MÁSTER EN INGENIERÍA WEB, UNIVERSIDAD DE OVIEDO

TRABAJO DE FIN DE MÁSTER

Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Escuela de Ingeniería Informática
Universidad de Oviedo

EGIDA: Sistemas para el despliegue automatizado de configuraciones de seguridad con detección temprana de errores

---

Autor:
Antonio Payá González

Supervisor:
José Manuel Redondo López

Proyecto de Investigación

June 14, 2021

Abstract

Automated deployment of security controls is a very important part of deploying a defense-in-depth strategy to secure machine infrastructures. This paper describes a prototype of a technique to perform automated deployment of validated and international security controls in a more controlled way using a DSL. This way, according to the target machine configuration or characteristics, this controlled deployment is less prone to negatively impact legitimate running services, and better capture system administrator expert knowledge, being able to share automated security scripts that obtain better hardening results. The initial results of this technique are promising enough to apply them on educational environment and develop them further to be applied on real infrastructures requiring this kind of security automation.

Resumen

El despliegue automatizado de controles de seguridad es una parte muy importante del despliegue de una estrategia de defensa en profundidad para securizar la infraestructura de las máquinas. Este artículo describe el prototipo de una técnica para realizar un despliegue automatizado de controles de seguridad validados e internacionales de forma más controlada utilizando un lenguaje de dominio específico. De esta forma, según la configuración o las características de la máquina objetivo, este despliegue controlado es menos propenso a impactar negativamente en los servicios legítimos en ejecución y permite captar mejor el conocimiento experto del administrador del sistema, pudiendo compartir scripts de seguridad automatizados que obtengan mejores resultados de hardening. Los resultados iniciales de esta técnica son lo suficientemente prometedores como para aplicarlos en un entorno educativo y seguir desarrollándolos para aplicarlos en infraestructuras reales que requieran este tipo de automatización de la seguridad.

Agradecimientos

En primer lugar, quisiera agradecer a mis padres y a mi hermana por todo lo que han hecho por mí ya que sin ellos no habría llegado hasta aquí y son los principales responsables de que esté finalizando este Máster y sobre todo de mi formación como persona.

También me gustaría agradecer a mi tutor José Manuel Redondo López por haberme permitido llegar tan lejos en este campo, y por todo el apoyo y ayuda que me ha brindado todo este tiempo. Agradecer a mi compañera Alba Cotarelo por la ayuda y esfuerzo por aguantarme durante el desarrollo del Máster y por la colaboración entre nuestros trabajos de fin de estudios.

Por último, agradecer a mi pareja por ayudarme y soportarme durante el grado y durante el Máster tanto en los buenos como en los mejores momentos.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO

1

INTRODUCCIÓN

1.1 Motivación

El proceso de securización de los sistemas informáticos pasa por una correcta implementación y aplicación de estándares internacionales de gestión de la seguridad de la información *(Information Security Management System, ISO 27001 [1])*. Desgraciadamente, la proliferación de ataques actual muestra que en muchas ocasiones estas implementaciones no se hacen de manera completa o adecuada.

La correcta implementación de un *Information Security Management System (ISMS)* puede facilitarse si se utilizan herramientas o mecanismos que permitan automatizar gran parte o todo el proceso. El protocolo *SCAP (Security Content Automation Protocol)* es un referente internacional que permite lograr esta automatización del proceso y, además, proporciona formatos y nomenclaturas estándar que permiten definir e intercambiar información entre usuarios finales y herramientas [2].

No obstante, la automatización por sí sola no basta, también hay que seleccionar las acciones a realizar (internacionales, con soporte, validadas, etc.). Existen varias fuentes posibles, pero en este proyecto hemos decidido utilizar los *CIS Benchmarks* [3]. Además de las ventajas que ofrecen, mediante el uso de los *CIS Controls* podemos mapear las acciones realizadas a elementos presentes en estándares [4] como el *ISO 27001*, facilitando su implementación.

Sin embargo, aplicar sin control todos los *CIS Benchmarks* puede perjudicar el rendimiento o invalidar servicios que se ejecutan legítimamente en el sistema (por ejemplo, los *CIS Benchmarks* contienen tares categorizadas como *Nivel 2 Workstation* que son peligrosas, ya que pueden afectar al correcto funcionamiento del sistema si no se ejecutan con precaución). Por ello, es necesario tener un control detallado de las operaciones a realizar y el conocimiento experto de un administrador para evitar problemas en su despliegue.

1.2 Finalidad del Proyecto

El presente proyecto plantea un sistema llamado **Egida** que permite el uso de un *Lenguaje de Dominio Específico* (DSL) para modelar dicho conocimiento y aplicar los *CIS Benchmarks* sobre infraestructuras de manera que sea posible adaptarlos a casos concretos de empresas y prevenir problemas en su implementación antes de que esta se produzca (a diferencia de otros proyectos y herramientas existentes donde los errores se producen durante la propia ejecución). En este proyecto se pretende modelar el conocimiento experto y tomar decisiones en base a lo que se encuentra en cada sistema a securizar.

Para lograr esto, se plantea el uso de una arquitectura de *Agentes* similar a la de productos como OSSEC [1] que, en lugar de proporcionar funcionalidades que permitan detectar intrusiones, se proporcionarán características del sistema de mayor o menor detalle que permitan dirigir la aplicación de controles de seguridad de los *CIS Benchmarks* de forma más inteligente y flexible, adaptándose a cada caso, siendo además posible la detección de potenciales problemas de manera temprana.

De esta forma, un administrador de sistemas o de seguridad entrenado puede crear perfiles de seguridad que dependan de las características del sistema objetivo. Estos perfiles pueden especificar que controles de seguridad se van a aplicar en función de las necesidades del sistema objetivo o que controles de seguridad deben evitarse.

El sistema propuesto tiene como objetivo una aplicación docente (sistema con bajo consumo de recursos) pero que pueda ser escalado a usos de entornos profesionales en el futuro.

1.2.1 Despliegue Automatizado de Configuraciones de Seguridad

Egida utilizará la especificación más actualizada de las guías de los **CIS Benchmark** para aplicar medidas de seguridad y realizar un blindaje de Sistemas Operativos Linux conocidos como punto de partida. Dado que la automatización con *SCAP* se cubre a través de los *CIS Benchmarks* (con coste), con Egida intentamos mejorar esta automatización introduciendo nuevos elementos con fases de control, verificación y remediación:

- **Utiliza herramientas de implementación de configuración estándar para distribuir la verificación de controles de seguridad y procedimientos de implementación**. En lugar de seguir el enfoque *SCAP*, que actualmente puede ser potencialmente incompatible con diferentes sistemas, utilizaremos una herramienta de gestión de configuración centralizada muy popular, utilizada por empresas muy importantes a nivel mundial: **Ansible**. De esta manera, el proceso de seguridad *hardening* podría integrarse con otros procesos de administración existentes, sin necesidad de instalar otras soluciones dentro de un grupo de software. *Ansible* está disponible para sistemas Linux, Windows y Mac OS, por lo tanto, Egida podría ser más fácil de portar. *Ansible* también proporciona dos ventajas adicionales muy importantes: la capacidad de trabajar sobre cualquier máquina en la que tengamos acceso por el protocolo *SSH* y la capacidad de no repetir operaciones que se detectan como ya aplicadas. Esto

[1] `https://www.ossec.net/` last accessed on 28, May

facilita enormemente las pruebas de control de seguridad y propagación del cumplimiento con un alto grado de eficiencia.

- **Control preciso de los controles de seguridad**: existen otras implementaciones gratuitas de los controles de seguridad de los *CIS Benchmarks* distribuibles a través de *Ansible* [5], [6], pero estos podrían estar incompletos, no estar debidamente mantenidos y, lo más importante, podrían seguir un enfoque de todo o nada. Esto significa que cada control en el *benchmark* se prueba o aplica sin permitir que los usuarios elijan grupos/secciones de control o selecciones que se adapten a un cierto propósito o especialización del servidor. Con **Egida** queremos conseguir un resultado mucho más preciso y una gestión flexible de los controles de seguridad que se pueden implementar, creando perfiles de control que pueden superponerse o combinarse a voluntad del usuario. Esto asegura desplegar justo lo que se necesita (favoreciendo especialización de la máquina), y el uso de combinaciones de grupos de control para crear perfiles de máquina, mejorar o adecuar los controles desplegados a ciertos usos típicos. La agregación de estos perfiles de control de seguridad para adaptarse a perfiles concretos de máquinas es también un conocimiento que nuestro proyecto de investigación genera y que otros usuarios pueden aprovechar o evaluar.

- **Los controles de seguridad no son los únicos elementos de seguridad**. Aunque los controles de seguridad son una base muy sólida de un sistema seguro, también existen múltiples programas de seguridad que pueden complementarlos y mejorar aún más la seguridad de la infraestructura (véase la Figura 4.1). Software como *proxies inversos*, cortafuegos perimetrales, cortafuegos basados en host, *NIDS*, *HIDS*, *WAFs*, etc. pueden ser desplegados vía *Ansible* en ciertas máquinas de la infraestructura, yendo más allá del alcance de los *CIS Benchmarks*. El proyecto**Egida** también quiere aplicar su enfoque flexible a estos elementos de seguridad de nivel superior, no sólo desplegándolos sobre máquinas específicas, sino también facilitando la elección de la selección adecuada de los controles de seguridad de los *CIS Benchmarks* más apropiados para las máquinas con este software. Con ello se pretende conseguir las mismas ventajas que ofrecen proyectos como *JShielder* [7], pero la mayor flexibilidad permitirá a los usuarios especializar mejor las misiones de las máquinas, en lugar de crear una única máquina que concentre todo el software de protección.

- **Facilidad de despliegue y medición de niveles de *hardening***. Teniendo en cuenta que el proyecto **Egida** gestionará una gran cantidad de información, es imprescindible proporcionar a sus usuarios una interfaz gráfica de usuario que facilite la inspección y la gestión de los diferentes niveles de seguridad de cada máquina de la infraestructura, así como evaluar el estado actual de la seguridad en una determinada máquina utilizando herramientas de evaluación típicas como *Lynis*. El objetivo de la investigación de **Egida** es también demostrar que es posible mejorar la seguridad general del sistema permitiendo a los usuarios tener un control fácil y preciso sobre las actividades de *hardening* que se despliegan en una infraestructura de servidores. Esto se hace proporcionando un centro de mando y control centralizado para desplegar diferentes perfiles de *hardening* que hemos descrito anteriormente sobre las máquinas de una infraestructura, permitiendo a los usuarios conocer el estado de seguridad actual de cada dispositivo del sistema. Otro objetivo es complementar

las actividades de vigilancia de *IDS/IPS* desplegando máquinas suficientemente seguras para cada función. Por lo tanto, esta investigación también intenta proporcionar una forma de desplegar eficazmente estrategias y perfiles de *hardening* automatizados en un entorno de red de una empresa privada de forma sencilla.

1.2.2 Validación Temprana de Errores

Las soluciones basadas en lenguajes de marcado *XML* como *OVAL* o *XCCDF* [2] proporcionan especificaciones para detallar y desplegar con precisión los controles de seguridad, pero no permiten incorporar lo que podemos llamar comprobaciones semánticas. Los errores de despliegue de *Ansible* también se descubren en tiempo de ejecución. El objetivo final del proyecto de investigación **Egida** es desarrollar un **DSL** (Lenguaje de Dominio Específico) que permita a los usuarios desplegar grupos de controles de seguridad a partir de los *CIS Benchmarks*, pero realizando validaciones tempranas de las máquinas y/o contenidos de la infraestructura antes del despliegue. Aspectos como inconsistencias con las máquinas que se esperan encontrar, puertos abiertos, software disponible, directorios necesarios, usuarios requeridos, etc. podrían impedir que un despliegue sea exitoso. **Egida** pretende explorar los límites de lo que se puede validar antes de realizar un despliegue, para que la probabilidad de encontrar un fallo en tiempo de ejecución sea significativamente menor. Un control de errores preciso y temprano también puede permitir a los administradores resolver los problemas en un tiempo sustancialmente menor, ya que no es necesario ejecutar el procedimiento de despliegue posteriormente hasta que no se encuentre ningún error. De esta manera, tratamos de seguir la filosofía ya probada con éxito con los escaneos de Nmap [8].

El objetivo es definir programas DSL que contengan la configuración de *hardening* de las máquinas que cumplen un rol específico de una empresa. De esta forma, podemos tener un programa que contenga toda la configuración de *hardening* (controles de seguridad más sus valores variables asociados) de sus servidores HTTP, servidores de Sistemas de Gestión de Bases de Datos o proxies. Estos archivos podrían ser validados por expertos adicionalmente a la validación que realiza el DSL, y una vez determinados como válidos, ser distribuidos y reutilizados en toda la infraestructura de la empresa. Si la configuración cambia o debe ser refinada, también queremos comprobar si el DSL es capaz de detectar las inconsistencias producidas en las nuevas versiones antes de su despliegue.

Por lo tanto, el proyecto de investigación **Egida** utiliza la lista de control de los *CIS Benchmarks* para implementar el *hardening* del sistema de forma altamente automatizada. También incluye software de protección adicional que mejora la seguridad de las máquinas y les permite contrarrestar determinados tipos de ataques. Además, permite especializar el control de las aplicaciones, por lo que las diferentes máquinas de la infraestructura pueden desplegar sólo aquellas que sean más adecuadas a su función o a la red que operan, permitiendo un mayor control sobre lo que se despliega. Esto es muy importante porque en una infraestructura de máquinas, cada máquina tiene su rol principal, y la mayoría de los recursos de la máquina deben ser utilizados para cumplirlo. Las medidas de protección implementadas dentro de las máquinas que ejecutan un servicio de cliente real deben tomar el menor número de recursos posible, dejando que otras máquinas de la infraestructura se especialicen en medidas de protección. Por último, **Egida** pretende detectar de forma temprana

los errores antes de los despliegues, para que éstos puedan completarse en menos tiempo.

1.3 Organización del Documento

El siguiente documento está organizado en 10 capítulos cuyo contenido se define a continuación:

- **Capítulo 1**: Contiene la introducción del proyecto, la motivación y la finalidad del proyecto.
- **Capítulo 2**: Se corresponde con la lista de objetivos del proyecto, así como sus posibles ámbitos de aplicación.
- **Capítulo 3**: Contiene la documentación asociada a las tres etapas de planificación y gestión del proyecto (Planificación del Proyecto, Ejecución del Proyecto y Cierre del Proyecto).
- **Capítulo 4**: En este capítulo se detalla el estado actual de los conocimientos científico técnicos relacionados con la temática del proyecto y proyectos relacionados.
- **Capítulo 5**: Describe el funcionamiento y arquitectura del sistema propuesto.
- **Capítulo 6**: Se corresponde con la descripción de todos los procesos relacionados con la metodología del trabajo seguida durante el proyecto.
- **Capítulo 7**: Contiene los resultados obtenidos en el proyecto, así como una interpretación de estos y su posterior discusión.
- **Capítulo 8**: En este capítulo se detallan las conclusiones y trabajo futuro del proyecto, así como la difusión de los resultados.
- **Bibliografía**: Se corresponde con las referencias bibliográficas del proyecto.
- **Anexos**: Otros documentos

CAPÍTULO

2

ÁMBITOS DE APLICACIÓN

2.1 Posibles Ámbitos de Aplicación

Como se ha mencionado anteriormente, el principal ámbito de aplicación del sistema propuesto en este proyecto de investigación es el educativo.

Dentro del ámbito educativo, **Egida** puede utilizarse para dos aplicaciones diferentes:

- **Como entorno educativo para aprender los diferentes tipos de controles de seguridad existentes y su impacto en el sistema**. Gracias a la interfaz gráfica de usuario (GUI) que proporciona **Egida**, se puede utilizar para estudiar las diferentes tareas de seguridad divididas en secciones o ámbitos de aplicación y comprobar que efecto tienen en el sistema. Además, **Egida** permite realizar auditorías de seguridad con herramientas conocidas como *Lynis*. De esta forma, se pueden aplicar los controles de seguridad a un sistema y evaluar como varían el resultado de estas auditorías.

- ***Hardening* de los sistemas informáticos del centro educativo**. Los centros educativos como las Universidades suelen contar con un gran número de sistemas informáticos que no tienen aplicadas medidas de seguridad o que directamente tienen una instalación de fábrica. Con **Egida**, el administrador de sistemas puede desarrollar un script en el DSL de **Egida** que especifique todos los controles de seguridad que se pueden aplicar a cada uno de los equipos en función de sus necesidades y automatizar su implementación.

No obstante, se plantea la solución propuesta **Egida** para que también pueda ser utilizado en el futuro en entornos profesionales.

CAPÍTULO

3

PLANIFICACIÓN Y GESTIÓN DEL TFM

3.1 Planificación del Proyecto

3.1.1 Identificación de Interesados

En el caso específico de este trabajo de investigación se ha tenido en cuenta la siguiente lista de interesados o stakeholders (ver Tabla 3.1).

Nombre	Cargo	Departamento
Antonio Payá González	Tutorando	Universidad de Oviedo
Alba Cotarelo Tuñón	Estudiante	Universidad de Oviedo
José Manuel Redondo López	Tutor del Proyecto	Universidad de Oviedo
Alumnos y profesores de seguridad y administración de sistemas	Posibles usuarios	
Administradores de sistemas	Posibles usuarios	

TABLA 3.1: Lista de interesados o stakeholders

3.1.2 OBS y PBS

Para la realización del proyecto se ha contado con la colaboración de tres perfiles diferentes (ver Tabla 3.2).

Nombre del recurso	Tipo	Iniciales	Grupo	Capacidad máxima	Tasa estándar	Tasa horas extra	Costo/Uso	Acumular	Calendario base	Código
Antonio Payá González	Trabajo	A	Principal	100 %	26,00 €/hora	0,00 €/hora	0,00 €	Prorrateo	Estándar	A
José Manuel Redondo López	Trabajo	J	Tutor	20 %	30,00 €/hora	0,00 €/hora	0,00 €	Prorrateo	Estándar	J
Alba Cotarelo Tuñón	Trabajo	C	Consultora	20 %	26,00 €/hora	0,00 €/hora	0,00 €	Prorrateo	Estándar	C

TABLA 3.2: Hoja de recursos del proyecto

La estructura de desglose del producto o PBS que se corresponde a la lista de entregables del proyecto se encuentra detallado en la Ilustración 3.1.

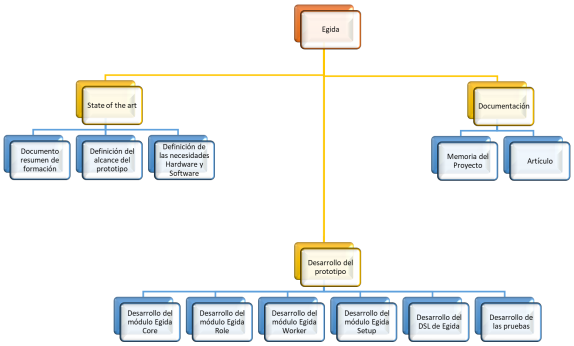

FIGURA 3.1: PBS del proyecto

3.1.3 Planificación inicial. WBS

En la Tabla 3.3, se puede observar el WBS resumido del proyecto con la planificación de las tareas divididas en seis grupos.

EDT	Nombre de tarea	Duración	Comienzo	Fin
1	Reuniones mensuales	170,5 días	jue 07/01/21	jue 06/05/21
2	Documentación e Investigación	160 días	jue 07/01/21	jue 29/04/21
3	Definición y diseño del prototipo	86,5 días	jue 07/01/21	mar 09/03/21
4	Desarrollo del prototipo	50,75 días	jue 28/01/21	vie 05/03/21
5	Desarrollo de los contenidos del Proyecto	53,5 días	vie 05/03/21	lun 12/04/21
6	Desarrollo de la Memoria del Proyecto	64 días	lun 08/03/21	mié 21/04/21
7	Desarrollo del Artículo del Proyecto	32,75 días	mié 21/04/21	jue 13/05/21

TABLA 3.3: Tabla de planificación de tareas resumida del proyecto

A continuación, se puede observar la Tabla de planificación de tareas completa del proyecto (Ver Tabla 3.4). Únicamente se muestran los campos EDT, Nombre de tarea, Duración, Comienzo, Fin y Predecesoras.

EDT	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Reuniones mensuales	170,5 días	jue 07/01/21	jue 06/05/21	
1.1	Reuniones mensuales 1	2 horas	jue 07/01/21	jue 07/01/21	
1.2	Reuniones mensuales 2	2 horas	jue 04/02/21	jue 04/02/21	
1.3	Reuniones mensuales 3	2 horas	jue 04/03/21	jue 04/03/21	
1.4	Reuniones mensuales 4	2 horas	jue 01/04/21	jue 01/04/21	
1.5	Reuniones mensuales 5	2 horas	jue 06/05/21	jue 06/05/21	
2	Documentación e Investigación	160 días	jue 07/01/21	jue 29/04/21	
2.1	Exploración de las herramientas de hardening actuales	10 días	jue 07/01/21	jue 14/01/21	2
2.2	Exploración de las normas y estándares de seguridad de Sistemas Operativos	8 días	vie 15/01/21	mié 20/01/21	2
2.3	Recopilación de reviews y papers en Automate-Hardening	5 días	jue 15/04/21	lun 19/04/21	5
2.4	Exploración principales vulnerabilidades en Sistemas Operativos	10 días	jue 14/01/21	mié 27/01/21	2
2.5	Recopilación de reviews y papers en DSLs	5 días	jue 08/04/21	lun 12/04/21	5
2.6	Documento recopilatorio de reviews y papers relevantes para el proyecto	0 días	jue 29/04/21	jue 29/04/21	10;12
2.7	Formación inicial necesaria de ciberseguridad	0 días	mié 27/01/21	mié 27/01/21	8;9;11
3	Definición y diseño del prototipo	86,5 días	jue 07/01/21	mar 09/03/21	

3.1	Descripción inicial del prototipo	0,25 días	jue 07/01/21	jue 07/01/21	
3.1.1	Descripción general del prototipo	1 hora	jue 07/01/21	jue 07/01/21	2
3.1.2	Descripción extensa del prototipo	1 hora	jue 07/01/21	jue 07/01/21	2
3.1.3	Realización del documento de descripción inicial del prototipo	0 días	jue 07/01/21	jue 07/01/21	17;18
3.2	Definición del alcance del prototipo	54 días	mié 27/01/21	vie 05/03/21	
3.2.1	Definición del alcance de los módulos	2,75 días	mié 27/01/21	jue 28/01/21	
3.2.1.1	Definición del alcance de Egida Core	2 horas	mié 27/01/21	mié 27/01/21	16
3.2.1.2	Definición del alcance de Egida Setup	2 horas	mié 27/01/21	jue 28/01/21	22
3.2.1.3	Definición del alcance de Egida API	2 horas	jue 28/01/21	jue 28/01/21	23
3.2.1.4	Definición del alcance de Egida Role	2 horas	jue 28/01/21	jue 28/01/21	24
3.2.1.5	Realización del documento de descripción del alcance de los módulos	0 días	jue 28/01/21	jue 28/01/21	22;23;24;25
3.2.2	Definición del alcance de las pruebas	2 días	jue 04/03/21	vie 05/03/21	
3.2.2.1	Definición del alcance de las pruebas unitarias	1 hora	jue 04/03/21	jue 04/03/21	4
3.2.2.2	Definición del alcance de las pruebas de aceptación	2 horas	jue 04/03/21	vie 05/03/21	4
3.2.2.3	Definición del alcance de las pruebas de rendimiento	2 horas	jue 04/03/21	vie 05/03/21	4
3.2.2.4	Realización del documento de descripción del alcance de las pruebas	0 días	vie 05/03/21	vie 05/03/21	28;29;30
3.3	Diseño del prototipo	3,25 días	vie 05/03/21	mar 09/03/21	
3.3.1	Arquitectura del prototipo	3 horas	vie 05/03/21	lun 08/03/21	21;27
3.3.2	Diagrama de componentes	2 horas	lun 08/03/21	lun 08/03/21	21;27
3.3.3	Diagrama de despliegue	4 horas	mar 09/03/21	mar 09/03/21	21;27
3.3.4	Realización de los diagramas referentes al diseño del prototipo	0 días	mar 09/03/21	mar 09/03/21	33;34;35
4	Desarrollo del prototipo	50,75 días	jue 28/01/21	vie 05/03/21	

4.1	Desarrollo de los módulos	43 días	jue 28/01/21	lun 01/03/21	
4.1.1	Desarrollo del módulo Egida Core	28,75 días	lun 01/02/21	vie 19/02/21	
4.1.1.1	Desarrollo de la interfaz CLI de Egida Menu	2 días	lun 01/02/21	jue 11/02/21	24
4.1.1.2	Desarrollo de la interfaz CLI de Egida Info	1 día	lun 01/02/21	lun 01/02/21	24
4.1.1.3	Desarrollo de la interfaz CLI de Egida Config	1 día	mar 02/02/21	mar 02/02/21	24
4.1.1.4	Desarrollo de la interfaz CLI de Egida Compile	6 días	mar 02/02/21	lun 15/02/21	24
4.1.1.5	Desarrollo del servicio Playbook	2 días	jue 04/02/21	mié 10/02/21	24
4.1.1.6	Desarrollo del DSL Aspida	21 días	jue 04/02/21	vie 19/02/21	
4.1.1.7	Realización del módulo Egida Core	0 días	vie 19/02/21	vie 19/02/21	40; 41; 42; 43; 44; 45
4.1.2	Desarrollo del módulo Egida API	2,25 días	jue 28/01/21	vie 29/01/21	
4.1.2.1	Desarrollo del servicio Packages	3 horas	vie 29/01/21	vie 29/01/21	24
4.1.2.2	Desarrollo del servicio Hardening	5 horas	jue 28/01/21	vie 29/01/21	24
4.1.2.3	Desarrollo del servicio Services	3 horas	vie 29/01/21	vie 29/01/21	24
4.1.2.4	Realización del módulo Egida API	0 días	vie 29/01/21	vie 29/01/21	52;53;54
4.1.3	Desarrollo del módulo Egida Setup	11,5 días	vie 19/02/21	lun 01/03/21	
4.1.3.1	Desarrollo del Rol Ansible	2 horas	vie 26/02/21	lun 01/03/21	24
4.1.3.2	Integración con Egida Core	2 horas	vie 19/02/21	vie 19/02/21	24
4.1.3.3	Realización del módulo Egida Setup	0 días	lun 01/03/21	lun 01/03/21	57;58
4.1.4	Desarrollo del módulo Egida Role	10,5 días	vie 19/02/21	vie 26/02/21	
4.1.4.1	Desarrollo del Rol Ansible	10 días	vie 19/02/21	vie 26/02/21	24
4.1.4.2	Integración con Egida Core	2 horas	vie 26/02/21	vie 26/02/21	24
4.1.4.3	Realización del módulo Egida Role	0 días	vie 26/02/21	vie 26/02/21	61;62
4.2	Desarrollo de las pruebas	7,75 días	lun 01/03/21	vie 05/03/21	
4.2.1	Desarrollo de las pruebas unitarias	5 días	mar 02/03/21	jue 04/03/21	38
4.2.2	Desarrollo de las pruebas de aceptación	1 día	lun 01/03/21	mar 02/03/21	38

4.2.3	Desarrollo de las pruebas de rendimiento	1 día	lun 01/03/21	lun 01/03/21	38
4.2.4	Realización del desarrollo de las pruebas	0 días	vie 05/03/21	vie 05/03/21	65;66;67
5	Desarrollo de los contenidos del Proyecto	53,5 días	vie 05/03/21	lun 12/04/21	
5.1	Resumen detallado del proyecto	1 día	mar 16/03/21	mar 16/03/21	37
5.2	Objetivos principales del proyecto	1 día	mié 17/03/21	mié 17/03/21	37
5.3	Trabajo relacionado del proyecto	15 días	mar 09/03/21	vie 19/03/21	37
5.4	Descripción detallada del sistema	2 días	lun 22/03/21	lun 22/03/21	37
5.5	Descripción de la metodología del proyecto	5 días	mar 23/03/21	lun 12/04/21	37;73
5.6	Resultados y discusiones del proyecto	1,75 días	vie 05/03/21	jue 25/03/21	37
5.7	Conclusiones y trabajo futuro del proyecto	0,5 días	vie 05/03/21	vie 05/03/21	37
5.8	Realización del documento general de contenidos del proyecto	0 días	lun 12/04/21	lun 12/04/21	70; 71; 72; 73; 74; 75; 76
6	Desarrollo de la Memoria del Proyecto	64 días	lun 08/03/21	mié 21/04/21	
6.1	Introducción	2 horas	mié 17/03/21	vie 26/03/21	70
6.2	Fijación de Objetivos	2 horas	mié 17/03/21	lun 19/04/21	71
6.3	Planificación y Gestión del TFM	4 horas	mar 13/04/21	mar 20/04/21	69
6.4	Estado Actual de los Conocimientos Científico-Técnicos	2 días	mié 24/03/21	vie 26/03/21	72
6.5	Descripción del Sistema	4 días	mar 23/03/21	lun 29/03/21	73
6.6	Metodología de Trabajo	2 días	lun 12/04/21	mar 13/04/21	74
6.7	Resultados Obtenidos	4 horas	jue 25/03/21	lun 29/03/21	75
6.8	Conclusiones y Trabajo Futuro	5 horas	lun 08/03/21	lun 08/03/21	76
6.9	Realización de la Memoria del Proyecto	0 días	mié 21/04/21	mié 21/04/21	79; 80; 81; 82; 83; 84; 85; 86
7	Desarrollo del Artículo del Proyecto	32,75 días	mié 21/04/21	jue 13/05/21	
7.1	Abstract	8 días	lun 26/04/21	lun 03/05/21	78

7.2	Resumen	10 días	vie 30/04/21	vie 07/05/21	78	
7.3	State of the Art	20 días	mié 21/04/21	mié 12/05/21	78	
7.4	Contenido	10 días	vie 23/04/21	mar 11/05/21	78	
7.5	Evaluación y discusión	9 días	mié 21/04/21	mié 12/05/21	78	
7.6	Conclusiones	5 días	mié 21/04/21	jue 13/05/21	78	
7.7	Trabajo Futuro	1 día	mié 21/04/21	mié 21/04/21	78	
7.8	Realización del Artículo del Proyecto	0 días	jue 13/05/21	jue 13/05/21	89; 91; 93; 95	90; 92; 94;

TABLA 3.4: Tabla de planificación de tareas del proyecto

3.1.4 Riesgos del Proyecto

Plan de Gestión de Riesgos

El siguiente apartado hace referencia al "Plan de Gestión de Riesgos" en el capítulo de los Anexos A al Plan de Gestión de Riesgos del Proyecto.

Identificación de Riesgos

A continuación, se muestra una tabla con el conjunto de riesgos identificados para este proyecto incluyendo la siguiente información:

- **ID**: Identificador del riesgo.
- **Nombre**: Descripción detallada de lo que significa el riesgo.
- Responsable: Personas o entidades implicadas en el riesgo.
- **Probabilidad**: Es la probabilidad de que el riesgo se manifieste. Las escalas de probabilidad se definirán en la "Matriz de Probabilidad e Impacto" dentro del Plan de Gestión de Riesgos del proyecto.
- **Impacto**: Impacto que el riesgo tiene en el proyecto, para calcular el impacto se utilizará "Matriz de Probabilidad e Impacto". Además, se asociará un valor dependiendo del impacto previsto en el presupuesto, la planificación, el alcance o la calidad del proyecto.
- **Priorización**: En una escala de 0 a 0.5, cuanto de importante es mantener el seguimiento de este riesgo.
- **Respuesta**: Que se hará si el riesgo ocurre.

Registro de Riesgos

ID	Nombre	Responsable
1	Baja de un empleado clave para el desarrollo del proyecto	Responsable de Recursos Humanos
2	Poca documentación sobre nuevos avances en ciberseguridad	Responsable Tecnológico
3	Aparición de una nueva vulnerabilidad de un Sistema Operativo	Responsable Tecnológico
4	Retraso en el desarrollo del módulo Egida Core del prototipo	Responsable de Compras
5	Retraso en el desarrollo del módulo Egida DSL del prototipo	Responsable Tecnológico
6	Retraso en el desarrollo del módulo Egida Setup del prototipo	Responsable Tecnológico
7	Retraso en el desarrollo del módulo Egida Worker del prototipo	Responsable Tecnológico
8	Retraso en el desarrollo del módulo Egida Role del prototipo	Responsable Tecnológico
9	Latencia y mal rendimiento del sistema en un entorno de producción	Responsable Tecnológico
10	Pocos repositorios de vulnerabilidades y respuestas a vulnerabilidades disponibles a un coste aceptable	Responsable de Compras
11	El sistema no cubre correctamente y de forma positiva los casos de uso diseñados	Responsable Tecnológico
12	Bajas temporales en el personal del proyecto a causa del COVID-19	Responsable de Recursos Humanos
13	Nueva versión no contemplada en la documentación de las métricas de seguridad utilizadas	Responsable Tecnológico
14	Retraso en la redacción final del artículo	Responsable Tecnológico
15	Retraso en la redacción final de la memoria	Responsable Tecnológico

TABLA 3.5: Registro de Riesgos

ID	Probabilidad	Impacto				Impacto	0,40
		Presup.	Planific.	Alcance	Calidad		Priorización
1	Baja	Alto	Medio	Crítico	Medio	0,27	
2	Muy Baja	Alto	Crítico	Alto	Alto	0,09	
3	Muy Alta	Alto	Medio	Alto	Alto	0,50	
4	Media	Bajo	Crítico	Alto	Muy Bajo	0,45	
5	Alta	Bajo	Crítico	Alto	Muy Bajo	0,63	
6	Baja	Bajo	Crítico	Alto	Muy Bajo	0,27	
7	Media	Bajo	Crítico	Alto	Muy Bajo	0,45	
8	Baja	Bajo	Crítico	Alto	Alto	0,27	
9	Media	Medio	Medio	Alto	Medio	0,27	
10	Baja	Crítico	Bajo	Alto	Alto	0,28	
11	Muy Baja	Bajo	Alto	Crítico	Alto	0,09	
12	Media	Bajo	Alto	Medio	Medio	0,28	
13	Media	Bajo	Bajo	Medio	Bajo	0,15	
14	Baja	Medio	Medio	Medio	Medio	0,09	
15	Baja	Crítico	Medio	Medio	Medio	0,09	

TABLA 3.6: Registro de Riesgos: Probabilidad e Impacto

ID	Impacto	0,40 Priorización	Response
1	0,27		Asumir. Adaptar el prototipo a la documentación disponible
2	0,09		Asumir. Adaptar el prototipo a la documentación disponible
3	0,50		Mitigar. Hacer uso de repositorios de vulnerabilidades que se actualicen periódicamente.
4	0,45		Mitigar. Asegurarse de que se tienen claros todos los aspectos del desarrollo del módulo Egida Core. Utilizar frameworks y tecnologías conocidas por los desarrolladores.
5	0,63		Mitigar. Asegurarse de que se tienen claros todos los aspectos del desarrollo del módulo Egida DSL. Basarse en lenuguajes ya conocidos para el desarrollo del DSL.
6	0,27		Mitigar. Asegurarse de que se tienen claros todos los aspectos del desarrollo del módulo Egida Setup. Utilizar frameworks y tecnologías con facilidad de despliegue.
7	0,45		Mitigar. Asegurarse de que se tienen claros todos los aspectos del desarrollo del módulo Egida Worker. Utilizar tecnologías ya conocidas por los desarrolladores.
8	0,27		Mitigar. En caso de dificultades con el uso de Ansible, desarrollar scripts que realicen la misma función.
9	0,28		Mitigar. Intentar realizar operaciones que realicen un menor consumo de memoria y CPU, utilizar tecnologías maduras y preparadas para producción.
10	0,27		Aceptar. Adaptar el prototipo a los repositorios disponibles.
11	0,09		Mitigar. Realizar pruebas unitarias y de aceptación durante el desarrollo del prototipo para comprobar si se están cumpliendo los casos de uso definidos.
12	0,27		Aceptar. Adaptar el prototipo a los recursos disponibles.
13	0,15		Aceptar. Adaptar el prototipo en caso de que la nueva versión sea muy diferente a la actual o, en caso de no poder realizar dicha adaptación, referenciar la nueva versión en la documentación.
14	0,09		Mitigar. Pasar de realizar reuniones mensuales a realizar reuniones semanales donde se impongan una serie de metas o baremos de corta duración que favorezcan el correcto avance.
15	0,09		Mitigar. Pasar de realizar reuniones mensuales a realizar reuniones semanales donde se impongan una serie de metas o baremos de corta duración que favorezcan el correcto avance.

TABLA 3.7: Registro de Riesgos: Impacto y Respuesta

3.1.5 Presupuesto Inicial

En este apartado se detallarán todos los costes relativos al proyecto y se especificarán los presupuestos iniciales de costes y de cliente.

Presupuesto de Costes

En la siguiente tabla (Ver 3.8) se detalla el presupuesto de costes del proyecto.

TABLA 3.8: Presupuesto de Costes Inicial del Proyecto

I 1	*I 2*	*I 3*	*I 4*	*Descripción*	*Cantidad*	*Unidades*	*Precio*	*Precio Total*	*Subtotal (4)*	*Subtotal (3)*	*Subtotal (2)*	*Total*
												74.571,20 €
01				**Documentación e Investigación**							11.612,80 €	
	01			**Exploración de las herramientas de hardening actuales**						4.256,00 €		
		01		Ingeniero del Software	40	horas	49,40 €		1.976,00 €			
		02		Director del Proyecto	40	horas	57,00 €		2.280,00 €			
	02			**Exploración de las normas y estándares de seguridad de SSOO**						3.404,80 €		
		01		Ingeniero del Software	32	horas	49,40 €		1.580,80 €			
		02		Director del Proyecto	32	horas	57,00 €		1.824,00 €			
	03			**Recopilación de reviews y papers en Automate-Hardening**						988,00 €		
		01		Ingeniero del Software	20	horas	49,40 €		988,00 €			
	04			**Exploración principales vulnerabilidades en Sistemas Operativos**						1.976,00 €		
		01		Ingeniero del Software	40	horas	49,40 €		1.976,00 €			
	05			**Recopilación de reviews y papers en DSLs**						988,00 €		
		01		Ingeniero del Software	20	horas	49,40 €		988,00 €			
02				**Diseño y definición del prototipo**							2.625,80 €	
	01			**Descripción inicial del prototipo**						155,80 €		
		01		Descripción general del prototipo					155,80 €			
			01	Ingeniero del Software	1	horas	49,40 €	49,40 €				
			02	Jefe de Proyecto	1	horas	57,00 €	57,00 €				
			03	Consultora	1	horas	49,40 €	49,40 €				
	02			**Definición del alcance de los módulos**						1.246,40 €		
		01		Definición del alcance de Egida Core					311,60 €			
			01	Ingeniero del Software	2	horas	49,40 €	98,80 €				

		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €	
		03	Consultora	2	horas	49,40 €	98,80 €	
	02		Definición del alcance de Egida Setup					311,60 €
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €	
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €	
		03	Consultora	2	horas	49,40 €	98,80 €	
	03		Definición del alcance de Egida API					311,60 €
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €	
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €	
		03	Consultora	2	horas	49,40 €	98,80 €	
	04		Definición del alcance de Egida Role					311,60 €
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €	
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €	
		03	Consultora	2	horas	49,40 €	98,80 €	
03			**Definición del alcance de las pruebas**					779,00 €
	01		Definición del alcance de las pruebas unitarias					155,80 €
		01	Ingeniero del Software	1	horas	49,40 €	49,40 €	
		02	Jefe de Proyecto	1	horas	57,00 €	57,00 €	
		03	Consultora	1	horas	49,40 €	49,40 €	
	02		Definición del alcance de las pruebas de aceptación					311,60 €
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €	
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €	
		03	Consultora	2	horas	49,40 €	98,80 €	
	03		Definición del alcance de las pruebas de rendimiento					311,60 €
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €	
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €	
		03	Consultora	2	horas	49,40 €	98,80 €	
04			**Diseño del prototipo**					444,60 €
	01		Arquitectura del prototipo					148,20 €

		01	Ingeniero del Software	3	horas	49,40 €	148,20 €			
	02		Diagrama de componentes					98,80 €		
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €			
	03		Diagrama de despliegue					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
03			**Desarrollo del prototipo**							8.249,80 €
01			**Desarrollo del módulo Egida Core**						5.532,80 €	
	01		Desarrollo de la interfaz CLI de Egida Menu					395,20 €		
		01	Ingeniero del Software	8	horas	49,40 €	395,20 €			
	02		Desarrollo de la interfaz CLI de Egida Info					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
	03		Desarrollo de la interfaz CLI de Egida Config					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
	04		Desarrollo de la interfaz CLI de Egida Compile					1.185,60 €		
		01	Ingeniero del Software	24	horas	49,40 €	1.185,60 €			
	05		Desarrollo del servicio Playbook					395,20 €		
		01	Ingeniero del Software	8	horas	49,40 €	395,20 €			
	06		Desarrollo del DSL Aspida					3.161,60 €		
		01	Ingeniero del Software	64	horas	49,40 €	3.161,60 €			
02			**Desarrollo del módulo Egida API**						543,40 €	
	01		Desarrollo del servicio packages					148,20 €		
		01	Ingeniero del Software	3	horas	49,40 €	148,20 €			
	02		Desarrollo del servicio hardening					247,00 €		
		01	Ingeniero del Software	5	horas	49,40 €	247,00 €			
	03		Desarrollo del servicio services					148,20 €		
		01	Ingeniero del Software	3	horas	49,40 €	148,20 €			
03			**Desarrollo del módulo Egida Setup**						197,60 €	
	01		Desarrollo del rol Ansible					98,80 €		

		01	Ingeniero del Software	2	horas	49,40 €	98,80 €			
	02		Integración con Egida Core					98,80 €		
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €			
	04		**Desarrollo del módulo Egida Role**						592,80 €	
	01		Desarrollo del rol Ansible					494,00 €		
		01	Ingeniero del Software	10	horas	49,40 €	494,00 €			
	02		Integración con Egida Core					98,80 €		
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €			
	05		**Desarrollo de las pruebas**						1.383,20 €	
	01		Desarrollo de las pruebas unitarias					988,00 €		
		01	Ingeniero del Software	20	horas	49,40 €	988,00 €			
	02		Desarrollo de las pruebas de aceptación					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
	03		Desarrollo de las pruebas de rendimiento					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
04			**Desarrollo de los contenidos del Proyecto**							5.187,00 €
01			**Resumen detallado del proyecto**						197,60 €	
	01		Ingeniero del Software	4	horas	49,40 €		197,60 €		
02			**Objetivos principales del proyecto**						197,60 €	
	01		Ingeniero del Software	4	horas	49,40 €		197,60 €		
03			**Trabajo relacionado del proyecto**						2.964,00 €	
	01		Ingeniero del Software	60	horas	49,40 €		2.964,00 €		
04			**Descripción detallada del sistema**						395,20 €	
	01		Ingeniero del Software	8	horas	49,40 €		395,20 €		
05			**Descripción de la metodología del proyecto**						988,00 €	
	01		Ingeniero del Software	20	horas	49,40 €		988,00 €		

06		**Resultados y discusiones del proyecto**					345,80 €	
	01	Ingeniero del Software	7	horas	49,40 €	345,80 €		
07		**Conclusiones y trabajo futuro del proyecto**					98,80 €	
	01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
05		**Desarrollo de la Memoria del Proyecto**						7.634,20 €
01		**Introducción**					311,60 €	
	01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
	02	Director del Proyecto	2	horas	57,00 €	114,00 €		
	03	Consultora	2	horas	49,40 €	98,80 €		
02		**Fijación de objetivos**					311,60 €	
	01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
	02	Director del Proyecto	2	horas	57,00 €	114,00 €		
	03	Consultora	2	horas	49,40 €	98,80 €		
03		**Planificación y gestión del proyecto**					623,20 €	
	01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	02	Director del Proyecto	4	horas	57,00 €	228,00 €		
	03	Consultora	4	horas	49,40 €	197,60 €		
04		**Estado actual de los conocimientos Cientifico-Técnicos**					1.246,40 €	
	01	Ingeniero del Software	8	horas	49,40 €	395,20 €		
	02	Director del Proyecto	8	horas	57,00 €	456,00 €		
	03	Consultora	8	horas	49,40 €	395,20 €		
05		**Descripción del sistema**					2.492,80 €	
	01	Ingeniero del Software	16	horas	49,40 €	790,40 €		
	02	Director del Proyecto	16	horas	57,00 €	912,00 €		
	03	Consultora	16	horas	49,40 €	790,40 €		
06		**Metodología de trabajo**					1.246,40 €	
	01	Ingeniero del Software	8	horas	49,40 €	395,20 €		
	02	Director del Proyecto	8	horas	57,00 €	456,00 €		
	03	Consultora	8	horas	49,40 €	395,20 €		
07		**Resultados obtenidos**					623,20 €	
	01	Ingeniero del Software	4	horas	49,40 €	197,60 €		

	02	Director del Proyecto	4	horas	57,00 €	228,00 €		
	03	Consultora	4	horas	49,40 €	197,60 €		
08		**Conclusiones y trabajo futuro**					779,00 €	
	01	Ingeniero del Software	5	horas	49,40 €	247,00 €		
	02	Director del Proyecto	5	horas	57,00 €	285,00 €		
	03	Consultora	5	horas	49,40 €	247,00 €		
06		**Desarrollo del artículo**						39.261,60 €
01		**Abstract**					4.985,60 €	
	01	Ingeniero del Software	32	horas	49,40 €	1.580,80 €		
	02	Director del Proyecto	32	horas	57,00 €	1.824,00 €		
	03	Consultora	32	horas	49,40 €	1.580,80 €		
02		**Resumen**					6.232,00 €	
	01	Ingeniero del Software	40	horas	49,40 €	1.976,00 €		
	02	Director del Proyecto	40	horas	57,00 €	2.280,00 €		
	03	Consultora	40	horas	49,40 €	1.976,00 €		
03		**State of the Art**					12.464,00 €	
	01	Ingeniero del Software	80	horas	49,40 €	3.952,00 €		
	02	Director del Proyecto	80	horas	57,00 €	4.560,00 €		
	03	Consultora	80	horas	49,40 €	3.952,00 €		
04		**Contenido**					6.232,00 €	
	01	Ingeniero del Software	40	horas	49,40 €	1.976,00 €		
	02	Director del Proyecto	40	horas	57,00 €	2.280,00 €		
	03	Consultora	40	horas	49,40 €	1.976,00 €		
05		**Evaluación y discusión**					5.608,80 €	
	01	Ingeniero del Software	36	horas	49,40 €	1.778,40 €		
	02	Director del Proyecto	36	horas	57,00 €	2.052,00 €		
	03	Consultora	36	horas	49,40 €	1.778,40 €		
06		**Conclusiones**					3.116,00 €	
	01	Ingeniero del Software	20	horas	49,40 €	988,00 €		
	02	Director del Proyecto	20	horas	57,00 €	1.140,00 €		
	03	Consultora	20	horas	49,40 €	988,00 €		
07		**Trabajo futuro**					623,20 €	
	01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	02	Director del Proyecto	4	horas	57,00 €	228,00 €		
	03	Consultora	4	horas	49,40 €	197,60 €		

Presupuesto del Cliente

Presupuesto del Cliente Detallado En la 3.9, se puede observar el presupuesto detallado del cliente junto con sus partidas.

TABLA 3.9: Presupuesto del Cliente Detallado

Partida	Item	Partida	Importe	Total
1		**Investigación y formación en nuevos avances en ciberseguridad**		**11,612.80 €**
	1	Documentación e investigación	9,636.80 €	
	2	Recopilación de reviews y papers	1,976.00 €	
2		**Diseño del prototipo y de las pruebas**		**2,625.80 €**
	1	Diseño del prototipo	1,846.80 €	
	2	Diseño de las pruebas	779.00 €	
3		**Desarrollo del prototipo**		**8,249.80 €**
	1	Egida Core	5,532.80 €	
	2	Egida API	543.40 €	
	3	Egida Setup	197.60 €	
	4	Egida Role	592.80 €	
	5	Desarrollo de las pruebas	1,383.20 €	
4		**Desarrollo de los contenidos del proyecto**		**5,187.00 €**
	1	Resumen detallado del proyecto	197.60 €	
	2	Objetivos principales del proyecto	197.60 €	
	3	Trabajo relacionado del proyecto	2,964.00 €	
	4	Descripción detallada del sistema	395.20 €	
	5	Descripción de la metodología del proyecto	988.00 €	
	6	Resultados y discusiones del proyecto	345.80 €	
	7	Conclusiones y trabajo futuro del proyecto	98.80 €	
5		**Desarrollo de la memoria del proyecto**		**7,634.20 €**
	1	Introducción	311.60 €	
	2	Fijación de objetivos	311.60 €	
	3	Planificación y gestión del proyecto	623.20 €	
	4	Estado actual de los conocimientos Científico-Técnicos	1,246.40 €	
	5	Descripción del sistema	2,492.80 €	
	6	Metodología de trabajo	1,246.40 €	
	7	Resultados obtenidos	623.20 €	
	8	Conclusiones y trabajo futuro	779.00 €	
6		**Desarrollo del Artículo**		**39,261.60 €**
	1	Abstract	4,985.60 €	
	2	Resumen	6,232.00 €	
	3	State of the Art	12,464.00 €	
	4	Contenido	6,232.00 €	
	5	Evaluación y discusión	5,608.80 €	
	6	Conclusiones	3,116.00 €	
	7	Trabajo futuro	623.20 €	
TOTAL CLIENTE				**74,571.20 €**

Presupuesto del Cliente Resumido En la 3.10, se muestra el presupuesto resumido del cliente junto con un gráfico resumen de los diferentes precios de las partidas principales.

Cod.	Partida	Total
1	Investigación y formación en nuevos avances en ciberseguridad	11,612.80 €
2	Diseño del prototipo y de las pruebas	2,625.80 €
3	Desarrollo del prototipo	8,249.80 €
4	Desarrollo de los contenidos del proyecto	5,187.00 €
5	Desarrollo de la memoria del proyecto	7,634.20 €
6	Desarrollo del artículo	39,261.60 €
TOTAL CLIENTE		**74,571.20 €**

TABLA 3.10: Presupuesto Resumido del Cliente

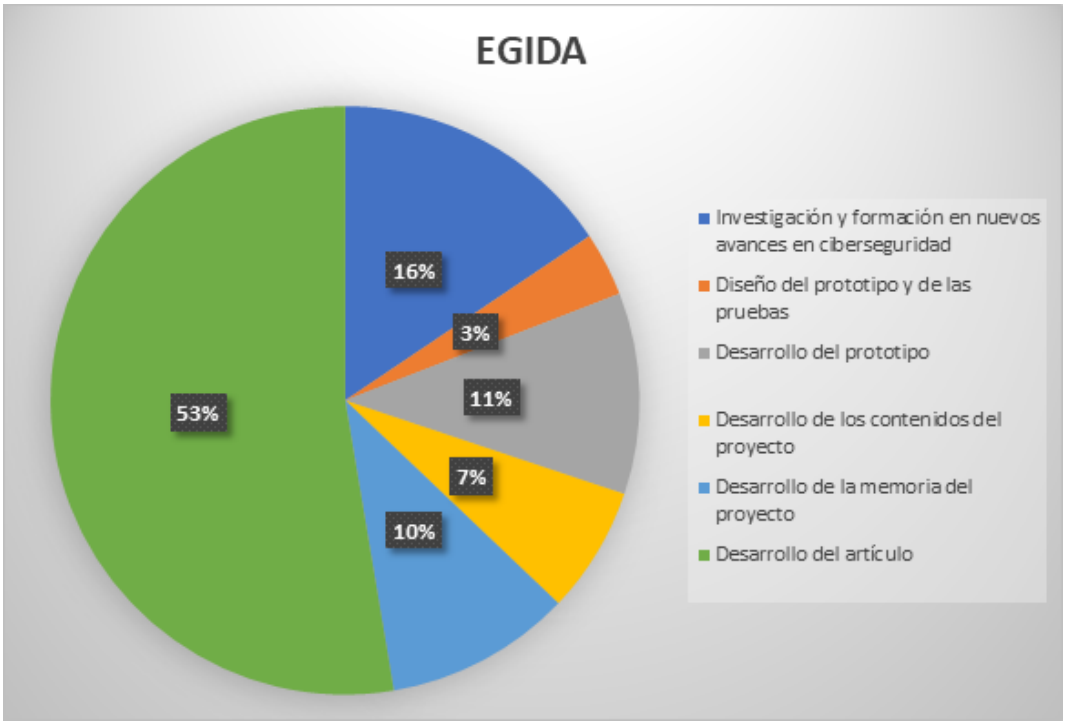

FIGURA 3.2: Presupuesto Resumido del Cliente

3.2 Ejecución del Proyecto

3.2.1 Plan de Seguimiento de la Planificación

En este apartado se detallará el seguimiento de las siete tareas principales que componen al proyecto de investigación (Reuniones semanales, Documentación e Investigación, Definición y diseño del Prototipo, Desarrollo del Prototipo, Desarrollo de los contenidos del Proyecto, Desarrollo de la Memoria y Desarrollo del Artículo).

En primer lugar, se realizaron todas las tareas correspondientes a Documentación e Investigación ya que eran necesarias para obtener cierto conocimiento sobre el estado actual del trabajo relacionado con el proyecto. Durante esta fase se identificaron herramientas útiles para el proyecto, así como posibles alternativas o competidores.

Una vez finalizada esta tarea se procedió con el diseño y definición de todas las características del prototipo del proyecto de investigación y su posterior desarrollo. Posteriormente, ya habiendo desarrollado el prototipo y todos sus módulos y pruebas, se procederá a la definición de los aspectos relacionados con la documentación del proyecto, metodología y resultados.

Finalmente, se realizará una inspección sobre el desarrollo del proyecto para poder redactar el informe de cierre de proyecto, planificación y presupuesto finales, así como un informe con las lecciones aprendidas durante el desarrollo del proyecto.

3.2.2 Bitácora de Incidencias del Proyecto

A continuación, se comentará un resumen de las incidencias que han ido ocurriendo durante el desarrollo del proyecto:

- El tiempo dedicado a la investigación del estado actual del trabajo relacionado debió ser mayor, ya que se ha tenido que ir dedicando horas extras para investigar más a fondo posibles alternativas.
- El desarrollo de la documentación podría haber llevado menos tiempo si se hubiese hecho en primer lugar el artículo y posteriormente la memoria. Esto es porque una vez realizado el artículo se han tenido que modificar apartados de la memoria para que sigan manteniendo el mismo formato.

3.2.3 Riesgos

TABLA 3.11: Seguimiento de los Riesgos del Proyecto

ID	Nombre	Respuesta	Seguimiento
1	Baja de un empleado clave para el desarrollo del proyecto		No se ha producido ninguna baja
2	Poca documentación sobre nuevos avances en ciberseguridad	Se ha adaptado el prototipo a la información disponible	Se han encontrado poca documentación únicamente en aspectos muy noveles del trabajo relacionado

3	Aparición de una nueva vulnerabilidad de un Sistema Operativo	Se han utilizado herramientas como los CIS Benchmarks actualizados y mantenidos por empresas con experiencia en el campo	Se han encontrado repositorios actualizados de vulnerabilidades existentes de productos software
4	Retraso en el desarrollo del módulo Egida Core del prototipo		No se ha producido retraso en el desarrollo del módulo
5	Retraso en el desarrollo del módulo Egida DSL del prototipo		No se ha producido retraso en el desarrollo del módulo
6	Retraso en el desarrollo del módulo Egida Setup del prototipo		No se ha producido retraso en el desarrollo del módulo
7	Retraso en el desarrollo del módulo Egida Worker del prototipo		No se ha producido retraso en el desarrollo del módulo
8	Retraso en el desarrollo del módulo Egida Role del prototipo		No se ha producido retraso en el desarrollo del módulo
9	Latencia y mal rendimiento del sistema en un entorno de producción	Se han utilizado herramientas probadas en entornos de producción reales como Ansible para la automatización de las tareas	El prototipo debe de realizar un gran número de tareas que pueden afectar al rendimiento del sistema
10	Pocos repositorios de vulnerabilidades y respuestas a vulnerabilidades disponibles a un coste aceptable		Se ha encontrado un repositorio actualizado y gratuito
11	El sistema no cubre correctamente y de forma positiva los casos de uso diseñados		El sistema cubre correctamente todos los casos de uso diseñados
12	Bajas temporales en el personal del proyecto a causa del COVID-19		No se han producido bajas a causa del COVID-19
13	Nueva versión no contemplada en la documentación de las métricas de seguridad utilizadas		Se ha producido una nueva versión de los CIS Benchmarks pero se ha podido actualizar correctamente el proyecto de acuerdo a esta versión
14	Retraso en la redacción final del artículo		No se ha producido retraso en el desarrollo del artículo

14	Retraso en la redacción final de la memoria	No se ha producido retraso en el desarrollo de la memoria

3.3 Cierre del Proyecto

3.3.1 Planificación Final

En la siguiente tabla (Ver 3.12), se puede observar el WBS final resumido del proyecto con la planificación de las tareas divididas en seis grupos.

EDT	Nombre de tarea	Duración	Comienzo	Fin
1	Reuniones mensuales	170,5 días	jue 07/01/21	jue 06/05/21
2	Documentación e Investigación	161 días	jue 07/01/21	jue 29/04/21
3	Definición y diseño del prototipo	86,5 días	jue 07/01/21	mar 09/03/21
4	Desarrollo del prototipo	50,75 días	jue 28/01/21	vie 05/03/21
5	Desarrollo de los contenidos del Proyecto	53,5 días	vie 05/03/21	lun 12/04/21
6	Desarrollo de la Memoria del Proyecto	64 días	lun 08/03/21	mié 21/04/21
7	Desarrollo del Artículo del Proyecto	32,75 días	mié 21/04/21	jue 13/05/21

TABLA 3.12: Tabla de tareas final resumida del proyecto

A continuación (Ver 3.13), se puede observar la Tabla de planificación de tareas completa final del proyecto (únicamente se muestran los campos EDT, Nombre de tarea, Duración, Comienzo, Fin y Predecesoras). Como se puede observar, se ha producido un retraso con respecto a la planificación inicial en las subtareas de Documentación e Investigación.

TABLA 3.13: Planificación Final del Proyecto

EDT	Nombre de tarea	Duración	Comienzo	Fin
1	Reuniones mensuales	170,5 días	jue 07/01/21	jue 06/05/21
1.1	Reuniones mensuales 1	2 horas	jue 07/01/21	jue 07/01/21
1.2	Reuniones mensuales 2	2 horas	jue 04/02/21	jue 04/02/21
1.3	Reuniones mensuales 3	2 horas	jue 04/03/21	jue 04/03/21
1.4	Reuniones mensuales 4	2 horas	jue 01/04/21	jue 01/04/21
1.5	Reuniones mensuales 5	2 horas	jue 06/05/21	jue 06/05/21
2	Documentación e Investigación	161 días	jue 07/01/21	jue 29/04/21
2.1	Exploración de las herramientas de hardening actuales	15 días	jue 07/01/21	mié 31/03/21
2.2	Exploración de las normas y estándares de seguridad de Sistemas Operativos	12 días	vie 15/01/21	mar 30/03/21
2.3	Recopilación de reviews y papers en Automate-Hardening	5 días	mié 14/04/21	vie 16/04/21
2.4	Exploración principales vulnerabilidades en Sistemas Operativos	10 días	jue 14/01/21	mié 27/01/21
2.5	Recopilación de reviews y papers en DSLs	6 días	jue 08/04/21	mié 14/04/21

2.6	Documento recopilatorio de reviews y papers relevantes para el proyecto	0 días	jue 29/04/21	jue 29/04/21
2.7	Formación inicial necesaria de ciberseguridad	0 días	mié 31/03/21	mié 31/03/21
3	Definición y diseño del prototipo	86,5 días	jue 07/01/21	mar 09/03/21
3.1	Descripción inicial del prototipo	0,25 días	jue 07/01/21	jue 07/01/21
3.1.1	Descripción general del prototipo	1 hora	jue 07/01/21	jue 07/01/21
3.1.2	Descripción extensa del prototipo	1 hora	jue 07/01/21	jue 07/01/21
3.1.3	Realización del documento de descripción inicial del prototipo	0 días	jue 07/01/21	jue 07/01/21
3.2	Definición del alcance del prototipo	54 días	mié 27/01/21	vie 05/03/21
3.2.1	Definición del alcance de los módulos	2,75 días	mié 27/01/21	jue 28/01/21
3.2.1.1	Definición del alcance de Egida Core	2 horas	mié 27/01/21	mié 27/01/21
3.2.1.2	Definición del alcance de Egida Setup	2 horas	mié 27/01/21	jue 28/01/21
3.2.1.3	Definición del alcance de Egida API	2 horas	jue 28/01/21	jue 28/01/21
3.2.1.4	Definición del alcance de Egida Role	2 horas	jue 28/01/21	jue 28/01/21
3.2.1.5	Realización del documento de descripción del alcance de los módulos	0 días	jue 28/01/21	jue 28/01/21
3.2.2	Definición del alcance de las pruebas	2 días	jue 04/03/21	vie 05/03/21
3.2.2.1	Definición del alcance de las pruebas unitarias	1 hora	jue 04/03/21	jue 04/03/21
3.2.2.2	Definición del alcance de las pruebas de aceptación	2 horas	jue 04/03/21	vie 05/03/21
3.2.2.3	Definición del alcance de las pruebas de rendimiento	2 horas	jue 04/03/21	vie 05/03/21
3.2.2.4	Realización del documento de descripción del alcance de las pruebas	0 días	vie 05/03/21	vie 05/03/21
3.3	Diseño del prototipo	3,25 días	vie 05/03/21	mar 09/03/21
3.3.1	Arquitectura del prototipo	3 horas	vie 05/03/21	lun 08/03/21
3.3.2	Diagrama de componentes	2 horas	lun 08/03/21	lun 08/03/21
3.3.3	Diagrama de despliegue	4 horas	mar 09/03/21	mar 09/03/21
3.3.4	Realización de los diagramas referentes al diseño del prototipo	0 días	mar 09/03/21	mar 09/03/21
4	Desarrollo del prototipo	50,75 días	jue 28/01/21	vie 05/03/21
4.1	Desarrollo de los módulos	43 días	jue 28/01/21	lun 01/03/21

4.1.1	Desarrollo del módulo Egida Core	28,75 días	lun 01/02/21	vie 19/02/21
4.1.1.1	Desarrollo de la interfaz CLI de Egida Menu	2 días	lun 01/02/21	jue 11/02/21
4.1.1.2	Desarrollo de la interfaz CLI de Egida Info	1 día	lun 01/02/21	lun 01/02/21
4.1.1.3	Desarrollo de la interfaz CLI de Egida Config	1 día	mar 02/02/21	mar 02/02/21
4.1.1.4	Desarrollo de la interfaz CLI de Egida Compile	6 días	mar 02/02/21	lun 15/02/21
4.1.1.5	Desarrollo del servicio Playbook	2 días	jue 04/02/21	mié 10/02/21
4.1.1.6	Desarrollo del DSL Aspida	21 días	jue 04/02/21	vie 19/02/21
4.1.1.6.1	Desarrollo de la gramática	1 día	jue 04/02/21	vie 12/02/21
4.1.1.6.2	Desarrollo de parser	2 días	jue 04/02/21	lun 15/02/21
4.1.1.6.3	Desarrollo del visitor de generación de código	10 días	jue 04/02/21	jue 18/02/21
4.1.1.6.4	Conexión con el servicio Playbook	3 días	jue 04/02/21	vie 19/02/21
4.1.1.7	Realización del módulo Egida Core	0 días	vie 19/02/21	vie 19/02/21
4.1.2	Desarrollo del módulo Egida API	2,25 días	jue 28/01/21	vie 29/01/21
4.1.2.1	Desarrollo del servicio Packages	3 horas	vie 29/01/21	vie 29/01/21
4.1.2.2	Desarrollo del servicio Hardening	5 horas	jue 28/01/21	vie 29/01/21
4.1.2.3	Desarrollo del servicio Services	3 horas	vie 29/01/21	vie 29/01/21
4.1.2.4	Realización del módulo Egida API	0 días	vie 29/01/21	vie 29/01/21
4.1.3	Desarrollo del módulo Egida Setup	11,5 días	vie 19/02/21	lun 01/03/21
4.1.3.1	Desarrollo del Rol Ansible	2 horas	vie 26/02/21	lun 01/03/21
4.1.3.2	Integración con Egida Core	2 horas	vie 19/02/21	vie 19/02/21
4.1.3.3	Realización del módulo Egida Setup	0 días	lun 01/03/21	lun 01/03/21
4.1.4	Desarrollo del módulo Egida Role	10,5 días	vie 19/02/21	vie 26/02/21
4.1.4.1	Desarrollo del Rol Ansible	10 días	vie 19/02/21	vie 26/02/21
4.1.4.2	Integración con Egida Core	2 horas	vie 26/02/21	vie 26/02/21
4.1.4.3	Realización del módulo Egida Role	0 días	vie 26/02/21	vie 26/02/21
4.2	Desarrollo de las pruebas	7,75 días	lun 01/03/21	vie 05/03/21
4.2.1	Desarrollo de las pruebas unitarias	5 días	mar 02/03/21	jue 04/03/21
4.2.2	Desarrollo de las pruebas de aceptación	1 día	lun 01/03/21	mar 02/03/21
4.2.3	Desarrollo de las pruebas de rendimiento	1 día	lun 01/03/21	lun 01/03/21
4.2.4	Realización del desarrollo de las pruebas	0 días	vie 05/03/21	vie 05/03/21

5	Desarrollo de los contenidos del Proyecto	53,5 días	vie 05/03/21	lun 12/04/21
5.1	Resúmen detallado del proyecto	1 día	mar 16/03/21	mar 16/03/21
5.2	Objetivos principales del proyecto	1 día	mié 17/03/21	mié 17/03/21
5.3	Trabajo relacionado del proyecto	15 días	mar 09/03/21	vie 19/03/21
5.4	Descripción detallada del sistema	2 días	lun 22/03/21	lun 22/03/21
5.5	Descripción de la metodología del proyecto	5 días	mar 23/03/21	lun 12/04/21
5.6	Resultados y discusiones del proyecto	1,75 días	vie 05/03/21	jue 25/03/21
5.7	Conclusiones y trabajo futuro del proyecto	0,5 días	vie 05/03/21	vie 05/03/21
5.8	Realización del documento general de contenidos del proyecto	0 días	lun 12/04/21	lun 12/04/21
6	Desarrollo de la Memoria del Proyecto	64 días	lun 08/03/21	mié 21/04/21
6.1	Introducción	2 horas	mié 17/03/21	vie 26/03/21
6.2	Fijación de Objetivos	2 horas	mié 17/03/21	lun 19/04/21
6.3	Planificación y Gestión del TFM	4 horas	mar 13/04/21	mar 20/04/21
6.4	Estado Actual de los Conocimientos Científico-Técnicos	2 días	mié 24/03/21	vie 26/03/21
6.5	Descripción del Sistema	4 días	mar 23/03/21	lun 29/03/21
6.6	Metodología de Trabajo	2 días	lun 12/04/21	mar 13/04/21
6.7	Resultados Obtenidos	4 horas	jue 25/03/21	lun 29/03/21
6.8	Conclusiones y Trabajo Futuro	5 horas	lun 08/03/21	lun 08/03/21
6.9	Realización de la Memoria del Proyecto	0 días	mié 21/04/21	mié 21/04/21
7	Desarrollo del Artículo del Proyecto	32,75 días	mié 21/04/21	jue 13/05/21
7.1	Abstract	8 días	lun 26/04/21	lun 03/05/21
7.2	Resumen	10 días	vie 30/04/21	vie 07/05/21
7.3	State of the Art	20 días	mié 21/04/21	mié 12/05/21
7.4	Contenido	10 días	vie 23/04/21	mar 11/05/21
7.5	Evaluación y discusión	9 días	mié 21/04/21	mié 12/05/21
7.6	Conclusiones	5 días	mié 21/04/21	jue 13/05/21
7.7	Trabajo Futuro	1 día	mié 21/04/21	mié 21/04/21
7.8	Realización del Artículo del Proyecto	0 días	jue 13/05/21	jue 13/05/21

3.3.2 Informe Final de Riesgos

Durante de la realización del proyecto se han podido mitigar los riesgos que han ido apareciendo sin que consigan tener un gran impacto en el proyecto. Los riesgos con mayor impacto como los retrasos en el desarrollo de los módulos no han llegado a producirse mientras que los riesgos con un impacto menor que han surgido se han podido solventar sin problemas.

3.3.3 Presupuesto Final

En este apartado se detallarán todos los costes relativos al proyecto y se especificarán los presupuestos finales de costes y de cliente en función de los cambios especificados en la planificación final

Presupuesto de Costes

En la siguiente tabla (Ver 3.14) se detalla el presupuesto de costes del proyecto.

TABLA 3.14: Presupuesto Final de Costes del Proyecto

I 1	I 2	I 3	I 4	Descripción	Cantidad	Unidades	Precio	Precio Total	Subtotal (4)	Subtotal (3)	Subtotal (2)	Total
												82.748,80 €
01				**Documentación e Investigación**							**19.790,40 €**	
	01			Exploración de las herramientas de hardening actuales						6.384,00 €		
		01		Ingeniero del Software	60	horas	49,40 €		2.964,00 €			
		02		Director del Proyecto	60	horas	57,00 €		3.420,00 €			
	02			Exploración de las normas y estándares de seguridad de SSOO						5.107,20 €		
		01		Ingeniero del Software	48	horas	49,40 €		2.371,20 €			
		02		Director del Proyecto	48	horas	57,00 €		2.736,00 €			
	03			Recopilación de reviews y papers en Automate-Hardening						1.976,00 €		
		01		Ingeniero del Software	40	horas	49,40 €		1.976,00 €			
	04			Exploración principales vulnerabilidades en Sistemas Operativos						3.952,00 €		
		01		Ingeniero del Software	80	horas	49,40 €		3.952,00 €			
	05			Recopilación de reviews y papers en DSLs						2.371,20 €		
		01		Ingeniero del Software	48	horas	49,40 €		2.371,20 €			
02				**Diseño y definición del prototipo**							**2.625,80 €**	
	01			Descripción inicial del prototipo						155,80 €		
		01		Descripción general del prototipo					155,80 €			
			01	Ingeniero del Software	1	horas	49,40 €	49,40 €				
			02	Jefe de Proyecto	1	horas	57,00 €	57,00 €				
			03	Consultora	1	horas	49,40 €	49,40 €				
	02			Definición del alcance de los módulos						1.246,40 €		
		01		Definición del alcance de Egida Core					311,60 €			
			01	Ingeniero del Software	2	horas	49,40 €	98,80 €				
			02	Jefe de Proyecto	2	horas	57,00 €	114,00 €				

		03	Consultora	2	horas	49,40 €	98,80 €		
	02		Definición del alcance de Egida Setup					311,60 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €		
		03	Consultora	2	horas	49,40 €	98,80 €		
	03		Definición del alcance de Egida API					311,60 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €		
		03	Consultora	2	horas	49,40 €	98,80 €		
	04		Definición del alcance de Egida Role					311,60 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €		
		03	Consultora	2	horas	49,40 €	98,80 €		
03			Definición del alcance de las pruebas						779,00 €
	01		Definición del alcance de las pruebas unitarias					155,80 €	
		01	Ingeniero del Software	1	horas	49,40 €	49,40 €		
		02	Jefe de Proyecto	1	horas	57,00 €	57,00 €		
		03	Consultora	1	horas	49,40 €	49,40 €		
	02		Definición del alcance de las pruebas de aceptación					311,60 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €		
		03	Consultora	2	horas	49,40 €	98,80 €		
	03		Definición del alcance de las pruebas de rendimiento					311,60 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
		02	Jefe de Proyecto	2	horas	57,00 €	114,00 €		
		03	Consultora	2	horas	49,40 €	98,80 €		
04			Diseño del prototipo						444,60 €
	01		Arquitectura del prototipo					148,20 €	
		01	Ingeniero del Software	3	horas	49,40 €	148,20 €		

	02		Diagrama de componentes					98,80 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
	03		Diagrama de despliegue					197,60 €	
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
03			**Desarrollo del prototipo**						**8.249,80 €**
01			Desarrollo del módulo Egida Core					5.532,80 €	
	01		Desarrollo de la interfaz CLI de Egida Menu					395,20 €	
		01	Ingeniero del Software	8	horas	49,40 €	395,20 €		
	02		Desarrollo de la interfaz CLI de Egida Info					197,60 €	
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	03		Desarrollo de la interfaz CLI de Egida Config					197,60 €	
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	04		Desarrollo de la interfaz CLI de Egida Compile					1.185,60 €	
		01	Ingeniero del Software	24	horas	49,40 €	1.185,60 €		
	05		Desarrollo del servicio Playbook					395,20 €	
		01	Ingeniero del Software	8	horas	49,40 €	395,20 €		
	06		Desarrollo del DSL Aspida					3.161,60 €	
		01	Ingeniero del Software	64	horas	49,40 €	3.161,60 €		
02			Desarrollo del módulo Egida API					543,40 €	
	01		Desarrollo del servicio packages					148,20 €	
		01	Ingeniero del Software	3	horas	49,40 €	148,20 €		
	02		Desarrollo del servicio hardening					247,00 €	
		01	Ingeniero del Software	5	horas	49,40 €	247,00 €		
	03		Desarrollo del servicio services					148,20 €	
		01	Ingeniero del Software	3	horas	49,40 €	148,20 €		
03			Desarrollo del módulo Egida Setup					197,60 €	
	01		Desarrollo del rol Ansible					98,80 €	
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €		

	02		Integración con Egida Core					98,80 €		
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €			
04			Desarrollo del módulo Egida Role						592,80 €	
	01		Desarrollo del rol Ansible					494,00 €		
		01	Ingeniero del Software	10	horas	49,40 €	494,00 €			
	02		Integración con Egida Core					98,80 €		
		01	Ingeniero del Software	2	horas	49,40 €	98,80 €			
05			Desarrollo de las pruebas						1.383,20 €	
	01		Desarrollo de las pruebas unitarias					988,00 €		
		01	Ingeniero del Software	20	horas	49,40 €	988,00 €			
	02		Desarrollo de las pruebas de aceptación					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
	03		Desarrollo de las pruebas de rendimiento					197,60 €		
		01	Ingeniero del Software	4	horas	49,40 €	197,60 €			
04			**Desarrollo de los contenidos del Proyecto**							**5.187,00 €**
01			Resumen detallado del proyecto						197,60 €	
	01		Ingeniero del Software	4	horas	49,40 €		197,60 €		
02			Objetivos principales del proyecto						197,60 €	
	01		Ingeniero del Software	4	horas	49,40 €		197,60 €		
03			Trabajo relacionado del proyecto						2.964,00 €	
	01		Ingeniero del Software	60	horas	49,40 €		2.964,00 €		
04			Descripción detallada del sistema						395,20 €	
	01		Ingeniero del Software	8	horas	49,40 €		395,20 €		
05			Descripción de la metodología del proyecto						988,00 €	
	01		Ingeniero del Software	20	horas	49,40 €		988,00 €		
06			Resultados y discusiones del proyecto						345,80 €	
	01		Ingeniero del Software	7	horas	49,40 €		345,80 €		

07		Conclusiones y trabajo futuro del proyecto					98,80 €	
	01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
05		**Desarrollo de la Memoria del Proyecto**						**7.634,20 €**
01		Introducción					311,60 €	
	01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
	02	Director del Proyecto	2	horas	57,00 €	114,00 €		
	03	Consultora	2	horas	49,40 €	98,80 €		
02		Fijación de objetivos					311,60 €	
	01	Ingeniero del Software	2	horas	49,40 €	98,80 €		
	02	Director del Proyecto	2	horas	57,00 €	114,00 €		
	03	Consultora	2	horas	49,40 €	98,80 €		
03		Planificación y gestión del proyecto					623,20 €	
	01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	02	Director del Proyecto	4	horas	57,00 €	228,00 €		
	03	Consultora	4	horas	49,40 €	197,60 €		
04		Estado actual de los conocimientos Cientifico-Técnicos					1.246,40 €	
	01	Ingeniero del Software	8	horas	49,40 €	395,20 €		
	02	Director del Proyecto	8	horas	57,00 €	456,00 €		
	03	Consultora	8	horas	49,40 €	395,20 €		
05		Descripción del sistema					2.492,80 €	
	01	Ingeniero del Software	16	horas	49,40 €	790,40 €		
	02	Director del Proyecto	16	horas	57,00 €	912,00 €		
	03	Consultora	16	horas	49,40 €	790,40 €		
06		Metodología de trabajo					1.246,40 €	
	01	Ingeniero del Software	8	horas	49,40 €	395,20 €		
	02	Director del Proyecto	8	horas	57,00 €	456,00 €		
	03	Consultora	8	horas	49,40 €	395,20 €		
07		Resultados obtenidos					623,20 €	
	01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	02	Director del Proyecto	4	horas	57,00 €	228,00 €		
	03	Consultora	4	horas	49,40 €	197,60 €		
08		Conclusiones y trabajo futuro					779,00 €	

	01	Ingeniero del Software	5	horas	49,40 €	247,00 €		
	02	Director del Proyecto	5	horas	57,00 €	285,00 €		
	03	Consultora	5	horas	49,40 €	247,00 €		
06		**Desarrollo del artículo**						**39.261,60 €**
01		Abstract					4.985,60 €	
	01	Ingeniero del Software	32	horas	49,40 €	1.580,80 €		
	02	Director del Proyecto	32	horas	57,00 €	1.824,00 €		
	03	Consultora	32	horas	49,40 €	1.580,80 €		
02		Resumen					6.232,00 €	
	01	Ingeniero del Software	40	horas	49,40 €	1.976,00 €		
	02	Director del Proyecto	40	horas	57,00 €	2.280,00 €		
	03	Consultora	40	horas	49,40 €	1.976,00 €		
03		State of the Art					12.464,00 €	
	01	Ingeniero del Software	80	horas	49,40 €	3.952,00 €		
	02	Director del Proyecto	80	horas	57,00 €	4.560,00 €		
	03	Consultora	80	horas	49,40 €	3.952,00 €		
04		Contenido					6.232,00 €	
	01	Ingeniero del Software	40	horas	49,40 €	1.976,00 €		
	02	Director del Proyecto	40	horas	57,00 €	2.280,00 €		
	03	Consultora	40	horas	49,40 €	1.976,00 €		
05		Evaluación y discusión					5.608,80 €	
	01	Ingeniero del Software	36	horas	49,40 €	1.778,40 €		
	02	Director del Proyecto	36	horas	57,00 €	2.052,00 €		
	03	Consultora	36	horas	49,40 €	1.778,40 €		
06		Conclusiones					3.116,00 €	
	01	Ingeniero del Software	20	horas	49,40 €	988,00 €		
	02	Director del Proyecto	20	horas	57,00 €	1.140,00 €		
	03	Consultora	20	horas	49,40 €	988,00 €		
07		Trabajo futuro					623,20 €	
	01	Ingeniero del Software	4	horas	49,40 €	197,60 €		
	02	Director del Proyecto	4	horas	57,00 €	228,00 €		
	03	Consultora	4	horas	49,40 €	197,60 €		
TOTAL								**82.748,80 €**

Presupuesto del Cliente

Presupuesto del Cliente Detallado En la 3.15, se puede observar el presupuesto detallado del cliente junto con sus partidas.

TABLA 3.15: Presupuesto Final del Cliente Detallado

Partida	Item	Partida	Importe	Total
1		Investigación y formación en nuevos avances en ciberseguridad		19.790,40 €
	1	Documentación e investigación	15.443,20 €	
	2	Recopilación de reviews y papers	4.347,20 €	
2		Diseño del prototipo y de las pruebas		2.625,80 €
	1	Diseño del prototipo	1.846,80 €	
	2	Diseño de las pruebas	779,00 €	
3		Desarrollo del prototipo		8.249,80 €
	1	Egida Core	5.532,80 €	
	2	Egida API	543,40 €	
	3	Egida Setup	197,60 €	
	4	Egida Role	592,80 €	
	5	Desarrollo de las pruebas	1.383,20 €	
4		Desarrollo de los contenidos del proyecto		5.187,00 €
	1	Resumen detallado del proyecto	197,60 €	
	2	Objetivos principales del proyecto	197,60 €	
	3	Trabajo relacionado del proyecto	2.964,00 €	
	4	Descripción detallada del sistema	395,20 €	
	5	Descripción de la metodología del proyecto	988,00 €	
	6	Resultados y discusiones del proyecto	345,80 €	
	7	Conclusiones y trabajo futuro del proyecto	98,80 €	
5		Desarrollo de la memoria del proyecto		7.634,20 €
	1	Introducción	311,60 €	
	2	Fijación de objetivos	311,60 €	
	3	Planificación y gestión del proyecto	623,20 €	
	4	Estado actual de los conocimientos Cientifico-Técnicos	1.246,40 €	
	5	Descripción del sistema	2.492,80 €	
	6	Metodología de trabajo	1.246,40 €	
	7	Resultados obtenidos	623,20 €	
	8	Conclusiones y trabajo futuro	779,00 €	
6		Desarrollo del Artículo		39.261,60 €
	1	Abstract	4.985,60 €	
	2	Resumen	6.232,00 €	
	3	State of the Art	12.464,00 €	
	4	Contenido	6.232,00 €	
	5	Evaluación y discusión	5.608,80 €	
	6	Conclusiones	3.116,00 €	
	7	Trabajo futuro	623,20 €	
TOTAL CLIENTE				**82.748,80 €**

Presupuesto del Cliente Resumido En la 3.16, se muestra el presupuesto resumido del cliente junto con un gráfico resumen de los diferentes precios de las partidas principales.

Cod.	Partida	Total
1	Investigación y formación en nuevos avances en ciberseguridad	19.790,40 €
2	Diseño del prototipo y de las pruebas	2.625,80 €
3	Desarrollo del prototipo	8.249,80 €
4	Desarrollo de los contenidos del proyecto	5.187,00 €
5	Desarrollo de la memoria del proyecto	7.634,20 €
6	Desarrollo del artículo	39.261,60 €
TOTAL CLIENTE		**82.748,80 €**

TABLA 3.16: Presupuesto Final Resumido del Cliente

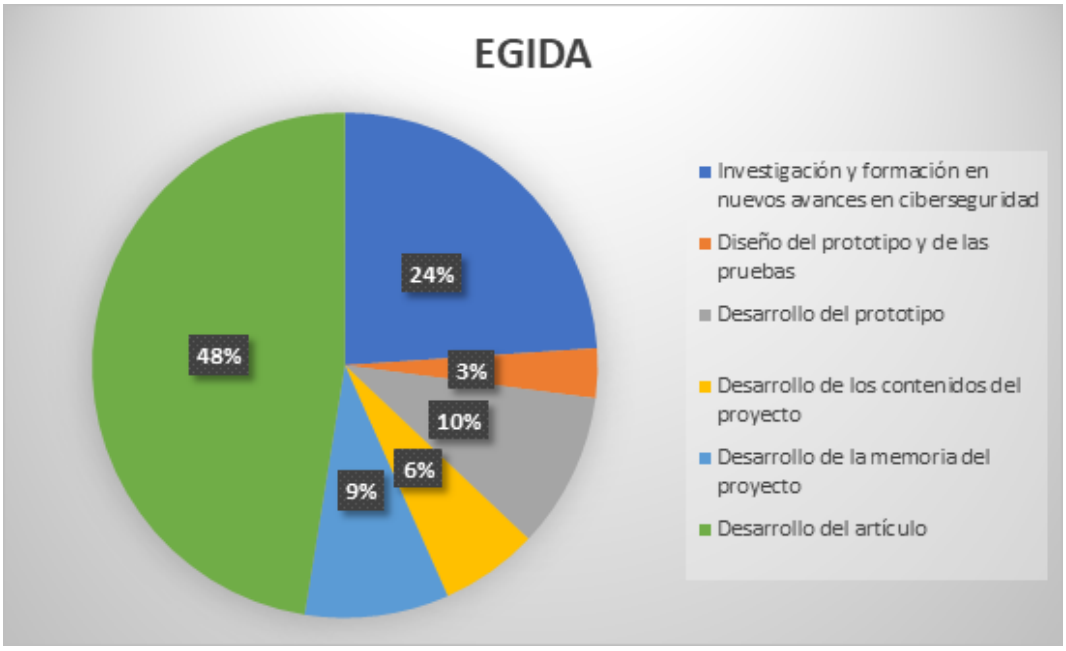

FIGURA 3.3: Presupuesto Resumido del Cliente

3.3.4 Informe de Lecciones Aprendidas

Gran parte de las lecciones aprendidas durante el desarrollo del proyecto se relacionan con el tiempo planeado para la investigación y documentación del estado de los conocimientos científico-técnicos, ya que existe mucha información al respecto y se ha dedicado más tiempo del planeado para dicho tipo de tareas.

Otro gran apunte es ir realizando de manera simultánea memoria y artículo del proyecto, sección por sección y comenzando por el artículo. De esta forma en el artículo se define de manera definitiva la estructura y contenido principal del apartado mientras que en la memoria se escribe una versión ampliada del contenido del artículo. Por último, se redactarían las secciones únicas de la memoria.

CAPÍTULO

4

ESTADO ACTUAL DE LOS CONOCIMIENTOS CIENTÍFICO-TÉCNICOS

4.1 Defense in Depth

La filosofía de **Defensa en Profundidad** (Defense in Depth) es un concepto heredado de los sistemas de la defensa militar que se aplica a todos los equipos que componen una infraestructura [9]. Una estrategia de Defensa en Profundidad bien definida e implementada puede llegar a ser capaz de prevenir y evitar una gran variedad de vectores de ataque, así como proporciona herramientas de monitorización y alarmas para accesos no autorizados al sistema [10].

En este tipo de estrategias se necesita establecer una serie de mecanismos defensivos en capas que se encarguen de proteger los datos e información importantes [11]. Si uno de estos mecanismos falla, otro puede detener el ataque [12]. Este enfoque múltiples capas tiene seguridad intencionalmente redundante y medidas para aumentar la seguridad de un sistema en su conjunto y abordar muchos vectores de ataque diferentes.

Mecanismos defensivos típicamente implementados son el análisis del tráfico de la red, análisis de comportamiento, software antivirus, software que se encarga de analizar la integridad de los datos, restricciones integradas en el código del software, restricciones de usuario y el hardening (blindaje) del sistema y de la infraestructura.

La 4.1 representa una infraestructura típica donde se implementa una filosofía de Defensa en Profundidad para lograr los máximos niveles de seguridad. Un cortafuegos perimetral es el único punto accesible desde Internet a la infraestructura de la empresa. El tráfico procedente de Internet se filtra previamente mediante servicios **CDN** [13] como **CloudFlare** que, dependiendo de los servicios contratados, puede incluir filtrado de tráfico y varias funcionalidades para protección **DDoS**. Una vez que el tráfico llega al cortafuegos externo de la empresa, si se permite, llega a un reverse proxy [14] que redirige la solicitud a la máquina de servicio adecuada. Los reverse proxy son la única máquina a la que el cliente puede enviar sus peticiones y su seguridad puede mejorarse desde otras máquinas del sistema, ocultando la arquitectura interna y los detalles de los servicios de la empresa al exterior. Esto también muestra otro principio: la especialización de la máquina. El reverse proxy puede concentrar una gran cantidad de funciones de seguridad y software mientras que los otros servidores pueden tener solo una capa básica de características de seguridad para concentrarse en brindar sus servicios internos más eficientemente. Es posible que se apliquen servicios similares a **Honeypot** [15] en esta capa externa para evitar ataques automatizados contra servicios peligrosos conocidos, bloqueando las direcciones IP de los clientes infractores.

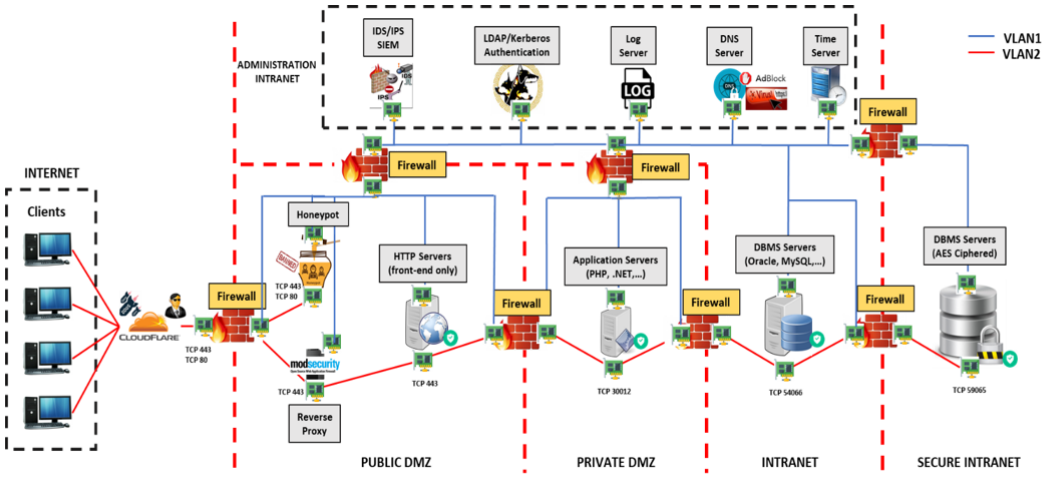

FIGURA 4.1: Filosofía de Defensa en Profundidad en una infraestructura segura de red

Una vez que la solicitud llega a una máquina de servicio en nuestra infraestructura, vemos que tenemos cuatro capas diferentes de redes están aisladas entre sí mediante un cortafuegos perimetral. Las aplicaciones web front-end, back-end y las bases de datos residen en redes independientes que no pueden verse entre sí. El acceso de una capa a otra es controlado a través del firewall, configurado para permitir solo solicitudes de máquinas y puertos específicos en un lado de la red a otras máquinas específicas y puertos al otro lado. Los números de puertos utilizados son bastante altos y no estándares, logrando así prevenir ataques automatizados. De esta forma, si un actor malintencionado se afianza en una capa de red, el resto de las capas están muy aisladas, por lo que se podría contener el ataque. Con respecto a los datos respaldados por Sistemas de Gestión de Bases de Datos, se implementan también otras dos capas de red donde los datos altamente críticos (tarjetas de pago, contraseñas, ...) residen en la capa más profunda, solo accesible desde un puñado de máquinas y usuarios autorizados.

La capa de Intranet es aquella a la que los empleados pueden acceder para realizar su trabajo diario. Es importante destacar que, si bien esta capa tiene acceso a los datos de la empresa, ya que es necesaria para brindar los servicios de la empresa, los datos críticos y los servicios web no están dentro de la misma red, lo que evita "trabajos internos.[en] contra de la empresa, la información y los servicios más valiosos. De nuevo, solo las máquinas autorizadas pueden realizar solicitudes a otros segmentos de red.

Finalmente, existe una intranet de administración paralela para centralizar los servicios de vigilancia IDS / IPS junto con un sistema de gestión de eventos e información de seguridad (SIEM), autenticación (utilizando el protocolo de alta seguridad **Kerberos** [16]), gestión de registros de toda la infraestructura, servidores DNS y un servidor de hora centralizado para sincronizar la hora de los sistemas y registrar eventos. Hay que destacar que esta intranet de administración se separa de la otra mediante el uso de una VLAN diferente, por lo que el tráfico de ambas fuentes no se puede mezclar.

4.2 Protocolos y Estándares de Seguridad

El hardening de un sistema informático no puede ser realmente eficaz sin pautas adecuadas y verificadas. Una estrategia adecuada de hardening debe contemplar los siguientes aspectos:

- Una lista completa de controles de seguridad que se deben aplicar para lograr el nivel de seguridad deseado para un sistema operativo específico o producto
- Una justificación de por qué se necesitan estos controles, en caso de que interfieran con el sistema esperado, actividades o funcionalidades
- Cómo comprobar si estos controles ya están aplicados

Desafortunadamente, muchas veces el Sistema Operativo y software instalados en las máquinas no han pasado por un proceso de hardening y se dejan únicamente con una configuración inicial por defecto tras la instalación. Además, cuando realmente se toman las medidas para aumentar la seguridad del sistema, estas podrían resultar inadecuadas por las siguientes razones:

- **El hardening se puede hacer manualmente, confiando solo en el conocimiento o la habilidad pura del administrador del sistema**. Este es un enfoque muy peligroso, ya que la experiencia del administrador puede ser incompleta en ciertas áreas, las acciones podrían no estar debidamente documentadas, o la implementación de los controles no se puede probar adecuadamente, debido a la falta de un procedimiento formal para realizar estas actividades. Los sistemas configurados de esta manera pueden terminar siendo muy seguros, pero normalmente la probabilidad de encontrar errores por no utilizar un procedimiento válido y confiable es muy alta.
- **El hardening se puede realizar siguiendo una lista incompleta o inadecuada de controles de seguridad**. Si el administrador del sistema decide seguir un procedimiento documentado para realizar operaciones de hardening en lugar de depender únicamente de su propia habilidad, pueden surgir más problemas potenciales, ya que los controles de seguridad pueden haberse obtenido de fuentes no confiables:
  - La lista de control puede estar incompleta. Puede que le falten los pasos necesarios para realizar hardening a partes críticas software.
  - Es posible que la lista de control no esté debidamente documentada. La implementación de algunos controles puede no estar definido, insuficientemente detallado, probado, o simplemente incompleto.
  - Es posible que la lista de control esté debidamente mantenida. Incluso si la lista de control está completa y su implementación debidamente detallada, los controles en sí pueden estar desactualizados, por lo que no son aplicables a versiones de software modernas que siguen siendo mantenidas por su propietario.

Para poder mantener seguros correctamente los sistemas, se necesita de una correcta implementación de estándares internacionales de gestión de la seguridad de la información ISO 27001, ISMS [17]. Desgraciadamente, la proliferación de ataques actual muestra que en muchas ocasiones estas implementaciones no se hacen de manera completa o adecuada.

La implementación de un ISMS puede facilitarse gracias a frameworks que permiten automatización, como el protocolo SCAP y el TCG framework.

4.2.1 SCAP

El protocolo SCAP (Security Content Automation Protocol) es un conjunto de estándares de automatización de seguridad que permiten evaluar el cumplimiento de políticas de seguridad y detectar la presencia de versiones vulnerables de software y de la configuración de los sistemas [18]. El protocolo SCAP [2] proporciona los formatos y nomenclaturas estándar para poder definir e intercambiar información entre usuarios finales y herramientas y cuenta además con un gran número de componentes que se pueden clasificar en las siguientes categorías en función de su finalidad:

- Enumeraciones
  - CVE: Common Vulnerability and Exposures
  - CCE: Common Configuration Enumeration
  - CPE: Common Platform Enumeration
- Métricas
  - CVSS: Common Vulnerability Score System
  - CCSS: Common Configuration Score System
- Lenguajes
  - XCCDF: Extensible Configuration Checklist Description Format
  - OVAL: Open Vulnerability and Assessment Language
  - OCIL: Open Checklist Interactive Language
- Formato de informes
  - ARF: Asset Reporting format
  - AI: Asset Identification
- Integridad
  - TMSAD: Trust Model for Security Automation Data
  - SWID: Software Identification

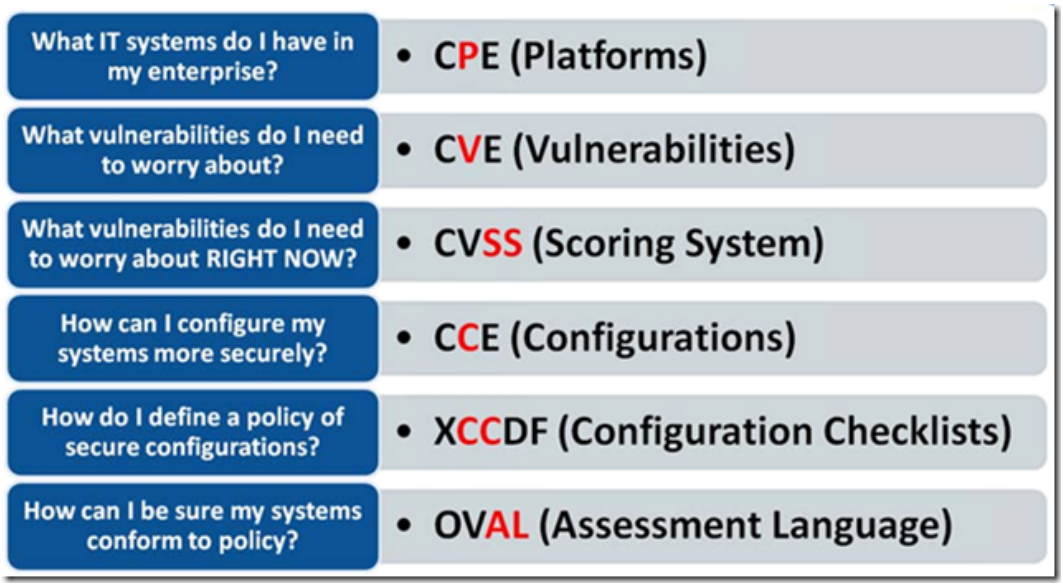

FIGURA 4.2: SCAP

CVE (Common Vulnerability and Exposures) [1] es un sistema de nominación y documentación de vulnerabilidades conocidas referenciadas con un identificador único. Cada vulnerabilidad contiene una descripción, qué versiones del software están afectadas, la posible solución al fallo (si existe) o cómo configurarlo para mitigar dicha vulnerabilidad, así como referencias a publicaciones o entradas de foros o blog donde se ha hecho pública la vulnerabilidad o se demuestra su explotación. Además, cada vulnerabilidad contiene una referencia pública en la web.

Otros sistemas muy similares a CVE son **CCE** (Common Configuration Enumeration) [2] con la diferencia de que su objetivo es la documentación de las configuraciones de los sistemas y **CPE** (Common Platflorm Enumeration) [3] que recoge e identifica sistemas tecnológicos, software y paquetes.

CVSS (Common Vulneravility Score System) [4] es un sistema de puntuación que permite estimar el impacto de las vulnerabilidades en función de sus características con el fin de definir su impacto. Este sistema de puntuación está dividido en tres grupos en función de la métrica: la puntuación base, temporal y ambiental. De esta forma se tienen en cuenta aspectos como posibles agravantes o atenuantes, así como elementos temporales como la disponibilidad de parches.

El sistema de puntuación **CCSS** (Common Configuration Score System) [19] es, al igual que el CCE es un derivado del CVE, un derivado del CVSS que permite medir el impacto de fallos de seguridad dependientes de determinadas configuraciones de los productos.

El protocolo SCAP cuenta con lenguajes como **XCCDF** (Extensible Configuration Checklist Description Format) [5] basado en XML desarrollado por NIST, la NSA, The MITRE Corporation y el Departamento de Seguridad Nacional de Estados Unidos que especifica listas de control de seguridad, puntos de referencia y documentación de configuración. El objetivo del lenguaje XCCDF es sustituir a la documentación de análisis y hardening de seguridad escrita manualmente.

Otro lenguaje del protocolo SCAP basado en XML es **OVAL** (Open Vulnerability and Assessment Language) [6] el cual permite estandarizar el reporte y evaluación del estado de la configuración de los sistemas. OVAL sigue tres pasos durante el proceso de evaluación de los sistemas: (i) Representar la información de la configuración requerida, (ii) Analizar y expresar los estados específicos del sistema y (iii) Reportar los resultados de la evaluación.

OCIL (Open Checklist Interactive Language) [7] especifica las pautas necesarias para poder definir una serie de checklists menos automatizables y que requieren una mayor intervención humana. Para ello OCIL cuenta con soporte para la definición formar de preguntas a los usuarios y para la definición de acciones necesarias en función de las respuestas del usuario.

---

[1] https://cve.mitre.org/about last accessed on May 09, 2021

[2] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/Common-Configuration-Enumeration-(CCE) last accessed on May 09, 2021

[3] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/cpe last accessed on May 09, 2021

[4] https://nvd.nist.gov/vuln-metrics/cvss last accessed on May 09, 2021

[5] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/xccd last accessed on May 09, 2021

[6] https://oval.mitre.org last accessed on May 09, 2021

[7] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/ocil last accessed on May 09, 2021

Lenguajes como OVAL, OCIL o XCCDF favorecen y agilizan el intercambio de información entre profesionales de la ciberseguridad, proveedores de software y auditores, así como permiten el desarrollo de herramientas y soluciones automáticas. Hay archivos escritos en estos lenguajes capaces de verificar o reforzar automáticamente una cantidad sustancial de sistemas operativos y o productos software. **Un ejemplo, es la Security Technical Implementation Guides (STIGs) of the Department of Defense (DoD) of United States** [20].

Dentro de la creación de formatos orientados a la creación de los reportes y la documentación, la identificación de activos o **AI** (Asset Identification) [8] proporciona las construcciones necesarias para identificar de forma exclusiva los assets basándose en identificadores conocidos y/o en información conocida sobre los activos. El **ARF** (Asset Reporting Format) [9] es un modelo de datos que permite expresar el formato de transporte de la información sobre dichos assets y su relación con los informes. El modelo de datos estandarizado facilita la presentación de informes, la correlación y la fusión de la información sobre activos en todas las organizaciones y entre ellas.

El **SWID** (Software Identification) es un sistema de etiquetas definido por la norma ISO / IEC 19779-2 [1] con el objetivo de poder identificar los productos software existentes en el mercado. Este sistema define un formato de metadatos estructurado que permite describir un producto software. La información de una etiqueta SWID proporciona a las herramientas de gestión de assets de software y de seguridad la valiosa información necesaria para automatizar la gestión de una instalación de software a lo largo del ciclo de vida de su despliegue.

Por otro lado, el **TMSAD** (Trusted Model for Security Automation Data) [10] describe un modelo de confianza común que se compone de recomendaciones sobre cómo utilizar las especificaciones existentes para representar firmas, hashes, información de claves e información de identidad en el contexto de un documento XML dentro del ámbito de la automatización de la seguridad.

Sin embargo, herramientas como SCAP, OVAL o XCCDF comparten una serie de deficiencias que impiden que se puedan aplicar con éxito (o solo se apliquen parcialmente en múltiples escenarios):

- **No son totalmente compatibles con varios sistemas operativos**: las herramientas OpenSCAP no funcionan en sistemas Windows (solo permiten el escaneo remoto de las máquinas compatibles) y tienen errores conocidos en las de Linux basadas en Debian.
- **Los archivos disponibles públicamente no cubren algunos productos muy populares utilizados por empresas privadas** o realizan hardening a las máquinas considerando perfiles de uso que pueden no adaptarse a los escenarios típicos de los consumidores.
- **Varios perfiles simplemente implementan la verificación automatizada del control de seguridad**. La remediación automatizada no se admite o solo se admite parcialmente.
- **Es posible que las especificaciones de algunos productos no se mantengan correctamente** y los archivos de las nuevas versiones del producto pueden no estar disponibles durante mucho tiempo. Esto, por ejemplo, sucede con las

[8] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/aid last accessed on May 09, 2021

[9] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/arf last accessed on May 09, 2021

[10] https://csrc.nist.gov/Projects/Security-Content-Automation-Protocol/Specifications/tmsad last accessed on May 09, 2021

versiones más recientes de Ubuntu, que pueden tomar meses para tener un archivo adecuado disponible.

- Por último, y en relación con la segunda deficiencia, **estos archivos de hardening pueden no capturar completamente los requisitos de seguridad de empresas privadas** con servicios expuestos a Internet que utilizan revisiones de software y/o Sistema Operativo con un perfil de usuario diferente.

4.2.2 TGC

El **TCG** (Trusted Computing Group) [21] es un consorcio de más de 200 empresas que desarrollan, definen y promueven un conjunto de especificaciones Open Source de hardware para conseguir proteger a los sistemas de ataques que no pueden ser protegidos por soluciones puramente software obteniendo una plataforma confiable. Para ello, el TCG define tres características que deben cumplir estos sistemas: (i) Medición de su propia integridad, registro e informe, (ii) Capacidades protegidas y (iii) Garantizar la exactitud de la información de estado de un componente.

Sin embargo, un importante obstáculo en el uso de soluciones basadas en el TCG es la ausencia de un repositorio público (publicado por el proveedor de software) de valores hash de todos los componentes de software. Además, el enfoque estático y rígido de la lista blanca no se adapta bien a los entornos de sistemas de sistemas distribuidos a gran escala.

4.2.3 CIS-CSC (Center of Internet Security – Critical Security Control)

CIS-CSC (Center of Internet Security - Critical Security Control) o también llamados CIS Controls [22] son una colección de directrices de buenas prácticas que las organizaciones pueden aplicar para reducir significativamente su superficie de ciberataque. Estas directrices están formuladas por un grupo de expertos en tecnología de la información utilizando la información obtenida de ataques reales y sus defensas efectivas.

FIGURA 4.3: CIS Benchmarks Logo

4.2.4 CIS Controls

Estas guías están compuestas por 20 actividades llamadas **Critical Security Controls (CSC)** o simplemente CIS Controls, y están divididas en tres categorías que permiten clasificar a una organización en **Grupos de Implementación (IG)**. Cada IG define que controles de seguridad debería implementar una organización en función de los recursos que tenga a su disposición, así como su nivel de riesgo. Esto permite que los controles puedan ser utilizados por casi cualquier tipo de organización, independientemente de su tamaño o recursos [23], [24].

- En el **primer grupo de implementación** se recogen controles de seguridad básicos, como los relacionados con el inventario y control de activos software y hardware, uso controlado de los privilegios administrativos o mantenimiento y monitorización.
- El **segundo grupo de implementación** contiene controles de seguridad que deberían de aplicar organizaciones con un mayor riesgo de exposición a ataques ya que estos se enfocan en aspectos como la protección perimetral, defensas contra malware, controles de acceso o configuración segura de firewalls, routers y switches.
- El **tercer grupo de implementación** está compuesto por controles de seguridad basados en la seguridad del software de aplicación, respuesta y gestión de incidentes o pruebas de penetración y ejercicios de equipo rojo.

Los CIS Controls pueden funcionar como recursos independientes o junto con frameworks adicionales y proporcionar un mapeo [4] de sus controles de seguridad a otros marcos como NIST o ISO/IEC 27001. Además, la automatización del protocolo SCAP puede realizarse a través de los CIS Benchmarks, pero con un coste.

4.2.5 CIS Benchmarks

Los CIS Benchmarks son las mejores prácticas documentadas del sector para la seguridad de los sistemas informáticos, el software y las redes. Actualmente hay más de 100 benchmarks que cubren un gran número de familias de productos de proveedores y contienen una lista detallada de las tareas de congestión de seguridad aplicables a ese producto. Los CIS Benchmarks proporcionan un mapeo aplicable a los controles del CIS que nos permite aplicar estos controles de seguridad al producto. Además, las tareas se dividen en dos perfiles según su impacto en el rendimiento del producto:

- El perfil de **Nivel 1** se corresponde con tareas que pueden implementarse con cierta rapidez y están diseñadas para no tener un gran impacto en el rendimiento del producto. El objetivo de este perfil es reducir la superficie de ataque reforzando la seguridad sin obstaculizar su funcionalidad.
- El perfil de **Nivel 2** ya se considera Defensa en Profundidad y está destinado a entornos críticos. Estas tareas pueden afectar al correcto funcionamiento si no se aplican adecuadamente.

4.3 Herramientas de Automatización de Configuraciones de Seguridad

La existencia de las guías y estándares anteriormente mencionados como SCAP o TCG, permiten la aparición de herramientas de hardening automáticas que traten de cumplir con la mayoría de sus apartados como **VM2** [25] que genera de forma automática máquinas virtuales, aplicándoles algunos de los CIS Benchmarks o como el script de configuración de un servidor web Linux propuesto por Michal Olencin [26]. Otra herramienta muy popular de automatización es **JShielder** [7], que permite el hardening de LAMP (Linux-Apache-Mysql/MariaDB-PHP) y LEMP (Linux-Nginx-MySQL/MariaDB-PHP).

Proyectos de investigación como ISCP [27] o el propuesto por Karel Durkota et al. [28] utilizan modelos para determinar los controles vulnerables y proporcionar directrices claras sobre cómo realizar el análisis de control y para representar las posibles acciones de un atacante mediante grafos de ataque respectivamente.

Uno de los proyectos más importantes, además de ser la implementación más popular del protocolo SCAP, es el **OpenSCAP Project** [29]. Es una colección de herramientas Open Source para implementar y hacer cumplir el estándar SCAP, que están certificadas hasta la versión de SCAP 1.2. Estas herramientas comprenden un marco de especificaciones multipropósito que admite la configuración automatizada, verificación de vulnerabilidades y parches, actividades de cumplimiento de control técnico y mediciones de seguridad. Además, OpenSCAP tiene soporte para el uso de ficheros escritos en los lenguajes OVAL y XCCDF, así como permite realizar informes de auditoría de un sistema en base a políticas ya definidas como las del CIS o los STIGs.

Existen otras herramientas que utilizan estas guías y estándares para realizar auditorías y tests al sistema como *Lynis*, *Chef Inspec* o *Prowler*. El resultado de estas herramientas es un informe con la checklist de los controles de seguridad que ha pasado o no el sistema.

CAPÍTULO

5

DESCRIPCIÓN DEL SISTEMA

5.1 Propuesta

Este proyecto de investigación propone un sistema de orquestación de servidores llamado Egida que permite realizar y desplegar configuraciones de seguridad (listas de control personalizadas) sobre una infraestructura de máquinas o servidores. Estas configuraciones de seguridad pueden blindar y proteger estos servidores mediante la aplicación de las medidas de seguridad deseadas en función del perfil del servidor.

De esta manera, dependiendo del uso que se le vaya a dar a un servidor, se le puede aplicar una configuración de seguridad personalizada que tenga en cuenta los requerimientos de su correcto funcionamiento (por ejemplo, una máquina destinada a funcionar como un servidor web debería de permitir el acceso HTTP al puerto 80).

5.1.1 Controles de Seguridad

Egida utiliza como punto de partida la especificación más actualizada de las guías de los CIS Benchmark para el hardening diferentes sistemas operativos Linux conocidos. Como la automatización mediante SCAP está cubierta a través del proyecto CIS Benchmark (con un coste), con Egida intentamos mejorar esta automatización introduciendo nuevos elementos dentro de las fases de verificación y remediación del control:

- **Uso de herramientas de despliegue de configuración estándar** para distribuir los procedimientos de verificación e implementación de los controles de seguridad. En lugar de seguir el enfoque SCAP, que actualmente puede ser potencialmente incompatible con diferentes sistemas, utilizamos una herramienta de gestión de configuración centralizada muy popular, utilizada por empresas muy importantes en todo el mundo: **Ansible** [1]. De esta manera, el proceso de hardening de la seguridad puede integrarse con otros procesos de administración existentes, sin necesidad de instalar otras soluciones dentro de un conjunto de software. Ansible está disponible para sistemas Linux, Windows y Mac OS, por lo que Egida puede ser portado más facilmente. Ansible también reporta dos ventajas adicionales muy importantes: la capacidad de trabajar sobre cualquier máquina en la que tengamos acceso SSH y la capacidad de no repetir operaciones que se detecten como ya aplicadas. Esto facilita enormemente las pruebas de control de seguridad y la propagación de la aplicación con un gran grado de eficiencia.

- **Los controles de seguridad no son los únicos elementos de seguridad**. Aunque los controles de seguridad son una base muy sólida de un sistema seguro, también existen múltiples programas de seguridad que pueden complementarlos y mejorar aún más la seguridad de la infraestructura. Software como proxies inversos, cortafuegos perimetrales, cortafuegos basados en host, NIDS, HIDS, WAFs, etc. pueden ser desplegados vía Ansible en ciertas máquinas de la infraestructura, yendo más allá del alcance de los Benchmarks CIS. El proyecto Egida también quiere aplicar su enfoque flexible a estos elementos de seguridad de nivel superior, no sólo desplegándolos sobre máquinas específicas, sino también facilitando la elección de la selección adecuada de los controles de seguridad de los CIS Benchmarks más apropiados para las máquinas con este software. Con ello se pretende conseguir las mismas ventajas que ofrecen proyectos como JShielder [7], pero la mayor flexibilidad permitirá a los usuarios especializar mejor las misiones de las máquinas, en lugar de crear una única máquina que concentre todo el software de protección.

[1] `https://www.ansible.com` last accessed on May 18, 2021

- **Control preciso de los controles de seguridad**: Existen otras implementaciones gratuitas de los controles del CIS Benchmark distribuidas a través de Ansible, pero podrían estar incompletas, no estar bien mantenidas y, lo más importante, seguir un enfoque de todo o nada. Esto significa que todos los controles del benchmark son probados o aplicados sin permitir a los usuarios elegir grupos/secciones de control o selecciones que se adapten a un determinado propósito o especialización del servidor. Con Egida queremos conseguir una gestión mucho más precisa y flexible de los controles de seguridad que pueden desplegarse, creando perfiles de control que pueden superponerse o combinarse a voluntad del usuario. Esto asegura desplegar sólo lo necesario (favoreciendo la especialización de las máquinas), y utilizar combinaciones de grupos de control para crear perfiles de máquinas, adecuando mejor los controles desplegados a determinados usos típicos. La agregación de estos perfiles de control de seguridad para adaptarse a perfiles concretos de máquinas es también un conocimiento que nuestro proyecto de investigación genera y que otros usuarios pueden aprovechar o evaluar.

Este tipo de información puede ser utilizada para cambiar el método de hardening utilizado en tiempo de ejecución mediante el uso de un script o un lenguaje de dominio específico, permitiéndonos automatizar aún más el proceso.

5.1.2 Lenguaje de Dominio Específico (DSL)

Como hemos dicho anteriormente, las soluciones basadas en el marcado XML como OVAL o XCCDF proporcionan especificaciones para detallar y desplegar con precisión los controles de seguridad, pero no permiten incorporar lo que podemos llamar comprobaciones semánticas. Los errores de despliegue de Ansible también se descubren en tiempo de ejecución. El objetivo final del proyecto de investigación Egida es desarrollar un DSL que permita a los usuarios desplegar grupos de controles de seguridad a partir de los CIS Benchmarks, pero realizando validaciones tempranas de las máquinas y/o contenidos de la infraestructura antes del despliegue. Aspectos como inconsistencias con las máquinas que se esperan encontrar, puertos abiertos, software disponible, directorios necesarios, usuarios requeridos, etc. podrían impedir que un despliegue sea exitoso. Egida pretende explorar los límites de lo que se puede validar antes de realizar un despliegue, para que la probabilidad de encontrar un fallo en tiempo de ejecución sea significativamente menor. Un control de errores preciso y temprano también puede permitir a los administradores resolver los problemas en un tiempo sustancialmente menor, ya que no es necesario ejecutar el procedimiento de despliegue posteriormente hasta que no se encuentre ningún error. Esta filosofía ya se probó con éxito con los escaneos de Nmap en [8].

El objetivo es definir programas DSL que contengan la configuración de hardening de las máquinas que cumplen un rol específico de una empresa. De esta forma, podemos tener un programa que contenga toda la configuración de hardening (controles de seguridad más sus valores variables asociados) de sus servidores HTTP, servidores SGBD o proxys. Estos archivos podrían ser validados por expertos adicionalmente a la validación que realiza el DSL, y una vez determinados como válidos, ser distribuidos y reutilizados en toda la infraestructura de la empresa. Si la configuración cambia o debe ser refinada, también queremos comprobar si el DSL es capaz de detectar las inconsistencias producidas en las nuevas versiones antes de su despliegue.

5.2 Estructura

Egida funciona con un patrón de Master-Worker (Ver 5.1) donde el administrador de sistemas tiene el control de la máquina Master con Egida instalado y desde donde se realizará la gestión de las configuraciones de seguridad de las máquinas Worker.

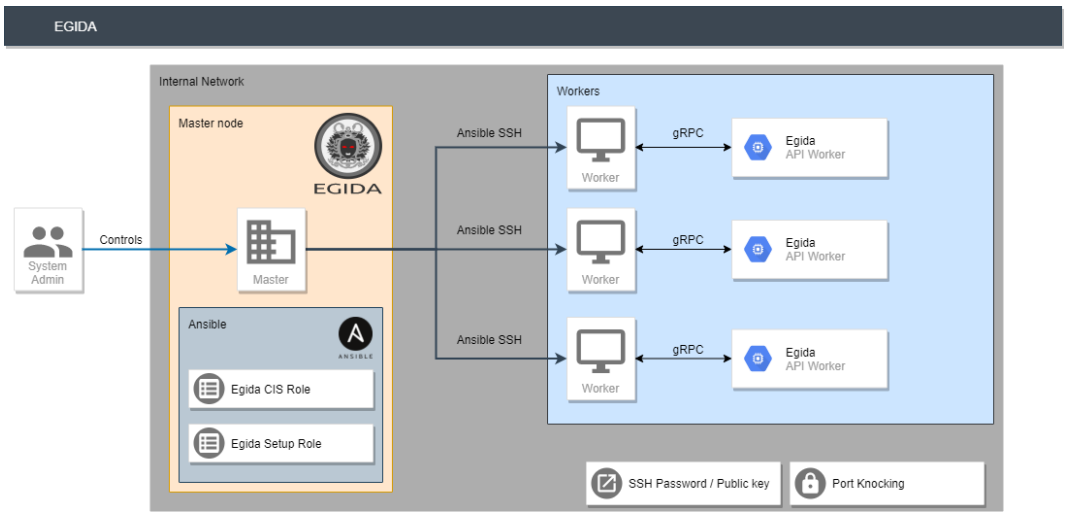

FIGURA 5.1: Diagrama del funcionamiento de Egida

Egida utiliza Ansible Roles para la configuración inicial de las máquinas, instalando el servicio Egida API Worker en un puerto de la máquina configurado con Port Knocking (por defecto el puerto se mantiene cerrado, hasta que el Master node realice knock de una manera determinada). Este servicio permite obtener información de la máquina destino mediante la cual poder personalizar la configuración de seguridad. La comunicación entre Egida y Egida API Worker se realiza mediante el protocolo gRPC.

Egida tiene tanto un manual de uso como de instalación que se puede visitar en en su página web (`https://antonioalfa22.github.io/egida`).

5.2.1 Egida Core

Egida Core o simplemente Egida es el programa principal del proyecto y es la interfaz que tienen los usuarios administradores de sistemas para interactuar con las máquinas que quieren configurar.

Egida permite dos modos de uso gracias a un menú interactivo y un Lenguaje de Dominio Específico (DSL) llamado Aspida. Egida está desarrollado completamente en Golang y utiliza la herramienta Antlr4 [2] para el desarrollo de su DSL. El código fuente está disponible en GitHub (`https://github.com/Egida-Kassandra/egida`).

[2] `https://www.antlr.org/` last accessed on 28, May

Egida Menú

FIGURA 5.2: Egida Menú

Se trata de un menú de consola interactivo que permite seleccionar las opciones de hardening que se quieren realizar en una máquina o conjunto de máquinas.

Actualmente únicamente se encuentra desarrollada la opción de hardening basada en los CIS Benchmarks, pero en un futuro se realizarán las opciones de configuración basadas en servidores LAMP y LEMP.

Dentro de la opción de hardening basada en los CIS Benchmarks, Egida Menú permite la personalización de la configuración de seguridad de la máquina seleccionando los controles, secciones o puntos concretos de los CIS Benchmarks que se quieren aplicar.

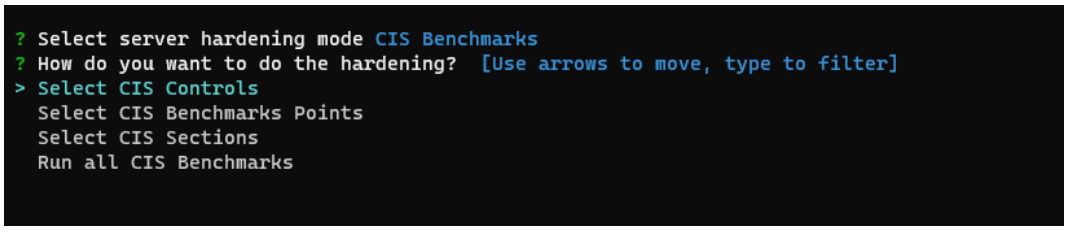

FIGURA 5.3: Egida Menú Selección de Opciones

CIS Controls Egida permite seleccionar de entre los siguientes controles de los CIS Benchmarks:

- 2. Inventory and Control of Software Assets
- 3. Continuous Vulnerability Management
- 4. Controlled Use of Administrative Privileges

- 5. Secure Configuration for Hardware and Software on Mobile Devices, Laptops, Workstations and Servers
- 6. Maintenance, Monitoring and Analysis of Audit Logs
- 8. Malware Defenses
- 9. Limitation and Control of Network Ports, Protocols and Services

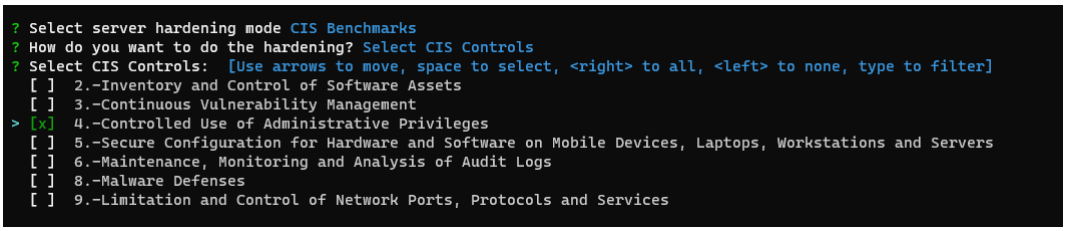

FIGURA 5.4: Egida Menú CIS Controls

CIS Sections Actualmente Egida permite seleccionar de entre las siguientes secciones de los CIS Benchmarks:

- 1.1-Filesystem Configuration
- 1.2-Console Software Updates
- 1.3-Configure sudo
- 1.4-Filesystem Integrity
- 1.5-Secure Boot Settings
- 1.6-Additional Process Hardering
- 1.7-Mandatory Access Control
- 1.8-Warning Banners
- 1.9-Updates
- 2.1-Initd Services
- 2.2-Special Purpose Services
- 2.3-Service Clients
- 3.1-Network Parameters Host
- 3.2-Network Parameters Host and Router
- 3.3-TCP Wrappers
- 3.4-Uncommon Network Protocols
- 3.5-Firewall Configuration
- 3.6-Wireless
- 3.7-Disable IPv6
- 4.1-Configure System Accounting (auditd)
- 4.2-Configure logging

- 4.3-Ensure logrotate is configured
- 5.1-Configure cron
- 5.2-SSH Server configuration
- 5.3-Configure PAM
- 5.4-User accounts and environment
- 5.5-Ensure root login is restricted to system console
- 5.6-Ensure access to the su command is restricted
- 6.1-System file permisions
- 6.2-User and Group Settings

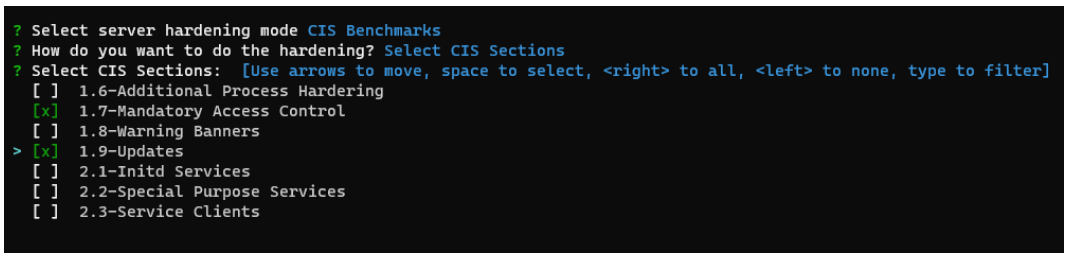

FIGURA 5.5: Egida Menú CIS Sections

CIS Points Egida permite seleccionar manualmente que puntos de los CIS Benchmarks se desea ejecutar.

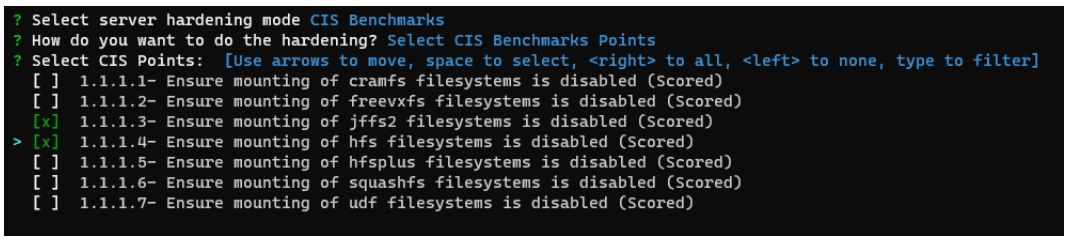

FIGURA 5.6: Egida Menú CIS Points

Egida DSL (Aspida)

Como se ha mencionado anteriormente, Egida cuenta con un Lenguaje de Dominio Específico (DSL) llamado Aspida que permite el desarrollo de scripts predefinidos de configuración que actúen de manera distinta dependiendo del estado de la máquina destino.

El lenguaje está orientado a facilitar la aplicación de controles de seguridad ligados a los perfiles de las máquinas, permitiendo también a los usuarios comprobar diferentes estados, condiciones o valores de variables de forma que puedan prevenir o reaccionar ante errores de despliegue antes de que se produzca el despliegue de la configuración.

El lenguaje también tiene como objetivo implementar un módulo de prevención de errores semánticos para que un determinado programa no pueda ser desplegado si el procesador del lenguaje detecta algún tipo de problema con la infraestructura

de despliegue, incompatibilidades entre los controles de seguridad, valores de variables erróneos y cualquier otra condición que pueda causar un error en tiempo de ejecución y pueda ser prevenido en tiempo de compilación (Ver 5.1.2).

Aspida además permite el uso de estructuras condicionales para proporcionar al usuario la capacidad de especificar diferentes tipos de acciones en función de la formación obtenida de la máquina de destino. La estructura de la gramática principal se muestra en el código siguiente (Ver 5.1).

```
1  program : main hosts tasks variables?;
2
3  // Blocks
4  main : MAIN_KW ':' '{' main_content '}';
5  hosts : HOSTS_KW ':' STRING NS;
6  tasks : TASKS_KW ':' '{' tasks_content '}';
7  variables : VARS_KW ':' '{' vars_content '}';
8
9  ...
10
11 // MAIN Block
12 main_content : main_prop (main_prop)*;
13 main_prop: name | connection | description
14
15 ...
16
17 // TASKS Block
18 tasks_content : tasks_prop (tasks_prop)* | ifStat (
       elifStat)* elseStat;
19 tasks_prop : sections | points | controls | exclusions |
       tags;
20
21 ...
22
23 // VARS Block
24 vars_content : vars_prop (vars_prop)*;
25 vars_prop : STRING ':' value | STRING ':' '{' vars_content
       '}';
26
27 ...
```

LISTING 5.1: Aspida Grammar

MAIN El bloque MAIN proporciona información sobre el script, como el nombre, una descripción o el tipo de conexión a la máquina de destino (local o SSH).

HOSTS Nombre del host o grupo de hosts en los que se en el que se va a ejecutar el script.

TASKS Este bloque es el más importante ya que contiene toda la información sobre las tareas que se van a realizar. Permite determinar qué secciones o puntos específicos de los CIS Benchmark se van a ejecutar o se van a excluir, así como para seleccionar los CIS Controls o ejecutar todas las tareas que corresponden a una etiqueta (por ejemplo, todas las tareas relacionadas con SSH). En este bloque, se pueden utilizar sentencias condicionales If-ElsIf-Else junto con la obtención de valores del estado de la máquina objetivo para controlar el flujo de ejecución del script.

VARS El bloque de variables permite dar un valor a cada una de las variables que se utilizarán durante la ejecución del script.

Algunos ejemplos de scripts escritos en Aspida se pueden observar en 5.7 y 5.8 .

```
MAIN : {
    name: "SSH Config";
    connection: LOCAL;
    description: "Aspida Test";
}

HOST : "localhost";

TASKS : {
    IF "machine.open_port" == 22 {
        sections: ["5.1", "5.3", "5.4"];
    }
    ELSE {
        sections: ["5.1", "5.2", "5.3", "5.4"];
    }
}

VARS : {
    "user_ssh": "antonio";
    "password": {
        "max_days": 365;
        "min_days": 7;
        "warn_age": 7;
        "inactive": 30;
    };
    "nameservers": ["8.8.8.8", "8.8.4.4"];
}
```

FIGURA 5.7: Ejemplo de script en Aspida

```
MAIN : {
    name: "Conditionals";
    connection: SSH;
    description: "Aspida Example";
}

HOST : "192.168.56.1";

TASKS : {
    IF "services.ufw" == "stopped"{
        sections: ["1.1", "1.2"];
        exclusions : ["1.1.1.3"];
    }
    ELIF "hardscores.lynis" <= 50 {
        points: ["1.1.1.5"];
    }
    ELIF "services.apache" == "STOPPED" {
        points: ["1.1.1.4"];
    }
    ELSE {
        controls: ["9.4"];
    }
}
```

FIGURA 5.8: Ejemplo de script en Aspida

El lenguaje Aspida tiene como objetivo ir más allá de las capacidades de los lenguajes de marcado basados en XML OVAL y XCCDF creando un DSL que da a los usuarios un mayor control sobre las actividades que realizan, pasando a un nivel de abstracción más alto ya que las operaciones reales están representadas por controles de seguridad de referencia CIS confiables, no siendo necesario definir nuestras propias operaciones de bajo nivel.

Hosts Groups

Egida permite la creación de grupos de hosts o máquinas de forma que se puedan agrupar servidores que compartan funcionalidad y aplicarles de manera automática la misma configuración de seguridad.

Variables

Muchas de las tareas y operaciones requieren datos que, dependiendo del tipo de instalación o configuración que se le quiera aplicar a una máquina, puede ser variable (por ejemplo, nombres de usuarios, contraseñas, etc.).

Para solucionar esto, Egida permite el uso de variables que puede modificar el usuario administrador del sistema tanto en el modo de ejecución del menú interactivo (a través de un fichero de configuración YAML) como en el Lenguaje de Dominio Específico (dentro del bloque de variables).

5.2.2 Egida Role CIS

El módulo Egida Role CIS es un Ansible Role que se instala junto con Egida en el Master node y que define todas las operaciones de configuración de seguridad basadas en los CIS Benchmarks que puede realizar Egida.

Este rol de Ansible contiene una tarea por cada punto correspondiente a los CIS Benchmarks. Cada una de estas tareas contiene etiquetas que indican el control, punto y sección de los CIS Benchmarks a la que pertenece, así como algunas etiquetas extra que proporcionan información extra como softwares o elementos afectados por dicha tarea.

Además de los controles de los CIS Benchmarks, añade algunas tareas adicionales y opcionales que pueden ayudar a blindar y proteger mejor una máquina.

Actualmente se encuentran implementados todos los puntos de los CIS Benchmarks a excepción de los siguientes:

- **1.1 Filesystem Configuration**
  - 1.1.15
  - 1.1.16
  - 1.1.17
  - 1.1.18
  - 1.1.19
  - 1.1.20
  - 1.1.21
  - 1.1.22
  - 1.1.23
- **4.1 Configure System accounting**
  - 4.1.1.4
  - 4.1.2.1
  - 4.1.2.2
  - 4.1.2.3
  - 4.1.11
- **4.2 Configure Logging**
  - 4.2.1.2
  - 4.2.1.3
  - 4.2.1.4
  - 4.2.1.5
  - 4.2.1.6
  - 4.2.2.1
  - 4.2.2.2
  - 4.2.2.3
  - 4.2.3
  - 4.3
- **5.4 User Accounts and environment**
  - 5.4.1.5
  - 5.4.2
  - 5.5

  - 5.6

- **6.1 System file permissions**
  - 6.1.1
  - 6.1.10
  - 6.1.11
  - 6.1.12
  - 6.1.13
  - 6.1.14

- **6.2 User Accounts and environment**
  - All

El código fuente se encuentra disponible en GiHub y en la lista de roles de Ansible Galaxy:

- https://github.com/Egida-Kassandra/egida-role-cis
- https://galaxy.ansible.com/antonioalfa22/egida_role_cis

5.2.3 Egida Role Setup

Se trata de un Ansible Role que se encarga de instalar el servicio Egida API Worker y las herramientas u opciones necesarias para el correcto funcionamiento de Egida en el sistema.

Además, configura el puerto que utiliza el servicio Egida API Worker para que se mantenga cerrado mediante Port Knocking. Cuando el módulo Egida Core requiera acceder al servicio para obtener información de la máquina, el puerto se abrirá temporalmente mediante un knock de una manera determinada y configurable.
El código fuente se encuentra disponible GitHub y en la lista de roles de Ansible Galaxy:

- https://github.com/Egida-Kassandra/egida-role-setup
- https://galaxy.ansible.com/antonioalfa22/egida_role_setup

5.2.4 Egida API Worker

Se trata de una API gRPC que se instala en una máquina Worker a través del Master Node y que funciona como servicio para proporcionar al módulo Egida Core información sobre la máquina como datos sobre los servicios en ejecución o parados, información de la máquina, paquetes instalados o resultados de las auditorías de seguridad mediante herramientas de evaluación.

Actualmente se está utilizando la herramienta Lynis para proporcionar información sobre el estado de seguridad de una máquina (mediante una puntuación) pero se pretende añadir otras herramientas complementarias.

Esta API gRPC se ejecuta como un servicio y por defecto está escuchando en el puerto 8128 pero este valor es configurable por el usuario.

Esta desarrollado completamente en Golang y comparte un fichero llamado "egida.proto" basado en los Protocol Buffers de Google con el módulo de Egida Core donde están definidas los diferentes tipos de peticiones que se pueden realizar.

El código se encuentra disponible en GitHub (https://github.com/Egida-Kassandra/egida-api-worker)

CAPÍTULO

6

METODOLOGÍA DEL TRABAJO

En esta sección trataremos de definir la metodología y pautas seguidas para poder realizar el posterior análisis y resultados sobre nuestra propuesta Egida.

Todas las mediciones y pruebas se han realizado utilizando un Sistema Operativo Ubuntu 18.04 LTS de 64 bits.

6.1 Casos de Uso

Para evaluar el sistema Egida, hemos definido 3 casos de uso compuestos por varias funcionalidades que debe cumplir el sistema. Cada uno de estos casos de uso trata de recoger los objetivos principales que deseamos cumplir con nuestra propuesta y se realizará la evaluación del sistema en función de estos.

6.1.1 Perfiles automatizados y personalizados de hardening del sistema

El sistema debe ser capaz de ejecutar automáticamente todas las opciones de hardening mencionadas anteriormente. El sistema también debería permitir al usuario personalizar el tipo de hardening que se va a realizar. Para comprobar esto, hemos definido cuatro pruebas diferentes:

1. **Ejecución automática de todos los controles de seguridad disponibles**. Realizaremos una auditoría con las herramientas **Lynis, Chef InSpec y OpenSCAP** y compararemos la puntuación obtenida antes y después de ejecutar todos los controles de seguridad.

2. **Ejecución de todas las tareas que pertenecen a un mismo CIS Control**. Para testear esta prueba ejecutaremos todos los puntos de referencia de un control concreto y comprobaremos si se han ejecutado todos o si falta alguno de ellos.

3. **Uso de etiquetas para ejecutar todos los controles de seguridad relacionados con un tema**. Por ejemplo, ejecutar todas las tareas y controles de seguridad que tengan relación con SSH o con CRON.

4. **Modificar el valor por defecto de las variables**. Por defecto, las tareas de seguridad que requieren de algún valor o variable tienen preasignados un valor que puede modificarse tanto en el script del DSL como en el fichero de variables si se utiliza la opción del menú.

6.1.2 Personalización de las operaciones de seguridad según las características de la máquina de destino

Como hemos comentado anteriormente (ver 5.1.2), el lenguaje de dominio específico debe permitir decidir qué controles de seguridad se aplican en una máquina en función de su estado y características actuales. De esta forma podemos modelar mejor la experiencia de un administrador de sistemas o de seguridad capacitado según estas características. Para evaluar esto, hemos definido dos pruebas diferentes que necesitan obtener información de la máquina objetivo.

1. **Cambiar la configuración en función de la información de la máquina de destino** (servicios, paquetes, puertos abiertos/cerrados, ...). Para comprobarlo, proponemos dos posibles situaciones reales.

   (a) Si el servicio **Apache** está funcionando y el puerto 80 está en uso, no ejecutar la versión 2.2.10. La tarea de los *CIS Benchmarks* 2.2.10 puede afectar al correcto funcionamiento del servicio **Apache**.
   (b) Si se detecta una Interfaz Gráfica de Usuario, no ejecutar la tarea de los *CIS Benchmarks* 2.2.2 ya que puede ocasionar que esta interfaz deje de funcionar o no lo haga correctamente.

(c) Si se detecta que se está utilizando *telnet* y a su vez *SSH* está habilitado, ejecutar la tarea de los *CIS Benchmarks* 2.3.4 que se encarga de deshabilitar *telnet*.

2. **Detectar si el sistema objetivo no obtiene una puntuación específica mediante una auditoría**. El uso de herramientas de auditoría como Lynis pueden determinar si una máquina necesita de un hardening más exhaustivo. Para ello se utilizará el score obtenido y se actuará en función de su resultado.

6.1.3 Detección temprana de errores de configuración

El lenguaje de dominio específico debe detectar posibles errores de configuración en tiempo de compilación y detectar cualquier tipo de problema con la infraestructura de despliegue, valores de variables erróneos y cualquier otra condición que pueda causar un error en tiempo de ejecución. La diferencia con otras herramientas como *Ansible* radica en que, mientras que estas herramientas detectan los errores una vez que ya han ocurrido, con **Egida** podemos obtener esta información previamente, durante la compilación del programa. Para evaluar esta funcionalidad, hemos definido las siguientes pruebas:

1. **Detección de errores léxicos y sintácticos**. El compilador del lenguaje debe de ser capaz de detectar errores tanto léxicos como sintácticos en tiempo de compilación y mostrar al usuario en que línea se ha producido el error y que tipo de valor se esperaba.

2. **Uso de warnings cuando se está utilizando el valor por defecto de una variable**. Debido a que el sistema asigna un valor por defecto a todos los parámetros que se necesitan durante la ejecución de los controles de seguridad, si no se modifican estos valores se utilizarán los de por defecto. En caso de que esto suceda, el compilador deberá de notificar al usuario de que se está utilizando una variable a la que no se ha asignado un nuevo valor y que por lo tanto se utilizará su valor por defecto.

3. **Uso de warnings para notificar a un usuario de que se va a ejecutar una tarea de Nivel 2 Workstation**. Estas tareas de los CIS Benchmarks son peligrosas ya que pueden afectar al correcto funcionamiento del sistema si no se ejecutan con precaución. Por lo tanto, el compilador debería de notificar al usuario que se va a ejecutar una de estas tareas mediante un mensaje de warning.

CAPÍTULO

7

RESULTADOS OBTENIDOS

7.1 Casos de Uso

En esta sección, detallaremos los resultados de las pruebas que hemos realizado sobre los casos de uso definidos para nuestro sistema propuesto y su posterior discusión.

7.1.1 Perfiles automatizados y personalizados de hardening del sistema

Para probar la primera sección de este caso de uso, se ejecutaron todos los controles de seguridad disponibles en Egida (véase 5.2.1). Se compararon los resultados de las herramientas de auditoría antes y después del hardening con su configuración inicial.

El valor de la puntuación obtenida en las distintas herramientas de auditoría puede variar en función de la configuración inicial del sistema. Los servicios instalados, los valores por defecto u otras características pueden afectar a la puntuación. Para todas las pruebas hemos tomado como estado inicial un Sistema Operativo con la configuración de instalación base.

Para la herramienta **Chef InSpec** hemos utilizado el perfil **CIS Distribution Independent Linux Benchmark** [3] que implementa el perfil **CIS Distribution Independent Linux 2.0.0 Benchmark** y para la auditoría **OpenSCAP** hemos utilizado el perfil CIS Benchmarks para Ubuntu 18.04 LTS (xccdf_org.ssgproject.content_profile_cis). En la herramienta **Lynis**, hemos tomado las pruebas ejecutadas y no saltadas. Hemos representado las advertencias y sugerencias como fallos.

Herramienta de Auditoría	Antes / Después	Correctos	Fallos	Otros	Score [%]
Chef Inspec	Antes	85	118	32	36.17
	Después	151	53	31	64.25
OpenSCAP	Antes	31	39	1	32.08
	Después	52	18	1	71.46
Lynis	Antes	78	72	0	52.00
	Después	126	24	0	84.00

TABLA 7.1: Resultados Chef InSpec, OpenSCAP y Lynis

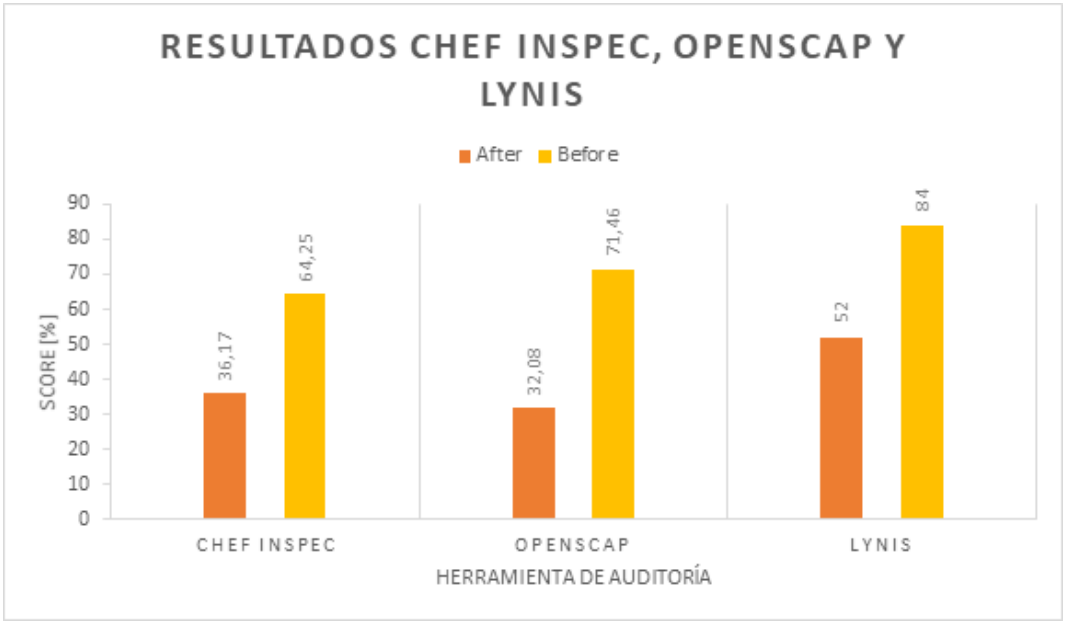

FIGURA 7.1: Resultados Chef InSpec, OpenSCAP y Lynis

La Tabla 7.1 muestra los resultados obtenidos al ejecutar las herramientas de auditoría de seguridad antes y después de aplicar los controles de seguridad disponibles con Egida.

Como podemos ver, tras aplicar Egida a la máquina, hemos conseguido aumentar la puntuación de hardening en un 28 % en la herramienta **Chef InSpec**, un 39,38 % en **OpenSCAP** y un 31 % en la herramienta de auditoría **Lynis**. Estos resultados muestran cómo el hardening adecuado de una máquina puede aumentar considerablemente su seguridad en comparación con su configuración inicial. Este incremento es aún más notable si tenemos en cuenta que no se han implementado todos los controles de seguridad disponibles de los CIS Benchmarks, por lo que este valor podría ser aún mayor.

Para probar el segundo apartado de este caso de uso hemos ejecutado uno a uno todos los CIS Controls disponibles en Egida y hemos comprobado si se corresponden con los esperados. La Tabla 7.2 muestra los resultados de los benchmarks esperados y ejecutados por cada CIS Control (sólo se muestran los CIS Controls con alguna tarea en los CIS Benchmarks del sistema objetivo). Como se puede ver en la Tabla 7.2, todos los controles de seguridad esperados se han ejecutado correctamente.

CIS Control	Benchmarks esperados	Benchmarks ejecutados
2.- Inventory and Control of Software Assets	8	8
3.-Continuous Vulnerability Management	6	6
4.-Controlled Use of Administrative Privileges	15	15
5.-Secure Con guration for Hardware and Software on Mobile Devices, Laptops, Workstations and Servers	52	52
6.-Maintenance, Monitoring and Analysis of Audit Logs	10	10
8.-Malware Defenses	3	3
9.-Limitation and Control of Network Ports, Protocols and Services	34	34
13.-Data Protection	1	1
14.-Controlled Access Based on the Need to Know	23	23
16.-Account Monitoring and Control	19	19
Total	171	171

TABLA 7.2: CIS Controls benchmarks ejecutados vs esperados

Para probar los dos últimos apartados de este caso de uso hemos desarrollado un script en Aspida que realiza todas las tareas relacionadas con sshd y cambia el valor de las variables por defecto. En fragmento de código siguiente, podemos ver

el código de Aspida necesario para ejecutar todas las tareas relacionadas con sshd y configurar el puerto ssh y las variables de usuario.

```
...
TASKS : {
  tags: ["sshd"];
}
VARS : {
  "user_ssh": "antonio";
  "sshd_access": {
    "ssh_port": 372
  };
}
```

LISTING 7.1: Aspida Use Case 1

7.1.2 Personalización de las operaciones de seguridad según las características de la máquina de destino

Para probar este caso de uso, hemos desarrollado un script en Aspida que, mediante el uso de sentencias condicionales (IF-ELIF-ELSE), comprueba el estado de la máquina destino y actúa en consecuencia. Como podemos ver en el fragmento de código siguiente, el lenguaje nos permite obtener información como el estado actual de un servicio o los puertos abiertos.

```
...
TASKS : {
  IF "services.apache" == "RUNNING" {
    IF "machine.open_port" == 80 {
      exclusions: ["2.2.10"];
    }
  }
  IF "machine.gui" == true {
    exclusions: ["2.2.2"];
  }

  IF "packages.telnet" == "INSTALLED" {
    IF "services.ssh" == "RUNNING" {
        points: ["2.3.4"];
    }
  }

  IF "hardscores.lynis" <= 50 {
    controls: ["2.6", "3.4", "3.5", "4.3", "4.4", "4.5",
       "4.8", "4.9", "5.1", "5.5", "8.3", "9.4"];
  } ELIF "hardscores.lynis" <= 60 {
    controls: ["5.1", "5.5", "8.3", "9.4"];
  } ELSE {
    controls: ["9.4"];
  }
}
...
```

LISTING 7.2: Aspida Use Case 2

En este script, estamos comprobando si el servicio **Apache** se está ejecutando y el puerto 80 está en uso. Si se cumplen estas dos condiciones, añadimos el punto 2.2.10 a la lista de tareas que no deben ejecutarse, ya que pueden afectar al correcto funcionamiento de este servicio.

Además, si la máquina de destino tiene una Interfaz Gráfica de Usuario activa, también añadimos el punto 2.2.2 a la lista de exclusiones ya que este punto puede afectar a los sistemas que hagan uso de este tipo de interfaces y perjudicar su rendimiento.

En el caso de que el paquete *telnet* esté instalado y además se esté ejecutando el servicio *SSH*, añadimos el punto 2.3.4 a la lista de ejecución ya que se encarga de deshabilitar *telnet*.
Por último, comprobamos la puntuación obtenida en la máquina de destino con la herramienta de auditoría Lynis y actuamos en función de esta puntuación aplicando más o menos controles de seguridad.

Tomar decisiones basadas en las características de la máquina objetivo nos da un extra de flexibilidad en el desarrollo de scripts de configuración que nos permiten especializar el hardening de una máquina. Esto nos facilita el blindaje en profundidad de un sistema aplicándole todos los controles de seguridad disponibles, exceptuando aquellos que puedan interferir en su funcionamiento.

7.1.3 Detección temprana de errores de configuración

Finalmente, para probar este caso de uso hemos desarrollado algunos scripts que hacen uso de variables no definidas, tienen errores léxicos o semánticos, o intentan ejecutar tareas de Nivel-2 Workstation.

En las figuras 7.2 y 7.3 podemos ver ejemplos de la salida producida por el compilador sobre errores léxico-sintácticos.

```
SyntaxError on line 7 : 5 --> missing 'HOST' at ':'
SyntaxError on line 9 : 6 --> missing 'TASKS' at ':'
```

FIGURA 7.2: Errores léxicos en tiempo de compilación

```
SyntaxError on line 13 : 7 --> mismatched input '{' expecting {'ELIF', 'ELSE'}
```

FIGURA 7.3: Errores sintácticos en tiempo de compilación

La figura 7.4 muestra un ejemplo del warning o advertencia producida al intentar ejecutar una tarea que utiliza una variable no definida (se utiliza su valor por defecto). Un ejemplo de los warnings producidos por el compilador cuando se intenta ejecutar una tarea de Nivel 2 se puede ver en la Figura 7.5.

```
WARNING: Variable  sshd_access not defined --> Its default value will be used
```

FIGURA 7.4: Warnings por variables no definidas

```
WARNING: Level 2 rule ( Ensure SSH AllowTcpForwarding is disabled ) is to be executed
```

FIGURA 7.5: Warnings por la ejecución de tareas de Nivel 2

En Aspida, el uso de variables no definidas por defecto y la ejecución de tareas de Nivel 2 se están marcando como warnings en lugar de errores. Esto es así ya que los errores detienen la ejecución del script mientras que los warnings permiten notificar al usuario sobre acciones que pueden ser peligrosas en determinados escenarios, pero no detienen el script.

Detectar errores en tiempo de compilación (como por ejemplo olvidar cambiar el valor de los puertos permitidos por el firewall o configurar el usuario de acceso SSH) nos permite, además de no perder tiempo esperando a que se produzca el error, evitar posibles configuraciones por defecto que puedan ser vulnerables o evitar la ejecución de controles de seguridad que puedan afectar al correcto funcionamiento del sistema.

CAPÍTULO

8

CONCLUSIONES Y TRABAJO FUTURO

8.1 Conclusiones y Trabajo Futuro

El sistema propuesto Egida, permite realizar despliegues automatizados de configuraciones de seguridad a uno o varios componentes de una infraestructura. El hardening de la máquina objetivo se puede personalizar en función de sus características para poder implementar el mayor número de controles de seguridad que no interfieran con su correcto funcionamiento. De esta forma, se crean diferentes perfiles de seguridad que pueden ser definidos por un administrador de sistemas o de seguridad entrenado.

Estos perfiles de seguridad se pueden definir gracias al Lenguaje de Dominio Específico que tiene Egida. El compilador del lenguaje realiza las comprobaciones de las sentencias que obtienen información de las características de la máquina objetivo y permite cambiar los controles de seguridad que se le van a aplicar mediante sentencias condicionales.

Por lo tanto, con Egida complementamos el trabajo de proyectos como SCAP o los CIS Benchmarks añadiendo una capa de personalización y prevención de errores basada en las características de cada máquina.

En el futuro se pretende mejorar la API de Egida para poder obtener más información de la máquina destino. Algunas características que se plantean son la comprobación de la existencia o no de determinados ficheros, comprobar los permisos sobre directorios y ficheros o comprobar valores de configuración de servicios conocidos.

También queremos mejorar el Lenguaje de Dominio Específico incluyendo elementos que permitan facilitar la tarea al desarrollador. Por ejemplo, la posibilidad de poder imprimir por consola mensajes (error, info y warning) o la declaración y uso de variables.

Todo el código fuente del proyecto Egida y la documentación de su uso se encuentra disponible en `https://egida-kassandra.github.io/egida`.

8.2 Difusión de Resultados

Se ha enviado el artículo con título ***Egida: Automated security configuration deployment systems with early error detection*** a la revista ***Computers & Security*** de *Elsevier* (*CiteScore 2020: 8.5, SJR 2020: 0.861, SNIP 2020: 2.535*).

Además, el proyecto **Egida**, junto con el proyecto **Kassandra**, han quedado como semi-finalistas en la **INNCYBER Innovation HUB 2020** [1] organizada por *EDP*, *Altice* y *Portuguese Air Force*.

FIGURA 8.1: Certificado INNCYBER Innovation Hub 2020

El proyecto **Egida** se incorporará también como parte del material de enseñanza impartido en la asignatura de *Seguridad* impartida en la Escuela de Ingeniería Informática de la Universidad de Oviedo [30].

[1] https://www.inncyberinnovationhub.com/

BIBLIOGRAFÍA

[1] I. J. S. Software y systems engineering, *ISO/IEC 19770-8:2020*, en, https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/25/72588.html, Accessed: 2021-05-12, 2020.

[2] D. Waltermire, S. Quinn, H. Booth, K. Scarfone y D. Prisaca, *The technical specification for the security content automation protocol (scap) version 1.3*, en, 2018. DOI: https://doi.org/10.6028/NIST.SP.800-126r3.

[3] C. of Internet Security, *Cis benchmarks*, en-US, https://github.com/dev-sec/cis-dil-benchmark, Accessed: 2021-05-20, 2020.

[4] ——, *Mapping and compliance*, https://www.cisecurity.org/cybersecurity-tools/mapping-compliance, Accessed: 2021-05-20, 2020.

[5] F. Utz, *Ubuntu1804-cis*, https://github.com/florianutz/Ubuntu1804-CIS, 2020.

[6] H. Harcourt, *Ansible-rhel8-cis-benchmarks*, https://github.com/HarryHarcourt/Ansible-RHEL8-CIS-Benchmarks, 2019.

[7] J. Soto, *Jshielder*, 2019.

[8] J. M. Redondo y D. Cuesta, «Towards improving productivity in nmap security audits», *Journal of Web Engineering*, vol. 18, n.º 7, págs. 539-578, 2019. DOI: 10.13052/jwe1540-9589.1871.

[9] M. Stytz, «Considering defense in depth for software applications», *IEEE Security and Privacy*, vol. 2, n.º 1, págs. 72-75, 2004. DOI: 10.1109/MSECP.2004.1264860.

[10] L. Cleghorn, «Network defense methodology: A comparison of defense in depth and defense in breadth», *Journal of Information Security*, vol. 4, págs. 144-149, 2013. DOI: 10.4236/jis.2013.43017.

[11] D. Kuipers y M. Fabro, «Control systems cyber security: Defense in depth strategies», Idaho National Laboratory (INL), inf. téc., 2006.

[12] W. R., D. Weaver y D. Farwood, *Guide to Network Defense and Countermeasures*. Cengage Learning, 2013, ISBN: 9781133727941.

[13] G Peng, «CDN: Content Distribution Network», *ArXiv*, vol. cs.NI/0411, 2004. dirección: http://arxiv.org/abs/cs.NI/0411069.

[14] P. Wurzinger, C. Platzer, C. Ludl, E. Kirda y C. Kruegel, «SWAP: Mitigating XSS attacks using a reverse proxy», en *Proceedings of the 2009 ICSE Workshop on Software Engineering for Secure Systems, SESS 2009*, 2009, págs. 33-39, ISBN: 9781424437252. DOI: 10.1109/IWSESS.2009.5068456.

[15] N. Provos, «A Virtual Honeypot Framework», en *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, ép. SSYM'04, USA: USENIX Association, 2004, pág. 1.

[16] J. Steiner, C. Neuman y J. Schiller, «Kerberos: An Authentication Service for Open Network Systems», en *USENIX CONFERENCE PROCEEDINGS*, 1988, págs. 191-202.

[17] J. S. Broderick, «Isms, security standards and security regulations», *Information Security Technical Report*, vol. 11, n.º 1, págs. 26-31, 2006. DOI: https://doi.org/10.1016/j.istr.2005.12.001.

[18] I. T. L. Computer Security Division, *Security content automation protocol: Csrc*, https://csrc.nist.gov/projects/security-content-automation-protocol, Accessed: 2021-05-25, 2020.

[19] P. Mell y K. Scarfone, *The common configuration scoring system (ccss): Metrics for software security configuration vulnerabilities*, en, 2010. DOI: https://doi.org/10.6028/NIST.IR.7502.

[20] Cyberx-Mw y Knowlton, *Security technical implementation guides (stigs)*, https://public.cyber.mil/stigs, Accessed: 2021-05-17, 2020.

[21] B. Berger, «Trusted computing group history», *Information Security Technical Report*, vol. 10, n.º 2, págs. 59-62, 2005. DOI: https://doi.org/10.1016/j.istr.2005.05.007.

[22] C. of Internet Security, *Cis controls*, en-US, https://www.cisecurity.org/controls/, Accessed: 2021-05-20, 2020.

[23] H. Winarno, F. Yasin, M. A. Prasetyo, F. Rohman, M. R. Shihab y B. Ranti, «It infrastructure security risk assessment using the center for internet security critical security control framework: A case study at insurance company», en *2020 3rd International Conference on Computer and Informatics Engineering (IC2IE)*, 2020, págs. 404-409. DOI: 10.1109/IC2IE50715.2020.9274594.

[24] B. Shamma y col., «Implementing cis critical security controls for organizations on a low-budget», Tesis doct., 2018.

[25] M. Spichkova, B. Li, L. Porter, L. Mason, Y. Lyu e Y. Weng, «Vm2: Automated security configuration and testing of virtual machine images», *Procedia Computer Science*, vol. 176, págs. 3610-3617, 2020. DOI: https://doi.org/10.1016/j.procs.2020.09.025.

[26] M. Olenčin y J. Perháč, «Automated configuration of a linux web server security», en *2019 IEEE 15th International Scientific Conference on Informatics*, 2019, págs. 000 491-000 496. DOI: 10.1109/Informatics47936.2019.9119272.

[27] N. Al-Safwani, Y. Fazea y H. Ibrahim, «Iscp: In-depth model for selecting critical security controls», *Computers and Security*, vol. 77, págs. 565-577, 2018. DOI: https://doi.org/10.1016/j.cose.2018.05.009.

[28] K. Durkota, V. Lisý, B. Bošanský, C. Kiekintveld y M. Pěchouček, «Hardening networks against strategic attackers using attack graph games», *Computers and Security*, vol. 87, pág. 101 578, 2019. DOI: https://doi.org/10.1016/j.cose.2019.101578.

[29] OpenSCAP, *Openscap portal*, http://www.open-scap.org/, Accessed: 2021-05-20, 2020.

[30] J. M. Redondo, «Improving concept learning through specialized digital fanzines», en *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 2021, págs. 134-143. DOI: 10.1109/ICSE-SEET52601.2021.00023.

APÉNDICE

A

PLAN DE GESTIÓN DE RIESGOS

A.1 Metodología

Este Plan de Gestión de Riesgos es el plan para el conjunto de riesgos identificados para el proyecto Egida. Según el PMBOK "El riesgo de un proyecto es un evento o condición incierta que, de producirse, tiene un efecto positivo o negativo en uno o más de los objetivos del proyecto, tales como el alcance, el cronograma, el costo y la calidad." La metodología para la gestión del riesgo tiene dos aspectos diferentes:

1. **Metodología General**. Metodología para el conjunto del Plan de Gestión de Riesgos.
2. **Metodología de Gestión de Riesgos**. Metodología para el ciclo de vida de todos los riesgos y las interrelaciones entre los riesgos identificados.

A.1.1 Metodología General

La Metodología general describe el proceso de gestión de riesgos para el conjunto del proyecto: desde el inicio hasta el final de los trabajos relacionados con el proyecto.

1. **Identificación inicial de riesgos**. Primer contacto con los riesgos del proyecto. Algunas técnicas útiles para la identificación inicial de riesgos son las siguientes:
   (a) Tormenta de ideas
   (b) Juicio de expertos
   (c) Tabla DAFO
2. **Elaboración del Plan de Gestión de Riesgos**. Creación del Plan de Gestión, los criterios para la gestión de riesgos, el Registro de Riesgos y la Hoja de Datos de riesgos para todos los riesgos priorizados.
3. **Monitorización de Riesgos** mientras que el proyecto está avanzando:
   (a) **Evaluación de riesgos Regular**. De vez en cuando, el riesgo debe ser evaluado con el fin de asegurar que no representan una amenaza para el avance del proyecto.
   (b) **Actualización del Plan de Gestión de Riesgos**. Muchos riesgos se mantienen durante todo el proyecto, pero no todos ellos. Algunos riesgos tienen un ciclo de vida más corto que la duración del proyecto. En el otro lado, ya que el proyecto evoluciona, nuevo riesgo puede aparecer u otros riesgos ya identificados, pero no importantes crecidas hasta el momento en que son una amenaza para el proyecto. En este momento se deben añadir al registro principal riesgo.
   (c) **Proyecto de actualización del plan** causas por las decisiones de la evaluación de riesgos. Las decisiones deben ser tomadas durante el proyecto con el fin de evitar o, al menos, minimizar el impacto de los riesgos.
4. **Informe Final de Riesgos**. Al final del proyecto, una voluntad informe final muestra toda la evolución del riesgo.

A.1.2 Metodología de Gestión de Riesgos

Los aspectos más técnicos de la gestión de riesgos están involucrados en este aspecto de la metodología. Para ese proyecto definimos seis pasos:

1. **Planificar la Gestión de Riesgos**: Es el proceso por el cual se define cómo realizar las actividades de gestión de los riesgos para un proyecto.

2. **Identificar los Riesgos**: Es el proceso por el cual se determinan los riesgos que pueden afectar el proyecto y se documentan sus características.

3. **Realizar el Análisis Cualitativo de Riesgos**: Es el proceso que consiste en priorizar los riesgos para realizar otros análisis o acciones posteriores, evaluando y combinando la probabilidad de ocurrencia y el impacto de dichos riesgos.

4. **Realizar el Análisis Cuantitativo de Riesgos**: Es el proceso que consiste en analizar numéricamente el efecto de los riesgos identificados sobre los objetivos generales del proyecto.

5. **Planificar la Respuesta a los Riesgos**: Es el proceso por el cual se desarrollan opciones y acciones para mejorar las oportunidades y reducir las amenazas a los objetivos del proyecto.

6. **Monitorizar y Controlar los Riesgos**: Es el proceso por el cual se implementan planes de respuesta a los riesgos, se rastrean los riesgos identificados, se monitorizan los riesgos residuales, se identifican nuevos riesgos y se evalúa la efectividad del proceso contra riesgos a través del proyecto

A.1.3 Conceptios Generales

Con el fin de evitar la ambigüedad, se deben definir tres conceptos utilizados en la Gestión de Riesgos:

- **Riesgo**: es cualquier evento que pueda causar un efecto en el proyecto. El efecto bien podría ser una amenaza o podría ser una oportunidad. La primera de ellas debe ser evitada y la segunda debe ser explotada.

- **Factor de riesgo**: Los factores de riesgo son circunstancias asociadas con el riesgo de que aumenta la posibilidad de conseguir los efectos descritos por el riesgo. Por lo general, los riesgos son imposibles o difíciles de medir o controlar directamente y las decisiones se toman controlando los factores de riesgo con el fin de evitar o minimizar (o potenciar) los efectos de riesgo.

- **Indicadores de Riesgo**: Los indicadores de riesgo son elementos asociados al riesgo y / o sus factores, que se puede medir y sirve como información acerca de la posibilidad de que se produzca el efecto del riesgo.

A.2 Herramientas y Tecnología

A.2.1 Tormenta de Ideas

Esta técnica será utilizada en un principio para la identificación de la lista principal de los riesgos. Consiste en la discusión acerca de los principales objetivos del proyecto y todas las cosas que pueden ir mal.

A.2.2 Juicio de Expertos

Esta técnica será utilizada para la identificación de la lista principal de los riesgos junto con el punto A.2.1 Tormenta de Ideas.

A.2.3 Las Evaluaciones Periódicas

Regularmente, todos los indicadores serán evaluados de acuerdo con el plan establecido en la hoja de riesgo. Todos estos resultados se discutirán en las reuniones de riesgos.

A.2.4 Reuniones

Regularmente, se debe incluir un punto de discusión en el orden del día de las reuniones con el fin de tomar decisiones acerca de los riesgos del proyecto y sus efectos.

A.2.5 Delphi

En caso de desacuerdo sobre cualquier riesgo o decisión, se puede un método Delphi utilizar para resolver el problema.

A.2.6 Diagramas Causa-Efecto

Es necesaria, debido a la complejidad de las relaciones entre un riesgo y sus factores de riesgo, un diagrama de causa-efecto se puede utilizar con el fin de mostrar una representación gráfica.

A.2.7 Cálculos Estadísticos

En muchos casos se requerirán cálculos estadísticos para resolver la evaluación cuantitativa.

A.2.8 Otros

Se usarán varios tipos de herramientas y procesos matemáticos para hacer las evaluaciones cuantitativas.

A.3 Roles y Responsabilidades

Se prevén lo siguientes roles:

- **Responsable de riesgos**: Responsable de la gestión de riesgos. Este papel lo jugará el propio Jefe de Proyecto.
- **Auxiliar de riesgos**: Personas encargadas del seguimiento y monitorización regular de los riesgos.
- **Identificador de riesgos**: Todo el equipo es responsable de la identificación de nuevos riesgos

A.4 Presupuesto

N/A

A.5 Calendario

N/A

A.6 Categorías de Riesgo

A continuación, se definen las categorías en las que un riesgo puede ser clasificado. Un riesgo puede pertenecer a una o más de las siguientes categorías (solo las subcategorías se utilizan para clasificar riesgos).

1. Técnico
    (a) Requisitos
    (b) Tecnología
    (c) Complejidad e Interfaces
    (d) Prestaciones y fiabilidad
    (e) Calidad
2. Externo
    (a) Subcontratistas y Proveedores
    (b) Regulación
    (c) Mercado
    (d) Usuario
    (e) Tiempo
3. Organizacional
    (a) Dependencias del Proyecto
    (b) Recursos
    (c) Financiación
    (d) Priorización
4. Gestión de Proyecto
    (a) Estimación
    (b) Planificación
    (c) Control
    (d) Comunicación

A.7 Definiciones de Probabilidad

	Impacto sobre los objetivos principales				
Objetivos	Muy Baja / 5 %	Baja / 10 %	Media / 20 %	Alta / 30 %	Muy Alta / 40 %
Presupuesto	Incremento del coste insignificante	Incremento del coste <10 %	Incremento del coste entre el 10–20 %	Incremento del coste entre el 20-30 %	Incremento del coste de más del 30 %
Planificación	Incremento del tiempo insignificante	Incremento del tiempo <5 %	Incremento del tiempo entre el 5-10 %	Incremento del tiempo entre el 10-15 %	Incremento del tiempo >15 %
Alcance	Reducciones del alcance inapreciables	Afectadas áreas poco importantes del alcance	Afectadas áreas importantes del alcance	Reducciones del alcance inaceptables para el cliente	El resultado final del proyecto no es realmente útil
Calidad	La reducción de la calidad es inapreciable	La reducción de la calidad solo es apreciable en momentos muy exigentes	La reducción de la calidad requiere de la aprobación del cliente	Reducción de la calidad inaceptable para el cliente	El resultado final del proyecto no es realmente útil

TABLA A.1: Plan de Gestión de Riesgos: Definiciones de impacto por objetivos

A.8 Matriz de Probabilidad e Impacto

Probabilidad	**Muy Alta**	***0,90***	0,05	0,14	0,27	0,50	0,81
	Alta	***0,70***	0,04	0,11	0,21	0,39	0,63
	Media	***0,50***	0,03	0,08	0,15	0,28	0,45
	Baja	***0,30***	0,02	0,05	0,09	0,17	0,27
	Muy Baja	***0,10***	0,01	0,02	0,03	0,06	0,09
			0,05	***0,15***	***0,30***	***0,55***	***0,90***
			Muy Bajo	**Bajo**	**Medio**	**Alto**	**Crítico**
			Impacto				

FIGURA A.1: Matriz de Probabilidad e Impacto

A.9 Planes de Contingencia

A.9.1 Riesgo número 3

1. Hacer una buena selección de repositorios de vulnerabilidades y de respuestas a vulnerabilidades en Sistemas Operativos.

   (a) Comprobar que dichos repositorios son actualizados periódicamente

   (b) En caso de que la vulnerabilidad no esté disponible en ningún repositorio incluirla en el repositorio temporal propio con un máximo de 6 horas de trabajo (Estimación: 6 * 20,00€ = 120€)

2. Actualizar los scripts de respuesta para que utilicen los repositorios.

   (a) Que cada Ingeniero Software actualice los scripts del módulo del que se encarga con un máximo de 4 horas de trabajo (Estimación: 4 * 20,00€ = 80€)

3. Actualizar la documentación del proyecto (Memoria y Artículo).

En caso de problemas, que se puedan resolver en un máximo de 10 horas (Estimación: 10 * 20,00€ = 200€).

A.10 Formatos de la Documentación

N/A

APÉNDICE

B

GNU FREE DOCUMENTATION LICENSE

B.1 PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "freeïn the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of çopyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

B.2 APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as ẓou". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version.ºf the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Sectionïs a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The Ïnvariant Sections.ªre certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The Çover Texts.ªre certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparentçopy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparentïs called .ºpaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page"means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page"means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher"means any person or entity that distributes copies of the Document to the public.

A section .Entitled XYZ"means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates

XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as .Acknowledgements", "Dedications", .Endorsements", or "History".) To "Preserve the Title.of such a section when you modify the Document means that it remains a section .Entitled XYZ.according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

B.3 VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

B.4 COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

B.5 MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement. C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "Historyïn the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History"section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. For any section Entitled .Acknowledgements.or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section Entitled .Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section to be Entitled .Endorsements.or to conflict in title with any Invariant Section. O. Preserve any Warranty Disclaimers. If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled .Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you

may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

B.6 COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "Historyïn the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled .Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled .Endorsements".

B.7 COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

B.8 AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an .ªggregateïf the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

B.9 TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled .Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

B.10 TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

B.11 FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/licenses/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License .ºr any later version.ªpplies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

B.12 RELICENSING

"Massive Multiauthor Collaboration Site"(or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration"(or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

ÇC-BY-SA"means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

Ïncorporate"means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is .[el]igible for relicensingïf it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

APÉNDICE

C

ARTÍCULO

Egida: Automated security configuration deployment systems with early error detection

Antonio Paya[a], Alba Cotarelo[a] and Jose Manuel Redondo[a,*]

[a]Department of Computer Science, University of Oviedo, Science Faculty, Office 240, C/Federico Garcia Lorca S/N, 33007, Oviedo, Spain

ARTICLE INFO

Keywords:
security control
automation
defense in depth
error detection

ABSTRACT

Automated deployment of security controls is a very important part of deploying a defense-in-depth strategy to secure machine infrastructures. This paper describes a prototype of a technique to perform automated deployment of validated and international security controls in a more controlled way using a DSL. This way, according to the target machine configuration or characteristics, this controlled deployment is less prone to negatively impact legitimate running services, and better capture system administrator expert knowledge, being able to share automated security scripts that obtain better hardening results. The initial results of this technique are promising enough to apply them on educational environment and develop them further to be applied on real infrastructures requiring this kind of security automation.

1. Introduction

The process of securing IT systems requires the correct implementation and application of international information security management standards (*Information Security Management System, ISO 27001* [19]). Unfortunately, the current proliferation of attacks shows that in some cases these implementations are not complete or adequate.

The correct implementation of an *Information Security Management System (ISMS)* can be easier if we use tools or mechanisms that allow us to automate parts or all of the process. The *SCAP (Security Content Automation Protocol)* is an international reference that allows achieving this automation process. In addition, provides standard formats and nomenclatures that allow defining and exchanging information between end users and tools.

However, automation alone is not enough; the actions to be performed (international, supported, validated, etc.) must also be selected. There are several sources of security controls available, but in this project we used *CIS Benchmarks* [4] because, in addition to the advantages they offer, by using *CIS Controls* we can map the actions performed [6] to elements present in standards such as *ISO 27001*, thus facilitating their implementation.

Nevertheless, unsupervised application of all *CIS Benchmarks* may impair performance or invalidate services that are legitimately running on the system (e.g. *CIS Benchmarks* contain tasks categorised as *Level 2 Workstation*, which are catalogued as potentially problematic, as they may affect the proper functioning of the system if is executed without proper preparation of the affected systems). Therefore, it is necessary to have a detailed control of the operations to be performed, and the expert knowledge of an administrator to avoid problems in their deployment.

This project proposes a system called Egida, that creates a *Domain Specific Language* (DSL) to model this knowledge and apply the *CIS Benchmarks* on infrastructures in such a way that it is possible to adapt them to specific cases. Egida also prevents problems in their implementation before they occur (unlike other existing projects and tools where errors occur during the execution itself). This project aims to model expert knowledge and make decisions based on what is found in each system to be secured.

To achieve this, we use an architecture that allows us to provide system characteristics with various degrees of detail that will allow the application of security controls of the *CIS Benchmarks* in a more intelligent and flexible way, adapting to each case, being also possible to detect potential problems previously to their execution. In this way, a trained system or security administrator can create security profiles that depend on the characteristics of the target system. These profiles can specify which security controls are to be applied based on the needs of the target system, or which security controls should be avoided.

The proposed system aims to be initially deployed on an educational environment (low resource consuming system) but can be scaled up to professional environment uses in the future.

2. Related Work

2.1. Defense in Depth Strategies

The *Defense in Depth* philosophy is a concept inherited from military defense that applies to every computer in an infrastructure [22]. A well-defined and well-implemented defense-in-depth strategy can prevent and avoid a wide variety of attack risks, and provide monitoring and alert tools to identify unauthorized access [7]. In this strategy, different security measures, divided into layers, are developed to minimize them [11]. These layers overlap each other to cover each other's potential deficiencies [15].

Typical defensive mechanisms implemented are network traffic analysis, behavioral analysis, anti-virus software, data

*Corresponding author
antoniopaya@outlook.com (A. Paya); albact7@gmail.com (A. Cotarelo); redondojose@uniovi.es (J.M. Redondo)
ORCID(s): 0000-0003-3733-3805 (A. Paya); 0000-0002-1514-0423 (A. Cotarelo); 0000-0002-0939-0186 (J.M. Redondo)

integrity analysis software, restrictions embedded in software code, user restrictions and system and infrastructure hardening.

Computer system hardening is therefore a key part in a defense-in-depth strategy. For that, we need to define a hardening strategy based on proper and verified guidelines that achieves the following features:

- A complete list of security controls to apply to achieve a desired security level for a specific OS or product
- A justification of why these controls are needed, in case they interfere with the expected system activities or functionalities
- How to check if these controls are already applied
- How to apply these controls when they are not present

2.2. Security Controls and Standards

In order to properly maintain secure systems and implement defense-in-depth strategies, it is necessary to correctly implement international information security management standards *ISO 27001, ISMS* [3].

The implementation of an *ISMS* can be facilitated with frameworks that allow automation, such as the *SCAP* protocol and the *TCG* framework.

2.2.1. SCAP

The *Security Content Automation Protocol (SCAP)* [8] is a set of NIST security automation standards for assessing compliance with security policies and detecting the presence of vulnerable versions of software and system configurations [23]. The *SCAP* protocol provides standard formats and nomenclatures for defining and exchanging information between end users and tools, and has a large number of components that can be classified into the following categories according to their purpose:

- Enumerations (*CVE*, *CCE* and *CPE*)
- Metrics (*CVSS* and *CCSS*)
- Languages (*XCCDF*, *OVAL* and *OCIL*)
- Report Format (*ARF* and *AI*)
- Integrity (*TMSAD* and *SWID*)

CVE (Common Vulnerability and Exposures) is a system for naming and documenting known vulnerabilities referenced with a unique identifier. Each vulnerability contains a description, which versions of the software are affected, the possible solutions (if any) and how to configure it to mitigate the vulnerability, as well as references to publications or forum or blog posts where the vulnerability has been made public or its exploitation is demonstrated. In addition, each vulnerability contains a public reference on the web. *CCE (Common Configuration Enumeration)* and *CPE (Common Platflorm Enumeration)* are very similar to CVE but their objectives are to document system configurations and to collect and identify technology systems, software and packages respectively.

CVSS (Common Vulneravility Score System) is a scoring system that allows estimating the impact of vulnerabilities based on their characteristics in order to define it. The *CCSS (Common Configuration Score System)* [12] is a derivation of the *CVSS* (like the *CCE* is a derivative of the *CVE*). The *CCSS* measures the impact of security flaws that depend on certain product configurations.

The *SCAP* protocol has languages such as *XCCDF (Extensible Configuration Checklist Description Format)*, based on *XML*, developed by NIST, the NSA, the MITRE Corporation, and the U.S. Department of Homeland Security to specify security checklists, reference points, and configuration documentation. The goal of the *XCCDF* language is to replace manually written security analysis and hardening documentation.

Other languages of the *SCAP* protocol based on *XML* are *OVAL (Open Vulnerability and Assessment Language)*, which allows to standardize the reporting and assessment of system configuration status, and *OCIL (Open Checklist Interactive Language)*, which specifies the necessary guidelines to define a series of less automatable checklists that require more human intervention.

Languages such as *OVAL*, *OCIL* or *XCCDF* favor and speed up the exchange of information between cybersecurity professionals, software vendors and auditors, as well as enable the development of automated tools and solutions. There are files written in these languages capable of automatically verifying or enforcing a substantial number of operating systems and/or software products. One example is the *Security Technical Implementation Guides (STIGs)* of the Department of Defense (DoD) of United States [9].

Regarding the formats for reporting and documentation, *Asset Identification (AI)* provides the necessary constructs to uniquely identify assets based on known identifiers and/or known information about the assets. The *ARF (Asset Reporting Format)* is a data model that allows expressing the format for conveying information about such assets and their relationship to reports. This standardized data model facilitates reporting, correlation and merging of asset information across and between organizations.

The *SWID (Software Identification)* is a label system defined by the *ISO/IEC 19779-2* [19] standard in order to identify software products on the market. On the other hand, the *TMSAD (Trusted Model for Security Automation Data)* describes a common trust model that consists of recommendations on how to use existing specifications to represent signatures, hashes, key information, and identity information in the context of an *XML* document within the scope of security automation.

However, *SCAP*, *OVAL* and *XCCDF* share a number of shortcomings that prevent them from being successfully applied (or only partially applied in multiple scenarios):

- The publicly available files do not cover some very popular products used by private companies

- Several profiles just implement automated security control verification. Automated remediation is not or partially supported. In other cases, these files are only to control manual security control verification and application
- Some products specifications may not be properly maintained and files for new product versions may not be available for a long time.
- Finally, and related to the second shortcoming, these hardening files may not fully capture the security requirements of private companies with services exposed to the Internet, whose users may require a different usage profile.

2.2.2. TCG (Trusted Computing Group)

The *TCG (Trusted Computing Group)* [2] is a consortium of more than 200 companies that develop, define and promote a set of open source hardware specifications to protect systems from attacks that cannot be protected by purely software solutions to obtain a reliable platform. To this end, the *TCG* defines three characteristics that these systems must meet: (i) Measuring their own integrity, logging and reporting, (ii) Protected capabilities and (iii) Ensuring the accuracy of a component's status information.

However, a major obstacle in the use of *TCG-based* solutions is the absence of a public repository (published by the software vendor) of hash values of all software components. In addition, the static and rigid whitelisting approach is not well suited to large-scale distributed environments.

2.2.3. CIS-CSC (Center of Internet Security – Critical Security Control)

The *Center for Internet Security Critical Security Controls for Effective Cyber Defense* or also called *CIS-CSC (Center of Internet Security - Critical Security Control)* [5] is a collection of best practice guidelines that organizations can implement to reduce their cyber-attack surface significantly. These guidelines are formulated by a group of information technology experts using information obtained from real attacks and their effective defenses.

These guides are made up of 20 (rev. 7, 2018) and 18 (rev. 8, 2021) activities called critical security controls (*CSC*) and are divided into three categories that allow an organization to be classified into *Implementation Groups (IG)*. Each *IG* defines which security controls an organization should implement depending on the resources it has at its disposal, as well as its level of risk. This allows the controls to can be used by almost any type of organization regardless of size or resources [24, 18].

- The first *IG* includes basic security controls, such as those related to inventory and control of software and hardware assets, controlled use of administrative privileges or maintenance and monitoring.
- The second *IG* contains security controls that should be applied by organizations with a higher risk of exposure to attacks, as they focus on aspects such as perimeter protection, malware defenses, access controls or secure configuration of firewalls, routers and switches.
- The third *IG* is composed of security controls based on application software security, incident response and management or penetration testing and red team exercises.

CIS Controls can work either as standalone resources or as companions to additional frameworks and provide a mapping [6] of their security controls to other frameworks such as NIST or *ISO/IEC 27001*. In addition, *SCAP* protocol automation can be performed via *CIS Benchmarks* but at a cost.

The *CIS Benchmarks* are documented industry best practices for securely configuring IT systems, software, and networks. There are currently over 100 benchmarks, that cover a large number of vendor product families and contain a detailed list of security configuration tasks applicable to that product. The *CIS Benchmarks* provide a mapping applicable to the *CIS Controls* that allow us to apply these security controls to the product.

In addition, the security controls are divided into two profiles according to their impact on the product's performance:

- The *Level 1* profile corresponds to tasks that can be implemented fairly quickly and are designed not to have a major impact on product performance. The objective of this profile is to reduce the attack surface by strengthening security without hampering its functionality.
- The *Level 2* profile is already considered defense-in-depth and is intended for critical environments. These tasks can affect the proper functioning if not applied properly.

2.3. Automated Tools for Security Configuration and Verification

The existence of the aforementioned guidelines and standards such as *SCAP* or *TCG*, allows the appearance of automatic hardening tools that try to comply with most of their sections such as *VM2* [21], which automatically generates virtual machines, applying some of the *CIS Benchmarks* to them, or the configuration script for a Linux web server proposed by Michal Olencin [13]. Another popular automation tool is *JShielder* [20], which allows the hardening of *LAMP (Linux-Apache-Mysql/MariaDB-PHP)* and *LEMP (Linux-Nginx-MySQL/MariaDB-PHP)*.

Research projects such as *ISCP* [1], or the one proposed by Karel Durkota et al. [10], use models to determine vulnerable controls and provide clear guidelines on how to perform control analysis and to represent the possible actions of an attacker using attack graphs respectively.

One of the most important projects, besides being the most popular implementation of the *SCAP* protocol, is the

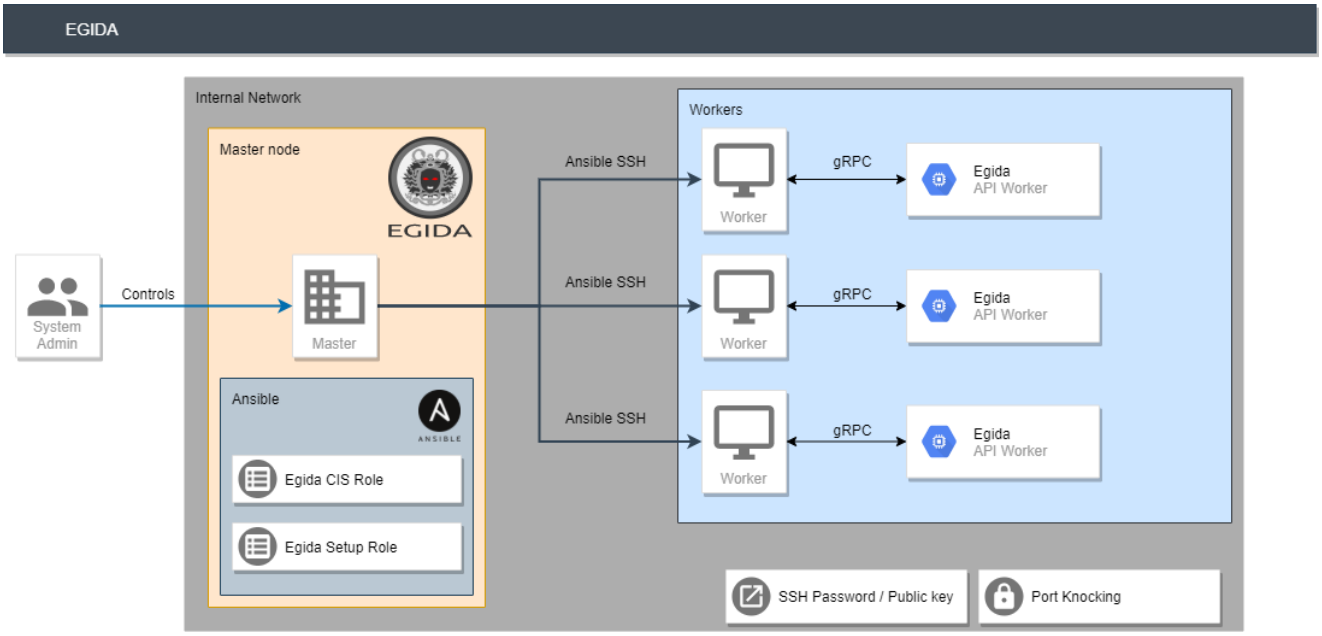

Figure 1: Egida structure

OpenSCAP Project [14]. It is a collection of open source tools for implementing and enforcing the *SCAP* standard, which are certified up to *SCAP* version 1.2. These tools comprise a multi-purpose specification framework that supports automated configuration, vulnerability checking and patching, technical control compliance activities, and security measurements. In addition, *OpenSCAP* has support for the use of files written in the *OVAL* and *XCCDF* languages, as well as allowing audit reports of a system based on already defined policies such as those of the *CIS* or *STIGs*. However, *OpenSCAP* tools do not work on Windows systems yet (only allow remote scanning of supported machines).

There are other tools that use these guidelines and standards to perform system audits and tests such as *Lynis*, *Chef Inspec* or *Prowler*. The result of these tools is a report with a checklist of the security controls that the system has passed or failed.

3. Egida

This research project proposes a server security orchestration system called Egida that allows to realize and deploy security configurations (security control checklists) on an infrastructure of machines or servers. These security configurations can shield and protect these servers by applying the desired security measures depending on the server profile.

3.1. Security Controls

Egida uses the *CIS Benchmarks* guides to implement system hardening in a highly automated way. As automation using *SCAP* is covered via the *CIS Benchmark* project (at a cost), with Egida we try to improve this automation introducing new elements within control verification and remediation phases.

We use the *Ansible* standard configuration deployment tools to distribute the security controls verification and implementation procedures. Instead of following the *SCAP* approach, that currently may be potentially incompatible with different systems, *Ansible* allows two additional very important advantages: the ability to work over any machine in which we have SSH access and the ability of not repeating operations that are detected as already applied. This greatly facilitates security control testing and enforcement propagation with a great degree of efficiency. *Ansible* is also a very popular centralized configuration management tool, used by very important companies worldwide.

There are other free implementations of the *CIS Benchmark* controls distributable via Ansible, but these could be incomplete, not properly maintained and, most importantly for our research, follow a not very granular approach, which may prevent legitimate system functionalities to work properly or efficiently if not deployed carefully (especially *CIS Level 2* security controls). With Egida we want to achieve a much more precise and flexible management of the security controls that may be deployed, creating smaller of more specialized control profiles that can be overlapped or combined at the user will. This ensures deploying just what it is needed (favoring more flexible machine specialization), and using control group combinations to create machine profiles, better suiting the deployed controls to certain typical usages. Aggregation of these security control profiles to suit concrete machine profiles is also knowledge that our research project generates that other users may be able to take advantage of or evaluate.

Even tough security controls are a very strong foundation of a secure system, there are also additional security software that may complement them and further enhance the infrastructure security, like reverse proxies, perimetral

firewalls, host-based firewalls, *NIDS, HIDS, WAFs*, etc. The Egida project also want to apply its flexible approach to these higher-level security elements, not only deploying them over specific machines, but also facilitating choosing the appropriate selection of *CIS Benchmarks* security controls most suitable for machines with this software. This aims to achieve the same benefits that projects like *JShielder* [20] offers, but the increased flexibility will allow users to better specialize machine missions, instead of creating a single machine concentrating all the protection software.

In addition, in order to achieve a better application of these security controls, Egida aims to obtain information on the state and configuration of the target system. This type of information can be used to change the hardening method during execution with the Egida DSL, allowing us to further automate the process.

3.2. Domain Specific Language

As we said, *XML* markup-based solutions like *OVAL* or *XCCDF* provide specifications to precisely detail and deploy security controls, but they do not allow to incorporate what we can call *semantic* checks. Ansible deployment errors are also discovered at runtime. The final goal of the Egida research project is to develop a DSL that allow users to deploy security control groups from the *CIS Benchmarks*, but performing early validations of the infrastructure machines and/or contents prior to the deployment. Things like inconsistencies with the machines that are expected to find, open ports, available software, necessary directories, required users, etc. could prevent a deployment to be successful. Egida aims to explore the limits of what can be validated prior to perform a deployment, so the probability of finding a runtime failure is significantly lower. A precise early error control may also enable administrators to solve problems in substantially less time, as the deployment procedure need not to be run subsequently until no error is found. This philosophy was successfully tried with *Nmap* scans in [17].

The purpose is to define *DSL* programs containing the hardening configuration of machines fulfilling a specific role of a company. This way, these programs contain all the hardening configuration (security controls plus their associated variable values) of their *HTTP* servers, *SGBD* servers or proxies. These files could be validated by experts additionally to the validation the *DSL* performs, and once determined to be valid, be distributed and reused all through the company infrastructure. If configuration changes or must be refined, we also want to check if the *DSL* is able to detect inconsistencies produced in the new versions prior to its deployment.

3.3. Structure

Egida works with a Master-Worker design pattern (see Figure 1) where the system administrator has control of the Master machine with the Egida software installed. From this Master machine the management of the security configurations of the Worker machines will be carried out.

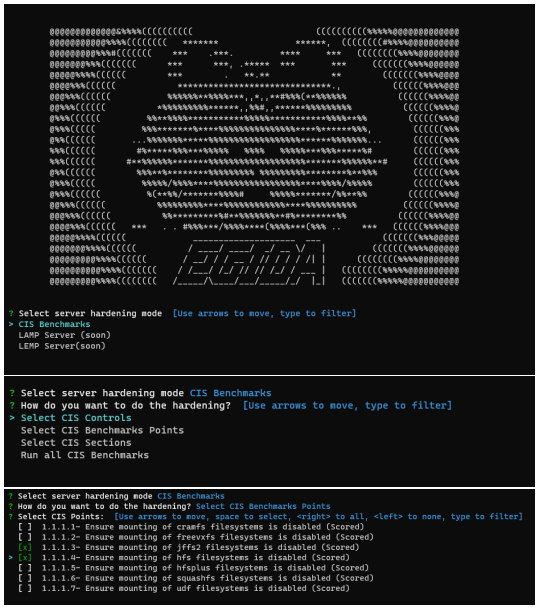

Figure 2: Egida menu

Egida is composed of four independent modules: *Egida Core*, *Egida Role CIS*, *Egida Role Setup* and *Egida API Worker*.

3.3.1. Egida Core

Egida Core is the main program of the project and is the interface that the system administrators use to interact with the machines they want to configure. Egida allows two modes of use: an interactive menu, and a *Domain Specific Language (DSL)* called Aspida.

Some of the tasks and operations can require variable data that depend on the type of installation or configuration that you want to apply to a machine (e.g. user names, passwords, etc.). To solve this, Egida allows the use of variables that can be modified by the system administrator user both in the interactive menu execution mode (through a *YAML* configuration file) and in the *Domain Specific Language* (within the variables block).

Menu It is an interactive console menu that allows you to select the hardening options you want to perform on a machine or set of machines. It allows the customization of the machine's security configuration by selecting the *CIS Controls*, *sections*, or specific points of the *CIS Benchmarks* to be applied. An example can be seen in the Figure 2.

Domain Specific Language As mentioned above, Egida has a *Domain Specific Language (DSL)* called Aspida that allows the development of predefined configuration scripts that act differently depending on the state of the target machine.

The language is aimed at facilitating the implementation of security controls linked to machine profiles, also allowing users to check different states, conditions, or variable values

so that they can prevent or react to deployment errors before configuration deployment occurs.

The language also aims to implement a semantic error prevention module so that a given program cannot be deployed if the language processor detects any kind of problem with the deployment infrastructure, incompatibilities between security controls, wrong variable values, and any other condition that may cause a run-time error and can be prevented at compile time (see 3.2).

Aspida allows the use of conditional structures in order to provide the user with the ability to specify different types of actions depending on the information obtained from the target machine. The main elements of the structure of the grammar is shown in Listing 1.

```
program : main hosts tasks variables?;

// Blocks
main : MAIN_KW ':' '{' main_content '}';
hosts : HOSTS_KW ':' STRING NS;
tasks : TASKS_KW  ':' '{' tasks_content '}';
variables : VARS_KW  ':' '{' vars_content '}';

...

// MAIN Block
main_content : main_prop (main_prop)*;
main_prop: name | connection | description

...

// TASKS Block
tasks_content : tasks_prop (tasks_prop) * | ifStat (elifStat)* elseStat;
tasks_prop : sections | points | controls | exclusions | tags;

...

// VARS Block
vars_content : vars_prop (vars_prop)*;
vars_prop : STRING ':' value | STRING ':' '{' vars_content '}';

...
```

Listing 1: Aspida Grammar

MAIN The MAIN block provides information about the script such as the name, a description or the type of connection to the target machine (Local or SSH).

HOSTS Name of the host or group of hosts on which the script is to be executed.

TASKS This block is the most important, as it contains all the information about the tasks to be performed. It allows you to define which sections or specific points of the *CIS Benchmakrs* to execute or exclude, as well as to select *CIS Controls*, or to execute all the tasks that correspond to a label (for example all the tasks that are related to SSH). In this block, If-ElsIf-Else conditional statements can be used along with obtaining values of the target machine state to control the script flow.

VARS The variables block allows you to give a value to each of the variables that will be used during script execution.

Some examples of programs written in Aspida are shown in Figure 3

3.3.2. *Egida Role CIS*

The Egida Role CIS module is an Ansible Role that defines all security configuration operations based on the *CIS Benchmarks* that Egida can perform.

This Ansible Role contains one task for each point corresponding to the *CIS Benchmarks*. Each of these tasks contains tags that indicate the control, point and section of the *CIS Benchmarks* to which it belongs, as well as some extra tags that provide extra information such as software or elements affected by that task.

In the evaluation of the prototype of this benchmark, most of the *Ubuntu 18.04 CIS Benchmarks* security controls are currently implemented, except for some tasks that require manual action and make automation difficult. In the future, the remaining security controls are expected to be developed.

3.3.3. *Egida Role Setup*

Is an Ansible Role that is responsible for installing the Egida API Worker service and the tools or options necessary for the correct functioning of Egida in the system.

3.3.4. *Egida API Worker*

The Egida API Worker is a *gRPC* API that is installed on a Worker machine through the Master Node, and works as a service to provide the Egida Core module with information about the machine, such as data about running or stopped services, machine information, installed packages or results of security audits using evaluation tools.

4. Evaluation

In this section we will try to evaluate our Egida proposal with some use cases. Subsequently, we will perform an analysis and discussion of the results obtained.

All measurements and tests have been performed using a 64-bit *Ubuntu 18.04 LTS* and its corresponding *CIS Benchmarks*. More Benchmarks covering other OS and/or products are expected to be incorporated in the future.

```
MAIN : {
    name: "Conditionals";
    connection: SSH;
    description: "Aspida Example";
}

HOST : "192.168.56.1";

TASKS : {
    IF "services.ufw" == "stopped"{
        sections: ["1.1", "1.2"];
        exclusions : ["1.1.1.3"];
    }
    ELIF "hardscores.lynis" ⩽ 50 {
        points: ["1.1.1.5"];
    }
    ELIF "services.apache" == "STOPPED" {
        points: ["1.1.1.4"];
    }
    ELSE {
        controls: ["9.4"];
    }
}
```

```
MAIN : {
    name: "SSH Config";
    connection: LOCAL;
    description: "Aspida Test";
}

HOST : "localhost";

TASKS : {
    IF "machine.open_port" == 22 {
        sections: ["5.1", "5.3", "5.4"];
    }
    ELSE {
        sections: ["5.1", "5.2", "5.3", "5.4"];
    }
}

VARS : {
    "user_ssh": "antonio";
    "password": {
        "max_days": 365;
        "min_days": 7;
        "warn_age": 7;
        "inactive": 30;
    };
    "nameservers": ["8.8.8.8", "8.8.4.4"];
}
```

Figure 3: Aspida DSL examples

4.1. Use Cases

In order to evaluate the Egida system, we have defined 3 use cases composed of several functionalities that should be fulfilled by the system.

4.1.1. Automated and customised system hardening profiles

The system should be able to automatically run all the hardening options mentioned above. The system should also allow a user to customize the hardening to be performed. To check this, we have defined four tests:

1. Automatic execution of all available security controls. We will perform an audit with *Lynis*, *Chef InSpec* and *OpenSCAP* tools and compare the score obtained before and after executing all security controls.
2. Execution of all tasks that belong to a single *CIS Control*. To test this test we will run all the benchmarks of a specific control and check if all of them have been run or if any of them are missing.
3. Use of tags to run all security controls related to a topic (e.g. *cron* or *sshd*)
4. Change default values using variables

4.1.2. Security operation customization according to the characteristics of the target machine

As we said in 3.2, The domain-specific language should make it possible to decide which security controls are applied on a machine based on its current state and characteristics. In this way, we can better model the experience of a trained system or security administrator according to these characteristics. To evaluate this, we have defined two different tests that need information from the target machine.

1. Change configuration based on target machine information (services, packages, open/closed ports, ...). To test this, we propose two possible real-life situations.
   (a) If the *apache* service is running and port 80 is in use, do not run 2.2.10. The *CIS Benchmark* 2.2.10 may affect the correct operation of the *apache* service.
   (b) If GUI is detected, do not run *CIS Benchmark* 2.2.2. Item 2.2.2 may affect the performance of the system's GUI.
   (c) If it is detected that *telnet* is being used and *SSH* is enabled, run the *CIS Benchmarks* 2.3.4 task which is responsible for disabling *telnet*.
2. Detect if the target system does not obtain a specific score through an audit of the *Lynis* tool.

4.1.3. Early configuration errors detection

The domain-specific language should detect possible configuration errors at compile time and detect any kind of problem with the deployment infrastructure, wrong variable values, and any other condition that may cause a run-time error. The difference with other tools such as *Ansible* lies in the fact that, while these tools detect errors once they have already occurred, with Egida we can obtain this information beforehand, during the compilation of the program. To evaluate this functionality, we have defined the following tests:

1. Syntactic and lexical error detection
2. Display warnings when the default value of a variable is used (its value has not been set and is going to be used)
3. Display Warnings when a *Level 2-Workstation* task is going to be executed

Audit Tool	After/Before	Successful	Failures	Other	Score [%]
Chef Inspec	Before	85	118	32	36.17
	After	151	53	31	64.25
OpenSCAP	Before	31	39	1	32.08
	After	52	18	1	71.46
Lynis	Before	78	72	0	52.00
	After	126	24	0	84.00

Table 1
Chef InSpec, OpenSCAP and Lynis results

4.2. Use Cases Evaluation Results

In this section, we will detail the results of the tests we have carried out on the use cases defined for our proposed system.

4.2.1. Automated and customised system hardening profiles

To test the first section of this use case, all security controls available in Egida (see 3.3.2) were run. We compared the results of audit tools before and after hardening with its initial configuration. The value of the score obtained in the different audit tools may vary depending on the initial configuration of the system. Installed services, defaults or other features may affect the score. For all the tests we have taken as the initial state an Operating System with the base installation configuration.

For the *Chef InSpec* tool we have used the *CIS Distribution Independent Linux Benchmark* [4] profile which implements the *CIS Distribution Independent Linux 2.0.0 Benchmark* and for the *OpenSCAP* audit we used the *CIS Benchmarks* profile for Ubuntu 18.04 LTS (*xccdf_org.ssgproject.content_profile_cis*). In the *Lynis* tool, we have taken the executed and not skipped tests and represented the warnings and suggestions as failures.

The table 1 shows the results obtained when running the security audit tools before and after applying the available security controls with Egida. As we can see, after applying Egida to the machine, we have managed to increase the hardening score by 28% in the *Chef InSpec* tool, 39.38% in *OpenSCAP* and 31% in *Lynis* audit tool. These results show how proper automated hardening of a machine can greatly increase its security compared to its initial configuration. This increase is even more noticeable if we consider that not all available security controls of the *CIS Benchmarks* have been implemented, so this value could be even higher.

To test the second section of this use case we have executed one by one all the *CIS Control* in Egida and checked if they correspond to the expected ones. Table 2 shows the results of the benchmarks expected and executed by each *CIS Controls* (only *CIS Controls* with some task in the *CIS Benchmarks* are shown). As can be seen in Table 2, all the expected security controls have been executed correctly, so Egida is able to successfully deploy all implemented security controls..

To test the last two sections of this use case we have developed a script in Aspida that performs all ssh related tasks and changes the value of the default variables. In Listing 2 we can see the Aspida code needed to run all the tasks related to *sshd* and configure the *SSH* port and user variables.

```
...
TASKS : {
  tags: ["sshd"];
}
VARS : {
  "user_ssh": "antonio";
  "sshd_access": {
    "ssh_port": 372
  };
}
```

Listing 2: Aspida Use Case 1

4.2.2. Security operation customization according to the characteristics of the target machine

To test this use case, we have developed a script in Aspida that, using conditionals, checks the state of the target machine and acts accordingly. As we can see in Listing 3, the language allows us to obtain information such as the current state of a service or the open ports.

In this script we are checking if the *Apache* service is running and port 80 is in use. If these two conditions are met, we add 2.2.10 to the list of tasks that should not be executed because it may affect the correct functioning of the service. In addition, if the target machine has an active GUI, we also add point 2.2.2 to the exclusions. In case the *telnet* package is installed and the *SSH* service is also running, we add point 2.3.4 to the execution list as it is in charge of disabling *telnet*. Finally, we check the score obtained on the target machine with the Lynis tool and we act on the basis of this score.

As shown in 3, making decisions based on the characteristics of the target machine gives us flexibility in the development of configuration scripts that allow us to specialise the hardening of a machine. This allows us to shield a system by applying all available security controls to it, except only those that interfere with its operation.

4.2.3. Early configuration errors detection

Finally, to test this use case we have developed some scripts that make use of undefined variables, have lexical or semantic errors, or try to execute *Level 2-Workstation* tasks.

In figures 4 and 5 we can see examples of the output produced by the compiler on lexical-sintactic errors. Figure

CIS Control v7	Expected benchmarks	Executed benchmarks
2.- Inventory and Control of Software Assets	8	8
3.-Continuous Vulnerability Management	6	6
4.-Controlled Use of Administrative Privileges	15	15
5.-Secure Configuration for Hardware and Software on Mobile Devices, Laptops, Workstations and Servers	52	52
6.-Maintenance, Monitoring and Analysis of Audit Logs	10	10
8.-Malware Defenses	3	3
9.-Limitation and Control of Network Ports, Protocols and Services	34	34
13.-Data Protection	1	1
14.-Controlled Access Based on the Need to Know	23	23
16.-Account Monitoring and Control	19	19
Total	171	171

Table 2
CIS Controls bechmakrs expected vs executed

6 shows an example of the warning produced when trying to execute a task that uses an undefined variable (its default value is used). An example of warnings produced by the compiler when trying to execute a Level 2 task can be seen in Figure 7.

```
...
TASKS : {
  IF "services.apache" == "RUNNING" {
    IF "machine.open_port" == 80 {
      exclusions: ["2.2.10"];
    }
  }
  IF "machine.gui" == true {
    exclusions: ["2.2.2"];
  }
  IF "packages.telnet" == "INSTALLED" {
    IF "services.ssh" == "RUNNING" {
        points: ["2.3.4"];
    }
  }
  IF "hardscores.lynis" <= 50 {
    controls: ["2.6", "3.4", "3.5",
        "4.3", "4.4", "4.5", "4.8",
        "4.9", "5.1", "5.5", "8.3",
        "9.4"];
  } ELIF "hardscores.lynis" <= 60 {
    controls: ["5.1", "5.5", "8.3",
        "9.4"];
  } ELSE {
    controls: ["9.4"];
  }
}
...
```

Listing 3: Aspida Use Case 2

```
SyntaxError on line 7 : 5 --> missing 'HOST' at ':'
SyntaxError on line 9 : 6 --> missing 'TASKS' at ':'
```

Figure 4: Compile time lexical errors

```
SyntaxError on line 13 : 7 --> mismatched input '{' expecting {'ELIF', 'ELSE'}
```

Figure 5: Compile time sintactic errors

```
WARNING: Variable  sshd_access not defined --> Its default value will be used
```

Figure 6: Compile time variables warnings

```
WARNING: Level 2 rule ( Ensure SSH AllowTcpForwarding is disabled ) is to be executed
```

Figure 7: Compile time level 2 tasks warnings

In Aspida, the use of variables not defined by default and the execution of Level 2 tasks are being marked as warnings instead of errors. This is done because errors stop the execution of the script. Instead, warnings allow the user to be notified of actions that may be dangerous in certain scenarios but do not stop the script.

Detecting errors at compile time (such as forgetting to change the value of the ports allowed by the firewall or setting the SSH access user) allows us, in addition to not wasting time waiting for the error to occur, to avoid possible default configurations that may be vulnerable, or to avoid running security controls that may affect the operation of the system.

5. Conclusions and Future Work

The proposed Egida system allows automated deployment of security configurations to one or more components of an infrastructure. The hardening of the target machine can be customised according to its characteristics in order to implement as many security controls as possible that do

not interfere with its correct operation. In this way, different security profiles are created that can be defined by a trained system or security administrator.

These security profiles can be defined with Egida's Domain Specific Language. The language compiler performs the checks of the statements that obtain information on the characteristics of the target machine, and allows changing the security controls to be applied by means of conditional statements.

Therefore, with Egida we complement the work of projects such as SCAP or the CIS Benchmarks by adding a layer of customisation flexibility, and error prevention based on the characteristics of each machine.

We intend to improve the Egida API in the future so we can obtain more information from the target machines. Some features that are being considered are checking the existence of certain files, checking the permissions on directories and files, or checking the configuration values of known services. The enhanced knowledge of the target systems will improve the ability to customize *CIS Benchmarks* application and improve the ability of translating expert administrator knowledge to them, based on its current configuration.

We also want to improve the Domain Specific Language by including elements to facilitate the developer's task. For example, the possibility of being able to print messages (error, info and warning) by console or the declaration and use of variables. All these future changes are aimed to fully deploy this security automation technique over the real infrastructures of a company, capturing the required information of their running systems.

This tool is also planned to be incorporated as part of the teaching materials of the Computer Security course of the School of Computer Engineering in the University of Oviedo [16]. All the source code of the Egida project and documentation of its use is available at https://egida-kassandra.github.io/egida.

References

[1] Al-Safwani, N., Fazea, Y., Ibrahim, H., 2018. Iscp: In-depth model for selecting critical security controls. Computers and Security 77, 565–577. doi:https://doi.org/10.1016/j.cose.2018.05.009.

[2] Berger, B., 2005. Trusted computing group history. Information Security Technical Report 10, 59–62. doi:https://doi.org/10.1016/j.istr.2005.05.007.

[3] Broderick, J.S., 2006. ISMS, security standards and security regulations. Information Security Technical Report 11, 26–31. doi:https://doi.org/10.1016/j.istr.2005.12.001.

[4] CIS, 2020a. CIS benchmarks. https://github.com/dev-sec/cis-dil-benchmark. Accessed: 2021-05-20.

[5] CIS, 2020b. CIS controls. https://www.cisecurity.org/controls/. Accessed: 2021-05-20.

[6] CIS, 2020c. Mapping and compliance. https://www.cisecurity.org/cybersecurity-tools/mapping-compliance. Accessed: 2021-05-20.

[7] Cleghorn, L., 2013. Network defense methodology: A comparison of defense in depth and defense in breadth. Journal of Information Security 4, 144–149. doi:10.4236/jis.2013.43017.

[8] Computer Security Division, I.T.L., 2020. Security content automation protocol: Csrc. https://csrc.nist.gov/projects/security-content-automation-protocol. Accessed: 2021-05-25.

[9] Cyberx-Mw, Knowlton, 2020. Security technical implementation guides (stigs). https://public.cyber.mil/stigs. Accessed: 2021-05-17.

[10] Durkota, K., Lisý, V., Bošanský, B., Kiekintveld, C., Pěchouček, M., 2019. Hardening networks against strategic attackers using attack graph games. Computers and Security 87, 101578. doi:https://doi.org/10.1016/j.cose.2019.101578.

[11] Kuipers, D., Fabro, M., 2006. Control systems cyber security: Defense in depth strategies. Technical Report. Idaho National Laboratory (INL).

[12] Mell, P., Scarfone, K., 2010. The common configuration scoring system (ccss): Metrics for software security configuration vulnerabilities. doi:https://doi.org/10.6028/NIST.IR.7502.

[13] Olenčin, M., Perháč, J., 2019. Automated configuration of a linux web server security, in: 2019 IEEE 15th International Scientific Conference on Informatics, pp. 000491–000496. doi:10.1109/Informatics47936.2019.9119272.

[14] OpenSCAP, 2020. Openscap portal. http://www.open-scap.org/. Accessed: 2021-05-20.

[15] R., W., Weaver, D., Farwood, D., 2013. Guide to Network Defense and Countermeasures. Cengage Learning.

[16] Redondo, J.M., 2021. Improving concept learning through specialized digital fanzines, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), pp. 134–143. doi:10.1109/ICSE-SEET52601.2021.00023.

[17] Redondo, J.M., Cuesta, D., 2019. Towards improving productivity in nmap security audits. Journal of Web Engineering 18, 539–578. doi:10.13052/jwe1540-9589.1871.

[18] Shamma, B., et al., 2018. Implementing CIS Critical Security Controls for Organizations on a Low-Budget. Ph.D. thesis. University of Houston.

[19] Software, I.J.S.., systems engineering, 2020. ISO/IEC 19770-8:2020. https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/25/72588.html. Accessed: 2021-05-12.

[20] Soto, J., 2019. Jshielder. https://github.com/Jsitech/JShielder. Accessed: 2021-05-26.

[21] Spichkova, M., Li, B., Porter, L., Mason, L., Lyu, Y., Weng, Y., 2020. Vm2: Automated security configuration and testing of virtual machine images. Procedia Computer Science 176, 3610–3617. doi:https://doi.org/10.1016/j.procs.2020.09.025.

[22] Stytz, M., 2004. Considering defense in depth for software applications. IEEE Security and Privacy 2, 72–75. doi:10.1109/MSECP.2004.1264860.

[23] Waltermire, D., Quinn, S., Booth, H., Scarfone, K., Prisaca, D., 2018. The technical specification for the security content automation protocol (scap) version 1.3. doi:https://doi.org/10.6028/NIST.SP.800-126r3.

[24] Winarno, H., Yasin, F., Prasetyo, M.A., Rohman, F., Shihab, M.R., Ranti, B., 2020. It infrastructure security risk assessment using the center for internet security critical security control framework: A case study at insurance company, in: 2020 3rd International Conference on Computer and Informatics Engineering (IC2IE), pp. 404–409. doi:10.1109/IC2IE50715.2020.9274594.

Antonio Payá González is currently finishing a Web Engineering Master Degree in the University of Oviedo, and working at ArcelorMittal R&D Asturias and ICUBE SL in the development of artificial intelligence solutions in the Additive Manufacturing field. He is also working on the implementation of secure network architectures for an additive manufacturing pilot plant and implementing new ways to automate the deployment of security configurations.

Alba Cotarelo is currently finishing a Web Engineering Master Degree in the University of Oviedo, and working with Machine Learning algorithms at Ingenica STS and ArcelorMittal R&D Asturias. She also worked in Artificial Intelligence at Model-Driven Engineering Research Group in Universidad de Ovieddo

Jose Manuel Redondo Lopez is an Associate Professor in the University of Oviedo, Spain. Received his B.Sc., M.Sc., and Ph.D. degrees in computer engineering from the same university in 2000, 2002, and 2007, respectively. He participated in various research projects funded by Microsoft Research and the Spanish Department of Science and Innovation. He has authored three books and over 20 articles. His research interests include computer security, dynamic languages and computational reflection.