



INGENIERÍA INFORMÁTICA

Memoria prácticas de empresa

Autor

Leon Elliott Fuller

Tutores

Pablo García Sánchez



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN
Granada, 2 de Junio de 2025

Índice general

1. Introducción	1
2. Descripción de la empresa	3
2.1. Actividad laboral	3
2.2. Personal cualificado	5
2.3. Dotación tecnológica	5
3. Trabajo Realizado	7
3.1. Modelo de Negocio y Soporte	7
3.2. Primera Tarea	8
3.3. Segunda Tarea	9
3.4. Tercera Tarea	10
3.5. Cuarta Tarea	12
3.6. Asignaturas Relacionadas	13
4. Valoración Personal	15
4.1. Valoración de la Experiencia	15
5. Conclusiones	17

Capítulo 1

Introducción

La realización de prácticas profesionales en una empresa representa una experiencia esencial para la formación académica del alumno, ya que facilita la puesta en práctica de los saberes adquiridos durante la carrera en un contexto real y contribuye al desarrollo de destrezas y competencias valiosas.

El propósito de esta memoria es mostrar con detalle las tareas ejecutadas, los retos afrontados y las soluciones puestas en marcha a lo largo del periodo de prácticas en Wazuh, compañía referente en soluciones de seguridad de código abierto fundada en 2015 con sede en Campbell, California [1]. Las prácticas se desarrollaron entre marzo y mayo de 2025, trabajando principalmente desde la oficina de Granada y en remoto.

Durante este periodo, se expondrán de manera pormenorizada las tecnologías utilizadas (Kubernetes, AWS, Docker, scripting en Bash), los obstáculos encontrados (problemas de despliegue, configuraciones TLS, firewall interno, etc.) y las estrategias adoptadas, así como su vinculación con las asignaturas relevantes del plan de estudios.

A lo largo del documento, la memoria se estructura de la siguiente manera:

- **Descripción de la empresa:** se contextualiza el entorno de Wazuh, su actividad principal, el equipo humano y la dotación tecnológica.
- **Trabajo realizado:** se detallan las tareas llevadas a cabo, los problemas técnicos planteados, las herramientas y soluciones implementadas, indicando qué asignaturas sustentan cada técnica.
- **Valoración personal:** se reflexiona sobre la experiencia adquirida, las competencias desarrolladas y el impacto profesional de este periodo.

Capítulo 2

Descripción de la empresa

La empresa donde he realizado las prácticas se llama **Wazuh, Inc.**, constituida como Delaware corporation con EIN 47-2523953 y su sede central situada en 1999 S. Bascom Ave, Suite 700 PMB#727, Campbell, CA 95008, Estados Unidos [2].

Wazuh fue fundada en 2015 por Santiago Basset como una bifurcación del proyecto OSSEC, consolidándose rápidamente como una de las plataformas de seguridad de código abierto más relevantes del mercado [3].

Según la propia compañía, Wazuh protege más de 15 millones de endpoints y da servicio a más de 100 000 usuarios empresariales[4].

2.1. Actividad laboral

Wazuh ofrece una plataforma unificada de XDR y SIEM para la protección de endpoints y cargas de trabajo en la nube. Sus funciones principales incluyen la gestión de logs, la monitorización de integridad de ficheros y la detección de vulnerabilidades [4][1]. Adicionalmente, Wazuh proporciona módulos de cumplimiento normativo para estándares como PCI DSS e HIPAA [5]. La plataforma soporta integraciones nativas con entornos Docker, Kubernetes y Amazon Web Services (AWS) para garantizar una monitorización continua y centralizada de infraestructuras[6].

Su agente multiplataforma soporta sistemas operativos como Linux, Windows, macOS, Solaris, AIX y HP-UX[6], e incorpora integraciones nativas con contenedores Docker, kubernetes y plataformas en la nube para garantizar el cumplimiento de normativas como PCI DSS o HIPAA [5].

Wazuh se basa en el agente Wazuh, que se implementa en los endpoints monitorizados, y en tres componentes centrales: el servidor Wazuh, el indexador Wazuh y el panel de control Wazuh.

- **El indexador Wazuh:** Es un motor de búsqueda de texto completo y análisis altamente escalable. Este componente central indexa y almacena las alertas generadas por el servidor Wazuh.

- **El servidor Wazuh:** Analiza los datos recibidos de los agentes. Los procesa mediante decodificadores y reglas, empleando inteligencia de amenazas para buscar indicadores de compromiso conocidos (IOCs). Un único servidor puede analizar datos de cientos o miles de agentes y escalar horizontalmente cuando se configura en un clúster. Este componente central también se utiliza para gestionar los agentes, configurándolos y actualizándolos de forma remota cuando es necesario.
- **El dashboard Wazuh:** Es la interfaz web de usuario para la visualización y el análisis de datos. Incluye paneles prediseñados para la búsqueda de amenazas, el cumplimiento normativo (por ejemplo, PCI DSS, RGPD, CIS, HIPAA, NIST 800-53), aplicaciones vulnerables detectadas, datos de monitorización de integridad de archivos, resultados de evaluación de configuración, eventos de monitorización de infraestructura cloud, entre otros. También se utiliza para gestionar la configuración de Wazuh y para monitorizar su estado.

Además de las capacidades de monitorización basadas en agentes, la plataforma Wazuh puede monitorizar dispositivos sin agente, como firewalls, switches, routers o IDS de red, entre otros. Por ejemplo, los datos de registros del sistema pueden recopilarse a través de Syslog, y su configuración puede vigilarse mediante sondeos periódicos de sus datos, por SSH o mediante una API.

El diagrama siguiente representa los componentes de Wazuh y el flujo de datos.

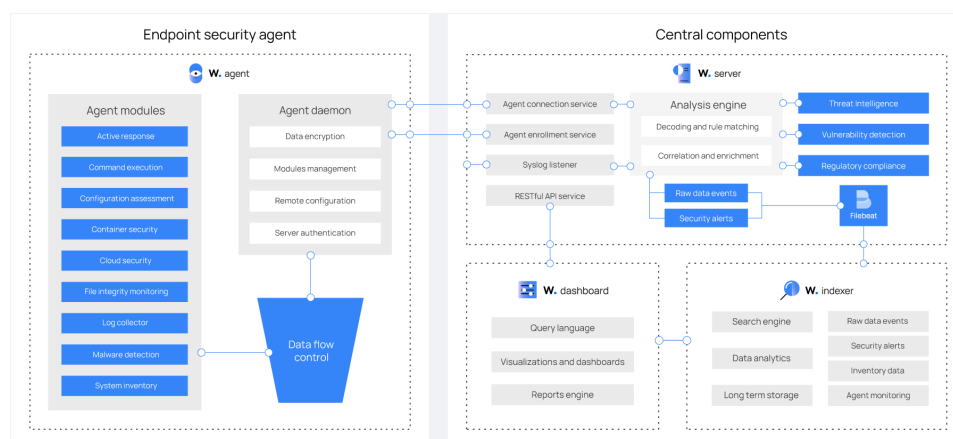


Figura 2.1: Componentes de Wazuh

2.2. Personal cualificado

La plantilla de Wazuh está entre los 200 y 500 empleados a nivel mundial, entre ingenieros de software, ingenieros de seguridad (soporte técnico), ingenieros en Cloud, ingenieros de QA, desarrolladores de C/C++, diseñadores UX/UI, especialistas en DevOps y expertos en ciberseguridad [7].

2.3. Dotación tecnológica

Mi puesto consistía de un trabajo remoto, por tanto, no he recibido ninguna dotación tecnológica y he usado mi portátil para el trabajo. Se dispone de una oficina en Granada donde los empleados disponen de portátiles proporcionados por la propia empresa. Cabe destacar, que tienen clústeres de cómputo en la nube mediante servicios de AWS para poder realizar pruebas de entornos en la nube. La infraestructura de Wazuh se puede desplegar completamente en Amazon Web Services (AWS), haciendo uso de instancias EC2, almacenamiento en S3 y servicios de monitorización como CloudWatch para procesar y analizar datos de seguridad en tiempo real [6]. AWS es una plataforma de computación en la nube que ofrece más de 200 servicios que incluyen potencia de cómputo, opciones de almacenamiento, capacidad de red, etc.

Para la asignación de tareas se utiliza Jira. Esta plataforma de gestión de proyectos y seguimiento de actividades permite a los equipos planificar, monitorizar y administrar el desarrollo de software de manera eficiente.

Factorial es el software de gestión de recursos humanos que utiliza la empresa y se encarga de la organización del equipo, el horario del equipo donde cada miembro de Wazuh incluye su zona horaria y su turno en la pestaña de empleado. También cabe destacar que se usa para controlar las vacaciones, ausencias y todo lo relacionado a los derechos de cada trabajador.

Por último, para la comunicación y organización se usa la plataforma Slack, que es una plataforma parecida a Discord, Microsoft Teams, etc. La idea es que los empleados se puedan comunicar entre sí, además de ir anotando en tiempo real el trabajo que se está realizando. Esto último sirve a modo de seguimiento, donde cada empleado al final de su jornada tendrá que poner un resumen del mismo.

Capítulo 3

Trabajo Realizado

Dentro de Wazuh, Inc. existen varios departamentos, tales como Cloud, Customer Success, DevOps, Desarrollo en C/C++, *Operations*, entre otros. Mi puesto se ubica en el departamento de *Operations*, ejerciendo como *Security Engineer*. El objetivo primordial de este equipo es brindar soporte técnico tanto a clientes como a otros equipos internos, actuando como enlace entre las necesidades de los clientes y las capacidades del producto.

3.1. Modelo de Negocio y Soporte

Wazuh es una plataforma de seguridad de código abierto y libre, cuyo producto principal puede descargarse y utilizarse sin coste. La principal fuente de ingresos proviene de los contratos de soporte profesional, mediante los cuales las organizaciones pagan por recibir asistencia técnica especializada.

- **Términos de Soporte Profesional:** Estos contratos especifican horarios de cobertura, tiempos de respuesta y prioridad de atención.
- **Portal de Soporte:** Los clientes utilizan el portal web de Wazuh para abrir *tickets* de soporte, gestionados por el equipo de *Operations*.
- **Herramientas de Ticketing:** Para la asignación y seguimiento de incidencias, empleamos Jira, sistema que permite planificar, rastrear y gestionar cada solicitud de soporte.

Cuando un cliente adquiere un contrato de soporte, recibe credenciales para el portal de soporte (SLA **Premium**, **Enterprise**, etc.). Cada incidencia o petición de mejora se registra como un *issue* en Jira, detallando:

1. Descripción del problema o requerimiento.
2. Prioridad y nivel de impacto para la infraestructura del cliente.

3. Información del entorno: versión de Wazuh, sistema operativo, configuración de agentes, etc.

Una vez creado el *ticket*, el jefe del equipo correspondiente de *Operations* asigna el caso a un *Security Engineer*, quien procede a realizar un diagnóstico inicial, reproduce el entorno si es necesario y propone una solución. Cabe destacar que existen 6 equipos, divididos esencialmente por zona horaria. Yo pertenezco al equipo 1, que se resume en zona horaria CET.

Dicho esto, las tareas que he realizado son bastante amplias, aunque todo viene relacionado a aumentar y vigilar la seguridad de los sistemas. También me gustaría destacar que las dos/tres primeras semanas fueron exclusivamente de training, para poder entender la complejidad del entorno, además de realizar varios deployments para poder entender mejor los sistemas con los que se trabaja.

3.2. Primera Tarea

En esta tarea se me asignó el desarrollo de un *script* en Bash destinado a realizar un diagnóstico exhaustivo de un entorno completo de Wazuh (Indexador, Servidor y Dashboard). El propósito principal era disponer de un baremo rápido que permitiera decidir si el sistema estaba preparado para una actualización importante, comprobando aspectos clave a través de llamadas a las API de Wazuh.

El *Wazuh Diagnosis Script* recopila información tanto del *Wazuh Manager* como del *Wazuh Indexer*, y evalúa varios parámetros críticos de salud:

- **Versión de Wazuh:** se verifica que la versión sea $\geq 4.5.0$, para garantizar compatibilidad con los nuevos módulos y correcciones de seguridad.
- **Uso de disco en /:** debe mantenerse por debajo del 85 % para evitar fallos durante el proceso de actualización.
- **Uso de disco en nodos indexadores:** cada nodo de Elasticsearch/OpenSearch se comprueba de forma individual para asegurar que ningún volumen se acerque al límite crítico (85 %).
- **Salud del clúster indexador:** se invoca la API de *cluster health* y se exige que el estado sea "green" (sin shards pendiente de reasignación ni índices en estado "amarillo" o rojo) [4]. Los shards en Elasticsearch son las unidades básicas de almacenamiento y distribución de datos dentro de un índice, lo que permite una gestión eficiente de los datos y capacidades de búsqueda. Cada índice puede dividirse en varios shards, que pueden distribuirse en diferentes nodos de un clúster para mejorar el rendimiento y la escalabilidad.

- **Disponibilidad de las APIs:** se realizan peticiones HTTP 200 a los endpoints principales del *Manager* e *Indexer* para confirmar que los servicios estén operativos.

Si alguna de estas comprobaciones falla, el entorno se marca como no apto para actualización, y el *script* detalla las métricas que requieren atención inmediata.

El desarrollo de este *script* requirió dominar diversas habilidades técnicas. Por un lado, fue necesario realizar una programación avanzada en Bash combinada con el manejo de JSON, empleando herramientas como **bash**, **curl**, **jq**, **grep** y **awk** para procesar tanto respuestas en formato JSON como texto plano. Además, implicó una comprensión profunda de las APIs REST de Wazuh, conociendo a fondo los endpoints del *Manager* y del *Indexer* y gestionando correctamente la autenticación mediante tokens JWT. Paralelamente, se aplicaron competencias de administración de sistemas Linux: recoger métricas de uso de disco, CPU y memoria, gestionar permisos y supervisar procesos a través de **systemctl**, **lsof**, **df** y **du**.

Al mismo tiempo, fue fundamental estructurar de forma coherente los resultados, diseñando un sistema de carpetas ordenado, generando logs detallados y creando archivos de salida estandarizados para facilitar la lectura y el análisis posterior. Finalmente, se implementaron mecanismos de gestión de excepciones y tolerancia a fallos, incorporando reintentos automáticos ante fallos en las llamadas API, captura de errores en cada paso del script y validaciones periódicas (por ejemplo, comprobaciones de códigos de estado HTTP o de valores numéricos de uso de disco) para garantizar que la ejecución continuara de manera robusta incluso si surgían problemas puntuales.

3.3. Segunda Tarea

En esta tarea se me encomendó configurar **Postfix** como servidor SMTP para el envío de notificaciones por correo electrónico desde el entorno de Wazuh. El objetivo era garantizar que, ante cualquier evento crítico o alerta generada por Wazuh, el sistema pudiera remitir automáticamente un mensaje a los destinatarios definidos. El propósito principal consistía en:

- Instalar y parametrizar Postfix en el servidor de Wazuh para que actuara como agente de envío SMTP.
- Definir el *relayhost*, las rutas de correo y las opciones de autenticación (en caso de usar un servidor externo).
- Verificar, mediante pruebas de conectividad y captura de tráfico, que los paquetes TCP relativos al protocolo SMTP (puerto 25 o 587) efectuasen correctamente el *handshake* (SYN / SYN-ACK / ACK) con el servidor de correo de destino.

Al principio, los intentos de `telnet smtp.office365.com 587` se quedaban colgados. Se comprobó que las reglas de *iptables* locales bloqueaban la conexión saliente. Se añadió la regla:

```
iptables -A OUTPUT -p tcp --dport 587 -j ACCEPT
```

y se guardaron los cambios con `iptables-persistent`.

Aún así seguía fallando sin razón ninguna (los logs de postfix no mostraban errores significantes, solamente se veía que no se estaba completando correctamente la conexión al servidor smtp de office365). Para descartar problemas del propio Postfix, realizamos varias pruebas de red que mostraron un comportamiento inesperado: tras completar el *three-way TCP handshake*, el servidor remoto enviaba un paquete `RST` y la sesión se cortaba antes de iniciar el protocolo SMTP. Esto apuntaba a que un dispositivo de seguridad (firewall) estaba restableciendo la conexión de forma silenciosa. Los comandos utilizados fueron:

```
sudo tcpdump -i any host smtp.office365.com and port 587 -w /tmp/587.pcap
# En otra terminal, se provoca el fallo:
telnet smtp.office365.com 587
# Tras detener tcpdump (Ctrl-C), se inspecciona el pcap:
tcpdump -nn -r /tmp/587.pcap -A
```

Al analizar la captura, se observó claramente el intercambio:

- SYN enviado desde el servidor Wazuh hacia `smtp.office365.com:587`.
- SYN{ACK de respuesta del servidor remoto.
- RST enviado inmediatamente por el firewall de la empresa, interrumpiendo la conexión antes de que se iniciase el diálogo SMTP.

De este modo, se confirmó que el *Palo Alto* (firewall corporativo) estaba restableciendo (reset) la sesión TCP. En colaboración con el equipo de redes, solicitamos que se abriera el puerto 587 en el firewall para permitir el tráfico SMTP, tras lo cual las pruebas con `telnet` y `tcpdump` mostraron que el *handshake* completo (SYN / SYN-ACK / ACK) se realizaba correctamente sin recibir `RST`.

3.4. Tercera Tarea

En esta tarea se desarrolló un **script** en Bash encargado de generar y enviar periódicamente un reporte en formato HTML (con posibilidad de convertirlo a PDF) sobre el estado de los agentes de Wazuh. El objetivo era automatizar mediante `cron` la creación diaria de un informe que incluyese,

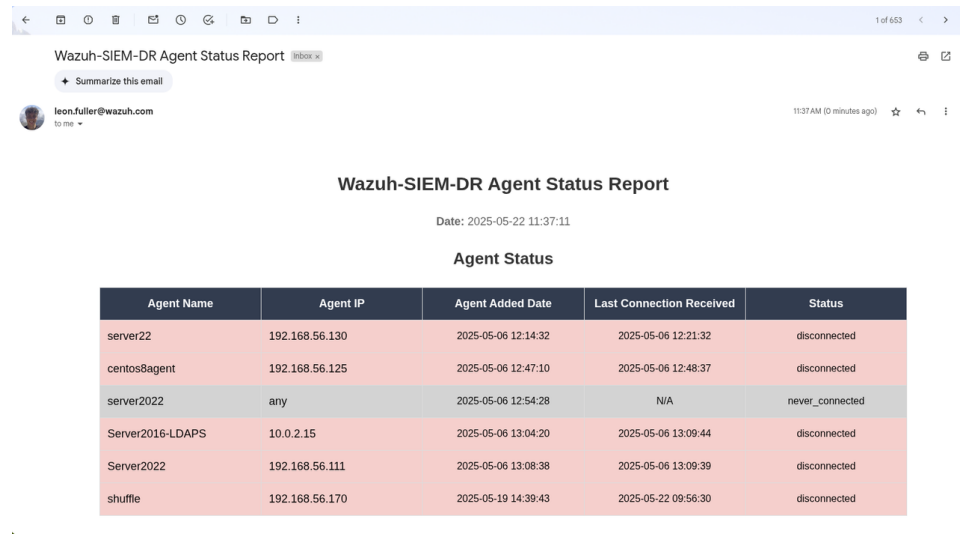
para cada agente, su nombre, IP, fecha de alta, último *keep-alive* y estado (*disconnected*, *pending*, *never_connected*, *active*). El flujo general y las dificultades técnicas fueron:

La creación del informe diario comienza con la autenticación contra la API de Wazuh: mediante una llamada `curl -u usuario:clave` al endpoint `/security/user/authenticate` obtenemos un token JWT que nos permite, a continuación, invocar el recurso `/agents?status=disconnected,pending,never_connected` para recuperar todos los agentes que no están en estado *activo*. Esta respuesta en JSON se procesa con `jq` (combinado con codificación en base64 para iterar cada registro en Bash) y de cada agente se extraen campos como nombre, IP, fecha de alta (`dateAdd`), última conexión (`lastKeepAlive`) y su estado, que luego convertimos a una fecha legible (`YYYY-MM-DD HH:MM:SS`) usando `date -d`.

Con esa información, construimos dinámicamente una única variable (`HTML_BODY`) que contiene un fragmento HTML con estilos CSS embebidos: cabeceras `<head>` con tipografía, colores y sombreado de filas; cada fila del `<table>` se colorea en función del estado del agente (por ejemplo, rojo para *disconnected*, amarillo para *pending*, verde para *active*). Una vez montado todo el contenido HTML, se encapsulan las cabeceras `Subject`, `Content-Type: text/html` y `From/To` en un `echo -e` que se envía a `sendmail -t`, de modo que el servidor Linux (con un MTA configurado) procesa este bloque HTML como cuerpo del mensaje y lo envía automáticamente. Finalmente, se configura un `cron` para que esta secuencia se ejecute cada mañana (por ejemplo, `0 6 * * * /ruta/diagnosis.sh`), redirigiendo la salida estándar y los posibles errores a un archivo de log rotativo en `/var/log/wazuh_agent_report.log` para auditoría y diagnóstico posterior.

Las principales dificultades surgieron al tratar con JSON en Bash, donde el uso combinado de `'jq'` y `'base64'` dentro de bucles `'for'` complica notablemente la sintaxis y hace que cualquier cambio en la estructura de la respuesta pueda romper el script. Además, construir un HTML completo embebido en variables de Bash obligó a escribir concatenaciones muy cuidadosas para evitar errores de escapado de comillas y caracteres especiales, lo que a menudo requería comprobaciones manuales del resultado generado. Otro reto fue convertir las marcas de tiempo ISO 8601 que devuelve la API a un formato legible localmente mediante `'date -d'`, ya que cada distribución de Linux interpreta las zonas horarias de forma ligeramente distinta.

A continuación se muestra un ejemplo del aspecto visual del informe recibido por correo. Los agentes fuera de servicio aparecen en rojo, los pendientes en amarillo y los activos en verde:



The screenshot shows an email interface with the subject 'Wazuh-SIEM-DR Agent Status Report'. The email body contains a table titled 'Agent Status' with the following data:

Agent Name	Agent IP	Agent Added Date	Last Connection Received	Status
server22	192.168.56.130	2025-05-06 12:14:32	2025-05-06 12:21:32	disconnected
centos8agent	192.168.56.125	2025-05-06 12:47:10	2025-05-06 12:48:37	disconnected
server2022	any	2025-05-06 12:54:28	N/A	never_connected
Server2016-LDAPS	10.0.2.15	2025-05-06 13:04:20	2025-05-06 13:09:44	disconnected
Server2022	192.168.56.111	2025-05-06 13:08:38	2025-05-06 13:09:39	disconnected
shuffle	192.168.56.170	2025-05-19 14:39:43	2025-05-22 09:56:30	disconnected

Figura 3.1: Ejemplo de reporte diario de estado de agentes (formato HTML renderizado).

Este enfoque basado en Bash garantiza alta difusión y compatibilidad en múltiples sistemas Linux, evitando la complejidad y las dependencias de un entorno Python, y facilita la obtención de estadísticas diarias sin intervención manual.

3.5. Cuarta Tarea

En esta última tarea se abordó el despliegue de un clúster de Wazuh Manager en *Amazon EKS* (Elastic Kubernetes Service), combinando tecnologías de AWS y Kubernetes para garantizar escalabilidad, tolerancia a fallos y comunicaciones seguras[8].

El primer reto consistió en provisionar volúmenes persistentes (PV/PVC) mediante **StorageClasses** basadas en **gp3** de Amazon EBS[9][10]. Inicialmente, los pods de Wazuh Manager no podían montar los volúmenes debido a permisos de acceso y restricciones de **ReadWriteOnce**[11]. Para solucionarlo, creamos un **PersistentVolumeClaim** de 100 GiB y configuramos correctamente el **Provisioner** de EBS con el CSI Driver (**ebs.csi.eks.amazonaws.com**)[10]. A continuación, se definió un **Deployment** de Docker que ejecutaba la imagen oficial de Wazuh Manager, montando el PVC en **/var/lib/elasticsearch** y en **/var/ossec/data** para garantizar la persistencia de índices y configuraciones[11].

El segundo desafío fue exponer los servicios mediante un **Service** de tipo **LoadBalancer**[12]. Al comienzo, el tráfico entrante no alcanzaba los pods por errores en las políticas de **Security Group** de AWS[13] y la falta de un **Ingress Controller**[14]. Ajustamos las reglas del **Security Group** para permi-

tir puertos TCP 1514 (comunicación con agentes) y 55000 (API/WebUI), y desplegamos un **Ingress Controller** de NGINX con **cert-manager** para gestionar automáticamente certificados TLS vía Let's Encrypt[14][15]. Así, las peticiones HTTPS al panel de Wazuh redirigían correctamente al servicio interno.

La tercera complejidad implicó habilitar TLS entre agentes y Manager[16]. Generamos una Autoridad de Certificación interna con **openssl**, creamos certificados de servidor que incluyeran el DNS del **LoadBalancer** y el nombre del servicio de Kubernetes, y almacenamos estos certificados en un **Secret** de Kubernetes[17]. Modificamos el **ConfigMap** de Wazuh Manager para referenciar el **Secret**, añadiendo en **ossec.conf** las rutas a **server_ca_path**, **agent_certificate_path** y **agent_key_path** para asegurar la validación de certificados por parte de los agentes[18].

Esta tarea integró a fondo conocimientos de Kubernetes (volúmenes, servicios, secretos, tolerancia a fallos)[19], AWS (EBS, EKS, Security Groups, Load Balancers)[13][12], Docker (construcción de imágenes y contenedores) y seguridad (TLS/autenticación)[16], evidenciando la complejidad de orquestar un entorno Wazuh distribuido en la nube y garantizando alto nivel de disponibilidad, escalabilidad y protección de datos.

3.6. Asignaturas Relacionadas

Para la ejecución de las tareas anteriores, resultaron fundamentales los conocimientos de las siguientes asignaturas del plan de estudios:

- **Sistemas Operativos:** Configuración avanzada de Linux y Windows, administración de servicios, scripting en Bash y PowerShell.
- **Ingeniería de Servidores:** Fundamentos de criptografía, detección de intrusiones, gestión de incidentes y aplicación de estándares de seguridad (PCI DSS, HIPAA).
- **Fundamentos de Redes:** Protocolos TCP/IP, configuración de firewalls, conceptos de IDS/IPS y análisis de tráfico con Wireshark/Suricata.
- **Fundamentos de Bases de Datos:** Ingesta y consulta de grandes volúmenes de datos (alertas) en Elasticsearch/OpenSearch, optimización de índices y consultas.
- **Ingeniería de Software:** Diseño de arquitecturas distribuidas, uso de herramientas de control de versiones (Git) y metodologías ágiles en proyectos de DevOps.

Capítulo 4

Valoración Personal

4.1. Valoración de la Experiencia

Durante el periodo de prácticas en Wazuh, pude adquirir competencias prácticas en despliegue y administración de soluciones SIEM/XDR, así como en el desarrollo e implementación de reglas de detección basadas en estándares de ciberseguridad. La exposición continua a entornos empresariales reales y la colaboración con profesionales de distintas áreas fortalecieron mi capacidad para trabajar bajo presión y resolver problemas complejos de manera eficiente.

He de destacar que entré en un equipo donde no había un jefe, es decir, en cada equipo de zona horaria hay un jefe del mismo, que se encarga de organizar, vigilar y ayudar a sus subordinados. Yo tuve que reportar directamente al jefe de todo el equipo de Operaciones. Esto hizo que mi recorrido sea más difícil, no obstante, tuve la ayuda necesaria y pude completar todos los obstáculos y retos que me encontré por delante.

Capítulo 5

Conclusiones

Durante estos tres meses de prácticas en Wazuh he tenido la oportunidad de colaborar con equipos multidisciplinares y entender en profundidad la organización y el funcionamiento de una empresa tecnológica de gran envergadura. Este periodo me ha permitido adquirir una visión amplia tanto del ámbito corporativo como del técnico, y experimentar de primera mano cómo se gestionan proyectos complejos en un entorno real. He aprendido a trabajar con herramientas y tecnologías punteras en ciberseguridad, tales como Kubernetes (EKS y entornos autogestionados), Terraform y Ansible para la provisión y configuración de infraestructura en AWS y OpenSearch/Elasticsearch para la indexación y análisis de logs. Asimismo, reforcé mis conocimientos sobre el despliegue de Wazuh Manager y Wazuh Agent en arquitecturas de alta disponibilidad, manejo de volúmenes persistentes y certificados TLS en entornos distribuidos. Gracias al soporte continuo de la comunidad open source de Wazuh (Reddit, Discord, Slack, GitHub), pude consultar y resolver rápidamente dudas, lo que facilitó enormemente la integración de buenas prácticas y soluciones ya contrastadas.

Por otro lado, he mejorado significativamente mi comprensión de los conceptos de redes y protocolos: aprendí a diagnosticar problemas con `tcpdump`, interpretar el intercambio de paquetes TCP (SYN, SYN-ACK, RST), configurar reglas de `iptables` para permitir tráfico SMTP y depurar conexiones SMTP/TLS con Postfix. Asimismo, profundicé en el uso de APIs REST de Wazuh, autenticación con tokens JWT, scripting avanzado en Bash (manipulación de JSON con `jq`, construcción dinámica de HTML/CSS, manejo de fechas y registros), y configuración de `cron` para automatizar tareas diarias. Esta experiencia global me ha dotado de un conjunto de competencias técnicas sólidas en administración de sistemas Linux, orquestación de contenedores, herramientas de automatización, gestión de seguridad en la nube y metodologías de diagnóstico y monitorización, que sin duda fortalecerán mi perfil profesional y me preparan para afrontar nuevos retos en el ámbito de la ciberseguridad y la ingeniería de infraestructura.

Bibliografía

- [1] Wazuh. About us. <https://wazuh.com/about-us/>, 2025. Accessed: 2025-05-28.
- [2] Wazuh. Support agreement. https://wazuh.com/docs/Wazuh_Support_agreement.pdf, 2022. Accessed: 2025-05-28.
- [3] Wikipedia contributors. Wazuh. <https://es.wikipedia.org/wiki/Wazuh>, 2025. Accessed: 2025-05-28.
- [4] Wazuh. Wazuh — open source xdr. open source siem. <https://wazuh.com/>, 2025. Accessed: 2025-05-28.
- [5] Wazuh Documentation. Regulatory compliance. <https://documentation.wazuh.com/current/compliance/index.html>, 2025. Accessed: 2025-05-28.
- [6] Wazuh Documentation. Wazuh agent — installation guide. <https://documentation.wazuh.com/current/installation-guide/wazuh-agent/index.html>, 2025. Accessed: 2025-05-28.
- [7] LinkedIn. Wazuh — linkedin. <https://www.linkedin.com/company/wazuh>, 2025. Accessed: 2025-05-28.
- [8] Wazuh Team. Deploying wazuh on kubernetes using aws eks, 2025. URL <https://wazuh.com/blog/deploying-wazuh-on-kubernetes-using-aws-eks/>. Accedido mayo 2025.
- [9] Amazon Web Services. Create a storage class, 2023. URL <https://docs.aws.amazon.com/eks/latest/userguide/create-storage-class.html>. Accedido mayo 2025.
- [10] AWS Containers Blog. Migrating amazon eks clusters from gp2 to gp3 ebs volumes, 2021. URL <https://aws.amazon.com/blogs/containers/migrating-amazon-eks-clusters-from-gp2-to-gp3-ebs-volumes/>. Accedido mayo 2025.

- [11] Wazuh Team. Deployment on kubernetes, 2025. URL <https://documentation.wazuh.com/current/deployment-options/deploying-with-kubernetes/kubernetes-deployment.html>. Accedido mayo 2025.
- [12] Amazon Web Services. Load balancing - amazon eks, 2025. URL <https://docs.aws.amazon.com/eks/latest/best-practices/load-balancing.html>. Accedido mayo 2025.
- [13] Amazon Web Services. View amazon eks security group requirements for clusters, 2025. URL <https://docs.aws.amazon.com/eks/latest/userguide/sec-group-reqs.html>. Accedido mayo 2025.
- [14] cert-manager Authors. Securing nginx-ingress, 2025. URL <https://cert-manager.io/docs/tutorials/acme/nginx-ingress/>. Accedido mayo 2025.
- [15] Wazuh Team. Configuring third-party ssl certificates, 2025. URL <https://documentation.wazuh.com/current/user-manual/wazuh-dashboard/configuring-third-party-certs/index.html>. Accedido mayo 2025.
- [16] Wazuh Team. Wazuh agent identity verification, 2025. URL <https://documentation.wazuh.com/current/user-manual/agent/agent-enrollment/security-options/agent-identity-verification.html>. Accedido mayo 2025.
- [17] Kubernetes Documentation. Secrets, 2025. URL <https://kubernetes.io/docs/concepts/configuration/secret/>. Accedido mayo 2025.
- [18] Wazuh Team. client - local configuration (ossec.conf), 2025. URL <https://documentation.wazuh.com/current/user-manual/reference/ossec-conf/client.html>. Accedido mayo 2025.
- [19] Wazuh Contributors. wazuh-kubernetes, 2024. URL <https://github.com/wazuh/wazuh-kubernetes>. Accedido mayo 2025.