ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

IMPLEMENTACIÓN DE SERVICIOS DE SEGURIDAD EN REDES MEDIANTE DEVOPS

IMPLEMENTACIÓN DE UN HIDS MEDIANTE HERRAMIENTAS DE DEVOPS

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN REDES Y TELECOMUNICACIONES

JEREMY LEONEL CAMPOVERDE ARMIJOS

DIRECTOR: FERNADO VINICIO BECERRA CAMACHO

DMQ, Agosto 2024

CERTIFICACIONES

Yo, Jeremy Leonel Campoverde Armijos declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Jeremy Campoverde

jeremy.campoverde@epn.edu.ec.com

jeremycampoverde1@gmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por JEREMY LEONEL CAMPOVDERDE ARMIJOS, bajo mi supervisión.

FERNANDO VINICIO BECERRA CAMACHO

Fernando.becerra@epn.edu.ec

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Jeremy Leonel Campoverde Armijos.

ÍNDICE DE CONTENIDOS

RESUMEN

El proyecto se centra en la implementación de servicios de seguridad de red utilizando DevOps. El objetivo general es desplegar soluciones de seguridad informática integrando herramientas DevOps. Los objetivos específicos incluyen el análisis de las herramientas DevOps, el diseño de soluciones de seguridad de red, la implementación de estas soluciones y la verificación de su rendimiento. Este enfoque garantiza un desarrollo estructurado alineado con los principios de DevOps.

El alcance del proyecto abarca varios aspectos clave. En primer lugar, el despliegue de un sistema de detección de intrusiones basado en host (HIDS) se automatiza utilizando herramientas DevOps como Ansible. Además, se despliega e integra un HIDS seleccionado (Tripwire) utilizando playbooks. Se realizan pruebas para verificar la correcta implementación del HIDS y se genera documentación detallada que describe todo el proceso.

El marco teórico aborda los conceptos fundamentales de HIDS y DevOps. Un HIDS es un sistema de seguridad que monitoriza y analiza el comportamiento de un host para detectar actividades maliciosas. DevOps, por su parte, es una metodología que promueve la colaboración de los equipos de desarrollo y operaciones con el propósito de poder automatizar procesos y mejorar la eficiencia. La integración de estas dos tecnologías permite crear soluciones de seguridad automatizadas.

Para poder realizar implementación, se utilizó Ansible para desplegar Tripwire a través de dos *playbooks*. El primero se encargó de la instalación automatizada de Tripwire en los sistemas de destino, garantizando una configuración repetible. El segundo playbook se desarrolló para aplicar nuevas reglas al documento de políticas de Tripwire, permitiendo la supervisión de archivos críticos adicionales.

Una vez diseñados los *playbooks*, se realizaron pruebas iniciales para validar su funcionamiento. Estas pruebas eran esenciales para garantizar que las reglas añadidas se hayan establecido de forma correcta. La fase de pruebas incluyó la comprobación de la instalación correcta de Tripwire y la aplicación de las nuevas reglas de monitorización. Los resultados de estas pruebas demostraron que se habían alcanzado con éxito los objetivos del proyecto.

PALABRAS CLAVE: IDS basado en host, Automatización, Despliegue,Seguridad informática, Seguridad de red

ABSTRACT

The project focuses on the implementation of network security services using DevOps. The overall objective is to deploy IT security solutions integrating DevOps tools. The specific objectives include the analysis of DevOps tools, the design of network security solutions, the implementation of these solutions and the verification of their performance. This approach ensures a structured development aligned with DevOps principles.

The scope of the project covers several key aspects. First, the deployment of a host-based intrusion detection system (HIDS) is automated using DevOps tools such as Ansible. In addition, a selected HIDS (Tripwire) is deployed and integrated using playbooks. Tests are performed to verify the correct implementation of the HIDS and detailed documentation describing the entire process is generated.

The theoretical framework addresses the fundamental concepts of HIDS and DevOps. A HIDS is a security system that monitors and analyzes the behavior of a host to detect malicious activity. DevOps, on the other hand, is a methodology that promotes the collaboration of development and operations teams in order to automate processes and improve efficiency. The integration of these two technologies makes it possible to create automated security solutions.

In order to implement, Ansible was used to deploy Tripwire through two playbooks. The first playbook handled the automated installation of Tripwire on the target systems, ensuring repeatable configuration. The second playbook was developed to apply new rules to the Tripwire policy document, allowing additional critical files to be monitored.

Once the playbooks were designed, initial testing was performed to validate their operation. These tests were essential to ensure that the added rules were set up correctly. The testing phase included verifying the correct installation of Tripwire and the application of the new monitoring rules. The results of these tests showed that the project objectives had been successfully achieved.

KEYWORDS: *Host-based IDS, Automation, Deployment,Computer Security, Network Security*

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

En este proyecto de titulación, se llevó a cabo una investigación sobre DevOps y las herramientas HIDS. Se examinaron múltiples fuentes y documentación para entender cómo funcionan estas herramientas, sus prestaciones y la mejor manera de integrarlas en un entorno de seguridad de red. Gracias al análisis se identificó Ansible como la herramienta DevOps adecuada y Tripwire como la solución HIDS para el proyecto.

Los servicios de seguridad de la red se implementaron utilizando Ansible. Se procedió a la instalación de Ansible en el equipo servidor y a la configuración de las conexiones SSH para la comunicación con el equipo cliente.

También se crearon dos *playbooks*: uno para instalar y configurar Tripwire y el otro para añadir reglas de supervisión de ficheros al archivo de políticas de Tripwire. Estos *playbooks* se diseñaron para automatizar el proceso de configuración y asegurar que las políticas de seguridad se aplican de manera uniforme en el sistema gestionado.

Con el fin de verificar el buen funcionamiento de los servicios de seguridad implementados, se han ejecutado *playbooks* y se han realizado comprobaciones del sistema de archivos. Se creó un archivo de prueba en el equipo cliente para verificar que Tripwire detectaba los cambios en el sistema de archivos. Los resultados mostraron que Tripwire supervisó correctamente los cambios, confirmando así que la implementación y la configuración se habían realizado correctamente.

Estas acciones aseguraron el cumplimiento de cada uno de los objetivos planteados para el proyecto, garantizando una implementación automatizada de soluciones de seguridad de red utilizando herramientas DevOps.

1.1 Objetivo general

Implementar servicios de seguridad en redes mediante DevOps.

1.2 Objetivos específicos

- Analizar la herramienta de DevOps que soluciona cada componente del proyecto de titulación.
- Diseñar la solución para cada servicio de seguridad en red mediante herramientas de DevOps.
- Implementar las soluciones mediante DevOps para el despliegue de los servicios de seguridad en red.

- Verificar el funcionamiento de cada servicio de seguridad en red implementado mediante DevOps.

1.3 Alcance

En el presente proyecto de titulación, se pretende que el estudiante aplique principios de DevOps con el fin de desplegar soluciones de seguridad informática, dicho proyecto consta de las siguientes partes que se describen a continuación.

- **Automatización del Despliegue:** Se pretende hacer uso de herramientas de DevOps para automatizar el despliegue de un *Host-based Intrusion detection System* (HIDS), además de crear *playbooks* que permitan la implementación automatizada de HIDS en hosts gestionados por ansible.
- **Implementación de un HIDS:** En este punto se tiene que seleccionar una solución HIDS por ejemplo (Tripwire, Ossec o Wazuh), además de la integración del HIDS por medio de la herramienta de DevOps que se escoja por medio de scripts o playbooks.
- **Pruebas de soluciones de seguridad:** Además de la realización de pruebas para verificar la correcta instalación de HIDS implementado.
- **Documentación y reportes:** Se debe generar una documentación que describa el proceso de implementación, configuración y pruebas del HIDS.

1.4 Marco Teórico

HIDS

Un HIDS es un sistema que va a permitir detectar a intrusos en un sistema, un HIDS se instala en equipos que requieran ser monitoreados de forma individual o grupal. El HIDS se tiene que instalar directamente en el equipo de computación que se necesite proteger, cuando el sistema de detección esta ya instalado y configurado esté procede a realizar un monitoreo continuo, lo que conlleva una supervisión de una manera periódica y en tiempo real de los eventos, archivos y procesos presentes en el dispositivo monitoreado [1].

Un HIDS se caracteriza por:

- Alertar a los equipos de seguridad o de TI sobre problemas de seguridad que puedan afectar al sistema, cada una de las alertas que se emitan deben tener una etiqueta que muestre el nivel de riesgo con el fin de que el personal de TI pueda saber si son alertas que requieran o no una atención inmediata. [2]

- Reportar sobre el estado de seguridad general del sistema, contienen los tipos de riesgos que están presentes en el sistema y la cantidad de los mismos. [2]
- Responder de forma automática con el fin de poder ayudar a subsanar riesgos que se han detectado, lo que permite ahorrar tiempo para los ingenieros y técnicos, en adición la automatización de respuestas permite bloquear riesgos de seguridad inmediatamente. [2]

Para que un HIDS pueda funcionar y emitir alertas tiene que contar con sensores que se conocen como *"HIDS-agents"*, dichos agentes se van a instalar en los equipos que se necesite monitorear, para poder cumplir su objetivo un HIDS va a utilizar dos métodos, uno basado en firmas y otros basados en anomalías. [3]

La detección basada en firmas básicamente analiza los paquetes de red con el fin de poder detectar características particulares de un ataque. Para poder lograr detectar estos comportamientos únicos en los paquetes en la detección basada en firmas se debe tener una base de datos con firmas de los ataques, que básicamente son patrones y características de amenazas conocidas.

Los paquetes que se reciben van a compararse con la base de datos con firmas de amenaza, si existe un patrón conocido en un paquete se lo marcara para poder en consecuencia hacerle frente dependiendo del nivel de riesgo. Para que este tipo de detección sea efectiva se tiene que actualizar con regularidad la base de datos, si no se actualiza de manera periódica hay la posibilidad de que se produzcan fallos por nuevas amenazas que aún no está presente en la base de datos. [4]

Por otra parte, la detección basada en anomalía va a emplear el aprendizaje de manera automática con el propósito de poder aplicar un modelo para el comportamiento normal de la red, en este tipo de detección se compara la actividad actual de un paquete con el modelo y va a mostrar las características que no concuerden. Aunque también tiene la característica de que pueda emitir falsos positivos, por ejemplo, cuando un usuario con los permisos necesarios para poder entrar al sistema para trabajar con datos sensibles ingresa esto podría emitir una alerta sobre el nuevo ingreso del usuario en este punto también es necesario etiquetar las alertas para poder determinar de forma correcta el nivel de seriedad de cada alerta [4] .

En el mundo del HIDS existen dos tipos:

- **HIDS basado en agentes:** Se basa en software de agentes, dicho software se instala en cada equipo que necesite monitoreo con el fin de poder recolectar

información, este tiene un enfoque conocido como *"heavier-weight"* por el hecho de que el uso de los recursos del host va a aumentar. [3]

- **HIDS sin agentes:** En este tipo de HIDS la información de cada uno de los hosts se va a recolectar sin necesidad de tener un agente instalado, este tipo de HIDS se considera uno de los más complejos de implementar, pero a comparación de los HIDS que se basan en software este consume menos recursos del host. [3]

Sin importar el tipo de HIDS que se desee desplegar, de forma general está conformado por tres partes:

- **El recopilador de datos:** Tanto los HIDS con agente como los sin agente utilizarán sensores para poder obtener datos de los hosts que están siendo monitoreados.
- **El Almacenamiento de datos:** Cuando los datos han sido recopilados, se proceden a guardar en una ubicación centralizada, los datos que se recopilen se mantendrán almacenados mientras se realiza el análisis de los mismos [3], luego se desechan, aunque se puede tener la opción de guardar estos datos para en un futuro poder hacer referencia a los mismos de ser necesario.
- **El Motor de análisis o motor analítico:** Es el encargado de procesar y evaluar las distintas fuentes de datos que se recolecten [3], tiene como objetivo buscar características y patrones que representen anomalías o riesgos de seguridad.

Ejemplos de soluciones HIDS

En el mercado existen soluciones de HIDS que permiten realizar el monitoreo de equipos en busca de riesgos y amenazas, aquí se presentan un par de ejemplos de los HIDS que se podría tomar en cuenta si se piensa en realizar la implementación dentro de una organización,

- **OSSEC HIDS:** Este es un HIDS de código abierto (open source) el cual permite monitorear un sistema en tiempo real, generando alertas ante amenazas cuando han sido detectadas. [5]
- **Wazuh:** Esta es una plataforma también de código abierto que se dedica principalmente a detectar, prevenir y responder cuando existen posibles amenazas. [6]
- **Tripwire:** Es una herramienta de monitorización de integridad de archivos (FIM) y de detección de intrusiones basada en host (HIDS). Su principal objetivo es el de detectar y alertar sobre los cambios no autorizados realizados en los sistemas

de archivos, en las configuraciones y en otros aspectos importantes de cualquier sistema, este sistema funciona de la siguiente manera:

- **Inicialización:** Tripwire genera una línea de base inicial de todos los ficheros y directorios especificados en el sistema. Esta referencia se almacena en un archivo cifrado para protegerlo contra la manipulación por malware. Se guardan las sumas de comprobación (hashes criptográficos) y otras características importantes como pueden ser los permisos, los cambios internos en los archivos y los detalles de la fecha y hora. [7]
- **Supervisión:** Continuamente o a determinados intervalos, Tripwire contrasta el estado actual de un sistema con la línea de base inicial [7]. Emplea algoritmos criptográficos con el fin de generar sumas de comprobación de los archivos actuales y las compara con las sumas de comprobación guardadas, sin que sea preciso guardar el contenido íntegro de los archivos que se encuentran en la base de datos.
- **Detección de cambios:** Si se detectan diferencias entre el estado inicial y el estado actual, Tripwire genera alertas. Dichas alertas pueden configurarse para notificar a los responsables del sistema cualquier cambio no autorizado, lo que contribuye a identificar posibles fallos de seguridad.
- **Generación de informes:** Tripwire crea informes detallados de los cambios detectados, lo cual facilita a los administradores revisar las modificaciones y adoptar las medidas necesarias para revertir los cambios no deseados o para investigar posibles fallos de seguridad. [7]

Tabla 1.1 Comparación entre Ossec y Tripwire

Característica	OSSEC HIDS	Tripwire HIDS
Licencia	*Open Source*	Versión *Open Source* y versiones comerciales
Monitoreo de integridad	Si	Si

Soporte para SIEM	Si (integración con varias soluciones SIEM)	Si (integración nativa con Tripwire Enterprise)
Alertas y notificaciones	Si (altamente configurable)	Sí (configurable y automatizable)
Informes	Si (informes detallados y configurables)	Si (informes avanzados y detallados)
Facilidad de configuración	Requiere conocimientos técnicos avanzados	Más fácil en la versión comercial
Soporte multiplataforma	Si (Windows, Linux, Unix, macOS)	Si (Windows, Linux, Unix, Solaris)
Automatización de respuestas	Limitada (depende de scripts personalizados)	Si (automatización avanzada en versiones comerciales)
Actualizaciones y Soporte	Comunidad (soporte limitado)	Comercial (soporte profesional y actualizaciones)
Escalabilidad	Adecuado para pequeñas y medianas empresas	Adecuado para medianas y grandes empresas
Integración con otras plataformas	Si (varias integraciones disponibles)	Si (amplia integración con herramientas de seguridad)

DevOps

DevOps consiste en una filosofía y un conjunto de prácticas culturales que permiten integrar los equipos de desarrollo de software (Dev) y de operaciones de TI (Ops) con el fin de mejorar la colaboración y la productividad mediante la automatización y la integración continua [8]. Esta manera de trabajar permite a los equipos ejercer sus labores conjuntamente a lo largo del ciclo de vida del programa, desde su desarrollo y comprobación hasta su implementación y mantenimiento.

El gran objetivo de DevOps es agilizar el tiempo de entrega de software de alta calidad, disminuir los errores y aumentar la eficiencia en la gestión de infraestructuras y despliegues [9]. Al automatizar los procesos repetitivos e implementar las prácticas de integración y entrega continuas, DevOps permite dar una respuesta más rápida a los

retos del mercado que se presentan actualmente y a las necesidades de los consumidores.

Para poder cumplir con su objetivo DevOps plantea una estructura a seguir la cual permite que se agilicen los procesos.

- **Planificación:** Se fijan los valores y necesidades empresariales para el desarrollo y explotación del software, fijando los objetivos y estrategias a seguir [9].
- **Codificación:** En esta nueva etapa se crea y diseña el software y se genera su código fuente.
- **Compilación:** Se gestionan las compilaciones y las versiones del software mediante herramientas automatizadas [9], con el fin de asegurar en todo momento la integridad del código y garantizar su disponibilidad para la realización de pruebas y su implementación.
- **Pruebas:** Se llevan a cabo pruebas para comprobar el funcionamiento del software en un entorno real, detectando posibles fallos [8].
- **Corrección de errores:** Si se identifican errores durante la fase de pruebas, se subsanan lo antes posible para asegurar la calidad del software.
- **Puesta en marcha:** Se procede a la gestión, coordinación, automatización y programación de las etapas de producción, así como a la configuración de la escalabilidad automática y seguridad de la plataforma [8].
- **Funcionamiento**: El software es administrado durante su etapa de producción, lo que permite que se asegura su disponibilidad y rendimiento [8].
- **Supervisión:** Se van a recolectar e identificar todos los resultados que se dieron durante el desarrollo y operación del software para posteriormente analizarlos, con el objetivo de mejorar continuamente los procesos [8].

2 METODOLOGÍA

Inicialmente, se llevó a cabo una investigación sobre DevOps y sobre softwares HIDS. Esta investigación fue esencial para comprender los fundamentos necesarios y proceder al desarrollo del trabajo de tesis. Durante esta fase, se revisaron diversas fuentes académicas y prácticas sobre DevOps y sistemas de detección de intrusiones basados en host (HIDS), lo que proporcionó una visión en detalle de cómo estas herramientas se podían integrar de manera adecuada en el proyecto.

Con una clara comprensión de la implementación de DevOps, se han diseñado dos *playbooks* concretos utilizando Ansible. El primero de ellos se creó con el fin de

automatizar la instalación de *Tripwire*, una herramienta de detección de intrusiones basada en host (HIDS). Este *playbook* incluyó todas las tareas necesarias para descargar, configurar e inicializar *Tripwire* en los sistemas de destino, garantizando una implementación estandarizada y repetible.

El segundo *playbook* fue desarrollado para aplicar nuevas reglas al documento de políticas de *Tripwire*, con el fin de monitorear archivos críticos adicionales. Este *playbook* permitió definir nuevas reglas para el monitoreo de directorios considerados importantes. Durante el diseño de estos *playbooks,* se consideraron las mejores prácticas en automatización y seguridad, garantizando que cada configuración y tarea contribuyera a una solución fiable.

Los dos *playbooks* se concibieron con la idea de ser dinámicos y adaptables, permitiendo ajustes según las necesidades del proyecto. Se hicieron verificaciones iniciales para validar el correcto funcionamiento de los *playbooks* y garantizar que las reglas establecidas precisas se apliquen de manera correcta. Esta fase de diseño y desarrollo fue fundamental para establecer una base sólida sobre la cual se cimentarían las siguientes etapas de implementación y verificación.

3 RESULTADOS

Se realiza la implementación de un HIDS conocido como *Tripwire*. La herramienta de DevOps Ansible permite instalar automáticamente la solución desde una máquina servidor a una máquina cliente. De igual manera, se implementan reglas adicionales al documento de políticas a través de este medio. Esto posibilita la adición remota de diversas reglas para monitorear los directorios críticos de un cliente.

La Tabla 1.1 Comparación entre Ossec y Tripwire compara dos softwares que, a consideración del autor, representan soluciones asequibles. Cada uno posee características particulares. En este proyecto se opta por *Tripwire*, ya que ofrece una solución de monitoreo de integridad de archivos con una amplia trayectoria. Esto brinda mayor confianza en la detección de cambios no autorizados en archivos críticos del sistema, lo cual es de alta importancia en entornos sensibles.

Cabe destacar que *Tripwire* es una solución probada y confiable. Este aspecto es fundamental al elegir una solución cuyas características de eficacia y escalabilidad estén respaldadas. Además, presenta informes detallados sobre los cambios realizados en el sistema, lo que resulta una buena opción para realizar auditorías, incluso sin contar con las capacidades adicionales de la versión comercial.

Otra ventaja de Tripwire reside en su carácter de código abierto. Esto implica contar con una comunidad activa y una base bien estructurada de recursos para solventar problemas e implementar mejores prácticas.

3.1 Analizar la herramienta de DevOps que soluciona cada componente del proyecto de titulación.

En este análisis, se examinarán dos herramientas clave en el ámbito de la integración y entrega continuas (CI/CD) y la gestión de configuraciones: Jenkins y Ansible. Estas herramientas han sido seleccionadas por su capacidad de automatizar tareas, mejorar la eficiencia y reducir errores en el proceso de desarrollo de software como también la administración de infraestructuras. A continuación, se detallan las características y beneficios de cada una de ellas.

Una de las herramientas que se ha seleccionado para analizar es *Jenkins* es un servidor de *open source,* diseñado para la integración continua (CI) y la entrega continua (CD) en los proyectos de desarrollo de software. La herramienta permite que los proyectos de software se compilen y prueben de manera continua, lo que facilita la integración de los cambios en un desarrollo y la entrega de versiones nuevas. Escrito en *Java*, *Jenkins* es una herramienta multiplataforma y se utiliza a través de una interfaz web, lo que la hace flexible y fácil de usar en varios entornos operativos [10].

Jenkins se usa generalmente para poder automatizar tareas repetitivas cuando se está desarrollando software, lo que permite que se mejore la eficiencia y reduciendo los errores que se puedan llegar a cometer, las características más notables de Jenkins son:

- **Integración continua y entrega continua:** *Jenkins* es capaz de funcionar tanto como un servidor de CI y también como el núcleo de un sistema de CD, administrando el proceso total de desarrollo de software.
- **Fácil instalación y configuración:** Al ser una aplicación autónoma basada en Java, Jenkins puede funcionar en múltiples sistemas operativos como Windows, Linux, macOS y otros sistemas Unix. La configuración es sencilla a través de su intuitiva interfaz web, que también proporciona detección de errores y ayuda integrada. [10].
- **Extensibilidad mediante *plugins*:** Jenkins cuenta con varios cientos de *plugins* que están disponibles en su *Update Center*, lo que hace posible la integración

con diversas tecnologías y herramientas que se aplican en el desarrollo y despliegue de software.

- **Arquitectura distribuida:** Gracias a su arquitectura distribuida, Jenkins es capaz de distribuir tareas entre varios equipos, lo que permite acelerar los procedimientos de compilación, prueba y despliegue. [10].
- **Compatibilidad y versatilidad:** Es compatible con diversos sistemas operativos y se ha adaptado recientemente para trabajar con los contenedores Docker, lo que aumenta su flexibilidad en distintos entornos de desarrollo. [10].

Por otro lado, la herramienta Ansible es una plataforma *open source* de gestión de configuraciones que va a permitir automatizar tareas en una gran cantidad de procesos informáticos, con esta plataforma existe la posibilidad de gestionar configuraciones, implementar aplicaciones, además de permitir la orquestación de sistemas. [11].

Esta plataforma a diferencia de otras soluciones de gestión de la configuración, no necesita de agentes que se encuentren instalados en los sistemas remotos, de tal manera que no requiere software específico para poder realizar los cambios y ejecutar comandos.

Ansible permite a los técnicos de DevOps gozar de algunos beneficios entre los cuales se pueden mencionar a continuación:

- Disminuir los recursos que se ocupan para gestión de equipos de TI.
- Facilita la automatización de los procesos al usar archivos YAML (*Yet Another Markup Language*), lo que ofrece la oportunidad de que los técnicos puedan entender estos archivos, dado que contienen algunos elementos de programación.
- Compatible con numerosas herramientas DevOps como Jira, GitHub, Selenium y Jenkins, lo que permite una integración fluida en diversos entornos de desarrollo y operaciones.

Ansible funciona de la siguiente manera:

- **Playbooks:** Ansible utiliza archivos YAML conocidos como *playbooks* para especificar tareas de automatización de forma minuciosa y estructurada. Dichos *playbooks* contienen una lista de tareas que se ejecutan en hosts gestionados y permiten describir el estado deseado en que se encuentra la infraestructura [11].
- **Arquitectura sin agente:** Ansible utiliza un modelo sin agentes, lo que quiere decir que no se requiere instalar software adicional para los nodos

gestionados. Para comunicarse con estos nodos utiliza SSH, lo que simplifica la administración y mejora la seguridad, ya que no hay ningún otro agente adicional que pueda presentar vulnerabilidades [12].

- **Módulos:** Ansible utiliza módulos, pequeños programas ejecutables destinados a ejecutar tareas concretas, como instalar software, configurar servicios y gestionar redes. Estos módulos se ejecutan en los nodos gestionados y envían los resultados de vuelta al controlador de Ansible [13].

En el contexto de DevOps, Ansible tiene un papel primordial en la integración y entrega continuas (CI/CD). Posibilita la automatización de los procesos repetitivos y susceptibles de errores y garantiza que las distintas aplicaciones se desplieguen de forma coherente en todos los entornos. Ansible ayuda a los equipos de DevOps a administrar sus infraestructuras como código, lo que permite la replicación de entornos y una rápida conmutación por error [14]. La posibilidad de Ansible de integrarse con otras herramientas DevOps, como Jenkins y Docker, amplía su usabilidad y permite automatizar los flujos de trabajo desde el desarrollo hasta la producción.

Otro punto fuerte de Ansible en el ámbito de DevOps es su sencillez de uso y su reducido coste de entrada. Al ser una herramienta sin agentes, disminuye la sobrecarga de la gestión de nodos y la dificultad de la configuración inicial. Asimismo, su sencilla sintaxis y su estructura de *playbook* YAML facilitan a los distintos departamentos su rápido uso sin necesidad de una curva de aprendizaje muy pronunciada. Todas estas características, junto con su flexibilidad y escalabilidad, convierten a Ansible en una valiosa solución para los equipos de DevOps que deseen aumentar la eficacia y fiabilidad de sus procesos de despliegue y gestión de infraestructuras [15].

A manera de comparación se presenta la siguiente tabla:

Tabla 3.1 Comparación entre Jenkins y Ansible

Aspecto	**Jenkins**	**Ansible**
Tipo de Herramienta	Servidor de entrega continua (CD) e integración continua (CI)	Plataforma de gestión de configuraciones y automatización
Licencia	*Open source*	*Open source*
Lenguaje de Programación	Java	Python
Instalación	Necesita instalarse en un servidor y puede requerir configuración en varios nodos	No necesita agentes instalados en los nodos gestionados; utiliza SSH para la comunicación

Configuración	Utiliza una interfaz web para configuración; los pipelines pueden ser complejos y requieren dedicación	Utiliza archivos YAML (*Playbooks*) para la configuración; fácil de entender y escribir
Arquitectura	Arquitectura basada en *plugins*; puede distribuir tareas a múltiples máquinas	Arquitectura sin agentes; utiliza módulos ejecutables para realizar tareas en nodos gestionados
Funcionalidades Principales	- Integración continua - Entrega continua - Automatización de pruebas -Notificaciones - Despliegue automatizado	-Gestión de configuraciones -Implementación de aplicaciones -Orquestación de sistemas - Automatización de tareas
Integración con Otras Herramientas	Compatible con herramientas DevOps como Git, GitHub, Docker, Slack, etc.	Compatible con herramientas DevOps como Jira, GitHub, Selenium, Jenkins, Docker, etc.
Manejo de Errores	Notifica al equipo si hay fallos en el build; facilita la detección y corrección de errores	Facilita la identificación y corrección de errores mediante *playbooks;* asegura despliegues coherentes en todos los entornos
Comunicación	Utiliza *plugins* para comunicarse con otras herramientas y servicios	Utiliza SSH para comunicarse con nodos gestionados; no requiere software adicional en los nodos
Escalabilidad	Altamente escalable mediante la adición de nodos esclavos para distribuir la carga	Altamente escalable gracias a su arquitectura sin agentes y la facilidad de duplicación de configuraciones
Ventajas	- Amplio soporte de plugins -Gran comunidad de usuarios -Automatiza despliegues y pruebas -Reduce el tiempo de desarrollo y entrega	- No requiere agentes - Fácil de usar y configurar. -Flexible y escalable. -Reduce la sobrecarga de gestión.
Desventajas	-Interfaz de usuario anticuada- Pipelines pueden ser complejos y requieren tiempo	- Puede tener una curva de aprendizaje inicial para nuevos usuarios

	-Necesita servidor de alojamiento y configuraciones técnicas	-La documentación puede ser limitada en algunas áreas

Tras analizar, se seleccionó Ansible para el proyecto por varias razones clave, entre las que destacan su eficacia y facilidad para poder gestionar la configuración y la automatización de tareas. A diferencia de otras soluciones, Ansible no requiere la instalación de agentes en sistemas remotos, lo que simplifica el despliegue y reduce la sobrecarga administrativa. Utiliza archivos YAML, conocidos como Playbooks, que son fáciles de entender y permiten a los ingenieros DevOps especificar tareas de automatización de forma estructurada [11]. La arquitectura que utiliza SSH para comunicarse con los nodos gestionados, mejora la seguridad y la gestión [12].

3.2 Diseñar la solución para cada servicio de seguridad en red mediante herramientas de DevOps.

Este proyecto se encuentra enfocado en la implementación automatizada de servicios de seguridad de red utilizando herramientas DevOps, específicamente Ansible. Un elemento esencial de la seguridad de red es la detección de intrusiones. Para abordar esto, se ha optado por implementar un sistema de detección de intrusiones basado en host (HIDS), específicamente la herramienta Tripwire.

Tripwire es una solución HIDS que permite supervisar y detectar cambios en archivos y carpetas críticos de un sistema. Con Tripwire, se puede garantizar la integridad del sistema detectando cambios no autorizados que puedan indicar intrusiones o actividades maliciosas.

Para implementar Tripwire de manera eficiente y reproducible, se usa Ansible, una herramienta de automatización que permite gestionar configuraciones y despliegues a gran escala sin la necesidad de agentes en los nodos gestionados. La elección de Ansible se debe a su facilidad de uso, su capacidad para automatizar tareas repetitivas y su compatibilidad con una amplia gama de sistemas.

El proceso de implementación de Tripwire mediante Ansible se describe a continuación:

- **Instalación de Ansible:** Antes de implementar Tripwire, se instala Ansible en el sistema de control. Ansible es elegido por su simplicidad en la configuración y su capacidad para ejecutar *playbooks* que describen el estado deseado del sistema gestionado.

- **Instalación de Tripwire:** Ansible se utiliza para automatizar la instalación de Tripwire en máquinas remotas. Esto se logra creando un *playbook* de Ansible que ejecuta comandos para instalar Tripwire.
- **Configuración de Tripwire:** Una vez instalado, Tripwire se configurará. Esto incluye la definición de archivos y carpetas críticos a monitorizar. Todo ello se realiza mediante un *playbook* de configuración.
- **Inicialización y ejecución de Tripwire:** Después de la configuración, Tripwire se inicializa. Ansible ejecuta los comandos necesarios presentes en el *playbook* de instalación del HIDS para esta inicialización.

Al utilizar Ansible, no solo se asegura una implementación consistente y reproducible de Tripwire en múltiples sistemas, sino que también se facilita la gestión continua y el despliegue de actualizaciones y configuraciones adicionales, lo que mejora significativamente la eficiencia operativa y la seguridad general de la red.

3.3 Implementar las soluciones mediante DevOps para el despliegue de los servicios de seguridad en red.

El tercer objetivo, se ha realizado plenamente mediante la instalación de Ansible, además, el desarrollo de dos *playbooks* de Ansible diseñados para automatizar la instalación y configuración de Tripwire, un sistema de detección de intrusiones en host (HIDS). El primer *playbook* se encarga de la instalación de Tripwire y sus dependencias en la máquina cliente, garantizando que el proceso se realiza de manera uniforme y repetible en cada entorno, a continuación, se explica el proceso de instalación de Ansible en la maquina servidor.

La instalación de Ansible es importante ya que para poder comunicarse con la maquina remota, ya que Ansible usa conexión por SSH para poder ejecutar comandos y *playbooks* sin la necesidad e intermediarios como agentes; para poder instalarlo y configurarlo se realizaron los siguientes pasos:

Se actualiza los paquetes del sistema y se instalan las dependencias necesarias como se muestra en la Figura 3.1 en la Figura 3.2:

- *apt-get update*

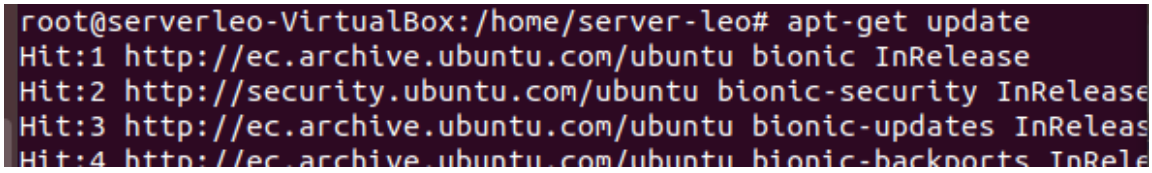

Figura 3.1 Comando apt-get update

- *apt-get install lsb-release software-properties-common*

```
root@serverleo-VirtualBox:/home/server-leo# apt-get install lsb-release softwar
e-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
lsb-release is already the newest version (9.20170808ubuntu1).
lsb-release set to manually installed.
The following packages were automatically installed and are no longer required:
  fonts-liberation2 fonts-opensymbol gir1.2-goa-1.0
```

Figura 3.2 Instalación de las dependencias.

- Después se tiene que configurar el repositorio Ansible y se actualizarán los paquetes del sistema, en máquinas Debian/Ubuntu se utilizara el repositorio PPA como se puede evidenciar en la Figura 3.3.
    - *apt-add-repository –y ppa:ansible/ansible*

```
root@serverleo-VirtualBox:/home/server-leo# apt-add-repository -y ppa:ansible/a
nsible
Hit:1 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:2 http://ec.archive.ubuntu.com/ubuntu bionic InRelease
Get:3 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic InRelease [15,9 kB
]
Hit:4 http://ec.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:5 http://ec.archive.ubuntu.com/ubuntu bionic-backports InRelease
Get:6 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic/main i386 Packages
 [704 B]
Get:7 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic/main amd64 Package
s [704 B]
Get:8 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic/main Translation-e
n [472 B]
Fetched 17,8 kB in 6s (3.197 B/s)
```

Figura 3.3 Agregación de repositorio de Ansible.

    - *apt-get update*
- Luego se tiene que instalar Ansible mediante el siguiente comando también se lo indica en la Figura 3.4.
    - *apt-get install Ansible*

```
root@serverleo-VirtualBox:/home/server-leo# apt install ansible
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  fonts-liberation2 fonts-opensymbol gir1.2-goa-1.0
  gir1.2-gst-plugins-base-1.0 gir1.2-gstreamer-1.0 gir1.2-gudev-1.0
  gir1.2-snapd-1 gir1.2-udisks-2.0 grilo-plugins-0.3-base gstreamer1.0-gtk3
  libboost-date-time1.65.1 libboost-filesystem1.65.1 libboost-iostreams1.65.1
  libboost-locale1.65.1 libcdr-0.1-1 libclucene-contribs1v5
  libclucene-core1v5 libcmis-0.5-5v5 libcolamd2 libdazzle-1.0-0
```

Figura 3.4 Instalación de Ansible.

Como ya se ha mencionado antes el software Ansible no tiene agentes y se conecta por medio de SSH para poder comunicarse con las maquinas remotas para poder conectarse se debe configurar la conexión remota, para eso se tiene que generar el par

de claves de autenticación, para que se logre conectar el servidor a la maquina remota sin necesidad de digitar una clave, para poder cumplir con esa función se tiene que seguir los siguientes pasos:

- Primero se tiene que cambiar a root y después hay que navegar hasta el directorio $HOME con los comandos que se indican a continuación y que a manera de muestra se indica en la Figura 3.5.
    - *cd ~*

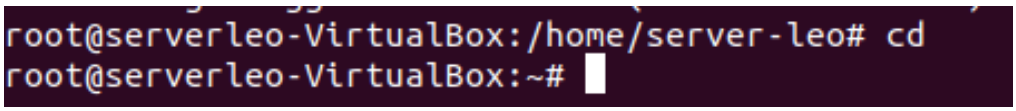

Figura 3.5 Moverse a la carpeta raíz.

- Se debe generar un par de claves de autentificación para el SSH, si se desea se puede incluir una frase, pero si no se da enter(vacío) y se aceptan las rutas donde se guardarán el par de claves y ya se habrán creado, y para hacer eso se lo realiza con el siguiente comando.
    - *ssh-keygen*

Cuando se envía este comando indicara en la pantalla que las claves se están generando, se aceptan las rutas y se crea la clave con una frase o con un enter(vacío) el resultado debe ser el como el de la Figura 3.6 se puede observar la creación del par de claves :

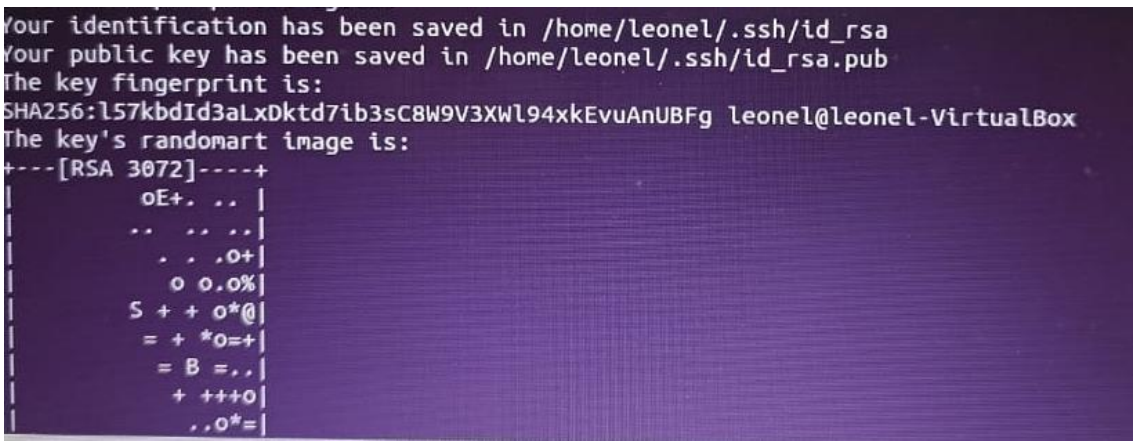

Figura 3.6 Generación del par de claves ssh.

- Una vez que las claves estén generadas se tiene que comprobar los permisos de las claves generadas ingresando el siguiente comando:
    - *ls –la ~/.ssh*

Se debe buscar el archivo que tiene por nombre “id_rsa” y se realiza la comprobación para asegurarse que tiene los permisos de lectura y escritura (r, w) , de la forma que se muestra en la Figura 3.7.

```
root@Server-Leo:~# ls -la ~/.ssh
total 24
drwx------ 2 root root 4096 Jun  4 15:46 .
drwx------ 9 root root 4096 Jun 11 20:34 ..
-rw------- 1 root root    0 Jun  2 12:50 authorized_keys
-rw------- 1 root root 2602 Jun  2 18:53 id_rsa
-rw-r--r-- 1 root root  569 Jun  2 18:53 id_rsa.pub
-rw------- 1 root root  506 Jun  7 01:18 known_hosts
-rw-r--r-- 1 root root  142 Jun  2 18:58 known_hosts.old
```

Figura 3.7 Revisión de permisos en id_rsa.

- Después que se revise que el archivo id_rsa tiene los permisos correctos, se tiene que otorgar permisos totales o 700 (permisos totales para el dueño “drwx”) al directorio “/root/.ssh/ “, mediante el siguiente comando:
    - *chmod 700 ~/.ssh*

  Se tiene que comprobar que el comando anterior si funciono y se aplicaron, los permisos correctos como se indica en la Figura 3.8.

```
root@Server-Leo:~# ls -la
total 2544
drwx------  9 root root    4096 Jun 11 20:34 .
drwxr-xr-x 19 root root    4096 Jun 17 07:48 ..
drwxr-xr-x  6 root root    4096 Jun 10 00:14 .ansible
-rw-------  1 root root   19571 Jun 18 20:08 .bash_history
-rw-r--r--  1 root root    3106 Oct 15  2021 .bashrc
drwx------  4 root root    4096 Jun 12 19:46 .cache
drwx------  3 root root    4096 Jun  2 18:49 .launchpadlib
-rw-------  1 root root      20 Jun  4 15:36 .lesshst
drwxr-xr-x  3 root root    4096 Jun  2 13:36 .local
-rw-------  1 root root     420 Jun  2 19:45 .mysql_history
-rw-r--r--  1 root root     161 Jul  9  2019 .profile
drwx------  2 root root    4096 Jun  4 15:46 .ssh
-rw-r--r--  1 root root       0 Jun  2 13:45 .sudo_as_admin_successful
-rw-------  1 root root     532 Jun 11 20:34 .viminfo
-rw-r--r--  1 root root     245 Jun  3 19:14 .wget-hsts
-rw-r--r--  1 root root 2518737 Jun  2 19:30 3.7.0.tar.gz
drwxrwxr-x  8 root root    4096 Jun  2 19:46 ossec-hids-3.7.0
drwx------  4 root root    4096 Jun  2 13:26 snap
```

Figura 3.8 Revisión de permisos en el direcctorio /root/.ssh.

- Luego se tiene que instalar e iniciar open-ssh-server en la maquina cliente por medio de los siguientes comandos:
    - *apt-get install openssh-server*
    - *systemctl start sshd*

  Después si los pasos se realizaron correctamente se tiene que comprobar el estado mediante el comando *systemctl status ssh* y debe arrojar el resultado la Figura 3.9.

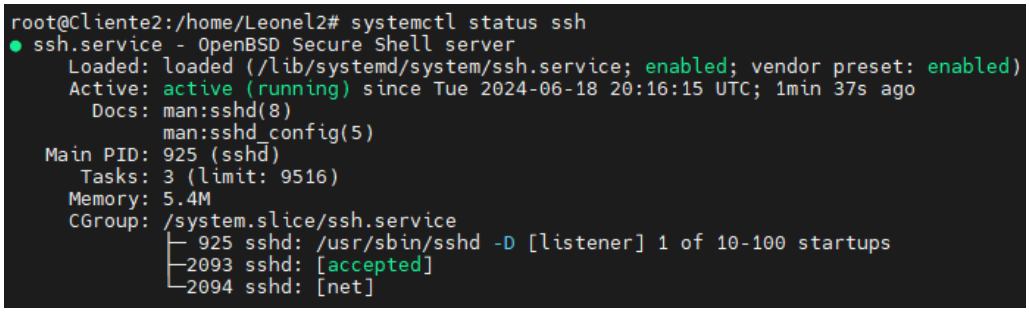

Figura 3.9 Revisión del estado de ssh.

- Como siguiente paso en la maquina cliente, hay que dirigirse al directorio $HOME, crear el directorio .ssh, y asignar los permisos totales 700 mediante haciendo uso de los siguientes comandos:
    - *cd ~*
    - *mkadir .ssh*
    - *chmod 700 .ssh/*
- Después se tiene que crear el archivo *authorized_keys* en el directorio .ssh/ que se creó anteriormente y se otorgaran los permisos necesarios 644(lectura escritura y ejecución):
    - *touch .ssh/authorized_keys*
    - *chmod 644 .ssh/autohorized_keys*

Se comprueba que se otorgaron, los permisos de manera correcta como se indica en la Figura 3.10,mediante el comando *ls -la .ssh.*

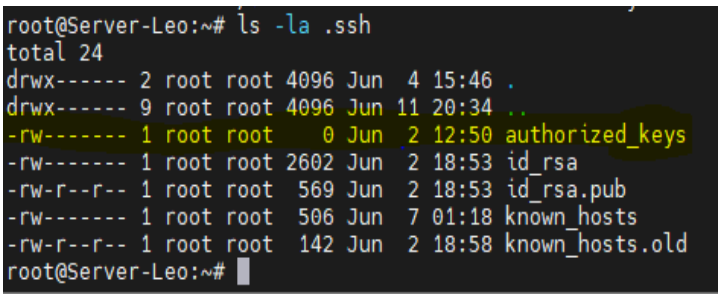

Figura 3.10 Revisión de permisos en archivo authorized_keys

- Luego se tiene que regresar a la maquina servidor y se copia la clave pública que se generó en pasos anteriores al directorio de la maquina cliente al archivo *authorized_keys* que está en la ruta ~/.ssh/authorized_keys, por medio del siguiente comando (hay que tener en cuenta el nombre del usuario de la maquina cliente y la dirección IP):

- *cat ~/.ssh/id_rsa.pub | ssh Leonel2@10.0.0.5 "cat >> ~/.ssh/authorized_keys"*

- Como siguiente paso hay que verificar en la maquina cliente que el archivo “sshd_config” que se encuentra en la ruta “/etc/ssh/sshd_config” permita realizar la autenticación de la clave pública, para eso se ingresa mediante nano a la ruta del archivo, una vez en dentro se comprueba que las líneas que se presentan en la Figura 3.11, PubkeyAuthenticatio yes y AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2 no se encuentren comentadas.

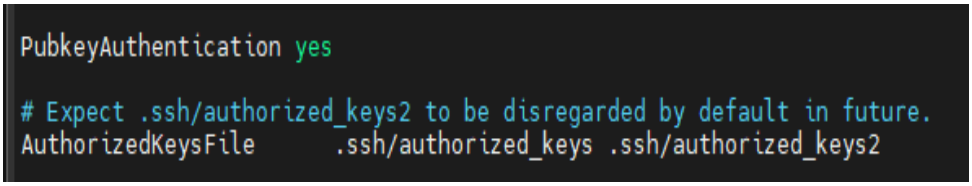

Figura 3.11 Líneas des comentadas del archivo sshd_config.

- Se tiene que reiniciar el servicio de SSH haciendo uso del siguiente comando:
  - *systemctl restar sshd*
- Después se comprueba la conexión ssh al cliente, si todo está correctamente no debe pedir ninguna clave de autenticación como en la Figura 3.12 y debe ingresar de forma automática.

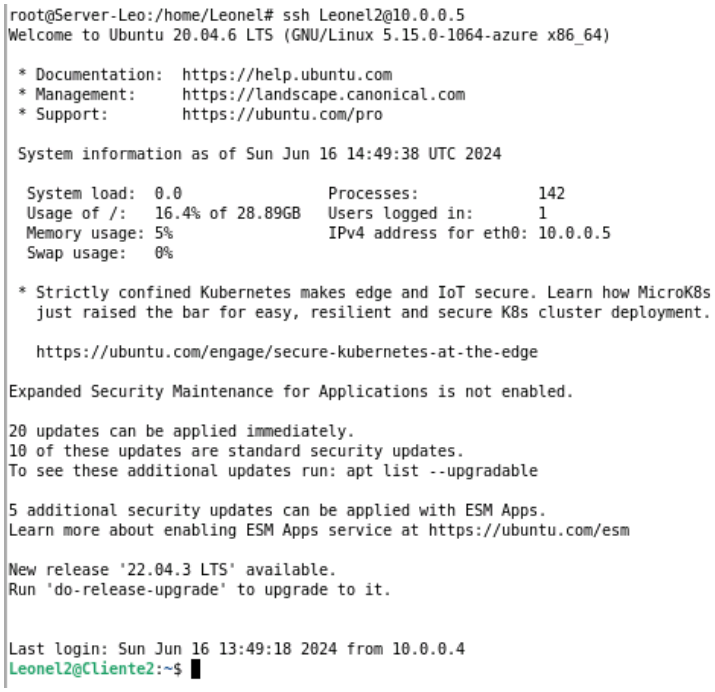

Figura 3.12 Conexión Automática de ssh(servidor-cliente).

- Luego en la maquina servidor se tiene que agregar los puntos finales para la gestión de Ansible mediante la entrada que se presenta en la Figura 3.13, en el documento hosts de Ansible que se encuentra en la ruta “/etc/ansible/hosts” que se puede observar en la Figura 3.14.

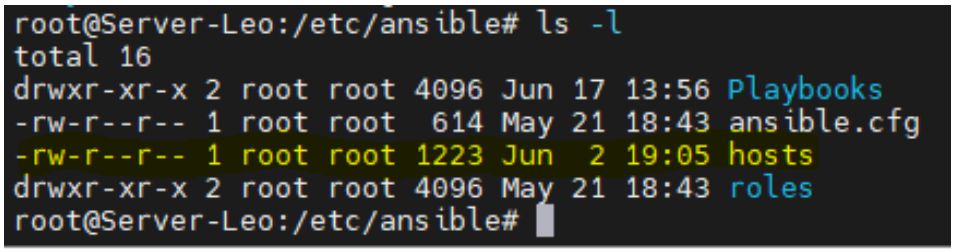

Figura 3.13 Documento de hosts de Ansible.

```
[all_in_one]
10.0.0.5  ansible ssh user=Leonel2
```

Figura 3.14 Inserción de usuario gestionado en archivo hosts

- Después de agregar el punto final al archivo de hosts se podrá hacer una comprobación de la conexión de Ansible con la maquina remota por medio del comando que se indica a continuación, el resultado debe ser el de la Figura 3.15**¡Error! No se encuentra el origen de la referencia.**:
    - *ansible all -m ping*

```
root@Server-Leo:/home/Leonel# ansible all -m ping
10.0.0.5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Figura 3.15 Prueba de conexión de ansible a máquina cliente

El segundo *playbook* se centra en la configuración avanzada de Tripwire mediante la inserción de reglas específicas en el archivo de políticas. Esto permite personalizar las políticas de supervisión y detección de intrusos y garantizar la supervisión de los archivos y carpetas que se monitorea. La automatización de estas tareas no sólo facilita la implementación inicial, sino que también garantiza que cualquier actualización de la política de seguridad se aplique de forma homogénea a todos los sistemas supervisados. Por lo tanto, se ha logrado una integración eficiente de prácticas de

automatización y gestión de configuración, cumpliendo cabalmente con el objetivo propuesto.

Partes del primer playbook:

```
---
#Instalación de Tripwire
 - hosts: 10.0.0.5 # En esta línea se especifica el host al cual se va a enviar las tareas.
   become: yes # El Become va a permitir que las tareas se ejecuten en modo root con el fin de no tener errores
   collections:
     - ansible.posix
```

Figura 3.16 Inicio del playbook InstalarTripwire.yml

En este primer fragmento Figura 3.16 del playbook, se está especificando el *host* y algunos parámetros importantes para ejecución de las tareas o *tasks*, en la línea "hosts:10.0.0.5", se está indicando que las tareas siguientes están dirigidas a ejecutarse en la maquina cuya dirección IP es 10.0.0.5 que corresponde a la maquina cliente, luego la línea que dice "become: yes*"* está indicando que las tareas que se van a ejecutar en el cliente tengan los permisos de root con el fin de que las tareas siguientes se ejecuten de una forma correcta.

Después se encuentra la sección *collections*, esta línea se refiere a las colecciones de módulos que se van a ocupar dentro del *playbook*, luego se encuentra la línea "ansible.posix" es una colección de módulo para manejar tareas comunes en sistemas Posix (*Portable Operating System Interface*), como la gestión de los usuarios y el manejo de archivos [16].

```
tasks: #Estas son las tareas a ejecutar

- name: Actualizar los paquetes de SO #name me permite poner un nombre referencial en la tarea.
  apt: #Con este modulo apt se puede actualizar paquetes tambien permite que instale o elimine paquetes
    update_cache: yes # Esta línea dentro del modulo apt me deja actualizar los paquetes del SO , funciona como apt-get update

- name: Instalar Tripwire en  maquina remota #Nombre de la tarea
  apt: #Modulo para instalar paquetes
    name: tripwire  # nombre de la aplicacion
    state: present # Esta línea me indica que el paquete de tripwire esta presente en el sistema si no lo esta lo instalara

- name: Configurar Postfix #Este es el nombr de la tarea
  debconf: #Este es el maódulo de Ansible se ocupa para configurar opciones de paquetes Debian antes de que sean instalados
    name: postfix #nombre del paquete que se desea configurar.
    question: "postfix/main_mail_type" # con question se automatiza la pregunta del tipo de instalación que se requeire
    value: "Intenet Site" #es esencialmente la respuesta a la pregunta especificada en question
    vtype: string # Es el tipo de dato que se envia en la respuesta que es es cadena de texto
```

Figura 3.17 Actualizar paquetes del sistema, Instalar Tripwire, Configurar Postfix

En la siguiente parte Figura 3.17 se inicia con la palabra "tasks" la cual se refiere al inicio de las tareas que se van a ejecutar en el cliente, después se encuentra la línea "name" esta parte de la tarea se usa para poder asignar un nombre para hacer saber de qué trata la tarea , luego se encuentra el módulo "apt" usa para poder gestionar paquetes en sistemas que se basen en Debian [16], dado que la maquina cliente es una maquina Ubuntu no tiene problemas con dicho módulo, también se encuentra la línea "update_cache" esta línea se usa para poder actualizar los paquetes funciona igual que poner apt-get update [18].

En la siguiente tarea de la Figura 3.17 que lleva por nombre Instalar Tripwire en máquina remota usa el módulo apt para poder instalar el paquete de Tripwire, luego se encuentra “name: tripwire” para indicar el nombre del paquete que se va a instalar, en la siguiente línea “state: present” se asegura que el paquete Tripwire esté presente en el sistema si no lo está lo procede a instalar [18].

La tarea que tiene como nombre Configurar Postfix, configura el paquete Postfix antes de su instalación. El módulo “debconf” permite preconfigurar opciones específicas. “question” especifica la pregunta de configuración, “value” proporciona la respuesta que en la mayoría de tutoriales de instalación de Tripwire es Internet, y “vtype” define el tipo de dato de la respuesta, que en este caso es una cadena de texto [19].

```
- name: Iniciar la base de datos de Tripwire
  ansible.builtin.expect: #Módulo que ayuda a manejar comandos que requieren de una tarea.
    command: tripwire --init #comando que se desea correr tripwire --init para correr la base de datos
    responses: #perimite que coloque una pregunta que requere de interacción
      'Please enter your local passphrase:': '' # pregunta por defecto que sale cuando se mande el comando anterior
  args: # esta línea me permite agregar argumentos extras
    timeout: 600 #le da un tiempo para que la tarea se culmine ya que la tarea si tarda un poco en ejecutarse

- name: Creacion de el directorio  no-directory.txt
  file: #Me permite gestionar ficheros en el sistema
    path: /tmp/tripwire #path se usa para colocar la ruta de donde se creara el archivo
    state: directory  # me dice que lo que se creara es un directorio si no existe se creara

- name: Copiar el archivo de twpol.txt (servidor-cliente)
  copy: #modulo para copiar un archivo desde mi maquina servidor al cliente
    src: /home/Leonel/Desktop/twpol.txt # me dice la fuente osea el origen del archivo
    dest : /etc/tripwire/twpol.txt # me dice el destino donde se pondra el archivo qyue se quiere copiar
    owner: root #es el dueño del archivo y en este caso es el usuario root
    group: root #dice que donde se copiara el archibo tambien tiene permisos de root
    mode: "0644" # el modo me deja darle permisos al archivo 6 lectura y escritura ,4 permisos para el grupo , 4 permisos para otros
```

Figura 3.18 Iniciar base de datos de Tripwire, Creación del directorio, Copiar políticas

En la tarea de la Figura 3.18 , que tiene por nombre Iniciar la base de los datos de Tripwire, el módulo “ansible.builti.expect” el cual funciona para poder manejar comandos que necesitan de una interacción.

Como lo que se busca es que sea automática la instalación este módulo ayuda a cumplir ese fin , después con “command” se puede enviar comandos en este caso el comando tripwire –init dicho comando inicia la base de datos pero después de correrlo tanto de forma manual como automática pide una contraseña , dicha contraseña en este caso no existe porque se ha hecho por defecto , más bien se tiene que colocar enter para poder continuar , para automatizar la entrada de enter se colocara “responses” para poder a continuación ingresar la pregunta correspondiente después de ejecutar el comando, además se usó “args” para poder agregar un timeout ya que el comando de iniciar la base de datos tanto si se lo envía de forma manual como automática se tarda hasta poder iniciar y correr la base de datos [20].

Después en la tarea que lleva por nombre Creacion del directorio no directory.txt en la Figura 3.18 se ha de crear el archivo no directory.txt dado que si este no está creado

cuando se inicie la base de datos se producirá un error, esta tarea crea un directorio en la ruta especificada utilizando el módulo file. El parámetro “path” define la ubicación del directorio por medio de una ruta, y “state: directory” indica que se debe crear un directorio si aún no se encuentra creado [21] [22].

En la tarea que tiene por nombre Copiar el archivo de twpol.sh también de la Figura 3.18, se utiliza el módulo “copy” para transferir un archivo desde la máquina local al cliente, “src” define la ubicación del archivo fuente, y “dest” especifica la ubicación de destino ósea la máquina cliente, “owner” y “group” establecen los permisos de propietario y grupo como root, respectivamente, mientras que “mode” va a definir los permisos del archivo los cuales son de lectura y escritura, permisos para el grupo y otros [23].

```
- name: Regenerar la Politica de Tripwire
  ansible.builtin.expect:
    command: twadmin -m P /etc/tripwire/twpol.txt #command hace que corra el comando twadmin para p
    responses:
        'Please enter your site passphrase:': ''

- name: Reiniciar la base de datos de Tripwire
  ansible.builtin.expect:
    command: tripwire --init
    responses:
        'Please enter your local passphrase:': ''
  args:
    timeout: 600

- name: Agregar la linea para recibir email en root
  lineinfile: # se encara de ver si esta o no la linea que se quiere agragar si no esta la agrega
    path: /etc/crontab #esta es la ruta del archivo donde se colocara la linea siguiente
    line: "0 0 * * * root tripwire --check --email-report" #Linea que se debe agregar para programa

- name: Reiniciar cron
  systemd: #este módulo me permite gestionar archivos del sistema como lo es el crontab
    name: cron #servicio de cron el cual se va a reinicial
    state: restarted #indica que el servico de cron se debe reiniciar
```

Figura 3.19 Regenerar política de Tripwire, Reiniciar base de datos, Agregación de línea al archivo crontab.

Como siguiente, en la tarea que tiene como nombre Regenerar la política de Tripwire, el módulo “ansible.builtin.expect” en la Figura 3.19, se emplea para correr el comando “twadmin -m P /etc/tripwire/twpol.txt”, el cual recarga el archivo de políticas de Tripwire. El módulo “responses” automáticamente responde a la solicitud de contraseña del sitio, lo que permite la ejecución sin necesidad de intervención manual [20], luego de esta tarea se tiene que reiniciar la base de datos de Tripwire nuevamente, para resaltar esta es una de las tareas repetitivas que se automatizo.

Después sigue la tarea de agregar la línea en el archivo crontab que está en la ruta /etc/crontab, dicha tarea usa el módulo “lineinfile” este se utiliza para asegurarse de que una línea específica esté presente en el archivo /etc/crontab, esta línea es impórtate ya que sin ella no se podrá recibir los reportes en el mail de root . La línea 0 0 * * * root tripwire --check --email-report programa una tarea cron para ejecutar un chequeo de

Tripwire y enviar un reporte por email a medianoche, se comprueba si en el archivo la línea existe en caso contrario de la agrega.

Para finalizar con la explicación de playbook encargado de instalar Tripwire (HIDS), en esta tarea se emplea el módulo “systemd” de la para gestionar el servicio cron. El parámetro “name” indica el servicio cron y “state: restarted” señala que el servicio se debe reiniciar con el fin de aplicar las modificaciones en el archivo crontab [24] .

Partes del segundo *playbook*:

El segundo *playbook* se centra en la configuración de Tripwire mediante la adición de reglas específicas al documento de políticas de la herramienta.

Este *playbook* permite personalizar las reglas de monitoreo y detección de intrusiones, lo que maximiza la efectividad de Tripwire al detectar actividades sospechosas y posibles intrusiones. Al automatizar la configuración de estas reglas, se asegura que todos los sistemas monitorizados sigan las mismas políticas de seguridad, garantizando así una protección coherente, este segundo *playbook* se encuentra formado de las siguientes partes.

```
---
- hosts: 10.0.0.5
  become: yes
  collections: #Colección de modulos a utilizar.
    - ansible.posix #Sirve para manejar tareas comunes en sistemas POSIX
```

Figura 3.20 Primera parte del playbook reglas.yml

En esta primera parte como en el anterior playbook se tiene que colocar los hosts que en los que se van a ejecutar las tareas, y colocar el “become” con el fin de que las tareas que se van a ejecutar en la maquina remota se realicen con permisos de root , como se indica en Figura 3.20.

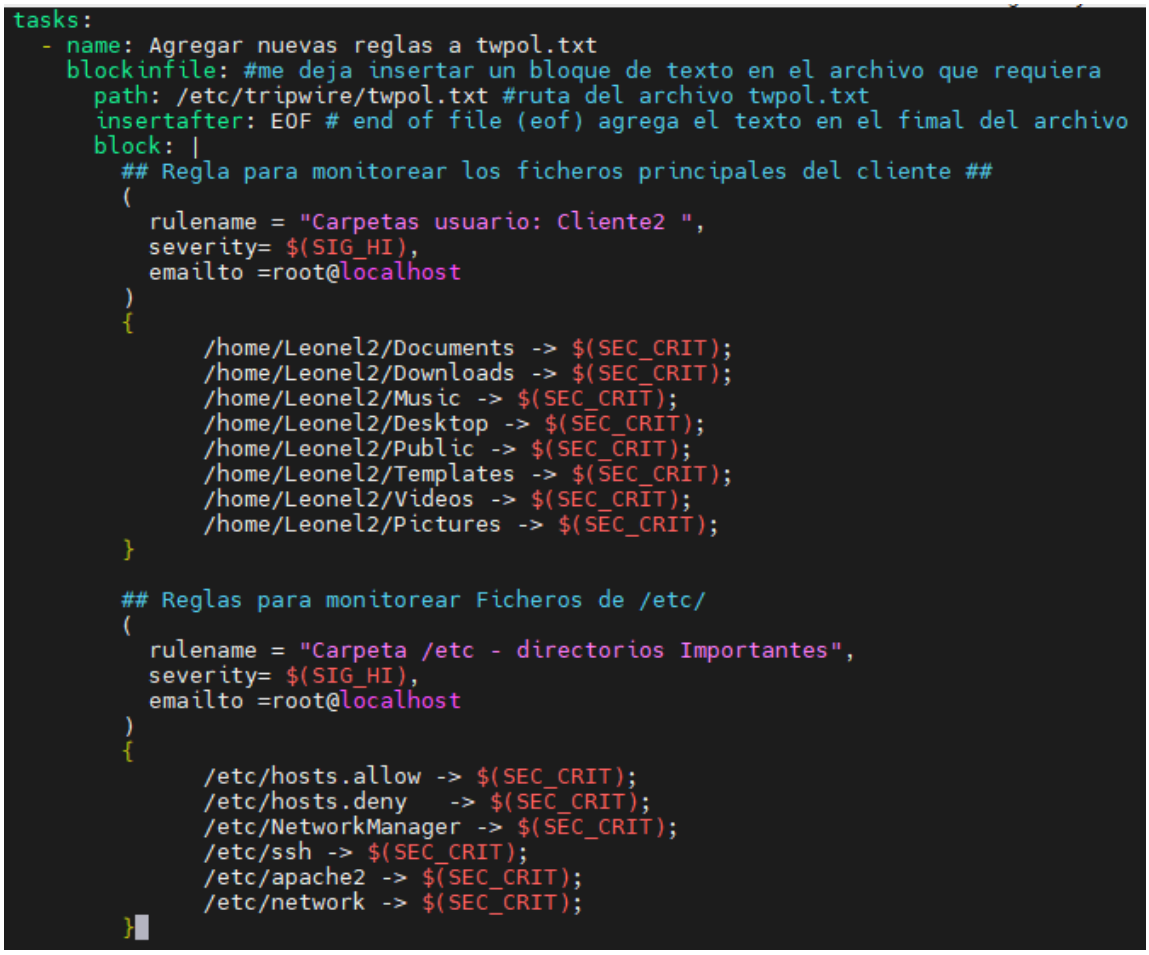

Figura 3.21 Agregar nuevas reglas al twpol.txt

En el bloque de código proporcionado de la Figura 3.21, la tarea se define como "Añadir nuevas reglas a twpol.txt". Esta tarea utiliza el módulo "blockinfile" de Ansible para insertar un bloque de texto en el archivo de configuración de políticas de Tripwire ubicado en /etc/tripwire/twpol.txt. La ruta de la línea: /etc/tripwire/twpol.txt indica la ruta del archivo que se va a modificar. La directiva "insertafter: EOF" establece que el bloque de texto debe insertarse al final del archivo, es decir, a continuación de la última línea existente (EOF, *End Of File*). Por último, el módulo "block: |" introduce el bloque de texto a insertar, permitiendo insertar varias líneas de texto simultáneamente dentro del fichero especificado. Este modo de configuración es útil para actualizar automáticamente las políticas de monitorización y garantizar que las nuevas reglas se aplican correctamente al final del archivo de configuración existente [25].

Este playbook cumple la función de agregar reglas nuevas para poder monitorear ficheros que a consideración del autor son importantes para el servicio de HIDS, los ficheros en cuestión son los archivos de usuario y archivos del sistema.

En los archivos del usuario, la monitorización de las carpetas de usuario es fundamental para detectar cambios no autorizados en documentos personales, descargas, música, escritorio y otras carpetas de importancia. De este modo, se puede detectar posibles fallos de seguridad.

En los archivos del sistema, las carpetas dentro de /etc/ contienen ajustes fundamentales del sistema. Controlando estos archivos, se puede descubrir cambios no autorizados que puedan comprometer la seguridad del sistema. Por ejemplo, los cambios en hosts.allow y hosts.deny pueden modificar la política de acceso del sistema.

```
- name: Regenerar la Política de Tripwire
  ansible.builtin.expect:
    command: twadmin -m P /etc/tripwire/twpol.txt
    responses:
      'Please enter your site passphrase:': ''

- name: Reiniciar la base de datos de Tripwire
  ansible.builtin.expect:
    command: tripwire --init
    responses:
      'Please enter your local passphrase:': ''
  args:
    timeout: 600

- name: Verificar si la base de datos se ha creado correctamente
  stat: #me va a dar informacion del archivo
    path: /var/lib/tripwire/Cliente2.twd
  register: db_status #Permite registrar el contenido de la variable que se coloco como db_status

- name: Mostrar el estado de la base de datos
  debug: #Muestra lo que son mensajes de depuración
    msg: "Estado de la base de datos: {{ db_status.stat.exists }}" #muestra un mensaje predeterminado
```

Figura 3.22 Regenerar la política de Tripwire, Reiniciar base de datos, Verificación de base de datos, Mostrar el estado de la base de datos.

La tarea de Regenerar la política de Tripwire de la Figura 3.22 es la misma que se utilizó en el playbook para instalar el HIDS lo mismo pasa con la tarea de reiniciar la base de datos, lo que certifica que con la creación de playbooks se pueden automatizar las tareas repetitivas en el sistema.

En la tarea llamada," Verificar si la base de datos de ha creado correctamente" de la Figura 3.22, la línea "- name:" comprueba si la base de datos se ha creado correctamente. Por otro lado, el módulo "stat" comprueba si ya existe el fichero de base de datos Tripwire Client2.twd, que se ubica en /var/lib/tripwire/. El módulo "stat" recaba información sobre el archivo especificado y el producto de esta operación se almacena en una variable llamada db_status. Esta variable almacena detalles sobre el estado del fichero, tales como su existencia, tamaño y permisos, de forma que se pueda comprobar que la base de datos se ha creado correctamente [26].

La línea "name:" Mostrar el estado de la base de datos utiliza el módulo "debug" para imprimir un mensaje en la salida de Ansible que indica si la base de datos existe. El mensaje, utilizando sintaxis Jinja2, accede a la variable "db_status" y muestra el valor

del atributo existe. Este valor será TRUE si el archivo de la base de datos existe y FALSE si no existe.

Estos playbooks no solo demuestran el cumplimiento del objetivo de diseñar soluciones de seguridad mediante DevOps, sino que también reflejan una integración eficiente de prácticas de automatización y gestión de configuración.

3.4 Verificar el funcionamiento de cada servicio de seguridad en red implementado mediante DevOps.

Para verificar el funcionamiento efectivo de los servicios de seguridad en red implementados mediante DevOps, se ha hecho correr los dos playbooks Figura 3.23 , Figura 3.24 ,con el fin de ver el resultado además se ha creado un nuevo archivo Figura 3.25 en el cliente con el fin de que en el monitoreo se plasme el cambio hecho mediante el chequeo de los ficheros Figura 3.26.

- Figura 3.23: Muestra la ejecución del playbook InstalarTripwire.yml, el cual instala Tripwire en la máquina remota, actualiza los paquetes del sistema operativo, configura Postfix, inicia y reinicia la base de datos de Tripwire, y realiza otras tareas necesarias para la correcta configuración de Tripwire.
- Figura 3.24: Muestra la ejecución del playbook reglas.yml, el cual agrega nuevas reglas a twpol.txt, regenera la política de Tripwire, reinicia la base de datos de Tripwire, y verifica que la base de datos se ha creado correctamente.

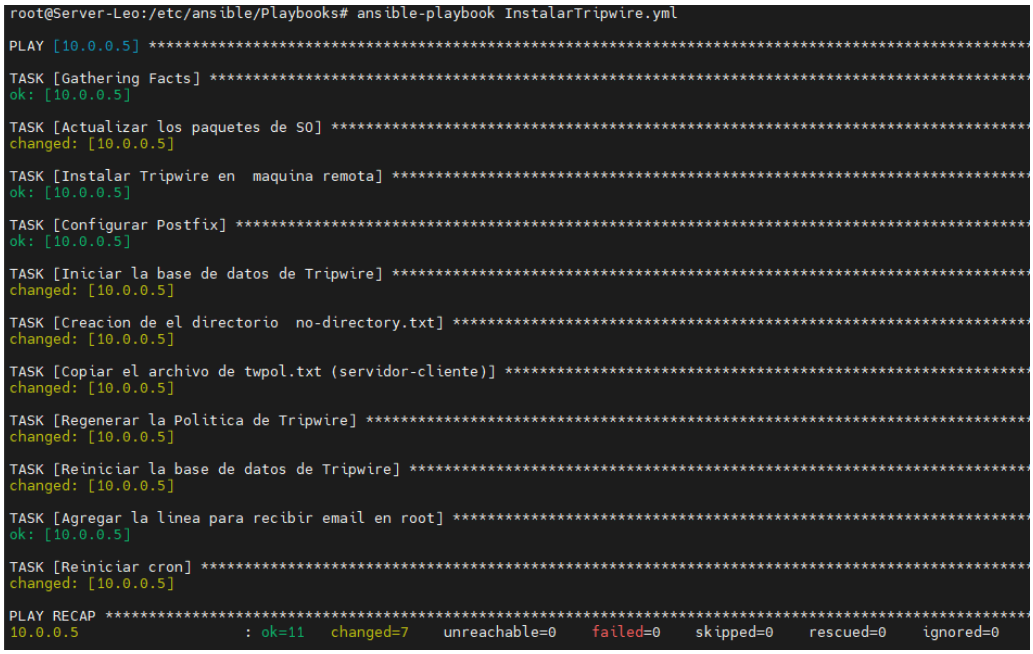

Figura 3.23 Resultado del playbook InstalarTripwire.yml

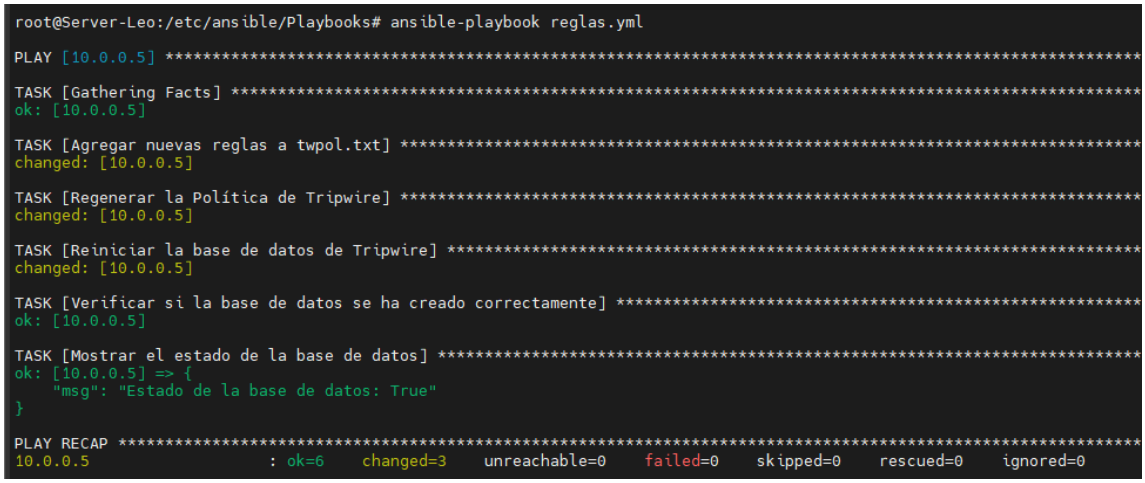

Figura 3.24 Resultado del Playbook reglas.yml

- Figura 3.25: Muestra la creación de un archivo denominado Archivopormonitorear.yml en el cliente. Este archivo se creó para verificar que Tripwire puede detectar cambios en el sistema de archivos.

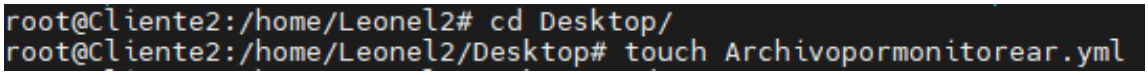

Figura 3.25 Creación de Archivo de prueba

- Figura 3.26: Muestra el resultado del comando tripwire --check, el cual realiza un chequeo completo del sistema de archivos y muestra cualquier cambio detectado. En este caso, se puede evidenciar que en el informe del HIDS se muestra la modificación realizada, confirmando que Tripwire está monitoreando correctamente los cambios en el sistema de archivos.

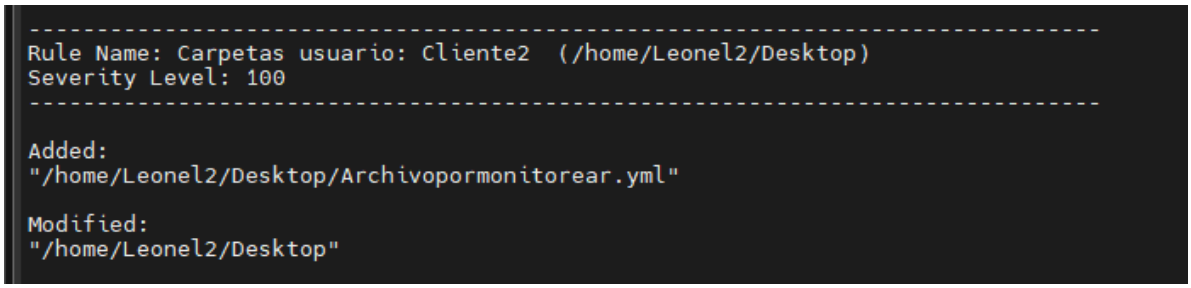

Figura 3.26 Comprobación de monitoreo del Archivo de prueba por Tripwire

Además, para poder confirmar que Tripwire se instaló correctamente en el cliente se presenta a continuación en el uso del comando tripwire -- version confirmado la versión de Tripwire que está instalado en la maquina cliente.

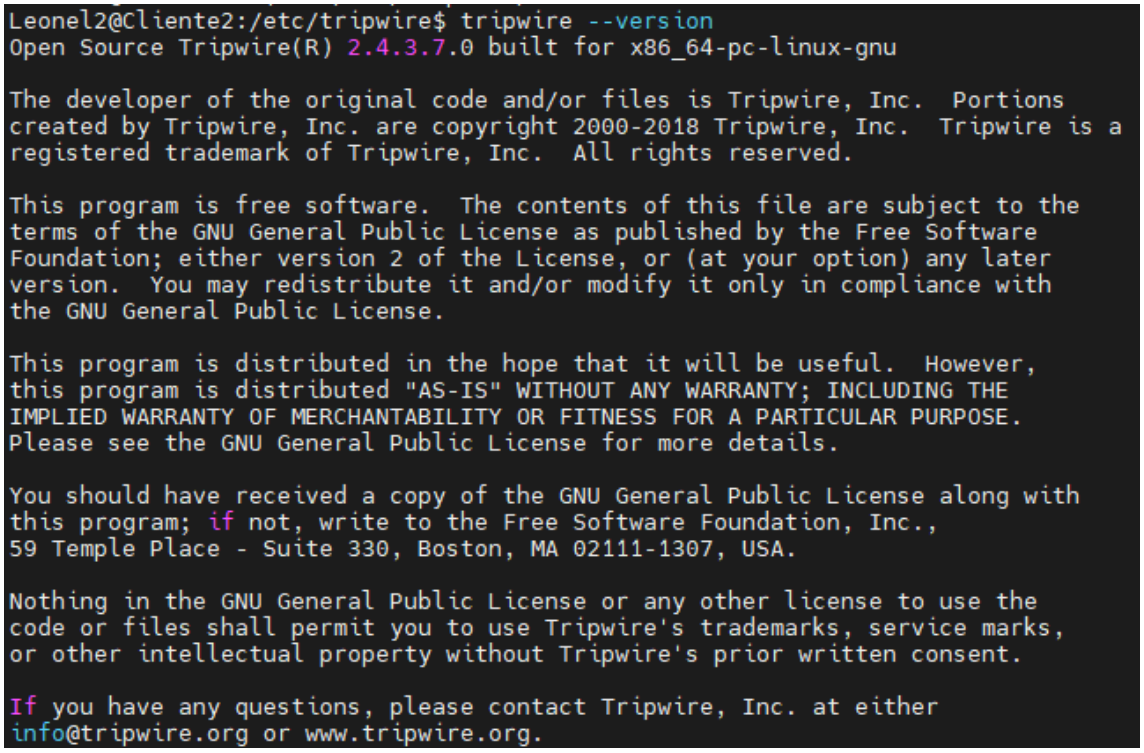

Figura 3.27 Confirmación de versión instalada en cliente

4 CONCLUSIONES

- El análisis detallado de Jenkins y Ansible ha aportado un conocimiento de sus capacidades en la integración continua, entrega continua y también la gestión de la configuración. Ansible aporta una solución eficaz y segura en la gestión de la configuración sin agentes en nodos gestionados. Gracias a este análisis se ha validado la elección de Ansible como herramienta central para el despliegue de HIDS.
- El diseño y la implementación automatizados del sistema de detección de intrusiones basado en host (HIDS) mediante Ansible han demostrado factible para mejorar la seguridad de la red. La capacidad de Ansible para gestionar configuraciones de forma coherente y reproducible en distintos entornos ha facilitado la configuración de Tripwire, asegurando la detección de cambios no autorizados.
- La exitosa implementación de Tripwire mediante Ansible ha reforzado la postura de seguridad de la red al permitir una gestión de los directorios y ficheros del sistema. Ansible ha facilitado la instalación, configuración de Tripwire sin la necesidad de agentes en los nodos gestionados, lo que mejora la fiabilidad y garantiza la detección de intrusiones si existen cambios en los archivos del sistema.

- Se realizó una verificación del funcionamiento de los servicios de seguridad de red implementados a través de DevOps, específicamente utilizando Tripwire mediante Ansible. La correcta ejecución de los playbooks "InstallTripwire.yml" y "rules.yml" aseguró la instalación y configuración adecuadas de Tripwire en el nodo remoto.
- Además, se creó un archivo de prueba en el cliente para monitorear los cambios en el sistema de archivos, confirmando así la capacidad de Tripwire para detectar modificaciones no autorizadas.

5 RECOMENDACIONES

- A manera de recomendación se estipula que se utilice el módulo de Ansible que permite la interacción para dar respuestas a ciertas preguntas que se pueden realizar dicho módulo se conoce como “ansible.building.expect” al momento de estar instalando algún sistema por medio de Ansible donde se requiera de una interacción.
- También se tiene que tener en cuenta que Ansible trabaja mediante el uso de una conexión SSH para poder comunicarse con los *hosts*, para que Ansible pueda comunicarse de manera adecuada a los *hosts* en el documento de *hosts* que está en la ruta /etc/ansible/hosts se deben encontrar plasmado lo que es la IP y el *username* al cual se quiere conectar.
- No se tiene que confundir el tipo de usuario cuando se va a realizar el “ansible all -m ping” ya que este comando únicamente va a funcionar en el usuario root.
- Si llegase a ocurrir algún error al momento de enviar el comando ansible all -m ping se tiene que ver el error que comúnmente suele ser que la carpeta /tmp, se encuentra llena, hay que vaciar la carpeta y enviar el comando mencionado anteriormente.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] «fastercapital.com,» [En línea]. Available: https://fastercapital.com/es/tema/%C2%BFqu%C3%A9-es-el-monitoreo-continuo-y-por-qu%C3%A9-es-importante.html. [Último acceso: 26 mayo 2024].

[2] «sysdig.com,» [En línea]. Available: https://sysdig.com/learn-cloud-native/detection-and-response/what-is-hids/. [Último acceso: 26 Mayo 2024].

[3] Redscan, «redscan.com,» [En línea]. Available: https://www.redscan.com/services/hids/. [Último acceso: 26 Mayo 2024].

[4] IBM, «ibm.com,» [En línea]. Available: https://www.ibm.com/mx-es/topics/intrusion-detection-system. [Último acceso: 26 Mayo 2024].

[5] C. Cilleruelo, «Keepcoding.io,» 7 12 2022. [En línea]. Available: https://keepcoding.io/blog/que-es-ossec/. [Último acceso: 26 Mayo 2024].

[6] «servicepilot.com,» [En línea]. Available: https://www.servicepilot.com/es/integration/monitoreo-wazuh/. [Último acceso: 26 Mayo 2024].

[7] Kaushik Sen, «upguard.com,» 17 Octubre 2021. [En línea]. Available: https://www.upguard.com/blog/tripwire-open-source-vs-ossec-which-is-right-for-you. [Último acceso: 03 Junio 2024].

[8] «aws.amazon.com,» Amazon Web Services, Inc., [En línea]. Available: https://aws.amazon.com/es/devops/what-is-devops/. [Último acceso: 26 Mayo 2024].

[9] hiberus, «hiberus.com,» [En línea]. Available: https://www.hiberus.com/crecemos-contigo/que-es-devops-y-por-que-apostar-por-esta-filosofia/. [Último acceso: 26 Mayo 2024].

[10] Bambu Editorial, «bambu-mobile.com,» 12 Octubre 2022. [En línea]. Available: https://bambu-mobile.com/para-que-sirve-jenkins/. [Último acceso: 24 Junio 2024].

[11] «redhat.com,» 21 Junio 2022. [En línea]. Available: https://www.redhat.com/es/topics/automation/learning-ansible-tutorial. [Último acceso: 27 Mayo 2024].

[12] Oracle, «oracle.com,» 13 Febrero 2024. [En línea]. Available: https://docs.oracle.com/es-ww/iaas/Content/ContEng/Tasks/contengconnectingworkernodesusingssh.htm. [Último acceso: 3 Junio 2024].

[13] Redhat, «redhat.com,» 01 Agosto 2023. [En línea]. Available: https://www.redhat.com/es/topics/automation/what-is-an-ansible-playbook. [Último acceso: 03 junio 2024].

[14] M. Mannambeth, «kodekloud.com,» 21 Septiembre 2021. [En línea]. Available: https://kodekloud.com/blog/ansible-in-devops/. [Último acceso: 13 Junio 2024].

[15] S. Manjaly, «blog.invgate,» 24 Marzo 2023. [En línea]. Available: https://blog.invgate.com/es/ansible. [Último acceso: 13 Junio 2024].

[16] «docs.ansible.com,» 13 Junio 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/ansible/posix/index.html. [Último acceso: 15 Junio 2024].

[17] «docs.ansible.com,» 27 Mayo 2022. [En línea]. Available: https://docs.ansible.com/ansible/2.9/modules/apt_module.html. [Último acceso: 2024 Junio 13].

[18] «docs.ansible.com,» 13 Junio 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_module.html. [Último acceso: 13 Junio 2024].

[19] «docs.ansible.com,» 27 Mayo 2024. [En línea]. Available: https://docs.ansible.com/ansible/2.9/modules/debconf_module.html. [Último acceso: 2024 Junio 13].

[20] «/docs.ansible.com,» 13 Junio 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/ansible/builtin/expect_module.html. [Último acceso: 13 Junio 2024].

[21] «/docs.ansible.com,» 30 Abril 2021. [En línea]. Available: https://docs.ansible.com/ansible/2.8/modules/file_module.html#file-module. [Último acceso: 13 Junio 2024].

[22] «https://www.howtoforge.com/,» [En línea]. Available: https://www.howtoforge.com/tutorial/how-to-monitor-and-detect-modified-files-using-tripwire-on-ubuntu-1604/. [Último acceso: 13 Junio 2024].

[23] «docs.ansible.com,» 13 Diciembre 2020. [En línea]. Available: https://docs.ansible.com/ansible/2.7/modules/copy_module.html. [Último acceso: 13 Junio 2024].

[24] «docs.ansible.com,» 27 Mayo 2022. [En línea]. Available: https://docs.ansible.com/ansible/2.9/modules/systemd_module.html. [Último acceso: 13 Junio 2024].

[25] 13 Junio 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/ansible/builtin/blockinfile_module.html. [Último acceso: 13 Junio 2024].

[26] «docs.ansible.com,» 27 Mayo 2022. [En línea]. Available: https://docs.ansible.com/ansible/2.9/modules/stat_module.html. [Último acceso: 13 Junio 2024].

7 ANEXOS

ANEXO I: Certificado de Originalidad

CERTIFICADO DE ORIGINALIDAD

Quito, D.M. XX de XXXX de 2022

De mi consideración:

Yo, FERNANDO VINICIO BECERRA CAMACHO, en calidad de Directora del Trabajo de Integración Curricular titulado IMPLEMENTACION DE UN HIDS MEDIANTE HERRAMIENTAS DE DEVOPS asociado al IMPLEMENTACIÓN DE SERVICIOS DE SEGURIDAD EN REDES MEDIANTE DEVOPS elaborado por la estudiante JEREMY LEONEL CAMPOVERDE ARMIJOS de la carrera en Tecnología Superior en Redes y Telecomunicaciones, certifico que he empleado la herramienta Turnitin para la revisión de originalidad del documento escrito completo, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del xxx%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el link del informe generado por la herramienta Turnitin.

LINK

Atentamente,

FERNANDO VINICIO BECERRA CAMACHO

Docente

Escuela de Formación de Tecnólogos

ANEXO II: Enlaces

Anexo II.I ***Código QR de video explicativo de funcionamiento de HIDS***

ANEXO III: Códigos Fuente

Las listas de los Anexos se muestran a continuación:

ANEXO I. Playbook para instalar Tripwire.

```
#Instalación de tripwire
    - hosts: 10.0.0.5
      become: yes
      collections:
      -ansible.posix
tasks:
- name: Actualizar los paquetes de SO
  apt:
    update_cache: yes
- name: Instalar Tripwire en maquina remota
  apt:
    name: tripwire
    state: present
- name: Configurar Postfix
  debconf
    name: postfix
    question: "postfix/main_mail_type"
    value: "Internet Site"
    vtype: string
- name: Iniciar la base de datos de Tripwire
  ansible.builtin.expect:
```

```
    command: tripwire --init
    responses
      'Please enter your local passphrase:':
    args:
    timeout: 600
- name: Creación del directorio no-directory.txt
  file:
    path: /tmp/tripwire
    state: directory
- name: Copiar el archivo de twpol.txt (servidor-cliente)
  copy:
    src: /home/Leonel/Desktop/twpol.txt
    dest: /etc/tripwire/twpol.txt
    owner: root
    group: root
    mode: "0644"
- name: Regenerar la política de tripwire
  ansible.builting.expect
    command: twadmin –m /etc/tripwire/twpol.txt
    responses:
          “Please enter your site local passphrase”: “”
- name: Reiniciar la base de datos de Tripwire
  ansible.builtin.expect:
    command: tripwire --init
    responses
      'Please enter your local passphrase:' : “”
    args:
```

```
    timeout: 600
- name: Agregar la línea para recibir el email en root
  lineinfile:
    path: /etc/crontab
    line: “0 0 * * * root tripwire –check –email-report”
   - name: Reiniciar cron
     systemd:
       name: cron
       state: restarted
```

ANEXO II. Play book reglas.yml

```
#
     -    hosts: 10.0.0.5
          become: yes
          collections:
          -ansible.posix
tasks:
     -    name: Agregar nuevas reglas a twpol.txt
          blockinfile:
            path : /etc/tripwire/twpol.txt
            insertafter: EOF
           block: |
          (
            rulename = "Carpetas usuario: Cliente2”,
            severity= $(SIG_HI),
            emailto = root@localhost
           )
                 {
```

```
                    /home/Leonel2/Documents        -> $(SEC_CRIT);
                    /home/Leonel2/Downloads        -> $(SEC_CRIT);
                    /home/Leonel2/Music        -> $(SEC_CRIT);

                   /home/Leonel2/Desktop      -> $(SEC_CRIT);
                   /home/Leonel2/Public        -> $(SEC_CRIT);
                   /home/Leonel2/Templates       -> $(SEC_CRIT);
                   /home/Leonel2/Videos       -> $(SEC_CRIT);
                    /home/Leonel2/Pictures       -> $(SEC_CRIT);
                }
# Reglas para monitorear ficheros de /etc/
        (
          rulename = "Carpeta /etc – directories importantes",
          severity= $(SIG_HI),
          emailto = root@localhost
         )
              {
                   /etc/hosts.allow        -> $(SEC_CRIT);
                   /etc/hosts.deny       -> $(SEC_CRIT);
                   /etc/NetworkManager        -> $(SEC_CRIT);
                   /etc/ssh      -> $(SEC_CRIT);
                   /etc/apache2    -> $(SEC_CRIT);
                   /etc/network      -> $(SEC_CRIT);
              }
        - name: Regenerar la política de tripwire
  ansible.builting.expect
    command: twadmin –m /etc/tripwire/twpol.txt
```

```
    responses:
          “Please enter your site local passphrase”: “”
- name: Reiniciar la base de datos de Tripwire
  ansible.builtin.expect:
    command: tripwire --init
    responses
      'Please enter your local passphrase:' : “”
    args:
    timeout: 600
- name: Reiniciar la base de datos de Tripwire
  stat:
     path: /var/lib/tripwire/Cliente2.twd
register: db_status
- name: Mostrar el estado de la base de datos
  dbug:
      msg: “Estado de la base de datos: {{db_status.stat.exist}}”
```