UNIVERSIDAD SAN PABLO - CEU

ESCUELA POLITÉCNICA SUPERIOR

GRADO EN INGENIERÍA DE SISTEMAS DE INFORMACIÓN



TRABAJO FIN DE GRADO

**Hermes - Herramienta Linux para actualización automática de servicios**

Autor: José Manuel Martínez Sánchez

Tutor: Álvaro Sánchez Picot

Septiembre 2022

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Datos del alumno   |  | | --- | | Nombre: |   Datos del Trabajo   |  | | --- | | TÍTULO DEL PROYECTO: |   Tribunal calificador   |  |  | | --- | --- | | Presidente: | Fdo.: |  |  |  | | --- | --- | | Secretario: | Fdo.: |  |  |  | | --- | --- | | Vocal: | Fdo.: |  |  | | --- | | Reunido este tribunal el \_\_\_ /\_\_\_ /\_\_\_\_\_\_\_, acuerda otorgar al Trabajo Fin de Grado presentado por D./Dña. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ la calificación de \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |

Resumen

En este documento se detallan los procesos efectuados para llevar a cabo el diseño y la implementación de un servicio para máquinas Linux el cual, analiza las versiones de los servicios instalados y actualiza los mismos hasta su última versión retrocompatible. El servicio se encuentra disponible para cualquier sistema operativo derivado de Debian, OpenSuse o Fedora, y puede gestionar tanto los servicios de una máquina (local o remota) como de varias máquinas.

Este servicio ha sido desarrollado en base a la vulnerabilidad presentada por la OWASP (acrónimo de Open Web Application Security Project, en inglés Proyecto Abierto de Seguridad de Aplicaciones Web) **Vulnerable and Outdated Components**. Esta vulnerabilidad resalta la criticidad de usar componentes hardware y software actualizados para reducir la posibilidad de sufrir un ataque.

Según la organización, los ataques debido a mala configuración de servicios en máquinas han incrementado a lo largo de los últimos 5 años. Es por que, la vulnerabilidad ha escalado a la 6º posición en el Top 10 de Riegos de Seguridad en Aplicaciones Web por la OWASP.

Palabras Clave

UNIX, Servicio Linux, Systemd, Python 3, TDD, OWASP

Abstract

This document details the process carried out for the design and implementation of a service for Linux machines which automatically analyzes the versions of the installed services and updates them to their latest backward compatible version. This service is available for any system derived from Debian, OpenSuse, Fedora or MacOS, and can manage the services of a machine locally, as well as remotely on various machines, through the SSH protocol.

This service has been developed due to the vulnerability presented by the OWASP (Open Web Application Security Project) **Vulnerable and Outdated Components**. This vulnerability highlights the criticality of using up-to-date hardware and software components to reduce the possibility of an attack.

According to the organization, attacks due to misconfiguration of services have increased over the last 5 years. For this reason, the vulnerability has climbed to 6th position in the Top 10 security risks in web applications by OWASP.

Keywords

UNIX, Linux service, Systemd, Python 3, TDD, OWASP

Índice de contenidos

[Capítulo 1 Introducción 1](#_Toc112972675)

[Capítulo 2 Gestión del proyecto 5](#_Toc112972676)

[2.1 Modelo de ciclo de vida 5](#_Toc112972677)

[2.2 Papeles desempeñados en el proyecto 6](#_Toc112972678)

[2.3 Planificación 6](#_Toc112972679)

[2.4 Presupuesto 8](#_Toc112972680)

[2.5 Ejecución 9](#_Toc112972681)

[Capítulo 3 Análisis 11](#_Toc112972682)

[3.1 Especificación de requisitos 11](#_Toc112972683)

[3.2 Análisis de los casos de uso y de las clases de análisis 15](#_Toc112972684)

[3.3 Análisis de seguridad 18](#_Toc112972685)

[Capítulo 4 Diseño e implementación 21](#_Toc112972686)

[4.1 Arquitectura del sistema 21](#_Toc112972687)

[4.2 Modelo de clases de diseño 22](#_Toc112972688)

[4.3 Diseño físico de datos 22](#_Toc112972689)

[4.4 Diseño de la interfaz de usuario 25](#_Toc112972690)

[4.5 Entorno de construcción 26](#_Toc112972691)

[4.6 Plan de pruebas 27](#_Toc112972692)

[4.7 Diagrama de infraestructuras de nivel 3 28](#_Toc112972693)

[Capítulo 5 Construcción 30](#_Toc112972694)

[5.1 Referencia al repositorio de software 30](#_Toc112972695)

[5.2 Manuales 30](#_Toc112972696)

[Capítulo 6 Conclusiones y líneas futuras 33](#_Toc112972697)

[Bibliografía 35](#_Toc112972698)

Índice de ilustraciones

Índice de tablas

# Introducción

Para este proyecto se ha desarrollado un servicio UNIX denominado Hermes cuya función es la de analizar y gestionar el versionado de los servicios instalados en una o más máquinas Linux. Gracias a este, se pretende que los servicios de cualquier sistema Linux derivados de SUSE, Debian y Fedora se mantengan actualizados a la última versión sin la intervención del usuario de la máquina. El servicio, se encuentra escrito en el lenguaje de programación Python 3 y esta adaptado para poder ser gestionado a través de la herramienta "systemctl".

En este documento se exponen detalladamente los procesos llevados a cabo para la realización del proyecto. Los procesos se encuentran divididos en varios capítulos que son:

1. **La gestión el proyecto**, especificando el modelo de ciclo de vida en el que se encuentra basada la gestión del proyecto, la planificación desarrollada en base al modelo, el presupuesto estimado para llevar a cabo el proyecto y el resultado de la planificación y presupuesto, una vez ejecutado el proyecto.
2. **La planificación del servicio**, definiendo los requisitos funcionales del servicio, detallando sus casos de uso y analizando su impacto desde el punto de vista de la seguridad.
3. **El diseño del servicio**, detallando la arquitectura del servicio, especificando el diseño de clases, los datos y la interfaz de usuario; enumerando los entornos usados para su construcción y detallando el plan de pruebas implementado para la validación de los requisitos.
4. **El servicio resultante**, referenciando el código desarrollado y los manuales descritos para su correcto uso.
5. **Las conclusiones**, reflexionando sobre posibles errores cometidos durante el desarrollo e identificando líneas futuras de desarrollo para el proyecto.

Dentro del proyecto, se han establecido diferentes objetivos principales con los que se espera validar el éxito global del mismo. Estos objetivos son:

* Desarrollar un servicio sencillo e intuitivo con el que, cualquier usuario con experiencia en servicios UNIX, pueda mantener las versiones de sus servicios actualizadas de forma automática.
* Desarrollar un servicio que permita mantener las actualizar los servicios de no solo una máquina, sino también de un conjunto elevado de sistemas al mismo tiempo.
* Desarrollar un servicio con el cual el usuario pueda obtener información relevante sobre el estado del versionado de las mismas de manera gráfica e intuitiva.

# Gestión del proyecto

## Modelo de ciclo de vida

Para el desarrollo del servicio en el que esta enfocado el proyecto, se usará como modelo de ciclo de vida el modelo en cascada o clásico. Este modelo se basa en la realización de las etapas del desarrollo del producto de forma lineal, de tal manera que solo se podrá empezar con una fase sí y solo si la fase anterior ha sido cumplida. La decisión de usar este modelo se basa en varias de las ventajas que ofrece, de las cuales el desarrollo del proyecto y las personas involucradas pueden beneficiarse. Estas ventajas son:

**Definición de una estructura clara**. El modelo permite establecer una estructura y simple, permitiendo crear un único flujo continuo de procesos definidos. Este flujo reduce las probabilidades de qué alguna parte del proyecto se quede sin terminar y, en el supuesto de que se produzca un obstáculo, este podrá ser atacado de inmediato por los participantes del proyecto.

**Difusión adecuada de la información.** Al establecer un proceso sumamente metódico, el intercambio de información en cuanto a los requerimientos generales del proyecto y específicos de cada una de sus partes mejora con respecto al uso de otras metodologías.

## Papeles desempeñados en el proyecto

Los personajes que formarán parte activa en el desarrollo del servicio serán el tutor del TFG (también conocido como el dueño del producto); y el estudiante que, debido al propósito del proyecto, realizará la función de diseñador, programador y tester del servicio.

Debido a los conocimientos del estudiante en la materia, adquiridos en diferentes clases cursadas durante la carrera; y a los conocimientos del tutor, no se ha considerado necesario la involucración de uno o más usuarios expertos para la resolución del problema. Aunque, en el que caso de que sea necesaria la involucración de un usuario experto debido a algún inconveniente, se tiene conocimiento de algunos profesores de la universidad que podrían ayudar.

Durante la ejecución del proyecto, el dueño del producto tendrá como función validar cada uno de los hitos a realizar. Para superar con éxito cada una cada uno de los hitos se deberán establecer unos parámetros de validación previos a la ejecución del proyecto.

En el caso del estudiante, sus funciones principales serán analizar el entorno para detectar otras posibles soluciones, establecer los requisitos y herramientas a usar, implementar una solución que cumpla con los requisitos y cerciorarse, mediante pruebas funcionales, el correcto desarrollo del servicio. Además, será responsable de actualizar el estado del proyecto al tutor cada vez que se finalice con una de las tareas impuestas en la planificación, mediante una reunión presencial.

## Planificación

Siguiendo con la metodología seleccionada, se ha diseñado una planificación en cascada dividida en 5 pasos o hitos. Estos hitos son:

1. **Análisis previo del entorno**. Antes de establecer los requisitos del servicio, se realiza un estudio del mercado en busca de algún producto o servicio que realice una función igual o similar a la del proyecto. Además, se estudian las diferentes tecnologías que pueden ser usadas para desarrollar el producto.
2. **Especificación de los requisitos**. Tras haber realizado el análisis, se establecen los requisitos funcionales y no funcionales que debe cumplir el servicio.
3. **Implementación de la solución**. Una vez definidos los requisitos, se desarrolla una solución en base a los requisitos marcados y el análisis realizado.
4. **Desarrollo de pruebas**. Una vez implementada la solución, se generará tanto una batería de pruebas como un entorno de simulación para poder testear el funcionamiento del servicio y comprobar que los requisitos especificados se cumplen.
5. **Documentación y despliegue**. Por último, se despliega el software desarrollado en conjunto con la documentación necesaria. La documentación debe incluir tanto la memoria del proyecto y como un manual de uso de la herramienta.

Aparte de los pasos previos, se incluye una tarea colchón la cual dará flexibilidad en tiempo de ejecución del proyecto en el caso de se supere el tiempo establecido en cualquiera de las tareas previas. Esta tarea tendrá un valor, en días, equivalente al 10% del tiempo total esperado para realización de las anteriores tareas.

Para poder observar con mayor detalle la planificación se adjunta el siguiente diagrama de Gantt, en el que se detallan los hitos establecidos y el tiempo estimado de realización de cada uno.

Gráfico, Escala de tiempo

Descripción generada automáticamente

Ilustración 2.3.1

## Presupuesto

Tras haber realizado el análisis del entorno y haber considerado las herramientas que van a ser necesarias para ejecutar el proyecto se estima que el presupuesto necesario será inferior a los 250 euros.

Se ha considerado un presupuesto tan bajo al no ser necesario realizar un desembolso económico inicial para la adquisición de componentes hardware (ordenador, periféricos, conexión a internet, etc.) y software (IDEs, máquinas virtuales, etc.) que van a ser necesarias para el servicio al contar ya con ellas.

Aún así, se reserva una cantidad considerable de efectivo para la realización de las pruebas, en el caso de que la capacidad actual de hardware disponible no permita ejecutarlas. De ser así, se hará uso de la infraestructura de AWS para lanzar diversas maquinas virtuales con las que realizar las pruebas sobre los diferentes sistemas operativos que alcanza el proyecto.

## Ejecución

Durante la realización del proyecto se han dado diferentes problemas que han afectado a la planificación y el presupuesto del proyecto. Los inconvenientes dados son a consecuencia del modelo de ciclo de vida que se seleccionó. En el momento de su selección, no se le detectaron las diferentes desventajas que se han dado. La principal desventaja que no se contemplo esta directamente ligada con el flujo de trabajo lineal del modelo.

Debido a este flujo, no se pudieron detectar varios problemas con la obtención de las versiones de los servicios en las diferentes distribuciones de Linux durante el hito de "Implementación de la solución" hasta que se alcanzo el "Desarrollo de las pruebas". Por este motivo, se tuvo que volver al proceso previo y corregir los errores detectados. Una vez resueltos, se tuvo que volver a testear todo el servicio con las nuevas modificaciones en la implementación. Este inconveniente afecto a los tiempos destinado para la realización de ambos hitos, pero no influyó en la fecha final del proyecto, debido a la creación de la tarea colchón la cual pudo amortizar el tiempo perdido con el tiempo asignado a esta.

En cuanto al presupuesto, sí que se ha hecho uso del capital establecido para las pruebas del servicio, pero este no ha superado al presupuesto original del proyecto. Cómo se preciso en el presupuesto, los fondos fueron destinados al despliegue de distintas máquinas virtuales con las distribuciones para las que están desarrollando el servicio.

# Análisis

## Especificación de requisitos

Los requisitos establecidos por los participantes del proyecto para el desarrollo del servicio son:

* *El servicio debe poder ser ejecutado desde cualquier máquina con una distribución derivada de UNIX que disponga del interprete de Python 3 y de la herramienta de gestión de servicios "systemctl".*
* *El servicio debe tener la capacidad de realizar sus funciones de análisis y actualización de servicios únicamente sobre máquinas Linux cuya distribución sea derivada de Fedora, SUSE y Debian.*
* *El servicio deberá extraer los servicios y versiones de los mismos instalados en una máquina cuya distribución se encuentre dentro de la incluidas en el proyecto.*
* *El servicio deberá tener la capacidad de actualizar automáticamente los servicios de una máquina a la última versión disponible, si la versión instalada es retrocompatible con la versión a instalar.*
* *En los casos donde los servicios no sean retrocompatibles con la última versión disponible, el servicio no deberá actualizarlos, independientemente de que si exista una actualización retrocompatible con la versión instalada.*
* *El servicio deberá de generar y enviar las instrucciones necesarias para la extracción de información y la actualización de los servicios con independencia del sistema operativo de la maquinas analizada.*
* *El servicio podrá ser ejecutadas sobre la máquina local o sobre una o más máquinas remotas, si estas cuentan con los sistemas operativos disponibles y disponen de la capacidad de realizar conexiones remotas.*
* *El servicio realizara sus operativas de forma automática, sin la intervención activa del usuario. El usuario solo debe introducir las máquinas analizadas y las credenciales de acceso a los mismos.*
* *El servicio debe realizar las conexiones con las máquinas analizadas a través del puerto 22/TCP, conocido por ser el puerto estándar para el servicio SSH.*
* *El servicio será capaz de repetir el análisis de las máquinas cada cierto tiempo. Este parámetro dispondrá de un valor en tiempo predefinido, pero el usuario deberá tener la capacidad de poder modificarlo.*
* *El servicio podrá realizar el análisis de las máquinas sobre una o más direcciones IP, rangos de direcciones IP o redes completas.*
* *El usuario deberá tener la capacidad de introducir el direccionamiento de las máquinas mediante direcciones IP, rangos de direcciones IP o redes completas.*
* *En los casos donde las máquinas a analizar requieran de una clave privada para su conexión remota, el usuario tendrá la capacidad de instanciar su ruta a la clave para que el servicio pueda efectuar la conexión.*
* *Las credenciales que se aporten de las máquinas deben contar con los permisos necesarios para poder efectuar cambios en el versionado de sus servicios instalados.*
* *En ningún momento el servicio debe ser deshabilitado en caso de error. En el caso de que surja uno, el servicio debe continuar con sus operativas. Solo se deshabilitará el servicio si el usuario lo desea.*
* *El servicio se deberá instalar en la máquina mediante un fichero de configuración. La ejecución de este será suficiente para que el usuario pueda usarlo.*
* *El servicio deberá contar un fichero de configuración que permita la desinstalación del servicio. La ejecución de este será suficiente para que se eliminen todos los ficheros del servicio.*
* *Los ficheros que no sean parte del funcionamiento del servicio (ficheros de instalación, configuración y resultados), tendrán únicamente permisos de lectura y escritura para el usuario responsable de la ejecución.*
* *Los ficheros que sean parte del funcionamiento del servicio (scripts), tendrán únicamente permisos de ejecución para el usuario responsable de la ejecución.*
* *Los usuarios, que no sean responsables de la ejecución del servicio, no contarán con permisos de lectura, escritura y ejecución de ninguno de los ficheros que sean parte del servicio.*
* *El servicio debe almacenar distintos datos resultantes de la ejecución del servicio. Estos datos no deben, en ningún momento, guardar información sensible con la cual se pueda identificar el direccionamiento de una máquina.*
* *El tiempo de análisis de una máquina debe ser inferior a los 2 minutos y debe tener la capacidad de analizar 20 máquinas como mínimo.*
* *El servicio dispondrá de un dashboard con el cual el usuario podrá observar datos relevantes sobre el estado del versionado de las maquinas analizadas. Los datos de este servicio provendrán de los datos almacenados de la ejecución del servicio.*
* *El dashboard solo estará disponible cuando el servicio se encuentre habilitado. En caso contrario, este no podrá ser accedido.*
* *El dashboard debe de contar con parámetros de configuración con el fin de que el usuario pueda obtener información más precisa de una métrica que se obtiene mediante la ejecución del servicio.*
* *Cada vez que el servicio sea habilitado la información previa será almacenada en un nuevo fichero cuyo nombre debe de identificar la fecha y de creación del documento.*
* *Cada vez que el servicio sea habilitado la nueva información será almacenada en el fichero destinado al resultado de la ejecución. La información previa que haya en ese documento debe de ser borrada.*

## Análisis de los casos de uso y de las clases de análisis

Habiendo definido los requisitos del servicio y realizando un estudio sobre las diferentes casuísticas de uso que se darán dar en el proyecto, se han determinado los casos de uso existentes que requieren de la involucración del usuario. La ejecución de estos casos es independientemente del sistema operativo de la máquina donde se ejecute el servicio, siempre que este se encuentre dentro de los sistemas que alcanza el proyecto; y de las máquinas donde se realice el análisis. Estos casos son:

**Caso Nº1 – Configuración de servicio.** Para que el sistema pueda ser ejecutado, este debe de ser previamente instalado y configurado. Para instalarlo, el usuario debe ejecutar, en la máquina, el fichero de instalación para copiar los ficheros del servicio en su directorio. Una vez instalado, deberá establecer los valores de configuración del análisis, situados en un fichero de configuración, para poder establecer el direccionamiento y credenciales de con las máquinas a analizar y el tiempo entre escaneos.

Diagrama

Descripción generada automáticamente

Ilustración 3.2.1

**Caso Nº2 – Habilitación o deshabilitación del servicio.** Al tratarse de un servicio, el usuario solo deberá habilitar o deshabilitar el mismo, después de haber instalado y configurado la herramienta, mediante el comando "systemctl". Además, cuando el sistema este habilitado, el usuario podrá acceder al dashboard con los datos generados por la eyección del servicio.

Diagrama

Descripción generada automáticamente

Ilustración 3.2.2

**Caso Nº3 – Desinstalación del servicio.** Cuando ya no desee usar la herramienta, el usuario podrá eliminar el servicio del sistema ejecutando fichero de desinstalación.

Diagrama

Descripción generada automáticamente con confianza media

Ilustración 3.2.3

A partir de los casos de uso extraídos, se ha obtenido como resultado el siguiente modelo de clases con el que se definirá la estructura del proyecto. Esta estructura cuenta con 8 ficheros en los que se incluyen:

* Instalador (install.sh): fichero bash cuya función es copiar los ficheros del proyecto, a la estructura de directorios del sistema, para poder ejecutar el servicio.
* Desinstalador (remove.sh): fichero bash cuya función es eliminar los ficheros del proyecto, tanto los originales como aquellos derivados de la ejecución del servicio, de la estructura de directorios del sistema.
* Fichero de configuración de Python (.conf): fichero de configuración de los parámetros, a incluir por el usuario, necesarios para la ejecución del servicio.
* Fichero de configuración de servicio (.service): fichero de configuración necesario para que la aplicación desarrollada sea reconocida como un servicio por el sistema.
* Script Python de configuración del servicio (service.py): script de Python donde se ejecutan los métodos desarrollados en "app.py".
* Script Python de aplicación del servicio (app.py): script de Python donde se instancian los métodos necesarios para el análisis de los sistemas y el lanzamiento de comandos.
* Script Python de creación del dashboard (dashboard.py): script de Python encargado de instanciar y generar el dashboard interactivo con los datos obtenidos de la ejecución del servicio.

Las conexiones entre estos ficheros se pueden ver representados en el siguiente diagrama de clases.

Diagrama

Descripción generada automáticamente

Ilustración 3.2.4

## Análisis de seguridad

Desde el punto de vista de la seguridad existe diferentes riesgos que pueden afectar a la confidencialidad, integridad y/o disponibilidad de las máquinas participes en el servicio. De los riesgos analizados, se han detectado 3 posibles amenazas que pueden afectar al servicio y a las máquinas analizadas.

**La integridad y confidencialidad de la información que se puede obtener de las máquinas a través de la configuración del servicio**. Para que se puedan analizar las máquinas, el servicio debe disponer de el direccionamiento IP y las credenciales de acceso de las mismas. Esta información, que el usuario proporciona, puede llegar a supone un riesgo de seguridad en el supuesto de que sea sustraída por un ciberdelicuente. Para limitar este riesgo, se da como solución limitar el de cualquier memoria volátil, como ficheros. En el caso de que el uso de estos sea necesario, se sugiere limitar los permisos de lectura y escritura a solo el usuario que vaya a ejecutar el servicio e incluso, limitar su acceso una vez el servicio se este ejecutando y la información ya este cargada.

**La confidencialidad de la información resultante de ejecutar el servicio sobre una máquina.** Otros los de riegos posibles a enfrentar es la fuga de datos sensibles procedentes del análisis de las máquinas, amenazando la integridad de las mismas. La información que se considera sensible incluye el direccionamiento IP de las máquinas, los nombres de los servicios o las versiones de los mismos entre otros. Si esta información fuese descubierta por un atacante, este dispondría de los recursos necesarios para realizar un ataque sobre los servicios de las máquinas analizadas, pudiendo crear un importante incidente de seguridad. Por lo tanto, se propone no almacenar ningún dato identificativo de las máquinas resultante de la ejecución, ya que no se consideran necesarios para desarrollar las funcionalidades que se quieren desarrollar para el proyecto ni aportar ningún valor a las funcionalidades que desean para el proyecto.

**La integridad, confidencialidad y disponibilidad de las máquinas a la hora de realizar una conexión remota.** El último riesgo que puede contraponer la seguridad de las máquinas participes del servicio es el uso de conexiones remotas. Esta operatividad, aun siendo fundamental su desarrollo en el servicio para el análisis de máquinas remota, manifiesta vulnerabilidades de las cuales los ciberdelicuente pueden aprovechar. En el caso de que la conexión no sea segura, un atacante podría obtener información de la transferencia de datos entre máquinas, creando una vulnerabilidad de la integridad y la confidencialidadde las mismas. Es por ello, que se recomienda usar un protocolo de conexión seguro que permita el cifrado de información, extremo a extremo, por medio de un canal seguro, como el protocolo SSH (Secure Shell). Este, aparte cumplir con los requisitos mencionados, cuenta con módulos prediseñados para su uso en diferentes lenguajes de programación, por lo que no sería necesario construir una solución desde cero.

# Diseño e implementación

## Arquitectura del sistema

A partir del análisis y de los requisitos descritos se ha elaborado el siguiente diagrama donde se representa la arquitectura del sistema, indicando sus interfaces, sus servicios y los componentes y datos en los que se encuentra apoyado el servicio.

Diagrama

Descripción generada automáticamente

Ilustración 4.1.1

## Modelo de clases de diseño

En base al diagrama de clases de análisis realizado durante el análisis y tras el estudio de las herramientas tecnológicas disponibles, se ha desarrollado el siguiente modelo de clases en el esta basado el proyecto. En este se detallan los métodos y funciones que cada fichero debe de poseer, las relaciones entre los documentos y las bibliotecas externas al lenguaje que han de ser usadas.

Diagrama

Descripción generada automáticamente

Ilustración 4.2.1

## Diseño físico de datos

El diseño de los datos que se ha implementado en el servicio afecta tanto a la configuración del servicio como a los datos resultantes de la ejecución. Estos datos cuentan con sistemas de almacenamiento distintos, los cuales almacenan estructuras de datos, valores y tipos de datos diferentes.

Para la configuración del servicio se registran cuatro valores, los cuales deben de ser cumplimentados por el usuario, que son necesarios para la ejecución de servicio. Estos se encuentran almacenados en fichero de configuración denominado "service.conf", localizado en el directorio "/etc/hermesd". Estos valores son:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Descripción** | **Valores permitidos** |
| network | Direccionamiento IP de las máquinas que se desean analizar. | * Dirección IP * Direcciones IP * Rango de IPs * Red |
| user | Usuario de acceso a las máquinas a analizar. | string |
| password | Contraseña del usuario de acceso. | string |
| key | Clave privada, en el caso de que sea necesaria para realizar la conexión remota a las máquinas | string |
| time | Tiempo de espera entre análisis en horas | integer |

Tabla 4.3.1

En cuanto a los datos generadas a partir de la ejecución del servicio, se almacenan diferentes métricas que permiten obtener una imagen detallada del estado de los servicios del sistema. Estas métricas son almacenadas en el documento CSV "hermes.csv", localizado en el directorio "/hermesd/". Cuando se deshabilita el servicio, los datos en este documento son copiados a uno nuevo con la fecha de deshabilitación del servicio como nombre. Los valores que se registran en este documento son:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Descripción** | **Valores permitidos** |
| Fecha | Fecha y hora de la ejecución de la máquina. | YYYY-MM-DD HH:MM:SS |
| Máquina | Sistema operativo de la máquina. | * Fedora * Debian * SUSE |
| Servicios Instalados | Número de servicios que se encuentran instalados en la máquina. | integer |
| Servicios Actualizados | Número de servicios que han sido actualizados en la máquina durante el análisis. | integer |
| Servicios OK | Número de servicios que disponen de la última versión disponible instalada en la máquina. | Integer |
| Servicios NOK | Número de servicios que no disponen de la última versión disponible instalada en la máquina. | Integer |

Tabla 4.3.2

## Diseño de la interfaz de usuario

Al ser el producto desarrollado un servicio de Linux, este no cuenta con una interfaz de usuario que permita ejecutar sus funciones disponibles. Este es ejecutado a través de una consola mediante la herramienta de administración de servicios de Linux "systemctl". El usuario únicamente puede habilitar o deshabilitar el servicio en el caso de que se quieran usar o no sus operativas. Aun así, desde una terminal, se pueden observar diferentes datos de la ejecución del servicio como, el estado del servicio o el log de la herramienta gracias a diferentes comandos presentes en la misma herramienta de administración.

A pesar de que no se cuenta con una interfaz, sí que se dispone de un dashboard interactivo. A través de este, el usuario puede observar gráficamente, mediante tablas y gráficas diferentes métricas que se obtienen a partir de los datos almacenados de la ejecución del servicio. Adicionalmente, al tratarse de un dashboard interactivo, existen ciertos parámetros que permiten extraer información más especifica sobre de las diferentes máquinas o servicios. El dashboard se encuentra soportado sobre una interfaz web, solo es accesible desde la máquina en donde se realiza el análisis y cuando el servicio se encuentre habilitado.

## Entorno de construcción

A lo largo de la ejecución de este proyecto se han usado distintas herramientas que han facilitado el desarrollo y las pruebas del servicio. Todas las herramientas que se han usado se encuentran catalogadas como software libre, es decir su uso no ha requerido de un pago. Las herramientas que se han usado son las siguientes

* **Terminal UNIX**. Al haber usado para el desarrollo del servicio una máquina con sistema operativo MacOS, se ha hecho uso de su terminal UNIX preinstalada. Esta se ha usado para la ejecución del servicio en todo el proceso de desarrollo, para la ejecución de las pruebas una vez se había finalizado el desarrollo y para el control de las diferentes versiones a través de la herramienta Git.
* **Visual Estudio Code**. Empleada para el desarrollo de los distintos ficheros de Python 3 y los ficheros de configuración necesarios para el servicio. Se decidió utilizar esta herramienta, en vez de otro editor de texto de terminal, debido a las facilidades que ofrece para programar gracias a los módulos que permiten el autocompletado de código.
* **Git**. Herramienta cuyo principal propósito es el control de las versiones del código con el fin de almacenar un registro de todo el trabajo realizado durante la realización del proyecto.
* **GitHub**. Servicio web que facilita el almacenamiento de las versiones registradas a través de hit en un repositorio en línea.
* **AWS**. Servicio web, proporcionado por Amazon, en el que se han ejecutado varias instancias de los diferentes sistemas operativos qué cubre el servicio. Al contrario que el resto de las herramientas, sí que se ha requerido de un pago para su uso.

## Plan de pruebas

Para validar la implementación del servicio, se ha diseñado un plan de pruebas dividido en dos partes. Este plan ha sido desarrollado con el objetivo de validar tanto los requisitos establecidos al comienzo del proyecto, como también las respuestas de los métodos implementados

En la primera parte de las pruebas, se diseñará una batería de pruebas unitarias que asegurarán que las respuestas dadas por los métodos sean iguales a los valores esperados. Esta batería estudiará cada casuística que se pueda dar en la extracción de la información, el análisis de los datos, la generación de instrucciones y la elaboración de métricas, incluyendo los casos en donde se obtengan errores. La conexión a las máquinas o el registro de los parámetros de configuración no deberán ser verificados al usarse módulos externos ya testeados. Las pruebas planteadas deben cubrir todos los sistemas operativos que se encuentran dentro del alcance del proyecto y cada uno de los casos que se de debe de contar con al menos 3 test para cerciorase de que las pruebas sean válidas.

Una vez se hayan elaborado los test y se compruebe que servicio supera estos sin error se elaborará la segunda parte de las pruebas, un entorno de prueba el cual comprobará el funcionamiento del servicio en conjunto. Para esta prueba, se instanciarán como mínimo 4 máquinas, 3 sistemas para realizar el análisis de sus servicios; y en 1 sistema para ejecutar el servicio. Las máquinas por analizar contarán con un sistema operativo distinto, de los incluidos en el alcance del proyecto; y con 6 servicios preinstalados que contendrán todas las posibles casuísticas que se puedan dar. De estos servicios, 2 se encontrarán actualizados a la última versión disponible; 2 no estarán actualizados, pero si podrán ser actualizados a la última versión al tener versiones retrocompatibles; y por último 2 no estarán actualizados y no se podrán actualizar al no disponer de versiones retrocompatibles. Para cerciorarse de que la prueba de entorno es un éxito, las versiones de los dos primeros servicios no deben ser modificadas, las versiones de los dos segundos deben de ser actualizadas y las versiones de los dos últimos deben conservarse.

En el momento que la solución desarrollada pase todas las pruebas implementadas en ambas partes, se considerarán validados los requisitos establecidos y los métodos desarrollados.

## Diagrama de infraestructuras de nivel 3

Imagen que contiene reloj, pantalla, medidor

Descripción generada automáticamente

Ilustración 4.7.1

# Construcción

## Referencia al repositorio de software

El servicio desarrollado para resolución de este proyecto se encuentra almacenado en el siguiente repositorio de GitHub [https://github.com/ josemanuel179/tfg](https://github.com/%20josemanuel179/tfg).

## Manuales

El servicio cuenta con a documentación necesaria para su adecuado uso dentro del mismo repositorio donde se almacena el código, en el siguiente enlace <https://github.com/josemanuel179/tfg#readme>.

# Conclusiones y líneas futuras

finalizar el proyecto, podemos resaltar que el resultado final desarrollado ha cumplido con las expectativas y objetivos impuestos además de satisfacer la necesidad principal de su desarrollo, creando un servicio que de forma activa reduce al mínimo el riesgo de un posible ataque de seguridad por no mantener actualizados los servicios en máquinas Linux.

El resultado más relevante del proyecto, a mi parecer, a sido poder desarrollar el servicio con la capacidad de poder realizar el análisis en varias máquinas remotas desde un mismo sistema. Gracias a esta cualidad, el servicio puede ejecutar el análisis tanto en la propia máquina cómo en un parque entero de servidores en remoto, reduciendo considerablemente la puesta en marcha del servicio y facilitando su uso al no tener que instalarlo y configurarlo cada máquina donde se desee analizar los servicios.

Aun así, se debe admitir que se han cometido varios errores, en concreto en la toma de decisiones al inicio del proyecto, siendo el error más claro la selección del modelo de ciclo de vida. El uso del modelo seleccionado ha generado más inconvenientes que ventajas que se observaron al principio. Sobre todo, al incrementar el tiempo de ejecución del producto al no contemplar la posibilidad de realizar el desarrollo y las pruebas del producto al mismo tiempo. Este inconveniente llevó al retroceso del hito de desarrollo, para solucionar los problemas detectados durante las pruebas, retrasando el fin de ambos hitos por varias semanas. Aún así, la fecha de fin de proyecto no sufrió modificación alguna debido a la tarea de colchón, con la cual se pudo hacer frente al tiempo perdido.

Cabe destacar que existen varios factores a mejorar a futuro. En primer lugar, se debería buscar una solución más optima y segura para insertar las credenciales de acceso a las máquinas, una solución en donde los datos no se encuentren almacenados en ningún documento permanente. Además, se debería ampliar la cantidad de métricas para poder poblar de más información de calidad el dashboard. El nombre de lo servicios, el estado de la comunicación remota o la respuesta a la conexión SSH podrían ser algunas métricas de utilidad para el usuario. Por último, y para mejorar la facilidad del servicio, sería conveniente que el servicio se pueda instalar y desinstalar a través de los sistemas de gestión y administración los paquetes de sus distribuciones, y no mediante ficheros de configuración.

En conclusión, gracias a este proyecto he podido demostrar la variedad de habilidades y conocimientos, los cuales he adquirido durante mi etapa en la universidad, para poder llevar a cabo la solución; además de adquirir información nueva sobre el campo de la ciberseguridad.

# Bibliografía

1. *configparser — Configuration file parser — Python 3.10.6 documentation*. (s. f.). Python Docs. <https://docs.python.org/3/library/configparser.html>
2. *Crear Servicios con Systemd en Ubuntu - El Taller del Bit*. (2020, 31 agosto). El Taller del BIT. <https://eltallerdelbit.com/crear-servicio-systemd-ubuntu/>
3. Dubey, D. (2020, 7 marzo). *How to Setup Python Script Autorun As a Service in Ubuntu 18.04*. <https://websofttechs.com/tutorials/how-to-setup-python-script-autorun-in-ubuntu-18-04/>
4. Galvez, V. (2022, 22 enero). *Create service systemctl-systemd in python - Towards Dev*. Medium. <https://towardsdev.com/create-service-systemctl-systemd-in-python-9a0e8b5ab6ae>
5. Gite, V. (2022, 7 febrero). *How to list all installed packages on OpenSUSE/SUSE Linux*. Cyberciti. <https://www.cyberciti.biz/faq/list-all-installed-packages-on-opensuse-suse-linux/>
6. Khan, W. (2021, 15 diciembre). *Setup a python script as a service through systemctl/systemd*. Medium. <https://medium.com/codex/setup-a-python-script-as-a-service-through-systemctl-systemd-f0cc55a42267>
7. OWASP. (s. f.). *OWASP Top Ten | OWASP Foundation*. <https://owasp.org/www-project-top-ten/>
8. *Python - IP Address*. (s. f.). Tutorialspoint. [https://www.tutorialspoint.com/ python\_network\_programming/python\_ip\_address.htm](https://www.tutorialspoint.com/%20python_network_programming/python_ip_address.htm)
9. Raj, A. (2020, 14 diciembre). *ConfigParser Module - Create Configuration Files in Python*. AskPython. <https://www.askpython.com/python-modules/configparser-module>
10. Real Python. (2021, 3 abril). *Reading and Writing CSV Files in Python*. <https://realpython.com/python-csv/>
11. *45 Zypper Commands to Manage ‘Suse’ Linux Package Management*. (2017, 27 julio). Tecmint. <https://www.tecmint.com/zypper-commands-to-manage-suse-linux-package-management/>
12. Gite, V. (2017, 23 diciembre). *How to use yum command on CentOS/RHEL*. Cyberciti. <https://www.cyberciti.biz/faq/rhel-centos-fedora-linux-yum-command-howto/>
13. Gite, V. (2022, 27 abril). *Apt Command Examples for Ubuntu/Debian Linux*. Cyberciti. <https://www.cyberciti.biz/faq/ubuntu-lts-debian-linux-apt-command-examples/>
14. P. (2019, 7 agosto). *🌟 Introducing Dash 🌟 - Plotly*. Medium. <https://medium.com/plotly/introducing-dash-5ecf7191b503>
15. Plotly. (s. f.). *Dash Overview*. <https://plotly.com/dash/>
16. Real Python. (2020, 17 diciembre). *Develop Data Visualization Interfaces in Python With Dash*. <https://realpython.com/python-dash/>
17. Tomar, A. (2022, 3 agosto). *Dash for Beginners: Create Interactive Python Dashboards*. Medium. <https://towardsdatascience.com/dash-for-beginners-create-interactive-python-dashboards-338bfcb6ffa4>
18. *Ubuntu Manpage: apt - command-line interface*. (s. f.). Ubuntu. <https://manpages.ubuntu.com/manpages/bionic/man8/apt.8.html>