; 5) And finally update the procedures of hardening. Our web application thus created allows in less than fifteen (15) minutes actions that previously took at least five (5) hours of time. This allows supervised network operators to save time and money, but also to improve their security standards in line with international standards.

Keywords

Hardening, Supervised Network, Cyber Security, Information System

1. Introduction

The digital and technological revolution is bringing its share of profound changes to the global economy. Technology is transforming jobs and skills. Companies, to satisfy their customers, are updating themselves by offering new up-to-date services in relation to technological developments. However, with this development, the world also faces an even greater cyber threat. Indeed, the information systems of organizations are very often victims of cyberattacks which affect both the company itself, whose activities are disrupted or even completely interrupted, but also impact, in a certain and sometimes irremediable way, the whole of its customers. One of the most reliable ways to prevent these attacks is to strengthen security configurations, also called system hardening or hardening.

System hardening is a collection of techniques and best practices aimed at reducing the vulnerability of applications, systems, infrastructure, firmware, and other areas. It is achieved by applying the latest patches and updates as well as following specific procedures and policies aimed at reducing the attack surface of the system [1]. The goal of system hardening is to reduce security risks by eliminating potential attack vectors and condensing the system attack surface. As a result, attackers and malware have fewer opportunities to penetrate a company's IT ecosystem.

Indeed, some companies carry out the hardening of their information systems still manually, as well as the verification of the hardening is done by the execution of a script that copies the configuration files. Members of the IS security team verify that configurations comply with corporate security policies. However, these are time-consuming, error-prone tasks that require successive checks. The work thus described is very costly in terms of human and financial resources, hence the interest in automating the system hardening process as well as its verification. It is in this perspective that this work fits.

Nowadays there is a plethora of platforms allowing the automation of Hardening, like Calcom Hardening Suite, Ansibln, Puppet … etc [2]. Our work aims

to remedy the weaknesses of the latter by proposing a more flexible device, capable of adapting to heterogeneous environments and having a better speed of execution as well as a lower cost of installation and maintenance.

The objectives to be achieved during this work are:

- Automate the execution of the hardening of servers and network equipment of the information system.
- Automate the verification of the hardening of servers and network equipment of the information system.
- Generate reports of compliance of servers and network equipment of the information system aligned with the security policy of Orange Cameroon.
- Correct the non-conformities noted during the verification.

2. State of the Art on Hardening

2.1. Definition of Hardening

According to the National Institute of Standards and Technology (NIST), the official definition of system hardening is: "a process of eliminating a means of attack by patching vulnerabilities and disabling non-essential services" [1].

System hardening is the process of correcting weaknesses and security vulnerabilities in systems. Hardening of systems is achieved by applying the latest patches and updates as well as following specific procedures and policies aimed at reducing the attack surface of the system [1].

Hardening can also be considered the process of securing a server or computer system by reducing its attack surface or vulnerability surface and its potential attack vectors. It is a form of protection against cyberattacks that involves closing loopholes in the system that cyber attackers frequently use to exploit the system and gain access to sensitive user data.

Hardening is therefore not a curative action and must be applied in order to avoid a problem and not following a problem. Similarly, an entire standard is not to be taken and applied blindly. However, some "good ideas" can be extracted and applied. Relying on hardening guides is a good basis for defining the prerequisites that the company's machines (client workstations and/or servers) must meet. By focusing on having these measures in place, a number of malicious acts can already be prevented. The security of the information system is all the better, as shown in **Figure 1**.

2.2. Importance of Hardening

Cyber security means such as VPNs, DMZs, anti-viruses, IDS/IPS are perimeter security solutions. They make it possible to implement barriers to prevent hackers from gaining access to the secure perimeter. However, they do not ensure the security of systems once there is a potential intrusion. This is where system hardening comes in. Indeed, hardening allows us to take into account hypotheses such as: the intrusion of the perimeter to be protected or the malicious actions of an employee.

Figure 1. The different aspects of hardening [1].

If very often some companies do not take the subject of system hardening very seriously, there are several reasons why they should integrate it into their security strategy.

2.3. Types of Hardening

Although the definition of systems hardening applies to an organization's entire IT infrastructure, there are several subsets of this idea that require different approaches and tools [2].

- **Network hardening**

Network devices are hardened to prevent unauthorized access to a network's infrastructure. In this type of hardening, vulnerabilities in the management and configuration of devices are sought and corrected in order to prevent their exploitation by malicious actors who wish to gain access to the network. Increasingly, hackers are using weaknesses in network device configuration and routing protocols to establish a persistent presence in a network rather than attacking specific endpoints.

- **Server hardening**

The process of server hardening involves securing a server's data, ports, components, functions, and permissions. These protocols are executed system wide on the hardware, firmware and software layers.

- **Application hardening**

Application hardening focuses on software installed on the network. An important aspect of application hardening sometimes referred to as software hardening or software application hardening is applying patches and updating vulnerabilities. Again, patch management through automation is often a key tool in this approach.

Application hardening also involves updating or rewriting application code to increase its security, or deploying additional software security solutions.

- **Data base hardening**

Database hardening focuses on reducing vulnerabilities in digital databases and Database Management Systems (DBMS). The objective is to strengthen the repositories of data, as well as the software used to interact with this data.

- **Operating system hardening**

Operating system hardening is all about securing a common target of cyberattacks: a server's operating system (OS). As with other types of software, hardening an operating system according to Figure 2 typically involves patch management that can monitor and automatically install updates, patches, and service packs.

2.4. How to Harden?

System hardening is a variable process. When it must be carried out, a list of hardening controls for the systems to be executed must be established upstream, taking into account the company's security policy and referring to the system hardening standards which are published by the organizations security such as CIS Center or NIST.

Once the hardening checks have been implemented on the equipment, it is necessary to check that they have been carried out in accordance with the hardening procedure shown in Figure 3.

2.5. Advantages/Disadvantages

Advantages

Hardening of systems brings us to [4]:

- **The securing of the system**: By reducing the attack surface that cybercriminals can use, you will rely on a more secure system both actively and passively. This is due, among other things, to the fact that you will be able to implement more secure passwords. By reducing the attack surface that cybercriminals can use, you will rely on a more secure system both actively and passively. This is due, among other things, to the fact that you will be able to implement more secure passwords.
- **Performance improvement**: Your computers will be able to run faster because you eliminate unnecessary overhead such as programs, services, users and ports that you don't use.

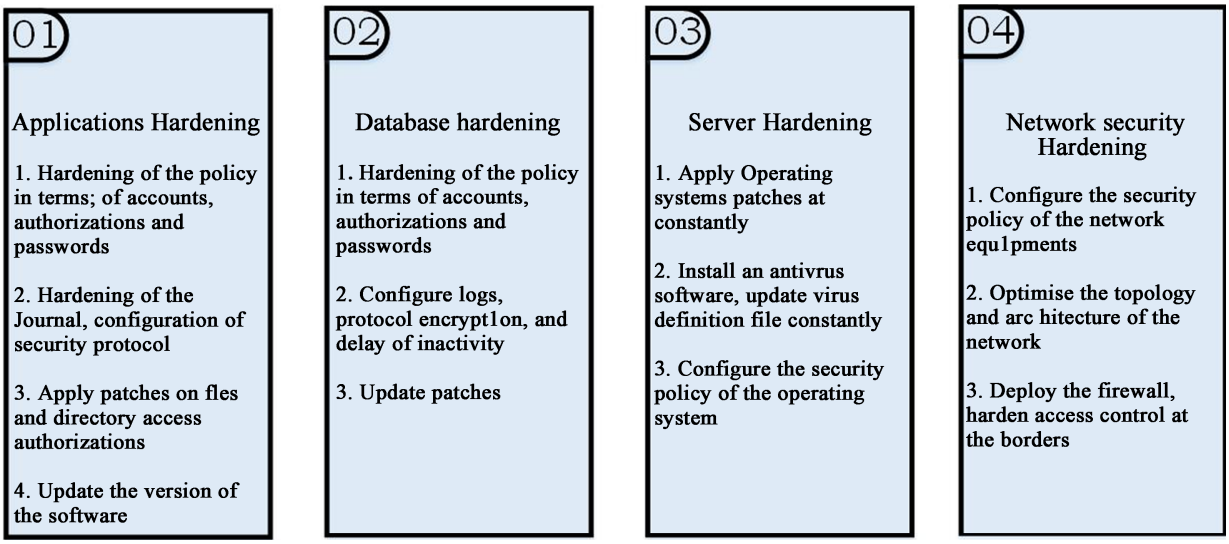


Figure 2. The types of hardening [3].

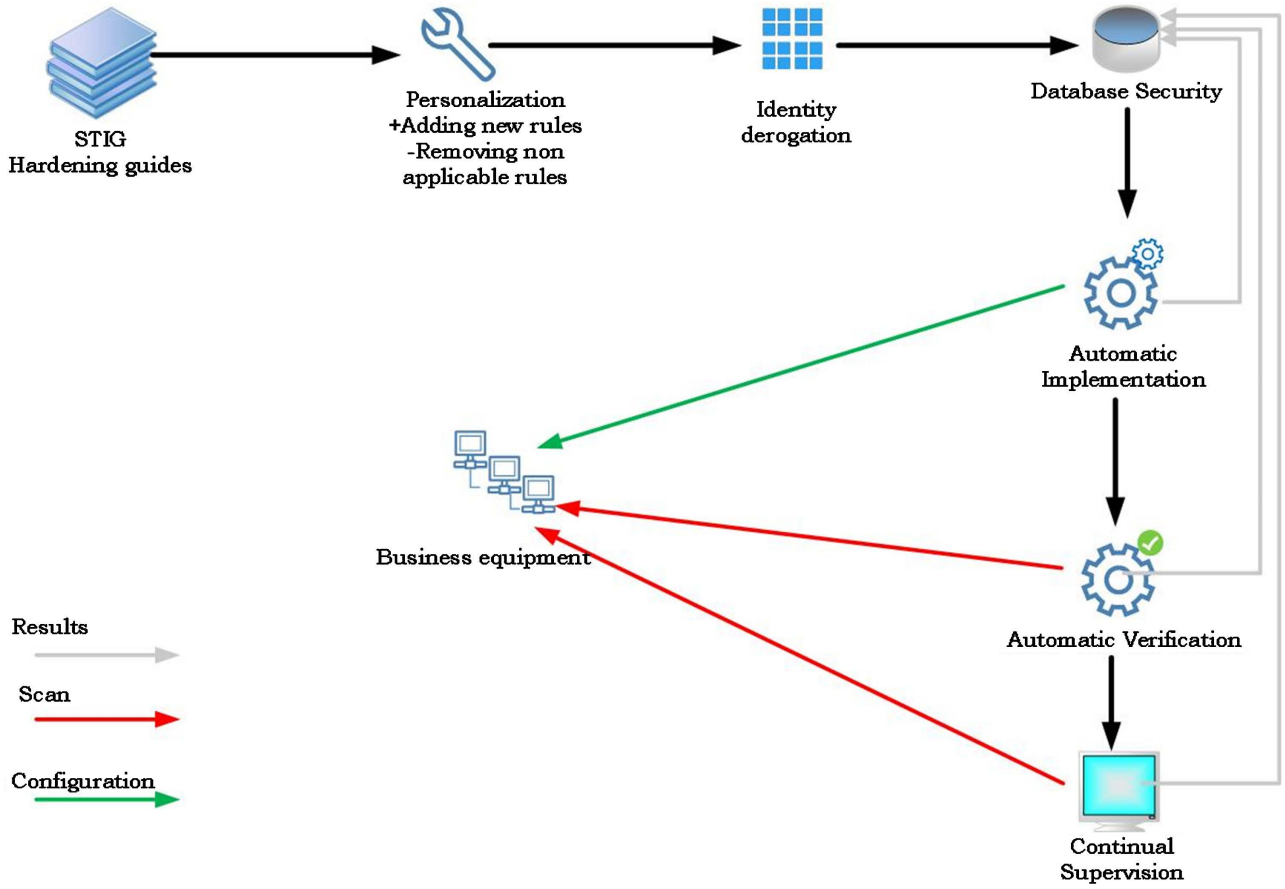


Figure 3. The hardening process [3].

- **Personalization**: your computers will be configured according to your needs because hardening allows you to change the default settings by the manufacturer of each particular software.
- **A better control**: at the time of the execution of the procedure, you simultaneously carry out a complete audit of your system, which leads you to obtain better control of it.
- **Confidentiality**: at the time of the execution of the procedure, you simultaneously carry out a complete audit of your system, which leads you to obtain better control of it.

The hardening of the systems will also make it possible to comply with best practices and avoid configuration errors.

2.6. Hardening Tools

1) Bastille

Bastille is an automatic hardening tool originally geared towards Red Hat and Mandrake Linux distributions. However, the bastille package provided in Debian (since Woody) has been modified to provide the same functionality for Debian GNU/Linux systems. It is a collection of PERL scripts that create a custom security configuration based on the answers provided by the administrator to a spe-

cific set of questions. It also performs an in-depth analysis of the system's current hardening level and its various security flaws, thereby reducing the chances of system compromise [2].

2) Microsoft SDL Threat Modelling Tool

Microsoft developed this tool with the aim of integrating threat modeling into the standard software development life cycle. The current version of the tool offers enhanced features such as better visualization and customization features, updated threat definitions, and more. Using this tool greatly reduces the effort required to identify security vulnerabilities and helps users take the necessary steps to counter them in the early stages of the SDL (software development lifecycle) [2].

3) Ansible

Ansible is an open source IT automation tool that automates provisioning, configuration management, application deployment, orchestration, and many other manual IT processes [2]. Unlike simpler management tools, with Ansible users (system administrators, developers, architects) can use automation features to install software, automate daily tasks, provision infrastructure, improve security and compliance, apply system patches and share their automated processes with the entire company.

4) CalCom Hardening Automation Suite

CalCom Hardening Automation Suite (CHS) [2] is a hardening automation platform designed to reduce operational costs and improve infrastructure security and compliance. CHS eliminates breakdowns and reduces curing costs by automating every step of the curing process.

5) Puppet

Puppet is a tool that helps manage and automate server setup. To use Puppet, it is necessary to define the desired state of the infrastructure systems to be managed [2]. This is accomplished by writing infrastructure code in Puppet's Domain Specific Language (DSL). Puppet code that will potentially be used with a wide range of devices and operating systems.

6) Chef Enterprise Automation Stack (EAS)

Chef Enterprise Automation Stack (EAS) is an automation platform enabling DevSecOps teams to build, deploy, manage and secure any application running on any infrastructure [5]:

- Align teams through a common set of tools and processes.
- Integrate conformance testing into each stage of the technology lifecycle.
- Ensure consistency, speed and security of application delivery on any infrastructure.

It is not specific to hardening but can be used for this purpose.

7) CIS-CAT Pro

CIS-CAT Pro Assessor assesses a system's cybersecurity posture against recommended policy settings [6]. The tool helps organizations save time and resources by supporting automated content with policy-setting recommendations based on globally recognized CIS benchmarks. The tool is kept in a location under the control of each member. Whether the organization uses virtual ma-

chines, in the cloud, in the network or on a local machine, CIS-CAT Pro helps ensure policy compliance. To allow for the greatest possible portability, CIS-CAT Pro is a Java application and requires a compatible JRE to run an assessment. Depending on the evaluation streams chosen by the organization, the JRE can reside on a target or a network drive.

8) Nessus

The Nessus tool is designed to scan a remote system and analyze various weak points that a malicious hacker can use to launch an attack. It is one of the most popular network scanners capable of checking vulnerabilities such as default password attacks, denial of service (DoS) attacks, etc. Versions after Nessus 3.0 also provide auditing functionality, helping to harden the system against known threats [2].

2.7. Comparaison between Existing Hardening Tools

A brief comparison of Hardening tools is shown in Table 1.

Table 1. Comparism table of existing Hardening tools.

	Chef	Puppet	Ansible	SaltStack
Architecture	Client/Server	Client/Server	Client/Server	Client/Server
Ease of installation	Average	Average	Very easy	Average
Language	Procedural: specifying how to perform a task.	Declarative: specifying only what to do.	Procedural: specifying how to perform a task.	Procedural: specifying how to perform a task.
Scalability	Scalable	Scalable	Scalable	Scalable
Management	Difficult because you have to learn Ruby DSL.	Difficult because you have to learn Puppet DSL.	Very easy	Very easy
Interoperability	High	High	High	High
Availability in the cloud	Amazon	Amazon/Azure	None	None
Protocole of communicattion	Kife tool	SSL	SSH	SSH
Environnement(s)	Ubuntu, Linux, Windows, Solaris …etc.	GNU/Linux, Mac OS X et Windows.	GNU/Linux, Mac OS X and Windows.	Linux, Unix and Windows.
Strong points	-Integrates well with Git, which provides strong version control; -A large collection of recipes is available.	-Strong community support from Puppet Labs; -Well-developed reporting mechanism.	-There is no need to install the agent on systems that require configuration; -YAML is extremely easy to understand and learn.	6Extremely easy to use once set up; -A good reporting mechanism that allows easy visualization of all operations.
Weak points	-Considerable learning time is required if one is not comfortable with Ruby.	-For performing advanced tasks, a good knowledge of Ruby is required; -The main server does not have much control.	-Execution speed is often slower than other tools; -YAML is not as powerful as most other languages.	-The installation phase is a little more difficult; -A relatively new web interface that is much less developed than other tools.

3. Materials, Tools and Methods

3.1. Material

The architecture of our supervised networks consists of servers with the operating system Red Hat version 7.x and Windows Server 2019, Cisco brand routers and switches, laptop computers

3.2. Development Tool, Libraries and Programming Languages Used

3.2.1. Justification of the Choices

The type of tool that we decided to make to meet the needs is a web application developed using the Python language using the Django framework. To connect remotely to the equipment, we used the SSH protocol using the Paramiko Python library.

The choice of this library is justified by [6]:

- Lines of code are relatively short;
- The python code is of low complexity to understand;
- The python code is of low complexity to understand;
- The ease of Paramiko to integrate into an application.

3.2.2. The Development Tools

Table 2 summarizes the main development tools used [7]:

Table 2. Development tools used.

Name of the tool	Description and functionality
Pycharm	JetBrains' PyCharm is a comprehensive integrated development environment that includes a highly automated toolchain to improve developer productivity. As the name suggests, the PyCharm IDE targets Python programmers.
Visual studio code	Visual Studio Code is a simplified code editor, which is free and developed in open source by Microsoft. It works on Windows, macOS and Linux. There is support for several programming languages, including C, C#, C++, CSS, HTML, Java, JavaScript, JSON, Markdown, PHP, Powershell, Python, TypeScript, YAML.
GNS3	GNS3 (Graphical Network Simulator) is an open source software that allows you to simulate complex networks while being as close as possible to the operation of real networks. This software provides an intuitive graphical user interface for designing and configuring virtual networks.
VMWare	VMware is a virtualization and cloud computing software provider. With VMware server virtualization, a hypervisor is installed on the physical server to allow multiple virtual machines (VMs) to run on the same physical server.
Git	Git is a development tool used for source code management. It is a free and open-source version control system used to efficiently manage small to very large projects.

3.2.3. Programming Language

Table 3 summarizes the main programming languages used [7]:

Table 3. Programming languages used.

Name of the language	Description and functionality
Python	Python is an interpreted, cross-paradigm, cross-platform programming language. It promotes structured, functional and object-oriented imperative programming. The latter is equipped with strong dynamic typing, automatic memory management by garbage collection and an exception management system; works on most computer platforms, from smartphones to mainframes. It is designed to maximize programmer productivity by offering high-level tools and easy-to-use syntax.
Django	Django is an open-source python framework dedicated to web 2.0 development. It's "The web framework for perfectionists under pressure". It is oriented for developers who need to produce a solid project quickly. As it is always complicated to start from scratch, Django offers a solid project base.
HTML	It is a language used to compose web pages. We speak of markup language and not of programming language, because the purpose of HTML is to frame the different elements present in a page (images, titles, paragraphs, etc.) with tags to allow them to be formatted secondarily. (via a style sheet) and to make sense.
CSS	CSS stands for Cascading Style Sheets. It is a style language whose syntax is extremely simple but its performance is remarkable. Indeed, CSS is concerned with the formatting of content embedded with HTML.
JavaScript	It's a programming language that allows you to create dynamically updated content, control multimedia content, animate images, and everything else you can. The JavaScript language is mainly used to improve the ergonomics of a website and/or a user application interface.

3.2.4. Libraries

Table 4 summarizes the main libraries used [7].

3.3. Method

3.3.1. Functionalities

The main features of our app are:

- Harden configurations of one or more devices: the user performs a set of hardening procedures on the devices;
- Verify that the hardening configurations have been executed correctly: the user executes the commands to verify the different hardening procedures;
- Manage hardening controls: user add, modify or remove one or more controls;
- Manage procedures: user adds procedures;
- Manage procedures: user adds procedures;

3.3.2. Nonfunctional Analysis

The technical constraints to which the application is subject are as follows:

Table 4. Libraries used.

Name of the library	Description and functionality
SQLite	SQLite is a library written in the C language that offers a relational database engine accessible by the SQL language. SQLite largely implements the SQL-92 standard and ACID properties. Unlike traditional database servers, such as MySQL or PostgreSQL, its particularity is not to reproduce the usual client-server scheme but to be directly integrated into programs.
Paramiko	Paramiko is used to program the sending of commands to network equipment via the SSH protocol. Using this library, users send commands that the network device will execute as if they had been entered into its CLI console from a keyboard directly attached to it. The result of these commands will be retrieved by the Python script which can display them on the administrator's screen.
Ajax	Ajax is mainly used to bring interactivity within web pages while saving server resources. Indeed, Ajax allows to communicate with the server using Javascript code in the background while the page is displayed on the screen. Thus the content of the page can be modified without it being necessary to transit and display the entire page. Ajax is particularly used for updating forms and shopping carts on most websites.
Boostrap	Bootstrap is a free and open-source web development framework. It is designed to ease the process of developing responsive and mobile-focused websites by providing a collection of syntaxes for design patterns.

- The interfaces of our application must be ergonomic and user-friendly;
- Availability: our application must be available at all times for use by entitled users, and must be easily accessible via any device;
- Security: Our application contains personal and sensitive information, so it must comply with the rules relating to the security of computer systems;
- Reliability: The results provided by the application must be reliable and effectively reflect the state of the database at the time of its interrogation, that is to say during the update of the data.

3.3.3. Flowchart of the Methodological Steps

The approach adopted for the realization of our solution is illustrated in Figure 4.

3.3.4. Tool Modelization

1) Use case diagram

The use case diagram in Figure 5 of our solution looks like this.

2) Sequence diagrams

Sequence diagrams for performing and verifying curing are shown in Figure 6 and Figure 7, respectively.

3.3.5. Class Diagram

The class diagram of our solution is shown in Figure 8 below.

3.4. Conception Tool

3.4.1. Challenges

The main challenges to be overcome in order to be able to design an automatic hardening management tool respecting the methodology described above are the following:

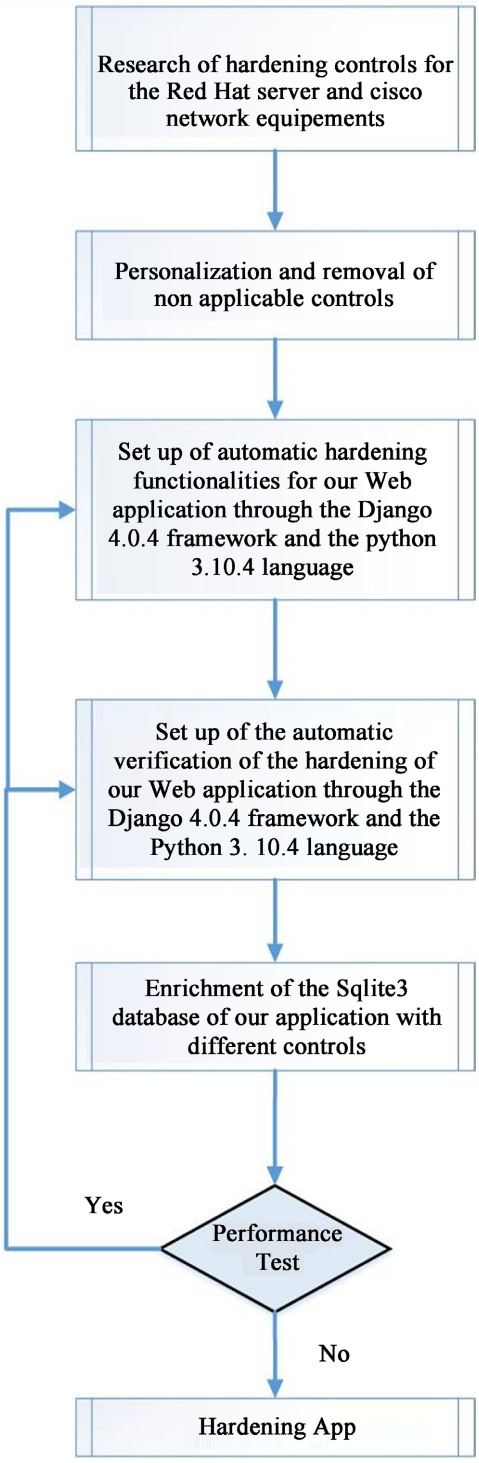


Figure 4. Flowchart of methodological steps.

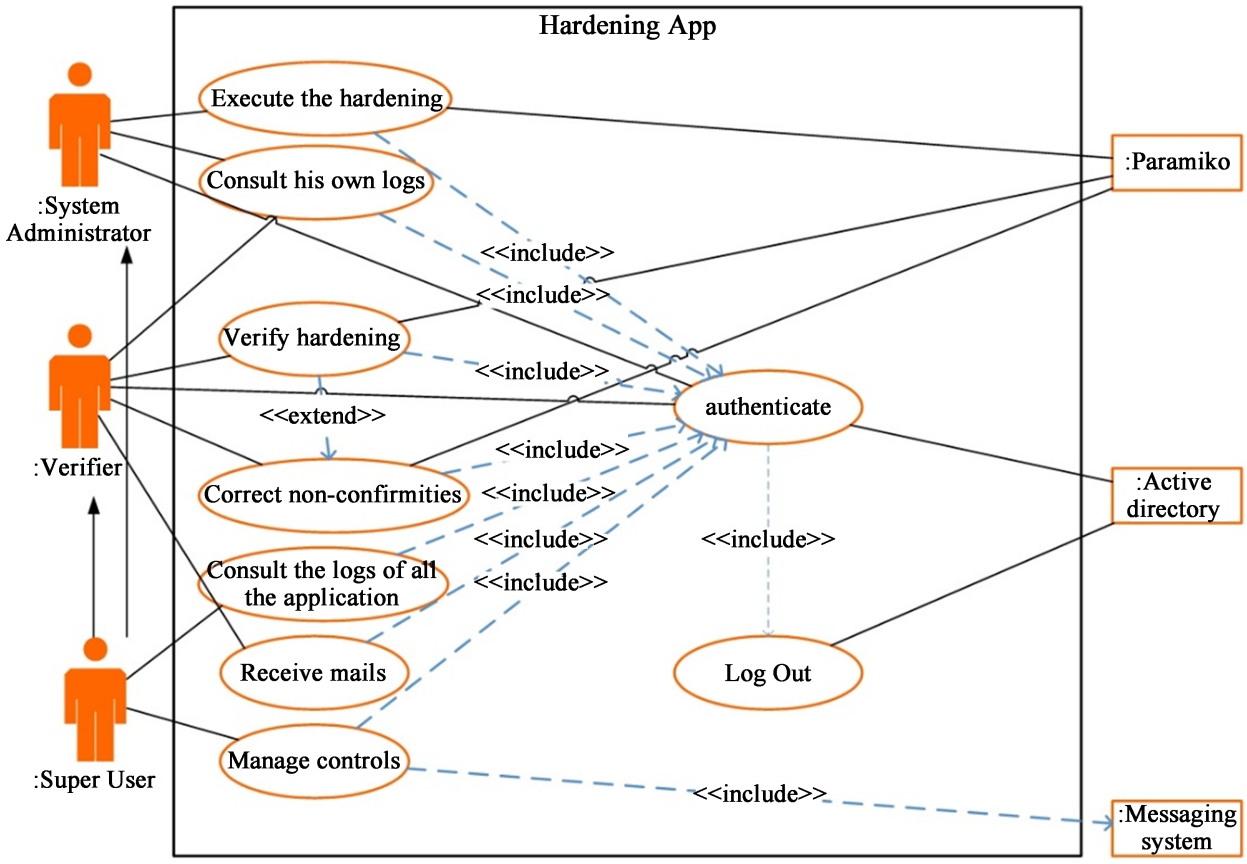


Figure 5. HardeningApp use case diagram.

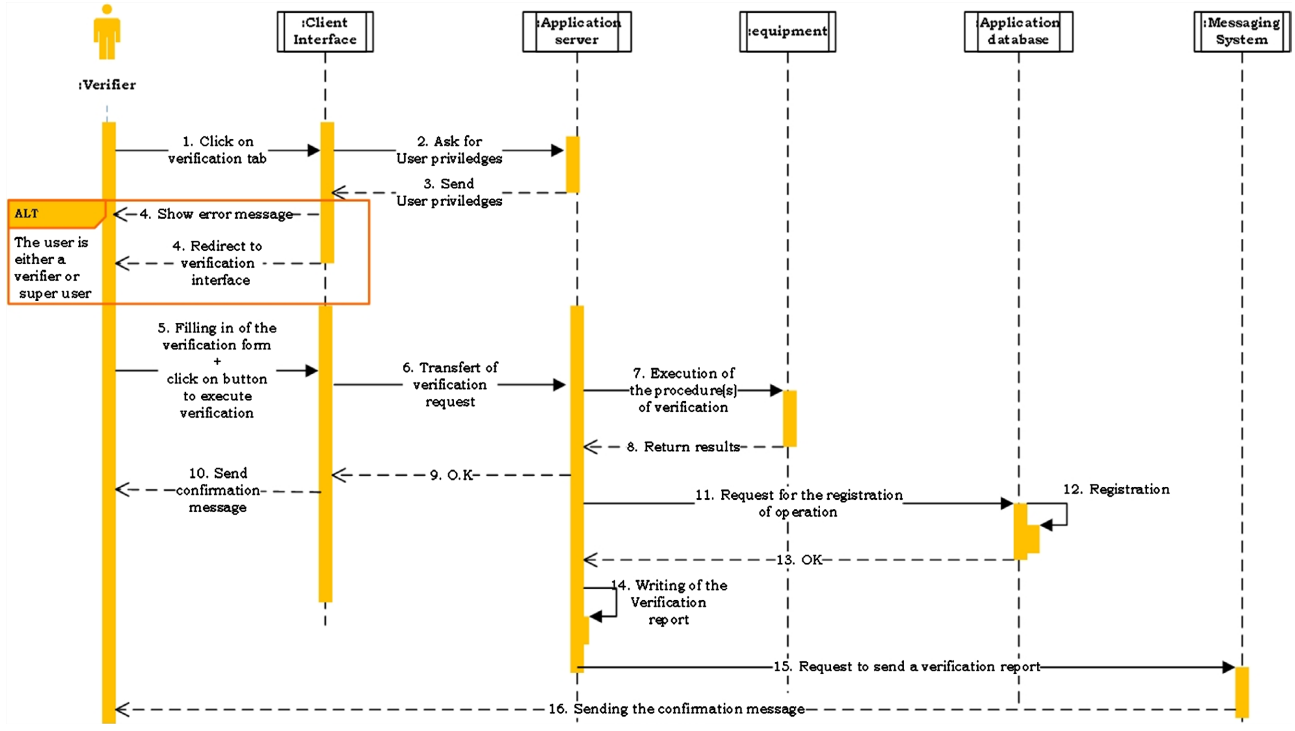


Figure 6. Sequence diagram of the "Execute hardening" use case.

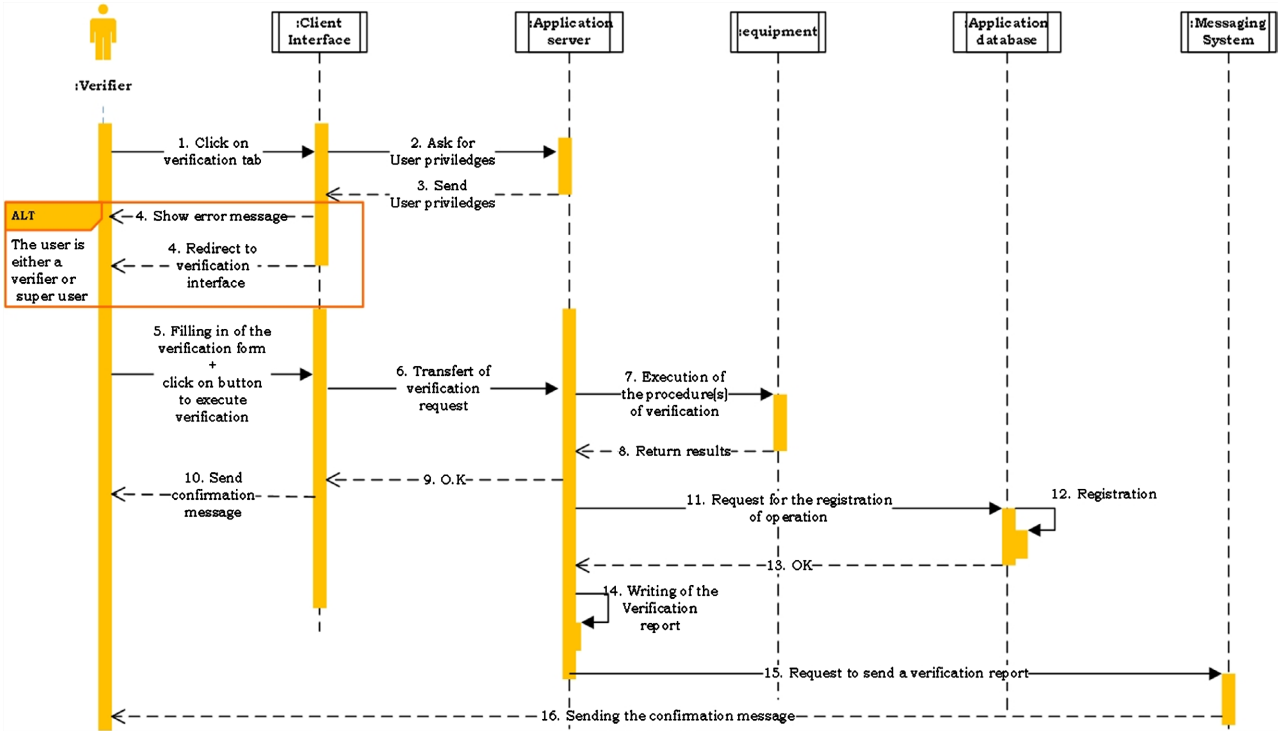


Figure 7. Sequence diagram of "verify hardening" use case.

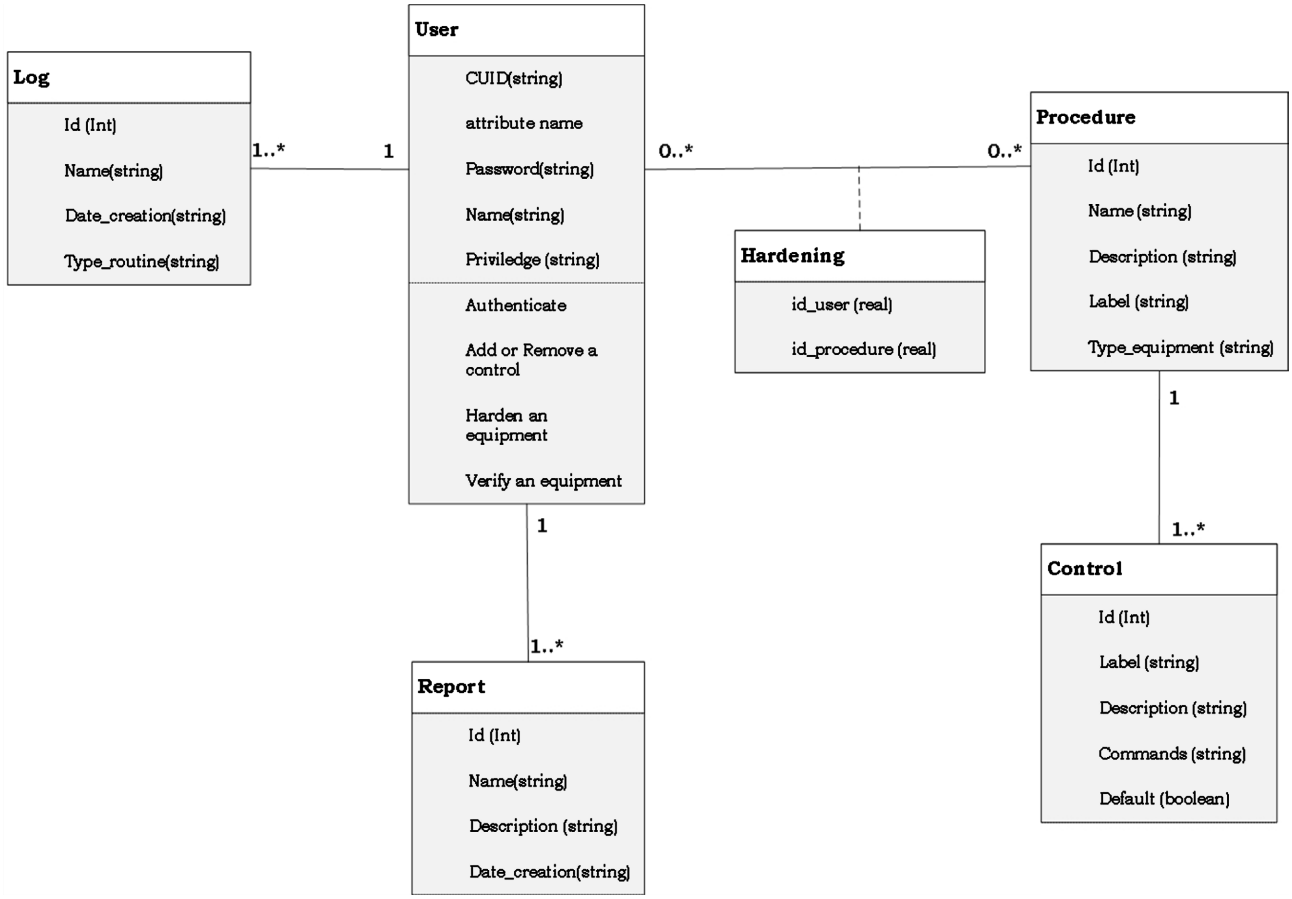


Figure 8. Class diagram.

- Create a hardening tool capable of hardening the configurations of several servers and network equipment at the same time and this in a reasonable time;
- Create a tool that checks that the configurations have been executed on one or more network devices at the same time in a reasonable time;
- Realize a tool that is able after the execution of the verification process to recover the controls that are not compliant and bring them back to the expected value;
- Send a verification report in.csv format by email after the execution of the hardening verification;
- Send confirmation messages by email after the execution of each routine;
- Allow to take a.csv file containing the parameters of different controls and add them to the list of controls of the application.
- Edit and delete controls;
- Update curing procedures.

3.4.2. System Architectures

1) Overall architecture of the solution

We propose to implement an architecture consisting of the following elements: an application server in which the backend, the webview and our paramiko library are hosted; an email server that hosts the API that allows us to send emails; the Active directory where all the user information of our tool and the database are stored.

Figure 9 below illustrates the overall architecture of our solution.

Functioning

- The active directory communicates with the application server via keycloak to authenticate users.
- The user via HTTP calls communicates with the webview of the application to be able to perform all the routines offered by the application.
- The webview interacted with the backend through http calls as well.

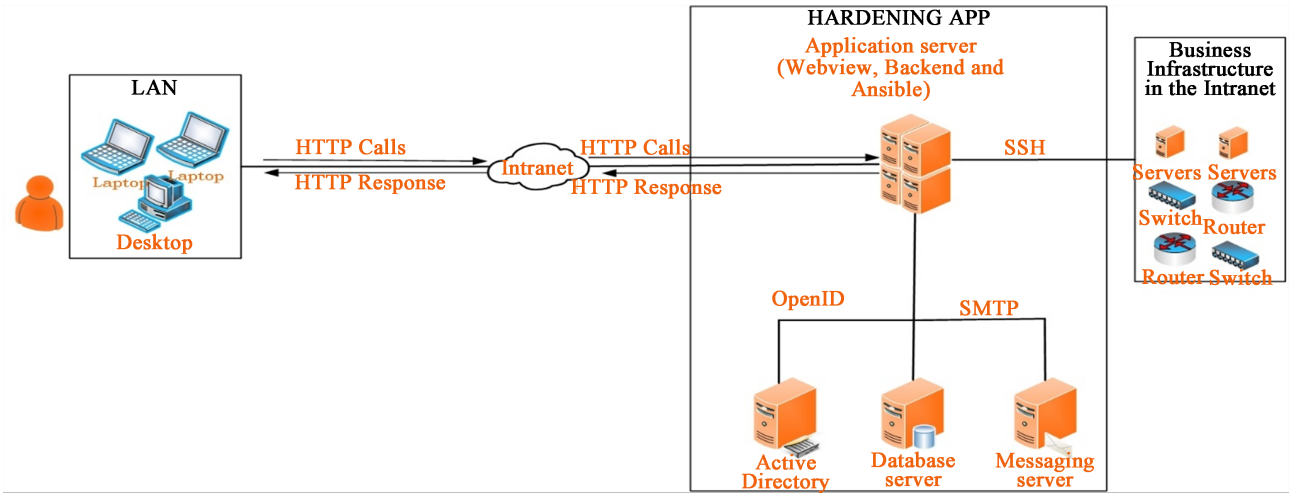


Figure 9. Overall Solution Architecture.

- Once the requests are sent to the backend, the latter is responsible for executing them.
  - If it is a hardening or a verification, the backend is responsible for executing the various commands on the remote equipment thanks to paramiko which sends the commands that the equipment will execute.
  - Whether adding, removing or modifying a control; once the query is passed to the backend, it executes it and asks the database to save the changes.
- Once the requests are made, emails are sent to the users. This work is done by the Orange Cameroon Swagger Microservices API consumed by our application.

2) Logical solution architecture

Figure 10 below presents the software architecture of our solution. It highlights the software and protocols we used along with their versions.

4. Results and Comments

4.1. Structural Architecture of the Application

Following the methodology presented above, we were able to set up an automatic hardening management tool called HardeningApp. The structural architecture of it is shown in Figure 11 below.

Our HardeningApp is structured as follows:

- The authentication interface: it is the entry page of the application on which the user authenticates;
- The home page: on this page the user to the application supervision data;

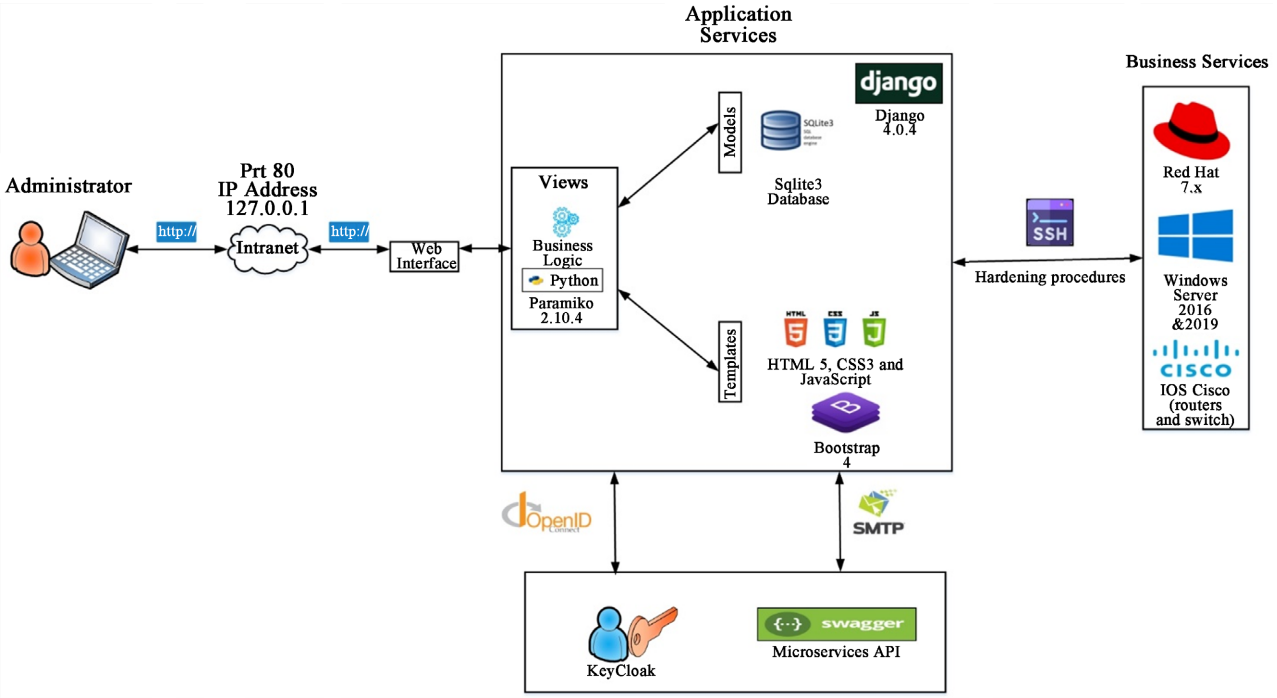


Figure 10. Logical architecture of our solution.

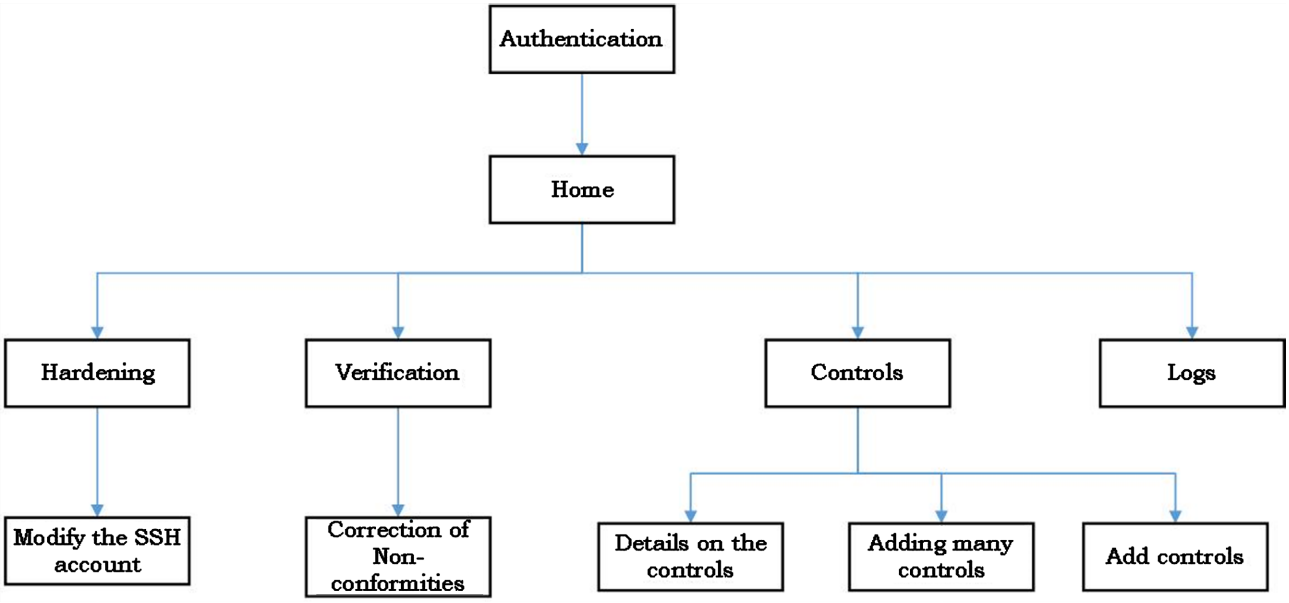


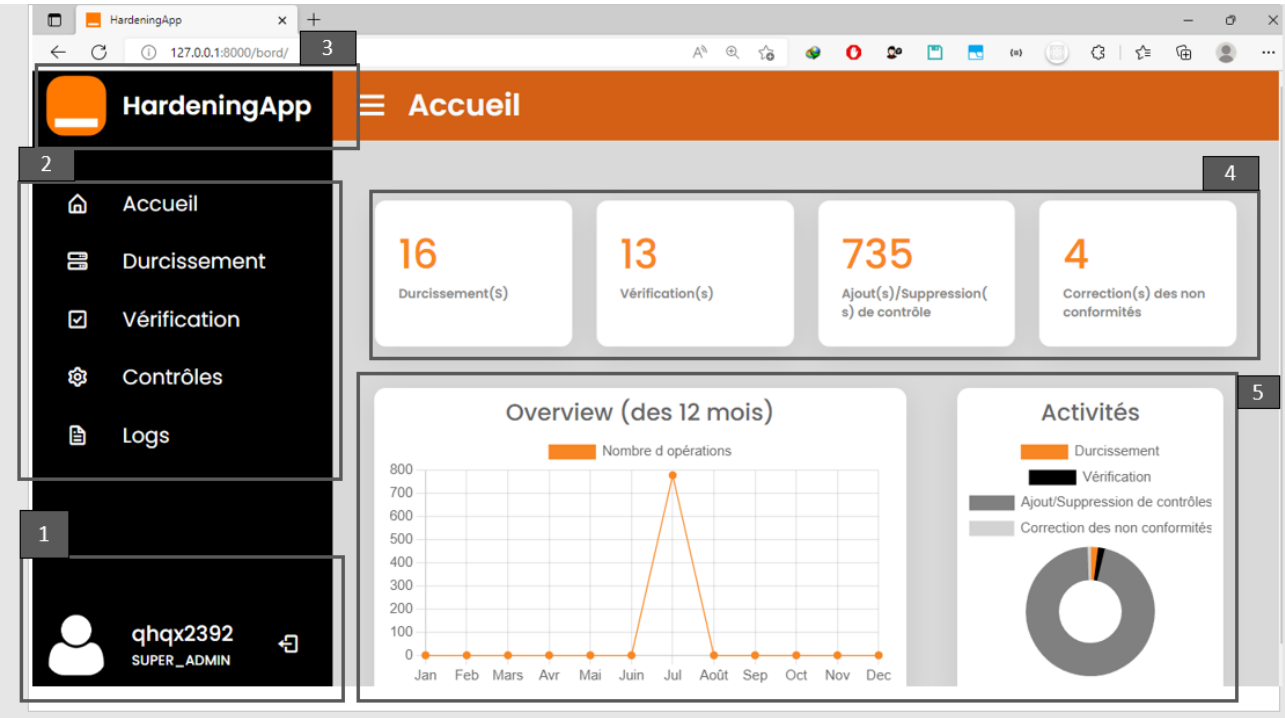
Figure 11. HardeningApp structural architecture.

- The hardening interface. Here the user enters into the application the parameters corresponding to the equipment to carry out the hardening;
- The verification interface. In this page, the user checks that the hardening has been carried out by entering the parameters of the equipment on which he wishes to check the hardening. This has a sub-page that can be accessed once the verification process has been completed:
  - The non-conformance correction interface where the user is presented with all control commands that have not been executed correctly and the user has the possibility to re-execute them.
- The control interface. On this page the user accesses all the controls registered in the application. He can modify them, delete them or add other controls. This page consists of the following sub-pages:
  - The detail interface: on which the user has access to all the characteristics of the selected control and can thus modify them;
  - The interface for adding a control. The user can add a control by filling in the different characteristics of the control in the add form;
  - The interface for adding several controls. The user adds multiple controls by importing a CSV file that includes all the characteristics of the controls to be added.
- The log interface. On this page the user has an overview of all the activities carried out on the application. In the case of hardening verification, he can download the verification report.

4.2. Tool Presentation

4.2.1. Home Page Interface

If the CUID and password are correct, the user goes to the home page. **Figure 12** illustrates the login page.



Legend: 1 This tab presents the active account on the application. It includes the CUID of the logged in person, their role and the log out button. 2 This section corresponds to the menu that gives us access to the other pages of the application. 3 On this part of the page we find the name of the application and its logo. 4 In this section, for each type of operation that can be performed on the application, the number of executions is entered. 5 In this part of the application, we find a graph which allows you to know the number of operations carried out on the application each month.

Figure 12. HardeninApp home interface.

4.2.2. Hardening Interface

Once the user clicks on the menu hardening button from the home page, he is taken to the hardening page represented by **Figure 13** and **Figure 14**.

Once the execution of the various hardening checks is complete, a confirmation email is sent to the user. **Figure 15** shows an example email after performing hardening on HardeningApp.

Depending on the user or the device, it may be necessary to change the ssh account. To do this, the user clicks on the edit SSH account button and is then redirected to the edit account page shown in **Figure 16**.

4.2.3. Verification Interface

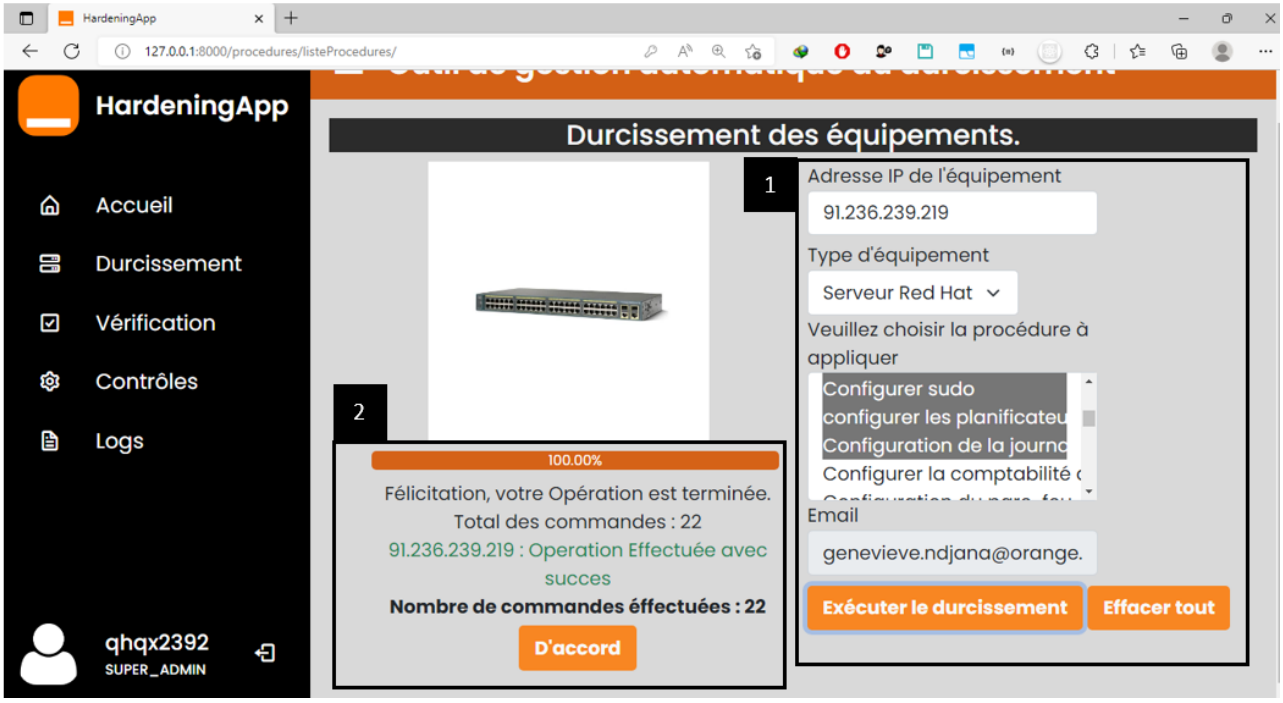
From the home page or from any other page, the user click on the verification button and he is directed to the verification page illustrated by the **Figure 17** and **Figure 18**.

1) This part of the verification page is identical with the exception of the Execute button the hardening in this case is Exexcute the verification.

2) This page section consists of:

- The loading bar that indicates execution percentage of the verification commands in a scalable manner;

Figure 13. Hardening interface 1.



Legend: 1 This part represents the form to be entered to carry out the hardening on an equipment. It consists: IP address fields: the user can enter one or more IP addresses of equipment of the same type; From the equipment type field: the user chooses between the different types of equipment (Red Hat Server, Windows Server, Router and Switch); From the procedure field: the user chooses the procedures to be carried out among the different procedures corresponding to his type of equipment; From the email field: the user enters his email address; Execute hardening button: it allows to start the hardening process on the equipment whose IP address has been entered; Clear All button to clear all previously filled in fields; 2 This part of the hardening page features: The loading bar that shows in a scalable way the percentage of execution of the hardening commands; The total indications of orders and number of orders carried out which respectively represent the total number of orders to be executed and the number of orders which have actually been carried out; The OK button to stop the hardening process and possibly start another one.

Figure 14. Hardening interface 2.

Figure 15. Example of email sent after performing a hardening.

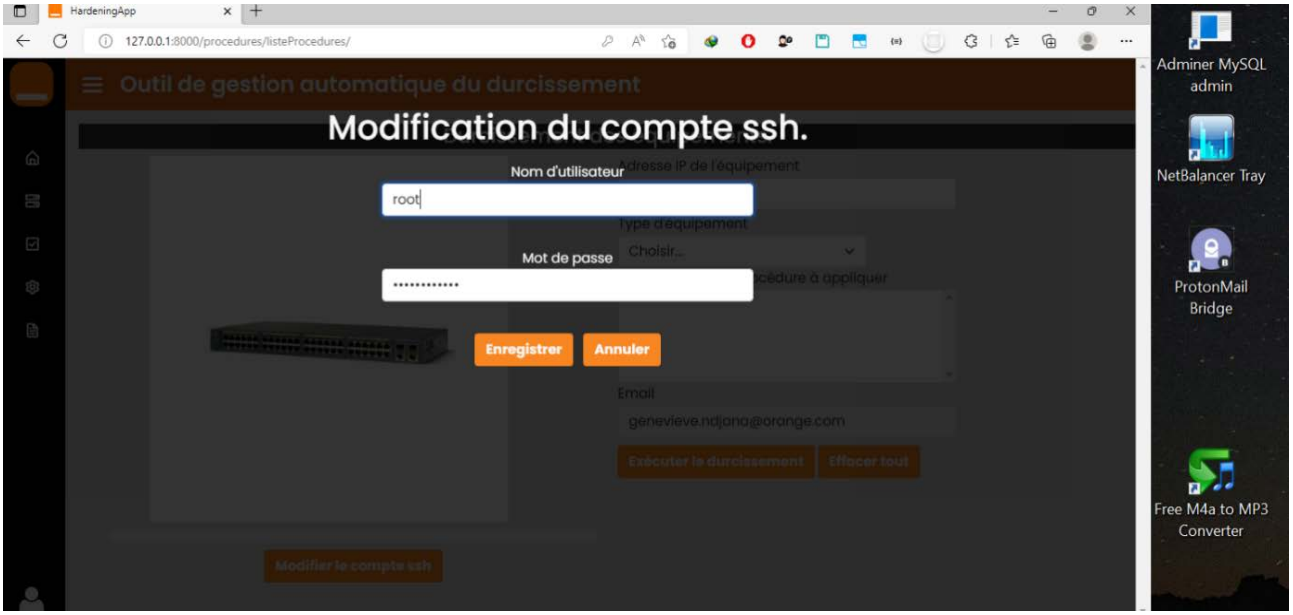


Figure 16. SSH account edit interface.

- The total indications of commands, the number of commands run and the number of non-conformities which respectively represent the total numebr of commands to be executed, The number of command that has been run and the number of commands that has not been executed with respect to the procedure;
- The Go button at the non-conformities gives access to the page for correcting the non-conformities.

Once the execution of the verification process for the hardening is finished, A

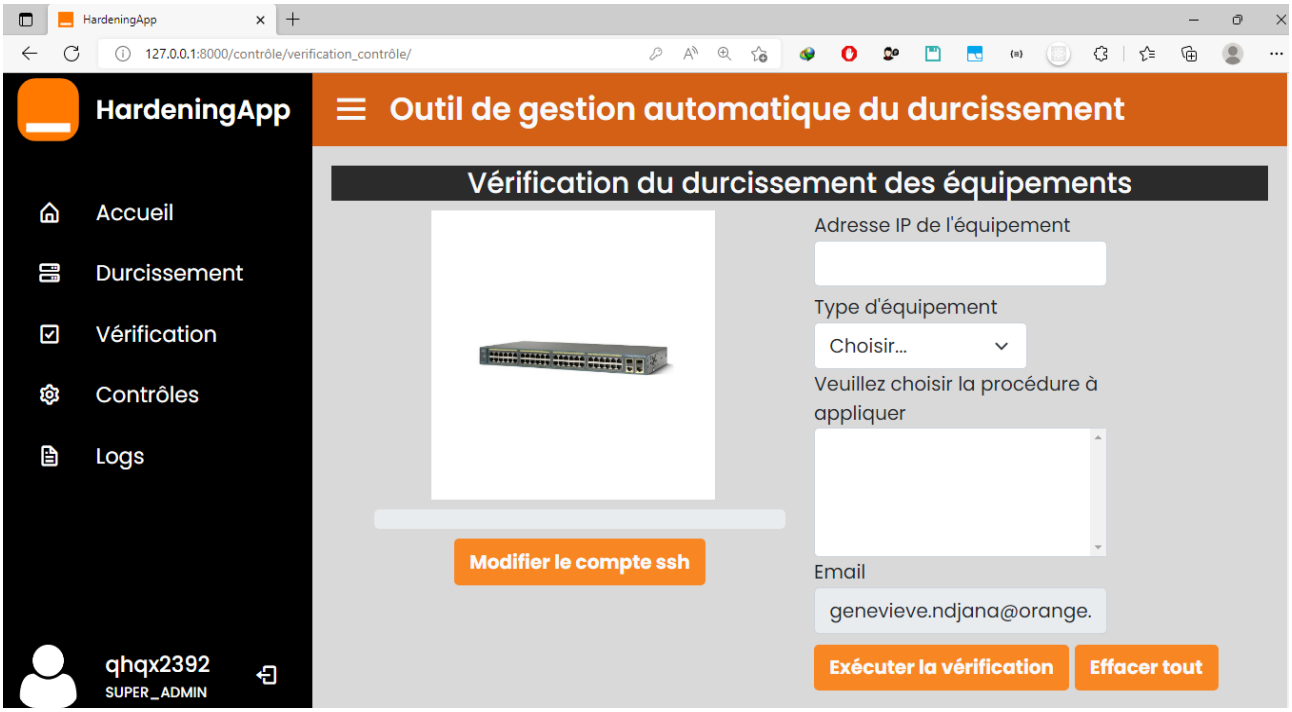


Figure 17. Verification interface 1.

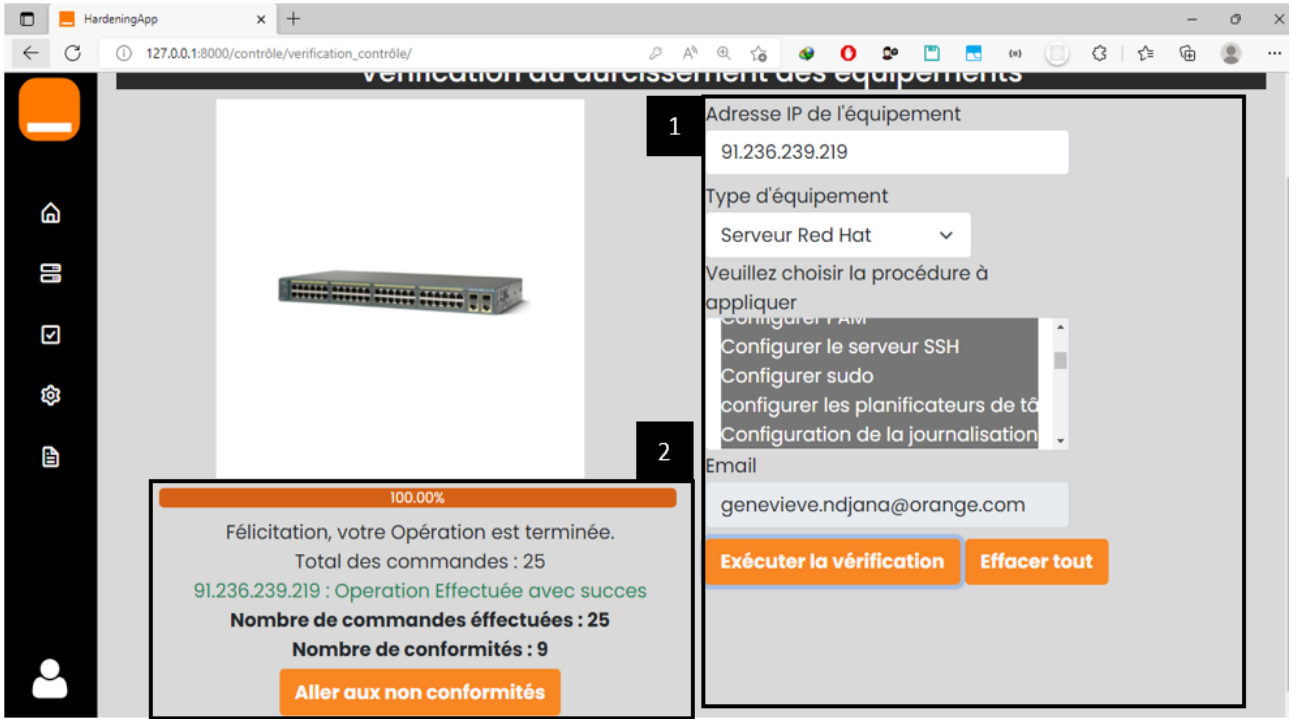


Figure 18. Verification interface 2.

confirmation mail is send containing a verification report. Figure 19 represent an example of mail after the realization of the hardening verification on HardeningApp.

On the following Figure 20 we have an example of verification report:

Répondre Répondre à tous Transférer MI

ven. 08/07/2022 09:57

HardeningApp <noreply@orange.cm>

Rapport de vérification des configuration de durcissement

À NDJANA Genevieve

Cc NDJANA Genevieve

En cas de problème lié à l'affichage de ce message, cliquez ici pour l'afficher dans un navigateur web.
Cliquez ici pour télécharger des images. Pour protéger la confidentialité, Outlook a empêché le téléchargement automatique de certaines images dans ce message.

Bonjour!

Veuillez retrouver en PJ le rapport de vérification des configurations de durcissement que vous venez d'éffectuer.

Cordialement,

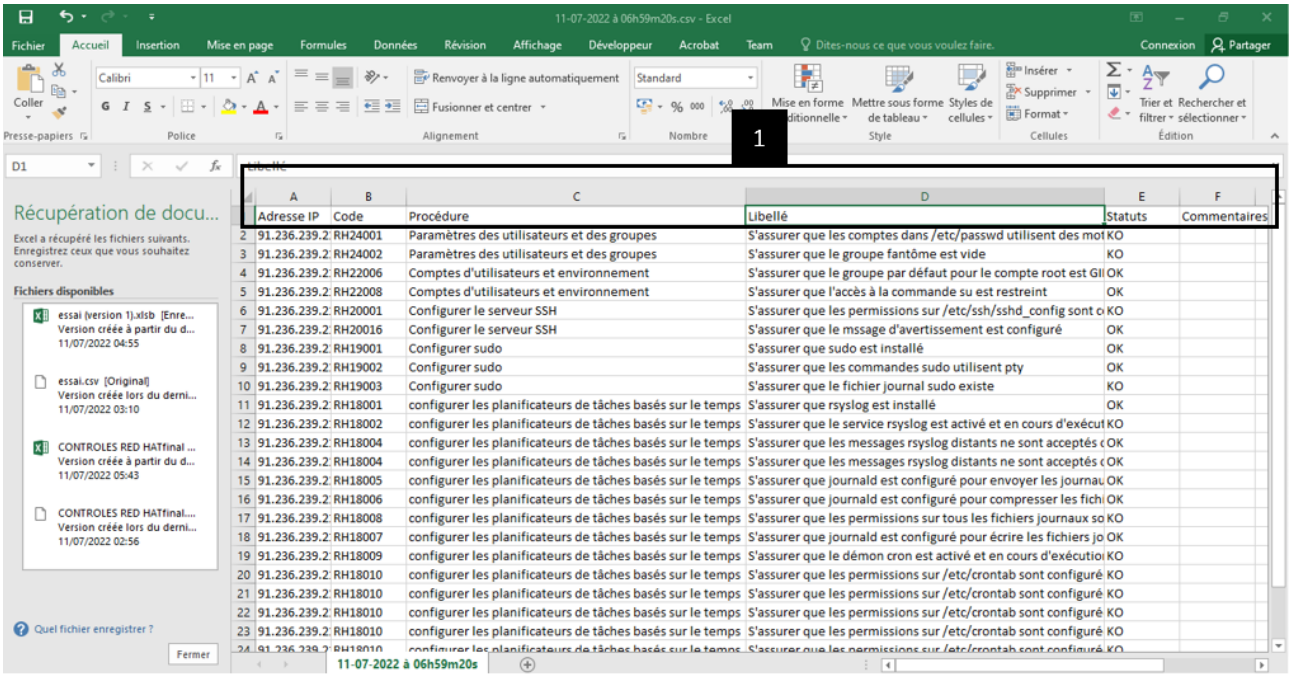
HardeningApp.

Adresse IP : 172.21.64.60

Type : Serveur Red Hat

Le : 08-07-2022 à 09h56m52s

Figure 19. Example of mail sends once verification is finished.



Adresse IP	Code	Procédure	Libellé	Statuts	Commentaires
91.236.239.2	RH24001	Paramètres des utilisateurs et des groupes	S'assurer que les comptes dans /etc/passwd utilisent des mot	KO	
91.236.239.2	RH24002	Paramètres des utilisateurs et des groupes	S'assurer que le groupe fantôme est vide	KO	
91.236.239.2	RH22006	Comptes d'utilisateurs et environnement	S'assurer que le groupe par défaut pour le compte root est Gl	OK	
91.236.239.2	RH22008	Comptes d'utilisateurs et environnement	S'assurer que l'accès à la commande su est restreint	OK	
91.236.239.2	RH20001	Configurer le serveur SSH	S'assurer que les permissions sur /etc/ssh/sshd_config sont c	KO	
91.236.239.2	RH20016	Configurer le serveur SSH	S'assurer que le mssage d'avertissement est configuré	OK	
91.236.239.2	RH19001	Configurer sudo	S'assurer que sudo est installé	OK	
91.236.239.2	RH19002	Configurer sudo	S'assurer que les commandes sudo utilisent pty	OK	
91.236.239.2	RH19003	Configurer sudo	S'assurer que le fichier journal sudo existe	KO	
91.236.239.2	RH18001	configurer les planificateurs de tâches basés sur le temps	S'assurer que rsyslog est installé	OK	
91.236.239.2	RH18002	configurer les planificateurs de tâches basés sur le temps	S'assurer que le service rsyslog est activé et en cours d'exécut	KO	
91.236.239.2	RH18004	configurer les planificateurs de tâches basés sur le temps	S'assurer que les messages rsyslog distants ne sont acceptés	OK	
91.236.239.2	RH18004	configurer les planificateurs de tâches basés sur le temps	S'assurer que les messages rsyslog distants ne sont acceptés	OK	
91.236.239.2	RH18005	configurer les planificateurs de tâches basés sur le temps	S'assurer que journald est configuré pour envoyer les journau	OK	
91.236.239.2	RH18006	configurer les planificateurs de tâches basés sur le temps	S'assurer que journald est configuré pour compresser les fich	OK	
91.236.239.2	RH18008	configurer les planificateurs de tâches basés sur le temps	S'assurer que les permissions sur tous les fichiers journaux so	KO	
91.236.239.2	RH18007	configurer les planificateurs de tâches basés sur le temps	S'assurer que journald est configuré pour écrire les fichiers jo	OK	
91.236.239.2	RH18009	configurer les planificateurs de tâches basés sur le temps	S'assurer que le démon cron est activé et en cours d'exécutio	KO	
91.236.239.2	RH18010	configurer les planificateurs de tâches basés sur le temps	S'assurer que les permissions sur /etc/crontab sont configuré	KO	
91.236.239.2	RH18010	configurer les planificateurs de tâches basés sur le temps	S'assurer que les permissions sur /etc/crontab sont configuré	KO	
91.236.239.2	RH18010	configurer les planificateurs de tâches basés sur le temps	S'assurer que les permissions sur /etc/crontab sont configuré	KO	
91.236.239.2	RH18010	configurer les planificateurs de tâches basés sur le temps	S'assurer que les permissions sur /etc/crontab sont configuré	KO	

Legend: Shows the verification report heading which is a file in the CSV format. It is made up of: The IP address of the device on which the verification has been done; Implemented control codes; Procedures belonging to controls; Expression of some implemented controls; The status K.O or O.K with the condition that the control commands are well executed or not; Comments to be filled by ITN Security teams.

Figure 20. Example of verification report.

4.2.4. Terminal of the Server

Once the execution process of hardening and of verification are run, respectively the execution of hardening commands and verification commands begins at the same time at the server level. Figure 21 and Figure 22 show us the commands that run there during hardening and hardening verification, respectively.

4.2.5. Correction of Non-Conformities Interface

Once the execution of the hardening verification process has been completed, the user has the possibility of executing once again the commands which during the realization of the hardening have not been executed correctly. To do this, the

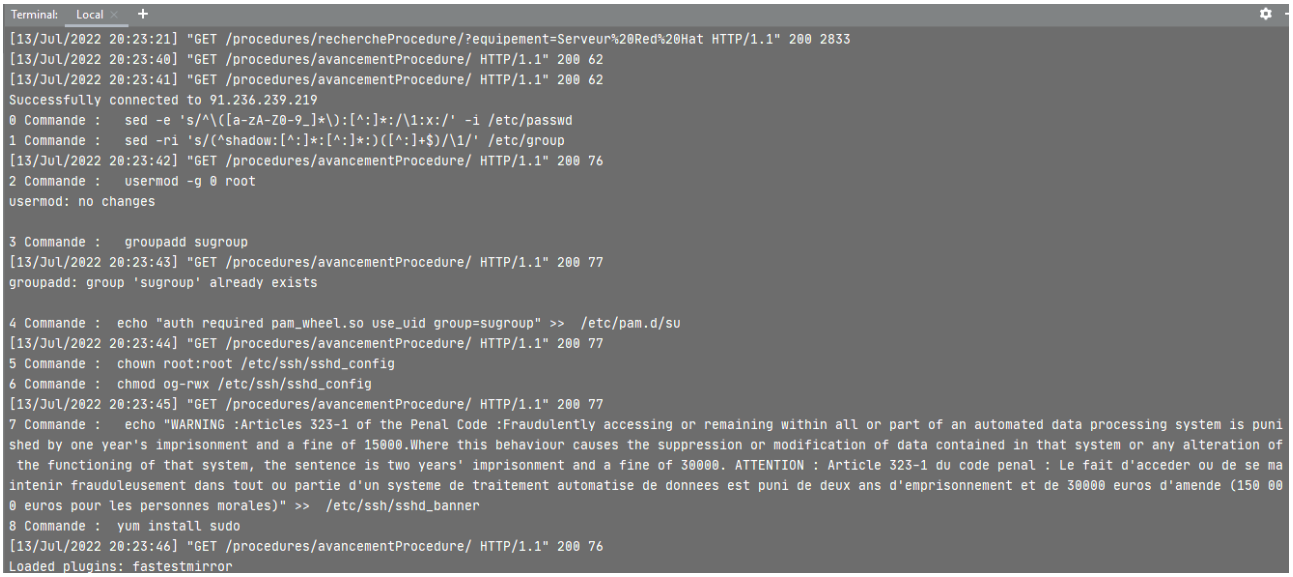


Figure 21. Terminal of a server when running hardening.

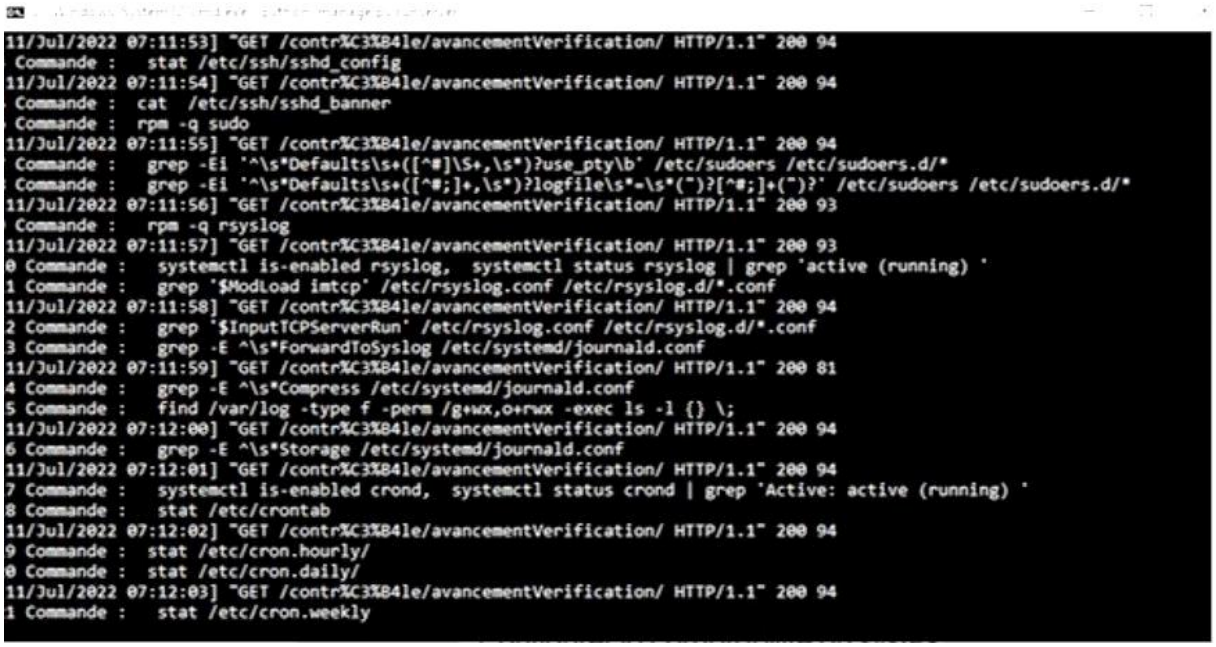


Figure 22. Terminal when running verification.

user clicks on the Go to non-conformities button. The page it is redirected to is as shown in Figure 23.

4.2.6. Control Interface

On this page, the user has access to all the controls registered in the application as shown in Figure 24.

To add several controls at the same time as shown in Figure 25 and Figure 26, the user must click on the button and upload a CSV file containing all the parameters of the controls to be added the following figures illustrate the process of adding several controls.

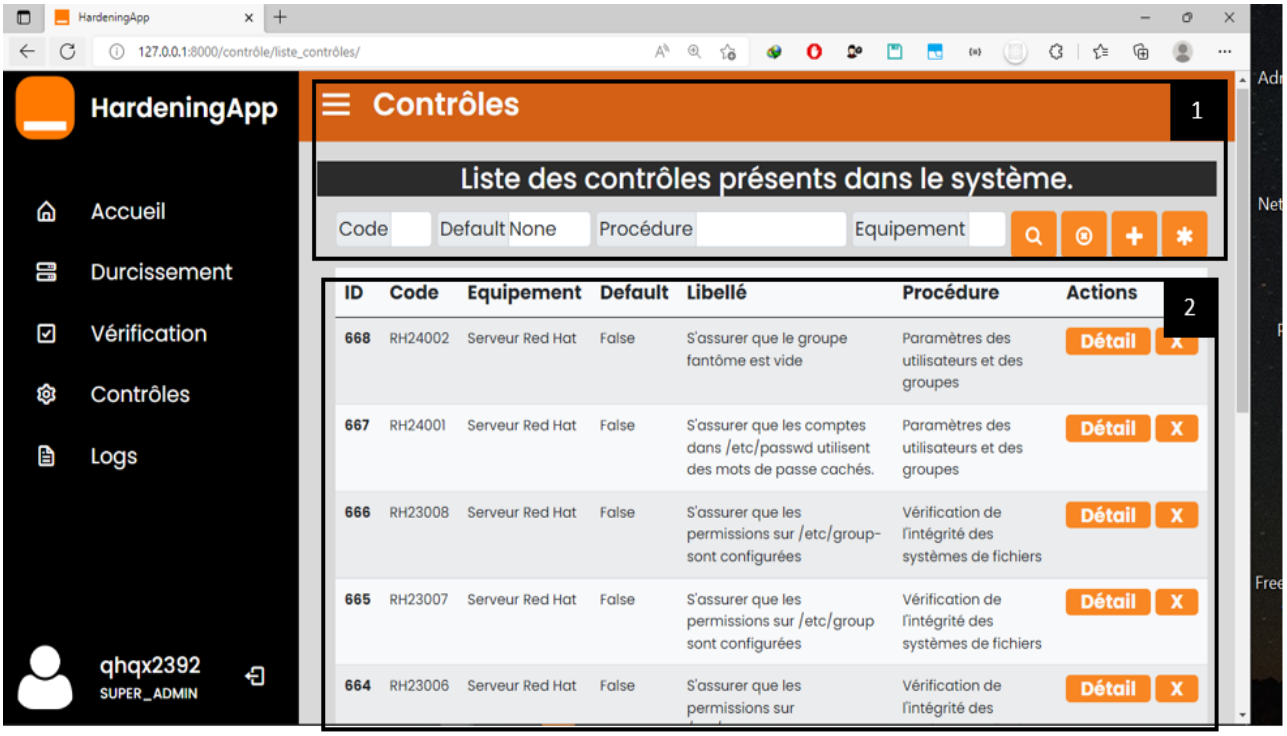
Once the controls have been added, they can be found in the list of controls present in the application.

4.2.7. Log Interface

Once the user clicks on the log button on the menu, he is redirected to the page shown in Figure 27.

Legend: 1 This part represents the form to be entered to execute the commands for non-compliant controls on one or more equipment. It consists: -IP address fields: it includes the IP address(es) filled in during the verification already filled in; -From the type of equipment field: it includes the type of equipment chosen during the verification already filled in; -From the commands field: it contains all the commands of the non-compliant controls; -The Execute button: it allows you to start the process of executing commands on the equipment whose IP addresses are entered; -Clear all button to clear all previously filled in fields; 2 Consist of: -The loading bar that shows in a scalable way the percentage of execution of the hardening commands; -The total indications of orders and number of orders carried out which respectively represent the total number of orders to be executed and the number of orders which have actually been carried out; -The OK button to stop the hardening process and possibly start another one.

Figure 23. Non conformity correction interface.

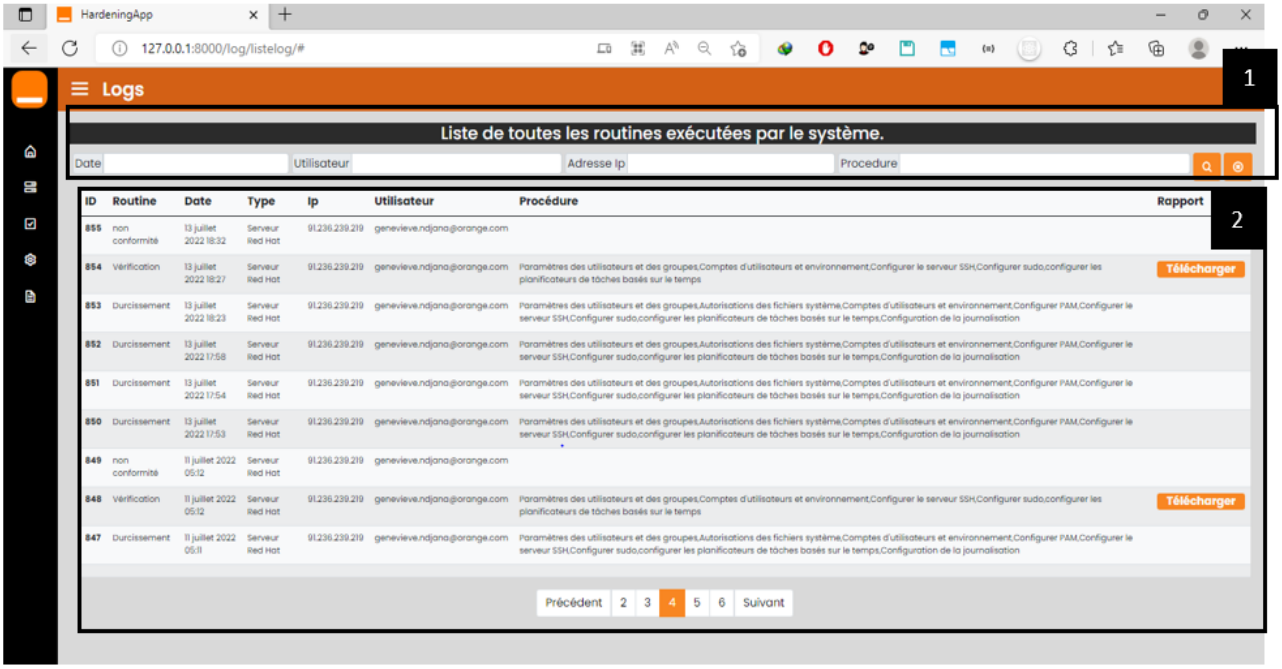


Legend: 1 On this tab, the user can search controls based on code, equipment type, procedure and default setting. It is also possible to add one control or several at a time. 2 This part is made up of the list that groups together all the controls and their parameters.

Figure 24. HardeningApp control interface.

Figure 25. Interface for adding multiple controls.

Figure 26. AV2106 control in the list of controls.



Legend: 1 On this tab, user can search logs based on date, IP address, procedure and user. 2 This part consists of the list which includes all the logs and their parameters, namely the date of the operation, the user who carried it out, the IP address of the equipment on which it is carried out, the procedures used, the type of routine (hardening, verification and addition/removal/modification of a control), the type of equipment and possibly a verification report that the user can download.

Figure 27. HardeningApp Log Interface.

4.2.8. Checking Configuration Files

Once the hardening configurations had been executed and verified, we executed a script on the test server which returned the configuration files that we browsed.

Figure 28 proves that the "Make sure address space randomization (ASLR) is enabled" check is applied because the kernel.randomize_va_space parameter is set to 2.

Figure 29 shows us that the control:

- "Ensure password expiration is 90 days" is enforced because PASS_MAX_DAYS is set to 90;
- "Make sure the minimum number of days between password changes is

```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).

kernel.randomize_va_space = 2

############################################################
###
### GENERAL NETWORK SECURITY OPTIONS ###
###
#Activer les cookies TCP SYN
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_syn_retries = 2
net.ipv4.tcp_synack_retries = 2
net.ipv4.tcp_max_syn_backlog = 4096
```

Figure 28. Address Space Randomization Value (ASLR).

```
login.defs - Bloc-notes
Fichier  Edition  Format  Affichage  Aide
#       PASS_MIN_LEN    Minimum acceptable password length.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   90
PASS_MIN_DAYS   0
PASS_MIN_LEN    5
PASS_WARN_AGE   7
```

Figure 29. Value of parameters related to passwords.

configured" is not correctly applied because the PASS_MIN_DAYS parameter is set to 0 instead of 1.

- "Make sure the minimum length of a password is 8" is not correctly enforced because the PASS_MIN_LEN parameter is set to 5 instead of 8.
- "Ensure password expiration warning days are 7" is applied because the PASS_WARN_AGE parameter is set to 7.

Figure 30 shows us that the "Make sure the warning message is configured" check is executed correctly because the message in the warning banner is correctly configured.

Figure 31 shows us that the "Make sure SELinux mode is enforced" check is executed because the SELinux is in "enforcing" mode.

5. Conclusion

The main objective of the work that we had to carry out was to strengthen the security of information systems by automating hardening mechanisms. We started by presenting the basic notions relating to hardening, then a methodological approach was adopted for the realization of this work by using automating the mechanisms of hardening of IS by a Web application. Through this, we

Figure 30. Warning message in ssh banner.

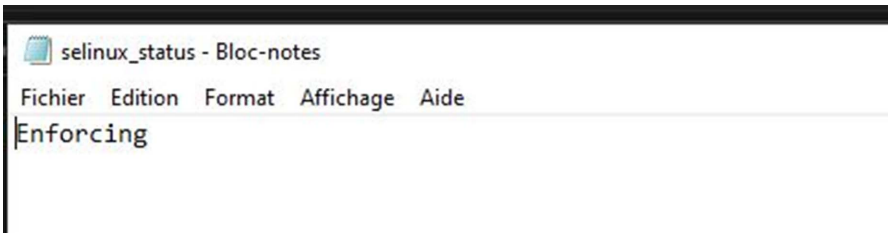


Figure 31. SELinux status.

were able to achieve the objectives defined at the start by making the choices of development tools, the hardening standards appropriate to our context, but also by setting up the architectures and design diagrams essential to the realization of our tool.

Thus, our application called HardeningApp was born with the following features:

- Automatic hardening of servers and network equipment;
- Verification of hardening configurations;
- Updating hardening procedures through adding, removing and modifying controls.

It appears that the work carried out gives satisfactory results, thus allowing any administrator of information systems to save time and efficiency in the management of hardening. All these functionalities are exposed in the f.

- Beyond hardening, build functionality that would provide access to server terminals and physical network equipment to perform various routines;
- Improve the accuracy of audit reports.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] (2022) Durcissement des systèmes: Introduction. https://social.technet.microsoft.com/wiki/contents/articles/25992.durcissement-des-systemes-introduction-fr-fr.aspx

[2] (2022) Best Hardening Tools. https://www.calcomsoftware.com/best-hardening-tools/

[3] Guide Complet du Durcissement des Systèmes en 2022 (2022).

https://www.ninjaone.com/fr/blog/guide-complet-du-durcissement-des-systemes-en-2022/

[4] (2022) Durcissement ou durcissement De quoi s'agit-il, à quoi sert-il et comment l'appliquer en informatique? https://www.informatique-mania.com/linformatique/durcissement/

[5] Chef Enterprise Automation Stack (2022). https://www.chef.io/products/enterprise-automation-stack#:~:text=Chef%20Enterprise%20Automation%20Stack%20(EAS,stage%20of%20the%20technology%20lifecycle

[6] (2022) What Is the Difference between Paramiko and Netmiko? https://fr.linuxteaching.com/article/what_is_the_difference_between_paramiko_and_netmiko

[7] Leka, G.E.N. (2022) Design and Production of an Automatic Management Tool for the Hardening of Servers and Network Equipment. *End-of-Study Dissertation with a View to Obtaining the Design Engineer Diploma in Telecommunications Engineering at ENSPY*, UYI.